Design and Analysis of a High-Powered Model Rocket

A Major Qualifying Project Report
Submitted to the Faculty of the -
WORCESTER POLYTECHNIC INSTITUTE
In Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science
In Aerospace Engineering
by

[illegible]
Tiana Am

[illegible]
Emily Abbe

[illegible]
Bryce Bragdon

[illegible]
Paul Coccomo

[illegible]
Garrett Devlin

[illegible]
Logan Frandsen

[illegible]
Sarah Hildreth

[illegible]
Megan Malito

[illegible]
Jacob Roller

[illegible]
Kevin Shultz

[illegible]
Mason Thyng

[illegible]
Hunter Wagner

March 27, 2022

Approved by: [illegible]
Zachary Taillefer, Advisor
Professor, Aerospace Engineering Department
WPI

Abstract

This paper describes the design, construction, and flight of a reusable high-power model rocket to a minimum altitude of 1000 ft. The Airframe and Recovery Systems, Flight Dynamics and Analysis, and Propulsion, Thermal, and Separation System subteams had various roles in the design, construction, and flight of the rocket. The ARS subteam designed the airframe, recovery bay, and fins, and chose altimeters and ejection method. The FDA subteam calculated the nosecone drag force, designed a flight computer to record data, and created an apogee detection algorithm in MATLAB. The PTSS subteam selected the Cesaroni I540 motor, designed an innovative separation system, and analyzed motor thermal loads using MATLAB and Cantera. All subteams collaborated for assembly and launch of the rocket.

Acknowledgements

The MQP team would like to thank the following individuals for their continuous guidance and support of our project:

- Project advisor Professor Taillefer for his support and critique throughout our project work.
- Professor Hera for her role in providing ANSYS, Fluent and COMSOL training as well as her one-on-one support throughout the project.

Table of Contents

List of Figures

List of Tables

Table of Authorship

Section	**Author**
Abstract	BB
Acknowledgments	Team
6.1 Executive Summary	LF, PC, TA
6.2 Project Goals	TA
6.3 Project Design Requirements	TA
6.4 Tasks	TA
6.5.1 Background Review: Model Rocketry Design Review	TA
6.5.2 Background Review: ARS	GD
6.5.3 Background Review: FDA	TA
6.5.4 Background Review: PTSS	BB
1.6 Design Overview	LF, MM
7.1.1 Airframe	GD, MM
7.1.2 Fins	EA, GD, MM
7.1.3 Nosecone	EA, MM
7.1.4 Recovery Bay	EA, LF, MM
7.1.5 Pre-Launch Procedure	GD, LF, MM
7.2.1 Altimeter Testing	LF
7.2.2 Airframe and Component Analysis	GD, MM
7.2.3 Ejection Test Results	LF
7.2.4 Launch Results	LF

8.1 Overview	KS
8.2.1 OpenRocket Model and Predictions	KS
8.2.2 Flight Computer	JR
8.2.3 Flight Software	KS
8.2.4 Nose Cone Drag Analysis	SH
8.3.1 Analysis of Rocket Performance	KS
8.3.2 Flight Computer Assembly and Electrical Testing	JR
8.3.3 Analysis of Aerodynamic Loads	SH
8.3.4 Results and Analysis from Flight Data	JR, KS
9.1.1 Motor Selection	HW
9.1.2 Thermal Analysis Cantera	MT
9.1.3 COMSOL Analysis of Solid Rocket Motor	PC, BB
9.1.4 Separation System Baseline	MT, BB
9.1.5 Separation System Innovative	PC
9.1.6 Creation of Innovative Mechanism	HW
9.1.7 Testing Procedure of Innovative Mechanism	BB
9.1.8 Motor Structure Integrity	HW
9.1.9 Umbilical Cord	HW
9.2.1 Motor Selection	HW
9.2.2 Thermal Analysis Cantera	MT
9.2.3 Thermal Analysis COMSOL	PC
9.2.4 Motor Post Flight Analysis	BB
9.2.5 Innovative Separation System Testing	HW

10.1 Conclusion and Recommendations: ARS	LF
10.2 Conclusion and Recommendations: FDA	KS
10.3 Conclusion and Recommendations: PTSS	PC
10.5 Broader Impacts	TA

1. Introduction

1.1. Executive Summary

The baseline rocket is a standard rocket that uses black powder to separate the stages. The black powder is ignited, building pressure inside the airframe until it is high enough to break shear pins holding the various airframe components together. The innovative system was designed to be non-energetic and instead would have a servo retract pins holding the airframe components together, allowing springs to push the airframes apart. The innovative system was designed to remain inside the rocket even during baseline flights, as it meant only one rocket needed to be constructed and the stability calculations would be consistent between the baseline and innovative rockets.

Most of the design decisions made by Airframe and Recovery System (ARS) were based on what materials had been bought by previous MQP teams, as that would be the most cost-effective choice. As such, the rocket consists of a 4-inch diameter, fiberglass airframe with a fiberglass nosecone. The coupler tube and inner tube were also bought by previous MQPs and were able to be used for our project. The fins of the rocket were plywood fins with a fiberglass layup to increase the strength of the fins.

The recovery system consists of the recovery bay and the parachutes themselves. The recovery bay consisted of a 3D printed sled mounted to aluminum bulkheads. A TeleMega altimeter was used as the backup altimeter, as a previous MQP team had purchased one. An RRC3 Sport altimeter was purchased to act as the primary altimeter. The rocket had two parachutes, a main parachute and a drogue parachute. The drogue parachute deployed from the lower airframe at apogee while the main parachute deployed from the upper airframe 500 feet above ground level.

ARS performed multiple tests before the launch to ensure the rocket would be able to be recovered and reused after launch. Multiple test layups were performed so the process for the layup could be perfected before doing the layup on the final rocket. Additionally, the two altimeters were tested to ensure they could ignite the black powder charges for the main and drogue parachutes.

During the launch, the airframe and recovery system performed as expected. Both parachutes properly deployed when they were programed to deploy, and the only damage found on the rocket was a single zip tie that sheered upon landing. Otherwise, the rocket could readily be flown again.

Prior to launch, Flight Dynamics and Analysis (FDA) analyzed various aerodynamic loads on the model rocket to ensure a vertical flight. In addition, FDA was tasked with designing a custom flight computer responsible for recording flight data such as altitude, acceleration and more. Multiple tests were performed to ensure quality of the sensors. Lastly, code was written to predict the rocket's apogee, and to detect various states of the rocket including take off, boosting, apogee, decent and more. Apogee detection is essential to the rocket's safe landing as it allows for the parachutes to be deployed in a timely manner. The code performed as expected and provided the custom flight computer with accurate altitude of 1684 ft. The state machine malfunctioned where the boost state timed out and led to cascade to ground state however, both parachutes deployed in a timely manner which ensure the rocket's safety.

Initially, the Propulsion, Thermal, and Separation Systems (PTSS) group had to select a motor for our rocket. The altitude goal of our rocket was 1000 ft, and our launch site had an altitude limit of 2000 ft. Additionally, the motor had to provide an off-rod velocity of about 50 ft/s to ensure stability on launch. We selected the CTI I540 motor which gave us an expected altitude of about 1400 ft and an off-rod velocity of 55 ft/s. The next tasks for the PTSS group were to determine stage separation systems and perform a thermal analysis of the rocket motor.

The PTSS group began working on the stage separation by determining the system for the baseline rocket. The two most common options for model rocketry are black powder or CO_2 ejection systems. While black powder is less expensive and simpler to use, it requires much more cleaning than a CO_2 system. The cost for a CO_2 system would be much more expensive so it was decided to use black powder.

Additionally, PTSS designed an innovative stage separation system with the intent of flying it on our innovative rocket. The system we designed used retractable pins to hold the nosecone to the upper airframe of the rocket. Using a servo, the pins would be retracted and allow compressed springs to extend and force the nosecone off the rocket. This mechanism would make cleanup and reassembly simpler since no black powder residue would need to be cleaned, and black powder would not need to be packed again. During testing of the system, we discovered that none of the spring options were suitable for ejecting the nosecone. We believe the springs do not provide the impulse required, so this mechanism was not flown.

Additionally, the PTSS group performed thermal analyses of the motor using both Cantera and COMSOL. Cantera was used to determine the properties inside the combustion chamber of the motor during the chemical reaction of the propellant grain burning. Using these outputs, the mass flow rate, temperature, pressure and exit velocity can be determine along with other properties. Mass flow rate is one of the key inputs into the COMSOL simulation which provides a heat transfer model and a fluid flow model. The fluid flow model showed the fluid leaving the motor and entering the nozzle, and the heat transfer model verified that no parts in the assembly would reach temperature levels where they would experience failure.

In the week leading up to the launch date, the MQP team spent time performing practice assemblies of the rocket, painting the rocket, and creating safety checklists. Our launch date was on 2/18/2023 in Durham, CT hosted by the CATO Rocketry Club. On the launch day, our team performed an ejection test to verify we had the correct amount of black powder to separate the airframe and pull our parachutes out. We followed the procedures outlined in the checklist we created and successfully launched our baseline rocket to an altitude of 1612 ft. This overshot our expected altitude, but the rocket was recovered safely and intact.

1.2. Project Goals

The three overall project goals are as follows:

- Design, build, and fly a reusable rocket to an altitude of 1500 feet
- Provide students with the opportunity to work as a team to design, build and test a moderately complex aerospace system in which the overall vehicle performance is critically tied in with the mass and performance of the individual components and assemblies
- Provide students with specialized training in and opportunity to apply software tools: MATLAB, ANSYS - Static Structural Analysis, ANSYS - Fluent, ANSYS - Dynamic Analysis, Cantera, others

The goals of the three sub teams, Airframe Recovery System, Flight Dynamics Analysis and Propulsion, Thermal and Separation System, are as follows:

- Airframe and Recovery System (ARS) Sub Team
    - Design and fabricate airframe structure

    - Lead integration of subsystems: payload, recovery, staging, avionics, and propulsion
    - Design, fabricate, and test both baseline and innovative recovery system
- Flight Dynamics Analysis (FDA) Sub Team
    - Perform analysis of aerodynamics loads on vehicle in flight
    - Lead selection and integration of avionics, including processor (if any), accelerometers, gyros, and altimeter
    - Perform simulation of rocket flight dynamics (attitude angles and rates, acceleration)
    - Perform analysis of rocket performance (altitude, range, etc.) in support of design activities and flight planning
    - Support integration of other subsystems: payload, recovery, staging, avionics, and propulsion
- Propulsion, Thermal and Separation Systems (PTSS) Sub Team
    - Selection, modeling, and test of (commercially available) motors for single and two-stage rockets as well as mounting and ignition system
    - Design, fabricate, and test both baseline and innovative stage separation systems.
    - Perform analysis of thermal loads from the motor(s) during flight.
    - Support integration of propulsion and staging subsystem

1.3. Project Design Requirements, Constraints, and Other Considerations

The design requirements, constraints, and considerations for the project were decided by the three sub-teams and were contingent upon the design standards of the National Association of Rocketry (NAR) for high-power model rockets.

1.3.1. Design Requirements

- Use a dual parachute deployment for the recovery system
- Use of black powder system for lower parachute separation system
- Use a Level-1 impulse motor
- Use a custom flight computer to record flight data

1.3.2. Design Constraints

- Include innovative system in the baseline rocket
- Use leftover components from previous year's model rocket including the airframe, parachutes, shock chords and nomex blankets
- Altimeter must have dual deploy capacity and the ability to log flight data

1.3.3. Design Considerations

- Rocket body will be made of fiberglass as it is easily accessible and will reduce weight
- Rocket will not exceed 2000 feet in flight in order to comply with local launch ranges

1.3.4. Safety Considerations

- Rocket simulations will occur before launch to ensure stability and safety
- The Rocket must comply with the safety guidelines provided by the NAR
- The rocket must be built with lightweight materials
- Only certified, commercially made model rocket motors are to be used
- The rocket must be launched with an electrical launch system and electric motor igniters
- The launch system must have a safety interlock in series with the launch switch
- Before launch, there must be a countdown and everyone must be at least 15 feet away when using D motors or smaller, and 30 feet when using larger rockets
- The rocket must be launched from a launch rod, tower, or rail that is pointed to within 30 degrees of the vertical
- If the rocket does not launch, the launcher's safety interlock must be removed, and a duration of 60 seconds must be counted before approaching the rocket
- The launch location will be in an outdoor, open space
- The recovery system will include a parachute or streamer, so the rocket returns safely and undamaged to the ground

1.4. Tasks

Table 1-1: Airframe and Recovery System (ARS) Sub Team Analysis Tasks

ARS Analysis Task 1:
Problem Statement:

Airframe design, fabrication and assembly of overall mechanical structure and integration of other subsystems (SolidWorks, ANSYS)
Solution Methodology:
• Building one innovative rocket design with separation system in nose cone • Use leftover material from last year’s model rocket
Analysis Products:
• OpenRocket for model rocket design

ARS Analysis Task 2:
Problem Statement:
Design, fabrication, and test of innovative recovery system (SolidWorks, MATLAB)
Solution Methodology:
• Recovery bay with 3D printed sled to mount electronics • Dual parachute deployment and dual end deployment rocket
Analysis Products:
• Use of SolidWorks for recovery bay

Table 1-2: Propulsion, Thermal and Separation System (PTSS) Sub Team Analysis Tasks

PTSS Analysis Task 1:
Problem Statement:

Selection, modeling, and test of (commercially available) motors for single and two-stage rockets as well as mounting and ignition system (SolidWorks, COMSOL, MATLAB, Cantera)
Solution Methodology:
• Selected Cesaroni I540-16A motor
Analysis Products:
• OpenRocket for motor selection • CAD for estimation of motor masses

PTSS Analysis Task 2:
Problem Statement:
Design, fabrication, and test of innovative stage separation systems
Solution Methodology:
• Nosecone separation system based on an electromagnetic system • Four pins mounted on linear rails that hold nosecone coupler and upper airframe together
Analysis Products:
• CAD for separation system • ANSYS to determine max deflection

PTSS Analysis Task 3:
Problem Statement:

Analysis of thermal loads from the motor(s) during flight
Solution Methodology:
• Calculated new properties, like temperature, pressure, density, specific heats, and mean molecular weight, that will define chamber properties used for future calculations • Use of last year's input file for the Cantera Toolbox
Analysis Products:
• Cantera Toolbox in MATLAB • Equation of exit velocity

Table 1-3: Flight Dynamics Analysis (FDA) Sub Team Analysis Tasks

FDA Analysis Task 1:
Problem Statement:
Analysis of aerodynamic loads on the vehicle during flight (FLUENT, MATLAB)
Solution Methodology:
• Calculate drag on nose cone during separation • Calculate max deformation on fins • Calculate tendency of roll coupling
Analysis Products:
• Equations of drag • AeroFinSim, XFLR5 and ANSYS • MATLAB and OpenRocket

FDA Analysis Task 2:
Problem Statement:
Selection, integration, and simulation of sensors for flight and vehicle dynamics, including accelerometers, gyros, and altimeter (MATLAB).
Solution Methodology:
• Develop flight computer that contains the following: accelerometer, rate gyros, magnetometer, barometer, and GPS
Analysis Products:
• Mass budget for comparison • Software to generate design

FDA Analysis Task 3:
Problem Statement:
Analysis of rocket performance (altitude, range, etc.) using commercially available software in support of design activities and flight planning. (MATLAB and specialized software TBD)
Solution Methodology:
• Calculate rate of change between time steps in barometer data • Predict apogee using Kalman filter and trigger events at predicted height
Analysis Products:
• MATLAB along with previously recorded flight data

1.5. Background and Literature Review

1.5.1. Model Rocketry Design Review

Successful model rockets are designed based on the principles of physics, aerodynamics, and thermodynamics. Physics models are necessary to predict various flight behaviors of the rocket. Aerodynamics forces and loads must be taken into consideration to guarantee stability in the rocket. Lasty, thermodynamics principles are used when considering different options of the motor of the rocket.

There exists various parts of a model rocket that need to be taken into consideration during the design process. Starting at the bottom of Figure 1-1**,** the motor is what accelerates the rocket upwards until burnout. The fins provide aerodynamic stability to the rocket. Inside the body tube are located different mechanisms including the flight computer which are responsible for measuring pressure and acceleration data. The parachute is an ideal way to recover a rocket; it releases at apogee carrying the rocket safely to the ground. Lasty, at the tip of the rocket exists the nosecone which is responsible for the aerodynamic drag on the rocket.

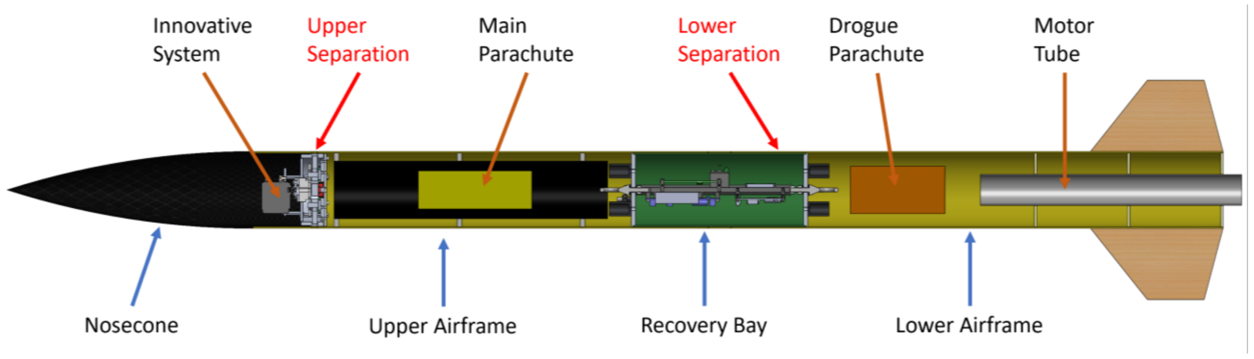
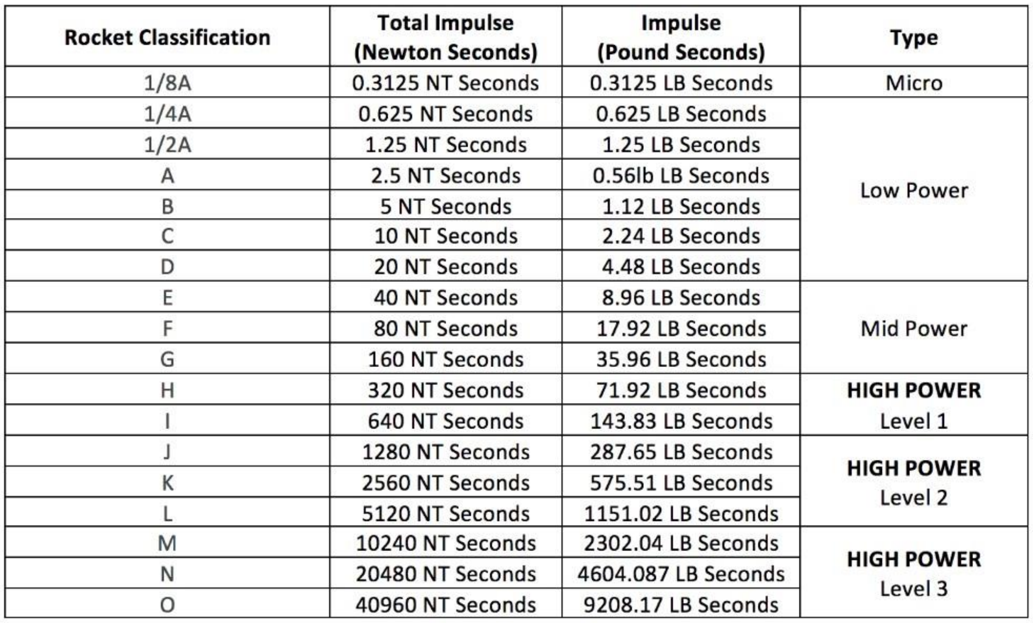


Figure 1-1: CAD Model of Team's Model Rocket

Model rockets are often categorized based off the size of their motor. The table of rocket motor sizes are below in Table 1-4. The more powerful the motor, the further along the alphabet the class is. Motors start at class 1/8A and may go all the way up to class T. Based on the project goals and requirements, our model rocket would be either a “H” or “I” motor.

Table 1-4: Rocket Motor Sizes [1]



Rocket Classification	Total Impulse (Newton Seconds)	Impulse (Pound Seconds)	Type
1/8A	0.3125 NT Seconds	0.3125 LB Seconds	Micro
1/4A	0.625 NT Seconds	0.625 LB Seconds	Low Power
1/2A	1.25 NT Seconds	1.25 LB Seconds	
A	2.5 NT Seconds	0.56lb LB Seconds	
B	5 NT Seconds	1.12 LB Seconds	
C	10 NT Seconds	2.24 LB Seconds	
D	20 NT Seconds	4.48 LB Seconds	
E	40 NT Seconds	8.96 LB Seconds	Mid Power
F	80 NT Seconds	17.92 LB Seconds	
G	160 NT Seconds	35.96 LB Seconds	
H	320 NT Seconds	71.92 LB Seconds	**HIGH POWER** Level 1
I	640 NT Seconds	143.83 LB Seconds	
J	1280 NT Seconds	287.65 LB Seconds	**HIGH POWER** Level 2
K	2560 NT Seconds	575.51 LB Seconds	
L	5120 NT Seconds	1151.02 LB Seconds	
M	10240 NT Seconds	2302.04 LB Seconds	**HIGH POWER** Level 3
N	20480 NT Seconds	4604.087 LB Seconds	
O	40960 NT Seconds	9208.17 LB Seconds	

1.5.2. Airframe and Recovery Systems

The design of the airframe and recovery systems, as well as their construction and assembly, fell to the ARS group. The scope of the ARS team primarily includes all outer pieces and any construction not considered payload or a motor. The selection and implementation of both a main and drogue parachute are also under the team's role.

The airframe of a rocket is what holds everything together. The design must withstand vastly different forces, from the engine's acceleration to wind resistance to ground impact. A successful frame is sturdy enough to survive the dynamic forces applied to it while being reliable enough to reach the target altitude and flight parameters. To meet these requirements, the appropriate material for the airframe and the aerodynamic parts, such as the nosecone and fins, must be explicitly selected for the planned parameters and tested.

The design, location, and implementation of parachutes will determine the rocket descent.

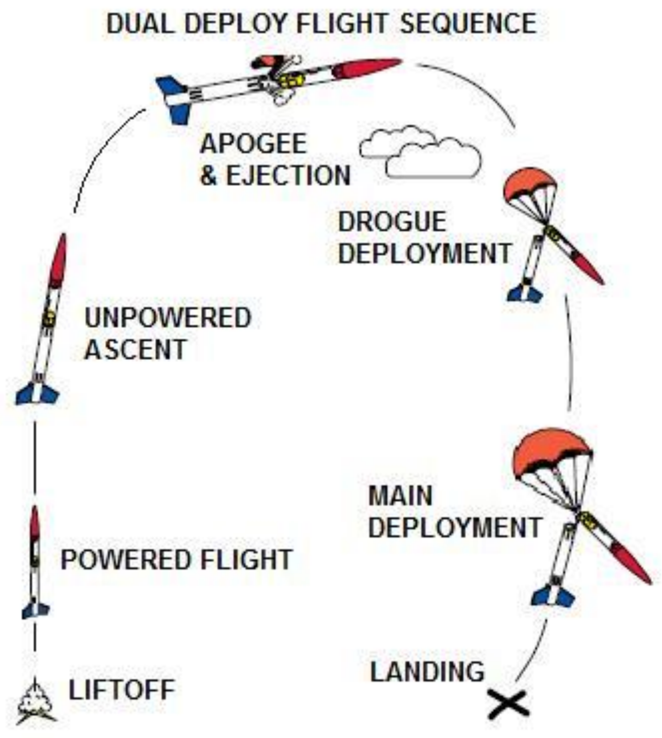


Figure 1-2: Diagram of Parachute Opening Sequence [2]

The main parachute is designed to slow the rocket to a safe velocity. "Safe," in this instance, means a velocity at which the rocket will not be damaged when it hits the ground. The design also includes a drogue parachute, which will deploy at apogee, prior to the main parachute as seen in Figure 1-2. The drogue's function is to slow down the rocket to a point where the main parachute can deploy without being compromised by aerodynamic forces. Drogue chutes are often smaller than the main, as they are not the primary method of slowing the entire airframe to a safe velocity.

Innovative Design Ideas Discussed

While brainstorming innovative ideas for the airframe and recovery system, the team looked for ways to make the rocket cheaper to build, easier to manufacture, and more convenient to fly. Some ideas the group came up with while brainstorming but did not consider beyond the idea stage were a variable diameter airframe, canards, airbrakes, and a Viking Lander-style airbag recovery system. The transitional diameter airframe would not have provided enough benefit to warrant the additional complexity it produced. The canards and airbrakes have been used on previous MQP rockets or High Power Rocketry Club rockets, so the team chose not to pursue them and focus on a different innovative design. The airbag recovery system would have been very complicated to develop and is a greater risk than the team was willing to accept in an actual design.

The team considered parachute reefing, which uses a mechanism to hold the main parachute closed when it is first deployed. Keeping the parachute closed reduces the drag the parachute produces, mimicking a drogue parachute. At a lower altitude, the mechanism holding the parachute closed releases, allowing it to expand to its full size. We considered reefing the main parachute because it is a single-end deployment and only requires one parachute. Additionally, with the use of the innovative stage separation mechanism, there would not need to be more than one separation mechanism. However, parachute reefing is a single-end deployment, which is a greater risk than we were willing to take with an untested stage separation mechanism.

Another innovative idea considered was an interchangeable fin-can, which would allow the team to quickly change the size of the motor tube and fin shape. An interchangeable fin-can would enable the team to prepare different motor tubes for different motors, which would help us find a launch site nearby since we could change motors quickly. Moreover, it would aid in reusability because it would allow the team to swiftly change parts in case something on the first fin-can was damaged after the flight. However, the group decided against an interchangeable fin-can because it would be difficult to conveniently connect to the rocket without developing a complicated attachment method. Additionally, the attachment method would add additional weight to the rocket, making motor selection more difficult. Lastly, in order to properly benefit from the interchangeable nature of the fin-can, the team would need to manufacture two fin-cans, which would add an additional cost and limit how much we could spend on other aspects of the rocket.

1.5.3. Flight Dynamics and Analysis

Apogee detection is essential in model rocketry because it determines when the parachute should be deployed for recovery. An early or late parachute deployment may negatively affect the recovery stage due to various forces acting against the parachute. An ideal method to detect apogee would be to integrate acceleration to obtain velocity, however this method typically results in high error if the rocket is not launched at a perfect vertical angle. The measured acceleration must be subtracted from the acceleration due to gravity which becomes difficult when the rocket is launched at an angle. This results in errors in the integrated values which only accumulate as time increases. The various forces need to be considered when detecting apogee are in Figure 1-3.

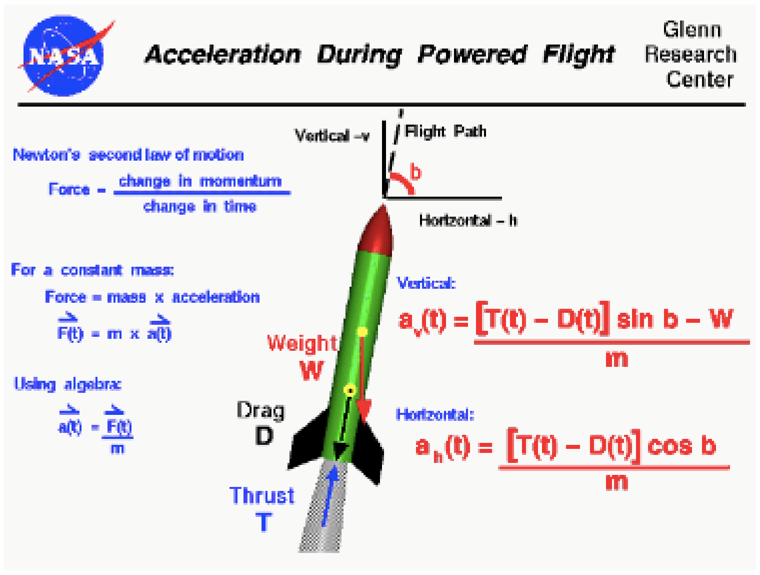

Acceleration During Powered Flight
Glenn Research Center
Newton's second law of motion
Force = change in momentum / change in time
For a constant mass:
Force = mass x acceleration
$\vec{F}(t) = m \times \vec{a}(t)$
Using algebra:
$\vec{a}(t) = \frac{\vec{F}(t)}{m}$
Vertical - v
Flight Path
b
Horizontal - h
Vertical:
$a_v(t) = \frac{[T(t) - D(t)] \sin b - W}{m}$
Horizontal:
$a_h(t) = \frac{[T(t) - D(t)] \cos b}{m}$
Weight W
Drag D
Thrust T

Figure 1-3: Diagram of Acceleration Force Acting on Rocket [3]

1.5.4. Propulsion, Thermal, and Separation Systems

Propulsion/Motor

The motor provides thrust to the rocket using various chemical reactions to heat up a material to cause it to expand and be ejected out of the motor. A nozzle is used to accelerate the flow of the hot gases, shown in Figure 1-4 below; the force of the hot exhaust gases on the rocket nozzle is what creates the thrust for the rocket.

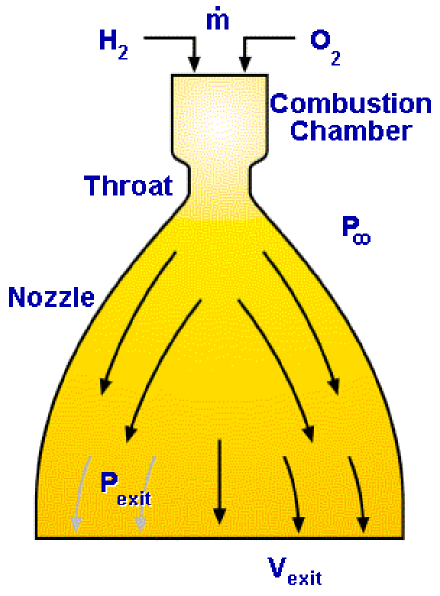


Figure 1-4: Basic Cartoon of a Nozzle on a Liquid Rocket Motor Source [4*]*

There are two major types of rocket engines images of these in Figure 1-5. The most common type is the liquid rocket motor. These rocket motors were used on SpaceX Falcon 9, Saturn V and SpaceX Falcon Heavy to name just a few [5]. Since liquid propellant has a large specific impulse (Isp), it is ideal for launching spacecraft off the planet quickly as well as having more fine control over the amount of thrust being produced (throttling) [6]. Liquid rocket motors are the most common rocket motor used but they are not the only one. The other type of rocket engine is the solid propellant rocket motor. These rocket motors use a solid fuel and oxidizer in the combustion chamber rather than liquid fuel. Solid rocket motors are usually used for smaller rockets where throttle variation is not very important; they can be used for missiles, satellite boosters, and model rockets [7]. The flame from inside the motor allows for hot gasses to be formed which, much like the liquid rocket motor, is ejected out the nozzle to create thrust [7]

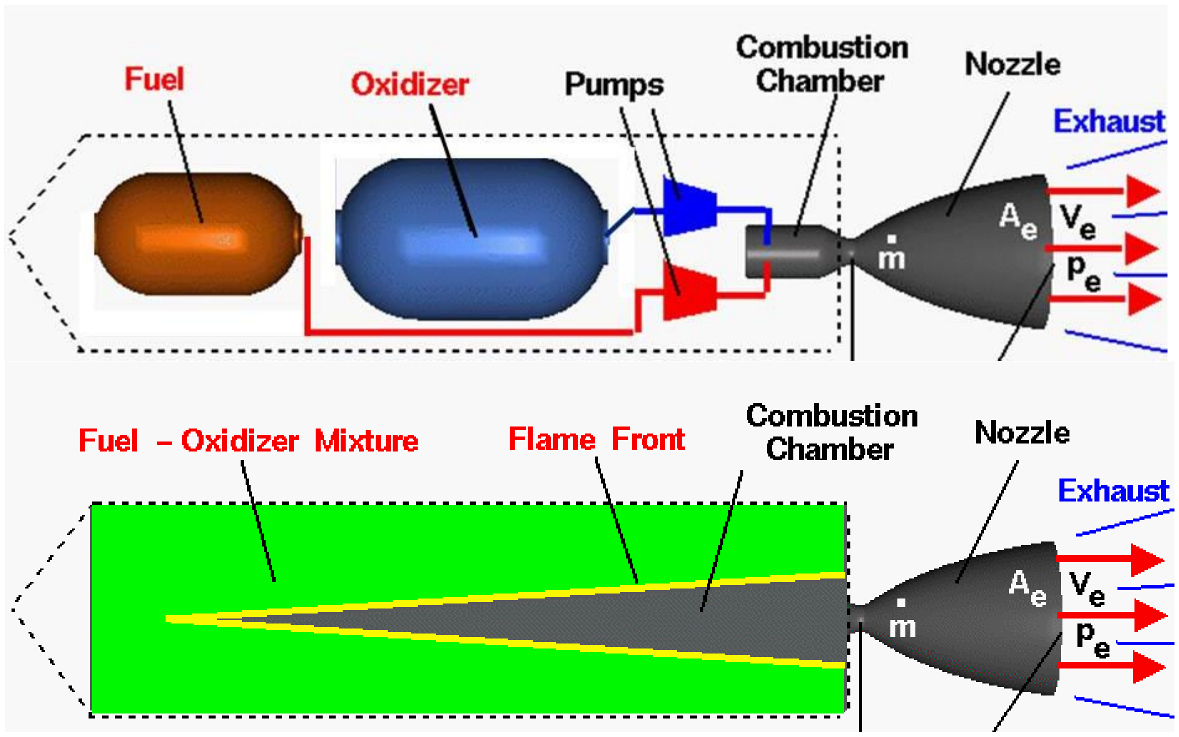


Figure 1-5: Images of Solid (Bottom) and Liquid (Top) Rocket Motors [7]

There are several types of solid rocket motors that can deliver different types of thrusts and Isps over various times.

Separation

The separation system in a model rocket is designed to allow the recovery system to be ejected from the rocket. This recovery system allows a slow and controlled descent back to the ground without posing a danger to anyone. For a separation system to be successful, there needs to be enough force to break apart the airframe of the rocket when desired. Typical model rockets use either a compressed CO2 charge or black powder charge.

Compressed CO2 charge is when a CO2 canister is placed inside the rocket and punctured when the separation is desired. This great force induced by the CO2 being released is strong enough to separate the airframe to allow recovery system to be deployed.

Black powder charge uses black powder and an ignition source to blow the rocket into two pieces. Black powder can be more dangerous since it is possible for it to damage the recovery system because it is igniting. Black powder charges are cheaper than compressed CO2 which makes them ideal if on a tight budget.

