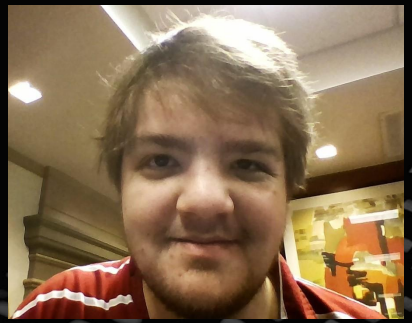
OSU AIAA

Oregon State University Chapter of the American Institute of Aeronautics and Astronautics

Team 41
Winter Progress Report



30k Spaceport America Cup 2017-18

Oregon State UNIVERSITY

Overview

Software and ground station components for the 2018 Spaceport America Cup 30K Challenge

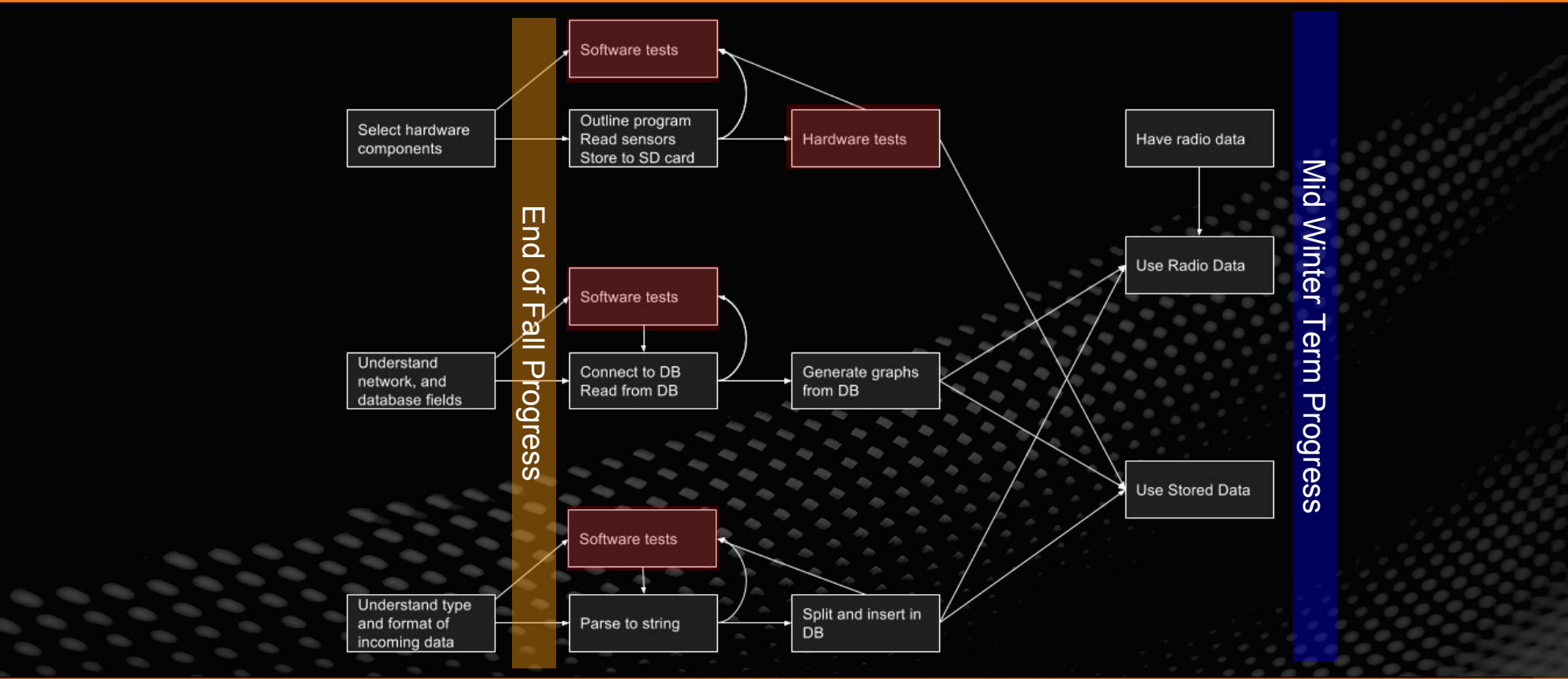
Our mission is to write flight avionics for both a rocket and scientific payload, as well as design a ground station capable of receiving and displaying live telemetry data from the rocket.

