CySat: Satellite Mission Design

DESIGN DOCUMENT

Team Number: sdmay21-25
Client: M2I
Adviser: Dr. Phillip Jones
Team Members/Roles:
Alexis Aurandt - OBC Lead, Payload Sub-Lead, and Voltage Boost Board Lead
Alex Constant - Ground Station Front-End Lead
Chandler Jurenic - Payload Lead and OBC Sub-Lead
Jeffrey Richardson - ADCS Lead
John Lenz - Radio Lead
Scott Dressler - EPS Lead and Voltage Boost Board Sub-Lead
Team Email: cysat-f2020@iastate.edu
Team Website: http://sdmay21-25.sd.ece.iastate.edu

Revised: November 15th/3.0

Executive Summary

Development Standards & Practices Used

- NASA and CubeSat hardware standards
- Consistent code commenting and documentation for software
- Agile software development
- Code reviews by team before merges
- UART
- I2C
- Python 3+ for Ground Station
- PC-104

Summary of Requirements

- Needs to power up after been deployed from the International Space Station
- Needs to stabilize and point itself towards earth
- Needs to take soil moisture readings from Earth via a microwave radiometer
- Needs to transmit data back to the ground station in Ames, IA
- Needs to be able to collect data for its orbit life (6 months)
- Must deorbit successfully at the end of lifetime
- Needs to meet NASA's CubeSat requirements

Applicable Courses from Iowa State University Curriculum

CPR E 288: Embedded Systems

CPR E 488: Embedded Systems Design

CPR E 489: Data Communications

COMS 309: Software Development Practices

New Skills/Knowledge acquired that was not taught in courses

- Python UI Development
- Python
- PC-104 Standard
- I2C Communications
- Raspberry Pi Environment
- PCB Soldering

Table of Contents

List of Figures/Tables/Symbols/Definitions

ADCS – Attitude Determination and Control System

EPS – Electrical Power System

M:2:I – Make to Innovate

OBC – On-board Computer

SDR – Software Defined Radio

UHF Radio- Ultra High Frequency Radio

CubeSat – A standard, miniaturized scientific satellite comprising connected 10 x 10 x 10 cm cubes

SCRUM – A project management workflow type in which short bursts of work are produced and provided to the client for feedback on a regular basis

1 Introduction

1.1 Acknowledgement

We would like to thank last year's senior design team for providing handoff documentation, providing guidance, and for being available for questions. Additionally, we would like to thank Dae-Young Lee for his expertise and his guidance with the ADCS sub-system. Finally, we want to thank Dr. Jones for meeting with us weekly and for giving us guidance.

Additionally, we would like to thank M:2:I for providing documentation and information about the current state of the CySat, as well as for providing remote access to necessary equipment.

1.2 Problem and Project Statement

The CySat is a cube satellite, which is a standardized form of miniaturized scientific satellite. CySat will be deployed from the International Space Station, after which it will orbit the earth for approximately 6 months with the goal of collecting and relaying soil moisture data back to our ground station in Howe Hall on the Iowa State University campus.

The CySat is a student project started and operated by M:2:I. The sole purpose of this project is to get students engaged in a hands-on project, and the driving force behind the project is students wanting to develop a satellite that will be launched into space. Originally, this project was only for Aerospace engineering students, but M:2:I soon realized they needed Computer, Electrical, and Software engineers to take care of the onboard electronics. This is where our senior design team comes in. There are many different subsystems that need our expertise.

The CySat comprises seven subsystems. These subsystems control the satellite's orientation with respect to the Earth, collect and process data, stream the data back to Earth during communications windows, and provide ground control for the satellite. The subsystems are as follows:

- The OBC communicates with all the other subsystems and ensures that the satellite is operating according to specifications.
- The ADCS stabilizes and points the satellite toward Earth.
- The EPS regulates power from the solar cells.
- The radio relays data and commands between the CySat and Earth.
- The ground station receives data, sends commands, and acts as the interface between the M2I satellite team and the satellite.
- The payload uses an SDR to gather soil moisture readings from earth.
- The voltage boost board converts voltage from the EPS to 7.4V required for the ADCS

1.3 Operational Environment

The CySat will be launched in space from the International Space Station, and it will orbit the Earth for six months. This necessitates that the internal hardware of the CySat, as well as software running on that hardware, are robust and capable of failure recovery with minimal loss, as well as the stress from the initial launch.

The Ground Station subsystem will be installed on a desktop machine with a Windows 10 install and Python 3+, which will be connected over UART to a Kenwood radio. We will be using 436.375 MHz to communicate between the ground station and the radio.

1.4 REQUIREMENTS

1.4.1 Functional Requirements

- Must power up no earlier than 30 minutes after deployment from the International Space Station
- Must stabilize and point itself towards earth for data collection
- Must take soil moisture readings from Earth via a microwave radiometer
- Must be capable of transmitting SDR data back to the ground station in Ames, IA at a rate of 400 kb per week while within 500km of the Ground Station
- Must operate battery heaters based on the current operating temperature so as to prevent battery charging at temperatures below 0° C
- Must disable 3.3 V and 5 V outputs if the operating temperature is greater than 55° C or if battery voltage falls below 3.5 V
- Must collect data for its orbit life, a minimum of 2 months and a maximum of 6 months
- Must meet NASA's CubeSat standards and regulations
- Must receive and execute commands issued by the Ground Station while within beacon range (500 km)
- Ground Station must keep logs of sent/received commands and data, separated daily or weekly
- Must successfully deorbit at the end of its lifespan, estimated to be 221 days
- Must begin detumbling when total orbital spin exceeds 40 rads/second

1.4.2 Nonfunctional Requirements

- Ground Station UI is performant and fault tolerant (no downtime)

1.5 INTENDED USERS AND USES

Our senior design team and the M:2:I team are the end users for the CySat. Users on the ground must be able to use the Ground Station to communicate with and control the CySat.

