**Flight Data Recorder Report**

**Objective:** Design a flight data recorder for the Design Build Fly plane that collects in-flight information. Provide the subteams with sensor data to quantify and improve the performance of the plane in preparation for the DBF competition in April.

**Parameters:** The FDR should collect data on the following in order of priority:

1. Altitude
2. Bank angle
3. Airspeed
4. Inertial Measurement Unit (orientation & acceleration)
5. Servo position
6. GPS

**Constraints:** The FDR must be independent from the control system with its own power source. It should be lightweight and compact to minimize its impact on the dynamics of the plane. Data should be collected consistently throughout the flight and be accessible to analyze at a later time. Budget should be within $500.

**Approximate Timeline:**

September: Research FDRs

October: Buy & test sensors ⇒ Build FDRv1 ⇒ Data verification

November: Build FDRv2 ⇒ 1st flight ⇒ data analysis ⇒ Buy parts for v3 ⇒ 2nd flight

December: Build FDRv3

January:

February:

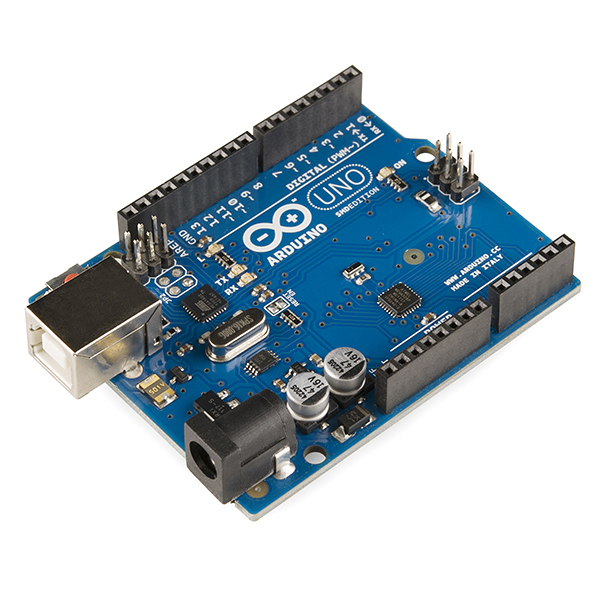
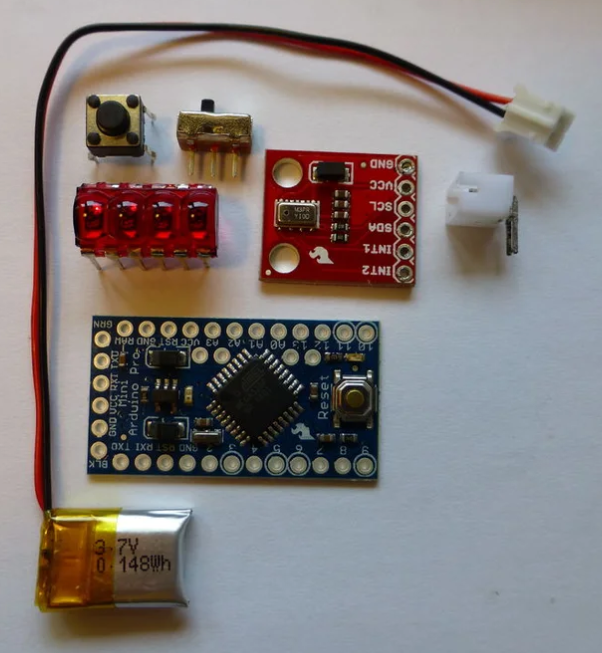
March:

April: DBF competition in Wichita, Kansas

**September:** Research FDRs

There are several flight data recorders options: Arduino, BlackBox, and Ardupilot. BlackBox is almost a plug and play data recorder, but it's primarily designed for quadcopters. An Arduino based system is very customizable and easy to program. ArduPilot has advanced features but it uses a proprietary software.

1) **Arduino** - Independent system. Records sensor data to SD card. Many board variations which allows us to customize to our needs.



