A Project Report

on

AUTONOMOUS DRONE (QUAD-COPTER) WITH WIRELESS & GESTURE CONTROL SYSTEM

Submitted for the partial fulfillment of the requirement

for the award of the Degree of

Bachelor of Technology

In

Electronics and Communication Engineering

By

Faiz Ahmad (1501031183)
Ch Madhu Kumar (1505031001)

Under the supervision of

Dr. Anil Kumar Gupta
Professor & Head
Department of EECE
DIT University

DIT UNIVERSITY, DEHRADUN, INDIA

MAY, 2019

DECLARATION

This is to certify that the project entitled **"AUTONOMOUS DRONE (QUAD-COPTER) WITH WIRELESS & GESTURE CONTROL SYSTEM"** in partial fulfillment of the requirement for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering submitted to DIT University, Dehradun, Uttarakhand, India is an authentic record of Bonafide work carried out by **Faiz Ahmad (1501031183) & CH Madhu Kumar (1505031001)**, under the supervision of **Dr. Anil Kumar Gupta,** Head of Department.

The matter embodied in this project has not been submitted for the award of any other degree or diploma to any university / Institution.

Signature

Faiz Ahmad (1501031183) **Ch Madhu Kumar (1505031001)**

Date: 8th May 2019

Place: Dehradun

CERTIFICATE

This is to certify that the project entitled **"AUTONOMOUS DRONE (QUAD-COPTER) WITH WIRELESS & GESTURE CONTROL SYSTEM"** in partial fulfillment of the requirement for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering submitted to DIT University, Dehradun, Uttarakhand, India, is an authentic record of Bonafide work carried out by **Faiz Ahmad (1501031183), Ch Madhu Kumar (1505031001)** under my guidance.

Signature and Name of Project Supervisor (s)

Dr. Anil Kumar Gupta (Head of Department)

Date: 8th May 2019.

Place: Dehradun

The project is successfully completed and the project report is submitted after the viva held on **26th April 2019.**

Project Coordinator

Mr. Anuj Kumar Sharma

Date: 8th May, 2019

Place: Dehradun

ACKNOLEDGEMENT

I am highly in debt to my project supervisor ***Dr. Anil Kumar Gupta (HOD),*** **Department of EECE** for his kind support and guidance without which this project work would not have been possible. His untiring encouragement has always been endless source of motivation for me.

I would like to take this opportunity to pay my sincere thanks to ***Dr.Anil Kumar Gupta (Head Department of EECE)*** who allowed me to carry out this project and provided me with his valuable guidance and suggestion.

I would like to give my sincere thanks and regards to ***Mr. Anuj Kumar Sharma***, ***Project Coordinator –ECE*** who has always provided me the moral support for completing this work.

I would like to dedicate this work to my parents and my all friends who has always supported me in motivational way towards understanding the life and the importance of work.

Last but not the least, I would like to thank friends whose toilsome efforts, constant inspiration and affection helped me to complete this work

Faiz Ahmad (1501031183)

Ch Madhu Kumar (1505031001)

B.Tech (ECE-B)

ABSTRACT

Mechatronics is a multidisciplinary branch of engineering that focuses on the engineering of both electrical and mechanical systems, and also includes a combination of robotics, electronics, computer, telecommunication,product engineering.

As technology advances over time, various subfields of engineering have succeeded in both adapting and multiplying. The intention of mechatronics is to produce a design solution that unifies each of these various subfields. Originally, the field of mechatronics was intended to be nothing more than a combination of mechanics and electronics, hence the name being a portmanteau of **mecha**nics and elec**tronics. Drones** are a great example of mechatronics in practice. They have many interconnected systems that depend on multiple disciplines to communicate and function as designed. In this article, we'll explain how mechatronics improve drone technology, explore the inner workings of drones and demonstrate the technology making them possible.

Drones, also known as unmanned aerial vehicles, are on the rise in recreational and in a wide range of industrial applications, such as security, defence, agriculture, energy, insurance and hydrology. Drones are essentially special flying robots that perform functionalities like capturing images, recording videos and sensing multimodal data from its environment. There are two types of drones based on their shape and size, fixed-wing and multi-rotor. Because of their versatility and small size, multi-rotor drones can operate where humans cannot, collect multimodal data, and intervene in occasions. Moreover, with the use of a guard hull, multi-rotor drones are very sturdy in collisions, which make them even more valuable for exploring uncharted areas. At present, flying robots are used in different businesses like parcel delivery systems. For example, companies like Amazon Prime and UPS are using multi-rotor drones to deliver their parcels. New York Police Department uses quad-copters in crime prevention. For the purposes of agriculture monitoring, for instance, the use of multiple sensors such as video and thermal infrared cameras are of benefit. Drones are especially useful in risky missions.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

ABOUT THE LEARNING FOR THE PROJECT

- We had the training in embedded system in 2nd year from Eduvance by Jonathan Joshi sir.
- In the end of 3rd year, we did summer training from CRIST BRILLICA in embedded system and IOT.
- In fourth year, we did online course on Drone technology and learnt to make quad-copters.
- We learnt Arduino coding on assembly language C++.
- Learnt Interfacing the Joystick Module & NRF module.
- Learnt Interfacing Accelerometer & Gyro sensor (MPU6050).

Chapter 1

Introduction

Quad-Copter is an Unmanned Aerial Vehicle (UAV) which is operated to fly independently. It's a multi-rotor helicopter that is lifted or propelled by 4 rotors (propellers).

Quad copter is a device with an intense mixture of Electronics, Mechanical & mainly on the principle of Aviation. The Quad copter has 4 motors whose speed of rotation & the direction of rotation changes according to the users desire to move the device in a particular direction (i.e. Takeoff motion, Landing motion, forward motion, backward motion, Left motion, Right motion.) The rotation of motion changes as per the transmitted signal sends from the 6-Channel transmitter. Each rotor produces both a thrust and torque about its centre of rotation, as well as a drag force opposite to the vehicle's direction of flight.

In our project, we are designing a little automated Quad-Copter programmed in a manner that will fly in a specific amount of range for specific period of time. Quad copter will move and hover around by keeping its stability maintained. In its automation we will introduce Altitude hold, Position hold, automatic landing when needed.

First, we are going to fly it on market available FLYSKY 6 channel PWM based transmitter and receiver.

During this we would check its stability and efficient control from the pilot side.

After the flying part, we would make our own made transmitter and receiver via pcb designing and will replace the market available transmitter & receiver.

Proper testing of the transmitter and receiver is to be done which include connection and signal check on transmitter and receiver.

After the successful completion of the self made transmitter and receiver, we would introduce gesture control system to it. This will include interfacing MPU6050 (accelerometer & gyro) and flex sensor with Arduino Nano.

The transmitter part will contain a glove mounted board with Arduino Nano and mpu6050 on top.The forefinger part will have flex sensor for throttling.

Then a receiver part will be made with Arduino Nano and NRF sensor for receiving and procession signals from transmitter and send in to flight controller board.

Chapter 2

Design description and methodology

We had a vision for this project time that is long. Our objectives are tough to obtain. With the help of our teachers, we hope success in this. We decided to split this project in 3 objectives. After completion of each target we will otherwise go forward will stay on it. The objectives of the 3 goals are explained below: -

- Target-1 – In this we will design the architecture that is basic of drone with different fundamental aspects of right specification. First, we will try to fly it with wireless RC (Remote Controller). It shall need lots of calibration on flight control board. Components which are building blocks of this droneare: -

1) Light weight frame (F450 DJI)
2) X4 high RPM DC motors (A2212 1000KV engine that is brushless
3) Electronic Speed control X4 (30A ESC)
4) Flight Control Board (KK 2.1.5)
5) Propellers (2 clockwise & 2 counter-clockwise) – (1045 propellers)
6) Battery & Chargers (Li-Po 11.1v, 2200mAh)
7) Radio receiver and transmitter(6-channel PWM or PPM based)

- Target-2 – After completion of target-1, we will make our transmitter that is self-made and considering joystick module via pcb designing. The components required for it are: -

1) 2 x PCBs
2) 2 x Joystick modules
3) 2 x Arduino Nano
4) 2 x NRF sensor with antenna

5) 9V Battery

6) Switch

7) 3.3v and 5v voltage regulator

8) Jumper wires

• Target-3 –Then we will introduce another control that is cordless based on hand Gesture. This control that is gesture may be implemented using Accelerometer and Gyro sensor on a hand piece which will be connected with control board wirelessly. Here the components we shall need are:-

1.) Arduino Nano with

2.) Sensors – MPU6050-6 Axis Gyro & accelerometer, flex sensor

Basic Terminologies

- **Type of Drone** – There are various formats of drones based on multi-propellers. Like Tri-copter, Quad-copter, Hexa-copter, Octa-copter etc.

➢ In this project we are using a drone with 4 rotors/propellers which is also called quad-copter.

➢ Quad-copters are also of two types:-

1)Plus(+) type 2) X – type

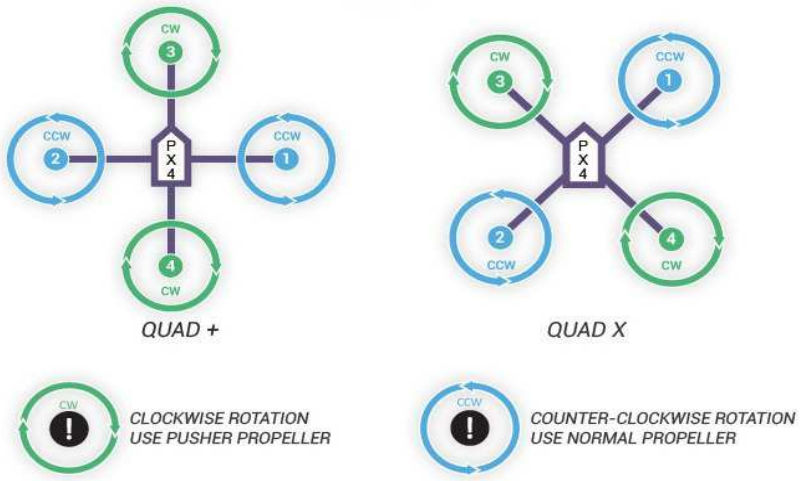


Figure 1: Quad-Copter type

Image Source: https://www.google.com/imghp?hl=EN

- We would be making a X – type Quad-copter.
- Reason behind choosing an X-type is because it's easy to control.
- Movement mechanism is strong and stable.

