Project Phase 2

Assistant Drone for the Visually Impaired

Thomas Strade - tstrade
Alex Tisdale - atisdale
CSE321

Table of Contents

Hardware Components

The Drone Companion consists of an Arduino Nano, two radar sensors, a gyroscope/accelerometer, an altitude sensor, a power distribution board, and four electronic speed controllers that each connect to their own brushless motor. More detail on each part is below:

Arduino Nano:

The Nano was chosen mostly due to its small size - an important factor on a drone where size and weight are strictly limited. .

Ultrasonic Sensors (HC-SR04):

The ultrasonic sensors each have four pins: a 5V input, an “echo” pulse signal input, a “trigger” pulse signal output, and ground. The front facing radar sensor has its trigger signal connected to Pin 14 on the Nano and its echo signal on Pin 15. The rear facing radar sensor has its trigger signal connected to Pin 16 and its echo signal on Pin 17. The ultrasonic radar sensors allow the drone to read the distance between itself and the user (rear), as well as itself and oncoming obstacles (front). Radar sensors keep this information simple and manageable compared to more expensive and complex sensors, such as LiDAR.

Gyroscope/Accelerometer (MPU6050):

A gyroscope is needed to stabilize the drone in flight, measuring three important spacial variables: pitch (tilting front to back), roll (tilting right to left), and yaw (rotating clockwise or counterclockwise). Similarly, the accelerometer is necessary for consistent and smooth flight movement. Four of the seven available pins from the MPU6050 are connected to the Nano: VCC input, Serial Clock (SCL), Serial Data (SDA), and ground. These are connected to pins 3.3V, A5, A4, and GND, respectively.

Altitude Sensor (BMP280):

There are 5 pins used from the altitude sensor which are 5V and ground. The following pins are connected to the arduino: SPI Clock(SCK) is connected to pin D13, Serial Datta In(SDI) is connected to D11, and Chip Select(CS) is connected to pin D10.
The altitude sensor is to keep the drone at a certain height above the ground.

Electronic Speed Controller (GOUPRC - 2S-4S ESC):

The ESC has three wires, but only two need to be connected to the Ardino. These are the ground and signal wires. The 4 ESC’s signal wires are connected to pins 9, 6, 5, and 3 on the arduino. The input voltage wire is excluded because the ESCs receive power from the LiPo battery, which is distributed by the PDB. The ESC wires connect to the motors through three wires, but any two wires can be switched to reverse the direction of the motor. The ESCs are used to control the calibration and speed of the motors, acting as an intermediary between the Nano and the motors. The ESCs can then set the throttle range at which the minimum and maximum speeds of the motors will be calibrated.

Brushless Motors (Flash Hobby - D-Power Series):

As mentioned under the ESC's description, the motors are connected by the three wires. The decision to use brushless motors over brushed motors is dependent on many factors.. Overall, brushless motors are more efficient due to the lack of friction inside the motor; additionally, brushless motors have higher power density that is important in the drone's take off and general flight capabilities. Other important aspects to mention are the speed control and maintenance. For a drone that is going to follow someone and should stop when obstacles are in the way, the drone will need to react almost immediately, which requires the precise speed control only achieved by the brushless motors. Lastly, excess hardware for the drone needs to be minimized. Brushed motors would introduce the possibility of brushes inside the motor needing to be replaced, a risk that is absent in brushless motors.

Infrared Sensor (IR):

The IR sensor has three pins which are 5V, ground, and the output signal pin. In order for the user to tell the drone to take off and land without coming in contact with the drone itself, there needs to be a wireless solution. At first, RC was considered as an option, but the cost of a controller and receiving was the main reason for taking another route. An infrared sensor is simple, effective, and reliable. Additionally, the only function needed is to tell the drone to take off or land.

