Design of an ESP32S3 based Satellite-Ground Communication System

1st
Adrián Israel Trejo Hern...
A01798838 - B.S. in Electronics Engineering

2nd
Diego Ángel Pérez Terán
A01733854 - B.S. in Electronics Engineering

3rd
José Roberto Grajales Ra...
A01733895 - B.S. in Electronics Engineering

4rd
Luis Martínez Páez
A01285395 - B.S. in Electronics Engineering

***Abstract*— This project simulates a satellite-ground station communication link using ESP32-S3-WROOM-1U boards. The initial phase establishes a basic chat system for stable Wi-Fi message transmission via UART conversion. Future developments will integrate telemetry sensors and enhance reliability through improved protocols and error correction, addressing real-world challenges like latency and signal interference. The goal is to create a robust, efficient communication framework for satellite operations.**

***Index Terms*— Satellite Communication, ESP32-S3, Wireless Data Transmission, Telemetry, Wi-Fi Protocols, K-means, Image Compression, SPI communication**

I. INTRODUCTION

Establishing a stable communication link between a satellite and a ground station is crucial in space systems. In this project, an attempt is made to simulate such a link using two ESP32-S3-WROOM-1U development modules that were chosen for their on-board Wi-Fi capability, low power consumption, and dual-core Xtensa LX7 processors with clock frequencies of up to 240 MHz, which are sufficient for real-time communication. The ESP32-S3 also has a large software support for rapid prototyping and development [1].

The project was divided into multiple development stages, following an incremental and iterative approach. The core functionalities of the system which are better described in Section III can be summarized as follows: First, the foundation of the communication system is a stable and continuous Wi-Fi link, established using an Access Point–Client architecture between the two ESP32S3 modules. This connection enables the real-time exchange of textual messages for the purpose of transmitting telemetry data and remote commands.

Sensor integration is another key aspect of this project, it aims to replicate an environmental and operational monitoring in the satellite. It includes sensors to measure ambient humidity and temperature, inertial data such as acceleration and position along the X, Y, and Z axes. Finally, electrical parameters such as current, voltage, and power consumption are monitored to simulate power management subsystems. To avoid synchronization issues, all sensors communicate with the microcontroller via the Serial Peripheral Interface (SPI), offering reliable communication.

To simulate mechanical responsiveness and interaction capabilities, 2 actuators were implemented. These are used to emulate actions such as reorientation or signaling through visual indicators, allowing the system to mimic basic physical responses to control inputs or environmental changes.

To enable friendly user interaction and better telemetry visualization, a graphical user interface (GUI) was developed using MATLAB's App Designer. The GUI serves as a command center which provides a live display of sensor data and the system status, it also allows the control for sending operational commands.

Visual data transmission was also included in the system through a camera in the PC connected to the ESP32S3, images are compressed using a custom algorithm based on the K-means clustering method, a technique developed and optimized during coursework. This method enables efficient image compression while retaining key visual features.

Finally, to emulate the satellite's autonomous power system, a solar panel was installed to capture light energy, which is then stored in a lithium-polymer (LiPo) battery. A boost converter is used to regulate and stabilize the output voltage, ensuring consistent power delivery to the system. This setup allows to simulate the satellite's capability to operate independently using solar energy.

II. THEORETICAL FRAMEWORK

A. Access Point (AP) – Client architecture

The term AP is used to refer to a networking device which can allow multiple devices to connect to a wireless network. It acts as a central hub that broadcasts Wi-Fi signals and manages connected devices. A perfect example of an Access Point is the router [2].

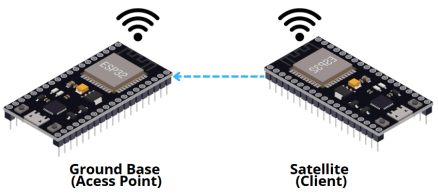

Figure 1. Ground Base-Satellite Wireless Connection Diagram

B. Transmission Control Protocol and Internet Protocol

The TCP/IP protocol is used in this project to manage the communication between the ground station and the satellite module. It defines how data is organized, sent, and received over a network, ensuring reliable transmission between both ends.

TCP (Transmission Control Protocol) is a connection-based protocol. It creates and maintains a stable link between the two devices during communication. One of its main functions is to check that the data arrives correctly and in the right order. If any part of the data is missing or damaged, TCP can ask for it to be sent again.