- **Propellers Movement –**We try to balance the Quad- copter by making the two pairs of propellers move in two different directions.
- One pair in Clock-wise direction.
- Another Pair in Counter Clock-wise direction.

- Clock-wise movement of 2 motors on one shaft.

- Counter Clock-wise movement of 2 motors on another shaft.

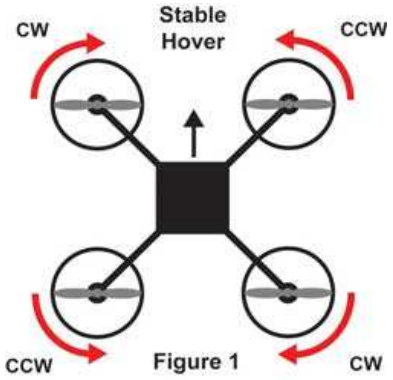


Image Source: https://www.google.com/imghp?hl=EN

Quad-Copter Movement- The Quad-Copter while flying moves along (X, Y, Z) axis of space according to aircraft movement principles.

- Along X axis – The movement is called "ROLL".
- Along Y axis – The movement is called "PITCH".
- Along Z axis – The movement is called "YAW".

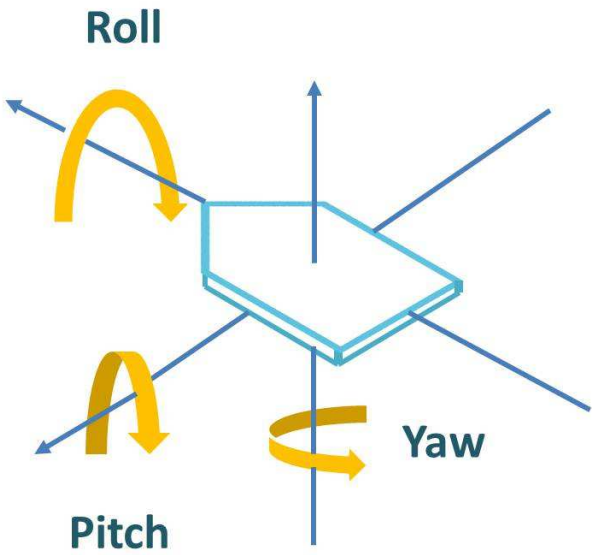


Fig 2: Movement

Image Source: https://www.google.com/imghp?hl=EN

Altitude Parameter – As the drone lifts off from the earth surface, gravitational full and temperature and pressure decreases. So a self made drone on normal flight controller boards is set to certain limits of altitude around 150 meters from the ground.

- In India we have certain rules and regulations governing the altitude limits of drone flying:-
  - Drones cannot be flown more than 400 feet vertically.
  - A permit is required for commercial drone operations (except for those in the Nano category flown below 50 feet and those in the Micro category flown below 200 feet).
  - Drones cannot be flown in areas specified as “No Fly Zones”, which include areas near airports, international borders.

GETTING STARTED

TARGET-1

Building blocks for the quad-copter are explained individually here:-

1.) FRAME – We would be using DJI F450 frame chassis.

- It's strong, versatile and well designed to balance the weight to the centre of the Quad-Copter.
- It has light weight carbon fibre centre plates with good power distribution.
- Frame weight- 282g

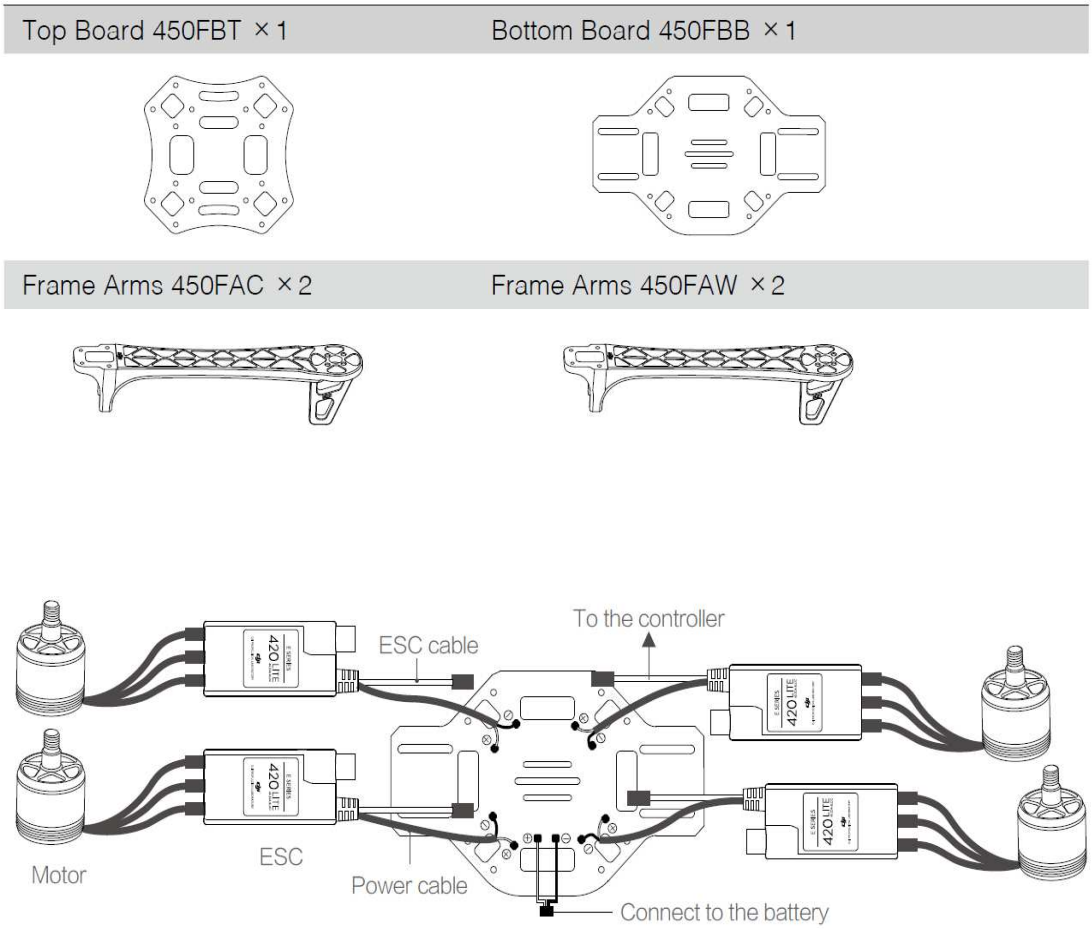
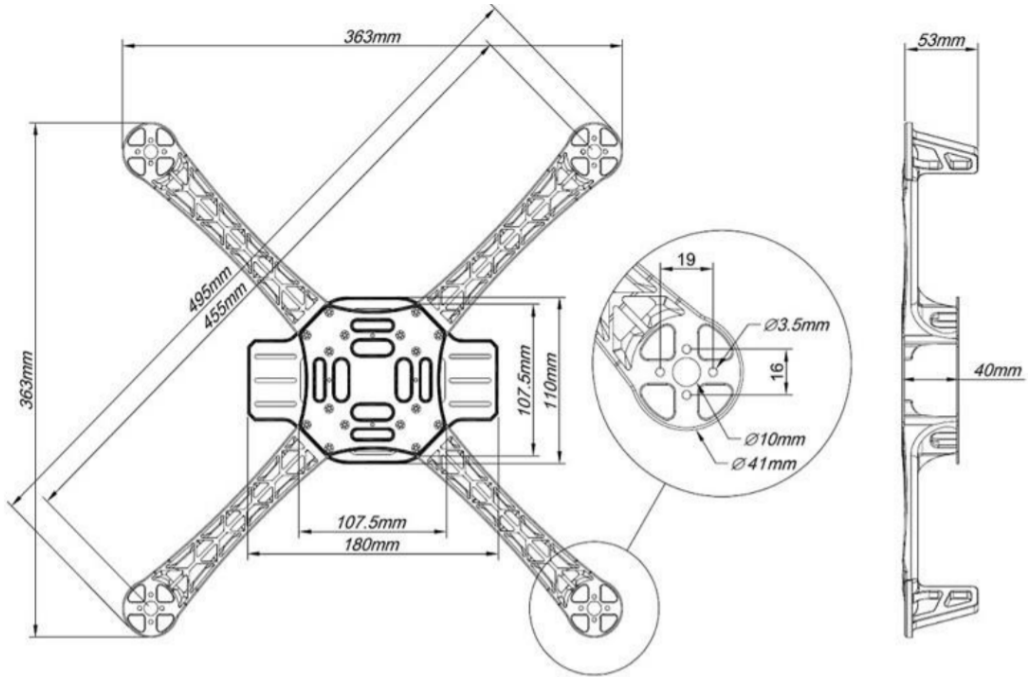


Figure 3

Image Source: https://www.google.com/imghp?hl=EN



Dimensions

Image Source: https://www.google.com/imghp?hl=EN

2) BRUSHLESS DC MOTORS –We would be using high RPM brushless DC motors because it provides more torque. Its rotating part is outside not inside.A brushless DC motor (also known as a BLDC motor or BL motor) is an electronically commuted DC motor which does not have brushes. The controller provides pulses of current to the motor windings which control the speed and torque of the synchronous motor.

