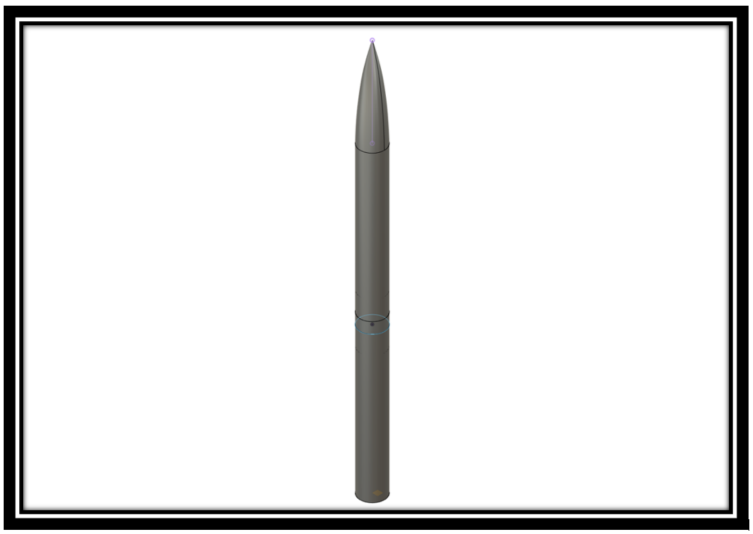
Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Fall 2022



Team 4
TVC GIMBAL ROCKET

Demoria Sherman
Milad Noori
Murphy Balsomi
Sergio Guerrero

Revision History

Revision	Date	Author(s)	Description
0.1	9.14.2022	MN	Document layout created and shared
0.2	9.14.2022	MN	Diagrams uploaded for each section
0.3	9.14.2022	MN, SG	Complete section 1; Introduction
0.4	10.16.2022	MN	Complete section 6; Communications Layer
0.5	10.16.2022	SG	Complete section 4; Control Systems Layer
0.6	10.20.2022	DS	Complete section 3; Flight Computer Layer
0.7	10.20.2022	MB	Complete section 5; Rocket Recovery Layer
0.8	10.21.2022	MN	Section 2; System Overview, Transferred from ADS
0.9	10.21.2022	MN,SG,DS,MB	Initial Review and feedback
1.0	10.21.2022	SG	Final Review and Submission
1.1	12.15.2022	MN	Final Review and Edit

CONTENTS

List of Figures

List of Tables

1 Introduction

TVC Gimbal Mount Rocket is designed and developed with two main requirements in mind. Be stabilized through Gimbal Mount system and be reusable through rocket recovery system, Parachute ejection system.

The ultimate goal of this model rocket is to return to the ground safely, if done successfully this would indicate a reusable rocket that can be launched and returned numerous times which will be efficient and sustainable for the industry.

The ability of re-usability opens up many doors for this rocket. If engineered properly it can be broken into sub-parts and distributed to younger aspiring engineers to get hands-on experience in constructing and testing their rockets.

Also, it can be used in classrooms across the nation as a demonstration to students of the heights that modern technologies can achieve in todayâs world. Lastly, it can be scaled up along other available technologies to perform a more complex task such as taking people or satellites to low earth orbit.

TVC Gimbal Rocket is developed in four layers of architecture:

Layer 1: Flight Computer

Layer 2: Control System (Gimbal Mount)

Layer 3: Rocket Recovery (Parachute Ejection)

