Internship Report on

Static Test Pad for Rocket Motor

At STAR – Space Technology and Aeronautical Rocketry

P.HEMANTH VERMA

Department Of Aeronautics

Hindustan Institute Of Technological Sciences

(7^{TH} January 2022 --- 7^{Th} February 2022)

Problem Statement:
To design a Rocket Motor Static Test Pad for testing and acquiring the required data for the performance and analysis of the high-powered rocket motors.

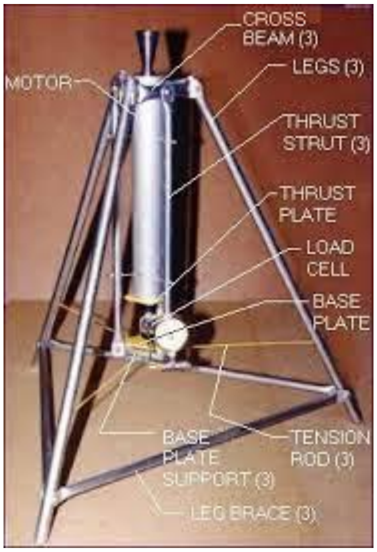
Department-specific Problem Statement:

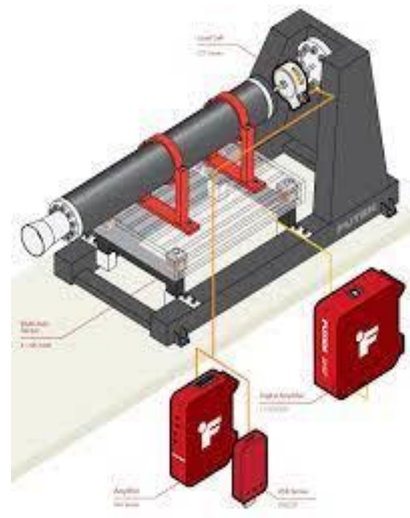
Department of Avionics systems:
- To design a system to acquire and process the data
- To design the electrical power system
- To build a test control system
- To acquire and store the required data for further analysis
- To design the systems for real-time data transmission and reception
- To develop or discover a technique/tool to store, display and analyze data

Designing:
- To design the static test pad (STP) mechanism with a good strength/weight ratio
- To design an STP with pleasing aesthetics and ergonomics
- To design a STP that can be assembled and dissembled easily
- To design a structure to install all electronics and power systems conveniently
- To perform the required simulations to minimize the errors while operating

1.0 What is a Static Test Pad?

The Static Test Pad is a test stand designed to conduct full-scale static test fires and other extensive subsystem tests of a solid rocket motor. It is often performed with computer modelling to characterize their thrust curves and Isp in static testing conditions for the propulsion of a rocket motor. Some standard Models of a Static Test Pad are illustrated below:





(The above-shown images are just for reference purposes; it isn't an actual representation of our STP)

1.1 Basic working principle of a Static Test Pad:

The static test pads are commonly used to test rocket motors and engines of various sizes. A program runs both the motor ignition and data acquisition systems to produce time-resolved thrust data. It is rooted in curbing any dynamic or pseudo forces as gravity acting on the body. Now, the sequence by which the static test pad is initiated is given by –

1. All the connections are made and verified. The load cell is mounted onto the central support beam and the motor onto the mounts.

2. Then, the load cell is connected to the avionics bay, Arduino, and amplifier is attached along with the Sd card.

3. A command is given through Arduino to start the ignition by NRF module that allows us to activate ignition remotely. When the motor starts, it exerts the thrust along the horizontal axis parallel; to the base, with exhaust towards the side.

4. The motor is fixed due to the motor mounts. Hence, it won't displace, translate or rotate, as the degree of freedoms is locked.

5. The load cell assembly is mounted to a fixed thrust measurement test stand securing the motor. As the motor is operating, thrust is measured with the load cell.

6. The rocket engine test stand mount prevents side loading on the cell, ensuring the load cell is only receiving axial loads.

7. Due to this, it vibrates to and fro like an oscillator. The vibrations generate stress in the load cell, directly connected to the motor mount. This stress would be converted to a digital signal by the HX711 Board, connected to the Arduino Uno via PCB as it is a force transducer, and stress is force per unit area.

8. Mass flow and Drag are captured with load cells. Thrust data can be live-streamed and logged for monitoring and later review and comparison against simulated models with external storage.

9. The data is continuously converted from analog to digital information throughout the test. We determine the calibration factor to see if there is an error in any sensors.

10. The testing automatically ends when the motor runs out of fuel. Hence, our static testing of the rocket motor is done using the static test pad.

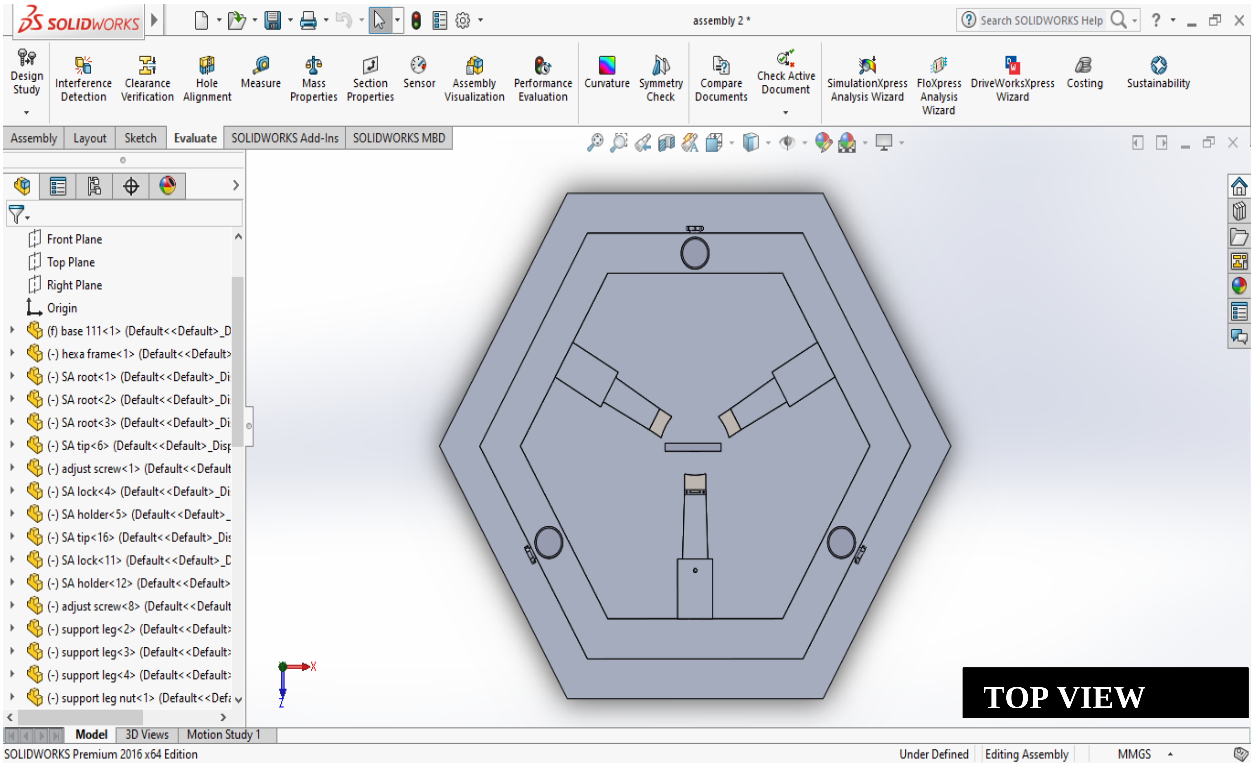
11. The data is stored onto the SD card for future calculations and as a backup system.

