Design, Development, and Implementation Of The 2018 RockSat-C Data Management System

A Senior Project Submitted to the

Department of Computer Systems Engineering Technology

of the School of Engineering, Technology, and Management at the Oregon Institute of Technology

in partial fulfillment of the requirements for the Degree of

**Bachelor of Science**

Steven Reeves

© June 2018

**Senior Project Approval Page**

The senior project of Steven Reeves for the Bachelor of Science degree was accepted by the evaluation committee and the Department of Computer Systems Engineering Technology at the Oregon Institute of Technology.

COMMITTEE APPROVALS:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Sherry Yang, Wilsonville Program Director (Oregon Institute of Technology)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Reviewer 2 (Oregon Institute of Technology)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Reviewer 3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Reviewer 4

**abstract**

RockSat-C is a program for students from all over the U.S. to design, build, and launch a sounding rocket payload out of NASA’s Wallops Flight Facility. A team of 10 undergraduates here at Oregon Tech designed and proposed this project in fall 2017. Oregon Tech’s proposal was accepted in January 2018 as one of 9 schools across the nation approved for launch. The primary mission is to design a complex, multi-experiment payload emulating the methodology of a satellite or other space vehicle. Mission success is defined as all systems and modular payloads functioning and collecting data.

This project is one of the experiments onboard the Oregon Tech RockSat-C mission. The objective of this project is to implement communication and data storage protocols for other modular experiments onboard. Four modular experiments will be connected to the Data Management System and data will be written to an SD card for each individual line of communication.

The second, off-board part of this project is a user interface to allow further analyzing of the collected data after launch. This program allows the user the ability to open and view different file types via a standalone Windows application.

Mission success of the Data Management System is defined as data being recorded and written to a local SD card during flight time and displayed with the user interface after recovery. Multiple full mission simulations have confirmed this functionality. Unfortunately, launch is scheduled for June 21, 2018, so this document will need to be updated with flight data.

**acknowledgements**

I would like to provide acknowledgements for all the faculty at Oregon Tech for providing an environment that leads to such creative and technical advancement. In particular, I would like to thank Sherry Yang for the guidance as I navigated my way through such an intense project. I’d also like to thank the other students that were a part of RockSat-C; Krystal Cruz, Wilson Davenport, Diego Garrido-Mendoza, Zach Hofmann, Andrew Horn, Caleb Ives, Chris Love, Jean-Luce Nabors, Thomas Pearce, and Jack Thomas. Additionally, I’m incredibly grateful to the faculty that stepped up to make sure RockSat-C could happen. Without the dedication of Andria Fultz and Dawn LoweWincentsen, we wouldn’t have had the means to make such a project happen. Finally, I want to thank everyone at the NASA Wallops Flight Facility for offering such an incredible opportunity for students all over the country. Emily Logan, Chris Koehler, and Audrey Viland deserve a special recognition for all the work they’ve done to connect NASA with students.

**List of Acronyms**

**Make sure this is in alphabetical order:**

**EDL** Environmental Data Logger

**FOG** Fiber Optic Gyroscope

**DMS** Data Management System

**IC** Integrated Circuit

**MCU** Microcontroller Unit

**MEH** Mesa Energy Harvester

**OS** Operating System

**REH** Rocket Energy Harvester

**RSE** Radiation Shielding Experiment

**SD** Secure Digital

**UI** User Interface

**USB** Universal Serial Bus

**WFF** Wallops Flight Facility

**Summary Table of Contents**

**Check to make sure these page numbers are correct:**

Chapter 1 Introduction 8

Chapter 2 Background 9

Chapter 3 Functional Description 9

Chapter 4 Detailed Description 11

Chapter 5 Test Results & Validation 12

Chapter 6 Economic and IP Analysis 13

Chapter 7 Summary 14

**Table of Contents**

**Check to make sure these page numbers are correct:**

Chapter 1 Introduction 8

Overview 8

Product Description 8

Existing Products 8

Summary 8

Report Outline 8

Chapter 2 Background 9

Overview 9

Background Information 9

State of the Art 9

Summary 9

Chapter 3 Functional Description 10

Overview 10

Functional Description 10

Block Diagram 10

Summary 10

Chapter 4 Detailed Description 11

Overview 11

Detailed Description 11

Architecture 11

Summary 11

Chapter 5 Test Results & Validation 12

Overview 12

Test Plan 12

Test Results 12

Discussion 12

Summary 12

Chapter 6 Economic and IP Analysis 13

Overview 13

Engineering Economic Analysis 13

Intellectual Property Analysis 13

Summary 13

Chapter 7 Summary 14

Overview 14

Project Summary 14

Future Direction 14

Concluding Remarks 14

# Introduction

## Includes a short pitch as to why your project has merit and should be acknowledged. Describe the purpose of the Software and its intended audience

## Overview

This chapter defines the problems to be addressed with this project. A project description, comparison to existing projects, and a summary are also included. Finally, an outline of the rest of the report is also included.

## Product Description (2 pages?)

The idea for this project is to make data management system for an on-board microcontroller. This microcontroller will be connected to four modular experiments. The entire system will be launched on a Sounding Rocket from the NASA Wallops Flight Facility in June 2018. The data management system will be responsible for monitoring all experiments. Furthermore, the data from each module will be written on an SD card to be examined later. Extensive error/bounds checking will be a big part of this data management system as well.

The second part of this project is an off-board user interface to allow users to view data after launch/retrieval. This user interface will be completely separate from the main system and will operate independently.

The project is responsible for the following tasks:

1. Module connections
   1. Recognize and connect to modules via USB
   2. Recognize and reconnect to disconnected modules
2. Data input streams
   1. Monitor data from different modules
   2. Error/bound check all data
   3. Organize and record data to SD card
   4. Shut off collection if one should fail
3. UI
   1. Display data for each experiment
   2. Display .csv and .txt files
4. Initiation
   1. Initiate all data collection on startup
   2. Log errors
   3. Timestamp log messages
   4. End data collection after allotted flight time
5. Standby mode
   1. Wait for module connections

## Existing Products

A similar system was launched on the 2017 RockSat-C mission. An excerpt from the “RockSat-C 2017 Final Report” is below:

“Although a Raspberry Pi had initially been considered as the primary controller for the payload, the Arduino Mega was eventually chosen. This was due to its relative simplicity, ease of use, and number of available I/O pins. Additionally, previous experience working with the Arduino platform would allow for less of a learning curve. Additionally, the large catalog of off the shelf components specifically designed to work with the Arduino platform made initial prototyping far easier. …At its core, this payload is essentially a large data logger. Additionally, many of the subsystems have readily available libraries and example code for reference.”

