Internship Report on

Design and Analysis of Rocket Motor Static Test Pad

At STAR - Space Technology and Aeronautical Rocketry

Varun Rayamajhi
Avionics Intern
Period: 20^{th} **August -** 19^{th} **September**
Duration: 30 Days

Rocket Motor Static Test Pad

What is a rocket motor static test pad?

Rocket Motor Static Test Pad is a stationary mechanism designed to hold a rocket motor and conduct its static test fires, during which different parameters of its operation are measured and stored by avionics system for analyzing its performance. These parameters include thrust, chamber pressure, exhaust gas temperature, and so on. Following is a picture of rocket motoe static test pad:-

Figure 1: Rocket Motor Static Test Pad

Basic Working Principle of Rocket Motor Static Test Pad

Rocket Motor Static Test Pad consists of two prominent components: Frame/Body and Avionics System.

Body

The body of a rocket motor static test pad is used to hold rocket motor statically during static test fires. Depending on the design, the rocket motor can be mounted horizontally or vertically. As well as that, the body of the test pad includes portability mechanism, if it has any. Additionally, it also includes structure to hold controlled fire extinguisher and avionics bay.

Avionics System

Avionics System is a major part of rocket motor static test pad. It is placed inside avionics bay. Avionics System is like the brain of the static test pad which controls the operation of rocket motor, data acquisition, and safety system. Avionics System consists of all electronic components such as temperature sensor, micro controller, load cell, and so on, which collectively work to conduct static test fires safely and store the acquired data.

Working

The experimental rocket motor is held into the motor mount of the test pad. All the electronics connections are made and verified and the electronics components are put inside the avionics bay. The load cell is mounted onto the central support frame. Igniter and sensors are put into their respective places; for instance, temperature sensor is placed near the combustion chamber if its purpose is to measure chamber temperature. To begin static test fires, a command is sent through micro controller to start ignition system. After ignition, the motor is fired up and the required data is collected by respective components and stored, if necessary, for future analysis. The testing ends automatically when the motor runs out of fuel whereas the data acquisition terminates when micro controller is powered off or by command, if there is any, from micro controller. This is the general working mechanism of rocket motor static test pad, but the working may differ with different types of static test pads. Some static test pads, for example, might have controlled fire extinguishing system while others may not have such system.

Necessity of Rocket Motor Static Test Pad

Space missions are expensive and so are launch vehicles. Launch vehicles are used for performing scientific missions. Such expensive, yet useful vehicles, can't be sent blindly to space for their missions. To ensure that the vehicles perform as intended on their missions, their static testings are inevitable. Static test pads allow the engineers to test rocket motors and analyze their performance parameters for further analyses, which enable them modify the rocket motor or nozzle or even fuel for best possible performance. Likewise, static tests allow engineers to scout for possible problems that may arise during the launch. It allows them to overcome such problems of the rocket motor beforehand. In all, static test pads are required for performance analysis of rocket motors as well as for high efficiency and economical reasons, too.

Softwares Used

Following softwares were used for following purposes:-

Table 1: Softwares and their purposes

S.N.	**Software**	**Purpose**
1.	Fusion 360	CAD Design, Assembly, and Simulation
2.	Proteus	Circuit Design and Simulation
3.	Arduino IDE	Code Development
4.	Eagle EDA	PCB designing
5.	MIT App Inventor	Mobile Application Development for remote controlling with mobile
6.	Processing	Graphical Interface Design for remote controlling with laptop

Components Used

The components used in the static test pad and their purposes are tabulated below:-

Table 2: Mechanical Design components and their purposes

S.N.	Component	Purpose
1.	T-Slot Aluminum Bars	Structural purpose
2.	Fasteners	Connecting gusset plate to the bars
3.	Gusset Plates	Connecting all the bars with each other
4.	Avionics Bay Housing (Casing)	Holding all the electronics
5.	Motor Mount	Holding the rocket motor axially
6.	Ball Transfer Units	Facilitating the axial motion of the motor while it is burning
7.	Nuts	Connecting fasteners and gusset plates with the T-Slot bars.

Table 3: Avionics components and their purposes

S.N.	Component	Purpose
1.	Arduino Uno R3	Microcontroller
2.	Button Type Load Cell	Measuring Thrust
3.	HX711 module	Converting analog signal from load cell to digital signal
4.	SD Card	Data Storage
5.	SD Card Module	Holding SD Card
6.	HC05 module	Creating bluetooth connection between microcontroller and laptop/mobile
7.	LEDs	Visual aid for safety purpose
8.	Push Button	For calibration purpose of load cell, if needed
9.	Buzzer	Auditory perception for safety purpose
10.	2 channel 5V Relay Module	For delayed ignition system and for controlling fire extinguisher
11.	Batteries	Powering Arduino and electricity source for igniter and solenoid
12.	Nichrome Wire	Igniter
13.	Solenoid	For controlled fire extinguisher mechanism
14.	MAX6675 module with k-type thermocouple	Measuring exhaust gas temperature

