University of Auckland: Ground Station One

System Overview

Sean Richards

The University of Auckland

Abstract

Historically, the biggest barrier for a manmade object to enter space has been the cost of materials and parts. But, with the advent of small 1U CubeSats we have seen an explosion in the number of satellites entering space. This relative expansion has given rise to Auckland Programme for Space Systems (APSS) and Te Pūnaha Ātea (Auckland Space Institute). The University's continuing investment into space research necessitates an upgrade to the existing ground station established in 2017 both to increase its capability, and to serve as a way for students to informally monitor spacecraft such as the APSS-1, APSS-2, and APSS-3 satellites, with more satellites in the program being designed each year. In this work, we present a detailed overview of the individual components of the ground station, an examination of how the system works holistically, and a suggested outline of future work to upgrade the station.

Acknowledgements

Kua tawhiti kē to haerenga mai, kia kore e haere tonu. He nui rawa o mahi, kia kore e mahi tonu.
You have come too far not to go further, you have done too much not to do more.

Māori Whakataukī

I am as always eternally thankful to Dr Nicholas Rattenbury, Dr John Cater, Mr Francis Moynihan Lavey (all of the University of Auckland), and Mr Dimitrios Tsifakis (of the Australian National University) for their advice in the upgrade, and helping to learn the ropes of amateur satellite tracking.

I must also acknowledge the campus radio station 95bFM. Although I am not a regular listener, their station as a known frequency rapidly became a benchmark to test that the throughput from the ground station to the analysis machine worked as intended.

Thanks, in no small part, to the seven people who got me into science in the first instance:

- Dr Derek Miller (of Veritasium),
- Ms Dianna Cowern (of PhysicsGirl),
- Mr Robert Lumkong, Mr Andy Dyson, and especially Mr Chris Whitaker (all of Kerikeri High School),
- Prof. Stephen Hawking,
- and of course, my mother, Ms Sue Richards.

Contents

List of Figures

Glossary

Notation	**Description**
AOS	Acquisition of Signal.
APSS	Auckland Programme for Space Systems.
LEO	Low-Earth Orbit.
LOS	Loss of Signal.
OSCAR	Orbiting Satellite Carrying Amateur Radio.
SDR	Software Defined Radio.
SGP	Simplified Perturbation Model.
TLE	Two Line Element set.
UHF	Ultra-High Frequency.
VHF	Very-High Frequency.

Chapter 1

Introduction

All civilizations become either spacefaring or extinct.

Carl Sagan

Space has oft been described as the final frontier; the first forms of rocketry emerged in China in the 11th century (Gongliang, Weide & Du 1044), however the first man-made object to enter outer space did not arrive until 1944 - MW 18014 launched by Germany, attaining an apogee of 176km. With the ever-advancing pace of technology, we have had twelve men walk on the moon, nearly six hundred enter outer space, and we have launched nearly five thousand satellites into space, with more entering orbit every few months. This rapid advance has enabled more and more individuals and entities to enter space, culminating in 2020 with the anticipated launch of APSS-1, the first satellite of the University of Auckland's *Auckland Programme for Space Systems (APSS).*

1.1 Prior Work and Scope

In 2019, under the supervision of Dr Nicholas Rattenbury, I was fortunate to work on the University of Auckland's ground station, under the purview of the APSS. My work (Richards 2019) formed the core of an upgrade to the University's ground station, moving it from an older LabJack (LabJack Corporation n.d.) based system that ran on Windows to a newer Arduino (*Arduino Uno rev3* n.d.) based system running on Ubuntu 19.04. This work built upon the prior work conducted by (Thirumalai 2017), and the work I carried out enabled the ground station to move closer to fulfilling its goals of being a way for student operators to informally monitor the APSS satellites.

However, as illustrated in Chapter 6 of (Richards 2019), the ground station was not rendered fully functional by the close of the year, and a tentative outline for work conducted over the summer was presented. In this report, we present a detailed discussion of that work, building towards a functional ground station that is flexible to support future missions.

The aim of this report is twofold. First, to provide a detailed overview of the design and implementation of the University of Auckland ground station. Prior documentation of the ground station was scattered and located in multiple places, hence the need for this report. Secondly, to provide a manual for a future operator to understand how the ground station works both holistically, and as individual components in a modular design.

1.1.1 Project Aims

(1) To repair the ground station – analysis machine connection as described in Chapter 3 of (Richards 2019),

(2) To complete the upgrade of the ground station from a single-mode, Ultra-High Frequency (UHF) only antenna, into a complete antenna array supporting Very-High Frequency (VHF) as well,

(a) The ground station as it existed prior to the upgrade.

(b) The ground station as it now exists after the upgrade.

Figure 1.1: The ground station before and after the upgrade

(3) To install environment sensors into the ground station control box, enabling more refined control of the conditions inside the box, and

(4) To enable the ground station to function autonomously.

Of these aims, Aim 1 and 2 were considered "mission-critical" and must be completed, Aims 3 and 4 were auxiliary aims that we wished to complete, but were not essential to completion of the project.

1.1.2 Scope of Project

APSS utilises the communications array at the Awarua Satellite Ground Station in order to provide mission-critical uplink and downlink. The purpose of *Ground Station One*[1] is to provide a platform to informally monitor the APSS satellites, experiment with amateur radio astronautics, and empower students to learn the fundamentals of how satellites communicate. Consequently, the exact methodology behind how the critical APSS satellites communicate is beyond the scope of this report.

1.2 Theory

The theory of satellite tracking is explored in an earlier dissertation (Richards 2019) and a brief refresher is presented here.

1.2.1 Modulation Schematics

We can write a carrier wave of frequency $\omega_c = 2\pi f_c$, amplitude A, and phase $\phi \in [0, 2\pi]$ as

$$x(t) = A\cos(\omega_c t + \phi) \tag{1.1}$$

This presents three parameters in which a signal $m(t)$ may be encoded.

[1] Affectionately nicknamed *Taringa*, as our "ear to the sky"

$$x(t) = A\big(1 + \Delta_{\mathrm{m}}^{\mathrm{AM}} m(t)\big)\cos(\omega_{\mathrm{c}} t + \phi) \tag{1.2}$$

$$x(t) = A\cos\left(2\pi\left(f_{\mathrm{c}} t + \Delta_{\mathrm{m}}^{\mathrm{FM}} \int m(t)\,\mathrm{d}t\right) + \phi\right) \tag{1.3}$$

$$x(t) = A\cos\big(\omega_{\mathrm{c}} t + \Delta_{\mathrm{m}}^{\mathrm{PM}} m(t) + \phi\big) \tag{1.4}$$

Where a subscript "c" indicates properties of the carrier wave, and $\Delta_{\mathrm{m}}^{\mathrm{T}}$ is the *degree of modulation coefficient* for modulation type T (i.e., Frequency Modulation (FM), Amplitude Modulation (AM), or Phase Modulation (PM)). Intuitively, Δ is a weight to assign to the modulation. If $\Delta \simeq 1$, the modulation will be less pronounced, if $\Delta \gg 1$, the modulation will appear extreme.
The resultant signal for the sample input 01000101 encoded as a pure square wave is shown in Fig. 1.4 on page 7. This admits easy demodulation via traditional radio; circuits may be implemented to remove the carrier wave using, for example, local oscillators. The advance of technology, however, has enabled radios to emerge where demodulation is implemented purely programmatically — this gives rise to Software Defined Radio (SDR), an inexpensive amateur radio implementation of radio communications.

1.2.2 Demodulation Techniques

The simplest way to achieve demodulation via SDR is to use a dipole antenna with an RTL-SDR (*RTL-SDR Blog V.3. Dongles User Guide* n.d.). The demodulation may be accomplished in software such as soundmodem (Kopanchuk n.d.)[2] or GNU Radio (GNU Radio Foundation n.d.). In GNURadio, the core of the script is the so-called *flowgraph*, which can be understood as a python or C script which may be implemented either graphically through *GNURadio Companion*, or implemented via traditional python or C code. A sample GNURadio flowgraph is shown in Fig. 1.2, and the corresponding python output is included as Appendix B.
This particular flowgraph accepts an RTL-SDR source plugged in to any USB port of the development computer. The flowgraph proceeds as follows:

(1) The *RTL-SDR Source* block is the source element for the flowgraph. Other SDR platforms are available, as well as random noise sources and TCP/UDP sources. The output of this block branches into two blocks, which are executed simultaneously.

(2) The *QT GUI* block provides a thorough graphical interface to view the (raw) signal. The GUI by default contains a frequency-gain plot, waterfall display, time domain display, and a constellation display of the signal, though these elements may be individually selected to allow the operator to construct any display they desire.

(3) The *Rational Resampler* block reconfigures the sample rate. If the input sample rate to the block is f_{s}, then the output is $i * \frac{f_{\mathrm{s}}}{d}$, where i is the *interpolation* value in the block, and d is the *decimation* value in the block.

(4) The *low pass filter* block implements a simple low pass filter to extract the given frequency from the range of frequencies incident on it. If the filtering function $\delta(f)$ is given by

$$\delta(f) = \mathcal{F}(\Delta(t)) \tag{1.5}$$

then the filtered signal $\hat{x}(t)$ of the incident signal $x(t)$ is given by

[2] Author's name anglicized

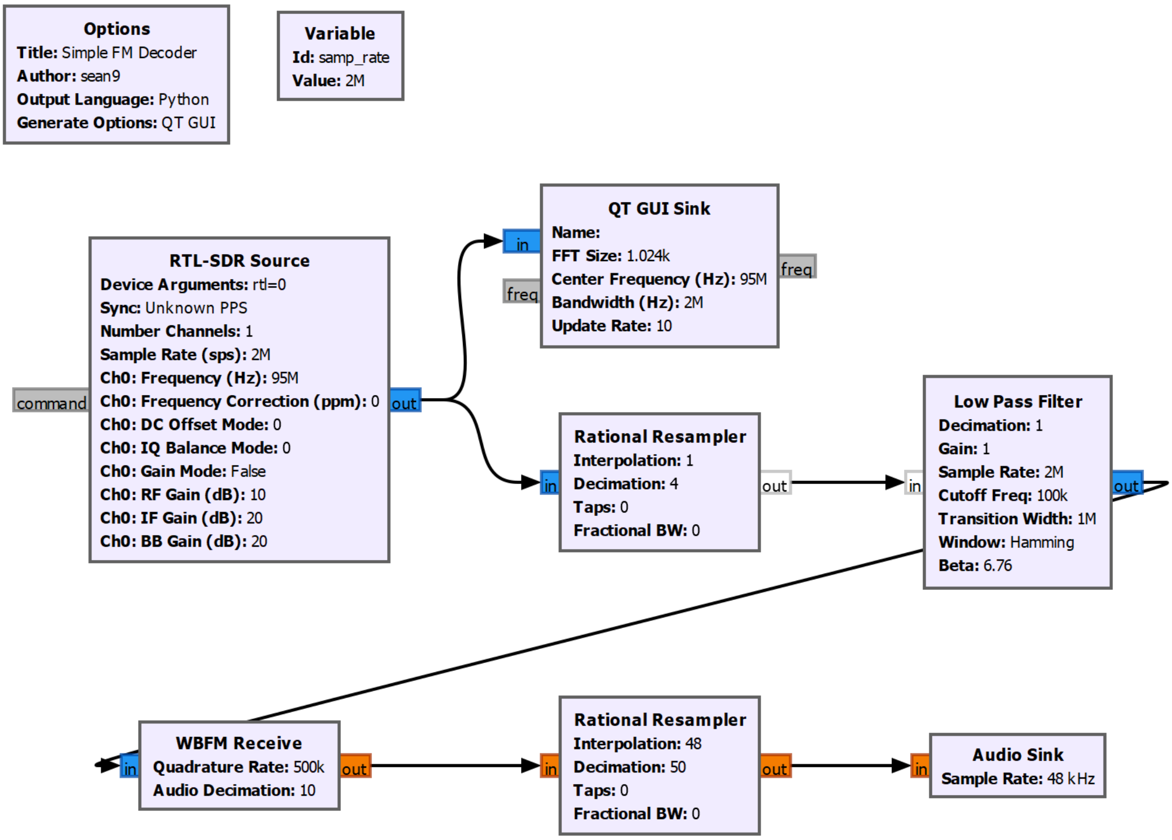


Figure 1.2: A sample GNURadio flowgraph to decode the FM radio station on 95 MHz – a campus radio station at the University called 95bFM. The flowgraph is represented pictorially here, though GNURadio is capable of exporting flowgraphs as C/C++ code, or Python code, such that it may be integrated into a ground station setup.

$$\hat{x}(t) = \mathcal{F}^{-1}(\delta(f)\mathcal{F}(x(t))) \tag{1.6}$$

$$= \mathcal{F}^{-1}(\mathcal{F}(\Delta(t))\mathcal{F}(x(t))) \tag{1.7}$$

$$= \Delta(t) * x(t) \tag{1.8}$$

by the convolution theorem.

