# Bluetooth 24-Hour Ambulatory Blood Pressure Monitor

Timothy Hewitt

16024996

Digital Systems Project

UFCFXK-30-3

# Table of Contents

## Contents

# Table of Figures

# Abstract

*This project involved the design and development of a Bluetooth enabled 24-hour ambulatory non-invasive blood pressure monitor. Using developmental boards, a custom android application was built in order to control the device and display a user interface, which presents a more modern take on traditional devices used in this field. In order to implement the hardware artefact, the oscillometric method was used in combination with the maximum amplitude algorithm along with digital signal processing techniques such as filtering which allowed for a signal produced by the pressure sensor to be used to calculate blood pressure readings correctly.*

# Acknowledgements

I would like to thank Craig Duffy for his much-needed direction at the start and continuing throughout the entirety of the project of which without significant progress would have not been made.
I would also like to thank my girlfriend Meggie for lending a voluntary ear (and arm!) whenever it was needed.

# Introduction

Blood pressure is one of the most commonly performed measurements within healthcare and is commonly achieved by means of a non-invasive automatic blood pressure monitor, whereby a cuff is inflated and then deflated with a sensor reading the changes in pressure of the cuff caused by the blood moving through the artery as it deflates. High blood pressure, more commonly known as hypertension can lead to a myriad of other health issues, including heart attack, stroke and internal organ failures such as kidneys. The main issue with hypertension is that it tends to develop slowly over a number of years, then culminates into a severe medical issue like the issues previously described.

Blood pressure is given as a reading of two numbers, one over the other (for example 120/80) which relate to the systolic and diastolic pressures within the arteries. The systolic pressure measurement is caused when the heart is actively beating and pushing blood through the arteries, whereas the diastolic pressure is when the heart is in between beats – when the cardiac ventricles relax and are preparing for the next contraction. According to the WHO (World Health Organisation) pressure measurements can be categorised as follows:

**Low**

Systolic: lower than 90 mmHg

Diastolic: lower than 60 mmHg

**Normal**

Systolic: lower than 140 mmHg

Diastolic: lower than 90 mmHg

**Possible hypertension**

Systolic: between 140 and 180 mmHg

Diastolic: between 90 and 110 mmHg

**Severe hypertension**

Systolic: higher than 180 mmHg

Diastolic: higher than 110 mmHg

## Why 24-Hour Monitoring?

Although blood pressure measurements are often conducted with a clinical situation, there are several advantages for performing ambulatory (at home) measurements over a 24-hour period.

Firstly, blood pressure is dynamic and is constantly changing based on the persons activity, what time of day it is, their emotional state and state of hydration among other factors. This means that collecting multiple readings will provide a much broader picture of the overall blood pressure for a

patient through the 24-hour period. Averaging readings or focusing in on the cause of specific readings should give a better idea of how to treat a patient and provide them with the best care.

Additionally, a phenomenon called white-coat hypertension can affect a patient's blood pressure. This is when a person exhibits hypertensive measurements while in a clinical setting, perhaps caused by anxiety of the situation which could lead to incorrect diagnosis. A study by Helvaci et al on white coat hypertension found that *"among 438 patients, 170 (38%) normotension (NT), 190 (43%) white coat hypertension (WCHT), 10 (2%) masked hypertension (MHT), and 68 (15%) sustained hypertension (HT) cases were detected".* This shows that 24-hour monitoring may provide a more valid result than singular measurements conducted clinically.

Currently, medically certified 24-hour monitors used by the NHS can cost upwards of £1000 and require recorded data to be downloaded once the device is returned. Accidental damage such as taking the device into the shower, or improper use can have a significant impact as often only a few of these devices are held per clinic. Therefore, this project aims to look into a proof of concept of designing and implementing a 24-hour ambulatory device that can be controlled and read over a Bluetooth smartphone connection – with data that could be sent over the internet to a patient's doctor or nurse.


*Figure 1 - Welch Allyn 7100 - Priced at £1195.69*

# Goals and Objectives

Goal: Create a 24-hour blood pressure monitor that can communicate via Bluetooth.

- Objective: The device can take a reading and calculate the users blood pressure.
- Objective: The device is able to measure and store readings periodically.
- Objective: If a reading fails the device will try again shortly after.
- Objective: The devices stored readings are able to be accessed securely via Bluetooth connection.

Goal: Create a mobile application that can connect to the monitor via Bluetooth

- Objective: The user can access the devices stored readings and perform menu operations.
- Objective: The user is able to initiate a reading from the application.
- Objective: The user can see results graphically.
- Objective: The user can access and change the devices settings.

Goal: The system is cost effective

- Objective: The monitor costs less than £50.
- Objective: The components should remain as accurate as possible.
- Objective: The device is able to be calibrated.

# Research

## Research Questions

Because of the distinct areas involved with the project, the research questions were divided into the three main sections of research; Algorithm, Hardware and Software.

### Algorithm

How are the systolic and diastolic measurements calculated from the value read by the pressure sensor?
How accurate does the algorithm need to be?
What other algorithms exist apart from oscillometric method?

### Hardware

What kind of pressure sensors are available and would be suitable?
What models of microcontroller would be suitable?
What current devices exist already?
What other hardware would be needed?

### Software

What software to use for the microcontroller?
Android Application research; how to connect to the device via Bluetooth?
Are there any libraries available for displaying pressure measurements graphically?

# Algorithms

## Background and Oscillometric method

Traditionally, blood pressure measurement was achieved by a manual sphygmomanometer with a doctor listening through stethoscope for the Korotkoff sounds made by the blood moving through the arteries, however automatic non-invasive blood pressure monitors today tend to use the oscillometric method whereby the signal recorded from the pressure sensor is analysed in order to produce the systolic, diastolic and mean arterial pressure measurements.

Although ramp-up methods exist in which the oscillations of the pulse are measured during cuff inflation most automatic monitors measure the oscillations after pumping up the cuff until the artery collapses, then slowly releasing the pressure(Fig 2). In order to process the pulses in the decaying pressure curve produced by the sensor, the oscillations need to be separated so that the maximum amplitude can be analysed for each pulse which results in an oscillometric waveform of peaks that increase then decrease as the deflation of the cuff occurs. Because of the nature of non-invasive blood pressure monitoring there is no physical way to extract the systolic and diastolic values so an algorithm must be used in order to calculate these measurements from the sensor data. There are many different techniques in processing an oscillometric waveform and algorithms on commercially manufactured devices are held as trade secrets, however the most popular algorithms generally require the oscillometric waveform to be smoothed into an amplitude envelope from the corresponding peaks of the waveform.

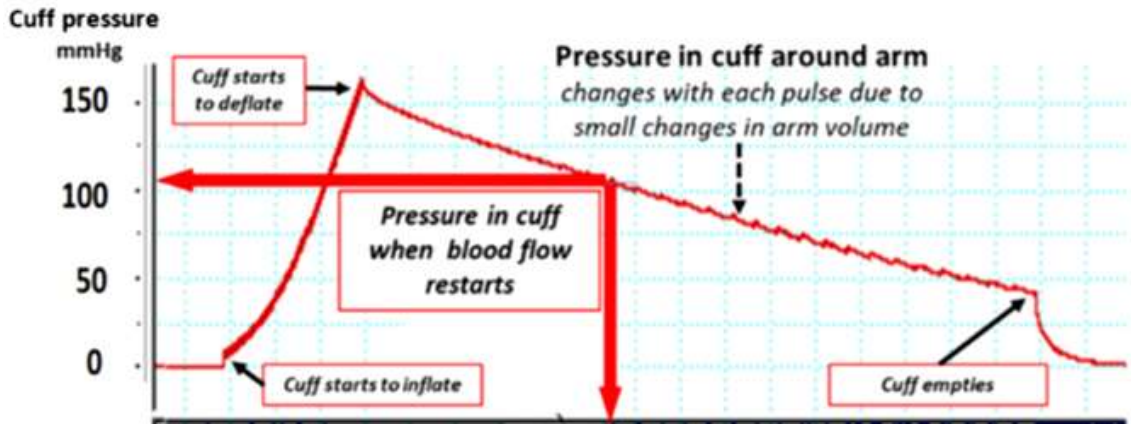



*Figure 2 – Graphical representation of cuff inflation/deflation routine*

## Validation Protocols

Although many devices on the market have not been independently validated, currently non-invasive blood pressure monitors that are available commercially are evaluated by two main protocols; the British Hypertension Society (BHS) protocol and the standard set by the US Association for the Advancement of Medical Instrumentation (AAMI). These protocols lay out guidelines for testing and calibration over a wide range of subjects with an ideal accuracy and aim to give each device a rating/grade based on its performance. To achieve grade A in the AAMI protocol for example, the test device must not differ from the mercury standard by a mean difference >5 mm Hg or a standard deviation >8 mm Hg. The AAMI standard for validating automatic BP monitors, requires measurements from at least 85 subjects with a minimum of 3 measurements per subject.

Table 1

Grading criteria used by the British Society of Hypertension.[1] Grades represent the cumulative percentage of readings falling within 5 mm Hg, 10 mm Hg, and 15 mm Hg of the mercury standard. All three percentages must be greater than or equal to the values shown for a specific grade to be awarded. Values are mm Hg

| | Absolute difference between standard and test device (%) | | |
|---|---|---|---|
| **Grade** | **≤5** | **≤10** | **≤15** |
| A | 60 | 85 | 95 |
| B | 50 | 75 | 90 |
| C | 40 | 65 | 85 |
| D | | Worse than C | |

*Figure 3 - BHS Grading Criteria for Blood Pressure Monitor Error Rate*

## Maximum Amplitude Algorithm

The maximum amplitude algorithm (MAA) is one of the simplest to implement and will be used for the initial prototype. Essentially the MAA uses ratios of the maximum amplitude in order to calculate the systolic and diastolic values as seen in Fig 2. The exact ratios to be defined are disputed and may be within a range between 0.45 to 0.73 for systolic and 0.69 to 0.83 for diastolic, also variants like cuff tightness, arterial stiffness and pulse rate can also affect these ratios, however other studies will be discussed that aim to improve upon this. For this method the maximum value from the oscillations of the pulse are indexed (OPI) using the oscillometric waveform(OMW), with the largest base to peak oscillation correlated to the mean arterial pressure. The chosen systolic and diastolic ratios are then applied to the mean arterial pressure and the peak of the indexed pulse closest to the left of the MAP for systolic and right for diastolic are taken as the corresponding values to display.

Chen et Al (2009) highlight the maximum amplitude algorithm in their assessment of algorithms for blood pressure measurement and look at two similar algorithms that may be able to be implemented. The first being the linear approximation algorithm, whereby two lines of best fit are used to approximate the amplitude envelope with the systolic and diastolic points on these lines calculated by using pre-defined ratios like the MAA. The second algorithm which was examined, was the points of rapidly increasing/decreasing slope, whereby the systolic point is taken to be where the slope of the envelope of the OMW increases the fastest and the diastolic point is taken to be where the slope of the envelope decreases the fastest. Essentially all three algorithms share a fundamental approach of using the index of the oscillations and their maximum height as its main criterion, meaning that separating the oscillometric pulses from the cuff pressure decaying in order to produce the oscillometric waveform is one of the key functions for the device in order to attempt to accurately gauge systolic and diastolic values.

