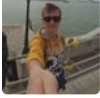




Arduino Drone | Quadcopter (3D Printed)



by Nikus

Some time ago (over 8 months) I was thinking about what I can build. I wanted to make an interesting robot/device that will be challenge for me and will encourage me to learn new things. I thought about the lot of robots but a lot of them were posted on the internet. And i thought about making a drone completely by myself, including flight controller, pilot, program and frame design. This is by far the longest build i have made, it takes a lot of time and effort to make it but finally after over 8 months it's ready and i am here to share it with you as completely open source project.

I just started my **facebook fanpage!** If you want to see what I am making, be up to date with my projects or need some help feel free to like it, thanks!

Here is link to video if you use an app

Info for begginers:

This project is not the simple one. You need some basic knwoledge of arduino programming, PCB's and electronics to make it. If you haven'y done anything like that before I advise you to start with something simpler like:

Arduino sensitive robot

[DIY bike tachometer \(speedometer\)](#)

[Very Simple Robot for Beginners](#)

But if you really want to make it, you should try, I am always here to help you.

My apel to you! :)

I am participate in contests right here on instructables. In one of them (microcontroller contest) there is macbook air as a grand prize. It could help me with calibrating PID for my drone because right now I haven't any laptop that I can go outside with so there is no way to calibrate it perfectly. I am also working on an app with my friend we want to publish it before holidays. We have a very very good idea for it and it can be very popular, I can't tell you what is it I can just tell you name of it - Socialize. We are finishing android version of this app. As you may know to make an iphone app you need t have apple computer. We haven't one and we haven't money for it :(So it will also help me with developing IOS version of our map that I will inform you about shortly. If you want to vote for my project you can do it by clicking vote! in upper right corner and then selecting contest in which you want vote for me. Thanks everyone! :D



<https://www.youtube.com/watch?v=PDB7qasURs4>



1. One of the first version of my drone

Step 1: LUDWIK DRONE THE WINNER!!!!

For some reasons and because of my friend lukmar I named my drone Ludwik. And Ludwik took part in International robotic tournament in Rybnik in Poland and it won first place in freestyle category! I even print for this contest like this label with Ludwik dron :) Above is the photo from this contest.



Step 2: Why My Own Program, Frame, Pilot and PCB?

You can ask why do you wrote your own program for it? Why do you made custom 3D printed frame (see next steps to see why I failed with it)? And additionally custom pilot and PCB's. Basically because I can :) I like making things on my own and learning how they works. Thanks to this type of thinking I learned a lot about quadcopters and how they works.

Code

The biggest problems I had with code for stabilizing it, it wasn't hard to write it but very hard to adjust and to get rid off some bugs. It's still not perfect but much closer to it and it works really good so far. When my code was ready I started to adjust PID regulators ([here you can read something more about PID](#)) and this was a terrible thing, I almost destroyed my room because of some really small bug in my code (for some reasons my code was decrementing int infinitely and when it reaches minimum value it goes to maximum value of int and my drone turned on motors 100% and hit wall and doors in my room). For a long time my PID regulator was adjusted very bad but every day I was closer to make it perfect. While this adjusting I broke my 3D printed frame 4 times :(

Something about 3D printed frame

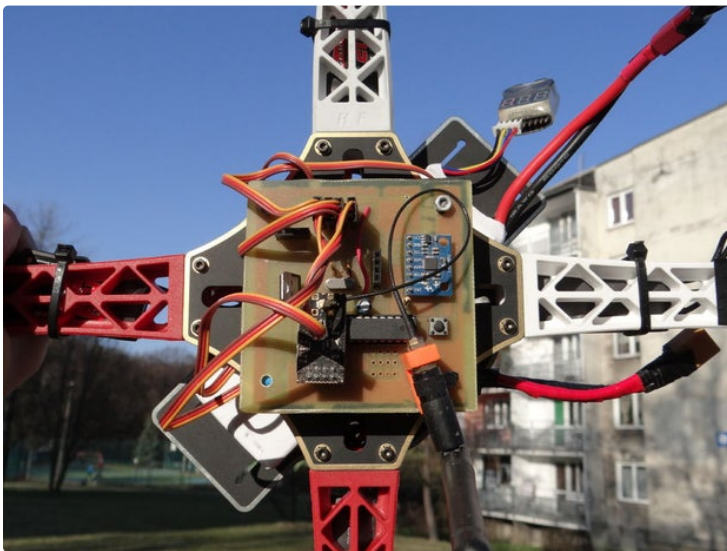
At the beggining I thought that it can be very usefull to make 3d printed frame. It was really possible that I will broke it at least few times (I was right :) but after 7 broken parts I decided to buy one, mainly because it is much stronger and rigid. I broke this frame during first flight with it :(

Pilot

Main reason is money. If you want to buy a pilot for drone you need to spend at least \$50, thats quite a lot. So I build my own for like \$20. And the good thing about it is that I can create as many channels as I want :)

PCB's

Because of my own code and my own pilot I have to make PCB's. I made 3 of them. The first one is power distributor for motors, second one is pilot and third one is of course flight controller. All of them use through hole components to make it easier to solder. In my final version I am not using power distribution board because my frame already have that.



Step 3: Quick Story

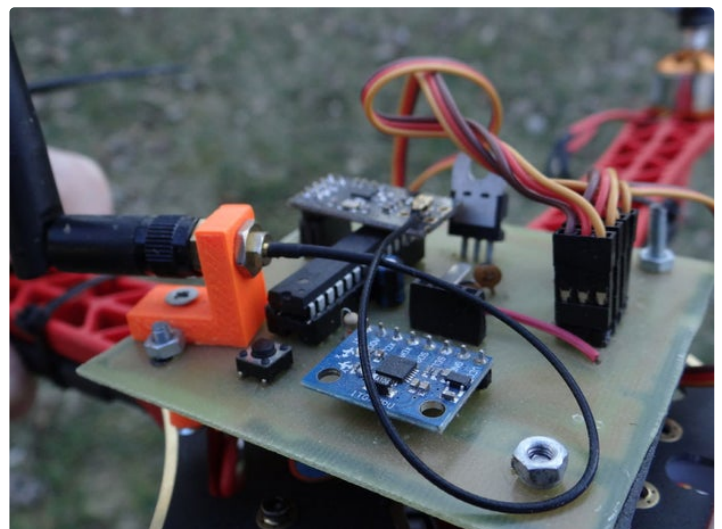
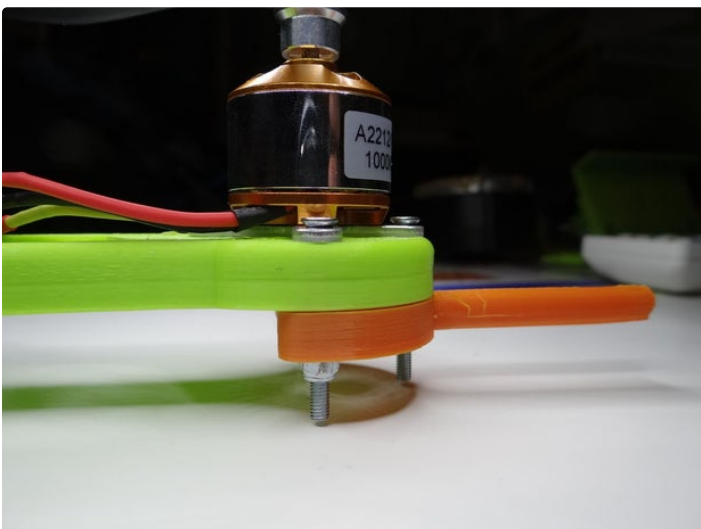
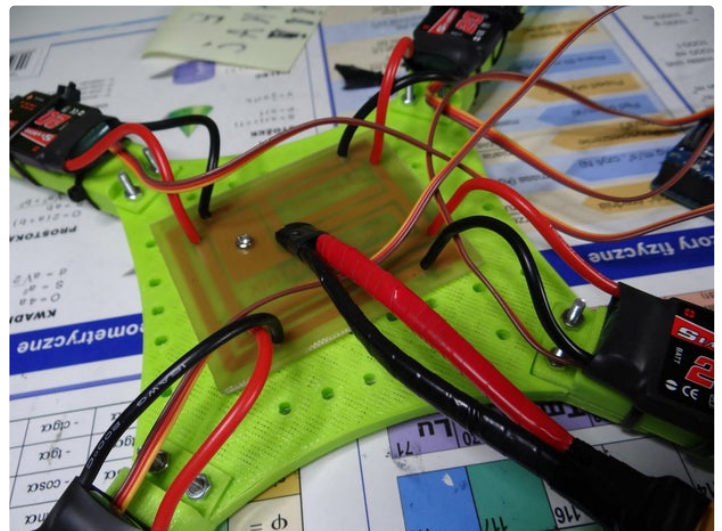
I want to write this quick story of building of this drone, if you don't want to read it just go to next step. I want to warn you that english is not my mother language and it's really common that there will be some mistakes, sorry for that.

I started building it in june 2016. Right now when I writing this it's March 2017 so it took me 9 months to make it. Mainly because of bad weather, rain, snow, windy days.

The first thing that I bought was motors, ESC's and battery from amazon. There is no amazon :(So I asked my uncle to buy me those parts (thanks uncle). Then I designed my 3D printed frame and print out everything. Then I started prototyping I mount arduino UNO with a breadboard on my frame connect MPU-6050 (this is gyroscope and accelerometer) and the best thing my "controller" :) which was breadboard with potentiometer connected with 5 meters long cables to arduino uno on my quadcopter :D It looked pretty funny and it wasn't very safe but it gave my

posibility to test my drone fast without dealing with radio yet. I also create some kind of holder for my drone to test it at home without the possibility of destroying something including me by drone. I also made some odd experiments with this drone, fortunately, my dad takes a few photos of it so you can see it above :) (thanks dad). During those experiments, I destroyed 2 lipo batteries (to be honest I don't know when and how). By testing it and flying in bad conditions or with bugs in my code I broke 3d printed frame 7 times. After 9 months of building, breaking, experimenting it is finally ready. But it's not a final version I still want to make some improvements, and I hope that you the makers will help me in it. I share all of those as open source so you can modify it, and do whatever you want (not for commercial use) remember to mention about my project somewhere.

