Electronics and Computer Science
Faculty of Physical Sciences and Engineering
University of Southampton

Aran McConnell

12th May 2020

Use of Embedded Devices in rehabilitation monitoring for leg surgery patients

Project supervisor: Prof. Neil White
Second examiner: Prof. George Chen

A project report submitted for the award of
MEng Electronic Engineering

Todo list

Abstract

This report has been written to demonstrate the creation of an Embedded Device that monitors leg surgery patients raising their leg for periods of time, and the creation of an accompanying Android App that displays this data to a user. Together these items allow a patient and doctor to monitor recovery and rehabilitation pre, and post surgery to improve their personal recovery and enhance the experience of future patients.
The device described is an Arduino based device utilising an Inertial Measurement Unit, whilst the application is developed using the Xamarin Forms framework.

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must change the statements in the boxes if you do not agree with them.

We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

Madwick's MARG Algorithm was utilised in the project, using the open source code provided on [14], and based on the work from [11].

Arduino Libraries Used:

Arduino BLE
Arduino_LSM9DS1
SPI
Mbed
RTOS

The Xamarin Forms Framework was used for the Android App with the following packages:

Plugin.BLE [1]
Plugin.Permissions
CSVHelper
Acr.UserDialogs

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself, or with my allocated group, and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs.

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

I have acknowledged all sources, and identified any content taken from elsewhere.

If your work involved research/studies (including surveys) on human participants,

their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

ERGO approval documents clearly shown in Appendix A **ERGO Reference Number: 52959**

ECS Statement of Originality Template, updated August 2018, Alex Weddell aiofficer@ecs.soton.ac.uk

Acknowledgements

I would like to thank my superverisor, Prof. Neil White for his guidance and support throughout, as well as Mahdi Shaban for his valuable advice.

Contents

List of Figures

Definitions and Abbreviations

IMU - Inertial Measurement Unit

BLE - Bluetooth Low Energy

ERGO - Ethics and Research Governance Online (The University process to approve studies, in my case the testing phase)

SoC - System on Chip

RTOS - Real-Time Operating System

API - Application Programming Interface

CSV - Comma Separated Variable

SMD - Surface Mount Device

Chapter 1

Introduction

1.1 Motivation

There is currently a huge application for electronic monitoring technology in healthcare, especially in the form of wearable or similar devices that can capture personalised medical data, which allows doctors to improve their medical advice, and tailor their care in a personalised way based on individual needs.
One specific area where this has applications is in ankle breakages. Before such a break can be operated on the patient is usually instructed to keep the foot raised for periods, which helps fluid drain, allowing it to be operated on. The main issue in this situation is that doctors are uncertain as to whether their advice is being followed, and if so to what extent. Having a device that could allow the doctor to monitor this situation would help them tailor their medical to advice to be more effective overall.
Additionally such a device could also be used to monitor steps in the post surgery rehabilitation stage to again give the doctor data to help them tailor their advice.

1.2 Goals and Aims

The goal of this project is to produce an embedded solution, using an existing microcontroller board (or a modified version) and an external memory IC in order to take measurements allowing the determination of when the patients leg is raised and to analyse walking in the rehabilitation stage. This device must be powered by a battery with a reasonable battery life, and to be packaged in a way that it can be worn around the ankle of a patient in a non-invasive way. The device must store its logged data locally so to remove the dependence on a smart phone.
The device must also be able to share this data with a smartphone App. over Bluetooth. This accompanying App. being the other half of the project, which must take the data from the device and store/display it in an intuitive and informative way for the user, being the patient and/or doctor.

Chapter 2

Literature Review

2.1 Medical Aspects

Breaks and fractures located somewhere in the leg and ankle are among some of the most common types of bone breakages [12]. According to [5], injuries to the ankle are the third most common sports injury, with injuries to the knee and lower leg also coming in the top ten injury areas.

2.1.1 Monitoring in a clinical environment

When using technology to monitor in Healthcare, there are a number of common terms, such as Medical IoT, Big Data, and "Digital Healthcare". All of which refer to the use of technology to collect personalised medical data in a clinical setting. The use of these technologies in Medicine is a rapidly growing market, and it will possibly have the "most important, and personal effect" on healthcare [7]. By 2020 it is estimated that health related IoT will make up 40% of the IoT market, making it the largest IoT category [7].

There are many examples of this "Digital healthcare" already in use, or in research stages. One such implementation is the "SENSE-PARK" sensor system [9] which uses a set of sensors to monitor the movement of Parkinson's Disease patients during day and night. The paper [9] details a trial involving using this sensor system and monitoring the "Health Related Quality of Life (HRQoL)" reported. This test found a positive correlation overall between use of the monitoring system and the reported HRQoL.

Some implementations of these technologies are more widespread in consumer products, such as the ECG feature on the apple watch, as discussed in the paper by C. Worsham and A. B. Jena [13]. This implementation is interesting as due to being a consumer product, it is seen to be inaccurate, and is thus dismissed by many doctors. However, it provides very useful data for a number of different scenarios.
The most interesting thing about the Apple Watch ECG is the number of people

with Apple Watches, and thus the sheer volume of data these devices produce. Despite the ECG not being approved to make a diagnosis [13], it, along with the use of algorithms, can be used to flag up possible conditions and advise that proper medical help is sought.
As highlighted in this paper, one of the most interesting uses for such data is that due to the sheer volume, it could be used to build up a large database. By then correlating this data with factors such as age, sex, ethnicity, etc., risk factors could be found for certain conditions, and small variations in the ECGs could be discovered as early signs of seemingly unrelated diseases.

2.2 Technology

2.2.1 Embedded Hardware Implementations

In order to achieve my goals, the Embedded Device must meet an number of requirements.
To achieve this a board with integrated sensors that will allow me to take the required measurements is required. The board must have Bluetooth connectivity to communicate with the device running the App., Bluetooth Low Energy (LE) is desirable here to increase battery life. Another important requirement is that the device contains some form of non-volatile memory so that the device can log data when not connected to the device running the Android application. The device also has many constraints regarding its physical properties, such as size and weight which affect its ease of use by a patient.

Texas Instruments CC2650 SensorTag

This board comes in very convenient package, being small and with a silicone cover that makes it physically ready to use for my application. It has a number of useful sensors, and Bluetooth LE already on board, as well as a coin cell holder. However, due to it's "complete" package it is very limited in expanding its capabilities and so for my application where I need a form of flash memory, it isn't quite suitable.

Libelium Waspmote

The Waspmote is an interesting solution that combines a base board with a number of add-on boards depending on the requirement. This board would have most of the features required by way of add-on boards, and even has an SD card slot onboard. This board however is too large for this application, and especially with all the add-on boards required, has an awkward form factor that makes it unsuitable for this project.

Arduino Nano 33 BLE Sense

This board has just been released recently, and so contains extremely modern features such as Bluetooth 5 LE, and a host of modern sensors. The form factor

is also suitably small. The only downside with this board is again the lack of onboard memory, and the lack of a battery/holder. This board however has great expandability so these missing features could be added with relative ease. The board is also compatible with the Arduino IDE and ARM's MbedOS making development for this board easier, and feature rich due to MbedOS support.

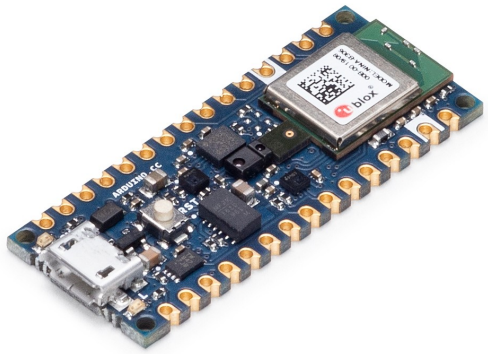


Figure 2.1: Arduino Nano 33 BLE Sense board, sourced from [3]

From all the discussed options, the Arduino Nano 33 BLE Sense is the most viable option, this is due to the rich array of sensors, Mbed support, attractive form factor and its essential expandability. The cost is also very reasonable compared to alternatives.

2.3 Algorithms

One issue when using IMUs in such an application is they suffer from error accumulated over time, or "drift". Due to the use of constant integration any small measurement errors build up over time to form significant errors, making them useless unless compensated for.
In order to deal with such errors, and also to determine rotation and position from the IMU measurements, we will need to make use of some form of algorithm.

2.3.1 Kalman Filter

The Kalman Filter is a widely used algorithm, which is not specific to working with IMUs, but can be used to make an "educated guess" [6] of the next state of the system where the information available is uncertain.
The Kalman Filter treats the uncertain variables as random and Gaussian, giving each a mean μ and a variance σ^2.
The Kalman filter is ideal for constantly changing systems and are computationally light so would be well suited for embedded applications. [6]

2.3.2 The Complementary Filter

The Complementary Filter [10] is a direct alternative to the Kalman filter for IMUs. It is much simpler, and even less computationally expensive allowing it to be implemented on resource constrained microcontrollers.
This filter uses the fact that an accelerometer does not experience drift errors, while a gyroscope is not susceptible to external forces. The filter takes these advantages of each sensor, and uses them to remove their individual disadvantages.

2.3.3 Madgwick MARG Algorithm

This algorithm was developed by Sebastian O.H. Madgwick as part of his PhD research [11]. It represents orientation in a quaternion form in order to prevent issues with representing orientation using Euler Angles, such as Gimbal Lock. This algorithm can also be used with MARGs (Magnetic, Angular Rate, and Gravity) systems. One of it's big advantages is the fact that it does not need a high sampling rate as with the Kalman filter and others, which is very useful in a memory and power constrained system.

Chapter 3

Specification

This project has a specification that must be met in order to be successful. This specification consists of a number of different requirements, which can be used to measure how successful the project has been at its conclusion.

3.1 Project Scope

The Scope of this project is to produce an Embedded Device and accompanying Android Application that will detect when the wearer's leg is raised. The Embedded Device will run off batteries and be small and unobtrusive to wear. It will be able to take the described readings without the need for a smartphone to be continuously paired and will store them locally in memory.
Regarding the application, it will allow some control of the Embedded Device and be used to download the data from the device's memory. It will then apply algorithms and processing to the raw data, and display it all in an intuitive and easy to use manner. This data will stay local on the device and I will not be working with any kind of server for "cloud" upload. The App. itself is designed to be used to analyse the data.

3.2 Requirements

Embedded Device Functions

- Record raw 9 axis IMU data at a set logging rate.
- Record without smartphone presence.
- Store an acceptable number of readings locally.
- Communicate with the Android App over BLE.
- Be powered by a CR2032 Battery, and have a reasonable battery life.
- Have a small footprint so as it would not be obtrusive to wear.

Android Application Functions

- Control the Embedded Device using BLE, e.g. start/stop logging.
- Download the data stored on the Embedded Device over BLE.
- Process the received data to determine at what points the leg was raised.
- Display the downloaded data in a clear and user friendly manner.
- Store multiple sets of readings and have them easily differentiable from one another.

3.3 Constraints

This project has a number of constraints which may limit its level of completion somewhat. These constraints may be technological, time based, or limited by official processes.

List of constraints

1. The project, including the report must be finished by the 5th of May 2020.
2. The testing phase is limited by ERGO approval.
3. To test effectively enough individuals need to take part in the study.
4. Delivery time of hardware and PCB.

Chapter 4

Design

This chapter will cover the design and structure of the project from a high level approach, as well as how different sections are connected to one another.

4.1 Overall System Architecture

The chosen architecture for my system can be split into two distinct sections. The Embedded Device, and the Android App. Each section is designed, and operates separately, with the BLE interface the only component that links them together. An overview of the system architecture is laid out in figure 4.1, showing the two sections of the project, along with how they interact with each other, and their associated components.