1.6 ASSUMPTIONS AND LIMITATIONS

1.6.1 Assumptions

- We assume correct installation of each sub-system by M:2:I.
- Software on the CySat is of the same version as what we used to implement the subsystem functionality

1.6.2 Limitations

- The hardware and software of the CubeSat must comply with NASA regulations as well as CubeSat standards, and the hardware must fit within a 10 x 10 x 10 cm cube in the satellite housing.
- Access to testing on CySat components is limited
- Few operation times in lab due to COVID regulations
- Mock launches are the only feasible way of testing the subsystems

1.7 Expected End Product and Deliverables

The hardware and software for a CubeSat will be delivered to M:2:I, who will hand the CubeSat off to CubeSat testing for prelaunch testing, after which the CubeSat will be launched to the International Space Station for deployment. The deliverables for this project are each of the subsystems; the expectations for each, as well as their tentative due dates, are laid out below.

The Ground Station (April 2020)

The Ground Station software will be installed on a desktop device in Howe Hall. The Ground Station will allow users at M:2:I to send commands to the satellite, and receive and store data from the satellite.

The Payload SDR (April 2020)

The payload's SDR system will be connected to the satellite via the OBC using UART. This SDR will be capable of collecting data with the radiometer and capable of transmitting the data back to the OBC. Though most of the subsystem will be partially completed throughout the Fall 2020 and Spring 2021 semesters, the finished payload subsystem will be ready by April 2020.

The ADCS (April 2020)

The ADCS must be able to switch between modes of operations such as active and passive detumbling. ADCS will send telemetry data to the OBC to be recorded / sent to the ground station via radio. ADCS will have designated modes of operation for gathering scientific information and transmitting to the ground station based off of orbit.

The UHF Radio (April 2020).

UHF Radio subsystem must be able to receive instructions from the ground station. Take the instructions and communicate instructions to OBC. OBC will then send the requested data back to the ground station through the UHF Radio.

The EPS (March 2020)

The EPS provides power to the rest of the satellite and reports its heath to the OBC. It controls these systems differently based on the operating temperature and the battery voltage level. The batteries are charged by the satellite's solar panels.

The OBC (April 2020)

The OBC is the heart of the entire CySat. The OBC must be able to communicate to all the different subsystems. It must be able to efficiently read messages and give commands in order to achieve the CySat's purpose of successfully relaying moisture data back to Earth.

The Voltage Boost Board (January 2020)

The voltage boost board take a 5V input from the EPS and amplifies it to 7.4V. This 7.4V is required for the ADCS.

Handoff Documentation (May 3rd)

If the CySat is not completed by the end of the Spring Semester, the responsibility to finish the project will be on next years' CySat team. As we approach the deadline to hand the project off to M:2:I, we will evaluate whether the required functionality is in place. If it appears that we may not finish, we will create handoff documentation for next year's team.

2 Project Plan

2.1 Task Decomposition

Due to the nature of the CySat, we will be decomposing tasks by subsystem. Each subsystem comprises a few subtasks. In general, the subtasks for each subsystem appear in order of their planned completion, and later tasks depend only on earlier tasks in the same subsystem. Where communication endpoints are not available for a subsystem, PuTTy or other serial transfer application, alongside a development board, will be used to test the implementation of the communication – in this way, we will avoid dependencies between subsystems during the development process.

2.1.1 Ground Station

1. Test/Implement Ground Station Communication

   The Ground Station must be able to communicate over a serial UART connection. The Ground Station has a custom packet protocol whose partial implementation must be tested and expanded upon as part of this task.

2. Test/Implement Ground Station Command Protocol

   The Ground Station must implement a list of common commands. This will require communication with M:2:I to determine which commands are desired for each subsystem. These commands are things like requests for the EPS battery voltage, or request that the Payload send data.

3. Implement Data and Command Logging

   The Ground Station must write and store text file logs of sent and received data and commands.

4. Implement Subsystem Health Visualization

   The Ground Station should present a visualization of each subsystem that includes relevant information, such as last health check and status.

5. User Testing (Ongoing)

   On an ongoing basis the Ground Station will be shown to members of M:2:I, who will test the Ground Station both for functionality and for usability. Results will drive UI and functional changes on an ongoing basis.

2.1.2 UHF Radio(Ultra High Frequency Radio)

1. Receive and send packets from computer to radio for debugging

   Initial command and packet structure creation and testing. One set of commands will be written in C for communicating via I2C with the OBC. Another similar set of commands will be written in python for communication with the Ground Station via UART.

2. Receive and send beacon and packets to Ground Station

   The UHF and Ground station should be able to communicate via UART without serial connection. The Ground Station will be integrating the commands created in task 1 to communicate.

3. Receive and send commands to OBC

   The UHF and OBC should be able to communicate via I2C without serial connection. The OBC will be integrating the commands created in task 1 to communicate.

4. Receive and send packets from Ground Station and OBC

   Integration between the Ground Station and the OBC. Ground Station should be able to transmit a request to the UHF radio which sends it via I2C to the OBC and vice versa.

2.1.3 EPS(Electrical Power System)

1. EPS Communication

   The EPS needs to be able to receive commands from the OBC and return health checks.

2. EPS heath checks

   The health checks must include accurate readings and proper units for various elements. Currently, these elements are as follows: battery temperature, voltage and current inputs and outputs, charging status, and battery voltage level. Additional measurements may be added to this list in the future.

3. EPS charge and discharge

   The charging capability of the EPS needs to be tested with the solar panels fabricated by M:2:I. Upon verification, the EPS must be able to keep up with the power budget created by M:2:I.

4. EPS battery protection

   The EPS needs to be able to operate as specified within the satellite's 4 modes of operation. The EPS must be able to activate the heaters or initiate charging based on the data provided in the health check.

2.1.4 Payload

1. Payload Communication

   The software defined radio of the payload will be transmitting data collected by the radiometer through UART to the OBC.

