Intraocular Pressure Measurement In Eye By Using Disposable Solid State Pressure Sensor

Anvesh Loka Loka
University of Alabama at Birmingham

INTRAOCULAR PRESSURE MEASUREMENT IN EYE BY USING DISPOSABLE SOLID STATE PRESSURE SENSOR

by

ANVESH LOKA

ABIDIN YILDIRIM, COMMITTEE CHAIR
ARIE NAKHMANI
MOHAMMAD HAIDER

A THESIS

Submitted to the graduate faculty of The University of Alabama at Birmingham,
in partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering

BIRMINGHAM, ALABAMA

2017

INTRAOCULAR PRESSURE MEASUREMENT IN EYE BY USING DISPOSABLE SOLID STATE PRESSURE SENSOR

ANVESH LOKA

ELECTRICAL AND COMPUTER ENGINEERING

ABSTRACT

There are many eye infections and eye diseases like Diabetic Retinopathy, Glaucoma, Age-related Macular Degeneration, Cataract, etc., which cause eye blindness. Glaucoma is one of the major eye diseases that can cause complete blindness. According to National Eye Institute (NIH), there are 2.72 million people are suffering from Glaucoma in the U.S. The primary cause for Glaucoma is higher Intraocular Pressure (IOP), so detecting IOP in the early stages is a vital aspect in the assessment of patients at hazard from glaucoma. The instrument used to measure IOP is called Tonometer and usually used for clinical purposes. However, some part of the Glaucoma research in the research labs tonometer is not practical. Therefore, there is a need to design and develop an IOP measuring device. This device should also have additional features to save the experimental data for later analyzing. Therefore, the primary objective of the current work is to design, develop, assemble, and to validate an electronic measuring device that would help the research on an eye in particular Glaucoma.

As a thesis study work, we have designed and developed a measuring device, which can show the real-time sensor data with a Graphical User Interface. The IOP measuring device uses commercially available disposable blood pressure sensor (5uV/V/mmHg), to measure IOP. The IOP device has self-calibration feature to offset any measurement errors due to elevation changes, or errors due to changes sensor sensitivity.

In addition to that, the device has a 4x20 (4 lines 20 characters) LCD that shows basic measurement results in addition to external VGA display. It is a very useful feature if there is no external VGA monitor available. Finally, the device has the capability to save the data into Universal Serial Bus (USB) drive for later analyzing the data.

Keywords: IOP measurement, Glaucoma, Tonometer

DEDICATION

I dedicate this thesis to my family, friends who has provided support throughout my life and always had faith in my success.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

Figure *Page*

LIST OF ABBREVIATIONS

AD	Analog-Digital
DC	Direct Current
GPIO	General Input and Output
GUI	Graphical User Interface
HDMI	High-Definition Multimedia Interface
IC	Integrated Circuit
IOP	Intraocular Pressure
IP	Internet Protocol
I/O	Input and Output
LAN	Local Area Network
LCD	Liquid Crystal Display
NIH	National Institutes of Health
PC	Personal Computer
PCB	Printed Circuit Board
RC	Resistor-Capacitor
RPi	Raspberry Pi
USB	Universal Serial Bus
VNC	Virtual Network Connections
VGA	Video Graphics Array

CHAPTER 1

INTRODUCTION

We all know about the significance of the visual sensations since we born. Some research data analysis shows that around 90% of the knowledge is acquired through the visual structure, by which human life greatly depends on [1]. Vision System is one of the vital and complicated parts of the human organic framework. The visual data prepare begins in the retina, which is the accepting or light-sensitive part of the eye [2].

There are many eye infections and eye diseases like Diabetic Retinopathy, Glaucoma, Age-related Macular degeneration, Cataract, etc., which cause eye blindness [3]. The major cause for Glaucoma is Higher IOP [4]. Therefore, detecting IOP is the vital aspect in the assessment of patients at hazard from glaucoma. To find out the different parameters of IOP, and factors affecting the IOP requires a dedicated research laboratory. This present thesis is to develop a device which can be used in the research environment study of the different IOP parameters.

1.1 Objectives and Goals

The main objective of the current work is to design, develop, assemble, and to validate an electronic measuring device that would help the research study on the IOP of eye. Inside this thesis work, the previous models have been examined which are best fitted for the clinical purposes. Generally, tonometers are used for the clinical research about glaucoma, however, it's not practically applicable with the research lab that experiments with donor's eyes. So, there is a need to design an IOP measuring device for that purpose with some additional features to record their experimental data so that they can allow to analysis the data later. To detect Glaucoma in early stages, the pressure inside eye or IOP plays a vital role. IOP is the vital aspect in the assessment of patients at hazard from glaucoma. This device should meet all the standards and it should be very easy to use with a sleek design.

1.1.1 Visual Process

To better comprehend the present thesis, a concise discourse of the human visual framework and its essential segments will be helpful. This study will cover only the basic anatomy of the eye along with some of its other components.

The capacity to see is dependent on the activity of various structures in and around the eye. Figure 1 demonstrates the basic parts of the eye's optical framework. When an eye focus on any object, the light rays get back from that object to the retina, which is located inside the eye and it is the place where optical imaging process starts. The visual data from the light beams are bowed, refracted, and centered on the focal point. The auto adjusting lens will confirm that the light beams will fall on the retina with sharp focus, still, the subsequent picture is upside down on retina. At the retina, the light beams are changed over to electrical impulses. These will transmit to cerebrum which is inside brain, through the optic nerve [5,6,7]. Inside the brain, the image is rotated again upside down so the brain can interpret the image with the exact object which is there outside. The lens and the cornea will auto adjust to focus on the object near and far so that they will keep maintaining the light beam centered on the focal point. If the focal point moves towards the eye lens then, they eye is suffering from near sight vision. By suing concave lens, we can correct the focal point onto the retina. This is the same principle with the far-sighted eye, we need to use a convex lens to keep the focal point on the retina which is usually far from the eyeball.

1.1.2 Intraocular Pressure

The increase in IOP leads to the Glaucoma eye disease. IOP is the pressure applied by the visual liquid called Aqueous Humor which is present in the foremost chamber of the eye.

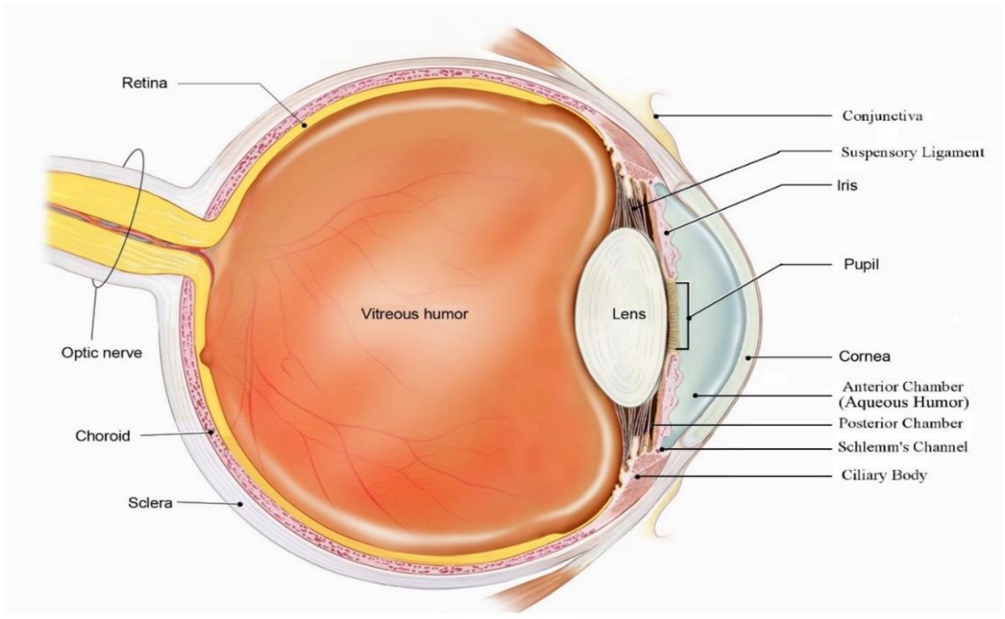


Figure 1. The anatomy of human eye ball

Ciliary body produces aqueous humor of the eye in the region of posterior chamber and moves to the anterior chamber through the pupillary opening (Figure 1) [8]. Then again from the anterior chamber, the fluid drains out through two disparate paths.

Most of the fluid leaves the eye by means of the trabecular meshwork, Schlemm's channel & episcleral veins (shown in Figure 2) [8]. The left-over fluid (around 10%–20%) leaves through the Uveoscleral path where the fluid transmits among the ciliary muscle groups [9].

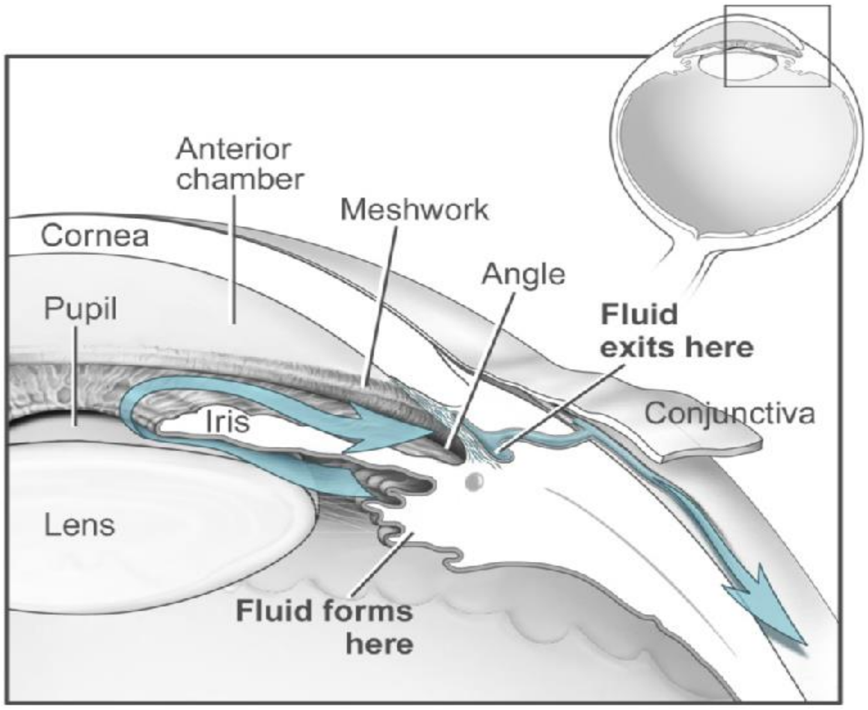


Figure 2. Intraocular fluid flow of path for a normal eye

Increased IOP directly means there is something blocking the drainage way which will increase the pressure in vitreous humor causing damage to optic nerve. If it's not treated on the right time, there are chances to get blind completely. Typical IOP is in the scope of 10-21 mmHg. On a broad view, IOP’s between 10-21 mmHg are considered to be normal with an average mean of 16 mmHg. The risk of developing glaucoma is much more when IOP is more than 23 mmHg compare to average IOP 16 mmHg [10]. IOP is more than normal in most of the Glaucoma patients, due to the condition that the aqueous fluid outflow is much less when compared to net inflow. This condition will occur when something is preventing the flow of fluid in the drainage way. If the “angle” shown in Figure 2 changes than the regular or they are any blocking to the drainage path, then IOP will start increase. Current glaucoma treatment methods are based on the principle to reduce the intraocular pressure and to slow down the progress of glaucoma effect. Hence, there should be some accurate tests and early methods to measure IOP. So, that there is a chance to treat patients with necessary initial treatment and diagnosis.

1.2 Background Study

1.2.1 Glaucoma

Glaucoma is the second major cause for vision losses [11]. Glaucoma is the combination of several diseases of the eye illnesses which results in harm to the optic nerve [12], and if it's not treated well, it can cause even blindness. Glaucoma is typically a disease with no symptoms before the advanced nerve optic damage occurs. So, investigating of glaucoma in its early stages and treatment is needed to prevent Glaucoma related diseases [13]. The most widely occurring one is open-angle glaucoma and with less occurring ones include closed-angle glaucoma and normal tension glaucoma. Open-angle glaucoma grows gradually after some time but with no symptoms. But, closed edge glaucoma can introduce step by step or suddenly [9].

Current medication is conducted to reduce IOP through medications. Eye drops are the basic medication to lower IOP. These will reduce the aqueous fluid excretion or by cleaning the drainage path to allow the fluid to move out from eye easily. The fluid will drain out via trabecular meshwork and Uveoscleral route. Regrettably, a good measuring test is not available till today. However, Goldmann Applanation Tonometry is widely in use for a long time, but it will not produce satisfactory results and sensitivity [14,15,16].

1.2.2 Statistical Numbers of Glaucoma

Glaucoma is one of the leading causes of blindness. There was more than 161 million people are with a visual disability, of whom about 37 million people are blind in 2002 [17]. The updated statistics shows around 60.5 million people suffered from glaucoma worldwide in 2010 [18], which shows that the people who are suffering from glaucoma is increased by 22% in 8 years of span. The mathematical calculation shows that the glaucoma disease is widespread and affected as 1 in each 131 persons in the USA. The financial effect is in the Estimation range of 2.4 billion to 3 billion US dollars annually for the treatment of glaucoma [19]. A statistical analysis of National Institutes of Health(NIH) shows around 120,000 Americans is visually impaired due to Glaucoma which represents 9-12% of the occurrences of blindness in the U.S. NIH also predicted that 79.6 million people are assumed to experience the ill effects of glaucoma by 2020, increasing from 60.5 million in 2010 [20]. The most well-known type of glaucoma is Open-Angle glaucoma, constituting about 74.0% of cases in 2010 [18].

1.3 History of IOP Measurement

Eye specialists have not generally allied the increased intraocular pressure can cause vision loss from glaucoma disease. A few people seemed to have noticed the hardness of the eye in this condition as far back as the tenth century. Raised Intraocular pressure was not regularly surveyed until the late 19th Century. In 1865, Albrecht von

Graefe designed the first instrument for measuring intraocular pressure and introduced a new approach to cataract Surgery by practicing a modified linear removal method which significantly diminishes the rate of infection [21].

The first practicable precision measurement was the Malakoff applanation measurement of the late 19th century. It was in extensive use throughout east Europe. Later, indentation tonometer was widely used throughout the world which was developed by Schiötz until late 1960's. The perfectly précised tonometer is started in 1950 by Goldmann's Applanation tonometer developed by Goldmann, which is most widely used tonometer in the world. Mckay-mark tonometer, the pneumotonometer, and the air puff applanation tonometer are also gaining their importance. The most accurate of all tonometers is Dynamic Contour Tonometer because it is independent of the central corneal thicknessand have a lower inter and intraobserver variability [22,23].

There are some new other types of tonometer which don't require any topical anesthesia while measuring like transpalpebral tonometer. However, high IOP cannot be counted directly for diagnostic purposes, but the part of IOP in the administration of Glaucomatous optic neuropathy remains critical [24]. However, Measuring Devices for research purposes are extremely rare and too expensive if found one. They are many different types of tonometers are available, which is basically used for clinical purposes. So, that's our requirement to develop a device which could ease the researchers to study more about the eye.

CHAPTER 2

DESIGN PROCESS

To obtain a good output result from a project, the detailed procedure is necessary. From starting of the project to the end product, the project needs to consider in all aspects and regular testing, which could affect the developing, designing, manufacturing and testing of the final product.

The initial step is to decide the objective of the project. In simplified words, what does the project plan to achieve? The second step would be the examination of the objectives. Shortly, which results will be created by the completed project. The final step is to see what all the resources are available. Along with the additional question that what all the particular material and human resources will be necessary to get the project done?

The next crucial step is to determining a time frame. After each small particular task, we need to calculate the available time and preplan our tasks to be accomplished based on the time remaining.

Finally, the cost of the project is to examined. Is this cost effective?

In our consideration, we had an adequate budget with a sophisticated laboratory for developing the project. Under these circumstances, we have considered some of the following steps to develop the project.