IP (Internet Protocol) works together with TCP by assigning an address to each data packet. This helps the network know where each packet should go. As the packets travel through the network, devices along the way use the IP address to forward them to the correct destination.

By using TCP/IP, the system is able to transmit information in a way that is both reliable and organized, which is essential for maintaining a good communication network [3].

C. RSSI and CSI

One important parameter in wireless communication systems is the Received Signal Strength Indicator (RSSI). As the name suggests, RSSI measures the strength of the signal received at the access point (AP). It is typically expressed in dBm, where values closer to 0 dBm represent stronger signals. Monitoring RSSI helps evaluate the quality of the wireless link between devices [4].

Another key concept is Channel State Information (CSI). CSI plays a central role in modern wireless sensing technologies, such as those based on Wi-Fi or LTE. It provides detailed information about the state of the communication channel by measuring how the signal behaves across different subcarriers. These measurements reflect how the wireless signal propagates through the environment. Since CSI contains geometric details of the signal path, it is often used for tasks like location tracking, movement detection, and environmental mapping [5].

D. I2C protocol

The Inter-Integrated Circuit (I2C) protocol is a popular communication method that enables microcontrollers to exchange data with various devices, such as the sensors integrated into this project. I2C uses just two wires: one for the clock signal (SCL) and another for the data (SDA), this greatly simplifies wiring.

I2C is a synchronous protocol, meaning that both the master (in this case, the satellite) and the slave devices (all the sensors) are synchronized using a shared clock line. Each device on the bus is assigned a unique address, allowing the master to communicate with multiple peripherals using the same two lines. This protocol is very useful in embedded systems with many sensors, as it supports multiple devices and helps avoid timing issues caused by small differences in clock speeds [6].

E. Types of sensors used and their specifications

Each sensor was selected based on its low power consumption, measurement range, and communication protocol compatibility with the ESP32-S3. The sensors communicate via the I2C protocol, allowing efficient data exchange and less wiring. More specific details are presented in Table 1 [7],[8],[9].

Table 1. Sensors and their operating characteristics

Sensor	Measured parameter	Range of error	Com Protocol	Average Power Consumption
INA219	Current, Voltage, Power	±1%	I2C	1 mA
MPU 6050	Acceleration, Orientation	±3%	I2C	Gyro - 3.6 mA Accel - 0.5 mA
AHT20	Temperature , Humidity	±0.3 °C ±2% rel humidity	I2C	0.023 mA

F. K-means clustering

K-means clustering is a famous algorithm that organizes data into groups (clusters) based on similarity. The basic idea behind everything is to group data points that are close to each other in value. The first step is to select a number of clusters (K) and generate a series of central points, called centroids. Each data point is then assigned to the nearest centroid, forming initial clusters. Once this is done, the centroids are recalculated as the average of all points in their respective clusters. This process repeats until the centroids stabilize, meaning the clusters no longer change significantly [10].

In the context of this project, k-means clustering will be used for image compression. By grouping similar colors in an image and reducing them to a limited set of representative values (the centroids), we can significantly reduce the amount of data needed to store or transmit the image—while preserving its essential visual features.

G. LiPo batteries

Lithium-Polymer (LiPo) batteries are widely used in portable electronic projects due to their high energy density, lightweight construction, and rechargeability. They offer a good balance between capacity and size, making them suitable for space-constrained applications like small satellites or embedded systems.

H. Boost converter

It is a DC-DC power converter that increases the input voltage to a higher output level. This is particularly useful when working with the solar panel and the LiPo battery, it will allow it to go from approximately 3 volts to 5 volts

needed to drive the ESP32 and all the sensors in the system, while keeping a stable operation in terms of the output fixed voltage.

III. DEVELOPMENT STAGES

1) Configuring the ESP32s as communication nodes

The ESP32 supports 802.11 b/g/n Wi-Fi standards, operating in the 2.4 GHz frequency band, allowing it to connect to existing networks, establish its own, or communicate directly with other devices using Wi-Fi Direct. As mentioned before, the ESP32 can function simultaneously as both a client and an access point (AP+STA mode). The ground station will be configured as the access point and the satellite as the client, enabling the station to receive telemetry from multiple satellites efficiently.

Ground Station (GND) is configured as an AP by using the WiFi.softAP function, which establishes a local network with a specified SSID and password, allowing other devices to connect directly to it without the presence of an external router [2]. This is a useful configuration because it assigns an IP address to connected devices. Meanwhile, the satellite (SAT) is in Station mode, enabling it to connect as a client to the ground station's AP using the WiFi.begin(ssid, password) function with the appropriate network credentials.

