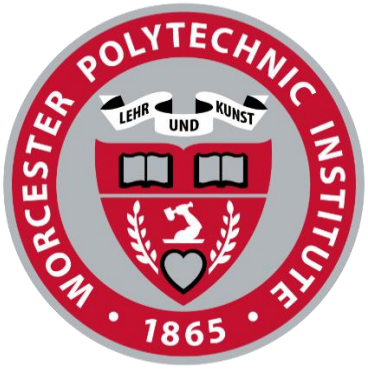


WPI

HAB Startup Guide

By Drew Solomon and 4.0 Team

Table of Contents

Notice from the Author

This guide was created for future WPI HAB project groups to become familiar with the current state of the project faster to begin making progress on improvements sooner. The 4.0 team had to spend the majority of the first project term piecing together info from scattered resources due to unexpected changes in administration, so we made this guide to prevent such an occurrence from troubling future groups.

It would be greatly appreciated if this guide could be passed on from each group to the next, with the additions made each successive year recorded.

Good luck and have fun!

-Drew Solomon, HAB 4.0

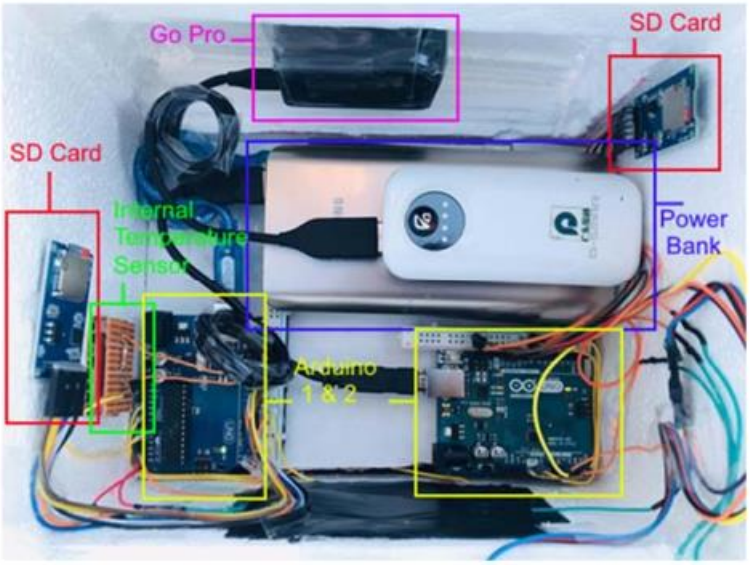
What the Project Is

The HAB Project was started in the 2019-20 academic year by ECE Professor Maqsood Mughal. The project had students build a payload of gas sensors to be carried into the sky by a large helium balloon, which would transmit live data to the ground. The data would be received by a base station connected to a laptop, which included a large antenna. A more complete set of data would be extracted from the payload when it was found after landing. The primary purpose of the project is to measure the level of certain gases in the atmosphere, and to compare the data from year to year to make conclusions about the changing state of air quality, etc. Each year new additions are made to improve upon the shortcomings of past iterations, and to expand the functionality of the payload. Most years 2 launches are done during the project period of (historically) 3 terms, with one during late fall and one during early spring/late winter.

Past Iterations of the HAB Project

HAB 1.0

The main goal of the project was to measure the concentration of greenhouse gases and pollutants in the atmosphere by making an HAB that was economic enough for small institutions and groups such as ourselves. Starting from the first year, this project was meant to be held annually such that data could be collected and compared over a long period of time. Gases to measure were determined based on criteria such as the Air Quality Index. The target altitude was near 100k ft., and they determined that temperatures as low as -55 degrees Celsius could be reached in the sky. The group did two launches in Albany, NY, and three total. During the last launch, wires disconnected and caused the air quality sensors to not record data to the SD cards. Air quality data was the largest weakness of the project due to this occurrence.



ECE Advisor: Prof. Maqsood Mughal

Launch Site: Albany, NY

Flight Time: 2 hours

New Additions:

- UV Sensor
- Nitrogen Dioxide Sensor
- Carbon Dioxide Sensor
- Ozone Sensor
- Barometric Sensor
- Temperature Sensor
- GoPro Camera
- Radar Reflector
- Spot Tracker GPS

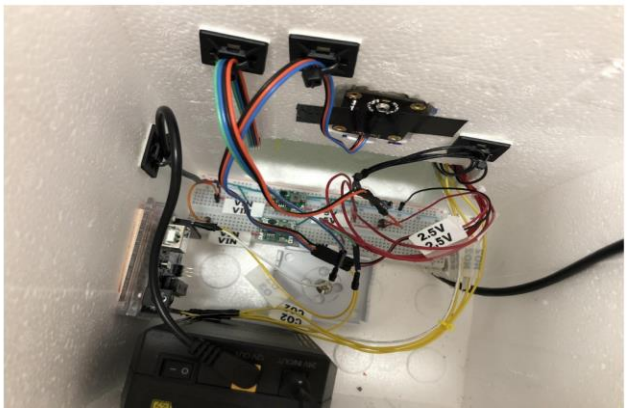
- Radar reflector

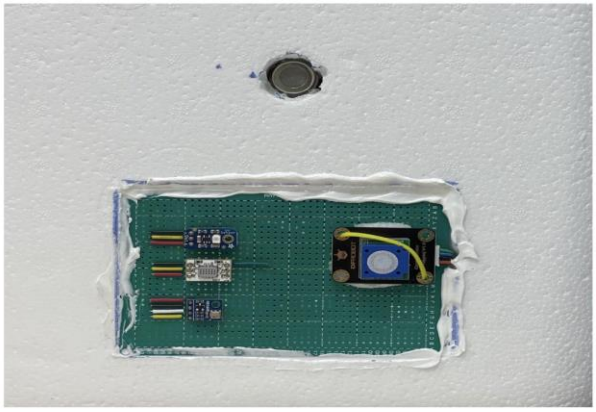
Problems for Next Year:

- Make wire connections more secure

HAB 2.0

This year's goals were to add real time data communication and the ability to terminate the flight to the payload from the year prior. The same payload from the first year was used as a starting point. The group attempted to make a system to terminate the flight earlier by heating a nichrome wire wrapped around a line attaching the balloon to the payload when certain conditions are met. It is unknown if this system was ever tested in the sky, as this is not specified in the report. The team purchased a radio antenna and receiver so that communication from the payload in the sky to the ground could occur. The team also used the ThingSpeak API to make graphs in real time, relaying the received payload data on the ground to the application.