2.1 Analyzing the existing model

Analyzed the previous model, which were used previously for clinical purposes. Although, there are some devices are used to measure IOP for research purposes, they are cumbersome to use and they need a lot of additional skills to setup the device. So, there is a need to develop a device which is inexpensive and should give the most accurate results and it should be easily portable.

2.2 Project Description

The current thesis involves designing, manufacturing, and testing an embedded microprocessor controlled measurement device, which allows the user to measure the IOP of an eye. The completed project doesn't require any specialized skills to set up the device.

2.5 Project Requirements

As mentioned in Chapter 1 and Chapter 2, the measurement of IOP plays a crucial role in diagnosing the Glaucoma in the early stages. There are some commercial tonometers but they are limited to clinical purposes with some inaccurate results [25]. So, after reviewing the current status, we decided to develop a device with the following minimum requirements.

1. Use a microprocessor, microcontroller, or embedded processor.
2. Use a reliable and affordable pressure sensor.
3. Implement a Graphical User Interface (GUI).
4. Use a thumb drive option to store data for later analysis.
5. Implementing a remote access to the device for further upgrades (optional).

In addition to this, we have added the additional Liquid Crystal Display (LCD) if we don’t have any external monitor. This LCD most of the information about the current measured IOP value, graphical code though we cannot see anything, and the pop-up messages of the information of transferring data into Univeral Serial Bus (USB) thumb drive.

2.5 Market Research

Currently, to study IOP and Glaucoma there are no dedicated IOP device to study Glaucoma with the donor's eye. Though there are some devices like AD Instruments – BP Amp, used to study IOP of animal models. There is a need of special setup for experimenting and measuring of the IOP with the donor’s eye with this device. It is not a standalone tonometer.

2.5 Design Considerations

There are many design parameters have to considered while designing a device. Due to the limited time and budget, there should be some précised design parameters by which we can gain the control over time and the design process. The three important parameters considered are:

2.5.1 Reliability

Overall the system reliability plays a vital role in this case because the user of this equipment is from the medical background and assuming they are inexperienced towards the operating of the electronic equipment. Therefore, for everyone operating and maintaining the device should be quite easy to understand. No complicated adjustments, calibrations on and around the device should be needed. It should be ‘plug in the device and ready to use’ process. Replacing the broken parts like the sensors should be easily performed without any special tools.

2.5.2 Budget

The Budget is an important factor in every project. Due to the limited research budget, the cost of the device is under $500.Tthe quality and reliability of the device shouldn’t affect. So, we compromised about the external display and shifted that budget in using more precise IC components.

2.5.3 Upgrade

There should be some easy mechanism to update the process in future. So, we have implemented the wireless access to Raspberry Pi (RPi) to update the firmware and to update the code by adding some additional hardware components. We can connect to RPi by using Real Virtual Network Computing (VNC) from your local machine (laptop or PC), But we need to have the same account logged in your machine and RPi and we can access RPi from anywhere across the globe.

CHAPTER 3

EXPERIMENTAL SETUP

3.1 Introduction

Measuring devices have been playing a crucial role in medical, engineering and industrial fields. The study of the IOP of the eye is also demanding for researchers and generally, needs an advanced tool to make the precision measurements. This chapter discusses the actual experiment setup in which the RPi has been included. The experimental setup is as shown in Figure 3 below.

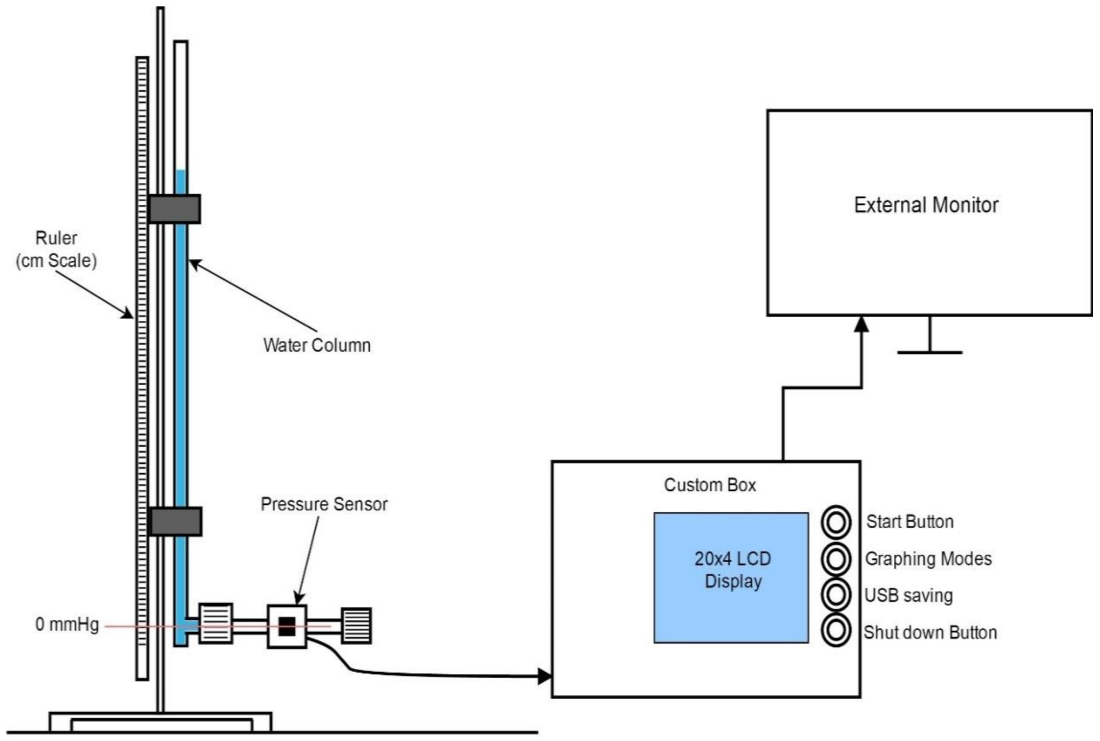


Figure 3. Experimental setup

The disposable solid state blood pressure has placed such a way that it is lined up 0 cm on the ruler as shown in Figure 3. The custom designed box will have all the electronic components inside it. Though the 20x4 LCD and the push buttons are on the front side of the box for user input and output. There is also an availability that we can connect to an external VGA display.

3.2 Functional Blocks

Functional block Diagram (Figure 4) shows how the experiment is set up and how all the data path from one component to the others. Firstly, the disposable solid state pressure transducer raw data is amplified. We have used the INA125P instrumentation amplifier followed by the OP-27 operational amplifier to amplify the signal such that the voltage level is good enough for AD converter. The amplified data if given to RC filter to reduce noise levels and to do necessary filtering.

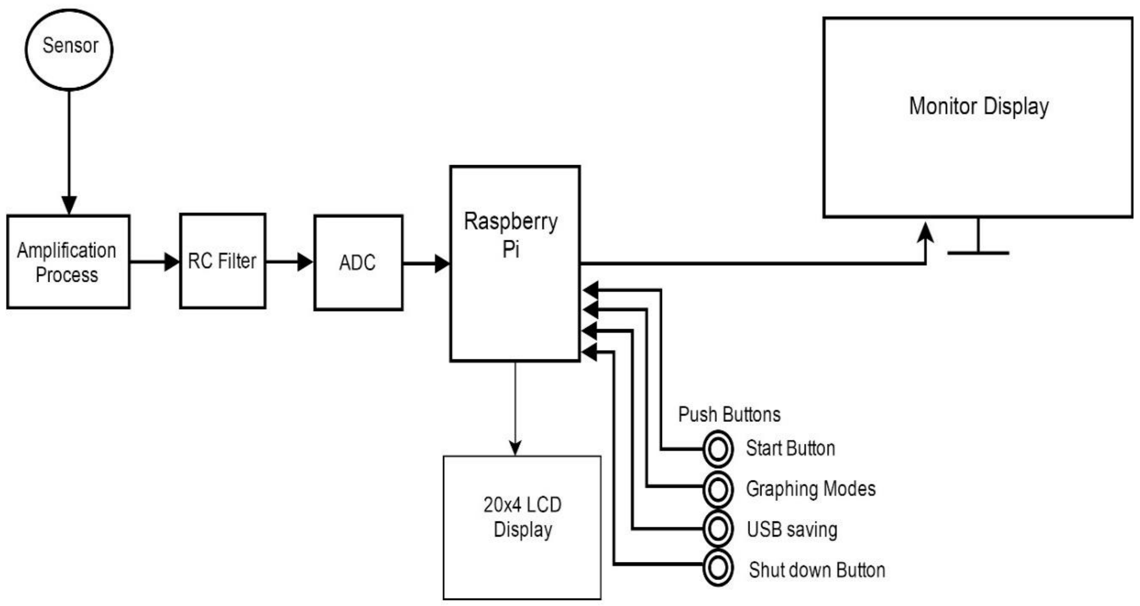


Figure 4. Functional block diagram

Since RPi will accept the input data only in 0's and 1's form, the filtering data is given to 12bit ADC for converting the analog values to digital values. Finally, RPi processes the data and will display on the 20x4 LCD and external VGA monitor if connected.

There are also 4 input push buttons for the RPi so that the user can give particular commands to the RPi through push buttons which are preprogrammed for particular tasks. The schematic diagram is shown as in Figure 5 below.

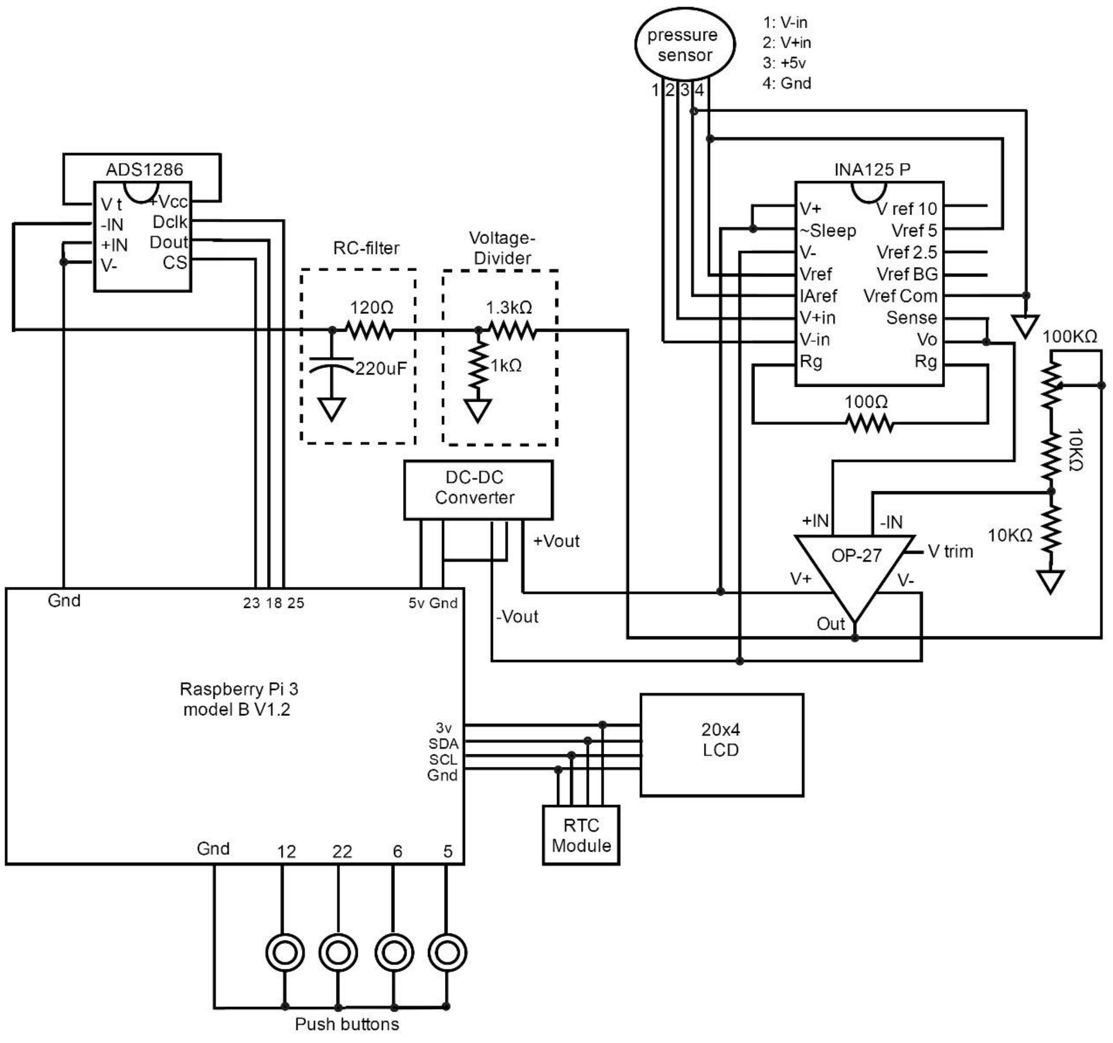


Figure 5. Circuit diagram

Although the LCD and RTC module are both connected to same pins of RPi (I^2C), each component will take different addresses.

3.2.1 The Proposed Custom Box Design

All the necessary components are embedded in a custom box. The custom box has a size of 10x10x3(LxWxH) inches. There is a sensor female connector on the front side and a power switch to turn on the device (Figure 6).

On the front panel, there is a 20x4 LCD i.e. 20 characters per line and 4 lines, onto the right side of the front panel. The LCD will display all the information such as the: currently measuring IOP value, IOP is in a low state or normal state or high state, all the necessary information about data saving to USB, all the account details of the VNC, during the experiment.

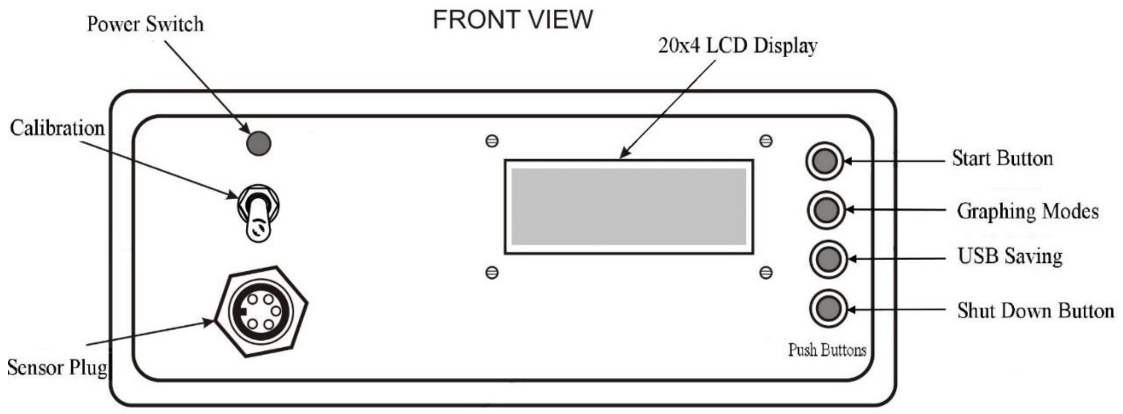


Figure 6. Front view of the Pressure Gauge

The customized box has four user inputs via 4 push buttons placed next to the 20x4 LCD. They have particular preprogrammed tasks such as:

1. The Start Button is used for starting the device.

2. The Graphing Modes button is used changing the different views of the plotting graphs.
3. The USB Saving button is used to save the experimented data into the thumb drive.
4. The Shut Down Button is used to shutdown the device.

There is an USB plugin slots are available at the back side of the device. It can be used to save the experimented results to memory sticks, which can be used later for analyzing purposes. The custom box is powered with AC source of 110v. The external monitor is also connected to this device through the VGA connector which is on the back panel as shown in Figure 7. LAN wire can also be connected to this device.

