***Project Report***

***On­­­***

**Project Title**

**Submitted for the requirement of**

**Project course**

BACHELOR OF ENGINEERING

**COMPUTER SCIENCE & ENGINEERING**

****

**Submitted to: Submitted By:**

**Project Teacher Student Group**

**(Size = 13, Times)**

**NAME**

**UID**

**(Size 12, Times)**

**Mentor Signature**

**(Name & E-code)**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**CHANDIGARH UNIVERSITY, GHARUAN**

**June 2021**

### CERTIFICATE

This is to certify that the work embodied in this Project Report entitled **“ ”** being submitted by **“ ” -** UID **“ ” ,** 5th Semester for partial fulfillment of the requirement for the degree of **“ Bachelor of Engineering in Computer Science & Engineering ”** discipline in “ **Chandigarh University** ” during the academic session July-Dec 2019 is a record of bonafide piece of work, carried out by student under my supervision and guidance in the **“ Department of Computer Science & Engineering ”, Chandigarh University.**

**APPROVED & GUIDED BY:**

### DECLARATION

### I, student of Bachelor of Engineering in Computer Science & Engineering, 5th Semester , session: Jan – June 2020, Chandigarh University, hereby declare that the work presented in this Project Report entitled “ ” is the outcome of my own work, is bona fide and correct to the best of my knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any other university or anywhere else for the award of any degree or any professional diploma.

**Student details and Signature**

**APPROVED & GUIDED BY:**

To our parents, teachers and all the well wishers out there . . .

### 

**ABSTRACT**

A quadcopter can achieve vertical flight in a stable manner and be used to monitor or collect data in a specific region such as mapping terrains. Technological advances have reduced the cost and increase the performance of the low power microcontrollers that allowed the general public to develop their own quadcopter. The aim of this project is to design a light weight quadcopter system using budget friendlyRaspberry pi. The quadcopter will be controlled from a laptop or a RC (Remote controller) from a certain distance wirelessly. This small and highly manageable system would have a ability to hack the other wireless network. This quadcopter when hovered and enters a wifi network is capable of hacking the network by various attack powered by kali Linux and send all the crucial data to the base station. The project would have an impact on carrying out future defense missions and would provide better visual and audio of the enemy and help to use them their weapons on them. It will have the ability to hack other drones and make the “zombie drone” which work on our command, faster with more efficiency than any other option. It could also be used as a measure for survey or surveillance. The project used quadcopter kit that included a frame, motors, electronic speed controllers, Raspberry Pi 3, and Kali Linux operating system. Batteries, a transmitter, a receiver, a GPS module, and a micro SD card adaptor were interfaced with the kit.

**TABLE OF CONTENT**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Contributions | | |  | i |
| Acknowledgements | | | | ii |
| List of Figures | | | | iv |
| List of Tables | | |  | v |
| Glossary | | |  | vi |
| 1 | Introduction | | | 1 |
| 2 | Background | | | 3 |
|  | 2.1 | UAV Quadcopter . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 3 |
|  |  | 2.1.1 | Flight Control . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 3 |
|  |  | 2.1.2 | Advantages of a Quadcopter . . . . . . . . . . . . . . . . . . . . . . . . . . . | 5 |
|  | 2.2 | Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 6 |
|  | 2.3 | Specifications and Goals . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 6 |
|  | 2.4 | Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 7 |
| 3 | System Design | | | 8 |
|  | 3.1 | Project Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 8 |
|  |  | 3.1.1 | The Research Stage . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 8 |
|  |  | 3.1.2 | Building & Interfacing Stage . . . . . . . . . . . . . . . . . . . . . . . . . . . | 8 |
|  |  | 3.1.3 | The Tuning/Calibration Stage . . . . . . . . . . . . . . . . . . . . . . . . . . | 9 |
|  |  | 3.1.4 | The Programming Stage . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 9 |
|  | 3.2 | Physical Components . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 11 |
|  |  | 3.2.1 | Electric Motors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 12 |
|  | 3.3 | Electronic Speed Controllers . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 13 |
|  |  | 3.3.1 | Flight Control Board v2.1 . . . . . . . . . . . . . . . . . . . . . . . . . . . | 14 |
|  |  | 3.3.2 | Transmitter and Receiver . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 14 |
|  |  | 3.3.3 | GPS Module . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 17 |
|  |  | 3.3.5 | Battery/Power Supply . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 18 |
|  | 3.4 | Raspberry Pi 3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 20 |
|  |  |  |  |  |
|  | 3.5 | Software . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 20 |
|  |  | 3.5.1 | Kali Linux . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 20 |
|  |  | 3.5.2 | CleanFlight Flight Controller Software & Configurator . . . . . . . . . . . . . . . . . . . . | 20 |
|  |  | 3.5.3 | Node . Js (AR-drone script),Aircrack-ng . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 23 |
|  |  | 3.5.4 | VNC Server. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 23 |

|  |  |  |  |
| --- | --- | --- | --- |
| 3.6 | Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 24 |
| 4 Experimental Results and Discussion | | | 25 |
| 4.1 | System Setup and Pentesting Testing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 25 |
| 4.2 | Flight Test, hacking and configuration with base-station . . . . . . . . . . . . . . . . . . . . . . . | | 40 |
|  | 4.2.1 | Initial Test Flight . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 40 |
| 4.3 | Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 42 |
|  | 4.3.1 | Diﬃculties . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 42 |
|  | 4.3.2 | Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 43 |
| 4.4 | Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 44 |
| 5 Conclusions | | | 45 |
| References | |  | 47 |

Glossary

9DOF 9 Degrees of freedom sensor board that has a 3-axis gyroscope, 3-axis accelerometer, and 3-axis magnetometer.

Accelerometer A sensor that measures acceleration.

AM Amplitude modulation. A technique used to strengthen radio waves.

Arduino Mega A development based board based on the ATMega microcontroller family.

Arduino Uno A development based board based on the ATMega microcontroller family.

Barometer A sensor that measures air pressure.

Baud rate The maximum number of signal changes allowed in a second. I can be interpreted as symbols per second, where each symbols can transmit one or more bits.

Bluetooth Standardized technology used to transfer data wirelessly.

BoB Break out board. BoB is a electrical board made specifically for easier access to the pins of a electrical component.

DCM Direction cosine matrix.

DSM, DSM2 Digital spectrum modulation. An application of DSSS.

DSSS Direct sequencing spread spectrum. A technique used to avoid radio channels that can cause inteference.

ESC Electronic speed controller. A ESC is an electric circuit that controls the speed of a motor.

FM Frequency modulation. A technique used to modulate radio waves to transmit information.

GPS Global Positioning System. A GPS receiver acquires information from multiple satellites to calculate its current position.

Gyroscope A sensor that measures orientation.

GUIC Globally unique identification code. A unique identification code that is assigned to every radio transmitter.

IDE Integrated Development Environment.

ISM Industrial, scientific, and medical radio bands. A radio band reserved specifically for industrial, scientific, and medical purposes.

LiPo Lithium polymer battery.

Magnetometer A sensor that measures the strength of magnetic fields.

NiCad Nickel Cadmium battery.

NiMH Nickel-Metal Hydride battery.

NMEA National Marine Electronics Association. Repsponsible for the GPS statndard sen-tences.

RC Radio Control

RF Radio frequency.

Roll The roll axis lies along the aircraft centerline. Movevement in a particular direction by turning on this axis.

Rotor (motor) Rotating part of the motor.

Rotor (wing) Rotary wing of a rotorcraft, helicopter, quadcopter, hexacopter.

RTF An acronym for Ready-to-Fly.

RPM Revolutions per minute. Unit to measure speed of a motor.

SD card Secure Digital memory device. Non-volatile meory card.

MIT Massachusetts Institute of Technology.

PID controller Proportional-Integral-Diﬀerential controller.

Pitch The pitch axis is perpendicular to the aircraft centerline. A pitch motion is an up or down movement of the nose of the aircraft.

PWM Pulse width modulation. A technique used to change the pulse width of a signal.

Stator Stationary part of a motor.

SPI Serial Peripheral Interface Bus. SPI is a full duplex synchronized serial bus.

UAV Unmanned aerial Vehicle. An aerial vehicle that with no onboard human controller.

UTC Coordinated universal time.

WiFi Standardized technology used to transfer signals wirelessy.

Yaw Movement about the vertical axis of an aircraft.

## 

## Project Description

**1. Introduction**

The Drone basically a Quadcopter which consists of 4 brushless motors, a rigid frame, an ESC, Flight controller, transmitter and receiver , Raspberry Pi, FPV Camera and most importantly Kali Linux Operating System specially designed for Raspberry Pi 3 having latest kernel RE4SON’S upgrade which is based on Linux kernel version 4.2.Due to this its efficiency is increased by 40%. The Raspberry Pi with the help Kali Linux is converted to a portable hacking machine so that when it gets connected with Quadcopter it turns out to be Swiss Army Knife which can eliminate the enemies by intruding into their area.

A quadcopter is an aerial vehicle that uses four rotors for lift, steering, and stabilization. Unlike other aerial vehicles, the quadcopter can achieve vertical flight in a more stable condition. The quadcopter is not aﬀected by the torque issues that a helicopter experiences due to the main rotor. Furthermore, due to the quadcopter’s cyclic design, it is easier to construct and maintain [2]. As the technology becomes more advanced and more accessible to the public, many engineers and researchers have started designing and implementing quadcopters for diﬀerent uses [3].