1.6. Design Overview

Figure 1-6 shows the final CAD assembly for the rocket. The innovative stage separation system is housed in the nosecone shoulder tube. For both the baseline and innovative rockets, the upper airframe separates from the nose cone. Inside the upper airframe is the inner tube, which was added to prevent the parachute and shock cord from getting stuck on the springs for the innovative stage separation. The upper airframe holds the main parachute, which deploys at 500 feet above ground level. At the other end of the upper airframe is the coupler tube, which houses the recovery bay. The coupler tube has a switch band on it, which allows the arming switches to be armed once the rocket is assembled and sits between the upper and lower airframe. The connection between the coupler tube and the lower airframe is where the lower separation point is. The lower airframe holds the drogue parachute, which deploys at apogee. The end of the lower airframe is where the motor mount is attached to the rocket. The end of the lower airframe is also where the fins and fiberglass layup are located.

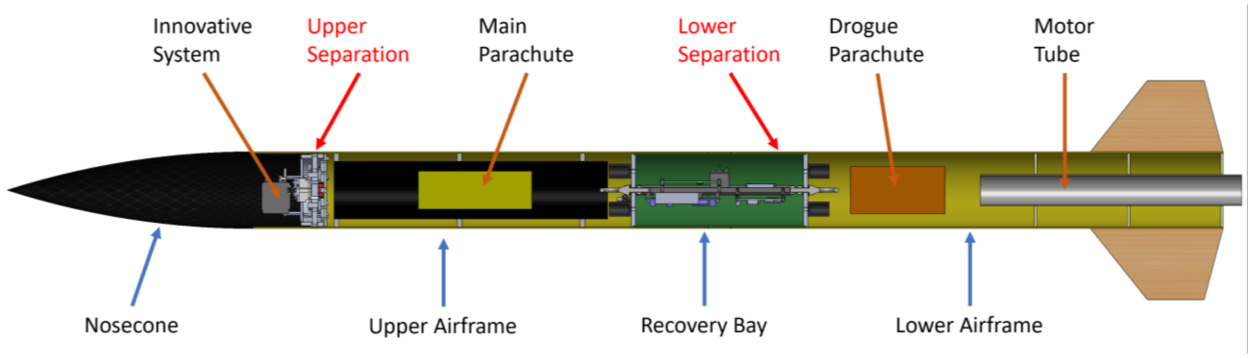


Figure 1-6: Annotated Rocket CAD

Both the baseline and innovative rocket used the same airframe, to minimize construction costs and complexity. For the baseline rocket, the innovative assembly was left inside the nosecone to maintain weight and stability, however a black powder separation method is used. The innovative assembly pins are prevented from extending during baseline use, and the nose cone and upper airframe are connected using shear pins.

Figure 1-7 shows the final OpenRocket schematic of the innovative design. This design includes the innovative spring separation system within the nose cone and two parachutes, the main located in the upper airframe and the drogue in the lower airframe. The main parachute is housed within an inner airframe connected to the innovative separation system and the recovery bay. The drogue parachute sits within the lower airframe and is connected to the recovery bay and the motor tube centering rings. The recovery bay holds two altimeters and a custom flight computer. A fiberglass layup is added to the fins on the exterior of the rocket. The upper airframe is screwed onto the recovery bay to prevent separation. In contrast, the nosecone and lower airframe are connected to the upper airframe and recovery bay by shear pins.

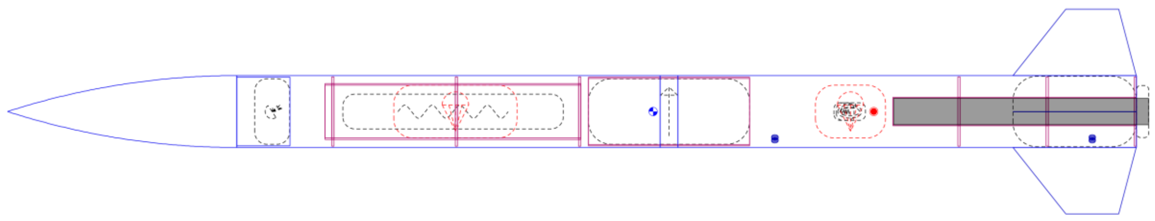


Figure 1-7: Final Rocket Design in OpenRocket

2. Airframe and Recovery Systems

2.1. Methodology

2.1.1. Airframe

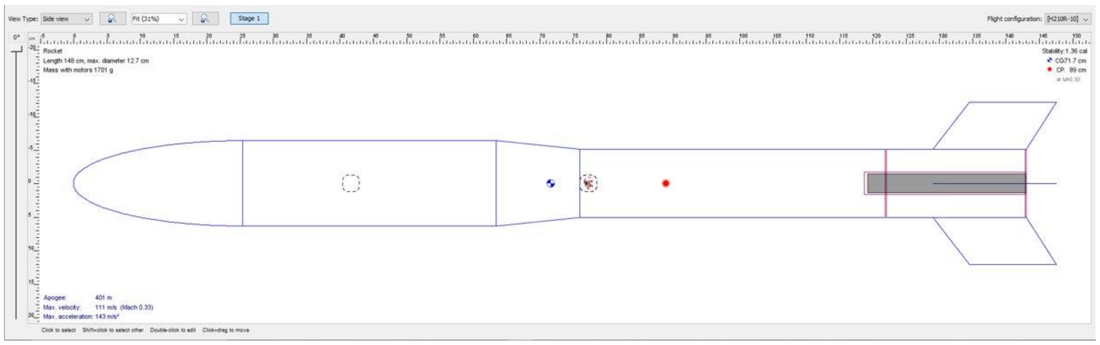
The team's first step in creating a high-powered model rocket was to choose what software to use for the baseline design of the vehicle. The group decided to use OpenRocket as the design software. OpenRocket consists of a vehicle construction page, a motor selection menu, and a simulation page. The construction and design page allows users to create custom rockets. OpenRocket has a list of standard parts, such as nosecone, body tube, fins, and parachute which can be customized in shape, size, material, and purpose. Once the overall design is complete, the user can select a motor mount and motor. OpenRocket gives a list of engines and their specifications. The simulation page can be adapted with different motor selections, weather conditions, and launch options to simulate different scenarios. This flexibility is beneficial for designing a high-powered rocket, as any changes made to the vehicle can be updated in the simulation to see how the expected performance changes. OpenRocket was selected primarily because most group members had previous experience with the software.

There were several constraints that the team had to decide on at the beginning of the design phase. The group first needed to consider how many rockets to build in total. There were two options: build a single innovative rocket, or both an innovative and baseline vehicle. The benefit of having two rockets is that removing hardware and problem-solving is unnecessary if the innovative system is not cleared to fly or has problems before launch. Instead, the team launches the baseline design with standard equipment simulating the payload. On the other hand, a single rocket minimizes the workload and allows for the complete focus on a single vehicle. The team chose to only build the innovative design to reduce the complexity of the project, especially during the construction and assembly phase.

A second constraint was whether to use leftover components from previous years' projects. Excess airframe parts consisted of a 4-inch and 5-inch body tube, a 4-inch ogive nose cone, and two 4-inch coupler tubes, all made from fiberglass. There were also three parachutes, two with a diameter of 36 inches and one with a diameter of 54 inches, along with various shock cords and Nomex blankets. The team decided to use the already available parts to reduce cost and complexity.

Outer Airframe

Once the team had decided on the constraints, the next step was to submit preliminary OpenRocket designs. There were two designs that were put forward, with the second being selected as the final. The first design, shown in Figure 2-1, was a multi-diameter airframe with a 5-inch upper body tube for the payload section and would transition down to 4 inches for the lower body tube and motor mount. The OpenRocket simulation was basic and used to show the overall possible look of the rocket.



```
Rocket
Length 148 cm, max. diameter 12.7 cm
Mass with motors 1701 g

Stability: 1.36 cal
CG:71.7 cm
CP: 89 cm
at M=0.30

Apogee:           401 m
Max. velocity:    111 m/s  (Mach 0.33)
Max. acceleration: 143 m/s²
```

Figure 2-1: Open Rocket Design with Variable Diameter Airframe

The transitional diameter did not see much development, as there were several issues with the fundamental design. First, the nose cone that the team already possessed was 4 inches in diameter, and a 5-inch upper body tube meant that it would be necessary to find and purchase a new one. The second major problem is that the two diameters on the rocket's exterior would complicate future simulations. While OpenRocket can perform the necessary initial simulations, the transitional design makes it more complicated to perform specific analyses on other software.

The second design, seen in Figure 2-2, made use of the single 4-inch diameter nosecone, upper and lower body tubes, and coupler that the team already had. After some refining, the group chose this second design to build. This second design consists of an upper body tube of 24 inches long and a lower body tube of 30 inches. The coupler is currently 10 inches long, but the design allows it to be shorter if necessary. The motor tube is around 10 inches long, with a 1.4-inch (36 mm) diameter motor mount to allow for more rocket motor options. The upper 54-inch diameter parachute acts as the primary, while the lower 36-inch parachute separates from the coupler via a black powder system and acts as the drogue.

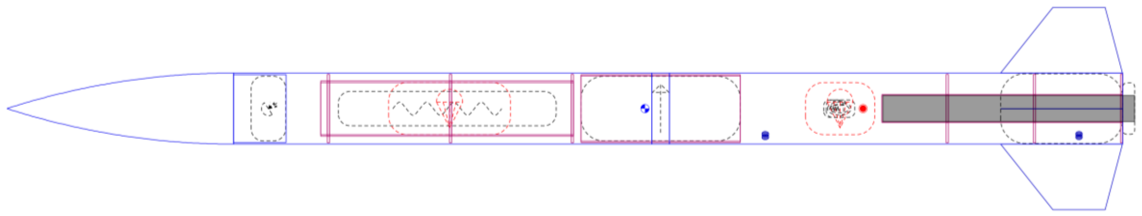


Figure 2-2: Final OpenRocket Design

This design includes the innovative separation system in the nose cone. The rocket is specified to carry up to 3 pounds of equipment, 1 pound each allotted for the recovery electronics, separation system, and flight computer. Some parts of this design are not final, such as the selected motor, the fins, and the recovery bay.

Inner Airframe

With this design, the upper airframe requires an inner airframe to store the main parachute and support the innovative separation system. This 3-inch diameter carbon-fiber inner tube was one of the unused parts from a previous design project. The tube was cut to 14.5 inches to fit within the upper airframe. Three plywood discs were laser-cut from 1/4-inch plywood and epoxied to the tube. After dry fits of the innovative system, the inner tube was epoxied within the upper airframe.

2.1.2. Fins

The team decided for the rocket to contain four trapezoidal shaped fins made of plywood based on ease of analysis. In preparation for the attachment of the fins, the surface of the airframe was prepared by being thoroughly sanded. The fins were then epoxied onto the airframe using a fin jig to ensure the fin's straightness while the epoxy cured overnight. The type of epoxy used to make the fillets was specifically designed with a high viscosity for the purpose of mounting fins to rockets. Having the consistency of a thick paste allowed the epoxy to be worked into a specific radius. The airframe tube was then wrapped with a fiberglass mesh to ensure strength.

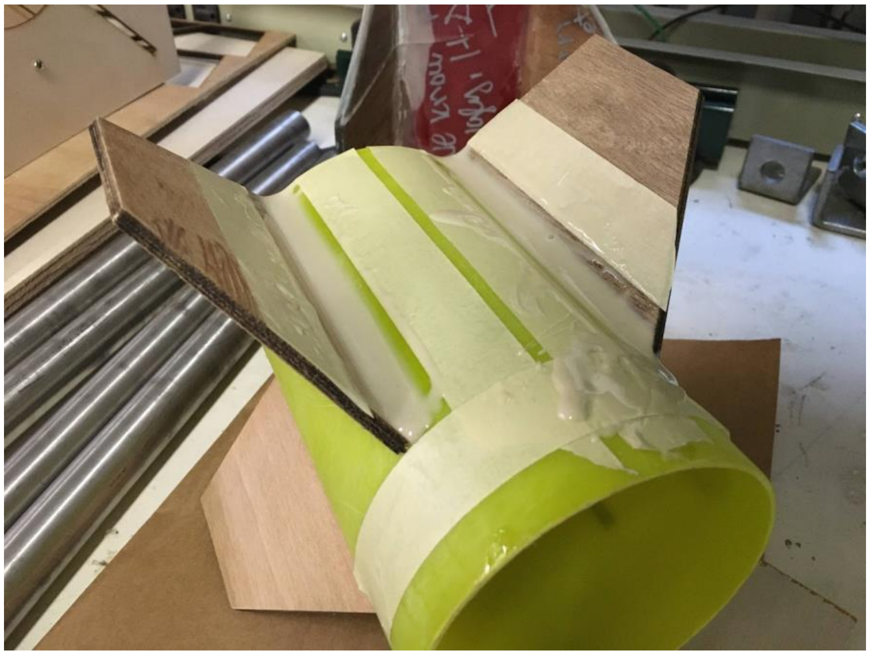


Figure 2-3: Practice Fillets on a Spare Piece of Airframe Tube

After more surface preparation, the fiberglass layup was then applied. A fiberglass layup is made up of a layer of epoxy and fiberglass mesh fabric that was cut in the shape of the fin and the area between the fins. After an initial coat of epoxy, the fiberglass fabric was laid on and covered by a second coat of epoxy, ensuring the fabric was completely saturated. After the fiberglass layups were complete, the airframe was then wrapped in a sheet of release film along with a sheet of breather. The purpose of the breather is to absorb the excess epoxy, while the release film ensures that the fin separates from the breather once cured. Both these wrappings gave the layup a smooth finish. After multiple attempts, a compression method was acquired to ensure that the epoxy and fiberglass mesh cured flat against the fin. A vacuum bag was used for this process. Once the epoxy cured, the excess fiberglass fabric was cut off the edges and sanded smooth.

There were several fiberglass layup attempts prior to the final fin application on the body tube. Three different methods were used to find the most effective way to apply the layup that minimized air bubbles, overlaps, and slipping of the sheets. The first layup attempt involved a body tube from a previous project and laser cut plywood fins of the same proportions as the final planned airframe. This attempt was completed as a proof of concept and to discover what issues might arise during the actual application. The fiberglass fabric was cut to hang over the fin which would allow them to be pasted together and then cut off. The layup was completed in two different pieces, with one sheet being put between two fins and the other being wrapped around the other side of the fin completely. This was to find out which method was more effective at covering the fins and body tube. Also of note was that the epoxy fillet was done by hand and had no standardization. Once completed, the results of the first layup attempt showed issues with the layup method used. The fabric overhang caused an issue where it folded on top of the wing itself. This was not able to be cut which proved that a minimal overlap was required. The larger wrap-around piece had considerably more bubbles than the single section, so the team decided to break future layups into four separate pieces. The nonstandard epoxy fillets also created large voids, which resulted in the decision to laser cut a fillet stick.

The second attempt was completed with a smaller body tube and fin size. This attempt had several improvements compared to the last. A fillet stick was used to make each fillet even, and in addition, the epoxy used for those fillets was changed to RocketPoxy for a stronger bond. Each section of the fiberglass was cut out to match the profile of the fins and body tube, and there were extra precautions taken to prevent epoxy from leaking into areas such as the inner part of the tube where it could cause problems with assembly. The resulting layup was improved, but still had issues that needed to be addressed. Epoxy was not evenly spread on some areas even with the fillet stick, and that was addressed by taping off sections of the rocket to prevent spillage. Again, the layups shifted slightly during wrapping of the spacer fabric, so a new method of clipping the fabric to the fiberglass using binder clips would be implemented. The fabric overhang, even though it was smaller, still had problems of folding over the fins and was unable to be sanded. In addition, air bubbles and voids still appeared in the layup, although significantly less than the first test.

In addition to providing more methodological improvements, the second layup test was unscientifically stress tested to find the possible breaking point of the fiberglass. This was not meant as an actual measuring method but more of an approximation to learn if the layups were as secure as the team believed. While waiting for the third layup to dry, the ARS team performed 20 drop tests consecutively, by simply picking up the layup, bringing it up to head height, and dropping it. When the layup showed no signs of deterioration, the approval was given to break it by standing on top of it. The layup successfully held up one member of the team, and only broke down after jumping up and down on it. Even then, the layup held together and instead the entire epoxy fillet had sheared off the body tube, proving that a single coat of the fiberglass layup could hold approximately 180 pounds of constant force on it. This gave the team confidence that the final layup with its planned three layers would hold during any ground impact scenario.

The issues above were addressed in the third and final layup, where the final methods for fiberglass layups were planned out. Each piece of fabric was precisely cut to the dimensions of the fins to prevent any overhang. Application of the layups and spacer fabric were much more carefully done to prevent the formation of air bubbles underneath. Two layers of fiberglass fabric were placed on the layup to prepare for the final planned three on the body tube. This attempt involved fins and a body tube of a similar size at the actual rocket to simulate the final piece, although the test body was made of a cardboard derived material. The third layup was successful in most aspects, and only two more changes were made to the methodology of application. First, each fiberglass would be marked exactly to the center of the fins for proper lineups and preventing overhang, and second, the spacer fabric as well as the fiberglass layup would be placed on the body tube centerline first, and then spread outwards to the fins. These changes minimized the number of bubbles and voids in the final application and eliminated any issues with overhang or sliding.

Fin Jig

The purpose of the fin jig is to ensure the rocket fins are mounted onto the airframe as straight as possible. The fin jig was designed in Solidworks using the 3D model of the lower tube.

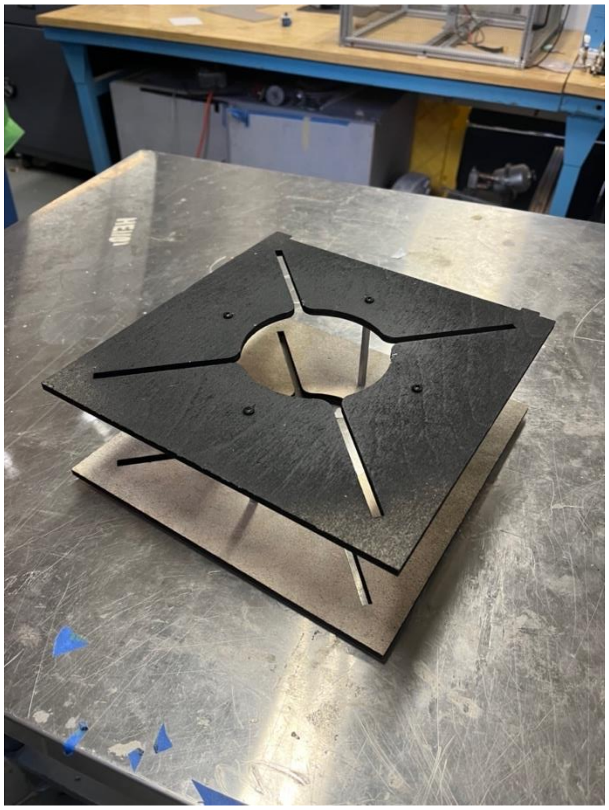


Figure 2-4: Picture of Constructed Fin Jig

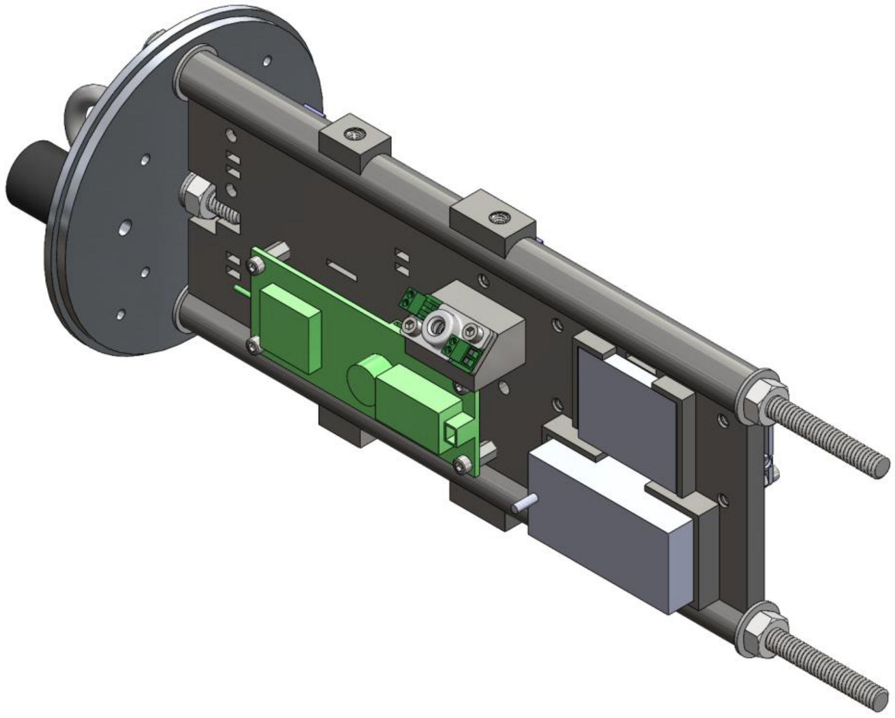
After adding tolerance, the final fin jig design became a snug cross shape with a hole in the center for the body tube on an 11-inch by 11-inch plate as seen in Figure 2-4. The design was laser-cut from 1/4-inch plywood to create the two plates and assembled using four 3-inch standoffs in between the plates. Due to the fit of the jig, the fins could be epoxied to the tube with minimal error. After the fiberglass layup, the jig was used as a stand throughout the build process.

2.1.3. Nosecone

Past design projects left behind two nose cones compatible with the current airframe diameter: one an ogive and one conical. Instead of designing a novel nose cone, using one of these previous nose cones allowed more budget to be allocated to the innovative system. The fiberglass ogive nose cone, weighing 1.1 pounds and measuring 16.5 inches from base to tip, was selected due to a smoother transition between the body tube and the tip. This is result of the curve at the base of the nose cone being tangent to the airframe. In contrast, the conical nose cone would have a sharp change in angle at the transition to airframe, giving it a higher drag coefficient than the ogive. In addition, the ogive nose cone gave more space to work with when designing the innovative separation system, which was located in the nose cone coupler.

2.1.4. Recovery bay

The recovery bay is the area of the rocket that houses the flight electronics that control the rocket's separation and the parachutes' deployment. Inside the recovery bay sits the recovery sled, which the recovery electronics are mounted onto. The recovery bay is the entire assembly of the recovery sled, coupler tube, and bulkheads, while the recovery sled only refers to the plate the altimeters and batteries are mounted to. The coupler tube’s outer diameter matches the inner diameter of the airframe, allowing the two to fit together. A switch band is added to the coupler tube to prevent the upper and lower airframe from contacting one another and allowing switch holes to be drilled into the coupler tube. Machined aluminum bulkheads were fitted to each end of the coupler tube and rest on the rim of the coupler tube. Each bulkhead has an eyebolt to attach to the parachute shock cord. The bulkheads are held in place with two threaded rods, preventing the recovery sled from rotating, and allowing nuts to tighten the bulkhead to the coupler tube. A smaller 3D printed plate was mounted beneath the aluminum plate to ensure the recovery sled remains centered in the coupler tube. Charge wells were mounted to the bulkheads to hold the black powder used for stage separation and parachute ejection.



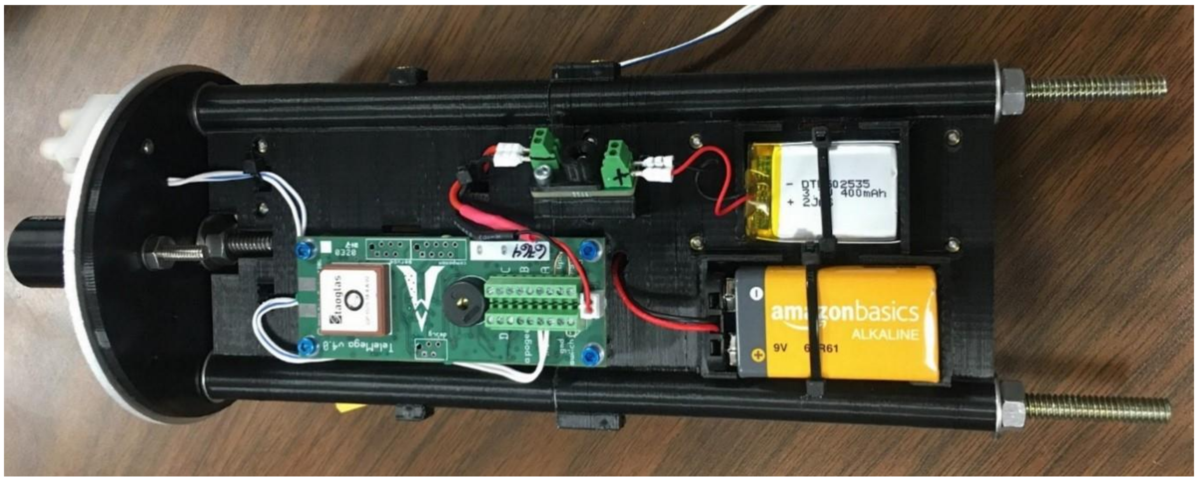


Figure 2-5: Recovery Bay. Top: CAD; Bottom: Final Assembly

The recovery sled, as seen in Figure 2-5, holds the two altimeters, the custom flight computer, and the batteries for the electronics. The sled was 3D printed with two holes running along the length of it, allowing the sled to fit onto the threaded rods holding the bulkheads to the coupler tube. One end of the sled is flush with the top bulkhead, while locknuts are tightened against the lower end of the recovery sled. Switch mounting blocks ensure the arming switches are tangent to the coupler tube, meaning the hex key used to arm the switch remains perpendicular to the coupler tube.

Threaded inserts were added to the sides of the recovery sled that prevented the recovery sled and bulkheads from rotating inside the coupler tube, and a second set of threaded inserts were added to allow a bolt to connect the upper airframe and coupler tube.

Wiring Diagram

To ensure the wires were properly accounted for in the design of the recovery bay, wiring diagrams were created. The wiring diagram shows the electronics, batteries, and wires in a simplified manner to verify all components have been accounted for in the design of the recovery bay. Additionally, the wiring diagram serves as instructions for wiring the electronics, as the components are arranged in the wiring diagram the way they would be in the recovery bay. The wiring diagram for one side of the recovery bay can be seen in Figure 2-6.

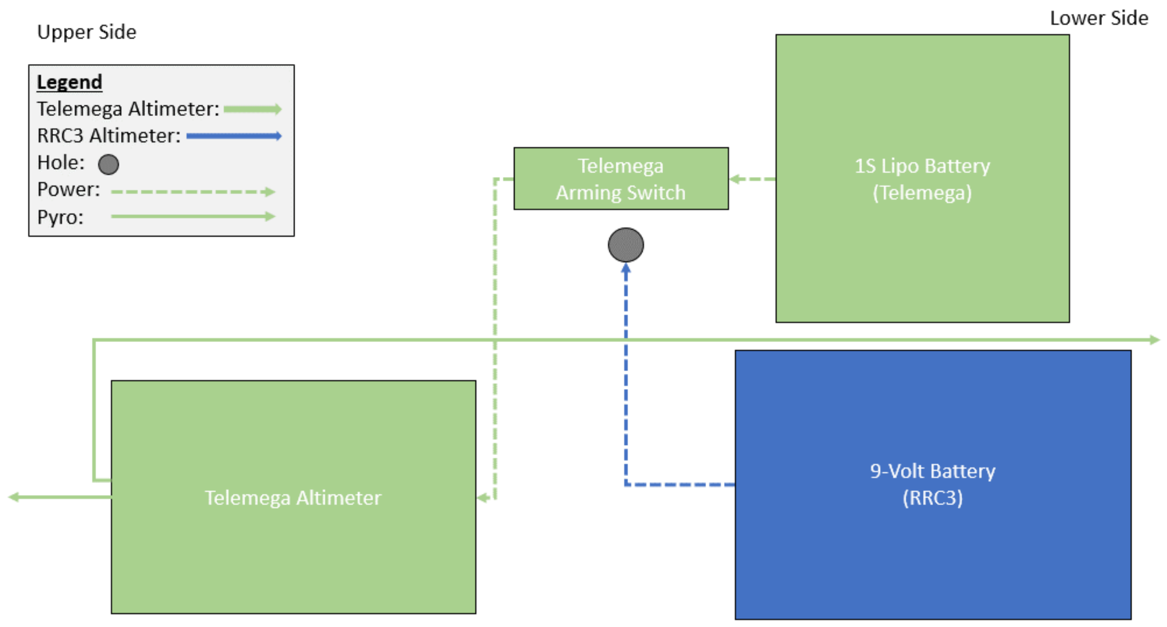


Figure 2-6: Left Side Wiring Diagram

In Figure 2-6, the green components represent the hardware associated with the Telemega altimeter while the blue components represent the components for the RRC3 Sport altimeter. The dashed line signifies that the wire carries power from the battery to the altimeter and the solid line indicates the wire carries the ignition charge used to fire the e-match. The gray circle indicates a hole in the plate to allow wires to pass through to the other side of the recovery bay.

The power from the battery passes through the arming switch, rather than being wired directly to the corresponding altimeter. The commercial off the shelf (COTS) altimeters have switch terminals and battery terminals on the board with the intention that the battery will be directly wired to the board, however our team decided it was safer to have a hard disconnect from the power source to further reduce the chance of an error occurring, such as an accidental firing of an e-match.

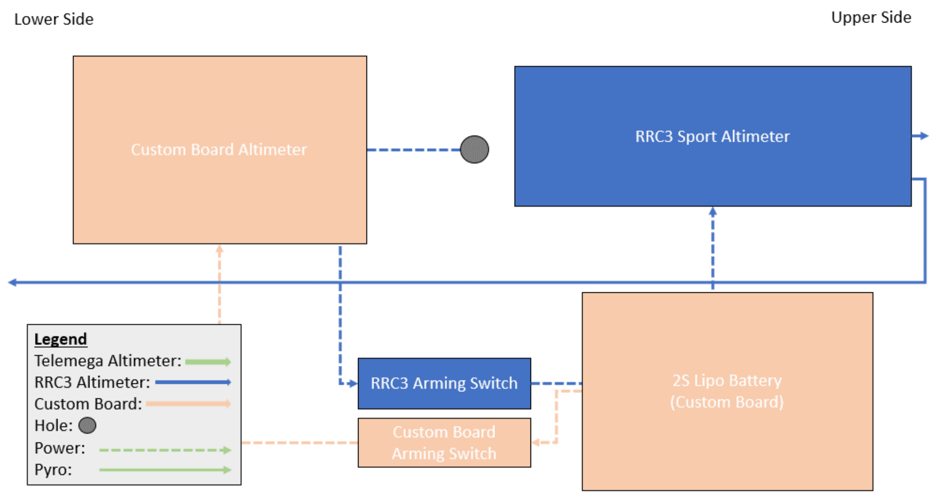


Figure 2-7: Right Side Wiring Diagram

Figure 2-7 shows the wiring diagram for the other side of the recovery bay. The tan components represent the switch mount, and wire used to power the custom board.

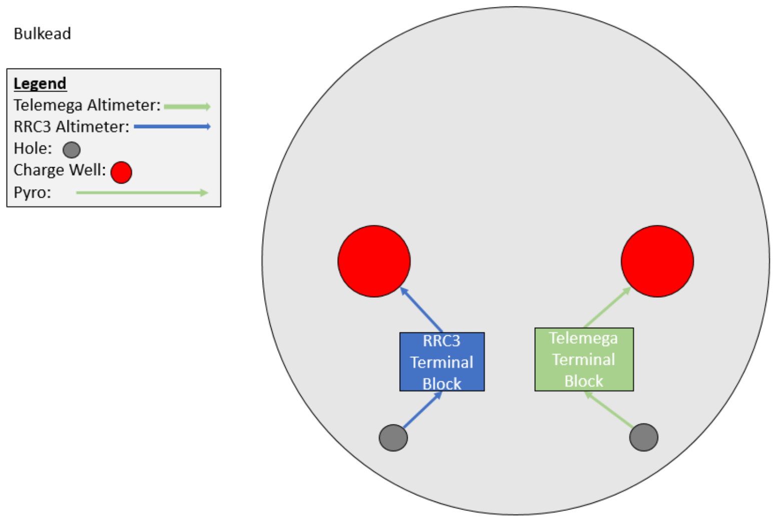


Figure 2-8: Bulkhead Wiring Diagram

Figure 2-8 shows the bulkhead plates since they are wired identically. The terminal blocks are the point where the wires carrying the charge from the altimeter meet the e-match wires used to detonate the black powder charges. The red circles are the charge wells that hold the black powder used in stage separation.

Altimeters

The team had two main constraints when choosing a commercial altimeter design. The altimeter needed dual deploy capability and the ability to log flight data. The cost, size, and setup of the altimeter must also be considered. Most commercial altimeters on the market today meet both primary qualifications, which allows for a large selection.

The team chose to use two different brands of altimeters to decrease the likelihood of both altimeters failing due to the same root cause. The team had access to a TeleMega 4, as seen in Figure 2-9, altimeter from a previous year's project. This altimeter is one of the largest on the market at 3.25 inches by 1.25 inches. This altimeter is capable of four additional pyro events with two accelerometers, has an onboard GPS receiver, a magnetic sensor, a barometric pressure sensor, and gyros. This altimeter is powered by a 1s LiPo battery. The TeleMega hit both primary qualifications, and the team's next step was testing the altimeter to ensure it was in working order. Once verified that the altimeter did work, the group decided to use the TeleMega due to already knowing the specifications and being able to save money. The next task was to assess and choose a second altimeter.

The team researched multiple different altimeters that meet the primary qualifications. Some altimeters could not be purchased because the product was either out of stock or no longer produced, such as the Raven 4 and the StratoLogger. Other altimeters, such as those from Eggtimer, required more setup after purchase, which the group agreed was an unnecessary complication. For both mass budget and design preference, the group decided on an altimeter that was close in size to the TeleMega that met all the qualifications. This second altimeter is the RRC3, as seen in Figure 2-10, Sport Altimeter by Missile Works. At 3.92 inches and 0.925 inches, the RRC3 is slightly longer and thinner than the TeleMega, which is 3.25 inches by 1.25 inches. The RRC3 is capable of dual deployment with an additional pyro event and has a barometric pressure sensor. While this altimeter may have fewer functions than the TeleMega, the RRC3 has everything required for the rocket and flight computers. In addition, the cost of the RRC3 was relatively low compared to other commercial altimeters and was available for purchase. Both companies manufactured their altimeters to run off a range of batteries, but the RRC3 is more optimized for 9V.