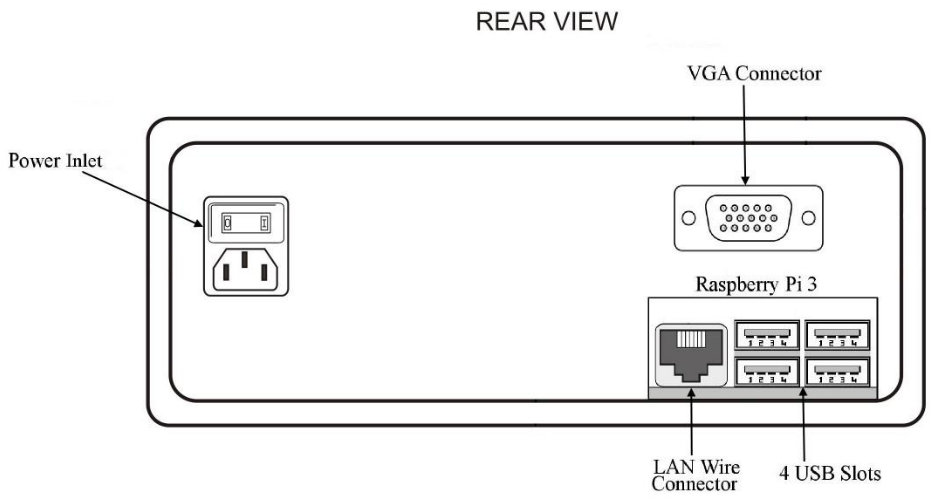


Figure 7. Rear view of the Pressure Gauge

3.2.2 PCB Design

Currently, all the circuit elements are present on the breadboard. A PCB design will be provided.

3.2.3 Water Column setup

3.2.3.1 Sensor

We have used the standard solid state disposable pressure transducer manufactured by Becton Dickinson. It will be connected to the tube on either side of the sensor (Figure 8), through which blood would be passing. The sensor sensitivity is 5μV/V/mmHg and the condition is that the sensor should be at the same height of the eye, so it will calculate the exact pressure within the eye.

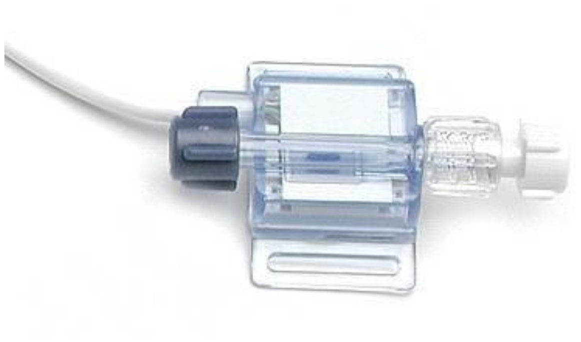


Figure 8. Solid-State pressure transducer

3.2.3.2 Water Column

Water Column setup is used to calibrate the device. Water column setup is as shown in Figure 9. We have used solid-state blood pressure transducer manufactured by Becton Dickinson which is attached to the lower part of the water column. The transducer has male/female Luer lock that helps easy to assemble and release air bubbles if it is necessary. These systems will have some outlet drainage systems in the middle of the tubes. If they are any air bubbles in the middle of the tubes they can let them go through that drainages systems.

As shown in Figure 9 there is a measuring ruler on the water column that starts from 0 cm upwards direction so that the user can observer the pressure according to the water level. The condition is that while experimenting, the eye cage device should be on the same height level with the pressure transducer.

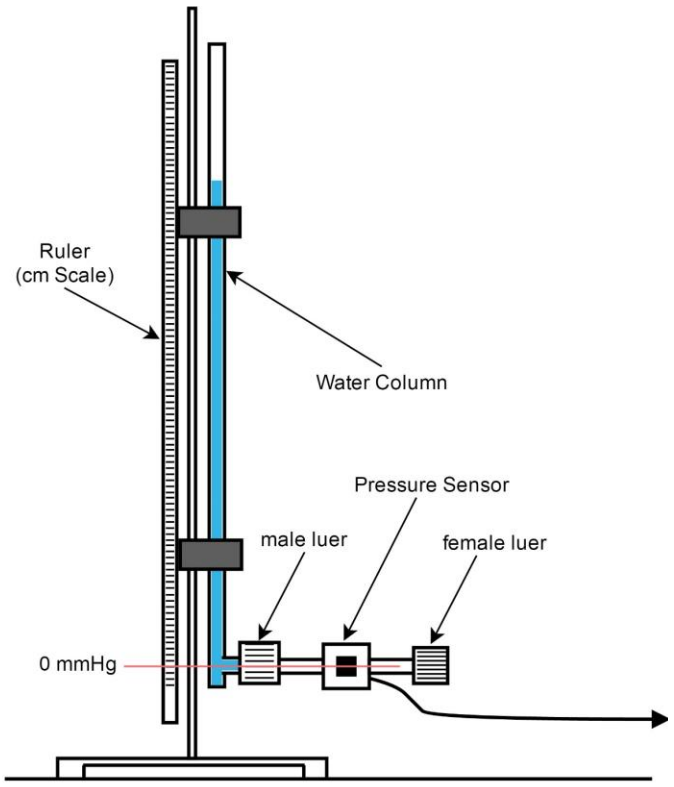


Figure 9. Water Colum setup

If they increase saline bag level, the pressure will increase and vice-versa during the experiment. They can use a saline bottle attached to the experiment so that they can change the pressure by changing the height of the bottle respectively.

CHAPTER 4

PRE-PROCESSING

This chapter describes the pre-processing of the signal transducer, which would be given to RPi as an input. There is three stage of the pre-processing of the signal as follows.

4.1 Amplification Process

Since the sensor produces 5µV/V/mmHg the signal from the transducer needs to be amplified. So, there is a two-stage amplification process involved.

The first stage we will amplify the signal from microvolts to millivolts with INA125P instrumentation amplifier and then the second stage we will again amplify the millivolts to volts with OP-27 operational amplifier, which is usable by the AD converter.

The first stage is to convert the microvolts to millivolts by using the Instrumentation amplifier INA125P. The unique feature for this amplifier is that it can provide the exact reference voltage of 2.5V, 5V, 10V. There is a need for this exact voltage reference for the sensor and AD converter because if there is any change in the supply voltage there will give an error in calculations. To avoid all the errors in reading, exact reference voltage reference is necessary. The precision voltage references give the flexibility to connect the solid-state pressure sensor and the AD converter directly to the INA 125 P.

The output after 1st stage amplification process is around 100 mV-200 mV. There is also an additional feature of the INA 125P to change the gain of the IC by changing the gain set of the resistor. To get the desired voltage of around 100 mV-250 mV, a gain of x550 is adjusted. Then the amplified signal is given as an input to the next amplifier, OP-27.

The second stage amplification is done with the operational amplifier. There should be precise amplification process with the low noise condition. OP-27 fits the requirements of amplifying the 100 mV-200 mV to the regular 0-5V so that raspberry can read this voltage after AD conversion. OP-27 is the amplifier used to get the necessary amplified voltage, with a gain of x10 times. OP-27 supplies input voltage to the Analog Digital (AD) Converter. The converted binary output is given to RPi. In the design phase, the amplifiers are tested with only single supply voltage. However, the outputs of the amplifiers are not linear, then we have changed the supply voltage to the dual power supply voltage. This increased the precision of the voltage references to the solid-state pressure transducer.

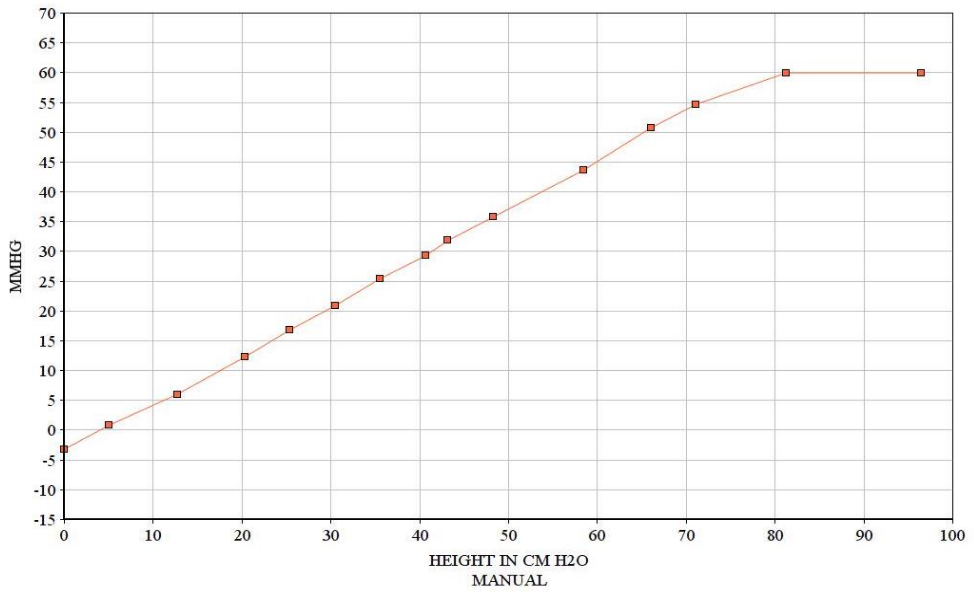


Figure 10. Linear output with pressure

These outputs from the amplification process is directly proportional to the pressure value from the sensor. The above figure shows the experimental results that there is a linear relationship in between the water in the water column and the output voltage. Generally, mmHg unit is used while doing the experiment, we should do a conversion between units. The conversion formula from mercury to water is as show below.

$$1\text{cm of } H_2O \text{ (water)} = 0.7355 \text{ mmHg}$$

So, if we have a water level of 25.4 cm then pressure on the sensor would be

$$P \text{ (mmHg)} = 0.7355 \times 25.4 \text{ cm} = 18.68 \text{ mmHg}$$

When the water level reaches the 83cm then after that the pressure is constant because it is the maximum value the sensor can handle. As we can see in above Figure 10. there is an offset value which is below the 0mmHg. We can calibrate this to 0 mmHg with the calibration knob provided on the front side of the device (please refer to 2.2.1).

4.2 RC filtering process

Due to the noisy environment, there is a need for filtering before sending the signal to AD converter. A RC filtering with the low-pass is implemented.

Low resistor value followed by the large capacitor will be able to get the smooth input to AD converter resulting in the noise free and clean AD converted values. The cutoff frequency of the low pass filter is 6.03Hz. The circuit diagram and the cutoff frequency is as shown in Figure 11. The formula for cut-off frequency is below.

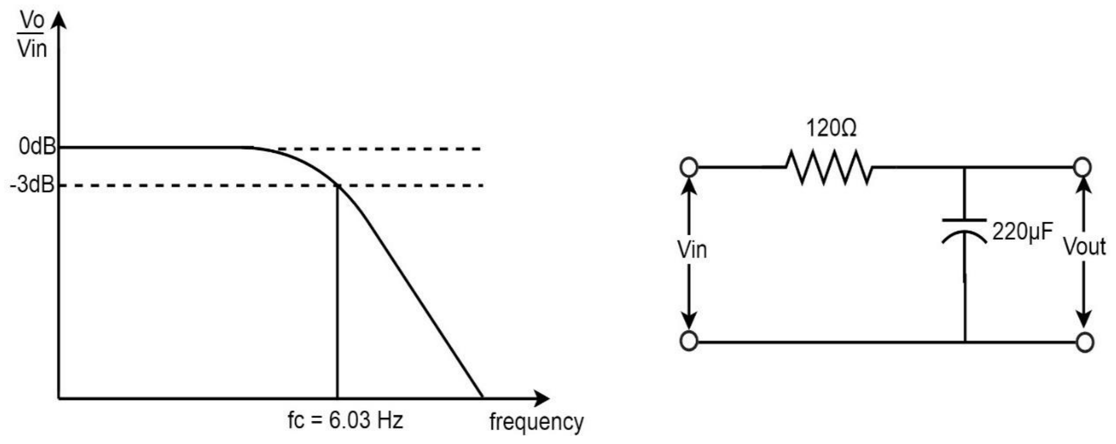


Figure 11. RC filter

$$f_c = \frac{1}{2\pi RC} \tag{4.2}$$

4.3 12-bit AD converter

There is a need to convert the analog voltage into the digital values because the microprocessor cannot understand the analog voltage levels. So, ADS 1286P 12-bit AD converter is used to convert the analog voltage to digital voltage. By using the 12-bit conversion we are expecting to eliminate the noise and conversion errors. So, the AD converter will get the input clock pulses from the RPi to synchronize and convert the analog signal to digital bits. The conversion speed depends on the clocking speed to AD converter. Though the concept is same as we need to keep the chip select pin low to put the AD converter in working mode or disable it. The converted bits then are given to RPi as a digital input. The formula to find out the AD converter value for the given input voltage is:

$$\frac{Resolution\ of\ the\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured} \tag{4.1}$$

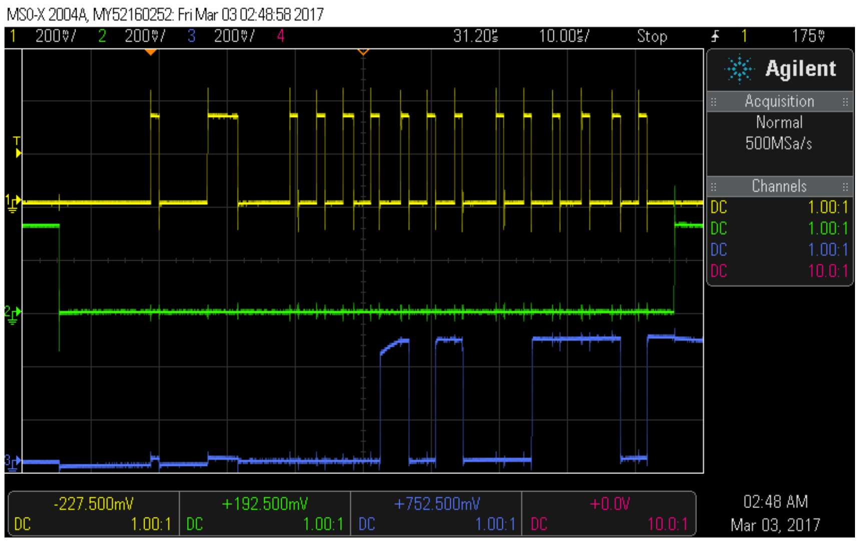


Figure 12. 12-bit AD converter

From the preliminary test, we have used the 4-channel Agilent oscilloscope to get the above image (Figure 12) and we measured the input voltage which is 0.407 mV and the resolution is 0 - 4095. So, by using the using the formula, we can find out the AD converter reading.

$$\frac{4095}{5} = \frac{AD\ converter\ reading}{0.407}$$

AD converter reading = 334_{10}

From Figure 10, if we calculate the binary values then it is 00101001110. If we convert the same binary values to decimal values then it is $(334)_{10}$, which is same as the value from the theoretical calculations. This is the precise way to convert the analog value to the decimal with minimum noise.

CHAPTER 5

IOP MEASUREMENT

5.1 Introduction

On the front panel of the customized box, there are push buttons by which we can control the whole measuring setup. We also do have a connector to which we can connect the solid-state blood pressure transducer. There is an LCD on the front side to see what is the status of the experiment; for instance, is the device running or not, what is the measured IOP value etc. On the back panel, we have the USB slots to attach thumb drives, by which the data will be saved if they have attached USB drive. If there is no thumb drive then, it will save in local folder of RPi. These small mechanical designs will fit the research laboratory environment with its small design. To note that this device is pretend to use only with donor eye. Although we didn’t use the developed device with the real donor eye, we used the device with water column and saline bottle.

5.2 Hardware Design

The measuring device consists of four major sections. They are the central processing unit, pressure sensor, Inputs and Outputs (I/O) pins, and output monitor. Left to the push buttons, there is an LCD which will show the status of the device and the different parameters which are running on the device. There is a Video Graphics Array (VGA) output on the back panel. Our basic work in this thesis was to develop an IOP measuring device with all the all the engineering skills and knowledge. This device will ease the researchers to study of IOP and parameters affecting IOP. A well-designed product should be very easy to use and it should be very easy to upgrade. The cost of the project is always one of the key factors. Our measuring device is under $500 which is much affordable than the one used for clinical purposes.

