Final Project
Vitals Beaconing System

EE 475 Fall 2017
Denis Jivaikin
Sandy Lee
Jiayou Zhao

TABLE OF CONTENTS

ABSTRACT

In the real world application of wireless information exchange, devices transfer data through Bluetooth Low Energy (BLE) channels. It is very common for the systems to establish a Bluetooth connection through advertisement broadcasting before proceeding with the information transfer. This lab explores the idea of transfering data through the advertisement messages, which originally were intended to notify the surrounding devices of your presence. Around this idea, Vitals Beaconing System (heart rate, temperature) is implemented through various hardware peripherals. The success of this project will be dependent on a series of tests at each stage of development. The initial tests examine the hardware functionality of the sensors and the development board. The second set of tests included basic functionality of the BLE advertisement and scanning on both the development board and the on-board computer. The final round of testing included the functionality of the final prototype of the Vitals Monitoring system in conjunction with a node that resends the advertisements through to prolong the distance of the message transfer. The final result of this project is an inexpensive way to wirelessly monitor the vitals of multiple people in a vicinity of a house or a hospital. With such general design, further improvements can be made with inclusion of additional sensors and features.

INTRODUCTION

Vitals Beaconing System (VBS) offers the healthcare professionals and home nurses an inexpensive and wireless way of monitoring vitals of multiple people in close (20-100 meter) proximity. VBS is capable of measuring heart rate, temperature, and acceleration of a person wearing the device. This information is sent over the BLE advertising channels with a purpose to distribute such information to all relevant devices that are listening. A node device is also available to prolong the distance of the message by 20 more meters. The goal of this project is to develop a prototype of a VBS and its additional node. The system is attached to a pair of glasses and wakes up when the device is picked up. In order to achieve this, a Nordic Semiconductor NRF52832 breakout board is used with a heart rate sensor, infrared temperature sensor, and an accelerometer connected. The chip is programmed using the C programming language. After the data is collected, parsed, and sent, it can be read by an on-board server that reads the advertisement data and displays it accordingly on a website hosted on the local Wifi network. The project was designed following a full development cycle, including UML and Gantt chart for planning. The initial tests examine the hardware functionality of the sensors and the development board. The second set of tests included basic functionality of the BLE advertisement and scanning on both the development board and the on-board computer. The final round of testing included the functionality of the final prototype of the Vitals Monitoring system in conjunction with a node that resends the advertisements through to prolong the distance of the message transfer. The final result of this project is an inexpensive way to wirelessly monitor the vitals of multiple people in a vicinity of a house or a hospital. In the future design changes improvements can be made with inclusion of additional sensors and features.

DISCUSSION OF THE LAB

Design Specification

System Description

This specification describes and defines the design requirements and specifications for a Bluetooth Low Energy (BLE) Vitals Beaconning System. The beacon is able to collect Vital Signs from a human user, filter the sensor data, and encode and transmit them over the BLE.

The system supports measurements of 2 vital signals but is designed in a way such that additional sensors supporting I2C protocol and operable below 5 V can be added with ease. The beacon is short range and battery operated with an option to add any additional nodes, that can transmit the data for longer distances. The beacon is able to receive and send data to all nearby BLE reading devices and create its own sharing network.

Specification of External Environment

The Vitals Beacon is to operate in a standard lab, outdoor, or household environment. This unit can be used on any person without any limitations.
This unit will support external battery operation.
Specific details are included under Operating Specifications.

System Input and Output Specification

System Inputs

- Temperature data from user: 34 to 40 ℃ ± 0.5 ℃ with 0.02 ℃ resolution
- Heart rate data from user: 0 to 200 bps
- Vitals sent from another user: data encoded and sent via BLE Beacon from another monitor

System Outputs

- BLE Signal output: the monitor will encode the following data for sending over the BLE Beacon network
    - Data
        - Encoded temperature data: 34 to 40 ℃ ± 0.5 ℃
        - Encoded heart rate data: 0 to 200 bps ± 2bps
        - 47 bytes - Standard BLE Advertising data bus
            - 28 bytes - Advertisement partition
                - 20 bytes - Beacon ID
                    - Used for transmitting data
    - Distance
        - 0-100 m
    - Signal Speed
        - 125 kbit/s - 2 Mbit/s
    - Latency

        - 6 ms
    - Power Consumption
        - 0.01-0.50 W
- UI output: The system will display the following data on a laptop application with all data presented in an easily readable manner
    - Encoded temperature data: 34 to 40 ℃ ± 0.5 ℃
    - Encoded heart rate data: 0 to 200 bps ± 2bps

UML Use Case Diagrams

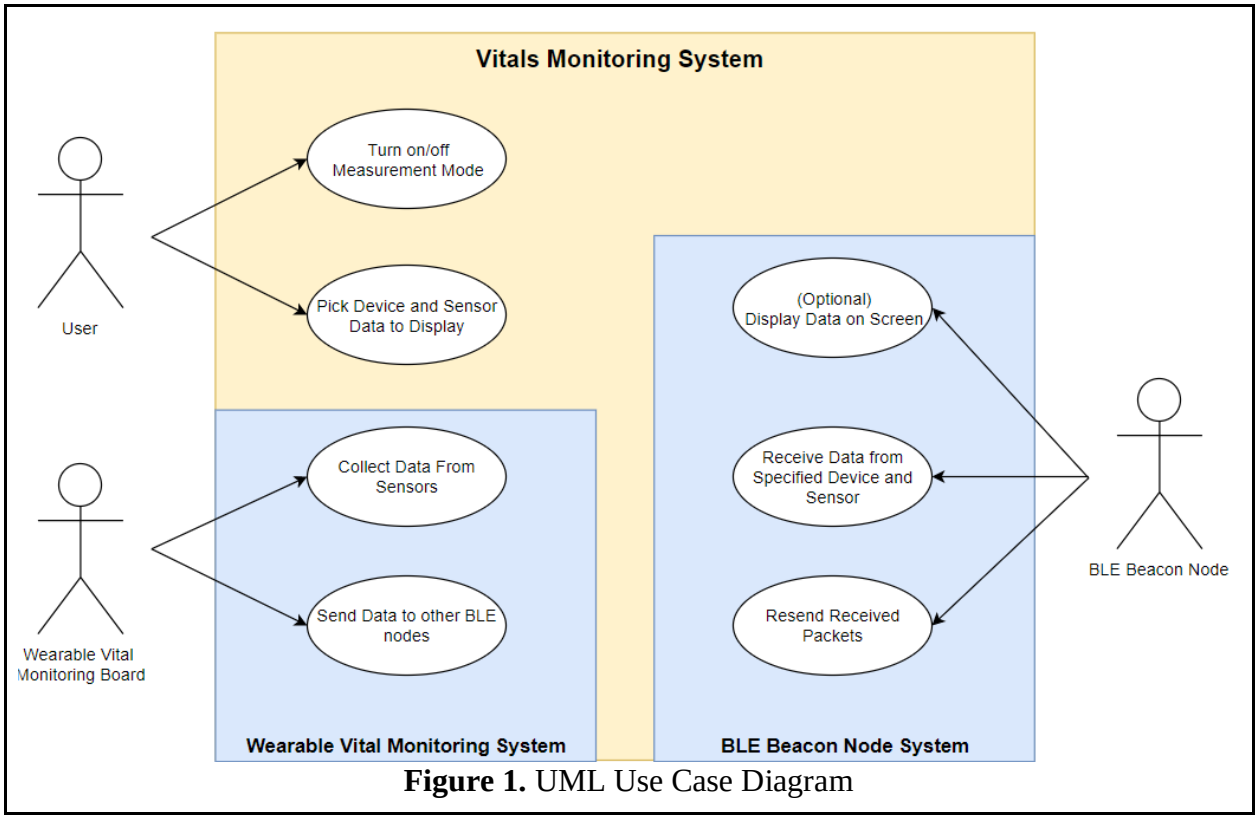


Figure 1. UML Use Case Diagram

Turn on/off Measurement Mode

➔ The user is able to turn on or turn off the device.

➔ Exception: the system is already on, no power.

Pick Device and Sensor Data to Display

➔ The user is able to pick which device to receive the data from and which or how many sensor reading to display.

➔ Exception: all the devices are off, device specified doesn't exist, out of memory.

Collect Data From Sensors

➔ The wearable vitals monitoring board is able to collect a multitude of sensor data.

➔ Exceptions: sensors disconnected, power off.

Send Data to other BLE nodes

- ➔ The wearable vitals monitoring board is able to transmit collected data over BLE to other listening devices.
- ➔ Exception: power off, sensor data not received.

Display Data On Screen

- ➔ Nodes that are connected to a monitor are able to display received data in real time.
- ➔ Exceptions: power off, no display, no received data.

Receive Data from Specified Device and Sensor

- ➔ Nodes are able to filter through all the BLE messages and read only from user specified device(s) and sensor(s).
- ➔ Exceptions: power off, no incoming signals, user request is not applicable.