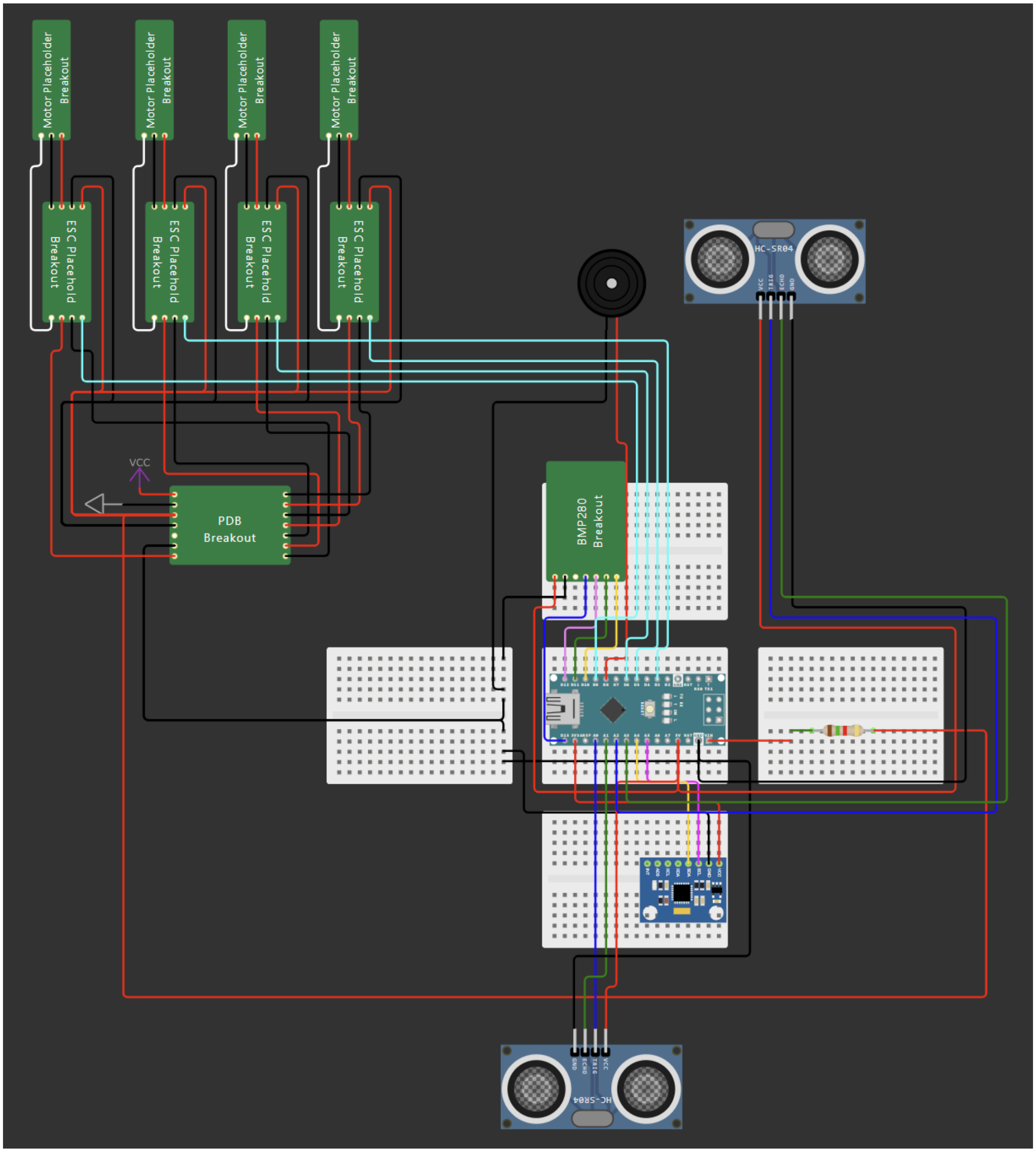
Buzzer:

The buzzer has two pins. One is for grounding and the other is for the input signal connected to Pin D8 on the Nano. The buzzer will emit a sound to alert the user when an obstacle is too close. Since the drone is intended for the blind, the buzzer can be used to aid with locating the drone when it is taking off and landing. Once the drone has landed, the buzzer will beep to allow the user to locate the drone. Different combinations of notes can be played by the buzzer so that the drone can communicate these different scenarios.

Power Distribution Board (PDB) and LiPo Battery:

The 2200 mAh LiPo battery is connected to the PDB, allocating the correct amount of power to the Nano, ESCs, and the other hardware. The 12 V output is cut down by a resistor to approximately 8V to better suit the Nano's input voltage rating of 7 V to 12 V. The PDB also evenly distributes power to each ESC through separate 12 V and ground outputs. Without the PDB, the LiPo battery would most likely damage most, if not all, of the drone's hardware. The battery is rated for approximately 20 minutes of flight time.

