MASSEY UNIVERSITY
ENGINEERING

# SCHOOL OF ENGINEERING AND ADVANCED TECHNOLOGY

A thesis presented in partial fulfilment

of the requirements for the degree of

Masters

in

Engineering

# A feasible and effective remote diagnosis system for healthcare

**Daniel Rowe**

**2018**

**SUPERVISOR**
**A. Dr Liqiong Tang**

## Summary:

This master thesis documents the research and developmental process in creating a feasible and effective remote diagnosis system for healthcare.

The research is within the sector of the rapidly expanding practice that is Telemedicine. Telemedicine is defined as “the remote diagnosis and treatment of patients by means of telecommunications technology” [1]. It has become a reality for delivering high quality healthcare to patients not only in remote locations but also to monitor elderly patients as well as patients living with chronic diseases. In the past decade, advancements in technologies have been fuelling the growth of telemedicine. Numerous healthcare products have appeared on the market and are moving towards smartphone applications (apps), some of which include remote consultation using video calling software. The majority of these products are stand-alone applications with limited monitoring of patient’s vital signs. A number of systems are novel, low cost and specifically designed for monitoring the patients’ vital signs, but few offer a fully integrated system for remote diagnosis. Remote healthcare diagnosis systems, especially affordable small devices that provide reliable data in real-time, still challenge researchers.

This research studied the currently available telemedicine systems, cloud database and the features of smartphone apps. The research confirmed the possibility to develop a hand-held system that could obtain remote patient vital signs and transmit the data, effectively in real-time to healthcare professionals for diagnosis via a cloud database and smartphones. A proposed system was designed, and a physical prototype developed. Tests made on the prototype proved the system can capture remote patient’s vital signs and transmit them to a doctor through a cloud database and a smartphone app. The system consists of three major units:

- A hand-held device that measures patient vital signs.
- A smartphone application with a simple user interface to communicate with health professionals via internet and to the device via Bluetooth.
- A cloud database for data transfer and the communication with the smartphone application through an internet connection.

The outcome of this research confirms it is feasible to develop small, economic and portable systems for health diagnosis. Such systems could be very useful for remote patients, especially those living with chronic diseases and requiring regular medical checks, without traveling to access health services to obtain professional treatment.

# Acknowledgement

In completing this Master of Engineering degree in Mechatronics, I would like to take this opportunity to thank:

My supervisor, Dr Liqiong Tang from the School of Engineering and Advanced Technology (SEAT), Massey University, Palmerston North, New Zealand for her guidance, time and advice throughout this project.

I also would like to thank, Beth, my partner, family members and friends who gave their support and guidance throughout this research.

# Table of Contents

# List of Figures

# List of Tables

# 1. Introduction

## 1.1 Telemedicine

This research falls into an area called telemedicine. It is a form of healthcare where the patient is diagnosed or treated remotely using communication technologies. Telemedicine is currently experiencing rapid growth which is being fuelled by the development of new technologies and growing consumer demand for more affordable healthcare. This type of healthcare is expected to grow at an annual rate of 14.3% through to 2020, increasing its value to $36.2 billion [2]. Telemedicine is called the "future" of healthcare and provides several unique benefits compared to conventional healthcare such as reducing hospital visits, cost savings and improved access to healthcare for remote patients.

## 1.2 Problems in Current Healthcare System

The high cost of today's healthcare is becoming an issue for both governments and families. It also poses challenges to researchers for viable solutions for future healthcare. Many studies document issues within the current healthcare systems. Hospitals, with state-of-the-art facilities and specialists, are generally located in large cities where the services are hard for remote patients to access. The aging population increases the urgency for healthcare solutions designed for monitoring and treatment of elderly patients who live in remote locations. Within traditional healthcare systems and services, it is difficult to provide practical solutions for such problems. Healthcare costs, as a whole, keep increasing, due to:

- Increasing aging population
- Increasing number of patients living with chronic diseases
- Remote and rural patients must travel to see a healthcare professional adding additional costs
- Majority of medical grade equipment is difficult to use without specific training

## 1.3 Aim of the Research and Objectives

The problems mentioned in Section 1.2 Problems in Current Healthcare System contribute to the strain on traditional healthcare systems. It is hard for current healthcare systems to provide effective ways to solve these problems. With the fast development in science and technology, telemedicine creates a new healthcare field and has attracted huge research interest. Future-focused healthcare management (diagnosis and treatment) could provide practical solutions for delivering economic and effective healthcare. The aim of this research is to investigate the feasibility of developing a user friendly, economic, portable diagnosis system for healthcare. To achieve this goal, the research had to fulfil the following objectives:

- Conduct a literature review on current available devices and systems for remote healthcare. This will include studying patient monitoring methods, data acquisition and communication methodologies to understand the features, advantages and drawbacks.

- Investigate database development platforms that have the potential to establish a cost effective and reliable remote healthcare database. It needs to be easily integrated with modern electronic systems such as cell phones and wearable devices.
- Study communication methodologies and systems focusing on wireless communication systems and products for remotely accessing healthcare service.
- Develop a small hand-held device that can wirelessly access health services and perform remote vital signs assessment.
- Design and build a physical prototype to evaluate the functions and system performance.

## 1.4 Challenges

A fully integrated healthcare system that has real-time remote diagnosis function presents many challenges. Such a system requires reliable data from the patient, high speed communication, secured database, and easy access without location constraint. Although there have been many attempts reported in recent years using smartphones, wearable devices, small personal monitoring systems etc., an effectively integrated healthcare diagnosis system is still under research. In healthcare, all medical devices and systems must be robust, accurate, reliable, and meet medical regulations and standards.

## 1.5 Research Scope

As stated previously in Section 1.4 Challenges, a fully integrated real time remote healthcare and diagnosis system hinges on a number of cutting-edge challenges. It is impossible for this research project to fulfil all the challenging tasks. To achieve the goal of this research, the project work will focus on the following aspects:

- Investigate fast and reliable communication methodologies and systems focusing on communication solutions that can be implemented in a small handheld device.
- Study vital signs sensing methods, technologies and systems. Electrode cardiogram (ECG) signal detection will be studied as a means of detecting the heart rate. This research involves analogue circuity, amplification and filtering circuits study, design and optimization.
- Cloud-based database technology and development platform will be studied as a means of storing and transferring information from the patient to the doctor and vice versa.
- Research into smartphone applications for app-based healthcare that could be integrated with the handhold diagnosis system.

## 1.6 Final Solution

The outcome of this research is positive and very encouraging. Based on current technologies, the research confirms that it is possible to develop a hand-held, cost effective and reliable remote healthcare diagnosis system. Such small systems could allow remote patients to access healthcare and communicate with their doctors and transfer their vital signs effectively in near real-time without location constraint. A testing prototype has been developed. The system consists of three key units: a hand-held device, a smartphone app and a cloud database.

The hand-held device is able to measure patient's heart rate and temperature as an example, remotely in near real time. The ECG signal detecting unit involves both software and hardware development. The hardware comprises analogue circuity design and electrode design. The software development consists of methodologies for validating ECG signals, determining the heart rate, and wireless communication for data transfer.

The smartphone app developed by this research is able to provide an effective communication link between the cloud database and the hand-held device with a simple user interface controlled by a user authentication function. The wireless communication was developed based on Bluetooth and WLAN. The entire system is designed to be used by both healthcare professionals and patients. Healthcare professionals can use the system to obtain patient information via the cloud database and patients can send their information to the same database.

A cloud storage and the database has been established. It allows information to be stored in the cloud and transferred between remote patients and healthcare professionals. Information from the patient collected by the hand-held device is transferred to the database via the smartphone app. The doctor can receive the data effectively in near real time for diagnosis or retrieve the patient information at a later time via the cloud database.

## 1.7 Organisation of Thesis

This master thesis consists of the following chapters.

*1. Introduction* – Gives an overview of this research.

*2. Literature Review* - Covers a literature review including telemedicine, outlines issues in commercial products and the challenges.

*3. Remote Diagnosis System Design* - Documents the architecture of the remote diagnosis system and sensing system.

*4. ECG Signal Acquisition and Filtering* - Investigates the methodologies for capturing ECG signals, ECG data amplification and filtering.

*5. Dry Electrode Study* - Covers electrode size and position optimization for obtaining strong ECG signals.

*6. Data Collection Methodologies* - Documents the methodologies for ECG signal validation and heart rate determination.

*7. Hand-Held Device Software Development* - Covers the development of the software in the hand-held device.

*8. Cloud-Based Database Establishment* - Documents the development of the cloud-based database.

*9. Smartphone Application* - Outlines the smartphone application design, functions and methods.

*10. Hand-Held Portable Healthcare Device Development and Prototyping* - Covers the physical hand-held device design and prototyping.

*11. Final System, Discussion and Conclusion* - Summaries the outcome of this research, outlines the future work and leads to the final conclusions.

# 2. Literature Review

This literature review focuses on methodologies and technologies for providing healthcare to patients who:

- Live in remote areas
- Require post operation check-ups
- Live at home with chronic diseases

This form of healthcare is known as telemedicine but can also be referred to as E-health, Mobile health, Tele-health and healthcare. This review studies the current status of telemedicine and identifies key issues and problems that need to be solved through further research and the development of new healthcare methodologies, technologies, devices, and equipment.

## 2.1 Telemedicine

A simple search defines telemedicine as "the remote diagnosis and treatment of patients by means of telecommunications technology" [1]. Telemedicine is currently "experiencing rapid growth and deployment across a variety of applications" [2]. It is regularly referred to as the future of healthcare. The two major drivers growing telemedical are:

- The fast and constant development of new technologies is building an infrastructure for telemedicine. The quality of the key components such as communication speed, reliability of wireless system, safety of cloud storage, and smart systems are continuously improving. These new inventions and technologies form a great platform for delivering high quality healthcare.
- The growing demand from aging population and chronic patients for more affordable and easily accessible healthcare [2], [3].

Lacktman [2] also stated that the estimated global telemedicine market is expected to grow at an annual rate of 14.3% through to 2020, increasing its value from $14.3 billion in 2014 to $36.2 billion in 2020. Telemedicine has a great potential and can provide a number of benefits for both the patients and the healthcare providers in terms of:

- Significantly reducing hospital visits [4] and the duration of hospital visits [2]
- Cost saving both for patients and healthcare providers [4], [3]
- Improving patient care and satisfaction [2]
- Easy access to quality healthcare service for rural and remote areas [5], [6]

A unique feature of telemedicine is that it reduces travel costs and incovenience to rural or remote patients while improving their health outcomes [6]. Telemedicine can also play an important role in regional care for patients with chronic disease or post surgery care [7]. [7] also stated:

- Telemedicine is a reality for providing medical care for elderly patients
- Telemedicine is useful in short time monitoring
- Simple, user friendly medical devices need to be specifically developed to provide high quality healthcare to elderly patients.

[8] undertook a systematic review of literature looking at studies which documented the use of telemedicine, in a home environment, for monitoring patients who have experienced heart failure. In this review, they concluded that telemedicine:

- Is easy to use and widely accepted by patients and health professionals
- Is economically viable
- Reduces hospital visits
- Improves the quality of life

There are many sub-categories within telemedicine some of which are as follows:

- Tele-emergency
- Tele-pharmacy
- Tele-cardiology

Andrew J. Potter, et al. [5] , conducted a study on the benefits of telemedicine in rural communites. They surveyed a number of clinicians in varied clinical settings regarding the benefits from implementation a hub facility offering telemedicine. The result of the study conducted by Andrew J. Potter, et al. [5] found that respondents had a positive perception around tele-emergency services and how telemedicine enhanced rural hospitals' reputation for high quality healthcare.

Heaney, et al. [9], conducted a study where software, called HealthPresence, was used in the National Health Service (NHS) in Scotland for conducting remote consultations. In this study they concluded that telemedicine consultations were safe and appropriate. The patients, who participated in this study, were 90% satisfied that their needs were met with the remote consultation. HealthPresence is a product developed by Cisco where patients attend a remote hub. This hub is a stand-alone unit containing screens and cameras which allow it to be set up anywhere. Within the remote hub patients are consulted by a healthcare professional using high quality video calling. HealthPresence allows for 3$^{rd}$ party medical devices to be used over their developed software and storage facilities.

Telemedicine has also been used in the postnatal area. A study conducted by [10] investigated how postnatal parents experienced the use of telemedicine following early discharge from hospital, in particular looking to see if their postnatal needs were met and a sense of security was experienced. The study concluded that many of the parents saw telemedicine (a smartphone application in this case) as a lifeline and a way of getting information they required to guide them after early discharge from hospital.

However, there are issues with telemedicine. One of the major issues, which could hold back the development and growth, is laws and legislation within states and countries. As telemedicine means patients could be consulting a medical profressional on the other side of the world. Issues arise around whether healthcare professionals need to be qualifed and registered to pratice and how to ensure the professional has appropriate traning.

## 2.2 Smartphone Apps

Advancement in the development of smartphones over recent years has led to a surge in telemedicine smartphone applications (apps). The majority of these apps are designed to monitor or manage health

conditions. A handful of apps have been designed to link patients with healthcare providers. Smartphone apps offer a low-cost method for providing care and transmitting data, exemplifying the concept of telemedicine. A review conducted by Holtz, Lauckner, & Whitten [11] provided a summary of the current healthcare mobile apps. In this review, they split apps into two categories:

- Clinical Services – Apps to aid healthcare professionals in delivering high quality healthcare like "Epocrates".
- Patient and Caregiver-Oriented App – these apps are designed for long term, regular monitoring of chronic conditions like diabetes. Examples of these are "Glooko" and "Nike +"

This review also outlines how much smartphones have become part of life. It suggests that there are over 5.9 billion mobile phone subscribers worldwide [11]. Holtz, Lauckner, & Whitten [11] also say that in the United States alone 84% of all adults have mobile phones.

What Holtz, Lauckner, & Whitten [11] failed to categorise is the increase in the use and development of apps designed to effectively replace doctors/physician's visits. These apps allow patients to make an appointment with the physician and using video call technology, the consultation is made. These appointments vary in time and cost from app to app. Some of these apps are listed below:

- Maven clinic
- American well
- Teladoc
- Doctors on Demand
- Cleveland Clinic Myconsult

These apps are developed with the philosophy of allowing patients to have convenient 24/7 access to healthcare, typically at a lower cost than a traditional visit to a clinic.

### 2.2.1 Maven Clinic

"Maven Clinic" was developed to "*make it easier for woman to get immediate, professional care from someone they trust"* [12]. They have developed a system where woman can easily connect to the correct healthcare specialists, as required, via a video appointment.

The process could not be simpler:

1. Browse - Browse providers to see what practitioner's best suits your needs.
2. Book - Book an appointment at a convenient time to you
3. Follow up - Follow up appointments if needed or follow-up messages.

Pricing varies from $18 USD for a 10 minute appointment up to $70 USD for 40 minute appointment depending on the practitioner required.

### 2.2.2 Doctors on Demand

"Doctors on demand" is an app where the patient video calls, effectively, a General Practitioner (GP). This video call includes assessment, diagnosis and prescriptions. They offer management of all illnesses from colds through to allergies, anxiety and depression.

The cost: $49 USD per visit.

### 2.2.3 Summary of Phone Apps

The literature study shows that the development in smartphones has created a medium for healthcare services to work through. In recent years, there have been many different approaches and new systems appearing on the market however most current apps for healthcare are generally categorised into:

- Remote consultation
- Health monitoring

## 2.3 Remote Healthcare Devices

Some of the literature reviewed proposed and developed telemedicine devices for obtaining electrode cardiogram (ECG) signals, typically for patient's living with chronic diseases. In general, they stated the need for the development of such products to address the issue of rising healthcare costs caused by an increasing number of patients and to improve the quality of life for patients living with chronic diseases. Wearable devices have been developed that are allowing patients to be remotely monitored from home.

An overview of wearable medical devices for home healthcare was conducted by Hung, Zhang, & Tai [13]. In this study, they described the importance of monitoring vital-signs in tele-home healthcare and how cellular communication can be added to a system to expand the service coverage. Fensli, Gunnarson, & Hejlesen [14] suggested two methods for collecting information:

- Real-time – Information is available to the healthcare professional at the time of recording the data
- Store and forward – Data is saved and then collected at a later time for analysis

ECG readings are one of the most clinically used examination tools in treatment of patients with chronic diseases [15]. Donati, Benini, Celli, Iacopetti, & Fanucci [15] commented that current ECG devices used in the healthcare enivroment are difficult for a non-skilled person to use and therefore the need to develop a more user friendly ECG monitoring system is justified.

Fensli, Gunnarson, & Hejlesen [14] developed a system to continuously record an ECG signal from a patient which can detect irregular ECG activity and then alarm hospital staff or appropriate caregiver. Their prototype included:

- ECG sensor which is attached to the patient's chest. This sensor includes a radio transmitter
- Using radio frequencies, the ECG signal is sent to a receiver called a Personal Device Assistance (PDA) or what we now call a Mobile Phone.
- The PDA then sends the data over General Packet Radio Services (GPRS), commonly referred to as 2G, to a remote Clinical Alarm Station (CAS)
- The CAS is located in a hospital or at an appropriate location and can act appropriately

Fensli, Gunnarson, & Hejlesen [14] then concluded that their system acts as a continous event recorder designed for patients who have survived cardiac arrest or another chronic disease. A device developed by Fensli, Gunnarson, & Gundersen [14] uses two forms of wireless communication to get ECG information to doctor. The patients wear an ECG sensor which sends data over radio frequency to a hand-held device (phone). The phone then uses GRPS or Global System for Mobile Communications (GSM) to send the information over the internet via servers so the doctor can then access the ECG information.

Another ECG signal monitoring system developed by Donati, Benini, Celli, Iacopetti, & Fanucci [15] [15] used:

- Dry metal eletrodes to acquire the ECG signal from the patients hands
- An instrumentation amplifer to amplify the signal
- Signal processing  in a mircocontroller

- Bluetooth interface for sending data from the device to a mobile phone application.

The system has a simple user interface and a liquid crystal display (LCD) screen to display the heart rate. Using the smartphone application, the ECG signal can be acquired. The developed system is easy to use compared to traditional ECG devices and gives similar results.

### 2.3.1 Medweb Hand-Held Telemedicine Kit

Medweb has developed what they call the smallest mobile telemedicine product available, shown in Figure 2.1. It is designed to meet the needs for "first-line" [16] patient's examinations. It is planned to be used by doctors/healthcare professionals in remote locations to collect data from patients and transmit data to a cloud storage system or to a specialist anywhere in the world. The device is advertised as a "mobile practice" [16] and contains a number of medical examination tools.

Figure 2.1 - Medweb Hand-Held Telemedicine Kit [16]

### 2.3.2 Globalmed

Global Med have developed a number of telemedicine devices. Two of these are called “clinical access station” and “transportable exam station”.

#### *2.3.2.1 Clinical Access Station*

The clinical access station is a modular system, shown in Figure 2.2, allowing a base product to be developed into speciality areas, as required. It is designed to be mobile and to be taken to the patient. Using wireless connectivity, the system can be connected to a network and information shared with other remote healthcare professionals.

Figure 2.2 - Globalmed - Clinical Access Station [17]

#### *2.3.2.2 Transportable Exam Station*

The aim of this system is to take the healthcare to the patient. The devices contained in the kit are designed for follow up care, in particular, chronic care management. The kit contains Wi-Fi and wireless capabilities, which allows remote healthcare providers the ability to video call or send images from anywhere. The device is shown in Figure 2.3 below.

Figure 2.3 - Globalmed - Transportable Exam Station [18]

### 2.3.3 AMD

AMD is a company that develops only telemedicine products including “Portable Teleclinic” and “Clinical Assist” to name a few.

#### *2.3.3.1 Portable Teleclinic*

The portable teleclinic, shown in Figure 2.4, is effectively a computer in a box that has inputs for medical devices like ECG leads. The system has been developed as “pack and go” [19] for situations where the medical gear needs to be carried in and set up with limited space. It comes with “AGNES Interactive” [19] which uses AMD software/methodology for sharing medical data and video calls with remote consulting physician.


**Figure 2.4 - AMD - Portable Teleclinic [19]**

#### *2.3.3.2 Clinical Assist*

This is a base platform which has been designed as a modular system to accommodate various applications and medical specialties. Modular systems have the advantage of keeping costs down while getting the features you require. The basic system includes a laptop and video calling hardware as well as the “AGNES Interactive” [19] which uses AMD software/methodology for sharing medical data and making video calls with remote consulting physicians. The Clinical Assist system is shown in Figure 2.5 below.

Figure 2.5 - AMD - Clinical Assist [20]

### 2.3.4 Summary of Devices Reviewed

The medical devices reviewed, in general, are expensive and are hospital grade devices. This means that typically they cannot be used by a patient without training. The other key issue with these systems is the cost, which typically is out of the reach of the average household. Other devices have been developed from an academic point of view, but these are typically not, fully integrated systems between doctor and patient.

## 2.4 Existing Methodologies

After reviewing existing equipment and systems that are used in telemedicine and healthcare applications, it has been observed that two main methodologies are required. These are the data acquisition of patient's vital signs and wireless communication technology. This section will discuss current methodologies used for ECG data acquisition and wireless communication for the transfer of healthcare data. It also looks at database methodologies that could be useful in storing data for multiple uses and related software.

### 2.4.1 ECG Signal and R Wave

A vital tool in diagnosing/monitoring patients is the collection of patient vital signs such as, heart rate. The ECG wave itself is used for complex diagnosis and monitoring of patients. Recording and monitoring of the heart rate has been used for a long time. Traditionally 12 lead ECG systems are used in healthcare to capture a patient's ECG signal. In a device designed to be used easily, it is not practical to use such a method, where electrodes are required to be placed in specific locations on the body.

The ECG wave is a measurement of the voltage difference over time between two points on the body created by the electrical signals controlling heart muscles.

For determining the heart rate, there are many different methods and techniques including infrared sensors and the physical counting of pulses. Another way is by looking at the small electrical signals used to control the muscles in the heart (ECG wave). From this signal, the heart rate can be determined. Figure 2.6 below shows a typical ECG wave.

Figure 2.6 - Typical ECG Wave Generated When the Heart Contracts [21]

The defining feature of this wave is the repeating pulse known as a R wave, Figure 2.6. To detect the small electrical signals, amplification and conditioning circuit is required to increase the voltage to a level that can be used on a standard microcontroller. The signal needs to have a well-defined pulse to allow for easy detection of the heart rate.

### 2.4.1.1 Basic ECG circuit

In theory, an ECG detection circuit is very simple, however practically this is not the case. There is a large amount of noise due to the nature of the signals being detected because of the following reasons:

- The signal strength is very weak and therefore requires a large amount of amplification.
- Noise due to movement of other muscles.
- The signals vary from person to person so a circuit which works on one person might not work as so well on another.
- Other noises in the environment.

A basic ECG data acquisition system contains the following sections.

1. Electrodes - Make contact with patient's skin and pick up the electrical signals used to control muscles in the body.

2. Pre-filtering process - The electrical signals are passed through a pre-filtering process to remove noise before the amplification process.

3. Amplification process - Once the signals have been filtered, the signals are amplified. To get an ECG signal the difference between two signals from the different measuring points needs to be found, typically achieved using a difference amplifier.

The signal can then be filtered again to further remove noise. The resulting signal is an ECG wave that can be used for monitoring and diagnosis.

### 2.4.1.2 R Wave ECG Detection

As mentioned in the above section 2.4.1.1 Basic ECG circuit, the R wave of the ECG signal is the defining feature of the wave. Liao, Na, & Rayside [22] documented an algorithm for detecting the R-peaks in a ECG wave which can then be used to determine heart rate. The developed algorithm was 99.4% acurate. To develop the algorithm Liao, Na, & Rayside [22] came up with the notion of measuring the area over the curve, with the idea being that if something is tall and narrow it will have a lot of emtpy space around it and therefore a large area over the curve. The code developed calculates the area above the curve based of the gradient of the curve. Due to the nature of the R wave which is steep and narow it has a large gradient which the algroithm see as a larger area. The time the area is calculated over is set at 60ms resulting in the R wave having a large area than the T or P wave. The differenct sections of the ECG wave is shown in Figure 2.6.

### 2.4.2 Communication

In developing a telemedicine system, communication is critical because the definition of telemedicine is healthcare at a distance. In recent years, there has been dramatic improvement in communication technologies, especially in wireless communication. Some of these advanced technologies are widely used in many applications. The following list presents some of the technologies and methodologies that can be used for healthcare systems.

- Bluetooth [15], [13]
- Licenced radio [23], [14]
- Cellular communication (GPRS/GSM/3G/2G) [23], [14]
- Wireless Local Area Network (WLAN)/ Wi-Fi
- Wide Area Network (WAN)
- ZigBee
- Wi-Fi Direct

Within the above, three forms of communication are commonly found on today's mobile phones. These are:

- Bluetooth
- Wi-Fi
- 3G or 4G

Advantages and disadvantages of a number of different common communication methods are discussed below.

#### *2.4.2.1 Bluetooth*

Bluetooth is a form of wireless communication which is used to transfer data between two or more devices in close proximity. The type of network created by Bluetooth is a personal area network (PAN) as it is created and used for personal use. Bluetooth is very commonly used in mobile phones as wells as tablets, laptops, headphones, printers and DVD players to name a few. It can achieve transfer speeds of 25 Mb/s. Generally speaking, Bluetooth is considered secure due to the low broadcast range meaning that any hacker has to be in close proximity. Below is a list of advantages and disadvantages of Bluetooth communication [24].

Advantages:

- Range of 100m on latest the versions
- Low processing power
- Simple to use
- Free
- Low interference with other networks

Disadvantages:

- Slow transfer speeds compared to Wi-Fi Direct

- Can easily be hacked by someone with basic knowledge, however they are required to be in close proximity

#### *2.4.2.2 Licenced Radio*

Radio Frequency (RF) uses electromagnetic waves to transport information. It is widely used to broadcast FM radio stations and within hobbyist equipment. Below is a list of advantages and disadvantages of Radio Frequency communication [25].

Advantages:

- Long range 20 miles with line of sight
- Free if using the standard, IEEE 802.11, for implementing a wireless local area network

Disadvantages:

- Low data transfer rate, in the order of 1 Mb/s range
- High potential for interference from other RF based system
- Limited security

#### *2.4.2.3 Cellular Communication*

GSM/2G/GPRS/3G/4G are all different forms of cell communication used in mobile phones. Each form (in order above) is effectively the next version up with different transfer speed and capabilities. GSM was developed in late 1980s. It stands for Global Standard for Mobile and is commonly referred to as 2G digital cellular technology. This supports calling and SMS (texting). GPRS (General Packet Radio Service) is an upgrade from GSM. GPRS is often mentioned as 2.5G (improvement on 2G but below 3G). It allows access to the internet and generally has low data transfer speeds (tens of kilobits per second (kbps)).

**3G**

3G was developed in 2001 to overcome issues with 2G, particularly the low transfer speed. The 3G can reach speeds 125Kb/s-2Mb/s which allows the user to browse the web. 3G is described as a family of standards which all work together to get improved speeds. Below is a list of advantages and disadvantages of 3G communication [26].

Advantages:

- High data transfer, up to 2Mb/s
- Network for maps and positioning services

Disadvantages:

- Cost

**4G**

4G is a replacement for 3G where high transfer speeds are achievable with better security. Introduced in 2010, 4G is slowly growing in popularity and being incorporated into mobile phones. Speeds are possible anywhere between 100Mb/s – 1 Gb/s. Below is a list of advantages and disadvantages of 4G communication [27].

Advantages:

- Video and movie streaming
- Higher speeds than 3G and Wi-Fi
- More secure than Wi-Fi and 3G
- Wide coverage, 30 miles

Disadvantages:

- Not available everywhere
- Cost
- Poor battery life on the device when using it

#### *2.4.2.4 Wireless Local Area Network/ Wi-Fi*

Wireless Local Area Network (WLAN) or Wi-Fi is a local network which is commonly found in offices or homes. As the name suggests, it is a local network, generally confined within a building. Below is a list of advantages and disadvantages of WLAN communication [28].

Advantages:

- Easy to use and install
- Low cost

Disadvantages:

- Security
- Transmission speeds – often have a limited maximum speed

#### *2.4.2.5 Wide Area Network*

Wide Area Network (WAN) covers a larger region. There is no defining point where the network changes from a WLAN to WAN. A WAN network could be used by a company to connect a head office in London to a Sales office in New York, achieving high transfer speeds. Below is a list of advantages and disadvantages of WAN communication [29].

Advantages:

- Centralized IT infrastructure
- Boosts privacy – no need to send data over internet
- Increased speed

Disadvantages:

- High setup costs
- Security concerns

#### *2.4.2.6 ZigBee*

ZigBee is a low power, low data rate wireless system with transfer rates up 250Kb/s, and coverage of 10m. It is designed for devices talking to devices. Below is a list of advantages and disadvantages of ZigBee communication [30].

Advantages:

- Low cost
- Simple to set up a network

Disadvantages:

- Not secure compared to Wi-Fi based system
- Limited range
- Limited transfer speeds

#### *2.4.2.7 Wi-Fi Direct*

Wi-Fi Direct uses Wi-Fi technologies to connect two devices together similarly to Bluetooth but allows Wi-Fi transfer speed. Below is a list of advantages and disadvantages of Wi-Fi Direct communication [31].

Advantages:

- Secure
- High transfer speeds 250Mb/s

Disadvantages:

- New technology that still is not supported on all phones

### 2.4.3 Database

A database is "*A collection of information or data that is organized so that it can easily be accessed, managed, and updated*" [32]. There are two types of database. These are known as SQL and NoSQL. Issac, [33] explans the differences between a SQL and NoSQL databases below.

A SQL database is often referred to as a relational database where the data is stored in a table with a set number of rows of data. Data is added in vertically (in columns) so it is called vertically scalable. SQL stands for structured query language.

A NoSQL is commonly referred to as a non-relational or distributed. As opposed to a SQL database, a NoSQL database can be a collection of key-values (data stored with a key that is used to uniquely identify it), documents etc., which does not have a standard schema that it needs to adhere to. A NoSQL database is called horizontally scalable and stands for unstructured query language.

Both types of database have advantages and disadvantages, depending on the type of data that needs to be stored. SQL database is very powerful for manipulating data within the database due to its structure and the complexity of the queries that can be created. A query returns data from the database; effectively, it is a 'get data' routine. A NoSQL database is very good for storing hierarchical data.

The implementation of a database can be done in many ways including; as a database in the cloud or on a server PC over a network. Using a server PC to host a database is generally done in a company network where the network is localised with computers on the network having access to the database. In this case, the data within the database is reasonably secure as it is only accessible from the local network. To implement a telemedicine system where data needs to be read or written from anywhere creates issues when using a server PC as it needs to have access to the outside world via the internet which makes the data less secure. Examples of programs used to create this type of database are MS-SQL and Oracle Express.

A cloud database is stored on a cloud server accessible only through the internet, but this means that it can be accessed from a number of devices including tablets and phones. Generally speaking, it is cheaper to implement a cloud database with data centralized to one location, but accessible from different locations. There still are issues around security due to the nature of the internet, but generally speaking, data is secure. Examples of programs that can create cloud database are MySQL and Google Firebase.

### 2.4.4 Software Development

Software development in the context of telemedicine is generally designed as user interface software. It needs to be simple for the user, look good and be functional. There are many different software development methodologies in use today, some of which are discussed below.

- Waterfall – This is modelled from a sequential development approach. The development flows downstream like a waterfall though various stages. These stages are:
    - Requirement analysis
    - Software design
    - Implementation
    - Testing
    - Integration
    - Deployment
    - Maintenance
- Prototyping – This method is about creating prototypes of the software being developed, e.g. incomplete versions. It is an approach that tests out particular features to reduce project risk.
- Rapid Application development – This method favours intuitive development where up-front planning is minimal. There is less planning, so the software is written faster and it is easier to change the requirements/specifications.

There are many different media forms in which the software can be developed. For telemedicine, key media platforms to develop software in are:

- Web base (webpage)
- Mobile phone applications
- Desktop (PC) application

Each has its own advantages and disadvantages depending on the platform the application is intended for. Web base applications have the advantage of being accessible everywhere and can be updated instantly. Phone applications are good when the application requires regular usage, complex calculations or native functionality like camera use and where data processing is required. Desktop applications are great for developing graphical user interfaces. They are generally more secure and demonstrate high performance, but they are not portable. For these three platforms, there are also many different integrated development environments (IDE), some of which are mentioned below:

- Web base – Aptana, Webstorm
- Mobile phone applications – Android studio, Intellij
- Desktop application - Visual studio

## 2.5 Proposed System

The literature review clearly indicates that commercial products are available which allow information to be sent from a remote location to a healthcare provider; however, these all come at a cost, generally out of reach for the average household. The commercial products contain medical grade equipment which requires a trained professional to accompany the apparatus to get accurate readings. Such systems typically contain 12 lead ECG sensors which require electrodes to be placed in the correct locations on the patient's body to get valid information. Donati, Benini, Celli, Iacopetti, & Fanucci [15] stated that for 'at home' settings, current ECG devices used in the healthcare environment are difficult for a non-skilled person to use and therefore the need to develop a more user friendly ECG monitoring system is required.

Articles studied as part of this literature review identified a number of novel devices designed for self-acquisition of ECG signals in a home setting. In these articles, the authors have created alternative solutions for capturing the patient's ECG signals as opposed to the more traditional 12 lead ECG technique. The end solution in all these cases, are devices capable of recording ECG signals. However, none of the articles reviewed have developed or proposed a fully integrated system as offered by the commercial products, where the doctor can video call a patient and access their vital signs on demand.

Many smartphone apps have been developed that allows a patient to remotely consult a healthcare professional using video call technology. This market seems to be well established and growing rapidly. What if the healthcare professional conducting the video call had access to real-time, vital signs from the patient to help with their diagnostics? This would bring the smartphone apps to the same level for monitoring/diagnosing as the commercial telemedicine devices. What if there was a telemedicine device that could be used alongside the pre-developed systems to accurately collect patient's vital signs such as the ECG and do so reliably and at a low cost?

Imagine a hand-held device that is small, low cost and easy to use, that can measure patient's vital signs and send this information, in real-time, to a healthcare professional to aid their diagnostic process. Such a system can work alongside pre-developed apps already used for conducting remote consultation. It will allow for a fully integrated system where all the required information is available at the current point in time, no matter where in the world the patient or the doctor is located. Such a device will fill a gap between the current commercial telemedicine devices and the smartphone apps already developed. This system could be used in a wide range of fields including general diagnosis, post operation check-ups and on patients living at home with chronic diseases - to a name a few.

The following research hypothesis was developed: A feasible and effective remote diagnosis system for healthcare. Such a system is to embrace the short comings discovered in the telemedicine field while conducting this review. In developing such a device, remote, elderly and chronic patients need to be considered in the design as a number of studies have expressed the view that telemedicine is the future for caring for such patients. Other key factors to consider are:

- The size
- The cost
- Accurate and reliable data collection
- Robust design
- Ease of use

For a system to be useful, data needs to be available at real-time, allowing doctors to obtain patient's current health data and diagnose patients with all the relevant information they require. Such a system does not exist in the current market for middle class families.

If a system achieves the research hypothesis and meets the key points listed above, then it will fill a gap in the field of healthcare that was identified in the literature review.

## 2.6 System Design Considerations

A number of design criteria were identified in the literature review. These form the basis for the specifications of the system to be developed.

Affordability is critical in any industry. For health services aimed at distance patients, the devices used by caregivers or patients must be economically viable. In the proposed system, the devices developed need to be at a price that can be accessible to the majority of members in society.

Another important feature for such devices is that they must be user friendly. Many of the journal articles reviewed identified the need to develop a system for elderly patients to use. A device that is easy to use makes it more appealing to patients. The design also needs to consider that the user might:

- Not be completely mobile, for example, post operation patients
- Not be familiar with new technologies such as elderly patients

For a device to be user friendly its size needs to be considered. The devices to be implemented in the proposed system should be small or even a handheld size. If it is too big, the user will not want to carry it around with them. All the commercial telemedicine devices reviewed were large and are not easily moved from the hospital or where they are set up. For a mobile system, it would need to fit in a suitcase. Having a system which is; small, easily moveable, and easily stored, would be a huge advantage. Of course, aesthetics and ergonomic design would also be an attractive marketing point.

Any healthcare devices must be robust and provide accurate data. When dealing with the health of humans, the accuracy of the information is vital as it can be the difference between mis-diagnosis or even life and death.

For a system to be useful for a healthcare professional, the data from the device needs to be presented to health professionals in real-time. This means that the healthcare professional can get information as it happens from the patient in a remote location. If the healthcare professional uses one of the pre-developed smartphone applications for conducting remote consultation using video call technology, for example Maven Clinic, then information will be available as it happens to add to the diagnosis.

# 3. Remote Diagnosis System Design

## 3.1 Sensing System Design

A diagnosis system could have a number of sensors to capture different types of patient vital signs. For the proposed system, it was decided to only select two key vital signs that are commonly used in patient diagnosis. These are:

- Heart rate (ECG)
- Temperature

These two vital signs can indicate a number of symptoms and are routinely used in health check-ups for patients with chronic diseases, as well as post-operative patients and for general diagnosis purposes. The following sections investigate ECG signal detecting.

### 3.1.1 Detecting ECG Signal

The ECG signal is the electrical signal used to control the heart muscles. An easy way to determine the heart rate is to see how often the ECG signal repeats itself. So, to measure the heart rate, the ECG signal is required. As an ECG signal is very weak, signal amplification is required. There are two possible ways to design the ECG sensing system for the proposed system. One is to develop an ECG amplification circuit; the other is to make use of the available chips or breakout boards available on the market.

#### *3.1.1.1 Development of an ECG Amplification Circuit*

During the literature review, a number of articles were found documenting the development of a circuit for monitoring ECG signals [34], [35]. Simple ECG application circuits were also found [36], [37], [38].

Based on the article by Harden [36] a test unit was developed to examine how well it could detect an ECG wave. A schematic diagram of the circuit is shown in **Error! Reference source not found.** and the outcome is presented in Figure 3.2.

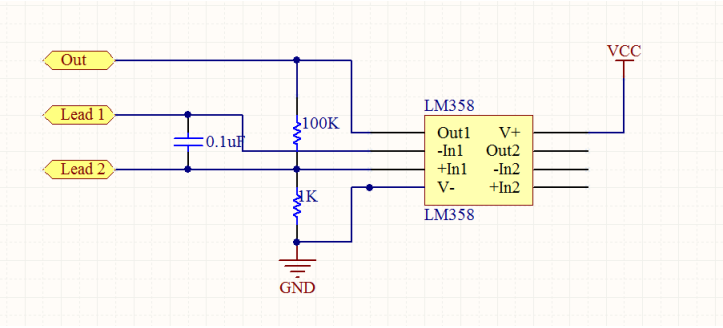



Figure 3.1 - Development of an ECG Application Circuit [36]

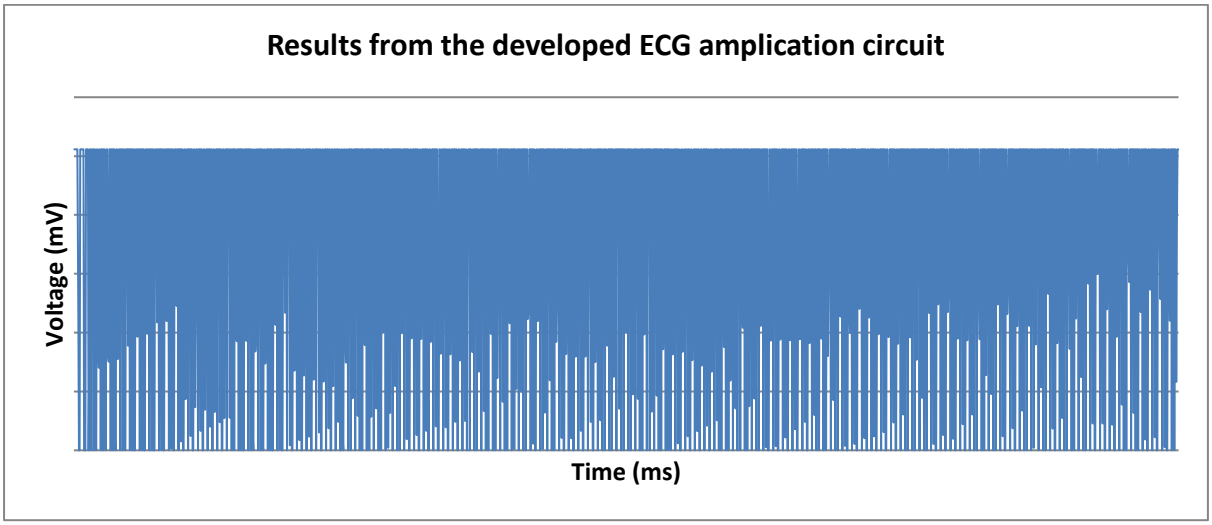



Figure 3.2 - Results from the Developed ECG Amplification Circuit

The results are poor and unworkable. There is no indication of a typical ECG wave. This category of electronics is difficult and requires a good understanding of analogue systems to get the correct gains and filtering. It is also very dangerous to make such systems as poor circuit design can result in high currents drawn from the body which can have huge effects on the human body. Even small currents, of the amplitude of 5 mA, can result in the feeling of a shock [39]. Higher currents in the order of 50-150mA can lead to respiratory arrest and death [39]. Many articles documenting the development of such systems use coins as electrodes and they go on to warn of the risk of burns around where the coins are placed if the circuit is not designed correctly.

### *3.1.1.2 ECG Sensing Using Commercial Chips and Breakout Boards*

A number of commercial chips and breakout boards have been developed. The sellers of such boards can't stress enough that the boards are "not intended to diagnose or treat of any conditions" [40]. In this section these boards, which can be commercially bought, are researched.

#### 3.1.1.2.1 BiTalino

BiTalino by Plux Wireless Biosignals have created products for the hobbyists who are developing their own health monitoring systems. Bitalino have developed a whole series of sensors and controllers aimed at the everyday person with basic knowledge of programming. BITalino claims their "*sensor allows data acquisition not only at the chest ("on-the-person"), but also at the hand palms ("off-the-person") and works both with pre-gelled and most types of dry electrodes. The bipolar configuration is ideal for low noise data acquisition"* [41]. The Bitalino board is small in size and reasonably priced for developing a low-cost system. The Bitalino have produced their own microcontroller to interface the various sensors they have developed. The sensors also work with Arduino and Raspberry Pie. It requires a three-lead electrode in order to detect an ECG wave.

The ECG chip created by BITalino is shown in Figure 3.3 and the specifications, features and cost are listed below.

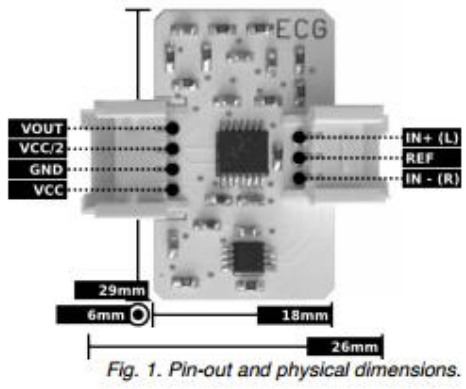



**Figure 3.3 - Bitalino ECG Board [41]**

Specifications [41]:

- Gain: 1100
- Range: ±1.5mV (with VCC = 3.3V)
- Bandwidth: 0.5-40Hz

Features [41]:

- Bipolar differential measurement
- Pre-conditioned analogue output
- High signal-to-noise ratio
- Raw data output

Cost:

- ECG Sensor - €22.50 [42] which correlates to $33.65 (NZD)
- Kit - €149.00 [43] which is approximately $222 (NZD)

#### 3.1.1.2.2 AD8232 chip

The AD8232 chip is a three-lead heart rate monitoring chip. This chip has been developed by Analog Devices for monitoring ECG waves. The chip has been added on to breakout boards to allow for easy use. "*The AD8232 is an integrated signal conditioning block for ECG and other biopotential measurement applications. It is designed to extract, amplify, and filter small biopotential signals in the presence of noisy conditions, such as those created by motion or remote electrode placement. This design allows for an ultralow power analogue-to-digital converter (ADC) or an embedded microcontroller to acquire the output signal easily*" [44]. The AD8232 chip or breakout board are both very low cost making it a practical option. The board is also designed to be connected to a three-lead electrode in order to monitor the ECG wave.

The image of the AD8232 chip is shown in Figure 3.4 and the specifications, features and cost are listed below.


**Figure 3.4 - AD8232 Break board [44]**

AD8232 Chip Benefits and Features [44]:

- Fully integrated single-lead ECG front end
- Low supply current: 170 µA (typical)
- High signal gain (G = 100) with DC blocking capabilities
- Fast restore feature improves filter settling

Cost:

- Chip itself is $6.23 NZD [45]
- Breakout board $26 NZD [40]

#### 3.1.1.2.3 E-Health

Another board which has been designed as a shield for Arduino Uno is called E-Health. A major drawback with the E-health sensor platform is the size; however, it does have flexibility in terms of other sensors that can be added in the future without the need to redesign the circuit. It also requires a three-lead electrode input to acquire the ECG signal.

The E-Health platform interfaces a number of different sensors:

1. Pulse and oxygen in blood sensor
2. Airflow sensor
3. Body temperature sensor
4. ECG
5. Glucometer
6. Blood pressure
7. EMG

Figure 3.5 below shows image of the board the costs are listed below.

**Figure 3.5 - E-Health Sensor Platform [46]**

Cost:

- Board itself €75 which is approximately $112 (NZD) [46]
- A kit which includes the board, ECG and EMG sensor, pulse and oxygen saturation sensor €200 which equals $300 (NZD) [46]

#### 3.1.1.2.4 Olimex

Olimex have developed a board for measuring ECG signals. It is developed as a shield for Arduino Uno microcontrollers. Each shield can have one ECG measuring tool associated with it. The board does not have a specialised chip for processing the ECG signal. It has its own built-in circuit consisting of op-amps and filters, which converts the three-lead electrode input into an ECG signal. The Olimex board is reasonably priced at €19.95 [47], approximately $30 (NZD). The major drawback with this Olimex board is the extra size it adds to the device to be developed.

An image of the board is shown in Figure 3.6 below.


Figure 3.6 - Olimex ECG/EMG Shield [47]

#### 3.1.1.2.5 Selection of ECG Board

In selecting an ECG board to integrate into a small, low cost device two key elements in selecting which board to use are cost and size. A summary of these boards and their cost and size is documented in **Error! Reference source not found.** below.

| Board | Cost (NZD) | Size |
|---|---|---|
| Bitalino | 33.65 | Small |
| AD8232 | 26 | Small |
| E-Health | 112 | Large |
| Olimex | 30 | Large |

Table 3.1 –Comparison of Cost and Size of Available ECG Boards

Based on the above research of available ECG chips and breakout boards and the comparison of the features, the E-health and Olimex boards were ruled out quickly due to the size they add to the system to be developed. The final decision was to use the AD8232 breakout board because of the good documentation and support for further system development at the time.