**Resources**

[Instructables: Compact Altimeter](https://www.instructables.com/id/The-Ultimate-Altimeter-A-compact-Arduino-altimeter/)

[GPS Tutorial /w Google Maps Overlay](https://www.instructables.com/id/RC-Flight-Data-RecorderBlack-Box/)

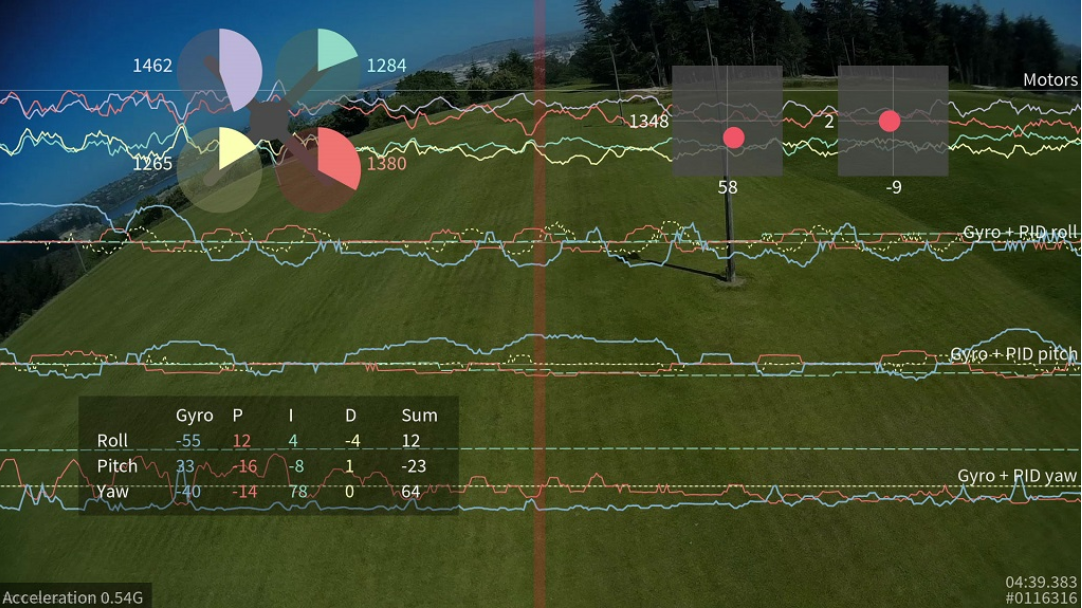
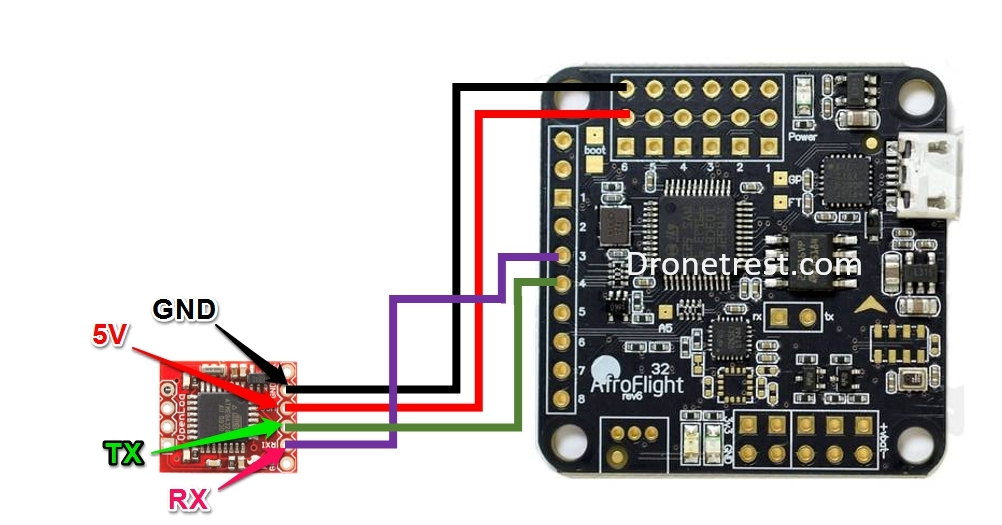
[SD Card Data Logging](https://www.youtube.com/watch?v=5Dp-XatLySM)

