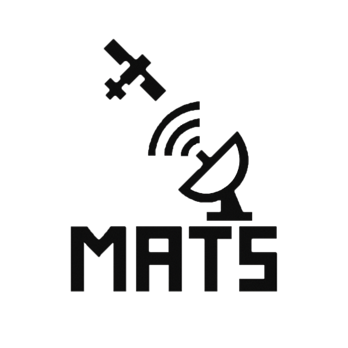
MATS Subsystem Test Plans

Mobile Antenna Tracking System (MATS)

9/26/2025

Zachary Murray, Dylan Vilcek, Ma Minghao



This document outlines the subsystem verification and test plan for the MATS. This document will outline the objectives of testing each subsystem, equipment and software required, as well as the procedures to verify the proper operation of the individual subsystems for successful integration into the overall system.

Table of Contents

[Receiver 4](#_Toc209781347)

[Subsystem Overview 4](#_Toc209781348)

[Subsystem Requirements and Specifications 4](#_Toc209781349)

[Objectives 5](#_Toc209781350)

[Required Equipment 5](#_Toc209781351)

[Testing Procedure 6](#_Toc209781352)

[Correct Drivers Installed 6](#_Toc209781353)

[I2C Interface Enablement 6](#_Toc209781354)

[Installed Software Stack 7](#_Toc209781355)

[Satellite Reception 8](#_Toc209781356)

[Satellite Transmission Decoding 9](#_Toc209781357)

[Subsystem Test Results 9](#_Toc209781358)

[User Interface 10](#_Toc209781359)

[Subsystem Overview 10](#_Toc209781360)

[Subsystem Requirements and Specifications 10](#_Toc209781361)

[Objectives 10](#_Toc209781362)

[Required Equipment 11](#_Toc209781363)

[Testing Procedure 11](#_Toc209781364)

[Subsystem Test Results 11](#_Toc209781365)

[Rotator 12](#_Toc209781366)

[Subsystem Overview 12](#_Toc209781367)

[System Requirements and Specifications 12](#_Toc209781368)

[Subsystem Requirements and Specifications 13](#_Toc209781369)

[Objectives 14](#_Toc209781370)

[Required Equipment 14](#_Toc209781371)

[Testing Procedure 15](#_Toc209781372)

[System Self-Test 15](#_Toc209781373)

[Azimuth Mobility Test 16](#_Toc209781374)

[Elevation Mobility Test 18](#_Toc209781375)

[Command Response Verification Test 19](#_Toc209781376)

[Rotational Mobility Test 21](#_Toc209781377)

[Rotational Mobility Test by I2C 22](#_Toc209781378)

[Power Consumption Test 24](#_Toc209781379)

[Subsystem Test Results 25](#_Toc209781380)

[Power Delivery 26](#_Toc209781381)

[Subsystem Overview 26](#_Toc209781382)

[Subsystem Requirements and Specifications 26](#_Toc209781383)

[Objectives 26](#_Toc209781384)

[Required Equipment 27](#_Toc209781385)

[Testing Procedure 27](#_Toc209781386)

[Input Verification 27](#_Toc209781387)

[Voltage Accuracy 27](#_Toc209781388)

[Load Regulation and Current Capacity 28](#_Toc209781389)

[Ripple and Noise 28](#_Toc209781390)

[Protection Testing 28](#_Toc209781391)

[Subsystem Test Results 29](#_Toc209781392)

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Author** | **Description** |
| 0.1 | 9/26/2025 | MATS Team | Initial draft created |
|  |  |  |  |
|  |  |  |  |

# Receiver

## Subsystem Overview

The receiver is responsible for being the central processor for the MATS. Its duties include being able to track a satellite given the TLE’s in Gpredict, receiving signals from the tracked satellites and decode the signals using SatDump. The receiver is equipped with an RTL-SDR for signal reception and utilizes a Raspberry Pi 5 running Raspian OS for its wide software support and community. Additionally, the Receiver subsystem will be equipped with GPS such that SatDump can accurately decode the received signals with map overlays and timestamps. To support the tracking of the satellite, the Receiver utilizes its GPS coordinates and the TLE data given by Gpredict and sends position data to the Rotator.

## Subsystem Requirements and Specifications

To enable the Receiver to function accurately and be integrated into the MATS successfully, the Receiver has the following signal characteristics and requirements.