Layer 4: Communications

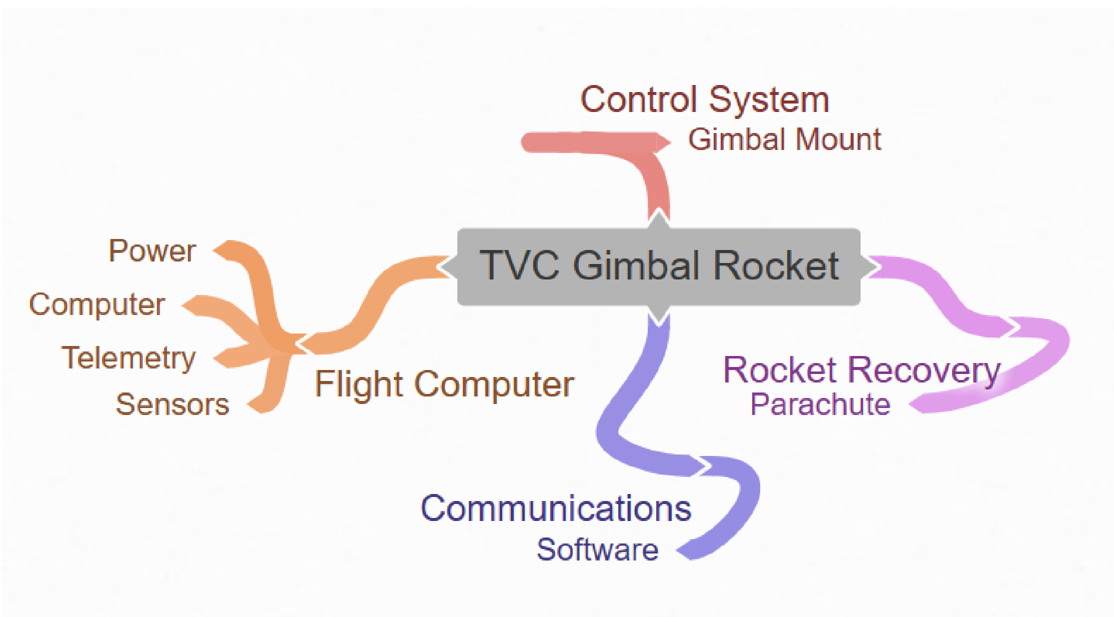


Figure 1: A simple architectural view of the system

2 System Overview

Here you can see a general layout of the data flow of the entire system with subsystems. As you can see in figure 3, data will be communicated back and forth using an I2C connection protocol to the micro controller.

Telemetry will be connected to the micro controller using the same communication protocol as well as a wireless communication to the ground telemetry module. The data flow coming from the two sensors on board will be used in software to be converted from raw data to data necessary to control the servoâs connected to the engine of the rocket.

The data from sensors Will also be used to send a signal to the parachute ejection system when at the correct altitude. Data and status will be sent wireless via the telemetry modules to a ground computer. through out the flight the rocket will be showing which state itâs in via data being sent from the micro controller to the peripheral devices connected such as LEDâs Buzzers and buttons. This gives a basic general description of the data flows being sent through the computer during flight.