## Systolic and Diastolic Ratios

Although the previously discussed MAA and linear approximation algorithm can be accurate, the issue with these algorithms lie with the systolic and diastolic ratios, which have to be chosen experimentally usually with a calibration device and/or with specialists using the auscultatory method in order to find the most suitable ratios. As Lee et Al (2013) state in their study *Blood Pressure Estimation Based on Maximum Amplitude Algorithm Employing Gaussian Mixture*

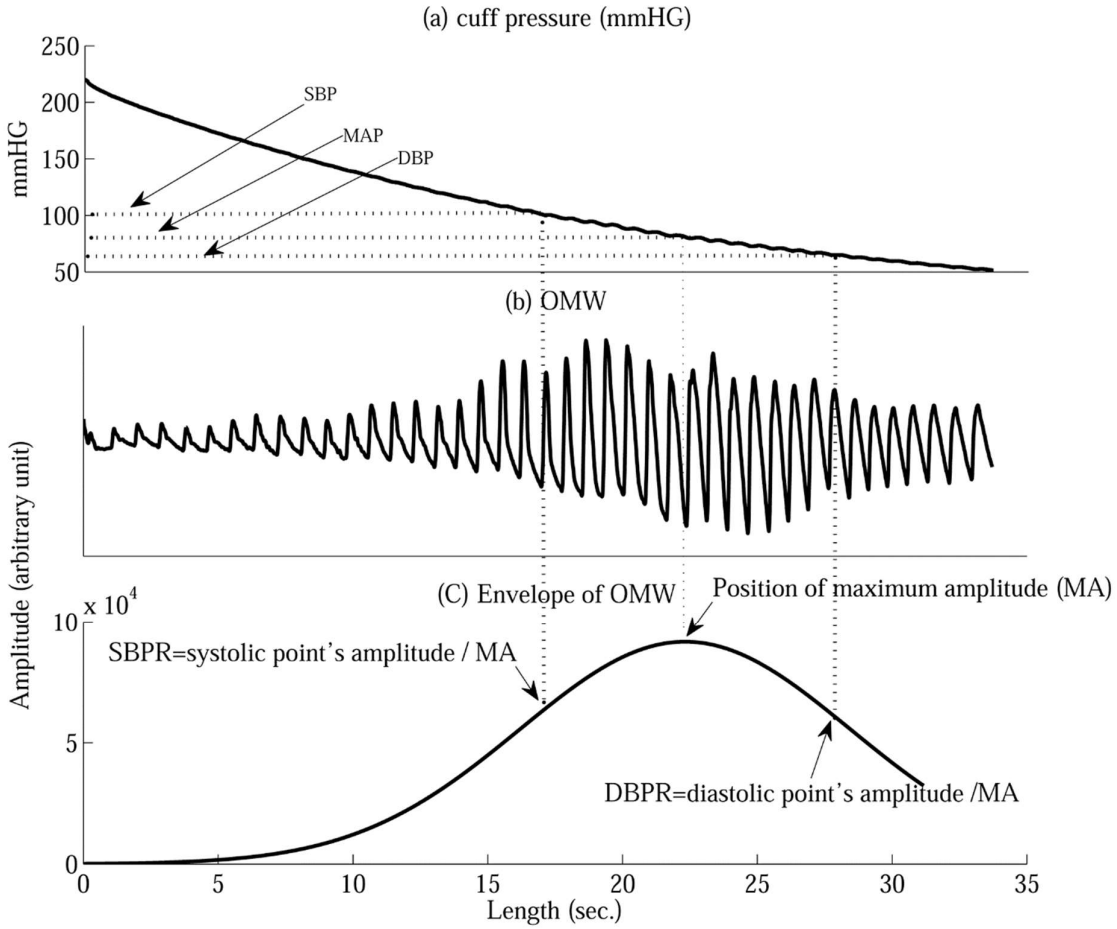



*Figure 4 - Cuff Deflation Correlation to Oscillometric Waveform*

*Regression* "the fixed ratio may be viewed as a value dependent on the measurements obtained for a specified group of subjects by minimizing the mean absolute error (MAE) relative to reference auscultatory measurements. If the ratios obtained from one group are used for another group, one would not be able to acquire reliable BP estimates." The study achieved a higher accuracy of systolic and diastolic ratios by firstly extracting relevant features from the oscillometric waveform (for example the mean arterial pressure, maximum amplitude of pulses, length of envelope etc) into a Gaussian mixture model (essentially multiple Gaussian distribution curves), which was then processed using a Gaussian mixture regression method. This resulted in a lower mean absolute error by around 3mmHg compared to the MAA used in the study, which were referenced to auscultatory nurse measurements taken of the 85 subjects used for the data set. Although an algorithm as complex as this study will most likely not be implemented, this highlights the importance of the ratios used in a traditional MAA in achieving an accurate result. As mentioned previously the AAMI standard protocol recommends a mean error value of less than 5mmHg , so this could be significant in producing a medically accurate device.

## Digital Signal Processing - Fast Fourier Transform, Filters

The fast Fourier transform (FFT) is an algorithm that produces the discrete Fourier transform (DFT) of a sequence of data. Essentially it takes in time or space domain data and produces frequency domain data. Although there were no blood pressure studies that showed their use of FFT in digital signal processing methods, it is often used in electrocardiogram analysis of heart/pulse rate, so may be useful when transforming the raw signal data into an oscillometric waveform. If the signal waveform is processed into its discrete Fourier transform, by analysing the highest amplitude frequency area it should provide a rough idea of the pulse pressure rate, this should aid in locating peaks for the pressure oscillations. .

The main signal processing method used in blood pressure measurement appears to be the use of filters. Koohi et al (*Coefficient-Free Blood Pressure Estimation Based on Arterial Lumen Area Oscillations in Oscillometric Methods*), used a Butterworth band-pass filter with a range from 0.5 – 20Hz in order to produce the oscillometric waveform needed for their mathematical model. Filtering allows noise to be reduced in a signal by altering the amplitude of certain frequencies which in turn allows specific frequencies to be focused on and analysed, which will be essential for implementation of a maximum amplitude algorithm. Filters tend to be categorised into two areas – finite impulse response and infinite impulse response. An impulse response can be described as a system output when presented with a short input signal – it is the filters appearance in its time domain. An IIR filter uses some or all of its output as the next input (recursive), making it infinite as it cannot reach zero, a FIR filter does not.

The equation for FIR and IIR filters is found below:


*Figure 5 - FIR/IIR Filter Equations*

Low-pass and high-pass filters are generally FIR, whereas band-pass filters like the Butterworth filter used in the study by Koohi et al are IIR.

# Hardware

## Initial Research

In order to determine which hardware components will be needed for a non-invasive blood pressure monitor, an unknown brand broken monitor donated by Craig Duffy was analysed with the resulting components displayed below.

| **Component** | **Information** | **Specifications** |
|---|---|---|
| Controller/Circuit Board | Pressure sensor mounted into board, attached to LCD screen via ribbon | Unknown brand/custom make |
| Pressure Sensor | Single port, most likely gauge type pressure sensor | |
| Motor/Pump | Motor/Pump attached as one unit, model number KPM27C-6B1, brand Koge | 6V DC Rated Voltage<br>4-7V DC Operating Voltage<br><10 Sec inflation time (0-300mmHg in 500cc tank)<br>>400mmHg Max Pressure<br>Max 3mmHg leakage (from 300mmHg at 500cc tank)<br><63db Noise Level (30cm Away) |
| Solenoid Valve | Solenoid valve attached with 2 wires, via 3-way tubing connector, Model number KSV05b | 6-12V DC Rated Voltage<br>60-45mA Rated Current<br>Exhaust time – Max 3 seconds from 300mmHg reduce to 15mmHg at 50cc tank<br>Max 3mmHg leakage from 300mmHg at 100cc tank<br>Mode: Input voltage closes valve |
| Exhaust Port | Small exhaust/release port attached to tubing via 3-way connector, presumably allows back pressure or is used to enable slow release of cuff pressure | No model/brand information |
| Tubing | Tubing attached directly to pressure sensor and other components via 3-way connectors, then to port allowing connection of cuff tubing | N/A |
| Power Source | 4 x AA batteries 1.5v<br>6V DC | N/A |

## Pressure Sensors

There are a large variety of pressure sensors available, so it is important to select one that is suitable and accurate enough for the application of a blood pressure monitor. Pressure sensors can generally be categorised into three main areas of measurement; Absolute, Gauge and Differential. Essentially these categories refer to the point of reference of which the sensor is comparing its measurement to. An absolute pressure sensor has a reference point of zero, or a perfect vacuum, meaning that its measurement is independent from environmental factors such as altitude or weather conditions. Gauge pressure sensors use the current atmospheric pressure as it's reference and measures the difference between this current ambient pressure and the pressure entering the sensor (they tend to have a vent which must not be obstructed in order to read the current atmospheric pressure). Differential pressure sensors usually have two input ports, with the resulting pressure measurement as the difference between these two ports. Therefore, for this application the suitable type of sensor would be either a gauge pressure sensor, or a differential pressure sensor with one port disconnected and exposed to the atmosphere. This also highlights the fact that gauge pressure sensors are in essence differential, the output being the difference between the atmospheric pressure and its input port.

For initial prototyping the main determining factor will simply be cost of the sensor, rather than aiming to be as medically accurate as possible. For this reason, after analysis of gauge type pressure sensors available on online electronic part stores the sensor selected is the NXP MP3V5050 series, at the time of writing costing £10.47. The MP3V5050GP operates with a minimum to maximum pressure of 0-50 kPa (kilopascal) with a maximum error of 2.5% between 0 °C to 85 °C. The sensor also operates with a supply voltage of 2.7-3.3V, making it ideal to be powered from the pin of a microcontroller. In order to read the measurement of the sensor, the voltage output pin is connected to an input pin on a desired microcontroller with the voltage correlating to the differential pressure as seen in Fig3.