5.3 Embedded Processor

First, the selection of the microcontroller or microprocessor is very important to any project. There should be a careful consideration in selecting the processor because there are different parameters of each embedded board must be considered like the speed, performance, and durability. To make this selection process, we should consider different products having different I/O pins, processing power and different modes of usage. In addition to that, Manufacturer websites are also helpful in the selecting the microcontroller for our project.

Because we have 12 I/O line we can do that with Arduino. Though we will not get the enough processing power with Arduino, therefore, we have decided to go with RPi. We can save data files into USB drives. We have to save the file name with the current time and date, so we have used RPi with a real-time clock module connect to 4 I/O lines via I2C communication. The RPi 3 having 1.2 Ghz processor which is nearly 25 times faster than regular Arduino and when compared to the price it is under 40 dollars, which is affordable. RPi is a standalone microcomputer where we can perform any task that a regular laptop does and we can also connect to an external monitor through VGA which is a key part of our project. VGA is still the standard connector for almost all the monitors. Since RPi doesn’t have VGA connector we are using a cable converter to convert from HDMI signal to VGA.

RPi has 40 I/O pins with 1GB of RAM. The built-in Wi-Fi option enabled us to operate the Rpi with Remote Desktop Connection, from same network and real VNC, from anywhere in the world. This feature is very convenient in upgrading the project or adding additional features. So, RPi is the best choice for our project in terms of performance and compatibility.

5.4 Virtual Network Connections (VNC)

By remote desktop connections, we can easily upgrade, update our project or even we can easily add new features. Real VNC is a virtual network software. It allows accessing the RPi if it is on the same network or even from any other network. This option gives the flexibility off accessing the RPi from the anywhere in the world without concerning about the IP address or Port numbers. We need to create an account for this feature and the same account should be used on both devices.

This the best feature of VNC is, if some of the research collaborator who lives in different geographical location, then the experiment place, then they can collaborate to the experiment by using real VNC to see how the experiment is going.

5.5 Integrated Development Environment (IDE)

We have selected to use Python 2.x series because, currently it is the latest version of the IDE and it's released in mid-2010, which is having a lot of additional library packages than its predecessor. Python is a scripting language and python libraries offer a wide range of applications. It has the capacity to control all the functions of the operating systems and the device.

5.6 Libraries

We have used various libraries for our Graphical user Interface (GUI) and for many other background works. There was a need to used different kinds of libraries to get the date, time, OS and for to access General Input and Output (GPIO). There was also a necessity for the socket libraries to get the IP address from the RPi to show in LCD.

5.7 Graphical User Interface(GUI)

we have used *matplotlib, pyplot* library along with *Tkinter* to get graphical outputs with graphs and to show a digital value in colors. We added both the libraries together in the output to get the real-time graphing with the labeling. So, we have to import all the additional supporting libraries for matplotlib. *Matplotlib* library is an effective tool for the good Graphical User Interface(GUI).

We are also saving the data files in *".xlsx"* format. It is in Microsoft Excel with the two-column, saving the X axis and Y axis respectively. As it is shown in the Figure 11, there is the real-time graph on the upper side of the display and below that they are some additional data displayed. All the different parameters are at the bottom of the display. The data is as followed: different modes of graphs, the range of the IOP value and then the measured IOP value itself. If the measured value is in the range of 0-10 mmHg then it will show as "Low" with cyan blue color. If it's in the range of 10-22 mmHg then it will show as "Normal" in green color. Finally, if it's greater than 22 mmHg then it will show "High" with red color. We have taken the time scale on the X-axis and then pressure mmHg on the

Y-axis. Initially, the device starts with the 50 sec time scale. If the experiment crosses the 50 sec range it will get doubled to 100 sec. Accordingly, the last X seconds modes show the 5 sec time range. As the experiment crosses 50 sec then the last X seconds gets double to 10 sec. Below different graphing mode messages, there is also an option for the USB saving. If user press the data saving button based on the device condition a popup message will show respectively. If the USB is connected, then after pressing the button, it will show that the “file is saved to USB” for the first time and if it is pressed again it will show that the “file is Updated to the USB”. If no USB devices is connected, then it will show that “No USB devices Found”. The example below explains the steps.

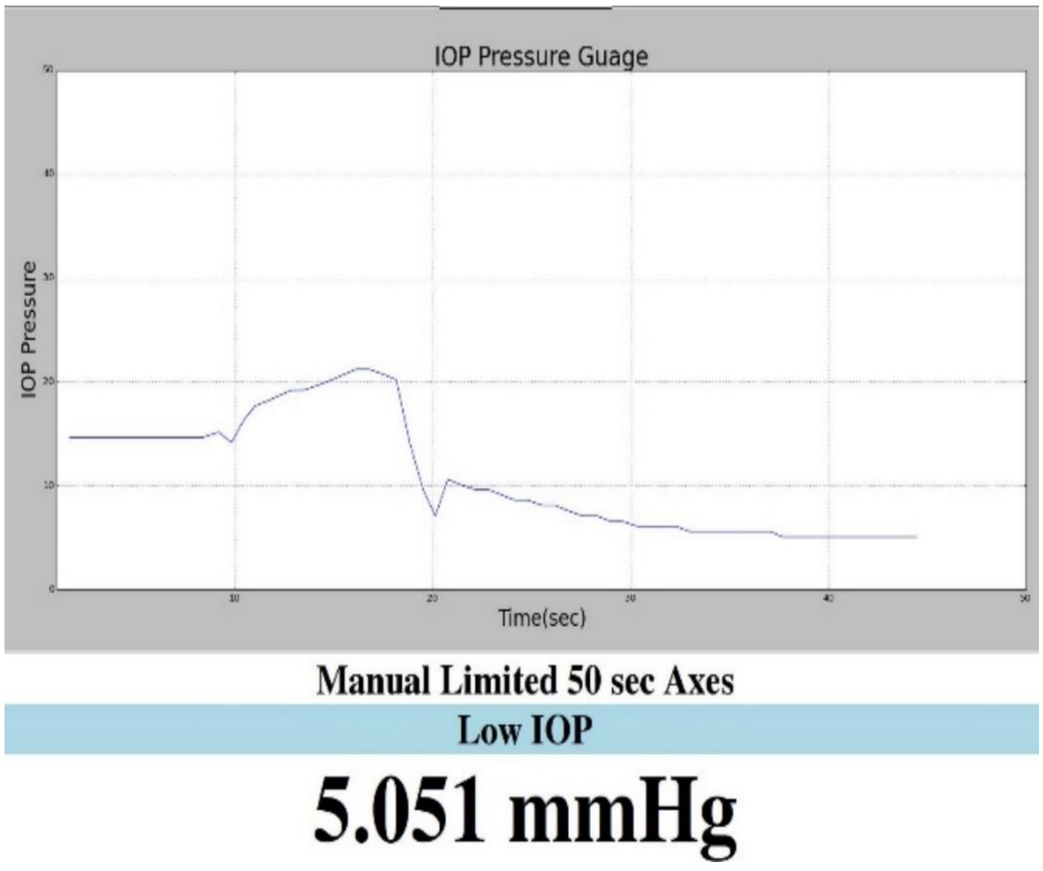
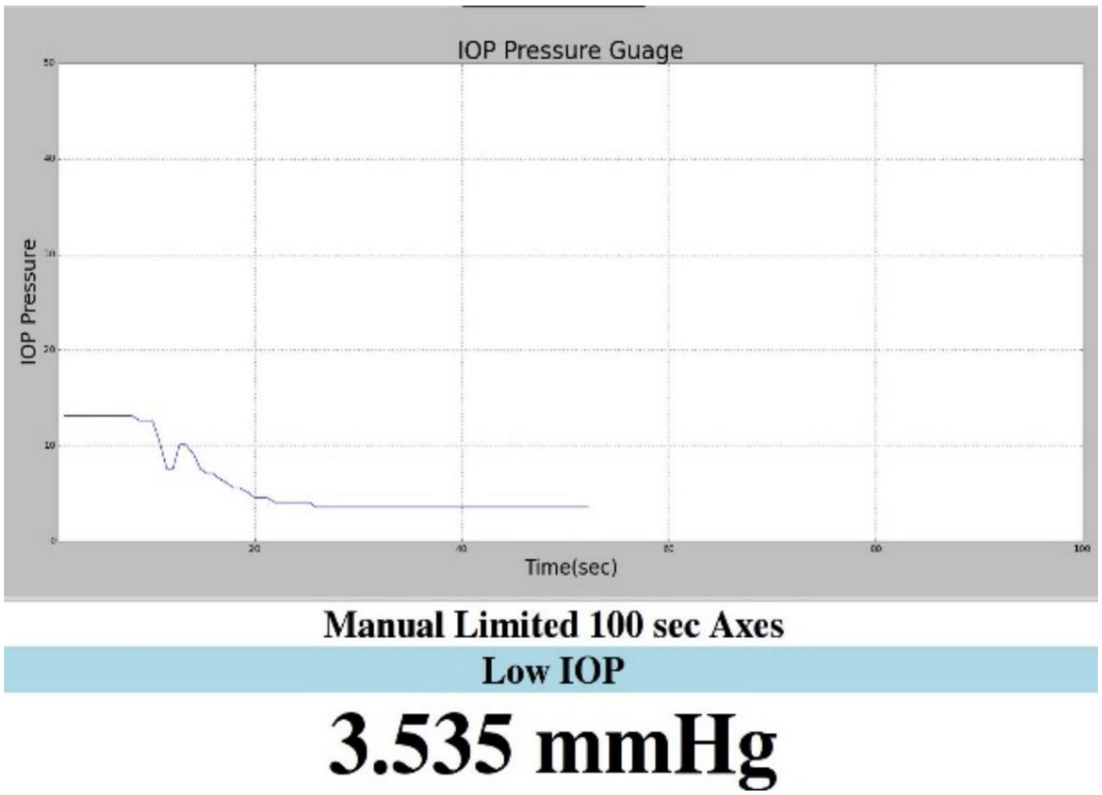


Figure 13. GUI

As shown in Figure 13, the device starts plotting the graph with 50 sec of range on X-axis. The value measured is less than 12mmHg, so it displayed as “Low IOP” in cyan blue color. Then after some time when the 50 sec range is exceeded then it will automatically increase its range of plotting to 100 sec as shown in Figure 14. Then we changed the graph plotting mode by pressing the Graphing mode button. Then it will display the “Auto Scale Axes” as shown in Figure 15. This auto scale plotting mode will always show the time range of 25 sec, irrespective of the length of the experiment.

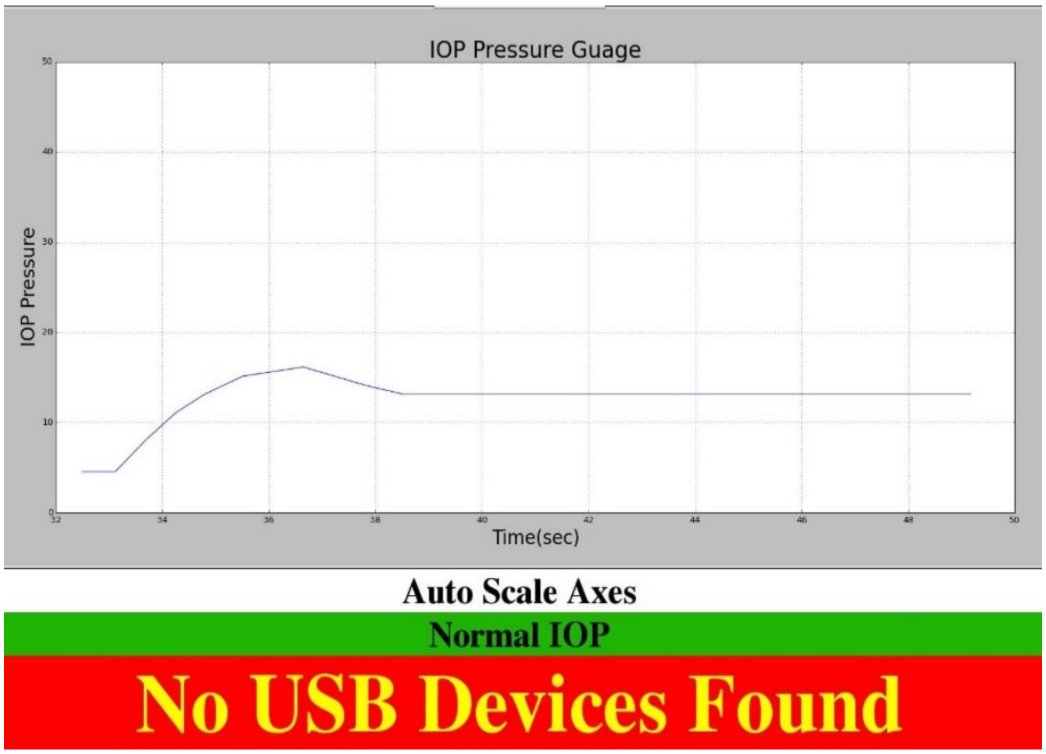


Manual Limited 100 sec Axes

Low IOP

3.535 mmHg

Figure 14. GUI with double time



Normal IOP

No USB Devices Found

Figure 15. GUI with no USB

The measured value here is greater than 12mmHg so the data is displayed as “Normal IOP”. As we tried to transfer data to USB without connecting one, As a result, the last message is displayed as “No USB Devices Found” to transfer the data.

Then we pressed again the graphing mode button is pressed so the time display mode is “Last 5 sec. And we decreased the water level in the column, so the pressure is dropped to “Low IOP” as shown in Figure 16. Then we connected the USB drive and tried to save the data into USB drive. The data transferred to USB drive with the pop up messaged showed as “File Saved to USB”.

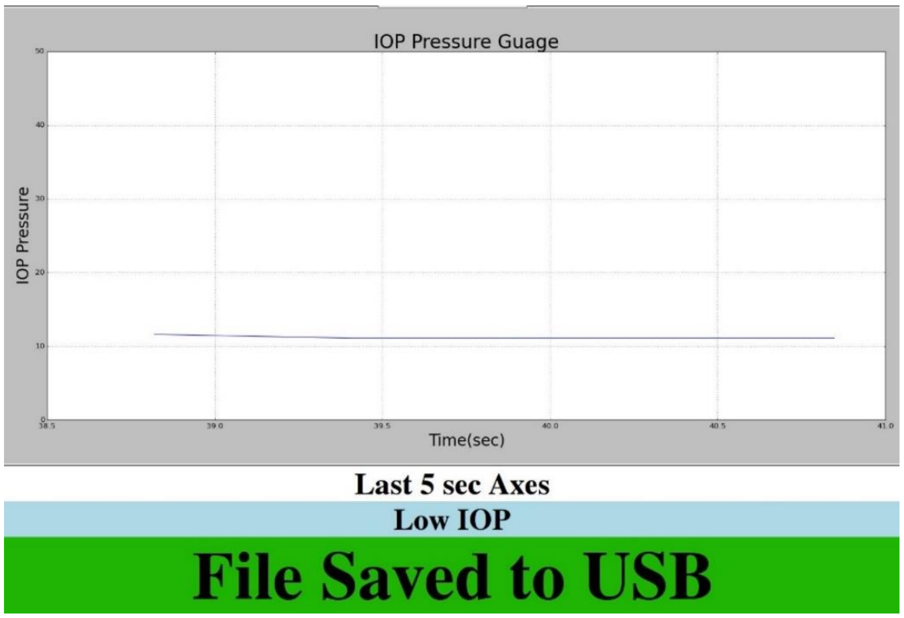


Figure 16. GUI with file saved to USB

We increased the pressure again by increasing the height of the water level in the water column. So, the IOP values are greater than 22 mmHg with a red color pop up message as “High IOP”.