(5) The *WBFM receive* block achieves the FM demodulation. It is effectively a "black box", insofar as one may input a modulated (perhaps noisy) signal and receive out a demodulated stream of data.

(6) The *Rational Resampler* again adjusts the sample rate.

(7) The *Audio Sink* block outputs the data to the computer speakers. In the case of 95bFM, this just outputs the commercial radio as one would hear from one's car, for example.

1.2.3 Solving the two-body problem

Appendix A of (Richards 2019) discusses the mathematical foundation of the two-body problem. This is solved numerically by the Simplified Perturbation Model (SGP) algorithm (specifically the SGP4 algorithm), proposed in (Hoots & Roehrich 1988) and revisited in (Vallado, Crawford, Hujsak & Kelso

Figure 1.3: A simple dipole antenna with an RTL-SDR (*RTL-SDR Blog V.3. Dongles User Guide* n.d.) to achieve demodulation. The antenna arms on the tripod are UHF, whereas the detatched arms to the right are VHF. The RTL-SDR is a USB dongle; it may be plugged in to any computer with capability to run the associated drivers, and used as if it were a conventional radio.

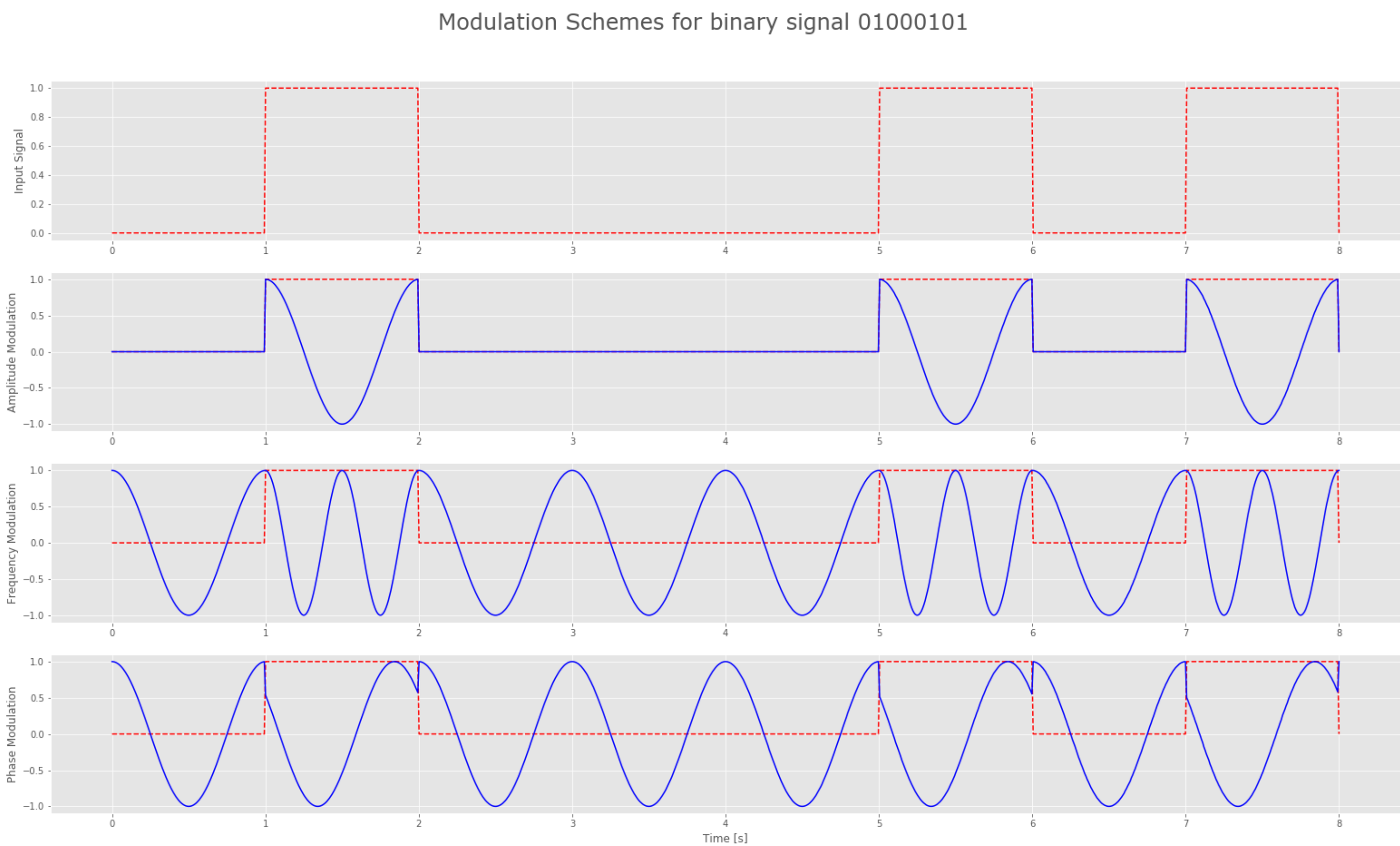


Figure 1.4: Sample Modulation Schematics for Ampltude Modulation (Eq. 1.2), Frequency Modulation (Eq. 1.3), and Phase Modulation (Eq. 1.4).

2006). The SGP4 algorithm encapsulates both gravitational effects and atmospheric drag, leading to a remarkably accurate computation. Indeed, the implementation we use from (satellogic 2019) within the gpredict engine computes the path of satellites to an accuracy of 1×10^{-9}.

Chapter 2

Literature Review

The seminal paper in the field of satellite tracking is “Spacetrack Report No. 3” (Hoots & Roehrich 1988) which laid the foundation for the Two Line Element set (TLE) data format. This was revisited in (Vallado et al. 2006) which updated the field in 2006 with modern code and analysis. Both reports present algorithms to compute solutions to the two body problem of satellite tracking, given some known TLE files.

In terms of institutional ground stations, we have (Shamutally, Soreefan, Suddhoo & Momplé 2018) and (Fischer & Scholtz 2010). The former illustrates how a ground station may be built using simple “off-the-shelf” components for relatively low cost. Conversely, the latter establishes the tracking system employed by the University of Vienna. Both provide inspiration for the work presented in this report.

In particular, (Shamutally et al. 2018) casts their Low-Earth Orbit (LEO) ground station in the context of supporting MIR-SAT1, which is the first spacecraft launched by the nation of Mauritius. They aim to engage students with their missions, to enable any interested student to listen in on the satellite communications, and discover the world of satellite tracking.

On the other hand is (Fischer & Scholtz 2010). The ground station at the University of Vienna is designed such that it may serve as the primary method of communication with their satellite, the BRITE-Constellation, and to be integrated into the Global Educational Network for Satellite Operations (GENSO). GENSO can be thought of as a crowdsourced network of ground stations, which may be leveraged by an individual or institution, to enable a greater coverage and communication time for individual satellites.

A key and recurring theme throughout the papers in the field, is that of separation of technology. For instance, in the Virginia Tech ground station (Hitefield, Leffke, Fowler & McGwier 2016), there is a clear deliniation between elements of the station. It is divided into four sections:

(1) The **Red** Network is the untrusted interface. This is primarily consisted of the public-facing elements of the ground station such as the end-user interface to submit jobs.

(2) The **Green** Network is the local, trusted network within the ground station complex.

(3) The **Blue** Network serves as an additional trusted segment for typically wireless connections within the complex.

(4) The **Orange** Network is the “Demilitarized Zone” (DMZ), which functions as an security layer from the red network. It is isolated from the trusted segments, and data travelling through the DMZ terminates in a secure web server.

Similarly, and in a more software-based approach, (Nel & Barry 2003) abstracts the ground station at the University of Stellenbosch in South Africa into different layers, permitting only essential communication between individual layers. This serves to isolate individual systems, to allow them to be swapped in and out as necessary, and provide an element of cybersecurity to the ground station. This is an important thing to consider for any ground station.

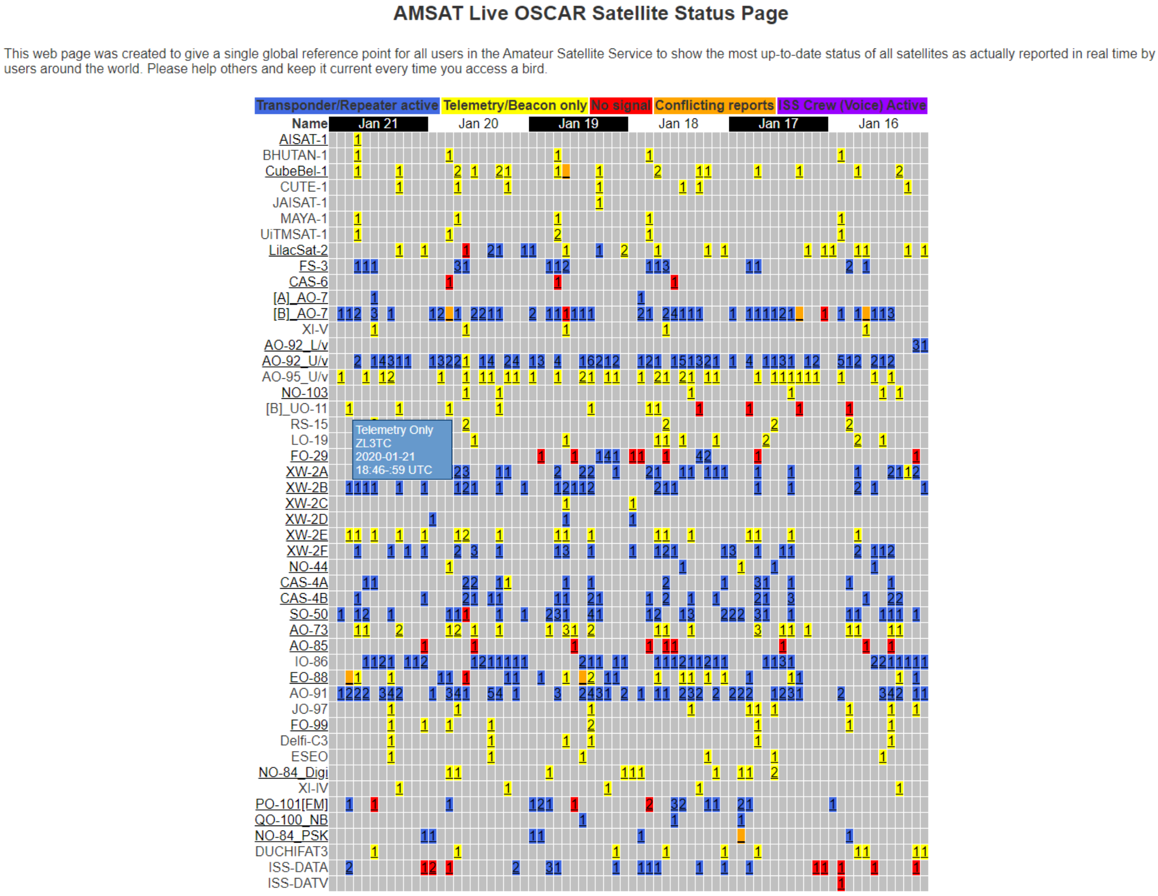


Figure 2.1: The AMSAT OSCAR tracking matrix. Each entry in the matrix shows attempts at tracking a given satellite, with the colour code representing the outcome. For instance, in this image the leftmost entry for [B]_UO-11 is highlighted, depicting operator ZL3TC attempting to track the satellite on 21 January 2020 at 18:46 UTC, and receiving only telemetry.

