Quick Doc:
Design of Wireless Blood Pressure Device

by

Doralice Ferreira

Electrical and Biomedical Engineering Design Project (4BI6)

Department of Electrical and Computer Engineering
McMaster University
Hamilton, Ontario, Canada

Quick Doc:
Design of Wireless Blood Pressure Device

by

Doralice Ferreira

Electrical and Biomedical Engineering
Faculty Advisor: Dr. Jim Reilly

Electrical and Biomedical Engineering Project Report
submitted in partial fulfillment of the degree of
Bachelor of Engineering

McMaster University
Hamilton, Ontario, Canada
April 27th, 2009

ABSTRACT

Today's unfortunate reality of an increasing number of patients coupled with the decreasing number of doctors within North America, ultimately manifests into intolerable waiting times in clinics and hospitals, alike. In attempt to maximize the resultant patient/doctor interaction, Quick Doc has been implemented to sequentially acquire a defined set of physiological measurements without the need of a practitioner. As such, an electrocardiogram signal, a blood pressure measurement and blood oxygen content are acquired and transmitted wirelessly to a base station where the data can be logged and further analyzed. Blood pressure specifically, is one of the most common and basic medical assessments. However, errors in measuring blood pressure with the auscultatory method often occur due to human error. This inaccuracy can be minimized by using the device's oscillometric method to detect oscillations within the cuff to determine the systolic, mean arterial, and diastolic pressures in real-time. Communication between the patient and the device is achieved via an LCD screen. The device's interface also prevents over inflation of the cuff and allows for immediate display of blood pressure measurements. Compatible results have been achieved with that of commercial blood pressure products. The theory behind the blood pressure device, hardware and software design, the experimental results, and efficacy of the device are presented.

Key words: blood pressure, systolic, diastolic, mean arterial pressure, oscillometric, auscultatory, oscillations

ACKNOWLEDGMENTS

Doralice Ferreira would like to acknowledge her parents Carlos Ferreira and Alice Ferreira for their endless love and support throughout her University career. They have offered her continuous words of wisdom and encouragement, particularly during her last year of education. Her brother Nelson Ferreira has also played an integral part in her success at McMaster University as he has given her the strength to continue when things seemed unmanagable.

Doralice would like to acknowledge Dr. Reilly for his guidance, support, counseling, and knowledge throughout the design and implementation of the blood pressure device. She would also like to thank Dr. Doyle for helping her to overcome her weakness and fear of programming. Furthermore, Jason Thong has offered his time and patience in helping her throughout the programming of the microcontroller. Also, Dr. Patriciu has been very helpful in the beginning stages of her design.

Doralice would like to acknowledge her group members Kundan Thind, Hamzah Qureshi, and Ashwin Ayyaswamy for spending time in the lab troubleshooting problems with the design and offering themselves as subject's for her results. She would also like to thank Janan Rajeevikaran and Ryan Pereira for supplying different hardware components and their assistance with the microcontroller. Finally, she would like to thank Jessica Torres for proofreading her final report.

TABLE OF CONTENTS

List of Tables

List of Figures

Nomenclature

A – Ampere

A/D – Analog to Digital Converter

AC – Alternating Current

AD – Analog to Digital Conveter

C – Capacitance

CBC – Canadian Broadcasting Corporation

CFP – College of Family Physicians

CO – Cardiac Output

CPU – Computer

CVP – Central Venous Pressure

DAQ – Data Acquisiton Unit

dB – Decibel

DP – Diastolic Blood Pressure

DC – Direct Current

ECG – Electrocardigram/Electrocardiograph

EN – Enable

F - Farad

Hg – Mercury

Hz – Hertz

k – Kilo-

kPa - Kilopascal

LCD – Liquid Crystal Display

LED – Light Emitting Diode

m – mili-

M – Mega-

MAP – Mean Arterial Pressure

mmHg – Millimeter of Mergury

µ – Micro-

s - Seconds

OSC – Oscillator

Ox – Oximeter

PIC – Microcontroller

R – Resistance

RS – Register Select

RW – Read/Write

Rx – Receiver

SP – Systolic Blood Pressure

SVR – Systemic Vascular Resistance

Tx – Transmitter

V – Voltage

Vs. – Versus

1 Introduction

1.1 Background

Today's unfortunate reality of an increasing number of patients coupled with the decreasing number of doctors within North America, ultimately manifests into intolerable waiting times in clinics and hospitals alike. This often discourages people from obtaining regular check-ups and deters them from receiving medical attention when it is most essential. Unfortunately, the privilege of free health care in Canada also has its setbacks when considering the prolonged wait times and the increasing number of those left without family practitioners. The Canadian Broadcasting Corporation (CBC) reports that in 2004, a number of organizations including: The College of Family Physicians (CFP), the Canadian Medical Association and the Royal College of Physicians and Surgeons collaborated on a National Physicians Survey which found that a family physician sees 117 patients per week on average [12]. In addition, family physicians work 70-80 hours per week and more than 70 per cent of family doctors provide some type of on-call service [9]. Furthermore, approximately 60 per cent of doctors do not regularly accept new patients while a research poll performed by the CFP of Canada in September of 2006 reported that 17% of Canadians do not have family physicians [10]. This equates to approximately five million Canadians without access to a family doctor with two of the five million having reported unsuccessful search attempts for a family practitioner.

Given the aforementioned statistics, it is evident that in North America and Canada specifically, health care professionals need to maximize their efficiency to attend to more patients per day. This can be achieved if the common tests performed by a practitioner on a normal visit could be done without their assistance. Not only would this make the most of the patient/doctor interaction by ensuring that the practitioner could focus on the patient's needs immediately, but this would also decrease waiting times in clinics and hospitals and increase the allowable number of visits per day. Therefore, in an attempt to maximize this resultant patient/doctor interaction, Quick Doc has been implemented to sequentially acquire a defined set of physiological measurements without the need of a practitioner. As such, an electrocardiogram signal, a blood pressure measurement and blood oxygen content are acquired and transmitted wirelessly to a base station where the data can be logged and further analyzed. Now vital information can be accessed by health care professionals so that they are able to set a course of action, make decisions or plan the next steps for their patients. However, it is crucial that these physiological measurements are able to be taken in the waiting room of a hospital or clinic but be displayed to the physician at their respective location. Unlike other biomedical measurement devices used in clinical settings today, Quick Doc is capable of acquiring multiple physiologically relevant signals and transmitting them wirelessly for further processing. As a group consisting of four members, there are different areas of responsibility that each will consider. I am responsible for the blood pressure monitoring component that will determine the blood pressure based on the oscillations within the pressure cuff. Ashwin Ayyaswamy is designing the electrocardiogram component with only two electrodes at the wrists. Hamzah Qureshi is responsible for the design of the pulse oximeter that will measure the oxygen content in the blood. Finally, Kundan Thind is responsible for processing each of the signals and displaying them accordingly.

1.2 Objectives

The objective pertaining to the design of Quick Doc is to ultimately implement a system that is able to decrease patient waiting time in a clinic while reducing the work load of the physician. This is accomplished by establishing an instrumentation system that is capable of acquiring three physiological signals including electrocardiogram, blood oxygen content and blood pressure. It is of utmost importance that the Quick Doc device takes the measurements in a non-invasive manner since the acquisition is to take place within a public setting. Also, because these measurements are taken in a different room from where the practitioner is likely to be, the Quick Doc system must be wireless. Although there are other medical devices that may take more than one of the aforementioned measurements at one time, such as the Combined Wrist Blood Pressure and ECG Monitors[15], a Blood Pressure and Pulse Oximeter[2], or a ECG/Pulse Oximeter[17]; there is no device that is capable of taking all three measurements simultaneously and wirelessly. As such, the ultimate vision for Quick Doc is a user-friendly device that can be easily implemented in a clinical setting and that allows patients to obtain these measurements on their own. The data for these vital signals can then be delivered to a base station for processing and better organization. This allows the practitioner, nurse or other health care personnel to analyze and diagnose the current health status of the patient prior to being attended.

An enhancement to Quick Doc given its wireless capabilities is to implement the system in a training or physical therapy setting. Many times in these environments, patients are connected to various electrodes which are connected to a main device via wires to monitor their activity. However, the immobility created due to the wires creates difficultly for the user and clinician alike to engage in certain exercises or activities without disconnecting or disabling their system. Therefore, the wireless capabilities of Quick Doc system would allow effective monitoring of said activities to gauge or mediate the training accordingly.

Ultimately, the aim of Quick Doc is to implement a system that is easy to employ without the assistance of a health care professional. The patient can now acquire their own ECG signal, blood oxygen content and arterial blood pressure within a few minutes and have it analyzed by a practitioner at the base station shortly after. Additionally, the blood pressure ratio, understood by most patients, is also displayed immediately to the user during acquisition via an LCD screen.

The importance of blood pressure often goes unrecognized even though high blood pressure is known as the silent killer [19]. High blood pressure stealthily damages the heart, blood vessels, kidneys and other organs with few or no symptoms. In fact, about half of all people with high blood pressure are completely unaware of it. Very occasionally, if blood pressure is exceptionally high, people may experience headaches, dizziness and blurred eyesight [20]. The term silent killer is especially used to reinforce that even if an individual in not experiencing any prior symptoms, they can suddenly experience a heart attack or stroke due to high blood pressure. Therefore, it is important to monitor blood pressure regularly so that signs of hypertension can be detected before it is too late.

Unlike hypertension, low blood pressure known as hypotension is pressure so low it causes symptoms or signs due to the low flow of blood through the arteries and veins. When the flow of blood is too low to deliver enough oxygen and nutrients to vital organs such as the brain, heart, and kidney, the organs do not function normally and may be permanently damaged. When there is insufficient blood pressure to deliver blood to the arteries that supply the heart's muscle with blood, a person can develop chest pain or even a heart attack [27].

For this reason, arterial blood pressure is one of the most common and basic medical assessments. However, errors in measuring blood pressure with conventional auscultatory methods (*cf.* 2.2 on page 12) often occur due to human error. Furthermore, these conventional methods require the skill and expertise of an additional person, commonly a health care professional. As such, the objective of the blood pressure component has been developed to minimize this inaccuracy and allow the patient to take their measurements single-handedly in a clinical or even home environment. This is achieved by using the oscillometric method (*cf.* 2.3 on page 13) to detect the oscillations within the cuff that occur from the pulsations of blood through the compressed artery. This device has been designed to be safe and easy to utilize with the effective display of instructions via the LCD screen. This interface provides a continuous line of communication between the user and the device to ensure accurate blood pressure measurements. This system is ideal for a clinic or hospital environment as it allows for information to be displayed to the user immediately and also provides the practitioner with the information at the base station. This way, information pertaining to the patient can be stored and at the same time, the patient can be aware of their current blood pressure. This eliminates the possibility of incorrect blood pressure readings due to patient motion because the patient has immediate access to the measurements. In the event that the pressure is unusually high or low, the patient can quickly take an additional reading to see if it corresponds with that of the previous. This simultaneous display of data to both parties prevents the practitioner from needing to acquire additional readings once they are with the patient.

Furthermore, performing a blood pressure measurement while the patient is relaxed eliminates the possibility of obtaining an error known as white coat hypertension [7].

This term is used to describe the increase in blood pressure measurement that results from the anxiety related to the examination performed by a health care professional. Blood pressure measurements taken in a doctor's office may not correctly characterize their typical blood pressure, resulting in a misdiagnosis of hypertension. Therefore, by allowing the patient to take their blood pressure at their own pace while in the waiting room of a clinic may reduce the patient's anxiety and increase the accuracy of their blood pressure reading accordingly.

1.3 General Approach to the Problem

The Quick Doc system consists of three separate circuit blocks that each acquires a physiological signal. As each signal is read from the patient it is connected to a transmitter via wires. This allows for the signals to be simultaneously transmitted wirelessly to a receiver. The receiver is connected to an amplification stage prior to being connected to the Data Acquisition Unit (DAQ). This post amplification of the signal is approximately 55 which is essential to have an interpretable signal at the output of the receiver. As with other transmission systems, the strength of the signal is diminished when it is wirelessly transmitted to the receiver. The Data Acquisition unit is used to acquire these signals which are imported into Matlab where the post processing of the data takes place. The computer will serve as the base station which the health care personnel can interact with to sequentially acquire the signals. Once the signals are acquired and processed, they are displayed to the practitioner after several minutes. A block diagram of the Quick Doc system outlining the aforementioned stages is shown in Figure 1.3.1 (*cf.* figure 1.3.1 on page 6). It is important to note that the signals are acquired sequentially and not all at once as presumed from the block diagram.

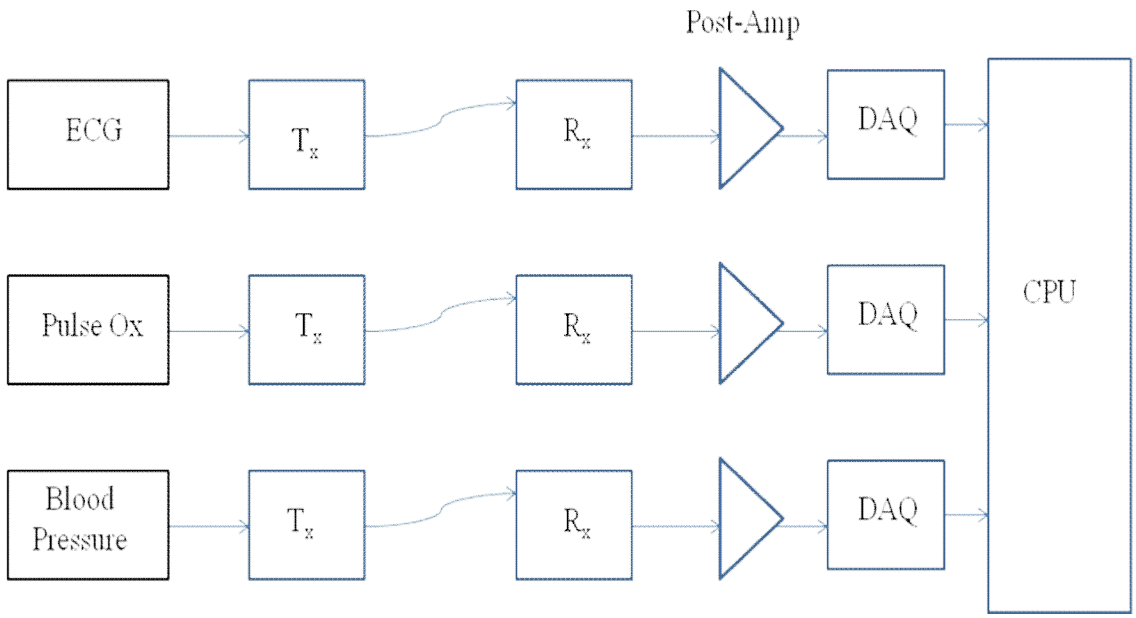


Figure 1.3.1 Block Diagram of Quick Doc. T_x represents the transmitter, R_x represents the receiver, DAQ is the data acquisition unit and the CPU is the base station. [18]

The blood pressure component of Quick Doc is focused around the oscillometric method. This oscillometric technique is based on the idea that the pumping of blood through the arteries by the heart causes the arteries to flex. These pressure variations will pass from the artery through the arm with attenuation and into the pressure cuff itself. While these pressure variations are small compared to the typical pressure applied by the cuff, they are nevertheless detectable by a pressure transducer. The voltage signal obtained by the sensor is filtered to extract the AC component and further amplified to produce interpretable oscillations. Essentially, this creates two distinct signals; the raw pressure in the cuff and the amplified peaks from this pressure, respectively. The raw pressure cuff signal and the filtered signal are digitized using the analog to digital converter pins of the microcontroller. This allows for sampling of the signals at the necessary times to essentially map the occurrence of the amplified oscillations to the respective pressure in the cuff. Ultimately, the microcontroller allows for the detection of peaks through the defined sequence of algorithms and computes the respective pressures. The measurements are displayed on an LCD screen allowing for immediate results and communication with the user throughout the full monitoring process.

1.4 Scope of the Project

There are several limitations concerning the Quick Doc system that should be assessed. Although the system is ideal given a clinical setting, it may not work as efficiently in a hospital setting due to the wireless range of approximately 500 feet indoors. In a clinic, this range is sufficient to span the waiting room to the practitioner's location. However, in a hospital where there are countless rooms and various floors, it may be unfeasible to acquire the physiological signals with the Quick Doc system. However, to eliminate this drawback, the base stations could be placed in certain areas of the hospitals within 500 feet of the waiting rooms to accommodate the range of the wireless transmission.

In addition, the Quick Doc system is not able to acquire the physiological signals all at one time. Due to the nature of the measurements, it would not be possible to take a patient's oxygen saturation simultaneously with blood pressure. The inflation of the cuff would cause occlusion of the artery and thus less perfusion to the hand, where the pulse oximeter acquires its signal. However, even if it were possible to acquire the physiological signals at once, the post processing at the base station has only been programmed to acquire the signals sequentially. Furthermore, a switch has not been implemented into the system to allow the transmitter to switch inputs depending on the signal that needs to be acquired. The output of the circuit block needs to be manually changed to connect to the input of the transmitter. As such, there was no need to have a timer to acquire the different signals sequentially. If a timer had been implemented into the system, then it would acquire the signals automatically with one touch of a button as opposed to requiring desired times to take each individual measurement.

There are also a few limitations concerning the blood pressure component of Quick Doc that necessitate consideration. As it is now, the design employs a manual pump to inflate the pressure cuff. Although an automatic pump may be more convenient for the user, issues of availability contributed to the said limitation. The pumps that were accessible were too large and required too high of a voltage to operate. In order to have a user-friendly device that is compact in size, it was not achievable with a large inflation pump regularly used for inflating larger objects (ie. mattresses, tires, etc.). The lack of a manual pump was the primary motivation behind the LCD screen displaying information to the user. Rather than using the LCD screen to solely display the blood pressure measurements, it now plays an integral part in the interaction between the patient and the device. Any apprehensions the user may have with using a manual pump can now be eliminated given the instructions that guide the user through the procedure. Not only does this provide the user with confidence to take their blood pressure measurement but it also serves as a safety precaution to prevent over inflation of the cuff.

Additionally, the pressure cuff should be tightened around the upper arm over the brachial artery, just above the elbow. This ensures greater accuracy of the blood pressure, since there are greater vibrations of the artery in this area. It is not recommended the cuff be placed around the wrist, as it is large in size and will be susceptible to greater motion artifacts caused by the movement of the hand or wrist. Likewise, the blood pressure device is intended to acquire measurements while the patient is stationary and not actively moving their arms or touching the pressure cuff during the process of deflation. As such, using this device in rehabilitation or physiotherapy settings would be quite difficult as movement is unavoidable. Furthermore, the device instructs the user to stop inflating at 170mmHg, and therefore does not take into account high blood pressure that may result from exercise or exhaustion. A high blood pressure measurement would still result in this case, but may not be an exact representation of the blood pressure exerted on the arterial walls.