Initial Wiring Design (Wokwi):



Software Components

Prototype Details and Project Design

The GitHub repository is split up into Hardware and Software directories. Their contents are as follows:

Hardware:

- Schematic for the design setup
- Specifications on all hardware purchased
- Datasheets
- Pinouts
- 3D printing designs for the drone's frame

Software:

- BMP280 test sketch
- MPU6050 test sketch
- HC-SR04 individual and dual test sketches
- ESC calibration and take off test sketches
- Buzzer test sketch

Codebase

The link to the Drone Companion repository is here.

Test scripts and Patterns

Before all the hardware is controlled in a single master program, a set of tests for each part of the drone's hardware first needs to see success . Each test sets up the inputs and outputs, with the inputs in the testing environments largely depending on human interaction. These tests will indicate whether or not each part works individually and where further troubleshooting needs to take place. The following tests have been developed for the hardware:

Altitude Sensor

This test prints values of temperature, pressure, and its calculated altitude based on where the BMP280 is relative to its startup position. The pressure and altitude's reaction to a change in height is printed to the serial monitor.

Gyroscope/Accelerometer

The MPU6050 is tested by analyzing its many output values of pitch, roll, yaw, and XYZ axis accelerations. When the drone is tilted upwards, the pitch value should increase. When the drone is tilted to the right, the roll value should increase. When the drone is turned left or right, the yaw values should adjust accordingly. For the acceleration, these outputs can be observed by simply moving the drone around and checking that the serial prints make intuitive sense (ex. The default y-axis acceleration is the acceleration due to gravity, 9.81 m/s).

Ultrasonic Sensor

Testing the ultrasonic sensors must consider several important details: the max distance it can read and the correct distance calculation. In this test, an object is held in front of the sensor and its output is printed out to the serial monitor. As the object is moved away from the sensor, the calculated distances increased, as expected. The distance is calculated based on the time between the transmitted trigger signal and the received echo signal, which is turned into either an imperial or metric distance using values defined by the manufacturer.

ESC/Motors

To test the ESCs and motors individually we did not need much human interaction due to there not being any sensors involved. In this test we confirmed that the motors calibrated and can spin at the range we provide it. By coding in a ramp, we can view the speed of each motor increasing to its fastest speed and decreasing to its slowest.

Buzzer

The buzzer will be tested individually to confirm that it works correctly and its output reactive to any input we give it. Two main tests will be used: one will verify that the buzzer produces a variety of sounds, and the other will verify that the buzzer responds to information sent from ultrasonic sensors.

Infrared Sensor

The IR sensor will be tested once the remote is obtained (ideally during the week of 11/11/2024). The IR remote will be tested to verify that it can activate the take off and landing sequences. The remote simply needs to be pointed at the sensor and it will print out the code it received to the serial monitor. These codes will be stored and labeled with their corresponding functions. This test can be repeated to ensure that the signal sent does not change.

Final Testing Plan

The plan to test the final product first involves testing each combination of hardware components working together and to move forward with testing the entire drone itself. From two parts we will build into testing how each piece of hardware interacts with each other in unison. Most of these tests will be between the sensors and the motors.

Gyroscope | ESCs/Motors

The most important relationship we are going to test is the gyroscope and its communication with the ESCs and motors. This will need to be tested because the gyro is what's going to stabilize the drone and allow it to hover in place. To perform this test we will hold the Gyroscope and mount the motors down. As we tilt and turn the gyroscope we will monitor each individual motor to confirm they are changing speed as expected. The point of this test is to prove that the drone will be able to hover in place and stabilize itself.

Altitude sensor | ESCs/Motors

In addition to stabilization, we need to ensure the drone maintains the same height. This test will be performed similarly to the gyroscope test where the motors will be mounted and we will manually move the altitude sensor. We will specify an altitude that the drone should stay at and view the motors speed as we move the sensor. When we move the altitude sensor up, we should see the motors slow down, and as we move the altitude sensor down we should see the motors speed up to bring the drone up to its original position. This test will show that the drone will remain at a set height the entire time its running.

Back Radar | ESCs/Motors

Radar1 will be pointing at the user so for this test we will monitor the motors speed based on the distance we move to and from the sensor. We have marked that motors 1 and 2 will be facing the user while motors and 4 are facing away. That being said, if we move closer to the sensor motors 1 and 2 should increase in speed until they move away from the sensor. If we move away from the radar we should see that motors 3 and 4 speed up until we move closer. This test is to show that the drone will follow the user.

