1320 Beal Ave.
Ann Arbor, MI 48109
(734) 764-3310
December 13, 2019

James Cutler, Associate Professor
Department of Aerospace Engineering
University of Michigan College of Engineering
1320 Beal Ave.
Ann Arbor, MI 48109

Subject: Final Report for the Design, Build, Test, and Operation of a High-Altitude Global Ballooning System

Dear Professor Cutler,

Enclosed is our final report for the design, build, test, and operation of a long endurance High-Altitude Global Ballooning System (HA-GBS) that is intended to be used in a new space systems engineering course at the University of Michigan.

The HA-GBS module flies on a hydrogen super-pressure balloon, which consists of four key subsystems: power (input and storage), communications, command and data handling, and payload. The power input and power storage subsystems consists of solar panels and a supercapacitor, respectively. Communication is handled by a radio capable of transmitting on multiple frequencies to a global amateur radio network. Command and data handling is centered on a microcontroller to control key processes on the HA-GBS. Lastly, payloads can consist of a variety of instruments such as temperature and pressure sensors that produce telemetry. According to our simulations and prior relevant works, this system is capable of flying for multiple months around the world. We were able to conduct short-duration laboratory operational tests, as well as one short-duration flight test of a modified system design, which ended pre-maturely due to an in-flight anomaly.

We used the facilities and equipment of the Michigan Exploration Laboratory (MXL) for the design, fabrication, and integration of each component of the HA-GBS, while our final test flight was launched from Hellenberg Park in Monroe, MI. The total cost of this project, including labor costs, was less than $30,000.

We would like to extend our sincere gratitude for your support on this project. We hope that our results will be beneficial to the development of both future revisions of the HA-GBS system and the new space systems course, as well as future research endeavours; positively impacting future Aerospace Engineering students at the University of Michigan. Thank you again for this opportunity.

Sincerely,

Alex Chen Daniel Gu Ethan Prober

Justin Schachter Ronnie Stone

AE 405 FINAL REPORT

High-Altitude Global Ballooning System:
Design, Build, Test, and Operation

High-Altitude Global Ballooning System: Design, Build, Test, and Operation

Authors:
Alex Chen (ahchenx@umich.edu)
Daniel Gu (dangu@umich.edu)
Ethan Prober (etprober@umich.edu)
Justin Schachter (jschach@umich.edu)
Ronnie Stone (ronpires@umich.edu)

Prepared For:
Profesor James Cutler (jwcutler@umich.edu)
Professor Peter Washabaugh (pete@umich.edu)

December 13, 2019

Contents

List of Figures

List of Tables

Acronyms

3V3 3.3 Volt Bus Voltage Rail. 17, 19, 22, 24, 45

5V 5 Volt Bus Voltage Rail. 17, 22, 45

ADC Analog-Digital Converter. 12, 18, 20, 22

AFSK Audio Frequency Shift Keying. 15

AMSAT-LU Radio Amateur Satellite Corporation - Argentina. 10

APRS Automatic Packet Reporting System. 10, 48

ATC Air Traffic Control. 54

CDH Command and Data Handling. 11, 20, 22, 39

COMMS Communications. 11

COTS Commercial-Off-The-Shelf. 9, 11

CPU Computer. 20

DBT Design-Build-Test. 9

DBTO Design-Build-Test-Operate. 9, 38, 55, 56

DMM Digital Multi-meter. 41, 45

EPS Electrical Power System. 11, 17, 22

ESD Electrostatic Discharge. 25

FAA Federal Aviation Administration. 30, 38

FCS Frame Check Sequence. 13

FFT Fast Fourier Transform. 49

FXB Francois Xavier Bagnoud. 48

GPS Global Positioning System. 12, 23

HA-GBS High-Altitude Global Ballooning System. 4, 9, 11, 30, 39, 40, 54–56

HT Handheld Transceiver. 32

I^2C Inter-Integrated Circuit. 22–24, 38, 39

IC Integrated Circuit. 12

IDE Integrated Development Environment. 39

ISP In-System Programming. 39

LED Light Emitting Diodes. 17, 22

MPP Maximum Power Point. 41

MXL Michigan Exploration Laboratory. 9, 26, 29, 54–56

NASA National Aeronautics and Space Administration. 10

NOTAM Notice to Airmen. 32

PCB printed circuit board. 12, 15, 18, 58

PTT Push-To-Talk. 28

PYLD Payload. 11

RF Radio Frequency. 15

RK4 4th Order Runge-Kutta. 51

S3FL Student Space Systems Fabrication Laboratory. 26

SOH State of Health. 12, 29, 55

SPI Serial Peripheral Interface. 22, 35, 38

STP Standard Temperature and Pressure. 53

TNC Terminal Node Controller. 15

UART Universal Asynchronous Receiver-Transmitter. 22, 23, 28, 38

WSPR Weak Signal Propagation Reporting. 10

1 Executive Summary

This final report details the design, build, test, and operation of a long-endurance, high-altitude global ballooning system (HA-GBS) for collecting data at altitudes of 10 km - 20 km and transmitting it to a global amateur radio network over the course of a year-long flight. The purpose of this project is to demonstrate long-term global flight capabilities, to provide a holistic space systems engineering experience, and to demonstrate a technology to be used in a new design-build-test-operate (DBTO) course for the Aerospace Engineering undergraduate curriculum at the University of Michigan. Current University of Michigan design-build-test (DBT) courses provide an adequate learning experience for the fundamentals of vehicle systems engineering, however they do not provide experience with spaceflight hardware and software in flight environments. For example, this includes long-duration operations, robust radio communications, ground systems, embedded electronics design, electrical testing, and low-level programming. This is the void we aim to fill with this project.

We have designed, built, and tested a prototype of the HA-GBS. The HA-GBS module is comprised of four subsystems: electrical power system, communications, command and data handling, and payload. It is lifted to its target altitude of 10 km - 20 km by a hydrogen super-pressure balloon and communicates to a global network of internet-connected stations. We flew our prototype HA-GBS module on a test flight on December 4, 2019. This test was expected to last eight hours, and our goals were to successfully reach the target altitude and transmit data collected from on-board sensors to a global amateur radio network once every thirty seconds. The flight ended prematurely due to an unknown anomaly three minutes after launch. However, we were able to run long-duration bench-top tests that validated the performance of the system's hardware and software. The prototype we flew on December 4, 2019 met all of its command and data handling and payload requirements but did not successfully fulfill all of the electrical power system, communications, or flight vehicle requirements. In total, the vehicle we built met 2 out of 4 high level criteria and 12 out of 19 low-level functional (effectiveness) requirements. We were not able to validate a solar power/supercapacitor power supply in time for the test flight—a modified commercial-off-the-shelf (COTS) solution was flown instead. We were also unable to validate the geofencing functionality as our prototype never traveled far enough to require a change in frequency. Finally, we were unable to manufacture/source a hydrogen super-pressure balloon capable of fulfilling our flight requirements, so we opted to use an MXL-heritage latex balloon filled with helium instead.

We recommend that development continue on this prototype for eventual use in Professor Washabaugh's proposed DBTO course or as a research platform for flight testing sensor payloads. We also recommend flying the same prototype in the Winter 2020 semester in order to continue testing the vehicle both on long-duration flights and on the ground. Further testing in atmospheric chambers and vibration tables is prudent, as design issues overlooked by not running these tests might have been the cause of our flight anomaly.

2 Design and Test of a High-Altitude Global Ballooning System

2.1 Background of Problem

The University of Michigan's undergraduate Aerospace Engineering curriculum currently lacks an introductory space systems engineering class run in a DBTO format for undergraduate students. The current design courses in the curriculum, the airship class (ENGR 100) and the hovercraft class (AE 205), offer valuable systems engineering experiences for freshman and sophomore students, but operate in a closed environment that do not afford students exposure to operating a flight system in a flight environment. Furthermore, the Michigan Exploration Laboratory (MXL) – a research lab at the University of Michigan that designs spacecraft – has recently had trouble recruiting qualified students. To fill both gaps, Professor Washabaugh, who runs the DBT courses, has proposed a new DBTO course that teaches complex systems engineering, embedded systems, and operations knowledge through designing, building, and testing a robust, cost-effective long-duration, high-altitude global ballooning system (HA-GBS). We have been asked to design, build, test, and operate a prototype of the HA-GBS and prove its viability for the proposed course.

2.2 Project Benefits

This undergraduate flight systems class would directly impact more than 60 Aerospace students each year, and an unquantifiable number of people and institutions downstream. The benefits of this class are twofold. First, it would fill a void in the current undergraduate aerospace program at Michigan by exposing students early on to space systems topics such as radio frequency communications (RF), embedded circuitry, power systems, flight software, and operations. This better prepares them to succeed in space systems internships, academic research, and early career positions. Second, this class will serve as a pipeline into MXL for interested and capable students. By designing the HA-GBS system to closely resemble the systems used in MXL for flight projects, students will gain experience with MXL hardware and software, allowing easy integration into the lab should they desire. This is beneficial to students, the Aerospace department, and industry – students expand their technical skillset, the department and MXL receives stronger students, and the industry receives better-trained entry-level space systems engineers.

2.3 Review of Relevant Prior Works

This work is closely aligned with the technology developments being made by NASA, industry leaders and hobbyists to find a suitable platform for long-endurance, long-range, high-altitude craft for surveillance of atmospheric conditions and Earth's surface. [6, 7, 8].

Our work was most closely related to the work of hobbyist Pieter Ibelings of RFSpace and an amateur satellite organization based in Argentina (AMSAT-LU) [18]. Ibelings has built a small global ballooning module to monitor weather data, obtain imagery, and transmit the collected data over WSPR (weak signal propagation reporting) and APRS (automatic packet reporting service) networks. Ibelings has answered questions from the MXL team but is not interested in collaborating on the design or mission. His module, the "Muon Tracker," and imagery captured by it are shown in Figure 1 and Figure 2, below [9]. AMSAT-LU also built a small (25 gram) global ballooning module to monitor weather data and transmit the collected data over the WSPR network, we have not contacted AMSAT-LU to see if there is an oppurtunity for educational collaboration.

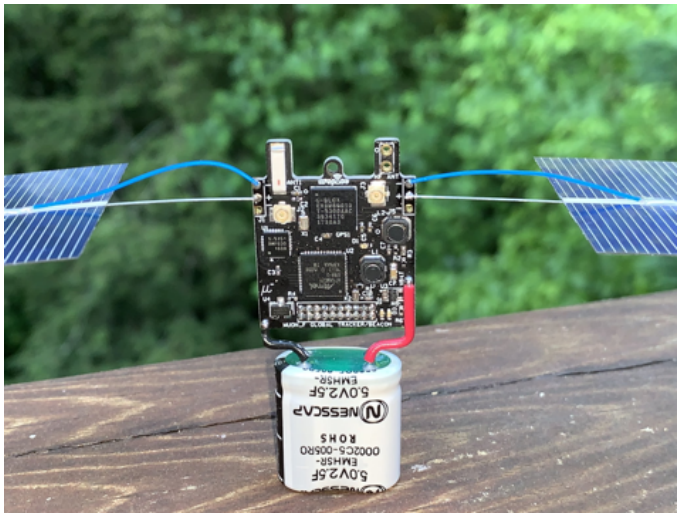


Figure 1: Pieter Ibelings' Muon Tracker with solar panels and a 5V 2.5F supercapacitor supplying power to a microcontroller, GPS, radio, and sensors [9].

Figure 2: An image downlinked with one of Pieter Ibelings' global trackers, demonstrating the feasibility of high resolution imagery on a lightweight global balloon platform [9].

3 The Design: Global Radiosonde Module

The HA-GBS is a system for collecting data at altitudes of 10 km - 20 km and transmitting it to a global amateur radio network over the course of a year-long flight. It is composed of four subsystems: the Electrical Power System (EPS), Communications (COMMS), Command and Data Handling (CDH), and Payload (PYLD). A hydrogen super-pressure balloon is used to lift the system to the desired altitude, but is not explicitly a part of the HA-GBS module. Figure 3 shows a block diagram of the HA-GBS vehicle, outlining the interactions between these subsystems and a list of key features.

Many components of our design are similar to the design of the TrackSoar, a commercial off-the-shelf (COTS) open source APRS tracker [16].

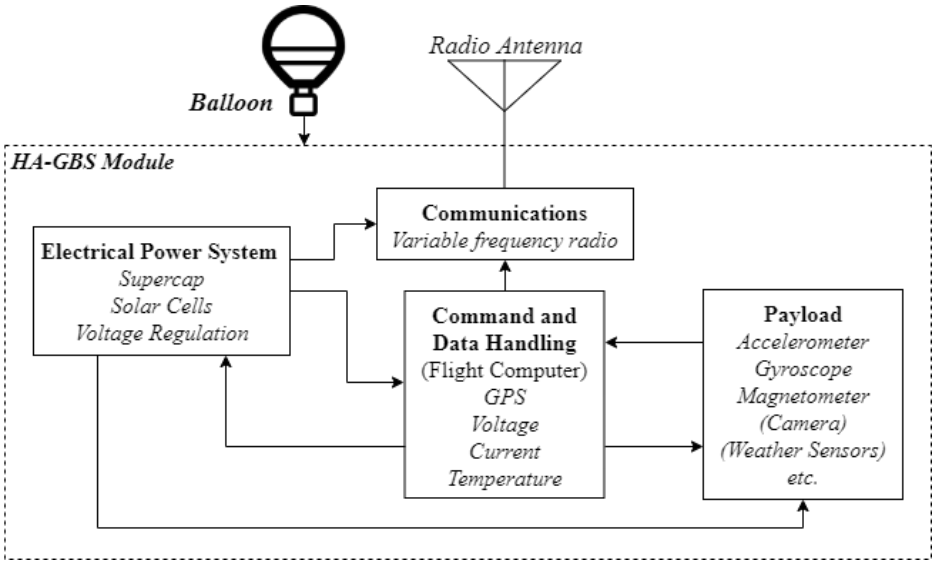


Figure 3: The HA-GBS module consists of four subsystems. It is lifted to the desired altitude by a hydrogen super-pressure balloon and transmits data to an amateur global radio network.

3.1 Radio Communications Link Design

A crucial aspect of aerospace vehicles is the ability to communicate with operators or other vehicles from far distances during flight operations. Normally this communication is done over a radio link between nodes in the system (i.e. point to point). In order to successfully transmit data from one point to another over a long distance, the transmitting radio must be properly designed to take into account the receiver specifications and system environment—antenna specifications, transmit power, data rate, range, atmosphere, etc.

The COMMS system on the HA-GBS takes the data packets formulated by the flight computer and transmits them to all ground stations within line-of-sight. The following sections describe this process in more detail.

3.1.1 Mission Data Flow Diagram

Figure 4 displays the mission data flow diagram for the HA-GBS. State of Health (SOH) telemetry is generated on-board the module during flight to give operators baseline knowledge of the well-being of the system. GPS position and time values are obtained from the UBlox MAX-M8Q Global Positioning System (GPS) module. Critical voltages and currents are obtained by an LTC2309 analog-to-digital converter (ADC) on the printed circuit board (PCB). This allows operators to monitor power distribution in real time on-board the HA-GBS. Temperature is read by an LM20 Integrated Circuit (IC) and allows operators to understand the environment the HA-GBS is operating in.

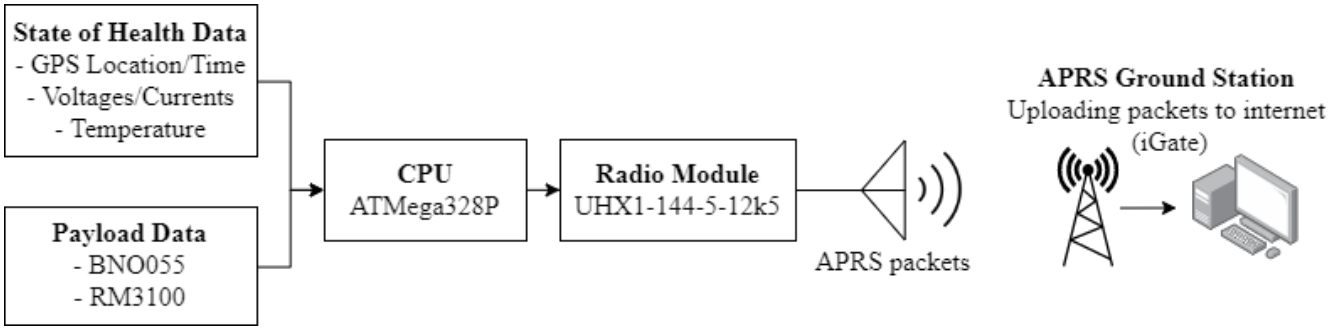


Figure 4: Flow of data through various components of the HA-GBS.

Payload data includes accelerometer, magnetometer, and gyroscope data from the Bosch BNO055 Absolute Orientation Sensor as well as magnetometer data from a PNI RM3100 Geomagnetic Sensor. Sensor data is read from each sensor and is placed into packets by the CDH subsystem, which are then transmitted to an amateur global radio network. This is described in the following sections.

3.1.2 Automatic Packet Reporting System

The communications link between the HA-GBS and operators on the ground is based on the Automatic Packet Reporting System (APRS). APRS is an global amateur radio network, meaning ground stations are distributed around the world and are supported by amateur radio enthusiasts. The system was developed in the 1980's by Bob Bruninga [4]. It is now widely used by high-altitude balloonists, weather reporting stations, and radio enthusiasts to track objects and transmit weather data or other telemetry.

APRS functions as follows: When an APRS packet is transmitted on the proper frequency and in the proper format, ground stations within line-of-sight of the transmitter receive the packet. Multiple ground stations at a single time can hear the packet, and it is possible that the packet is not heard at all if there are no ground stations nearby. Ground stations generally consist of a packet repeater, called a digipeater, which receives, stores, and then re-transmits the packet. Eventually, the packet is received by what is called an I-gate, which uploads the packet to the internet where it is then displayed on the website aprs.fi.

The APRS network frequency varies as a function of region around the globe [4]. Table 1 shows all of the APRS frequencies of different regions.

Table 1: APRS transmission frequencies around the world.

Region	Frequency [MHz]
Columbia, Chile, Indonesia, Malaysia, North America, Thailand	144.390
New Zealand	144.575
China	144.640
Japan	144.660
Europe, Russia, some parts of Africa	144.800
Argentina, Paraguay, Uruguay	144.930
Australia	145.175
Brazil	145.570

3.1.3 Packet Format

The APRS network, being a global amateur radio network, has a very specific set of acceptable packet formats. All of them follow the AX.25 link-layer framing protocol, which at a high level defines the data structure of the packet. Table 2 displays the AX.25 format used for an APRS packet [4].

Table 2: Data contents of the AX.25 frame format that APRS transmissions use.

Section	Data Field	# of Bytes
Header	Flag	1
	Destination Address	7
	Source Address	7
	Digipeater Addresses (0-8)	0-56
	Control Field	1
	Protocol ID	1
Information Field	Data Type Identifier	1
	APRS Data	n
	APRS Data Extension	0-7
	Comment	0-256
Footer	FCS	2
	Flag	1

All APRS packets contain three sections: Header, Information Field, and Footer. The Header starts with a Flag byte (0x7E), which marks the start of the frame. This is followed by routing information that tells a receiving station who/what the packet is meant for (Destination Address), who the packet is coming from (Source Address), what digipeater(s) the packet is to be re-transmitted to (Digipeater Addresses), and what kind of AX.25 packet it is (Control Field and Protocol ID).

The Information Field includes the APRS Data that is being sent. The first byte, the Data Type Identifier, defines the format of the remainder of the section. This is followed by the APRS Data in the proper format that is specific to the data type. Different data types of APRS packets include weather packets for weather data, position with or without timestamp, telemetry data, among others. The size (in bytes) of the APRS Data field depends on the data type, which is why in Table 2 it is denoted "n". The APRS Data Extension field is an optional addition to the APRS Data field, and its size differs with the data type but never exceeds 7 bytes. Finally, the comment is a plain text addendum to all types of APRS Data fields that may contain more useful information.

The Footer contains two bytes of Frame Check Sequence (FCS) that are used by the receiving station to check the integrity of the packet. This is followed by another flag byte (0x7E), marking the end of the frame.

The HA-GBS implementation of the APRS packet is described in Table 3.

Table 3: HA-GBS APRS packet with all data fields displayed.

Data Field	**Example Value**	**Justification**	**Bytes**
Flag	0x7E	Start of frame	1
Destination Address	APRS	Broadcasting to general APRS network	4
Source Address	K6ELD13	Team members' Amateur Radio Technician call sign	7
Control Field	0x03	Declare this an Unnumbered Information (UI) frame	1
Protocol ID	0xF0	Declare no layer 3 (network layer) protocols	1
Data Type Identifier	/	Position with timestamp (No APRS Messaging) packet	1
Timestamp	030119z	Day, Hour, Minute zulu time	7
GPS Latitude	4217.60N/	Degrees, Minutes, hundredths of Minutes, and Direction	9
GPS Longitude	08342.68W	Degrees, Minutes, hundredths of Minutes, and Direction	10
Station Identifier	O/	Declare the station a balloon	2
GPS Altitude	259063,	Altitude [mm]	1-9
GPS Fix Type	3,	Describes quality of GPS lock	2
GPS Satellites in View	5,	Number of GPS satellites the GPS antenna is reading	2-3
ADC Channel 0	1762,	LM20 temperature [mV]	5
ADC Channel 1	1,	Solar input voltage [mV]	3
ADC Channel 2	1,	Solar input current [mV]	3
ADC Channel 3	3377,	5V Voltage (pre-regulation) [mV]	5
ADC Channel 4	59,	5V Current (pre-regulation) [mV]	3
ADC Channel 5	3299,	3V3 Voltage (regulated) [mV]	5
ADC Channel 6	43,	3V3 Current (regulated) [mV]	3
BNO055 Temperature	13,	BNO055 temperature [C]	3-4
BNO055 Accelerometer	2.54,-3.02,-11.92,	BNO055 Accelerometer data in X, Y, and Z axes [m/s^2]	15-21
BNO055 Gyroscope	-0.25,29.19,1.25,	BNO055 Gyroscope data in X, Y, and Z axes [deg/s]	15-21
BNO055 Magnetometer	11.50,14.88,46.19,	BNO055 Magnetometer data in X, Y, and Z axes [μT]	15-21
RM3100 Magnetometer	-15.27,4.47,47.01,	RM3100 Magnetometer data in X, Y, and Z axes [μT]	15-21
Comment	GO BLUE!	Spread the good word of U of M	8
FCS	0x01A4	Verify integrity of packet	2
Flag	0x7E	End of frame	1
Total			149-182

This is an unusual implementation of the Position with Timestamp (No APRS Messaging) APRS packet, as only some of our data lives in the "APRS Data" field from Table 2. In Table 3, fields "Data Type Identifier" through "Station Identifier" make up "APRS Data". The remaining fields ("GPS Altitude" through "Comment") are technically all part of the Comment. This allows us to fit more data into our packet than what is normally allotted.

3.1.4 Radio Frequency Transmission

AX.25 packets are transmitted over radio frequency (RF) using 1200 baud (1200 bit/s) Audio Frequency Shift Keying (AFSK) modulation. This means that binary bits from the packet are converted to 1200Hz and 2200Hz tones (to represent 0's and 1's, respectively) that are overlaid onto a carrier signal. This is done using a software Terminal Node Controller (TNC) emulator written by Santa Barbara Hacker Space [16].

To be courteous to APRS operators within line-of-sight of the HA-GBS, transmissions are spaced in thirty second intervals (2 Hz).

3.2 Electrical Design

The following subsections detail the electrical design of each of the four subsystems that make up the HA-GBS module. Our fabricated (soldered) printed circuit board (PCB) can be seen in Figure 5 where each of the major components of the electrical system are labeled. The top-level schematic that connects each of the lower-level implementations of each subsystem can be seen in Figure 6.

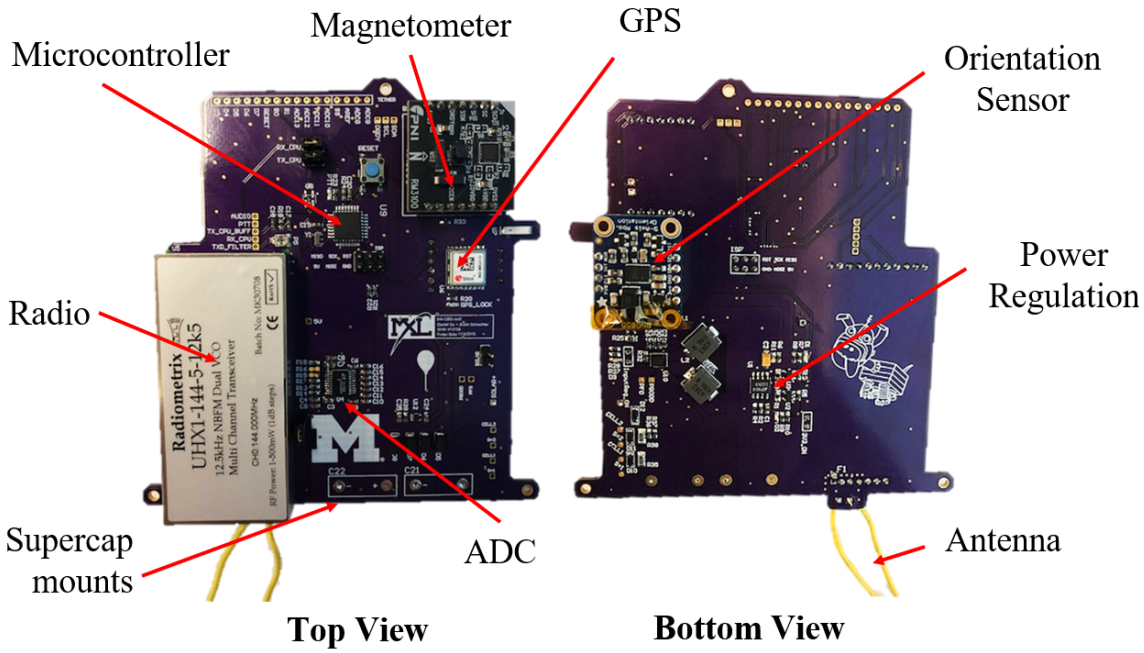


Figure 5: Image of the fabricated HA-GBS module PCB with all electrical components soldered by team members Daniel Gu and Justin Schachter. The electrical power system's supercapacitors and solar panels are not shown.