In addition, when measuring blood pressure with this device, an accurate reading requires that the user not drink coffee, smoke cigarettes, or engage in strenuous exercise for approximately 30 minutes before taking the reading. Also, a full bladder may have a small effect on blood pressure readings, so if the urge to urinate exists, the patient should do so before the reading [41]. For best results the patient should sit upright in a chair with their feet flat on the floor and with limbs uncrossed to prevent incorrect readings due to motion as aforementioned. During the reading, the arm that is used should be relaxed and kept at heart level.

By default, the blood pressure pump deflates at a rate of 3mmHg/sec once the user is not actively pumping. This allows for a constant deflation rate and thus constant sampling of the data amongst users. If the user was responsible for releasing the valve, it is evident that the level of which it is released would vary from patient to patient. On account of the patients not being practitioners themselves, they would not have an understanding of the appropriate level at which to open the valve. If the valve is opened too much, the air within the cuff will be released at a rate greater than 3mmHg/sec, and the measurement will not be as accurate. As such, the user is instructed to stop inflating, but is not instructed to release the valve. However, in the event that the patient is uncomfortable with the pressure placed on the artery, they have the option to open the valve to discharge the air.

Also, because the instructions to the user are presented in the form of words, illiterate patients should be assisted by a health care professional. Attempts were made in attaching a buzzer or alarm device to the monitor that would sound to notify the user to cease inflation of the cuff however, it was not possible to do so given that it was out of stock and no longer available for shipment due to the lead component it contained. In the event that a buzzer is implemented into the system, it should to be high enough for the patient to hear but not overly high as to shock or alarm the patient. An overly alarming sound could potentially cause sudden motion of the arm or a temporary increase in the heart rate and blood pressure accordingly.

In addition, the instructions are written in English and thus only understood by those who read the language. However, multiculturalism in Canada demands accessibility to other languages just as readily as English. Therefore, a limitation of this design is that there is only an option for one language. By placing a button or a switch, there can be an option for various languages that can be presented to the user instructing them to initiate and cease the inflation of the pressure cuff respectively.

2 Literature Review

2.1 Blood Pressure

Pressure is the force per unit area and is used to describe fluids in a gaseous or liquid form. At rest, fluid pressure is transmitted equally to all its parts and, at any one point, is the same in all directions [32]. Pressure plays a significant role in the human circulatory system. Blood pressure is the pressure that is exerted by blood against the walls of the arteries as it travels from the heart to all parts of the body through the systemic circulatory system [14]. The force needed to circulate this blood is measured in millimeters of mercury (mmHg) because the traditional measuring device, called a sphygmomanometer, uses a glass column that is filled with mercury (whose chemical symbol is Hg) and is marked in millimeters [42].

Blood pressure reflects both how strongly the heart is working and what condition the arteries are in. Blood pressure is a function of cardiac output and systemic vascular resistance, specifically the multiplication of the two variables (*cf.* Appedix A 7.1.1 on page 44). The cardiac output is the amount of blood the heart pumps per minute while the systemic vascular resistance is the pressure the walls of the arteries exert on the flowing blood [3]. As blood pushes into the arteries with each heartbeat, it forces the artery walls to expand, naturally returning to its original shape as the blood flow recedes. Therefore, it is expected that as the vessels' flexibility decrease or their stiffness increase, that arterial resistance will increase. Similarly, an increase in the vessel's narrowness or tightness will result in an increase of blood pressure at any level of flow. Likewise, as cardiac output or arterial resistance increases, blood pressure increases accordingly.

Blood pressure is commonly referred to in terms of systolic and diastolic pressure. Systolic pressure is the peak pressure reached in the arteries, occurring near the beginning of the cardiac cycle when the ventricles are contracting [28]. Diastolic pressure on the other hand refers to the minimum pressure in the arteries which occurs near the end of the cardiac cycle when the ventricles are filled with blood [26]. As such, blood pressure is always given as two values: systolic and diastolic pressures, written as the numerator and denominator respectively.

The exact value obtained for this ratio of systolic over diastolic is not as significant as the range of values within this ratio is found. For instance, normal blood pressure ranges from 90 – 119mmHg for systolic and 60-70mmHg for diastolic. However, if both values are significantly less than 90 or 60mmHg for systolic and diastolic respectively, then the patient may have hypotension (abnormally low blood pressure). On the other hand, if the systolic and diastolic blood pressures are over 120mmHg and 80mmHg respectively, then the patient may have hypertension (abnormally high blood pressure). It is important to note that there are several stages of hypertension: prehypertension has a range of 120-119mmHg, stage 1 hypertension extends from 140-159mmHg, and stage 2 hypertension considers systolic pressures greater than 160mmHg. [36]

A significant component of the blood pressure measurement that is often overlooked is the mean arterial pressure (MAP). Mean arterial pressure is a function of cardiac output, systemic vascular resistance and central venous pressure (*cf.* Appedix A 7.1.2 on page 44). At normal resting heart rates, the mean arterial pressure can be approximated as a function of systolic and diastolic pressures. The relationship between the blood pressures is approximated as $MAP \simeq DP + \frac{1}{3}(SP - DP)$ (*cf.* Appedix A 7.1.2 on page 44) however, at high heart rates the MAP is more closely approximated by the arithmetic mean of systolic and diastolic pressures due to the change in shape of the function of the arterial pressure. MAP is considered to be the perfusion pressure seen by organs within the body. If the MAP is greater than 60 mmHg, it is enough to sustain the organs of the average person. Conversely, if the pressure is significantly below the value, the end organ will not get enough blood flow and will become ischemic [33]. Therefore, it is important to get a measurement for mean arterial pressure in addition to the standard systolic and diastolic pressures.

2.2 Evolution of Blood Pressure Measurement

The first observed instance of blood pressure measurement occurred in 1733. The procedure consisted of the insertion of a brass pipe, which was connected to a glass tube, into an artery. It was not until 1847 that human blood pressure was recorded using Carl Ludwig's kymograph with catheters inserted directly into the artery. Though an improvement, both methods were very invasive and not appropriate for a clinical setting. Shortly after in 1855, Karl Vierordt made an important discovery that would forever change the way blood pressure was to be measured. Vierordt discovered that with enough pressure, arterial pulse could be obliterated and did so through the use of an inflatable cuff placed around the arm to constrict the artery. In 1881, Samuel Siegfried Karl Ritter von Basch invented the sphygmomanometer, which consisted of a water-filled bag connected to a manometer. The manometer was used to determine the pressure required to obliterate the arterial pulse. Later in 1896, Scipione Riva-Rocci developed the mercury sphygmomanometer. An inflatable cuff was placed over the upper arm to constrict the brachial artery. This cuff was connected to a glass manometer filled with mercury to measure the pressure exerted onto the arm. However, this sphygmomanometer could only be used to determine the systolic blood pressure. It wasn't until 1905 that Nikolai Korotkoff first observed the sounds created by the constriction of the artery. Korotkoff found that there were characteristic sounds at certain points in the inflation and deflation of the cuff. These sounds, labeled Korotkoff sounds [40] after their discoverer, were created as a result of the abnormal passage of blood through the artery corresponding to systolic and diastolic blood pressures. This discovery thus gave rise to the auscultatory method that utilizes a stethoscope and a sphygmomanometer to listen for the korotkoff sounds of blood flowing through the artery. [6]

This conventional method of measuring blood pressure requires a cuff to be placed around the upper arm at roughly the same vertical height as the heart, and inflated manually until the artery is completely occluded [1]. Listening with the stethoscope to the brachial artery at the elbow, the practitioner slowly releases the pressure in the cuff. When blood just starts to flow in the artery, the turbulent flow creates a pounding sound known as the first Korotkoff sound [40]. The pressure at which this sound is first heard is measured as the systolic blood pressure. The cuff pressure is further released until no sound can be heard. The pressure at which the sound ceases is referred to as the diastolic arterial pressure. [4]

Further advancements in the detection of blood pressure have been made with the advent of the oscillometric method.

2.3 Oscillometric Method

Inevitably, inconsistencies in the measuring of the various blood pressures with the auscultatory method often occur as a result of human error. This inaccuracy can be minimized by using an oscillometric method to detect oscillations within the cuff [5] to determine the systolic, mean arterial and diastolic pressures accordingly. The oscillometric method predates the method of Korotkoff but was not originally as popular. However, it is now the standard method for automated blood pressure measurement.

In 1885 the French physiologist Marey observed that, if he placed a patient's arm in a pressure chamber then the pressure of the chamber would fluctuate with the pulse and the magnitude of the fluctuation would vary with the pressure of the chamber. The oscillometric method is based on the principle that these fluctuations correspond to the occluding effect on the artery of pressure applied uniformly to the arm and that the same effect can be observed in the pressure of an occluding cuff [29]. As such, when blood flow is present but restricted, the cuff pressure monitored by a pressure sensor will vary periodically in synchrony with the cyclic expansion and contraction of the brachial artery, producing oscillations. These oscillations are small compared to the typical pressure applied by the cuff because the variations in pressure need to pass from the artery through the arm of the patient with attenuation into the cuff where it is being measured. Nevertheless, they are detectable by the pressure transducer which provides a measurable and interpretable output. As such, the values of systolic, diastolic, and mean arterial pressure can be computed using defined algorithms. This ultimately eliminates the dependence on human skill and judgement as well as provides a reading of mean arterial pressure during the cardiac cycle that was otherwise impossible to obtain with conventional methods.

3 Statement of Problem and Methodology of Solution

The blood pressure device includes an inflatable cuff that is wrapped around the upper arm. Once the cuff is comfortably adjusted, the patient can turn on the device to read instructions from the LCD screen. Upon power-up, the LCD screen will display “START INFLATING”. As the cuff if manually inflated by the user, the cuff pressure and thus the pressure applied to the arm of the patient increases. The pressure is applied to the brachial artery of the upper arm, just above the elbow. During this process of manual inflation, the LCD screen instructs the user to “KEEP INFLATING” all the while displaying the current cuff pressure in real-time. If the pressure applied to the cuff is increased beyond the highest blood pressure level in the artery beneath the cuff, the artery will be forced closed. The blood pressure in the brachial artery is not constant, but varies with time in relation to the beating of the heart. Following a contraction of the heart to pump blood through the circulatory system, the blood pressure increases to a maximum level, known as the systolic pressure. When the pressure in the cuff is inflated above this level, the brachial artery is permanently closed and the patient can terminate the inflation. However, the pressure at which this occlusion of the artery occurs varies from person to person. Therefore, a pressure of 170mmHg is high enough to ensure that the majority of patients using the device will have their artery completely occluded. In addition however, 170mmHg is still remains low enough as to not cause any damage to the artery or harm to the patient. Therefore, to ensure that inflation is stopped at this safe and practical pressure of 170mmHg, the LCD screen instructs the user to “STOP INFLATING.” As the pressure in the cuff is reduced from 170mmHg to below the systolic pressure, the brachial artery beneath the cuff will begin to open and close with each heart beat as the blood pressure level first exceeds the cuff pressure and then continues to fall below the cuff pressure. The arterial walls act in a non-linear fashion with respect to the blood pressure level. Thus, as the blood pressure exceeds the cuff pressure, the artery will snap open and produce measurable oscillations in the cuff. These oscillations are detected by the pressure sensor which can then be extracted and amplified with a high pass filter and amplifier respectively. As the cuff pressure is reduced further, the cuff pressure will be brought below the mean arterial pressure and ultimately, the diastolic pressure level. At this level, the brachial artery beneath the cuff remains open throughout the heart cycle. As such, oscillations in the cuff should be visibly reduced at the output of the pressure sensor.

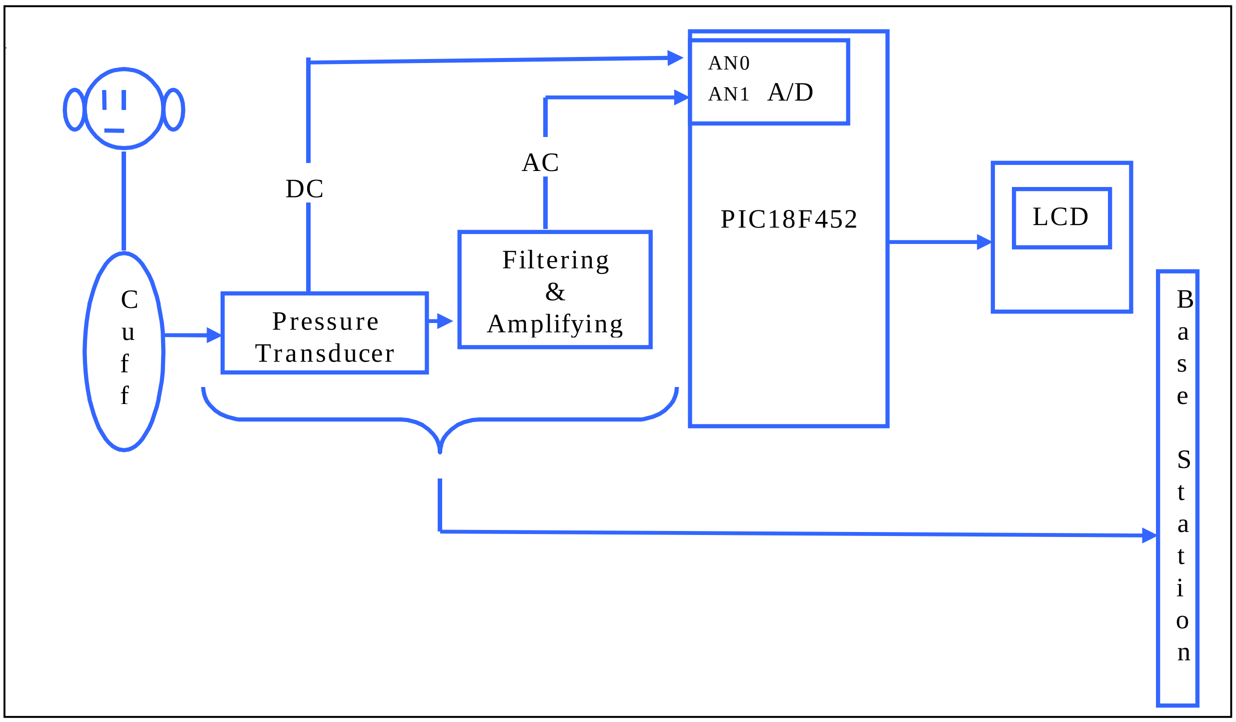


Figure 3.1: Schematic Block Diagram of blood pressure device ([29],[34],[38],[43])

A schematic block diagram of the blood pressure device employing the design is presented in Figure 3.1 (*cf.* figure 3.1 on page 15). The blood pressure monitoring system includes an inflatable cuff that is adapted to be wrapped around the arm of a patient whose blood pressure is to be determined. The cuff is connected to the electronic circuitry via an unobstructed air tube. The air tube is sealed to the input of a pressure transducer that will continuously measure the pressure inside the cuff. As such, the maximum pressure of 170mmHg can be detected and dealt with accordingly so that over inflation of the cuff is prevented. Furthermore, the pressure transducer essentially detects the oscillations that occur within the cuff once it is deflated from this maximum pressure.

The output of the sensor, in voltage, is split into two paths for two different purposes. One is used as the cuff pressure while the other is further processed by a high pass filter and amplifier to represent a DC and AC signal respectively. Since the cuff pressure signal is conditioned by the internal op–amp, the cuff pressure can be directly interfaced with the analog–to–digital converter for digitization. The other path will filter and amplify the raw cuff pressure signal to extract an amplified version of the oscillations which are caused by the expansion of the patient's arm each time pressure in the arm increases during the cardiac cycle. This extracted AC signal is also interfaced with the analog-to digital converter in the microcontroller. As such, the filtered signal of the peaks can be mapped back to the original pressure sensor signal to yield the systolic, mean arterial and diastolic pressures. Once the microcontroller has performed the necessary algorithms and calculations, the measurements are displayed to the patient via an LCD screen. In addition to the real-time measurements of the blood pressure, the signals from the electronic circuitry are also transmitted wirelessly to a receiver. The data acquisition unit acquires the signals and exports them to the base station for further processing.

Furthermore, a theoretical model for determining the systolic, mean arterial and diastolic pressures was developed and implemented into the algorithm in the microcontroller. However, difficulties arose when attempts were made to translate theory into practice. This intricacy was illustrated in determining the blood pressures using C programming language. Theoretically, the onset of the oscillations should occur at systolic pressure and the disappearance of the oscillations should occur at diastolic pressure. However, due to the extreme sensitivity of the pressure transducer, the onset of oscillations actually occurs well above systolic pressure and the oscillations do not disappear until well below diastolic pressure. Therefore, a different model was created to account for this discrepancy. As aforementioned, the filtered and amplified version of the cuff pressure signal allows for the detection of oscillation peaks that occur while the blood resumes flow through the artery. These peaks are then mapped back to the original cuff pressure signal to determine the series of pressures. The occurrence of the systolic pressure is considered only once the peaks from the filtered signal surpass a predefined reference voltage to ignore any minor peaks that occur due to noise. The peaks that exceed this reference voltage are then counted until a predefined number of oscillations have been achieved. Once these two conditions have been satisfied, the cuff pressure signal is sampled and the respective pressure is displayed on the LCD. Subsequently, the mean arterial pressure is determined by employing an insertion sort algorithm to find the three largest peaks from the filtered signal and take an average of their respective cuff pressures. Finally, the diastolic pressure is calculated using the relationship outlined in Appendix A (*cf.* 7.2.5 on page 45). Ultimately, this approach allows for greater accuracy than the theoretical method suggested.

4 Experimental or Design Procedures

4.1 Hardware

4.1.1 Pressure Detection

The pressure transducer used to convert the pressure in the cuff into a voltage is the MPX5050GP [44]. Similar to other pressure sensors, it employs the piezoresistive principle to convert pressure to an electrical signal. A silicon chip is micro-machined to give a diaphragm around which four resistors are diffused in a bridge configuration. A fluorosilicone gel, which protects the die from harsh media, isolates the die surface and wire bonds from the environment while allowing the pressure signal to be transmitted to the sensor diaphragm. Application of pressure to the diaphragm results in a change in the value of these resistors creating a high level analog output signal that is proportional to the applied pressure [22]. This pressure sensor is ideal given that it has been designed specifically for the pressure media to be dry air and because it has a sufficient pressure range of 0 kPa to 50 kPa corresponding to 0 to 376mmHg [44]. At most, the patient will only need to inflate the cuff to approximately 170mmHg which is well within the limits of this pressure sensor. The pressure is converted into a voltage output within the sensor given the linear relationship outlined in Figure 4.1.1.1 (*cf.* figure 4.1.1.1 on page 18).