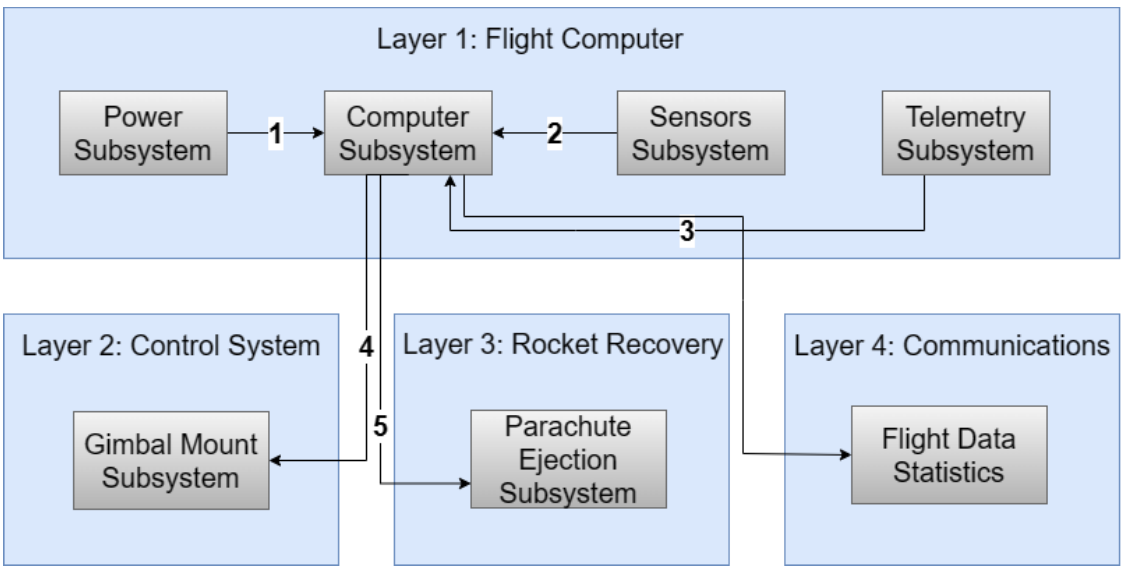


Figure 2: System Architecture

3 LAYER 1 (FLIGHT COMPUTER) SUBSYSTEMS

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be omitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project.

3.1 LAYER HARDWARE

The hardware components involved in this project consist of a teensy 4.1 as our micro-controller. It will be controlling the subsystems through gpio, rx/tx, and I2C protocol.

3.2 LAYER OPERATING SYSTEM

The teensy 4.1 uses a ARM Cortex-M7 processor, and capable of running on linux, windows, and mac.

3.3 LAYER SOFTWARE DEPENDENCIES

The teensy 4.1 utilizes the IDE known as teensyduino to operate on any desired os. from there you can download custom or ready made libraries or code them yourself.

3.4 LAYER DETAILS

The high level purpose is to control the sensors and peripherals of the entire flight computer.

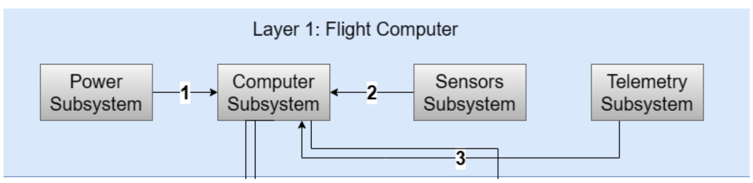


Figure 3: Example subsystem description diagram

3.4.1 SUBSYSTEM HARDWARE

This system only uses the subsystems involved in the teensy 4.1 created by PJRC.

3.4.2 SUBSYSTEM OPERATING SYSTEM

No operating system required specifically.

3.4.3 SUBSYSTEM SOFTWARE DEPENDENCIES

The teensy 4.1 utilizes the IDE known as teensyduino to operate on any desired os. from there you can download custom or ready made libraries or code them yourself.

3.4.4 SUBSYSTEM PROGRAMMING LANGUAGES

The programming language for this device is Arduino witch is (C/C++).

3.4.5 SUBSYSTEM DATA STRUCTURES

The data is structure by using a state space machine that take data from the sensors and while transmitting them through radio telemetry in real time.

3.4.6 SUBSYSTEM DATA PROCESSING

Only the state space machine was utilized for the architecture of the main flight computer software.

4 Layer 2 (Control System Layer) Subsystems

The Control System Layer consists of all subsystems that work to control the rocket on its flight. The subsystems have been combined to on subsystem that better depicts its purpose in this project. There are four main parts to this system that all work together to control the rocket, those parts being the Gimbal Mount, Servos, Engine, and the Battery. Each of these will be linked together in some way either it being physically or through software to assure control on the rocket.

4.1 Layer Hardware

As hardware we have the physical gimbal mount, this mount was 3D printed and is used to hold both servos as well as the engine. The servos are to be able to move freely, in their respective direction. Each Servo will be connected to the engine to move the engine propulsion in a circular motion. There is also the battery which will be used to power the entire system. This battery will be connected to a Buck Converter which will step down the voltage to 5V which is what is needed for all components on the board.

4.2 Layer Operating System

We are using the Arduino Software to program the Teensy 4.1 Micro-Controller.

4.3 Layer Software Dependencies

Libraries Include - Adafruit BNO08X used for the IMU board. Adafruit Sensor used for the Barmoeter sensor on the board. Adafruit BNO055 used for thr BNO055 IMU board on the flight computer prototype Servo Library to allow to create servo objects and send various PWN signals and angle signals to each respective Servo. Also used for the Fan.