Architecture

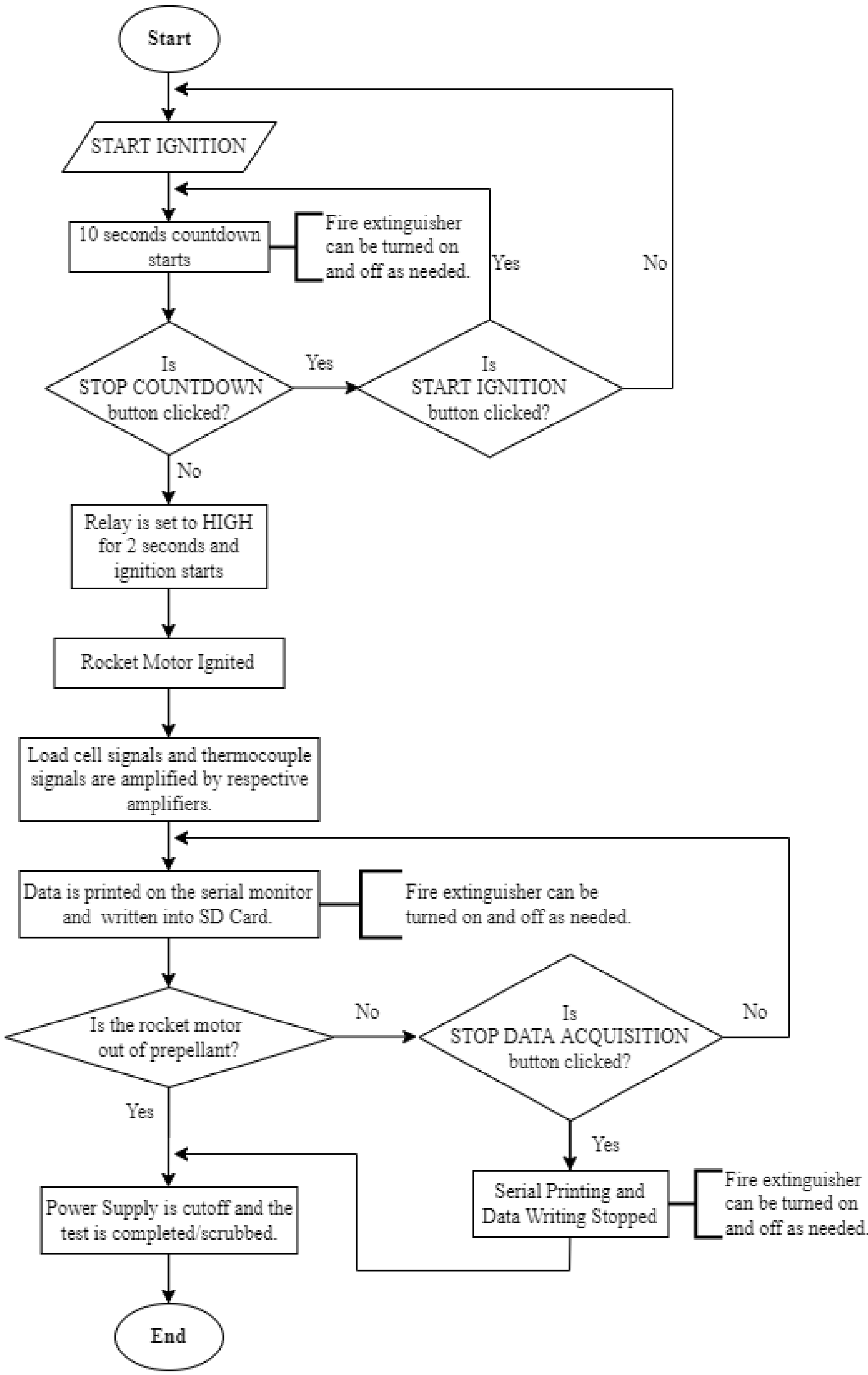


Figure 2: Flowchart of Avionics System

Objectives

The objective of the internship was to design a static test pad for high-powered rocket motor. Following were the projected properties of the static test pad:-

1. Able to acquire and process the requires data
2. Real-time data transmission and reception
3. Data storage for future analysis
4. Good strength/weight ratio
5. Good aesthetics and ergonomics
6. Easy to assemble and dissemble
7. Easy to accommodate all electronics and power system conveniently
8. Prone to minimum errors

Methodology

This report primarily focuses on the avionics of the static test pad.

Avionics

The anatomy of contents discussed in this section is as follows: (1) my approach to solving problem statement, (2) a small account of the components mentioned earlier, and (3) Features of Avionics System.

First thing I did before starting my work was reading the problem statement and finding out what I'm supposed to do. After finding the purpose of the problem statement, I made an algorithm (step-wise procedures) on how I'll be working on the problem statement. Following is the procedure I followed for solving the problem statement:-

1. Research what type of data (parameters) is needed to analyze the performance of rocket motor.
2. Research about necessary sensors and other electronic components according to the parameters needed.
3. Finalize the electronic components.
4. Design circuit for load cell in Proteus.
5. Write sketch for load cell in Arduino IDE.
6. Simulate load cell in Proteus.
7. Repeat 4, 5, and 6 for other components.

8. Design circuit containing all electronic components in Proteus.
9. Write Arduino sketch and simulate circuit in Proteus.
10. If problems arise, work on them. If not, design PCB schematic in Eagle.
11. Calculate total cost of avionics system.

In every step of the aforementioned algorithm, I was always brainstorming about the problems that may arise during motor test and whether the avionics system can help solve those problems. Afterwards, I worked on solving the problems.

After conducting research, I finalized the components mentioned earlier in the report. I used Arduino Uno R3 as a microcontroller. For measuring thrust, our team decided to use button type load cell. For load cell amplifier, I used HX711 module. Regarding temperature sensors, I had many options, but PT100 sensor and k-type thermocouple were the major contenders. I decided to use k-type thermocouple over PT100 because of its wide temperature range (0°C - 1024°C) and a little bit low cost compared to PT100 sensor. For thermocouple amplifier, I had the options of AD595 and MAX6675 amplifiers. I chose MAX6675 amplifier mainly because of its high in-built resolution of 12 bits as compared to 10 bit of AD595 and also because of its cheap cost than AD595. For data storage, I used micro SD Card module. LEDs were used as visual aid for safety purpose. Three LEDs were used: Green, Yellow, and Red. Green LED indicates that the temperature recorded by thermocouple is within the range. Red LED indicates that the temperature is beyond the limit. Continuously glowing Yellow LED indicates that the ignition system hasn't started yet while a blinking yellow LED indicates that the ten seconds countdown for ignition is going on. LOW yellow LED indicates that the motor has been ignited and the test is being carried out. Likewise, I have also used buzzer, which produces sound when the temperature crosses the limit. For remotely controlling ignition system and fire extinguisher, I have used 2 channel 5V relay module. For ignition system, the relay module acts as a one time switch only while it acts as a switch for fire extinguisher for whole duration of rocket motor test. Solenoid is used for controlling the handle of fire extinguisher. 32 AWG Nichrome 60 wire is used as an igniter. For remote controlling, I have used HC05 bluetooth module.

The avionics system has a remote controlled ignition system. Using button controlled ignition system was an ad-hoc, so I thought of making remote controlled ignition system. Ignition system can be initiated by typing command in serial monitor or graphical interface developed using Processing or mobile application. From now on in the report, I refer these three methods of remotely controlling avionics as "THREE methods". If any problem occurs during the countdown, the ignition system can be halted by any of the THREE methods. After solving the problem, ignition system can be started again by same process as earlier. The idea of stopping countdown and halting ignition system was suggested by the shaper Harsh Patel. Below is the code specific to ignition system in my main sketch:-

```
if (Serial.available()){
  Incoming_value = Serial.read();
  if(Incoming_value == 'o' || Incoming_value == 'O' ){
    digitalWrite(fire, HIGH);
  }
  else if(Incoming_value == 'p' || Incoming_value == 'P'){
    digitalWrite(fire, LOW);
  }
  else if (digitalRead(yellow) == HIGH){
    if (Incoming_value == '1'){
      Serial.println("                 COUNTDOWN STARTED ");
      Serial.println(" ");

      for (int i=10; i>=1; i--){
        if (Serial.available()){
           Incoming_value = Serial.read();
        }
        if (Incoming_value != 's' && Incoming_value != 'S') {
          if(Incoming_value == 'o' || Incoming_value == 'O'){ digitalWrite(fire, HIGH); }
          if(Incoming_value == 'p' || Incoming_value == 'P'){digitalWrite(fire, LOW);}
          digitalWrite(yellow, !digitalRead(yellow));
          Serial.print("                   ");
          Serial.print(i);
          Serial.println("                   ");
          delay(1000);//countdown
       }
       if(Incoming_value == 's' || Incoming_value == 'S'){
         Serial.println("IGNITION HALTED: Type '1' or click 'START IGNITION' button for restarting countdown.");
         digitalWrite(yellow, HIGH);
```

Figure 3: Code specific to Ignition System and Fire Extinguisher

The above code also contains commands to control fire extinguisher. Lines near the word "fire" are specific to fire extinguisher.

