# Drone\_DOF

**By**

**Jay Jadav**

**Arman Velani**

**Gursehaj Harika**

**Date: 25th April, 2019**

# Declaration of Joint Authorship

We, Gursehaj Harika, Jay Jadav and Arman Velani confirm that this breakdown of authorship represents our contribution to the work submitted for assessment. The contribution of work by all other authors, individual and separate to the work group, in any form (ideas, equation, figures, texts, tables, programs), are properly acknowledged at the point of use. A list of the references used is included. Arman Velani, designed the PCB, casing. Gursehaj Harika, developed the android application, general UI. Jay Jadav, created the database and its communication with the sensors.

Authors Signed on

1 …………………………………. ……………………………

2 …………………………………. ………………………………

3 …………………………………. ………………………………

# Proposal

2019-01-17

***Proposal for the development of IoT for SensorsEffectors***

Prepared by Jay Jadav, Arman Velani, and Gursehaj Harika  
*Computer Engineering Technology Students*<https://github.com/GursehajHarika/Drone_DOF>

**Executive Summary**

As students in the Computer Engineering Technology program, we will be integrating the knowledge and skills we have learned from our program into this Internet of Things themed capstone project. This proposal requests the approval to build the hardware portion that will connect to a database as well as to a mobile device application. The internet connected hardware will include a custom PCB with the following sensors and actuators,

* MAG3110 3 axis magnetometer
* MS5611 barometric pressure sensor
* MMA8451 3 axis accelerometer

The database will store user's information, height from the barometric pressure sensor and GPS location with combination of magnetometer and accelerometer. The mobile device functionality will include measuring the Height/elevation and the GPS location of the Device using multiple sensors which are connected to Raspberry Pi using a PCB. The measured data is then sent to the database for storage purposes. And will be further detailed in the mobile application proposal. I will be collaborating with the Prototype lab and Prof's for power management. The hardware will be completed in CENG 317 Hardware Production Techniques independently and the application will be completed in CENG 319 Software Project. These will be integrated together in the subsequent term in CENG 355 Computer Systems Project as a member of a 2 or 3 student group.

**Background**

The project that we are working on uses the height of the device attached to it to measure the altitude and record it to the database. This data is then used by user using an android application to modify their flight course and gives a sense of security to the user. The mobile devices such as a drone tends to get lost easily. The moderately priced product usually have a GPS built in to find it still they face some connectivity issues due to different factors such as trees or buildings . The built in GPS tracker in our project does not uses satellite to get GPS signals thus gives the user full access to its position and elevation.

A barometric sensor senses the height of device attached to it and can even tell what the altitude of the device is. The barometric pressure sensor is paired with a GPS Receiver (it can be made using Accelerometer and Magnetometer). It will give the position and height of the device the product is attached to.

Existing products on the market include Maboshi. (2018, November 07). Arduino GPS Drone RC Boat. Retrieved from <https://www.hackster.io/maboshi/arduino-gps-drone-rc-boat-45d6f4> . I have searched for prior art via Humber’s IEEE subscription selecting Barometric sensor project and have found and read “Sung-Hyun and Han-Bai, "A study on the fabrication and electrical characteristics of barometric sensors for USN” which provides insight into similar efforts.

In the Computer Engineering Technology program we have learned about the following topics from the respective relevant courses:

* Java Docs from CENG 212 Programming Techniques In Java,
* Construction of circuits from CENG 215 Digital And Interfacing Systems,
* Rapid application development and Gantt charts from CENG 216 Intro to Software Engineering,
* Micro computing from CENG 252 Embedded Systems,
* SQL from CENG 254 Database With Java,
* Web access of databases from CENG 256 Internet Scripting; and,
* Wireless protocols such as 802.11 from TECH152 Telecom Networks.

This knowledge and skill set will enable me to build the subsystems and integrate them together as my capstone project.

**Methodology**

This proposal is assigned in the first week of class and is due at the beginning of class in the second week of the fall semester. My coursework will focus on the first two of the 3 phases of this project:  
 Phase 1 Hardware build.  
 Phase 2 System integration.  
 Phase 3 Demonstration to future employers.

*Phase 1 Hardware build*

The hardware build is completed in the fall term. It fits within the CENG Project maximum dimensions of 12 13/16" x 6" x 2 7/8" (32.5cm x 15.25cm x 7.25cm) which represents the space below the tray in the parts kit. The highest AC voltage that will be used is 16Vrms from a wall adaptor from which +/- 15V or as high as 45 VDC can be obtained. Maximum power consumption will be 20 Watts.

*Phase 2 System integration*

The system integration will be completed in the fall term.

*Phase 3 Demonstration to future employers*

This project will showcase the knowledge and skills that I have learned to potential employers.

The brief description below provides rough effort and non-labor estimates respectively for each phase. A Gantt chart will be added by week 3 to provide more project schedule details and a more complete budget will be added by week 4. It is important to start tasks as soon as possible to be able to meet deadlines.

**Concluding remarks**

This proposal presents a plan for providing an IoT solution for Drone\_DOF. This is an opportunity to integrate the knowledge and skills developed in our program to create a collaborative IoT capstone project demonstrating our ability to learn how to support projects such as the initiatives described. I request approval of this project.

**Citation:**

1. Maboshi. (2018, November 07). Arduino GPS Drone RC Boat. Retrieved from <https://www.hackster.io/maboshi/arduino-gps-drone-rc-boat-45d6f4>
2. Autonomous High Altitude Glider. (n.d.). Retrieved from <https://create.arduino.cc/projecthub/53982/autonomous-high-altitude-glider-055aa3?ref=tag&ref_id=drones&offset=9>
3. Vignesh. (2018, December 17). Intelligence Monitoring Drone System. Retrieved from <https://www.hackster.io/enigma-plasma-8/intelligence-monitoring-drone-system-727470>
4. Velani, A. (n.d.). November 27th, 2018 (Week 13). Retrieved from <https://armanvelani.github.io/3-AxisAccelerometer/>

# Abstract

The Drone\_DOF is a drone attachment unit comprising of various sensors that allows user to know their precise location as well as the change in the pressure that it faces with the changes in the altitude. By integrating the drone \_DOF we are able to address the problem of locating your mobile drone with minimal to less error in its position with the help of magnetometer sensor(MAG3110) and accelerometer sensor(MMA8451); so that the user is always updated with its accurate position. Moreover the barometric pressure sensor(MS5611) help user to know the elevation as well as the safety height above which flying of the drone would be dangerous and out of the connectivity zone from the android as well as the firebase database. The device also includes a 5000 mAh additional power source (battery) that powers itself and avoids having any loose wiring to the whole unit. The device also has the option to supply power directly to the raspberry pi. As for now, we are still working on integrating one of the sensor which we are facing problem to integrate. If user is interested to use the Drone\_DOF attachment for commercial purpose he/ she should contact with the sales department of the company. If user is looking to get the Drone\_DOF attachment manufactured on an industrial scale for resale or other purposes he/she should contact CEO of the company. In order to use it as an educational tool please contact the Director of the company, Drone DOF team will be grateful to support any such cause.

# Contents

Contents

[Drone\_DOF 1](#_Toc7107954)

[Declaration of Joint Authorship 2](#_Toc7107955)

[Proposal 5](#_Toc7107956)

[Abstract 11](#_Toc7107957)

[Contents 13](#_Toc7107958)

[ List of Illustrations or diagrams: 15](#_Toc7107959)

[Introduction 17](#_Toc7107960)

[Project Description 19](#_Toc7107961)

[Requirements 19](#_Toc7107962)

[Hardware 19](#_Toc7107963)

[Software 20](#_Toc7107964)

[Database 20](#_Toc7107965)

[Build Instructions 21](#_Toc7107966)

[Introduction 21](#_Toc7107967)

[System Diagram 22](#_Toc7107968)

[Progress 23](#_Toc7107969)

[Time Commitment 23](#_Toc7107970)

[Budget 23](#_Toc7107971)

[Mechanical Assembly 23](#_Toc7107972)

[**Soldering** 23](#_Toc7107973)

[Steps for Soldering 23](#_Toc7107974)

[Set up 26](#_Toc7107975)

[Raspberry pi setup 26](#_Toc7107976)

[Make it wireless 28](#_Toc7107977)

[PCB 31](#_Toc7107978)

[Coral Case 31](#_Toc7107979)

[Unit-Testing 31](#_Toc7107980)

[Production-Testing 35](#_Toc7107981)

[SENSOR 1: Barometric sensor (MS5611) 36](#_Toc7107982)

[Introduction 36](#_Toc7107983)

[Parts, Tools and Budget 37](#_Toc7107984)

[Mechanical Assembly 39](#_Toc7107985)

[SENSOR 2: 3-Axis Accelerometer (MMA8451) 42](#_Toc7107986)

[Introduction 42](#_Toc7107987)

[Initial Preparation 42](#_Toc7107988)

[Parts, Tools and Budget 42](#_Toc7107989)

[Mechanical Assembly 43](#_Toc7107990)

[SENSOR 3: 3 axis Magnetometer (MAG3110) 47](#_Toc7107991)

[Introduction 47](#_Toc7107992)

[Functionality 48](#_Toc7107993)

[Parts, Tools and Budget 48](#_Toc7107994)

[Mechanical Assembly 50](#_Toc7107995)

[Conclusion 57](#_Toc7107996)

[Recommendation 58](#_Toc7107997)

[Bibliography 59](#_Toc7107998)

[Glossary of terms: 61](#_Toc7107999)

[Appendices 64](#_Toc7108000)

# List of Illustrations or diagrams:

[Figure 1 - System Diagram 22](#_Toc7109087)

[Figure 2 - Soldering1 24](#_Toc7109088)

[Figure 3 - Soldering2 24](#_Toc7109089)

[Figure 4 - Soldering3 25](#_Toc7109090)

[Figure 5- Soldering 4 25](#_Toc7109091)

[Figure 6- Rpi Load Page 27](#_Toc7109092)

[Figure 7- Fresh Screen after setup 28](#_Toc7109093)

[Figure 8- Wi-Fi Status 29](#_Toc7109094)

[Figure 9 - Wi-Fi Properties 30](#_Toc7109095)

[Figure 10 -Remote Desktop Connection 31](#_Toc7109096)

[Figure 11- MS5611 37](#_Toc7109097)

[Figure 12- Soldering Machine 38](#_Toc7109098)

[Figure 13- Raspberry Screen 40](#_Toc7109099)

[Figure 14- Readings of MS5611 41](#_Toc7109100)

[Figure 15-BreadBoard 44](#_Toc7109101)

[Figure 16- Readings of MMA8451 45](#_Toc7109102)

[Figure 17-MAG3110 47](#_Toc7109103)