[Guide to SD Card Module](https://randomnerdtutorials.com/guide-to-sd-card-module-with-arduino/)

[Adafruit Mini Datalogging Board](https://www.adafruit.com/product/2795?gclid=EAIaIQobChMI7IX52J7r5AIVBIzICh1J8wCzEAYYASABEgLAz_D_BwE)

|  |  |
| --- | --- |
| **Pros** | **Cons** |
| - Allows more customizability  - different boards and sensors  - Many tutorials for individual sensors | - Difficult to interface with ground station during flight  - Might get very complicated with lots of sensors (I2C Protocol might be a solution) |

2) **BlackBox** - Basically a SD card module with a built-in gyroscope, accelerometer, and barometer. It also plugs into the flight controller to record ESC readings. It’s designed for multi-rotors so it might not be compatible with our setup. Comes with software for data playback.

****

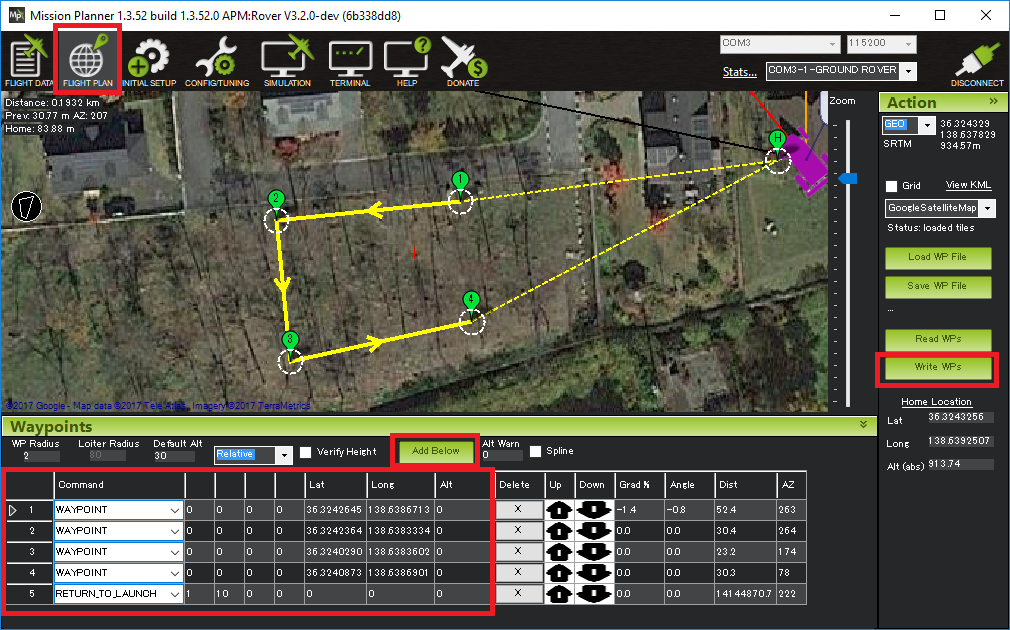
**Resources**

[Set-up BlackBox Tutorial](https://oscarliang.com/setup-blackbox-cleanflight/)

[Github Repository](https://github.com/cleanflight/cleanflight/blob/master/docs/Blackbox.md#enabling-the-blackbox-cli)

|  |  |
| --- | --- |
| **Pros** | **Cons** |
| - Only $12  - Same functionality as Arduino  - Comes with data playback software | - Requires specific flight controller: Naze32  - Designed for multirotors, not planes  - Might be limited by playback software if we need a different visualization |

3) **ArduPilot** - a flight software that runs on Pixhawk controller boards. It would be a permanent feature in the plane as it is a computerized controller that connects directly to all ESC's, servos, and sensors. It allows for full autonomous capability and ability to program flight paths.

