GAUCHOSAT: IN-ORBIT SOLAR CELL PERFORMANCE TESTING

Dakota Barnes, Nik Belle, Kpazawala Windross, Brady Gin
Teaching Assistant: Hunter Larson
Faculty: Yogananda Isukapalli
Company Sponsor: Angstrom Designs
University of California, Santa Barbara

1 ABSTRACT

Powering satellites efficiently is essential for the success of long-term space missions. To advance solar cell technology, it is necessary to evaluate performance under real orbital conditions. GauchoSat is a student-built 1U CubeSat developed as a UC Santa Barbara computer engineering capstone project for Angstrom Designs to address this need by collecting and transmitting IV curve data from space. It features a compact payload known as the Aerospace Measurement Unit (AMU), which captures real-time current, voltage, temperature, and sun angle data of solar cells during orbit. This information is downlinked using a custom UHF communication protocol, enabling analysis of solar cell efficiency in actual space environments. GauchoSat demonstrates a low-cost, modular approach to deploying space-based testing platforms, accelerating the development of next-generation satellite power systems.

2 INTRODUCTION

Motivation: Angstrom Designs specializes in testing advanced solar cells for space applications. While their ground-based laboratory testing is highly controlled and precise, it lacks the environmental realism of true orbital conditions. To close this gap, Angstrom enlisted our team to explore ways to evaluate their customers' solar technology performance in space, particularly under the varying temperatures, radiation exposure, and light conditions of low-Earth orbit. Our goal was to develop a system capable of gathering detailed IV (current-voltage) curves from orbit using Aerospace Measurement Units (AMUs). These compact, high-precision instruments allow for accurate monitoring of solar cell performance, enabling data-driven improvements to space power systems.

CubeSat Initiative: CubeSats have become increasingly popular to "Democratize Space", to provide universities, startups, and research teams with affordable and manageable satellite development opportunities. Building a nanosatellite is the most practical and cost-effective way to measure solar cell performance in space. CubeSats, small satellites typically measuring 10 cm per side and weighing 1 kg, provide a highly approachable entry point to

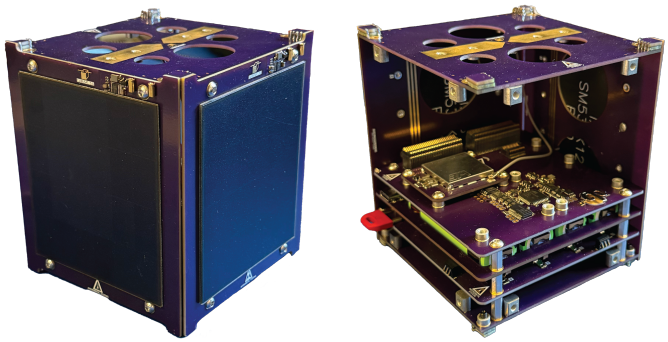

Figure 1: Space-ready GauchoSat pictured with two side panels removed to showcase the controller board.

spaceflight for universities, startups, and research teams. Despite their small size, CubeSats can be configured with a complete set of spacecraft subsystems, including a flight computer, communication system, attitude determination and control system (ADCS), power system, and mission-specific payload. CubeSat missions typically require up to two years of development to accommodate system integration, testing, and launch readiness.

GauchoSat: GauchoSat is our team's 1U CubeSat designed to evaluate solar cell performance in orbit on behalf of Angstrom Designs. Our team was challenged to complete GauchoSat on a condensed nine-month timeline, demanding an accelerated and highly coordinated engineering effort. The satellite includes a flight computer running a QP-nano state machine for mission control, a UHF radio using a custom message protocol for data transmission, and Angstrom's AMU as the payload for IV curve collection. Supporting subsystems include a power system with solar panels and batteries, and a telemetry monitoring system. Our design focuses on modularity, robustness, and low power consumption, with software and hardware optimized for the limited bandwidth and tight energy budget typical of CubeSat missions. GauchoSat's streamlined architecture makes it an effective platform for in-orbit solar testing, while remaining adaptable for future research needs.