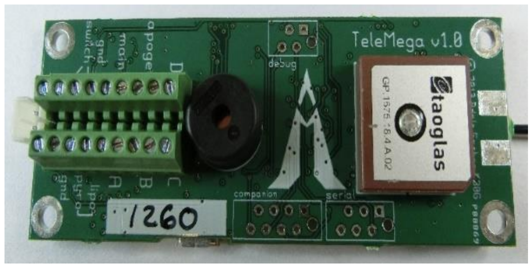


Figure 2-9: TeleMega v1.0 Altimeter [8]

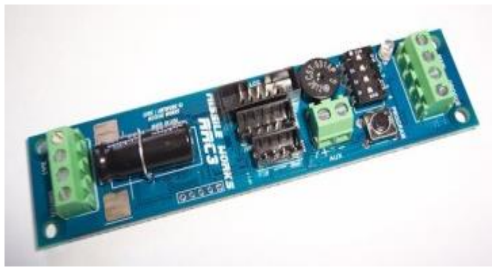


Figure 2-10: RRC3 Altimeter [9]

Recovery Parachutes

Model rockets generally have two parachute styles, single parachute deployment or dual parachute deployment. Single parachute deployment has only one parachute that releases at apogee and is generally used for smaller rockets. On larger rockets, the size of the parachute would allow the rocket to drift considerably after it's deployed, making retrieval difficult. Additionally, deploying a large parachute at high velocity produces a tremendous force on the bulkhead. Dual parachute deployment utilizes two parachutes, a drogue parachute and a main parachute. The drogue parachute deploys first, generally at apogee, and slows the parachute down as it descends to a safe, but relatively fast speed. This faster fall reduces how far the rocket will drift during its descent. Additionally, the deployment of a smaller parachute produces a smaller force than the deployment of a larger parachute at the same speed. The main, larger parachute is deployed closer to the ground to slow the rocket down to a safe velocity before it lands. The force produced by the deployment of the main parachute is reduced because the descent rate is reduced by the drogue parachute.

If using dual parachute deployment, there are two different methods to deploy the parachutes: single-end deployment and dual-end deployment. Single-end deployment has one separation point, and all the parachutes deploy out of the rocket at that point. The drogue will deploy first if the rocket has a drogue parachute and a main parachute. At the deployment altitude for the main parachute, a mechanism inside the rocket will release the main parachute. In some cases, the force of the drogue parachute may be enough to pull the main parachute out; however, an additional set of black powder charges may need to be used to eject the main parachute. Dual-end deployment has the rocket separate at two points, meaning the parachutes will come out of two separate points in the rocket. The first separation will separate the rocket and deploy the drogue parachute. A second set of separation charges at a lower altitude will detonate and separate the rocket at a different point, ejecting the main parachute.

The team's rocket is a dual parachute deployment and dual-end deployment rocket. Dual parachute deployment was chosen because it reduces the deployment forces on the parachute mounting plate and reduces the rocket's drift distance, which improves the likelihood of being able to recover it. Dual-end deployment was chosen because it works better with the innovative separation mechanism design. Since the innovative stage separation mechanism is untested, a single-end deployment rocket would be a greater risk since a potential failure of the separation mechanism would mean no parachute is deployed.

The main parachute is the one that will be deployed from the innovative stage separation mechanism, since it was deemed safer to have just the drogue parachute deploy than just the main parachute, should the innovative separation mechanism fail. Having just the main parachute deploy without the drogue parachute would produce tremendous forces on the mounting plate, potentially damaging the mounting mechanism. Only the drogue parachute deploying would still be dangerous, but it would be less risk of failure than just the main parachute deploying, as the descent would be slowed.

Shock Cords

Once separated, the three sections of the rocket are held together by shock cords, one cord connecting the nose cone to the upper airframe, the other connecting the upper airframe to the lower airframe. During ascent, the cords are stowed alongside the parachutes in the inner body tube bundled in a z-fold. These cords are one-inch tubular Nylon climbing ropes rated for up to 18kN of force, and each cord is twenty feet in length. It is important that the length of each shock cord be long enough so that the cord is not pulled taught from the force of separation, but short enough that the entire cord and parachute are pulled completely out of the rocket body.

The length of shock cord needed was determined by ensuring the components could not rotate into each other under the worst-case scenario. The components and shock cord lines were represented horizontally with semicircles to represent the maximum rotation of the component about the parachute attachment point. The diagram for the drogue shock cord can be seen in Figure 2-11

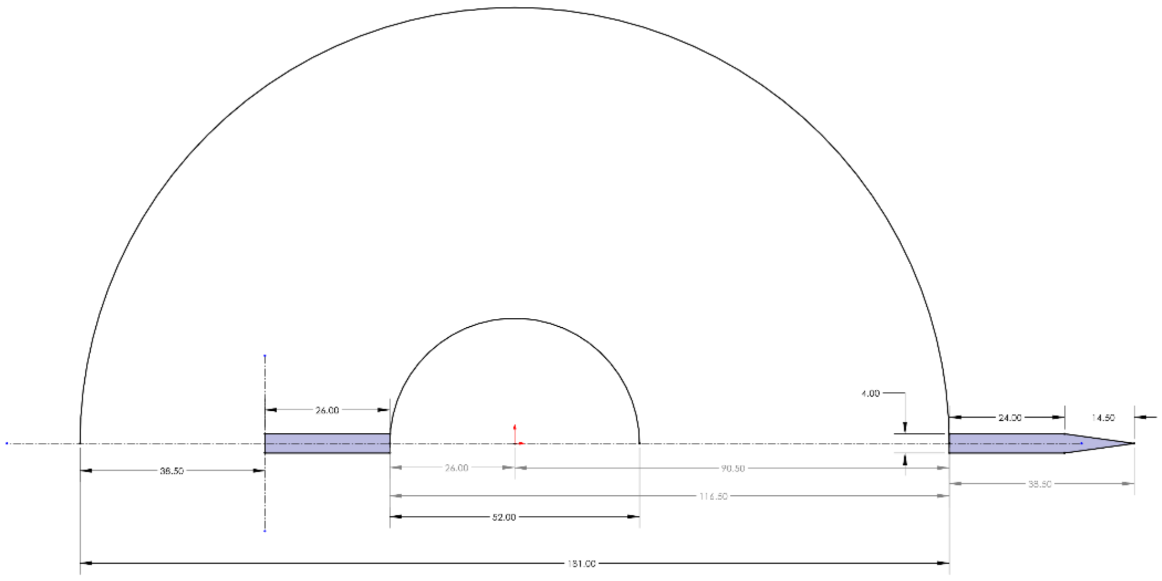


Figure 2-11: Diagram of the Drogue Shock Cord

In the Figure 2-11, the black numbers are the input numbers, while the gray numbers are dependent on the black numbers and represent the length of shock cord needed, as well as the distance from the end of either body tube to the parachute. Based on the model, the lower airframe requires 116.5 inches of shock cord between the body tubes. Additional shock cord was added to account for the knots connecting to the quick links, and for shock cord that is inside the lower airframe. Bowline knots were used to tie the shock cord to the quick links and 12 inches of shock cord were needed for each knot. There were 12 inches of shock cord in the lower airframe, which combined with the additional knot shock cord meant an additional 48 inches of shock cord was needed, for a total of 164.5 inches was needed.

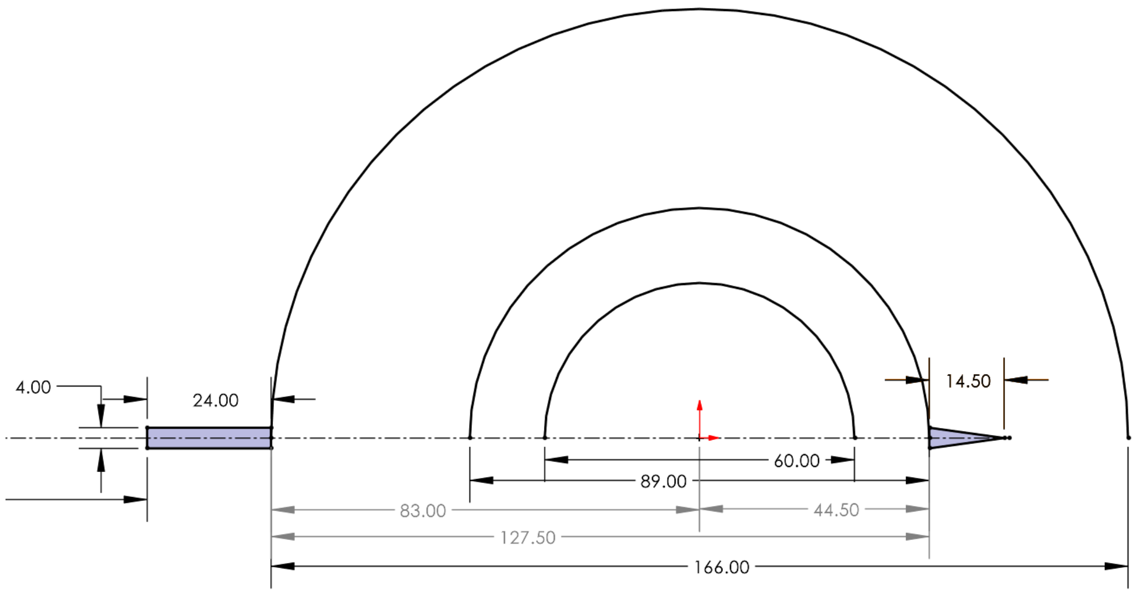


Figure 2-12: Diagram for the Upper Airframe with the Main Parachute

Figure 2-12 shows the upper airframe diagram which followed the same procedure as the lower shock cord diagram. However, since the main parachute is bigger than the drogue parachute, an additional 60 inches of shock cord was added to minimize the chance of the nose cone hitting the main parachute. With the additional shock cord for the main parachute, 36 inches of shock cord for the knots, and 20 inches of shock cord inside the upper airframe, the total shock cord needed for the upper airframe was 222 inches. The simplified lines diagram with just the lengths of shock cord needed can be seen in Figure 2-13.

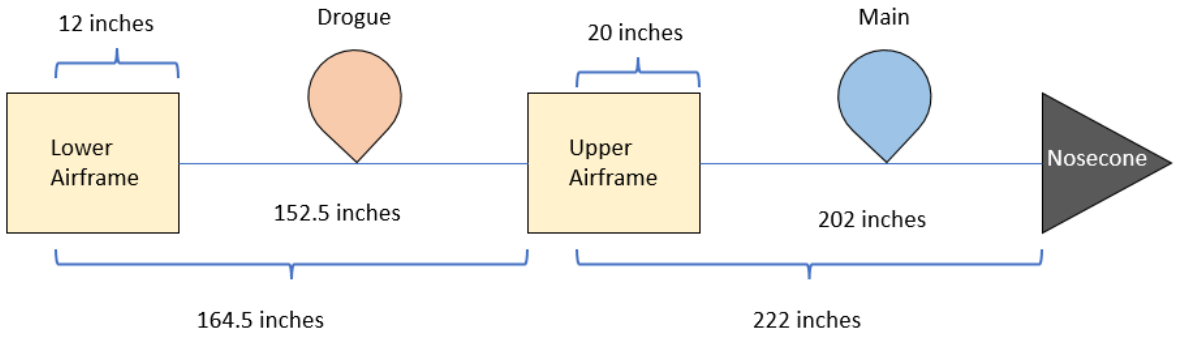


Figure 2-13: Line Diagram with Drogue and Main Parachute Deployed

2.1.5. Pre-Launch Procedure

Ejection Test Procedure

Prior to launch, the ejection system needed to be tested to verify the amount of black powder necessary to separate the airframe and deploy the parachute. The ejection test was completed before the rocket launch. The ejection test was as follows:

1) Load black powder into charge wells.
2) Place rocket in test position and clear the area.
3) Supply charge to e-match.

The required black powder was calculated before the ejection test and based on the internal volume of the rocket and the number shear pins connecting the body of the rocket. Based on the location of the ejection test, a test stand was used to ensure the test was safe and an accurate representation of the force needed to separate the rocket.

For an ejection test, the e-match tail was placed through the recovery bay and out of the switch hole. The tail of the e-match was then connected to a longer wire allowing the tester to stand back from the rocket. Once the test was ready to be performed, the ends of the wire were connected to a LiPo battery, completing the circuit and igniting the black powder.

Assembly Steps and To-Do List

The team decided to come up with an assembly list for both the recovery bay and the airframe. This list is meant to speed up the process of fabricating and assembling both sections. While only a simple bullet point list of steps, having them written out allows the team to have what we need to do and in what order with no confusion.

For the recovery bay, the steps were as follows:

1) Finalize design of recovery bay and wiring chart.
2) Have the bulkheads machined.
3) 3D print the recovery bay through one of the available spaces on campus.
4) Cut switch band ring from airframe material.
5) Surface prep the exterior of the recovery bay and interior of the switch band.
6) Epoxy the recovery bay and switch band together.
7) Make any necessary modifications to the bulkheads to be able to both attach to the recovery bay and fit inside the rocket.
8) Assemble the test article to check connections and operation. If the test article passes this inspection, it will become flight hardware. If the recovery bay fails inspection, a redesign and reprint of the bay will follow.
9) Drill switch band and attachment holes in the recovery bay.

For the lower airframe, the steps were as follows:

1) Buy wood and other required materials for assembly.

2) Laser cut wooden parts such as the base plates for the fin jig, and the fins themselves. This is done through one of the available laser cutters on campus.
3) Assemble the fin jig with the cut pieces and the standoff rods.
4) Once the test body tube arrives, practice epoxying the fins to the rocket and applying a fiberglass layup to the lower section of the body tube.
5) Compare actual layup mass to estimated mass and make any required changes to the airframe based off the new total.
6) Cut airframe to the required length and surface prep the fin can area.
7) Final fin epoxying and fiberglass layup on the lower body tube intended for flight.
8) Drill shear pin holes.

For the upper airframe, the steps were as follows:

1) Cut the upper and inner airframe to the required length.
2) Surface prep the inner airframe for centering ring attachment.
3) Laser-cut centering the three rings from ¼-inch plywood.
4) Epoxy the centering rings to the inner airframe with one ring at the very edge and the other two evenly spread out along the tube.
5) Dry fit the inner airframe and the upper airframe to ensure a tight fit, sand off any excess.
6) Dry fit the innovative separation system and nose cone with the airframes.
7) Surface prep the exterior of the inner tube and centering rings and the interior of the upper airframe.
8) Epoxy the inner airframe to the upper airframe.

9) Drill recovery bay attachment, shear pin, and innovative separation system holes.

Launch Sites

One of the major difficulties that the High Power Model Rocket (HPMR) team faced during the project was finding a launch site that would be open during the team's launch window. New England weather is unpredictable during this season, so no launch date was guaranteed to happen. Another issue was that most launch sites do not publish their planned yearly schedules before the new year starts, so it was difficult find a tentative date before January. In addition, not every launch site can support a high-powered rocket. To relieve the stress of finding a launch site early on, the team reached out to the Rhode Island Model Rocket Association (RiMRA). Ultimately, RiMRA's facility was not used to fly the rocket due to not having a launch date during desired times but served as a backup launch site. Luckily, CATO, a sport rocketry club located in Durham, CT, released their launch schedule following the new year. The HPMR team decided to fly the rocket February 18th at White's Field located in Durham, CT.

2.2. Results

2.2.1. Altimeter Testing

To verify that the selected altimeters will work with the rest of the flight hardware, they needed to be tested. The necessary tests are verifying the altimeter collects data, can be programmed, and can ignite an e-match.

Telemega

The Telemega was tested first, as it was the first altimeter available to test. The Telemega was able to be connected to a laptop via USB and output its sensor readings into the AltOS program provided by the manufacturer, Altus Metrum. The past flight data could also be collected from the program, which allows the data to be analyzed post-flight. The AltOS program also allows users to change the settings of the altimeter, such as apogee delay and main deployment height.

The final test was ensuring the Telemega can ignite an e-match, which can be done through AltOS. It took several attempts for us to ensure everything was set up correctly, but once the Telemega was properly set up, it was able to ignite an e-match. Since the flight battery was not available at the time of testing, a comparable battery was used to power the altimeter.

RRC3 Sport

When the RRC3 Sport was initially received, we did not have the USB connector so we could not properly test the altimeter. However, one test we could perform was checking if a 9-volt battery could ignite an e-match. 9-volt batteries typically have a lower current than LiPo batteries, which meant it was important to test if a 9-volt battery had enough current to ignite an e-match. Since the RRC3 could not be used for the test, an EasyMini altimeter was used, since it can use a 9-volt battery as well. Using a 9-volt battery, the EasyMini was able to ignite an e-match, meaning a 9-volt battery can be used to ignite the e-match.

Once we received the USB connector for the RRC3 Sport we were able to connect the altimeter to a laptop and view the sensor readings in the mDACs software, provided by the altimeter manufacturer. The readings were accurate to our environment, so we next used the mDACs software to simulate a flight. The simulated flight sent the altimeter data similar to what it would receive during an actual flight, and the altimeter responded based on how it was programed. When the altimeter reached its simulated apogee, the drogue e-match was successfully ignited, and when the simulated flight reached 500 feet, the programmed main deployment height, the main e-match was successfully ignited.

2.2.2. Airframe and Component Analysis

The ARS team developed simulations of several different possible scenarios that could occur and the impact they would have on the structure of the rocket. The primary focus was on fin force analysis. More specifically, what type of forces or loads would a fin experience in the worst-case scenario, that being the fin striking the ground at a 90-degree angle. This would apply the most force on the fin and likely see it snap or bend if the layup was not strong enough. The ARS team attempted to use Solidworks and ANSYS to determine the forces that would be imparted in this case.

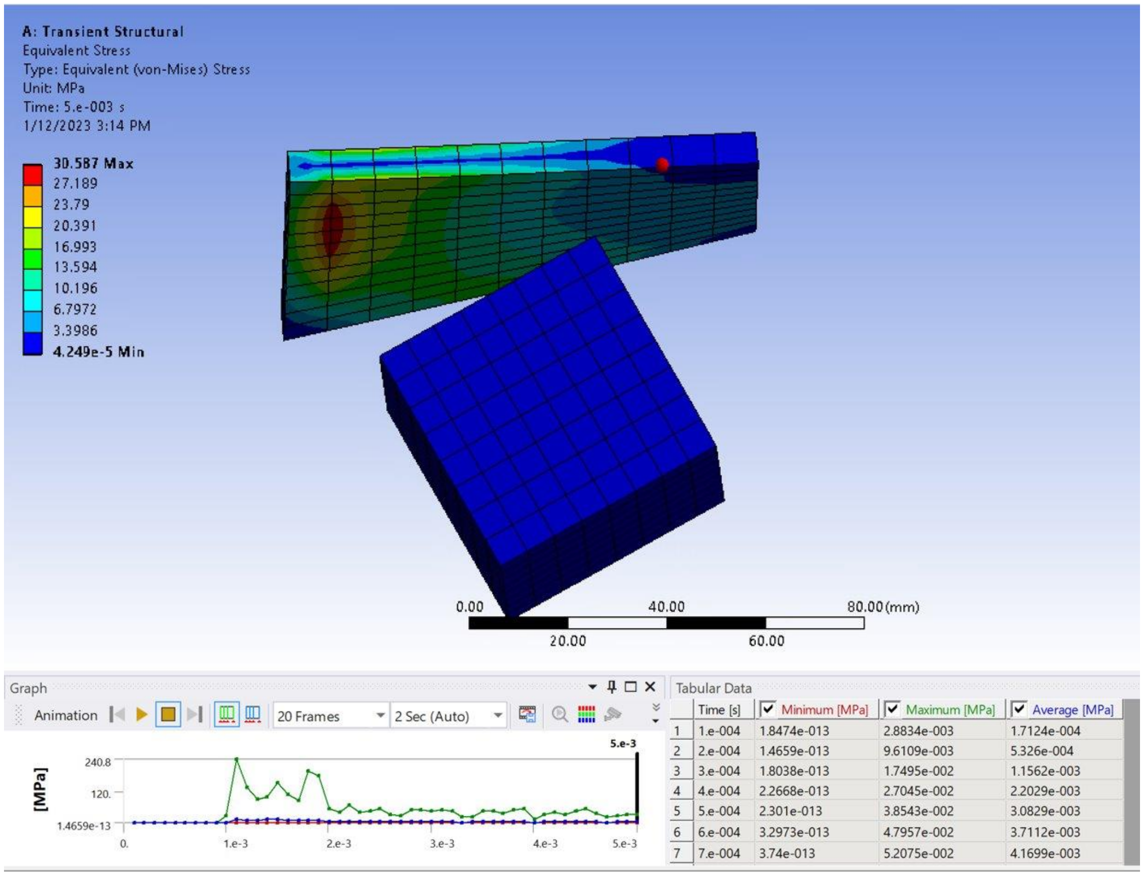
Solidworks Analysis

Solidworks was the first software used to analyze the fins and this is primarily because the group wanted to verify the ANSYS results once that simulation was complete. However, as with ANSYS, Solidworks had many issues with developing the simulation to the point where the team decided to focus solely on ANSYS.

The primary issue that was encountered using Solidworks was defining material components and force positions. The team had problems defining what the fins were made of, since we wanted to simulate the composite structure of the wooden fin and the fiberglass and epoxy layup. It was decided to simulate what would happen to the wooden fin, and even then, the team had to define a custom material for the wood since the material selection that was present did not have the properties of our laser cut fins. When the actual test was completed, the fins did not move with the expected pattern. According to Solidworks, the fins were bending by less than a millimeter, which did not make sense since a large force was imparted on the fin and pinned the side that is to be attached to the body tube. After attempting to troubleshoot, the team did not conclude as to why this was the output of the analysis and decided to scrap the verification test and instead focus solely on ANSYS as it would provide a more in-depth result. This would be done by having a striker hit the side of the fin to simulate the fin striking the ground, which was not an analysis that could be done in Solidworks as easily.

ANSYS Analysis

The Transient Structural model was used in the ANSYS simulation to simulate the fin strike more accurately. The fin geometry was uploaded to the project using the Solidworks model. The striker, for simplicities sake, was a 50 mm by 50 mm cube designed in ANSYS' Design Modeler. Within the Engineering Data tab, the material types for both structures were defined following the worst-case scenario. Concrete and oak wood were selected for the striker and fin, respectively. The two models were lined up within the project mechanics window to ensure the simulation ran as intended. The connections tab defines a rough contact between the striker and the fin. Standard Earth gravity was applied to the whole system under the transient tab. A fixed support was added to the fin where it would be attached to the lower body tube. Under the solution tab, the equivalent stress is determined to find the striker's impact on the fin.


A: Transient Structural
Equivalent Stress
Type: Equivalent (von-Mises) Stress
Unit: MPa
Time: 5.e-003 s
1/12/2023 3:14 PM

	Time [s]	✓ Minimum [MPa]	✓ Maximum [MPa]	✓ Average [MPa]
1	1.e-004	1.8474e-013	2.8834e-003	1.7124e-004
2	2.e-004	1.4659e-013	9.6109e-003	5.326e-004
3	3.e-004	1.8038e-013	1.7495e-002	1.1562e-003
4	4.e-004	2.2668e-013	2.7045e-002	2.2029e-003
5	5.e-004	2.301e-013	3.8543e-002	3.0829e-003
6	6.e-004	3.2973e-013	4.7957e-002	3.7112e-003
7	7.e-004	3.74e-013	5.2075e-002	4.1699e-003

Figure 2-14: Fin Strength ANSYS Simulation

In Figure 2-14, the red area is where the corner of the striker first hits and shows the most deformation. The program produces a stress of 30.587 MPa in the red area. The strength of birch plywood depends on the load's angle to the grain's face. The tensile strength of birch ranges from 21.5 MPa to 62.5 MPa, and the compressive strength ranges from 19.4 MPa to 26.5 MPa [10] Based on the simulation data, plywood-only fins would break in the worst-case scenario. This simulation shows that our decision to increase the strength of the fin can with a fiberglass layup is reasonable.

2.2.3. Ejection Test Results

When performing the ejection test of the upper airframe, the initial estimate of 1.2 grams of black powder was too low. The shear pins broke, separating the nose cone and the upper airframe, but the parachute was not ejected from the upper airframe. As such, for the flight the primary charge used was 1.5 grams of black powder, and the backup was 1.6 grams of black powder.

Since the lower airframe did not have atypical features like the inner tube of the upper airframe, the team was more confident with the black powder calculations performed beforehand. Additionally, the parachute was not as tightly packed in the lower airframe, so the parachute is able to be pulled out easier. As such, after seeing the successful separation of the nosecone and upper airframe, the team was confident the estimated 1.4 grams of black powder was a suitable primary ejection charge, with a backup charge of 1.8 grams. Lastly, the motor used for flight had a built-in ejection charge to fire after the motor burned, so there was an additional layer of redundancy in case the primary and backup black powder charges were not sufficient to separate the airframes and eject the parachute.

2.2.4. Launch Results

The launch of the rocket went as predicted, based on the previous analysis. The fiberglass layup of the fins survived the forces from launch and landing, with no signs of damage. The only component of the recovery bay that suffered damage was the zip tie used to hold the 9-volt battery supplying power to the RRC3 Sport altimeter. The zip tie sheared due to the forces produced by the rocket landing, since the RRC3 recorded data throughout the flight and only stopped recording after the accelerometer indicated landing forces and the barometer indicated ground level. The 9-volt battery was loose in the recovery bay when the rocket was retrieved upon landing. The 9-volt battery was likely the only one to shear a zip tie as it is the heaviest battery in the recovery bay.

Additionally, a nomex blanket, used to protect the parachutes from the black powder charges, was burned to the point of detaching from the rocket as seen in Figure 2-15. The corner of the Nomex blanket, with the hole to attach the quick link, was burned through by the ejection charge.

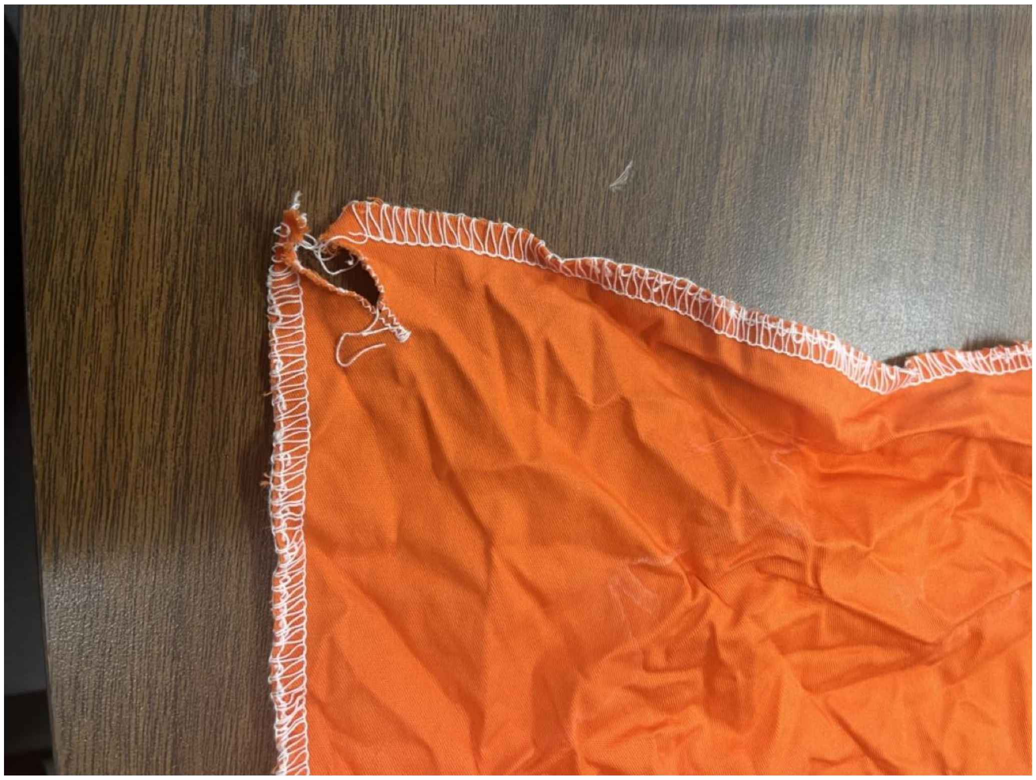


Figure 2-15: Burnt Nomex Blanket

This corner was the narrowest part of the Nomex, that also experienced direct forces by being connected to the quick link, so it was the most likely part of the blanket to be torn. Lastly, the Nomex blanket was borrowed from the model rocketry club, so it had been previously flown and subjected to multiple ejection charges.

3. Flight Dynamics and Analysis

3.1. Overview

The Flight Dynamics Analysis (FDA) team was tasked with completing various studies and evaluations of the vehicle's flight performance. The investigation into these flight characteristics of the launch vehicle and the systems to analyze them are important aspects of constructing and flying a model rocket. The primary contribution to the team from the FDA team was in the form of an onboard custom flight computer running custom flight modeling software. Additionally, the team also analyzed the drag of the nose cone, the fin-flutter, the aerodynamic performance of the fins, and the coupling of the roll and pitch motions.

3.2. Methodology

3.2.1. OpenRocket Model and Predictions

For the majority of the flight performance and prediction, the team used a trusted open-source rocket simulation software called OpenRocket. By importing known environmental aspects such as the altitude of the launch site above sea level, the geolocation of the launch site, the expected average windspeed, and various other factors, the software exports a simulated trajectory of the vehicle. This software has been used for many years by the high-power rocketry community and by other rocketry organizations at WPI. Below, in Figure 3-1, the predicted flight of the rocket for a 7mph wind is depicted. In the simulation, the rocket has a predicted apogee of 418 meters (1371.39 ft) and a total flight time of 45.7 seconds.

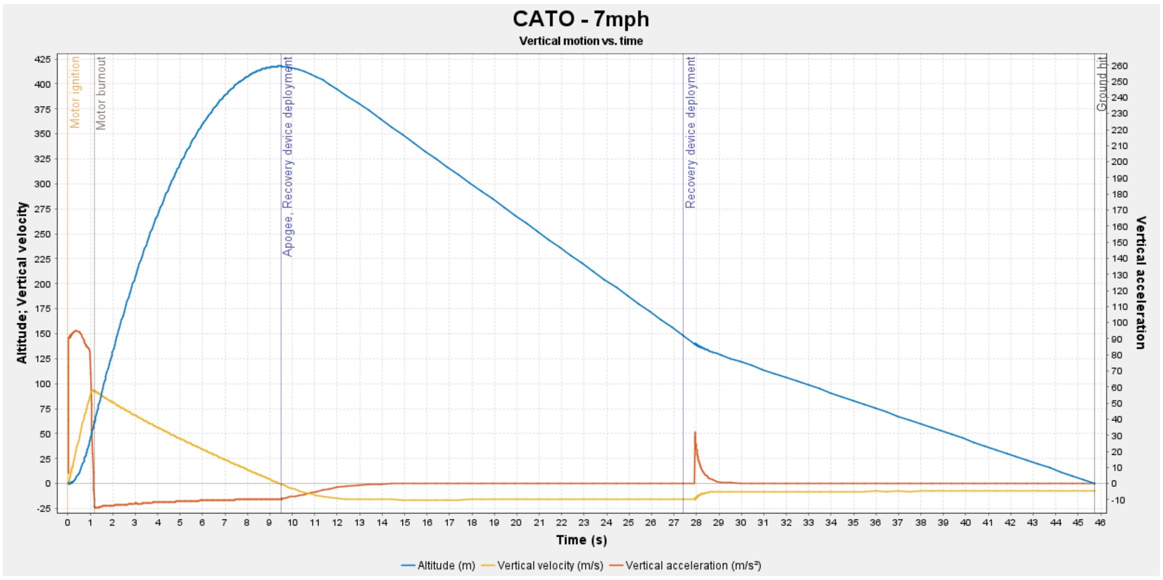
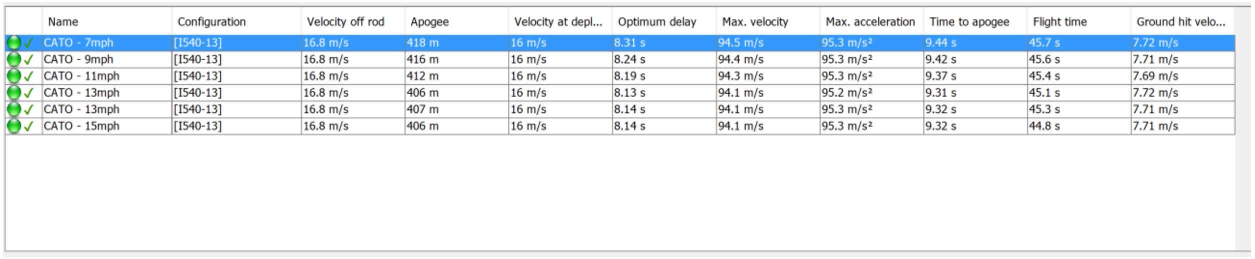


Figure 3-1: OpenRocket Flight Prediction

The rocket is predicted to hit apogee 9.44 seconds into the flight. The simulation was run for multiple different wind speeds to determine the range of possible flight profiles that the rocket could encounter. These are listed below in Figure 3-2. These simulations show how wind speed has very little effect on the apogee of the rocket.



	Name	Configuration	Velocity off rod	Apogee	Velocity at depl...	Optimum delay	Max. velocity	Max. acceleration	Time to apogee	Flight time	Ground hit velo...
● ✓	CATO - 7mph	[I540-13]	16.8 m/s	418 m	16 m/s	8.31 s	94.5 m/s	95.3 m/s²	9.44 s	45.7 s	7.72 m/s
● ✓	CATO - 9mph	[I540-13]	16.8 m/s	416 m	16 m/s	8.24 s	94.4 m/s	95.3 m/s²	9.42 s	45.6 s	7.71 m/s
● ✓	CATO - 11mph	[I540-13]	16.8 m/s	412 m	16 m/s	8.19 s	94.3 m/s	95.3 m/s²	9.37 s	45.4 s	7.69 m/s
● ✓	CATO - 13mph	[I540-13]	16.8 m/s	406 m	16 m/s	8.13 s	94.1 m/s	95.2 m/s²	9.31 s	45.1 s	7.72 m/s
● ✓	CATO - 13mph	[I540-13]	16.8 m/s	407 m	16 m/s	8.14 s	94.1 m/s	95.3 m/s²	9.32 s	45.3 s	7.71 m/s
● ✓	CATO - 15mph	[I540-13]	16.8 m/s	406 m	16 m/s	8.14 s	94.1 m/s	95.3 m/s²	9.32 s	44.8 s	7.71 m/s

Figure 3-2: OpenRocket Simulations

3.2.2. Flight Computer

To reliably control the custom separation system and all other electro-mechanical systems integrated into the launch vehicle, its flight computer needs to be capable of sensing and predicting the vehicle’s motion through space. The computer must also be capable of outputting pulse width modulation (PWM) signals and firing pyrotechnic charges. There are few commercial altimeters capable of triggering pyro charges as well as outputting PWM signals to control servos. The TeleMega, the main COTS altimeter with these capabilities, has a large footprint and costs hundreds of dollars while only running pre-compiled, non-editable flight software. Additionally, although the manufacturer lists the TeleMega as being capable of driving PWM servos, that capability has never been implemented into their proprietary firmware which makes it unsuitable for actuating the custom separation system. To overcome the shortfalls of the TeleMega, the team decided to design and fabricate a custom flight computer according to the project's requirements.