Resend Received Packets

- ➔ The nodes are able to receive BLE signals specific to the Vitals Monitoring System and resend them further to prolong the range of a message.
- ➔ Exceptions: no power, no incoming BLE signals, BLE signals are not applicable.

User Interface

The user interface will be accessible through an application (optional cell phone app or html) with BLE capabilities. The application should be able to support the following functionalities.

- Power On/Off
- Display:
  Each of the data will be displayed in a single row (see example below), with different colors and lines indicating the trend of any changes. The user should be able to select exactly which one he/she want to measure.

  Data:

  - Pulse Rate
  - Body Temperature

  If any of the data is beyond or lower than safety range, then an alarm will show up to notify any user reading the data.
- Panel:
  At default, all of the data should be zero and the line should be flat.

  Since the user might want to focus on only some of the data instead of all of them, the user should be able to disable some of the data display.

  Information read from another device can also be displayed on the screen based on selection.
- Trigger:
  All of the data should be triggered and update themselves every time a signal comes in

The information of vital signals shall be presented on the screen of an application (optional cell phone app or html). The displayed results will be rounded to nearest whole number.

The display shall be readable in direct sunlight and from any angle.

The screen will appear as follows:

Figure 2. User Interface Application example

System Functional Specification

The system is intended to measure and transmit 3 different kinds of vitals data comprising heart rate, body temperature and respiratory rate. These data are encoded into 47 bytes of data and transmitted via BLE beacon. The encoded data can be either received by another nearby monitor within 50 meters or by a laptop with bluetooth capabilities. If received by a laptop, the data can be displayed on the user interface so that they are easily understandable and readable.

Operating Specifications

The system shall be operating in any inhabitable environment

- Temperature range: -90 - 60 ℃
- Humidity up to 90% RH non condensing
- Power 5 VDC
    - The system shall operate for a minimum of 8 hours on a fully charged battery
- Aging Rate
    - 90 day < 3×10^{-8}
    - 6 month < 6×10^{-7}
    - 1 year < 25×10^{-6}

Reliability and Safety Specification

The Vitals Beacon shall comply with the following safety standards

- ANSI/ISA S82.01:1994 / Safety standard for electrical & electronic test, measuring, controlling, & related equipment / General requirement

FCC for commercial instruments

The MTBF will be a minimum of 10,000 hours

Design Procedure

Data Acquisition

I2C

The I2C protocol on the NRF52832 chip has a few frequencies to choose from. Due to the limitations from the two sensors, the lowest frequency, 100 kHz was chosen. For each data acquisition, the master sends a read request, and receives the requested data back from the slave. The data is stored in the corresponding buffer upon request awaiting to be sent out.

ADC

The analog to digital converter is internal to the nrf52832 chip and has 8 bit accuracy, which is enough to be used in a low powered mode and still provide with a reasonable resolution. ADC is used to read heart rate data that is sampled by an internal ADC buffer at a rate of 40 ms. after 17 samples the data is packaged and sent out.

Data Transfer

BLE

To transmit the message over the Bluetooth, nRF built in SoftDevice in Figure 3 was used to control the Bluetooth stack. SoftDevice is a wireless protocol stack library for building System on Chip solutions. Within the bluetooth stack Generic Access Profile (GAP) was used to configure the hardware to act as a Broadcaster and/or the Observer. In BLE applications the Broadcastor is always sending out the advertisement data to listening devices and the Observer is the device that always scans for the advertisement messages. The VBS main station is always using the Broadcaster capability and the Linux server is always using the Observer capability. However the node has to be both the Broadcaster and the Observer, which cannot be executed in parallel. For this purpose a scheduler is configured to allow for the Observer and Broadcaster to work when the other is not functioning.

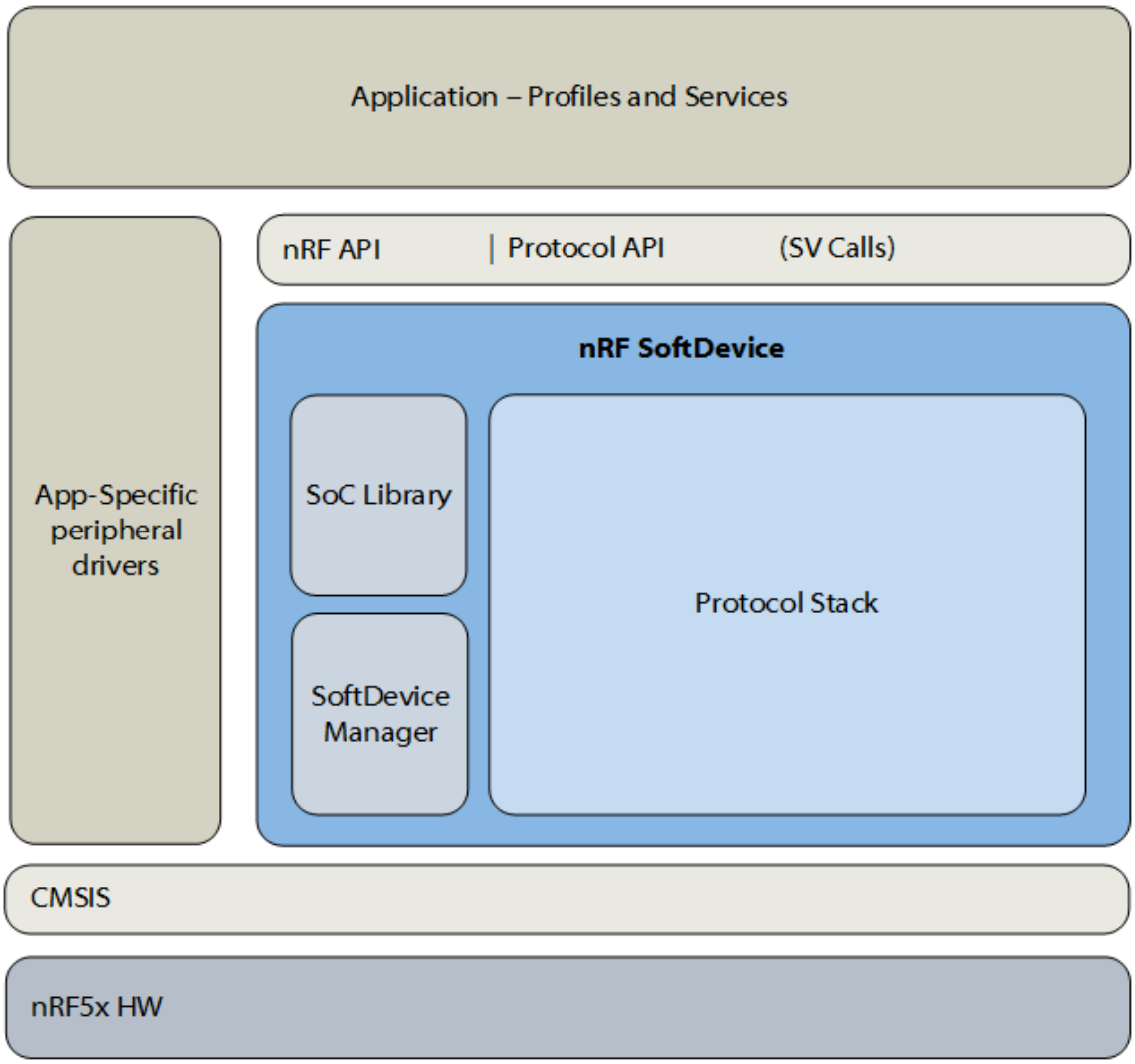


Figure 3. nRF Software Structure

Data Receiver

Linux

As mentioned previously, the Linux server takes the role of the Observer in the GAP parameters, where only scanning for the advertisement messages is used. This is all done with the help of the internal drivers configured on Linux OS: *BlueZ*. BlueZ is the official Linux Bluetooth protocol stack, which acts similarly to the SoftDevice, however it operates under a different hardware.

User Interface/Wifi

Webscoketd is used to setup and configure a Linux system into a server that is able to become a hotspot to display any kind of data over Wifi. Every device that is connected to the Wifi broadcasted by the pi (the developing board) can be the display server. The server is designed to be real-time, which means the server can receive data from the node station, display data, generate a dynamic line chart,

calculate BPM (beat per minute) and update everything without any missings or errors. In order to give users better overviews of their body information, the line chart is designed to be dynamic, smooth, and easy-to-see. The server generate and update the dynamic chart per three readings, and there are 17 heart beats measurement in each reading. In another word, the plot consisting of 51 points can show at least two peaks at the same moment, and the movement of the line can also demonstrate the trend of changing in heart rhythm. Meanwhile, for the purpose of conveniences, we provide several user interfaces including “stop”, “resume” and “print”. Furthermore, the server is expected to be reliable, which means the server should be able to keep running without error for a long time as long as the connection is not lost.