3 FLIGHT PLAN

Timeline: The GauchoSat mission follows a tightly scheduled 35-day flight plan, determined by the satellite's low-Earth orbit decay window before reentering and burning up in Earth's atmosphere. The flight is divided into four key phases:

- **Launch:** GauchoSat is deployed into orbit by a commercial launch provider. This marks the beginning of onboard operations, as the satellite begins passive power generation and waits for ground command verification.
- **Initialization (35 hours):** Upon deployment, the flight computer enters an initialization state, fully charging onboard batteries and sequentially testing each subsystem to ensure operational readiness in the space environment.

- **Detumble (4 hours):** A brief detumbling phase attempts to reduce the satellite's rotation after launch using passive or software-based techniques. This is critical to stabilize the orientation and ensure consistent radio communication and accurate payload measurements.
- **Science (32 days):** In the main mission phase, GauchoSat enters a repeating loop of collecting IV curves and telemetry data, storing it onboard, charging, and periodically downlinking packets to the ground station. This cycle continues for the remaining 32 days of the satellite's lifetime.

This streamlined plan is designed to maximize useful data collection within the short orbital window, while accounting for power, memory, and communication limitations.

Constraints: To achieve this flight plan within the confines of a 1U CubeSat, every design decision had to account for tight constraints on power, memory, mass, and volume. With only a small solar panel surface area and limited battery storage, the flight software flow had to be optimized for low power consumption, implementing careful duty cycling and low-power modes wherever possible.

Similarly, firmware had to be written with minimal RAM and flash usage, given the limited memory available on the onboard microcontroller. Achieving robust mission functionality within these constraints required close coordination between hardware choices, software scheduling, and firmware architecture.

4 HARDWARE

GauchoSat: We found that the hardest part of building a CubeSat was determining where to start, because there is no single solution that meets every mission's requirements. We began with a power budget based on the expected demands of our payload, the Aerospace Measurement Unit (AMU). Beyond the current draw, size, and weight of our payload, we had to consider the pointing ability required for the AMUs to gather meaningful data.

This meant we had to account for integrating a robust attitude determination and control system (ADCS), responsible for detumbling the CubeSat—slowing its spin post-deployment to enable proper radio communication—and orienting the test solar cells toward the sun for accurate measurements. While initial research and design considerations for the ADCS were conducted, developing and integrating a full ADCS was outside the scope of our nine-month project timeline. This component will be further developed by Angstrom using the foundational research our team initiated.

From there, we worked backward to identify hardware that could meet our mission's needs while fitting within our size, cost, and timeline constraints. After consulting professors, other universities, and industry experts, it became clear that we had three main options:

- **Build individual components from scratch:** Affordable but risky due to our inability to test if our hardware would survive the physical degradation caused by space.
- **Use commercial off-the-shelf (COTS) space-rated parts:** Reliable, but expensive

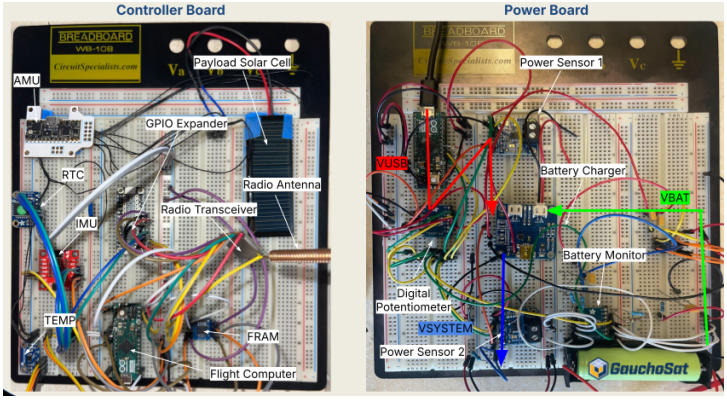

Figure 2: Flat Sat with the Controller Board (left), and the Power Board (right).

and difficult to integrate within our strict weight, power, memory, and storage constraints.