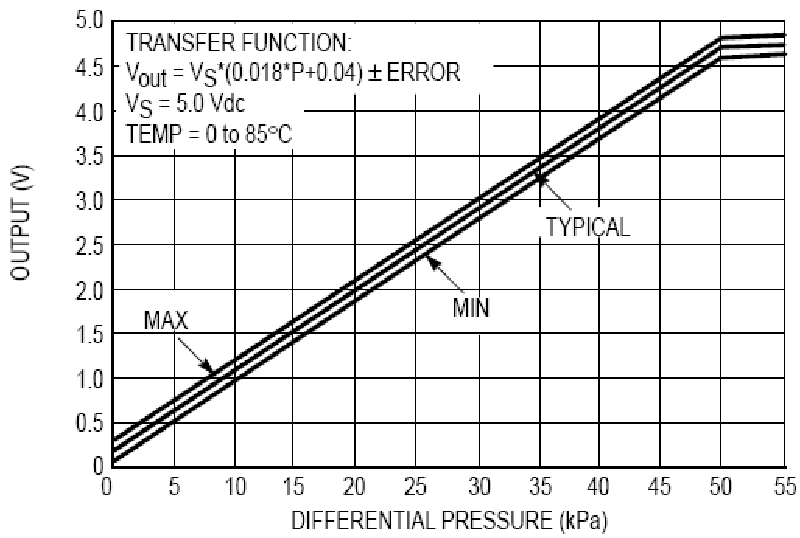


Figure 4.1.1.1 Voltage verses Pressure Relationship [44]

The output voltage of the pressure sensor given a 5V input ranges from 0 to 4.75V, which is an ideal range for the input of the analog to digital (A/D) converter. Moreover, the pressure transducer is already signal-conditioned by its internal operational amplifier and thus can be directly interfaced with an analog to digital converter for digitization [44]. In addition to the cuff pressure signal serving as an input to the microcontroller's A/D converter, it must also undergo filtering and amplification to extract the oscillations.

4.1.2 Filtering and Amplification

As realized in Figure 4.1.2.1 (*cf.* figure 4.1.2.1 on page 19) it is clear that there are two pieces of information in the pressure signal: the underlying pressure to which the cuff has been inflated or deflated and the oscillations present in the signal. The underlying signal is a low frequency signal comparable to DC and the fluctuating signal that is cardiac synchronous which is AC.

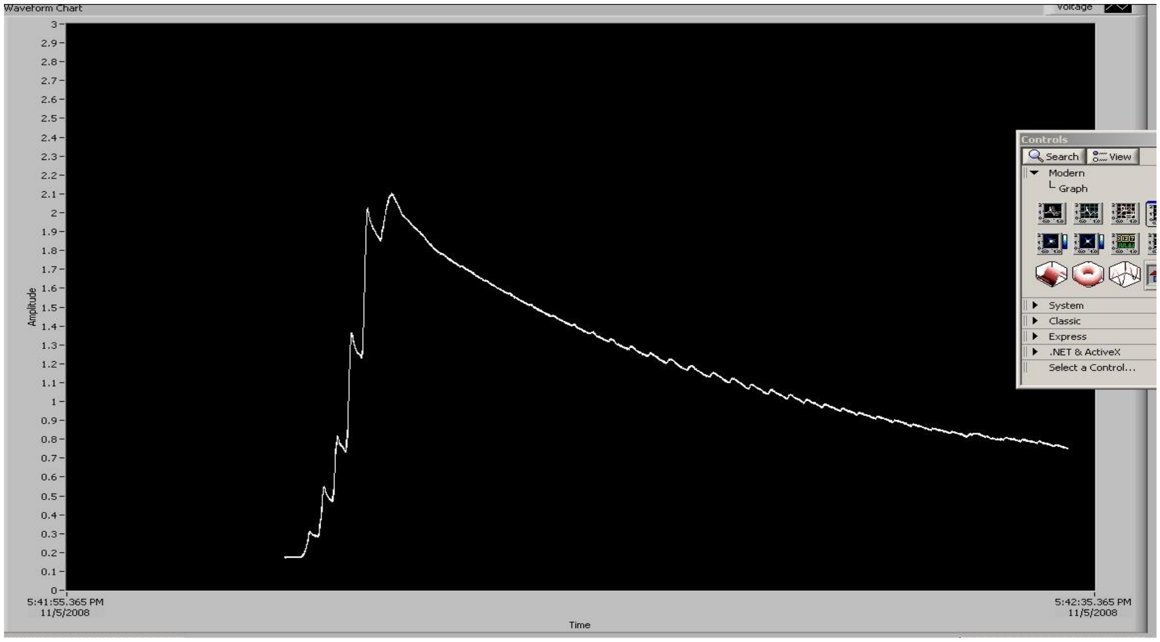


Figure 4.1.2.1: Cuff Pressure Signal

The AC signal, which is oscillating at approximately 1Hz or 60 beats/minute, is extracted from the raw pressure signal (0-0.04Hz) with a two-pole high pass filter. By selecting a cutoff frequency higher than 0.04Hz, it will completely block the low frequencies or DC allowing for a constant baseline of each oscillation allotting the same reference for comparison. As such, the two-pole high pass filter has one cutoff of 0.338 Hz which accommodates for slower heart rates of approximately 20 beats/second. In addition, there is a second cutoff of 0.386 Hz that has been chosen to ensure that the oscillation signal is not distorted nor lost (*cf.* figure 7.2.1 on page 44). Although this would only be possible if the cutoff was lower than the fastest potential heart rate.

The two-pole filter offers higher contrast or sharper cut-off from its passband to stopband, as opposed to the single pole filter. As such, the amplitude of the filter at one tenth the 3 dB frequency is attenuated a total of 40 dB as opposed to 20 dB. The 40dB drop is the filter response rolling-down rate at the passband edge. The 2-pole filter presents a cleaner cut-off of unwanted frequency components without having those frequency components at the passband edge distorted. In addition, the initial roll-off in the region above the 3 dB frequency starts later for a two pole filter than it does for a single pole filter. Therefore, the two pole filter has a flatter response at a higher frequency than that of a single pole filter [13]. The phase response of the two-pole high pass filter doubles as well. This results in the change in phase of the two-pole filter to be 90 degrees at the 3 dB frequency and ultimately approaches 180 degrees at infinity. Whereas the change in phase of a single pole filter is 45 degrees at the 3 dB frequency and approaches 90 degrees at infinity [39]. In general, two-pole filters offer higher contrast at the pass-/stop-band edge. However, it is important to note that more fluctuations or ripples appear inside the passband which may bring in distortions to the signal to be retained. For the purpose of this design, the possibility of slight ripples is not significant. What is important is that the two-pole filter employed is closer to an ideal filter at the edge, allowing for greater attenuation of the lower frequencies.

The two-pole high pass filter consists of two RC networks which determine the two aforementioned cutoff frequencies. The two cutoff frequencies can be approximated by the following two equations with their respective calculations in *Appendix A*
(*cf.* figure 7.2.1 on page 44). In addition to the filter, the amplifier provides a high gain to

$$f_1 = \frac{1}{2\pi R1C1} \qquad f_2 = \frac{1}{2\pi R3C2}$$

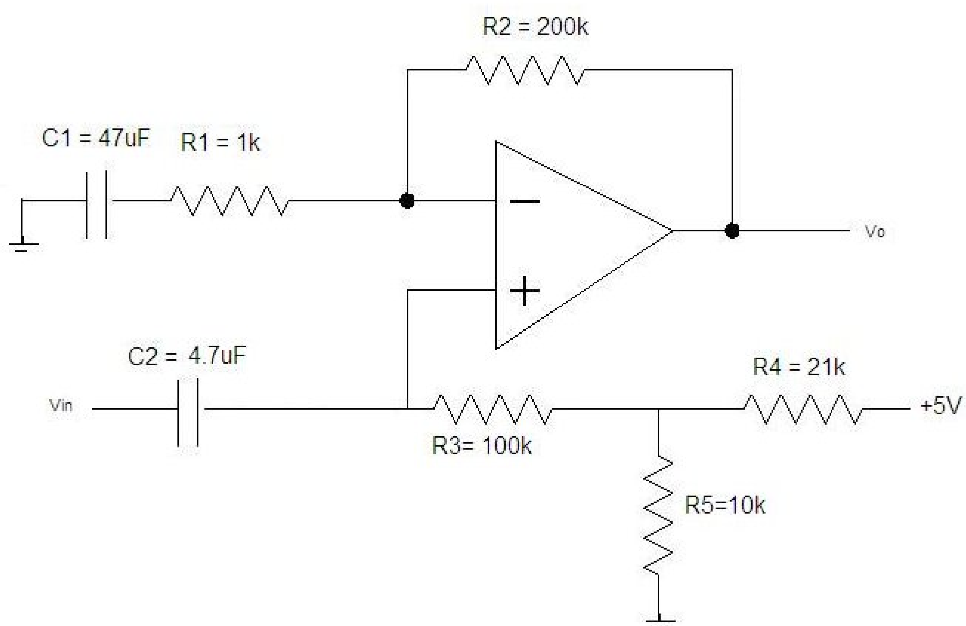


Figure 4.1.2.2 Two-pole High Pass Filter and Amplifer [8]

the extracted AC signal to create defined peaks at a measurable voltage for the input of the analog to digital converter. The oscillations that occur are very small in amplitude, varying from 1mmHg - 3 mmHg or 12mV to 36mV respectively, from person to person. As such, an amplifier with a gain of over 100 was used to amplify the oscillation signal to about 5mV to 3V, which is within the output limit of the amplifier [8].

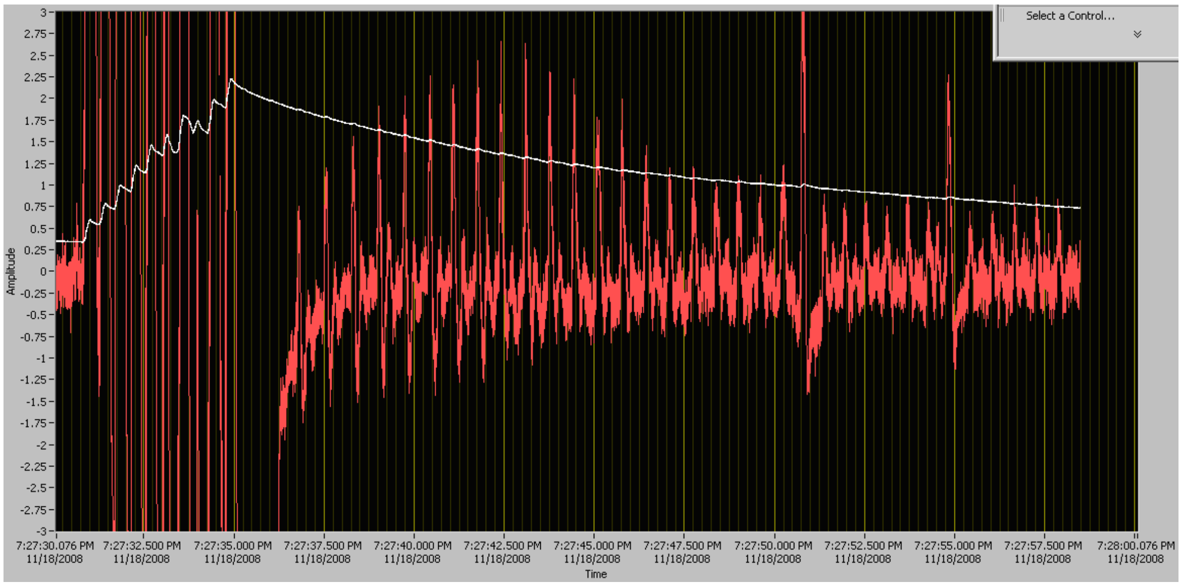


Figure 4.1.2.3: Cuff Pressure Signal and Extracted Signal using two-pole Highpass Filter and Amplifier

The HA17741 is a frequency compensated operational amplifier that was used to design the active filter. It has an allowable Vcc and Vee of +18V and -18V, as well as an allowable input voltage of ± 15V [16]. This is ideal since the input voltages, Vcc and Vee employed fall within the range of ± 5V. In addition, the operational amplifier allows for a high voltage gain which is necessary to avoid saturation of the amplifier.

Alternative filters could have also been designed using this operational amplifier, such as a single pole high pass or a band pass filter with a lower cutoff of around 0.5 Hz (*cf.* figure 7.3.1 on page 45). In experimenting with these alternative filter deigns, it was found that the oscillations were successfully extracted from the DC component of the signal. However, the filtered oscillations were not as highly defined as the two-pole active high pass filter employed in this design. It is imperative that the extracted oscillations be distinct and large enough to distinguish one from the other.

4.1.3 Analog to Digital Conversion

Once the pressure signal is filtered and amplified, both the raw and processed signals are sent into the PIC Microcontroller for further analysis. The microcontroller will need to correlate pressure at any given moment in time with that of the fluctuations at a specific point in time. In order for the microcontroller to be able to analyze any incoming voltages, the analog signals must be converted to digital values using the 10-bit Analog-to-Digital Converter which is featured in the PIC18F452. The positive and negative reference voltages at 0V and 5V were defined giving a minimum converted value of 0 and a maximum value of 1024 binary, respectively. The conversion from analog to digital must be achieved with sufficient accuracy and at a sufficient rate to capture the salient features. The microprocessor will need to correlate pressure at any given moment with the location of an oscillation at that point in time. Consequently, the pins AN0 and AN1 of the A/D converters on the microprocessor run at the same rate and thus take samples at the same time. The fastest possible data acquisition time that the ADC would theoretically allow was 1.6μs, which equated to a maximum sampling rate of 625 kHz (*cf.* figure 7.2.2 on page 44). As such, a data conversion rate of 12.5μs allowing for a sampling rate of 80 kHz was employed, given that the crystal oscillator is 16MHz.

4.1.4 Microcontroller

Ultimately, the PIC18F452 microprocessor is the heart of the blood pressure device. This runs all the necessary algorithms to interpret the two signals. As aforementioned, this microcontroller is ideal in that it already has a built-in 10-bit ADC converter, providing a higher resolution in comparison to the standard 8-bit ADC converters. Furthermore, it has 40 pins in total with four different ports, which allowed sufficient number of ports for the simultaneous configuration of the LCD screen and separate LEDS used to test the output of the ADC converter during the prototyping stages.

The microcontroller requires certain support hardware to allow it to function. A crystal oscillator is an electronic circuit that uses the mechanical resonance of a vibrating crystal of piezoelectric material to create an electrical signal with a very precise frequency. This frequency of 16MHz provides the external portion of the oscillator function for clocking the microcontroller while providing a stable base for time based functions. This microcontroller is capable of operating in eight different oscillator modes depending on the crystal used [31]. Connecting the 16MHz crystal to the OSC1 and OSC2 pins established a high speed crystal/resonator mode. As seen in Figure 4.1.4.1 (*cf.* figure 4.1.4.1 on page 23), it is clear that the crystal oscillator cannot be connected to ground directly and must have one capacitor at each end. The configuration uses two capacitors of capacitance 1uF and 560pF specifically, although other values of approximately the same range would also allow the crystal oscillator to operate successfully.

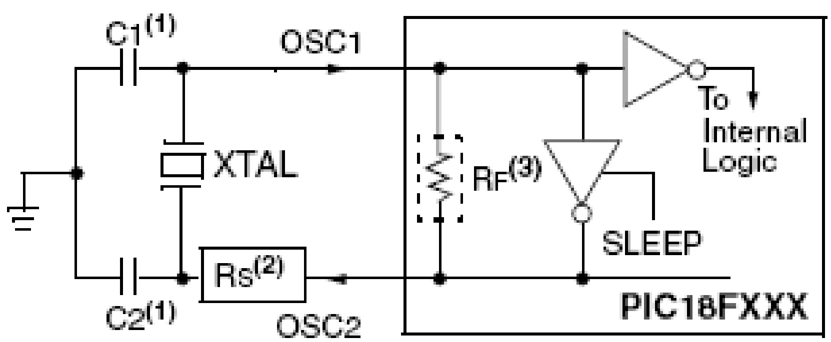


Figure 4.1.4.1 Crystal Operation Configuration [31]

In addition, the PC 1602-L 16x2 Character Display [30] was connected to Port B and Port D of the microcontroller.

4.1.5 LCD Display

The Hitachi HD44780 LCD is a dot-matrix liquid crystal display controller and driver already built into the PC 1602-L character display. The HD44780 receives data from an external source, the PIC18F452 microcontroller, and communicates directly with the LCD. It has been configured to drive a dot-matrix liquid crystal display under the control of this 8-bit microprocessor [21]. The timing requirements of the HD44780 are difficult to violate with the PIC18F452 running at maximum clock. Both the PIC18F452 and the

HD44780 need to come out of reset when power is turned on. Therefore, if the PIC18F452 is slightly faster at coming out of reset than the HD44780, the HD may miss instructions sent to it by the PIC. As such, a delay of 15 milliseconds after PIC reset is usually sufficient to guarantee that the HD44780 is also running [25].

In addition, the low power supply (2.7V to 5.5V) of the HD44780 is suitable for this design since the power supply is not expected to be greater than 9V. Since the LCD is a 16x2 display this corresponds to a display of up to two 16-character lines, which is sufficient to communicate instructions to the user and legibly display the blood pressure measurements [155]. However, since the LCD is a two line display, the systolic pressure measurement is displayed first and then the mean arterial pressure and the diastolic pressure are displayed together on the first line and second line of the LCD respectively.

The HD44780 standard requires three control lines as well as eight I/O lines for the data bus. The LCD is operated with an 8-bit data bus which requires a total of eleven data lines: three control lines plus the eight lines for the data bus. The three control lines are referred to as EN, RS, and RW. The EN is known as the Enable line. This control line is used to tell the LCD that data is being sent. To ensure that data is sent successfully to the LCD, the enable line set to low (0) while the other two control lines put data on the data bus. When the other lines are completely ready, the enable to set to high (1) and a minimum delay of 15ms is used to eventually bring it low again. The RS line is the known as the Register Select line. When RS is low (0), the data is treated as a command or special instruction, such as clearing the screen, clearing a specific line or positioning the cursor at a desired location on the LCD. When RS is high (1), the data being sent is text data which should be displayed on the screen. The RW line is the Read/Write control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying or reading the LCD. The majority of the time the RW is used as a write command and therefore will normally be set to low. Finally, since the data bus consists of 8 lines as aforementioned, the lines are referred to as DB0, DB1, DB2, DB3, DB4, DB5, DB6, and DB7. ([21],[30]) All the appropriate connections were made from the microcontroller to all the data and power lines of the LCD module. Ports B and D specifically were used for the connections since Port A contains most of the Analog inputs and Port C contains the communications capabilities of the PIC. The Port D pins of the microcontroller were connected to DB0 - DB7 of the LCD as the connections for the data bus. The pins for power supply ground (Vss), power supply of 5V (Vdd), contrast adjust (Vo), register select signal (RS), data read/write (RW) and enable (E) of the LCD were connected to Port B pins on the microcontroller. For instance, B0 is used as the circuit on configuration, B1 is set as the enable, B2 is set as the read/write and finally B3 is set as the register select signal. Finally, pins VB1 and VB0 on the LCD are used as the backlight with 5V and the backlight ground respectively. Although low resistor values can be used to tie input only pins to reduce noise, this was not prepared in setting up the LCD. ([21],[30])

The L7805CV Voltage Regulator [23] was used to ensure that the input voltage was approximately 5V for the microcontroller, LCD display, operational amplifier, and the pressure transducer.