ECE Advisor: Prof. Maqsood Mughal

Launch Site: Lincoln Park, Albany NY

Flight Time: 2 hours

New Additions:

- Added a Raspberry Pi to use in tandem with the already present Arduino
- Powered system off of a single power supply to reduce weight
- Switched out some sensors for new ones (unknown why)
- Payload organizational overhauls
- Flight termination system using a MOSFET that dumps current through nichrome wire
- Radio communication to transmit sensor data live to ground (antenna and receiver)
- Used 3D flight tool that shows location of balloon pop in sky in addition to landing site (ASTRA simulation tool)
- Broke up into 2 teams during launch – one to launch balloon and one further along flight path to receive data from it

Problems for Next Year:

- Payload suffered damages from being pried open by resident of landing site
    - Make a clear note to attach to the payload that says this is a student project + give contact information
- Payload weighed more than expected and thus had more time in the air and traveled further than expected
- Poor insulation and loose wires caused sensors to fail at high altitudes
    - Only functional up until 16,700 ft, and on descent below this altitude

HAB 3.0

The main goal of this year was to improve the communication from the payload to the ground and collect a whole launch worth of data in real time. A formal base station was created for the first time, consisting of a wooden box with a motor attached to it to the receiver antenna may rotate. Unfortunately, during a launch this year the payload that had been used since year 1 was lost in the Atlantic Ocean, and a replacement payload was worked on after. The replacement payload was tested on a hike up Mt. Wachusett in Massachusetts to test sensor functionality. By the end of the project the new payload was not complete, and the base station was still a work in progress.

ECE Advisor: Prof. Maqsood Mughal

Launch Site: Iorio Park, Cobleskill NY and Beverwyck Park, Albany NY

Flight Time: Unknown (fell into ocean)

New Additions:

- Second, redundant GPS sensor connected to the Pi
    - Reorganized computer/sensor system to incorporate GPS sensor
- More secure wire connections to ensure no malfunctions mid-flight
- New radio hardware to better transmit flight data

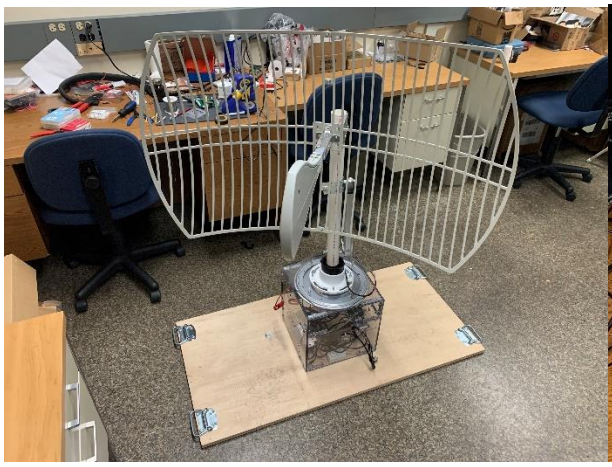
- Base station to mount receiver antenna during launches
- Wrote new code for Raspberry Pi using Python instead of C
    - Uses threading – Arduino, Pi, Transmit threads
- Rewrote Arduino code to be more efficient
- Reduction of payload box size for better insulation
- Parachute release device which automatically deploys parachute after dropping below set altitude

Problems for Next Year:

- Replacement payload not complete – needs part replacements and soldering
    - This payload has not been flown yet – make sure it's ready for flight
- Base station incomplete – no code to run motors and needs structural overhaul
- Organizational overhaul of the payload box – better insulation
- Implementing the early flight termination system
- Get an ESP32 to use with the base station motor
- Get an amateur radio license to operate at normally prohibited frequencies
- Find an NO and/or methane sensor to use
- Helium shortage is getting worse, find a new supplier

HAB 4.0

HAB 4.0 was a very unique year for the HAB project. It was the first year without Professor Mughal as the project advisor (Professor Gregory Noetscher took over), and it was the first year since the beginning where there was not a complete and flight-proven payload to start off with. Despite losing the project's main source of practical knowledge, a great deal of proactiveness and initiative led to some great changes for the project. This is also the first year to not have a flight of the payload – helium had grown to be such an expensive resource that once the team had finished acquiring hardware for advancements, there was no longer enough funds.



ECE Advisor: Prof. Gregory Noetscher

Final Test Site: Green Hill Park, Worcester MA

Flight Time: N/A (Payload was run for about an hour straight on the ground)

New Additions:

- Overhauled base station with 2 degrees of motion and code to align with payload in flight using GPS
- Overhauled payload organization and code
    - CSV files now used to store data
- Implemented early flight termination system with nichrome wire
    - Includes a programmable ellipse-shaped GPS boundary
- Payload equipped with image transmission capability
- New sensors:
    - Methane sensor
    - New UV sensor
- Startup guide created to help future years assimilate with the current state of the project easier

Problems for Next Year:

- Improvement of the base station's motion and payload tracking capabilities
    - Use a motor with an encoder and a linear actuator with an encoder or more limit switches. (PID Control)
- A return to real time graphing of flight data w/ ThingSpeak API used in past iterations
- Experiment with sending messages from base station to payload
    - Early Termination System trigger message
- Fly the payload after more progress is made!

Current State of the Project (Post HAB 4.0)

As the project stands currently, the payload functionality is solid and trustworthy. Any improvements made to the payload code at this point would be for quality of life and ease of testing, the core functionality seems to be working great. The main improvements to be made are on the base station, which has come a long way since HAB 3.0 (and was completely overhauled) but unfortunately didn't have enough time for testing to work out all its kinks. The base station is supposed to be able to point at the payload based on its GPS location relative to the base station, but with the current hardware it often is unable to do so. Because of the friction induced when the shaft of the satellite rotates, it often doesn't make the full rotation it needs to get to its next position. This then causes the satellite to be out of synch with the computer system telling it how far to turn next.

The other place for improvement is in the unexplored territory of sending messages to the payload from the base station. Usually communication works the other way around with the payload sending its sensor data to the base station. There's some potential interesting features to implement here such as remotely activating the Early Termination System from the ground.

Since HAB 4.0 didn't have time to focus too much on the processing of test data (let alone actual flight data), we chose not to delve into the realm of real-time data processing and graphing. Past teams used the ThingSpeak API to do this, and it generally seems like a good idea given how powerful the tool is.

Initial Preparations

Get Project Room Keys

The workspace for the project (which unless it changes, should be AK 316) is always locked and will require the project members to each acquire a room key from the ECE department. A $20 deposit will be required, and from the date the office receives the key request, it may take up to 2 weeks for the key to be made. Contact Colleen Sweeney (sweeney@wpi.edu) with your key requests.

Register with ECE Shop

In order to begin using your project budget (which was $250/ECE team member during 4.0) you need to register your team with the ECE shop on the first floor. William (Bill) Appleyard (wlappleyard@wpi.edu) is the man you want to talk to, and he should have print forms for you to fill out with your member and advisor details. When you need to buy parts for the project, send him an email with links to the pages and he will order them ASAP. Notably, Bill doesn't usually send a confirmation email that he has ordered parts but he will let you know when they arrive.

WPI MakerBucks Program – Free Budget Money

The WPI MakerBucks program is run through Foisie and funds both personal and school related projects. During 4.0 the program was giving $200 to go towards certain project expenses for those who applied and were accepted. The program is not super well known and doesn't receive many applicants, so we were able to get in with ease. The only catch is occasional meetings to discuss use of MakerBucks funds and project progress. The main contact for this program is Mitra Anand (mvanand@wpi.edu), and the current website for the program is https://www.wpi.edu/about/innovation-entrepreneurship/programs/makerbucks.