****

**Resources**

[Types of ArduPilot Boards](https://dojofordrones.com/drone-flight-controller/)

[Autopilot plane based off Simulink](https://lubin.kerhuel.eu/project/autopilot-plane/)

[Cool RC Pitot tube guide](https://lubin.kerhuel.eu/post/pitot-build/#electronics)

|  |  |
| --- | --- |
| **Pros** | **Cons** |
| - Features 3D waypoints, auto take-off & landing, & autonomous capability  - Can be setup with radio for in-flight communication with ground station  - Can use as a controller, instead of just data recorder  - Active open source community & peripherals | - Capabilities are overkill for what we need  - $30 APM 2.6 board no longer supported  - Standard hardware is $240 Pixhawk 2.1  - Many other board options, with some under $50, but not sure how supported they are  - Learning curve to ArduPilot software |

**In Summary…**

Arduino is the most manageable option and gives us plenty of data. It is something we could implement realistically in our timeline. Additionally, its an independent system, unlike BlackBox and Ardupilot, so we could easily remove it from the plane. With Arduino, we should be able to collect all the parameters we need in-flight and store the data to an SD card to be analyzed afterwards.

**October:** Buy & test sensors ⇒ Build FDRv1 ⇒ Data verification

For v1, we will equip the Arduino with an altimeter, IMU, and SD card module. The objective is to send sensor data to the SD card and playing back the data using a Python script. We decided to buy the parts from Adafruit, as they provide documentation, tutorials, and Arduino libraries for all their parts.

**Parts**

Arduino Uno

[Adafruit Micro SD Card Module](https://www.adafruit.com/product/254)

[Micro SD card /w Adapter](https://www.amazon.com/Sandisk-Ultra-Micro-UHS-I-Adapter/dp/B073K14CVB/ref=pd_bxgy_147_2/139-6866451-1557159?_encoding=UTF8&pd_rd_i=B073K14CVB&pd_rd_r=7b7b7fbf-4527-4c58-82c2-7356fead953c&pd_rd_w=GfYjH&pd_rd_wg=hpalZ&pf_rd_p=3edd75bb-e36e-488e-b666-80dd1a52c658&pf_rd_r=VB5PA4RM1WVNZH9ZCD08&psc=1&refRID=VB5PA4RM1WVNZH9ZCD08)

[Adafruit BMP388: Altimeter & Barometric Pressure](https://www.adafruit.com/product/3966)

[Adafruit BNO055 IMU](https://www.adafruit.com/product/2472)

I followed the Adafruit tutorials for each of the sensors. They show how to wire the sensors to the Arduino and how to read the sensor data on the Arduino. I used a breadboard for v1 because it allows for rapid and flexible prototyping. The IMU and Altimeter are wired using I2C protocol. Both sensors support I2C which allows them to be wired on the same Arduino pin. This helps us conserve the amount of pins we use on the Arduino.

