**Date:** 12/2/2022  
**Total hours:** 25+  
**Description of design efforts:**

**IMU**

This was mostly last week, but I created an algorithm to modify in real time the parameter controlling the relative trust the complementary filter has in the gyro vs. the accelerometer. It works by sensing when the length of the vector comprising the total acceleration felt by the accelerometer is greater in magnitude than 1G, indicating that noise or movement is taking place. The differential is the used to generate a correction factor to apply to the parameter ‘Alpha’ which estimates which measurement is more accurate.

Text

Description automatically generated with medium confidence

Fig. 15- . Source code of filter correction algorithm

In addition to the LSM6DSO, there is also an MPU6500 on the PCB, which has not as yet been used. I also wrote driver code for that this week, which wasn’t hard seeing as I already had code for it from an old project. This meant the bulk of the effort was just switching from the previous project’s SPI implementation to the current one. I then cloned the LSM6DSO driver’s calibration and sensor fusion code, and got things nearly working, and then 4

Due to my original code structure, there was no way to change or specify the CS pin, meaning that the SPI bus would blindly use the CS pin it chose on startup, rather than using different ones per device. So I created a structure that holds the IMU data that can be accessed by any other C file, as well as adding functionality and pin setup code to select which CS pin is active at that point in time, and the SPI driver will switch to using that one.

Text

Description automatically generated

Fig. 15- . SPI bus selection

With this new structure working, I tried running the full PID loop + motor + radio code to test the response of the drone. Unfortunately, I noticed immediately that there was a huge issue with the noise induced in the data by the vibration from the motors running.

I found an application that can perform a real time FFT on incoming serial data, to take a look at the noise spectrum while the motors are running. I was hoping all the noise was high frequency and could be filtered, however it was broadband, and the gyro was hitting it’s max dynamic range as well.