As well as academic usage, the advent of inexpensive ground stations running on open-source hardware/-software stacks has caused a hobbyist culture to emerge. This is primarily manifested in the Amateur Radio in Space cooperative (based in the United States), which provides services to the community world-wide. Most notably, they provide the AMSAT Orbiting Satellite Carrying Amateur Radio (OSCAR) tracking matrix (Radio Amateur Satellite Corporation 2020) as shown in Fig. 2.1.
Moreover, individual operators such as (Rupprecht 2020) compile and disseminate various pieces of useful information on individual crafts. A sample of Rupprecht's information is shown in Fig. 2.2. The full dataset is extensive, though does not cover every satellite in the sky.

Status of active Satellites on Amateur Radio Frequencies

last update: Jan 06, 2020

Satellite	Status	NORAD	Uplink	Downlink	Beacon	Mode	Callsign	Reports	Info	IARU freq coord	Telemetry Decoder
FLORIPASAT-1	ACTIVE	44885	.	.	145.900	1k2 FSK	PY0EFS	latest report	details	YES	.
CAS-6	INACTIVE	tbd	435.280	145.925	145.890 145.910	4k8 GMSK CW	BJ1SO	latest report	details	YES	.
OPS-SAT	ACTIVE	44878	.	.	437.200	9k6 GMSK	DP0OPS	latest report	details	YES	
DUCHIFAT-3	ACTIVE	44854	145.970	436.400	436.400	V/U FM, 9k6 BPSK	4X4HSL-1	latest report	.	YES	
ATL-1	ACTIVE	44830	.	.	437.175	12.5kbps GMSK	.	latest report	details	YES	.
SMOG-P	ACTIVE	44832	.	.	437.150	12.5kbps GMSK	.	latest report	details	YES	.
ACRUX-1	ACTIVE	44369	.		437.200, 437.425	9k6 GMSK		latest report	details	YES	
FOSSASAT-1	INACTIVE	44829	.		436.700	RTTY 45bd		latest report	details	YES	
TRSI	INACTIVE	44831	.		437.075	GMSK		latest report	details	YES	
LUOJIA-1 01	ACTIVE	43485	.		437.250	4k8 GMSK		latest report	details	NO	
JAISAT	ACTIVE	44419	.		435.700	4k8 GMSK		latest report	details	YES	
SOKRAT	ACTIVE	44404	.		436.000	4k8 GMSK Mobitex		latest report	details	NO	
AMGU-1	ACTIVE	44394	.		436.250	4k8 GMSK Mobitex		latest report	details	NO	
EXOCONNECT	INACTIVE	44413	.		435.700	4k8 GMSK Mobitex		latest report	details	NO	
LIGHTSAT	ACTIVE	44393	.		435.700	4k8 GMSK Mobitex		latest report	details	NO	
LIGHTSAIL-2	ACTIVE	44420	.		437.025	9k6 FSK		latest report	details	YES	
NO-103 (BRICSAT2)	ACTIVE	44355	145.825	145.825	437.605	1k2 AFSK 9k6 FSK		latest report	details	YES	
NO-104 (PSAT2)	ACTIVE	44354	145.825	145.825	145.825	1k2 AFSK		latest report	details	YES	
KRAKSAT	ACTIVE	44427	.		435.500	1k2 AFSK	SR9KRA-6	latest report	details	YES	
SWIATOWID	ACTIVE	44426	.		435.500	1k2 AFSK	SR6SAT-6	latest report	details	YES	
LUCKY-7	ACTIVE	44406	.		437.525	4k8 GFSK	OK0SAT	latest report	details	YES	
ARMADILLO	ACTIVE	44352	.		437.525	19k2 FSK	WH2XYR	latest report	details	YES	
UGUISU	ACTIVE	44330	.		437.375	CW, 4k8 FSK	JG6YLE	latest report	details	YES	
NEPALISAT-1	ACTIVE	44331	.		437.375	CW, 4k8 FSK	JG6YLF	latest report	details	YES	

Figure 2.2: Spacecraft information sample as compiled by Mike Rupprecht (DK3WN). As an example data point, the satellite KRAKSAT can be seen to have NORAD ID 44427, callsign SR9KRA-6, a beacon frequency of 435.5 MHz, and data modulation type of Audio Frequency Shift Keying (AFSK – roughly equivalent to traditional FM) at a baudrate of 1200Bd.

Chapter 3

Technical Overview

A schematic of the hardware of the ground station is presented in Fig. 3.1 on the next page. The system can broadly be cast into three main categories as illustrated in that schematic:

- RF Backend,
- RF Frontend, and
- Control Systems.

This division enables flexibility and allows for individual components to be exchanged as required.
We examine individual components of the systems below.

3.1 RF Backend

The RF backend comprises two antennae: a Yagi Y6U UHF antenna (Hi-Tec Aerials 2013) and a 2MCP14 VHF antenna (M2Inc 2019). The UHF antenna was installed in 2017 under the work of Anand Thirumulai (Thirumalai 2017). The VHF antenna was installed on the 14 January 2020 and 17 January 2020, allowing us the ability to downlink from a wider range of satellites, and uplink to APSS-1 if required.[1]

3.2 RF Frontend

The RF frontend comprises a Diamond MX72A (Diamond Antennas 2019) to achieve multiplexing of the antenna signals, which feeds into an LNA-580 low noise amplifier (EverythingRF n.d.) and then an Airspy R2 SDR (*Airspy R2* 2019). The SDR acts as a peripherial component to a Raspberry Pi 3B (Raspberry Pi Foundation 2019), which is the foundation of the entire software.
The critical problem here was powering the LNA. Prior to the upgrade this was achieved using a bench power supply, however this has limitations insofar as it requires an extra mains outlet; space can be a concern in ground station setups. The bench supply was also noisy and occaisionally did not output the full 5V we expected. Consequently when designing the cables to control the rotator controller we elected to use a 6-13V power supply available on the Yaesu G5500 Rotator (Yaesu 2019) we employ. This is passed through a voltage regulator (shown in Fig. 3.3 on page 15) to effectively step it down to 5V.

3.3 Control Systems

The control systems comprise a Yaesu G5500 Rotator (Yaesu 2019), an Arduino Uno rev3 (*Arduino Uno rev3* n.d.), and a Raspberry Pi 3B (Raspberry Pi Foundation 2019).
The arduino is powered via USB cable from the Raspberry Pi, which is in turn powered from mains power. We found we could not power it using the same method as the LNA, as the Yaesu rotator can only supply 200 mA of current, whereas the Raspberry Pi can demand up to 2A which, without serious

[1] Only in the event of loss of communication with our commercial partners.

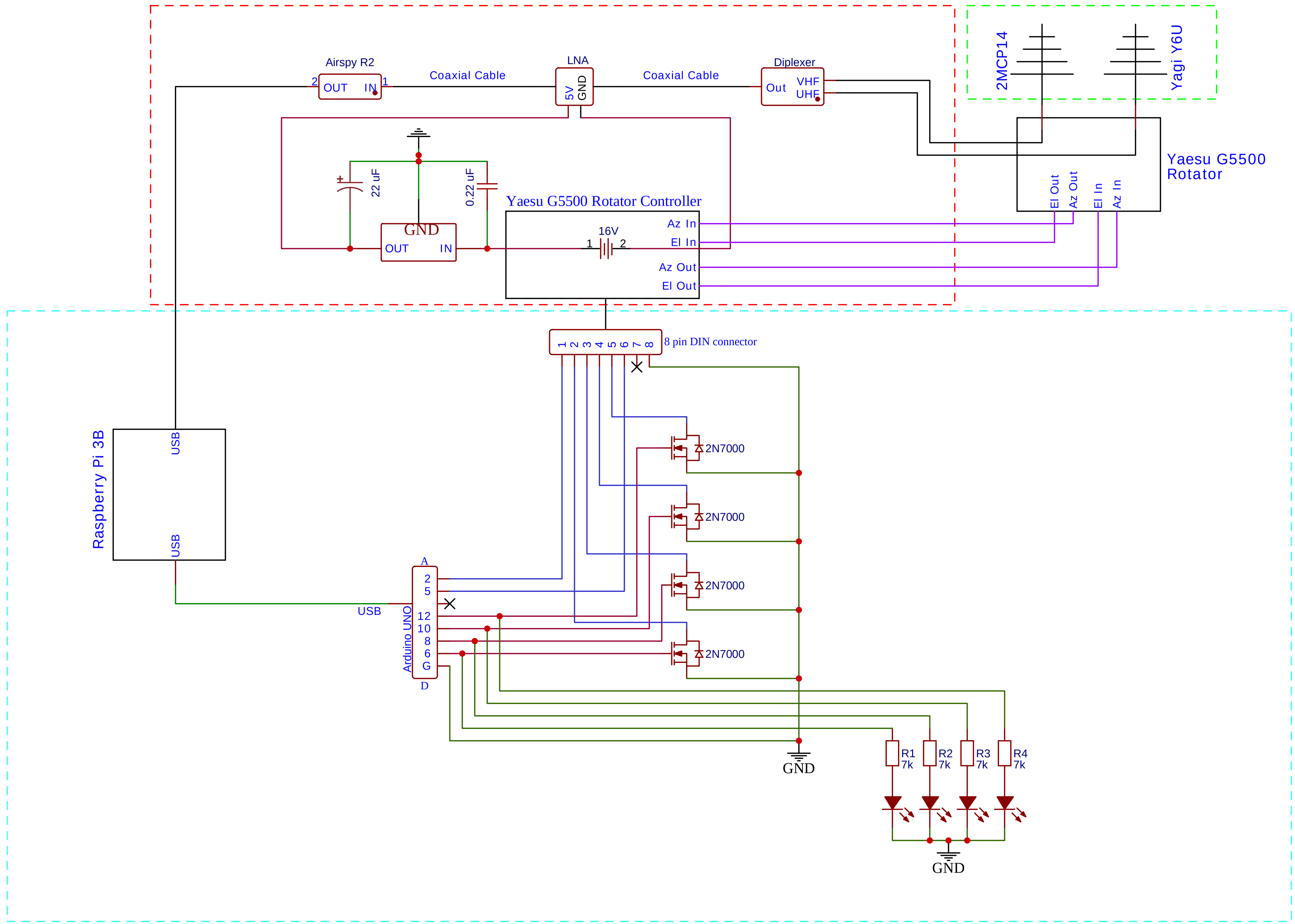


Figure 3.1: A schematic of the University ground station. The dashed boxes represent the different categories discussed at the start of Chapter 3 — red denotes the RF backend, green is the RF frontend, and cyan denotes the control systems. The Yaesu G5500 rotator belongs to the control systems group.

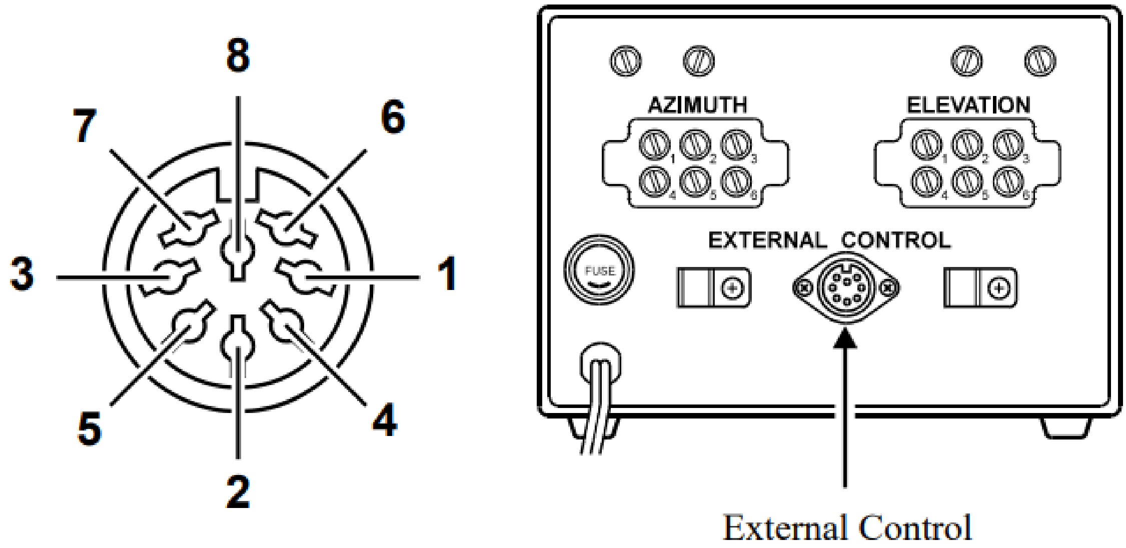


Figure 3.2: Pinout diagram of the Yaesu G5500 rotator external control DIN pin. Image from (Yaesu 2019), pin functionality explained in Table 3.1.