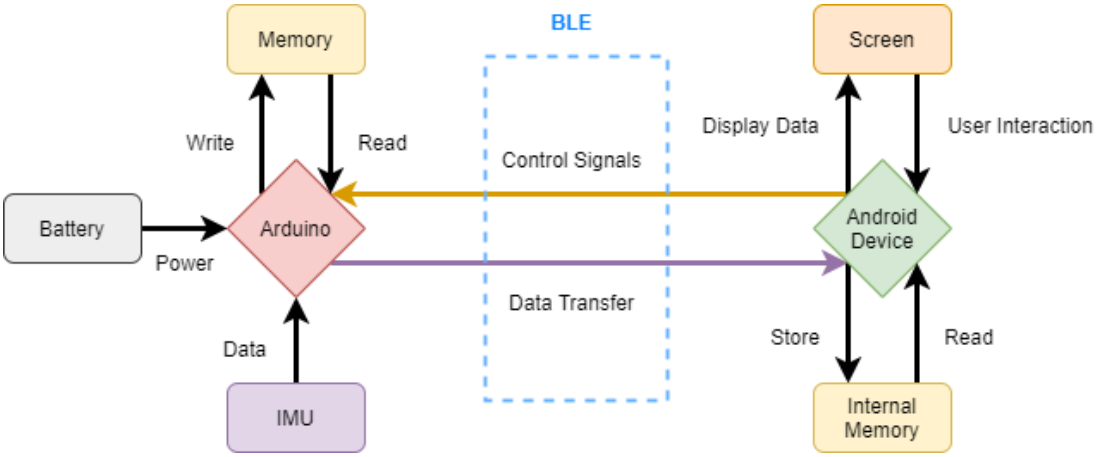


Figure 4.1: Full system diagram

4.2 The Embedded System

4.2.1 Hardware

The embedded system has a very simple overall structure. It consists mainly of the Arduino Nano 33 BLE Sense board, which contains two key components for this project. The first of which being the nRF52840 SoC (Grey outline in figure 4.2) from Nordic Semiconductor, which consists of an ARM Cortex-M4 32-bit 64MHz Processor and a Bluetooth 5 Transceiver. The second key component is the LSM9DS1 IMU from STMicroelectronics (Orange outline in figure 4.2), which is a 9-axis IMU, consisting of a 3-axis accelerometer, gyroscope, and magnetometer. This board was selected as the ARM Cortex-M4 included is powerful enough for any requirements of this project, whilst still being very power efficient and low cost. The main driving factors in choosing this board was the built in Bluetooth Transceiver (including the BLE standard which is essential for maximising battery life) and the included IMU, which is precise enough for this application. The fusion of these components in such a form factor made this board a compelling choice.

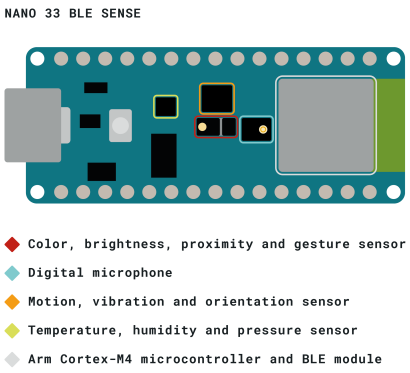


Figure 4.2: Arduino Key Component Diagram, sourced from [3]

The rest of the hardware consists of a Flash Memory IC to allow non-volatile storage of readings taken. The chip used here was the SST25VF080B 8Mbit SPI Serial Flash chip from SST, this chip was chosen as flash memory is durable and can be read and written easily. This chip itself was readily available and low cost, which was a main driver in the choice. A larger memory capacity that 8Mbit would be desirable, however due to its availability this chip was sufficient for purpose.

Finally in order to power the system a CR2032 battery holder is included as a simple and low cost power source with a small footprint.
As the footprint and packaging of the device was a design concern, all three components were to be combined on a small PCB of an appropriate shape and size. This PCB would then fit inside a 3D printed enclosure to ensure the components

would be protected in use, and that the device would be comfortable to wear strapped around the ankle.

4.2.2 Embedded Software

The software side of the Embedded Device incorporates raw sensor data collection, data storage and BLE functionality.
The processor on the Arduino 33 BLE supports the Mbed RTOS from ARM, being the first board in the Arduino line-up to do so. Therefore I decided to use features of the RTOS to have a multithreaded structure to the firmware, and allow code to run concurrently.

Two different threads will be required, one to handle the "Logging" functionality of the device (handling the taking of readings and storing these in memory), and one to handle BLE functionality. This needs to run alongside the Logging thread, as the state of BLE variables and BLE functions may change at any time. The logging thread must remain uninterrupted during such events.

Figure 4.3 shows a flow diagram of the two threads described above. These are the high level functions performed by each thread. The order of execution and response to control signals is also shown.

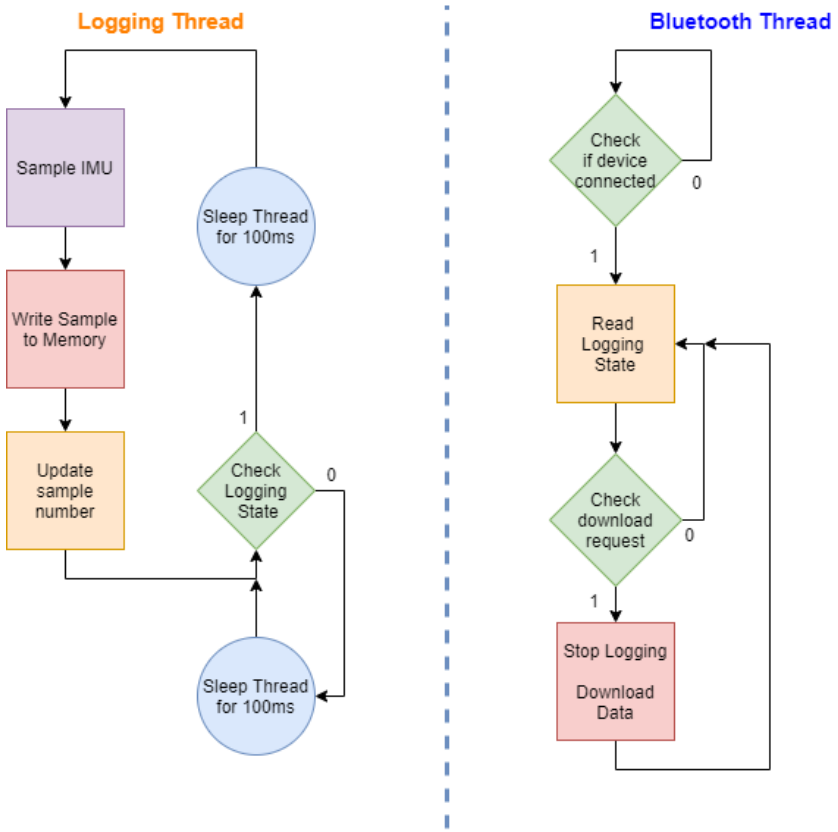


Figure 4.3: Embedded software flow diagram

The Logging thread has one main control signal, "Logging State" (where 1 = logging and 0 = not logging). When this signal is 1, the thread will take IMU measurements and store these in Flash Memory. The sample number will then be incremented to keep track of the number of samples taken. This process will repeat every 100ms, giving a 10Hz sample rate. If "Logging State" is 0, logging is paused, whilst a change to "Logging State" is monitored every 100ms.

The Bluetooth thread allows control signals to be changed alongside the Logging Thread. With no connection, the device will poll for a new connection. When a device connects it will read the "Logging State" from the BLE characteristic and update the local variable. When connected the "Download Request" BLE characteristic is polled, and if 1, a download mode will be entered, where "Logging State" is set to 0 and a download protocol communicating with the Android app will be initiated.

Along with these threads, some setup code will be run before the tasks begin. This will involve setting up the IMU, and BLE by adding the required services/characteristics and advertising services. The memory will also be setup here by configuring the SPI interface and reading information stored on the chip to recover the current write position, in case of a power cycle.

4.3 The Android Application

The Android Application is the other side of the coin, it forms the second half of the project. The purpose of the App. is to interact with the Embedded Device and to analyse the collected data.

4.3.1 Hardware

The only hardware required for the Android Application is a device running Android 4.3 or later. The device must have a Bluetooth transceiver supporting the BLE standard, and Android 4.3 onwards, as this is the Android version which added BLE support to the API.
The App is downloadable on any device as long as it meets the specifications above, which could be useful for demonstration. For the purposes of development I will use my personal Android Phone (OnePlus 6), the specifications of which are:

- 8GB RAM
- Android 8.1 (Oreo)
- Snapdragon 845 CPU
- Bluetooth 5.0 (BLE Enabled)

4.3.2 Software

The Android Application will be produced using the Xamarin Forms Framework from Microsoft. The other option considered was using the Kotlin Language from Google in Android Studio. As Kotlin is a language designed consciously for writing Android Applications this seemed like a good option. However as Xamarin Forms is written in C#, I chose this method due to my experience in C/C++.

The main functions of the Application are:

- Connect to the Embedded Device over BLE
- Display information about the device
- Control the basic functions of the Embedded Device
- Download data from the Embedded Device
- Process the data received
- Display the data in a clear form

Pages

The Xamarin Forms model uses different "Pages" which are effectively different screens/activities. For my application I will have a number of different pages incorporating different functions. The flow diagram (figure 4.4) below shows a high level view of the different pages planned, and how they are linked.

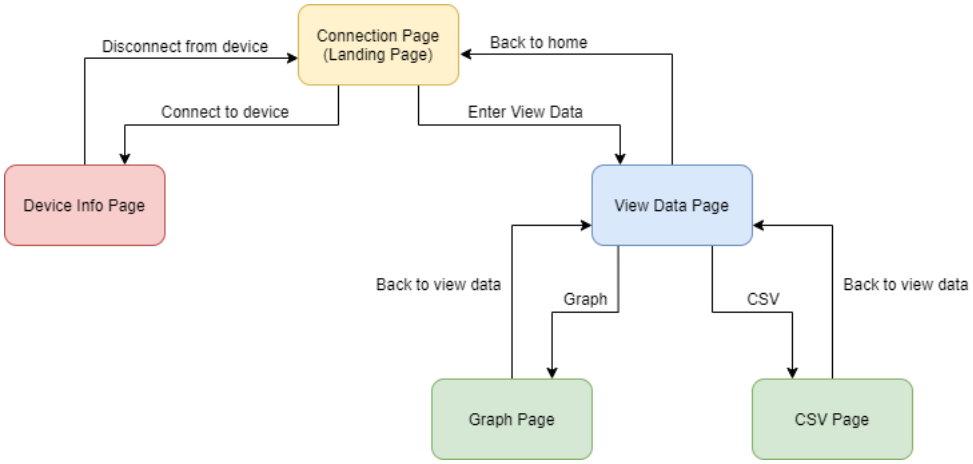


Figure 4.4: Android Application Flow Diagram

When the App. is launched the landing page will be displayed, which will be the "Connection Page". This page contains buttons to launch different pages such as the "View Data" Page, as well as a list of current Bluetooth devices currently detected. By selecting the Embedded Device in the list, the Android device will connect and launch the "Device Info" Page. The page will also include a "Scan"

button which will initiate a BLE scan and update device list.

The "Device Info" Page will handle all interactions with the Embedded Device directly. Firstly it will display information about the Device such as the BLE address, and whether the device is currently logging, and will allow this Logging State to be toggled. There will also be a button that will initiate the download function between the Embedded Device and Android Device. The final function of this page is to process the received data and store the results. Pressing the Android back button will return to the "Connection" Page and disconnect from the device.

The "View Data" Page acts as a menu downloaded data. From here there will be a list of datasets that have been downloaded to the device, these will be labelled by the time period they correspond to. With each dataset there will be three accompanying buttons, one to delete the dataset, one to view the data in the "Graph" Page, and one to view the raw data in a "CSV" Page. A back button press will return to the "Connection" Page. When the "Graph" Page is selected for a specific dataset, this will launch a page with a graph elevation state, against time, giving a brief summary of the readings. Scrolling down will move to more detailed graphs such as elevation angle over time. Exiting this page through a back button press will return to the "View Data" Page.