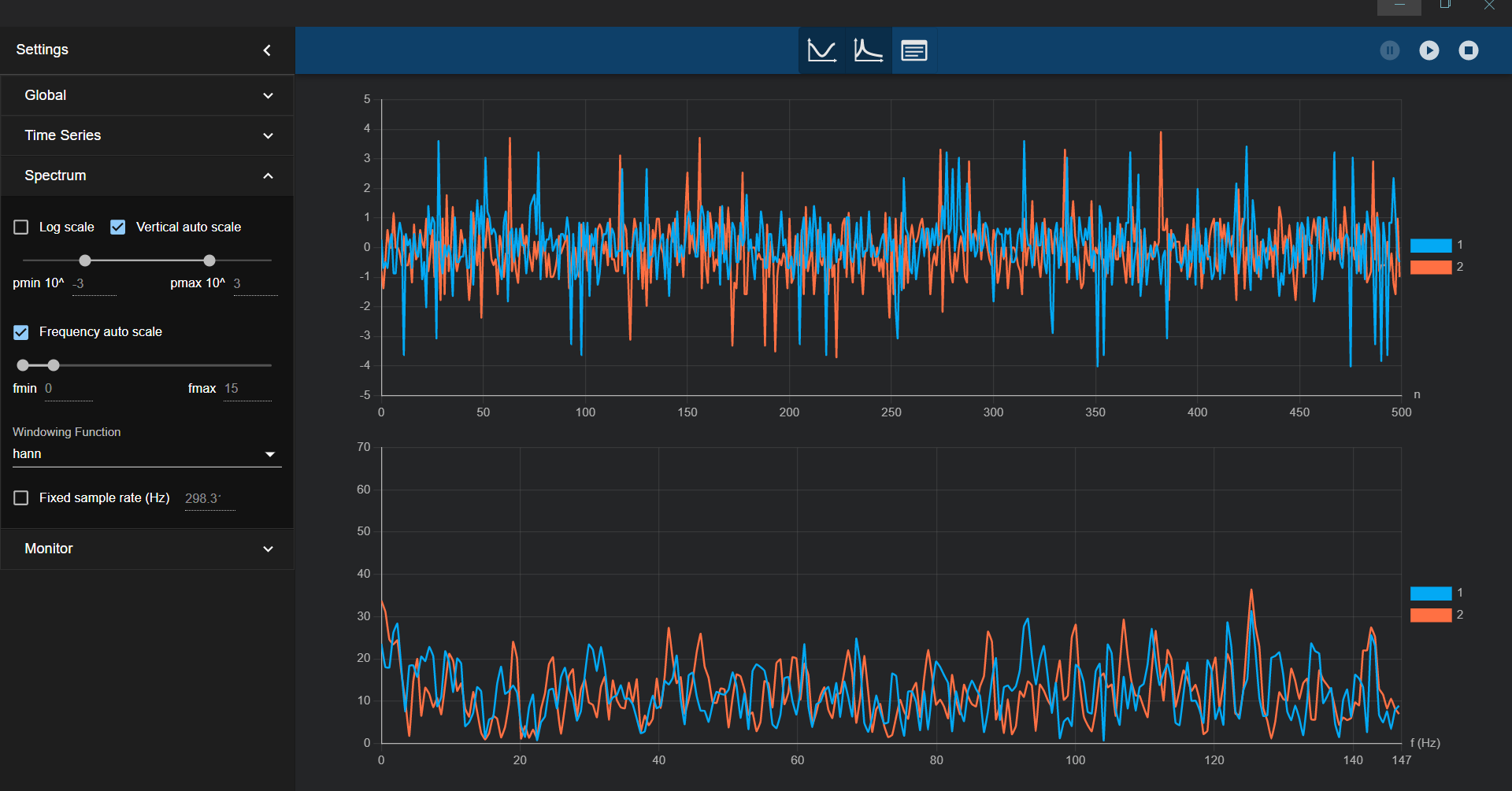
****

Fig. 15- . Serial Spectrogram

With the motors running and the drone still, the pitch and roll quickly diverged from an accurate measurement of the heading.

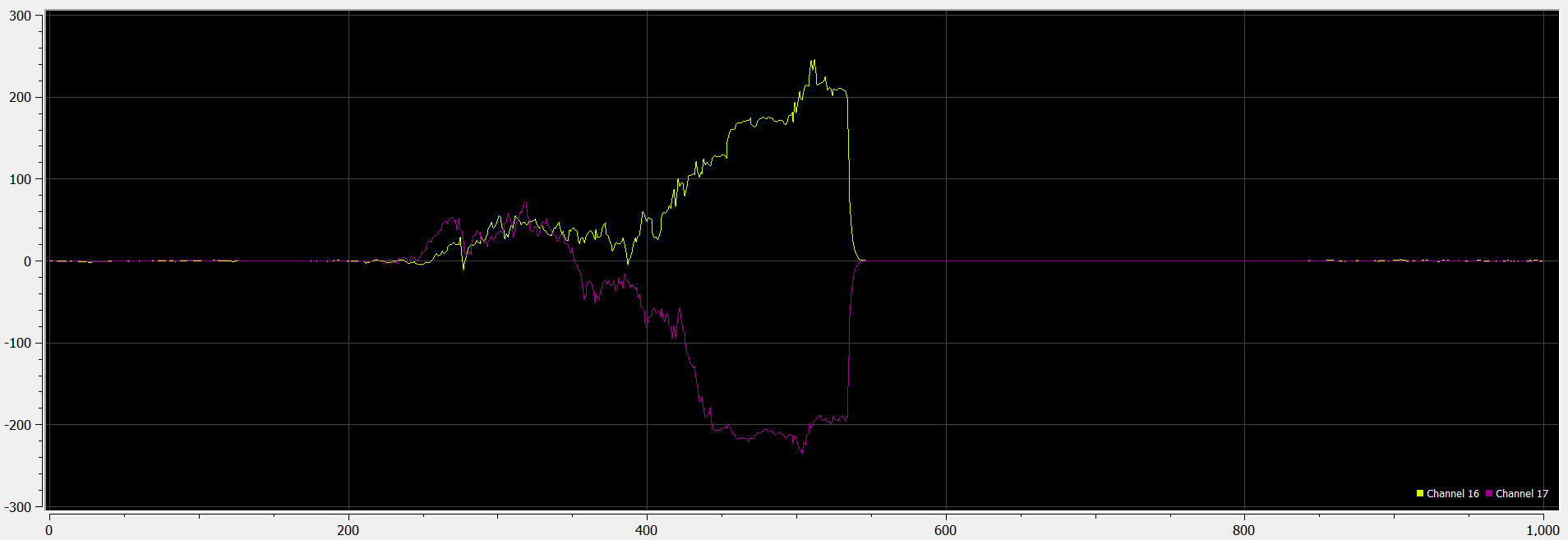
****

Fig. 15- . Very messed up gyro readings

**Mechanical**

In order to fix the vibration issues, I had to use an alternate mount to the original rigid mounting setup. I found a piece of soft anti-static foam, and glued the 3D printed mount onto the foam, and then onto the motor controller.