The main difference is last year’s data management and this year’s is the number of modules collecting data. RockSat-C 2018 has a total of seven modules recording data. The microcontroller for this project is the Raspberry Pi Zero W, which is different than the MCU flown last year.

Unfortunately, due to the nature of this project, there aren’t existing products to compare to.

## Summary

RockSat-C 2018 has a goal of creating a universal mounting system for student capstone projects that would benefit from hostile environment testing. This modular system will help maximize the availability of functional experiments within the payload volume.

Onboard experiments include:

**Environmental Data Logger** – Gathers data from the surrounding environment such as pressure, humidity, and temperature

**Fiber Optic Gyroscope** – Gathers gyroscopic data using a fiber optic cable

**Mesa Energy Harvester** – Uses piezo elements to produce electrical current

**Rocket Energy Harvester** – Uses piezo elements in different formations to produce electrical current

**Radiation Shielding Experiment** - Compares three types of shielding on Geiger–Müller tubes to test effectiveness of radiation deterrent

**Optic Window Camera** – Gathers images during flight

The DMS will act as a data redundancy mechanism for the four connected experiments on board. The data from these experiments is very valuable not only scientifically, but monetarily as well. Furthermore, Oregon Tech capstone projects are being flown on this mission, and the integrity of the data they return is valuable for the both the student and school itself.

## Report Outline

The following report is outlined as such:

* Background
  + Background information on project
  + State of the Art
* Functional Description
  + Functional requirements of the system
  + High level descriptions of how to fulfill requirements
* Detailed Description
  + Expanded look into high level requirements
  + Details on how requirements at this level were achieved
  + Reasoning of choices made
* Testing and Validation
  + Test plan and methodologies used during implementation
  + Test results and descriptions
  + Discussion on testing methodologies
* Economic and IP Analysis
  + Engineering economic analysis on cost of development
  + Intellectual property analysis on copyright effects
* Final Summary and Conclusion

# Background

## Overview

This chapter provides background information necessary to understand the project from both a value and technical perspective. State-of-the-art of the project area is discussed along with the current technologies being used in the industry.

## Background Information

RockSat-C is a follow-up program to the RockOn Workshop where customers design their own sounding rocket payload and compete for a spot on the launch vehicle. The RockOn Workshop teaches participants how to build a sounding rocket payload in three days. The goal is that participants will take what they learn during the workshop and return the next year with an original payload to fly with RockSat-C or RockSat-X programs.

The RockSat-C program is designed to provide students with access to low-cost flight opportunities and further develop their engineering skills. The RockSat-C flight is made possible through significant cost sharing provided by Wallops Flight Facility (WFF) and the launch fees paid by RockSat-C teams. A portion of the RockSat-C launch fees are invested back into the RockOn Workshop. The RockSat-C and RockOn Workshop canisters fly on the same rocket each year.

The program uses a modular canister system to allow for simple integration to the WFF Sub-SEM ring assembly. This standardized approach simplifies final integration and allows for more focus on the design of the payload. The organizers of the RockSat-C program guide the RockSat-C customers through the design process in the fall with multiple design reviews, leading projects to a Critical Design Review level design in December. Based on available space in the rocket, the most developed and capable projects are selected for flight in January. These projects then make their first payment and begin building. The projects continue to have subsystem and system testing reviews with the RockSat-C program manager through May. Any special requirements that arise for payloads are communicated to WFF through the RockSat program manager. The program culminates in June when the teams travel to WFF in Virginia for inspection, integration to the rocket, launch and recovery. The intent of the RockSat-C program is to provide hands-on experiences to students and faculty advisors to better equip them for supporting the future technical workforce needs of the United States and/or helping those students and faculty advisors become principal investigators on future NASA science missions. Therefore, RockSat-C is limited to U.S. educational institutions; only payloads from U.S. educational institutions are eligible to participate in the RockSat-C program. For the purpose of RockSat-C, ‘educational institution’ is defined broadly and includes, but is not limited to, the following: universities, colleges, technical schools, public and private high school, middle school and grade school, science museums, etc. Organizations which are not included in the above listing, are encouraged to contact Colorado Space Grant Consortium (COSGC) to clarify their eligibility in the program. In addition, U.S. entities (e.g. industry, research institutions, etc.) that fall outside of the eligibility conditions listed above, but that are interested in participating in the program, are encouraged to team with an eligible U.S. educational institution. Teaming between educational institutions and industry or other interests is allowed and encouraged.

## State of the Art

The RockSat program has been running for 10 years, and has launched many different individual experiments. Between RockOn, RockSat-C, and Rocksat-X there have been numerous experiments sent and received by the WFF itself. The technology used to encapsulate experiments has evolved over the years and has been tried and true at this point. Being that the RockSat program has had some time to evolve, the documentation and guidance provided (while minimal) was effective.

Sounding rockets themselves have been in use since as early as 1962 by the Andøya Space Center in Norway. NASA has been using sounding rockets for over 40 years, primarily for low gravity and material based research. This sounding rocket program is primarily used by universities for upper atmospheric research. The lower cost makes sounding rockets an attractive alternative as they do not need expensive boosters or extended telemetry. As a result, mission costs are substantially less than those required for orbiter missions. The sounding rocket program takes advantage of a high degree of commonality and in many cases, only the experiment provided by the designer is changed. In some cases (such as almost all astronomy, planetary, solar, and microgravity missions), the payloads are recovered which means the costs of the experiment and sub-systems are spread out over many missions.

Not only are sounding rocket missions carried out at very low cost, but also the payload can be developed in a very short time frame, sometimes as quickly as 3 months. This rapid response enables scientists to react quickly to new phenomena and to incorporate the latest, most up-to-date technology in their experiments.

The DMS is very particular to the modules it’s connected to, so systems like it are only vaguely similar. The individual components of the system have had quite some time to advance and they have a lot of documentation and implementation examples. MCUs, USB ICs, and SD cards have been around for a while and are still evolving today.

## Summary

This chapter provides some background of the RockSat program and canister system used for launch. The state-of-the-art in regards to the RockSat program, sounding rockets, and materials to build the DMS are discussed as well.