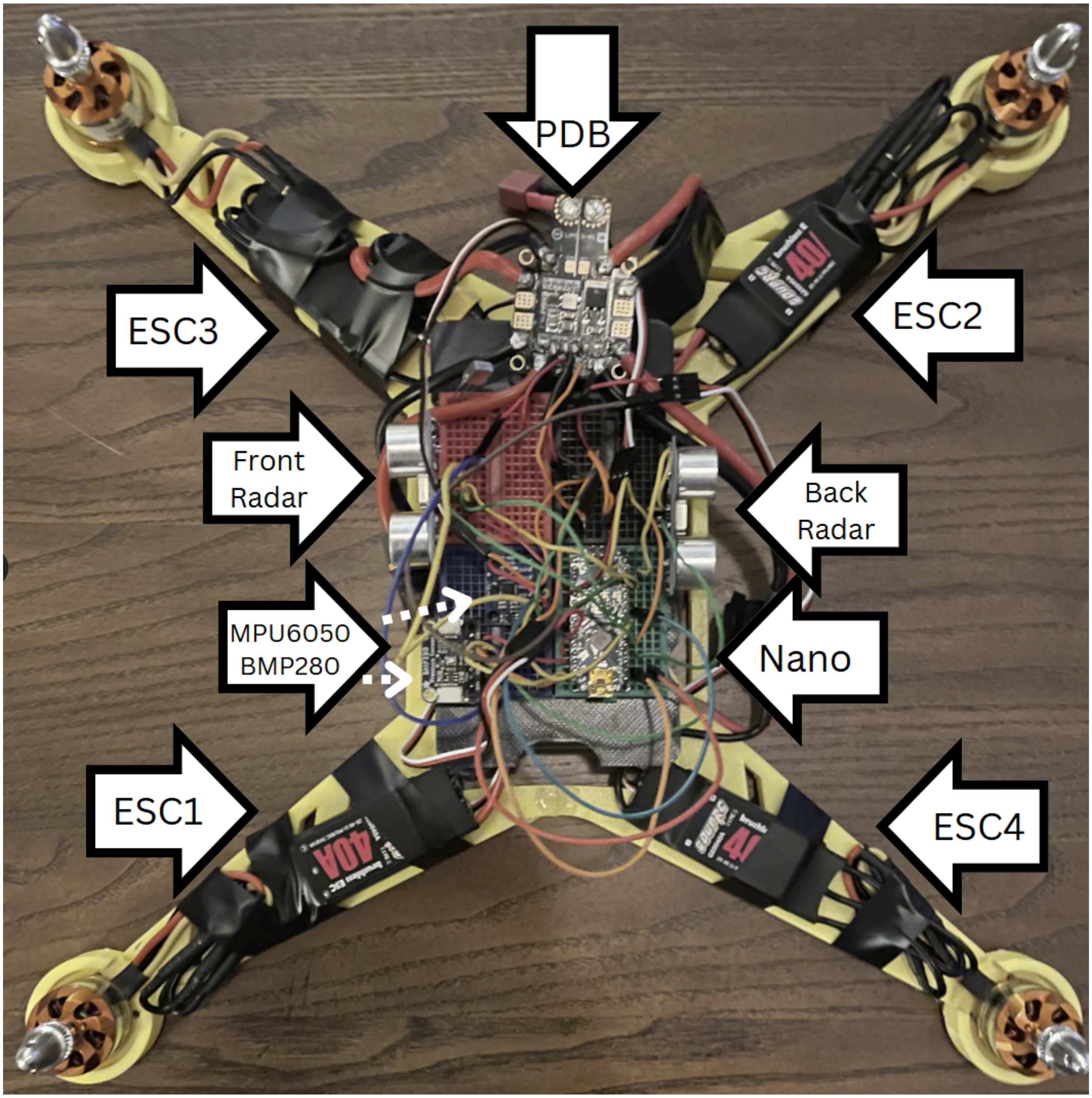
Front Radar | Buzzer

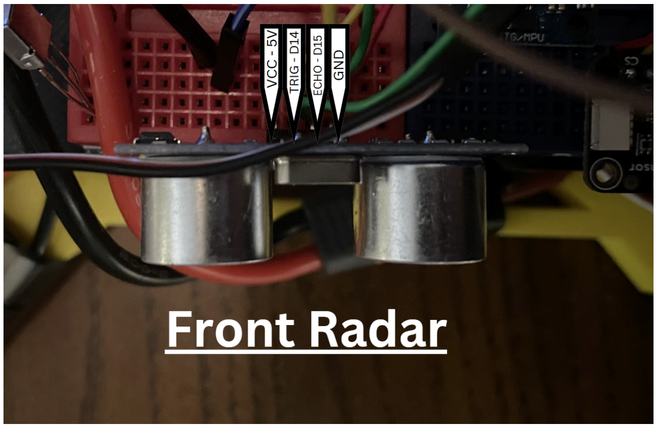
The Front Radar will be pointing away from the user and facing any oncoming obstacles. The test will first ensure a proper distance is set before the buzzer begins warning of an oncoming obstacle, then when the obstacle gets too close for comfort, the drone should avoid moving forward. This test can then be combined with the Back Radar to ensure that the motors and buzzer respond appropriately to both ultrasonic sensor outputs.