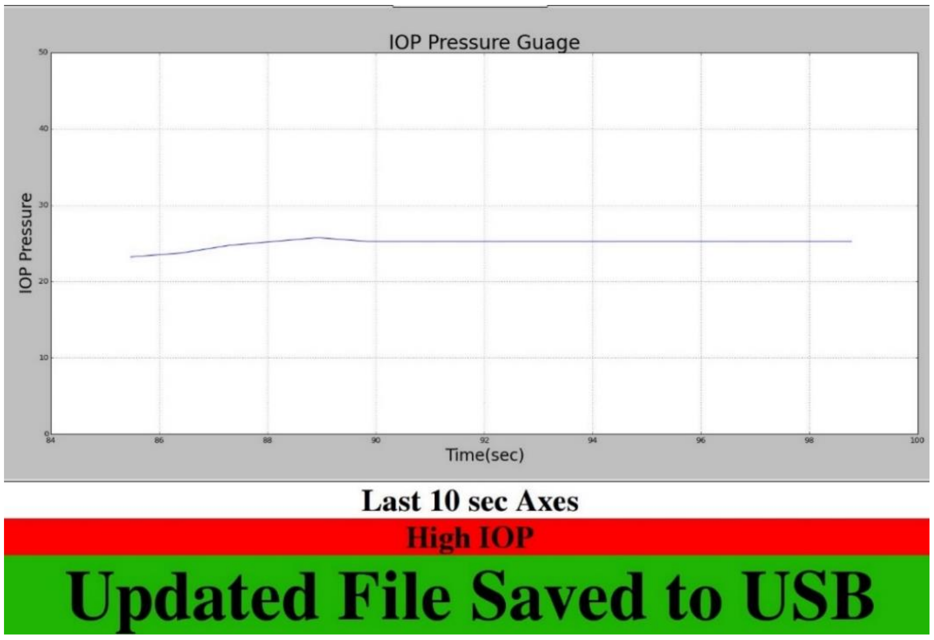


Figure 17. GUI with file updated to USB

The graphing mode of last 5 sec is doubled to 10 sec as told above. That will keep on doubling if the regular gets doubled as shown in Figure 17. Then after some time, we tried to save the data again to USB drive connected. Now there is a pop-up message like "Updated File Saved to USB". So, every time it's called from second time it will overwrite the file with the new data.

To achieve this, we are using the "*Shutil"* library to ease the process of moving files from one location to another. We are using the "*subprocess"* library to call the actual program from subprogram i.e.the main code will call based on the condition that if the user presses the green start button and used the global variable library for the different file name.

5.8 Heat Dissipation

The experimental work on the donor's eye will sometimes last around 7-8 hours. So, the device should be capable of delivering the same performance throughout the experiment. Although RPi has its own power management system. i.e. when the RPi temperature reaches to the maximum operating point then it will automatically shutdown to secure the device from damaging due to heat. We have used the heat sink, with the cooling fan to keep the RPi in normal operating temperature that allows operating RPi for long hours of operation.

5.9 Real-Time Clock Module

We have used the real-time clock module to set the RPi time with the actual time of the real world. We have used the DS3231 RTC module (shown in Figure 18), which is very affordable and reliable. The Average lifespan of this module is more than one year because it will charge itself from the Rpi power supply. Since we are saving the file name with the “Time Stamp” the real-time clock of the RPi places an important role in our project because So each time we run our experiment to measure IOP, a new file is created with the current time and date as a filename

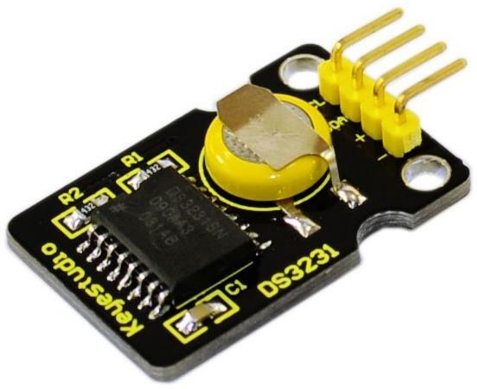


Figure 18. RTC module.

There are several alternative ways to create the file name. Initially, we did the file name as DataX.txt then the X-valve will start from ‘0’ and goes on increasing as we run our project setup. Then later we realized that sorting the data files are getting complex. So, we decided to save the file name with the date and time stamp, which is very convenient in sorting the files and when it took place. DS3231 clock module is easy to setup and the RPi loads the data and time from RTC module every time when it starts.

5.10 DC-DC Converter

We are powering up the whole system by using the 5V/2Amp, which is sufficient for the RPi and all other electronic components. We are using special operational amplifiers which require dual supply voltage with ranges in between ±3v to ±18v. Therefore, there was a need to create the dual power supply voltages from the single supply.

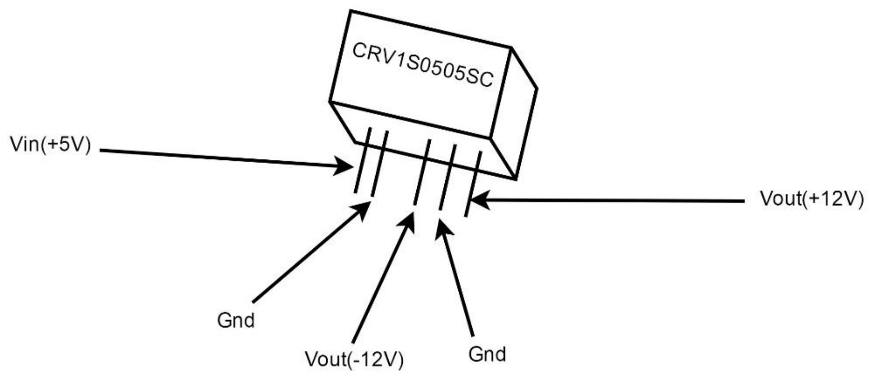


Figure 19. DC-DC converter

We could use the dual supply transformer inside our customized box to supply for RPi and the amplifiers which need the dual supply voltages. However, that would affect the sensitive signal with a lot of noise created by the transformer which is placed near to it. We decided to use a commercially available DC-DC converter which is very small in size and will not produce noticeable noises.

5.11 LCD

The 20x4 LCD assembled on the front panel of the custom enclosure, which is placed left side of the push buttons. The LCD is used to show the status of the device and will be important if there are no external VGA display available. Initially, when the system boots the LCD will show a welcome message. The first line of the LCD, displays the status of the device with measured IOP value. The second line is programmed to show the different ranges of IOP such as low, normal, high. The third line shows the USB status

(Figure 20). For example, when the user presses the push button to save the data to the USB then that will show whether the USB is connected or not.

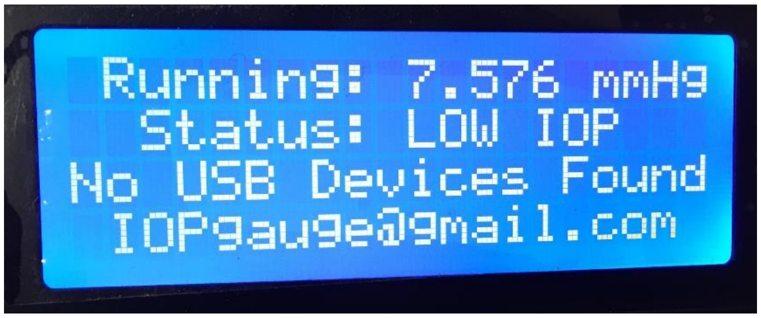

```
Running: 7.576 mmHg
 Status: LOW IOP
No USB Devices Found
IOPgauge@gmail.com
```

Figure 20. LCD

The fourth shows the real VNC account details. Due to the limited 20x4 characters in LCD, we have to limit my characters and things to show in the display.

The 20x4 LCD would help in some critical cases where some labs have no permissions to the analog devices in the labs, which will cause some distortion to the measurements.

5.12 Flow Charts

5.12.1 *System Flow chart*

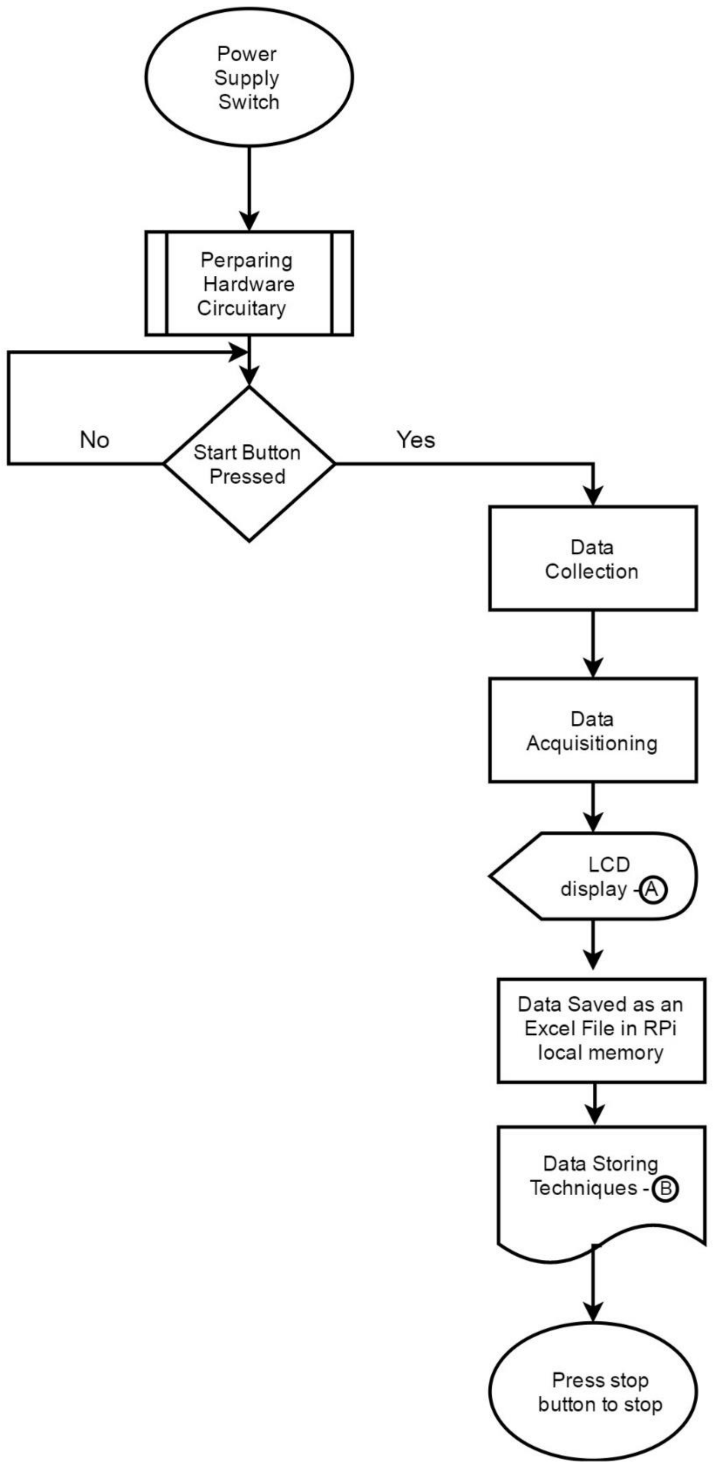

Power Supply Switch
Perparing Hardware Circuitary
No
Start Button Pressed
Yes
Data Collection
Data Acquisitioning
LCD display - A
Data Saved as an Excel File in RPi local memory
Data Storing Techniques - B
Press stop button to stop

Figure 21. Flow chart for complete system

5.12.2 LCD Flow Chart

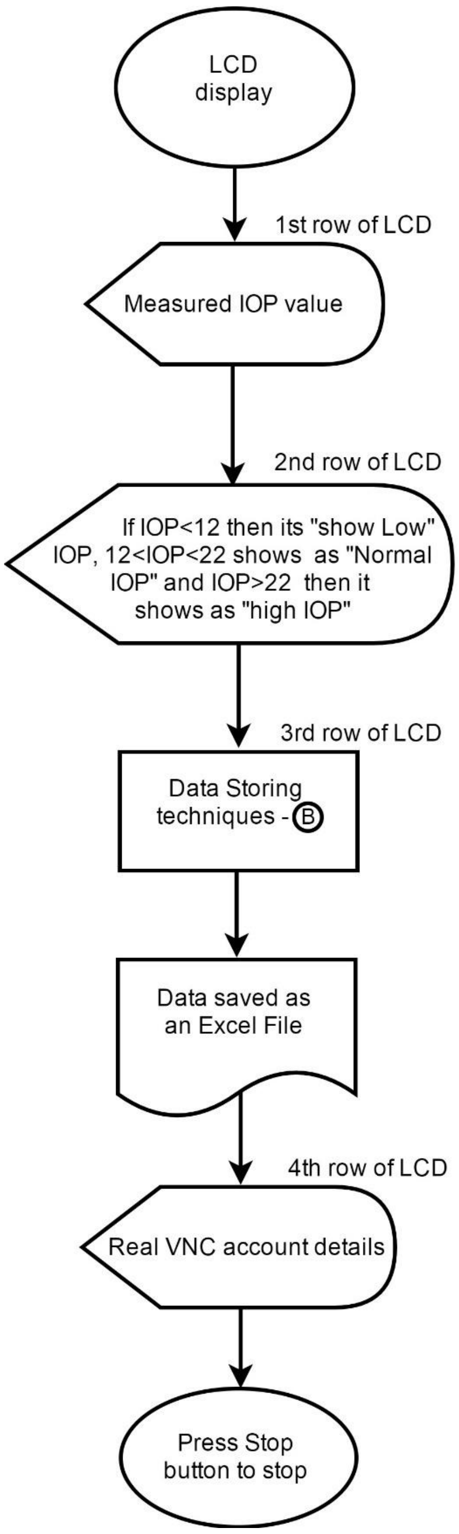

LCD display
1st row of LCD
Measured IOP value
2nd row of LCD
If IOP<12 then its "show Low" IOP, 12<IOP<22 shows as "Normal IOP" and IOP>22 then it shows as "high IOP"
3rd row of LCD
Data Storing techniques - B
Data saved as an Excel File
4th row of LCD
Real VNC account details
Press Stop button to stop

Figure 22. Flow chart for LCD

5.12.3 *Data Storing Techniques*

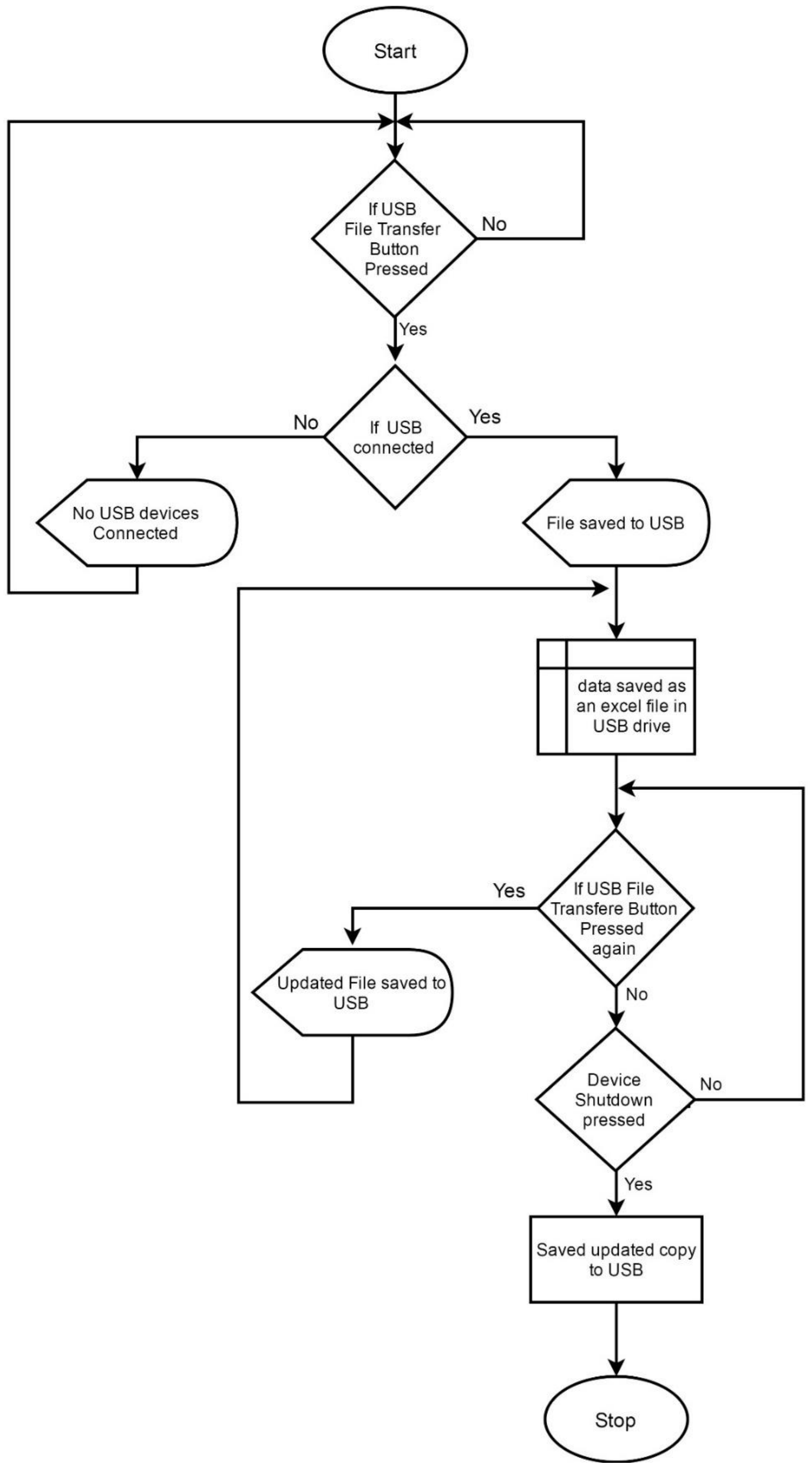

Start
If USB File Transfer Button Pressed
No
Yes
If USB connected
No
Yes
No USB devices Connected
File saved to USB
data saved as an excel file in USB drive
If USB File Transfere Button Pressed again
Yes
No
Updated File saved to USB
Device Shutdown pressed
No
Yes
Saved updated copy to USB
Stop