# Functional Description

## Overview

This chapter provides a functional description of both parts of the DMS. Included in this description are block diagrams of how each section of the DMS works at a high level. System requirements for both parts of the project are addressed as well.

## Functional Description

The main part of the DMS is the on-board electronics. This system is responsible for collecting data from 4 modules, separating the data and recording it to local storage in separate files. A Raspberry Pi Zero W is used for the MCU, while a USB hub is used to port all the connections. A 5v power regulator is used to maintain voltage from the power line while the Pi’s SD card stores the data. This system must treat each module independently and have the ability to reconnect if needed. This system is to run for a total of 25 minutes, with the first 3 minutes having no connection with any module. This is due to the flight schedule shown in figure X. System requirements for the Rasbian distribution of Linux that runs on the MCU are minimal.

The secondary part of the DMS is the user interface. This small executable file will search its working directory and give the user the ability to display .txt and .csv files. As the program implements fundamental functionality, system requirements are very low. Figure X shows the basic navigation flow through windows.

## Block Diagram

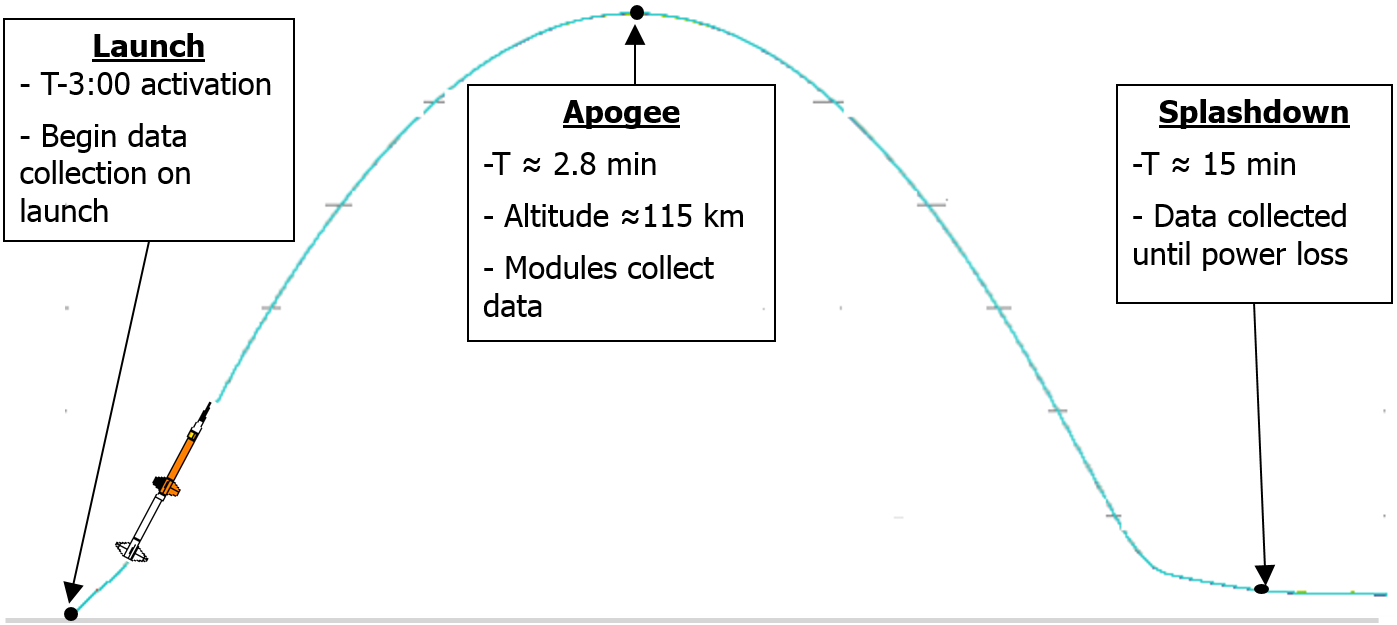


Figure X: Concept of operations

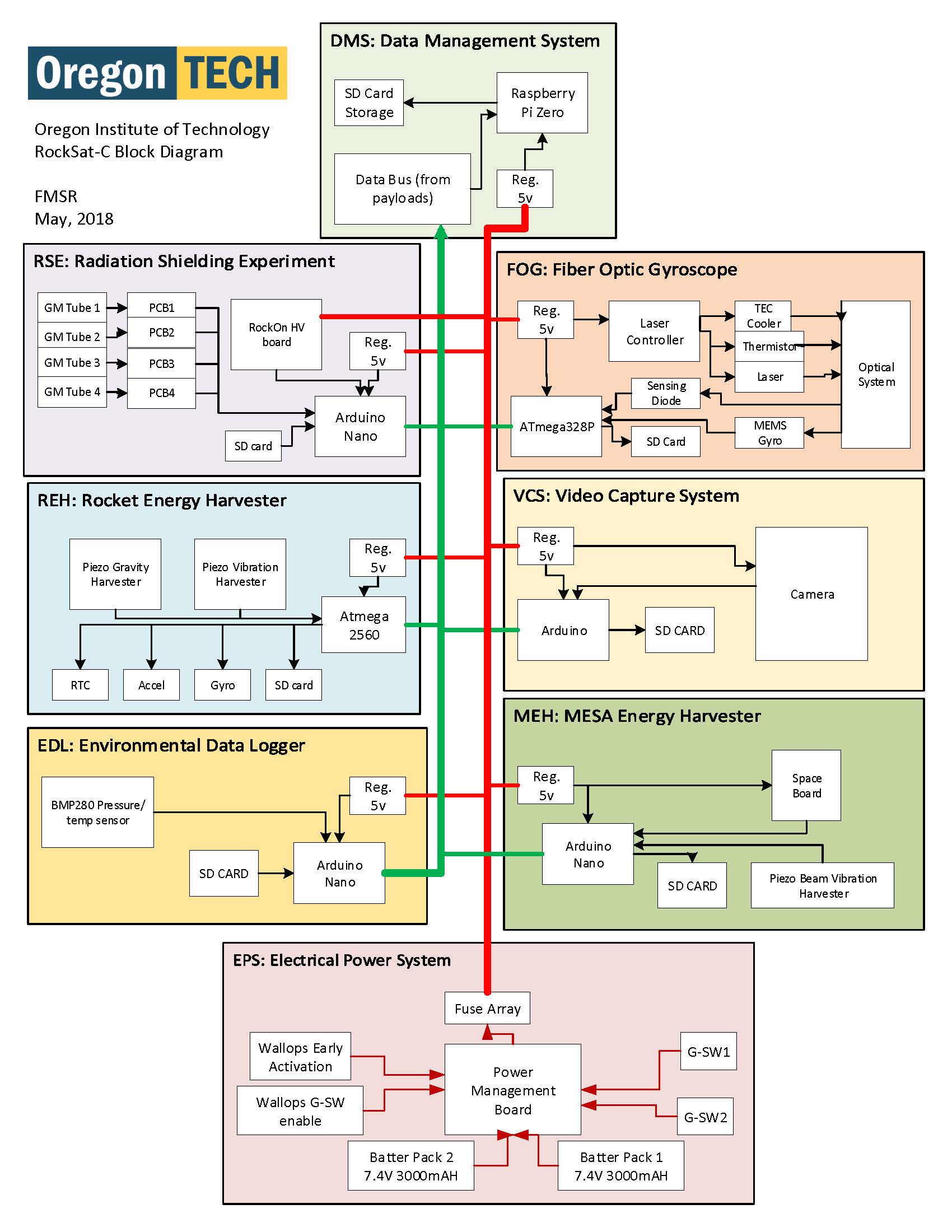


Figure X: RockSat-C 2018 block diagram

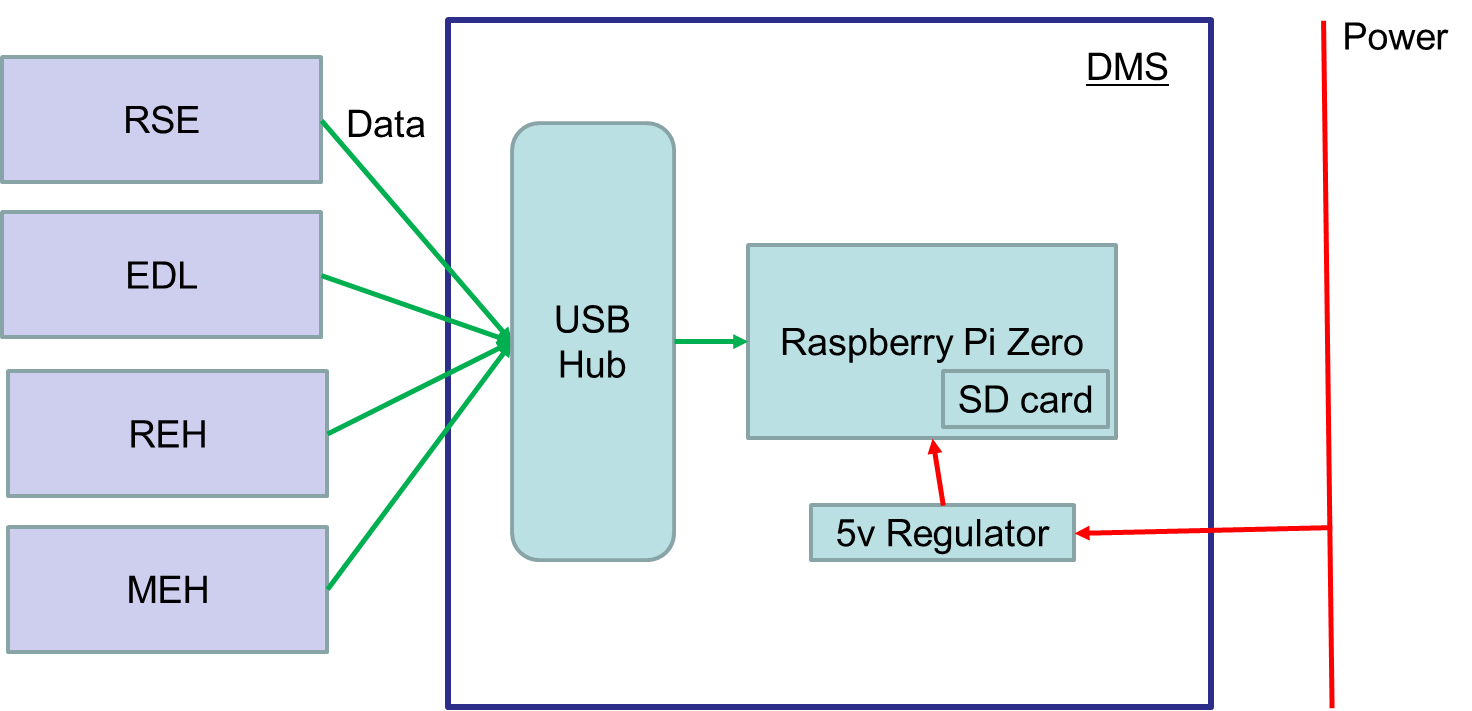


Figure X: DMS block diagram

Main page

Quick Start

Login

Sign Up

Data Display

Login Page

Sign Up Page

.txt Display

.csv Display

Figure X: DMS UI block diagram