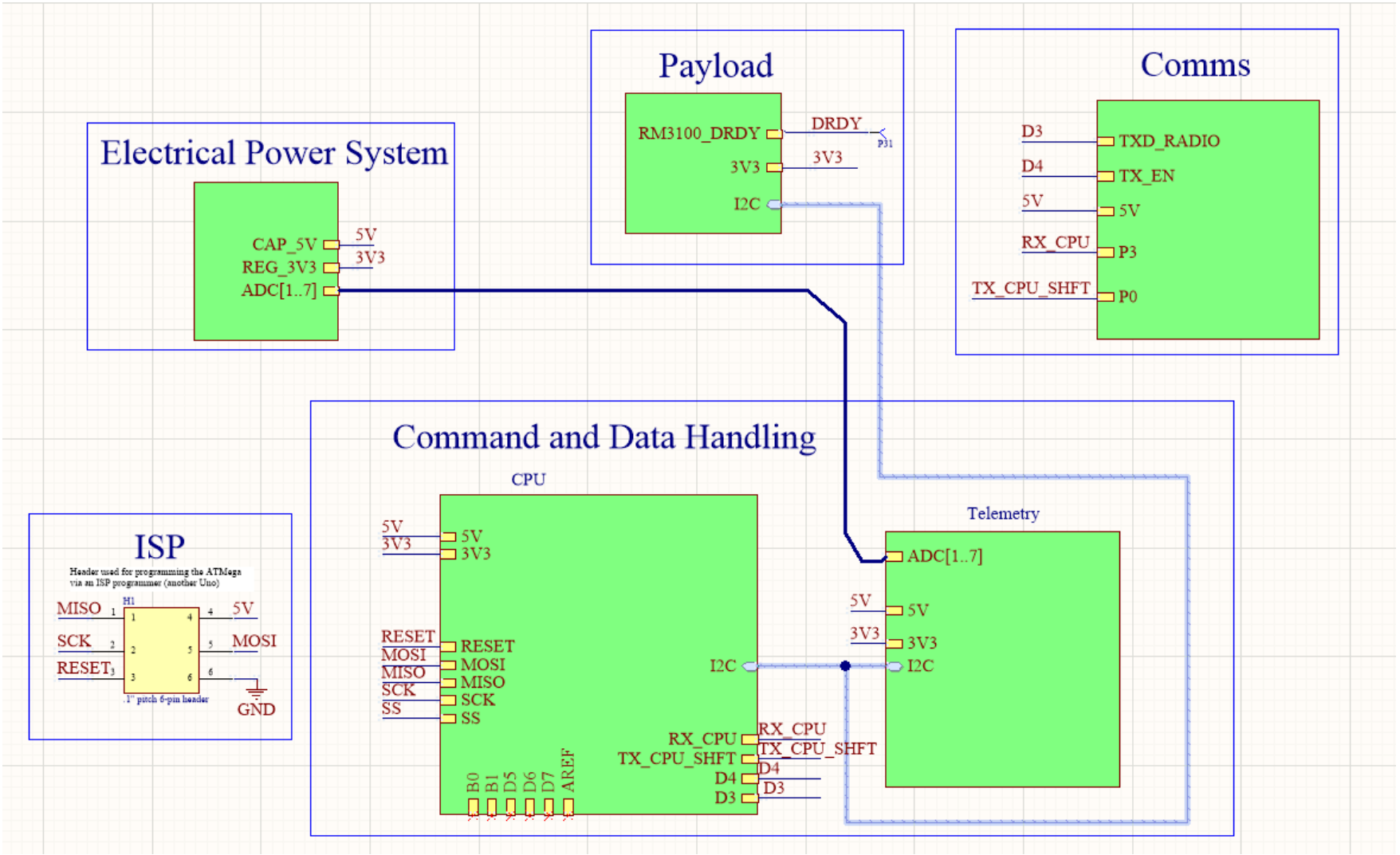


Figure 6: Top-level Altium Designer schematic implemented by the team that shows how each of the subsystems on the HA-GBS module are connected to one another. Also on this schematic is the hardware programming interface (ISP) for uploading the bootloader and flight software.

3.2.1 Electrical Power System

The electrical design of the Electrical Power System (EPS) consists of four main components: solar cells, input regulation, supercapacitors, and output regulation. The connections between these four components can be seen in the EPS top-level schematic in Figure 7. Power is generated from the solar cells, and then passed to the input regulators to charge the supercapacitors. The power from the supercapacitors is then passed to output regulation to generate the 3.3 volt bus voltage rail (3V3) and 5 volt bus voltage rail (5V). Each of the four main components and the two bus voltage rails have light emitting diodes (LED)s and jumpers to allow for isolated debugging.

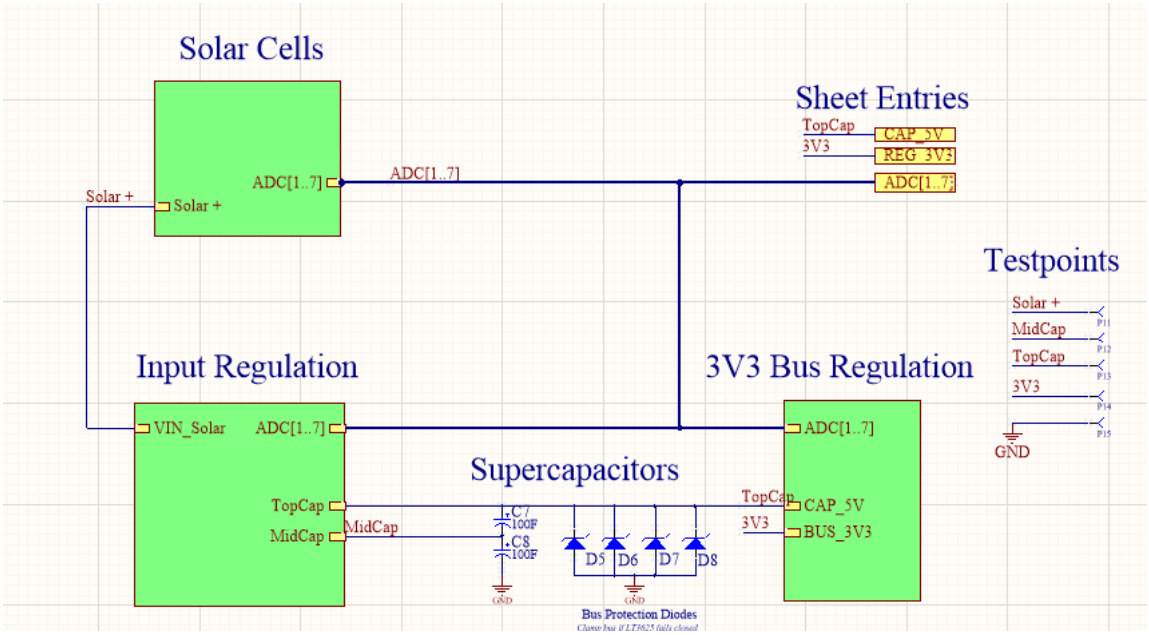


Figure 7: Top schematic for the EPS subsystem, illustrating the flow of power from solar to bus.

An estimated power budget is shown in Table 4. The EPS must be capable of providing this power to the system.

Table 4: HA-GBS Power Budget

Item	**Power [W]**
CDH	0.2
EPS	0.1
Radiometrix UHX1	0.5
Payload	0.4
System Total	1.2
Margin	0.12
Overhead	0.12
Total Required	1.44

3.2.1.1 Solar Cells The solar cells on-board the HA-GBS module provide power generation capabilities to be able to recharge the power storage system during flight. The schematic for the solar cells can be seen in Figure 8. Because the HA-GBS is intended for long-duration, high-altitude flights, an important trade-off to consider in the solar cell design is solar cell mass vs. power able to be generated for a given cell as well as its ability to provide power in different solar conditions —with the understanding that in eclipse conditions the solar cells will not be able to provide power generation capabilities to the system.

In our design studies, we considered 5 different types of solar cells with varying costs, power generation capabilities (voltage and current), and weights. Ultimately, considering the behaviors of the different cells, we decided to outfit

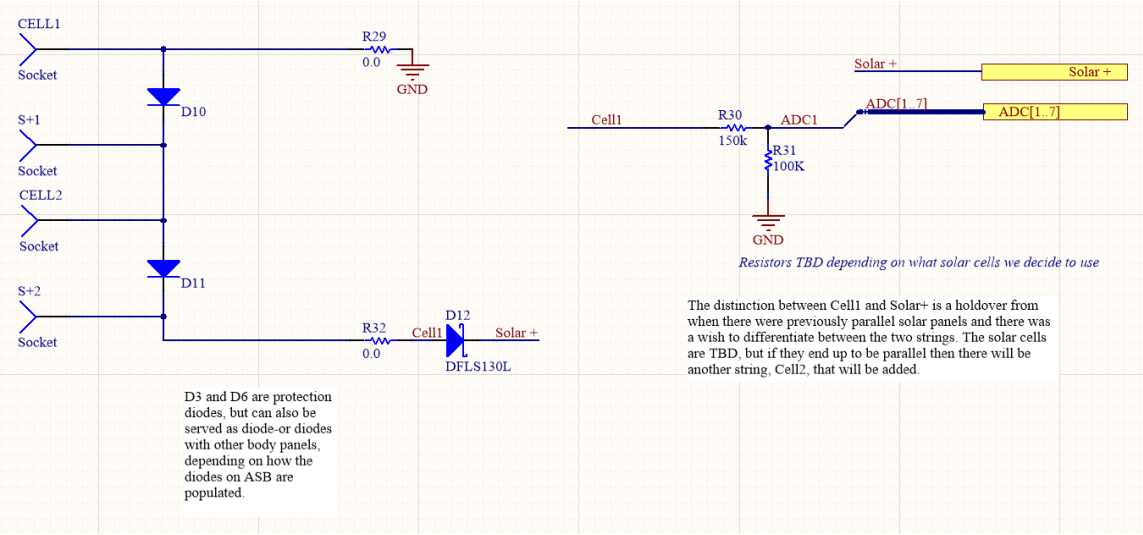


Figure 8: Solar schematic for the EPS subsystem containing many implementations of solar cell protection.

the HA-GBS EPS with 2 IXOLAR SM531K08L solar cells as they were the superior in total mass added to the system as well as power generation. Analysis comparing all of the cells is provided in 6.1.2.1.

On the PCB the solar cells connected are via wire harnesses that are soldered to test points (`S+1` and `CELL1`) which can be seen in the EPS schematics in Figure 7 and Figure 9. An ADC line (ADC Channel 1) is connected here for solar cell voltage monitoring.

For protection, we have bypass diodes (D10, D11) connected in parallel to each solar string —this prevents current from flowing backwards when only a single cell in the series string is shaded. The blocking diode (D12) performs the same function, but for the entire string in consideration of the load.

3.2.1.2 Input Regulation Input regulation takes the power collected by the solar cells and safely charges and balances our energy storage device, two series supercapacitors. The schematic for input regulation can be seen in Figure 9. The main component, the LTC3625 chip, is a 1A high efficiency supercapacitor charger. This converter is capable of charging our supercapacitors while also maintaining the high power draw of the rest of the HA-GBS. It is configured in dual inductor mode, allowing both inductors to serve as the power path for the buck and the boost converters within the LTC3625. This allows them to run simultaneously, greatly reducing the total charge time. They are set 90°apart on the PCB to prevent magnetic coupling, as well as reduce noise for our on-board magnetometer payloads.

Current monitoring on the input of the LTC3625 is provided by a MAX9634, a well established current sensor in MXL heritage.

Unused pin functions of the LTC3625 that were not included in this revision but available for use in future versions are detailed in Table 5.

Table 5: LT3625 Supercapacitor Charger Unused Functionality

Pin on Chip	Function
EN	Allows the charger to be turned on and off. Currently pulled high to keep it always on
PGOOD	Indicates when V_{out} is at its final regulation value and ready to go. It pulls high at 92.5%, and low when it drops below 89.5%
PFI, PFO	These two pins provide input failure notifications if V_{in} drops below a specified value.

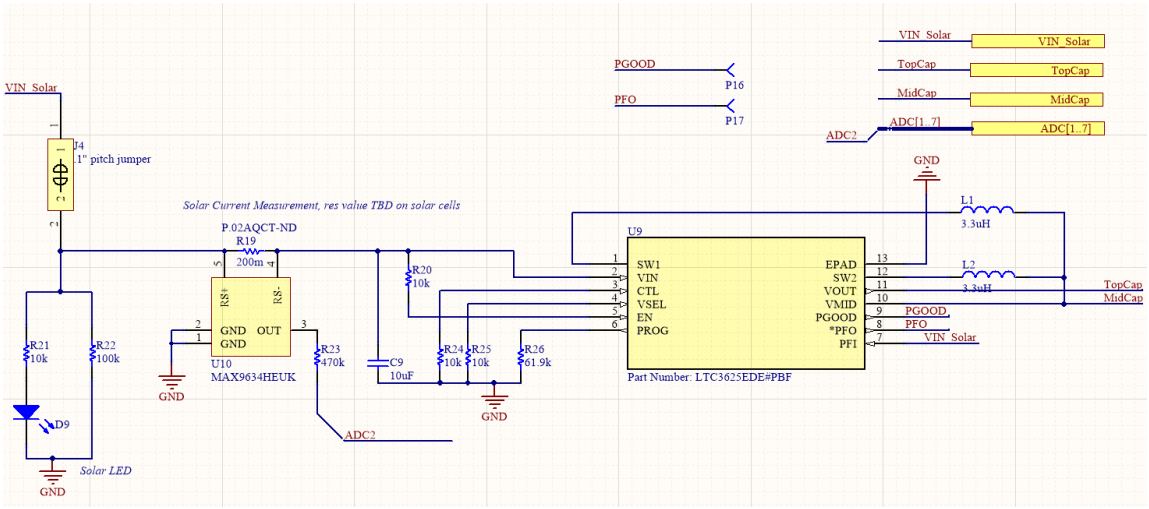


Figure 9: Input regulation schematic for the EPS subsystem illustrating the implementation of the LTC3625 supercapacitor charger, along with associated telemetry points.

3.2.1.3 Supercapacitors For power storage, we chose to use supercapacitors for their high cycle count, high efficiency, fast charge times, wide operating temperature range, and low mass. The schematic for the supercapacitors can be seen in 10. The HA-GBS concept of operations is that it will turn on in daylight, and power down in eclipse when the solar panels cannot charge the supercapacitors. We do not require extremely frequent data collection as the projected lifespan of the mission is in the multi-month range. Downlinking as many transmissions as the flight environment can provide power for will provide operators/students enough insight to predict trajectory and understand parameters about the environment from the payloads.

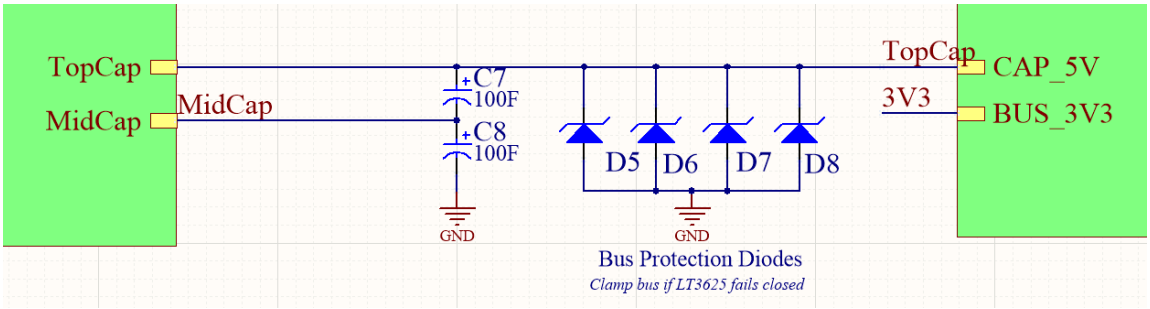


Figure 10: Supercapacitor schematic for the EPS subsystem illustrating the zener array for over-voltage protection.

Though we are flying above the cloud line, there are still chances for passing clouds. The 100-F supercapacitors we are using allow for 7-min of operation in eclipse (without any solar cell input power), as shown in 6.1.2. A small rechargeable battery can be added if nighttime operation is desired, but this will affect system mass and balloon size requirements.

Also included in the schematics are four 5.6V Zener Diodes for over-voltage protection in the event the LTC3625 or LT1129-3.3 fails high. The system is able to operate at 5.7V, albeit handicapped and not advised as this is dangerous for most 5V rated microprocessors (such as the ATMega328P) [5].

3.2.1.4 Output Regulation Output regulation is controlled by a LT1129-3.3 chip, selected for its low dropout voltage of 400 mV at max load. This allows us to regulate a 3V3 voltage rail efficiently from 5V and gives a sufficient margin before regulation failure.

Like input regulation noted in Section 3.2.1.2, two voltage dividers and two MAX9634 current sensors are used here before and after the LT1129-3.3, allowing us to monitor power characteristics of the converter, and calculate

efficiency. Shutdown functionality (SHDN) is also not utilized, but can be if desired in the future.

It is important to note that our 5V bus voltage rail is unregulated from the supercapacitors. This should not be an issue as we are only operating in sunlight, and expect the supercapacitors to be fully charged during nominal operation. Regardless, brown-out characteristics and testing will be performed to ensure the HA-GBS can safely shutdown and recovery in the event of undervoltage/power loss (caused by night-time, prolonged shading, etc).

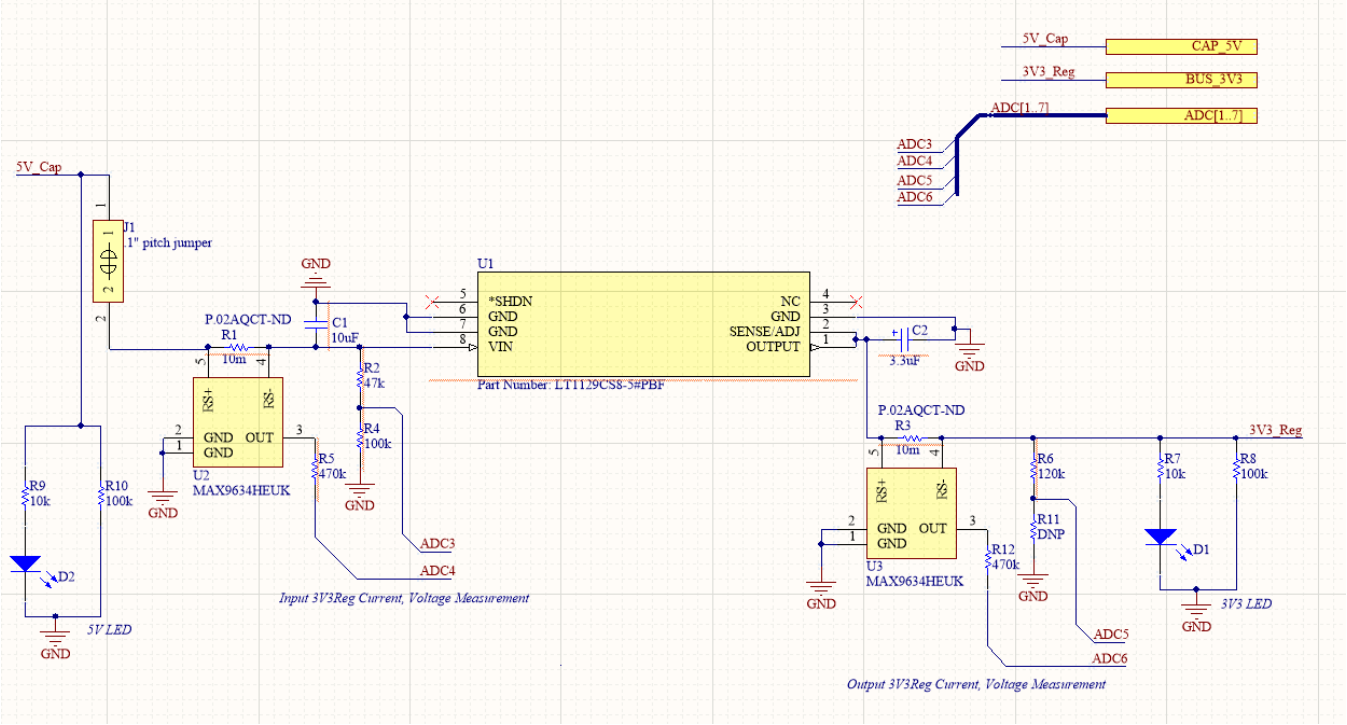


Figure 11: Output regulation schematic for the EPS subsystem illustrating the implementation of the LT1129-3.3 low dropout voltage regulator, along with associated telemetry points.

3.2.2 Command and Data Handling

The CDH electrical design consists of two main, separate schematic sheets: Computer (CPU) and telemetry ADC. The CPU schematic can be seen below in Figure 12 and the telemetry schematic can be seen in Figure 13.

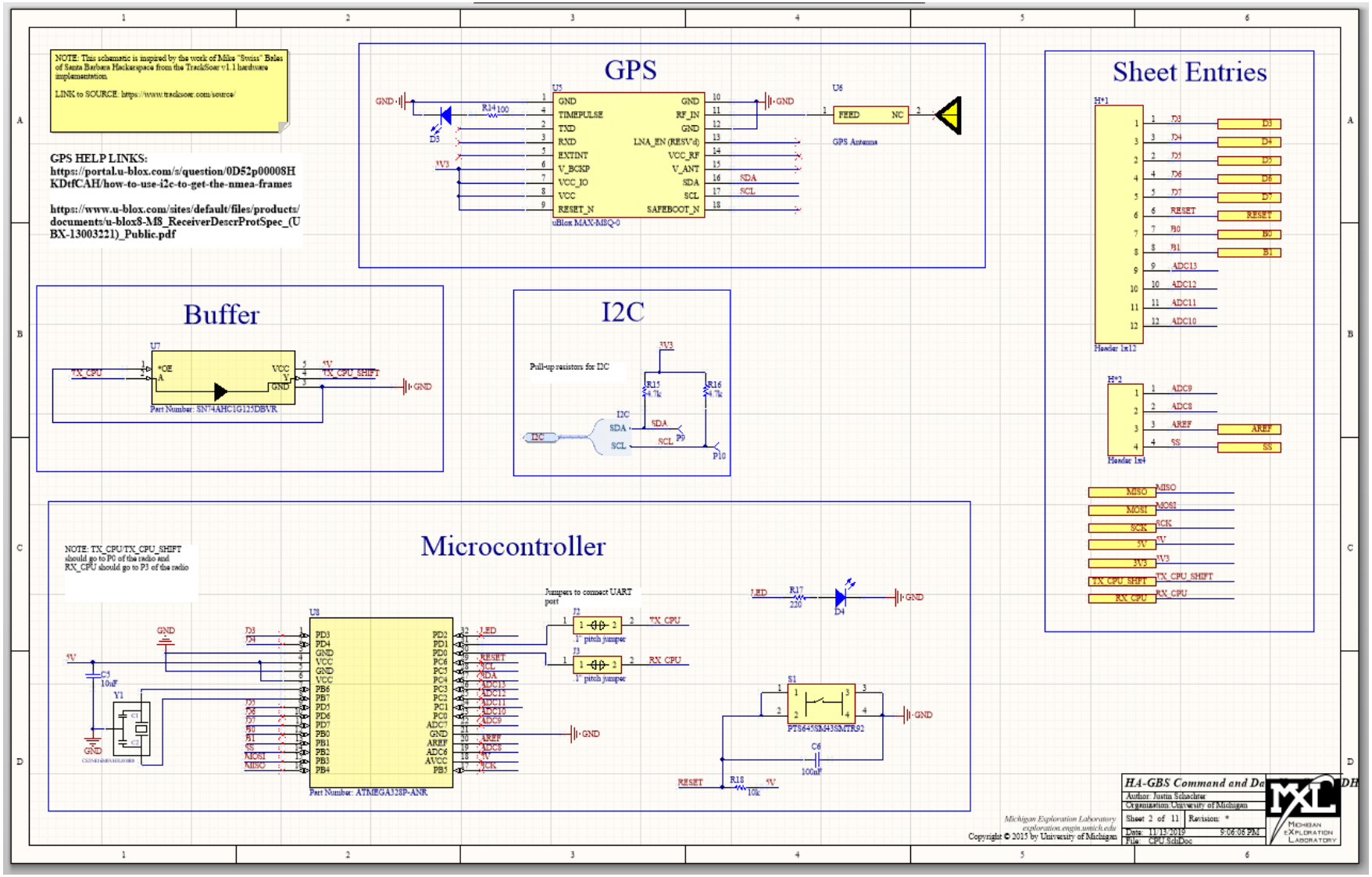


Figure 12: CPU schematic for the CDH subsystem, containing the implementations for the micro-controller, GPS, processor/communication interfaces, and ports for the state of health telemetry ADC

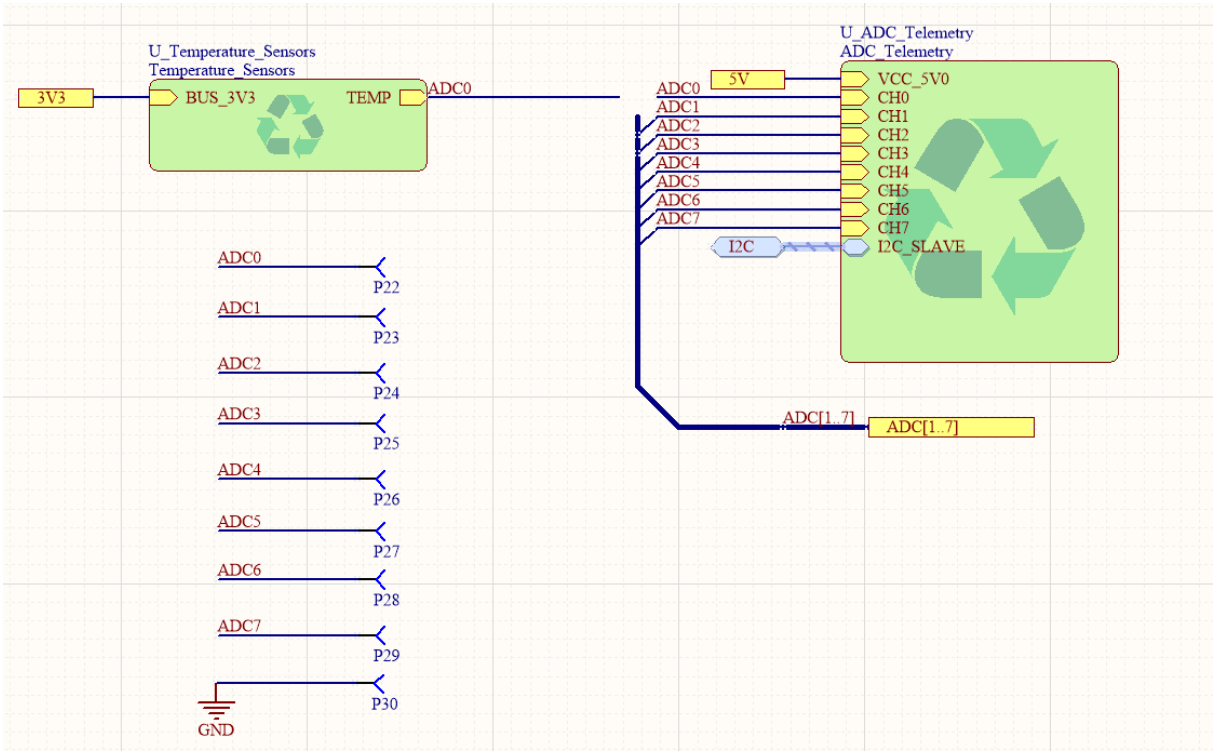


Figure 13: Telemetry schematic for the CDH subsystem, containing the implementations for the state of health telemetry ADC, allowing the rest of the module to have access to this circuitry.

