The Ultimate Guide to Building a Quadcopter From Scratch

by dakcheungcheng

This project started all the way back when I was in 6th grade and completely new to electronics, when I thought to myself, "Hey, let's build a drone....it shouldn't be that hard......right?" Yea, guess what, it's hard. Almost 4 years later, this project is coming to a close, and along the way, I have built my engineering knowledge from the ground up, and I will be sharing my knowledge with everyone in the Instructables community so that if another 6th grader were to find themselves upon this Instructable, I hope they will not spend nearly as much time trying to build a drone, but still acquire the same knowledge that I have.

This drone tutorial go cover how I built my drone as well as go deep into the nitty-gritty aspects from the construction of the frame all the way to the code. I hope that by reading this instructable, you will be able to fully understand and admire the concepts and theory behind quadcopter building. I would like to thank and recommend some Youtube channels that I have watched and learned from which helped arm me with the knowledge that I have today:

- Joop Brokking: https://www.youtube.com/user/MacPuffdog (He has an AMAZING playlist for drone building)

- Great Scott: https://www.youtube.com/user/greatscottlab (Great electronics information)

- Electronoobs: https://www.youtube.com/channel/UCjiVhIvGmRZixSzup... (Clear explaination of PID)

- Electroboom: rel="nofollow">https://www.youtube.com/user/msadaghd (Funny and also great electronics information)

Without further ado, here's the breakdown of this instructable.

Some parts are written as if a complete beginner was reading it:

- Steps 1-2: Physics and About Quadcopters

- Steps 3-6: Frame Design/Construction

- Steps 7-12: Power Electronics

- Steps 13-15: Control Electronics

- Steps 16-24: Wiring the Quadcopter

- Steps 25-31: Code, Calibration, Setup, and Theory

- Steps 32-34: PID Tuning and Maiden Flight-ish

- Steps 35: Summary, Future Upgrades/Add-Ons, Final Remarks

Note: Though this drone is made of cardboard, this is no slouch when it comes to flying. Fully loaded with the battery, the all up weight (AUW) is just above 800g, and with each motor pushing out up to 740g of thrust. Theoretically, this drone has a thrust-to-weight ratio about 4:1. In perspective, a DJI Phantom has a thrust to weight ratio of about 2:1.

Step 1: Physics

Newton's 3 Laws of Motion

First Law: What is at rest will stay at rest and what is in motion will stay in motion, unless there is an external force acting on it.

Second Law: The force required to accelerate an object to a certain velocity depends on the mass (F=ma)

Third Law: For every action there is an equal and opposite reaction

Based off of the first law, a drone will stay on the unless you decide to move it. A drone will move if you move it.

Based off of the second law, it is clear if you have a lighter drone, you will have longer flight times (less force required to accelerate it), quicker reaction time (if you decide to put more force, the drone will accelerate faster), and better maneuverability.

Based off of the third law, drone propellers throw air down with a certain force, the air molecules will push up on the propellers with the same force, therefore the drone moves up. There is another aspect to this law however, and is the fact that anything rotating will generate a small spinning force in the opposite direction, called torque. Just like how a person needs to "push off" against the ground in order to walk forwards, the motor needs to continuously "push off" against something in order to rotate. That is why if helicopters only have the main rotor, the torque generated by the main propeller will send the body of the helicopter spinning in the opposite direction, and it needs a small rotor on the tail to prevent that.

Step 2: What Is a Quadcopter?

It's a helicopter with 4 propellers.

In order to explain what makes a quadcopter stand out from the rest of the flying contraptions will take a lot more than 6 words. But before we go into that, we need to define some key terms I will be using to describe movement.

- **X-axis:** the axis that points left/right or west/east

- **Y-axis:** the axis that points forwards/backwards or north/south

- **Z-axis:** the axis that points up/down

- **Roll:** when the aerial vehicle tilts left/right; rotates about the y axis

- **Pitch:** when the aerial vehicle tilts forwards/backwards; rotates about the x axis

- **Yaw:** when the aerial vehicle spins in place left or right; rotates about the z axis

- **Degrees of Freedom (DOF):** the number of independent factors that can be assessed (I'll explain this term in better detail later, with an example)

Helicopters have one main propeller that can provide the acceleration, but need the small rotor in the back to counteract the torque created by the big spinning propeller. Additionally, there are countless other control mechanisms to physically tilt the main propeller in order for the helicopter to move in the 2D-plane.

Bicopters have **two** equal-sized **counter-rotating propellers** that provide the acceleration, roll, and the yaw control, but needs the assistance of two servos to tilt the propellers in order to maintain balance and control the pitch.

Tricopters have **three** equal-sized propellers that provide the roll, pitch, and acceleration control. **Two** of the **propellers spin clockwise** and the **remaining propeller spins counterclockwise**. Due to the extra propeller spinning clockwise, there is <u>no control</u> over the **yaw** angle, so a servo must be use to tilt the last motor in order to compensate and gain yaw control.

Quadcopters, with **four** propellers, have the least amount of physical propellers and motors that can allow it to travel in 4 degrees of freedom (move linearly in the x-axis, move linearly in the y-axis, move linearly in the z-axis, and accelerate). However, if one motor fails, it will be the demise of the quadcopter.....unless there is specific code that can compensate.

Hexacopters and **Octocopters** have 6 and 8 propellers, respectively. They offer an advantage over quadcopters that if it loses one propeller, it will still be able to and safely, but at the cost of increase complexity, decreased flight time, higher cost, and less maneuverability.

Unlike planes, quadcopters are naturally unstable and will need software assistance. Overall, the greatest challenge in the construction of the quadcopter will not be the hardware, but instead, the code itself.

Step 3: Frame Selection

The frame makes up around 90% of a quadcopter....so it's kinda important. The first thing you have to consider is the size of the frame you want. Note, the number represents the approximate diagonal arm span of the drone and is measured in millimeters (so if I say I have a 250 frame, the frame is about 250 mm diagonally from arm tip to arm tip). The size of the drone you make depends on your use case, but the general rule of thumb is that the smaller the frame, the more agile and maneuverable the drone is, but it will be less stable. A larger frame will be less agile and maneuverable, but it will be much more stable. In this instructable, I plan to make a size 550 quadcopter.

Step 4: Frame Material

There are a lot of different materials you can use to build a quadcopter, each with their own advantages and disadvantages. Here are some of the most popular materials:

- Carbon Fiber (strongest and lightest, but expensive)
- Wood
- Aluminum
- Foam
- Plastic

But I believe that with some engineering, you can make a working quadcopter frame out of almost anything. So as a result, I decided challenge myself and make my quadcopter out of cardboard (:D).

Step 5: Frame Arm Orientation

There are three main types of quadcopter frames, the "X" frame, the "+" frame, and the "H" frame. There are also some other obscure types like the Deadcat or hybrid H-X frames, but we're not going to go into those.

The "X" frame is the classic drone frame, where the arms are at right angles to each other and the quadcopter looks nice and symmetrical. Most commercial drones like the DJI Phantom are based off of this frame type.

The "+" frame version is virtually identical to the "X" frame - the only difference is the orientation when it is flying in the direction that you call "forwards". The "+" orientation is not as commonly used because the arm sticking out when flying "forwards" will block the view of the camera.

The "H" frame looks like an H and are mainly found on racing drones. An advantage that the "H" frame has over the "X" and "+" is there is a lot more room available to mount the electronics and batteries.

For my drone, I decided to go with the "X" frame.

Step 6: Designing and Assembling the Frame

I planned to make my drone come together like a puzzle with a minimal use of adhesives. All the plates and pieces will slide into each other and are locked with additional arm lock pieces. I also used some zip ties to secure everything together even more.

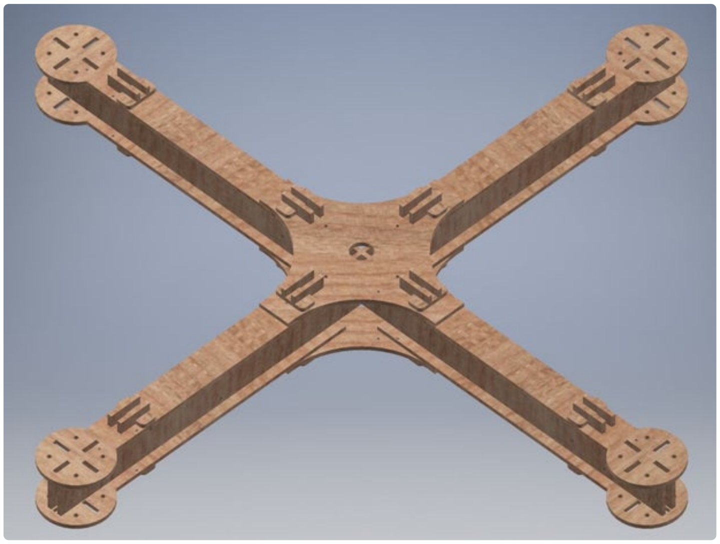
I initially created my idea using a computer-aided design (CAD) software called AutoDesk Inventor Professional and imported the design into AutoCad to laser cut. The DWG files are attached. My frame consists of arms that mimic the shape of I-beams, as it will give cardboard arms the structural integrity the quadcopter arms will need.

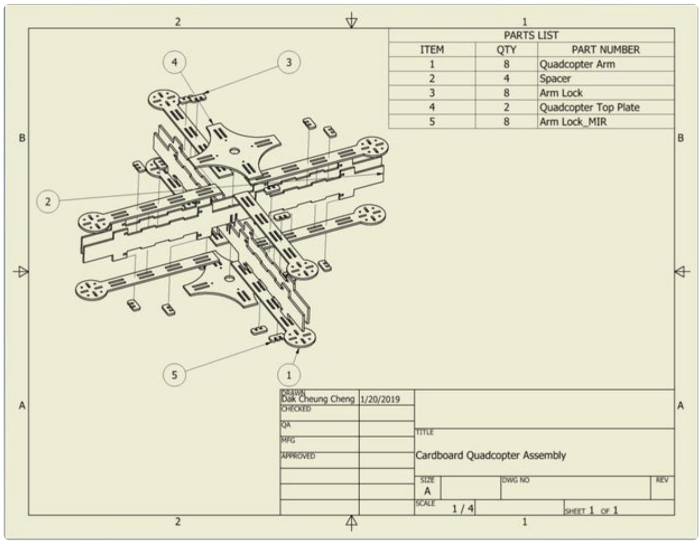
Going to my local makerspace, I laser cut all the pieces out of cardboard and surprisingly, the result was better than I expected.

The bare frame weight is around 98 grams.

Assemble the frame and use zipties to hold everything securely in place.

NOTE: The "Arm Lock MIR" in the drawings are the same thing as "Arm Lock".

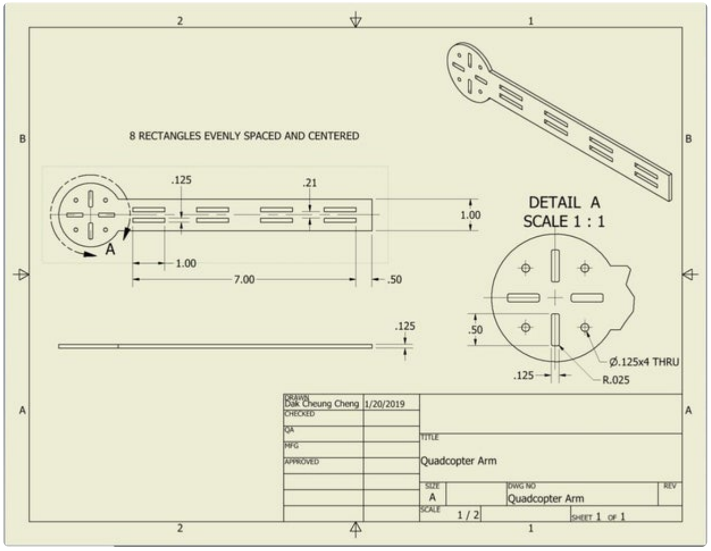




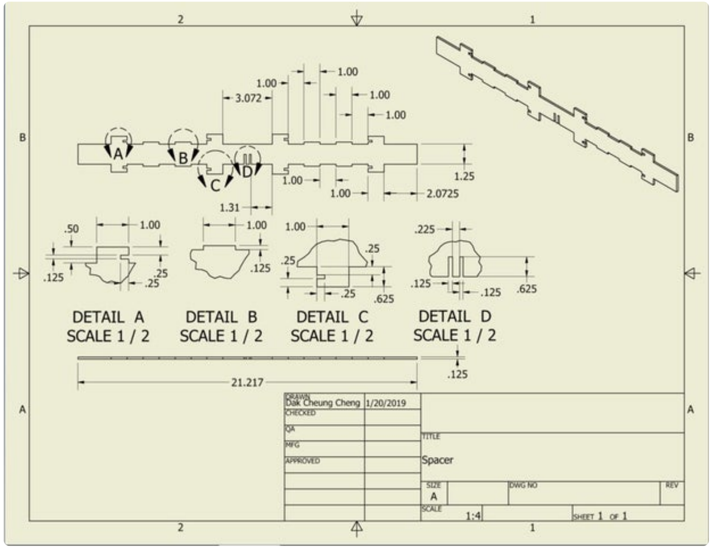
PARTS LIST

ITEM	QTY	PART NUMBER
1	8	Quadcopter Arm
2	4	Spacer
3	8	Arm Lock
4	2	Quadcopter Top Plate
5	8	Arm Lock_MIR

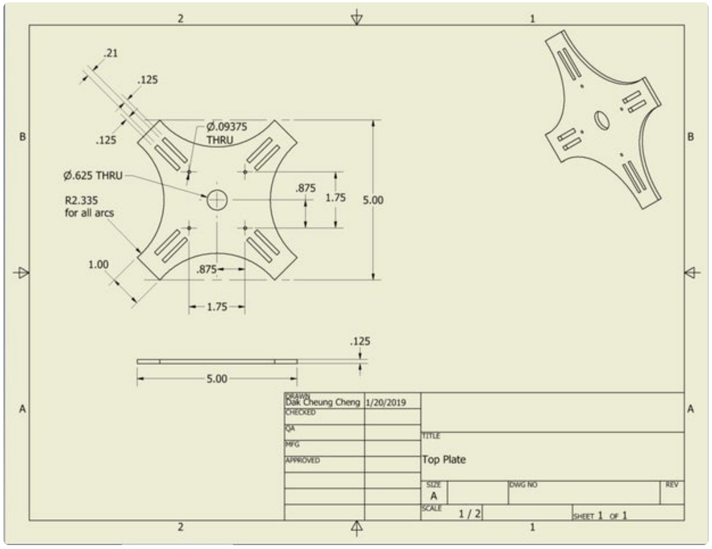
DRAWN Dak Cheung Cheng 1/20/2019
CHECKED
QA
MFG
APPROVED
TITLE Cardboard Quadcopter Assembly
SIZE A
DWG NO
REV
SCALE 1 / 4
SHEET 1 OF 1

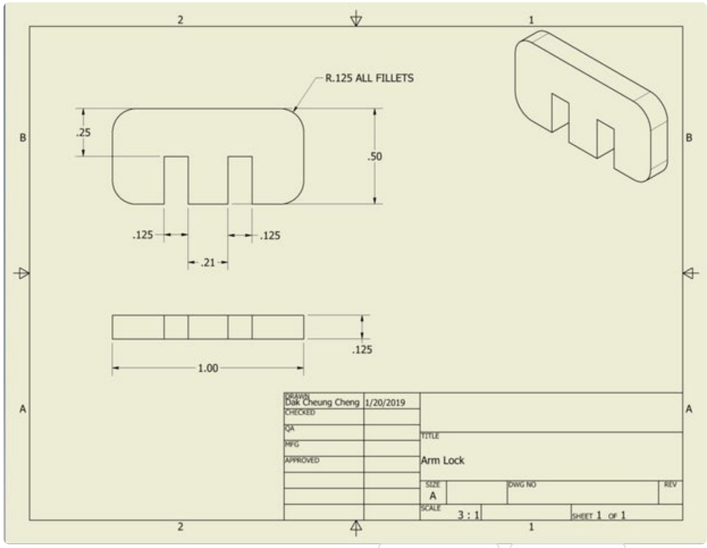


DRAWN Dak Cheung Cheng 1/20/2019
CHECKED
QA
MFG
APPROVED
TITLE Quadcopter Arm
SIZE A
DWG NO Quadcopter Arm
REV
SCALE 1 / 2
SHEET 1 OF 1



DRAWN Dak Cheung Cheng 1/20/2019
CHECKED
QA
MFG
APPROVED
TITLE Spacer
SIZE A
DWG NO
REV
SCALE 1:4
SHEET 1 OF 1





https://www.instructabl... View in 3D Download

https://www.instructabl... View in 3D Download

https://www.instructabl... View in 3D Download

https://www.instructabl... View in 3D Download

Step 7: POWAAAAAA!!!!