2. Payload Data Collection

   The radiometer will be using a GNU radio to collect data from the surface of the earth via commands from the OBC, which then transfers the data using the UHF antenna.

3. Payload Functionality

   The payload will be powered by the EPS and require the OBC to use any of the data collected by the radiometer.

2.1.5 ADCS(Attitude Determination and Control System)

1. Storing Telemetry data when out of ground station range

   The ADCS will store telemetry data to the onboard SD card when not within range of the ground station. This way, the properties of the orbit can be tracked and handled as soon as the satellite is within range of the ground station.

2. Mode activation control

   The ADCS will have multiple modes of operation defined to handle different types of orbital spin. The ADCS reports telemetry data to the OBC which will decide which operation mode to run the ADCS. During the initial orbit, the ADCS will not try to correct the orbital spin to avoid worsening the spin with faulty sensors.

3. 8-bit Health check

   The ADCS will perform a health check directly after the deactivation period follow launch. This a testing criterion for the mock launch, and is used to demonstration proper connection with the OBC.

4. Re-entry

   At the end of the satellite mission, the ADCS must send the satellite into the upper atmosphere to burn up on re-entry and not add to the amount of pollution orbiting Earth.

2.1.6 OBC(On-Board Computer)

1. OBC Communication with Interrupts

   The OBC must be able to receive and send commands to each of the different subsystems: Ground Station, EPS, UHF Radio, ADCS, and SDR Payload. The OBC must be able to communicate to the EPS, UHF Radio, and ADCS through a single I2C bus, and the OBC must be able to communicate to the SDR Payload over UART. The OBC will communicate to the Ground Station through the UHF Radio. Once everything is communicating, interrupts must be put in place. Specifically, there will be UART and I2C receive interrupts. This will allow the OBC to not have to be idle while waiting for responses.

2. OBC Bootloader

   A stretch goal for the OBC is to develop a bootloader so it can receive live patches from the Ground Station. The implementation of a bootloader is still being investigated, and may or may not end up in the final project depending on if there is time to develop this before April.

2.1.7 Voltage Boost Board

1. Voltage Boost Board Construction and Testing

   The voltage boost board must be able to take a 5V input from the EPS and convert it to 7.4V that the ADCS requires. The PCB has already been fabricated and the components have already been purchased. All that remains is the soldering of the components and testing of the board's functionality.

2.2 Risks And Risk Management/Mitigation

Many of the risks associated with this project are difficult to mitigate, as they are related to the satellite's performance in space. The mitigation for these types of risks is to test extensively, consistently, and well. Below are a few of the risks our team has chosen to identify based on current knowledge of the CySat project.

Risk	Explanation	Estimated Probability
Loss of communication with Ground Control	Risk of losing communication with the ground control station based off of the tumbling of the CySat and the doppler shift	10% Mitigated by having passive detumbling modes for when communication cannot be established
Difficulties setting two-way communication with a single radio module	Radio can be set as a transceiver on the CySat, but previous teams have had difficulty with this configuration	60% Mitigated by installing a second radio system on the CySat.
Connecting OBC to other subsystems	When combining separate subsystems there is a possibility of the connection messing up already working functionalities	50% Since the 6 subsystems are communicated through Git, we can just rollback the error the connection caused.
Loss of power	Battery operated CySat could lose power mid operation.	20% CySat has on board solar panels to recharge its battery EPS sends battery status to OBC
Task exceeds expected time	We expect sometime in the project to have a task or tasks which will need to be re-evaluated based off of the difficulty they are currently presenting.	90% Mitigated by using the scrum workflow and by communication blockers throughout the team / Dr. Jones to solve pending issues.

Table 1: Risks and Risk Management

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestones are generally grouped by associated subsystem. Some milestones refer to multiple subsystems and their intercommunication.

Milestone: Ground Station Communication

Metrics: Ground station sends/receives packets over serial UART connection with local radio. Messages will be transmitted to the Ground Station send through the UHF Radio, and back to the Ground Station.

Evaluation Criteria: Ground Station will not fail to send or receive any packets over the local Kenwood radio.

Milestone: Ground Station Logging

Metrics: Ground Station logs all commands and data sent/received

Evaluation Criteria: The Ground Station will log all data in order that it is produced/received

Milestone: Ground Station Health Visualization

Metrics: Ground Station accurately reports last received state of health checks from each subsystem.

Evaluation Criteria: The Ground Station must report 100% accurately the received state of subsystem health checks.

Milestone: Communication with OBC

Metrics: The OBC must be able to communicate to the EPS, UHF Radio, and ADCS through a single I2C bus, and the OBC must be able to communicate to the SDR Payload over UART.

Evaluation Criteria: Everything must be able to communicate seamlessly

Milestone: ADCS has multiple operational modes defined

Metrics: Code for operational modes are implemented, such as active and passive detumble. Operate within specified **

Evaluation Criteria: Operational modes are defined based off of the operation mode control flow for ADCS. Each operation modes takes in telemetry data as well as OBC command data.

Milestone: OBC Communication Optimization

> **Metrics:** Creating interrupts for UART and I2C will increase the speed and response time of the OBC to outside communication from other subsystems.
>
> **Evaluation Criteria:** The response time of the OBC will increase by ~50%

Milestone: Voltage Boost Board Completion

> **Metrics:** Complete the voltage boost board to properly transform 5v to 7.4V.
>
> **Evaluation Criteria:** Connect the boost board to a dummy load, test it with the expected current load for the ADCS. The transformed voltage must be within 0.1V.

2.4 Project Timeline/Schedule

Our project timeline is grouped according to subsystem. This timeline delineates tentative due dates for numbered tasks from section 2.1. As we are using a SCRUM environment with week-long sprints, we afford ourselves the ability to change course and add/subtract both tasks and subtasks while maintaining a structured approach to project completion. Below is the initial, tentative breakdown of tasks:

Figure 1: Task Breakdown

2.5 Project Tracking Procedures

Our team will use a Gitlab repository for version control. We will use Gitlab's issue boards to keep track of tasks and their relevant commits and branches, and to build and maintain a backlog of tasks. See section 3.6 for a more in-depth discussion of the specific development process we will adhere to over the course of the project.