Microcontroller Selection

To achieve the required flexibility and capabilities, the custom flight computer was based on an embedded microcontroller, a suite of digital sensors, and a data-logging telemetry system. SparkFun's line of MicroMod products contains a wide range of embedded processors, which are designed to fit into an M.2 connector while still providing the computational horsepower needed for predictive modeling and apogee detection. This COTS microcontroller solution greatly reduces the design complexity, eliminating the need to design complex supporting circuitry required for microcontrollers while adding flexibility to the design. The initial microcontroller chosen was the STM32F405 housed on a MicroMod processor board. The STM32 is the industry standard for rocketry altimeters and UAV flight controllers however, upon testing the processor board, it was discovered that several core pieces of functionality were missing, such as lacking compatibility with some of the modules on the flight computer. Additionally, the STM32 lacked the necessary IO (both PWM and ADC pins) needed for the design.

Figure 3-3: SparkFun MicroMod STM32F405 Processor Board 10]

Due to the lack of features on the STM32 processor board, a replacement was selected: the Teensy processor board. The Teensy is another processor board available in SparkFun's MicroMod line and has more power, as well as all the pins and functionality required for the flight computer. Additionally, the Teensy is easily programmed using the Arduino Integrated Development Environment (IDE), greatly reducing the barrier to entry when switching to a higher-power processor. As the flight computer was designed to use a MicroMod processor board, this switch was as simple as ordering the different processor and updating the pin mapping in software.

Figure 3-4: SparkFun MicroMod Teensy Processor Board [12]

Sensor Selection

Five core sensors were chosen for the flight computer, all to feed data back to the microcontroller for calculations of the rocket's state and trajectory: accelerometer, rate gyros, magnetometer, barometer, and GPS. The accelerometer, rate gyros, and magnetometer will serve as the computer's inertial navigation system, while the barometer directly measures pressure which can be used to calculate altitude. The GPS is a supplemental sensor that can be used in the update step of a Kalman Filter for added precision or for use in recovery efforts to locate the rocket.

For the inertial sensors, a 9-degree-of-freedom (DOF) inertial measurement unit (IMU) is ideal, as it integrates 3D accelerometers, rate gyros, and magnetometers in one package, making electrical and software implementation simpler. However, 9DOF IMUs have become hard to source due to lack of production, low stock, and prohibitively high costs. 6DOF IMUs, which omit the magnetometers, are more widely available and still provide the benefit of significant ease of integration. The TDK ICM-42688-P was selected as the 6DOF IMU for the flight controller as it has low noise characteristics, high precision, and a relatively low cost. However, it only has an acceleration range of ±16 g, which could easily be saturated during deployment events—crucial times to log data. Because of this limitation, a secondary, high-g accelerometer is needed. This accelerometer would be able to log data throughout the entire flight at a lower resolution than the precision (low-g) accelerometers can. The high-g accelerometer chosen for this flight computer is the ADXL375, which has a range of ±200 g and is widely used in commercial rocket altimeters for inertial sensing.

Since neither the ICM-42688-P nor the ADXL375 have built-in magnetometers, a MMC5983MA 3-axis magnetometer was added to the board for directional sensing. The

MMC5983MA was chosen due to its wide availability and ease of integration with Arduino and similar microcontrollers.

A barometer was included to supplement the inertial sensors and serve as the computer's primary altitude sensing device. The MS5611 barometer included on the flight computer is in the same family of sensors used in most commercial rocket altimeters. It has a measurement range of 10 – 1200 mbar, which is well beyond the flight limits of this vehicle and most high-power rockets.

Finally, a u-blox SAM-M10Q GPS module was selected to be the GPS sensor for the flight computer. The SAM-M10Q has industry-standard accuracy for units of this size and power and has the added benefit of having an integrated GPS patch antenna. This integrated antenna significantly reduces the RF component of the schematic and board design, as the GPS unit can be treated as self-contained, meaning only external factors need to be considered in the RF design.

Telemetry and Data-Logging Component Selection

To transmit telemetry from the flight computer to a ground-based receiver computer, a radio transmitter module was required. Due to its low cost, ease of implementation, and widespread availability, the LoRa protocol was the obvious choice over other telemetry protocols. The RFM95W from HopeRF was chosen as the flight computer's LoRa transceiver module as it was the smallest all-in-one unit commonly available and did not require any supporting RF circuitry. The RFM95W did, however, need a supporting connector to attach an antenna for transmitting. Typical breakout boards for the RFM95W from Adafruit and SparkFun contain either U.FL or SMA antenna connectors. U.FL was ruled out for this design as it is prone to disconnection under vibrations and extreme loads, making it not suitable for rocketry applications. SMA, on the other hand, is extremely vibration robust as it is a metal screw-locking connecter and must be tightened down to achieve proper contact. Despite its superior mechanical performance, SMA was also not selected for the antenna connector due to its large size and high weight, being a large metal connector. Deviating from the typical LoRa antenna connectors, Micro-miniature coaxial (MCCX) was implemented as it is small and lightweight, compared to the SMA connectors and has a positive lock-snap mechanism to ensure vibration resistant. Although MMCX is not commonly seen on LoRa systems, it is extremely popular among UAV electronics and is known to be versatile, easy to use, and vibration resistant.

Although the flight computer was designed to continuously transmit flight telemetry, an onboard data-logging system was also included to provide a backup data storage approach if the transmitter failed, or if a flight was conducted without transmitting enabled. The first component selected for the data-logging system was a W25 flash chip from Winbond. This is a standard SPI flash package found on many microcontrollers, as well as other COTS rocketry altimeters. Although the W25 flash is extremely fast and has the capacity to store all the data required, a microSD card slot was also included on the flight controller as it is easier to read by the end user since it can be simply plugged into a computer without the need for specialized data-logging software.

Test Boards

Before final design of the flight computer was completed, two test boards were created to evaluate the performance of the selected sensors and supporting components. The first board, the telemetry board, contained the GPS and LoRa modules as well as supporting passive components and indicator LEDs. The telemetry board also included the MMCX connector for attaching a suitable antenna. The second test board, the sensor board, contained the barometer, accelerometers, and magnetometer sensors as well as their required passive components. These boards were designed in EAGLE using the same methods as the final flight computer design and were fabricated by OSH Park. The boards functioned as expected, allowing for the full flight computer design to continue. Only one issue was found in the manufacturing of the boards, where the profile cutout for the MMCX was not present and had to be manually cut before assembly. The test boards connected to the microcontroller for testing can be seen below, in Figure 3-5.

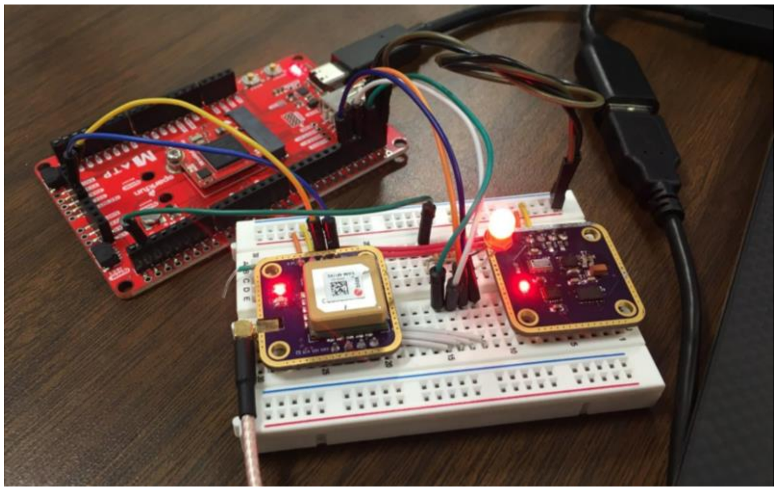


Figure 3-5: Telemetry (left) and Sensor (right) Test Boards

Schematic Capture

Both schematic capture and board layout for the flight computer was done in the EDA tool Autodesk EAGLE. Schematic capture started with the main centerpiece of the computer: the SparkFun MicroMod M.2 connector. Based on this component, electrical nets were created and named corresponding to their functions (signal, power, etc.) as shown in Figure 3-6. All schematic drawings can be found in Appendix D and were created in accordance with the manufacturers' recommendations as well as previous designs, such as breakout boards.

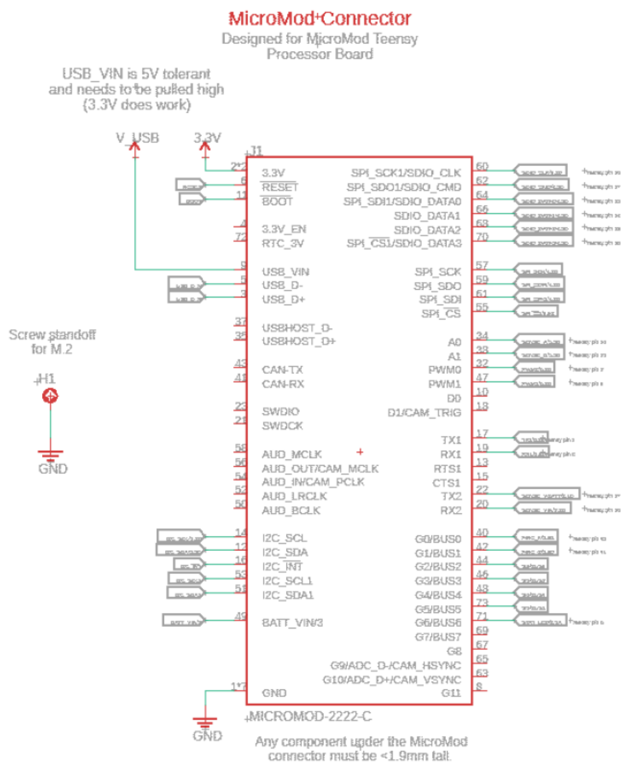


Figure 3-6: Schematic Segment for MicroMod Connector

Board Layout

With the schematic capture for the flight computer completed, the PCB layout was conducted in EAGLE, linking seamlessly with the schematics. Initially, components were separated out into groups, or “blocks”, corresponding to their function and location in the schematic document. These blocks were then placed in a logical arrangement in 2D space, with components such as sensors being grouped near each other to ease signal routing. Certain larger components, such as the MicroMod connector and RFM95W LoRa module were among the first to be placed on the board due to their size dictating the overall layout of the board. Once initial placements of components were completed, the board outline could be created. As the components were packed tightly with overall board dimensions in mind, the outline was reasonably small for the functionality of the flight computer, at 1.2” x 2.6”. Four #4-40 mounting holes were added to the board with a spacing of 1.0” x 2.4”, as shown below in Figure 3-7.

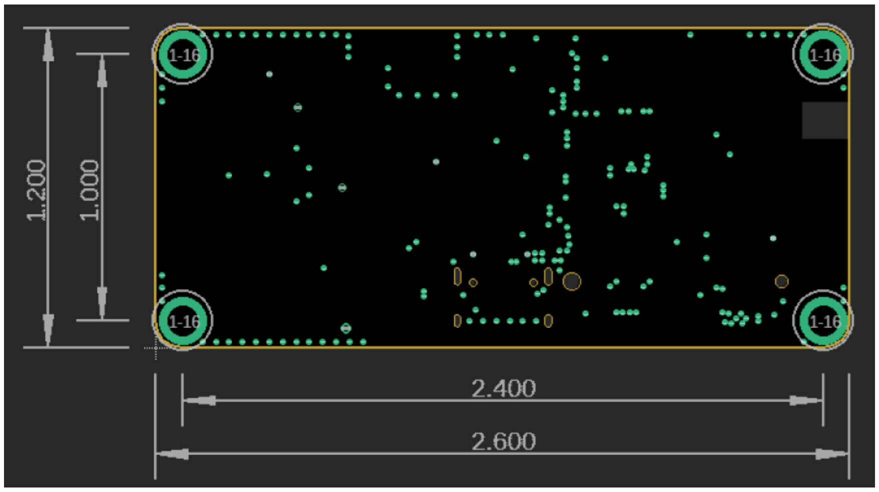


Figure 3-7: Flight Computer Physical Dimensions and Outline

Using the placement of major components in their blocks, traces were routed to create electrical connections corresponding to the power and signal nets created in the schematic. For certain traces, such as current-carrying power traces and the trace for the antenna connector, the width of the trace was tuned to ensure proper functionality. Otherwise, a default trace width of 8 mil (0.008”) was used as it was well within the manufacturer’s capabilities and allowed a tight packing of signals on the board. As components were placed on both the front and the back sides of the board, vias were necessary to electrically connect signals between planes. Although they add resistance and are larger than traces, vias cannot be avoided on complex circuit boards. The final board layout can be seen in Figure 3-8, with red areas indicating copper traces or pads on the top layer, and blue indicating the bottom.

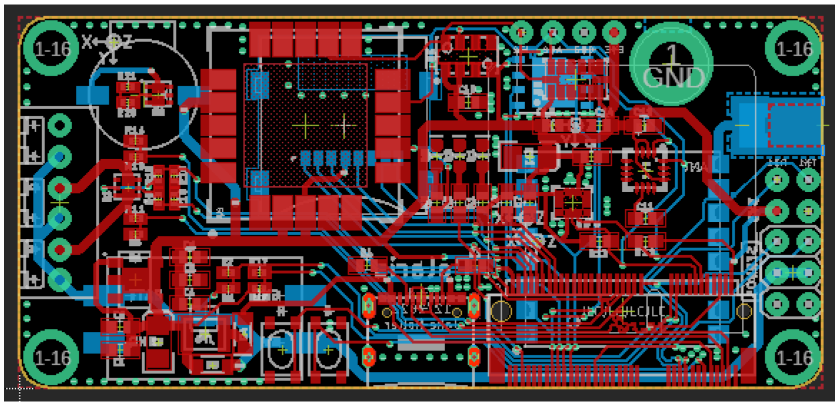


Figure 3-8: Flight Computer Board Layout with Traces

After careful review of the flight computer’s schematics and PCB layout, the boards were sent out to be manufactured. As the standard for most circuit boards, 1.6mm thickness was chosen due to its low cost and lead times compared to thinner boards that may save weight and the overall height of the board. Although slightly more expensive, Electroless Nickel Immersion Gold (ENIG) surface treatment was chosen for the board plating to improve solderability for the many fine-pitched components. ENIG was also called out in the datasheet for the two accelerometers as recommended due to their specific package footprint.

3.2.3. Flight Software

The flight software that runs on the custom electronics package was written in MathWorks MATLAB software. This developmental choice was a result of being familiar with the MATLAB software and its ability to easily troubleshoot mathematical systems. MATLAB was used for the development and testing of the system code. The final versions of the MATLAB scripts were translated into C++ for use on the custom electronics.

Overall Structure

The overall code structure consists of two major sections. The first section, the initialization section, is where initial values and variables are defined. Many of these values are constants used in calculations or values measured prior such as system noise. The second and most major section of code is the continuously running while-loop. After initial setup, this loop runs until the electronics run out of power, the rocket is destroyed, or the system as successfully detected a landing on the ground. To electrically model the system there is a logic system of *if()* and *ifelse()* statements that allow the loop to model the rocket in six distinct stages. Each time the loop is run, a Kalman Filter (KF) is run to estimate the state of the vehicle. Depending on the state of the system, the values from this KF will be stored and used to determine if the vehicle needs to transition from one stage of the flight to another. The sections below describe each stage of the flight software and what must happen for it to transition from that stage into the next stage of the flight. Additionally, there are additional logic statements that allow the code to transition between our precision and our high-G accelerometer. By inputting the maximum value of our precision accelerometer, the code can be set to automatically switch between the two units so that acceleration is always measured.

On-Pad Armed

This is the first stage of the rocket and the default that it enters the moment the system is supplied power. The rocket assumes that it is aimed vertically on the launch rails and has been armed for launch. This stage does not use the KF to estimate its state as we want raw accelerometer data to tell us the moment that we detect launch. Any delay in this detection can damage the system's accuracy in the future. To exit this stage, the system must wait a minimum time so that operators arming the rocket have an opportunity to walk away and then the accelerometers must detect a value greater than the minimum required for launch.

Boost Phase

The boost phase is a short-duration high-G environment. During this phase, the system is expected to briefly use the high-G accelerometer before switching back to the precision accelerometer. To exit this phase of flight, the system must both hit a minimum time delay set by the expected motor time to burnout and must experience a negative acceleration indicating that the rocket is no longer experiencing a thrust force from the motor.

Ascent Coast

This section describes the unpowered and upward flight of the rocket. In this section, the rocket begins its detection of apogee to determine when to deploy the drogue parachute. The apogee detection system must be robust enough to both detect apogee but not fire any charges without certainty that it is in fact at apogee.

During the ascent coast, the software must complete apogee detection. The purpose of apogee detection is to sense when the rocket hits the highest point of its flight so that a computer can trigger a separation event. Apogee is the most ideal moment for the flight computer or altimeter to trigger a separation mechanism to deploy the drogue parachute due to the low flight loads on the vehicle. The apogee can be described as the maximum value of an altitude over time plot. The difficulty in apogee detection is because the system doesn’t know what the maximum altitude is until after the rocket hits the apogee. Therefore, to trigger an ejection charge at apogee, flight software can be written to either fire immediately after apogee or at apogee based on a real-time prediction of the rocket’s flight. The paragraphs below will cover some of the logic checks implemented into apogee detection software.

Logic and safety checks are critical parts to ensure the redundancy of the flight software. It is important to consider edge cases that could cause the system to trigger inadvertently. For this, a series of simple logic checks must be met before any apogee detection algorithm can run. Additional logic checks not mentioned here can be viewed in the flight code itself seen in Appendix E.

One event that would be catastrophic for the rocket would be for the system to trigger an ejection charge during the vertical ascent of the vehicle. This mistake could happen if there is a transient pressure spike, which could occur in cases such as transonic behavior. After taking the slope between two measured altitude values, the system has effectively found the instantaneous velocity of the rocket. If the instantaneous velocity is greater than 0.5 ft/second, then the system will not be able to trigger the descent counter.

Another way to ensure that the system does not trigger early is to require a minimum altitude for deployment. For this, we chose an arbitrary value of 100 ft as the minimum deployment altitude. This requirement is completed by another if-statement that stops the loop from triggering if the altitude is below 100 ft.

After ensuring that our code can account for various unexpected scenarios, we need to make sure that we can detect the apogee reliably. Using a KF to create a combined barometer and accelerometer measurement, we can extract the measured altitude from the change in pressure and acceleration registered on the unit. Kalman Filters combine a model of a dynamic system with noise measurements to produce an optimal estimate for the position of the vehicle. The system and measurement models are equations (3-1) and (3-2) below, respectively. The system's state at step k is predicted by multiplying the Φ_k matrix with the state matrix at step k-1 then adding in noise from the system model. Equation (3-2) then changes the system into a measurement.

$$x_k = \Phi_{k-1} \cdot x_{k-1} + w_{k-1} \tag{3-1}$$

$$z_k = H_k \cdot x_k + v_k \tag{3-2}$$

The matrices used in the dynamic model are as follows:

$$x_k = \begin{bmatrix} s_k \\ v_k \\ a_k \end{bmatrix}, \Phi_k = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & t \\ 0 & 0 & 1 \end{bmatrix}, H_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-3}$$

The state estimate extrapolation and state estimate update formulas are (3-4) and (3-5) below, respectively. The state estimate extrapolation (3-4) predicts the current value of the state using the previous and state transition matrix. Then the state estimate is updated in equation (3-5).

$$\overline{x}_{k(-)} = \Phi_{k-1} \cdot \overline{x}_{k-1(+)} \tag{3-4}$$

$$\overline{x}_{k(+)} = \overline{x}_{k-1(-)} + K_k\left[z_k - H_k \cdot \overline{x}_{k(-)}\right] \tag{3-5}$$

The error covariance extrapolation (3-6) and update equations (3-7) are below; They estimate the noise statistics using standard deviation matrix Q.

$$P_{k(-)} = \Phi_{k-1(-)} \cdot P_{k-1(+)} \cdot \Phi_{k-1}^T + Q_{k-1} \tag{3-6}$$

$$P_{k(+)} = [I - K_k \cdot H_k] \cdot P_{k(-)} \tag{3-7}$$

The matrix Q is defined below:

$$Q_k = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \sigma_m^2 \end{bmatrix} \tag{3-8}$$

Lastly, the Kalman gain matrix is equation (3-9) below.

$$K_k = P_{k(-)} H_k^T \left[H_k P_{k(-)} H_k^T + R_k\right]^{-1} \tag{3-9}$$

The matrix R is defined below:

$$R_k = \begin{bmatrix} \sigma_s^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \sigma_a^2 \end{bmatrix} \tag{3-10}$$

From this, the system can calculate the rate of change of the rocket's altitude. If the slope of the height is negative for a set number of timestamp readings, then the system has registered apogee and will trigger any events that occur at apogee. The first iteration of an apogee detection system utilized only barometer measurements. The introduction of the KF filter as described above allows for a better prediction of the altitude of the rocket with respect to time. The code can be seen in Appendix E.

State 2 in the code shows that when the slope, *dh*, is negative, the algorithm increases a counter variable by one instance. It also records the time stamp of the first moment it detects this descent instance. Each time the loop runs, the lower if-statement checks if the counter has been incremented more than five times. The logic behind this is that if the altitude's slope is negative for a significant number of instances, then the rocket is descending and has thus passed apogee. This allows the system to enter the for-loop and trigger any apogee events.

To test the algorithm, numerous data references were used. The system was primarily tested using an export of altitude vs. time CSV file produced by our OpenRocket simulation model. A smoothing system was also developed to incorporate the data expected from real flight data. For this, we ran the program on actual flight data provided by the university's High Power Rocketry Club. The data from these flights showed the combined barometer-accelerometer reading has a slower update rate than the microprocessor's clock frequency. To account for this frequency difference, a logic system was created that only tests the slope of the system when the KF provides a new data point update that differs from the prior measured data. This system accounts for the step-function resemblance of the sensor outputs.

To exit this stage of the flight, the system must meet the apogee detection parameters discussed above and thus trigger the apogee events.

Drogue Descent

For this section of the flight, we can assume that the drogue parachute has successfully been deployed and that the rocket is now descending under some velocity less than a raw ballistic descent. For this, we still need to detect the altitude to deploy the main parachute. The deployment altitude for the main parachute is determined and set before the rocket is launched. This works by simply reading the altitude predictions from the KF and looking to see if that altitude has been reached. Once this altitude is hit, the system triggers the deployment of the main parachute and enters the next stage.

Searching for Ground

In this stage, the rocket is descending slowly and is looking for its impact on the ground. For this, we want the system to transition out of this state once it has hit the ground. To determine if it has hit the ground, we are looking to see if it has detected no change in altitude for multiple occurrences. This allows the system to ignore any updrafts that might cause the rocket to go up or down in altitude while descending. If the rocket registers that it has been stationary for a determined interval, then it transitions into the final stage of the flight.

Store and Shutdown

In this final stage, we have determined that the rocket has landed safely on the ground. Now we are intent on storing and securing all the rocket components. This would involve firing any remaining energetic charges, mechanisms, or components that might cause damage to a person of the ground crew recovering the rocket. After these stages are done, the flight computer stores all the data to the SD card and stops the collection of new data so that no data is overwritten. This marks the end of the flight profile.

Final Logic Checks

During each stage of the flight, there are various timers and timeout conditions that are set prior to launch by the operator. All these timeouts rely on the OpenRocket predictions of how long each stage of the flight should last. For example, if the rocket detects a boost phase that is significantly longer than the boost phase that is rated for the motor, the system will automatically transition to the next stage of the flight by assuming that something went wrong in the intended state transition logic.

3.2.4. Nose Cone Drag Analysis

The drag force on a vehicle is the aerodynamic force of flight that acts in the opposite direction of the vehicle’s motion. This aerodynamic force is due to the interaction of a solid body with a fluid. In the case of a HPMR, drag is one of the main forces that the separation system needs to overcome in order to separate the nose cone from the airframe. Without this separation, the parachute cannot deploy. The nose cone of the HPMR was treated as the control volume of the analysis, and forces on the rest of the rocket body were not included.

The equation for drag force is modeled below:

$$D = C_d \cdot \rho \cdot \frac{V^2}{2} \cdot A \qquad (3\text{-}11)$$

To calculate the drag on the nose cone during separation, the velocity of the rocket during descent and the air density at sea level were used. For this initial analysis, the known drag coefficient of an ogive nose cone was used since that is the shape of the HPMR's nose cone.

The maximum drag force the separation system will need to overcome to separate the nose cone from the airframe was calculated to be:

$$D \approx 0.00403\ Newtons$$

When researching the validity of this result, it was discovered that the drag on an ogive nose cone at a descent speed would be minimal. This comparison is shown in Figure 3-9, where different nose cone shapes at standard temperature and pressure were simulated at various velocities.

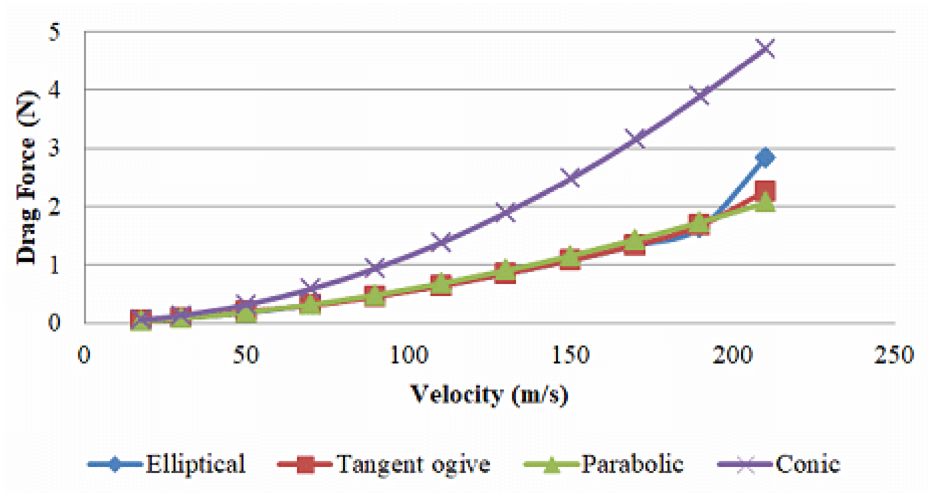


Figure 3-9: Analysis of Drag Force for Different Nose Cone Design [13]

3.3. Results

3.3.1. Kalman Filter Testing

From running the flight software on MATLAB with imported flight data the following results were produced and can be seen in Figure 3-10 and Figure 3-11.

```
fire_charge = 1
apogee_value_detected_at = 296.3489
altitude_charge_fired_at = 295.1621
time_of_apogee_start = 8.5400
time_of_fire = 8.6000
n = 3
reached apogee
deploy main
time_of_main_deploy = 11.0800
fire_main_charge = 1
n = 4
ground_counter = 1
ground_counter = 2
ground_counter = 3
ground_counter = 4
ground_counter = 5
ground detected
time_of_ground_impact = 13.0700
altitude_of_ground_detect = -55.4424
n = 5
data stored
i = 1492
```

Figure 3-10: Simulation Printout

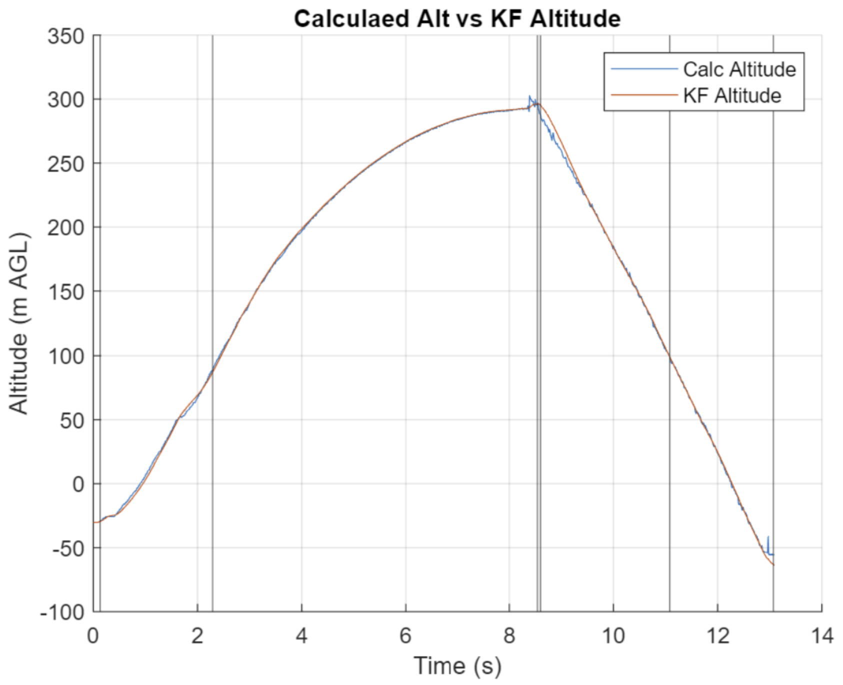


Figure 3-11: Simulated Flight Path w/ State Bars

From the figures above we can see that the system tracks the flight of the rocket for most of the trajectory. It can be seen that the system is throwing some negative altitude values when it is near the ground which means that something is off in the way we are measuring or calculating the flight for those sections. Addressing sensor bias can address this issue. Despite the offset altitude, the system tracks that altitude well and can recognize the transitions in the flight phase adequately. The two lines plotting altitude track each other very closely. The plot of calculated altitude state vs the KF altitude state is expected to track less closely as the model bias and sensor noise of the system is more accurately modeled into the system. Overall, these printouts confirmed that the flight software is performing as designed and correctly modeling the state of the rocket over the flight path. Using prior flight data enabled the tuning of the standard deviations that were plugged into the Kalman filter.

Quadcopter Flights

In addition to running the code on simulated and past rocketry flights, the team also tried to test the altimeter by attaching it to a remote-controlled quadcopter. The intention of these tests was to achieve a parabolic flight trajectory to determine if the altimeter is able to detect liftoff and apogee, which are the most important parts of the flight for the computer to detect properly. The tests consisted of attaching the flight computer and battery system to a racing quadcopter and then having the pilot complete a series of desired flight profiles. The general flight profile consisted of high vertical acceleration and then a slowly decreasing vertical thrust until the quadcopter was only experiencing the effects of drag and gravity. After the quadcopter passed the apogee of its flight, the pilot would throttle up and safely land the vehicle. Each flight varied in the level and duration of acceleration as the pilot was not able to perfectly replicate the same flight repeatedly. The first round of quadcopter testing demonstrated how sensitive the sensors were to vibration so the need for a hard mounting structure was determined. The addition of a 3D-printed mounting plate to the vehicle allowed the computer to record cleaner data with lower magnitudes of noise. This second round of flights also allowed the pilot to tune the flight profile so that the hardware experienced an upward coast before the apogee which is more similar to the flight of a rocket.

3.3.2. Flight Computer Assembly and Electrical Testing

After the boards for the flight computer were received from the manufacturer, they were inspected and tested for continuity. After these initial tests, the boards were assembled, as seen in Figure 3-12, using solder paste and components, using the designed layout as a reference when placing components. Upon completion of assembly, the circuit board was again tested for continuity and no unexpected short-circuits were found. The board was then initially powered on through its USB-C port which resulted in no power LED lighting, and no voltage present on the 3.3V pins. After further inspection of the board, it was found that the power indicator LED had been placed 180 degrees to how it was designed, causing the LED to block the current path from 3.3V to ground, and eventually blow out. This most likely triggered the power circuitry's over-current protection feature which shut the board down, protecting the other components and yielding 0V on the 3.3V line. Once the indicator LED was replaced, the board was inspected once more and then powered on successfully. Then, the MicroMod Teensy processor board was slotted into the M.2 connector and fastened into place using its standoff. Once plugged into a computer running the Arduino IDE, the microcontroller was successfully able to be programmed to test all the sensors and peripherals, which functioned as expected.

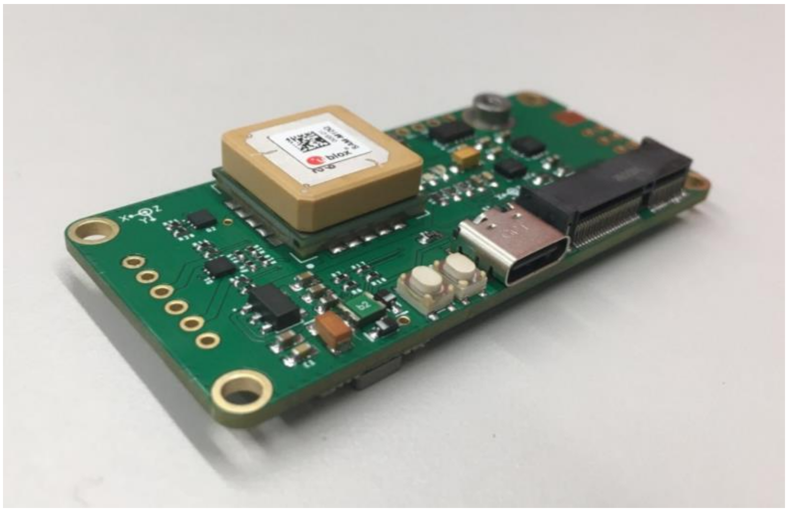


Figure 3-12: Assembled Flight Computer Circuit Board

After the successful sensor tests, the pyro charge MOSFETs were tested and functioned as expected, however the MOSFET controlling the buzzer did not function. Although the buzzer-MOSFET combination worked in testing off the board, inspection of the board's layout and schematics revealed that the PWM signal for the buzzer was not connected properly to the M.2 connector for the microcontroller, and thus would not function. As a quick fix, a 30 AWG wire was soldered between a PWM pin on the microcontroller and the gate circuitry of the buzzer MOSFET to restore functionality.

3.3.3. Analysis of Aerodynamic Loads

This section covers the analysis of aerodynamic loads on the rocket and the resulting stability.

Fin Flutter

Fin flutter is an important aerodynamic phenomenon to consider when designing a high-powered model rocket. Flutter is an aeroelastic instability that occurs when the forces acting on the fins of the rocket cause them to deform [14]. This deformation then results in an oscillatory motion that can become so extreme that the rocket fins permanently deform or break, leading to a total failure of the rocket. To prevent this type of failure, it is vital to perform an aerodynamic analysis on the fins to ensure that the loads the rocket will experience during flight will not cause extreme flutter [14].

The flutter analysis done on the team's high-powered rocket was performed in ANSYS Static Structural with a Modal Analysis, assuming a worst-case scenario angle of attack of five degrees at the maximum velocity the rocket will experience, 110 m/s. However, to accurately simulate fin flutter, the location of the loads acting on the fin must be found. This was accomplished using XFLR5, assuming a thin plate airfoil. The fins can be represented as a thin plate airfoil due to their minimal thickness and rectangular-like 2D shape. The location along the chord where the drag and lift forces act on the fin are shown in the XFLR5 results in Figure 3-13 at the yellow arrow. This location is at about the quarter-chord point of the thin plate.