4.4 Subsystem 1

This subsystem consists of the physical gimbal mount. This gimbal mount is a 3D printed part which is inserted at the bottom of the rocket. It is designed to hold 2 servos, one for the X axis and one for the Y axis. By connecting each servos to the engine piece, it will give the engine propulsion a range to move in both X and Y axis.

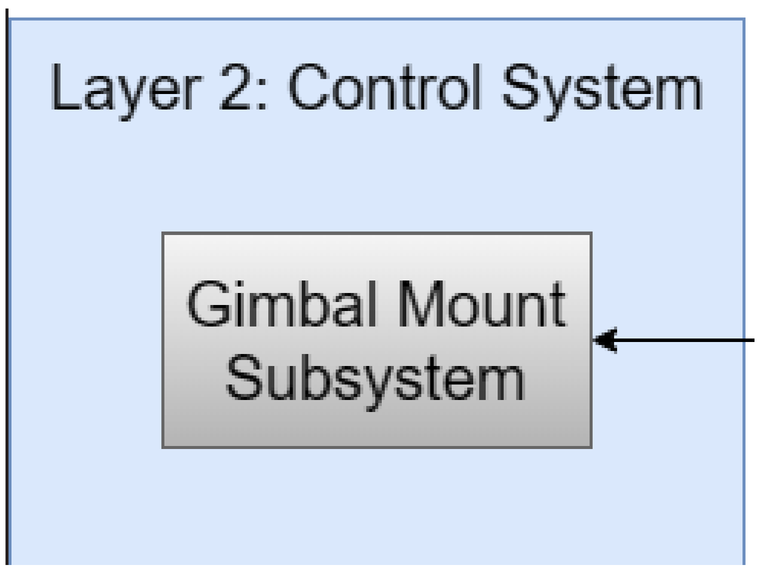


Figure 4: Example subsystem description diagram

4.4.1 Subsystem Hardware

Physical 3D Printed Gimbal Mount, this will hold all components to control the system such as the Servos Motor from Miuzei and the Estes Pro Series 2 F15-8 Engine.

4.4.2 Subsystem Operating System

There is no Operating System for this subsystem. However we are using the Arduino IDE to flash and program the Teensy 4.1 micro controller.

4.4.3 Subsystem Software Dependencies

Fusion 360 for 3D design. The Servos use a Servo.h library from Arduino. This library is used to create a Servo Object for each Servo and allow us to control the angle and take data from the Servo.

4.4.4 Subsystem Programming Languages

Programming Language used is C/C++

4.4.5 Subsystem Data Structures

Communication and Data are being passed to the Teensy 4.1 through I2C which allow both servos to send and receive data at the same time.

4.4.6 Subsystem Data Processing

An Algorithm we used to program these Servos to work is PID which is Sort for Proportional, Integral, Derivative. The Proportional side is used to reach the end point from the start point in a quicker more exponential way. The Integral part is used to process the overshooting that may occur when moving from the start point to the end point. And the Derivative part is used to to smooth and diminish the approach to the end point.

5 Layer 3 (Rocket Recovery) Subsystems

The parachute that will use in this project is 24" red Rip-stop nylon parachute with 6 sides and 6 braided nylon shroud lines.

5.1 Layer Hardware

Model: 29101; Material: Plastic Durable and colorful: Red; Weight: 11 g (0.39 oz); Parachute Shape: Octagon; Descent Rate (Cd): varies Shroud Line Length: 32" (81.28 cm); Parachute Area: 69.12 inches; Manufactured by: Dino Chutes, Inc

5.2 Layer Operating System

No operating system involved

5.3 Layer Software Dependencies

Computer Subsystem Software from Level 1

5.4 Subsystem 1

The rocket will deploy the parachute after reaching a minimum of 25 feet above the ground by sending a signal from the teensy to the small charge .

