# High Altitude Weather Balloon Launch for Measuring Environmental Pollution

Major Qualifying Project

*A Major Qualifying Project Report Submitted to the faculty of the Electrical and Computer Engineering Department at Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science*

Submitted by:

Leo Gross

Lucas Falsarella Guerreiro

Tae Hyun Je

Zachary Langois

Advised by:

Prof. Maqsood Ali Mughal

Prof. Selcuk Guceri

April 6th, 2021

## Abstract

The rise in environmental pollution has rapidly increased in the past century, causing a wide range of issues including harmful impact on human health, climate change, socio-economic impacts, etc. That is why it is important that we continually monitor and collect data on greenhouse gas (GHG) emissions and take timely measures to reduce these negative impacts. To better understand GHG emissions at different atmospheric levels, we launched a relatively low-cost weather balloon equipped with atmospheric sensors and other electronics into the stratosphere, reaching heights of 70,000 ft and more. Our project recorded the concentration levels of GHGs and captured near-space imagery data. The payload was retrieved safely after the complete duration of the flight using the SPOT Gen 3. Our team launched a total of 2 weather balloons. However, we did not meet our criteria, but a solid foundation was established for future iterations.

## Acknowledgements

We would like to thank Professor Maqsood Mughal for his guidance on our project, his knowledge of environmental science and electrical engineering was vital to the accomplishments of this project. His undying support throughout our successes and failures kept our team focused on the task at hand. We would also like to thank staff from the Federal Aviation Agency (FAA) and the Air Traffic Control in Boston, MA and Albany, NY for their assistance before and during our launch. For making our launches possible, we would also like to thank Michael’s Party Rentals for providing us with helium gas.

# Authorship

| **Section** | **Primary Author(s)** | **Editor(s)** |
|---|---|---|
| Title Page | All | All |
| Abstract | Je | All |
| Acknowledgements | Je | All |
| Authorship | All | All |
| List of Figures | Je | All |
| List of Tables | Je | All |
| Table of Contents | Je | All |
| Introduction | Je & Gross | All |
| Background | Guerreiro | All |
| Methodology | All | All |
| Results | All | All |
| Conclusion & Recommendations | Langois | All |
| References | Je | All |
| Appendices | Je | All |

# List of Figures

# List of Tables

# Table of Contents

# Chapter 1. Introduction

## 1.1 Significance

The goal of this project was to implement two new features, real time data communication and the ability to terminate the flight of the High-Altitude Balloon (HAB). Real time data communication was achieved using the Wi-Fi wireless communication protocol, while the flight termination was achieved using a novel heated strand cutting element. This will allow for easier collection of greenhouse gas (GHG) data and better control of flight as compared to traditional methods.

## 1.2 Scope

The scope of this project was to build a HAB capable of measuring GHG concentrations as it ascended to the Stratosphere. The planned final altitude was at least 70,000 ft above sea level. To capture the imagery data during its ascent, a GoPro was needed. The HAB also needed a way to transmit back its global positioning system coordinates for retrieval purposes. Furthermore, it needed to be capable of transmitting sensor data back down to the ground via Wi-Fi bridge every ~5 seconds, where the data would then be uploaded online for real time graphing. The environmental data required multiple sensors, which consisted of a UV sensor, Ozone sensor, Nitrous Oxide sensor, and Carbon Dioxide sensor. Also, due to the intricacy of this project, it was paramount that this project possessed a well-documented guide explaining our thinking process, equipment/component selection, and code design.

# Chapter 2. Background

In recent years, there has been a large focus on greenhouse gas (GHG) emissions and their detrimental effects such as global warming and pollution. Air pollution alone is responsible for over 60,000 deaths in the United States each year (Rowell & Rossman, 2015). The use of carbon-based fuels has been as widespread as its consequences throughout history. Pollution has had an impact on both on the stability of the environment and population health. Over the last century, events such as extended periods of smog have caused death and health complications to residents of many industrialized cities. Incidents such as Donora, Pennsylvania in 1948, where a five-day smog killed 20 people in a town of over 14,000 residents, and London, United Kingdom in 1952, where several days of smog killed over 4,000 people by causing damage to the population's respiratory systems.

Understanding this danger is only the first step in combating the problem. Over the past few decades, there has been an increase in awareness and focus regarding pollution and carbon-based fuels emissions. In the United States, this effort is partially led by the Office of Air Quality and Planning and Standards, founded by the Environmental protection Agency (EPA). The EPA measures a set of air pollutants on a yearly basis. These measurements include sulfur dioxide ($SO_2$), nitrogen dioxide ($NO_2$), particulate matter, carbon monoxide (CO), ozone ($O_3$), and lead (Pb). Additionally, the Interrogative Panel on Climate Change (IPCC) of the United Nations lists Carbon Dioxide ($CO_2$), Methane ($CH_4$), and Nitrous Oxide ($N_2O$) as the main GHGs based on emission (Ahmed & Dadhich & Olivier & Rogner & Sheikho & Yamaguchi, 2015).

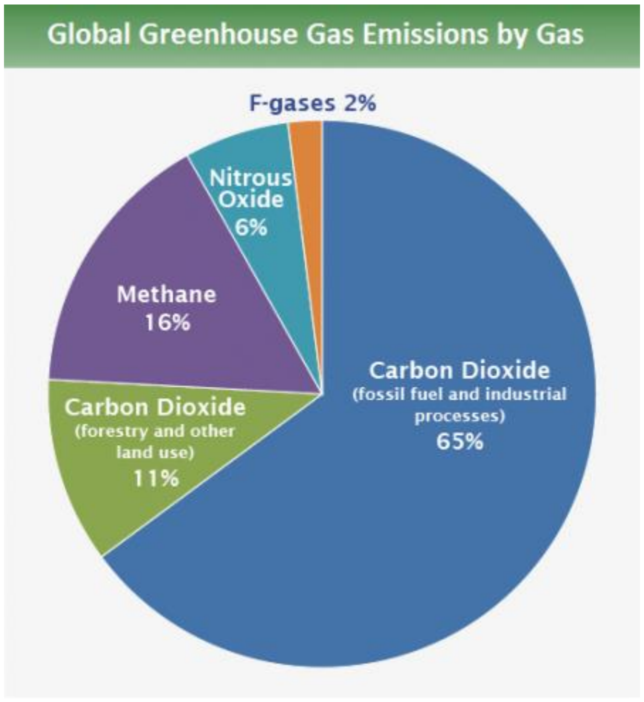


*Figure 2.1 GHG Emissions by Gas. Source: IPCC (2014)*

Observing a specific GHG emission over the years will reveal an upward trend. A study in 2017 shows this trend specifically regarding $CO_2$ emissions (Boden & Marland & Andres, 2017).

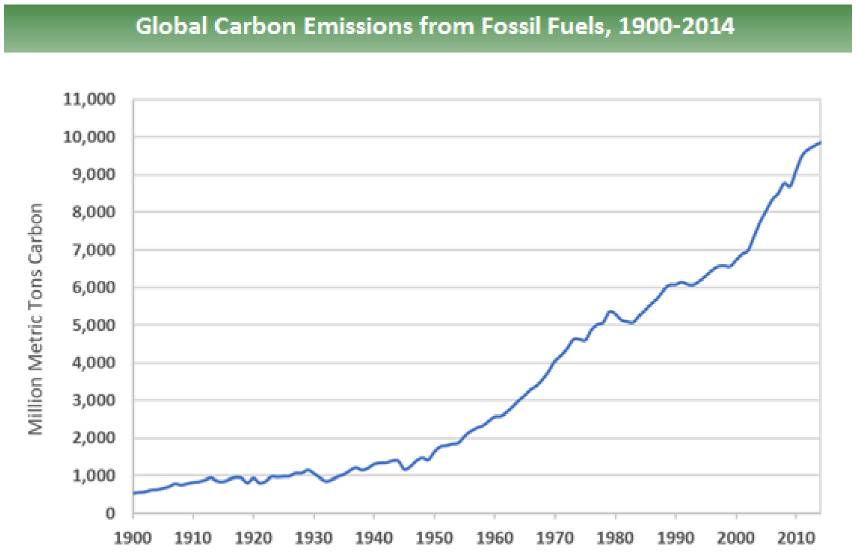


*Figure 2.2. Carbon emissions over time. Source: Boden, Marland, and Andres (2017)*

Furthermore, high altitude air monitoring with an emphasis on reducing GHG emissions began more recently. The discovery that pollutants can spread to different continents, affecting the destination's environment, and the popularization of civil aviation, increasing the level of pollution at high altitudes, have both sparked an interest in high altitude pollution monitoring. High Altitude Balloons (HAB) are of the many methods used to monitor air quality in the atmosphere. This project is focused on developing a HAB and equipping it with the electronics to collect data on harmful pollutants and factors that are affected by them.

# Chapter 3. Methodology

Being the second team to work on and carry out this project, we had a good basis of information and methodologies to work with from the previous team's documentation of their project (Jiang & Lee & Ru, 2020). From the previous team's documentation, work, and results, we wanted to make improvements pertaining to sensors and payload selection, flight progress and simulation, and cable management of the components to derive better performance and results. Our goal for the High Altitude balloon (HAB) launch was to employ a latex, helium-filled atmospheric weather balloon that travels to the edge of the outer space reaching altitudes of over 60,000 ft, gathering scientific measurements from Earth's atmosphere, such as air pressure and temperature, and most importantly greenhouse gas (GHG) (e.g. CO2, NO2, $O_3$) compositions and pollution levels, as well as capturing near-space imagery data.

## 3.1 Pre- and Post-Flight Objectives

Before the launch of the HAB, there were several objectives to complete. Like the previous team, we decided to have more than one launch, so we decided to have a first launch in mid-November, and a second launch in mid-March. We also planned on using Lincoln Park in Albany, NY as the site for both of our launches. We chose Lincoln Park as our launch site to make use of the large open field available in the park to ensure we had space for a safe launch of our HAB. The reason for wanting to launch further west was to prevent our HAB flying out into the Atlantic Ocean, since the wind current on the eastern coast flows from west to east. All the pre and post flight objectives we had have been listed in our table including all of the major pre and post flight objectives (Table 1.).

*Table 3.1. List of pre- and post-flight objectives completed by our project group during our work on this project associated with the launches of our HAB.*

| **Objectives** | **Pre-Flight** | **Post-Flight** |
|---|---|---|
| **1)** | Weigh payload. | Track payload's location using the Spot Tracker app. |
| **2)** | Simulate HAB performance and flight path. | Follow the payload to maintain real-time data communication. |
| **3)** | Retrieve the necessary amount of helium for the HAB (200 cubic feet in our case). | Maintain a path with the payload in line of sight for as long as possible. |
| **4)** | Assemble a radar reflector | Once the balloon bursts, follow the payload's descent path. |
| **5)** | Select launch site and date and call ahead to reserve the space. required | Monitor the location of the payload on the Spot Tracker app until it is no longer moving. |
| **6)** | Activate Spot Tracker. | Reach the location of the payload. |
| **7)** | Call the Federal Aviation Agency and Boston Air Traffic Control to notify them of our launch. | Retrieve payload. |
| **8)** | Test payload electronics a final time prior to launch. | |
| **9)** | Inflate the balloon and seal it. | |
| **10)** | Launch. | |

*Table 3.2. Contact information for the appropriate authorities which we contacted prior to our HAB launches.*

| Contact | Phone Number | When to Call |
|---|---|---|
| Federal Aviation Agency | 1(886) 835-5322 | A week before the projected launch date, and then fifteen minutes before the launch |
| Boston Air Traffic Control | 1(603) 594-5500 | A week before the projected launch date, and then fifteen minutes before the launch |

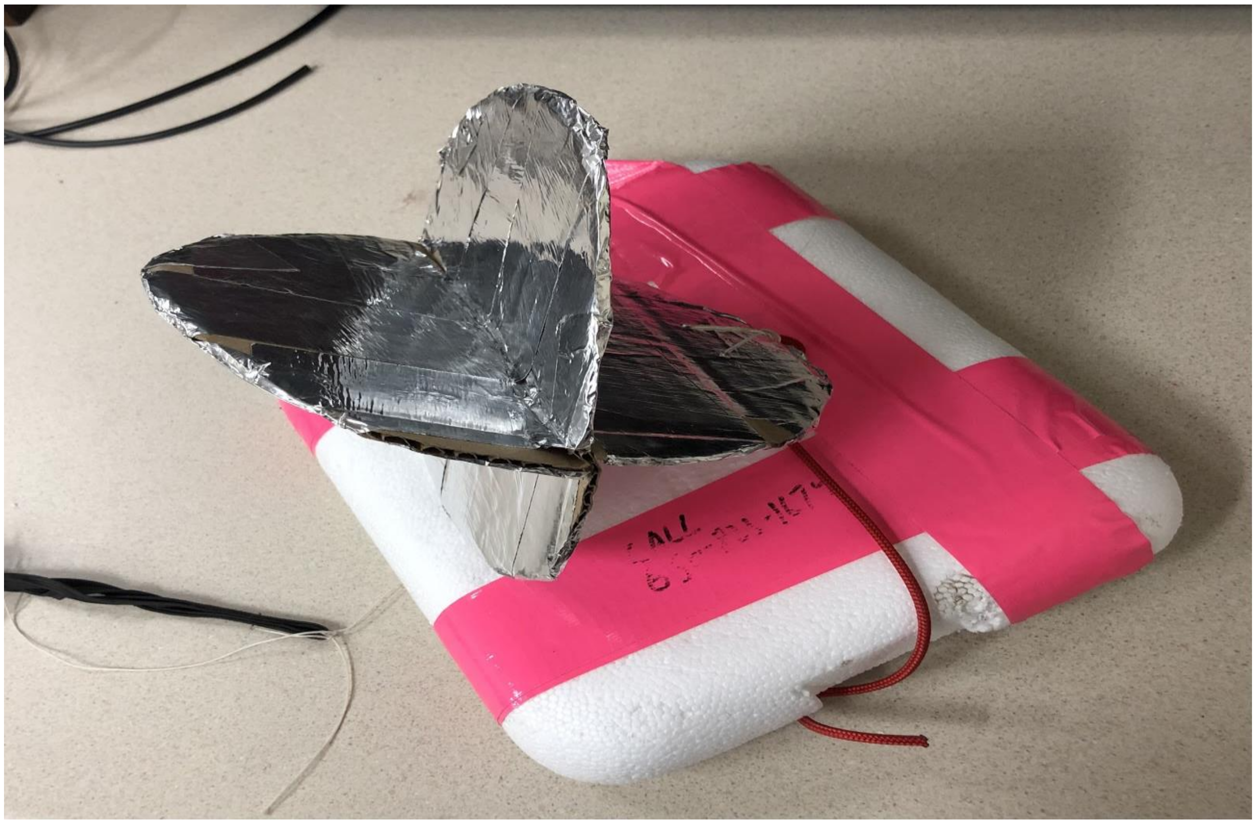
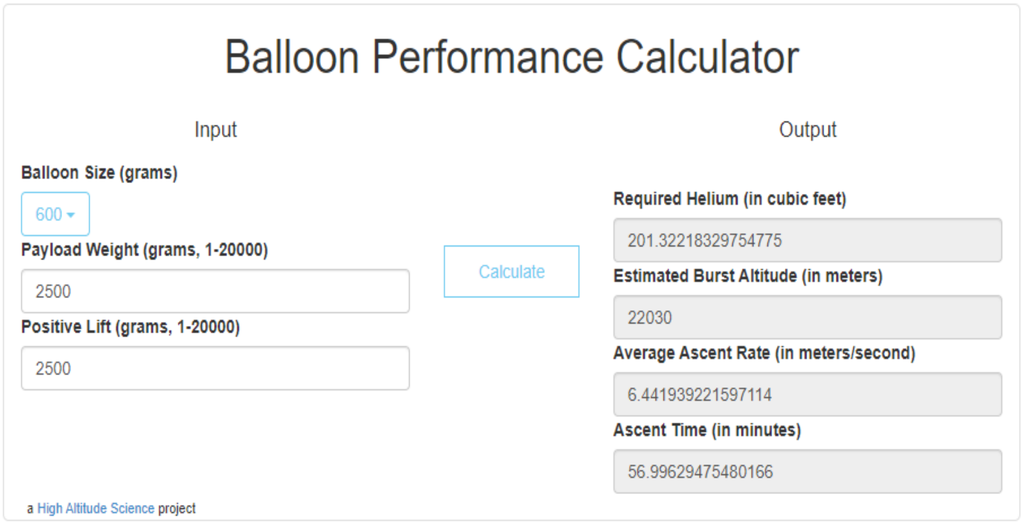


*Figure 3.1. The radar reflector which we constructed and attached to our HAB. The radar reflector was made using pieces of cardboard and aluminum tape.*

One of the most important objectives to be completed prior to each of our HAB launches was the simulation of the flight path and performance of our HAB. The performance calculator to simulate our HAB's performance was the High-Altitude Science Balloon Performance Calculator, and for simulating the HAB's flight path we used the Cambridge University Spaceflight Landing Predictor. The use of these simulation software was essential in determining a few important details associated with our HAB, including how much helium we needed for our HAB, what the flight path of our HAB would roughly look like, and when it would reach its burst altitude. To simulate the performance of our HAB, we simply had to input a few details into the balloon performance calculator - our balloon size (grams), payload's weight (grams), and positive lift (grams). To simulate our HAB's flight path using the flight path simulator we needed to input data including launch altitude, launch time and date, the HAB's ascent rate, anticipated burst altitude, and HAB's descent rate.