Project Goals

This project can be (greatly) simplified to three primary goals:

1. Design a ground station to receive and display flight data.
2. Write payload avionics to control a scientific experiment.
3. Write rocket avionics to log kinematics and detect apogee.

Development Roadmap



Ground Station

1. Physical build and components
2. Raspberry Pi 3B:
    a. Create ad-hoc WiFi network
    b. Host MariaDB database
    c. Serve NodeJS website with flight data
3. Raspberry Pi Zero (4):
    a. Parse incoming telemetry data

Ground Station

The physical build includes:

1 - 8”x10”x6” yellow case

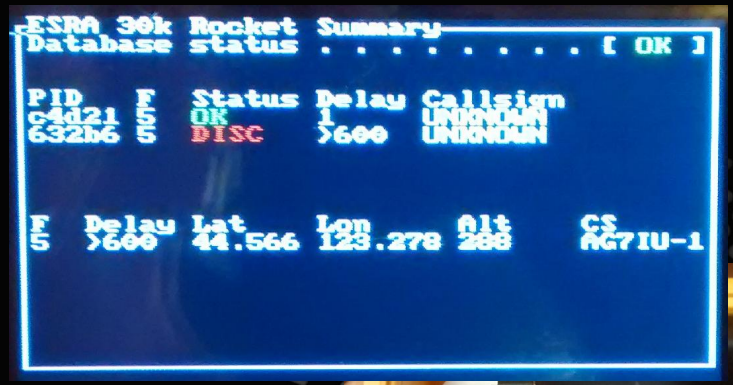
1 - Raspberry Pi B3

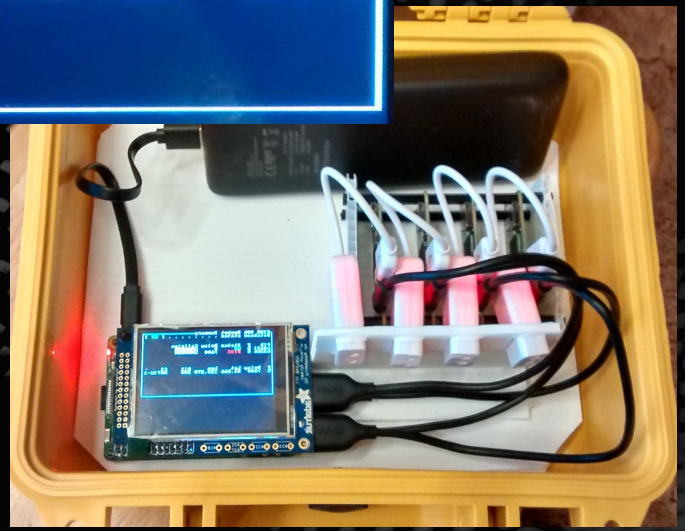
4 - Raspberry Pi Zero W

4 - USB sound cards

1 - 2.7” LCD Display

1 - 22,000 mAh rechargeable battery



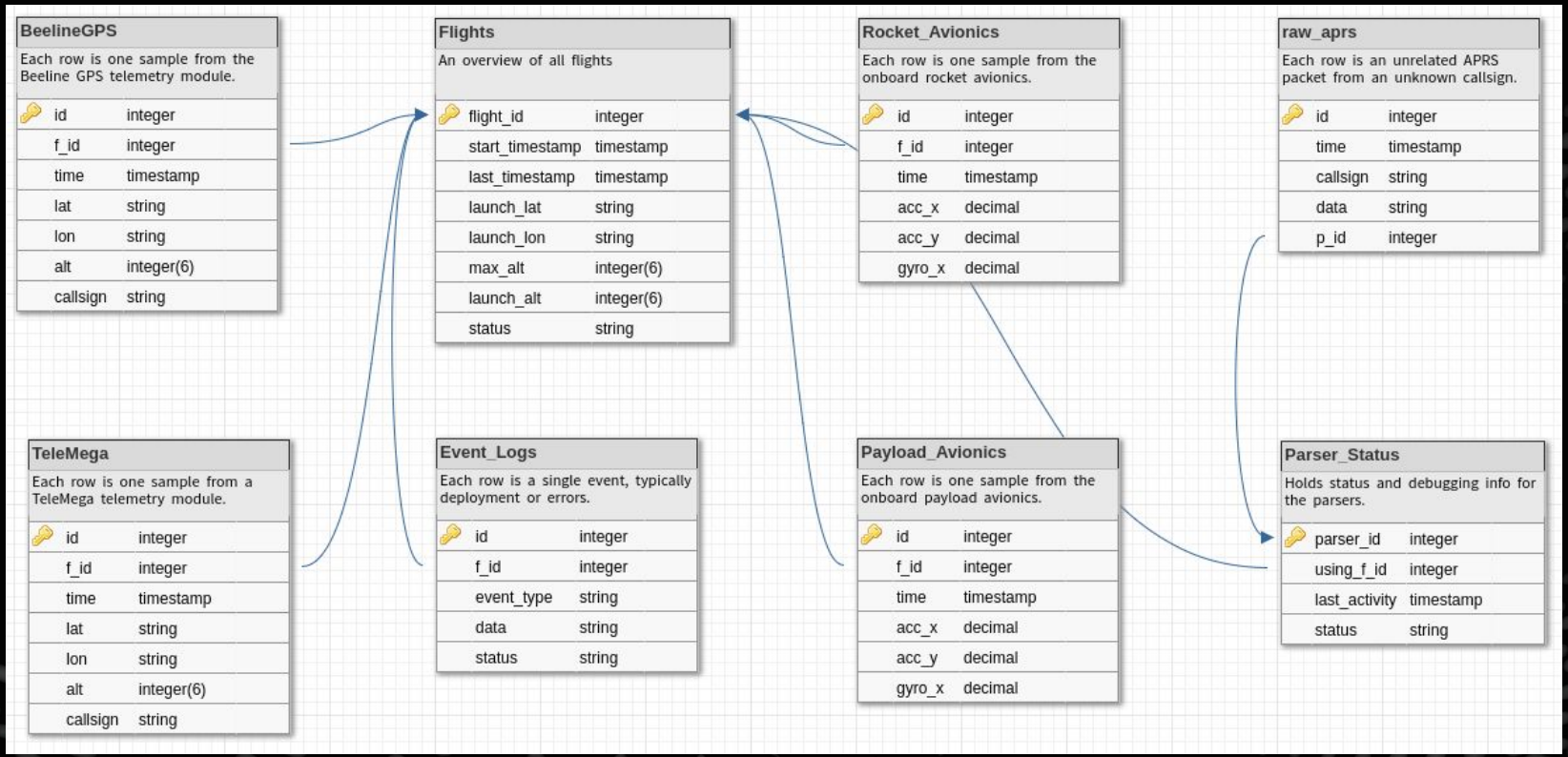


Ground Station - Database

The MariaDB database stores data from sources including:

- Flight logs from both avionics programs
- BeelineGPS and TeleMega telemetry units
- Parsing computers
- Error logs

Ground Station - Database



Ground Station - Parsers

The ground station includes 4 Raspberry Pi Zeros:

- Each Zero listens to a single radio frequency
- Completed one end-to-end test using radio TX and RX
- Range tests performed using audio recorded by ECE subteam
- Performance comparable to COTS hardware decoders

Ground Station - Parsers

- All Zeros use identical SD cards
- The cpu serial number uniquely identifies each Zero in the DB
- All Zeros run headless, the parsing program is a daemon
- Minimal text output, writes directly to database

Ground Station - Networking

Networking:

- Using USB OTG to connect the Pi zeros to the main Pi 3B
- Main Raspberry Pi serves a Wi-Fi Network
- Works for testing
- Stress test revealed issues

Ground Station - NodeJS

NodeJS website:

- Served by our main Raspberry pi 3
- A GUI for our graphs and flight data
- Implemented using express-handlebars
- Makes frequent queries to the database

