# ECE445:Team 42
# Project Proposal: Custom Flight Controller for an FPV Drone for Beginners

TA: Jason Jung

By:
Hulya Goodwin (hyg2)
Muhammad Rabbani (rabbani3)
Jaelynn Abdullah (jja8)

Feb 13, 2025

# 1. Introduction

## 1.1 Problem

There are plenty of new drone users with no previous experience in flying a First Person View (FPV) drone. It doesn't make sense for them to purchase a really expensive first drone, as there's a high probability of them crashing it. The drone will most likely break upon multiple crashes, or suffer damage that devalues it through poor weather conditions such as rain as well. When looking at drones to purchase online, they typically fall into the category of 'racing', 'quadcopters', 'GPS', 'FPV', 'Mini'', etc. There is a gap in the market for drones for beginners that are specialized to be cheap and durable.

## 1.2 Solution

To address this gap, and the unique requirements of a beginner's FPV drone, the proposed solution is to create a custom flight controller that will be affordable, accessible, and beginner friendly. The FC will have the basics necessary for an FPV drone (Camera, IMU, Radio Controller/Transmitter), but also a sensor to determine if there is rain to alert the new user to stop flying. The FC will be compatible with the most common Open Source software for FPV drones, which is Betaflight, so that new users can get accustomed to the software they will most likely be using for their more advanced future FPV drones as well. If the user is content with our FC, they can easily continue to use it for their future drones, since they can customize their frames/ESCs because our FC will be compatible with a range of Electronic Speed Controllers (ESCs) and drone frames. In order to showcase this solution, we will use the FC to control an FPV drone.