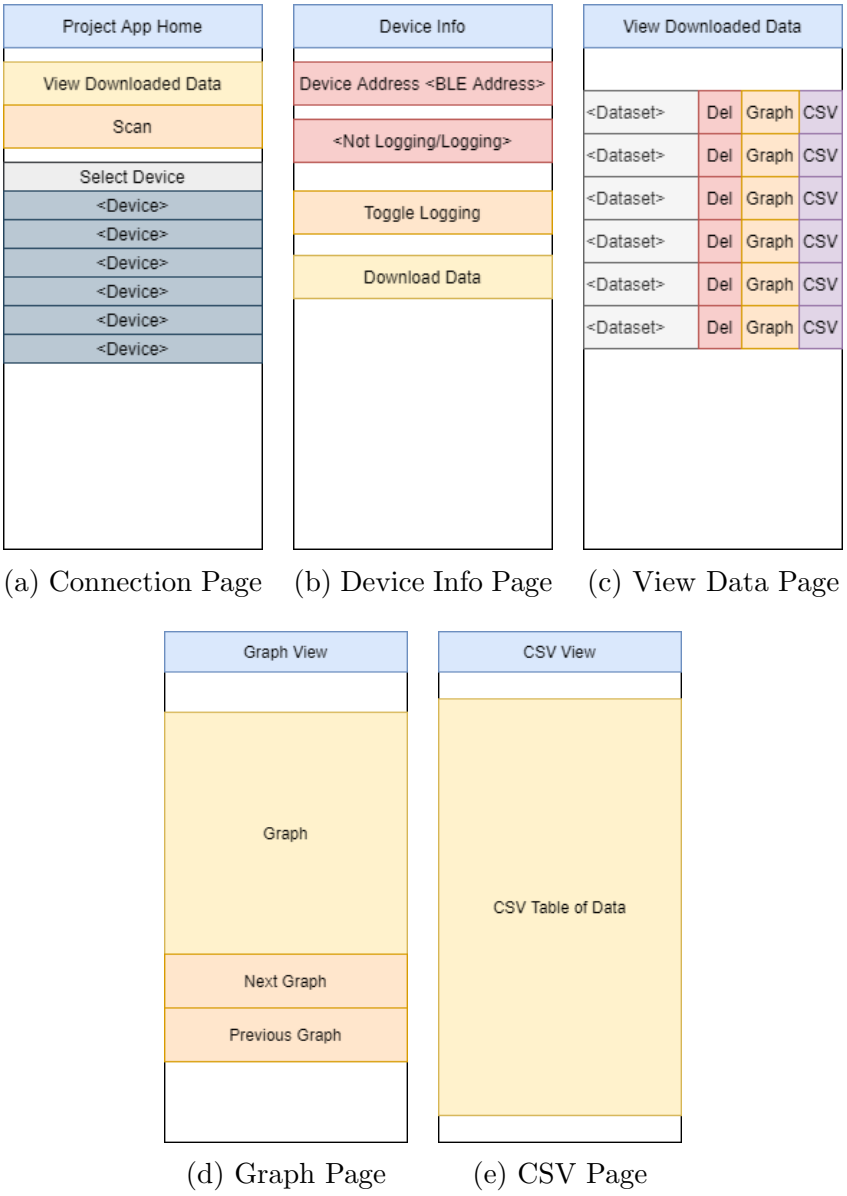
The "CSV" Page, when selected for a particular dataset will show a table of the raw CSV data (after processing) for a more detailed analysis. Exiting this page with a back button press will similarly return to the "View Data" Page. The basic layout for each Page can be seen in Figure 4.5, which shows the locations of buttons, lists, titles, and other elements.

Data Storage

Data received over the BLE connection will need to be stored on the Android device in a sensible manner, as will the results from processing original data. To do this I will use CSV files, one for each dataset, and one for each corresponding processed dataset. This will require permissions to access the storage on the Android device, as the CSV files will be saved directly to the device filesystem, and read by the App later. This makes it much easier to extract the data to a laptop for further analysis if necessary, and eases troubleshooting.

Algorithms

Once the data is received over BLE, it will be processed by the Android App to determine at what points the leg is elevated. This will likely be done using the Madgwick's MARG Algorithm [11] to stabilize the raw IMU data and convert it into a usable form. Large changes in the angles given will then be used to determine leg elevation.



(a) Connection Page (b) Device Info Page (c) View Data Page

(d) Graph Page (e) CSV Page

Figure 4.5: Layouts of the Application's different Pages

4.4 BLE Protocols

The Embedded Device and the Android App are mostly independent, but are connected through a BLE connection for a number of functions.

4.4.1 Control

The first function of the BLE connection is to control the Embedded Device using the App. This will be done using a set of Characteristics within an "Info" Service, using these the logging state of the device will be readable by the App, which can then modify this state. The number of readings will also be visible.

4.4.2 Data Download

A set of BLE Characteristics on a "Transfer" Service will be used to control the download of data. Modifying these through the App will initiate the Transfer Protocol. Through a set of Characteristics used as flags, the App and Embedded Device will signal when data has been read by the App., or sent by the Embedded Device. Between each confirmation, the characteristics below will be updated:

- Accelerometer X Value
- Accelerometer Y Value
- Accelerometer Z Value
- Gyroscope X Value
- Gyroscope Y Value
- Gyroscope Z Value
- Magnetometer X Value
- Magnetometer Y Value
- Magnetometer Z Value

The data will be transferred reading by reading, until the all data is transferred which will be signalled with a characteristic. The time of the first log will also be transferred, and using the logging rate the time of each reading can be determined. This is sufficient and reduces the data transferred.
Services and Characteristics to be used are as such:

Info Service

- Logging State
- Number of Readings
- Year
- Month
- Day
- Hour
- Minute

Transfer Service

- Download Requested

- Packet Sent
- Packet Read
- Transfer Complete
- 9 Characteristics for each value in a reading (see list above)

Chapter 5

Implementation

This chapter will describe the work that was carried out to meet the design, and requirements laid out above.

5.1 Embedded Device Development Environment

In order to program and develop for the Arduino Nano 33 BLE, there were two main options. As the board supports ARM's Mbed OS, which I wanted to use features of, using Mbed toolchain was a compelling choice, particularly the Command Line Interface as it could be run offline, and was feature rich. However when using the Mbed CLI I could not detect the Arduino as a target for programming the second option was explored.
This was the Arduino IDE, which was what this board was designed to use. I initially didn't choose the Arduino IDE due to its perceived support of Mbed OS, and some Mbed functions would clash with those in the Arduino libraries. However the Arduino IDE libraries were extremely useful for this project and easy to include. As the BLE and IMU libraries were designed to work with this specific board, there were no issues including them.

5.2 Embedded Hardware Testing

Before I started to build the Embedded System, testing had to be done on each component to evaluate its functionality. This would allow me to modify my design at this point if a particular component didn't function as envisioned. Carrying out this process prevented such issues from being discovered later on, when their impact the project will be much larger.
Before a PCB was designed for the device, and components were soldered to it, I first tested whether they would work together. To do this I prototyped my system on a breadboard.
Using a DIP version of the memory chip, a CR2032 Breakout Board, and the Arduino Nano 33 BLE with headers soldered on, I setup a breadboard prototype as shown in figure 5.1. By programming the Arduino with a blink sketch, and

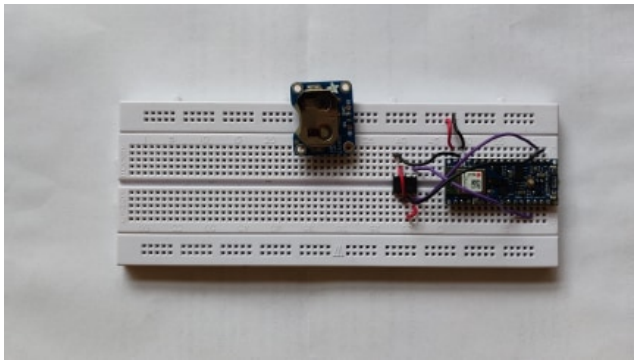


Figure 5.1: Breadboard prototype of embedded system

running it solely on the battery I confirmed that these components worked fine. The memory chip could not be confirmed as working initially, without some form of library.

5.3 IMU Sampling

The lowest level task for the Embedded System was sampling the raw accelerometer, gyroscope, and magnetometer measurements. The IMU of the Arduino Nano 33 BLE is the LSM9DS1, which has an Arduino library with all the required functionality.
During setup, an function needs to be called to setup the IMU, this involves calling the function "IMU.begin" and checking for errors. The sampling process then is very simple, involving calling the "¡sensor¿Available" function for each of the 3 sensors in the IMU (e.g. "accelerationAvailable"), and if this returns no error the function "read¡sensor¿" is called for that sensor. This process is run on a loop. A typedef struct (listing 5.1) is defined to store the readings from the current sample, giving the IMU functions a place to store the values it reads from the IMU itself. An instance of this is declared globally.

```
typedef struct {
  //accel
  float ax;
  float ay;
  float az;
  //gyro
  float gx;
  float gy;
  float gz;
  //mag
  float mx;
  float my;
  float mz;

} SensorData;
```

Listing 5.1: Typedef Struct to Store Readings

5.3.1 Sampling Rate

The rate with which the IMU is sampled is determined by a combination of factors. Sampling at the lowest rate that still provides ample information to determine leg elevation is optimal, as this will reduce the processing required in sampling. It will also allow the storage of more samples in the flash memory IC and thus allow the device to log for longer periods without reliance on an Android Device. Furthermore reducing the sample rate will reduce the time to transfer data over BLE for a period of time measured.
The advantage to increasing the sample rate however is an increase in accuracy of calculated leg elevation. Once the data is sampled, it will be processed by Madgwick's MARG Algorithm [11]. One of the advantages of this algorithm is a claimed high level of accuracy at a low sample rate, which fits the requirements of this application well. Madgwick's MARG algorithm claims a "static error $< 2°$ and dynamic error $< 7°$ while sampling at 10 Hz" [11]. For this reason 10Hz was chosen as the sample rate.

To achieve this sample rate, the sampling function was run as in a thread. After each sample the thread was put to sleep for 100ms. As taking the samples is computation inexpensive, and takes little time, this should achieve a sample rate extremely close to 10Hz, and due to the use of a sleep state, power will be saved.

5.3.2 Logging Control

As the device is not set to continuously log, there needs to be a way to pause/resume logging. This is done by an if statement checking the "loggingState" variable, and will only take readings if this is true. The loggingState may be modified by the BLE code.

5.4 Flash Memory

The next distinct section of the system is the Flash Memory and its use in conjunction with the data logging process.

5.4.1 Library and Testing

Initially to confirm that the SST25VF080B Flash Chip worked as expected, I connected its DIP package up to the Arduino on a breadboard. The Arduino Nano 33 BLE pins 11, 12, and 13 are reserved for MOSI, MISO, and SCK respectively, which forms the SPI interface. I then used pin 10 as the Slave Select line.
In the Arduino Library Manager there are a number of libraries for use with SPI memory chips. Looking at these, many seemed imperfect for my setup, due to setup or SPI commands. Most of these also an excess of functions not required for my project, therefore I chose to write my own library for this chip.

The library created used the "SPI.h" library for SPI commands and allowed me to interface with the chip and perform more application specific functions. The first was to setup the memory chip itself, this involved setting up the pins on the Arduino and setting the WRSR (Write-Status-Register) up correctly, by sending "06H" to turn on write enable and allow the WRSR to be modified. Then a "01H" followed by a "00H" set the correct bits in the WRSR to allow the chip to be written to.
The other functions included are to wipe the chip, and to read and write to bytes. For the read and write byte functions, the flash chip works by first receiving the read/write SPI command, and three address bytes, followed by the data byte on the write command. For the read command, the data byte returned by the chip and this is returned from the function. Each functions has 3 address byte arguments to allow for easier cycling of addresses.
These functions were then tested by iterating between 20 addresses, writing a number between 1 and 21 to each. These same addresses were then read printed to the serial console. The erase was then tested and the value of the addresses printed again.

5.4.2 Memory Organisation

In order to store as many readings as possible on the Flash Chip, and thus allow the device to log for as long as possible, the readings must be stored in an efficient manner. The organisation of the memory is as shown in table 5.1. The

Memory Address	**Contents**
0x000000 - 0xFFFFF8	IMU Readings (in 9 Byte sections)
0xFFFFF9 - 0xFFFFFB	Number of Samples
0xFFFFFC - 0xFFFFFF	Start Date/Time

Table 5.1: Memory Organisation

IMU Readings are arranged with one byte for the X, Y, and Z values from the Accelerometer, Gyroscope, and Magnetometer. This 9 Byte pattern is repeated for each reading. At the end of the memory address range are the Number of Current samples taken, and the Date/Time that sampling began. The number of samples is simply a 24bit number, which has 2^{24} (16777216) possible values, which is sufficient for the number of samples this memory chip can hold (1864134).
The start date/time is stored to allow the time of each reading to be worked out using this, the known sample rate, and number of a specific reading. This is encoded into 4 bytes to make it as compact as possible, each part only having the required bits for its full range: For example, with 2^5 (32) possible values, 5 bits is sufficient for the day of the month, and if the year is begun at 2020, the the 5 bit field encodes up to 2052. The start time only needed to be precise down to the minute level for this application, as reading time is not critical, but is tagged along to give context of when a reading was taken.