I hope you enjoy this quick story. Right now we can start building a drone :D





Step 4: Parts

For this project we need a lot of parts, because it is actually a little bit complicated. The overall cost of them is about \$200 not so much for a drone, but it's because we create a flight controller and pilot by ourselves. Here is complete list with links to parts that I bought from different suppliers:

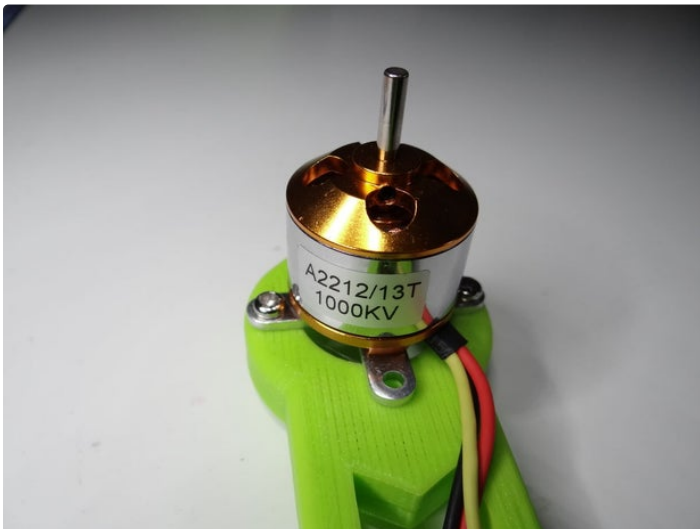
- Motors - I just bought the cheapest one, that works easily for this kind of drone.
- ESC's - depends on motors that you choose max current of ESC should be a little bit bigger than max current of motor. It must have BEC because the flight controller is powered by it. We will also 4 of them.
- Battery - 3S (11.1V) Li-Po should be around 3000mAh to get 15 mins of flight time. Make sure that it can deliver enough current.
- Frame - at the beginning it was 3D printed and in the future, it will be, but for now I used very popular F450 frame.
- LiPo charger - you must charge lipo batteries with a special charger with a balancer. Remember to choose right power plug
- Propellers - can be different but this one works the best. I advise you to buy some more, they are easy to break. They need to have a hole not thread. You choose some other color if you want
- Battery protection/monitor - never connect battery to anything without it, I destroyed 2 Lipo's because I forgot about it

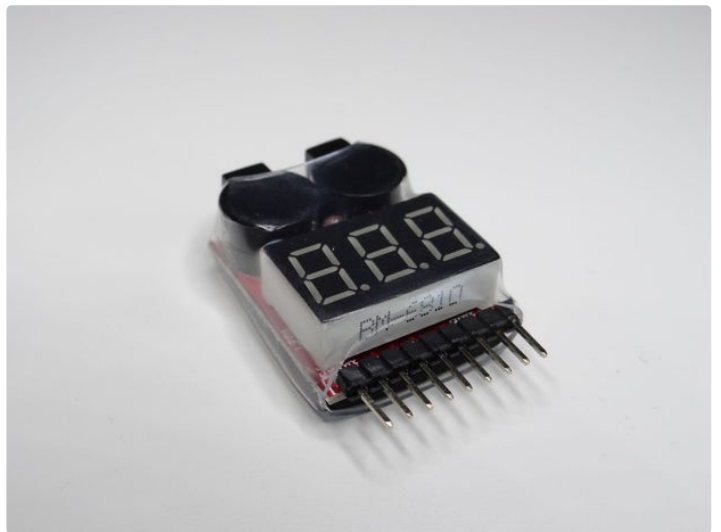
And here are parts for flight controller:

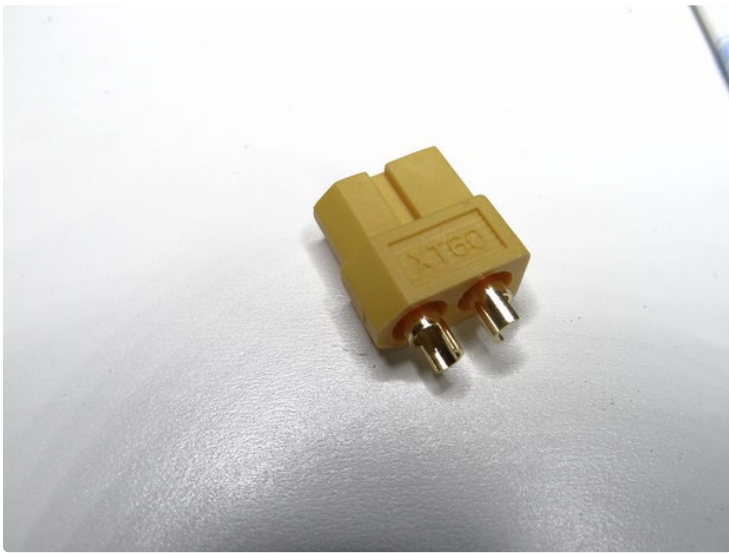
- Atmega328 - I advise you to buy it in local shop
- NRF24L01 - radio modul, we need it to create radio communication, remember to buy version with external antenna because it has much bigger range
- MPU6050 - gyroscope and accelerometer in one module
- Some smaller parts:
 - capacitors 22pF (x2)
 - capacitor 10uF
 - resistor 4,7kOhm
 - 3,3V linear stabilizer
 - some goldpins (female and male)

The last thing is pilot:

- NRF24L01 - radio modul, we need it to create radio communication, remember to buy version with external antenna because it has much bigger range
- Atmega328 - I advise you to buy it in local shop
- Joysticks - 2 of them
- Some smaller parts:
 - capacitors 22pF (x2)
 - capacitor 10uF (x2)
 - resistor 4,7kOhm
 - 3,3V linear stabilizer
 - LED diode







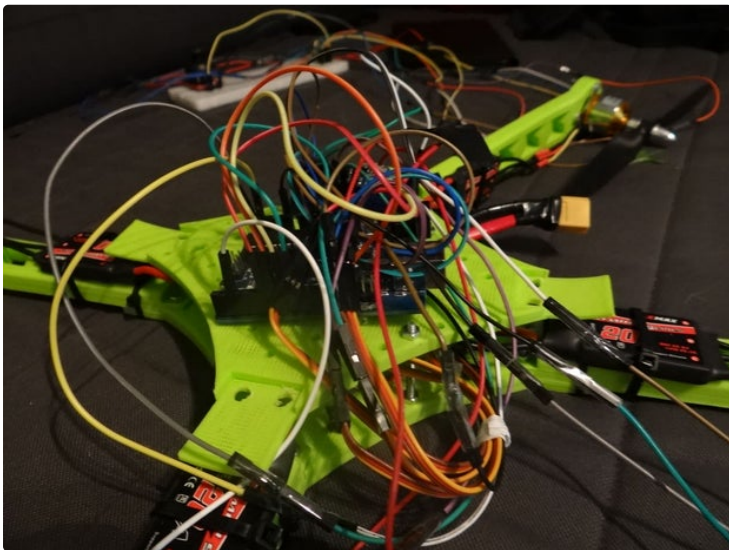
Step 5: Prototyping

The first thing to do was prototyping, to check how everything works and if circuits are good connected. I screwed up arduino UNO to drone frame and mounted mpu6050 in breadboard which was fixed to frame with double sided tape. "Pilot" which was equal to potentiometer in breadboard wired with 5m long cables to my drone. At this point I created the world's first wire drone :) After testing, in my wooden frame, outdoor tests and some changes in connections design I created eagle schematic and then PCB. Above you can see some photos of my drone with a lot of

cables on the top.

On one of those photos you can see red cable with black tape on it, guess what is this :) it's some kind of kill switch which just cut off power for arduino board.

This frame for testing a drone is made out of wood just to hold it in place and let it rotate on one axis, thats all.



Step 6: Code Explonation

Flight controller

This code isn't as hard as you can think. It just use 3 PID algorithms for stabilizing it in 3 axis (x, y, z also could pitch, yaw and roll), code for getting angles from mpu6050's DMP (digital motion processing), and also handle radio receiving and some basic math to calculate data received by radio. It stabilize itself at 100 times a second (100Hz) that's enough for this drone.

PID algorithms:

There are a lot of great explanations of how PID works, and because I not a good teacher I wouldn't try to explain how it works. I use those values for PID because after lots of test I find them best one, but they still can be better, Kd is much too big. I am not the PID tune expert and I have never done it on any other drone than this. For different motors, propellers, frame this value can be different. If you will find better one send them to me and I will test it on my drone.

MPU6050:

This code is just copied from MPU6050 library. I wanted to write my own, faster library for it but I couldn't find any good datasheet of it and I was too lazy :) If there is ready library which works fine (but could be faster) why not to use it?

NRF24L01

Here are just few lines to receive array of 8 bit numbers from pilot and convert it to throttle and rotation on x, y and z axis. You can also add more values and using them turn on lights on drone or anything you like.

Pilot

Code for pilot is straight forward. Just reads joystick values and send them over NRF24L01 that's all, nothing complicated. Radio speed is reduced to increase range.