[Figure 18-Functionality 48](#_Toc7109104)

[Figure 19-RPI 3B+ 49](#_Toc7109105)

[Figure 20- MAG3110(2) 50](#_Toc7109106)

[Figure 21 Pinout for RPI 51](#_Toc7109107)

[Figure 22-Readings of MAG3110 52](#_Toc7109108)

[Figure 23 Label Dispenser image 1 54](#_Toc7109109)

[Figure 24 Label Dispenser Working Model 55](#_Toc7109110)

[Figure 25 Graphical Explanation 56](#_Toc7109111)

# Introduction

The Drone\_DOF is a drone attachment unit comprising of various sensors that allows user to determine their precise location as well as the change in the pressure that it faces with the changes in the altitude. Our effort is to provide a normal but an effective solution to the drone industry. The problem that we are addressing is that the products that move or fly have a high risk of getting lost, crashing or even getting destroyed when being used or being controlled by amateurs. If lost they are hard to find. So if we are able to address this problem and possibly tackle it, we can implement the same process for numerous other industry such as marine exploration, automotive industry, aviation industry, also for defense and military purposes. Over our course of time in integrating the project, we came across numerous different ideas that we wanted to add to our project to make it one of its kind. Some of them being: ( a) adding an alert system that would notify the user on his application once there is a drastic change of altitude or position. (b) Adding a LCD screen to show the data on the raspberry pie. (c) Research and test a shock-proof casing for the sensor to make sure it does not get damaged. We decided to add all this possible feature at the starting of the project, but because of the time constraint of the course we were not able to catch up with the shockproof case, LCD screen and the alert system. The different approaches used by us were to make the whole unit light in weight, easily compatible, user-friendly and also make it clear as well as easily accessible for the development and research team.

# Project Description

Firstly, the Hardware of the project was integrated. All the sensors were individually tested first and then a common PCB was designed for all the three sensors. All the sensors were soldered to the PCB and connected to the Raspberry Pie. The android application was created from scratch using android studio. Login page, data-log page, account page, live-reading page and contact page were created separately by each individual of the team. Lastly, the Database was created using firewall from google to store the live readings from the sensor and then to display it on the application. At the end everything was integrated together and this is how DRONE\_DOF team successfully completed the capstone project.

## Requirements

### Hardware

Designing the printed circuit board for all 3 sensors i.e. MAG3110 3-Axis magnetometer, MMA8451 3 Axis accelerometer and MS5611 Barometric pressure sensor and a 10,000 MAH power source to power the Raspberry Pi3 and test the casing design that was printed using acrylic sheet with a3-D printer for durability, **Arman *Velani*** *will be responsible for this requirement.* Forsoldering all the three sensor using a Soldering Iron provided by Humber College to the PCB along with the battery cells being used in the 10,000 MAH power source as well as testing the PCB designed on fritzing application for short circuiting ***Jay Jadav*** *will be responsible for this requirement.* Designing the casing and assembled the PCB and casing together along with testing everything as a whole is ***Gursehaj Harika’s*** responsibility.

### Software

For software, android studio version 3.3 will be used to make the mobile application and its J-unit testing and saving user selections using shared preference is on ***Arman Velani***’s shoulders. Android application’s portrait as well as landscape design and its real-time permissions is what ***Jay Jadav*** is responsible for. Application general UI and its logical error corrections, break points to check functions functionality and user credentials with offline authentication (Combining database and android application) as well as work on publishing the application to the Google play store is on ***Gursehaj Harika.***

### Database

Firebase database design and making sure the CRUD (Create, Read, Update, and Delete) will be ***Arman velani***’s responsibility. Reading values from the sensor and storing it in the database as well as setting up database permission for vulnerabilities and security is what ***Gursehaj Harika*** is responsible for. Reading values from the database to the application as well as the time–stamp of when the reading was stored and save it in the application along with the user’s activity and login time for administrative purposes.

## Build Instructions

### Introduction

The Drone\_DOF is a drone attachment unit comprising of various sensors that allows user to determine their precise location as well as the change in the pressure that it faces with the changes in the altitude. Our effort is to provide a normal but an effective solution to the drone industry. The problem that we are addressing is that the products that move or fly have a high risk of getting lost, crashing or even getting destroyed when being used or being controlled by amateurs. If lost they are hard to find. So if we are able to address this problem and possibly tackle it, we can implement the same process for numerous other industry such as marine exploration, automotive industry, aviation industry, also for defense and military purposes. Over our course of time in integrating the project, we came across numerous different ideas that we wanted to add to our project to make it one of its kind. Some of them being: (a) adding an alert system that would notify the user on his application once there is a drastic change of altitude or position. (b) Adding a LCD screen to show the data on the raspberry pie. (c) Research and test a shock-proof casing for the sensor to make sure it does not get damaged. We decided to add all this possible feature at the starting of the project, but because of the time constraint of the course we were not able to catch up with the shockproof case, LCD screen and the alert system. The different approaches used by us were to make the whole unit light in weight, easily compatible, user-friendly and also make it clear as well as easily accessible for the development and research team.

### System Diagram

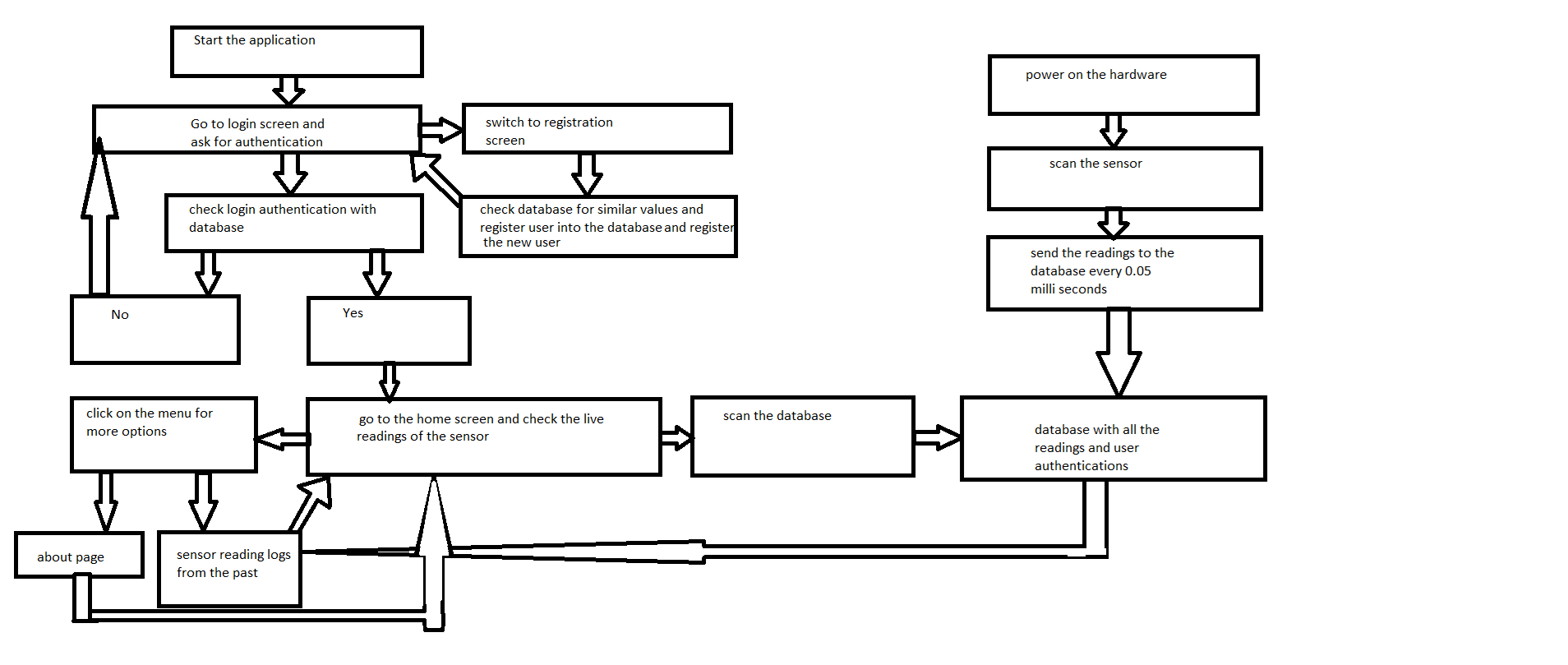
[](https://user-images.githubusercontent.com/43188450/48367169-deea5600-e67d-11e8-9f60-53604ff9a58f.png)

Figure 1 - System Diagram

### Progress

Till now we have worked on integrating all the sensors on the same PCB as well as making an acrylic case for the whole drone attachment which includes PCB with sensor on it as well as portable power source.

### Time Commitment

This schedule uses a weekly breakdown that follows the CENG 355 class schedule for the winter semester of 2019. This project could be completed in a more compact time frame if the builder so chooses. The schedule is helpful in outlining the overall flow and the order in which each milestone for the project is completed. Originally the project was completed over a 15 week semester, however it could more reasonably be completed in 1-2 week(s) dependent on how many of the materials the builder already possesses, access to the facilities necessary in producing the PCB, and shipping times.

### Budget

The individual budget is under Parts, Tools and budget of each of the sensors.

### Mechanical Assembly

## **Soldering**

### Steps for Soldering

1. Heat up your iron (600-700 degrees F or 315-370 degrees C)

[](https://user-images.githubusercontent.com/42980862/49781457-cb63f680-fce0-11e8-82eb-613669997e40.PNG)

Figure 2 - Soldering1

1. Make sure connection are mechanically stable using helping hands to keep parts steady.

[](https://user-images.githubusercontent.com/42980862/49781552-2b5a9d00-fce1-11e8-82dd-deca47e9e57e.PNG)

Figure 3 - Soldering2

1. Clean iron that builds oxide layer, which inhibits heat transfer and solder adhesion using sponge wire.

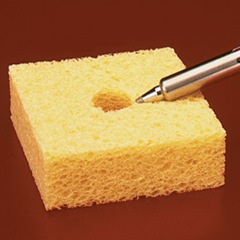
[](https://user-images.githubusercontent.com/42980862/49781575-43cab780-fce1-11e8-95ad-500c5b4705eb.png)

Figure 4 - Soldering3

1. Apply heat and solder (soldering time sometimes varies)

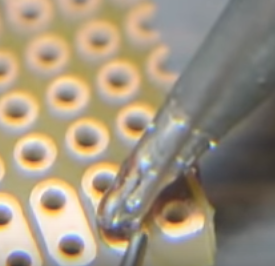
[](https://user-images.githubusercontent.com/42980862/49781732-dd926480-fce1-11e8-9c6f-daf2468aa2b8.PNG)

Figure 5- Soldering 4

1. Inspect the join ( Smooth and shiny surface can be observed using a microscope)

## Set up

### Raspberry pi setup

#### a. Setup SD card

* **NOOBS** is the recommended as well as easy to install software for initializing Raspberry Pi.
* Insert SD card underneath Rpi's micro SD card slot

#### b. Connections

* HDMI cable, Ethernet cable, Keyboard, mouse.