One problem that may arise during motor test is the collection of false/invalid data when a problem occurs during the test such as motor sliding off the motor mount due to lack of pre-check of motor clamps placement, fire (unwanted), and so on. In such case, data acquisition can be stopped by any of the THREE methods. Actually, it is one of the problems that I thought it might arise during the test. Below is the code specific to stopping data acquisition:-

```
}else if (digitalRead(yellow) == LOW && Incoming_value == '0'){
  val = false;
  Serial.println("DATA ACQUISITION AND TESTING STOPPED");
  while(1){
   if (Serial.available()){
     Incoming_value = Serial.read();
     if (Incoming_value == 'o' || Incoming_value == 'O'){
       digitalWrite(fire, HIGH);
     }
     if (Incoming_value == 'p' || Incoming_value == 'P'){
       digitalWrite(fire, LOW);
     }
   }
  }
 }
```

Figure 4: Code specific to stopping data acquisition

Static test fires are meant to analyze the performance parameters of rocket motor. Thus, data storage is important in static testing. For data storage, I have used SD Card, which stors data in .txt file.

For data analysis, Microsoft Excel can be used. I have used “,” as a delimiter between different columns of data, so the data stored in .txt file can be imported from Excel easily. Thermocouple measures exhaust gas temperature and load cell measures thrust. With information of thrust, exhaust gas temperature, and specifications of the rocket motor, we can calculate different parameters for analyzing the performance of the motor. For example, chamber pressure, total thrust, average thrust, maximum thrust, specific impulse, effective exhaust velocity, burn time, etc. Below is the code specific to data storage:-

```
  Data = SD.open("DataAcquisition.txt", FILE_WRITE);
  Data.seek(EOF);
  if (Data)
  {
    Data.print(elapsedTime);
    Data.print(", ");
    Data.print(thrust, 2);
    Data.print(", ");
    Data.print(temp, 2);
    Data.print("\n");
    Data.close();
    Serial.println("Data Written Successfully");
  }else{
     Serial.println("Couldn't write data");
     Serial.println(" ");
  }
  delay(100);
}
```

Figure 5: Code specific to Data Storage and use of “," as a delimiter

In the Arduino sketch, I’ve used different values (letters and numbers) as commands for Arduino to perform tasks of ignition, fire extinguisher, and data acquisition stoppage accordingly. Remembering different variables might be troublesome and not too practical, so I used Processing software to generate a graphical interface through which commands can be sent by clicking buttons when the laptop’s bluetooth is connected to HC05 bluetooth module. For example, pressing “START IGNITION” button will send letter ‘1’ to Arduino through HC05 module, which means arduino will start ignition as it is programmed to do so only once on receiving ‘1’. Below are pictures of the interface generated by Processing Software and Processing code specific to sending commands to arduino:-

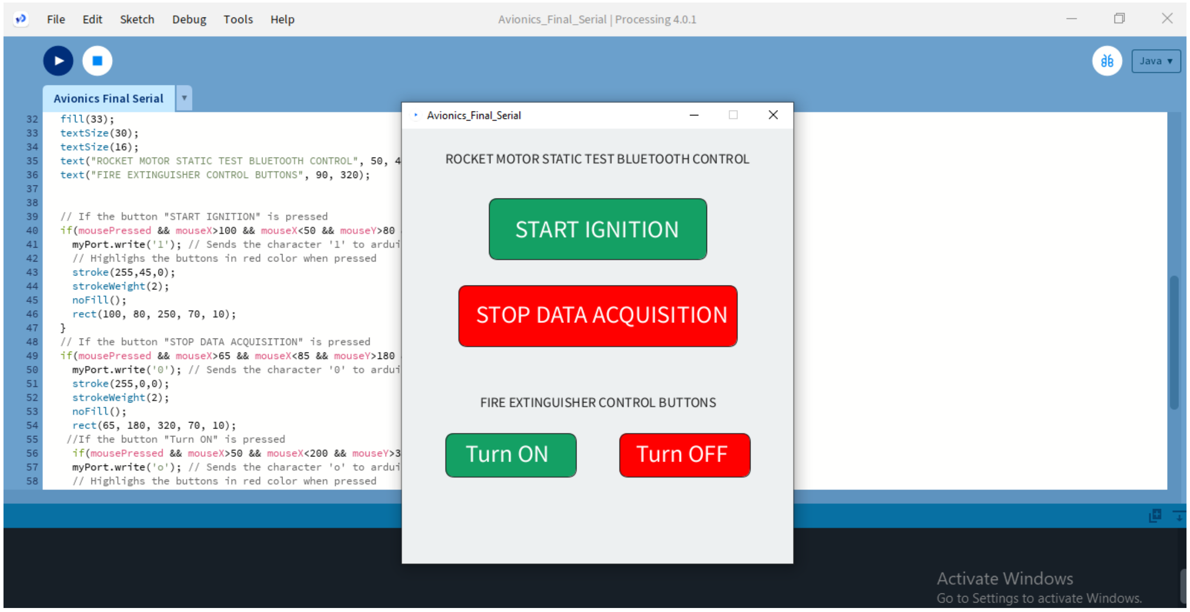
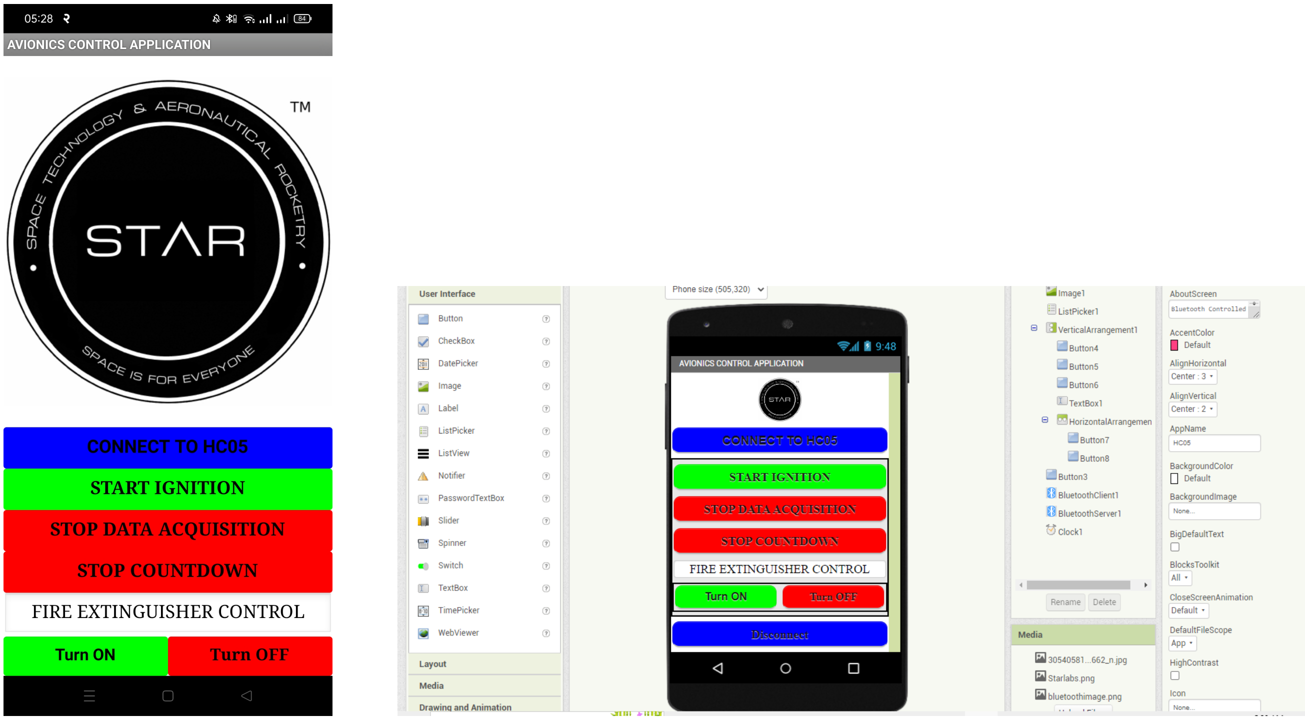