Table : Receiver Subsystem Requirements

|  |  |
| --- | --- |
| **Receiver Subsystem** |  |
| Power Requirement | 5VDC 5A |
| I/O Characteristics | USB2-A Interface HDMI I2C |
| Receiving Equipment | 50Ω Antenna line  External SMA Antenna Connector |
| Software | Raspbian OS  SatDump  Gpredict  RTL-SDR drivers/software |

## Objectives

The objectives for testing the receiver include verifying that the receiver has the correct drivers and software stack, being able to receive satellite communication downlinks, and decode these downlinks into meaningful data. Subsystem components that will be tested include:

* Installation of the correct RTL-SDR Drivers
* I2C enabled
* Correct software stack installed
  + SatDump
  + GPredict
  + GPS Driver
  + SDR++
* Reception of common satellites
* Satellite transmission decodes

These objectives ensure that the receiver can correctly send commands via I2C, receive and decode Satellite transmissions. Early testing to verify the I2C bus sends the correct 32 bits will also be executed, ensuring smooth integration with the rotator.

## Required Equipment

The required equipment to test the Receiver can be found below. Additionally, the testing scripts and programs can be found at the project’s GitHub[[1]](#footnote-2). Required hardware testing is as follows:

* Receiver hardware
* Analog Discovery 2
* V-Dipole if testing VHF or equivalent L-band antenna if testing using L-band (can substitute for other antennas as region requires.)
* Multimeter

Optional equipment is listed below for use with certain testing scripts and hardware but not necessarily required:

* Laptop with serial port
* NanoVNA
* LNA/LNB for satellite reception (Satellite Dependent)

## Testing Procedure

### Correct Drivers Installed

Installation of the correct RTL-SDR drivers is crucial. The installer script should fetch the latest version, build, and install the correct driver. However, sometimes Udev rules can become a problem and not allow the correct RTL-SDR driver to initialize; it’s important that it is checked to verify that the correct driver is installed, and the generic driver be disabled. To ensure that the correct drivers are installed, [placeholder] outlines the procedure to be followed.

Table : Correct RTL-SDR Driver Install Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Check that DVB and rtl12832 drivers are not loaded using lsmod | Drivers not loaded |  |
| Plug in RTL-SDR V4 into Receiver | N/A | N/A |
| Run lsusb to confirm the device is detected | RTL-SDR found |  |
| Run rtl\_test -t or rtl\_test -p to verify the driver claims the device | No "usb\_open error" message |  |

### I2C Interface Enablement

Verify that the Raspberry Pi’s I2C interface is enabled by the installer script and accessible by user-space applications. Table 3 outlines this procedure.

Table : I2C Interface Testing

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Inspect /boot/firmware/config.txt |  |  |
| Confirm dtparam=i2c\_arm=on is present and not commented out | Line not commented out |  |
| Verify the invoking user has been added to the i2c group |  |  |
| run the following: groups $USER | grep i2c | User Added |  |
| Verify the kernel driver is loaded. Run:  lsmod | grep i2c\_bcm | Driver loaded |  |
| With the system booted, execute:  sudo i2cdetect -y 1 | A grid of addresses is displayed |  |
| Use the Analog Discovery 2 to verify I2C traffic using the provided i2c\_ad2\_test.py script. | I2C present on AD2 |  |

### Installed Software Stack

The receiver must have the correct software stack for correct operation with the other subsystems. This test section ensures that the correct software, such as GPS drivers, SatDump, Gpredict, and SDR++ are installed on the system. Table 4 shows the procedure. A test script has been provided in the github repository for automated testing.

Table : Receiver Software Stack Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Run automated verification script | Script executes and prints results for each component. |  |
| Check GPS drivers | Output shows: [+] Checking gpsd... PASS |  |
| Check SatDump installation | Output shows: [+] Checking SatDump... PASS |  |
| Check Gpredict installation | Output shows: [+] Checking Gpredict... PASS |  |
| Check SDR++ installation | Output shows: [+] Checking SDR++... PASS |  |
| Review summary | Script ends with === Verification complete === and no FAIL messages. |  |

### Satellite Reception

Successful satellite reception is required for the MATS to perform it’s core functionality. Shown below in Table 5, the testing procedure verifies that that SatDump can record a satellite transmission.

Table : Satellite Reception Test

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect RTL-SDR Blog V4 dongle to Pi USB 3.0 port and attach VHF antenna (QFH, V-dipole, turnstile, etc.) tuned for ~137 MHz. | Hardware connected firmly, antenna outdoors or with clear sky view. |  |
| Use Gpredict to check upcoming Meteor-M2-4 pass (137.9 MHz). Record start time, max elevation, and pass duration. | Pass info visible, frequency: **137.900 MHz**. |  |
| Launch SDR++ and configure RTL-SDR input. Set center frequency to 137.9 MHz and sample rate to ~2.048 MSPS. | Device initializes without error, spectrum visible. |  |
| Observe waterfall during predicted pass. | Wideband (~140 kHz) signal appears centered at 137.9 MHz, rising above noise floor as pass begins. |  |
| (Optional) Verify via SatDump spectrum mode without decoding. | Console/log shows device streaming, RF power increases near 137.9 MHz during pass. |  |
| Monitor until end of pass. | Signal fades as satellite sets. No USB or driver errors reported. |  |

### Satellite Transmission Decoding

Testing of the receiver’s decoding stack ensures that satellite transmissions are accurately reconstructed with the correct GPS coordinates and map overlays. Table 6 below outlines the testing procedure.