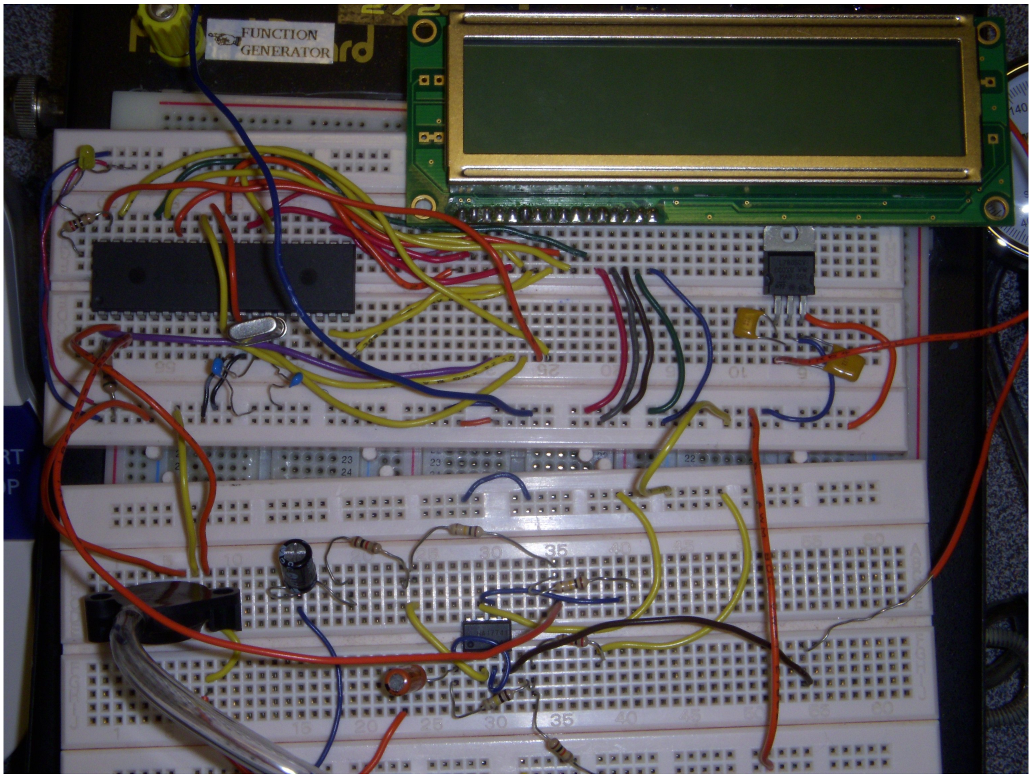


Figure 4.1: Prototype of Blood Pressure Device

4.2 Software

Microchip's MPLAB Integrated Development Environment was used alongside the MPLAB C18 C compiler to program the PIC18F452 microcontroller.

4.2.1 Analog to Digital Conversion

After the pressure transducer converts the pressure taken from a patient and sends the corresponding voltage into the microcontroller, the ADC converts this analog voltage value into a digital value that can be processed. There is a minimum required delay between data acquisitions for accurate conversions, as previously explained (*cf.* figure 4.1.3 on page 23). This delay is implemented in the code through a function call to the delay header file. After these calculations are made, the ADC can be initialized. The initial configuration consists of selecting the number of analog pins, input channel, A/D conversion clock, and turning on the ADC. Channel AN0 and AN1 were selected for the raw and processed input signals, respectively. If the input to the ADC_conv function is GRAPH1, then the OpenADC command is used for the AN0 pin to digitize the cuff pressure signal accordingly. However, if the input to the function is GRAPH2, then AN1 pin is used to digitize the filtered signal. In either case, there is a delay before the ConvertADC function is called to start the conversion from analog to digital. The value returned is the result from the ReadADC function call (*cf. Appendix B* 8.1 on page 51). One restriction is that the ADC can only acquire and convert one signal at a time and although the transition time to select a desired input was minimal, I still had to delegate at which times I wanted to sample a specific signal. Much of this process was recognized through the predefined functions included with MPLAB C18 Libraries.

4.2.2 LCD Display

In addition to the ADC, the LCD was also initialized and configured appropriately to serve as an interface for debugging the code as well providing a means of communication between the device and the user. Initializing the LCD was one of the greater challenges because it is very specific to timing and order. The LCD can be initialized either by the internal reset circuit or sending set of commands. For this design, the LCD has been initialized by instructions. Once the LCD is powered on, a delay of 15ms is needed after the Vdd rises to 4.5V. Then data/command is clocked in by taking E (RB1) from low to high and back low. Data is clocked on the falling edge of E.

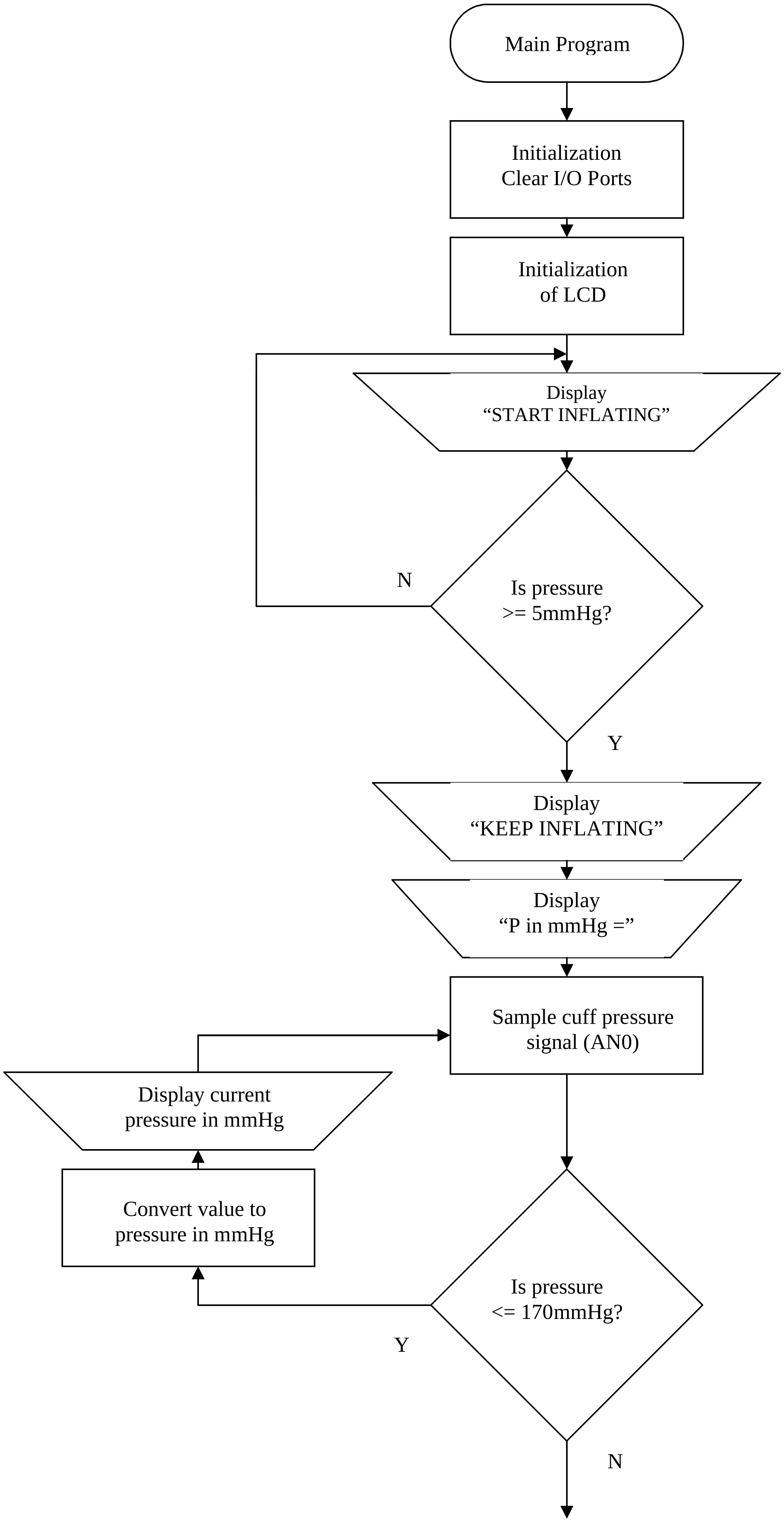
An internal reset circuit automatically initializes the HD44780 when the power is turned on. Then the following instructions are executed during the initialization [24]. To start, a delay of 15ms is used followed by setting pins DB4 and DB5 to high. The enable is then set to high and then low with a 15ms delay in between. This aforementioned sequence of events is repeated four times until the DB3 pin is also set to high to allow for the LCD to be set at two lines with 8-bits each. Again enable is set to high and then low with a 15ms interval as described earlier. Then to set the display on, the cursor on and the blinking of the cursor on, pins DB0, DB1, DB2 and DB3 are set to high followed by the enable of high to low [25] (*cf. Appendix B* 8.1 on page 52). This terminates the specific and rigorous initialization procedure of the LCD. (*cf.* 4.1.5 on page 23).

Various problems were encountered in initializing the LCD. The initial step was simply trying to get a blinking block cursor in the upper left hand corner. However, the majority of the time, there would be a block of rows or no display at all. This meant that it was not stepping through the source code properly as it was not initializing the LCD module, clearing the screen, nor setting the cursor to blink.

In addition to the initialization of the LCD, sub-programs to send a string or data to the LCD were also written with timing and delays in mind. The program LCD_Send_String takes in a character at a time and sends it through the LCD_Send_Data program so that the values can actually be displayed onto the LCD. Like with the initialization of the LCD, to be able to print a character onto the screen also requires a delay of 15ms and then setting the register select signal, so that a character can be written. This sequence of setting pins to high and low is required to write a line of characters, one by one. It is essential that the enable to set to high, followed by a 15ms delay and then set to low again (*cf. Appendix B* 8.1 on page 52). Depending on the oscillator used, this delay between the characters being written on the screen can become quite extensive and choppy. However, with the crystal oscillator used and the 5V input, the characters are written on the LCD almost instantaneously. In addition to strings, numbers acquired as the blood pressure measurements also go through the LCD_Send_Data program to be displayed onto the LCD. It is important to note that each value returned to the main program as systolic, mean arterial, or diastolic pressure are converted to single digits first and then sent to the LCD. As such, when the sub-program receives a character, it sends it directly to the LCD module, which places it at the cursor's current position. The cursor is then advanced to the next position and prints the following character (*cf. Appendix B* 8.1 on page 51).

4.2.3 Main Program

The main program begins by initializing all input and output ports and the global variables used throughout the sub-programs. The only two pins that have been assigned as inputs are the AN0 and AN1 pins for the analog to digital converter and all other ports B, C and D have been assigned as outputs. Once the blood pressure device has been turned on, the LCD displays "START INFLATING". This will continue to show until the patient inflates the cuff to over 5mmHg, at which time the LCD will display "KEEP INFLATING". This allows the program to verify that the user understands the instructions and is ready to take the blood pressure measurement. From this point, the analog to digital converter continuously samples the raw signal from the AN0 pin while instructing the user to "KEEP INFLATING" on the first line and "P in mmHg = " on the second line. Since the ADC is continuously sampling the cuff pressure signal, the value from the ADC is converted into voltage and then to pressure, being constantly updated and displayed on the screen. However, once the pressure within the cuff reaches 170 mmHg, the LCD instructs the user to "STOP INFLATING" and calls the function Measure_Systolic. Once the Measure_Sytolic function has returned a value to the main program, the value of the systolic pressure will be displayed on the LCD. The Measure_MAP is then called and the value returned is also displayed on the LCD by overwriting the previous systolic pressure displayed. Finally, both the systolic and MAP pressures returned serve as inputs when the main program calls the Calculate_Diastolic function, at which time the program ends. (*cf. Appendix B* 8.1 on page 48).



Main Program
Initialization
Clear I/O Ports
Initialization
of LCD
Display
“START INFLATING”
Is pressure
>= 5mmHg?
N
Y
Display
“KEEP INFLATING”
Display
“P in mmHg =”
Sample cuff pressure
signal (AN0)
Display current
pressure in mmHg
Convert value to
pressure in mmHg
Is pressure
<= 170mmHg?
Y
N

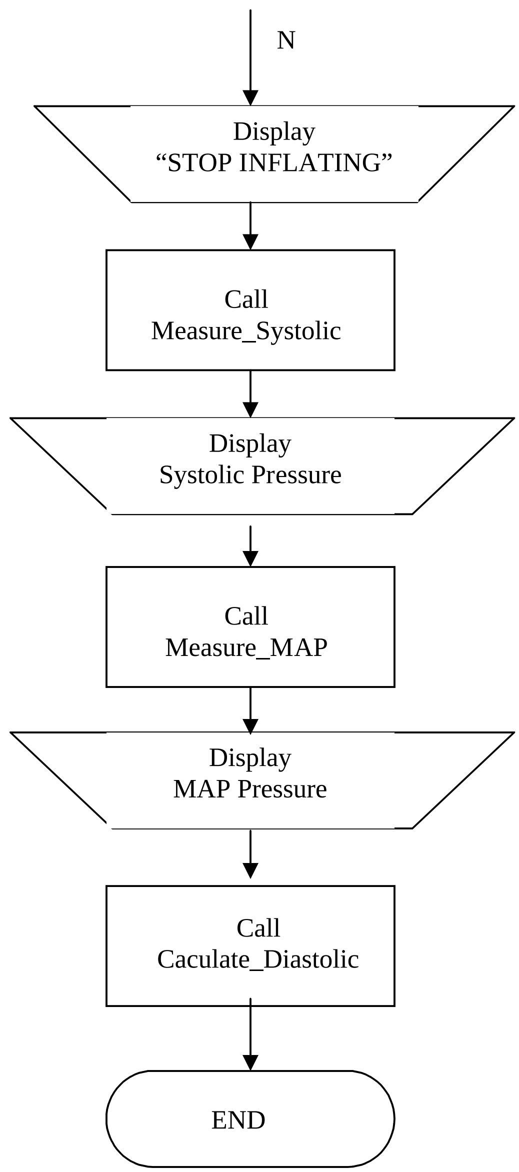
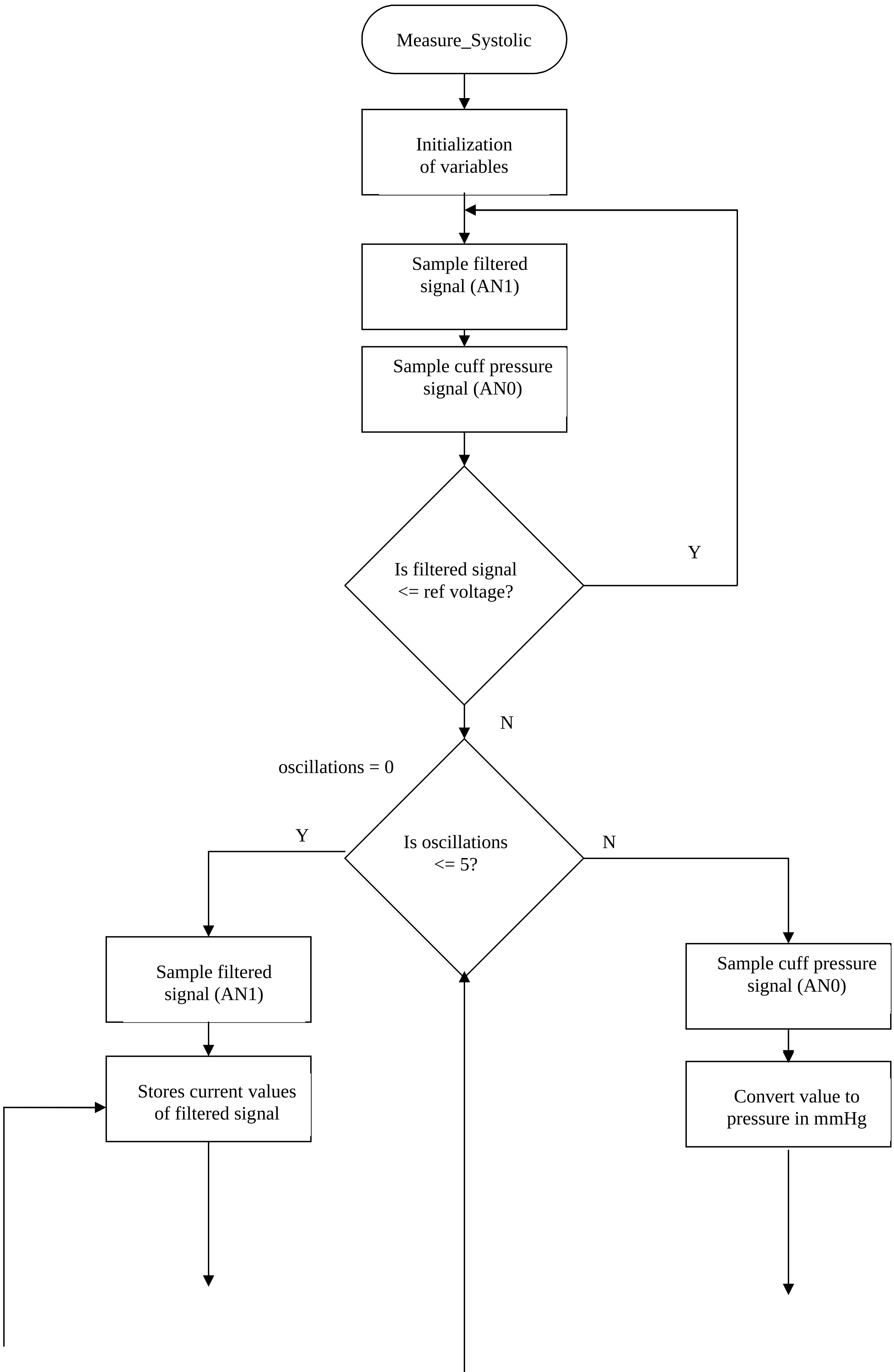


Figure 4.2.3.1: Flowchart of Main Program

4.2.4 Systolic Pressure Measurement

The algorithm for Measure_Systolic was very difficult to compute and involved a series of trials and errors before obtaining an accurate value for the systolic pressure. As seen in Figure 4.1.2.3 (*cf.* figure 4.1.2.3 on page 21) it is evident that the initial peak of the filtered signal does not correspond to the systolic pressure. Therefore, it was important to acknowledge that oscillations occur within the cuff even before the blood has broken through the opening of the artery. On account of the pressure sensor being so sensitive, it detects the oscillations that have occurred as a result of the blood trying to force its way through the blocked artery. Furthermore, the filtered signal has many oscillations that cannot be considered as peaks and thus must be ignored when computing the algorithm. Therefore, to account for these setbacks, the algorithm must specify a certain number of peaks before it can be considered the peak corresponding to the systolic pressure. However, the counting of the peaks can only occur if they are above a specified reference voltage. This eliminates the possibility of inadvertently mistaking noise for peaks. Although the initial algorithm consisted of pre-defining a reference voltage that once reached, was considered the systolic pressure. It was found that this method would not work because the reference voltage would be user dependant. If the patient naturally produced larger oscillations, then the reference voltage would be reached much sooner and thus would correspond to a significantly higher systolic pressure. As such, a lower reference voltage was defined to simply ignore the noise of the signal and five oscillations were counted above the predefined voltage. Also, to ensure that only peaks were being detected was problematic as the processing of the signal is occurring in real-time. An oscillation is only considered a peak if the current value along the signal is less than the previous sample by approximately 10mmHg and the previous sample is greater than its previous sample by 10mmHg. If both conditions are true, then it is counted as a peak and the detection repeats until five peaks are identified. Once out of the loop, the sample from pin AN0 corresponding to that last counted peak is converted from binary to voltage and voltage to pressure in mmHg through several function calls (*cf. Appendix B* 8.1 on page 52). This is the final systolic measurement that is returned to the main function so that it can be displayed to the user via its communication interface.