Field	Year	Month	Day	Hour	Minute
Bits	[0 - 4]	[5 - 8]	[9 - 13]	[14 - 18]	[19 - 24]

Table 5.2: Data/time memory fields

5.4.3 Logging Process

During the logging process, memory functions are used to read and write the memory sections as described in table 5.1. First of all the "Start Date/Time" and "Number of Samples" sections have a read function and a write function each. There functions use a global variable to hold the value of the Number of Samples (in a uint32_t) and Start Date/Time (in a typedef struct of uint8_ts). This is to make these variables visible to other parts of the code. To write to memory, a combination of bit shifting and bitwise ORs are used until they are encoded in to the corresponding number of bytes, these bytes are then written to the designated memory address. To read the values from memory, the designated bytes are read and a reverse process of the bitwise operations is carried out.

```
160  void writeReadings(SensorData readings)
161  {
162    //start at byte 7
163    uint8_t add1, add2, add3;
164    if(numReadings > 1864121) loggingState = 0;
165    else
166    {
167      uint32_t memPos = numReadings*9;
168
169      //accel
170      add3 = memPos & 0b00000000000000000000000011111111;
171      add2 = (memPos >> 8) & 0b00000000000000000000000011111111;
172      add1 = (memPos >> 16) & 0b00000000000000000000000011111111;
173      memWriteByte(add1, add2, add3, (round(readings.ax*31)+128));
174      memPos += 1;
175      add3 = memPos & 0b00000000000000000000000011111111;
176      add2 = (memPos >> 8) & 0b00000000000000000000000011111111;
177      add1 = (memPos >> 16) & 0b00000000000000000000000011111111;
178      memWriteByte(add1, add2, add3, (round(readings.ay*31)+128));
179      memPos += 1;
180      add3 = memPos & 0b00000000000000000000000011111111;
181      add2 = (memPos >> 8) & 0b00000000000000000000000011111111;
182      add1 = (memPos >> 16) & 0b00000000000000000000000011111111;
183      memWriteByte(add1, add2, add3, (round(readings.az*31)+128));
184
```

Figure 5.2: Example of bitwise operations to write to memory

Due to the way readings are stored as floats, and are not directly compatible with the byte type in memory, reading and writing is more complex. To determine the address that readings are stored in, the "Number of Readings" is multiplied by 9 to get the start address, and the next 9 bytes written to. The "Number of Reading's is then incremented by one so that the next set of readings will be at the correct address.
To solve this issue, I converted to the floats to bytes by multiplying the value by a number that meant its full range would be equal or close to 128 (half of 256 or 2^8), as readings could also have the same magnitude but be negative. The value would then be rounded, and added to 128 to make sure that the value was

positive.
For each sensor this meant deciding what multiplier to use. As low acceleration values were expected, I limited the magnitude of the accelerometer readings to below the sensor's full range to increase the resolution. As I did not expect acceleration values over $\pm 4g$ I used a multiplier of 31 which limited my reading to $\pm 4.13g$. On the read command a conversion back to float was not included due to the fact that I planned to sendbytes over the BLE interface, and so the conversion would happen on the Android Device. To allow this the read command would return a typedef struct similar to that used for the write command, in which each value was of the uint8_t type.

Every time the IMU sampling thread samples the IMU, the readings are written to memory using the described function, along with the Number of Readings in case of a power cycle.

5.5 BLE on the Embedded Device

The last component in the embedded system is the Bluetooth Low Energy Interface. The use of this is to allow control of the logging of the Embedded Device and to facilitate transfer of the data it collects.

The Arduino IDE Library Manager includes a BLE library for Arduino boards, which makes using the BLE on the Arduino Nano 33 BLE very easy, the documentation is also extensive and makes working with this library much simpler.

5.5.1 Services and Characteristics

BLE uses Services, with associated Characteristics to communicate between the Central device and Peripheral device. For this application I used two Services, an "Info" Service to contain general control Characteristics for the Embedded Device, and a "Transfer" Service to contain Characteristics required for the data transfer process. The list of Services and Characteristics used is in listing 5.2.

```
1  //Device Info Service
2  BLEService service_deviceInfo(Info_ServiceUUID);
3
4  //Device Info Characteristics
5  BLEByteCharacteristic char_Logging(Logging_CharUUID, BLERead |
       BLEWrite | BLEIndicate);
6  BLEByteCharacteristic char_Readings(NumberOfReadings_CharUUID,
       BLERead);
7  BLEByteCharacteristic char_Year(Year_CharUUID, BLERead);
8  BLEByteCharacteristic char_Month(Month_CharUUID, BLERead);
9  BLEByteCharacteristic char_Day(Day_CharUUID, BLERead);
10 BLEByteCharacteristic char_Hour(Hour_CharUUID, BLERead);
11 BLEByteCharacteristic char_Minute(Minute_CharUUID, BLERead);
12
13 //Data Transfer Service
```

```
14 BLEService service_dataTransfer(Transfer_ServiceUUID);
15
16 //Data Transfer Characteristics
17 BLEByteCharacteristic char_DownloadRequest(
      DownloadRequest_CharUUID, BLERead | BLEWrite);
18 BLEByteCharacteristic char_PacketSent(PacketSent_CharUUID,
      BLERead | BLEWrite | BLENotify);
19 BLEByteCharacteristic char_PacketRead(PacketRead_CharUUID,
      BLERead | BLEWrite | BLENotify);
20 BLEByteCharacteristic char_TransferComplete(
      TransferComplete_CharUUID, BLERead | BLEWrite | BLENotify);
21
22 BLEByteCharacteristic char_AccelX(AccelX_CharUUID, BLERead);
23 BLEByteCharacteristic char_AccelY(AccelY_CharUUID, BLERead);
24 BLEByteCharacteristic char_AccelZ(AccelZ_CharUUID, BLERead);
25 BLEByteCharacteristic char_GyroX(GyroX_CharUUID, BLERead);
26 BLEByteCharacteristic char_GyroY(GyroY_CharUUID, BLERead);
27 BLEByteCharacteristic char_GyroZ(GyroZ_CharUUID, BLERead);
28 BLEByteCharacteristic char_MagX(MagX_CharUUID, BLERead);
29 BLEByteCharacteristic char_MagY(MagY_CharUUID, BLERead);
30 BLEByteCharacteristic char_MagZ(MagZ_CharUUID, BLERead);
31
32 //BLE Device Declaration
33 BLEDevice selectedDevice;
34
35 uint8_t packetRead, packetSent;
36
37 bool bleSetup()
38 {
39   if(!BLE.begin()) return false;
40
41   loggingState = 0;
```

Listing 5.2: Declaration of Services and Characteristics

5.5.2 Setup

In Figure 5.3 the BLE Peripheral/Central model is shown, where the Embedded Device takes the role of Peripheral Device, and the Android Device as Central. As shown, the peripheral device advertises a set of Services, with associated Characteristics. The Central Device can then connect to the Peripheral Device to interact with these services. Some setup is required for the Embedded Device in order to match this model and begin advertising.
Once the BLE device has been initiated correctly using "BLE.begin", a name must be given to the device which it broadcasts under, and the two services must be advertised. The values of each characteristic are initialised, giving them a known value. Logging State and Number of Readings are initialised from the local variables, while everything else is initialised as 0. The rest of the setup involves adding each characteristic to the appropriate service, and advertising the BLE peripheral device and its services.

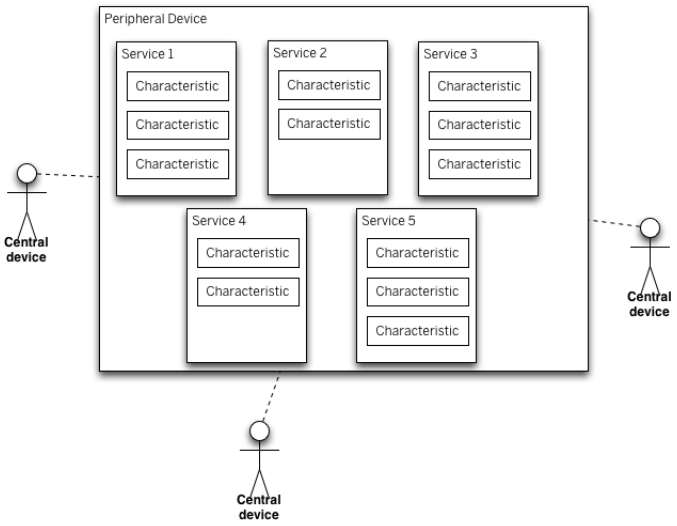


Figure 5.3: BLE Peripheral/Central Model, sourced from [4]

5.5.3 BLE Task

After setup, BLE functionality is run in a thread to prevent interference with the logging process, which is critical and must occur as scheduled, unblocked. This allows BLE to advertise in the background. The thread creates a "BLE.central" object, which will not be initialised until a central device connects to the Peripheral. On connection, there are two main functions that are carried out repeatedly. On each loop the Logging State characteristic is read, and the local logging state is updated, this allows for real time control of the Embedded Device's logging via a connected Central. The "Download Request" Characteristic is also constantly read in order to trigger a data transfer process.
When the download request Characteristic is non zero, the logging process is paused and the data transfer function is called. Upon the Central disconnecting, this loop breaks and the BLE task waits for a new connection.

5.6 Embedded Device PCB

In order to improve the form factor of the device, for final use and to ease development, a PCB was designed and populated t0 form the physical device. The PCB itself was designed in Autodesk Eagle, and is a 2 layer board. It consists of two rows of through-hole pins for the Arduino to slot in to, as well as pads for the SMD Flash Memory chip, and pins for the battery holder. Also included are 4 mounting holes for the PCB to be mounted to an enclosure. The form factor is not much larger than the Arduino itself, and uses flat components in order to create a comfortable form factor to be worn. The Eagle design of the PCB, populated PCB are shown in figures 5.4 and 5.5.

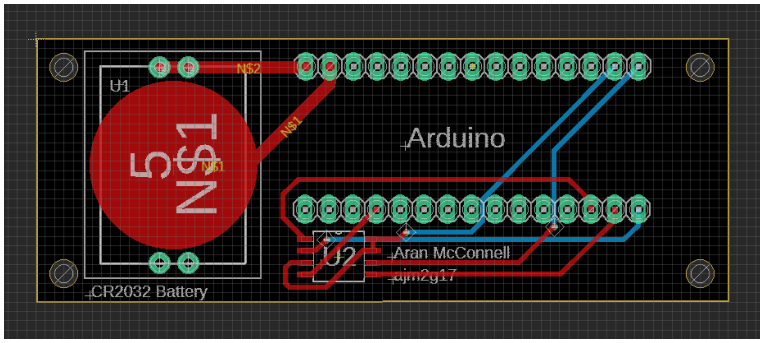


Figure 5.4: Eagle CAD Board Layout

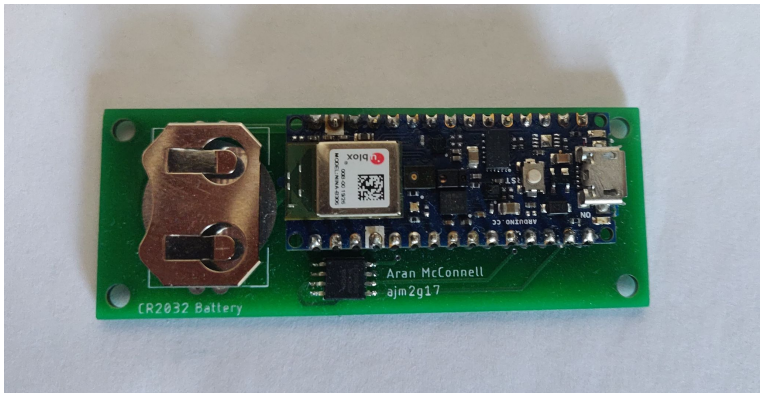


Figure 5.5: Final populated PCB

5.7 Application Connection Page

Using the Xamarin Forms framework, each page has two files associated with it. Firstly a .xaml file which handles the layout, and secondly an .xaml.cs file accompanies this which handles the functionality of the page, and linking certain actions to UI elements specified in the .xaml.

5.7.1 Layout

The layout for this page is defined in ConnectionPage.xaml, and follows the design laid out in figure 4.5a. Once the title has been set, the Stack Layout object is utilised with three elements. Firstly two buttons to view data and scan, and then a listview object. This listview will have title of "Select Device" and then will display the Name and Address of the Bluetooth devices that it is linked to.