Now, onto my favorite part about any engineering project: the electronics!

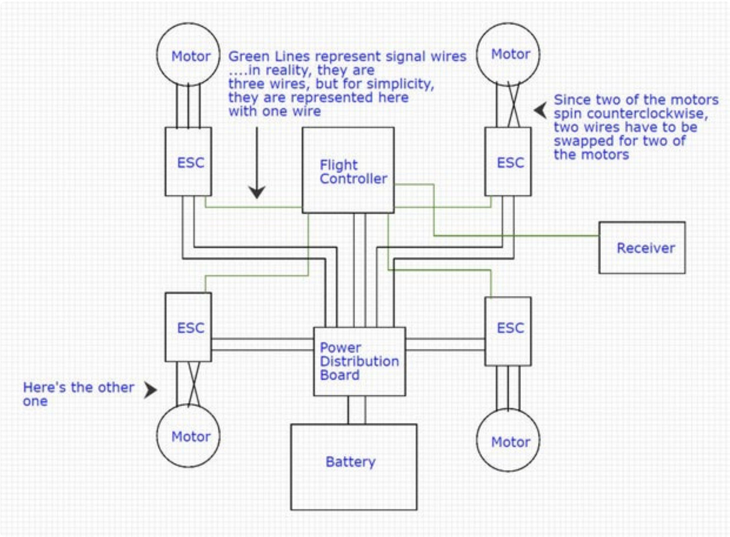
In a nutshell, the main electronics are pretty simple to understand: the battery power gets split and directed between the 4 individual motors and the control board. The control board decodes the signals sent by the controller and taking that information as well as the tilt information into account, controls the motors accordingly.

Refer to the schematics above for more information.

For those who are completely new to reading a schematic........

CRASH COURSE ON HOW TO READ A SCHEMATIC:

Follow the lines to see where and how they connect. Overlapping lines do not mean those wires are connected, unless there is a dot at the intersection.


Motor
Green Lines represent signal wires ....in reality, they are three wires, but for simplicity, they are represented here with one wire
Motor
Since two of the motors spin counterclockwise, two wires have to be swapped for two of the motors
ESC
Flight Controller
ESC
Receiver
ESC
Power Distribution Board
ESC
Here's the other one
Motor
Battery
Motor

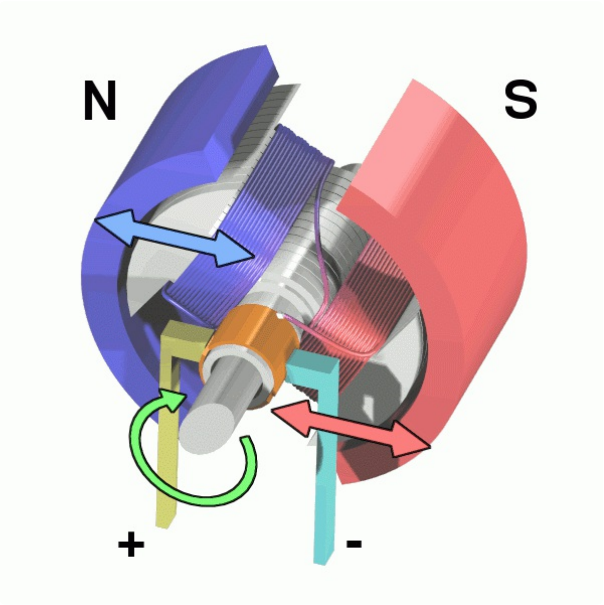
Step 8: Motors and Electronic Speed Controllers

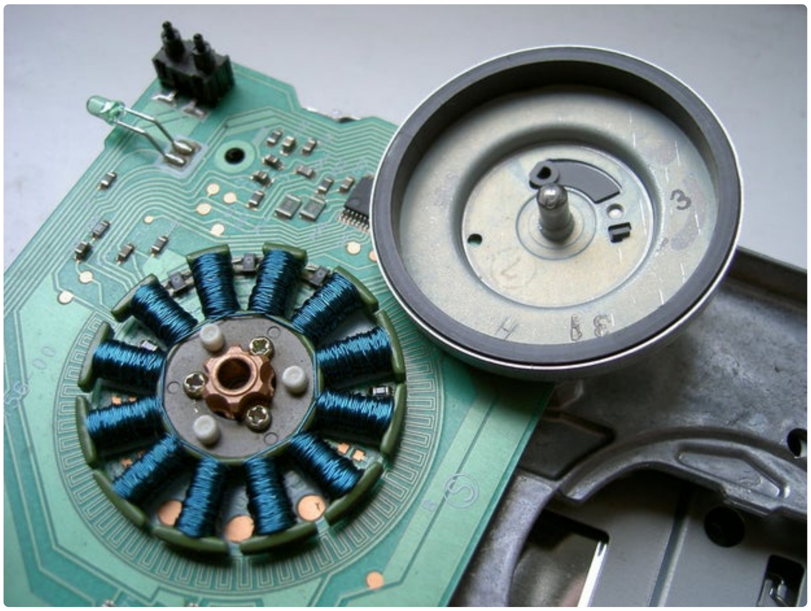
All motors have two parts, the rotor and the stator. The rotor is the part that spins and the stator is the part that stays in place.

Motors come in two main classes: brushed and brushless. Brushed motors have two wires and generally have the coils on the rotor and magnets on the stator. When the coils are energized, it will create a magnetic field that will attract or repel the magnets on the stator, causing the rotor to spin. The coils get the energy through "brushes" and as the rotor spins, the direction of electric flow will change accordingly, keeping the motor spinning. These motors are inexpensive, but generally spin slower and are less efficient due to the friction of the brushes on the rotor. This is why most drones use motors found in the latter class, brushless motors.

Brushless motors have three wires. Due to the structure of brushless motors, they cannot run directly off of the battery. Their structure is opposite of brushed motors, they have the magnets on the rotor and the coils on the stator. Now, the poles of the electromagnet will no longer flip with the rotation of the rotor, so a component called the electronic speed controller (ESC) is necessary. The main job of these devices is to turn on and off the coils in a timed manner in order to keep the motor running. Brushless motors are a lot faster and more efficient as they do not suffer from additional friction from brushes and their speed depends on the switching rate of the ESC. Almost all high-quality drones use brushless motors.

Brushless motors have two subclasses, inrunners and outrunners. The difference between the two are what each motor calls the rotor and stator. Inrunner motors have the rotor on the inside, where only the shaft of the motor spins and the casing outside stays in place. Outrunner motors on the other hand, have the rotor on the outside, so in outrunners, the whole casing spins and only the base of the motor, which the stator is mounted upon, stays in place. Generally, outrunner motors have a lower kV rating.....which is what I will explain in the next step.


N
S
+
-



Step 9: All Those Darn Numbers and Letters....What Do They Mean?

When looking at motors, ESCs, and batteries, you get bombarded with scary numbers with weird units....this step will explain all of those.

Let's say on the motor, you saw this: A2212 1000kV 12N14P

kV: This unit is found on motors and does not stand for "kilovolts". The letter "k" in this case stands for a constant and "V" is for volts. Basically what this tells you is the approximate number of revolutions per minute (rpms) the motor will spin per volt you put in. For example, the motor is 1000 kV, it means that this motor will spin about 1000 rpm at one volt, 2000 rpm at two volts, and so on. Generally, a motor with a lower kV rating will spin at slower rpms, but will have more torque, which is the turning force.

A2212: Ignore the A, but the first two numbers you see (the 22) tell you the diameter of the motor itself or the stator in millimeters. The second pair (the 12) tell you the height of the motor itself or the stator in millimeters.

12N14P: The number in front of the "N" tells you how many electromagnets/coils the stator has and the number in front of the "P" tells you how many permanent magnets are on the rotor.

The following numbers apply to all batteries, ESCs, and motors

A or Amps: This is usually found on ESCs and motor datasheets. This is the amount of punch the ESC can provide to the motor. Make sure that this amount is at least 20-30% higher than the amp rating of your motor. It is important to remember that you cannot shove amps into a motor, a motor will pull as many amps as it needs. SO if your motor only pulls 8 amps maximum, it is not necessary to have a 60 amp esc, because you will never pull anywhere near that 60 amps and you will be carrying around unnecessary weight. However, don't pair a motor that can pull up to 60A with an ESC with a 20A rating because that ESC is going to fry faster than you can say "kV". On ESCs, there is sometimes a "burst" amperage that the ESC can supply for a couple of seconds before irreversible damage. Only use this number for reference.

Battery labels: 3s 2000mAH 20c

mAH or milliamp-hour: This is found on batteries. This is the capacity of the battery, or how much electricity the battery can supply before it dies. Divide this number by 1000 (to convert it to Amp-hour) and multiply by the C rating to get the maximum number of Amps the battery can supply before damage. You can also use the Amp-hour value to calculate a crude approximate of your flight time. In this case, 2000mAH --> 2 AH, meaning that this battery can supply a continuous 2 amps for one hour before dying. Keep in mind that you are never going to discharge a battery all the way, so this is an estimate.

C rating or just C: This number along with the mAH can tell you the maximum amount of amps that this battery can supply. Just like an ESC, there is also a "burst C rating" which is the maximum amount of amps the battery can provide for a short period of time (like about 10-30 seconds). So with the example battery, it can provide 40A of continuous current (2000mAH --> 2 AH --> 2AH * 20C --> 40A).

3s: This is found on batteries and tells you how many cells are wired in series and parallel. Assuming we are talking about Lithium Polymer (LiPo) batteries, each cell has a nominal or "normal" voltage of 3.7v, but when fully charged, they are 4.2v, though some LiPos can go up to 4.35v per cell, and are labelled as LiHV or high voltage. So in this case, 3s means that there are three LiPo cells wired in series, which means this battery has a nominal voltage of 11.1v (3 * 3.7) and a maximum voltage of 12.6v (3 * 4.2). This is also found on ESCs and tells you how much voltage the ESC can handle.

BEC: Battery Eliminator Circuit. It is a circuit found inside ESCs that provide a steady 5v (or 6v...depending on your ESC) to power receivers, the flight controller board, servos, etc. Not all ESCs have a BEC.

Step 10: My Layout

Here are the parts that I have decided to use:

- 1000kV brushless outrunner motors (x4): https://www.amazon.com/Hobbypower-Brushless-Outrun...

- 10x4.5in propellers: https://www.amazon.com/Genuine-Gemfan-10x4-5-Prope...

- 30A Blheli ESCs: https://www.amazon.com/BLHeli-32-Brushless-Control...

- Power Distribution Board: https://www.ebay.com/itm/Multirotor-ESC-Power-Dist...

- XT-60 connectors: https://www.amazon.com/Finware-Female-Bullet-Conne...

- Bullet Connectors (if your ESC does not come with these): https://www.amazon.com/JFtech-Bullet-Banana-Connec...

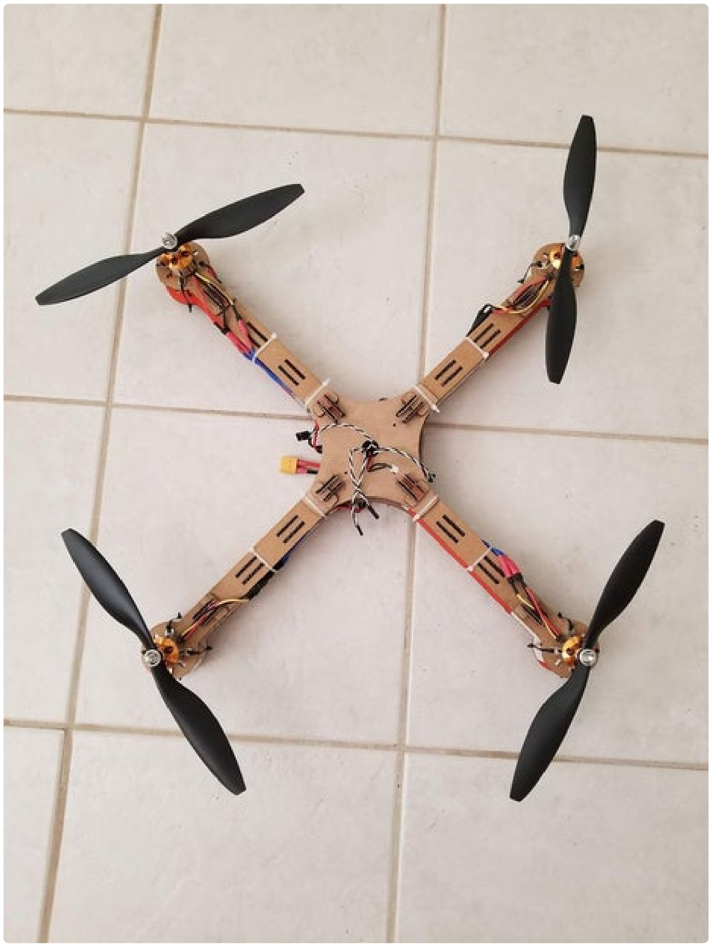
- 30dB earplugs: https://www.lowes.com/pd/Safety-Works-Ear-Plugs/10...

- Heat Shrink Tubing: https://www.amazon.com/NTE-Electronics-47-23448-BK...

- DIY flight controller (which I well get to in Step 14)

- DIY battery pack (which I will explain in the next step)

Tip: Don't go cheap on motors and ESCs, as poor quality parts will result in poor performance. If you are serious into drone building, put a bit of money into quality parts, it will be worth it in the long run. I do not recommend the no-brand motors I got, but I have provided the link anyways.



Step 11: Battery Choice

LiPo batteries are the most popular batteries for drones in general, but they scare me:

//www.youtube.com/embed/aRutyaxmdrs

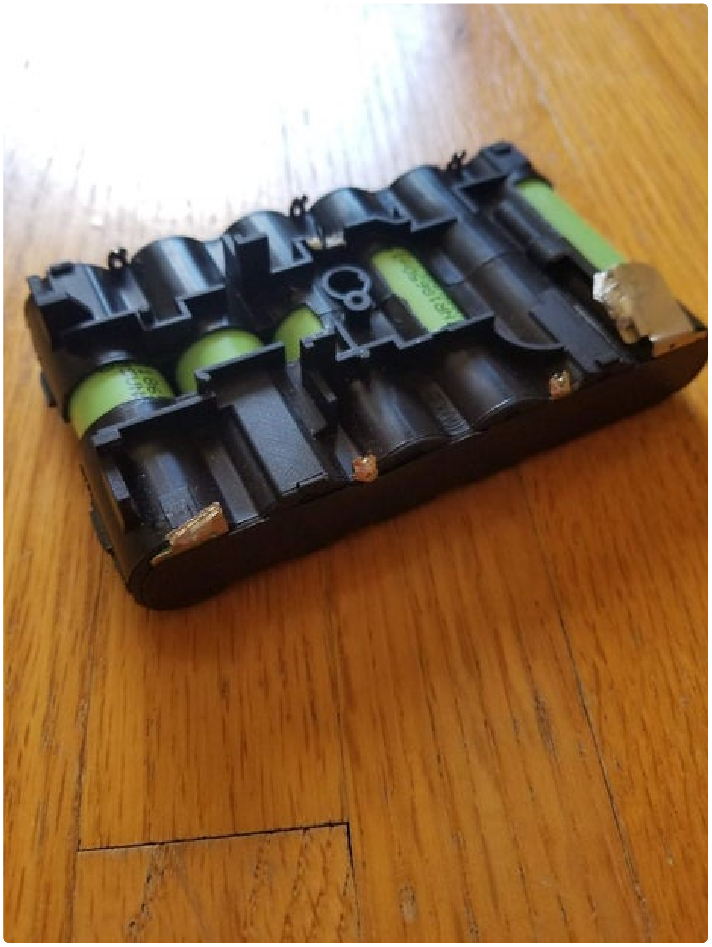
That is why I decided to take the path that not many people have taken: making my own battery pack.

Note, if this is your first drone, I do not recommend building your own battery, it is better to buy a good battery and a good charger.

I salvaged cells from a lithium drill battery pack to build my pack. The cells found in a lithium drill battery pack are called 18650 cells and are also commonly found in old laptop batteries. However, not all 18650 cells are created equal. The ones found in laptop batteries are made to discharge slowly, but can last for a long time. Drills on the other hand, require a lot of continuous power but not for hours on an end. Drones are very similar when it comes to power usage- they need that punch of power, but flying for three hours on end is not a necessity.

I got my batteries from this battery pack: https://www.lowes.com/pd/Kobalt-24-volt-Max-1-5-Am...