System Description

Overall the system consists of two main components: the main station in Figure 4 and the node station in Figure 5. The main station is responsible for gathering sensor data, assembling a BLE advertisement message, and broadcasting this data to all of the nearby devices. There are 4 main subsystems that add to this: Analog to Digital Converter, I2C Channel, BLE Signal Assembler, and BLE Beacon Transmitter.

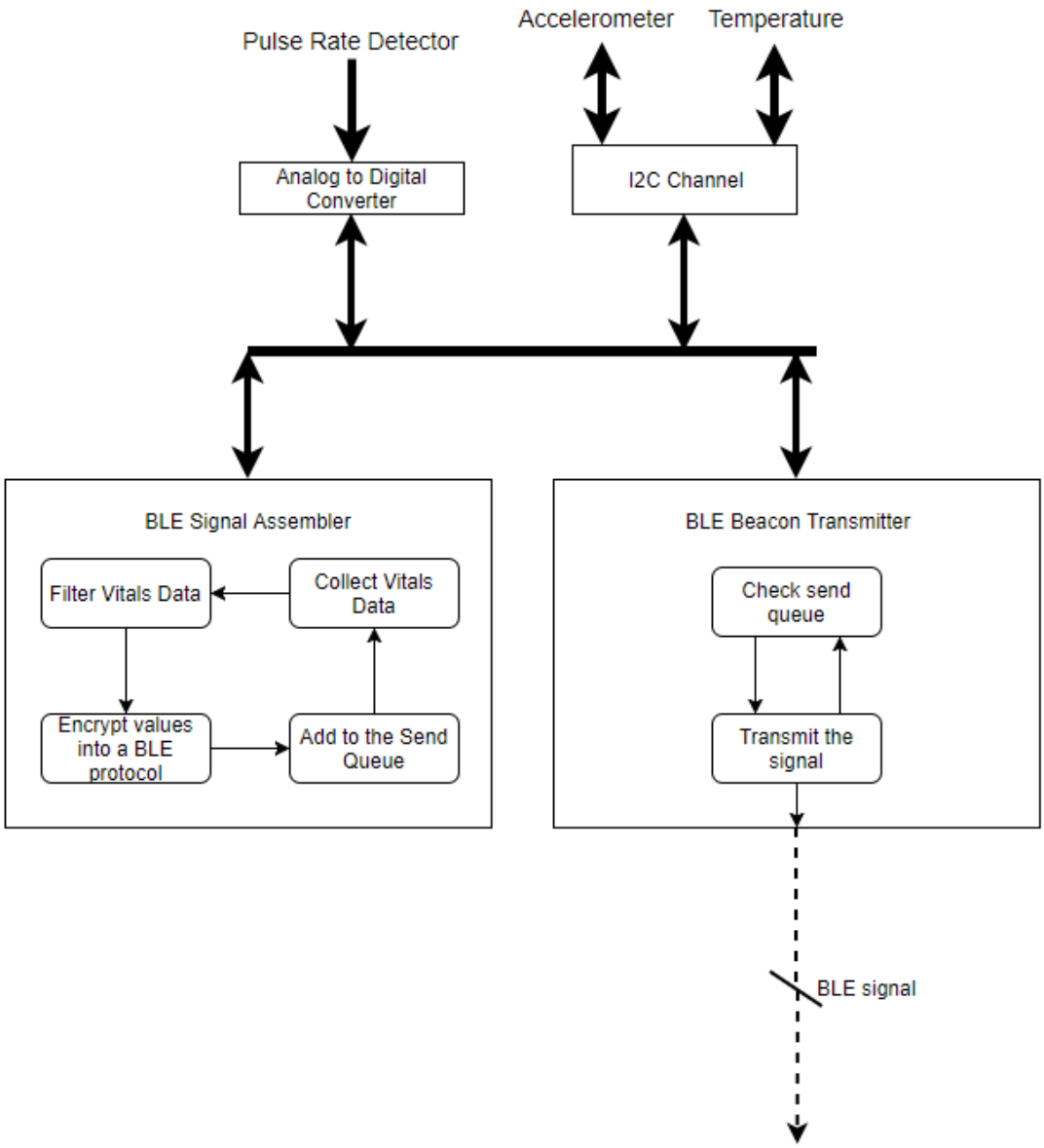


Figure 4. Block diagram for VBS Main Station

Analog to Digital Converting (ADC) Subsystem - the ADC subsystem shall convert the incoming analog vitals signal between 0-5 V into a 8 bit digital format. The subsystem passes on these digital signals to the BLE signal assembler for encoding.

I2C Subsystem - the I2C subsystem shall interface with 2 slaves sending commands and receiving data at 100 kHz. The data received can be either 8-bits or 16-bits, depending where the data was obtained. The subsystem passes on these data to the BLE signal assembler for encoding.

BLE Signal Assembler Subsystem - the BLE Signal Assembler subsystem receives the ADC measurements, filters the data for noise, and then encrypts the data with the

network code and device name into a 26 bytes as the Manufacturer Data to be sent out by the BLE beacon transmitter subsystem.
Beacon ID:

- Network code: 1 byte
- Device specific name: 1 byte
- Data length: 1 byte
- Index of the data: 1 byte
- Sensor data stack: 22 bytes
    - Heart Rate ADC data: 17 bytes
    - Temperature data: 2 bytes
    - Accelerometer data: 3 bytes

BLE Beacon Transmitter Subsystem - the BLE Beacon Transmitter subsystem is responsible for checking the send queue and sending out the next piece of BLE signal in the queue to any device that has BLE capabilities.

As seen in Figure 5, the node station is responsible for receiving incoming advertisement packages, filtering them for the packages from the main station, and sending them right away. The node station is made out of BLE Beacon Receiver and a Beacon Transmitter.

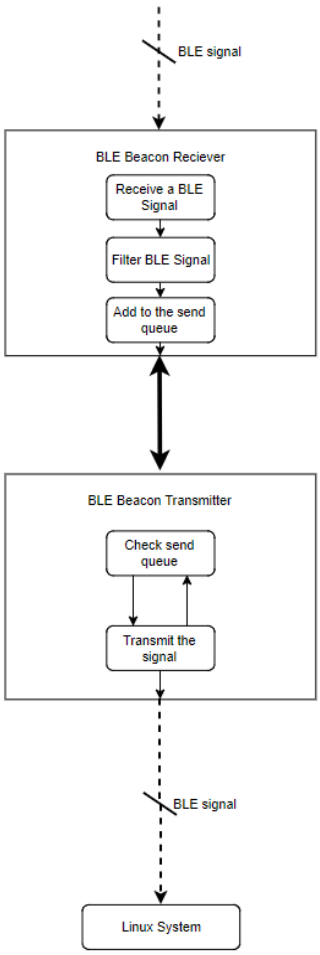


Figure 5. Block diagram for VBS Node Station

BLE Beacon Receiver Subsystem - the BLE Beacon Receiver Subsystem is responsible for scanning for BLE signals from other beacons in the same network and transmitting the data back into the field. This allows for the BLE signals to travel further distances, from projected 50 meter distance.

BLE Beacon Transmitter Subsystem - the BLE Beacon Transmitter subsystem is responsible for checking the send queue and sending out the next piece of BLE signal in the queue to any device that has BLE capabilities.

Timing Constraints

The entire system runs on tight timing restrains, since the purpose of the system is to produce the user with almost real time results. To make sure all the data maintained reasonable resolution, device operated in the low-power mode, and the stations had enough time to advertise the data, the timing of all of the components had to be taken into account. For this purpose it was decided to have ADC measurements to be taken every 40 ms. And after 17 measurements have been collected, an advertising message is built and is advertised at a limit of 100 ms per message. This allows for the message to be sent out

6-7 times until the ADC buffer fills up and a new message has to be sent out. Usually this is enough for the node to catch the message and resend it.

Software Implementation

Software from this project was developed from two main sources: nRF Software Development Kit made by Nordic Semiconductors from their nRF52 boards and BlueZ, a linux compatible bluetooth stack. The I2C, ADC, and Bluetooth drivers were already precompiled on the nRF52 board, as well as timiers, GPIO interrupts, and all other interrupts that were used in this project.

Functional Decomposition

As seen in the UML Use Case diagram in Figure 1, The system is divided into three parts: the main station, node station, and linux server/user. Each system needs input from the other one in order to function correctly.

Main Station: As seen in Figure 6 below, the main station is able to collect a variety of sensor data through the microcontroller enables I2C and ADC channels. ADC submodule is responsible for reading and filtering the data from the heart rate data. The I2C channel was responsible for reading the filtering the data from the temperature and accelerometer sensors. The main station also sends out advertisement messages with heart rate, temperature, and accelerometer data encoded into it.

Node Station: As seen in Figure 7 below, the node station is able to collect BLE advertisement messages, filter them for main station messages and re-advertise them again.