Figure 6: Graphical Interface generated with Processing Software

```
// If the button "START IGNITION" is pressed
if(mousePressed && mouseX>100 && mouseX<50 && mouseY>80 && mouseY<150){
  myPort.write('1'); // Sends the character '1' to arduino, which then stops data acquisition.
  // Highlighs the buttons in red color when pressed
  stroke(255,45,0);
  strokeWeight(2);
  noFill();
  rect(100, 80, 250, 70, 10);
}
// If the button "STOP DATA ACQUISITION" is pressed
if(mousePressed && mouseX>65 && mouseX<85 && mouseY>180 && mouseY<250){
  myPort.write('0'); // Sends the character '0' to arduino, which then stops data acquisition.
  stroke(255,0,0);
  strokeWeight(2);
  noFill();
  rect(65, 180, 320, 70, 10);
 //If the button "Turn ON" is pressed
```

Figure 7: Code specific to sending commands to Arduino through HC05

I also thought of controlling avionics system using mobile phone. At first, I had planned to use Blynk application (legacy version), but I found that the legacy version had been shut down. With the new version, I didn't find any options for bluetooth connection, and I felt the new version to be a bit difficult than the legacy version. Therefore, I switched to another method of controlling avionics from mobile phone. I used MIT App Inventor to create a simple mobile application which sends commands (in the form of text) to Arduino when the mobile is connected with HC05 bluetooth module. Below are the pictures of mobile application and designer view and code blocks view of application in MIT App Inventor:-