Look into Helium Logistics ASAP

Helium is currently in short supply due to a worldwide shortage, and getting it will very likely be one of the most irritating parts of the project. It will also likely be the single most expensive item on your bill for the year, so getting an idea of how much it will cost is extremely important for budgeting. Please check the Helium section further into the guide for details.

Apply to ECE Department if Extra Funding Needed

The 4.0 team did not end up flying due to budget restrictions, but in talking to the ECE department we found out that early on in the project (probably as early as possible) you can apply for extra funding if it is critical to the success of the MQP. If we knew helium was going to be as expensive as it was when we were going to fly, we would have done so if we knew this was an option. Mention what happened to the 4.0 team in your case to the department!

Get Familiar with Legacy Resources

Efforts have been made to make it easier for future teams to get acclimated to the work already done on the project, this guide included. We highly suggest you poke around the files from past years, particularly the one before you since it's likely the most organized and detailed. Taking a moment to check out the BOM, schematics, code repository, etc. will do wonders for getting started.

Getting the System Working

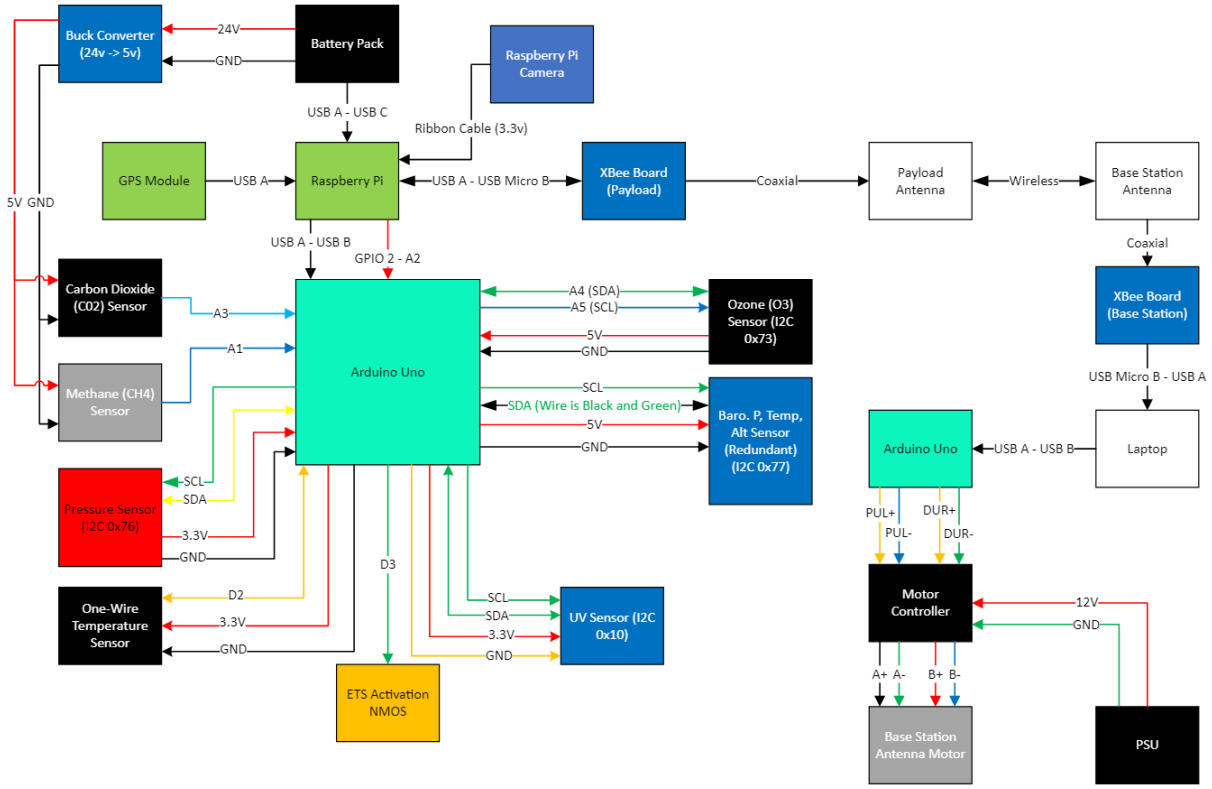
Payload

The payload's "brain" for lack of a better term is a Raspberry Pi. The Pi powers and communicates with an Arduino which periodically samples from an array of sensors and relays the data to the Pi. The Pi also has a GPS module and camera connected directly to it. The Pi is also connected to a battery pack which powers everything in the payload and is included in the payload box, as well as a radio module which transmits data to the ground via radio. In addition to powering the Pi, the battery also heats the gas sensors and supplies power to the Early Termination System when activated, with separate connections for both. Components with an SDA and SCL wire function through the I2C protocol, which is largely handled by dedicated Arduino libraries for the each of these parts.

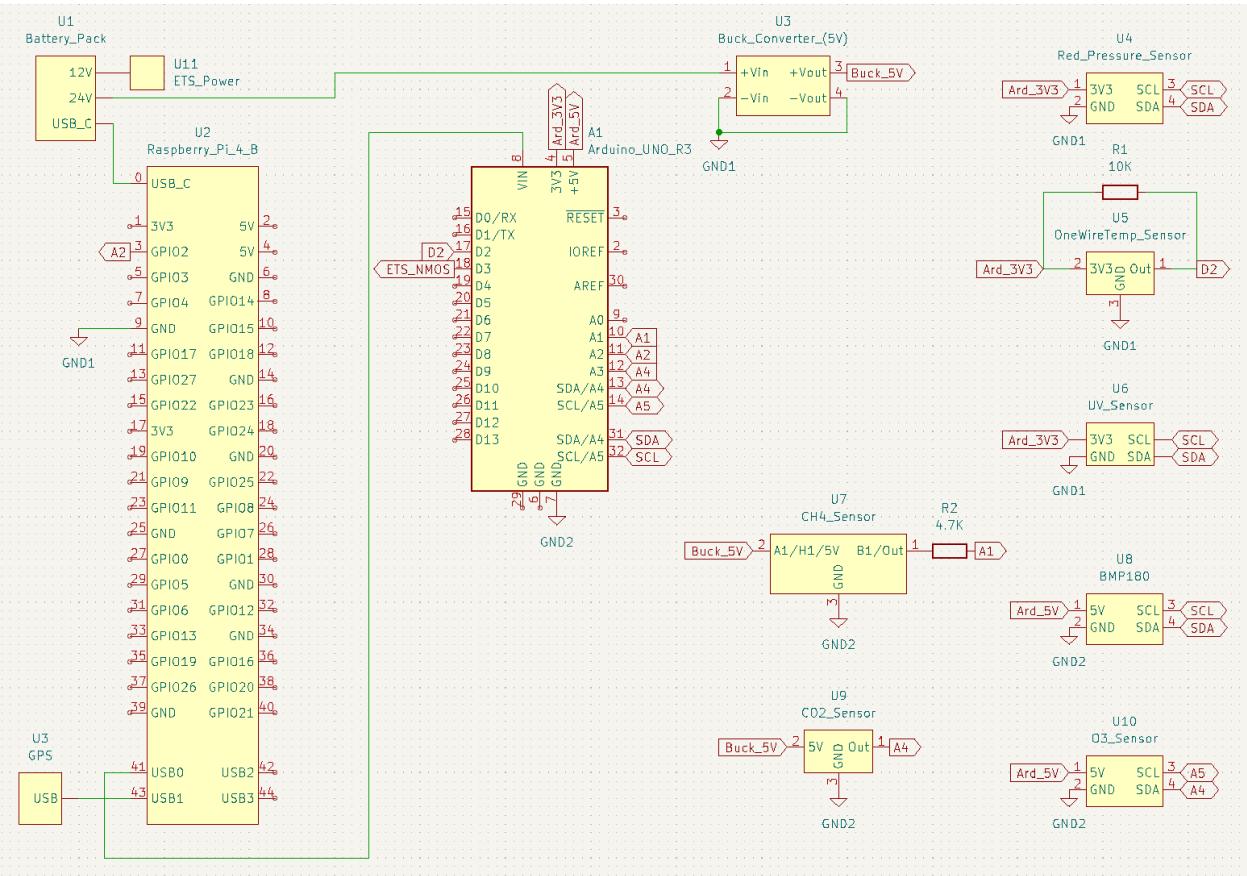
Sensor	Product Name	Operation Range (°C)	Voltage Input (V)	Sensing Range	Output Type
Carbon Dioxide (CO2)	Gravity: Analog Electrochemical Carbon Dioxide Sensor	-20 - 50	3.7 - 5	0-10000ppm ± 100ppm @ 400ppm	Analog
Ozone (O3)	Gravity: Electrochemical Ozone Sensor	-20 - 50	3.3 - 5.5	<=10ppm, 10ppb resolution	Digital (I2C)
Methane (CH4)	Methane CNG Gas Sensor - MQ-4	-10 - 50	5±.1	300 −10000ppm	Analog
Barometric Pressure	SparkFun Pressure Sensor Breakout - MS5803-14BA	-40 - 85	1.8 - 3.6	0 - 14 Bar, 0.2 mBar Resolution	Digital (I2C)