### 3.1.2 Temperature Detection

Incorporating a temperature sensor into a telemedicine system will allow another key vital sign to be monitored. The human body temperature is very critical. Any small changes in the body's temperature can have huge implications for the patient. Measuring the body's temperature can be quick way of indicating if the patient is unwell. Measuring the temperature of a patient can be done in a number of ways. Typically, it is done:

- Under the patient's armpit
- In the patient's ear
- In the patient's mouth

It is important for the sensor to have a low thermal time constant. The thermal time constant is the time required for the thermistor to change 63.2% of the total difference temperature. The "normal" adult body temperature range is 36.1 – 37.2 degree Celsius [48]. Most adults will experience a fever if their body temperature is over 38 degrees. Hypothermia happens when the body temperature is too low which can be deadly. Heatstroke is the term given when the body temperature is too high which can also be deadly.

A wide range of temperature sensors have been developed incorporating a number of different technologies. There are two ways to measure the temperature [49]:

- Contact Temperature sensors – These sensors are physically in contact with the object being measured.
- Non-Contact – These sensors use convection and radiation to monitor the changes in temperature. These do not have to be in contact with the object being measured.

Some of the more popular temperatures sensors are [49]:

- Thermostat
- Thermistor
- Resistive Temperature Detectors
- Thermocouple

### 3.1.2.1 Thermostat

A thermostat is effectivity an on/off sensor; recording if the temperature is above the set point or below the set point. The set point at which it turns on/off is defined by the physical makeup of the thermostat. A thermostat is made up of different metals which have different linear expansion rates. As the sensor is heated, the metal expands, creating the on/off effect. Figure 3.7 shows a typical thermostat

Figure 3.7 – Thermostat [50]

### 3.1.2.2 Thermistor

A thermistor is a special type of resistor in which the physical resistance changes when it is exposed to a change in temperature. There are two types of thermistor:

- NTC – Negative temperature coefficient – The higher the temperature the lower the resistance
- PTC – Positive temperature coefficient – The lower the temperature the lower the resistance

As the thermistor is a passive device, current needs to be passed through the sensor to get a temperature. Typically to determine the resistance of the thermistor in a microcontroller circuit, another resistor of known value is placed in series with the thermistor. Then the voltage drop caused by the change in resistance can be measured by the microcontroller. Typically, the thermistor is covered in a glass which makes them easy to break. However, they have a quick response to a change in temperature and are very accurate with a high repeatability. Figure 3.8 below shows a typical thermistor

Figure 3.8 - NTC Thermistor [51]

### 3.1.2.3 Resistive Temperature Detectors

A resistive temperature detector (RTD) is similar to the thermistor, in which the resistance changes with temperature. The key difference is the output is extremely linear and gives very accurate measurements. The down side of the RTD is they typically have a poor thermal sensitivity. That means a change in temperature only produces a small output change, in the order of 1 ohm per degree change. Figure 3.9 below shows an RTD

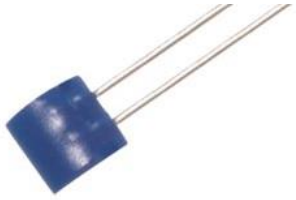


Figure 3.9 - Resistive Temperature Detector (RTD) [52]

### 3.1.2.4 Thermocouple

Thermocouples are commonly used. Effectively, these work by measuring a voltage difference across two points which can then be used to determine the temperature. Typically, a thermocouple is composed of two different metals joined at a junction. The point, at which the two different metals are joined, is placed in the heat (hot measuring junction). Another point is used as a second measuring junction, which is out of the heat (cold measuring junction). A voltage difference is created between the two junctions, which varies with temperature.

Thermocouples are easy to use and respond very rapidly to changes in temperature. They are small in size and have a very wide sensing range. Due to the wide sensing range, thermocouples are commonly used in the industrial applications. Thermocouples have a relatively high cost. The lowest priced thermocouple on RS components was $8.56 [53]. Figure 3.10 shows a typical thermocouple.

Figure 3.10 – Thermocouple [53]

### 3.1.2.5 Non-Contact Temperature sensors

Non-Contact temperature sensors measure the temperature without touching the object, as the name suggests. This type of temperature sensor is used when the object is difficult to access or requires a fast response time. A lot of these sensors use infrared technology for measuring the temperature. Generally, they are expensive to buy, have a wide temperature sensing range and quick response time to a change in temperature.

### 3.1.2.6 Chosen Temperature Detection Method

It was decided to use a contact temperature sensor as opposed to a non-contact type because; of the low temperature range (30-40 degrees), it's easy to establish contact between the sensor and the measuring point, and low cost. Table 3.2 below shows a summary of the temperature sensors reviewed.

| Temperature Sensor Family | Cost | Accuracy | Temperature Range | Response time |
|---|---|---|---|---|
| Thermostat | Low | Medium | Low | Medium |
| Thermistor | Low | High | Medium | High |
| RTD | Low | High | High | Low |
| Thermocouple | High | High | High | High |
| Non-Contact | High | High | High | High |

Table 3.2 - Temperature Sensor Summary

The thermistor type of temperature detection was selected because of its accuracy, response speed and repeatability, and low cost which are all important properties for tools within the healthcare industry.

The speed at which a thermistor reacts to a change in temperature is measured by the thermal time constant. A low thermal time constant indicates its responses very quickly to a change in temperature.

An NTC thermistor was sourced from RS Components shown in Figure 3.11 below. The key factor in selecting the thermistor was its low thermal time constant and low price. The selected thermistor had the lowest thermal time constant available from RS Components, (0.5 seconds), and costs $6.49 [54]. Below is an image of the chosen thermistor.

Figure 3.11 - Chosen temperature sensor - NTC thermistor [54]

## 3.2 System Design and Architecture

This section documents the system design and architecture to realise the project aim of developing a feasible and effective remote diagnosis system for healthcare. To achieve this aim, the following points are key design factors which will be discussed within this section.

- Communication
- Software
- Database
- Data processing
- Power supply

### 3.2.1 Communication System Design

Communication is a very important part of any telemedicine device. By definition, telemedicine is the remote diagnosis and treatment of patients by means of telecommunications which means any device needs to be able to communicate to remote locations. Possible communication methods including their advantages and disadvantages are discussed in section 2.4.2 Communication.

Communication is required to get the data from the device in a remote location to a healthcare professional. Using a cloud facility, this can be split into two sections:

- Data from the device to a cloud database
- Data from the database to the healthcare professional

Using a cloud database allows information to be written from anywhere and data to be read from anywhere. It means that the device does not require a direct communication link to the doctor, making the system design easier. The cloud storage facility will be made up of a database. The system needs to handle multiple users at the same time and be a hub for storing a large number of user data. Sending data to a cloud storage facility requires internet access and can be realised in a number of different ways:

- Directly send the data from the device to the cloud using WLAN
- Send the data from the device to a local processor before sending it to the cloud over WLAN

Sending data from the device directly to the cloud is possible but this would require the device to manage and process the user login. This is required to ensure that data is handled correctly and saved in the correct locations within the database. Adding the user login to the physical device will make it larger and more complex to operate because user feedback (screen) is also required as well as user inputs (buttons). The patient would have to be able to login to authenticate that the information is coming from them. It therefore makes more sense to use a processing system to handle information coming in from the device and then saving this to the database. This allows the device to be smaller and easier to use. WLAN communication is required to save the data from the processing system to the database. Typically, unless a device has more than one card or channel, it cannot communicate with more than one device at the same time, i.e. the processing system cannot be writing data to the database using WLAN at the same time as reading data from another device using WLAN. This leaves three options:

- Use two forms of communication. One from the device to the processing system and another different form for writing data to the database from the processing system.
- Use WLAN for both communication links. This means that data will need to be saved and buffered in the processing system and saved to the database at a later time, which is not ideal as the data is no longer real-time.
- Using two WLAN channels. The processing system is then required to have two WLAN cards.

For ease of use, it was decided to use two different forms of communication which can run simultaneously. It was chosen to create a smartphone/tablet application (app) as a processing system. There are over 5.9 billion mobile phone subscribers worldwide [11] and smartphones are only going to become more popular. An app is a very easy and user-friendly way of communicating to the device and more importantly a gateway for patients to easily login to ensure that information is stored correctly and to authenticate that the information is from them. Typically, smartphones have two forms of communication:

- WLAN (WiFi)/3G/4G
- Bluetooth

WLAN is required as the communication channel between the app and the database. To keep the system simple, it was chosen to use Bluetooth as the communication link between the device and the app.

Bluetooth will be used so the device can send/receive information from a smartphone app. Then using WLAN, the app will be able to read and write data from a cloud database. The healthcare professional will be able to access information from the database using a smartphone application as well as using WLAN. These forms of communication are commonly available and easy to assess. Figure 3.12 shows the proposed communication links.

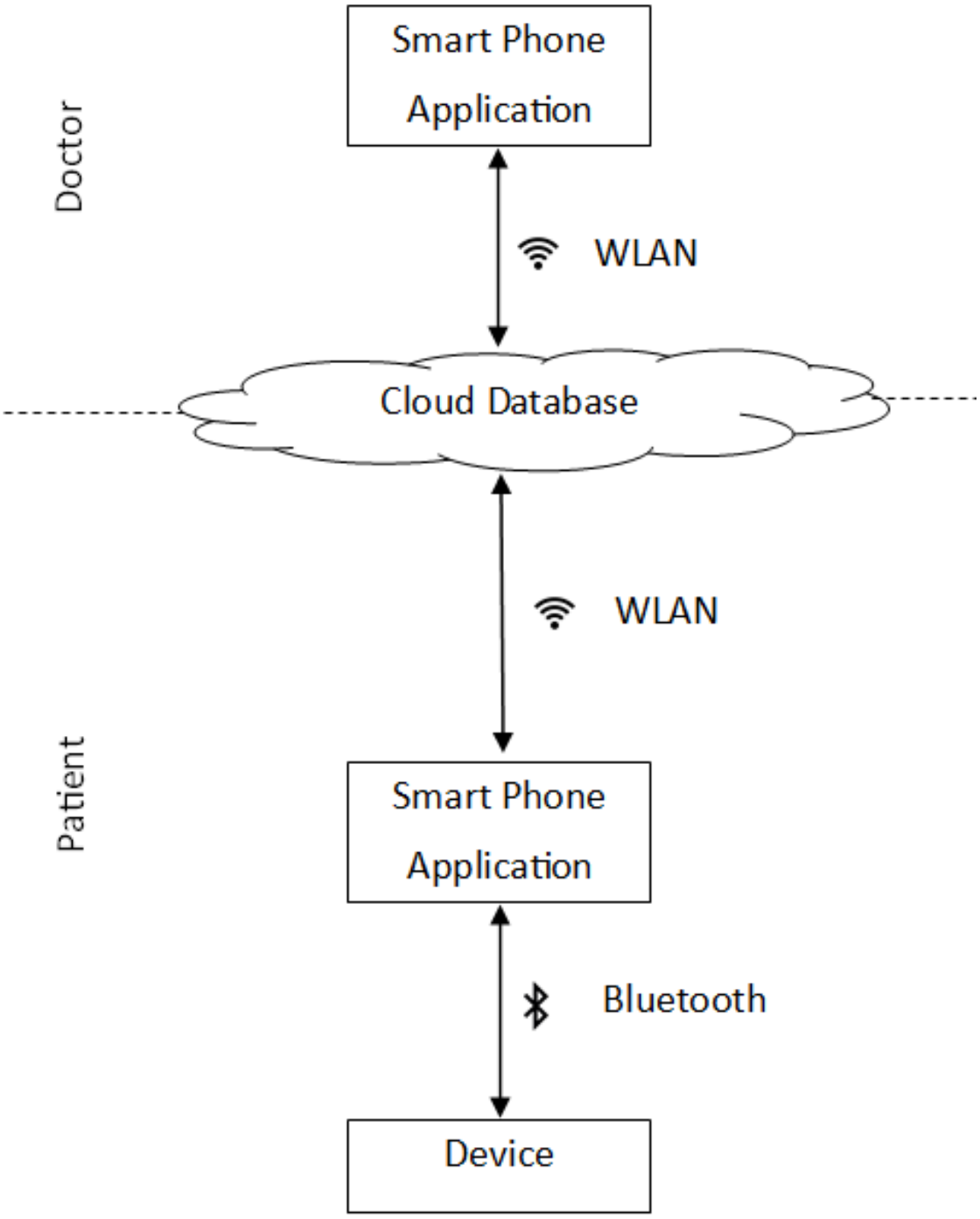



Figure 3.12 - Proposed Communication Links

### 3.2.2 Software Design

As stated in the above section, 3.2.1 Communication System Design, a smartphone application is to be developed to create a communication link between the device and the cloud storage system. This will also allow the user to easily interact with the system. Smartphone applications are generally written in Java. There are two main operating systems (OS) running on smartphones. These are:

- Android
- IOS (IPhone)

The Android OS was chosen to create the application because it is free to develop an app, it is well supported, and it is easy to install and run on a smartphone. A decision was made to develop the application in software called Intellij which is a Java integrated development environment (IDE) for developing computer software.

Intellij is well supported with a developed IDE. Android Studio is another, commonly used software used for developing applications for Android which is based the on the Intellij IDE.

The application has three important jobs that it must be able to do, they are as follows:

- User login and authentication. The system needs to know who the user is and where to save the information to, so it can be accessed by a healthcare professional.
- Communicate to the device over Bluetooth. This will involve sending commands and receiving data back.
- Communicate to the cloud database to save the capture information from the device.

It is also very important that it is simple and easy to use as this is what the user will be using to interface with the system. Making the application functional but also aesthetically pleasing will be vital in the end solution.

### 3.2.3 Database Design

As stated, in the above section, 3.2.1 Communication System Design, a database will be used to transfer the information from the patient to the healthcare professional. Due to the nature of the application, a NoSQL database structure was chosen. Google Firebase was selected to develop and host the NoSQL database.

Firebase is a real-time database which means once data is written to the database; it is synced across connected devices in milliseconds. Firebase is simple to use with an intuitive and easy to use API (Application Programming Interface) which can be added to the Intellij IDE project. Using Google Firebase also offers other benefits:

- Storage facilities which can be used for backing up data.
- Authentication of users which can be used to handle user login and to access the database and storage facilities.

### 3.2.4 Data Processing Design

A microcontroller is ideal for controlling the device and processing information from sensors. There are numerous microcontrollers available on the market and several different options were considered. When selecting the microcontroller there were no predefined requirements. As Arduino microcontrollers are widely used and well documented, interest was focused on Arduino's. In selecting the microcontroller, a number of important factors need to be considered:

- Microcontroller size. Smaller is better as this affects the physical size of the end device.
- Speed of the processor – faster is better, as it is required to process a large amount of analogue data quickly.
- Large on-board memory for storing data and variables.

A number of Arduino's were reviewed which are summarised in Table 3.3 below.

| Arduino | Clock Speed (MHz) | Processor Size (Bit) | Flash Memory (kB) | Size (mm) | Special Features |
|---|---|---|---|---|---|
| 101 [55] | 32 | 32 | 196 | 69.6 x 53.4 | Bluetooth BLE and Accelerometer |
| Due [56] | 84 | 32 | 512 | 101.52 x 53.3 | 2 Analog Outputs |
| Zero [57] | 48 | 32 | 256 | 69.6 x 53.4 | 1 x 10 Bit DAC |
| Nano [58] | 16 | 8 | 16 | 45 x 18 | NA |

Table 3.3 - Summary of Arduino Microcontrollers Available on the Market

The Arduino Nano is small, but it does not have the required processing power or storage. This controller was tested but difficulties were encountered due to its small memory size and low processing speed.

Although Arduino Due is powerful, it was found to be physically too large.

The Arduino 101/Zero are a modern version of the Arduino Uno. Based on an Arduino Uno's shape, size and IO. These both have much greater processing power and memory compared to the Uno. The Arduino 101 has built in Bluetooth (BLE) and 6 axis accelerometer/gyro. These additional features will be utilised and reduce the need for add-on components like a communication module. The Arduino Zero has an improved, faster processer with more memory.

Arduino 101 was finally selected as the microcontroller for the data processing system due to its extra functionality. If, during development, it is found that the 101 lacks processing power or memory then an Arduino Zero could be considered as a suitable replacement for the 101.

### 3.2.5 Power Supply Design

Batteries are made up from a number of cells in series to get the required voltage. There are many different types of batteries with diverse chemical make-ups giving each type of battery different properties. There are two families of batteries; primary and secondary which are non-rechargeable and rechargeable respectively.

Primary batteries typically have a high-energy density, made for single use only, and perform well in low drain applications. Secondary batteries have a lower energy density however, they perform well in high drain applications and being rechargeable, they do not have to be replaced. It was chosen to use a secondary battery type because they are rechargeable, meaning the device does not have to be designed to allow easy access to the batteries.

Three key characteristics of batteries, which can be measured and are useful when comparing batteries, are:

- The specific energy
- The cost
- The power

The specific energy takes into account the voltage of the cell. The specific energy is measure in watt hours (Wh) and is defined as:

$$Specific\ Energy\ (Wh) = Current\ Capacity\ (Ah)\ x\ Voltage\ (v)$$

**Equation 1**

The battery power is defined as:

$$Power\ (W) = Discharge\ Rate(A)\ x\ Voltage\ (v)$$

**Equation 2**

Figure 3.13 below shows the specific energy per kg (Wh/kg) vs different commonly available battery chemistries.

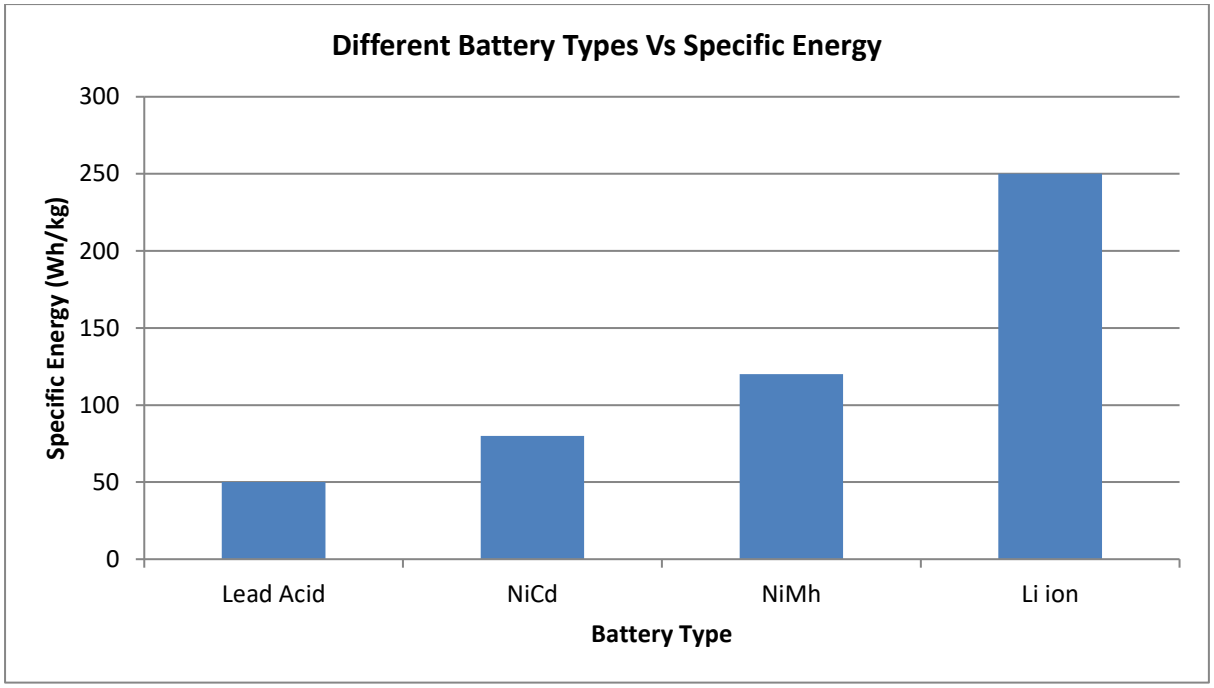



Figure 3.13 - Different Battery Types vs Specific Energy [59]

The Li ion (lithium ion) battery has the highest energy per kilogram of battery. Another key property for selecting batteries is the physical cost. The main driver in the physical cost of battery is the chemical makeup. Figure 3.14 below shows cost of per kWh for each different battery types.

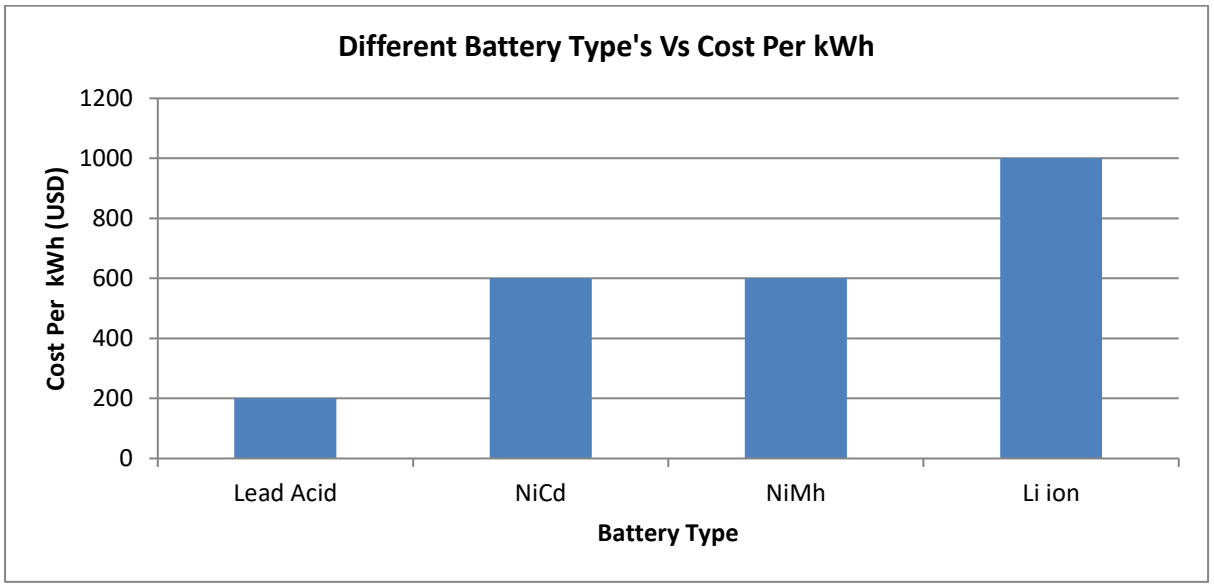



Figure 3.14 - Different Battery Type's vs Cost Per kWh [59]

When comparing Figure 3.13 with Figure 3.14 shown above, it can be observed that the battery types with a higher specific energy cost more. What these graphs do not show is the number of cycle times. In general Li ion batteries have the higher number of recharge cycles compared to lead acid.

Each the four battery types shown in the graphs are discussed in more detail below.

#### *3.2.5.1 Lead Acid*

Lead acid batteries have been around for a long time. Chemically, as the name suggests they contain lead which is cheap when considering the costs-per-watt. There are few batteries that can deliver a bulk power as cheaply. Below show the advantages and disadvantages for the lead acid battery type [60].

Advantages:

- Low cost
- Low self-discharge
- High discharge currents
- High temperature performance

Disadvantages:

- Low specific energy
- Slow charge
- Limited cycle life

#### *3.2.5.2 Nickel – cadmium*

Nickel – cadmium (NiCd) battery type is commonly used by hobbyists for radio control cars and planes etc. Listed below are the advantages and disadvantages for the NiCd battery type [61].

Advantages:

- Quick charging
- Good load performance
- Long shelf-life
- Low temperature performance
- Priced reasonably

Disadvantages:

- Low specific energy
- High self-discharge
- Memory effect

#### *3.2.5.3 Nickel-metal-hydride*

Nickel-metal-hydride (NiMH) provides 40% higher specific energy than NiCd batteries but had a major drawback with a quicker self-discharge rate, 20% in the first 24 hours. This battery type has been developed into AA and AAA sizes to try and replace the traditional alkaline battery. The list below shows the advantages and disadvantages for the NiMH battery type [61].

Advantages:

- Less prone to memory than NiCd
- Higher specific energy then NiCd

Disadvantages:

- High self-discharge

#### *3.2.5.4 Lithium-ion*

There are many different types of Lithium-ion (Li-ion) batteries. A Lithium Polymer (Lipo) battery falls into this family. LiPo batteries are also very commonly used in the hobbyist area. Listed below are the advantages and disadvantages for the Li-ion battery type [62].

Advantages:

- High specific energy

Disadvantages:

- Cost
- Prone to memory

### *3.2.5.5 Chosen Battery Type*

The system is required to be low cost, but also to operate for a long time without the need for recharging to make it user friendly. To power the selected microcontroller, Arduino 101, a minimum of 5V is required. Table 3.4 below summary's the four secondary battery types reviewed.

| Battery Type | Energy Density | Discharge Rate | Self-Discharge Rate | Cost | Memory Prone |
|---|---|---|---|---|---|
| Lead Acid | High | Low | Low | Low | Medium |
| NiCd | Low | High | High | Low | High |
| NiMH | Low | High | High | Low | Low |
| Li-ion | High | Medium | Medium | Medium | Low |

Table 3.4 - Power Supply Design Summary

A Lithium Polymer (LiPo) battery type was selected to power the system because of its high-energy density and low self-discharge, it also has a more forgiving nature compared to other battery types. The cost of the battery is justified by the gains it will provide to the system. When selecting the LiPo cells to create the battery, the smallest cells available were selected. Being LiPo battery the nominally cell voltage is 3.7V meaning that two cells are required. The selected battery specifications are shown below [63]:

- Capacity: 740 mAh
- Discharge rate: 14.8A
- Specific energy: 2.738 Wh per cell
- Size: 64x35x5mm
- Cost: $3.02 AUD

An image of the cell is shown in Figure 3.15 below.


Figure 3.15 - Selected Battery Cell [63]

### 3.2.6 System Design and Architecture Summary

Figure 3.16 below shows the designed system architecture for achieving the aim of the project. It is a summary of all the decisions made in this section, Remote Diagnosis System Design. An Android application will be developed in an Intellij programming environment to handle communication links to the cloud database and to the device. It will also be the user interface which needs to be user friendly to cater for patients familiar with using apps as well as those less experienced. A NoSQL cloud database (firebase) will be used to store data, creating a link between the patient and the doctor. The device will use an Arduino 101 and will capture two key vital signs; temperature using an NTC thermistor and heart rate using electrodes and an AD8232 chip.

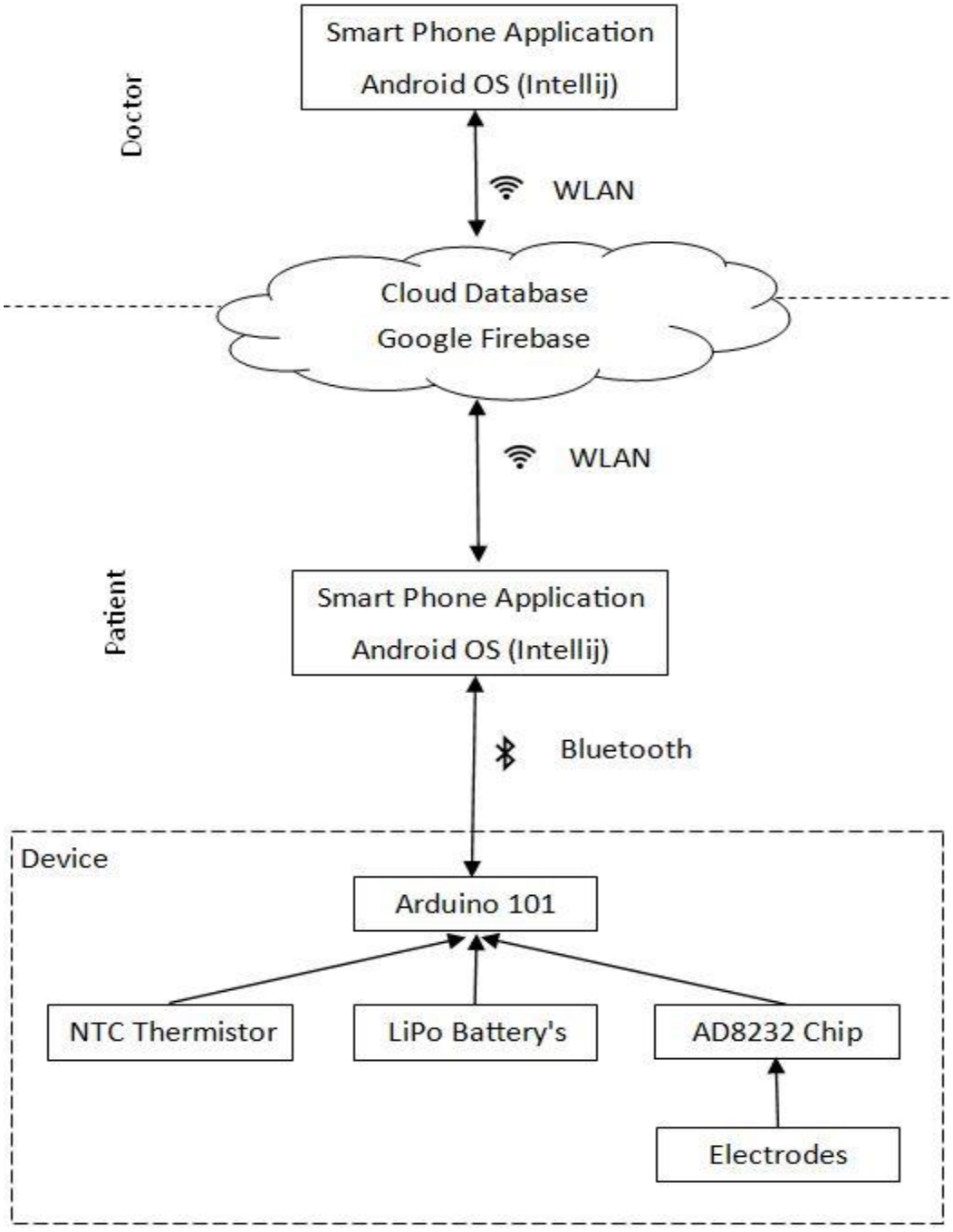



Figure 3.16- System Design and Architecture Flow Diagram

# 4. ECG Signal Acquisition and Filtering

## 4.1 Electrode Placement

One of the major tasks is to develop a small system that is able to accurately capture ECG signals. Electrode placement is crucial to get valid information. The positioning of the electrodes plays a vital role in the system to be developed as this is a main factor in the size of the device. The developers and manufactures of the chosen AD8232 break out board recommended using a three-lead electrode approach. The three-lead electrode approach is also widely used in medical ECG equipment; therefore, it was selected as the starting platform and the resulting ECG signal is used as a bench mark to compare and evaluate other approaches. Figure 4.1 illustrates the suggested electrode arrangement using three-lead electrodes with the AD8232 board. Figure 4.2 shows the corresponding ECG signal. The signal has well defined pulses with some noise.

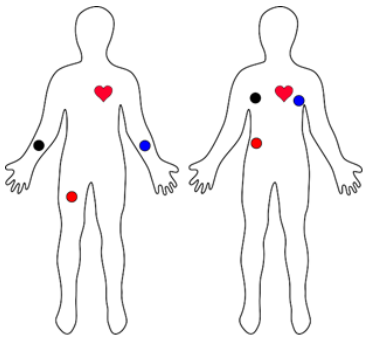


Figure 4.1 - AD8232 Heart Rate Monitor Electrode Placement, Position 1 [64]

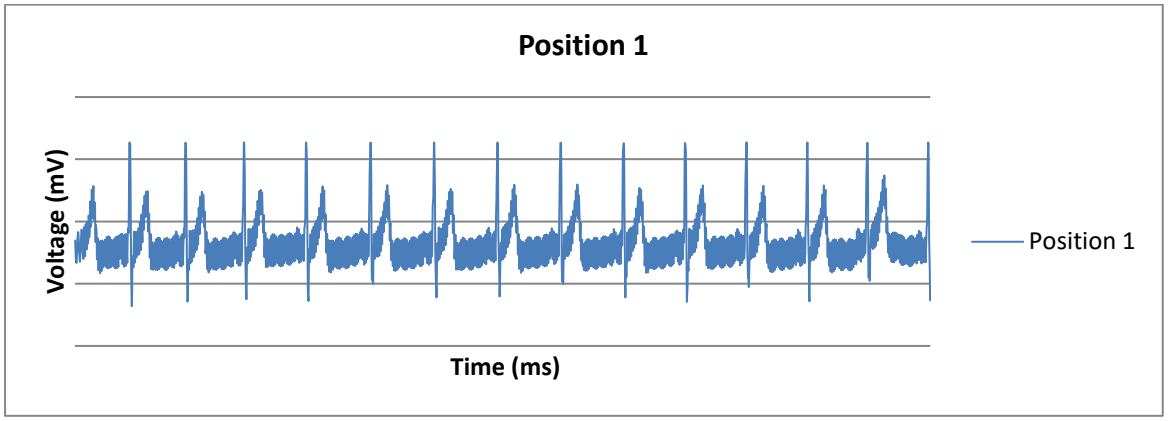



Figure 4.2 - ECG Wave Captured with Electrodes Placed in Position 1

Although this electrode arrangement obtains clear ECG signals, it will be difficult to implement in a small or hand-held portable device due to the required position of the electrodes as shown in Figure 4.1.

A study was conducted to investigate if the electrodes could be moved closer together. Different electrode arrangements were considered. Figure 4.3 and Figure 4.4 are two different electrode arrangements that purposely put the three electrodes very close. Figure 4.5 presents the outcome of these two arrangements vs the suggested electrode placement shown in Figure 4.1.

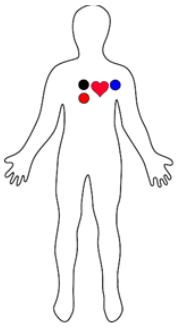


Figure 4.3 - Electrodes Placement Position 2

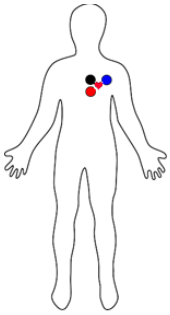


Figure 4.4 - Electrodes Placement Position 3

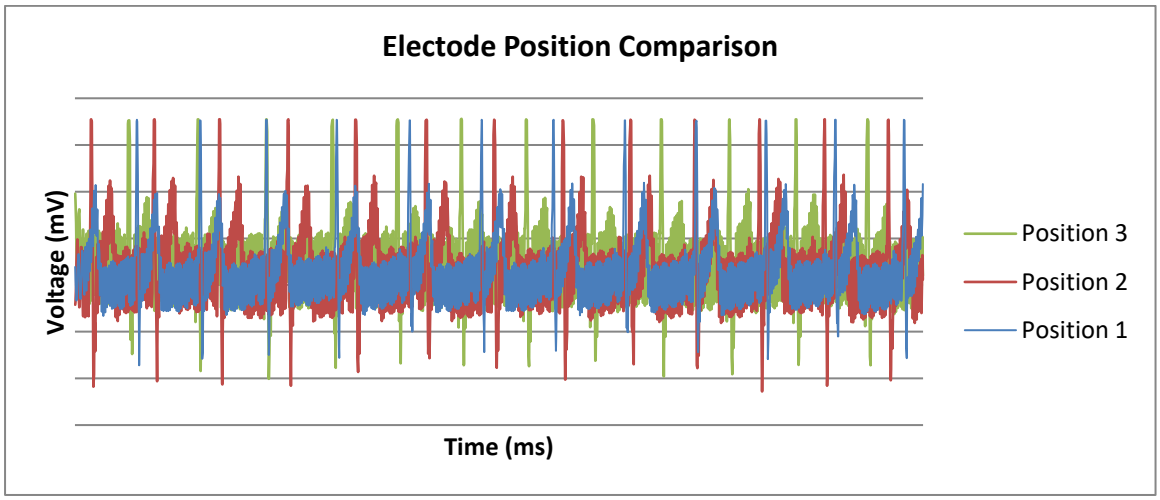



Figure 4.5 - 3 Lead Electrode Placement Comparisons

As the electrodes are placed closer to the heart, the signal strength remains constant (the pulses remain the same strength), however there is more noise in the signal. As the signal is still well defined, the data can easy be processed in a microcontroller to determine the heart rate.

## 4.2 Investigation into the Possibility of Using Two Electrodes

Using three electrodes means that the size of the device to be developed would have to be large enough to accommodate them. If two electrodes could be used instead of three, it would dramatically reduce the size of the device.

Further study found that the datasheet for the AD8232 chip also supports a two-electrode arrangement, where the manufactures suggests a two-lead electrode arrangement as shown in Figure 4.6. The break out board for this is shown in Figure 4.7. The two-lead system removes the third electrode which is located in the right leg (RL) direction. This input is instead connected to both the Right arm (RA) and Left arm (LA).

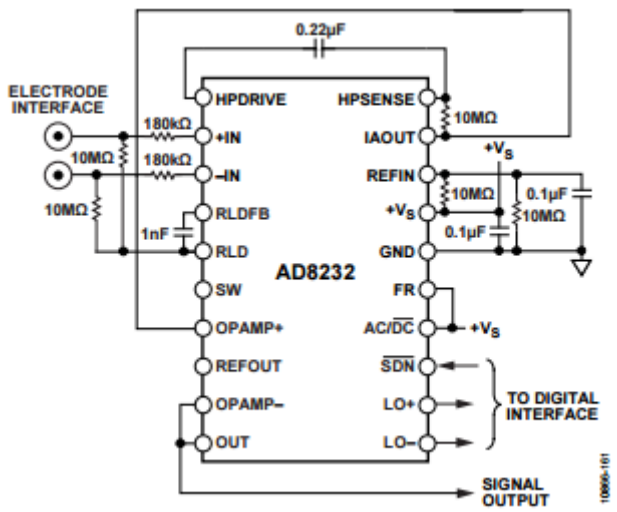



Figure 4.6 - AD8232 Data Sheet 2 Electrode Example Circuit [65]

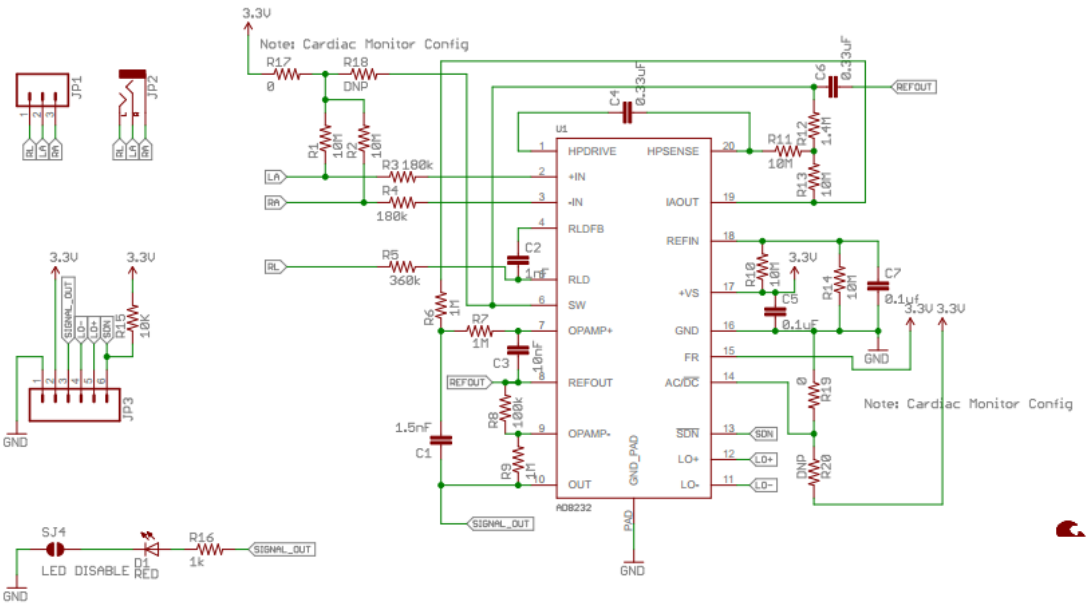



Figure 4.7 - AD8232 Breakout Board Circuit [66]

The RL electrode input is wired to a 360 KΩ resistor and the two driving electrodes LA and RA are connected to a 3.3V volt supply through a series of resistors. Modifying the AD8232 break out board from a three lead to a two lead ECG circuit is next to impossible. Figure 4.8 shows the differences between the AD8232 breakout board and how the datasheet suggests wiring the AD8232 chip for the two electrode ECG detection.

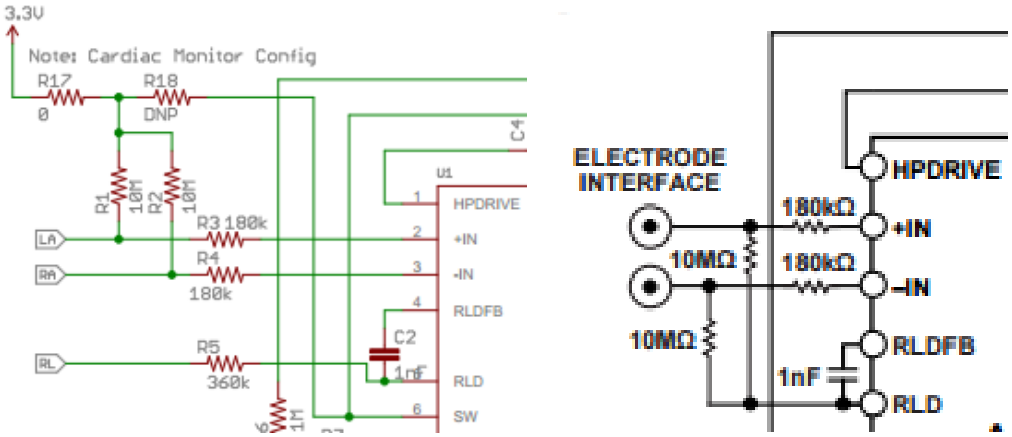



Figure 4.8 – Comparison between AD8232 Breakout Board Circuit and the AD8232 Datasheet 2-Lead Example Circuit

Following the idea suggested in the data sheet of connecting RL to RA and LA through a 10MΩ resistor, a trial was established. The aim of the trail was to determine the best resistances value to use when connecting the RL to LA/RA as the AD8232 circuit has a 360KΩ resistor in series with the input as shown in Figure 4.8. In running the trial, it was found that adding a capacitator across LA and RA electrodes could help remove noise from the signal.

Figure 4.9 is the test circuit created on a bread board to help determine the ideal values of the resistors and capacitor to obtain a useful two-lead electrode signal.

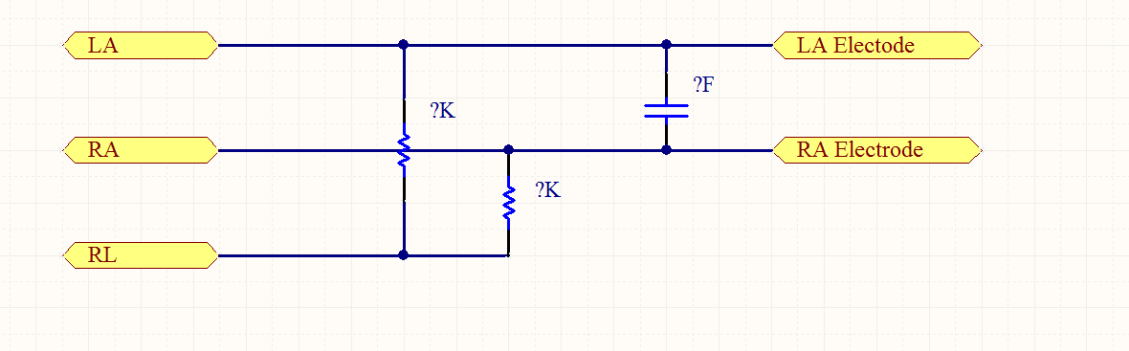



Figure 4.9 - Test Circuit for Two-Lead Electrode Arrangement

Using a number of resistors and capacitors that were on hand, a quick and simple trial was undertaken.

Six tests were conducted to help determine which set of values for the resistors and capacitor resulted in the best ECG wave. Table 4.1 contains the values of the resistors and capacitor that were used in the six tests and Figure 4.10-Figure 4.15 shows the outcomes of the six tests.

