Internship Report on

Rocket Motor Static Test Pad

At STAR – Space Technology and Aeronautical Rocketry

Jatin Dhall (AVIONICS INTERN)

VELLORE INSTITUTE OF TECHNOLOGY,VELLORE

17TH JULY - 16TH AUGUST

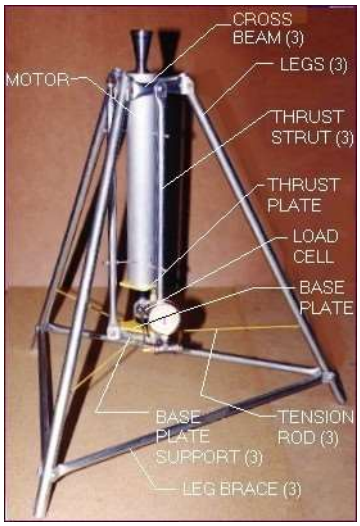
PROBLEM STATEMENT

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

Rocket Motor Static Test Pad

What is a Rocket Motor Static Test Pad?

A Static Test Pad is a device that is used to measure thrust-time characteristics and other properties like chamber pressure and temperature of a solid rocket motor. It is a standard process in the propulsion testing of a rocket motor.



Basic working principle of Rocket Motor Static Test Pad:

The Frame/Body:
The frame is designed to hold the rocket motor in place and provide stability and support while rocket motor is fired for testing.

The Avionics:
The avionics include the components responsible for collecting and analysing data using various sensors, for example the load cell (for measuring the thrust), SD Card (for saving data), HC-O5(Bluetooth module) for transmitting the data to the GUI and receiving commands from it, Ignition system, LED, Buzzers.

Why to make Rocket Motor Static Test Pad?

Rocket motors are often tested before they are integrated with the rocket for launch. These static test pads are used to test the rocket motor's various parameters. Static fire tests include firing a rocket motor at full thrust. The motor is fired for a few seconds, where various parameters are recorded in real time and analysed after the test to deem the motor ready for launch.

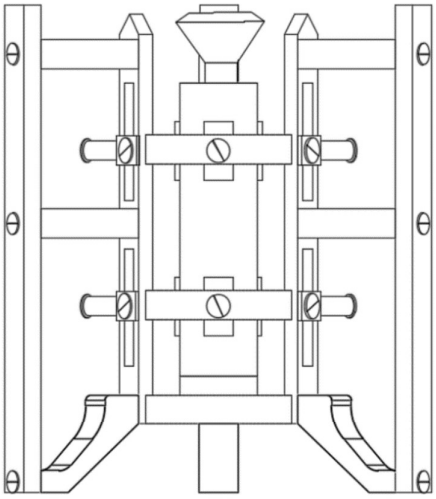
Some of the parameters that are recorded in real time using the rocket motor static test pad are :

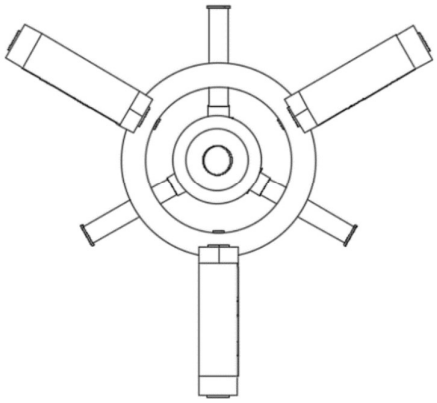
- Thrust
- Temperature
- Propellant flow gradient
- Pressure
- Burn Time
- Tank Pressure

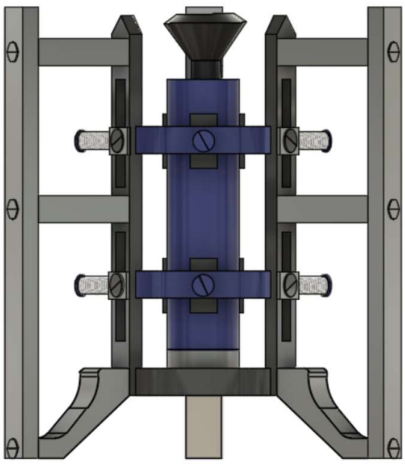
Architecture:

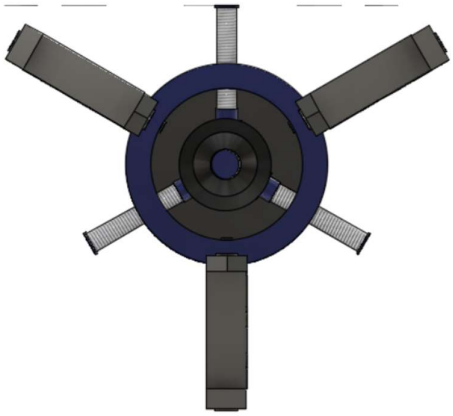
DESIGN

We built this whole thing as a cylinder. There are three square rods around it vertically and six rod horizontally .After that a round block is made on top of the loadcell will be located and motor will be installed on it .We can tighten and fix the motor with six screws on the slide and also to mount the motor the two rings can be slid upwards and downwards according to the length of the motor