3.2.2.1 Micro-controller

The electrical design work of the CDH's micro-controller was inspired by the work of SBHX from the TrackSoar v1.1 APRS tracker module [16]. Our team's implementation was augmented for our purposes. The schematic for the micro-controller can be seen in Appendix A.1.

It is based on an Atmel ATMega328P processor and every pin on the processor is either used internally to the processor or broken out to the board so that it can be used even if it is not explicitly used in our mission outfitting. All of the communication ports (SPI, I^2C, and UART) have test points and pins that can be used for testing and adding additional hardware to the board, post production. There is a 16MHz ceramic resonator connected to the processor in order to be able to give the processor's digital communications a clock signal. Reset circuitry has been added to give the user a manual reboot option and to give the hardware watchdog timer capability–if software crashes or hangs during flight it will autonomously detect the fault and reset itself. A multipurpose LED has been provided to give the user indication of both startup and COMMS transmissions.

Additionally, on this schematic is a GPS that is included to provide the positioning part of the critical state of health telemetry.

3.2.2.2 State of Health Telemetry

The electrical design for the state of health telemetry portion of the CDH subsystem consists of the implementation of an embedded LTC2309 ADC with various desired analog measurements across EPS and CDH, as well as GPS for positioning (discussed in Section 3.2.2).

The ADC collects information on solar panel input voltage and current; 3V3 rail current and voltage, 5V current and voltage; and board ambient temperature. The ADC communicates over I^2C so that the processor can query its measurements.

The GPS (uBlox MAX-M8Q) is an independent module that collects information on position (3 meter resolution), altitude, and time. The GPS communicates over I^2C so that the processor can query its measurements.

3.2.2.3 Interfaces

The electrical design for the interfaces portion of the CDH consists of both embedded interfaces between modules and external hardware interfaces.

The embedded interfaces are wires that connect the CDH to EPS for power supply and from the ADC for measuring currents and voltages; wires that connect the CDH to payloads for communication with them; and wires that connect the CDH to COMMS to be able to send signals to control the radio.

The external hardware interfaces are either testpoints or header ports that give a human operator access to electrical nets on the module PCB. This gives the design accessible testing features and post production prototyping features. Every communication line and important wire connecting to other subsystems that are not meant to be prototyped have testpoints (large vias). Every unused pin on the processor is broken out to standard 0.1" pitch headers on the board so that the module can be prototyped if additional features are desired such as additional ADC channels or analog/digital port connections.

3.2.3 Communications

The electrical design of the COMMS subsystem connects the Radiometrix UHX1-144-5-12k5 variable frequency radio module to the microcontroller via a serial (UART) connection and to the unregulated 5V power supply. It also includes a low-pass filter, tuned for 2411 Hz, to ensure that the signal inputted to the radio from the microcontroller has a high signal-to-noise ratio.

3.2.4 Payload

The electrical design of the PYLD subsystem connects payload sensors to the microcontroller for data collection and to the voltage rails for power supply. The current design integrates two payloads: a Bosch BNO055 orientation sensor and a PNI RM3100 magnetometer. The payload schematic can be seen in Figure 14.

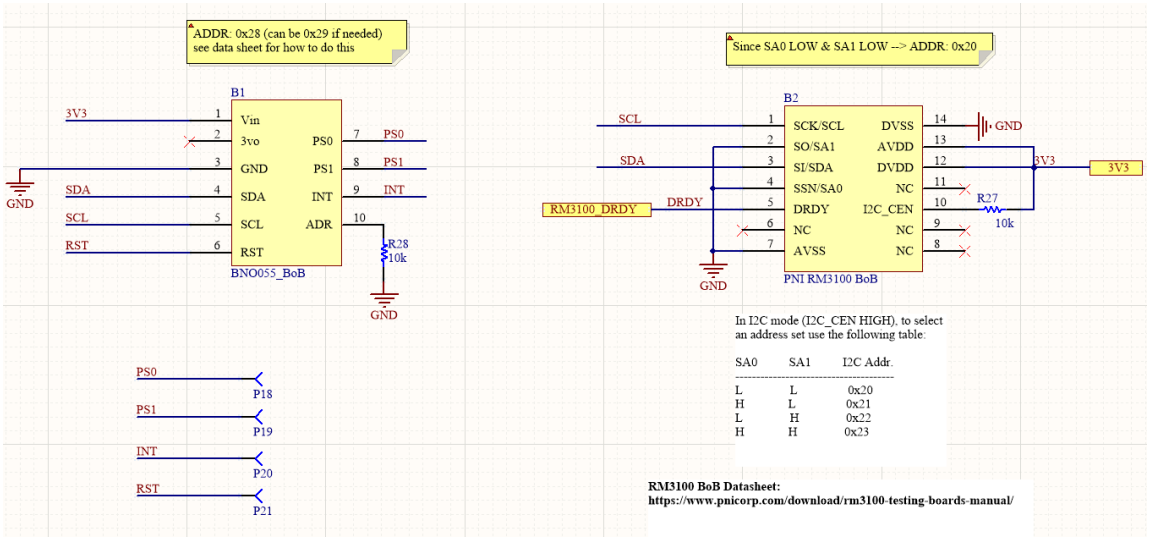


Figure 14: Electrical design of the PYLD subsystem showing the Bosch BNO055 on the left and PNI RM3100 sensors on the right.

3.2.4.1 Bosch BNO055

The Bosch BNO055 is a 10 degree-of-freedom orientation sensor (3 magnetic axes, 3 gyroscope axes, 3 accelerometer axes, and 1 temperature reading) shown in figure 15. It is commonly used in MXL high-altitude ballooning missions since it has a high data output rate. It is powered on a regulated 3V3 rail, and communicates on the I^2C bus . The circuitry was designed to select a valid I^2C address.

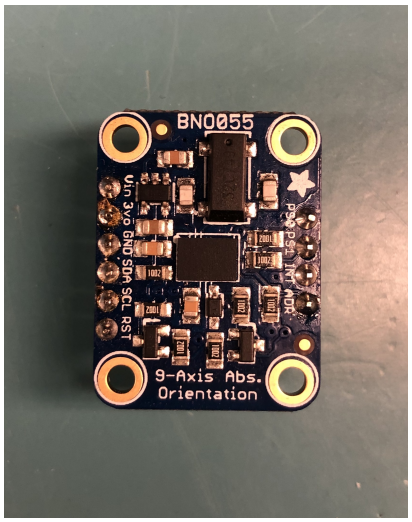


Figure 15: The Bosch BNO055 orientation sensor is a payload included on our prototype HA-GBS and contains an accelerometer, gyroscope, magnetometer, and temperature sensor.

3.2.4.2 PNI RM3100

The PNI RM3100 is a high-resolution 3-axis magnetometer shown in Figure 16, which measures the instantaneous local magnetic field at a given location. This is a sensor MXL is looking to test and gain flight heritage with because its heritage magnetometer, the Honeywell HMC5883, has been declared obsolete by its manufacturer. A high-altitude flight of the sensor, with verification from the Bosch BNO055 magnetic readings, is very valuable to Professor Cutler and MXL, which is why it was included as a payload for our prototype HA-GBS.

It is powered on a regulated 3V3 rail, and can communicate on both SPI and I^2C buses. We implemented circuitry to use the I^2C bus and to select a valid I^2C address.

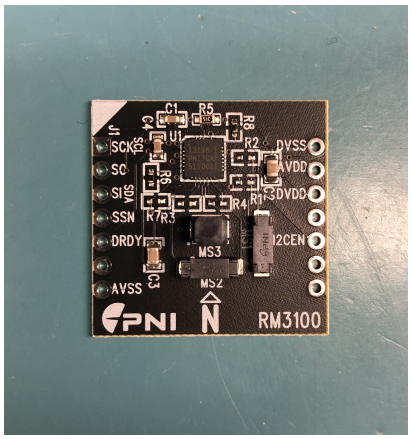


Figure 16: The PNI RM3100 is a magnetometer included as a payload on our prototype HA-GBS.

3.3 Mechanical Design

This section describes the overall mechanical design of the HA-GBS. It can be divided into two main components: the module and the balloon train. The module is the PCB with all of the electronics and operational hardware. The balloon train is the combined system of the module, the lifting balloon envelope, and the tether that connects the module to the balloon envelope.

3.3.1 Module

The module has dimensions of 10.60cm x 10.50cm and a fabricated mass of 79.5 grams (without solar panels and supercapacitors). It is fully independent from the balloon train and is superficially shielded from weather effects, such as dust, moisture, and temperature variations through a conformal coating (thin polymeric film applied before flight). A final layer of weather and electrostatic discharge (ESD) protection is added to the module through heat shrink wrapping or electrical tape.

A detailed 3D-rendering of the module (before any weatherproof coating) is displayed in Figure 17. A labeled, fabricated image of the module is displayed in Figure 5.

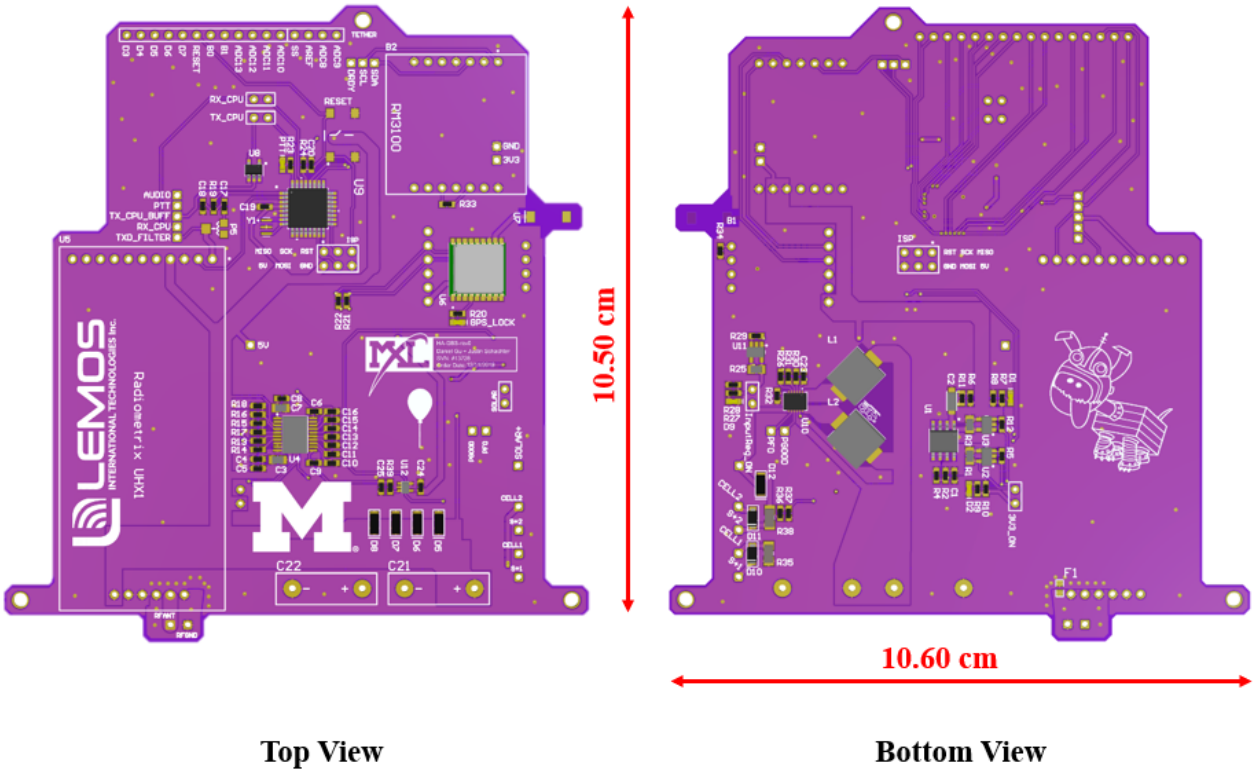


Figure 17: 3D-rendering of the designed HA-GBS module, made in Altium Designer, the software package the team used to design the module.

3.3.2 Balloon Train

The balloon train is the combined system of the balloon envelope, tether, and module that is responsible for lifting the module to the desired flight altitude and sustaining a steady altitude for its flight duration. A diagram of this design can be seen in Figure 18. Since we intended to use a super-pressure balloon for this mission, which MXL has technological heritage with, a number of new design considerations had to be taken into account.

A detailed study on the design of a balloon envelope for the HA-GBS is included in Section 6.1.5. This includes

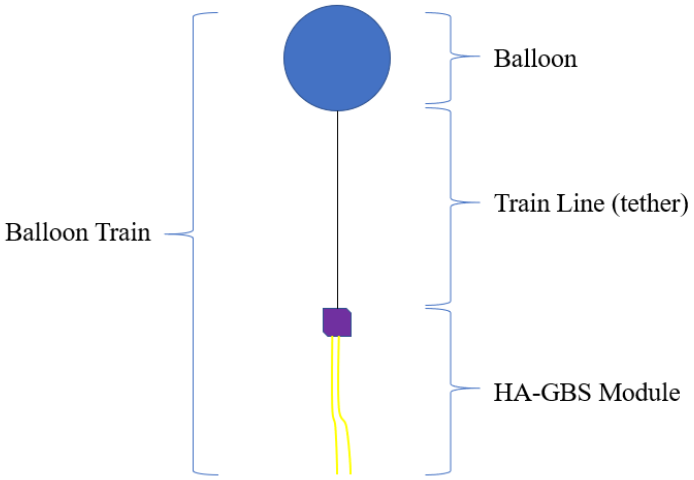


Figure 18: Diagram of HA-GBS balloon train, which is different than most MXL balloon trains since it does not include a radar reflector or parachute since both the balloon and module are sufficiently small by FAA Part 101 standards [3]

simulation of the balloon material and lifting gas interactions with the atmosphere for long periods of time using numerical methods, as well as trade studies on considered materials to be used. Appendix B.1 also includes safety procedures used by MXL for filling stratospheric latex balloons with helium and necessary changes that will need to be made in order to work with other desired gases.

3.3.2.1 Balloon Fabric

A super-pressure balloon is a lifting balloon envelope designed to fly with a positive internal pressure at all times in order to preserve its lifting gas volume. Keeping the volume of the balloon constant is critical for stability purposes, as it allows the system to remain at equilibrium with the surrounding environment and not burst (causing mission termination). However, this equilibrium condition creates a pressure differential that puts the balloon fabric at a significant amount of tensile stress. This requires the balloon fabric to have a high tensile strength and a low elasticity.

Moreover, in order for a long-duration mission to be feasible, the balloon fabric must have a very low permeability coefficient (i.e. a leakage rate coefficient). This ensures that the mass of lifting gas inside the balloon can also be kept relatively constant. Therefore, instead of the traditional latex balloons used for MXL's previous 32 high-altitude balloon flights, the chosen material for this mission was a thin coated polyethylene film, which has a high maximum tensile strength of 200 MPa and a permeability coefficient with an order of magnitude of $10^{-8}\left[\frac{scc\ cm}{cm^2\ s\ Bar}\right]$.

However, depending on mission objectives, a latex balloon, or any other type of flight envelop, can be used for shorter, experimental flights or for other purposes.

3.3.2.2 Lifting Gas

Historically, helium has been the main lifting gas for Michigan's ballooning and airship systems (MXL and the Student Space Systems Fabrication Laboratory, S3FL). This is due to its low density relative to air and high operational safety (non-toxic and non-flammable). However, it is a non-renewable noble gas, which makes it not eco-friendly and relatively expensive when compared to other gases in the market. Therefore, the chosen lifting gas for this mission was hydrogen, which produces approximately 10% more lift and costs on average 2.5 times less.

3.4 Software Design

The software for the vehicle runs on the microcontroller and controls the processes that occur on-board during flight. It is written in C++ and consists of seven main components: the main script, data handler, and drivers for the radio, GPS, BNO055, RM3100, and LTC2309. The functionality of each of these components are described in the subsections below. Figure 19 contains a flowchart that shows the details of how our software architecture functions when all of these components are put together into a single system.

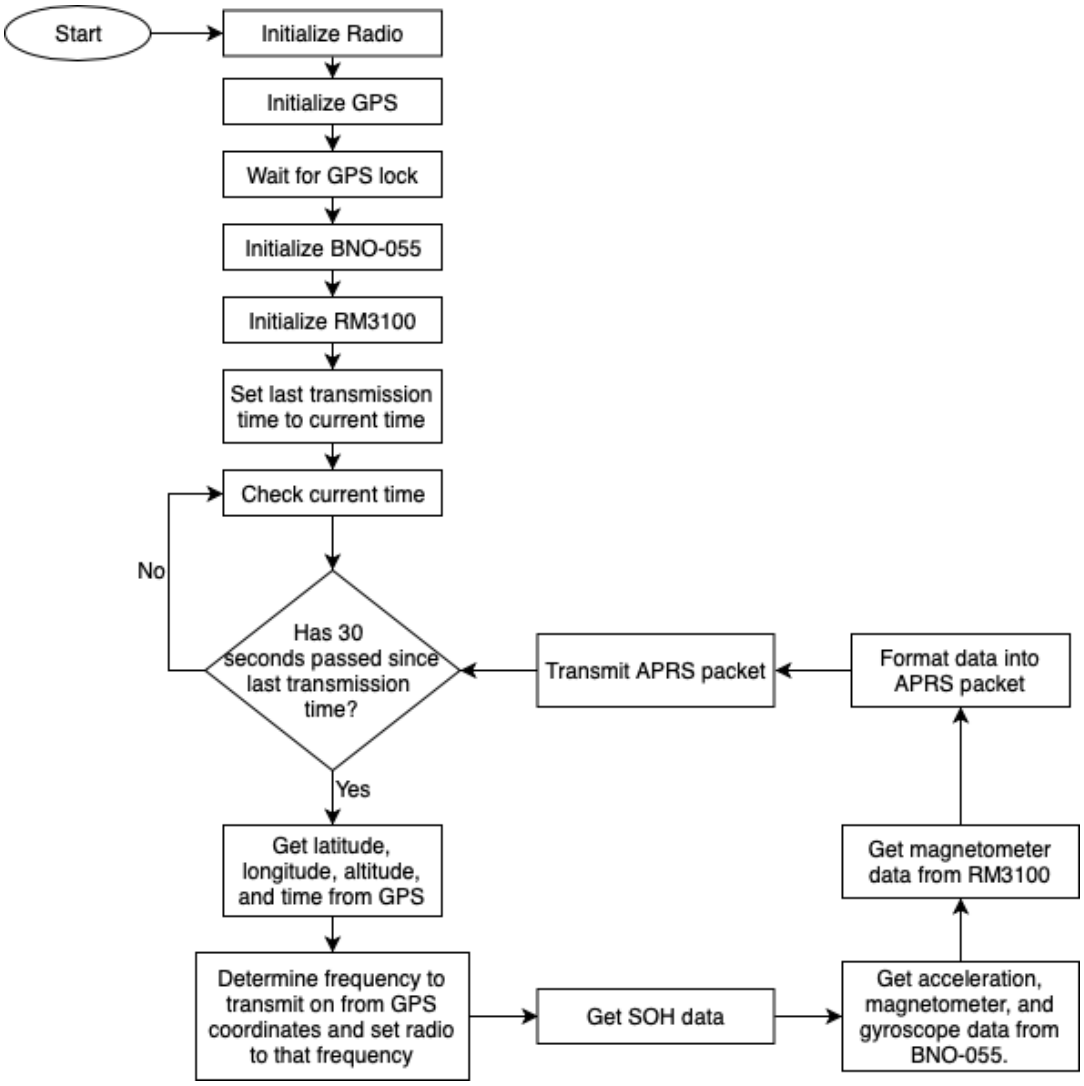


Figure 19: The vehicle is programmed to initialize all components on startup and send packets with data collected on-board once every 30 seconds.

3.4.1 Main Script

The main script is derived from Tracksoar code and contains two main functions: the setup and the loop. The setup function is run once when the microcontroller first starts up and initializes the watchdog timer (see 3.4.1.1 for more details), the I^2C bus with the microcontroller as master, and each of the microcontroller's five peripherals: the radio, GPS, BNO055, RM3100, and LTC2309. After the setup function finishes running, the microcontroller will continuously call the loop function in an infinite loop. The loop function, each time it is called, resets the watchdog timer, checks to see if 30 seconds have passed since last transmission, and performs data handling if it has.

3.4.1.1 Watchdog Timer

The watchdog timer is used to guard against the software hanging or freezing during flight. If this were to happen during flight without a watchdog timer implemented, the microcontroller would become unresponsive and there would be no way to fix it, potentially ending a mission. To remedy this, we run a watchdog timer on a separate processor thread, which is periodically reset as the code runs. If the watchdog timer is not reset after a certain amount of time, usually caused by the code hanging, the watchdog timer will kill the thread running the main code and reset the microcontroller. We set our watchdog timer to activate after eight seconds, and to reset every time the loop function runs successfully. The loop function takes at most two seconds to run (from experimentation with our system), including a transmission. This means that if the watchdog timer reaches eight seconds, it is very likely that the software has hung and the microcontroller needs to be reset.

3.4.2 Data Handling

The data handling component of the microcontroller software is also derived from Tracksoar code and is responsible for creating and transmitting the data packet. It collects the relevant data from the GPS, BNO055, RM3100, and LTC2309 using functions from their respective drivers. The data is then formatted into an APRS data packet described in section 3.1.3. Once the packet is created, the Push-to-Talk (PTT) pin on the radio is enabled, the data packet is transmitted, and the PTT pin is disabled.

3.4.3 Drivers

The microcontroller software requires drivers for the radio, GPS, BNO055, RM3100, and LTC2309 because each of these peripherals have their own unique methods of interfacing with devices connected to them. Drivers abstract these methods into an easily understandable software library so that the data handler or any other part of the software can interact with these devices without knowing the complexities of how these methods work. For example, the GPS driver provides functions to retrieve latitude and longitude data that can be called without the user of the function knowing the exact commands that are sent to the GPS to produce these result.

3.4.3.1 Radio Driver

The radio driver provides functions that allow for control of the radio by communicating with it through UART. It includes functions for setup, setting each of the 16 available channels, choosing a channel to transmit on, enabling PTT, and disabling PTT. The radio transmits when PTT is enabled and stops transmitting when it is disabled. This driver was written from scratch since there was no open source or MXL heritage code to draw from.

3.4.3.2 GPS Driver

The driver for the Ublox MAX-M8Q GPS was open source code and we used it without any modifications. It provides functions to setup the GPS as a slave on the I^2C bus and retrieve latitude, longitude, altitude, and time data.

3.4.3.3 BNO055 Driver

The driver for the BNO055 orientation sensor was open source code and we used it without any modifications. It provides functions to setup the BNO055 as a slave on I^2C bus and retrieve measurements from the temperature sensor, accelerometer, gyroscope, and magnetometer.

3.4.3.4 RM3100 Driver

The RM3100 driver we used was a heavily modified version of an open source driver originally written for the PIC32MX microcontroller. It was adapted to work on our Atmega328P microcontroller and allows us to request and retrieve magnetometer readings with the RM3100 as an I^2C slave.

3.4.3.5 LTC2309 Driver

The LTC2309 driver was written from scratch while referencing an MXL heritage LTC2309 driver written for the MSP430 microcontroller. It sets up the LTC2309 as an I^2C slave and allows us to read values from each of its eight channels, seven of which we use to collect SOH telemetry.