The outcomes from tests shown in Figure 4.10-Figure 4.15 have the same number of data points (1000) and have the same scale on the y axis. This is to help compare and determine which combination is the best. The position of the electrodes on the chest was kept the same for all six tests.

| Test | Resistance | Capacitance |
|---|---|---|
| 1 | 50KΩ | Na |
| 2 | 150KΩ | Na |
| 3 | 150KΩ | 0.1 uF |
| 4 | 150KΩ | 1uF |
| 5 | 2.4MΩ | 1uF |
| 6 | 240KΩ | 0.1uF |

Table 4.1 - The Values of the Resistors and Capacitor in the Six Tests

Test 1- No capacitor between electrode inputs and 50KΩ from RL to LA and RA

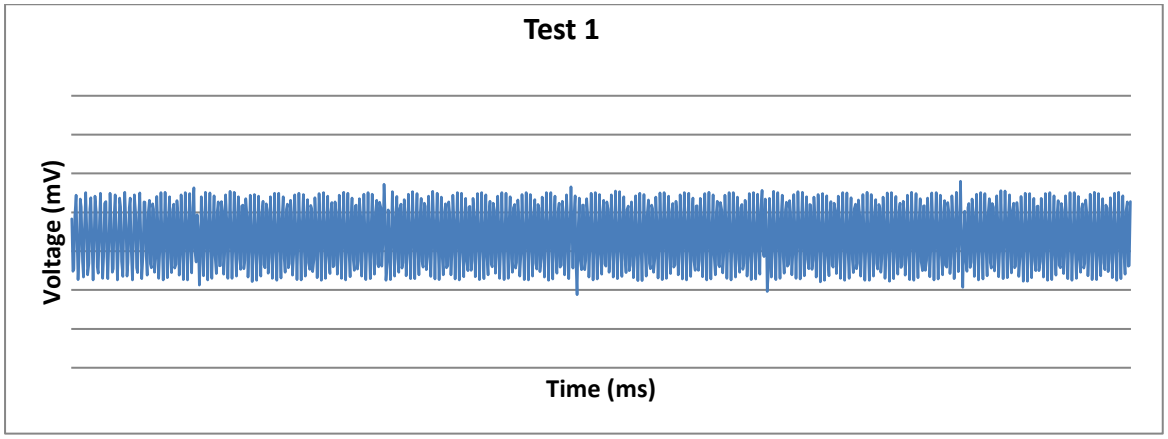



Figure 4.10 - Value Checking - Test 1 Results

Test 2 - No capacitor between electrode inputs and 150KΩ from RL to LA and RA

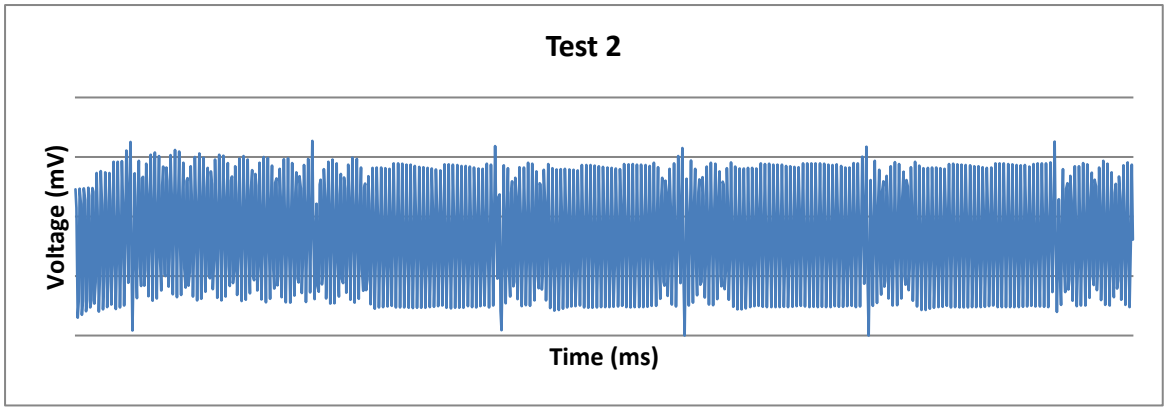



Figure 4.11 - Value Checking - Test 2 Results

Test 3 - 0.1uF capacitor between electrode inputs and 150KΩ from RL to LA and RA

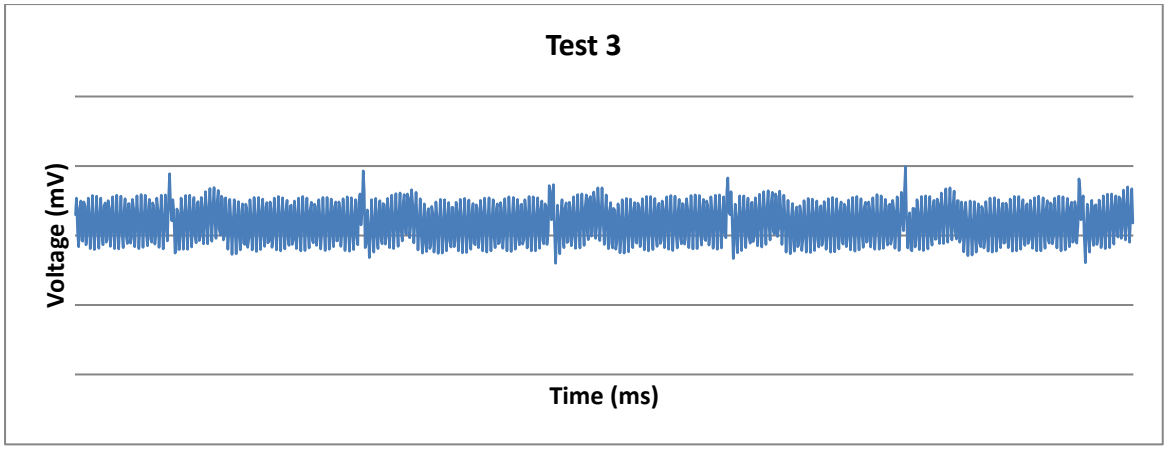



Figure 4.12 - Value Checking - Test 3 Results

Test 4-1uF capacitor between electrode inputs and 150KΩ from RL to LA and RA

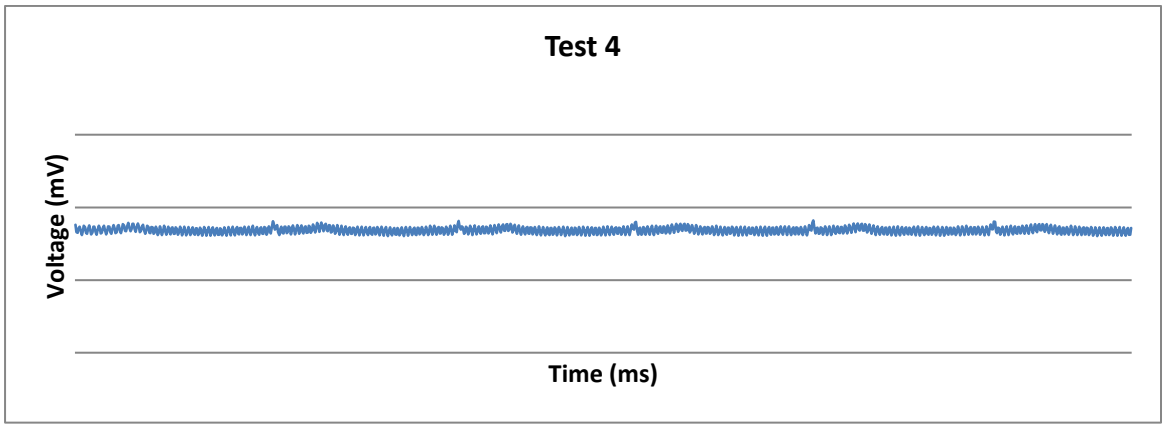



Figure 4.13 - Value Checking - Test 4 Results

Test 5 - 1uF capacitor between electrode inputs and 2.4MΩ from RL to LA and RA

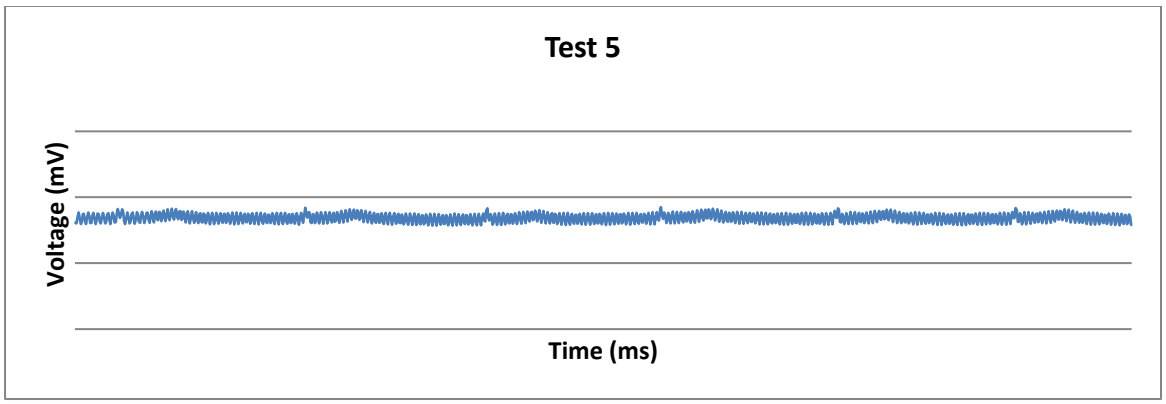



Figure 4.14 - Value Checking - Test 5 Results

Test 6 - 0.1uF capacitor between electrode inputs and 240KΩ from RL to LA and RA

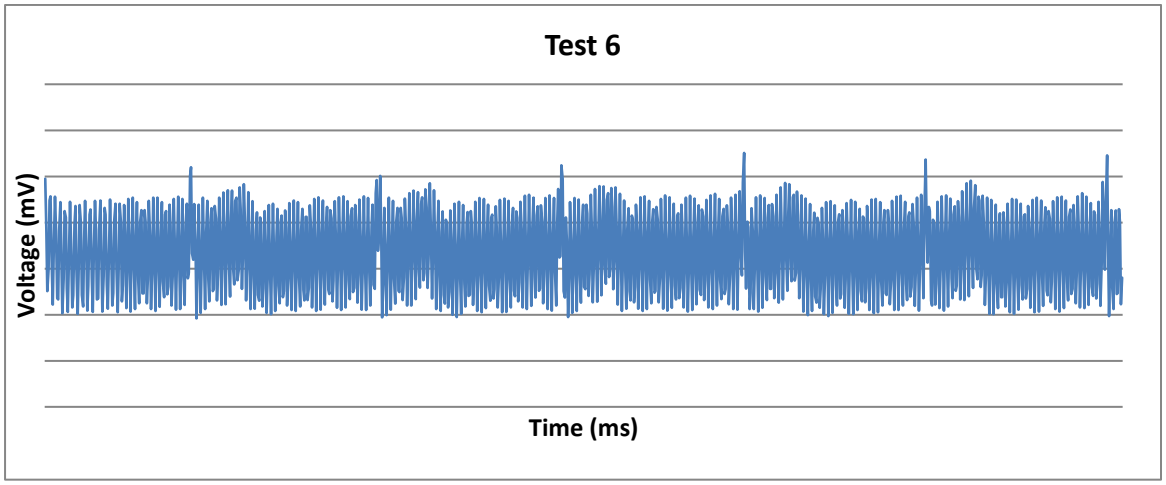



Figure 4.15 - Value Checking - Test 6 Results

The outcomes of the six tests clearly show that the signals are very noisy. This could be due to the way the AD8232 breakout board is wired compared to the suggested wiring for a two-lead ECG. In summarising the results:

- A higher resistance value reduces the signal strength
- The higher the capacitor value, the more the signal is dampened
- A low resistance creates noises and reduces the pulse peak.

Studying the six outcomes, the best values came from test 3, Figure 4.12, in which the resistors were 150KΩ with a 0.1uF capacitor. Figure 4.16 shows the circuit that gave the best outcome. Obviously, the signal is very noisy and requires filtering before it can become useful data which leads to section 4.3 Filter Design and Implementation.

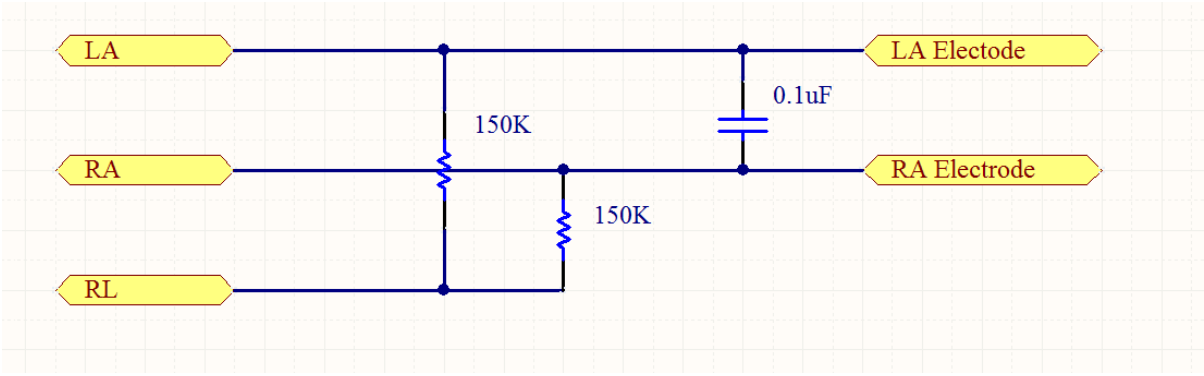



Figure 4.16 – Three-Lead Electrode into Two-lead Electrode Circuit

## 4.3 Filter Design and Implementation

Using a two-electrode arrangement, as developed in the previous section, the raw signal still requires pre-processing to remove the noise to make the signal usable. Figure 4.17 below, shows a typical signal output from the AD8232 board using the two-electrode approach documented above in section, 4.2 Investigation into the Possibility of Using Two Electrodes

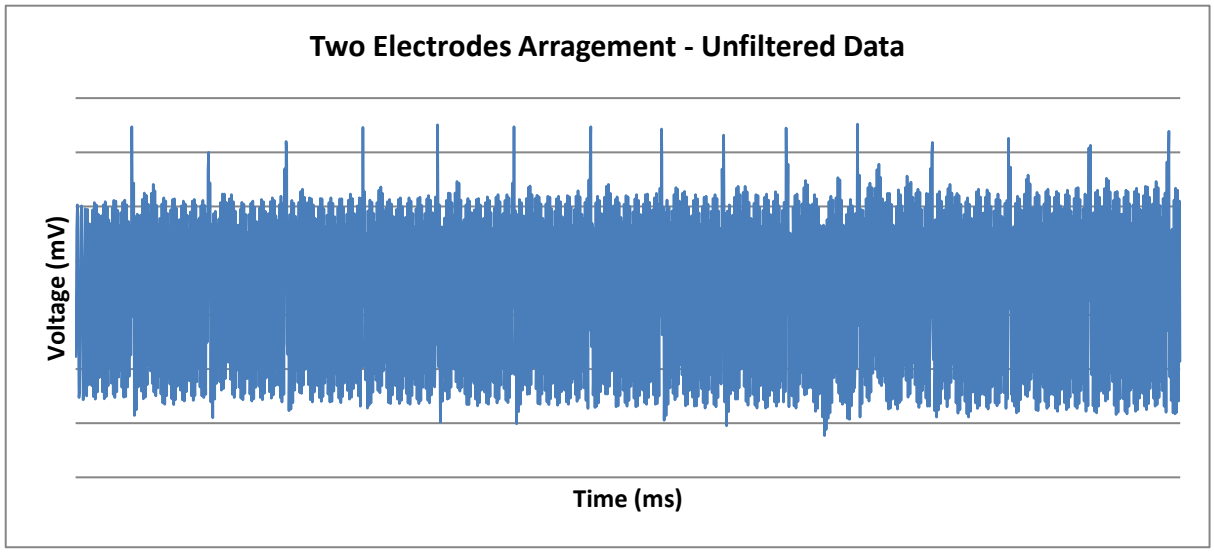



Figure 4.17 - Two Electrode Arrangement – Unfiltered Raw Data

The noise within the signal has made it infeasible to use. Using an oscilloscope to measure the frequency of the noise, revealed a 50 Hz signal was present as shown in Figure 4.18.

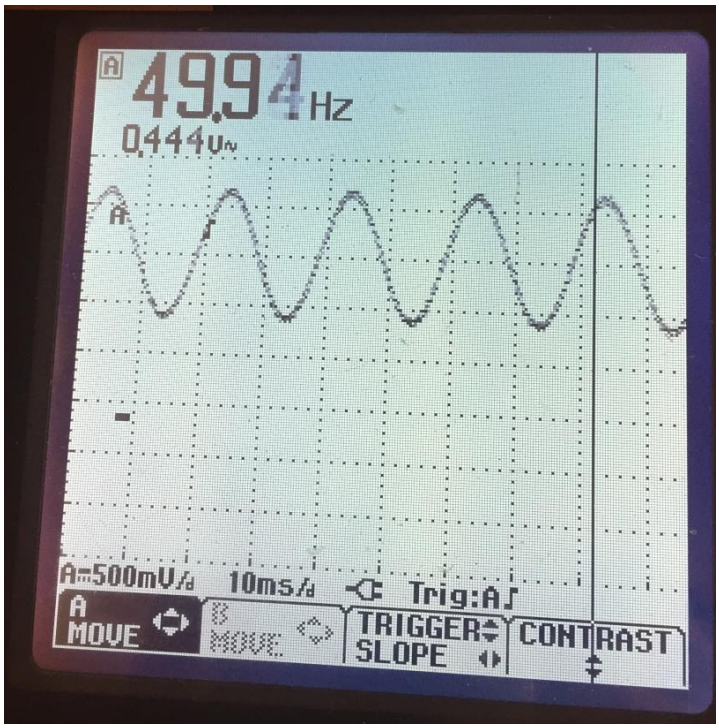



Figure 4.18 - Oscilloscope Reading of Noise in the Two Electrode Arrangement

## 4.4 Low pass filter design for ECG signal pre-processing

The fastest recorded human heart rate was 480 BPM (Beats per Minute) [67]. This patient had tachyarrhythmia, which means the heart rate exceeds the normal resting heart rate which generally is around 60-100 BPM. Using the fastest recorded heart rate as a parameter to develop a filter, gives a signal frequency of 8Hz. The filter needs to be designed to allow this frequency through. An investigation was conducted to see if a low pass filter can be designed to filter out noise and to let any signal below 8Hz through uninterrupted.

There are two main types of filters; passive and active. A passive filter uses only passive components (resistors, capacitors and inductors). Whereas an active filter may contain passive components as well as active components like op-amps. In general, both types of filters have trade-offs depending on the application.

General points about passive filters:

- No power required
- Can handle large currents and high voltages
- Reliable
- No band width limitations

General points about active filters:

- Easier to design
- Can produce high gains
- Easier to tune
- Small size

To see the difference between the two types of filters working on ECG signals, two low pass filters were designed; one passive, and one active. The cut off frequency was defined as 10Hz. The cut off frequency is a characteristic of the filter which defines the point at which the gain of the signal becomes -3db. 10Hz was selected as this is well below the noise frequency and above the frequency of the data which needs to be kept. The gain of a signal can be defined in a Bode plot, as shown in Figure 4.19. After the cut of frequency, the gain decreases 20db over each decade in a first order system.

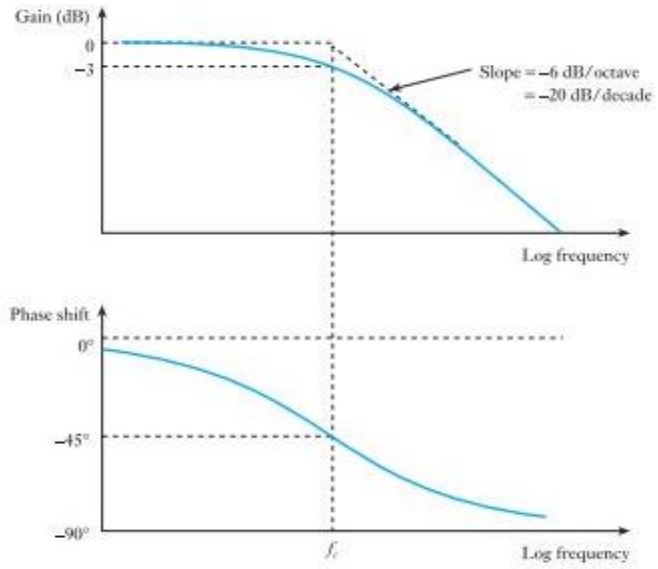


Figure 4.19 - Typical Gain Response of a Low Pass Filter [68]

Another form of filtering is an averaging function in a microcontroller. The underlining principle behind this method is that if there was no noise, the average signal would be a straight line. To achieve this, sampling needs to be done at a rate of at least twice the noise frequency. Figure 4.20 below shows an example where software averaging can remove noise from a signal. In this figure; the blue line shows the raw unfiltered signal; the green crosses show the sample points. Averaging the sample points will result in the red line, which is the filtered signal.

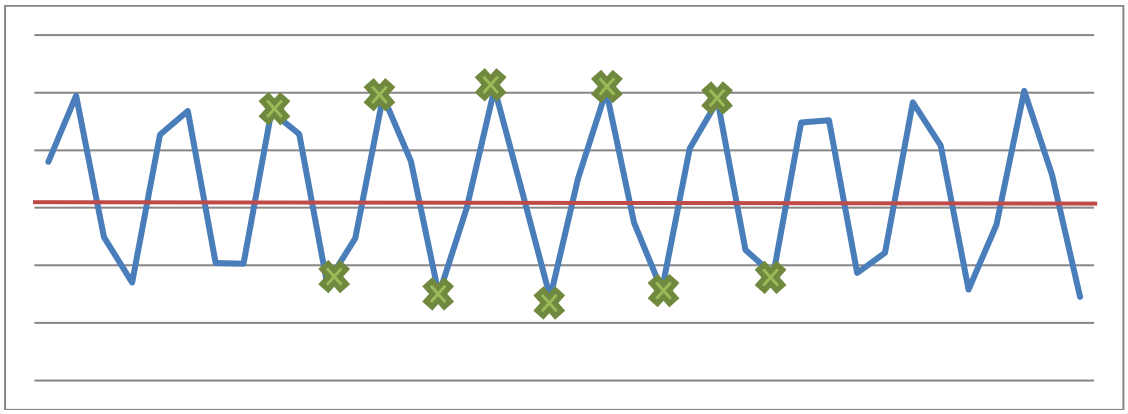


Figure 4.20 - Noise Filtering Using an Averaging Method

This shows that averaging the samples cancels each other out, removing the noise. The important part to note with this method is that the samples are taken at an even multiple of the noise frequency. This is a very simple and low-cost method for removing noises from a signal.

### 4.4.1 Active Filter Design

The design of the active filter is based around an op-amp. An online tool was used to develop the filter called analog.com [69].

Using this tool, two characteristics are required to create a filter:

- Dead band frequency. The dead band frequency is the point where the gain of the signal becomes -3db as shown in Figure 4.19. For this filter, it was selected to have the dead band frequency as 10Hz. A heart rate of 10Hz translates to 600 BPM which is above the highest recorded heart rate discussed above in section, 4.4 Low pass filter design for ECG signal pre-processing.
- Stop band frequency. This is effectively the frequency of the signal to be rejected. For this filter, it was chosen for the stop band gain of -40db to try and keep the order of the filter lower (high order filter requires more components, therefore is larger). It was chosen to have a stop band frequency of 40 Hz as it's below the 50Hz noise frequency which is required to be rejected. Effectively any signal with a frequency greater then 40Hz should get rejected, removing the noise identified in section, 4.3 Filter Design and Implementation

Entering these values into the software, a $4^{th}$ Order Butterworth low pass filter was obtained as shown in Figure 4.21. To determine the usefulness of this filter at removing noise from the signal, the circuit shown in Figure 4.22 was created on a bread board and the outcome signal is presented in Figure 4.23.

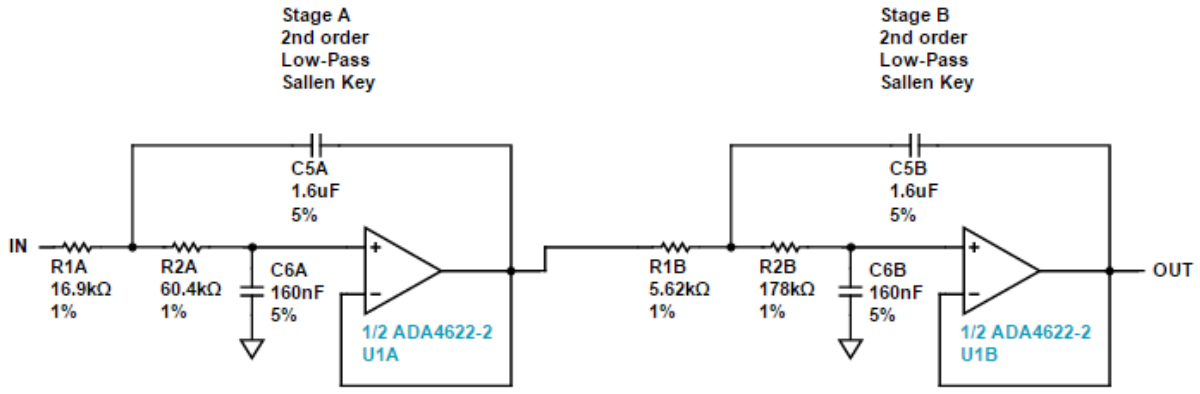



Figure 4.21 – The 4th Order, Butterworth, Low Pass Filter

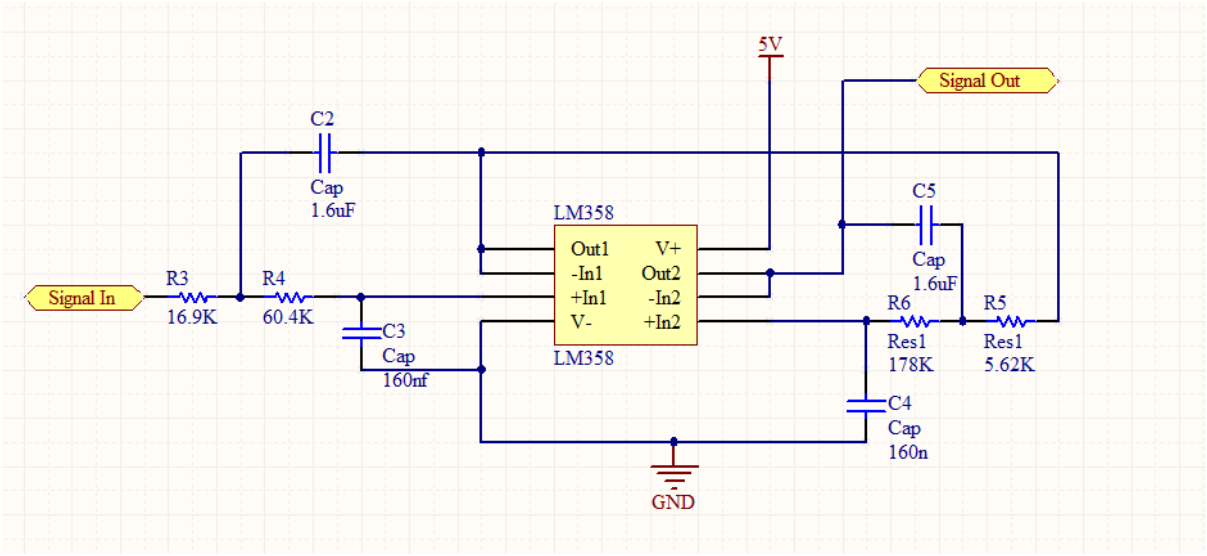



Figure 4.22 - 4th Order Butterworth, Low Pass Active Filter Schematic

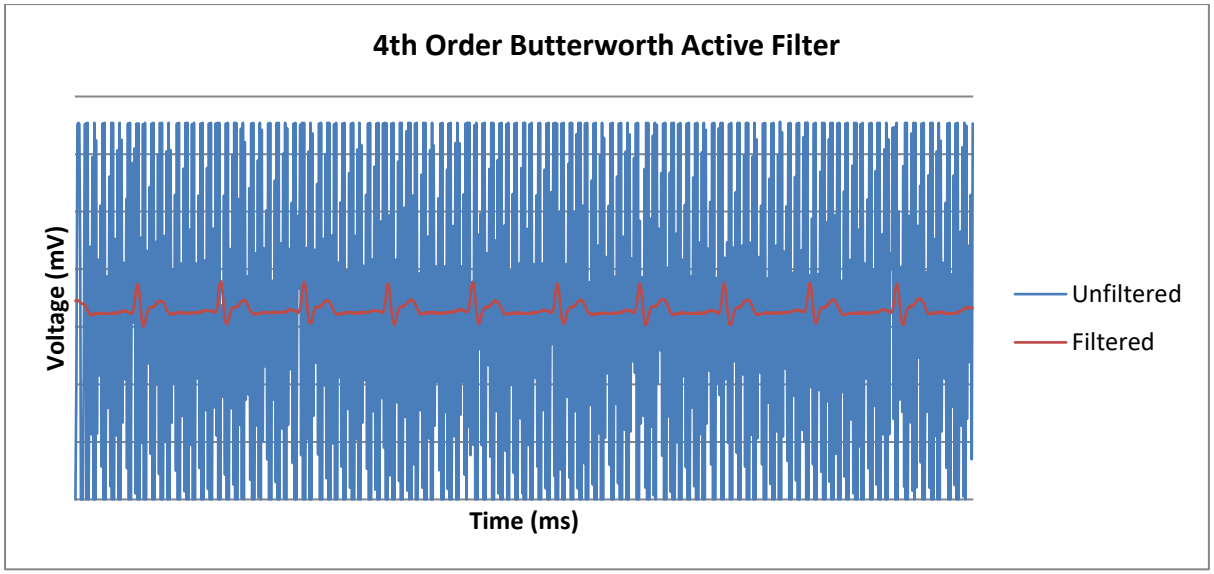



Figure 4.23 - 4th Order, Butterworth, Low Pass Filter Test

It can be observed that the developed active filter works well at removing noise. The final signal has a clear repeating pattern, which means a two-lead electrode has the potential to be used with a microcontroller for processing the ECG signal.

### 4.4.2 Passive Filter Design

A simple low pass 1st order passive filter was designed based around a capacitor and resistor which are small and readily available.

The cut off frequency for the low pass passive filter is determined using the following equation.

$$Fc = \frac{1}{2\pi\tau} = \frac{1}{2\pi RC}$$

Equation 3

Where $F_c$ is the cut off frequency, R resistance, C capacitance

By working backwards with Fc=10Hz, a capacitor value of 1uF was selected which meant the required resistance value could be calculated.

$$R = \frac{1}{10 \times 2\pi \times 0.000001} = 15.9\ K\Omega$$

The simple 1st order low pass passive filter circuit is shown in Figure 4.24.

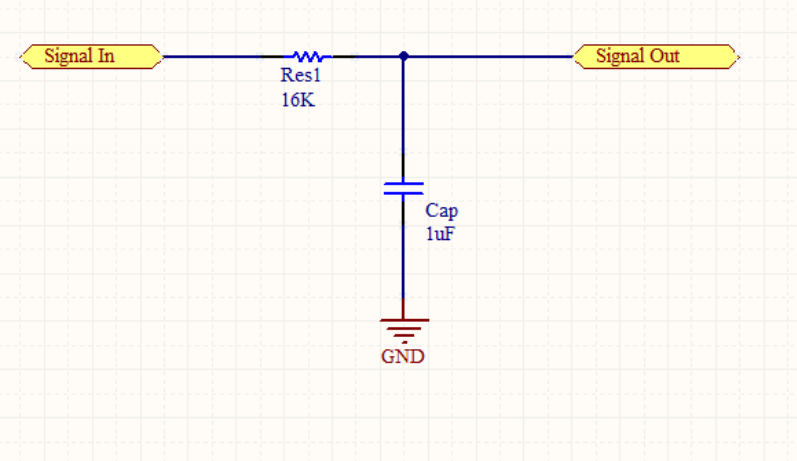



Figure 4.24 – The 1st Order Low Pass Passive Filter Circuit

To compare the two filters, a 4th order passive filter is required because the active filter developed was a 4th order filter. The simplest way to create a 4th order passive filter is to use the 1st order circuit, four times. The 4th order passive filter shown in Figure 4.25 is created.

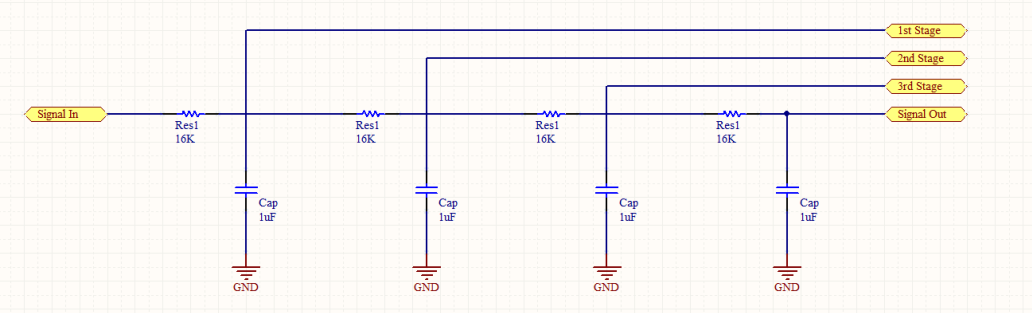



Figure 4.25 – The 4th Order Low Pass Passive Filter Circuit

This circuit was created on a bread board and tested to determine how well it removes noise from the ECG signals. The results of the tests are presented in Figure 4.26, showing the signal at various stages of the filter.

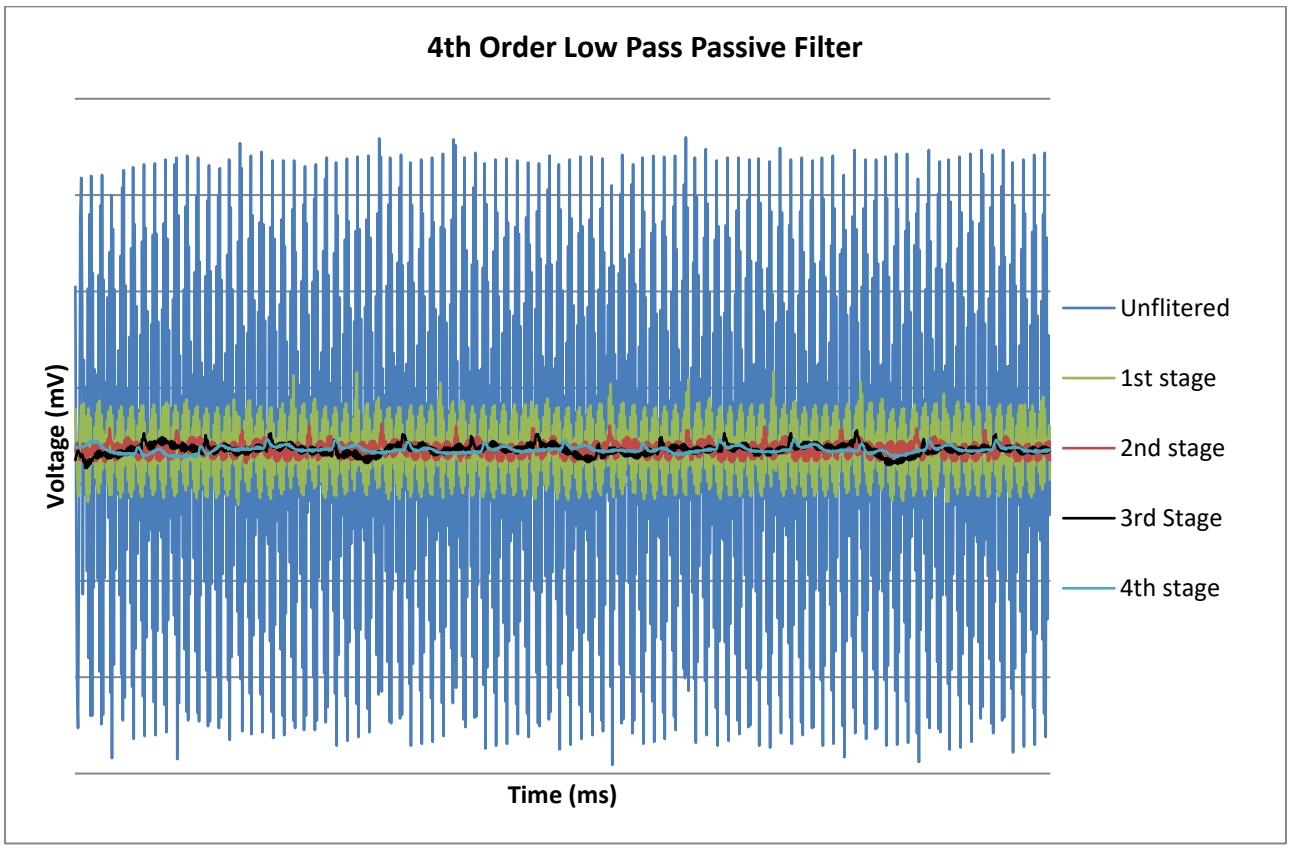



Figure 4.26 - 4th Order, Low Pass, Passive Filter Test

The passive filter can be used to remove noise from the ECG signal. The effects of each stage in the filter can be seen in Figure 4.26. The 4$^{th}$ stage signal is weak, but it removed most of the noise. Note that the signal for each stage does not line up with each other. This was because each signal had to be taken at a different times due to the time the microcontroller (Arduino) takes to print the data to the serial monitor. If they were all recorded and printed at the same time, then data would be missing, making the results less valuable

### 4.4.3 Software Averaging

A trial was also made to see if software averaging could remove noise from the ECG signal. Software averaging requires samples to be taken from the data at a rate twice or an even multiple of the noise frequency. For the noise frequency in the signal, the sample rate needs to be at 100Hz, or 200Hz, or 300Hz etc.

Below is a sample of code for calculating the running average using a 500Hz sample rate.

```
void loop() {

        //Save analogue value so it can't change
        int rawvalue = analogRead(A0);
        Serial.print(rawvalue);
        Serial.print('\t');

        //Remove last value from average and add the new value
        average = average - ave[count];
        ave[count] = rawvalue;
        average = average + ave[count];
        count++;
        if (count >= 9) count = 0;
        Serial.println(average / 10);

        //Wait to get correct sample rate of 500Hz
        while ((micros() - timedelay) < 2000) { //500hz
                delayMicroseconds(1);
        }
        timedelay = micros();
}
```

A trial was carried out using sample rates of 500Hz and 1000Hz. The results are shown in Figure 4.27 and Figure 4.28. The better result came from using the 500Hz sample rate, but the outcome signal still had noise in it. The higher sample rate had less noise but meant the peaks in the ECG wave are cancelled out as it is averaged with more data. This results in a signal with less noise but is less defined which is not useful in a microcontroller.

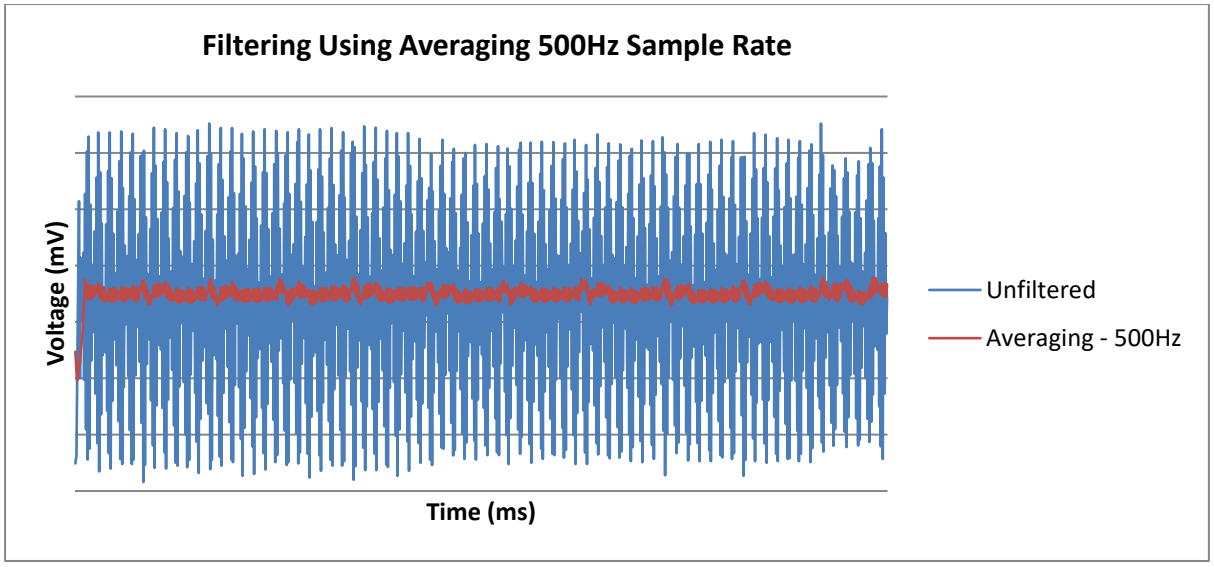



Figure 4.27 - Filtering Using Averaging 500Hz Sample Rate

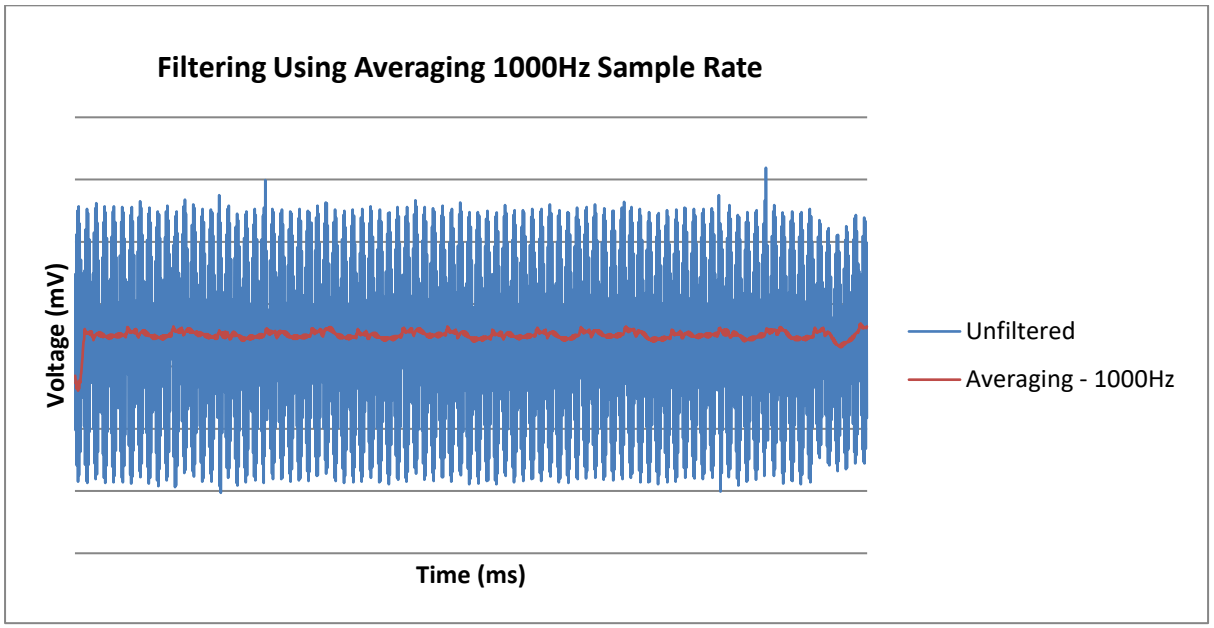



Figure 4.28 - Filtering Using Averaging 1000Hz Sample Rate

### 4.4.4 Comparison and Final Result

A comparison was made on the tested filtering methods and the graphs obtained by using different filters are shown in Figure 4.29. Looking at the graph, there is one filter that out performs the rest. This is the 4$^{th}$ order Butterworth low pass active filter. The signal strength obtained is the strongest and has less residual noises. As a result, this filter is selected to be used in the ECG device developed.

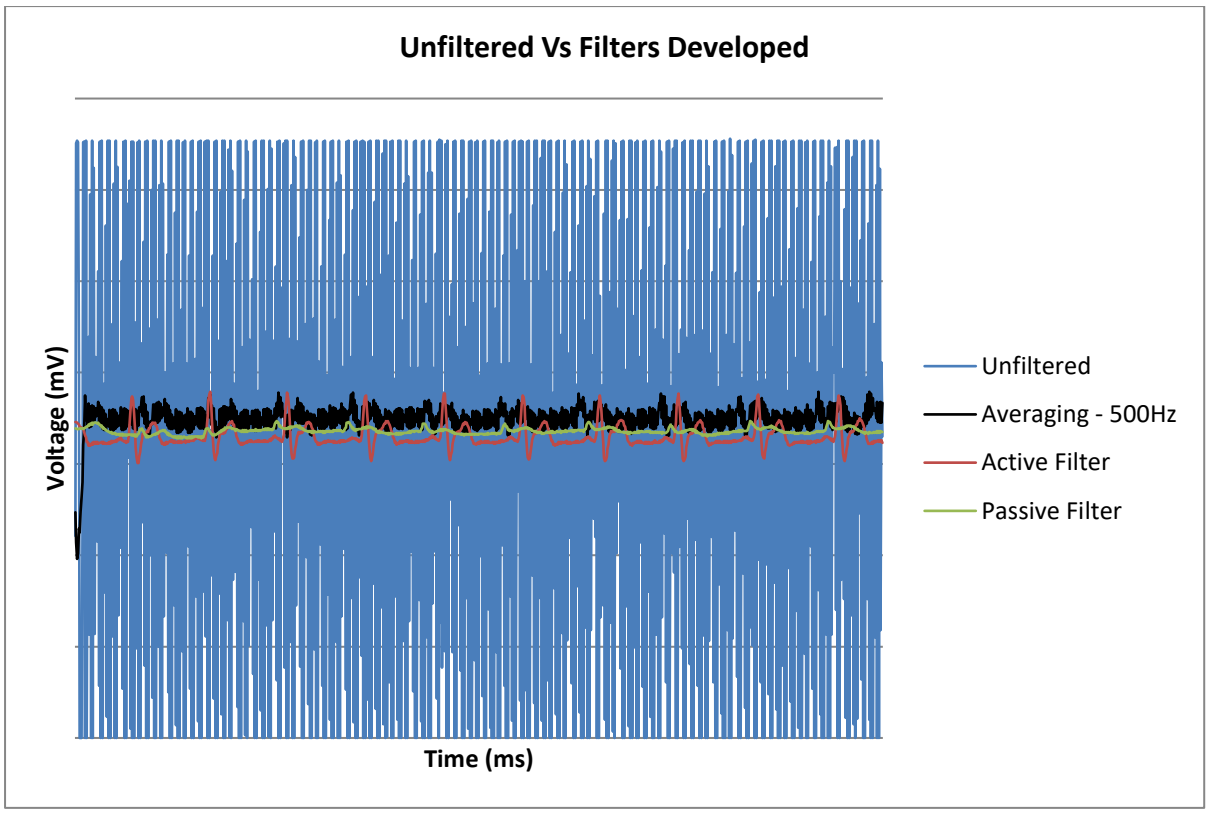



Figure 4.29 - Unfiltered Data vs Developed Filters

## 4.5 Conclusions

Figure 4.30 below compares the initial three-lead ECG signal to the developed two-lead ECG system. It can be seen that using the two-lead ECG introduces a large amount of noise, hence the filtering is required; however, the signal strength is similar to that of the three-lead ECG. Filtering of the two-lead ECG signal removes noise but at the cost of the signal strength, which is lost. In comparison, the developed two-lead ECG system can produce a signal that is smoother and cleaner but is weaker in strength compared to the initial three-lead signal.

