

RECOVERY TEAM PRESENTATION (B2)

WEEK 4



TASKS ACCOMPLISHED

Developed and implemented user interface for remote ignition

Carried out piston test

Research on filters

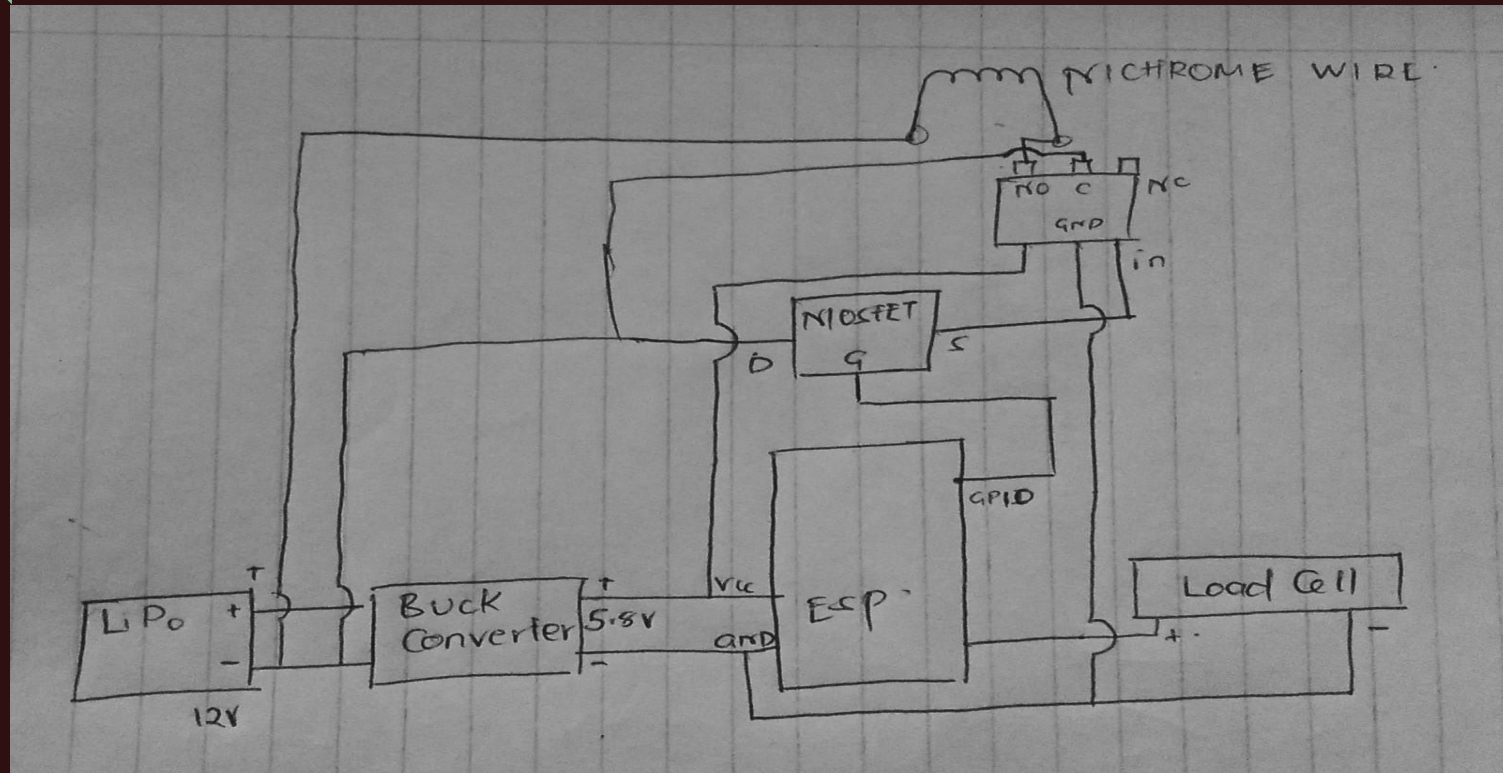
Solved the issue of burning of Esps



PISTON TEST USING RELAY

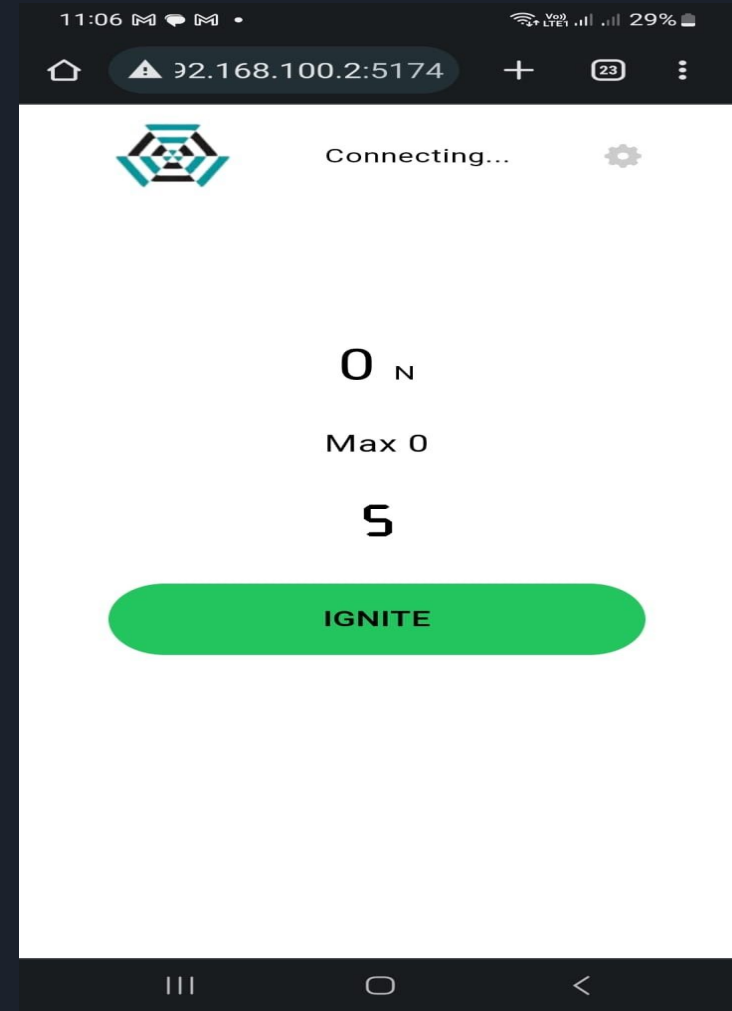
Implemented the use of relay together with mosfets which is efficient for high voltage switching applications.

PISTON TEST BLOCK DIAGRAM CIRCUIT



UI FOR IGNITION

Developed a webpage to
communicate with the esp
via wifi





USER INTERFACE DEMO

<https://drive.google.com/file/d/10D9IDH8CkzcVPSJ0idOOQsX3SLxiniU5t/view?usp=drivesdk>



PISTON TEST

- Was not successful at first due to recoil of the housing- the piston did not deploy with enough force.
- <https://drive.google.com/file/d/1-5AOoIMS5Q6KSJB2wUa-alfdwCPcQHC/view?usp=drivesdk>
- Mechanism to counter the recoil was implemented.
- Test was to be done with incrementing amounts of crimson powder from 3 grams
- Readings were too small— 10N for 3 grams of crimson powder
 - 1N for 5 grams of crimson powder
- https://drive.google.com/file/d/1-6YCFqJYnkgNBJ39D-n_E0FPGy12FGK2/view?usp=drivesdk
- The force exerted by the piston acted for a very short period of time— load cell could not record a reasonable amount of force
- Find a reliable method to measure the amount of force needed to eject deploy parachute.



GPS MODULE

GPS module is set.

We will use it together with the map application in the base station repository to locate the rocket after it lands.

Also can be implemented to detect velocity.

GPS_module


```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <ArduinoJson.hpp>