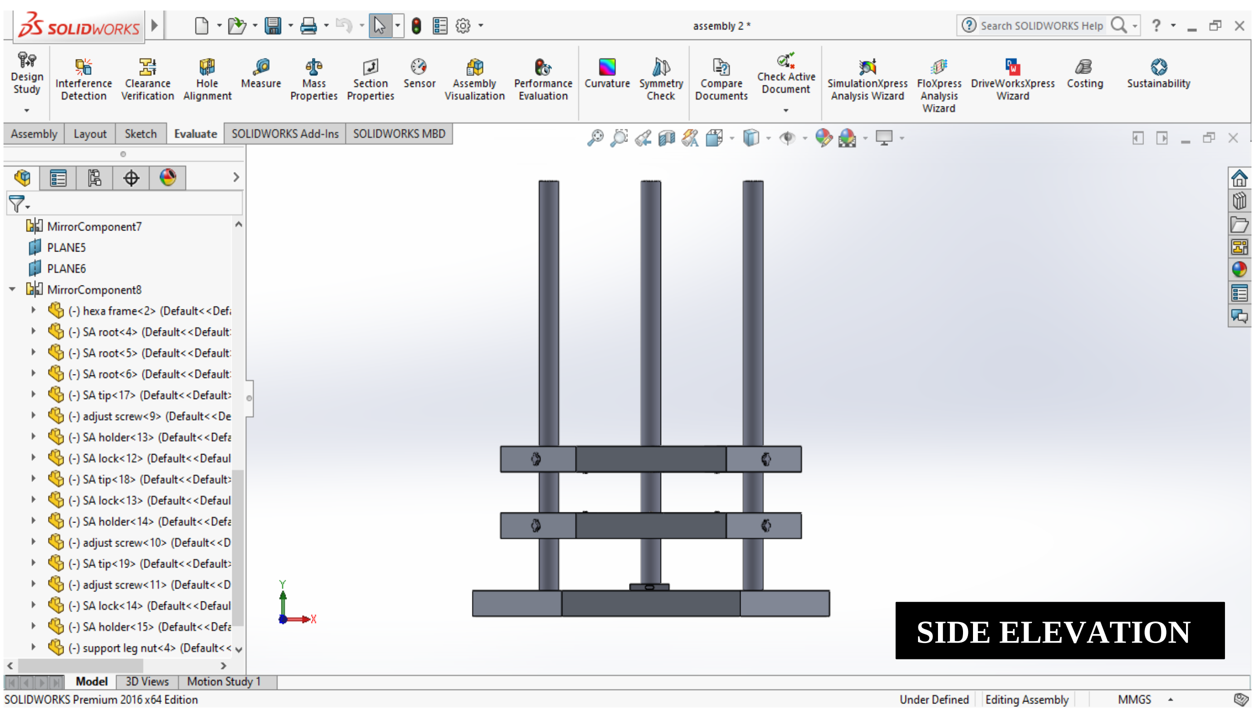
1.2 Why make a Static Test Pad?

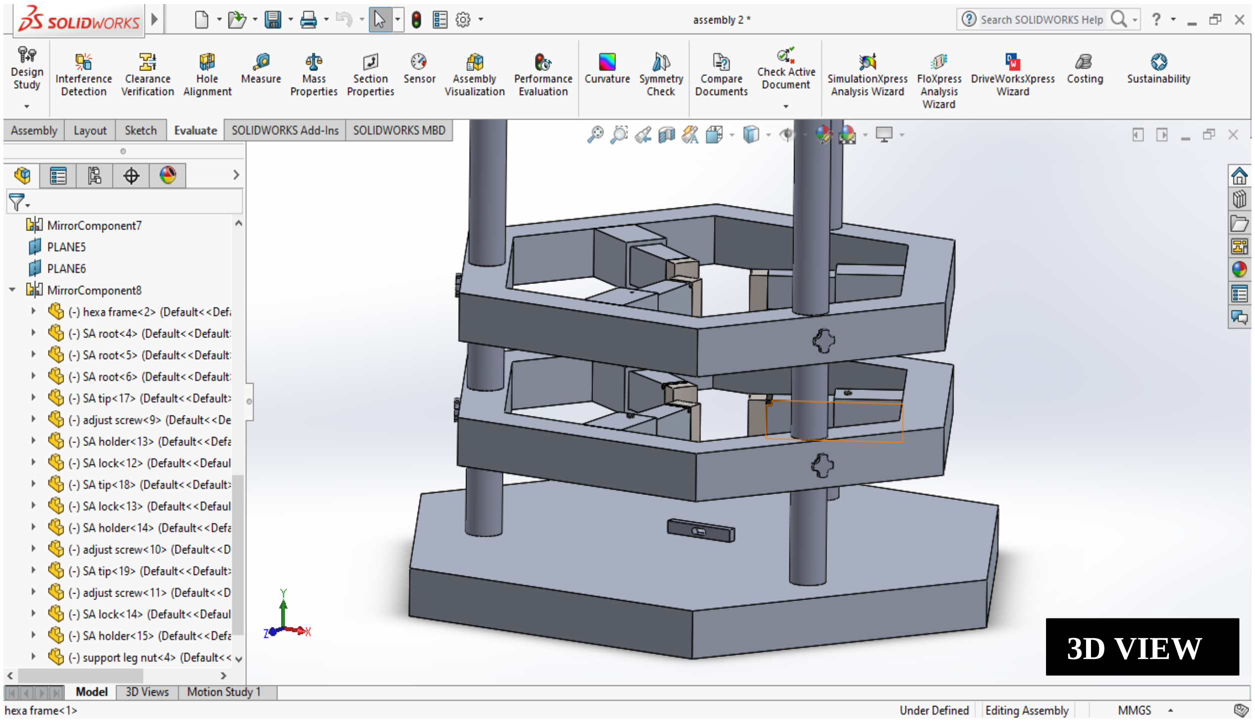
The last decade has seen an almost exponential increase in the number of model rocket launches by hobbyists and high-powered rockets for academic, outreach purposes, and experimental motor design. The emergence of new technologies like rapid prototyping, including 3D printing, is changing rocket motor design. This has demanded a series of small-scale static fire tests of soon-to-be-manufactured methods to explore the performance of rocket motors. A simple means of testing thrust is through a thrust measuring system (TMS) that uses load cells to detect the amount of thrust produced. The test pad tests the rocket engines in different regimes of operation by enabling the variation of parameters like:

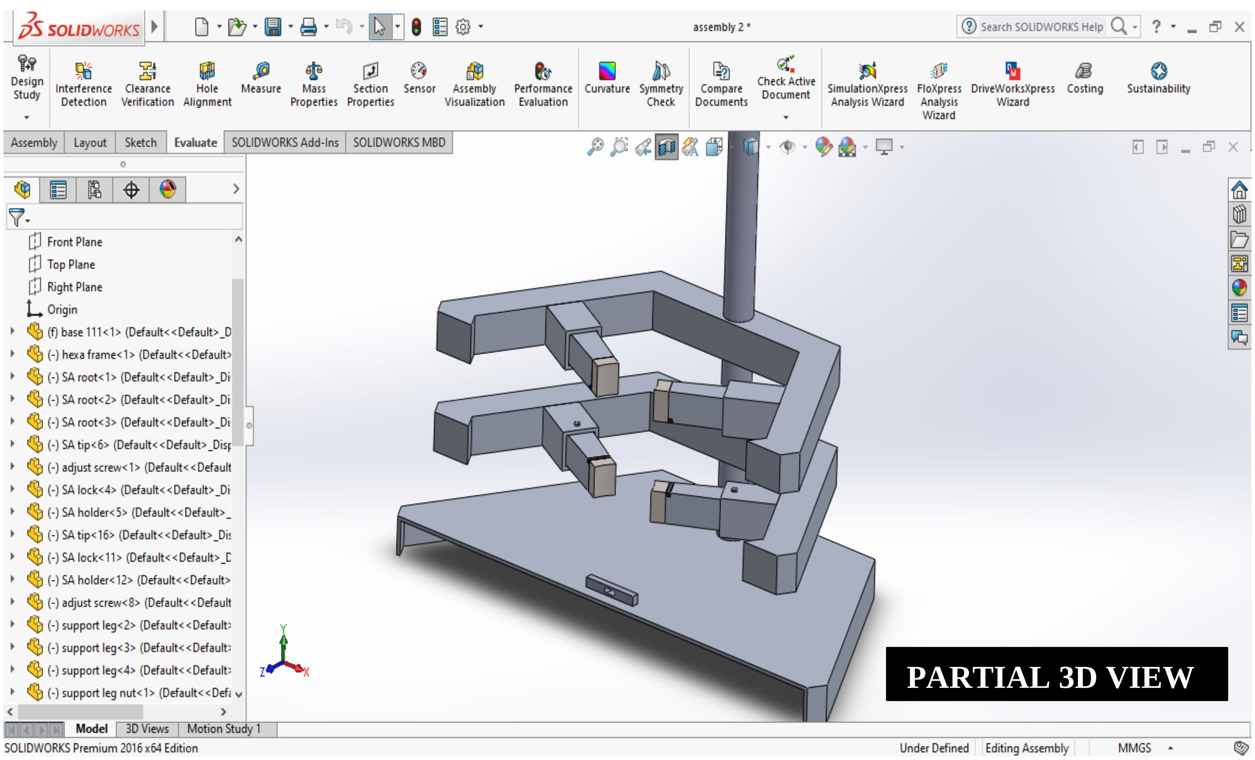
1. Tank pressure
2. mass flow rate
3. Drag (aka; kick)
4. firing time
5. real-time thrust

2.0 Architecture of The Designed Static Test Pad



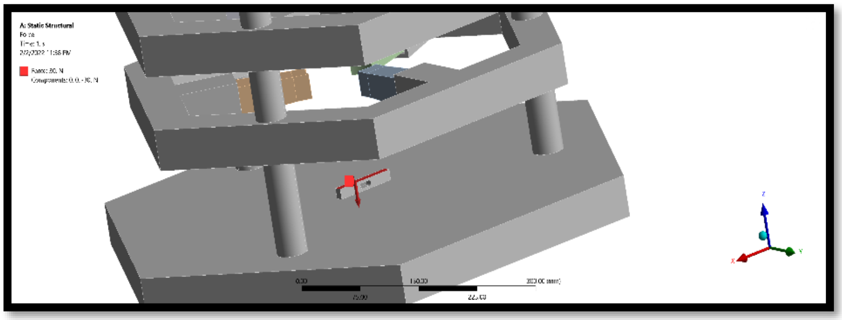


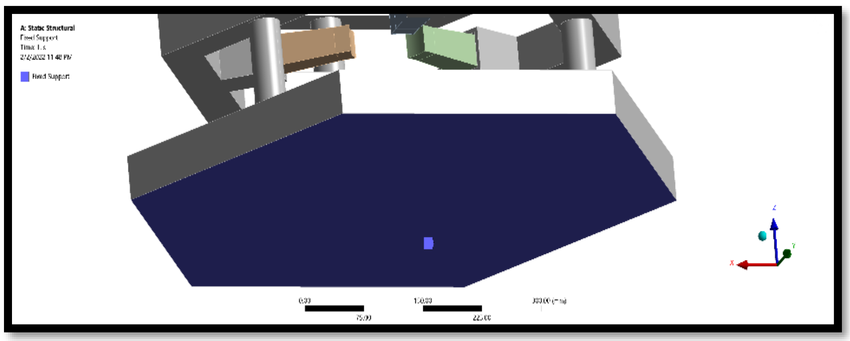

SOLIDWORKS
assembly 2 *
Design Study
Interference Detection
Clearance Verification
Hole Alignment
Measure
Mass Properties
Section Properties
Sensor
Assembly Visualization
Performance Evaluation
Curvature
Symmetry Check
Compare Documents
Check Active Document
SimulationXpress Analysis Wizard
FloXpress Analysis Wizard
DriveWorksXpress Wizard
Costing
Sustainability
Assembly
Layout
Sketch
Evaluate
SOLIDWORKS Add-Ins
SOLIDWORKS MBD
MirrorComponent7
PLANE5
PLANE6
MirrorComponent8
3D VIEW
Model
3D Views
Motion Study 1
hexa frame<1>
Under Defined
Editing Assembly
MMGS