Balloon Performance Calculator

Input

Balloon Size (grams): 600

Payload Weight (grams, 1-20000): 2500

Positive Lift (grams, 1-20000): 2500

Output

Required Helium (in cubic feet): 201.32218329754775

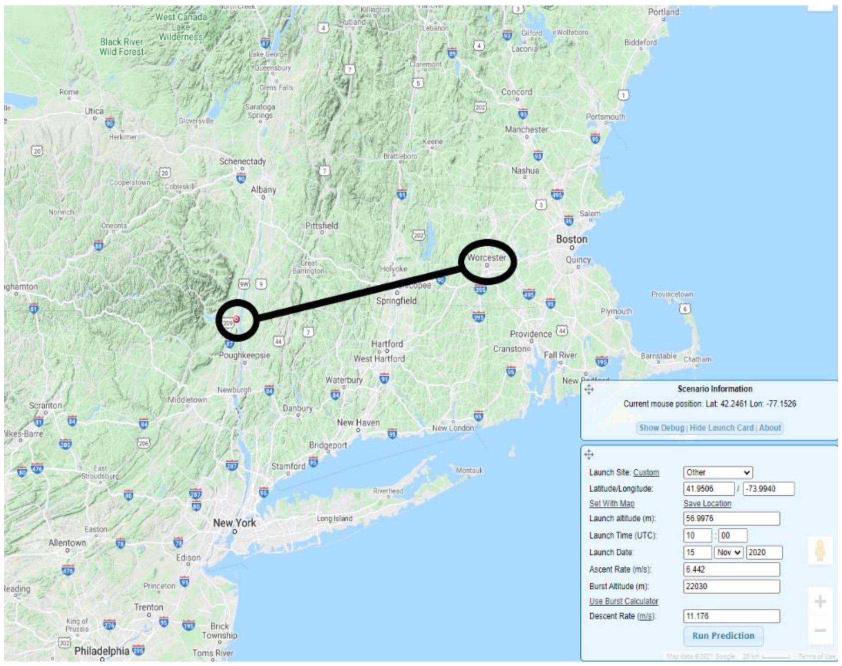
Estimated Burst Altitude (in meters): 22030

Average Ascent Rate (in meters/second): 6.441939221597114

Ascent Time (in minutes): 56.99629475480166

a High Altitude Science project

*Figure 3.2. Interface of the High-Altitude Science Balloon Performance Calculator*



Scenario Information

Current mouse position: Lat: 42.2461 Lon: -77.1526

| | |
|---|---|
| Launch Site: Custom | Other |
| Latitude/Longitude: | 41.9506 / -73.9940 |
| Launch altitude (m): | 56.9976 |
| Launch Time (UTC): | 10 : 00 |
| Launch Date: | 15 Nov 2020 |
| Ascent Rate (m/s): | 6.442 |
| Burst Altitude (m): | 22030 |
| Descent Rate (m/s): | 11.176 |

*Figure 3.3. Interface of the Cambridge University Spaceflight Landing Predictor, with Worcester, MA (right circle) and Lincoln Park in Albany, NY (left circle) focused on.*

After we successfully launched our HAB, we monitored its location using the Spot 3 Satellite GPS Messenger. The GPS tracker service we used implemented an app which allowed us to track our HAB's location through the tracker device we had inside of our payload. Tracking our HAB's flight path was relatively simple, and the location it was at was refreshed about every five minutes on the Spot tracker app. After monitoring the payload's location over the duration of the estimated flight time we had calculated using the performance calculator, we waited until we noticed that the location of the payload was no longer changing. Once the payload location was no longer changing and appeared to be consistent, we knew where the payload was located, and we travelled to this location to retrieve it.

## 3.2 Design

The High-Altitude Balloon (HAB) consisted of several components including packaging, hardware, and software. In this section we will break down our decision path to our final design. Physically, the HAB consisted of three components. The balloon, the parachute, and the payload. The balloon contained helium gas to create lift. The parachute was used to slow the payload for landing, allowing for safe recovery. And the payload contained the sensing hardware, microcontroller, computer, and the antenna.

## 3.3 Packaging

Many of the hardware components have an operating range of -20$^{o}$C to 70$^{o}$C. So, it was important that we placed all components for the payload inside of an insulated Styrofoam box. With temperatures reaching -40$^{o}$C in the stratosphere, keeping the components within operating temperatures was incredibly important. Furthermore, we stuffed the payload with packing peanuts to improve insulation as well as to prevent any disturbance within the payload, keeping the electrical components in place, especially during the initial ascent when the payload was accelerating.

## 3.4 Hardware

### 3.4.1 Electronics

#### *3.4.1.1 Sensors*

For our project, we were seeking to collect barometric and environmental data including readings of our HAB's altitude, UV, temperature, pressure readings, and GHG ($CO_2$, $NO_2$, $O_3$) concentrations. To accomplish this, we selected sensors which could gather the data we wanted to collect, interface with our Arduino Uno, and could operate over a wide range of temperatures. At an altitude of 20 km the temperature is roughly -56.5°C, so we chose sensors that would operate effectively under this specification. We also accounted for the heat generated in our payload by our microcontroller, computer, and 24V battery, as well as the insulation inside of the payload to increase the temperature our sensors would be operating in slightly as well.

*Table 3.3. List of sensors utilized in our HAB's payload.*

| **Sensor** | **Operation Range (Temp. - °C)** | **Voltage Input (V)** | **Output Type** | **Selection Reason** |
|---|---|---|---|---|
| Jtek BMP180 (Barometric, Pressure, Temp.) | -40 - 85 | 3.3 - 5 | Digital | Temperature operation range, $I^2C$ interface |
| DFROBOT Gravity V1.2 ($CO_2$) | -20 - 50 | 3.7 - 5 | Analog | Temperature operation range, simple usage, standalone from soldering board |
| GUVA-S12D (UV) | -30 - 85 | 2.7 - 5 | Analog | Temperature operation range, simple usage, supply voltage directly off from Arduino Uno |
| MiCS-2714 ($NO_2$) | -30 - 85 | 1.7 - 2.5 | Analog | Temperature operation range, simple usage, supply voltage directly off from Arduino Uno |
| Gravity I2C ($O_3$) | 20 - 50 | 3.3 - 5 | Digital | Temperature Operation range, $I^2C$ interface |

### *3.4.1.2 Supplying Power to the Sensors*

In addition to the sensors used to gather data, we had a few more electronic components in our HAB's payload. For supplying power to our sensors, we elected to use two buck converters. A buck converter, also known as a step-down converter, can take a DC input voltage and stepping it down to a lower output voltage to a load. For our sensors, we needed two buck converters to take a 12V DC voltage from our battery within the payload and step it down to supply voltage to our sensors. The BMP180, $CO_2$, UV, $NO_2$, and $O_3$ sensors essentially 5V to operate, so one buck converter would provide 5V to these sensors. The $NO_2$ sensor required essentially 2.5V to operate so the second buck converter would provide 2.5V to this sensor. To satisfy this requirement for our project, we chose to use the eBoot Mini MP1584EN DC-DC buck converter. The operating temperature of this device is -45°C - 85°C, which mostly suits the operational temperature range of this project, and because the buck converters are contained within our well-insulated payload we chose to go with this model of buck converter. This buck converter is a converter module which makes use of the MP1584EN chip and can step down input voltages between 4.5V to 28V to an output voltage of 0.8V to 20V. Inside of our payload we had a battery with a 12V DC output, and this battery would be connected to the input of both the buck converters we used. The efficiency of eBoot Mini MP1584EN is also 96% with an output voltage ripple of 30mV, so we felt confident with this converter's performance over an extended period.

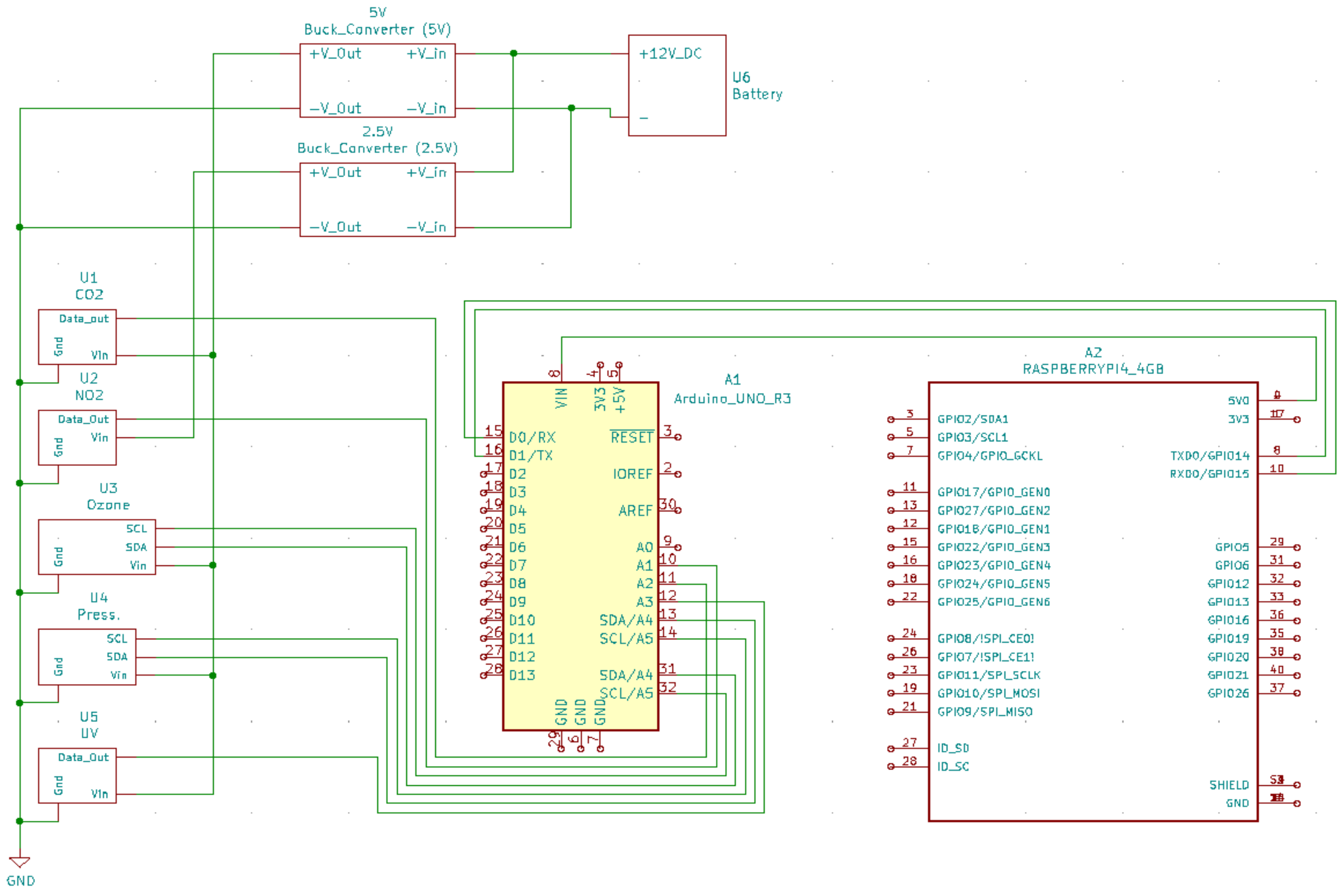


*Figure 3.4 Circuit Diagram. Also, it should be noted that the Raspberry Pi4 powers the Arduino Uno through a USB-b cable, and this has been represented in our schematic with the wired connection of Vin of the Uno to the 5VO (an output voltage port) of the Pi.*