Outer Rotor Design

In outer rotor design, the rotor surrounds the winding which is located in the core of the motor. The magnets in the rotor trap the heat of the motor inside and do not allow to dissipate from the motor. Such type of designed motor operates at lower rated current and has low cogging torque.

- The model we would be using is A2212 1000Kv
- We need 4 of these motors.
    - The numbers A2212 means that motor is 22mm wide and 12 mm in height.
    - The number 1000Kv means 1000 cycles in 1 min at 1 Volt of potential given.

Fig 4: Brushless DC Motors

Image Source: https://www.google.com/imghp?hl=EN

3) Propellers –These are the rotors which cut the air and pulls the drone from ground. We would be using 4 propellers on all those 4 motors.

- We would be using Model “1045” for the Quad-copter.
- There will be 2 pairs of propellers of different orientations.
- 2 propellers will be moving in clockwise direction on one shaft.
- And the other 2 propellers will be moving in Counter clockwise direction on another shaft.
- The number 1045 means:-
    - 10 inch in width.
    - 45mm of displacement in 1 revolution of blades.

Fig 5: 1045 fibre propellers

Image Source: https://www.google.com/imghp?hl=EN

4) ESC (Electronic Speed Controller) -- Circuit meant to control the speed of the Dc motors, its direction and possibly acts as a breaking system.

- We would be using 30Ampere rating model of ESC
- It provides the precision in the speed of motors.
- We would be using 4 ESC's for those 4 motors which will be controlled by flight controller board.
- Since it's a Quad-Copter, little up's & down's in motor speed will lead to crash. So ESC calibration is must and will be done by Flight controller Board.

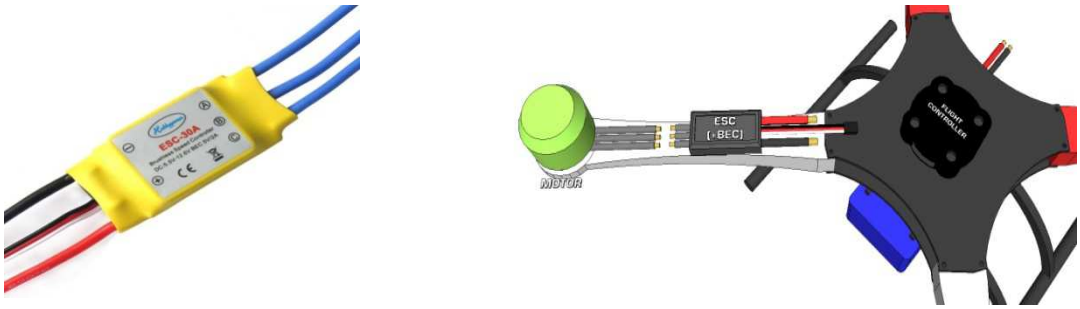


Fig 6: Electronic speed controller

Image Source: https://www.google.com/imghp?hl=EN

5) Flight Control Board– Itsthe brain of the aircraft. It's basically a **circuit board** with sensors that detects orientation changes of your drone. It also receives user commands, and **controls** the motors in order to keep the Quad-Copter in the air.

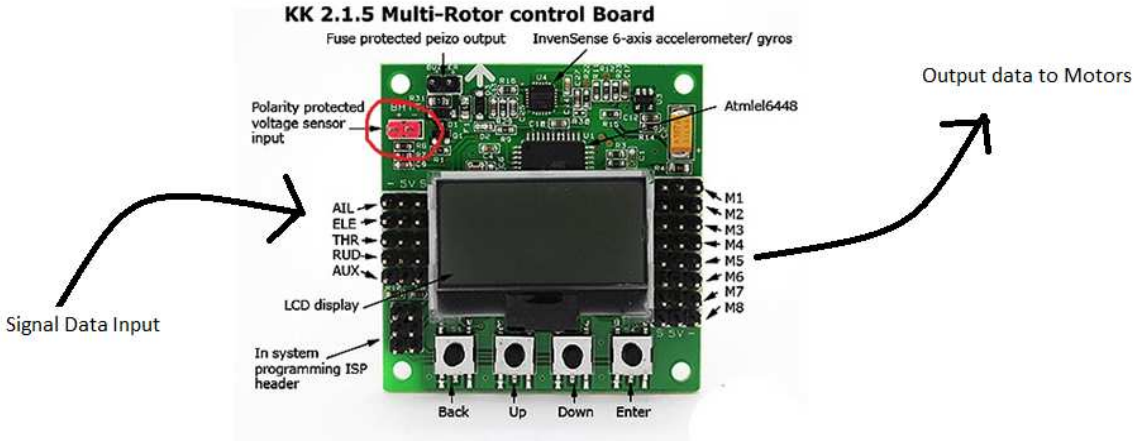
- We would be using FCB with good quality Gyro, accelerometer.
- We would choose the FCB based on fast microprocessor rather than slow microcontrollers, so that the automation like altitude hold, position hold, automatic landing etc can be done through it efficiently.
- We would be using FCB Model - (KK 2.1.5) based on budget and criterions.
- Better processor leads to better performance. These flight controllers would decrease the risk of crash during the flying of the full drone body.

KK2.1.5

KK 2.1.5 is a board with ATMEL mega 664PA, 8-bit AVR RISC based microcontroller with 64K of memory. It is easy for the beginner to start with and has firmware pre-defined in it. While activating or deactivating the board there is an audio warning from the piezo buzzer of KK 2.1.5.It is the most stable board because it has inbuilt gyroscope, 6050 MPU, and auto level function. This board has eight motor outputs, five control inputs, an LCD display, polarity protected voltage sensor input, an ISP header, six-axis Accelerometer/gyroscope, a fuse protected piezo output. The user-defined signals from K.K.board are processed by ATMEL 644PA IC and these control signals are passed to the ESC's installed on the frame of the drone.

Fig 7: Flight controller board (KK 2.1.5)

Image Source: https://www.google.com/imghp?hl=EN



Pin Diagram

Image Source: https://www.google.com/imghp?hl=EN

6) Battery & charger – It would be used to power up the motors by powering the ESC through power distribution board. It would also be used to power up the flight controller board and other sensors connected to FCB like GPS module, Receiver Module etc.

- We would be using Lithium Polymer battery which is provides very high unmatched current flow.
- Li-Po battery is the best and is used in modern electric cars too.
- We would be using a model of 2200mAH – 4000mAH with 2cells having 3.7V per cell.

Fig 8: Li-Po battery

Image source: https://www.amazon.in/

7) Radio Transmitter and Receiver – It's the most important component of the human to drone interface. It is used to control the drone through basic finger sticks on transmitter through high range radio signals.

- Its 6-channel based Transmitter and receiver.
- It's based on PWM or PPM signalling in some models.
- Communication frequency is at 2.4 Ghz frequency.
- The response time is so fast because it uses high-tech protocols to communicate.
- The data received by the receiver is decoded very fast and sent to FCB for response on ESC.
- We would be choosing the right model on the basis of quality, brand and price later.

Fig 9: Radio transmitter & Receiver

Image Source: https://www.banggood.in/

After bringing all these components, we would construct the drone through assembling, Soldering, attaching, Pasting etc.

Our Model

Fig 10: Model of Target-1

Image Source: By our camera

TARGET-2

- After completion of target-1, we will make our self-made transmitter and receiver based on joystick module via PCB designing.
- The components required for it are:-

1) 2 x PCBs- Copper/Aluminium - PCBs are to be used.

Fig 11

Image Source: https://www.google.com/imghp?hl=EN

2) 2 x Joystick modules - When we listen the word "**Joystick**" we think of Game controllers. If we talk about Electronics there are many useful application of Joystick. These type of module are mostly used in Arduino based DIY projects and Robot Control. As we know, the module gives analog output so it can be used for feeding the analog input based on direction or movement. It can also be connected to a movable camera to control its movement.

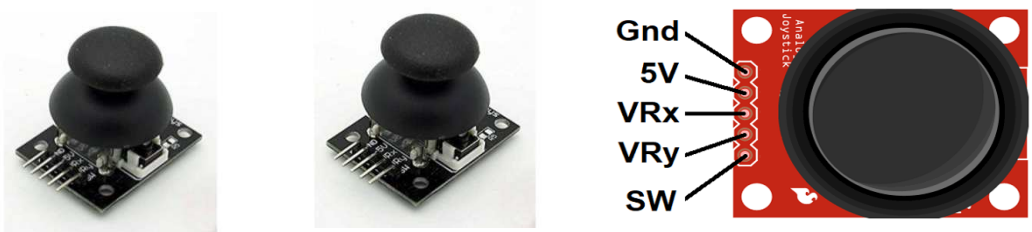
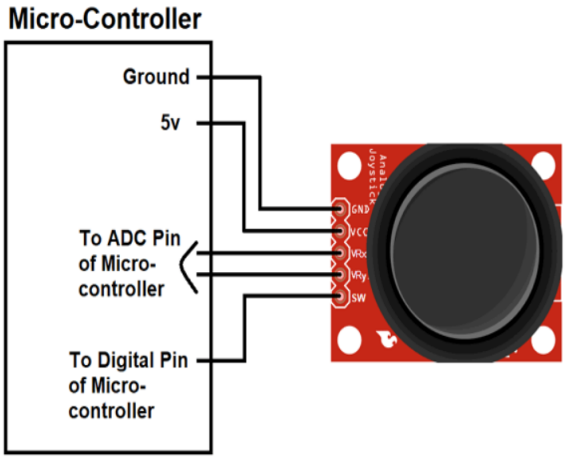
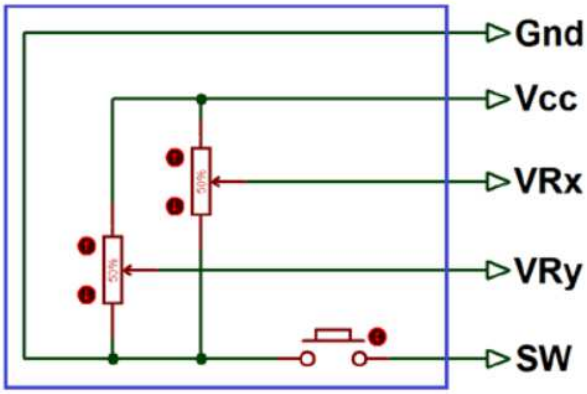