AVIONICS

With respect to the avionics system, we wanted to design a system that could be controlled via a Bluetooth module and a GUI. The Avionics system created by us, consists of the following sensors :

- Load Cell : We used a Strain Gauge Load cell, because it is the most commonly used Load cell
- Amplifier : We used HX711, to amplify the output from the Load Cell for Arduino to process the thrust
- Temperature Sensor : We used LM35 to measure the temperature of the avionics bay
- Ignition System : Nichrome Wire and a relay switch is being used as the ignition system
- Backup Data Storage : We have used SD card as backup storage
- Bluetooth Module : We have used HCO5, to transmit data to the GUI, and receive commands from it.

Components and Softwares used to build a Rocket Motor Static Test Pad

Components

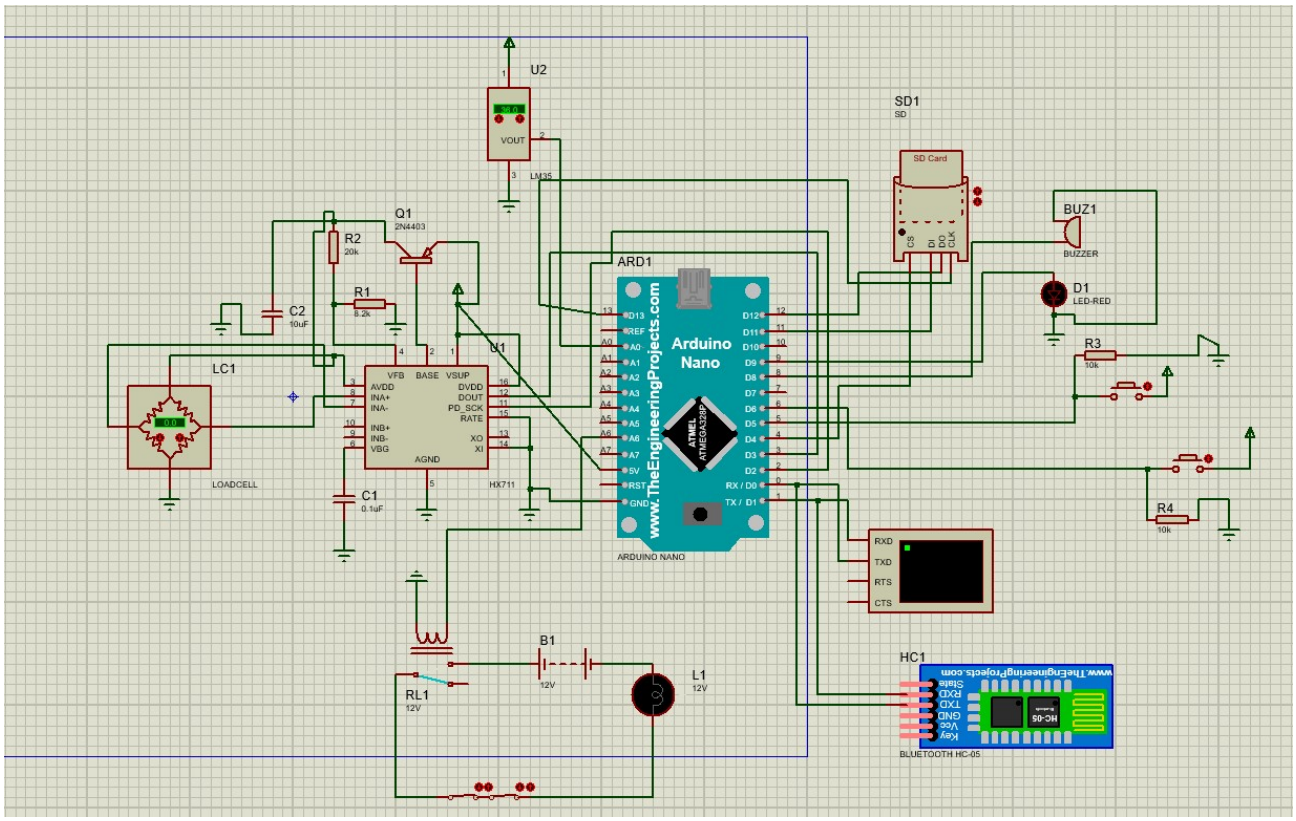
- Square steel bars
- Square steel rods
- M5, M6, M8, M9 bolts
- Rubber Blocks
- Relay
- Arduino Nano
- Load Cell
- Amplifier

- LM35 Temperature Sensor
- Backup Data Storage(SD Card)
- Bluetooth Module(HCO5)
- PCB
- Ignition System
- LED
- Buzzer