## 1.3 Visual Aid

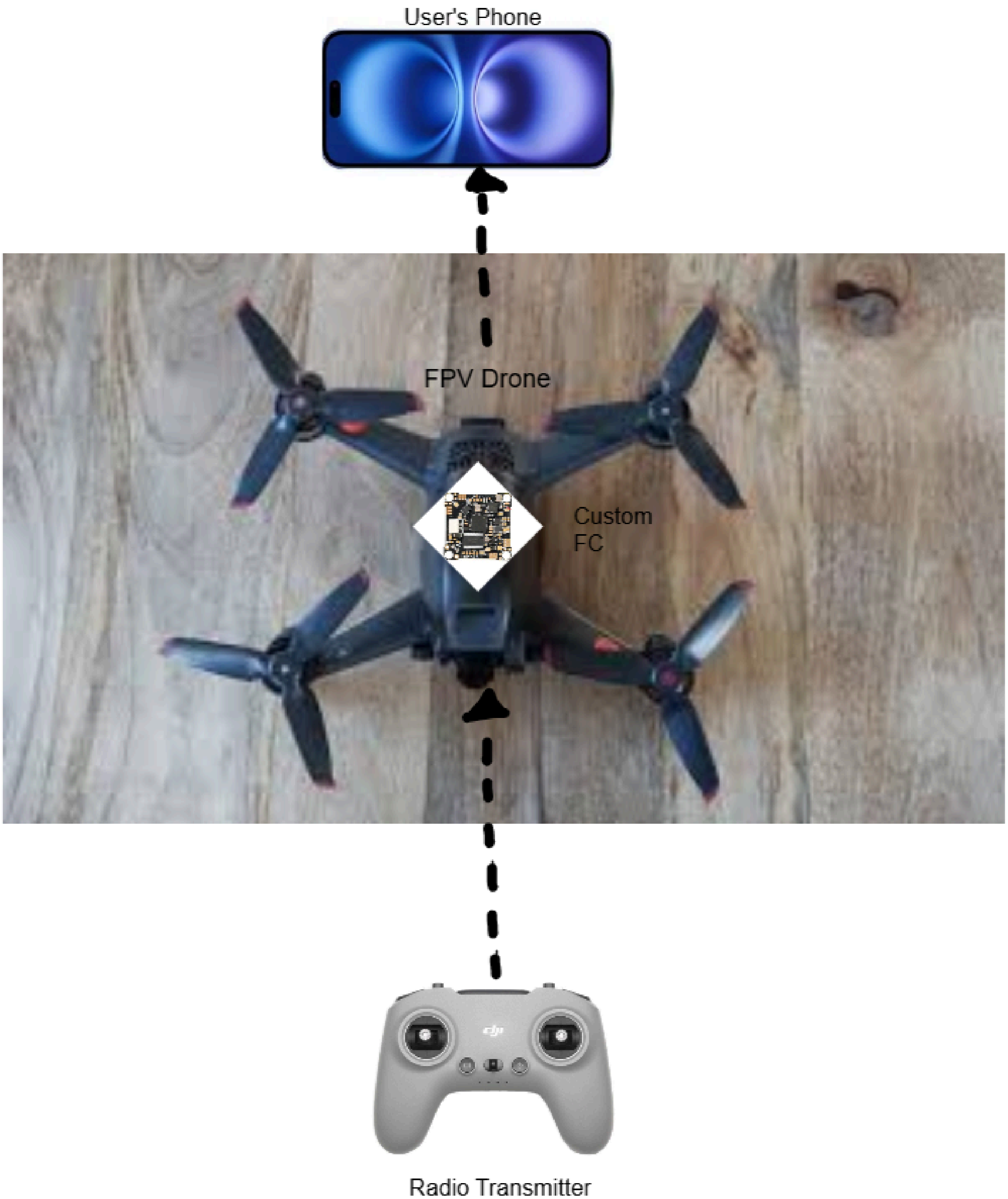


*Figure 1, High Level Visual Aid of Solution*

## 1.3 High Level Requirements

1. Demonstrate a functional flight controller that controls the motors to make balanced motions in the entire 360° plane. Motors can be powered and controlled for up to a full minute. .

2. Demonstrates that the flight controller receives/sends accurate data from/to the microcontroller and integrated sensors. Video can stream 20+ FPS and sensors receive accurate live data within 5% error. Latency is below 100ms.

3. Drone has a functional system that turns on an LED and beeps when humidity sensors sense 90% air humidity or any raindrops. Humidity alert is audible and visible to users up to 10 feet away.

