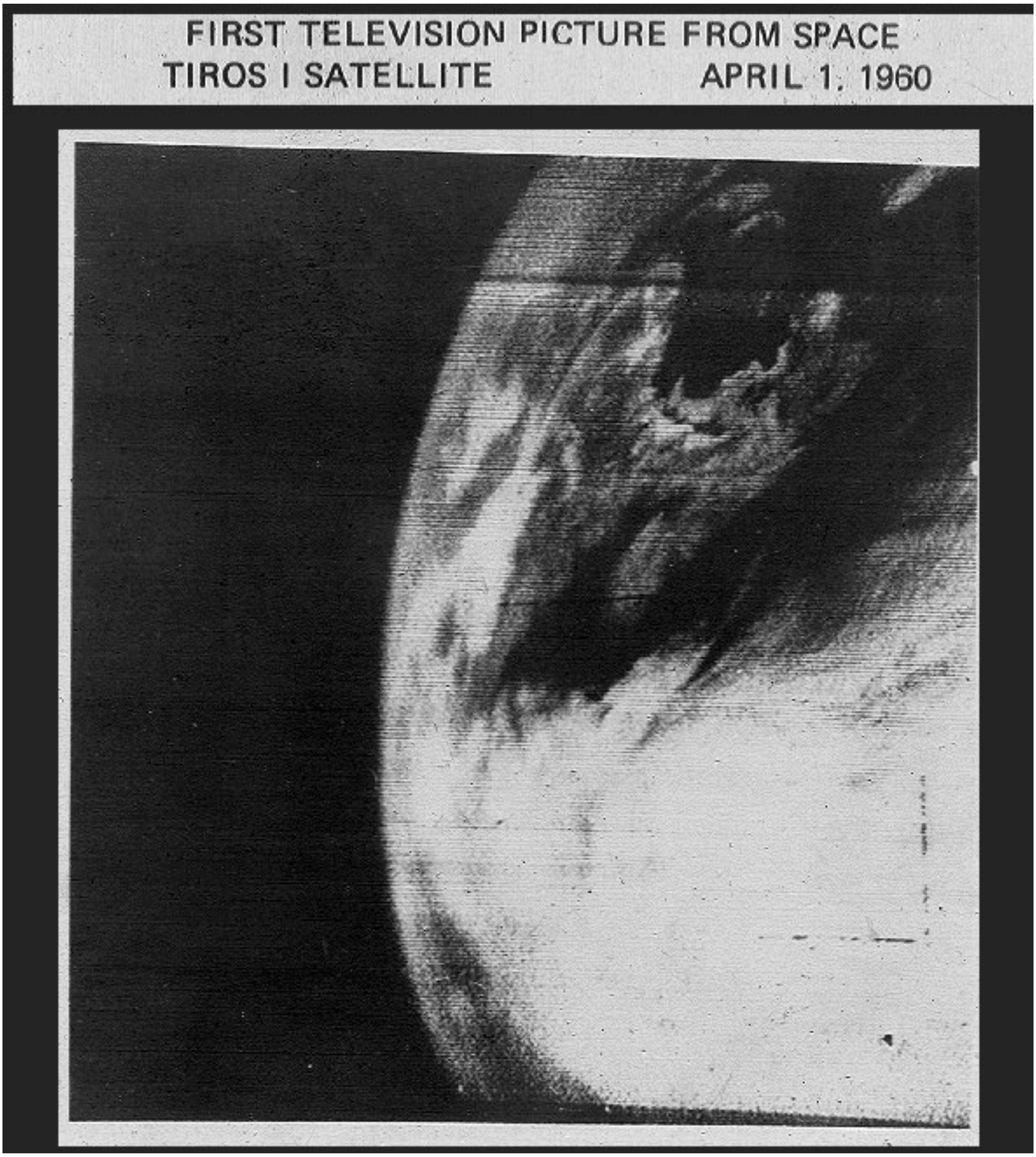

Development of 1u CubeSat

Design & Development of 1u CubeSat for educational & cost effective purpose in Nepal

Prepared by: Aashish Madhikarmi
Bachelor of Computer Engineering, Nitte Meenakshi Institute Of Technology

28 October 2025

INTRODUCTION

Background and motivation

The CubeSat standard has revolutionised the space industry by making satellite technology accessible to universities, research institutions, and independent developers worldwide. While this movement has democratised access to space, a significant gap remains between theoretical knowledge and practical application in many educational settings, especially in emerging technological hubs like Nepal. The prohibitive costs of flight-ready hardware and specialised testing equipment present a major barrier to entry for aspiring students and developers. This project was born out of a personal motivation to overcome this challenge, creating a professional level, educational platform that replicates the core functions of a satellite using innovative and affordable design principles.