Softwares used

- Arduino IDE
- IntelliJ
- Proteus
- Eagle
- Fusion 360

CIRCUIT



CODE

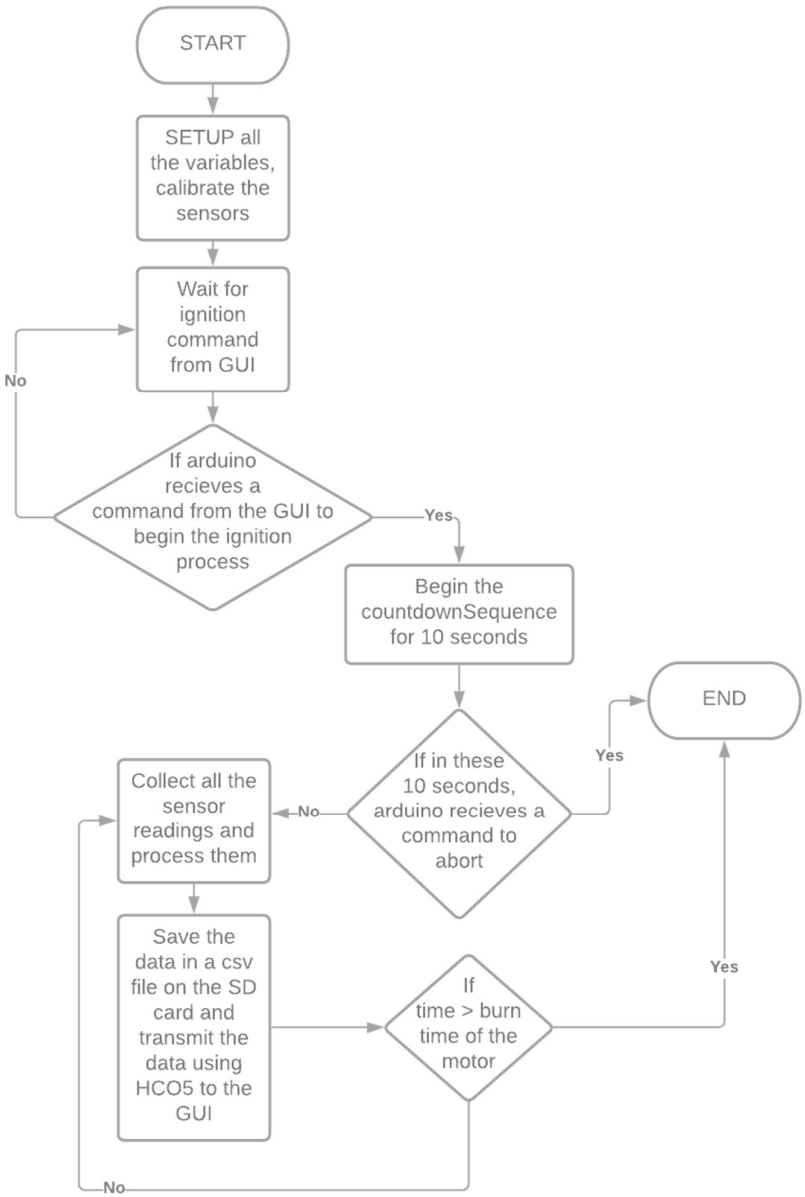
```
#include <SD.h>
#include <SPI.h>
#include "HX711.h"

#define DOUT  3
#define CLK  2

HX711 scale;

float calibration_factor = 352950.0; //97050 worked for my 440lb max scale setup

const int chipSelect = 4;

const int relay = 7;

const int abortbtn = 6;
int abortFlag = 0;

const int ignitionBtn = 5;
int ignitionFlag = 0;

int ledPin = 13;
int go = 0;

const int buzzer = 9;


const int lm35_pin = A0;  /* LM35 O/P pin */
int temp_adc_val;
float temp_val;


void setup() {
  Serial.begin(9600);

  pinMode(relay,OUTPUT);
  pinMode(ignitionBtn,INPUT);
  pinMode(abortbtn,INPUT);

  Serial.println("HX711 calibration sketch");
  Serial.println("Remove all weight from scale");
  Serial.println("After readings begin, place known weight on scale");
  Serial.println("Press + or a to increase calibration factor");
  Serial.println("Press - or z to decrease calibration factor");

  scale.begin(DOUT, CLK);
  scale.set_scale();
  scale.tare(); //Reset the scale to 0
```

```
  long zero_factor = scale.read_average(); //Get a baseline reading
  Serial.print("Zero factor: "); //This can be used to remove the need to tare the scale.
 Useful in permanent scale projects.
  Serial.println(zero_factor);
  pinMode(13,OUTPUT);


  Serial.print("Initializing SD card...");

  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    while (1);
  }
  Serial.println("card initialized.");

  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  pinMode(lm35_pin,INPUT);
  pinMode(buzzer,OUTPUT);

  String dataString;
  dataString = "Thrust,AVTemperature";
  File dataFile = SD.open("log.csv",FILE_WRITE);
    if(dataFile)
    {
      dataFile.println(dataString);
      dataFile.close();
      Serial.println(dataString);
    }
    else
    {
      Serial.println("Couldn't access file");
    }
}

int countDownSequence()
{
  Serial.println("CountDown Sequence Beginning");
  for(int i=10;i>=0;i--)
  {
    Serial.println("T-" + i);
    if(digitalRead(abortbtn) == HIGH)
    {
      abortFlag = 1;
      break;
    }
    digitalWrite(ledPin,HIGH);
    tone(buzzer,1000);
    delay(100);
```

```
    digitalWrite(ledPin,LOW);
    noTone(buzzer)
    delay(900);
  }
  if(abortFlag == 0)
  {
    Serial.println("IGNITION!!!");
    return 1;
  }
  else
  {
    Serial.println("ABORTED!!!!");
    return 0;
  }
}

void loop() {
  if(digitalRead(ignitionBtn) == HIGH)
  {
    ignitionFlag = 1;

    go = countDownSequence();
  }

  if(ignitionFlag == 1 && go == 1){


  temp_adc_val = analogRead(lm35_pin);  /* Read Temperature temp_adc_val=analog to digita
l converted value  */
  temp_val = (temp_adc_val * 4.88)/10;
  Serial.print("Temperature = ");
  Serial.print(temp_val);
  Serial.print(" Degree Celsius");
  Serial.println();

  scale.set_scale(calibration_factor); //Adjust to this calibration factor

  float p=scale.get_units()*9.8;
  scale.set_scale(calibration_factor); //Adjust to this calibration factor
  Serial.print("Thrust: ");
  Serial.print(p, 2);
  Serial.print(" Newton");
  Serial.println();


  String dataString;
  dataString = String(p,2) + "," + String(temp_val);
  File dataFile = SD.open("log.csv",FILE_WRITE);
    if(dataFile)
    {
```