(a) in mobile (b) in designer view of MIT App Inventor

Figure 8: Mobile Application View

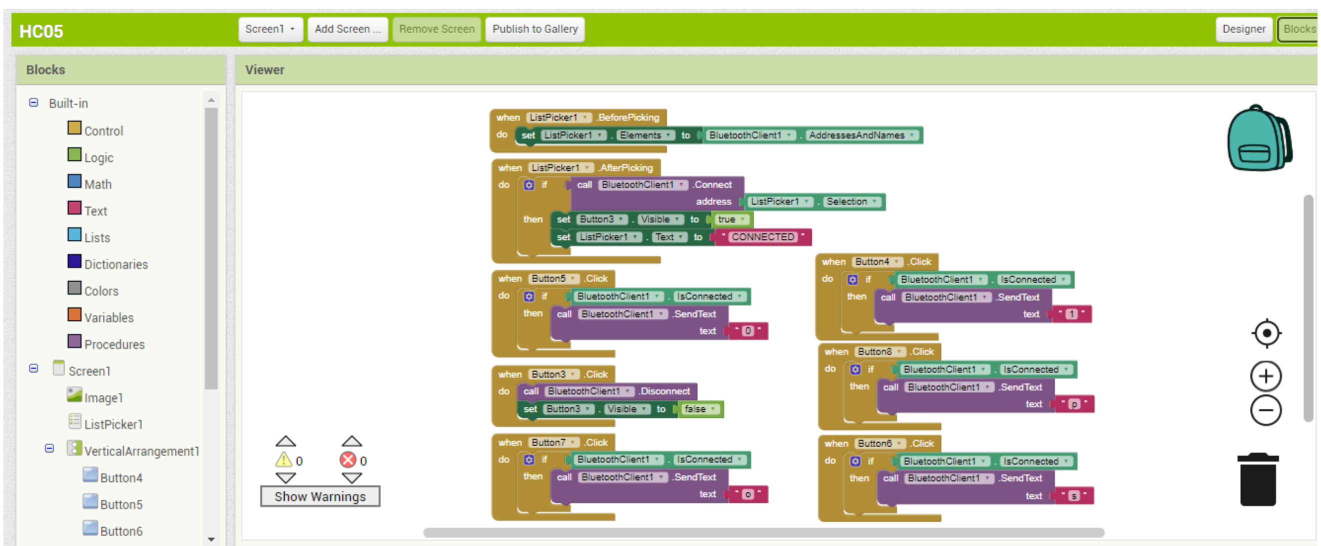


Figure 9: Code Blocks view of application in MIT App Inventor

While designing circuit and writing Arduino sketches, I didn't face many problems. While I did face a long-term (probably a day) problem while writing code for stopping countdown and restarting ignition, I didn't face any other major problems. To solve the mentioned problem, I used hit and trial method: I was trying different loops, functions, etc until I got it the way I wanted it. As mentioned earlier in the algorithm of my work, I designed circuit for specific components first. When I completed simulating all components of avionics in Proteus, I combined other circuits in the Proteus file containing load cell circuit. While combining circuits, I chose same pins for each components. For example, if I had used pin 7 for buzzer while simulating it individually, I'd also use pin 7 for buzzer in the main circuit. This strategy of designing main circuit helped me with the code too. For code, I didn't have to write it from scratch. I just had to look for specific portion of code from individual codes for each components and use that portion in the main code. I didn't face any problem in simulation of my main circuit probably due to my strategy of "integrating" individual codes and individual circuits. One advantage I had in circuit simulation was that I was working in two laptops simultaneously which helped me avoid problems of connecting wires to different pins in the main circuit than those pins used in individual circuits. On the other hand, my first draft of PCB design was not practical. The shaper Harsh Patel suggested me to make PCB design as realistic and simple as possible. Then, I made few changes to the first draft of PCB design, but the second draft had some incompleteness because I didn't use screw terminals for connecting load cell to HX711 module. Lastly, I added screw terminals, and this time, my PCB design was completed. Below are the pictures of Proteus Circuit Design and PCB design:-

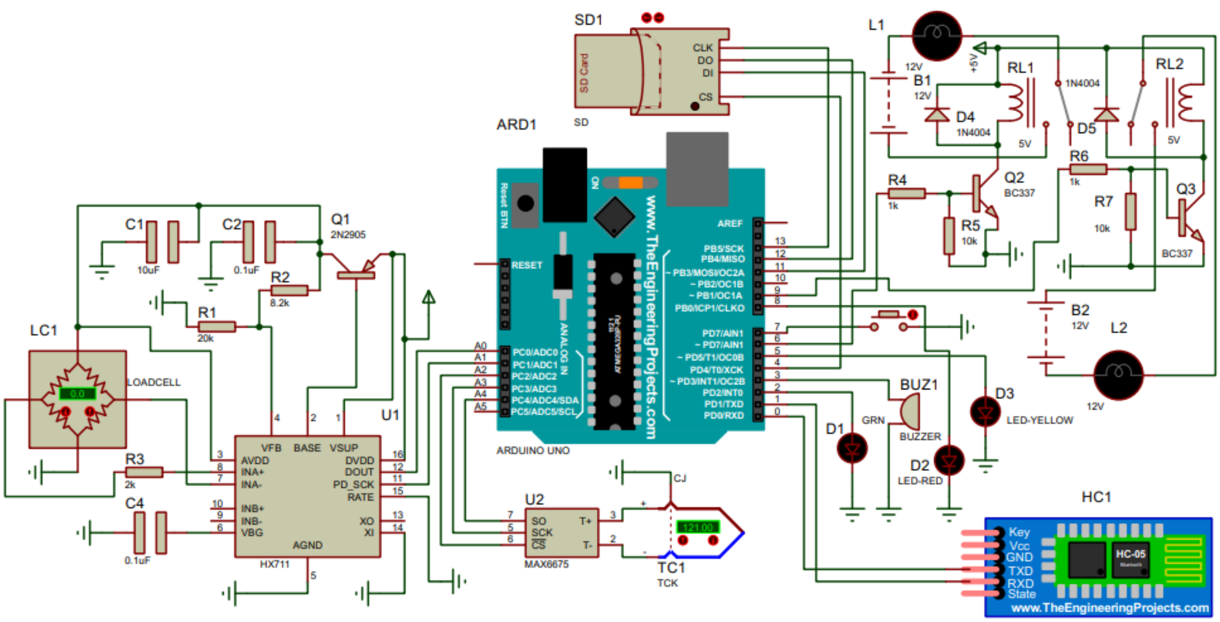


Figure 10: Circuit Design

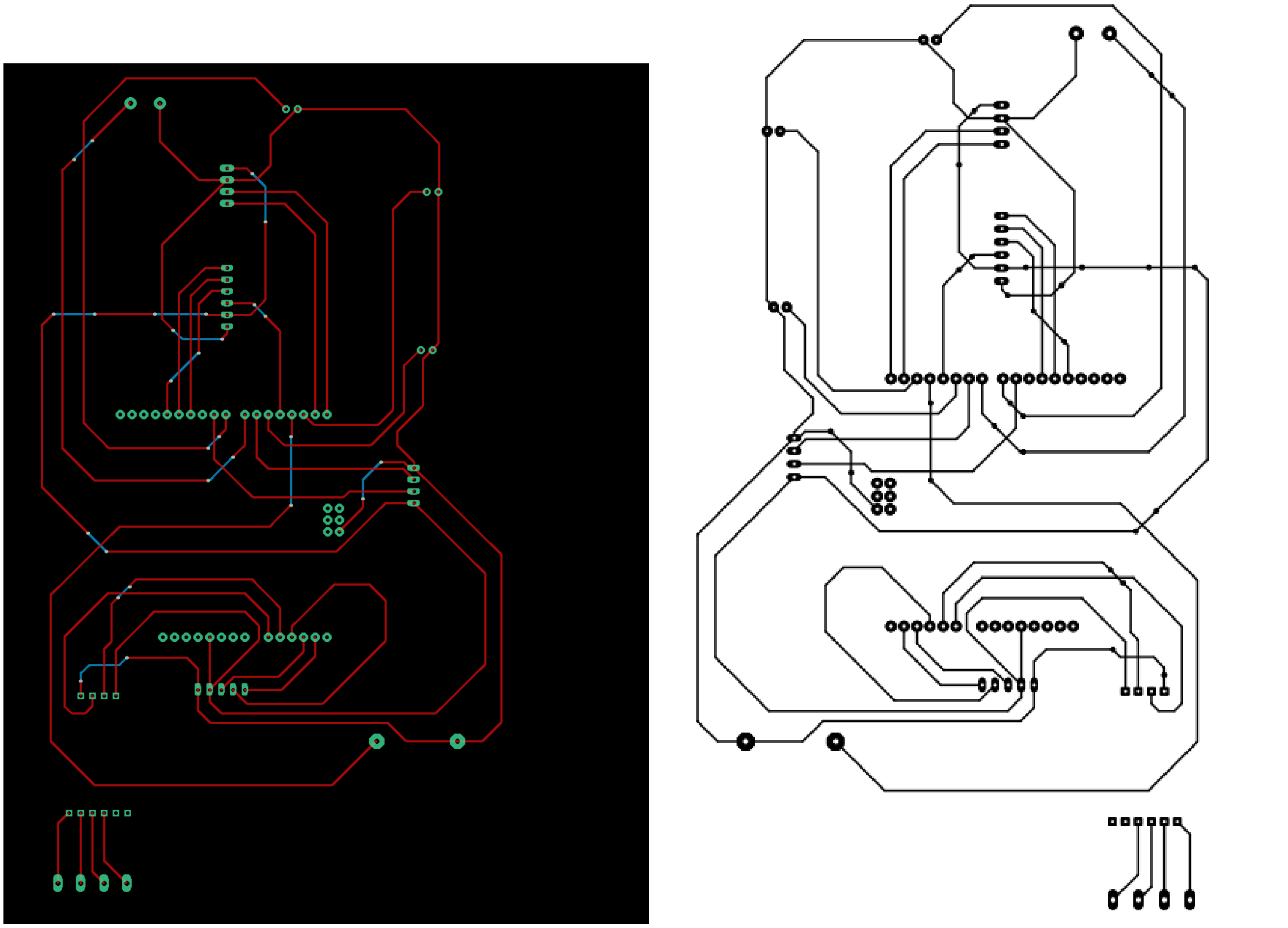


Figure 11: PCB Design

Guide for how to build Avionics:

1. Research what parameters you need to analyze for performance analysis of rocket motor.
2. Research about electronic components according to the parameters and your other needs such as safety system.
3. Design your circuit in Proteus. Before designing whole circuit, design circuits for individual electronic components first.
4. Write your sketch in Arduino IDE and upload hex file to Proteus Circuit.
5. Simulate your circuit in Proteus.
6. If the simulation is working as intended, design the PCB in Eagle EDA for manufacture purpose. If not, solve the problem.

7. Assemble all electronic components in the PCB and upload sketch to your microcontroller.

Mechanical Design

The mechanical design team chose T-Slot Aluminum Bars over normal bars because of their flexible design, reliable strong structure, and fast and precise assembly. Moreover, using these bars will rule out the use of welding. Design team used fasteners, which are specially designed for T-slots. They are used to connect gusset plate to the bars. The design team included L gussets, Box gussets, and T gussets for connecting all the bars with each other. An avionics bay is a box that holds all the electronics such as the printed circuit board, The design team has used a glass cover on avionics bay housing to facilitate clear view of electronics from outside. Likewise, motor mount will hold the motor axially and the ball transfer units will be attached to this shaft to facilitate the axial motion of the motor while it is burning. Ball transfer units are omni-directional load-bearing spherical balls mounted inside a restraining fixture. They are identical in principle to a computer trackball (a pointing device). Typically, the design involves a single large ball supported by smaller ball bearings. There will be M7 nuts, which will be used to connect the fasteners and gusset plates with T-slot bars.

Guide for how to build Design

1. Research on the basic components which you'll require to make the Static Test Pad.
2. Select the components you require and make a CAD model of each of them using Fusion 360.
3. Assemble all the components using rigid joints in Fusion 360.
4. Simulate the model of the test pad using Fusion 360 by putting necessary constraints and generating manual and automatic contacts. Then, give appropriate load to the place of load cell according to the class of motor for which the test pad is made.

Following are some pictures of the components used in the design of the static test pad:-