Measure_Systolic
Initialization of variables
Sample filtered signal (AN1)
Sample cuff pressure signal (AN0)
Is filtered signal <= ref voltage?
Y
N
oscillations = 0
Is oscillations <= 5?
Y
N
Sample filtered signal (AN1)
Stores current values of filtered signal
Sample cuff pressure signal (AN0)
Convert value to pressure in mmHg

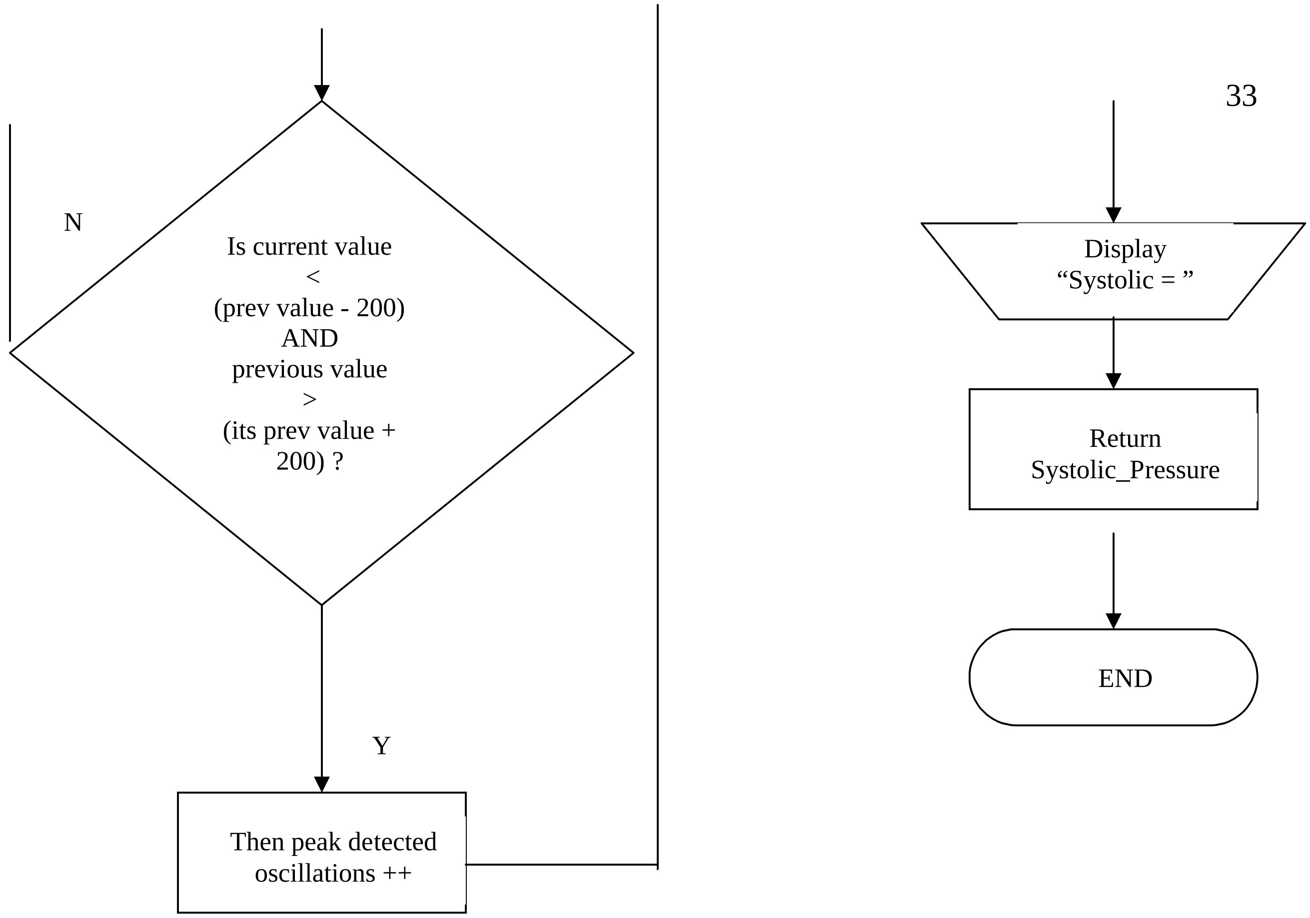
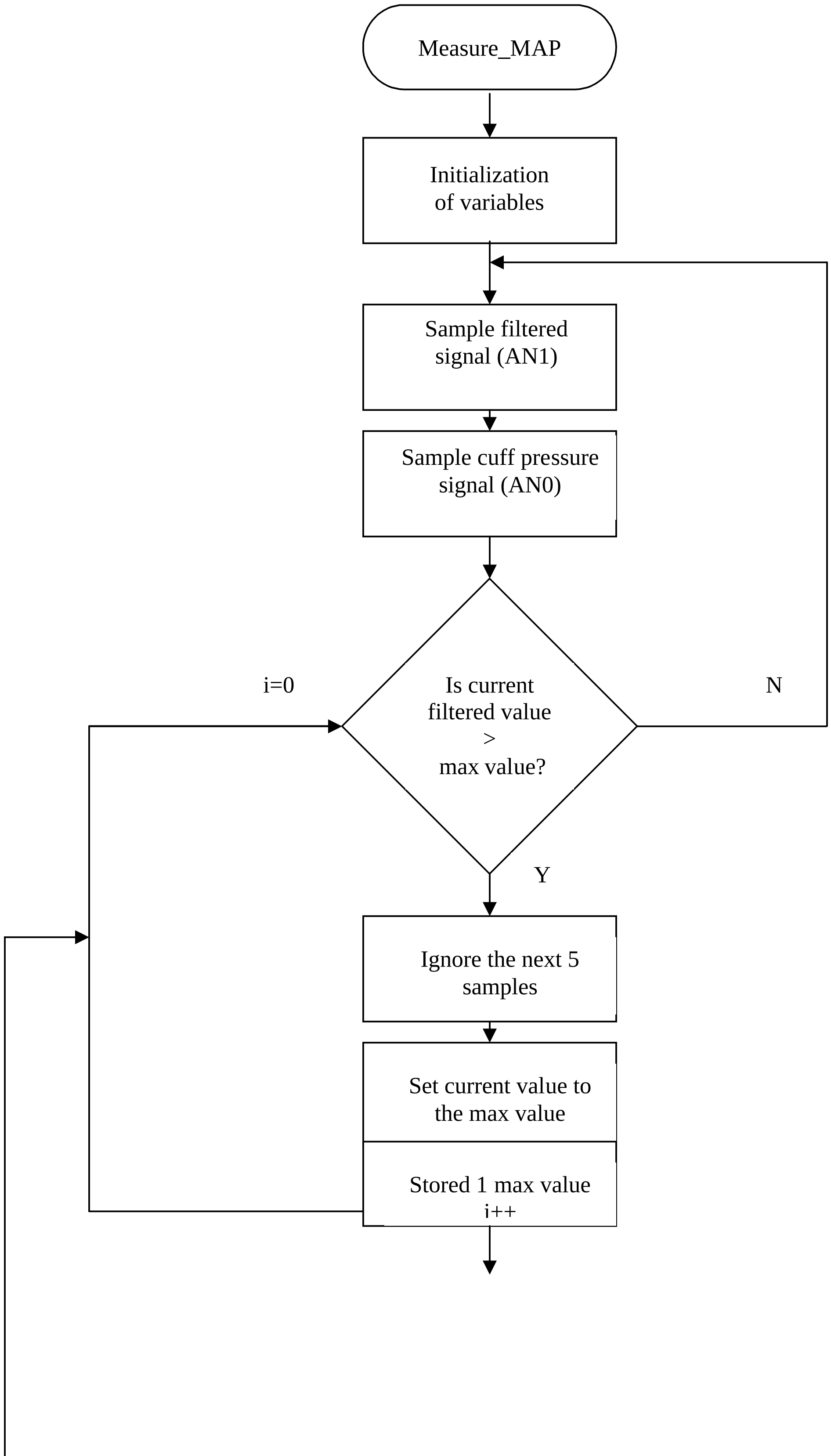
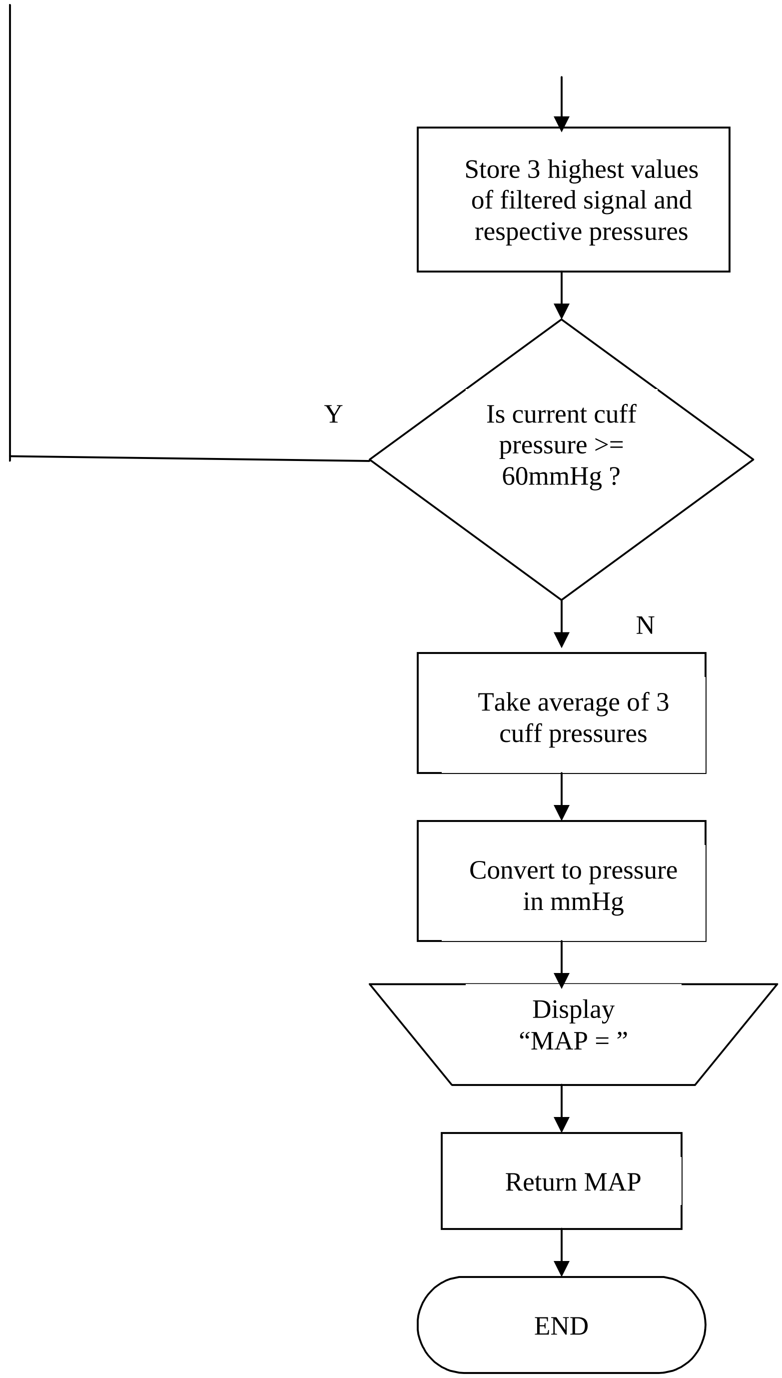


Figure 4.2.4.1: Flowchart of Systolic Pressure Measurement

4.2.5 Mean Arterial Pressure Measurement

Once the systolic pressure has been determined, the program calls Measure_MAP to determine the mean arterial pressure which is sensibly lower than the systolic pressure. Theoretically, the mean arterial pressure is the highest peak of the filtered signal located at the centre of the array of peaks. A maximum sorting algorithm was initially implemented whereby the max value of the samples was constantly stored or over written by the new highest value. Practically however, the location of the highest peak may not necessarily be at the centre as seen in figure 7.3.2 in *Appendix A* (*cf.* 7.3.2 on page 46), making the algorithm for detecting the mean arterial pressure more complex. To account for this, an insertion sort algorithm was implemented so that the pressure corresponding to the three highest samples are stored and averaged. However, due to the high sampling rate of the microcontroller, the three highest samples may be from the same main peak. To avoid this setback, five samples are ignored once the highest peak is stored into the array. This algorithm accounts for the possibility of the highest peak deviating from the centre of the sequence of peaks and thus provides the patient with an accurate mean arterial pressure. This algorithm differs from that of the systolic measurement because both the raw and filtered signals are continuously stored as pairs. This allows for the computation of the average raw pressure from the corresponding sorting of the peaks (*cf.* *Appendix B* 8.1 on page 52).

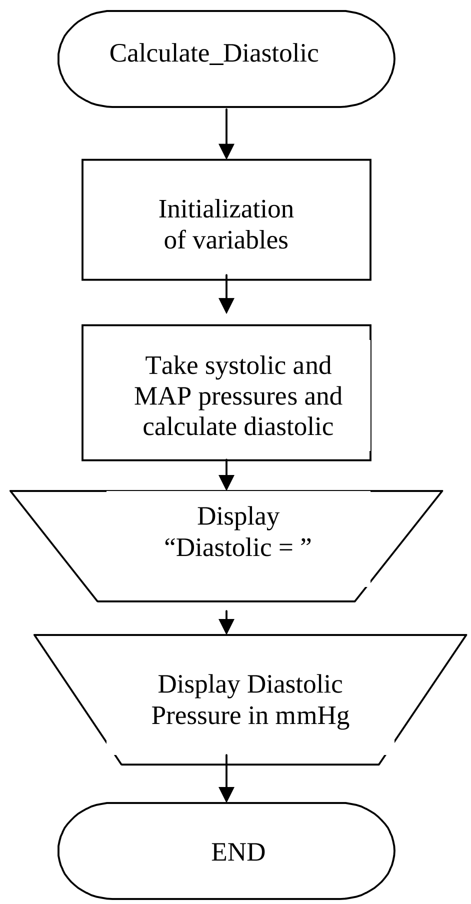




4.2.5.1 Flowchart of Mean Arterial Pressure Measurement

4.2.6 Diastolic Pressure Calculation

Once this measurement is complete, the function Calculate_Diastolic is called to calculate the diastolic pressure. The diastolic pressure is calculated using the aforementioned relationship relating the systolic and mean arterial pressure outlined in the *Appendix A* (*cf. Appendix A* 7.2.5 on page 45). Though this method for determining the diastolic pressure is wholly dependant on the values obtained for the other measurements and as a result, may lead to low values if systolic is too high or conversely, unrealistically low values if the mean is too high, completing it with a different algorithm would not be possible. Again, like with the systolic pressure algorithm, a predefined reference voltage could have also been chosen to determine the diastolic pressure. If the filtered peaks no longer surpassed the reference voltage, then the corresponding pressure would be the final measurement. However, this again introduces the dilemma of the reference voltage being user dependant. Therefore, if the patient had naturally large oscillations, then the resulting pressure would be very low, if at all detected. Similarly, if the patient has naturally small oscillations, then the resulting pressure would be unrealistically high. As such, the algorithm set forth in this design is the most accurate technique to compute the diastolic pressure. Again, once the diastolic pressure is calculated and converted to its appropriate units, it is displayed to the user. This terminates the sequence of measurements required to determine and display the systolic, mean arterial and diastolic pressures of the patient.



4.2.6.1 Flowchart of Diastolic Pressure Calculation

4.2.7 Voltage and Pressure Conversions

When the analog signals enter into the analog to digital converter, they are converted into binary values ranging from 0 – 1023 in binary and 0 -5V respectively. The binary values range from 0 – 1023 because the analog to digital converter is 10-bits. However, the relationship that exists due to the pressure sensor, converts voltage into pressure. Therefore, the binary values obtained from the A/D converter are converted back to voltage using the linear relationship outlined in *Appendix A* (*cf.* 7.2.3 on page 45).

In addition, the voltage is also converted into pressure in units of kPa and then into mmHg using the relationship in *Appendix A* (*cf.* 7.2.4 on page 45).

5 Results and Discussion

The results of the blood pressure device are comparable with that of a commercial blood pressure monitor. However, as with commercial products, it is necessary to that user remain stationary to obtain the most accurate results. Movement should be ceased throughout the entire duration of the measurement, specifically during the acquisition of the systolic and mean arterial pressures. As aforementioned, the pressure sensor is very sensitive to movement and will often display higher values for systolic pressure if motion occurs at that onset of the deflation of the cuff. Also, during the acquisition of the mean arterial pressure, if there is movement or touching of the blood pressure cuff at any specific instance, it will measure the oscillations caused by the interference at that moment as the highest peak and result in an incorrect reading.

The accuracy of the Quick Doc blood pressure device was measured as a percentage error using the blood pressures measured with the OMRON HEM- 711DLX as the theoretical values. Table 5.1 outlines a comparison of the blood pressures acquired with both systems (*cf.* table 5.1 on page 38) with the respective data plotted in Figure 5.1 (*cf.* figure 5.1 on page 38).