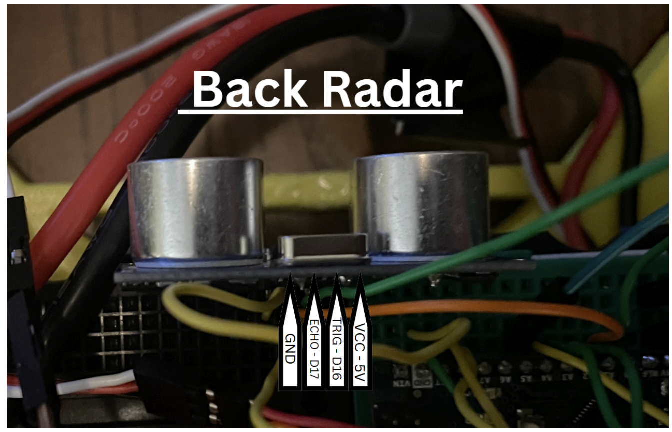
Infrared Sensor | ESCs/Motors

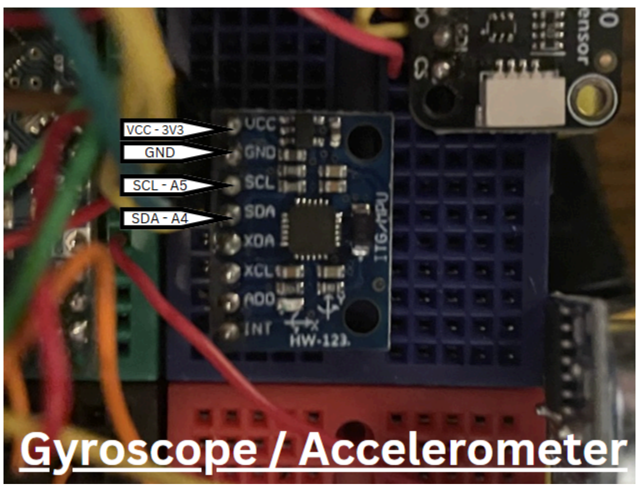
This test will be simple and requires only one reaction. While the motors are spinning, we will press the off button on the IR remote and monitor the motors, ensuring that they slowly decrease in speed and turn off. This test will let us know that the communication between the IR sensor and the ESCs is working properly and that pressing the off button will indeed turn the motors off.