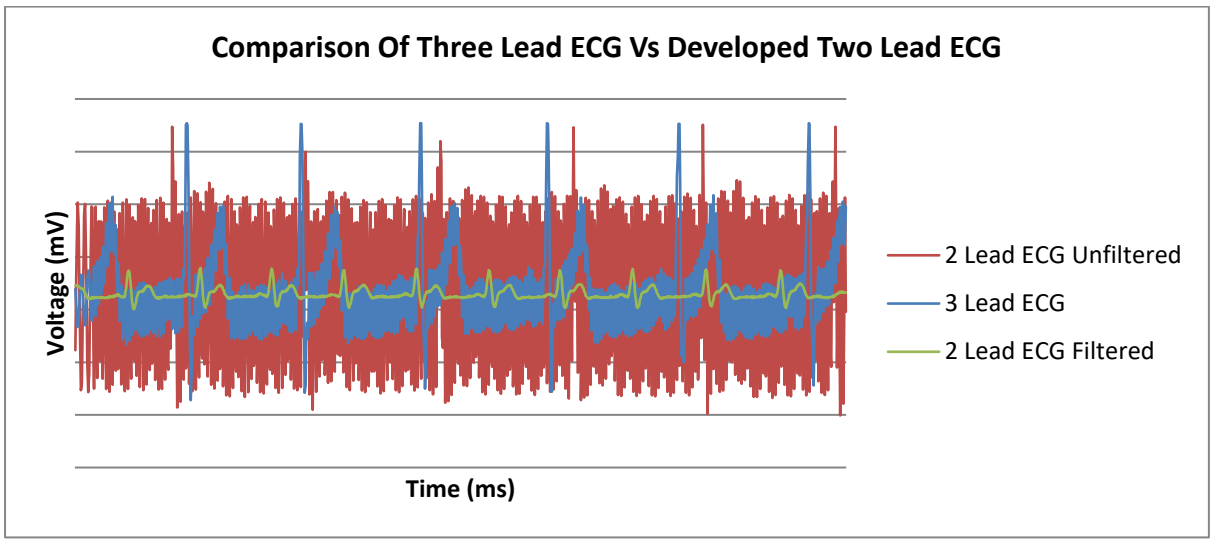



Figure 4.30 - Three-lead ECG Signal vs the Developed Two-lead ECG Signal

# 5. Dry Electrode Study

All the development work for this research and testing so far has been done using commercially available, sticky electrodes, which are typically used when measuring the patient's ECG signal in hospitals, healthcare organisations and research institutes. Sticky electrodes are used because they stick to the patient's body and do not need to be held in place. The conductive gel on the electrode improves the conduction of the electrical signals.

Sticky electrodes are designed to be used once. Typically, sticky electrodes generate better ECG signals than dry electrodes as they are less affected by moment between the skin and the electrodes [70]. Dry electrodes refer to electrodes which are dry because they do not need conductive gel and do not stick to the patient's skin.

An investigation was carried out to see if a set of dry electrodes could be used for the ECG device rather than sticky electrodes therefore to determine whether they could capture the ECG signals as strongly as the sticky electrodes. If dry electrodes are possible to develop this would allow the users of the device to place the electrodes and reposition them with ease. It will also keep the cost down as there is no need to replace the electrodes.

Factors that may influence the signal strength of an ECG wave include:

- Electrode size
- Electrode placement
- Number of electrodes
- Use of Conductive gel

The first question addressed was the size of the dry electrode. To determine the best size and the best position for the dry electrodes in a hand-held device, a study was set up and tests were conducted.

## 5.1 Objectives and Test Method

A study was established to investigate the influences of electrode size, placement and conductive fabric on ECG signal strength. The size of electrodes and their placement position defines the size of the developed hand-held device.

Metal is a good conductor of electrical signals. Aluminium was chosen as a material to create test electrodes from due to its availability, cost and machinability. Figure 5.1 below shows the electrode mount design. It has a grove to allow for easy mounting. It has a grub screw though the side to attach wire to the mount. The shaft down the centre is tapped with a thread to allow for quick and easy changing of electrode sizes.

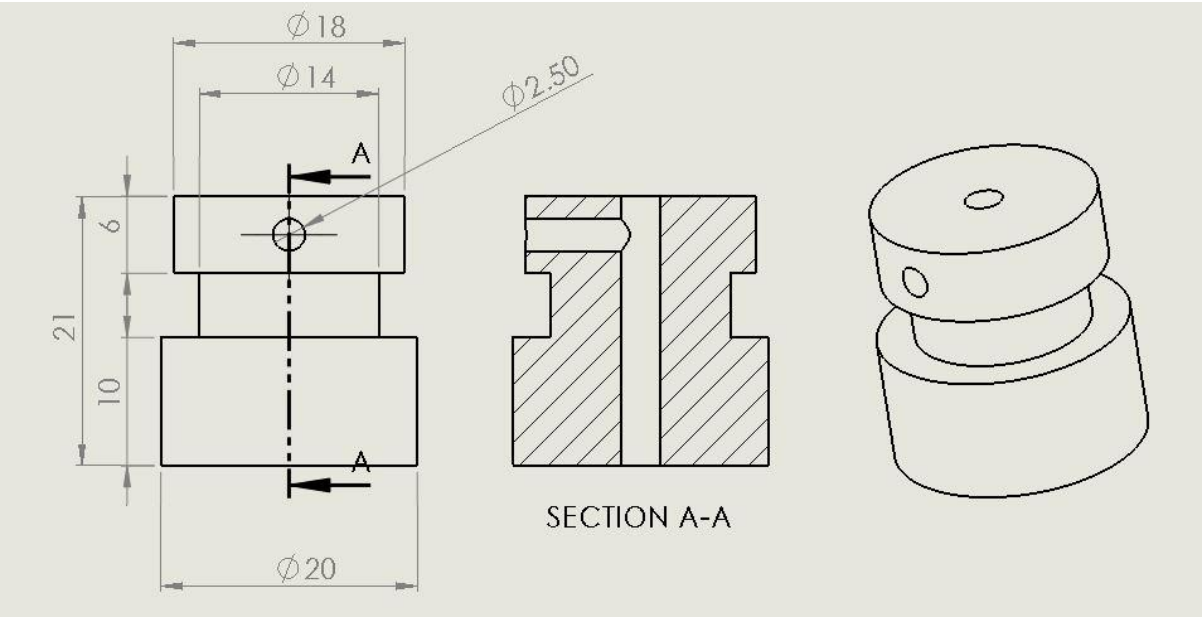



Figure 5.1 - Electrode Mount Design

The following parameters were used when conducting the study:

- Electrode size (25mm, 35mm and 45mm diameter aluminium)
- Electrode placement (10mm, 35mm and 60mm distance apart from the closest point)
- Conductive fabric (Fabric layer which contacts electricity, On or Off)

To measure which factor or combination gives the best outcome, the ECG signal is observed recording the average range between maximum and minimum values during each cycle (Signal amplitude). A Minitab Design of Experiment (DOE) project was established for logging and analysing the results. Each of the combinations were repeated three times to reduce variation in the data. The order was also randomised to reduce variation.

For the convenience of the test, a rig was developed, to give the desired distance between the electrodes, as well as making it easy to change the size of the electrodes and to add the conductive fabric - as shown in Figure 5.2. Also, a set of different size aluminium electrodes, for conducting the tests were made - as shown in Figure 5.3.

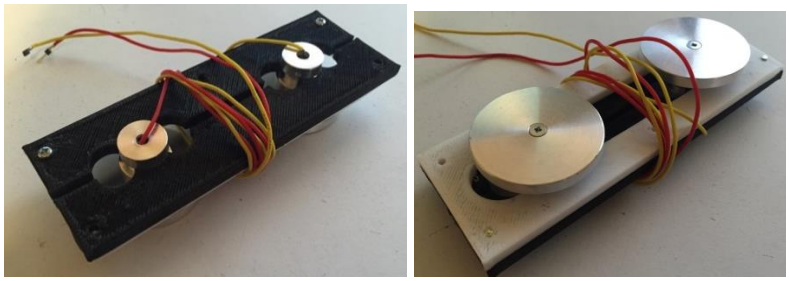


Figure 5.2 – Top and Bottom of DOE Test Rig

Figure 5.3 - Various Aluminium Electrode Sizes

## 5.2 DOE Setup

In Minitab, a 3-factor DOE was set up. Two of these factors (electrode size and electrode placement) were set up using three levels as shown in Table 5.1.

| Electrode Size | Electrode Placement | Conductive Fabric | Trial 1 (average amplitude) | Trial2 (average amplitude) | Trial 3 (average amplitude) |
|---|---|---|---|---|---|
| 25 | 10 | Off | | | |
| 25 | 10 | On | | | |
| 25 | 35 | Off | | | |
| 25 | 35 | On | | | |
| 25 | 60 | Off | | | |
| 25 | 60 | On | | | |
| 35 | 10 | Off | | | |
| 35 | 10 | On | | | |
| 35 | 35 | Off | | | |
| 35 | 35 | On | | | |
| 35 | 60 | Off | | | |
| 35 | 60 | On | | | |
| 45 | 10 | Off | | | |
| 45 | 10 | On | | | |
| 45 | 35 | Off | | | |
| 45 | 35 | On | | | |
| 45 | 60 | Off | | | |
| 45 | 60 | On | | | |

Table 5.1 - DOE Setup

## 5.3 DOE Results

During the running of the experiment, some observations were noted:

- There was a large variance in the results between the trials. Small movements in the position of where the electrodes were placed on the body had a huge influence on the amplitude of the signal. Small movements also affected the amount of noise within the signal.
- The larger distance between the electrodes, the design of the test rig and electrodes themselves made it difficult to get good skin contact due to the contour of human body.

The results of the experiment are shown in Table 5.2. Generally, the average amplitude increases as the diameter of the electrode get larger. An example of calculating the average amplitude of the ECG signal is shown in Figure 5.4 below. In this trial, the average amplitude was determined to be 55.

| Electrode Size | Electrode Placement | Conductive Fabric | Trail 1 (average amplitude mV) | Trail 2 (average amplitude mV) | Trail 3 (average amplitude mV) |
|---|---|---|---|---|---|
| 25 | 10 | Off | 20 | 20 | 20 |
| 25 | 10 | On | 20 | 20 | 15 |
| 25 | 35 | Off | 25 | 20 | 30 |
| 25 | 35 | On | 30 | 30 | 30 |
| 25 | 60 | Off | 30 | 25 | 25 |
| 25 | 60 | On | 20 | 25 | 30 |
| 35 | 10 | Off | 15 | 30 | 40 |
| 35 | 10 | On | 35 | 30 | 30 |
| 35 | 35 | Off | 50 | 30 | 70 |
| 35 | 35 | On | 50 | 50 | 45 |
| 35 | 60 | Off | 50 | 40 | 50 |
| 35 | 60 | On | 25 | 40 | 50 |
| 45 | 10 | Off | 55 | 30 | 75 |
| 45 | 10 | On | 40 | 30 | 40 |
| 45 | 35 | Off | 60 | 55 | 60 |
| 45 | 35 | On | 50 | 60 | 65 |
| 45 | 60 | Off | 65 | 55 | 50 |
| 45 | 60 | On | 25 | 60 | 50 |

Table 5.2 - DOE Experiment Results

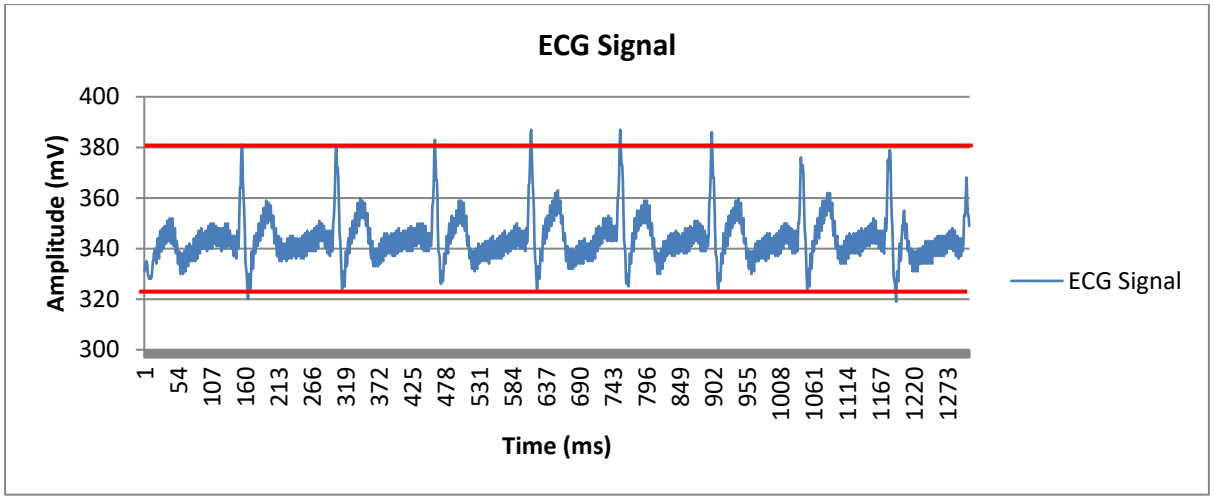



Figure 5.4 – DOE Amplitude Calculation Example

## 5.4 Minitab results

When analysing the DOE, Minitab automatically outputs data (analysis of variance) which indicates the importance of each factor, which is shown in Figure 5.5. To start with, all the factors and the interactions were included in the analysis. In this study, an alpha value of 5% was used, which means any factor with a P-Value less than 0.05 is significant.

```
Source                                                  P-Value
Model                                                     0.000
  Linear                                                  0.000
    Electrode Size                                        0.000
    Electrode Placement                                   0.001
    Conductive Fabric                                     0.185
  2-Way Interactions                                      0.638
    Electrode Size*Electrode Placement                    0.723
    Electrode Size*Conductive Fabric                      0.318
    Electrode Placement*Conductive Fabric                 0.446
  3-Way Interactions                                      0.742
    Electrode Size*Electrode Placement*Conductive Fabric  0.742
Error
Total

Model Summary

      S    R-sq  R-sq(adj)  R-sq(pred)
10.0692  72.16%     59.01%      37.35%
```

Figure 5.5 – DOE Minitab Analysis of Variance

The output data shows that electrode size and placement are the most important factors when it comes to determining the ECG signal strength. Just below the analysis of variance, in Figure 5.5, is the model summary. The R values indicate how well the model generated fits the data. In this case, the R-sq value is 72.16% suggesting that the model cannot explain the amount of variability. It was observed when running the experiment that very small changes in the placement of the electrodes on the body can create large variation in the signal strength from trial to trial. The R-sq(adj) factor considers the number of predictors used in the model. The resulting value was 59.01% which indicates that the model can be improved significantly by reducing the number of predictors in the model.

Once Minitab has completed the analysis, it outputs the residual plots shown in Figure 5.6. The residual plots show that the experiment was valid. Analysis:

- Residual vs Fitted value – In this plot, random pattern with a centre line around 0 suggests the data is valid.
- Histogram of residuals – In this graph a bell-shaped curve (normal distribution) suggests the data is valid.
- Residuals vs order – In this plot a random pattern is ideal.

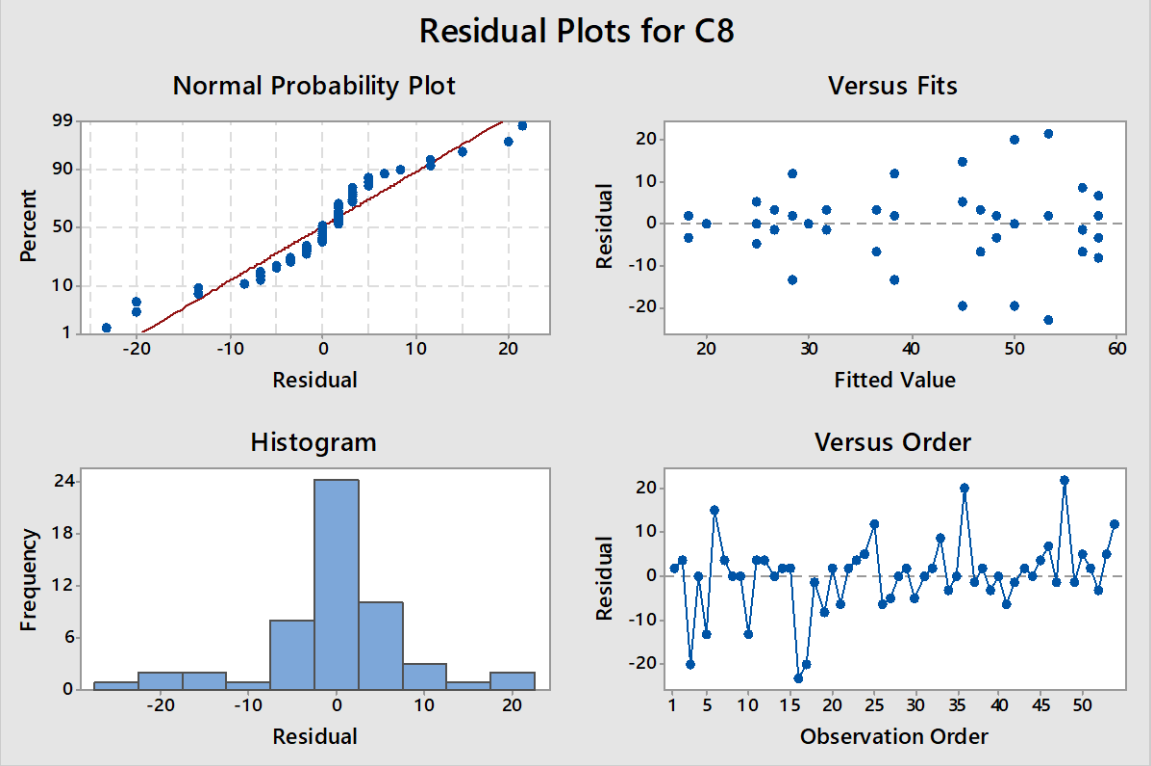



Figure 5.6 - DOE Residual Plots

The main effects plot is shown in Figure 5.7. The steeper the line is, the more significant that factor is. Increasing the electrode size from 25mm to 35mm is more significant in increasing the amplitude of the ECG signal strength than increasing the electrode size from 35mm to 45mm. The output of the analysis of variance, Figure 5.5, also confirms that electrode size and electrode placements are the most significant factors.

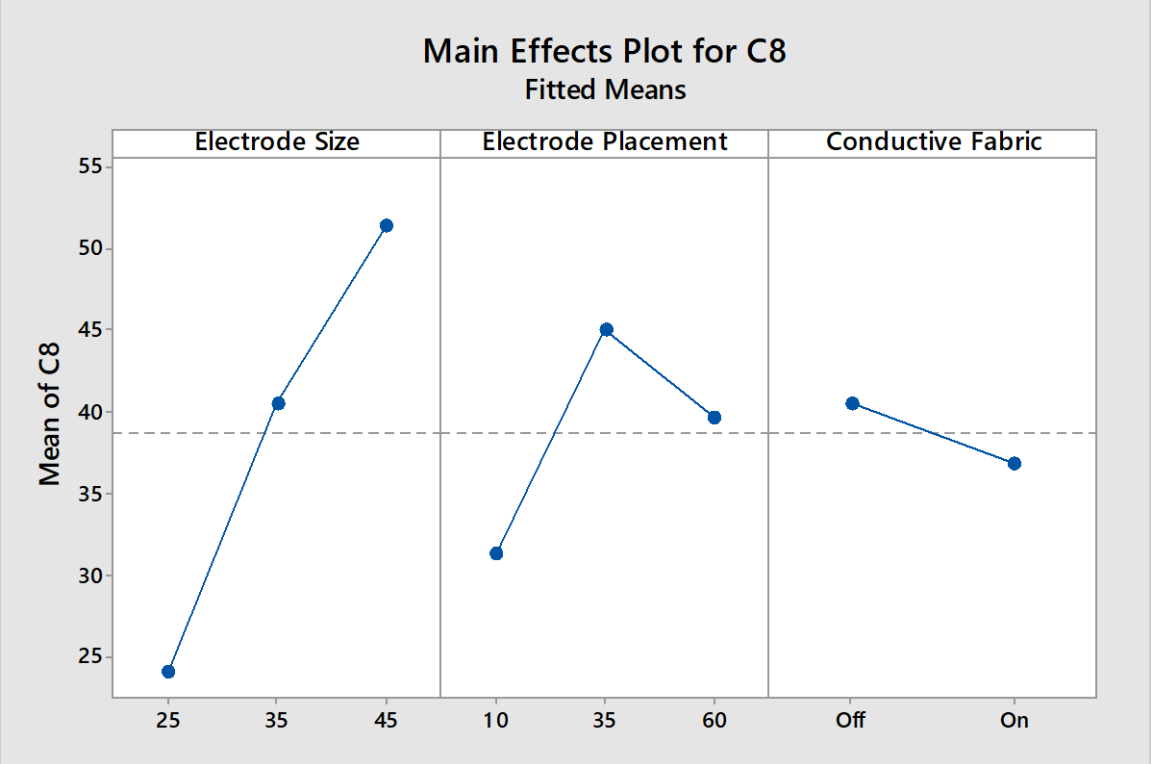



Figure 5.7 - DOE Main Effects Plot

The Interaction plot in Figure 5.8 shows if there is any important interaction between the factors. Factor interactions are important if they intercept another. The interaction shown in, Figure 5.8, shows no major interactions.

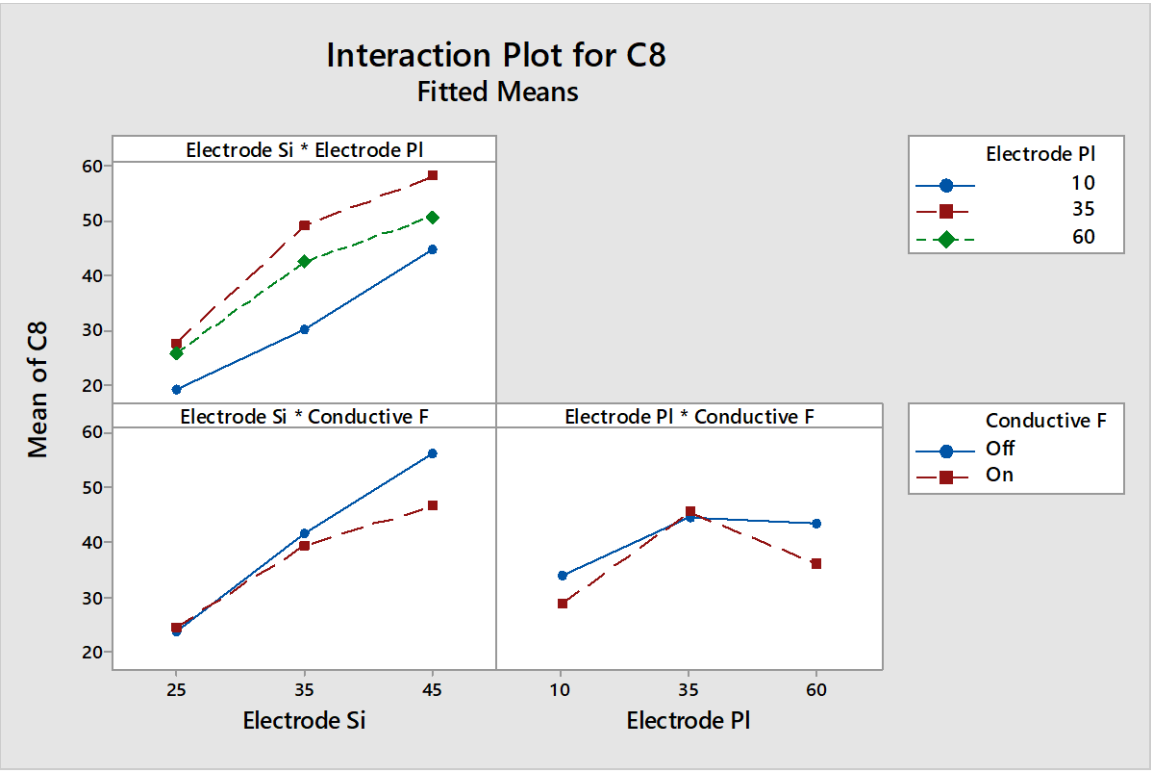



Figure 5.8 - DOE Interaction Plot

Minitab also can generate an optimised solution as shown in Figure 5.9. The optimised solution returns the values of each factor that gives the best result (highest amplitude ECG). These are:

- 45mm diameter electrode
- 35mm distance apart
- No conductive fabric

Using these values will on average, give the highest amplitude (strongest signal).

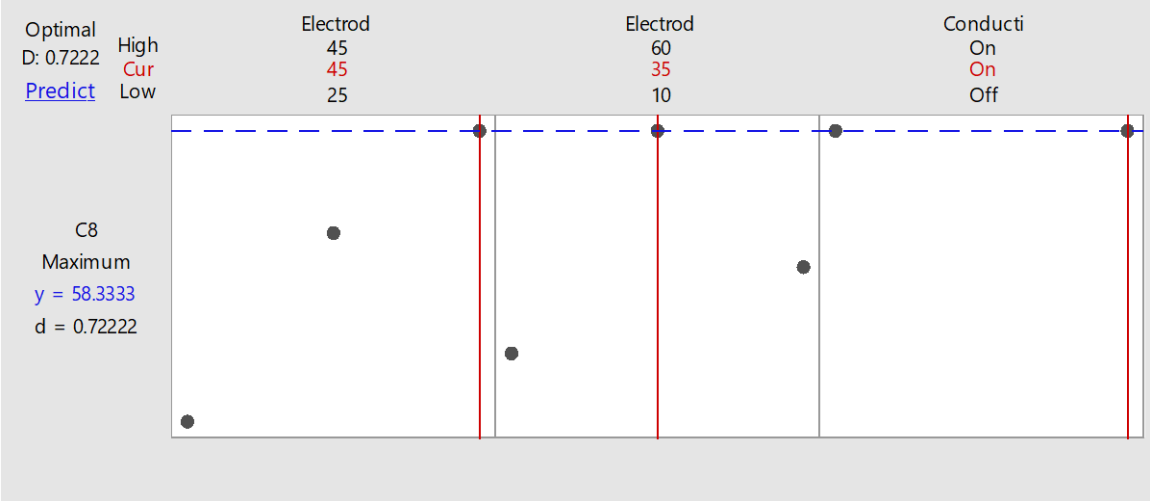



Figure 5.9 - DOE Optimised Solution

Based on the results generated by Minitab, factors which were not significant were removed, leaving:

- Electrode size
- Electrode position

The analysis was then re-run and returned the results in Figure 5.10. All the factors in the analysis are significant. The R-sq value (64.51%) has decreased from 72.16% suggesting the model does not explain the variance in the data as well as before. The R-sq(adj) has shifted from 59.01% to 61.62% which suggests, with the number of predictors used, it is better than the model with all the predictors used.

Analysis of Variance

| Source | DF | Adj SS | Adj MS | F-Value | P-Value |
|---|---|---|---|---|---|
| Model | 4 | 8457 | 2114.35 | 22.27 | 0.000 |
| Linear | 4 | 8457 | 2114.35 | 22.27 | 0.000 |
| Electrode Size | 2 | 6762 | 3381.02 | 35.61 | 0.000 |
| Electrode Placement | 2 | 1695 | 847.69 | 8.93 | 0.000 |
| Error | 49 | 4652 | 94.94 | | |
| Lack-of-Fit | 13 | 1002 | 77.07 | 0.76 | 0.694 |
| Pure Error | 36 | 3650 | 101.39 | | |
| Total | 53 | 13109 | | | |

Model Summary

| S | R-sq | R-sq(adj) | R-sq(pred) |
|---|---|---|---|
| 9.74350 | 64.51% | 61.62% | 56.90% |

Figure 5.10 - DOE Reduced Model Analysis

## 5.5 Verification

The DOE experiment conducted above concluded that the highest signal strength was obtained using a 45mm diameter electrode placed 35mm apart. This was then used and compared against commercial sticky electrodes. For this justification, it was critical that the electrodes were placed in the same location. To ensure this, the position was marked on the chest.

Figure 5.11 and Figure 5.12 show the comparison between aluminium electrodes and the sticky electrodes using the two-lead ECG circuit and the 4$^{th}$ order low pass Butterworth filter. In sample 1 the aluminium electrodes were better than the commercial sticky electrodes. In the second sample the commercial electrodes were slightly better. This result is positive and shows that dry electrodes can be manufactured to produce a similar result as commercial sticky electrodes.

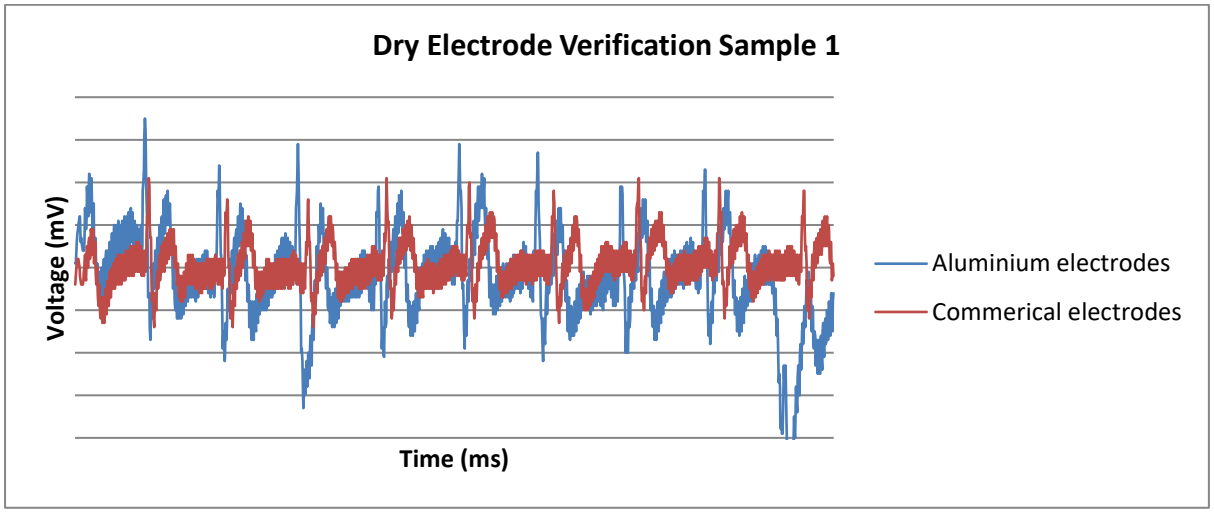



Figure 5.11 - DOE Verification Sample 1

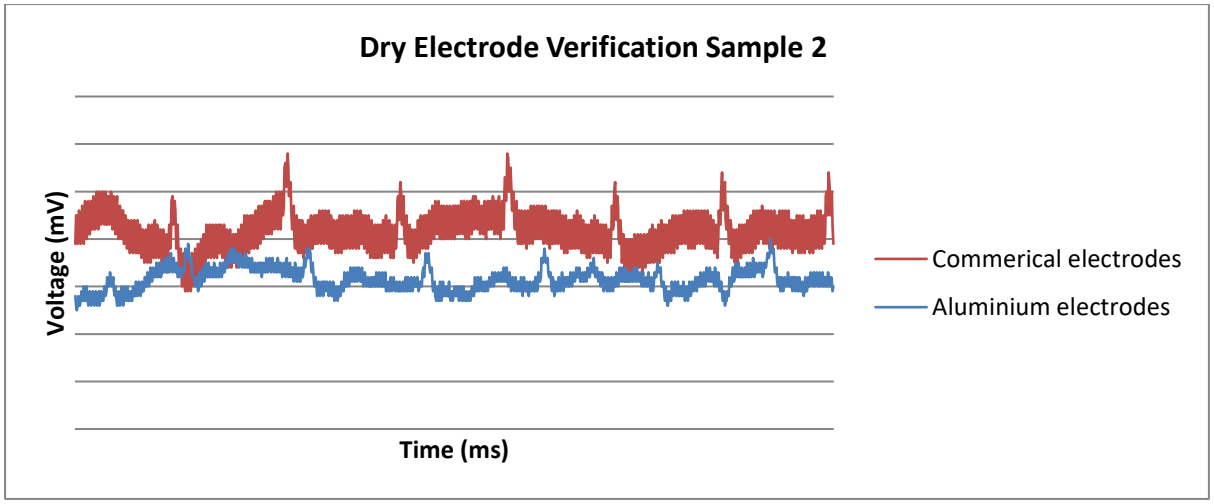



Figure 5.12 - DOE Verification Sample 2

## 5.6 Conclusion

- The aluminium electrodes are on par with commercial sticky electrodes in terms of the quality of the signal collected.
- This experiment concluded that using 45mm diameter electrodes, spaced 35mm apart generated the highest amplitude ECG wave. Conductive fabric had little effect on the outcome.
- The large flat aluminium electrodes and the depth from the chassis limits skin contact. For a valid result the electrodes require good skin contact and the contour of the body makes this harder with the larger electrodes at certain location. Deeper soft/flexible electrodes may work better.
- There was a large amount of variance from test to test. Small movements in position of the electrode caused a huge difference in signal strength.

## 5.6 Soft electrodes

The DOE study concluded that large flat aluminium electrodes are difficult to place on the chest due to the contour of the body. To get a strong ECG signal, a good skin contact is required. To improve the functionality of the aluminium electrodes and to get a stronger ECG signal, soft electrodes were considered maintaining, two key parameters;

- The same cross-sectional area of the 45mm diameter aluminium electrodes
- The same distance apart (35mm)

The concept being that these will mould to the user's body shape ensuring good skin contact. The soft electrodes where made from conductive fabric with a foam insert as shown in Figure 5.13.

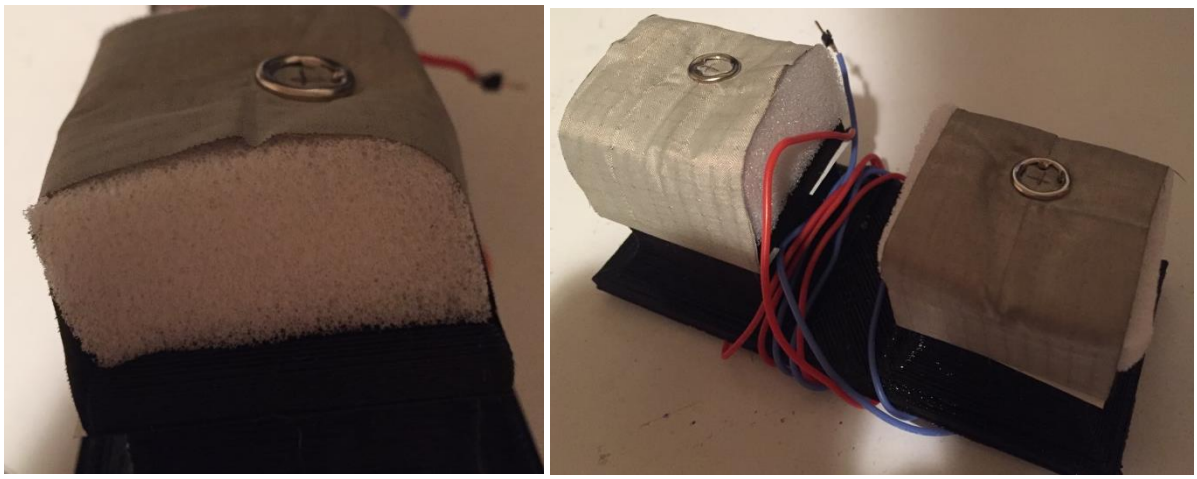


Figure 5.13 - Soft Electrode Development

These electrodes were tested and compared to aluminium electrodes. The results are presented in Figure 5.14 and Figure 5.15. When conducting the tests, the placement of the electrodes on the chest was marked to ensure that they were placed at the same position to eliminate the variability caused by placement.

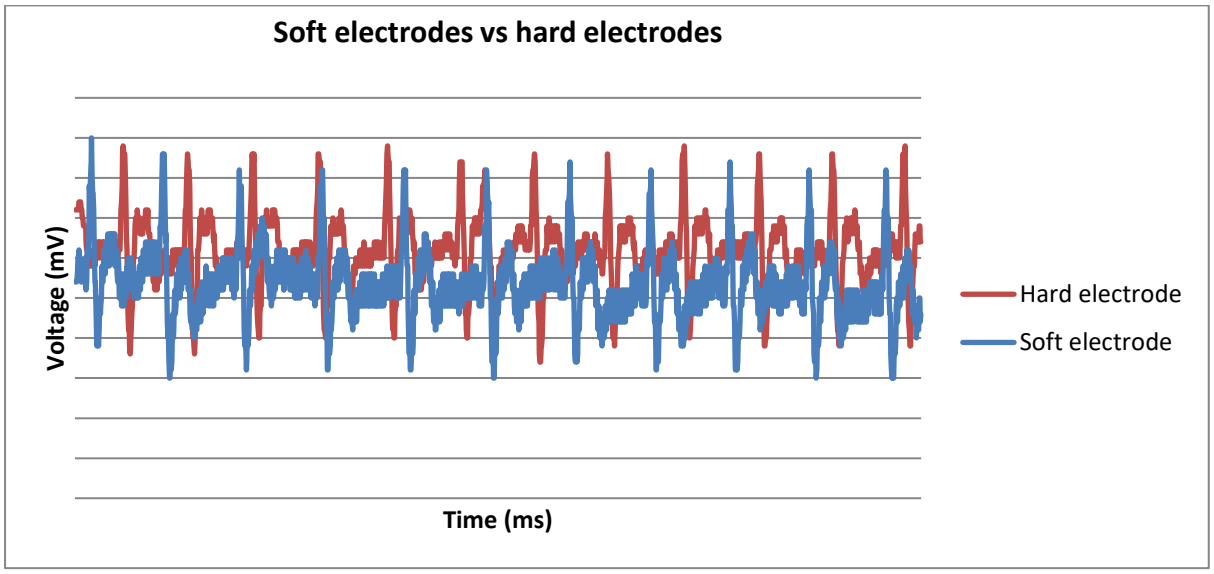



Figure 5.14 - Soft vs Hard Electrodes Sample 1

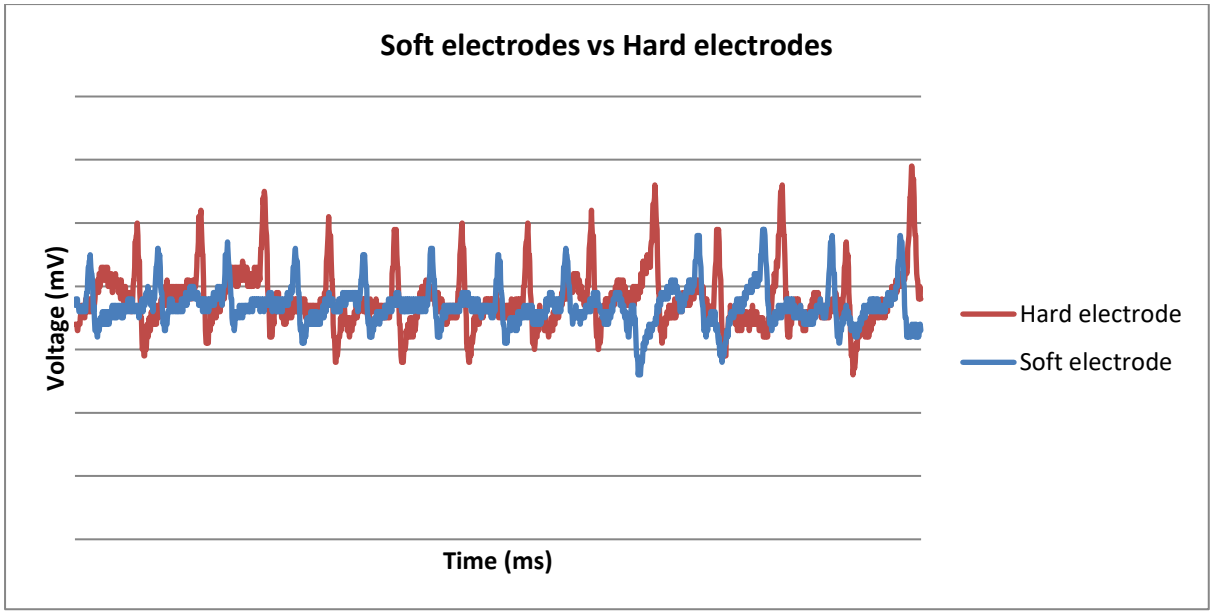



Figure 5.15 - Soft vs Hard Electrodes, Sample 2

### 5.6.1 Comparison and Results

- In Sample 1, the soft electrodes collected an ECG signal slightly stronger than the aluminium electrodes. In the second sample the soft electrodes generated a wave that was weaker. Overall the soft electrodes are just as effective as the hard electrodes.
- The soft electrodes were easier to use and more comfortable on the skin.

## 5.7 Conclusions

The developed soft electrodes perform on par with the commercially brought sticky electrodes when used in the developed two electrode system and placed in the same location on the body. Comparing the developed two-lead ECG system using soft electrodes to the original three-lead ECG signal, the soft electrodes acquire a signal of lower strength and less noise. This can be seen in Figure 5.16. It was noted that the position of the electrodes greatly influences the signal strength. The position of the two-lead electrodes in this case was different to Figure 4.30 where the three-lead ECG was compared to the developed two-lead ECG system.

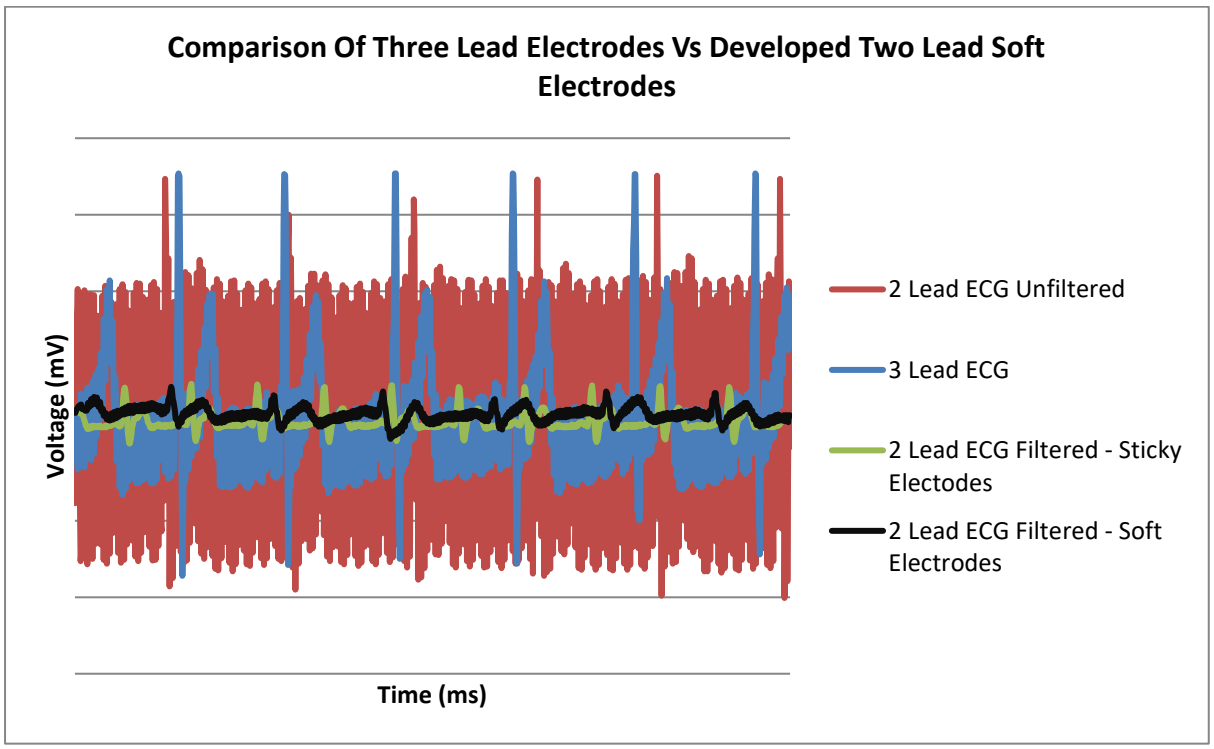



Figure 5.16 - Comparison of Three-lead Electrodes vs Developed Two-lead Soft Electrodes

# 6. Data Collection Methodologies

## 6.1 ECG Data Verification

The ECG wave is created from small electrical signals that control the muscles in the heart. Electrical signals are also used to control other muscles in the body. When using a microcontroller for ECG diagnostics it is very difficult to determine if the signal detected is an ECG signal or noise due to other muscle control. When the wave is shown in a graph form, it is very clear for a human to verify if the data is ECG. However, implementing a code in a microcontroller to identify can be challenging due to a number of factors:

- Frequency of the wave changes (heart rate changes).
- Signal strength varies from person to person.
- The signal itself varies depending on how the electrodes are positioned on the patient.
- Noise.

This is where an accelerometer can become useful. An accelerometer measures the acceleration (acc) typically along three axes; x, y and z. The basic idea proposed was:

- If the accelerometer data was stable, then the signal acquired should be an ECG wave.
- If the accelerometer data jumps or moves frequently then the signal acquired will contain noises due to other muscles moving.

Figure 69 below shows accelerometer data for a patient moving vs a still patient and the resulting ECG signal. When the patient is moving the ECG signal is no longer valid. With the aid of an accelerometer, detecting whether the patient is stationary is possible. Initially, it was thought the accelerometer could pick up the small movements in the chest due to the heart beating which would match up with peaks in the R wave. Testing showed that this was not always the case and another, more robust method was needed.

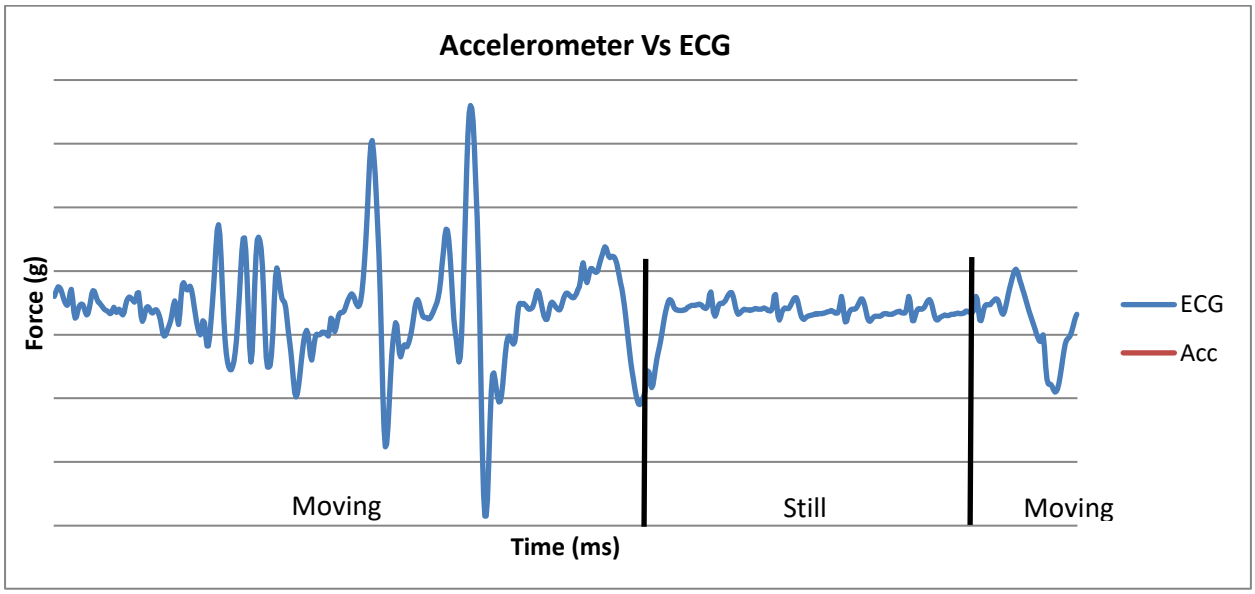



Figure 6.1 - Accelerometer Data vs ECG

### 6.1.1 ECG Data Verification Version 1

Using a standalone accelerometer, the graph of the raw accelerometer data in the z axis (this is in the forward direction if the patient is standing up) is shown in Figure 6.2. From this graph, it is easy to determine the difference when the accelerometer is stationary compared to moving.