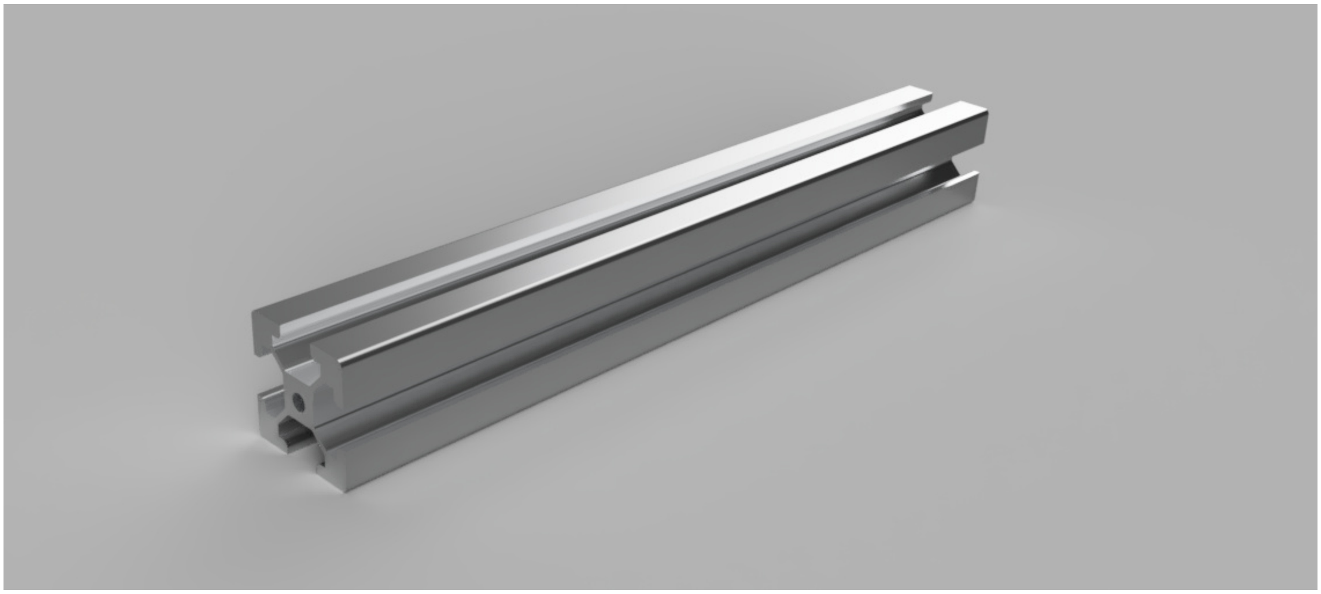


Figure 12: T-Slot Bar

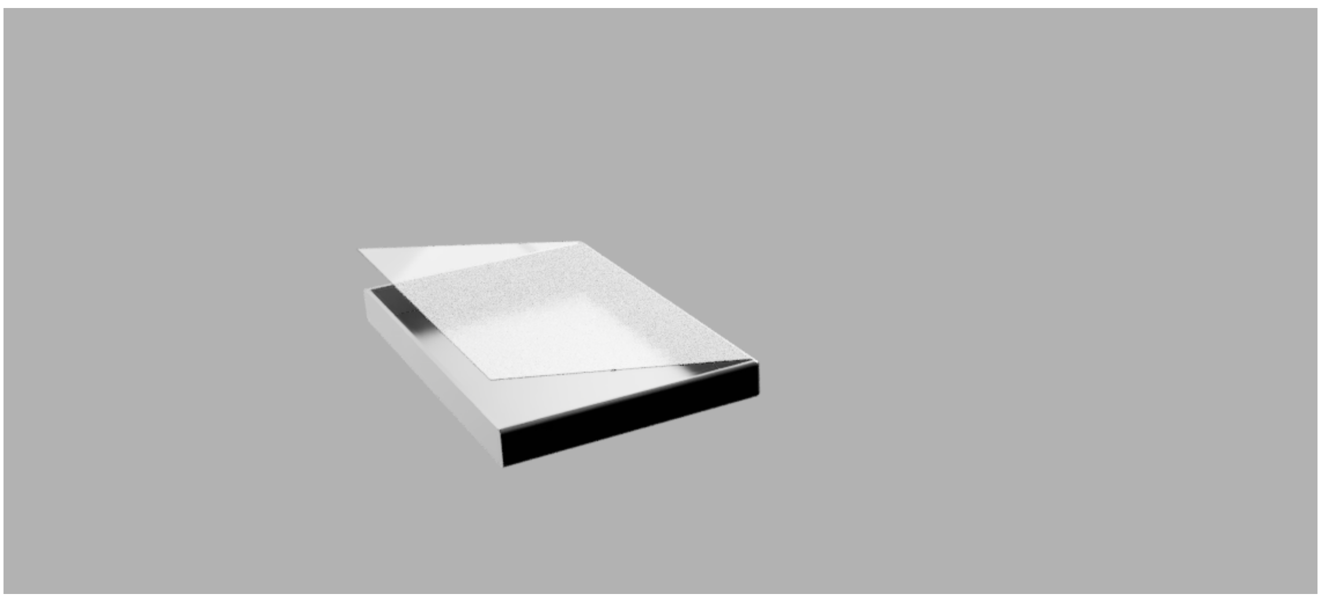


Figure 13: Avionics Bay Housing

Figure 14: Box Gusset

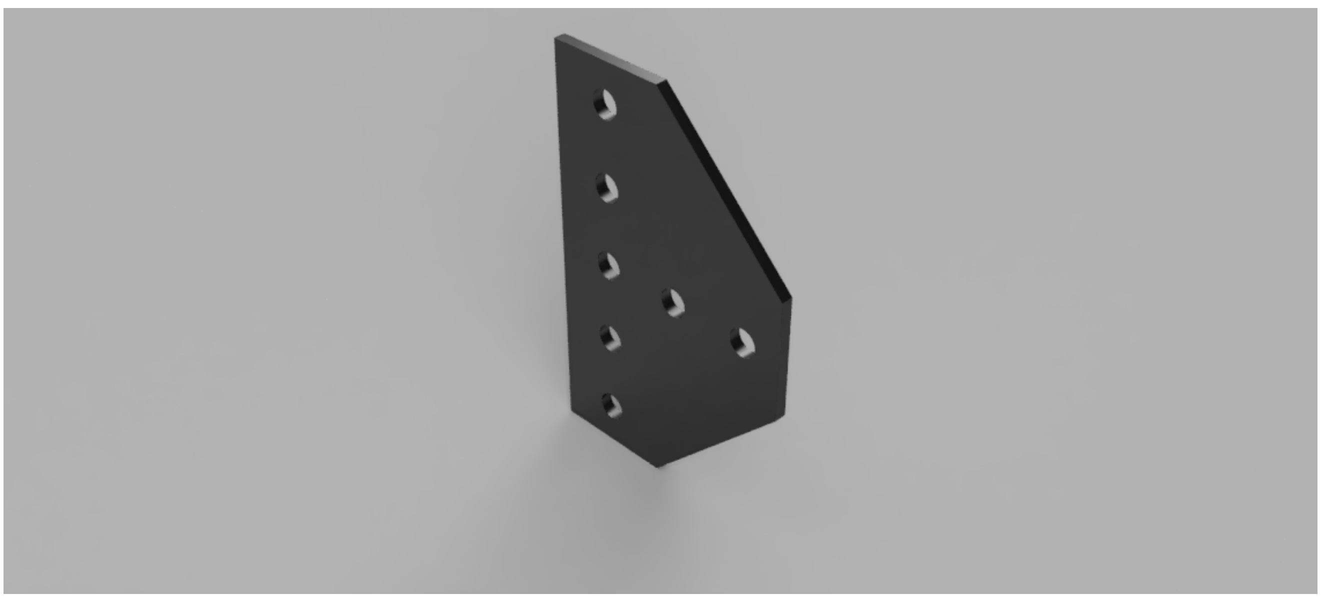


Figure 15: T Gusset

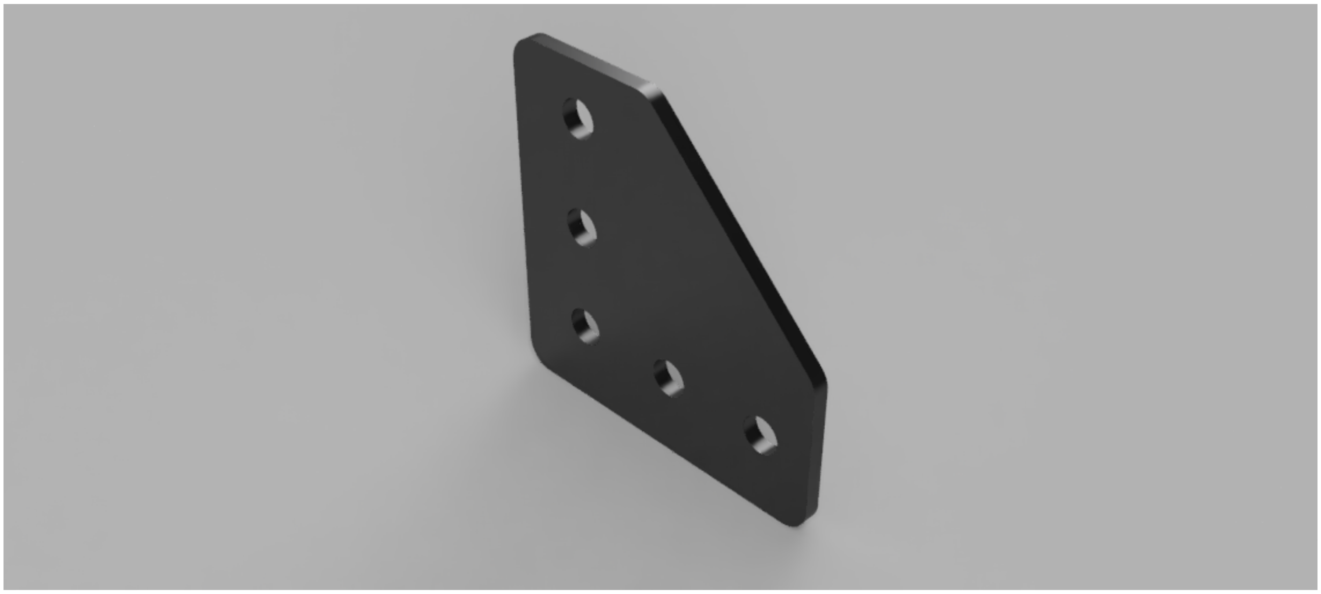


Figure 16: L Gusset

Figure 17: Fastener

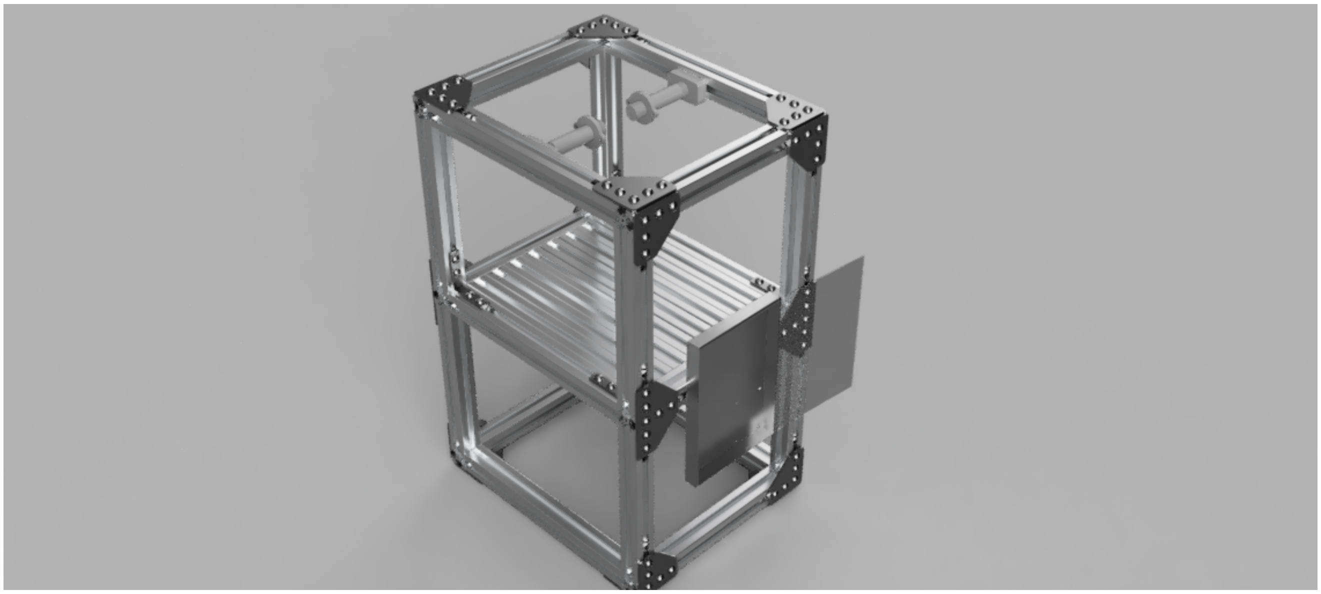


Figure 18: Assembly of the Static Test Pad

Cost

The costs of components used in mechanical design and avionics of the static test pad are tabulated below:-

Table 4: Total Cost of components used in Mechanical Design

S.N.	Components	Dimensions	Quantity	Cost [INR]
1.	T-Slot Aluminum Bars	235mm*30mm*30mm	8	592.2
2.	T-Slot Aluminium Bars	305mm*30mm*30mm	20	1921.5
3.	T-Slot Fasteners	25mm*13mm*5mm	52	3640
4.	Gusset Plates (Corner)	55mm*55mm*3mm	16	80
5.	Gusset Plates (3 edged)	100mm*55mm*3mm	4	24
6.	Box Gusset Plate	55mm*55mm*3mm	4	14
7.	Avionics Bay Housing	137.5mm*203mm*15mm	1	33.48
8.	Ball Transfer Unit	40mm dia; 60mm length	4	560
9.	Motor Mount Shafts	95mm*24mm	4	460
10.	Nuts	M7 15mm	52	104
11.	**Total**			**7429.18**

Table 5: Total Cost of electronic components in Indian and Nepali markets

S.N.	**Component**	**Cost (Indian Market) [INR]**	**Cost (Nepali Market) [NPR]**
1.	Arduino Uno R3	800	1150
2.	HX711 Module	79	150
3.	Button Type Load Cell	6000	9600
4.	SD Card Module	71	180
5.	SD Card	300	450
6.	HC05 Bluetooth Module	210	710
7.	LEDs	5	15
8.	Push Button	10	10
9.	Buzzer	40	25
10.	2 Channel 5V Relay Module	130	280
11.	Batteries	236	380
12.	Nichrome Wire	35	60
13.	Solenoid	600	750
14.	MAX6675 module with k-type thermocouple	400	435
15.	**Total**	**INR 8916**	**NPR 14195 (INR 8872)**

Sketches

The link for Processing Software sketch is HERE. The link for Arduino IDE Software sketch is HERE.

Observation

Circuit design in Proteus software was correct as well as the Arduino sketch. Circuit simulation was working properly in Proteus. As expected, countdown was of ten seconds and ignition duration was of two seconds. The fire extinguisher could be turned on and off any time of the simulation. The sensor readings were collected