Pin Number (from pinout)	Arduino Pin	Functionality
8	NC	Common ground
6	A5	Azimuth readout (between 2-4.5V DC)
1	A2	Elevation readout (between 2-4.5V DC)
4	D10	CCW rotation
2	D6	CW rotation
5	D12	Rotate down
3	D8	Rotate up
7	Voltage Regulator	6-13V DC supply ($I \leq 200$ mA)

Table 3.1: Yaesu G5500 DIN pin configuration.

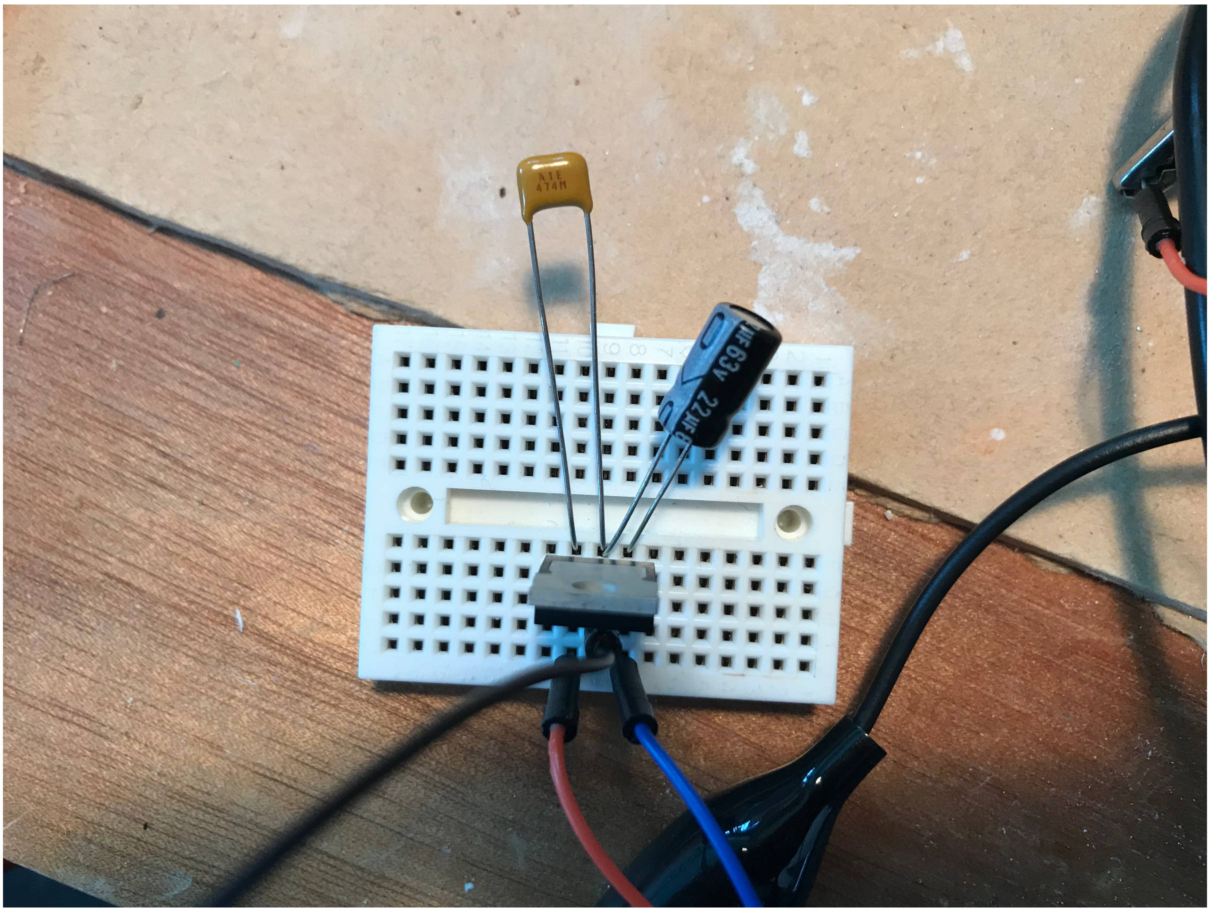


Figure 3.3: The 5V voltage regulator employed in the ground station. The red wire is input, blue is output, and black is common ground. The capacitances of the capacitors are 0.22 μF (ceramic, left) and 22 μF (electrolytic, right).

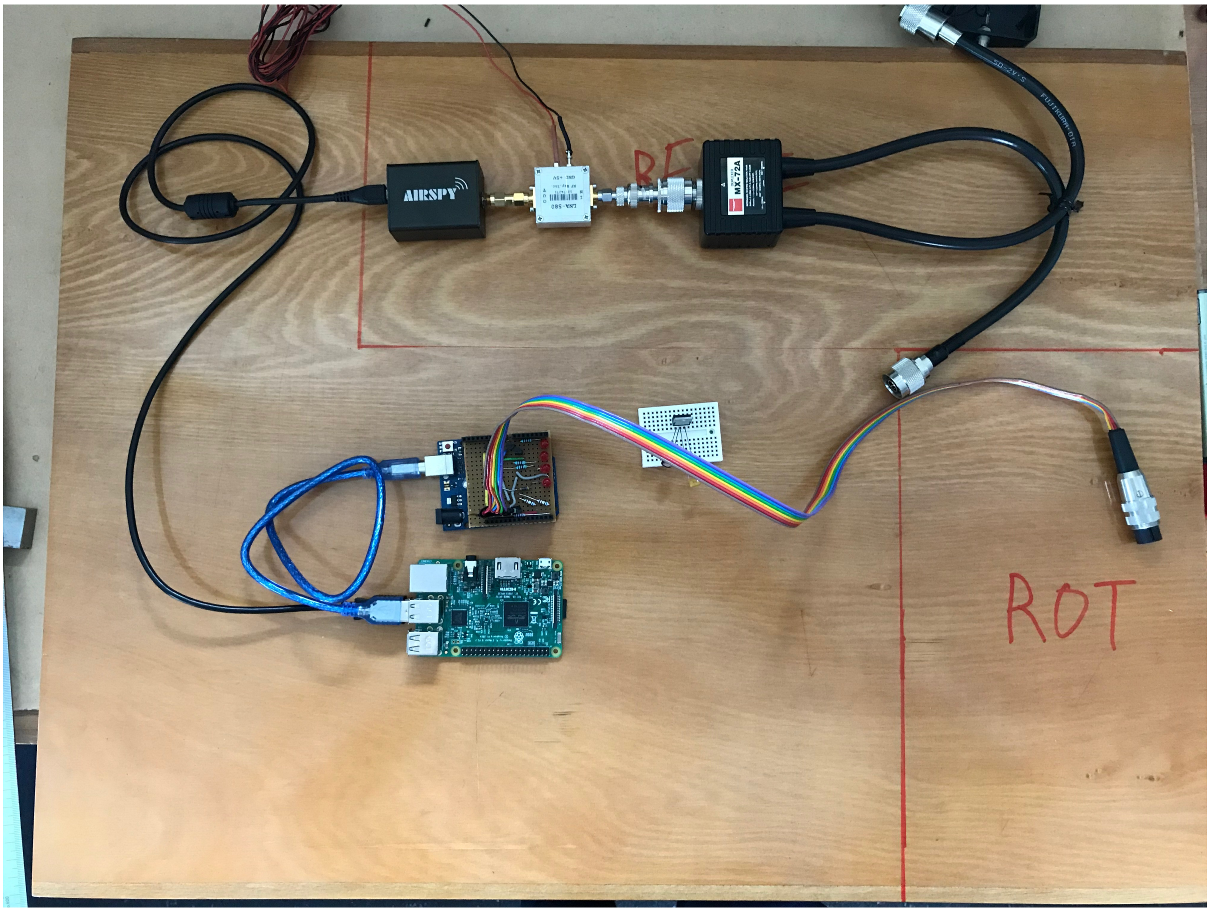


Figure 3.4: The beginnings of laying out the upgraded ground station

reconfiguration of the ground station, is achievable only with mains power. The Yaesu is also powered via mains power.

The Raspberry Pi is configured to permit both SSH access and has TightVNC (GlavSoft LLC n.d.) installed in order to facilitate remote access, depending on the user's requirements.

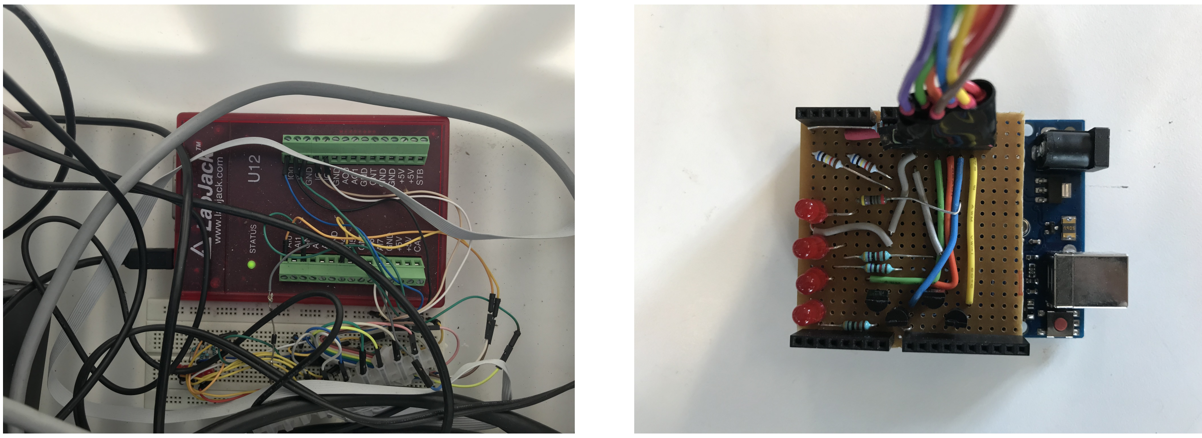
The main aim of the control systems upgrade was to utilise open-source software wherever possible. Thus, the raspberry pi runs the Raspbian operating system and the Arduino is flashed with a modified version of (Good n.d.). This in tandem allows us to achieve a small, open-source, and easily modifiable control system. Side-by-side shots of the old and new rotator controllers are shown in Fig. 3.5.

The rotator controller utilises four 2N7000 MOSFETs to act as digital switches. The wiring diagram for the digital switches is shown in Fig. 3.6

3.4 Analysis Machine to Ground Station connection

One of the major challenges presented in (Richards 2019) was the connection between the local analysis machine and the ground station. It was identified as being of critical importance, however the link had seemed to be broken due to bugs in the gr-osmocom patch used. This issue was rectified by employing the SoapySDR (pothosware n.d.) library to act as a TCP server between the ground station and analysis machine.

Both the Raspberry Pi and the analysis machine run the libraries SoapySDR and SoapyRemote (ibid). Additionally, the Raspberry Pi runs SoapyAirspy (ibid). The net effect is that the analysis machine sees the Airspy as if it was a local peripheral device, and GQRX naïvely loads it and interacts with it



(a) The LabJack U12 rotator circuit used before the upgrade.

(b) The Arduino Uno rotator circuit used after the upgrade. It is less than $\frac{1}{3}$ of the size of the LabJack.

Figure 3.5: The rotator controller circuits before and after the upgrade.