## Summary

This chapter summarized the functionality of the DMS at a high level. Block diagrams show the general flow of information and organization of components for both the on-board and user interface sections. An overall block diagram of the entire RockSat-C mission shows the DMS’s place. System requirements, while minimal, are addressed and a basic timeline of launch is described.

# Detailed Description

## Overview

This chapter provides explicit information about the requirements for the project and how they were met. Major components and relationships are shown in this section. Data flow through specific sections are shown in figures below. Navigation of the UI is explained in this chapter as well. Technologies, protocols, error handling, initiation, and closure are all explained in this section.

## Detailed Description and Architecture

The data flow for the DMS begins at each modules serial out functionality. As each module collects data, it sends it out via the USB port on the MCU. Data then flows over the RX/TX lines in a USB cable until it gets to the USB hub. There the IC on the USB allocates a port to the line of communication and sends that data through another USB connection to the DMS MCU itself. There, the operating system (Raspbian Linux distribution) recognizes the USB connection. A Python program “main.py” will be running on the DMS’s MCU that will recognize this connection through the Pyserial library. A thread is allocated to each serial connection and data is parsed and written to individual .txt files.

The second part of data flow in this project is the user moving the SD card to another machine for inspection via the DMS UI. At this point, the DMS UI will open, allow the user to navigate to a “new data” page and pick the file to open. Once chosen, the user UI will load the data from the .txt or .csv file and display it in a new window.