Table : Satellite Transmission Decode Test

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect RTL-SDR Blog V4 dongle to Pi USB 3.0 port and antenna suitable for target satellite (e.g., QFH for VHF). | Hardware connected firmly, antenna with clear sky view. | Hardware connected firmly, antenna with clear sky view. |
| Use Gpredict to schedule and monitor an upcoming pass of a decodable satellite (e.g., NOAA APT at 137.1–137.9 MHz or Meteor-M2-4 at 137.9 MHz). | Pass data visible with start time, elevation, and duration. |  |
| Launch SatDump in live mode with correct frequency and sample rate. | SatDump initializes without error, SDR engaged. |  |
| Observe console/log during pass. | Frames are received and decoded; progress messages visible. |  |
| At end of pass, review generated products (e.g., map overlays, imagery, data files). | Output files (e.g., PNG, HDF5) are generated in SatDump output directory with valid content. |  |

## Subsystem Test Results

After execution of all test sections (drivers, I2C, software stack, reception, and decoding), results should be consolidated in the following table.

|  |  |  |
| --- | --- | --- |
| **Test Section** | **Pass/Fail** | **Notes** |
| RTL-SDR Driver Install |  |  |
| I²C Interface Enablement |  |  |
| Software Stack Verification |  |  |
| Satellite Reception (M2-4, 137.9 MHz) |  |  |
| Satellite Transmission Decoding |  |  |

# User Interface

## Subsystem Overview

The User Interface subsystem provides the primary operator touchpoint within the MATS. It delivers feedback through a 7” touchscreen, communicates system status via LED indicators, and provides removable media access through USB and SD ports. The UI runs a lightweight Raspberry Pi OS desktop, auto-launching SatDump at startup. Status LED’s are driven using a Python daemon that subscribes to SatDump telemetry, providing live feedback for RF lock, recording, and power state.

## Subsystem Requirements and Specifications

Table : User Interface Subsystem Specifications

|  |  |
| --- | --- |
| Category | Requirement |
| Display | 7 Inch capacitive touchscreen with Linux driver support. |
| Inputs/outputs | USB-A user port, SD/microSD card reader, power switch, RF & Power status LEDs. |
| Environmental | Operating 0ºC-50ºC, front panel IP-54 splash resistance |
| Electrical | Single 5V DC rail, 2A steady-state draw |
| Software | I2C touch controller, GPIO LEDs |
| Mechanical | Front-panel mounting to aluminum chassis, maximum depth behind panel mm |

## Objectives

The testing objectives for the User Interface are:

* Verify touchscreen responsiveness and accuracy
* Confirm automatic startup and SatDump launch.
* Validate USB and SD card hot-plug enumeration
* Verify LED indicators function under scripted test
* Confirm LEDs respond correctly to SatDump Telemetry.

These objectives ensure the User Interface provides reliable control and feedback to operators.

## Required Equipment

* UI Hardware
* Test Media: 32 GB microSD card and 128 GB USB Flash drive
* Multimeter
* Python I2C test script (led\_test.py

## Testing Procedure

Testing of the user interface is primarily focused on the core functionality with interacting with the MATS. Table 8 shows the tests to be performed for successful verification.

Table : User Interface Testing

|  |  |  |  |
| --- | --- | --- | --- |
| **Test** | **Method** | **Expected Result** | **Observed Result** |
| Display Touch Response | Run 10-point accuracy grid or calibration app | Touch deviation < 2 mm |  |
| SatDump Auto-Launch | Boot system and observe startup | Pi logs into desktop and SatDump auto-launches within ~15 s |  |
| USB Enumeration | Hot-plug 128 GB USB drive | Drive mounts automatically in <3 s |  |
| SD Card Enumeration | Hot-plug 32 GB microSD card via panel reader | Card mounts automatically in <3 s |  |
| LED Functionality | Run led\_test.py scripted I²C pattern test | Each LED (Power, RF, Recording) cycles correctly through colors |  |
| LED Telemetry Response | Start/stop SatDump recording session | RF/Recording LEDs toggle in real time with telemetry |  |