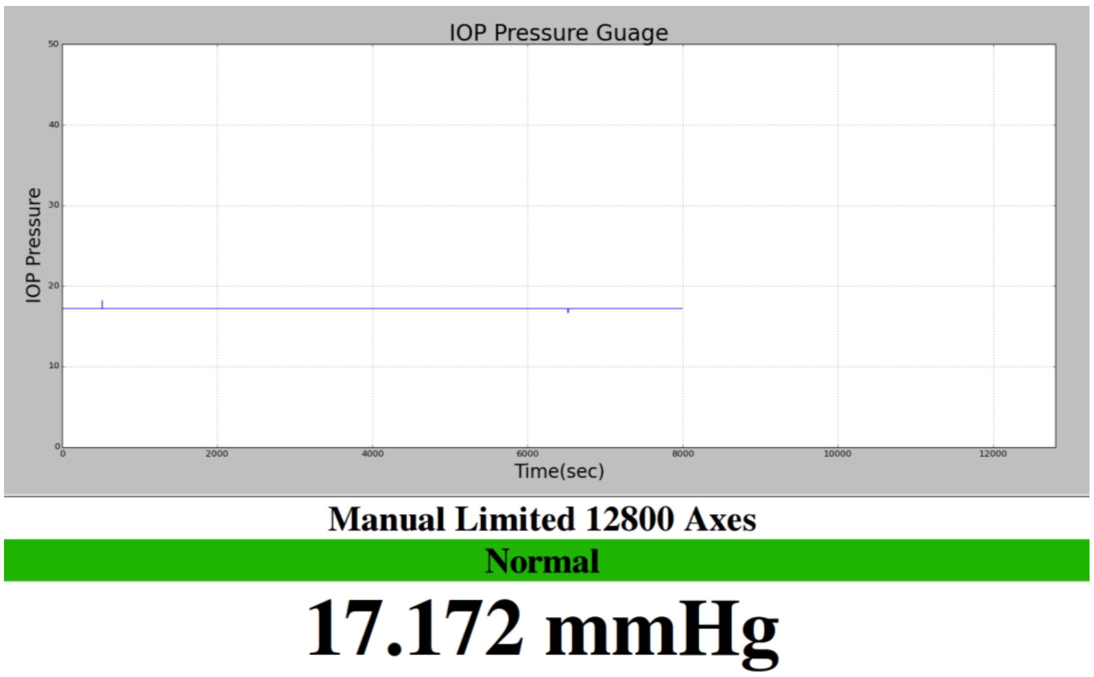
Figure 23. Flow chart for data storing technique.

CHAPTER 6

CONCLUSION AND FUTURE RESEARCH

6.1 Conclusions and Summary

We have designed a measuring device to measure IOP of a donor's eye. The device has various additional features to view on the external monitor, including a data saving option to the USB. These GUI will be in full-screen mode, so the user cannot see anything else except the graphing plots as shown in Figure 24. If there is any USB connected and when the user presses the USB data saving button, then the data will directly save in that USB drive and leave a duplicate file in the local RPi memory for future use.



Manual Limited 12800 Axes

Normal

17.172 mmHg

Figure 24. GUI with long range graph

The USB device can be connected at any time during the experiment and it needs to push the data transfer button to transfer the experiment file from RPi memory to USB drive (as shown in Figure 24).

There is also an additional feature that if the user already saved the data file to USB, if he wants to copy the file for the same experiment again the pop-up message will show "the updated file saved to USB". That will overwrite the file with the current values. There are also the different graphing modes for a better output. The auto scale option will let us see the auto scale axes which the range of current 25 seconds plotting range. The last x-axes option is to see the last X seconds of axes, x will be depending on the regular graph mode. This X will increase depending on the time length of the experiment. The additional feature is to use real VNC to view or to operate the device remotely from anywhere in the world based on the condition that RPi is connected to the network.

6.2 Limitations & Future Work

Due to the limited time for thesis work, our project has some limitations. The device is preprogrammed to do only particular tasks. But the major drawback is that when we take RPi to a new place then the device needs to setup to connect to the local network. Still, there is a need to implement the better filtering algorithms to eliminate sudden impulse's in the graph. There is also an additional feature we can implement in this device it we can use the cloud storage for the data files, so that they can access from anywhere.

We have calibrated the device for the initial setup, however, if we relocated the device, then we need to calibrate the device again. Since the pressure may differ from one place to another and with respect to altitude. In other words, it will have different pressure value on the first floor when compared to the fifth floor. Therefore, we need to use a second sensor (for ambient pressure) to auto calibrate the device. Although we didn’t do the auto calibration, in that case, there is a way that we propose to use the second sensor for auto-calibrating the device. The first sensor will be used to measure the IOP values and second sensor measure the surrounding room pressure. So, we have both values from a different sensor, and the difference between those two value can be eliminated during the experiment. Instead of making all the complex mathematical calculations, there is a way that we can auto zero the device when it starts. The first sensor value is corrected to zero according to the reference value taken from the second sensor which is in the same place but opened to room pressure.

LIST OF REFERENCES

[1] Kuno, Y., Yagi, T., and Uchikawa, Y., “*Development of a Fish-Eye VR System with Human Visual Functioning and Biological Signals*” [Electronic version] (1999). Retrieved August 2001, from http://www.cmplx.cse.nagoya-u.ac.jp.

[2] Abidin Yildirim, “Processor control for neutral density filter controller,” Master Thesis, Dept. Elect. Eng., The University of Alabama at Birmingham(UAB), AL, 2003.

[3] National Eye Institute: https://nei.nih.gov/eyedata, retrieved on January 21st 2017.

[4] M. Kaneko et al., “*Measurement and Analysis of Human Eye Excited by an Air Pulse,*” 2006 IEEE International Conference on Multisensory Fusion and Integration for Intelligent Systems, Heidelberg, 2006, pp. 353-358. doi: 10.1109/MFI.2006.265632.

[5] Helms, N. R., and Belcher, M. C. “Lighting for Energy-Efficient Luminous Environments” [Electronic version]. Chapters, 1, 2, 3, and 5. (1991). Retrieved June 2001, from The University of Kansas, Architectural Engineering, Lighting, and Electrical Systems Option Web site: http://www.saud.ku.edu/book/eye/struct.htm (This document may not be accessible).

[6] J. Nishiyama, I. Kao and M. Kaneko, “*IOP measurement using air-puff tonometry: Dynamic modeling of human eyeball with experimental results*”, *2013* 10th International Conference on Ubiquitous Robots and Ambient Intelligence *(URAI)*, Jeju, 2013, pp.134-139.doi: 10.1109/URAI. 2013.6677496.

[7] Molavi, D. W. “*Neuroscience Tutorial, Eye, and Retina*” [Electronic version]. (1997). Retrieved August 2001, from The Washington University School of Medicine.

[8] National Eye Institute: *https://nei.nih.gov/photo*, retrieved on February 11th 2017.

[9] K. C. Katuri, S. Asrani and M. K. Ramasubramanian, "*Intraocular Pressure Monitoring Sensors,*" in *IEEE Sensors Journal*, vol. 8, no. 1, pp. 12-19, Jan. 2008.doi: 10.1109/JSEN.2007. 912539.

[10] Alexander V. Luce, Eniko T. Enikov and Bradley J. Nelson, “*Design of automated digital eye palpation exam for intraocular pressure measurement*,” Complex Medical Engineering(CME), 2009, ICME International Conference on 9-11 April 2009.

[11] Quigley HA. “*Number of people with glaucoma worldwide,”* British Journal of Ophthalmology, 80(5), 389-93 May 1996.

[12] G. Chitnis, T. Maleki, B. Samuels, L. B. Cantor and B. Ziaie, "*A Minimally Invasive Implantable Wireless Pressure Sensor for Continuous IOP Monitoring*," in *IEEE* Transactions on Biomedical Engineering, vol. 60, no.1, pp. 250-256, Jan. 2013. doi: 10.1109/TBME.2012.2205248.

[13] Lee, DA, and EJ Higginbotham, “*Glaucoma and its treatment: A review,*” Am. J. Health-Syst Pharm, 62, Apr. 1, 2005.

[14] Young WC (1989), “*Roark's formulas for stress and strain,”* 6th ed. New York, *N Y,* McGraw-Hill.

[15] Pye, D and GJ Orssengo, “Determination of the true intraocular pressure and modulus of elasticity of the human cornea in vivo,” Bull. Math. Biology, 61, 551-72 1999.

[16] Liu, J, and CJ Roberts, “*Influence of corneal biomechanical properties on intraocular pressure measurement,*” J.Cataract. Refract. Surg., *3* 1, 146-55, 2005.

[17] Resnikoff S, Pascolini D, Etya’ale D et al. (2004) “*Global data on visual impairment in the year 2002*”, Bull World Health Organ 82(11): 844–851.

[18] Quigley HA, Broman AT (2006) “*The number of people with glaucoma worldwide in 2010 and 2020*”. British Journal of Ophthalmology 90(3):262–267.

[19] A. Faul, M. Turner and J. Naber, "*Implantable wireless microsystems for the measurement of intraocular pressure*," 2011 IEEE 54th International Midwest Symposium on Circuits and Systems *(MWSCAS)*, Seoul, 2011, pp. 1-4. doi: 10.1109/MWSCAS.2011.6026619 originally Adapted from Alward, W. "The Requisites in Ophthalmology: Glaucoma ". St Louis, MO. Mosby, 2000.

[20] H. A. Quigley and A. T. Broman, “*The number of people with glaucoma worldwide in 2010 and 2020*”, *Br. J. Ophthalmol.*, vol. 90, no. 3, pp. 262–267, Mar. 2006.

[21] Tan, SY, Zia, Jk. (Sep 2007), “*Albrecht von Grefe(182-1870): founder of scientific ophthalmology*”, published in Singapore Medical journal. 48(9): 797-8. PMID 17728957.

[22] Goldmann H., Schmidt T. Opthalmologica. -1957-Bd 136.-S.221-231.

[23] Claude Kaufmann; Lucas M. Bachmann; Michael A. Thiel, “*Comparison of Dynamic Contour Tonometry with Goldmann Applanation Tonometry*”, Investigative Ophthalmology & Visual Science September 2004, Vol.45, 3118-3121. doi:10.1167/iovs.04-0018.

[24] Stamper RL, “*A history of intraocular pressure and its measurement*”, Optometry and Visual Sciences. 2011 Jan;88(1): E16-28, doi: 10.1097/OPX.0b013e318205a4e7. PubMed PMID: 21150677.

[25] P. Marchetto, T. Alvarez, R. Greene and G. Thomas, "*Error measurement for a portable tonometer,*" Proceedings of the IEEE 28th Annual Northeast Bioengineering Conference *(IEEE Cat. No.02CH37342)*, Philadelphia, PA, 2002, pp. 85-86, doi: 10.1109/NEBC.2002.999477.

VENDOR LIST

[1] https://www.amazon.com/Raspberry-Pi-RASPBERRYPI3-MODB-1GB-Model-Motherboard/dp/B01CD5VC92/ref=sr_1_2?s=pc&ie=UTF8&qid=1487901348&sr=1-2&keywords=raspberry+pi+3

[2] https://www.amazon.com/SunFounder-DS3231-Precision-Raspberry-Arduino/dp/B00HF4NUSS/ref=sr_1_2?s=pc&ie=UTF8&qid=1487901414&sr=1-2&keywords=ds3231

[3] http://www.digikey.com/product-detail/en/texasinstruments/INA125P/INA125P-ND/254670

[4] http://www.digikey.com/product-detail/en/texasinstruments/ADS1286P/ADS1286P-ND/254615

[5] http://www.digikey.com/product-detail/en/analog-devices-inc inc/OP27GPZ/OP27GPZ-ND/820340

APPENDIX A

SETTING UP RASPBERRY PI

A.1 Installing Rasbian operating system for RPi 3

Download the Raspbian operating system with Jessie lite pixel to your machine (laptop or PC) which is based on Debian. Then use a 7zip extractor to extract the image file which we can download free from their websites and have been tested to unzip the image correctly.

Raspbian comes preinstalled with a lot of softwares for education, programming, and general use. It has Python, Scratch, Sonic Pi, java, Mathematica and more.

Copy that extracted file to micro secure digital(SD) card which should be more than 4Giga bytes(GB)

Then plug in the SD card, power supply, and necessary hardware components to your raspberry pi and start the raspberry pi. Which should usually take around 30 min to install the operating system. Then from the next time, it will take less than five seconds to boot up the device and RPi will boot in PIXEL mode.

A.2 Updating Raspberry Pi to the lastest software

Even if we install the latest Rasbian Jessie with the pixel, we need to update the firmware and other if any. It’s good if we update and upgrade every time before installing something.

- $sudo apt-get update
- $sudo apt-get upgrade

A.3 Installing Python 2.x

Although raspbian jessie with pixel operating system is preloaded with python. But, there are the commands to install python 2.x.

- $sudo apt-get update
- $sudo apt-get upgrade
- $sudo apt-get install python2

A.4 Installing GPIO libraries

We need to install some additional libraries if we want to access to General Purpose Input and Output (GPIO) pins from python code. This will help you to send commands to external hardware modules and to take data from outside through GPIO pins.

- $sudo apt-get update
- $sudo apt-get upgrade
- $sudo apt-get -y install python-rpi.gpio

A.5 Installing Graphical User Interface libraries

Raspberry operating system Rasbian Jessie with the pixel is not having the sophisticated graphical libraries. So, we need to install some graphical interface libraries.

- $sudo apt-get update
- $sudo apt-get upgrade
- $sudo apt-get install python-matplotlib
- $sudo apt-get install python-tk

These commands will install all the supporting graphical libraries.

A.6 Installing Real Clock Time module

We need to keep up with the time because the file names are saved with the time stamp. So, I'm using the real-time Clock (RTC) Module. We need to enable I^{2}C communication.

- $sudo raspi-config

Then, go to advanced options and select I2C and turn it to "YES" and select "OK".

We have used DS-3231 RTC which is connected to the raspberry pi. This RTC has a lithium cell which stores the time information and provides to RPI when booting up.

- $sudo apt-get update
- $sudo apt-get -y upgrade

We need to modify the following file

- $sudo nano /etc/modules

We need to add "rtc-ds1307" if it isn't there already. Then we can save the file by using CTRL+X and then enter Y to save it. Then shutdown the system by "sudo halt" and add the RTC module to RPI and start again.