3.5 Future System Capabilities

A major challenge in designing the HA-GBS at the top-level was that since this project was proposed to be an actual flight vehicle, our first implementation would not be the last. Therefore, it was required that the system be designed in such a way that allowed our team, or future students, to rework certain elements (changing out blocks on the block diagram) without needing to redesign the entire system. Certain aspects of the design are stand-ins for capabilities that were not implemented given our team's time constraints and technical capabilities. The system block diagram in Figure 3 and its lower-level children show that the module was designed using classical systems engineering practices to be accessible to different implementations.

The current design of the HA-GBS cis optimized for a successful technology demonstration within the scope of the AE 405 design course. This means that not all of the desired components and capabilities are implemented in this version.

4 Flight Test

On December 4, 2019, we launched a prototype version of the HA-GBS module from Monroe, MI at 10:17 AM on a flight intended to last eight hours. The purpose of this test was to prove operational capability of the HA-GBS in its flight environment.

4.1 Pre-Flight Modifications

Due to time constraints related to the conclusion of the AE 405 course at the end of the Fall 2019 semester, certain design points were modified so that our team could launch before the poster session on December 5, 2019. We were unable to verify that our solar panels could reliably charge the supercapacitors, as MXL's solar emulator usually used to test solar panels was in an unusable state for the duration of the Fall 2019 semester. As a result, a battery pack of 6 AA batteries (three series, two parallel) was manufactured to power the 5V bus voltage rail at 4.8V for the intended duration of the flight instead of our solar panel/supercapacitor electrical power system described in 3.2.1.

Additionally, we were unable to manufacture or source a hydrogen super-pressure balloon capable of fulfilling the requirements outlined in Section 5.1.5 in time for this flight. Thus, a latex balloon from Kaymont Consolidated Industries filled with helium was used instead as MXL has extensive flight heritage with these balloons.

As a result of these design changes, several other additions to the flight vehicle were required to fulfill the requirements in Sections 5.1.5.2 - 5.1.5.4. These included an insulating shell around the HA-GBS module, as the AA battery pack is more susceptible to cold temperatures in the operating environment than the supercapacitors would have been. Due to the increase in system mass from the insulation and the battery pack, a larger latex balloon was required to lift the system (2000 g from originally planned 600 g). A parachute and radar reflector were also

added to the payload train to more closely follow FAA federal regulations now that we had a much larger flight vehicle.

Other changes included soldering a switch in the battery pack circuit so we can easily power on the vehicle at the launch site, and the use of hand warmers in the insulation to further keep the batteries warm during flight.

Figure 20 shows the final configuration of the HA-GBS prototype flown, and Figure 21 shows the final configuration of the full train during balloon fill.

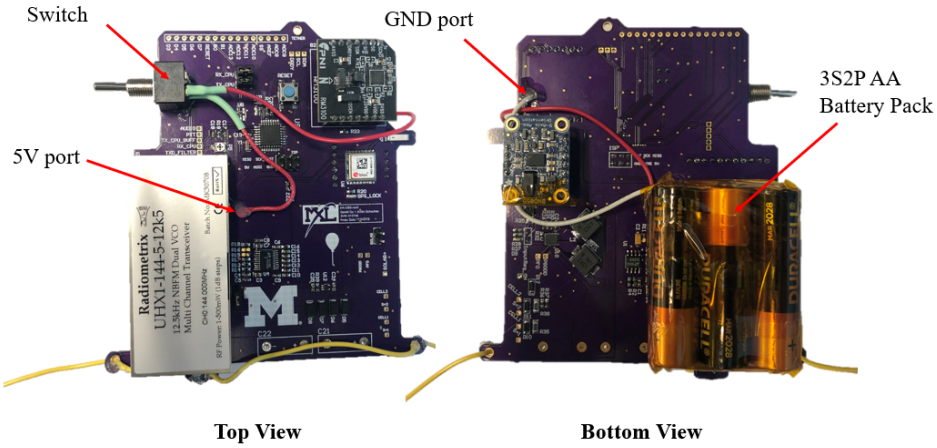


Figure 20: HA-GBS module with pre-flight modifications installed on December 3, 2019.

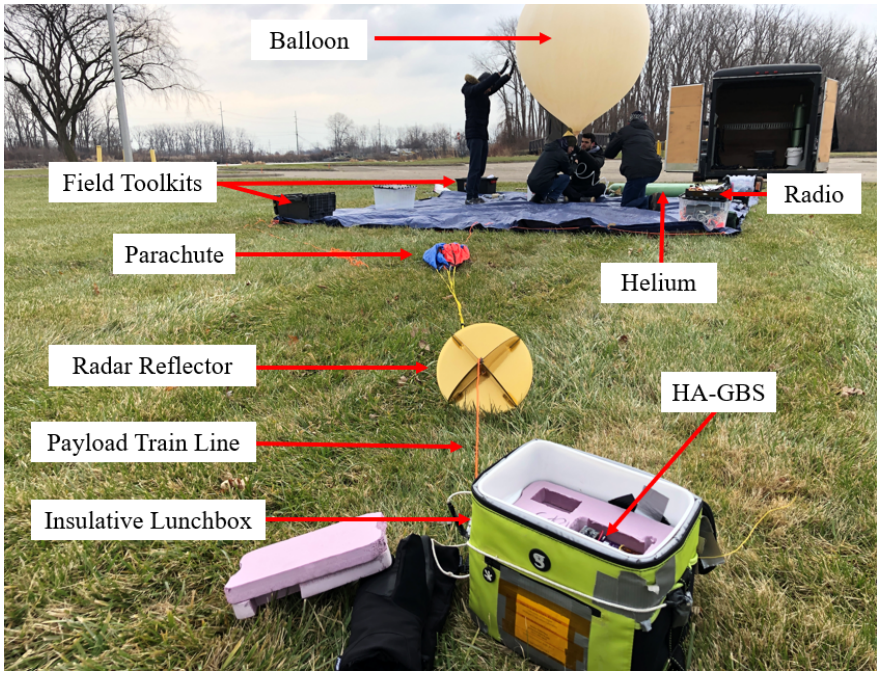


Figure 21: HA-GBS team filling the latex balloon prior to flight in Monroe, MI. Labeled are key components of the field operation for filling the balloon, setting up for flight, and the actual vehicle/payload train.

4.2 Pre-Flight Procedure

The procedure for this test flight was drawn from the MXL heritage high altitude balloon flight procedure. This is described in more detail in the following sections.

4.2.1 Flight Simulations

About 12 hours prior to launch, flight simulations were run using the ASTRA High Altitude Balloon Flight Planner [13]. Our desired launch site was Peach Mountain Observatory in Dexter, MI. However, weather patterns for the time of launch caused our predicted flight path to pass almost directly over Detroit Metropolitan Airport (DTW). To avoid issues related to flying a balloon in the Class Bravo airspace surrounding DTW [2], we decided to move the launch site to Hellenberg Park in Monroe MI, as the southeasterly winds would take the vehicle away from nearby air traffic.

Figure 22 shows our predicted flight path from Hellenberg Park.

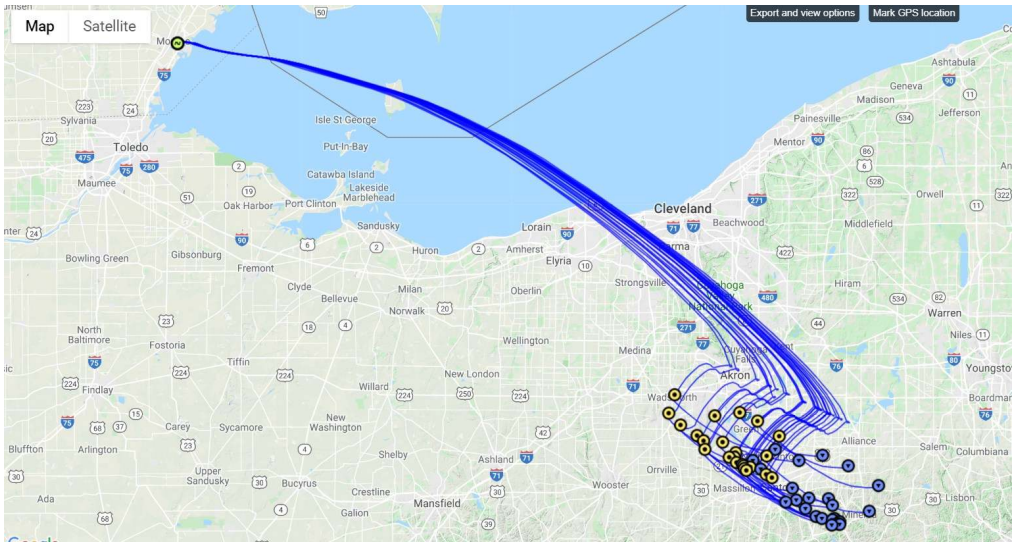


Figure 22: The vehicle's predicted flight path, generated using the ASTRA High Altitude Balloon Flight Planner [13]. Boundary conditions included 1850g of neck lift and launch at approximately 10 AM from Hellenberg Park in Monroe, MI.

4.2.2 Launch Logistics

A large amount of support equipment is required for a high altitude balloon launch. Appendix B.1 contains the full MXL flight procedure checklist, which includes inventory checklists for supplies and equipment to be brought to the launch site. All items were loaded and secured in a trailer approximately 4 hours prior to launch, including 3.5 K-sized tanks of helium.

The payload train, including the parachute, radar reflector, and HA-GBS module inside insulation, was assembled prior to departure for the launch site. Thus the only steps required at the launch site were to fill the balloon with the correct amount of helium, tie off the neck using a combination of zip ties and duct tape, close the battery circuit by flipping the switch, and release the balloon. Figure 23 shows the balloon being filled with all five members of our team present.

Figure 23: The HA-GBS balloon was filled from helium tanks that were brought to the launch site.

4.2.3 Notice to Airmen

According to FAA Regulation Part 101.1 Sub-Part D [3], we were required to notify Air Traffic Control (ATC) in the region of our flight plan. This is done by filing a Notice to Airmen (NOTAM), which gives an alert to aircraft operators in the region that we will be operating in the vicinity. The NOTAM was filed by calling the hot-line approximately three hours prior to launch.

4.3 Launch and Loss of Contact

We released our prototype HA-GBS on a helium balloon at approximately 10:17 A.M. (EST) and maintained visual contact with it, as shown in Figure 24, until it disappeared in the clouds. At first, we were able to receive transmissions on our HT as expected and the APRS network received a data packet at 10:19:05 A.M. However, after receiving one last packet on the HT at 10:19:35 A.M., we lost contact with the HA-GBS and did not receive any additional transmissions. In total, we received five packets on the HT and just one on the APRS network before losing contact, after launch. In total, we received 36 packets, starting from when we turned the HA-GBS module on at 9:58 A.M. while it was still on the ground.

Figure 24: The HA-GBS could be seen rising on its helium balloon on its flight test until it was obscured by clouds.

4.4 Failure Analysis

Our team had several hypotheses for why we lost contact with the HA-GBS. Each is described and analyzed in the following subsections.

4.4.1 Overfilled Balloon Hypothesis

When filling the balloon, we periodically remove the helium supply and temporarily tie off the neck of the balloon, attaching it to a ballast to determine how much more or less helium is required before release. The calculation to determine the mass of the ballast should only take into account the mass of the payload train, plus a net lift margin to produce acceleration upwards.

After releasing the HA-GBS, our team realized that we took into account the mass of the latex balloon itself in our ballast mass calculation. Therefore, when the balloon became buoyant in lifting the ballast, it was actually lifting around 2000g more than anticipated. This overfilled state of the balloon caused it to rise much faster than anticipated. We see this reflected in the ascent speed gathered from the data we received before losing contact, shown in Figure 25.

Simulations conducted after the test flight showed that with the amount that we inflated the balloon, the balloon should have burst at approximately 30,000 m. However, the last transmission we received on the HT indicated an altitude of 977 m as measured by the GPS on-board. Additionally, had the balloon burst at that altitude, we would expect the parachute to slow the descent down enough such that the HA-GBS would be able to transmit at least once while falling. We received no transmission either on the HT or on the APRS network indicating a decline in altitude. Therefore, we consider this hypothesis unlikely to have ended the mission at the time we lost contact.

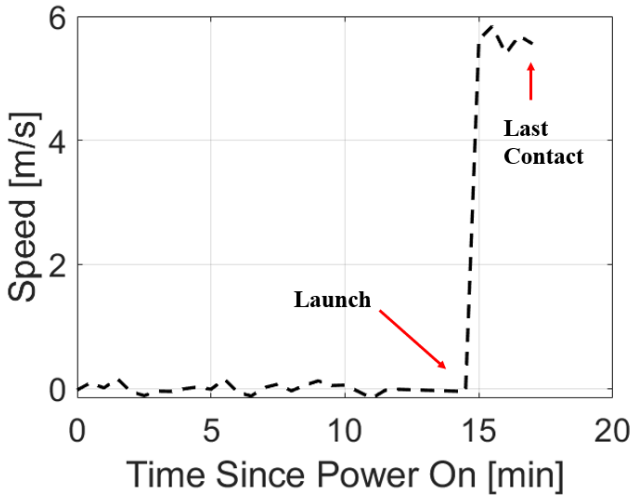


Figure 25: Ascent rate data gathered from the flight test shows launch at just before 15 minutes after the HA-GBS was powered on and a higher than expected ascent rate of approximately 6 m/s before losing contact.

4.4.2 Switch Flip Hypothesis

As described in Section 4.1, an electrical switch was soldered into the battery circuit to power on the HA-GBS at the launch site. It is conceivable that at some point around 3 minutes after launch, rapid movements caused something (possibly the hand warmers that were inserted for heat) to flip the switch to the open position, terminating the operation of the HA-GBS. However, there is no way to determine if this hypothesis is correct.

4.4.3 Conclusion of Failure Analysis

Since our prototype HA-GBS was unrecoverable and the small set of data packets that we received is not very illuminating, it is impossible to determine the actual cause of why we lost contact with the vehicle after just three minutes. For future flight tests, we plan on conducting more rigorous testing of the vehicle in flight environment conditions while on the ground to mitigate the risk of another mission failure.

5 Assessment Criteria

The success of the HA-GBS design is assessed on four major criteria: effectiveness, feasibility, desirability, and affordability. The effectiveness criterion includes all of the functional requirements for all components of our system, which makes it the most important criterion.

5.1 Effectiveness: Functional Requirements

We have defined a set of functional requirements for the different subsystems within the HA-GBS. If a subsystem fulfills each requirement then it can be considered effective. If all subsystems are considered effective, then the HA-GBS overall design can be considered effective. There are five requirements for the CDH subsystem, six for the EPS subsystem, four for the COMMS subsystem, and two for the PYLD subsystem. There are also two vehicle operations requirements, which encompass miscellaneous requirements for flight. These requirements are used as metrics of success for the effectiveness criterion. They are explained in more detail in the following sections. Our system is evaluated in section 6.1 for its effectiveness.

5.1.1 Command and Data Handling

The CDH functional requirements are described in Sections 5.1.1.1 - 5.1.1.5.

5.1.1.1 Communication Buses

The CDH shall have SPI, I^2C, and UART communications buses. Most embedded hardware sensors communicate on some combination of SPI, I^2C, and UART, and only in rare cases just one. If the CDH does not have any one of either SPI, I^2C, or UART communication buses, the inherent system design will not be able to accommodate any plug-and-play payload or state of health telemetry sensor possible in a particular mission's customized module sensor outfitting, thus making it a constrained design, which is not effective for the module.

5.1.1.2 Hardware Programming Interfaces

The CDH shall be user programmable via a hardware interface. Embedded computing processors have an ability to be programmed via a communication interface (serial or parallel) by programming hardware. If no hardware interface is connected to this communication interface, it will be impossible to program the device with custom code or upload a bootloader.

5.1.1.3 On-Processor Memory

The CDH shall have enough on-processor memory to maintain programmed software, as well as collect data from each payload and each SOH telemetry point. The code on-board the module, stored on the CDH's processor's flash memory, generates both static and dynamic memory via static and dynamic variables. If the CDH processor does not have enough memory on-board to handle both the custom-written code for it or enough memory for the variables that the code generates, then the CDH subsystem or its software is not viable to fly. If more memory is allocated than what is allowed for in the hardware on-board, then the system will crash, failing this requirement.

5.1.1.4 Automation

The CDH shall conduct its operational sequence autonomously in an infinite loop without human-in-the-loop interaction. When the HA-GBS is in flight, no human will be able to control its actions via manual interfacing with the hardware or via the on-board radio (the current system configuration only allows transmission of radio packets, not receiving). Therefore, the CDH system must be able to operate autonomously after power cycles and potential software crashes, otherwise it is not safe for flight.

5.1.1.5 State of Health Telemetry

The CDH shall have a GPS module for accurate time and position, as well as an ADC for measuring, at minimum, input power, regulated power, and on-board temperature. When the HA-GBS is in flight, no human can probe the system to understand if its electrical systems are operating as desired. On the ground, this is normally done with a multi-meter. Therefore, the system must output its state of health in its transmitted packet so that operators can understand the system's operational life cycle and how it responds to stimuli during flight. In order to do this, the CDH must have knowledge of critical voltages and currents as well as module temperature.

5.1.2 Electrical Power System

The EPS functional requirements are described in Sections 5.1.2.1 - 5.1.2.5.

5.1.2.1 Solar Cell Output

The EPS shall generate enough power to sustain nominal operation. All the electrical systems on the HA-GBS require power. If the EPS cannot stay power positive, then the HA-GBS will not be able to turn on or function.

5.1.2.2 Energy Storage

The EPS shall store enough energy to ensure operation during intermittent shading. Though we will be flying above the cloud layer, there is still the chance of shading occurring. The HA-GBS should store enough energy to operate through these short durations.

5.1.2.3 Bus Voltage Rails

The EPS shall maintain a 3V3 voltage rail and a 5V voltage rail under nominal load. Many of the components on the HA-GBS require a clean, regulated 3V3 supply in order to operate, or risk erratic behavior. The 5V rail comes from the nominal voltage of the supercapacitor, and is maintained from input regulation when the solar cells are illuminated. It is important to verify that it can also maintain 5V and not drop drastically when the radio transmits.

5.1.2.4 Over-Voltage Protection

The EPS shall have over-voltage protection to prevent catastrophic runaway capacitor failure. If a converter in EPS fails in an overvoltage mode, this has the potential of creating runaway catastrophic effects as the leakage current increases drastically.

5.1.2.5 Power Recovery

The EPS shall be able to turn on autonomously upon illumination. This is vital to mission success as the EPS will power down during eclipse. It should be able to recover consistently and power the HA-GBS as the sun rises.

5.1.3 Communications

The COMMS functional requirements are described in sections 5.1.3.1 - 5.1.3.3.

5.1.3.1 Global Downlink Capability

COMMS shall transmit data to a global radio network such that packets are uploaded to the internet for analysis. In order to receive data remotely from the HA-GBS, it must be transmitted over RF. Without a downlink connection, operators have no way of knowing the position or state of the vehicle. An uplink connection is not necessary, as the module will operate autonomously (without a human in the loop). Given our goal of circumnavigating the globe for a long period of time, we need to be able to receive data remotely over the internet.

5.1.3.2 Packet Formatting

COMMS shall organize the data produced by the CDH into a packet format that is consistent with APRS standard. The APRS network supports a very specific set of packet formats, described in section 3.1.3. Without the proper formatting, APRS receivers and digipeaters will reject the packet, and our data will not be uploaded to the internet.

5.1.3.3 Geo-fenced Transmission

COMMS shall use GPS position data to select a region-specific APRS frequency to transmit on. The APRS network operates on different frequencies depending on region, as shown in Table 1. For example, when the HA-GBS is over Europe, we need to transmit our APRS packets on 144.800 MHz for local digipeaters and receivers to receive them. Thus, the HA-GBS must autonomously select the transmission frequency based on GPS coordinates.

5.1.4 Payload

The PYLD functional requirements are described in Sections 5.1.4.1 - 5.1.4.2.

5.1.4.1 Voltage Rails

A payload desired to be integrated shall be allowed to be powered on the available voltage rails. In order to ensure that any payload that is desired to be flown during a mission will be powered safely in its lifetime, it must have the ability to be powered by the regulated or unregulated voltage rails. The data sheet for the payload should have the allowable voltage ranges for the hardware. If hardware is supplied power on a non-suitable voltage rail (either out of range or incorrect regulation) it is possible to permanently damage the device, sometimes fatally.

5.1.4.2 Communication Interface

A payload desired to be integrated shall be able to communicate over SPI, I^2C, or UART communication buses since these are the buses supported by the CDH. This is required for any payload to be successfully integrated into the HA-GBS so that the CDH and payloads can understand each other.

5.1.5 Flight Vehicle

The flight vehicle functional requirements are described in sections 5.1.5.1 - 5.1.5.4.

5.1.5.1 Lifting Gas

The flight vehicle shall lift itself using hydrogen as the lifting gas. This would make the system both environmentally and economically sustainable. The historically used helium gas is a precious non-renewable resource that must be preserved.

5.1.5.2 Mission Duration

The flight vehicle shall maintain operational status for a minimum of eight weeks. In order for the system to be successfully integrated in a future DBTO course, it must be operable for at least half of one semester, which in our case is 7 weeks for fall term and 8 weeks for winter term.

5.1.5.3 Operational Environment

The flight vehicle shall lift itself to the desired mission altitude window. This altitude is defined as approximately 20 km in daylight and 15 km in eclipse, therefore ensuring that all hardware in the module are in their expected operational environment. Any global ballooning system must ideally be able to operate above the troposphere (13 km on average), where it can benefit from the atmospheric jet streams for mobility and avoid adverse weather effects. Additionally, high altitude flights operate in lower temperatures, which in return minimizes the lifting gas leakage rate, allowing a longer mission duration.

5.1.5.4 Federal Regulations

The flight vehicle shall abide by all sections of the FAA Code Part 101.1 Sub-Part D Regulations. This is critical to the system's success since its operation would be illegal otherwise.

5.2 Feasibility

The secondary consideration after effectiveness in determining the success of the HA-GBS system is feasibility. Our team had finite resources in terms of time and experience to develop this system, and therefore we had to make feasible design choices. Our design is considered feasible if our team was technically capable of implementing it within the time constraints of a semester-long AE 405 course.

5.3 Desirability

Components of the HA-GBS must also be desirable within the context of the goal of our project. Specifically, we designed this module to serve as an introduction to MXL technology in a space systems DBTO course. Our system design is considered desirable if it uses MXL spaceflight technology or technology that Professor Washabaugh has identified as desirable to use in the flight vehicle design.

5.4 Affordability

Affordability is the final consideration when designing the HA-GBS, as our team had a limited development budget in the scope of AE 405, and the AERO department had a limited budget in the scope of purchasing materials on a larger scale for the DBTO course. Also, affordability encapsulates the environmental costs associated with releasing a global balloon module.

6 Evaluation of Criteria

This section will describe the verification of the assessment criteria detailed in Section 5. These evaluations are either supported by analyses of our design or experimental proofs.

6.1 Effectiveness: Functional Requirements Evaluation

The following subsections detail the evaluation of the HA-GBS for system and design effectiveness.

6.1.1 Command and Data Handling

The effectiveness of the Command and Data Handling subsystem is evaluated in Sections 6.1.1.1-6.1.1.5.

6.1.1.1 Communication Buses

Since the Atmel ATMega328P processor used in our implementation is capable of SPI, I^2C, and UART (one port) hardware communication protocols, as stated in the ATMega328P's datasheet [5], the functional requirement detailed in Section 5.1.1.1 is fulfilled.

No hardware currently on-board the module uses the SPI bus, but all of the sensors can communicate over I^2C as noted in their datasheets [14, 12, 1, 17]. The Radiometrix UHX-1 radio is connected to the one on-processor UART port and is capable of serial communications per its datasheet [15]. Therefore, the CDH communication bus functional requirement is fulfilled.

6.1.1.2 Hardware Programming Interfaces

Figure 6 describes the implementation of the HA-GBS at a top-level. It also describes the implemented ISP (in-system programming) interface that allows us to upload a suitable bootloader and custom flight software. Therefore, this fulfills the functional requirement detailed in Section 5.1.1.2.

For this revision of the HA-GBS, we use an Arduino Uno bootloader and compiled scripts that are uploaded via the Arduino IDE. Upon fabricating the system and attaching the ATMega328P processor, we were able to connect the ISP interface to a computer and upload the bootloader and custom written code. Successful upload of the code from the IDE was verification that this requirement was successfully fulfilled.

6.1.1.3 On-Processor Memory

The Arduino IDE (available for free) offers a tool for calculating the amount of memory used for global variables at compile-time. If this amounts to more than 80% of available memory, the IDE will produce a warning stating that stability problems may occur. The version of the software that we flew on our flight test used 875 bytes of memory for global variables at compile-time, which is about 42% of the 2048 bytes available, and did not produce any warnings from the compiler. A breakdown of our memory usage can be seen in Table 6.

Table 6: HA-GBS has a sufficient amount of memory available to be used for local variables at run-time

Variable Type	**Memory (bytes)**
Global	875
Local (Available at Run-Time)	1173
Total	2048

After the program is compiled, it is the responsibility of the programmer to ensure that the program does not use more memory than available at run-time, causing an overflow error. In C++ programs, there are two types of memory: static and dynamic memory. Static memory is allocated when a function is called and de-allocated when the function returns, whereas dynamic memory is allocated and de-allocated when determined by the programmer. As described in section 3.4, the program runs a setup function on startup and then repeatedly calls a loop function in an infinite loop. Thus, the main concern is that either the setup function or the loop function will allocate too much static memory and cause an overflow error, or that dynamic memory is allocated in the loop function but not de-allocated, causing memory to leak and the program to overflow after several iterations of the loop. Since we tested our software and it made it through both the setup and more than zero iterations of the loop function without crashing, we know that neither function allocates too much static memory. Additionally, we do not allocate any dynamic memory in the loop function, so memory leaks are not a problem. Therefore, we have reasonably certainty to assess that our CDH subsystem fulfills the functional requirement detailed in Section 5.1.1.3.