`

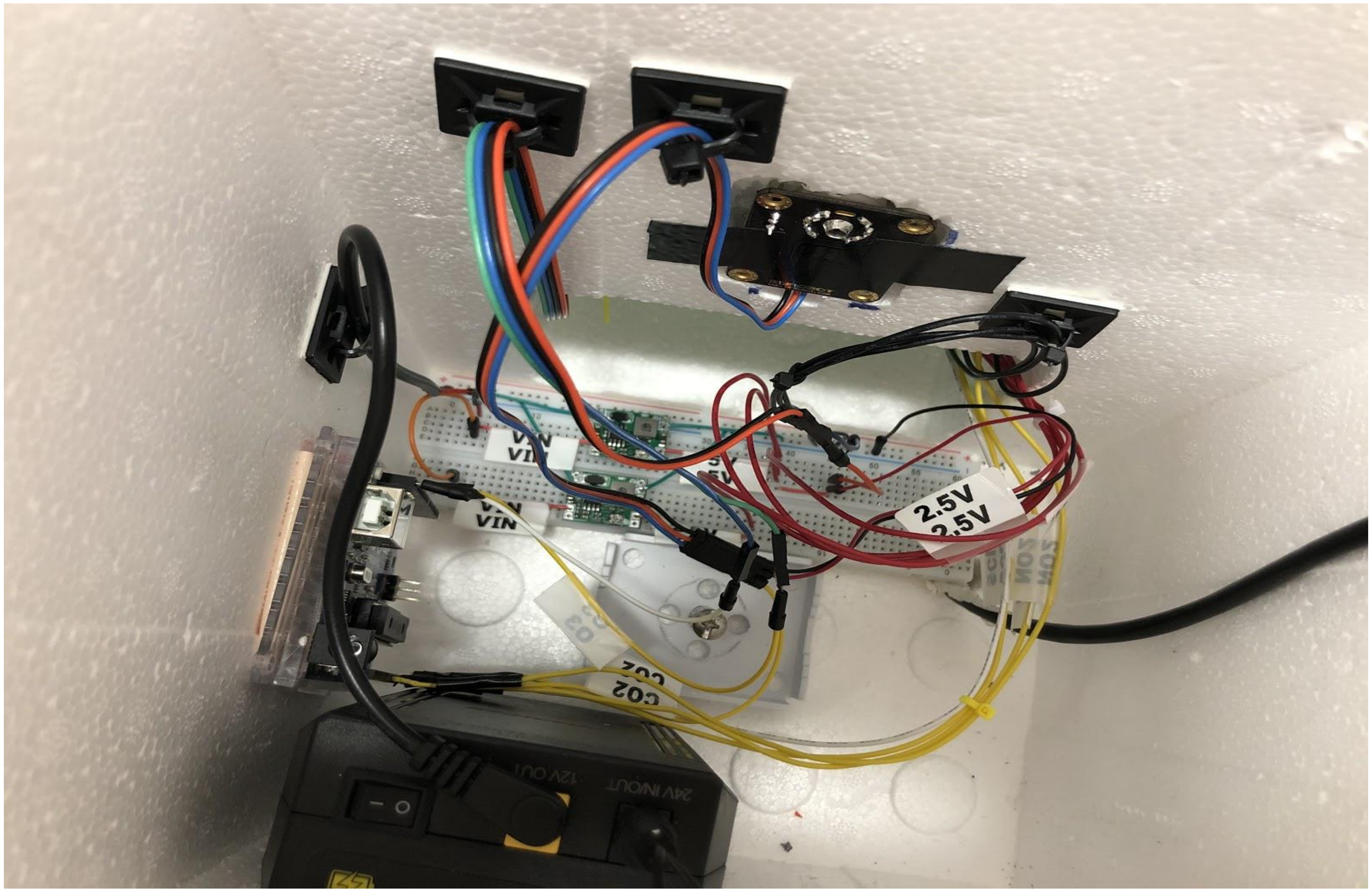


*Figure 3.5. The electronics within our payload*

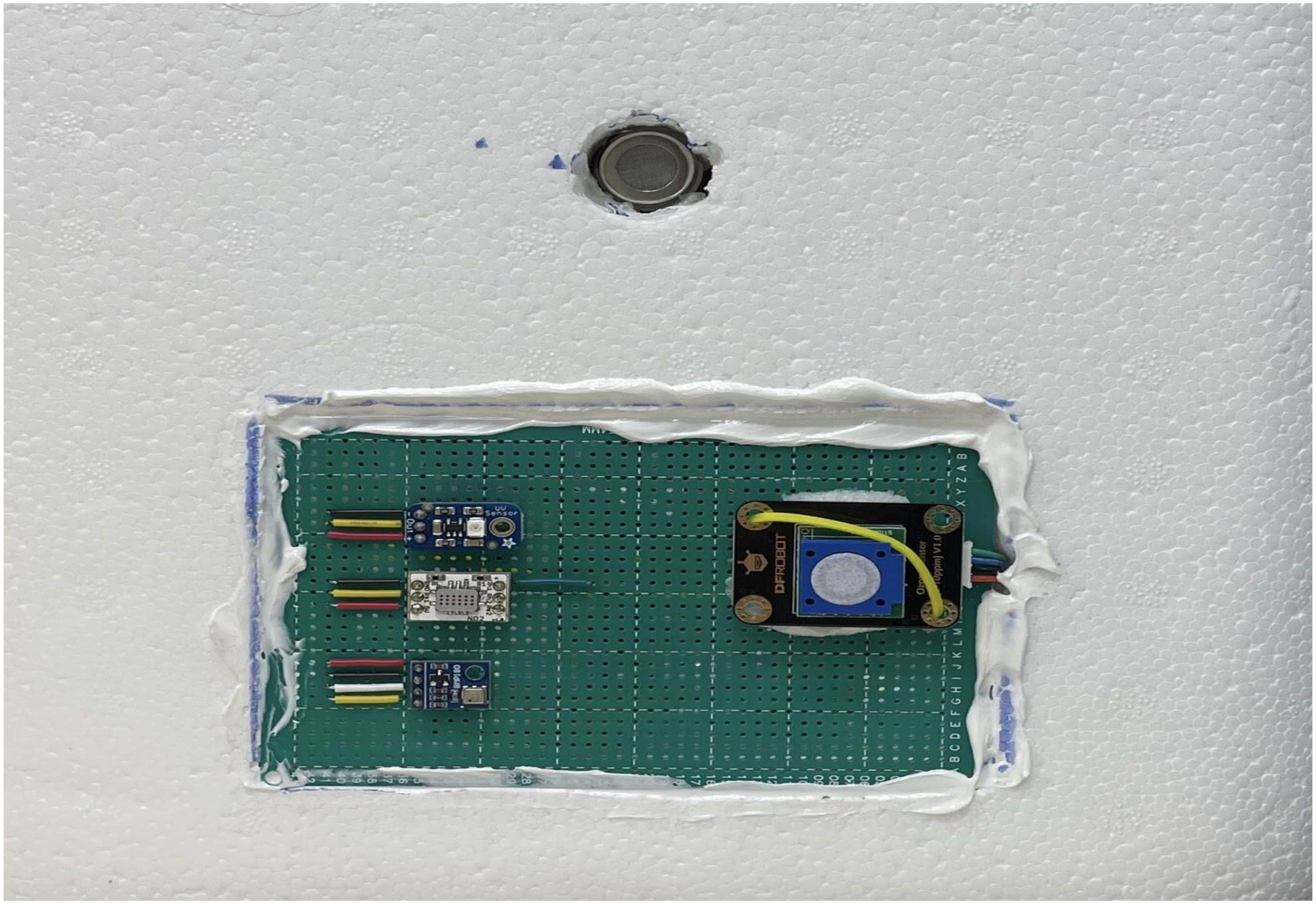


*Figure 3.6. Sensors utilized in our payload ($CO_2$, UV, $NO_2$, BMP180, and $O_3$)*

### 3.4.2 Flight Termination Unit

The flight termination unit is designed to sit on top of the payload, and when a signal is received from the Arduino, a nichrome wire will be connected to ground cutting the line to the balloon.

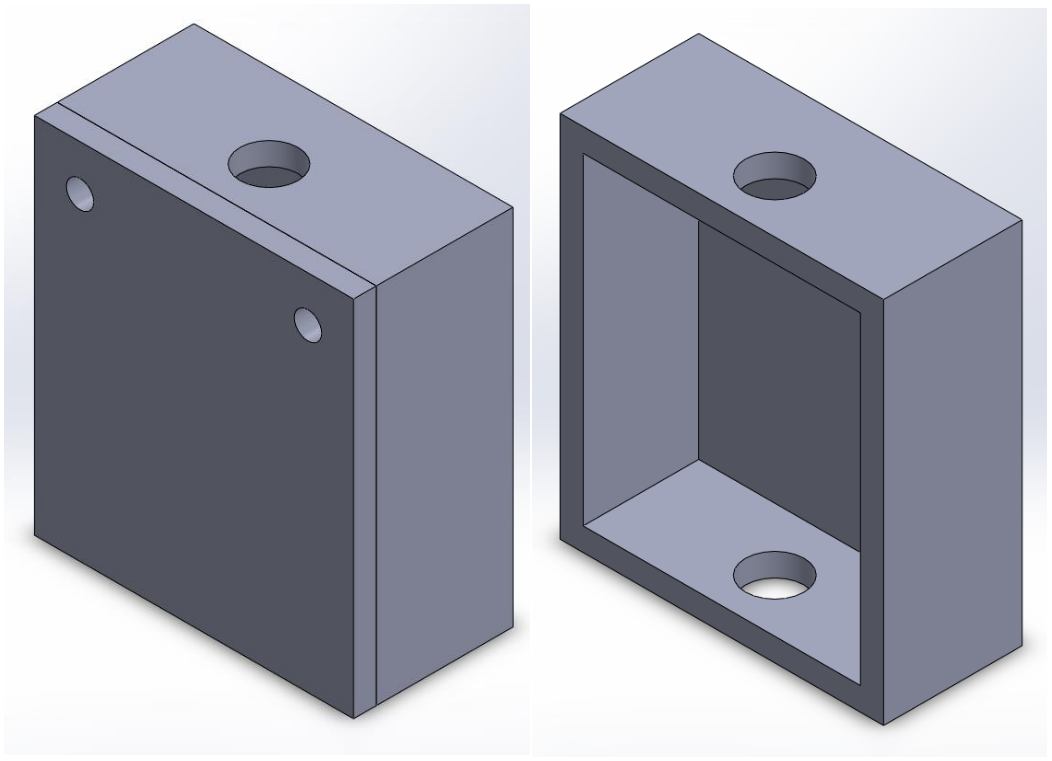


*Figure 3.7. Flight termination casing, with and without cover.*

This design was considered due to it not requiring a wireless link between the Arduino and the flight termination unit. Conventional flight termination units utilize low frequency RF communication, but this flight termination unit uses a hardwired connection between the MOSFET and the microcontroller. This can be seen in Figure 3.8, the circuit schematic.

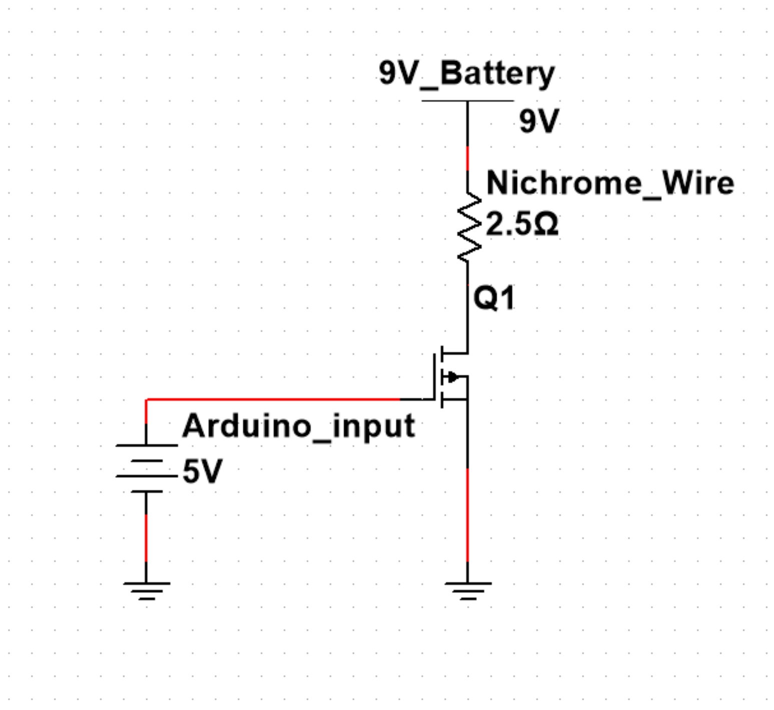


*Figure 3.8. Diagram of the flight termination circuit.*

For the flight termination unit there is little concern about creating massive current in the nichrome wire, as the end goal is to heat up the wire such that it will cut the rope connecting the balloon to the payload. Each flight the battery should be replaced such that maximum current can be achieved. The termination of flight is determined by either a certain altitude being reached (70,000 ft), or if flight time has exceeded a threshold (5 min).

### 3.4.3 Establishing a Wi-Fi Bridge

The criteria necessary for communication in this project was the ability to transmit over approximately 60,000 ft. Original consideration was given to using the cellular band. However, the range of antennas of the cellular band, specifically the consumer off the shelf models, was very low. In addition, FCC regulations make it illegal and subject to thousands of dollars in fines to modify antennas, hence, we opted for a different approach. There are three unlicensed bands that we can broadcast over, 900MHz, 2.4 GHz and 5GHz. While there are other bands available to amateurs, (the HAM band and the Amateur TV band) you need a FCC license to operate in these bands. After looking at the consumer off the shelf market, we selected a model Ubiquiti LiteBeam AC 5GHz (see Figure 11.), which is capable of 10MHz, 2.4GHz, and 5Ghz. This is also a lightweight antenna, with 16 DBi gain and weighs in at 1.2 lbs. and costs $120 (Ubiquiti Networks, 2015).

An equivalent 900MHz antenna of similar gain costs $100 but weighs 5.5lbs. With weight and price being the limiting factor, we decided to go with the Ubiquiti equipment. To go with our antenna, we needed a base station to receive signals. The base station we chose was the Ubiquiti LBE-5AC-GEN2-US since it had paired software with the Ubiquiti LiteBeam AC 5GHz (Ubiquiti Networks, 2017). The base station is pictured below in Figure 3.10.

*Figure 3.9. Ubiquiti LiteBeam AC 5GHz Antenna*

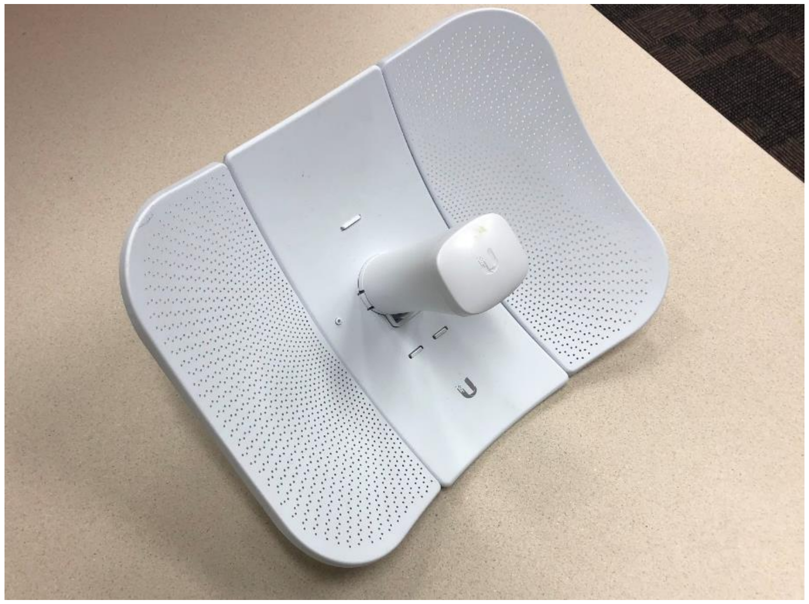


*Figure 3.10. Ubiquiti LBE-5AC-GEN2-US Base Station Dish*

## 3.5 Software

### 3.5.1 Physical Communication System

To achieve communication with the HAB during flight, we needed two computer systems: ground control station and flight computer (see Figure 3.11). The ground control station was our means of receiving data that was transmitted from the payload; this was managed by a Linux computer. The flight computer was tasked with retrieving the data from the Arduino Uno via USB. The data was then “beamed” down to the ground once it had been structured into a data packet. We used a Raspberry Pi 4 as our flight controller since it was cheap and possessed an OS.

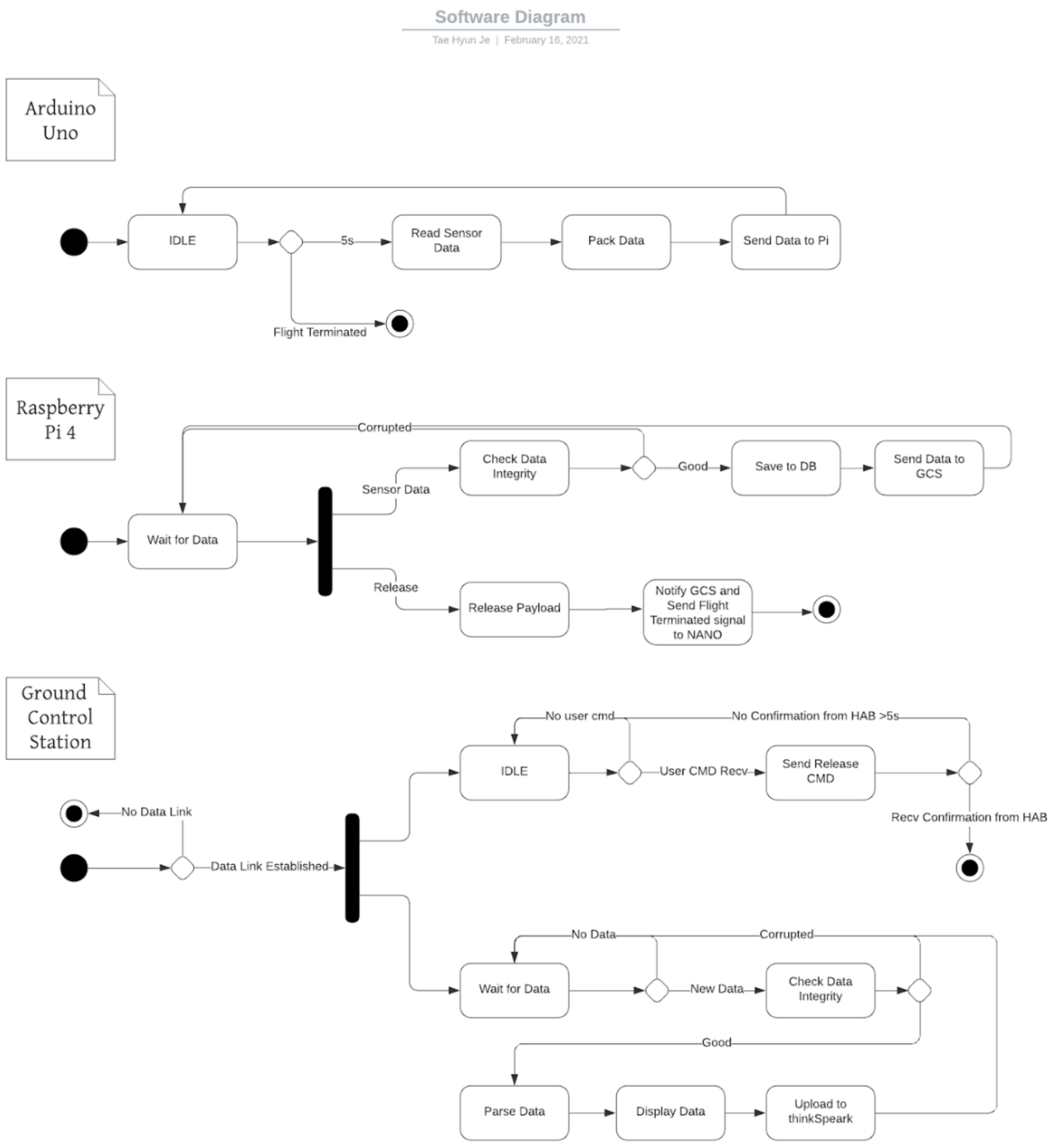
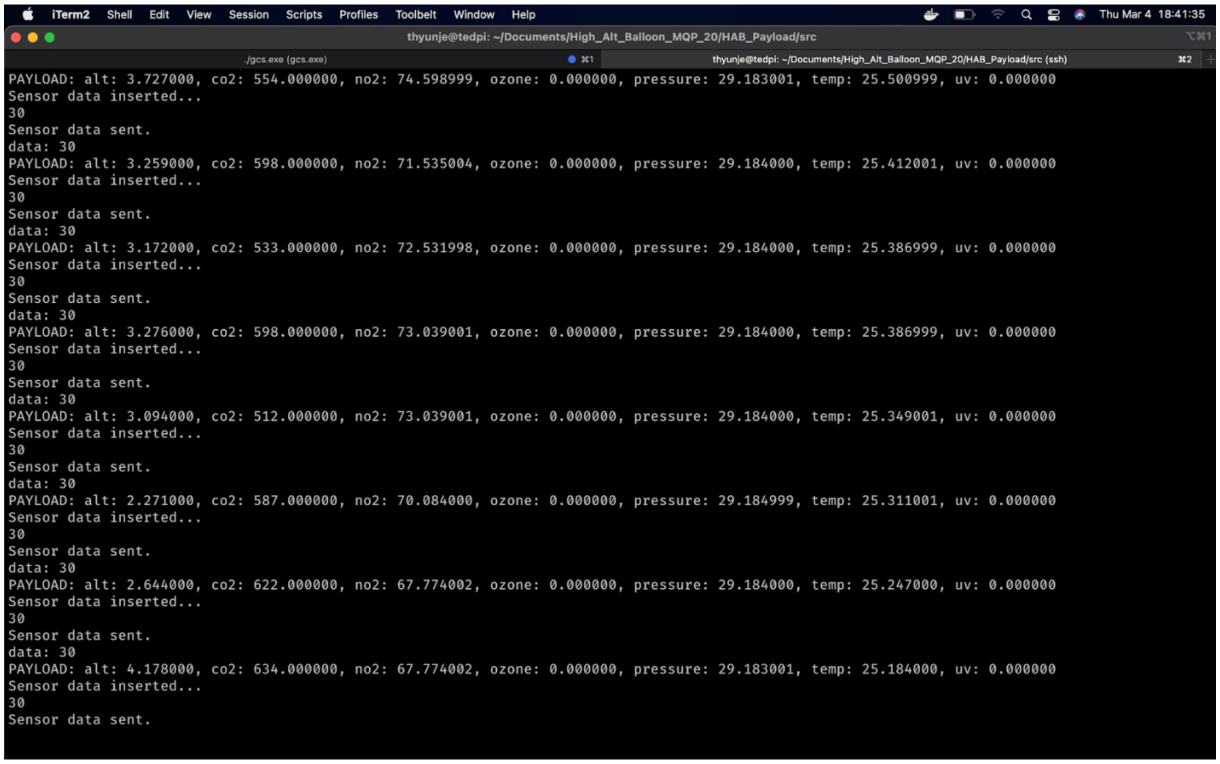