SOLIDWORKS
assembly 2 *
Front Plane
Top Plane
Right Plane
Origin
PARTIAL 3D VIEW
Model
3D Views
Motion Study 1
SOLIDWORKS Premium 2016 x64 Edition
Under Defined
Editing Assembly
MMGS

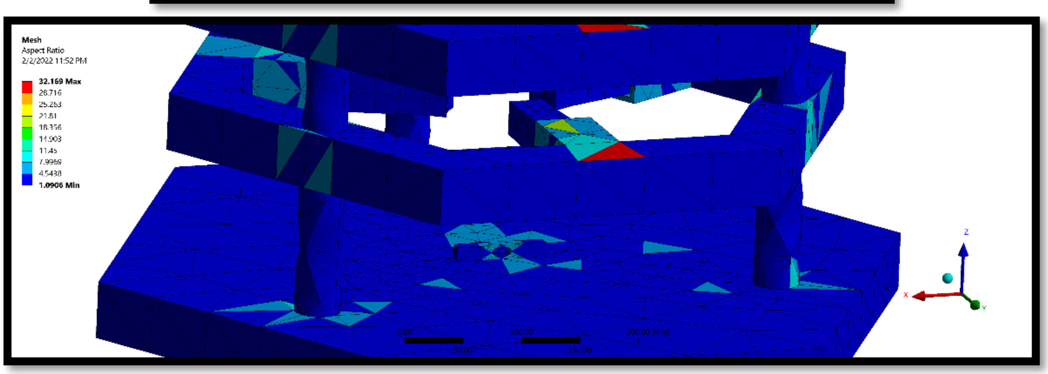
2.1 Analysis of The Architecture

The first boundary conditions were applied as a force (which will change due to different thrust) in the load cell and a fixed supported in the lower face.

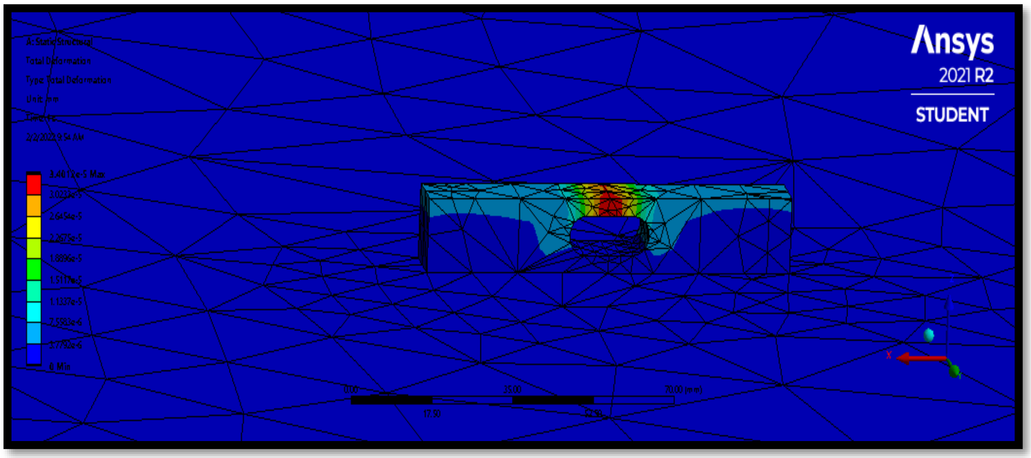




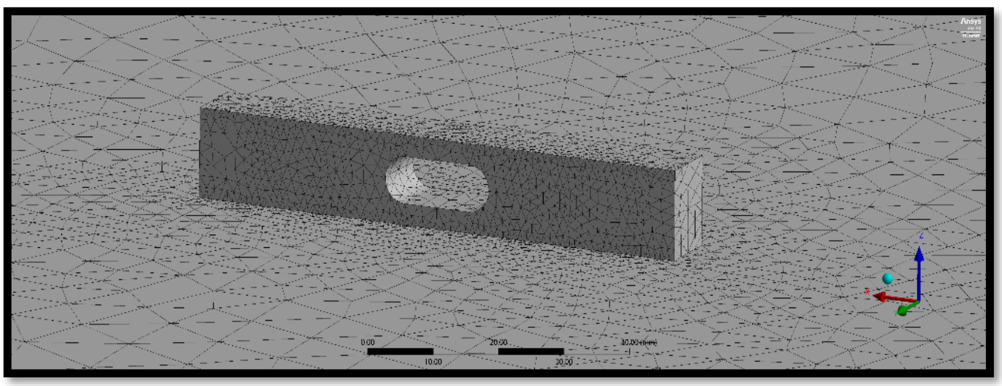
Generating a default mesh in the static test pad geometry. We get:



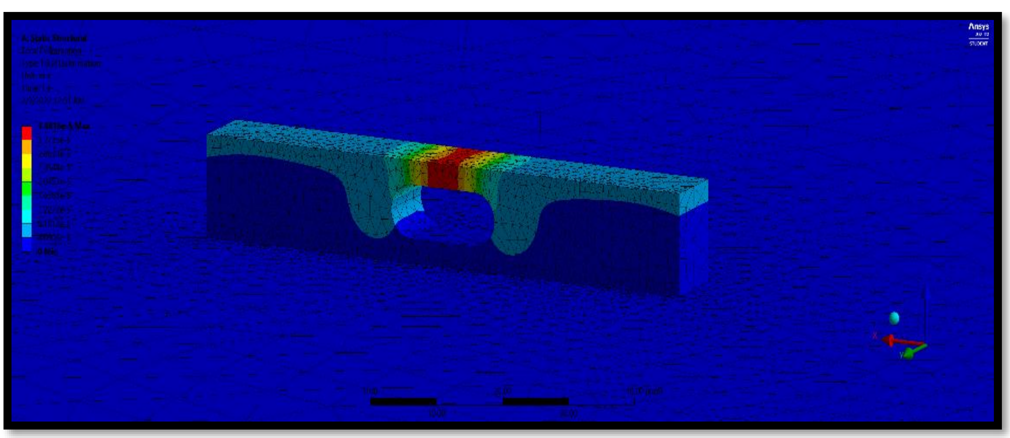
This course mesh has in the load cell (interest zone) a high aspect ratio, and this will affect the interpolation process, so with a 30 Newton of thrust force, we have the following result:



It shows a non-fine convergence in the adjacent elements and an unstructured mesh, so we generated a refinement mesh only in the load cell to perform the interpolations process and obtain better results and in the other parts is not necessary because there the stresses will be the same at the Saint Venart's principle.

