The Thesis entitled

The Design, Construction and Implementation of an Autonomous Outdoor Quadcopter using an RPi microcomputer and a APM 2.8 flight controller

Submitted To

THE MAHARAJA SAYAJIRAO UNIVERSITY OF BARODA

In Fulfillment of the Requirement for the Degree of

BACHELOR OF ENGINEERING

in

MECHANICAL ENGINEERING

by

Kishan Ajudiya (PRN - 2017033800116786)

Under the guidance of

Mr. Akash Pandey

DEPARTMENT OF MECHANICAL ENGINEERING

FACULTY OF TECHNOLOGY AND ENGINEERING

THE MAHARAJA SAYAJIRAO UNIVERSITY OF BARODA

CERTIFICATE

This is to certify that the dissertation report entitled **"The Design, Construction and Implementation of an Autonomous Outdoor Quadcopter using an RPi microcomputer and a APM 2.8 flight controller"**, being submitted by the following students in the partial fulfilment of the requirement for the award of the degree of **"BACHELOR OF ENGINEERING (MECHANICAL)"** at The Maharaja Sayajirao University of Baroda, Vadodara.

This is a record of their own work carries out under our supervision and guidance. The matter embodied in the thesis, is to the best of our knowledge, has not been submitted for the award of any other degree or diploma of any institute or university.

Name	Exam Number	Roll Number	PRN
Kishan Ajudiya	808079	253	2017033800116786

GUIDE

Dr. Akash Pandey,
Assistant Professor,
Department of Mechanical Engineering,
Faculty of Technology and Engineering,
The Maharaja Sayajirao University of Baroda.

HEAD

Pro. (Dr) J.M. Prajapati
Department of Mechanical Engineering,
Faculty of Technology and Engineering,
The Maharaja Sayajirao University of Baroda.

ACKNOWLEDGEMENTS

I would like to thank Professor Dr. Akash Pandey for giving me the opportunity to work under his supervision and he was always available to help me throughout the entire project.

We owe a deep sense of gratitude to Ex-Head MR. D.S. Sharma and Head Mr. J.M. Prajapati for his motivation, guidance and for timely provision of all the necessary facilities required during our project work.

ABSTRACT

This thesis focuses on the design and construction of an outdoor aerial vehicle (quadcopter) with autonomous flight functions and the development and implementation of an on-board computing module in an unmanned aerial vehicle (UAV) based on a low-cost single-board computer.

In recent years Quadcopter has been used in many applications such as military, security & surveillance, service delivery, disaster rescue and much more due to its flexibility of flying. Drone control is a complex task that requires knowledge of working principles of both the drone being controlled and its controller. Keeping track of the quadcopter's trajectory, as well as simultaneously manipulating the transmitter controls, can be a difficult task for most users. Therefore, minimizing the cognitive load associated with the transmitter operation is the focus of this research. This thesis proposes speech commands as a replacement for the standard RF transmitter-based form of quadcopter control. The solution presented takes advantage of the widespread availability of smartphones and internet access.

The Quadcopter will execute an autonomous flight using the concept of companion PC. Raspberry PI 4 (RPI4) will control the Quadcopter by commanding the controller of the drone (Pixhawk) by using the DroneKit-Python API to send MAVLink messages to the ArduPilot. This concept is useful to perform an additional task to the autopilot and provide such a smart capability like image processing and path planning which cannot be done by the flight controller alone.

The construction phase of the study resulted in the creation of an operational quadcopter which achieved autonomous flight.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

RPi ……Raspberry Pi
PC ……Personal Computer
RTL ……Return to Home
MAV ……Micro Air Vehicle
UDP ……Use Data Protocol
UAV ……Unmanned Aerial Vehicle
APM ……ArduPilot Mega
GUI ……Graphic User Interface
RPM ……Rotations per minute
CPU ……Control Processing Unit
UAV ……Unmanned Aerial Vehicle
BLDC ……Brushless Direct Current
ESC ……Electronic Speed Controller
GPS……Global Positioning System
FPV ……First-Person-View
LiPo ……Lithium Polymer
OS ……Operating System
RPI ……RaspberryPi
UAV ……Unmanned Aerial Vehicle

CHAPTER 1: INTRODUCTION

1.1 Research Background

The Unmanned Aerial Vehicle (UAV) or drone is a vehicle without a pilot on it. There is a system for the UAV called the Unmanned Aerial System (UAS) that allows communication with the physical drone. Drones are often controlled by a human pilot which is a user input by the way of using the remote control that is known as Radio Controller (RC). But also, they can be autonomously controlled by the system integrated on the drones themselves without using the RC input. The UAS is installed on the companion PC which is called an onboard computer.

There are three major kinds of drones:

1. fixed wings.
2. single motor.
3. multirotor.

Figure 1.1: Fixed Wing Drones

Figure 1.2: Single Motor Drones

Figure 1.3: Multirotor Drones

The multirotor is the most common among them, they are classified depending on the number of motors they have such as tri-copter (three motors), Quadcopter (QC, four motors), hex copter (six motors) and octocopter (eight motors). The most common multirotor is quadcopter. The traditional fixed-wing UAVs can fly for long distances but require runways or wide-open spaces for taking -off and landing. On the other hand, the more trending multirotor UAVs are extremely maneuverable but cannot be used for long-distance flights because of their slower speeds and relatively higher consumption of energy. The flight is

more stable compared with a single rotor and the fixed wing in terms of taking off, landing and controlling the maneuvering of the multi-rotor.

The drone was initially developed for a military purpose. Most of the studies are focused on issues of higher performance hardware platforms, software systems, aerodynamic models, and autonomous flight control. In 2009, 3D Robotics and the Swiss Federal Institute of Technology in Zurich introduced the open-source Autopilot system that is named (ArduPilot Mega) (APM) and its newer versions named Pixhawk. The new production of drones supports various and important types of sensors such as barometer, air speed, air pollution, Light Detection and Ranging (LiDAR), Ultrasonic sensors, etc., with support of wireless communication for example Wi-Fi, 4G and so on.

Several studies have been devoted and deployed with various issues towards design and implementation of using quadcopter for many applications such as: delivery service, monitoring, video transfer, military, reconnaissance mission, rescue and product conveyance for instance, delivering medical supplies to inaccessible places, film creation.

1.2 Purpose of Research

The Unmanned Aerial Vehicle (UAV) has been widely used in disaster rescue assignment and gives us a higher possibility to find more victims. In recent years, many research teams have proposed various programs on how to make the UAV be more smart, efficient, reliable. But there are still many limitations for UAV that should be improved. Such as the signal transmission will be affected by distance and the obstructions, and for the underground area where can't receive the signal from air, the success rate of finding survivors would be greatly affected.

The autopilot that is used in this paper is the open-source code software named ArduPilot. ArduPilot is a free software package and it contains all the software needed to monitor the drone, reading sensors inputs and control the drone according to the data received from the sensors. This software is available for programmers and researchers. It’s been developed over 5+ years by a team of various skilled engineers and computer scientists. It is the only autopilot computer code capable of controlling any vehicle system possible, from standard airplanes, multirotor, and helicopters, to boats and submarines.

The result of controlling the quadcopter in term of takeoff, fly to specific location, and landing was done successfully. The implementation on the real quadcopter is done successfully and it gives desired output.

1.3 Objectives

The chief objective of the project is to design, construct and demonstrate the viability of a quadcopter drone based on an RPi (RaspberryPi) microcomputer and APM 2.8 flight controller. In the on-board computing module, the software must handle the interface with the hardware that deals with the image and GPS acquisition, as well as with the data transmission system installed in the UAV, in order to acquire image and GPS coordinates and send them to the base station. Other information from sensor signals can also be sent over the data link. All the acquired data should be properly encoded and compressed before transmission. In the base station, the software must receive the information from the UAV and display the data. The GPS coordinates will be used to track the trajectory on a map, while displaying the acquired image.

In particular the main objectives of this project are:

1. Design and fabricate a full-sized quadcopter prototype.
2. Identifying principle targets and objectives.
3. Implementation of the interface software, between the on-board computer, the sensors, the GPS and the video camera.
4. Implementation of the communication between the on-board computer and the base station.
5. Implementation of the user interface, at the base station.

CHAPTER 2: PRINCIPLES OF QUADCOPTER OPERATION

2.1 Aerodynamics of Quadcopter

Quadcopter consists of four motors connected to the frame body at equal distances from each other. Two of motors rotates in clockwise, the other two motors rotate in counterclockwise in order to provide the specified thrust to lift the Quadcopter in the air.

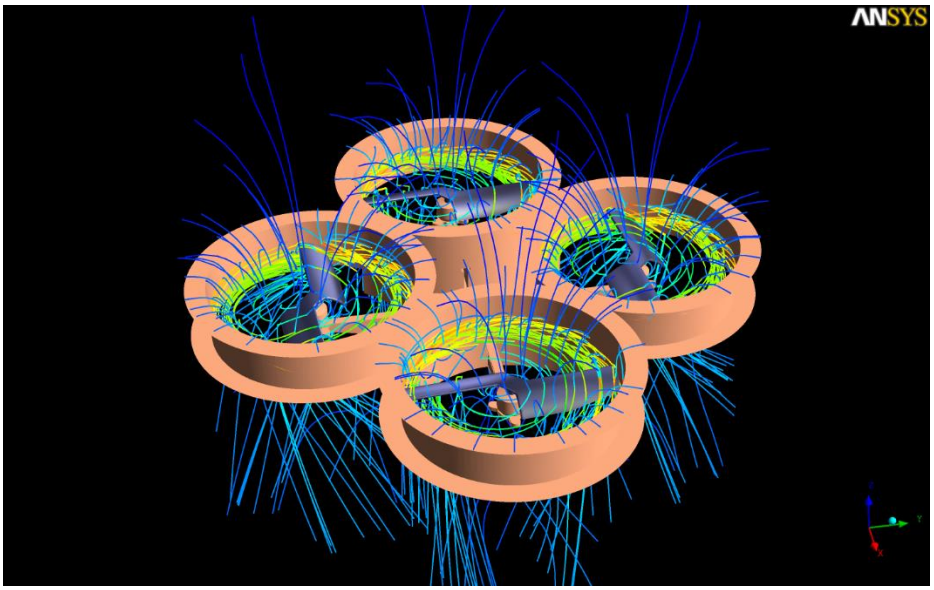

Figure 2.1: Aerodynamics of Quadcopter

So, basically a drone (specially quadcopters) has two pairs of propellers (two in a clockwise direction and another two in an anticlockwise direction). The speed of each motor is individually controlled to control the movement of the drone.

2.2 Working Principle

The quadcopter works over two principles:
(i) **Newton's Third Law of Motion**: In order to create a movement forward, the quad has to push the air backward. Thus, for every backward motion of air, there is a forward motion of quadcopter.
(ii) **Bernoulli's Principle**: According to Bernoulli's theorem there is an increase in velocity if there is a decrease in pressure and vice versa. Hence in order to increase the velocity the pressure at that point needs to be decreased. Bernoulli's principle helps explain that an aircraft can achieve lift because of the shape of its wings. They are shaped so that that air

flows faster over the top of the wing and slower underneath. Fast moving air equals low air pressure while slow moving air equals high air pressure. The high air pressure underneath the wings will therefore push the aircraft up through the lower air pressure.

2.3 Movements of the Quadcopter

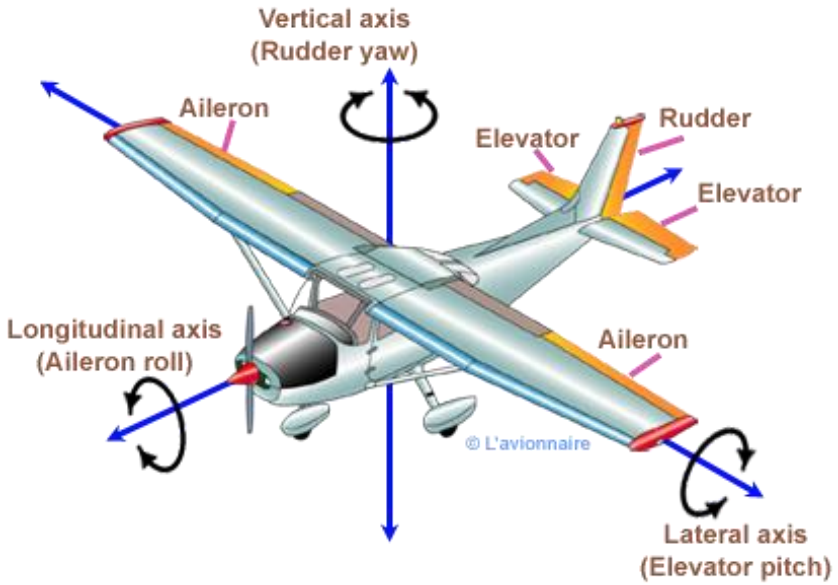

Figure 2.2: Movements of the Quadcopter

There are four basic movements of the quadcopter:

1. **Throttle:** Throttle is considered when there is an upward and downward motion of the quad. It is used for take-off and landing of the quad. It is created by rotating all 4 rotors at the same speed.
2. **Elevator (pitch):** Elevator is considered when there is a forward and backward motion of the quad. In an X-type quad forward motion is created by rotating the 2 rear rotors at greater speed compared to the front rotors whereas for backward motion the front rotors have higher speed than rear rotors.
3. **Aileron (roll):** Aileron is considered when there are right and left movements. It is used for changing directions in the moving drone. It is created by rotating the left 2 rotors to rotate at a higher speed than the right rotors for the right turn and vice versa for the left turn.
4. **Rudder (yaw):** Rudder action is used to create clockwise and anti-clockwise movement of the same spin to rotate at a greater speed than the other two.

2.4 Control System of the Quadcopter

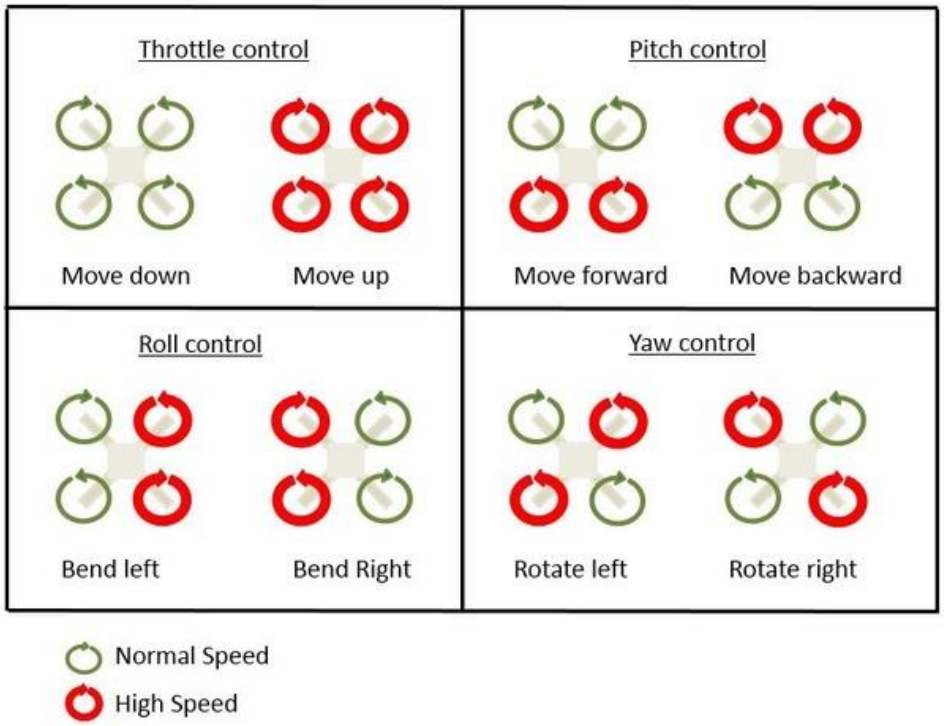

Figure 2.3: Control System of the Quadcopter

Torque and thrust plays an important role in flying drones. Well, a torque is nothing but a twisting force that tends to cause rotation. Alternatively, the capability of rotating an object around a fixed axis is known as torque. It is symbolized as τ (Tau). Mathematically, torque is the vector product of force (F) and the distance (r) of the axis. So, we can write:

$$\vec{\tau} = \vec{F} \times \vec{r} \qquad \text{so } \tau = Fr\sin\theta$$

Where θ is the angle between the force and the distance from the center of the axis. Thrust is simply pushing something suddenly or with propulsive force. Mathematically, thrust is the product of pressure (P) and area (A).

So, we can say, Thrust = P x A.

Small control board is used to control the drone. The control board has a few sensors that provide the necessary signals to move the propellers at the proper speed, and in the right direction. Inside the control board, there is a gyroscope and accelerometer that provide the orientation information of the drone. The RC receiver gets a signal from the RC transmitter and sends it to the microcontroller of the control board, and the ESCs connected to the microcontroller are then controlled to provide the necessary speed.

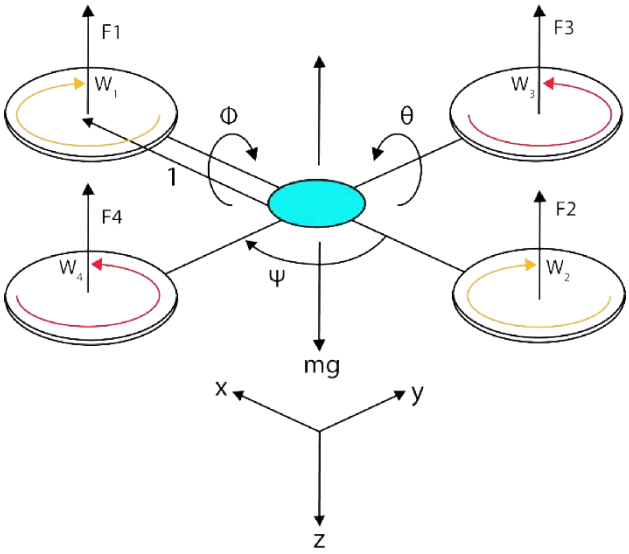

Figure 2.4: Force and Motion Diagram

Mathematically, thrust (*T*) will be proportional to the square of the angular velocity (*w*) of the propellers. The thrust is perpendicular to the *Z* direction of the drone. So, we can write:
Here, K_a is a constant. As the propellers rotate and create a thrust in the *Z* direction, there must be an opposite force in the drone. Let the movement be M_i, which will also be equal to the right side of the previous equation.

Therefore, $M_i = K_b \times \omega^2$, where K_b is a constant. We used two different constants because the force might be slightly decreased or increased, due to the friction of the air particle or the dust.