Both codes you can find below.

```
<p>#include "I2Cdev.h"<br>#include <servo.h>
#include <spi.h>
#include "RF24.h"
#include "MPU6050_6Axis_MotionApps20.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
#include "Wire.h"
#endif
//we need this to see nrf24l01 configuration in serial
#include "printf.h"</spi.h></servo.h></p><p>//end of libraries #####</p><p>MPU6050 mpu;</p><p>float x_rotation, y_
rotation, z_rotation;
Servo motor1;
Servo motor2;
Servo motor3;
Servo motor4;
bool first_loop = true;
//radio
RF24 radio(9,10);
uint8_t data[6];
const uint64_t pipe = 0xE8E8F0F0E1LL;
long last_received;
int controll_number = 159;</p><p>//values = 5.2, 0.02, 1500
//variables for movement and positions #####
//for my quadcopter this are the best settings for pid
float x_kp = 5, x_ki = 0.02, x_kd = 1100; //values for PID X axis
int max_pid = 300;
float x_p_out, x_i_out, x_d_out, x_output; //outputs for PID
```

```

float x_now, x_lasttime = 0, x_timechange;
float x_input, x_lastinput = 0, x_setpoint = 0;
float x_error, x_errorsum = 0, x_d_error, x_lasterror;</p><p>//values = 5.2, 0.02, 1500
float y_kp = 5, y_ki = 0.02, y_kd = 1100; //values for PID Y axis
float y_p_out, y_i_out, y_d_out, y_output; //outputs for PID
float y_now, y_lasttime = 0, y_timechange;
float y_input, y_lastinput = 0, y_setpoint = 0;
float y_error, y_errorsum = 0, y_d_error, y_lasterror;</p><p>float z_kp = 2, z_ki = 0, z_kd = 0; //values for PID Z axis
float z_p_out, z_i_out, z_d_out, z_output; //outputs for PID
float z_now, z_lasttime = 0, z_timechange;
float z_input, z_lastinput = 0, z_setpoint = 0;
float z_error, z_errorsum = 0, z_d_error, z_lasterror;</p><p>//set it to 0 and see on serial port what is the value for x and y rotation, use only if your flightcontroller board is
not perfectly leveled. If your board is perfectly leveled set it to 0
float x_level_error = 0;
float y_level_error = 0;</p><p>*/
*
*
* JUNE 2016 - APRIL 2017
* C by Nikodem Bartnik
* nikodem.bartnik@gmail.com
* nikodembartnik.pl
*
*
*/</p><p>#define INTERRUPT_PIN 2 // use pin 2 on Arduino Uno & most boards
int motor1_power;
int motor2_power;
int motor3_power;
int motor4_power;</p><p>float allmotors_power = 600;</p><p>// MPU control/status vars
bool dmpReady = false; // set true if DMP init was successful
uint8_t mpulntStatus; // holds actual interrupt status byte from MPU
uint8_t devStatus; // return status after each device operation (0 = success, !0 = error)
uint16_t packetSize; // expected DMP packet size (default is 42 bytes)
uint16_t fifoCount; // count of all bytes currently in FIFO
uint8_t fifoBuffer[64]; // FIFO storage buffer</p><p>// orientation/motion vars
Quaternion q; // [w, x, y, z] quaternion container</p><p>VectorFloat gravity; // [x, y, z] gravity vector</p><p>float rotation[3]; // [yaw, pitch, r
oll] yaw/pitch/roll container and gravity vector
int safe_lock = 1;</p><p>volatile bool mpulInterrupt = false; // indicates whether MPU interrupt pin has gone high
void dmpDataReady() {
    mpulInterrupt = true;
}</p><p>void setup() {
    // join I2C bus (I2Cdev library doesn't do this automatically)
    #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
        Wire.begin();
        Wire.setClock(400000); // 400kHz I2C clock. Comment this line if having compilation difficulties
    #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
        Fastwire::setup(400, true);
    #endif</p><p>printf_begin();</p><p>Serial.begin(115200);</p><p>Serial.println("Initializing I2C devices...");
    mpul.initialize();
    pinMode(INTERRUPT_PIN, INPUT);</p><p> // verify connection
    Serial.println("Testing device connections...");
    Serial.println(mpu.testConnection() ? "MPU6050 connection successful" : "MPU6050 connection failed");
// bmp.begin();
radio.begin();
delay(1000);
radio.setDataRate(RF24_250KBPS);
radio.setPALevel(RF24_PA_MAX);</p><p>radio.openReadingPipe(1,pipe);
radio.startListening();

</p><p> // load and configure the DMP
Serial.println("Initializing DMP...");
devStatus = mpu.dmpInitialize();</p><p> // gyro offsets here
mpu.setXGyroOffset(87);
mpu.setYGyroOffset(77);
mpu.setZGyroOffset(110);
mpu.setZAccelOffset(2287);
mpu.setYAccelOffset(-1283);
mpu.setXAccelOffset(-3083);</p><p> // make sure it worked (returns 0 if so)
if (devStatus == 0) {</p><p>Serial.println("Enabling DMP...");
    mpu.setDMPEntered(true);</p><p> attachInterrupt(digitalPinToInterrupt(INTERRUPT_PIN), dmpDataReady, RISING);
    mpulntStatus = mpu.getIntStatus();</p><p>
    dmpReady = true;</p><p> packetSize = mpu.dmpGetFIFOPageSize();
} else {
    // ERROR!
    // 1 = initial memory load failed

```



```

    // 2 = DMP configuration updates failed
    // (if it's going to break, usually the code will be 1)
    Serial.print("DMP Initialization failed (code ");
    Serial.print(devStatus);
    Serial.println(")");
}

motor1.attach(5);
motor2.attach(8);
motor3.attach(7);
motor4.attach(4);
pinMode(A0, INPUT);
pinMode(A1, INPUT);
digitalWrite(A0, LOW);
motor1.write(0);
motor2.write(0);
motor3.write(0);
motor4.write(0);

radio.printDetails();

void loop() {
    if (radio.available()) {

        bool done = false;
        while (!done){

            done = radio.read(data, sizeof(data));
            // Serial.print("Controll number: ");
            // Serial.println(data[0]);

            if((millis()-last_received) < 3000){
                if(data[0] == controll_number){
                    Serial.print("DATA1: ");
                    Serial.println(data[1]);
                    allmotors_power = map(data[1], 0, 255, 800, 1500);
                    if(allmotors_power < 0){
                        allmotors_power = 0;
                    }
                }
                //allmotors_power = map(data[1], 0, 255, 800, 1600);
                x_setpoint = data[3] - 20;
                y_setpoint = data[2] - 20;
                Serial.print("PID OUT X: ");
                Serial.print(x_rotation);
                Serial.print(" PID OUT Y: ");
                Serial.print(y_rotation);
                Serial.print("Z NOW: ");
                Serial.println(z_rotation);
                Serial.print(" TIME: ");
                Serial.println(millis());
                // Serial.print("MOTORS POWER: ");
                Serial.println(allmotors_power);

            }
        }
        // Serial.println(data[1]);
        if(done == true){
            last_received = millis();
        }
    }
}

if((millis()-last_received) > 3000 && allmotors_power > 0){
    safe_lock = 0;
}

// if programming failed, don't try to do anything
// if (!dmpReady) return;
while (!mpuInterrupt && fifoCount < packetSize) {

}

// reset interrupt flag and get INT_STATUS byte
mpuInterrupt = false;
mpuIntStatus = mpu.getIntStatus();
// get current FIFO count
fifoCount = mpu.getFIFOCount();
// check for overflow (this should never happen unless our code is too inefficient)
if ((mpuIntStatus & 0x10) || fifoCount == 1024) {
    // otherwise, check for DMP data ready interrupt (this should happen frequently)
} else if (mpuIntStatus & 0x02) {
    // wait for correct available data length, should be a VERY short wait
    while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();
    // read a packet from FIFO
    mpu.getFIFOBytes(fifoBuffer, packetSize);

    // track FIFO count here in case there is > 1 packet available
    // (this lets us immediately read more without waiting for an interrupt)
    fifoCount -= packetSize;
    if(safe_lock == 1){

```

```

mpu.dmpGetQuaternion(&q, fifoBuffer);
mpu.dmpGetGravity(&gravity, &q);
mpu.dmpGetYawPitchRoll(rotation, &q, &gravity);</p><p>    x_rotation = rotation[1] * 180/M_PI - x_level_error;
y_rotation = rotation[2] * 180/M_PI - y_level_error;
z_rotation = rotation[0] * 180/M_PI;</p><p>/*
if(pressure_loop_number == 10){
    // Serial.print("Pressure: ");
    //Serial.println(bmp.readAltitude());
    pressure_loop_number = 0;
    allmotors_power = 1000;
}
pressure_loop_number++;
*/</p><p>    if(first_loop == true){
        z_setpoint = z_rotation;
        // current_altitude = bmp.readAltitude();
        //set_altitude = current_altitude;
        first_loop = false;
    }

    motor1_power = allmotors_power;
    motor2_power = allmotors_power;
    motor3_power = allmotors_power;
    motor4_power = allmotors_power;
    if(allmotors_power > 1500){
        allmotors_power = 1500;

        </p><p>
        x_output = calculatePID(0, x_rotation);
        y_output = calculatePID(1, y_rotation);
        z_output = calculatePID(2, z_rotation);</p><p>        motor1_power += x_output/2;
        motor1_power += z_output;
        motor4_power -= x_output/2;
        motor4_power += z_output;</p><p>        motor2_power -= y_output/2;
        motor2_power -= z_output;
        motor3_power += y_output/2;
        motor3_power -= z_output;

        motor1.writeMicroseconds(motor1_power);
        motor4.writeMicroseconds(motor4_power);
        motor2.writeMicroseconds(motor2_power);
        motor3.writeMicroseconds(motor3_power);
        mpu.resetFIFO();

    }else{
        motor1.write(0);
        motor2.write(0);
        motor3.write(0);
        motor4.write(0);
    }
}
}
*/</p><p> float calculatePID(int _axis, float _angel){</p><p>    // X AXIS
    if(_axis == 0){
        x_now = millis();
        x_timechange = x_now - x_lasttime;
        x_error = x_setpoint - _angel;
        x_p_out = (x_kp * x_error);

        x_errorsum = (x_errorsum + x_error);
        if(x_errorsum > 1023){
            x_errorsum = 1023;
        }
        if(x_errorsum < -1023){
            x_errorsum = -1023;
        }
        x_i_out = x_ki * x_errorsum;
        x_d_error = (x_error - x_lasterror) / x_timechange;
        x_d_out = x_kd * x_d_error;
        x_lasterror = x_error;
        x_output = x_p_out + x_i_out + x_d_out;
        if(x_output > max_pid){