Inside the pack, I got Samsung INR18650-15L cells, which according to their datasheet, can provide 18A of continuous current and have a nominal capacity of 1500 mAH. I will wire the pack into a 3s2p pack. The 2p stands for 2 parallel, and when batteries are wired in parallel, the overall capacity (the mAH) and the maximum current (A) will increase, but the voltage will stay the same. Therefore, I will have a battery pack that has a 11.1v nominal voltage, 12.6 v maximum voltage, 3000 mAH capacity (1500 mAH * 2p), and 36A continuous output current (18A * 2p).



Step 12: Battery Construction

I am too poor to afford a spot welder to make my battery. I have to come up with another plan.

So....back to AutoDesk Inventor I go to design a battery pack. I want a modular battery pack that enables to quickly remove the cells when I want to charge them and put them back together for flight.

The CAD drawings and STL files are attached.

The battery holders are 3D printed at my local makerspace (a total of 12 modules were needed).

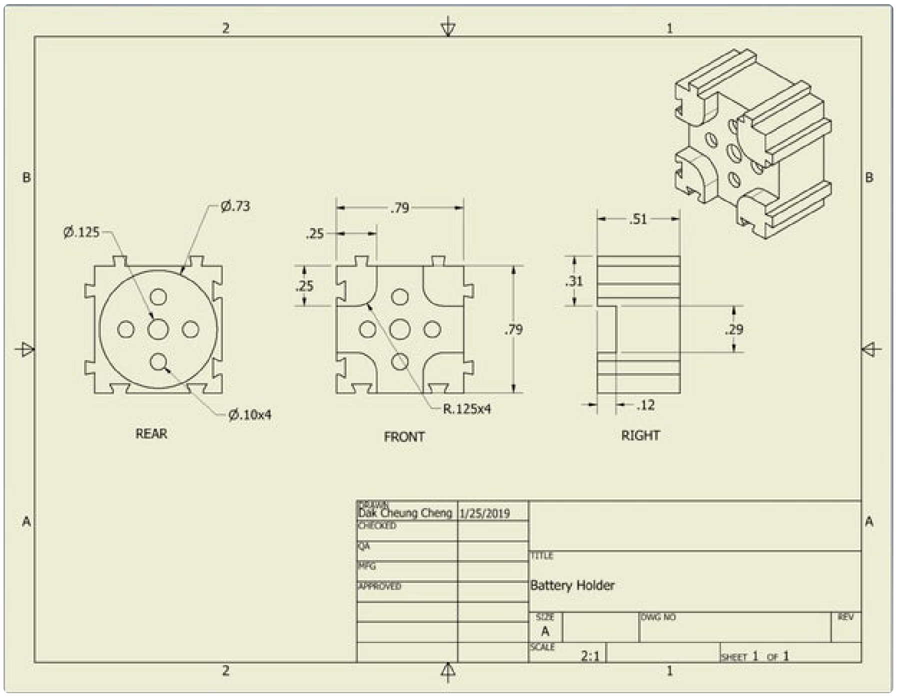
To make the connections within the batteries, I cut strips of metal out of aluminium soda cans and used aluminum rivets to secure the soda can and the power wires to the battery holders....and as a bonus, the heads of the rivets act as contact points for the batteries. The pack will be physically held together with thick rubber bands.

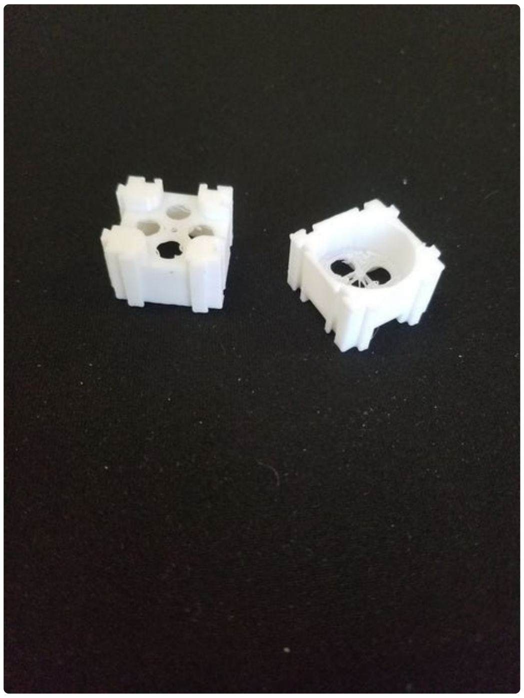
(Note: Check step 34 for updated battery design)

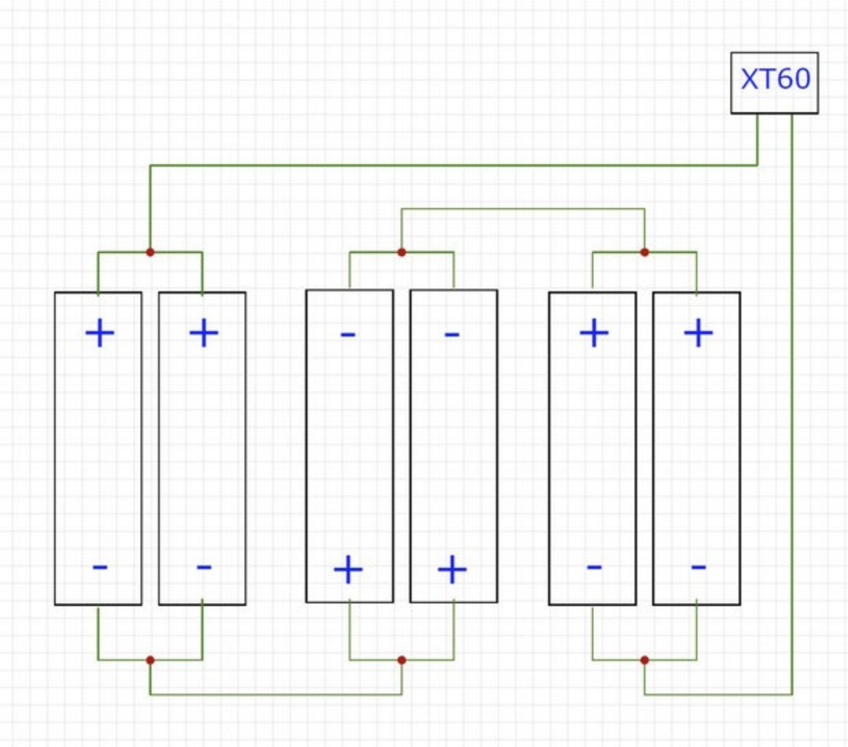
IMPORTANT: IF THIS IS DONE INCORRECTLY, IT MAY CAUSE A LOT OF DAMAGE:

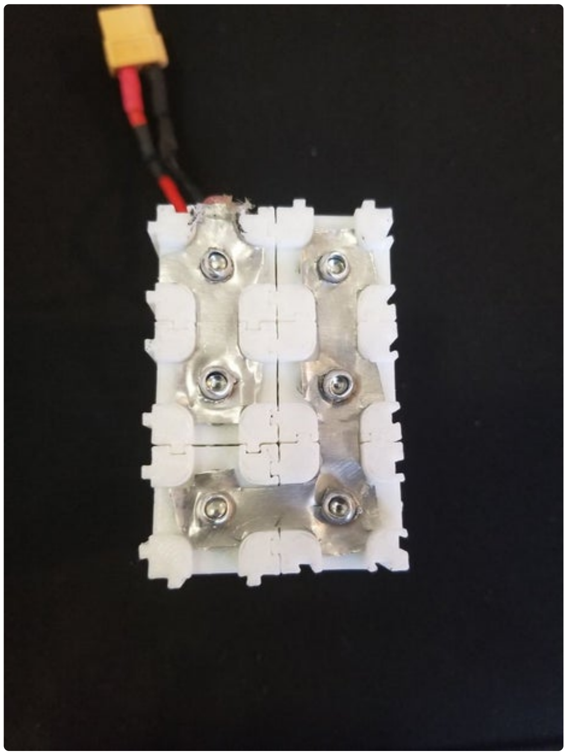
Create two identical "plates" of the battery holder, place the batteries in the orientation as shown, and place the second plate EXACTLY as shown in the drawing. Use the schematic of the battery pack to make sure you are doing it right.

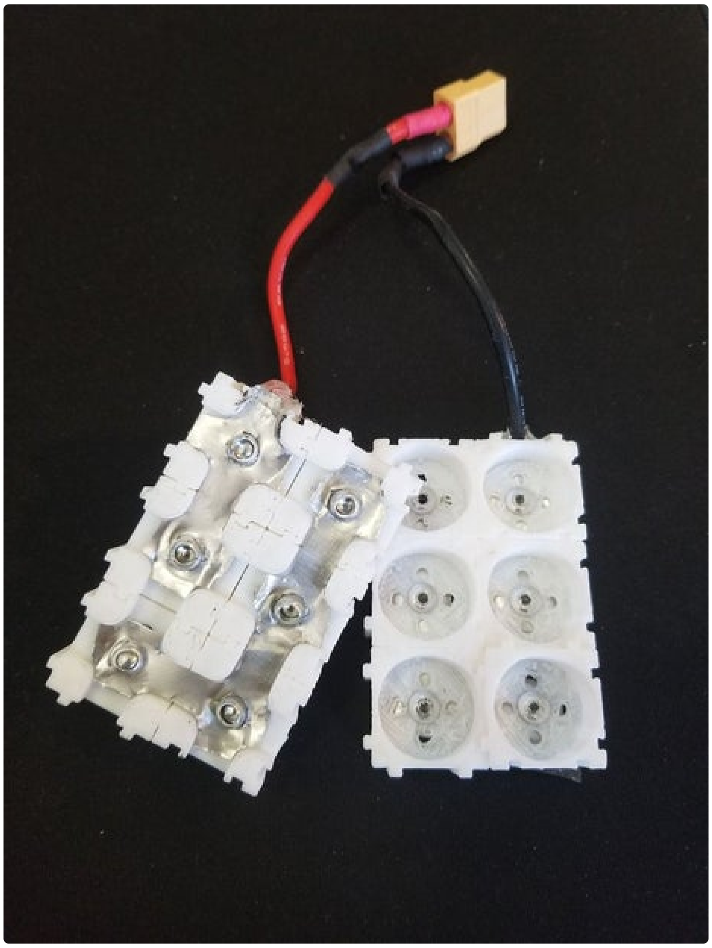
To charge the battery, I take it apart and charge the cells individually using a charger like this: https://www.walmart.com/ip/Tbest-4-Slots-Battery-C....

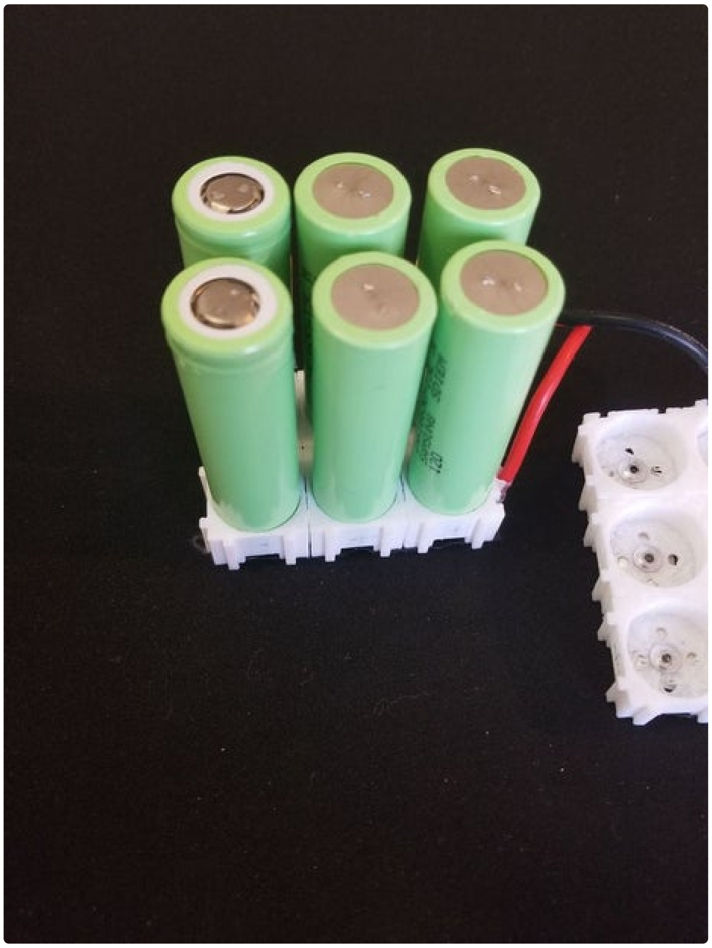

2
1
B
A
Ø.125
Ø.73
Ø.10x4
REAR
.79
.25
.25
.79
R.125x4
FRONT
.51
.31
.29
.12
RIGHT
DRAWN
Dak Cheung Cheng
1/25/2019
CHECKED
QA
MFG
APPROVED
TITLE
Battery Holder
SIZE
A
DWG NO
REV
SCALE
2:1
SHEET 1 OF 1

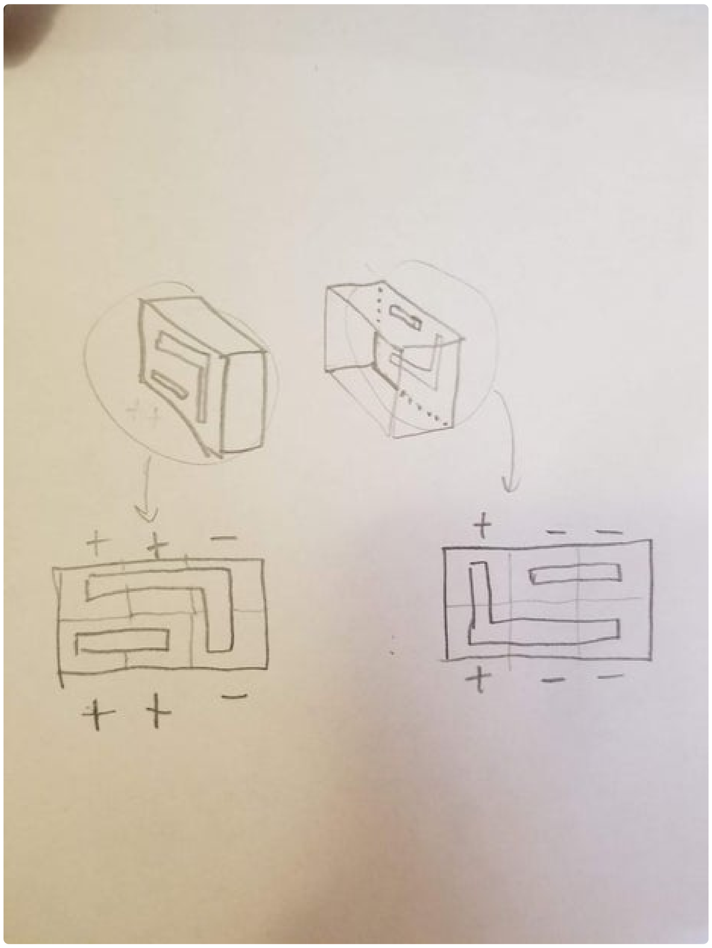


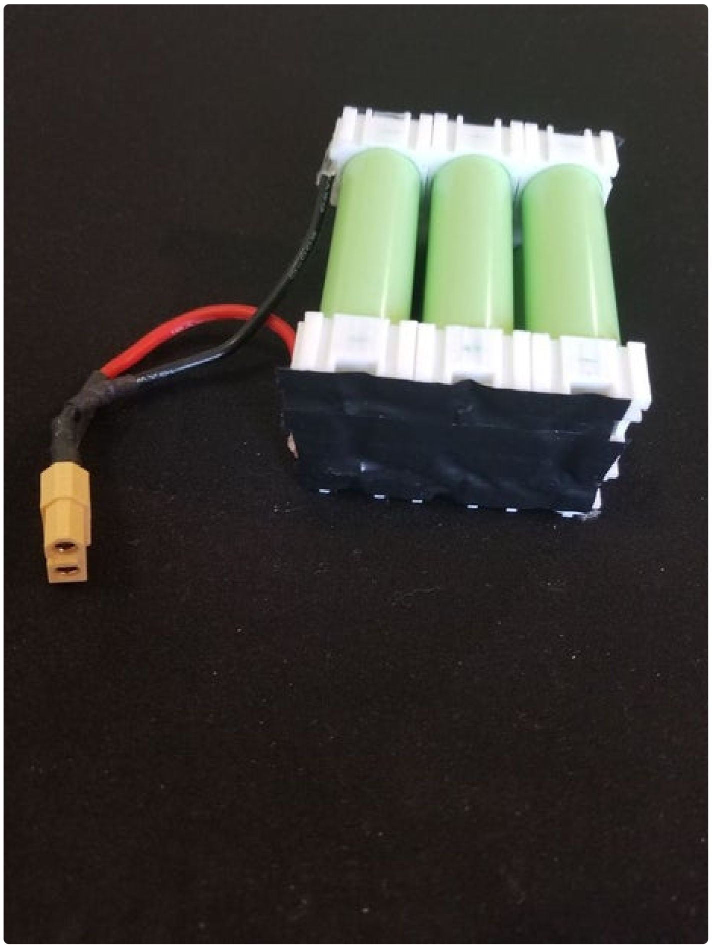

XT60
+
+
-
-
-
-
+
+
+
+
-
-











https://www.instructabl...