5.7.2 Functionality

The connection page mostly deals with connecting to the peripheral BLE device. To enable BLE on the Xamarin Forms platform I used the "Plugin.BLE" pack-

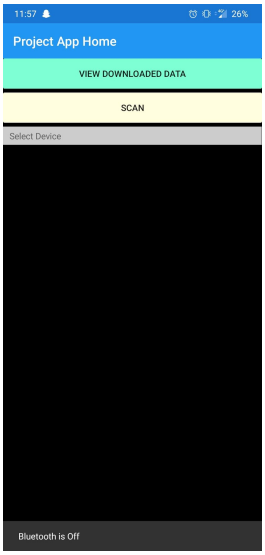
Figure 5.6: The Connection Page running on an Android Phone

age [1] from the NuGet Package Manager.
The Bluetooth setup process using this package was relatively straightforward. Using the "CrossBluetoothLE.Current" class from this package, it was possible to setup a BLE object and an adapter object. At the beginning an ObservableCollection of Bluetooth Devices was declared, as well as an object for the selected BLE Device.
When a instance of the Connection Page is loaded (e.g. on App. launch), a number of things happen. Firstly two events are added to the DeviceDiscovered and DeviceConnected triggers on the adapter. The DeviceDiscovered event will update the list of devices, while DeviceConnected is for debug. The List of devices is then linked with the listview in the layout, which sets this list of devices as it's content. Finally the function "TryScanning" is called, meaning, that upon launch a scan for devices is automatic.
To prevent all Bluetooth Devices in the area appearing in the list, and clogging it up, there is filter which will only add a device to the list if it's name is "Data Transfer", which is the broadcast name specified on the Embedded Device.

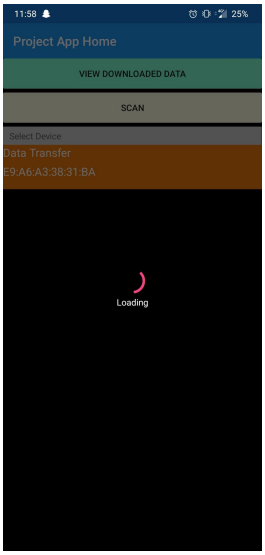
When a device in the list is tapped, this triggers a task which connects to the device using the adapter, and then if the connection is successful, a new instance of the Device Info Page will be created and pushed to the screen. This Device Info page instance will be passed the current adapter and current connected device as parameters so they can be used in that page without them being setup again.

A couple of other packages were used on this page for other features. As BLE re-

quires Android location permissions to be used, I used the Plugin.Permissions [8] package from the NuGet Manager to prompt for location permissions to be allowed upon the App.'s launch. I also used this package to prompt for storage access permissions, as other parts of the App. will require this. The final package used here was Acr.UserDialogs [2], which allows for graphical prompts to make usage of the App. smoother. I used this to show a pop-up message (or toast) with the text "Bluetooth is off" when the App. tried to scan with Bluetooth disabled (Figure 5.7a). I also used this package to add a loading graphic when switching between pages (Figure 5.7b) and carrying out other tasks in the background, this gave user feedback that made it clear the App. was functioning.



(a) Pop-up when Bluetooth is off



(b) Loading Graphic

Figure 5.7: UserDialog Functions

5.8 Android Device Info Page

5.8.1 Layout

After the Screen title "Selected Device", the layout of this page consists of two labels, and two buttons. The Labels show the Device Name and the logging state respectively. To do this they both display variables that are linked to the .cs code so can be altered during runtime, they are not set values.
The first button is "Toggle Logging State", used to be able to pause and resume the logging on a connected Embedded Device. The last button then is to "Download Data", and will trigger the data transfer process between the Embedded Device and the Android Device.

(a) Screen on launch

(b) After logging is toggled

Figure 5.8: Device Info Page Layout

5.8.2 Functionality

The functionality of this page is very simple, with the download process being an exception. Using the Device and Adapter objects passed to the page as arguments, the device's name can be found a displayed. The logging toggle button works by, first reading the "loggingState" characteristic on launch and setting the text and colour as appropriate. Then on press, the characteristic is written to with the opposite value and the text updated as such. This write will be detected by the Embedded Device which will respond appropriately.

5.9 Data Transfer Process

The data transfer process is a protocol that transfers measurement data stored in the Flash memory of the Embedded Device, and transfers it over BLE to the Android Device. Therefore it requires both devices to follow a strict process.

5.9.1 Embedded Device Perspective

In the transfer process, the Embedded Device initiates each transfer and as it is the data provider, it takes control of the transfer. The process works by sending each reading one by one, using acknowledgements to control the data flow, forming an asynchronous transfer protocol.

The transfer occurs in a loop, which iterates from 0 up to the number of readings stored on the device. When it initiates, it firstly waits for the "Packet Read" characteristic to be set to 1 (on the initial iteration this will already be set to 1). This works as an acknowledgement from the Android Device that the previous

packet has been received. Once Packet Read is set to 1, it is set back to 0 to prevent skipping to the next reading prematurely.

```
1
2  for(int i = 0; i < numReadings; i++)
3  {
4
5    while(!char_PacketRead.value())
6    {
7      Serial.println(char_PacketRead.value());
8    }
9    char_PacketRead.writeValue(0x00);
10
11   packetRead = 0;
12   packetSent = 0;
13
14   Serial.print("Transfer number: ");
15   Serial.println(i);
16   SensorData_bytes temp;
17   temp = memReadPacket(i);
18   char_AccelX.writeValue(temp.ax);
19   char_AccelY.writeValue(temp.ay);
20   char_AccelZ.writeValue(temp.az);
21   char_GyroX.writeValue(temp.gx);
22   char_GyroY.writeValue(temp.gy);
23   char_GyroZ.writeValue(temp.gz);
24   char_MagX.writeValue(temp.mx);
25   char_MagY.writeValue(temp.my);
26   char_MagZ.writeValue(temp.mz);
27
28   char_PacketSent.writeValue(0x01);
29   packetSent = 1;
30   Serial.println("Packet Complete");
31 }
```

Listing 5.3: Loop to control data transfer on Embedded Device

Next the corresponding reading is read from memory, and written to the Accel_X ... Mag_Z characteristics, the "Packet Sent" characteristic is then set to 1 to signal to the Android Device that the packet has been sent. The process then repeats if there is more data to be sent. Once all readings have been transferred, the "Transfer Complete" characteristic is set to 1 to signal this to the Android Device. Finally the memory is fully erased and number of readings set to 0, ready to take take readings again.

5.9.2 Android Perspective

On the Android Device end of the process, once the required characteristics are set up, the process is triggered by setting the "DownloadReq" Characteristic to a 1.
The App then waits for the "PacketSent" Characteristic to be set to 1, to signal that the reading characteristics have been populated with the current reading. These characteristics are then read and the "PacketRead" characteristic set to 1.

This process then repeats until the "TransferComplete" characteristic has been set to 1, on which it will exit the process and normal state of the Device Info page will resume.

When testing the transfer process however, a number of issues were encountered. The main one involved the Android App being unable to resolve the value of the BLE characteristics controlling the transfer such as "PacketSent" and "PacketRead". Instead they would be read as NULL, which would then cause an exception to be thrown and the App. to ultimately crash. One reason for this could be due to the two different devices trying to read/write to the characteristic simultaneously, however I could not solve this issue which left me stuck at this point. Without the ability to transfer data to the Android App it was difficult to proceed.

5.10 Data Processing

A large part of the project involves taking the raw IMU data, and processing it in order to determine leg elevation over time. In order to do this, I wanted to use the Madgwick's MARG algorithm [11].

The Madgwick MARG algorithm comes in the form of both a C library, and a C# library from an x-io Technologies page [14]. The header for the C library is showing in figure 5.9. My original implementation put the processing stage using this algorithm on the Embedded Side, where sampled data was run directly through the MadgwickAHRS (Attitude and heading reference system) Algorithm. The quaternion result would then replace the data stored in memory, and then be transferred to the Android Device for further processing. When testing this

```
13  #ifndef MadgwickAHRS_h
14  #define MadgwickAHRS_h
15
16  //---------------------------------------------------------------------------------------------------
17  // Variable declaration
18
19  extern volatile float beta;             // algorithm gain
20  extern volatile float q0, q1, q2, q3;   // quaternion of sensor frame relative to auxiliary frame
21
22  //---------------------------------------------------------------------------------------------------
23  // Function declarations
24
25  void MadgwickAHRSupdate(float gx, float gy, float gz, float ax, float ay, float az, float mx, float my, float mz);
26  void MadgwickAHRSupdateIMU(float gx, float gy, float gz, float ax, float ay, float az);
27
28  #endif
```

Figure 5.9: Header file for MadgwickAHRS algorithm

algorithm in a Embedded Setting, I set the sample rate to 10Hz, based on section 5.4 in Madgwick's Report [11], which discusses Sample Rate vs Performance. The report suggests there is no benefit of samples rates over 20Hz for this algorithm, and that a 10Hz sample rate would lead to low static errors of $<2°$. β, the filter gain was then set at 0.041, as section 5.3 of the report demonstrates this as being the best trade-off between static and dynamic error.

Despite this parameter tuning, the quaternion result returned from the algorithm drifted extremely rapidly when the device was laid at rest, and despite trying different sample rates, this issue could not be fixed. For this reason I decided to move all processing to the Android Device so that the embedded section of the project could be completed. The C# library also came with a example project and I hoped this would prove useful.

Chapter 6

Testing

In this chapter I will describe the testing process for the individual parts of my system, as well as the system as a whole. The system will be evaluated against the project requirements set out in Section 3.2. I will also discuss barriers that arose which prevented testing that was planned.

6.1 Embedded System Testing

6.1.1 IMU Logging

The first requirement defined for the Embedded Device was to "Record raw 9 axis IMU data at a set logging rate". To evaluate this I modified the program to print logged data to the serial console on each log. Then when the device was reset I monitored the serial console and timed how long it took to get 20 readings. As seen in figure 6.1, 9 axis readings are taken, and the measured rate (averaged over 20 readings) was 10Hz as specified, and thus this requirement was met.

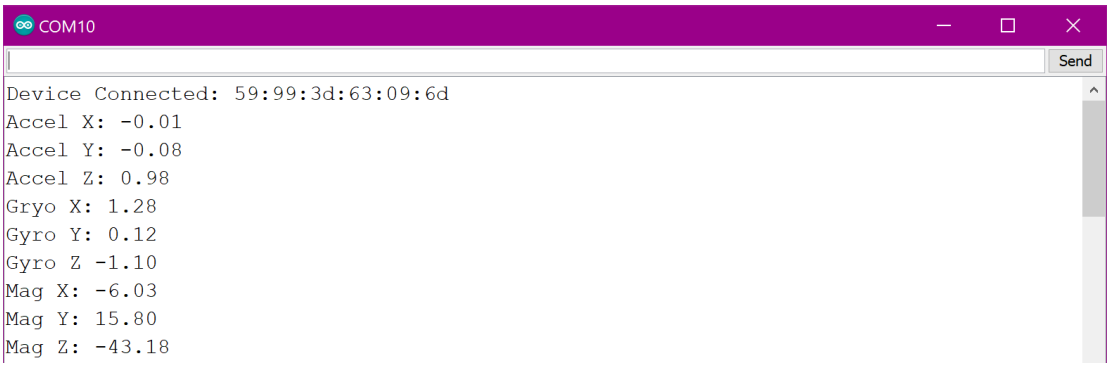


Figure 6.1: Serial console showing readings during logging

6.1.2 Memory Storage

In order to test whether reading were stored in memory correctly, I allowed the device to log for a few minutes, long enough to take a large number of readings.

The logged values were printed to the console and recorded. Then the device was reprogrammed with code that read the first 50 addresses and printed them to the console. Comparing these with the results allowed me to verify that this worked as expected. This test demonstrated that the device met requirements 2 and 3, to record without a smartphone, and to store an acceptable number of readings locally. The acceptable number defined here is considered met as the memory chip used was calculated to store such a number of readings using the defined storage scheme.

6.1.3 BLE

One of the requirements is that the Embedded Device be able to "Communicate with the Android App over BLE". Specifically being able to communicate with the the App. will be addressed as part of the Android requirements, however using a generic Bluetooth App. such as "nRF Connect", the services and characteristics of the Embedded Device can be inspected and are as expected, which confirms that the BLE interface is configured correctly.

6.1.4 Miscellaneous

The final two requirements for the Embedded Device are that it be powered by a CR2032 battery, and have a small footprint. The device itself can easily be proven to be powered by the CR2032, by inserting one and using the nRF Connect App to inspect that the BLE interface is advertising as expected. The battery life however cannot be estimated due to lack of access to a high accuracy ammeter (or power analyser). Due to the low current drawn by the device, such specialist equipment is required to obtain an estimate.
The footprint of the device, based on the PCB meets the requirements as it is even smaller than expected, being not much bigger than the Arduino itself at 74mm x 28mm. The Device was planned to have a 3D printed enclosure around it, however this was never completed due to University closure.