*Figure 3.11. Software activity diagram of UNO, Pi, and GCS.*

### 3.5.2 Network Communication System

Network communication was accomplished with a UDP protocol. UDP, user datagram protocol, is a fast network communication protocol that allows data packet transfer between two or more systems. While it is not as reliable as TCP since it does not guarantee the integrity of the data and lacks agreement between two systems, it is usable in our situation because we will be transmitting at a rate of 0.2 Hz, or 5 seconds; we had a multitude of sensor data. Furthermore, our ground control station was not active throughout the projected mission flight time. So, by allowing the flight controller to constantly send data back down without waiting for a connection, like TCP, it made it easier for us to simply look for new data being written to a specific port. To test this, we were able to set up a UDP server-client program with our Ubiquiti devices (see Figure 3.12).


iTerm2
Shell
Edit
View
Session
Scripts
Profiles
Toolbelt
Window
Help
Thu Mar 4 18:41:35
thyunje@tedpi: ~/Documents/High_Alt_Balloon_MQP_20/HAB_Payload/src
./gcs.exe (gcs.exe)
⌘1
thyunje@tedpi: ~/Documents/High_Alt_Balloon_MQP_20/HAB_Payload/src (ssh)
⌘2

```
PAYLOAD: alt: 3.727000, co2: 554.000000, no2: 74.598999, ozone: 0.000000, pressure: 29.183001, temp: 25.500999, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 3.259000, co2: 598.000000, no2: 71.535004, ozone: 0.000000, pressure: 29.184000, temp: 25.412001, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 3.172000, co2: 533.000000, no2: 72.531998, ozone: 0.000000, pressure: 29.184000, temp: 25.386999, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 3.276000, co2: 598.000000, no2: 73.039001, ozone: 0.000000, pressure: 29.184000, temp: 25.386999, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 3.094000, co2: 512.000000, no2: 73.039001, ozone: 0.000000, pressure: 29.184000, temp: 25.349001, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 2.271000, co2: 587.000000, no2: 70.084000, ozone: 0.000000, pressure: 29.184999, temp: 25.311001, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 2.644000, co2: 622.000000, no2: 67.774002, ozone: 0.000000, pressure: 29.184000, temp: 25.247000, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
data: 30
PAYLOAD: alt: 4.178000, co2: 634.000000, no2: 67.774002, ozone: 0.000000, pressure: 29.183001, temp: 25.184000, uv: 0.000000
Sensor data inserted...
30
Sensor data sent.
```

*Figure 3.12. Receiving test environmental data on the ground control station.*

### 3.5.3 Data Infrastructure

Our data infrastructure relies on custom made data packets with a one’s complement checksum and an SQLite database. The data packets possessed our sensor data and the checksum was used to check the integrity of the data once it had been received by the ground control station. To save the data, we used an SQLite database manager, both on the flight controller and the ground control station. The language used for this is C.

Our data packet consists of utilizing byte guaranteed types and 3 different headers (see Figure 3.13). One of C's built-in libraries is stdint.h. We used it to access variable types such as uint8_t (8 bits) and uint32_t (32 bits). The compiler will guarantee that the size is 8 bits and 32 bits, respectively. When it comes to bit manipulation and packing it is safer to use variables like these, rather than an int type that can be 2 or 4 bytes depending on the size of the data. The headers for the data packet are payload type, payload, and checksum. The payload type was used to define what the payload of the data packet possesses, sensor or release data. The release data simply tells the flight controller that it is time to terminate the flight and the sensor data is self-explanatory: our recent launch did not test the flight termination functionality. There are seven types of data that we recorded: pressure, $NO_2$, temperature, UV, $CO_2$, Ozone, and altitude, all of which are 32 bits. They were initially collected into a float variable type and then multiplied by a factor of ten to the fourth. This made it easier to store into the uint32_t variable and break it into bytes for the payload section. To guarantee that no bit was flipped/lost, the last byte in the payload possessed a checksum value, which was used by the ground control station.

We used an SQL based database. The reason why we chose to use this rather than writing to a text file is because we can manipulate/retrieve specific data. We can retrieve data at a certain altitude and above, for example. This will make it easier for us to not only process the data into graphs, but to deeply analyze greenhouse gas concentration levels.

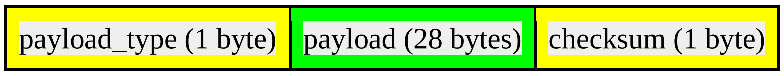
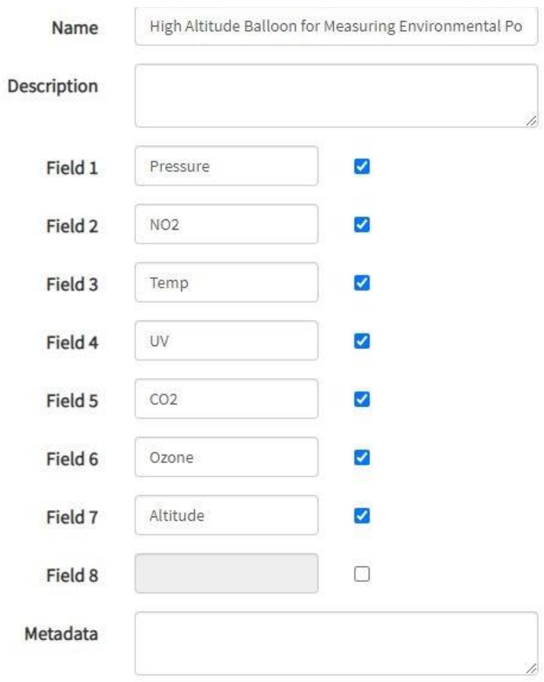


*Figure 3.13. Data packet header.*

### 3.5.4 Real Time Data Graph

One of the main goals of this project is to graph data in real time. To achieve this, our team decided to use Mathworks' ThingSpeak API. ThingSpeak is an open source IoT application that allows users to upload data for real time graphing. For our project, we set up seven different graphs for our sensor data (see Figure 3.14). This allowed use to check each sensor data uniquely.

The ThingSpeak functionality of this project was written only on the ground control station. Once data had been received, we were able to use ThingSpeak's API to upload data. To establish connection, we used one of our teammates' hotspot on their phone. When ThingSpeak successfully obtained our data, the graphs were updated accordingly (see Figure 3.15). However, because we were using a free account, we were not able to upload data every 5 seconds. An estimated 5-10 seconds was required between uploads. However, because we had the data stored locally, it did not really impact our goal.



| | | |
|---|---|---|
| Name | High Altitude Balloon for Measuring Environmental Po | |
| Description | | |
| Field 1 | Pressure | ✓ |
| Field 2 | NO2 | ✓ |
| Field 3 | Temp | ✓ |
| Field 4 | UV | ✓ |
| Field 5 | CO2 | ✓ |
| Field 6 | Ozone | ✓ |
| Field 7 | Altitude | ✓ |
| Field 8 | | ☐ |
| Metadata | | |

*Figure 3.14. Channel settings for ThingSpeak.*

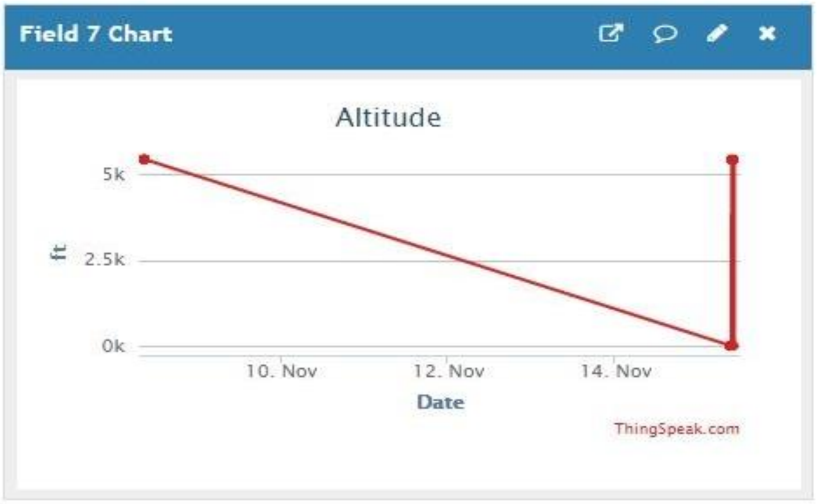


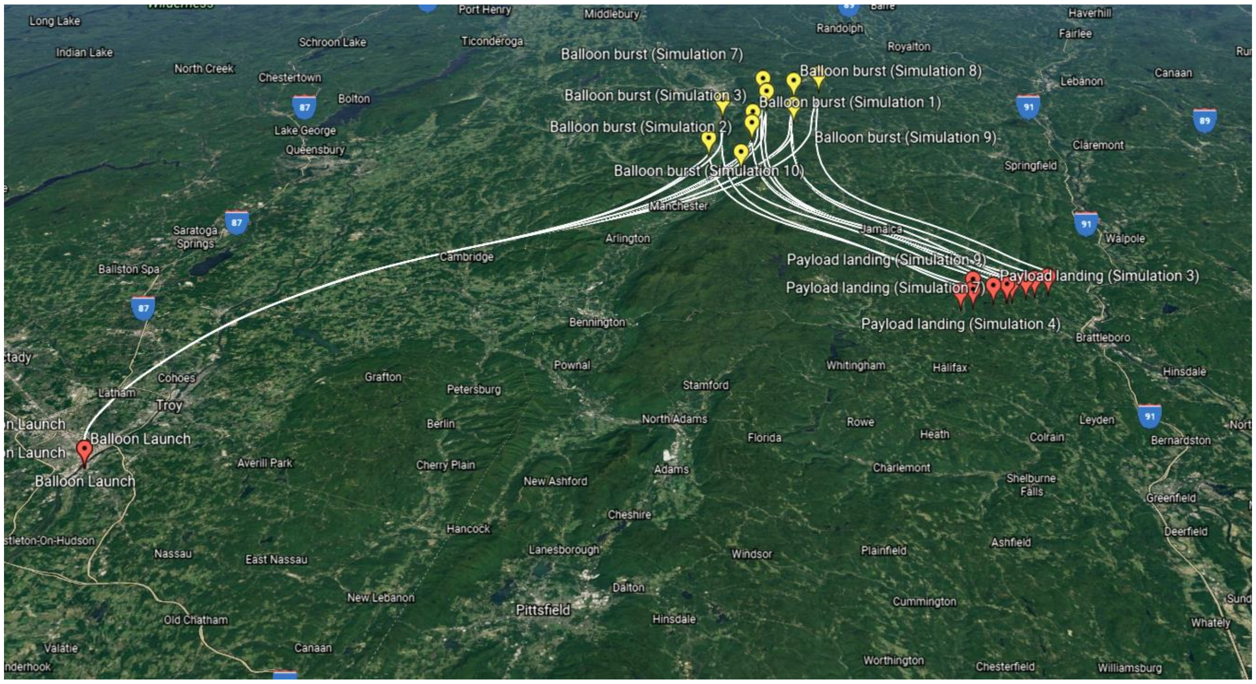
*Figure 3.15. ThingSpeak chart with test data.*

# Chapter 4. Results

This section details the results and findings of our first and final launch. We provide the results of our first two launches and their outcomes and the sensor data obtained.

## 4.1 First Launch

Our first launch took place on November 15, 2020. Our simulation projected a path northeast from our launch site (see Figure 4.1).



Balloon burst (Simulation 7)
Balloon burst (Simulation 8)
Balloon burst (Simulation 3)
Balloon burst (Simulation 1)
Balloon burst (Simulation 2)
Balloon burst (Simulation 9)
Balloon burst (Simulation 10)
Payload landing (Simulation 9)
Payload landing (Simulation 3)
Payload landing (Simulation 7)
Payload landing (Simulation 4)
Balloon Launch
Balloon Launch

*Figure 4.1. Expected flight path, starting in Lincoln Park, NY , and landing in Putney, NH.*

Flight path is dependent on the nozzle lift and wind conditions present. When we launched the balloon, we experienced a different flight path. This can be seen in Figure 4.2. We were able to obtain a real time path during our launch with our spot tracker.

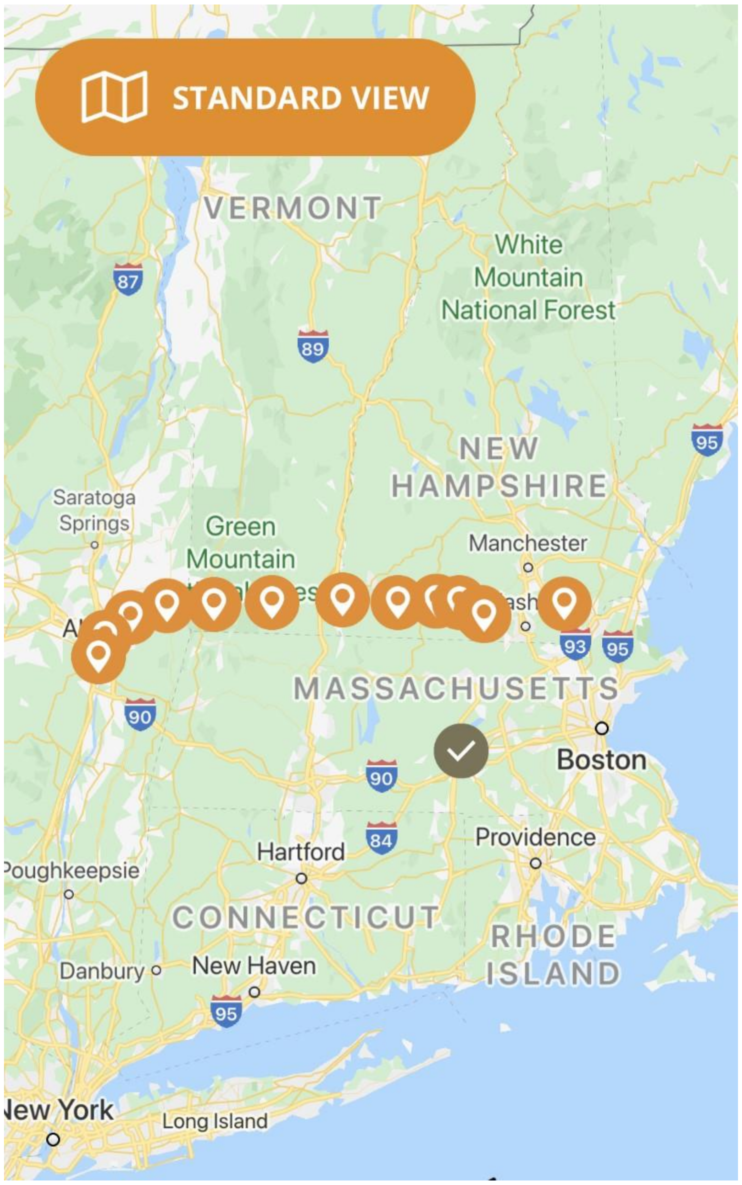


*Figure 4.2. Actual recorded flight path from GPS, started in Lincoln Park, Albany NY, landed in Derry, NH.*

Nozzle lift is dependent on the overall weight of the payload and the amount of helium in the balloon. When preparing for this launch, the payload weighed more than expected, around 2.5 kg. This resulted in a lower nozzle lift, meaning more time in the air and more distance traveled. Fortunately, the HAB did not land in the ocean, but in the yard of a New Hampshire resident.