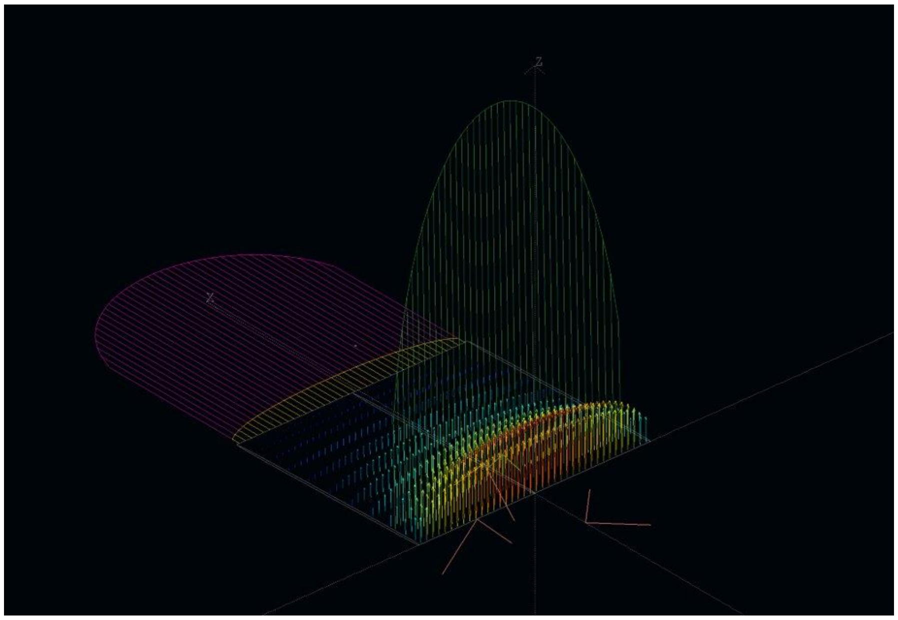


Figure 3-13: XFLR5 Analysis of a Thin Flat Plate

After defining the location of the loads and opening the fin model in ANSYS, this quarter-chord location was inputted as the location on the fin where the specified force is acting. This will be parallel to the rocket body and acting on the top of the fin in the ‘x’ direction. The force defined is the drag force acting on the rocket during flight. A drag force of 30 Newtons, found through the OpenRocket Simulation of the high-powered rocket, was used. Since the drag that the fins will individually feel will be much less, this drag force number was used as the maximum the fins will experience. The material chosen for the fin was oak wood, as it was the material closest to our actual fin material. In Figure 3-14, the maximum deformation due to the drag force can be seen as 0.0000002448 meters. The maximum stress felt by the fin, 1.3235 X 10^5 Pa (0.13235 MPa), is also shown in Figure 3-15. Both the deformation and stress experienced by the fin was extremely minimal and not likely to cause extreme oscillatory motion. The maximum stress found was also within a factor of safety of two for the Young’s Modulus of the fin’s material.

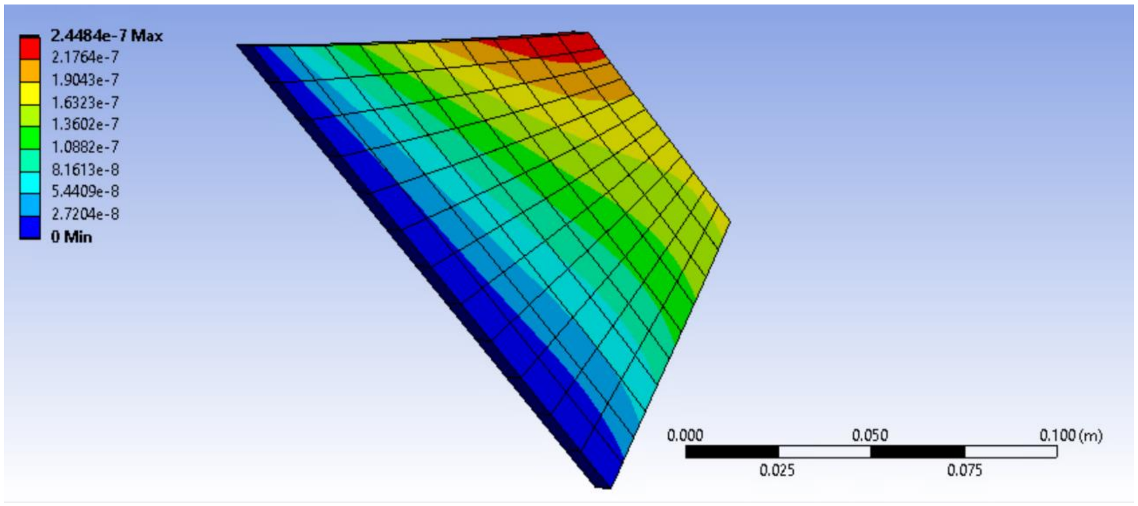


Figure 3-14: ANSYS Deformation Analysis

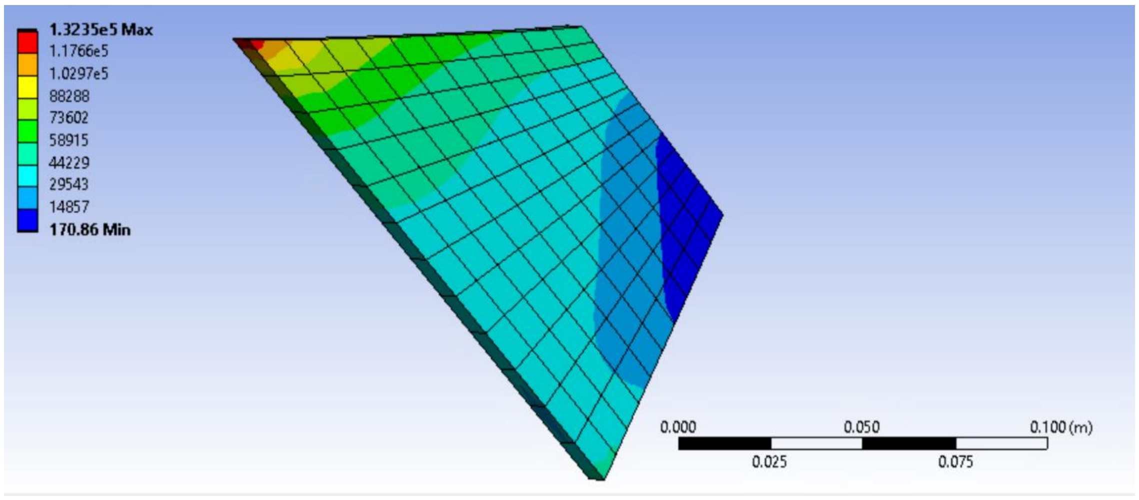
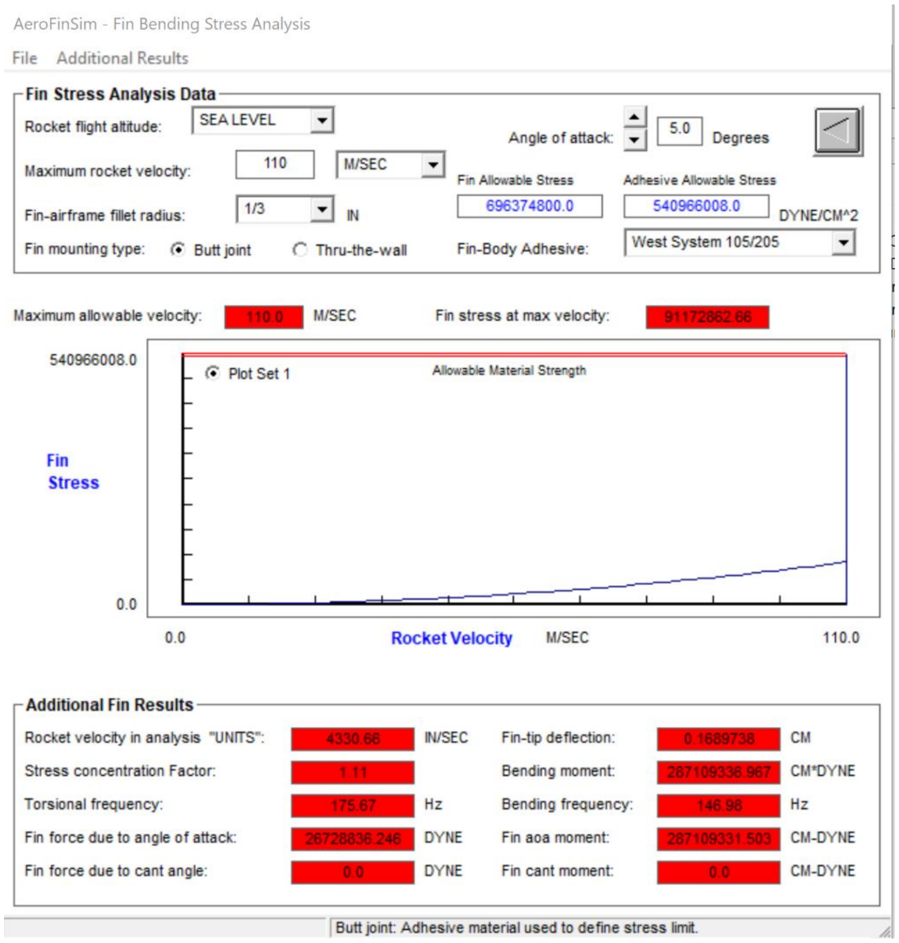


Figure 3-15: ANSYS Stress Analysis

To verify the results found through the ANSYS simulation, another simulation software called AeroFinSim was used. This software accurately estimates torsion-flexure flutter and is widely used to analyze fin flutter. First, the dimensions and material of the rocket's fins were inputted into the software [15]. The material chosen for the fins was oak wood, which was as close as possible to the actual material used on the rocket. Next, the maximum velocity and the angle of attack were input, and the simulation was run. In Figure 3-16, the results of AeroFinSim are displayed. In this figure, a graph of fin stress vs the rocket's velocity can be seen, where the maximum stress the fin can experience during flight before it breaks is 540966008 $Dyne/cm^2$ This is about 69 MPa. The stress that the fin experiences at 110 m/s is very minimal, as can be seen in the graph in Figure 3-16.



Fin Stress Analysis Data

Rocket flight altitude: SEA LEVEL

Maximum rocket velocity: 110 M/SEC

Fin-airframe fillet radius: 1/3 IN

Fin mounting type: [✓ Butt joint] Thru-the-wall

Angle of attack: 5.0 Degrees

Fin Allowable Stress	Adhesive Allowable Stress	
696374800.0	540966008.0	DYNE/CM^2

Fin-Body Adhesive: West System 105/205

Maximum allowable velocity: 110.0 M/SEC

Fin stress at max velocity: 91172862.66

Additional Fin Results

Rocket velocity in analysis "UNITS":	4330.66	IN/SEC	Fin-tip deflection:	0.1689738	CM
Stress concentration Factor:	1.11		Bending moment:	287109336.967	CM*DYNE
Torsional frequency:	175.67	Hz	Bending frequency:	146.98	Hz
Fin force due to angle of attack:	26728836.246	DYNE	Fin aoa moment:	287109331.503	CM-DYNE
Fin force due to cant angle:	0.0	DYNE	Fin cant moment:	0.0	CM-DYNE

Butt joint: Adhesive material used to define stress limit.

Figure 3-16: AeroFinSim Stress Results

The maximum stress on the fin found through ANSYS matches with the minimal stress found through AeroFinSim. With both results, it can be confidently assumed that the forces experienced by the rocket during flight will not cause detrimental fin flutter.

Resonant Motion

During the flight of a high-powered model rocket, the vehicle is susceptible to an anomaly known as roll coupling. This phenomenon occurs when the roll rate of the rocket becomes equal to the natural frequency of the pitch rate, causing resonant motion. The rocket will begin to tumble and become unstable once this threshold is reached.

A determining factor in the roll rate of the rocket is the canted angle of the fins. In simulations like OpenRocket, it is assumed that the fin cant angles are zero, resulting in a very small roll rate. However, this is hard to achieve. Thus, making it important to assess a high-powered model rocket's tendency to reach resonant motion.

The first step in this analysis was finding the natural frequency of the team's rocket and the corrective moment coefficient. These two equations are shown below:

$$W_n = \sqrt{\left(\frac{C_1}{I_L}\right)} \tag{3-12}$$

$$C_1 = \frac{\rho}{2} \cdot V^2 \cdot A \cdot C_{Na}(Z - W) \tag{3-13}$$

Where C_1 is the corrective moment coefficient and I_L is the longitudinal moment of inertia. Using variables found through OpenRocket, it was found that the natural frequency of the rocket was approximately 22 rad/sec.

Next, the roll rates for specified cant angles were found also through OpenRocket. Fin cant angles of one through five degrees were inputted into the simulation to show which cant angle would force the natural frequency to equal the roll rate. The graph of these results is shown in Figure 3-17.

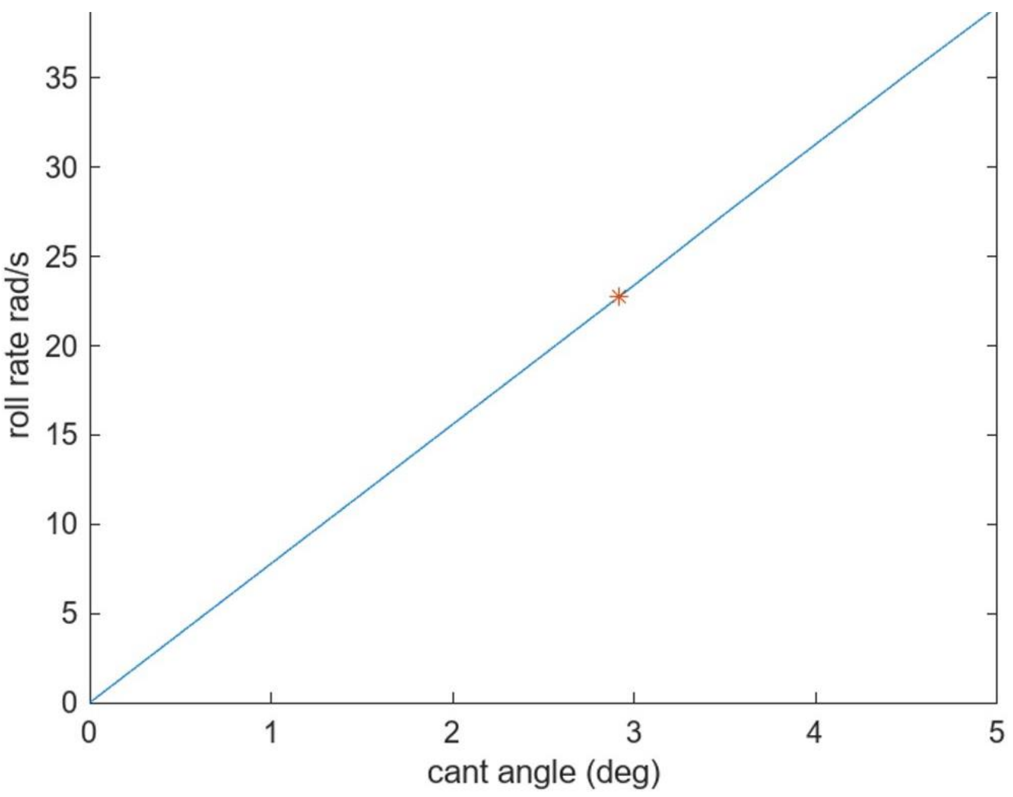


Figure 3-17: Roll Rate vs Fin Cant Angle at Maximum Velocity

The red star on the graph represents the natural frequency. The results suggest that roll coupling will occur at approximately three degrees of cant, so a fin cant angle of three degrees is the maximum angle the fins can be set at.

3.3.4. Results and Analysis from Flight Data

This section covers the failures and overall results of the rocket's flight from observed and recorded data.

Altimeter Results and Comparison to OpenRocket

Our OpenRocket model predicted a max apogee of 418 meters (1371.39 ft) and a total flight time of 45.7 seconds as described in section 3.2.1. The onboard commercial altimeters that fire the separation charges also record the flight of the rocket derived from their internal models and onboard sensors. Comparison between the simulated flight profile and the onboard flight readings can help us determine how accurate our prediction of the flight was. From the graph below in Figure 3-18, the RRC3 displays a maximum altitude of 1632.215 ft (497.50 meters) above ground level and a total flight time of around 45 seconds. Apogee is recorded at 11.4 seconds into flight.

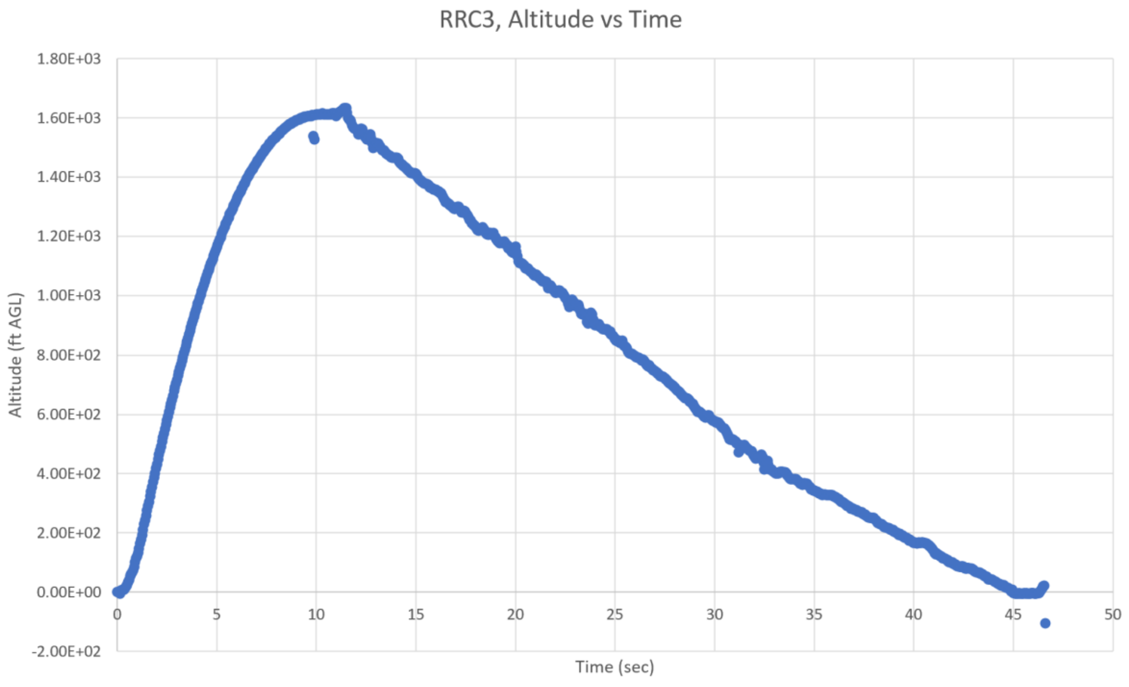


Figure 3-18: RRC3 Altitude Graph

Similarly, the TeleMega recorded a maximum altitude of 497.8 meters (1633.20 ft) and a total flight time of around 45 seconds. The apogee occurs 11.20 seconds into the flight. This flight profile can be seen in Figure 3-19.

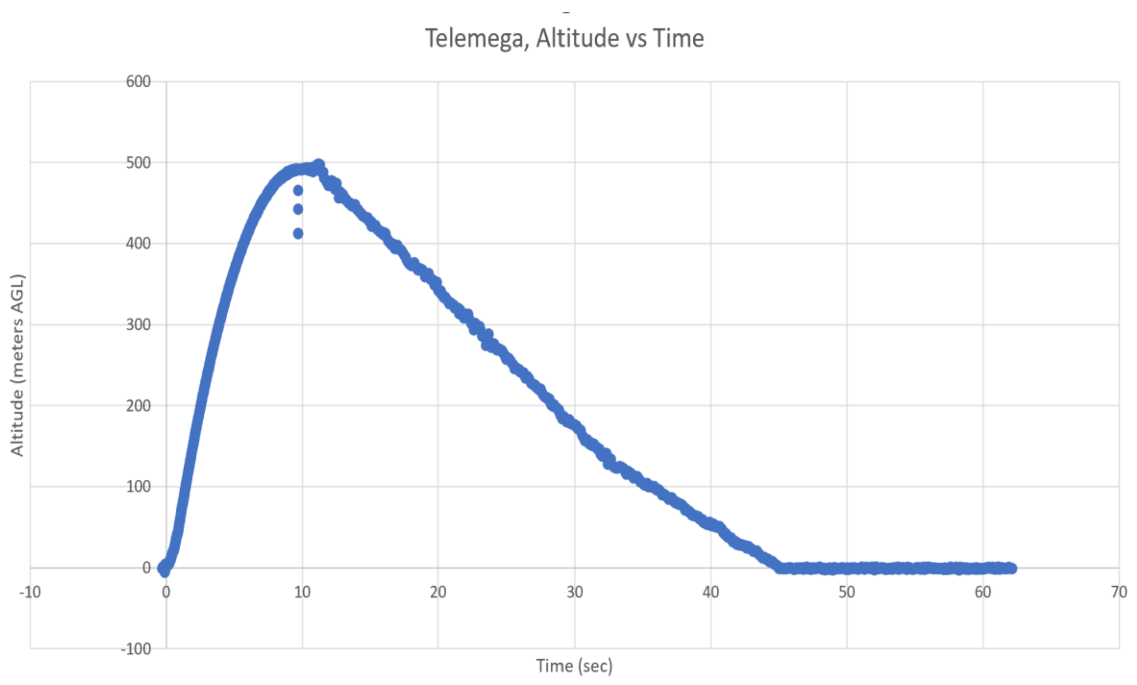


Figure 3-19: TeleMega Flight Data

The flight data indicates that our OpenRocket simulation underpredicted the apogee of the rocket by an average of 79.65 meters. This discrepancy in altitude can generally be attributed to variance in the weather, surface roughness of the rocket, and unique burn patterns. Environmental conditions that change from the simulation are the major cause for discrepancy as even small changes in humidity and temperature will impact the altitude of the vehicle. Upon inspection of the vehicle’s acceleration, the magnitude and duration of acceleration associated with the motor matches the performance values provided by the manufacturer. This indicates that the motor was not a major factor in the

Flight Computer Performance

The data logged by the custom flight computer indicates that the Kalman filter functioned as expected, providing a filtered estimate of the vehicle’s position and velocity throughout its flight. A comparison of the Kalman filter altitude estimate and purely barometric altitude estimate can be seen below, in Figure 3-20.

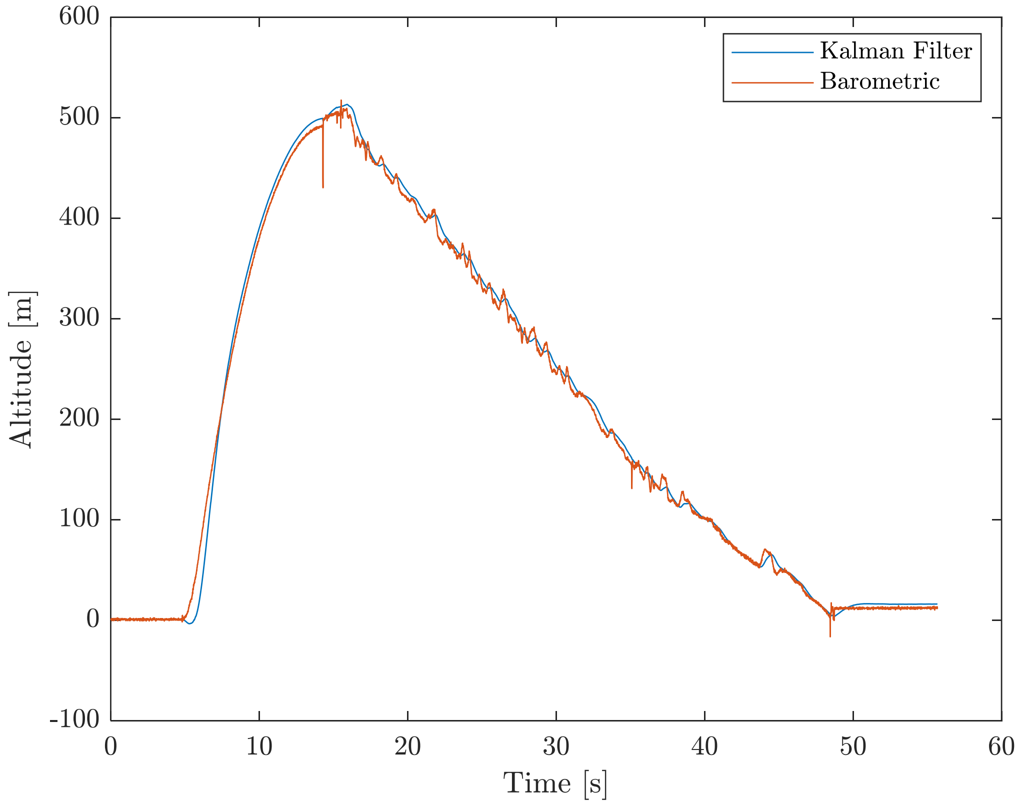


Figure 3-20: Flight Computer Altitude Comparison.

Although the Kalman filter performed as expected, the data from the flight computer indicates that the state machine did not trigger as expected, and thus did not detect apogee at the correct point in the flight. Figure 3-21 shows the state increase almost instantly during coast from state 1 (powered ascent) all the way to state 5 (on ground, landed). As mentioned previously, the state machine was coded such that after a specific timeout period in each state, it would automatically transition to the next state. Looking at the data from the flight computer, the boost state lasted for 5 seconds before transitioning, which was the timeout value programmed.

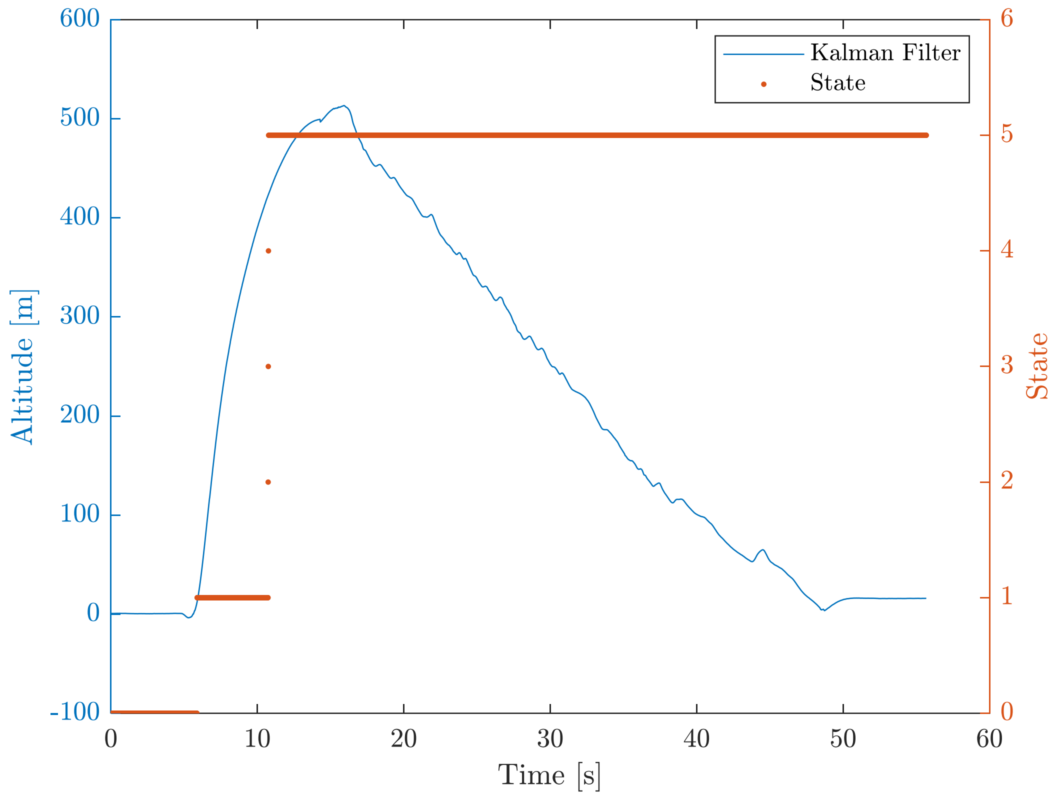


Figure 3-21: Flight Computer Altitude and State.

Looking more closely at the area of interest for the state change, Figure 3-22 shows the ½ second window where the state change occurred. From the plot, the state change occurred at a rate of one state per timestep (0.01 s). After review of the flight code, there was a bug that made it such that if any state timed out, the following state would then be presumed timed out, and so on, until the state reached the maximum value.

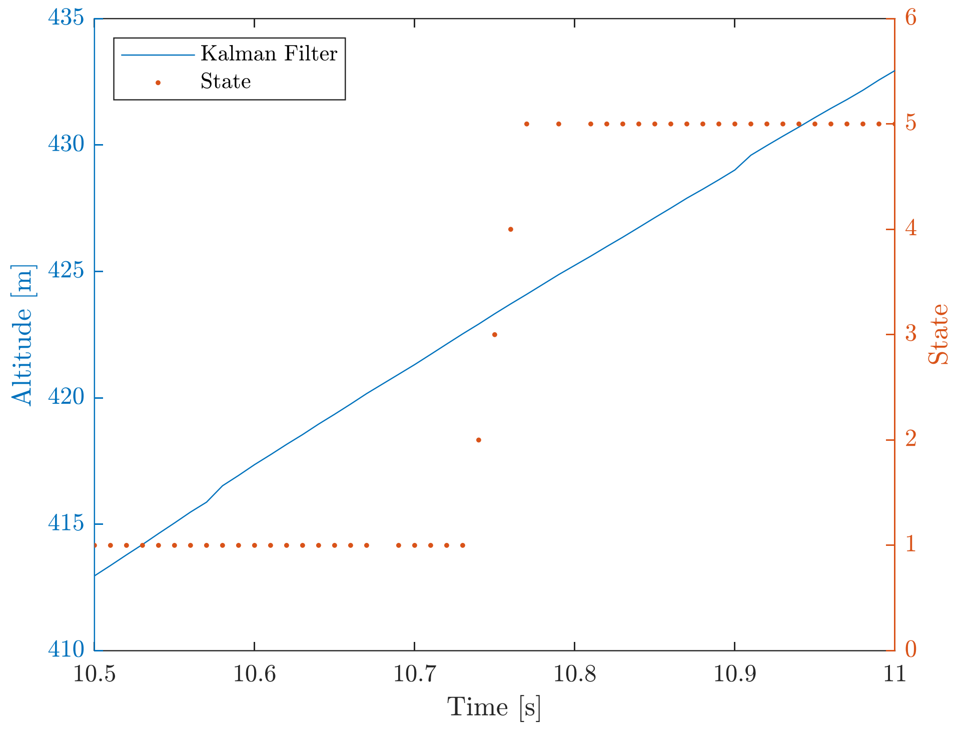


Figure 3-22: State Change Area of Interest.

Although the state change cascading immediately led to the apogee being incorrectly detected, it was not the root cause of the failure of the software. Figure 3-23 shows the measured acceleration as well as the state over the period of the motor burn. The state did not update to 1 (boost) until after motor burnout, where the measured acceleration went positive. This was caused by the difference between the acceleration measured by the accelerometer and the acceleration required for the state machine logic. As expected, the accelerometer measured data as it "experienced", meaning that as the rocket accelerated upwards, the acceleration was measured negative. To translate this measured acceleration into the real acceleration of the vehicle along its trajectory, the reading would simply need to be multiplied by negative one. This was not implemented in the final flight code, which led to the flight computer first seeing positive acceleration during coast, which triggered the launch state change.

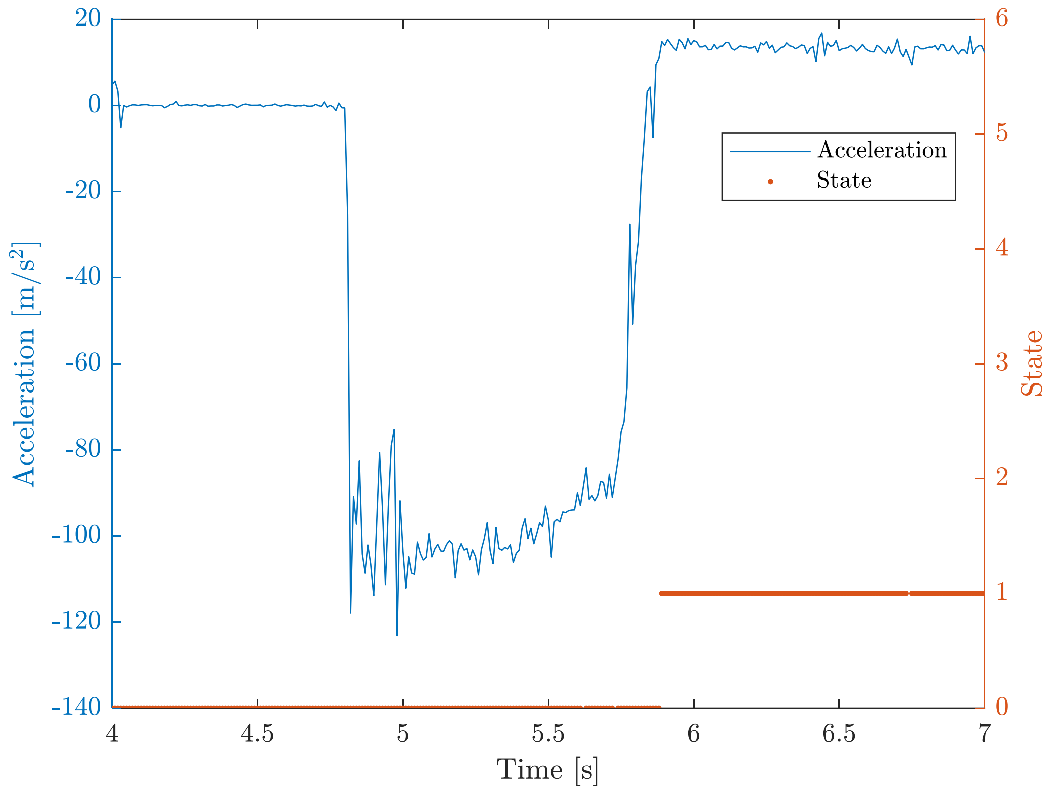


Figure 3-23: Motor Burn and State Change after Burnout.