**Note** - Switch computer display to HDMI from VGA to initialize raspberry PI - Must see red led on rpi and blinking green led. - During the setup, system will ask user to set username, password

#### c. Setting up files for internet connection

* Issue the following command: sudo /etc/network/interfaces to make changes in wpa\_supplicant.conf which looks like:

network= {

ssid="myWi-Fi@Humber"

key\_mgmt=WPA-EAP

auth\_alg=OPEN

eap=PEAP

identity="Username or N-Number"

password="Password"

phase1="peaplabel=0"

phase2="auth=MSCHAPV2"

priority=999

proactive\_key\_caching=1

}

**Make sure to reboot!**

* In order to run program from Putty, turn on SSH options from Interface option from Rpi
* For Sensor readings, enable GPIO feature

**Startup Screen looks like:**

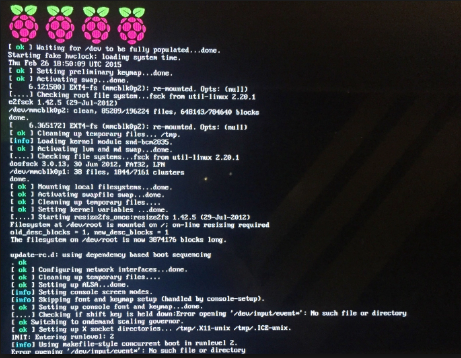
[](https://user-images.githubusercontent.com/42980862/49777295-f1cc6680-fccd-11e8-8aad-ea40f830cb08.PNG)

Figure 6- Rpi Load Page

*Fresh Screen after setup:*

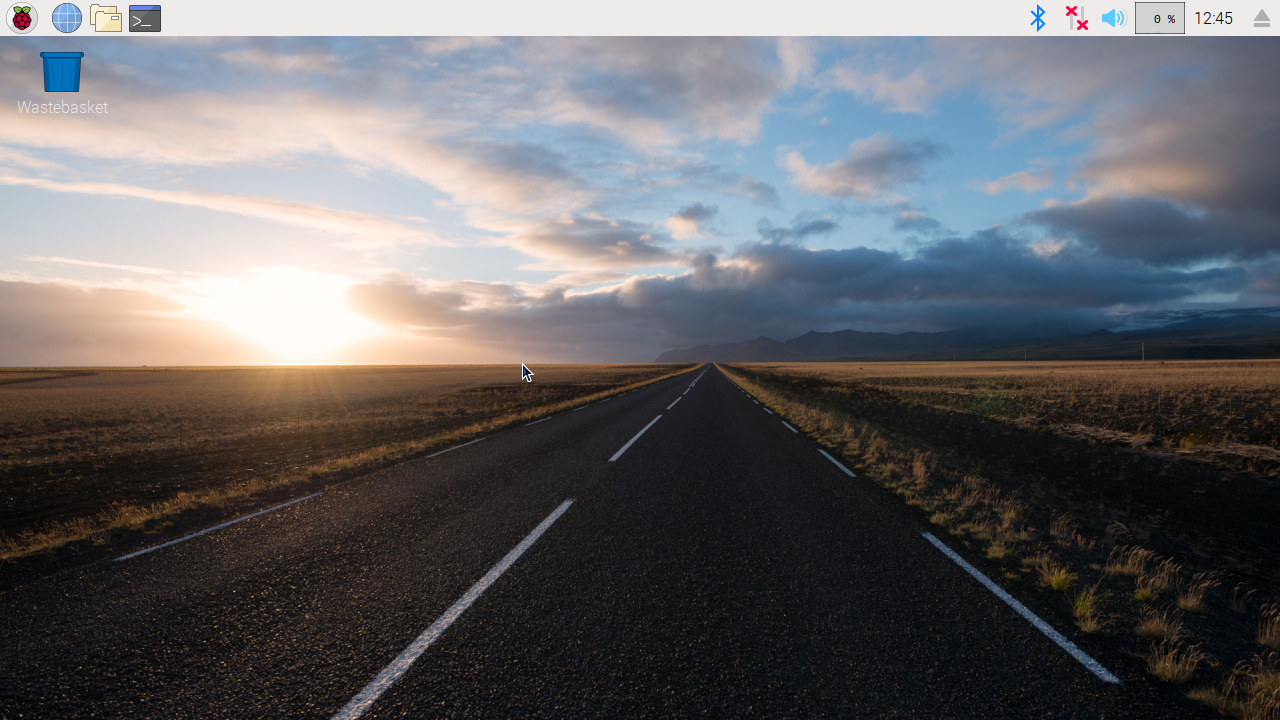
[](https://user-images.githubusercontent.com/42980862/49777538-e62d6f80-fcce-11e8-9446-ed2d50d9548e.png)

Figure 7- Fresh Screen after setup

### Make it wireless

In order to get rid of using HDMI, mouse and a keyboard there's an alternative steps:

* Before making any connection to Rpi make sure to perform the following steps

a. Go to Network and Sharing Center option from control panel and go to your actual Wi-Fi connection to which computer is connected to and then go to properties/Sharing which looks like:

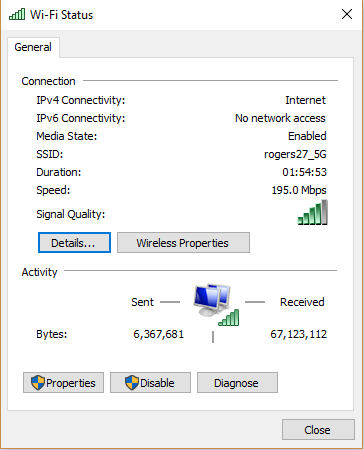
[](https://user-images.githubusercontent.com/42980862/49777786-f1cd6600-fccf-11e8-9720-2d31cc633486.PNG)

Figure 8- Wi-Fi Status

b. Enable the following option:

* +  Allow other network users to connect through this computer's internet connection. And in the dropdown list choose your Ethernet option

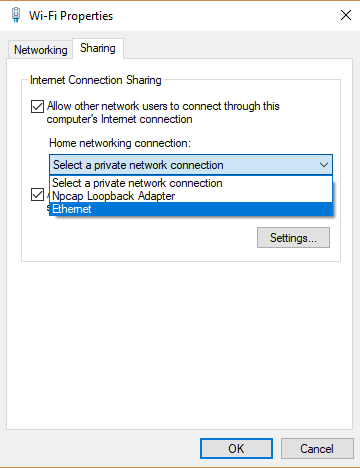
[](https://user-images.githubusercontent.com/42980862/49778059-445b5200-fcd1-11e8-9e32-a4653de4528f.PNG)

Figure 9 - Wi-Fi Properties

This will assign IP address to your raspberry Pi.

c. Go to remote desktop connection and go for raspberrypi.mashome.net

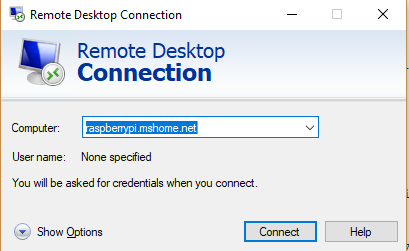
[](https://user-images.githubusercontent.com/42980862/49778147-ab790680-fcd1-11e8-9469-78ef9798cf00.png)

Figure 10 -Remote Desktop Connection

### PCB

But in order to integrate all of them on the same PCB we had to design a new PCB that could accommodate all three sensors.

You can download the file from [here.](https://github.com/GursehajHarika/Drone_DOF/raw/master/Sem6_PCBv2.fzz)

### Coral Case

We also had to design a coral case that can accommodate the raspberry with the sensors on PCB as well as the power source stacked in the same case.

You can download the file from [here.](https://github.com/GursehajHarika/Drone_DOF/raw/master/Case_by_ARMANVELANIv1.cdr)

## Unit-Testing

If you are creating an individual project you can follow the instructions above and test it at each and every step before proceeding further. For example test out the breadboard then test out the PCB then test out the code then check as measurement and test out the case and so on.

If you are considering it as a mass production project I would suggest to have some necessary changes like create a printed multilayer circuit board which takes less space than the design above. Also another thing to consider for the PCB is to make it water resistant. The next item to look for would be the casing as a stackable case idea is not so good for a mass production because it’s delicate. Instead you can go for plastic 3D printed cases and not acrylic laser cut cases.

Respected Kristian,

    As a team, we have so far completed and submitted our proposal, as well as our requirements specifications for hardware and software. Secondly, we also worked on documentation of introduction, abstract, declaration of joint ownership in the technical report. Moreover, we also successfully integrating sensors of all the group members on the same PCB board. We also have a proper acrylic laser cut designed case for the whole unit. We already had our app and database upright to store the sensor’s current readings and logs .We are now successfully able to send our real-time data from the sensor to our database. Lastly, we also updated our build instruction in our technical report.

As independent members,

Gursehaj Harika

Adapted sensors reading with the database in order to make it work with our application. Worked on updating and fixing minor bugs and glitches here in there in the application. Modified the python code to send data from the sensor to the database.

Jay Jadav

Updated the python files of each of the group members. Majorly worked on the 3-Axis accelerometers python file. It was corrupted and needed to be updated according to the project specifications.

Arman Velani

First designed a PCB to integrate all the three sensors, but it wasn’t perfect and needed to be modified. Hence designed a second PCB to integrate all the sensors. Soldered different kinds of headers on PCB to easily attach/detach sensors on it. Worked on fixing the corrupted operating system. As the OS was not ready to boot on multiuser mode, we had to format the SD card and reinstall the new operating system.

Problems

So far we encountered some minor as well as major problems in our project

* We had to design the PCB board twice as the first one designed didn’t match the layout of all the sensors. No changes were made to the case design.
* We had to figure out different python code for one of the sensors as the previous one we had was corrupted.
* The operating system was corrupted. It was starting on emergency shell only. We later took the backup of our files and formatted the SD card. The operating system was reinstalled and the codes were uploaded to the raspberry pi.

Financial Update

    -The labor and part cost for printing the PCB would approximately cost around $25 CAD.