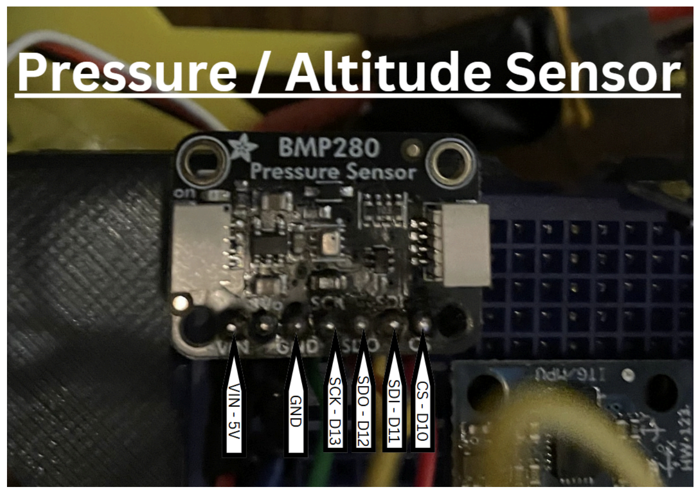
Prototype Images

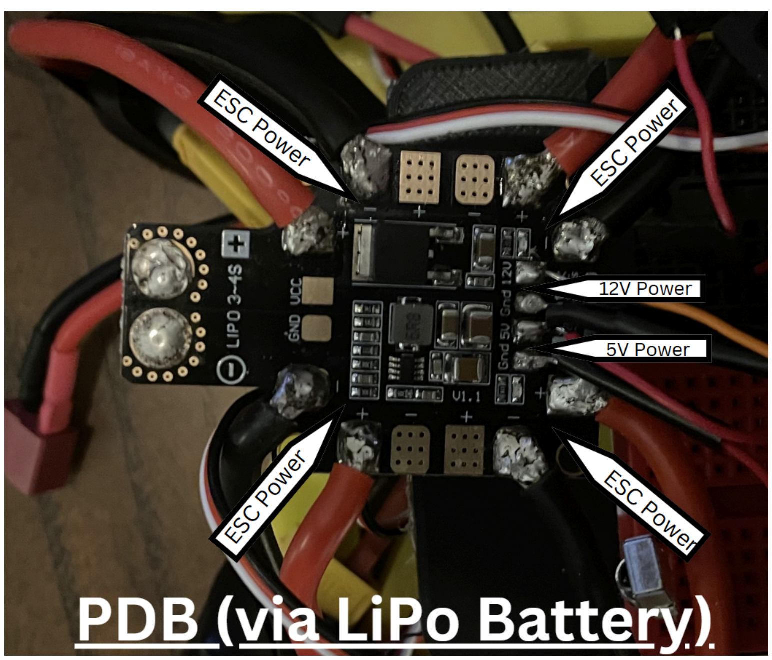



VCC - 5V
TRIG - D14
ECHO - D15
GND
Front Radar


Back Radar
GND
ECHO - D17
TRIG - D16
VCC - 5V


VCC - 3V3
GND
SCL - A5
SDA - A4
VCC
GND
SCL
SDA
XDA
XCL
ADO
INT
HW-123
Gyroscope / Accelerometer


Pressure / Altitude Sensor
BMP280
Pressure Sensor
VIN - 5V
GND
SCK - D13
SDO - D12
SDI - D11
CS - D10


ESC Power
ESC Power
12V Power
5V Power
ESC Power
ESC Power
LIPO 3-4S
GND
VCC
PDB (via LiPo Battery)

Observations and Notes

Currently functioning:

The main hardware components have been individually tested and verified that input and output can be sent and received. The radars require essentially no setup other than attaching their pins during the setup, and currently report distances in terms of centimeters using a small calculation function defined by us. If we consider it necessary, the radar manual gives the formula for reporting distance in terms of feet, but using centimeters better aligns with the 0 to 180 degree rotation signals sent to the motors. Both the radar sensors and motors use the Servo library, which is beneficial to our limited program memory. The ESCs and motors took a bit of debugging to consistently nail down the arming sequence. This is usually done with an RC controller, so we had to experiment with different delays in the setup to ensure the motors were ready to go. After calibration, we have successfully gotten the motors to ramp the speed up and down during the main loop, as well as come to complete stop after a certain condition was met. In this case, the condition was simply an integer flag that was changed at some point during the loop, altering the control flow to then execute the code needed to stop the motors. The gyroscope and altitude sensor have both also been verified as functional. The altitude sensor currently reports its height in feet, but the unit doesn't really matter as long as we can use the value to restrict how high the drone will end up flying. The only setup required is declaring the Adafruit BMP280 object with the Arduino pin number that the sensor's Chip Select (CS) pin is wired to. The gyroscope also uses its own library that provides the MPU6050 object, a calibration function, a basic gyroscope vector reading, and a normalized vector reading.

To be functional:

The buzzer and IR sensor still need to be tested, but these are not our main concern. The biggest challenge is going to be getting the radars, altitude sensor, and gyroscope to all work together to control the drone's flight through the ESCs. Mainly, the gyroscopic calculations and the drone's reactions to this input are one of the biggest barriers in any drone project. Much of the code that exists online is for RC controlled drones, making adapting the code into our project difficult.