Table 5.1: Comparison between Quick Doc and OMRON commercial product

Subject	QUICK DOC Blood Pressure Device			OMRON HEM -711DLX		% Error	
	Systolic	MAP	Diastolic	Systolic	Diastolic	Sys	Dias
1	101	90	85	94	77	6.9	10.4
2	101	91	87	98	82	3.0	6.1
3	101	85	77	93	73	8.6	5.5
4	127	100	87	122	83	4.1	4.8
5	137	110	97	129	88	6.2	10.2
6	130	89	69	124	71	4.8	2.9
7	115	96	87	120	83	4.2	4.8
8	137	116	106	137	95	0	11.6
9	110	____	97	108	80	1.8	21.1

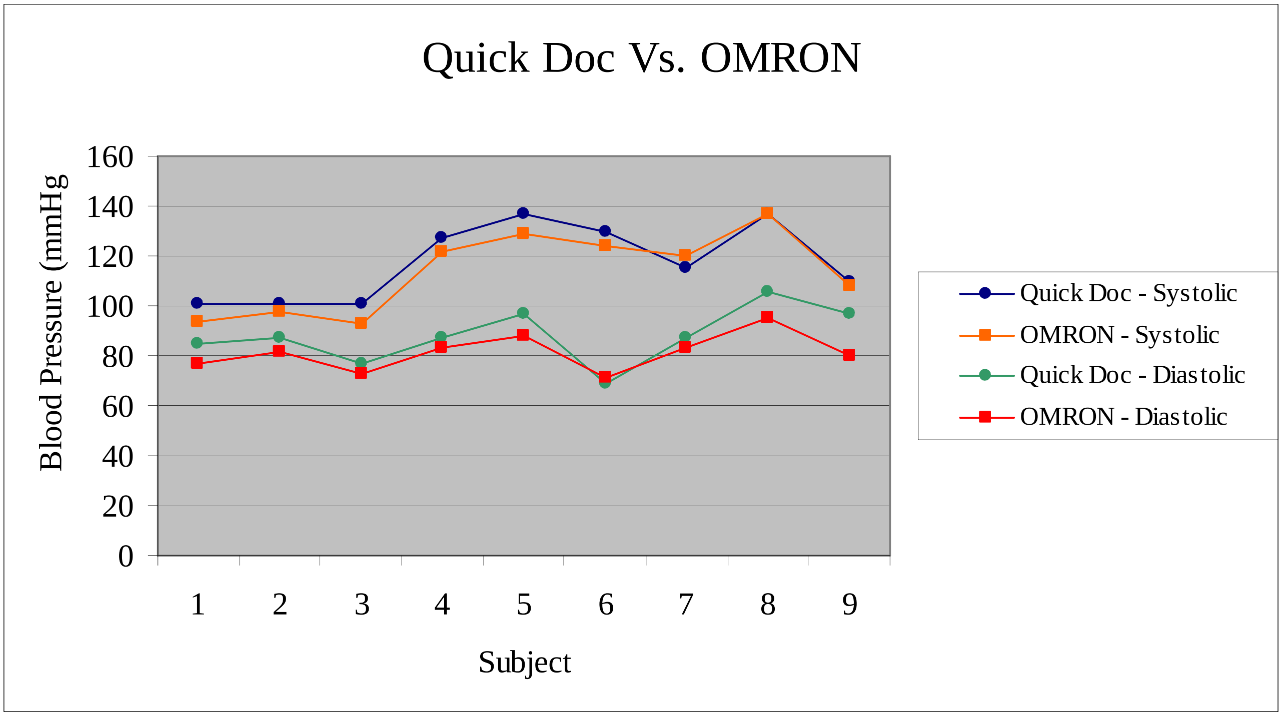


Figure 5.1 - Quick Doc vs. Omron. Graphical Representation of Table 5.1 showing correlation of systolic and diastolic pressures taken by Quick Doc and OMRON blood pressure devices.

As displayed in the results in the table and figure alike, the Quick Doc blood pressure device produces similar measurements to that of the OMRON HEM-711DLX. When tested in the lab, the majority of the subject’s blood pressures were taken a few minutes apart. However, subject’s 8 and 9 took the blood pressure simultaneously with the Quick Doc and the OMRON devices on their left and right arms respectively.

There are a few important similarities and differences to note in the results table. Firstly, it is evident that the mean arterial pressure has not been compared with that of a commercial product, as only systolic and diastolic pressures are displayed on the OMRON HEM-711DLX monitor utilized. However, the MAP values obtained using the Quick Doc blood pressure device appear to be accurate as they range between 90mmHg and 110mmHg, which is considered the norm.

It is apparent that the systolic pressure has a lower percentage error than that of the diastolic pressure. The larger discrepancy for the diastolic pressure is a consequence of the algorithm used for calculating the diastolic pressure. As discussed in the software section, the diastolic pressure is merely a calculation using the relationship between the systolic and mean arterial pressure. Therefore, the value obtained for diastolic is directly dependant on that of the systolic and mean arterial alike. In the event that the systolic pressure or the mean arterial pressure ia higher or lower than what it should be, it will directly impact the calculation for diastolic, especially if both systolic and mean arterial are off. However, as noted in the table above, there is no consistent discrepancy with the diastolic pressures. Some measurements with the Quick Doc are higher than the OMRON, whereas in some instances it is lower. Perhaps, using the relationship between the three different pressures was not the most accurate means of determining the diastolic pressure. However, given the aforementioned problems of the amplitudes of the peaks produced by the oscillations varying from individual to individual, setting a pre-defined reference voltage did not seem plausible.

In addition, the systolic pressures from the Quick Doc device and OMRON system are accurate to within 0 - 9%. This is a very low percentage error, especially since the exact value of systolic is not of crucial importance. Rather, the range of which the systolic pressure is in is of greater significance. As seen in the table, it is evident that all subjects would fall within the same blood pressure range of 100 -120, 120-140, etc. Therefore, a subject with a systolic pressure signifying hypotension with the Quick Doc would receive the same diagnosis with the OMRON.

Also, when working with the commercialized product, it is advised that a minimum of three blood pressure readings are acquired sequentially, so that the average of the pressures is the true representation of the actual blood pressure of the patient. When comparing the sequential readings of the Quick Doc and the OMRON, as seen with the same subject in samples 1-3 in the comparison table, the Quick Doc gives a consistent reading of 101mmHg whereas the OMRON device gives a systolic pressure ranging from 73 to 82mmHg. Therefore, when acquiring blood pressures from the same patient at approximately 1 minute intervals with the Quick Doc blood pressure device, it is more consistent, and thus believed to be more accurate than that of the measurements of the OMRON commercial product.

As aforementioned, subjects 8 and 9 took their blood pressure with the Quick Doc device on their right arm and the OMRON device on their left arm simultaneously, to avoid any discrepancy of their blood pressure over time. This really proved to be effective in acquiring similar results, particularly for the systolic pressure. A snapshot of the systolic pressure of subject 9 can be found in Figure 5.2 (*cf.* figure 5.2 on page 41).

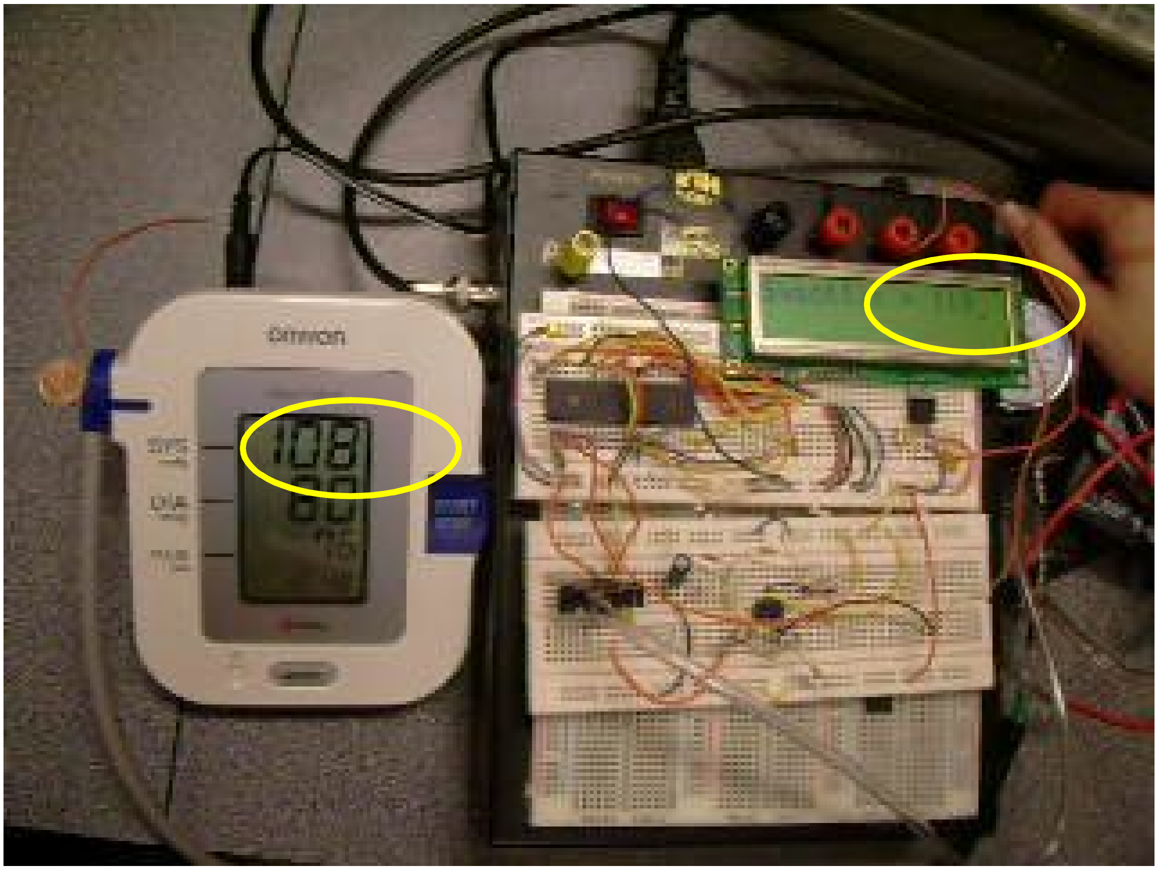


Figure 5.2: Subject 9 acquiring blood pressure with Quick Doc device and OMRON system simultaneously. Result: OMRON 108mmHg and Quick Doc 110mmHg

Overall, I am pleased with the results of the Quick Doc blood pressure device as they are a good representation of the pressure exerted on the arterial walls of the patient. Figure 5.1 (*cf.* figure 5.1 on page 38) clearly depicts that the systolic and diastolic blood pressures taken with both devices follows the same trend, within a moderate error.

In taking the blood pressure measurements, it was realized that if used correctly it takes approximately 30 – 45 seconds to perform the blood pressure acquisition, from start to finish. This time does not include the time taken to wrap the cuff around the upper arm.

Since this is a medical instrumentation device, the safety of the user is of paramount importance. As such, the microcontroller is programmed to instruct the user via the LCD to stop inflating once the pressure reaches 170mmHg. This prevents over-inflation of the cuff causing great discomfort of the arm. For the majority of people, 170mmHg is above their typical systolic pressure and thus is a safe pressure to have as the maximum limit.

Prior to experimentation in the lab, precautions were taken to ensure that the device was working properly. As aforementioned the device requires low voltages to operate and therefore does not put any patient at harm. Also, in doing this investigation on various subjects, it is evident that this device should be appropriate for use by most adults in terms of its safety. However, this device has only been tested on healthy undergraduate students, which my not fit the general population. As such, is not recommended that those with known blood pressure or heart problems utilize this device until further testing is conducted to ensure safety to these individuals. Also, the pressure cuff implemented in this design is appropriate for use by the average adult with an arm size ranging from 9-13 inches in circumference. Individuals whose arm circumferences fall outside of this range may not be given accurate measurements.

6 Conclusions and Recommendations

The design of a blood pressure device employing the MPX5050GP pressure sensor and the PIC18F452 microcontroller successfully acquires the necessary signals and provides an interpretable result. This result is the systolic, mean arterial and diastolic blood pressures of a patient. The device has proven to be effective in providing the measurements in real-time to the user via an LCD display. As such, the data is presented to the user as it is being processed by the microcontroller. Through the experimentation and analysis of the Quick Doc blood pressure system, it is evident that it is extremely user-friendly and easily operated by one individual. The single-handed capabilities of Quick Doc ultimately satisfy the objective set prior to the implementation of the design. The objective of maximizing the patient/doctor interaction by eliminating conventional procedures, such that of blood pressure, being performed by the practitioner has been accomplished. The practitioner can now have access to the patient’s current blood pressure, blood oxygen content and electrocardiogram signal prior to attending to the patient. The wireless capabilities of Quick Doc allow for these predefined physiological signals to be transmitted from the waiting room of a clinic or hospital to the base station of the practitioner. As such, the applications of Quick Doc are endless, provided that minor alterations to the design are made.

The aforementioned limitations of Quick Doc and of the blood pressure component specifically should serve as a foundation to the future improvements of the device. Also, other physiological signals or conventional measurements such as temperature can easily be implemented as an extension to the current system. Furthermore, individual components of Quick Doc, such as blood pressure measurement, could also be used separately in a home environment where data can be stored on a continuous basis. This allows for constant monitoring of blood pressure so that the user is able to adjust their habits and lifestyle accordingly.

Ultimately, Quick Doc is the future of medical institutions where time and efficiency are of utmost importance.

7 Appendix A

7.1 Formulas

7.1.1 Blood Pressure

$$BP = CO \times SVR$$

7.1.2 Mean Arterial Pressure

$$MAP = (CO \cdot SVR) + CVP.$$

$$MAP \approx P_{dias} + \frac{1}{3}(P_{sys} - P_{dias}).$$

7.2 Calculations

7.2.1 – Poles of Two –Pole High Pass Filter

$$f_1 = \frac{1}{2\pi R1C1} \qquad f_2 = \frac{1}{2\pi R3C2}$$

$$f_1 = \frac{1}{2\pi(1k)(47u)} \qquad f_2 = \frac{1}{2\pi(100k)(4.7u)}$$

$$f_1 = \frac{1}{2\pi R1C1} \qquad f_2 = \frac{1}{2\pi R3C2}$$

$$\boldsymbol{f_1 = 0.338\ Hz} \qquad \boldsymbol{f_2 = 0.386\ Hz}$$

7.2.2 – Sampling Rate & Delay

Cycles = (TimeDelay * Fosc) / 4

Cycles = (15ms * 16MHz) / 4

Cycles = 60,000

7.2.3 – Binary to Voltage Conversion

Voltage = 5* (Binary/1023)

7.2.4 – Voltage to Pressure Conversion

1mmHg = 0.133kPa

P = mV + b , where m = slope and b = y offset/intercept

m = (2 – 1.5)/(20 – 15) = 10

b = - 0.5

P = 10V – 0.5

7.2.5 – Diastolic Pressure Calculation

$$MAP \simeq DP + \frac{1}{3}(SP - DP)$$

MAP = DP + 1/3(SP-DP)

MAP = 2/3DP + 1/3SP

2/3DP = MAP - 1/3SP

DP = 3/2(MAP - 1/3SP)

7.3 Hardware

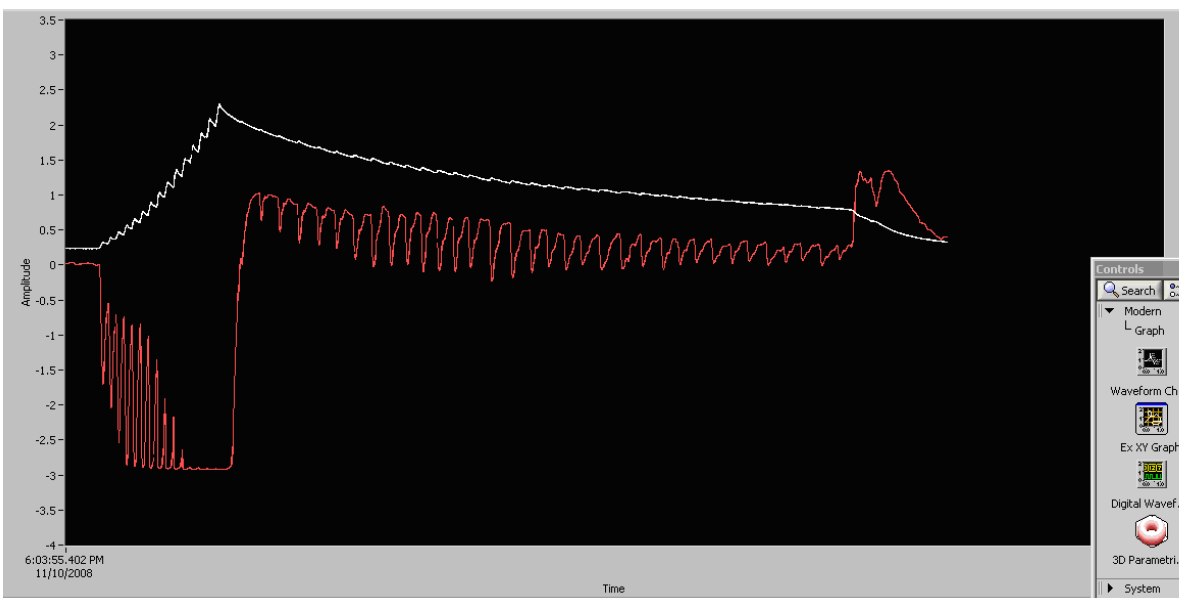


Figure 7.3.1: Cuff Pressure Signal and Extracted Oscillation Signal using Bandpass Filter

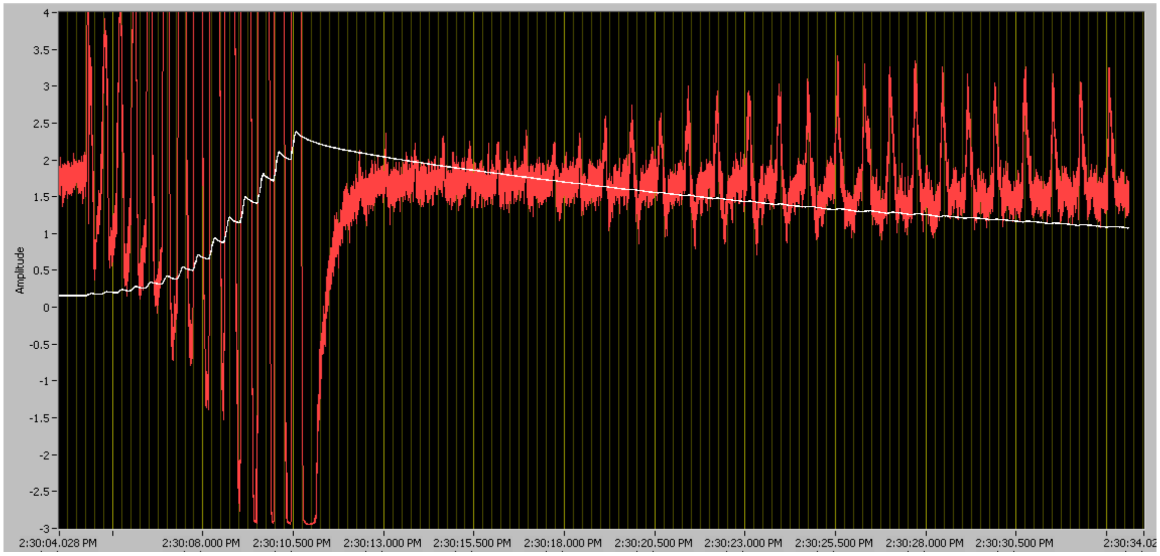


Figure 7.3.2: Filtered Signal where the highest peak is not the midpoint of Systolic and Diastolic Pressure