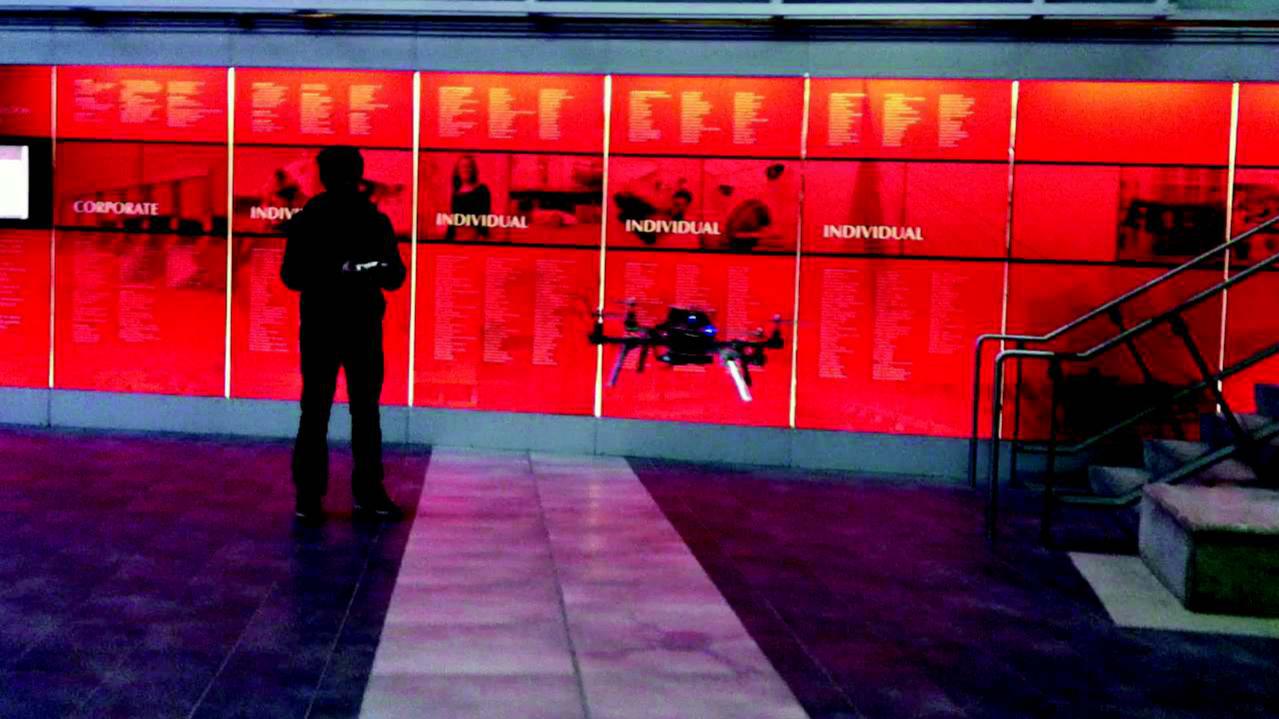


Figure 1.1: Flying Quadcopter.

Various groups such as the military, engineers, researchers, and hobbyists have been developing quadcopters to understand diﬀerent technical areas. For example, quadcopters can be used for re-connaissance and collecting data. This could range from searching for survival victims in a disaster area to checking the state of electrical power lines. Some quadcopters in production today can hold light payloads, such as food and medical supplies, and deliver them to areas where normal planes cannot reach [4]. Many amateur radio operators have designed and built their own multi-copters. Universities, such as MIT, have been studying and doing research for the quadcopter over the past couple of years [5]. This would be a great opportunity for students of the Chandigarh University to join this area of study.

**2. Background**

This section includes the general background information on an unmanned aerial vehicle (UAV) quadcopters. This section also covers our goals and specifications for the project undertaken.

**2.1 UAV Quadcopter**

A UAV quadcopter is an unmanned aerial vehicle with four rotating rotors used for lift and movement. It uses an electronic control system and electronic sensors to help stabilize itself. Quadcopter parts have been decreasing in price over the past couple of years due to technological advances. As a result more hobbyists, universities, and industries are taking advantage of this opportunity to design and develop applications for the quadcopter.

**2.1.1 Flight Control**

A quadcopter consists of four motors evenly distributed along the quadcopter frame as can be seen in figure 2.1 below. The circles represent the spinning rotors of the quadcopter and the arrows represent the rotation direction. Motors one and three rotate in a clockwise direction using pusher rotors. Motor two and four rotate in a counter-clockwise direction using puller rotors. Each motor produces a thrust and torque about the center of the quadcopter. Due to the opposite spinning di-rections of the motors, the net torque about the center of the quadcopter is ideally zero, producing zero angular acceleration. This eliminates the need for yaw stabilization.

A vertical force is created by increasing the speed of all the motors by the same amount of throt-tle. As the vertical forces overcome the gravitational forces of the earth, the quadcopter begins to rise in altitude. Figure 2.2 shows the vertical movement of the quadcopter. As above, the circles represent the spinning rotors, the larger arrows represent the direction the rotors are spinning, and the black arrows represent the forces caused by the spinning rotors.

Pitch is provided by increasing (or decreasing) the speed of the front or rear motors. This causes the quadcopter to turn along the x axis. The overall vertical thrust is the same as hovering due to the left and right motors; hence only pitch angle acceleration is changed. Figure 2.3 shows an example of pitch movement of a quadcopter. As the front motor slows down, the forces created by the corresponding rotor are less then the forces created by the back rotor. These forces are rep-resented by the blue arrows. These forces cause the quadcopter to tip forward and this movement is represented by the red arrow.

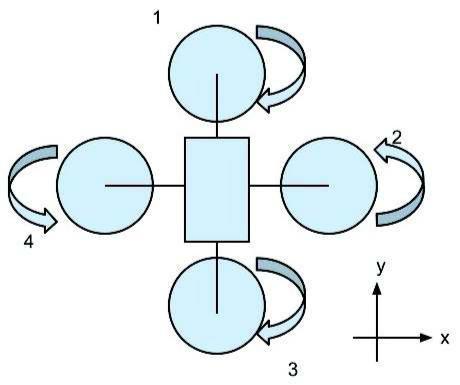


Figure 2.1: Quadcopter: Motor rotation directions.

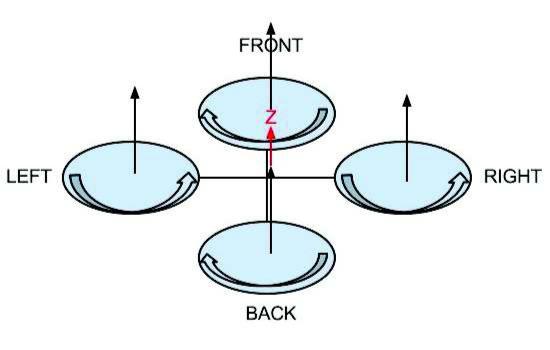


Figure 2.2: Quadcopter: Vertical thrust movement.

Roll is provided by increasing (or decreasing) the speed of the left rotor speed and right motors. This causes the quadcopter to turn along the y axis. The overall vertical thrust is the same as hovering due to the front and back motors; hence only roll angle acceleration is changed. Figure 2.4 shows an example of roll movement of a quadcopter. As the right motor slows down, the forces created by the corresponding rotor are less then the forces created by the left rotor. These forces are represented by the blue arrows. This causes the quadcopter to tip to the right and this movement is represented by the red arrow.

Yaw is provided by increasing (or decreasing) the speed of the front and rear motors or by increasing (or decreasing) the speed of the left and right motors. This causes the quadcopter to turn along its vertical axis in the direction of the stronger spinning rotors. Figure 2.5 shows an example of yaw movement of a quadcopter. As the front and back motor slows down, the forces created by the corresponding rotors are less then the forces created by the left and right rotors. The quadcopter will begin to rotate in the same direction as the faster spinning rotors due to the diﬀerencen in torque forces. This movement is represented by the red arrow.

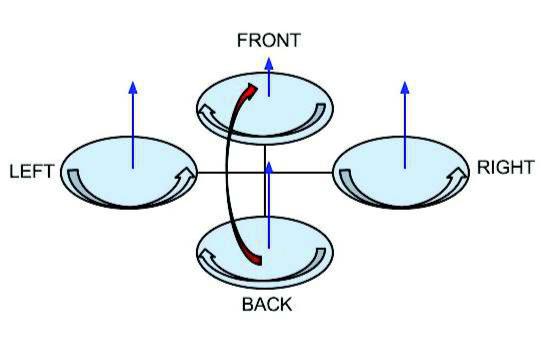


Figure 2.3: Quadcopter: Pitch movement.

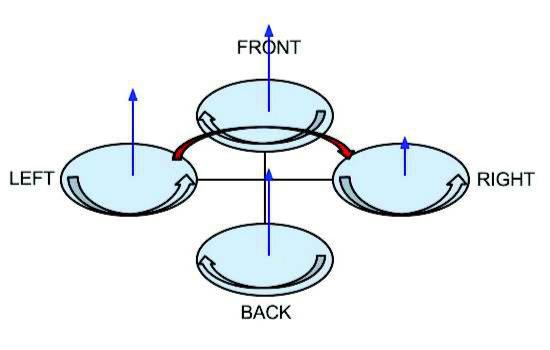


Figure 2.4: Quadcopter: Roll movement.

**2.1.2 Advantages of a Quadcopter**

There are many advantages to quadcopters compared to other aircrafts. A quadcopter does not require a large area to obtain lift, like a fixed wing aircraft does. The quadcopter creates thrust with four evenly distributed motors along its frame. A helicopter suﬀers from torque issue due to its main rotor. The design of the quadcopter does not suﬀer from the same torque issues as the helicopter. The counter balancing forces of the spinning motors cancel out the torque forces caused by each motor causing the quadcopter to balance itself. Because the quadcopter uses four rotors instead of one main rotor, it requires less kinetic energy per rotor for the same amount of thrust when compared to the helicopter. Due to this and its symmetrical design, a quadcopter maintenance and manufacturing costs are relatively lower then other aircrafts [6].

**2.2 Motivation**

When we had all individually registered for this course none of us were quite sure what we planned to design going into this. But we all knew we wanted something interesting, and importantly something fun. We had seen previous projects, the home security systems the rovers lots of interesting ideas but none of them really clicked. When the basis of a copter was suggested we began to build off that, just a drone on its own was not enough, we needed more This project created a platform to learn about unmanned aerial vehicles such as a quadcopter. This expands the scope of the computer engineering education to include the control and the understanding of the mechanical components. The quadcopter has many applications that we are interested to develop like mapping and reconnaissance especially in a defense and dangerous area. It also open up the possibilities to broaden the understanding and applications of control systems, stabilization, artificial intelligence and computer image processing as it applies to the quadcopter.

**2.3 Specifications and Goals**

The goal of our project is to design, implement, and test a stable flying UAV quadcopter that can be used to Penetrate all the networks that are in the range, it also has ability to hack other quadcopter or devices .It also have of self destruction .It collect and save Global Positioning System (GPS) data. Our plan was to make a quadcopter and by adding the required components to give the quadcopter the capabilities to gather and log data autonomously or many more features are added to it. If this goal is accomplished, our team would also like to design and implement some autonomous weapon by which it is all in one weapon for wars.