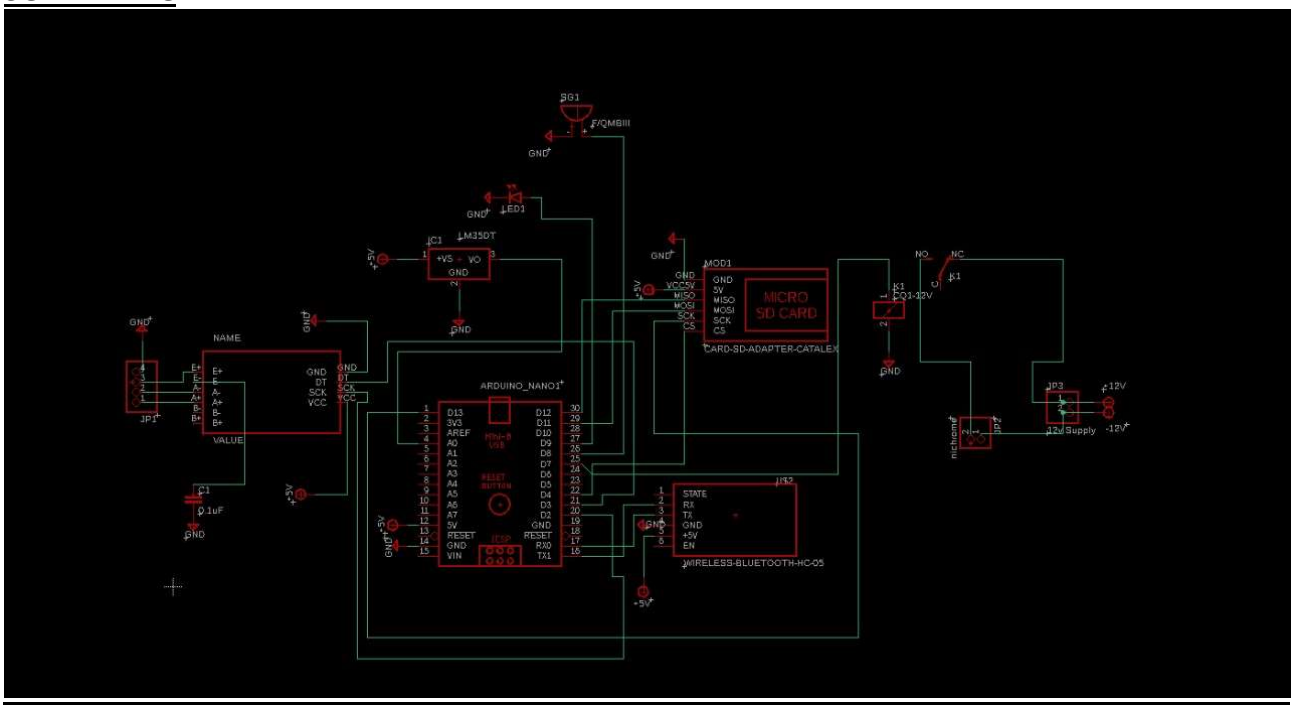
```
      dataFile.println(dataString);
      dataFile.close();
      Serial.println(dataString);
    }
    else
    {
      Serial.println("Couldn't access file");
    }
  delay(1000);

}
```