6.1.1.4 Automation

The CDH subsystem is fully capable of performing all necessary vehicle processes without a human in-the-loop for an extended duration of time. As described in section 3.4, the microcontroller is programmed to collect data from all of its peripherals, format it into a packet, and transmit the packet, all done autonomously once every 30 seconds.

The microcontroller is also programmed to guard against software issues that occur during flight. If the system loses and regains power (power cycling) or if the program crashes due to any unforeseen errors, the microcontroller

will restart the program automatically. Additionally, if the software hangs for any reason, a watchdog timer (see section 3.4.1.1) was included to force the mission program to restart.

In Section 6.1.1.5 we display our data collected from a long bench-top demonstration of the HA-GBS, showing that the on-board software can operate autonomously without a human in the loop for long periods of time. This satisfies the automation criterion in Section 5.1.1.4.

6.1.1.5 State of Health Telemetry

The HA-GBS includes a GPS to determine its position and flight. Figure 31 shows the GPS location from a ground test, plotted on a map. This proves that it was capable of measuring and relaying its time and position to the APRS network as well as time series state of health telemetry.

In Section 3.2.2.2 the HA-GBS includes an LTC2309 ADC to measure input power, regulated power, and on-board temperature. The on-board temperature measured by the ADC can also be corroborated by the temperature sensor on the BNO055, not shown in this section. Figure 26 shows LM20 from a bench-top test that lasted two hours. This plot shows that data can be monitored and streamed from the radio, accurately detailing states of health (in this case temperature). All other state of health telemetry plots (3V3 and 5V current and voltages, as well as solar cell current and voltage) from this test can be seen in Figures A.2 - A.7. Therefore, this fulfills the requirement in Section 5.1.1.5.

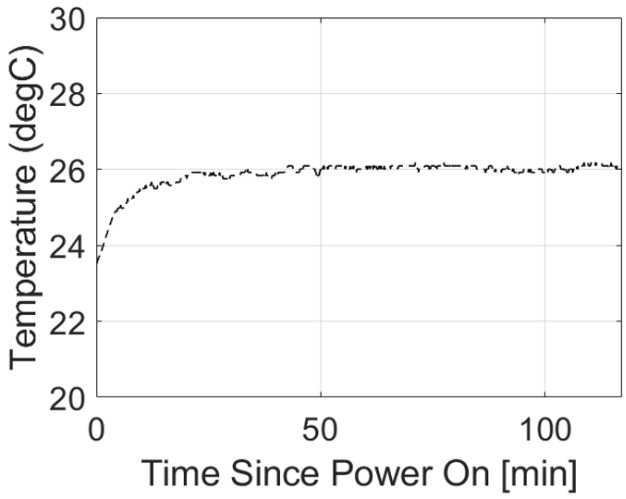


Figure 26: LM20 state of health telemetry plot measuring on-board temperature during a two-hour bench-top test on December 2, 2019. Since the module was inside on a bench-top it is expected that the temperature would stabilize after some time after being powered as the module does not produce a lot of heat.

,

6.1.2 Electrical Power System

The effectiveness of the Electrical Power System is evaluated in sections 6.1.2.1-6.1.2.5.

6.1.2.1 Solar Panel Output

The total power budget of the HA-GBS is illustrated in Table 4 in Section 3.2.1. To verify that the EPS is capable of meeting the specified power requirements of the system, the selected solar cells noted in Section 3.2.1.1 were characterized for power generated at varied illumination intensities and varied incident angles from the light source.

However, the facilities used by MXL to power our solar emulator were unavailable this semester, so we were not able to fully characterize our solar cells. We were only able to characterize for power generated using generic indoor lighting. This unfortunately did not yield accurate performance data.

To begin our indoor illumination characterization test, each solar cell tested was set on a flat surface 2 feet away from an indoor light. A digital multimeter (DMM) was connected in parallel along with a DC load across the positive and ground leads of the solar cell. The light was then turned on, and the DC load was turned on once a voltage was registered on the DMM. We then incremented the load from 1 mA upwards until the generated voltage was no longer stable (floors to 0 V). The previous load was then recorded as the maximum load. The results of this test are shown in the following text matrix.

Table 7: The SM531K08L generates 54% more power than the generic cells in indoor conditions.

Cell	**Voltage (V)**	**Current (A)**	**Power (W)**
IXYS SM531K08L Cell 1	2.83	0.002	0.00566
IXYS SM531K08L Cell 2	2.58	0.002	0.00516
Average:			0.00541
Generic Blue eBay Cell 1	0.302	0.013	0.00392
Generic Blue eBay Cell 2	0.238	0.013	0.00309
Average:			0.00351

In direct sunlight, SM531K08L cells should have been able to reach 4.46 V at 0.205 A at its maximum power point (MPP) according to its manufacturer, supplying 0.913 W of power [11]. Our test resulted in just 0.005 W, which is insignificant compared to its specification, and would not be able to support the HA-GBS. We were unable to make a meaningful conclusion from these results as the test was not an accurate reflection of real-world conditions. Without the ability to further prove the potential of these cells however, they currently fail the power generation requirement. Our solution to remedy this failure for our test flight is addressed in Section 4.

6.1.2.2 Energy Storage

Using the 100F supercapacitors, the EPS is able to power the HA-GBS for 17 minutes, which is more than satisfactory as we are flying above the cloud layer and do not anticipate much shading. This satisfies the requirement originally set in Section 5.1.2.2. The discharge rate for two options of supercapacitors (100F and 2.5F) were characterized as shown below.

Two capacitors, the 2.5F and 100F, were first charged to full capacity through the HA-GBS's onboard supercapacitor balancer (see section 3.2.1.2). A power supply set to 4V with a 2A current limit was connected to the Solar+ line as seen in Figure 9. 4V was chosen to replicate the approximate center of the SM531K08L cells. A DC load and multimeter were then connected in parallel across 5V and GND, also noted in Figure 9. The DC load allowed for safely discharging the capacitors in the event the procedure needed to be terminated. The power supply was then turned on, and capacitors allowed to charge to full. Due to time constraints, the charging data yielded in this procedure was recorded via video recording, and noted afterwards by hand in the test matrices below. Once fully charged at 4.7V, the supercapacitors were then allowed to discharge while being timed, and were considered dead at 2.8V.

Table 8: The 2.5F supercapacitors only charge to a maximum voltage of 4.35V, too low for the HA-GBS.

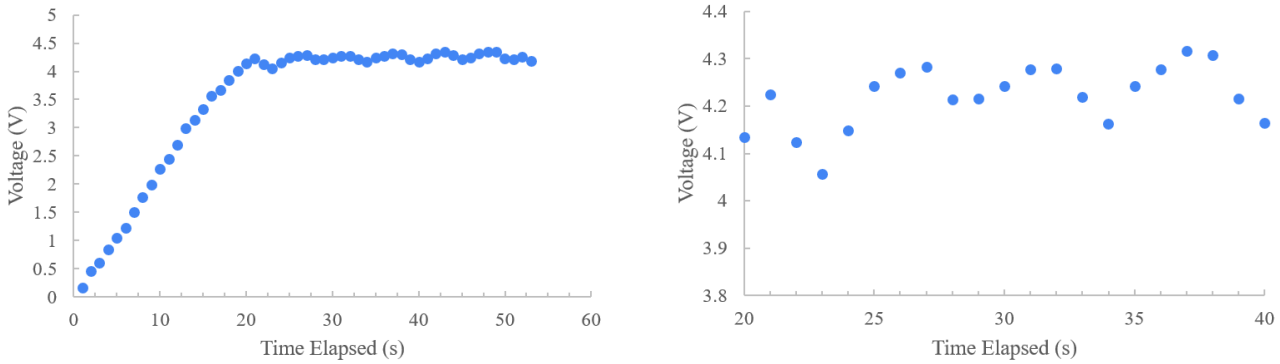
Time (s)	**Voltage (V)**	**Time (s)**	**Voltage (V)**
1	0.164	28	4.214
2	0.455	29	4.216
3	0.6127	30	4.242
4	0.8438	31	4.278
5	1.0551	32	4.279
6	1.2238	33	4.219
7	1.5041	34	4.162
8	1.7751	35	4.242
9	1.9969	36	4.278
10	2.266	37	4.317
11	2.4461	38	4.308
12	2.6973	39	4.216
13	2.9923	40	4.164
14	3.145	41	4.233
15	3.328	42	4.312
16	3.563	43	4.341
17	3.661	44	4.284
18	3.839	45	4.211
19	4.014	46	4.236
20	4.135	47	4.31
21	4.224	48	4.348
22	4.124	49	4.346
23	4.057	50	4.234
24	4.149	51	4.215
25	4.242	52	4.264
26	4.27	53	4.18
27	4.282		

Table 9: The 100F supercapacitors take approximately 20 minutes to fully charge to 4.7V.

Time (s)	**Voltage (V)**	**Time (s)**	**Voltage (V)**	**Time (s)**	**Voltage (V)**	**Time (s)**	**Voltage (V)**
0	0.002	250	1.463	712	3.741	1221	4.647
5	0.006	260	1.504	733	3.833	1242	4.644
10	0.120	265	1.528	743	3.871	1252	4.645
15	0.172	270	1.555	749	3.891	1263	4.635
20	0.221	275	1.580	754	3.927	1273	4.635
26	0.280	285	1.639	774	3.981	1291	4.634
30	0.325	290	1.667	785	4.004	1302	4.637
35	0.369	295	1.697	790	4.010	1312	4.642
40	0.412	300	1.725	795	4.018	1322	4.643
45	0.454	309	1.790	805	4.029	1333	4.632
50	0.496	320	1.801	816	4.063	1343	4.631
55	0.534	330	1.903	826	4.083	1351	4.634
60	0.569	337	1.945	836	4.096	1362	4.626
65	0.604	340	1.960	847	4.119	1372	4.628
75	0.638	351	2.020	857	4.149	1382	4.630
80	0.716	361	2.080	867	4.159	1393	4.631
90	0.732	382	2.198	888	4.188	1411	4.643
95	0.762	392	2.208	898	4.208	1421	4.646
100	0.789	402	2.300	909	4.229	1432	4.637
110	0.867	423	2.390	929	4.274	1453	4.626
120	0.919	444	2.461	960	4.342	1474	4.627
130	0.967	464	2.544	981	4.345	1495	4.627
140	1.013	485	2.605	1001	4.385	1515	4.631
150	1.057	506	2.667	1022	4.418	1536	4.637
160	1.101	526	2.803	1041	4.448	1557	4.632
170	1.142	547	3.053	1061	4.490	1578	4.633
180	1.183	567	3.268	1082	4.519	1599	4.633
190	1.227	588	3.462	1103	4.560	1619	4.637
200	1.265	609	3.633	1123	4.590	1640	4.643
210	1.304	631	3.768	1144	4.617	1661	4.640
215	1.324	640	3.804	1151	4.609	1671	4.639
220	1.343	644	3.821	1162	4.627		
225	1.362	650	3.844	1172	4.636		
230	1.381	660	3.816	1182	4.644		
235	1.401	671	3.782	1193	4.647		
240	1.421	681	3.754	1203	4.643		
245	1.442	691	3.737	1211	4.644		

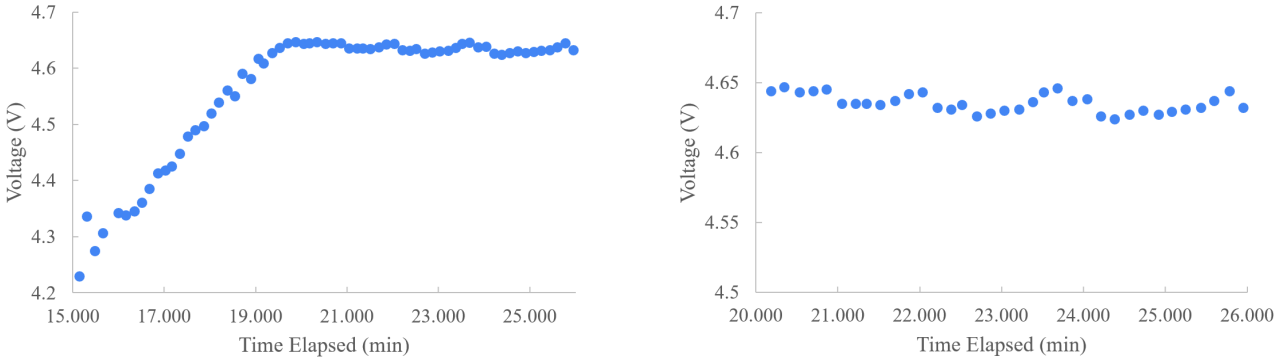
Figures 27 and 28 are generated from the previous test matrices, Tables 8 and 9. It is evident there that while charging the 2.5F capacitor, it maxed at 4.3V due to capacitor characteristics that were not previously foreseen. Because of this (this capacitor is no longer usable for the HA-GBS), the procedure was truncated early for this capacitor, and thus no discharging times were recorded for 2.5F.

The expected discharge times were solved for using the capacitor discharge equation: $v = v_0 e^{\frac{-t}{RC}}$ where v is the voltage being discharged to (2.8V), v_0 represents the starting voltage (4.7V), R is the equivalent resistance of the system, t represents the time, and C is the capacitance of the capacitor being discharged. We can solve for R from $R = \frac{V}{I}$, where v is the voltage of the capacitors, and I is the current (0.170A) calculated from our power budget in Table 4. To solve for time, we rearrange the previous equation to get: $t = -RC \ln \frac{v}{v_0}$.



(a) **(b)**

Figure 27: Figure (a) shows the full 25F charging characterization and (b) shows a close-up where it does not fully charge to 4.7V.



(a) **(b)**

Figure 28: Figure (a) shows the full 100F charging characterization and (b) shows a close-up at full charge where the balancing can be observed.

From the above figures, we can observe the 100F capacitor charging to around 4.65V, which falls within our tolerancing. Regarding discharge times, Table 10 shows that it takes a full 30 minutes to discharge completely, far exceeding our expectations and more than satisfying the requirement set in Section 5.1.2.2. However, as mentioned in Section 4.1, the modifications to the EPS voided use of this capability, and so the prototype flown fails this criterion.

Table 10: The 100F supercapacitors supply power to the HA-GBS longer than anticipated.

Capacitor	Expected Discharge Time (min)	Recorded Time (min)	Comments
2.5F	0.6	–	No discharge time was recorded.
100F	23	30	

6.1.2.3 Bus Voltage Rails

We designed the HA-GBS to have both a 3V3 and 5V bus voltage rail. To prove the effectiveness of the 3V3 rail, we performed powered tests at nominal load conditions (all systems on) to confirm that the it held within ±5% of its target voltage. The procedure for this test was as follows: a power supply was first connected in parallel across the 5V rail and GND, as seen in Figure 11. This was done to emulate the supercapacitor voltage. A DMM and DC load was then connected across the 3V3 rail and GND to measure its voltage, as well as to emulate nominal loads from the bus. The power supply was set to 4.7V with a 2 A current limit, and then turned on to measure the no-load voltage. The DC load was then turned on and set to the appropriate load from the power budget in Table 4, and the nominal-load voltage was also recorded in the test matrix below. As demonstrated in Table 11, the EPS successfully maintained a 3V3 voltage rail within ±0.3%, which meets our requirement.

Table 11: The 3V3 voltage rail stays within ±0.3% of 3.3 V under nominal load.

Test	Load (A)	Voltage (V)	Difference	Pass/Fail?
No Load	0	3.31	0.3%	Pass
Nominal Load	0.038	3.29	-0.3%	Pass

As mentioned in Section 3.2.1.4, the 5V rail is unregulated from the supercapacitor. The input regulators charge the supercapacitors to a maximum of 4.8V. This should be around 4.7V when the system is operating and not transmitting. To confirm that the supercapacitor can maintain the voltage rail under nominal load, it was tested both while connected to input regulation, as well as while it was disconnected from input regulation. While connected to input regulation, it sustained ±5% of 4.7 V without any problems. While disconnected, it successfully sustained within 20% of 4.7 V for 12 minutes while the radio was transmitting every 5 seconds.

The procedure for the 5 V test with input regulation was performed as follows: just like in Section 6.1.2.2, a power supply was set to 4.7 V with a 2 A current limit and connected to the solar input line as seen in Figure 9 to emulate solar panel voltage. A DC load and multimeter were then connected in parallel across 5V and GND, also noted in Figure 9. The DC load allowed us to safely discharge the capacitors in the event the procedure needed to be truncated. The power supply was then turned on and the voltages were recorded while the capacitors were allowed to fully charge. This was performed with the HA-GBS transmitting. After an appropriate amount of data points were collected, the system was discharged safely using the DC load once again. The results of this test can be seen in Table 12.

Table 12: The 5V rail meets the 4.7±0.235V criteria when transmitting.

Time (s)	Voltage (V)
0	4.61
1	4.601
2	4.612
3	4.617
4	4.621
5	4.625
6	4.616
7	4.626

Table 13 compares the no-load 5V data gathered from Table 9 (and seen in Figure 28), and compares it against the data from Table 12. The 5V rail passes flawlessly, and falls within the bounds when powered.

The data from Table 12 has been distilled into Figure 29. Dips are seen when the HA-GBS transmits, at a period of 5 seconds. Even with such a high transmit rate, the 5V rail stays stable and meets the requirements.

Table 13: The 5V rail falls to within 1% of nominal value in both transmitting and resting states.

Test	Load (A)	Nominal Voltage (V)	Low Voltage (V)	Criteria (V)	Pass/Fail?
No Load	0	4.64	4.62	4.7±0.05	Pass
Nominal Load	0.08	4.63	4.6	4.7±0.05	Pass

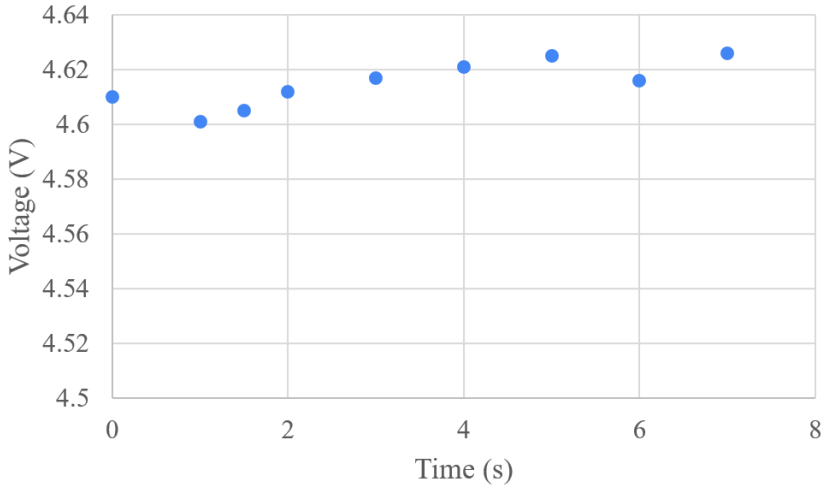


Figure 29: The 5V rail meets the 4.7±5% criteria when transmitting.

The procedure for testing the duration for which the 5V Voltage Rail lasts while unpowered follows much of the same setup as Section 6.1.2.2, the outlier being that major discharge events were recorded in the test matrices below, with timestamps noted. Note that as the 2.5 F capacitors were disqualified in previous testing, only the 100F capacitors were considered here.

Table 14: The HA-GBS continues to transmit for 18 minutes below -20% of 4.7V.

Expected Time (min)	Recorded Time (min)	Event	Comments
17	20	Reach full capacity at 4.7V	The capacitor only tops out at 4.7V due to losses.
28	32	Discharged to 3.7V	This is 20% below 4.7V.
28	35	Stops continuously transmitting at 3.55V	Transmits come infrequently and sporadically, with a lot of noise.
40	50	System can no longer sustain power.	LEDs are off and system is completely dead at 2.8V

Even as the voltage of the 5V rail dropped below 20% of 4.7V, it still continued to sporadically and quietly transmit for another 20 minutes after that threshold. This duration exceeded expectations as we will be transmitting only once per minute.

Since both the 3V3 and 5V rails performed as expected or better, we can consider that this criterion is satisfied.

6.1.2.4 Over-Voltage Protection

The EPS has an array of 5.6V zener diodes for over-voltage protection as seen in Figure 10. To confirm that these diodes performed nominally, they were tested at sustained over-voltage conditions and confirmed to hold the voltage at $\pm 0.1V$ without failure. This test is described below.

A power supply is connected in parallel to a zener diode. The positive end is placed on the positive end of the zener diode to emulate the positive side of the capacitor, and the negative end is connected to the negative end of the zener to emulate ground. This can be seen in Figure 10. The power supply is set to 6V, with a current limit of 2A. The power supply is turned on. If the power supply is not shunted to 5.6±0.1V, immediately turn the supply off. Otherwise record the value and then turn off the supply quickly. This process was repeated for 5 zener diodes.

Table 15: The zener diodes activate appropriately within 5.6±0.1V.

Zener Diode	Voltage Cutoff (V)	Criteria (V)	Pass/Fail
1	5.62	5.6±0.1	Pass
2	5.69	5.6±0.1	Pass
3	5.61	5.6±0.1	Pass
4	5.60	5.6±0.1	Pass
5	5.69	5.6±0.1	Pass
Avg	5.64	5.6±0.1	Pass

The zener diodes meet the criteria for passing and fall within their rated tolerances. They will protect against overvoltage of the supercapacitors, and thus meet this requirement.

6.1.2.5 Power Recovery

We tested the power recovery ability of the EPS subsystem by charging, discharging, and then fully charging it again to verify that the HA-GBS could then power on. This test was performed with two differently sized capacitors: 2.5F and 100F. They were charged to full capacity through the EPS to emulate sunlight with a power supply set to 4 V and 2 A, timing once it was powered. The system was then disconnected from the power supply to allow it to passively discharge through operation. Once the system was fully unpowered, the power supply was then reconnected and the system was charged to full capacity once more. This procedure was repeated 3 times before changing capacitors and repeating the entire process again. A recovery was considered successful if the HA-GBS was able to transmit properly after regaining power. The results of this test can be seen in Table 16.

Table 16: The HA-GBS recovers from brownout without issues 6 times in a row.

Trial	Did it recover? (Y/N)	Capacitor	Time to recover (min)
1	Y	2.5F	0.33
2	Y	2.5F	0.36
3	Y	2.5F	0.37
4	Y	100F	17.0
5	Y	100F	16.2
6	Y	100F	16.4

The test revealed that brownout was not an issue for the HA-GBS, and that it should be able to recover without fail, although recovery times differed depending on the size of the capacitor. Nevertheless, we are confident in the power recovery abilities of the EPS and we consider this criteria to be met by the HA-GBS as designed. However, as mentioned in Section 4.1, the modifications to the EPS voided use of this capability, and so the prototype flown fails this criterion.

6.1.3 Communications

The effectiveness of the Communications subsystem is evaluated in sections 6.1.3.1-6.1.3.3.

6.1.3.1 Global Downlink Capability

As described in Section 3.4, the software that governs the operation of the HA-GBS is adapted from the open-source software implemented in the TrackSoar module [16]. On September 19, 2019, we conducted a low-altitude

flight test of a TrackSoar module from MXL inventory on the Wave Field outside of the Francois Xavier Bagnoud (FXB) building. The purpose of this flight was prove a baseline operational capability of transmitting GPS position over the APRS network.

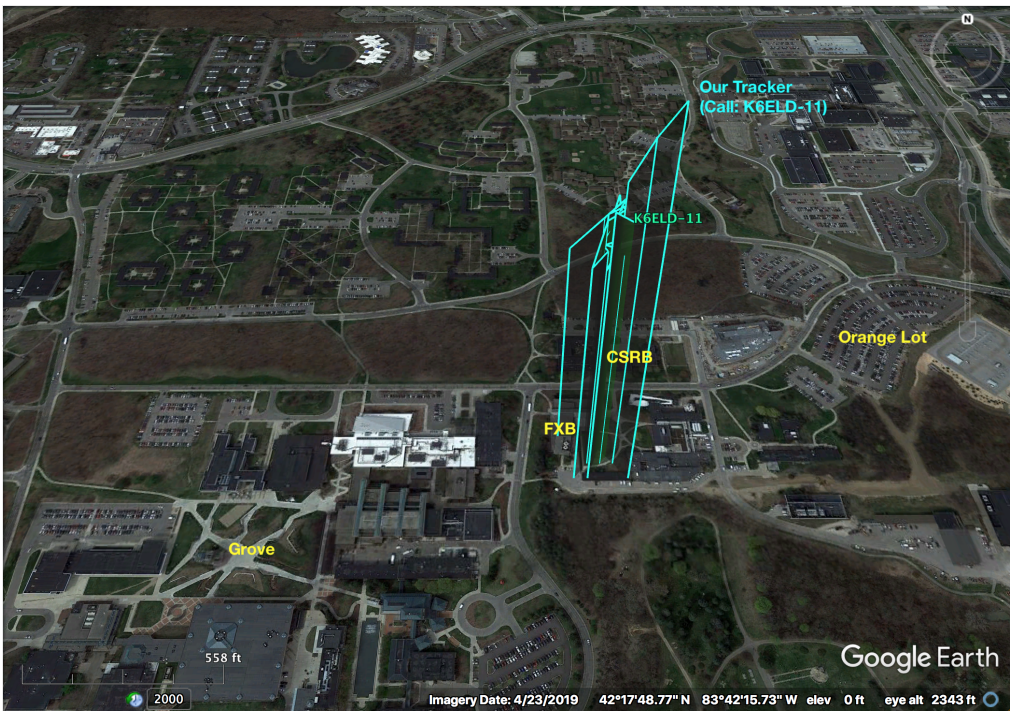


Figure 30: The TrackSoar module was able to transmit accurate GPS position data over the APRS network. Each blue line in the image depicts an instantaneous 3-D euclidean position of the module over a GPS coordinate on Google Earth from ground to the module's altitude.