## Subsystem Test Results

|  |  |  |
| --- | --- | --- |
| **Test Section** | **Pass/Fail** | **Notes** |
| Touchscreen Accuracy |  |  |
| Auto-Launch of SatDump |  |  |
| USB Enumeration |  |  |
| SD Card Enumeration |  |  |
| LED Functionality |  |  |
| LED Telemetry Response |  |  |

# Rotator

## Subsystem Overview

The purpose of this subsystem verification plan is to establish a structured and repeatable process for verifying the performance, functionality, and compliance of the Rotator subsystem. This document provides a detailed framework for conducting verification activities, ensuring that all requirements are met before integration into the larger MATS. By following this plan, verification activities will be performed efficiently, ensuring the subsystem meets its design and functional requirements before proceeding to the next phase of system development.

## System Requirements and Specifications

The MATS aims to be small, lightweight, and cost effective, providing an encompassing solution compared to other commercial systems, which are large, heavy, and expensive. The MATS encompasses all the hardware required to accurately position the antenna, according to the TLE information for a given satellite. Utilizing GPS for positional data, the only input the user requires is to know which satellite is passing overhead. The MATS will be deployed by supplying power from any power system onboard many aircraft, marine vessels, emergency vehicles, and power systems commonly found in developing communities. Figure 1 shows the overall system diagram, showing how the subsystems interact with each other. The requirements enable the MATS to be low cost and reliable, while still having accurate satellite tracking.

A diagram of a power supply system

AI-generated content may be incorrect.

Figure : System Diagram

## Subsystem Requirements and Specifications

The Rotator system is responsible for rotating the antenna, moving it to a position given by the receiver system to track low earth orbit satellites. Due to the requirements given by the receiver, this system needs to receive an I2C signal for obtaining the azimuth and elevation location required and positioning the antenna to the specific coordinate within the specific time. System requirements can be found in Table 16.

Table : Rotator Subsystem Requirements

|  |  |
| --- | --- |
| **Requirement** | **Expectation** |
| Power Requirement | 3V3, 12VDC |
| Interface | I2C |
| Azimuth Accuracy | ±1.6° |
| Elevation Accuracy | ±1.6° |
| Power Consumption | < 75W |
| Cost | < $110 |

A diagram is provided in Figure 2 outlining the functional blocks and connections within the Rotator.

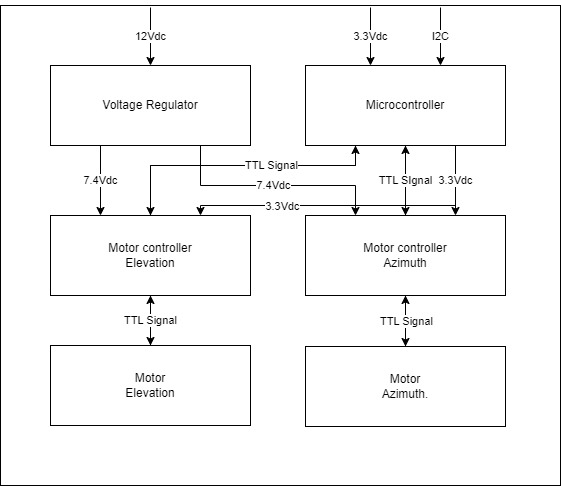


Figure : Rotator Block Diagram

These requirements ensure the proper operation of the Rotator, allowing the MATS to accurately track low earth orbit satellites.

## Objectives

These tests are intended to test the Rotator for proper operation. The rest of this document outlines the equipment and tests required to confirm that the Rotator is working as intended, and ready to be integrated into the larger MATS system. In these tests, the functional tests include:

* System Self-Test
* Azimuth Mobility Test
* Elevation Mobility Test
* Command Response Verification Test
* Rotational Mobility Test
* Power Consumption Test

## Required Equipment

For the following tests to be performed, some equipment should be obtained. This equipment is chosen for its simplicity of use and being readily available. The equipment required to perform these tests include:

* Laptop
* DMM
* Power Supply
* I2C Generator (raspberry pi, microcontroller, analog discovery)
* Stopwatch
* Protractor
* Wattmeter
* Compass

## Testing Procedure

This section details the verification procedures required to validate the Rotator. Each test ensures compliance with functional, performance, and reliability requirements. Data acquired from these tests should be documented in the provided tables.

### System Self-Test

The system self-test’s purpose is to verify that the azimuth and elevation motors start, perform the respective self-checks, and return to the original position. Table 10 outlines the procedure and measurements to be taken.