FLOWCHART

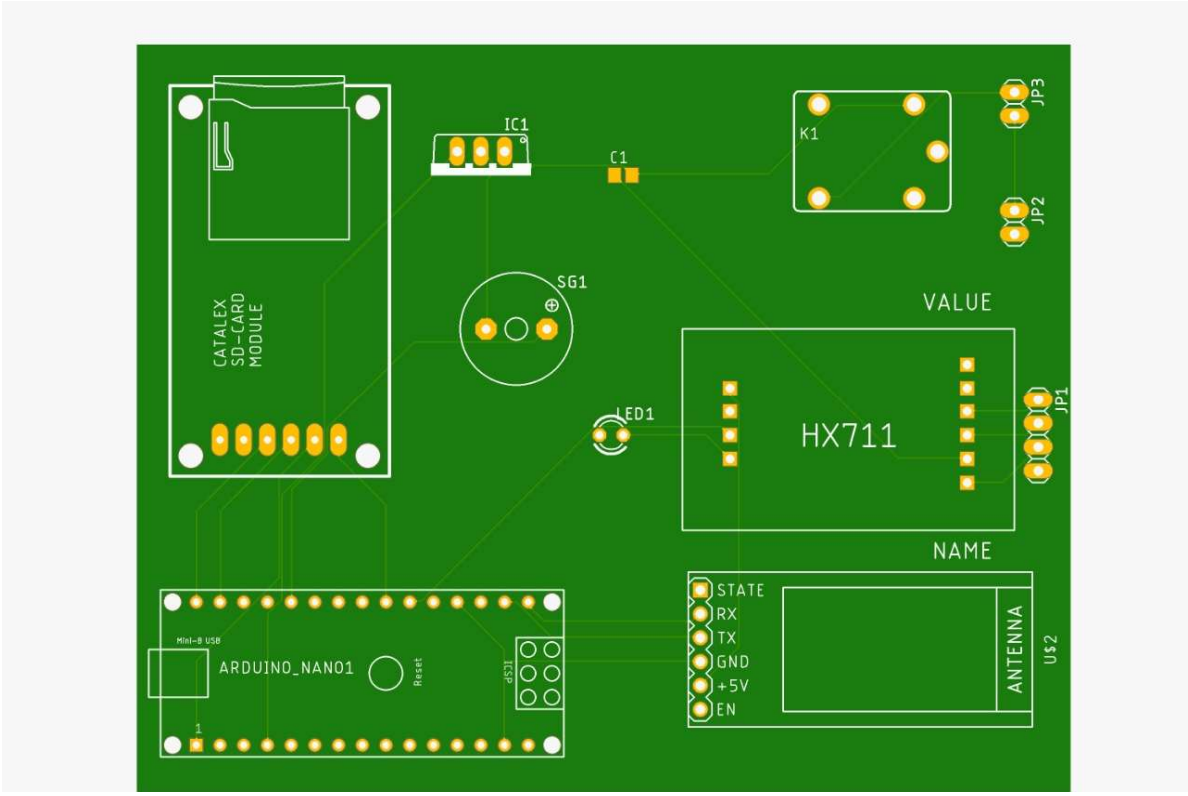


PCB DESIGN

SCHEMATIC



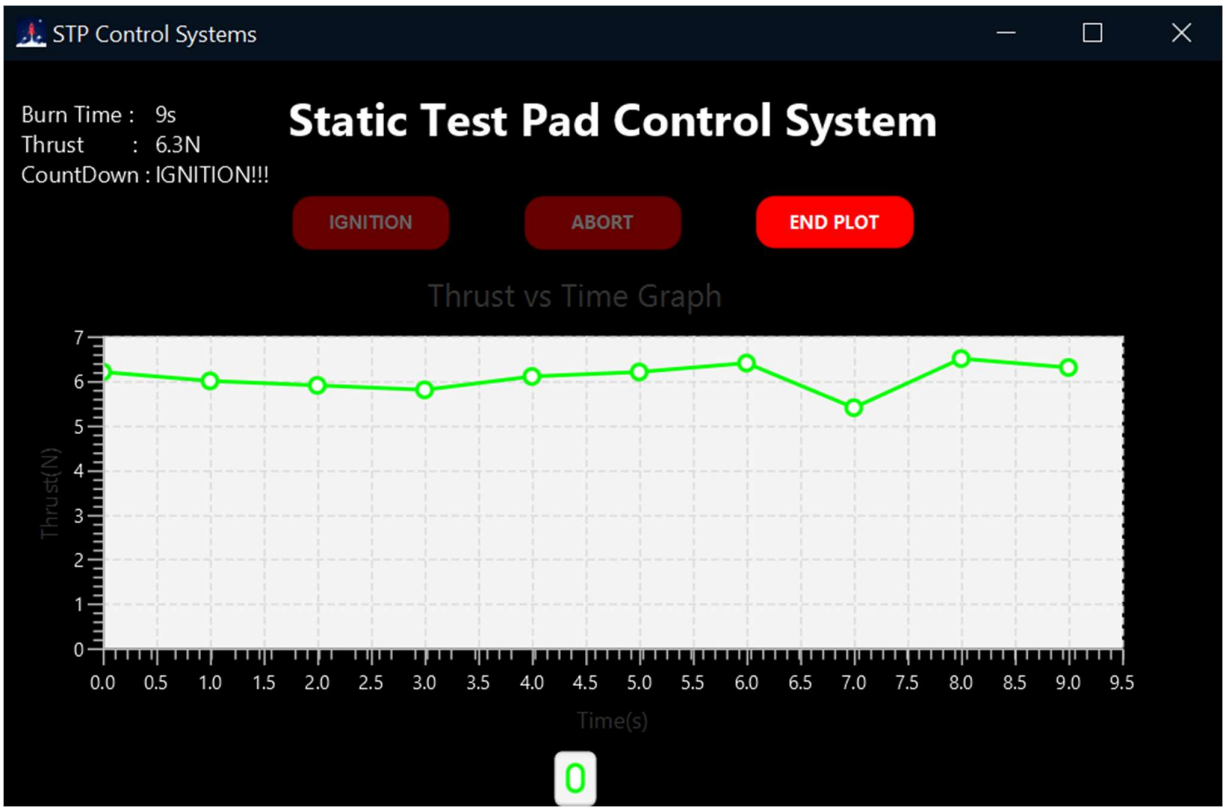