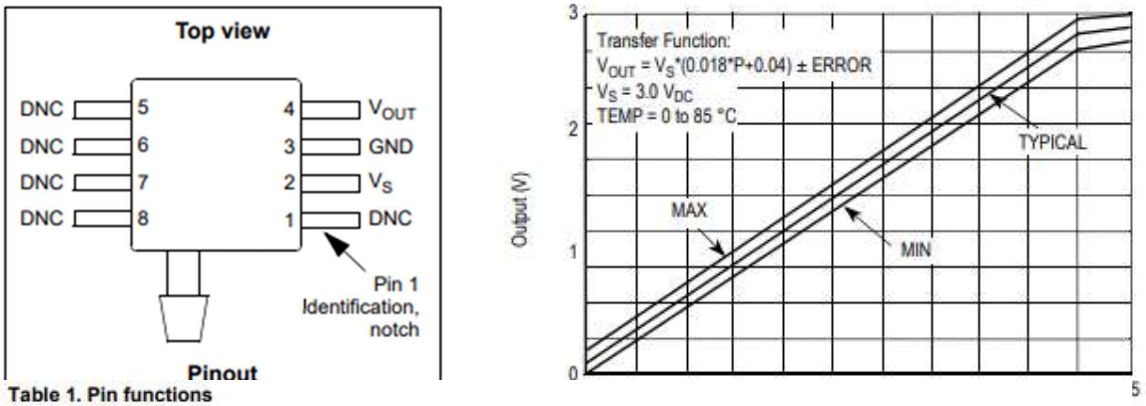



*Figure 6 - NXP MP35050GP Pinout and Transfer Function*

**Table 1. Pin functions**

| Pin | Name | Function |
|---|---|---|
| 1 | DNC | Do |
| 2 | $V_S$ | Voltage supply |
| 3 | GND | Ground |
| 4 | $V_{OUT}$ | Output voltage |
| 5 | DNC | Do not connect to external circuitry or ground. |
| 6 | DNC | Do not connect to external circuitry or ground. |
| 7 | DNC | Do not connect to external circuitry or ground. |

*Figure 7 - NXP MP35050GP Pin Functions*

## Microcontrollers

In order to aid research into what models of microcontroller will be suitable a list of requirements for the microcontroller was composed as follows:

- There should be enough storage for both:
    - Measured Readings over a 24-hour period (These can be stored simply as 3 integers for systolic, diastolic and MAP measurements, although date and time would ideally be stored as well)
    - Analysing and calculating the systolic and diastolic measurements from the pressure sensor readings, for example taking an average cuff decay time of 30 seconds, if the sensor data is to be read and stored every 5ms to an unsigned short int (2 bytes) then for a full measurement at least 12 Kilobytes of memory would be needed purely for the recorded sensor data. Using a float, which uses 4bytes of memory would double the required storage needed.
- The controller should be able to supply 3v in order to power the sensor
- The controller needs to have at least 2 output pins (pump/solenoid valve) and 1 input pin (pressure sensor)
- The controller needs a suitable ADC (Analogue to digital conversion) resolution
- The controller should be available at a suitable price point

Because the entire project development will be a definite learning experience two microcontrollers were selected in order to aid the speed of prototyping and further development. The first microcontroller selected will be the BBC Micro: Bit, this is because of prior use, its simplicity and low cost, as well as having Bluetooth capabilities. The Micro:Bit has limitations however, particularly in terms of its storage (the Micro:Bit has only 16k for program memory operations) and it's analogue-digital conversion which is only at a rate of 10 bits. The Micro:Bit will be developed using the DAL (device abstraction layer) and runtime designed by Lancaster University and will be coded in C++.

The second microcontroller that will be used for development is the slightly more complex ST Nucleo STM32L053. This board features a similar architecture as the Micro:Bit using the same Cortex M0+ which should aid in porting over any previously made code developed on the Micro:Bit. The board also features 64KB of flash memory and a higher 12-bit ADC conversion. Another advantage of this board is the ability to run in sleep or ultra-low power modes, which will assist in the 24-hour operation of the device. According to the ST datasheet, sleep mode power consumption at 16 MHz is about 1 mA with all peripherals off. FreeRTOS, which is a real time operating system used industry wide for microcontrollers of various applications, will be used to develop on the Nucleo board if OS scheduling is deemed necessary, however due to the memory size restrictions of this board, attempts will be made to develop a program with as little code print as possible. Additionally, an OLED screen was purchased which can be controlled via the microcontroller using I2C (I-squared-C) connections. This can be used to display the measurements taken during a reading, or any other user information such as Bluetooth connection status or the current operational mode. Unfortunately, as this controller does not have on-board Bluetooth communication capabilities an additional module will be needed. The HC-08 Bluetooth module was selected due to its low cost and significant number of online resources available. The HC-08 module communicates via a serial connection through the UART transmit and receive pins which can be read by the Nucleo. The code for this board can be developed on several platforms, including the online MBED compiler or the STM Cube IDE which has

the big advantage of step through debugging, this is because the Nucleo boards feature an embedded ST-Link which communicates via a USB serial connection. Coded printf statements can also be sent over this channel, allowing them to be read and displayed on a PC as long as a serial port has been opened (programs like CoolTerm can assist in this).

## Bluetooth (Bluetooth Low Energy)

Bluetooth and BLE are wireless technologies that allows devices to communicate over the 2.4ghz frequency band, generally over a short distance which is typically less than 10 meters. The advantages of Bluetooth for this project are that it is a cheap wireless technology that is readily available on almost all smartphones, is low power and relatively easy to implement. Bluetooth operates on a layer of protocols in order for devices to connect with each other, discover services and communicate wirelessly. The main way in which Bluetooth communicates data is by using Generic Attribute Profiles (GATT), which allows small pieces of data to be sent over the connection. Each GATT profile contains a service and each service has some characteristics. Every attribute is identified by a unique 128-bit string (UUID).

For the Micro:Bit, the GATT services which will needed to be accessed are the IO Pin Service, which will allow a connected device to access the data received by the pressure sensor, and the UART service, which will allow the connected device to send pre-defined commands to the Micro:Bit in order to initiate readings. According to the Micro:Bit runtime specifications the IO Pin Service characteristics are “structured as a variable length array of up to 19 Pin Number / Value pairs. Pin Number and Value are each uint8 fields.” However, because the Micro:Bit has a 10-bit resolution there is a loss of resolution when using the IO Pin Service, so it may be more appropriate for calculated pressure readings to be sent over the UART service to allow for more accurate measurements.

With the ST Nucleo, the BLE connection via the HC-08 module is dealt with by the serial connection through the UART Tx/Rx pins on the board. The HC-08, being a very simple and cheap module only has one service/characteristic that can be read and written to over its BLE connection, which is essentially the UART Tx/Rx connections attached to the board. The module features a switchable low power mode, which may be useful for conserving power while the device remains in a 24hour operational setup. To enable messages to be received from the BLE module while the device is still performing operations, an interrupt routine will need to be setup while developing the code for the microcontroller. This will allow messages to be stored and read later while the device is performing another task, like calculating a reading.

# Requirements

This chapter identifies the main functional and non-functional requirements for the device and corresponding application. The requirements were gathered from research, analysis of current devices, and project development. The prioritisation system used here is the MoSCoW system (must, should, could, won't).

## Functional Requirements

| Number | Description | Priority |
|---|---|---|
| FR1 | Measure blood pressure accurately with both systolic and diastolic measurements | M |
| FR2 | Be able to store at least 24 hours of readings | M |
| FR3 | Have enough power to last over 24 hours | M |
| FR4 | Broadcast data over Bluetooth connection | M |
| FR5 | The hardware should be light enough and unobtrusive | M |
| FR6 | The pressure sensor should match medical/clinical standards (AAMI, ANSI, BHS, EHS) | C |
| *FR7* | *Moved to non-functional NFR11* | |
| FR8 | The system should be controlled via Bluetooth (Single reading, Set 24 Hour/Time Period etc) | S |
| FR9 | Be fully automated when in 24-hour mode | M |
| FR10 | Be able to be manually switched off in case of failure | M |
| FR11 | Monitor when max pressure is achieved/artery collapses to minimise pressure time for user (vs setting static inflation time) | C |
| FR12 | Reading can be performed manually from device without Bluetooth | C |
| FR13 | Previous readings are stored and can be accessed and displayed from the device | S |

## Non-Functional Requirements

| Number | Description | Priority |
|---|---|---|
| NFR1 | The device should be controllable from IOS and Android applications | C |
| NFR2 | The monitor should automatically send readings (previously stored) when paired with a device | S |
| NFR3 | The application should have user accounts or a method of storing and distinguishing different user's data | C |
| NFR4 | The device should cost as little as possible whilst maintaining accuracy | S |
| NFR5 | The user can upload data to the cloud anonymously (for averaging data of multiple users) | W |
| NFR5 | The devices power source should be rechargeable | C |
| NFR6 | The devices motor/pump should be quiet | C |
| NFR7 | Inflate and process reading quickly to minimise discomfort (pump needs sufficient power) | C |
| NFR8 | Display the user's measurements graphically on the application | C |
| NFR9 | The device connection and all data must be secure and encrypted | S |
| NFR10 | The user should be able to change the devices operational functionality (Inflation time, frequency of measurements etc) | C |
| NFR11 | The monitor should attach to the cuff without a large amount of tubing | S |

## Requirements Testing

Based on the requirements outlined in the above section, tests that measure whether the requirements have been met were developed in order to validate the success of the device and its functionality.

| | |
|---|---|
| **Test Number** | 1 |
| **Requirements Tested** | FR1,FR12 |
| **Test Description** | The device performs a single reading and displays the resulting measurement |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm |
| **Flow** | 1. User presses start/hardware button on the device<br>2. User waits until inflation and deflation process has completed<br>3. Device displays user measurements after calculation |
| **Success Criteria** | Reading is displayed successfully with no erroneous result (within an expected range of systolic/diastolic values) |

| | |
|---|---|
| **Test Number** | 2 |
| **Requirements Tested** | FR1, FR4, FR8, NFR1 |
| **Test Description** | The device performs a single reading and displays the resulting measurement over its Bluetooth connection |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm<br>A suitable device is connected via Bluetooth |
| **Flow** | 1. User selects a single reading on connected device<br>2. User waits until inflation and deflation process has completed<br>3. Device displays user measurements after calculation |
| **Success Criteria** | Reading is displayed to the connected device successfully with no erroneous result (within an expected range of systolic/diastolic values) |

| | |
|---|---|
| **Test Number** | 3 |
| **Requirements Tested** | FR1, FR2, FR3, FR9, FR12, FR13 |
| **Test Description** | The device performs multiple readings over 24-hour period and stores the resulting measurements |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User (or testing object) has attached cuff |
| **Flow** | 1. User presses start/hardware button on the device (24 hour)<br>2. User waits until 24-hour period is over, and device has completed all measurements<br>3. User cycles through readings made over 24-hour period |
| **Success Criteria** | Readings are stored to the device successfully with any erroneous results being displayed (e.g. failed reading, not within expected sys/diastolic range) |

| | |
|---|---|
| **Test Number** | 4 |
| **Requirements Tested** | FR10 |
| **Test Description** | The device can be switched off while performing a reading and the cuff is deflated |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm<br>User has initialised a singular reading |
| **Flow** | 1. User presses start/hardware button on the device<br>2. User waits until inflation process has begun<br>3. User presses hardware button on the device to cancel the reading |
| **Success Criteria** | The reading is cancelled, and the cuff is deflated |

| | |
|---|---|
| **Test Number** | 5 |
| **Requirements Tested** | FR4, FR8, NFR1, NFR8 |
| **Test Description** | The device can perform a reading and display the pressure sensors data graphically |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm<br>User has connected to the device via Bluetooth<br>User has initialised a singular reading |
| **Flow** | 1. User presses start/hardware button on the device<br>2. User waits until inflation and deflation process has completed<br>3. Device displays resulting measure on a graph to the connected Bluetooth device |
| **Success Criteria** | The reading taken is displayed on the application graphically with no erroneous result |

# Design

## High Level Design

The device has four main components that are attached via tubing: the cuff, pressure sensor, motor/pump and solenoid valve. In order to allow pressure to slowly release after inflation, a small release valve is also fitted. In order to connect the solenoid valve and release valve to the tubular system a 4-way connector will be used. Initial design considerations are to place the pressure sensor closest to the cuff to ensure that it is the first point of backpressure as the cuff deflates, however different positions will need to be attempted with the results recorded to determine the difference in measurements based on the sensor location. The initial design of the tubing system is outlined below.