4. Propulsion, Thermal, and Separation Systems

4.1. Methodology

4.1.1. Motor Selection

The motor selection process was mostly done using OpenRocket. A simulation was created for the innovative rocket mass to allow easy use for one motor for both rockets. The estimates of mass were found using CAD models with estimates based on material ideas. Using the Open Rocket simulation allowed for a variety of motors to be tested. There were three primary motor brands that came up from the recommendations from Open Rocket. These brands were Animal Motor Works (AMW), Aerotech, and Cesaroni Technology Incorporated (CTI). Since Aerotech motors mostly are assembled by the consumer it was decided that these motors would be avoided to allow a better chance of the rocket not failing by having professional made motors from the factory. AMW motors are extremely hard to find with fast shipping times, so they were also avoided. It was decided to focus more on CTI motors due to their abundance and faster shipping times. We found three viable CTI motors that would suit our needs. We wanted to find motors that had a predicted apogee between 1000 and 2000 feet and an off-rod velocity of at least 50 ft/s. The off-rod velocity is important because it allows the rocket to become aerodynamically stable once it leaves the launch rod.

4.1.2. Thermal Analysis Cantera

The thermal analysis of the selected rocket motor was performed using the Cantera toolbox for MATLAB. The Cantera toolbox provides a multitude of useful functions for working with chemical reactions. It should be noted that analysis assumes the components in the motor are in a gaseous form. This greatly simplifies the calculations and adds a negligible difference to the end results. For the initial conditions, temperature, pressure, and the mole fraction of the motor must be defined; the temperature is the ignition temperature of the motor, the pressure is the atmospheric pressure, and the mole fraction is for the species of the motor. The motor's ignition temperature and mole fraction were obtained from the motor's safety data sheet and were found to be 553.15 K (280°C) and 80% Ammonium Perchlorate with 20% Aluminum respectively. The pressure was set to the known atmospheric pressure at sea level, 101.3 KPa.

To perform an analysis, Cantera requires an input file that contains information about the species that will be used. While Cantera comes with many pre-built input files, none of them contain information for the aluminum species. Therefore, the input file created by the team from a previous iteration of the HPMR MQP was used.

With the proper input file and values for the composition set, we use the *equilibrate* function to force the mixture into equilibrium while being sure to specify that the specific internal energy and specific volume should be kept constant. Once the mixture is in equilibrium, we can extract the new properties from the mixture that will define our chamber properties used for future calculations. These properties in the chamber include the temperature, pressure, density, specific heats at constant volume and pressure, and mean molecular weight of the mixture.

Table 4-1: Cantera Input Properties

Input Variable	Value
Ammonium Perchlorate Mole Fraction	0.8
Aluminum Mole Fraction	0.2
Ammonium Perchlorate Molar Mass	$117.4891\ \left[\frac{g}{mol}\right]$
Aluminum Molar Mass	$26.982\ \left[\frac{g}{mol}\right]$
Ignition Temperature	$553.15\ [K]$
Ignition Pressure	$101300\ [Pa]$

Another piece of information needed to perform these calculations is the geometry of the nozzle. We measured our motor rocket nozzle and used it for our calculations. The throat diameter was 11.75 [*mm*] (0.4626 [*in*]) and an exit diameter of 18.7 [*mm*] (0.736 [*in*]). This gives a throat area and exit area of 108.4 [mm^2] and 274.6 [mm^2] respectively, and therefore an area ratio of 2.5328. Lastly, by dividing the specific heat at constant pressure by the specific heat at constant volume we can find the specific heat ratio.

Providing the specific heat ratio and the area ratio to the *flowisentropic()* function in MATLAB's aerospace toolbox, the values for the Mach number, temperature ration, pressure ratio, and density ratio at the exit can be obtained. Multiplying each of these by the chamber values obtained from Cantera gives us the properties at the exit of the nozzle.

The exit properties can be used to calculate thrust and specific impulse. First, we find the exit velocity using equation 9 below:

$$u_e = \sqrt{\frac{2\gamma}{\gamma - 1}\frac{\overline{R}}{\overline{M}_m}T_1\left(1 - \left(\frac{P_e}{P_1}\right)^{\frac{\gamma-1}{\gamma}}\right)} \tag{4-1}$$

$$u_e = \sqrt{\frac{2(1.21)}{(1.21) - 1}\frac{8314.5}{29}(2887)\left(1 - \left(\frac{76889}{1.81 \cdot 10^6}\right)^{\frac{1.21-1}{1.21}}\right)}$$

$$u_e = 2003 \left[\frac{m}{s}\right]$$

Using the exit velocity we can find the mass flow rate using equation 10, $\dot{m}$:

$$\dot{m} = \rho_e u_e A_e \tag{4-2}$$

$$\dot{m} = (0.1617)(2003)(2.85 \cdot 10^{-4})$$

$$\dot{m} = 0.0923 \left[\frac{kg}{s}\right]$$

Now, we can find thrust using the thrust equation (11):

$$T = \dot{m} \cdot u_e + (P_e - P_a)A_e \tag{4-3}$$

$$T = (0.0923)(2003) + (76889 - 101300)(2.85 \cdot 10^{-4})$$

$$T = 177.9\ [N]$$

Last year also performed an analysis through NASA's Chemical Equilibrium with Applications (CEA) website. We decided to repeat their analysis, however in reviewing their input values, we found some discrepancies in their calculations, mainly in the chamber to exit pressure ratio, initial ambient pressure, and oxidizer to fuel ratio. A table of the values we used is provided below in Table 4-2, Table 4-3 and Table 4-4 the pressure value was the same as the ignition pressure, the chamber to exit pressure ratio was obtained from evaluating the pressures returned by the initial Cantera results, and the oxidizer to fuel ratio is from comparing the mole ratio we initially provided with their molar masses to obtain the weights. After making these adjustments and running the software, a chart was created to show the difference in the resulting mole fraction which is shown in Figure 4-1.

Table 4-2: Property Table

Property	Value
Pressure	$1\ [atm]$
Chamber/Exit Pressure Ratio	23.5338
Oxidizer/Fuel Weight Ratio	17.4174

Table 4-3: Fuel Properties

Fuel Property	Value
Element Composition	Al
Weight Percentage of Fuel	100%
Temperature	$300\ [K]$

Table 4-4: Oxidizer Properties

Oxidizer Property	Value
Element Composition	$NH_4ClO_4(I)$

Weight Percentage of Oxidizer	100%
Temperature	$300\ [K]$

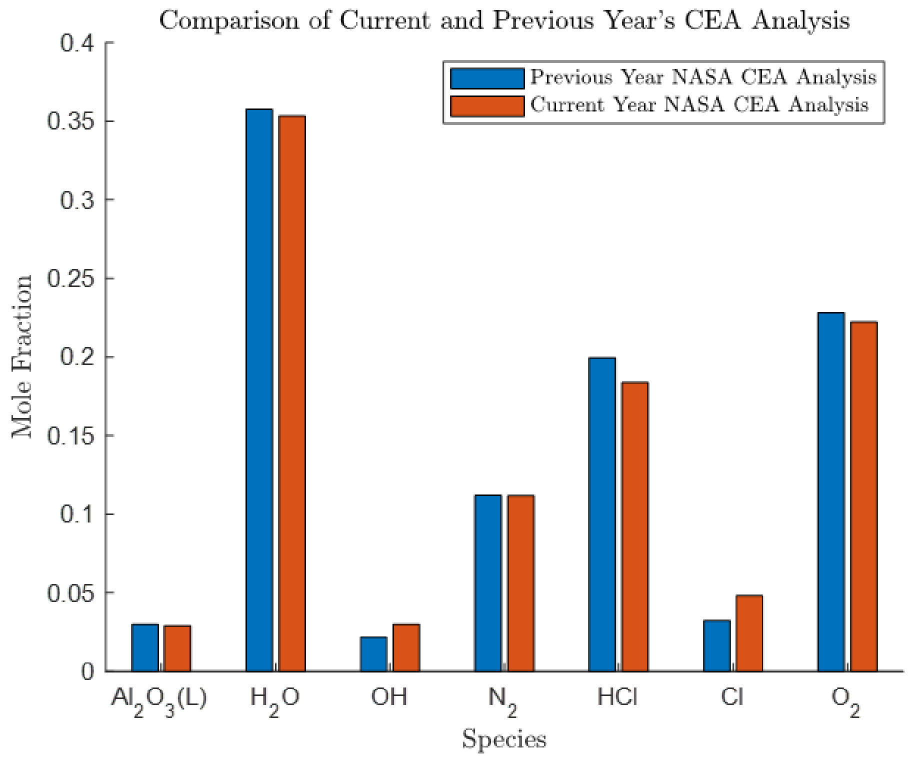


Figure 4-1: CEA Comparison Between Previous MQP and Current Project.

4.1.3. COMSOL Analysis of Solid Rocket Motor

COMSOL was used for the thermal analysis of our rocket. COMSOL allowed us to determine the fluid flow and heat transfer properties of the motor assembly. The multiphysics feature in COMSOL allowed us to link the two systems so that we could get a more accurate model for how the heat transfer and fluid flow interact.

In COMSOL, the user must define domains for the 2D asymmetric model we used. These domains are the physical pieces of the model that will be rotated about an axis to create a symmetric model. Figure 4-2 illustrates the domains we will be using and includes a legend for what each one represents.

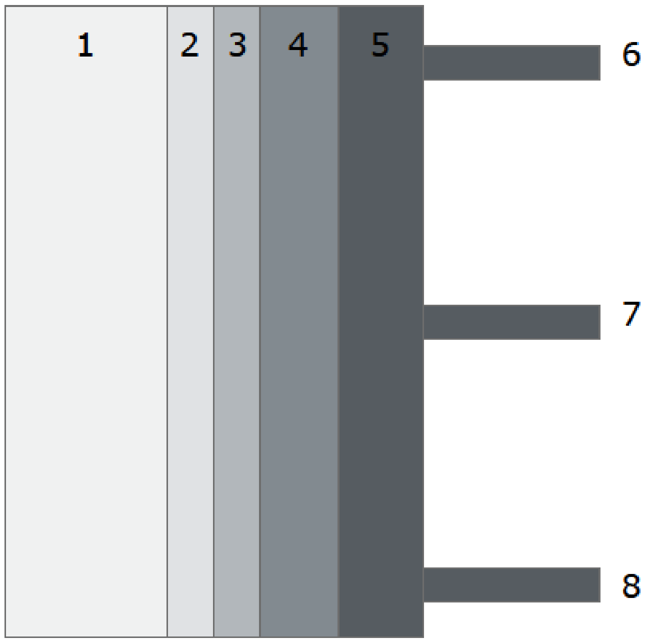


Figure 4-2: Layers of the Rocket Motor. Legend found below.

Domain:	Description
1	Combustion Gas
2	Combustion Flame Zone
3	Propellant Grain
4	Aluminum Casing
5	Fiberglass Tube Motor Mount
6	Aluminum Centering Ring
7 & 8	Wooden Centering Rings

When rotated this creates a 3D model that COMSOL will use to solve the heat transfer and fluid flow problems. To size the domains, we were able to gather the geometries for 5 through 8 from the CAD models we had created. For the motor which consists of domains 1 through 4, we knew the diameter of the total motor was 38mm. We defined the flame front and the propellant grain as 1mm each and then used a ratio of 6.8:1 for the combustion gas to the aluminum casing for the remaining radius.

We defined custom material properties for the combustion gas, flame zone, and propellant grain in COMSOL. The rest of the materials were already included in their libraries and were available to use.

Heat Transfer

COMSOL was used to calculate and monitor the heat transfer of the rocket motor throughout its burn. The point of the heat transfer analysis is to understand how hot the motor centering rings as well as the outer motor casing would get under normal operation. This is done to ensure that the materials we used can withstand any temperatures reached by the motor burn. If these materials could not handle the heat it would risk failure in flight while the motor is burning which could cause catastrophic failure and damage to people nearby. To model this different material was inserted into the model for the material properties that the team would be using as discussed above.

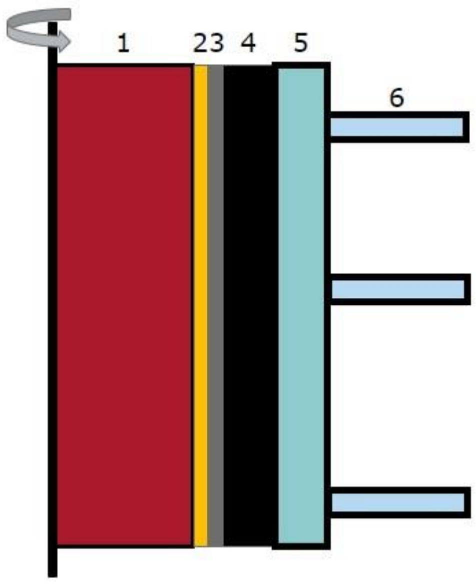


Figure 4-3: COMSOL Domain Model

Figure 4-3 shows the heat transfer model being used in all materials. To model heat flux, different boundary conditions were defined. The bold lines on the right side of the model indicated heat flux to the ambient air at 298K. The internal walls between every layer had defined heat fluxes based on the imported material properties for the materials used.

The main property that was inputted by the team was the heat generation coefficient. This coefficient was determined from the mass flow result from Cantera, the grain size of the rocket motor being used, and the heat of reaction from the Cantera model of the rocket motor. The equation (12) for this input can be shown below:

$$Q_0 = \frac{\dot{m_{fuel}} h_{RP}}{V} \tag{4-4}$$

a. This heat source information will allow the connection between the Cantera model and the COMSOL model to be realized. This connection allows for a more accurate representation of the heat transfer model for real world application to be used since real numbers from the chemical reactions are being used. This heat flux calculation will allow the team to understand how the rocket performs under its burn time.

Fluid Flow

To correctly calculate the heat transfer model in a solid rocket motor, the combustion gas exiting the motor must be considered. A weakly compressible laminar flow model was developed to model this combustion gas.

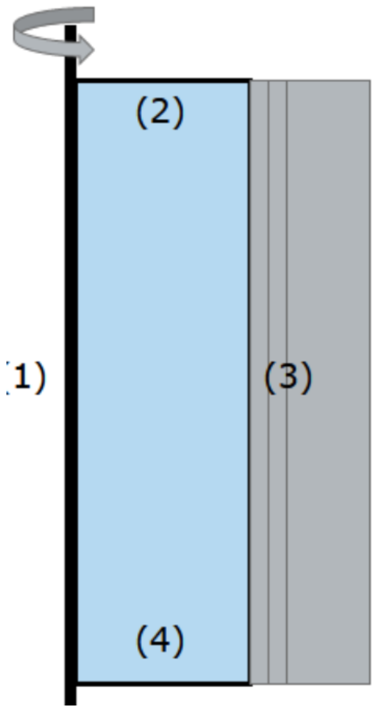


Figure 4-4: Diagram of Fluid Flow Model Domain in COMSOL

Figure 4-4 on the right is an illustration of the domain that COMSOL is using to model the fluid. It only uses domain 1 from the previous sections since it is the combustion gas. There are four boundaries in this simulation.where the domain is rotated about. There are no governing equations for this boundary and velocity vectors cannot go past it. Boundary 2 is defined as a wall.

$$\boldsymbol{u} = 0 \mid \boldsymbol{u} = velocity\ field$$

At boundary 2 the fluid cannot move past it and the wall remains stationary relative to the rest of the motor structure, so the velocity field at that point is defined as zero.

Boundaries 3 and 4 are a bit more complicated since they are the inlet and outlet respectively. The inlet is determined by the mass flow through the boundary.

$$-\int_{\partial\Omega} \rho(\boldsymbol{u} \cdot \boldsymbol{n}) d_{bc} dS = \dot{m} \qquad (4\text{-}5)$$

The equation (13) above is used by COMSOL to determine the velocity field and properties at the inlet. It is dependent on the mass flow and the physical properties of the combustion gas such as density. COMSOL gathers this information from the material properties defined for the combustion gas. For the outlet we use the following equations in COMSOL:

$$[-p\boldsymbol{I} + \boldsymbol{K}]n = -\hat{p}_0\boldsymbol{n}$$

$$\boldsymbol{K} = \mu(\nabla\mathbf{u} + (\nabla\boldsymbol{u})^T) - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})\mathbf{I}$$

$$\hat{p}_0 \leq p_0$$

$\boldsymbol{I}$ - identity matrix

$\boldsymbol{u}$ - velocity vector (m/s)

p_0 - absolute pressure (Pa)

μ - dynamic viscosity (Pa*s)

$\boldsymbol{n}$ - normal vector

The variable that these equations depend on is the pressure difference between the external temperature and the temperature inside the rocket motor, which is calculated from Cantera analysis. This difference in pressure is what drives the combustion gas outside of the end of the rocket motor.

Heat flux through solids occurs when heat energy is transferred through a solid material. There are two main mechanisms for heat transfer in solids: conduction and radiation.

Conduction is the transfer of heat energy through direct contact between particles in a solid material. It occurs as the particles in a solid vibrate and collide with one another, transferring some of their energy to their neighbors. The rate of conduction through a solid is determined by the thermal conductivity of the material, which is a measure of its ability to conduct heat.

Radiation is the transfer of heat energy through electromagnetic waves. It occurs as the particles in a solid vibrate and emit electromagnetic waves, which transfer energy to other objects that they encounter. The rate of radiation through a solid is determined by the emissivity of the material, which is a measure of its ability to emit electromagnetic waves.

Heat flux through liquids occurs when heat energy is transferred through a liquid. The main mechanism for heat transfer in liquids is convection. Convection occurs as the heated liquid expands and becomes less dense, rising to the top of the container. As it cools and becomes denser, it sinks to the bottom of the container, creating a circular flow known as a convection current. The rate of convection in a liquid is determined by the heat transfer coefficient, which is a measure of the ability of the liquid to transfer heat, and the temperature difference between the liquid and the surrounding environment. Other factors that can influence the heat flux through a liquid include the density and specific heat capacity of the liquid, as well as the flow rate. COMSOL uses the two equations to model the heat transfer either through a liquid or a solid.

$$\rho C_p \frac{\partial T}{\partial t} + \rho C_p u \cdot \nabla T + \nabla \cdot q \; = \; Q + Q_{ted} \tag{4-6}$$

$$q \; = \; -k \nabla T \tag{4-7}$$

Equations 15 and 16 are used to track the heat flux through a solid. The second equation is used during a steady state when the conditions don’t change with respect to time.

$$\rho C_p \frac{\partial T}{\partial t} + \rho C_{pu} \cdot \nabla T + \nabla \cdot q \; = \; Q \; + \; Q_p \; + \; Q_{vd} \tag{4-8}$$

$$q \; = \; -k\nabla T \tag{4-9}$$

Equations 17 and 18 are used to track the heat flux through a liquid. The second equation is used during a steady state when the conditions don’t change with respect to time.

Physics Coupling

To correctly solve this problem COMSOL Multiphysics coupling needed to be used to allow both heat transfer and fluid flow to be solved simultaneously since the heat transfer is dependent on the fluid flow. COMSOL was unable to solve the problem at the same concurrently since it did not have a successful solution. To prevent this problem, a stationary solution was used to solve the fluid flow portion of the problem since by the end of the burn time the rocket motor can be assumed to be in steady state flow. Next, the output of the stationary solution was used as the input of the time dependent solution for the fluid flow portion. This allowed accurate results for the temperature distribution of the rocket motor as well as the fluid velocity.

4.1.4. Separation System Baseline Rocket

All model rockets are required to have a recovery system to be flown and safely recover the rocket. Stage separation systems are often used to separate the rocket airframe and release a parachute. Being able to reliably separate two halves of the airframe allows the parachute to deploy so the rocket can be safely recovered and reused. To separate the stages for the recovery of a high-powered model rocket, there are two commonly used systems that we considered for use in the baseline rocket.

The first system involves igniting a black powder charge in the airframe. This generates gases inside of the closed volume and builds pressure until it forces the two sections of the airframe to separate. While this method of separation is easily obtainable and cost-effective, there is still the possibility that the reaction could leave behind a residue or that the detonation may damage one of the onboard systems by burning or breaking it from the immense pressure.

A CO_2 separation system was considered as well. This system uses a canister of pressurized CO_2 which rapidly expands when punctured, and uses the building pressure to separate the airframe. Although similar in principle to the black powder system, the CO_2 system has a much smaller chance of damaging other onboard systems due to the lack of actual detonation and residue. It is also much more reliable and predictable than the black powder system. One drawback of CO_2 systems is the larger cost. The CO_2 system we would use costs about \$200 whereas black powder typically costs under \$30. Due to budget constraints, it was decided that the black powder separation system would be used.

4.1.5. Separation System Innovative Rocket

For the innovative design aspect of our rocket, we decided to pursue a nosecone separation system based on an electromechanical system rather than an energetic one. While conventional systems like black powder and CO_2 work well, we wanted to see if a system could be created without the use of energetics. Removing energetics from the rocket design can help improve safety in the creation and use of our model rocket. Black powder is a highly explosive material and CO_2 is compressed gas. If either of them are handled poorly it could result in injury.

Our design consists of four pins mounted on linear rails that hold the nosecone coupler and upper airframe together. When the main parachute is commanded to deploy, the pins will be retracted using a plate with circular slots cut out of it that is rotated by a servo. Once the pins are retracted, there will be nothing holding the nosecone to the upper airframe where the main parachute is contained. To ensure positive separation of the nosecone and upper airframe, four springs are mounted on the bottom plate that will push off a centering ring placed below the system. The force of these springs being released would theoretically ensure proper stage separation, which can be verified during testing. An overview of the design can be found below in Figure 4-5.

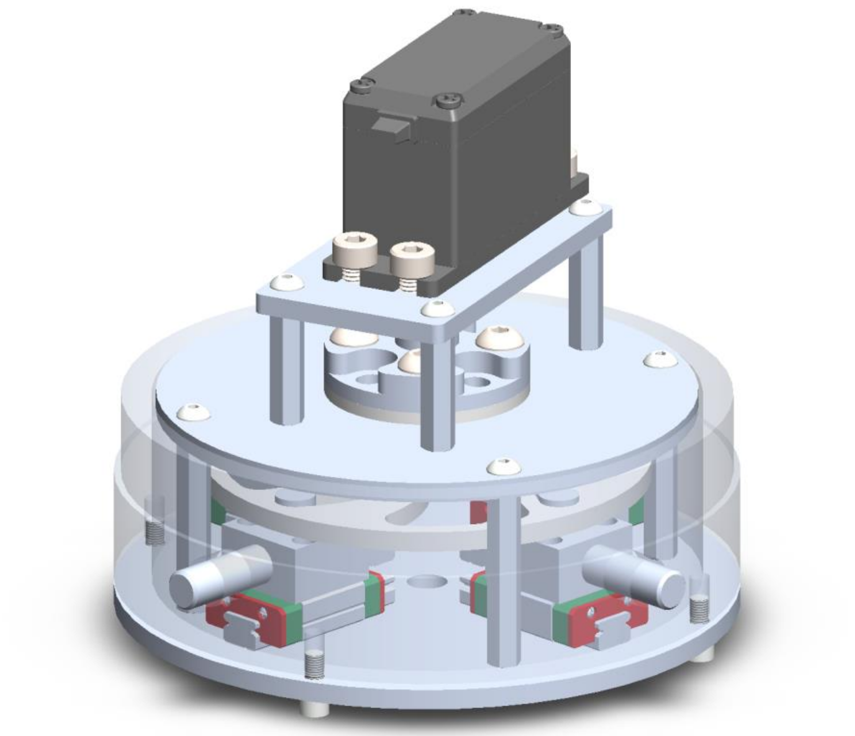


Figure 4-5: CAD Overview of Separation System.

This system offers a few benefits compared to black powder and CO_2 systems. In both of those cases, it must be ensured that the section of the rocket that the gases are released into is completely pressurized. If that section is not pressurized, the forces will not be able to break the shear pins and separate the stages. Shear pins are metal pins that are designed to break when a specific amount of force is applied to them. In this case, the force separating the rocket are the springs which do not require a pressurized container to work. In our system, unless a pin is deformed, it will be reusable many times without the need to replenish energetics.

One aspect of the mechanism we explored multiple options for was the part that would hold the upper airframe and nosecone together. The few options that were explored were a “pin” and a “fin” design as ideas for this part, as seen below in Figure 4-6 and Figure 4-7.

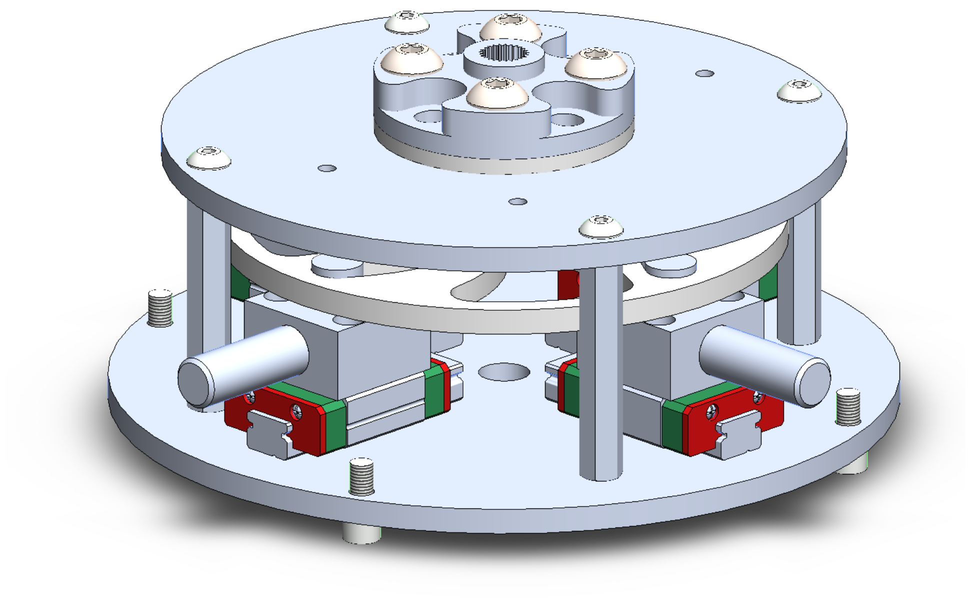


Figure 4-6: Pin Retention Concept

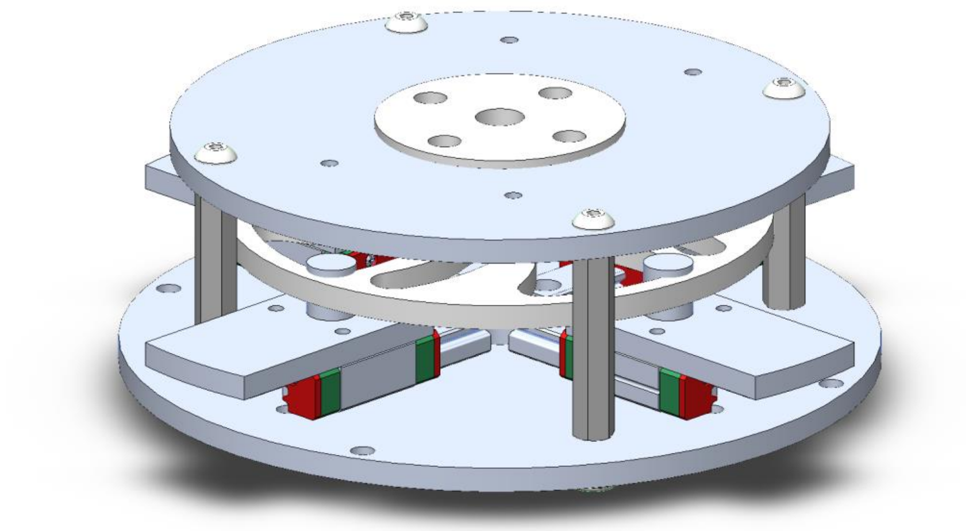


Figure 4-7: Fin Retention Concept

There were numerous benefits and disadvantages to both options. The pin would be more difficult to manufacture as it would require holes on multiple faces of the part to allow a metal dowel to be inserted and to connect it to the linear rails. The fin would just be cut out of a flat plate and have holes drilled into it. However, it would be much easier to drill holes in the airframe then cut the slots for the fins.

Preliminary structural analysis was performed using ANSYS to determine which of the two design options would be more likely to deform. In our design, the pin/fin can deform to a maximum of 1/32 in. In ANSYS, we loaded in the 3D model for both the pin and the fin. We then applied a deformation of 1/32 in at the end of the object. Finally, set ANSYS to solve the stress required to cause that deformation. Below are the resulting graphs for both the fin and the pin.

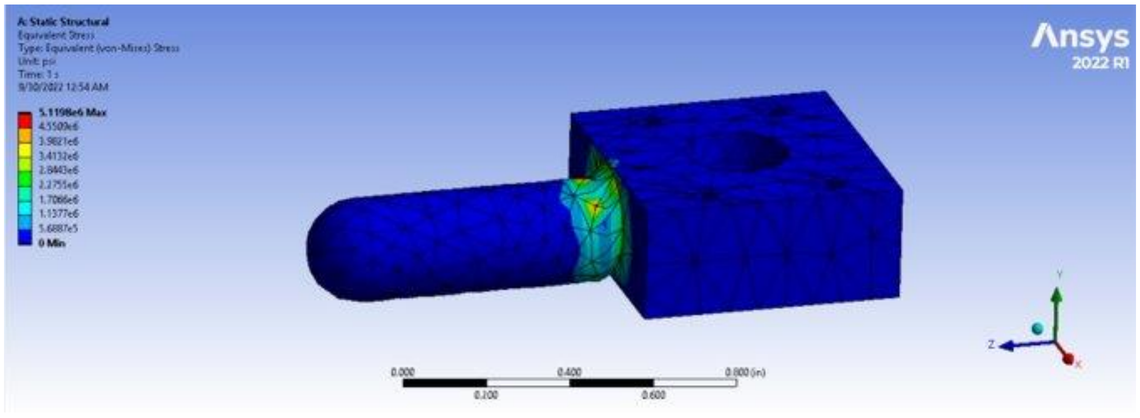


Figure 4-8: Max Deflection Simulated on the Pin Design

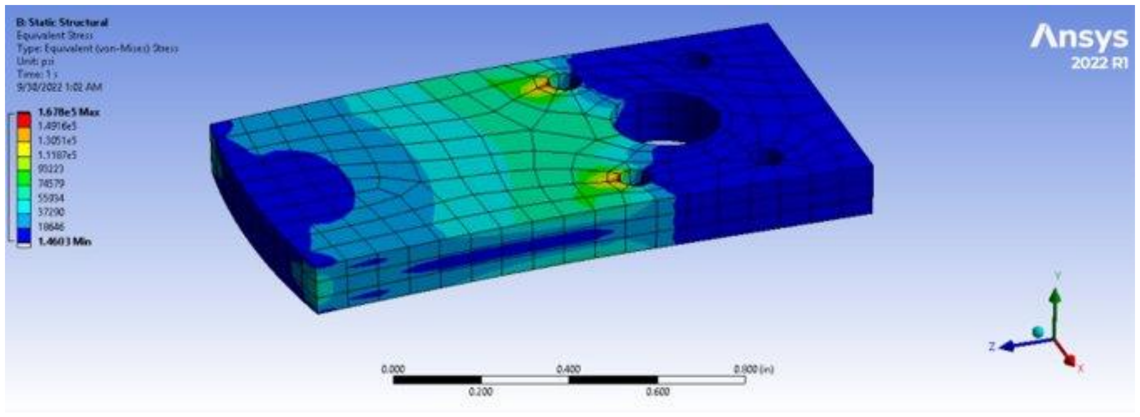


Figure 4-9: Max Deflection Simulated on the Fin design

The stress that the pin design would have to undergo to deform to the maximum is $5.12 \cdot 10^6$ psi while the stress for the fin design is $1.67 \cdot 10^5$ psi. Therefore, we can say that the pin will require a greater stress to deform and fail than the fin design. Due to this and the other factors outlined earlier, we decided to pursue a pin design.

4.1.6. Creation of Innovative Mechanism

The creation of our innovative mechanism was a multi-step process that required the purchase of standard hardware from suppliers, machining complex parts out of house, and machining simple parts in the Washburn Shop. For machining the simple parts in the CAD, the parts were taken to CAM software which can turn the design into tool paths the end mill can read and then create the part. This took multiple attempts to make one part and then check the holes for tolerance and clearance. Once all the final adjustments were made the machine then produced 4 identical pieces that were ready for use in the mechanism. The mechanism was put together and taken apart many times over the course of two weeks to fix tolerancing issues and plan out holes that needed to be drilled for wires to pass through without interference from moving parts.

4.1.7. Testing Procedure of Innovative Mechanism

To ensure the innovative mechanism would work under normal operation, test procedures were made to ensure the safety and reliability of the mechanism before in-flight use. The testing procedure involves experimentally determining the minimum sized springs that can be used and still eject the nosecone reliably and safely. The basic outline of the testing procedure is the following bulleted list:

Testing procedure:

Initial Test: Proof of concept

Using lighter springs with just assembly and nosecone:

- Attach rope to nosecone and rocket for manually disassembly if necessary
- Place assembly together on stand
- Using long wire to activate the servo manually with kill switch in middle
- Activate servo and see if mechanism comes apart
- If comes apart then good
    - If it does not come apart use ropes to carefully pull it apart staying away in case something lets go suddenly
- IF all is well move to next test

Testing force of parachute

- With drogue chute inserted use a force meter to measure the force required to pull drogue out.
- Add that force to the force of drag x 1.5 to get actual force necessary to pull drogue out
- Find the springs needed for that and test the mechanism with those springs to ensure all goes well (see test section 1 above

Seeing the above testing procedure, the first goal of the test is to determine if the mechanism would work the way it is designed. The second part of this test is to determine which springs are needed so that not too much force is used which ensures safer operation when people are loading the mechanism for flight.

4.1.8. Motor Structure Integrity

ANSYS simulation software was used to check the structural integrity of the motor mount and the forces it would undergo during launch. For this test the motor mount was subjected to 626 Newtons over a period of 1.18 s to replicate what it will experience during takeoff. Below are the two different tests ran using those configurations, one for the deformation of the parts and the other for max stress.

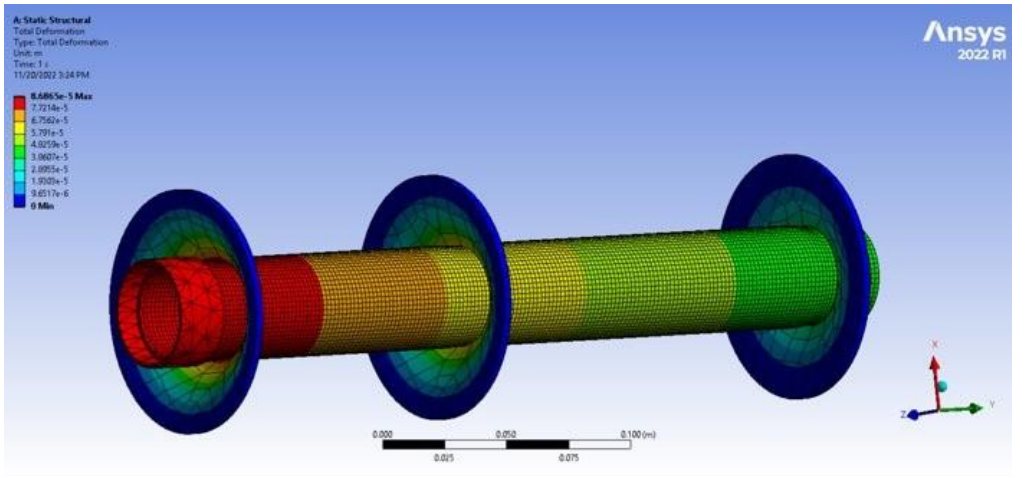


Figure 4-10: Max Deflection Simulated on the Motor Mount

The motor mount was expected to deflect a total of $8.69 \cdot 10^{-5}$ meters, which is within acceptable margins to not cause problems during the launch of the rocket.