Linux Server: As seen in Figure 8 below, the Linux Server is able to collect BLE advertisement messages filter them for node or main station messages, decode messages into real data, and display this data on a hosted website.

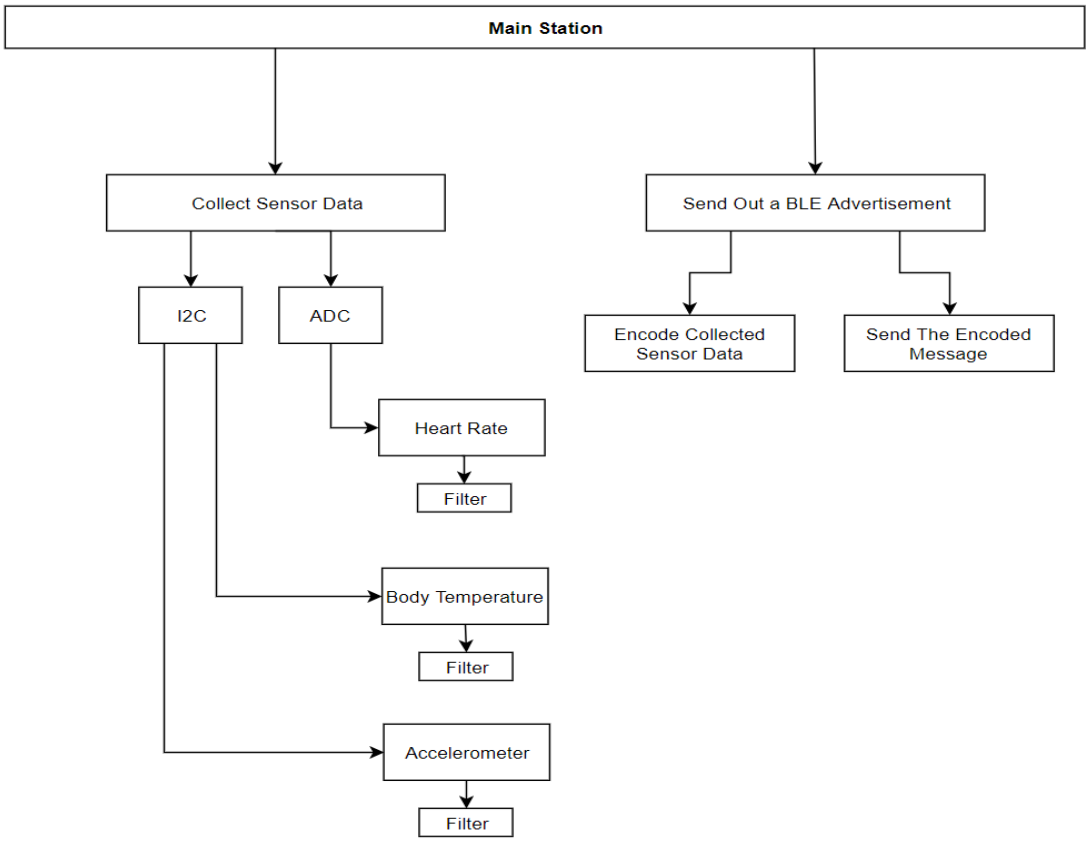


Figure 6. Main Station Functional Diagram

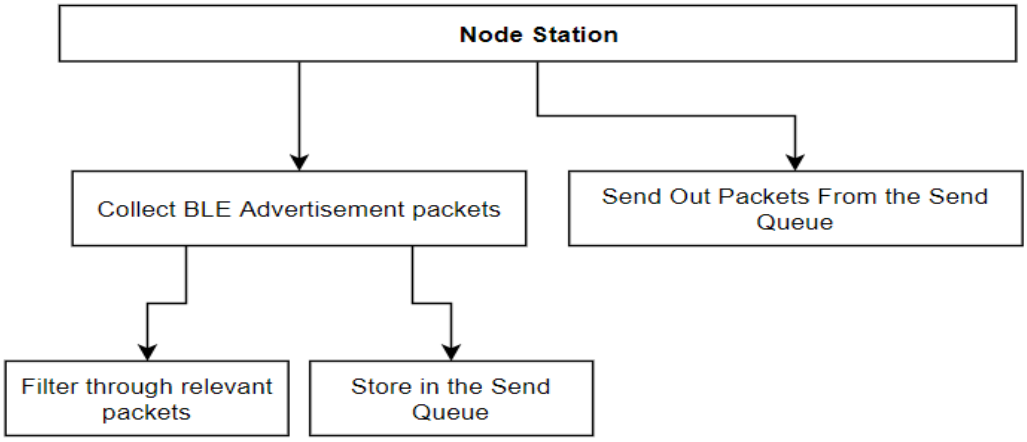


Figure 7. Node Station Functional Diagram

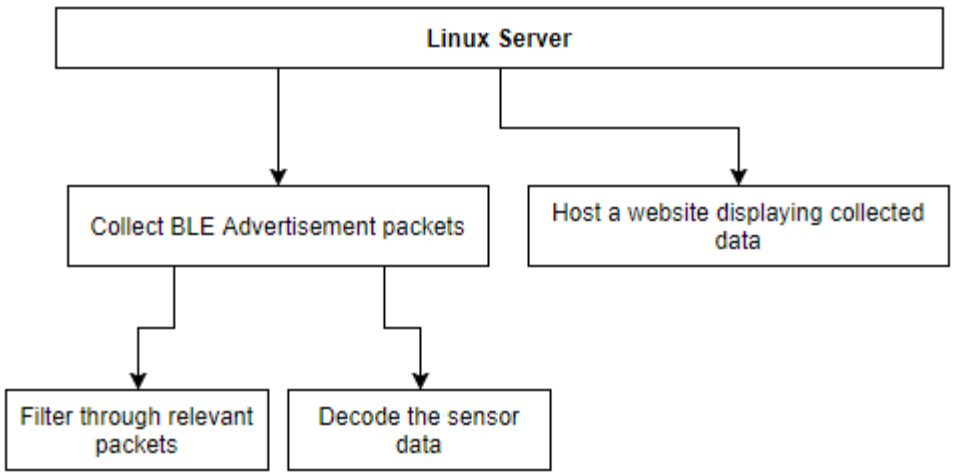


Figure 8. Linux Server Functional Diagram

Data Flow

The main data flow can be seen in Figure 9, where the data is collected with the main station microcontroller, encoded, and sent over BLE advertisement package to all listening devices. Then the BLE data can be caught by the node station and redistributed again to prolong the distance of the message. After the data is resent, the Linux Server can catch this BLE advertisement and display its content on a webpage hosted through Wifi to any internet acceptable device.

Main Station: The main station itself has an internal data flow that can be seen in Figure 10. The microcontroller mostly takes in data from other hardware components. Through ADC it is able to read raw voltage data from the heart rate sensor. Through the SDL and SCL the controller is able to request temperature or acceleration data. And thanks to the GPIO interrupt, the microcontroller is able to wake up from a power down mode.