Table : System Self-Test Procedure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect power supply to Subsystem Under Test (SUT) as shown in Figure 3. | N/A |  |  |
| 2 | Using a protractor, take an initial position measurement at 90 degrees. | 90 |  |  |
| 3 | Turn on the power to the SUT and observe the behavior. | N/A |  |  |
| 4 | Verify the servo motors have performed their self-start checks. | Servo 1 and 2 online |  |  |
| 5 | Using the protractor, measure the position deviation from original position measured in step 2. | < 1.8° |  |  |

If the Rotator’s motors successfully perform the startup self-checks and return to the original position, the Rotator should be tested for functional performance.

A diagram of a power supply system

AI-generated content may be incorrect.

Figure : Mobility Test Setup

### Azimuth Mobility Test

The Azimuth Mobility Test verifies that the Rotator is capable of completely rotating 360 degrees in Azimuth. This check ensures that the Rotator will not need to “flip” when the satellite is passing directly overhead. This ensures that the signal will not be impacted while tracking directly overhead satellites.

To perform this test, the system should still be connected to power and the following steps outlined in Table 11 are to be performed.

Table : Azimuth Mobility Test Procedure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect Test Equipment and power supply to Subsystem Under Test (SUT) as shown in Figure 3. | N/A |  |  |
| 2 | Using a compass, take an initial measurement position measurement. | N/A |  |  |
| 3 | Using the code provided in Figure 4, send a control input that moves the azimuth to rotate 90º. | N/A |  |  |
| 4 | Utilizing the stopwatch, measure the amount of time taken to complete the rotation. | <180s |  |  |
| 5 | Using the compass, measure the azimuth angle after rotation. | 90° ± 1.8° |  |  |

A screenshot of a computer program

AI-generated content may be incorrect.

Figure : Azimuth Test Code

This check determines the freedom of movement of the Rotator subsystem’s azimuth direction. Upon failure, determine failure cause and document accordingly.

### Elevation Mobility Test

The Elevation Mobility Test determines the Rotator’s freedom in elevation. The elevation is where the antenna is pointed and is primarily responsible for holding the antenna at a given height above the horizon. This operation is required for proper signal strength, as deviations in elevation can lead to signal loss.

The procedure to outline this test is shown in Table 12.

Table : Elevation Mobility Test Procedures

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect Test Equipment and power supply to Subsystem Under Test (SUT) as shown in Figure 3. | N/A |  |  |
| 2 | Using a compass, take an initial Elevation measurement. | N/A |  |  |
| 3 | Using the code provided in Figure 5, send a control input that moves the elevation to 90º in elevation. | N/A |  |  |
| 4 | Utilizing the stopwatch, measure the amount of time taken to complete the rotation. | <180s |  |  |
| 5 | Using a compass, measure the elevation angle after the rotation is complete. | 90° ± 1.8° |  |  |

A computer screen with text and numbers

AI-generated content may be incorrect.

Figure : Elevation Test Code

Verification of this system ensures that the Rotator has freedom of movement in elevation, which enables smooth movement allowing for the least amount of signal loss. Upon failure, determine failure cause and document accordingly.

### Command Response Verification Test

The command response verification test ensures that the system can receive a command and move to the given coordinates. This test verifies that the Rotator can smoothly operate both the azimuth and elevation at the same time, ensuring that there is minimal signal loss during tracking.

To perform this test, follow the procedure outlined in Table 13.

Table : Command Response Verification Test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect Test Equipment and power supply to Subsystem Under Test (SUT) as shown in Figure 3. | N/A |  |  |
| 2 | Using a protractor, take an initial Elevation measurement at 0 degrees. | N/A |  |  |
| 3 | Using the protractor, take an initial azimuth measurement at 90 degrees. | N/A |  |  |
| 4 | Using the code provided in Figure 6, send a control input that moves 72º azimuth and 36º in elevation. | N/A |  |  |
| 5 | Utilizing the stopwatch, measure the amount of time taken to complete the movement | <180s |  |  |
| 6 | Using a compass, measure the elevation angle after the rotation is complete. | 36 ± 1.8° |  |  |
| 7 | Using a compass, measure the azimuth angle after the rotation is complete. | 72 ± 1.8° |  |  |

A computer screen shot of a program

AI-generated content may be incorrect.

Figure : Command Response Test Code

Completion of this test ensures that the Rotator is capable of smoothly tracking and positioning an antenna where it needs to be for obtaining a good signal. Upon failure, determine cause and document accordingly.

### Rotational Mobility Test

The rotational mobility test will determine if the Rotator is capable of operating to the limit of azimuth and elevation required for accurate satellite tracking. This test will verify that the Rotator can move to more extreme values of azimuth and elevation.