The TrackSoar module successfully transmitted GPS data (latitude, longitude, altitude, time, speed, and heading) to local APRS digipeaters. The data was pulled from aprs.fi and plotted in three dimensions on Google Earth. Figure 30 shows the resulting trajectory.

This verifies that we can transmit to the APRS network and have them be uploaded to the internet. Also in Figure 31, our implementation of the software and module hardware demonstrates the ability to downlink data to the APRS network.

6.1.3.2 Packet Formatting

To validate that the COMMS system is capable of properly packetizing the data to be transmitted, we tested our fully assembled system by carrying it around the University of Michigan's North Campus and seeing if our transmissions could be properly received, decoded, and plotted on aprs.fi by the APRS network. This is possible because there is an APRS digipeater located on the roof of the EECS building on North Campus that can receive our transmissions and publish them to aprs.fi. When we performed this test, our data packets were received and decoded exactly as expected and our position was accurately plotted. Our position plot for this test on aprs.fi and a decoded packet can be seen in Figure 31.

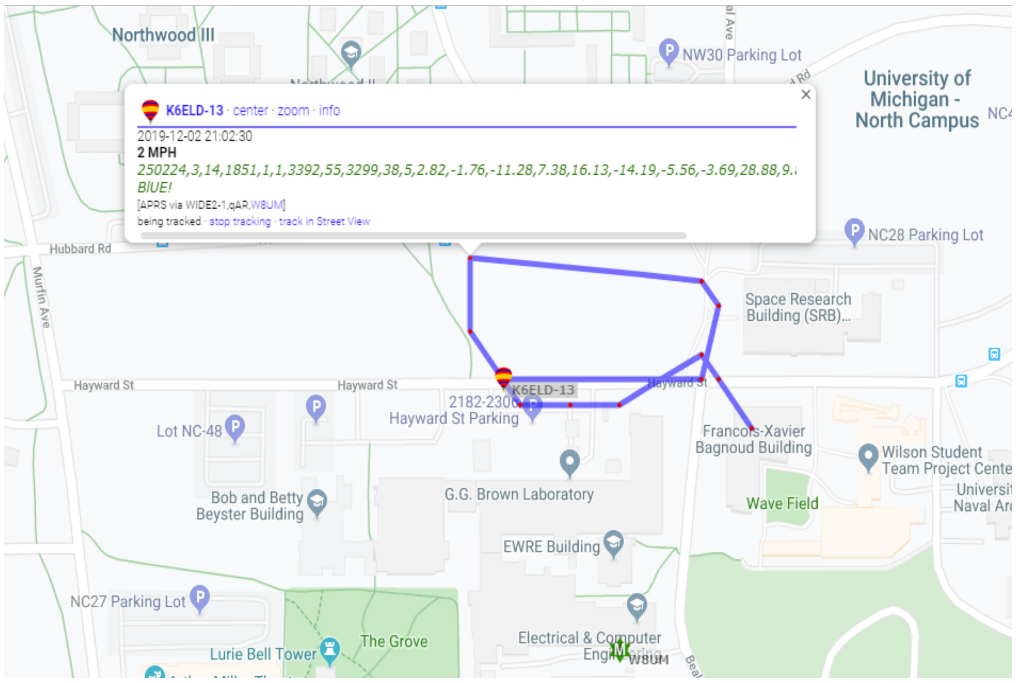


Figure 31: Data packets from the HA-GBS can be properly received, decoded, and plotted by the APRS network during a ground test where team member Alex Chen walked around North Campus on December 2, 2019.

6.1.3.3 Geo-fenced Transmission

The UHX1 radio has the capability of transmitting on any frequency within a predetermined 2 MHz range. This is sufficient for our implementation, as all global APRS frequencies are between 144 and 146 MHz (see Table 1).

On October 22, 2019, we ran a test procedure to confirm that our radio is capable of transmitting on all of the frequencies necessary to be heard by the APRS network throughout the world. At a high level, we transmitted a blank audio tone (i.e. just the carrier signal, no modulation) over a subset of the APRS frequencies listed in Table 1, and recorded the resulting FFT (Fast Fourier Transform) plot on an Agilent n9912 FieldFox RF Analyzer. These plots are shown in Figure 32, and they confirm that the radio can be commanded to transmit on a desired frequency. Note that the actual peak frequencies vary slightly from the commanded values, but this is acceptable given the 25 kHz bandwidth of most APRS digipeaters. The full test procedure can be found in Section B.2.

The frequencies omitted from this test included those requiring 6 significant digits (i.e. 144.575 MHz), as at the time of the test we had not implemented capability to select such frequencies. We now have this capability.

We have written our software to be capable of selecting a frequency to transmit on based on GPS coordinates. However, we have not yet tested this portion of the software. For our flight test, we simply commanded the radio to transmit on 144.39 MHz (the APRS frequency for most of North America) regardless of location since we did not expect the vehicle to leave North America. Since we have confirmed that our radio is capable of transmitting on all necessary frequencies, we can consider the HA-GBS as designed to be capable of geofenced transmissions. However, the prototype flown on December 4th, 2019 did not fly far enough to require a change in APRS transmission frequency (see Table 1), thus it fails this criterion.

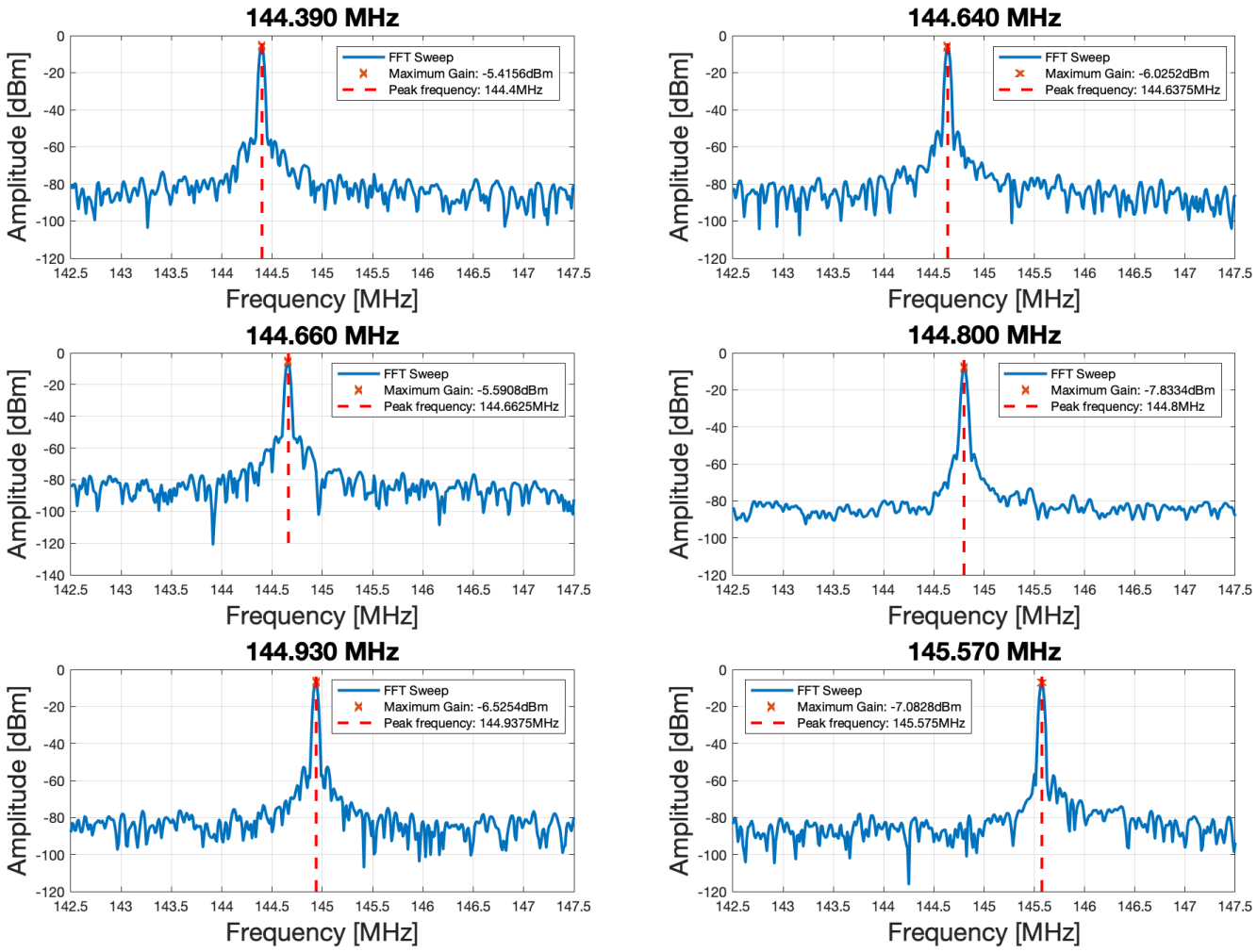


Figure 32: These FFT plots from FieldFox spectrum analyzer confirm that the UHX1 radio can be commanded via HA-GBS software to transmit data over a desired frequency.

6.1.4 Payload

The effectiveness of the Payload subsystem is evaluated in sections 6.1.4.1 - 6.1.4.2.

6.1.4.1 Voltage Rails

Both payloads, the BNO055 and RM3100, are capable of being powered by the 3V3 regulated voltage rail as per their datasheets and recommended designs [1, 14]. Therefore, this criterion in Section 5.1.4 is met.

6.1.4.2 Communication Interface

The two payloads that we flew on our test flight, the BNO055 and RM3100, both support communications through I^2C buses. Thus, our payload satisfies this criterion.

6.1.5 Flight Vehicle

The effectiveness of all subsystems put together is evaluated in sections 6.1.5.1-6.1.5.4.

6.1.5.1 Lifting Gas

The system has been completely simulated using hydrogen as the lifting gas and it shows no sign of inferiority over helium for the purposes of the project's goal. The numerical simulations are thoroughly described in section 6.1.5.2. Therefore, this successfully defines the system as fully environmentally sustainable, and also economically feasible for a large DBTO engineering class, as requested by professor Washabaugh.

However, due to time constraints, we were not able to construct a hydrogen balloon in time for our test flight. Thus, we flew on a helium balloon and our HA-GBS does not yet pass this criterion.

6.1.5.2 Mission Duration

A comprehensive numerical flight simulation has corroborated the possibility of a mission duration of at least 2 weeks. The simulation consisted of an ODE integrator using a fourth-order Runge-Kutta (RK4) method. Appendix B.4 details the analysis done with Python code. The main variables involved in the simulation are:

- System Weight $(W_{total})\,[N]$
- Balloon Temperature $(T_{balloon})\,[K]$
- Balloon Volume $(V_{balloon})\,[m^3]$
- Buoyant Force $(F_B)\,[N]$
- Drag Force $(F_d)\,[N]$
- Simulation Time (t) [weeks]

A representation of the physical system with the given variables can be seen in Figure 33. Further, a brief description of the main equations used in the simulation is outlined below.

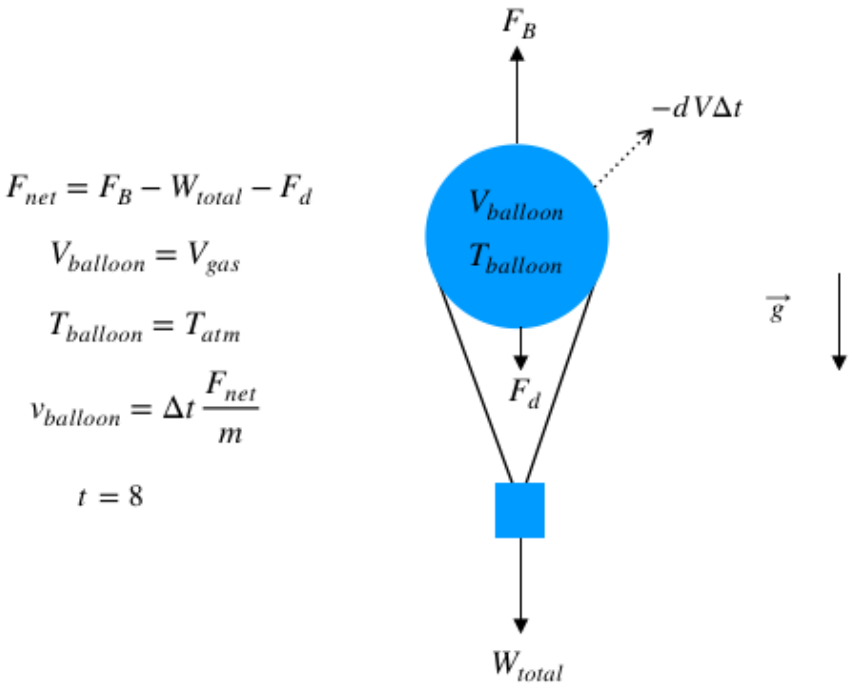


Figure 33: Physical system description of mission for numerical flight simulations

The system weight is defined as the sum of the weights of the module, the balloon envelope, and the volume of gas being displaced:

$$W_{total} = W_{fabric} + W_{H_2} + W_{module}$$

However, given the negligible weight of the fabric compared to the other terms, and that the material used for the fabric is an independent variable in this study, the equation was reduced to:

$$W_{total} = W_{H_2} + W_{module}$$

The amount of hydrogen necessary for the balloon, and therefore its weight, was estimated via balancing a free-body diagram as shown in section 6.1.5.3. The module weight was determined experimentally.

The buoyant force is defined as a function of the density and volume of gas being displaced by the balloon. Note that it is independent of the gas inside the balloon, that is, what is actually displacing the air:

$$F_B = \rho_{air} V_{balloon} g$$

The balloon volume was assumed to the constant throughout the simulation. This is a valid model since superpressure balloon designs must have negligible volume variations during operation as to not disturb the balance of forces. Air density is computed depending on the current altitude (h)using the standard atmosphere assumption for both isothermal and constant-gradient layers. For instance, if the current balloon altitude is within the troposphere, it's lapse rate (ξ) is tabulated, and the following equations can be used:

$$T_{air} = T_{atm} + h\xi$$

$$p_{air} = p_{atm}\left(\frac{T}{T_{atm}}\right)^{\left(-g/(\xi * R_{air})\right)}$$

$$\rho_{air} = \frac{p_{air}}{T_{air} R_{air}}$$

In order to model the drag forces present in the system, a couple of assumptions were made. The balloon was assumed to be a perfect sphere, which is geometrically very close to the actual pumpkin shaped superpressure design. This allows the use of tabulated drag coefficients at different Reynolds numbers. A common value for the drag coefficient of a sphere is 0.47, the number used in this simulation. The drag force is then given as a function of the balloon velocity:

$$F_d = \frac{1}{2} C d \rho_{air} A^2$$

Finally, the leakage rate of the balloon was also considered in this simulation. It is a function of the fabric's permeability coefficent (P_K), the film thickness (t), the pressure differential across the film (Δp), and the contact area (A_{surf}). The permeability coefficient (P_K) was chosen for mylar, which is a common material used in superpressure balloon design, and has its value tabulated by a 1968 NASA study [10]. The leakage rate can the be calculated as:

$$\frac{\Delta V}{\Delta t} = \frac{P_K A_{surf} \Delta p}{t}$$

Now, since this is essentially a 1-D problem, we can define the numerical state and the numerical state derivative of the balloon as a 2x1 vector:

$$\boldsymbol{u} = \left[y, \dot{y}\right]$$

$$\dot{\boldsymbol{u}} = \left[\frac{dh}{dt}, \frac{F_{net}}{m}\right]$$

Using RK4, the physicality of the system shown in Figure 33, and very conservative parameters, altitude plot shown in Figure 34 can be generated.

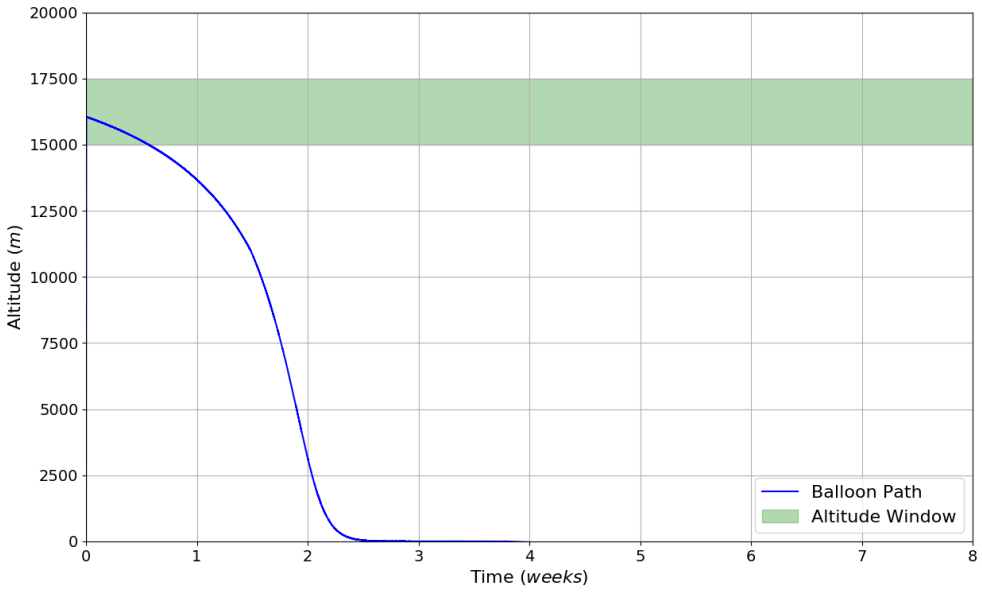


Figure 34: Altitude variation as a function of time of a super-pressure hydrogen balloon

As seen in Figure 34, the flight vehicle can sustain altitude in the desired range for about two weeks, six short of the criterion requirement. It is important to keep in mind, however, that the simulation is highly conservative and assumes higher leakage rates than expected, as well as an equilibrium altitude of 17500 m instead of the expected 20000 m.

Since we were not able to construct a hydrogen balloon with an 8-week lifetime in time for our test flight, we have not yet been able to confirm these results. Thus, we currently do not satisfy this criterion.

6.1.5.3 Operational Environment

The system reaches operating conditions (17500 m) relatively fast as demonstrated by the numerical simulation described in section 6.1.5.2. Furthermore, an additional simulation computing the balloon's stability line was performed, estimating that the desired altitude can be reached with as little as 5 meters cubed of hydrogen at STP. This was done by performing a a free-body diagram balancing analysis where, once equilibrium is reached, the only forces involved are weight and buoyancy.

$$\sum F_y = 0$$

$$F_B - W = 0$$

$$g\left(\rho_{air} V_{gas} - \rho_{gas} V_{gas} - m_{module}\right) = 0$$

Appendix B.5 details the analysis done with Python code. The analysis then produces the following results shown in figure 35

This implies that the operational environment intended is well in reach of the flight vehicle's capabilities. However, since our test flight ended prematurely, we have not been able to verify that this criterion is satisfied. Our last recorded data packet indicated that the vehicle reached 977 m in altitude.

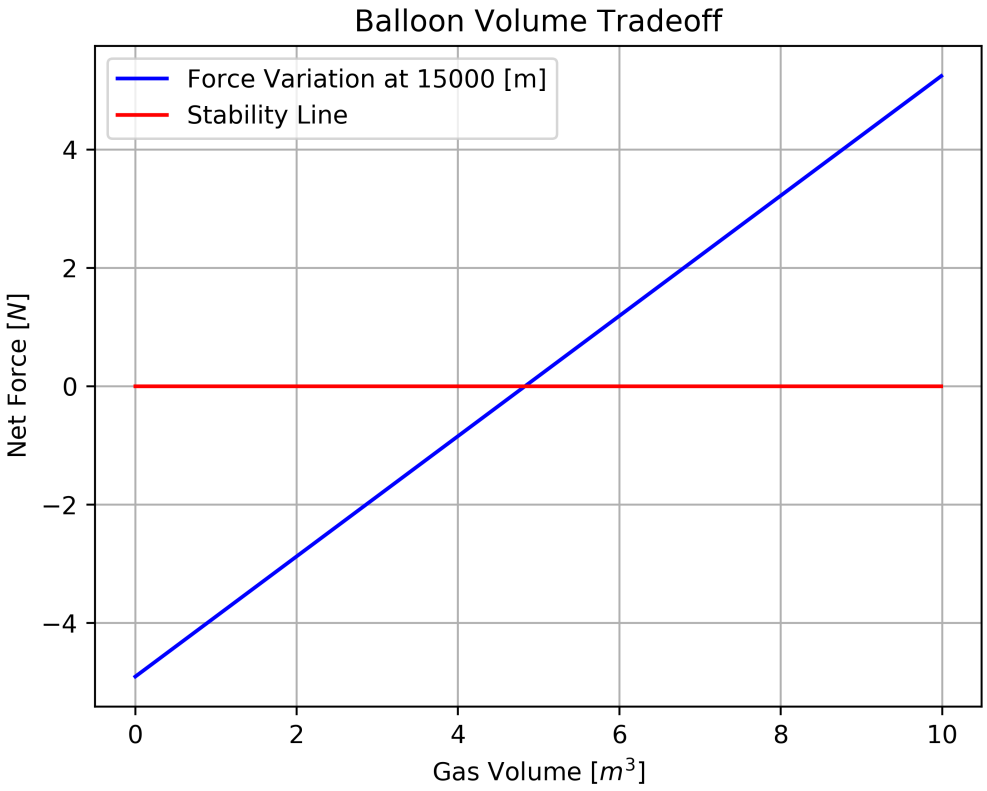


Figure 35: Buoyancy Stability Tradeoff

6.1.5.4 Federal Regulations

The system successfully passes the operating limitations, equipment and marking requirements, notice requirements, and balloon position reports as described by the FAA Code Part 101.1 Sub-Part D. This is empirically demonstrable since the altitude window does not disturb air traffic, the launch location is sufficiently far from populated areas, and the balloon module can provide all necessary telemetry if requested by ATC. Further, MXL has SOPs that require us to interact with the FAA and ATC, providing them our flight path and call sign as an added buffer of safety.

6.2 Feasibility Evaluation

We were able to construct a viable flight vehicle and conduct a test flight in the time allotted by Aero 405, proving the feasibility of the HA-GBS project. Our team members had the technical capability to design, build, test, and operate all subsystems of the HA-GBS and were able to do so before the conclusion of the course.

6.3 Desirability Evaluation

We considered three factors in evaluating the desirability of the HA-GBS: the microcontroller, programming language, and lifting gas.

The microcontroller is considered desirable if it comes from MXL heritage. We chose to use the Atmel ATMega328P as our microcontroller. Unfortunately, MXL has never used this microcontroller in any of its flight systems, which makes our HA-GBS less desireable. However, as development of the HA-GBS progresses beyond Aero 405, we intend to switch to the Texas Instruments MSP-430 microcontroller, which has been used extensively in previous

MXL flight systems.

Professor Washabaugh has identified that low-level C is the preferred programming language for teaching embedded software in his proposed DBTO course. Our choices were to use C or C++. We chose to use C++ in our prototype of the HA-GBS since the open source programs that our software uses are written in C++. However, as development of the HA-GBS progresses and we change to the MSP-430 microcontroller, we intend to rewrite our software in C.

Our choices for lifting gas were hydrogen and helium, which MXL has used for all of its previous ballooning missions. Professor Washabaugh has stated that he prefers the use of hydrogen as a lifting gas for the HA-GBS. This is because hydrogen can be produced in his lab, providing him flexibility in sourcing the lifting gas. Additionally, hydrogen is more environmentally friendly, provides about 10% more lift per unit volume than helium, and is on average 2.5 times cheaper than helium. However, we were forced to use helium for our test flight since hydrogen capabilities are still under development.

Since none of the microcontroller, programming language, and lifting gas currently satisfy their respective requirements, we cannot yet consider the HA-GBS to be desireable.

6.4 Affordability Evaluation

Our first affordability concern was that the HA-GBS could be built using a limited source of funding. Our project was sponsored by Professor Washabaugh and Professor Cutler, and they had sufficient funds for us to purchase all required components.

Our second concern was that our HA-GBS prototype can be affordably implemented in Professor Washabaugh's proposed DBTO course. The total cost of our prototype was less than $2000, which is an appropriate amount for the proposed course.

Our last consideration was the environmental costs of our project. Hydrogen is considered more environmentally friendly than helium. By choosing hydrogen as our lifting gas, we have optimized our system to be environmentally friendly.

Since we meet all three of our affordability requirements, we can consider the HA-GBS to satisfy the affordability criteria.

7 Conclusion

We have designed, built, and tested a prototype of the HA-GBS system. The HA-GBS module is comprised of four subsystems: electrical power system, communications, command and data handling, and payload. It is intended to fly around the world on a hydrogen super-pressure balloon at an altitude of 10 km - 20 km on multiple-month missions while transmitting SOH and payload data to a global amateur radio network.

We conducted a test flight of our prototype, a modified version of our design in Section 3. The test was intended to last eight hours, but it ended prematurely when the prototype stopped transmitting for unknown reasons after we received only one data packet on the APRS network, and 32 packets on our handheld receiver. Even though we had an unsuccessful first flight test, our prototype satisfies 2 out of its 4 high-level assessment criteria and 12 out of 19 functional requirements (effectiveness criteria), detailed in Section 5. We recommend that development of the HA-GBS continue for eventual use in Professor Washabaugh's proposed DBTO course as well as in future MXL or University of Michigan high-altitude research endeavours.

8 Alternatives

Alternative design choices for our prototype HA-GBS included using different or additional payloads and choosing a different microprocessor to acheive better desireability criterion performance. These alternatives are planned to be explored in future design iterations of the HA-GBS and are described in the following subsections.