I initially just tried using O-rings at the mount points, however this wasn’t sufficient.

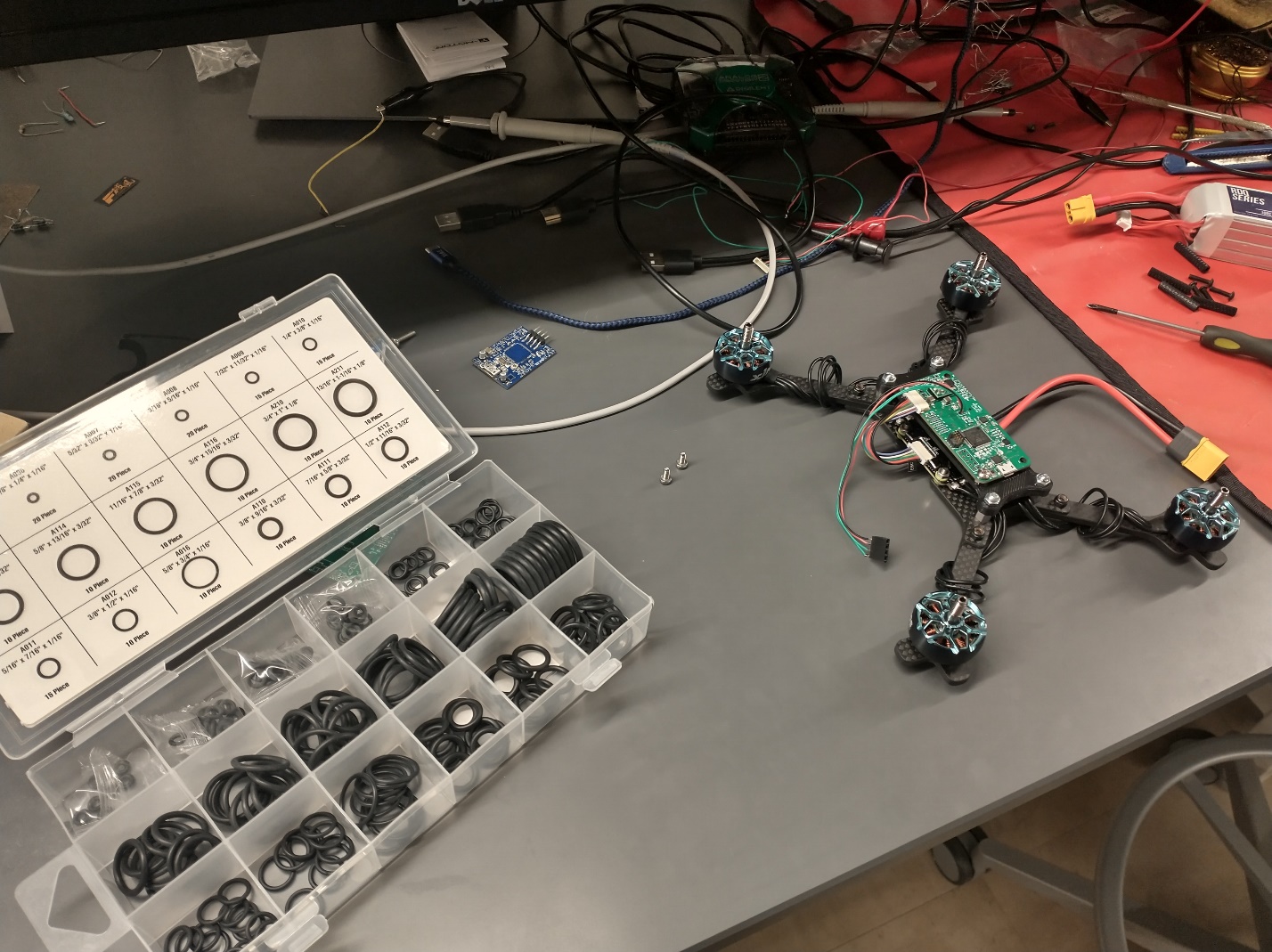


Fig. 15- . Trying to use O-rings to damped vibration