Project objectives

The primary objectives of this project were to:

- Design and fabricate a structure that conforms to the standard 1U CubeSat form factor (10×10×10 cm) using custom-designed PCBs and a 3D-printed chassis.
- Develop a modular, hybrid architecture by integrating a Raspberry Pi Zero W for high-level command and data handling with an Arduino Nano for real-time sensor processing
- Implement key satellite subsystems, including a custom-built Electrical Power System (EPS), an Attitude Determination and Control System (ADCS) simulation using magnetorquers, and a payload with multiple sensors.
- Establish a functional communication link using a software-defined radio (SDR) approach with the open-source rpitx software to transmit telemetry on the 433 MHz band.
- Create comprehensive technical documentation, including schematics, code, and a detailed report, to serve as a reproducible and open-source educational resource.

EXECUTIVE SUMMARY

Project Title

Development of a Low-Cost 1U CubeSat Simulator for STEM Education

This proposal outlines the development of a low-cost, 1U CubeSat simulator to serve as a practical, hands-on tool for education in satellite engineering and STEM fields. The high cost and technical complexity of real satellite hardware present a significant barrier to entry for students and educational institutions. This project directly addresses that challenge by providing an affordable and accessible platform for learning about small satellite systems and operations. The simulator features a 3D-printed 1U structure and is equipped with functional subsystems for Command and Data Handling (C&DH), Electrical Power (EPS), simulated Attitude Determination and Control (ADCS), and Communication, along with a payload sensor. The system is designed to wirelessly transmit telemetry data to a custom-built ground station, effectively replicating a real satellite mission on a small scale. By creating this independent project, I aim to demonstrate how complex technical concepts can be made accessible through innovative and cost-effective design. My ultimate goal is to offer this simulator and its accompanying educational materials to local institutions, thereby enhancing practical space science education and inspiring the next generation of engineers and scientists in Nepal.

Problem Statement

Traditional education in satellite technology often relies heavily on theory due to the prohibitive costs and specialised equipment needed for hands-on work. This gap between theoretical knowledge and practical application limits a student's understanding and development of real-world engineering skills. It discourages promising students who lack access to expensive, institutional-level equipment from pursuing careers in the space industry.

Solution

To overcome this barrier, I have independently developed a functional 1U CubeSat simulator. This platform, built using standard 3D-printing and cost-effective, readily available electronics, offers a hands-on learning experience that accurately replicates the operational and functional aspects of a real satellite. The simulator is a comprehensive, self-contained unit that serves as a powerful and accessible teaching tool.

Methodology And System Architecture

- **System overview**

The simulator's architecture is built upon four custom-fabricated boards that stack vertically, minimising the internal volume and streamlining inter-component communication. These boards are:

1. **Onboard Computer Board:** Houses the Raspberry Pi Zero W and primary sensors for data collection.
2. **Battery Management Board:** Manages power storage and monitors usage. Integrates the CN3085 charger module and manages power storage.
3. **Stem Board:** Manages all sensor data acquisition and processes real-time data using an Arduino Nano before transmitting it to the main computer.
4. **Attitude Control Board:** Contains the magnetorquers for attitude simulation.

- **Command and data handling (C&DH)**

1. **Central Processor:** A Raspberry Pi Zero W acts as the central C&DH unit, managing data acquisition from the high-speed MPU9250 and BMP280 sensors. It handles higher-level functions, such as telemetry formatting and command execution. Its integrated Wi-Fi is used for local configuration, but the 433 MHz radio provides the primary "simulated" long-range communication.
2. **Subsystem Management:** The Raspberry Pi communicates with the Arduino Nano via I2C bus, using it to collect pre-processed sensor data and relay commands.

- **Sensor integration**

1. **Raspberry Pi Sensor Suite:** The MPU9250 IMU is directly connected to the Raspberry Pi to provide real-time, 9-axis orientation data (acceleration, gyroscope, and magnetometer). The BMP280 pressure and temperature sensor is also connected directly to the Raspberry Pi to collect environmental data.
2. **Arduino Nano Power Telemetry (Stem Board):** An Arduino Nano on the "Stem Board" acts as a dedicated co-processor for real-time power monitoring. It offloads precise timing requirements for three INA219 sensors from the Linux-based Raspberry Pi. The sensors monitor the battery pack and two simulated solar panels. The Nano's firmware collects and filters the data before transmitting a clean telemetry stream to the Raspberry Pi.