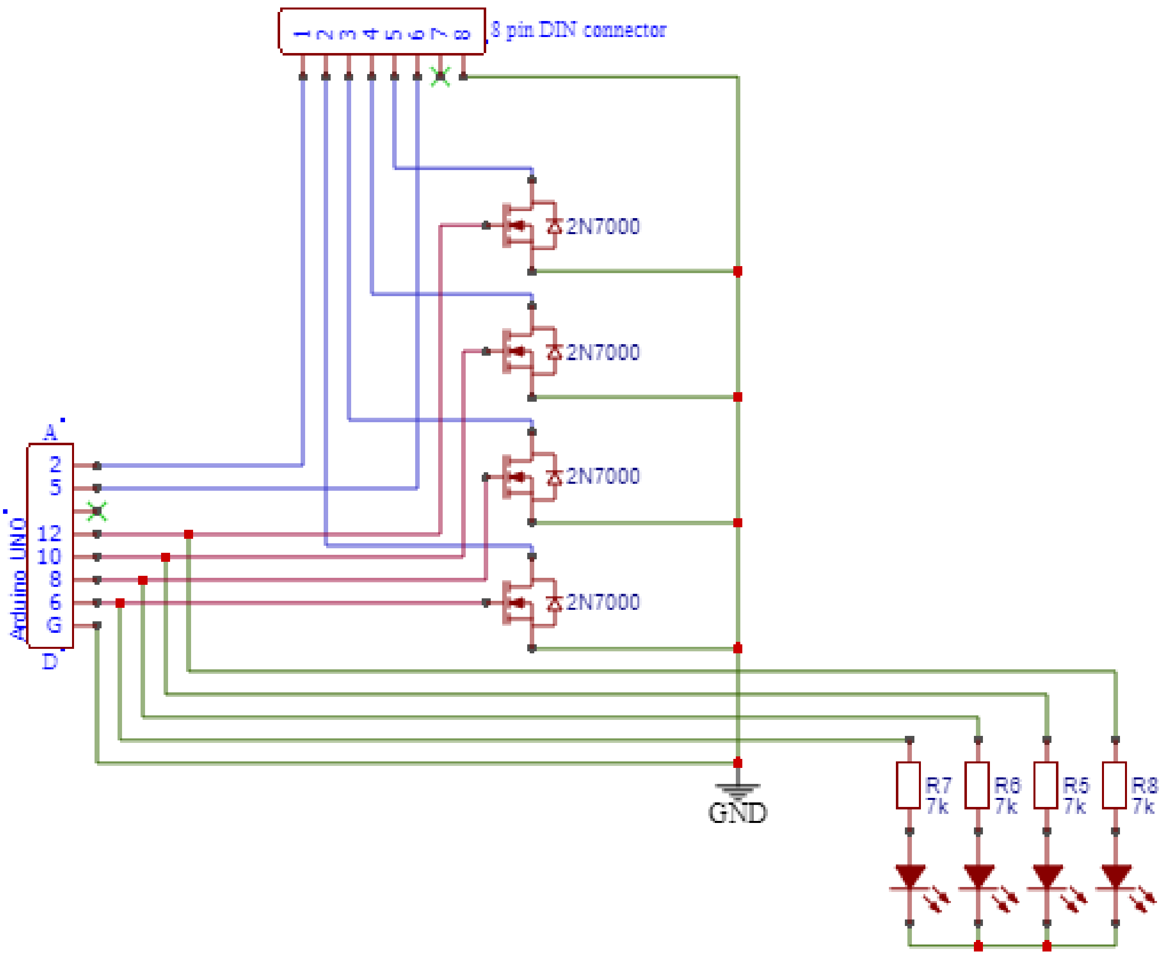


Figure 3.6: The MOSFET-based digital switches that realise rotation control of the ground station.

as such. Consequently we may avoid the issue of trying to manually communicate via TCP – an early testing version of the ground station had a GNU Radio flowgraph which took input from the SDR and immediately output the same signal on port 4242 via TCP. Whilst this would work in principle (and would allow us to offload demodulation and processing to the Raspberry Pi), we would not be able to have the GQRX to GPredict synchronisation, which is of equal importance to the functionality of the ground station.

3.5 The VHF antenna

On 14 January 2020 and 17 January 2020, a team of researchers and students[2] installed the VHF antenna. The work plan is included as Appendix A, and the process of installing the antenna is shown in Fig. 3.7. The reason the antenna was installed over two days is due to adverse weather conditions. After installing the new steel boom arm, we noticed an onset of rain. As a precaution we paused the upgrade to determine if the rain was passing showers or indicative of heavy rain – the latter occurred, and as such we aborted the upgrade and reconvened on the second day.
The new array sports a steel boom arm to support the heavier load. The total weight acting on the boom arm is 3.6kg on the UHF end, and approx 2.7kg on the VHF end. As we were able to approximate the UHF/counterweight end as a point mass, to balance the torques due to the weights of the antennae, we required

$$\tau_{\text{UHF}} = 3.6gd \tag{3.1}$$

$$\tau_{\text{VHF}} = 2.7 \times g \times 67.4 \tag{3.2}$$

$$\begin{aligned} \tau_{\text{UHF}} = \tau_{\text{VHF}} &\implies 3.6gd = 181.98g \\ &\implies d \simeq 50.4 \text{ cm} \end{aligned}$$

[2]Namely, Dr Nicholas (Nick) Rattenbury, Mr Felicien Filleul, Mr Nico Reichenbach, Ms Antonella Caldarelli and Mr Sean Richards

(a) Installation of the VHF antenna onto the steel boom arm.

(b) Raising the mast and reinstalling the UHF antenna.

(c) The finalised antenna array.

(d) Nico Reichenbach reaching for a tool.

Figure 3.7: The installation process of the VHF antenna.

Chapter 4

Operations

The work of the ground station operator is to identify, isolate, and demodulate signals. This is accomplished by a software stack on the main analysis computer, consisting of gpredict (Csete 2008) and GQRX (Csete 2010). The workflow the operator must undertake is outlined in the following sections.

4.1 Identification of viable passes

The operator first must use gpredict to identify a viable pass. What makes a given pass viable depends on a few factors and is, in the most general sense, an art rather than a hard science.

For any pass to be viable, it must rise sufficiently far above the horizon. Because of the layout of the ground station, this depends on the quadrants the satellite passes through. Any pass in the northernmost quadrants rising above 10° or so is sufficient, however to the southeast the ground station is bounded by tall metal spires and the Chemistry building. Thus the minimum threshold for a pass there rises to around 25°. As a general rule of thumb, any pass in any quadrant with a maximum elevation of above 30° will be sufficient.

The operator must also be aware of the different sources of noise that ground station is subject to. Most notably the ground station is bounded to the northwest by downtown Auckland. This presents the challenge of hearing signals from car fobs which[1] transmit at 433MHz, in the amateur radio band. This appears in the SDR viewer as a stationary source, adding noise and potentially obscuring other, more relevant, signals. The ground station is located on Level 8 of the Science building; consequently these signals will generally be weak in intensity however they may still appear.

The satellite CAS-4A shown in the polar plot in Fig, 4.1 (in yellow) is not viable. The satellite barely crosses the horizon and does so in the northeastern quadrant which, although mostly devoid of obstructions, terminates with the metal spires.

4.2 Operating the ground station

The raspberry pi has a dynamic IP address; consequently it will periodically change. This typically occurs between reboots of the ground station. If the operator carries out steps such as commanding the rotator to move and it does not, the first thing the operator should do is confirm whether or not this has occured, by way of the `nslookup` command. Specifically, by using `nslookup raspberrypi.uoa.auckland.ac.nz` command, one will see output similar to Fig. 4.2.

Whilst the primary workhouse of the ground station operator is GQRX, actual radio and rotator control is realised in gpredict, as shown in Fig. 4.3a and Fig. 4.3b. Firstly, radio control is where the operator selects the satellite transponder they wish to listen to, in the red box. They engage the gpredict to GQRX throughput by pressing "Engage" in the blue box (ensuring the device for downlink is set to "GQRX", and then wait for the AOS counter in green to reach zero; this corresponds to the satellite crossing the horizon.

[1] for cars manufactured in Asia or Europe

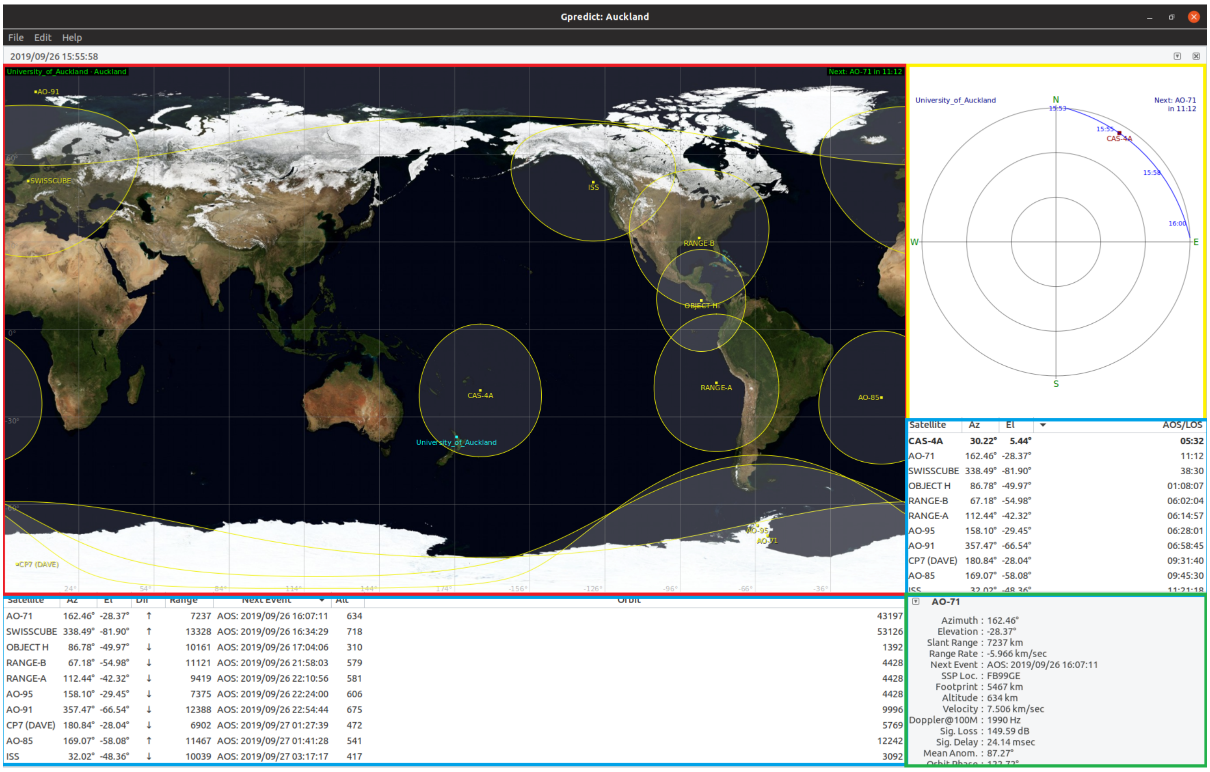
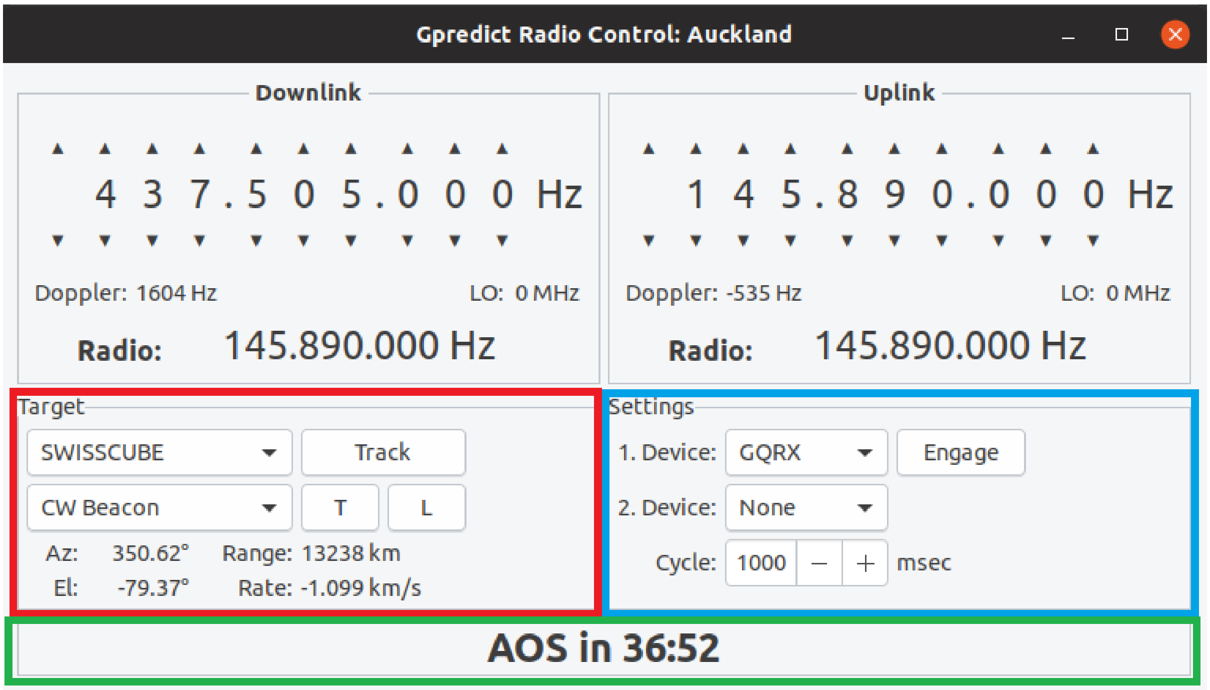
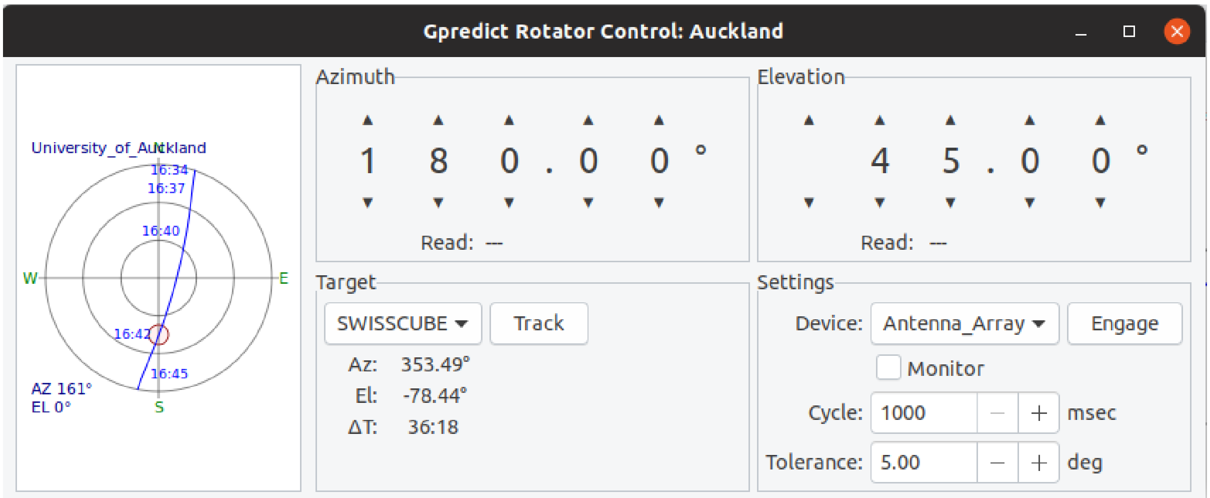