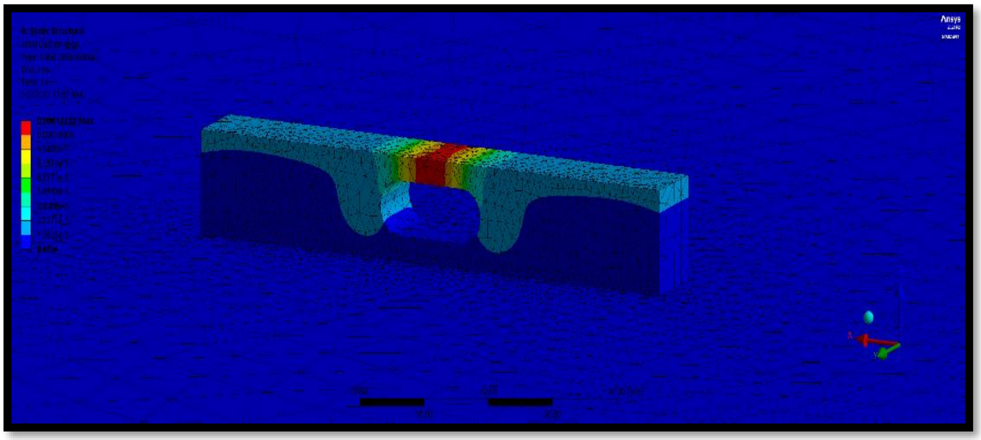


This mesh will improve the results in the interest zone.

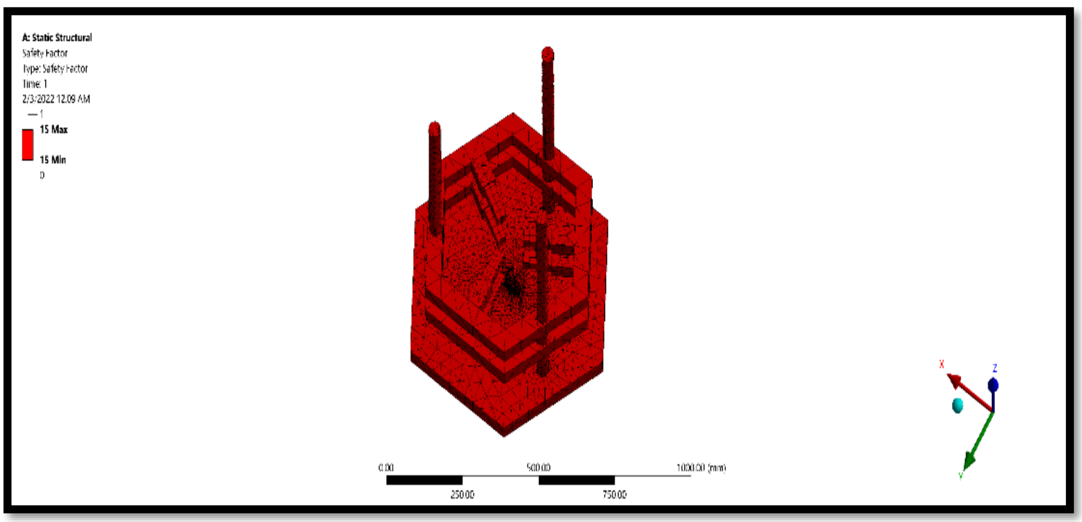


So, we can see a better interpolations process with a fine and structured mesh, generating a simulation close to reality with better results applying the same 30 Newton force.

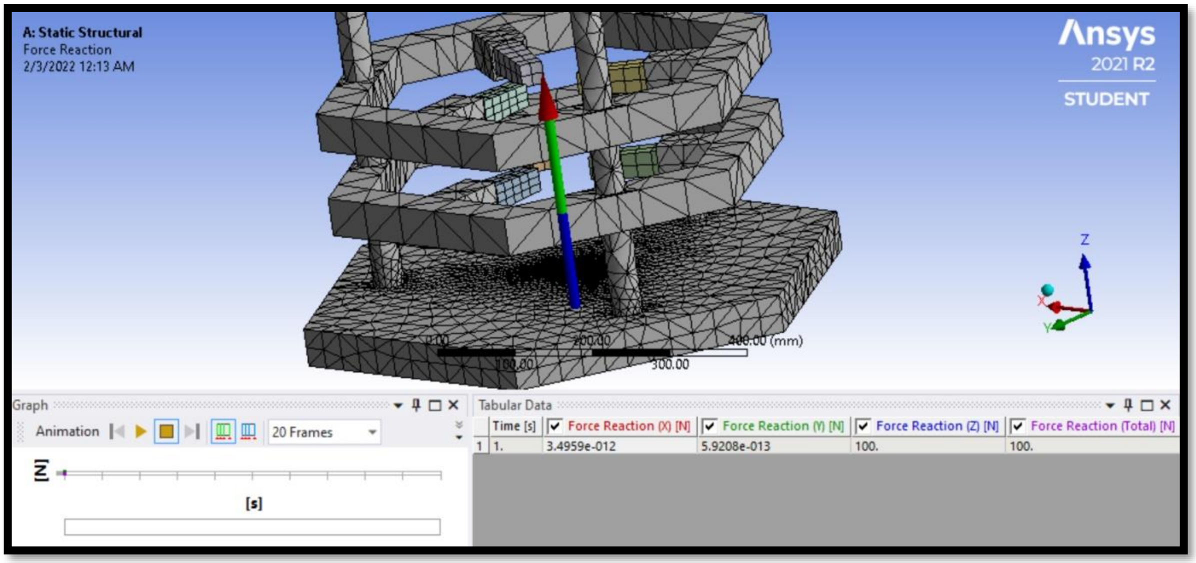
Furthermore, we put an upper thrust of 100 Newton and obtained this result.



