CubeSat Communications System

Project Members

Zee Fisher - lfisher2020@fau.edu

David Miloseski - dmiloseski2020@fau.edu

Morgan Benavidez - mbenavidez2020@fau.edu

Jonathan Mazurkiewicz - jmazurkiewic2022@fau.edu

Carson Van Buren - cvanburen2019@fau.edu

Advisor

Dr. George Sklivanitis - gsklivanitis@fau.edu

Florida Atlantic University

Department of Electrical Engineering and Computer Science

Fall 2023

Project Summary:

The project aims to develop a radio communication system tailored for a CubeSat, a small satellite platform. Leveraging lightweight and power-efficient components, the system prioritizes efficient data transmission, error correction, and compatibility with the CubeSat's size constraints. This innovative radio system incorporates Software-Defined Radio (SDR), providing additional versatility, efficiency, and effectiveness.

Table of Contents

1 Introduction

1.1 Problem Description

Traditional satellite communication systems are vastly immense projects and require a large amount of planning, coordination, resources, and time to bring to completion. They involve many hardware components and once they are launched, cannot be easily fixed, or altered. In addition, communication is becoming flooded at the UHF frequency, resulting in a lack of efficacy in communicating data.

In recent years, there has been a shift in the production of CubeSats. CubeSats are exceedingly small in nature (10x10x10 or 1U – up to 6U) and are launched alongside existing launches and are deployed in space, where they can begin to orbit. This provides the ability for many additional applications for satellites in space with the additional constraints of size and power.

Especially in radio communications, the available size and power greatly increase the difficulty of establishing a communication link. There exists a technology, namely, software-defined radio (SDR) that can take advantage of the reconfigurable nature of Field Programmable Gate Arrays (FPGAs) for the digital processing of radio frequencies. This permits the user to operate within the specified design constraints, as SDR's are low power and small enough to fit on a CubeSat.

Still, achieving a reliable communication link from CubeSat to ground station at the desired frequency is a non-trivial task. Noise, free-space loss, and many other factors disrupt radio transmissions across large distances. Finding the correct software, setting up a working ground station capable of decoding signals, encoding signals from the SDR in a way that mitigates error, and creating a reliable communication link is the chief problem our project will aim to solve.

1.2 Significance

The ability to successfully send data from space to Earth in a system that is relatively low-cost creates an ability for many applications. The transmission of data is an integral piece that could pave the way for advancement of space research. While CubeSats already exist, FAU does not currently have a CubeSat program. This will permit research to be done at FAU that was not able to be done previously. We will be collaborating with the Biology and Mechanical Engineering Departments to launch the CubeSat in the near future for the purpose of testing the viability of genetically modified plants in the challenging environments of space. The communication system will be responsible for transmitting sensor data. If this data cannot be decoded, the experiment will need to be redone.

1.3 Goals and Objectives

The purpose of our experiment is to utilize SDR to achieve a successful communication link between the CubeSat in development and the ground station. This includes proper hardware selection for transmission, encoding bits of data at the specified frequency band, and decoding the data at the ground station. The main objective will be to establish this communication link to transmit any type of data necessitated by the experiment.

In order to achieve these goals, there are several milestones that must be met. Once one objective is achieved, which is comprised of many subtasks, we may proceed to the next objective. Some objectives will be developed in parallel, such as simulation.

1. Select components for the satellite and ground station that can communicate at a specific frequency and protocol.
2. Perform simulation and analysis of modelling RF transmission via Gnu Radio software.
3. Assemble ground station and test actual signal transmission.
4. Perform link budget analysis to ensure sufficient antenna gain.
5. Encode and decode bits between satellite SDR and ground SDR.
6. Increase difficulty of testing parameters by introducing noise and reducing power.
7. Live test by decoding information from CubeSat during balloon flight.

These objectives are general milestones. The specific communication protocol will influence many factors like bits per second and signal to noise ratio, as well as component hardware selection. The final design must fit within 1U of space (10x10x10) and use minimal power.

1.4 Literature Survey

The design of our satellite to ground station communication system requires the study of radio communication and its established design principles. To understand what goes into making a satellite-ground radio system we need to review: how software-defined radios are designed, how they are designed for satellite to ground communication, what are the standards for communication, and example satellite designs. As part of our research, we will review current documentation on existing design platforms, design standardizations, and satellite specifications.