The final quadcopter design had to meet the following specifications:

1. The quadcopter must be capable of flying and having tremendous speed.
2. The quadcopter must be capable of determining its current location using GPS data.
3. The quadcopter must be capable of hacking.
4. The quadcopter must be able to perform hack other drones by its own by simply typing :
   * air-node.

**2.4 Summary**

The aim of this project was to build a autonomous quadcopter that can be used hack other system by hovering it. This project was a great learning opportunity for us to apply our engineering knowledge. The thesis addresses the diﬃculties our group faced throughout the project.. It also open up the possibilities to broaden the understanding and applications of control systems, stabilization, artificial intelligence and computer image processing as it applies to the quadcopter.

**3. System Design**

This section explains in details how the project was approached. All hardware and software components that will be used will be explained in this section of the report. Problems encountered and solutions to these problems will not be mentioned in this section of the report.

**3.1 Project Overview**

The ultimate goal of this project is to design a UAV quadcopter that is able to store and log information. This project could be used in many applications, ranging from checking the condition of power lines to searching for survival victims in a disaster area. The end result of this project could be used for these applications in future developments in this field.

Due to the time constraints of our project, it was not possible to design and build a quadopter from scratch. Therefore, it was decided that our group would purchase an existing quadcopter kit and interface the necessary components to fly, stablize and log data. This project was split into four main stages; the research stage, the building and interfacing stage, the tuning and calibration stage, and the programming stage. Each stage will be explained more thoroughly in the following sections of this document.

**3.1.1 The Research Stage**

The research stage was a critical stage that provided our team with the knowledge necessary to complete the other stages of our project. This stage was an ongoing process that our team had to return to many times during the development process to gain the knowledge needed to continue on with the project. Our research encompassed a wide range of sources, which included studies done at diﬀerent universities and hobby enthusiast sources. Our reseach included the aerodynamics of the quadcopter, theory and principle of each quadcopter component, and how the Aeroquad soft-ware [7] should be modified to fit the purposes of our project.

3.1.2 Building & Interfacing Stage

This stage started when the ordered parts started arriving. During this stage we focused on verifying and testing each component thoroughly. The testing process will be explained in greater detail in a later section in this document. After each part was verified to be working correctly, we combined the components together. The frame and control board were assembled, the motors and electronic speed controllers (ESC) were mounted, and the GPS module, SD card adapter, and the transmitter and receiver were interfaced. The next step was to tune and calibrate the quadcopter.

**3.1.3 The Tuning/Calibration Stage**

During this stage the quadcopter proportional-integral-derivative (PID) control system was tuned. A tuning stand was built to mount the quadcopter on to help us tune the PID system of the quadcopter. This was a tedious process. After some initial tuning, the quadcopter was ready for its first flight test. The PID testing process and results of the flight test will be explained in greater detail later section in this document.

**3.1.4 The Programming Stage**

In this stage of our project we were to design four diﬀerent commands for our quadcopter. The preliminary designs for these commands have been drawn out and will be explained in greater detail in a later section of this document. Unfortunately we did not get very far in this stage and programming for these commands has yet to be started.

**3.3 Physical Components**

This subsection describes all the physical components of our project and how they work. As well as the reasoning for choosing the GPS module, SD card, battery, and transmitter and receiver, Raspberry pi, wireless Adapter, connecting wire, connectors etc.

**3.3.2 Electric Motors**



# Figure 3.1: Edge Racing 2204 2300KV FPV Motor

There are two types of motors, brushless and brushed. A basic brushless motor has three electromagnetic windings separated evenly on the stationary part of the motor, called a stator. Permanent magnets are located on the rotating part of the motor, called a rotor. The rotor will begin to rotate when two of the three windings are supplied with a voltage, creating a magnetic field. The magnetic field created by the windings pushes and pull against the magnetic field cre-ated by the magnets on the rotor. When the magnetic fields are aligned, two diﬀerent windings are driven and a new magnetic field is created causing the rotor to continue to turn. A controller is used to determine the current rotor position relative to the windings and drive the necessary windings in certain sequence to turn the rotor in the proper direction. The rotor will turn in the opposite direction if the sequence is reversed.

A brushed motor works in a similar fashion. A brushed motor has the opposite orientation of brushless motor. The electromagnetic windings are located on the rotor and the magnets are located on the stator. Like a brushless motor, the rotor on a brushed motor turns due to the attractive and repulsive forces of the two magnetic fields created by the windings and the magnets. The main diﬀerence between these two motors is that a brushed motor uses a mechanical switch to change the polarity of the windings to turn the rotor. A brushed motor is more susceptible to wear and tear because it uses a mechanical mechanism to check and change the polarity of its windings. It is also limited in speed due to the same reason [8].

**3.4 Electronic Speed Controllers**

These ESC can be seen in figure 3.3. These ESC can push 30 amperes of current continuously, can push currents up to 40 amperes for ten seconds based on voltage output, and handle speeds up to 210,000 rotations per minute (RPM). One ESC was found to be malfunctioning during testing and extra ESCs were purchased. We will go into more details about this later in the document.

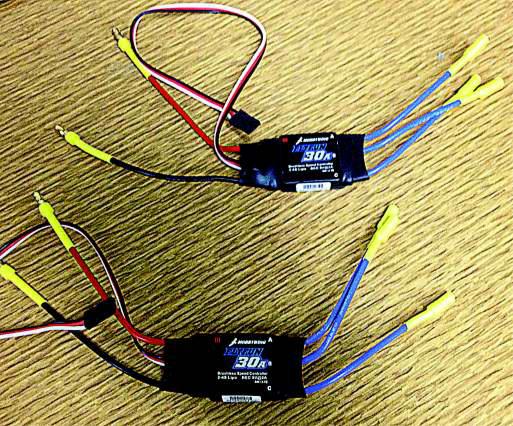


Figure 3.2: 30A DYS brushless ESC

An electronic speed controller is an electrical circuit that controls the speed of an electric motor and the direction a motor rotates. A motor turns because of the magnetic forces created by the windings and the magnets within the motor.

For a brushless motor, the speed of the motor will depend on the frequency of the winding drive sequence. On a basic brushless motor, there are three windings that are controlled using pulse width modulated (PWM) signals. Two windings will be driven at a time to create the necessary magnetic forces to turn the rotor. The greater the frequency sent to the motors, the faster the rotor will turn due to the magnetic forces. The frequency of the signals is adjusted by changing the pulse width of the signal. Smaller pulse widths will increase the frequency of a PWM sig-nal because more pulses can be sent to the windings in the same time duration, and vice versa for large pulse widths. A brushed ESC works in the same manner but only two control signals are used.

# **3.3.1** [**Naze 32 Revision 6 Flight Controller**](http://www.dronetrest.com/t/naze-32-revision-6-flight-controller-guide/1605)

[**Guides**](http://www.dronetrest.com/c/guides)

The [Naze 32929](http://www.unmannedtechshop.co.uk/naze-32/) Rev 6 board has made some nice layout revisions such as finnaly having the USB board mounted on t he side for easy access on your miniquad al last!

## Whats new in Revision 6?

* USB port has been moved to the right of the board so you dont need to mount the board sideways anymore!
* IMU sensors upgraded to MPU 6500 (Rev 5 uses older MPU 6050)
* Receiver connectors are no longer solder pads but common pin hole headers style. This means that you no longer need to directly solder your receiver wires to the flight controller. Also older revisions had issues of pads delaminating when board got too hot.
* Fully pin compatible with Rev 5 boards for accesories such as OSD.
* Integrated SBUS inverter, now you can connect your FRsky SBUS receivers directly with no extra inverter!
* Dedicated Spectrum Satellite port.
* Additional Flash memory so you can use blackbox recording directly. With the Acro version you get 2Mb (16Mbit) and the Full version you get 16Mb (128Mbit). To learn more check our guide on [Naze 32 versions](http://www.dronetrest.com/t/naze-32-versions-explained-and-what-to-look-out-for/1580).

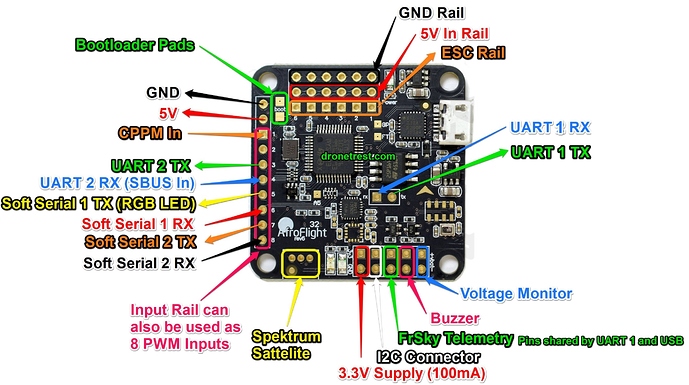
[[](http://www.dronetrest.com/uploads/db5290/original/2X/9/977a2a3924cfb004ccd942228e96e4657a292b5b.jpg)](http://www.dronetrest.com/uploads/db5290/original/2X/9/977a2a3924cfb004ccd942228e96e4657a292b5b.jpg)

Figure 3.3: Naze 32 versionsC

Here is a closeup of the auxillary connectors on the Naze 32 board which include an extra 3.3V output at 100mA, a I2C connector, the buzzer connector, and battery voltage monitor pins. One thing to note is that the Frsky Telemetry pins are shared by the UART 1 TX, which is in turn shared by the USB. So if your board is connected to your PC via USB you cannot use the UART 1 ports.

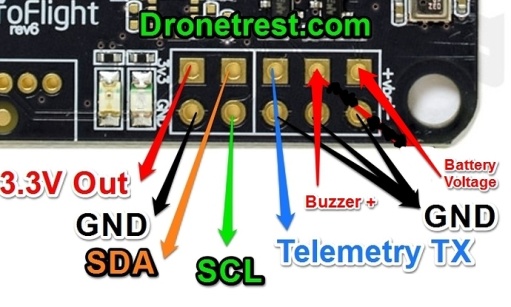
[[](http://www.dronetrest.com/uploads/db5290/original/2X/1/1bf4b386894eabb585290a2865ac89dd9239d86f.jpg)](http://www.dronetrest.com/uploads/db5290/original/2X/1/1bf4b386894eabb585290a2865ac89dd9239d86f.jpg)

Figure 3.4: UART

## Typical Naze32 Connection diagram

The diagram below shows a fairly common setup on a FPV racing quadcopter using a CPPM receiver, optional GPS module, and an RGB LED bar.