We will also be using slack for scheduling meetings, communicating about tasks, and asking questions. The repository will be managed by Jeffrey Richardson.

Finally, we will be producing weekly status report detailing tasks performed, hours worked, current impediments to progress, and planned work for the next week to our faculty advisor as well as to M:2:I.

2.6 Personnel Effort Requirements

Our tentative personnel effort requirements are laid out below, based on our initial understanding of project requirements and knowledge of the previous CySat teams' committed hours:

Substem	Task	Description	Hours
Ground Station	UHF Radio Communication	Sending, receiving, interpreting, and responding to CySat data	20
	Database Implementation for Payload Data	Permanent storage of Payload data, stored chronologically	25
	Database Implementation for Logging Data	Permanent storage of Commands Sent/Received and other Logs	15
	Common Command Implementation	Implementation of common commands to be sent to CySat	20
	Custom Command Implementation	Implementation of custom user command creation and use	12
	Ground Station Visualization Capabilities	Visualization of Satellite/Subsystems	20
	--	Total hours for Ground Station	112
UHF Radio	Receive and send packets from computer into radio for debugging	Hello world, Packet structure, addtional functional	30
	Receive and send health check to OBC	Make sure UHF is running properly	5
	Receive and send commands to OBC	Prompts OBC to access data or communicate with other subsystems	25
	Receive and send beacon and packets to Ground Station	Line of communctation between the satellite and its users	25
	Receive and send packets from Ground Station and OBC	Integration between OBC and Ground Station communcation	25
	--	Total hours for UHF Radio	110
Payload	Transfer mode/capture mode of SDR	Sending/collecting data via the SDR	20
	Communication to/from OBC	Being able to send or receive commands, data, etc between SDR & OB	25
	Radiometer application work	Getting application to run on embedded Linux start up	20
	UART functionality	UART communication testing and completion	10
	Coding OBC functionality with SDR	Programming the OBC to be able to command the SDR using UART	20
	Coding OBC functionality with UHF antenna	Programming the OBC to transfer data using the UHF antenna	20
		Total hours from Payload	115
ADCS	Storing Telemetry data when out of ground station range	Recording telemetry dat for when the CySat is within range	12
	Mode activation control	Major component of programing the ADCS, flow control for op modes	40
	Set magnetometer configuration	Comput magnetometer offset and sensitivity matrix	12
	Y-Wheel Ramp-up Test	Testing for Y-Wheel Ramp-up Test	12
	Y-momentum mode commissioning 1 & 2	2 stages for Y-momentum mode commissioning	12
	Sun/Nadir sensor test	Testing of sun sensor for determining position	12
	Active comms control	Ground station control operational modes	12
	8-bit Health check	Additional time for health check of the system	15
		Total hours for ADCS	127
EPS	Communication to/from OBC	Update I2C to new version	40
	Health check	New I2C protocols add more parameters to be checked	25
	Charging and discharging	Measure and calculate the energy v. time of the batteries	30
	Battery protection	Change operation based on the data from the health check	15
	--	Total hours for EPS	110
OBC	Communication to all Subsystems	Sending, receiving, interpreting, and responding to all other subsyster	30
	Optimization	Look into FreeRTOS and use interrupts for UART and I2C	30
	Live Updates	Create bootloader that allows for live patches	30
	Land Tests	Implement mock mock launch, mock launch, and mock mission	50
	--	Total hours for OBC	140

Figure 2: Personnel Effort Requirements

Hourly time commitments are based on our understanding of the complexities of each subsystem, as well as their current states, and our estimations of the amount of work still needed for their completion. These hourly estimates are subject to change as the project moves forward.

2.7 Other Resource Requirements

This project is in direct collaboration with M:2:I. The M:2:I team has and will continue to assist us with access to appropriate hardware and lab time. They will also give us feedback on what needs to be done, and how it affects the overall satellite.

2.8 Financial Requirements

The CySat's finances are managed by M:2:I. The CySat senior design team will make no decisions that affect these finances.

3 Design

3.1 Previous Work And Literature

The CySat software project was inherited from last year's senior design and was originally started by Iowa State's M:2:I in 2017. As such, the design process has already been completed. Due to the project being several years old, the records of previous work are extensive and accessible through CyBox and Iowa State's ECE GitLab. Due to the completed nature of the CySat's design, citations below are references to likely sources of information for the design process as M:2:I designed the CySat satellite. The associated websites, which can be found in the references section for each cited number, contain detailed information about the initiatives, products, and documentation relevant to the design of the CySat.

CubeSats have been in use for about 15 years, and all are fairly uniform in terms of design and operation. The CubeSat initiative is meant to provide "opportunities for small satellite payloads build by universities, high-schools, and non-profit organizations to fly on upcoming launches" [1]. It is meant to provide a means for non-professional organizations to work on solving problems in satellite design, as well as obtain potentially valuable scientific data for their own purposes or goals.

Part of what makes CubeSats special is that they conform to a strictly defined set of design parameters, which makes the process of meeting the NASA requirements simpler for CubeSat designers. Accordingly, a number of commercial companies supply various components that are CubeSat compatible. In this case, several of the CySat's subsystems are from the company EnduroSat[2], and the CySat's research focus is the health of vegetation.

It is important to keep the shortcomings the previous CubeSats in mind, because a very large percentage of all CubeSats fail. The goal of last year's senior design team was 'to not have a paperweight in space,' and it is being followed this year as well [3].

3.2 Design Thinking

There is little to no room for design thinking in this project. CySat is a 3U Cube Satellite, which have rigid design specifications that must be met in order for NASA to approve inclusion of the satellite into the payload. In addition, the CySat as a project has been in development since 2017 under of the supervision of the M:2:I community at Iowa State. The operational mission and internal hardware design of the satellite were specified long before our senior design team took on this project, and specific modules were selected from EnduroSat products available for CubeSats for the purpose of this project.