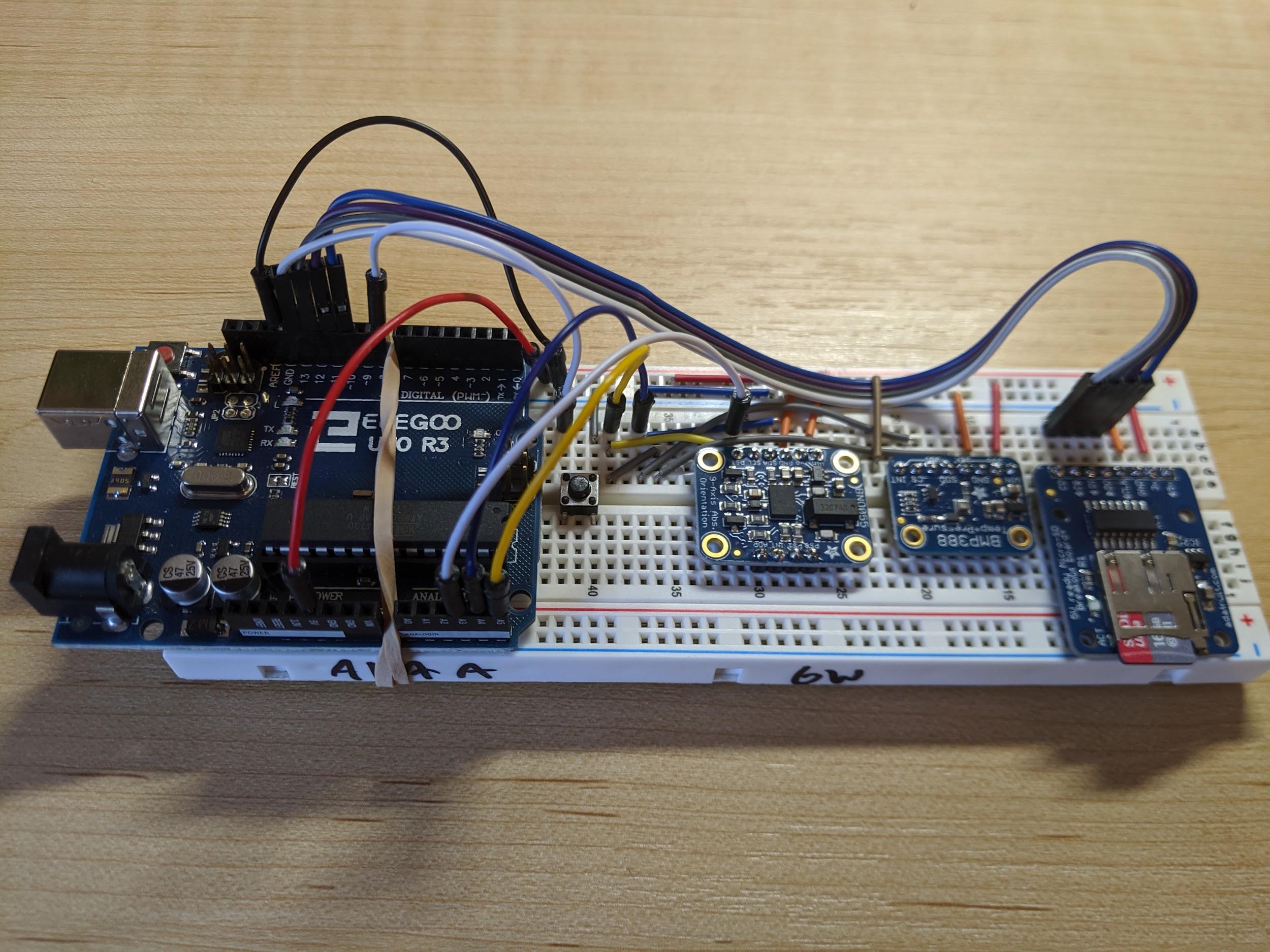
**Adafruit Tutorials**

[SD Module Guide](https://learn.adafruit.com/adafruit-micro-sd-breakout-board-card-tutorial?view=all#introduction)

[IMU Guide](https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview)

[Altimeter Guide](https://learn.adafruit.com/adafruit-bmp388?view=all)

**FDRv1** (after rewiring)

****

**Testing Sensors**

* Install libraries
* Print sensor data (IMU, temp, and altitude) to serial port

**Test Writing Data to Text File**

The FDR will poll all the sensors every 100 ms and write the data to a text file on the SD card. Each line contains 6 floats. They correspond to heading, pitch, roll, temp, pressure, and altitude.

**Testing Data Visualization**

* Experimented with Adafruit BNO055 orientation 3D model interface using Processing
  + <https://www.allaboutcircuits.com/projects/bosch-absolute-orientation-sensor-bno055/>
  + Found fixed jet processing with aerospace measurements: <https://forums.adafruit.com/viewtopic.php?f=19&t=137357>
  + Replaced jet with plane file
* Experimented with transmitting temperature, pressure, and altitude values to serial port
  + Initially it only sent one reading per line
    - Oscilloscope.py only read every 3rd line
  + Improved to read multiple pieces of data in one line
* Working on writing my own live data graph python script that can take data from txt file or serial port