# 2.0 Design

## 2.1 Block Design

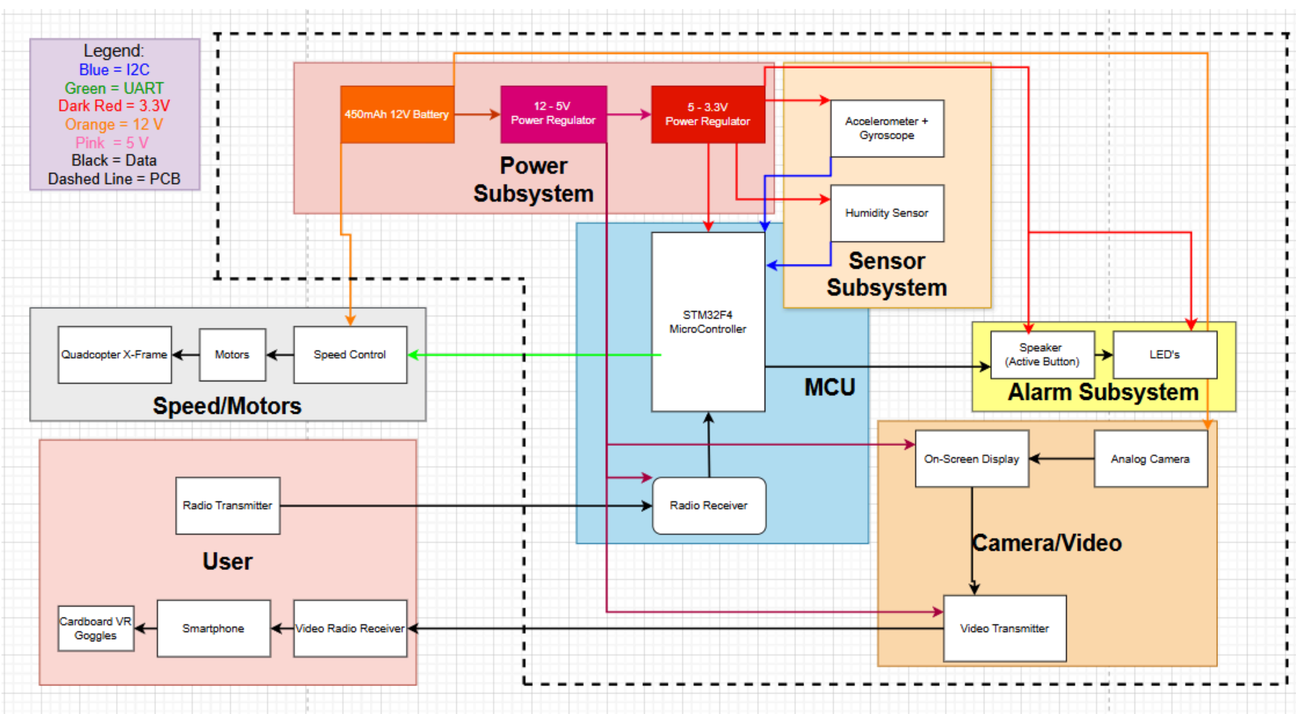


*Figure 2, Block Design of Proposed Project*

## 2.2 Subsystem Overview

**<u>MCU</u>:**

For the main control unit, we are opting for a STM32F405 microcontroller to provide communication between all the subsystems. By modifying the open-source Betaflight software to add additional capabilities, it will be responsible for controlling the ESC to control the speeds of the brushless motors, triggering the alarm system in high humidity conditions, transmitting video data to the user's device, receiving input from the remote controller to change the direction of flight through the radio receiver, and recording IMU data.

**Power:**

The power subsystem is responsible for powering the components on board. The 12V LiPo battery is lightweight and a commonly used component for FPV drones. The total calculated current draw is around 33A with a predicted flight time of around 1.5 minutes. The regulators will step down the voltage to 5V and 3.3V to power the different subsystems.

**Sensors:**

The IMU will measure the drone's acceleration and angle as it is controlled by the user. Using this data, we can detect at what speed the drone is under maximum load, what angle the drone is at to allow the user to correct the flight path, and the position of the drone.

The humidity detector will be a crucial component as it will be connected to the alarm subsystem that alerts the user of a high humidity level and override controls to have the FPV drone hover down and power off to preserve the drone,

**Alarm:**

The alarm subsystem will provide visual and audio cues that are triggered when the rone is in a high humidity environment. The user should be able to hear the speaker and see the LED when they are within 300m radius away from the drone.

**Motors:**

The motors subsystem consists of the speed controller, the brushless motors, and quadcopter frame. Located in all 4 corners of the quadcopter frame will be the brushless motors that are controlled by the ESC to control the direction and speed of travel. The directions of travel will include up, down, left, right, straight, and backwards using the joysticks on the remote controller. Each motor draws up to 8A and must be continuous during flight.

**Camera:**

The camera subsystem is what will define ‘FPV’ for our drown. This allows new users to view what the drone sees. The MAX7456 will be responsible for converting the analog data to be broadcasted through the VTX via radio which will be received by a receiver connected to the user device.

**User:**

On the user end, they will be responsible for controlling the drone via a remote controller that has a transmitter and utilizing a device connected to a receiver for video transmission. The user can view the camera during operation while wearing the cardboard goggles.

## 2.3 Subsystem Requirements

For each subsystem in your block diagram, you should include a highly detailed block description. Each description must include a statement indicating how the block contributes to the overall design dictated by the high-level requirements. Any interfaces with other blocks must be defined clearly and quantitatively. Include a list of requirements where if any of these requirements were removed, the subsystem would fail to function. Good example: Power Subsystem must be able to supply at least 500mA to the rest of the system continuously at 5V +/- 0.1V.