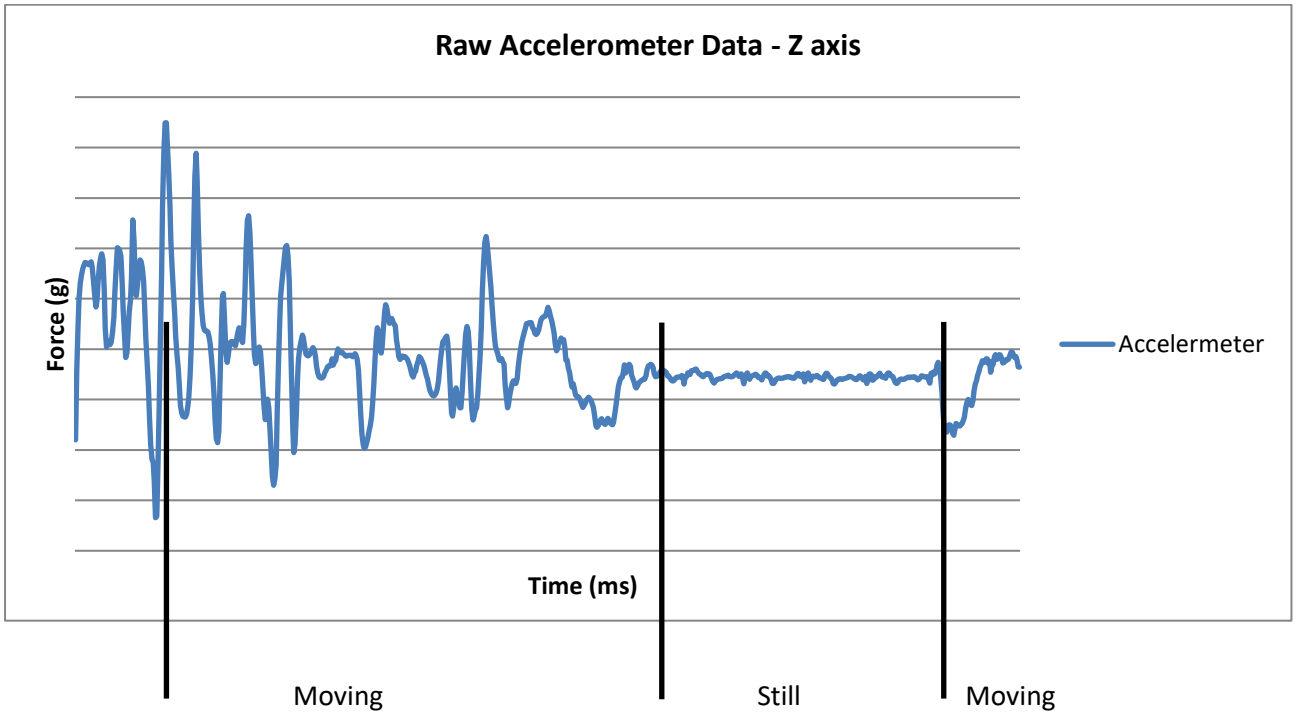



Figure 6.2 - ECG Data Verification Version 1 - Raw Accelerometer Data

**Proposed Method**

Monitor the accelerometer data in the z axis and see if the values are stable. To achieve this, values are initially stored for maximum and minimum. The initial values are then compared to the incoming values to check if a new maximum or minimum is observed, if so then they are saved as the new maximum or minimum value. A check is then done to see if the difference between the maximum and minimum is larger than a pre-defined range which would suggest:

- The patient is making too much movement.
- The ECG is invalid because the signal is more likely to be muscle movement.

In such a case where the ECG becomes invalid, the max and min values are reset. Once the signal becomes stable the ECG signal becomes valid. A test was run on this proposed method and Figure 6.3 shows the outcome of the testing.

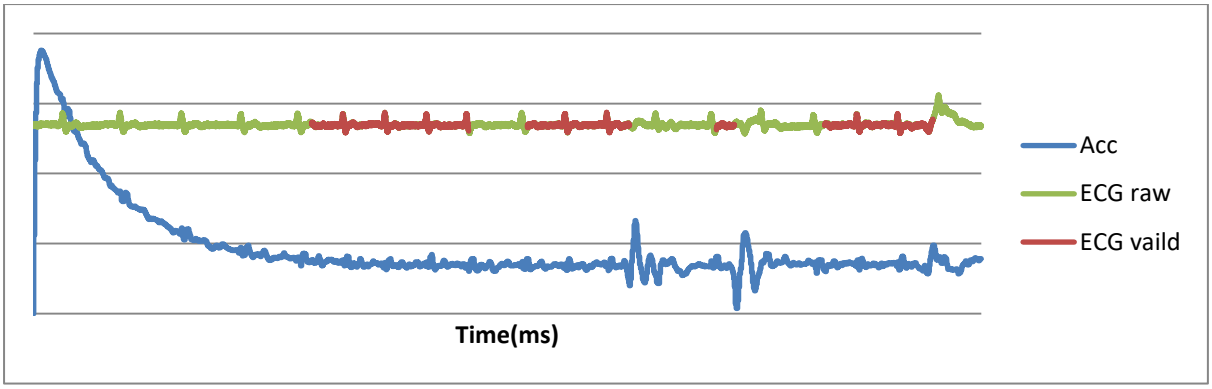



Figure 6.3 - ECG Data Verification Version 1 - Results

When the acceleration data becomes stable the ECG signal becomes valid. One major issue with this method was the ECG signal becomes invalid even though the accelerometer data is stable. This is due to resetting the maximum and minimum values. Also, the range in which is used to determine if the ECG signal is invalid is predefined. This would be good if the system could initially learn what this range should be from the patient.

### 6.1.2 ECG Data Verification Version 2

Using an Arduino 101 with the built-in accelerometer, the accelerometer data is shown in Figure 6.4.

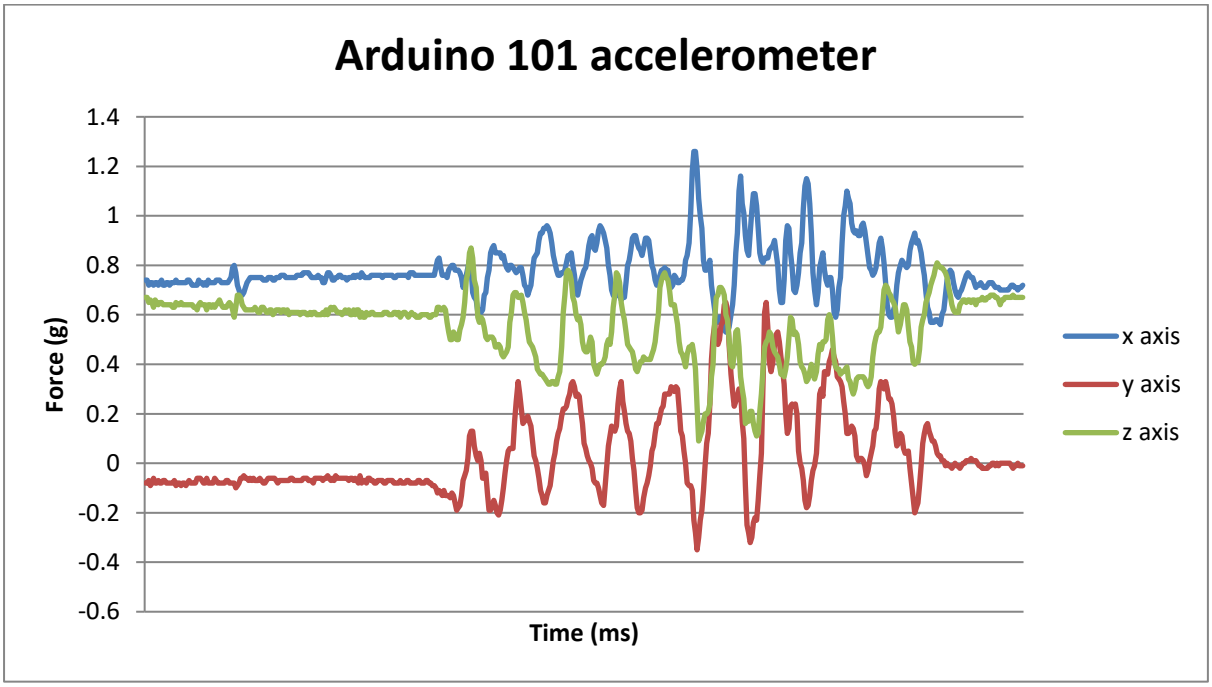



Figure 6.4 - ECG Data Verification Version 2 - Raw Acceleration from Arduino 101

**Proposed Method 1:**

Running averages can be used to smooth the raw data by reducing the effects of spikes, provided the sample size is large enough. With that in mind, a smaller sample size will be more susceptible to spikes in the raw data. Is it then possible to say if the larger running average equals the smaller running average then the accelerometer data is stable?

Firstly, the raw accelerometer data is averaged, using a running average of 10 samples. This is to help smooth out raw data and remove possible noise in the signal.

Then, two running averages will be used to determine if the ECG is valid:

- Running average of 50 samples long using the running average of the raw accelerometer data. Being smaller in size to the second running average means it is more affected by spikes in the raw acc data and changes quickly.
- Running average of 100 samples long using the running average of the raw accelerometer data. Being a larger sample size means that it is less affected by the raw acc data and changes slower.

Then compare the 50 samples moving average with the 100 samples moving average. If the two averages are close to one and another then the data is stable. Testing this method is shown in Figure 6.5.

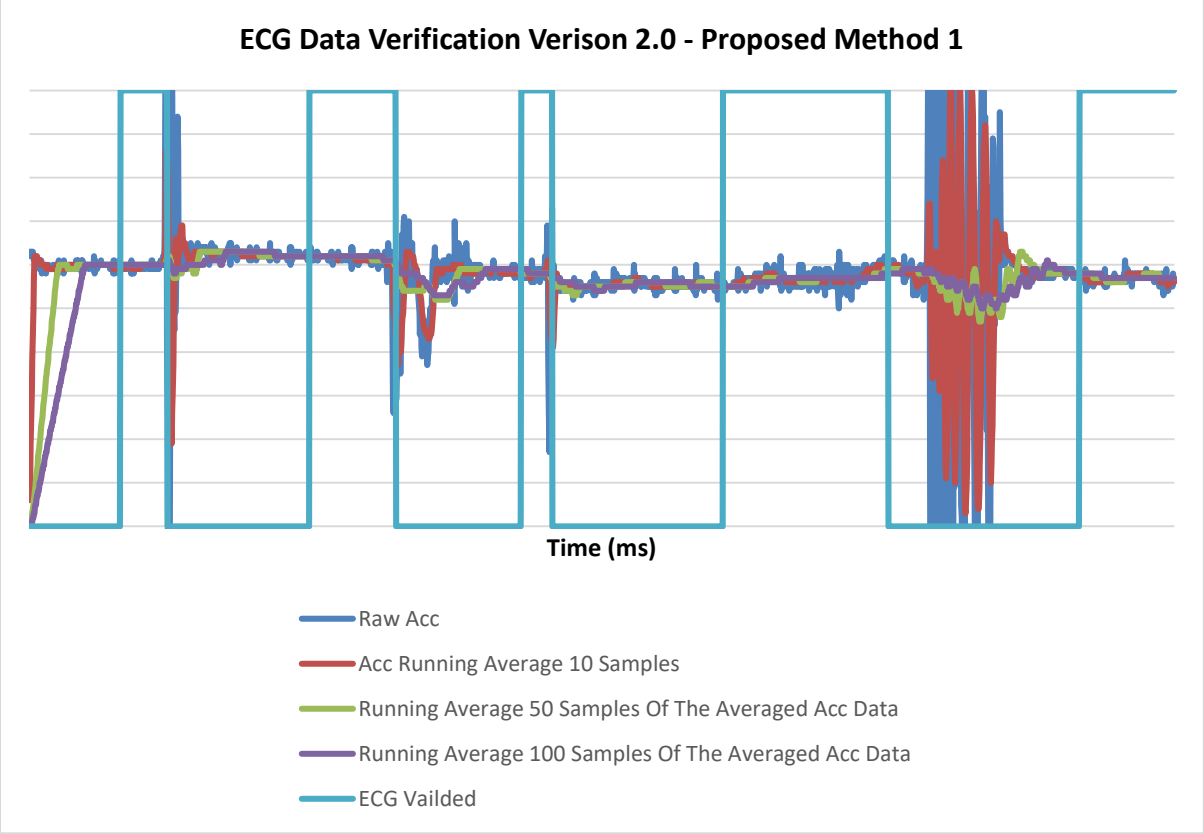



Figure 6.5 - ECG Data Verification Version 2 – Proposed Method 1 – Results

Looking closely at the graph above in some cases the system is slow to respond to sharp changes in the accelerometer data. This is due to the effects of the running average. To overcome this, the use of a threshold is implemented to speed up the effects of sharp spikes (Proposed Method 2).

**Proposed Method 2**

Introduce a maximum/minimum value for the last 100 sample size. At the start of the 100-sample running average the initial values are recorded as the min and max. The next 100 sample points are then compared to see if a new maximum or minimum value is found and if so, the maximum or minimum are updated. Once the 100 sample points have been read, the max/min are reset to the first value recorded in the sample and then repeats the process.

The maximum and minimum values are then used to define threshold values in calculating the next running average 50 sample sizes, which is shown below.

```
//Defining Thresholds based of difference of the last max and last min to the average
float mid=acc2/sample2;
float UpperThreshold= mid +(LastMax-mid)*3;
float LowerThreshold= mid -(mid-LastMin)*3;
```

The difference between the maximum/minimum and the running average is then multiplied by three to make it less susceptible to small changes in the accelerometer data but still detect sudden jumps. The value three came about from testing and optimising the code. The values calculated above then become the thresholds. Then the current accelerometer data is compared to the. If it's above the upper threshold value, then multiply the value by 500 and vice versa. It is multiplied by 500 to make the effects of this immediate instead of being averaged by 50 other data points. This makes it quick to respond to rapid changes. The code is shown below.

```
 if UpperThreshold <az {
 acc=acc+az*500;
 acc=acc-Array[count];
 Array[count]=az*500;
 }
 else if LowerThreshold >az {
 acc=acc+az*500;
 acc=acc-Array[count];
 Array[count]=az*500;
 }
else{
 acc=acc+globalacc/globalsample;
 acc=acc-Array[count];
 Array[count]=globalacc/globalsample;
}
```

After the threshold code is added on to the initial ECG data verification version 2, the code works well and makes the system respond quicker to rapid changes in the accelerometer data. Figure 6.6 is the result of ECG data verification version 2 using the proposed method 2.

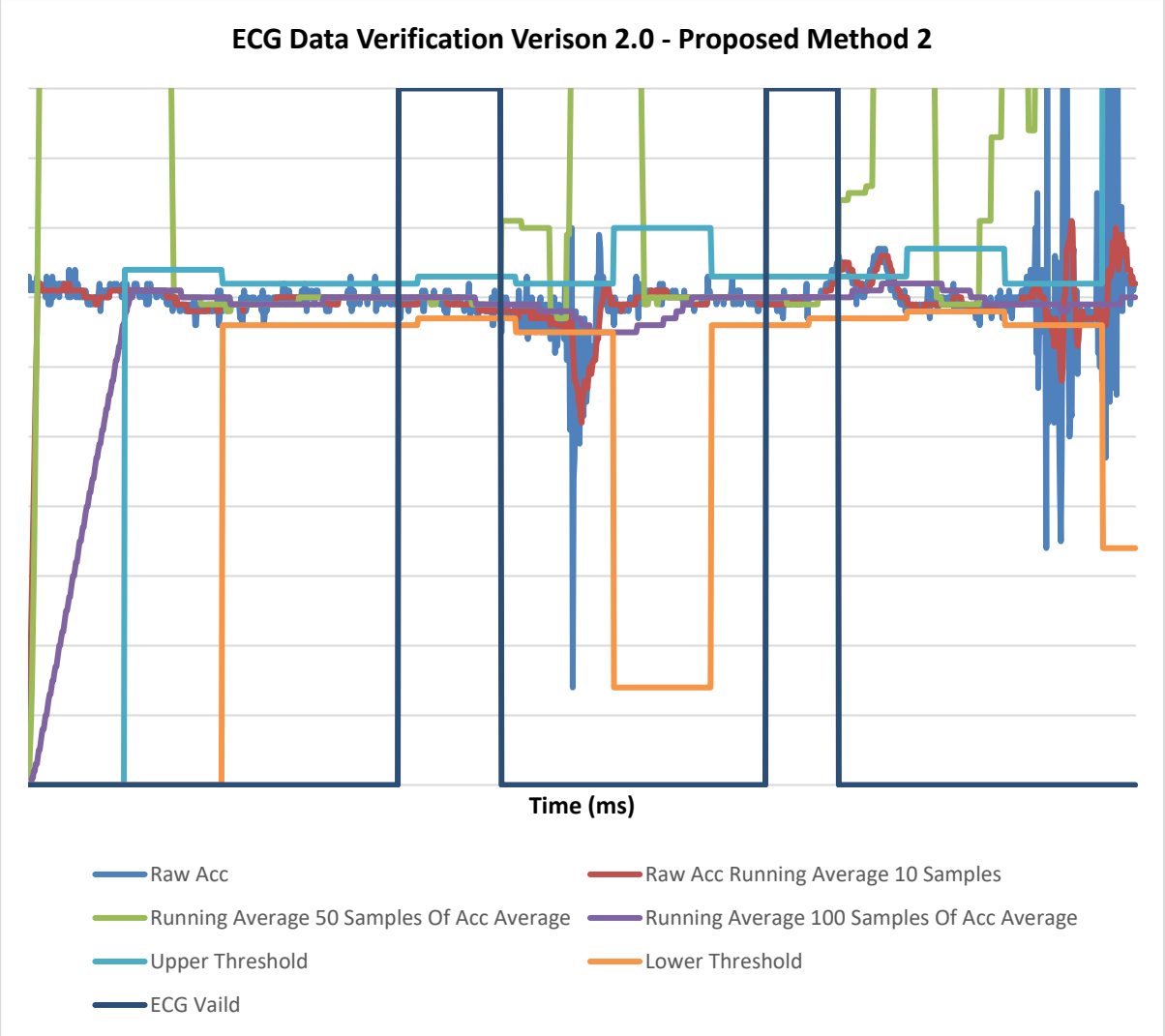



Figure 6.6 - ECG Data Verification Version 2 – Proposed Method 2 - Results

The final code developed for ECG data verification version 2, proposed method 2 was tested on two samples as shown in Figure 6.7 and Figure 6.8. It responds as it should. The program is used to determine whether the data detected from the electrodes is a valid ECG signal or noise generated from the movement of other muscles.

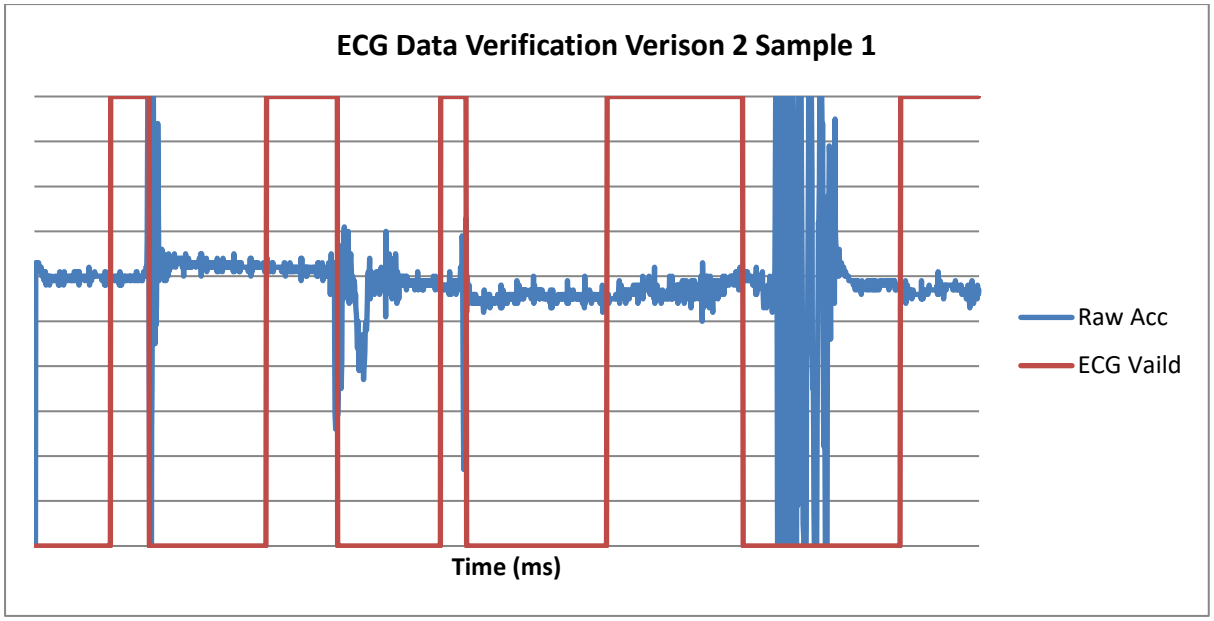



Figure 6.7 - The Outcome of Sample 1 Using ECG Data Verification Version 2

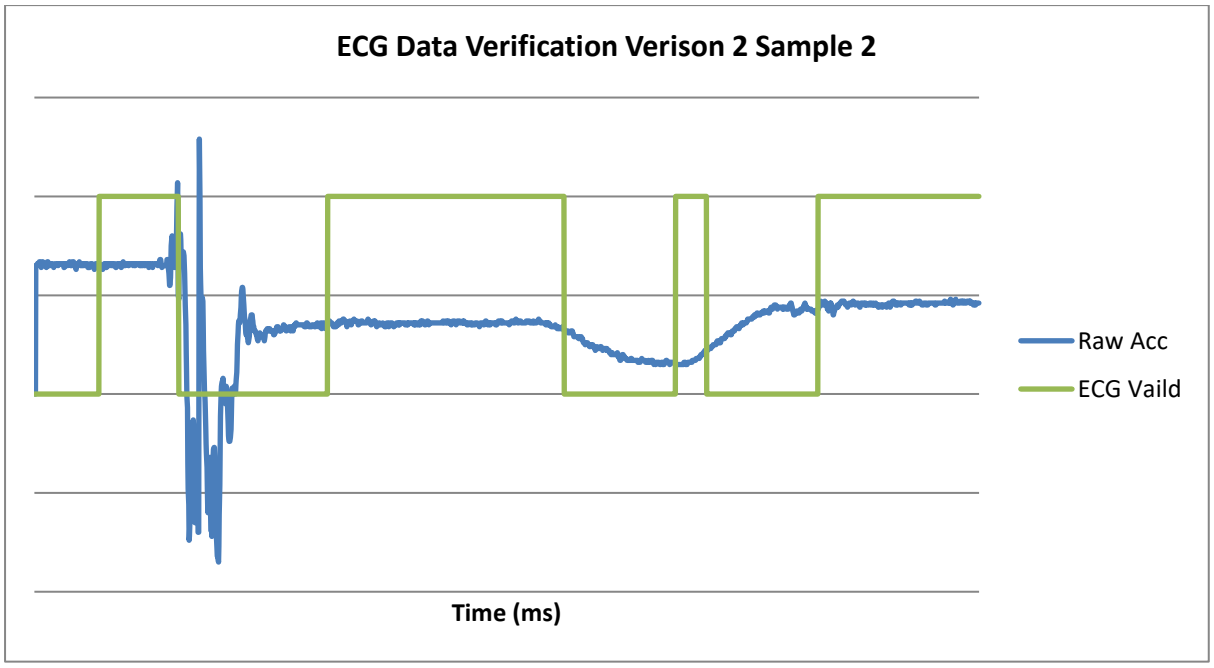



Figure 6.8 - The Outcome of Sample 2 Using ECG Data Verification Version 2

## 6.2 ECG Pulse Detection

This section documents the developed methodologies for collecting the patient's heart rate, based on the valid ECG wave discussed in the above section, 6.1 ECG Data Verification. The major issue with detecting the heart rate from an ECG wave is that the wave itself varies from person to person and is even affected by how the electrodes are positioned on the patient. Figure 6.9 shows an example of how the signal can vary. Code used to detect the heart rate needs to be robust and needs to work on all different types of ECG signals.

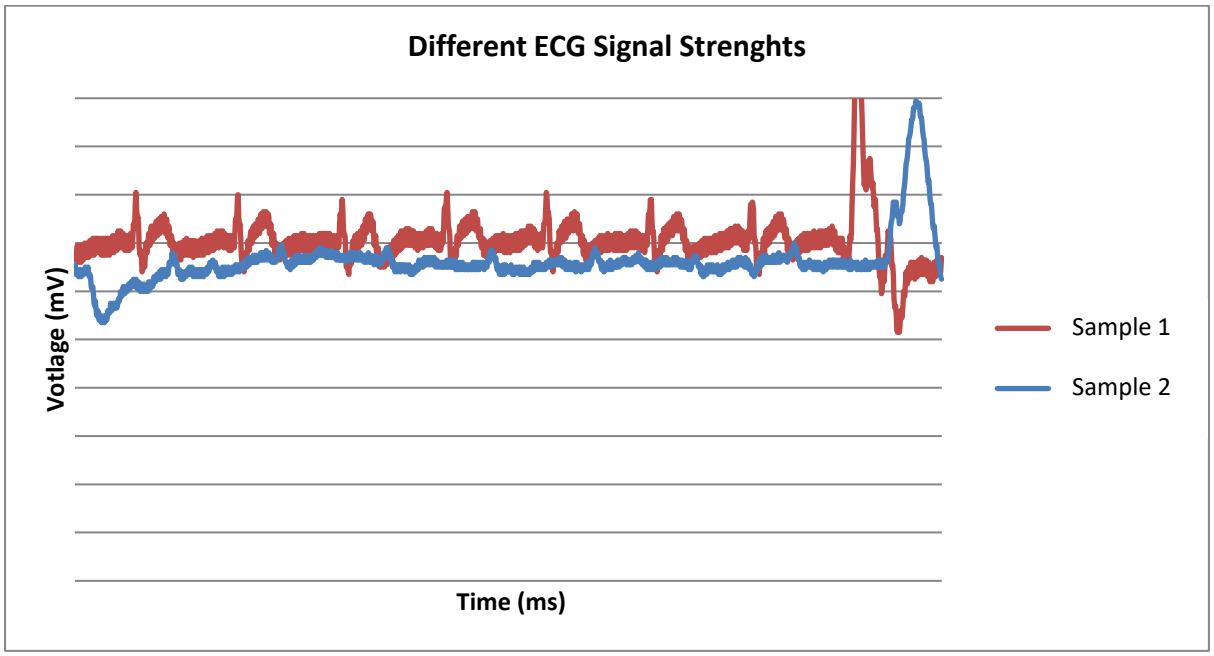



Figure 6.9 - ECG Pulse Detection - Different Signal Strengths

### 6.2.1 ECG Pulse Detection Calibration

To allow for different ECG signal strengths, values cannot be hard coded, therefore a calibration section is required to determine expected values.

**Proposed Idea**

To start with, listen to the ECG signal for five seconds. Five seconds was selected as a pulse every five seconds results in a heart rate of 12 beats a minute. This is well below the lowest recorded human heart rate of 27 BPM [71]. In this time frame, at least one pulse should have occurred. Over the five seconds, record the maximum and minimum values which can then be used for selecting a threshold for the detection of the R wave. After running the calibration method, the recorded maximum and minimum values are saved. Figure 6.10 shows the results of the ECG pulse detection calibration.

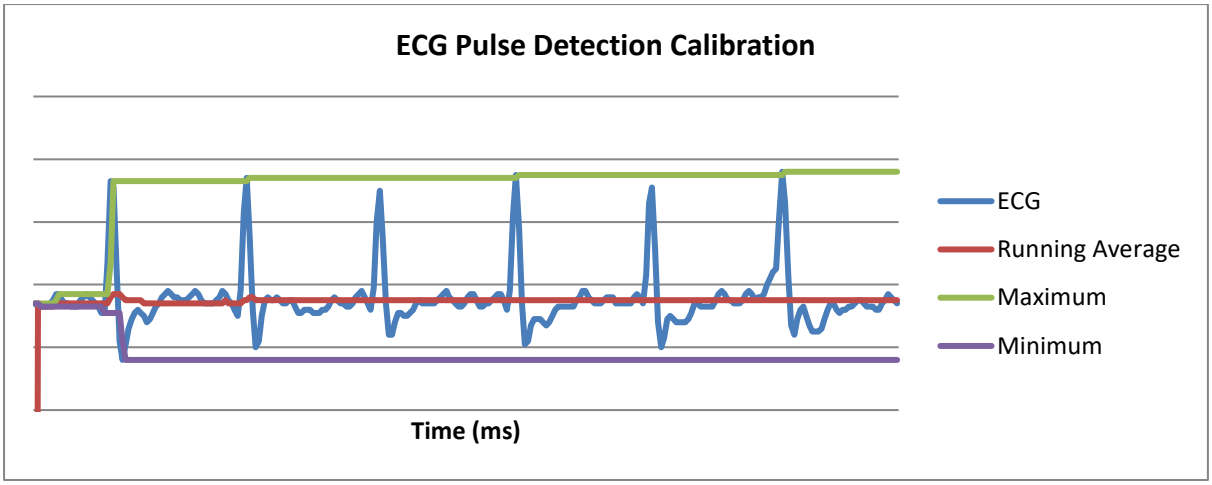



Figure 6.10 - ECG Pulse Detection Calibration

### 6.2.2 ECG Pulse Detection Version 1

The calibration method shown above, 6.2.1 ECG Pulse Detection Calibration, determines the maximum and minimum values for the ECG wave in the five second period it runs. The maximum and minimum values are then used to create upper and lower thresholds for detecting when a pulse occurs. A pulse occurs when the ECG value gets above the upper threshold and then drops below the lower threshold within a pre-defined time.

The thresholds are calculated by first finding the difference between the maximum/minimum and the running average. These differences are then multiplied by 0.7 and then added/subtracted from the running average to give upper and lower thresholds. The value of 0.7 was chosen through experimentation where the next R wave peak might not have the same amplitude as the previous. To prevent the system from missing the R wave peak, the thresholds are multiplied by 0.7 moving the thresholds closer to the running average.

Once a new pulse is detected, the maximum and minimum values over the last pulse is saved and reset. These values are then used for calculating the thresholds to find the next pulse. Effectively the next pulse (R wave) is detected by using threshold values calculated over the last pulse. If a pulse is not detected within five seconds the calibration code is then called again to determine new thresholds. Figure 6.11 shows the results of using the proposed method to detect ECG Pulse.

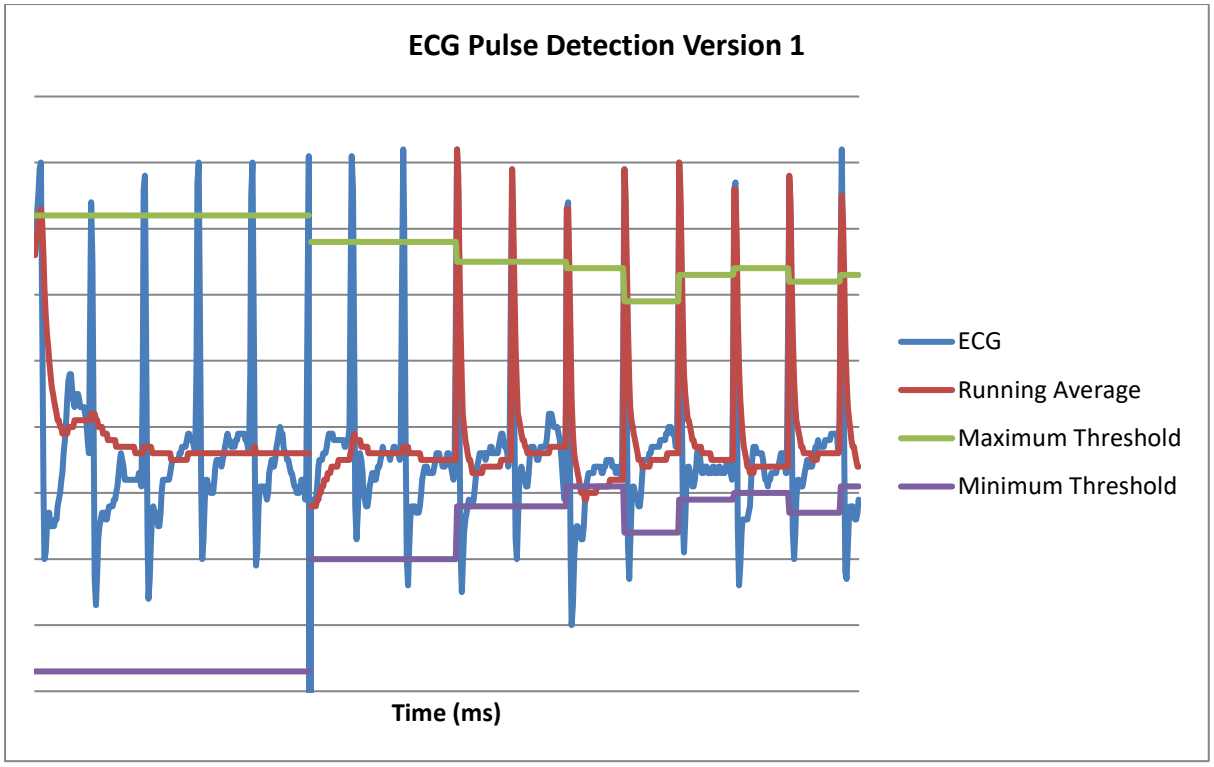



Figure 6.11 – The Outcome of Using ECG Pulse Detection Version 1

In testing the methodology, it became clear that there were issues. The ECG signal at times struggled to get below the lower threshold. The upper threshold was more consistent. It can be seen from the plot that it takes a while to detect the pulse but once it detects it, it can then follow it.

### 6.2.3 ECG Pulse Detection Version 2

Version two uses the same calibration method, but it only looks for the maximum value. After an R wave is detected the signal drops back below the running average. Instead of waiting for the signal to get above the upper threshold and then drop down below a lower threshold, the proposed method waits for the signal to get above the upper threshold and drop below the running average. This method worked well, and the results are shown in Figure 6.12.

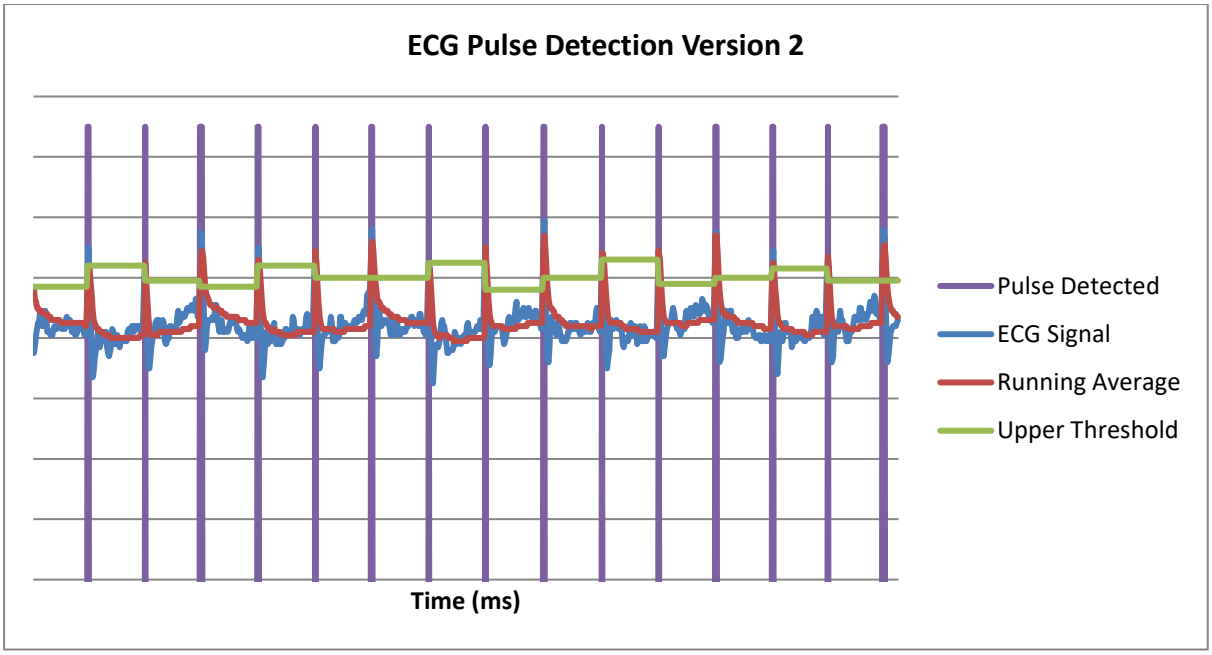



Figure 6.12 - The Result of ECG Pulse Detection Version 2

## 6.3 Temperature Detection

Temperature is another very common vital sign used in healthcare diagnosis. A temperature sensing unit is designed and implemented in the proposed device.

The temperature sensor chosen was an NTC thermistor. This was chosen because of its low thermal time constant, meaning it reaches thermal equilibrium quickly. The resistance of an NTC thermistor changes with temperature. The data sheet for the thermistor contains the resistances at various temperatures. The normal temperature for a human is between 35 to 37 degrees. Temperature vs resistance for the selected NTC thermistor datasheet is graphed in Figure 6.13.

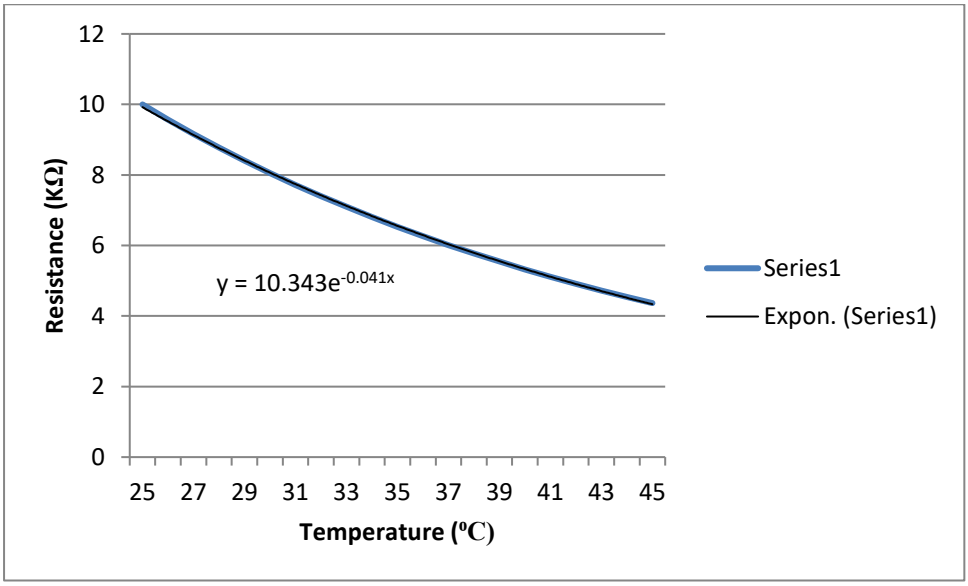



Figure 6.13 - Temperature Detection – Thermistor Temperature Extrapolation Graph

A line of best fit was generated to give an equation to model the resistance vs temperature. The equation is then used to convert the measured resistance from the thermistor into a temperature. A resistor of known value, 10KΩ, is placed in series with the thermistor. The voltage drop across the 10KΩ resistor is measured using an analogue input on the microcontroller. This circuit is shown in Figure 6.14 below.

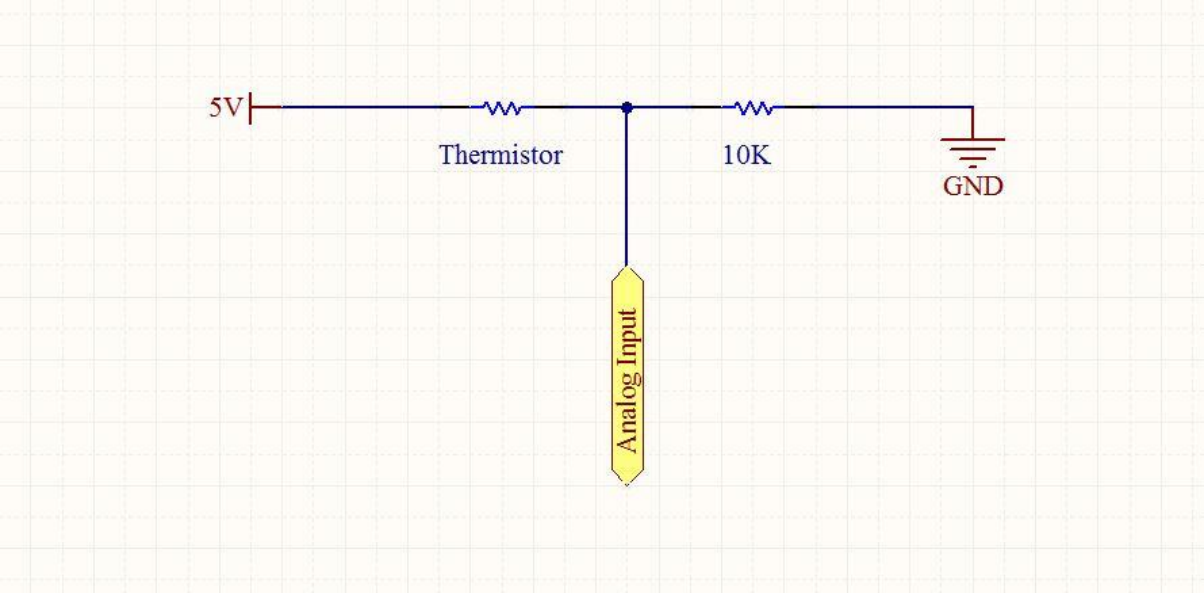



Figure 6.14 - Temperature Detection Circuit

The voltage measured can then be used to determine the resistance of the thermistor and therefore the measured temperature. Testing this algorithm showed the temperature sensor behaved as expected, responding quickly and showed that no filter is required on this signal. Below is the code used in the microcontroller to determine the measured temperature.

```
//Measure the voltage drop across a known a resistance (10K)
V10k=(analogRead(A1)*5);
V10k=(V10k/1023);

//Determine the voltage drop across the NTC and then calculated the resistance of the NTC
Vntc=(5-V10k);
Rntc=(10*V10k);
Rntc=Rntc/Vntc;

//From the resistance of the NTC, using the equation from the line of best fit, calculate the
temperature
temp=Rntc/10.343;
temp=log(temp);
temp=(temp/-0.041)+24;
```

# 7. Hand-Held Device Software Development

The hand-held device incorporates an Arduino 101 microcontroller. This section documents the software developed for the microcontroller.

## 7.1 Functionality

The Arduino program implements the methodologies documented in Data Collection Methodologies. The methodologies are summarised below:

- ECG Data verification – An accelerometer is used to determine if the patient is moving or still. When the patient is moving the ECG signal detected could be noise. If the verification test returns true, the patient is still, and the ECG data is considered valid.
- ECG Pulse Detection – This determines the patient's heart rate based off the R section of the ECG signal.
- Temperature Detection – The temperature of the patient is measured using an NTC thermistor.

The Arduino program (code) handles commands sent from the smartphone application. It can receive one of four commands from the application. Commands are sent as a number in the data format integer (Int). Below are the commands that can be sent from the application and their meanings.

- 1 – Measure ECG/heart rate. To do this command the controller uses the ECG data verification to verify the ECG signal and then uses the ECG pulse detection to measure the heart rate. It then sends the ECG and heart rate data to the application
- 2 – Measure Temperature. This command runs the temperature detection code and sends the temperature data to the application.
- 5 – Turn device off
- 9 – Stop measuring. Stop measuring the current command

In the main routine of the code, data related to the current vital sign measured is sent to the application at set intervals. The code that runs depends on the current command sent from the application. If the command '1' is being sent from the application, then the code runs the ECG Data verification first, before attempting to run the ECG pulse detection - effectively the code in the methodologies section. Throughout every cycle, it checks to see if the ECG data is still valid.

## 7.2 Methods

### 7.2.1 Power Control

The code controls the power for the system. When the microcontroller is first turned on and runs the setup code, it first checks the cell voltage of the batteries. The voltage of a LiPo battery cell should not be discharged below 3.7V. If they are discharged below this level, then the cell can get damaged. Figure 7.1 below shows the power circuit schematic used in the developed hand-held device.

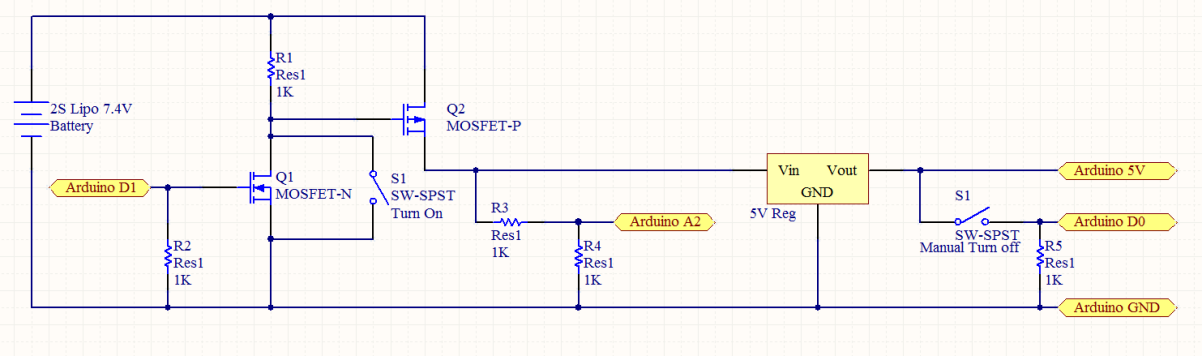



Figure 7.1 - Developed Device Power Circuit Schematic

If the cell voltage is above 3.7V the Mosfet that controls the power is latched (Q1 in Figure 7.1). The manual power switch on the device bypasses this Mosfet. If the battery voltage is below 3.7V then the Mosfet isn't latched and when the user releases the switch the system is turned off. Every cycle of the code a check is done to see if:

- The cell voltage is below 3.7V
- If the user has pressed the switch again (manual turn off)
- If a turn off command has been sent
- If Auto timeout has been reached (see 7.2.6.1 Timeout section)

If any of these conditions are met, then the Mosfet (Q1) is unlatched and the power to the system is turned off. When the system is first turned on, it waits until the bypass switch has been released before moving on in the code. If this switch remains on then the automatic turn off functions will not work, and the user won't be able to manually turn it off, therefore the code waits until the switch is released before moving on.