Fig-12

Image Source: https://www.google.com/imghp?hl=EN





Pin Diagram Internal Structure

Image Source: https://components101.com/modules/joystick-module

Pin Configuration

- GND: ground
- +5V: 5V DC
- VRx: voltage proportional to x position
- VRy: voltage proportional to y position 5. SW: switch pushbutton

Features

- Two independent Potentiometer: one for each axis (X and Y)
- Auto return to center position
- Low weight
- Cup-type Knob
- Compatible to interface with Arduino or with most microcontrollers

Technical Specifications

1. Operating Voltage: 5V
2. Internal Potentiometer value: 10k
3. 2.54mm pin interface leads
4. Dimensions: 1.57 in x 1.02 in x 1.26 in (4.0 cm x 2.6 cm x 3.2 cm)
5. Operating temperature: 0 to 70 °C

3) 2 x Arduino Nano - It is a Microcontroller board developed by Arduino.cc and based on Atmega328p / Atmega168.

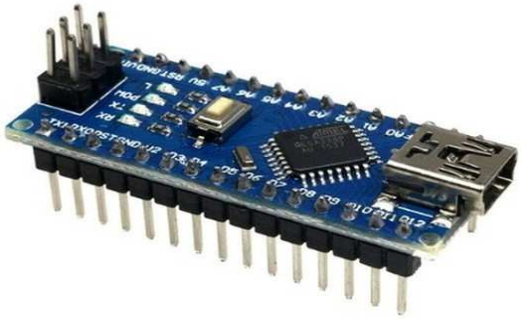
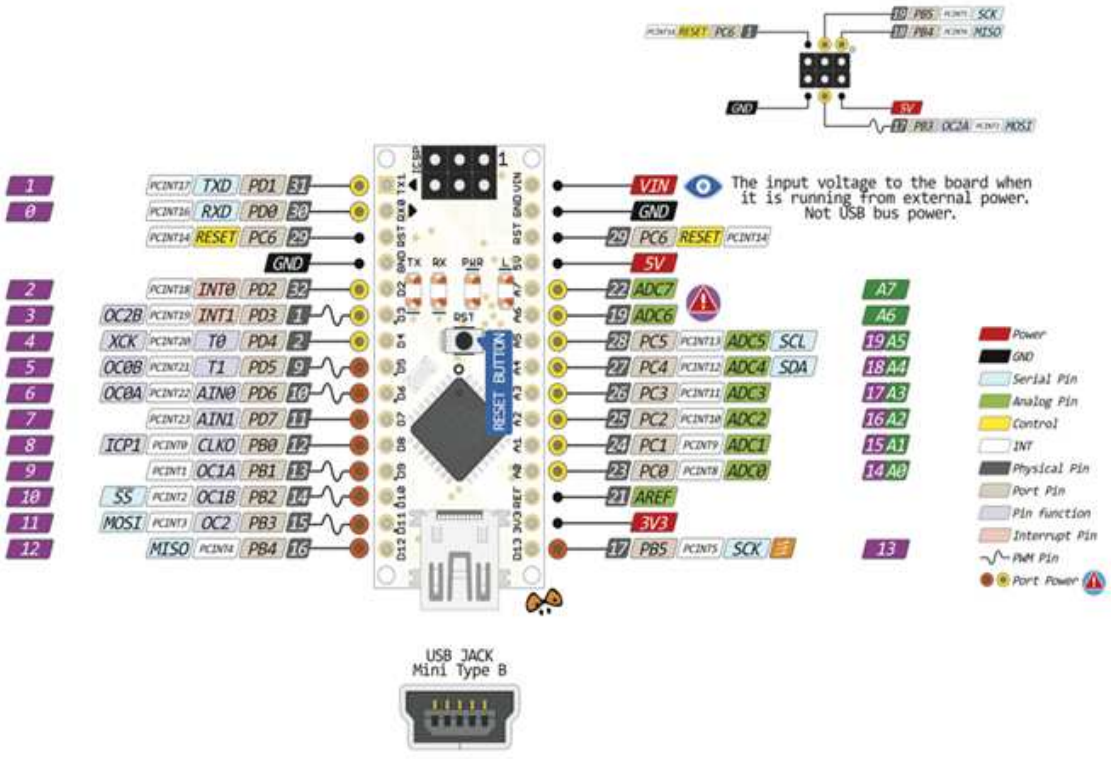


Fig: - 13

Image Source: https://www.google.com/imghp?hl=EN



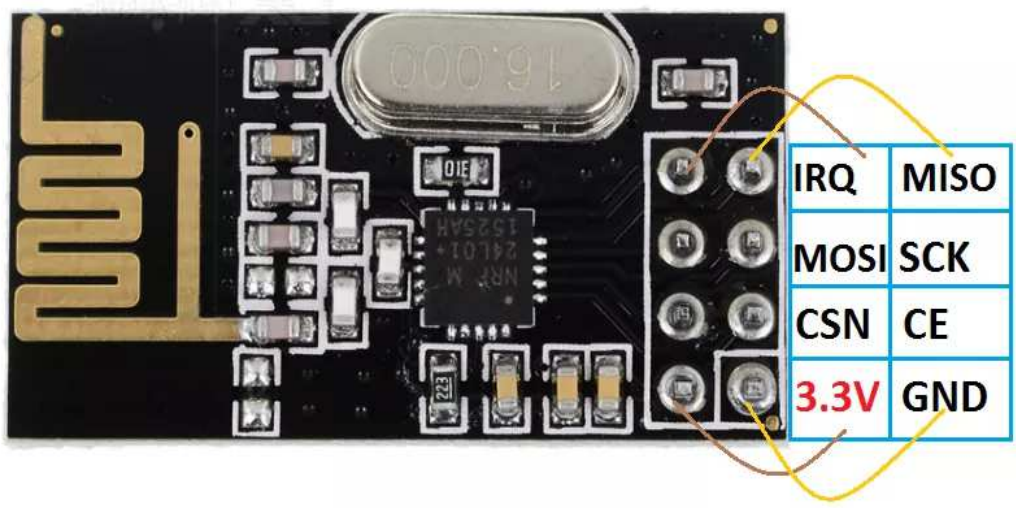
Pin Diagram

Image Source: https://www.google.com/imghp?hl=EN

4) **2 x NRF sensors with antenna** - NRF24L01 Module transceiver module which means that it can both send and receive the data. These modules are very cheap, smaller in size and have a lot of specifications. Some of the specifications of these modules are as follows:-

Specifications of NRF24L01 Module

- Power consumption is around 12mA during transmission which is even lesser than the led.
- It can operate with baud rates from 250Kbps up to 2 Mbps.
- Its range can reach up to 100 meters if used in open space and with antenna.
- It can both send and receive the data simultaneously.
- Each module can communicate with up to 6 other modules.
- It uses the 2.4 GHz band.
- It can send 1 to 25 bytes of raw data at the transmission rate of 1 MB.



Pin Diagram

Image Source: https://components101.com/

Fig-14

Image Source: https://www.google.com/imghp?hl=EN

5) **3.3v and 5v voltage regulator** - A voltage regulator is a system designed to automatically maintain a constant voltage level. A voltage regulator may use a simple feed-forward design or may include negative feedback. It may use an electromechanical mechanism, or electronic components.