FINAL PCB



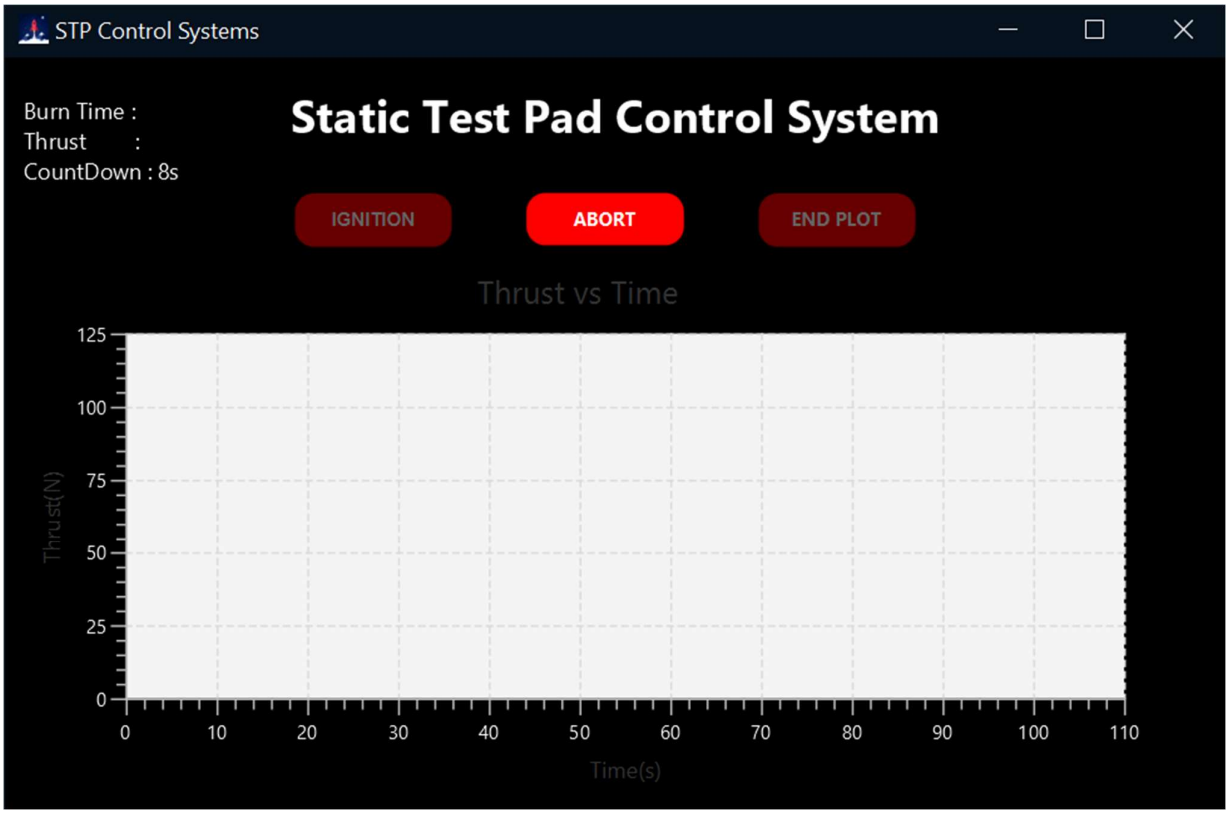
GUI

The GUI consists of the following features :-