To perform this test, follow the steps outlined in Table 14.

Table : Rotational Mobility Test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect Test Equipment and power supply to Subsystem Under Test (SUT) as shown in Figure 3. | N/A |  |  |
| 2 | Using a compass, take an initial Elevation measurement. | N/A |  |  |
| 3 | Using the compass, take an initial azimuth measurement. | N/A |  |  |
| 4 | Using code provided in Figure 8 And use Python code provided by Figure 9. send a control input that moves 172° azimuth and 86° in elevation. | N/A |  |  |
| 5 | Utilizing the stopwatch, measure the amount of time taken to complete the movement | <180s |  |  |
| 6 | Using a compass, measure the elevation angle after the rotation is complete. | 86 ± 1.8° |  |  |
| 7 | Using a compass, measure the azimuth angle after the rotation is complete. | 172 ± 1.8° |  |  |

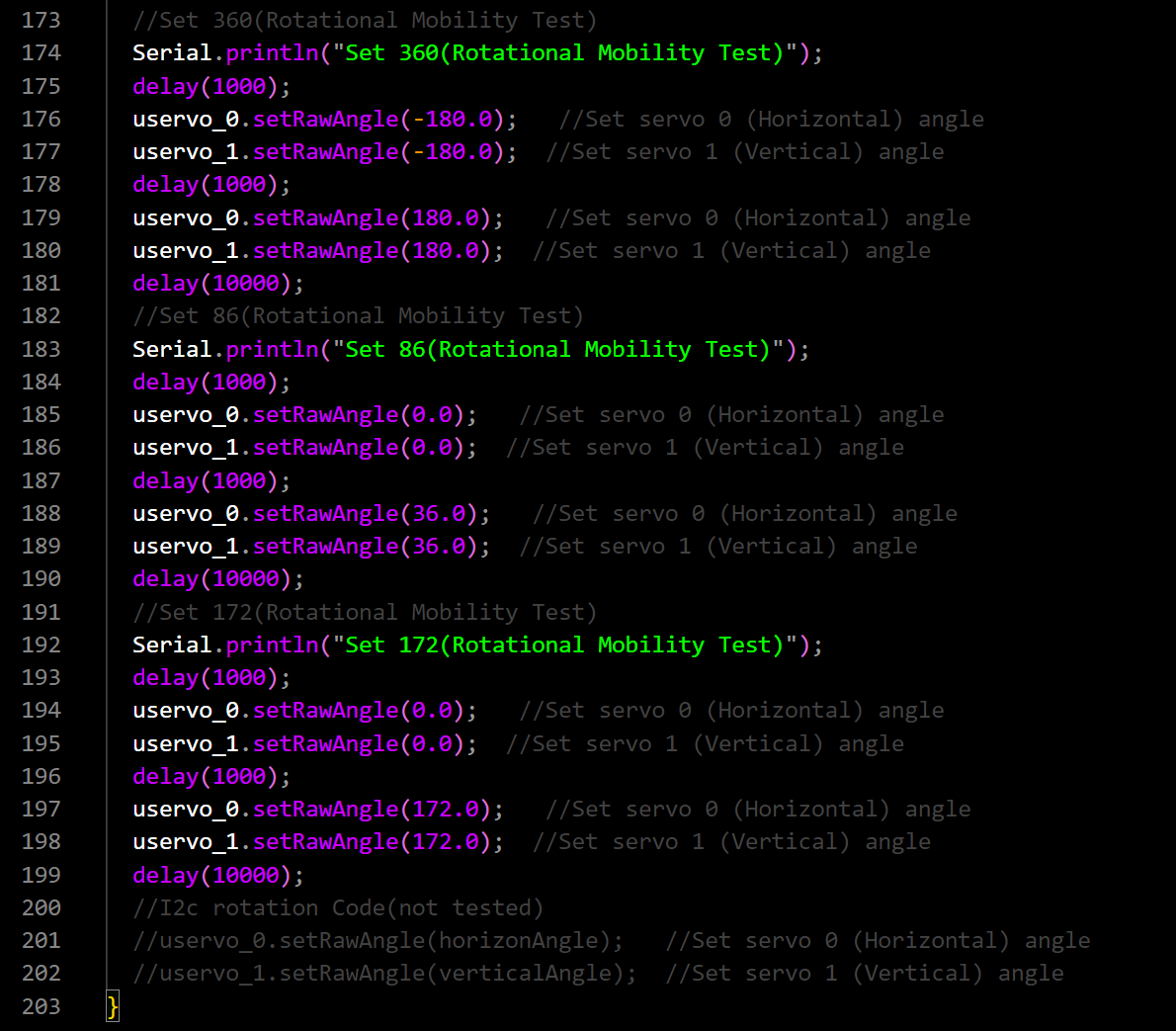


Figure : Rotational Mobility Test

Completion of this test ensures that the Rotator is capable of operating at the limits of required angles for tracking low earth orbit satellites. Upon failure, determine cause and document accordingly.