2) Setting Up the Point-to-Point Architecture

The communication between SAT and GND uses TCP/IP because it makes sure all the data arrives in the right order and without errors. Messages are sent as plain text, and each one starts with a tag that shows who sent it and what kind of message it is (for example: SAT:MSG:<payload> or GND:ACK:<message>). There are also special command messages that let the user start or stop the sending of telemetry data like signal strength (RSSI), using simple commands like /cmd start or /cmd stop. To avoid confusion or missed messages, both the SAT and GND send back confirmation messages (ACKs) when they receive something, and they also let the user know when command mode is active.

3) Implementation of SAT-GND code basis and sensors initialization

ESP32's UART interface, initialized at 115200 baud via Serial.begin() within the setup() function. This baud rate offers a fast and reliable link between the ESP32 and the serial monitor, enabling low-latency debugging and communication. UART0, the default serial port mapped to GPIO1 (TX) and GPIO3 (RX), is used for interactions with a host PC or terminal. Messages are sent as plaintext with structured prefixes to identify the sender, receiver, and type of message, aiding in both debugging and command parsing.

For better organization and modularity, both the satellite and ground station segments work with functions such as displayHeader() to indicate the start of a session, alongside two key functions: transMessages() and receiveMessages(). These manage bidirectional message flow—either by relaying serial input to the client or receiving incoming data, including acknowledgment flags for confirmed delivery. Additionally, a basic Command Mode is implemented within receiveMessages() to handle commands like RSSI checks or sensor queries.

On the satellite side, connectivity is managed through the functions connectToWiFi(), which uses the ESP32 Wifi function WiFi.begin(ssid, password). WiFi.status parameter is constantly monitored to ensure a persistent connection to both the network and the ground station.

Sensor integration is handled using GPIO8 (SDA) and GPIO9 (SCL) as the communication lines. Each sensor is initialized with its corresponding .begin() function and validated via boolean flags (AHT20_ON, MPU6050_ON, and INA219_ON). Sensor readings are acquired using standard library functions such as aht20.getEvent(), mpu6050.getEvent(), and INA219's suite of methods like getBusVoltage_V(), getCurrent_mA(), and getPower_mW().

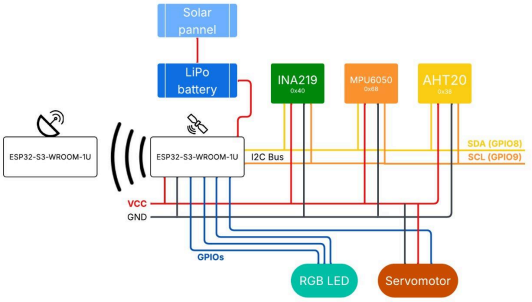

Figure 2. Sensors Diagram Connection to SAT

Additionally, the system monitors Wi-Fi signal strength using WiFi.RSSI() and provides command-based interaction via receiveMessages(). This function processes input such as /tel temperature, /tel current, or /cmd save, responding with live data or performing actions like saving state to persistent memory. The command structure facilitates remote telemetry access and control from the ground station.

```
GND:LOC: /tel temperature
SAT:ACK: /tel temperature
SAT:MSG: Temperature at SAT is 22.99 °C
GND:LOC: /tel humidity
SAT:ACK: /tel humidity
SAT:MSG: Humidity at SAT is 34.79 %
GND:LOC: /tel acceleration
SAT:ACK: /tel acceleration
SAT:MSG: Acceleration of SAT is [X: 0.32 | Y: 0.46 | Z: 10.82] m/s²
GND:LOC: /tel gyro
SAT:ACK: /tel gyro
SAT:MSG: Gyroscope of SAT [X: -0.02 | Y: -0.00 | Z: -0.02] °/s
```

Figure 3. Example of messages sent/received from GND station

4) Experimental Analysis of RSSI and CSI Measurements

Understanding the wireless communication channel between the simulated satellite and ground station is crucial to ensuring a reliable and efficient telemetry link. This section focuses on characterizing the channel by analyzing two key metrics: Received Signal Strength Indicator (RSSI) and Channel State Information (CSI). These measurements are good indicators of signal quality and help predict link performance under different conditions.