**Module MCU:** Each MCU will differ based on its own functionality. The way it sends data out over USB is the part that pertains to the DMS.

Experiments

USB

RSE

Figure X: Data flow diagram of module MCU

**USB hub:** The USB hub uses and IC to route multiple connections to the DMS MCU via one USB port. An FE1.1s USB 2.0 High Speed 4-Port Hub Controller is used for this.

USB

USB hub

IC

USB

USB

USB

USB

Figure X: Data flow diagram of USB hub

**DMS MCU:** The DMS uses the integrated parts of a Raspberry Pi Zero W and the Rasbian distribution of Linux to receive the information coming in via USB. A python program runs within the Linux environment that deals with reading and writing accordingly.

Main.py: This program runs at startup of the operating system. It uses the DataThread function and starts a new function for 4 threads based on the port given by the USB hub and regulated baudrate (9600).

DataThread.py: This function waits for a serial connection to be successfully made, then converts the bytes into ascii characters to be written to a .txt file. This is where errors with connection are caught, and exceptions are dealt with. An additional thread is initiated to check all other thread’s health. If a modules disconnects, the thread tries to keep connecting until successful. The handshake protocol is encapsulated inside the Pyserial library. A timer is kept in this function for 1500 seconds or 25 minutes. After the timer is over, threads are closed and the program exits.

Main.py

DataThread functions

Linux OS

USB

.txt files

SD card

DMS MCU

Figure X: Data flow diagram of the DMS MCU

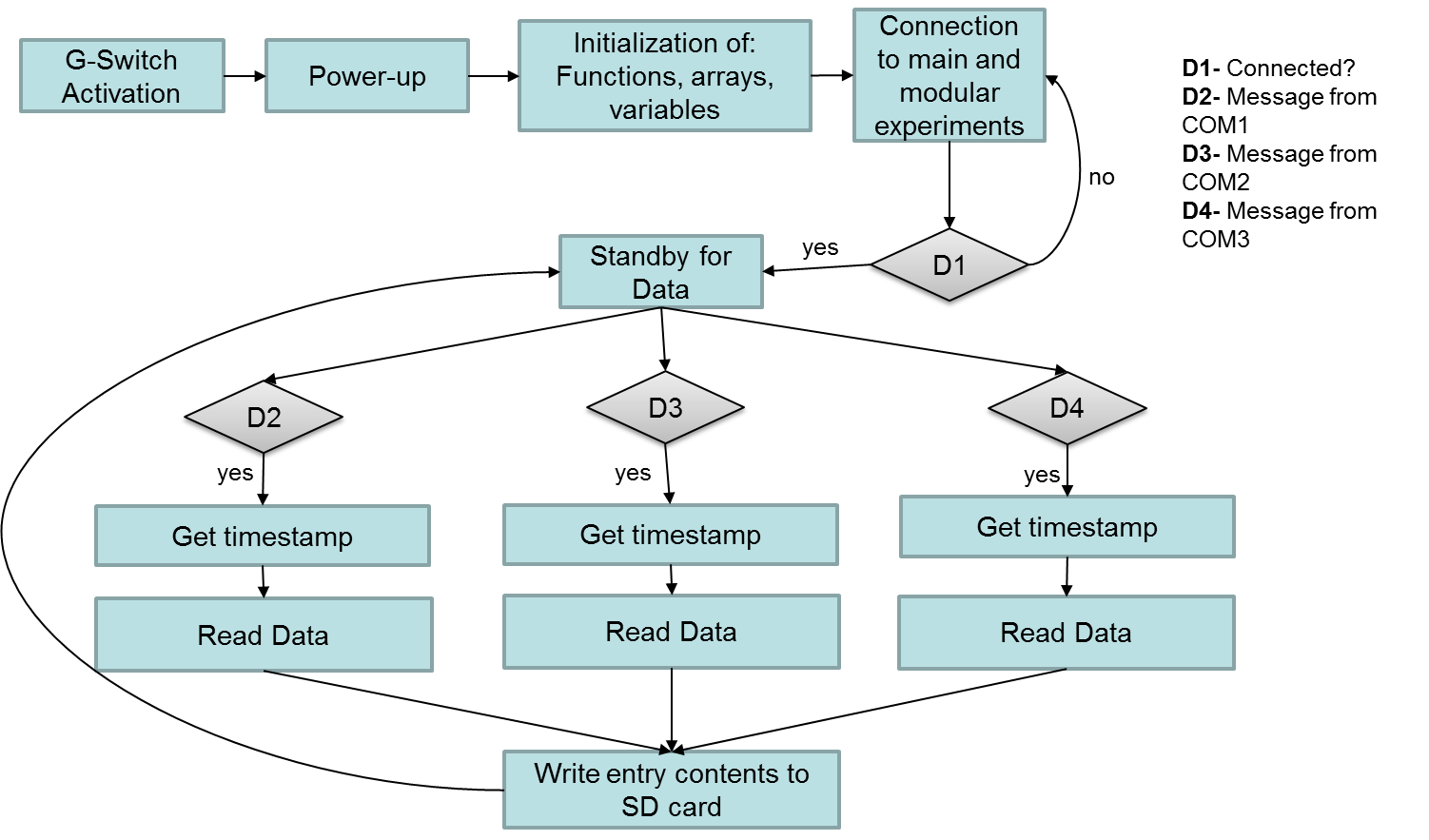
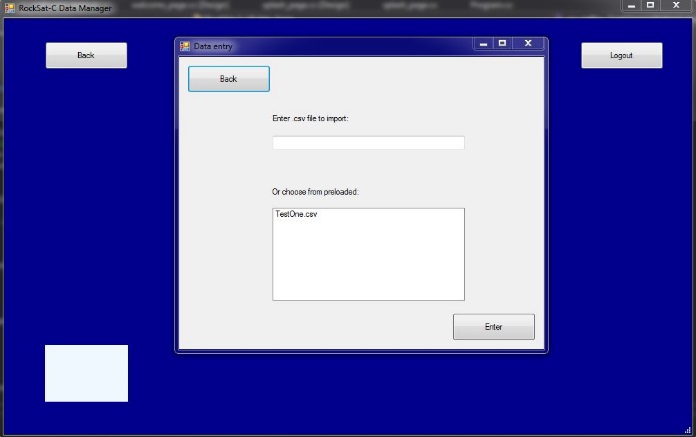


Figure X: Software flow diagram for 3 modules

**DMS UI:** The UI on this project was built with C# using a .NET framework. Windows forms are used to display choices and data as shown in the following figure. Basic user prompts and error checking are implemented at this level.