One Wire Thermometer	Temperature Sensor - High Temperature, Waterproof (DS18B20)	-55 - 125	3 - 5.5	-10°C to +85°C ±0.5°C accuracy	Analog
Ultraviolet Light (UV)	SparkFun UV Light Sensor Breakout - VEML6075 (Qwiic)	-40 - 85	1.7 – 3.6	280-400nm Light Wavelength	Digital (I2C)
Barometric Pressure, Temp, Altitude (Used as Backup)	JBtek BMP180 Barometric Pressure, Temperature and Altitude Sensor	-40 - 85	3.3 - 5	Pressure: 300hPa - 1100hPa Altitude: <= 30,000ft	Digital (I2C)
GPS Module	GPS & GLONASS Antenna Module YIC93030PGMFUGG-U8	-40 - 85	2.8 - 5.5	~2.5m Max Position Accuracy, Altitude: <= 50,000m	Digital (NMEA 0183)
Camera Module	Raspberry Pi Camera Module v2	-20 - 60	3.3	3280 x 2464 Still Picture Resolution	MIPI Camera Serial Interface (CSI-2)

Above is a table of all the components currently included in the system, and the connections between components are detailed below, with color coding of the wires corresponding to the colors of the wiring in the payload:

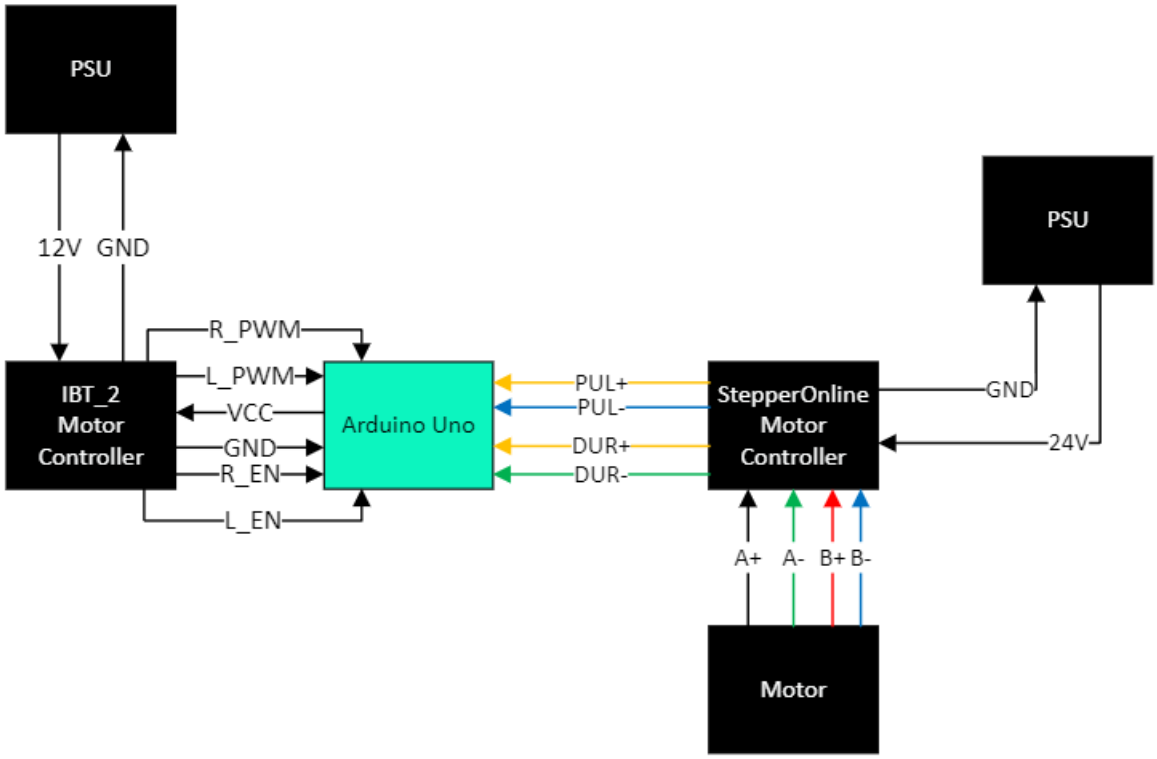


Below is an electrical schematic of the payload connections with some more detail:

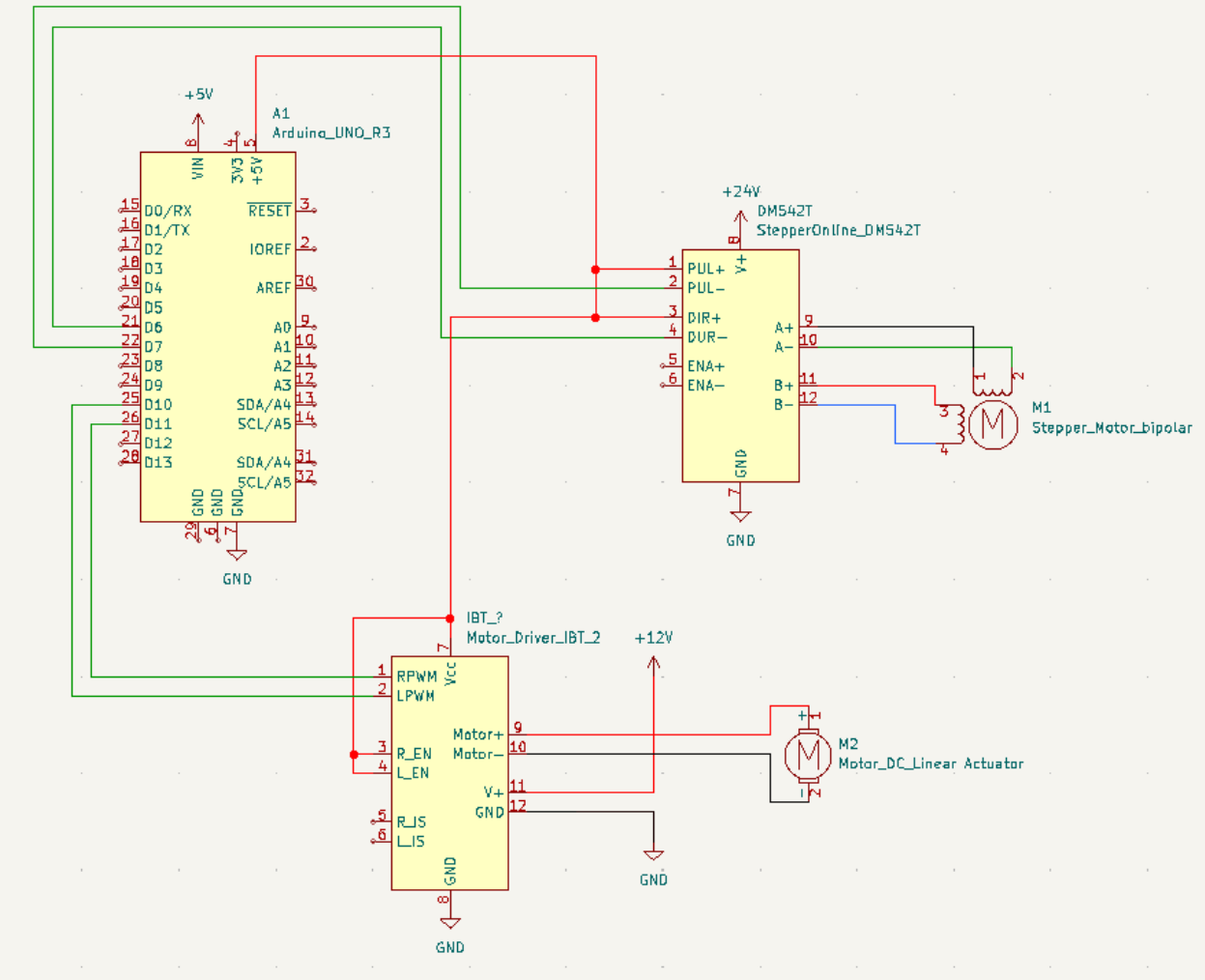


Base Station

The base station receives data transmitted from the payload and rotates the receiver antenna so that it points at the payload. A basic diagram of base station connections can be found above, with a more detailed diagram shown below. The high-torque motors are driven by a stepper motor controller, managed by a microcontroller. The code run on the microcontroller from a laptop is used to determine the positioning of the motors so that the antenna points in the right direction based on payload GPS data. Make sure the 12V goes the IBT_2 Motor Controller, and the 24V PSU goes to the StepperOnline Motor Controller. Make sure the Anderson Connector for the Linear Actuator Power Pole connectionis made and that the USB cables the Arduino and XBee are plugged to the base station laptop.



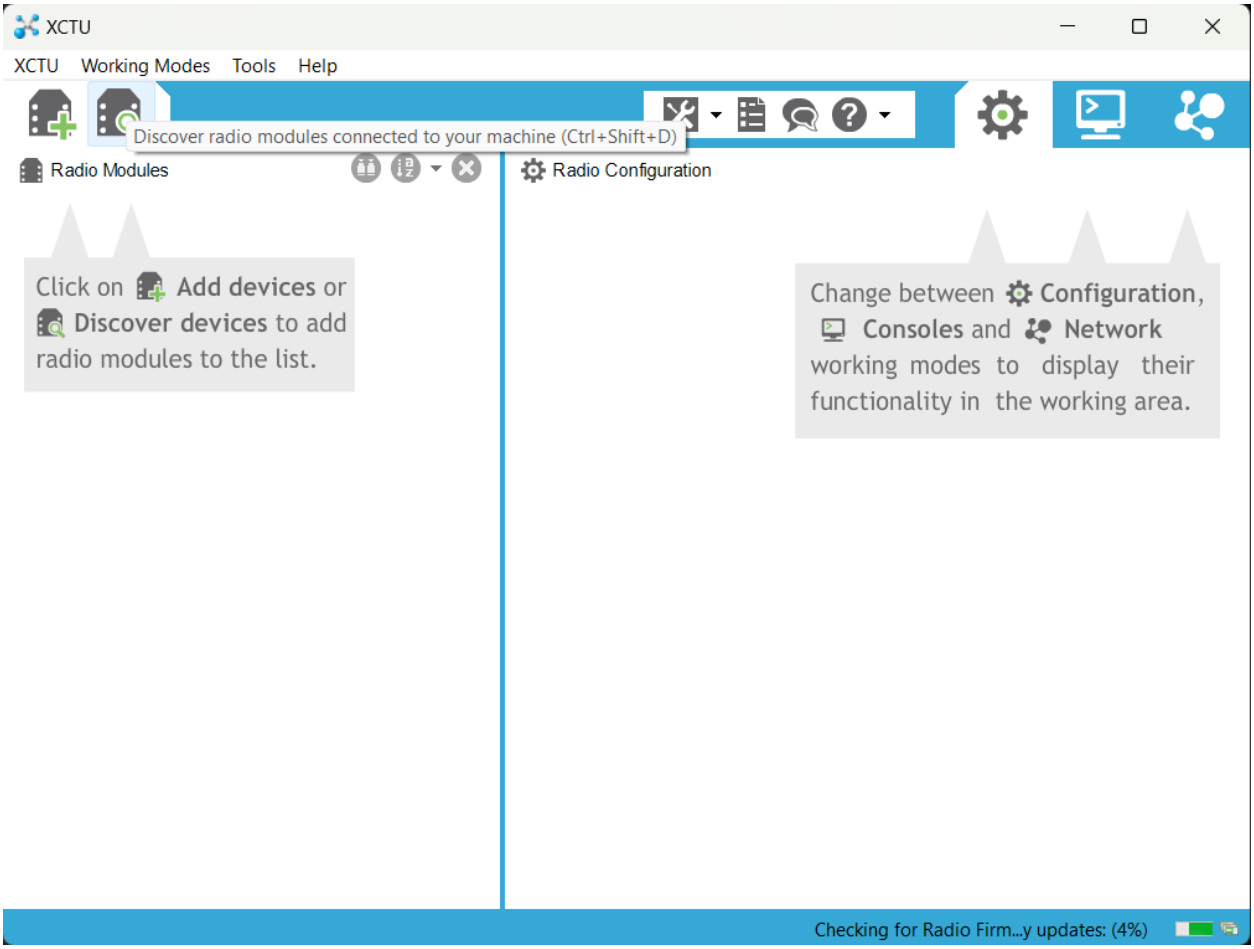
Below is an electrical schematic of the payload connections with some more detail:



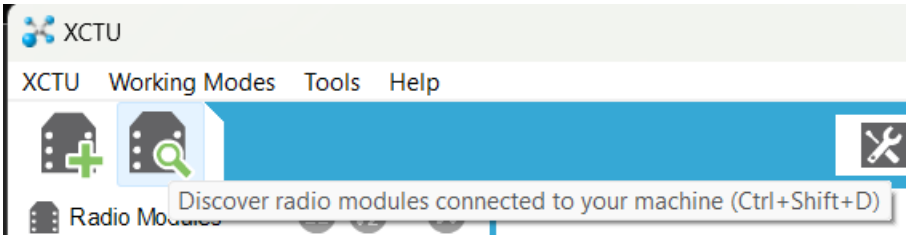
Setting Up Base Station Laptop

XCTU Software

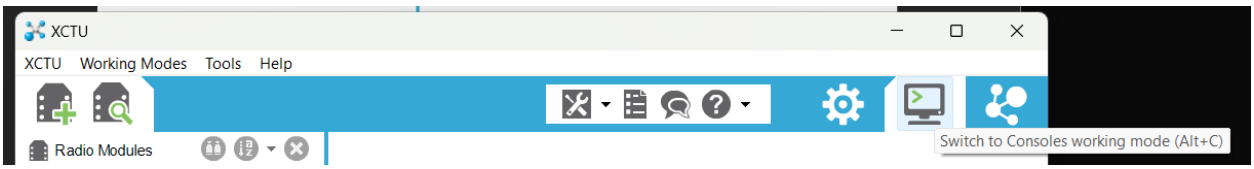
In order to receive information from the radio module on the base station when the payload is transmitting data to it, you'll need to hook up a laptop by USB to the module. You'll need a free software called XCTU (https://www.digi.com/products/embedded-systems/digi-xbee/digi-xbee-tools/xctu), which is made by the manufacturer of the module.



Above is what the main menu will look like.



Once you have the radio module plugged in and XCTU opened, click the discover radio modules button in the top right corner of the window. Follow the discovery process to find what USB COM port the radio is connected to and set it up with the software.



Once setup is complete, click the switch to consoles working mode button (looks like a computer monitor) and you will be able to view the radio's received data transmissions on the console. Click the button that says "open" on the left side of the console window to begin viewing the radio

transmissions. NOTE: If you aren't getting any messages on the console, double check that you have the payload set to transmit and see if the TX light is flashing on its respective radio module. The RX light should be flashing on the base station module as well.

Interfacing with Radio Through Python (Unfinished)

Helium

How to Acquire

Helium will be a pain to find, especially helium that can be delivered to your launch site. Transporting helium without certifications is highly illegal and unsafe, and unfortunately limits options by a lot and forces us to have someone deliver it for us. Here's some tips to keep in mind when finding your supplier:

- Your main suppliers are going to be **welding supply stores** and **party rental stores**
    - Had better luck with welding stores, party stores often only sell in small tanks
- Call each viable option you find for a quote – this is non-committal, just so you can compare prices from all your options
- Pay through the ECE shop, and have the ECE administration office make the account with the supplier
    - Getting reimbursed by the shop for payments can take months – just pay up front with your WPI project funds instead
- Make sure you mention you are with WPI and that WPI is a tax-free institution
    - This will shave the tax off your quotes and lower the cost… nice!
    - Will need to have the supplier fill out a W-9 form and return to Deborah Thompson (dlthompson@wpi.edu) at the ECE administration office (ask her for the form as well)
- Expect a deposit to be added to your bill if doing business with a supplier for the first time
    - Usually this will be refunded when the tank is returned
- Some suppliers will only sell to organizations/individuals who already have an account because of the shortage, so expect a lot of places to shut you down on that note
- The stock of suppliers changes so frequently, you can't exactly predict who will and won't have stock when it comes near launch day
    - Suppliers won't hold helium for you usually
    - Check in with your helium suppliers 1-2 weeks before launch on their stock, and buy ASAP
- Each supplier has their own delivery day – you will need to plan your exact launch day around when your supplier can deliver to you
    - Most suppliers we saw only delivered on a single weekday
- Make sure to request a dolly and a regulator! More on what their purposes are in the section below

Although HAB 4.0 didn't end up having a flight, our projected supplier for a launch in Cobleskill, NY was going to be **Linde Welding Gas & Equipment Center**, since they had stock at the time and were willing to cooperate with a delivery to the launch site. We highly recommend contacting them! Their complete contact info should be in the BOM doc for HAB 4.0.

Safety

Pressurized tanks of gas are EXTREMELY dangerous if not used properly, but by the time you get your hands on one you will already be well-informed on how to use one. Your advisor is in charge of making sure that you are trained to handle helium tanks, but you are in charge of scheduling necessary training sessions, etc. Ahead of the training sessions you will need to attend, here are some big tips:

- Move the tanks by "waddling" them – never attempt to take it off the ground, just wobble it forward bit by bit
- A cylinder dolly should be used to move the tank anything more than a few feet, and is much easier to use as well
- Make sure the tank stays **upright at ALL times**
- **DO NOT** transport a tank in a personal vehicle! It is **ILLEGAL** and **UNSAFE AS HELL FOR THE PASSENGERS**. Get a supplier close to your launch site and get it delivered day of!
- The tank will **need a regulator** – it fits to the nozzle of the tank and regulates how much gas leaves the tank when opened. Without one, if you open the tank all gas will leave at once… kind of acts like a bomb?
    - Has a few gauges on it – can't miss it
    - Check with your tank supplier to make sure there is a regulator on the tank you rent!

Certifications and Training

Only one team member is required to have full safety training (at least during 4.0, worth double checking on), but it is suggested that as many people as possible attend. The safety training is divided into two components:

Goddard Chem Dept. Training

This session was only a brief half-hour long and had us handle some empty gas tanks at Goddard. Regardless, still important to try before handling an actual tank of gas. Contact Paula Moravek (pmoravek@wpi.edu) to set up this meeting.

Environmental Health and Safety (EHS) Training

In getting approval for purchasing a helium gas tank, you may be asked to do this training, which is held at Gateway and consists of a single 2-hour session run twice a week or so. This is a general lab safety course, and while we aren't working in a chem lab during this project, has good safety pointers on best practice when handling dangerous materials. When Drew Solomon of HAB 4.0 took it, there wasn't any information given on handling gas tanks, but the administrator pointed him to someone who could give training on how to use a regulator on a gas tank. Register for this session here: https://cems-web-p-u01.wpi.edu/CEMS/Dashboard.

HAM Radio License

An amateur radio license called the HAM license will be necessary for communicating data at frequencies high enough to send large data such as pictures and video. The license grants access to all Amateur Radio frequencies above 30MHz (notably not ALL frequencies above 30MHz!). The license is earned through an exam of 40(?) questions, which according to the 4.0 team's Nicholas Chantre was not very difficult and could easily be studied for.