- Experimental Analysis of RSSI

An empty room with a space of 6 by 6 meters, was specifically prepared to evaluate wireless channel characteristics. To ensure consistency in measurement height and avoid obstructions, chairs were placed at regular intervals of 1.5 meters apart, serving as supports for both the satellite and ground station. A coordinate system was defined to map the space, enabling structured measurements at the multiple fixed points mentioned before. Then a spatial representation of the channel conditions was created through interpolation and a heatmap.

Table 2. Position in the room vs dBm received

X/Y	0m	1.5m	3m	4.5m	6m
0m	-41 dBm	-46 dBm	-44.67 dBm	-54.33 dBm	-49.67 dBm
2m	-40.67 dBm	-45.33 dBm	-41.33 dBm	-56.33 dBm	-52.33 dBm
4m	-49 dBm	-50.33 dBm	-44.67 dBm	-45.67 dBm	-51.33 dBm
6m	-43.67 dBm	-46 dBm	-45 dBm	-45.33 dBm	-57 dBm

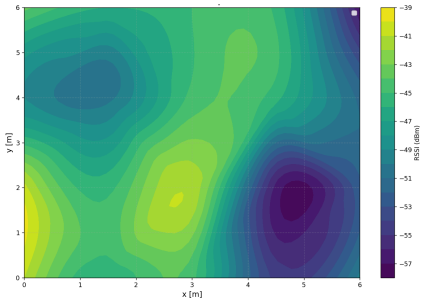

Figure 4. Heatmap of dBm interpolated

The figure 4 displays a clear visualization of how signal strength varies across the measurement area. The first conclusion that can be drawn from the figure is that signal fluctuations are common within a close range. However, a general trend is observed: the highest dBm values are concentrated near the origin point, and signal strength tends to decrease as the distance from this point increases. This supports the expected behavior of signal attenuation with distance. To explore this idea, a plot of signal strength received vs euclidean distance to the receiver is shown in figure 5.

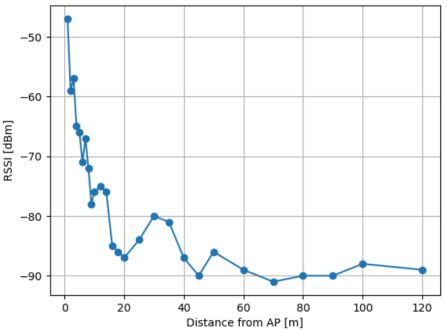

Figure 5. RSSI received vs Distance from AP

From figure 5, it can be seen that it is between 100 meters and 120 meters when the signal strength is oscillating around -90 dBm, for a ESP32 the minimum value of RSSI to maintain a good connection is considered to be above -89 dBm [11], thus it is concluded that the effective communication range of this system is around 100 and 120 meters. Also path loss was calculated and is presented in figure 6.

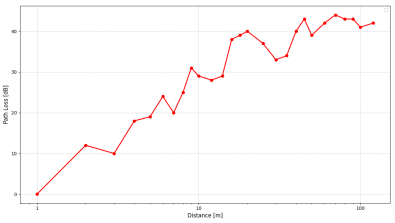

Figure 6. Path loss along the distance

To get more information from the measured RSSI data, the Close-In Free Space Reference Distance model was applied to estimate the path loss exponent (n), which characterizes how the signal attenuates over distance. The CI model is expressed as:

$$PL_{CI}(f,d)[dB] = FSPL(f,d(1m))[dB] + 10n \cdot log(d) + x\sigma_{SF} \quad (1)$$

$$FSPL(f,d(1m))[dB] = 20log\left(\frac{4\pi f}{c}\right) \quad (2)$$

Where f is the carrier frequency (2.4 GHz in this case), c is the speed of light, d is the transmitter-receiver separation distance, n is the path loss exponent, and X□₀ represents the shadow fading modeled as a Gaussian random variable with standard deviation σSF

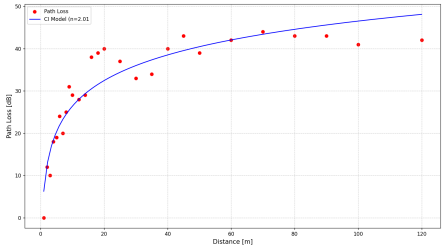

Figure 7. CI - FSPL model and data

Free-space path loss at a reference distance of 1 meter was calculated using a frequency of 2.4 GHz. To fit the CI model to the data, the curve_fit function from scipy.optimize was used; this function performs "non-linear least squares optimization" to estimate the best-fit parameters. This helped to determine the path loss exponent and the shadow fading offset. Calculations of the standard deviation of the shadow fading were made by comparing the measured and modeled path loss values.