When the payload was recovered, it had suffered damages from the landing, and it had been forced open by the resident (see Figure 4.3). However, all components were recovered from the crash and verified to insure working condition.

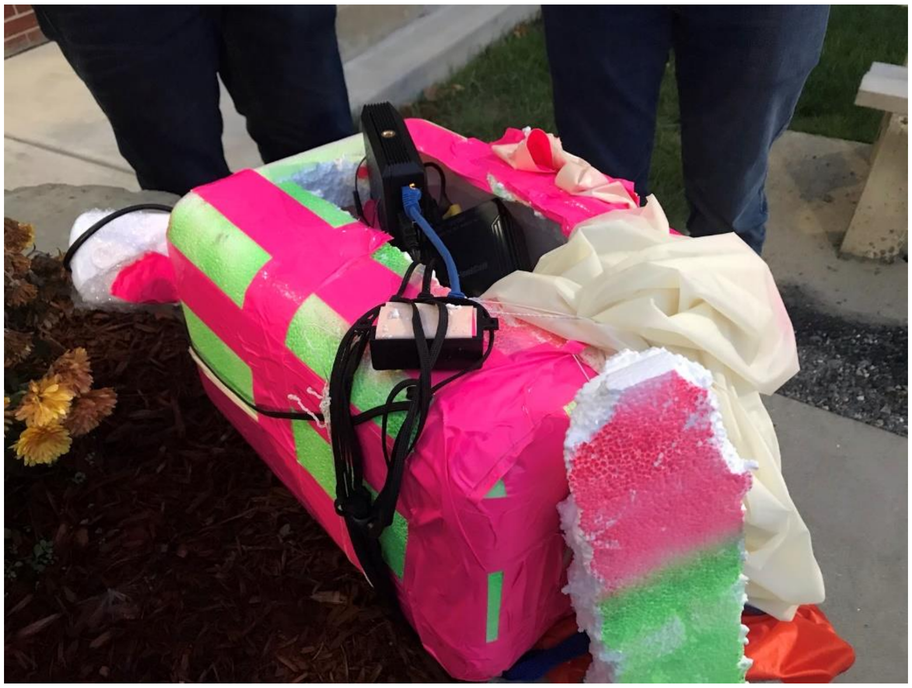
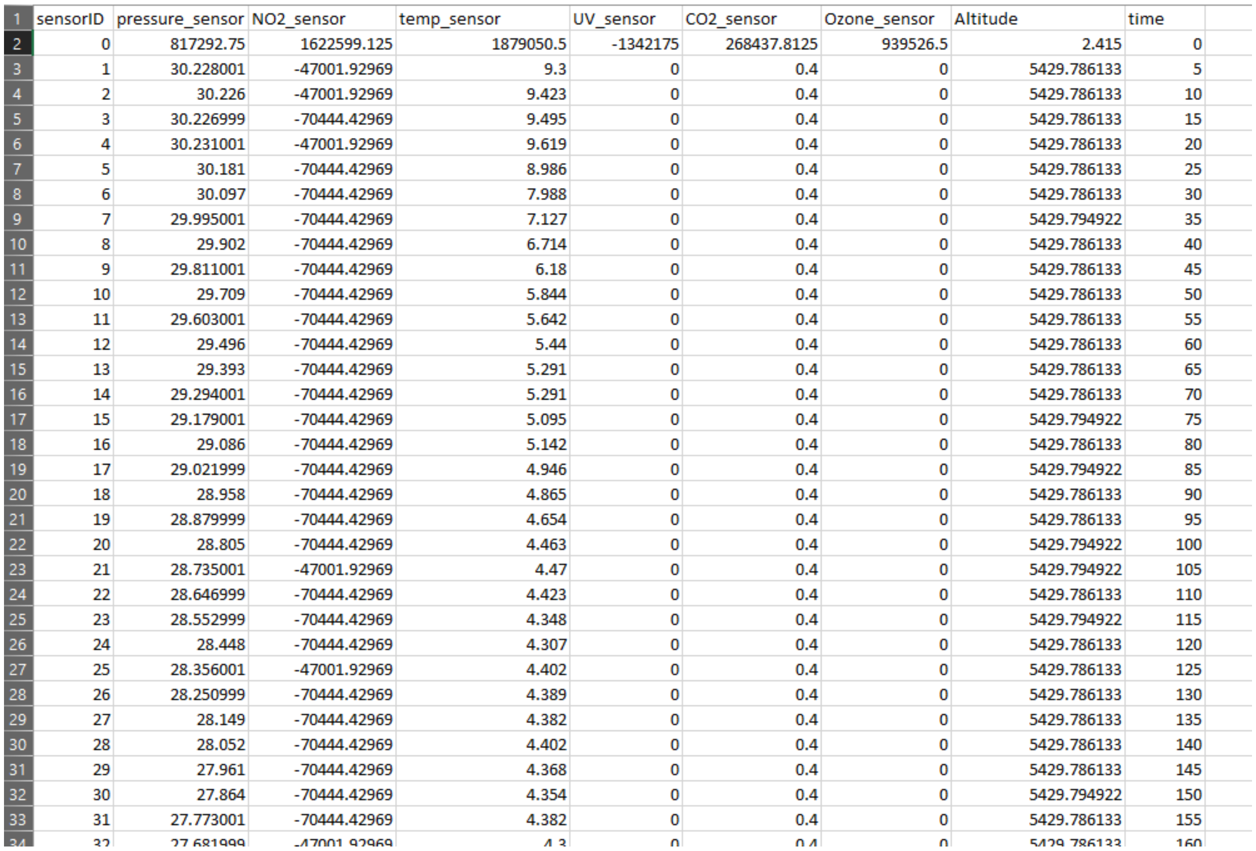


*Figure 4.3. The recovered payload after the first launch.*

## 4.2 First Launch Results

Our first launch lasted approximately two hours. Real-time data collection was only active during the first 5 minutes of the launch. Past this point, our GCS continually attempted to reconnect, but failed to do so. However, during its flight time, we believed that our sensors were collecting the desired data. After we retrieved the payload, we realized that our data was mostly incorrect. Out of the six sensors, only two of them reported the correct data, while the rest either reported incorrect and inaccurate data or appeared to not have been working during launch (see Figure 4.4). Furthermore, we noticed that we only collected about 17 minutes of sensor data.



| 1 | sensorID | pressure_sensor | NO2_sensor | temp_sensor | UV_sensor | CO2_sensor | Ozone_sensor | Altitude | time |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 0 | 817292.75 | 1622599.125 | 1879050.5 | -1342175 | 268437.8125 | 939526.5 | 2.415 | 0 |
| 3 | 1 | 30.228001 | -47001.92969 | 9.3 | 0 | 0.4 | 0 | 5429.786133 | 5 |
| 4 | 2 | 30.226 | -47001.92969 | 9.423 | 0 | 0.4 | 0 | 5429.786133 | 10 |
| 5 | 3 | 30.226999 | -70444.42969 | 9.495 | 0 | 0.4 | 0 | 5429.786133 | 15 |
| 6 | 4 | 30.231001 | -47001.92969 | 9.619 | 0 | 0.4 | 0 | 5429.786133 | 20 |
| 7 | 5 | 30.181 | -70444.42969 | 8.986 | 0 | 0.4 | 0 | 5429.786133 | 25 |
| 8 | 6 | 30.097 | -70444.42969 | 7.988 | 0 | 0.4 | 0 | 5429.786133 | 30 |
| 9 | 7 | 29.995001 | -70444.42969 | 7.127 | 0 | 0.4 | 0 | 5429.794922 | 35 |
| 10 | 8 | 29.902 | -70444.42969 | 6.714 | 0 | 0.4 | 0 | 5429.786133 | 40 |
| 11 | 9 | 29.811001 | -70444.42969 | 6.18 | 0 | 0.4 | 0 | 5429.786133 | 45 |
| 12 | 10 | 29.709 | -70444.42969 | 5.844 | 0 | 0.4 | 0 | 5429.786133 | 50 |
| 13 | 11 | 29.603001 | -70444.42969 | 5.642 | 0 | 0.4 | 0 | 5429.786133 | 55 |
| 14 | 12 | 29.496 | -70444.42969 | 5.44 | 0 | 0.4 | 0 | 5429.786133 | 60 |
| 15 | 13 | 29.393 | -70444.42969 | 5.291 | 0 | 0.4 | 0 | 5429.786133 | 65 |
| 16 | 14 | 29.294001 | -70444.42969 | 5.291 | 0 | 0.4 | 0 | 5429.786133 | 70 |
| 17 | 15 | 29.179001 | -70444.42969 | 5.095 | 0 | 0.4 | 0 | 5429.794922 | 75 |
| 18 | 16 | 29.086 | -70444.42969 | 5.142 | 0 | 0.4 | 0 | 5429.786133 | 80 |
| 19 | 17 | 29.021999 | -70444.42969 | 4.946 | 0 | 0.4 | 0 | 5429.794922 | 85 |
| 20 | 18 | 28.958 | -70444.42969 | 4.865 | 0 | 0.4 | 0 | 5429.786133 | 90 |
| 21 | 19 | 28.879999 | -70444.42969 | 4.654 | 0 | 0.4 | 0 | 5429.786133 | 95 |
| 22 | 20 | 28.805 | -70444.42969 | 4.463 | 0 | 0.4 | 0 | 5429.794922 | 100 |
| 23 | 21 | 28.735001 | -47001.92969 | 4.47 | 0 | 0.4 | 0 | 5429.794922 | 105 |
| 24 | 22 | 28.646999 | -70444.42969 | 4.423 | 0 | 0.4 | 0 | 5429.786133 | 110 |
| 25 | 23 | 28.552999 | -70444.42969 | 4.348 | 0 | 0.4 | 0 | 5429.794922 | 115 |
| 26 | 24 | 28.448 | -70444.42969 | 4.307 | 0 | 0.4 | 0 | 5429.786133 | 120 |
| 27 | 25 | 28.356001 | -47001.92969 | 4.402 | 0 | 0.4 | 0 | 5429.786133 | 125 |
| 28 | 26 | 28.250999 | -70444.42969 | 4.389 | 0 | 0.4 | 0 | 5429.786133 | 130 |
| 29 | 27 | 28.149 | -70444.42969 | 4.382 | 0 | 0.4 | 0 | 5429.786133 | 135 |
| 30 | 28 | 28.052 | -70444.42969 | 4.402 | 0 | 0.4 | 0 | 5429.786133 | 140 |
| 31 | 29 | 27.961 | -70444.42969 | 4.368 | 0 | 0.4 | 0 | 5429.786133 | 145 |
| 32 | 30 | 27.864 | -70444.42969 | 4.354 | 0 | 0.4 | 0 | 5429.794922 | 150 |
| 33 | 31 | 27.773001 | -70444.42969 | 4.382 | 0 | 0.4 | 0 | 5429.786133 | 155 |
| 34 | 32 | 27.681999 | -47001.92969 | 4.3 | 0 | 0.4 | 0 | 5429.786133 | 160 |

*Figure 4.4. Initial readings from the first launch.*

While our Ubiquiti equipment can communicate at 130,000 ft when configured into 10 MHz, it fails when the alignment of the dish and antenna do not meet. When testing this equipment, we failed to consider this, resulting in no real-time data after the 5-minute mark in our first launch.

The only sensor data that we correctly obtained was temperature and pressure. The results were indicative of the research we had done prior to our launch (see Background). Furthermore, we can assume that these readings are correct because our temperature readings began to drop as the balloon climbed higher into the atmosphere. Also, pressure readings began to dip as well, which occurs when reaching higher altitudes (see Figure 4.5 & Figure 4.6). As for the other sensors, we believed that poor insulation and/or loose wires affected the system during its ascent or when we were packing the payload with packing peanuts. Our next launch took these potential issues into consideration, which yielded better results.

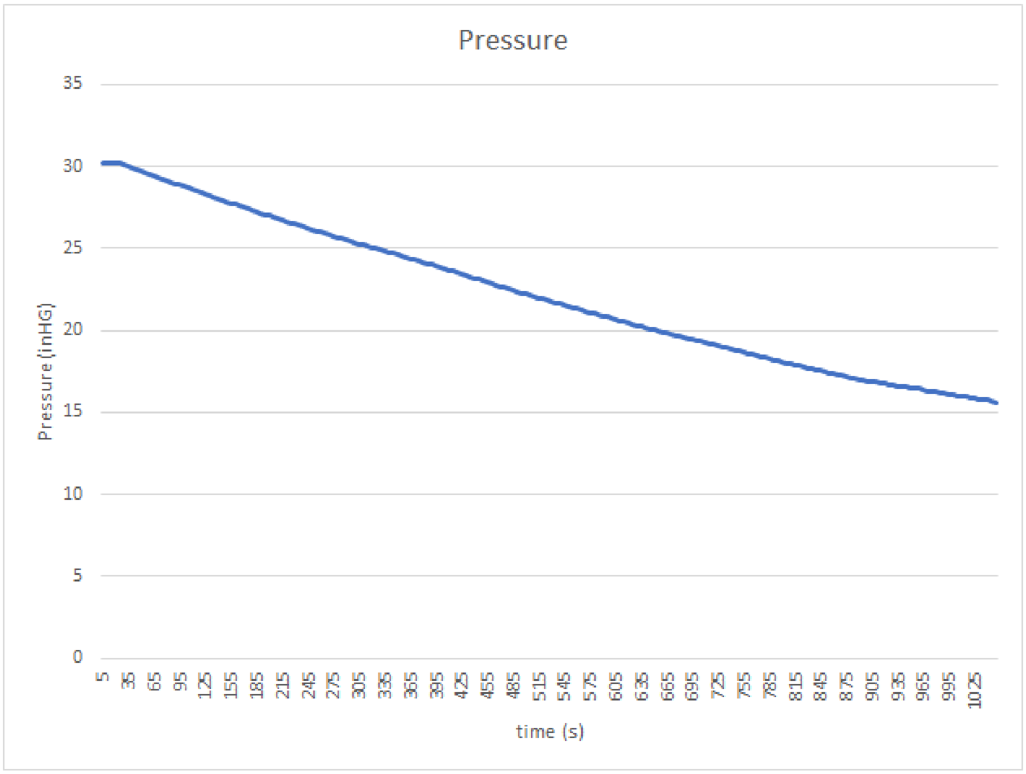


*Figure 4.5. Pressure data from the first launch, Pressure (inHG) vs Time (s).*

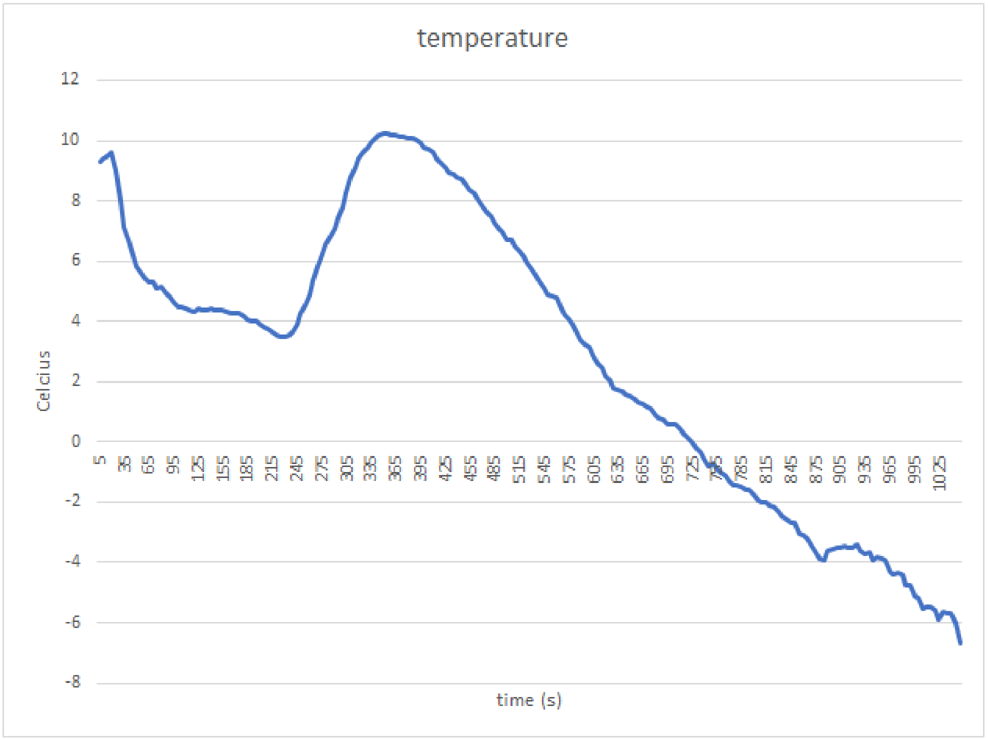


*Figure 4.6. Temperature data from the first launch, Temperature (Celsius) vs Time (s).*

Based on the number of sample points, which is every 5 seconds, the software only lasted about 17 minutes. We concluded that once we lost the payload, which resulted in the termination of our SSH connection, it terminated the program since the user was automatically logged out by the OS. A simple solution to this was using a program called "screen". This allowed us to keep a running process, our program, alive by detaching it from the main interface.