View in 3D

Download

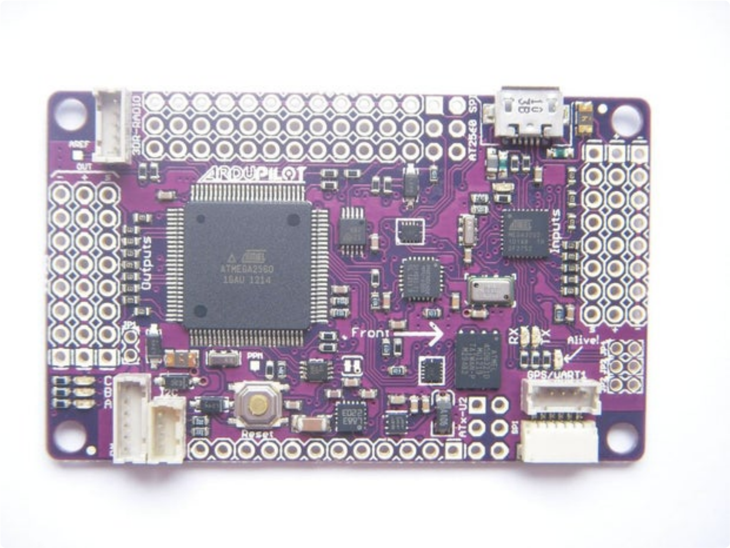
Step 13: Control

There is this ad I've seen somewhere that states "Power is nothing without control", and that is very true when it comes to quadcopters.

Now for many beginners, it is good enough to go buy a commercial flight controller board like an Ardupilot, which is based off of the Arduino Mega and can be programmed with the Arduino IDE (https://www.newegg.com/Product/Product.aspx?Item=9...), NAZA 32 (rel="nofollow">https://www.ebay.com/i/322013155847?chn=ps), etc and use that instead. You will probably get better performance that way, but what is the fun in that?

So being the naive child I am, I decided to make my own flight controller...how hard can that be?

Sigh.



Step 14: Flight Controller

This is the brains of the drone. Let's make it. I will assume soldering skills are present.

You will need:

- 1x 5v Arduino Pro Mini (or clone): https://www.microcenter.com/product/486546/arduino...

- 1x GY-87 10 DOF triple axis gyroscope, accelerometer, magnetometer, and barometer: https://www.amazon.com/GUWANJI-MPU6050-HMC5883L-BM...

- Resistors

- 2.2k resistor

- 6.8k resistor

- 40x Header pins

- 1x LM7805 5v voltage regulator (I dug mine out of an old ESC which I burned up): https://www.adafruit.com/product/2164

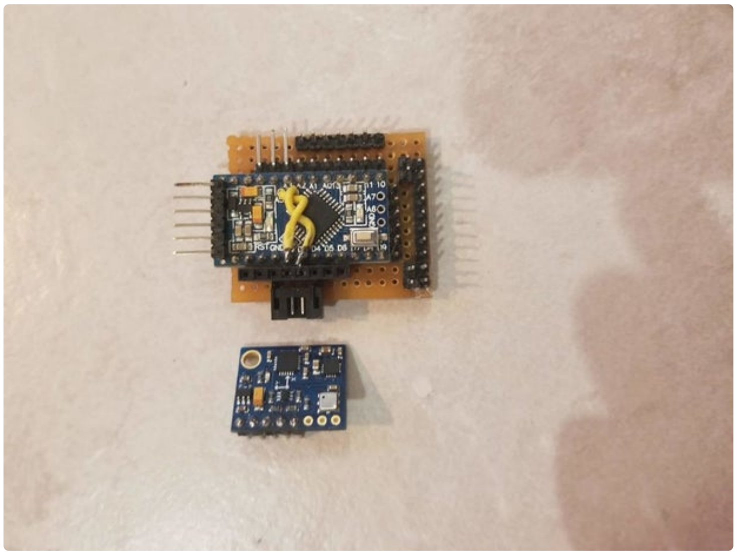
- Wires

- A ton of programming skills

- Perfboard

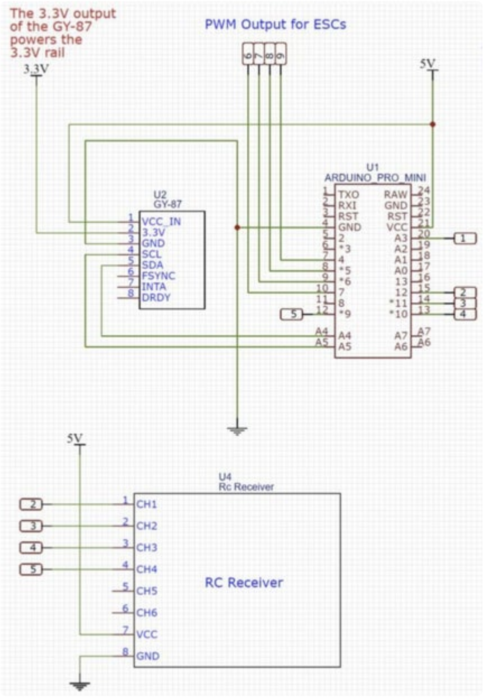
- 2.4 Ghz transmitter and receiver (at least 4 channels, I used a Flysky FS-i6X: https://www.amazon.com/Flysky-FS-TM10-Transmitter-...)

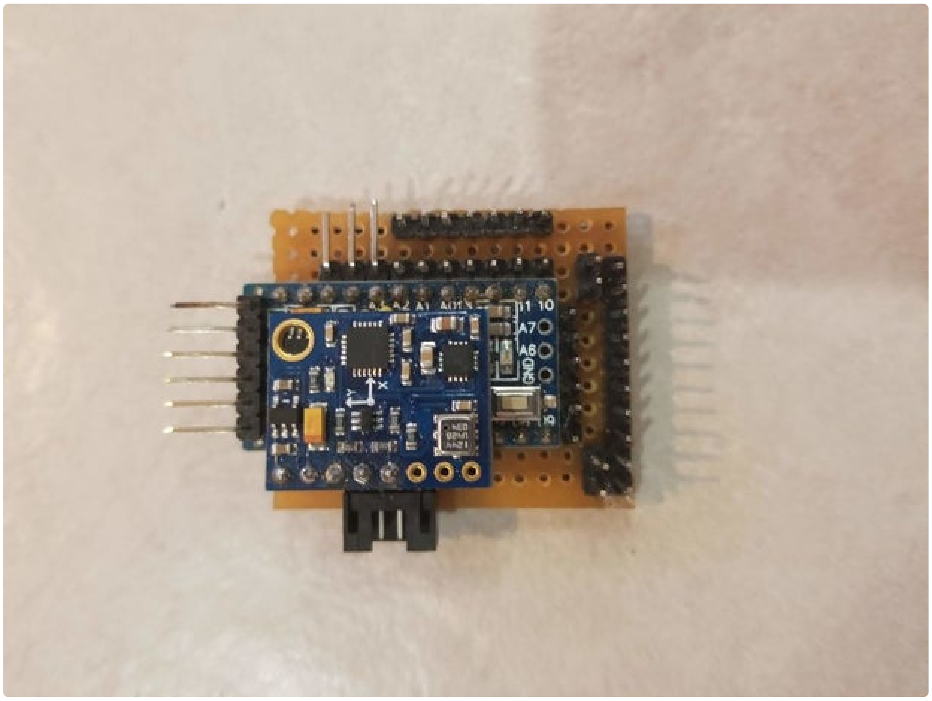
Most components within the quadcopter communicate via PWM. You will see wire "ribbons" (called servo connectors) with three wires: red, black, and white (or sometimes, orange, red, and brown). Red is for 5V (positive), white/orange is for signal, and black/brown is for GND (negative). I will explain the communication methods in the "Theory" section.

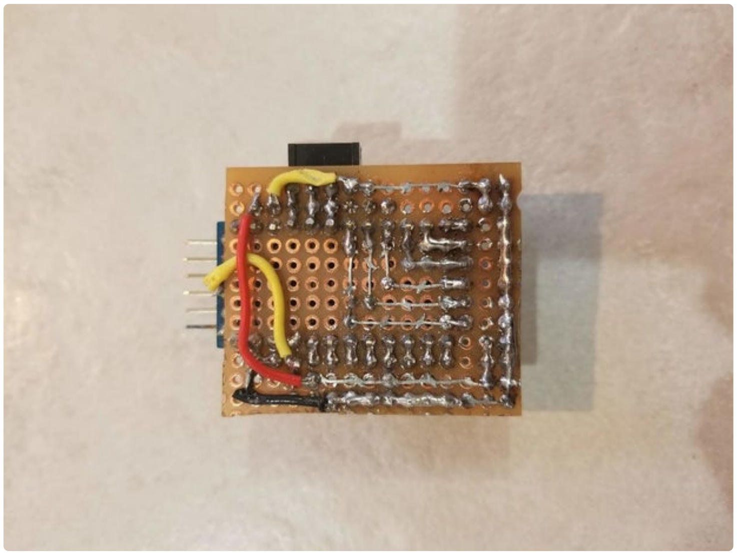


Step 15: Flight Controller Schematic

The Arduino Pro Mini is powered by the 5v regulator (built into the voltage sense wire...look at step 18) and communicates with the GY-87 gyroscope via the I2C bus. The voltage of the battery is read through analog input pin 3 (A3) through a voltage divider (also built into the voltage sense wire).

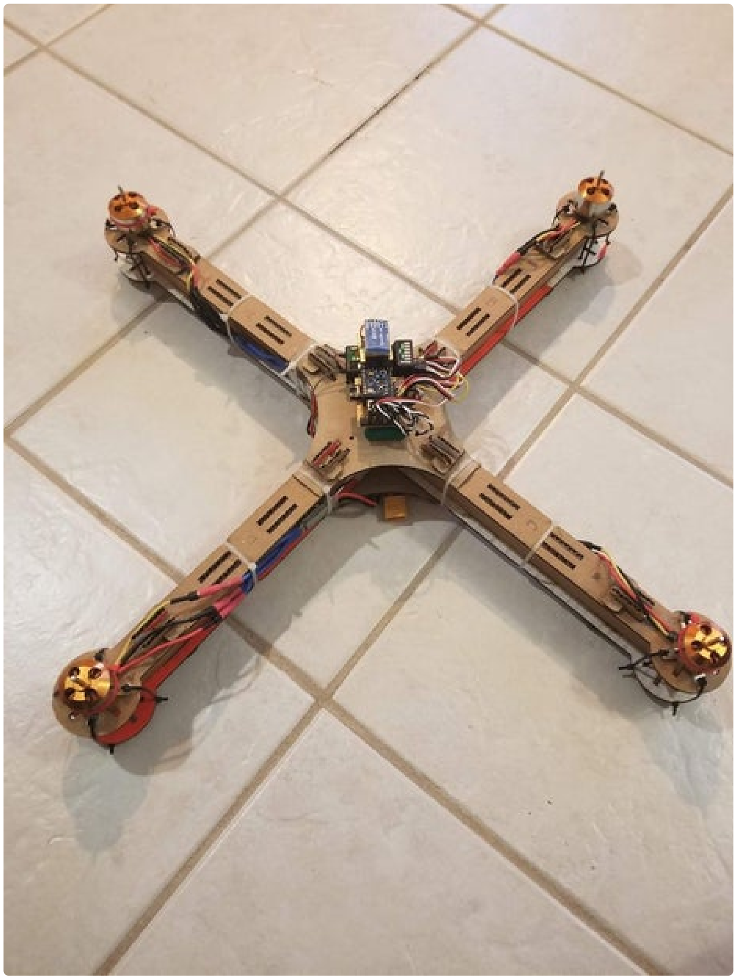






Step 16: Wiring the Whole Quadcopter

The title may sound intimidating, but to be honest, this is probably the easiest part of the whole process of quadcopter building. Still not convinced? Well, I'll show you....



Step 17: Solder (all) the Connectors

Female bullet connectors get soldered onto the three wires coming out from the ESC and male bullet connectors get soldered onto the two power wires from the ESC.

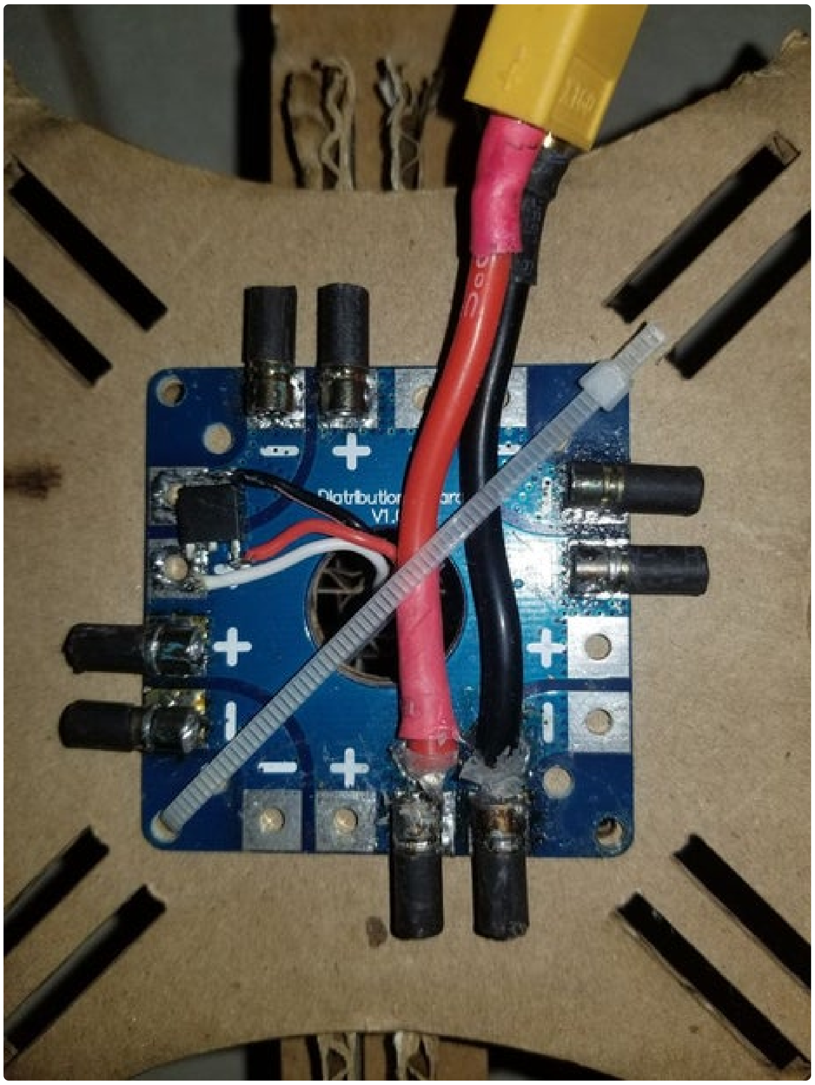
Look at this video if you need help:

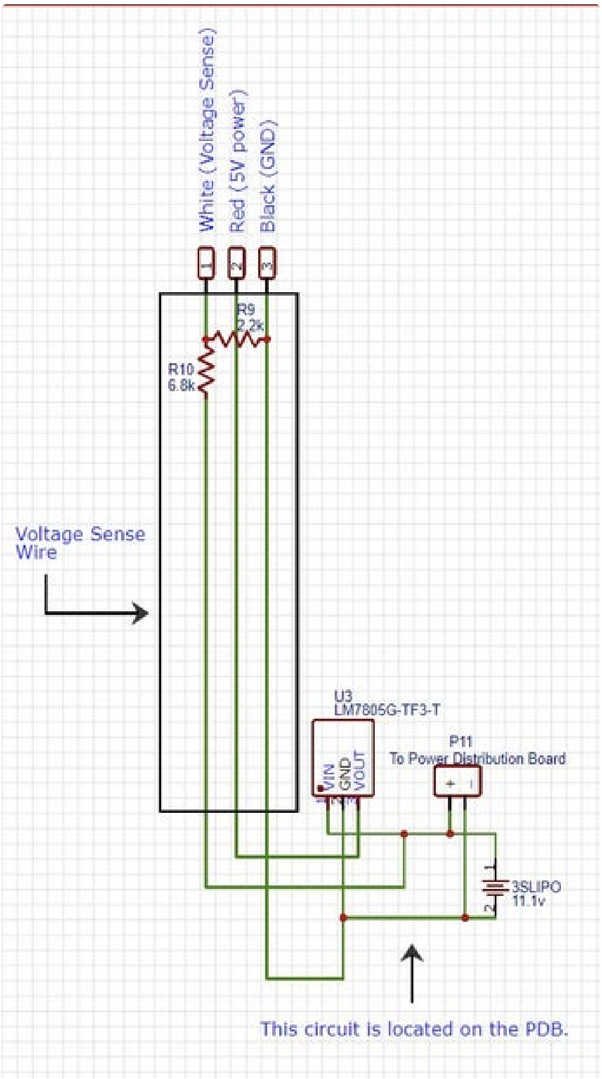
//www.youtube.com/embed/P3ddyiEPNwc

Female bullet connectors also get soldered onto the Power Distribution Board (PDB). First, lay them sideways and clamp them on the metal contacts. Then, melt the solder all around it to make sure it stays in place.