The opposite pair of propellers are M_x and M_y. According to the definition of movements, if the distance of the center of the drone and a propeller is *l*, we canwrite the following equations:

$$M_x = |F_1 - F_2| \times l$$
$$M_y = |F_3 - F_4| \times l$$
$$T \propto \omega^2$$
$$T = K_a \times \omega^2$$

The weight of the drone $W = mg$. The weight always acts in the direction of thequadcopter. From Newton's second law of motion,

$$force = mass \times acceleration\ (linear)$$

The torque can be defined with the help of inertia as follows:

2.4.1 Hovering

$$Torque = Inertia \times acceleration\ (angular)$$

To hover the quadcopter, the weight of the drone must be equal to all the upwardforces of the propellers, where the movement must be equal to zero:

$$mg = F_1 + F_2 + F_3 + F_4$$

For flying, the upward force must be greater than the weight. So, if we subtract the mass of the drone from the upward forces of the propellers, we will get the equation of motion of the drone flying. Let's say it is D_*. So,

$$D_* = F_1 + F_2 + F_3 + F_4 - mg$$

2.4.2 Rising or climbing or taking off

To fly the drone over the ground, the equations will be changed as follows:

$$mg < F_1 + F_2 + F_3 + F_4$$

$$D_* = F_1 + F_2 + F_3 + F_4 - mg > 0$$

2.4.3Dropping or descent or falling

The drone will not fly if the weight is more than the upward forces or $D_* < 0$:

$$mg > F_1 + F_2 + F_3 + F_4$$

$$D_* = F_1 + F_2 + F_3 + F_4 - mg < 0$$

2.5 Load Calculation

Table 1 below shows the most components which are used in the design of the prototype with its weight. The weight is too important as it is needed to calculate the thrust, power, and flight time.

Parameter	**Quantity**	**Weight (grams)**
Frame	1	135
BLDC motor	4	203.2
Flight controller	1	79.2
Electronic speed controller (ESC)	4	64
Ublox GPS Neo-M8N module	1	21
LiPo battery 3000 mAh	1	255.4
Raspberry PI 4 and other connectors and cables	1	133.3
Receiver	1	8.9
Total Weight		900

Table 1: Quadcopter Component Weight

The whole weight of the quadcopter was estimated in table 1. The Motors and propellers together produce the required thrust to lift the quadcopter in the air, the estimated AWU All Weight Up is 900 grams, the suggested payload to be used is 150 grams and the thrust requirements from the all motors must be at least double of 1050 grams. (900 + 150 = 1050 grams), the thrust value is 1050 × 2 = 2100 grams, therefore each motor must produce 525 g of thrust force. Motors are chosen depending on the value ofKV. It is calculated by the formula in equation 1:

$$RPM = KV\ rating \times Voltage\ input \quad (1)$$

Substituting the values of RPM and Voltage input:

KV rating = 24420/11.1 = 2200 Kv.

Propellers are kind of fan which convert the rotational motion into thrust. The propeller is specified on the basis of its pitch and diameter in inches.

$$Power\ (watts) = PC \times D^4 \times P \times RPM^3$$

Where:
PC: is the propellers constant (1.11 for APC propellers).
D: is the diameter of propellers in feet. (D=6 inch)
P: is the pitch of the propellers in feet. (P=3 inch)
Substituting the values of the prototype:

$$Power(watts) = 1.11 \times (0.5)4 \times (0.25) \times (24.42)3 = 252.57 \approx 253\ W \quad (2)$$

(The theoretical power is slightly high because of neglecting current losses, friction and other losses.)

From the table, Maximum thrust generated at the 239 W power for 1 motor is 732 grams. So, the maximum thrust of quadcopter is 2928 grams.

Empty mass of quadcopter = 900 gram + payload 150 = 1050 grams, then the quadcopter can fly safely with this payload. The result of the calculation shows that it would capable of flying with 150 grams of the load.

2.6 Flight Time Calculation

To estimate the flight time of the QC with payload, the actual flight test has been done with loads 150 grams. The flight time formula is described in equation 3:

$$\text{Time} = \text{Capacity} * \frac{\text{Discharge}}{\text{AAD}} \quad (3)$$

Capacity: is the capacity of the battery, and it's calculated in either in (mAh) milliampere hour or (Ah) Ampere hour.
Discharge is that the battery discharge that you simply allow throughout the flight. As LiPo batteries are often damaged if totally discharged, it is common practice never to discharge them by over 80%
AAD: The average ampere draw (AAD) of the QC, calculated in amperes.

Then the flying time is:

$$\text{Time} = \frac{3*0.8}{21.5}$$

$$= 6.7 \text{ minute}$$

CHAPTER 3: MECHANICAL DESIGN AND COMPONENTS

3.1 Frame Design

The quadcopter's most visible component is its frame, which must be strong enough to withstand the varying degrees of stress exerted on it during flight. At the same time, the frame must be lightweight in order to reduce the load on the motors and to increase flight time. The carbon fiber is mostly accepted material for frame designing because of its high stiffness to weight ratio. The Carbon Fiber Frames are very rigid and improve stability and flight performance.

Figure 3.1: Q250 Carbon Fiber Frame

The Q250 Quadcopter Super Strong Carbon Fiber frame is a mini-size quadcopter frame. It is manufactured with excellent quality Carbon Fiber for making it sturdier and stronger. This Frame has incredible strength and reduced weight. It is made from Carbon Fiber which makes it tough and durable. The carbon fiber arms have 4mm thickness, which ensure no more arm breakage at the motor mounts on a hard landing. The arms have dampers at the end, which protects motors during the crash.

The frame is lightweight and possesses thick arms, so as to give them good mechanical strength. The mini-size design of this quadcopter frame makes it aerodynamically efficient which will require lower throttle to hover. The design allows the whole take-off weight to be

controlled at about 1000-1200 grams. The frame has a wheelbase of 250mm and weighs around 160gm.

3.2 Brushless DC motor

BLDC motor works on the principle of Lorentz force law. The Lorentz force law states that whenever a current carrying conductor is placed in a magnetic field, it experiences a force. As a consequence of reaction force, the magnet will experience an equal and opposite force. In the BLDC motor, the current carrying conductor is stationary and the permanent magnet is moving.

When the stator coils get a supply from source, it becomes electromagnet and starts producing the uniform field in the air gap. Though the source of supply is DC, switching makes to generate an AC voltage waveform with trapezoidal shape. Due to the force of interaction between electromagnet stator and permanent magnet rotor, the rotor continues to rotate.

With the switching of windings as High and Low signals, corresponding windings are energized as North and South poles. The permanent magnet rotor with North and South poles aligns with stator poles which causes the motor to rotate.

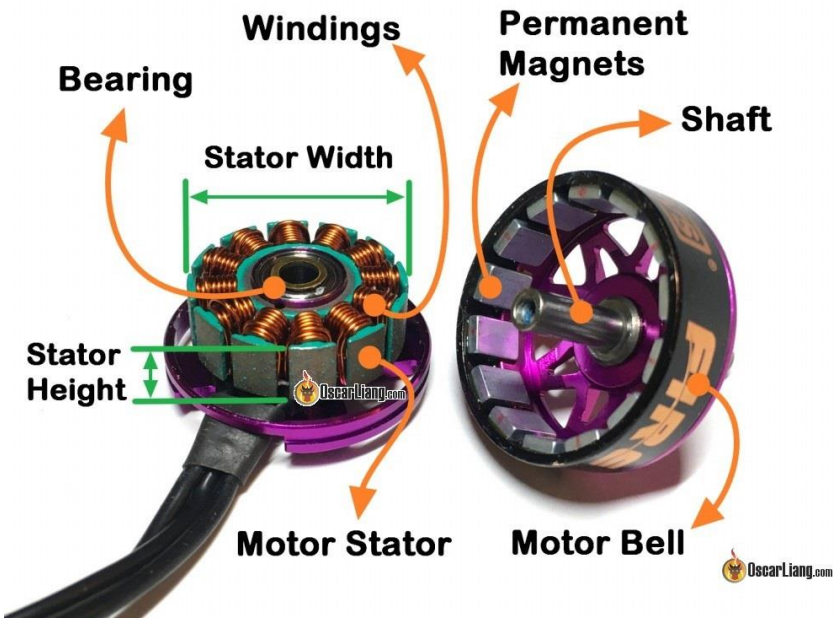

Figure 3.2: Brushless DC motor

In this motor, the permanent magnets attach to the rotor. The current-carrying conductors or armature windings are located on the stator. They use electrical commutation to convert electrical energy into mechanical energy.

The main design difference between a brushed and brushless motors is the replacement of mechanical commutator with an electric switch circuit. A BLDC Motor is a type of synchronous motor in the sense that the magnetic field generated by the stator and the rotor revolve at the same frequency.

Brushless motors do not have any current carrying commutators. The field inside a brushless motor is switched through an amplifier which is triggered by the commutating device like an optical encoder.

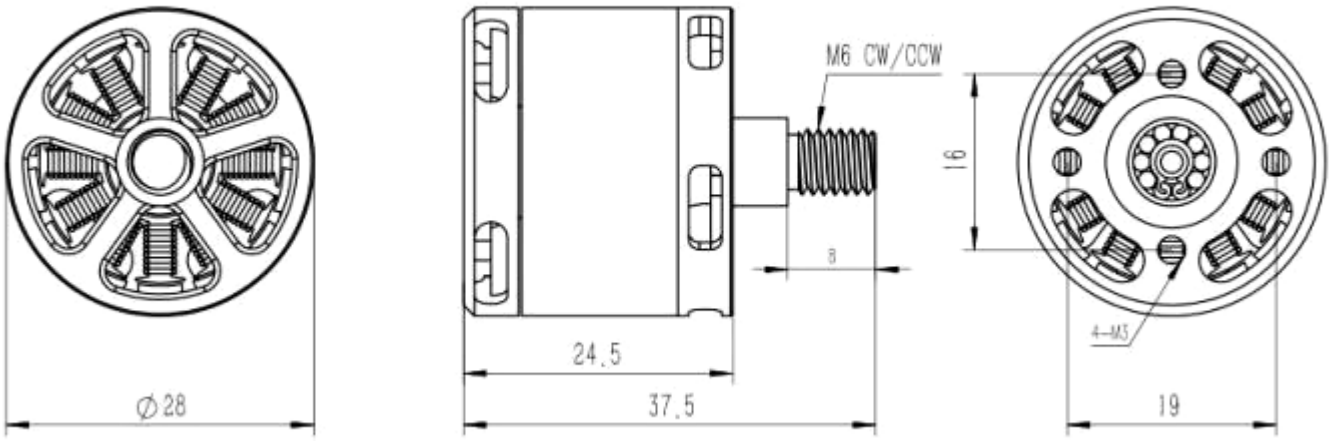

Figure 3.3: Dimensions of A2212 2200 KV BLDC

Brushless DC motors have higher speed range and lower electric noise generation. It has high efficiency and high output power to size ratio due to the use of permanent magnet rotor and high speed of operation even in loaded and unloaded conditions due to the absence of brushes that limits the speed.

But there are some limitations like the electronic speed controller is required to operate. Moreover, İt has harmonic content in back EMF so some torque ripple will occur at motor torque. İt is expensive because of the permanent magnets. There is a temperature limit on the rotor because of the magnets. Demagnetization possibility also limits the input current of BLDC.

3.2.1 Theoretical Performance

The performance of a brushless DC-motor is nearly identical to a brushed DC motor. The major difference lies in the BLDC's commutator being integrated into the speed controller,

MODEL	KV (rpm/V)	Voltage (V)	Prop	Load Current(A)	Power (W)	Pull (g)	Efficiency (g/W)	Lipo Cell	Weight (g)
A2212	930	11.1	1060	9.8	109	660	6.1	2-4S	52
	1000		1047	15.6	173	885	5.1		
	1400		9050	19.0	210	910	4.3		
	1800		8060	20.8	231	805	3.5		
	2200		6030	21.5	239	732	3.1	2-3S	
	2450		6030	25.2	280	815	2.9		

Table 2: Theoretical Performance of BLDC Motor

while a brushed DC motor's commutator are located directly inside the motor. Kv rating is the relationship between the RPM and the voltage.

For the motor chosen in the present project:

Kv = 2200 rpm/volts

Propeller dimension = 6030

Power required to lift 1 kg = 322.58 Watt

Efficiency = 3.1 gram/Watt

3.2.2 Specifications

- Max Efficiency: 80%
- Internal Resistance: 90mΩ
- Load Current: 21.5A
- Weight: 49g
- Dimensions: 27.5x30 (mm)

3.3 Propeller

The propeller works by displacing the air pulling it behind itself (the action), this movement of air then results in the aircraft being pushed forward from the resulting pressure difference (the opposite reaction). The more air that is pulled behind the propeller, the more thrust or forward propulsion is generated.

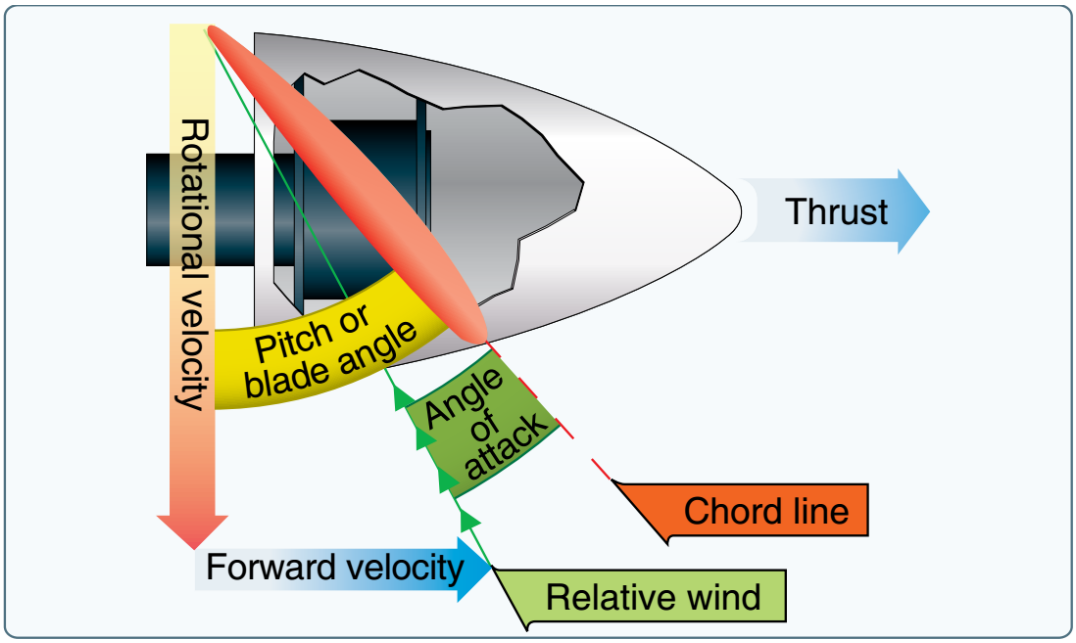

Figure 3.4: Working Principles of Propeller

- Chord Line: The chord line of a propeller is an imaginary line drawn through the center of the blade from its leading edge (at the hub) to its trailing edge (tip).

- Pitch: The blades of a propeller are not straight; they are on an angle similar to that of a screw. The pitch is effectively a measure of how far the propeller would move forwards in one revolution. The pitch is used to control the speed of the air leaving the back of the propeller. The pitch of a propeller blade changes as you move along its surface from one end to the other. It is steepest or shortest at the central hub and shallowest at the outer tip.

- Blade Angle: This is the angle between the chord line and the plane of rotation and is measured (in degrees) at a specific point along the length of the blade.

- Angle Of Attack: This is defined as the angle at which the air strikes the propeller blade. In simple terms the angle of attack can be described as the difference

between where a wing is pointing and where it is going. Increasing the angle of attack results in an increase in both lift and induced drag, up to the point of a stall. The twist of a propeller blade is used to maintain a more constant angle of attack along the length of the blade to counteract the differences in blade speed at the hub and the tip of the propeller.

For the propeller chosen in the present project:

Length = 6 inch

Pitch = 4.5 inch

Figure 3.5: 6045 Carbon Nylon Propellers

The Orange HD Propellers 6045(6X4.5) Carbon Nylon Props are the high-quality propellers specially designed for multi-copters. The pitch indicates that with each rotation the propeller moves 4.5 inches in a translative direction. These Propellers are light in weight and high strength having 15° angle design at the end of the propeller to avoid whirlpool while the quadcopter is flying. The Carbon Nylon Props are of high endurance and flexibility for great impact. It helps to improve the air-powered efficiency and aero foil stability.

Propeller Aerodynamics

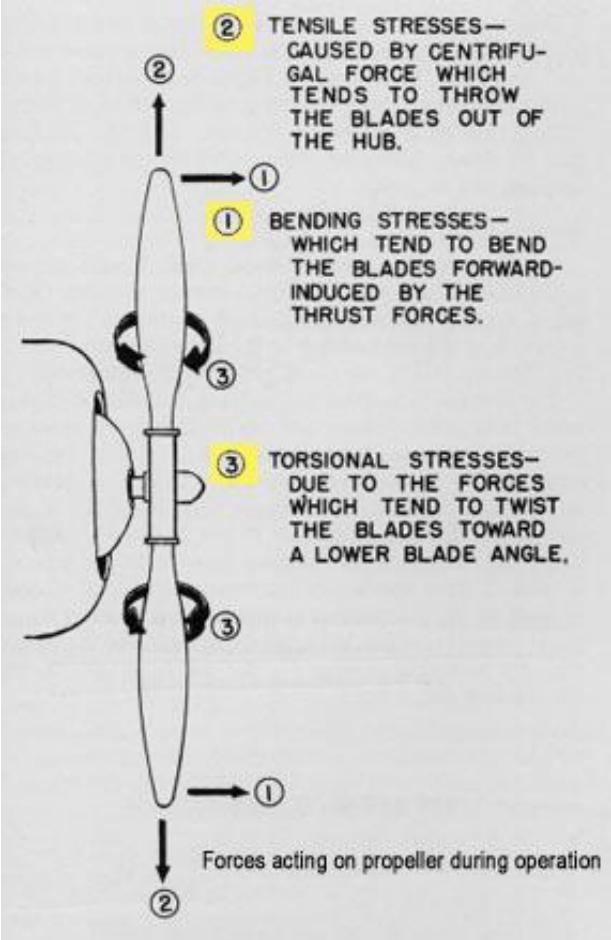

Figure 3.6: Forces acting on the Propeller during Operation

Three main stresses act upon the propellers while the quadcopter is in flight. These are bending stress, tensile stress and torsion stress. Bending stress is caused by the thrust force, which tends to bend the blade forward as the quadcopter moves through the air. Tensile stress is caused by centrifugal force, which for the purposes of the present study is insignificant. Torsion stress is produced in the rotating blades by their dual twisting motions, one of which is the stress resulting from the action of the blade and is referred to as the aerodynamic twisting moment.

3.4 Electronic Speed Controller

An electronic speed controller (ESC) is an electronic circuit that controls and regulates the speed of an electric motor. It may also provide reversing of the motor and dynamic braking. Miniature electronic speed controls are used in electrically powered radio-controlled models.