at the rate of fifty samples per second. Data Acquisition Stoppage system was also working properly. PCB design was also deemed correct by the shaper. Since I didn't have HC05 bluetooth module, I couldn't test mobile application and laptop with HC05, but I did check mobile application by modifying code blocks to start countdown in mobile screen when "START IGNITION" was clicked so as to allow the buttons to work when mobile is not connected to bluetooth, and it was working properly. The total cost of electronic components is estimated at INR 8916 in the Indian market and INR 8872 (NPR 14195) in the Nepali market.

Results

This report elaborates the avionics system of rocket motor static test pad. Circuit was simulated multiple times to validate whether it's working properly or not. Mobile Application and Graphical

Interface were generated using MIT App Inventor and Processing software for mobile and laptop respectively. These applications are expected to be useful in real time static testing. I also expect them to work as intended when mobile or laptop is connected to HC05 bluetooth module.

Conclusion

Avionics System for Static Test Pad with predefined requirements was successfully designed and simulated virtually. There were also features other than the predefined ones. These features include stopping data acquisition, stopping countdown, remote controlled ignition system, and controlled fire extinguishing. In all, an avionics system with reasonable price for the features it provides was designed successfully.

Precautions

Following precautions need to be considered while performing static test fires:-

1. Keep a safe distance from static test pad during static testing.
2. Do not perform static testing in or near residential area.
3. Perform under proper supervision if you are not experienced in static tests.
4. Be aware of local government's policies about rocket motor testing.
5. Be sure to check all the electrical connections in the avionics bay.
6. Connect RXD and TXD pins of HC05 to TXD and RXD pins of arduino respectively only after uploading code to Arduino.