Sincerely,

Arman Velani

Drone\_DOF Team

p.s. the link to Arman’s sensor: [here](https://www.googleadservices.com/pagead/aclk?sa=L&ai=DChcSEwjigrvt2YLhAhUOqmkKHScACMIYABAAGgJpcQ&ae=1&ohost=www.google.com&cid=CAESEOD22v1gVu0SVep-8MWaiw8&sig=AOD64_2iCmM2kN90D08rzLGHlo_--yx9Kg&q=&ved=2ahUKEwiXt7Tt2YLhAhVPaq0KHRwaDqkQ0Qx6BAgLEAE&adurl=https://www.digikey.ca/catalog/en/partgroup/mma8451-triple-axis-accelerometer/49492%3Futm_adgroup%3DGeneral%26mkwid%3Ds0TpRamVn%26pcrid%3D267488348137%26pkw%3D%26pmt%3Db%26pdv%3Dc%26productid%3D%26slid%3D%26gclid%3DCjwKCAjw96fkBRA2EiwAKZjFTZLoN2fePJxN-qBznux24_dHv4iXUkwXYNEsHJGpDUwF2KQayz-nmhoCI2sQAvD_BwE)

## Production-Testing

For the industrial production of the Drone\_DOF, many manual steps can be made automated. Raspberry Pie do not need to program individually, the OS can be pre-installed on a SD card. All the Hardware elements such as sensors used in Drone\_DOF can be tested out mechanically, without any human interferences. The whole PCB designing process can be made automated. The whole Soldering job of the Project can be made automated with the use of industrial soldering. The Case of the Drone\_DOF can be made using 3D printing machine instead of using laser-cut acrylic case in order to make it stronger and also lighter in weight. Lastly a whole assembly line can be set up in order to assemble all the parts in one, and can have a final testing of the whole unit at the end just before sending them to the market.

## SENSOR 1: Barometric sensor (MS5611)

### Introduction

Have you ever wanted to know what your height was or how high something goes without calculating its speed and figuring out its distance traveled? My senor (Barometric Pressure Senor is what you're looking for.

This Sensor is able to read the height of the object or the temperature around it. Here’s how this sensor works with an android application. If you wish to work on the application as well. I would highly recommend you clicking on here's the System diagram on how it will work, application as well.

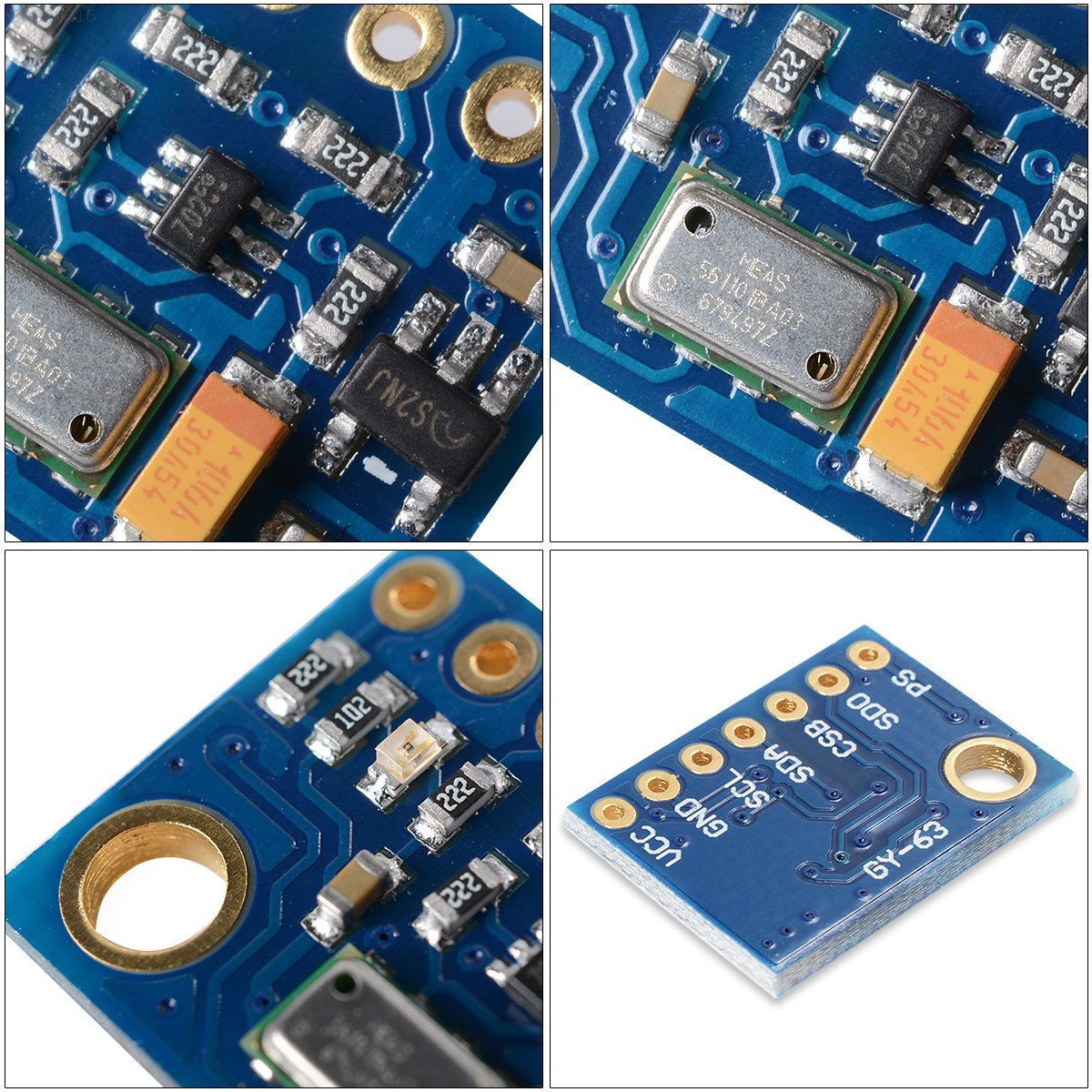
[](https://user-images.githubusercontent.com/43185907/49833042-5935f500-fd66-11e8-84f8-3ecc64b90181.jpg)

Figure 11- MS5611

### Parts, Tools and Budget

These are the parts and Tools required to make,

Please Note: - Some of the parts will take up to a month to be delivered to your door step.

[](https://user-images.githubusercontent.com/43185907/49832105-bda38500-fd63-11e8-91b4-6f54d5e169e2.jpeg)

Figure 12- Soldering Machine

Lead free soldering wire (Recommended) or a soldering wire.

Barometric Pressure Sensor - $11.99

And a Parts Kit that I bought from Humber but you can get it from where ever you feel like, as long as it has the same tools.

### Mechanical Assembly

Once you get your Raspberry Pi 3. And keep your micro-SD card handy, you'll be needing it soon. Go to Raspberry Pi’s Website and download Raspbian full desktop version and get a software to format the SD card to NFTS format. Flash your Raspbian OS to your SD Card. Connect your Raspberry Pi to a Monitor. Grab a Keyboard, a mouse and a HTMI or VGA (Depending on what type of monitor you have.

**Please Note: - Do not Power On your raspberry Pi before you have plugged in all your I/O Devices.**

Wait for the RASPBIAN OS to boot up and initialize itself and you'll be greeted with a home Screen that looks like this.

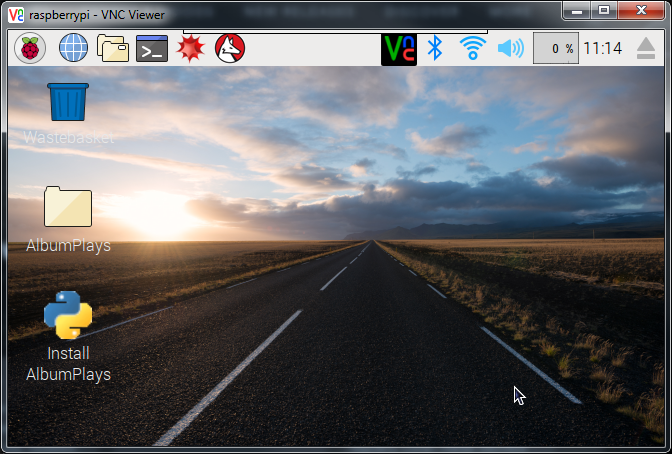
[](https://user-images.githubusercontent.com/43185907/49835201-753c9500-fd6c-11e8-8d28-7d7a5e1cb914.png)

Figure 13- Raspberry Screen

Once you’re case is complete and raspberry pi is connecting to a display we have to work on making the Raspberry Pi Remotely accessible, Go to YouTube and follow this [tutorial](https://www.youtube.com/watch?v=WAFaw2Mbnko) after you're done with the tutorial your Raspberry Pi should work with remote desktop and now you don't need a keyboard and a mouse or even a monitor just to run your Raspberry Pi.

Now open Terminal and Type in i2c detect -y 1 and you should see 0x76. if not please look at your PCB Design and your circuit.

Now Create a [Python Code](https://github.com/GursehajHarika/barometricsensor/blob/master/ms5611.py) and that code will run and you'll be able to get some readings from it which should look something like this.