## 4.3 Second Launch

Our second launch was on March 6th, 2021 in Lincoln Park, NY, repeating the same launch site as previous launches. We simulated the flight path using the ASTRA simulation tool in the week leading up to the launch date. Figure 4.7 details the 3D flight path generated by the tool.

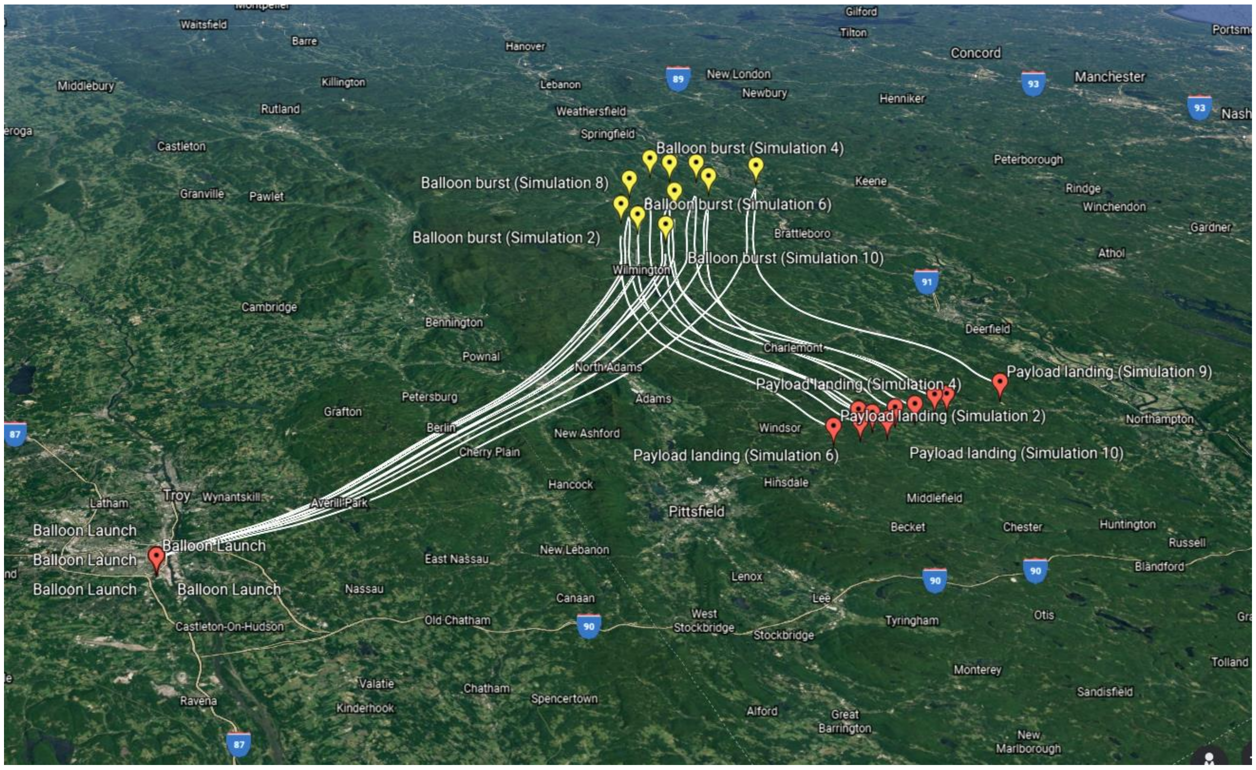

Balloon burst (Simulation 4)
Balloon burst (Simulation 8)
Balloon burst (Simulation 6)
Balloon burst (Simulation 2)
Balloon burst (Simulation 10)
Payload landing (Simulation 9)
Payload landing (Simulation 4)
Payload landing (Simulation 2)
Payload landing (Simulation 6)
Payload landing (Simulation 10)
Balloon Launch
Balloon Launch
Balloon Launch
Balloon Launch
Balloon Launch
Pittsfield
Troy

*Figure 4.7. Expected flight path, starting in Lincoln Park, NY, landing in Cummington, MA.*

As before, the flight path is dependent upon weather conditions, weight of the payload, and nozzle lift. The actual flight path differed, as the payload traveled more northeastern than expected. This can be seen in Figure 4.8, showing the launch from Albany, NY, and landing in Orange, MA.

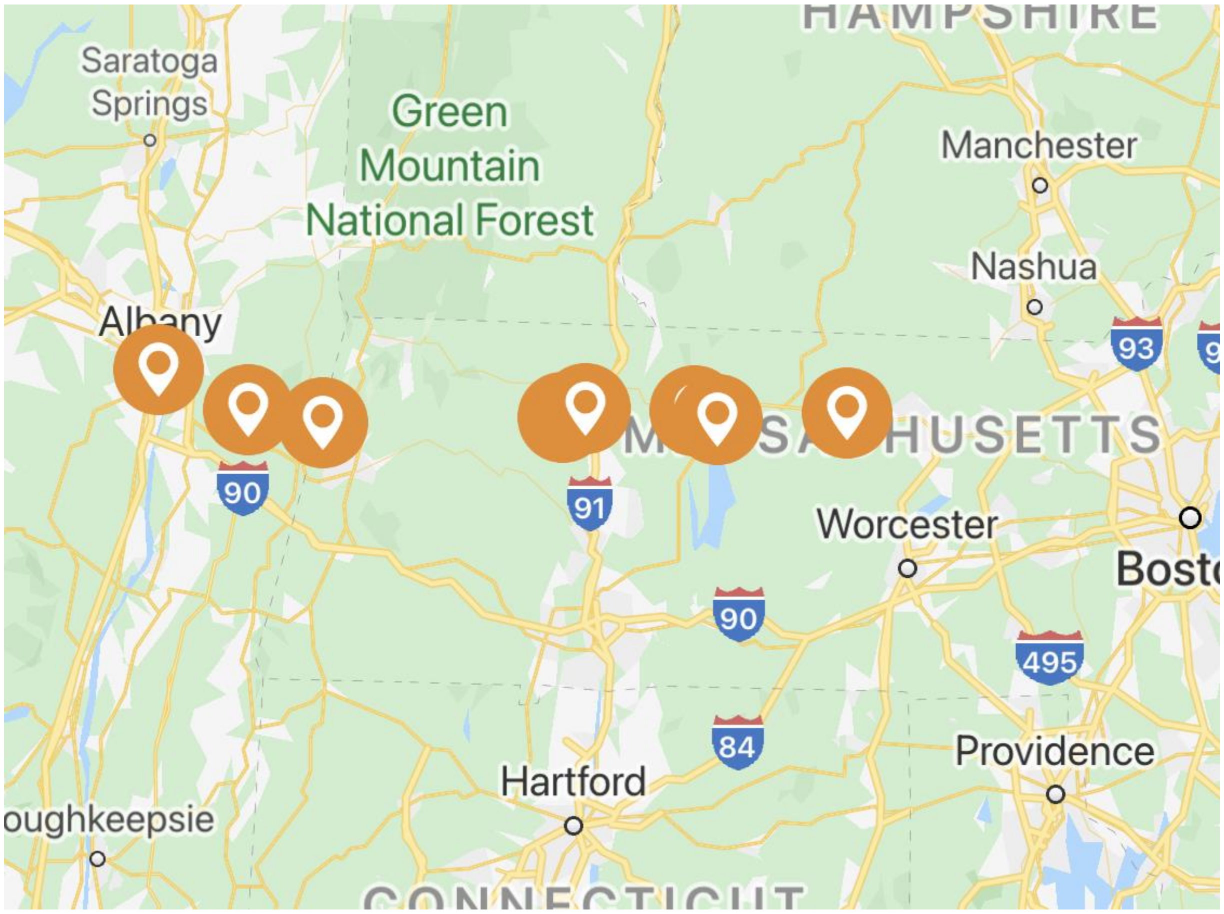


*Figure 4.8. Actual flight path, starting in Lincoln Park, NY, landing in Orange, MA.*

## 4.4 Second Launch Results

The second launch proved to be more successful than the first launch. Unlike before, all of our sensors were in operating condition during the flight (see Figure 4.9 - Figure 4.15). They reported data from our initial altitude up to about 16,700 ft; errors seen on the figures are explained in the next section. Furthermore, our software ran throughout the entirety of the flight, collecting data from start to finish. However, we were still not able to fully achieve real-time data. Data was collected for the first 5-10 minutes of the flight before losing connection. We achieved this by breaking our team into two groups: one to launch the balloon and another to follow the balloon’s projected flight trajectory via SPOT Gen 3. While the results we obtained fell short of our goals, it was still a major improvement from our first launch.

Figures 4.9 - 4.15 are the sensor data that we obtained during the flight. It demonstrates how our system was functional up to 16,700 ft. After our payload ascended beyond that point, our data flat lined until it started descending.