8.1 Other Payloads

The current HA-GBS payload outfitting only has an orientation sensor and an experimental magnetometer that MXL wants to test for use in future spacecraft. However, there are many other choices for sensors we could have chosen to integrate. In the future, since we have designed the capability to integrate many other types of payloads, it is possible to integrate useful sensors for monitoring weather (humidity, pressure, electric potential, etc.) or other experimental devices. It should be noted that students that take the potential larger scale DBTO course will likely be creative and want to fly payloads not mentioned in this section or in this report

8.2 Processor

We had a choice to use a Texas Instruments MSP-430, which has extensive heritage from previous MXL flight systems, instead of the Atmel ATMega3228P that we chose. The MSP-430 is an extremely low power microcontroller, has MXL flight heritage, and is a more desirable platform for students to learn embedded programming on. The only downside is that it is more difficult to program. In future iterations of the HA-GBS design, we plan on switching to the MSP-430 as our microcontroller to take advantage of the benefits described above.

9 Omissions and Limitations

As described in section 4.1, certain aspects of the HA-GBS intended design were not ready for flight testing before the deadline of the AE 405 poster session on December 5th, 2019, and thus did not make it onto the vehicle flown on December 4th, 2019. These components are listed below, with reasons as to why they were ommitted.

1. Solar Panels: Did not have access to facilities required to successfully validate.
2. Supercapacitors: With no solar panels, supercapacitors as energy storage would only last 20 minutes.
3. Super-pressure balloon: Did not have time to procure or manufacture and validate a super-pressure balloon that could meet requirements listed in Section 5.1.5.3.
4. Hydrogen as fill gas: Did not have time to validate safety procedures with handling Hydrogen.
5. Geofencing capability: Flight test was not designed to fly far enough to trigger a geofenced transmission frequency change by leaving the continent (See Table 1 for frequencies by region.

In the criteria evaluation for these components, we considered both their performance in validation tests run on the ground and on the flight. They will be implemented in future design iterations of the HA-GBS.

10 Schedule

We were able to complete all of the tasks for our project before the end of the semester. Throughout the semester, we made numerous changes to the schedule. The biggest change we made was that we were originally planning on conducting a third flight test lasting two months. This was canceled as we realized that we would not have time for it. Our original schedule can be seen in Figure 36 and our final schedule can be seen in Figure 37.

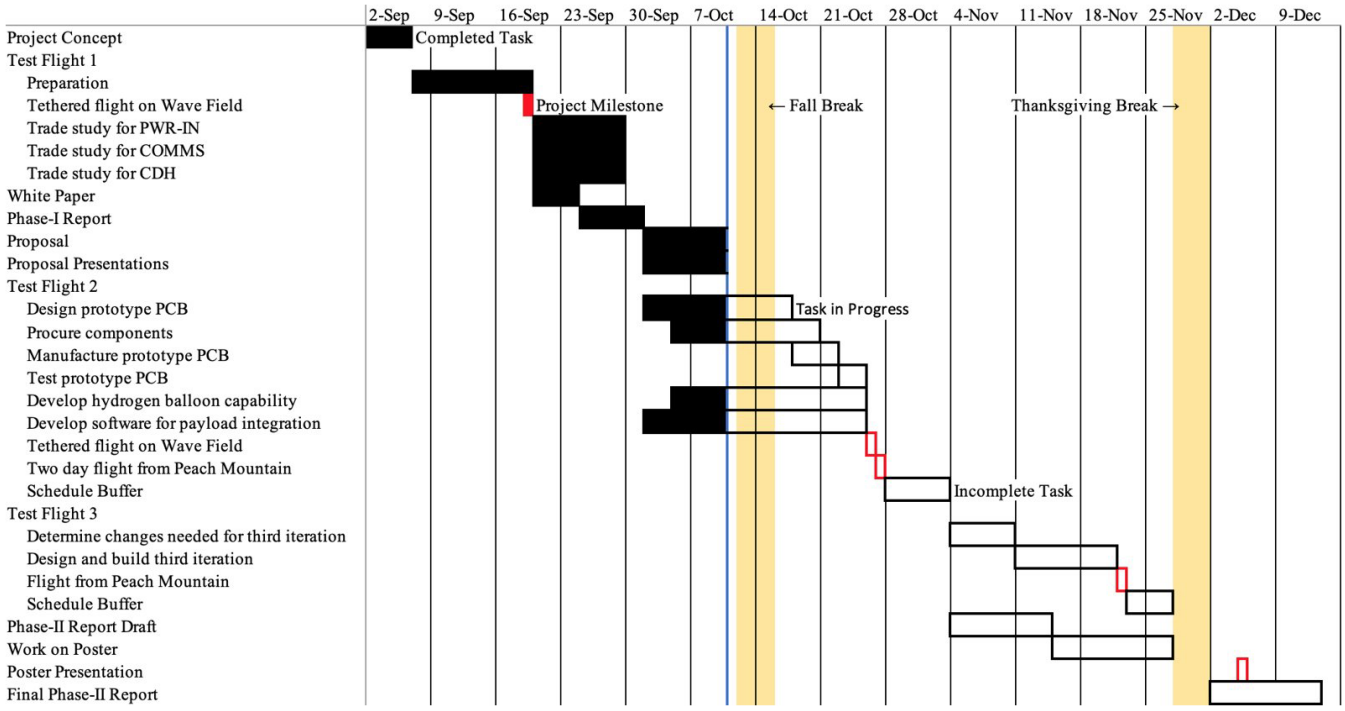


Figure 36: Our original plan as of October 10, 2019 was to conduct three test flights

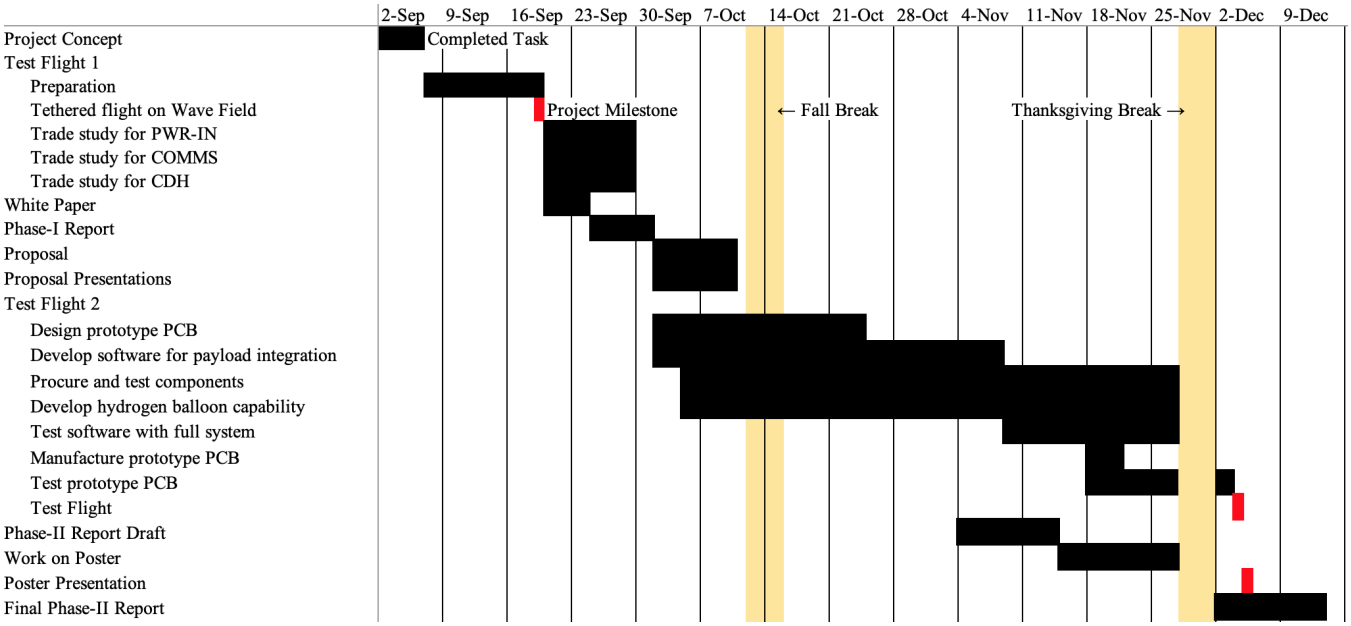


Figure 37: After revising our schedule, we conducted only two flight tests this semester and delayed our third flight test to next semester.

11 Comparison with Proposed Cost

Table 17 displays our final cost budget. In our project proposal, we estimated a total materials cost of $1853. With a 25% overhead and 10% margin, this brought our total proposed budget to $2501.55. Since our grand total for the project was $1477.07, we are under-budget by 41%.

Table 17: Final budget of $1477.07 is well under proposed budget of $2501.55.

Item(s) Purchased	Quantity	Unit Cost (USD)	Total Cost (USD)
PNI RM3100 Magnetometer	2	20.00	40.00
Radiometrix Radios	2	99.00	198.00
Solar Cells	1	43.74	43.74
PCB Order	1	102.13	102.13
PCB Components	1	473.20	473.20
Kaymont Latex Balloons	1	120.00	120.00
K-sized Helium Tanks	2	250.00	500.00
Grand Total			**1477.07**

12 References

[1] Adafruit/Bosch. *Adafruit BNO055 Absolute Orientation Sensor*, 11 2019. url: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-bno055-absolute-orientation-sensor.pdf.

[2] Federal Aviation Administration. Detroit class b airspace, 2019.

[3] Federal Aviation Administration. Part 101—moored balloons, kites, amateur rockets, unmanned free balloons, and certain model aircraft, 2019.

[4] APRS Working Group. *APRS Protocol Reference*, 08 2000. url: http://www.aprs.org/doc/APRS101.PDF?fbclid=IwAR21NJ_JkLkrnn_kg514sD6XWmeXZZ7PVzYo5qgLR5ct\DGgGWSoZ66CW8Y4.

[5] Atmel. *8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*, 1 2015. doi: 7810D-AVR-01/15.

[6] James Dolce and Anthony Colozza. High-altitude, long-endurance airships for coastal surveillance. 2005. doi: NASA/TM-2005-213427.

[7] Imke Durre et al. High-altitude, long-endurance airships for coastal surveillance. *Journal of Climate*, 19(1):53–68, 2006. doi: 10.1175/jcli3594.1.

[8] John Nash et al. Introduction to upper air measurements with radiosondes and other in situ observing systems. *Integrated Ground-Based Observing Systems Applications for Climate, Meteorology and Civil Protection Magazine*, 2007. url: https://www.wmo.int/pages/prog/www/IMOP/meetings/Upper-Air/ET-IOC-3/Doc3.1(1).pdf.

[9] Peter Ibelings (ibelings). Tweet: (2019-4-30) muon tracker, 2019.

[10] IIT Research Institute (for NASA). *Permeability Data for Aerospace Applications*, 03 1968. doi: N69-14111.

[11] IXYS. *IXOLAR High Efficiency SolarMD*, 09 2018. url: http://ixapps.ixys.com/DataSheet/SM531K08L.pdff.

[12] Linear Technologies Corporation. *8-Channel, 12-Bit SAR ADC with I2C Interface*, 01 2019. url: https://www.analog.com/media/en/technical-documentation/data-sheets/2309fd.pdf.

[13] University of Southampton. Astra high altitude balloon flight planner, 2019.

[14] PNI. *RM3100 Breakout Board Manual v22*, 08 2017. url: https://www.pnicorp.com/download/rm3100-testing-boards-manual/.

[15] Radiometrix. *NBFM Multichannel 500mW VHF Transceiver*, 11 2007. url: `http://www.radiometrix.com/files/additional/uhx1.pdf`.

[16] Santa Barabara Hacker Space (SBHX). Tracksoar — open-source aprs tracker, 2018.

[17] uBlox. *MAX-M8 series u-blox M8 concurrent GNSS modules*, 05 2019. url: https://www.u-blox.com/sites/default/files/MAX-M8-FW3$_{D}ataSheet_{\%}28UBX - 15031506\%29.pdf$.

[18] AMSAT UK. Lu picoballoons travel the around the globe, 2019.

A Appendix - Figures

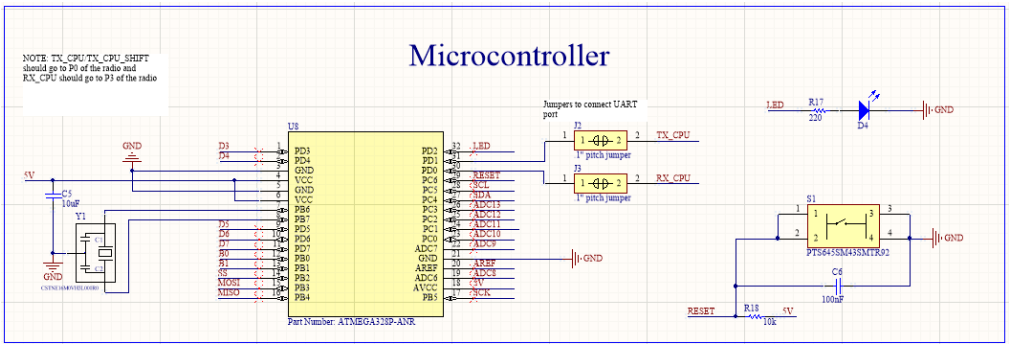


Figure A.1: CDH micro-controller schematic, showing the implementation for the Atmel ATMega328P with a 16MHz clock, reset circuitry, and a multipurpose LED

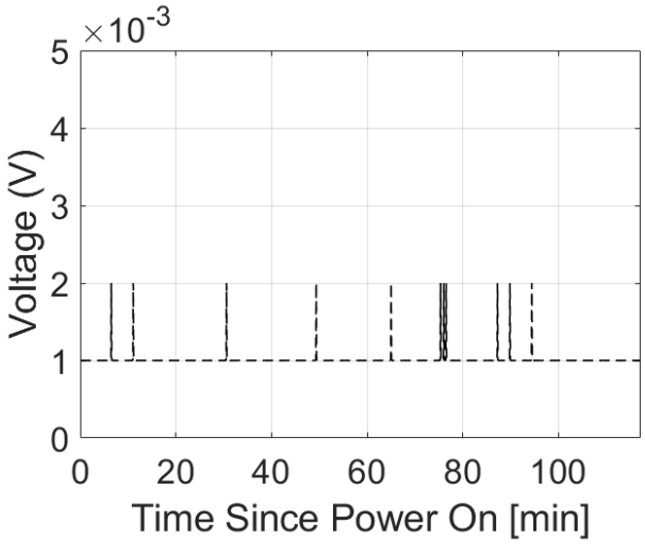


Figure A.2: State of health telemetry point from the ADC showing solar cell input voltage during a two-hour bench-top test on December 2, 2019. Since solar cells were not used during this test no power was measured as being input.

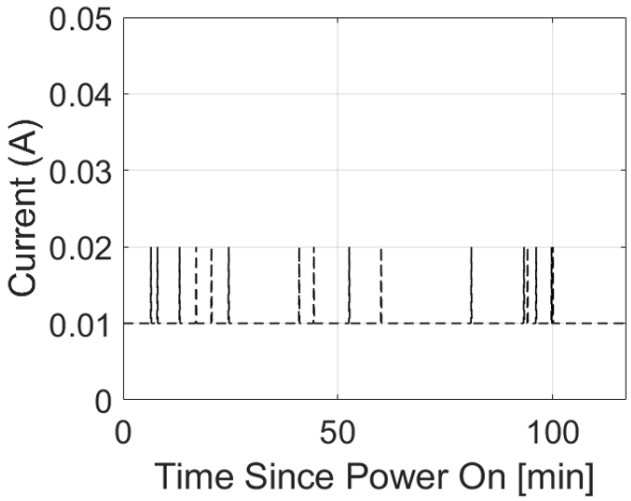


Figure A.3: State of health telemetry point from the ADC showing solar cell input current during a two-hour bench-top test on December 2, 2019. Since solar cells were not used during this test no power was measured as being input.

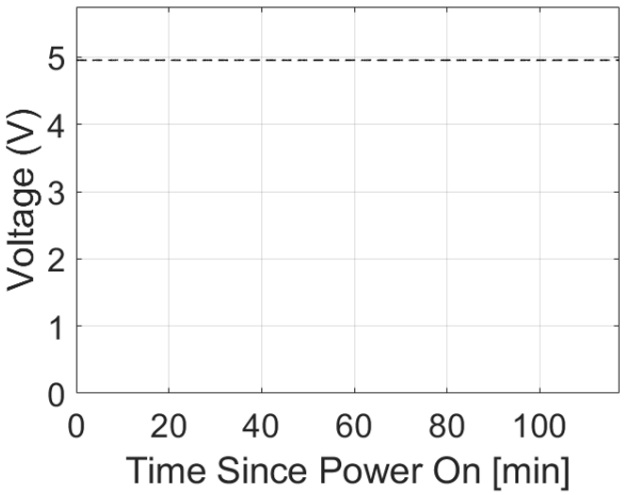


Figure A.4: State of health telemetry point from the ADC showing 5V rail voltage during a two-hour bench-top test on December 2, 2019. Since the module was powered by a DC power supply this voltage was expected to be constant.

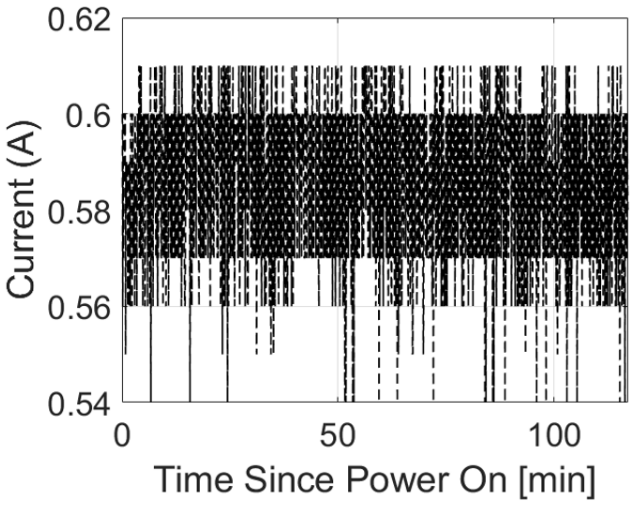


Figure A.5: State of health telemetry point from the ADC showing 5V rail current during a two-hour bench-top test on December 2, 2019.

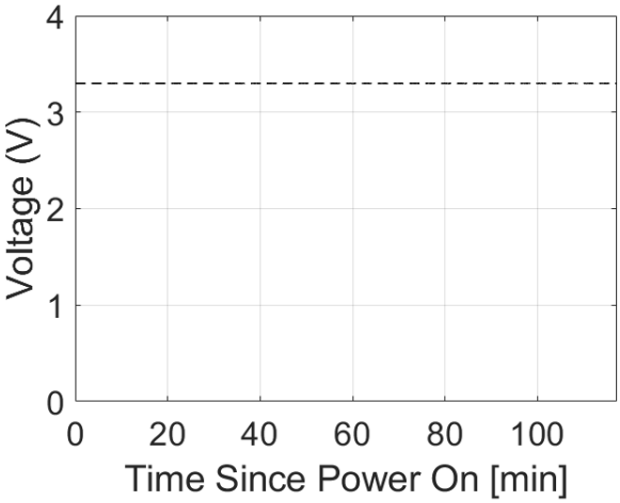


Figure A.6: State of health telemetry point from the ADC showing 3V3 rail voltage during a two-hour bench-top test on December 2, 2019. Since the rail regulated, this voltage was expected to be constant.

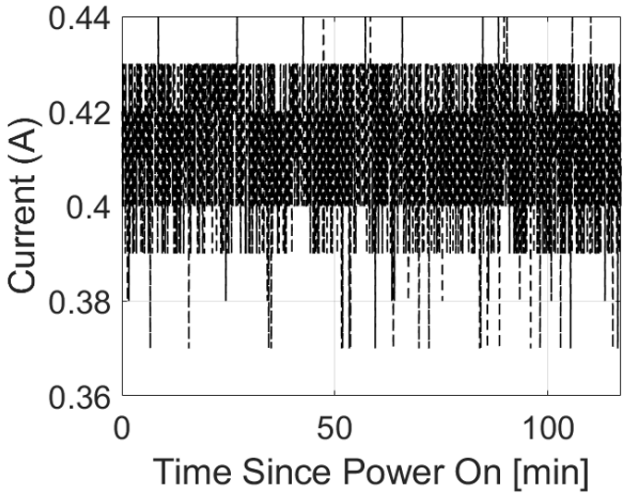
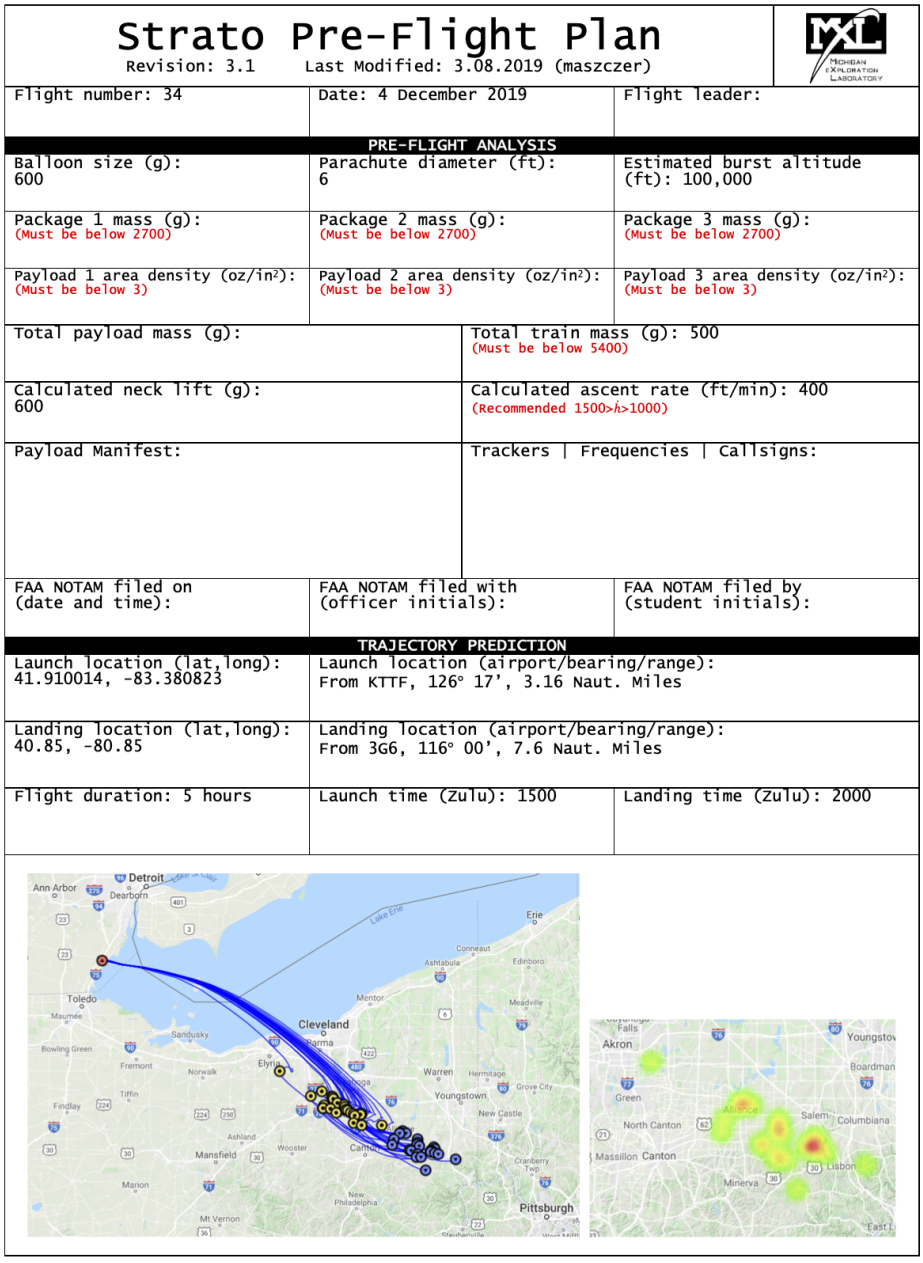


Figure A.7: State of health telemetry point from the ADC showing 3V3 rail current during a two-hour bench-top test on December 2, 2019.