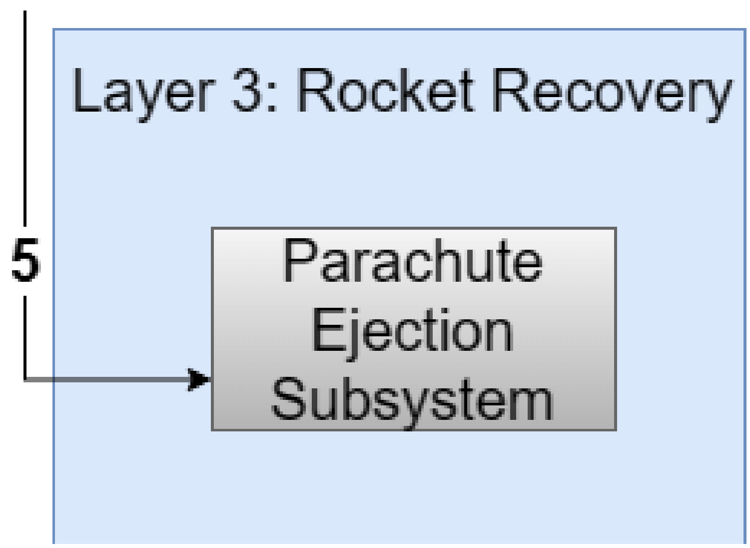


Figure 5: Example subsystem description diagram

5.4.1 Subsystem Hardware

5V Battery

5.4.2 Subsystem Operating System

No operating systems required by the subsystem.

5.4.3 Subsystem Software Dependencies

Computer Subsystem Software from Level 1

5.4.4 Subsystem Programming Languages

Programming Language used is C/C++

5.4.5 Subsystem Data Structures

There are no data structures

5.4.6 Subsystem Data Processing

There is no Data Processing.

6 Layer 4 (Communications) Subsystems

For this layer we are currently using Arduino IDE and visualize the data statistics manually. The data to the application will be fetching to this layer from Layer 1, Flight Computer Layer. The focus currently is to focus on achieving product goals. The communication layer can be extended to another project to build an application that would provide better visualization using data from Telemetry. The current version saves data in the memory card that can be used by the user for further analysis. Developing a ground system application can enable user to view the telemetry data while the rocket is in flight. A good example to follow for project extension is Mission Planner desktop application by Ardupilot.

6.1 Layer Hardware

The application depends on the Telemetry data or the data saved on memory card attached to Teensy.

6.2 Layer Operating System

The Aruino application available for Windows, Linux, and macOS. Chromebook users can access the Arduino Web Editor.

6.3 Layer Software Dependencies

None

6.4 Subsystem 1

The application is going to be the only subsystem for the layer and is connected to two subsystems of layer 1; Flight Computer and the Telemetry. The visibility aspect of User Interface is key for Data Visualization. Therefore, the application is planned to have a window with speedometer on the left, that will show the speed of the rocket and window on the right, to show the status of the rocket launch stages; launch, apogee height, and recovery (ground) status. For now, we have decided to use launch pad for launch and calibrate manually.