To design a radio system, we will use GNU Radio [1] to design software that will operate onboard a software defined radio (SDR). GNU Radio's own documentation provides enough information to understand how to use the software to design a radio. It also provides a set of tutorials for specific radio designs. These include radio designs for Narrow Band FM, QPSK, BPSK, and FSK. [2] Our project design will include the use of these technologies, so knowing how to build them in the GNU radio software will save time in our design and implementation. In particular, we will see that satellites often use FSK. GNU Radio's tutorial provides an entire flow graph for the transmitter and receiver ends of this radio. In fact, it makes the design of such a system relatively simple for our project, providing a module for the system itself. [3]

With GNU Radio alone, it is possible for us to design a software radio for the satellite and the ground station. However, most satellites work with the SatNOGS ground stations and software. [4] For design our ground station and satellite, we will reference the SatNOGS documentation for current communication stands, and ground station design specifications. [5] The wiki dives into the specifics of building a ground station. This includes the actual physical design of the ground station. It explains the process of installing the software on a Raspberry Pi, which will help us avoid designing extra software for the ground station we didn't need to. It also provides information on how to add your station and satellite to their network. [5] The SatNOGS documentation provides instructions for setting up SatNOGS as part of a GNU Radio plus SDR configuration. [6]. We will be referencing this as part of our ground station design.

For the specifics of communication, we look to the standard design protocols currently used by other satellites and radios. AX.25 is an older protocol that is very common among amateur radio operations. Version 2.2, released in 1998, aims to provide a simple protocol for amateur radio communication between two parties. The data in the AX.25 format is processed into frames, which transport meta-data about the packet and the "information" that a user wants to convey. [7]. It also includes a series of state-machines that serve as logical examples when implementing the standard. In developing our communication standard, even if we choose not to use AX.25, it will serve as a good example for radio design.

Our ground station and satellite radio system will not be entering space for this project. However, it is the intent of the later completed system to enter space. Keeping this in mind we will design our system around space fairing standards. This includes considering the CCSDS standards: TC Space Data Link Protocol, Space Packet Protocol, and the Recommendations for Space Data Systems Standards. These protocols and standards are included in the CCSDS Blue Books, and we will likely explore further if needed. [8] The Telecommand TC Space Data Link Protocol is for ground-to-space or a space-to-space missions. The protocol promises efficient data transfer with different service types, allowing repeated and asynchronous communication. [9] We will rely on this protocol for communication in our own design. Even if we do not, or choose to only implement it minimally, we will set a foundation for further communication across this protocol; that is in regard to the projects directly following this one. The Space Packet Protocol will provide us with the actual structure of our packets. The document specifies the exact structure of the primitives, how they work together, and what responses each component should expect during communication. [10] When creating the signal inside the satellite the frequency and modulation we use should follow a standard as well. This will help future projects built on this one with getting approval for communication, and also with this project's own deliverables. Recommendations for Space Data Systems Standards is a standard for the actual modulation and frequency of space communication. Part 1 covers everything from the symbols, the operation ranges, the functions for filtering, and more. It also includes specific instructions for building the signal with different parameters. [11]

With this project, it may be pertinent to study successful satellite radios and ground station systems. The FUNcube-1 is a satellite with a hardware radio in space and a ground station SDR. It has a detailed manual covering the specifics of the satellite's design, including the radio and ground station dongle. It covers the specific standard they use and what frequency they broadcast at. The FUNcube-1 both sends and receives data, but our satellite will only send data. [12] As an example, even if not completely similar, it provides enough information to serve as a guide for our design.

2 Proposed Design

2.1 Project Requirements

Functionality Requirements:

F1: The system shall determine the frequency band.
F2: The system shall, as of now, use the UHF band and explore different bands when the project is ready for space deployment.

F3: The system shall encode data at the proper frequency by encoding bits onto pulses to travel over a long distance.

F4: The system shall transmit data from the satellite to the ground station.

F5: The system shall decode on the ground station to convert data to readable information based upon the received signal, decoding pulses into usable binary bits representing information.

F6: The system shall adequately respond to noise/disruptions in the signal by developing techniques for handling noisy signals, radiation, obscurity, or unknown factors.

F7: The system shall clean the data to turn the analog signal into digital correctly using error correction and digital signal processing.

F8: The system shall store information in the database, which will store information after processing/cleaning.

F9: The system shall ensure a sufficient data rate, determining the speed of transmission (bits/second), affected by frequency selection.

F10: The system shall determine the antenna gain based upon the size of the antenna and the available power mathematically. It needs to provide the minimum amount of gain for the antenna to be able to receive a clear signal, considering factors such as noise.

F11: The system shall select a modulation scheme suited for the task (BPSK, GMSK, QPSK) based on factors including data rate, bandwidth, signal fidelity, interference, and noise.

Usability Requirements:

U1: The system shall preserve data integrity to ensure that the user has clean data and can make informed decisions based upon that data.
U2: The system shall consider power requirements/constraints with a 12V battery pack initially available, and it will change to solar panels when ready for launch.