**MCU**:
STM32F4

- Must be able to configure the microcontroller with BetaFlight through our laptop.
- Must be able to take in data from the sensor subsystem i.e. the speed/orientation of the drone at a clock frequency of 400 kHz and the 16 bit level of relative humidity and temperature from the humidity sensor by taking 4 measurements/second.
- Must be able to take in the RF data from the Radio Receiver on how to move the drone through UART.

Radio Receiver

- Must be able to communicate with the user sub-system by receiving data from the radio transmitter (determines direction of movement) within the range of a 500Hz- 1000Hz receive refresh range.
- Must be able to send the received RF data into the STM32 through UART.

**Power:**
LiPo 14.8V 450 mAh Battery

- Must provide continuous current of 33A and deliver 14.8V to power the flight controller
  - Maximum amount of 32A to Motors
  - 1A for remaining sensors and camera subsystem
- Must be able to withstand multiple flights with the drone
- Must be lightweight to allow flight with a frame weight of around 9g
- Voltage Regulator 12V to 5V

- Must provide clean 5V to power receiver and camera subsystem (+/- .3V)
- Voltage Regulator 5V to 3.3V
- Must provide clean 3.3V to power microcontroller and sensor subsystem (+/- .1V)

**Sensors:**

Humidity Detector

- Must be able to accurately represent the relative humidity(RH) percentage in the air around the PCB by taking 6 measurements/second of the RH and saving this value into a 16-bit register.
- Must be able to communicate relative humidity% to the MCU sub-system and send the ALERT interrupt to the STM32 if RH is above 90%.

Accelerometer + Gyroscope

- Must be able to accurately represent the acceleration of the physical drone by taking measurements in the X,Y,Z axis with maximum measurable acceleration before saturation set to +16g (g equals about 9.81 m/s^2)
- Must be able to communicate the acceleration to the MCU sub-system through I2C protocol at 400 kHz..
- Must be able to accurately represent the pitch, yaw, and roll of the physical drone by taking measurements with full scale range +1000%s .
- Must be able to communicate the pitch, yaw, and roll to the MCU sub-system through I2C protocol at 400 kHz.

**Camera/Video:**

Analog Camera

- Must be able to capture 20+ FPS video feed from on top of the drone
- Must be able to live stream the video feed to the OSD in 20+FPS and below 100ms latency.

On-Screen Display

- Must be able to overlay sensor data onto the screen's live feed.
- Must be able to overlay battery percentage onto the screen's live feed.

Video Transmitter

- Must be able to live stream both the video feed and OSD overlay to the video receiver below 100ms latency

**User:**

Video Receiver

- Must be able to receive live stream from the Video Transmitter in 20+FPS quality and below 100ms latency.

Radio Transmitter

- Must be able to wirelessly send data to the Radio Receiver in accordance with physical joystick inputs under 100ms latency.

Smartphone
- Must be able to receive live video stream from the Video Receiver in 20+FPS quality and below 100ms latency.

Cardboard VR Goggles
- Must fit the smartphone and give the optical illusion of VR goggles.
- Must be sturdy enough to support the weight of the smartphone for multiple wears.

**Alarm:**
Speaker
- Must be able to receive high DC input from MCU in response to humidity sensor to create sound of 100 dB audible within 100 ft.

LED's
- Must be able to light up on high DC input from MCU in response to humidity sensor

**Speed/Motors:**
Electronic Speed Control
- Must be able to receive control inputs from the MCU from the Radio Transmitter to send varying PWM signals to speed up or slow down the motors/propellers.