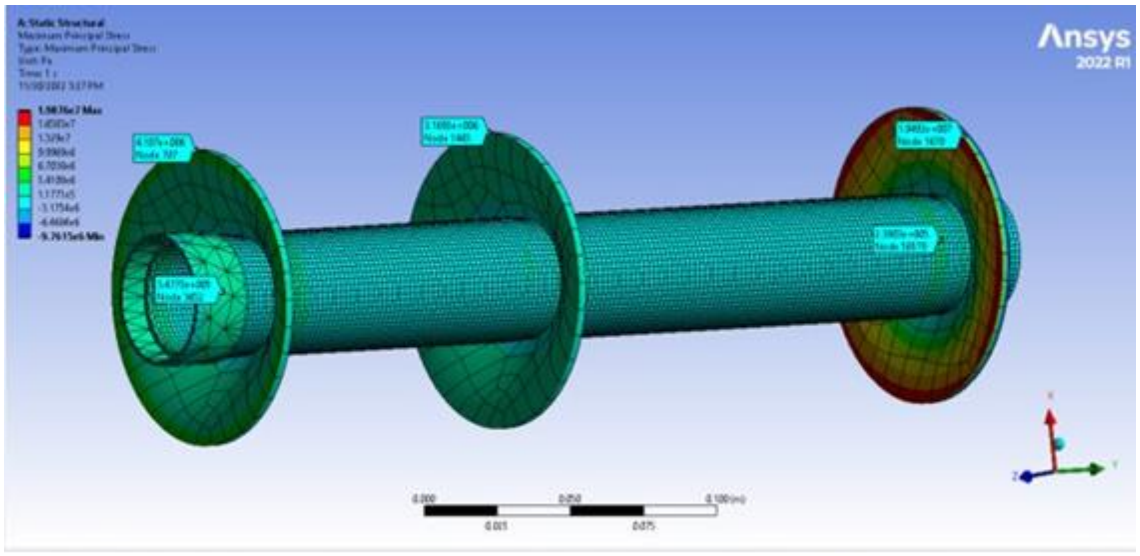


Figure 4-11: Max Stress Simulated on the Motor Mount

The motor mount is expected to reach a maximum stress of 19 MPa at rim of the last centering ring. This centering ring is made of aluminum and can withstand 207 MPa before failure, this is an order of magnitude less than the breaking point so that ring will be able to withstand the launch forces experienced. The same can be said with the other two centering rings that are made of plywood which have a max stress of 27.6 MPa but only reach 4.11 MPa on them. Additionally, the epoxy attaching the centering rings to the fiberglass tube will be experiencing this stress. The motor tube which is made of fiberglass has a shear stress of 58 MPa but only reaches 0.24 MPa during this test.

4.1.9. Umbilical Cord

To power and control the servo that is running our mechanism, we ran a long power cable to the recovery bay electronics and connected it with the recovery flight computer. The umbilical cord is used over an additional power supply and flight computer to lower the complexity of the final rocket design and save on mass. In total, we save around one pound in electronics and countless hours coding an additional flight computer. However, the umbilical cord comes with risks. The cord can get caught in the parachute when it is trying to deploy and cause catastrophic failure. To mitigate this the power cable is wrapped around the shock cord that connects the nose cone to the airframe. This should lower the chance that the parachute will get caught.

4.2. Results

4.2.1. Motor Selection

To achieve an off-the-rail velocity of over 50 ft/s and have the rocket apogee be between 1000 and 2000 ft, the team decided to go with the Cesaroni I540-16A motor. This solid rocket motor will provide a maximum thrust of 625 N, a total burn time of 1.18 s, and an impulse of 635 Ns. The motor also included its own timed black powder ejection system. This is just a cap on the end of the motor with delayed ejection. This motor will give our rocket an estimated off the rail velocity of 57.6 ft/s and reach a max apogee of 1549 feet. These estimates are based on our Open Rocket design which includes the final mass of the rocket and its aerodynamic design. Other motors were tested using Open Rocket, however, they all had shortcomings that stopped them from being selected as the final motor. A big issue we ran into with most motors was that they were sold out or no longer in production. They also had suboptimal off-the-rail velocities that were below the desired minimal, so the rocket would not have been stable in flight. Some of the other motors also burned for too long and would reach above our limit of 2000 ft. This would force us to look for other launch sites that were not within our general area and decrease the chance that we would be able to launch.

Additionally, we defined two backup motors in case our rocket ends up being over our estimated weight. One of these is the CTI J1055-7 motor and the other is the J1055-17 motor. The J1055-7 had an off-rod velocity of 75.6 ft/s and an apogee of 1961 ft. The J1055-17 had an off-rod velocity of 59.9 ft/s and an apogee of 1985 ft. Additionally, our baseline rocket motor selection is also the I540-16A since it results in an off-rod velocity of 59.7 ft/s and an apogee of 1769 ft.

4.2.2. Thermal Analysis Cantera

Once the mole fraction and initial parameters of the motor have been defined, we can gather information about its state after launch by using Cantera's *equilibrate* function. This function sets the motor at a chemical equilibrium while keeping specified properties constant. In our case, we specified internal energy and specific volume. We could then obtain the new properties of the motor at equilibrium, which are shown in Table 4-5.

Table 4-5: Chamber Properties Produced by Cantera

Property	Value
Enthalpy	$-3.459 \cdot 10^7 \left[\frac{J}{mol}\right]$
Temperature	$2887\ [K]$

Pressure	$1.809\ [MPa]$
Density	$2.189\ \left[\frac{kg}{m^3}\right]$
Specific Heat with constant volume	$1349\ \left[\frac{J}{K \cdot kg}\right]$
Specific Heat with constant pressure	$1635\ \left[\frac{J}{K \cdot kg}\right]$
Mean Molecular Weight	$29.04\ \left[\frac{kg}{kmol}\right]$

Using the properties of the motor, we can use equations in section 4.1.2 to get the properties of the motor, which are provided in Table 4-6. The properties of the motor also includes the new mole fraction, which is provided in Table 4-7

Table 4-6: Motor Properties

Property	Value
Velocity (u_e)	$1828\ \left[\frac{m}{s}\right]$
Mass Flow Rate ($\dot{m}$)	$0.1405\ \left[\frac{kg}{s}\right]$
Thrust (T)	$270\ [N]$

Table 4-7: Chamber Mole Fraction After Combustion

Species	Mole Fraction
Al_2O_3	0.0292

H_2O	0.3544
OH	0.0384
N_2	0.1169
HCl	0.1877
Cl	0.0461
O_2	0.2272

With results from Cantera, we can add those to the plot comparing the NASA Chemical Equilibrium Applications (CEA) analysis from the current and previous year. As seen in Figure 4-12 for all the species except for one, O_2, our new estimation was closer to the Cantera results than the previous year.

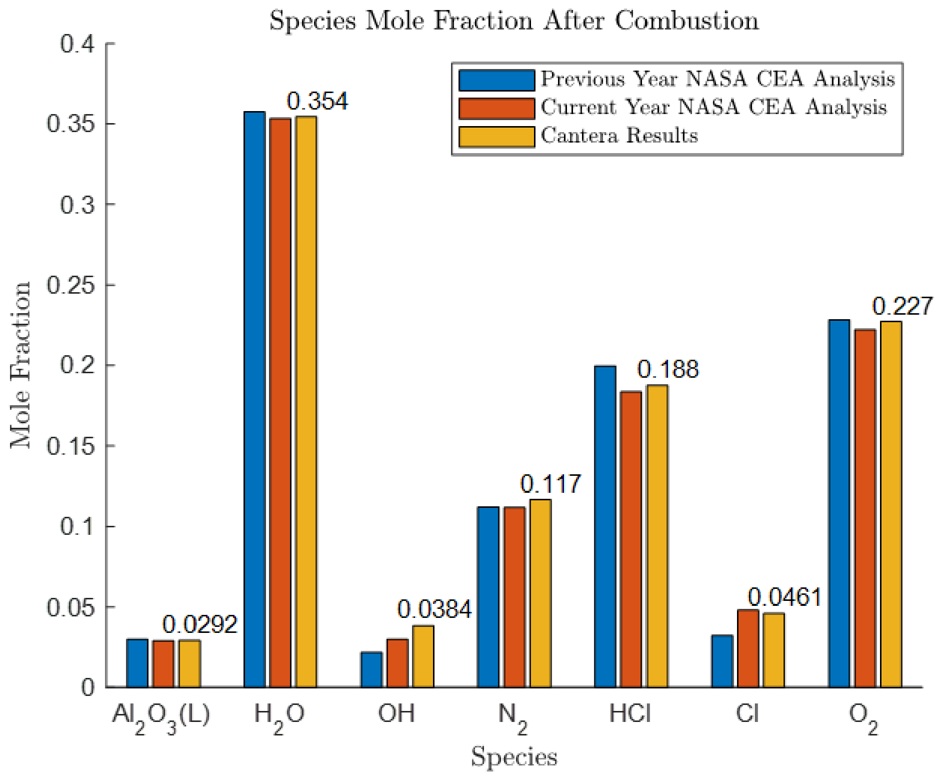


Figure 4-12: Figure of NASA CEA Comparison with Cantera Results.

4.2.3. Thermal Analysis COMSOL

Using COMSOL we were able to obtain velocity and thermal distribution models using the methods outlined in previous sections. The model geometry included not only the motor and its aluminum casing, but also the fiberglass motor mount and the centering rings.

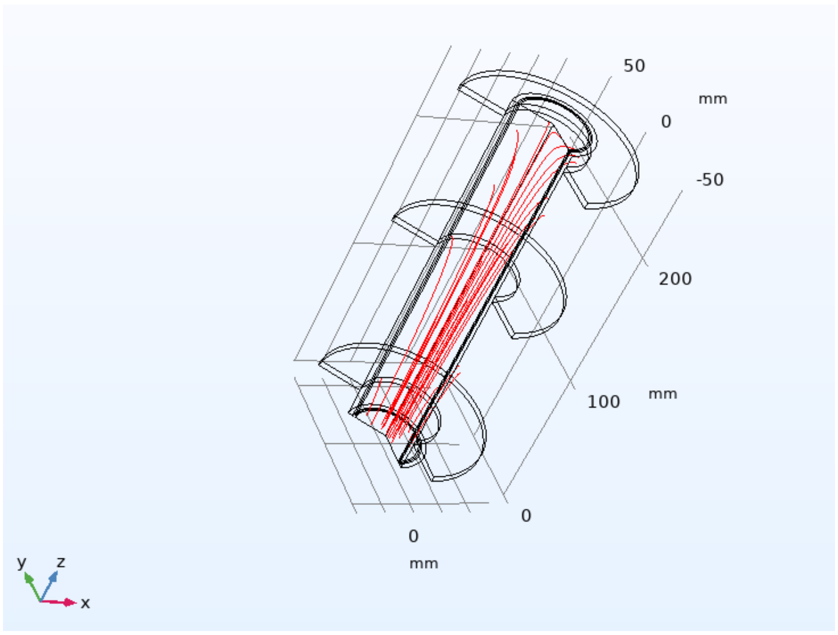


Figure 4-13: Velocity Streamlines in the Motor Case

The velocity streamlines show the direction of flow from the inlet wall to the outlet at the bottom of the motor. The fluid is accelerated downward out of the motor and into the rocket nozzle. This is the expected behavior of the fluid as the gas enters to the chamber from the combustion on the side of the grain and then the only path out of the motor is through the nozzle at the bottom.

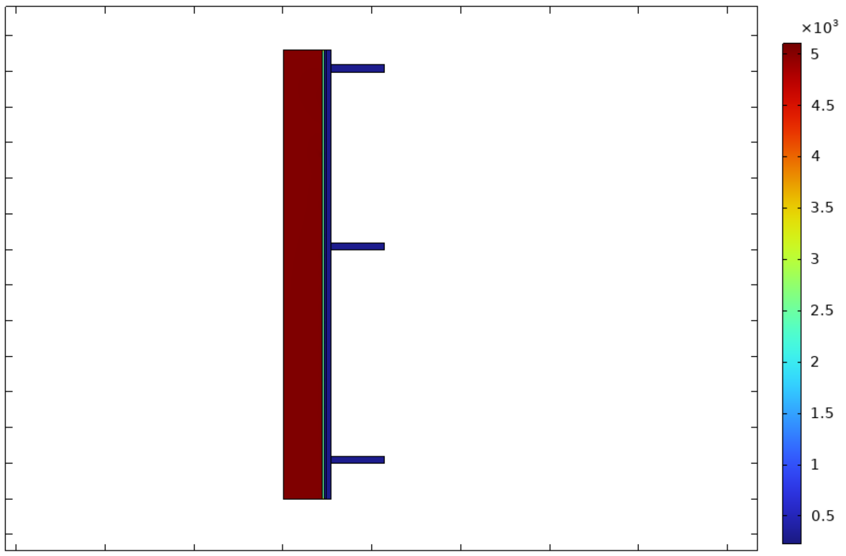


Figure 4-14: Temperature Distribution in Motor (units in K)

The temperature graph in the motor assembly can be found above. As expected, the domain with the highest temperature would be the combustion gas. The gas is the product of the propellant reaction so as expected it would have the highest temperature. The heat is transferred though the motor assembly and into the aluminum motor casing, the fiberglass tube and the centering rings. The aluminum casing reaches a temperature of 500 K, the fiberglass tube reaches a temperature of 320K and the centering rings reach a temperature of 300 K. These components do not reach concerning temperature levels and should not experience any temperature related failures during flight.

4.2.4. Motor Post Flight Analysis

After the fight the rocket, an inspection was done to see if all components performed as expected. Looking at the motor tube assembly post-flight it was noted that nothing abnormal occurred. There were no signs of overheating on the external of the motor and the assembly did not have any signs of failure occurring. This shows that the thermal analysis was accurate in determining that the exterior of the motor casing would not get hot enough to cause any damage. Since there were no stress cracks, deformations, or breaks in the motor centering rings the structural analysis appears to be correct. Since there were no fails present in the motor assembly it can be shown that the results, we obtained were true and that the motor upheld the standards that it was designed for.

4.2.5. Innovative Separation System Testing

The innovative mechanism went through rigorous testing to ensure it could be flight ready and safe for launch. The system was first tested separately of the whole rocket to see if the servo can actuate the pins through their whole range of motion. This first test revealed that the guide didn’t allow the pins to pull back far enough to fit entirely within its enclosure. Once this was tested numerous times and confirmed with the new changes that the pins had their full range of motion the assembly was put inside the coupler tube that would connect it to the nose cone. This coupler tube had cutouts to allow the pins to move in and out. It was once again tested in this enclosure for a full range of motion. This time the cutouts were stopping the pins, so the enclosure was removed, and the cutouts were made larger. It was again reassembled and with the larger holes the assembly had its full range of motion. This same process was repeated with the airframe attached to the nose cone and holes drilled out of the airframe corresponding to the holes in the enclosure. This confirmed to us that the pins would be able to engage and disengage while holding the nose cone and the upper airframe together. The mechanism was never tested to engage and disengage while under load, which could be a potential failure point.

The spring ejection system was mounted with the 4 strongest springs to the bottom of the innovative assembly and put into the nose cone assembly jig. The upper airframe was then lowered into a place where the holes could line up. The airframe was then ratcheted down to where the hole aligned perfectly, and two metal rods were then inserted into opposite holes to hold the nose cone and lower airframe together. It was then removed from the jig and the two rods were then quickly released. This resulted in the nose cone only moving a few centimeters up and not having enough force to separate it from the parachute. Given that this test was with the strongest springs the team had access to this idea was deemed unsafe and not flight worthy. Instead, black powder was used in its place to separate the nose cone from the upper airframe. The innovative mechanism was still flown on the launch to keep the mass and stability of the rocket the same.

5. Conclusion and Recommendations

5.1. Airframe Recovery System

Throughout the project, the ARS team designed and constructed the airframe of the rocket, recovery system, and recovery bay. Some setbacks did occur, primarily during the construction portion of the rocket. Many of the parts that were ordered took much longer to arrive than originally planned for, so there were periods where not much work was available, followed by very busy periods with a many different things that needed to be done at the same time. The primary hold up was the materials to perform the fiberglass layup, which delayed other sub-teams from working on the lower airframe. Moreover, the fiberglass layup was more difficult than anticipated, so multiple tests needed to be done quickly to ensure a proper layup could be applied to the final rocket. Based the process throughout the term, ARS has created several recommendations for future MQP teams. The recommendations are:

- Order materials as soon as possible. While it is important to consider all possible options, some materials may have long lead times so it is crucial to order them early. Additionally, unexpected shipping delays may occur so ordering early reduces the impact of potential delays.
- Create detailed procedures before performing tests. This recommendation stems from the first fin layup test the team performed, as the general plan was not clear enough to follow in the moment. Additionally, as the layup was being performed situations we did not initially anticipate occurred and we were unsure exactly how to properly proceed. However, since the layup needed to be done within the epoxy's working time, the layup needed to be completed without time to search for more answers.

5.2. Flight Dynamics Analysis

Throughout the duration of the project the FDA group experienced some setbacks and some successes. Creation and management of the OpenRocket model ensured that the vehicle would remain a safe and capable launch vehicle. From this simulation software, we were able to increase the capabilities of our custom flight software that ran on our custom flight computer. While we experienced some bugs in our code, some small changes as mentioned in the prior sections should resolve these errors to make a fully functioning system. Additionally, we investigated other characteristics of rocket flights such as fin-flutter and moment coupling so that we verified the safety of the vehicle. Looking back on our work, we have various recommendations that should be helpful to future rocketry MQP's or individuals hoping to complete similar tasks. They are as follows:

- Flight testing is crucial. If possible, completing multiple rocket flights is the only true way to verify confidence in any flight code or flight computer that is designed. Analyzing the data from these flights and iterating the code with each flight is the best way to find and debug errors. These test flights can be done on smaller and cheaper rockets so that more testing can be completed prior to a full-scale rocket.
- Create a validation system for code so that code works the way it was intended by the writer. We recommend following a similar standard as used by autopilots for FAA certified flight computers. This system follows a method of the writer of the code writing an additional script that runs the flight code to test it for errors. This would test each feature, timeout, and logic structure that could possibly exist in the code. This test code would be run by a completely separate member of the

team who would validate that the flight code passes the test code successfully. This testing individual should not be involved in the creation of either the test code or the flight code so that they remain independent of it passing or failing.

5.3. Propulsion, Thermal and Separation System

Over the course of this project the PTSS sub-team completed thermal analysis on our rocket motor using Cantera and COMSOL. Fluid flow and heat transfer models were developed and verified that the motor assembly would not fail during flight. Additionally, we developed an innovative stage separation mechanism using retractable pins and compressed springs. The pins would hold the nosecone to the upper airframe and when they were retracted compressed springs would eject the nosecone and pull out the main parachute. Unfortunately, during testing we learned the springs we selected were not strong enough to eject the nosecone so it was not flown. For future projects, we have a couple recommendations for improvements and further work that can be done:

- The innovative stage separation mechanism can be modified to include springs that are longer and extend past the end of the airframe when fully extended. By selecting these larger springs it should guarantee that the nosecone be fully ejected once the spring is allowed to extend. This would require the inner airframe to be reassembled to fit larger springs. Ideally the springs would also rest on a metal centering ring rather than the current wooden centering ring.
- The next steps for thermal analysis would be to analyze the fluid flow out of the rocket nozzle. Unfortunately, we did not have the time during this project to perform that analysis using COMSOL, but this would be something to expand upon in future projects.

5.4. Broader Impacts

It was reported that 2022 was the year with the most successful rocket launches into orbit to date [16]. As rocket launches become a growing part of our generation, the passion for high power model rocketry grows with it. There exists both positive and negative effects that result from this growing passion.

Exposure to model rockets is an efficient way to increase individuals' interests in areas of science, technology, engineering and math (STEM). It is a simple yet impactful way to introduce individuals into the field of STEM. This hands-on learning experience may engage individuals more compared to traditional conceptual learning in a classroom setting. The youth who are fortunate enough to participate in model rocketry may become future engineers, scientists, astronauts, pilots and more. Increasing youth involvement in the growing passion model rocketry may increase involvement in the STEM field along with minority representation in the STEM field. However, model rockets may have a negative impact to our community. At the CATO launch site, there were a few rockets and parts of rockets that flew out of the club's vicinity and were unretrievable. The HPMR team did not stay long enough to see if these parts were retrieved but if they were not, this negatively impacts the environment. This affects the existing lifeforms in the area, such as the animals, while polluting the area, such as bodies of water, as well. In addition, toxic particles from may be released from the rockets which negatively impact the environment. Nevertheless, the decision to have this be a reusable rocket rather than a single-use rocket along with constructing the rocket using existing material from a previous project is a better choice for the environment.

6. References

[1] Gasmire, Charlie. "Model Rocket Engine Sizes and Classifications." *The Model Rocket, 30 July 2020,* https://themodelrocket.com/model-rocket-engine-sizes-and-classifications/.

[2] Dunbar, Brian. "Parachute Tests." *NASA, NASA,* https://www.nasa.gov/mission_pages/constellation/multimedia/parachute_tests.html.

[3] "Beginner's Guide to Aeronautics*."* *NASA*, NASA, https://www.grc.nasa.gov/WWW/k-12/airplane/.

[4] "Propulsion System." *NASA,* NASA, https://www.grc.nasa.gov/www/k-12/rocket/rocket.html#:~:text=The%20propulsion%20of%20a%20rocket,the%20air%20and%20through%20space.

[5] Lederman, Jason. "Watch: 24 of the Most Famous Space Launches Ever." *Popular Science*, 26 Apr. 2021, https://www.popsci.com/most-famous-space-launches-ever-video/.

[6] "Rocket Principles." *Goddard,* http://abyss.uoregon.edu/~js/space/lectures/lec03.html

[7] "Solid Rocket Engine." *NASA*, NASA, https://www.grc.nasa.gov/www/k-12/rocket/srockth.html.

[8] "TeleMega." *Atlus Metrum*, https://altusmetrum.org/TeleMega/.

[9] Missile Works Corporation - RRC3, *https://www.missileworks.com/rrc3/.*

[10] Tianxiang Wang, et al. "In-Plane Mechanical Properties of Birch Plywood." *Construction and Building Materials*, Elsevier, 18 May 2022,

https://www.sciencedirect.com/science/article/pii/S0950061822015252#:~:text=Tensile%20properties%20of%20the%20birch,is%20more%20than%20100%20MPa.

[11] “Sparkfun MicroMod STM32 Processor.” DEV-17713 - SparkFun Electronics, https://www.sparkfun.com/products/17713.

[12] 57bravo, et al. “Sparkfun MicroMod Teensy Processor.” *DEV-16402 - SparkFun Electronics*, https://www.sparkfun.com/products/16402.

[13] Carvalho, Lucas & Claudino, Geovanio. (2019). CFD Analysis of Drag Force for Different Nose Cone Design

[14] Fin Flutter Analysis .” *California Polytechnic State University*, https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1113&context=aerosp.

[15] AeroFinSim.” AeroFinSim *Description Page,* https://www.aerorocket.com/finsim.html.

[16] Witze, Alexandra. “2022 Was a Record Year for Space Launches.” *Nature News*, Nature Publishing Group, 11 Jan. 2023, https://www.nature.com/articles/d41586-023-00048-7.

7. Appendices

Appendix A: Equations in FDA Analysis Task 1

Condition	**Equation**	**Variable**

Drag Equation	$D = C_d \cdot \rho \cdot \frac{V^2}{2} \cdot A$	C_d = Drag Coefficient
		ρ = Density (kg/m^3)
		V = Airspeed (m/s)
		A = Cross Sectional Area (m^2)

Appendix B: Equations in FDA Analysis Task 3

Condition	**Equation**	**Variable**
System Model	$x_k = \Phi_{k-1} \cdot x_{k-1} \cdot w_{k-1}$	x_k = State of System at step k Φ_k = Time Matrix w_k = Noise in system model
Measurement Model	$z_k = H_k \cdot x_k + v_k$	z_k = Measurement v_k = Measurment Noise H_k Transforms the system into a measurement

State Estimate Extrapolation	$\overline{x}_{k(-)} = \Phi_{k-1} \cdot \overline{x}_{k-1(+)}$	
State Estimate Update	$\overline{x}_{k(+)} = \overline{x}_{k(-)} + K_k\left[z_k - H_k \cdot \overline{x}_{k(-)}\right]$	K_k = Kalman Gain Matrix
Error Covariance Extrapolation	$P_{k(-)} = \Phi_{k-1} \cdot P_{k-1(+)} \cdot \Phi_{k-1}^{T} + Q_{k-1}$	$P_{k(-)}$ = Error Covariance Update Q = Standard Deviation
Error Covariance Update	$P_{k(+)} = [I - K_k \cdot H_k] \cdot P_{k(-)}$	
Kalman Gain Matrix	$K_k = P_{k(-)}H_k^T\left[H_k P_{k(-)} H_k^T + R_k\right]^{-1}$	R_k = Standard Deviation

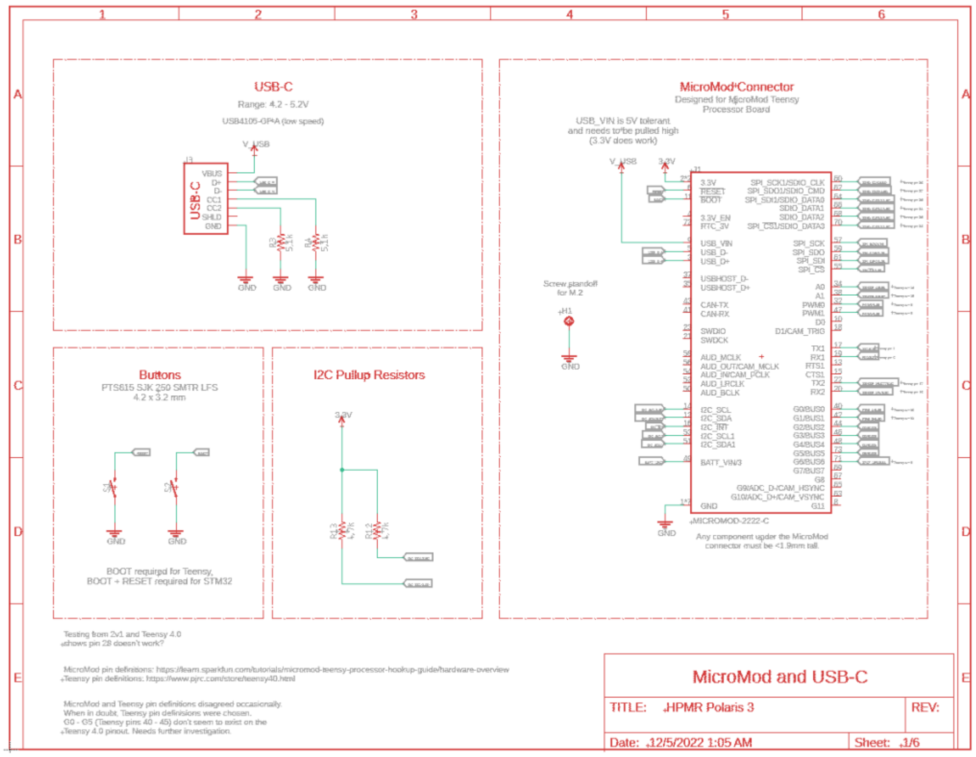
Appendix C: Equations in PTSS Analysis Task 3

Condition	**Equation**	**Variable**
Exit Velocity	$u_e = \sqrt{\frac{2\gamma}{\gamma - 1}\frac{\overline{R}}{\overline{M_m}}T_1 \cdot \left(1 - \left(\frac{P_e}{P_1}\right)^{\frac{\gamma-1}{1}}\right)}$	$\gamma =$ Specific Heat Ratio = 1.21 $\overline{R}$ = Gas Constant = 8314.5 $\overline{M}$ = Mean Molecular Weight T = Temperature

		P_e = Exit Pressure P_1 = Pressure
Mass Flow Rate	$\dot{m} = \rho_e u_e A_e$	$\rho =$ Density u_e = Exit Velocity A_e = Exit Area
Thrust	$T = \dot{m} \cdot u_e + (P_e - P_a) \cdot A_e$	$\dot{m}$ = Mass Flow Rate u_e =Exit Velocity P_e = Exit Pressure P_a = Area Pressure A_e = Exit Area
Heat Generation Coefficient	$Q_0 = \frac{\dot{m_{fuel}} \cdot h_{RP}}{V}$	$\dot{m_{fuel}}$ = Mass flow rate of Fuel h_{RP} = Grain size of the Rocket Motor V = Heat of Reaction of the Rocket Motor
Inlet Properties	$-\int_{\partial\Omega} \rho(u \cdot n) d_{bc}\, dS = m$	$\rho =$ Density $u =$ Velocity Vector $n =$ Normal Vector

Heat Transfer Model through a Solid	$\rho C_p \frac{\partial T}{\partial t} + \rho C_p u \cdot \nabla T + \nabla \cdot q$ $= Q + Q_{ted}$	$\rho =$ solid density (kg/m^3) C_p = Solid Heat Capacity at Constant Pressure (J/kg-K) K = Solid Thermal Conductivity (W/m-K) U = Velocity Field (m/s) Q = Heat Source (W/m^3) Q_{ted} = Thermoelastic Damping (W/m^3)
Heat Flux through a Solid	$q = -k\nabla T$	k = Solid Thermal Conductivity T = Temperature
Heat Transfer Model through a Liquid	$\rho C_p \frac{\partial T}{\partial t} + \rho C_{pu} \cdot \nabla T + \nabla \cdot q$ $= Q + Q_p + Q_{vd}$	Q_p = Pressure Work (W/m^3) Q_{vd} = Viscous Dissipation
Heat Flux through a Liquid	$q = -k\nabla T$	k = Solid Thermal Conductivity T = Temperature

Appendix D: Flight Computer Schematic Diagrams



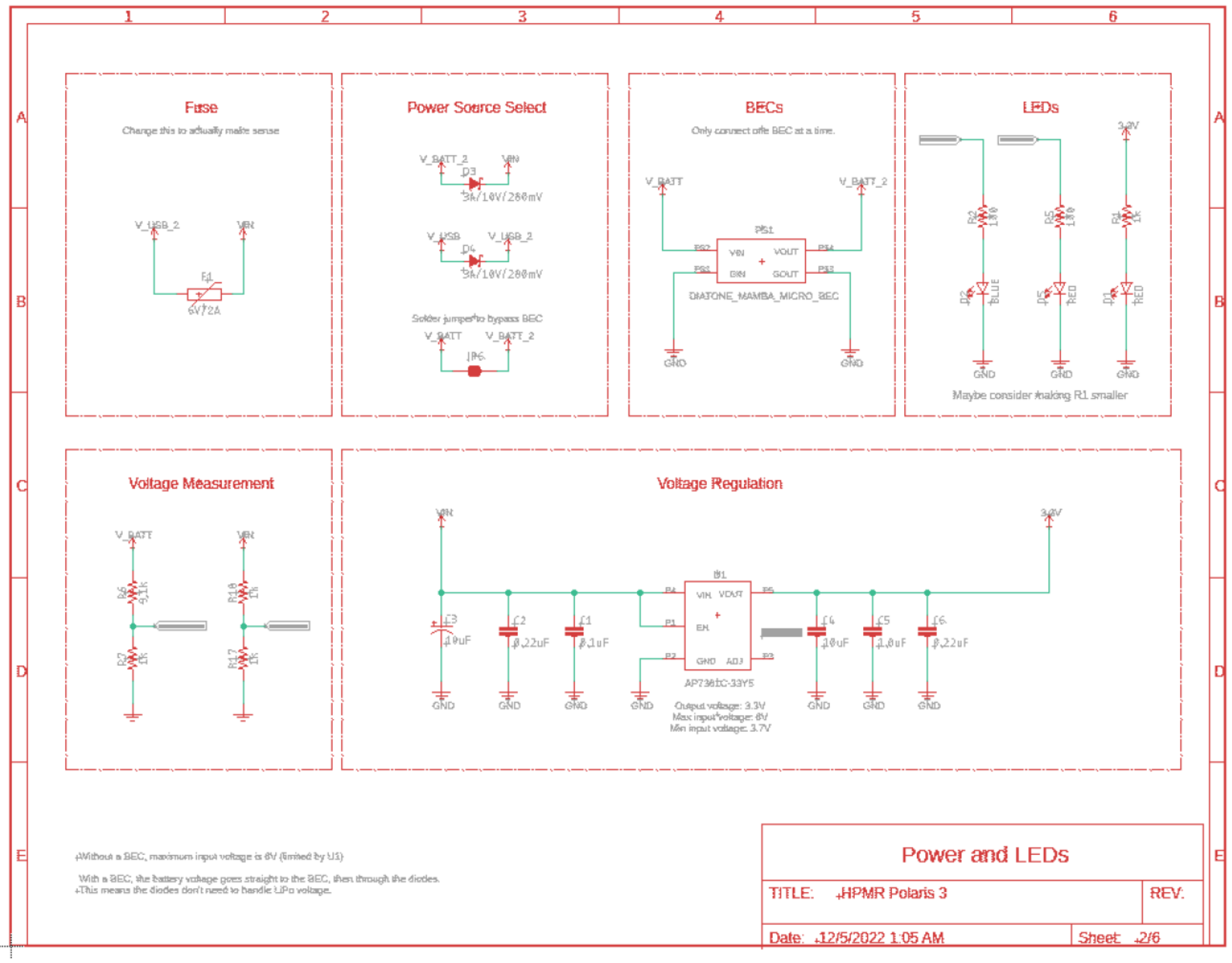
Testing from 2v1 and Teensy 4.0
shows pin 28 doesn't work?