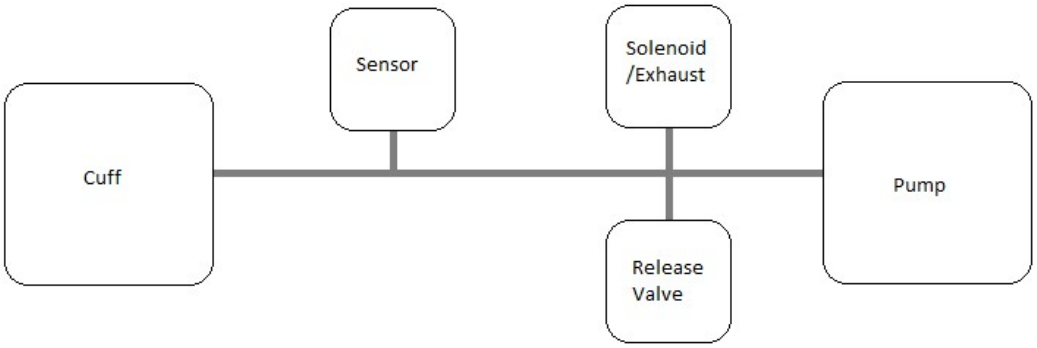



*Figure 8 - Monitor Tubing Design*

## Design 1 – Micro:Bit

The initial design components for the design using the Micro:Bit and costing are described below.

| Hardware | Description | Price |
|---|---|---|
| **Microcontroller** | Micro:Bit – Previous use, simple to implement, has Bluetooth functionality, low power and small physically, powered from micro USB/battery pack | £12.58 |
| **Pressure Sensor** | NXP MP3V5050GP – Within 2.5% accuracy 3V | £10.47 |
| **Motor/Pump** | KPM27C-6B1 micro air pump, 6V | £1.47 |
| **Solenoid Valve** | KSV05b 6V/12V | £3.15 |
| **Misc.** | Tubing and Cuff (Sourced from previously purchased monitor) | (£15.44) |
| | Prototyping Components – Breadboard, Wires, Transistor etc | £5.89 |
| **Total** | | £33.56 (£49.00) |

Because the Micro:Bit outputs power at 3V and the rated power requirement for the motor/pump and solenoid valve are 6V, a transistor is needed with an additional power supply in order to ensure the components can draw enough power. A simple circuit diagram is defined below.

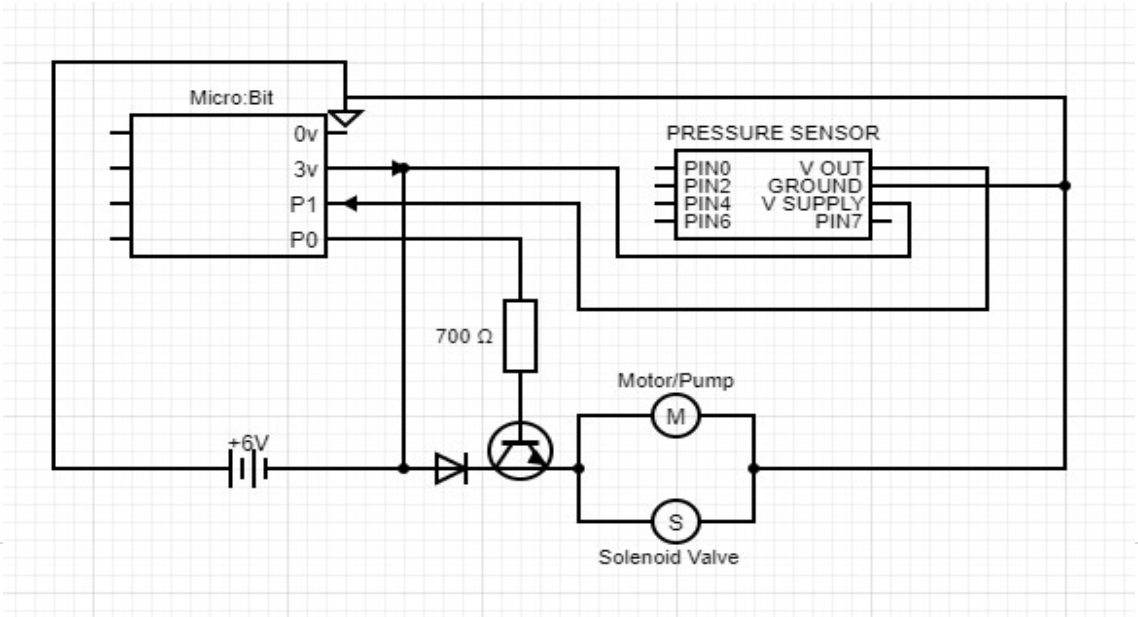



*Figure 9 - Micro:Bit Circuit Diagram*

## State Diagram

For the flow of events for a singular measurement cycle, a state diagram is outlined below.

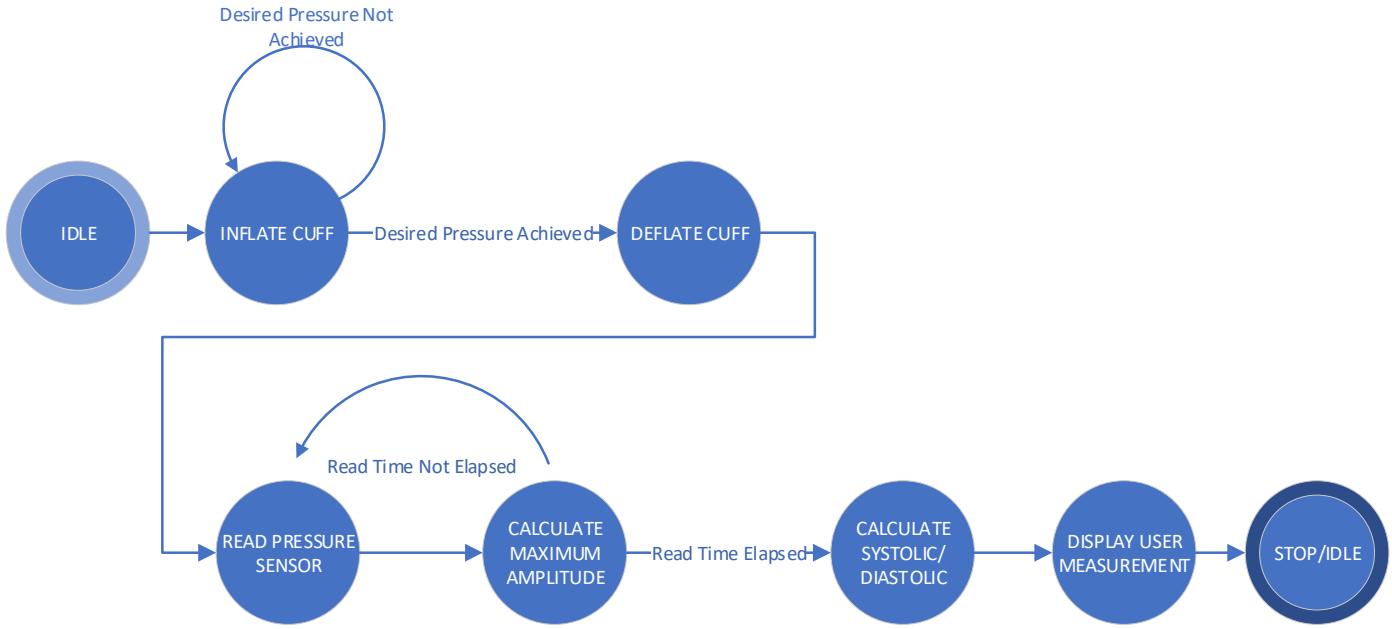



*Figure 10 - Singular Reading State Diagram*

## Design 2 – ST Nucleo Component List

| Hardware | Description | Price |
|---|---|---|
| **Microcontroller** | ST Nucleo – L053R8 – Ultra low power module, 32Bit Cortex – M0+ CPU. 64KB Flash 8KB SRAM | £11.43 |
| **Pressure Sensor** | NXP MP3V5050GP – Within 2.5% accuracy , 3V | £10.47 |
| **Motor/Pump** | KPM27C-6B1 micro air pump, 6V | £1.47 |
| **Solenoid Valve** | JQF1 – 3V Solenoid Valve | £4.99 |
| **Bluetooth Module** | HC – 08 BLE Serial Bluetooth Module | £5.99 |
| **Display** | OLED SSD1306 Display | £5.49 |
| **Misc.** | Tubing and Cuff (Sourced from previously purchased monitor) | (£15.44) |
| | Prototyping Components – Breadboard, Wires, Transistor etc | £5.89 |
| **Total** | | £45.73<br>(£61.17) |

## Android Application Design

Below is the prototype android wireframe designed for the application.