The resulting path loss exponent was estimated at n = 2.01, closely aligning with the theoretical value for free-space propagation (n = 2). This confirms that the indoor environment where the measurements were taken—an empty 6m x 6m room with minimal obstructions—introduced negligible additional attenuation. The standard deviation of the shadow fading was calculated at 3.94 dB, which is within the expected range for open or lightly obstructed indoor environments.

Overall, the CI model provided a good fit to the measured data and confirmed the predictable nature of signal decay.

- Experimental Analysis of CSI Measurements

To complement the RSSI characterization of the system, an experiment was conducted to analyze the Channel State Information (CSI). The experiment employed a compiled program distributed during the course, configured to capture CSI samples at a rate of one measurement every 200 milliseconds. To observe the impact of physical obstructions, a person was positioned between the ESP32 and the AP to temporarily block the LOS. The individual remained in place for several seconds before moving out of the path, restoring LOS. This process was repeated for five iterations to generate a sequence of samples and analyze them.

A plot of the Noise Floor versus Timestamp was generated using microseconds as the time unit. Average Noise Floor was also calculated across all collected samples. Values were first converted to a linear scale (mW), averaged arithmetically, and then converted back to dBm.

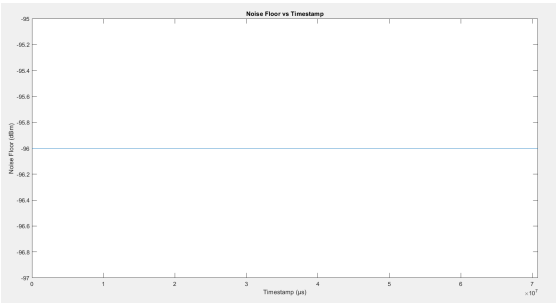

Figure 8. Noise Floor over time

The resulting plot revealed that the Noise Floor remained constant at -96 dBm throughout the entire measurement duration. This constant value indicates a potential hardware limitation inherent to the ESP32S3. It is believed that it suggests that the Wi-Fi chipset reports a fixed noise floor value, possibly due to low measurement resolution.

To further understand the channel dynamics, amplitude and phase of all subcarriers were analyzed as functions of time. Using the same CSI dataset, two separate plots were generated: one displays the amplitude variation of each subcarrier over the timestamp, and the other one displays the corresponding phase variation.

In both plots, the approximate intervals of human obstruction when a person (the team member: Roberto Grajales) was positioned between the Client and the Access Point were clearly marked. These intervals were visually marked with vertical lines, and are provided in figure 9.

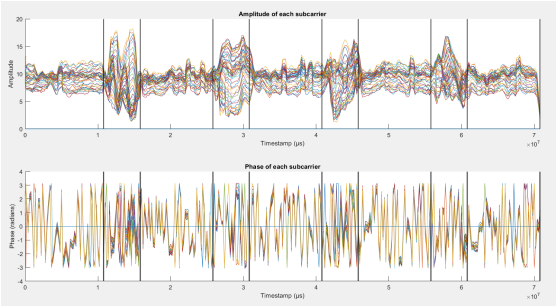

Figure 9. Amplitude and phase of subcarriers on CSI, with periods of body obstruction.

```
Amplitude Standard Deviation during Obstruction Periods:
    1.9528    1.2725    1.5220    1.4762

Amplitude Standard Deviation during No Obstruction Periods:
    0.4850    0.5799    0.6984    0.5380    0.9840

Phase Standard Deviation during Obstruction Periods:
    1.4747    1.4643    1.5219    1.4945

Phase Standard Deviation during No Obstruction Periods:
    1.6358    1.3283    1.4894    1.5506    1.3741
```

Figure 10. Metrics of amplitude and phase subcarriers on CSI

Standard deviation of both amplitude and phase across all subcarriers was computed separately for the obstruction and no obstruction periods, results of this analysis are presented in Figure 10. From the same figure, clear changes in the wireless channel when a person is near the transceiver are observed. This is particularly evident in the amplitude standard deviation, which significantly increases during obstruction (1.27 to 1.95) periods compared to no obstruction (0.48 to 0.98). There are dynamic multipath effects and scattering caused by the human body blocking or reflecting the signal path.