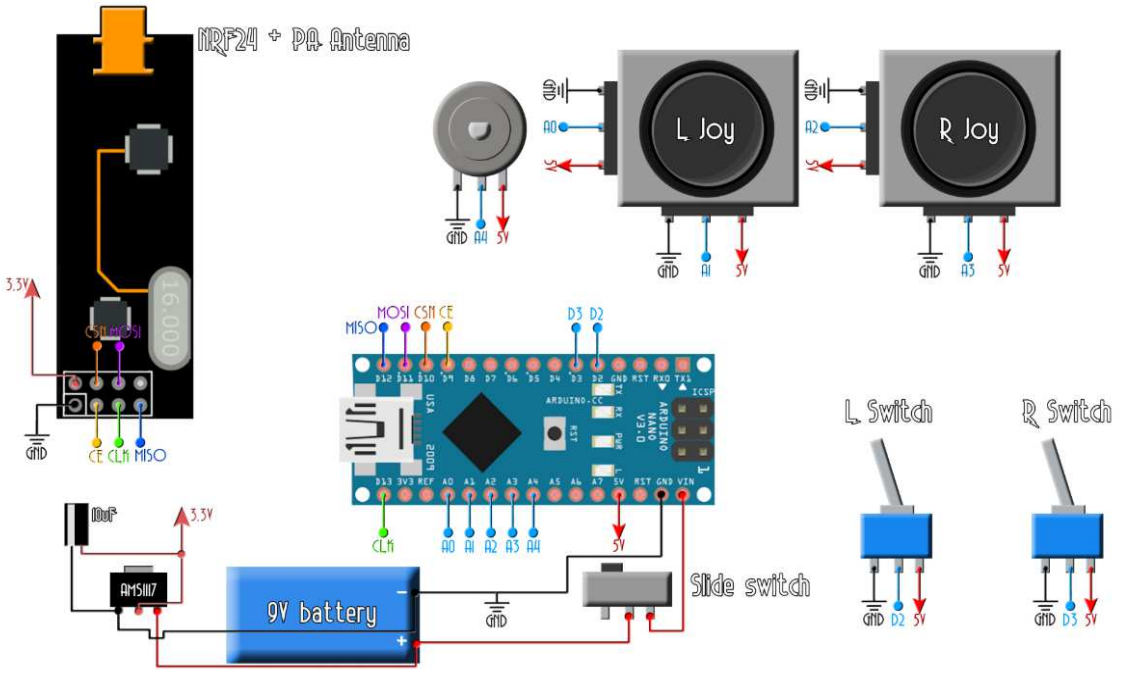
Fig- 15

Image Source: https://www.google.com/imghp?hl=EN

Design for Transmitter

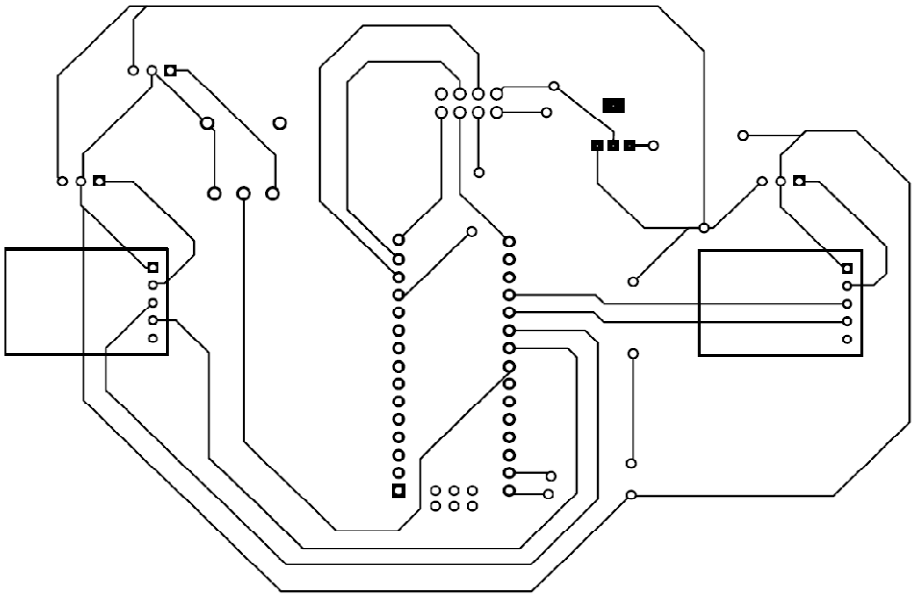
The components required for building the transmitter are as follows:-

- 1 Arduino Nano
- 2 Analog joysticks
- 1 NRF module
- 3.3v voltage regulator
- 5v voltage regulator
- 10uf and 100nf capacitors
- Switch
- Analog potentiometer



Schematic Diagram

Source: Software Schematic



PCB design

Source: Our designed output from online platform (https://easyeda.com/)

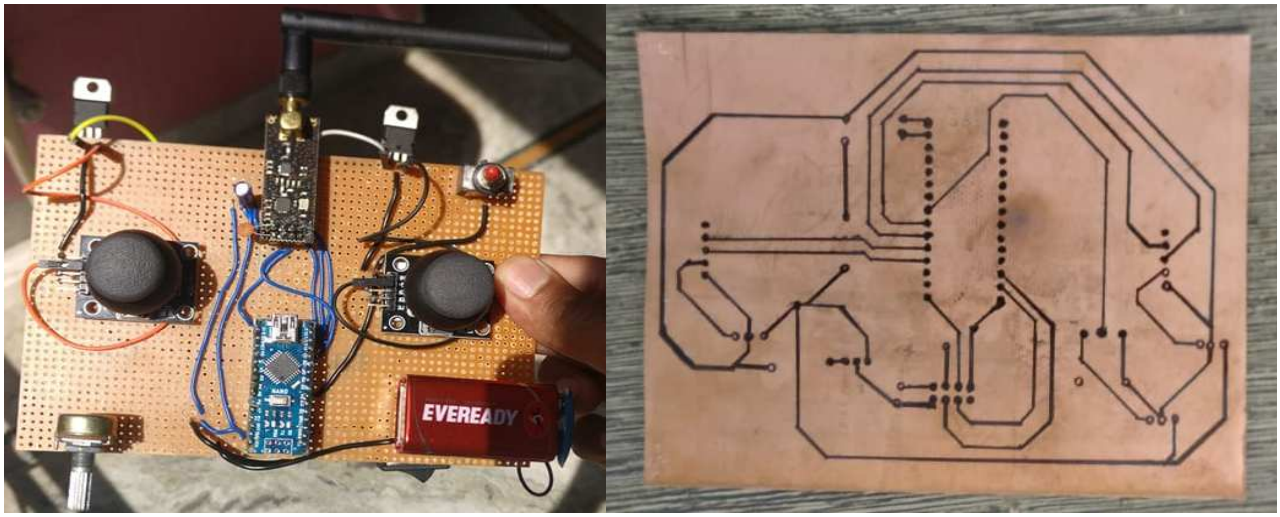


Fig 16 - Model of Target-2 (transmitter)

Image Source: Shot taken from our camera

Methodology

After designing PCB, we would do the following: -

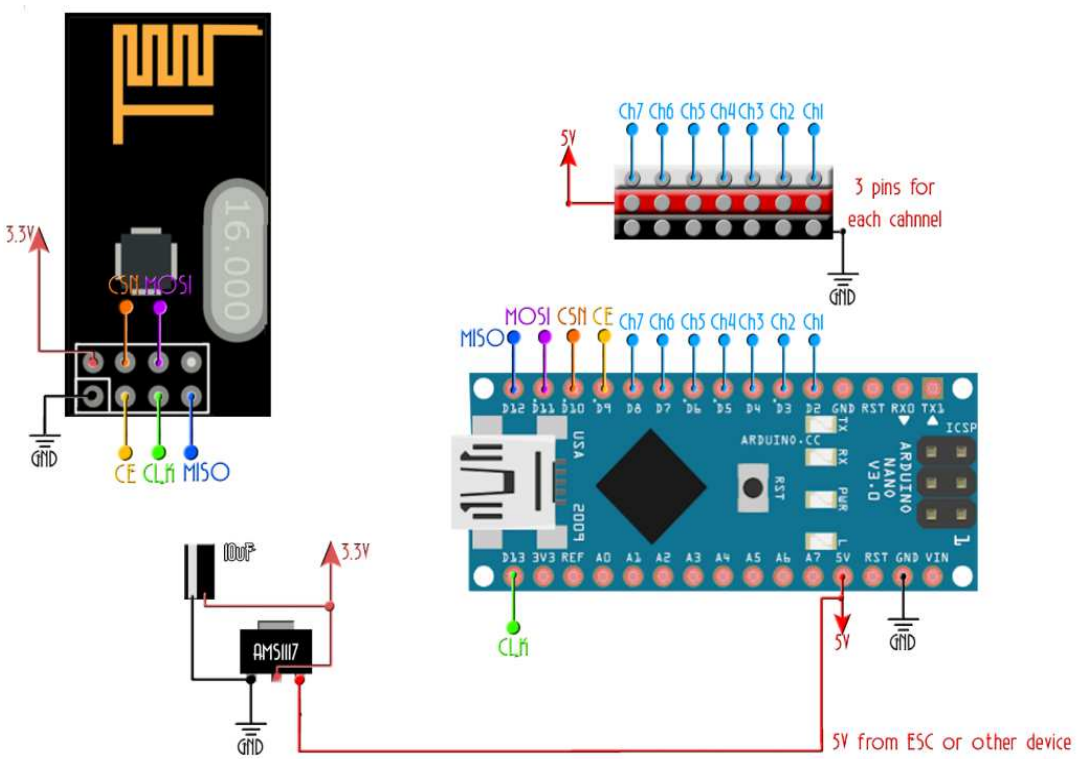
- We would do the circuit simulation manually on bread board and check everything is working fine.
- We solder each and every component to its right position at right pins.
- Then we would do the programming and upload the codes on Arduino Nano.
- Then we would do the testing part which is in next chapter.

Code: - Is in the ANNEXURE.