```

```

        x_output = max_pid;
    }else if(x_output < -(max_pid)){
        x_output = -(max_pid);
    }
    x_lasttime = millis();
    return x_output;
}

// Y AXIS
else if(_axis == 1){
    y_now = millis();
    y_timechange = y_now - y_lasttime;
    y_error = y_setpoint - _angel;
    y_p_out = (y_kp * y_error);
    y_errorsum = (y_errorsum + y_error) * y_timechange;
    if(y_errorsum > 1023){
        y_errorsum = 1023;
    }
    if(y_errorsum < -1023){
        y_errorsum = -1023;
    }
    y_i_out = y_ki * y_errorsum;
    y_d_error = (y_error - y_lasterror) / y_timechange;
    y_d_out = y_kd * y_d_error;
    y_lasterror = y_error;
    y_output = y_p_out + y_i_out + y_d_out;
    if(y_output > max_pid){
        y_output = max_pid;
    }else if(y_output < -(max_pid)){
        y_output = -(max_pid);
    }
    y_lasttime = millis();
    return y_output;
}

// ALTITUDE
} else if(_axis == 2){
    // return (set_altitude - current_altitude) * 20;
} else if(_axis == 2){
    z_now = millis();
    z_timechange = z_now - z_lasttime;
    z_error = z_setpoint - _angel;
    z_p_out = (z_kp * z_error);
    z_errorsum = (z_errorsum + z_error) * z_timechange;
    if(z_errorsum > 1023){
        z_errorsum = 1023;
    }
    if(z_errorsum < -1023){
        z_errorsum = -1023;
    }
    z_i_out = z_ki * z_errorsum;
    z_d_error = (z_error - z_lasterror) / z_timechange;
    z_d_out = z_kd * y_d_error;
    z_lasterror = y_error;
    z_output = z_p_out + z_i_out + z_d_out;
    if(z_output > max_pid){
        z_output = max_pid;
    }else if(z_output < -(max_pid)){
        z_output = -(max_pid);
    }
    z_lasttime = millis();
    return z_output;
}

// ALTITUDE
} else if(_axis == 2){
    // return (set_altitude - current_altitude) * 20;
} else{
    return 0;
}
}

```



```

<p>*/</p><p> *
*
* JUNE 2016 - APRIL 2017
* C by Nikodem Bartnik
* nikodem.bartnik@gmail.com
* nikodembartnik.pl
*
*
*/</p><p>#include <spi.h><br>#include "RF24.h"</spi.h></p><p>#define MAX_TILT 20</p><p>RF24 radio(9,10);</p><p>int a = 0;
uint8_t data[6];
int controll_number = 159;
int safe_lock = 0;
int x_offset, y_offset;</p><p>const uint64_t pipe = 0xE8E8F0F0E1LL;</p><p>void setup(void){

    Serial.begin(57600);</p><p> pinMode(4, OUTPUT);
pinMode(3, INPUT);
pinMode(A0, INPUT);
pinMode(A1, INPUT);
pinMode(A2, INPUT);
pinMode(A3, INPUT);
digitalWrite(3, HIGH);
digitalWrite(4, HIGH);</p><p> radio.begin();</p><p>radio.setDataRate(RF24_250KBPS);
radio.setPALevel(RF24_PA_MAX);</p><p> radio.openWritingPipe(pipe);
radio.printDetails();</p><p>}</p><p>void loop(void)
{
    if(!digitalRead(3)){
        Serial.print("LOW 1");
        delay(1000);
        if(!digitalRead(3)){
            Serial.print("LOW 2");
            delay(1000);
            if(!digitalRead(3)){
                Serial.print("LOW 3");
                if(safe_lock == 0){
                    safe_lock = 1;
                }else{
                    safe_lock = 0;
                }
            }
        }
    }
    }</p><p>int power = map(analogRead(A2), 0, 1023, 0, 255);
int x = map(analogRead(A1), 0, 1023, 0, 255);
int y = map(analogRead(A0), 0, 1023, 0, 255);
int rotation = map(analogRead(A3), 0, 1023, 0, 255);</p><p>if(x > 150){
    x = map(x, 150, 255, 0, MAX_TILT);
}else if(x < 105){
    x = map(x, 105, 0, 0, -MAX_TILT);
}else{
    x = 0;
}</p><p>if(y > 150){
    y = map(y, 150, 255, 0, MAX_TILT);
}else if(y < 105){
    y = map(y, 105, 0, 0, -MAX_TILT);
}else{
    y = 0;
}</p><p>if(rotation > 150){
    rotation = map(rotation, 150, 255, 0, MAX_TILT);
}else if(rotation < 105){
    rotation = map(rotation, 105, 0, 0, -MAX_TILT);
}else{
    rotation = 0;
}</p><p> data[0] = controll_number;
data[1] = power;</p><p> // + 10 because you can't send negative number
data[2] = x + MAX_TILT;
data[3] = y + MAX_TILT;
data[4] = rotation + MAX_TILT;
data[5] = safe_lock;</p><p>radio.write( data, sizeof(data) );</p><p> delay(8);
}</p>

```



<https://www.instructabl...>

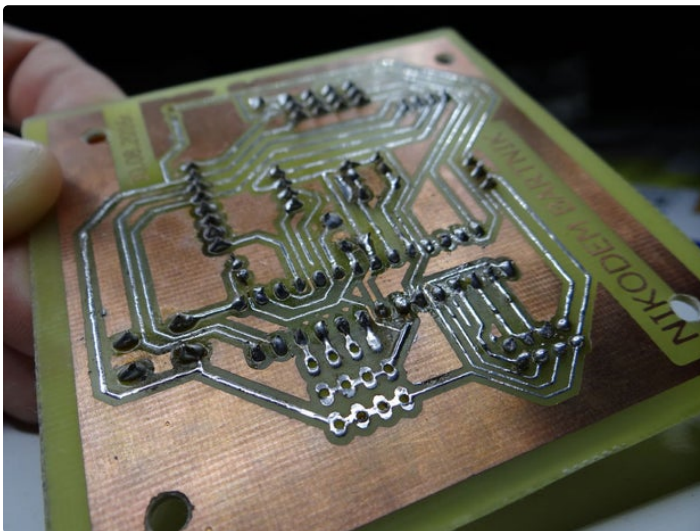
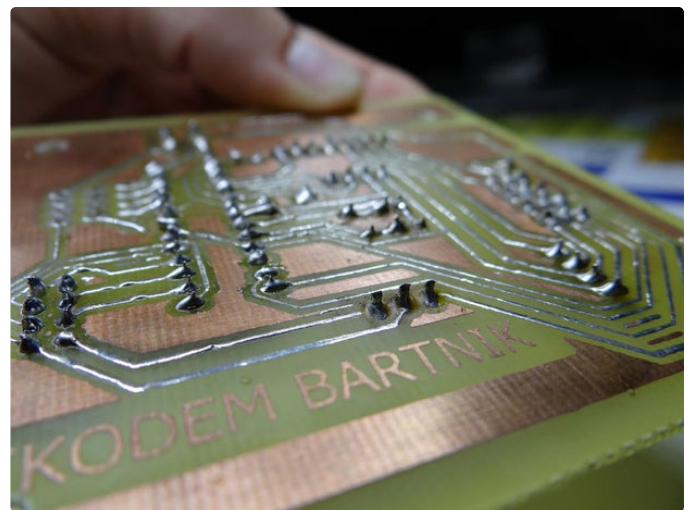
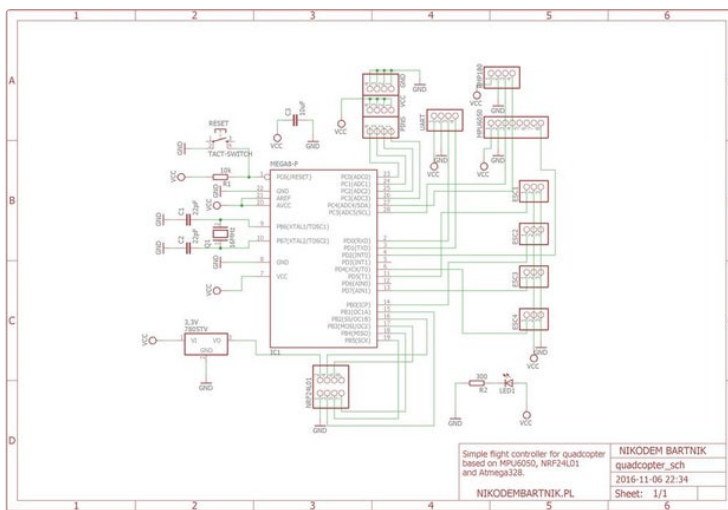
Download