- **Purchase a full CubeSat kit:** Expensive up front, but already optimized for integration of our payload and firmware within the constraints mentioned above.

We ultimately selected the Interorbital Systems 1U CubeSat kit, which provided a chassis, solar panels, power management hardware, radio, telemetry interfaces, and a PCB with an integrated flight computer. This platform was not only cost-effective relative to other space-rated kits and COTS components, but also fully compatible with our payload.

Choosing Interorbital allowed us to eliminate the risk of hardware failure due to our inability to accurately simulate the physical conditions of space. Additionally, it shifted our focus to physically integrating our payload into the CubeSat, developing reliable firmware to meet our mission needs, and creating a testing environment to ensure success within our timeline.

By delivering a tested, launch-ready system, GauchoSat ensures that Angstrom has a reliable vehicle to validate AMU performance in orbit without requiring deep spacecraft hardware expertise.

FlatSat: In parallel with the product-level flight hardware assembly and testing, we built a FlatSat. In other words, a ground-based replica of the GauchoSat, to support both development and future mission replication. FlatSats are widely used in the aerospace community as a way to prototype and debug CubeSat functionality outside of the space environment. Ours was constructed on a breadboard using Adafruit versions of each subsystem component, allowing us to mimic flight software behavior and validate data flow between the microcontroller and all peripherals. The FlatSat not only enabled rapid testing during development, but also provides Angstrom with a replicable blueprint for building and validating future CubeSats from the ground-up. If issues arise during flight, the FlatSat also serves as a valuable debugging tool for reproducing behavior on the ground and testing firmware updates in a controlled environment.

Controller Board: The controller board serves as the central processing hub of the GauchoSat, managing peripherals, executing the state machine, collecting payload data, and

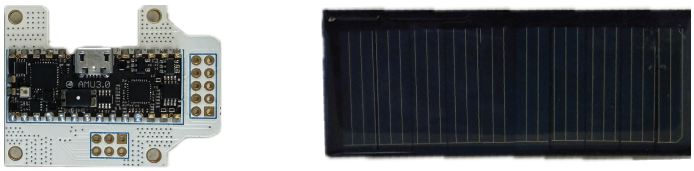

Figure 3: Aerospace Measurement Unit (left) and Solar Cell (right).

handling radio communications. The key components are below:

- **ATMEGA32U4 (Flight Computer):** Chosen for its simplicity in programming and development, this microcontroller enabled rapid initial software deployment. However, it revealed limitations in processing speed, memory, and flash capacity, prompting consideration of a more powerful processor in future iterations.
- **Radio Transceiver:** Integrated on the radio board and connected via SPI, the transceiver was selected after extensive research into alternatives such as the Iridium Satellite Network and various frequency bands (L-Band, S-Band, Ka-Band). It offered ease of integration, affordability, and compatibility with existing Amsat ground station networks for transmitting custom 32-byte packets. Further details are provided in the Ground Station section.
- **Dipole Antenna:** The dipole antenna enables the CubeSat to transmit messages to Earth regardless of orientation. It is constructed by cutting measuring tape into two equal lengths to match the 434 MHz wavelength.
- **Aerospace Measurement Units (AMUs):** Developed by Aerospace Corporation and integrated by Angstrom Designs, these units interface via I2C to evaluate solar panel performance. The AMUs measure current-voltage (IV) characteristics, sun angle, and temperature, providing key metrics such as maximum power output—critical for solar cell efficiency assessment.
- **RTC with Coin Cell Battery:** This real-time clock module ensures accurate timekeeping even when the controller board is powered down, allowing for precise mission timestamping.
- **9 DOF IMU:** This Inertial Measurement Unit integrates an accelerometer, gyroscope, magnetometer, and temperature sensor to determine CubeSat orientation and axis positioning, supporting attitude determination and control.
- **Temperature Sensor:** A highly sensitive sensor used for monitoring the CubeSat's internal temperature in degrees Celsius, essential for thermal regulation and safety.
- **8 MB FRAM:** Used to store peripheral data directly, this FRAM module reduces flash and RAM usage on the flight computer. It retains data even during low-power charging states and is connected via SPI.
- **GPIO Expander:** This component increases the number of available general-purpose input/output (GPIO) pins, enabling connection to multiple peripherals—especially the

radio—while conserving resources on the main flight controller.