The male XT-60 connector gets soldered to the PDB as well. Solder wires to the XT-60 connector and solder the wires to the PDB.

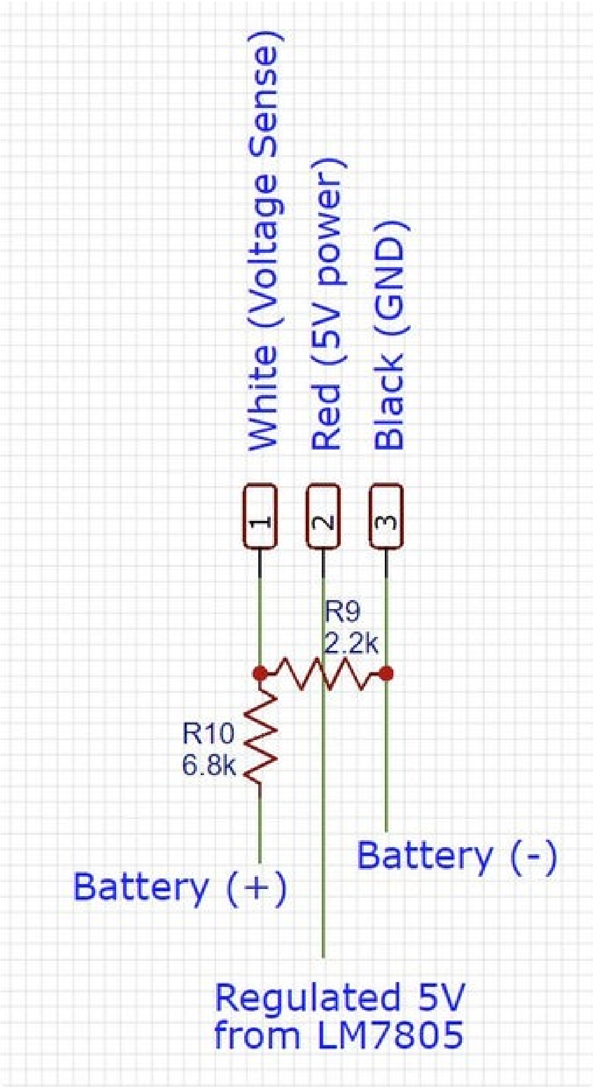
Create the voltage regulator circuit on the PDB.

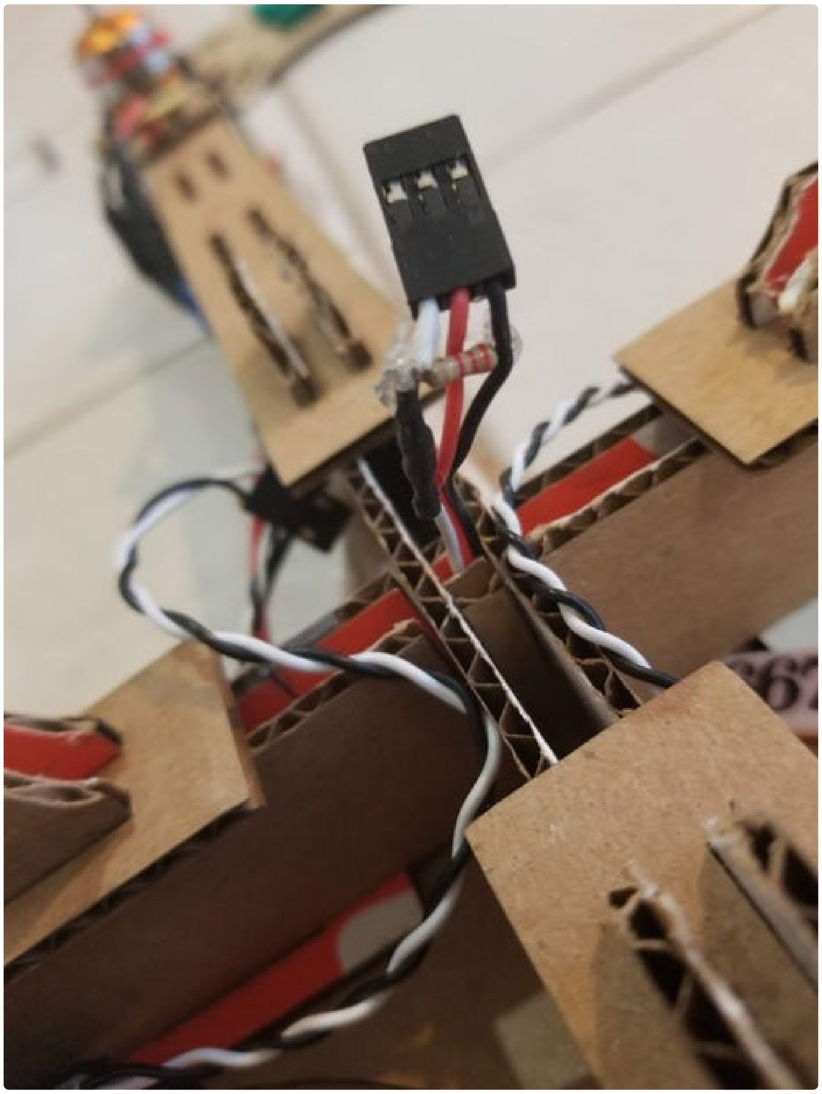




Step 18: Make the Voltage Sense Wire

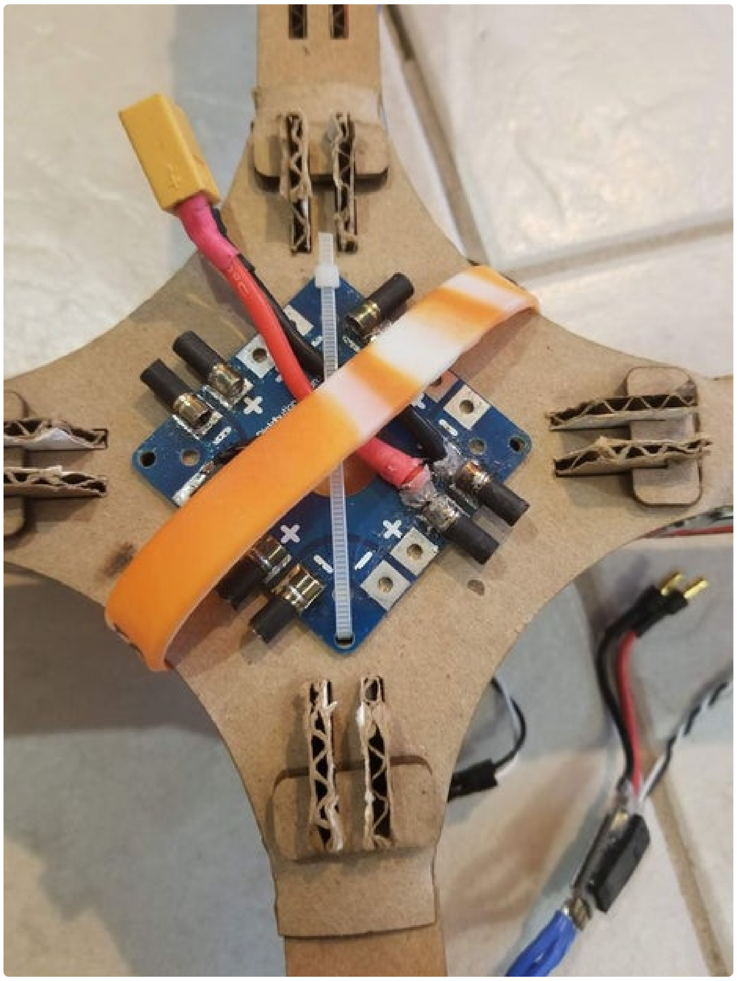
This wire will allow the Arduino to determine if the battery voltage is too low as well as provide 5V regulated power to the Arduino.





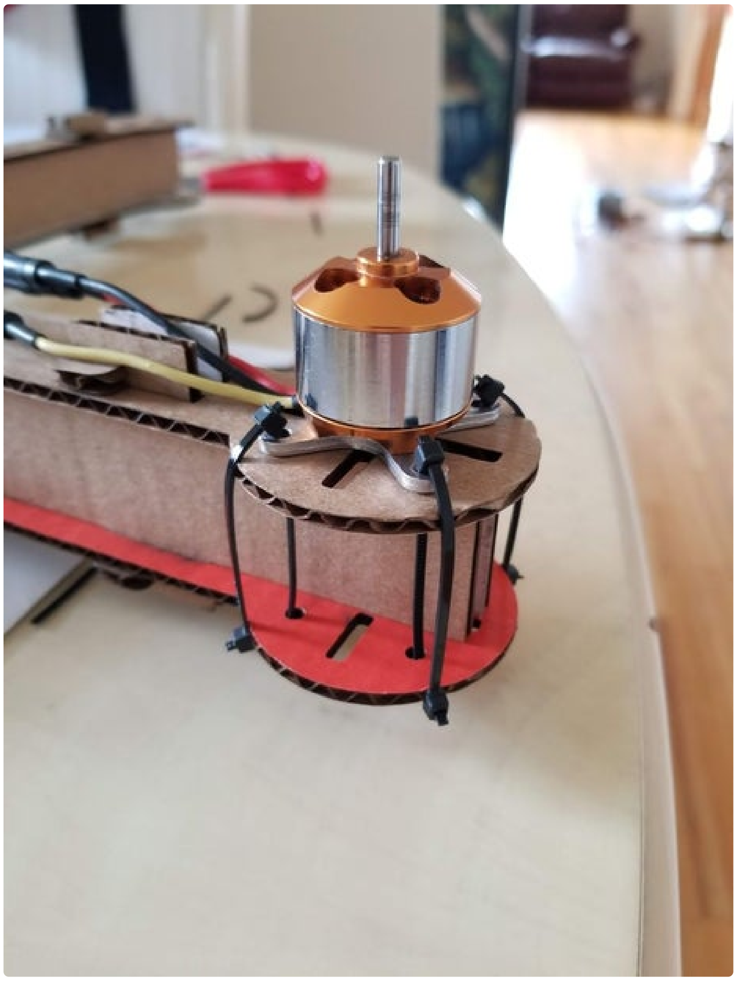
Step 19: Mount the PDB

Use zipties to mount the PDB onto the bottom plate of the quadcopter and feed the sense wire through the hole and up to the top of the quadcopter.



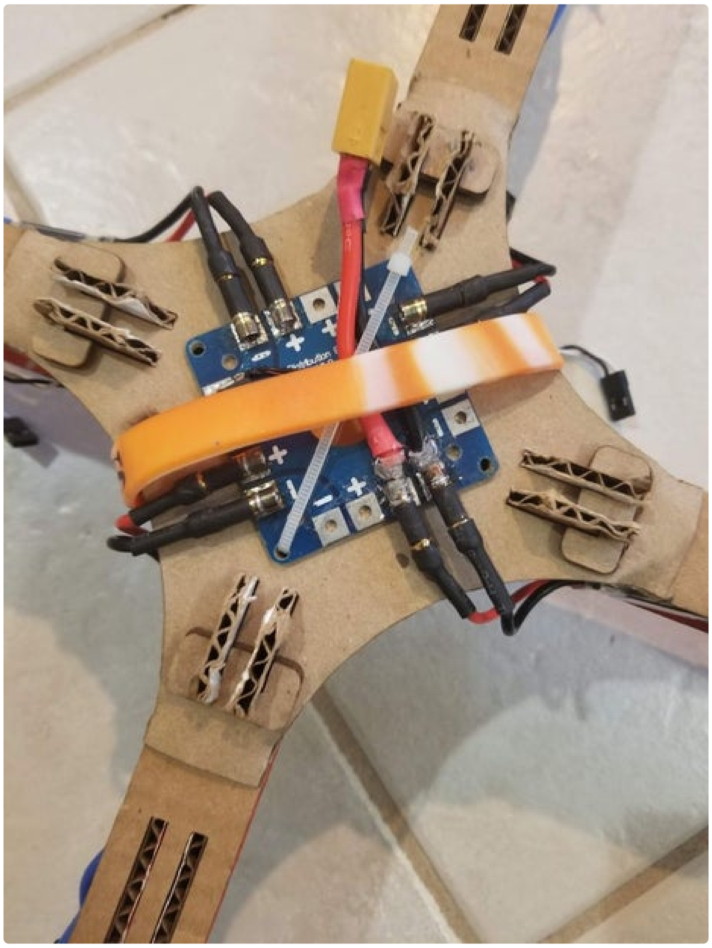
Step 20: Mount the Motors

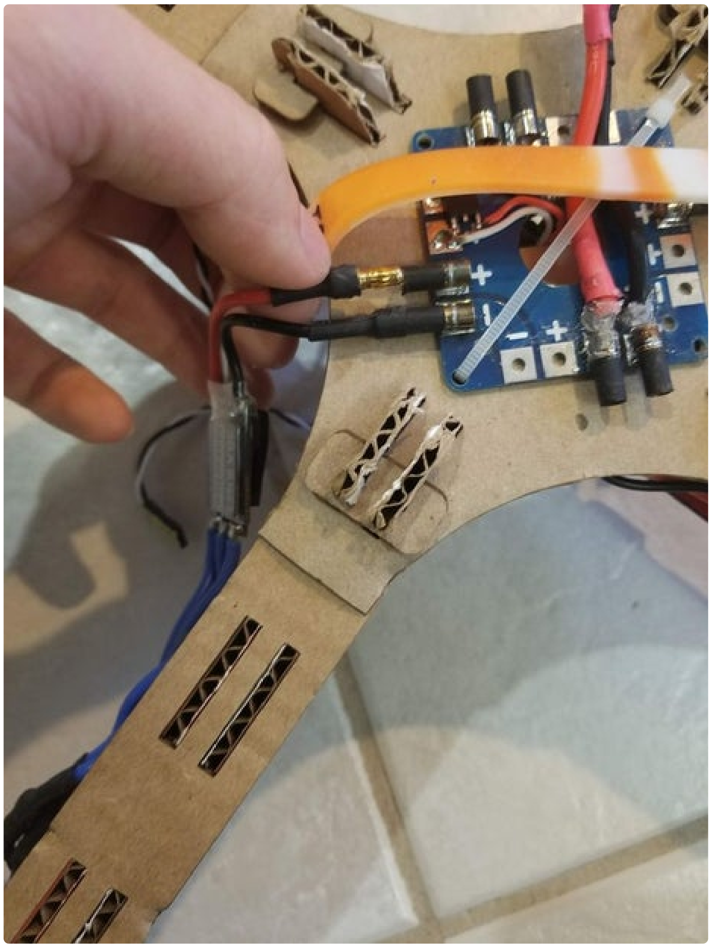
Since it is impossible to screw through cardboard and expect there to be a strong mount, I decided to use zipties again and they work very well.