Home

* Quickstart
* Login
* Signup

Data Entry

* Back
* Logout
* Choose file
* Enter

Figure X: DMS UI basic navigation

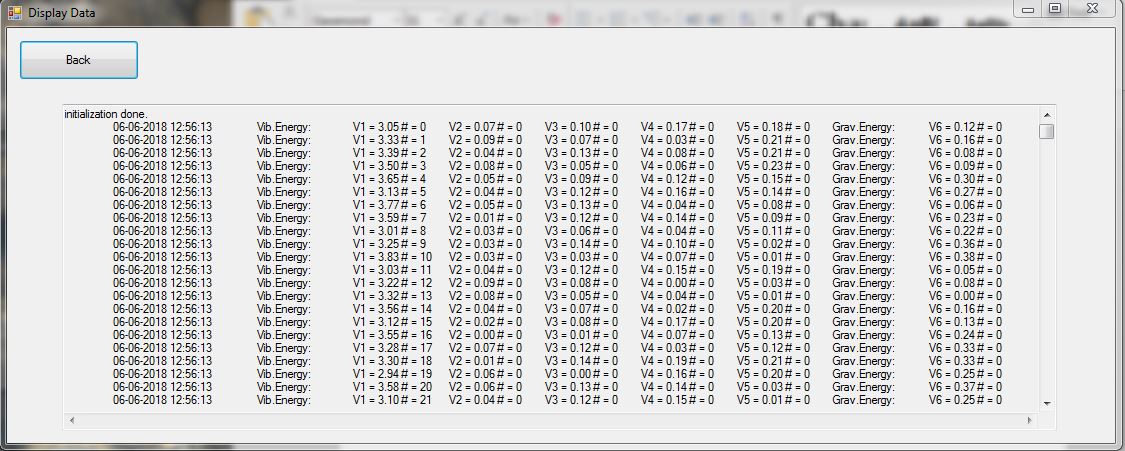


Figure X: DMS UI .txt display

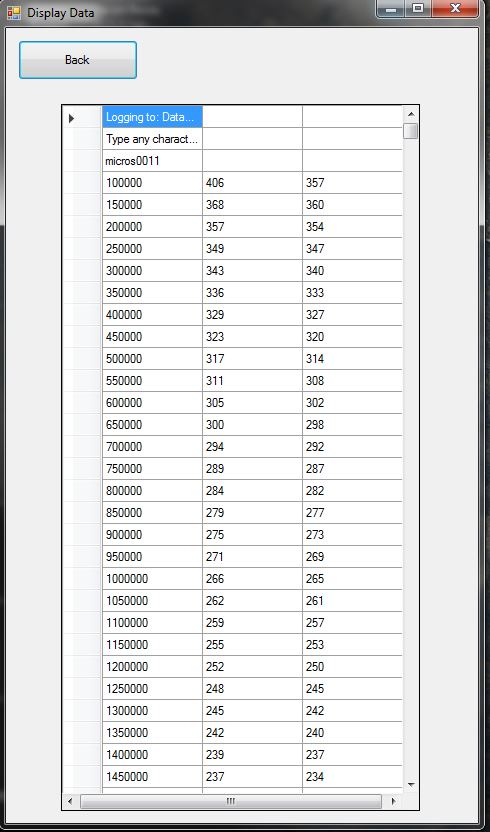


Figure X: DMS UI .csv display

Overall, the element that the two sections share is the SD card for data. Due to the nature of this project, the UI wasn’t able to be integrated directly into the main.py program. Once the canister is recovered after launch, the SD card will need to be physically removed and placed into a different computer. An overall architecture of the entire system is included below.

SD card

DMS MCU

Modules

Module data

SD card

DMS UI

Computer

Figure X: Overall Architecture of both DMS elements

## 

## Summary

This chapter provides detailed information on how each element of both DMS systems works to implement the required functionality. Major components and their relationships are shown, as well as minor components. Data flow is shown in detail and manipulation techniques are explained. Example of the UI and how a user might navigate around it is shown and technologies used are mentioned. Additional technologies and methods on closure, initiation, error handling, and communication protocols are explained as well.

# Test Results & Validation

## Overview

This chapter describes the test plan and validation techniques used in this project. Launch for the RockSat-C project is planned for June 21, 2018. All the test results in this section were created using mock code and prototype MCUs.

## Test Plan

The test plan for this project and tentative deadlines were as follows:

UI design, prototype, implementation, and testing

Sept - May

Data Management System/ Redundancy design

Nov – Dec 15

Data Management System/ Redundancy prototyping

Dec 15 – Jan 15

Data Management System integration

Jan 15 – April 1

Refinement and margin for debugging

April 1– May 1

Video conferences with WFF were frequent and subsystem testing was a point of conversation every meeting. Examples of test results are included in the Test Results section of this chapter.

Validation of functionality was tested using MCUs with actual flight code running. Flight code for each module was edited to produce mock output, as actual sensors weren’t attached. Test driven development was used throughout development, to make sure each block of code was functioning as needed. The DMS UI was user tested by various people for basic navigation and functionality.