Ground Station - Display

Graphs Overview

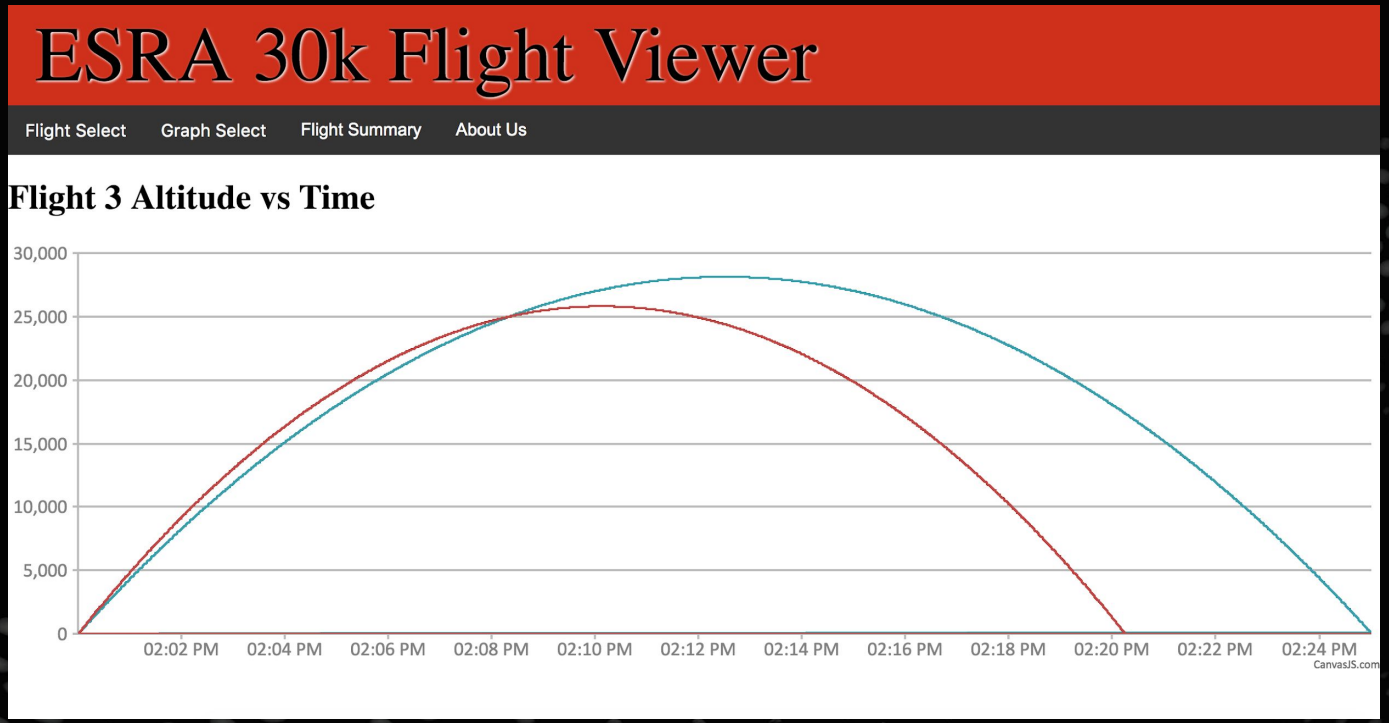
- Query database
- Use CanvasJS
- Update every second
- Can take inputs from numerous sources

Ground Station - Display

Line Graphs

- Altitude vs Time and Vertical Velocity
- Use CanvasJS
- Use any number of sources
- Based on Flight ID