#include <WiFi.h>
#include <Wire.h>
#include <PubSubClient.h>

#define RX 16
#define TX 17
static const uint32_t GPSbaud = 9600;
const char* ssid = "Nezuko";
const char* password = "NezukoChan542";
const char* mqtt_serv = "192.168.0.102";

HardwareSerial hard(2);
TinyGPSPlus gps;
WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200);
    hard.begin(GPSbaud, SERIAL_8N1, RX, TX);
    Serial.println();
    Serial.print("connecting to: ");
    Serial.println(ssid);
    // Connecting to WiFi
    WiFi.begin(ssid, password);
```


FILTERS

- 
- Researched on Kalman, Bayesian and complementary filters
 - Complementary filter works with two sensors by applying high pass filter in one and low pass filter on the other
 - Less processor intensive
 - Faster estimates of angle
 - Simple to implement
 - Can not handle more than two filters
-
- Bayesian filtering applies Bayes rule- uses probability to filter out values
 - Allows robots to update their most likely position based on the most recent data from sensors



KALMAN FILTER

Kalman filter- more effective at filtering and providing accurate estimates

- Takes into account the physical properties of the system
- Difficult to code but possible
- Would kill processor time
- Settled on kalman filter



SET BACKS

inconclusive results from the piston test

Issues with labeling of footprints in the original pcb design- has to be redone (including routing)



Next week's tasks

- Design and implementation of another test to determine amount of force to push the nose cone
- Correction of pcb and etching
- 3d printing of the piston
- Writing and testing code for kalman filter