Motors
- Must be able to speed up according to varying PWM signals from the ESC

Quadcopter X-Frame
- Must be able to house motors, propellers, PCB, and battery.
- Must be able to move throughout the 3D plane according to motor controls

## 2.4 Tolerance Analysis

## 2.4.1 - Noise Impacting Video Transmission

The part of the design that's the biggest risk to our design is the video transmission from our analog camera to our smartphone. Between noise from ESC, we need to ensure we have a filter to limit the amount of noise from the circuit for the worst case scenario. With Betaflight, they do not recommend filtering lower than 70 Hz, however, most high frequencies that would affect our VTX occur higher than 150-200 Hz. In addition, we will simulate a noise that has an amplitude of 500 mV. To filter this noise out, we are using a LPF that has a 1k ohm resistor and 470 microFarad capacitor. This configuration will allow an attenuation of ~.5V. With this, we can ensure that the ESC has a clean DC power source, and in the best case scenario that there is no noise, there still will be around 14V being supplied to the ESC, which is within the operating regime.

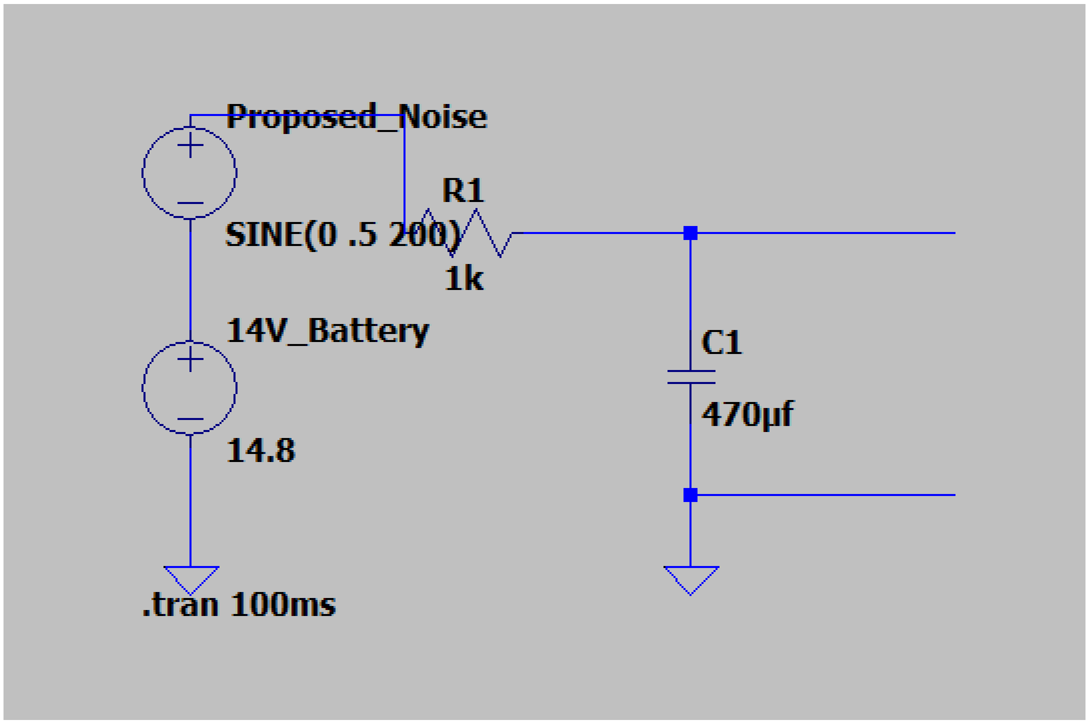


*Figure 2.1, Schematic of LPF*

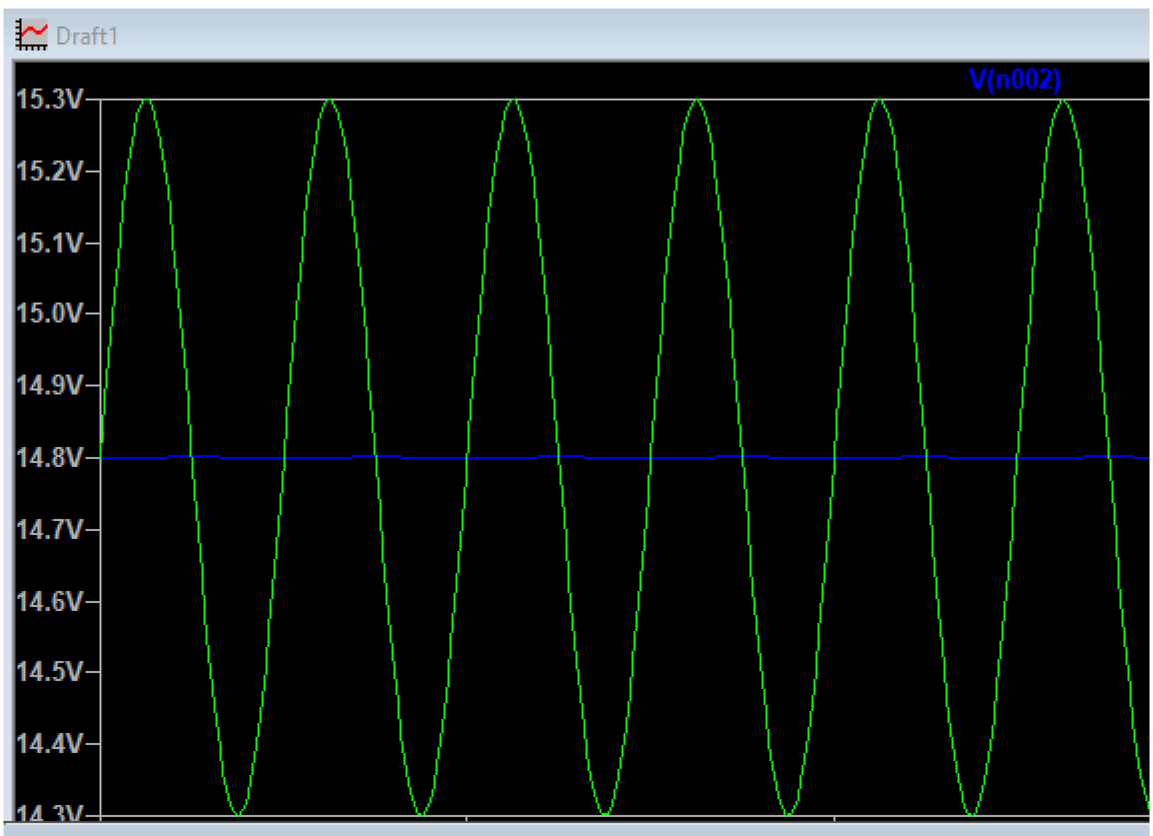


*Figure 2.2, Waveform of voltage input with noise (green sine wave) and filtered output (blue line)*