**Working Demo**

**November:** Build FDRv2 ⇒ 1st flight ⇒ data analysis ⇒ Buy parts for v3 ⇒ 2nd flight

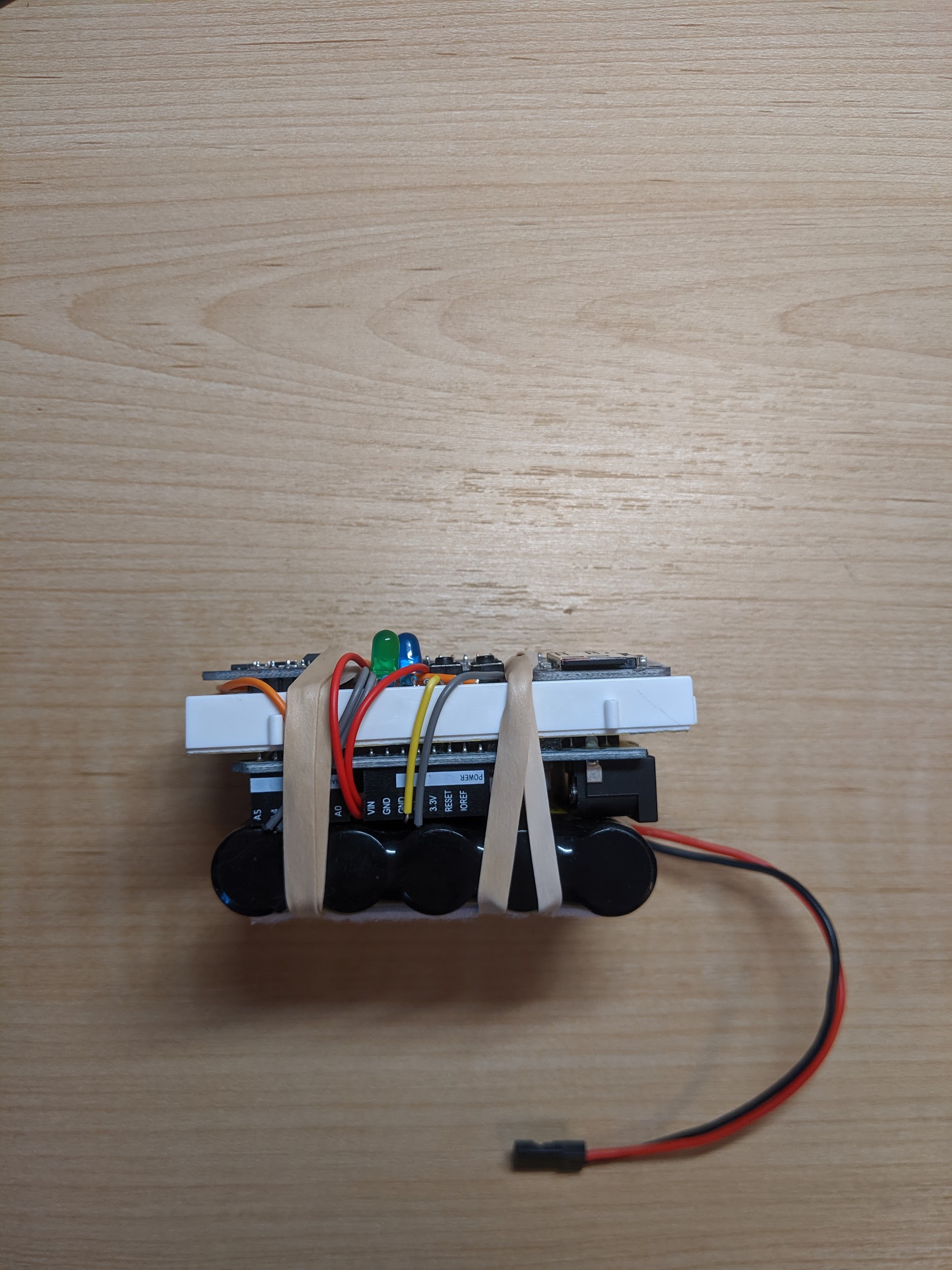
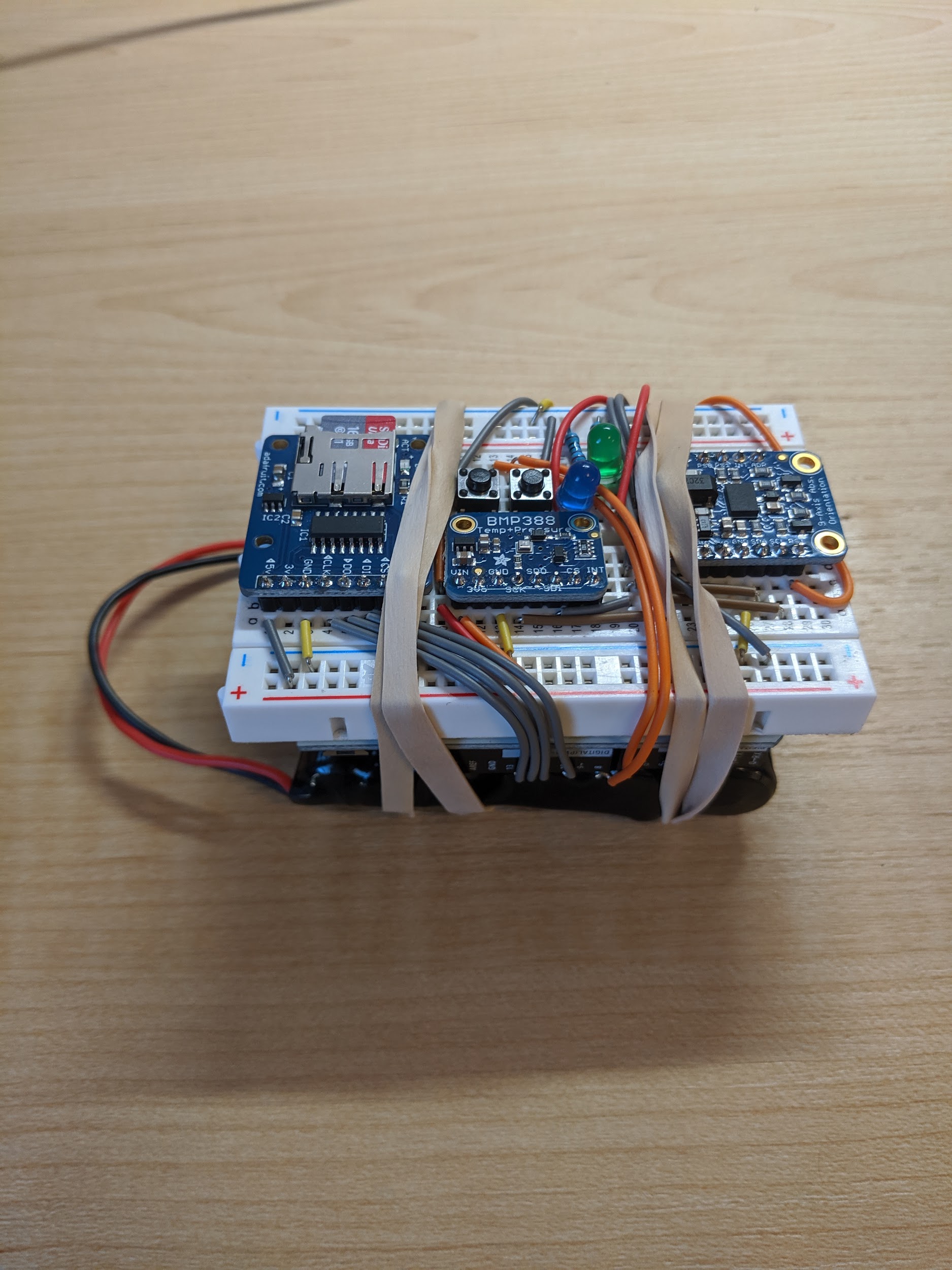
To prepare the FDR for its first flight, I decided to upgrade it to FDRv2 which would have a smaller form factor. I decided to move the components to a breadboard that was half the size and mount the Arduino on the bottom of the breadboard. I wrote down the pin layout to make rewiring easier. I also used flat jumper wires to keep the profile neater. FDRv2 also used a 4 cell battery that was rubber banded to the bottom.