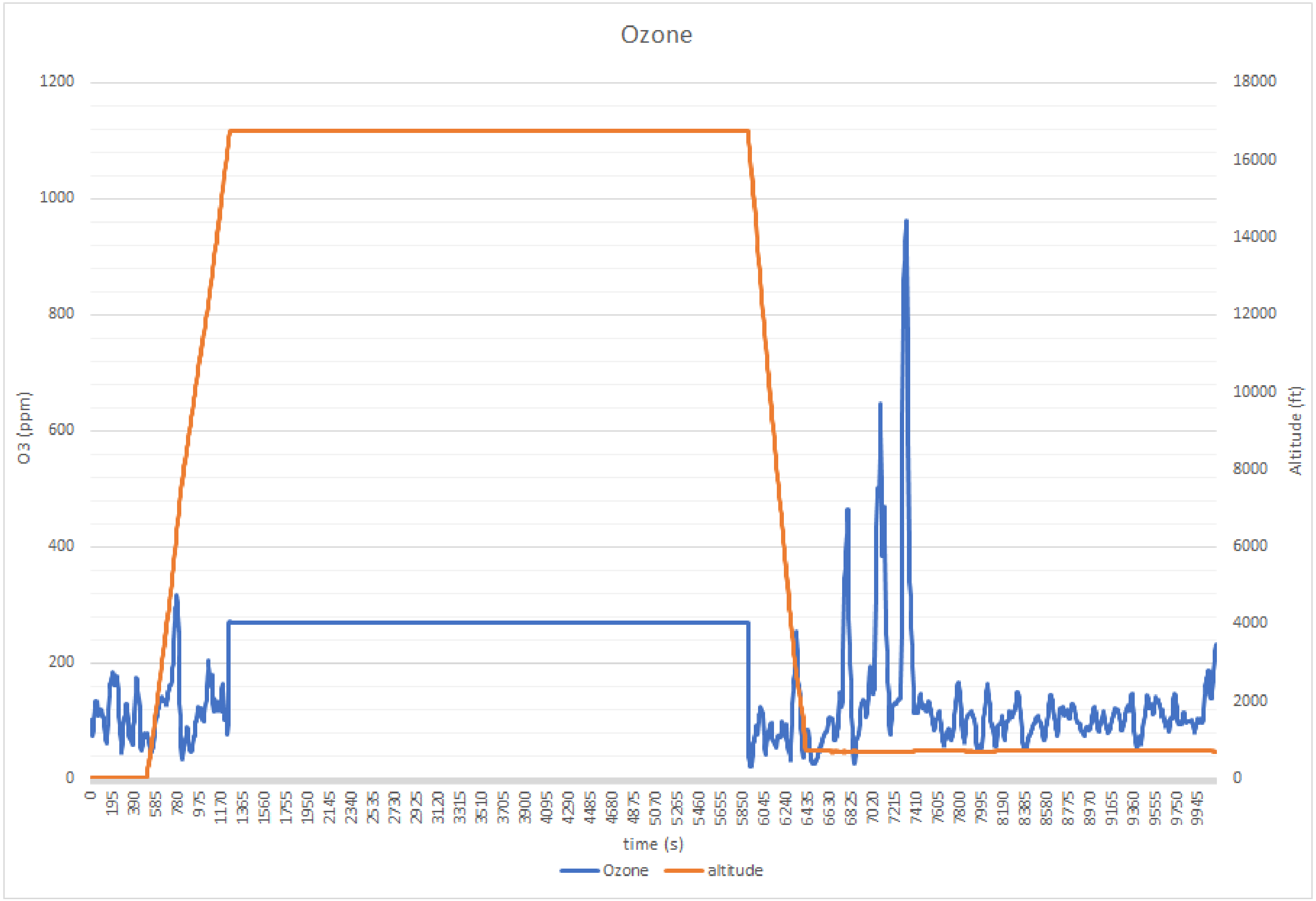


*Figure 4.9. Ozone (ppm), Altitude (ft) vs Time (s).*

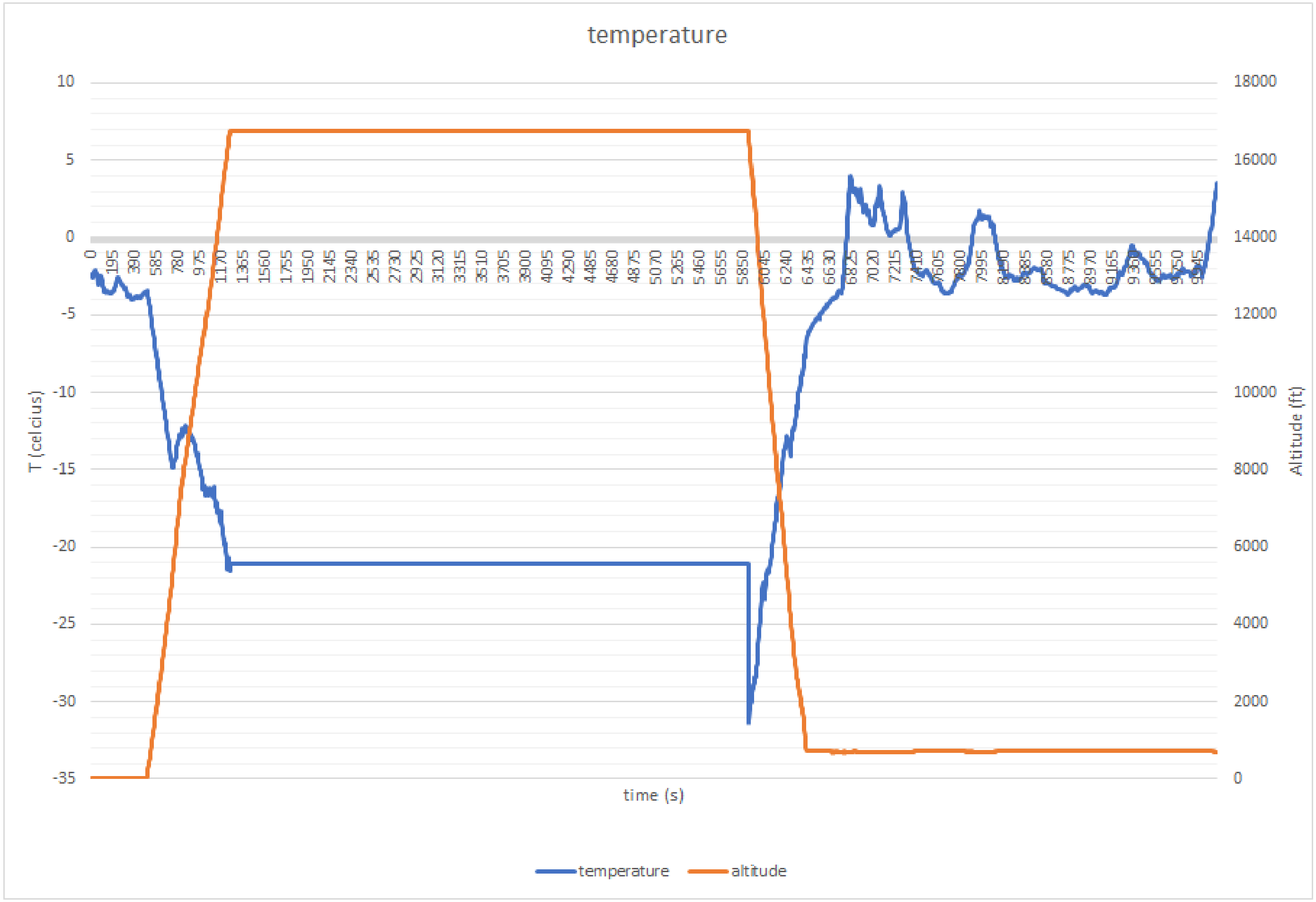


*Figure 4.10. Temperature (Celsius), Altitude (ft) vs Time (s).*

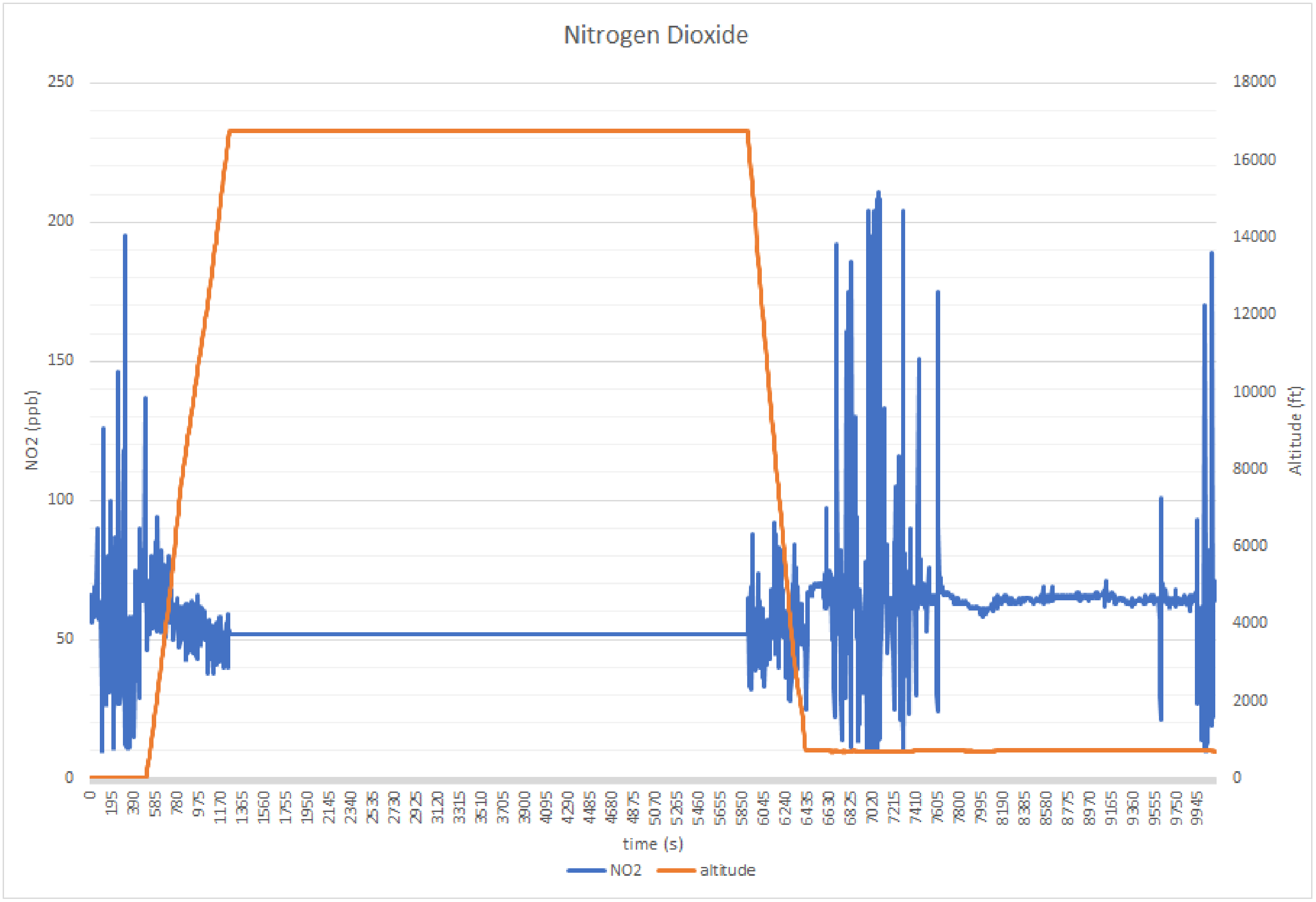


*Figure 4.11. Nitrogen Dioxide (ppb), Altitude (ft) vs Time (s).*

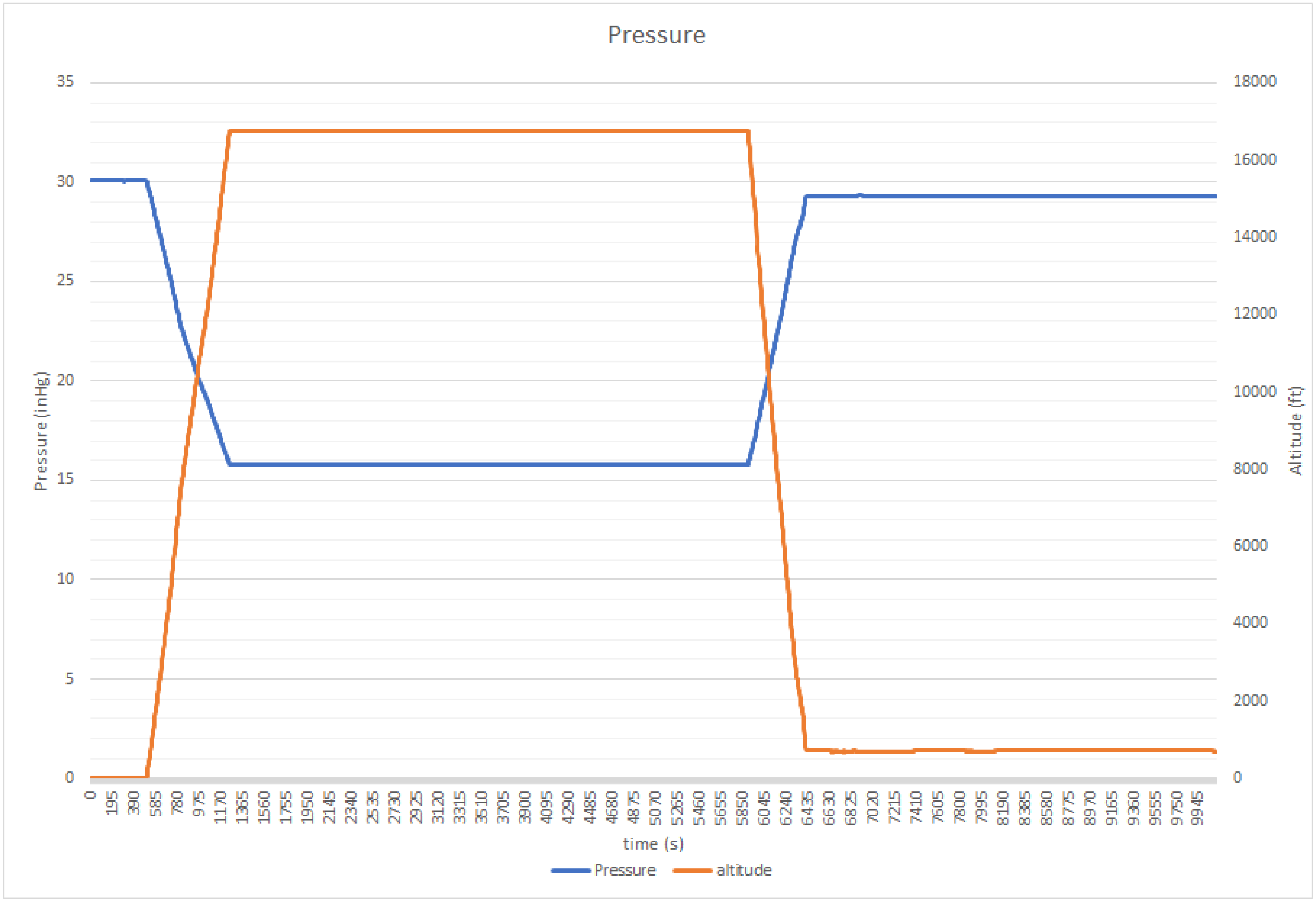


*Figure 4.12. Pressure (inHg), Altitude (ft) vs Time (s).*

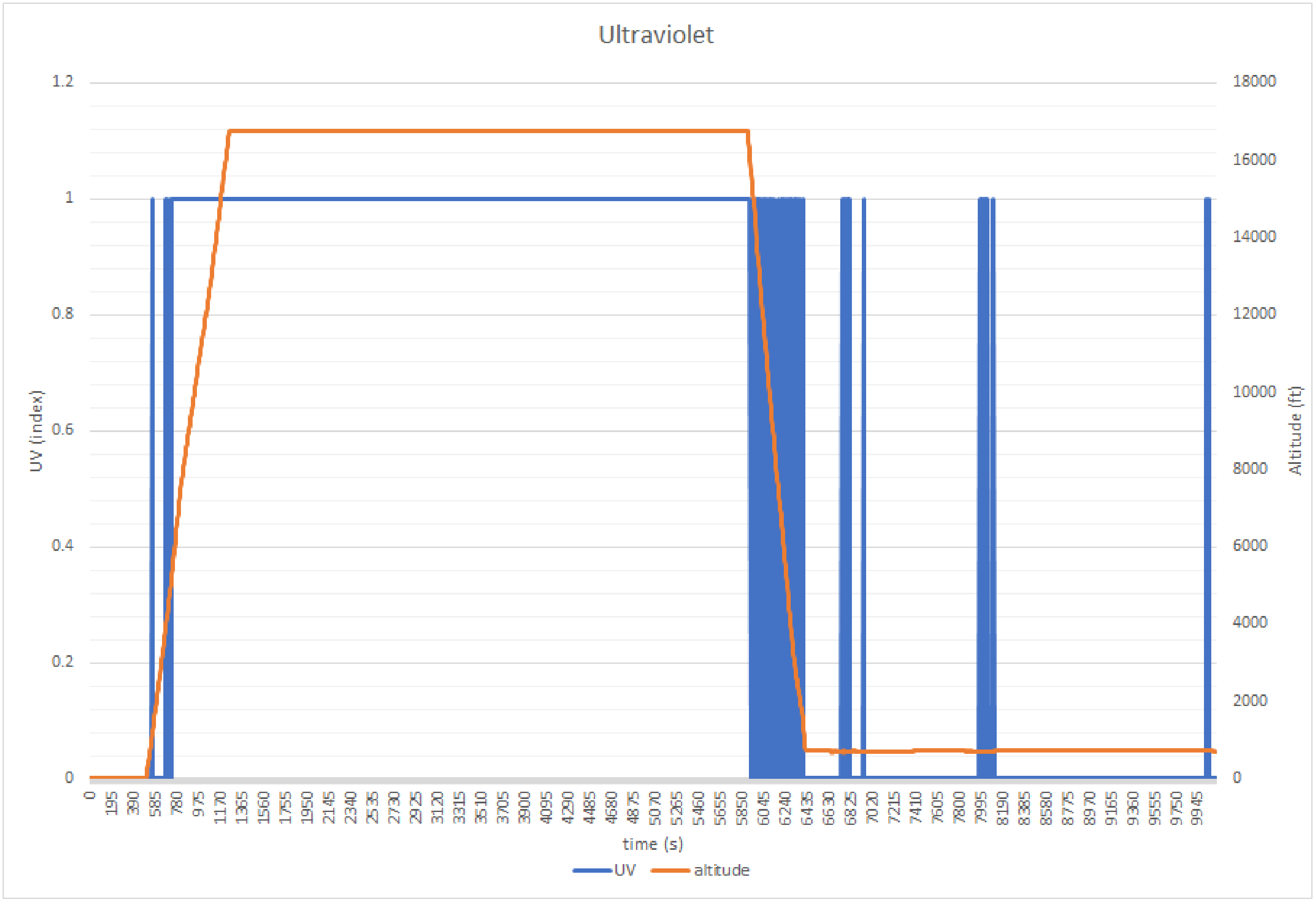


*Figure 4.13. Ultraviolet (UV Index), Altitude (ft) vs Time (s).*

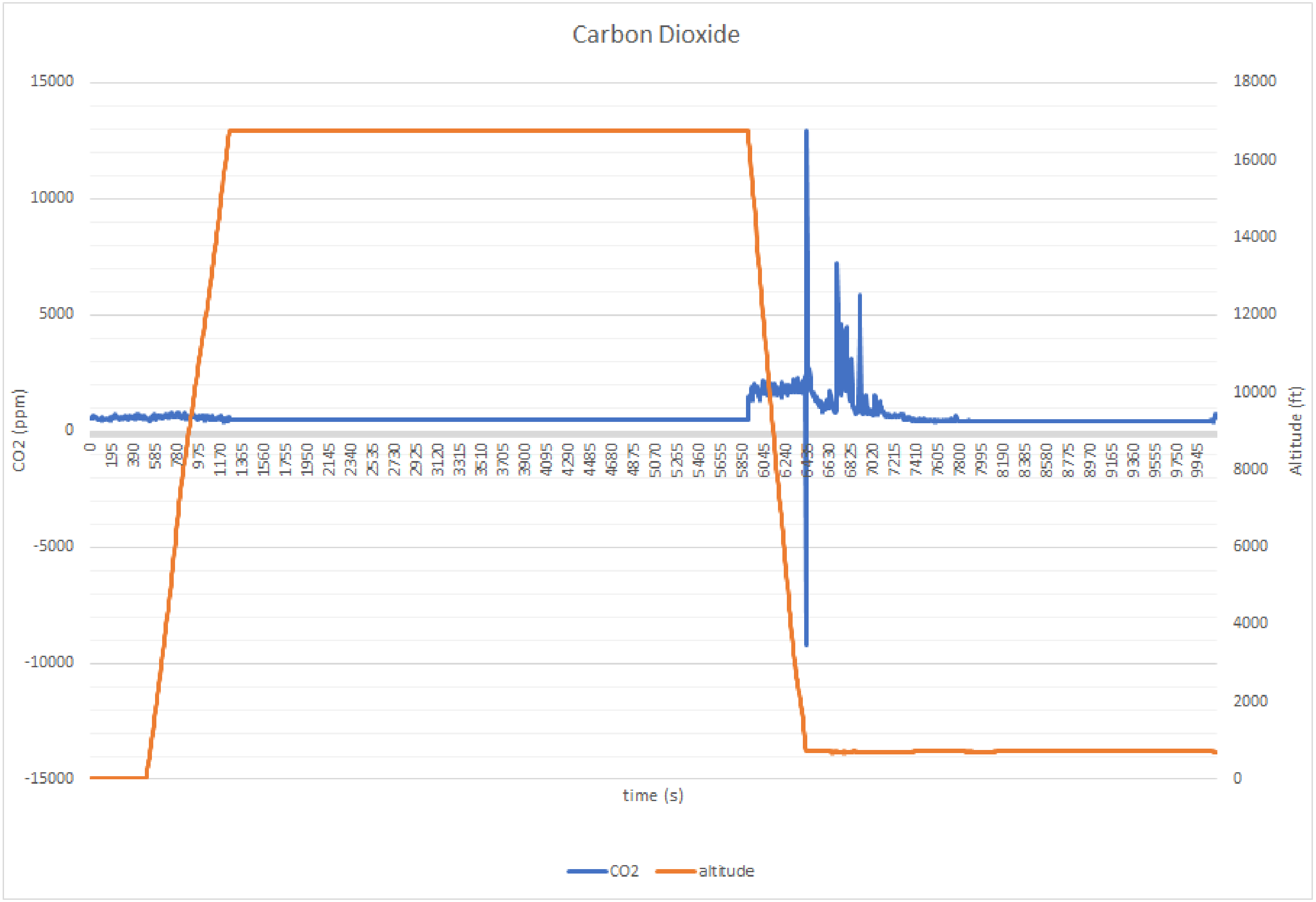


*Figure 4.14. Carbon Dioxide (ppm), Altitude (ft) vs Time (s).*

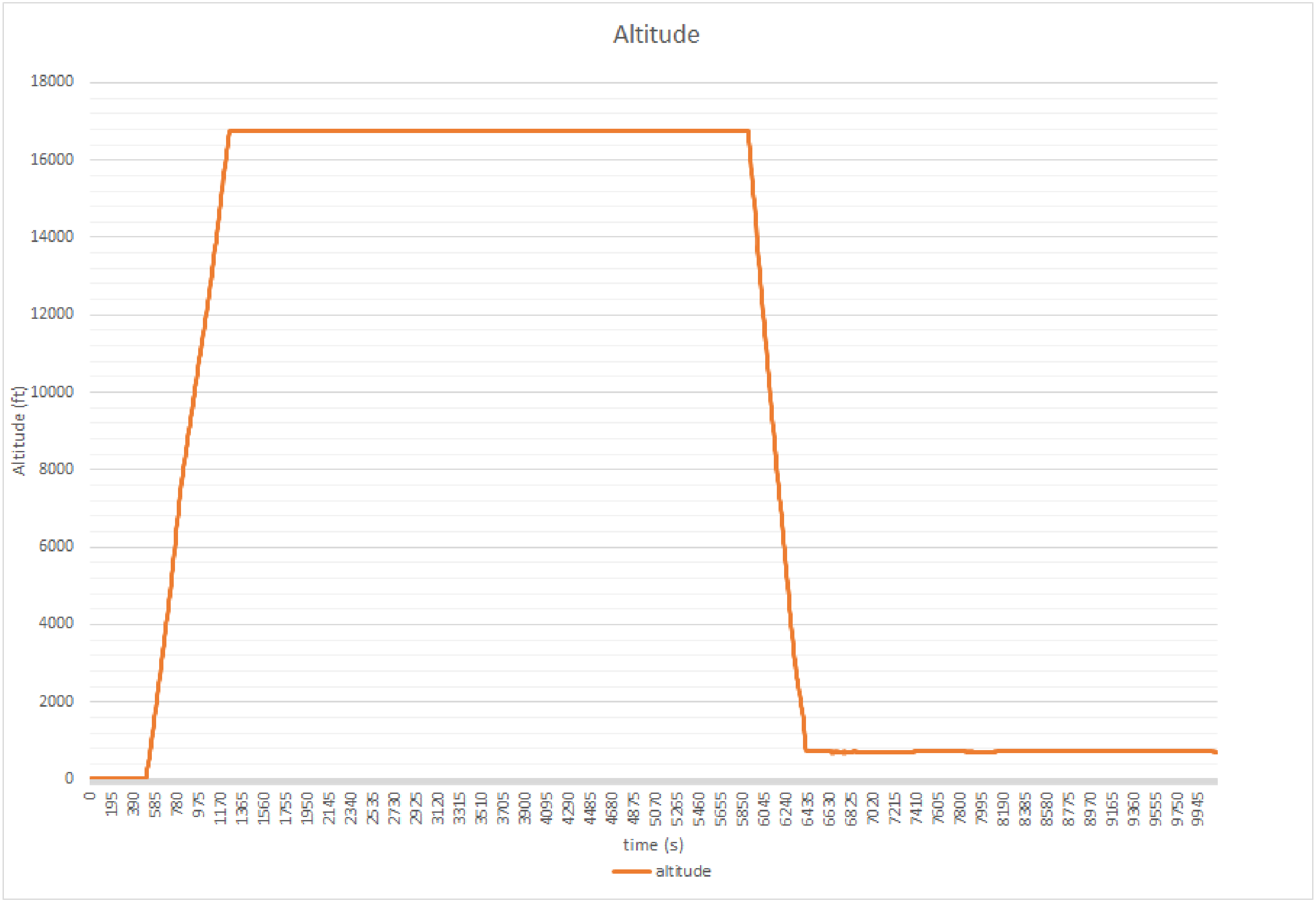


*Figure 4.15. Altitude (ft) vs Time (s).*

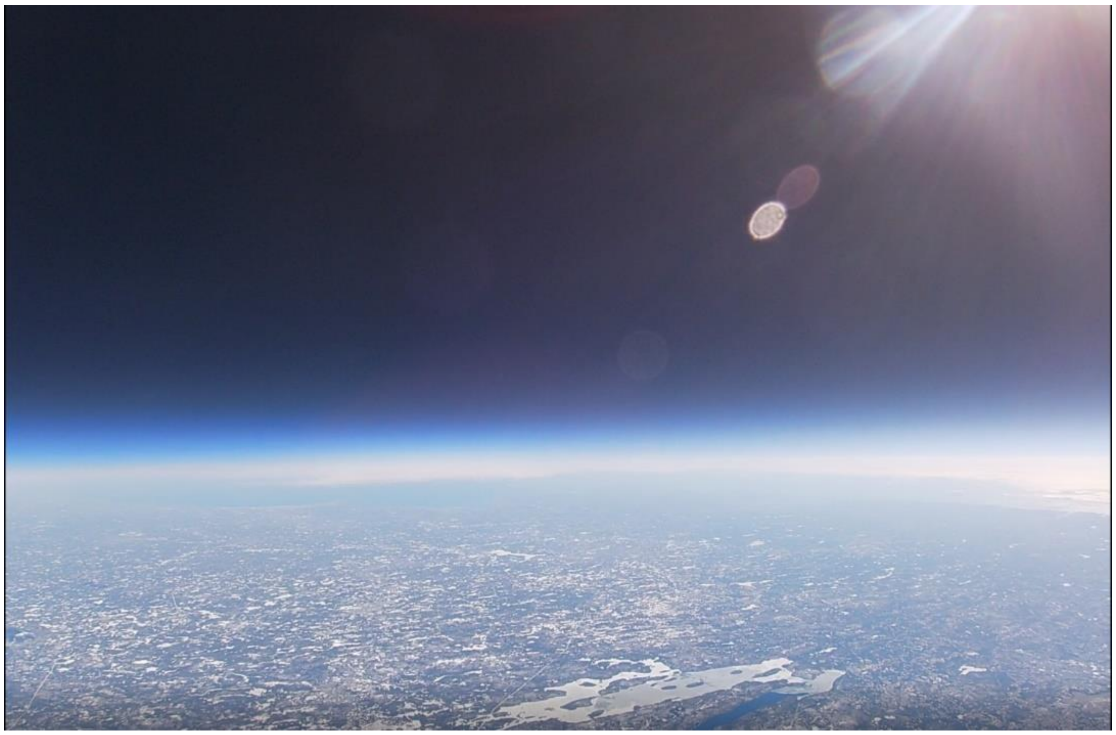


*Figure 4.16. Image captured at an altitude of approximately 16,500ft.*

## 4.5 Second Launch Miscellaneous Errors

As seen in the data collected from our second launch (Fig. 4.9 - 4.15), once our HAB reached approximately 16,700ft the collection of new data had stopped, and the data being collected was repeating until the payload reached a certain altitude on its descent. We believe that this stop in new data collection is associated with a problem concerning the Arduino Uno we used, since the Raspberry Pi we were using continued to write the data to the SD card after new data from the sensors had stopped being collected. The sensors also do not appear to have been the problem, since they were still collecting data during this stop in new data collection, and once the payload had descended to a lower altitude the data being collected had returned to normal. After analysis of the data and the integrity of the payload's electronics, our team believes that the

Arduino was unresponsive for some time due to the potential of small, momentary fluctuations in the voltage or current that the sensors in our payload would send to the Arduino. This most likely created a period of time when the Arduino was unable to send new data along to the Raspberry Pi, which resulted in our loss of data starting at 16,700ft. Data collection resumed after the payload reached 16,700ft after the burst. The data collected outside of that range is in accordance to the values recorded during testing of the sensors.

Another issue that we recognized from the data collected during our second launch was that the UV data collected only ranged from 0 to 1 on the UV index scale. One possible explanation for this problem could be that the payload was constantly moving during the flight of the HAB, and the sporadic movement and shaking of the payload could have made it difficult for the UV sensor to interact with light directly in a way that it was capable of collecting data properly. In our testing of the sensors prior to our second launch, the UV data we received from the sensor looked correct, since we saw an increase in the UV index value as we exposed the UV sensor to sunlight. When the UV sensor was under this testing situation, the payload was being held steadily and did not experience much movement or shaking, which clearly made the sensor capable of gathering data from the incident beams of UV light interacting with the face of the sensor. One possible solution for this problem could be to have the UV sensor positioned somewhere on the payload where it will always be exposed directly to UV light.

# Chapter 5. Conclusion & Recommendations

As the second team to work on this project, we were able to build on the previous team's work in a few ways to introduce new features to the HAB. One of these new features is the capability of our HAB's payload to transmit data from the sensors to a base station in real-time. The second feature introduced to the HAB by our team was the ability to terminate the flight at a certain point in the HAB's flight time. The third feature was the introduction of buck converters. The payload was using a single power source and buck converters to power all the sensors and microcontrollers. This reduced the weight and provided stable voltage outputs to the sensors. Our team hopes that in future iterations of the HAB project, even more features can be added to the HAB.

Over the course of three terms working on this project, our team made use of the knowledge and skills we have acquired from our time as Electrical and Computer Engineering students in all phases of our project's development. We researched extensively and made use of datasheets and online resources to reach conclusions for the electronics we wanted to use in our payload and for data communication. From this research we were able to create an elaborate system of electronics which collected data and transmitted this data to our ground station in real-time while also recording it on an SD card. We also carried out detailed research on how to go about predicting and simulating the flight and performance of our HAB. Finally, we worked thoroughly on the system level functionality of all facets of our project to ensure all stages of this project were operational.