U3: The system shall adhere to size constraints with 1U available, roughly estimated at 10x10x10cm

U4: The system shall maintain consistent frequency to ensure transmissions remain in the specified band for the user to interpret data successfully from the ground station.

U5: The system shall meet compatibility requirements by ensuring that communication protocols are in line with the receiving ground station for decoding.

U6: The system shall operate in various modes according to a state diagram, including emergency, deploy antenna, low power, low battery, and data transmission (low, high, none).

U7: The system shall correct deployment by verifying that the satellite was deployed correctly and sending a 'pass' signal via Morse Code.

Safety Requirements:

S1: The system shall perform an emergency shutdown if there is a power error or fault to ensure safety.
S2: The system shall develop backup systems, including a low-power alternative to only send position/coordinates and limited data if power reserves are low.

S3: The system shall ensure the antenna is deployed safely by making sure that a proper amount of time has passed before deploying the antenna for the safety of the spacecraft/orbit.

S4: The system shall ensure that the ground station and satellite are operating on the same frequency, as transmissions not on the correct frequency cannot be received by the ground station.

S5: The system shall follow all FCC guidelines and regulations for radio transmissions, as transmissions are regulated and must adhere to FCC guidelines for transmission.

S6: The system shall meet licensing requirements by obtaining an amateur HAM radio license to transmit in the UHF band.

2.2 Project Design

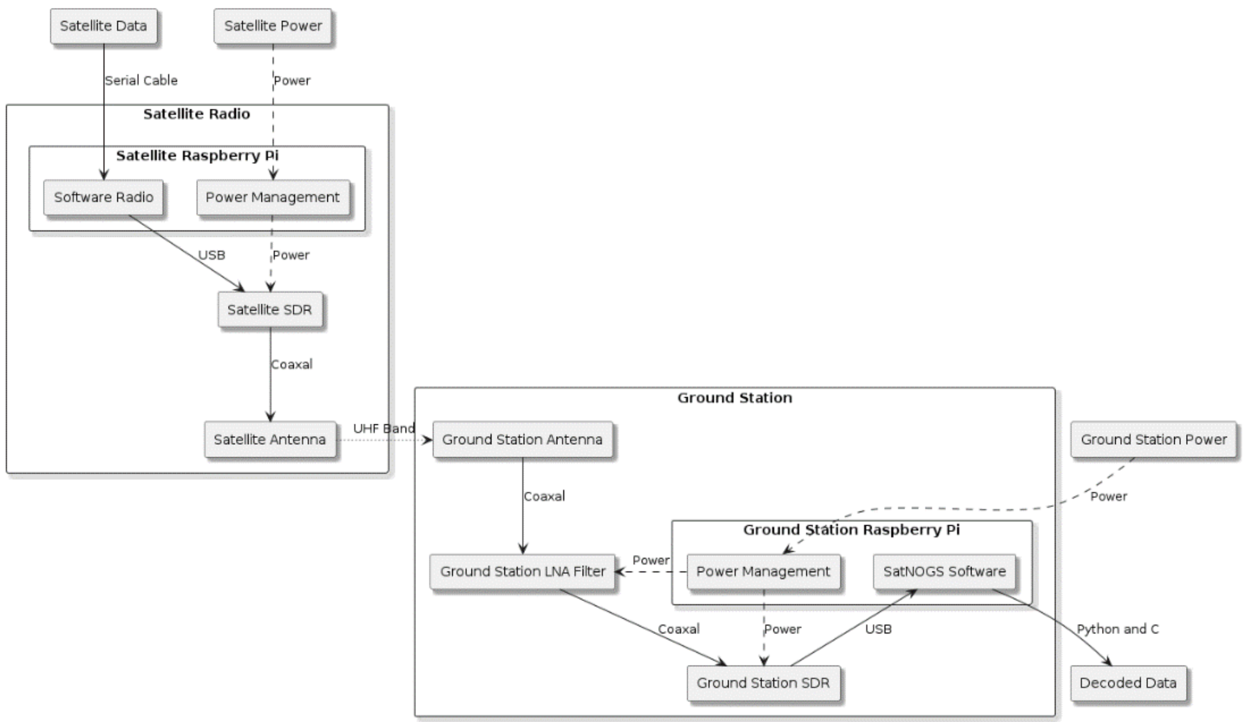
Our proposed design has two main parts: the radio system inside the satellite and our ground station. The satellite's radio system is designed to be lightweight and efficient, allowing it to send data effectively. On the ground, our station works to communicate with the satellite during its orbit, keeping track of where it is, and making sure data exchanges go smoothly. These parts work together to create a reliable communication system, crucial for controlling the mission and getting data back from the satellite.