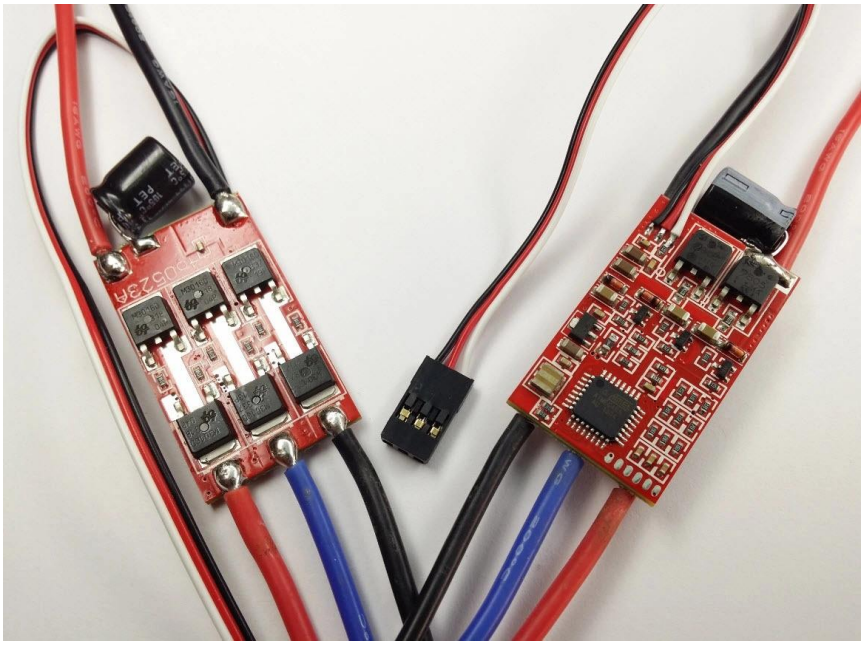

Figure 3.7: SimonK 30A Electronic Speed Controller

SimonK 30A BLDC ESC Electronic Speed Controller is specifically made for quadcopters and multi-rotors, which provides faster and better motor speed control giving better flight performance compared to other available ESCs. It can drive motors which consume up to 30A. It works on 2S-3S LiPo batteries. This electronic speed controller offers a battery eliminator circuit (BEC) that provides 5V and 2A to the receiver so we don't need an extra receiver battery. It also includes backwards-polarity protection and protection on the 5V receiver line, this means that if you accidentally attach a battery backward it won't destroy your motor controller and other BECs won't affect the ESC.

Connection type	Wire Colour	Function
Power	Red	7.4 to 14.8V
	Black	Ground
BLDC Motor Connections	Three Blue Wires	BLDC ESC connections
Servo Connector	White	Throttle Input
	Red	5V, 2Amp Out
	Black	Ground

Table 3: Pinout Of ESC

3.5 Flight Controller

A flight controller (FC) is a small circuit board of varying complexity. Its function is to direct the RPM of each motor in response to input. A command from the pilot for the multi-rotor to move forward is fed into the flight controller, which determines how to manipulate the motors accordingly.

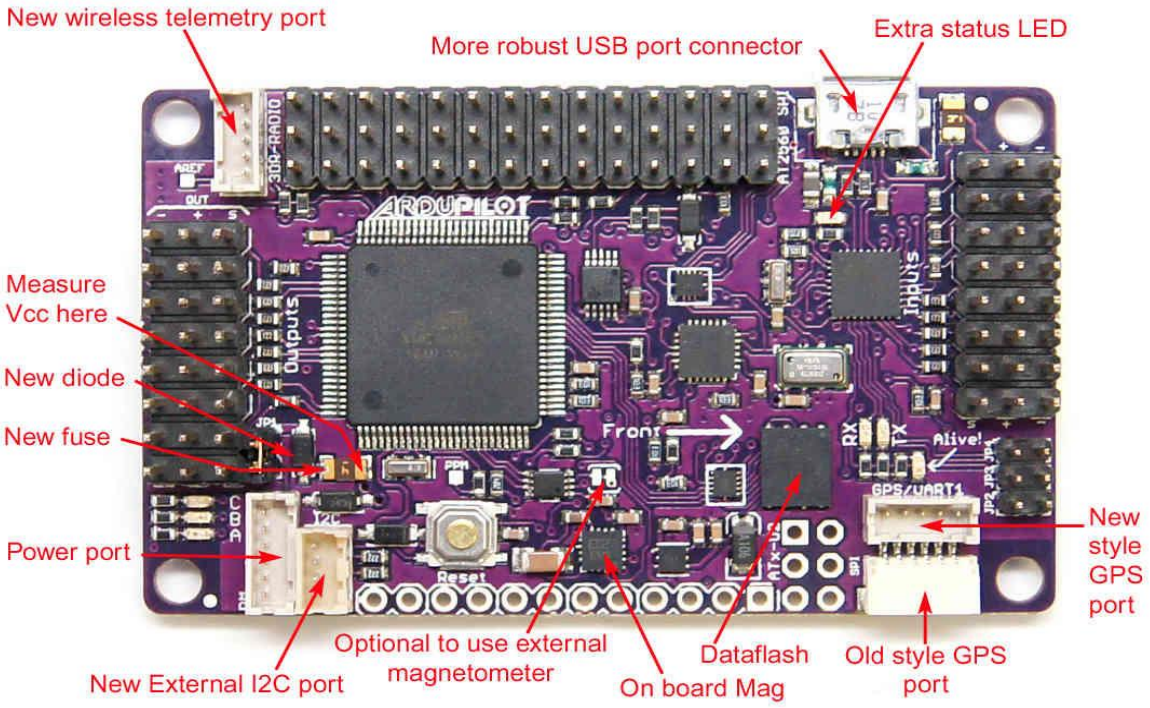

Figure 3.8: APM 2.8 Flight Controller Pinout

The majority of flight controllers also employ sensors to supplement their calculations. These range from simple gyroscopes for orientation to barometers for automatically holding altitudes. GPS can also be used for auto-pilot or fail-safe purposes.

APM 2.8 Multicopter Flight Controller is an upgraded version with a Built-in Compass for FPV RC Drone. The module has the option to use the built-in compass and external compass via a jumper. This makes the APM 2.8 ideal for use with multi-copters and rovers.

The APM 2.8 Multicopter Flight Controller is a complete open-source autopilot system. It allows the user to turn any fixed, rotary-wing. In addition, it turns a multirotor vehicle (even cars and boats) into a fully autonomous vehicle. Meanwhile, it is capable of performing programmed GPS missions with waypoints.

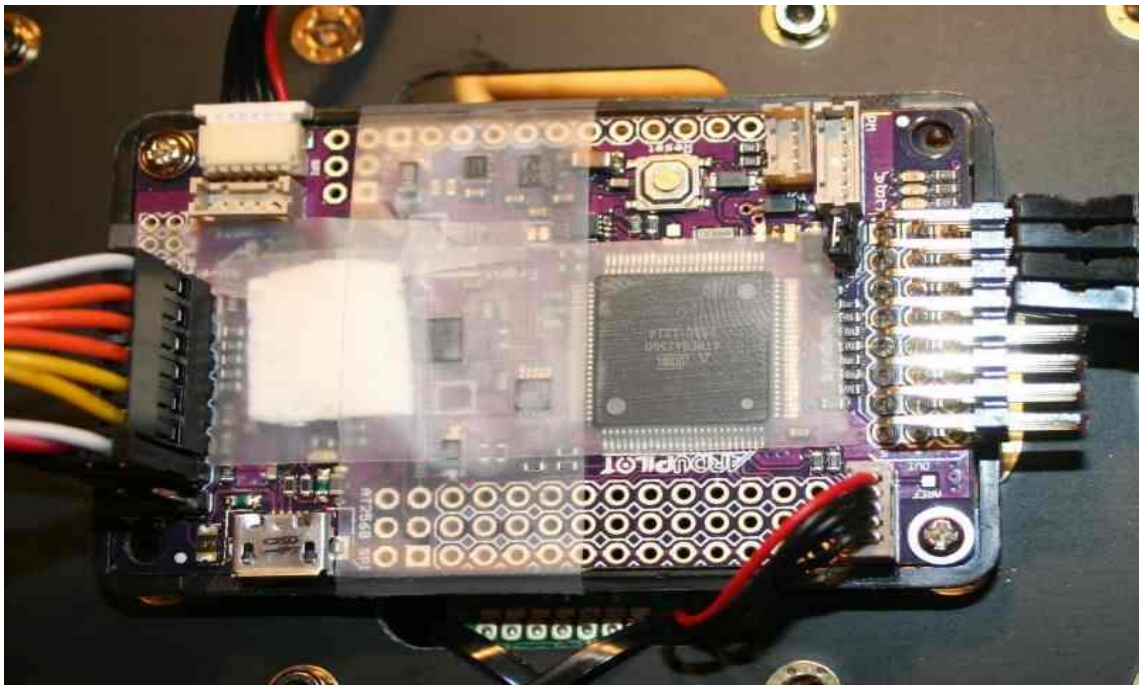

Figure 3.9: APM 2.8 Flight Controller

ArduPilot provides fully autonomous, semi-autonomous and fully manual flight modes, programmable missions with 3D waypoints, optional geofencing. It can simulate with a variety of simulators, including ArduPilot SITL. Large number of navigation sensors supported, including several models of RTK GPSs, traditional L1 GPSs, barometers, magnetometers, laser and sonar rangefinders, optical flow, ADS-B transponder, infrared, airspeed, sensors, and computer vision/motion capture devices. Sensors can communicate via SPI, I^2C, CAN Bus, Serial communication, SMBus. It also provides failsafe for loss of radio contact, GPS and breaching a predefined boundary, minimum battery power level. Support

for navigation in GPS denied environments, with vision-based positioning, optical flow, SLAM, Ultra-Wideband positioning. APM 2.8 can operate Integration and communication with powerful secondary, or "companion", computers Raspberry Pi.

Moreover, it has onboard 4 MegaByte Data flash chips for automatic data logging, barometric pressure sensor upgraded to MS5611-01BA03, from Measurement Specialties, optional off-board GPS, a uBlox LEA-6H module with Compass. It is one of the first open-source autopilot systems to use Invensense's 6 DoF Accelerometer/Gyro MPU-6000.

3.6 Global Positioning System

Figure 3.10: Ready to Sky GPS Module

The Global Positioning System (GPS) is a satellite navigation system that provides positioning, navigation and time information (PNT) anywhere on Earth, provided when there is an unobstructed line of sight to four or more GPS satellites. The timing service is implemented by incorporating in each GPS satellite a high accuracy atomic clock. The satellites permanently broadcast their own time to the receiver, so they can synchronize themselves. Besides the information about the time of each satellite, the satellites also broadcast their current position. With the information about the time the message was sent and the speed (speed of light), it is possible for the GPS module to calculate the distance

between him and the satellites. By knowing the position of the satellites, which is sent in the message and by calculating the distance between the GPS module and the satellite, it is possible for the GPS module to calculate his own position.

3.7 Transmitter and Receiver

Radio control consists of two major components, the transmitter (Tx) and receiver (Rx). The receiver is of the PWM type, indicating that it uses separate servo wires for each channel. Since there are a total of six channels in this control system, the receiver has six separate input wires. The minimum number of channels for controlling the quadcopter is four, as only pitch, roll, yaw and throttle need to be controlled to maintain flight. The first three primary flight control channels are designated as Elevator, Aileron and Rudder. By adding more channels, it is possible to control such auxiliary features as flight mode, camera trigger, camera gimbal control, landing gear and some others.

For a quadcopter, as for any UAV, its radio-operated remote-control system constitutes the single means of directing the vehicle's movements. A radio transmitter sends various signals, which are then modulated into the proper channels. The electronic receptors in the quadcopter receive these radio signals and demodulate them so that each signal controls a different task. For example, one such signal from the radio transmitter directs the movement of the throttle 34 analog stick. The more pressure applied to this lever, the greater is the value of throttle being modulated and transmitted directly to the quadcopter. When the vehicle receives the signal, all values are demodulated and the channels governing the throttle increase the motor speed to an appropriate degree.

For the present project, the transmitter frequency is 2.6 GHz, which is the most common frequency used for small, remote-controlled ground and air vehicles. The antenna is small enough to be easily portable. However, the corresponding range is shorter than that possible with the 27/72 MHz frequencies. For this particular transmitter the maximum range is approximately 500 meters. Such a range is adequate for the purpose since the quadcopter will be used mainly in autonomous flight mode.

Figure 3.11: FlySky CT6B Transmitter

FlySky CT6B 2.4Ghz 6CH Transmitter with FS-R6B Receiver is the popular 6 Channel Radio CT6B manufactured by FlySky. FlySky CT6B 2.4 GHZ 6CH transmitter is an entry-level 2.4 GHz radio system offering the reliability of 2.4 GHz signal technology and a receiver with 6 channels. CT6B 2.4 GHZ 6CH transmitter radio is a value for money, entry-level 6 channel transmitter, ideal for quadcopters and multicopters that require the 6ch operation.

This radio has two retract switches and proportional flap dials in easy reach for channels 5 and 6. It can be powered by 8 x AA Size Batteries or a 12V Power Supply. This comes with a trainer port to help beginners learn flying. It can be configured by connecting it to the computer. Use the T6config software to configure your radio on a computer.

FlySky CT6B Features:

1. Super active and passive anti-jamming capabilities.
2. Very low power consumption.
3. High receiving sensitivity.
4. 8 model memory, digital control.
5. We can program by PC with the included software.
6. Full range 2.4GHz 6-channel radio.
7. 4 Types (Airplane, Heli90, Heli120, Heli140).
8. Use a linear spread of fine paragraphs with an excess antenna.

3.8 Raspberry Pi (Companion Computer)

Figure 3.12: Raspberry Pi (Companion Computer)

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins, allowing you to control electronic components for physical computing and explore the Internet of Things (IoT).

Raspberry Pi is one of the most used minicomputers of the market. One of the strengths of Raspberry Pi are the connectivity options, allowing HDMI out, RCA out, audio out, the connection of two USB devices, Ethernet port, GPIO pins, and an SD card. The GPIO pins and the two USB ports of model B are an important feature for this project, since they can be used to connect the sensors modules.

The single board computer is the main component of the mobile station that will implement many of the software modules of this project. This single board computer interfaces with the video camera, the GPS receiver and the remaining UAV sensors, and combines all of the information into a protocol suitable for streaming via the communication system.

3.9 Li Po Battery

Figure 3.13: 3000mAh Li po battery

The battery is also a major component in the project, for only the battery's power supply determines the time that the quadcopter can remain airborne. Also of consequence is the weight factor as the battery is the heaviest component mounted in the frame. A lithium polymer (LiPo) battery is selected for the project due to its advantageous power-to-weight ratio and its ability to deliver sufficient power output. The battery that powers the motor is a three-cell LiPo battery. The battery yields 11.1 volts and 3000 mAh to drive the motors. It can feed 120 amperes continually and peak at 240 amperes. It has three LiPo cells in serial connection. Each cell is 3.7 volts, producing a total of 11.1 volts. This particular battery discharges with a 40 C rating, which designates the amount of continuous current that can be drawn: 3000mAh * 40C = 120A.

3.10 Power-distribution board (PDB)

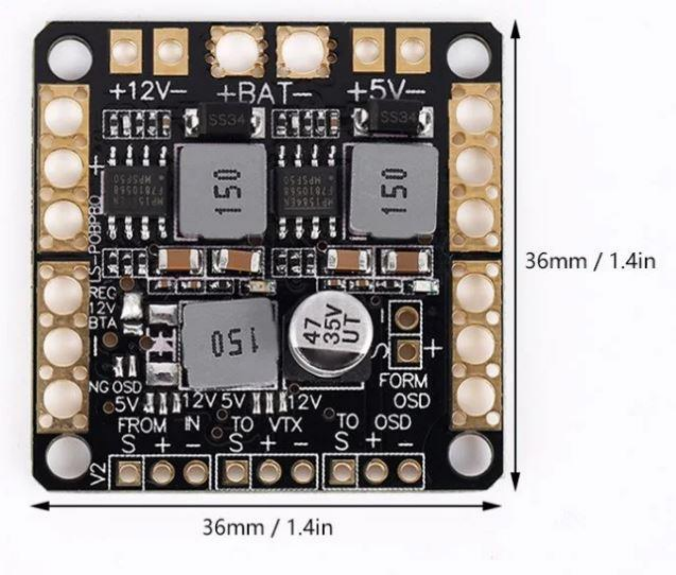

Figure 3.14: Power Distribution Board

Power Distribution Board (PDB) is used to distribute the power from your flight battery to all different components of the multirotor. PDB's are one of the simplest components on a Multirotor and therefore are one of the easiest to choose. There are 3 main points that should be considered when using a PDB for multirotor:

1. Size and layout
2. Voltages and Current Capability (on board voltage regulators)
3. Additional features

It is an equally important part of the project, as the RPi and APM 2.8 also need power. They both require 4.5 - 5.5 volts, which is well within the capacity of a regular three-cell LiPo battery. To achieve the lower voltage, power supply from PDB is separately provided to RPI.