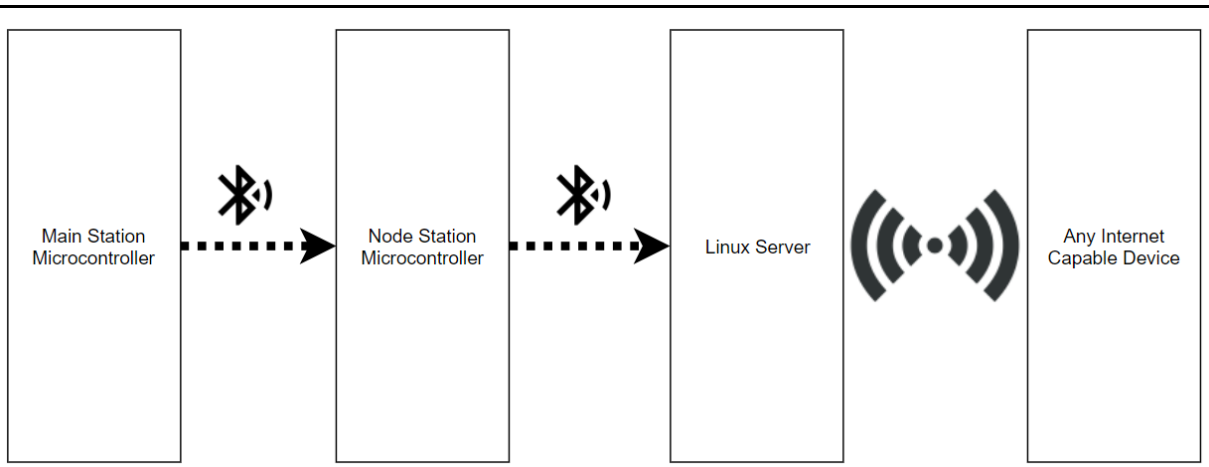


Figure 9. BLE and Wifi Data Flow Diagram

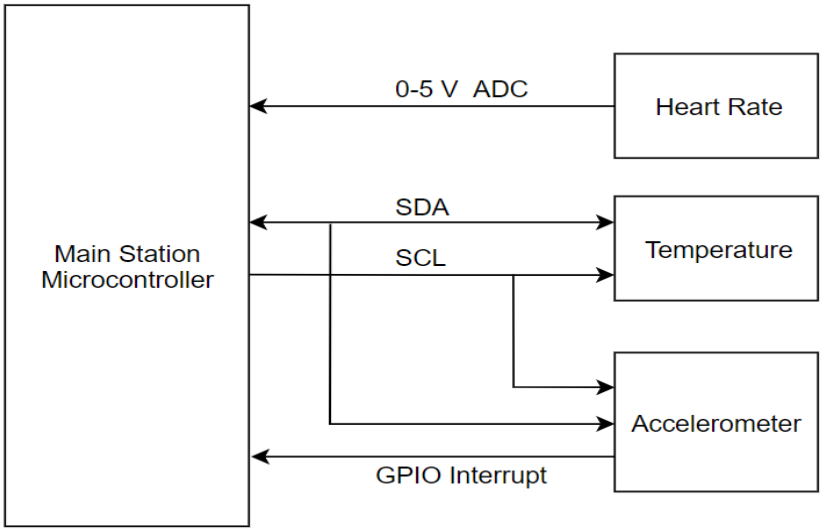


Figure 10. Main Station Data Flow Diagram

Activity Diagrams

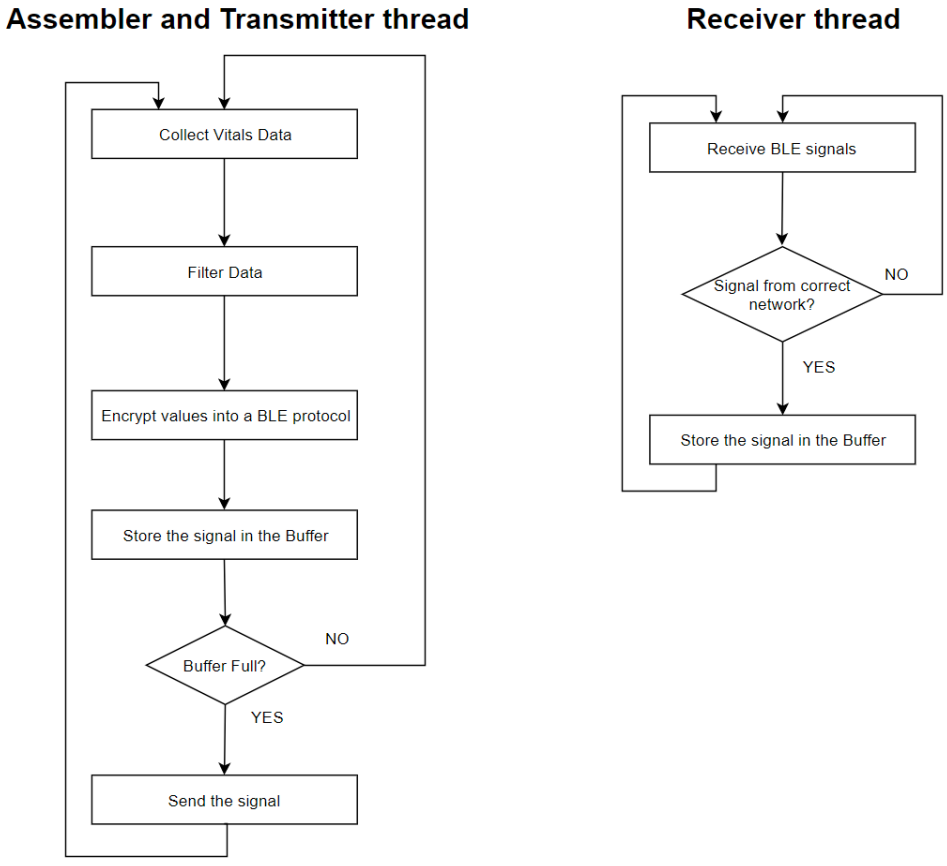


Figure 11. Activity Diagram for VBS Main Controller and Node

Hardware Implementation

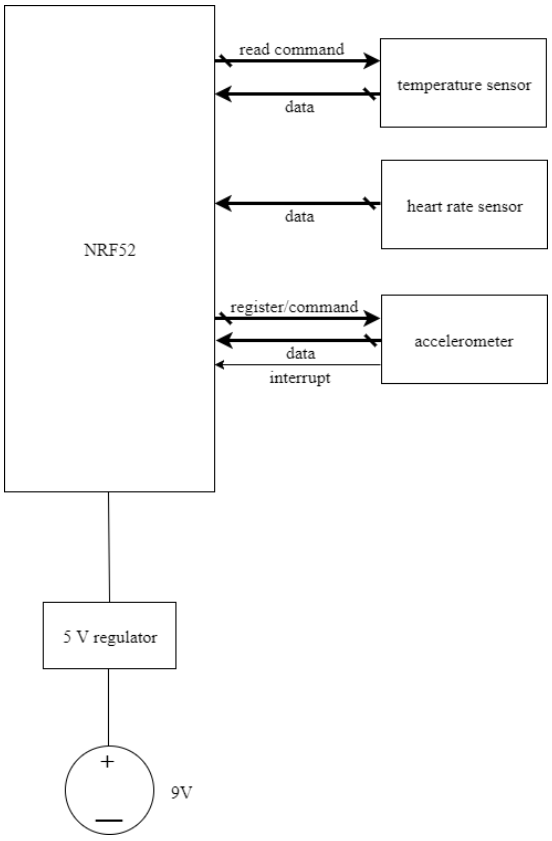


Figure 12. High Level Hardware Block Diagram

The whole system is implemented using two NRF52832 microcontrollers; one serving as the main system and one serving as the node. The main system collects data from the sensors and broadcasts the data in an encoded advertising message. The node picks up the advertising message from the main device and re-broadcasts it. By doing so, it increases the distance the main system and linux server can be apart from each other, giving the main device more flexibility and mobility. The whole system functions at 3.3 V thanks to an on-board voltage regulator on the NRF52832 breakout board.

Temperature sensor

Figure 13. MLX90614 Infrared temperature sensor

The temperature sensor is interfaced with the microcontroller through I2C protocol. Its SDA and SCL pins are connected to two GPIO pins on the microcontroller that are then configured to be working under the I2C protocol. The sensor stores three measurements in its EEPROM: ambient temperature, object 1 temperature, object 2 temperature. For the purpose of this project, the microcontroller requests the data of object 1, and this is done by sending the read command to the slave address of the temperature sensor. The temperature sensor returns an 16-bit temperature data that is collected through the ADC module inside the sensor. The temperature data is has a resolution of 0.02 and is measured in degrees Kelvin.

Pulse rate sensor

Figure 14. SEN11574 Pulse rate sensor

The pulse rate sensor is connected to the microcontroller via an ADC port. The ADC port is configured with the low-frequency timer embedded in the chip to measure 17 8-bit values at an interval of 40 ms.

Accelerometer

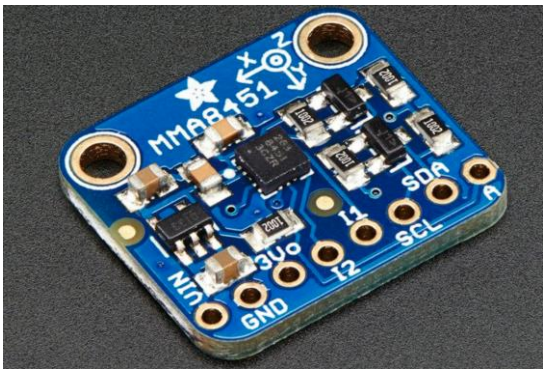


Figure 15. MMA8451 Accelerometer

The accelerometer also interfaces with the microcontroller through I2C protocol. It has two selectable slave addresses, depending on how the address pin is wired. If the address pin is wired to ground, it has a slave address of 0x1C, and 0x1D if it is wired to the 3Vo pin. For the purpose of this project, it is wired to ground. The accelerometer is configured to be in motion mode, meaning that when motion is detected, an interrupt signal will be generated at interrupt pin 1, which is signaled by pulling the line low. It is also configured to be in fast data mode, meaning 8-bit data in each direction is collected instead of 14-bit, resulting in the resolution of the data to be 0.0627g.

TEST PLAN

Since the whole system consist of several single parts, testing each of them individually is the first thing that needs to be done before testing the system as a whole. Based on the design, we have two NRF52832 microcontrollers, one as main device and one as node. For the main device, the data collecting feature should be tested first, since accurate data is the most critical. Since the data collecting process is based on sensors, the analog to digital converts, these are the actual elements to check. After that, even though the sensors can collect data, how to pass the data to the micotroller is the next problem. As a result, the next thing is to check the communication protocol between the microcontroller and the sensors, which is the I2C. If the main device can acquire data as desired, next thing need to be tested is the data transfer, between the main device and the node. After that the only thing left is the data receiver that is a web-based display system. The display system should be able to act based on user choice, receive the transferred data, and display the data real-time. After all individual parts are working as desired, then they can be combined as a whole system and tested.