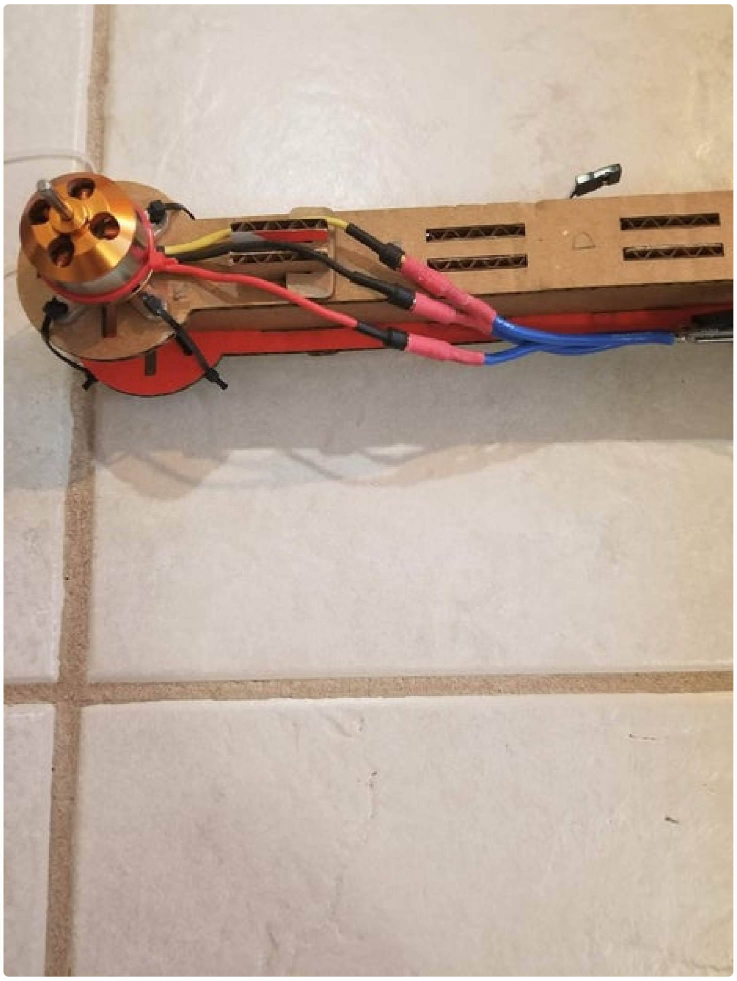


Step 21: Mount and Wire the ESCs

Use double-sided sticky tape to mount the ESCs and plug the power wires of the ESC to the PDB. Then, plug the three wires coming out of the other side of the ESC into the motor (it does not matter which way....we will fix that later if necessary).

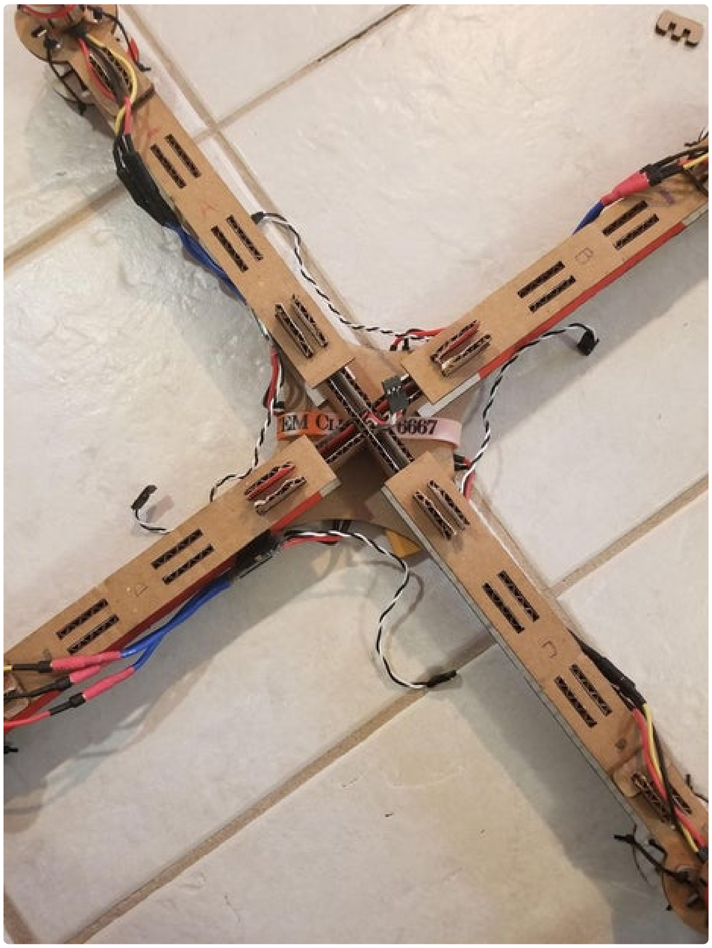


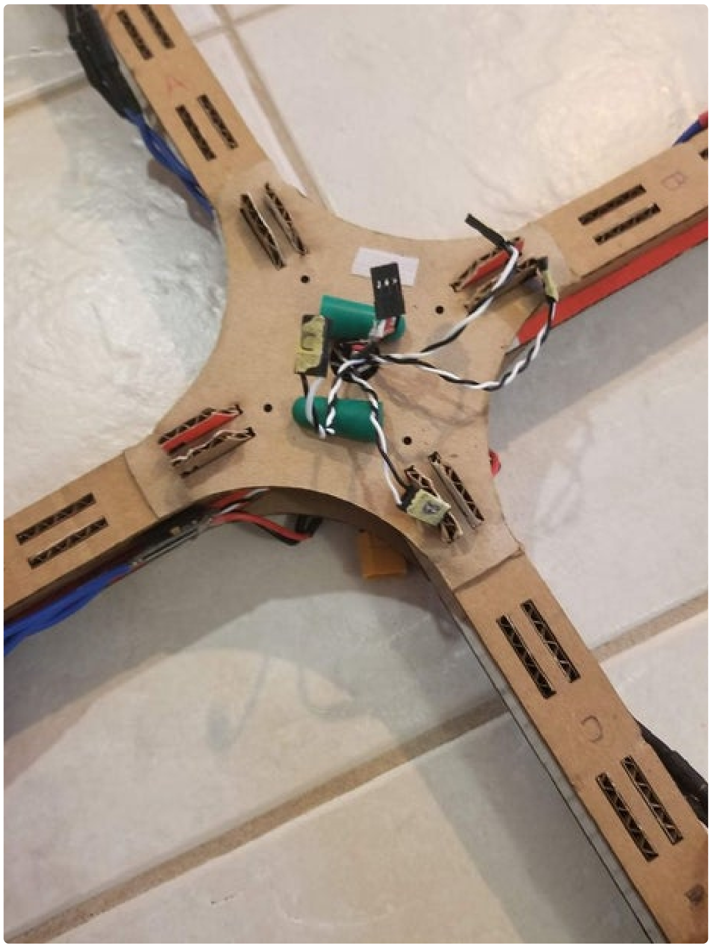


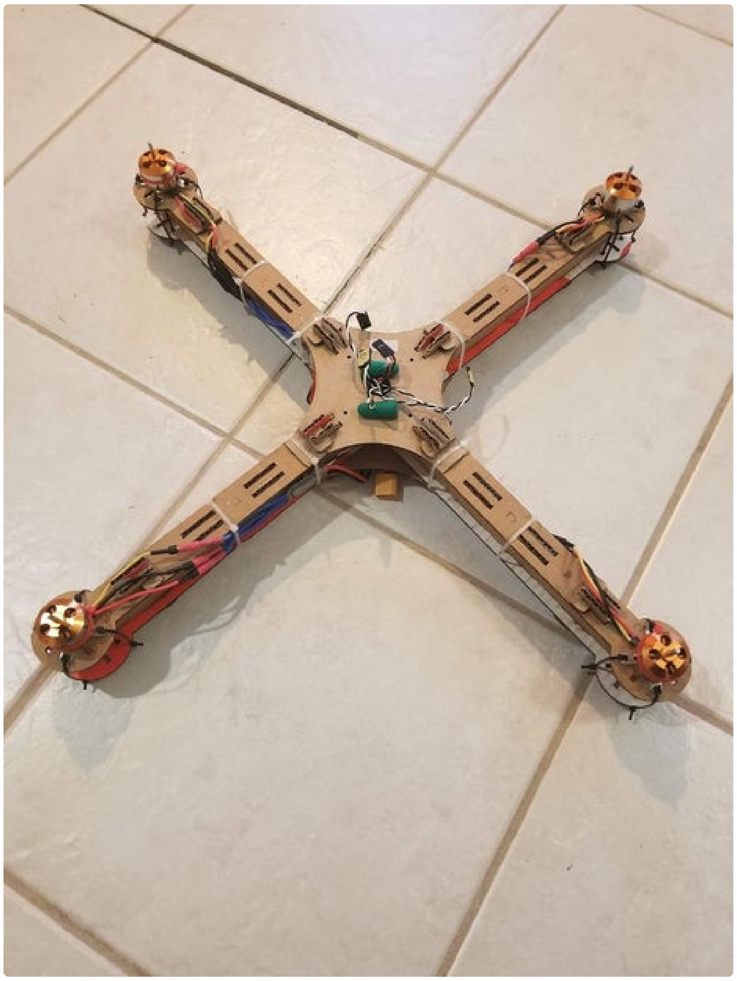


Step 22: Wire Management

Feed the signal wires of the ESC to the top of the quadcopter and insert the top plate. Glue 30dB earplugs to the top plate. These earplugs will act as vibration dampeners for the flight controller.



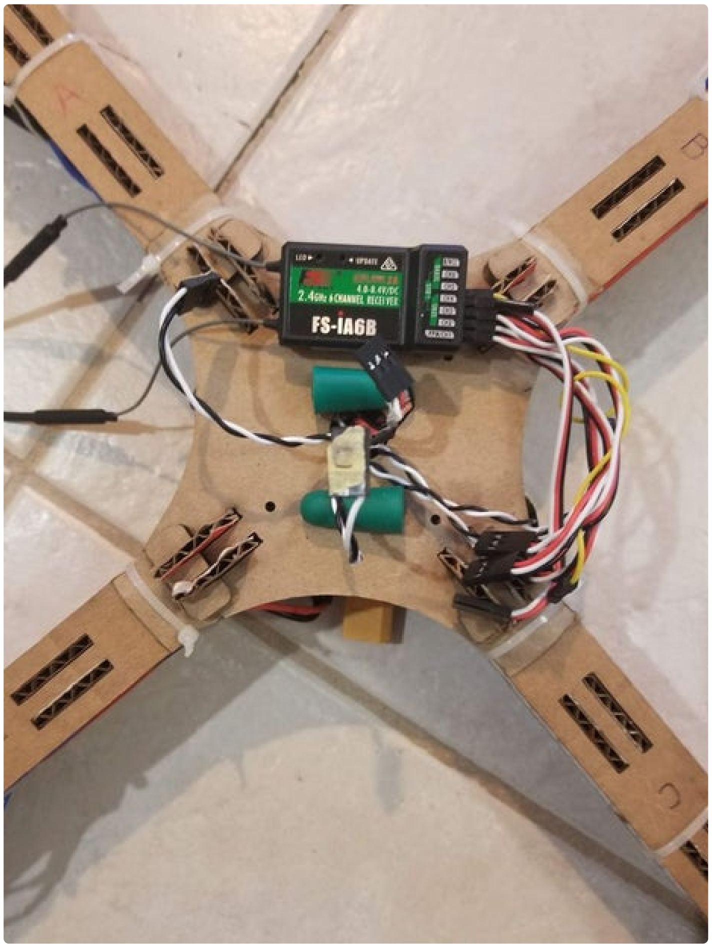


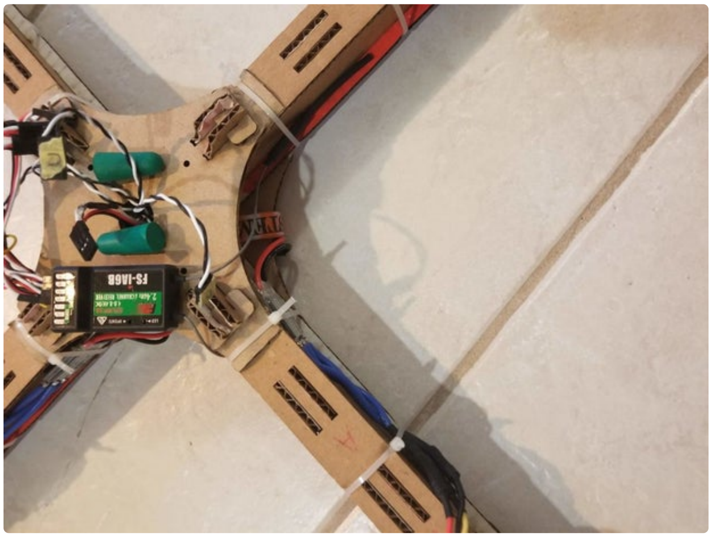


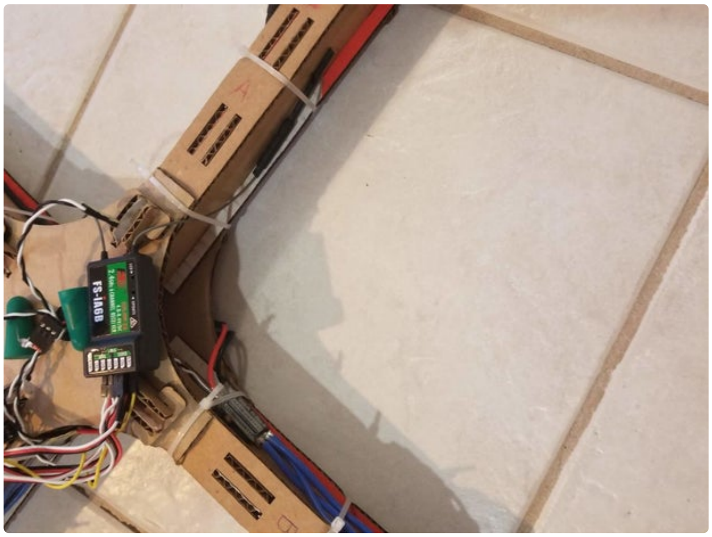
Step 23: Mount the Receiver

Put velcro on the receiver as well as the top plate of the quadcopter and mount it. Feed the antenna wires under the zipties on the arms, 90 degrees with respect to each other.

FS-IA6B


2.4GHz 6CHANNEL RECEIVER
4.0-8.4V/DC
FS-IA6B
LED
UPDATE
A
B
C


FS-IA6B
A


FS-IA6B
A
B

Step 24: Wire the Flight Controller In

Plug in the voltage sense wire to the flight controller and make the following connections:

Receiver --> Arduino

CH1 --> 12

CH2 --> 11

CH3 --> 10

CH4 --> 9

Motor --> Arduino

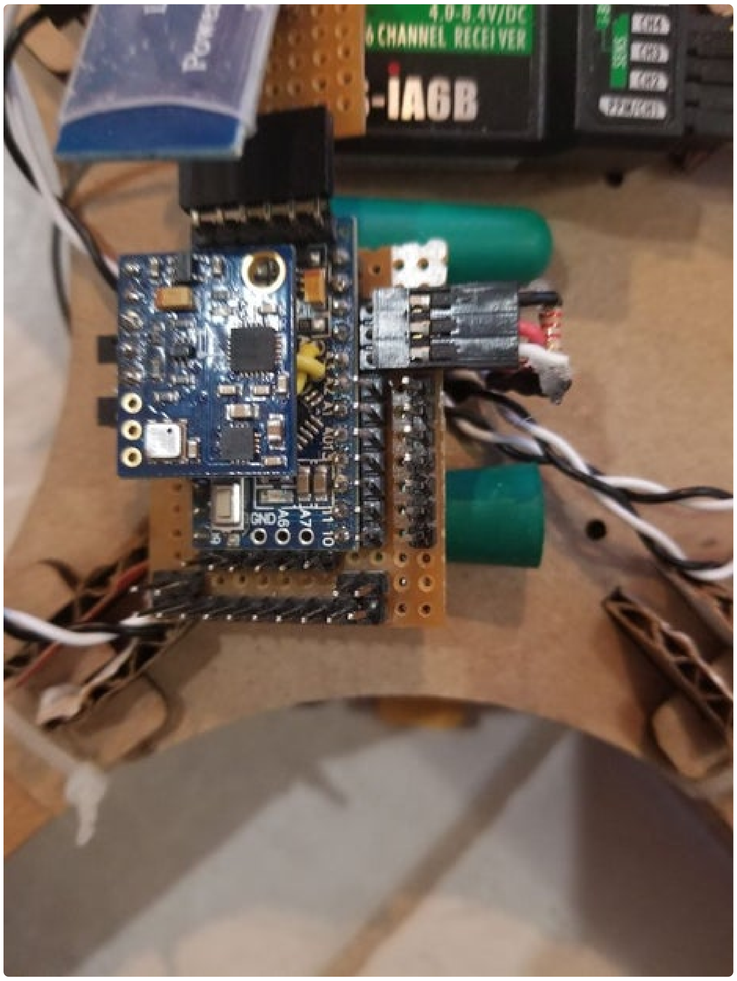
Top left --> 4

Top right --> 5

Bottom left --> 7

Bottom right --> 6

Then, secure the flight controller on top of the earplugs.


4.0-8.4V/DC
6 CHANNEL RECEIVER
-iA6B
CH4
CH3
CH2
PPM/CH1
GND
A6
A7
11
10

Step 25: Programming Board

To make our lives easier when it comes to the tuning, let's build ourselves a programming board we can attach to the Arduino that will allow us to program wirelessly.