Design for Receiver

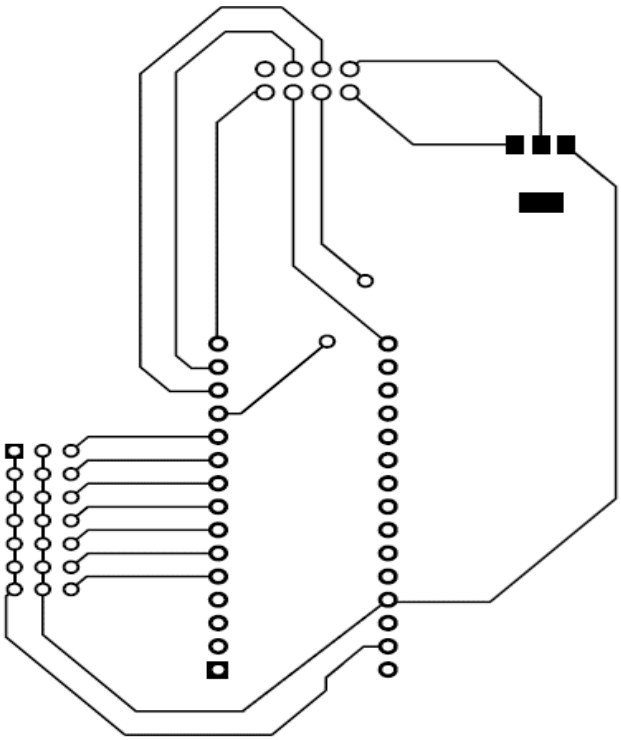
Following components were used to build the receiver:-

- Arduino Nano
- NRF module
- Pin bridges
- 10uf and 100nf capacitors
- 3.3v voltage regulator



Schematic Diagram

Source: Software Schematic



PCB design

Source: Our designed output from online platform (https://easyeda.com/)

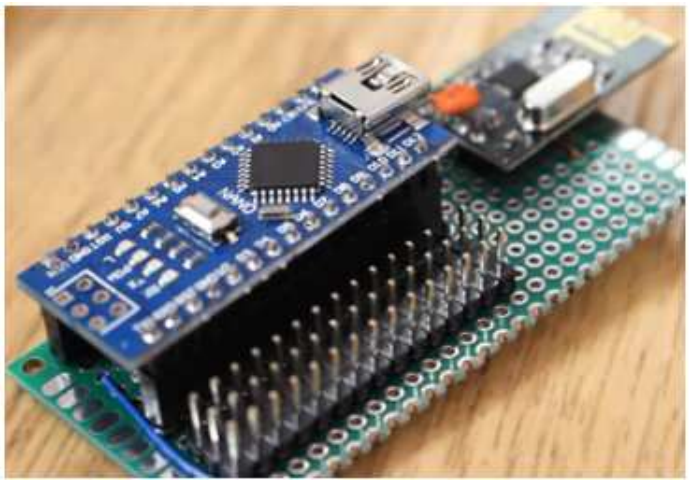


Fig- 17 - Model of Target-2 (receiver)

Image Source: Shot taken from our camera

Methodology

After designing PCB, we would do the following: -

- We would do the circuit simulation manually on bread board and check everything is working fine.
- We solder each and every component to its right position at right pins.
- Then we would do the programming and upload the codes on Arduino Nano.
- Then we would do the testing part which is in next chapter.

Code:- Is in the ANNEXURE.

TARGET-3

GESTURE CONTROL SYSTEM

- After completion of target-2, we will make our self-made transmitter and receiver based on mpu6050 and flex sensor via PCB designing.
- Components required for it are as follows:-

1) **Flex sensor –** It's a flexible band with a sensor that measures the amount of deflection or bending. It will be used for giving throttle to the drone.

➢ It will be attached to the forefinger of the glove.
➢ It's bending will me calibrated with potentiometer.

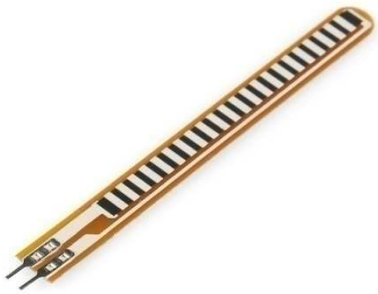


Fig 18: Flex sensor

Image Source: https://learn.sparkfun.com/

2) **MPU6050 –** It's a 3 axis Gyroscope and 3 axis Accelerometer sensors.

➢ It will be attached to the top of the glove.
➢ It's movement will be calibrated with potentiometer for following motion of drone:-
o Pitch (forward & backward) motion
o Roll (Right & Left) motion

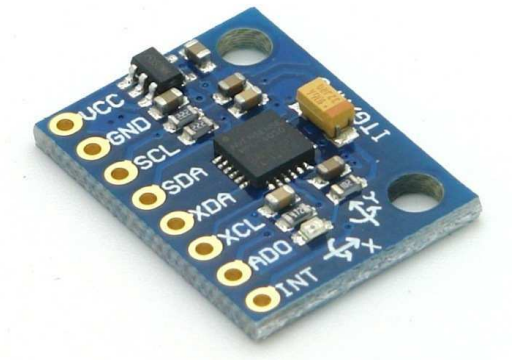


Fig: 19 MPU 6050

Image Source: https://www.google.com/imghp?hl=EN

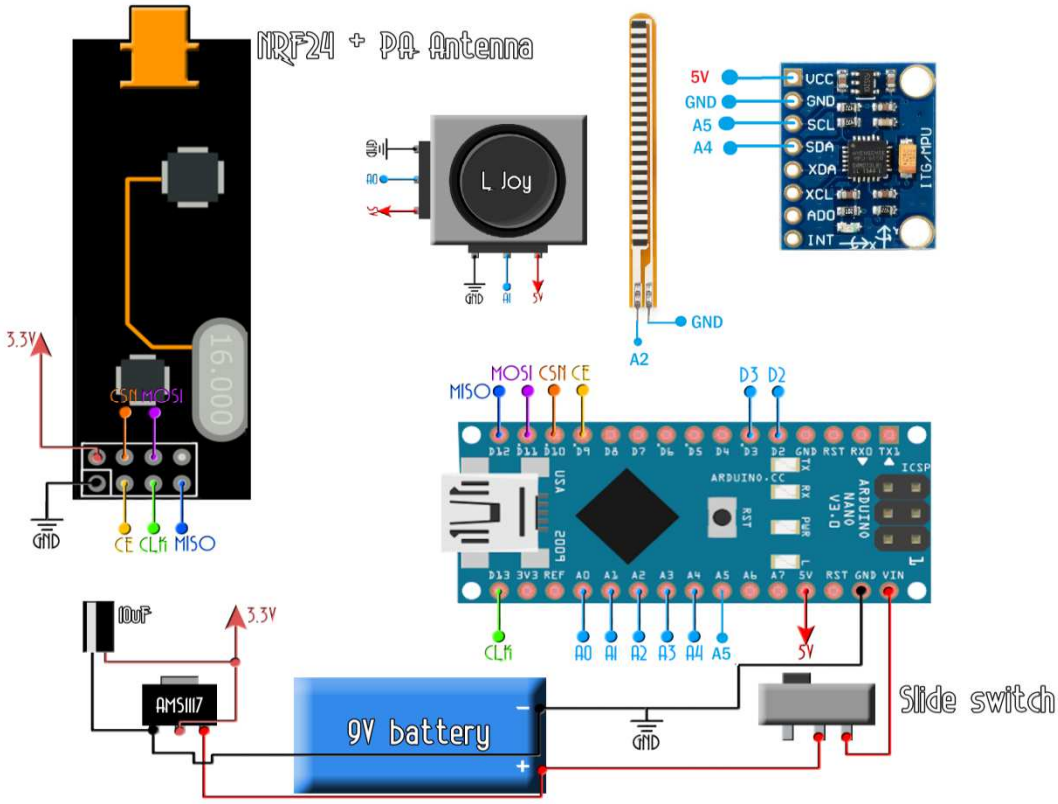
3. Arduino Nano

4. NRF module

5. Voltage regulators

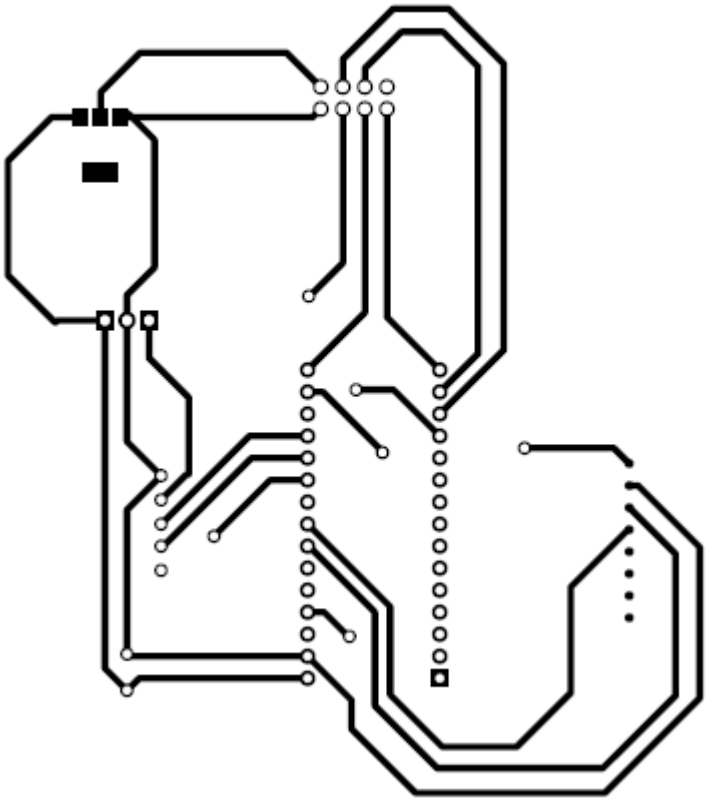
6. Joystick Module

Design For Transmitter



Schematic Diagram

Source: Software Schematic



PCB design

Source: Our designed output from online platform (https://easyeda.com/)

Fig: 20 Models for Target-3 (transmitter)

Source: Picture taken by our camera

Methodology

After designing the PCB, we would do the following things:-

- We would do the circuit simulation manually on breadboard and check everything is working fine.
- We would solder each and every component at its right position at right pins.
- Then we would do the programming and upload the codes on arduinonano.
- Then we would do the testing part which is in next chapter.

Code: - Is in the ANNEXURE.

Chapter 3

Testing & Verification

We have split the testing part in three phases according to design targets of chapter 2:-

Phase 1-Testing

When the drone body assembly is complete, we do some calibrations in the flight controller board:-

1. **Mode selection –** This is done to notify the flight controller board that what kind of drone is to be operated.