MicroMod pin definitions: https://learn.sparkfun.com/tutorials/micromod-teensy-processor-hookup-guide/hardware-overview
Teensy pin definitions: https://www.pjrc.com/store/teensy40.html

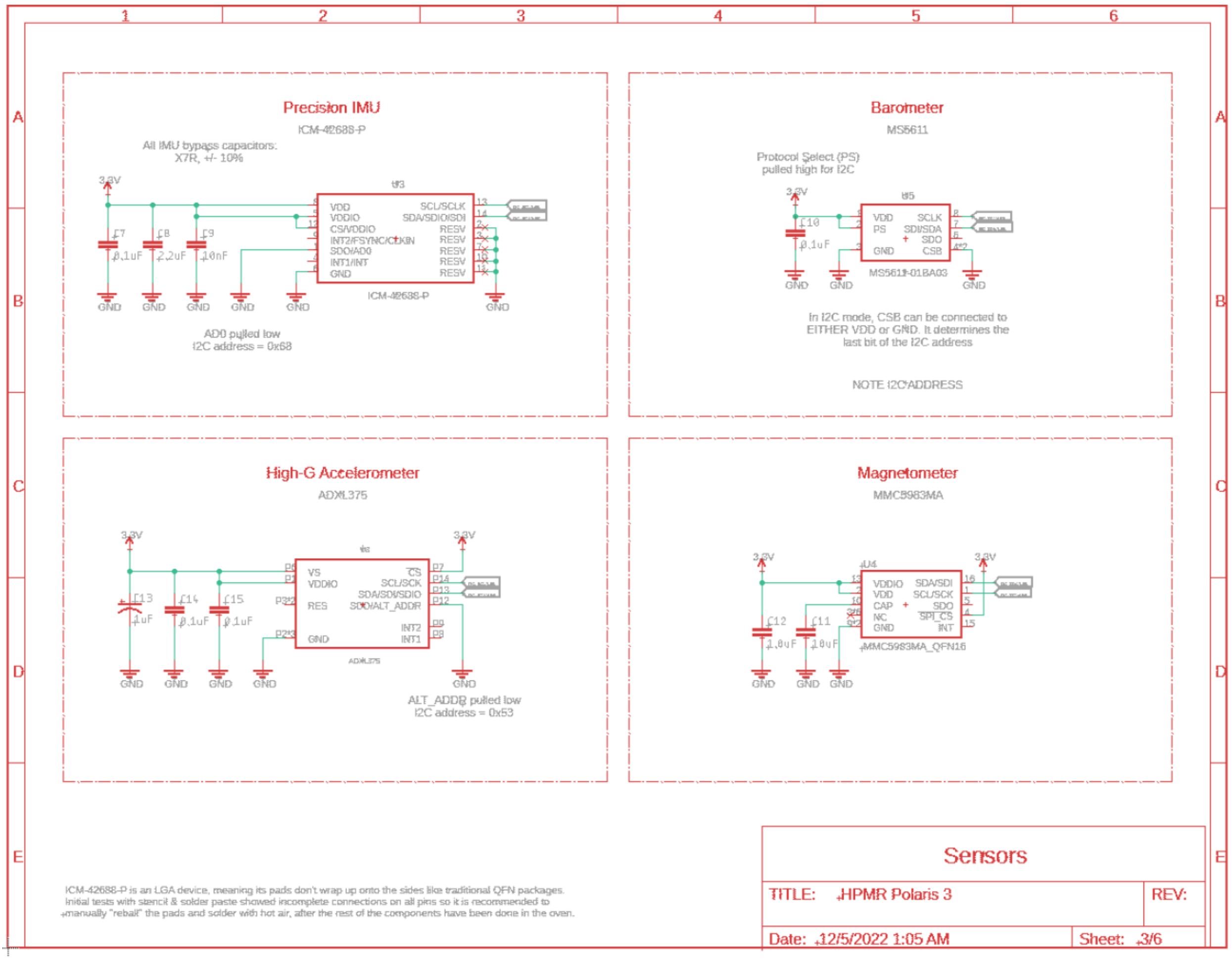
MicroMod and Teensy pin definitions disagreed occasionally.
When in doubt, Teensy pin definisions were chosen.
G0 - G5 (Teensy pins 40 - 45) don't seem to exist on the
Teensy 4.0 pinout. Needs further investigation.

MicroMod and USB-C

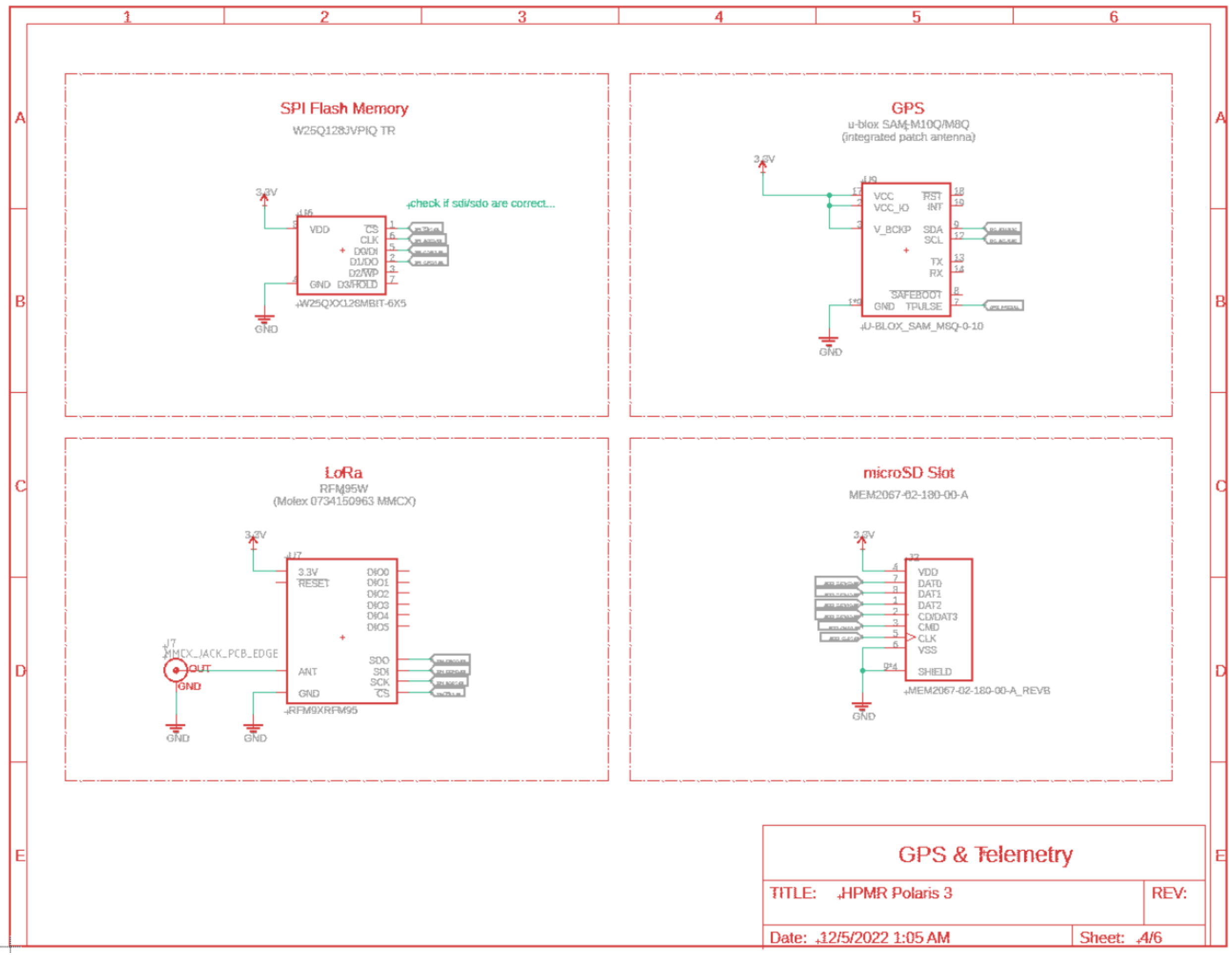
TITLE: HPMR Polaris 3	REV:
Date: 12/5/2022 1:05 AM	Sheet: 1/6

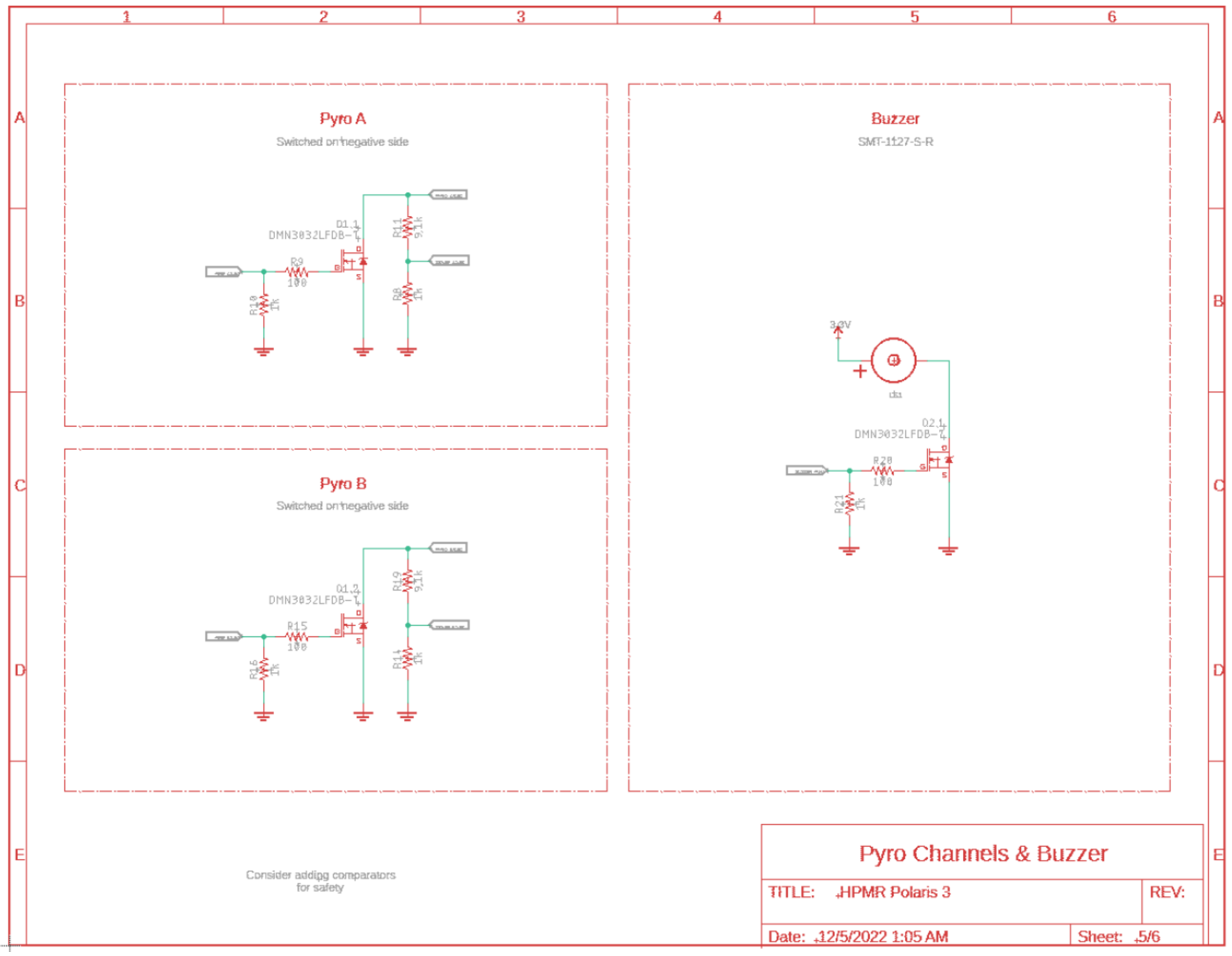


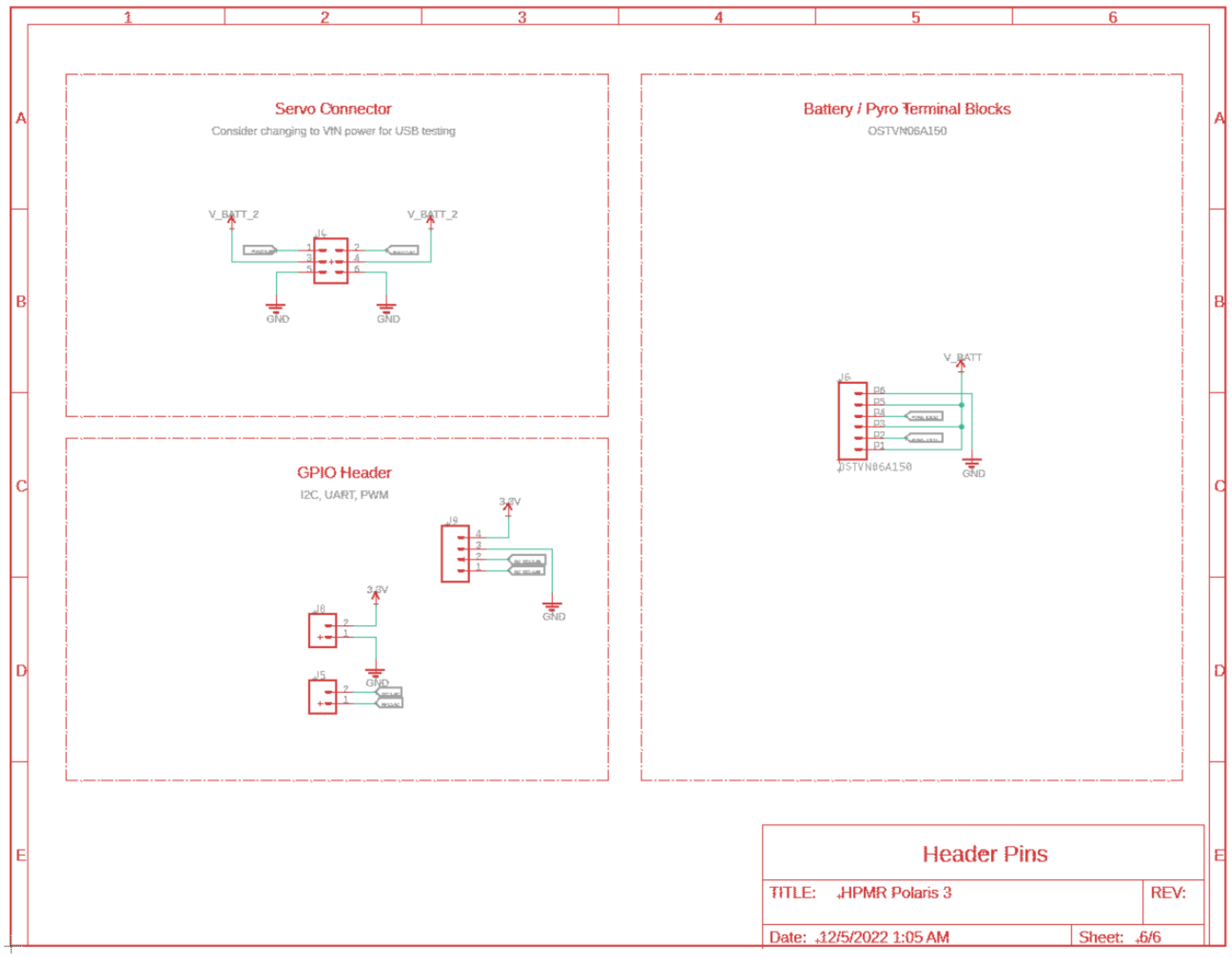
Fuse
Change this to actually make sense
V_USB_2
VIN
F1
6V/2A
Power Source Select
V_BATT_2
VIN
D3
3A/10V/280mV
V_USB
V_USB_2
D4
3A/10V/280mV
Solder jumper to bypass BEC
V_BATT
V_BATT_2
JP6
BECs
Only connect one BEC at a time.
V_BATT
V_BATT_2
PS1
VIN
VOUT
GIN
GOUT
DIATONE_MAMBA_MICRO_BEC
GND
GND
LEDs
3.3V
R2
100
R5
100
R1
1k
D2
BLUE
D5
RED
D1
RED
GND
GND
GND
Maybe consider making R1 smaller
Voltage Measurement
V_BATT
VIN
R6
9.1k
R18
1k
R3
1k
R17
1k
Voltage Regulation
VIN
3.3V
U1
VIN
VOUT
EN
GND
ADJ
P4
P5
P1
P2
P3
AP7381C-33Y5
C3
10uF
C2
0.22uF
C1
0.1uF
C4
10uF
C5
1.0uF
C6
0.22uF
GND
Output voltage: 3.3V
Max input voltage: 6V
Min input voltage: 3.7V
Without a BEC, maximum input voltage is 6V (limited by U1)
With a BEC, the battery voltage goes straight to the BEC, then through the diodes.
This means the diodes don't need to handle LiPo voltage.
Power and LEDs
TITLE: HPMR Polaris 3
REV:
Date: 12/5/2022 1:05 AM
Sheet: 2/6
1
2
3
4
5
6
A
B
C
D
E



Precision IMU
ICM-42688-P
All IMU bypass capacitors: X7R, +/- 10%
3.3V
C7 0.1uF
C8 2.2uF
C9 10nF
U3
VDD
VDDIO
CS/VDDIO
INT2/FSYNC/CLKIN
SDO/AD0
INT1/INT
GND
SCL/SCLK
SDA/SDIO/SDI
RESV
ICM-42688-P
GND
AD0 pulled low
I2C address = 0x68
Barometer
MS5611
Protocol Select (PS) pulled high for I2C
U5
VDD
PS
GND
SCLK
SDI/SDA
SDO
CSB
C10 0.1uF
MS5611-01BA03
In I2C mode, CSB can be connected to EITHER VDD or GND. It determines the last bit of the I2C address
NOTE I2C ADDRESS
High-G Accelerometer
ADXL375
U6
VS
VDDIO
RES
GND
CS
SCL/SCK
SDA/SDI/SDIO
SDO/ALT_ADDR
INT2
INT1
C13 1uF
C14 0.1uF
C15 0.1uF
ADXL375
ALT_ADDR pulled low
I2C address = 0x53
Magnetometer
MMC5983MA
U4
VDDIO
VDD
CAP
NC
GND
SDA/SDI
SCL/SCK
SDO
SPI_CS
INT
C12 1.0uF
C11 10uF
MMC5983MA_QFN16
Sensors
TITLE: HPMR Polaris 3
REV:
Date: 12/5/2022 1:05 AM
Sheet: 3/6
ICM-42688-P is an LGA device, meaning its pads don't wrap up onto the sides like traditional QFN packages. Initial tests with stencil & solder paste showed incomplete connections on all pins so it is recommended to manually "reball" the pads and solder with hot air, after the rest of the components have been done in the oven.


SPI Flash Memory
W25Q128JVPIQ TR
3.3V
U6
VDD
CS
CLK
D0/DI
D1/DO
D2/WP
GND
D3/HOLD
GND
W25QXX128MBIT-6X5
check if sdi/sdo are correct...
GPS
u-blox SAM-M10Q/M8Q
(integrated patch antenna)
3.3V
U9
VCC
VCC_IO
V_BCKP
RST
INT
SDA
SCL
TX
RX
SAFEBOOT
GND
TPULSE
GND
U-BLOX_SAM_MSQ-0-10
LoRa
RFM95W
(Molex 0734150963 MMCX)
3.3V
U7
3.3V
RESET
DIO0
DIO1
DIO2
DIO3
DIO4
DIO5
J7
MMCX_JACK_PCB_EDGE
OUT
GND
ANT
GND
SDO
SDI
SCK
CS
GND
GND
RFM9XRFM95
microSD Slot
MEM2067-02-180-00-A
3.3V
J2
VDD
DAT0
DAT1
DAT2
CD/DAT3
CMD
CLK
VSS
SHIELD
GND
MEM2067-02-180-00-A_REVB
GPS & Telemetry
TITLE: HPMR Polaris 3
REV:
Date: 12/5/2022 1:05 AM
Sheet: 4/6


1
2
3
4
5
6
A
B
C
D
E
Pyro A
Switched on negative side
D1.1
DMN3032LFDB-7
R9
100
R10
1k
R11
9.1k
R8
1k
Pyro B
Switched on negative side
D1.2
DMN3032LFDB-7
R15
100
R16
1k
R19
9.1k
R14
1k
Buzzer
SMT-1127-S-R
3.3V
+
Q2.1
DMN3032LFDB-7
R20
100
R21
1k
Consider adding comparators
for safety
Pyro Channels & Buzzer
TITLE: HPMR Polaris 3
REV:
Date: 12/5/2022 1:05 AM
Sheet: 5/6



1
2
3
4
5
6
A
B
C
D
E
Servo Connector
Consider changing to VIN power for USB testing
V_BATT_2
V_BATT_2
J4
GND
GND
Battery / Pyro Terminal Blocks
OSTVN06A150
V_BATT
J6
P6
P5
P4
P3
P2
P1
OSTVN06A150
GND
GPIO Header
I2C, UART, PWM
3.3V
J9
GND
3.3V
J8
GND
J5
Header Pins
TITLE: HPMR Polaris 3
REV:
Date: 12/5/2022 1:05 AM
Sheet: 6/6

Appendix E: MATLAB Code for Flight Computer

```
clear variables; close all; clc;


data = readmatrix("C:.........shared folder\FDA\021823 Flight Data\Cropped
Polaris Data (Without printout).csv");


pb = 101325; % static pressure (pressure at sea level) [Pa]
Tb = 288.15; % standard temperature (temperature at sea level [K]
Lb = -0.0065; % standard temperature lapse rate [K/m]
hb = 0; % height at bottom of atmospheric layer [m]
R = 8.31432; % universal gas constant [N*m/mol*K]
g0 = 9.80665; % gravitational acceleration constant [m/s2]
M = 0.0289644; % molar mass of Earth's air [kg/mol]


std_dev_baro = .1;    %0.1% guess at baro stv
std_dev_accell = .25; %guess at accell stv




R_me_covariance          = [std_dev_baro^2];            % Measurement error
covariance (2x2 matrix)
P_ee_covariance     = eye(2);


C = [1,0];


i = 1;
n = 0;
t_ref = 0;


min_time_before_launch_detect = 1; %seconds reqired since turned on before it can
start running
time_before_launch_detect_met = 0;


gs_required_for_launch = 1.5;   %Accel required for launch in G's
accell_required_for_launch = gs_required_for_launch * 9.81; %m/sec^2
accel_of_launch_detected = 0;
```

```
gs_required_to_switch_to_regular_accell = 11;   %switches to small accell when
less than this value
accell_required_for_switching_to_regular_accell =
gs_required_to_switch_to_regular_accell * 9.81;   %m/sec^2


time_max_boost = 5;% sec
minium_negative_accell_to_detect_burnout = -0.05*g0; %m/sec^2
expected_motor_burn_time = .25; %seconds of expected motor burn


state_store = []; %storage
final_data_store = [];




decrease_vel_at_burnout_req_met = 0;
accel_burnout_req = 0;
time_req_burn_met = 0;


timeout_apogee = 0;
time_of_timeout_apogee = 15; %seconds, time of coast that is deemed acceptable
before realizing something is wrong


time_of_timeout_main = 60; %seconds, time of main that is deemed acceptable
before realizing something is wrong
timeout_main = 0;


Count_Decent = 0;
fire_charge = 0;
last_non_zero_deltah = 5;
apogee_value_detected = 0;  %ft
altitude_charge_fired_at = 0; %ft
time_of_first_decent = 0;   %s


time_req_after_drouge = 0;
time_delay_for_ejection_gases = 1.5; %sec delay
deploy_main_req = 0;
main_deploy_alt = 152.4; %meters agl


maximum_alt_for_landed_state = 30.48; %assume it must be below 75 meters from
initial alt
near_ground_alt = 0;
```

```
no_change_alt_detected = 0;


time_of_timeout_ground = 120; %seconds, time of ground that is deemed acceptable
before realizing something is wrong
timeout_ground = 0;
ground_counter = 0;


time_of_coast_starting = 0;


while(i <= length(data(:,1))-1)
   curr_data = get_curr_data(i, data);
   time = curr_data(1);
   press = curr_data(2);
   temp = curr_data(3);
   accel_pre = curr_data(4)*g0;
   accel_lar = curr_data(4)*g0;
   dt = curr_data(6);


    if(i == 1)
        start_alt = hb + (Tb/Lb) * ((press/pb).^(-R*Lb/(g0*M)) - 1);
        t_ref = time;
    end
    h_data_ASL = hb + (Tb/Lb) * ((press/pb).^(-R*Lb/(g0*M)) - 1);
    time = time - t_ref;

    % using first ASL to convert all to AGL
    curr_alt = h_data_ASL - start_alt;
```

For testing only use precision acceleromenter right now

```
        %call all accel data from beefy acccell
%     if(accel_lar <=  accell_required_for_switching_to_regular_accell)
%         curr_accel = accel_pre;             %start pulling from the regular
accelerometer
%     else
%         curr_accel = accel_lar;
%     end


    curr_accel = accel_pre;
    if(n == 0) %use the read values before launch, otherwise use old x-hat
        x_hat = [curr_alt; 0];
        curr_vel = 0;
        curr_vel_Baro_der = 0;
    else
        %x_hat is stored in the system, it will use the prior instance as
        %the value
        curr_vel = state_store(i-1,7) + curr_accel*dt;
        curr_vel_Baro_der = (curr_alt - state_store(i-1,6))/dt;
    end


    F_k1 = [1, dt;
            0, 1];
    G1_k1 = [0.5*dt*dt; dt];
    Q_pe_covariance = G1_k1*G1_k1'*(std_dev_accell^2);


    zk = [curr_alt];


    x_minus = F_k1 * x_hat +  G1_k1*curr_accel;
     % always read new pressure and accell measurements
     P_minus = F_k1*P_ee_covariance*F_k1' + Q_pe_covariance;


     y_k = zk - C*x_minus;

     S_k = C*P_minus*C' + R_me_covariance';


    L_Kalman_gain  = P_minus*C'*(inv(S_k));


    x_hat = x_minus + L_Kalman_gain*y_k;
    P_ee_covariance            = (eye(2) - L_Kalman_gain * C) * P_minus;


    kh_alt = x_hat(1);
    kh_vel = x_hat(2);
    kh_accel = curr_accel;


    curr_state = [time, press, temp, accel_pre, accel_lar, curr_alt, curr_vel,
curr_accel, kh_alt, kh_vel, kh_accel, curr_vel_Baro_der];
    state_store(i,:) = curr_state;
    if(n > 0 && n < 3) %only use filterd data from states 1 to deployment of
drouge.
        curr_alt = kh_alt;
        curr_vel = kh_vel;
        curr_accel = kh_accel;
    else
        %dont use the filtered values for pre-launch and after apogee
    end


%%State logic now values are calculated


    if(n == 0) %On Pad Armed
        if(time >= min_time_before_launch_detect)
            time_before_launch_detect_met = 1;
        end
        if(curr_accel >= accell_required_for_launch)
            accel_of_launch_detected = 1;
        end


        if(accel_of_launch_detected == 1 && time_before_launch_detect_met == 1)
            %store the initial time of launch
            time_of_launch_detect = time
            n = 1;
            %launch is now detected
        end

    elseif(n == 1) %boost phase
        time_since_state_change = time - time_of_launch_detect;
```

Currently using kf stuff for this but we could change that

```
        last_vel = state_store(i-1,10);
        curr_vel = state_store(i,10);
        if(curr_accel <= 0)
            test = 1;
        end


        if(curr_vel <= last_vel)
            decrease_vel_at_burnout_req_met = 1;
            time_of_decrease_vel_at_burnout_req_met = time;
        end

        if(curr_accel <= minium_negative_accell_to_detect_burnout)
            accel_burnout_req = 1;
            time_of_accel_burnout_req = time;
        end

        if(time_since_state_change >= expected_motor_burn_time)
            time_req_burn_met = 1;
            time_of_time_req_burn_met = time;
        end

        if(decrease_vel_at_burnout_req_met == 1 && accel_burnout_req == 1 &&
time_req_burn_met == 1)
            %burnout is confirmed, enter coast phase
            time_of_coast_starting = time
            n = 2;
        end

        if(time_since_state_change > time_max_boost) %if state lasts longer than
5 seconds, throw error
            %ERROR DETECTED
            n = 10;
            disp('Timeout error in boost phase')
        end

         %If the system detects a sharp change in acceleration and velocity
                %then we move out of this state and only pass on the last state
                %data (pos, vel, accel) onto the next state as we start the
kalman
                %filter.




    elseif(n == 2) %Upwards Coast, check for apogee
        time_since_state_change = time - time_of_coast_starting;


        prior_alt = state_store(i-1,9);
        dh = curr_vel;

        if(dh == 0)
            dh = last_non_zero_deltah;
        else
            last_non_zero_deltah = dh;
            dh = last_non_zero_deltah;
        end

        if (curr_alt < 30)
            %nothing happens because altitude is below 30 meters
        elseif (dh > 0.5)   %velocity checkout
```

```
            %Nothing happens if velocity is too high in the acent phase
        else
            if(dh < 0)
                Count_Decent = Count_Decent + 1;
                Count_Decent
                disp(['Time of neg dh = ', num2str(time)])
                i;


                if(Count_Decent == 1)
                    time_of_first_decent = time;
                end
            end

            if(Count_Decent > 5 && last_non_zero_deltah < 5)
                Count_Decent;
                fire_charge = 1
                apogee_value_detected_at = state_store(i-6,9)
                altitude_charge_fired_at = curr_alt
                time_of_apogee_start = state_store(i-6,1)
                time_of_apogee_ref_launch = time_of_apogee_start -
time_of_launch_detect
                time_of_fire = time
                n = 3
                disp('reached apogee');
            end
        end


        %as a backup, if the time-exceeds some standard amount of time,
        %fire the drouge to save the rocket
        if(time_since_state_change >= time_of_timeout_apogee)
            timeout_apogee = 1;
            disp('timeout reached during coast, enter emergency deployment
charge')
            time_of_fire = time
            fire_charge = 1
            altitude_charge_fired_at = curr_alt
            n = 3
        end


    elseif(n == 3) %Apogee, %now looking for main altitude
        %there is a pressure spike due to ejecting drouge, therefore we
        %want to put a time-delay on checking for main-deployment alt

        %at this point we can assume that we are decending and therefore we
        %don't have to do as much logic as we did for the drouge.
        time_since_state_change = time - time_of_fire;


        if(time_since_state_change >= time_delay_for_ejection_gases)
            time_req_after_drouge = 1;
        end


        if(curr_alt <= main_deploy_alt)
            deploy_main_req = 1;
        end


        if(time_req_after_drouge == 1 && deploy_main_req == 1)
            disp('deploy main')
            time_of_main_deploy = time
            fire_main_charge = 1
            alt_of_main_deploy = curr_alt;
            n = 4
        end


        %as a backup, if the time-exceeds some standard amount of time,
        %fire the main to save the rocket
        if(time_since_state_change >= time_of_timeout_main)
            timeout_main = 1;
            disp('timeout reached during drouge, enter emergency deployment
charge of main')
            time_of_main_deploy = time
            fire_main_charge = 1
            alt_of_main_deploy = curr_alt
            n = 4
        end

    elseif(n == 4) %Ejection Completed, looking for the ground

        %use barometer
        curr_alt = state_store(i,6);


        time_since_state_change = time - time_of_main_deploy;
        time;


        prior_alt = state_store(i-1,6);
        if(curr_alt <= maximum_alt_for_landed_state)
            near_ground_alt = 1;


            if(abs(curr_alt - prior_alt) <= .1)
                abs(curr_alt - prior_alt)/abs(prior_alt);
                no_change_alt_detected = 1;
                ground_counter = ground_counter+1;
            else
                ground_counter = 0;
            end
        end


        if(near_ground_alt == 1 && no_change_alt_detected == 1 && ground_counter
>= 10)
            disp('ground detected')
            time_of_ground_impact = state_store(i-1,1)
            altitude_of_ground_detect = curr_alt
            n = 5
        end

        %as a backup, if the time-exceeds some standard amount of time,
        %enter landed mode to save the data
        if(time_since_state_change >= time_of_timeout_ground)
            timeout_ground = 1;
            disp('timeout reached during ground, enter storage of data')
            time_of_ground_impact = time
            altitude_of_ground_detect = curr_alt
            n = 5
        end

    elseif(n == 5)% store data and cut program
        disp('data stored')
            final_data_store = state_store;
            i = length(data(:,1))+1 % break the script here for testng
    else
                %ERROR DETECTED
    end
          i = i +1;
end
%
figure(1)
hold on;
grid on;
title('Barometer Alt vs KF-Filtered Altitude');
xlabel('Time (s)');
ylabel('Altitude (m AGL)');
```

```matlab
plot(state_store(:,1), state_store(:,6))
plot(state_store(:,1), state_store(:,9))
xline(time_of_launch_detect)
xline(time_of_coast_starting, 'Color', 'g')
xline(time_of_apogee_start)
xline(time_of_fire)
xline(time_of_main_deploy)
%xline(time_of_ground_impact)
legend('Barometer Alt', 'KF-Filtered Altitude');
hold off;
%
figure(2)
hold on;
grid on;
title('Calculated accel vs KF Acceleration');
xlabel('Time (s)');
ylabel('Acceleration (m/s*2)');
plot(state_store(:,1), state_store(:,8))
plot(state_store(:,1), state_store(:,11))
legend('Calculated Acceleration', 'KF Acceleration');
hold off;
%



figure(3)
hold on;
grid on;
title('the three states');
xlabel('time')
plot(state_store(:,1), state_store(:,6))    %cur alt
plot(state_store(:,1), state_store(:,7))    %cur vel
plot(state_store(:,1), state_store(:,8))    %cur accel
plot(state_store(:,1), state_store(:,9))    %kh alt
plot(state_store(:,1), state_store(:,10))  %kh vel
plot(state_store(:,1), state_store(:,11))   %kh accel
plot(state_store(:,1), state_store(:,12))   %baro-der-vel
legend('cur alt', 'cur vel', 'cur acccel', 'kh alt', 'kh vel', 'kh accel', 'baro-
der-vel');
xline(time_of_decrease_vel_at_burnout_req_met);
xline(time_of_accel_burnout_req);
xline(time_of_time_req_burn_met);
xline(time_of_coast_starting);
hold off;


%
```

```
% figure(5)
% hold on;
% grid on;
% title('Trace of P (The Covariance');
% xlabel('Time (s)');
% ylabel('P (Covariance, aka: Level of uncertancy in state)');
% plot(time, trace_P_store)
% hold off;




% figure(4)
% hold on
% grid on
% title('state plot')
% line([0,time_of_launch_detect], [0,0])
% line([time_of_launch_detect, time_of_coast_starting], [1,1])
% line([time_of_coast_starting, time_of_apogee_start], [2,2])
% line([time_of_apogee_start, time_of_fire], [3,3])
% line([time_of_fire, time_of_main_deploy], [4,4])
% line([time_of_main_deploy, time_of_ground_impact], [5,5])
% hold off
```

```
function curr = get_curr_data(i, data)
    time = (data(i, 1))/1000;


    dt = data(i,2);


    temp = data(i,10);
    press = data(i,8)*100;

    accel_pre = data(i,11); %m/sec2's
    accel_lar = data(i,11); %data(i,17); %for the sake of testing code, the
precise and the large accel are the same dataset

   % pause(0.01); %set time delay for system refresh rate, this is the dt
   % so make sure this matches real life

    curr = [time; press; temp; accel_pre; accel_lar; dt];
end
```