Figure 4.1: The home screen of gpredict. The red box is a map view of the earth, centered at 180° longitude such that New Zealand is roughly in the center. Outlined in yellow is a polar plot of the current sky; the origin represents zenith, and spacecraft are shown as red dots moving along blue "sky tracks". Outlined in blue (repeated twice) is a list of all satellites currently being tracked along with their current azimuth and elevation (as measured relative to the ground station), slant direction, range, time of next event (Acquisition of Signal (AOS)/Loss of Signal (LOS)), altitude, and orbit number. Finally outlined in green is the "single-sat view", a breakdown of the orbital parameters and engineering design of the satellite.

Figure 4.2: The output of looking up the ground station IP address. The *Address* field underneath the raspberrypi hostname is the IP address of the ground station.



(a) Radio Control



(b) Rotator Control

Figure 4.3: The two primary control windows in gpredict.

Once the satellite is over the horizon, the operator is free to begin the process of identifying signals within GQRX. For the purposes of an example, Fig. 4.4 shows GQRX listening to the campus radio station 95bFM. This is a stationary, broadband source – in the instance of a satellite there will be some shift in the frequencies due to the Doppler effect. The transmission amplitudes are also generally much smaller.
The four major control areas in GQRX are the top bar and the three tabs in the right tab pane. The latter is shown in Fig. 4.5 (though only the Receiver Options menu is particularly interesting), and the former in Fig. 4.6. The top of the top bar shows the version of GQRX and the device string. The anatomy of the string is as follows:

(1) device_id - the internal device ID in GQRX for the connection type. Set to 0 for a remote connection.

(2) driver - the name of the driver. Set to "remote" for a remote connection.

(3) remote - the IP address and port of the SDR on the raspberry pi.

(4) remote:driver - the driver of the SDR on the *remote* ground station.

(5) serial - the serial number of the SDR

(6) soapy - internal to SoapySDR, default is 2.

Thus, the complete device string for our usage is (with the newline omitted)

device_id=0,driver=remote,remote=tcp://130.216.50.29:55132,
remote:driver=airspy,serial=440464c8:3859384f,soapy=2

The three important controls are the play button (first in list), the computer chip (second in list) and the two connected computers (seventh in list). The functions are as follows:

(1) The **Play button** begins GQRX's reception functionality. It must be enabled for the ground station to function.

(2) The **Computer chip** allows the operator to adjust the device string, to adjust critical parameters such as the sample rate, and to adjust preferences.

(3) The **connected computers** activates the GQRX/gpredict throughput. This allows gpredict and GQRX to communicate the doppler corrections, in order for GQRX to correct for the Doppler effect.

4.3 Packet Decoding

In the special case of AFSK modulation, at 1200baud, AX.25 packets can be decoded directly in GQRX. However, in the words of Alexandru Csete (Csete 2017):

> In fact, the AFSK1200 decoder was never intended to be anything else than an experiment and should never have been included in gqrx. It's not even a very good afsk decoder from a performance point of view.

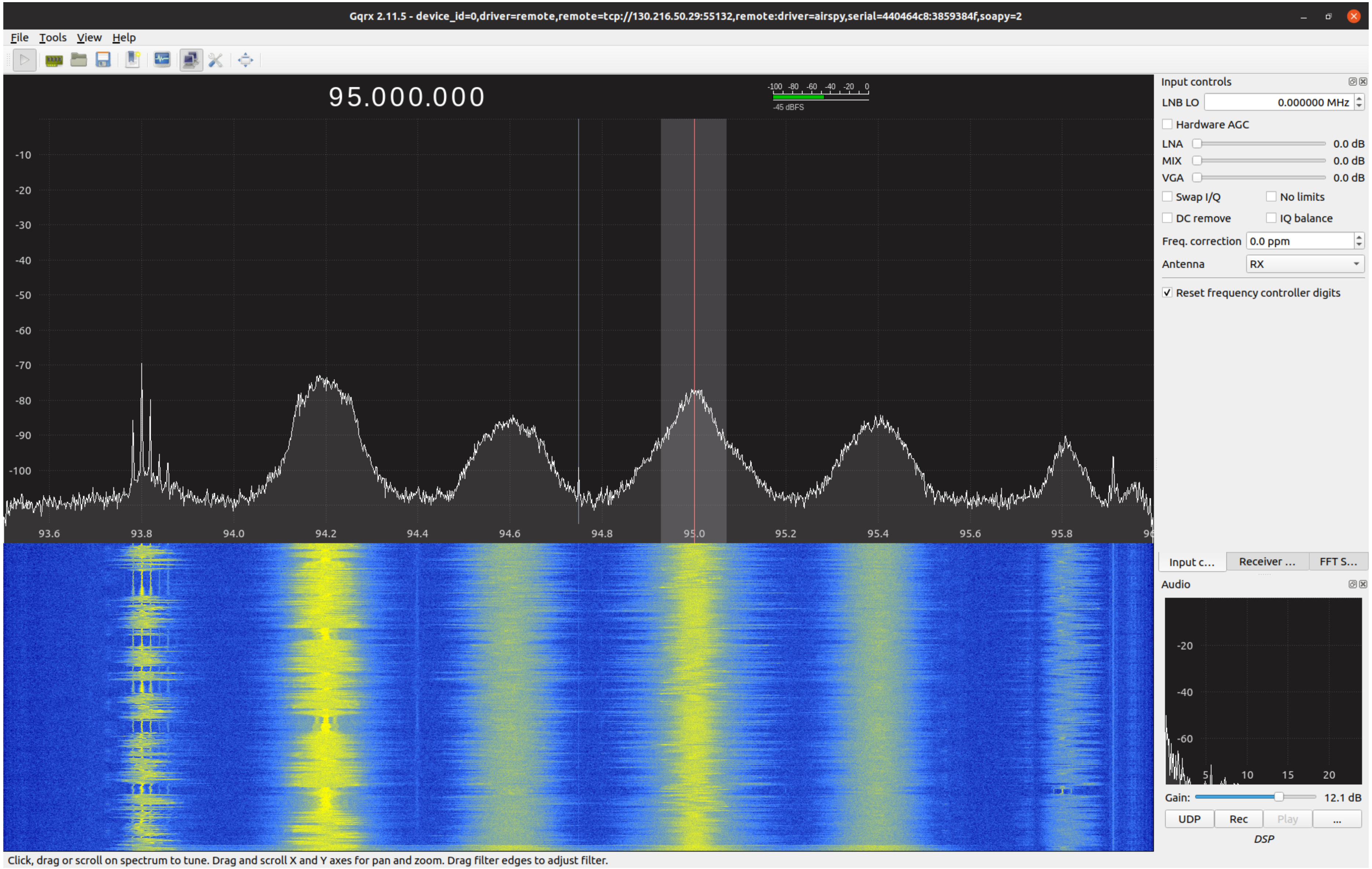
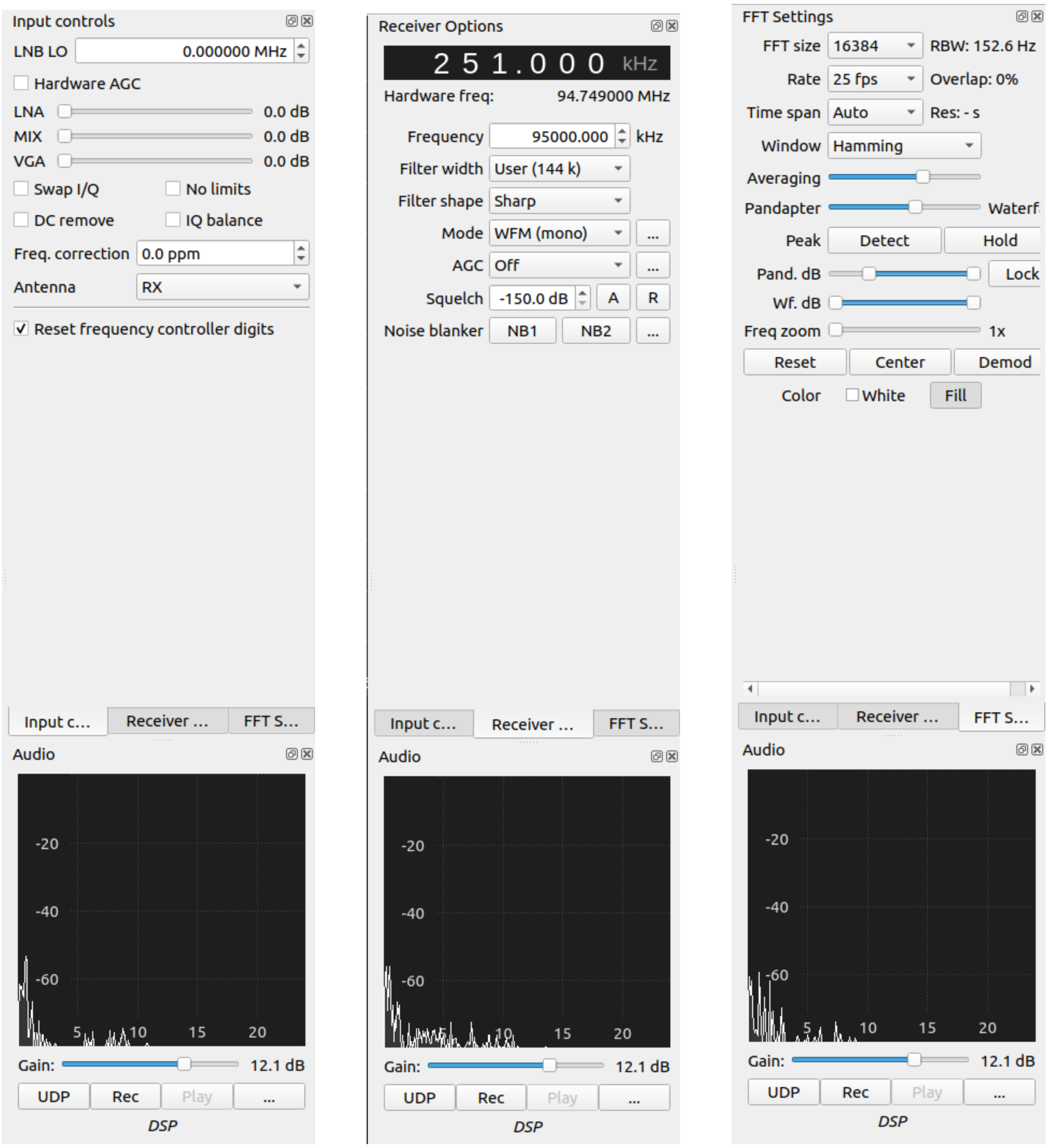


Figure 4.4: GQRX tuned in to 95bFM.