- **Electrical power system (EPS)**

1. **Battery Management Board:** Three NiMH 1.2V AA batteries are connected in series. The Arduino Nano uses one INA219 sensor on the Power Telemetry Board to monitor the batteries, providing crucial telemetry on the system's power status.
2. **Solar Power Monitoring:** This board connects to simulated solar panels.The Arduino Nano uses two INA219 sensors on the Power Telemetry Board to monitor the output from these panels, providing data on power generation.
3. **Charging:** A CN3085 NiMH charger module is integrated onto this custom PCB. The input from the solar panels connects directly to this module, which regulates the voltage and current to safely and efficiently charge the NiMH battery pack.

- **Attitude control simulation**

1. **Magnetorquers:** Two magnetorquers are implemented on the attitude control board for simulated attitude control in the X and Y axes.
2. **Control Logic:** The Raspberry Pi Zero W receives IMU data from its sensor to determine the simulator's orientation. It then sends commands to the attitude control board to energise the magnetorquers, demonstrating a closed-loop control system.

- **Communication and software**

The simulator's communication subsystem is a highly innovative aspect of this project, leveraging a software-defined radio approach to transmit telemetry. Instead of using an off-the-shelf radio module, i use a custom solution to generate the RF signal directly from the Raspberry Pi's GPIO pin. The transmission process utilises rpitx software on a Raspberry Pi to generate precise digital timing signals, configure a Phase-Locked Loop (PLL) clock to 433.9 MHz, and route this signal to GPIO4 to create a square wave RF carrier. This square wave is then filtered by a Low-Pass Filter (LPF) into a clean sine wave before being transmitted via an antenna for reception and decoding by an RTL-SDR at a ground station.

1. **Hardware:** The system uses the Raspberry Pi Zero W's General Purpose Clock (GPCLK0) routed to a GPIO pin. The signal is then sent to a custom-designed impedance-matching and filtering network on the PCB, which connects to an SMA connector and an external antenna.
2. **Software (rpitx):** The open-source rpitx software is run on the Raspberry Pi. This powerful tool generates precise timing signals that modulate the carrier wave with telemetry data.

Firmwares

Raspberry Pi Zero W Firmware

The firmware on the Raspberry Pi Zero W serves as the central command and data handling unit. Its primary role is to manage high-level functions, execute mission logic, and handle communication with the ground station. The key functions to add to the firmware are:

- **Sensor Data Acquisition:** Code to read data from the onboard MPU9250 (IMU) and BMP280 (pressure/temperature) sensors, which are connected directly to the Raspberry Pi's I2C bus.
- **Subsystem Communication:** An I2C master implementation to request pre-processed power telemetry from the Arduino Nano and to send commands for the Attitude Determination and Control System (ADCS).
- **Attitude Control Logic:** A module that processes the IMU data to determine the simulator's orientation and, based on a control algorithm, sends commands to the Arduino Nano to energise the magnetorquers for attitude correction.
- **Telemetry Handling:** Code to collect all sensor and system status data, format it into a telemetry packet, and use the rpitx software to modulate and transmit this data via a GPIO pin on the 433 MHz band.

Arduino Nano Firmware

The firmware on the Arduino Nano acts as a dedicated co-processor to offload real-time, time-sensitive tasks from the Raspberry Pi. Its focus is on continuous monitoring and direct control of specific hardware components. The key functions to add to the firmware are:

- **Real-time Power Monitoring:** A continuous loop that polls the three INA219 current/power monitors to collect voltage and current data from the battery pack and the two simulated solar panels. This pre-processed data is stored, ready to be sent to the Raspberry Pi.
- **I2C Slave Interface:** An implementation that enables the Arduino to act as an I2C slave, listening for requests from the Raspberry Pi. When a request is received, it provides the stored power telemetry or executes commands.
- **Magnetorquer Control:** A module that directly controls the DRV8833 H-bridge motor driver. It receives commands from the Raspberry Pi via I2C and sets the appropriate GPIO pins to energise the X and Y magnetorquer coils, thereby controlling the satellite's simulated attitude.