We need to test whether RPI is detected the RTC module or not by using the following command.

- $sudo i2cdetect -y 1

We should see some HEX decimal values of RTC module.

To set up the module to be loaded every time when RPi starts we need to edit another file.

- $sudo nano /etc/rc.local

Then we need to add the following lines

- echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
- hwclock -s

Then we can save the file by using CTRL+X and then enter Y to save it. Then reboot the system.

We can see the date by using “date” command. If the date needs to adjust, then we can use the following commands

- sudo date -s "dd month year h:m:s"
- sudo date -s “25 mar 207 17:47:44”

Once the date is set you can write this date to RTC module by using

- $sudo hwclock -w

To read the date every when time RPi starts,

- $sudo hwclock -r

Installing Libraries for saving the file as Excel. We are saving all the experiments data in the local folder or a thumb drive if available in the Excel format. Excel format is the easy way to get the graphical view in our personal computers too. So, we need a library by which we can save the files in excel format with date and time as a file name.

- $sudo apt-get install python-setup tools
- $sudo easy_install pip
- $sudo easy_install install openpyxl

A.7 Installing LCD libraries

We are using 20x4-character LCD display which will show all the functional status of the device. This would be of great use if we don’t have an external display.

First, we need to enable I2C as explained in RTC module section. Then we need to install some of the libraries.

- $sudo apt-get install i2c-tools
- $sudo apt-get install python-smbus.

Then check whether the LCD display is connected to Rpi or not by using

- $sudo i2cdetect -y 1

Then when you downloaded the library file from *circuit basics* website, we need to change the I2C address in the code of line 19 to the address which you got by *sudo i2cdetect -y 1*. This file should be in the same folder, in which you have your main code.

A.8 Installing libraries for Real VNC

We can only install one of the remote desktop software in raspberry, either xrdp or real VNC for remote desktop connections. Real VNC is far better xrdp in many aspects.

Installing Libraries for Virtual Network Connections:

If we want to connect to RPi from anywhere then we need to install some additional libraries:

- $sudo apt-get update
- $sudo apt-get upgrade realvnc-vnc-server realvnc-vnc-viewer

If we booted Rpi in pixel mode, we need to go to Menu > Preferences > Raspberry Pi Configuration > Interfaces and check that VNC is set to Enabled. Go to Real VNC Website and create an account with an email address and password and the open Real VNC in Rpi and select Licensing from the VNC server menu and sign into your VNC account with the email address and password provided. Then you will find your Raspberry pi under the name of your team.

A.9 Setting the code in Auto-Start mode

Once everything is setup then you need to set your code to run every time RPi boots up. We need to use the following commands to setup autostart code in RPi.

- $sudo nano /etc/profile

Then add you code path at the end of that file and then press Ctrl+X,“Y”, enter.

- $sudo /home/pi/Your Folder Name/Your Filename.py

APPENDIX B

DEVICE INSTRUCTION MANUAL

B.1 Basic Instructions to turn on Device

1) Power up the device using the power Adapter given to you.
2) Initial setup is required to set the Wi-Fi.
3) Make sure you can see the welcome screen on the LCD display, which is arranged on the front side of the Customized box as well as in external monitor.
4) A welcome note will come on the LCD screen followed by the IP addresses and the VNC account details list to which raspberry is connected.
5) Then follow the Instruction which will appear on the LCD screen and GUI.
6) A pop will show on the LCD screen to push the start button to start your experiment. And we programmed RPI to auto-adjust the resolution of the monitor which are of different sizes.
7) Power on the raspberry and tell the people who want to collaborate the experiment remotely from anywhere in the world by VNC and then push the green button to start the experiment. So, they can collaborate for the experiment remotely.
8) Initially, the device GUI graph plotting starts with the 500-sec time scale and if the experiment time crosses the 50 sec then it will double the graphing time to 100 sec. This process will continue until the experiment lasts. The last x number of axes will also get doubled based on the graphing scale mode.
9) Then the LCD will show the status of the device with the IOP measuring value in the first line and in the second line it will show the mode of the graph and the third line is dedicated to showing the popups of data saving to USB thumb drive.
10) If you want to save the data into the thumb drive, then connect the USB drive to one of the four USB ports and then press the USB data saving button to transfer files from

local RPi memory to USB memory with a pop-up saying that experiment “file saved to USB”. If you press the same data transfer button again then again the pop-up will show as the “Updated file is saved to USB”.

11) If there is a connected USB before the experiment starts, then the file would be saved in USB and local memory. If there aren’t any connected and if you are trying to transfer file, then a pop-up message will come within red color “No USB connected”.

12) Once you finish the experiment make sure to press the shutdown button. So that you won’t corrupt the device or the raspberry Pi.

13) Don’t be misunderstand that when the LCD backlight is on means that the raspberry pi is working. It’s not the raspberry pi will give that power supply to its GPIO pins even when it's off if it is connected to power supply.

B.2 Device Working Status

14) Once you set up all the monitors and USB then you can push the start button to start the experiment then you can see the graph as shown in Figure 13 with the graph on the followed by the measured value below.

15) There is also have an indicator to show what kind of the graph mode is in use now and another indicator to show the status of the measured value. If the measured value goes too high, then it will convert to a red color as shown below to alter the user.

16) Implemented different color modes for different ranges of the measured value in real time.

17) If the measured value is in below normal range(0-12mmHg) of IOP then it will show as Below Normal with the background color of ‘Cyan’. Similarly, if it has in the range

of 12-22mmHg, 22-26mmHg, greater than 26mmHg, then it will show as Normal, High, very High with ‘Green’,’ Orange’ and ‘red’ background colors respectively.

B.3 Settting Up Remote Desktop Connections

18) If there is no initial Setup done, most probably you are not on the Wi-Fi network, then there is only one way to get this one by connecting an Ethernet LAN wire to the device then putting down the IP which you will see on the LCD display when the device starts in the remote desktop connection in Laptop’s and works only if you are in the same network. By Using real VNC, we can easily connect to RPi from anywhere in the world. But these devices should be on a single account and five people can access this at a time.

B.4 Saving Missed Files to USB

19) If you missed any on the experiment data files to save onto your thumb drive, you should connect the keyboard and mouse to the device. The USB slots are located on the back side of the device and then start the experiment and then press ALT + F4, which will close the graphing plot window and the Desktop Graphical User Interface (GUI) will come just like similar to Laptops.
20) Then use the mouse to go to windows explorer and open PROJECT folder and then you can find your experiment data file with the date and time file name. Connect your thumb drive and then drag and drop files to your thumb drive location.
21) There is also a standard way to copy your files, Right click then select ‘copy’ and go to your target thumb drive location. Then again right click and then select ‘paste’ to copy your files from the measuring device to your thumb drive.

B.5 Calibrating the Device if relocated to another Place

22) The device has calibrated before but if it's relocated to another place, there is a need to calibrate the device again due to the pressure change from place to place. So the initial calibration is needed to use the device. The Calibration procedure is very simple and can be done at any time without any additional skills needed. Calibration is necessary to cancel the offsets or else we will get the abnormal values.
23) It is always good to replace the sensor with the new one. If there isn’t the new sensor available, then there is a need to clean the sensor due to the particles or debris will be left over from the previous experiment. Rinse the sensor with the warm water to clean the sensor pipes. Then place the sensor where the air can circulate more to dry any water droplets left over inside the pipes.
24) Then attach the sensor to the same level of the measuring eye device setup. If there is any different value show other than the 0mmHg then rotate the knob which is on the front side of the device (refer to Figure 6) Rotate the clock wise or anti-clock wise to get the values to 0mmHg and then start your experiment.

APPENDIX C

PYTHON SKETCH

There are three seperate codes for our working project:

1) 1st code will call the second code by the push button with some welcome messages.
2) 2nd code will the actual code of the graphical user interface.
3) The 3rd code will show the shutting down messages.

1st program code:

```
'''
created by Anvesh Loka
********************
Importing libraries
General RPi Libraries
********************
'''
import subprocess
import RPi.GPIO as GPIO
import time
import os
import I2C_LCD_driver
import glob
import signal
from threading import Thread
'''
********************
Rpi GPIO libraries
Rpi Graphical user Interface Libraries
Matplotlib and tkinkter
********************
'''
import matplotlib.pyplot as plt
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg,
NavigationToolbar2TkAgg
from Tkinter import *
import signal
import matplotlib.font_manager as font_manager
from matplotlib.figure import Figure
matplotlib.use('TkAgg')

'''
```

```
********************
setting Up GPIO pins
********************
'''
GPIO.setmode(GPIO.BCM)
GPIO.setup(12, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(26,GPIO.OUT)
GPIO.setup(21,GPIO.OUT)
time.sleep(2)
a=1
'''
********************
Setting up LCD display and obtaining IP values
********************
'''
my_ip = subprocess.Popen(['ifconfig wlan0 | awk "/inet /" | cut -d":" -f 2 | cut -d" " -f1'],
stdout=subprocess.PIPE, shell=True)
my_i = subprocess.Popen(['ifconfig eth0 | awk "/inet /" | cut -d":" -f 2 | cut -d" " -f1'],
stdout=subprocess.PIPE, shell=True)
(IP,errors) = my_ip.communicate()
(I,errors) = my_i.communicate()
my_ip.stdout.close()
my_i.stdout.close()
str_pad = " " * 16
st="WiFi:"
eth="Eth :"
newip=IP[:-1]
neweth=I[:-1]
wifi = st + newip
nowifi= st + "No wifi Connectd"
ethwire = eth + neweth
noeth=eth+ "No Eth connectd"
'''
********************
Function to blink status led
********************
'''
def led():
    GPIO.output(26,True)
    time.sleep(0.05)
    GPIO.output(26,False)
    time.sleep(0.08)
'''
********************
Starting GUI parameters
********************
```

```
'''
fi, ax = plt.subplots()
lines, = ax.plot([],[])
root = Tk()
root.title("IOP Measuring Device")
root.attributes('-fullscreen', True)
left=Frame(root)
left.pack(side=TOP,anchor=CENTER,fill=BOTH,expand=1)
west=Frame(root)
west.pack(side=TOP,anchor=CENTER,fill=BOTH, expand=0)
east=Frame(root)
east.pack(side=TOP,anchor=CENTER,fill=BOTH, expand=1)
b=Label(west,font=('times',55 , 'bold'),bg='white')
c=Label(east,font=('times',55 , 'bold'), bg='white')
d=Label(east,font=('times',55 , 'bold'), bg='white')
led()
'''
********************
To show welcome note on LCD & GUI
********************
'''
mylcd = I2C_LCD_driver.lcd()
a=1
mylcd.lcd_clear()
mylcd.lcd_display_string("Welcome To" ,1)
mylcd.lcd_display_string("IOP" ,2,6)
mylcd.lcd_display_string("Measuring Device" ,3,3)
b.config(b,text="Welcome to")
b.pack(anchor=S,fill=X,expand=0)
c.config(b,text="IOP")
c.pack(anchor=S,fill=X,expand=0)
d.config(b,text="Measuring Device")
d.pack(anchor=S,fill=X,expand=0)
root.update()
led()
time.sleep(3)
'''
********************
To show IP addresses and VNC account details
********************
'''
led()
print 'le'
vn="VNC:"
vnc=vn+ " Email:-  "
email="IOPgauge@gmail.com"
```

```
pswd="Password: Anvesh66"
mylcd.lcd_clear()
mylcd.lcd_display_string("IP Addresses",1,3)
time.sleep(0.5)
if IP:
    mylcd.lcd_display_string(wifi,2)
else:
    mylcd.lcd_display_string(nowifi,2)
led()
if I:
    mylcd.lcd_display_string(ethwire,4)
else:
    mylcd.lcd_display_string(noeth,4)

mylcd.lcd_clear()
mylcd.lcd_display_string(vnc,1,0)
mylcd.lcd_display_string(email,3,0)
mylcd.lcd_display_string(pswd,4,0)
led()
b.config(b,text=vnc)
b.pack(anchor=S,fill=X,expand=0)
c.config(b,text=email)
c.pack(anchor=S,fill=X,expand=0)
d.config(b,text=pswd)
d.pack(anchor=S,fill=X,expand=0)
root.update()
time.sleep(3)

'''
********************
To show the command to start experiment in LCD
********************
'''
mylcd.lcd_clear()
mylcd.lcd_display_string("Push the ",1)
mylcd.lcd_display_string("Green Start Button",2)

mylcd.lcd_display_string("to start",3)
mylcd.lcd_display_string("your experiment",4)

while a<3:
    print("push the START button to run the program")
    '''
    ********************
    To show the command to start experiment in GUI
    ********************
    '''
    b.config(b,text="Push the")
    b.pack(anchor=S,fill=X,expand=0)
    c.config(b,highlightthickness=1,fg='green',text="Green start button")
    c.pack(anchor=N,fill=X,expand=0)
    d.config(b,text="to start your Experiment")
    d.pack(anchor=N,fill=X,expand=0)
    root.update()
    led()
    print GPIO.input(12)
    '''
    ********************
    Condition to check the buttion is pressed or not
    if yes, this code will calls the main code by subproces
    ********************
    '''
    if (GPIO.input(12)==False):
        led()
        root.destroy()
        GPIO.output(21,True)
        time.sleep(0.08)
        GPIO.output(21,False)
        print"calling main code"
        subprocess.call("python /home/pi/project/fcode.py", shell=True)
```