As such, we are making no high-level design decisions. Instead, our team's focus is on the implementation of commands in the various subsystems to accomplish the goals listed for each subsystem in section 2.1 Our team's goal is implementation of previously determined M:2:I designs, which themselves were defined based on the CubeSat specifications and available EnduroSat components.

3.3 Proposed Design

As indicated in earlier sections, each of the modules below, aside from the debugging computer, which is used only in development, are subsystems of the CySat satellite. These subsystems are standard fare for CubeSats. The SDR Payload is specific to the CySat.

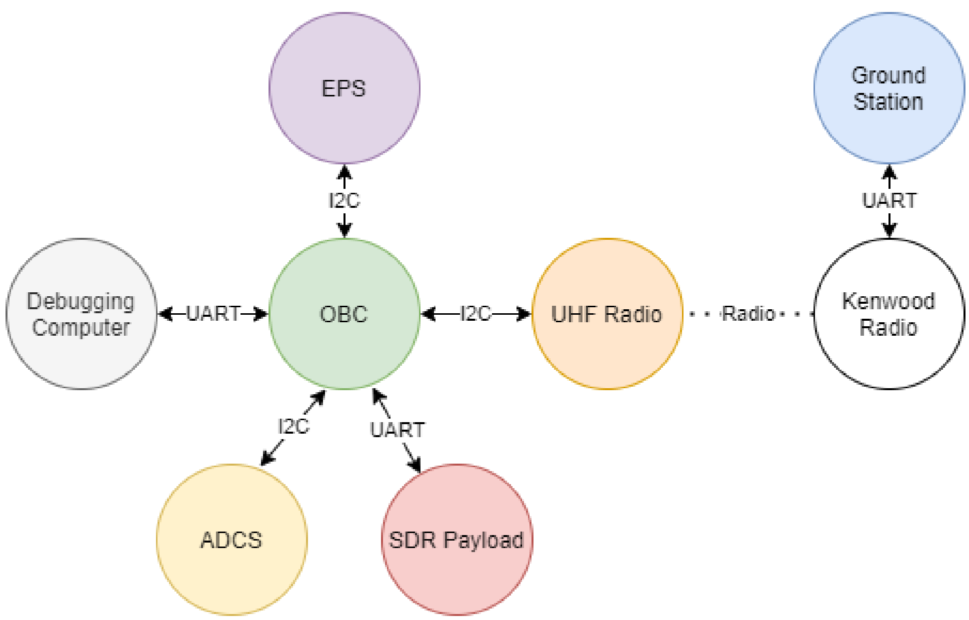

Figure 3: CySat Subsystem Design

The ADCS, EPS, and UHF are all on the same I2C bus because this simplifies the communication protocol between the OBC and those subsystems. The SDR is connected over UART because the FPGA for the SDR only has a UART connection. Additionally, the OBC has only two UART connections – one of which will be used for a connection to a debugging computer while in development.The Ground Station is connected to the Kenwood radio over UART as that is the only available serial connection with which the PC running the Ground Station may interface. The ADCS is of particular importance here, as the SDR payload requires that the satellite be pointed towards earth in order to function. Although many CubeSats do not include the ADCS, the CySat does, as the ADCS will be needed to orient the satellite so that the SDR can take data. The connections in Figure 3 are referenced below in relation to which requirement they help to fulfill.

Requirement	**Relevant Connections**	**Explanation**
Must power up after been deployed from the International Space Station	OBC/EPS	The OBC and EPS will communicate to ensure that the satellite has power and powers up after the required 30-minute waiting period after deployment from the ISS
Must stabilize and point itself towards earth	OBC/ADCS	The OBC will communicate with the ADCS to begin and maintain detumbling and orientation
Must take soil moisture readings from Earth via a microwave radiometer	OBC/SDR Payload	The SDR collects data, which the OBC will retrieve using UART and then compile
Must transmit data back to the ground station in Ames, IA	OBC/UHF Radio/Kenwood/Ground Station	The OBC will send data to the Kenwood Radio through the UHF, which will be received by the Kenwood radio which will communicate that data to the Ground Station
Must collect data for its orbit life (6 months)	All connections	All connection on the satellite work towards the overall mission goal of the satellite
Must meet NASA's CubeSat standards and regulations	All connections	All connections have been specified by M:2:I to conform to these standards and regulations
Must receive and execute commands issued by the Ground Station	Ground Station / Kenwood/UHF/ OBC	The Ground Station will communicate through the UART connection to the Kenwood, which will communicate with the OBC through the UHF Radio
Must successfully deorbit at the end of its lifespan	Ground Station / Kenwood/UHF/OBC /ADCS	The ground station will signal an EOL beacon to the satellite through the connections described above, and the OBC will communicate the intention to begin deorbiting to the ADCS

TABLE 2: SUBSYSTEM FUNCTIONALITY DESIGN

The non-functional requirement for the Ground Station performance is related to implementation, rather than design.

The design decision not shown in the diagram above is the CySat Packet Protocol. This design decision is somewhat between a design and implementation decision. The overview of the packet protocol is that it is a standardized packet size that allows for easy decomposing of packets to allow for routing of messages by the OBC. It comprises a 1 byte start character, a 1 byte subsystem type, a 2 byte command, a 1 byte data length field, N bytes of data, ang 1 byte checksum.

3.4 Technology Considerations

Due to the expensive nature of the satellite subsystem parts, as well as the lack of remote access to the lab, in some cases, it can be necessary to use a stand-in discovery board instead of the OBC. For this purpose, we've chosen the STM32F429ZI Discovery board. This board uses the same chipset as the OBC, but the pin outs are different. This makes this discovery board a good stand-in for the OBC, as code written on it will function the same as it will on the final OBC.