About phase, although some values are slightly higher during obstruction periods (e.g., 1.52 vs. 1.49), in general, the phase standard deviation remains within a similar range across both conditions (approximately 1.32 to 1.63 during no obstruction and 1.46 to 1.52 during obstruction). Amplitude seems to be a more sensitive metric than phase for detecting human presence in this setup.

Based on the previously calculated average noise floor and the known transmitter power, the received power over distance was modeled using the Close-InFree Space Path Loss model (mentioned in "Experimental Analysis of RSSI"). By applying the estimated path loss exponent

n=2.01, the model allowed for the projection of signal attenuation as a function of distance. The analysis assumes that the communication link is no longer viable once the received power falls below the noise floor threshold. This approach provides a practical estimation of the maximum operational range of the wireless link.

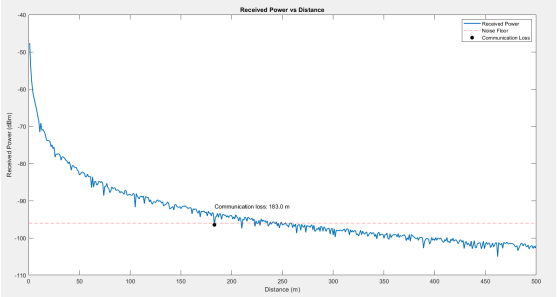

Figure 11. Received power vs. distance graph based on the CI model

According to the model, the communication link loss occurs at a distance of 183 meters. However, there are several factors that influence this result. First, and most importantly, the random variable X **σ** varies greatly with each simulation, which influences where the received power reaches the Noise Floor, thanks to the noise that it adds.. The other important factor that affects the result is the calculated path loss exponent, which can have some error margin and alter the graph.

To validate the estimated cutoff distance derived from the CI path loss model, additional Channel State Information measurements were conducted using the same application. The receiver was positioned at distances of approximately 120 meters and 140 meters from the transmitter. At 140 meters, no successful communication could be established between the ground station and the satellite. Even at 120 meters, the connection was quickly lost after receiving only one or two packets. Attempts to initiate the connection at a closer range and then relocate the receiver to the test distance also failed to maintain communication, confirming a total packet loss beyond the modeled cutoff range.

Given these observations, a packet loss rate of 100% was assumed for distances beyond the cutoff. For the valid measurements, a minimum of 100 sequential packets were recorded. To evaluate packet reliability, the CSV log was first reviewed using the sequence numbers, but no missing packets were identified by this method. Therefore, the analysis shifted to evaluating inter-packet timing using the timestamp data. Since packets are expected every 200,000 microseconds, a tolerance of ±50,000 microseconds was applied to detect delays or retransmissions. Based on this approach, 156 packets were found to exceed the expected interval, corresponding to a delay rate of 38.24%. This result highlights the degradation in link performance as the system approaches its operational limit.

Table 3. Packet Loss Analysis Near and Beyond the Estimated Cutoff Distance

	Few meters short of the cutoff distance (120 m)	Cutoff Distance	Past the Cutoff Distance
Packets lost	156	–	–
Total Packets	408	–	--
Packet loss Percentage	38.24%	100%	100%

5) Design of the GUI

GUI was developed using MATLAB App Designer, its purpose is to provide a user-friendly, interactive environment where various sensor data can be requested and displayed. The user can control system features such as RGB LEDs, servos, and image capture through button presses. Communication with the GND is simplified by only communicating with it through serial communication.

The code handles various events by executing specific functions in response. These functions send the appropriate command, wait for a response, and then update the display accordingly, they also have a try/catch for error handling. Serial ports are automatically detected, making it easy for the user to select the right one.

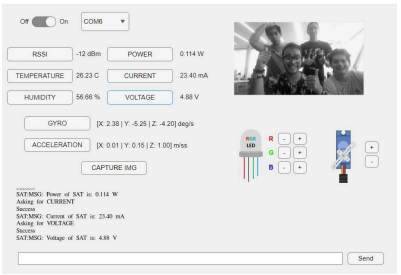

Figure 12. Final GUI

6) Image compression using K-Means clustering

To explore image compression techniques suitable for wireless transmission under constrained bandwidth, we implemented a method based on K-Means clustering for color quantization. The method consists of reducing the number of unique colors in an image by grouping similar pixels into clusters, replacing them with their respective centroid values. Two color models were evaluated: RGB and YCbCr, each with different values of K applied to their respective channels, this analysis was first performed with the famous image of Lenna, and results are shown in, figure 131.