FAA Certification

Once you have the final payload with all systems working, everything finalized and ready to launch. You must find out the final weight of the payload. With this information you are going to use 2 flight calculators.

- Performance Metrics
- Flight Path

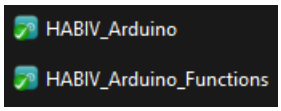
For the performance metrics, the weight and the positive lift are the same, and input the size of the balloon. The flight path is to input the weight, the day and time of launch, launch altitude, and descent rate. Use the burst calculator to calculate the Ascent rate and burst altitude. With these calculations you must call the FAA and send them these flight plans. A contact for the FAA is in the contact sheet. You must contact them 1 week prior to launch. Then again 15 minutes to launch depending on the path you might need to call Albany (ALB), Boston (BOS), and Yankee (BDL) (Bradley-Windsor Locks, CT) ATC (Air Traffic Control) towers. Make sure you put the radar deflector on the payload, so it shows up on Radar. It is attached to the bottom of the payload box.

Project Code Overview

Feel free to change the names of these files but remember to update them in this doc!

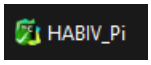
Arduino Code

The Arduino code is an infinite loop that repeatedly queries the sensor array attached to it and transmits this data to the Pi via serial communication. The code is separated into two files:



HABIV_Arduino holds the initialization code, main loop, and globals, while HABIV_Arduino_Functions holds the functions used to query the sensors and format the data into proper text for the Pi to use. These files use the Serial Arduino library heavily, so it may be useful to become familiar with its functions and their nuances before changing this code. The Pi largely runs separate from the Pi on its own closed loop, and does not take responses from the Pi.

Raspberry Pi Code



The main code to use on the Raspberry Pi during flight is HABIV_Pi. Unlike the Arduino code, this runs using multiple threads, each with a specific purpose. For example, one thread handles the receiving of data from the Arduino, one handles almost all functionality related to the GPS module, and one takes care of all radio transmission. The most complex section of the Pi code is the thread for the ETS system.

$$\frac{(x-h)^2}{r_x^2} + \frac{(y-k)^2}{r_y^2} \leq 1.$$

(x, y) = Current lat and long coords of flight

(h, k) = Lat and Long coords of flight starting point

Rx, Ry = Ellipse x and y axis radii

To make a boundary for the payload to fly within, the above equation is used. All (x, y) coordinate pairs that solve the inequality are within the specified ellipse, so when the inequality is no longer true the balloon has flown outside of the boundary. The ETS loop checks repeatedly checks to see if the inequality is untrue, and if it is so for longer than the specified amount of GPS readings, then the ETS system will be activated.

Another tricky part of using the Pi is that sometimes you'll get a USB port error when you try to run the code. This is because the USB devices (GPS, Arduino, radio) are given a name like USB0 based on the order they were plugged into the Pi, and thus it can be pretty variable. However, if you start up the Pi with all the devices plugged in it should assign the same names to them every time (as long as their positions aren't moved ever, probably?). We highly recommend you figure this out so testing can go faster… figuring out how to deal with this issue took a good deal of time.

Base Station Code

The base station code is used for turning both the stepper motor and the linear actuator. Using both Pythagorean theorem, Trigonometry, and the Haversine formula it tells the base station antenna to rotate and point towards the payload. You first need to turn on the payload at the origin to determine the location and set the origin of the base station. Then the payload moves through the air using the base station coordinates as a reference for the rest of the formulas to work and yield and angle to rotate to and an angle to point to. The linear actuator uses the differences in the latitude and longitude to create a diagonal line from the payload to the base station using the haversine formula. The haversine formula is to find distances on spherical shapes like Earth. While Earth is not a perfect sphere (oblate spheroid) this is the best estimate we can give. The it uses Trigonometry to figure out the angle to point to using the Inverse Tangent function. Taking in the diagonal distance found previously and using the difference in altitude to plug into the Inverse Tangent function and it spits out an angle. Which is then taken as an input to the pointing function for the linear actuator and it either retracts 1in or extends 1in for every difference of 9 degrees.

Getting Started with the Raspberry Pi

Before you even get started, first thing you'll need is a computer monitor since the Pi has no screen. If your project space doesn't have any to use, you can use the mini monitor used for setting up at launch sites.

The first thing you'll want to do when you start working on the computer system for the project is to make sure the Raspberry Pi is up to date. To learn how to update the Pi, refer to this link:

https://jamesjdavis.medium.com/how-to-update-raspberry-pi-just-follow-these-easy-steps-ac507cf70238

Make sure to check your version of Python 3 and update if necessary as well, since it is used in the project's code.

The code for the project should currently be location in a folder named hab4.0code or something of the like. Here you'll find the entire git repository for the project. For your safety, DO NOT DELETE THIS FOLDER! It is already set up with git/the repository and setting it up again will be a pain. The Pi comes with a Python IDE installed already, Thonny, which is already completely configured and ready to go.

Using Git on the Pi and Personal Computers

Git is an extremely useful tool for programming, especially in teams. It lets you:

- Store and download code from the cloud
- Version control
- Multiple people can work on code from their own devices
- And more!

We highly recommend learning the basics of git (pulling, pushing, adding, committing, cloning, etc.) before using the repository so that it stays intact and organized. On desktop the software we used to

manage the git repository is called Sourcetree, which gives you a nice GUI to work with and simplifies the process of using git to just clicking some buttons.

However, the Raspberry Pi does not have the convenience of Sourcetree when it comes to git: you need to use the command line to perform operations. If you know the basics of git the commands will quickly make sense to you, but since you're using command line you'll need to do some changing of the current directory before doing so, since the current directory must be the highest folder of the local repository copy. There's a bit of a learning curve to this, but don't freak out if a mistake is made: that's what the version control is for. If you accidentally make a commit that say, erases the whole repository on accident (this did happen once), all you need to do is reverse the repository to an earlier commit. Git is a real lifesaver!

Command Line Inputs for Cloning the Git Repository

```
(make a folder in your home directory named hab4.0code)

cd hab4.0code

git clone git@bitbucket.org:cborsari/hab4.0code.git
```

Command Line Inputs for Pulling from Git Repository

```
cd hab4.0code

git pull
```

Command Line Inputs for Adding, Committing, and Pushing to Git Repository

```
cd hab4.0code

git add --all

git commit -m "[Put Commit Note Here]"

git push -u origin master
```

SSH Key for Repository and Raspberry Pi

In order to make it so the Raspberry Pi could communicate with the Git repository, an SSH key was generated by the Pi and registered with the repository. If you use the same repository as HAB 4.0 you won't need to worry about this, but if you use a new repository you'll probably need to repeat this process for it.