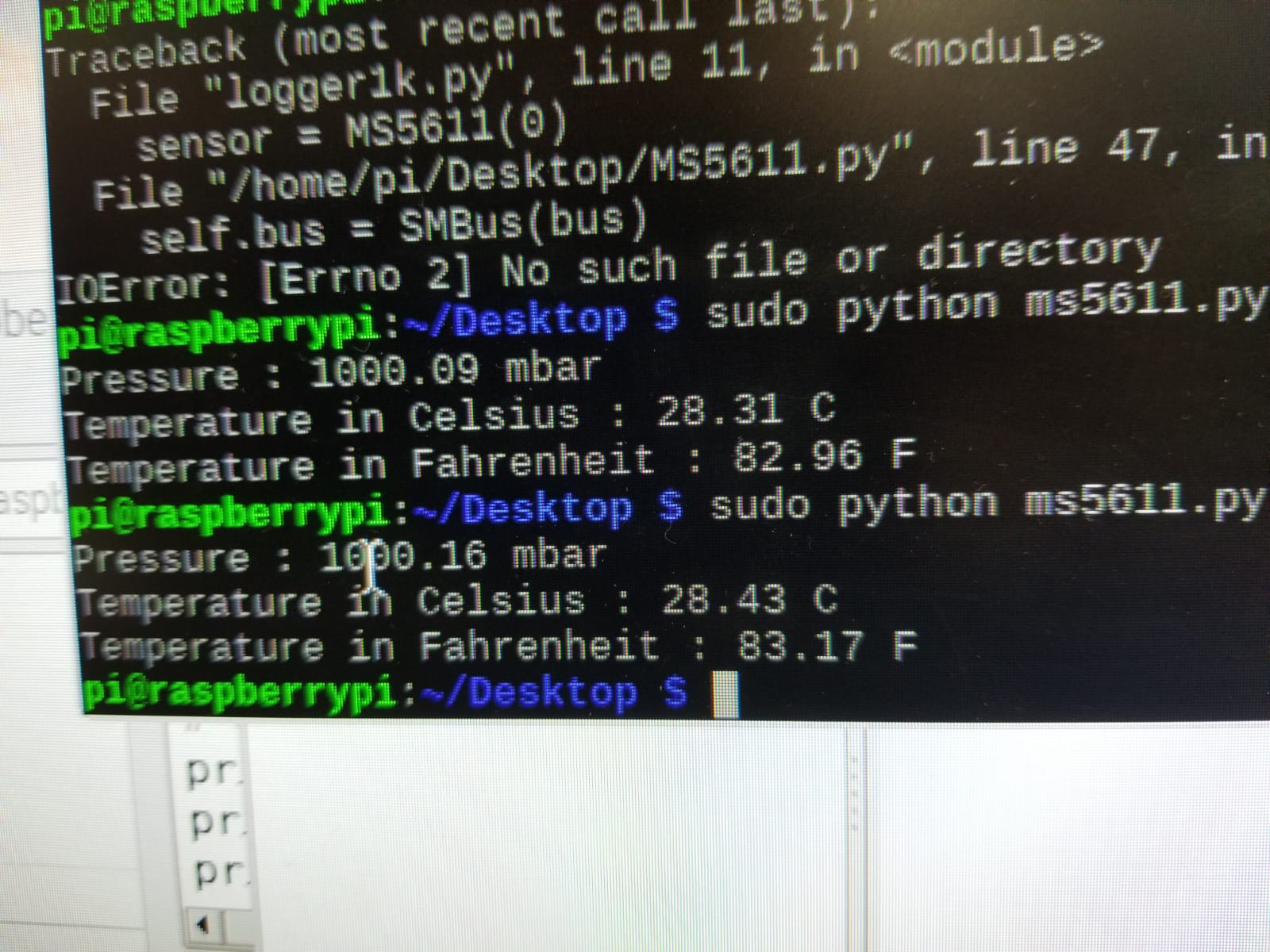
[](https://user-images.githubusercontent.com/43185907/48626353-a5686200-e97f-11e8-95df-3de6a83a0182.jpeg)

Figure 14- Readings of MS5611

And in order to run it, type in “sudo python YOUR\_FILE\_NAME.py "and you should be good to go.

## SENSOR 2: 3-Axis Accelerometer (MMA8451)

### Introduction

The MMA8451 is a low-cost but high-precision digital accelerometer that uses repeated-start I2C mode, with adjustable data rata and 'range' (+/-2/4/8).In this project it is used to measure the accel readings using raspberry pi 3.

### Initial Preparation

Before starting with any project create your Budget about the project. Once you have your Budget create a Schedule for your project. My project total Budget was $384 and the total time to complete the project was 4 months. The budget may vary as the prices of the parts changes

### Parts, Tools and Budget

1. Raspberry Pi 3 Ultimate Starter kit-32 GB Edition. $141.17
2. Adafruit Triple-Axis Accelerometer - 2/4/8g @ 14 bit - MMA8451 $30.5
3. Anker Unibody Aluminum USB 3.0 to Rj45 Gigabit Ethernet Adapter Supporting 10/100/1000 Mbps Ethernet [RTL8153 Chipset] $31.53
4. Parts kit [tool box] $135.60
5. headers [1xGPIO female header, [1x8 female header](https://www.creatroninc.com/product/stackable-header-set-for-arduino/?search_query=8+pin+stackable+header&results=48), [Stackable Headers](https://www.creatroninc.com/product/stackable-header-for-raspberry-pi/)]
6. [Raspberry Pi 3 and sensor case].(https://github.com/ArmanVelani/3-AxisAccelerometer/raw/master/Pi2Casestack.cdr)
7. LED, soldering station, Tweezers other [cables](https://www.amazon.ca/Copper-Jumper-Single-Conductor-AWG30/dp/B00HR6JX80/ref=sr_1_7?ie=UTF8&qid=1544565319&sr=8-7&keywords=jumper+cables+pcb)
8. Wireless keyboard and mouse with usb receiver. $45.20
9. 4xM2.5 bolts, 4xM2.5 nut. 10.One HDMI monitor
10. One HDMI cable.

### Mechanical Assembly

#### Setup Raspberry Pi

This section explains how to setup the whole things to test your MMA8451 device on a Raspberry Pi. Once you have all the parts with you start with your raspberry Pi. Make Sure you do not power on the Raspberry Pi, Monitor or any other Components before finishing the setup. Connect a wireless keyboard/Mouse to your Raspberry Pi. Also attach one end of HDMI cable to your Raspberry Pi and other end to the Monitor. Once connected power on the raspberry pi and complete the Setup. For Help on Setting up raspberry Pi 3 visit this [YouTube Video.](https://www.youtube.com/watch?v=xBlxuf_LSCM)

#### Connection

Now you need to Power off your raspberry Pi and disconnect the Power Source. Connect you MMMA8451 using some jumper cables as shown in the image below.

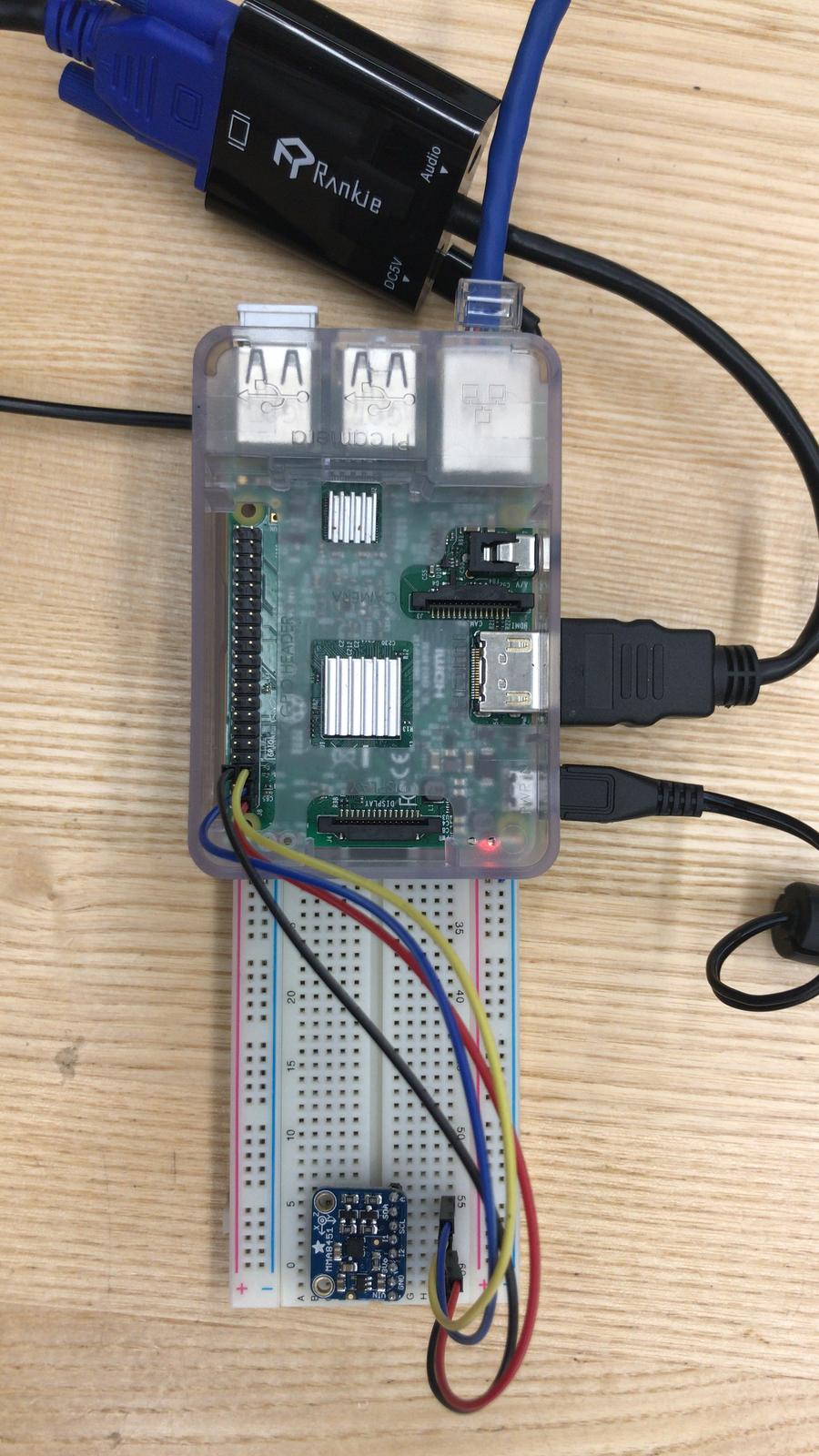
[](https://user-images.githubusercontent.com/43188450/48366908-35a36000-e67d-11e8-9476-107ec0f8258f.jpeg)

Figure 15-BreadBoard

* raspberry Ground- MMA8451 Ground
* raspberry 3.3V- MMA8451 3.3V
* raspberry SCA- MMA8451 SCA
* raspberry SCL- MMA8451 SCL

Once you have your Connections connect Power to raspberry pi and power it on. User command I2C Detect to check if the Sensor is connected properly.

#### Code

Now you have you PCB ready but it is not still reading the code in order to read from the sensor you will need a python code. [Get the code](https://github.com/ArmanVelani/3-AxisAccelerometer/blob/master/PythonCode.txt). Compile the code and run it using 'sudo python3 ' this is the [video](https://github.com/ArmanVelani/3-AxisAccelerometer/raw/master/WhatsApp%20Video%202018-11-13%20at%2010.57.24%20PM.mp4) of how the sensor should work when the code is running successfully. This is how the [code output](https://user-images.githubusercontent.com/43188450/48862483-81988800-ed94-11e8-902c-ec6e4a4453eb.jpeg) and [PCB](https://user-images.githubusercontent.com/43188450/48862531-a260dd80-ed94-11e8-9e08-a28f0783966a.jpeg) looks.