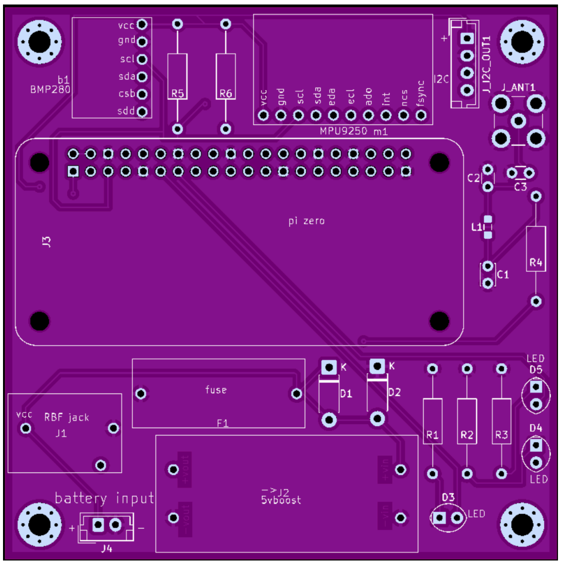

Figure 1.1 Onboard computer board

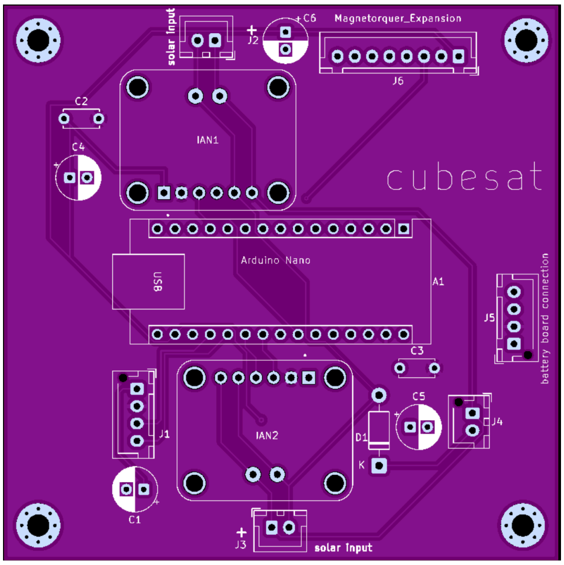

Figure 1.2 Stem board

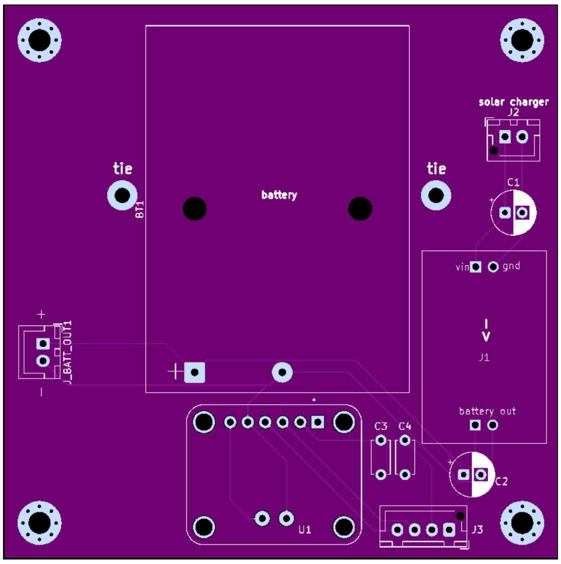

Figure 1.3 Battery board

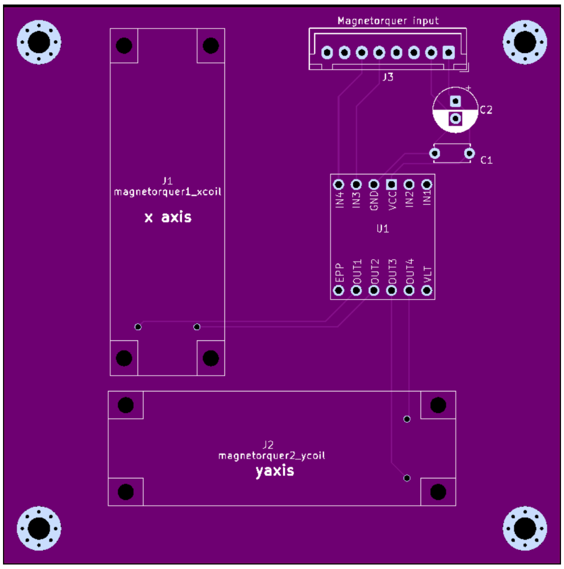

Figure 1.4 Magnetorquer

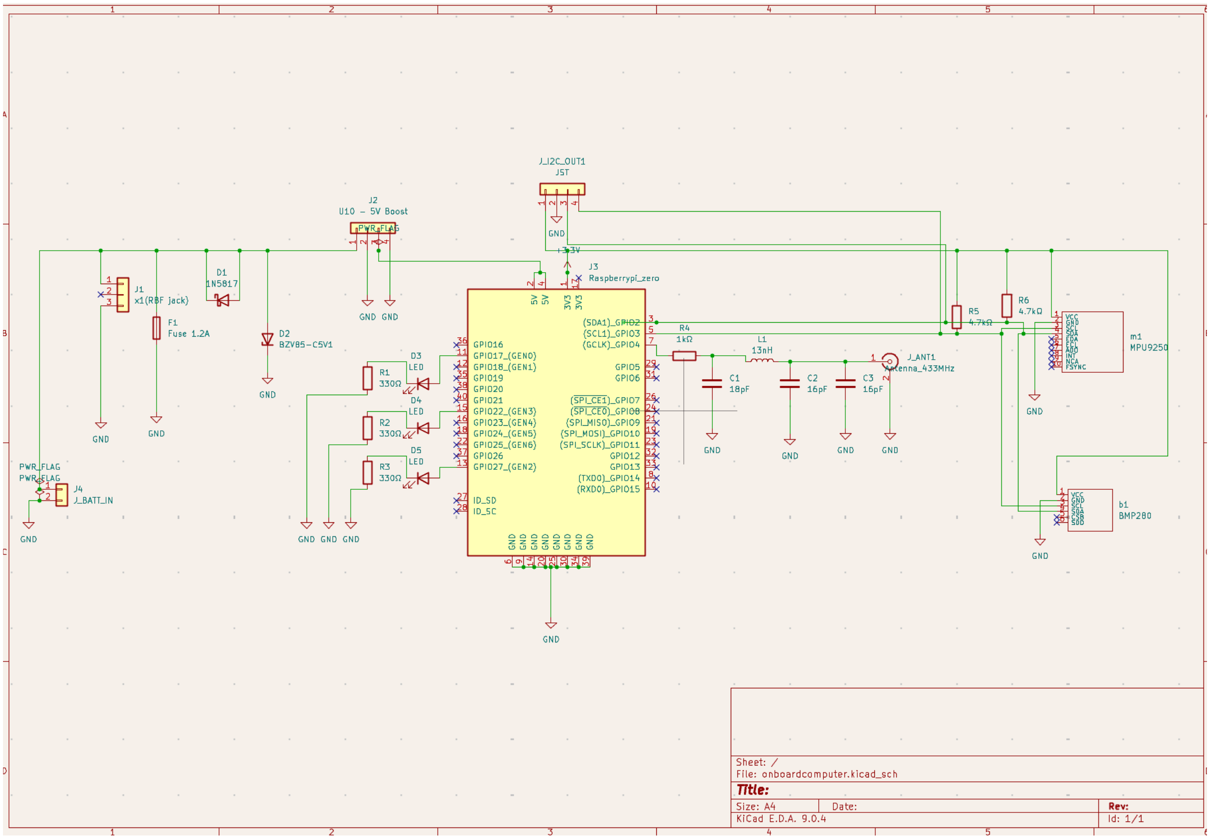

Figure 2.1 Onboard computer board Schematics

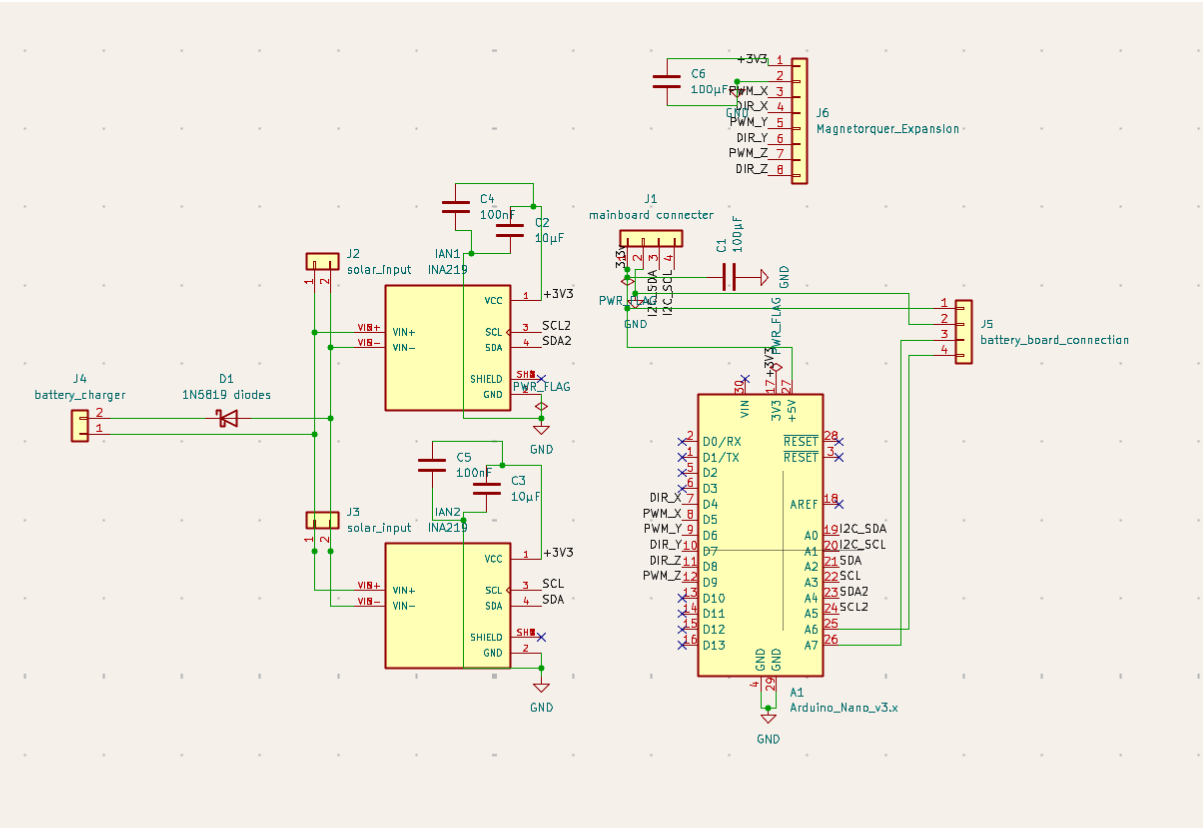

Figure 2.2 Stem board Schematics

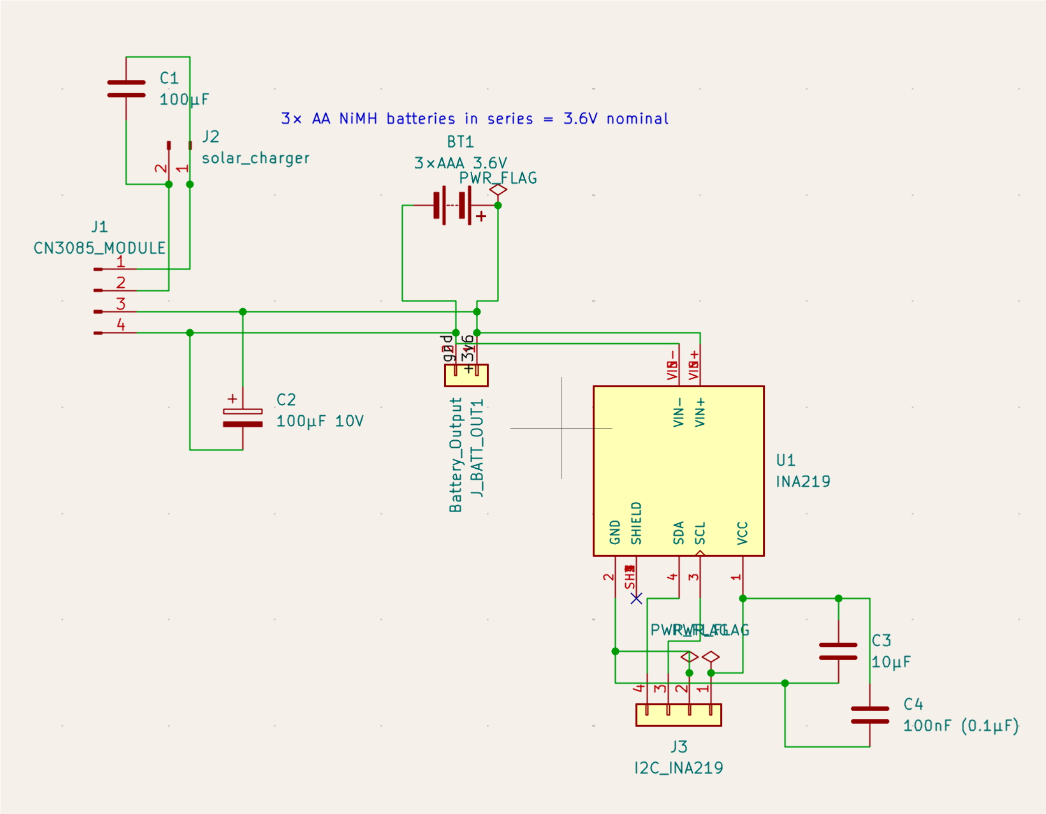