Power Board: The Power Board is responsible for power monitoring, distribution, protection, and regulation across the GauchoSat system. It interfaces directly with the battery, solar panels, and system loads, while also relaying critical telemetry to the controller board. Below are the key components and their functions:

- **ATMEGA32U4 (Power MCU):** The ATmega32U4 on the power board acts as the Power Management Microcontroller, coordinating sensors, chargers, and regulators to ensure safe and efficient power operation. It also facilitates communication between the power board and the flight controller.
- **Battery Charger:** This charge management controller handles safe and efficient charging of the Li-Ion battery from solar or USB sources. It integrates full charge cycle control, power path management, and protection features, ensuring reliable battery operation and maximizing system uptime.
- **Solar Voltage Regulator:** A high-efficiency, low-input-voltage boost-buck DC/DC converter. On the power board, it plays a critical role in regulating and stabilizing voltage from the solar panels.
- **Battery Monitor:** This battery management IC measures battery voltage and current through dedicated registers. It also includes protection and fault monitoring capabilities using an integrated protection register.
- **Digital Potentiometer:** A dual-channel 10 kΩ digital potentiometer used to adjust analog reference voltages. It controls input current from solar or USB sources by tuning feedback signals in power management circuits, enabling programmable and efficient current limiting.
- **Power Monitoring Sensors:** Two modules monitor current, voltage, and power across key power lines. Their readings provide real-time telemetry on subsystem consumption, enabling intelligent load management.

Ground Station: The ground station is a critical component designed to communicate with GauchoSat, providing both testing capabilities and ongoing mission support. It plays a crucial role by enabling retrieval of payload data and facilitating the transmission and reception of custom messages to and from the satellite. Key components include:

- **RTL-SDR:** A versatile universal frequency receiver connected via USB and coupled with an extendable dipole antenna. Our implementation utilized the `SDR++` software on macOS, allowing for visualization and debugging of radio signals during communication setup.
- **ATMEGA32U4 (Ground Station Controller):** Used consistently across both the ground station and GauchoSat to ensure compatibility and streamline firmware development. The microcontroller configures the radio to receive incoming messages and forwards them to the ground station user interface. It also manages the transmission of user-generated commands back to the CubeSat.

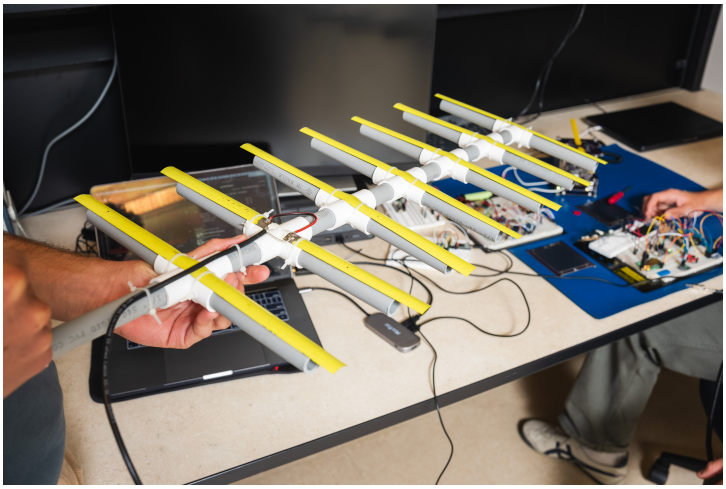

Figure 4: DIY Yagi Antenna for 70cm Band.