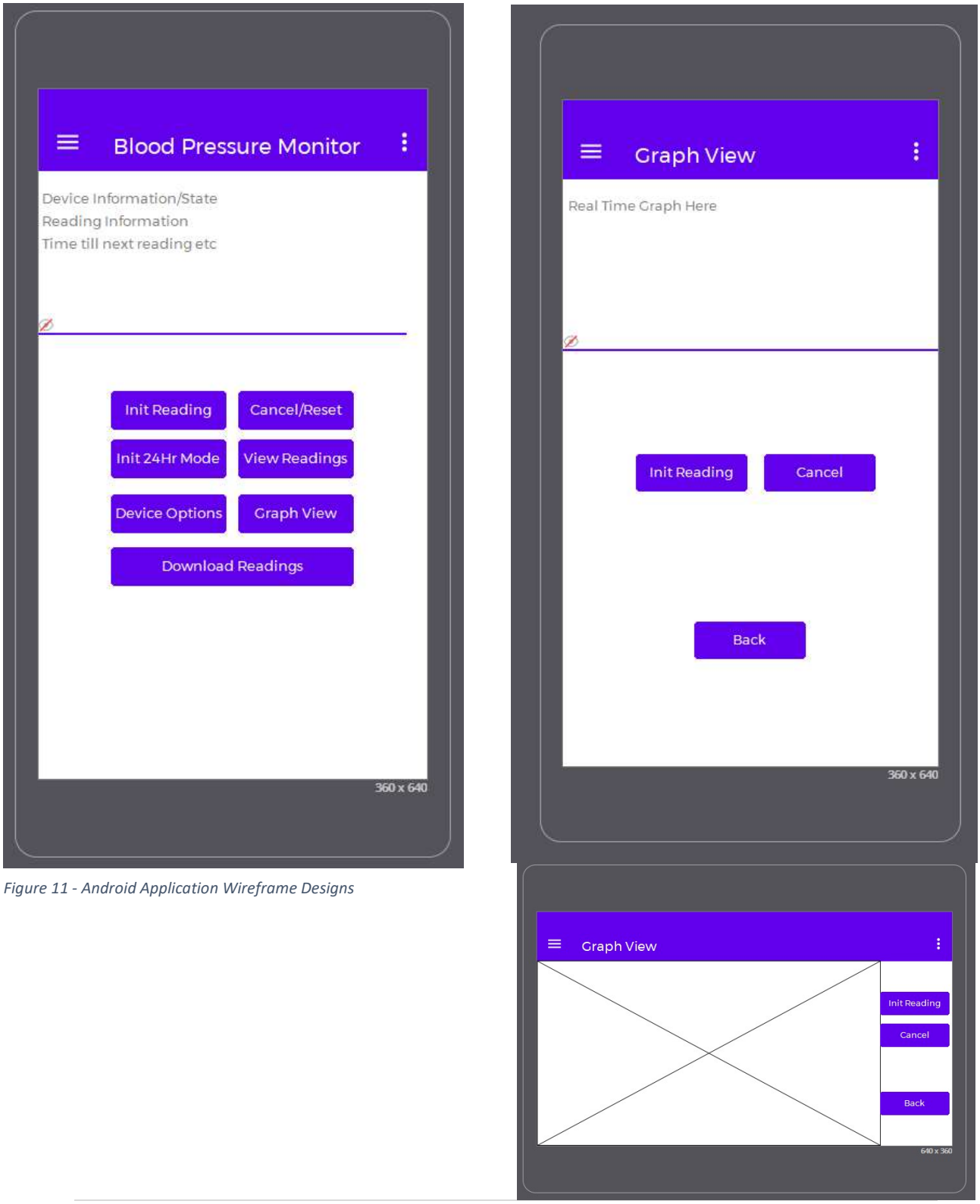



*Figure 11 - Android Application Wireframe Designs*

# Implementation

This section covers the overall build strategy as well as documenting some of the more complex functionality and programming needed for the hardware and software artefacts. As the project was largely a learning experience and there was little prior knowledge about embedded systems or digital signal processing, the implementation was conducted simultaneously with research which allowed for development of simple functionality to gain a good basis of knowledge before moving on to the more complex elements needed to calculate valid measurements from the pressure sensor.

## Methodology

The implementation methodology did not follow one strict approach due to the nature of the project, some tasks would require completion before another began, for example implementing serial data communication before reading that data in the application, therefore requiring a waterfall implementation strategy. Whereas tasks such as implementation of filters and signal processing would require an iterative approach of implementation and testing to ensure the results were as desired.

## Development Process

The development strategy was split up firstly by which microcontroller was being used and then into tasks relating to the current area of the build, for example communication or peripherals. The development phases were selected in order to ensure the learning curve was as linear as possible and to ensure that a firm grasp of the basics of embedded system programming were understood before moving on to additional functionality. In order to measure the success of each task, testing was conducted periodically after each set of tasks to ensure system behaviour was as expected. This allowed for rapid development without the need for additional testing strategies.

| Phase | Microcontroller | Area | Task |
|---|---|---|---|
| 1 | Micro:Bit | GPIO/DAL | Understand how the GPIO(Pins) are accessed and controlled via the device abstraction layer<br>Connect pressure sensor power and ground and output voltage to desired analogue input pin |
| | | Peripherals/Display | Write code to display current pressure sensor value to Micro:Bit LED matrix |
| | | Testing | Test and observe reading displayed |
| | | Hardware Components | Understand how to use an additional power supply and control it with the Micro:Bit GPIO (Transistor/Diode)<br>Connect power supply to transistor collector, motor/pump (6V) to transistor emitter |
| | | GPIO/DAL | Connect desired GPIO to diode, connect diode to transistor base, ensure motor/pump ground connected<br>Write code to turn on/off motor/pump on user button press |
| | | Testing | Test user button and check motor turns on/off as intended |
| | | Hardware Components | Connect solenoid valve to desired GPIO pin |
| | | Code | Develop cuff inflation routine by controlling pins attached to motor/pump and solenoid valve<br>Read in analogue value from pressure sensor on deflation and convert to mmHg as per NXP datasheet<br>Display converted pressure sensor data to LED matrix periodically |
| | | Testing | Test inflation routine and observe values are within reasonable range (0-200mmHg) |
| 2 | Micro:Bit | Communication | Understand the Bluetooth/message Bus and UART Micro:Bit services<br>Implement a routine to read in any messages over UART if a Bluetooth device is connected and display |
| | | Testing | Using an application (Bluetooth Serial Terminal) send messages and observe results |
| | | Code | Implement handling of serial messages - initiate/cancel cuff inflation routine<br>Implement sensor data send serially |
| 3 | Nucleo | GPIO/HAL | Understand the STM - HAL library<br>Connect hardware to desired pins |
| | | Code | Port previously made routines to Nucleo |
| | | Testing | Perform all previous tests |
| | MATLAB | Signal Processing | Record data for one measurement over serial connection<br>Analyse signal to determine best method to extract oscillometric pulse index (FFT/Filters) |

| | | | |
|---|---|---|---|
| | | | Write function to determine peaks of oscillometric pulse index (OPI)<br>Define maximum amplitude algorithm ratios and use OPI to determine MAP, Sys and Dia measurements |
| | | Code | Implement signal filtering methods in C++ and compare results to previous analysis (Compare OPI)<br>Implement maximum amplitude algorithm functions and compare results |
| 4 | Nucleo | Code | Port maximum amplitude algorithm to Nucleo |
| | | Communication | Implement HC 08 BLE module and test serial commands, implement Tx/Rx interrupt routine<br>Implement sensor data send serially |
| | | Peripherals/Display | Connect SSD1306 OLED screen to required connections (I2C)<br>Understand I2C communication protocol |
| | | Peripherals/Display | Implement I2C communication to SSD1306 and writing algorithm results to screen<br>Test inflation routine and SSD1306 results display |
| | | Code | Implement handling of erroneous algorithm measurements and error display<br>Implement 24-hour mode (take reading every hour) and sleep cycle - possible file handling<br>Implement 24-hour data send serially upon application request |
| 5 | Android | Application | Implement application views - Devices/Connection View, Main View, Graph View, Menus<br>Implement Bluetooth manager, scanning, displaying and connection of devices |
| | | Testing | Test device connection/Bluetooth manager implementation |
| | | Application | Implement GATT service/characteristic read and write<br>Implement serial service, serial socket, serial listener (read/write to Bluetooth GATT characteristic from serial data) |
| | | Testing | Test serial read and write to Nucleo - initiate measurement routines + send sensor data |
| | | Application | Implement graph view back end via MPChart or GraphView libraries, plot real time sensor data upon cuff inflation routine |
| | | Testing | Test graph view with inflation routine<br>Perform any additional tests - 24hour mode, forced erroneous measurement, data transfer of 24hour readings |

*Figure 12 - Build Development Phases*

## Phase 1 – Micro:Bit

The goal of this development phase was to gain an initial understanding of how to control and use the hardware components needed to inflate and deflate the cuff and read the pressure differences from the sensor. A breakout board from Kitronic allowed easier access to the pins of the device, with the pinout used to connect the pressure sensor pins to the corresponding board I/O.

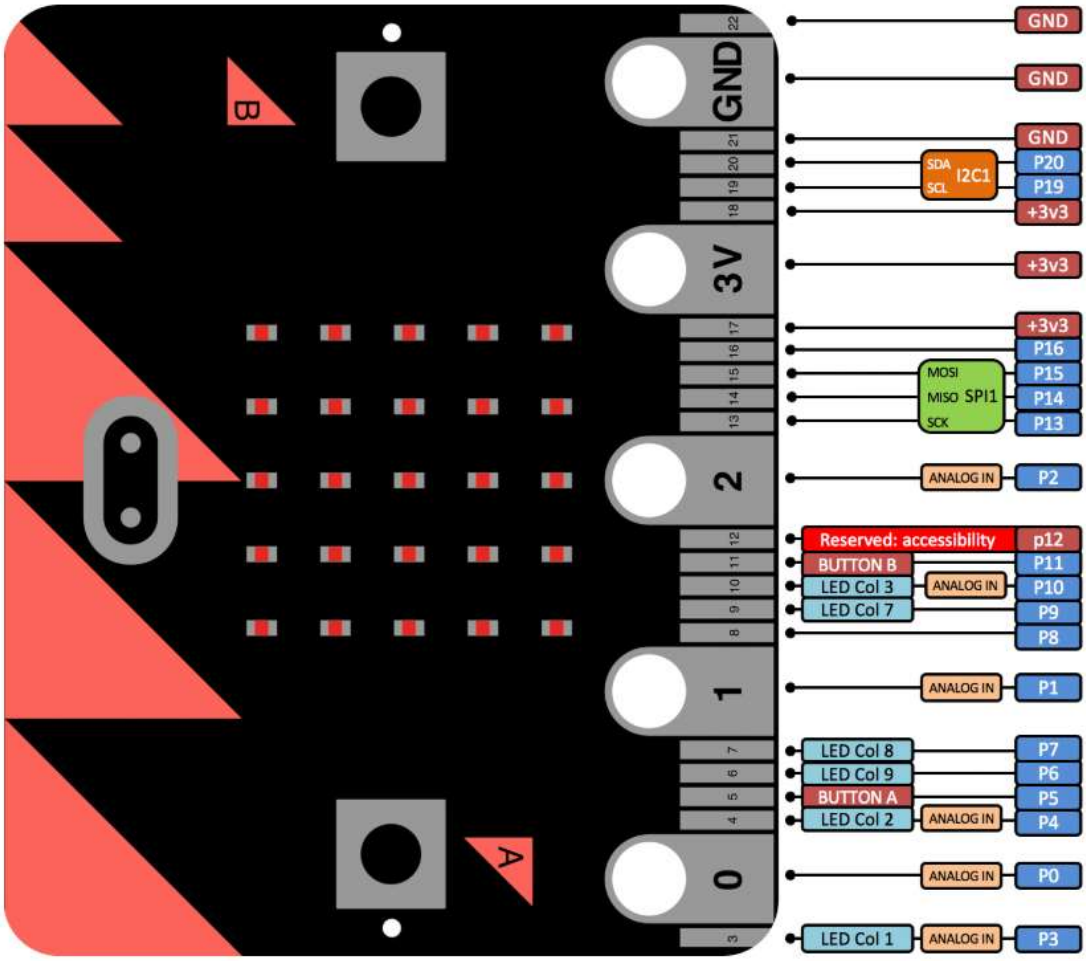



*Figure 13 - Micro:Bit Pinout*

Upon polling the sensor reading and displaying on the Micro:Bit LED matrix every second the voltage output of the sensor seemed to lie between 0.05 - 0.06. As the analogue input generates a floating-point value between 0 – 1 we can take this as between 0.15 – 0.18V. Using the transfer function of the NXP datasheet:

$$\textit{Voltage Out} = \textit{Voltage Supply} * (0.018 * P + 0.04) \pm \textit{ERROR}$$

This equates to a pressure measurement of around 0.3-0.8 kPa or 3-6 mmHg. As the pressure sensor is fluctuating by around 3mmHg, this could mean noise in the signal, possibly caused by bad grounds, incorrect voltage, or the fact the Micro:Bit was being powered by USB connection. With the error rate 2.5 VFSS (full scale span) being defined as “the algebraic difference between the output voltage at full-rated pressure and the output voltage at the minimum rated pressure” we can take

this as approximately an error rate of ± 0.0675V. The FSS is defined as typically 2.7V due to the signal saturating above this, which could give us a potential error rating of ± 0.972 kPa or 7.2mmHg. This is quite a significant error rate, so an upgrade could be to install the Honeywell ABPMANN005PGAA3 sensor available for a similar price, which is rated to 34kPa/258mmHg with an error rate of approximately ± 3.8mmHg. This would be more suitable for complying with BHS/AAMI standards which require a mean standard deviation of <5mmHg.