(a) The *Input Controls* settings.

(b) The *Receiver Options* settings.

(c) The *FFT Settings*.

Figure 4.5: The three major control windows in GQRX. Of particular interest – as it is the only one the operator will commonly be encountering – is the *Receiver Options* menu. This is where, in the *Mode* option, the operator may select a demodulation type such as wide FM, enabling the operator to identify by ear the correct modulation. In the case of commercial radio, this also allows the operator to listen in to it.

Figure 4.6: The Top Bar of GQRX.

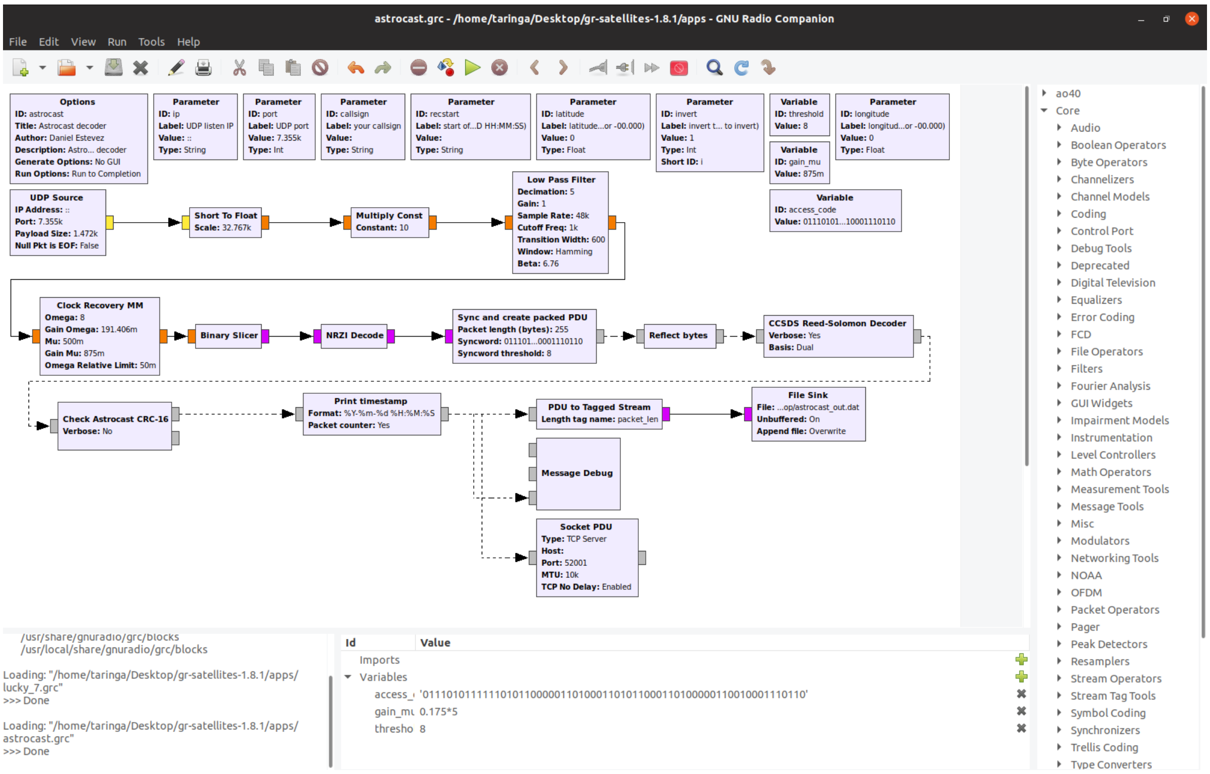


Figure 4.7: A flowgraph designed to decode a signal from the ASTROCAST satellite.

For this reason, GQRX does not support any other decoders than this one special case. Consequently, the simplest way to decode the packets outside of GQRX is to use a the command line utility DireWolf (Langner 2015). By configuring GQRX to output data via UDP on port 7355, we are able to interface direwolf with GQRX and thus decode packets using relatively few pieces of software.
The limitation of direwolf is that is restricted to very few modulation types. It is suitable, for example, of descoding AFSK1200 packets. At higher baudrates, direwolf automatically chooses other modulation types. In such cases, GNURadio is used to decode more modulation types. For instance, the flowgraph in Fig. 4.7 implements a flowgraph[2] to decode the telemetry from ASTROCAST. It produces the output shown in Fig. 4.8. Of importance, this flowgraph pipes its output to a file (entitled astrocast_out.dat) and to a TCP socket on port 52001. Thus the decoded data can be streamed to any interested party within the University firewall, and also saved for postprocessing after the pass.

4.4 Summary of process of operation

From booting the ground station analysis machine, the operator must undertake the following methodology to begin the tracking:

(1) Open gpredict's **Rotator Control** and **Radio Control** as shown in Fig. 4.3.

- **(a)** In **Radio Control**, ensure the target is set correctly (including the transponder), click Track, Tune. In the settings pane, ensure the first device is set to GQRX, and click Engage.
- **(b)** In the **Rotator Control** window, ensure the "Target" and "Device" options are set correctly, then click Track and Engage. The "Tolerance" can be kept as is or set to $\simeq 3°$ at the lowest. At this point the antenna will begin moving.

[2]Originally written by (Esteves n.d.), modified by Sean Richards

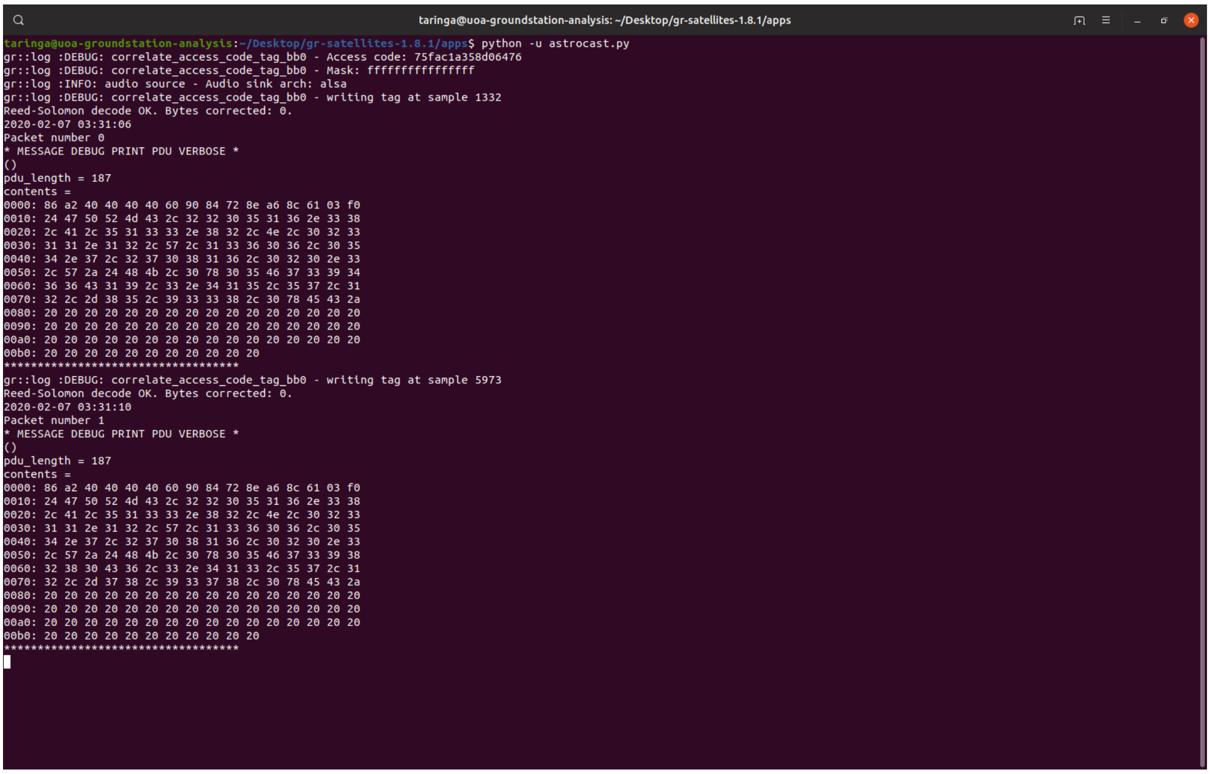


Figure 4.8: A packet transmitted by ASTROCAST. This particular packet was decoded from a recording provided by (Esteves n.d.), as attempts to track the ASTROCAST satellite via the ground station proved unreliable, as it did not transmit when we listened. The salient part of the packet is the `contents` line, which is a hex encoding of the packet.

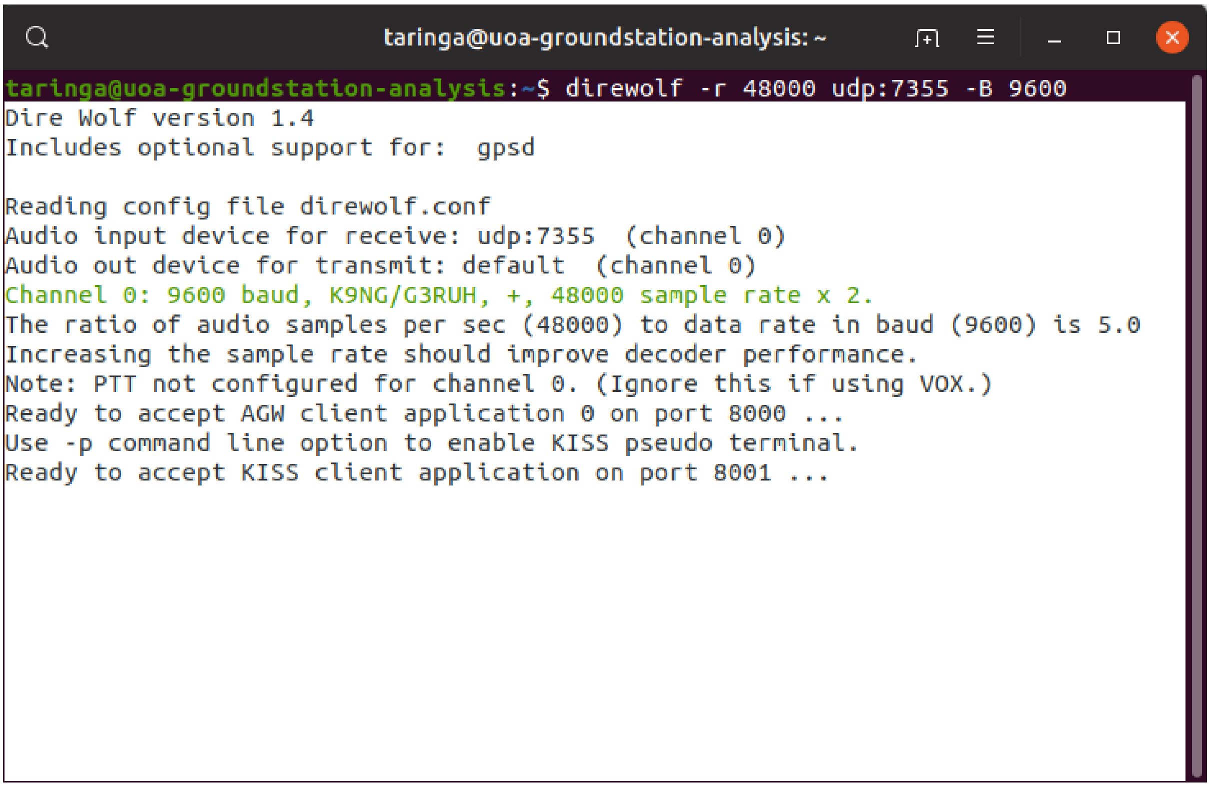


Figure 4.9: The home screen of the direwolf utility. The syntax of the command as written is: `direwolf -r <sample rate> udp:<udp port> -B <signal baudrate>`.