- **Radio Transceiver:** Identical to the FlatSat module integrated into the GauchoSat PCB. This transceiver includes a high-pass filter and power amplifier to manage communications effectively. It transmits and receives 32-byte packets, which are encrypted and decrypted between the ground station and CubeSat.
- **DIY Yagi Antenna:** Constructed from tape measure segments, PVC pipes, and coaxial cable, this antenna is specifically tuned for the 434 MHz (70 cm) frequency band, providing approximately 10 dB of gain. While suitable for initial testing, a circularly polarized antenna and automated tracking system will be required for reliable communication during orbital operations due to the satellite's expected tumbling.

5 SOFTWARE AND FIRMWARE

Controller Board Firmware: The GauchoSat firmware is built around a real-time finite state machine using the QP-nano framework, chosen for its lightweight design and built-in state management features. QP-nano abstracts away the low-level handling of state transitions, reducing the risk of deadlocks or unreachable states while enabling our CubeSat to respond autonomously to its environment. This framework provides a clear structure for managing mission-critical operations on a microcontroller with limited memory and processing power. The state machine transitions from the launch state to the orbit state when a mechanical switch on GauchoSat is triggered upon deployment into space. Within the orbit state, the system cycles between two primary modes: charging and active. A timer-based interrupt drives a periodic tick signal in the active state, where the battery level is checked on each tick. If the battery level drops below 15%, the system transitions into charging mode and remains there until the batteries reach full capacity. This ensures that critical operations are always performed with sufficient power reserves. In the active state, there are two substates: payload and radio. In the payload substate, the firmware alternates between detumble and telemetry phases. During detumble, the satellite reduces its rotation rate to stabilize orientation and optimize data collection. Once stabilized, the telemetry substate

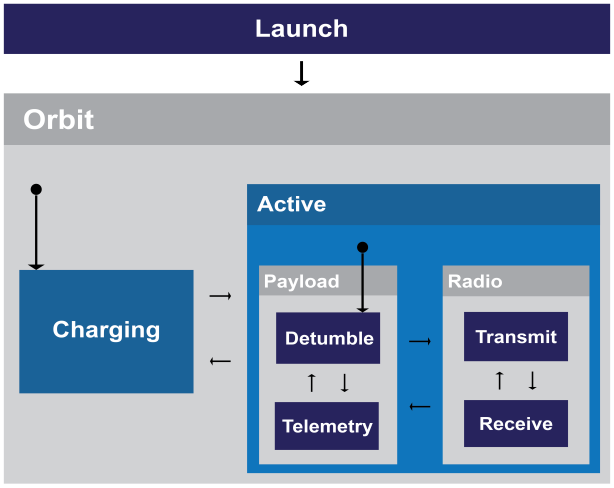

Figure 5: GauchoSat Software Flow.

collects data from onboard sensors, including the IMU, temperature sensor, RTC, and AMUs, and stores it in FRAM. Data from each sensor is written to indexed sections of FRAM, which allows for efficient partitioning and minimizes RAM usage by eliminating the need for large in-memory buffers. FRAM's non-volatile nature and acceptable access speeds made it a strong fit for our low-throughput data needs. After telemetry collection, the system transitions into the radio substate, where it reads stored data from FRAM, constructs packets, and transmits them using the appropriate communication strategy. Following transmission, the satellite enters a brief receive window, typically five minutes, to listen for incoming commands from the ground station. The firmware supports both production and test modes. In production mode, the state machine operates autonomously, and telemetry data is monitored via a Python-based ground interface that receives and decodes downlinked packets. In test mode, GauchoSat enters a passive listening state, allowing regression tests to be performed by sending messages and observing responses. This dual-mode system enables both operational reliability and pre-flight debugging with minimal code divergence.