With the idea of adding new features to this project with every year that passes, our team would like to offer recommendations to future groups who will be working on this project as well. During our time on this project, we realized that although our basis for real-time data communication was cost-effective and relatively reliable, more research and testing should be done on communications in the future to make improvements on the capability of the HAB to transmit data in real-time without a loss of communication at any point during the flight. We also believe that a different style Styrofoam box, one that is longer and has less height, would be more practical for fitting electronics and other parts inside of the payload. Next, we recommend that the following team should produce the data over the packet radio system. This will allow for long range transmission of the data using the HAM radio protocol. Finally, testing an Arduino Uno with all of the sensors connected over a period of two hours or so would appear to be essential in ensuring data collection from the sensors is without any error, since this would allow for a future team to see if this was the issue we encountered during our second launch and avoid it with a new solution.

The goal of this project is to build a HAB capable of collecting data of GHG concentrations at different altitudes for comparison with data presented in literature that was collected in scientific studies of air quality. We hope that future teams working on the HAB project will be able to make use of the data we have collected from our time working on this project to make practical evaluations of the change over time in air quality. The documentation of our thoughts, testing, and data collection will hopefully guide the next team's decisions and enhancements to this project.

## References

Bosch Sensortec. (2013, April 5). BMP180 Digital pressure sensor. Retrieved February 16, 2021, from BMP180 Digital Pressure Sensor

Ahmed, E., Dadhich, P., Olivier, J. Rogner, H., Sheikho, K., & Yamaguchi, M. (2015). *Global Warming: History, Science, and Solutions*(REP.) IPCC, pp. 111 - 140

Boden, T.A., Marland, G., and Andres, R.J. (2017). Global, Regional, and National Fossil-Fuel CO2 Emissions. Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, U.S. Department of Energy, Oak Ridge, Tenn., U.S.A. doi 10.3334/CDIAC/00001_V2017.

DFRobot. Gravity: Analog C02 Gas Sensor For Arduino (MG-811 Sensor). Retrieved February 16, 2021, from Gravity: Analog CO2 Gas Sensor For Arduino (MG-811 Sensor)

DFRobot. Gravity: I2C Ozone Sensor (0-10ppm). Retrieved February 16, 2021, from Gravity: I2C Ozone (Q3) Sensor

High Altitude Science. Eagle Pro Weather Balloon Kit. Retrieved October 19, 2020, from Eagle Pro Weather Balloon Kit – High Altitude Science.

Jiang, J., Lee, J., & Ru, H. (2020). *High Altitude Balloon* (Undergraduate Major Qualifying Project No. E-project-040620-170353). Retrieved from Worcester Polytechnic Institute Electronic Projects Collection: High Altitude Balloon

Linsey, Rebecca. Climate Change: Atmospheric Carbon Dioxide: NOAA Climate.gov. (2020, August 14) Retrieved February 05, 2021, from NOAA: Climate Change: Atmospheric Carbon Dioxide.

Macgyver603. How to Stream Video, Pictures, and Data From 90,000ft. (2017, September 18). Retrieved October 19, 2020, from How to Stream Video, Pictures, and Data From 90,000ft

Monolithic Power Systems (2011, August 8.) MP1584. Retrieved February 17, 2021, from MP1584 3A, 1.5MHz, 28V Step-Down Converter

Roithner LaserTechnik (2011, July 2). GUVA-S12SD. Retrieved February 16, 2021, from GUVA-S12SD

Rowell, E., & Rossman, J. (2015, November 15). A brief history of air pollution. Retrieved February 17, 2021, from A brief history of air pollution

SGX SENSORTECH. The MiCS-2714 is a compact MOS Sensor (rev 6). Retrieved February 16, 2021, from The MiCS-2714 is a compact MOS sensor.

Ubiquiti Networks. (2015). LiteAP 5 Ghz airMAX AC AP. Retrieved February 16, 2021, from LiteAP™ AC Datasheet

Ubiquiti Networks. (2017). Ubiquiti LBE-5AC-GEN2-US. Retrieved March 4, 2021, from LiteBeam® LBE AC Gen2 Datasheet

Wang, Zhanzhao, Min Huang, Lulu Qian, Baowei Zhao, and Guangming Wang. (2020, April 7) High-Altitude Balloon-Based Sensor System Design and Implementation. Retrieved October 19, 2020, from High-Altitude Balloon-Based Sensor System Design and Implementation.

# Appendices

## Appendix A: GitHub Link

Github repository for code

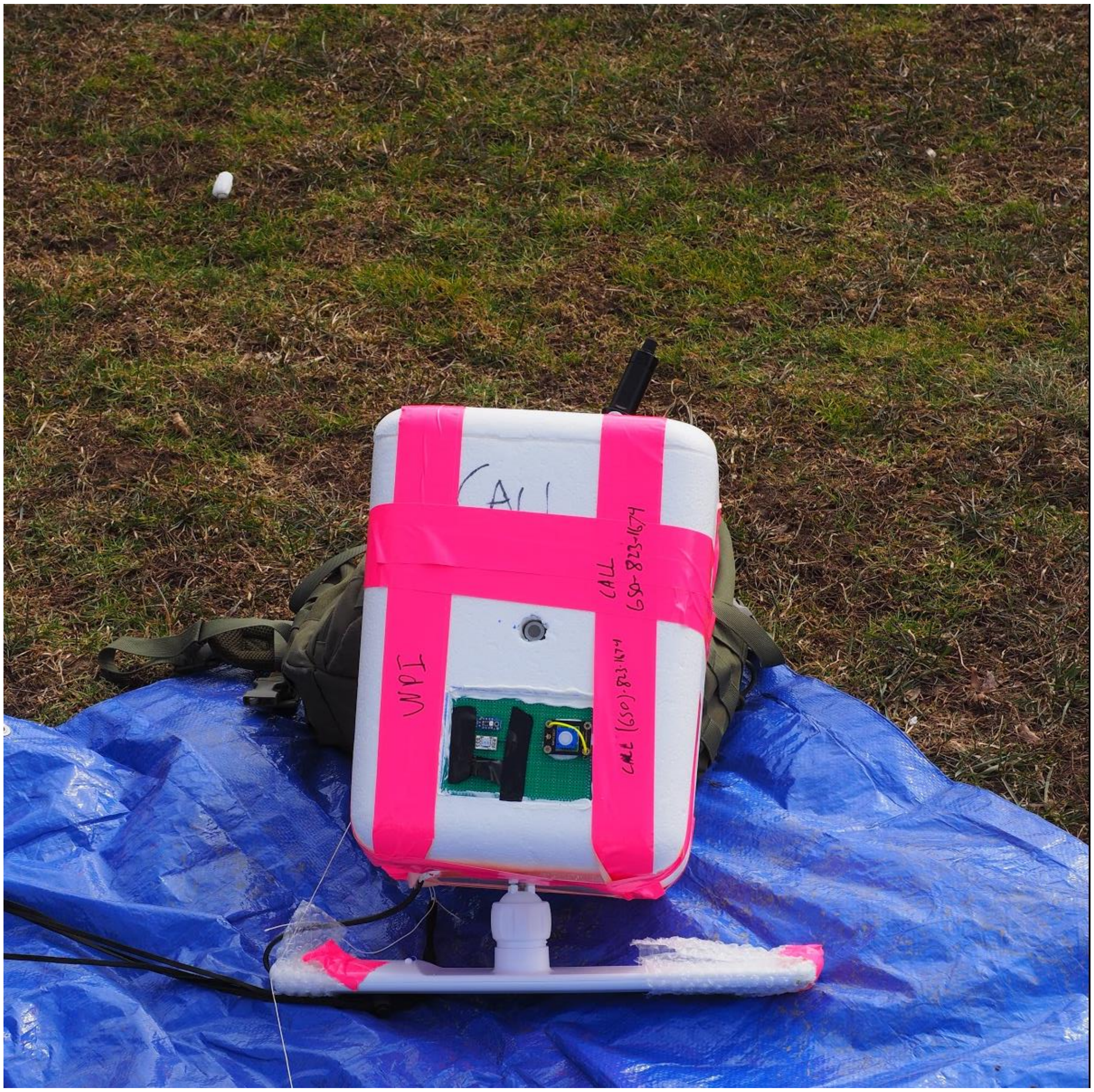
## Appendix B: Livestream recording of first launch

Weather Balloon Launch | 70,000 ft into the Stratosphere | HAB 2.0

## Appendix C: Setup

## Appendix D: Filling Balloon with Helium Gas

## Appendix E: Second Launch Payload at Albany, NY



## Appendix F: Retrieval of HAB