### Rotational Mobility Test by I2C

The rotational mobility test will determine if the Rotator is capable of operating to the limit of azimuth and elevation required for accurate satellite tracking by receiving system. This test will verify that the Rotator can receive correct messages from receive system and keep tracking right position.

To perform this test, follow the steps outlined in Table 14.

Table 7: Rotational Mobility Test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect Test Equipment and power supply to Subsystem Under Test (SUT) as shown in Figure 3. | N/A |  |  |
| 2 | Using a compass, take an initial Elevation measurement. | N/A |  |  |
| 3 | Using the compass, take an initial azimuth measurement. | N/A |  |  |
| 4 | Using code provided in Figure 7, send a control input that moves 172° azimuth and 86° in elevation. | N/A |  |  |
| 5 | Utilizing the stopwatch, measure the amount of time taken to complete the movement | <180s |  |  |
| 6 | Using a compass, measure the elevation angle after the rotation is complete. | 86 ± 1.8° |  |  |
| 7 | Using a compass, measure the azimuth angle after the rotation is complete. | 172 ± 1.8° |  |  |

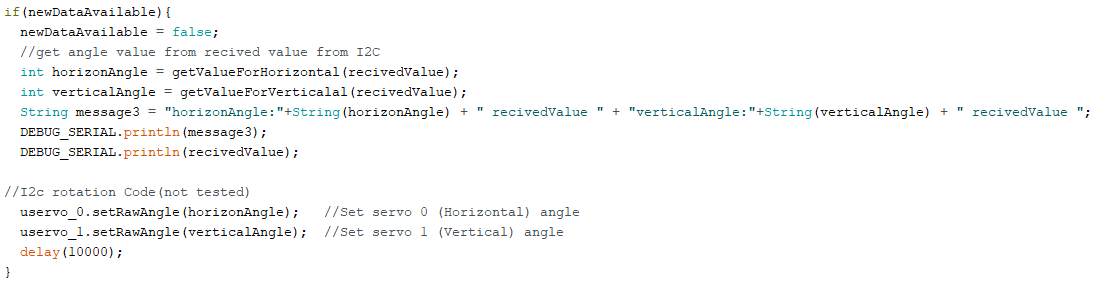


Figure 8: Rotational Mobility Test

Completion of this test ensures that the Rotator can receive satellites location from reciver for tracking low earth orbit satellites. Upon failure, determine cause and document accordingly.

### Power Consumption Test

The power consumption test verifies that the MATS has a power consumption that enables it to perform utilizing available power on board many aircraft, emergency vehicles, and maritime vessels. Table 15 outlines the procedure to successfully perform these checks, and Figure 8 shows the set equipment setup.

Table : Power Consumption Test Procedure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test/Verification Step | Expected Value | Measured Value | Pass or Fail |
| 1 | Connect Test equipment and the subsystem under test (SUT) as shown in Figure 8. | N/A |  |  |
| 2 | Connect Power Supply to SUT. | N/A |  |  |
| 3 | Using I2C, send a control input that moves 90º azimuth and 45º in elevation. | N/A |  |  |
| 4 | Record power consumed by the SUT as measured by a wattmeter | < 50W |  |  |

The power consumption test validates that the rotator operates with a power consumption that will not damage fuses, wiring and other equipment on board airplanes, emergency vehicles and maritime vessels.

A diagram of a power supply system

AI-generated content may be incorrect.

Figure : Power Consumption Equipment Setup

## Subsystem Test Results

The successful execution of this subsystem verification test plan ensures that the Rotator subsystem meets all functional, performance, and reliability requirements before integration into the MATS. Following the verification procedures outlined in this document, a skilled technician can systematically validate the subsystem’s compliance with the design specifications, identify potential issues, and document verification results accordingly.

# Power Delivery

## Subsystem Overview

System overview and verification are critical to ensure that the MATS performs as intended. This document outlines the testing procedures for the Power Subsystem, which is responsible for converting the system’s 12–24 V input into regulated voltage rails required by all other subsystems. The Power Subsystem provides 7.5 V, 5 V, and 3.3 V outputs to support the Rotator, Receiver, and User Interface subsystems. This test plan aims to verify the proper functionality and reliability of the Power Subsystem independently, prior to integration into the full MATS system.