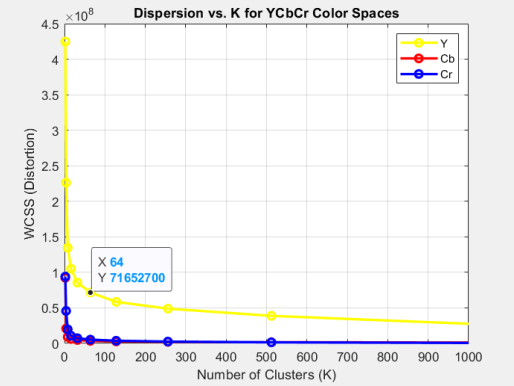

Figure 13. Total distortion vs number of clusters (YCbCr)

Using the elbow method to balance compression efficiency and image quality, we determined the optimal K values for RGB and YCbCr, but in the present work the Y channel will be the principal focus, as it represents the brightness information of the image, which is the most critical component for visual perception. Cb and Cr channels carry color details, to which the human eye is significantly less sensitive, therefore the compression was made only on the Y channel to simplify processing and data handling. By the elbow method, figure x+1 shows that the best value for K is 64.

It is important to clarify that the version of the image being sent is not the compressed data itself, but rather the reconstructed image obtained after compression. The compression process using the K-means method was implemented for learning purposes, to explore how image data can be reduced using clustering.

However, in order to transmit the truly compressed form, it would be necessary to send both the centroids and the cluster IDs, and then perform the reconstruction at the ground station. Since this step was not implemented in this project, the current approach sends the already reconstructed image directly. This simplifies the transmission process but does not preserve the benefits of reduced data size that a full compression transmission reconstruction process would offer.

IV. HARDWARE DESIGN AND ORGANIZATION

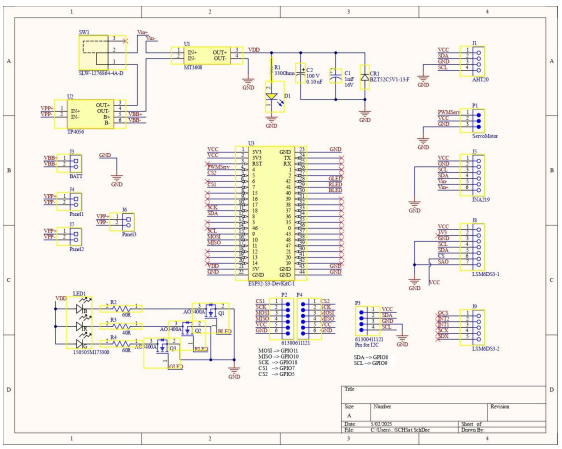

Figure 14. System Schematic Diagram

A PCB was designed to be both efficient and expandable. MOSFETs were included to drive the RGB LEDs, allowing higher brightness without putting too much current load on the ESP32. Also, each sensor has its own dedicated port, which simplifies wiring and improves modularity.

To support future upgrades, the design includes extra I²C and SPI buses. Lastly, thanks to improved power regulation and distribution, the PCB is prepared to handle more solar panels. This opens the door for a more robust power system, which is essential for longer or more demanding missions.

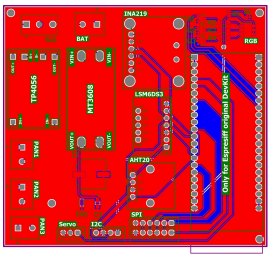

Fig 15. PCB Layout of the Satellite

Unfortunately, due to delivery issues, the PCB was not received in time for physical testing or integration. Despite this, the full design is complete and ready to be implemented. However, the design process provided valuable insight into satellite system integration, power distribution, and PCB layout optimization. The final implemented SAT and GND are presented in future 16 and figure 17.