CHAPTER 4: ASSEMBLY OF HARDWARE

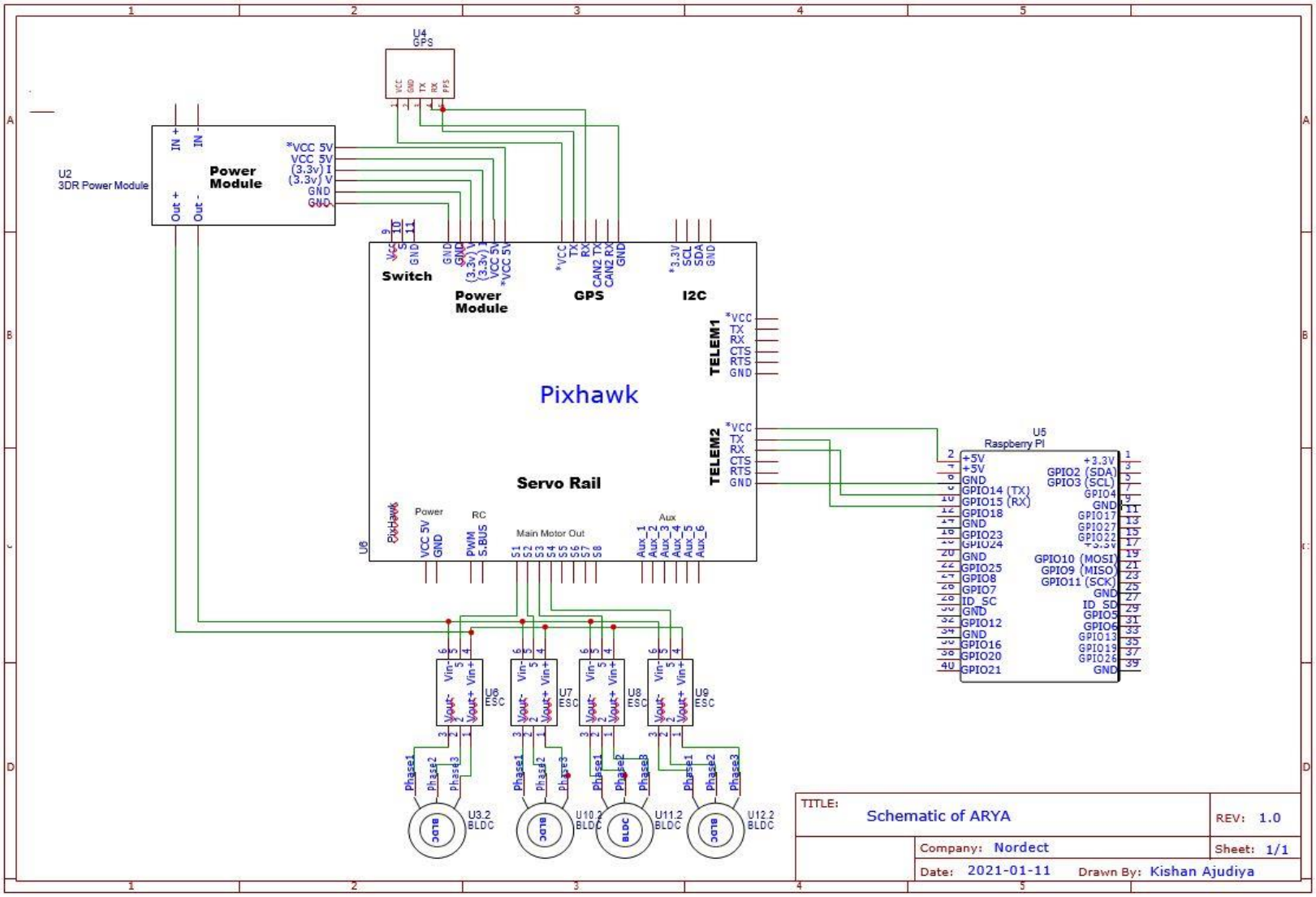

Figure 4.1: Pinout of hardware assembly

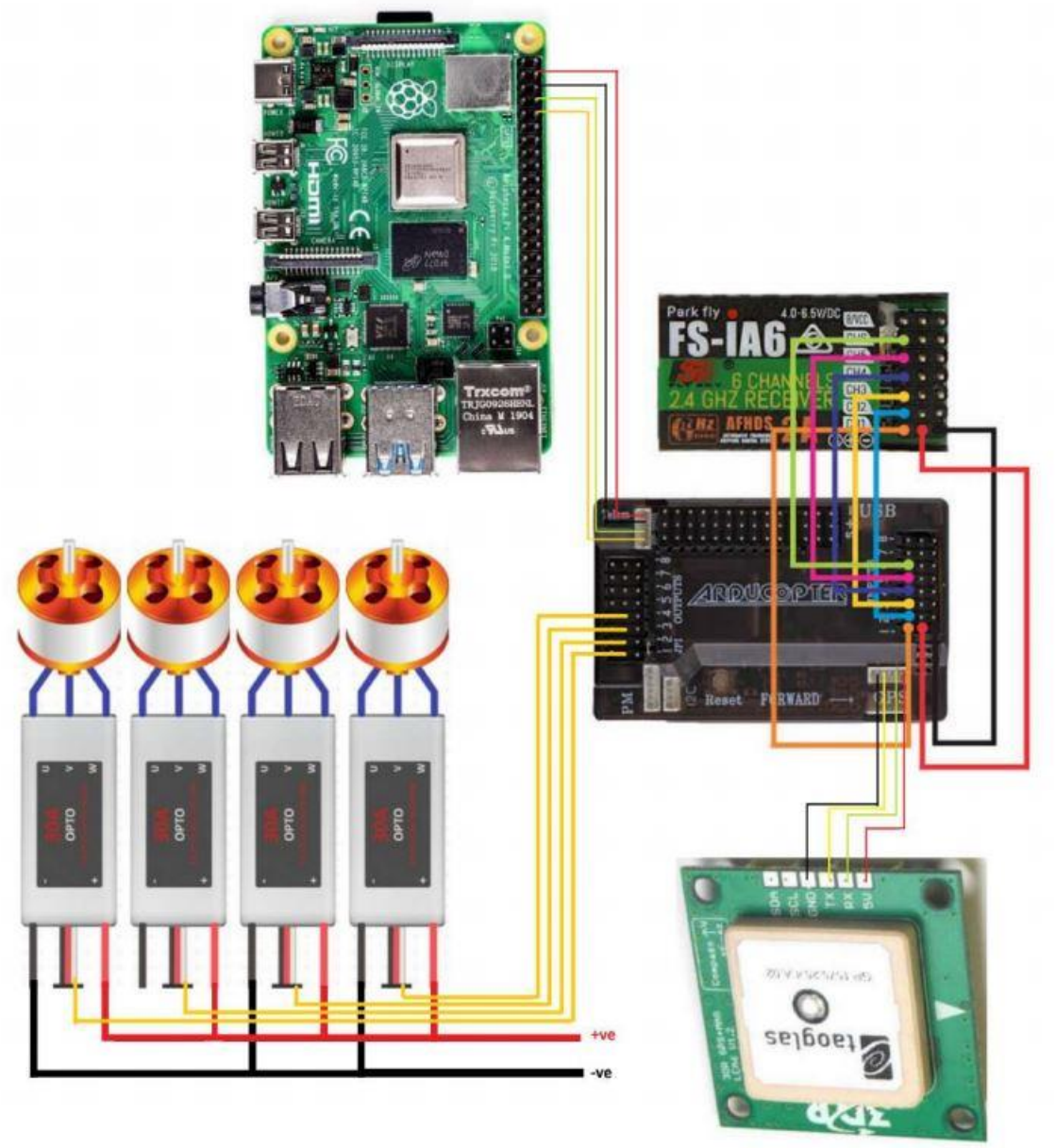

Figure 4.2: Hardware Assembly Circuit Diagram

CHAPTER 5: INSTALLATION AND SET-UP OF GROUND CONTROL STATION(GCS)

ArduPilot can be used with many different ground stations. Some of the more popular GCS systems are provided below:

- Mission Planner (Windows, Linux, Android): Install Mission Planner This GCS is the most compatible and closely tracks new features and updates in ArduPilot. It functions best in the Windows environment.
- QGroundControl (Windows, Mac OS X, Linux, Android and iOS). This is the most popular GCS for Android based systems and provides the best presentation on small format devices like cell phones. Does not have all the capabilities of Mission Planner, but has a more user-friendly interface for some functions. Operates well on all platforms.
- APM Planner 2 (Windows, Mac OS X, Linux): APM Planner 2 -Provides basic functionality, but is not currently well maintained and updated.

5.1 Mission Planner Overview

Mission Planner is a full-featured ground station application for the ArduPilot open-source autopilot project.
Mission Planner is a ground control station for Plane, Copter and Rover. It is compatible with Windows only. Mission Planner can be used as a configuration utility or as a dynamic control supplement for your autonomous vehicle.

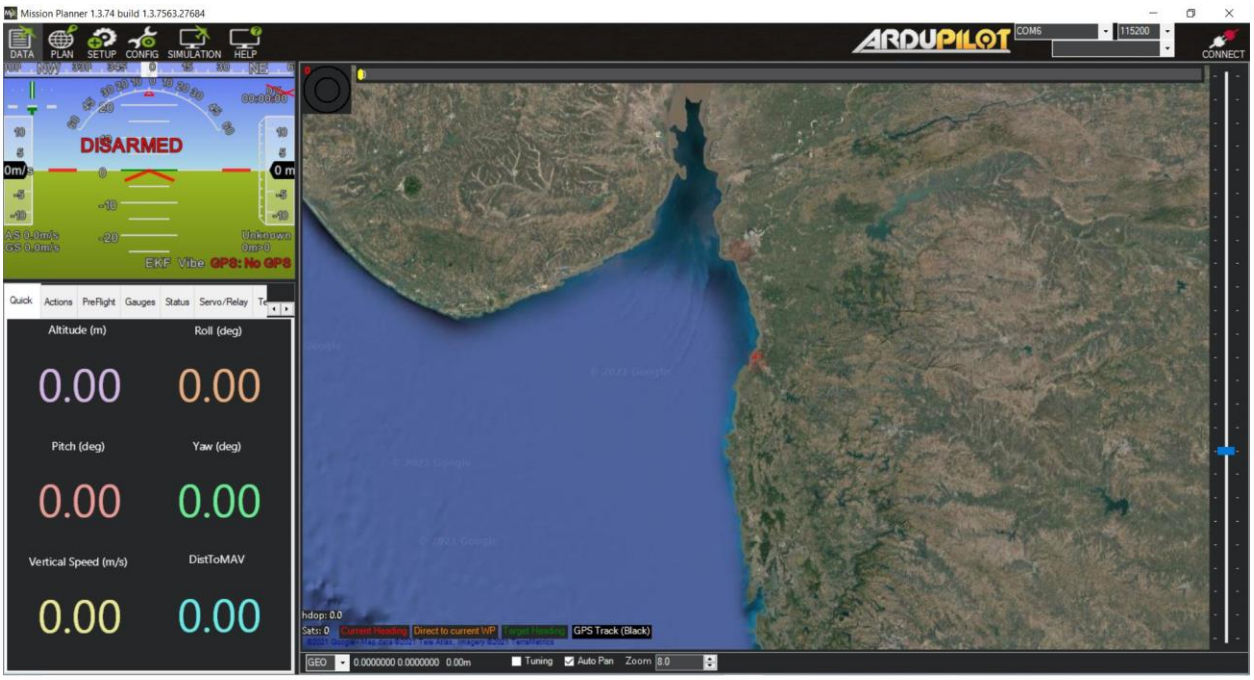

Figure 5.1: Mission Planner

Here are just a few things you can do with Mission Planner:

- Load the firmware (the software) into the autopilot board (i.e., Pixhawk series) that controls your vehicle.
- Setup, configure, and tune your vehicle for optimum performance.
- Plan, save and load autonomous missions into you autopilot with simple point-and-click way-point entry on Google or other maps.
- Download and analyze mission logs created by your autopilot.
- Interface with a PC flight simulator to create a full hardware-in-the-loop UAV simulator.
- With appropriate telemetry hardware you can:
    - Monitor your vehicle's status while in operation.
    - Record telemetry logs which contain much more information the on-board autopilot logs.
    - View and analyze the telemetry logs.
    - Operate your vehicle in FPV (first person view).

5.2 Installing Mission Planner (Windows)

The below instructions show how to install *Mission Planner* on Windows.

- Download the latest Mission Planner installer.
- Double click on the downloaded **.msi** file to run the installer.

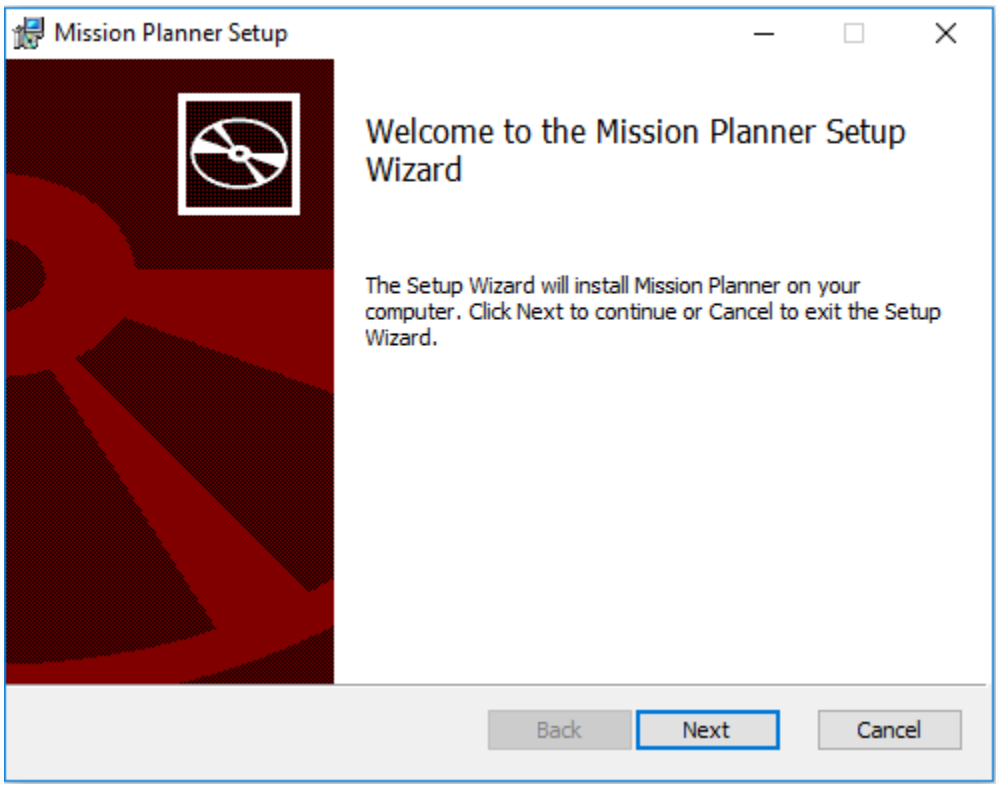

Figure 5.2: Mission Planner Setup

- Follow the instructions to complete the setup process. The installation utility will automatically install any necessary software drivers.
- If you receive the warning pictured below, select **Install this driver software anyway** to continue.

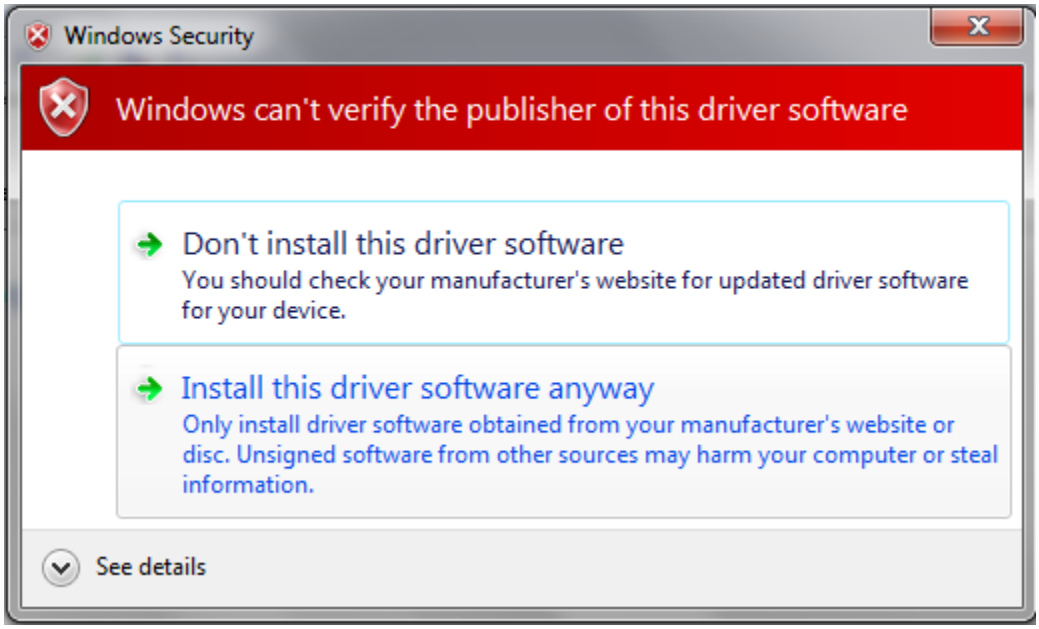

Figure 5.3: Windows Security Warning

Mission Planner is normally installed in the **C:\Program Files (x86)\Mission Planner** folder.

An icon to open the *Mission Planner* is created according to your instructions during the installation. Once installation is complete, open *Mission Planner* by clicking on its system icon.
Then you can either:

- Connect Mission Planner to AutoPilot in order to receive telemetry and control the vehicle OR
- Load Firmware OR plan autonomous missions .

5.3 Loading Firmware

Once you've installed a ground station on your computer, connect the autopilot using the micro-USB cable as shown below. Use a direct USB port on your computer (not a USB hub).

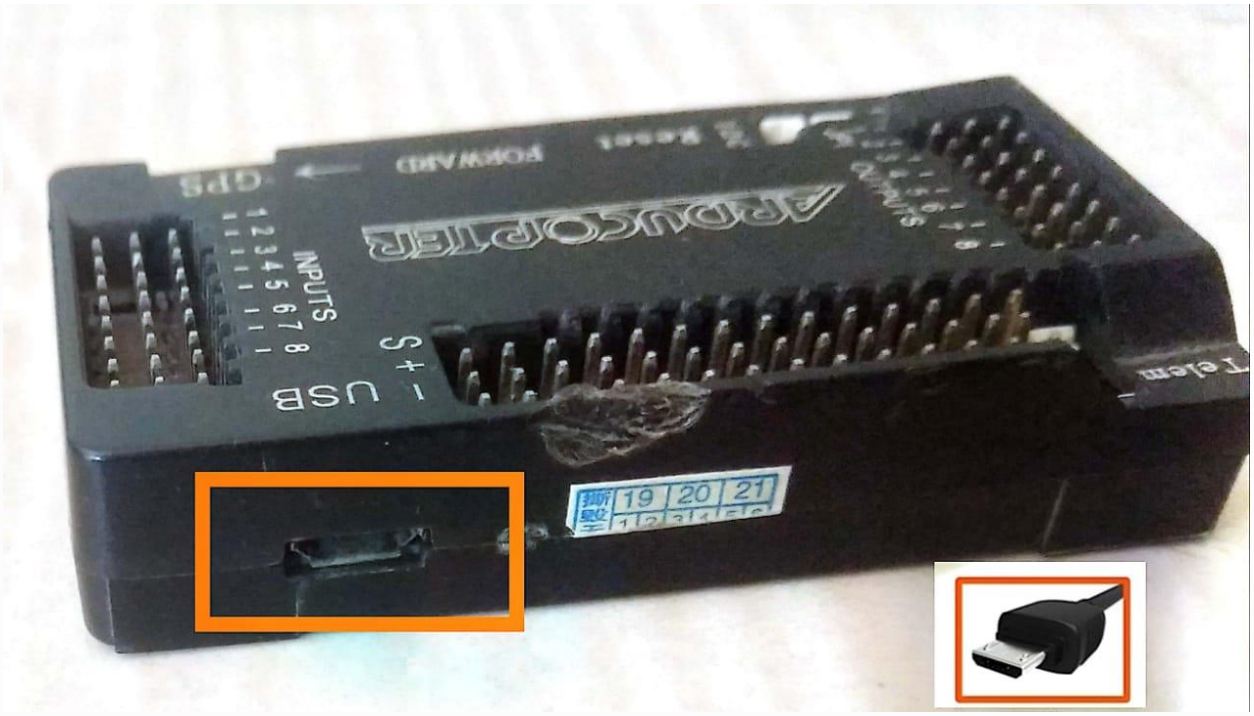

Figure 5.4: APM 2.8 USB connection

Windows should automatically detect and install the correct driver software.

Select the COM port

If using the *Mission Planner,* select the COM port drop-down on the upper-right corner of the screen (near the **Connect** button). Select **AUTO** or the specific port for your board. Set the Baud rate to **115200** as shown.

Figure 5.5: COM Port drop-down menu

5.4 Install firmware

On the Mission Planner's **SETUP | Install Firmware** screen, select the appropriate icon that matches your frame (i.e., Quad, Hexa). Answer **Yes** when it asks you "Are you sure?".

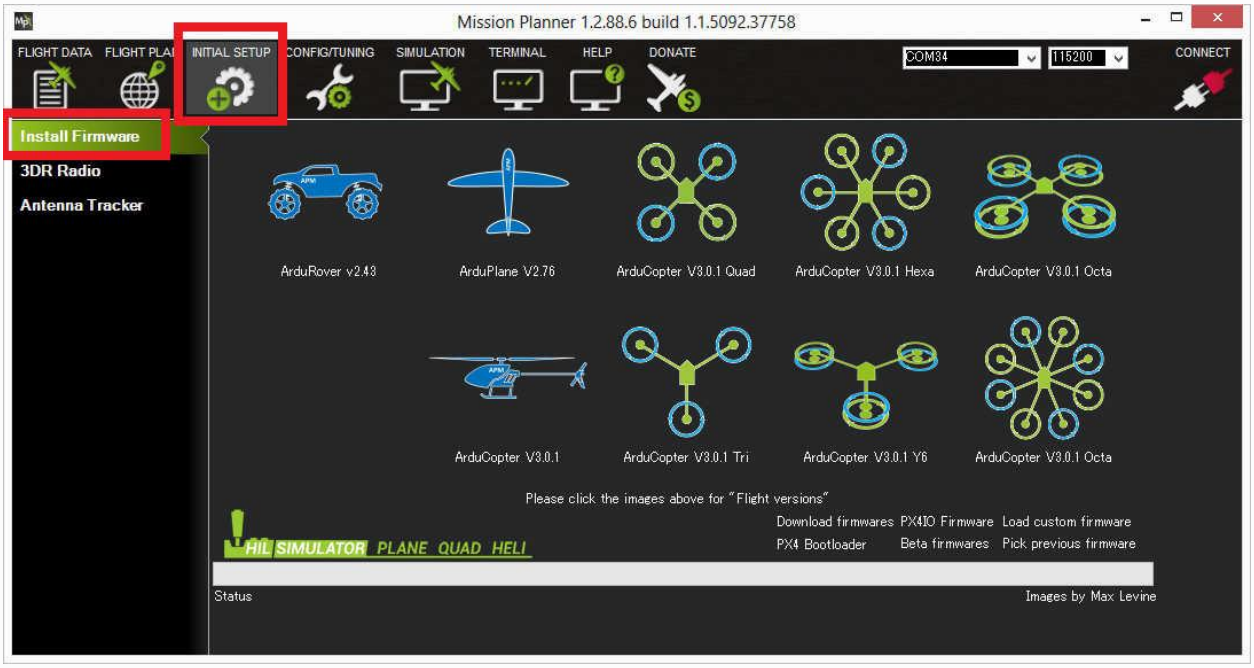

Figure 5.6: Mission Planner: Install Firmware Screen

Next it will try to detect which board you are using and it may ask you to unplug the board, press OK, and plug it back in to detect the board type.