   Method:-

   - Switch on the flight controller board.
   - Go to mode selection and choose Quad-Copter with X-type frame.
   - Proceed to next and next to finish and come to back page. Now the mode is selected.

2. **Throttle calibration** – In this every esc is calibrated equally with respect to transmitter throttle stick. This is done so that every motor rotates at equal speed.

   Method:-

   - Switch ON the paired transmitter and take the throttle stick to max.
   - Then keep pressed 1^{st} and 4^{th} button on the flight controller board at the same time and keeping it pressed, power up the flight controller.
   - The screen will light up, and will show throttle calibration.
   - Then by keeping 1^{st} and 4^{th} button pressed on flight controller board, take the throttle stick to minimum on the transmitter. There will be a beep sound coming from the board. It means throttle calibration has been done.

3. **Accelerometer calibration** – This is the feature provided by the flight controller board to have the zero setting for the accelerometer sensor in the board. This is done to ensure the balancing of the quad-copter.

   Method:-

   - Switch on the flight controller board and scroll down to ACC. Calibration.
   - Put the Quad-copter on a flat surface and press the acc. Calibration and start it.
   - The board will calibrate itself in 3-4 seconds and then will show its x,y,z value of accelerometer reading.
   - Hence the drone is calibrated for self balance.

4. **Transmitter zero setting**- This is done to check the signal coming from transmitter at zero setting. This is done to ensure that during stable state of drone, it doesn't get unwanted low signals from transmitter.

   Method:-

   - Switch on the flight controller board and go to receiver test.
   - Then switch on the transmitter by setting everything to zero.
   - If on the receiver test screen if some values fluctuate, it means you transmitter is not set at zero.
   - Then we have to set the transmitter to its zero value through the toggle buttons provided on the transmitter for each stick.
   - Keep toggling until the receiver test screen on flight controller board shows zero value.

5. **PI editor setting:** -This is done to ensure the reaction time of the quad-copter is fast or slow for different movements like pitch, roll, and yaw.

   Method:-

   - Switch on the flight controller and go to PI setting.
   - In Aileron(roll) & Elevator (pitch) section put the following numbers:-
     - P Gain - 75
     - P Limit – 50
     - I Gain – 40

- I Limit – 20

- In Rudder (Yaw) section put the following number for better response:-
  - P Gain - 75
  - P Limit – 20
  - I Gain – 30
  - I Limit – 10

6. **Self-level setting –** This is done for self levelling of the quad-copter.

   Method:-

- Switch on the Flight controller board and go to Self-Level setting.
- Put the following numbers for better control:-
  - P Gain - 70
  - P Limit - 20

Then finally when all the calibration is done we do the flight testing in an open area.

Result & verification

The throttle was equal for all motors.

Acc calibration made the drone stable during take-off.

Piloting the quad-copter was difficult due to weight of the quad-copter.

We went through many crashes which damaged the flight controller and propellers.

Finally we were able to fly it successfully. The maximum flight time was 7-10 minutes for 2200mAH Li-Po battery.

Phase 2 -Testing

After the complete soldering of the components of transmitter and receiver in their respective PCBs we check the following things: -

- Connections are first checked for its conductions among tracks.
- Voltage among every component is checked.
- Their signals are checked.

Result & Verification

- Connections were right and fine.
- There was voltage drop seen on the board. Every component was not getting the right amount of voltage.
    - Output from 9v battery was – 6.76
    - AMS 1117 3.3 voltage regulator – 2.62v
    - NRF input—2.1 v
    - 5v pin of Arduino – 3.58v
    - Joystick module – 3.62
    - 5v voltage regulator- 3.65v
- Signals were not being sent from the transmitter.
- The receiver side was working fine and there was no voltage drops.
- The receiver was able to receive the signal.

Phase 3 -Testing

After soldering the components in the transmitter: -

- Each and every connection is checked.
- Voltage among the components are checked.
- Signals are checked.

Result & Verification

- Every connection is right and fine.
- There was huge amount of voltage drop among the components. Sufficient voltage was not available.
- The voltage readings are as follows: -
    - Output from 9v battery was – 5.94v
    - AMS 1117 3.3 voltage regulator – 2.92v
    - NRF input—2.4 v
    - 5v pin of Arduino – 3.78v
    - MPU6050- 3.66V
    - 5v voltage regulator- 3.86v

Chapter 4

Results & Future scope

The results for target-1 were fine. The drone was hovering and flying with accuracy. Every motion from pitch, roll, yaw and vertical throttling was fine. Though it was little difficult to hold it at certain altitude because of absence of barometric sensor in the flight controller. There is a flight test video submitted to the department.

In target-2, the controller was working fine, though it was not able to effectively control the quad-copter because of inaccurate gimble sensor in joysticks.

In target-3 the design of PCB and soldering was fine and precise. There was some voltage drop issue in the circuit which was due to impedence over the board. The NRF module was not getting enough power for transmitting data.

The future works on this project is to make the gesture based transmitter efficient and functional. Calibration and tuning is to be done with the quad-copter and gesture transmitter. Tying up the loose ends in the circuit is to be done, and improvement in the quality of connection between transmitter and receiver is to be achieved.

Through the innovative perspective, the future work on this project can be done in following domains:-

- A first person view stereo camera can be introduced in it for various purpose like aerial photography, geo mapping etc
- It can be used for delivering small consignments to the desired coordinates with the help of advance flight controller boards which comes with GPS sensors. Flight controller boards like APM, PIXHAWK, DJI NAZA etc
- It can be used for thermal mapping in the field of defence and crime fighting.
- It can be used for spying too.

CONCLUSION

Hence the fabrication of the quad-copter and its hand movement control has been carried out precisely. The flight testing has passed series of flight tests. After so many crashes, it is able to fly. Our goal of this project is to enable the hand gesture basic control mechanism with maximum possible accuracy. We investigated on software development for this project.

The results were not up to the mark on printed PCB due to certain reasons. There is need of more work and time required for the gesture control part.

REFERENCES

1. Peter Dalmaris from Udemy
2. Hackaday
3. Electronoobs

ANNEXURE

Code for Transmitter (Target-2)

```
#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>


const uint64_t my_radio_pipe = 0xE8E8F0F0E1LL;


RF24 radio(9, 10);


// The sizeof this struct should not exceed 32 bytes

struct Data_to_be_sent {

  byte ch1;

  byte ch2;

  byte ch3;

  byte ch4;

  byte ch5;

  byte ch6;

  byte ch7;

};

//Create a variable with the structure above and name it sent_data

Data_to_be_sentsent_data;
```

```
void setup()
{
radio.begin();
radio.setAutoAck(false);
radio.setDataRate(RF24_250KBPS);
radio.openWritingPipe(my_radio_pipe);

  //Reset each channel value
  sent_data.ch1 = 127;
  sent_data.ch2 = 127;
  sent_data.ch3 = 127;
  sent_data.ch4 = 127;
  sent_data.ch5 = 0;
  sent_data.ch6 = 0;
  sent_data.ch7 = 0;
}

/**************************************************/
void loop()
{
  /*If your channel is reversed, just swap 0 to 255 by 255 to 0 below
  EXAMPLE:
  Normal:    data.ch1 = map( analogRead(A0), 0, 1024, 0, 255);
```

```
  Reversed:  data.ch1 = map( analogRead(A0), 0, 1024, 255, 0);  */


  sent_data.ch1 = map( analogRead(A0), 0, 1024, 0, 255);

  sent_data.ch2 = map( analogRead(A1), 0, 1024, 0, 255);

  sent_data.ch3 = map( analogRead(A2), 0, 1024, 0, 255);

  sent_data.ch4 = map( analogRead(A3), 0, 1024, 0, 255);

  sent_data.ch5 = digitalRead(2);

  sent_data.ch6 = digitalRead(3);

  sent_data.ch7 = map( analogRead(A4), 0, 1024, 0, 255);


radio.write(&sent_data, sizeof(Data_to_be_sent));

}
```