**Pin Layout**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SD Module** | | **BMP388** | | **BNO055** | |
| **CD** | N/A | **INT** | N/A | **RST** | A3 |
| **CS** | D10 | **CS** | N/A | **SCL** | A5 |
| **DI** | D11 | **SDI** | A4 | **SDA** | A4 |
| **DO** | D12 | **SDO** | N/A | **GND** | GND |
| **CLK** | D13 | **SCK** | A5 | **3Vo** | N/A |
| **GND** | GND | **GND** | GND | **VIN** | 5V |
| **3V** | N/A | **3Vo** | N/A |  |  |
| **5V** | 5V | **5V** | 5V |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SD Button** | | **SD LED** | | **Reset Button** | | **Status LED** | |
| **LEFT** | D7 | **POS** | D6 | **LEFT** | D8 | **POS** | D5 |
| **RIGHT** | GND | **NEG** | GND | **RIGHT** | GND | **NEG** | GND |

**FDRv2**

****

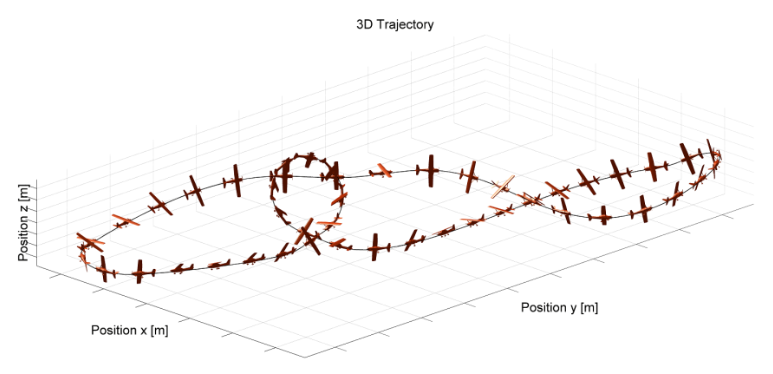
**Flight Day**

**Flight Data Processing**

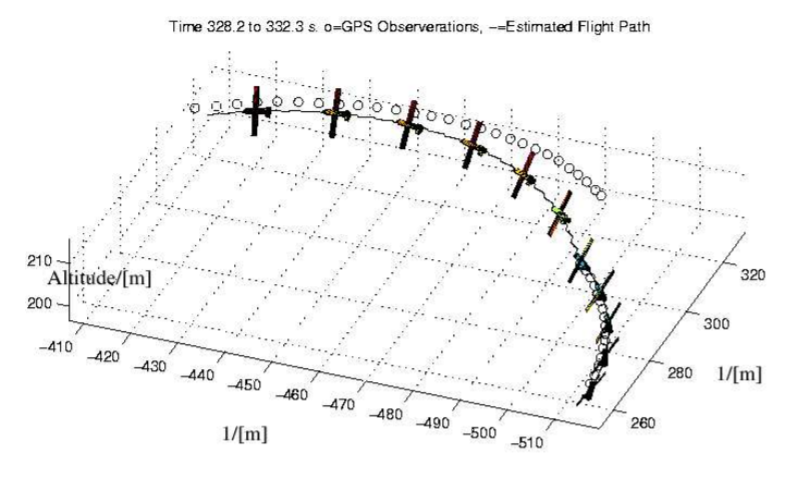
**Preparing for FDRv3**

**Flight Gear/Gazebo Flight Sim/Matlab aerospace toolbox**

**Goal for Flight Visualization**

****

<https://www.fsd.lrg.tum.de/research/trajectory-optimization/>



Page 106: <https://www.researchgate.net/publication/228787754_On_a_matlabjava_testbed_for_mobile_robots>

Pitot tube

<https://makersportal.com/blog/2019/02/06/arduino-pitot-tube-wind-speed-theory-and-experiment>