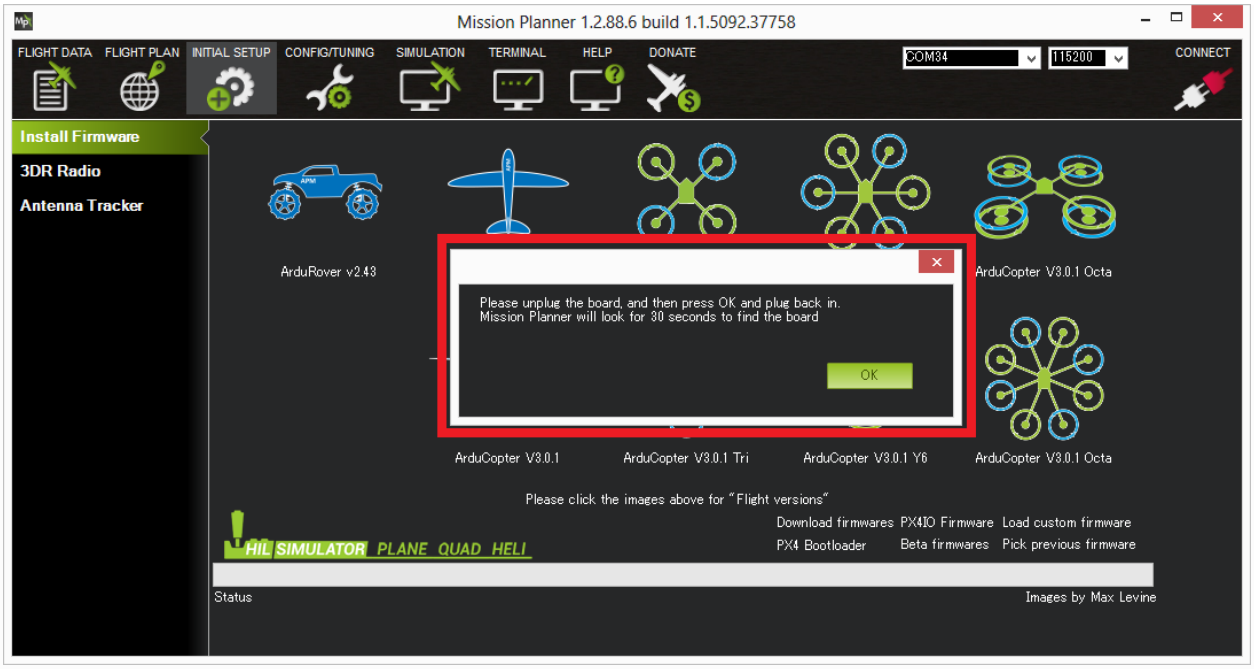

Figure 5.7: Mission Planner: Install Firmware Prompt

If all goes well you will see some status appear on the bottom right including the words, "erase…", "program…", "verify..." and "Upload Done". The firmware has been successfully uploaded to the board.

It usually takes a few seconds for the bootloader to exit and enter the main code after programming or a power-up. Wait to press CONNECT until this occurs.

Testing

You can test the firmware is basically working by switching to the *Mission Planner Flight Data* screen and pressing the **Connect** button. The HUD should update as you tilt the board.

Connect Mission Planner to AutoPilot has more information on connecting to Mission Planner.

5.5 Connect Mission Planner to AutoPilot

To establish a connection, you must first choose the communication method/channel you want to use, and then set up the physical hardware and Windows device drivers. You can connect the PC and autopilot using USB cables, Telemetry Radios, Bluetooth, IP connections etc.

On *Mission Planner*, the connection and data rate are set up using the drop-down boxes in the upper right portion of the screen.

Figure 5.8: COM Port

Once you've attached the USB or Telemetry Radio, Windows will automatically assign your autopilot a COM port number, and that will show in the drop-down menu (the actual number does not matter). The appropriate data rate for the connection is also set (typically the USB connection data rate is 115200 and the radio connection rate is 57600).

Select the desired port and data rate and then press the **Connect** button to connect to the autopilot. After connecting **Mission Planner** will download parameters from the autopilot and the button will change to **Disconnect** as shown:

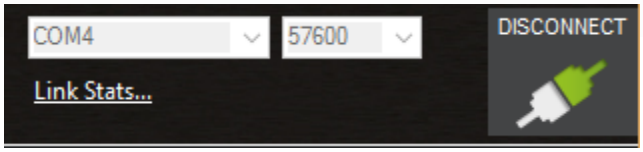

Figure 5.9: COM Port

Troubleshooting

If Mission Planner is unable to connect:

- Check that the correct baud rate is used for the selected method (115200 on USB or 57600 on Radio/Telemetry)
- If attaching via USB, be sure that a few seconds after power up have passed before attempting to connect. If you attempted to connect during the bootloader initialization time, Windows may get the wrong USB information. Connection attempts after this may require that the USB connection be unplugged and re-plugged, then wait for bootloader to enter the main code (few seconds), then attempt the connection. Occasionally, MP must be restarted if an attempt to connect is made while in the bootloader initialization period.
- If using a COM port on Windows, check that the connection's COM port exists in the Windows Device Manager's list of serial ports.
- If using a USB port, try a different physical USB port

- If using a UDP or TCP connection, check that your firewall is not blocking IP traffic

You should also ensure that the autopilot controller board has appropriate ArduPilot firmware installed and has booted correctly. (on Pixhawk there are useful LEDs and Sounds which can tell you the state of the autopilot).

5.6 Mission Planner Initial SETUP

This section of Mission Planner, invoked by the Menu item SETUP at the top of Mission Planner, has several subsections. The subsection is where you set up and configure your auto pilot to prepare it for your particular vehicle. Typically, these sections are "must do" actions that are required. What you see when you enter this section depends on whether or not you are connected.

Mandatory Hardware

You will see this menu item If the auto pilot is connected. Click this menu item to see the items you must setup before you attempt to operate your vehicle.

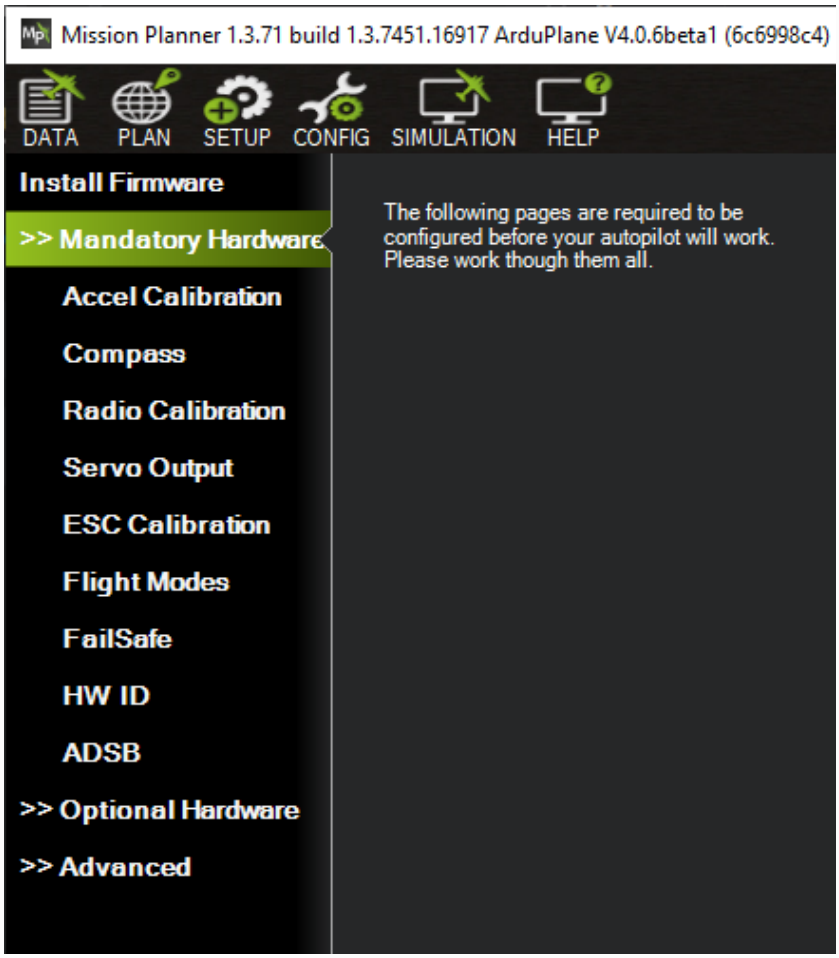

Figure 5.10: Mandatory Hardware

Before operating the vehicle, you must setup:

- Accel Calibration
- Compass (optional for Plane)
- Radio Calibration
- Servo Output: configure each output's function. Default values get loaded upon initial firmware install, but be sure to check them.
- ESC Calibration for Copter only: (not required for ESCs running DShot protocol, but must be configured in ArduPilot) Plane uses its own ESC Calibration technique, but is also a mandatory setup item.
- Flight Modes
- Failsafe

5.61 Accelerometer Calibration

Accelerometer calibration is mandatory in ArduPilot. Accelerometer calibration cannot be performed while vehicle is armed.

If the board is mounted in a non-standard orientation (i.e., arrow is not pointing forward) then please ensure the position properly before doing the accelerometer calibration.
Under Setup | Mandatory Hardware, select Accel Calibration from the left-side menu.

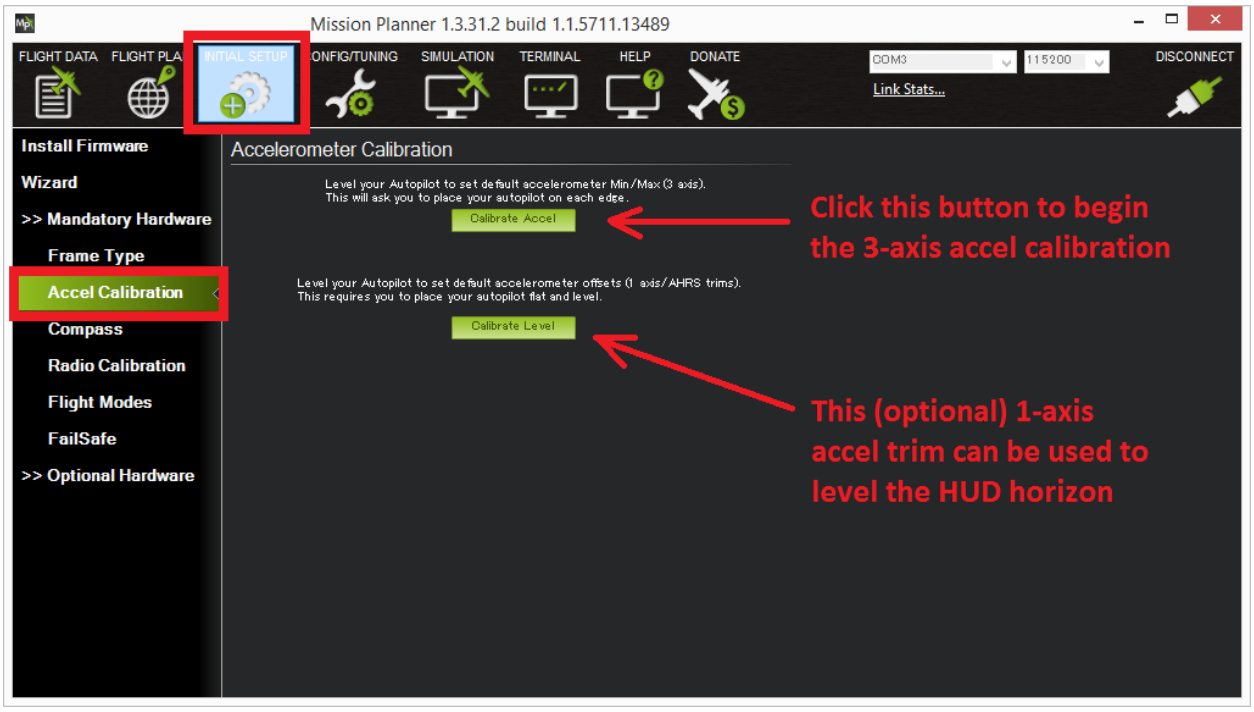

Figure 5.11: Accelerometer Calibration

Click **Calibrate Accel** to start the calibration.

Mission Planner will prompt you to place the vehicle each calibration position. Press any key to indicate that the autopilot is in position and then proceed to the next orientation.

The calibration positions are: level, on right side, left side, nose down, nose up and on its back.

Figure 5.12: Accelerometer Calibration Positions (Copter)

It is important that the vehicle is kept still immediately after pressing the key for each step. This is more important than getting the angle exactly right, i.e., left being 90deg to horizontal, etc. Except for the first "LEVEL", the positions can be within 20 degs of being exact. Being still in each position as you press the key is much more important.
You may need to calibrate the board before it is mounted if the size/shape of the vehicle makes this difficult.
The level position is the most important to get right as this will be the attitude that your controller considers level while flying.

- Proceed through the required positions, using the **Click when Done** button after each position is reached.
- When you've completed the calibration process, Mission Planner will display "Calibration Successful!" as shown below.

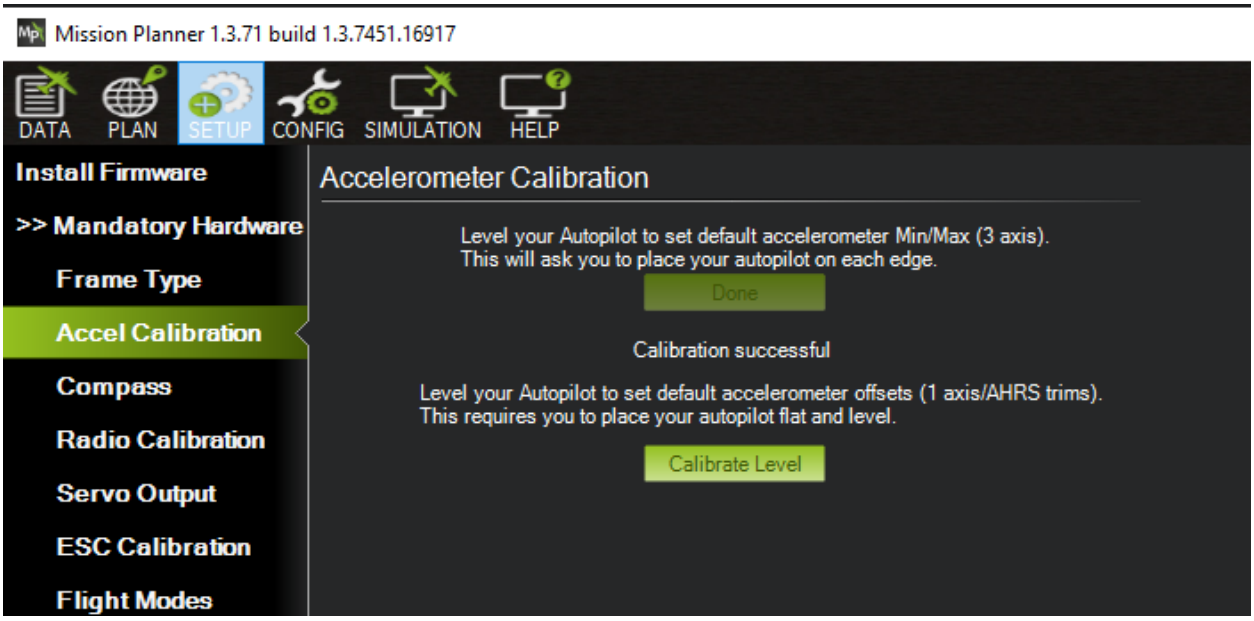

Figure 5.13: Mission Planner: Calibration Successful

5.62 Compass Calibration

Compass calibration cannot be performed while vehicle is armed. Do not calibrate the compasses near any metallic or magnetic field producing object (computers, cell phones, metal desks, power supplies, etc.) or incorrect calibration will occur.

It is not necessary to recalibrate the compass when the vehicle is flown at a new location because ArduPilot includes a "world magnetic model" which allows converting the location's magnetic North to true North without recalibrating. In addition, the location's "inclination" is calibrated at startup and then again soon after takeoff. It is important that when compass calibration is done, the vehicle have a good 3D gps lock, in order to assure the best setup. If necessary, move outdoors in order to get a good 3D gps lock before doing the compass calibration.

Under SETUP| Mandatory Hardware select Compass.