# 3.0 Ethics and Safety

## 3.1 Ethics

With open-source hardware and software, we have a responsibility to ensure that our final product should be available for the public to continue the development of beginner friendly drones in accordance with IEEE 7.8.I.2 [1]. Moreover, with our criteria, we want to ensure that the user has honest performance reports with the appropriate tolerancing to upkeep safety and the integrity of our project.

## 3.2 Safety

With our project there are important considerations to keep in mind to ensure that the drone targeted towards new users protects the user and the environment in which the drone will take flight. Through our design, we considered thermal generation, safe voltage delivery within the flight controller, and battery management to prevent fire hazards. With the total load from the battery reaching around 33A, ensuring that users do not experience electrical shock while operating the drone is the highest priority. The battery will be housed and the right copper path size will be used to prevent shorting. Any wired connections will be bundled and twisted as needed to prevent loose connections. These and other safety considerations are based around 7.8.I.1 in IEEE's Code of Ethics to ensure the safety and use ethical design practices [1]. Moreover, by recognizing that the users are newer, in accordance with IEEE 7.8.I.6 [1], considerations were made into the design to limit the amount of training experience required to fly the drone and diagnose issues that occur. The alarm system and the built-in protection for the parts we want to order heavily contribute to our goal of protecting our users.

References

[1] IEEE, *IEEE Code of Ethics*, IEEE Standard 7.8, 2020. Available: https://www.ieee.org/about/corporate/governance/p7-8.html