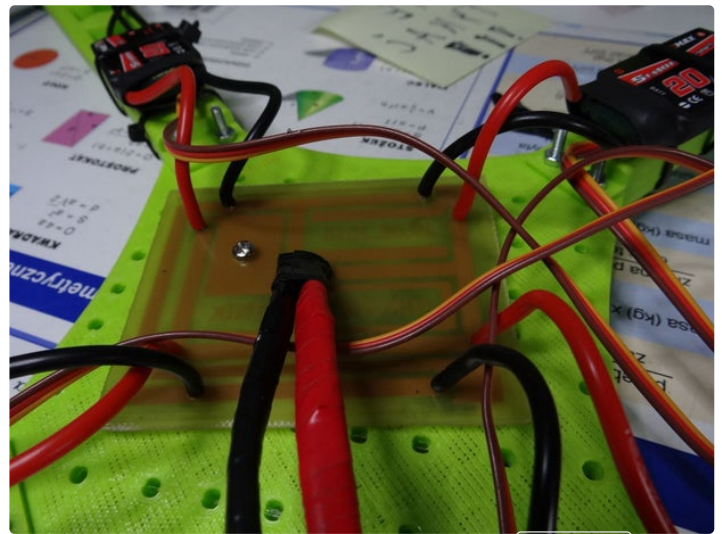
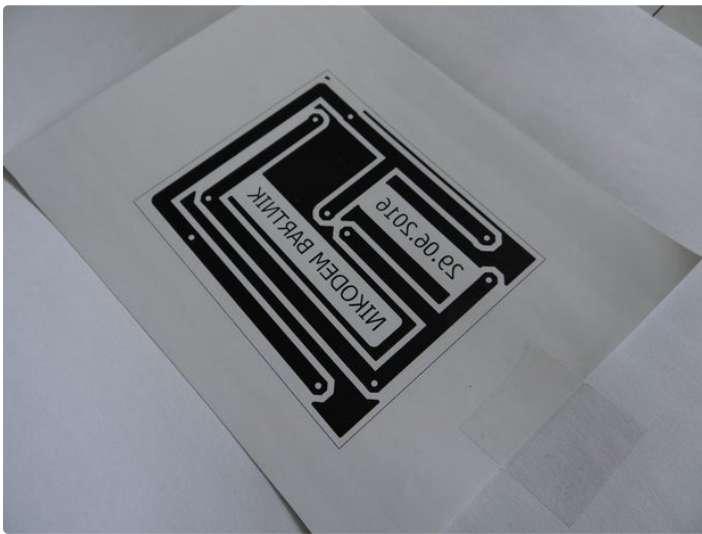
Step 7: Flight Controller PCB (and Power Distribution)




Right there you can find all files including .brd, .sch and .pdf for printing. And some photos of how I made them. Traces are very small so the board is not so easy to make. There are not a lot to write about so just enjoy the photos and files :)

I also made a power distribution board and it was used on 3D printed version of my drone, but right now I am not using it since frame that I bought have power distribution board build in to the frame.

How to make PCB's



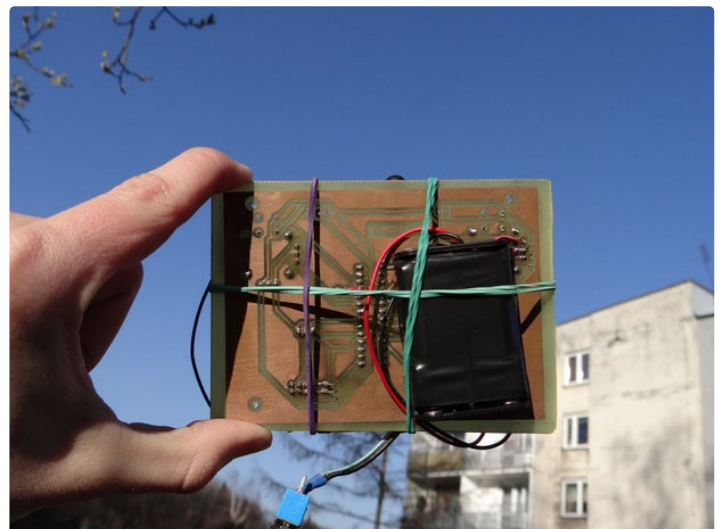
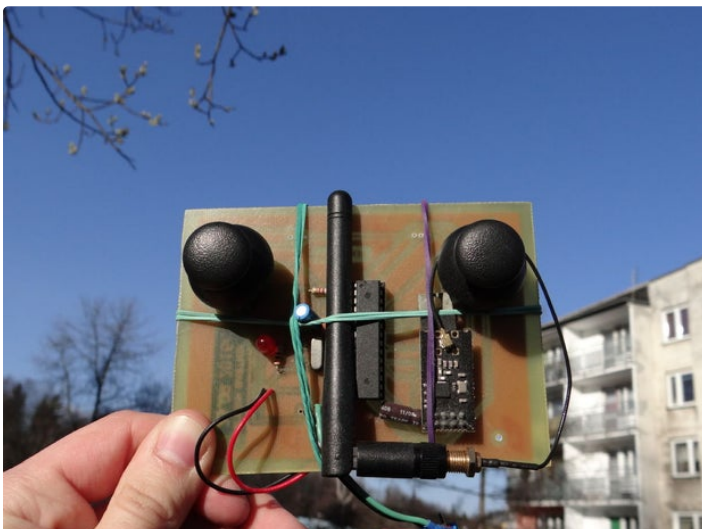


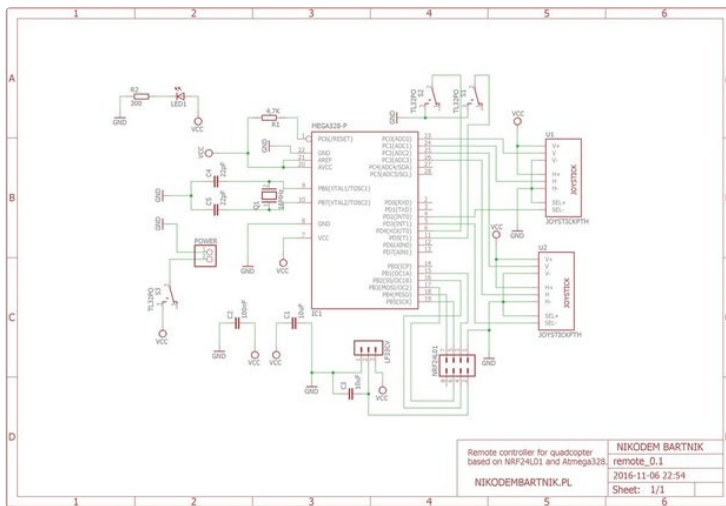
- 
<https://www.instructabl...>
Download
- 
<https://www.instructabl...>
Download
- 
<https://www.instructabl...>
Download

Step 8: Pilot

Quick story of pilot. The first version of it was potentiometer on a breadboard as I mentioned earlier. It was connected to drone with 5 meters long cables. Then I build real wireless pilot on breadboard with arduino pro mini, two joysticks and battery. Work's great but this mess with cables are so bad. So while I was waiting for my frame I decided to design and

make PCB for it. And thats the final version, max size of PCB in free eagle version to make it more handy. Looks pretty good, works perfectly. I added right there 2 switches for eventual light switching or maybe even some kind of thrower :) Schematic, PCB and program for it you can find in previous steps.





<https://www.instructabl...>

Download

<https://www.instructabl...>

Download

<https://www.instructabl...>

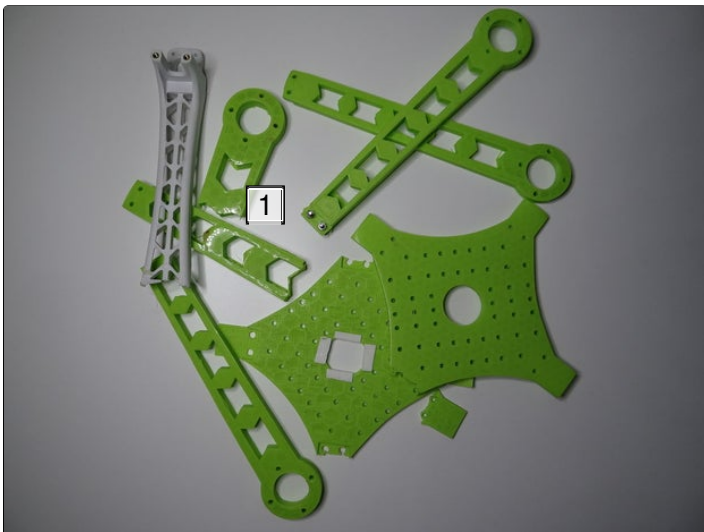
Download

Step 9: Failed 3D Printed Frame

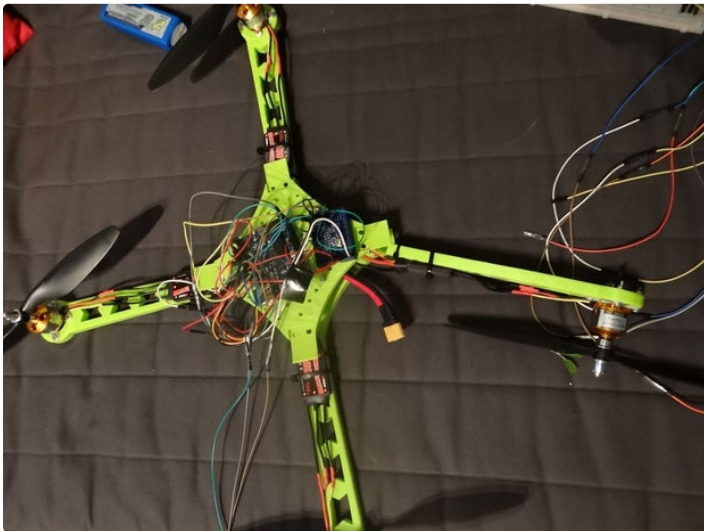
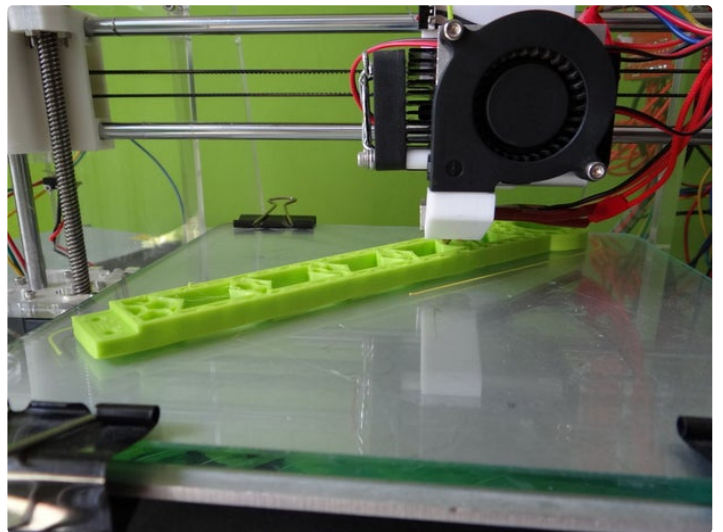
This is my 3D printed frame for my drone it's bad, don't print it. I just put it right there to show you how I made it, how it looks like and how you shouldn't design a frame for a drone. Printing time all parts for this frame is about 13 hours. I broke 7 part of it and after that I decided to give up at this point with this frame and buy one. I preferred to focus on my program and then when I finish it completely I will design comletly new frame, that will be stronger and smaller (this one is actually a little to big).