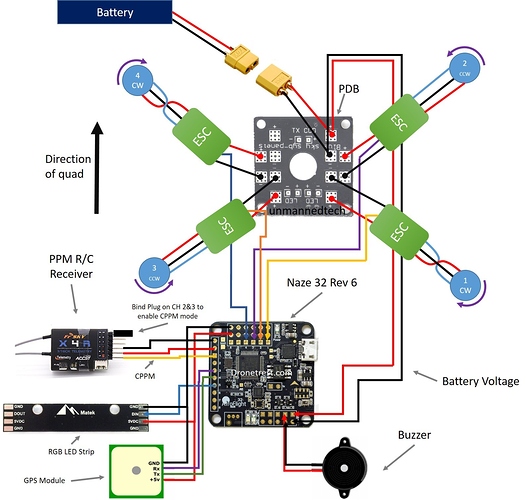
[](http://www.dronetrest.com/uploads/db5290/original/2X/1/13615e836e576d74c578c5da74e82a5b1035dd6f.jpg" \o "Naze-32-Connection-Diagram-Dronetrest.jpg)

Figure 3.5: NAZE32

## Flashing Cleanflight onto Naze 32 Rev 6

Contrary to many forums it is definitely possible to load and run cleanflight easily on the Rev 6 Naze 32 boards that we sell. Once the board is connected to your PC the drivers should install automatically. Then launch the [cleanflight GUI1.8k](https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en) and write the latest firmware version to your board as shown below.

[[](http://www.dronetrest.com/uploads/db5290/original/2X/b/b9343d23d6192fe1b4a5233cc7e3dbf1eaeda9d0.jpg)](http://www.dronetrest.com/uploads/db5290/original/2X/b/b9343d23d6192fe1b4a5233cc7e3dbf1eaeda9d0.jpg" \o "Firmware-Flash-naze32.jpg)

Figure 3.6: NAZE32 Rev6

## Connecting a RGB LED strip

Adding an [RGB LED strip537](http://www.unmannedtechshop.co.uk/matek-ws2812b-led-and-5v-buzzer-6-rgb-chips-led-indicator-for-naze32-skyline32/) to your Naze 32 is one of the coolest features for FPV flying. Not only does it let you see your quadcopter easily, but you can also add smart features to your LED, so when you are turning left, the Left LED's can flash. When you slow down the lights can glow bright red, just like a car brake lights.

Depending on your RGB LED strip power requirements you might need to connect it directly your battery, or BEC. In the diagram below the LED stip runs on 5V so we can power it directly from the ESC rail. The control signal plugs into RC 5 on your naze32.

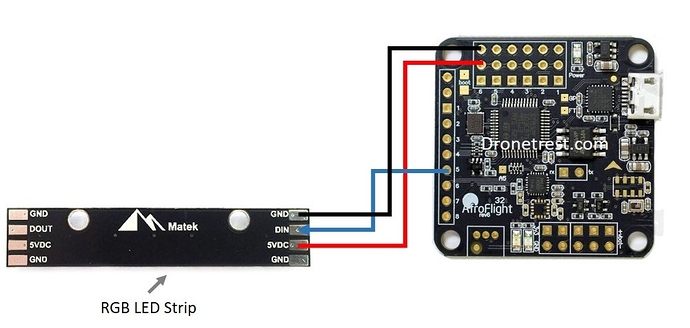
[[](http://www.dronetrest.com/uploads/db5290/original/2X/7/7f52f1c40ddf8e778615b65ddd8711832677fad0.jpg)](http://www.dronetrest.com/uploads/db5290/original/2X/7/7f52f1c40ddf8e778615b65ddd8711832677fad0.jpg" \o "Naze32-RGB-LED-connection.jpg)

Figure 3.7: REG LED Strip

Once connected you will need to configure the RGB LED strip in clenaflight. Here is a great video showing you how to do this

**3.3.2 Transmitter and Receiver**

Our group came up with three diﬀerent options for wireless communication system for our quadcopter. There are three main options to choose from which include a radio transmitter and receiver, a WiFi module, or a Bluetooth module. For our purposes we would need at least six channels to control our quadcopter. The six channels correspond to throttle, roll, yaw, pitch, one channel to switch between acrobatic mode and stable mode, and one or more channels for our auto-commands. More details of the of the two diﬀerent modes will be discussed in a the PID tuning section. Below is a list of the options we looked at.

Spektrum DX7S transmitter and AR800 receiver

* Price: $334.00
* Support up to 7 channels at 2.4 GHz
* Range up to 305 meters (1000 feet)
* Support DSM2
* Easy to interface

Sparkfun WiFi module

* Price: $95.00
* Smartphone compatibility
* Support up to 8 channels
* Range only up to 50 meters

BLTE Arduino Bluetooth shield

* Price: $45.00
* Bluetooth 4.0
* Low power consumption
* Range only up to 10 meters

Our group decided to choose the Spektrum DX7S transmitter and the AR800 receiver because it had the required amount of channels needed for our quadcopter purposes. The transmitter and receiver also had the greatest range of transmission was also easy to interface with our control board. If we were to interface the other WiFi module or the Bluetooth module, separate channels need to be programmed and large amounts of programming would be needed. Therefore the Spek-trum transmitter and receiver was chosen and shown in figure 3.5.

The Spektrum DX7S transmitter and receiver were properly interfaced with the quadcopter controller board. Some diﬃculties encountered during interfacing were that both the controller board and receiver used male pins. The quadcopter kit that was purchased did not come with the necessary cables to connect these two components together. Female-to-female servo cables were

purchased to solve this problem.



Figure 3.8: Spektrum DX7S receiver(left) and transmitter(right).

Wireless telecommunication is the transfer of information between two or more points that are not physically connected. Radio is the transmission of signals through free space by electromagnetic waves. The frequency range in radio is from 3 kHz to 300 GHz [9]. There are diﬀerent techniques used in electronic communication; which include Amplitude modulation (AM), frequency modula-tion (FM), DSSS, DSM, and DSM2. AM works by varying the strength of the transmitted signal in relation to the information being sent. In the other hand, FM works by varying the frequencies of the transmitted signal.

In the past, radio technologies like FM are widely used in radio control (RC). The 27 MHz to 49 MHz frequencies bands are designated by law for use with RC airplanes and aircraft. The designated frequencies for RC planes or aircraft fall into the 72 MHz band and each separate fre-quency has been given a unique channel number. The dedicated remote control frequencies avoid interference by unlicensed use [10]. Frequency checker is used to make sure no one is occupying the same channels. If there are two people using the same channel to control the aircraft, interference would occur and aircraft might crash.

In RC control the latest technology to avoid interference of signal occurs is Direct Sequencing Spread Spectrum (DSSS). DSSS operates within the 2.4GHz frequency band. This new technology was a form of secure radio signal transmission. DSM and DSM2 are applications of DSSS. A Glob-ally Unique Identification Code (GUIC) is assigned to every radio transmitter during manufacture. The receiver is programmed to identify that unique in what is known as the ”‘binding process”’,

and so the transmitter and receiver lock together with the same code which will block out other code [11]. The binding process occurs a couple of seconds every time the system is powered up. Once the transmitter and receiver locked together, the transmitted signal is spread out over a wide band before being identified, collected and re-assembled by the receiver

**3.3.3 GPS Module**

Our quadcopter needed a GPS module that was small, fast and accurate, consume as little power as possible, and was easy to interface with the existing control board. We looked at three diﬀerent GPS receivers and chose the best one that would meet our requirements. Table 3.6 shows a list of features we looked at when comparing the GPS modules.

Table 3.1: GPS comparision table.



We initially chose the GS407 Helical GPS receiver because it was the smallest, lightest, and most accurate module. Although it did not have the fastest baud rate, we thought that it was quick enough for the purposes we were using it for. Another reason why we chose the GS407 receiver over the others was because it came with a break out board that was specifically made to directly connect to the Aeroquad control board. This allowed for easy interfacing between the two components. Un-fortunately we ran into a few problems that we will mention in further detail later in this document.

The second GPS receiver module purchased was the Parallax PMB-648 receiver using the SiRF-starIII chipset. Although this module was slower and heavier than the GS407 module, the PMB-648 receiver cost less and was readily available at time of purchase. The module was also easy to in-terface because it came with a connector with wires already connected to the module instead of a fragile break out board.

A GPS is a satellite network operated by the United States of America Department of Defense. This network of satellites transmits data about its current location and time. A GPS receiver passively retrieves this data from multiple satellites to estimate its position. By estimating the distance between more than three satellites, a GPS receiver can determine its current position in

**3.4.4 Battery/Power Supply**

For our project we need a power supply that is low cost, light weight, reusable, and has enough power for at least ten minutes of flight. Rechargeable batteries were chosen for our project due to reuse value. Currently there are three main types of rechargeable batteries available commercially for radio controlled models, nickel-cadmium (NiCad), nickel-metal hydride (NiMH), and lithium polymer (LiPo) batteries.

NiCad batteries have a low internal resistance that allows for high-power output, can operate a large temperature range, but suﬀers from “memory” loss. This term memory refers to the amount of capacity the battery can store after each discharge. The overall capacity of the NiCad battery will decrease over a duration of time. NiMH batteries are similar to NiCad batteries except they can hold 30% more capacity, but suﬀer from a larger memory loss [12].

LiPo batteries can hold 30% more capacity and are much lighter than a NiMH battery. LiPo batteries also suﬀer from a lower memory loss compared to the NiMH battery. The disadvantages to this battery are that these type of batteries are prone to overheating and overcharging the bat-teries could lead to fire. Extreme care must be taken when using these type of batteries [12].

Due to these reasons, our group has chosen to use a LiPo battery. At this point and time we had just started choosing parts for our quadcopter. To choose the size of battery we had to make some calculations. All calculations and measurements were based on datasheet specifications. To calculate the amount of thrust we needed to overcome gravity, the overall weight of the quadcopter and all its components was found to be about 1550 grams. We may want to add extra components in the future, so we made these calculations with that in mind and assumed the total weight to be 1800 grams. Our quadcopter has four motors, therefore 450 grams of thrust was needed for each motor to overcome the forces of gravity.