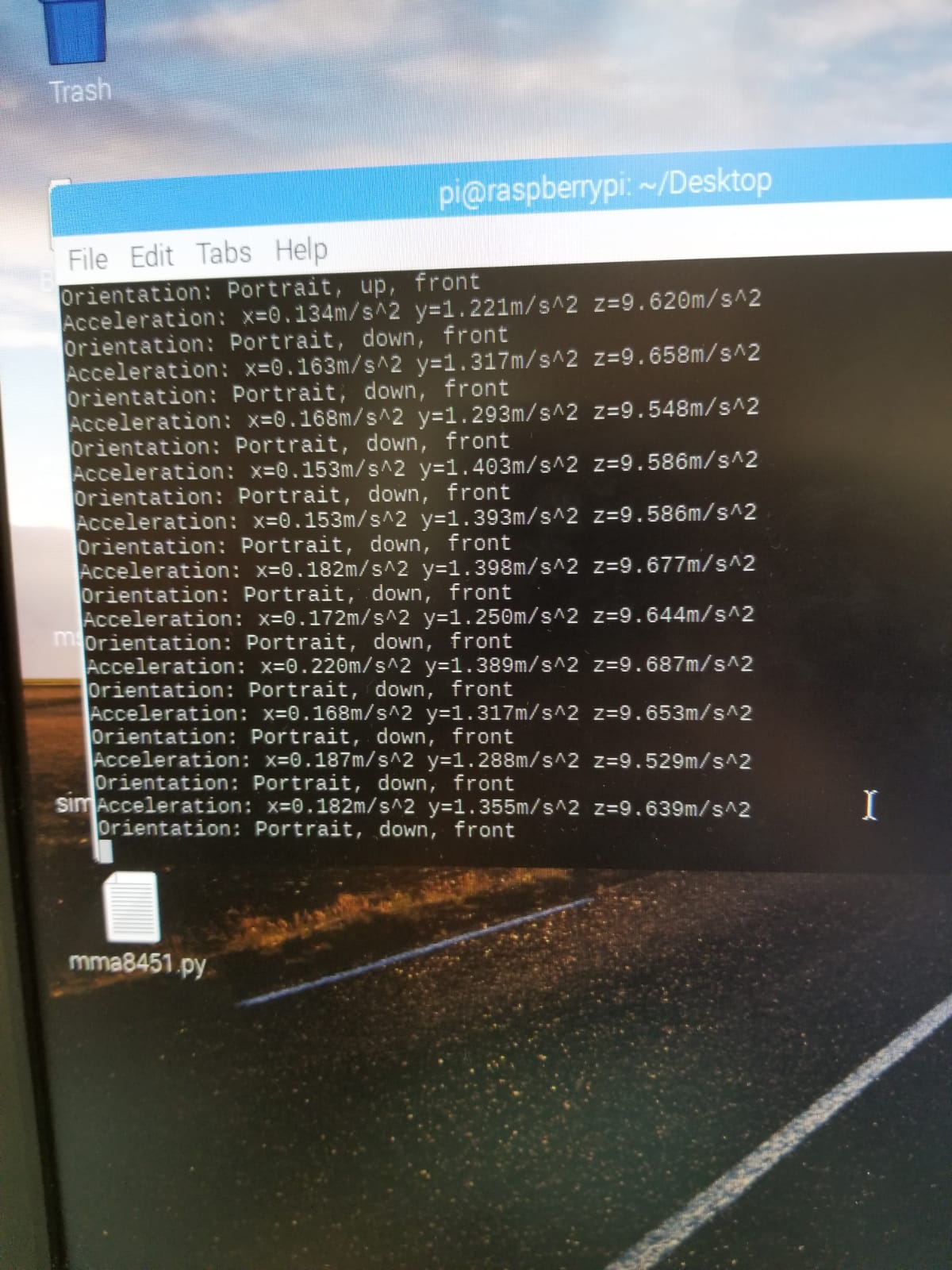


Figure 16- Readings of MMA8451

## SENSOR 3: 3 axis Magnetometer (MAG3110)

### Introduction

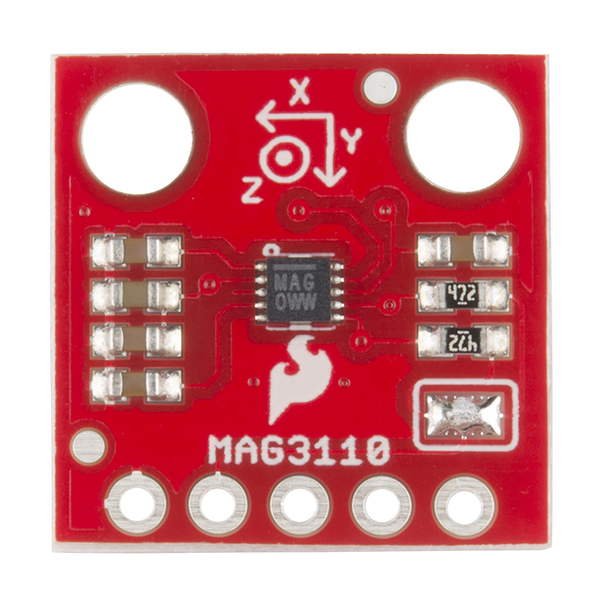
Freescale's MAG3110 is a small, low-power, digital 3-axis magnetometer. The device can be used in conjunction with a 3-axis accelerometer to realize an orientation independent electronic compass that can provide accurate heading information.[](https://user-images.githubusercontent.com/43185906/49823957-e2d9c880-fd4e-11e8-8f70-44ee31da6da7.jpg)

Figure 17-MAG3110

You can buy Raspberry Pie from [here](https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/) and the MAG3110 sensor from [here](https://www.sparkfun.com/products/12670).

### Functionality

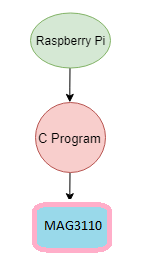
[](https://user-images.githubusercontent.com/43185906/49824195-8c20be80-fd4f-11e8-9433-6161749a942a.png)

Figure 18-Functionality

### Parts, Tools and Budget

The total cost of producing this project is heavily inflated due to the cost of the soldering kit that was supplied in the lab during development. Any generic soldering iron can be used for this project.

The total cost after removing the soldering kit is: **$398.27 CAD** after HST. This includes all the tools used in completing the project (e.g. wire cutters, needle nose pliers, breadboard, etc.)

Notable purchases include: Raspberry Pi 3B+ Kit ($107.29 CAD) and MAG3110 3-axis Magnetometer ($24.76 CAD).

#### Raspberry Pi 3b+

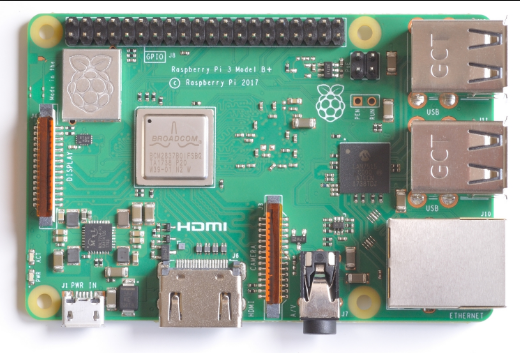
[](https://user-images.githubusercontent.com/42980862/49776194-078b5d00-fcc9-11e8-8d61-f96a17dfd31c.PNG)

Figure 19-RPI 3B+

Also known as Rpi 3b+ is the newest as well as fastest pocket sized computer.

#### Features and Specifications:

* 1.4GHz 64-bit quad-core processor
* dual-band wireless LAN
* Bluetooth 4.2/BLE
* Faster Ethernet
* Power-over-Ethernet support
* 4 USB 2.0 ports
* 5V/2.5A DC power input
* Micro SD for setting up Operating System as well as storing personal data

### Mechanical Assembly

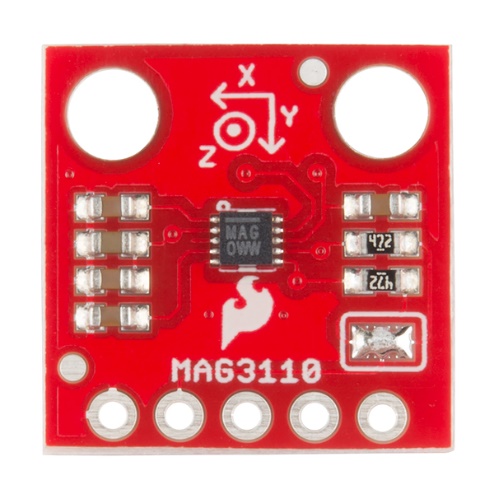
[](https://user-images.githubusercontent.com/43185906/49823957-e2d9c880-fd4e-11e8-8f70-44ee31da6da7.jpg)

Figure 20- MAG3110(2)

#### Features and Specifications:

* 1.95V to 3.6V Supply Voltage
* 7-bit I2C address = 0x0E
* Full Scale Range ±1000 μT
* Sensitivity of 0.10 μT
* Pull Up Resistor Jumper
* 13.3 x 14.5 mm

#### Steps for pinout:

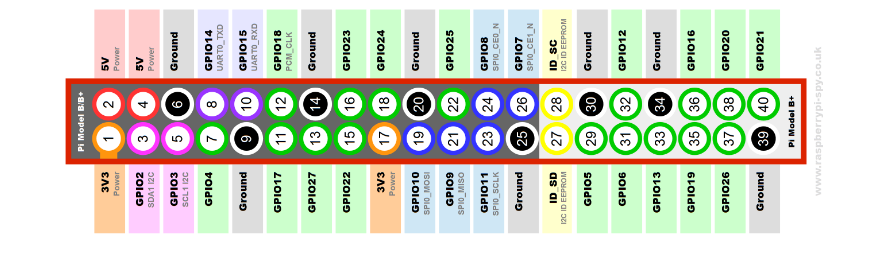
[](https://user-images.githubusercontent.com/43185906/49830052-1a9c3c80-fd5e-11e8-863e-13e068d79e84.PNG)

Figure 21 Pinout for RPI

1. Solder your sensor
2. Follow the RPI Header reference for your pinout to breadboard
3. Look for the 3V/5V pin from your Rpi and connect it to breadboard (pin no.1)
4. Connect VCC to 3V power supply and GND to common ground (pin no.5 in 2nd Row)
5. Now connect your SCL to I2C clock SCL pin on your RPi(pin no.3 in 2nd Row)
6. Connect SDA to I2C clock SDA pin on your RPI (which is pin no.2 in 2nd Row)

#### Breadboard

To ensure the MAG3110 has been connected properly for I2C communications, the following command should be entered in to the Pi's terminal: sudo i2cdetect -y 1. This will display a simple graphic listing each device connected to the I2C bus and its corresponding address. The address the MAG3110 uses is 0x0e.

To construct the PCB, the prototype lab located at Humber College was used. However, any third party production facility may be used, as the files are universally accepted as an industry standard. The two images below show the PCB constructed from both the top and bottom. A 6 pin header has been soldered to the top of the PCB to connect to the MAG3110 and a 68 pin header has been soldered to the bottom to connect to the GPIO pins on the Pi.