Using the transfer function, a conversion method was implemented to output the sensors voltage as mmHg and the solenoid valve and motor/pump power, ground and GPIO pins were connected as required. The solenoid valve is 2-way energized shut, so power needs to be applied to close the valve. The motor, as rated at 6V was attached and controlled via way of transistor PN2222A in order to draw from the additional power supply when voltage was outputted from the Micro:Bit pin.

Initially, the cuff inflation routine (started by button press), was simply to take the current system time and switch the device state from 'idle' to 'reading', whereby the forever loop would switch the output high to motor and solenoid pins if the elapsed time was less than 20 seconds, then the motor to low, solenoid to high if the elapsed time was less than 60, then change state back to idle upon completion. Testing showed this was a sufficient pumping time to achieve pressure enough to collapse the artery in the upper arm, and the exhaust valve attached to the tubing system allowed enough pressure to be steadily released from the system. As mentioned in requirement FR11, ideally this will be updated to allow a dynamic inflation time by monitoring the pressure value on inflation in order to minimise user discomfort.

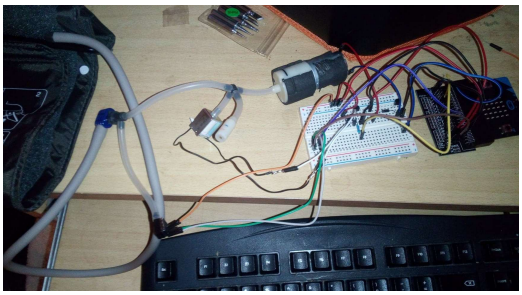


*Figure 14 - Micro:Bit Implementation*

## Phase 2 – Micro:Bit UART Communication

The goal for phase two was simply to enable simple strings to be sent to and from the Micro:Bit in order to further understand the UART communication protocol. UART/serial commands are intended to be used for operational commands for example changing device state to 24 hour or perform a reading. The Micro:Bit DAL runtime documentation highlights the simplicity of implementing this on the device, by enabling the UART Bluetooth service allows messages to be read until a defined end of message character is received. Testing using the android application Serial Bluetooth Terminal proved successful and strings were able to be sent and echoed back from the Micro:Bit. It should be noted that no serial input buffer, interrupt routine or message time out were implemented, all of which would be essential features for ensuring successful serial communication handling. The decision was made to forego sending the sensor data serially at this point, as the Nucleo board provided additional complexity necessary for implementing the previously mentioned features.

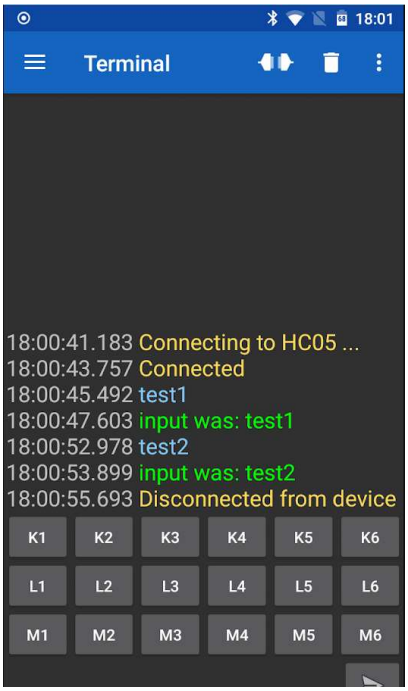


*Figure 15 - Bluetooth Serial Terminal Example*

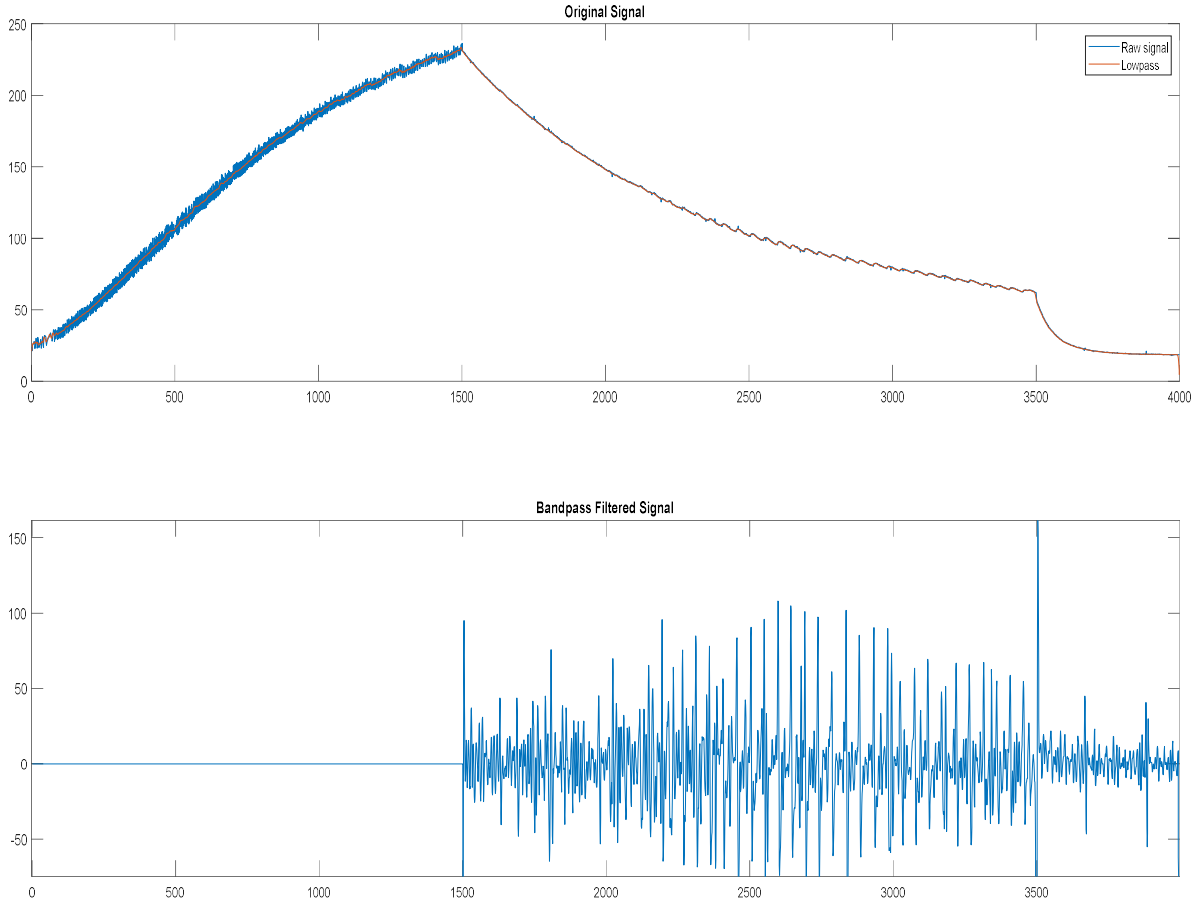
## Phase 3 – Nucleo and Matlab (Signal Analysis)

The aim for phase 3 was firstly to understand the differences between the ST HAL (hardware abstraction layer) compared to the Micro:Bit in order to successfully port over any previously made routines to the new device. Then, sensor data from a single cycle of the cuff inflation/deflation routine would be captured in order to analyse and visualize the results. The advantage of the Nucleo board is that the USB connection can act as a serial port, so data can be sent back to a connected computer and read using a program like CoolTerm.

In order to analyse the recorded sensor data, Matlab was selected due to its great graphing capabilities and ease of use for applying signal processing functions. The sensor voltage output was recorded at a sample rate of 50Hz for initial testing (average frequency of a resting heart rate usually lies between 1-1.67Hz) resulting in 4000 data points. Upon initial analysis and visualisation of the signal, it was immediately apparent some noise existed in the signal, so digital filtering would be

```
Fs = 50; % Sampling Frequency
N = 10; % Order
Fc = 0.2; % Cutoff Frequency
flag = 'scale'; % Sampling Flag
win = hamming(N+1);
b = firl(N, Fc/(Fs/2), 'low', win, flag);
Hd = dfilt.dffir(b);
%grpdelay(Hd,2048,Fs); % plot group delay
D = mean(grpdelay(Hd)); % filter delay in samples
FilteredBP = filter(Hd, [x; zeros(D,1)]); % Append D zeros to the input data
FilteredBP = FilteredBP(D+1:end); % Shift data to compensate for delay
```
*Figure 16 - Filter Design in MATLAB*

essential in order to smooth out signal spikes. For this reason, a simple low pass filter with an order of 10 and a cut-off frequency of 0.2Hz was introduced. The effects of this filter can be seen in sub plot 1 of figure (FIG NUM). It is important to note, that when applying a filter this does introduce




delay to the resulting signal data (dependant on the filter order), so this should be accounted for when display results. As stated before, one of the advantages of Matlab is the ease of use in implement signal processing functions, so a filter can be designed relatively simply, using the finite impulse response function fir1() to design a FIR filter (see Fig Num). Once the signal had been low passed, the next step would be to extract the oscillometric pulse index needed for use in the maximum amplitude algorithm. Based on the work by Koohi et al (*Coefficient-Free Blood Pressure Estimation Based on Arterial Lumen Area Oscillations in Oscillometric Methods*), a Butterworth band pass filter was designed with an order of 2 and cut-off frequencies between 0.2 – 20Hz. Once this filter was applied to the previously processed signal this produced pleasing results (sub plot 2 FIG NUM), albeit with some noise artefacts remaining, however the waveform should be good enough in order to correlate the peaks in amplitude to the corresponding pressures recorded, therefore the MAP and systolic and diastolic pressures can be extracted. More research and testing are definitely needed in this area, in order to fine tune the system and produce a result as accurate as possible. However due to the timeframe of the project and the need to develop the filtering methods in C/C++ for the microcontroller this will need to be done at a later date. Additionally, although Matlab has an inbuilt function for finding peaks of a signal, implementation on the Nucleo will be more complex. Initial thoughts are to develop a divide and conquer algorithm with supplementary parameters such as peak threshold and peak to peak distance in order to extract the required values from the oscillometric waveform generated by the signal processing.

## Phase 4 – MAA Development and Additional Functionality

The goal of phase four was to implement the methods determined in the signal analysis phase onto the Nucleo board, as well as finish routines for any additional functionality as required such as the 24-hour mode. As an added objective, the OLED SSD1306 screen was connected via the I2C bus in order to provide simple updates and device state to the user. The development of implementing the display proved easier than expected, due to a freely available library for the SSD1308 existing, which was easily adapted to the SSD1306 allowing for characters, strings and bitmaps to be written to the display.

However, the implementation of digital signal processing proved more difficult as this was arguably the most complex point of the project, and with little knowledge of the mathematics behind filtering a significant portion of time was spent researching the best methods in order to achieve results similar to the Matlab testing. Additionally, limitations with the Nucleo board, in particular its flash memory size, emerged due to the desire to use additional libraries designed for IIR (infinite impulse response) filtering, which would allow for real time filtering sample by sample (https://github.com/vinniefalco/DSPFilters, https://github.com/berndporr/iir1).