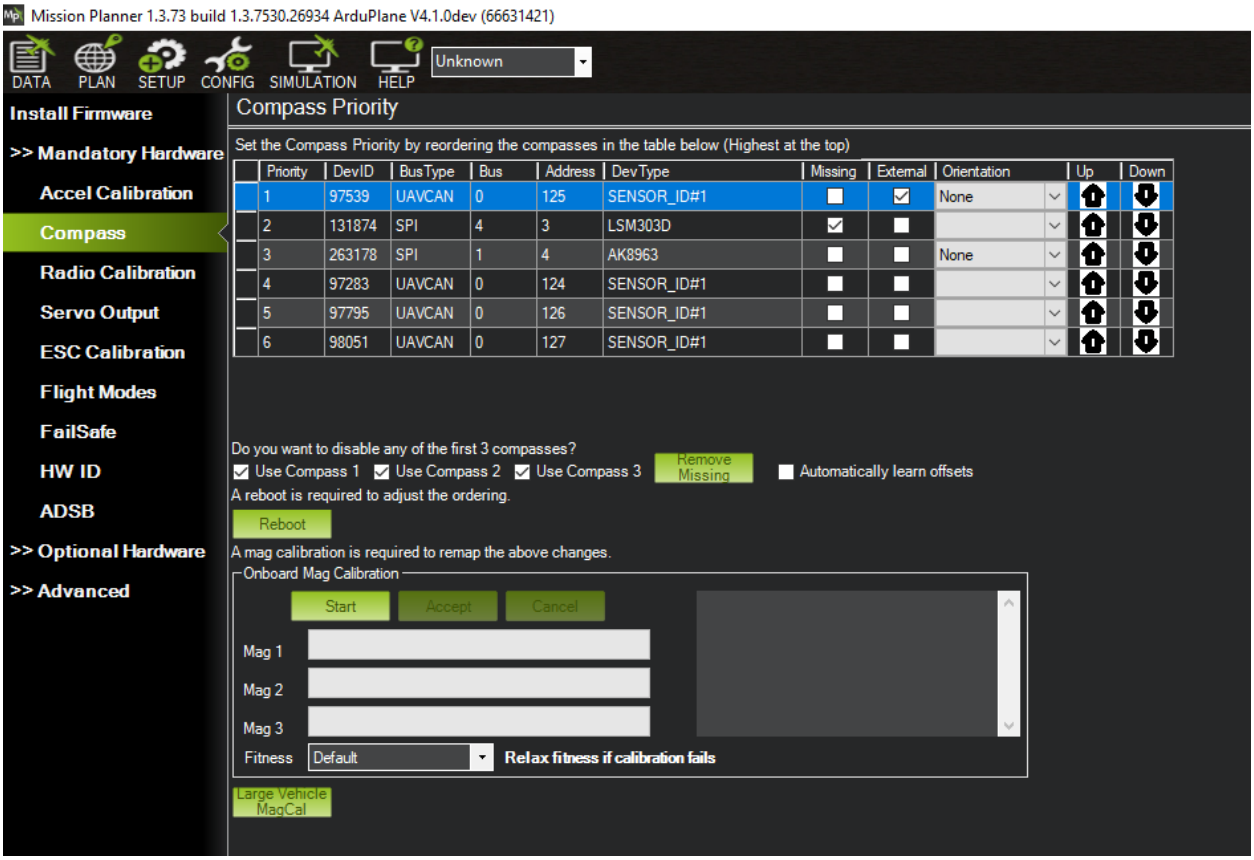

Figure 5.14: Mission Planner: Compass Calibration

You may wish to disable any internal compasses if you are consistently seeing the "inconsistent compasses" pre-arm message often and you are sure that the external compass is calibrated.
"Onboard Calibration" is a calibration routine that runs on the autopilot. This method is more accurate than the older "Offboard Calibration" (aka "Live Calibration") which runs on the ground station because in addition to offsets, scaling and orientation are also automatically determined.

To perform the onboard calibration of all compasses:
click the "Onboard Mag Calibration" section's "Start" button

- if your autopilot has a buzzer attached you should hear a single tone followed by short beep once per second
- hold the vehicle in the air and rotate it so that each side (front, back, left, right, top and bottom) points down towards the earth for a few seconds in turn. Consider a full 360-degree turn with each turn pointing a different direction of the vehicle to the ground. It will result in 6 full turns plus possibly some additional time and turns to confirm the calibration or retry if it initially does not pass.

Figure 5.15: 360-deegree turns for compass calibration

- as the vehicle is rotated the green bars should extend further and further to the right until the calibration completes
- upon successful completion three rising tones will be emitted and a "Please reboot the autopilot" window will appear and you will need to reboot the autopilot before it is possible to arm the vehicle.

If calibration fails:

- you will hear an "unhappy" failure tone, the green bars may reset to the left, and the calibration routine may restart (depending upon the ground station). Mission

Planner will automatically retry, so continue to rotate the vehicle as instructed above.

- if a compass is not calibrating, consider moving to a different area away from magnetic disturbances, and remove electronics from your pockets.
- if, after multiple attempts, the compass has not passed the calibration, Press the "Cancel" button and change the "Fitness" drop-down to a more relaxed setting and try again.
- if compass calibration still fails it may help to raise COMPASS_OFFS_MAX from 850 to 2000 or even 3000
- finally, if a single compass is not calibrating and you trust the others, disable it.

5.63 Radio Control Calibration

RC transmitters allow the pilot to set the flight mode, control the vehicle's movement and orientation and also turn on/off auxiliary functions (i.e., raising and lowering landing gear, etc).

RC Calibration involves capturing each RC input channel's minimum, maximum and "trim" values so that ArduPilot can correctly interpret the input.

Check the Transmitter's Setup

- Ensure the battery is disconnected (this is important because it is possible to accidentally arm the vehicle during the RC calibration process).
- Ensure the RC receiver is connected to the autopilot.
- Turn on your RC transmitter and if it has "trim tabs" ensure they are in the middle.
- Connect the autopilot to the PC using a USB cable.
- On the Mission Planner press the "Connect" button and open Mission Planner's **INITIAL SETUP | Mandatory Hardware | Radio Calibration** screen.
- Some green bars should appear showing the ArduPilot is receiving input from the Transmitter/Receiver. If no bars appear check the receiver's LED:
    - No lights may indicate that it is incorrectly wired to the autopilot. Look for connectors that may have been inserted upside down.
    - A Red or flashing LED may indicate that your RC transmitter/receiver need be bound.

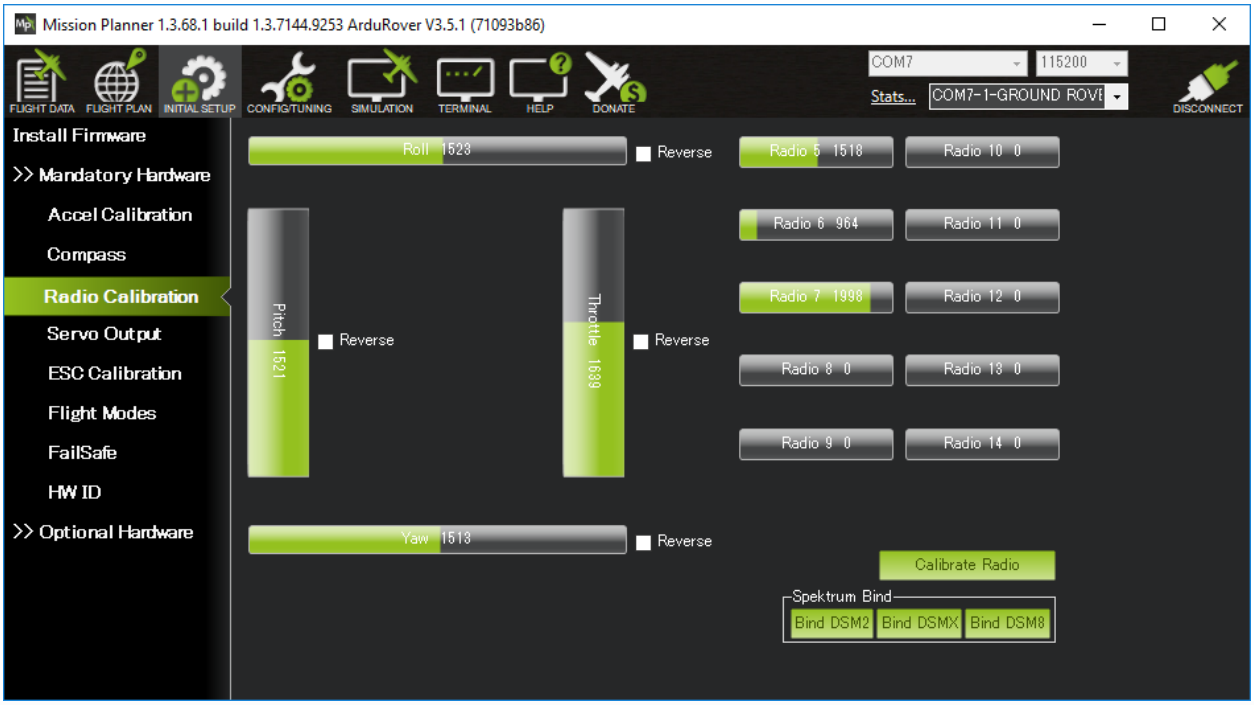

Figure 5.16: Radio Calibration

- Check the channel mapping in the transmitter (ie. check which inputs channels are controlled by the transmitter's sticks, switches and knobs) by moving the sticks, knobs and switches and observing which (if any) green bars move. If this is the first time the transmitter is being used with ArduPilot it is likely that the transmitter's channel mapping will need to be changed and normally this is done on the transmitter itself using its built-in configuration menu
    - Determine if your transmitter is Mode1 or Mode2 (see below)
    - Roll stick should control channel 1
    - Pitch stick should control channel 2
    - Throttle stick should control channel 3
    - Yaw stick should control channel 4
- Move the transmitter's roll, pitch, throttle and yaw sticks and ensure the green bars move in the correct direction:
    - for roll, throttle and yaw channels, the green bars should move in the same direction as the transmitter's physical sticks
    - for pitch, the green bar should move in the opposite direction to the transmitter's physical stick

- if one of the green bars moves in the incorrect direction reverse the channel in the transmitter itself. If it is not possible to reverse the channel in the transmitter you may reverse the channel in ArduPilot.

- Open Mission Planner's **INITIAL SETUP | Mandatory Hardware | Radio Calibration** screen.
- Click on the green "Calibrate Radio" button on the bottom right.
- Press "OK" when prompted to check the radio control equipment is on, battery is not connected, and propellers are not attached.

Figure 5.17: Radio Calibration Homepage

- Move the transmitter's control sticks, knobs and switches to their limits. Red lines will appear across the calibration bars to show minimum and maximum values seen so far.

Figure 5.18: Radio Calibration Successful

- Select **Click when Done.**
- A window will appear with the prompt, “Ensure all your sticks are centred and throttle is down and click ok to continue”. Move the throttle to zero and press “OK”.
- Mission Planner will show a summary of the calibration data. Normal values are around 1100 for minimums and 1900 for maximums.

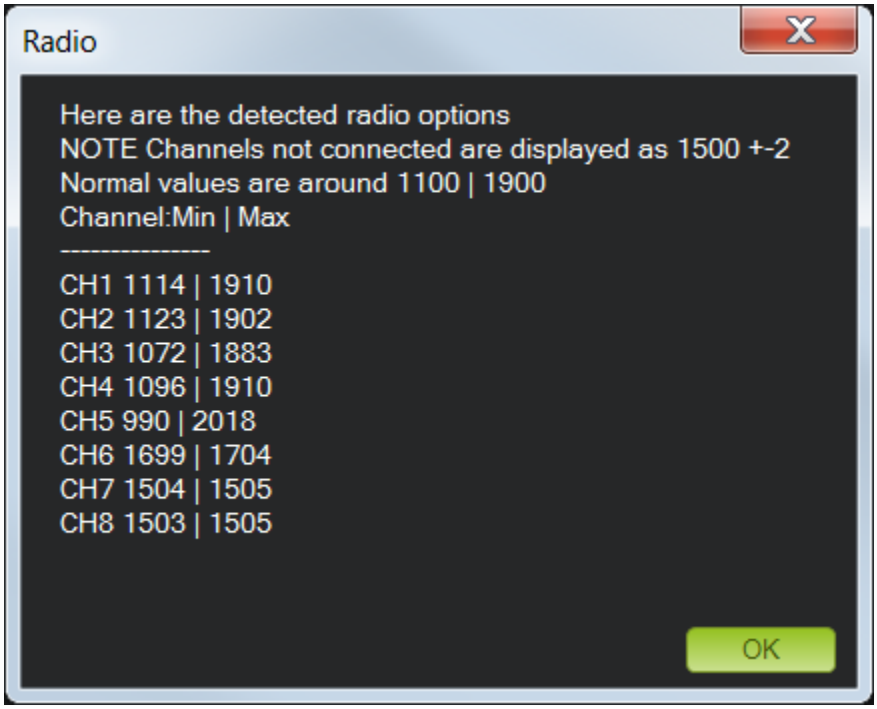

Figure 5.19: Radio Calibration Data

5.64 Electronic Speed Controller (ESC) Calibration

Electronic speed controllers are responsible for spinning the motors at the speed requested by the autopilot. Most ESCs need to be calibrated so that they know the minimum and maximum pwm values that the flight controller will send. This page provides instructions for calibrating ESCs.

About ESC Calibration

ESC calibration will vary based on what brand of ESC you are using, so always refer to the documentation for the brand of ESC you are using for specific information (such as tones). "All at once" calibration works well for most ESCs, so it is good idea to attempt it first and if that fails try the "Manual ESC-by-ESC" method.

- Some brands of ESC do not allow calibration and will not arm unless you adjust your radio's throttle end-points so that the minimum throttle is around 1000 PWM and maximum is around 2000. Note that if you change the end-points on your TX you must re-do the Radio Calibration. Alternatively with Copter-3.4 (and

higher) you may manually set the MOT_PWM_MIN to 1000 and MOT_PWM_MAX to 2000.

- Begin this procedure only after you have completed the radio control calibration and Connect ESCs and motors part of the Autopilot System Assembly Instructions. Next follow these steps:

Before calibrating ESCs, please ensure that your copter has NO PROPS on it and that the APM is NOT CONNECTED to your computer via USB and the Lipo battery is disconnected.

Figure 5.20: Precautions before ESC calibration

All at once calibration

1. Turn on your transmitter and put the throttle stick at maximum.

Figure 5.21: Throttle Maximum

2. Connect the Lipo battery. The autopilot's red, blue and yellow LEDs will light up in a cyclical pattern. This means the it's ready to go into ESC calibration mode the next time you plug it in.

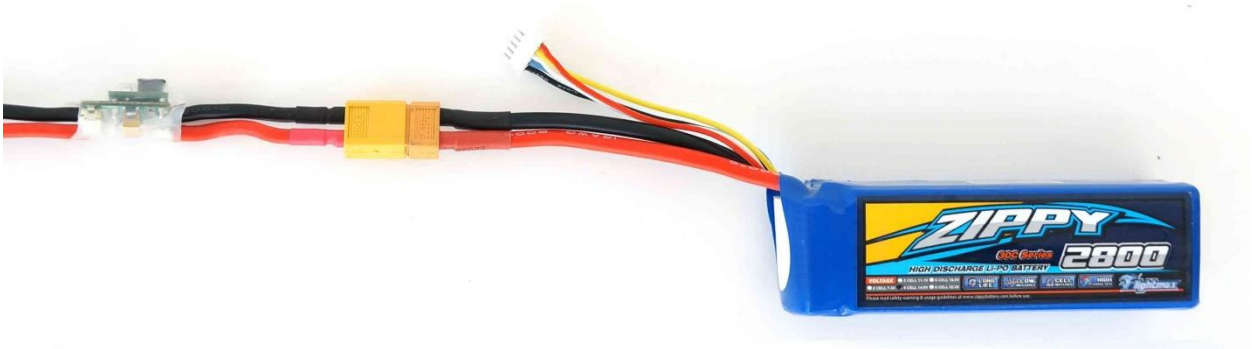

Figure 5.22: Connect Battery

3. With the transmitter throttle stick still high, disconnect and reconnect the battery.

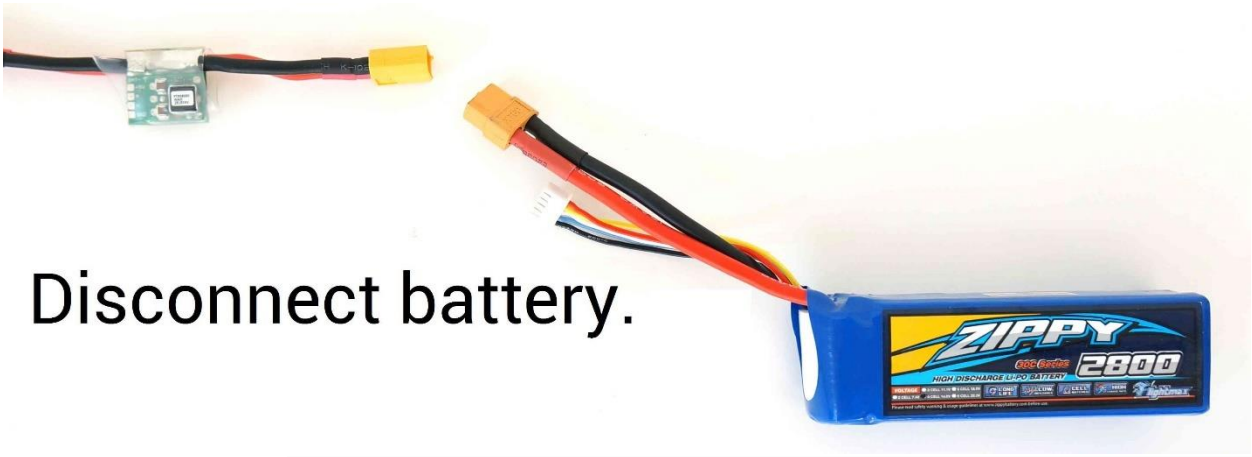

Figure 5.23: Disconnect Battery

Figure 5.24: Connect Battery

4. For Autopilots with a safety switch, push it until the LED displays solid red
5. The autopilot is now in ESC calibration mode. (On an APM you may notice the red and blue LEDs blinking alternatively on and off like a police car).
6. Wait for your ESCs to emit the musical tone, the regular number of beeps indicating your battery's cell count (i.e., 3 for 3S, 4 for 4S) and then an additional two beeps to indicate that the maximum throttle has been captured.
7. Pull the transmitter's throttle stick down to its minimum position.

Figure 5.25: Throttle Minimum

8. The ESCs should then emit a long tone indicating that the minimum throttle has been captured and the calibration is complete.
9. If the long tone indicating successful calibration was heard, the ESCs are "live" now and if you raise the throttle a bit they should spin. Test that the motors spin by raising the throttle a bit and then lowering it again.
10. Set the throttle to minimum and disconnect the battery to exit ESC-calibration mode.

Manual ESC-by-ESC Calibration

1. Plug one of your ESC three-wire cables into the throttle channel of the RC receiver. (This is usually channel 3.)
2. Turn on the transmitter and set throttle stick to maximum (full up).
3. Connect the LiPo battery
4. You will hear a musical tone then two beeps.
5. After the two beeps, lower the throttle stick to full down.
6. You will then hear a number of beeps (one for each battery cell you're using) and finally a single long beep indicating the end points have been set and the ESC is calibrated.
7. Disconnect battery. Repeat these steps for all ESCs.
8. If it appears that the ESC's did not calibrate then the throttle channel on the transmitter might need to be reversed.
9. If you are still having trouble after trying these methods (for example, ESCs still beep continuously) try lowering your throttle trim 50%.
10. You can also try powering your APM board via the USB first to boot it up before plugging in the LiPo.

Semi-Automatic ESC-by-ESC Calibration

1. Connect to the autopilot from a ground station such as the Mission Planner and set the ESC_CALIBRATION parameter to 3
2. Disconnect the battery and USB cable so the autopilot powers down
3. Connect the battery
4. The arming tone will be played (if the vehicle has a buzzer attached)

5. If using a autopilot with a safety button (like the Pixhawk) press it until it displays solid red
6. You will hear a musical tone then two beeps
7. A few seconds later you should hear a number of beeps (one for each battery cell you're using) and finally a single long beep indicating the end points have been set and the ESC is calibrated
8. Disconnect the battery and power up again.

Testing

Once you have calibrated your ESCs, you can test them by plugging in your LiPo. Remember: no propellers!

- Ensure your transmitter's flight mode switch is set to "Stabilize Mode".
- Arm your copter.
- Give a small amount of throttle. All motors should spin at about same speed and they should start at the same time. If the motors do not all start at the same time and spin at the same speed, the ESC's are still not properly calibrated.
- Disarm your copter.

Troubleshooting

The All-at-once ESC calibration mode simply causes the APM to pass through the pilot's throttle directly through to the ESCs. If you power up the APM while in this mode you'll send the same PWM signal to all the ESCs. That's all it does. Many ESCs use full throttle at startup to enter programming mode, full throttle position is then saved as the upper end point and when you pull the throttle down to zero, that position is saved as the lower end point.