Also, with the maximum thrust, we obtain a factor of safety of 15, we have much more certainty in our design, and it will be structurally very safe.



In other words, we can affirm that this is a good simulation. In every simulation, the resultant reaction forces in the vertical direction are equal to the thrust applied and in the other two laws are a resultant load that tends to zero.



Finally, we can see the maximum deformation and Von Mises Stress of the load cell for different thrust forces in the following table.

Trust [N]	Deformation [mm]	Von Mises Stress [MPa]
30	3,7e-5	0,4
40	5e-5	0,55
50	6e-5	0,7
60	7e-5	0,83
70	9e-5	0,97
80	1e-5	1,1
90	0,00011	1,2
100	0,00012	1,4

3.0 AVIONICS

Avionics are vital to design a Rocket Motor Static Test Pad for testing and acquiring the required data for the high-powered rocket motors' performance analysis. As a member of the same, I tried to develop economic and efficient ideas for our team. Thankfully the team members were beneficial as well. We together completed this problem statement by dividing the task amongst us.

3.1 Components in the Avionics Bay

1. **Arduino Uno R3:** The Arduino Uno R3 is a microcontroller board based on a removable, dual-inline-package (DIP) ATmega328 AVR microcontroller

2. **Amplifier:** An amplifier, electronic amplifier, or amp is an electronic device that can increase the power of a signal. We may need to amplify the digital signal of the data collected through the load cell, and for that, we can use an AD595 Amplifier.

3. **Load cell**: A load cell (or loadcell) is a transducer that converts force into measurable electrical output. Although there are many varieties of force sensors, strain gauge load cells are the most commonly used type

4. **Alternative Data Storage System (SD Card):** We have to store and process the data collected through Arduino. Also, we need to have a backup in case of data loss, and due to that, we need SD Card and SD Cardholder

5. **LED and Buzzer:** Both were used to give proper warning and indications before the motor is ignited.

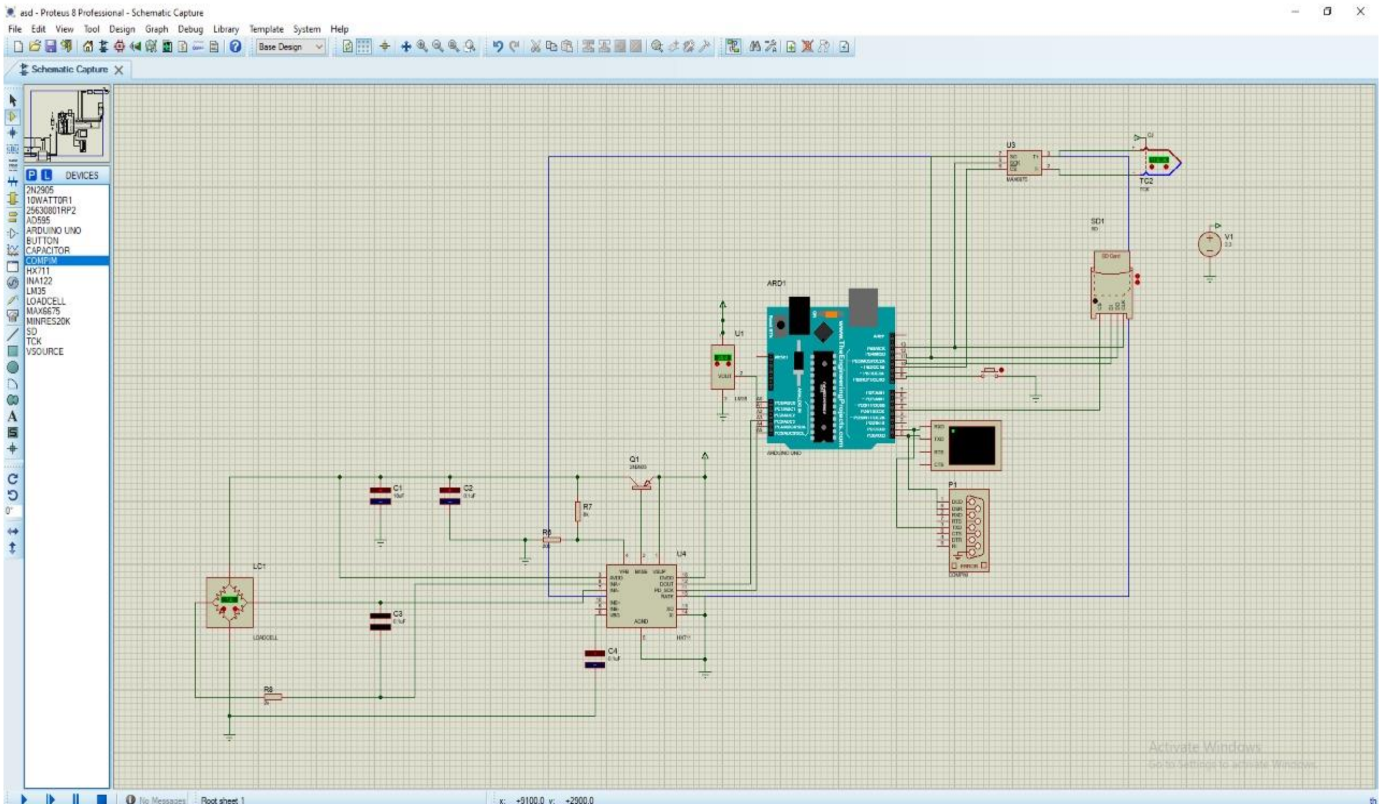
3.2 Working

1. When a motor is affixed to the load cell, the igniter is placed and wired, and the momentary pushbutton is activated, a 10-second warning time begins, with flashing a bright RED LED.