With FIR filtering the samples taken from the sensor will need to be stored in a buffer or array, so sensor data sampling was adjusted to only be recorded during the deflation of the cuff in order to conserve memory.

Designing the lowpass FIR filter as lightweight as possible was challenging, however there are several online resources that can help with this problem. TFilter, an online FIR filter designing tool aids in developing low pass, band pass, and high pass filters with the ability to generate source code in C once the filter has been setup as well as the ability to visualize the frequency response curve for the designed filter.

*Figure 17 - Lowpass and Bandpass Filter Results*

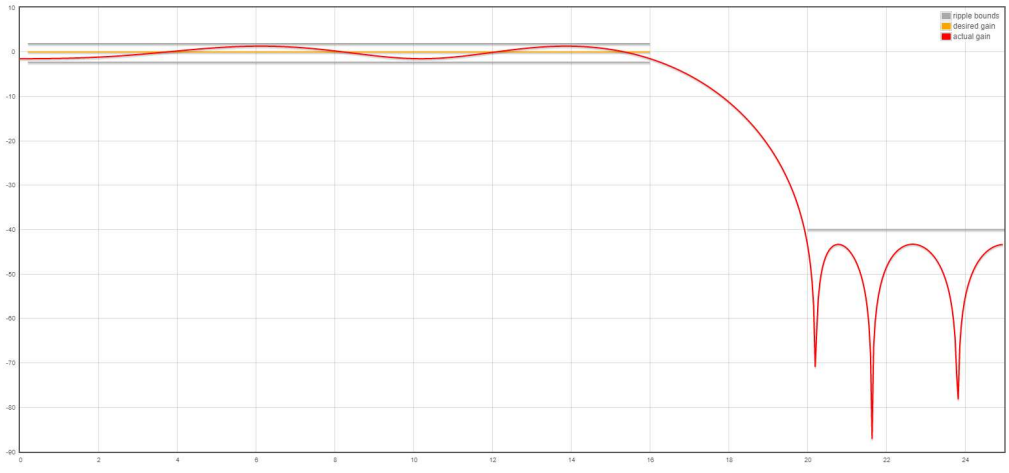



*Figure 18 - TFilter Lowpass Filter Design Example*

The interactive filter design by Tony Fisher (https://www-users.cs.york.ac.uk/~fisher/mkfilter/) helped in designing the Butterworth bandpass filter needed to produce the oscillometric waveform. Although the results from these filters differed from the filters designed in Matlab, the processed

signal should be sufficient in order to perform the peak analysis required for the maximum amplitude algorithm.

As well as implementing signal processing methods, this phase also added additional serial communication protocols necessary for the device to function to the user as expected. This involved setting up an interrupt routine caused by the transmit and receive pins on the Nucleo to ensure that the user would be able to send commands to the device, for example cancelling a reading, which would be read by the device and processed even if the device was currently performing a different action.

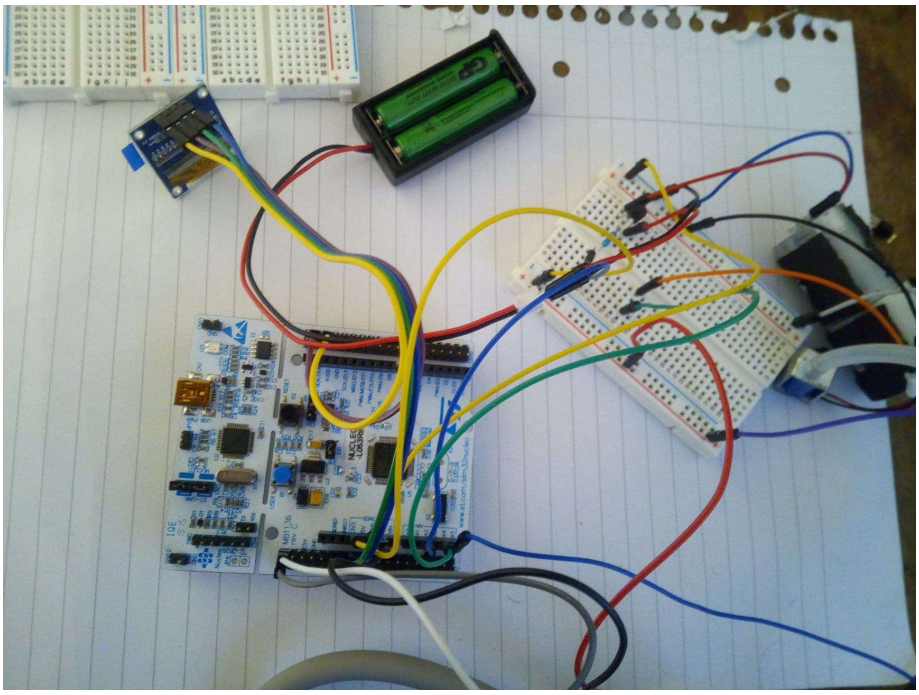


*Figure 19 - ST Nucleo Board Development Progress*

## Phase 5 – Android Application Development

Phase five aims are to develop the android application that will provide a user interface for controlling the main operational functionality for the hardware device and displaying key information to the user. Additionally, a secondary goal is to develop a view for displaying the filtered sensor data graphically in real time when a reading is initiated. However, as real time sensor data will need to be processed using an IIR filter in order to produce a graphically smooth result the raw sensor data will need to be filtered by the android application which may produce different results than the microcontroller depending on the filters that are implemented therefore additional signal analysis may be required.

Using android studio and coding in Java, key API's that will need to be implemented are the Bluetooth Manager/BLE GATT Callback to handle the Bluetooth connection and reading and writing of GATT characteristics, Service API to allow serial data to be collected in the background, and Listeners to update the user interface with any necessary data. In order to implement a real time graphical view, the library MPAndroidChart can be used to display the sensor data, although dynamic/real time data is not officially supported by this library, initial tests and video tutorials demonstrate that it can be achieved.

# Testing

This section documents the requirements testing as well as additional validation testing performed against another commercial non-invasive automatic blood pressure monitor produced by A&D Medical (Model UA-611). The method used by the A&D monitor is unknown but thought to be oscillometric with a maximum amplitude algorithm.

## Validation Test Results

A total of 2 readings per test were conducted two university students, one healthy male and one healthy female, although it should be noted that effects of a previous flu-like illness may have still been present which could have affected results. Tests were conducted at rest with at least 5 minutes in between each test. In order to demonstrate the relationship of changing the systolic and diastolic ratios used in the maximum amplitude algorithm, the data from the tests from the Bluetooth device were recorded and processed separately using the ratios 0.85/0.85 (ratio 1) and 0.7/0.75 (ratio 2) for systolic and diastolic values respectively. Testing results are displayed below.

Male:

| Test Number | Device | Systolic (mmHg) | Diastolic (mmHg) | MAP (Mean Arterial Pressure) (mmHg) |
|---|---|---|---|---|
| 1 | A&D | 131 | 84 | 99.67 |
| 2 | A&D | 129 | 86 | 100.33 |
| 3 (Ratio 1) | Bluetooth | 117 | 103.5 | 110.2 (108) |
| 4 (Ratio 1) | Bluetooth | 122.4 | 90.76 | 99.18 (101.3) |
| 3 (Ratio 2) | Bluetooth | 124 | 101.4 | 110.2 (108.93) |
| 4 (Ratio 2) | Bluetooth | 114 | 89.11 | 99.18 (97.33) |

Female:

| Test Number | Device | Systolic (mmHg) | Diastolic (mmHg) | MAP (Mean Arterial Pressure) (mmHg) |
|---|---|---|---|---|
| 1 | A&D | 105 | 59 | 74.33 |
| 2 | A&D | 103 | 68 | 79.67 |
| 3 (Ratio 1) | Bluetooth | 102.8 | 89.84 | 96.12 (94.16) |
| 4 (Ratio 1) | Bluetooth | 106.5 | 90.32 | 95.23 (95.71) |
| 3 (Ratio 2) | Bluetooth | 107.8 | 86.82 | 96.12 (93.81) |
| 4 (Ratio 2) | Bluetooth | 108.3 | 85.43 | 95.71 (93.05) |

## Results Analysis

As the MAP is not shown on the A&D device the MAP was calculated using the formula: MAP = 1/3 * SBP + 2/3 * DBP. For the Bluetooth device the MAP result in brackets is the MAP calculated using this formula rather than using the highest amplitude peak in the recorded data.

Although the systolic values recorded for both testing subjects are relatively similar in both devices, it is interesting to note how the A&D device calculates the diastolic values as significantly lower than the algorithm employed in the Bluetooth device. This could be for a number of reasons, however the simplest is that the ratios employed for the calculating the diastolic pressure are much lower in the A&D device. Further testing of changing the ratio values could bring these numbers closer together, however it is hard to know which algorithm is performing the most accurately without some way of calibrating the devices.

## Requirements Testing

The test results from the testing routines developed in the requirement section are defined below.

| | |
|---|---|
| **Test Number** | 1 |
| **Requirements Tested** | FR1,FR12 |
| **Test Description** | The device performs a single reading and displays the resulting measurement |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm |
| **Flow** | 4. User presses start/hardware button on the device<br>5. User waits until inflation and deflation process has completed<br>6. Device displays user measurements after calculation |
| **Success Criteria** | Reading is displayed successfully with no erroneous result (within an expected range of systolic/diastolic values) |
| **Test Result** | FAIL<br>Ability to start readings manually from the hardware device have not been implemented at this stage |

| | |
|---|---|
| **Test Number** | 2 |
| **Requirements Tested** | FR1, FR4, FR8, NFR1 |
| **Test Description** | The device performs a single reading and displays the resulting measurement over its Bluetooth connection |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm<br>A suitable device is connected via Bluetooth |
| **Flow** | 4. User selects a single reading on connected device<br>5. User waits until inflation and deflation process has completed<br>6. Device displays user measurements after calculation |
| **Success Criteria** | Reading is displayed to the connected device successfully with no erroneous result (within an expected range of systolic/diastolic values) |
| **Test Result** | PASS |

| | |
|---|---|
| **Test Number** | 3 |
| **Requirements Tested** | FR1, FR2, FR3, FR9, FR12, FR13 |
| **Test Description** | The device performs multiple readings over 24-hour period and stores the resulting measurements |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User (or testing object) has attached cuff |
| **Flow** | 4. User presses start/hardware button on the device (24 hour)<br>5. User waits until 24-hour period is over, and device has completed all measurements<br>6. User cycles through readings made over 24-hour period |
| **Success Criteria** | Readings are stored to the device successfully with any erroneous results being displayed (e.g. failed reading, not within expected sys/diastolic range) |

| | |
|---|---|
| **Test Result** | PARTIAL PASS<br>Testing has not been conducted for a full 24-hours as of writing, however periodic reading tests have been successful |

| | |
|---|---|
| **Test Number** | 4 |
| **Requirements Tested** | FR10 |
| **Test Description** | The device can be switched off while performing a reading and the cuff is deflated |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm<br>User has initialised a singular reading |
| **Flow** | 4. User presses start/hardware button on the device<br>5. User waits until inflation process has begun<br>6. User presses hardware button on the device to cancel the reading |
| **Success Criteria** | The reading is cancelled, and the cuff is deflated |
| **Test Result** | PASS<br>User can press RESET button at any time to reset the device |

| | |
|---|---|
| **Test Number** | 5 |
| **Requirements Tested** | FR4, FR8, NFR1, NFR8 |
| **Test Description** | The device can perform a reading and display the pressure sensors data graphically |
| **Preconditions (If applicable)** | The device is on<br>The cuff is attached to the device<br>User has attached cuff to upper arm<br>User has connected to the device via Bluetooth<br>User has initialised a singular reading |
| **Flow** | 4. User presses start/hardware button on the device<br>5. User waits until inflation and deflation process has completed |

| | |
|---|---|
| | 6. Device displays resulting measure on a graph to the connected Bluetooth device |
| **Success Criteria** | The reading taken is displayed on the application graphically with no erroneous result |
| **Test Result** | FAIL<br>Graphical View has not been implemented at time of testing |

# Evaluation

This section provides an evaluation to the successes of the project and aims to highlight areas of the project that went well, and what could have been improved.