Using the data sheets we estimated the amount of power needed to run the motors and other components to be about 70 watts. Using the following equation,

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| I = | P | = |  | 70W | = 6.3A. | (3.1) |
| V | 11.1V | |
|  |  |  |  |

where I is current in amperes, P is power in watts, and V is voltage in volts. We calculated the amount of current needed for a 3 celled LiPo battery running at 11.1 volts to be about 6.3 amperes. Multiply this by the amount of motors and we needed about 22.2 amperes to fly the quadcopter. Using Peukert’s Law [13], we can determine the capacity amount needed for a estimated flight duration of about 10 minutes.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | T × Ik | | | 10min × 22.2A | | |  |  |
| C = |  |  | = |  |  |  | = 3.7AH | (3.2) |
| 60 | | 60 | |  |
|  |  |  |  |  |

C is battery capacity measured in amperes-hour, k is Peukert’s constant which we assumed to be 1, and T is time measured in minutes. The calculated capacity per hour was calculated to be 3.7 amperes-hour. We rounded this value up and chose to purchase the Turnigy 3 cell LiPo battery with a capacity of 4000 milliamperes-hour. All these calculations were based on rough estimates, since we did not any components available for testing at the time of purchase of the battery. We over specified the weight to make had enough power to fly ourquadcopter. The estimated weight

was 1800 grams. The final measured weight of the quadcopter kit and all of its components were measured to by 1684 grams. Figure 3.7 shows the battery we chose and the balancer needed to charge the battery safely.



Figure 3.9: Battery and Balancer

**3.4 Raspberry pi**

A Raspberry Pi is a credit card-sized computer originally designed for education, inspired by the 1981 BBC Micro. Creator Eben Upton's goal was to create a low-cost device that would improve programming skills and hardware understanding at the pre-university level. But thanks to its small size and accessible price, it was quickly adopted by tinkerers, makers, and electronics enthusiasts for projects that require more than a basic microcontroller (such as Arduino devices).

The Raspberry Pi is slower than a modern laptop or desktop but is still a complete Linux computer and can provide all the expected abilities that implies, at a low-power consumption level.

## Is the Raspberry Pi open hardware?

The Raspberry Pi is open hardware, with the exception of the primary chip on the Raspberry Pi, the [Broadcomm SoC](http://www.raspberrypi.org/documentation/hardware/raspberrypi/) (System on a Chip), which runs many of the main components of the board–CPU, graphics, memory, the USB controller, etc. Many of the projects made with a Raspberry Pi are open and well-documented as well and are things you can build and modify yourself.

**3.5 Software**

This subsection describes all the software components of our project and how they work.

**3.5.1 Sticky Finger’s Kali-Pi – The pocket size, finger friendly, lean mean hacking machine**

“[Sticky Fingers Kali-Pi](https://whitedome.com.au/re4son/kali-pi/) combines Kali Linux and the Re4son Kernel into a hassle-free package that turns any Raspberry Pi into a powerful Swiss Army knife with finger-friendly touch screen interface that can provide backend support in your own network, or operate autonomously behind enemy lines, posing as a humidity sensor. Out of the box, the Kali-Pi comes preconfigured with all essential services, such as SSH, VNC, FTP, HTTP, and Wi-Fi backdoor, as well as the [MANA toolkit](https://github.com/sensepost/mana), [Snort](https://www.snort.org/), [Kismet](https://www.kismetwireless.net/), [Metasploit](https://metasploit.com/), etc. and a finger-friendly touch screen interface to control it all.”

Built on a Raspberry Pi 0/0W/1/2/3, this setup can be quick-installed from a pre-configured image or built from scratch, and has options for a TFT touch screen with a custom-built intuitive touch menu, bluetooth, injection support, and more. He’s even strapped this puppy onto a drone for some high-flying “research”.

At the heart of this cool project is the [Re4son-Kernel](https://whitedome.com.au/re4son/re4son-kernel/) for Raspberry Pi. Built to work with the Pi 0/0W/1/2/3, the Re4son-Kernel, “allows Kali Linux to get the best out of any Raspberry Pi by supplying support for the onboard wifi and Bluetooth, wifi injection patches, additional hardware support, security enhancements, headers, sources, etc.”

It also provides complete armel support for the Pi 1, Zero, and Zero W and armhf support for the Pi 2 and 3 and all versions are 100% compatible with the stock Kali Linux Kernel.

**3.5.2 CleanFlight**

Official "CleanFlight" Flight Controller Software, Information Thread

Cleanflight is a firmware and GUI set that can be used on multirotor aircraft and fixed-wing aircraft, it supports a variety for shapes and motor counts, and is not limited to quadcopters, hexacopters, octocopters, tricopters and planes. Cleanflight is Open-Source flight controller software which is 32-bit version of the original 8-bit MultiWii code.  
  
Firmware Source

<http://github.com/cleanflight/cleanflight>  
  
GUI Source

<http://github.com/cleanflight/cleanflight-configurator>  
  
Firmware Download

[https://github.com/cleanflight/clean...ree/master/obj](https://github.com/cleanflight/cleanflight/tree/master/obj)  
  
GUI Download

[https://chrome.google.com/webstore/d...ajinjgmkahmfgb](https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb)  
  
Documentation   
[https://github.com/cleanflight/clean...ee/master/docs](https://github.com/cleanflight/cleanflight/tree/master/docs)  
  
  
IRC URL

irc://freenode.net/#cleanflight  
  
IRC Channel

#cleanflight

**3.5.3 Node.JS**

Node.js® is a JavaScript runtime built on [Chrome's V8 JavaScript engine](https://developers.google.com/v8/). Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, [npm](https://www.npmjs.com/), is the largest ecosystem of open source libraries in the world.

#### aircrack-ng

I use [aircrack-ng](http://www.aircrack-ng.org/) to put our wireless device into monitor mode to find our drones and drone owners. I then use aireplay-ng to [deauthenticate](http://www.aircrack-ng.org/doku.php?id=deauthentication) the true owner of the drone I'm targeting. Once deauthenticated, I can connect as the drone is waiting for its owner to reconnect.

#### node-ar-drone

I use [node-ar-drone](https://github.com/felixge/node-ar-drone) to control the newly enslaved drone via Javascript and [node.js](http://nodejs.org/).

**3.5.4 VNC Server VNC (VIRTUAL NETWORK COMPUTING)**

Sometimes it is not convenient to work directly on the Raspberry Pi. Maybe you would like to work on it from another device by remote control.

VNC is a graphical desktop sharing system that allows you to remotely control the desktop interface of one computer (running VNC Server) from another computer or mobile device (running VNC Viewer). VNC Viewer transmits the keyboard and either mouse or touch events to VNC Server, and receives updates to the screen in return.

You will see the desktop of the Raspberry Pi inside a window on your computer or mobile device. You'll be able to control it as though you were working on the Raspberry Pi itself.

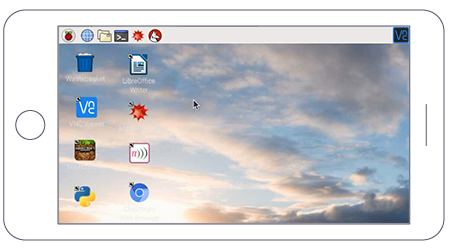


Figure 3.10: desktop of the Raspberry Pi

VNC Connect from RealVNC is included with Raspbian. It consists of both VNC Server, which allows you to control your Raspberry Pi remotely, and VNC Viewer, which allows you to control desktop computers remotely from your Raspberry Pi should you want to.

You must enable VNC Server before you can use it: instructions for this are given below. By default, VNC Server gives you remote access to the graphical desktop that is running on your Raspberry Pi, as though you were sitting in front of it.

However, you can also use VNC Server to gain graphical remote access to your Raspberry Pi if it is headless or not running a graphical desktop. For more information on this, see **Creating a virtual desktop**, further below.

**3.6 Summary**

In this section the hardware and software components were explained in detail. The reasons for choosing each component was made clear.

**4. Experimental Results and Discussion**

In this section of the document we will be discussing the verification and testing of each hard-ware and software component. All problems will be described in detail and the solutions we made to solve these problems. In this section we will also discuss our overall results of the project and what we could have done to improve upon our project. Future work for this project will also be mentioned in this section of the document.

**4.1 System Setup and Penetration Testing**

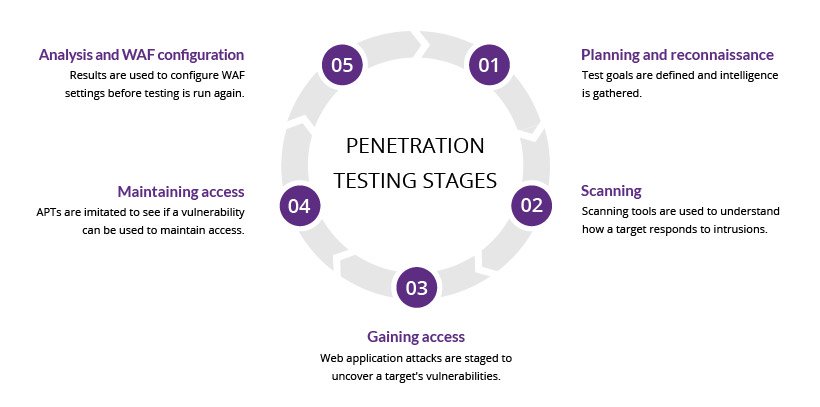
WHAT IS PENETRATION TESTING

A penetration test, also known as a pen test, is a simulated cyberattack against your computer system to check for exploitable vulnerabilities. In the context of web application security, penetration testing is commonly used to augment a [web application firewall (WAF)](https://www.incapsula.com/website-security/web-application-firewall.html).

Pen testing can involve the attempted breaching of any number of application systems, (e.g., application protocol interfaces (APIs), frontend/backend servers) to uncover vulnerabilities, such as unsanitized inputs that are susceptible to code injection attacks.Insights provided by the penetration test can be used to fine-tune your WAF security policies and patch detected vulnerabilities.

PENETRATION TESTING STAGES

The pen testing process can be broken down into five stages.



**Figure 4.1: Testing Stage**

1. **Planning and reconnaissance**

The first stage involves:

* + Defining the scope and goals of a test, including the systems to be addressed and the testing methods to be used.
  + Gathering intelligence (e.g., network and domain names, mail server) to better understand how a target works and its potential vulnerabilities.