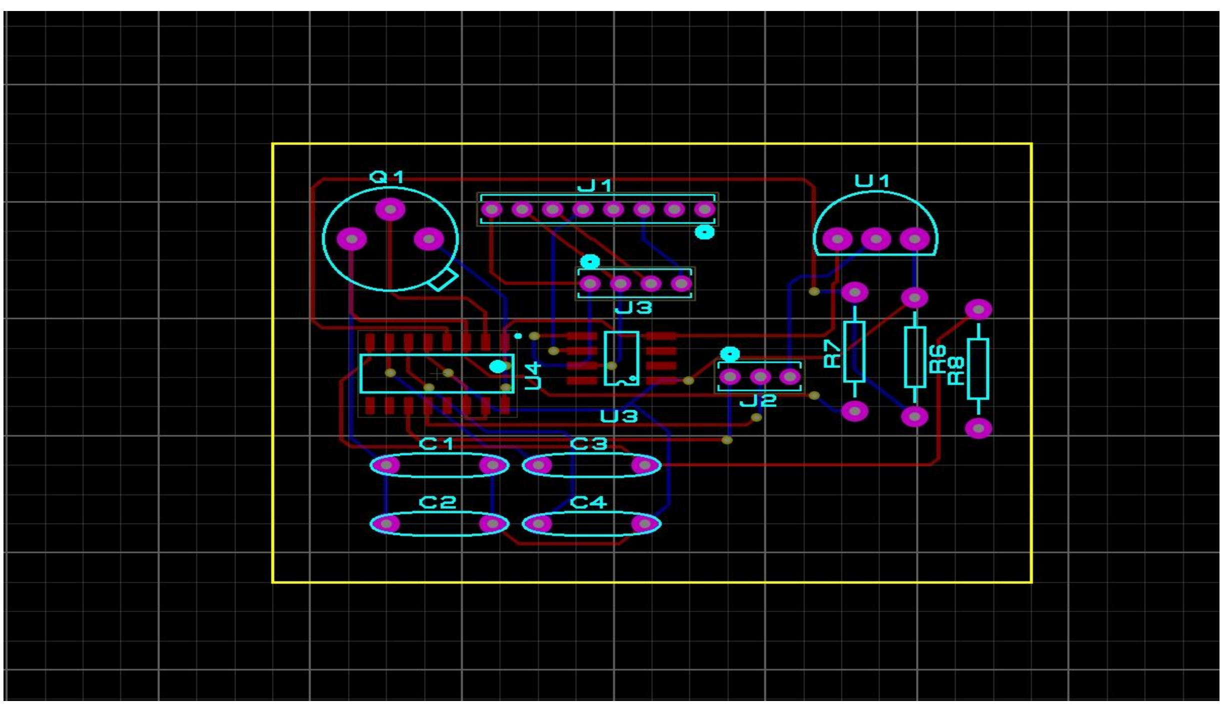
2. When the 10 seconds are up, the relay is closed, allowing the full 12V from a cell battery to the igniter. Immediately afterward, the Arduino logs the time and begins acquiring force data.

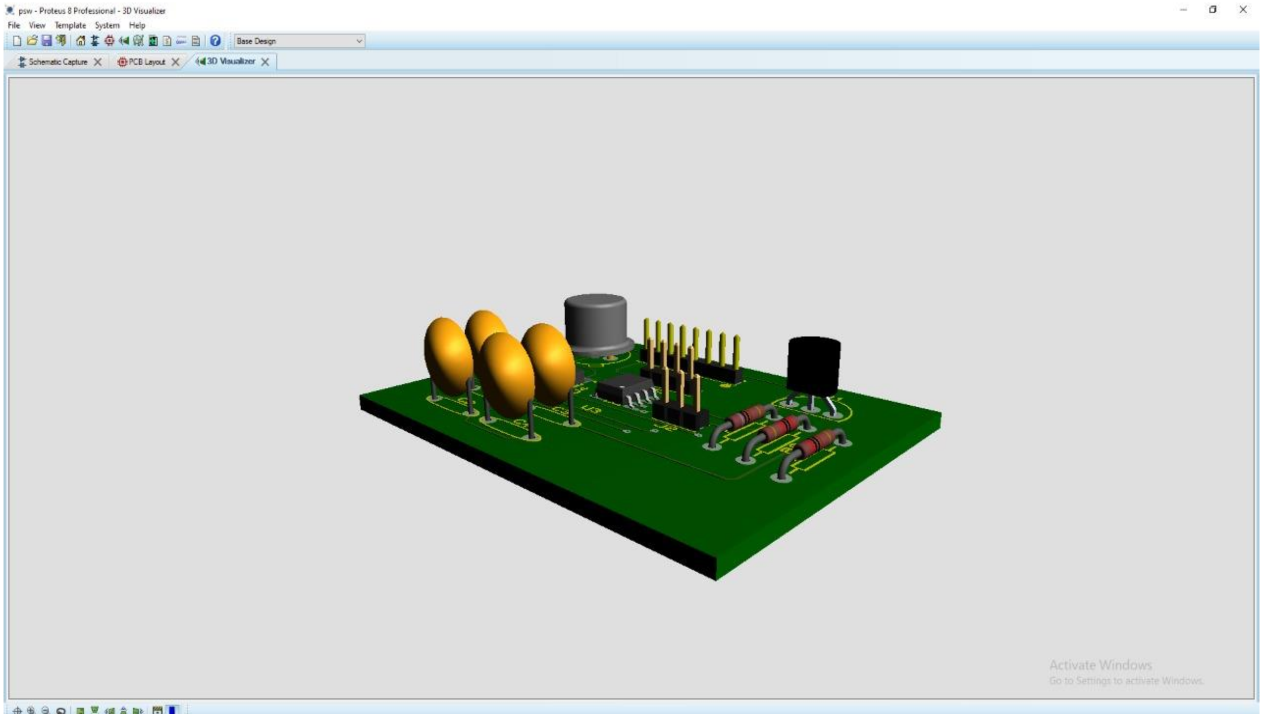
3. We have also attached an SD Card to the Arduino to store the data later.