Code for Receiver (Target-2)

```
#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>

#include <Servo.h>  //To create PWM signals we need this lybrary

const uint64_t pipeIn = 0xE8E8F0F0E1LL;       //Remember that this code is the same as in the transmitter

RF24 radio(9, 10);  //CSN and CE pins

// The sizeof this struct should not exceed 32 bytes

struct Received_data {

  byte ch1;

  byte ch2;

  byte ch3;

  byte ch4;

  byte ch5;

  byte ch6;

  byte ch7;

};

Received_datareceived_data;

Servo channel_1;

Servo channel_2;

Servo channel_3;

Servo channel_4;
```

```
Servo channel_5;
Servo channel_6;
Servo channel_7;

int ch1_value = 0;
int ch2_value = 0;
int ch3_value = 0;
int ch4_value = 0;
int ch5_value = 0;
int ch6_value = 0;
int ch7_value = 0;

void reset_the_Data()
{
  // 'safe' values to use when NO radio input is detected
  received_data.ch1 = 0;      //Throttle (channel 1) to 0
  received_data.ch2 = 127;
  received_data.ch3 = 127;
  received_data.ch4 = 127;
  received_data.ch5 = 0;
  received_data.ch6 = 0;
  received_data.ch7 = 0;
}
```

```
/**************************************************/
void setup()
{
  //Attach the servo signal on pins from D2 to D8
  channel_1.attach(2);
  channel_2.attach(3);
  channel_3.attach(4);
  channel_4.attach(5);
  channel_5.attach(6);
  channel_6.attach(7);
  channel_7.attach(8);

  //We reset the received values
reset_the_Data();

  //Once again, begin and radio configuration
radio.begin();
radio.setAutoAck(false);
radio.setDataRate(RF24_250KBPS);
radio.openReadingPipe(1,pipeIn);

  //We start the radio comunication
radio.startListening();
```

```
}

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

unsigned long lastRecvTime = 0;

//We create the function that will read the data each certain time
void receive_the_data()
{
  while ( radio.available() ) {
radio.read(&received_data, sizeof(Received_data));
lastRecvTime = millis(); //Here we receive the data
}
}

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

void loop()
{
  //Receive the radio data
receive_the_data();
```

```
//////////This small if will reset the data if signal is lost for 1 sec.
////////////////////////////////////////////////////////////////////
  unsigned long now = millis();
  if ( now - lastRecvTime> 1000 ) {
    // signal lost?
reset_the_Data();
    //Go up and change the initial values if you want depending on
    //your aplications. Put 0 for throttle in case of drones so it won't
    //fly away
  }



  ch1_value = map(received_data.ch1,0,255,1000,2000);
  ch2_value = map(received_data.ch2,0,255,1000,2000);
  ch3_value = map(received_data.ch3,0,255,1000,2000);
  ch4_value = map(received_data.ch4,0,255,1000,2000);
  ch5_value = map(received_data.ch5,0,1,1000,2000);
  ch6_value = map(received_data.ch6,0,1,1000,2000);
  ch7_value = map(received_data.ch7,0,255,1000,2000);

  //Creathe the PWM signals
  channel_1.writeMicroseconds(ch1_value);
  channel_2.writeMicroseconds(ch2_value);
```

```
  channel_3.writeMicroseconds(ch3_value);
  channel_4.writeMicroseconds(ch4_value);
  channel_5.writeMicroseconds(ch5_value);
  channel_6.writeMicroseconds(ch6_value);
  channel_7.writeMicroseconds(ch7_value);


}//Loop end
```

Code for Gesture Transmitter (Target-3)

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <Wire.h>
//these variables are for mpu 6050. angle calculation
int gyro_x, gyro_y, gyro_z;
long acc_x, acc_y, acc_z, acc_total_vector;
int temperature;
long gyro_x_cal, gyro_y_cal, gyro_z_cal;
long loop_timer;
int lcd_loop_counter;
float angle_pitch, angle_roll;
int angle_pitch_buffer, angle_roll_buffer;
booleanset_gyro_angles;
float angle_roll_acc, angle_pitch_acc;
float angle_pitch_output, angle_roll_output;
const uint64_t my_radio_pipe = 0xE8E8F0F0E1LL; // same for the receiver

RF24 radio(9, 10);  //Set CE and CSN pins

// The sizeof this struct should not exceed 32 bytes
struct Data_to_be_sent {
  byte ch1;
  byte ch2;
  byte ch3;
  byte ch4;
  byte ch5;
  byte ch6;
```

```
  byte ch7;
};
//Create a variable with the structure above and name it sent_data
Data_to_be_sentsent_data;
void setup()
{
digitalWrite(13,HIGH); // led will on when calibration is started

Wire.begin();  // begin i2c connection with mpu 6050
radio.begin();
radio.setAutoAck(false);
radio.setDataRate(RF24_250KBPS);
radio.openWritingPipe(my_radio_pipe);

  //Reset each channel value
  sent_data.ch1 = 127;
  sent_data.ch2 = 127;
  sent_data.ch3 = 127;
  sent_data.ch4 = 127;
  sent_data.ch5 = 0;
  sent_data.ch6 = 0;
  sent_data.ch7 = 0;
setup_mpu_6050_registers();
  for (int cal_int = 0; cal_int<2000 ;cal_int ++){
    read_mpu_6050_data();
gyro_x_cal += gyro_x;
gyro_y_cal += gyro_y;
gyro_z_cal += gyro_z;
delay(3);
```

```
  }
gyro_x_cal /= 2000;
gyro_y_cal /= 2000;
gyro_z_cal /= 2000;

delay(2000);
digitalWrite(13,LOW);

Serial.begin(9600);
}
void loop()
{
read_mpu_6050_data();
gyro_x -= gyro_x_cal;
gyro_y -= gyro_y_cal;
gyro_z -= gyro_z_cal;

  //Gyro angle calculations
  //0.0000611 = 1 / (250Hz / 65.5)
angle_pitch += gyro_x * 0.0000611;
angle_roll += gyro_y * 0.0000611;

  //0.000001066 = 0.0000611 * (3.142(PI) / 180degr) The Arduino sin function is in radians
angle_pitch += angle_roll * sin(gyro_z * 0.000001066);
angle_roll -= angle_pitch * sin(gyro_z * 0.000001066);

  //Accelerometer angle calculations
acc_total_vector = sqrt((acc_x*acc_x)+(acc_y*acc_y)+(acc_z*acc_z));
  //57.296 = 1 / (3.142 / 180) The Arduino asin function is in radians
```

```
angle_pitch_acc = asin((float)acc_y/acc_total_vector)* 57.296;
angle_roll_acc = asin((float)acc_x/acc_total_vector)* -57.296;

  //Place the MPU-6050 spirit level and note the values in the following two lines for calibration
angle_pitch_acc -= +1.1;
angle_roll_acc -= -0.0;
  if(set_gyro_angles){
angle_pitch = angle_pitch * 0.9996 + angle_pitch_acc * 0.0004;
angle_roll = angle_roll * 0.9996 + angle_roll_acc * 0.0004;
  }
else{
angle_pitch = angle_pitch_acc;
angle_roll = angle_roll_acc;
set_gyro_angles = true;
  }

  //To dampen the pitch and roll angles a complementary filter is used
angle_pitch_output = angle_pitch_output * 0.9 + angle_pitch * 0.1;
angle_roll_output = angle_roll_output * 0.9 + angle_roll * 0.1;
Serial.print(angle_pitch_output);
Serial.print(" ");
Serial.println(angle_roll_output);
  /*If your channel is reversed, just swap 0 to 255 by 255 to 0 below
  EXAMPLE:
  Normal:    data.ch1 = map( analogRead(A0), 0, 1024, 0, 255);
  Reversed:  data.ch1 = map( analogRead(A0), 0, 1024, 255, 0);  */

  sent_data.ch1 = map(angle_pitch_output , -90, 90, 0, 255);
  sent_data.ch2 = map( angle_roll_output, -90, 90, 0, 255);
```

```
  sent_data.ch3 = map( analogRead(A0), 0, 1024, 0, 255);
  sent_data.ch4 = map( analogRead(A1), 0, 1024, 0, 255);
  sent_data.ch5 = digitalRead(2);
  sent_data.ch6 = digitalRead(3);
  sent_data.ch7 = map( analogRead(A2), 0, 1024, 0, 255);

radio.write(&sent_data, sizeof(Data_to_be_sent));
}
void read_mpu_6050_data(){
Wire.beginTransmission(0x68);
Wire.write(0x3B);
Wire.endTransmission();
Wire.requestFrom(0x68,14);
while(Wire.available() < 14);
acc_x = Wire.read()<<8|Wire.read();
acc_y = Wire.read()<<8|Wire.read();
acc_z = Wire.read()<<8|Wire.read();
  temperature = Wire.read()<<8|Wire.read();
gyro_x = Wire.read()<<8|Wire.read();
gyro_y = Wire.read()<<8|Wire.read();
gyro_z = Wire.read()<<8|Wire.read();

}
void setup_mpu_6050_registers(){
  //Activate the MPU-6050
Wire.beginTransmission(0x68);
Wire.write(0x6B);
Wire.write(0x00);
Wire.endTransmission();
```

```
  //Configure the accelerometer (+/-8g)
Wire.beginTransmission(0x68);
Wire.write(0x1C);
Wire.write(0x10);
Wire.endTransmission();
  //Configure the gyro (500dps full scale)
Wire.beginTransmission(0x68);
Wire.write(0x1B);
Wire.write(0x08);
Wire.endTransmission();
}
```

Price of components used in the project: -

Equipment used		Price
• DJI F450 frame		Rs 700
• 1400KV BLDC motors		Rs 4 x 400 = Rs 1600
• 1045 Propellers		Rs 4 x 100 = Rs 400
• Flight Controller (KK 2.1.5)		Rs 2000
• Flysky Transmitter & Receiver		Rs 3650
• ESCs (30 amp)		Rs 4 x 450 = RS 1800
• 2200mah Li-Po battery		Rs 1500
• Li-Po Charger		Rs 500
• PCBs		Rs 2 x 190 = RS 380
• Arduino Nano		Rs 2 x 300 = Rs 600
• NRF 24L01 module		Rs 2 x 250 = Rs 500
• Joystick module		Rs 2 x 170 = Rs 340
• 3.3v voltage regulator		Rs 2 x 20 = Rs 40
• 5v voltage regulator		Rs 2 x 50 = 100
• 9v battery		Rs 80
• Analog potentiometer		Rs 150
• Switch		Rs 30
• MPU6050		Rs 250
• FLEX SENSOR		Rs 250

Urkund Analysis Result

Analysed Document: .madhu.challa.docx (D51637475)
Submitted: 5/7/2019 5:50:00 AM
Submitted By: abhishek.kumar@dituniversity.edu.in
Significance: 4 %

Sources included in the report:

https://lastminuteengineers.com/nrf24l01-arduino-wireless-communication/
https://electronicshobbyists.com/nrf24l01-interfacing-with-arduino-wireless-communication/
https://www.factoryforward.com/nrf24l01-rf-transceiver-module-communication-arduino/
http://iosrjen.org/Papers/ICPRASET%202K18/surya/Volume%201/aero/2.%2006-10.pdf

Instances where selected sources appear:

7