1. **Scanning**

The next step is to understand how the target application will respond to various intrusion attempts. This is typically done using:

* + **Static analysis** – Inspecting an application's code to estimate the way it behaves while running. These tools can scan the entirety of the code in a single pass.
  + **Dynamic analysis** – Inspecting an application's code in a running state. This is a more practical way of scanning, as it provides a real-time view into an application's performance.

1. **Gaining access**

This stage uses web application attacks, such as [cross-site scripting](https://www.incapsula.com/web-application-security/cross-site-scripting-xss-attacks.html), [SQL injection](https://www.incapsula.com/web-application-security/sql-injection.html) and [backdoors](https://www.incapsula.com/web-application-security/backdoor-shell-attack.html), to uncover a target's vulnerabilities. Testers then try and exploit these vulnerabilities, typically by escalating privileges, stealing data, intercepting traffic, etc., to understand the damage they can cause.

1. **Maintaining access**

The goal of this stage is to see if the vulnerability can be used to achieve a persistent presence in the exploited system— long enough for a bad actor to gain in-depth access. The idea is to imitate [advanced persistent threats](https://www.incapsula.com/web-application-security/apt-advanced-persistent-threat.html), which often remain in a system for months in order to steal an organization’s most sensitive data.

1. **Analysis**

The results of the penetration test are then compiled into a report detailing:

* + Specific vulnerabilities that were exploited
  + Sensitive data that was accessed
  + The amount of time the pen tester was able to remain in the system undetected

This information is analyzed by security personnel to help configure an enterprise's WAF settings and other application security solutions to patch vulnerabilities and protect against future attacks.

## PENETRATION TESTING METHODS

### EXTERNAL TESTING

External penetration tests target the assets of a company that are visible on the internet, e.g., the web application itself, the company website, and email and domain name servers (DNS). The goal is to gain access and extract valuable data.

INTERNAL TESTING

In an internal test, a tester with access to an application behind its firewall simulates an attack by a malicious insider. This isn't necessarily simulating a rogue employee. A common starting scenario can be an employee whose credentials were stolen due to a [phishing attack](https://www.incapsula.com/web-application-security/phishing-attack-scam.html).

BLIND TESTING

In a blind test, a tester is only given the name of the enterprise that's being targeted. This gives security personnel a real-time look into how an actual application assault would take place.

DOUBLE BLIND TESTING

In a double blind test, security personnel have no prior knowledge of the simulated attack. As in the real world, they won't have any time to shore up their defenses before an attempted breach.

TARGETED TESTING

In this scenario, both the tester and security personnel work together and keep each other appraised of their movements. This is a valuable training exercise that provides a security team with real-time feedback from a hacker's point of view.

## PENETRATION TESTING AND WEB APPLICATION FIREWALLS

Penetration testing and WAFs are exclusive, yet mutually beneficial security measures.

For many kinds of pen testing (with the exception of blind and double blind tests), the tester is likely to use WAF data, such as logs, to locate and exploit an application's weak spots.

In turn, WAF administrators can benefit from pen testing data. After a test is completed, WAF configurations can be updated to secure against the weak spots discovered in the test.

Finally, pen testing satisfies some of the compliance requirements for security auditing procedures, including [PCI DSS](https://www.incapsula.com/web-application-security/pci-dss-certification.html) and [SOC 2](https://www.incapsula.com/web-application-security/soc-2-compliance.html). Certain standards, such as PCI-DSS 6.6, can be satisfied only through the use of a certified WAF. Doing so, however, doesn’t make pen testing any less useful due to its aforementioned benefits and ability to improve on WAF configurations.

**4.1.3 Naze32 FC**

A flight controller (a.k.a FC) is the brain of the aircraft, which is basically a circuit board that has built-in sensors that detects orientation changes. It also receives user commands, and controls the motors in order to keep the quadcopter in the air.

Nearly all flight controllers have basic sensors such as Gyro (Gyroscopes) and Acc (Accelerometer). Some FC might include more advanced sensors such as Barometer (barometric pressure sensors) and magnetometer (compass).

Flight controller is also a hub for many other peripherals, such as GPS, LED, Sonar sensor etc.

Flight controllers for quadcopters are rapidly evolving: smaller, using better processors and hardware and getting more and more features integrated.

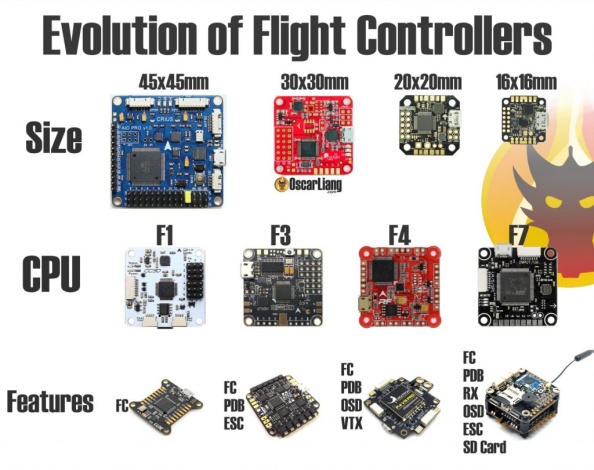
[](https://oscarliang.com/ctt/uploads/2017/04/fc-flight-controller-evolution-size-processor-features-mini-quad-racing-drone.jpg)

Figure 4.2: Fight Controller

# FC Firmware

Apart from the difference in hardware, they might also use different firmware that are specialized in different applications.

Modern FC firmware normally can be configured via software on a computer or smartphone. “Tuning” is the term we use in the multirotor hobby when we change PID, rates and other configuration settings, and it changes how the multirotor performs. The GUI and parameters are different from firmware to firmware, so there is certain level of learning curve getting into each of them.

[](https://oscarliang.com/ctt/uploads/2017/05/fc-firmware-list-mini-quad-rc-qaducotper-fpv-racing-drone.jpg)

Figure 4.3: FC Firmware

Here is a [list of popular FC firmware](https://oscarliang.com/mini-quad-fc-firmware/) available for mini quad. If you are totally have no clue which one to choose, my recommendation would be Betaflight, Raceflight and KISS. These all have excellent flight performance and you cannot go wrong with any of them.

Once you have picked a flight controller firmware, you can then look at what controller boards are compatible.

### Processor

Currently, you have 4 main types of CPU to choose from: F1, F3, F4 and F7. We recommend getting a F3 or F4 FC for now, as we have reached the limit of F1, and [F7 FC](http://intofpv.com/t-vr-race-f7-fc-first-ever-f7-flight-controller) are still new and needs time to be improved.

Table 4.1: CPU type

|  |  |  |  |
| --- | --- | --- | --- |
| **F1** | **F3** | **F4** | **F7** |
| 72MHz | 72MHz | 168MHz | 216MHz |

To learn more about the [differences between F1, F3, F4 and F7 processors make sure to check out this article](https://oscarliang.com/f1-f3-f4-flight-controller/).

[](https://oscarliang.com/ctt/uploads/2017/05/flight-controller-fc-proccessor-stm32-f1-f3-f4-f7-cpu-chip.jpg)

Figure 4.4: Processor

From left to right: STM32 F1, F3, F4

### Number of UART’s and Inverters

UART is the hardware serial port that allows you to connect external devices such as digital receiver (SBUS and Spektrum), OSD, SmartPort Telemetry, Race Transponder, VTX control etc. Each UART has 2 pins, TX is for transmitting data and RX for receiving data.

The UART count depends on the type of processor and the board design, for example, F1 FC normally has 2 UART’s, while F3 and F4 can have between 3 to 5 and F7 can have 8 or more.

UART stands for **Universal Asynchronous Receiver/Transmitter**.

Table 4.2: CPU type

|  |  |  |  |
| --- | --- | --- | --- |
| **F1** | **F3** | **F4** | **F7** |
| 2 UART’s | 3-5 UART’s | 3-5 UART’s | 8+ UART’s |

Because for some reason, Frsky SBUS and SmartPort signals are inverted, and therefore it requires an inverter in the UART’s to read these signals.

F1 and F4 FC normally have none or very limited number of built-in inverters, and Frsky radio users could run into problems with these boards. There are workaround for this, e.g. using external inverter, or [getting uninverted signal from the RX](https://oscarliang.com/uninverted-sbus-smart-port-frsky-receivers/). F3 and F7 do not have this problem and all of their UART’s have built-in inverters, so they work out of the box.

However in the case of running out of hardware serial ports, you can use the feature “soft-serial” which is a way of emulating UART port using software. The downside of this is the lower baud rate (update rate), and putting more load on your CPU. Therefore this is not recommended for timing critical tasks such as your receiver.

### Gyro: the type of IMU and BUS

The IMU sensor used on a flight controller can be found in the following list. The job of an IMU sensor is to measure the quadcopter’s movement and orientation.

Table 4.3: CPU type

|  |  |  |
| --- | --- | --- |
| IMU | Possible Communication Protocol (BUS) | Max. Effective Gyro Sampling Frequency |
| MPU6000 | SPI, i2c | 8K |
| MPU6050 | i2c | 4K |
| MPU6500 | SPI, i2c | 32K |
| MPU9150 | i2c | 4K |
| MPU9250 | SPI, i2c | 32K |
| ICM20602 | SPI, i2c | 32K |
| ICM20689 | SPI, i2c | 32K |

There are two main properties of IMU we need to consider in a flight controller: max sampling rate, and how susceptible to noise they are (both electrical and mechanical noise).

Currently the most widely used IMU is the **MPU6000** as it supports up to 8KHz sampling rate, and proven to be one of the most rubust IMU against noise. The general consensus is to avoid MPU’s including **MPU6500** and **MPU9250** which are noisier despite the higher sampling speed.

[Soft mounting your FC](https://oscarliang.com/soft-mounting-fc-motors/) and [adding low ESR capacitor to your flight controller](https://oscarliang.com/capacitors-mini-quad/) might help reduce noise getting to Gyro.

**SPI** and **i2c** are the types of “BUS”, or communication protocol used to connect the IMU sensor to the processor. It can have a significant impact on the effective sampling rate and therefore the maximum flight controller looptime. The preferred BUS is **SPI**, which allows you to run Gyro at a much higher refresh rate than **I2C** which has a limit of 4KHz.

Note that **MPU9150** is effectively the MPU6050 with integrated AK8975 magnetometer, while the **MPU9250** is the MPU6500 with the same magnetometer.