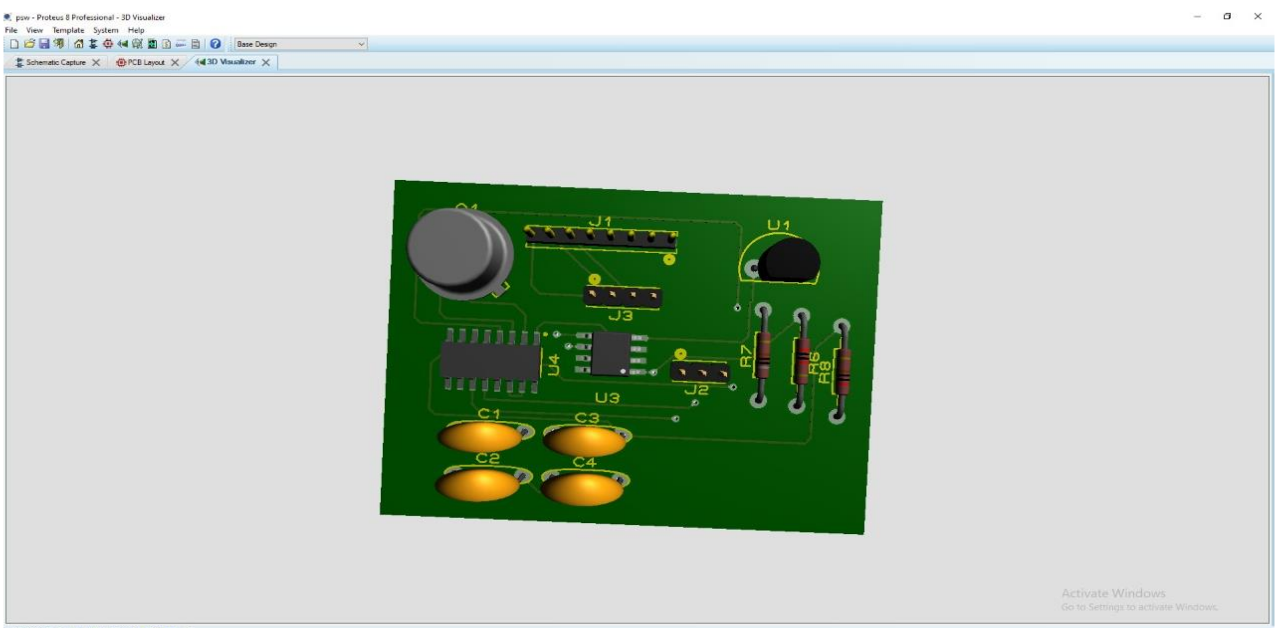
3.3 Circuit Diagram



3.4 Printed Circuit Board (PCB) Design




psw - Proteus 8 Professional - 3D Visualizer
File View Template System Help
Base Design
Schematic Capture
PCB Layout
3D Visualizer
Activate Windows
Go to Settings to activate Windows.


psw - Proteus 8 Professional - 3D Visualizer
File View Template System Help
Base Design
Schematic Capture
PCB Layout
3D Visualizer
J1
U1
J3
U4
J2
R7
R6
R8
U3
C1
C3
C2
C4
Activate Windows
Go to Settings to activate Windows.

3.5 CODE FOR THE WORKING OF ARDUINO :

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

#define DT A0
#define SCK A1
#define sw 8

long sample=0;
float val=0;
float temp;
long count=0;

unsigned long readCount(void)
{
unsigned long Count;
unsigned char i;
pinMode(DT, OUTPUT);
digitalWrite(DT,HIGH);
digitalWrite(SCK,LOW);
Count=0;
pinMode(DT, INPUT);
while(digitalRead(DT));
for (i=0;i<24;i++)
{
digitalWrite(SCK,HIGH);
```

```
Count=Count<<1;
digitalWrite(SCK,LOW);
if(digitalRead(DT))
Count++;
}
digitalWrite(SCK,HIGH);
Count=Count^0x800000;
digitalWrite(SCK,LOW);
return(Count);
}

void setup()
{
pinMode(SCK, OUTPUT);
pinMode(sw, INPUT_PULLUP);
lcd.begin(16, 2);
lcd.print(" THRUST ");
lcd.setCursor(0,1);
lcd.print(" Measurement ");
delay(1000);
lcd.clear();
calibrate();

//relay
// initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);

//LM35 temperature sensor
```

```
Serial.begin(9600);
  analogReference(INTERNAL);
  lcd.begin (16,2);                    // the codes written here is to run the programme once
  lcd.setCursor(0, 0);
  lcd.print("   Made by    ");
  lcd.setCursor(0, 1);
  lcd.print(" TEAM ENIGMA  ");
  delay (100);
  lcd.clear();
}

void loop()
{
count= readCount();
int w=(((count-sample)/val)-2*((count-sample)/val));
lcd.setCursor(0,0);
lcd.print("Measured THRUST");
lcd.setCursor(0,1);
lcd.print(w);
lcd.print("N ");

if(digitalRead(sw)==0)
{
val=0;
sample=0;
w=0;
count=0;
```

```
calibrate();
}

 //on loop blinking led
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(2000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second

//LM35 temperature sensor
  temp = analogRead(A2);             //temp = temp * 0.48828125;  //temp=temp*(5.0/1023.0)*100;
  temp=temp*1100/(1024*10);
  lcd.print("TEMP: ");
  lcd.print(temp);
  lcd.println("*C");
  delay(500);
  lcd.clear();                                  //Let system settle

}
void calibrate()
{
lcd.clear();
lcd.print("Calibrating...");
lcd.setCursor(0,1);
lcd.print("Please Wait...");
for(int i=0;i<100;i++)

{
```

```
count=readCount();
sample+=count;
}
sample/=100;
lcd.clear();
lcd.print("Put 100g weight");
count=0;
while(count<1000)
{
count=readCount();
count=sample-count;
}
lcd.clear();
lcd.print("Please Wait....");
delay(2000);
for(int i=0;i<100;i++)
{
count=readCount();
val+=sample-count;
}
val=val/100.0;
val=val/100.0; // put here your calibrating weight
lcd.clear();
}
```

4.0 List of Software that we used to build our Static Test Pad

- **Arduino IDE**–This is the software with which we did all of our programmings with the electronics systems. It was the primary system to command the static test pad.
- **Proteus Professional** – Proteus was used in simulating the working of the electronics part of our project. We built circuits and connections for relay through the nrf module, calibration factor, and storing data in SD Card.
- **SOLIDWORKS** – software that we used in our design. We designed all of the initial concept components and prototypes on Solidworks, and hence, all the designing was done on Solidworks.
- **ANSYS-** was used to do the analytical part of the design.

5.0 Precautions:

- Make sure that the static test pad can take a wide range of motor thrusts than the maximum thrust you intend to use. Adequate Safety factors are essential in such cases.
- Double check on the placement of the static test stand you erect by using a spirit level and clearing the ground off any combustible materials.
- Place the Static test pad on a rooted object of greater mass than the thrust of the rocket motor you are testing, plus also keep in mind the center of mass must be closer to the ground.
- Ensure to check all the electronic connections on the PCB and the nuts and bolts before initiating the command to fire the ignition.
- Keep an eye on the environment and local weather to avoid any damages in case of any inhibitors.
- In the unfortunate case of fumes, exhaust, and sparks being uncontrollable, have a fire extinguisher in proximity.
- Always maintain a safe distance from the static test pad during operation.