The CubeSat Radio Communication System initiates its workflow by collecting raw data - we have not determined if this data will be in digital or analog form yet, as it will be handled by another team. If it is analog, the data will need to be converted to digital format. There will be a Raspberry Pi on board the satellite that will handle this part as well as formatting the digital data according to the requirements of the communication protocol and modulation scheme. This may involve segmentation, packetization, and error-checking. We may also introduce compression, filtering, and basic analysis to reduce the amount of data that needs to be transmitted to Earth.

The Raspberry Pi will communicate with the SDR to modulate the processed data into a suitable radio-frequency signal. It will configure the SDR parameters based on the communication requirements and mission objectives. The modulated signal is then sent to the antenna for transmission to the ground station. We are currently considering the ISIS GNSS Patch Antenna, but this may change as the project unfolds. The Raspberry Pi will coordinate the timing of transmission based on the satellite's orbit and communication window with the ground station.

On the ground station side, our system integrates an antenna designed for accurate tracking of CubeSat passes. A Low-Noise Amplifier (LNA) enhances signal reception during communication windows, further amplifying the ground station's capabilities. The ground station's Raspberry Pi serves as the processing unit, interfacing with the SatNOGS software suite and the SDR. SatNOGS is an open-source platform that provides tools for satellite tracking, data recording, and analysis. The SDR will demodulate the signal and provide usable data on the ground, which accomplishes the mission of successfully acquiring usable data generated on the satellite.

2.3 Block Diagram



System Level Design:

Our CubeSat system operates as a well-coordinated information relay, starting with a microcontroller that accepts data from the satellite. This collected data is then transformed into a modulated signal through our Software-Defined Radio (SDR) before being transmitted via an antenna. At the receiving end, our ground station is equipped with a specialized antenna tuned to our satellite's frequency. To ensure the purity of the incoming signal, we employ a Low-Noise Amplifier (LNA) to filter out unwanted interference. The SDR at the ground station then demodulates the received signal, making the original data accessible. This decoded information is subsequently processed by locally hosted software, ultimately restoring it to its original format. This streamlined process ensures a reliable flow of information between the satellite and ground station.

Ground Station Subsystem:

Team Member(s) Responsible: Carson Van Buren

The ground station subsystem is responsible for precisely capturing signals from our satellite. Its antenna is carefully aimed at the satellite, enabling it to pick up radio signals accurately. To ensure data purity, the Low-Noise Amplifier (LNA) acts as a filter, eliminating unwanted signals like nearby FM radio stations and other radio traffic that is not relevant to our mission. The filtered signal is then passed to the Software-Defined Radio (SDR), which is configured to the specific frequency we used for data transmission from the CubeSat. This configuration ensures a proper match between the captured signal and our satellite's transmission. Next, the signal is transmitted to the Raspberry Pi, which hosts the SatNOGS ground station software. This software is instrumental in handling and processing the data, making it accessible and facilitating the essential communication between the ground station and the satellite.

Data Collection and Transmission Subsystem

Team Member(s) Responsible: Morgan Benavidez

We have selected the Raspberry Pi as our microcontroller onboard the satellite for its adequate processing power and compatibility with our software-defined radio (SDR). The SDR allows for more flexibility and adaptability in handling various radio communication protocols and frequencies. Data will first be collected from the satellite by a serial connection from our Raspberry Pi to hardware being prepared by another team.

To ensure the data's successful transmission, there will be a dedicated program running on the Raspberry Pi responsible for processing the incoming data. As power constraints are an obstacle, we may need to perform operations to see if the data has changed enough from the satellite's last transmission to warrant spending the necessary power needed to send information. After these operations, the data will then be encoded. Once encoded, the data will be passed by USB connection from the Raspberry Pi to the SDR to undergo modulation and transmission.

Satellite Power Management Subsystem

Team Member(s) Responsible: Zee Fisher

Power will be routed from the Satellite into the Raspberry Pi's main power. Then, the Raspberry Pi will switch power to the SDR when ready to broadcast. The Power Management Subsystem will monitor power drawn from the Raspberry Pi and the SDR. Each component will be powered will standard connection for that device. The Raspberry Pi will be connected to power via its USB-C port. The SDR will receive power from the Raspberry Pi via its own USB connection.

The Goal of this system is to ensure both the proper functionality of the radio equipment, and the safety of all satellite components. Functionality is ensured by activating power to the antenna when broadcasting is ready. Power to the Raspberry Pi must also remain consistent, and part of this system's role is to manage that. Safety is also a necessity. The design of this subsystem will ensure that during the operation of the Satellite Radio, no shorting or disconnects will occur. Components will also receive power at the rate and level that they are designed to support.