More and more FC now starting to use the**ICM20602**, for example the Raceflight Revolt V2, as well as the **ICM20689** on the [Kakute F4](https://oscarliang.com/review-holybro-kakute-f4-aio-fc/). They can both run at 32KHz, however it’s reported that these IMU are more susceptible than the older MPU6000 therefore these FC’s generally require soft-mounting.

The MPU chip model number can be found on the flight controller, for example this Invensense MPU-6000.

[](https://oscarliang.com/ctt/uploads/2017/04/flight-controller-fc-mpu-6000-6050-6500-gyro-accelerometer.jpg)

Figure 4.5: IMU Gyro Sensor on a flight controller

### Flash Memory or SD Logger for Blackbox

Onboard flash memory are used to store [blackbox](https://oscarliang.com/setup-blackbox-cleanflight/) data, which can be used to tuning and troubleshooting. Flash memory is more convenient and cheaper to use, but it’s extremely slow to download data, and size (2MB to 16MB) is often limited to 2 to 4 mins of flying.

Many flight controllers these days comes with built-in “Open Logger” which allows you to insert your SD card and record blackbox data. A 2GB card is enough for days of recordings.

If your FC doesn’t have flash memory or SD logger, you can also just get an external logger and connect it to your FC via serial port.

### Connector Types

I personally would avoid plastic JST connectors as they are less durable although they allow you to connect/disconnect more easily. I like solder pads, but sometimes with low quality boards, they tend to tear out when stressed or overheated with solder iron. I also like “through-holes” which gives you the option of direct soldering or using header pins.

[](https://oscarliang.com/ctt/uploads/2017/04/FC-flight-controller-solder-pads-through-holes-plastic-jst-connector.jpg)

Figure 4.6: Connector

* Pro Tip: [How to remove header pins on a FC](https://oscarliang.com/how-to-remove-header-pins/).
* Pro Tip: With solder pads, did you know that it’s possible to [fix them if they get peeled off](https://oscarliang.com/repair-ripped-off-solder-pads/)?

### Integrated Voltage Regulator

With voltage regulator integrated in the flight controller, it can be powered directly from LiPo battery, or even provide 5V even 12V voltage outputs for other gear. It also means the FC can monitor your battery voltage too without extra VBAT wiring.

### Integrated PDB

When PDB is integrated into the FC as one board, your ESC and LiPo battery can be connected directly to the FC and no additional PDB or wire harness is required (one board vs 2 boards). But the downside is usually the tiny soldering pads on the FC that is extremely hard to work with given the tight space.

**Integrated Current Sensor**

Having a current sensor has been proven invaluable: it’s a much better indicator than VBAT for when you should land and great tool for testing.

More about [current sensor and calibration](https://oscarliang.com/current-sensor-calibration/).

### Boot Button

A boot button (or bootloader button) can put FC into bootloader mode when pressed. This allows you to “force” flash firmware in case normal firmware flashing doesn’t work  ([why use bootloader button on FC](https://oscarliang.com/cannot-connect-to-naze32-brick-fix/)).

Originally FC’s provide 2 solder pads for you to bridge when bootloader mode is required. With a boot button it makes it so much easier.

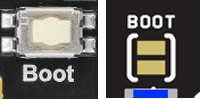
[](https://oscarliang.com/ctt/uploads/2017/04/flight-controller-fc-boot-bootloader-button-pads-solder-bridge.jpg)

Figure 4.7: Boot Button

Left: Boot Button; Right: Boot Pads

### Other features you might find on flight controllers

* Integrated OSD – anyone who has worked with MinimOSD knows what a PITA it is to setup, if you like OSD this feature is a must have
* Integrated VTX – the main benefit of it is the overall weight saving and compactness, some even allows you to change VTX settings directly from your FC
* Barometer/Mag (compass) – these extra sensors are less important for racers and mini quad
* RX Support – make sure the FC support the type of [radio receiver protocol](https://oscarliang.com/pwm-ppm-sbus-dsm2-dsmx-sumd-difference/) you plan to use, such as PWM, PPM, SBUS, Spektrum Satellite and so on
* AIO options – “All in one” flight controller has everything necessary built into one single board, such as FC, ESC, PDB, RX and so on. The downside is if something fails, it’s more likely that you’d have to replace the whole setup
* Infrared transponder support – allows you to use IR detectors with the FC directly to measure your lap time in some race events

**4.1.4 GPS and Data logging**

A program was designed and coded using the Ardunio IDE to test the GPS receiver and SD card adaptor. Two libraries were used in this program. The TinyGPS library developed by Mikal Hart was used to decode the NMEA data received by the GPS receiver. The SD card library was used to help read and write information to SD cards. This library was built on the sdfatlib developed by William Greiman. The library supports FAT16 and FAT32 file systems. A basic flow chart of how the program works can be seen in figure 4.5.

During testing with the GS407 GPS receiver, we ran into many diﬀerent problems. The first problem we encountered was that the BoB that came with our GPS module did not match up with the control board, as expected. We found out it was designed for an older version of our control board. This was easily solved, the wires just needed to be re-soldered diﬀerently to connect the ports to each other

The second problem also included the same BoB. The BoB was extremely fragile. The BoB connector used to connect the BoB to the GPS module came oﬀ. Our quickest solution was to solder wires directly to the GPS unit. This was hard to do because the pins were extremely small. A smaller gauge of wire was needed to solve this problem.

After the wiring was completed, we started testing the GS407 GPS receiver using the program mentioned earlier. Data was being transferred between the GPS unit and the microcontroller, but GPS receiver could not get a satellite fix. We tried testing in many diﬀerent locations, indoors and outdoors, to no success. We assumed that the GPS was not working properly and purchased a new GPS module, the Parallax PMB-648, for compa

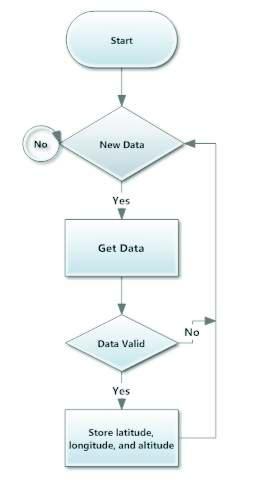


Figure 4.8: Flowchart of GPS data logging program.

The new GPS receiver module ordered was the PMB-648 using the SiRFstarIII chipset. Al-though this module was slower and heavier than the GS407, the module was easily to interface with the Arduino boards. The GPS module got a satellite fix almost immediately after connecting the GPS to power. The overall speed diﬀerences were unnoticeable during testing. In figure 4.6 below, shows the output of PMB-648 module, using the testing programmed developed above.

4.1.5 PID Tuning

There are three proportional-integral-derivative controller (PID) controllers performing error correction for pitch, roll, and yaw using inputs from the 9DOF sensor board. The output of each PID controller is used to adjust the orientation and stabilize the quadcopter along one axis. Each PID controller algorithm involves three separate constant parameters, the proportional, the inte-gral and the derivative values, denoted P, I and D. P indicates the present errors, I indicates the accumulation of past errors, and D indicates the prediction of future errors. Figure 4.9 is a block diagram of a PID controller.

In the process of PID tuning, the gains of the proportional, integral, and derivative terms have to be chosen correctly. Otherwise, the quadcopter may not be able to stabilize itself. For example, the higher the P value, the stronger the motor responds. If P is set to be too high, oscillations can be observed because of overshooting.

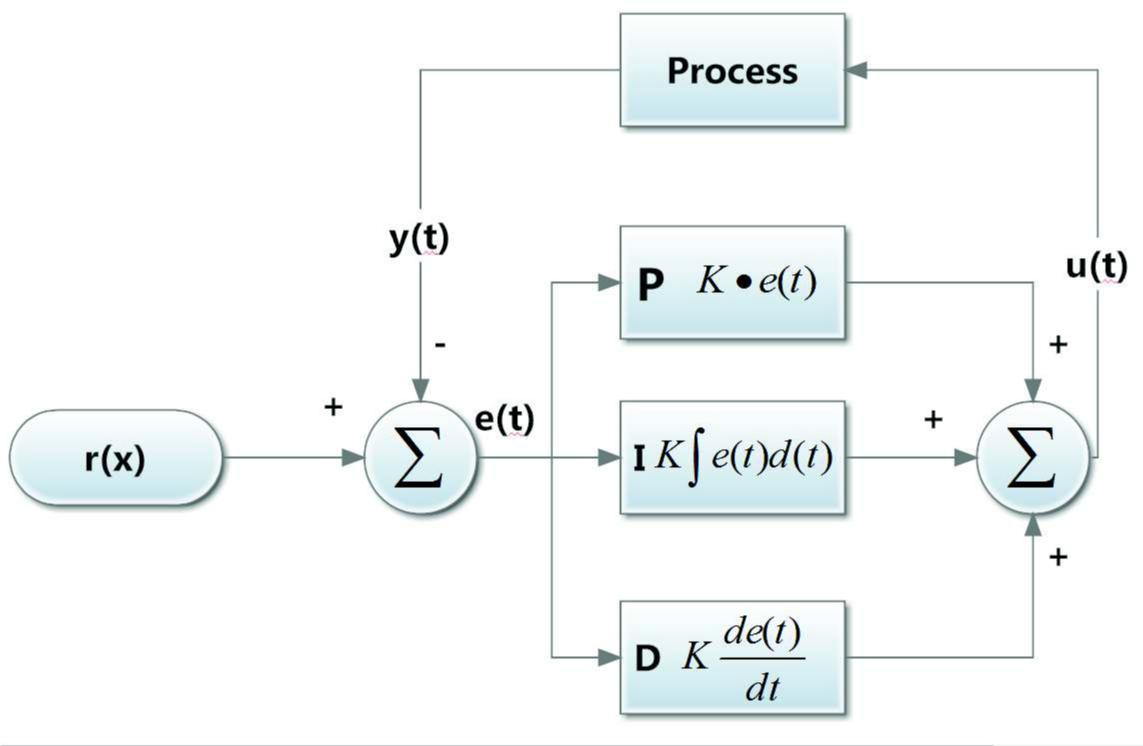


Figure 4.9: Block diagram of PID controller in feedback loop.

PID tuning required that the quadcopter was supported along its axis to check how the quad-copter reacted to certain forces applied on the axis. For our first test bench, we modified a aluminum camera stand shown in figure 4.10 and mounted the quadcopter to the stand.

The advantage of this PID tuning design is that the stand is portable. However, the stand was not strong enough to hold the weight of the quadcopter and the thrust created by the quadcopter tipped the stand over.