### 7.2.2 Communication with a Smartphone - Application Development

Using a smartphone for communication and control is a very active research area. Many Apps have been developed for different purposes. As part of the research, an app was developed for the proposed healthcare device. The device communicates to a smartphone using Bluetooth BLE. Bluetooth BLE has characteristics which effectively act as different communication channels. The characteristics definitions in the microcontroller need to match with the smartphone application. Following is the definition of the characteristics:

```
BLEService batteryService("19B10000-E8F2-537E-4F6C-D104768A1214"); // BLE Service

//Arduino To App
BLEUnsignedCharCharacteristic HeartRateChar("19B10001-E8F2-537E-4F6C-D104768A1214",
BLERead | BLEWrite |  BLENotify);
BLEUnsignedCharCharacteristic Temperature("19B10002-E8F2-537E-4F6C-D104768A1214", BLERead
| BLEWrite |  BLENotify);
BLECharacteristic ECG("19B10003-E8F2-537E-4F6C-D104768A1214",  BLERead | BLEWrite |
BLENotify, 3);
//BLEUnsignedCharCharacteristic ECG("19B10003-E8F2-537E-4F6C-D104768A1214",  BLERead |
BLEWrite |  BLENotify);
BLEUnsignedCharCharacteristic Battery("19B10004-E8F2-537E-4F6C-D104768A1214",  BLERead |
BLEWrite |  BLENotify);
BLEUnsignedCharCharacteristic Acknowledge("19B10005-E8F2-537E-4F6C-D104768A1214",
BLERead | BLEWrite |  BLENotify);
BLEIntCharacteristic OutSpare("19B10006-E8F2-537E-4F6C-D104768A1214", BLERead| BLEWrite |
BLENotify);

//Send From App to arduino
BLEUnsignedCharCharacteristic Commands("19B10011-E8F2-537E-4F6C-D104768A1214", BLERead |
BLEWrite | BLENotify);
BLEIntCharacteristic InSpare("19B10012-E8F2-537E-4F6C-D104768A1214", BLERead | BLENotify);
```

The way that the Bluetooth characteristics have been set up, allows one type of data only to be sent over each. This means that there is no need to send an identifier with the data. An example of an identifier for temperature could be a 't' prefixing the actual temperature data e.g. 30 degrees, then the data sent would be t30. With a characteristic set up for sending the temperature data, only temperature data is sent over this channel, removing the need for the identifier. This is less data to send and simpler to process.

### 7.2.3 Processing Commands

Commands sent from the smartphone application are processed in the microcontroller. Commands are sent in the data format of a character (char). To convert a char into an integer, 48 is subtracted off the read command value (in the ascii table a char value of 48 equals 0, so a char value of 57 equals the value 9). Below is the code which handles commands from the app.

```
void CheckCommand() {
  if (Commands.written()) {
    Request = (Commands.value()) - 48;
    while (Commands.written());

    Acknowledge.setValue(Request);
    if (Request == 1) {
      Serial.println("Heart Rate Command Recived");
      Command = 1;
      TimeOut = micros();
    }
```

```
  else if (Request == 2) {
    Serial.println("Temperature Command Recived");
    Command = 2;
    TimeOut = micros();
  }
  else if (Request == 5) {
    Serial.println("App turn off");
    Command = 5;
    TimeOut = micros();
  }
  else if (Request == 9) {
    Serial.println("Stop Command Recived");
    Command = 9;
    TimeOut = micros();
  }

 }
}
```

Note that the command is sent back to the app using the following code:

```
  Request = (Commands.value()) - 48;
  Acknowledge.setValue(Request);
```

The “acknowledge” characteristic is the feedback to the application that the command has been received at the device.

### 7.2.4 Sending ECG/Heart Rate Data

ECG data is sent over the ECG characteristic every 4 ms. The data sent is the raw analogue input from the AD8232 chip through the developed Butterworth filter. The values of an analogue read in an Arduino microcontroller can vary from 0-1023. The analogue input in an Arduino 101 has 10-bit resolution, rated for 5v. This results in a max value of 1023 at 5V. This cannot fit into a char data type which is the data format Bluetooth uses to send data. The largest value of the char data type is 255; hence it must be broken up. The raw analogue value is broken into 3 sets of chars and then sent. Below is the code for sending the ECG data to the application

```
int send1 = analogRead(A0);
int part1, part2, part3;
part1 = send1 % 10;
send1 = send1 - (part1);
part2 = send1 % 100;
send1 = send1 - (part2);
part3 = send1;
unsigned char heartRateCharArray[3] = { (char)(part3 / 100), (char)(part2 / 10), (char)part1 };
ECG.setValue(heartRateCharArray, 3);
```

Since the ECG data is sent at a quick regular interval (4ms), it is also used to send data on the current status of the ECG validation detection and the heart rate pulse calibration. The following statuses are also sent over the ECG characteristic and their meanings are shown below:

- 1 – Accelerometer detected movement and the ECG is no longer a valid signal – remain still
- 2 – Code is listening to the user's heart rate for calibration for pulse detection

The heart rate data is a running average of the last 10 readings. It is sent every 100 ms to the application. Below is the code for sending this to the application:

*HeartRateChar.setValue(HeartRateAve / 10);*

### 7.2.5 Sending Temperature Data

The temperature data is sent to the application every second. Below is the code which sends the temperature data to the application

Temperature.setValue(int(temp));

### 7.2.6 Other Features Implemented in the Software

#### *7.2.6.1 Timeout*

The timeout feature means that if the device was accidentally turned on, it then turns itself off after a period to save batteries. It's also a failsafe for if communication between the device and the app is lost and for some reason and the device still "thinks" it's connected. The app sends a command to the device every second when it's receiving data. For example, if the user requests to measure the temperature the app sends a '2' command every second.

How this feature works is, if the device does not see a command from the app within 1 minute, it turns itself off. Also, if the device was measuring the temperature and it does not see a command again within 20 seconds, it stops measuring the temperature. In short there is a handshake between the application and the device and if this handshake is lost, it then stops measuring and then turns itself off.

#### *7.2.6.2 Auto Low Voltage Detection*

For every cycle of the code, a check is made of the cell voltage. If the voltage of the cell is below 3.7V, the device turns off. Below is the code for checking the cell voltage:

```
void CheckVoltage() {
 float voltage = (analogRead(A2) * 3.3) / (1023); //3.3 volt reference
 Serial.println(voltage);
 Serial.println(analogRead(A2));
 if (analogRead(A2) < 662) { //7.4V across both cells - 39k resistor and 16K resistor 662=3.7V
   TurnOff();
 }
```

## 8. Cloud-Based Database Establishment

The App uses a cloud storage system called Google Firebase to handle storage and transfer information between remote patients and Doctors. Firebase is a free system, designed to make app development simple. It has many features including:

- Real-time database
- Authentication
- Storage

It comes with built in analytics software allowing insights to app usage. It is free to start with and can be scaled with ease, to allow for the extra storage and traffic as the app grows. Also, it has been developed to be used across multiple platforms such as:

- IOS – IPhone base
- Android – Android base
- Web – website base

The app developed uses three key features in Firebase. It uses the Authentication system to handle user login. Only authenticated users have access to the database and storage facilities. A Real-time database is used as a link to transfer commands and information between the doctor and the patient. Information is synchronized across connected devices in milliseconds and it handles complexities of real-time synchronization. Finally, the storage features are used to store and retrieve backed up data.

## 8.1 Database Design

The database is used for two key features:

- Sending commands from the doctor to the device via the smartphone app
- Sending information from the device to the doctor via the smartphone app

Firebase literature states that for speed the database should be a "flat layout". A flat layout means instead of having folders inside folders you have all the data in one folder. Once a user creates an account, the app creates a new sub file in the database under the email account that was used to create the account.

Names in the database cannot include a dot '.' which created a problem as all email address a have '.' in the "Servers Name.Com" etc. General rules around email addresses state several characters that cannot be used in email address, one of which is a comma ',’. However, this is not entirely true because an email address can have speech marks " " and inside these, any characters can be used. It is assumed that there will not normally be any comma's ',' in an email address, so when creating a sub file for the user, the dot '.' is replaced with a comma. For example:

Test@gmail.com becomes Test@gmail,com

The database is structured in a flat layout. An example of an account in the database is shown below:

- Test@gmail,com
    - Battery
    - Command
    - Connected
    - Connection Request
    - Date Of Birth
    - Doctor
    - ECG
    - ECGCharateristic
    - First Name
    - HeartRate
    - History
        - ECG
        - Temperature
    - Last Name
    - LastBLE
    - Temperature
    - TemperatureCharateristic

Each of the entries used in the database is discussed below:

### 8.1.1 Battery

'Battery' is used to transfer the device's current battery voltage level (as a percentage) so the doctor can determine the time remaining before it goes flat. This feature has not yet been implemented.

### 8.1.2 Command

'Command' is used to transfer commands from the doctor's application to the device. The commands and their meanings are shown below.

- 2 - Measure temperature
- 3 – Measure ECG/Heart rate
- 4 – Release Control – hand control back to the patient
- 9 – Stop measuring

### 8.1.3 Connected

'Connected' is used to show the doctor if the patient has connected to the device. Two states are available in the connected field. The two states and their meanings are listed below:

- 1 – Patient has logged in into the app (authenticated)
- 2 – Patient has logged in and has connected to the device via Bluetooth

### 8.1.4 Connection Request

The idea for this was that when the doctor connected to a patient, the connection request flag would be turned on, to see if the patient is being monitored by another healthcare professional. If not, then the doctor could monitor this patient. This feature was added so that only one healthcare professional can monitor a patient at a time. This feature is currently not being used as it is thought that there could be a need for more than one doctor/healthcare professional to monitor a patient at the same time.

### 8.1.5 Date of Birth

This stores the patient's date of birth.

### 8.1.6 Doctor

'Doctor' is used to identify if the user is a doctor or patient. If this flag is set true, then the user is defined as being a doctor. Currently this is hardcoded and cannot be changed from the app.

### 8.1.7 ECG

This stores the current ECG value as measured from the device.

### 8.1.8 ECG Characteristic

The ECG characteristic is a Boolean value. Bluetooth communication has several characteristic (channels) for transmitting data between the app and the device. To be able to measure the ECG/heart rate from the device a number of communication channels have to be available. This bit becomes true when the channels have been established and the ECG can then be recorded. This bit controls the ability for the doctor to send a "measure the ECG" command to the device.

### 8.1.9 First Name

This stores the patient's first name.

### 8.1.10 Heart Rate

This stores the current heart rate value as measured on the device.

### 8.1.11 History

The history section in the database is effectively a look up table for the storage feature in Firebase. To download a file from the cloud storage facility offered by Firebase, the file name is required. At the time of developing this app, Firebase did not have a method for returning the file names of files stored in the storage facility. To be able to download files and therefore offer the historical data feature in the application, a system had to be developed for saving the file names of pervious records, hence a look up table was created. The look up table is effectively a list of file names stored in the storage facility.

Every time a new record is saved in the storage facility a new entry is added into this table. Data entered into this table is stored by the time and date it is created, for example, if an ECG record was created on 2016-11-07 at 21-53 then this would be entered in to look up table as 1611072153. Being a number allows for effective handling within the system for:

- Searching within records in a defined time range.
- Entries are ordered in the date they were created.

This allows a large number of records to be saved and stored by the date they are created, with a link to the file name in the storage facility. The history section has two sub folders both with references to the saved files name.

- ECG
- Temperature

### 8.1.12 Last Name

This stores the last name of the patient.

### 8.1.13 Last BLE

This field holds the last Bluetooth address which the patient was connected to. This means that when the patient uses the app and tries to connect to the device, it automatically scans for this address first and if it is available, displays it to the patient.

### 8.1.14 Temperature

This stores the current temperature value as measured on the device.

### 8.1.15 Temperature Characteristic

The temperature characteristic is a boolean value. The Bluetooth communication has several characterises (channels) for transmitting data between the app and the device. To be able to measure the temperature from the device a number of communication channels have to be available. This flag becomes true when the channels have been established and the temperature can then be recorded. This bit controls the ability for the doctor to send a “measure temperature” command to the device.

## 8.2 Storage

The storage feature saves a backup of the data which can be review at a later date. It has a similar structure to the database. It is as follows:

- Test@gmail,com
    - ECG
    - Temperature

Inside the ECG subfolder contains all the ECG records saved and the temperature subfolder contains all the temperature records. The file names of these are located in the database, history subfolder as mentioned above section, 8.1.11 History.

## 8.3 Authentication

Authentication is handled within Google Firebase itself. To sign in and gain access to the database and storage system the user must sign in using an email and password. Once logged in the user can read and write data from the database but the app restricts this to their subfolder and only allows certain data fields to be written to.

# 9. Smartphone Application

This section documents the smartphone application (App) developed through this research project. It acts as the user interface and plays an important role in the communication between the device and the cloud database. The smartphone application is regularly referred to as the doctor's or patient's app. These are the same app. When it is referred to as the patient's app, it is referring to the patient's activities within the application, designed to be used by a patient and vice versa with the doctor's app.

## 9.1 Login Screen

The screen shown in Figure 9.1 is the starting screen and controls the whole app. It allows the user to login into the database through the authentication process as either a doctor or patient provided there is an internet connection. If the user is not authenticated, then they can either create an account from here or use the device as a non-authenticated user.


Figure 9.1 - Smartphone Application - Login Screen

### 9.1.2 Functionality

The main function of the login screen is to authenticate the user, so the healthcare professional knows the information is coming from the correct user.

### 9.1.2 Methods

The key methods in the login screen are:

- User authentication
- Password reset
- Create account

This section discusses these methods.

#### *9.1.2.1 Authenticate (Login)*

The user logs in using an email address and password which has been set up by the user when the account was created. When the user pushes the login button on the app, it checks to see if the user has entered in a username and password. If this has been done, it then checks for internet connection before attempting to authenticate the user. To authenticate the user, a method from the Firebase application programming interface (API) called “signInWithEmailAndPassword” is used which has a built in call back function (automatically triggered once the function has returned with a result) called "OnCompleteListener” which indicates if the sign in was successful. The authentication code is shown below.

```
public void login(String email, String password){
    mAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(this,
new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            Log.d(TAG, "signInWithEmail:onComplete:" + task.isSuccessful());
            if (!task.isSuccessful()) {
                Log.w(TAG, "signInWithEmail", task.getException());
                LoadingBar.setVisibility(View.GONE);
                Information.setText("Authentication Failed, Check User Name And
Password");
            }
            else{
                Information.setText("Authentication Complete");

                CheckDoctorAccount=UserName.getText().toString();
                CheckDoctor(CheckDoctorAccount);

            }

                }
            });
```

If the authentication method is successful, then the app will load one of two screens depending on if the user is a listed as a patient or doctor. A flag in the user’s database is used to indicate if the user is a doctor. To read this value from the database, the Firebase API has a “Listener” function which is attached to the user’s subfolder in the database. A “Listener” returns data for the field it is attached too. It returns a “dataSnapshot” of all the information in that subfolder. A “dataSnapshot” contents information saved at the time the data was read. From here it can be seen if the user is registered as a doctor and if so open the screen designed to be used for a doctor, and if not open the screen designed to be used for a patient. The Code for this is shown below.

```
mDatabase.child(String.valueOf(EmailDotsRemoved)).addListenerForSingleValueEvent(
        new ValueEventListener() {
            @Override
```

```
            public void onDataChange(DataSnapshot dataSnapshot) {
                // Get user value
                LoadingBar.setVisibility(View.GONE);
                Object data = dataSnapshot.getValue();
                Log.i("info", "Doctor Data "+
dataSnapshot.child("Doctor").getValue());
                if
(dataSnapshot.child("Doctor").getValue().toString().matches("true")){
                    Log.i("info:","Doctor Login");
                    CreateAccountBar.setVisibility(View.GONE);
                    DoctorDashboard(DBUserEmail);
                }
                else {
                    CreateAccountBar.setVisibility(View.GONE);
                    dashboard(DBUserEmail);
                }
            }
            @Override
            public void onCancelled(DatabaseError databaseError) {
                Log.w(TAG, "getUser:onCancelled", databaseError.toException());
            }
        });
```

### 9.1.2.2 Reset Password

The app allows the user to reset their password if the user forgets it. This is done using the "sendPasswordResetEmail" function in the Firebase API. It requires the user to enter an email address for a reset password to be sent to. This method has an "onCompleteListener" to tell the app if it has successfully reset the password or has failed. The code is shown below.

```
public void ResetPasswordFunction(String emailAddress){
    mAuth.sendPasswordResetEmail(emailAddress).addOnCompleteListener(new
OnCompleteListener<Void>() {
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if (task.isSuccessful()) {
                //Log.d(TAG, "Email sent.");
                ResetPasswordBar.setVisibility(View.GONE);
                ResetInformation.setText("Password sent to: " +
ResetEmail.getText().toString());
            }
            else {
                ResetInformation.setText("Failded to reset password" );
                ResetPasswordBar.setVisibility(View.GONE);
            }
                }
            });
}
```

### 9.1.2.3 Create Account

The login page also allows a users to create a new account for the Firebase Authentication. This is done by using the "createUserWithEmailAndPassword" method in the Firebase API. The user is required to enter in an email address, a password twice which must match each other, a first and last name and a date of birth, all of which is stored in the database if the "createUserWithEmailAndPassword" method returns successful. Some of the create new user account code is shown below.

```
public void CreateNewAccount(String email, String password){
    mAuth.createUserWithEmailAndPassword(email,
password).addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
```

```
        @Override

        public void onComplete(@NonNull Task<AuthResult> task) {
            Log.d(TAG, "createUserWithEmail:onComplete:" + task.isSuccessful());

            if (!task.isSuccessful()) {
                CreateAccountInfo.setText("Account not created");
            }
            else{
                CreateAccountInfo.setText("Created a new account");
               //     CreateAccountBar.setVisibility(View.GONE);
                mDatabase = FirebaseDatabase.getInstance().getReference();

                StringBuilder EmailDotsRemoved = new
StringBuilder(CreateAccountEmail.getText().toString().length());
                for (int i = 0; i <
CreateAccountEmail.getText().toString().length(); i++) {
                    if
(CreateAccountEmail.getText().toString().toCharArray()[i]=='.') {
                        EmailDotsRemoved.append(',');
                    }
                    else
EmailDotsRemoved.append(CreateAccountEmail.getText().toString().toCharArray()[i]);
                }
                EmailDotsRemoved.toString();


                Log.i("info","Email with dots removed " +EmailDotsRemoved);
              //  mDatabase.child(String.valueOf(EmailDotsRemoved));


mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Doctor").setValue(false);

mDatabase.child(String.valueOf(EmailDotsRemoved)).child("HeartRate").setValue(0);

mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Temperature").setValue(0);

mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Command").setValue(0);

mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Battery").setValue(0);

mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Connected").setValue(0);
                mDatabase.child(String.valueOf(EmailDotsRemoved)).child("First
Name").setValue(CreateAccountFirstName.getText().toString());
                mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Last
Name").setValue(CreateAccountLastName.getText().toString());
                mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Date Of
Birth").setValue(CreateAccountDOB.getText().toString());
                mDatabase.child(String.valueOf(EmailDotsRemoved)).child("Connection
Request").setValue(0);

                CheckDoctor(CreateAccountEmail.getText().toString());
            }
        }

    });
}
```

### 9.1.3 Features

The key feature in the login screen is to check for internet connection. This section discusses this feature.

### 9.1.3.1 Internet Connection

The main function of the app is to transfer information from a user in a remote location to a doctor. To do this, a real-time database was established using Google Firebase. To be able to transfer information from the app to the database and vice versa, internet connection is required. In a smartphone this can either be 3g/4g or Wi-Fi. Before authenticating the user or doing any activity which requires the database, internet connectivity is checked. The code for this check is shown below:

```
ConnectivityManager connManager = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo mWifi = connManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
NetworkInfo moblie = connManager.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
//Check Wifi
if (mWifi != null)
if (mWifi.isConnected()) {
    login(UserName.getText().toString(), Password.getText().toString());
}
//Check Mobile data 4g/3g
else if (moblie != null){
    if (moblie.isConnected()) {
        login(UserName.getText().toString(), Password.getText().toString());
    }
}
//If either is connected then display no internet connection
else {
    Information.setText("Internet Connection Is Required Before Login Can Occur");
    connectInternet();
}
```

If the app determines there is no internet connection then a dialog box appears saying "Internet Connection Is Required to Login and Use the Application, Connect to Internet and Reopen EHealth or Use the Non-Login User Method (Not Recommended)" with two buttons:

1. Takes the user to the Wi-Fi settings page on the smartphone
2. Closes the dialog box.

The code for the dialog box is shown below:

```
public void connectInternet(){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setMessage("Internet Connection Is Required To Login And Use The
Application, Connect To Internet And Reopen EHealth Or Use The Non-Login User
Method (Not Recommended)")
            .setTitle("Unable To Login")
            .setCancelable(true)
            .setPositiveButton("Wi-Fi Settings",
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id) {
                            Intent i = new Intent(Settings.ACTION_WIFI_SETTINGS);
                            startActivity(i);
                        }
                    }
            )
            .setNegativeButton("Cancel",
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id) {
                            MainActivity.this.onStart();
                        }
                    }
            );
    AlertDialog alert = builder.create();
    alert.show();
}
```

## 9.2 Patient Dashboard Screen

The patient dashboard is the screen used by authenticated users who are not registered as a doctor. This activity in the app has a primary task of transferring information from the device to the database and vice versa. A screen shot of the application is shown in Figure 9.2 below.


Figure 9.2 - Smartphone Application - Patient Dashboard Screen

### 9.2.1 Functionality

The main function of the patient dashboard screen is to communicate with the device and to the Google Firebase Database.

It controls the device by sending commands, either from the patient or from the doctor through the database. It handles the incoming information from the device and saves it to the database.

### 9.2.2 Methods

The key methods in this screen are:

- Bluetooth communication
- Database communication
- Receiving data from the device

This section discusses these methods.

### *9.2.2.1 Communication – Bluetooth*

**Characteristics:**

As mentioned in section, 7.2.2 Communication with a Smartphone - Application Development, the device uses Bluetooth BLE and has several characteristic. These characteristics must match the characteristics in the Arduino code in the device.

The following characteristics are set in the application.

```
public static String EHealthUUID = "19b10000-e8f2-537e-4f6c-d104768a1214";
//Bluetooth Service called Ehealth
public static String HeartRateUUID = "19b10001-e8f2-537e-4f6c-d104768a1214";
public static String TempUUID = "19b10002-e8f2-537e-4f6c-d104768a1214";
public static String ECGUUID = "19b10003-e8f2-537e-4f6c-d104768a1214";
public static String BatteryUUID = "19b10004-e8f2-537e-4f6c-d104768a1214";
public static String AcknowledgeUUID = "19b10005-e8f2-537e-4f6c-d104768a1214";
public static String CommandUUID = "19b10011-e8f2-537e-4f6c-d104768a1214";
```

The Bluetooth code runs as a service, which runs in the background of the app. This handles all Bluetooth activities from sending to receiving data, connecting/disconnecting etc. The code is modified from examples provided on the Android developer website.

**Receiving Data:**

Data can be sent from the device to the application over several characteristics, as follows:

- Heart Rate – All heart rate data is sent over this characteristic
- Temperature - All Temperature data is sent over this characteristic
- ECG - All Temperature data is sent over this characteristic
- Acknowledge – Feedback to the application from the device is sent over this characteristic

The following code is used for receiving data from the device in the Bluetooth service, which is then sent to the patient dashboard activity. When new data is received, the service calls a method called “boardcastUpdate” which checks to see which characteristic the data was sent over. It then reads the data into a string and then using the “intent.putExtra” command, it sends the data back to the main activity. The “boardcastUpdate” method is shown snippet of code below.

```
private void broadcastUpdate(final String action,
                             final BluetoothGattCharacteristic characteristic) {
    final Intent intent = new Intent(action);

    if
(GattAttributesUUIDs.HeartRateUUID.matches(characteristic.getUuid().toString())) {
        final byte[] data = characteristic.getValue();
        if (data != null && data.length > 0) {
            final StringBuilder stringBuilder = new StringBuilder(data.length);
            for(byte byteChar : data)
                stringBuilder.append(String.format("%d ", byteChar));
            intent.putExtra("Type","HeartRate");
            intent.putExtra(EXTRA_DATA, stringBuilder.toString());
        }
    }

    else if
```

```
(GattAttributesUUIDs.ECGUUID.matches(characteristic.getUuid().toString())) {
        final byte[] data = characteristic.getValue();
        if (data != null && data.length > 0) {
            final StringBuilder stringBuilder = new StringBuilder(data.length);
            for(byte byteChar : data)
                stringBuilder.append(String.format("%d ", byteChar));
            intent.putExtra("Type","ECG");
            intent.putExtra(EXTRA_DATA, stringBuilder.toString());
        }
    }
    else if
(GattAttributesUUIDs.AcknowledgeUUID.matches(characteristic.getUuid().toString()))
{
        final byte[] data = characteristic.getValue();
        if (data != null && data.length > 0) {
            final StringBuilder stringBuilder = new StringBuilder(data.length);
            for(byte byteChar : data)
                stringBuilder.append(String.format("%d ", byteChar));
            intent.putExtra("Type","Acknowledge");
            intent.putExtra(EXTRA_DATA, stringBuilder.toString());
        }
    }
    else if
(GattAttributesUUIDs.TempUUID.matches(characteristic.getUuid().toString())) {
        final byte[] data = characteristic.getValue();
        if (data != null && data.length > 0) {
            final StringBuilder stringBuilder = new StringBuilder(data.length);
            for(byte byteChar : data)
                stringBuilder.append(String.format("%d ", byteChar));
            intent.putExtra("Type","Temperature");
            intent.putExtra(EXTRA_DATA, stringBuilder.toString());
        }
    }
    else {
        intent.putExtra("Type", "Nothing");
        intent.putExtra(EXTRA_DATA, "Nothing Receved");
    }
    sendBroadcast(intent);
}
```

The services send data back to the main activity using a function called “sendBroadcast”. This sends a boardcast which the main activity is listening for “BroadcastReceiver”. The code shown below details the “BroadcastReceiver” in the main activity for handling temperature data sent from the Bluetooth service to the main activity.

```
private final BroadcastReceiver mGattUpdateReceiver = new BroadcastReceiver() {
        @TargetApi(Build.VERSION_CODES.HONEYCOMB)
        @Override
        public void onReceive(Context context, Intent intent) {
            final String action = intent.getAction();

if (BluetoothLeService.ACTION_DATA_AVAILABLE.equals(action)) {
                String type = intent.getStringExtra("Type");
                if (type != null) {
                    Log.i("info", "type: " + type);

                    if (type.matches("Temperature")) {

                        String data = (intent.getStringExtra(EXTRA_DATA));

                        if (data != null) {
                            TemperatureFieldIO.setText(data);
```

```
                        }
                    }
                }
            }
        }
    };
```

To read data from a specific characteristic, a “readCharacteristic” method is used. This requires the characteristic to be entered, which the data is to be read from. For example, to read the temperature the following code is used;

```
mBluetoothLeService.readCharacteristic(CharacteristicTemp);
```

If data is available on that characteristic, then the “BluetoothLESerive.broadcastUpdate” method is called which then sends data back to the main activity and invokes the “BroadcastReceiver”.

**Sending Data:**

Sending data is handled by the “sendCharacteristic” method in the Bluetooth service. Data is sent to the device using the command characteristics “CharacteristicCommand” in the data format of an int. The Data sent then gets converted into byte data type which the Bluetooth service then sends. There are four commands which can be sent to the device from the app. The commands and their meanings are shown below:

- 9 - Stop measuring
- 2 - Measure Temperature
- 1 - Measure ECG/Heart rate
- 5 - Turn off device

Below shows the code for the function “sendCharacteristic” in the Bluetooth service which handles sending data to the device.

```
public void sendCharacteristic(BluetoothGattCharacteristic characteristic,int
senddata) {
    if (mBluetoothGatt == null) {
        return;
    }
    BluetoothGattCharacteristic charac = characteristic;
    if (charac == null) {
        return;
    }
    String str= String.valueOf(senddata);
    byte value = (byte) senddata;
    charac.setValue(str);
    boolean status = mBluetoothGatt.writeCharacteristic(charac);
    return;
}
```

In the main activity, the following code is used to send data

```
mBluetoothLeService.sendCharacteristic(CharacteristicCommand, 2);
```

**Scanning for Devices**

Connecting to a Bluetooth device requires a user to select the device to connect to; therefore, scanning for devices is handled by the main activity. When scanning for devices, the first thing the app does, is to see if the smartphone supports Bluetooth BLE and checks if the Bluetooth is turned on. If not, it prompts the user to turn on Bluetooth. The following section of code checks if the smartphone supports Bluetooth BLE and then prompts the user to enable Bluetooth if it's turned off, before it starts scanning for devices.

```
public void SearchBLE() {
    mHandler = new Handler();

    // Use this check to determine whether BLE is supported on the device.  Then
you can
    // selectively disable BLE-related features.
    if (!getPackageManager().hasSystemFeature(PackageManager.FEATURE_BLUETOOTH_LE))
{
        Toast.makeText(this, R.string.ble_not_supported,
Toast.LENGTH_SHORT).show();
        finish();
        ConnectToDeviceInfo.setText("Your Phone Doesn't Support Bluetooth BLE");
    }

    // Initializes a Bluetooth adapter.  For API level 18 and above, get a
reference to
    // BluetoothAdapter through BluetoothManager.
    final BluetoothManager bluetoothManager =
            (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
    mBluetoothAdapter = bluetoothManager.getAdapter();

    // Checks if Bluetooth is supported on the device.
    if (mBluetoothAdapter == null) {
        Toast.makeText(this, R.string.error_bluetooth_not_supported,
Toast.LENGTH_SHORT).show();
        finish();
        ConnectToDeviceInfo.setText("Your Phone Doesn't Support Bluetooth BLE");
        return;
    }
    if (!mBluetoothAdapter.isEnabled()) {
        if (!mBluetoothAdapter.isEnabled()) {
            Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);

        }
    }

    // Initializes list view adapter.
    mLeDeviceListAdapter = new LeDeviceListAdapter();
    scanLeDevice(true);



}
```

Bluetooth devices are then searched for. Every time a new Bluetooth device is found, a call back method called "mLEScanCallback" is invoked which saves the address of the new Bluetooth device found and displays the device name using an adaptor called "LeDeviceListAdapter". The code for scanning Bluetooth devices ("scanLeDevice"), call back when a Bluetooth device found ("mLeScanCallback") and handling the Bluetooth device data ("LeDeviceListAdapter") is shown below.

```
private void scanLeDevice(final boolean enable) {
    if (enable) {
        // Stops scanning after a pre-defined scan period.
        mHandler.postDelayed(new Runnable() {
            @TargetApi(Build.VERSION_CODES.JELLY_BEAN_MR2)
            @Override
            public void run() {
                mScanning = false;
                ConnectToDeviceBar.setVisibility(View.GONE);
                mBluetoothAdapter.stopLeScan(mLeScanCallback);
                invalidateOptionsMenu();
            }
        }, SCAN_PERIOD);

        mScanning = true;
        mBluetoothAdapter.startLeScan(mLeScanCallback);
    } else {
        mScanning = false;
        mBluetoothAdapter.stopLeScan(mLeScanCallback);

    }
    invalidateOptionsMenu();
}

private BluetoothAdapter.LeScanCallback mLeScanCallback = new
BluetoothAdapter.LeScanCallback() {

    @Override
    public void onLeScan(final BluetoothDevice device, int rssi, byte[] scanRecord)
{
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                mLeDeviceListAdapter.addDevice(device);
                ConnectToDeviceInfo.setText("Select The Device To Connect To");
            }
        });
    }
};

private class LeDeviceListAdapter extends BaseAdapter {
    //   private ArrayList<BluetoothDevice> mLeDevices;
    private LayoutInflater mInflator;

    public LeDeviceListAdapter() {
        super();
        mLeDevices = new ArrayList<BluetoothDevice>();
        mInflator = DeviceControlActivity.this.getLayoutInflater();
    }

    public void addDevice(BluetoothDevice device) {
        if (!mLeDevices.contains(device)) {
            mLeDevices.add(device);
            ConnectToDeviceBar.setVisibility(View.GONE);
            if (LastBLE!=null) {
                if (device.getAddress().matches(LastBLE)) {
                    ConnectToDeviceInfo.setText("Last Connected Device");
                    Device1.setText(device.getName());
                    Device1.setVisibility(View.VISIBLE);
                }
            }
        else {
            if (DeviceFound == 0) {
                Device1.setText(device.getName());
                Device1.setVisibility(View.VISIBLE);
                DeviceFound++;

            } else if (DeviceFound == 1) {
```

```
                Device2.setText(device.getName());
                Device2.setVisibility(View.VISIBLE);
                DeviceFound++;
            } else if (DeviceFound == 2) {
                Device3.setText(device.getName());
                Device3.setVisibility(View.VISIBLE);
                DeviceFound++;
            } else if (DeviceFound == 3) {
                Device4.setText(device.getName());
                Device4.setVisibility(View.VISIBLE);
                DeviceFound++;
            }
        }
    }
    public BluetoothDevice getDevice(int position) {
        return mLeDevices.get(position);
    }
}
```

**Bluetooth Connection**

Once the user selects the BLE device they wish to connect to, the app then connects to that device by using the following snippet of code:

```
mBluetoothGatt = device.connectGatt(this, false, mGattCallback);
mBluetoothDeviceAddress = address;
```

**Bluetooth Disconnection**

Disconnecting from the device is done by using the follow section of code

```
mBluetoothGatt.disconnect();
```

### *9.2.2.2 Communication – Database*

Communication from the database to the app is used when the doctor sends commands (From the doctor's activity in the application) to the device. To handle this, function from the Firebase API called "ValueEventListener" is used and is attached to the users (patients) command section in the database. Every time the value of the command changes, this function is called. The code for this is shown below:

```
ValueEventListener CommandListener = new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        if (dataSnapshot.getValue() == null) {
        } else {
            Command = Integer.parseInt(dataSnapshot.getValue().toString());
            if (Command > 1) {
            } else {
            }
            //Measure Temp Command From Doctor
            if (Command == 3) {
                HeartRateData();

            }
            //Heart Rate Command
            else if (Command == 2) {
                TemperatureData();
            }

            //Stop Measuring Command
```

```
            else if (Command == 9) {
                StopData();
            }
            //Reasle Command - Give Control Back To User
            else if (Command == 4) {
                StopData();
            }
        }
    }

    @Override
    public void onCancelled(DatabaseError databaseError) {
        // Getting Post failed, log a message
        Log.w(TAG, "loadPost:onCancelled", databaseError.toException());
        // ...
    }
};
```

### 9.2.2.3 Getting Data from the Device

The user or the doctor selects a function – for example measure the temperature. To measure the temperature the app sends a command to the device, requesting that the temperature is to be measured. This command is sent every 10ms until the app receives an acknowledgement. The acknowledgement is simply the command sent back from the device indicating that the device has received the command and that temperature data will be sent through shortly. The code shown below is used for sending the temperature command to the device using a method called "TemperatureData".

```
public void TemperatureData() {
    CommandSent = 2;
    HeartRateScreen.setVisibility(View.GONE);
    TemperatureScreen.setVisibility(View.VISIBLE);
    TemperatureInfo.setText("Sending Command To Device");
    if (CharacteristicCommand != null && CharacteristicAcknowledge != null &&
CharacteristicTemp != null) {
        TemperatureInfo.setText("");
        StopCurrentCommand();
        StopCommand.setVisibility(View.VISIBLE);
        MeasureTempCommandOn = 1;

        timerTempRequest = new Timer("FileWather", true);
        //ever 100ms sent a new command to bluetooth untill the comand is sent back
        timerTempRequest.scheduleAtFixedRate(new TempCommand(), 0, 10);
        timerTempRequestReponse = new Timer("FileWather", true);
        timerTempRequestReponse.scheduleAtFixedRate(new ComandRespones(), 0, 10);

    } else if (CharacteristicCommand == null) {
        TemperatureInfo.setText("Missing Bluetooth Service, Try Disconnecting And
Reconnecting To Device");
    } else if (CharacteristicAcknowledge == null) {
        TemperatureInfo.setText("Missing Bluetooth Service, Try Disconnecting And
Reconnecting To Device");
    } else if (CharacteristicTemp == null) {
        TemperatureInfo.setText("Missing Bluetooth Service, Try Disconnecting And
Reconnecting To Device");

    }


}
```

Once the command has been acknowledged, the app stops sending the temperature command to the device and then starts listening for data. Listening for data is done by reading the temperature characteristic every 100ms. The code below is taken from a function called “ComandResponse” for handling the temperature acknowledgement. The “CommandResponse” function monitors the command being send to the device and the acknowledgment being send back from the device.

```
if (byteChar == 2 && CommandSent == byteChar) {
    if (MeasureTempCommandOn > 0) {
        timerTempRequest.cancel();
        timerTempRequestReponse.cancel();
        MeasureTempCommandOn = 2;
        try {
            if (outputWriter == null) {
                CreateFile("Temperature");
            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        TemperatureUpdate = new Timer("FileWather", true);
        //   TemperatureInfo.setText("");
        TemperatureUpdate.schedule(new TempListern(), 0, 100);
    }
```

The same applies for sending ECG/Heart rate and stop commands from the user or from the doctor interface via the Google Firebase to the device.

### 9.2.3 Features

This section documents key features in the patient dashboard screen. Key features in the screen are:

- Data backup (History)
- Turn off device
- Automatic scanning for last connected Bluetooth device
- No device control until all Bluetooth characteristics are available

#### *9.2.3.1 Data Backup (History)*

The data backup feature or history saves data from previous vital signs readings. Every time a new measurement is captured the app creates a Microsoft Excel file for saving data to, which can then be used later for reviewing past measurements. As the data is captured, it is saved to the excel file. Once the current measurement has stopped or a stop command has been sent, the app saves the excel file and uploads this to the Google Firebase Storage system. Figure 9.3, below is a screen shot of the app showing a historical ECG data recording.

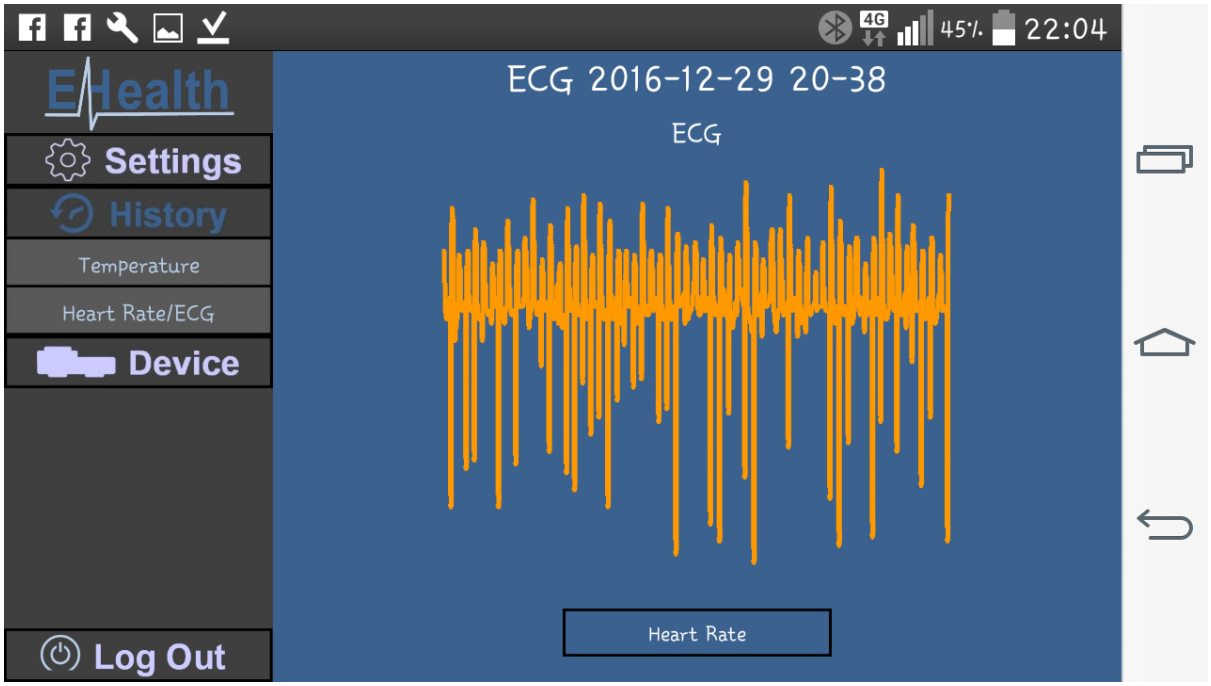



Figure 9.3 - Smartphone Application - Patient Dashboard Screen Feature - History

With the data been saved in a Microsoft Excel file, once it is downloaded to the smartphone, it can then be opened in excel or an equivalent app. This allows the user to manipulate the data as they please. Figure 9.4 below shows the same historical data opened in Microsoft Excel.

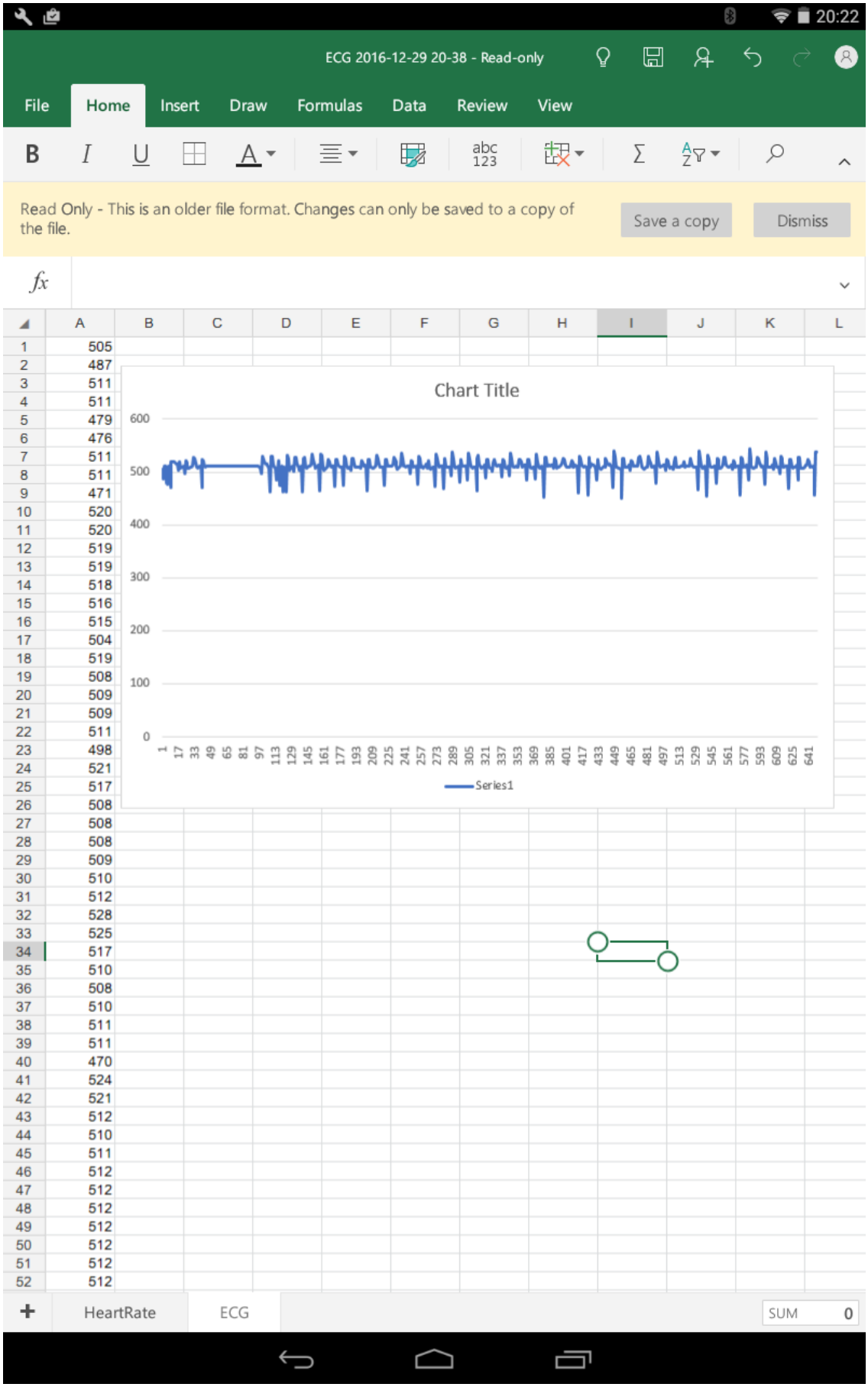



Figure 9.4 - Smartphone Application - Patient Dashboard Screen Feature - Opening Historical Data in Microsoft Excel

To help find past records the user can search within a selected date range. Figure 9.5 below shows the search feature in the app.

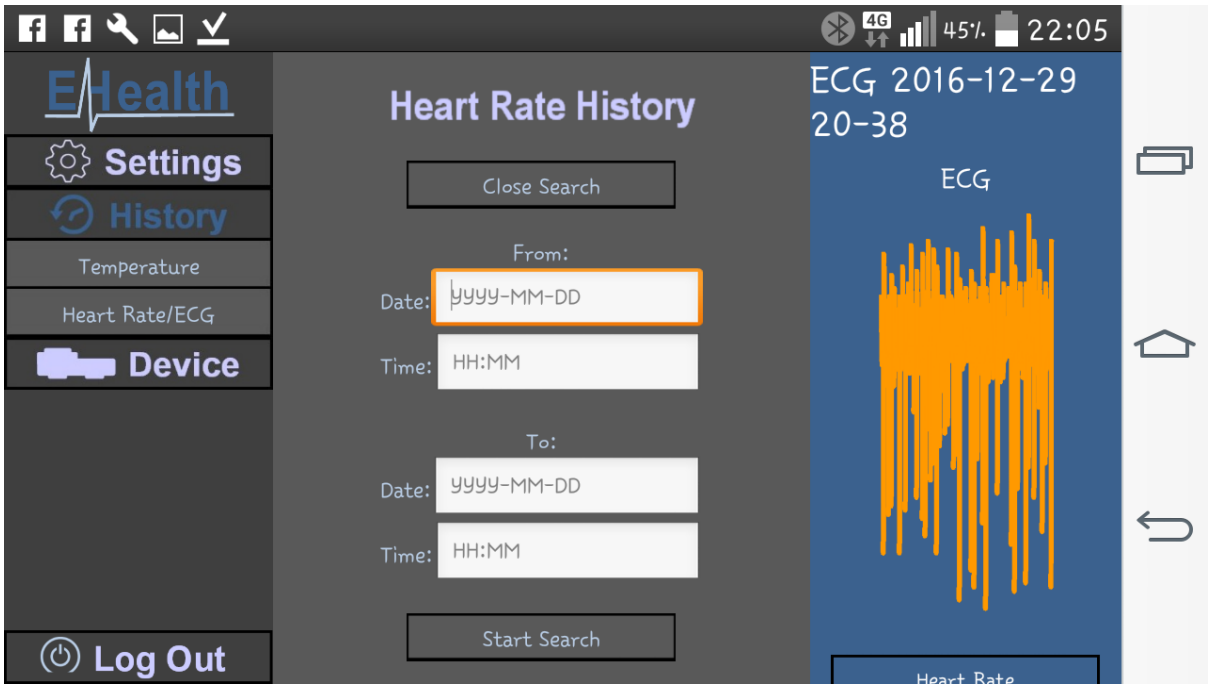



Figure 9.5 - Smartphone Application - Patient Dashboard Screen Feature - History Searching

Data is laid out in a chronological way where the most recent data is shown first. Results are shown in a simple and easy to select manor as the screen shot shown in Figure 9.6.