To design all of those parts I used fusion360 in my opinion the best 3D designing software. You can check it out [here](#). I also recommend you to watch [fusion360's youtube channel](#) there are a lot of great tutorials, updates and usefull tips.

I also broke down 2 Li-Po batteries :(



1. That's parts of frame that I broke



<https://www.instructabl...>

View in 3D

Download



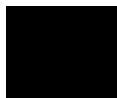
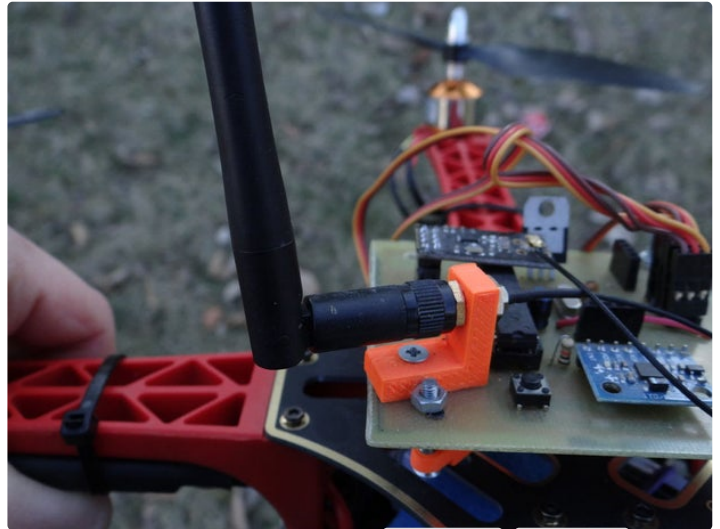
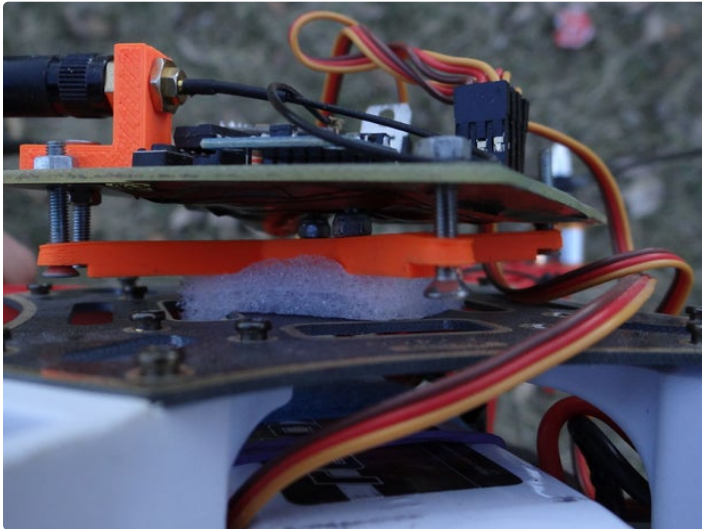
<https://www.instructabl...>

[View in 3D](#)

[Download](#)

Step 10: Some Usefull 3D Prints

Because some parts that were 3D printed are usefull I put .stl files here if you like to print them out. There is just antenna holder for flight controller (you need to drill new hole in PCB just don't drill through traces) and holder for PCB of flight controller. I put a small like sponge under this holder to limit vibrations and srcrew it down with M3 screws.



<https://www.instructabl...>

[View in 3D](#)

[Download](#)



<https://www.instructabl...>

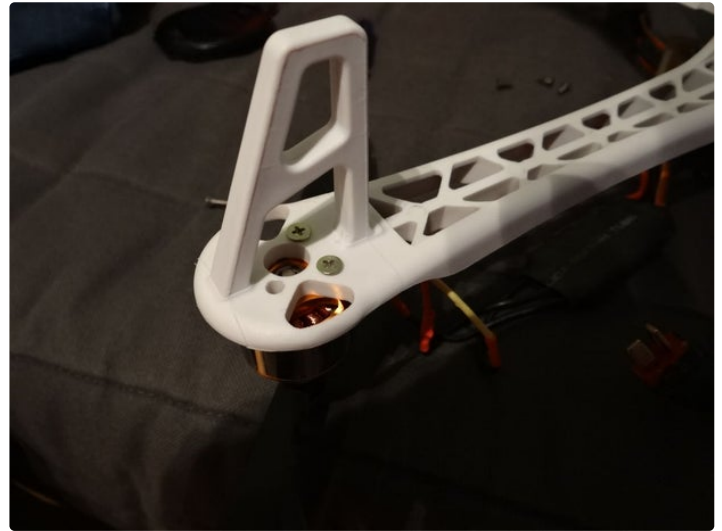
[View in 3D](#)

[Download](#)

Step 11: Assembly

There are some photos and quick explonation of how to assemble all parts of drone. First of all you have to assemble the frame, there are 4 arms and 2 main plates it's very simple. After that you can mount your brushless motor on frame. Use 4 short screws to mount them on the arms (screws can't be too long because they can damage colis of the motor). Then you can solder ESC to the motor. Remember that left

front motor and rear right must be soldered in the same way. Right front and rear left must be connected in the same way but you have to swap 2 phases of motor. I mount ESC to frame with zip tie. Remember to tight propellers very strongly, it is very usefull to use tiny screwdriver for it.



Step 12: Testing and Upgrades

I spent most time of making it on this stage. It was endless code changing, testing, code changing testing and so on. Above you can see some photos of it. One time when I test stabilization and I added piece of code to decrement throttle continuously to make it landing softly after radio disconnection (right now I know that it is impossible without barometer or GPS). I forgot to add in this code protection at some level to don't let the number infinitely down. And what

happens when number go infinitely down? At some point in to goes from max minus to max plus and then my drone has turned their motors with full throttle it goes up with this frame hit my door and wall and fall. I was on the opposite side. It was so close to hit me. And that's the reason that I wear ski helmet during rest of tests :)



1. Some kind of test, that I was thinking is good



 <https://www.instructabl...>

Download

 <https://www.instructabl...>

View in 3D

Download

Step 13: First Really Successful Flight

There were a lot of problems during this build some of them because of my errors some of them no. One time I forgot to add minus on the y axis and my drone instead of stabilize itself, was doing exactly opposite thing. But after a lot of troubles, I achieved first succesful flight, 10 perfect seonds of flying :) just up and down but it was stable and that's what I wanted to have. Above you can see video from one of the firsts flights (with 3D printed frame) and this was the first time that I thought that maybe it will finaly work. And then winter came it was snowing, it was cold and wet. I spend most time at home doing other things than drone. I was skiing and drone was waiting for better weather. At February 2017 I started working on it again I changed a program a little bit and do more test. It was better and better every test. And after some time I achieved this what you can see right now.

It's not perfect but it's very good and the most important thing it's mine! There are still things to do to make it better, the most important one is calibrating PID for roll and pitch but there is no way for me to do it right now, I haven't laptop, but I need one to go outside with it and with drone and calibrate it by changing PID constants slightly, uploading program, testing and repeat as long as it will hover perfectly. Because I haven't a laptop I can't do this :(

And here is my request to you

If you like my project and want to help me, vote for me in contests (in microcontroller contest the prize is macbook air which could help my with fine-tuning this project). Thanks!