To run the following code on rpi, issue the following command sudo python3 MAG3110.py

This is the code to be used:-

**MAG3110.py** [MAG3110.zip](https://github.com/JayJadav/smartware/files/2618144/MAG3110.zip)

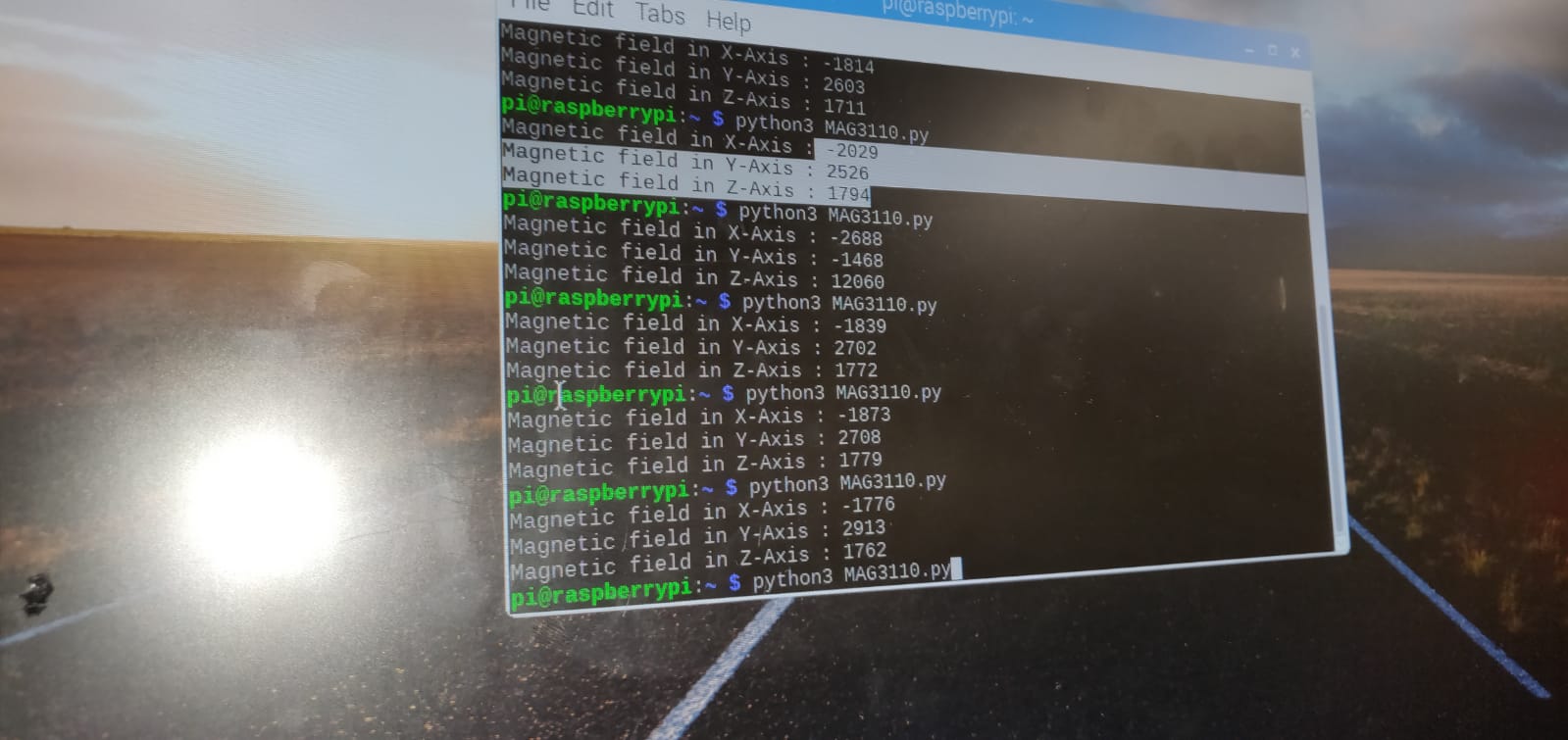
**Readings:** [](https://user-images.githubusercontent.com/43185906/49055855-34b71600-f1c7-11e8-91f5-171996c0d903.jpeg)

Figure 22-Readings of MAG3110

# Collaboration

Besides this project, we also collaborated with the mechanical engineering technology department on making a label dispenser. The students from mechanical engineering department were supposed to find a drone and create a custom and lighter case for the drone and attachment. The students of the mechanical department never showed up or contacted us with their part of work. Still we worked on our part as it seemed to be an interesting side project. With the help of our professor Mr. Kristian Medri we started by building the code for it. The project was totally based on the Arduino Uno. We did not have to spend any more on the budget as all the materials were provided by engineering department. We used three universal breadboards to create a temporary circuit. Here we connected an ultrasonic sensor to detect the presence of human hand. The sensor then starts a motor on which the label roll is attached to. The label had a small black dot at the end which could be used to stop the motor once a sticker is outside. We used a led and a photo diode to detect the black spot on the label. A delay was set using the code so that the label does not stop until it is completely outside. Since no contact or update was received from the other students we eventually lost the interest in the project and just paused working on it for now. In future we are planning to create a custom PCB instead of using breadboard. Also, we will be creating a laser cute acrylics case for the design. Some of the parts like the roller for the label roll and the other gears will be 3d printed. This whole project was on schedule and so far, everything was completed in almost 5 hours of time.