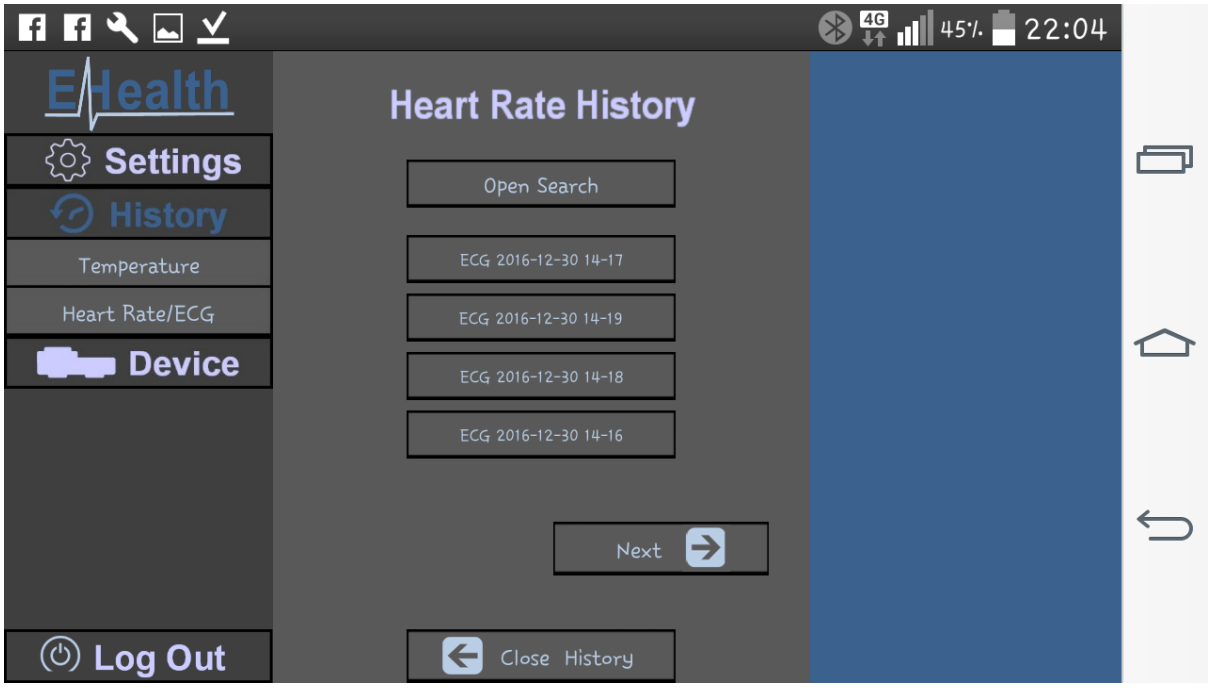



Figure 9.6 - Smartphone Application - Patient Dashboard Screen Feature - History Results

#### *9.2.3.2 Turn off Device*

The device can be turned off from the app, once Bluetooth communication has been established. To do this, a command is sent over the command characteristic which tells the device to turn off.

#### *9.2.3.3 Automatic Scanning for Last Connected Bluetooth Device*

When the user starts scanning for Bluetooth devices, the app checks to see if the last connected device is turned on by scanning for the last saved Bluetooth address. To do this the app first checks the database to see if a Bluetooth address is saved. It then scans for available Bluetooth devices and if it finds one that matches the address of the last connected device, it then displays this.

#### *9.2.3.4 No Device Control until All Characteristics Available*

Measurement methods are not available until all the characteristics required to perform the measurements are available. For example, to measure the ECG/Heart rate four characteristics are required. They are as follows:

- Command – To send the command to the device
- Acknowledge – To know the device has received the command
- ECG – Sends the ECG data from the device to the app
- Heart rate – Sends the Heart rate data from the device to the app

If any one of the four characteristics listed above is missing then the ECG/Heart rate can't be measured, so the button used to control this measurement is not available.

## 9.3 Doctor's Dashboard

The doctor's dashboard becomes available to the user if during the authentication/login process the account is flagged as being a doctor. This activity is similar in appearance to the patient's dashboard screen. The main difference with this activity is that it's not controlling the device directly. It is only required to communicate to the Firebase database. Figure 9.7 shows the doctor's dashboard screen.


Figure 9.7 - Smartphone Application - Doctor Dashboard Screen

### 9.3.1 Functionality

The main function of the doctor's dashboard is to communicate to Google Firebase Database which in turn communicates with the device at a remote location.

It is designed to get information from the database and to send commands over the database to the patient's app. Here, the term "patient's app" is referred to as the user who is not registered as being a doctor. The patient is using a different smartphone to the doctor.

### 9.3.2 Methods

The doctor's dashboard has several key methods associated with it. They are as follows:

- Connecting to a patient
- Sending commands to a remote (patients) device
- Retrieving data

This section discusses these methods.

### 9.3.2.1 Connecting to a Patient

For the doctor to control the patient's device, first they must select the patient's account that they wish to view. This is done by clicking on the "Patient" section in the menu. Once connected to a patient's account, the user can access the patient's historical data in a similar manner as discussed in the section above in section 9.2.3.1 Data Backup (History). Once the patient is logged in to their account and is connected to the device and all the required Bluetooth characteristics have been found, then the doctor can send commands to the device. Figure 9.8 below shows a screen shot of the doctor logging into a patient's account.


Figure 9.8 - Smartphone Application - Doctor Dashboard Screen - Connecting To Patient

### 9.3.2.2 Sending Commands to a Remote Device

Commands are sent to the remote device using the database. To achieve this, commands are written to the connected patient's command field within the database. Once a command is written to the database, the patient's app then process the command and passes this onto the remote device as discussed in section 9.2.2.2 Communication – Database. The remote device then collects the information and sends this back to the doctor through the database using the method discussed in section, 9.2.2.3 Getting Data from the Device. The commands the doctor can send and their meanings are shown below:

- 2 - Measure temperature
- 3 - Measure heart rate and ECG
- 4 – Release control – Hand control of the device back to the patient
- 9 – Stop current measurement

Writing to the patients sub folder in the database is simple and is done in one line of code, as follows:

```
mDatabase.child(String.valueOf(PatientEmailWithoutDots)).child("Command").setValue(
2);
```

#### *9.3.2.3 Retrieving Data*

Once a command has been sent to the device via the database and the patient's smartphone, then the app attaches an "addValueEventListener" to the data field which is expected to change once the data has been measured from the device and sent back to the database. An "addValueEventListener" is a method in the Firebase API which is used to call a function every time a selected field changes. For example, if the doctor wishes to measure the patient's temperature, a temperature command is written to the database. An "addValueEventListener" is added to the temperature field in the database so every time the temperature field changes, a function is called in the doctor's app which updates the current temperature field in the app. Below is the code for doing this:

Where the "addValueEventListener" Is added. When data in the temperature field changes, call "TempListener" method

```
mDatabase.child(String.valueOf(PatientEmailWithoutDots)).child("Temperature").addVa
lueEventListener(TempListener);
```

The Function "TempListener" is called when temperature field in the database changes. This updates a text field on the app.

```
ValueEventListener TempListener = new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        Log.i("info: ", "Temperature Changed On Patient: " + dataSnapshot);
        if (dataSnapshot.getValue()== null){}
        else {
            TemperatureFeild.setText((dataSnapshot.getValue().toString()));
        }
    }
    @Override
    public void onCancelled(DatabaseError databaseError) {
        // Getting Post failed, log a message
        Log.w(TAG, "loadPost:onCancelled", databaseError.toException());
        // ...
    }
};
```

### 9.3.3 Features

This section documents key features in the doctor's dashboard screen. The key features in the screen are:

- Patient lockout
- Historical data

#### *9.3.3.1 Patient Lock Out*

Once a doctor sends a command to the device, the patient is locked out from sending commands to the device. Effectively the doctor has control of the device until they release control or log out from monitoring the patient. This means the doctor can get the information they require without the patient interfering and changing commands sent to the device.

#### *9.3.3.2 Historical Data*

The doctor, once logged into a patients account, has access to the patient's historical data as shown in the patient's dashboard screen section 9.2.3.1 Data Backup (History) above. This data can be displayed on the app itself or opened in Microsoft Excel or equivalent app for data manipulation.

## 9.4 Non-Authenticated Dashboard

The Non-Authenticated system is very basic. In this, the user can connect to the device, send commands and get data back. No data is saved to the database or backed up to the storage system as the user's identity is not known. Figure 9.9 below shows the non-authenticated dashboard activity.


Figure 9.9 - Smartphone Application - Non-Authenticated Dashboard Screen

### 9.4.1 Functionality

The main function of the non-authenticated screen is to communicate to the device. As the user is not authenticated, data cannot be saved to the database as it is not known who this information has come from.

### 9.4.2 Method

The non-authenticated dashboard has a number of key methods associated with it. They are as follows:

- Getting data from the device
- Communication - Bluetooth

This section discusses these methods.

#### *9.4.2.1 Getting Data from the Device*

Getting data from the device is done the same way as in the patient's dashboard screen, as discussed in section 9.2.2.3 Getting Data from the Device above.

#### *9.4.2.2 Communication - Bluetooth*

Communicating to the device is done the same way as in the patient's dashboard screen, as discussed in section 9.2.2.1 Communication – Bluetooth above.

# 10. Hand-Held Portable Healthcare Device Development and Prototyping

This section documents the development of the physical device. Key design parameters to consider in the device are:

- Physical size of the system
- Ease of operation (user friendly)
- Aesthetics
- Data Accuracy
- Data Collection Speed
- Repeatability

## 10.1 Hardware Selection and Prototype Development

The major hardware components were discussed and selected in Remote Diagnosis System Design section, which were:

- ECG Detection – Ad8232 chip
- Microcontroller - Arduino 101 with built in Bluetooth BLE and an accelerometer
- LiPo Battery
- Temperature sensor – NTC Thermistor

ECG Signal Acquisition and Filtering and Dry Electrode Study has shown that for acquiring ECG signals, a two-lead electrode input is capable of getting an adequate ECG signal to determine the heart rate. This conclusion is adopted in the hand-held sensing device. Other hardware was also added either to make the device easier to use or to improve the functionality. These include:

- Neopixel ring – Circle made up of individually addressable LED's. Used for visual feedback on what the device is actually doing and can therefore indicate if there are communication faults
- SD card reader/writer for backing up data – Not implemented in this design but considered for future development.
- On/Off Switch
- Battery Charging Input Plug

## 10.2 Prototype Design

### 10.2.1 Printed Circuit Board Design for the Hand-Held System

A schematic was drawn up for the system. This board connects the Arduino 101 to peripheral sensors and hardware. Key components to this schematic are:

- Two-lead electrode input into a three-lead electrode ECG processing board
- $4^{th}$ order Butterworth low pass filter
- Thermistor temperature sensor
- Power input
- Battery voltage detection

- Neopixel ring
- Power control switch
- SD card wiring – Not implement in the current version.

The image in Figure 10.1 shows the schematic for the device. The corresponding Printer Circuit Board (PCB) is presented in Figure 10.2. The PCB developed was shaped to match the Arduino 101 shield to allow for easy connections.

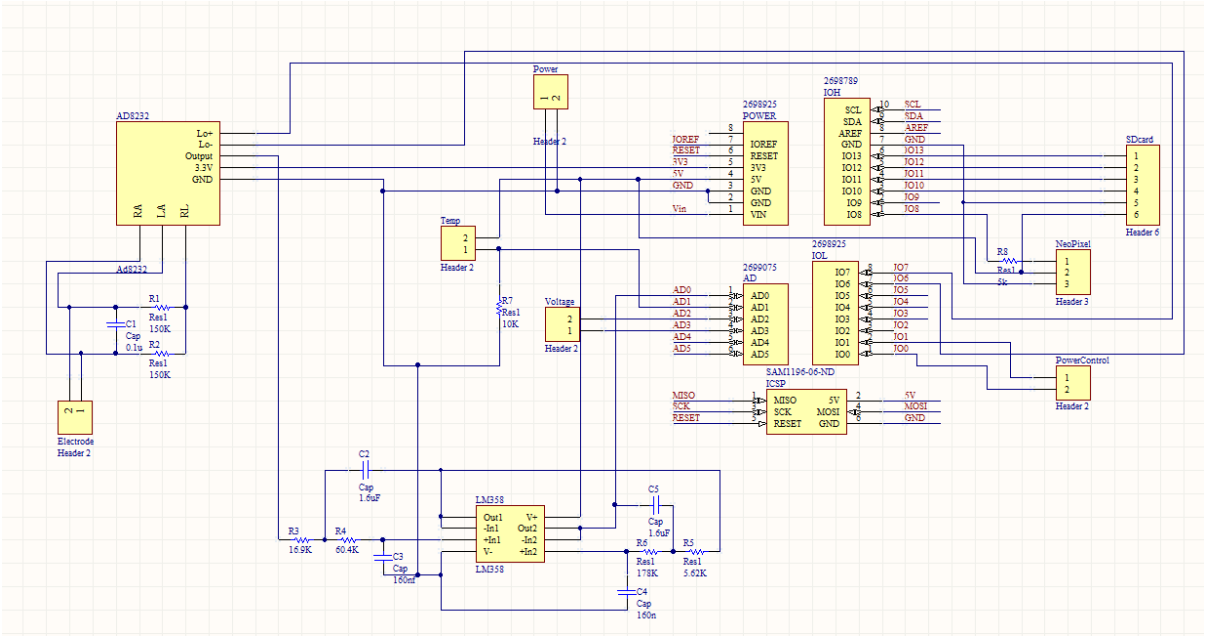



**Figure 10.1 - PCB Board Schematic of the Device**

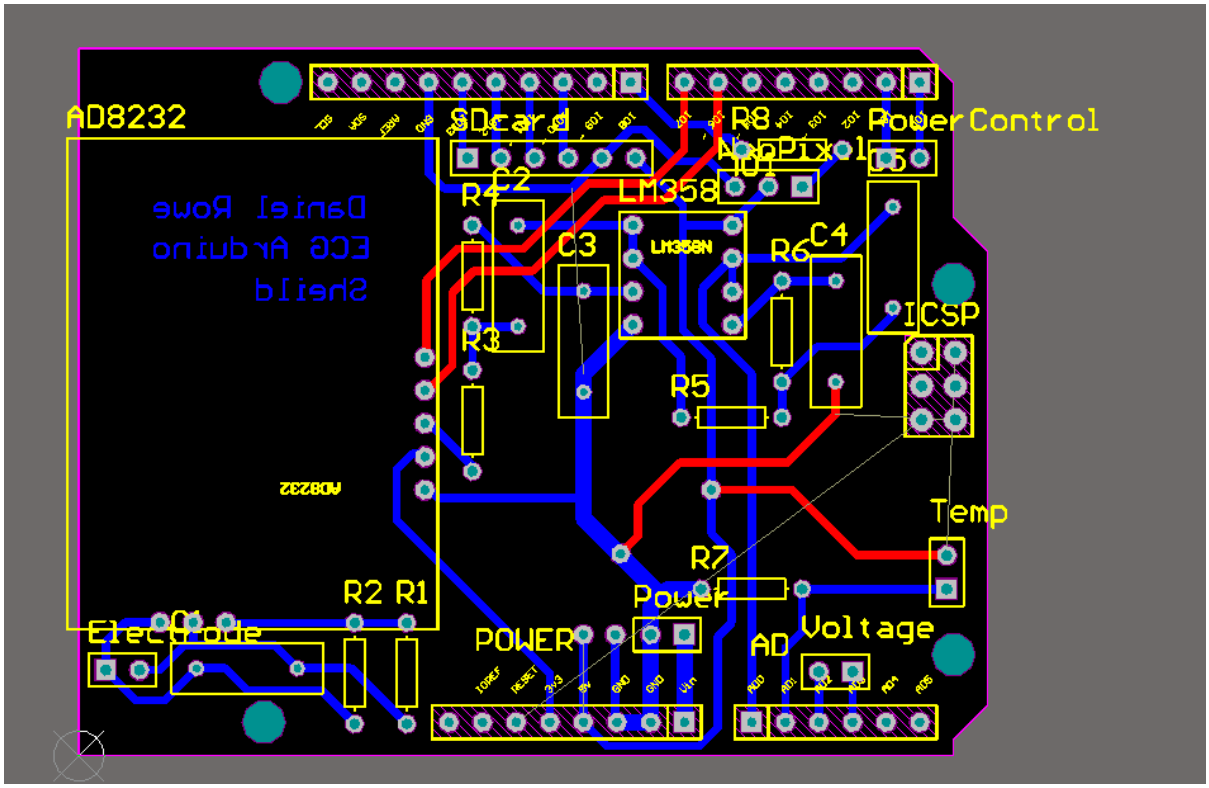



Figure 10.2 - The PCB Board of the System

### 10.2.2 Power Circuit

A power circuit was developed to control the power to the system. The schematic of the system is shown in Figure 10.3. It is designed around having a "momentary maintain" slider switch which is used to turn the system power on and to manually turn the system power off. To achieve this, when the switch (S1 as shown in Figure 10.3) is turned on, it bypasses Q1 allowing current to flow through Q2 which supplies power to the Arduino. The Arduino then latches a digital output on Q1 meaning the switch can be let go and power is maintained. To turn the power off, the Arduino unlatches the digital output controlling Q1 either automatically or at the request of the user using the same switch that turns on the device.

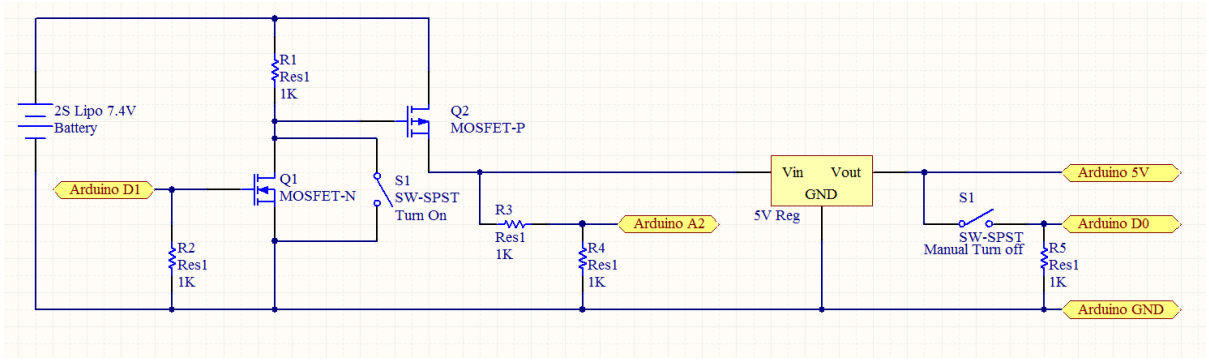



Figure 10.3 - Developed Device Power Circuit Schematic

### 10.2.3 Physical Hand-Held Device Design

The physical shape of the device is very important. This is what the users see and essentially how they will judge the system. It is very important to develop a system that is aesthetically pleasing and comfortable to hold and use.

The key factors governing the size of the design are the spacing and size of the electrodes. These determine the minimum length of the device. At the end of Dry Electrode Study, it was concluded to achieve the best ECG signal, electrodes should have a cross sectional area to 45mm and be 35 mm apart. The other major factor is the height of the electronics. In this system, there is an Arduino 101 with a PCB shield on top of this, which is required to fit inside the device.

Two prototypes were developed which are documented below. The prototypes were manufactured using 3D printers. The base has a foam layer around it to make it softer against the patient's chest.

#### *10.2.3.1 Version 1*

This was the first design. The electronics are located at the front of the device which is why this is higher. Having the electronics at the front allowed the rear of the device to slimmer and slicker, flowing nicely into a handle. The handle has finger marks in the sides suggesting how to hold the device. Figure 10.4 is images of the design.

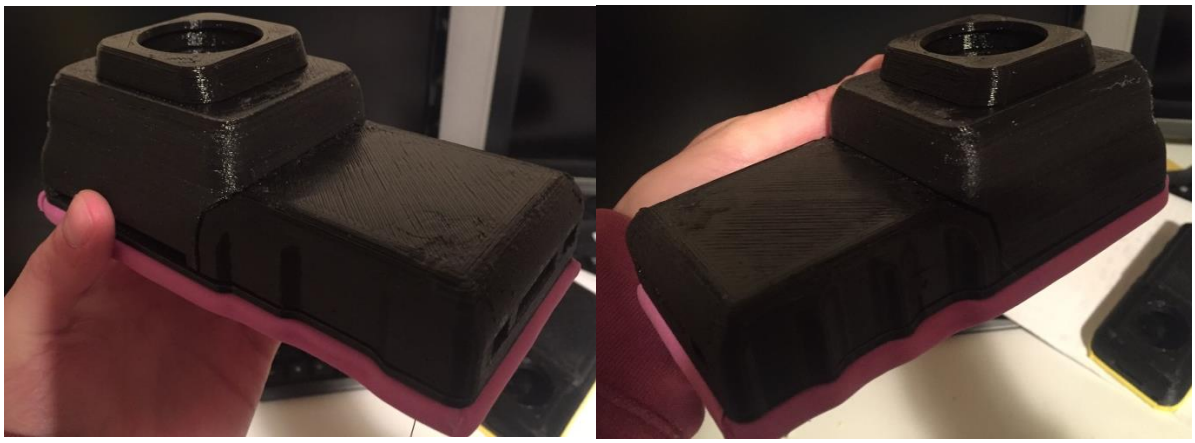


Figure 10.4 - Developed Device - Physical Design Version 1

#### *10.2.3.2 Version 2*

Version 2 was developed later. In this design, the electronics are located at the rear of the device. Again, handle marks where added at the rear to suggest where to hold the device. The Neopixel ring is still located at the front of the device. The difference between the designs is that in version 2 the electronics are spread out meaning the height is the same along the whole device. Figure 10.5 shows the images of Version 2.

Figure 10.5 - Developed Device - Physical Design Version 2

### 10.2.4 Features

This section documents key features in the device. These are as follows:

- Temperature sensor
- Neopixel ring
- Battery charging socket
- Lipo batteries

#### *10.2.4.1 Temperature Sensor*

The physical versions have built in temperature sensors that fold away when not in use, as shown in Figure 10.6.

Figure 10.6 - Device Development Features - Temperature Sensor

The temperature sensor is stored on a slider which can be moved in and out of a recess in the body of the device. The temperature sensor is attached to a wire and is wrapped around the slider. To measure their temperature the user must slide out the sensor and unwind the wire. The wire is 300 mm long allowing the sensor to easily be placed under the patient's armpit for an accurate measurement. Once the temperature is measured, the patient needs to wind it back onto the slider and push it back into the body of the device.

### *10.2.4.2 Neopixel Ring*

The Neopixel ring was added to the design to give the user feedback on the current status of the device. Figure 10.7 presents some images of the Neopixel ring in various statuses.

**Figure 10.7 - Device Development Features - Neopixel Ring**

The Neopixel ring can show a number of different colours. The colours and their meanings are listed below:

- White – First powered on, the bypass switch is still turned on
- Blue – Waiting for Bluetooth connection
- Green – Connected to Bluetooth and waiting for commands
- Yellow – Measuring temperature
- Red – Measure ECG/heart rate
- Red pulse – Heart rate detected
- White pulse x 2 – Turn off command – either from app, manually, time out or due to the battery voltage being low. If it is turned on again and it has two white pulses, it means battery is flat

### *10.2.4.3 Battery Charging Socket*

The battery charging is done through a balance charger. There is a plug located on the device to allow for easy charging of the batteries without the need to open the device. Figure 10.8 shows the charging plug.

**Figure 10.8 - Device Development Features - Battery Charing Socket**

### *10.2.4.4 Lipo Batteries*

Two Lipo cells are used in the system to give a nominal voltage of 7.4V. Each cell has a capacity of 740mAh. The power a battery can supply can be calculated by:

$$Battery\ Energy\ (Wh) = Capacity(Ah)\ x\ Voltage\ (V)$$

**Equation 4**

The battery system should be able to supply 5.476Wh. Using a multimeter, the current supplied to the system was measured as 151mA, shown in Figure 10.9. Using Equation 4, the power consumed by the system is 1.1174W (0.151A x 7.4V). The batteries are capable of suppling just under 5.5Wh so the system should be able to run for just under 5 hours on a single charge which is considered to be more than adequate. As the battery drains, the supply voltage also drops, so the actual total run time could be less than this.

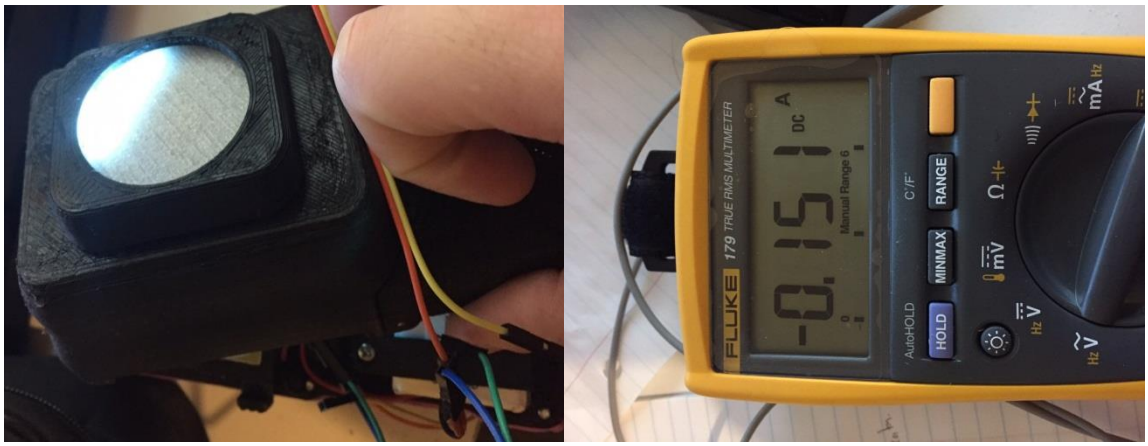



**Figure 10.9 - Device Development Features - Lipo Battery Run Time Measurement**

# 11. Final System, Discussion and Conclusion

## 11.1 Final System

The final outcome of this research consists of:

- A hand-held device that is capable of acquiring the patient's ECG signal for measuring the patient's heart rate and body temperature.
- A smartphone application for doctors or patients to control the device and save information to a cloud database and storage
- A cloud database and storage to save historical data and to transfer information in real-time from remote patients to doctors.

The final system produced by this research is shown in Figure 11.1. It allows a doctor to:

- Establish a wireless connection with a remote patient
- Measure the patient's heart rate and temperature
- Obtain the patient's ECG and temperature data in effectively near real-time
- Store patient's health data in cloud

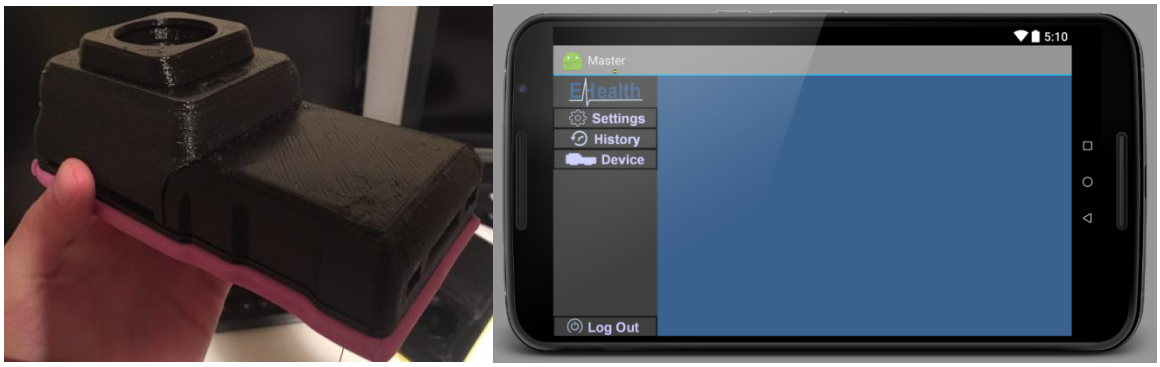



Figure 11.1 - Final Solution

### 11.1.1 System Architecture

The system architecture is presented in Figure 11.2 and Figure 11.3. The diagram shown in Figure 11.2 shows how the key components in the system interact with each other.

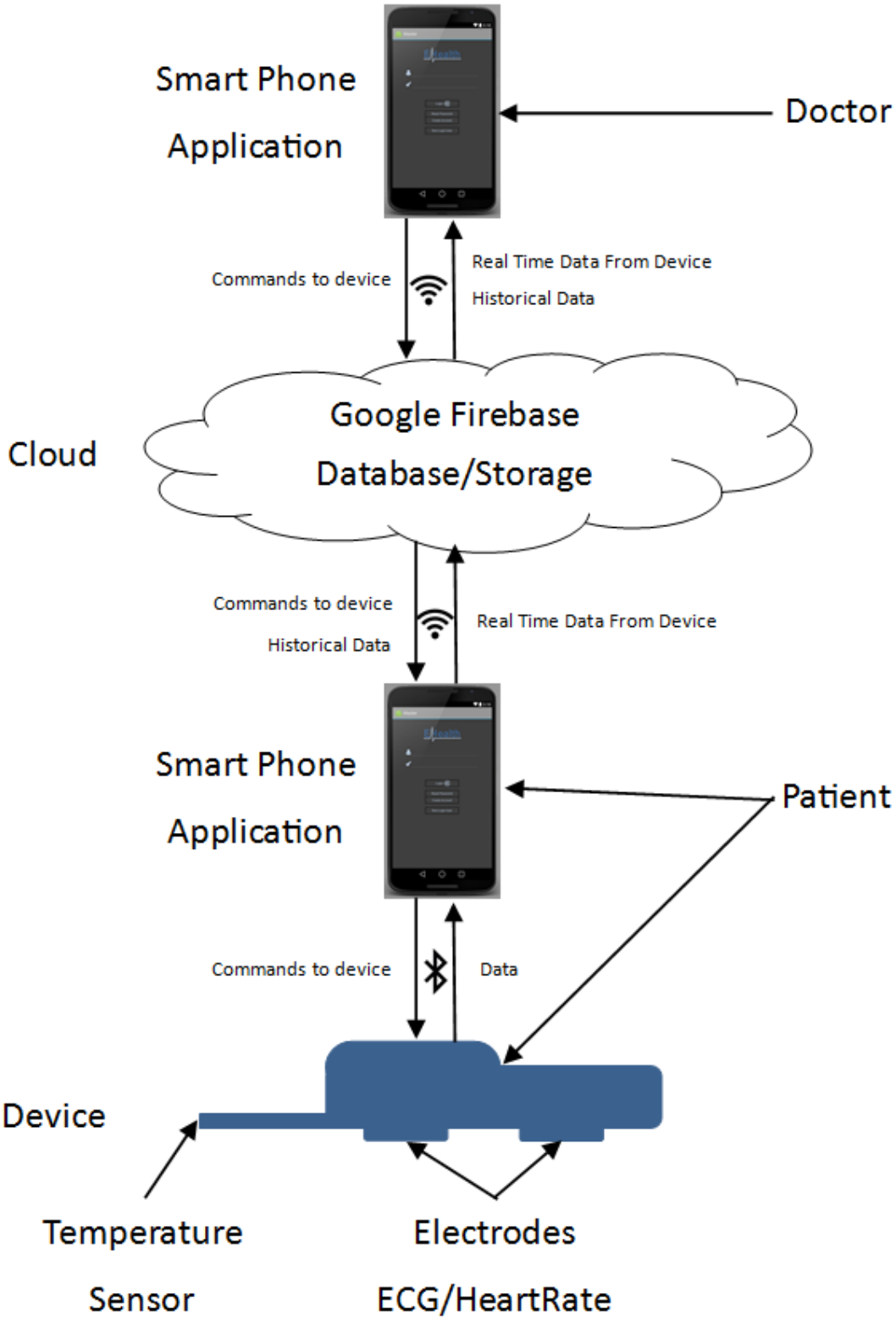



Figure 11.2 – Final Solution Flow Chart

The system can handle multiple users at the same time. The image shown in Figure 11.3 illustrates how multiple users and doctors could use the system.

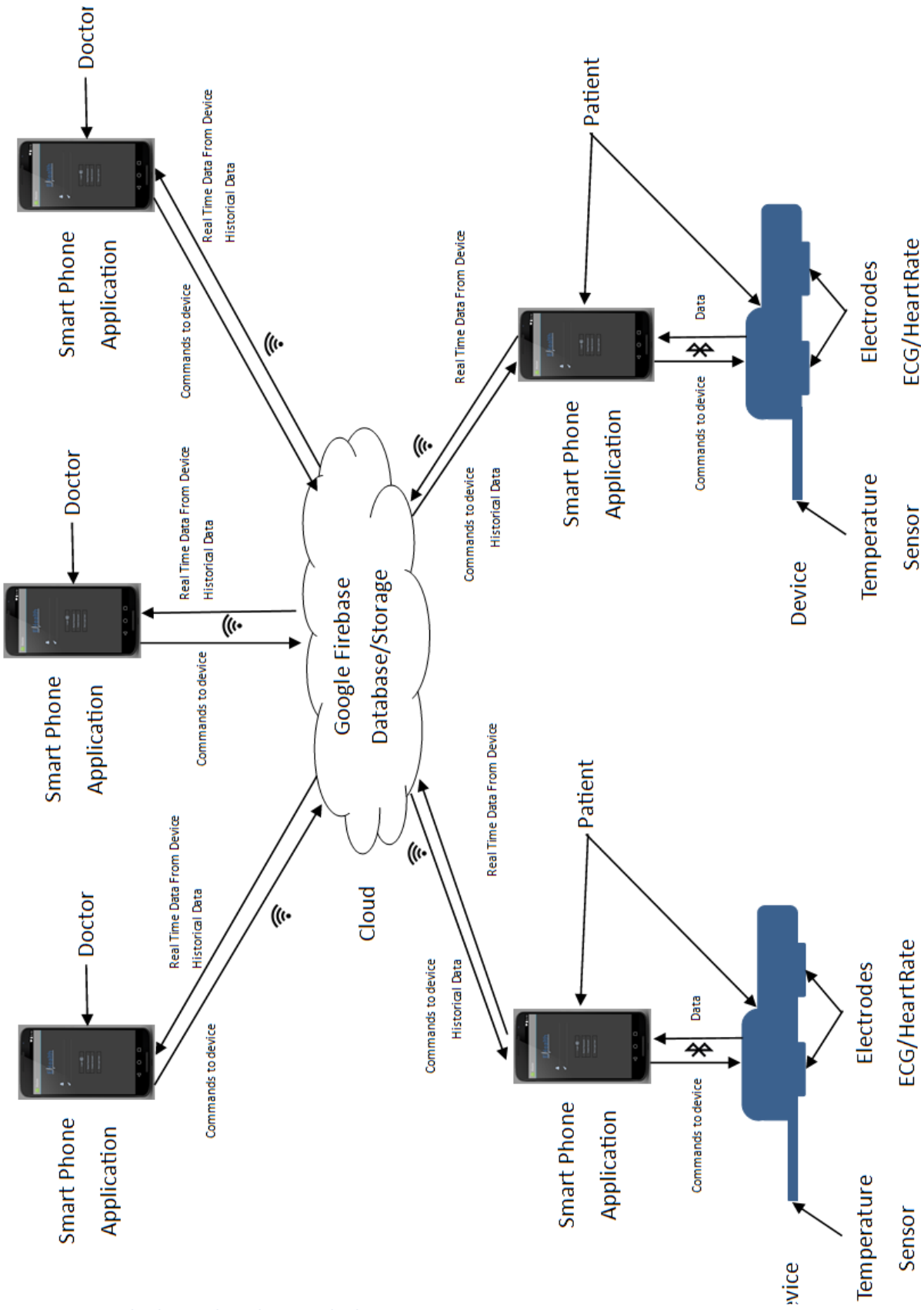



Figure 11.3 - Final Solution Flow Chart - Multiple Uses

### 11.1.2 General Operation

Patient's Side:

1. The patient turns on the device and opens the smartphone application and logs in. The login method is handled by Firebase authentication function and it requires internet access.
2. Once logged in, the patient then connects to the device over Bluetooth and when connected the patient can send commands to the device. Four commands can be sent to the device:
    a. Measure temperature
    b. Measure heart rate and ECG
    c. Stop measuring
    d. Turn off device
3. The device then does as it is commanded to. If user wishes to measure the heart rate, they place the device on their chest, over their heart, so that the electrodes are touching the skin. In the application the patient then presses the measure heart rate button to start measuring the heart rate.
4. The device picks up the ECG signal using the electrodes and from it measures the heart rate and sends the data back to the app.
5. The app then displays the data, processes it in to a Microsoft Excel file and writes the data to the database over the internet.
6. Once the user has finished measuring the heart rate they press the stop command, the app then saves the Microsoft Excel file, and backs this up using the Firebase storage system over the internet.

Doctor's Side

1. The doctor uses a similar process. They open the application and login using the Firebase authentication method over internet.
2. If the user is listed as a doctor, it takes the user to the doctor screen in the app.
3. Here the doctor must first connect to the patient. This is done by entering in an email account for the patient they wish to monitor.
4. Once connected to the patient who is logged in and connected to the device then the doctor can send commands directly to the device.
5. The commands are written to the database which the patient's app can see, process and send them to the device.
6. The device then runs the command and sends the data back to the patient's app, which then displays the data, processes it in a Microsoft Excel file and writes the data to the database over the internet.
7. The doctor's app then sees this data in the database and updates the display on the application in near real-time.
8. Once the doctor has finished measuring the vital sign, by pressing the stop command, the patient's app saves the Microsoft Excel file containing the data and saves this to the Firebase storage system over the internet.

### 11.1.3 Cost

One of the original goals was to develop a cost-effective healthcare device. Through the entire development process, the cost has been one measure for selecting the elements to use. This effort made the cost for the final system within the desired cost range and became an economic choice.

## 11.2 Verification

Verification is required to ensure information gathered is correct. In the medical industry accuracy of the information is critical in providing the correct healthcare to patients. The device developed measures two key vital signs:

- Heart Rate
- Temperature

The heart rate is calculated internally within the device based on the ECG signal captured from the two electrodes placed on the patient's chest. The heart rate and temperature can be verified using the following methods:

- Heart rate can be checked by counting the number of pulses over a period of time. This can be done by feeling the patient's pulse and counting the number of pulses over a minute, which gives a heart rate (BPM).
- The temperature can be verified by using a thermometer which can be brought from a pharmacy to record the temperature of a patient.

Both these methods were used to verify the data captured from the device. The device uses the ECG signal to calculate the heart rate which is sent through the Smartphone App to the healthcare professional. Therefore, it is also required to verify the ECG signal used within the system. Validating the ECG signal is a little more complicated as this needs expensive medical grade equipment.

As mentioned in the literature review, section 2.3 Remote Healthcare Devices; many medical commercial products are available on the market which measure and record healthcare data. A method of validating the ECG signal is to compare the ECG signals obtained by the device developed through this research with ones captured using commercially available medical equipment. It is assumed that commercial products that claim to record medical grade ECG signals have themselves been tested and verified to record accurate ECG data.

The commercial product selected to conduct the verification is manufactured by a company called Shimmer technology. Shimmer has been developing and manufacturing medical-grade wearable sensors for over ten years. Shimmer states they partner with scientists to undergo ground breaking research and bring their innovative sensing devices to the market. A "Consensys ECG Development Kit" was used to verify the ECG signal. Figure 11.4, below shows the Consensys ECG system.


Figure 11.4 - Consensys ECG System [72]

The Kit contains a base and modules (“Shimmers”). Electrodes are connected to the Shimmer. The Shimmer uses a five-lead electrode configuration. ECG signals are recorded by the Shimmer itself using a start/stop button on the front. Plugging the Shimmer into the base allows ECG data to be exported into Microsoft Excel. This kit cost €1436 which is approximately $2350 (NZD).

The verification process focused on two key stages in the device developed by this research :

1. The raw ECG data captured by the prototype device.
2. The ECG data transferred to the App.

In conducting the test, it is vital to attain accurate readings between the Shimmer system and the developed device. To achieve this, a number of controls were introduced, as listed below, followed by the implementation.

1. Correct electrode placement.
2. Record information from the two devices at the same time
3. Reduce errors created by humans by reducing human inputs.

To achieve the final two points above, the following method was used. The Shimmer device is started and stopped using a button. This cannot be avoided. When checking the raw ECG data, the code in the developed device was modified to automatically record 10 seconds of data. This means that only the Shimmer needs to be started and stopped at the correct time

The electrodes were positioned using the placement suggested by Shimmer. Figure 11.5 below, shows the suggested electrode placement. Electrodes where placed at RA, LA, RL and LL as well as one placed at one of the locations between V1-V6. The blue squares show the placement of the developed device electrodes.

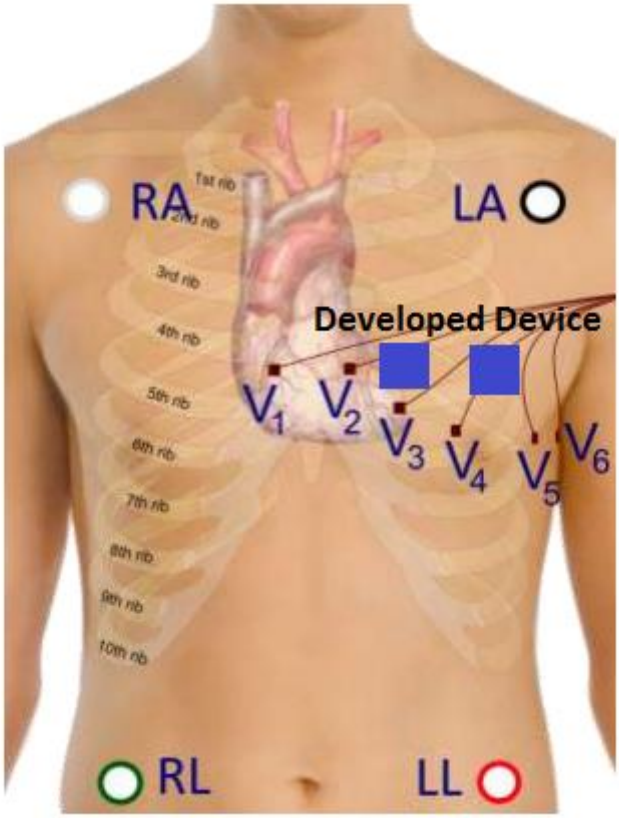



Figure 11.5 - Shimmers Suggested Electrode Placement with the Developed Devices Electrode Placement

Figure 11.6, below shows the two systems placed on a chest for conducting the tests.

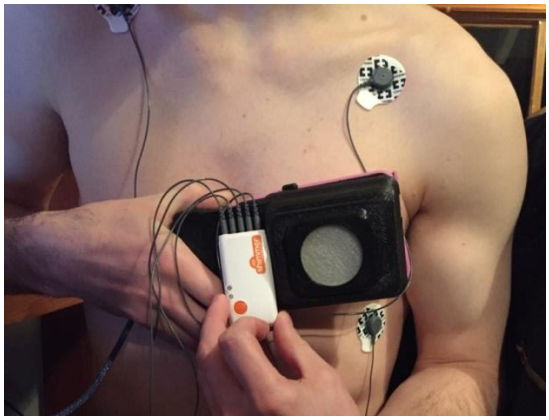


Figure 11.6 - Electrode Placement of the Two Systems

### 11.2.1 Results:

In conducting the verification, a few issues were found as follows.

- Shimmer records data in mV typically between 0-30mV. The developed device records data between 0-5V. To compare the two signals on one graph, one of the signals needs to be scaled.
- Human factor. The starting/stopping of the Shimmer device introduced shifts in the data which makes it difficult to compare the two devices.

- Shimmer records data at 19.53ms intervals. The developed device records data every 5.21ms which again makes it difficult to plot the two signals on the same graph.

The results are shown in the graphs below. In these graphs, the scale of the Shimmer system has been changed. The first graph shows the raw data.

The second, shows the normalised data, where the scale on the X-Axis of the Shimmer data has been adjusted to show 10 seconds of data (the same as what is automatically produced by the developed device). The developed device creates 1918 data points over 10 seconds (new data every 5.21 ms). Data from the Shimmer is outputted every 19.53ms. Therefore, to plot the two data series on the same graph with the same x- axis scale, the Shimmer data needs to be adjusted by 3.74 ms (19.53/5.21).

Three samples were taken and the raw data and normalised data for these samples are shown in Figure 11.7 - Figure 11.12.