6.2 Android Device

6.2.1 BLE

The first two requirements set out for the Android Device relate to BLE functionality, the first being the ability to start/stop logging on the Embedded Device using the Application. This was tested using a similar method to the embedded logging tests, by setting up the Embedded Device to print logs to the serial console. With the App. connected, initially nothing was printed to the console, but when logging state was toggled via the App., the console started printing readings. The logging state could also be paused in this way. Therefore this requirement was met.

The next requirement, for the data transfer process to work over BLE, was not met, as described in section 5.8. Testing of this function was limited as testing the process caused the Android App to crash and very little could be determined from this. The failure to meet this requirement severely limited the completion of a number of following requirements.

6.2.2 Other requirements

There are three more requirements set out for the Android Application:

1. Process the received data to determine at what points the leg was raised
2. Display the downloaded data in a clear and user friendly manner
3. Store the multiple sets of readings and have them easily differentiable from one another

Due to limitations imposed by not completing the data transfer requirement, none of these requirements could be completed either. Whilst some components of these tasks were tested, such as storing different sets of readings in different CSV files, progress was not made on them due to project time being spent on trying to complete previous requirements, leaving no time for these. These requirements also required the previous to be completed fully.

6.3 Full System Test

Due to the incomplete state of the project, a full system test was not possible. The most complete test that could be carried out was by capturing data on the Embedded Device (with logging controlled by the Android App), and the data extracted from the Embedded Device for analysis.

It was then possible to carry out this test, but with the device being used by a number of participants as part of a study to collected data. The data collected from this could be used to help tune processing algorithms to make the device more reliable in elevation angle determination. To carry this out, an ethics application was produced, and approved through the University ERGO system. The ERGO documents for this application can be found in Appendix A.

Chapter 7

Project Management

In order to manage my project and ensure that, throughout the process, I was on track to completion, I carried out a number of project management tasks.

7.1 Skills Analysis

I used an analysis of my own skills in order to highlight my strengths and therefore pinpoint areas which would take longer due to involving weaker skills.

Skill	**Rating (0 - 5)**
Writing	4
Researching	3
Project Management	3
Critical Analysis	3
C/C++	5
Kotlin/App Development	0
Hardware Debugging	4
Mathematical Knowledge	4

Table 7.1: Skills Analysis Table

7.2 GANTT Chart

At the onset of the Project, I produced a GANTT chart to allow myself to plan out the many different components of the project. This also helped me keep track of my progress.

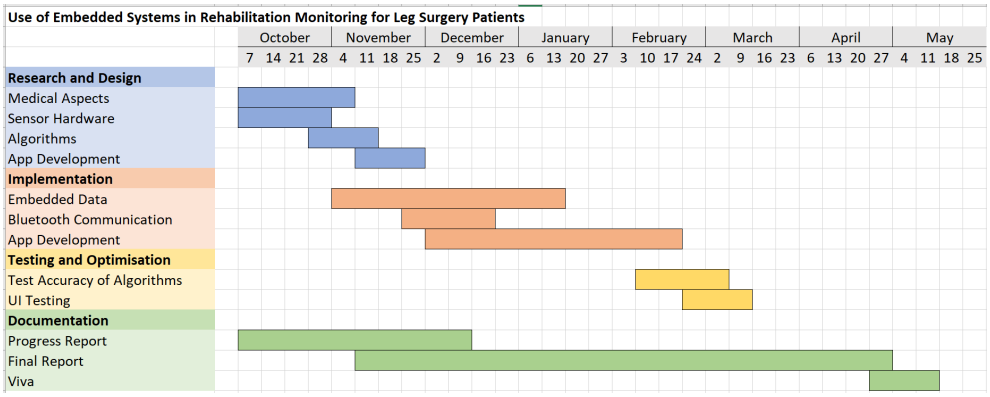


Figure 7.1: Original GANTT Chart

In the last few weeks of the first semester I updated my Gantt chart to reflect the progress I had already made and my updated estimations. Due to the time scale and complexity of the project, many things changed from my initial GANTT chart.

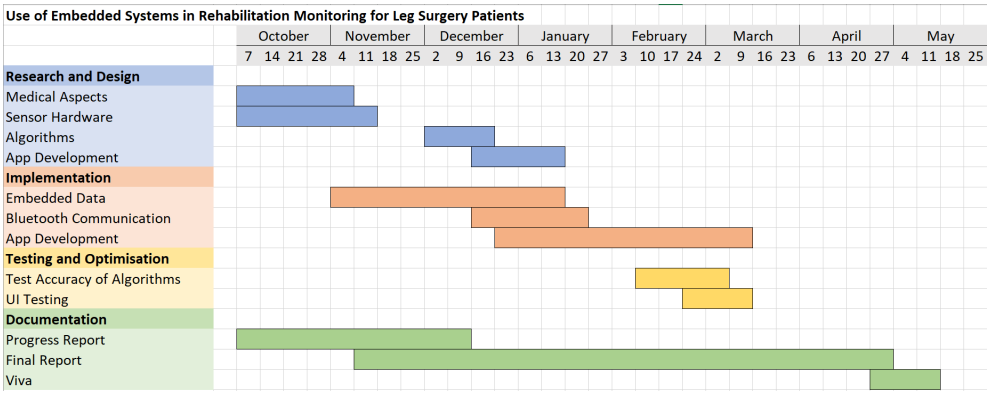


Figure 7.2: Updated GANTT Chart

7.3 Risk Analysis

Risk ID	Risk	Probability (1-5)	Severity (1-5)	Overall Risk (1-25)	Mitigation Action
1	Illness	3	2	6	Allocate extra time after illness to ensure work is completed.
2	Computer Breaking	2	5	10	All work is done in a git repository which is regularly pushed to the University Gitlab server. This way it is also accessible remotely or from another machine
3	Under estimation of work	3	4	12	In this case more time must be allocated to the project, and a meeting with my supervisor will be arrange to discuss possible solutions.
4	Over estimation of work	2	2	4	Extension tasks can be found and the scope of the project extended as required.
5	Lack of programming skill	3	4	12	Tutorials online can be used to improve my programming skills in the required language.
6	Embedded hardware breaks	4	3	12	New hardware will have to be purchased, and while waiting for this to arrive tasks that do not require the hardware will be focussed on.

Table 7.2: Risk Analysis

7.4 Agile Methodologies

As there was no set schedule for this project, and the deadline was far from the beginning, I had to manage it by allocating time each week to work on it, in the following format: This adds up to be 10 hours a week in Semester 1 and 14 hours

Semester Number	**Mon**	**Tues**	**Weds**	**Thurs**	**Fri**
1	5 hours	3 hours	2 hours	-	-
2	-	4 hours	3 hours	4 hours	3 hours

Table 7.3: Work Schedule for Project

in Semester 2, which was appropriate considering the module weightings. This did however change week on week due to commitments in other modules and duties outside on University. It at least served as a rough baseline of how many hours to allocate to the project each week.

I also met with my Supervisor on a weekly basis to discuss any issues I had and to allow them to keep track of my progress to ensure I completed the project in time.

Chapter 8

Evaluation

In this chapter, I will evaluate the overall success of the project, and whether goals and aims were met. I will also address important metrics such as cost, and evaluate my project planning and time management throughout.

8.1 Progress, goals, and aims

The goal of this project was the produce a solution involving a logging device that could log independently, and an accompanying Android App. to control this device and display recorded data in a clear manner.
As discussed in the Testing chapter, many of the requirements for the project were met, however a number were not which means that on a broad scale the goal of the project was not fully met. On the embedded side of the project, almost all aims were met and great progress was made. The project resulted in a very effective Embedded Device, able to take the measurements required independently, in a small, low power, BLE enabled package.
The Android App. on the other hand fell short of many aims. While it was useful for control of the Embedded Device, it fell short in data transfer, data processing, and displaying the recorded data. Failure of these aims was largely due to technical issues that arose, and could not be solved quickly. Better project management may have alleviated many of these issues.

8.2 Cost

The cost in this project was fully in hardware required for the embedded system. As shown in the cost breakdown in Appendix B, the full project cost was £63.95, which is very reasonable for a prototype of the system, as well as a finished device. The main cost driver here was the Arduino, however, due to its essential role in the device this is expected, the rest of the components needed for the system being very low cost.
The unit cost for each device however is £36.63, which is only marginally higher than the Arduino cost alone. This cost was kept down by using very simple

and cheap external components, that only added the required functionality and nothing more. The PCB manufacturer used (JLC PCB) was also a budget option and helped to keep the cost down significantly. The cost could be further reduced by sourcing the Arduino at a lower cost, due it being the largest cost driver.

8.3 Project Management

My project management methodologies are detailed in chapter 7. The first task undertaken was a skills analysis, which was filled out accurately and allowed me to ultimately allocate more time for tasks that I was less knowledgeable about. However using this grid I allocated less time than needed to aspects of the embedded system as, despite my recorded strength in this area, due to its nature many issues came up that could not have been foreseen by such a grid.
My use on GANTT charts were very useful in planning out my project, even if it was really hard to stick to as planned, it gave me an outline of where I should be at in different aspects of the project at any one time. An important decision was that to produce a modified GANTT chart, as I underestimated the time that some aspects of the implementation required, and this updated chart was much more realistic to follow, due to my increased understanding when I produced it. Despite this, I still did not keep to the timescales set out very well, and due to issues with certain areas, others fell behind.
From my list of risks identified in the risk analysis, risks 3 and 5 were encountered. These were managed as planned, through allocating more time to certain aspects of the project, and uses online resources to overcome such hurdles. Risk 6 was also encountered, but had little impact due to a replacement being ordered swiftly.

Throughout the project I stuck to the schedule in section 7.4 reasonably well. This kept me working at a constant and sustainable pace throughout the project, with small deviations week on week. Later on however, when pressure increased, and when issues were encountered, the time spent did increase from this schedule. Meeting weekly with my supervisor and discussing my progress also helped to keep me on track.

Chapter 9

Conclusion

This chapter will focus on concluding the project, summarising what has been achieved, and also serves to discuss what future what could be carried out.

This project has produced a low cost, power efficient, and effective IMU logging device in an attractive form factor. Compared to other solutions, the logging device is able to collect readings independently, without the need for a smartphone due to the flash memory addition. The device also uses BLE and an RTOS for reduced power use.
The App. produced allows the logging on the Embedded Device to be stopped/started, and allows viewing of this state and and the device's address. Whilst some progress was made on other features such as the BLE transfer protocol, and processing data, they were left unfinished and can not be included as features.

9.1 Further Work

In terms of further work, this can be split in to two categories. Planned work that was left unfinished, and extension work that was initially not part of the scope.

The first piece of unfinished work involves completing the BLE data transfer protocol so data can be transferred completely. Next would be completing the data processing stage, to determine leg elevation using a set of algorithms. The rest of the unfinished work is then concerned with building the pages in the application that allow the data to be viewed in graphical and CSV form.
There are also small improvements to be made on the Embedded Device, by designing and building an enclosure for it, as well as improving battery life by using low power states appropriately.

Future work for this project would include possibly adding a step count/analysis function to the App. to help with rehabilitation after surgery, as well as generally improving accuracy, and possibly adding a form of database integration to store data in a sort of "cloud" environment.

Word Count: 9933

Calculated using the TexCount command line tool.

Bibliography

[1] https://www.nuget.org/packages/Plugin.BLE Adrian Seceleanu, Sven-Michael Stübe. Plugin.ble (nuget package manager.

[2] https://www.nuget.org/packages/Acr.UserDialogs/ Allan Ritchie. Acr.userdialogs (nuget package manager.

[3] https://store.arduino.cc/arduino-nano-33-ble-sense-with-headers Arduino Project. Arduino nano 33 ble sense product page.

[4] https://www.arduino.cc/en/Reference/ArduinoBLE Arduino Project. Arduino ble library reference.

[5] P. Baquie and P. Brukner. Injuries presenting to an australian sports medicine centre: A 12-month study. *Clinical Journal of Sports Medicine*, 1997.

[6] Bzarg.com. How a kalman filter works, in pictures.

[7] D. V. Dimitrov. Medical internet of things and big data in healthcare. *Healthcare Informatics Research*, 2016.

[8] https://www.nuget.org/packages/Plugin.Permissions/ James Montemagno. Plugin.permissions (nuget package manager.

[9] Svenja Hucker Olga Scheck Markus A. Hobert Ana Teresa Santos Øyvind Fagerbakke Frank Larsen Joaquim J. Ferreira Walter Maetzler Janet M.T. van Uem, Katrin S. Maier. Twelve-week sensor assessment in parkinson's disease: Impact on quality of life. *International Parkinson and Movement Disorder Society*, 2016.