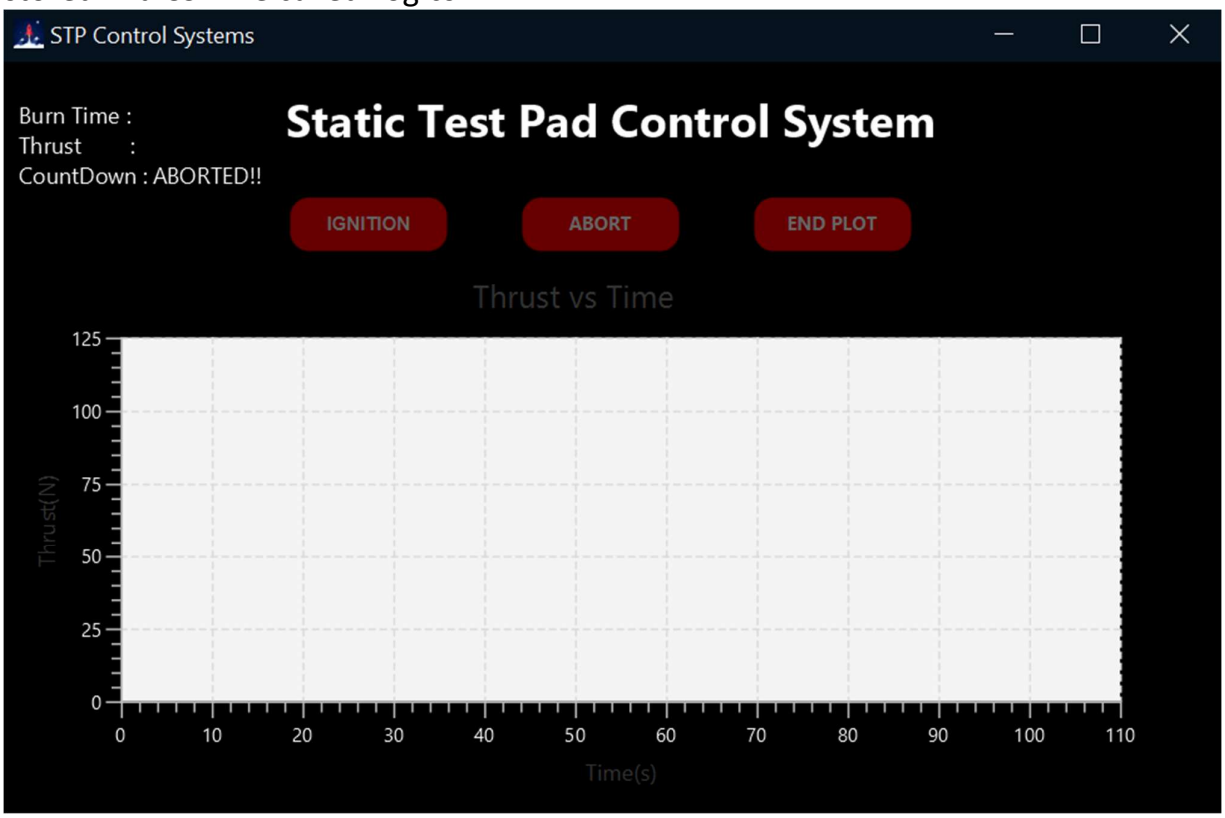
- IGNITION BUTTON
- ABORT BUTTON
- END PLOT
- LINE CHART
- BURN TIME
- THRUST VALUE
- COUNTDOWN