Sample 1:

Raw Data

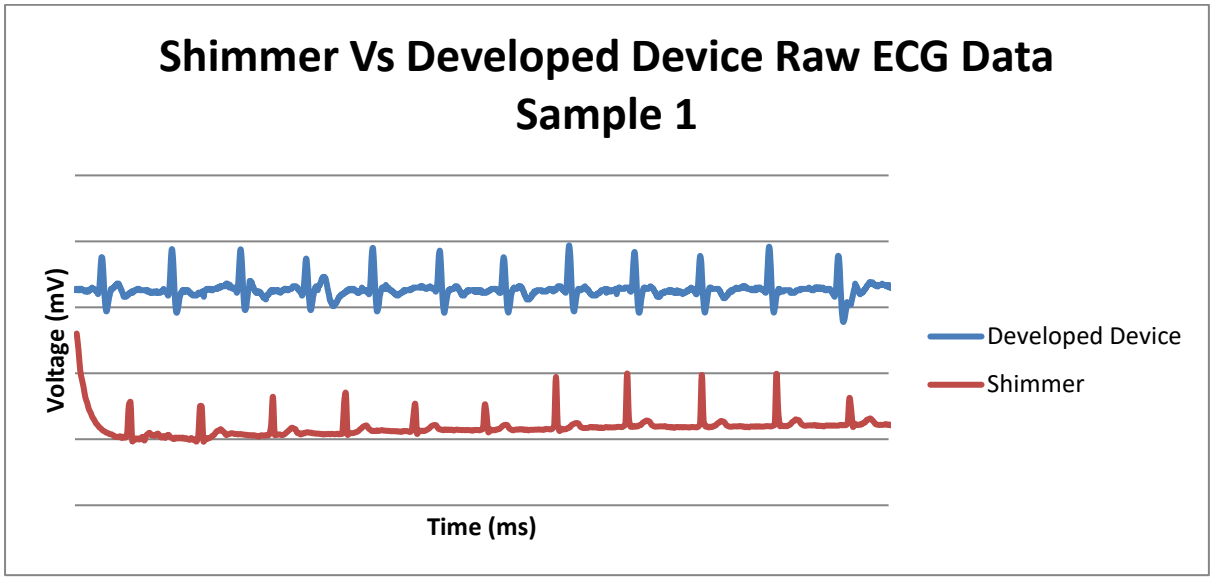



Figure 11.7 - Sample 1 - Shimmer vs Developed Device Raw ECG Data

Normalized:

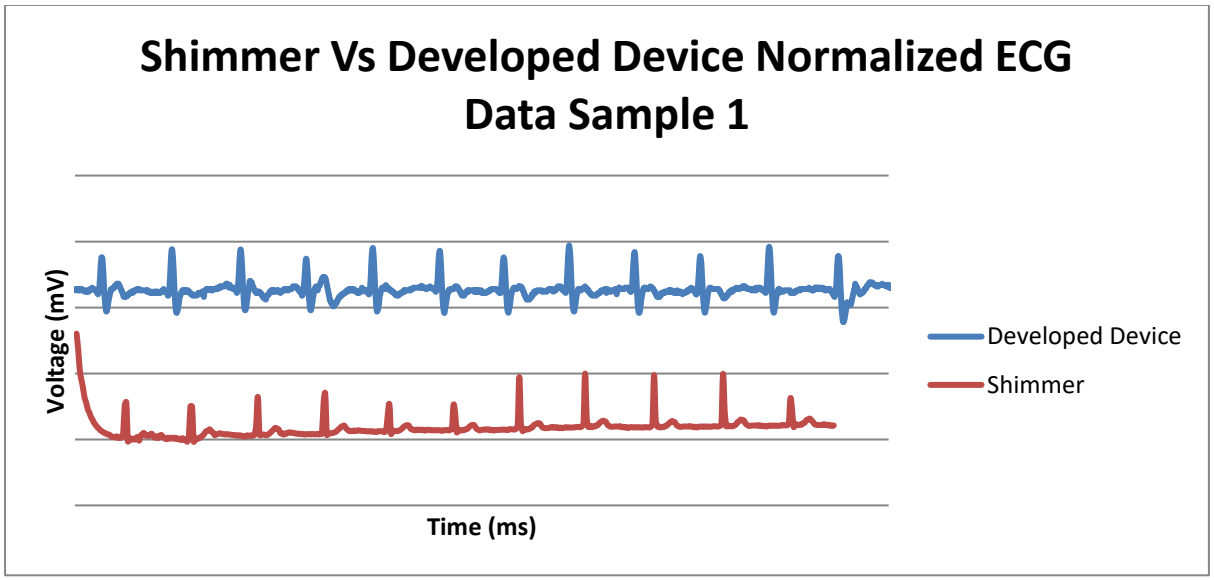



Figure 11.8 - Sample 1 - Shimmer vs Developed Device Normalized ECG Data

Sample 2:

Raw Data:

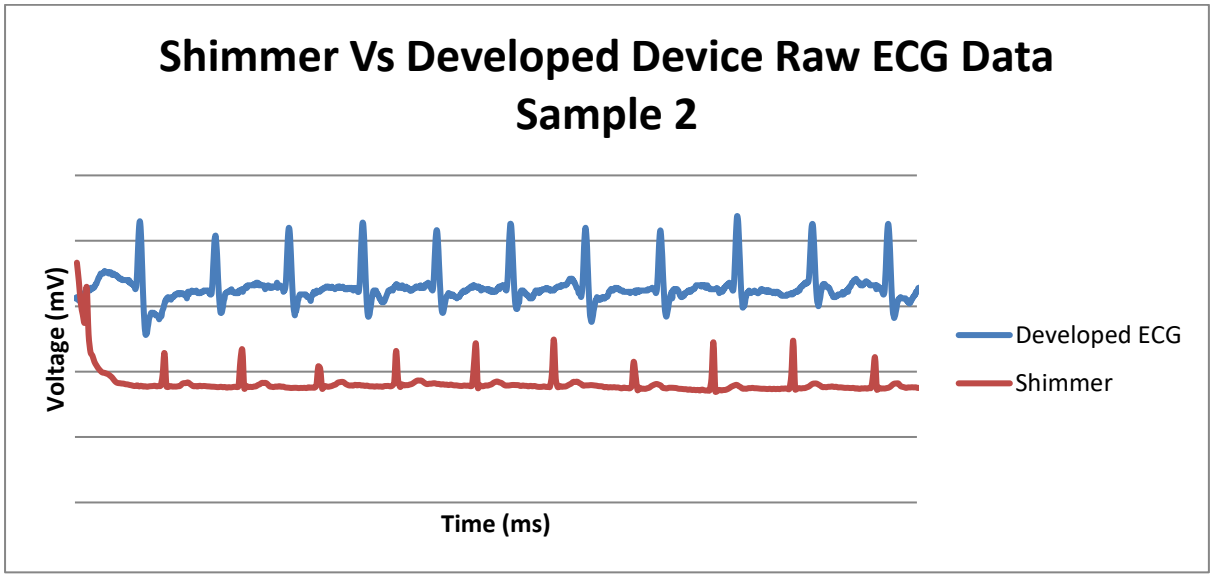



Figure 11.9 - Sample 2 - Shimmer vs Developed Device Raw ECG Data

Normalized:

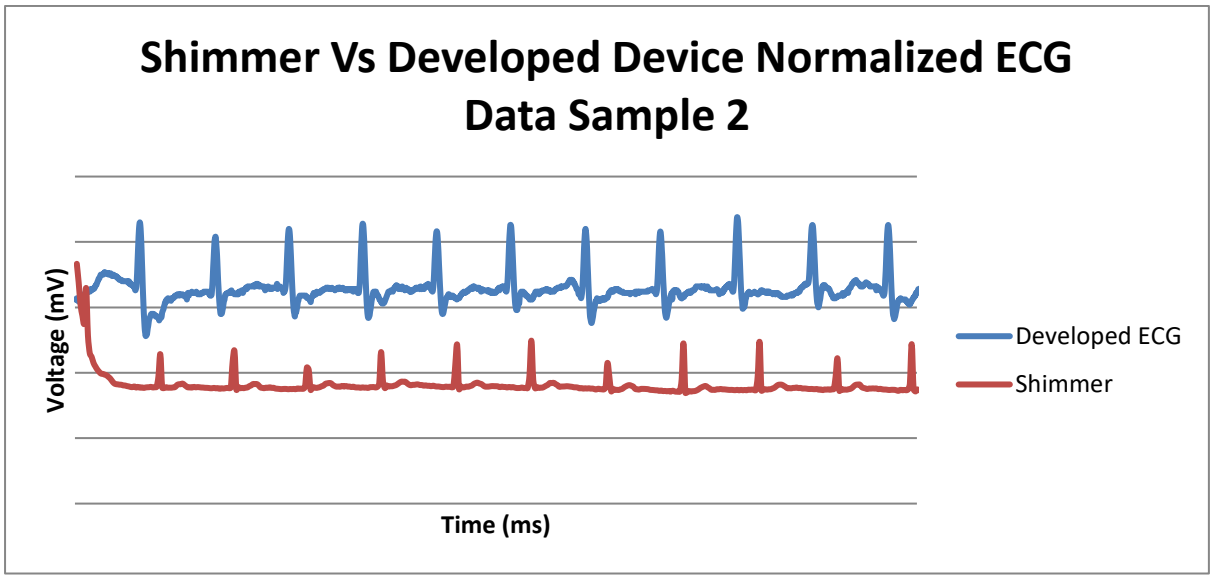



Figure 11.10 - Sample 2 - Shimmer vs Developed Device Normalized ECG Data

Sample 3:

Raw Data:

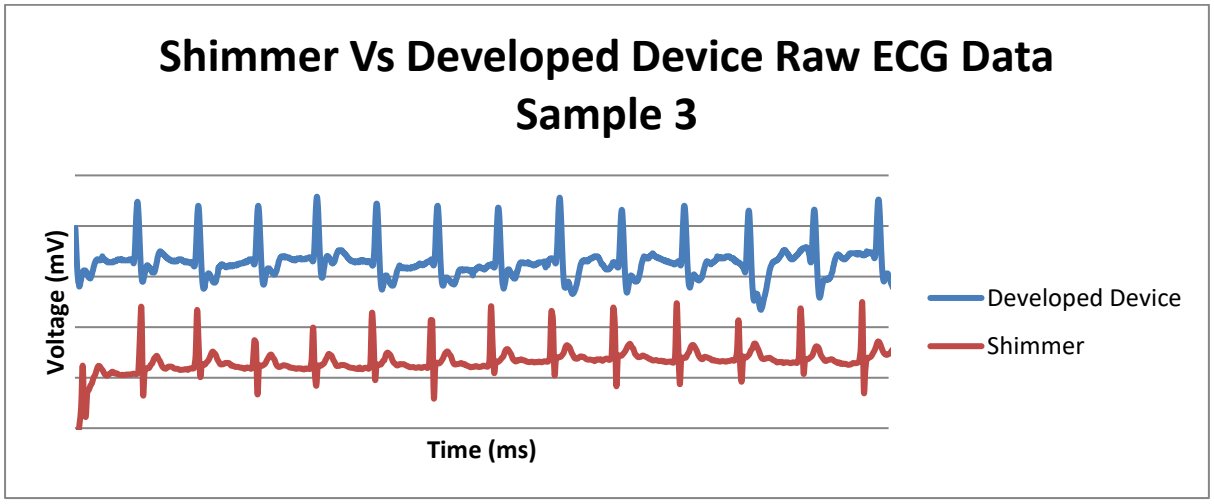



Figure 11.11 - Sample 3 - Shimmer vs Developed Device Raw ECG Data

Normalized:

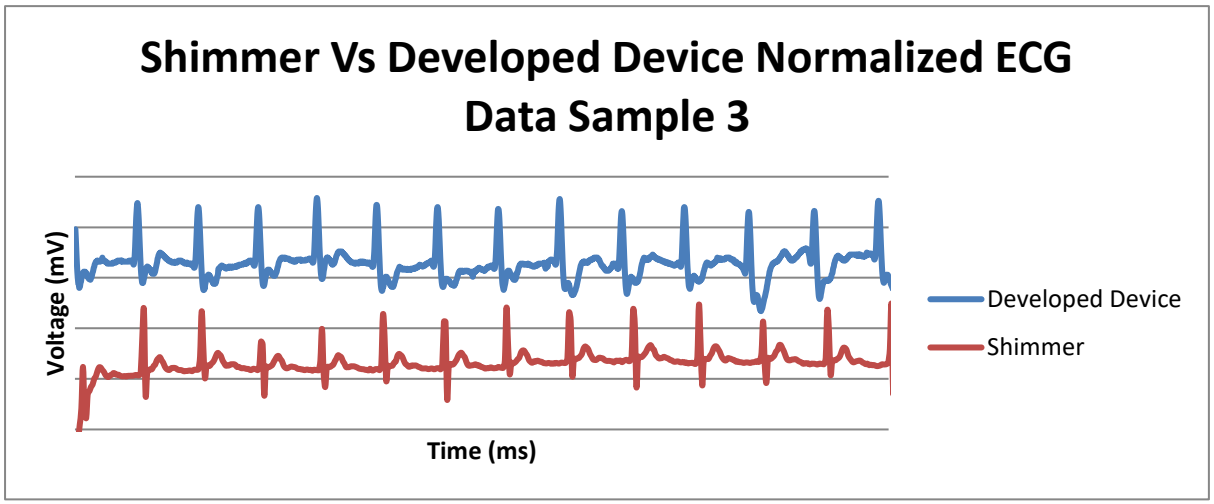



Figure 11.12 - Sample 3 - Shimmer vs Developed Device Normalized ECG Data

Once the data had been normalised (the x-axis has the same scale), there is a better correlation. It can be observed that the pulse (R-wave) occurs at the same frequency. The waves are different. This is due to the fact that the two systems are measuring different parts of the same ECG signal (ECG view). The developed device uses two electrodes placed over the patient's heart, where the Shimmer uses five electrodes placed all over the body to create one signal. This results in different waves. For the purpose of this project, it is deemed that the three samples shown in the above figures validate the raw ECG signal.

The second location where the ECG data was to be verified was between the data saved by the App and Shimmer. Figure 11.13, shows ECG data collected at the App and by the Shimmer device.

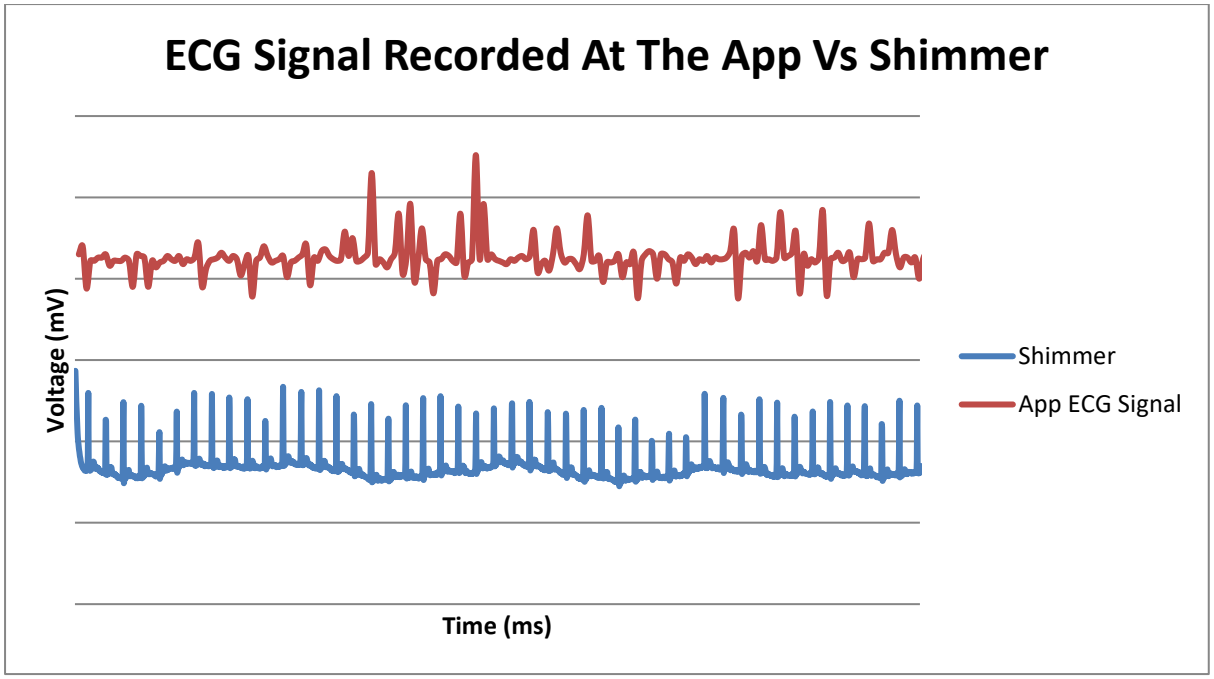



Figure 11.13 - ECG Signal Recorded at the App vs Shimmer

As expected the ECG signal observed at the application doesn't match the true ECG signal measured by the Shimmer device. This is due to the time it takes to process and send data to the App via wireless communication. To overcome this problem, further research is needed on wireless communication between the App and the device. This could include further investigation on communication speed, data format and buffer size to obtain the most reliable data at the fastest possible speed.

### 11.2.2 Conclusions:

1. The Raw ECG data measured by the device occurs the same frequency compared with the commercial equipment was acceptable.
2. The R waves "sharp peaks" in the ECG graphs line up correctly, which validates the heart rate as this is calculated by monitoring the ECG signal, looking for the R waves.
3. The Raw data coming into the device is valid however, the ECG data sent over Bluetooth to the App occasionally misses information. This can be improved by implement a buffer or alternatively further research on wireless data transfer methodologies.
4. It is also difficult to get the two systems recording at exactly the same time due to the human factor in starting and stopping the Shimmer recording. This results in the data being offset a bit from each other.

In Conclusion the raw ECG data captured by the developed device is valid. To transfer the data wirelessly to an App further research into other methodologies is required.

## 11.3 Future Work

The device presented is the first prototype of the proposed system. The research and building of the prototype revealed there are many more features that could be added, and further improvements could be made such as:

- Redoing the two-electrode circuit using the AD8232 Chip as per the schematic supplied with the AD8232 datasheet for wiring a two-lead ECG. The AD8232 break out board used in this device did not allow the chip to be wired in the suggested way for a two-lead ECG. This may improve the final ECG signal strength
- Bluetooth communication between the device and the application. Bluetooth allows a interface to be established to the device using a smartphone application. From earlier work using Bluetooth 2.0 in a similar application, Bluetooth BLE provides huge improvements for example:
    - Improved data transfer rates
    - Bluetooth characteristics
    - Low energy consumption

  Although Bluetooth BLE can send data at 25Mb/s, Bluetooth 4.0 could be an improvement as it has an even greater data transfer rate.
- Adding more sensors to give more flexibility e.g.
    - Blood pressure
    - Blood sugar
- Carry out extensive testing of the vital signs measured from this hand-held device vs a number of hospital grade equipment. It also needs to be checked to see how repeatable the results are and to determine the reliability of the code. Another important test is to check how reliable the system is when measuring vital signs on different patients.
- To implement an SD card storage as a backup of the cloud storage. This could be used to add additional functionality to the device which could include continuous monitoring of patients as opposed to diagnosis.
- At this stage, the system can take some time to detect the patient's heart rate from the ECG signal. This is to do with the code and how it tunes in to the ECG signal to calculate the heart rate. Ideally this process would be speed up.
- Test the speed of the Firebase system once there is more users and traffic on the database. It is possible that, when there are more users and information needed to be stored on the database, the upload/download time could be increased, and the transfer of information could slow down. This needs to be tested as it is one of the key components in the system

## 11.4 Discussion and Conclusion

The initial research hypothesis was to investigate a feasible and effective remote diagnosis system for healthcare. The research outcome confirms this hypothesis. The physical prototype realised the functionality and features set in the hypothesis.

The research followed the aim and objectives set in section 1.3 Aim of the Research and Objectives, and considered the functionalities, potential users, device size and cost, and ease of use. The developed system has the potential to be a useful medical tool going forward, especially if it can be integrated into a pre-developed medical consultation smartphone application like Maven or Doctors on demand. A large number of journal articles state that telemedicine is the future of healthcare. A system like this has great potential for future healthcare service.

The research clearly shows that patient's vital signs and other medical information can be sent from remote locations to a healthcare professional in almost real-time using the internet and a cloud database. It could provide a near real-time link to existing apps. The final prototype, which is not a completed product ready to be sold to the medical industry, but it is a proof of concept, showing medical diagnosis could be conducted in this manor at a low cost. In short, the research outlines new systems and methods for remote healthcare diagnostics.

The ECG signal acquired from the developed device is very clean which is a useful diagnosis tool for doctors. Using a two-lead ECG system has allowed for the device to be small and hand-held. The developed low pass filter with the two-lead ECG system and soft electrodes has resulted in a noise free and well-defined ECG signal. However, the ECG signal could be even more well-defined which would make the heart rate detection more accurate and faster to detect. This would also be more useful to healthcare professional monitoring and diagnosing heart conditions where ECG signal acquisition is required.

The Google Firebase handles the traffic well and is one of the triumphs of this project. This forms the back bone of the system and provides fast data transfer speed. After the initial testing of the system, the Firebase database and storage shows no sign of slowing down. It also keeps data secure as users need to be authenticated to access the information.

The temperature sensor has a very quick response time to a change in temperature and is accurate. Mounting the sensor so it can be taken out of the device makes it more user friendly.

# Bibliography


[1] Oxford Dictionaries, "English Oxford living Dictionaries," [Online]. Available: https://en.oxforddictionaries.com/definition/telemedicine. [Accessed 10 12 2016].

[2] N. M. Lacktman, "Five Telemedicine Trends Transforming Health Care in 2016," *Journal of Health Care Compliance - January-February 2016,* pp. 43-44,58, 2016.

[3] G. Jeff and O. Matt, "Telemedicine, care models for the 21st century," *Vynamic Insight,* 2016.

[4] B. A. Levine and D. Goldschlag, "Can telemedicine boost our ailing healthcare system?," *Contemporary OB/GYN (CONTEMP OB GYN), Jul2015,* pp. 36-39, 2015.

[5] M. Andrew J. Potter, P. Marcia M. Ward, M. Nabil Natafgi, F. Ullrich, M. C. A., B. L. Amanda and M. J. Keith, "Perceptions of the Benefits of Telemedicine in Rural Communities," *Perspectives in Health Information Management, Summer2016,* pp. 1-13, 2016.

[6] A. Banbury, A. Roots and S. Nancarrow, "Rapid review of applications of e-health and remote monitoring for rural residents.," *Australian Journal of rural health Oct2014 22(5),* pp. 21-22, 2014.

[7] N. Van Den Berg, M. Schumann, K. Karft and W. Hoffmann, "Telemedicine and telecare for older patients - A systematic review," *Maturitas Volume 72, Issue 2, October,* pp. 94-114, 2012.

[8] A. Martinez, E. Everss, J. Rojo-Alvarez, D. Figal and A. Garcia-Alberola, "A systematic review of the literature on home monitoring for patients with heart failure.," *J Telemed Telecare 2006, 12(5),* pp. 234-41, 2006.

[9] D. Heaney, J. Caldow, C. McClusky, G. King, K. Webster, F. Mair and J. Ferguson, "The introduction of a new consulting technology into the National Health Service (NHS) for Scotland," *Telemed J E Health. 2009 Jul-Aug;15(6),* pp. 546-51, 2009.

[10] D. Danbjorh, L. Wagner, B. Kristensen and J. Clemensen, "Intervention among new parents followed up by an interview study exploring their experiences of telemedicine after early postnatal discharge," *Midwifery 31 (2015),* pp. 574-581, 2015.

[11] B. E. Holtz, C. Lauckner and P. Whitten, "Mobile health: The way forward in providing innovative health care solutions," *Telemedicine: Emerging Technologys, Applications and Impack on Health Care outcomes,* 2015.

[12] Maven Clinic Co, "Maven Clinic Aboust," [Online]. Available: https://www.mavenclinic.com/about. [Accessed 22 01 2017].

[13] K. Hung, Y. Zhang and B. Tai, "Wearable Medical Devices for Tele-Home Healthcare," *Conference Proceedings: ... Annual International Conference Of The IEEE Engineering In Medicine And Biology*

*Society. IEEE Engineering In Medicine And Biology Society. Annual Conference [Conf Proc IEEE Eng Med Biol Soc] 2004; Vol. 3,* pp. 5384-5387, 2004.

[14] R. Fensli, E. Gunnarson and O. Hejlesen, "A wireless ECG system for continuous event recording and communication to a clinical alarm station," *Conference Proceedings: ... Annual International Conference Of The IEEE Engineering In Medicine And Biology Society. IEEE Engineering In Medicine And Biology Society. Annual Conference [Conf Proc IEEE Eng Med Biol Soc] 2004; Vol. 3,* pp. 2208-11, 2004.

[15] M. Donati, A. Benini, A. Celli, F. Lacopetti and F. Fanucci, "A novel device for self-acquisition of ECG signal in telemedicine systems for chronic patients," *In: Proceedings - IEEE Symposium on Computers and Communications, 2016 IEEE Symposium on Computers and Communication, ISCC 2016. (Proceedings - IEEE Symposium on Computers and Communications, 15 August 2016,* pp. 202-207, 2016.

[16] Medweb, "Medweb Hand-held Telemedicine Kit," [Online]. Available: http://www.medweb.com/docs/MedwebDEK.pdf. [Accessed 14 12 2016].

[17] Globalmed, "Clinical Access Station," [Online]. Available: https://www.globalmed.com/products/telemedicine-carts/clinical-access-station. [Accessed 14 12 2016].

[18] Globalmed, "Clinical Access Station," [Online]. Available: https://www.globalmed.com/products/telemedicine-carts/clinical-access-station. [Accessed 14 12 2016].

[19] AMD, "Portable TeleClinic," [Online]. Available: http://www.amdtelemedicine.com/telemedicine-equipment/portable-teleclinic.html. [Accessed 14 12 2016].

[20] AMD, "Clinical Assist," [Online]. Available: http://www.amdtelemedicine.com/telemedicine-equipment/clinical-assist-system.html. [Accessed 14 12 2016].

[21] "Understanding the ECG: Reading the waves," *Harvard Health Letter,* vol. 36(4), pp. 5-6, 02 2011.

[22] Y. Liao, R.-X. Na and D. Rayside, "Accurate ECG R-peak detection for telemedicine," *2014 IEEE Canada International Humanitarian Technology Conference (IHTC),* 2014.

[23] R. Fensli, E. Gunnarson and T. Gundersen, "A wearable ECG-recording system for continuous arrhythmia monitoring in a wireless tele-home-care situation," *Proceedings - IEEE Symposium on Computer-Based Medical Systems, Proceedings - 18th IEEE Symposium on Computer-Based Medical Systems. (Proceedings - IEEE Symposium on Computer-Based Medical Systems, 2005,* pp. 407-412, 2005.

[24] Buzzle, “Bluetooth Technology: A Summary of its Advantages and Disadvantages,” [Online]. Available: http://www.buzzle.com/articles/advantages-and-disadvantages-of-bluetooth-technology.html. [Accessed 22 01 2017].

[25] eTutorials, “Understanding RD Signals,” [Online]. Available: http://etutorials.org/Networking/wn/Chapter+3.+Radio+Frequency+and+Light+Signal+Fundamentals+The+Invisible+Medium/Understanding+RF+Signals/. [Accessed 22 01 2017].

[26] A. Schriber, “Pros and Cons of 3D Internet Technology,” [Online]. Available: http://internet-access-guide.com/pros-and-cons-of-3g-internet-technology/. [Accessed 22 01 2017].

[27] Lifewire, “4G Mobile Network: The Pros and the Cons,” [Online]. Available: https://www.lifewire.com/3g-vs-4g-mobile-networks-the-health-factor-2373258. [Accessed 22 01 2017].

[28] American EHR, “The Pros and Cons of Wireless and Local Networks,” 17 08 2011. [Online]. Available: http://www.americanehr.com/blog/2011/08/the-pros-and-cons-of-wireless-and-local-networks/. [Accessed 22 01 2017].

[29] B. Wood, “The Advantages and Disadvantages of WANs,” 25 08 2016. [Online]. Available: http://purple.ai/advantages-disadvantages-wans/. [Accessed 22 01 2017].

[30] RF Wireless World, “Advantages of Zigbee | Disadvantages of Zigbee,” [Online]. Available: http://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-zigbee.html. [Accessed 22 01 2017].

[31] H. Soffar, “Wi-Fi Direct uses , advantages and disadvantages,” 25 06 2016. [Online]. Available: http://www.online-sciences.com/technology/wi-fi-direct-uses-advantages-and-disadvantages/. [Accessed 22 01 2017].

[32] Tech Target, “Database,” 01 2017. [Online]. Available: http://searchsqlserver.techtarget.com/definition/database. [Accessed 22 01 2017].

[33] L. P. Issac, “SQL vs NoSQL Database Differences Explained with few Example DB,” 14 01 2014. [Online]. Available: http://www.thegeekstuff.com/2014/01/sql-vs-nosql-db/. [Accessed 22 01 2017].

[34] D. Dobrev, “Two-electrode low supply voltage electrocardiogram signal amplifier,” *Medical & Biological Engineering and Computing 2004, Vol 42,* pp. 272-276, 2004.

[35] R. F. Thaddeus, Fulford-Jones, Gu-Yeon and W. Matt, “A Portable, Low-Power, Wireless Two-Lead EKG System,” *Proceedings of the 26th annual international conference of the IEEE EMBS San Francisco, CA, USA,* pp. 2141-2144, September 1-5 2004.

[36] S. Harden, “Electrocardiography for Cheap: DIY ECG Uses One LM741 Op-Amp, Five Resistors, and ‘Penny Electrodes’,” 31 08 2016. [Online]. Available:

https://blog.adafruit.com/2016/08/31/electrocardiography-for-cheap-diy-ecg-uses-one-lm741-op-amp-five-resistors-and-penny-electrodes/. [Accessed 09 01 2017].

[37] J. Nguyen, "Building an Electrocardiograph," [Online]. Available: https://www.eng.utah.edu/~jnguyen/ecg/instructions.html. [Accessed 09 01 2017].

[38] Scott, "DIY ECG Machine On The Cheap," 14 08 2009. [Online]. Available: http://www.swharden.com/wp/2009-08-14-diy-ecg-machine-on-the-cheap/. [Accessed 09 01 2017].

[39] Elcosh, "Electrical safety: Safety and Health for electrical trades," [Online]. Available: http://www.elcosh.org/document/1624/888/d000543/section2.html. [Accessed 5 12 2016].

[40] Nicegear, "AD8232 Single Lead Heart Rate Monitor," [Online]. Available: https://nicegear.co.nz/sensors/ad8232-single-lead-heart-rate-monitor/. [Accessed 29 11 2016].

[41] Bitalino, "Electrocardiography (ECG) Sensor Data Sheet," [Online]. Available: http://bitalino.com/datasheets/REVOLUTION_ECG_Sensor_Datasheet.pdf. [Accessed 29 11 2016].

[42] Bitalino, "Electrocardiography (ECG) Sensor," [Online]. Available: https://store.plux.info/bitalino-sensors/10-electrocardiography-ecg-sensor-11111111111.html. [Accessed 29 11 2016].

[43] Bitalino, "Bitalino Revolution Board Kit BT," [Online]. Available: https://store.plux.info/kits/33-bitalino-revolution-board-bt-810121001.html. [Accessed 29 11 2016].

[44] Analog Devices, "AD8232 Single - Lead Heart Rate Monitor Analog Front End," [Online]. Available: http://www.analog.com/en/products/application-specific/medical/ecg/ad8232.html#product-overview. [Accessed 29 11 2016].

[45] Mouser Electronics, "Analog Devices AD8232ACPZ-WP," [Online]. Available: http://nz.mouser.com/ProductDetail/Analog-Devices/AD8232ACPZ-WP/?qs=sGAEpiMZZMv9Q1JI0Mo%2ftVQ6dTOYCblL. [Accessed 29 11 2016].

[46] Cooking hacks, "e-Health Sensor Shield V2.0 for Arduino, Raspberry Pi and Intel Gailieo," [Online]. Available: https://www.cooking-hacks.com/ehealth-sensor-shield-biometric-medical-arduino-raspberry-pi. [Accessed 29 11 2016].

[47] Olimex, "Shield-EKG-EMG," [Online]. Available: https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG/. [Accessed 29 11 2016].

[48] WebbMD, "Normal body temperature," [Online]. Available: http://www.webmd.boots.com/a-to-z-guides/normal-body-temperature. [Accessed 07 01 2017].

[49] Electronics Tutorials, "Temperature Sensors," [Online]. Available: http://www.electronics-tutorials.ws/io/io_3.html. [Accessed 08 01 2017].

[50] RS Components, “Honeywell NC 10 A Thermostat, Solder Tag Termination, 0°C +186°C,” [Online]. Available: http://nz.rs-online.com/web/p/thermostats/2295834/. [Accessed 08 01 2017].

[51] RS Components, “AVX 100kΩ 160mW Measurement NTC Thermistor, 8s,” [Online]. Available: http://nz.rs-online.com/web/p/thermistors/6974550/. [Accessed 08 01 2017].

[52] Omega, “RTD Elements: OMEGAFILM® Flat Profile Thin Film Platinum with Ceramic Base and Glass Coating,” [Online]. Available: http://www.omega.com/pptst/F1500_F2000_F4000.html. [Accessed 08 01 2017].

[53] RS Components, “NewRS Pro T Type Thermocouple 1m Cable -75°C → +250°C,” [Online]. Available: http://nz.rs-online.com/web/p/thermocouples/1236328/. [Accessed 08 01 2017].

[54] RS Components, “TE Connectivity Glass Encapsulated NTC Thermistor, 0.5,” [Online]. Available: http://nz.rs-online.com/web/p/products/8937174/. [Accessed 08 01 2017].

[55] Arduino, “Arduino 101,” [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoard101. [Accessed 08 01 2017].

[56] Arduino, “Arduino Due,” [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardDue. [Accessed 08 01 2017].

[57] Arduino, “Arduino Zero,” [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardZero. [Accessed 08 01 2017].

[58] Arduino, “Arduino Nano,” [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardNano. [Accessed 08 01 2017].

[59] Battery University, “BU-1006: Cost of Mobile Power,” [Online]. Available: http://batteryuniversity.com/learn/article/bu_1006_cost_of_mobile_power. [Accessed 08 01 2017].

[60] Battery University, “BU-201: How does the Lead Acid Battery Work?,” [Online]. Available: http://batteryuniversity.com/learn/article/lead_based_batteries. [Accessed 08 01 2017].

[61] Battery University, “BU-203: Nickel-based Batteries,” [Online]. Available: http://batteryuniversity.com/learn/article/nickel_based_batteries. [Accessed 08 01 2017].

[62] Battery University, “BU-205: Types of Lithium-ion,” [Online]. Available: http://batteryuniversity.com/learn/article/types_of_lithium_ion. [Accessed 08 01 2017].

[63] Hobby King, “ZIPPY 740mAh 20C Single Cell,” [Online]. Available: https://hobbyking.com/en_us/zippy-740mah-20c-single-cell.html. [Accessed 08 01 2017].

[64] SparkFun, “AD8232 Heart Rate Monitor Hookup Guide,” [Online]. Available: https://learn.sparkfun.com/tutorials/ad8232-heart-rate-monitor-hookup-guide. [Accessed 09 01 2017].

[65] Analog Devices, “Single-Lead, Heart Rate Monitor Front End,” [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/AD8232.pdf. [Accessed 09 01 2017].

[66] SparkFun, “AD8232_Heart_Rate_Monitor_v10.sch,” [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Biometric/AD8232_Heart_Rate_Monitor_v10.pdf. [Accessed 09 01 2017].

[67] D. H. Spodick, S. Goyal, L. Chhabra, N. Goel and L. Prajapat, “Mouse Heart Rate in a Human: Diagnostic Mystery of an Extreme Tachyarrhythmia,” *Indian Pacing Electrophysiol J. 2012 Jan-Feb; 12,* pp. 32-35, 2012.

[68] N. Storey, Electronics A Systems Approach Fourth Edition, Essex: Pearson, 2009.

[69] Analog Devices, “Analog Filter Wizard,” [Online]. Available: http://www.analog.com/designtools/en/filterwizard/. [Accessed 19 01 2017].

[70] E. Kuronen, “Epic Sensors in Electrocardiogram Measurement,” 2013.

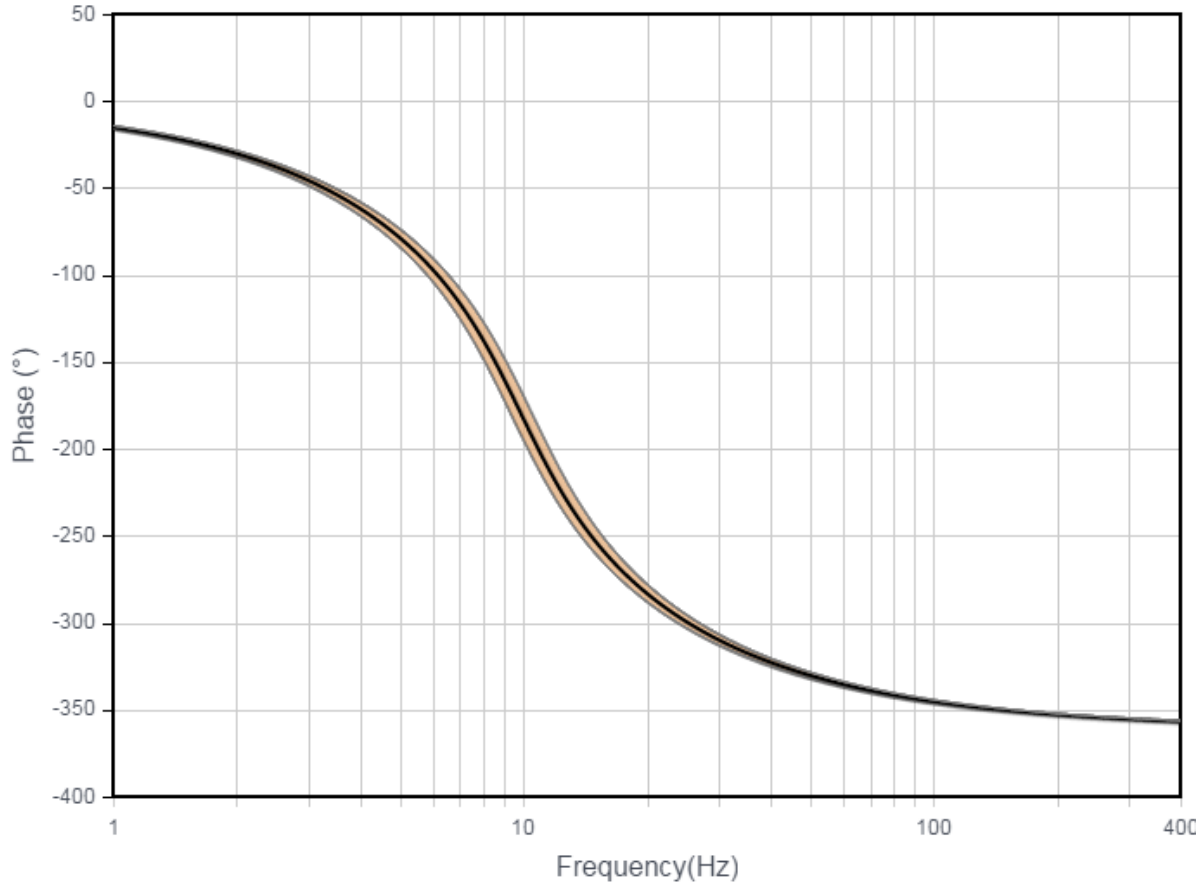
[71] Guinness World Records, “Lowest Heart Rate,” [Online]. Available: http://www.guinnessworldrecords.com/world-records/lowest-heart-rate. [Accessed 23 01 2017].

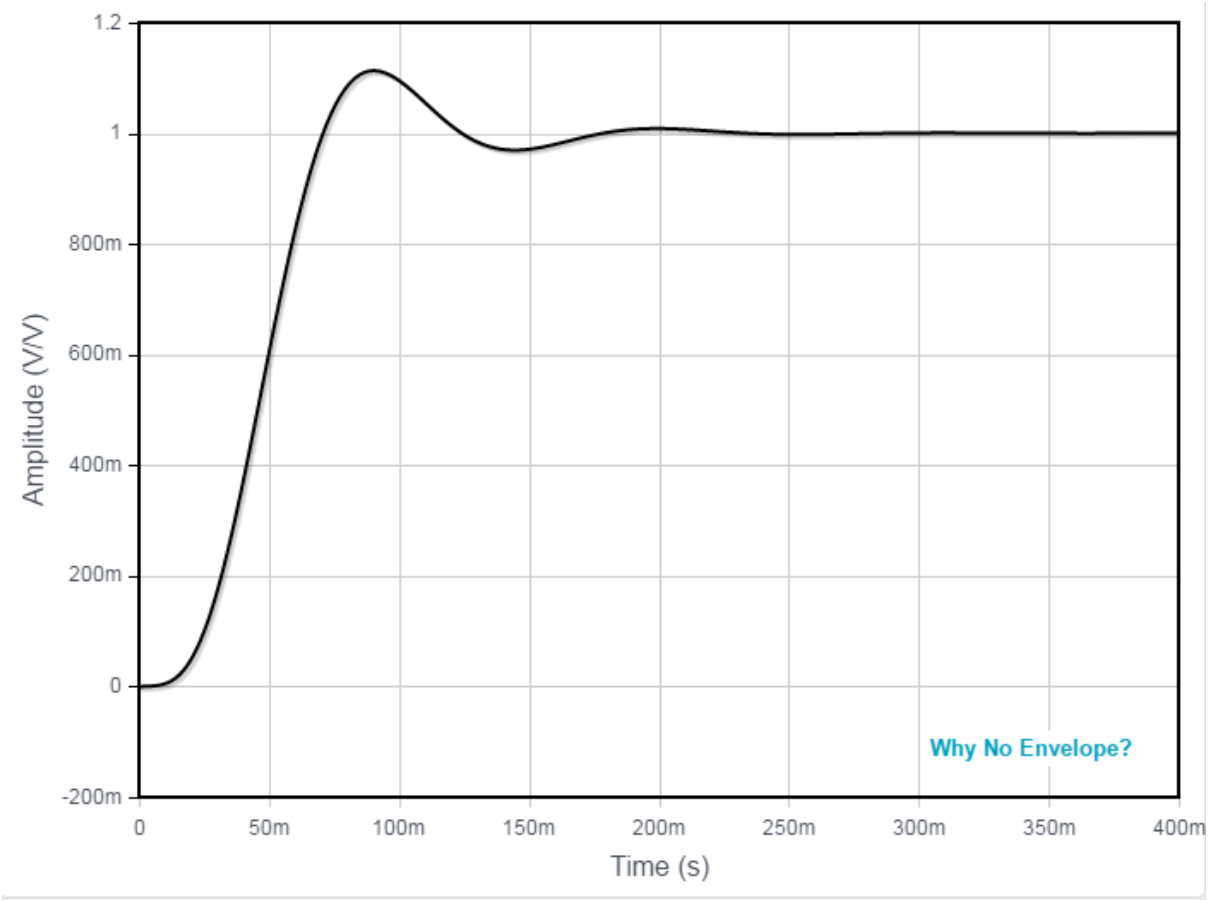
[72] Shimmer, “Consensys ECG Development Kits,” [Online]. Available: http://www.shimmersensing.com/products/ecg-development-kit. [Accessed 28 01 2017].

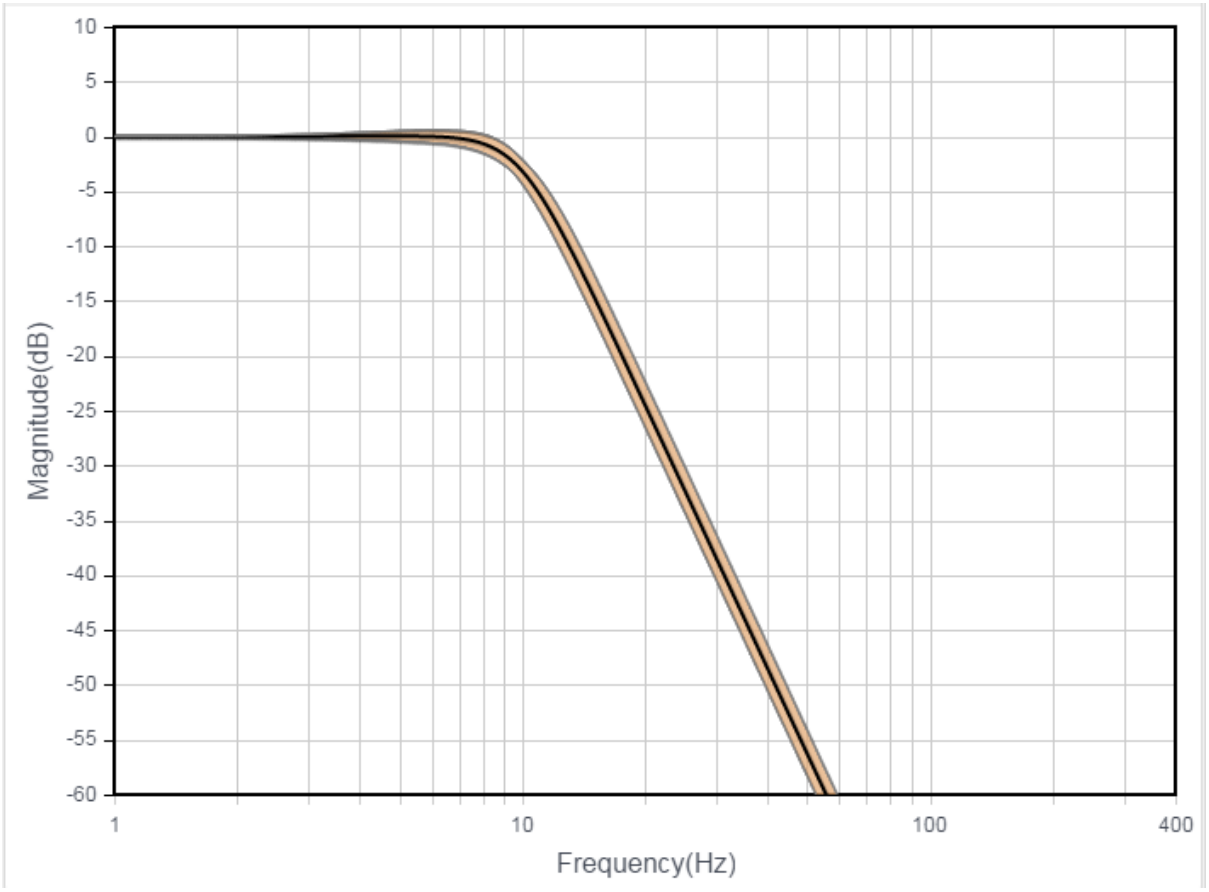
[73] D. Dobrev, “Two-electrode low supply voltage electrocariogram signal amplifier,” *Medical & Biological Engineering & Computing 2004, Vol 42,* pp. 272-276, 2004.

[74] D. Rowe and L. Tang, “Mobile Diagnosis Clinic,” p. 6, 2015.

## Appendix A – Active Filter Characteristics




1.2
1
800m
600m
400m
200m
0
-200m
Amplitude (V/V)
0
50m
100m
150m
200m
250m
300m
350m
400m
Time (s)
Why No Envelope?


10
5
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
-50
-55
-60
Magnitude(dB)
1
10
100
400
Frequency(Hz)