Controller Board Firmware: To facilitate robust and flexible communication, we developed a custom messaging framework that works uniformly across multiple communication interfaces: radio, serial, and SPI. While the physical transport layer differs across these strategies, the packet structure and message handling logic remain consistent. This modularity simplifies development and ensures that core communication features can be tested in different environments using the same firmware. For debugging and local development, we use serial communication, allowing us to plug the CubeSat into a computer, send commands, and monitor behavior in real time. In flight, we use the radio interface for wireless communication with the ground station. Internally, we leverage SPI to send messages from the controller board to the power board, such as fetching power system telemetry when requested. The messaging system is built to support asynchronous reception and handling of messages. When the CubeSat is in receive or test mode, incoming messages are parsed

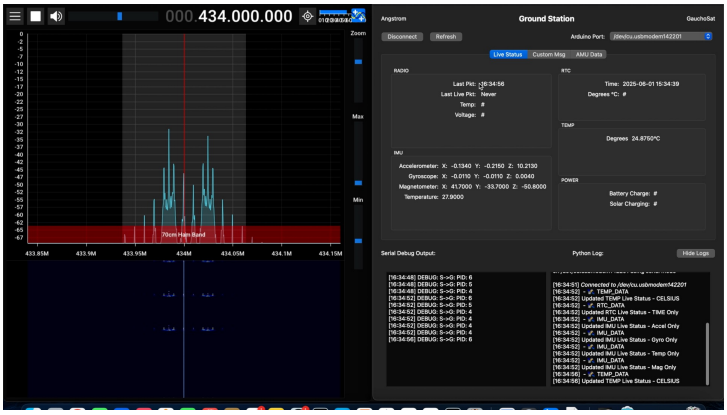

Figure 6: Custom Ground Station UI.

and routed to the appropriate handler class, which executes the corresponding command. Conversely, in production mode, the framework allows the firmware to package and transmit telemetry data or status updates back to the ground station or local computer. Each message packet follows a lightweight format:

- A header containing predefined start bytes, a message ID, and message length
- A data payload section
- A CRC16 checksum to ensure data integrity

Packets are constructed into byte streams before transmission. Upon receipt, the byte stream is parsed, verified with the checksum, and routed based on the message ID. This consistent structure allows messages to be easily extended, debugged, and validated across all communication channels, and provides a strong foundation for both real-time telemetry and interactive control during flight.

Power Board Firmware: The Power Board firmware manages power telemetry and communication between the power subsystem and the main flight computer. It initializes and interfaces with onboard sensors, two power monitors, a digital potentiometer, and a battery monitor over I2C. It listens for SPI packets from the main controller, interprets sensor data requests, and responds with real-time power readings (current, voltage, and power). The firmware ensures autonomous power subsystem operation and supports modular telemetry collection for both regulated power lines. The power board firmware can only be loaded on to the GauchoSat through an ICSP programmer.

Ground Station Firmware: The firmware running on the ground station utilizes the messaging protocol to communicate with the radio module over serial. Under normal operation, the firmware places the radio in receive mode, continuously monitoring for incoming messages from GauchoSat. When a command is initiated via the ground station user interface (UI), the firmware temporarily switches the radio into transmit mode, sending the requested command before reverting to receive mode. This dynamic switching ensures efficient and responsive communication with the satellite.

Ground Station User Interface: The custom Python-based ground station provides an intuitive interface for real-time interaction with the GauchoSat. Upon startup, users can select the connected microcontroller from available devices. The main interface displays the satellite's live status, reflecting telemetry data transmitted from GauchoSat. A dedicated command interface allows users to send custom commands, enabling targeted data requests or operational controls. Additionally, a specialized view presents Aerospace Measurement Unit (AMU) data collected from the satellite payload. All sent and received commands are systematically logged to a CSV file, ensuring comprehensive record-keeping for post-flight analysis and troubleshooting. Additionally, the SDR++ software which is connected to the RTL-SDR dongle is used to visualize the frequency transmissions over 434 MHz.