SDR (Software Defined Radio) Subsystem:

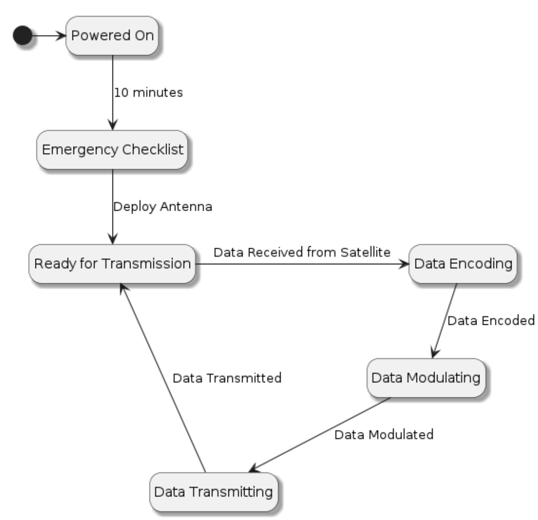
Team Member(s) Responsible: Jonathan Mazurkiewicz

Software defined radio is a physical hardware component that is capable of handling analog signals inside of software instead of requiring specific analog electrical components. Modulation, demodulation, digital signal processing, filtering, and frequency selection are all tasks that would have required specific analog components. However, with SDR, all of this can be done with software. This greatly increases the reconfigurability of radio transmissions based upon the needs of the user. Internally, the SDR contains an Analog-to-Digital Converter, a sound card, a microphone, an antenna, and a general-purpose processor. Changes to the software change how the SDR can interact with signals, what type of signals, and how digital signal processing is achieved.

We will have two SDRs: one for the satellite and one for the ground station. While we will not be transmitting data from the ground station to the satellite at this point, in the future we will be able to send control signals to the satellite, if required. The satellite SDR will be responsible for modulating a signal at a specific carrier frequency that is amplified by the antenna on the satellite to be received by the ground station.

2.4 State Diagram

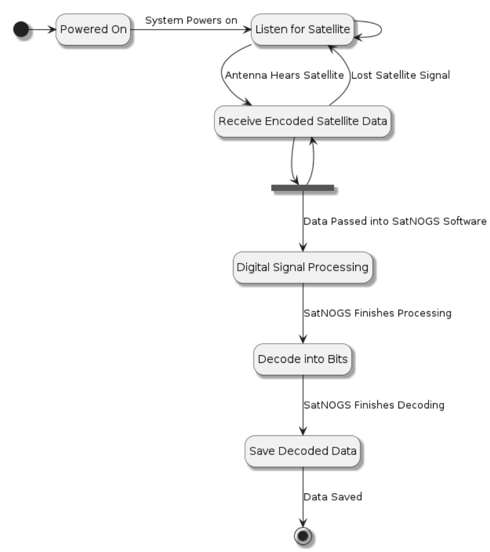
Satellite:



Satellite:

Transition	Description
Powered On to Emergency Checklist	After 10 minutes of going online, the emergency checklist is started.
Emergency Checklist to Ready for Transmission	After the emergency checklist is completed, the antenna is deployed.
Read for Transmission to Data Encoding	As data is received from the satellite, the software radio will begin encoding the data.
Data Encoding to Data Modulating	As data is encoded, the software radio will begin modulating the data.
Data Modulating to Data Transmitting.	As data is modulated, it will be transmitted via the SDR and antenna.
Data Transmitting to Read for Transmission	After data is transmitted, the Software Radio will enter its ready state.

Ground Station:



Ground Station:

Transition	Description
Powered On to Listen for Satellite	When the system is powered on, it will begin listening for the satellite's signal.
Listen for Satellite to Listen for Satellite	The system will keep checking for the satellite if it does not capture a signal.
Listen for Satellite to Receive Encoded Satellite Data	Once the system hears the satellite, it will begin retrieving encoded data.
Receive Encoded Satellite Data to Fork	Once data is being received, the process forks, as data can be recorded and decoded at the same time.
Receive Encoded Satellite Data to Listen for Satellite	When the signal is lost, the ground station will stop receiving data.

Fork to Receive Encoded Satellite Data	The system will continue to listen for data as it decodes what it has.
Fork to Digital Signal Processing	Data that needs to be decoded will be sent to the SatNOGS software for processing.
Digital Signal Processing to Decode into Bits	Once SatNOGS finishes processing data, it will begin decoding.
Decode into Bits to Save Decoded Data	Once SatNOGS finishes decoding, data is saved locally.

3 Implementation

3.1 Hardware Implementation

This section summarizes the hardware needed/acquired for our proposed systems. This table states the hardware necessary, and the system it is used in.