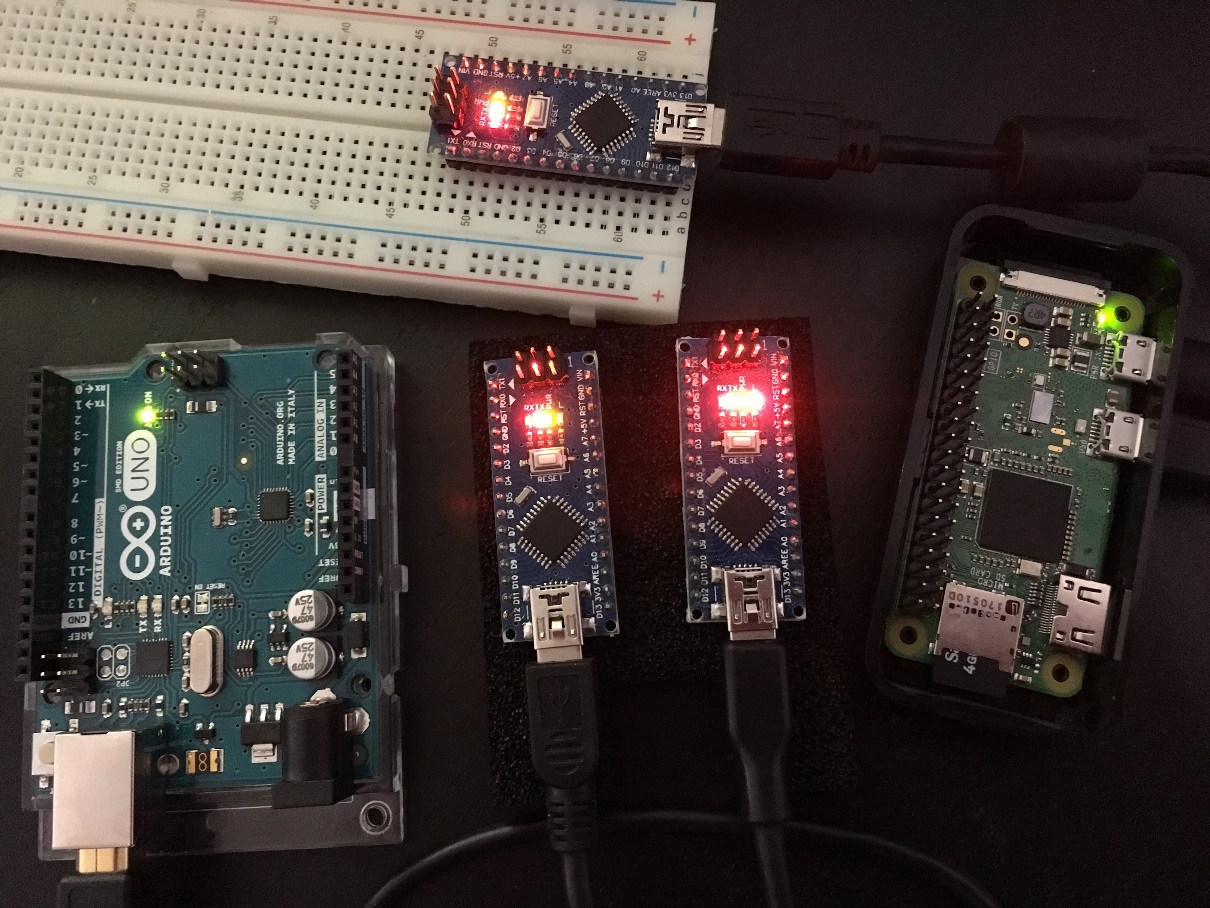


Figure X: MCUs used for testing



Figure X: Mock code set up for MEH module

## Test Results

Test results that were presented to WFF during teleconferences are provided below.