Figure 2.3 Battery board Schematics

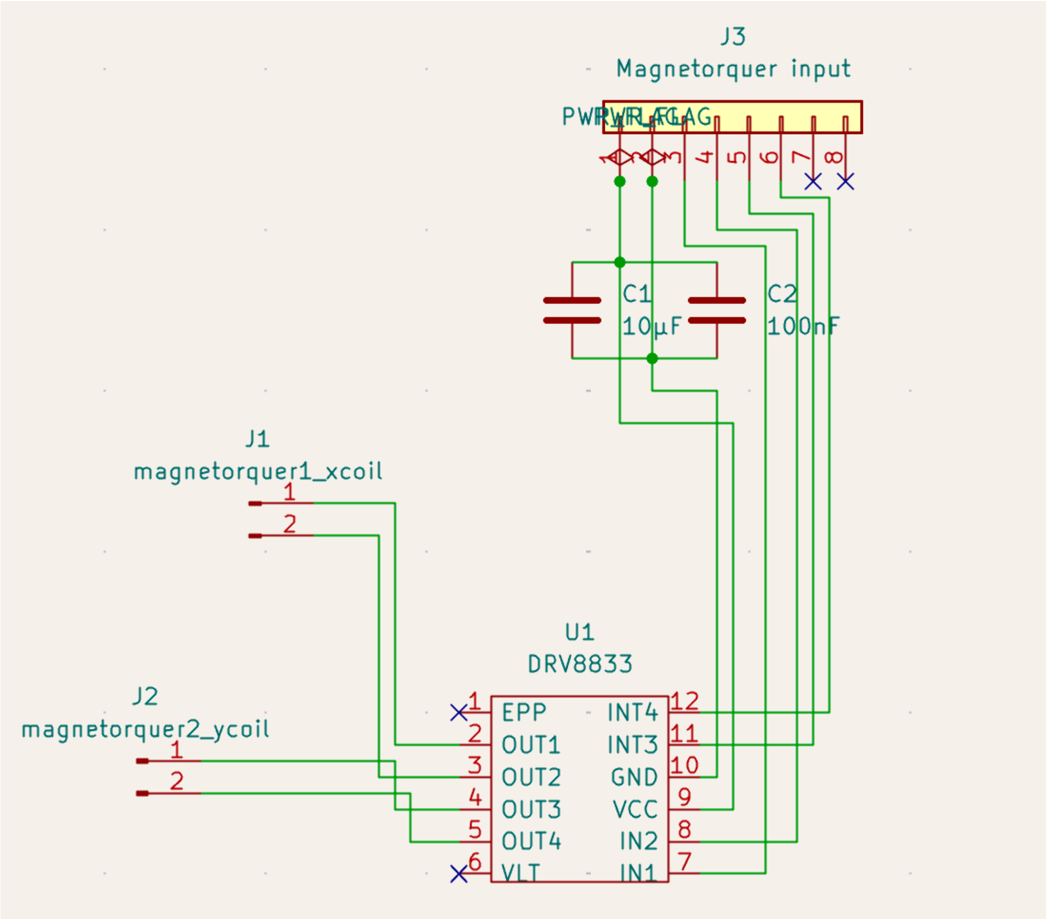

Figure 2.4 Magnetorquer board Schematics

DELIVERABLES AND TIMELINE

Deliverables

- **Hardware deliverables**

1. **Fully assembled 1U CubeSat simulator unit:** A functional, ready-to-demonstrate unit integrated into its custom 3D-printed frame, with all four boards stacked and interconnected.
2. **Ground station receiver unit:** A functional setup for receiving and logging telemetry data from the simulator.
3. **Simulated solar panels:** The physical solar panel units used to provide power and test the INA219 sensors.

Timeline

Phase	Duration	Milestone
Phase 1:Design Review	6 weeks	Finalise system architecture, complete schematics for all four custom PCBs (Pi board, Stem board, battery board, attitude control board).
Phase 2: Fabrication	2 weeks	3D print frame, order all components, custom PCB manufacturing and receive the finished boards. This timeline includes potential delays for quality checks.
Phase 3: Integration	3 days	Stack boards, wire subsystems, perform first power-on tests, and debug hardware issues.
Phase 4: Software Dev.	2 weeks	Write and test firmware for Arduino, Python scripts for Raspberry Pi, and ground station software.
Phase 5: System Testing	1 weeks	Conduct full system tests, communication range tests (using 433 MHz), and power budget validation.
Phase 6: Documentation	2 days	Write technical report and prepare all educational materials.