The previous CySat team used an STM32Workbench Eclipse plug-in for debugging and running code on the Endurosat OBC and the STM32F429ZI Discovery Board. Our team decided to move forward with using the STM32CubeIDE. The STM32CubeIDE allows for faster debugging (i.e. stepping over, stepping into, etc. are faster on the IDE). The STM32CubeIDE also allows for easy integration between the STM32CubeMX which allows for easy initialization of the different modules and associated interrupts (UART, I2C, etc.)

The Raspberry Pi is a good replacement for the SDR because it is capable of UART communication and also runs Linux so porting information over this and the SDR is similar.

The Ground Station UI is already partially implemented using Python 3+ and the TKinter UI library. Python, as a language, is generally used for scripting short commands, rather than for UI purposes. However, Python does have a powerful built in UART compatible library which is used to connect with the local radio, as well as a simple and effective logging library which will be used to log commands and data. Given that the Ground Station does minimal processing of data (it simply records what it receives and sends small packets) and minimal rendering of complex UI elements, Python is an excellent and accessible choice of language for the Ground Station. In addition, default Python installs include a library for creating a virtual environment in which the Ground Station can be run, which ensures library version dependencies are met.

3.5 Design Analysis

M2I was responsible for this design, and they've given us and a previous group this design to implement. The design of the satellite (given in section 3.3) meets all of the requirements laid out in section 1.4 Because this is not our design, there is little analysis to perform, and so we note here that should a design level difficulty occur, we will make this difficulty clear to M2I and work towards a solution.

The inclusion of the ADCS subsystem is a good design choice for this satellite. Though many cube satellites launch without an ADCS, the improved accuracy for the payload is worth the additional weight and power consumption.

3.6 Development Process

In order to manage these tasks, we will use a modified SCRUM process with week-long sprints. During our weekly meetings with our faculty advisor and M:2:I members, we will create stories, which will map to Gitlab issues, that correspond to required functionalities for the CySat. A story encompasses a single, small unit of functionality. In the case of the Ground Station, for example, this could be the implementation of page wherein users can send signals to a specific subsystem. Each story will have an associated checklist comprising required functionality, documentation, and a brief description of the task. We will then order these stories by priority, again in discussion with M:2:I and our faculty advisor. Following this meeting, the CySat team will agree upon a number of story points for each story. These story points will roughly equate to person-effort-hours, although the relationship will not be 1 to 1.

At the beginning of each sprint, the CySat team will meet in order to select from the prioritized backlog a number of stories that will be worked during the sprint. At the end of each sprint, we will evaluate how many points have been completed, and how many have fallen through. In this way, we will be able to quantitatively track progress on tasks and address slow-downs and unexpected blockers as they arise. In order to determine whether a story is complete, we will use a definition of done. For our team, this means that a completed story meets the following criteria:

1. The code functionally meets the stated requirements in the story
2. The code has been reviewed by at least one other CySat team member, and one of the subsystems lead or sub-lead.
3. The code is documented
4. In the case of documentation tasks, the documentation is understandable and exhaustive
5. Other functionalities for the associated subsystem remain in place

Each story will be worked in a Git branch. Branches will require review and must meet the above definition of done before they are merged into master. When a team member places a task into review, they must include with it test procedures which will outline a manner in which others may test the code. In the case that tests cannot be performed by other team members, a PowerPoint or document must be included showing screenshots of the functionality. Code reviews will consist of team members inspecting the code for potential defects as well as confirming that they are able to perform the functions that are part of the test procedures.

In addition to the above, the team will engage in daily “stand up meetings” consisting of a short check in with each team member, during which they explain their plans for the day and reveal any blockers that may stand in their way. These meetings may be conducted over slack, requiring only that team members update the team daily with a short status report.

The SCRUM approach described above allows for consistent communication between members of the CySat team and the M:2:I community, and the short sprint time ensures that if a task fails, or falls behind, the team will be made aware of it quickly and can remove the blockers that prevent the task’s completion.

Once necessary functionalities an in place, we will enter a testing phase, during which we will perform acceptance testing, described in section 4.3, as well as extensive manual testing of the interfaces between the subsystems.

Below is a diagram outlining the CySat team process. The initial stage has been completed, and we will be entering the development stage over winter break and into the coming semester.