Chosen hardware:

Nooelec Lana LNA	Ground Station
RTL-SDR Blog SDR v3	Ground Station
Foldable 3-ELement Yagi Antenna	Ground Station
2x Raspberry Pi 3 Model B	Ground Station, CubeSat
Ettus USRP B205mini-I SDR	CubeSat
ISIS GNSS patch antenna	CubeSat
12-in COAX, SMA to SMA	CubeSat
35-Ft COAX, SMA to SMA	Ground Station
18-in COAX, SMA to UHF	Ground Station
Rotating Adjustable Tripod	Ground Station

In configuring our ground station hardware, we methodically selected components to optimize performance and functionality. Leading our setup is the RTL-SDR Blog SDR v3, a versatile software-defined radio compatible with our ground station software for efficient signal processing. Next is the Raspberry Pi 3 Model B, a compact single-board computer chosen for its high performance and compatibility with our software and overall hardware setup. Enhancing our communication capabilities, we implemented a foldable 3-Element Yagi Antenna, emphasizing its directional characteristics for precise satellite targeting. To bolster signal reception, the Nooelec Lana LNA, a low-noise amplifier, was strategically included. Recognized for its effective noise filtration, this component significantly improves signal clarity and strength, especially vital for capturing signals from our designated satellites, including the CubeSat. Our hardware selection reflects a deliberate engineering approach aimed at achieving optimal performance in ground station operations.

For the hardware implementation of our CubeSat, we have carefully chosen specific components to ensure optimal functionality. At the core of our setup is the Ettus USRP B205mini-I SDR, a software-defined radio capable of transmitting more robust signals compared to the RTL-SDR. Its compact dimensions align seamlessly with the rest of the onboard components in the CubeSat. Complementing this is the Raspberry Pi 3 Model B, selected for its compatibility with our chosen software and overall hardware configuration. To facilitate Earth-space communication, we integrated an ISIS GNSS patch antenna. This low-profile antenna is a common choice for such applications, contributing to the efficient operation of our CubeSat in the realm of space communication.

3.2 Software Implementation

For the software implementation of our ground station, we are utilizing an image of Raspberry Pi OS which has software pre-installed that is capable of capturing, decoding, and uploading signals to a dashboard on the SatNOGS website. This software image is maintained by the SatNOGS group.

For the software implementation of our CubeSat we are primarily utilizing GNURadio, a GUI software and Python library which does a lot of the heavy lifting for signal processing. The GNURadio library is preconfigured with existing modulation schemes, frequencies, etc. which allows for parameterization.

We will be primarily programming in Python for our project, as most of the SDR programming can be done in Python, and using 2 Raspberry Pis facilitates this, as Raspberry Pi OS comes preinstalled with Python.

3.3 User Interface and Data Communcations

For our user interface, we are utilizing the web dashboard provided by SatNOGS, which provides us with data waterfalls, signal audio, and decoded data. Said data will be data which is captured in real-time from our CubeSat. Our CubeSat will be sending telemetry data which is encoded with the BPSK modulation scheme, and our ground station will be collecting this modulated data, demodulating it, storing it in a local database, and displaying the data on the SatNOGS dashboard.

3.4 Testing

We will be using MATLAB as a tool for our team's simulation and link budget analysis of satellite and ground station communications. This enables our team to simulate the entire communication link, considering the satellite's position in space, antenna characteristics, and ground station parameters. MATLAB's signal processing capabilities come into play for realistic modeling of communication channels, including signal propagation, atmospheric effects, and potential interference. The link budget analysis, crucial for determining the system's overall performance, can be efficiently conducted using MATLAB's built-in functions, helping our team optimize power budgets, antenna gains, and signal-to-noise ratios. By leveraging MATLAB's simulation capabilities, we can iteratively refine and enhance CubeSat's communication system, ensuring its reliability in a variety of operational scenarios.

We will also be testing our built components before the final launch. We have already begun testing our ground station as it is set up and receiving packets from satellites already. Once we have our satellite built as well, we will test if they can communicate with each other. These tests will allow us to ensure that our final launch goes as smoothly as possible. Our goal is to minimize noise that will interfere with the packets we are sending from the satellite to the ground station through our testing efforts.

Our testing procedure for our individual parts is as follows: Use our ground station antenna and SDR with existing software to capture known good radio signals, to ensure the antenna and SDR are in working order. Then we will use the CubeSat antenna and SDR to send radio signals to see if they are in working order. Then we will incorporate the LNA to test to see if it is effective at filtering out unwanted radio noise. After testing our hardware with known working software, we can progress to testing with our custom programs so we can debug any issues present in our code.