A circuit board with wires

Description automatically generated with low confidence

Fig. 15- . Trying out foam

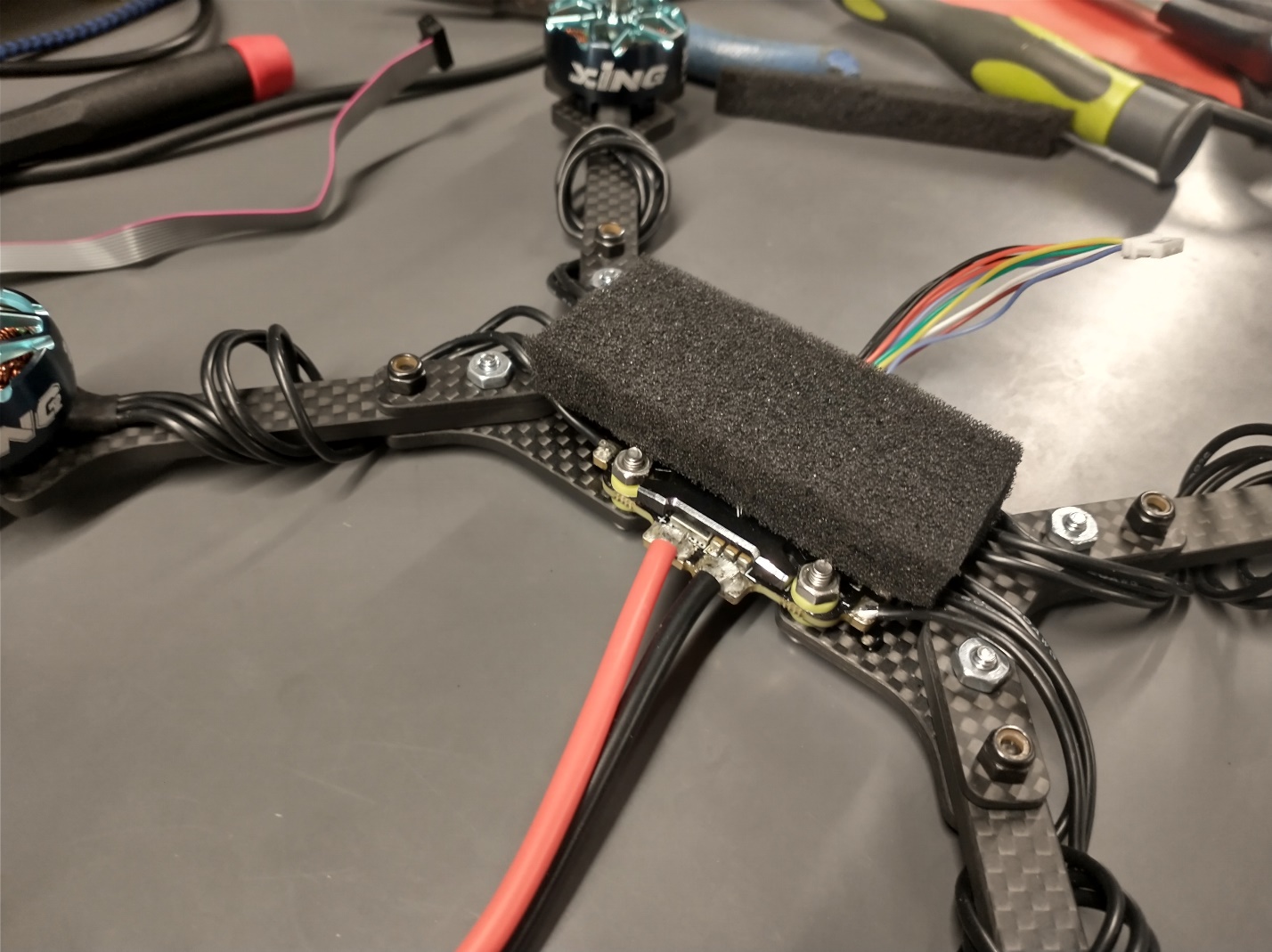


Fig. 15- . Foam sized properly