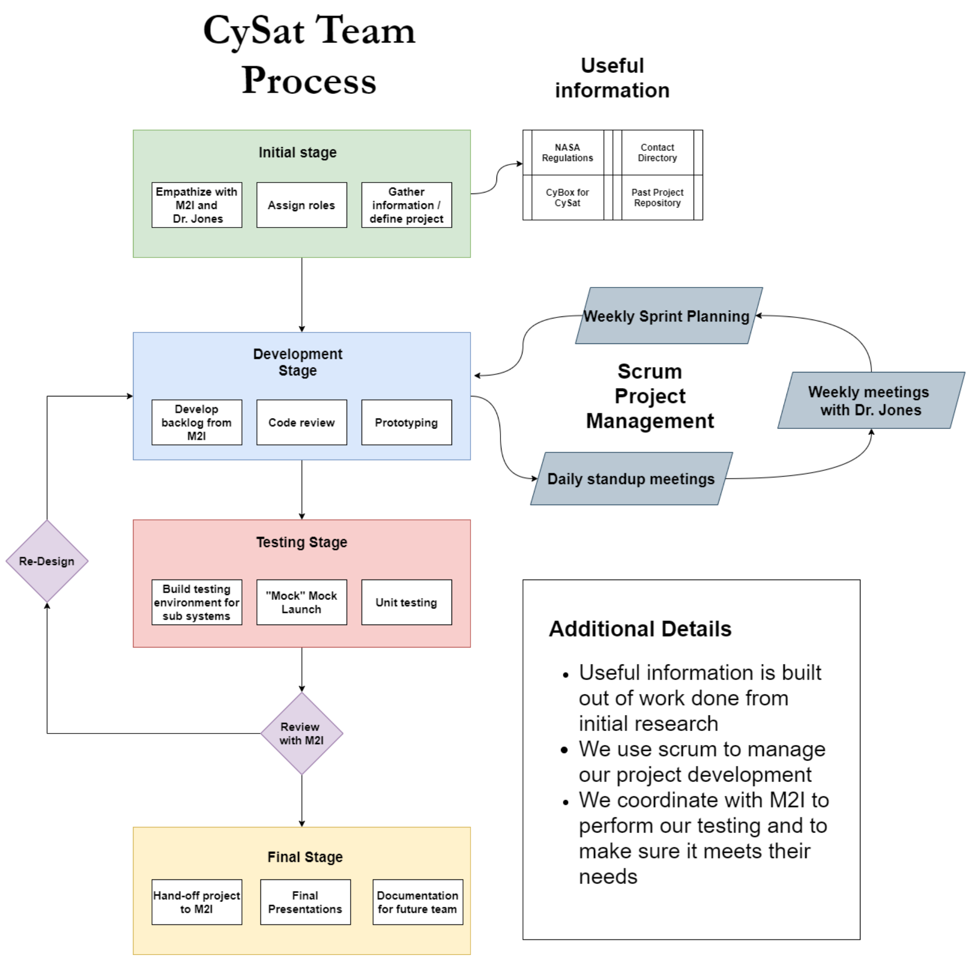

FIGURE 4: CYSAT TEAM PROCESS

3.7 DESIGN PLAN

The proposed design from section 3.3 is the design plan for the CySat project, as the design was decided upon by M:2:I prior to our involvement in the project. Its relationships to functional requirements are shown in the table in section 3.3, and will not be reiterated here in the interest of space.

The subsystems will be implemented in parallel over the coming months, with frequent checks to confirm and reconfirm the continued performance of previously implemented functionality, as described in Section 3.6, and further in the following section.

4 Testing

4.1 Unit Testing

Due to the nature of the CySat subsystems, there is little need for unit testing – the functionalities of the subsystems are largely related to communications between the subsystems, which is in the purview of interface/integration testing.

However, the Ground Station UI will be able to be unit tested. For the Ground Station, each time new functionality is provided, an associated test must be included. These tests will ensure that each request packet type is created correctly, as well as that received packets (whose origin will be mocked) are decoded correctly. These test cases will be run and must pass before any changes to the Ground Station are merged into the master Gitlab branch.

4.2 Interface/Integration Testing

The communications interfaces between different subsystems will be tested using development boards which will emulate the other side of the connection. We will adhere to a strict communication protocol, as well as ensure that emulated connection types are the same as final design connection types. This will help to ensure that once all components are fit together before the "mock mock launch" (described below) they will communicate in expected ways. This testing will be manually performed by the CySat senior design team as we implement the project, and consistent communication between subsystem leads will ensure that, if discrepancies arise between expected communication and actual communication, those discrepancies are solved before moving forward.

Additionally, the performance of the EPS will be evaluated while powering the other subsystems of the satellite. It should be capable of performing up to the standards listed in the power budget created by M:2:I.

4.3 Acceptance Testing

Our main testing will involve connecting the OBC, EPS, ADCS, SDR, UHF Radio, and Ground Station. The EPS, ADCS, SDR, and UHF Radio must be able to properly communicate with the OBC. They will receive commands from the OBC, and they must perform the required task properly. Additionally, the OBC must be able to schedule tasks for each of these subsystems. The UHF radio will be relaying commands between the OBC and the Ground Station.

The first test we will be running is called the "Mock" mock launch. This tests the communication between the OBC, the UHF Radio, and the Ground Station. The steps for this test launch are as follows:

1. Connect to Handheld Remote Machine(Ground Station)
2. Connect to OBC/UHF Remote Machine(Satellite)
3. On the satellite, power the pumpkin board with the power supply
4. Flash the OBC with the most current software
5. Power off the power supply
6. Open the Ground Station application on the Ground Station computer
7. Connect to the Handheld radio on the Ground Station
8. Power on the power supply and the following events should be observed automatically:
    a. OBC is powered on

b. OBC simulates from handoff to NASA until power on of OBC
    i. Prints "RBF inserted"
    ii. Prints "Handoff to NASA…Loading into P-Pod"
    iii. Prints "Kill switch depressed"
    iv. Prints "RBF removed"
    v. Prints "Kill switch released (EPS power on)"
    vi. Prints "Entering Main()"
c. OBC starts a 1 minute timer and waits, simulating the 30 minute period of inactivity following deployment from the International Space Station
d. OBC turns on other modules in Satellite
    i. Simulates turning on UHF transceiver by printing "Commanding EPS to enable Output 5 (UHF transceiver)"
    ii. Simulates turning on SDR by printing "Commanding EPS to enable Output 3 (SDR)"
    iii. Simulates turning on Boost board by printing "Commanding EPS to enable Output 1 (boost enable)"
e. OBC simulates deployment of magnetometer by printing "Commanding ??? to deploy the magnetometer"
f. OBC simulates deployment of antenna by printing "Sending 0x1F (deploy all antennas with algorithm 1) to I2C slave address 0x33 (antenna)"
    i. Optional: Read the states of the UHF antenna and then deploy via algorithm 2 if needed
g. OBC will send commands to the UHF Radio to configure the beacon text and period "Hello, Earth! I am ISU's CySat-I" with period of 1 minute
h. OBC will send command to the UHF Radio to enable transparent mode
i. OBC will send command to the UHF Radio to enable the beacon
j. OBC will simulate beginning the detumbling sequence by printing "Beginning detumbling sequence"
k. At some point after step h, the Ground Station will receive the beacon being sent by the Satellite. At this point the beacon should send a "Beacon Shut Off request" command as described in the Ground Station | Cysat Packet Protocol documentation.
l. At some point after step k, the OBC will receive the "Beacon Shut off request" command
m. OBC will turn off the beacon
n. OBC will send a "Beacon shut off response" command
o. Ground Station sends an "Initial Health Check Request" command
p. OBC will run initial health checks (8-bit good/bad flags for each subsystem) and compile them into a response
    i. The EPS, ADCS, and SDR will simulate their health checks; we will assume they are all good.
    ii. The OBC and UHF Radio will run their actual health checks
    iii. OBC will send packet to Ground Station
q. Ground Station will receive initial health checks and display them to the user.
r. Ground Station will send a "Enter Main Operating Phase" request
s. OBC will receive this and send a "Enter Main Operating Phase" response packet back to the Ground Station
t. OBC will enter the main loop
    i. OBC will print "Main operating loop entered"

After this test passes, we will be stop simulating the EPS, ADCS, and the SDR payload, and these systems will be added to the tests. These more complex tests still need to be constructed along with expected results. In the end, we will form a mock launch and a mock mission with all subsystems communicating with each other as we would expect in space.