BUDGET

Budget Table

Description	Quantity	Price
Raspberry Pi Zero W with 40-pin GPIO Header (2x20)	1	Rs 4550
16GB microSD Card	1	Rs 950
NiMH Rechargeable Batteries	3	Rs 360
Battery Holder (3x AA)	1	Rs 60
MT3608(5V Boost Converter Module)	1	Rs 300
1N5817 Schottky Diode & 5.1V Zener Diode (BZV85-C5V1)	1	Rs 40
Fuse 1.2A & holder	1	Rs 60
M3 brass standoff	20	Rs 350
3.5mm Audio Jack (Male & Female)	1	Rs 200
Solar Panels (5.5V 120mA)	4	Rs 1,600
13nH SMD Inductor	1	Rs 20
18 & 16 pF Ceramic Capacitor	1	Rs 20
SMA Female Connector (PCB mount)	1	Rs 150
433MHz Antenna	2	Rs 2,000
330Ω, 4.7kΩ & 1kΩ Resistors	3	Rs 80
MPU9250 & BMP280 Module	1	Rs 550
LED 5mm	3	Rs 30
INA219 Current Sensor Module	3	Rs 1350
CN3085 NiMH Charger Module	1	Rs 250
JST-XH 8-pin Male & Female Connector	1	Rs 60
JST-XH 2 & 4-pin Male & Female Connector	6 & 4 each	Rs 320
Capacitors 100 µF, 10 µF, 0.1 µF	6	Rs 170
Electromagnetic coil	60-70m	Rs 2000
DRV8833 H-bridge motor driver	1	Rs 200
Pcb & 3d frame fabrication	4	Rs 8,340
Total		**Rs 21,600**

PROJECT IMPACT AND EDUCATIONAL VALUE

Educational impact and hands-on learning

The 1U CubeSat simulator was designed to bridge the significant gap between theoretical knowledge and practical application in satellite technology. It provides a tangible, working model that demystifies complex aerospace engineering concepts, making them accessible to students at a fraction of the cost of traditional educational programs. By engaging directly with a physical system, students can:

- **Understand satellite architecture:** They can visually and functionally grasp how different subsystems like C&DH, EPS, and ADCS are integrated and communicate.
- **Debug and problem-solve:** The process of troubleshooting a working hardware and software system provides invaluable real-world engineering experience.
- **Engage with real-world data:** The ability to receive and analyse live telemetry data from the simulator, such as power consumption and orientation data, gives students a genuine understanding of satellite operations.

Fostering advanced and specialised skills

This project goes beyond a simple educational kit. Its development and design foster a range of highly sought-after technical skills that are directly relevant to modern engineering and technology fields.

- **Embedded systems and hybrid architecture:** The use of a Raspberry Pi Zero W and an Arduino Nano in a hybrid architecture teaches students how to choose and integrate microcontrollers for specific tasks, optimising a system for both high-level processing and real-time control.
- **Custom PCB design:** The custom PCBs are a core deliverable. Designing these boards from scratch teaches critical skills in electronics design automation (EDA) software, schematic capture, and PCB layout, which are foundational for any hardware-centric career.
- **Software-defined radio (SDR):** The innovative use of rpitx to generate the 433 MHz signal is a prime example of SDR. This approach introduces students to a cutting-edge field of electronics and programming, demonstrating how software can be used to control radio frequency hardware.

A scalable and reproducible educational model

A key goal of this project is to create an open-source, reproducible model for STEM education. The comprehensive documentation including the full technical report, Gerber files for the custom PCBs, and commented code ensures that the simulator can be easily replicated. This makes it a powerful, scalable tool that can be deployed across multiple educational institutions without a prohibitive cost barrier.

Inspiring the next generation of innovators in Nepal

Nepal is an emerging player in the global space community, with ongoing projects and growing interest in the field. This simulator can serve as a foundational tool to help nurture this interest locally. By providing a platform for hands-on experience, the project aims to inspire students and future engineers in Nepal to explore careers in technology and space science, contributing to the country's technological and scientific advancement.

CONCLUSION AND NEXT STEPS

Conclusion

The development of the 1U CubeSat simulator successfully met its core objectives. It stands as a testament to the power of independent, resource-conscious engineering and demonstrates that complex aerospace concepts can be made accessible and tangible. By using a custom, stacked-board architecture, a software defined radio for communication, and a hybrid microcontroller system, the project not only created a functional educational tool but also showcased a high level of technical proficiency and innovative problem-solving. The simulator is a powerful proof of concept, and its comprehensive documentation makes it a repeatable and valuable resource for STEM education.

Next steps

With the successful completion of the simulator, the next phase is to deploy it as a tool for learning and inspiration. I propose a formal partnership to integrate this simulator into your educational programs. Specifically, I would like to:

- **Conduct a live demonstration** of the simulator's functionality, including telemetry transmission and ground station data analysis.
- **Develop and lead workshops** to teach students about embedded systems, custom hardware design, and satellite operations using the simulator.
- **Collaborate on future iterations** to further enhance the simulator's capabilities and tailor it to specific educational needs.