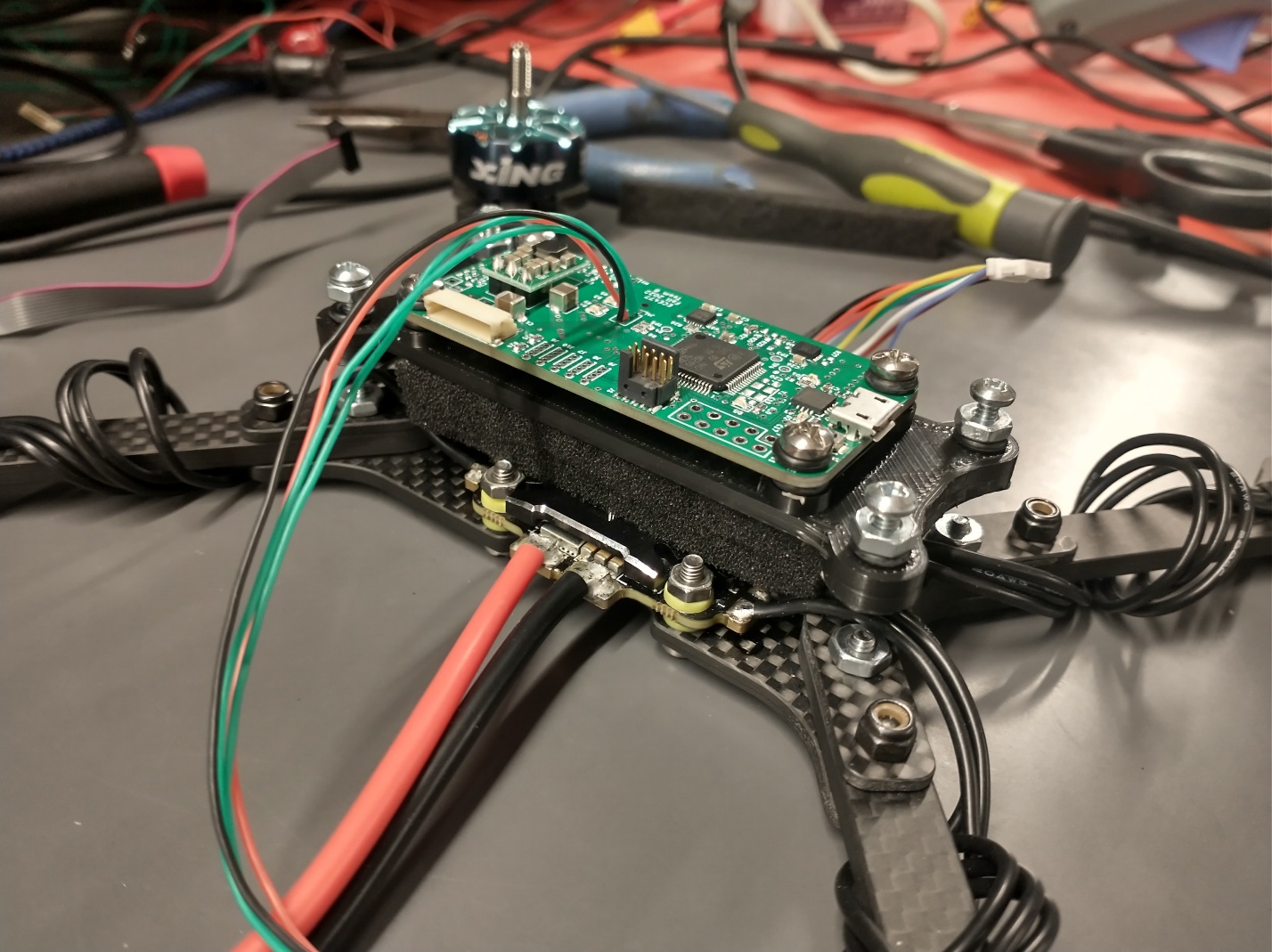


Fig. 15- . Trying foam and rubber mounts

I also mounted the battery to the bottom of the drone using some very strong Velcro-ish stuff.