4 Development Plan and Schedule

4.1 Outline of The Plan

Up to this point in the development of our CubeSat project, we have done the following. Through analyzing previous research papers and patents in the field of CubeSat, we have been able to learn from past CubeSat projects to better plan the development of our own. We have a much more expanded knowledge base on many key topics that will help inform us in the major decisions we have to make. There are a lot of restrictions on which radio frequencies unlicensed amateurs can use. As of now, we plan to use a UHF frequency as it requires no license. However, we are looking into applying for a license for other frequencies like C Band. We have also done extensive research on which parts we will be using for the ground station. In fact, we have already built a working ground station using (Nooelec Lana LNA, 3-Element Yagi Antenna, RTL-SDR v3, Raspberry Pi 3 model B, SatNOGS software image). In addition, we have set up the software we will be using to collect packets from the satellite called SatNOGS.

The next steps we are working on are creating a simulation in MATLAB for our budget link analysis and research as well as working on our encoding scheme. To finish our ground station, we will be searching for a good place to mount it as well as test to see how far its reach is in terms of grabbing packets from satellites. Then we must research satellite parts, build the satellite, and test to see if it can send data to our ground station. We will test with difficulty and noise parameters in hopes to eliminate those factors as much as possible during our live test. Finally, after two live tests we will have our final launch where we plan to send our satellite one hundred thousand feet into the air where it will collect data and send it down to our ground station.

4.2 Work Breakdown and Milestones

Major Milestones

System	Task	Estimated Completion Date
Ground Station	Build Ground Station	11/30/2023
Satellite	Research and Acquire Parts	11/30/2023
Simulation	MATLAB Simulation Ready	11/30/2023
Satellite	Assemble Satellite Parts	01/15/2024
Ground Station	Decoding Algorithm Complete	02/28/20224
Satellite	Encoding Algorithm Complete	02/28/2024
Simulation	Test Ground and Satellite Transmission	02/28/2024
Test	Final Launch	04/01/2024

Work Breakdown

Carson Van Buren

Build ground station	CB Carson Van Buren	11/30	Ground Station
Explore mounting location	CB Carson Van Buren	11/30	Ground Station
Install necessary drivers and equipment - Ground Station	CB Carson Van Buren	11/30	Ground Station
Research how SatNOGS software works to decode data	CB Carson Van Buren	11/30	Ground Station
Assemble equipment	CB Carson Van Buren	11/30	General Tasks / R...
Communicate with current ground station for advice	CB Carson Van Buren	11/30	General Tasks / R...
Capture existing signals from satellites 0/1	CB Carson Van Buren	1/2024	Ground Station
Simple Test of Transmission Between Satellite and Ground	ZF JM MB CB DM	2/2024	Testing / Milesto...
FINAL LAUNCH	ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 1	ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 2	ZF JM MB CB DM	4/2024	Testing / Milesto...

David Miloseski

Simulation in Matlab Presentation		DM David Miloseski	11/15	Tasks for 11/16/2...
Use Matlab To Test Output Of Parts (Ongoing)		DM David Miloseski	11/30	Simulation
CALCULATIONS	0/3	DM David Miloseski	12/15	General Tasks / R...
CHECK ALL SAFETY STANDARDS		DM David Miloseski	12/15	General Tasks / R...
Link Budget Analysis	0/3	DM David Miloseski	1/2024	Simulation
Assemble Satellite		ZF DM	1/2024	Satellite
Simple Test of Transmission Between Satellite and Ground		ZF JM MB CB DM	2/2024	Testing / Milesto...
FINAL LAUNCH		ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 1		ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 2		ZF JM MB CB DM	4/2024	Testing / Milesto...

Jonathan Mazurkiewicz

Explain Encoding Scheme (BPSK)		JM MB	11/15	Tasks for 11/16/2...
Give Mech Engineering Team Parts List	0/1	JM Jonathan Mazurkiewicz	12/1	Satellite
Install necessary drivers for equipment - satellite station		JM Jonathan Mazurkiewicz	12/1	Satellite
GNU RADIO MODEL PAIRED WITH SDR - TEST		JM MB	12/31	Testing / Milesto...
Begin development of encoding algorithms		JM MB	1/2024	Satellite
Complete Encoding Algorithm	0/1	JM MB	2/2024	Satellite
SUCCESSFULLY ENCODE DATA (100%)		JM MB	3/2024	Satellite
Gnu Radio Model Transmission		ZF JM	11/30	Satellite
Research and Acquire Satellite Parts	0/3	ZF JM	11/30	Satellite
Interconnection between satellite components		ZF JM	12/1	Satellite
Simple Test of Transmission Between Satellite and Ground		ZF JM MB CB DM	2/2024	Testing / Milesto...
INCREASED DIFFICULTY TESTING PARAMETERS	0/2	ZF JM MB	3/2024	Testing / Milesto...
FINAL LAUNCH		ZF JM MB CB DM	4/2024	Testing / Milesto...