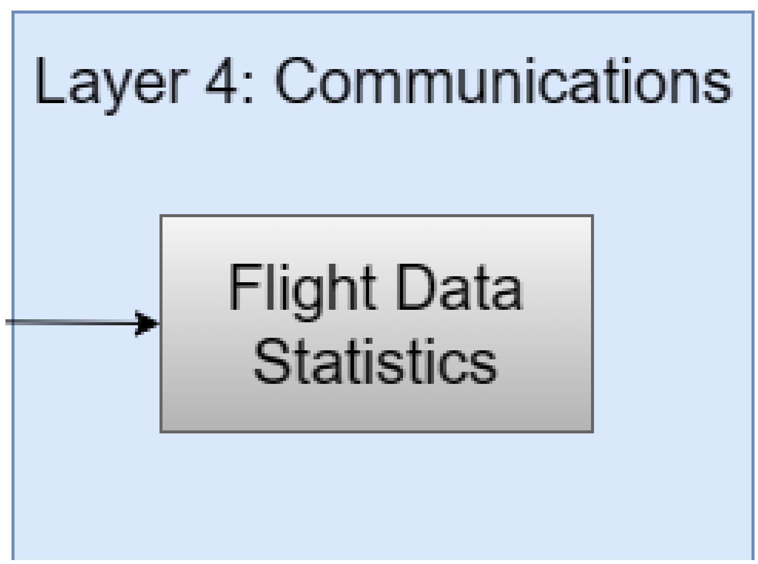


Figure 6: Example subsystem description diagram

6.4.1 Subsystem Hardware

Telemetry and Memory card are two main hardware components besides the Flight Computer Layer hardware components, such as IMU, and Micro Controller taht helps with data transmission and backup.

6.4.2 Subsystem Operating System

Windows, Linux, macOS, and Chrome OS

6.4.3 Subsystem Software Dependencies

The data is main dependency for the subsystem as it need the fetched data either through telemetry or memory card to visualize the launch statistics.

6.4.4 Subsystem Programming Languages

C++

6.4.5 Subsystem Data Structures

The data is fetched to the subsystem through Telemetry or the memory card in micro controller in comma delimited values.

6.4.6 Subsystem Data Processing

The data is currently processed by Ardunio and shown in the IDE window and saved in the memory card. In the future this layer can include an application similar to ArduPilot mission planner to better visualize the data. For example a Speedometer in left side and status for different stages of the rocket launch on right side.

7 Appendix A

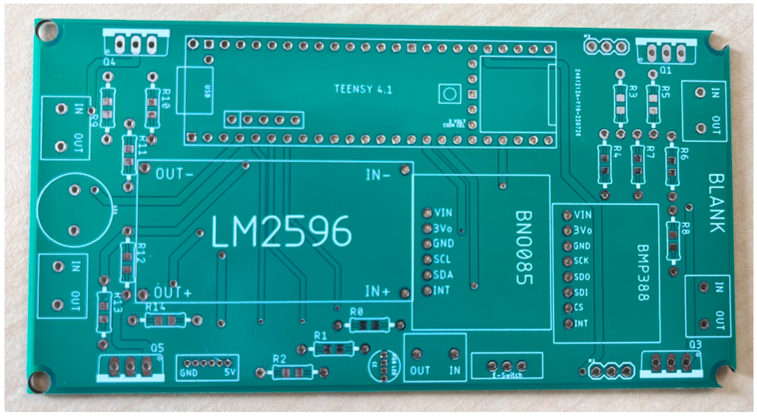


Figure 7: PCB Design Layout

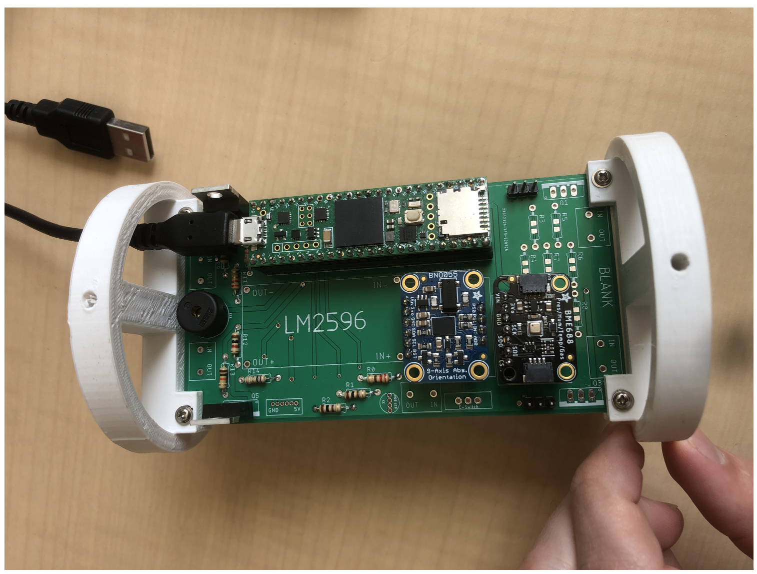


Figure 8: Assembled Flight Computer

REFERENCES