7.4 Parts List

Table 7.4.1: Components and the Prices

COMPONENTS	**PRICE**
Blood pressure cuff w/ gauge	$25.00
Pressure Sensor MPX5050GP	$25.00 x 2 = $50.00
Microchip PIC18F452	$5.00
10-bit ADC (built into PIC18F452)	________
HA 17741 op-amp	Sample
PC 1602-L 16x2 Character Display	$10.00
Hitachi HD44780 LCD controller/driver (built into PC 1602-L)	________
L7805CV Voltage Regulator	Free
Resistors and Capacitors	Free
Breadboard	$10.00
Microchip PICSTART Plus Programmer	Borrowed

7.5 List of Computer Programs

A list of the computer programs throughout the design and implementation of the blood pressure device are listed below:

- Microchip's MPLAB Integrated Development Environment
- MPLAB C18 C Compiler
- Proteus VSM simulate PIC microcontroller
- LabVIEW
- Microsoft Office 2003 (Microsoft Word and Excel)

8 Appendix B

8.1 Source Code

The source code corresponding to the algorithms outlined in the flowcharts throughout the report can be found below.

```
/****************QUICK DOC: BLOOD PRESSURE CODE**********************

Doralice Ferreira
Electrical and Biomedical Engineering IV
Elec Eng 6BI6

*********************************************************************/

#include <stdio.h>
#include <p18f452.h>
#include <delays.h>
#include <string.h>
#include <adc.h>
#pragma config WDT= OFF

/*********************** PIN CONNECTIONS ***************/
// PORTD = D0 - D7 on LCD
// PORTB: B0 = CCT ON, B1 = E, B2 = RW, B3 = RS

int ADC_conv(int graph_number);
void lcd_init();
int check_busy();
void delay_15ms();
void LCD_Send_Data(unsigned char var);
void LCD_Send_String(char *var);
double Convert_ADCtoVoltage (double Current_ADC);
double Convert_VoltagetoPressure (double Current_Voltage);
int Measure_Systolic();
int Measure_MAP();
void Calculate_Diastolic(int Systolic_Pressure, int MAP);


/**************** GLOBAL VARIABLES ****************/
int Systolic_Pressure;               //Final Systolic Pressure(mmHg)
int Diastolic_Pressure;                      //Final Diastolic Pressure(mmHg)
int MAP;                                             //Final Mean Arterial Pressure (mmHg)
double Current_Graph1_Voltage;               //Graph1 Voltage
double Current_Graph2_Voltage;               //Graph2 Voltage
int GRAPH1 = 1;                                      //Define numerical value for graph 1
for input selection
int GRAPH2 = 2;                                      //Define numerical value for graph 2
for input selection
int Xp,X, X1, X2, X3, Xd, Xs, Xm;
double Xv;

/************************ MAIN PROGRAM ****************************/
void main ()
{

// FOR DISPLAYING NUMBERS
       int Xp,X, X1, X2, X3, Xd, Xs, Xm;
       double Xv;
       char i, j, c;
       char run = 1;
       int GRAPH1 = 1;                                       //Define numerical value for
graph 1 for input selection
```

```
        int GRAPH2 = 2;                                     //Define numerical value for
graph 2 for input selection
        double Systolic_Pressure;                   //Final Systolic Pressure(mmHg)
        double Diastolic_Pressure;                  //Final Diastolic Pressure(mmHg)
        double MAP;                                         //Final Mean Arterial
Pressure (mmHg)

        char mybuff4[20] = "START INFLATING";
        char mybuff [20] = "KEEP INFLATING";
        char mybuff2 [20] = "P in mmHg= ";
        char mybuff3 [20] = "STOP INFLATING";

        TRISB = 0;
        PORTB = 0b10000000;    // Circuit ON
        TRISC = 0;
        PORTC = 0;
        TRISD = 0;
        PORTD = 0;
        lcd_init();
        delay_15ms();

        LCD_Send_String( mybuff4 );

//      Delay1KTCYx(2000);                      //delay 500ms
//      Delay1KTCYx(2000);                      //delay 500ms
//      Delay1KTCYx(2000);                      //delay 500ms
//      Delay1KTCYx(2000);                      //delay 500ms

        X = ADC_conv(GRAPH1);
        Xv = Convert_ADCtoVoltage (X);
        Xp = Convert_VoltagetoPressure (Xv);

        if (Xp >= 5){

                delay_15ms();                   //GO HOME
                PORTBbits.RB3 = 0;              //Set RS
                PORTD = 0b00000001;
                PORTBbits.RB1 = 1;              //Enable H->L
                PORTBbits.RB1 = 0;

                LCD_Send_String( mybuff );

                delay_15ms();                   //GOTO 2ND LINE
                PORTBbits.RB3 = 0;              //Set RS
                PORTD = 0b10101000;
                PORTBbits.RB1 = 1;              //Enable H->L
                PORTBbits.RB1 = 0;

                LCD_Send_String( mybuff2 );
        }

        while(run == 1){

                X = ADC_conv(GRAPH1);
                Xv = Convert_ADCtoVoltage (X);
                Xp = Convert_VoltagetoPressure (Xv);
                X1 = Xp/100;
                X2 =(Xp-(Xp-Xp%100))/10;
                X3 = Xp%10;

                if (Xp <= 170 ) {
                        // FOR DISPLAYING NUMBERS
                        LCD_Send_Data(X1+48);
                        LCD_Send_Data(X2+48);
                        LCD_Send_Data(X3+48);

                        for (c=0;c<3;c++){
                                delay_15ms();
                                PORTBbits.RB3 = 0;
                                PORTD = 0b00010000;             //Shift (cursor, left)
                                PORTBbits.RB1 = 1;              //Enable H->L
```

```
                                delay_15ms();
                                PORTBbits.RB1 = 0;
                        }
                }
                else {
                        delay_15ms();                   //GO HOME
                        PORTBbits.RB3 = 0;              //Set RS
                        PORTD = 0b00000001;
                        PORTBbits.RB1 = 1;              //Enable H->L
                        PORTBbits.RB1 = 0;

                        LCD_Send_String( mybuff3 );

                        Delay1KTCYx(2000);                      //delay 500ms
                        Delay1KTCYx(2000);                      //delay 500ms

                        delay_15ms();                   //GO HOME
                        PORTBbits.RB3 = 0;              //Set RS
                        PORTD = 0b00000001;
                        PORTBbits.RB1 = 1;              //Enable H->L
                        PORTBbits.RB1 = 0;

                        Delay1KTCYx(2000);                      //delay 500ms
                        Delay1KTCYx(2000);                      //delay 500ms
                        Delay1KTCYx(2000);                      //delay 500ms
                        Delay1KTCYx(2000);                      //delay 500ms
                        Delay1KTCYx(2000);                      //delay 500ms
                        Delay1KTCYx(2000);                      //delay 500ms

                        Systolic_Pressure = Measure_Systolic();
                        Xs = Systolic_Pressure;
                        X1 = Xs/100;
                        X2 = (Xs-(Xs-Xs%100))/10;
                        X3 = Xs%10;

                        LCD_Send_Data(X1+48);
                        LCD_Send_Data(X2+48);
                        LCD_Send_Data(X3+48);

                        MAP = Measure_MAP();
                        Xm = MAP;
                        X1 = Xm/100;
                        X2 = (Xm-(Xm-Xm%100))/10;
                        X3 = Xm%10;
                        LCD_Send_Data(X1+48);
                        LCD_Send_Data(X2+48);
                        LCD_Send_Data(X3+48);

                        Calculate_Diastolic(Systolic_Pressure, MAP);

                        run = 0;                                        //Measurement
complete, end program
                }
        }
        //      sprintf(tempString, "%d", X);
        //      for (i=512; i!=0; i >>= 1) {
        //              if (X&i) LCD_Send_Data('1');
        //              else LCD_Send_Data('0');
        //      }

        while(1);
}

/*********************** ADC **************************/
//// graph_number variable decides which Analog input to convert
//// GRAPH1 = AN0 = ADC_CH0 = Raw Signal
//// GRAPH2 = AN1 = ADC_CH1 = Filtered Signal

int ADC_conv(int graph_number){

        int result;
```

```
	int result1;
	int result2;

	if (graph_number == 1){

		OpenADC( ADC_FOSC_2 & ADC_RIGHT_JUST & ADC_8ANA_0REF, ADC_CH0 &
ADC_INT_OFF );

		Delay10TCYx( 5 );			// Delay for 50TCY

		ConvertADC();				// Start conversion
		while( BusyADC() );		// Wait for completion

		result = ReadADC();		// Read result

		result1 = result & 0x00ff;
		result2 = (result & 0xff00)>>8;
	}
	else if (graph_number == 2) {

		OpenADC( ADC_FOSC_2 & ADC_RIGHT_JUST & ADC_8ANA_0REF, ADC_CH1 &
ADC_INT_OFF );

		Delay10TCYx( 5 );			// Delay for 50TCY

		ConvertADC();				// Start conversion
		while( BusyADC() );		// Wait for completion

		result = ReadADC();		// Read result

		result1 = result & 0x00ff;
		result2 = (result & 0xff00)>>8;
	}
	CloseADC();				// Disable A/D converter

	return result;
}

/************************ SEND CHARACTER **************************/
void LCD_Send_Data(unsigned char var){

	delay_15ms();
	PORTBbits.RB3 = 1;		//Set RS
	PORTD = var;			//Write char
	PORTBbits.RB1 = 1;		//Enable H->L
	delay_15ms();
	PORTBbits.RB1 = 0;
	return;
}

/************************ SEND STRING ****************************/
void LCD_Send_String(char *var){

	while(*var){
		LCD_Send_Data(*var);
		var++;
		delay_15ms();
	}
	return;
}

/************************ INITIALIZE LCD *****************************/
void lcd_init(){

	delay_15ms();					//Initialize LCD
	PORTD = 0b00110000;
	PORTBbits.RB1 = 1;		//Enable
	delay_15ms();
	PORTBbits.RB1 = 0;

	delay_15ms();
```

```
        PORTD = 0b00110000;
        PORTBbits.RB1 = 1;            //Enable
        delay_15ms();
        PORTBbits.RB1 = 0;

        delay_15ms();
        PORTD = 0b00110000;
        PORTBbits.RB1 = 1;            //Enable
        delay_15ms();
        PORTBbits.RB1 = 0;

        delay_15ms();
        PORTD = 0b00111000;                           //Initial Function Set: 8 bits, 2
lines,
        PORTBbits.RB1 = 1;            //Enable
        delay_15ms();
        PORTBbits.RB1 = 0;

        delay_15ms();                                 //Display ON, Curson ON, Blink ON
        PORTD = 0b00001111;
        PORTBbits.RB1 = 1;            //Enable
        delay_15ms();
        PORTBbits.RB1 = 0;

return;
}
/************************ DELAY 15MS ****************************/
void delay_15ms(){

        Delay1KTCYx(100);                             // Delay of 15ms
                                                              // Cycles = (TimeDelay
* Fosc) / 4
                                                              // Cycles = (15ms *
16MHz) / 4
                                                              // Cycles = 60,000
                                                              // Refer to delay
header file
return;
}

/******************** CONVERT ADC TO VOLTAGE ****************/
double Convert_ADCtoVoltage (double Current_ADC) {
// Linear Relationship
// V = 5*Current_ADC/255

        double V = 0b0000000101* (Current_ADC / 0b1111111111);

return V;
}

/******************** CONVERT VOLTAGE TO PRESSURE ****************/
double Convert_VoltagetoPressure (double Current_Voltage) {
// 1mmHg = 0.133kPa
// y = mV + b         m = (2-1.55)/(20-15) = 10    , b = -0.5
// y = 10V  - 0.5
        double mmHg;
        double kpa;

        if (Current_Voltage <= 0.3){
                mmHg = 0;
        }
        else{
                kpa = 0b0000001010 *Current_Voltage - 0.5;
                mmHg = kpa *7.50062;
        }
return mmHg;
}

/******************* MEASURE SYSTOLIC PRESSURE ****************/
int Measure_Systolic(){
// Using GRAPH2 input, after 3 oscillations, switch to GRAPH1 input and
```

```
//Convert voltage using ADC, then convert to systolic pressure(mmHg)

        int max_val = 0;
        int oscillations =0;
        double current_val = 0;
        int current_raw;
        int Systolic_Pressure;
        int N = 20;
        int old_samples[20];
        int c = 0;
        int peak;
        int i=1;
        double Current_Graph1_Voltage;
        char mybuff[20] = "Sysotlic = ";
        int GRAPH1= 1;
        int GRAPH2= 2;
        int run =1;
        double ref_voltage = 1.7;      //Reference voltage is user dependant

while (run==1){

        current_val = Convert_ADCtoVoltage(ADC_conv(GRAPH2));
        current_raw = ADC_conv(GRAPH1);

        if(current_val <= ref_voltage){
                current_val = Convert_ADCtoVoltage(ADC_conv(GRAPH2));
                current_raw = ADC_conv(GRAPH1);
        }
        else if(current_val> ref_voltage){

                if(oscillations <= 5){
                        current_val = ADC_conv(GRAPH2);

                        for(i=0; i<N-1; i++){
                                old_samples[i]= old_samples[i+1];
                        }
                        old_samples[N-1] = current_val;
                        peak = 0;
                        for (i=1; i<N-1; i++) {
                                if (current_val<(old_samples[i]-200) &&
old_samples[i]>(old_samples[i-1]+200)) {
                                        peak = 1;
                                        break;
                                }
                        }
                        if (peak==1){
                                oscillations++;
                        }
                }
                else {
                        Current_Graph1_Voltage = Convert_ADCtoVoltage (current_raw);

                        Systolic_Pressure =
Convert_VoltagetoPressure(Current_Graph1_Voltage);
                        LCD_Send_String( mybuff );
                        run =0;
                }
        }
}
return Systolic_Pressure;
}
///**************** MEASURE MEAN ARTERIAL PRESSURE **************/
int Measure_MAP(){

        char run=1, i, j;
        short current_val, current_raw;
        short max_val[3]={0,0,0}, max_raw[3]={0,0,0};
        double Current_Graph1_Voltage;
        int MAP;
        char mybuff[20] = "MAP = ";
        short ignore=0;
```

```
	while (run) {
		current_val = ADC_conv(GRAPH2);
		current_raw = ADC_conv(GRAPH1);
		if (ignore>0) ignore--;

		if (ignore==0) {
			for (i=0; i<3; i++) {
				if (current_val>max_val[i]) {
					ignore = 5;
					for (j=2; j>i; j--) {
						max_val[j] = max_val[j-1];
						max_raw[j] = max_raw[j-1];
					}
					max_val[i] = current_val;
					max_raw[i] = current_raw;
					break;
				}
			}
		}

		if (current_raw <= 143 && max_raw[2]!=0) {   //exit condition
			run = 0;
			current_raw = (max_raw[0]+max_raw[1]+max_raw[2]+1)/3;

			Current_Graph1_Voltage = Convert_ADCtoVoltage(current_raw);
			MAP = Convert_VoltagetoPressure(Current_Graph1_Voltage);

			delay_15ms();			//GO HOME
			PORTBbits.RB3 = 0;		//Set RS
			PORTD = 0b00000001;
			PORTBbits.RB1 = 1;		//Enable H->L
			PORTBbits.RB1 = 0;
			LCD_Send_String( mybuff );
		}
	}

	return MAP;
}

/*************** CALCULATE DIASTOLIC PRESSURE ************/
void Calculate_Diastolic(int Systolic_Pressure, int MAP){
// MAP = DP + 1/3(SP-DP)
// MAP = 2/3DP + 1/3SP
// 2/3DP = MAP - 1/3SP
// DP = 3/2(MAP - 1/3SP)

	int Diastolic_Pressure;
	char mybuff[20] = "Diastolic = ";

	Diastolic_Pressure = 1.5*(MAP - Systolic_Pressure/3);

	delay_15ms();			//GOTO 2ND LINE
	PORTBbits.RB3 = 0;		//Set RS
	PORTD = 0b10101000;
	PORTBbits.RB1 = 1;		//Enable H->L
	PORTBbits.RB1 = 0;

	LCD_Send_String( mybuff );
	Xd = Diastolic_Pressure;
	X1 = Xd/100;
	X2 = (Xd-(Xd-Xd%100))/10;
	X3 = Xd%10;
	LCD_Send_Data(X1+48);
	LCD_Send_Data(X2+48);
	LCD_Send_Data(X3+48);

	return;
}
```