TEST SPECIFICATION

Main Station

ADC

To successfully test the ADC functionality, the value that is sent through the wire should match the value measured on the oscilloscope with some precision. Since for the heart rate only the peaks are of interest, the resolution of ADC doesn't make a lot of difference.

I2C

For the I2C feature, the master should be able to send some specified values, for example, 0x0E to a register address of the slave. The slave should be able to correctly receive, store and send the data back to master. Master should be able to receive the value written back by slave.

BLE

For the BLE communication, the main device should be able to send out a BLE advertisement. The feature requires the device to encode desired message, and to send it out. First thing to be tested is if the message can be seen by other devices in close proximity, and then increase the range between the devices and test again.

Node

BLE

To successfully test the functionality of the node station, the BLE aspects of the station had to be in the working conditions. This has to be done by first testing if the incoming advertisement packets were available to be read and sent out. After the initial part of the testing, the concurrency of the advertisement and scanning of BLE had to be tested to make sure that all the signals coming in could be caught and all the outgoing signals were enough to be caught by the server.

Data Receiver

Functionalities

Since the web-based server provides several user interfaces, the system should be able to act as desired once a user press any specified buttons including "stop", "resume" and "print". At the same time, all information and the chart should be easy to see.

Accuracy & Efficiency

The system should be able to plot a line chart according to the latest three readings of heart rhythm. The system should be able to receive all data without any missings, and the data received should exactly match to the data being transferred. Since the received data is real-time, the system should also be real-time, and can update the line chart per three readings. Except for the line chart, other information displayed, including temperature

and accelerations, should also be real-time, and be able to update once new readings are received.

Reliability

The system should be able to keep running without any error pops up as long as the connection between the device and the server is not lost. Each individual part should work individually and not interfere with each other.

TEST CASES

Main device

ADC

The ADC configuration can be tested with 3 simple tests:

- Testing when the input is Ground, which should show up as 0 in the 8-bit configuration.
- Testing when the input is 5V, which should show up as 255 in 8-bit configuration, when the reference voltage is set to 5V.
- Testing with the oscilloscope using a sine wave with low frequency and an amplitude of 2.5 V between 0 and 5V.

I2C

There were several individual functions need to be checked in I2C protocol.

- For configuration, the logic analyzer was a useful tool. By sending a value 0xAF from master to slave and observing the SCL and SDA signals through the oscilloscope, the configuration could be checked easily. If the configuration is correct, then the signals should appear in a particular pattern.
- For the “master write”, oscilloscope was useful. By observing the signals, the writing feature could be tested. If the SDA signal matches what were sent by the master, then the writing feature should be fine.
- For the “master read”, it was tested together with the “slave write back”
- For the “slave read”, LED was useful. LED should light only when the value received by slave matches the value sent by master. In other word, if LED lights, then the reading feature was fine.
- For the “slave write back” and “master read”, LED should light only when the value received by master matches the value sent by slave. In other word, if LED lights up, then the features were fine.

BLE

BLE communication could be tested by either using a Linux server or using nRF developed application for iPhone: nRF Connect. Some of the tests that are needed to establish BLE are:

- Sending a BLE encoded advertisement package continuously. Using nRF Connect, check if the device and message appear in “Connections.” The device should have no name and doesn't have a connection option. The manufacturing data should be the same as specified in the code.
- Sending out BLE encoded advertisement packets while changing it once in awhile. This test would allow to test the fluidity of the BLE advertisement module, and would allow the data to be modified on the fly.

Data Receiver

Functionalities

There were three buttons with individual functionalities, the “stop” button, “resume” button and the “print” button. The testing was straightforward. If the system stops receiving data, the line chart stops “moving”, and the other information displayed is not updated any more after the “stop” button is pressed, then this part passed the test successfully. If the system restart receiving data, the line chart restart “moving”, the other information is updated, and the received data matches the latest data passed in after the “resume” button is pressed, then this part passed the test successfully. For the “print” button, if a window allowing the user to print the current chart pops up, then this part passed the test successfully.

Accuracy & Efficiency

For this part, test was done by observing the line chart and other displayed information. The line chart should be real-time, and the line should be flat at default. The line chart should change its pattern dramatically right after I pressed the sensor. If, at the same time, each point on the line chart matches the latest data received, then both the accuracy and efficiency of the chart should be fine. Similar approaches were made while testing the other information. If they are updated when a new reading comes in every time, and if the displayed value matches the received value, then this part passed the test.

Reliability

For the connection, test was pretty simple. The background was set to be green if connection is established, and white if connection is lost. If the background keeps green, and the webpage does not crash for a long time while the display system is running, it means this part passed the test.

FAILURE MODE ANALYSIS

Temperature sensor

If SDA is stuck at 0, the temperature sensor will be unable to receive or send any data through I2C, meaning the temperature received will always be stuck at 0. If SDA is stuck at 1, the temperature sensor will be unable to receive or send any data through I2C, meaning the temperature received will always be stuck at 0xFF.

If SDL is stuck at 0 or 1, the I2C will not have a functioning clock, meaning that the data obtained from the temperature sensor will be 0xFF, signalling that there is an error.

Pulse rate sensor

If the output voltage from the pulse rate sensor is stuck at Vdd, this means that the measurement from the ADC is always stuck at 0xFF. In turn, pulse rate cannot be measured since, the measured voltage is always above the threshold value for pulse rate calculations. If the output voltage from the pulse rate sensor is stuck at 0, this means the measurement taken in from the ADC will always be stuck at 0x00. This also means that pulse rate cannot be correctly measured since the measured voltage is never above the threshold voltage for pulse rate calculations.

Accelerometer

If the address is stuck at 0, the slave address of the accelerometer will be 0x1C. If the address is stuck at 1, the slave address of the accelerometer will be 0x1D.

If SDA is stuck at 0, the accelerometer will be unable to receive or send any data through I2C, meaning the temperature received will always be stuck at 0. If SDA is stuck at 1, the accelerometer will be unable to receive or send any data through I2C, meaning the temperature received will always be stuck at 0xFF.

If SDL is stuck at 0 or 1, the I2C will not have a functioning clock, meaning that the data obtained from the accelerometer will be 0xFF, signalling that there is an error.

If the interrupt pin 1 is stuck at 0, the interrupt when motion is detected will never be triggered, since the GPIO port is looking for a high to low transition. This means that the main station will never be woken up from sleep. If the interrupt pin 1 is stuck at 1, the interrupt for when motion is detected will never be triggered, since the GPIO point is looking for a high to low transition which will never happen. This means the system can never be woken up after it goes to sleep.

PRESENTATION, DISCUSSION, ANALYSIS OF RESULTS

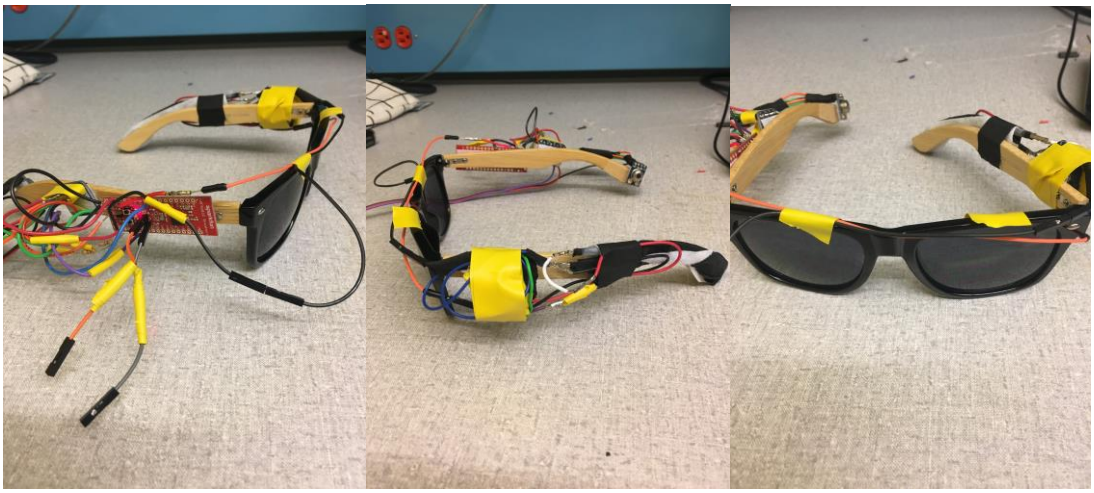


Figure 16. Pictures of final product mounted on glasses

The final assembly of the project was mounted onto glasses so that accurate measurements from both the pulse rate sensor and infrared temperature sensor. The system initially was intended to be on the wrist, but because of the limitations of the sensors chosen, this was not possible. For the temperature sensor, the temperature measured was a bit below body temperature; for the pulse rate sensor, the skin on the wrist is too thick for the sensor to get accurate readings.

