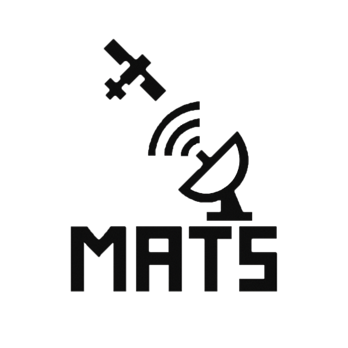
MATS Subsystem Test Plans

Mobile Antenna Tracking System (MATS)

9/25/2025

Zachary Murray, Dylan Vilcek, Ma Minghao



This document outlines the subsystem verification and test plan for the [subsystem name] subsystem aboard the MATS. This document will outline the objectives of the test, equipment and software required, as well as the procedures to verify the proper operation of the subsystem.

Table of Contents

[Subsystem Overview 3](#_Toc209683898)

[Subsystem Requirements and Specifications 3](#_Toc209683899)

[Objectives 3](#_Toc209683900)

[Required Equipment 4](#_Toc209683901)

[Testing Procedure 4](#_Toc209683902)

[Correct Drivers Installed 4](#_Toc209683903)

[I2C Interface Enablement 5](#_Toc209683904)

[Installed Software Stack 5](#_Toc209683905)

[Satellite Reception 6](#_Toc209683906)

[Satellite Transmission Decoding 7](#_Toc209683907)

[Subsystem Test Results 8](#_Toc209683908)

## 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 : 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. [placeholder] 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

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. [placeholder] 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 |  |  |

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