6 TESTING AND SIMULATION

Python Testing Environment: To support robust development and validation of GauchoSat's firmware, we built a flexible Python-based testing environment that enables real-time monitoring, command execution, and automated regression testing. This environment plays a critical role in both interactive debugging during development and formal testing prior to deployment. At the core of the testing environment are a set of Python dataclasses each representing a major onboard subsystem, including the AMUs, telemetry sensors, radio, and power system. These classes encapsulate methods for parsing incoming messages and extracting meaningful information from the raw data packets. When the CubeSat transmits data, the Python environment listens over serial communication, interprets the message based on its ID, and routes it to the corresponding dataclass for structured analysis. This allows for clean data visualization, logging, and validation during both test mode and live production monitoring. The Python interface also provides utilities for constructing and sending custom command packets to the GauchoSat using any supported communication strategy (serial, radio, or SPI). These commands are used to trigger specific actions onboard, such as requesting telemetry, initiating subsystem tests, or resetting the state machine. Critically, the environment supports automated regression testing, where predefined test scripts construct packets, send them to the flight computer, and then listen for and validate the responses. Assertions are made on the returned data to confirm that functionality remains intact after changes to the firmware. This helped ensure that newly added features did not compromise previously working subsystems, a key requirement given the tightly integrated nature of the GauchoSat software. The testing environment also integrates cleanly with graphical UIs, making it a powerful tool not only for developers but also for mission operators to monitor satellite health and command behavior in a more user-friendly format.

Simulations: Simulation played a critical role in developing and validating the GauchoSat's state machine and overall software functionality. A MATLAB CubeSat simulation tool provided solar power generation data based on sunlight exposure, amplitude, and angle of incidence throughout the satellite's orbit. These power output values were fed into our system to test whether the state machine responded correctly to real-world orbital conditions. By simulating variations in available power, we verified that the state machine correctly transitioned between charging and active states based on power levels, ensuring reliable operation and preventing mission failure from power budget breaches.

7 CHALLENGES

The GauchoSat project presented a set of unique and rigorous challenges, primarily driven by a drastically condensed development timeline. Typical CubeSat projects span approximately two years; however, our team was tasked with delivering the satellite within just nine months. This accelerated schedule required critical trade-offs and rapid decision-making. Navigating the aerospace industry's complexities, we quickly adapted to significant constraints, including costly components, extended lead times, and limited availability of open-source resources. Due to the high stakes involved, extensive preliminary research was essential to ensure accurate first-time hardware selections, as opportunities for iterative testing were severely limited. Additionally, we faced the substantial challenge of reverse-engineering and rebuilding a complex satellite system, often with limited documentation and components that arrived late into our project timeline. Each team member encountered substantial hurdles while integrating subsystems such as radios, FRAM, and power management systems. Compounding these challenges was an exceptionally condensed window for validation and integration testing, far below aerospace industry standards. Lastly, embedded software development was heavily constrained by limited memory resources of only 2.5 KB RAM and 32 KB Flash. This necessitated highly optimized software solutions, custom lightweight communication protocols, and efficient data storage in external FRAM modules.

8 FUTURE WORK

Future Work will focus on several key areas to enhance the satellite's capabilities and readiness for deployment. This includes purchasing or developing industry-grade ground station tracking hardware or collaborating with amateur radio groups to ensure robust communication capabilities. Additionally, creating a payload PCB with an updated version of the Aerospace Measurement Units (AMUs) and integrating these with test solar cells is critical. We plan to either build or purchase an attitude determination and control system to ensure proper orientation in orbit. Furthermore, rigorous testing procedures will be established to verify that our software, firmware, and hardware systems meet the stringent demands of space conditions. Finally, scheduling a CubeSat launch is essential to bring GauchoSat into operational status and fulfill its scientific mission objectives.

9 CONCLUSION

GauchoSat represents a fully integrated, end-to-end CubeSat platform developed under an accelerated timeline to address the challenge of evaluating solar cell performance in orbit. Through careful system design, a modular software architecture, and custom hardware integration, our team built a functional 1U satellite capable of autonomous operation, data collection, and communication with Earth. Despite the demanding constraints of power, memory, and development time, we successfully demonstrated that low-cost CubeSats can serve as powerful tools for space-based technology validation. GauchoSat not only provides Angstrom Designs with a reliable in-orbit testbed for their Aerospace Measurement Units, but also establishes a framework for future missions and educational satellite development.