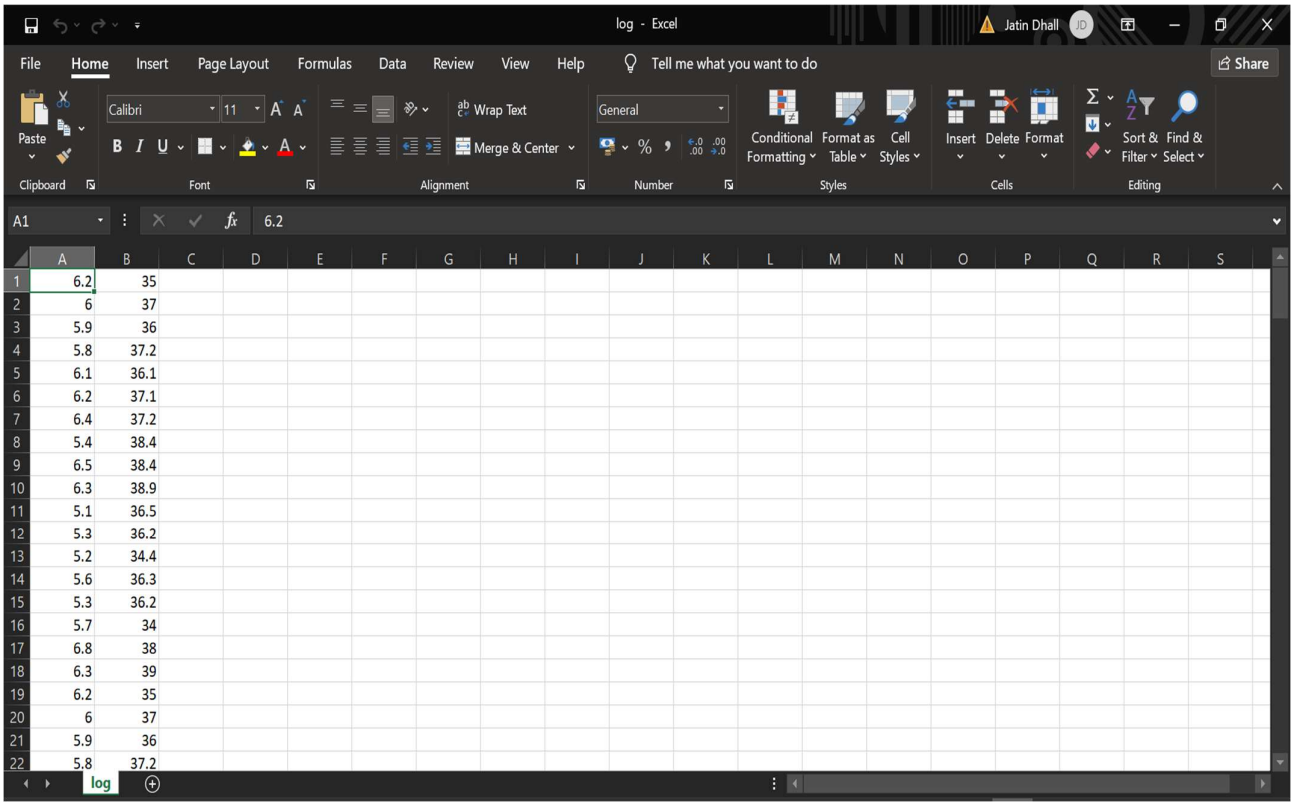
The way, the GUI works is, once the IGNITION button is pressed, the HCO5 library used in the GUI code will send a command to the Avionics system to ignite. There will be a 10 second countdown. If in these 10 seconds, the abort button is pressed, then the GUI will send a command to the Avionics system to abort the test. At any point if we wish to end plotting, then the End Plot button can be pressed for it.



As soon as the Count down sequence is over and if the abort button is not pressed, then the Line chart will receive real time data values from the Avionics system to be plotted and stored in a CSV file called Log.csv .



As soon as the data values stop being received by the GUI, we consider the static test fire to be over, and the Line chart will stop plotting.



	A	B
1	6.2	35
2	6	37
3	5.9	36
4	5.8	37.2
5	6.1	36.1
6	6.2	37.1
7	6.4	37.2
8	5.4	38.4
9	6.5	38.4
10	6.3	38.9
11	5.1	36.5
12	5.3	36.2
13	5.2	34.4
14	5.6	36.3
15	5.3	36.2
16	5.7	34
17	6.8	38
18	6.3	39
19	6.2	35
20	6	37
21	5.9	36
22	5.8	37.2

Procedure to build a Rocket Motor Static Test Pad:

To build a Rocket Motor Static Test Pad, we need to take into consideration, both the design aspect as well as the avionics aspect.

DESIGN

While designing the rocket motor static test pad, we need :-

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

AVIONICS

While designing the avionics system, we need :-

- 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 analyse data

Observation:

After conducting various simulations of the Static test pad design, Avionics System and the GUI, our observations were that :-

- The static test pad has successfully passed all the simulation tests
- The Proteus simulations passed all tests
- The GUI works well and data is being stored for further analysis

Results:

This report has detailed the avionics and GUI systems followed in designing the Static Test Pad. It has described each of the sensors used in the avionics system and their working as well as the working of the GUI. The simulations that have been conducted by us, ensure that the avionics system as well as the design system of the test pad will work well.

Conclusion:

The efforts of the design and avionics team has helped us in creating a great Static Test Pad for testing High-Powered Rocket Motors that can measure all the Data parameters required to test the feasibility of the rocket motor in a real launch. I would like to thank all my team members for working hard to come up with this product and to all the shapers at STAR Labs,Surat,India.

Precautions:

The precautions that are needed to be taken while using this product are :-

- One must stay atleast 10 to 15 meters away from the test pad
- The simulations that have been conducted, are all virtual so there could be certain changes that might be needed to be made in the real product