If after calibration your motors do NOT spin same speed nor start at the same time, repeat the calibration process. If you tried the auto calibration above and it didn't work or the ESCs do not drive the motors identically, try the manual calibration method described above. That should work almost every time. (Rarely after a full manual calibration you will also need to do an additional final automatic calibration).

CHAPTER 6: INSTALLATION AND SET-UP OF RASPBIAN BUSTER FOR RASPBERRY PI-4

Required Hardware:

1. Raspberry Pi 4 Model B
2. 8 GB+ MicroSD card
3. MicroSD card reader

6.1 Download the Buster Image

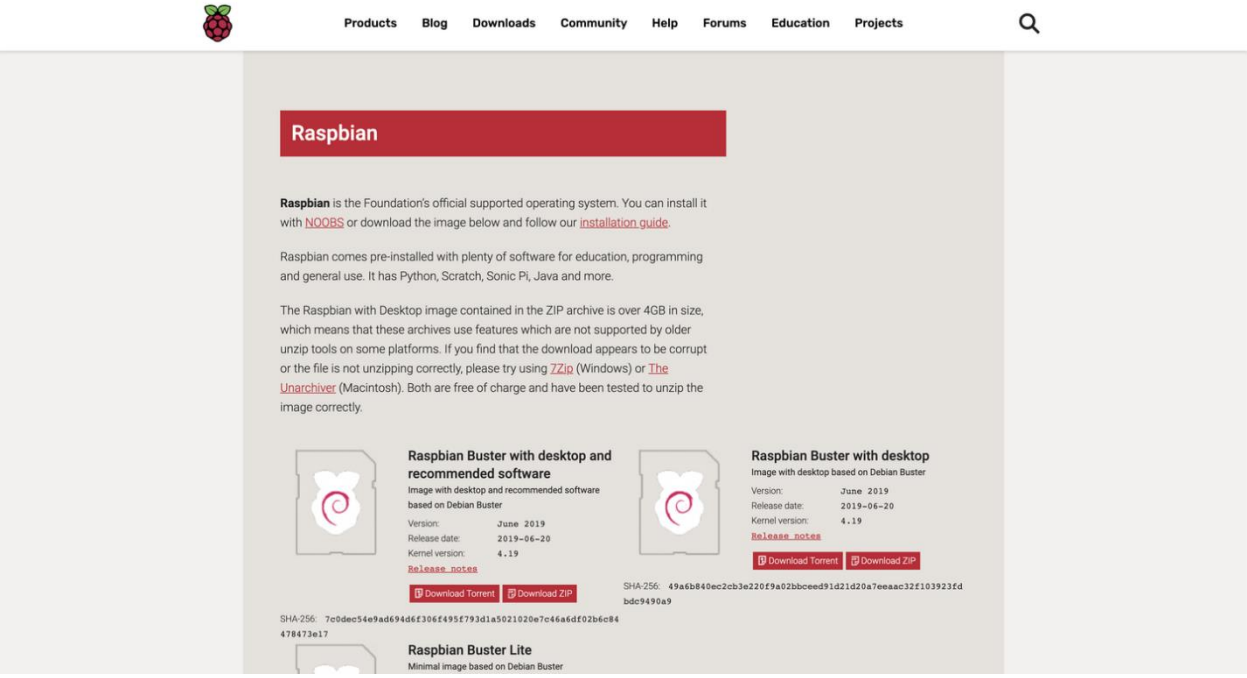

Figure 6.1: Raspbian Buster Image

You can find the Raspbian Buster download here. There are three options to choose from: Raspbian Buster with desktop and recommended software, Raspbian Buster with desktop, Raspbian Buster Lite. Any of these options will work.

Choose Raspbian Buster with desktop (or Raspbian Buster with desktop and recommended software) if you want to have access to the Raspbian GUI (graphical user interface); in other words, if you want to log in and be able to access a desktop, icons, etc. like you would with Windows or macOS.

Choose Raspbian Buster Lite if you're comfortable working on the command line and your project doesn't require a GUI. This option is typically for more advanced users, and it's often a good choice as it's a smaller distribution, uses less power, and fewer resources.

6.2 Download Etcher and Flash the Disc Image using Etcher

To get started, download and open Etcher.

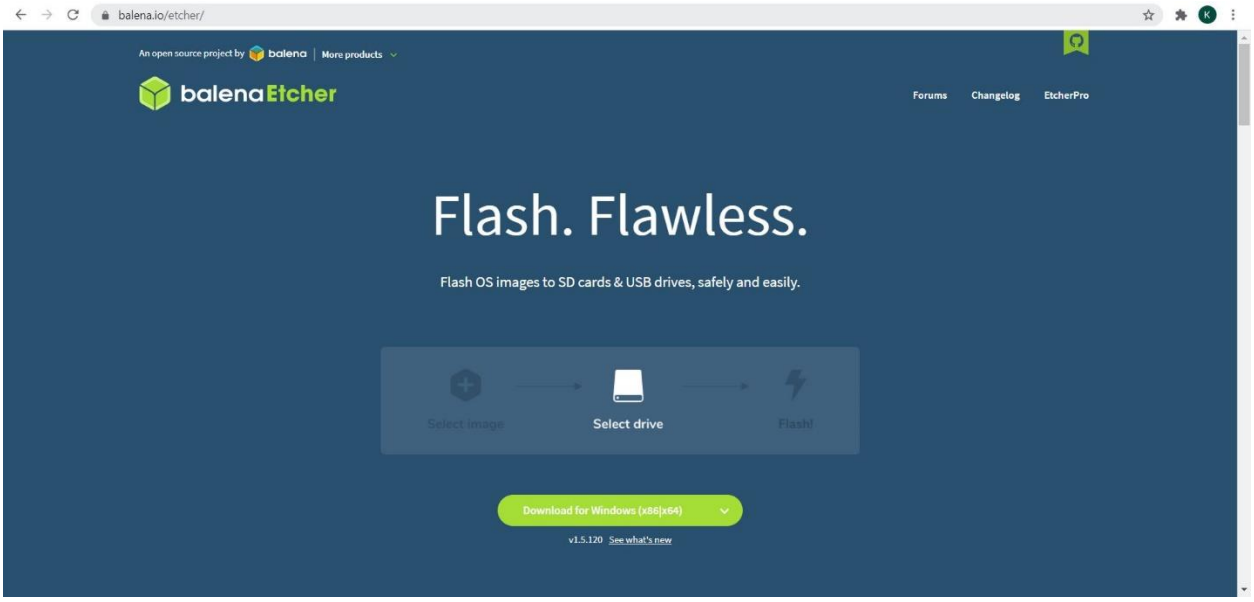

Figure 6.2: Download and Install Etcher

Put the MicroSD card into computer using an Adapter.

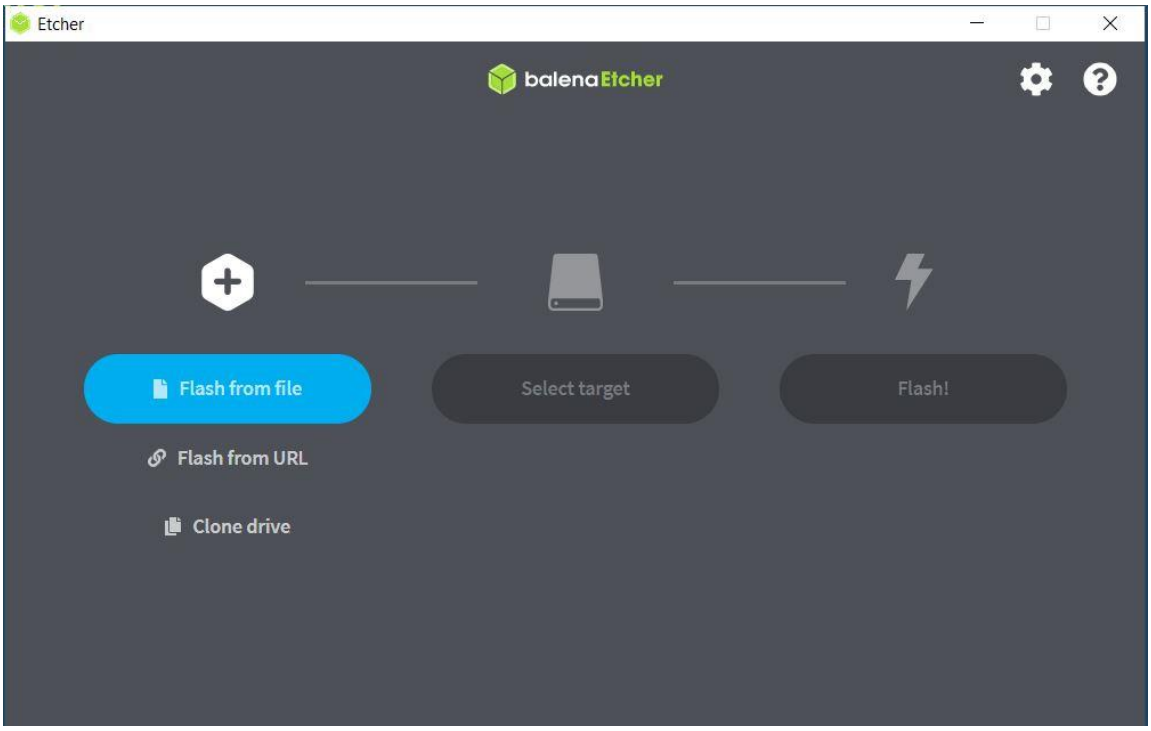

Figure 6.3: Flash the Disc Image

Select the Raspbian disk image you just downloaded, select your Micro SD card, then click Flash. This will take a few minutes, but after Etcher finishes, you're ready to start using Buster!

6.3 Boot the Raspberry Pi 4

At this point, you're ready to insert the newly flashed SD card into your Pi and boot up!

Start up your Raspberry Pi

Your Raspberry Pi doesn't have a power switch. As soon as you connect it to a power outlet, it will turn on.

- Plug the power supply into a socket and connect it to your Raspberry Pi's power port.

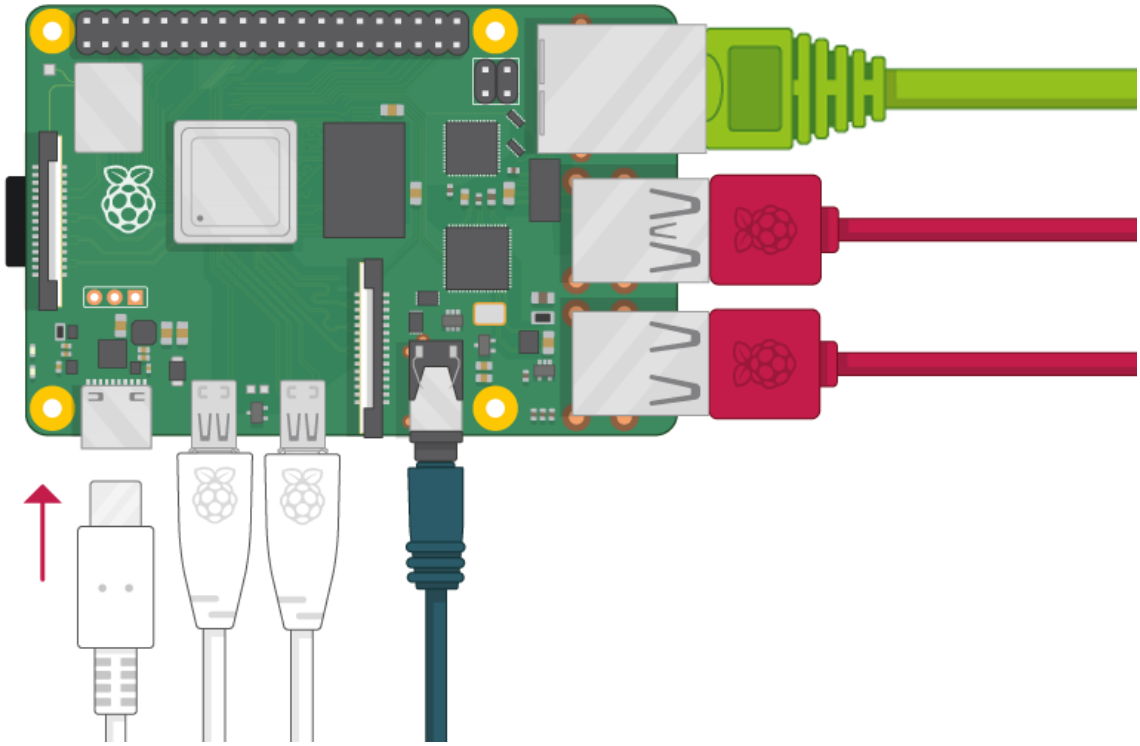

Figure 6.4: Raspberry Pi LED Indication

You should see a red LED light up on the Raspberry Pi, which indicates that Raspberry Pi is connected to power. As it starts up (this is also called booting), you will see raspberries appear in the top left-hand corner of your screen.

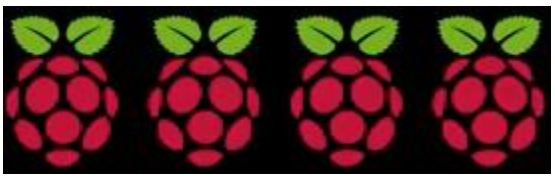

Figure 6.5: HDMI Display

After a few seconds the Raspberry Pi OS desktop will appear.

Figure 6.6: Raspberry Pi OS desktop

6.4 Finishing the setup

When you start your Raspberry Pi for the first time, the Welcome to Raspberry Pi application will pop up and guide you through the initial setup.

Figure 6.7: Welcome to Raspberry Pi

- Click on Next to start the setup.
- Set your Country, Language, and Time zone, then click on Next again.

Figure 6.8: Raspberry Pi Set up 1

- Enter a new password for your Raspberry Pi and click on Next.

Figure 6.9: Raspberry Pi Set up 2

- Connect to your wireless network by selecting its name, entering the password, and clicking on Next.

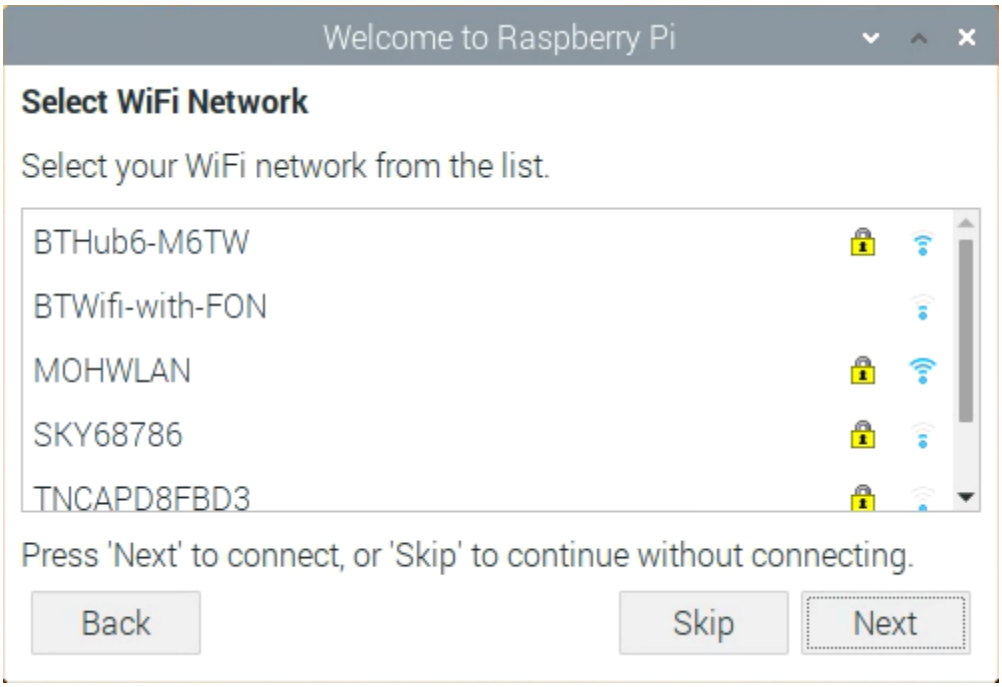

Figure 6.10: Raspberry Pi Set up 3

Note: If your model of Raspberry Pi doesn't have wireless connectivity, you won't see this screen.

Note: Wait until the wireless connection icon appears and the correct time is shown before trying to update the software.

- Click on Next, and let the wizard check for updates to Raspberry Pi OS and install them (this might take a little while).

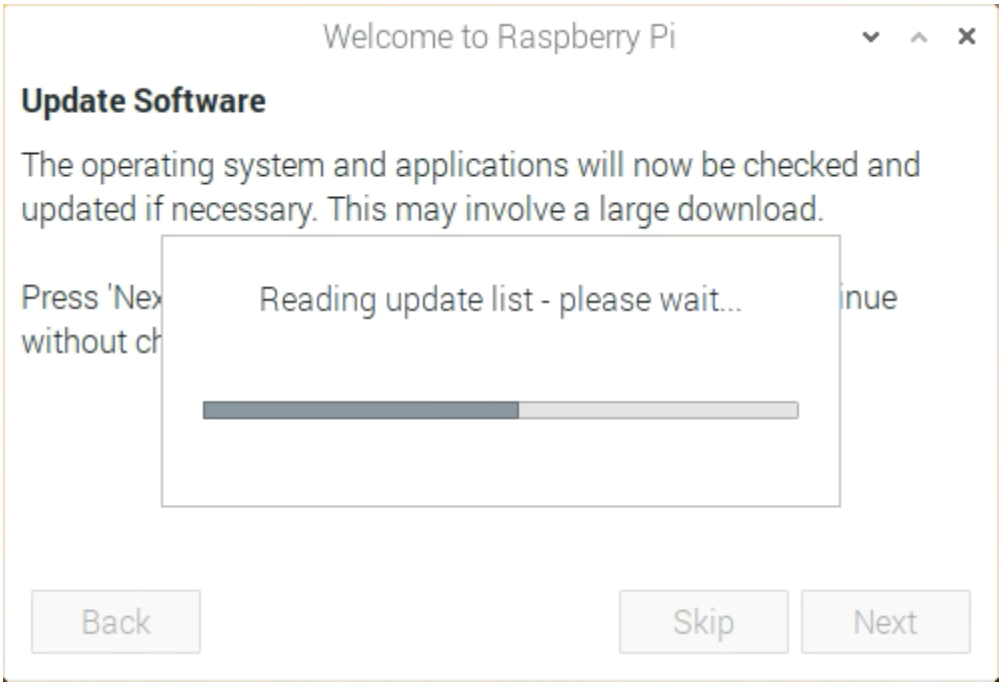

Figure 6.11: Raspberry Pi Set up 4

- Click on Restart to finish the setup.

Note: You will only need to reboot if that's necessary to complete an update.

Figure 6.12: Raspberry Pi Set up 5

CHAPTER 7: COMMUNICATING WITH RASPBERRY PI VIA MAVLINK

Now, connect and configure a Raspberry Pi (RPi) so that it is able to communicate with a flight controller using the MAVLink protocol over a serial connection. This can be used to perform additional tasks such as image recognition which simply cannot be done by the flight controller due to the memory requirements for storing images.