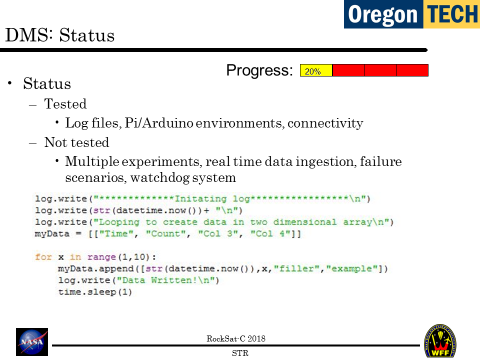


Figure X: Subsystem Testing Review, presented 2/14/2018

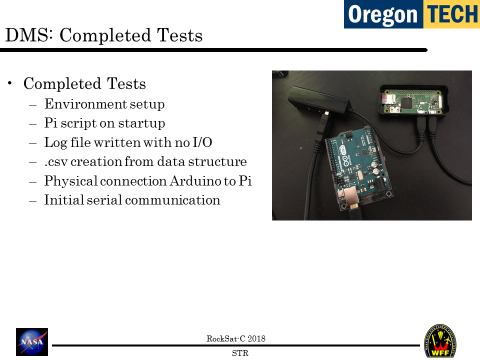


Figure X: Subsystem Testing Review, presented 2/14/2018

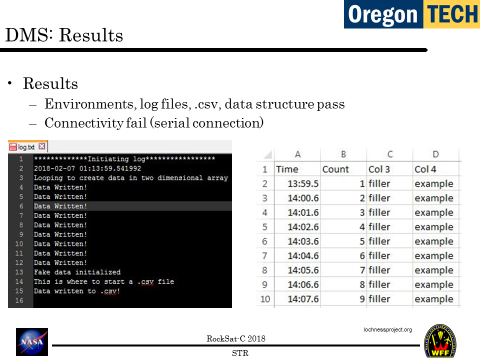


Figure X: Subsystem Testing Review, presented 2/14/2018

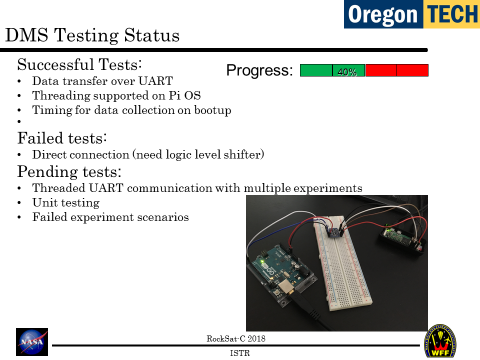


Figure X: Integrated Subsystem Testing Review, presented 3/27/2018

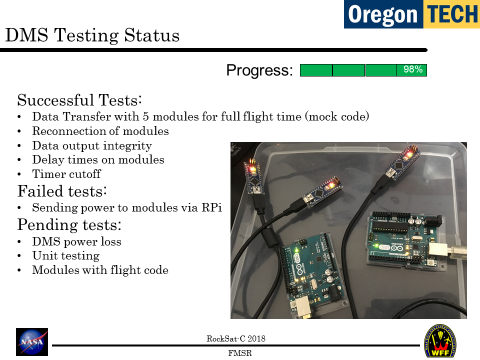


Figure X: Full Mission Simulation Review, presented 5/4/2018

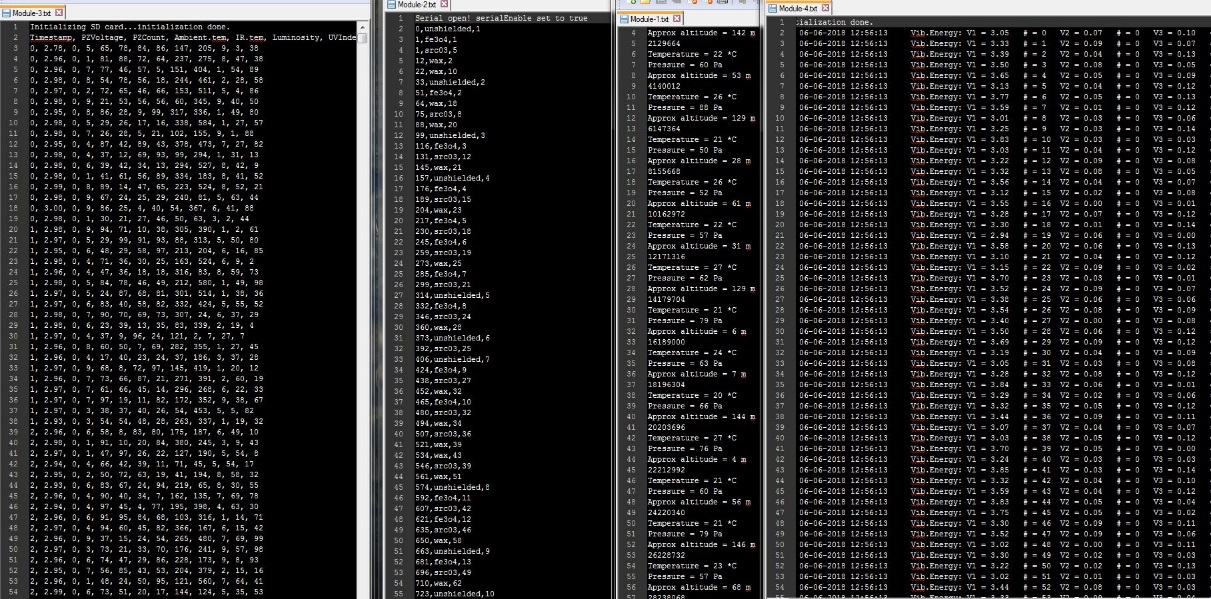


Figure X: Final output using mock flight code

Usability of the UI was also tested during the Oregon Tech Project Symposium 6/6/2018. Multiple full mission simulations with 4 mock modules running were also run at this time.

## Discussion

Test results are described below per each function requirement.

1. Module connections - Passed
   1. Recognize and connect to modules via USB
   2. Recognize and reconnect to disconnected modules

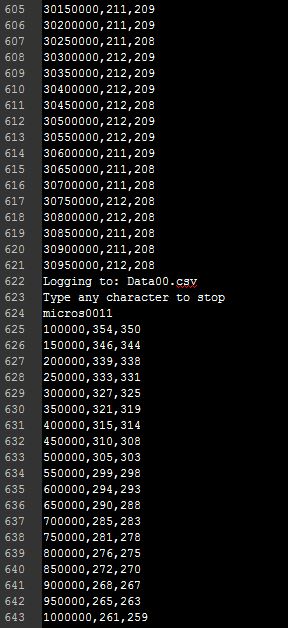


Figure X: .csv file showing reconnection of module at line 622

1. Data input streams – Passed, shown in figure X (all files)
   1. Monitor data from different modules
   2. Error/bound check all data
   3. Organize and record data to SD card
   4. Shut off collection if one should fail
2. UI – Passed, shown in figure X (UI displaying data)
   1. Display data for each experiment
   2. Display .csv and .txt files
3. Initiation – Passed, shown in successful full mission simulations
   1. Initiate all data collection on startup
   2. Log errors
   3. Timestamp log messages
   4. End data collection after allotted flight time
4. Standby mode – Passed, shown in full mission simulations
   1. Wait for module connections

Overall, all the specifications for the experiment were met. A few off-ramps had to be made, but the main functionality of the project is intact and working correctly.

## Summary

This chapter explains the test plan used for this project and the results found. Examples of testing reports delivered to WFF are included as well as an overall discussion on the test results versus the specifications of the project. Being that launch is scheduled after this paper is finished, no actual launch data is included.

# Economic and IP Analysis

Address any issues related to intellectual property here. These may include Copyright, Trademark, Patent, or Trade Secret considerations that have affected your project or should be taken into consideration.. The first item in the intellectual property statement is to designate who owns the intellectual property. Next, designate which sorts of intellectual property protections are in place - if there is a multitude, you may wish to simply say "All intellectual property rights of company x are owned by j."

Once you have done this, then you will want to state what the rights to the work by outside agents are. For example, if you run a website, you may state, "Items provided on this website may be downloaded for personal use only."

Fourth, you will want to explain any licensing policies you will have. For example, if you have invented a process, and it is patented, you may license the process for a fee. Be explicit in this.

Finally, state what the legal consequences are for violating intellectual property law.

## Overview

Provide an overview of the chapter here. For some projects, this chapter may not be relevant.

[This section should be written during the second or third term of senior project sequence.]

## Engineering Economic Analysis

Include the details of a basic engineering economic analysis. Make sure to include the cost of parts, R&D, engineering time, etc. (1-2 pages)

[This section should be written during the second or third term of senior project sequence.]

## Intellectual Property Analysis

Address any issues related to intellectual property here. These may include Copyright, Trademark, Patent, or Trade Secret considerations that have affected your project or should be taken into consideration. (1-2 pages)

[This section should be written during the second or third term of senior project sequence, but issues should be checked during the design process.]

## Summary

Summarize the chapter

[This section should be written during the second or third term of senior project sequence.]

# Summary

Provide a summary of the project. Discuss potential future directions or possibilities for improvement. Include some concluding remarks.

## Overview

Provide an overview of the chapter here.

[This section should be written during the third term of senior project sequence.]

## Project Summary

Provide a summary of the project. (1 page)

[This section should be written during the third term of senior project sequence.]

## Future Direction

(1 page)

[This section should be written during the third term of senior project sequence.]

## Concluding Remarks

Include some concluding remarks.

[This section should be written during the third term of senior project sequence.]

References

Include references using IEEE transactions citation format here.

NASA

User Guide

2017 RockSat-C

Appendix

Use the appendix to attach additional schematics, code, and other relevant reference material that has not been included as part of the report. If these documents are to be printed separately and attached with the rest of the report *a posteriori,* please use this section to create a table of contents and provide a short explanation/overview of each of the documents attached.