2nd program code

```
'''
created by Anvesh Loka
********************
Importing libraries
General RPi Libraries
********************
'''
import datetime
import time
import os
import subprocess
from subprocess import call
import glob
import signal
'''
********************
Rpi GPIO libraries
Rpi Graphical user Interface Libraries
Matplotlib and tkinkter
********************
'''
import RPi.GPIO as GPIO
import matplotlib.pyplot as plt
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg,
NavigationToolbar2TkAgg
from Tkinter import *
import matplotlib.font_manager as font_manager
from matplotlib.figure import Figure
matplotlib.use('TkAgg')
'''
********************
To create and add data Excel files
To tranfer files from RPi to USB drive
To Drive LCD Display
To obtain IP address of RPi
********************
'''
from openpyxl import Workbook
import shutil
import I2C_LCD_driver
import socket
import fcntl
import struct
'''
********************
Date and Time stamp of file names and folder directory
********************
'''
now=datetime.datetime.now().strftime("%b %d %H;%M;%S %Y")
fname=str(now)+".xlsx"
fsave='/home/pi/project/'
basedir = '/dev/disk/by-path/'
'''
********************
setting Up GPIO pins
********************
'''
GPIO.setmode(GPIO.BCM)
GPIO.setup(22, GPIO.IN, pull_up_down=GPIO.PUD_UP) #graph mode change switch
GPIO.setup(5, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(6, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(26,GPIO.OUT)
GPIO.setup(21,GPIO.OUT)
'''
********************
Declaring GPIO pins, variables and Arrays
********************
'''
wbook = Workbook()
ws = wbook.active
DEBUG = 0
CLOCK = 23
DATAIN = 18
CAP = 24
CHIPSEL = 25
flag=0
GPIO.setup(CAP, GPIO.OUT)
GPIO.setup(DATAIN, GPIO.IN)
GPIO.setup(CLOCK, GPIO.OUT)
GPIO.setup(CHIPSEL, GPIO.OUT)
potentiometer_adc = 0
previousvalue = 0       # this keeps track of the last potentiometer value
tolerance = 0
potmanual=[]         #x axis Manual Scale
volumemanual=[]       #y axis Manual Scale
volumeauto=[]      #y axes Auto Scale
potauto=[]         #X axes Auto Scale
volumelast=[]      # Y axes Last X axes
potlast=[]         # X axes Last X axes
timescale=0
countt=1
t1=time.time()
tr=1
usbtime1=1
tB=0
tA=0
pencil=50           # For Manual Scale
pen=10              # for last x axes
lastx="A"
toshow=1
xlabel="B"
xa=0
countgraph=1
GPIO.output(21,False) #for Buzzer
flag1=0;
rdisk=0;
usbtime="B";
trailer=0
```

```
truck=1;
truck1=1;
truck2=0;
suv=0;
global IOPfolder
compact=0;
earth="B"
moon="B"
vn="VNC:"
vnc=vn+ " Email:-  "
email="IOPgauge@gmail.com"
pswd="Password: Anvesh66"

def handler(signum, frame):
   raise KeyboardInterrupt      #print 'Ctrl+Z pressed, but ignored'
'''
********************
Starting LCD & GUI parameters
********************
'''
mylcd = I2C_LCD_driver.lcd()
mylcd.lcd_clear()
fi, ax = plt.subplots()
lines, = ax.plot([],[])
ax.grid(True)
ax.set_xlabel('Time(sec)',size=25)
ax.set_ylabel('IOP Pressure',size=25)
ax.set_title("IOP Pressure Guage",size=30)
root = Tk()
root.title("IOP Measuring Device")
root.attributes('-fullscreen', True)
left=Frame(root)
left.pack(anchor=S,fill=BOTH,expand=1)#axes,NE will not expand auto for more text
right=Frame(root)
right.pack(side=BOTTOM,anchor=NE,fill=BOTH, expand=0)# for digital number
west=Frame(root)
west.pack(side=BOTTOM,anchor=NE,fill=BOTH, expand=0)#rangeeee
east=Frame(root)
east.pack(side=BOTTOM,anchor=NE,fill=BOTH, expand=0)
v=Label(right,font=('times',90 , 'bold'),bg='white') # for digital number
a=Label(east,font=('times',40 , 'bold'), bg='white') #for differernt axes
b=Label(west,font=('times',40 , 'bold'),bg='white')  # color range
c = FigureCanvasTkAgg(fi, master=left)
c.show()
c.get_tk_widget().pack(side=TOP,fill=BOTH, expand=1)
```

```
'''
*******************
Function for plotting graphs
*******************
'''
def show(xdata, ydata,scale):

        lines.set_xdata(xdata)
        lines.set_ydata(ydata)
        if (scale==0):
           ax.set_xlim(auto=True)
        else:
           ax.set_xlim(1,scale)
        ax.set_ylim(0,50)
        #Need both of these in order to rescale
        ax.relim()
        ax.autoscale_view()
        fi.canvas.flush_events()
        c.show()
        if (scale==0):
           v.config(text="Auto Scaling ")
'''
*******************
Function for USB save option
*******************
'''
def usbsave(opt):
        basedir = '/dev/disk/by-path/'
        flag1=0
        IOPfolder=1
        global rdisk
        wbook.save(fsave+fname)
        for d in os.listdir(basedir):
        #Only show usb disks and not partitions
                if 'usb' in d and 'part' not in d:
                        path = os.path.join(basedir, d)
                        link = os.readlink(path)
                        print '/dev/' + os.path.basename(link)
                        print "usb dir"
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(21,GPIO.OUT)
        GPIO.output(21,False)
        for d in os.listdir('/media/pi'):
                    flag1=flag1+1
                    print d
        if flag1>0:
                path="/media/pi/"+d+"/IOP gauge"
                if not os.path.exists(path):
                        directory=os.path.dirname(path)
                        os.mkdir(path, 0755)
                        print "IOPfolder"
                IOPfolder=IOPfolder+1
                pat="/media/pi/"+d+"/IOP gauge/"
                pat=pat+fname
                fname2="/home/pi/project/"+fname
                shutil.copy2(fname2,path)
                if rdisk:
                    v.config(bg='#20B405',text="Updated File Saved to USB")
                    v.pack(anchor=NE,fill=X,expand=1)
                    mylcd.lcd_clear()
                    mylcd.lcd_display_string("Updated File to USB" ,3,1)
                else:
                    v.config(bg='#20B405',text="File Saved to USB")
                    v.pack(anchor=NE,fill=X,expand=1)
                    mylcd.lcd_clear()
                    mylcd.lcd_display_string("File Saved to USB" ,3,2)
                    usbtime= "A"
                    rdisk=1
        if flag1==0:
                    mylcd.lcd_clear()
                    usbtime= "A"
                    mylcd.lcd_display_string("                    ",3,0)
                    mylcd.lcd_display_string("No USB Devices Found",3,0)
                    v.config(bg='red',fg='yellow',text="No USB Devices Found")
                    rdisk=0
        if opt:
                    path="/media/pi/"+d+"/"
                    command='umount'+' '+path
                    os.system(command)
'''
********************
Function for shutting down the system
********************
'''
def off():
    time.sleep(0.6)
    GPIO.output(21,False)
    usbsave(1)
    #c.close()
    mylcd.lcd_clear()
    mylcd.lcd_display_string("                    ",3,0)
    mylcd.lcd_display_string("                    ",1,0)
    mylcd.lcd_display_string("Shutting Down",2,2)
    a=Label(right,font=('digital',90 , 'bold'),bg='red')
    a.config(bg='#FF0000',text="Shutting Down the Device")
    a.pack(anchor=NW,fill=X,expand=0)
    root.update
    subprocess.call("python /home/pi/project/off.py", shell=True)
    plt.close('all')
    time.sleep(3)
    wbook.save(fsave+fname)
    mylcd.lcd_display_string("Have a great day",2,2)
    time.sleep(3)
    plt.close('all')
    root.destroy()
    time.sleep(1)
    GPIO.output(26,False)
    mylcd.lcd_clear()
    os.system("sudo shutdown -h now")
'''
********************
Function for Pulsing ADC
********************
'''
def ADC(adcnum, clockpin, mosipin, misopin, cspin):
        if ((adcnum > 15) or (adcnum < 0)):
                return -1
        GPIO.output(cspin, True)
        GPIO.output(clockpin, False)  # start clock low
        GPIO.output(cspin, False)     # bring CS low
        commandout = adcnum
        commandout |= 0x18  # start bit + single-ended bit
        commandout <<= 3  # we only need to send 5 bits here
        for i in range(1):
                if (commandout & 0x80):
                        GPIO.output(mosipin, True)
                else:
                        GPIO.output(mosipin, False)
                commandout <<= 1
                GPIO.output(clockpin, True)
                GPIO.output(clockpin, False)
        adcout = 0
        for i in range(14):   # read in one empty bit, one null bit and 12 ADC bits
                GPIO.output(clockpin, True)
                GPIO.output(clockpin, False)
                adcout <<= 1
```

```
            if (GPIO.input(misopin)):
                  adcout |= 0x1
      GPIO.output(cspin, True)
      adcout >>= 1
      return adcout
'''
********************
main Loop
********************
'''
try:
      while 1:
            '''
            ********************
            Logic for maintaining input speed in plotting
            Calling plusing ADC Function
            ********************
            '''
            GPIO.output(26,True)
            t2=time.time()
            dift=t2-t1
            if DEBUG:
               print dift
            t1=t2
            timescale=timescale+dift
            if DEBUG:
               print timescale
            signal.signal(signal.SIGTSTP, handler)
            inputADC = ADC(potentiometer_adc, CLOCK, CAP, DATAIN, CHIPSEL)
            setting = abs(inputADC - previousvalue)
            if DEBUG:
                  print "inputADC:", inputADC
                  print "setting:", setting
                  print "previousvLUE", previousvalue
            if DEBUG:
                  print "inputADC_changed", inputADC
            IOPmeasured = inputADC /40.966        # convert 12bitadc0 (0-4096) trim pot
read into 0-50 IOP mmHg
            IOPmeasured = round(IOPmeasured)        # round out decimal value
            IOPmeasured = int(IOPmeasured)/1.98        # cast volume as integer
            print 'input = {input}%' .format(input = IOPmeasured)

            if DEBUG:
                  print "IOPmeasured", IOPmeasured
                  print "tri_pot_changed", IOPmeasured
            previousvalue = inputADC
```

```
'''
********************
Data Sorting for different modes of graphs
To show different Graphical plotting axes
********************
'''
if timescale<pencil:
        volumemanual.append(IOPmeasured)
        potmanual.append(timescale)
volumeauto.append(IOPmeasured)
potauto.append(timescale)
if (IOPmeasured>22):
    GPIO.output(21,False)
    time.sleep(0.02)
    GPIO.output(21,True)
volumelast.append(IOPmeasured)
potlast.append(timescale)
if timescale<=pencil:
             phone="A"        #show(potl,volumelist1,pencil)
             print "hi"
elif (timescale>pencil):
        pencil=pencil*2;
        pen=pen+10;
        lastx="B"
        xa=xa+1
        countgraph=0
if phone=="A":
     countgraph=countgraph+1
     if countgraph>pencil/10+xa:
            del volumelast[0]
            del potlast[0]
if timescale > 20:
         del volumeauto[0]
         del potauto[0]
if (GPIO.input(22)==False):
    GPIO.output(21,True)
    time.sleep(0.01)
    if flag==1:
        flag=2
    elif flag==2:
        flag=0
    elif flag==0:
        flag=1
```

```
if (IOPmeasured>22):
    GPIO.output(21,False)
    time.sleep(0.02)
    GPIO.output(21,True)

if (GPIO.input(6)==False):
    GPIO.output(21,True)
    time.sleep(0.01)
    usbsave(0)
    earth="A"    # activate to plot again the axes on LCD
    usbtime="A"
    ws=wbook.active

if flag==1:
    print("Pin 11 is LOW = Auto Scale axes ")
    a.config(a,text="Auto Scale Axes")
    xaxes="A"
    a.pack(anchor=NE,fill=X,expand=1)
    GPIO.output(21,False)
    show(potauto,volumeauto,0)

elif flag==0:
     print("Pin 11 is HIGH=manual limited axes")
     xaxes="M"
     a.config(a,text="Manual Limited "+str(pencil)+ " sec Axes")
     a.pack(anchor=NE,fill=X,expand=1)
     GPIO.output(21,False)
     if timescale<=pencil:
                show(potmanual,volumemanual,pencil)
     elif (timescale>pencil):
                 pencil=pencil*2;
                 pen=pen+1;
                 show(potmanual,volumelistmanual,pencil)

elif flag==2:
    print(" Last 100(x) Axes")
    xaxes="L"
    a.config(a,text="Last "+str(pencil/10) +" sec Axes")
    a.pack(anchor=NE,fill=X,expand=1)
    GPIO.output(21,False)
    show(potlast,volumelast,0)

GPIO.output(21,False)
IOPmeasured=round(IOPmeasured,3);
if (IOPmeasured>22):
   GPIO.output(21,False)
   time.sleep(0.02)
   GPIO.output(21,True)
'''
********************
Data appending to excel for each second
********************
'''
if ((timescale//1)!=tr):
   c1="A"+str(tr)
   c2="B"+str(tr)
   ws[c1] = tr
   ws[c2]= IOPmeasured
   print tr
   tr+=1
countt=countt+1
if usbtime== "A":
   usbtime1=usbtime1+1

if (IOPmeasured>22):
   GPIO.output(21,False)
   time.sleep(0.08)
   GPIO.output(21,True)
if usbtime1>4:
   mylcd.lcd_display_string("                    ",3,0)
   usbtime1=1
   usbtime="B"
'''
********************
LCD logic for showing Pop up messages
********************
'''
mylcd.lcd_display_string("Running: ",1,1)
mylcd.lcd_display_string(str(IOPmeasured),1,10)
mylcd.lcd_display_string("mmHg",1,16)
'''
********************
To show the range of IOP on screen
********************
'''
if (IOPmeasured>=0 and IOPmeasured<12):
   b.config(bg='#ADD8E6',text="Low IOP")
   mylcd.lcd_display_string("Status: LOW IOP      ",2,2)
   b.pack(anchor=NE,fill=X,expand=1)
if (IOPmeasured>=12 and IOPmeasured<=22):
   b.config(bg='#20B405',text="Normal IOP")
```

```
    b.pack(anchor=NE,fill=X,expand=1)
    mylcd.lcd_display_string("Status: NORMAL IOP",2,2)
if (IOPmeasured>22):
    GPIO.output(21,True)
    time.sleep(0.001)
    b.config(bg='#FF0000',text="High IOP")
    b.pack(anchor=NE,fill=X,expand=1)
    mylcd.lcd_display_string("Status: HIGH IOP     ",2,2)
'''
********************
Logic to show VNC account Details on LCD
********************
'''
if truck<5:
    mylcd.lcd_display_string(vnc,4,1)

    truck1=truck1+1
    truck=truck+1
if truck1>=5:
   trailer=1
   truck=7
   truck2=0
   mylcd.lcd_display_string("                    ",4,0)
if trailer==1:
   truck1=1
   truck2=truck2+1
   mylcd.lcd_display_string(email,4,1)

if truck2>5:
   suv=1
   truck2=0
   trailer=0
   mylcd.lcd_display_string("                    ",4,0)
if suv==1:
        compact=compact+1
        mylcd.lcd_display_string(pswd,4,1)
if compact>=5:
        suv=6
        truck=0
        compact=1
        midsize=1
        mylcd.lcd_display_string("                    ",4,0)
GPIO.output(21,False)
GPIO.output(26,False)
if (GPIO.input(5)==False):
        GPIO.output(21,True)
```

```
                    time.sleep(0.01)
                    v.config(text="Shutting down the Device")
                    v.pack(anchor=NE,fill=X,expand=0)
                    root.update()
                    mylcd.lcd_clear()
                    mylcd.lcd_display_string("Preparing To ",2,2)
                    mylcd.lcd_display_string("Shut Down ",3,3)
                    off()
               else:
                  v.config(bg='white',fg='black',text=str(IOPmeasured)+" "+"mmHg")
                  v.pack(anchor=NE,fill=X,expand=0)
 except KeyboardInterrupt:
   # exits when you press CTRL+C
    v.config(text="offff")
    v.config(text="Shutting down the Device")
    v.pack(anchor=NE,fill=X,expand=1)
    print "\nPlotting Graph is closed Bye\n"
    plt.close('all')
    mylcd.lcd_clear()
    GPIO.cleanup()
    wbook.save(fsave+fname)
    usbsave(0)
#Excptional Error
except:
   print "\nOther error or exception occurred!\n good bye\n"
   plt.close('all')
   mylcd.lcd_clear()
   wbook.save(fsave+fname)
   usbsave(0)
   GPIO.cleanup()
   print("good bye")
finally:
    print "***************** ******"
    GPIO.cleanup()
    mylcd.lcd_clear()
    plt.close('all')
    GPIO.cleanup()
    wbook.save(fsave+fname)
    usbsave(0)
    print("bye")
```

3rd program code:

```
'''
created by Anvesh Loka
********************
Importing libraries
General RPi Libraries
********************
'''
import subprocess
import RPi.GPIO as GPIO
import time
import os
import I2C_LCD_driver
import glob
import signal
from threading import Thread
'''
********************
Rpi GPIO libraries
Rpi Graphical user Interface Libraries
Matplotlib and tkinkter
********************
'''
import matplotlib.pyplot as plt
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg,
NavigationToolbar2TkAgg
from Tkinter import *
import signal
import matplotlib.font_manager as font_manager
from matplotlib.figure import Figure
matplotlib.use('TkAgg')
'''
********************
Starting GUI parameters
********************
'''
fi, ax = plt.subplots()
lines, = ax.plot([],[])
root = Tk()
root.title("IOP Measuring Device")
root.attributes('-fullscreen', True)
left=Frame(root)
left.pack(side=TOP,anchor=CENTER,fill=BOTH,expand=1)
```

```
west=Frame(root)
west.pack(side=TOP,anchor=CENTER,fill=BOTH, expand=0)
east=Frame(root)
east.pack(side=TOP,anchor=CENTER,fill=BOTH, expand=1)
b=Label(west,font=('times',55 , 'bold'),bg='white')
c=Label(east,font=('times',55 , 'bold'), bg='white')
d=Label(east,font=('times',55 , 'bold'), bg='white')
b.config(b,fg='red',text="Shutting down ")
b.pack(anchor=S,fill=X,expand=0)
c.config(b,fg='red',text="the device")
c.pack(anchor=S,fill=X,expand=0)
root.update()
time.sleep(3)
b.config(b,fg='black',text="Have a great day")
b.pack(anchor=S,fill=X,expand=0)
c.config(b,text=" ")
c.pack(anchor=S,fill=X,expand=0)
root.update()
time.sleep(3)
print "goof"
root.destroy()
```