You will need:

- 1x HC-05 Bluetooth Module

- 2x 2n2222 NPN bipolar junction transistor (or equivalent)

- Resistors

- 1.8k (x1)

- 3.3k (x1)

- 10k (x2)

- 1x 0.01 microfarad ceramic capacitor

- Some wires

- 6x female header pins

- Arduino Uno for HC-05 setup

- Perfboard

- Laptop/Computer with bluetooth

Set up the HC-05:

Connect the HC-05 to the Arduino Uno according to the schematic, hold down the button on the HC-05, power the Arduino with your laptop, and open the Arduino IDE.

Upload the test code and open the Serial Monitor.

Make sure that the Line Ending box is set to NL & CR (instead of No line ending) and at 9600 baud.

Type "AT" (without the quotes) into the Serial Monitor and hit enter, you should get a response of "OK".

If so, then set the baud rate of the HC-05 to the programming baud rate of the Arduino Pro Mini with the command "AT+UART=57600,0,0" (without quotes)

You should get a response of:

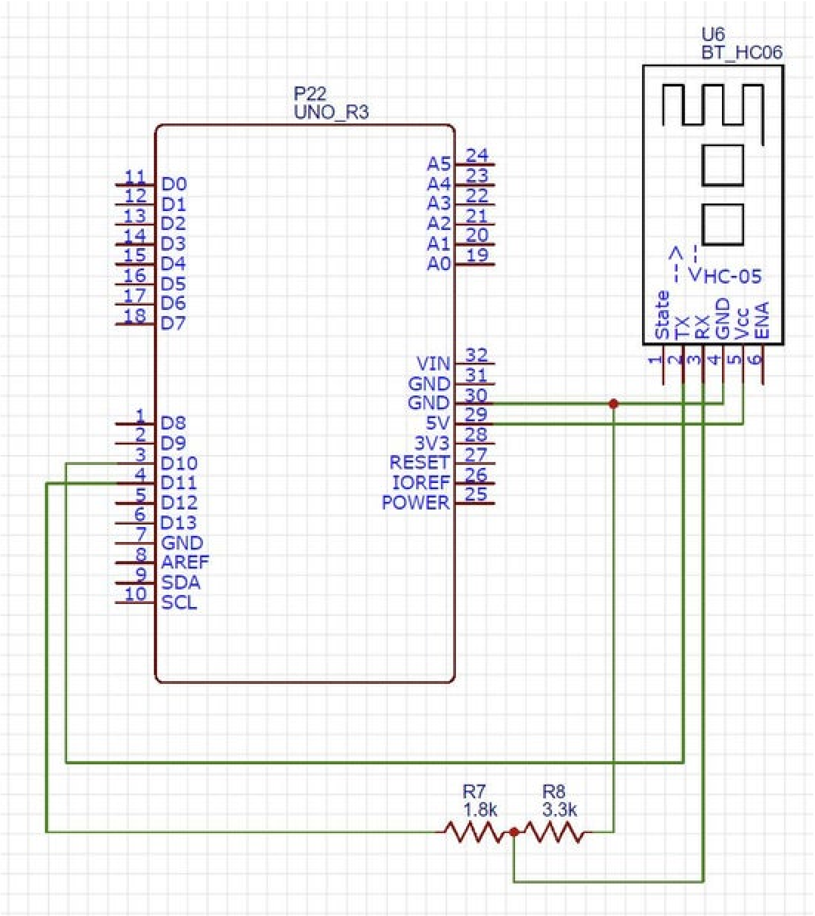
```
+UART: 57600,0,0

OK
```

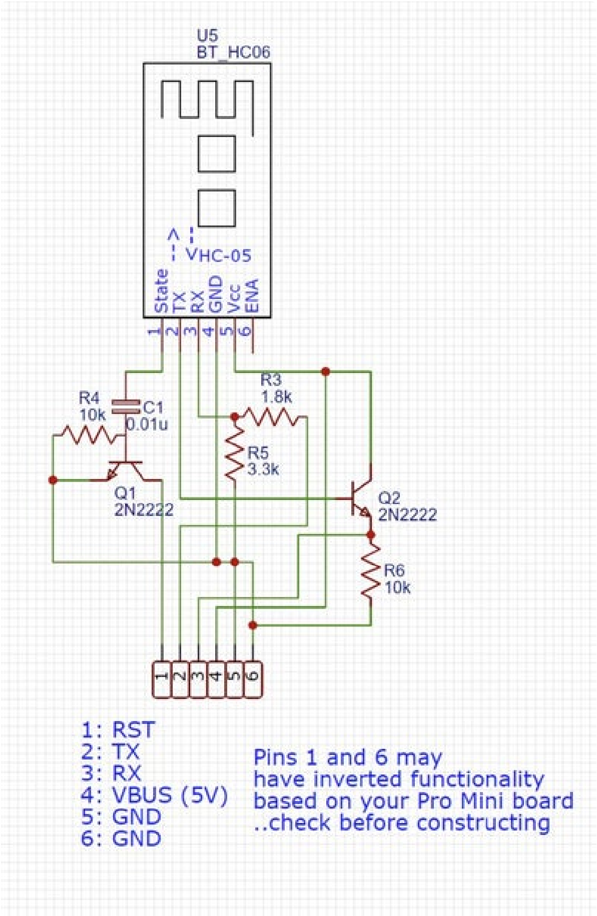
If you want to change the password, use this command "AT+PSWD=xxxx" (replace the x's with your numerical password). This is optional.

To connect with your laptop for programming, unplug the HC-05 from your laptop and power it from an external source. Then, turn on your laptop's bluetooth and find the device called "HC-05". When you try to pair with it, it will prompt you for a password, in which you will type the password that you set previously or if you chose not to change the password, the default is "1234". If that does not work, try "0000".

Construction of the Programming Board:

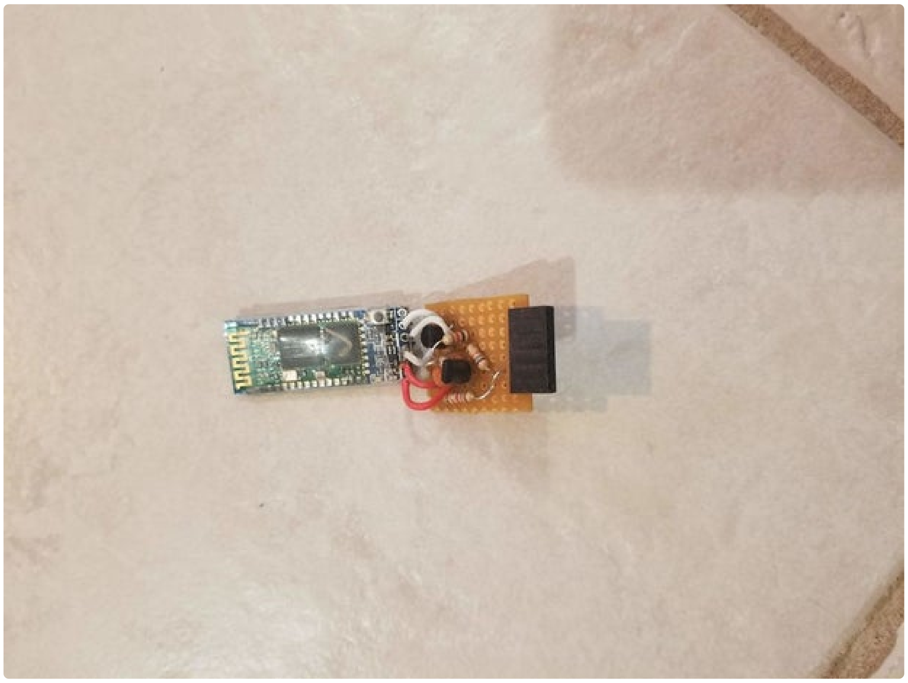
Refer to the schematic for construction information.





1: RST
2: TX
3: RX
4: VBUS (5V)
5: GND
6: GND

Pins 1 and 6 may have inverted functionality based on your Pro Mini board ..check before constructing



https://www.instructabl…

Download

Step 26: Theory (PWM Communication)

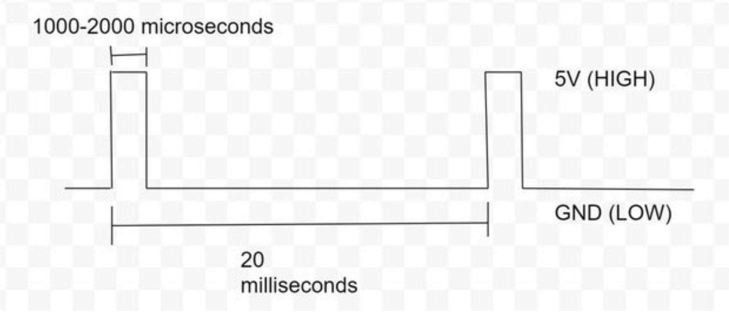
In the following sections I explain how the flight controller "thinks" and how things communicate with each other inside a quadcopter.

You will notice that many things like the receiver and motors connect to the flight controller with three-wire cables, which are called *servo connectors*. The red and the black/brown wires are your 5V and GND, respectively, and your white/orange wire is the signal wire, which sends pulse width modulation (PWM) signals like the image shown above.

By varying the amount of time the signal is "high" or connected to 5V, control information can be quickly sent and received. The signal sent out from the receiver at a 50 hertz (Hz) refresh rate and the "high" pulse varies between 1000 microseconds to 2000 microseconds (1-2 milliseconds). There are 1 million microseconds in a second...so that pulse is *very* fast.

Based on the image above, you can see that the whole signal takes about 20 milliseconds. The signal is only high for 1000-2000 microseconds, but for the remaining time, the signal is low. There are 1000 milliseconds in a second, and if you divide 1000/20, you get 50 Hz, indicating that there can be 50 twenty-millisecond signals sent within a second.

Now, the 50 Hz refresh rate may sound very fast to you, but for quadcopters, a 50Hz refresh rate is too slow to keep the quadcopter balanced. Why? Well, only the duration of the "high" pulse carries the motor speed information, and the low pulse is only there to help the flight controller differentiate between the first signal and the next signal and does not carry any important information. So in other words, making the "low" pulse way too long is a waste of time, and this is certainly the case. That is why the PWM interface for the motors is clocked much faster, in this case, 250 Hz, or 250 times a second! If we do the calculations: 1000/250, we can see that each signal only lasts for 4 milliseconds! In off-the-shelf flight controllers, refresh rates can be as high as a couple of kilohertz, but the Arduino is not fast enough to achieve those speeds.

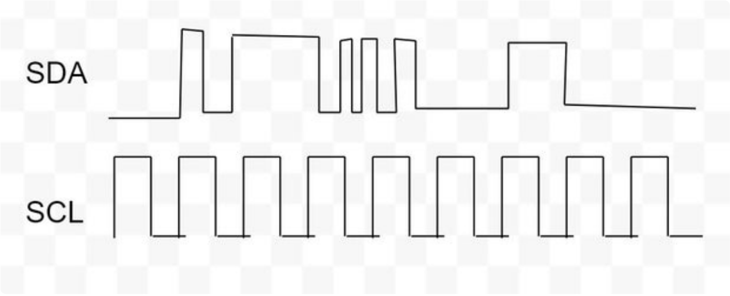


Step 27: More Theory (I2C Communication)

Flying a drone is 98% computer and 2% human. Why? Because humans are slow. Like *reeeeaaaaaalllllllyyyyy* slow. In fact it is nearly impossible to fly a quadcopter without a flight controller to automatically adjust and stabilize the quadcopter hundreds of times per second. In the previous step, I mentioned how receiver information has a refresh rate of 50 Hz, and that is good enough because human reaction times are not going to be that fast anyway (average response time to a touch stimulus, the fastest, is around 150 milliseconds).

However, how does the flight controller know that the quadcopter is tilting? Well, in my drone, I used a GY-87 10DOF IMU, or inertial measurement unit. This module is capable of measuring acceleration in the x, y, and z axes, can measure rotation rate around the x, y, and z, axes, the angle from magnetic north on the x, y, and z axes, and can measure air pressure as well. Using all these pieces of information, the quadcopter can accurately determine the tilt. I will go into more detail in the next step.

The IMU is getting all this good information, but how does it tell all this to the Arduino....and fast enough so it is not too late to save a falling quadcopter? This most certainly cannot use PWM, as it would be way too slow to send all these 10 pieces of information all at once. Instead, it uses a communication protocol called I2C (pronounced "eye squared see", but also known as TWI, or two wire interface), which as the name suggests, uses two wires called SDA (serial data) and SCL (serial clock). This communication method can allow maximum speeds of up to *400 kHz* or 400,000 times per second!



How does it work?

Well, I2C communication occurs between a *master* (the device who wants the information) and a *slave* (the device that gives the information), and the slave has its own address, or name, in hexadecimal form. It kinda works like a really small band, let's say a drummer and a rapper. **The drummer keeps the beat and the rapper delivers the message.** The SCL line keeps both devices synchronized, the SDA line sends a chain of pulses of different lengths to send different messages. This is called *serial* communication because the information packets are delivered sequentially, one after another. With this method, over 100 devices can be connected and interfaced independently with only two wires! Really neat stuff.

Step 28: Even More Theory (Sensors, Math, and Angle Calculation)

Each sensor has their own advantages and disadvantages, and this is why we need a ton of sensors.

Gyroscope (or gyro): Measures angular velocity, or the rate at which the quadcopter is spinning in degrees per second. So if you know the speed at which the quadcopter is spinning and the amount of time passed, you can calculate the angle. This is by far the most accurate and reliable sensor on any drone, as the values are not affected by vibration and linear motion (motion in a straight line). However, gyros do suffer from drifting (called low frequency noise), where small errors add up to make huge problems.

Accelerometer: Measures linear acceleration and knows which way is down (it uses the acceleration of gravity to determine that). By using trigonometry (look at pictures), we can figure out the angle which it is tilted at. However, this sensor's values are very noisy and is greatly affected by quadcopter motion, vibration from motors, and high frequency noise.

Accelerometers also suffer from gimbal lock:

For example, if the quadcopter is perfectly level, the z axis will point upwards and the x and y axes will be parallel to the earth. In this case, if the quadcopter were to pitch and roll, the x or y axes will no longer be parallel to the earth and the z axis will no longer be perpendicular to the ground, therefore, registering an acceleration on the axes. This enables the accelerometer to calculate pitch and roll. However, if the quadcopter were to yaw, the accelerometer will not sense any change because the Z axis will still be pointing straight upwards and the x and y axes will still be parallel to the earth. We lose one axis of rotation.

Magnetometer: Tells you where is magnetic north and how many degrees you are away from it. However, because this sensor measures magnetic fields, if you have big coils with a lot of electricity running though it or big metal things with magnets spinning at thousands of rpms (cough cough your motors), then the measurement will be affected. These sensors need to be manually calibrated from time to time (check the next step for information).

Barometer: Measures air pressure. Using this information we can approximate altitude. This sensor can be affected by light (yes it is that sensitive).

A quadcopter needs to measure pitch, roll, and yaw angles, as well as position and altitude (both optional).

Pitch and Roll: The accelerometer and gyroscope data are both used to calculate the angle. The gyro does most of the angle measurement but the accelerometer is there to check and correct the gyro values if it has drifted too far.

Yaw: The gyro and magnetometer are used to calculate the angle. In this case, the magnetometer replaces the accelerometer and corrects the gyro drift.

Altitude: The barometer and GPS are used to calculate the altitude. Using only the barometer is fine for most cases, but if you really need precise altitude information (altitude hold for example), GPS can be used to correct barometer errors

Position: GPS....doesn't need anything for compensation as it is very accurate, but GPS has a really slow refresh rate (10 Hz).

Accelerometer:

Roll Angle: $\theta = atan2(\frac{acc_x}{\sqrt{acc_y{}^2+acc_z{}^2}}) \times \frac{180}{\pi}$

Pitch Angle: $\theta = atan2(\frac{acc_y}{\sqrt{acc_x{}^2+acc_z{}^2}}) \times \frac{180}{\pi}$

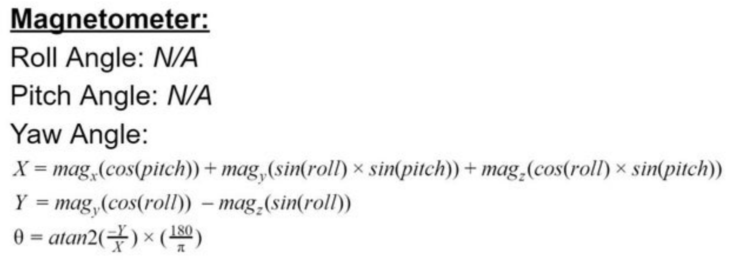
Yaw Angle: *N/A*

Gyroscope:

Roll Angle: $\theta = \theta_{y_{original}} + (rotation\ rate\ of\ y)(time\ elapsed)$

Pitch Angle: $\theta = \theta_{x_{original}} + (rotation\ rate\ of\ x)(time\ elapsed)$

Yaw Angle: $\theta = \theta_{z_{original}} + (rotation\ rate\ of\ z)(time\ elapsed)$



Magnetometer:

Roll Angle: *N/A*

Pitch Angle: *N/A*

Yaw Angle:

$$X = mag_x(cos(pitch)) + mag_y(sin(roll) \times sin(pitch)) + mag_z(cos(roll) \times sin(pitch))$$

$$Y = mag_y(cos(roll)) - mag_z(sin(roll))$$

$$\theta = atan2(\frac{-Y}{X}) \times (\frac{180}{\pi})$$

Step 29: Transmitter and Receiver Info

If you just bought a transmitter and receiver, you have to let the transmitter know that it will be talking to the specific receiver that came with your box and vice-versa. This process is called binding and it takes literally 10 seconds.