[10] https://www.pieter-jan.com/node/11 Pieter-Jan. Reading a imu without kalman: The complementary filter.

[11] Sebastian O. H. Madgwick ; Andrew J. L. Harrison ; Ravi Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. *IEEE*, 2010.

[12] B. Wedro. Broken bones (types of bone fractures. *MedicineNet*, 2019.

[13] C. Worsham and A. B. Jena. Why doctors shouldn't dismiss the apple watch's new ecg app. *Harvard Business Review*, 2018.

[14] https://x-io.co.uk/open-source-imu-and-ahrs-algorithms/ x-io Technologies. Open source imu and ahrs algorithms.

Appendix A

ERGO Documents

A.1 Ethics Application Form

UNIVERSITY OF Southampton

FEPS Ethics Committee
FEPS Ethics Application Form Ver 1

Refer to the *Instructions* and to the *Guide* documents for a glossary of the key phrases in **bold** and for an explanation of the information required in each section. The *Templates* document provides some text that may be helpful in preparing some of the required appendices.

Replace the highlighted text with the appropriate information.

Note that the size of the text entry boxes provided on this form does **not** indicate the expected amount of information; instead, refer to the *Instructions* and to the *Guide* documents in providing the complete information required in each section. Do **not** duplicate information from one text box to another. Do not otherwise edit this form.

Reference number: **ERGO/**FEPS**/52959**	Submission version: 1	Date: 2020-03-10
Name of **investigator**(s): Aran McConnell		
Name of supervisor(s) (if student **investigator**(s)): Prof. Neil White		
Title of study: Use of embedded devices in rehabilitation monitoring for leg surgery patients.		

Note that failure to follow the University's policy on Ethics may lead to disciplinary action concerning Misconduct or a breach of Academic Integrity.

By submitting this application, the investigator(s) undertake to:

- Conduct the study in accordance with University policies governing:
  Ethics (http://www.southampton.ac.uk/ris/policies/ethics.html);
  Data management (http://www.southampton.ac.uk/library/research/researchdata/);
  Health and Safety (http://www.southampton.ac.uk/healthandsafety);
  Academic Integrity (http://www.calendar.soton.ac.uk/sectionIV/academic-integrity-statement.html.
- Ensure the study Reference number ERGO/FEPS/52959 is prominently displayed on all advertising and study materials, and is reported on all media and in all publications;
- Conduct the study in accordance with the information provided in the application, its appendices, and any other documents submitted;
- Submit the study for re-review (as an amendment through ERGO) or seek FEPS EC advice if any changes, circumstances, or outcomes materially affect the study or the information given;
- Promptly advise an appropriate authority (Research Integrity and Governance team) of any adverse study outcomes;
- Submit an end-of-study form if required to do so.

20022019

REFER TO THE INSTRUCTIONS AND GUIDE DOCUMENTS WHEN COMPLETING THIS FORM AND THE TEMPLATES DOCUMENT WHEN PREPARING THE REQUIRED APPENDICES.

STUDY DETAILS

What are the aims and objectives of this study?

The aim of this study to test a small embedded measurement device and associated android app. The objectives of this study are to evaluate the accuracy of the device and gather data to be analysed.

Background of the study (*a brief rationale for conducting the study)*

In leg surgeries, the patient is often asked to keep their leg raised for periods before the surgery can be carried out. There is a lack of information whether patients actually do this and how this correlates with recovery. The device in question is to be used to determine whether a person's leg is raised or not, and to measure their steps when walking. It does this by simply measuring is orientation and acceleration in space. The device would then give doctors data on these actions which could be used to develop medical device in this area.

Key research question *(Specify hypothesis if applicable*)

Can the device designed in the project accurately determine when a person's leg is raised and accurately measure their steps?

Study design *(Give a **brief** outline of the study design and why it is being used*)

The study will involve the participant placing a small device inside their sock, or similarly strapped to the ankle. The participant will then be asked to raise the leg with the device attached several times while being seated, this will be done a number of times to different elevations and at different speeds. This will allow data to be captured of the device being used in its desired setting, in different ways to introduce some variance to the data. The data can then be analysed to see if the device is accurate in measuring a raised leg.

The participant will then be asked to walk around at a average and steady pace for a few minutes in order to capture some stepping data, which can be then analysed to see if the device can accurately measure steps.

PRE-STUDY

Characterise the proposed **participants**

The participants will be other students known to the investigator. The only requirement is for them to be able to walk around comfortably and raise their leg while seated, otherwise there will be no other discriminating factors.

Describe how **participants** will be approached
The participant will be asked personally by the investigator as they will be all be known to them.

Describe how inclusion / exclusion criteria will be applied (if any)
Only an ability to physically carry out the simple movements required (See above) is required, there are no other exclusion/inclusion criteria.

Describe how **participants** will decide whether or not to take part
By filling out the supplied consent form fully.

Participant Information (Appendix (i))

Provide the **Participant Information** in the form that it will be given to **participants** as Appendix (i). All studies must provide **participant information**.

Consent Form/Information (Appendix (iii))

Provide the **Consent Form** (or the request for consent) in the form that it will be given to **participants** as Appendix (iii). All studies must obtain **participant** consent. Some studies may obtain verbal consent (and only present consent information), other studies will require written consent, as explained in the *Instructions, Guide,* and *Templates* documents.

DURING THE STUDY

Describe the study procedures as they will be experienced by the **participants**
The study will involve one session lasting between 5 and 10 minutes. During this session the participant will be asked to strap a small circuit to the side of their ankle (either by placing it in your sock or otherwise). They will then be asked to raise and lower the leg with the device, whilst seated. This will take 2-3 minutes and will involve lifting it at different speeds and to different heights. They will then be asked to walk steadily around the room for a couple of minutes whilst still wearing the device.

Identify how, when, where, and what kind of data will be recorded (not just the formal research data, but including all other study data such as e-mail addresses and signed consent forms)
The only data recorded is the research data, this will be collected in a common learning space at the University in the weeks following ethics approval. The participants will be asked to attend, and the testing as described above will be done individually, one at a time. Multiple sessions may be needed to get through 10 participants. The data recorded will consist of accelerometer, magnetometer, and gyroscope measurements from the device which will give its orientation in space at regular time intervals. Signed consent forms will only collect the name of the participant before the study begins.

Participant questionnaire/data gathering methods (Appendix (ii))

As Appendix (ii), reproduce any and all **participant** questionnaires or data gathering instruments in the exact forms that they will be given to or experienced by **participants**. If conducting less formal data collection, or data collection that does not involve direct questioning or observation of participants (eg secondary data or "big data"), provide specific information concerning the methods that will be used to obtain the data of the study.

POST-STUDY

Identify how, when, and where data will be stored, processed, and destroyed
The data will be collected on a phone connected to the device via Bluetooth. The phone is password protected. The data will be later transferred to a password protected laptop to be analysed. The data will be deleted once it has been analysed and discussed in the project report. Consent form will be scanned and stored on the laptop described, and the physical copy destroyed. The data on these forms will not be used in the study and is only to prove that the participant has given consent to take part.

STUDY CHARACTERISTICS

(L.1) The study is funded by a commercial organisation: **No** (delete one) If 'Yes', provide details of the funder or funding agency *here.*

(L.2) There are **restrictions** upon the study: **No** (delete one) If 'Yes', explain the nature and necessity of the **restrictions** *here.*

(L.3) Access to **participants** is through a third party: **No** (delete one)

If 'Yes', provide evidence of your permission to contact them as (v) in the *Checklist* below. Do *not* provide explanation or information on this matter here.

(M.1) **Personal data** is or *may be collected or processed: **No** (delete one)
Data will be processed outside the UK: -**No** (delete one)

If 'Yes' to either question, provide the **DPA Plan** as (iv) in the *Checklist* below. Do *not* provide information or explanation on this matter here. Note that using or recording e-mail addresses, telephone numbers, signed consent forms, or similar study-related **personal data** requires M.1 to be "Yes".

(* Secondary data / "big data" may be *de*-anonymised, or may contain **personal data**. If so, answer 'Yes'.)

(M.2) There is **inducement** to **participants**: **No** (delete one)

If 'Yes', explain the nature and necessity of the inducement *here.*

(M.3) The study is **intrusive**: **No** (delete one)

If 'Yes', provide the **Risk Management Plan,** the **Debrief Plan,** and Technical Details as (vi), (vii), and (ix) in the *Checklist* below, and explain *here* the nature and necessity of the intrusion(s).

(M.4) There is **risk of harm** during the study: **No** (delete one)

If 'Yes', provide the **Risk Management Plan**, the **Contact Information**, the **Debrief Plan**, and Technical Details as (vi), (vii), (viii), and (ix) in the *Checklist* below, and explain *here* the necessity of the risks.

(M.5) The true purpose of the study will be hidden from **participants**: **No** (delete one)
The study involves **deception** of **participants**: **No** (delete one)

If 'Yes' to either question, provide the **Debrief Plan** and Technical Details as (vii) and (ix) in the *Checklist* below, and explain *here* the necessity of the deception.

(M.6) **Participants** may be minors or otherwise have **diminished capacity**: **No** (delete one)

If 'Yes', AND if one or more Study Characteristics in categories M or H applies, provide the **Risk Management Plan,** the **Contact Information**, and Technical Details as (vi), (vii), & (ix) in the *Checklist* below, and explain *here* the special arrangements that will ensure informed consent.

(M.7) **Special category personal data** is collected or processed: **No** (delete one)

If 'Yes', provide the **DPA Plan** and Technical Details as (iv) and (ix) in the *Checklist* below. Do *not* provide explanation or information on this matter here.

(H.1) The study involves: **invasive** equipment, material(s), or process(es); or **participants** who are not able to withdraw at any time and for any reason; or animals; or human tissue; or biological samples: **No** (delete one)

If 'Yes', provide Technical Details and further justifications as (ix) and (x) in the *Checklist* below. Do *not* provide explanation or information on these matters here. Note that the study will require separate approval by the Research Governance Office.

Technical details

If one or more Study Characteristics in categories M.3 to M.7 or H applies, provide the description of the technical details of the experimental or study design, the power calculation(s) which yield the required sample size(s), and how the data will be analysed, as separate appendices.

CHECKLIST OF DOCUMENTS TO UPLOAD

Please provide the following forms, *naming the files as explicitly* as possible, e.g., "Participant Information", "Questionnaire", "Consent Form", "DPA Plan", "Permission to contact", "Risk Management Plan", "Debrief Plan", "Contact Information", and/or "Technical details" as appropriate. Each document must specify the reference number in the form ERGO/FPSE/**52959**, the document version number, and its date of last edit.

(i): **Participant Information** in the form that it will be given to **participants.**

(ii): Data collection method (eg for secondary data or "big data") / **Participant** Questionnaire in the form that it will be given to **participants.**

(iii): **Consent Form** (or consent information if no **personal data** is collected) in the form that it will be given to **participants.**

(iv): **DPA Plan.**

(v): Evidence of permission to contact (prospective) **participants** through any third party.

(vi): **Risk Management Plan**.

(vii): **Debrief Plan**.

(viii): **Contact Information**.

(ix): Technical details of the experimental or study design, the power calculation(s) for the required sample size(s), and how the data will be analysed.

(x): Further details and justifications in the case of: **invasive** equipment, material(s), or process(es); **participants** who are not able to withdraw at any time and for any reason; animals; human tissue; or biological samples.

A.2 Participant Information Form

UNIVERSITY OF Southampton

Participant Information Sheet

Study Title: Use of embedded devices in rehabilitation monitoring for leg surgery patients.
Researcher:
ERGO number:

You are being invited to take part in the above research study. To help you decide whether you would like to take part or not, it is important that you understand why the research is being done and what it will involve. Please read the information below carefully and ask questions if anything is not clear or you would like more information before you decide to take part in this research. You may like to discuss it with others but it is up to you to decide whether or not to take part. If you are happy to participate you will be asked to sign a consent form.

What is the research about?
This study is a part of a Third Year Individual Project as part of a BEng Electronic Engineering degree. I am a third year Electronics Student carrying out this study.
The aim of this study/test is to determine whether the device designed during the project can accurately detect when a person's leg is raised, and whether it can accurately measure steps. This test will hopefully generate readings that can be analysed later on to determine effectiveness of the device and also help tune it to be more effective.

Why have I been asked to participate?
You have been asked to participate as you are known to the investigator personally. There will be around 10 people total in this study.

What will happen to me if I take part?
The study will involve one session lasting between 5 and 10 minutes. During this session you will be asked to strap a small circuit to the side of your ankle (either by placing it in your sock or otherwise).