Ground Station - Display

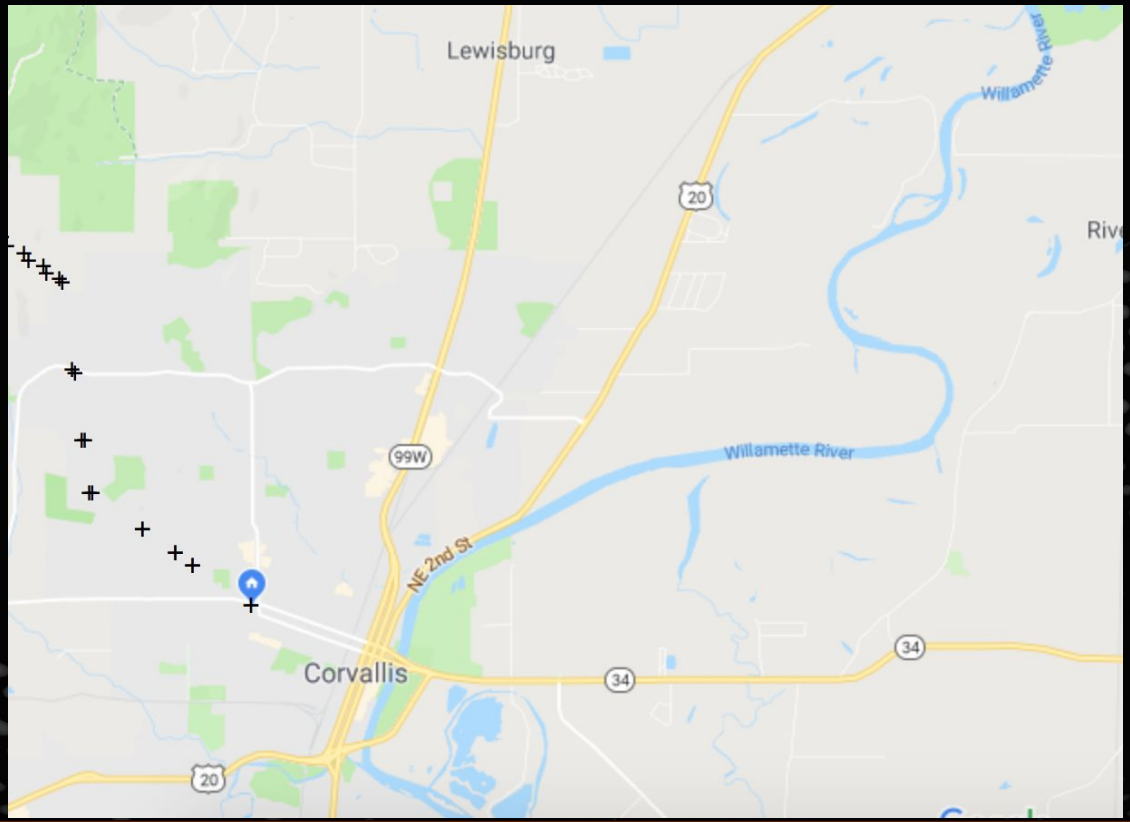


Ground Station - Display

Map

- GPS coordinate
- Calculate location based on map
- Any number of sources
- Based on Flight ID

Ground Station - Display



Ground Station - Display

Planned Features

- Use image for map
- Change color for map
- Acceleration graph
- True velocity/acceleration graphs
- Altitude or time in map

Sensor Avionics

Sensors Currently Implemented

- MPU - Accelerometer
- BMP - Altimeter (will not be used)
- MPL - Altimeter
- HMC - Magnetometer (may not be utilized)

Not Implemented

- Real Time Clock

Sensor Avionics

Speed Issues

- Bus is a significant speed barrier, language is not
- Most sensor update at less than 1khz
- Writing speed not heavily affected by opening and closing, further testing may be done

Payload Avionics

Our goal is to create 10-12 seconds of zero acceleration for a scientific experiment inside the payload.

Payload avionics record sensors on the payload as it descends.

10” propeller and motor will reduce drag from air resistance.

Counterweight motor will reduce rotational forces.

Payload Avionics

The avionics loop currently:

- Reads 22 values from several sensors
- Calculates motor speed using PID loop
- Controls motor with a PWM output
- Logs data to CSV file

Rocket Avionics

Overview

- Very similar implementation to payload avionics
- Read and log sensor values
- Detect apogee

Rocket Avionics

Apogee Detection

- Mission critical task
- Can’t detect too early or too late
- Not being used for competition

Rocket Avionics

Apogee Sensors

- Magnetometer is common in commercial applications
- Possible Interference from metal
- Using altitude and acceleration data together will be more reliable

Rocket Avionics

If (total acceleration <= somewhat close to zero g's)

Start a countdown timer. If apogee hasn't been logged when it goes off, log apogee event.

If (total acceleration <= very close to zero g's)

Start a much shorter countdown timer. If apogee hasn't been logged when it goes off, log apogee event.

If (altitude begins to decrease)

Log apogee event

If (total acceleration hits one of our thresholds and then starts to increase again)

Apogee has occurred. If it hasn't been logged, log it now.

Conclusions