Our group decided to make a custom stand for our PID tuning purposes. The quadcopter was mounted on a copper pipe, held together by a wooden stand shown in figure 4.11. This only allowed the quadcopter to turn on one axis, allowing us to test and tune the PID one axis at a time. The wooden stand was much sturdier than the camera stand and was unaﬀected by the forces caused by the quadcopter.

The quadcopter has two modes, acrobatic mode and stable mode. These mode are controlled by the mode channel of the transmitter. acrobatic mode uses only gyroscope data to stabilize the Aeroquad. Acrobatic mode detects changes in angular rates on a single axis and adjust the motor to resist the angular change, from a level position. On the other hand, stable mode uses both gyroscope data and accelerometer data to maintain a level position. The quadcopter is able to stabilize itself after PID tuning. To perform the PID tuning, acrobatic mode tuning was done before the stable mode tuning because stable mode isolates the parameters of roll and pitch. Then

tuning was done for the stable mode.

The quadcopter was tuned using the method mentioned above. In acrobatic mode, we slowly adjusted the PID values and checked how responsive the motors were. When we felt that the quad-copter could adjust to the angular changes, we changed to stable mode to check if the quadcopter could automatically stabilize itself when a force was applied to one of the axis. The PID was slowly adjusted so that the quadcopter could adjust itself with minimal oscillations. This was done for both the pitch, roll, and yaw axis.

**4.2 Flight Test**

In this section of the document we will discuss how our initial flight test went and the problems we encountered during this flight test.

4.2.1 Initial Test Flight

After the PID tuning, the quadcopter was able to ready for its first flight test. The quadcopter was able to fly in a very stable manner without any observable osciallations and the hacking process is completely done without any error.

4.2.2 How quadcopter is able to hack?

The quadcopter in additional to performing tasks that the humans are incapable of for example inspection of high structures, etc has been installed with Kali Linux operating system with Raspberry Pi which is capable of hacking any wifi network by the function of air-crack-ng, fluxion, Metasploit ,brute force attack and many others.

The Quadcopter uses the above functionalities to hack the security such as WPA-2SK, WEP, WPA and WPS by deauthenticating the users which are connected to the wireless network.

When the Quadcopter enters any network range it injects a packet injection which disintegrates all the users from the network for a specific time and when the user retries to connect with the network then the wifi security key entered by the user is submitted to the attacker in several passes. The attacker who wants to hack the wifi network with quadcopter has Kali Linux and kernel RE4SON’S which is specially designed for Raspberry Pi which increases the efficiency by 40% .With this we are ready with a portable Hacking machine which is very efficient and further is durable.

“Sticky Fingers Kali-Pi combines Kali Linux and the Re4son Kernel into a hassle-free package that turns any Raspberry Pi into a powerful Swiss Army knife with finger-friendly touch screen interface that can provide backend support in your own network, or operate autonomously behind enemy lines, posing as a humidity sensor. Out of the box, the Kali-Pi comes preconfigured with all essential services, such as SSH, VNC, FTP, HTTP, and Wi-Fi backdoor, as well as the MANA toolkit, Snort, Kismet, Metasploit, etc. and a finger-friendly touch screen interface to control it all.”

Built on a Raspberry Pi 0/0W/1/2/3, this setup can be quick-installed from a pre-configured image or built from scratch, and has options for a TFT touch screen with a custom-built intuitive touch menu, Bluetooth, injection support, and more. He’s even strapped this puppy onto a drone for some high-flying “research”.

**4.3 Discussion**

4.3.1 Diﬃculties

We encountered delays in ordering parts. We ordered parts in late September and early October and parts did not arrive at the expected times. Our initial parts started arriving early November but some parts did not arrive until the last week of November. This delayed our schedule and caused us to fall behind.

1. There were many safety issues during testing the quadcopter with rotors. The spinning rotors could spin quick enough to cause serious harm to a person. Before each test with the rotor attached, our group would have to clear the room and make sure that no one would come into contact with the rotors of the quadcopter. During testing of the motors, all non-group members were told to leave the room. The room was locked and testing was done only care-fully starting at minimum throttle. If the quadcopter ran properly for a duration of time, then we would increase the throttle by 10%. This tedious process was used to ensure safety.
2. There was a problem with the battery connectors being diﬀerent. We did not notice the difference in connectors at the time of purchase. The JST-HX connectors on the battery did not match the EC3 connectors on the quadcopter. Fortunately we were able to find the necessary connectors at a local RC hobby shop and we were able to re-solder the new connectors to the battery.
3. Setting up the channels of the transmitter and receiver was also very important. The initial transmitter channel configurations did not match the quadcopter channel configurations. To solve this problem each channel of the transmitter needed to be tested separately to identify what it was controlling on the quadcopter and set to the correct corresponding channels.

1. **Conclusions**

This section discusses the results of the project. All information here is repeated from earlier sections of this report. This product is basically a hacking product. This drone is made up of various hardware and kali Linux is used as an operation system in it. This product is very useful when we talk about war and security. It also contains various sensors and camera which is used to identify the activities what is happens around this drone. This drone itself a very intelligent because it can handle various activities like trigging bullets when required during war .it can also have ability to find out or identifies if there is any immigrant. It also have a ability to find a single person by hacking mobile phones, camera, it matches voice of the intruder and face by face recognition system and many more thing we can do with because it also have characteristic to learn by the surrounding. What makes this drone is special that it has a self destructive program if our every attacks fails to kill the intruder then it will self destroy by itself and all the area in the range will be blast .so, there is a no chance to survival of any intruder or enemy in its range .the whole communication is done with help of satellite between drone and base station.

**Bibliography**

1. “Map of winnipeg, manitoba,” February 2013. [Online]. Available: https://maps.google.ca/
2. R. Mahony, P. Pounds, and P. Corke. (2006, December) Modelling and control of a quad-rotor robot. In the Proceedings of the Australasian Conference on Robotics and Automation. Auckland, New Zealand. Accessed: September 2012. [Online]. Available: http://www.araa.asn.au/acra/acra2006/papers/paper 5 26.pdf
3. I. Sa and P. Corke. Vertical infrastructure inspection using a quadcopter and shared autnomy control. Brisbane, Austrailia. Accessed: September 2012. [Online]. Available: https://wiki.qut.edu.au/download/attachments/107284164/FSR2012 Inkyu 5th sub.pdf?vers%20ion=1&modificationDate=1336532095000
4. V. Ross. (2011, August) In development: Networks of unmanned quadcopters to ferry

medicine to isolated areas. Discover Magazine. Accessed: September 2012. [Online]. Avail-

able: http://blogs.discovermagazine.com/80beats/2011/08/30/in-development-networks-of-% 20unmanned-quadcopters-to-ferry-medicine-to-isolated-areas/#.UTqfsxxgCXg



[5] G. Elliot. (2005, November) Development of an autonomous quadro-

tor flying platform. Accessed: September 2012. [Online]. Avail-

able: http://blogs.discovermagazine.com/80beats/2011/08/30/in-development-networks-of-% 20unmanned-quadcopters-to-ferry-medicine-to-isolated-areas/#.UTqfsxxgCXg

1. G. M. Hoﬀman, H. Huang, S. L. Waslander, and C. J. Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. Accessed: October 2012. [Online]. Available: http://hoﬀmann.stanford.edu/papers/Quadrotor Dynamics GNC07.pdf
2. T. Carancho. aeroquad software. Accessed: September 2012. [Online]. Available: https://code.google.com/p/aeroquad/
3. S. W. Colton. (2010, June) Design and prototyping methods for brushless motors and motor control. Massachusetts, United States. Accessed: October 2012. [Online]. Available: http://web.mit.edu/scolton/www/SCThG.pdf
4. R. F. Graf, “Dictionary of electronics,” 1974., accessed: January 2013. [Online]. Available: http://www.rc-airplane-world.com/spread-spectrum.html
5. “Canadian municipalities and the regulation of radio antennae and their support structures,” accessed: January 2013. [Online]. Available: http://www.ic.gc.ca/eic/site/smt-gst.nsf/eng/ sf09386.html
6. “72 mhz vs 2.4ghz radio systems compared,” February 2013, accessed: January 2013.

[Online]. Available: http://sjrcf.com/docs/72mhz%20vs%202.4%20ghz.pdf

1. P. Energy, “Battery types: Which batteries to use?” accessed: October 2012. [Online].

Available: http://www.pureenergybattery.com/pdf/batterytypes.pdf

[13] “Peukert’s law, a nerd’s attempt to explain battery capacity,” ac-

cessed: September 2012. [Online]. Available: http://www.batterystuﬀ.com/kb/tools/

peukert-s-law-a-nerds-attempt-to-explain-battery-capacity.html

1. Starlino, “Dcm tutorial - an introduction to orientation kinematics,” accessed: January 2013.

[Online]. Available: [http://www.starlino.com/dcm tutorial.html](http://www.starlino.com/dcm%20tutorial.html)

1. Laden Bin News, Wikipedia.
2. Warren R. (1982). The Helicopters. The Epic of Flight (Chicago: Time-Life Books). p. 28. [ISBN](http://en.wikipedia.org/wiki/International_Standard_Book_Number) [0-8094-3350-8](http://en.wikipedia.org/wiki/Special:BookSources/0-8094-3350-8)
3. ["A Successful French Helicopter"](http://www.flightglobal.com/pdfarchive/view/1924/1924%20-%200047.html) Flight 24 January 1924 p47
4. "Helicopters of the World" Flight 2 November 1956 p722]
5. The Quadrotor’s Coming of Age". Retrieved 29 December 2014.
6. The Quadrotor’s Coming of Age". Retrieved 29 December 2014
7. "Aeryon Scout Quadrotor Spies On Bad Guys From Above". Retrieved 29 December2014.
8. University of MANITOBA Final Report Testing of UAV.
9. By [Acosta](http://edition.cnn.com/profiles/jim-acosta-profile) Jim, CNN Senior White House Correspondent, Updated 1818 GMT (0118 HKT) April 27, 2015
10. Dna , Thursday, 11 December 2014 - 8:40pm IST | Agency: PTI

25 . Bussiness Standards, Press Trust of India | New Delhi , December 15, 2014 Last Updated at 00:35 IST

26. https://github.com/

27. https://samy.pl/skyjack/

28 .https://www.kali.org/news/kali-drones-portable-ctf-builds-raspberry-pi-craziness-and-more/