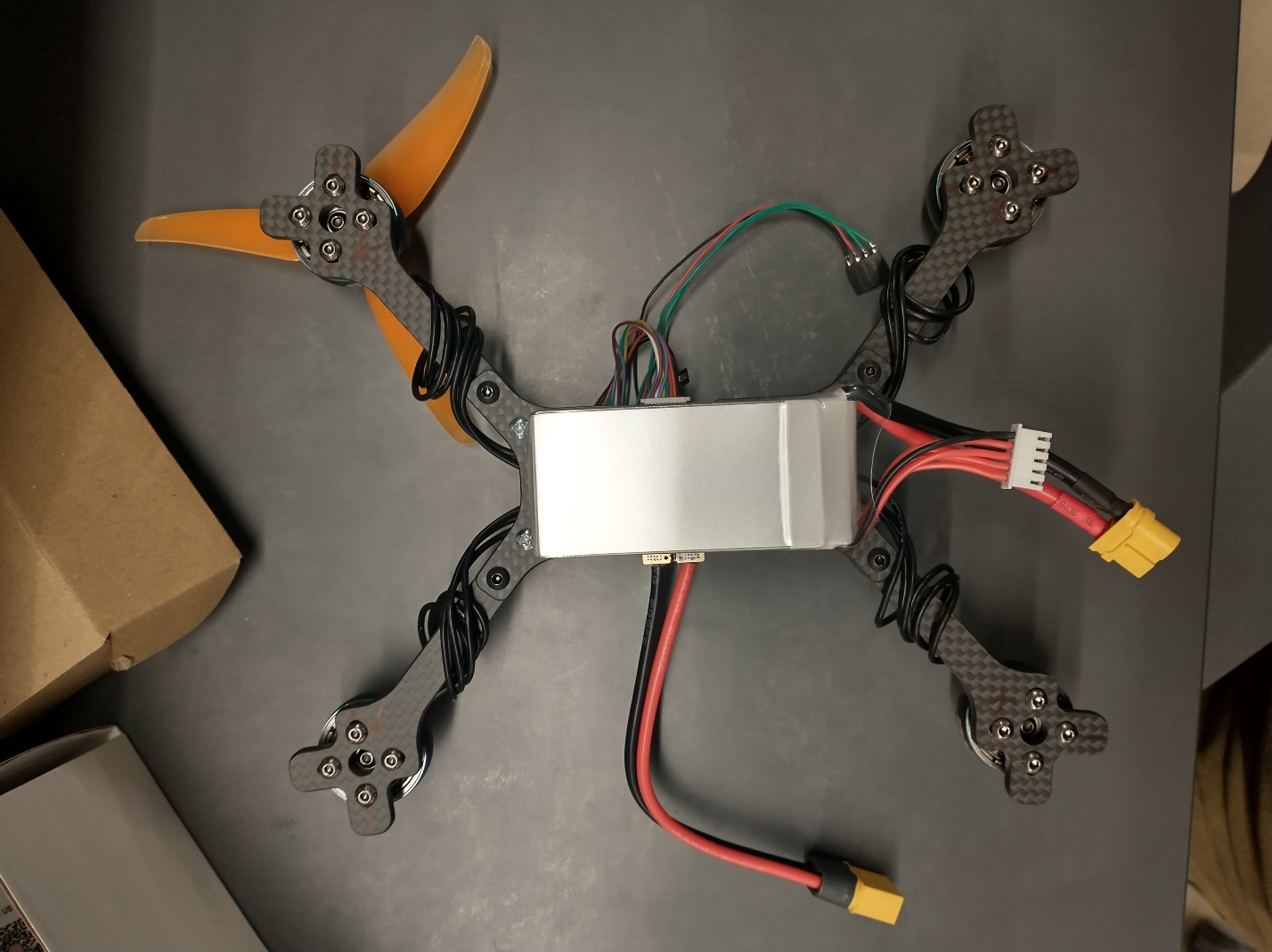
****

Fig. 15- . Battery mounted

**Electrical**

Usage of the batteries finally necessitated charging, so we used a typical multi-cell LiPo charger to charge them.

The radio also had to be permanently mounted, as it had been connected with a temporary connector initially.

**To be described later:**

**Radio mounting**

**PI UART**

**Frame setup**

**Wiring**

**Testing**

**Motor configuration**

**Flashing Nano**

**Trying other stuff**

**Wiring it up**

**Usage of app**

**Configuration and motor spin direction**

**Safety and arming code**

**Startup message**

**PID loop improvements**

**Motor LPF with hanning window**

**Show MATLAB**