4.4 Results

The Ground Station was provided with no associated unit tests. The unit tests for the Ground Station are being written and will provide a solid basis of assurance that the Ground Station is adhering correctly to CySat packet protocol.

We have completed initial interface testing for the Ground Station and the OBC. We have tested that each is able to communicate to properly over UART and have found no bugs or improper behavior yet.

We have not been able to do much more testing than mentioned above, as the COVID-19 pandemic has prevented us from being able to be present in the lab where the subsystems are located. Our team is working with M:2:I to establish remote access capabilities so that we may test the current functionalities of the remaining subsystems.

5 Implementation

Our team will have weekly sprint meetings during the winter break. Before the end of the winter term, our team will have the boost board components soldered and tested, and all subsystems established with remote access.

The Ground Station UI is largely implemented – what remains is the implementation of additional commands, the logging of those commands, and the visualization of subsystem health for the user. The UHF Radio has a list of commands from the data sheet which need to be implemented. What needs to be implemented is the commands between the UHF and the Ground Station. The ADCS contains the software for running each operational mode, what needs to be implemented is the code that determines which operational mode to run. The Payload radiometer and data collection has already been implemented. The emulation to replace the SDR until access to the real SDR is possible is almost set up. Which will allow the payload lead to finish communication with the OBC.

A stretch goal for winter break is the completion of a successful mock launch of the satellite. We want to work during the winter semester to get ahead in the project before the start of our implementation period next semester. After the winter break, our team will continue working towards the above goals until the handoff date. If this project is not completed before the hand off date, additional handoff documentation will be added for future engineers.

6 Closing Material

6.1 Conclusion

The goal, overall, is to produce software for the subsystems of the CySat that allows for those subsystems to effectively and consistently communicate the commands necessary to run a working satellite.

Each subsystem must be capable of consistent communication with the OBC, including the Ground Station. This communication must be implemented according to the design given in section 3.3, and the satellite must be able to power on, orient itself, and collect data for streaming back to the Ground Station during its lifetime, which the Ground Station will then log. After that lifetime, the satellite must be capable of deorbiting.

6.2 References

[1] Jackson, S. (2017, February 17). NASA's CubeSat Launch Initiative. Retrieved October 26, 2020, from https://www.nasa.gov/directorates/heo/home/CubeSats_initiative

[2] EnduroSat - Class-leading CubeSat Modules, NanoSats & Space Services. (n.d.). Retrieved October 26, 2020, from https://www.endurosat.com/?v=7516fd43adaa

[3] Iowa State's CubeSat Project. (n.d.). Retrieved October 26, 2020, from https://m2i.aere.iastate.edu/cysat/

6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar data that does not directly pertain to the problem but helps support it, include it here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc,. PCB testing issues etc., Software bugs etc.

The Endurosat and CubeADCS manuals can be found through their websites or in the CyBox repository for the CySat.

EnduroSat Electrical Power System – User Manual

EnduroSat Electrical Power System – I2C Protocol Revision 2.2

EnduroSat UHF Transceiver Type II User Manual

CubeADCS – Commissioning Manual

CubeADCS – Reference Manual

CubeADCS – User Manual

CubeADCS – Option Sheet

Kenwood Instructional Manual 144/440 MHz FM DUAL BANDER TH-D72A

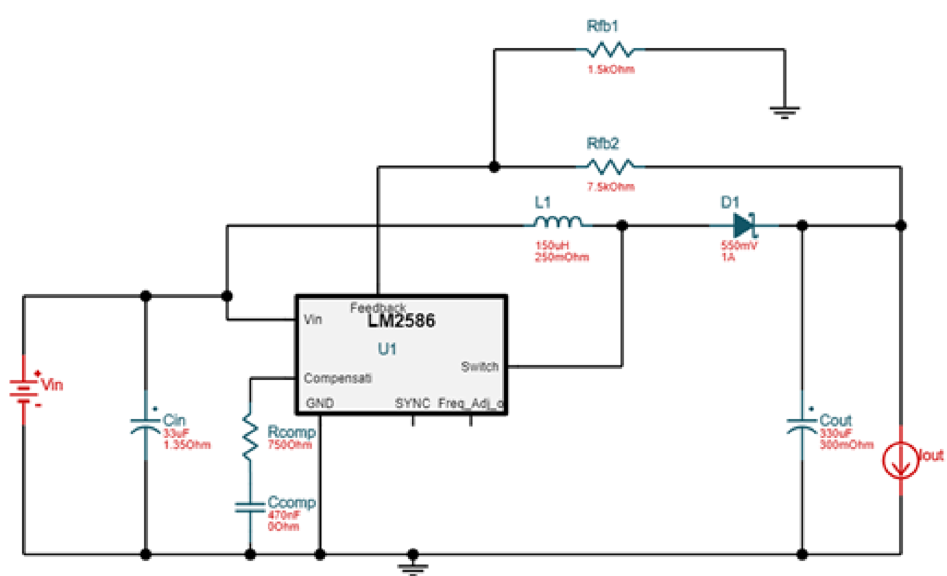

Figure 5: Voltage Boost Board Circuit Design