<https://www.youtube.com/watch?v=xQPY4mWkHNU>

Step 14: The Final Version (0.1) and What's Next

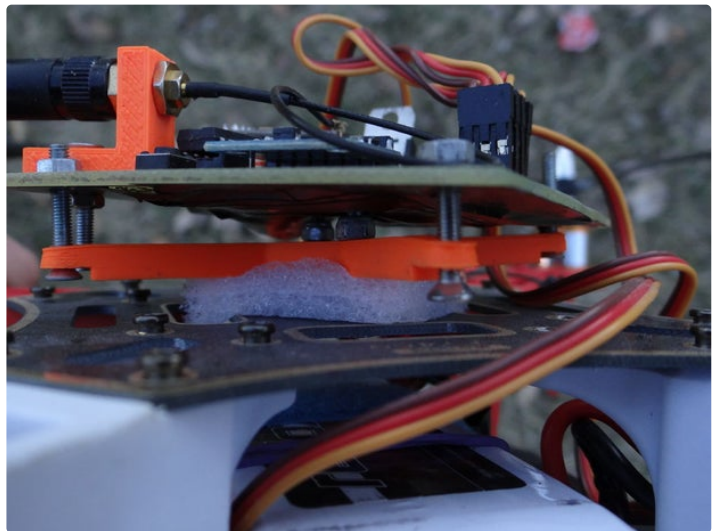
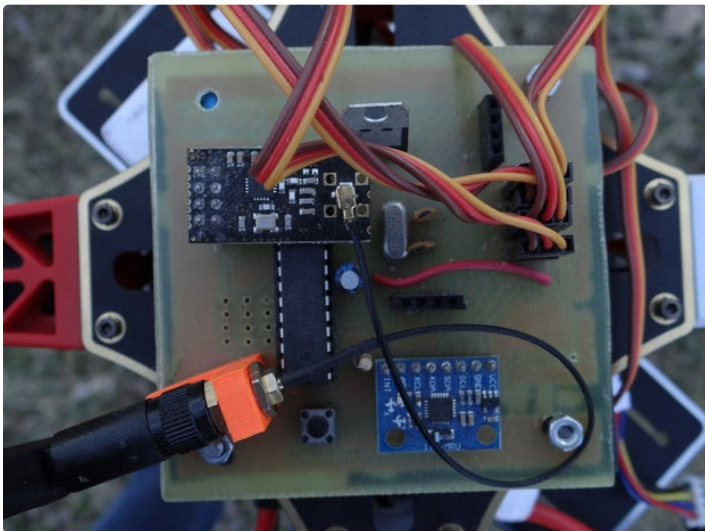
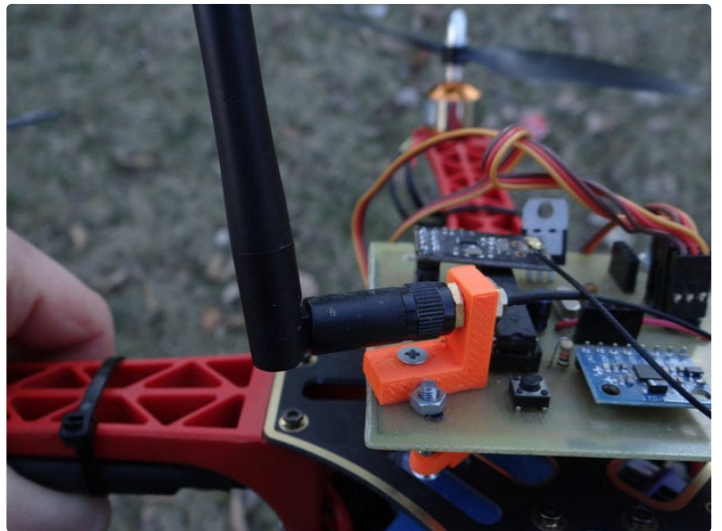
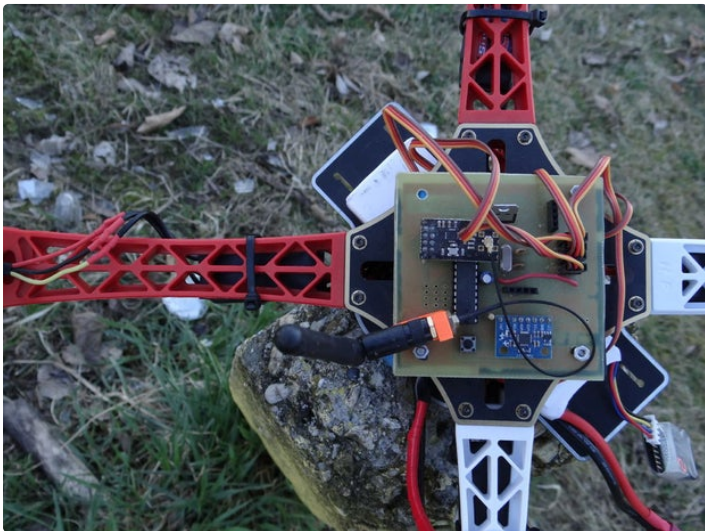
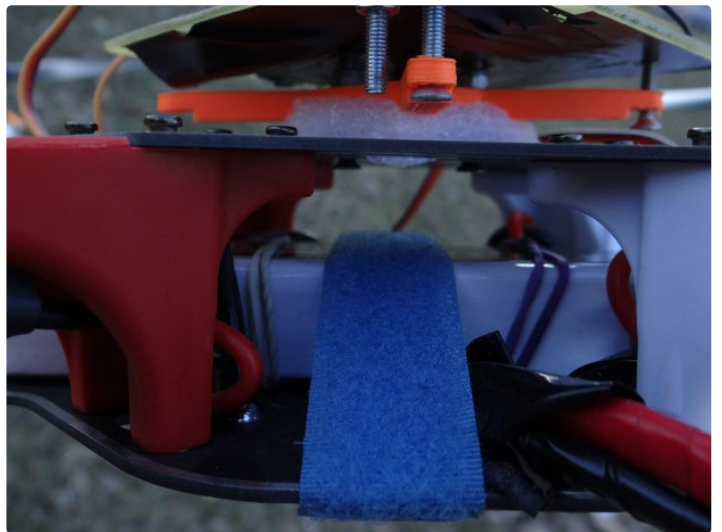
So as I said this is the (not) final version. For now I will finish this build (Last thing to do is PID calibrating). I just want to go for some other builds and learning new things. But if you want you can help me with this project, we can work together on it to make the best, the simplest arduino drone that anyone can make. If you have build my project send me some photos of it on ma mail (nikodem.bartnik@gmail.com) or in the comments bellow.

If you have any questions just ask in the comments, on my mail or on my new **facebook fanpage!** I am here to help you. You can also write what you think about this project or how to improve it.

Thanks for reading don't forget to follow me on instructables, on facebook and on youtube.

Have a nice day everyone!

<https://www.youtube.com/watch?v=PDB7qasURs4>





Arduino kodu yüklerken hata mesajı alıyorum Ne yapmalıyım.



Hi .. i want to ask you about the black wire that connecting between nrf module and its antenna?? Did it come with the module ?? Or you added it ??



Great project. Can I ask why you just didn't use the opensource Arducopter for the code? It would have saved you some damage. <http://ardupilot.org> Plenty of support from this community for the code.



I have 3d printed your design and made a quad copter. Currently i am facing problem in gyro + accelerometer sensor MPU6050, in this roll pitch angles are changing because of motor vibration. What should i do??



am new to coding, can someone help me please?



please Nikus, am new to coding, were should i start from?



Nikus can you code with Python language?



hi Nikus am new to quadcopter can you help me out ?



Hi, be sure you putted in the folder where the source file is, everything that came in the ludwik_drone_flight_controller zip file. If the arduino IDE is asking you to open the file in a separate folder, then rename the folder exactly the same as the source file (.ino) without the (.ino) extension and put the files in this folder.

You may have a folder called "ludwik_drone_flight_controller", inside this folder another one with the name of the source file without the .ino, "quadcopter_stabilization_2" and inside this last folder all the header files and cpp files that came in the zip file.

In add be sure to delete in the library folder all the rf24 libraries (make sure before that you have a copy of them) and unzip the one from here: <https://github.com/maniacbug/RF24>.

If after you did this the code still does not compile, then go to the libraries folder in your documents/arduino folder, create a new folder called LUDWIK_DRONE and put inside this folder all the files that came in the "ludwik_drone_flight_controller" zip file except the .ino file.

Now it may compile without any problem.

Greetings.



Hi Nikus,

Page not found in your (<https://github.com/maniacbug/RF24>.)

Please help.



Hi, sorry but for reasons I do not understand the period was inserted as a part of the web link...
Here you have a clean and tested one:

<https://github.com/maniacbug/RF24>

Greetings.

PD: If it does not work neither, copy the link and search it on google, it will appear for sure.



Does anyone know what Arduino I should use for the quadcopter?



You can use Arduino UNO. It's both effective and cheap.



Thanks for sharing your project. I would like to only use one or two motors to test a control circuit.
Could you help me with a schematic and code on how to do that.



Bro very amazing.

Good job!



I can't compile this code. This error is showed. Please help me to fix it



Update your libraries and would compile ok.



I update my libraries but still can't compile.



Have you tried using ARM Cortex M4? It has several advantages over the ATmega such as higher clock speeds and hardware division. There is an Arduino core for them now and some really nice dev boards



no, I don't. Maybe in the second version I will use it, but because of the speed of mpu, there wouldn't be a huge difference in stability



Recently I just picked up an ST-LINK and a couple of "Black Pill" boards. The core still needs a little work though there are issues uploading the code over USB if the dev board doesn't have the hardware reset resistors. I think that it can be fixed in software but I got a little hung up trying to trace it back to where it asserts.



I'm sorry to see this project so late and not being able to vote for. Anyone who has built a quad from scratch knows very well the amount of hours you may have dedicated to this project. In my view, your effort deserves my solid recognition. I can't vote but I wanted to share it with you.
Congratulations!



thanks man! awesome to hear that :) I actually really wanted to win a MacBook with this project to improve it and calibrate PID algorithms. Fortunately I won a MacBook with my voice controlled lamp so I will improve this project, and maybe create a Ludwik drone V2. Thanks for your comment ;)



Nieśmiertelna karteczka "nie tykać" znów namierzona. Witam ponownie i gratuluję. Nie jestem przekonany czy elementy drukowane 3D to najlepszy pomysł do drona, ale cała reszta miażdży !!!



dzięki! najlepszy z pewnością nie jest, ale bez problemu by działał gdyby tylko trochę mądrzej to zaprojektować :) w drugiej wersji to poprawię



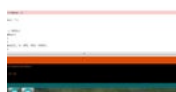
how did you upload this codes to the atmega323 ICs?.Also can you send me your codes used for make this drone with original arduino uno board.And i bought all your parts but still can't compile your code.



Code is the same no matter if you use Arduino board or my won controller. Make sure that you download a .zip file, copying just .ino file wouldn't work because you need additional files and libraries that are included in .zip



Thx for your reply.I add all libraries and try to copmile the code but still can't compile it.after i add libraries my early error was gone but a new error showed up.A picture of the error is in below. Please help me to fix this error.



Great job with the drone itself and the description!

And I thought that "Ludwik Dron" was a reference to https://pl.wikipedia.org/wiki/Ludwik_Dorn :)



Thanks! How do you know that? Are you from Poland?



Nice work. It seemed that you won 3rd price, but I think that you deserve the Grand Price. Thanks for sharing, your programming way of the PID control made me understand it. Keep on working and fly safety!!!

PD: Laws are not so hard for nearly a toy with a 400 meters range. Only take special care near airports, inform the tower, and if you are going to use FPV mode, for privacy reasons. Observe safety instructions and have fun!!!