B Appendix - Attachments

B.1 MXL Strato Pre-Flight Plan



Strato Pre-Flight Plan

Revision: 3.1 Last Modified: 3.08.2019 (maszczer)

Flight number: 34	Date: 4 December 2019	Flight leader:
PRE-FLIGHT ANALYSIS		
Balloon size (g): 600	Parachute diameter (ft): 6	Estimated burst altitude (ft): 100,000
Package 1 mass (g): (Must be below 2700)	Package 2 mass (g): (Must be below 2700)	Package 3 mass (g): (Must be below 2700)
Payload 1 area density (oz/in²): (Must be below 3)	Payload 2 area density (oz/in²): (Must be below 3)	Payload 3 area density (oz/in²): (Must be below 3)
Total payload mass (g):	Total train mass (g): 500 (Must be below 5400)	
Calculated neck lift (g): 600	Calculated ascent rate (ft/min): 400 (Recommended 1500>h>1000)	
Payload Manifest:	Trackers | Frequencies | Callsigns:	
FAA NOTAM filed on (date and time):	FAA NOTAM filed with (officer initials):	FAA NOTAM filed by (student initials):
TRAJECTORY PREDICTION		
Launch location (lat,long): 41.910014, -83.380823	Launch location (airport/bearing/range): From KTTF, 126° 17', 3.16 Naut. Miles	
Landing location (lat,long): 40.85, -80.85	Landing location (airport/bearing/range): From 3G6, 116° 00', 7.6 Naut. Miles	
Flight duration: 5 hours	Launch time (Zulu): 1500	Landing time (Zulu): 2000

Figure B.1: Page 1 of MXL Strato Pre-Flight Plan detailing launch procedures for our test flight

TEAM AND ROLE ASSIGNMENTS		
Apollo Driver:	Chase Car Driver:	3rd Car Driver:
Apollo Radio Operator:	Chase Car Radio Operator:	3rd Car Radio Operator:
Prelaunch Team 1:	Prelaunch Team 2:	Prelaunch Team 3:
Chase Team:	Apollo/Clean-Up Team:	Ground Station Team:

LAUNCH SITE MEASUREMENTS		
Time of departure from FXB (local):	Time of arrival at launch site (local):	
Local pressure (Pa):	Local temperature (C):	Local relative humidity:
Ground Winds (mph, bearing):	Cloud Cover: ☐ Clear ☐ Partly ☐ Mostly ☐ Complete	
Actual launch time (local):	Actual landing time (local):	Actual recovery time (local):
Landing Environment:	Actual landing location (lat, long):	

Figure B.2: Page 2 of MXL Strato Pre-Flight Plan detailing launch procedures for our test flight

T-1 Day Standard Checklist

- □ Verify Weather Prediction to ensure valid trajectory
- □ File NOTAM with FAA
- □ Check contents of all flight bins
  - o Fill Kit
  - o Ground Kit
  - o Tarp Bin
  - o Mobile Ground Station Kit Red
  - o Mobile Ground Station Kit Yellow
- □ Charge/Change Batteries
  - o Kenwood HTs (2)
  - o GoPros
  - o Tracker and FTU
- □ Verify operational status of mobile ground stations and trackers (Remember to turn off after testing)
- □ Refuel chase vehicles
- □ Check on Apollo:
  - o Tire pressure
  - o Live battery
  - o Fuel level
- □ Prepare ballast bucket
- □ Assign flight day roles
- □ Prepare payloads and lay out train
- □ Fill out Strato Flight Plan

Name: _____ Date: _____

Pre-Departure Standard Checklist

- □ Verify weather prediction to ensure valid trajectory
- □ Print driving directions
- □ Load helium tanks into trailer
- □ Check that helium tanks are securely fastened
- □ Load flight bins into trailer
  - o Ground Kit
  - o Fill Kit
  - o Tarp Bin
- □ Install Mobile Ground Station
  - o One station on Apollo
  - o One station on Chase Car
- □ Test VHF/UHF comms
  - o UHF: _____ MHz
  - o VHF: _____ MHz
- □ Set APRS beacon on Apollo
- □ Record departure time

Name: _____ Date: _____

Figure B.3: Page 3 of MXL Strato Pre-Flight Plan detailing launch procedures for our test flight

Launch Site Standard Checklist

- □ Record arrival time and ambient conditions

Ground Team Checklist

- □ Unload all boxes and helium tanks
- □ Lay tarp on ground at launch site and pin down with boxes and tanks or stakes
- □ Place launch supplies on tarp
  - o Balloon (Primary and backup)
  - o Nitrile or Latex gloves
  - o Helium Tanks
  - o Zip-ties
  - o Duct tape
  - o Regulator, hose, and fill valve
  - o Wrench
  - o Scissors
  - o Ballast can
  - o Fish scale
- □ At T-15 minutes, begin fill process. ALL PEOPLE WHO MAY TOUCH BALLOON MUST BE WEARING GLOVES
  - o Put second tarp over top of first tarp (if winds > 2 mph)
  - o Position at least three people holding the top tarp (if applicable) and one person on fill/helium duty
  - o Use helium-regulator-hose-fill adapter assembly to inflate balloon about ¾ of the way
- □ Remove top tarp (if applicable) if flight team is finished with T-10 minutes tasks. Flight team helps hold balloon in place
- □ Continue filling until ballast just lifted
- □ At T-30 seconds, complete any launch-time tasks

Flight Team Checklist

- □ Unload all flight hardware and place near launch site
- □ Connect train elements via prepared lines, with top of train near balloon fill site
- □ Tighten all mechanical connections between train elements with wrench
- □ Make and secure any electrical connections
- □ At T-15 minutes, power on auxiliary tracker and apply anti-fog solution to GoPros
- □ At T-10 minutes, turn on GoPros and begin recording. Secure GoPros in housing.
- □ Record launch time

Figure B.4: Page 4 of MXL Strato Pre-Flight Plan detailing launch procedures for our test flight

Materials Standard Checklist

Ground Kit
- ☐ Metal bings
- ☐ Carabiners
- ☐ Latex gloves
- ☐ Adjustable wrench
- ☐ Scissors
- ☐ Ground stakes (8)
- ☐ Noah-style tarp
- ☐ Soldering iron
- ☐ Multimeter and probes
- ☐ Anti-fog solution

Fill Kit
- ☐ Balloon (2x)
- ☐ Helium tank regulator
- ☐ Fill adapter/valve
- ☐ Latex/Nitrile Gloves
- ☐ Zip Ties
- ☐ Duct tape
- ☐ Adjustable wrench
- ☐ Fish scale
- ☐ Scissors

Toolbox
- ☐ Zip ties
- ☐ Flashlight
- ☐ Solder
- ☐ Solder wick
- ☐ Screwdrivers
- ☐ Writing utensils
- ☐ Electrical tape
- ☐ Hot glue gun
- ☐ Hot glue sticks
- ☐ Jumper wire
- ☐ Adjustable wrench
- ☐ Teflon tape
- ☐ Tape measure
- ☐ Wire cutters
- ☐ Wire strippers
- ☐ Box knife

Other
- ☐ Power inverter (car alternator to 110V AC)
- ☐ Gas generator
- ☐ Stakes for Noah Tarp
- ☐ Water can
- ☐ Helium tanks
- ☐ Kenwood Bag
  - o 2 Kenwood D72 HTs
  - o Charger
  - o 3 Antennas

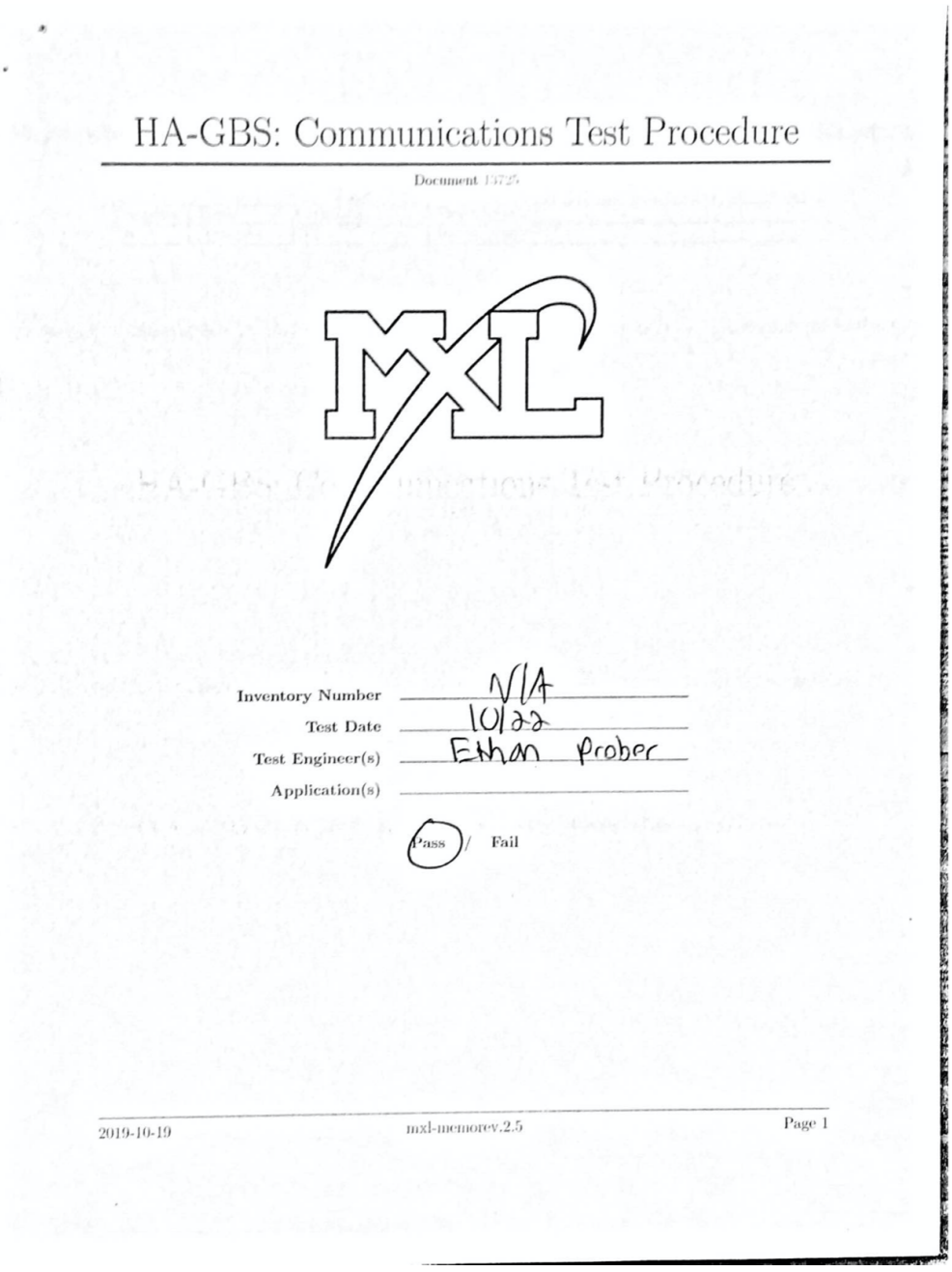
Tarp Bin
- ☐ Tarps

Mobile Ground Station: Yellow
- ☐ Kenwood D710
- ☐ Kenwood D710 display
- ☐ Kenwood D710 display mount
- ☐ Half-wave mag-mount antenna
- ☐ Power cord
- ☐ Mic
- ☐ Avmap
- ☐ Avmap mount
- ☐ GPS
- ☐ GPS antenna

Mobile Ground Station: Red
- ☐ Kenwood D710
- ☐ Kenwood D710 display
- ☐ Kenwood D710 display mount
- ☐ Half-wave mag-mount antenna
- ☐ Power cord
- ☐ Mic
- ☐ Avmap
- ☐ Avmap mount
- ☐ GPS
- ☐ GPS antenna

Figure B.5: Page 5 of MXL Strato Pre-Flight Plan detailing launch procedures for our test flight

B.2 APRS Frequency Tuning Test Procedure



HA-GBS: Communications Test Procedure

Document 1372-

Inventory Number N/A

Test Date 10/22

Test Engineer(s) Ethan Prober

Application(s)

(Pass) / Fail

2019-10-19 mxl-memorev.2.5 Page 1

Figure B.6: Page 1 of APRS frequency tuning test procedure

Revision History

Revision	Date	Author(s)	Description
1.0	2019-10-19	ETP	Initial Release

Figure B.7: Page 2 of APRS frequency tuning test procedure

Contents

Figure B.8: Page 3 of APRS frequency tuning test procedure

1 Introduction

The purpose of the Communications (COMMS) subsystem is to reliably downlink state-of-health, payload telemetry, GPS location, and mission data from the HA-GBS to a global amateur radio network. This subsystem is crucial to successful operations of the HA-GBS as it is the sole connection between the balloon module and the ground. The communications link will be a one-way downlink, meaning we will not have the capability to transmit to the vehicle from the ground. Figure 1 details the flow of data through the HA-GBS.

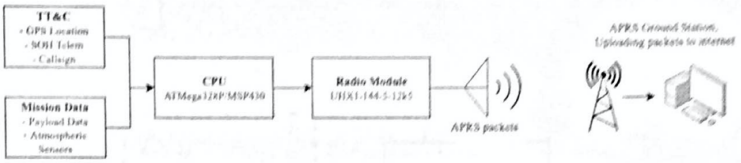


Figure 1: Mission Data Flow Diagram for HA-GBS COMMS System

The purpose of this test procedure is to determine integration readiness of the COMMS system into the HA-GBS. We will be assessing the performance of a singular radio module on its ability to:

1. Communicate with the radio over its serial interface.
2. Transmit on a software-defined frequency.

The test is designed to be run on a single radio to isolate its functionality from variables that might arise through use with other untested hardware and software.

The transmission frequency needs to be software-defined because as the HA-GBS circumnavigates the globe, it will need to transmit at different frequencies to be picked up by the Automatic Packet Reporting System (APRS) network. Table 1 displays the different frequencies the HA-GBS needs to be able to transmit at.

Table 2: APRS Frequencies for Different Global Regions

Region	Frequency [MHz]
Columbia, Chile, Indonesia, Malaysia, North America, Thailand	144.390
New Zealand	144.575
China	144.640
Japan	144.660
Europe, Russia, some parts of Africa	144.800
Argentina, Paraguay, Uruguay	144.930
Australia	145.175
Brazil	145.570

Figure B.9: Page 4 of APRS frequency tuning test procedure

2 Configuration Setup

2.1 Configuration Diagram

Figure 2 shows the configuration diagram of this test. Refer to this section when setting up the test.

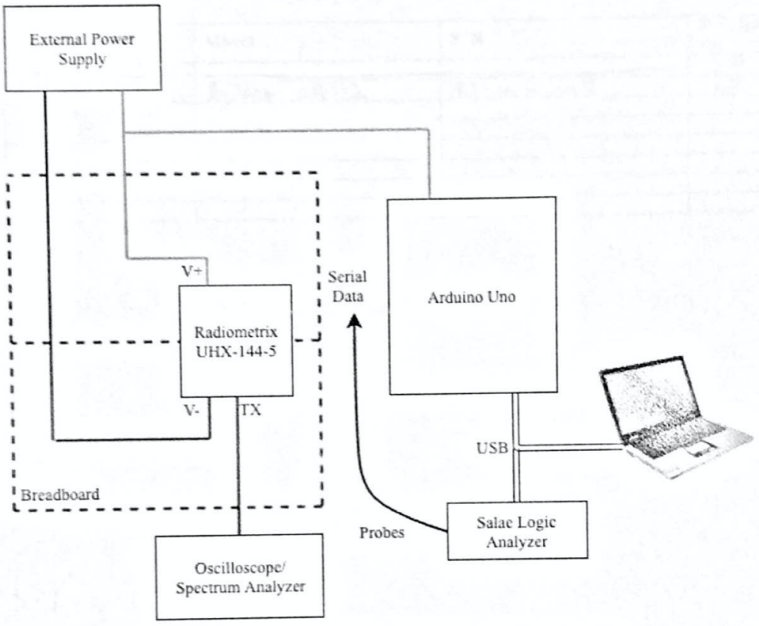


Figure 2: Configuration Diagram for Communications Functional Test

Figure B.10: Page 5 of APRS frequency tuning test procedure

2.2 Materials Log

Table 3 displays the materials required to run this test. Upon collecting all components, document the model and serial number (S/N) for each. Use the extra space for any miscellaneous components used.

Table 3: Materials Log for the COMMS Functional Test

Component	Model	S/N	T.E. Initial
~~Power Supply~~			EP
FieldFox RF Analyzer	Agilent n9912	MY51463298	EP
Salae Logic Analyzer			
Arduino Microcontroller			

Figure B.11: Page 6 of APRS frequency tuning test procedure

3 Procedure

The following sections give in chronological order the steps required to complete this procedure. Upon successful completion of each step, initial and date in the space provided. A high-level flowchart of the procedure is shown in Figure 3.

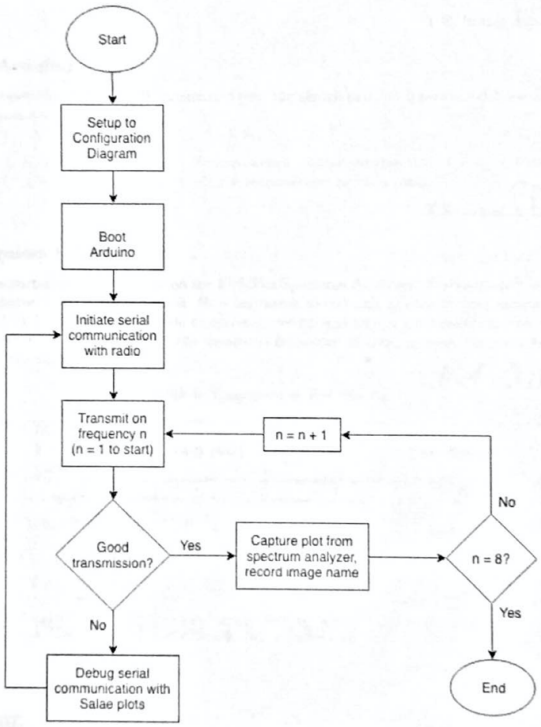


Figure 3: Test Procedure Flowchart

Figure B.12: Page 7 of APRS frequency tuning test procedure

3.1 Test Setup

Confirm experimental test setup matches desired test setup described in Figure 2. Ensure materials and instruments used are recorded in Table 3. Note, all components should be placed on Electro-Static Discharge (ESD) safe surfaces, and an ESD band must be worn at all times by the Test Engineer.

T.E. Initial: EP Date: 10/22

3.2 Boot Arduino

On the laptop, open the Arduino IDE program. Open the sketch entitled InsertSketchNameHere, and check it for completeness and correctness.

WARNING: DO NOT enable the transmission before connecting RF Out from radio to FieldFox with an attenuator or to a load.

T.E. Initial: EP Date: 10/22

3.3 Transmission Tests

You will now characterize transmissions on the FieldFox Spectrum Analyzer. The frequency is determined by altering a parameter in the Arduino sketch. Run the sketch to transmit at each of the frequencies in Table X. Follow the N9912A FieldFox User's Guide to correctly set up and trigger the Spectrum Analyzer to capture the transmission's Fourier plot. Record the measured frequency of transmission, the gain, bandwidth, and the file name of the captured plot.

Ref: 0 dB, Att: 10dB

Table 4: Transmission Test Matrix

Tested Frequency [MHz]	Measured Frequency [MHz]	Gain [dB]	Bandwidth [kHz]	File Name	T.E.
144.390				14439	
144.575					
144.640	144.643552	-9.63	25	14464	
144.660	144.657973	-9.42	25	14466	
144.800	144798321	-10.88	25	14480	
144.930	144933505	-11.49	25	14493	
145.175					
145.570	145.571517	-10.84	25	14557	

T.E. Initial:____ Date:____

4 Cleanup

Return all MXL instruments to their proper locations, clear

Figure B.13: Page 8 of APRS frequency tuning test procedure

B.3 HA-GBS Flight Code

Our code repository is a University of Michgigan GitLab repository. As our code base is quite large, it is not feasible to publish all of our drivers and software scripts into this document. If getting access to the HA-GBS software repository is desired please contact Justin Schachter at `jschach@umich.edu` for access.

The link to the repository is: `https://gitlab.eecs.umich.edu/jschach/aero405-ha-gbs-code`.

B.4 Balloon Simulation Code

```
# Purpose:
# ------------------------------------------------------
# To perform a superpressure hydrogen balloon flight simulation

# Libraries:
# ------------------------------------------------------
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Rectangle

# Parameters:
# ------------------------------------------------------

R = 287.058                       # Specific gas constant for air [J/
kg]
g = 9.81                 # Gravitational acceleration [m/s^2]
T0 = 288.15                       # Sea-level temperature [K]
p0 = 101325                       # Sea-level pressure [Pa]
rho0 = 1.225                      # Sea-level density [kg/m^3]
xi_trops = -0.0065                # Troposphere lapse rate [.]
rho_h2 = 0.090                    # Hydrogen density [kg/m^3]
rho_he = 0.179                    # Helium density [kg/m^3]
mass = 0.5                        # System mass [kg]
Tsim = 86400*7*8                  # Simulation time [s]
Nt = Tsim*3                       # Data points for simulation [.]
Nx = 2000                         # Data points for atmosphere [.]
V = 6.77                          # Assumed balloon volume [m^3]
Cd = 0.47                         # Spherical drag coefficient [.]
A = 1.0                  # Balloon Cross Sectional Area [m^2]
Pcff_h2 = 0.1*10e-8               # Hydrogen Permeability at -46C
[STD]
film_thickness = 0.001           # Mylar thickness [m]

# Defining the Standard Atmosphere:
# ------------------------------------------------------

h = np.linspace(0, 20000, Nx+1)
T = np.zeros_like(h)
p = np.zeros_like(h)
rho = np.zeros_like(h)

# Constant-Gradient Layer:

for i in range (0, 1101):
        T[i] = T0 + xi_trops*(h[i])
        p[i] = p0*(T[i]/T0)**(-g/(xi_trops*R))
        rho[i] = p[i]/(R*T[i])

# Isothermal Layer:
```

Figure B.14: Balloon Simulation Code

```
for j in range (1101, 2001):
        T[j] = 216.65
        p[j] = p[i]*np.exp((-g*(h[j]-h[i]))/(T[j]*R))
        rho[j] = p[j]/(R*T[j])

# Numerical Integration:
# ------------------------------------------------

def getf(Un, fnet):
        u_dot = np.array([Un[1], fnet/(mass+rho_h2*V)])
        return u_dot

Tn = np.linspace(0, Tsim, Nt+1)   # Time array
Un = np.zeros([2, Nt+1])                                # State
corresponds to altitude and vertical speed [h,vy]
dt = float(Tsim)/(Nt)                                                   #
Constant time step

# Note that the initial state of the balloon is [0,0]

for k in range(1,Nt):

        # Find corresponding atmospheric conditions:

        h_temp = Un[0,k]                                # Current
balloon altitude
        index = (np.abs(h - h_temp)).argmin()                   #
Binary searches index in altitude array
        fballoon = g*(rho[index]*V - rho_h2*V - mass)           #
Computers net force for current atmospheric conditions
        ffriction = -0.5*Cd*rho[index]*A*(Un[1,k])**2

        if(Un[1,k] <= 0):
                ffriction = -ffriction

        fnet = ffriction + fballoon

        # Perform Runge-Kutta:

        f0 = getf(Un[:,k], fnet)
        f1 = getf(Un[:,k] + 0.5*dt*f0, fnet)
        f2 = getf(Un[:,k] + 0.5*dt*f1, fnet)
        f3 = getf(Un[:,k] + dt*f2, fnet)
        Un[:,k+1] = Un[:,k] + (dt/6.0)*(f0 + 2*f1 + 2*f2 + f3)

        Tn[k+1] = Tn[k] + dt

        # Leakage Rate:

        Pcff = (Pcff_h2*10e-6)/(10e-4*100000)
```

Figure B.15: Balloon Simulation Code (continued)

```
        dV = (Pcff*63*dt*(p[0]-p[index]))/film_thickness
        V = V - dV

        #print(k)

u_plot = Un[0,:]
plt.plot(Tn/604800.0, u_plot, label = 'Balloon Path', color = 'b')

someX, someY = 0.0, 15000.0
width = Tsim/604800.0
height = 2500

currentAxis = plt.gca()
currentAxis.add_patch(Rectangle((someX, someY), width,
height,alpha=0.3,color='g',label='Altitude Window'))

plt.xlabel(r"Time $(weeks)$", fontsize=16)
plt.xticks(fontsize=14, rotation=0)
plt.yticks(fontsize=14, rotation=0)
plt.ylabel(r"Altitude $(m)$", fontsize=16)
plt.ylim([0,20000])
plt.xlim([0,Tsim/604800.0])
plt.grid()
plt.legend(loc="lower right", fontsize=16)
plt.savefig("H2Sim", dpi=500)
plt.show()
```

Figure B.16: Balloon Simulation Code (continued)

B.5 Balloon Volume Calculator Code

```
# Purpose:
# ------------------------------------------------------
# To perform a superpressure hydrogen balloon volume analysis

# Libraries:
# ------------------------------------------------------
import numpy as np
import matplotlib.pyplot as plt

def gas_volume(m, h_flight):

        # Parameters:
        # ------------------------------------------------------

        R = 287.058                     # Specific gas constant for
air [J/kg]
        g = 9.81                # Gravitational acceleration [m/s^2]
        T0 = 288.15                     # Sea-level temperature [K]
        p0 = 101325                     # Sea-level pressure [Pa]
        rho0 = 1.225                    # Sea-level density [kg/m^3]
        xi_trops = -0.0065              # Troposphere lapse rate [.]
        rho_h2 = 0.090                  # Hydrogen density [kg/m^3]
        Nx = 2000                       # Data points for atmosphere
[.]

        # Defining the Standard Atmosphere:
        # ------------------------------------------------------

        h = np.linspace(0, 20000, Nx+1)
        T = np.zeros_like(h)
        p = np.zeros_like(h)
        rho = np.zeros_like(h)

        # Constant-Gradient Layer:

        for i in range (0, 1101):
                T[i] = T0 + xi_trops*(h[i])
                p[i] = p0*(T[i]/T0)**(-g/(xi_trops*R))
                rho[i] = p[i]/(R*T[i])

        # Isothermal Layer:

        for j in range (1101, 2001):
                T[j] = 216.65
                p[j] = p[i]*np.exp((-g*(h[j]-h[i]))/(T[j]*R))
                rho[j] = p[j]/(R*T[j])

        # Driver:
        # ------------------------------------------------------
```

Figure B.17: Balloon Volume Calculator Code

```
        index = (np.abs(h - h_flight)).argmin()   # Binary searches
index in altitude array
        rho_flight = rho[index]                                    #
Air density at desired flight altitude
        V = np.arange(0,10,0.01)
        F_net = g*(rho_flight*V - rho_h2*V - m)
        zeros = np.zeros(len(V))

        plt.plot(V, F_net, color='b', label='Force Variation at 15000
[m]')
        plt.plot(V, zeros, color='r', label='Stability Line')
        plt.xlabel(r"Gas Volume $[m^3]$")
        plt.ylabel(r"Net Force $[N]$")
        plt.title("Balloon Volume Tradeoff")
        plt.legend()
        plt.grid()
        plt.savefig("Ballon_Volume", dpi=500)

gas_volume(0.5, 15000)

# ------------------------------------------------------
# END
```

Figure B.18: Balloon Volume Calculator Code (continued)