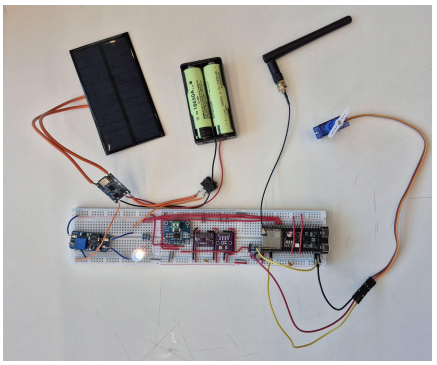

Fig 16. Implemented Satellite in protoboard

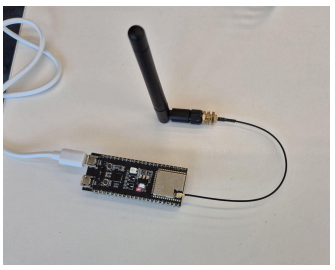

Fig 17. GND Station

V. RESULTS

The system successfully transmitted an image in approximately 35 seconds, divided into 390 chunks, confirming reliable data handling across the wireless link. Power consumption on the ESP32 during operation averaged around 820 mW, full operation of the system even with servo running can be achieved only with the LiPo batteries, the boost converter and the solar panels. At a distance of 5 meters, the average signal strength was measured at -40 dBm, indicating a stable connection under short-range conditions. These results demonstrate the feasibility of image transmission using the ESP32, although further optimization in data throughput and energy efficiency could enhance its performance in more demanding or mobile scenarios.

VI. FUTURE DEVELOPMENTS AND CONCLUSION

Future developments can focus on optimizing the code for the GND, SAT, and GUI by incorporating timers and potentially migrating the project to the ESP-IDF framework to access CSI data. Additional improvements include packaging the satellite using acrylic to enclose the hardware components, designing a dedicated antenna holder, implementing the custom PCB, and integrating an optical sensor to capture images directly, removing dependency on a PC. Image processing could also be handled by the ESP's instead of MATLAB. Furthermore, with more time, the image compression method can be fully implemented to send clusters with their IDs and reconstruct the image efficiently.

The complexity of the design process of a Communication System was proved through the making of the project, at the beginning esp-idf environment was contemplated in order to set a higher goal in terms of learning, but the simple process of deploying an Access Point and trying to receive strings of information even with examples from esp-idf was challenging enough to reconsider the decision and develop the project in the arduino environment, RSSI and Channel State Information analysis was another key in the understanding of how wifi works and how noisy can the data look. At the end the objectives of this project were successfully achieved with solar energy integration, actuators, sensorics and even the design of a PCB to add fanciness.

References

1. OpenELAB Technology Ltd. (2024, October 12). Understanding ESP32-C3 and ESP32-S3: A Comprehensive Guide to Espressif's IoT Powerhouses. Retrieved from: https://openelab.io/blogs/learn/understanding-esp32-c3-and-esp32-S3

2. Santos, S., & Santos, S. (2019, April 23). ESP32 Access Point (AP) for web Server | Random nerd tutorials. Random Nerd Tutorials. https://randomnerdtutorials.com/esp32-access-point-ap-web-server/

3. Yasar, K., Shacklett, M. E., & Novotny, A. (2024, September 26). What is TCP/IP? Search Networking. https://www.techtarget.com/searchnetworking/definition/TCP-IP

4. Understanding RSSI levels | MetaGeek. (n.d.). Fix-mix. https://www.metageek.com/training/resources/understanding-rssi/

5. Understanding CSI | Hands-on Wireless Sensing with Wi-Fi: A Tutorial. (n.d.). https://tns.thss.tsinghua.edu.cn/wst/docs/pre

6. I2C - SparkFun Learn. (n.d.). https://learn.sparkfun.com/tutorials/i2c/all

7. Adafruit Industries. (2021). Adafruit AHT20 Temperature & Humidity Sensor – Guide. https://cdn-learn.adafruit.com/downloads/pdf/adafruit-aht20.pdf

8. Adafruit Industries. (2014). MPU6050 6-DOF Accelerometer and Gyro – Guide. https://cdn-learn.adafruit.com/downloads/pdf/mpu6050-6-dof-accelerometer-and-gyro.pdf

9. Digi-Key Electronics. (2021). INA219 Product Overview – Current Sensor Breakout Board, Adafruit Industries. https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/1860/904_Web.pdf

10. Sharma, P. (2025, May 1). K-Means Clustering Algorithm. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/#What_Is_K-Means_Clustering

11. Rendeiro, P., Leite, J. L., Silva, L., Silva, M., Sales, G., & Ramalho, L. (2022). Performance evaluation of ESP32 in outdoor links. Anais Do XL Simpósio Brasileiro De Telecomunicações E Processamento De Sinais. https://doi.org/10.14209/sbrt.2022.1570825034

12. Fidel Alejandro Rodríguez Corbo (2025) Github Repo. https://github.com/alek-07/PDS_IMG

Appendix

The following link contains a onedrive chapter where all relevant files for the project reside (GND.ino, SAT.ino, User Interface app, image compression codes and the schematic in pdf)