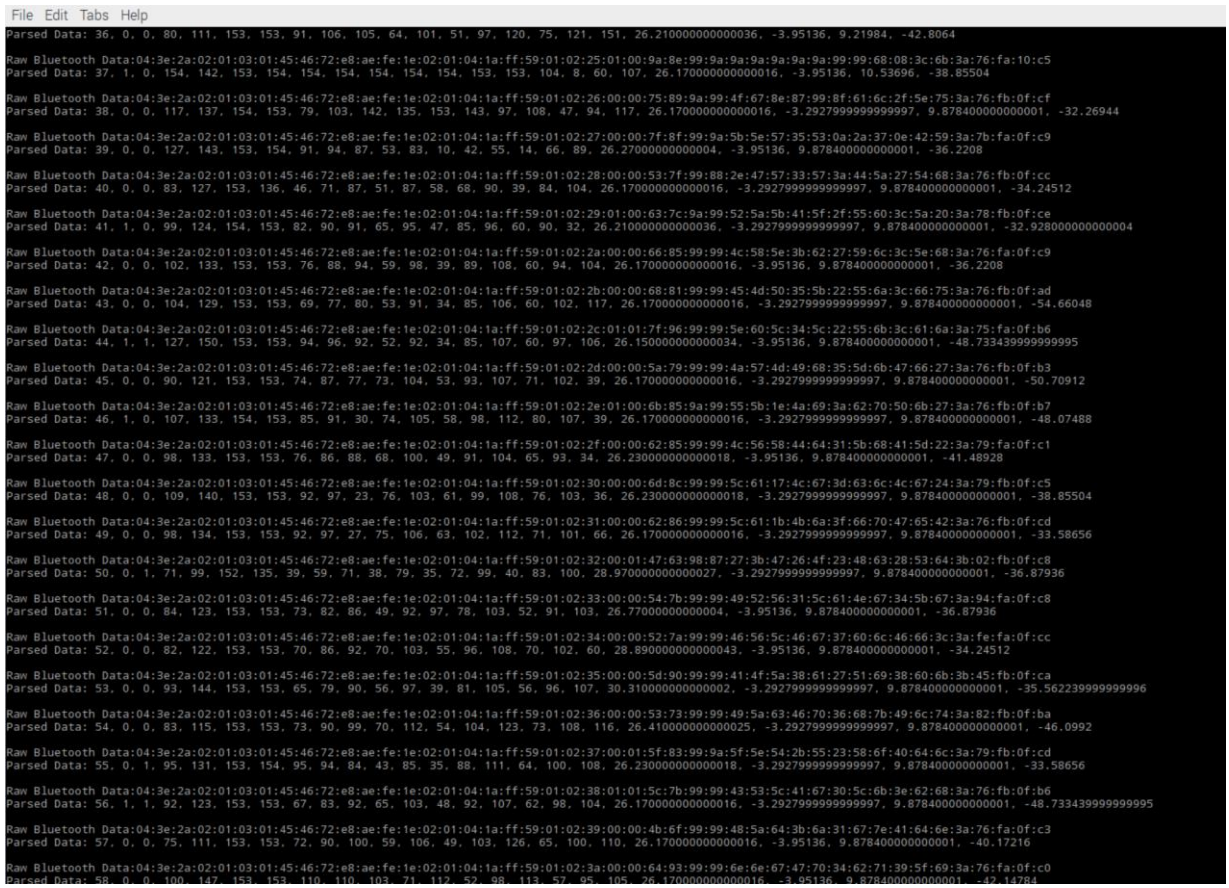


Figure 17. Data received by the Linux server sent from the main station

The results shown in Figure 17 proves that the messages sent by the main station was received correctly by the Linux server. Messages received from the device identification number, 5900 are printed out on the screen, and the data matches the debugging messages from the main station that are displayed through UART on another computer.

Figure 18. Data received from both the node and the main station

As can be seen from Figure 18, the Linux server is able to receive messages from both the node and the main station. The data index from the node and main station matches up, which shows that the node is capable of advertising and receiving at the same time.

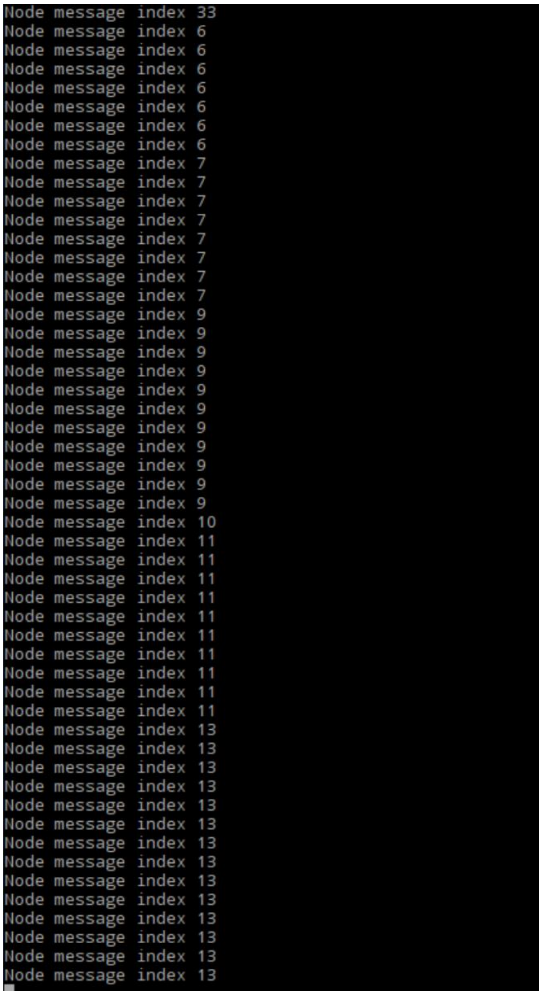


Figure 19. Messages received from the node

One of the biggest challenges when sending messages from the node was the timing constraints. As seen in Figure 19, messages sent from the node are dropped once in awhile. However, this does not affect the data plot in any significant way since one message only represent 40 ms worth of data collected from the pulse rate sensor.

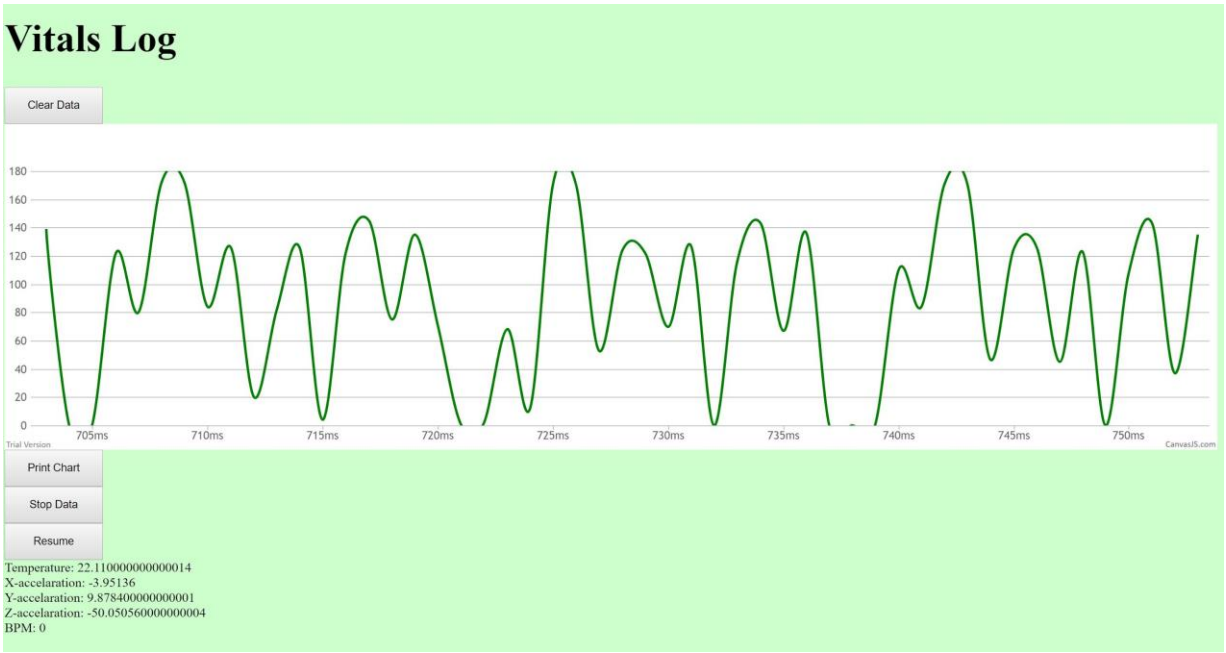


Figure 20. Plot from pulse rate sensor with 3.3 V

Figure 21. Plot from pulse rate sensor with < 3 V

As can be seen from the Figure 20, the plotted output from the pulse rate sensor reflects a lot of noise. This is because the when the voltage difference between minimum and maximum values are greater, there are more room for the noise. So when the input voltage is decreased to less than 3 V, the resulting plot is a lot cleaner as seen in Figure 21.