Figure Label Dispenser image 1

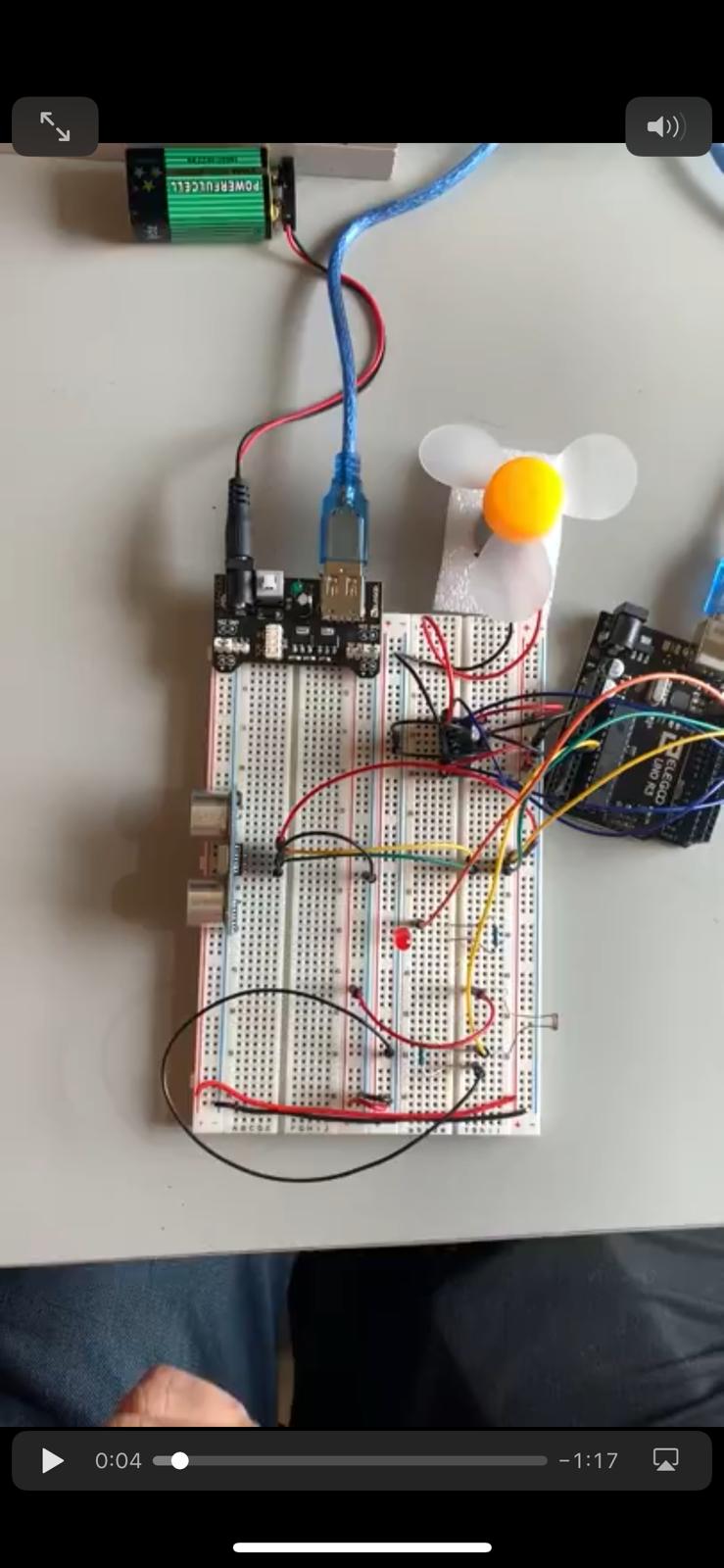


Figure Label Dispenser Working Model

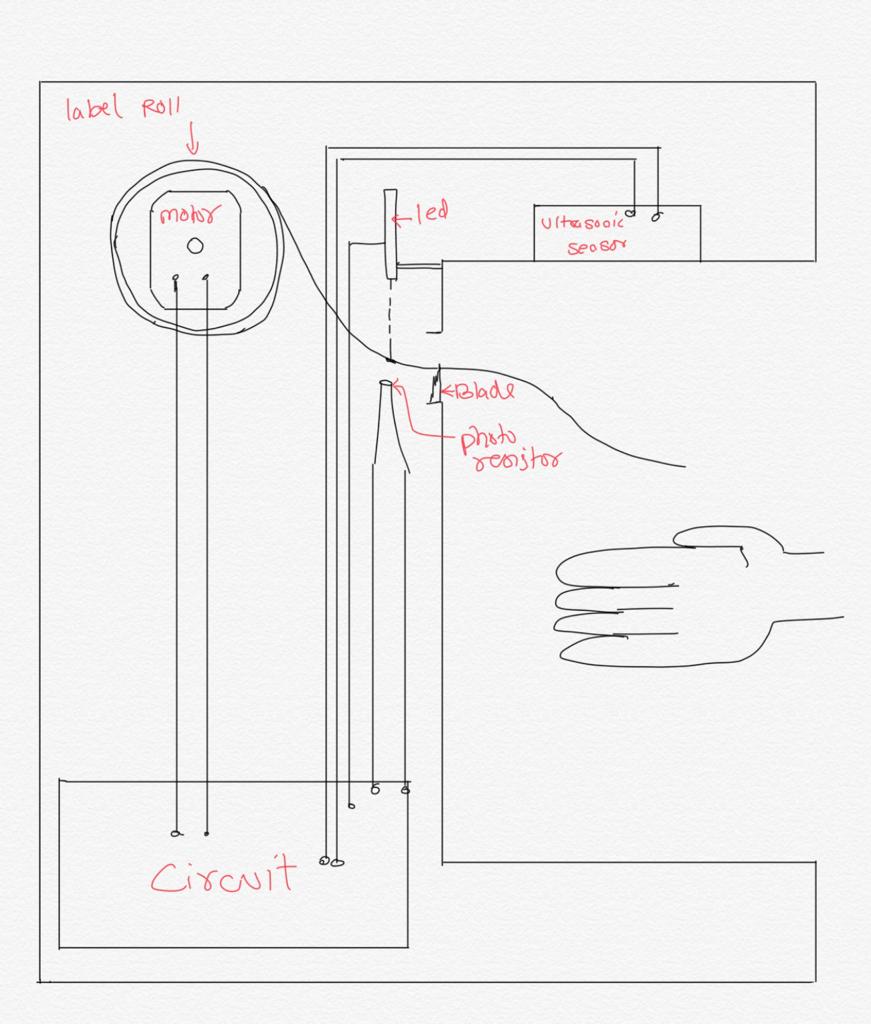


Figure Graphical Explanation

# Conclusion

The Drone\_DOF is a drone attachment unit comprising of various sensors that allows user to know their precise location as well as the change in the pressure that it faces with the changes in the altitude. By integrating the drone \_DOF we are able to address the problem of locating your mobile drone with minimal to less error in its position with the help of magnetometer sensor(MAG3110) and accelerometer sensor(MMA8451); so that the user is always updated with its accurate position. Moreover the barometric pressure sensor(MS5611) help user to know the elevation as well as the safety height above which flying of the drone would be dangerous and out of the connectivity zone from the android as well as the firebase database. The device also includes a 5000 mAh additional power source (battery) that powers itself and avoids having any loose wiring to the whole unit. The device also has the option to supply power directly to the raspberry pi.

# Recommendation

There are couple of recommendations that we as a team think that should be applied to the final product is we go for industrial production. The model we have right now uses Wi-Fi configuration and does not support cellular or long distance configuration. So the next model should have a SIM card attached to it, and will be able to work on cellular data, so that we are able to configure the drone for long distance area. The second thing that we would like to modify is that outer-case as the case right now we have is made of laser-cut acrylic sheets which is not sturdy enough for the outer atmosphere, so what we like to be added is a 3D designed plastic case which is sturdy enough to function in outer atmosphere without any malfunction. . With minor modifications the same type of components and programming could be implemented at each of the plants and the financial benefits could fairly quickly cover the capital costs. The simple payback period (disregarding interest and full time operator costs) could be as little as two years

# Bibliography

The problem that the Today in 21st century there are multiple flying objects, mostly Drones. The products that move or fly have a high risk of getting lost, crashing or even getting destroyed when being used or being controlled by amateurs. Numerous industries such as marine exploration, automotive industry, aviation industry, also defense and military now uses drones for places they can’t reach or they don’t want to go. This is a high risk to safety as it can crash over someone or can cause fire. Also if lost they are hard to find. There should be a way to get feedback from these devices in order to avoid major accidents. Our team came forward and found a solution to this. The citation source that were used are as follows:

1. Maboshi. (2018, November 07). Arduino GPS Drone RC Boat. Retrieved from <https://www.hackster.io/maboshi/arduino-gps-drone-rc-boat-45d6f4>
2. Autonomous High Altitude Glider. (n.d.). Retrieved from <https://create.arduino.cc/projecthub/53982/autonomous-high-altitude-glider-055aa3?ref=tag&ref_id=drones&offset=9>
3. Vignesh. (2018, December 17). Intelligence Monitoring Drone System. Retrieved from <https://www.hackster.io/enigma-plasma-8/intelligence-monitoring-drone-system-727470>
4. Velani, A. (n.d.). November 27th, 2018 (Week 13). Retrieved from <https://armanvelani.github.io/3-AxisAccelerometer/>

For the sensors and the raspberry pie, the product and the code for it was derived from:

1) Sparkfun

<https://www.sparkfun.com>

2) Adafruit

<https://www.adafruit.com>

3) Github

<https://github.com>

# Glossary of terms:

**Python -** Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

**mAh -** mAh means milliamp Hour and is a unit that measures (electric) power over time. It is commonly used to measure the energy capacity of a battery. In general, the more mAh and the longer the battery capacity or battery life.

**Magnetometer -** A magnetometer or magnetic sensor is an instrument that measures magnetism—either the magnetization of a magnetic material like a ferromagnet, or the direction, strength, or relative change of a magnetic field at a particular location.

**Accelerometer-** An accelerometer is a device that measures proper acceleration. Proper acceleration, being the acceleration of a body in its own instantaneous rest frame, is not the same as coordinate acceleration, being the acceleration in a fixed coordinate system.

**Barometric pressure sensor-** The barometric sensor, also commonly known as the barometric air pressure sensor (BAP), is a type of engine management sensor commonly found on many vehicles. It is responsible for measuring the atmospheric pressure of the environment that the vehicle is driving in. Different environments will have different atmospheric pressures, which will have an effect on how the vehicle runs.

**CRUD-** In computer programming, create, read, update, and delete are the four basic functions of persistent storage. Alternate words are sometimes used when defining the four basic functions of CRUD, such as retrieve instead of read, modify instead of update, or destroy instead of delete.

**Soldering-** It is a process in which two or more items are joined together by melting and putting a filler metal (solder) into the joint, the filler metal having a lower melting point than the adjoining metal. Unlike welding, soldering does not involve melting the work pieces.

**N00BS-** n00b is a slang term for a novice or newcomer, or somebody inexperienced in a profession or activity. Contemporary use can particularly refer to a beginner or new user of computers, often concerning Internet activity, such as online gaming or Linux use.

**VGA Cable-** A Video Graphics Array connector is a three-row 15-pin DE-15 connector. The 15-pin VGA connector was provided on many video cards, computer monitors, laptop computers, projectors, and high definition television sets.

**SSH command-** SSH, also known as Secure Shell or Secure Socket Shell, is a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network. SSH also refers to the suite of utilities that implement the SSH protocol.

**GPIO -** GPIO. Stands for "General Purpose Input/Output." GPIO is a type of pin found on an integrated circuit that does not have a specific function. While most pins have a dedicated purpose, such as sending a signal to a certain component, the function of a GPIO pin is customizable and can be controlled by software.

# Appendices

**Conversion of magnetometer‘s readings to readable values**

|  |
| --- |
|  |
| xMag=data[0]\*256+data[1] |
|  | if xMag > 32767 : |
|  | xMag -= 65536 |
|  |  |
|  | yMag = data[2] \* 256 + data[3] |
|  | if yMag > 32767 : |
|  | yMag -= 65536 |
|  |  |
|  | zMag = data[4] \* 256 + data[5] |
|  | if zMag > 32767 : |
|  | zMag -= 65536 |
|  |  |

**Getting Time Stamp**

Using Jan 1st 1970 as start time to calculate the total number of seconds elapsed.

|  |  |
| --- | --- |
|  | ts = calendar.timegm(time.gmtime()) |
|  | print (ts) |

Imports used

|  |
| --- |
|  |
| import calendar |
|  | import time |
|  |  |

**Conversion of Barometric Pressure’s Readings**

|  |  |
| --- | --- |
|  | value = bus.read\_i2c\_block\_data(0x77, 0x00, 3) |
|  | D2 = value[0] \* 65536 + value[1] \* 256 + value[2] |
|  |  |
|  | dT = D2 - C5 \* 256 |
|  | TEMP = 2000 + dT \* C6 / 8388608 |
|  | OFF = C2 \* 65536 + (C4 \* dT) / 128 |
|  | SENS = C1 \* 32768 + (C3 \* dT ) / 256 |
|  | T2 = 0 |
|  | OFF2 = 0 |
|  | SENS2 = 0 |
|  |  |
|  | if TEMP >= 2000 : |
|  | T2 = 0 |
|  | OFF2 = 0 |
|  | SENS2 = 0 |
|  | elif TEMP < 2000 : |
|  | T2 = (dT \* dT) / 2147483648 |
|  | OFF2 = 5 \* ((TEMP - 2000) \* (TEMP - 2000)) / 2 |
|  | SENS2 = 5 \* ((TEMP - 2000) \* (TEMP - 2000)) / 4 |
|  | if TEMP < -1500 : |
|  | OFF2 = OFF2 + 7 \* ((TEMP + 1500) \* (TEMP + 1500)) |
|  | SENS2 = SENS2 + 11 \* ((TEMP + 1500) \* (TEMP + 1500)) / 2 |
|  |  |
|  | TEMP = TEMP - T2 |
|  | OFF = OFF - OFF2 |
|  | SENS = SENS - SENS2 |
|  | pressure = ((((D1 \* SENS) / 2097152) - OFF) / 32768.0) / 100.0 |
|  | cTemp = TEMP / 100.0 |
|  | fTemp = cTemp \* 1.8 + 32  Code for Arduino and the Collaborated Project |
|  | #include <NewPing.h>    #define TRIGGER\_PIN 12  #define ECHO\_PIN 11  #define MAX\_DISTANCE 200  //Constants  const int pResistor = A0; // Photoresistor at Arduino analog pin A0  const int ledPin=9;  //Motor A  int enableA = 5;  int MotorA1 = 6;  int MotorA2 = 7;  int value;    NewPing sonar(TRIGGER\_PIN, ECHO\_PIN, MAX\_DISTANCE);    void setup() {  //configure pin modes  pinMode (enableA, OUTPUT);  pinMode (MotorA1, OUTPUT);  pinMode (MotorA2, OUTPUT);  Serial.begin(115200);  pinMode(ledPin, OUTPUT); // Set lepPin - 9 pin as an output  pinMode(pResistor, INPUT);// Set pResistor - A0 pin as an input (optional)  }    void loop() {    delay(50);  Serial.print("Ping: ");  Serial.print(sonar.ping\_cm());  Serial.println("cm");  digitalWrite (enableA, LOW);  if (sonar.ping\_cm()<10){  //enabling motor A  Serial.println ("Enabling Motors");  digitalWrite (enableA, HIGH);  //delay (1000);  //do something  //Serial.println ("Motion Forward");  digitalWrite (MotorA1, LOW);  digitalWrite (MotorA2, HIGH);  value = analogRead(pResistor);    //You can change value "25"  //digitalWrite(ledPin, HIGH);  if (value > 25){  digitalWrite(ledPin, HIGH); //Turn led off  digitalWrite (MotorA2, HIGH);  digitalWrite (MotorA1, LOW);  digitalWrite (enableA, HIGH);  }  else{  digitalWrite(ledPin, LOW); //Turn led on  digitalWrite (enableA, LOW);  delay(5000);  }    }  else{  Serial.println ("Stoping motors");  //stop  //digitalWrite (enableA, LOW);  //digitalWrite (enableA, HIGH);  }  } |