## Subsystem Requirements and Specifications

Using MP2329C buck converters, the design generates 7.5 V, 5 V, and 3.3 V outputs to supply the Rotator, Central Receiver, and User Interface. To ensure reliable operation, the subsystem must meet requirements for voltage accuracy, current delivery, ripple and noise, efficiency, and fault protection. System requirements can be found in Table 1.

Table : Rotator Subsystem Requirements

|  |  |
| --- | --- |
| **Requirement** | **Expectation** |
| Input Voltage Range | 12-24 VDC |
| Output Rails | 7.5V, 5V, 3v3 |
| Maximum Current | 6A, 5A, 5A |
| Voltage Accuracy | ± 5% of nominal |
| Ripple Voltage | ≤ 50 mVpp |
| Protection | Overcurrent and short-circuit safe |

## Objectives

The objectives to test and verify operation for include:

* Verify each output rail maintains voltage within tolerance across the load
* Confirm maximum current delivery for each rail
* Measure and evaluate ripple/noise
* Validate startup, shutdown, and sequencing behavior
* Test protection circuitry to ensure safe fault handling

## Required Equipment

The following is a list of the equipment needed to carry out the tests and verification plans for the Power Subsystem of the MATS.

* Programmable DC power supply
* Adjustable resistive loads
* Digital multimeter
* Oscilloscope

## Testing Procedure

This section covers in detail the testing procedure for each part of the Power Subsystem.

## Input Verification

Table : Input Verification Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect programmable DC power supply to subsystem input. | N/A | N/A |
| Connect DMM to measure input voltage/current. | N/A | N/A |
| Apply 12 V input with no load attached. | Subsystem powers on, outputs stable. |  |
| Increase input to 14 V, no load. | Outputs remain stable. |  |
| Repeat in 2 V increments up to 24 V, no load. | Outputs remain stable at all input voltages. |  |
| Apply representative load to each rail at 12 V input. | Subsystem powers on, outputs remain stable. |  |
| Repeat input sweep (12–24 V) with load attached. | Outputs remain stable within tolerance across input range. |  |

## 

## Voltage Accuracy

Table : Voltage accuracy Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect DMM probes to each output rail (7.5 V, 5 V, 3.3 V). Apply nominal 12 V input, no load. | All rails within ±5% of nominal. |  |
| Apply nominal input with rated load on 7.5 V rail, measure output. | Voltage within ±5% of nominal. |  |
| Apply nominal input with rated load on 5 V rail, measure output. | Voltage within ±5% of nominal. |  |
| Apply nominal input with rated load on 3.3 V rail, measure output. | Voltage within ±5% of nominal. |  |

## Load Regulation and Current Capacity

Table : Load Regulation and Current Capacity Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect adjustable load to 7.5 V rail. Increase current gradually to rated limit. | Voltage deviation remains within spec. |  |
| Repeat test for 5 V rail. | Voltage deviation within spec. |  |
| Repeat test for 3.3 V rail. | Voltage deviation within spec. |  |
| Load all rails simultaneously to rated current. | Subsystem remains stable, no shutdown/thermal fault. |  |

## Ripple and Noise

Table : Ripple and Noise Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect oscilloscope probe across 7.5 V rail under nominal load. Measure Vpp. | Ripple ≤ 50 mVpp. |  |
| Repeat for 5 V rail. | Ripple ≤ 50 mVpp. |  |
| Repeat for 3.3 V rail. | Ripple ≤ 50 mVpp. |  |

## Protection Testing

Table : Protection Testing Test Procedure

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Result** | **Observed Result** |
| Connect electronic load, increase 7.5 V rail current above rated limit. | Overcurrent protection engages, subsystem safe. |  |
| Repeat test on 5 V rail. | Protection engages safely. |  |
| Repeat test on 3.3 V rail. | Protection engages safely. |  |
| Briefly short 7.5 V rail using current-limited supply. | Subsystem shuts down or limits safely, recovers after fault. |  |
| Repeat short-circuit test for 5 V rail. | Subsystem recovers after fault. |  |
| Repeat short-circuit test for 3.3 V rail. | Subsystem recovers after fault. |  |

## Subsystem Test Results

|  |  |  |
| --- | --- | --- |
| **Test Section** | **Pass/Fail** | **Notes** |
| Input Verification |  |  |
| Voltage Accuracy |  |  |
| Load Regulation & Current Capacity |  |  |
| Ripple and Noise |  |  |
| Protection Features |  |  |

1. <https://github.com/ZacharyRMu/MATS> [↑](#footnote-ref-2)