Morgan Benavidez

Explain Encoding Scheme (BPSK)		JM MB	11/15	Tasks for 11/16/2...
GNU RADIO MODEL PAIRED WITH SDR - TEST		JM MB	12/31	Testing / Milesto...
Begin development of encoding algorithms		JM MB	1/2024	Satellite
Complete Encoding Algorithm	0/1	JM MB	2/2024	Satellite
SUCCESSFULLY ENCODE DATA (100%)		JM MB	3/2024	Satellite
Begin development of decoding algorithms		MB Morgan Benavidez	1/2024	Ground Station
Complete Decoding Algorithms		MB Morgan Benavidez	2/2024	Ground Station
COMPLETE GROUND STATION - DECODE(90%)		MB Morgan Benavidez	3/2024	Ground Station
Simple Test of Transmission Between Satellite and Ground		ZF JM MB CB DM	2/2024	Testing / Milesto...
INCREASED DIFFICULTY TESTING PARAMETERS	0/2	ZF JM MB	3/2024	Testing / Milesto...
FINAL LAUNCH		ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 1		ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 2		ZF JM MB CB DM	4/2024	Testing / Milesto...

Zee Fisher

Explain Communication Standards	0/3	Zee Fisher	11/15	Tasks for 11/16/2...
Gnu Radio Model Transmission		ZF JM	11/30	Satellite
Research and Acquire Satellite Parts	0/3	ZF JM	11/30	Satellite
Research Parity for Data		Zee Fisher	11/30	General Tasks / R...
Select communications protocol		Zee Fisher	11/30	Testing / Milesto...
Interconnection between satellite components		ZF JM	12/1	Satellite
Assemble Satellite		ZF DM	1/2024	Satellite
Simple Test of Transmission Between Satellite and Ground		ZF JM MB CB DM	2/2024	Testing / Milesto...
INCREASED DIFFICULTY TESTING PARAMETERS	0/2	ZF JM MB	3/2024	Testing / Milesto...
FINAL LAUNCH		ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 1		ZF JM MB CB DM	4/2024	Testing / Milesto...
Live Test 2		ZF JM MB CB DM	4/2024	Testing / Milesto...

4.2 Budget Analysis of Parts

Part	Location	Price
Nooelec Lana LNA	Ground Station	TBD
RTL-SDR Blog SDR v3	Ground Station	TBD
Foldable 3-ELement Yagi Antenna	Ground Station	TBD
2x Raspberry Pi 3 Model B	Ground Station, CubeSat	TBD
Ettus USRP B205mini-I SDR	CubeSat	TBD
ISIS GNSS patch antenna	CubeSat	TBD
12-in COAX, SMA to SMA	CubeSat	TBD
35-Ft COAX, SMA to SMA	Ground Station	TBD
18-in COAX, SMA to UHF	Ground Station	TBD
Rotating Adjustable Tripod	Ground Station	TBD

5 References

[1] "GNU Radio - The Free & Open Source Radio Ecosystem," GNU Radio. https://www.gnuradio.org/

[2] "Tutorials - GNU Radio." https://wiki.gnuradio.org/index.php/Tutorials

[3] "Simulation example: FSK - GNU Radio." https://wiki.gnuradio.org/index.php?title=Simulation_example:_FSK

[4] "SatNOGS," SatNOGS. https://satnogs.org/

[5] “SatNOGS Wiki.” https://wiki.satnogs.org/Main_Page

[6] “SDR setup - SatNOGS Wiki.” https://wiki.satnogs.org/SDR_Setup

[7] “AX.25 Link Access Protocol for Amateur Packet Radio,” Tucson Amateur Packet Radio Corporation, Jul. 1998. http://www.tapr.org/pdf/AX25.2.2.pdf

[8] Arsc, “CCSDS.org - Blue Books: Recommended standards.” https://public.ccsds.org/Publications/BlueBooks.aspx

[9] “TC SPACE DATA LINK PROTOCOL,” CCSDS, Oct. 2021. https://public.ccsds.org/Pubs/232x0b4c1.pdf

[10] “SPACE PACKET PROTOCOL,” CCSDS, Jun. 2020. https://public.ccsds.org/Pubs/133x0b2e1.pdf

[11] “RADIO FREQUENCY AND MODULATION SYSTEMS— PART 1 EARTH STATIONS AND SPACECRAFT,” *CCSDS*, Oct. 2021. https://public.ccsds.org/Pubs/401x0b32.pdf

[12] “The FUNcube Handbook,” AMSAT-UK, Nov. 2013. https://funcubetest2.files.wordpress.com/2010/11/funcube-handbook-en_v13.pdf