Testing Process (HAB 4.0)

On Campus (WPI Football Field)

In order to test the GPS functionality of the payload and later to test the radio communication between the payload and base station, initial testing was done on the WPI football field. This is a great first location for outside testing since it's close to AK where you'll probably have your project supplies, and it has a good deal of flat land to work with. The following functionality was tested here:

1. Payload GPS reception and data handling

2. Ability of ETS system to cut balloon rope
3. Reliability of ETS activation via crossing GPS boundary
4. Payload to base station communication over short distance
5. Ability of base station to rotate towards payload over short distance

Off Campus (Green Hill Park, Worcester MA)

To do more intensive testing after the first rounds at the WPI football field, the 4.0 team sought a location with both flat ground and hills, with little obstruction so that the radio would not be interrupted. The location settled on was Green Hill Park in Worcester MA, a public park with up to 30m differences in altitude to work with. The following functionality was tested here:

1. Payload to base station communication over long distance
2. Ability of base station to rotate towards payload over long distance
3. Endurance test of payload code (code ran for entire testing period)

Processing Test Data

As the project currently stands, processing the data from testing is a breeze:

1. Change the title of the CSV file of importance so that it stands out from the rest
2. Upload to Git repository
3. Pull Git repository on computer of choice
4. Convert CSV file to Excel Workbook (if it asks, UTF-8 is the format of the CSV file)
5. Graph as you normally would with Excel. Easy as that!

HAB 4.0 had the timestamp of the datapoints as the X axis for almost every graph, with the exception of the Latitude – Longitude coordinate graph. This graph needs some extra formatting for technical reasons given that it's a graph of coordinates. In order to display this data correctly, the graph must have an X to Y ratio of **1.372 : 2**. Otherwise, the travel path of the payload will not apply correctly to a flat image of a map.

Early Termination System Setup

As described thoroughly in the HAB4.0 report the Early Termination System or ETS was added to the payload in order to prevent it from getting lost due to excessive drift into undesired areas. Although the primary goal of this system is to prevent emergencies that would result in a total loss of the payload it is designed to be used during every flight regardless of emergency to shed balloon mass when it is losing altitude before parachute deployment. For this reason, it is essential that the future HAB teams have a concrete understanding of how this system functions and how to perform maintenance on it. This system can really be boiled down into three separate parts. These parts include:

1. ETS trigger circuit
2. ETS Heating Element
3. ETS Code

The goal of this section of the startup guide is to help develop an understanding of how the first two items in that list function and what will need to be done prior to a flight in order to ensure that everything goes smoothly. More information on the third item in the list above can be found in the HAB 4.0 report.

How the ETS works

Without diving too deeply into the details of how this system is designed, which is described thoroughly in the HAB 4.0 report, the basic functionality of the ETS can be described as follows. During every flight the payload location is closely monitored through GPS tracking. Prior to this flight a boundary is set in software which determines all the areas in which the HAB can safely travel without being at risk of blowing into a body of water or any other undesirable areas. In the event that GPS determines the location of the HAB to be outside of this boundary the ETS is triggered. This results in Arduino pin D3 being set to logic high which toggles a MOSFET switching circuit (connected via green wire) allowing current to flow into a Nichrome heating element fixed to the line that connects the payload to the balloon. This heater severs the payload from the balloon allowing it to fall safely from the location at which it was tigered to avoid any further undesirable drift.

During a successful flight, where the HAB does not cross the predefined boundary, The ETS is automatically tigered when the payload has dropped below a specific altitude in order to sever the extra mass of the now popped balloon and clear room for parachute deployment.

Resetting the ETS after use

The only portion of this system that should have need to be reset prior to a flight is the heating element is self. This heating element consists of a metal loop with a piece of AWG26 Nichrome 60 wire, which can be found on amazon if need be) spanning across it. The metal loop is purely structural and provides a method of keeping the Nichrome wire taught and pressed firmly against the balloon line. A picture of this heating element can be seen below.

It is essential to pay close attention to specific elements of this heater to ensure proper functionality. The orange line seen running through the middle of the loop is the line connected the balloon on the top end to the payload on the bottom end. At the end of every flight this line will need to be replaced. The most important things to note here are as follows:

1. Past the point of the nichrome wire there are no zip ties connecting the orange line to any other part of the heater, this must happen as this portion of the line is connected to the balloon and needs to be free to separate from the system once it is cut.
2. The only zip ties that will ever need to be cut are the ones fixing the heater wires to the line below the heater. All other zip ties should not be removed, and replacement line should be fed through the zip tie connecting the line to the metal loop in the center of the image.
3. It can be seen that the nichrome wired is wrapped four times around the orange line in the middle of the loop, it is at this point that the line will be cut. These loops should remain intact and once the old, severed line is carefully removed a new line should be fed through these loops. The loops are fairly tight, but it is essential that excessive pressure is not applied to the Nichrome at any point as it is fragile. It was determined that the easiest way to insert a new line into these loops was the combination of a slow twisting and pushing motion as if you are screwing the line into a hole.
4. Ensure that there is an adequate amount of extra line above the heater for tying it to the balloon.

In the event that the Nichrome wire breaks, which a new line is being fed through it the coil will need to be rebuilt. Details on the exact length of the Nichrome wire as well as its resistivity ect. Can be found in the HAB4.0 report. Sections of blue rubber tape can be seen in the provided image of the coil above; this

is a plumber's tape that tolerates up to 500° F and can be found at hardware stores. A base layer of this tape is wrapped around the metal loop to prevent the nichrome from conducting with the loop. Since the Nichrome needs to be a very specific length to reach the proper temperature the extra nichrome is coiled around the tape on either side of the loop. An additional layer of tape is wrapped over the extra Nichrome wire to minimize the area of exposed heated wire.

Although the heating element is designed to be reused there is always a chance that it could be damaged during a flight or while it is being reset. If this happens what is provided above as well as the ETS section of the HAB4.0 report should provide enough information to rebuild the heating element from scratch.

Resources

Here's some links to some useful sites and diagrams, some of which can be found in the HAB project files. Good luck with your project! - Drew

Project Code Repository

The repository of code used in the project can be found here:

https://bitbucket.org/cborsari/hab4.0code/src/master/

Ask the previous year's team about acquiring permissions to edit or download if necessary.

Connections Chart

A chart which displays all wire connections and communications between modules can be found here:

https://wpi0.sharepoint.com/:u:/s/gr-HABMk.4MQP/ET954J88UeNOkGp-Wdm6d8YB2XJJFUMw94HN0oP3oCn9JQ?e=wIbtDj

Contacts List

A list of contacts used in the 4.0 year's project are included in the link below as well as in the project materials which will be passed on to subsequent years:

https://wpi0.sharepoint.com/:x:/s/gr-HABMk.4MQP/EXHig8hP6edIrMOf8bAKd4oBSHus4PytsAteZjJ9-sjG3w?e=nvQbnn

Bill of Materials

A bill of materials including parts used in the 3.0 and 4.0 year's projects are included in the link below as well as in the project materials which will be passed on to subsequent years:

https://wpi0.sharepoint.com/:x:/s/gr-HABMk.4MQP/EZHVduDI9rtPoenI2Vb63bQBNPsXAFyShb249PrnsCH5ow?e=9e8rcl

Performance Metrics Calculator

A calculator used to get the data performance of the balloon:

http://tools.highaltitudescience.com/

Flight Plan Calculator

A calculator used to get the predicted flight plan of the balloon:

https://predict.habhub.org/