7.1 Connecting the Flight controller and RPi Hardware

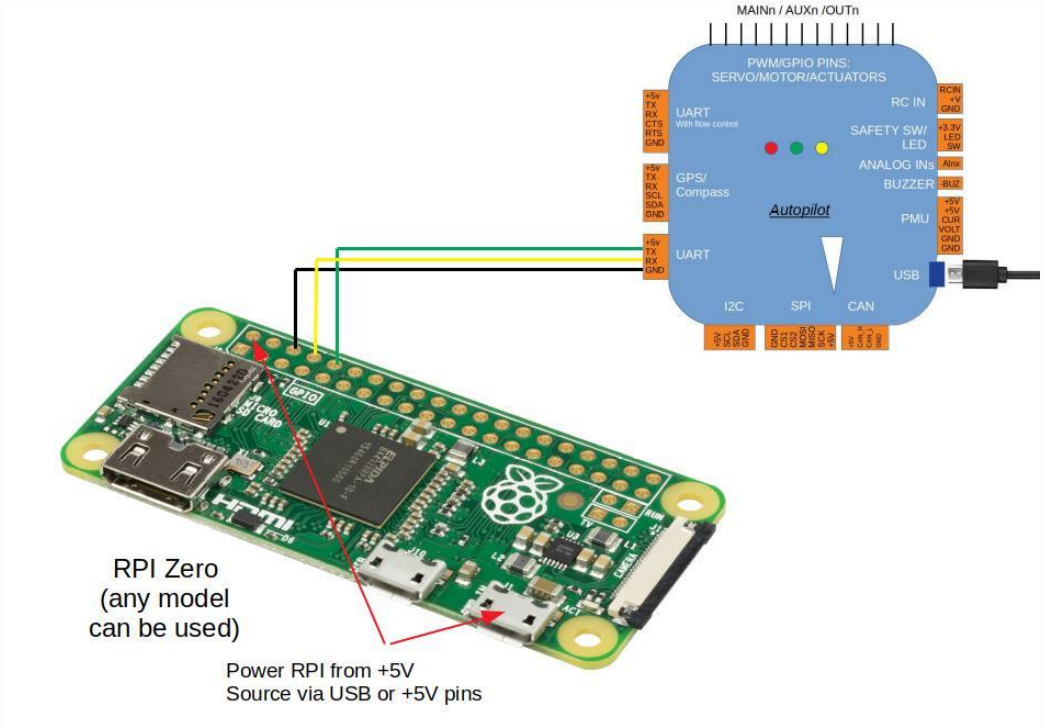

Figure 7.1: Hardware connection between APM 2.8 and Raspberry Pi0

Connect the flight controller's TELEM2 port to the RPi's Ground, TX and RX pins as shown in the image above.
The RPi can be powered by connecting +5V source to the +5V pin **or** from USB in.
Addon boards such as the Pi-Connect can simplify the connection of the RPi by providing a power supply and telemetry port.

7.2 Setting up the flight controller

Connect to the flight controller with a ground station (i.e., Mission Planner) and set the following parameters:

- SERIAL2_PROTOCOL = 2 (the default) to enable MAVLink 2 on the serial port.
- SERIAL2_BAUD = 921 so the flight controller can communicate with the RPi at 921600 baud.
- LOG_BACKEND_TYPE = 3 if you are using APSync to stream the dataflash log files to the RPi

7.3 Configure the serial port (UART)

If not already configured, the Raspberry Pi's serial port (UART) will need to be enabled. Use the Raspberry Pi configuration utility for this.

Type:

```
sudo raspi-config
```

And in the utility, select "Interfacing Options":

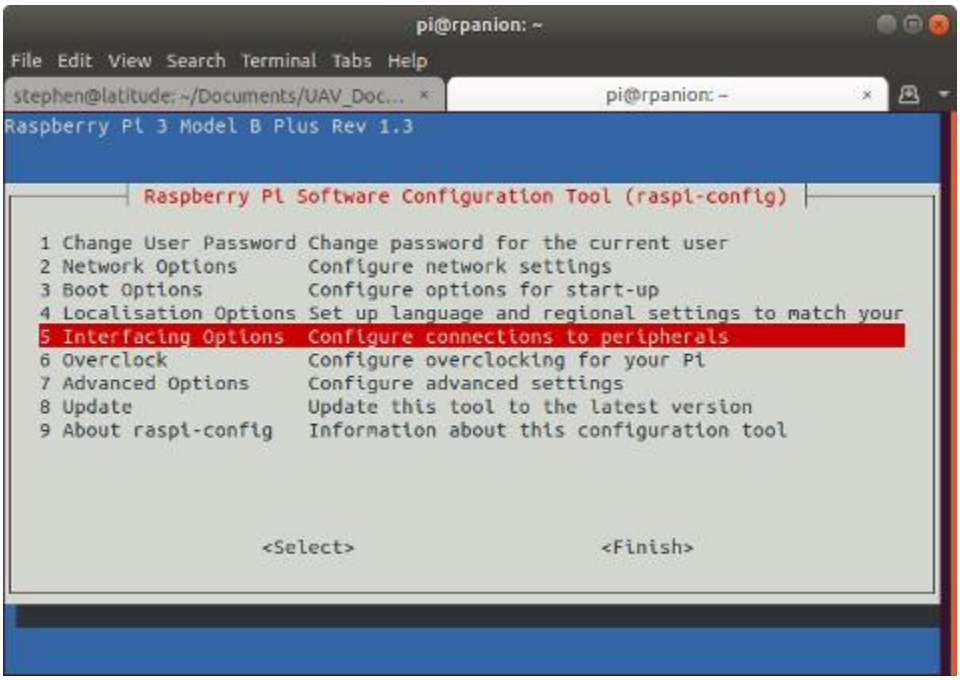

Figure 7.2: Serial Port configuration

And then "Serial":

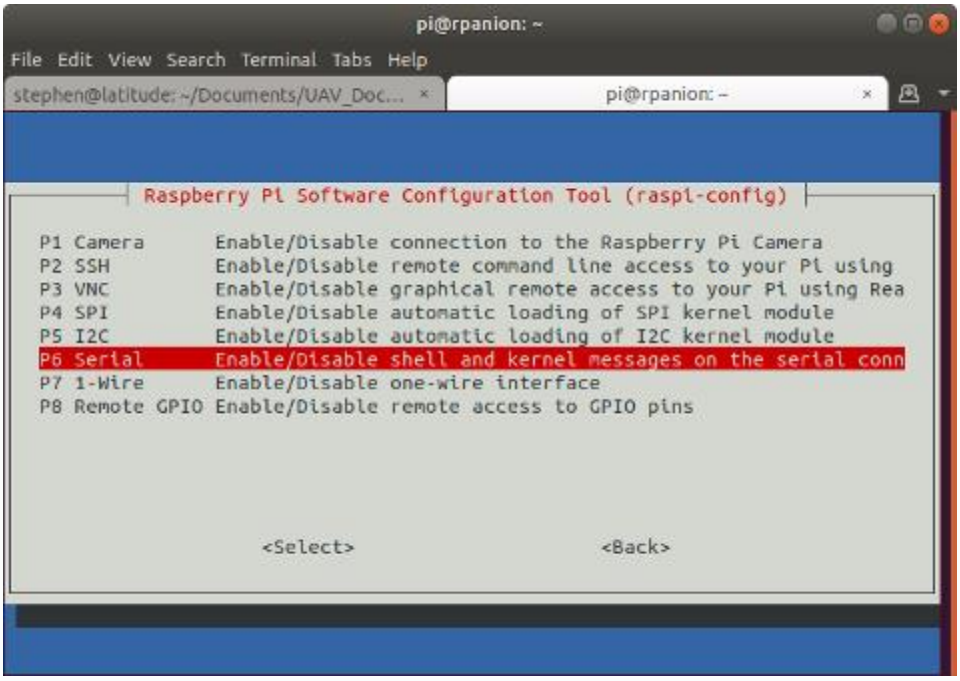

Figure 7.3: UART configuration

When prompted, select `no` to "Would you like a login shell to be accessible over serial?".

When prompted, select `yes` to "Would you like the serial port hardware to be enabled?".

Reboot the Raspberry Pi when you are done.

The Raspberry Pi's serial port will now be usable on `/dev/serial0`.

7.4 MAVProxy

MAVProxy can be used to send commands to the flight controller from the Pi. It can also be used to route telemetry to other network endpoints.
To test the RPi and flight controller are able to communicate with each other first ensure the RPi and flight controller are powered, then in a console on the RPi type:

```
python3 mavproxy.py --master=/dev/serial0 --baudrate 921600 --aircraft MyCopter
```

Once MAVProxy has started you should be able to type in the following command to display the `ARMING_CHECK` parameters value

```
param show ARMING_CHECK
param set ARMING_CHECK 0
```

arm throttle

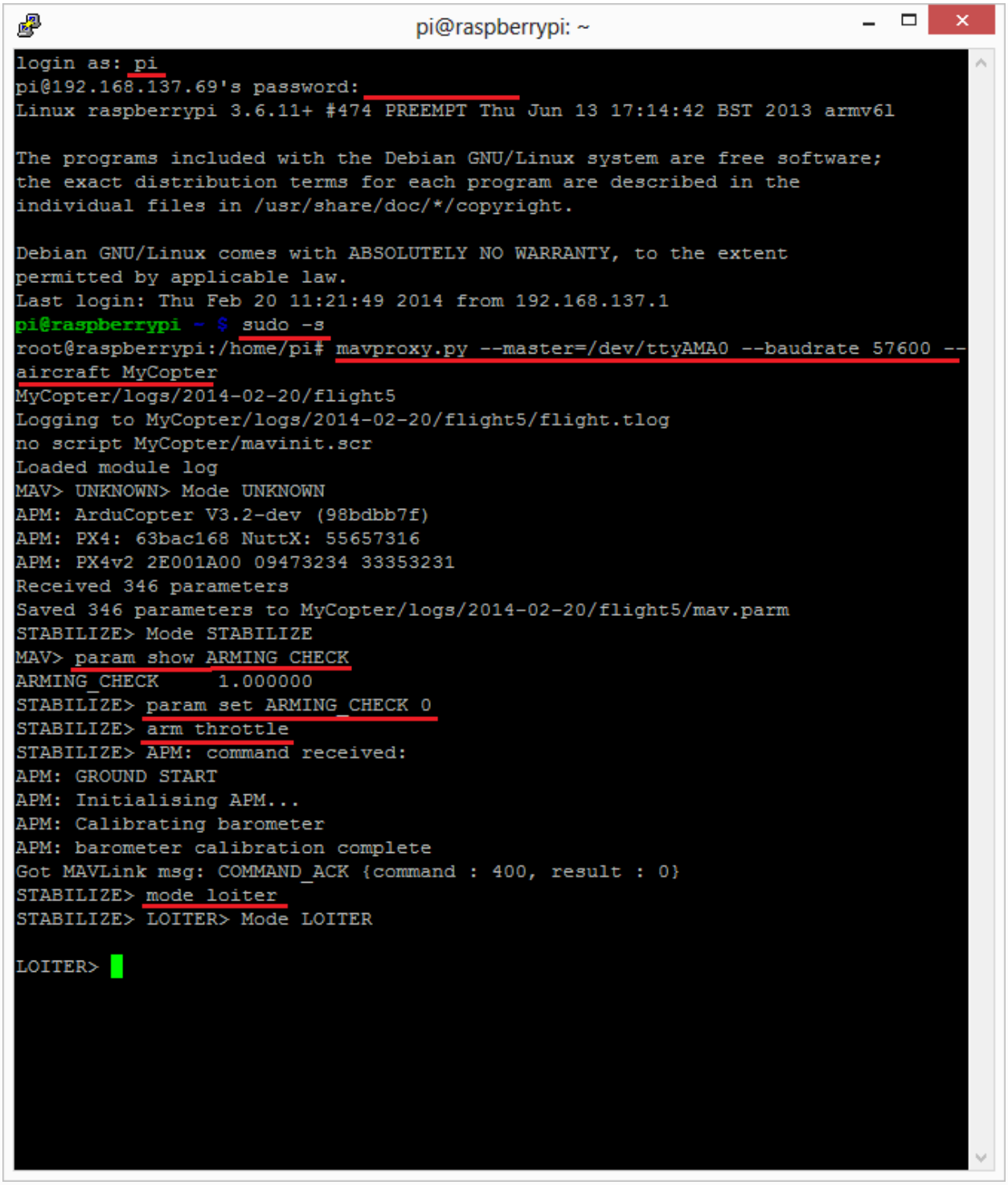

Figure 7.4: MAVProxy connection

7.5 Mavlink-router

Mavlink-router is used to route telemetry between the RPi's serial port and any network endpoints.

After installing, edit the mavlink-router config file's `/etc/mavlink-router/main.conf` UART section to:

```
[UartEndpoint to_fc]
Device = /dev/serial0
Baud = 921600
```

You will also need to add an additional UDP endpoint allow other ground stations on the same network to connect to the Pi. Edit the mavlink-router config file `/etc/mavlink-router/main.conf` to include:

```
[UdpEndpoint to_14550_external]
Mode = eavesdropping
Address = 0.0.0.0
Port = 14550
PortLock = 0
```

7.6 DroneKit

DroneKit-Python enables developers to make applications that run on a RPi and communicate with the ArduPilot flight controller utilizing a low-idleness interface. Locally available applications can essentially improve the autopilot. These applications add more prominent knowledge to vehicle conduct and perform errands that are computationally escalated or time sensitive (PC vision, way arranging, or 3D displays). DroneKit-Python can likewise be utilized for ground station applications by speaking with vehicles over a higher dormancy RF-connect.

The DroneKit speaks with the Pixhawk through the MAVLink protocol. It gives automatic access to an associated vehicle's telemetry, state, and parameter data. MAVLink empowers both mission administration tasks and direct control over vehicle development.

DroneKit-Python, a Python based version of DroneKit chosen for this thesis, works better with vehicles that impart utilizing the MAVLink convention. It can run on Linux, Mac, or Windows. DroneKit-Python provides the following classes and methods to:

- Connect to a vehicle (or multiple vehicles) from a script.
- Set vehicle state/telemetry and parameter information.
- Receive asynchronous notification of state changes.
- Guide a UAV to specified position (GUIDED mode).
- Send custom messages to control UAV movement.
- Create and manage waypoint missions (AUTO mode).
- Override RC channel settings.

7.6.1 Installation

DroneKit-Python is installed on the RPi from a software package management system known as pip. The first step is to install pip and python-dev with the help of the following command:

```
sudo apt-get install python-pip python-dev
```

Then, pip is used to install DroneKit-Python.

```
pip install dronekit
```

Upon installing DroneKit-Python, there are a few general considerations. The first is that some orders might be quietly overlooked by the autopilot. If this is the case, it is in a state in which it cannot securely follow up. Secondly, messages are interrupted or not sent most of the time. The last point to consider is that commands can be received by the autopilot from numerous sources.

7.7 Connecting with the Mission Planner

The flight controller will respond to MAVLink commands received through Telemetry 1 and Telemetry 2 ports meaning that both the RPi and the regular ground station (i.e., Mission planner, etc) can be connected. In addition it is possible to connect the Mission Planner to the MAVProxy application running on the RPi similar to how it is done for SITL.

Primarily this means adding an `--out <ipaddress>:14550` to the MAVProxy startup command with the being the address of the PC running the mission planner. On windows the `ipconfig` can be used to determine that IP address. On the computer used to write this wiki page the MAVProxy command became:

```
mavproxy.py --master=/dev/ttyAMA0 --baudrate 57600 --out 192.168.137.1:14550 --aircraft MyCopter
```

Connecting with the mission planner is shown below:

Figure 7.5: Successful UDP connection

Here is the Python script to autonomously takeoff and land the drone.

```
from dronekit import connect, VehicleMode, LocationGlobalRelative
from pymavlink import mavutil
import time
import argparse
parser = argparse.ArgumentParser()
parser.add_argument('--connect', default='127.0.0.1:14550')
args = parser.parse_args()

# Connect to the Vehicle
print 'Connecting to vehicle on: %s' % args.connect
vehicle = connect(args.connect, baud=921600, wait_ready=True)
#921600 is the baudrate that you have set in the mission plannar or qgc

# Function to arm and then takeoff to a user specified altitude
def arm_and_takeoff(aTargetAltitude):

  print "Basic pre-arm checks"
  # Don't let the user try to arm until autopilot is ready
  while not vehicle.is_armable:
    print " Waiting for vehicle to initialise..."
    time.sleep(1)

  print "Arming motors"
  # Copter should arm in GUIDED mode
  vehicle.mode    = VehicleMode("GUIDED")
  vehicle.armed   = True

  while not vehicle.armed:
    print " Waiting for arming..."
    time.sleep(1)

  print "Taking off!"
  vehicle.simple_takeoff(aTargetAltitude) # Take off to target altitude

  # Check that vehicle has reached takeoff altitude
  while True:
    print " Altitude: ", vehicle.location.global_relative_frame.alt
    #Break and return from function just below target altitude.
    if vehicle.location.global_relative_frame.alt>=aTargetAltitude*0.95:
      print "Reached target altitude"
      break
    time.sleep(1)

# Initialize the takeoff sequence to 15m
arm_and_takeoff(15)

print("Take off complete")

# Hover for 10 seconds
time.sleep(15)

print("Now let's land")
vehicle.mode = VehicleMode("LAND")

# Close vehicle object
vehicle.close()
```

Figure 7.6: Python script for take-off and landing

CHAPTER 8: CONCLUSION

This research suggests a better method of communication between a drone and its user. Laptop control of a drone ensures that the user has privacy and reduces the need for a complex hand controller. Most Android smartphones can exhibit control for a drone by installing some developed applications. The user can control movements and predefined missions loaded in the drone. A mini Quad-X model was involved in testing the controllability using the software. Several issues related to the design and hardware were noticed in the model and resolved. The Quad-X model was more successful and was more stable during its flight. Reliable hardware components will eliminate any inconsistencies and can lead to more accurate and unbiased results.

The successful takeoff and landing were achieved using Python script and the Robotics Operating System (ROS) framework on an onboard Raspberry Pi, an onboard companion computer to perform machine learning algorithms to detect various regions of purpose through video. It provides an accessible and inexpensive platform that works on any web and SSH capable computer as a base station. It enables manual PID control, state estimation, and high-level planning, obstacle avoidance.

The result of controlling the quadcopter in term of takeoff, fly to specific location, and landing was done successfully. The implementation on the real quadcopter is done successfully and it gives desired output.

REFERENCES

[1] "3DR DroneKit," 3D Robotics, 2015. [Online]. Available: http://android.dronekit.io/pebble_app.html.
[2] D. Labs, "Google Play Store," droidplannerlabs@gmail.com, [Online]. Available: https://play.google.com/store/apps/details?id=org.droidplanner.android&hl=en_US
[3] Open source Ardupilot https://ardupilot.org/
[4] "Raspberry udp issue," 2015. [Online]. Available: https://github.com/ArduPilot/MAVProxy/issues/121
[5] 3DR, "Connecting pixhawk with Raspberry Pi," [Online]. Available: http://ardupilot.org/dev/docs/Raspberry-Pi-via-mavlink.html.
[6] 3DR, "Startup options for MavProxy," [Online]. Available: https://ardupilot.github.io/MAVProxy/html/getting_started/starting.html.
[7] ArduPilot, "ArduPilot," Dev Team, 2016. [Online]. Available: http://ardupilot.org/dev/docs/Raspberry-Pi-via-mavlink.html.
[8] 3DR, "Dronekit-python," [Online]. Available: https://github.com/dronekit/dronekit-python