You will then be asked to raise and lower the leg with the device, whilst seated. This will take 2-3 minutes and will involve lifting it at different speeds and to different heights.
You will then be asked to walk steadily around the room for a couple of minutes whilst still wearing the device.

Are there any benefits in my taking part?
Taking part in this study will benefit my induvial project greatly and contribute to the field of medical electronics.

Are there any risks involved?
There may be a small risk of injury associated with raising or lowering the leg, such as pulling muscles. This also applies to walking. Participants should only take part if they feel fit and healthy enough to carry out the exercises mentioned.

What data will be collected?
The data collected will be in the form of accelerometer data, gyroscope data, and magnetometer data from the device. This will result in an orientation of the device in space being collected at regular time intervals over the course of the test. The data will be collected by the investigator. No personal data will be collected of any kind, the data described above will be completely anonymous.

The data will be connected on a phone connected to the device via Bluetooth, this phone is password protected. The data may later be transferred to a password protected laptop for analysis.

Will my participation be confidential?

Your participation and the information we collect about you during the course of the research will be kept strictly confidential.

Only members of the research team and responsible members of the University of Southampton may be given access to data about you for monitoring purposes and/or to carry out an audit of the study to ensure that the research is complying with applicable regulations. Individuals from regulatory authorities (people who check that we are carrying out the study correctly) may require access to your data. All of these people have a duty to keep your information, as a research participant, strictly confidential.

No personal details are collected whatsoever and so the participation is as a result confidential.

Do I have to take part?
No, it is entirely up to you to decide whether or not to take part. If you decide you want to take part, you will need to give verbal consent. If you do not want to take part, all that is required is to tell the investigator verbally.

What happens if I change my mind?
You have the right to change your mind and withdraw at any time without giving a reason and without your participant rights *(or routine care if a patient)* being affected.

If you change your mind halfway through the study, you can opt-out by verbally communicating this to the investigator.

If you withdraw from the study, we will keep the information about you that we have already obtained for the purposes of achieving the objectives of the study only.

What will happen to the results of the research?
Your personal details will remain strictly confidential. Research findings made available in any reports or publications will not include information that can directly identify you without your specific consent.

The recorded data will be analysed, and the results will be written about in my Individual Project Report. The data may also be graphed or displayed in tables inside the report.

Where can I get more information?
By contacting the investigator, Aran McConnell using the email ajm2g17@soton.ac.uk.

What happens if there is a problem?
If you have a concern about any aspect of this study, you should speak to the researchers who will do their best to answer your questions.
If you remain unhappy or have a complaint about any aspect of this study, please contact the University of Southampton Research Integrity and Governance Manager (023 8059 5058, rgoinfo@soton.ac.uk).

Data Protection Privacy Notice
The University of Southampton conducts research to the highest standards of research integrity. As a publicly-funded organisation, the University has to ensure that it is in the public interest when we use personally-identifiable information about people who have agreed to take part in research. This means that when you agree to take part in a research study, we will use information about you in the ways needed, and for the purposes specified, to conduct and complete the research project. Under data protection law, 'Personal data' means any information that relates to and is capable of identifying a living individual. The University's data protection policy governing the use of personal

data by the University can be found on its website (https://www.southampton.ac.uk/legalservices/what-we-do/data-protection-and-foi.page).

This Participant Information Sheet tells you what data will be collected for this project and whether this includes any personal data. Please ask the research team if you have any questions or are unclear what data is being collected about you.

Our privacy notice for research participants provides more information on how the University of Southampton collects and uses your personal data when you take part in one of our research projects and can be found at http://www.southampton.ac.uk/assets/sharepoint/intranet/ls/Public/Research%20and%20Integrity%20Privacy%20Notice/Privacy%20Notice%20for%20Research%20Participants.pdf

Any personal data we collect in this study will be used only for the purposes of carrying out our research and will be handled according to the University's policies in line with data protection law. If any personal data is used from which you can be identified directly, it will not be disclosed to anyone else without your consent unless the University of Southampton is required by law to disclose it.

Data protection law requires us to have a valid legal reason ('lawful basis') to process and use your Personal data. The lawful basis for processing personal information in this research study is for the performance of a task carried out in the public interest. Personal data collected for research will not be used for any other purpose.

For the purposes of data protection law, the University of Southampton is the 'Data Controller' for this study, which means that we are responsible for looking after your information and using it properly. The University of Southampton will keep identifiable information about you for 1 years after the study has finished after which time any link between you and your information will be removed.

To safeguard your rights, we will use the minimum personal data necessary to achieve our research study objectives. Your data protection rights – such as to access, change, or transfer such information - may be limited, however, in order for the research output to be reliable and accurate. The University will not do anything with your personal data that you would not reasonably expect.

If you have any questions about how your personal data is used, or wish to exercise any of your rights, please consult the University's data protection webpage (https://www.southampton.ac.uk/legalservices/what-we-do/data-protection-and-foi.page) where you can make a request using our online form. If you need further assistance, please contact the University's Data Protection Officer (data.protection@soton.ac.uk).

The data will be anonymised.

Thank you for reading this information form, and if you have agreed to, taking part in the study.

A.3 FEPS Consent Form

UNIVERSITY OF Southampton

CONSENT FORM

Study title: Using embedded device in rehabilitation monitoring of leg surgery patients.

Researcher name: Aran McConnell
ERGO number:**52959**

Please initial the box(es) if you agree with the statement(s):

I have read and understood the information sheet (*2020-03-13, Version 1 of participant information sheet*) and have had the opportunity to ask questions about the study.	
I agree to take part in this research project and agree for my data to be used for the purpose of this study.	
I understand my participation is voluntary and I may withdraw (at any time) for any reason without my participation rights being affected.	

Name of participant (print name)……

Signature of participant……

Date……

Name of researcher (print name)……

Signature of researcher ……

Date……

I understand that should I withdraw from the study then the information collected about me up to this point may still be used for the purposes of achieving the objectives of the study only.

I understand that I will not be directly identified in any reports of the research.

I understand that my personal information collected about me such as my name or where I live will not be shared beyond the study team.

[13/03/2020 [Version 1]

A.4 FEPS Simple Risk Assessment

UNIVERSITY OF Southampton

Health & Safety Risk Assessment Template

Work task / activity	Study: Use of embedded devices in rehabilitation monitoring for leg surgery patients. (**ERGO/FEPS/52959**)				
Assessor	Aran McConnell	**Responsible Manager**	Aran McConnell	**Date**	11/03/2020
Faculty / Service	ECS	**Academic Unit / Team**		**Location**	
Brief description of task / activity	Student Study to test a Device used to collect data on leg surgery patients, the test requires the participant to wear the device and lift their leg while seated. It also requires them to walk around while wearing the device.				

Reasonably foreseeable hazards	Inherent risk	Controls	Residual risk
Muscle strain by lifting of the leg	Low x Med High	Make sure only those fit and healthy enough to be unlikely to be injured to take part in the study.	Low x Med High
Tripping and falling while walking	Low x Med High	Keep the area of the study free of any trip hazards.	Low x Med High
	Low Med High		Low Med High

UNIVERSITY OF Southampton

Risk assessment checklist

- ☑ Risk assessments must be 'suitable and sufficient', that is, should cover all relevant issues and include enough detail.
- ☑ Work tasks & activities should be risk assessed, and not, as such, substances (but rather use of substances), or equipment (but rather use of equipment), or locations (but rather activities therein), or people (but rather what they do).
- ☑ This template is for 'general' risk assessments, and is suitable for most hazards, but certain hazards require additional regulatory and technical detail, such as ionising radiations, biological agents, genetic modification, noise, hazardous chemicals, etc.
- ☑ Risk assessments can be generic, provided they are 'suitable and sufficient', that is, identify all reasonably foreseeable hazards, meaningfully estimate risk, and delineate effective controls.
- ☑ 'Hazards' are things with the potential to cause harm.
- ☑ The qualification 'reasonably foreseeable' is applied to hazards to indicate that far-fetched, improbable hazards need not be considered, and also neither need the obvious hazards of everyday life.
- ☑ 'Inherent' risk is that before controls are applied.
- ☑ Risk should be estimated using the matrix on the next page.
- ☑ 'Controls' are measures to eliminate or reduce risk.
- ☑ 'Residual' risk is that after controls are applied.
- ☑ The assessment should consider:
  - ☑ Any competency, training and supervision that may be necessary.
  - ☑ Reasonably foreseeable emergencies, and include suitable contingency plans.
  - ☑ Any health surveillance that may be necessary.
  - ☑ Any waste management or other environmental issues that may arise.
- ☑ The declaration at the end of the assessment must be signed by the responsible manager, principal investigator, project leader, etc.

UNIVERSITY OF Southampton

Risk estimation matrix

High risk – requires controls to reduce risk before activity / task can commence (or continue).

Medium risk – requires controls to reduce risk as much and as soon as is reasonably practicable.

Low risk – all risk should be reduced to this tolerable level, so far as is reasonably practicable.

Reasonably foreseeable consequence severity / Likelihood [3] of consequence	Minor superficial injury; or slight and temporary health effect; or trivial damage to equipment / building; or minimal disruption to work activities	Moderate significant injury or illness [1]; or temporary minor disability; or minor damage to equipment / building; or slight disruption to work activities	Major serious injury or illness [2]; or significant or permanent disability; or significant damage to equipment / building; or significant disruption to work activities	Critical fatal injury or illness; or substantial and permanent disability; or severe damage to equipment / building; or extensive disruption to work activities	Catastrophic fatal injury or illness for multiple persons; or enormous damage to equipment / building; or disastrous disruption to work activities
Almost certain very high probability, at or approaching 100%	high risk	high risk	high risk	high risk	high risk
Likely high probability, 1 in 10 chance or higher, once in two weeks or longer for activities on a daily basis	medium risk	high risk	high risk	high risk	high risk
Possible significant probability, 1 in 100 chance or higher, once in six months or longer for activities on a daily basis	low risk	medium risk	high risk	high risk	high risk
Unlikely low probability, 1 in 1,000 chance or higher, once in four years or longer for activities on a daily basis	low risk	low risk	medium risk	high risk	high risk
Rare very low probability, 1 in 10,000 chance or higher, once in a decade or longer for activities on a daily basis	low risk	low risk	low risk	medium risk	high risk
Almost never extremely low probability, less than 1 in 100,000 chance, once in a century or longer for activities on a daily basis	low risk	low risk	low risk	low risk	medium risk

[1] 'Significant injury' could include, for example, laceration, burn, concussion, serious sprain, minor fracture, etc. 'Significant illness' could include, for example, dermatitis, minor work-related musculoskeletal conditions, partial hearing loss, etc.

[2] 'Serious injury' could include fracture or dislocation (other than fingers, thumbs or toes), amputation, loss of sight, penetration or burn to eye, serious electric shock, asphyxia, or any injury leading to unconsciousness or requiring resuscitation or admittance to hospital for more than 24 hours. 'Serious illness' could include, for example, requiring medical treatment after chemical or biological or radiological exposure, severe musculoskeletal conditions, severe dermatitis, asthma, etc.

[3] For likelihoods in between the listed values, use the higher likelihood to estimate risk.

Appendix B

Cost Breakdown

Project Cost Breakdown				
Item	**Cost**	**Quantity**	**Shipping**	**Total**
Arduino	£33.90	1	0	£33.90
Memory Chip (Through Hole)	£0.88	1	0	£0.88
Memory Chip (SMD)	£0.92	1	£17.8	£18.72
Battery Holder	£0.27	1	0	£0.27
Battery Holder (Breakout Board)	£2.46	1	0	£2.46
PCB	£1.61	5	£6.11	£7.72
Total Cost				£63.95

Total Project Cost: £63.95

Cost Per Unit: £36.63

This cost per unit excludes items used for prototyping such as the through hole memory chip,and battery holder breakout. It also only includes the cost of 1 PCB, and excludes shipping for components as, due to components being ordered in one order, this cost is shared and is attributed to development in this case for simplicity.

Appendix C

Design Archive Contents

C.1 C Code (For Embedded Device)

EmbeddedCode

1. EmbeddedCode.ino (Arduino Sketch File)
2. BLE.h
3. BLE.cpp
4. IMU.h
5. IMU.cpp
6. Memory.h
7. Memory.cpp
8. Definitions.h
9. Definitions.cpp

C.2 C# Xamarin Forms Code

App. Code

1. DataCapture_v3.sln
2. **DataCapture_v3**
3. **DataCapture_v3.Android**

C.3 PCB Design Files

PCB

1. RehabilitationDevice.brd
2. RehabilitationDevice.sch