ANALYSIS OF ERRORS

Several errors that surfaced during the debugging phase and their solutions are as follow:

- The server did not receive all of the BLE advertisement packages
    - Reason: the node was either too far away from the server or there was some kind of the abstraction.
        - Solution: to move the node closer
    - Reason: the node wasn't working concurrently with reading and sending advertisement data.
        - Solution: Change scan intervals and window to optimize the performance.
- The main station would deplead the coin cell batteries after few hours of activity
    - Reason: the system continuously used the BLE stack
    - Solution: timeout the system after 30 messages and turn on the system based on the GPIO interrupt from the accelerometer due to movement.
- The web server was getting slow and slow after running for a while
    - Reason: the size of the storing array was keep increasing as more and more data came in.
    - Solution: the design approach was abandoned, and new design was applied.
- The data displayed and the line chart was not real-time. Obvious lag existed, sometimes the plot would become blank, and sometimes the server would stop responding
    - Reason: in the previous design, data process, line chart plotting, and information display was designed as three individual parts that were asynchronous. The previous way was that the system would process and store the data first in the array, and data would be fetched whenever needed. Since it was almost impossible to make the timers of the three individually parts totally consistent, some values would become "NULL" after running a while.
    - Solution: in the new design, we combined the plotting and information display into the processing part. Every time a new data came in, values to be displayed would be updated, and the chart would be updated once the system had processed three data.
- The BPM (beat per minute) was not accurate enough.
    - Reason: because of the noise or other factors, sometimes a peak contained 3 or 4 values greater than the threshold value, but sometimes a peak contains 10 more values greater than the threshold value. Normally these values were different by 1 or 2. It was hard to determine an optimal threshold that could help us accurately determine the time interval between two adjacent peaks showing in the chart.
    - Solution: one of the traditional ways to calculate the BPM is to print out the chart and calculate the BPM with a special ruler. We added a printing feature that allows a user to print out the current chart, giving the user a chance to calculate the BPM in another way.

SUMMARY AND CONCLUSION

The project was Vitals Beaconnning System, that collects heart rate, temperature, and accelerometer data and sends it out using BLE advertising packets. These advertisement packets, as proven in this project, can be collected by various BLE capable device and resent, to prolong the message distance, or decoded and used by medical professionals to monitor the vitals of patients in real time using local internet connection. This project presents many further opportunities and questions into the design of systems that use any kind of sensor station. Any product in the field can be using the BLE beaconnning system to distribute information wirelessly.

Overall this project was successful. With the help of nRF52 hardware a prototype for VBS was built around the glasses to allow the user be unrestrictive in movement while using the product. There have been challenges present in this project that have been overcome with the help of numerous hours of research and trials-of-error. For example, BLE communication not working and server overloading. However after all of the errors have been handled, the system worked without a problem, allowing further iterations of the project to be taken into account.

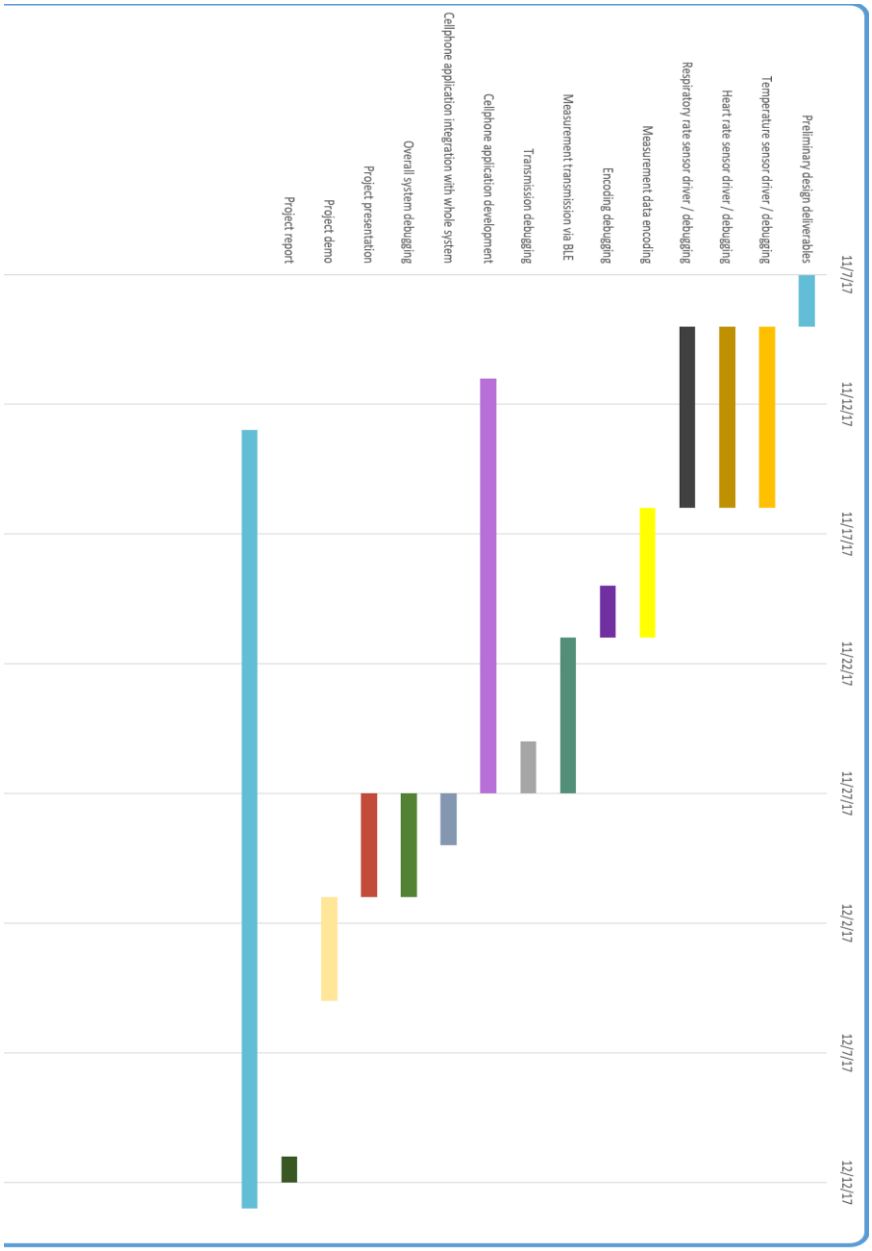
APPENDICES

Bill Of Materials

Item	**Amount**	**Cost**
nRF52832	2	39.9
Infrared Thermometer MLX90614	1	19.95
Pulse sensor module	1	12.26
MMA8451 accelerometer	1	7.95
5V voltage regulator	1	0.63
3V coin battery	3	9.75
	Total	90.44

Gantt Chart And Plan

Task Name	Start Date	End Date	Duration	Person
Preliminary design deliverables	11/7/2017	11/9/2017	2	All
Temperature sensor driver / debugging	11/9/2017	11/16/2017	7	Sandy
Heart rate sensor driver / debugging	11/9/2017	11/16/2017	7	Denis
Respiratory rate sensor driver / debugging	11/9/2017	11/16/2017	7	Jiayou
Measurement data encoding	11/16/2017	11/21/2017	5	Jiayou
Encoding debugging	11/19/2017	11/21/2017	2	Jiayou
Measurement transmission via BLE	11/21/2017	11/27/2017	6	Denis
Transmission debugging	11/25/2017	11/27/2017	2	Denis
Cellphone application development	11/11/2017	11/27/2017	16	Sandy
Cellphone application integration with whole system	11/27/2017	11/29/2017	2	Sandy
Overall system debugging	11/27/2017	12/1/2017	4	Denis
Project presentation	12/1/2017	12/5/2017	4	All
Project demo	12/11/2017	12/12/2017	1	All
Project report	11/13/2017	12/13/2017	30	All


Preliminary design deliverables
Temperature sensor driver / debugging
Heart rate sensor driver / debugging
Respiratory rate sensor driver / debugging
Measurement data encoding
Encoding debugging
Measurement transmission via BLE
Transmission debugging
Cellphone application development
Cellphone application integration with whole system
Overall system debugging
Project presentation
Project demo
Project report
11/7/17
11/12/17
11/17/17
11/22/17
11/27/17
12/2/17
12/7/17
12/12/17

This report and its contents are solely work of the participants below.

Denis Jivaikin

Sandy Lee

Jiayou Zhao