## Goals and Objectives

All goals described in the goals and objectives section have been achieved, however some of the objectives at this moment have not been successful.

The project successfully created a blood pressure monitor that was able to communicate via Bluetooth to an android device which was able to initiate and calculate a reading and store the readings periodically in the case of 24-hour operation. However, due to time constraints and lack of ability to test on real subjects, the 24-hour operation did not have sufficient testing, although time periodic tests were successful.

At the time of writing, one objective that was not met was the ability to automatically retry a failed reading shortly after an erroneous or unsuccessful attempt. This was simply due to a lack of time in the implementation phase; however, this should not be a particularly hard objective to implement in the future by analysing the measurement and comparing it to a set of arbitrary values, retrying if outside of them.

The application can successfully control and operate the device as required, and the user is able to view the reading data from the mobile device, however currently there is no ability to change settings on the device or calibrate the sensor.

The hardware components of the monitor were realised at the desired price point of under £50.00 for both designs, achieving success at building a device at a reasonable price point.

One area which will definitely require further improvement is the lack of Bluetooth security in the device. Due to a large proportion of the time spent understanding and implementing the signal processing and algorithm segments, security was left as an afterthought and was not successfully implemented. This would be essential on a commercial hardware artefact, as the device can be controlled and initiated over Bluetooth, posing a large security risk.

## Research

The decision to research essential topics when they were required in the implementation stage was essential to the success of the project as it allowed a good base of understanding before moving on to the more complex elements such as the digital signal processing. Various methods of research were undertaken, including browsing scientific journals, viewing instructional videos and reading books. In particular the book The Scientist and Engineer's Guide to Digital Signal Processing by Steven W. Smith was invaluable in gaining a basis of understanding about signal processing techniques.

## Implementation

The implementation strategy worked well, and the decision to split up the build strategy into phases that increased in complexity was a good method of allowing understanding before moving onto the more complex elements. However, the complexity of the digital signal processing was not fully realised until the implementation stage, which did mean that it was hard to manage time effectively at that stage due to such a significant portion of the project being focused on this area. This led to some other areas suffering, for example as mentioned before, Bluetooth security was completely left out as it was not seen as immediately functional, although would be crucial in a real-world scenario.

Working on the Micro:Bit as almost a pre-prototype platform allowed for an ease of understanding before the ST Nucleo, however due to its limitations that were swiftly encountered, it may have been more time effective to simply dive in to learning about the Nucleo, especially due to the quality of resources and instructions available that were extremely helpful when learning about the device.

With android application, even though simple, implementation was difficult, but again due to a huge number of online resources and documentation available was achievable. Luckily previous experience in Java and IOS development eased this process, which without the application would most likely not have been implemented.

## Further Improvements

Aside from the lack of Bluetooth security mentioned above, there are a number of future improvements that could be implemented in order to provide a more complete and useful device.

In 24-hour operation the added function of a buzzer or alarm as well as updating the display when there is 5 minutes before another reading will commence would be a good improvement as this will alert the user to allow them to ensure they are at rest and the most accurate measurement is recorded. Additionally, if connected by Bluetooth a notification could be sent to the user's phone, meaning that the user would not need to constantly wear the arm band as long as a connection was maintained.

One improvement which could heavily impact the users' comfort is to try and implement the work achieved by the study by Koohi et al (Coefficient-Free Blood Pressure Estimation Based on Arterial Lumen Area Oscillations in Oscillometric Methods ). The study presented a novel approach to the algorithm used in blood pressure measurement which only used the diastolic region of information in the oscillometric waveform, essentially pressure values from 80mmHg to 20mmHg. This could be extremely helpful in reducing discomfort when performing multiple readings over 24 hours, as the cuff would not need to be inflated to maximum pressures, therefore reducing the overall time of measurement and the overall pressure sustained upon the upper arm of the user. The issue with implementing this study is that the lower pressure areas of deflation are those most affected by noise, particularly in this projects system, so removal of the noise or improving the hardware aspects of the system to reduce it would be paramount. The study achieved results that were demonstratable as accurate as the maximum amplitude algorithm, so is definitely something to consider for the future.

In order to improve the accuracy for readings, and to try and aim for BHS and AAMI validation the sensor component would need to be upgraded to produce a result with as little error rate as possible. As mentioned in the implementation section, moving from an error rate of 2.5% to 1.5% can reduce the error rate by approximately 3mmHg, which could be crucial in maintaining the low error rate needed for BHS/AAMI validation protocols.

# Conclusion

In conclusion, this project has made a good attempt at achieving the goals and objectives set out at the start of the report, which has resulted in a hardware and software artefact capable of performing automatic blood pressure measurements and communicating the results via Bluetooth connection to a user of the android application. Implementation of the maximum amplitude algorithm and oscillometric method studied in in the research section was successful although the complexity of processing the signal received from sensor was not realised until partial progression into the project had been made and was definitely overlooked at the initial conception of the system design.

The project was most definitely a learning experience, with particular emphases on digital signal processing, of which little was known prior to starting the project, so progress was not as linear as hoped. However, in testament to the multitude of resources and advice online and after much learning and testing through different means such as MATLAB, digital signal processing techniques were able to be implemented successfully onto the developmental board.

Perhaps more significantly, this project has helped in developing a considerable interest in the field of embedded systems programming, of which little had been taught prior. A more powerful and sizeable ST Nucleo board was purchased towards the end of the project with the intention of porting the code developed to the new board in order to see the benefits of a faster processor and more memory, as well as seeing what other projects could be attempted, particularly with a RTOS (real time operating system).

Finally, with a longer development time and implementing some of the improvements suggested in the evaluation section, this project could pose a significant improvement to the devices currently used in professional settings. With the device costing under £50, and if clinical validation protocols are able to be achieved, this could have the potential to save money in a real-world scenario.

*Source code will be made public at the repository [https://github.com/t3-hewitt/BPMonV2](https://github.com/t3-hewitt/BPMonV2) upon completion of this report.*

# References


Sphygmomanometers-Part NI. 2: Clinical Validation of Automated Measurement Type. ANSI/AAMI/ISO Standard 81060–2; 2009.

Helvaci, Mehmet Rami, and Mahmut Seyhanli. “What a High Prevalence of White Coat Hypertension in Society.” Internal Medicine, vol. 45, no. 10, 2006, pp. 671–674., doi:10.2169/internalmedicine.45.1650.

O'brien E, Petrie J, Littler W, de Swiet M, Padfield PL, O'malley K et al. The British Hypertension Society protocol for the evaluation of automated and semi-automated blood pressure measuring devices with special reference to ambulatory systems. Journal of hypertension. 1990. July 1;8(7):607–19.

Welch, A. (2019) Welch Allyn 7100 Ambulatory Blood Pressure Monitor with Hypertension Software. Available from: https://www.medisave.co.uk/welch-allyn-7100-ambulatory-blood-pressure-monitor-hyper.html?gclid=EAIaIQobChMIg5iCv6Gn5wIVB7DtCh0pvAGxEAQYAyABEgKDDvD_BwE .

Chandrasekhar, A., Yavarimanesh, M., Hahn, J., Sung, S., Chen, C., Cheng, H. and Mukkamala, R. (2019) Formulas to Explain Popular Oscillometric Blood Pressure Estimation Algorithms. Frontiers in Physiology. 10 pp.1415.

Celler, B.G., Argha, A., Le, P.N. and Ambikairajah, E. (2018) Novel methods of testing and calibration of oscillometric blood pressure monitors. Plos One. 13 (8), pp.e0201123.

de la O Serna,Jose Antonio, Van Moer, W., Barbe, K., Högskolan i Gävle, Akademin för teknik och miljö, Avdelningen för elektronik,matematik och naturvetenskap and Elektronik (2013) Using Alternating Kalman Filtering to Analyse Oscillometric Blood Pressure Waveforms. IEEE Transactions on Instrumentation and Measurement. 62 (10), pp.2621-2628.

Lee, S., Rajan, S., Jeon, G., Chang, J., Dajani, H.R. and Groza, V.Z. (2015; 2017) Oscillometric blood pressure estimation by combining nonparametric bootstrap with Gaussian mixture model. Computers in Biology and Medicine. 85 pp.112-124.

Forouzanfar, M., Ahmad, S., Batkin, I., Dajani, H.R., Groza, V.Z. and Bolic, M. (2015) Model-Based Mean Arterial Pressure Estimation Using Simultaneous Electrocardiogram and Oscillometric Blood Pressure Measurements. IEEE Transactions on Instrumentation and Measurement. 64 (9), pp.2443-2452.

Luo, N., Dai, W., Li, C., Zhou, Z., Lu, L., Poon, C.C.Y., Chen, S., Zhang, Y. and Zhao, N. (2016) Flexible Piezoresistive Sensor Patch Enabling Ultralow Power Cuffless Blood Pressure Measurement. Advanced Functional Materials. 26 (8), pp.1178-1187.

Sapiński A. Standard algorithm of blood-pressure measurement by the oscillometric method. Medical and Biological Engineering and Computing. 1992. November 1;30(6):671–

Lewis, P.S., Chapman, N., Chowienczyk, P. et al. Oscillometric measurement of blood pressure: a simplified explanation. A technical note on behalf of the British and Irish Hypertension Society. J Hum Hypertens 33, 349–351 (2019).

Lopez, S. (2012) Blood Pressure Monitor Fundamentals and Design. . (2), .

Zhe-Min , L., Cheng-Hung , C., Nai-Kuan , C. and Yuan-Hsiang , L. (2014) Bluetooth Low Energy (BLE) based blood pressure monitoring system. IEEE Explore.

Chandrasekhar, A., Yavarimanesh, M., Hahn, J., Sung, S., Chen, C., Cheng, H. and Mukkamala, R. (2019) Formulas to Explain Popular Oscillometric Blood Pressure Estimation Algorithms. Frontiers in Physiology. 10 pp.1415.

Chen,S.; Groza,V Z.; Bolic,M.; Dajani,H R.
Assessment of algorithms for oscillometric blood pressure measurement
IEEE Instrumentation and Measurement Technology Conference, 2009, 1763-1767, IEEE

Taylor–Fourier Analysis of Blood Pressure Oscillometric Waveforms Joś e Antonio de la O Serna, Senior Member, IEEE