8.2 Source Code (Trial and Error)

The source code that was implemented throughout the design, but not used into the final product can be found below.

```
/*************************************************************************/
/*               SYSTOLIC PRESSURE (trial & error)                       */
/*************************************************************************/

//      while (run ==1){
//              current_val = Convert_ADCtoVoltage(ADC_conv(GRAPH2));
//              current_raw = ADC_conv(GRAPH1);
//
//              if(current_val <= ref_voltage){
//                      current_val = Convert_ADCtoVoltage(ADC_conv(GRAPH2));
//                      current_raw = ADC_conv(GRAPH1);
//              }
//              else if(current_val> ref_voltage){
//              Current_Graph1_Voltage = Convert_ADCtoVoltage (current_raw);
//              Systolic_Pressure = Convert_VoltagetoPressure(Current_Graph1_Voltage);
//              run=0;
//              LCD_Send_String( mybuff );
//              }
//      }
//      return Systolic_Pressure;
//      }

/*********************************************************/
/////////////////////////////////////////////////////

//      int num_max=3;
//      int max[3]={0,0,0};
//      int max_ori[3]={0,0,0};
//      int temp;
//      int i = 0;
//      int j =0;
//      int tmp =0;
//      int current_ori=0;
//      double syst_v=0;
//      int syst_p=0;
//      int avg_syst=0;
//      int run1 = 1;
//      int peak =0;
//
//      while(run1 == 1){
//
//              LCD_Send_String( mybuff );
//              current_val = ADC_conv(GRAPH2);
//              max[0]=current_val;
//              current_val = ADC_conv(GRAPH2);
//              max[1]=current_val;
//              current_val = ADC_conv(GRAPH2);
//              max[2]=current_val;
//              LCD_Send_Data('a');
//
//      Xs =  max[2];
//      X1 = Xs/100;
//      X2 = (Xs-(Xs-Xs%100))/10;
//      X3 = Xs%10;
//      LCD_Send_Data(X1+48);
//      LCD_Send_Data(X2+48);
//      LCD_Send_Data(X3+48);
//
//              /////////bubble sort for max values
//              for (i=0; i<num_max-1; i++) {
```

```
//              for (j=0; j<num_max-1-i; j++)
//                if (max[j+1] < max[j]) {  /* compare the two neighbours */
//                       tmp = max[j];         /* swap max[j] and max[j+1]      */
//                       max[j] = max[j+1];
//                       max[j+1] = tmp;
//                   LCD_Send_Data('b');
//      Xs =  max[0];
//      X1 = Xs/100;
//      X2 = (Xs-(Xs-Xs%100))/10;
//      X3 = Xs%10;
//      LCD_Send_Data(X1+48);
//      LCD_Send_Data(X2+48);
//      LCD_Send_Data(X3+48);
//                   }
//
//           }
//
//           while(run==1)  {
//                   LCD_Send_Data('w');
//
////                 current_val = ADC_conv(GRAPH2);
////                 if(current_val>max[0]) {
////                         max[2]=max[1];
////                         max[1]=max[0];
////                         max[0]=current_val;
////    LCD_Send_Data('c');
//
//                   current_val = ADC_conv(GRAPH2);
//                   if(current_val>max[2]) {
//                           max[0]=max[1];
//                           max[1]=max[2];
//                           max[2]=current_val;
//      LCD_Send_Data('c');
//
//
//                           current_ori = ADC_conv(GRAPH1);
//                           max_ori[0]=max_ori[1];
//                           max_ori[1]=max_ori[2];
//                           max_ori[2]=current_ori;
//           LCD_Send_Data('d');
//                           syst_v = Convert_ADCtoVoltage(current_ori);
//                           syst_p = Convert_VoltagetoPressure(syst_v);
//      Xs =  syst_p;
//      X1 = Xs/100;
//      X2 = (Xs-(Xs-Xs%100))/10;
//      X3 = Xs%10;
//      LCD_Send_Data(X1+48);
//      LCD_Send_Data(X2+48);
//      LCD_Send_Data(X3+48);
//                   }
//
//                   if(syst_p <= 80)      {
//
//                           delay_15ms();                //GOTO 2ND LINE
//                           PORTBbits.RB3 = 0;           //Set RS
//                           PORTD = 0b10101000;
//                           PORTBbits.RB1 = 1;           //Enable H->L
//                           PORTBbits.RB1 = 0;
//
//                           LCD_Send_String( mybuff );
//                           LCD_Send_Data('e');
//                           run =0;
//                   }
//           }
//           avg_syst=(max_ori[0]+max_ori[1]+max_ori[2])/3;
//           LCD_Send_Data('f');
//           run1 =0;
//
//           return avg_syst;
//           }
//      }
```

```
/////////////////////////////////////////////////////////////////////
//
//while (run==1){
//
//	current_val = Convert_ADCtoVoltage(ADC_conv(GRAPH2));
//	current_raw = ADC_conv(GRAPH1);
//
//	if(current_val <= ref_voltage){
//		current_val = Convert_ADCtoVoltage(ADC_conv(GRAPH2));
//		current_raw = ADC_conv(GRAPH1);
//	}
//	else if(current_val> ref_voltage){
//
//		if(oscillations <= 3){
//			current_val = ADC_conv(GRAPH2);
//
//			for(i=0; i<N-1; i++){
//				old_samples[i]= old_samples[i+1];
//			}
//			old_samples[N-1] = current_val;
//			peak = 0;
//			for (i=0; i<N-1; i++) {
//				if (current_val<old_samples[i]) {
//					peak = 1;
//					break;
//				}
//			}
//			if (peak==1){
//				oscillations++;
//			}
//		}
//		else {
//			Current_Graph1_Voltage = ADC_conv(GRAPH1);
//			Current_Graph1_Voltage = Convert_ADCtoVoltage
(Current_Graph1_Voltage);
//			Systolic_Pressure =
Convert_VoltagetoPressure(Current_Graph1_Voltage);
//			LCD_Send_String( mybuff );
//			run =0;
//		}
//	}
//}
//return Systolic_Pressure;
//}
/*****************************************************************/
/*while (run==1){
	if(oscillations <= 5){
		current_val = ADC_conv(GRAPH2);
		for(i=0; i<N-1; i++){
			old_samples[i]= old_samples[i+1];
		}
		old_samples[N-1] = current_val;
		peak = 0;
		for (i=0; i<N-1; i++) {
			if (current_val<old_samples[i]) {
				peak = 1;
				break;
			}
		}
		if (peak==1){
			oscillations++;

		}
	}
	else {
		Current_Graph1_Voltage = ADC_conv(GRAPH1);
		Current_Graph1_Voltage = Convert_ADCtoVoltage (Current_Graph1_Voltage);

		Systolic_Pressure = Convert_VoltagetoPressure(Current_Graph1_Voltage);
		LCD_Send_String( mybuff );
```

```
                run =0;
        }
}
        return Systolic_Pressure;
}
*/
/*********************************************************************/
/*              MEAN ARTERIAL PRESSURE (trial & error)              */
/*********************************************************************/
/*
        int max_val = 0;
        int max_raw = 0;
        double current_val = 0;
        int current_raw = 0;
        int run = 1;
        int c =0;
        int MAP;
        char mybuff[20] = "MAP = ";
        double Current_Graph1_Voltage;
        int GRAPH1 = 1;
        int GRAPH2 = 2;

        int run = 1;
        char is_peak;
        int i, this_val, this_raw;
        int last_val[10], last_raw;
        int last_peak_val[2], last_peak_raw;
        int this_avg_peak_val, this_avg_peak_raw;
        int last_avg_peak_val=0, last_avg_peak_raw;
        int peak_count=0, sample_count=0;
        double Current_Graph1_Voltage;
        int MAP;
        char mybuff[20] = "MAP = ";

        while (run==1) {
                this_val = ADC_conv(2);
                this_raw = ADC_conv(1);

                if (i<10) i++;
                else {
                        i=0;


                sample_count++;

                is_peak = 0;
                if (sample_count>=10 && this_val<last_val[0]) {      //we guess last_val[0]
is the peak
                        is_peak = 1;
                        for (i=1; i<10; i++) {
                                if (last_val[0]<last_val[i]) { is_peak = 0; break; }
                        }
                }

                if (is_peak) {
                        peak_count++;
                        if (peak_count>=3) {  //need at least 3 peaks to compute average
                                this_avg_peak_val =
(last_peak_val[0]+last_peak_val[1]+last_val[0]);
                                this_avg_peak_raw = last_peak_raw;
                                if (this_avg_peak_val<last_avg_peak_val) {   //done (avg
peak now going down)
                                        run = 0;
                                        delay_15ms();                //GO HOME
                                        PORTBbits.RB3 = 0;           //Set RS
                                        PORTD = 0b00000001;
                                        PORTBbits.RB1 = 1;           //Enable H->L
                                        PORTBbits.RB1 = 0;
                                        LCD_Send_String( mybuff );
                                        Current_Graph1_Voltage =
Convert_ADCtoVoltage(last_avg_peak_raw);
```

```
                                    MAP =
Convert_VoltagetoPressure(Current_Graph1_Voltage);
                              }
                              last_avg_peak_val = this_avg_peak_val;
                              last_avg_peak_raw = this_avg_peak_raw;
                       }
                       last_peak_val[1] = last_peak_val[0];
                       last_peak_val[0] = last_val[0];      //last_val[0] is the current
peak
                       last_peak_raw = last_raw;
               }

               for (i=9; i>0; i--) last_val[i] = last_val[i-1];
               last_val[0] = this_val;
               last_raw = this_raw;
       }

//     int c_temp, i;
//     char to_hex[16] = "0123456789abcdef";
//     char is_peak;
//     int last_peak_index=0, samples=0;
//     int peaks=0;

       while (run ==1){
               current_val = ADC_conv(GRAPH2);
               current_raw = ADC_conv(GRAPH1);
//             samples++;

               if(current_val > max_val){
                       max_val = current_val;
                       max_raw = current_raw;
               }
               //c =
Convert_VoltagetoPressure((double)Convert_ADCtoVoltage(current_raw));
//
//             if (current_val > 500 && (samples-last_peak_index)>1000) {
//                     last_peak_index = samples;
//                     peaks++;
//
//                     LCD_Send_Data('p');
//             }
               //if (c <= 50 && max_val!=0){
               if (current_raw <= 143 && max_val!=0) {      //exit condition
                       run = 0;

                       delay_15ms();                 //GO HOME
                       PORTBbits.RB3 = 0;            //Set RS
                       PORTD = 0b00000001;
                       PORTBbits.RB1 = 1;            //Enable H->L
                       PORTBbits.RB1 = 0;

//                     for (i=7; i>=0; i--) {
//                             c_temp = 0xf & (peaks>>(4*i));
//                             LCD_Send_Data(to_hex[c_temp]);
//                     }
                       Current_Graph1_Voltage = Convert_ADCtoVoltage (max_raw);
                       MAP = Convert_VoltagetoPressure(Current_Graph1_Voltage);
                               LCD_Send_String( mybuff );
               }
       }
*/
/***************************************************************************
       int num_max=3;
       int max[3]={0,0,0};
       int max_ori[3]={0,0,0};
       int temp;
       int i = 0;
       int j =0;
       int tmp =0;
       int current_ori=0;
       double syst_v=0;
```

```
        int MAP_p;
   int MAP_v;
        int avg_MAP;
        int run1 = 1;
        int peak =0;


        while(run1 == 1){

                LCD_Send_String( mybuff );
                current_val = ADC_conv(GRAPH2);
                max[0]=current_val;
                current_val = ADC_conv(GRAPH2);
                max[1]=current_val;
                current_val = ADC_conv(GRAPH2);
                max[2]=current_val;
                LCD_Send_Data('a');


                ////////bubble sort for max values
                for (i = 1; i < n; ++k) {
             int key = a[k];
                    int i = k - 1;
                    while ((i >= 0) && (key < a[i])) {
                      a[i + 1] = a[i];
                      --i;
                    }
                    a[i + 1] = key;
                  }
}
                while(run==1)  {
                        LCD_Send_Data('w');

                                current_val = ADC_conv(GRAPH2);

                                if(current_val>max[2]){
                                        max[0]=max[1];
                                        max[1]=max[2];
                                        max[2]=current_val;

                                        current_ori = ADC_conv(GRAPH1);
                                        max_ori[0]=max_ori[1];
                                        max_ori[1]=max_ori[2];
                                        max_ori[2]=current_ori;
                                }

                                c =
Convert_VoltagetoPressure((double)Convert_ADCtoVoltage(current_ori));

                                if (c <= 50 && max_val!=0){

                                        delay_15ms();                   //GOTO 2ND LINE
                                        PORTBbits.RB3 = 0;              //Set RS
                                        PORTD = 0b10101000;
                                        PORTBbits.RB1 = 1;              //Enable H->L
                                        PORTBbits.RB1 = 0;

                                        LCD_Send_String( mybuff );
                                        LCD_Send_Data('e');
                                        run =0;
                                }
                        }
                        avg_MAP=(max_ori[0]+max_ori[1]+max_ori[2])/3;
                        avg_MAP = Convert_ADCtoVoltage(avg_MAP);
                        MAP = Convert_VoltagetoPressure(avg_MAP);
                        run1 =0;

                        return MAP;
                }
        }
```

7.1 References

[1] American Heart Association, “Blood Pressure Testing and Measurement” [Online] http://www.americanheart.org/presenter.jhtml?identifier=4470 [Accessed: Oct 3, 2008]

[2] B.A Friedman, D.L. Bordon, R. Medero, “Blood Pressure and Pulse Oximeter” U.S. Patent 5,509,908, May 10, 1994

[3] Biological Sciences, BioMed 108 Human Physiology “Cardiac Output” [Online] http://www.biosbcc.net/doohan/sample/htm/COandMAPhtm.htm [Accessed: March, ‘09]

[4] Blood Pressure and Hypertension, “Auscultatory Method” [Online] http://www.blood-pressure-hypertension.com/how-to-measure/measure-blood-pressure-8.shtml [Accessed: Oct. 2, 2008]

[5] Blood Pressure and Hypertension, “Oscillometric Method” [Online] http://www.blood-pressure-hypertension.com/how-to-measure/measure-blood-pressure-8.shtml [Accessed: Oct. 2, 2008]

[6] Blood Pressure Monitoring, “History of Blood Pressure Measurement” [Online] http://www.medphys.ucl.ac.uk/teaching/undergrad/projects/2003/group_03/history.html [Accessed: March, 2009]

[7] Blood pressure monitoring “White Coat Hypertension” [Online] http://www.blood-pressure-monitoring.org/white-coat-hypertension.htm [Accessed: April, 2009]

[8] C.S. Chua, Siew Mun Hin, “Digital Blood Pressure Meter” [Online] www.freescale.com [Accessed: October, 2008]

[9] CBC News “Family Doctors” [Online] http://www.cbc.ca/news/background/healthcare/familydoctors.html[Accessed Jan, 2009]

[10] CFP-MFC “Waiting for a family doctor” [Online] http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1949098[Accessed Jan, ‘09]

[11] College of Family Physicians of Canada. Public opinion poll on physician wait time [Decima Research poll conducted for the College] Mississauga, Ont: College of Family Physicians of Canada; 2006.

[12] College of Family Physicians of Canada. When the clock starts ticking: wait times in primary care [discussion paper] Mississauga, Ont: College of Family Physicians of Canada; [Online] http://www.cfpc.ca/local/files/Communications/Wait_Times_Oct06_Eng.pdf [Accessed Jan, 2009].

[13] Dr. Xun Li, McMaster University Professor

[14] Glen Elert, The Physics Factbook “Pressure in the Human Circulatory System” 2001.

[15] H.L. Platt, A.M. Shell, V. Jankov, “Combined Wrist Blood Pressure and ECG Monitor.” U.S. Patent 2007/0100247 A1, May 3, 2007.

[16] HA17741 Data Sheet
http://www.datasheetcatalog.com/datasheets_pdf/H/A/1/7/HA17741.shtml

[17] Harvard Apparatus, A Havard Bioscience Company, “ECG/Pulse Oximeter” [Online].
Available:http://www.harvardbioscience.com/webapp/wcs/stores/servlet/BRSSearch?catalogId=11051&storeId=10001&productId=41217&parent_category_rn=&top_category=&isDoc=&langId=1&pageId=ProductDetail&division=HAI&ftCatalogId=10053&searchMethod=ftSearch&searchType=ALL&searchQuery=ECG%2FPulse+Oximeter&tabSelect=references#fulldescriptionsearchtab [Accessed: Oct. 2, 2008]

[18] Hamzah Qureshi, McMaster University Student

[19] High Blood Pressure – The Silent Killer [Online]
http://www.weightlossresources.co.uk/body_weight/blood_pressure.htm [Accessed Jan, 2009]

[20] High Blood Pressure Symptoms – Spotting the Silent Killer [Online]
http://ezinearticles.com/?High-Blood-Pressure-Symptoms---Spotting-the-Silent-Killer&id=1837514 [Accessed Jan, 2009]

[21] Hitachi HD44780
http://www.alldatasheet.com/datasheet-pdf/pdf/63673/HITACHI/HD44780.html

[22] Jaspreet Singh, M M Nayak1, K Nagachenchaiah, “Linearity and Sensitivity Issues in Piezoresistive Pressure Sensors” [Online]
http://www.sclindia.com/scl_society/Recent%20Publications/Pressure%20sensnsor.pdf [Accessed: March, 2009]

[23] L7805CV Voltage Regulator
http://www.datasheetcatalog.com/datasheets_pdf/L/7/8/0/L7805.shtml

[24] LCD Info Page [Online]
http://www.geocities.com/dinceraydin/lcd/index.html [Accessed: Jan, 2009]

[25] LCD interfacing with Microcontrollers tutorial [Online]
http://matidavid.com/pic/LCD%20interfacing/initialization.htm [Accessed: Jan, 2009]

[26] Medicine Net, “Definition of Diastolic” [Online] http://www.medterms.com/script/main/art.asp?articlekey=16164 [Accessed: Oct 4, 2008]

[27] Medicine Net, “Low Blood Pressure(Hypotension)” [Online] http://www.medicinenet.com/low_blood_pressure/article.htm [Accessed March, 2009]

[28] Medicine Net, “Definition of Systolic” [Online] http://www.medterms.com/script/main/art.asp?articlekey=16163 [Accessed: Oct 4, 2008]

[44] MPX5050GP Data Sheet http://www.datasheetcatalog.com/datasheets_pdf/M/P/X/5/MPX5050GP.shtml

[29] Oscillometry [Online] http://www.robots.ox.ac.uk/~neil/teaching/lectures/med_elec/notes8.pdf [Accessed: October, 2009]

[30] PC 1602-L Data Sheet http://www.datasheetsite.com/datasheet/PC1602

[31] PIC18F452 Data Sheet http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010296

[32] Pressure [Online] http://hyperphysics.phy-astr.gsu.edu/hbase/press.html [Accessed: Jan, 2009]

[33] Richard E. Klabunde, Ph.D., Cariological Physiology Concepts “Mean Arterial Pressure” [Online] http://www.cvphysiology.com/Blood%20Pressure/BP006.htm [Accessed: Jan, 2009]

[34] Rudolf A. Hatschek, “Blood pressure measuring device and method” US Patent 5309916, May 10, 1994

[35] Shu-Mei Wu, Xing Ou-Yang, Yeong-Dar Chen, Tung-Chuang Jan, Chao-Wang Chen, “Method and Apparatus for Oscillometric Blood-Pressure Measurement” US Pataent 6,458,085, Oct 1, 2002

[36] Stages of Hypertension [Online] http://www.everydayhealth.com/hypertension/understanding/stages-of-hypertension.aspx [Accessed: April, 2009]

[37] Starfield B. The importance of primary care to health. Med Reporter. 1999. [Online] http://medicalreporter.health.org/tmr0699/importance_of_primary_care_to_he.htm. [Accessed Jan, 2009]

[38] W. Wattanapanitch, W. Suampun “Portable Digital Blood Pressure Monitor” [Online] http://www.people.cornell.edu/pages/ws62/ [Accessed: October, 2008]

[39] What is a High Pass Filter? [Online] http://www-k.ext.ti.com/SRVS/Data/ti/KnowledgeBases/analog/document/faqs/hp.htm [Accessed: Jan, 2009]

[40] Wikipedia, “Korotkoff Sounds” [Online] http://en.wikipedia.org/wiki/Korotkoff_sound [Accessed: Oct 2, 2008]

[41] Wikipedia, Blood Pressure “Home Monitoring” [Online] http://en.wikipedia.org/wiki/Blood_pressure [Accessed: October, 2008]

[42] Wikipedia, Blood Pressure “Measurement” [Online] http://en.wikipedia.org/wiki/Blood_pressure [Accessed: March, 2008]

[43] W.J. Kaspari, H. Wong, J.L. Kirch, “Blood Pressure Measuring Apparatus” U.S. Patent 4,058,177, Nov. 15, 1977

VITA

NAME: Doralice Ferreira
PLACE OF BIRTH: Toronto, Ontario
YEAR OF BIRTH: 1987
SECONDARY EDUCATION: Dante Alighieri Academy (2001- 2005)
HONOURS and AWARDS: Dr. Harry Lyman Hooker Scholarship 2008
McMaster President's Award 2005
Nortel Networks Scholarship 2005
Dean's Honour List 2008, 2009
Governor General Award of Academic Excellence

Doralice Ferreira is currently completing level IV of Electrical and Biomedical Engineering at McMaster University in Hamilton, Ontario. She is a Real Estate Sales Representative currently working for HomeLife/ City Hill Real Estate Brokerage in Toronto, Ontario.