1. Plug the bind cable into the "BAT" (battery) port on your receiver.

The bind cable is the little black loop of wire with a servo connector at the end

2. Power on the receiver with the Arduino flight controller

3. While holding the "BIND" key on your transmitter, turn it on

4. Lights should blink and when everything stops blinking, you can unplug the bind cable and you are done!

Step 30: Calibration

There are two things we need to calibrate before we can fly, the motors and the magnetometer.

Calibrate the Motors:

We need to do this in order to tell the ESCs what the throttle range is. This is very important, because if you do not take the time to calibrate your motors, your motors may start at different times, not start at all, or may not utilize the full range of the throttle correctly.

Moreover, the standard timing for an ESC is 50 Hz, which I mentioned previously, is way too slow for a drone, so a calibration will allow the ESC timing to be scaled up to 250 Hz.

This is fairly simple. Upload the code attached to the Arduino (called "motor_calibrate") and follow this procedure:

1. Unplug battery

2. Take off your propellers

3. Take off your propellers

4. Did you do steps 2 and 3?

5. Set your throttle stick on your RC transmitter to full throttle.

6. Check and make sure your propellers are off

7. Plug in the battery

8. Your ESCs will beep a startup tone and then it will beep once or twice

9. After you hear the final beeps, lower your throttle to minimum throttle and wait for a confirmation beep.

10. Then move your throttle stick up and down, all motors should spin up at the same time

11. Check if the motors are spinning in the right directions: top left- counterclockwise, top right- clockwise, bottom left- clockwise, bottom right- counterclockwise.

12. If any motor is not spinning in the right direction, simply swap any two of the wires connected to the ESC and the motor will change direction.

13. Unplug battery

14. When you are ready for the main code and PID tuning, put the propellers on. Depending on which way your motors are spinning, remember that the elevated edge of the propeller blade cuts into the air first.

Calibrate the magentometer:

If the magnetometer is not calibrated, the yaw angle values will be off.

1. Upload the code called "calibrate_mag" to the Arduino.

2. Open the Serial Monitor (use No line ending and 9600 baud) and spin your quadcopter around the z-axis at least 3 times slowly.

3. Record the final two numbers. These are your offset values.

4. When you get around to uploading the final flight controller code, insert the offset values into the code (there will be comments in the code which indicates where you should put your offset values).

https://www.instructabl... Download

https://www.instructabl... Download

Step 31: Code

I attached the code for the quadcopter, note that this code is nowhere near perfect and does not take advantage of all the sensors and input devices I've mentioned. In fact it, is the bare minimum required for flight. However, if you are a seasoned programmer, go ahead and adjust it to make other cool stuff happen.

To use the bluetooth programming board to program the Arduino, turn on your computer's bluetooth and open the Arduino IDE. Go to Tools --> Ports and note the available ports. Power on the Arduino, go back into Ports and look at the new port that appeared. Click on this port and upload the code and it should upload wirelessly.

Note: The refresh rate of the code is 250 Hz, which means it takes 4000 microseconds to complete a loop. If you add more code, make sure the looptime is not greater than 4000 microseconds, or angle calculation will be off. Remember the gyro outputs the rate of rotation, so it must be multiplied by the elapsed time in order to calculate angle. Calculating elapsed time every loop makes the program too sluggish, so setting the looptime to a constant time (4 milliseconds) will enable the Arduino to calculate the gyro angle much faster.

https://www.instructabl...

Download

Step 32: PID Tuning

Before you can actually fly, you need to tune your PID controller. Essentially, you initially make the P-gain equal to 1.00, upload the code to the Arduino, fly, land, change more values, and repeat until you get something flying nice.

The PID controller is responsible for how responsive the quadcopter is going to be. There are a ton of different methods to PID tuning, but this method works for me:

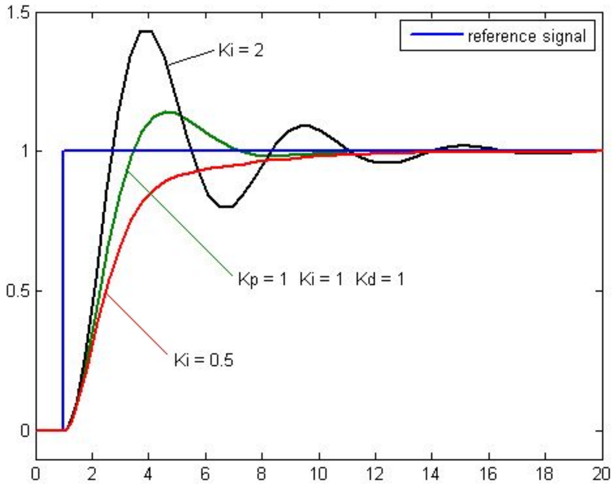
Increase the P-gain (whole or half number intervals): Increases the responsiveness (or the "kick") of the quadcopter (aka determines how much the quadcopter speeds up the motors when there is little tilt). Keep increasing this value until the quadcopter starts to overcompensate and starts wobbling.

Increasethe D-gain (intervals of about 0.5): Dampens the P-gain. For example, a human hitting the drone or sudden gust of wind will cause the drone to try to rocket back to stability, but the D-gain provides a "force" to resist the drone from overshooting. Increase this until the quadcopter starts vibrating, then decrease it by around .5.

Increase the I-gain (intervals of 0.001): Corrects small quadcopter drift and helps the quadcopter actually return to zero degrees.

PID tuning is long, brutal, and will require some (a lot of) patience. Play with the values in this order until you get satisfactory performance.

For more information I recommend watching Joop Brokking's channel on youtube as he explains a lot more through his videos in a rather entertaining way.



Step 33: Flight....fail

<u>When you try your best but you don't succeed.....</u>

This was the first flight with my original code that I did not post because it's bad.

Now as you can see, my drone did not take off successfully in this footage, and there's a couple of reasons why. 1. The batteries were too discharged

- I spent the whole day tuning the PID on the quadcopter and never bothered to check the voltage of the battery (preferably above 12 volts) before taking it out to fly....so it is essential to have fresh batteries before flight. With a discharged battery, your motors will not produce the expected amount of thrust and your drone will not fly.

2. PID values were out of wack

- Even with a whole day of PID tuning, you can see that I didn't do a good job of it, as see by the large wobbling around the y-axis. Decreasing the P gain of the roll can solve the problem (note, if the drone was oscillating very rapidly, then it is the fault of the D gain).

3. Battery Connections were not good

- If you caught the quick beeps from the ESCs at the end of the video, you can tell that the quadcopter had powered off and powered on again. This is the result of bad connections that rocked loose during the wobbling.

https://www.instructabl…

Download

Step 34: Fix Those Problems

Problem 1 is easy to solve....charge the batteries!!!

Problem 2 will require more hours of tuning.

Problem 3 is also fairly easy to solve. Apparently trying to clamp the battery wires underneath the soda can and hot-gluing them in place is not enough....so here's my solution:

- Solder the battery wires to washers

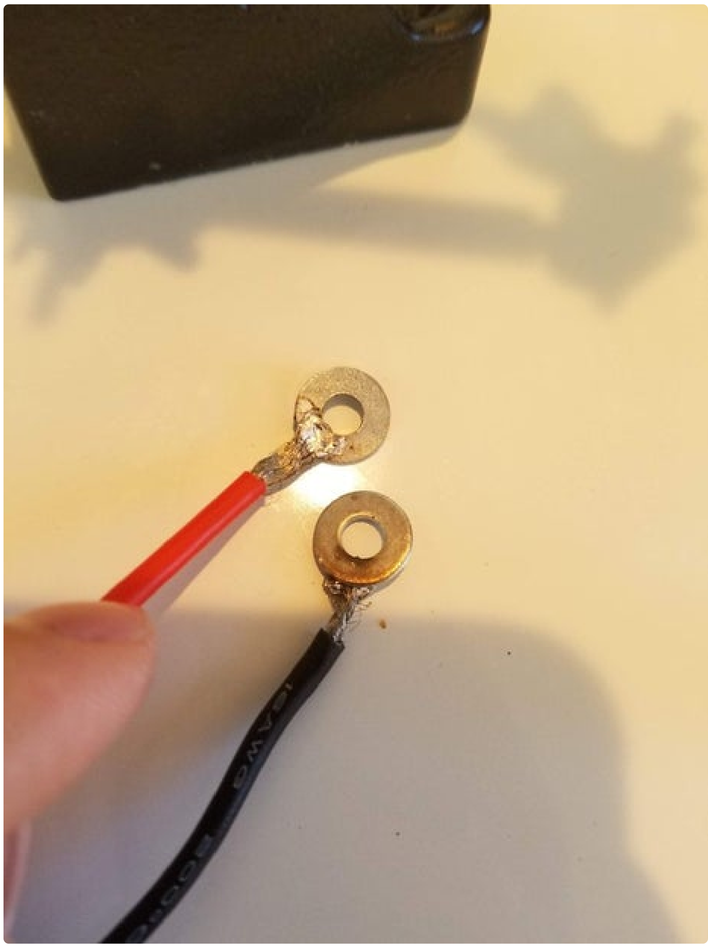
- Pre-sand the washers with 100 grit sandpaper so the solder will stick to the washers better

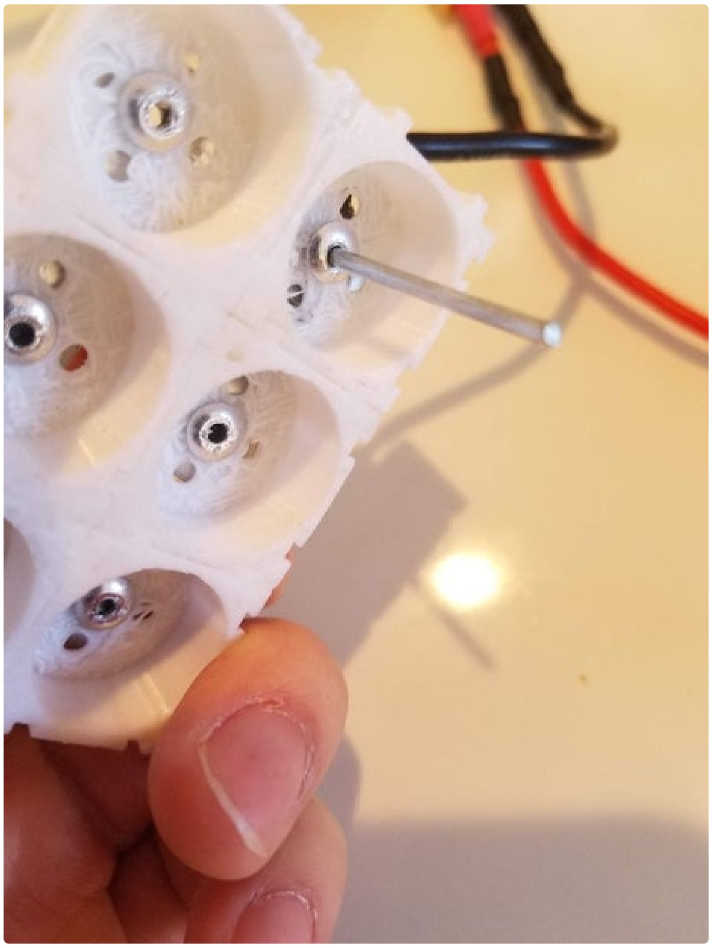
- Solder the wires.

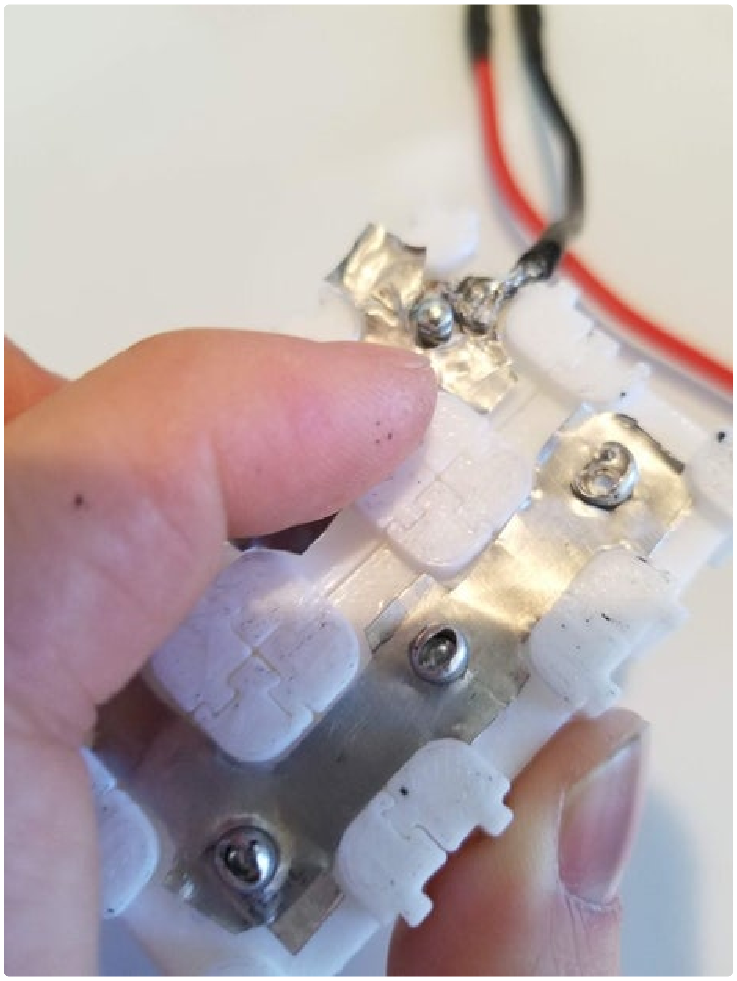
- Slide the rivet through the washer

- Put the soda can strip on top, and rivet everything together

Clearly, this was a much more robust solution.







Step 35: Final Remarks

So congratulations, you've finally reached the end of this really long instructable. I hope that you've learned a lot of about the art of drone building and electronics as much as I did and I hope that I did not bore you too much.

There are a ton of upgrades I can make to the quadcopter, but if I had the time (and money) to do so, I would:

- Upgrade the battery to higher quality pack, because my DIY battery pack is heavier and requires a lot more maintenance, but the DIY pack has taught me a lot more.

- Integrate a Raspberry Pi Zero W and a camera, so I can put in computer vision, data logging, FPV (first person view), aerial photography, and much more.

- Upgrade the motors and propellers

- Add GPS and position hold features (perhaps even autonomous flight)

- Add a camera gimbal

Some use cases for my quadcopter:

- Seek out my school bus when it comes so I can make it to my stop on time (really useful to have on cold days, where I can limit the amount of time I spend outside)

- Take cool pictures

- Drop water balloons on people during hot summer days :D

- The possibilities are limitless!

Unfortunately the failed flight damaged my last stock of propellers, and I will need to order more. If I manage to do any of the above and find some time within my busy high school life, I will post an updated instructable with a successful flight (hopefully) as well as highlight more things that I have done with this project.

So in the end, thank you all for taking the time to read my instructable, and now I'd better catch up on my homework....

Thanks for pointing that out; I am in no way an expert at this, but the interesting thing is that I have seen many explanations about how wings fly and how lift works, including the pressure difference explanation. But there are a couple of things that bother me about that explanation. If it was purely the pressure difference, then flat wings like paper airplanes and cheap foamies will never fly (https://www.youtube.com/watch?v=Vmhb7rRIBuI). Sticking an arm out the window will not create lift until you change the angle of attack, and direct the air downwards. Airplane wings are in a shape of an airfoil to improve efficiency, and if the angle of attack on airplane wings were to be changed, it will produce more lift, which is exactly what control surfaces do. Additionally, there are also stunt planes that can fly upside down, because they have symmetrical wings which do not create pressure differences very well. But I completely agree with the way that rockets work, as in space there is no air to push against, it is purely action-reaction of exploding gases.

Amazing explaining well done!!

This is so comprehensive! Thank you for sharing your work :D