For people that have several compilation problems, the source of them could be the rf24 library version. Use this one Ludwik wrote below: <https://github.com/maniacbug/RF24>.



Thank you!

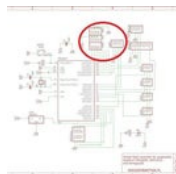


merci bien et je vous encourage



Hello I am in the process of completing this project

I have a question, What is this in the picture



i build this one, but my serial monitor cant receive data from transmitter,can you help me please?



I don't really understand all what you wrote but I enjoy it very much. thanks you



any kind of ATMEga 328 would do it ???



I'm going to have to throw my hat in the ring and disagree with this as well as it pertains to the United States (I can't reliably address other areas); addressing the points:

1. This drone would not be illegal to fly in the USA, provided it meets the FAA requirements for drones (<https://www.faa.gov/uas/>) - note the recent court ruling (May 19, 2017 - U.S. Court of Appeals for the District of Columbia Circuit in *Taylor v. Huerta*). The summary here basically reads "Owners of model aircraft which are operated in compliance with section 336 are not required to register. Owners of all other small unmanned aircraft, including newly-purchased unmanned aircraft not operated exclusively in compliance with section 336, remain subject to the registration requirement."

Section 336 basically says that if the model aircraft (drone) is for hobby or recreational purposes (ie - non-commercial usage), and weighs less than 55 lbs (~25 kg), and does not interfere with actual full-size aircraft, and is flown in compliance with a community-based national organization (like the Academy of Model Aeronautics here in the USA) - you're a-ok to fly.

Also - if you fly near an airport (within 5 miles), you have to give notice to the tower/airport in some particular manner (read it for clarification).

But basically, that's it. The part about being in compliance with an organization like the AMA (essentially being a member and following their bylaws) is there for the insurance purposes the AMA provides, along with other guidance and such. The AMA rules are basically similar, though (<http://www.modelaircraft.org/files/105.pdf>). Getting a membership with them is cheap insurance (literally) in the event of a problem, furthermore, it opens up the ability to use their various flying fields around the country.

Section 336, though, is much better (and you can thank orgs like the AMA for that, which helped to fight for this) than what the FAA had before. IIRC, section 336 was the original wording, then the FAA changed it, required registration (not of the drone - of yourself - because you could only get a single number for all of your drones!) if the weight of it went over a certain amount (half a kilogram IIRC), and more. This was challenged, fought, and ultimately found untenable by the courts and struck down.

2. The Arduino is based on the Atmel ATMEga328P microcontroller - it is a part of a family of microcontroller produced by that company (the ATMEga family). It is not "hobbyist grade" - it is a commercially available microcontroller family used in many commercial and medical-grade products (I know for a fact that it is used by some electric power chair manufacturers for their controllers - and they have to go thru a whole host of certification as medical devices).

The microcontroller itself is not the issue here, but rather how the system is designed both mechanically, electronically, and how the software integrates the two. If there are any faults with any of these parts, then things could fail. It is up to the hobbyist building such a drone to do his or her best to make sure that in the event of a failure, the drone can land as safely as possible. Note, though, that nothing can rescue a drone should a propeller shatter or a motor dies or similar - that drone is falling out of the sky. So in such a case, not flying over people is most prudent.

3. Again, I can't speak toward Europe or the UK - but I believe I have addressed everything per the United States above. While at one time (prior to May 2017) the United States (via the FAA) had a regulation in place for a short time that limited drones and required registration, that law is no longer in effect.

I should also note a couple of other things. First, when the FAA law was in effect, it did lead to something interesting: Smaller drones. Very small drones, including ones with FPV (first person

view) for fun and racing, which could be easily flown indoors (the FAA law didn't really address indoor vs outdoor flying - section 336 doesn't address this either; both mainly because no one could envision such a thing!). It's probably safe to say that if you fly indoors in a private setting, the FAA laws/rules don't apply. But you're going to be limited on drone size simply because flying larger drones indoors can be very challenging. If you do decide to do so, keep in mind the people aspect; even small drones can cause injury and damage.

Secondly, note that things change a lot if you plan to do anything commercially with a drone. Section 336 will not apply if you decide to use your drone to make money or otherwise operate in a commercial (non-hobby or non-recreational) manner. Should you get caught (note that's a big if - it isn't like there are drone police running around, and the actual cops have better things to do than police this kind of stuff) - things could get very hairy quickly. If you are serious about something like this (like flying to sell aerial footage, photos, surveying, etc) - it would be best to look into the laws (perhaps consult a lawyer even) and what you should do to stay "legal". You will likely have to register the drone with the FAA, and have private insurance in case of accidents.

Lastly - section 336 aside - as long as you aren't flying your drone like an a**hole and practicing some kind of restraint and safety, you shouldn't have any problems at all. Use common sense: Don't fly over people, don't fly near airports, don't fly near or on AMA flying fields if you don't have an AMA membership (unless you have a waiver or something from the local chapter), don't fly low over private property unless you have clearance from the owner (note that I put the "low" in there - basically, fly high enough to avoid disturbing the owners of the property - that'll be upwards of 200 feet or so, maybe a bit higher depending on the size of the drone - still, you are on your own here, and some people are definitely crazy and will try to shoot your drone down, because they believe bullets that miss won't hurt anyone, but "that darn drone is trying to peek in their winders").

Ultimately when it comes to drones or any other kind of hobbyist R/C devices (airplanes, helicopters, cars, you name it) - just be kind, courteous, respectful, and as safe as you possibly can as an operator, and comply with what few laws actually do exist. One of the greater ones (and I haven't read this instructable fully) is on what you use for a radio: Make sure it is legal to use for your R/C class (aircraft vs ground/water craft) and jurisdiction (in the USA, that would be what the FCC considers legal). Certain frequencies that may be legal in Europe (for instance, since this instructable is from a hobbyist in/near Poland if I am reading things right) that aren't in the USA and vice-versa. If in doubt, do your research. That said, 2.4 GHz spread-spectrum controllers have pretty much made that issue moot; as far as I know, they are legal worldwide. Also, using wifi for control shouldn't be an issue either for similar reasons. Using cellular phone service for control, though, may prove to be problematic, and a lot of research should be done (I am not sure what, if any laws, apply to such use). Basically, there are a lot of onerous laws regarding radio frequencies, and when you stray outside of the usage and guidelines set aside by the FCC for hobbyist R/C usage, things get tricky and problematic quickly. At a certain point, you may want to consult your local chapter of the ARRL (<http://www.arrl.org/>) for guidance.

I hope this reply gives people a little more confidence and guidance on what they can and cannot do with a hobby drone. I know it is confusing and difficult to keep up with the changing laws, but it must be done and understood. Happy flying, everyone!



is the joystick module 2 axis?



Yes that's correct. Always are some people that don't think on global safety.

But I think sellers and/or companies making mostly drones with

legislation. Fly time is limited to approx to 15 min and weight to 700 grams. Because kids don't know much about laws. I think we must have some driver license like for a car, when we want bigger drones for example, taking a movie or so.



Everybody from RC people know, that is a market regulated principles for making good business of selling drones. USA & EU don't want to people stay creative and productive anymore. Even know more as they do. Only a staying like a sheep.



I've flown RC since '89 and had my AMA card that entire time, so I appreciate your need to protect the hobby, however your facts are plain wrong. The FAA regulations (at least in the US) are based on what the drone/quad is doing (height, line of sight, range to other aircraft etc.) and not how and what it is made of. If you think the "off the shelf" drones are pre-qualified or any less flawed, then look up "drone flyaway" on YouTube. There is not a commercial drone available that hasn't fallen out of the sky.



Nobody can say hasn't fallen out of the sky.

<https://www.youtube.com/watch?v=p9T6-KPFRq8>



Are there protection for this frame to protect the propellers from hitting walls or something?

Thank you very much,

Paulo



It is always good to estimate the risks in flying, just like in any other form of traffic: privacy issues/general disturbance, safety threats to people/animals and also material damage.

Regulations do vary from country to country, even inside Europe. There is no EU wide regulation at the moment. In Finland we have very little obstacles for hobby flying. Keep it below 150 meters and obey the flight restriction zones (like military, airports). It is legal to fly even over densely populated areas.

If I was testing my own flight controller, I'd have the drone anchored somehow or have a redundant method of automatically shutting it down in case of out-of-control state. Some tests can be done without propellers, moving the drone manually and observing the reactions.

Legislation is one way into looking at things, but regardless of it, the drone builders must try to keep it safe, though the development testing isn't standardized and controlled. When the developers and other flyers play it fair, it keeps the hobby in green light, so to say.

I'm actually waiting for (free) registration for all drones, but that could come with lots of technical problems. Firstly, what is the registration for? Does it include technical inspection or is it a tax or just an ID? How to register your DIY drone? Thinking along brings up these questions. Who monitors the registration or acts upon incidents with registered drones? At first I'd go with legal ID and a "register plate" so that if a registered drone is found after hitting, say, a car, the owner can be reached. Whatever the future brings, I'm afraid there will always be people who don't know or don't care, as well as malfunctioning gear. It is not that grim after all, people get killed on bicycles, and we cannot get helmets in the law.



I have implemented method to shut drone down after losing connection or just turning off the transmitter so if my drone will do something wrong I can just shut down the remote and it will fall instantly.



Very good idea, but of course as SamiJ13 says lakes are a bit bad then. Maybe a failsafe mechanism with a recorded GPS location.

Congratulations! I'm thinking of making a drone one of these days.



That can be good for testing, but I would not want that in finished product. Transmission could be disrupted by transmitter falling from your hands etc. Not nice if the drone shuts down. But I understand it is hard to implement something safe and sure. I'm thinking now about lakes. How to equip my drone to prevent it from sinking in case it goes to water :)



Hi, How to decide Gyro and Accelerometer offset



You have to lay it flat on the ground (perfectly leveled ground) and then values that you will see on serial port will be your offset.