(2) In gqrx, activate the throughput by pressing the **connected computers** button, and activate the connection by pressing the **play** button.

- **(a)** Ensure the “Mode” option in the **Receiver Options** settings (see Fig. 4.5b) are set to the correct modulation schema. The mode is listed with the satellite parameters in gpredict or, if necessary, on (Rupprecht 2020).

(3) Decode the signal by implementing a GNURadio flowgraph or by using direwolf to decode the packets.

Chapter 5

Conclusion

We presented a detailed overview of *Taringa*, the University of Auckland's satellite ground station. Although this is not designed to function as the main ground station for APSS-I, being currently incapable of uplink, we demonstrate suitable functionality of the station to serve as a downlink monitoring station, though true downlink will occur at Awarua Satellite Ground Station.

5.1 Suggested Future Work

The current ground station, while modular and easy to adjust, suffers from being located outside. During summer, the temperature inside the box plus the temperature outside can make it difficult to maintain from an Operations Management perspective. Moreover the ground station becomes completely inaccessible in wet weather. Thus, future operators should consider moving the analysis components of the ground station inside, retaining essentially only the antennae and rotator outside.

Moreover, the fan inside the ground station has the unfortunate side effect of inducing artefacts into the SDR. These appear as periodic and regularly spaced pulses in the FFT display in GQRX, and have the effect of drowning out low-power signals. Consequently a better solution will need to be employed in order to regulate temperature within the ground station.

Finally, there is scope for full automation of pass tracking. GQRX and GPredict have the benefit of being able to accept TCP connections in order to facilitate remote tracking; consequently one may implement some form of scheduling algorithm in order to "queue up" passes and monitor any pass at any time of the day (or night, as the case may be).

Appendices

Appendix A

VHF antenna installation plan

The upgrade plan was signed prior to commencement of the upgrade by Dr Nicholas Rattenbury and Mr Sean Richards; the signatures are omitted for privacy reasons. The initial plan was to complete the upgrade on Monday, 16 December 2019 with a fallback day of Wednesday 18 – adverse weather conditions on both days meant this was not feasible, and consequently the upgrade was completed in mid-January 2020.

On the day, some aspects of the upgrade plan had to be modified. This was due to unforeseen circumstances surrounding the upgrade. For instance, we found that the steel pole had been polished to a higher standard than the aluminium pole we were replacing; consequently when the counterweight was installed we found it would rotate under its own weight on the low-friction steel arm. Consequently, we had to insert a weatherproof heatshrink layer between the boom arm and the counterweight to increase the friction between the two.

Personnel Identification

(1) Sean Richards
(2) Nicholas Rattenbury
(3) Felicien Filleul
(4) Nico Reichenbach

Upgrade Plan

Notes:

The “task description” column identifies which person should perform which task in each step. For instance, “(1) person to supervise, (2) person to move rotator controller, (3) & (4) person to mount antenna” means the person assigned to (1) in “Personnel Identification” above should supervise this step, the person assigned to (2) should move the rotator controller, and the people assigned to (3) and (4) should mount the antenna.

If an individual is *not* assigned a task, they are expected to observe and identify risks to Health & Safety as required.

Individuals are suggested to bring a large bottle of water and sunglasses.

Tasks in **red** identify hazardous steps. These steps must, where possible, have a supervisor present and a person ready to assist if required.

Upgrade Commences: Monday 16 December 2019, 1000 hours.

Personnel Required	Description	Task Description
DAYS BEFORE UPGRADE		
1	Weather clearance by 1700 hours previous day.	(1) to clear weather
1	Prepare tools as per “Tool List” section	(1) to complete
1	Assemble VHF antenna as per M2INC manual	(1) to complete
DAY OF UPGRADE		
ALL	Health & Safety Preparation	Apply sunscreen, hard-hats, required PPE

1	Movement of ground station box	(1) to move the ground station box away from work site
1	Command rotator to point due **EAST** or **WEST** (so UHF antenna is on far side from shack)	(1) to complete
1	Depower ground station	(1) to complete
2	Unbolt pole such that it may rotate down	(1) to unbolt pole, (2) to take weight
3	Pull pole down, and rest on the metal platform	(2) to walk pole down, (1) person to assist them in rotation and set-down, (3) to supervise and ensure safety of (2), (4) standing close ready to assist if required
2	Remove UHF antenna from boom arm	(4) to photograph orientation of antenna. (1) to unbolt, (3) to take weight of antenna
2	Remove counterweight from boom arm	(2) to unbolt, (4) to take weight of antenna
3	Remove boom arm from rotator (consult page 7 of Yaesu manual)	(1) to unbolt, (2) to supervise & confirm manual being followed, (3) to remove the boom arm with (1) to assist as required.
3	Install steel boom arm into rotator (consult pages 7 and 8 of Yaesu manual)	(1) to insert boom into slot, (4) to measure and ensure

		centering, (3) to bolt in
3	Install UHF antenna onto boom arm. **It must be oriented the same was as when removed.** Install at 400cm from center point.	(1) to measure distance, (2) to hold antenna in place, (3) to bolt in
3	Install counterweight onto boom arm on the **same side** as the UHF antenna. It must be placed at $d_{cw} \approx 1.06 d_{UHF}$ (approx. 424cm from center point). Exact position may vary when next step is carried out.	As above
3	Install VHF antenna onto boom arm. **It must be equidistant from the UHF antenna.** Maximum component width is 40.125in (approx. 101cm) so we should have plenty of room with a 12in clearance around antenna.	As above
4	Pull pole up	(1) and (2) to lift pole, (2) to walk it up, (1) and (2) to hold in place for following step
	Refasten pole	(3) to bolt in. (4) to be standing close by ready to assist if required.
POST DAY OF UPGRADE		
1	Plug UHF and VHF antennae coax cables into diplexer	(1) to complete
1	Testing (on-going).	Calibration of rotator controller if required. Testing VHF antenna points correctly.

Upgrade Completed: Monday 16 December 2019, 1200 hours.

Tool List

1. Measuring Tape
2. Phillips head screwdriver
3. Flat head screwdriver
4. Crescent spanner * 2
5. WD-40
6. Insulation tape
7. Scissors

Alternative Plans

There is a slight possibility of rain on Monday 16 December.

The last possible go/no-go time is Sunday 15 December 2019, 1900 hours. In the event of adverse weather forecasted, **Sean Richards** to notify all of a no-go via email by this time.

The alternative day for upgrade will be **Wednesday 18 December 2019** (same time). In the unlikely event of adverse weather this day, **Sean Richards** will advise by 1900 hours on Tuesday 17 December 2019, the upgrade will be postponed until the new year, and **Sean Richards** will advise of new go-time.

Health & Safety

Individual PPE must be worn at all times, a supervisor must be present when hazardous steps are to be taken.

In the event of a Health & Safety risk occurring, the upgrade will be paused whilst the issue is identified and dealt with.

Nicholas Rattenbury has final authority on any actions that have the potential to be a health & safety risk.

Signed & Approved

Sean Richards	
Nicholas Rattenbury	

Appendix B

Sample Flowgraph output

The following code snippet is the python code output of the flowgraph described by Fig. 1.2, which is repeated here as Fig. B.1. It may be optionally compiled into C/C++ if desired.

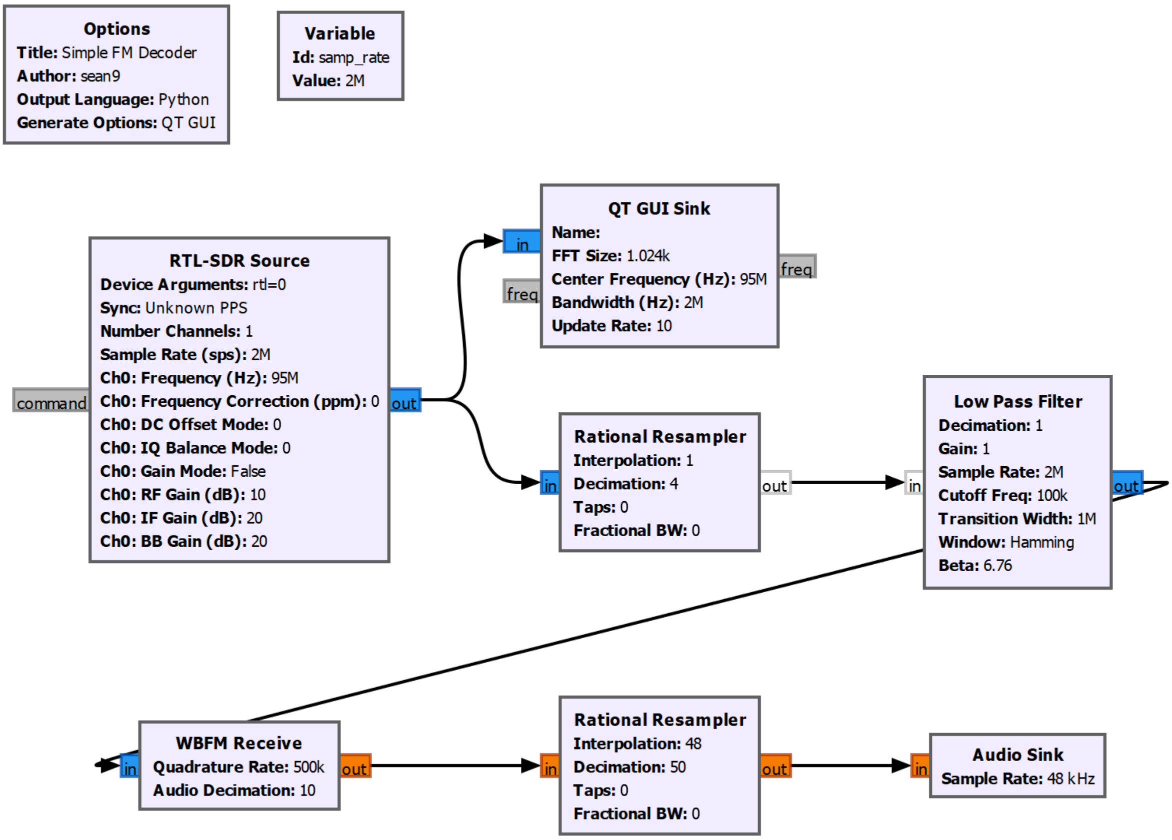


Figure B.1: A sample GNURadio flowgraph to decode the FM radio station on 95 MHz – a campus radio station at the University called 95bFM. The flowgraph is represented pictorially here, though GNURadio is capable of exporting flowgraphs as C/C++ code, or Python code, such that it may be integrated into a ground station setup.

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

#
# SPDX-License-Identifier: GPL-3.0
#
# GNU Radio Python Flow Graph
# Title: Simple FM Decoder
# Author: Sean Richards
```

```python
# GNU Radio version: 3.8.0.0

from distutils.version import StrictVersion

if __name__ == '__main__':
    import ctypes
    import sys
    if sys.platform.startswith('linux'):
        try:
            x11 = ctypes.cdll.LoadLibrary('libX11.so')
            x11.XInitThreads()
        except:
            print("Warning: failed to XInitThreads()")

from PyQt5 import Qt
from gnuradio import qtgui
from gnuradio.filter import firdes
import sip
from gnuradio import analog
from gnuradio import audio
from gnuradio import filter
from gnuradio import gr
import sys
import signal
from argparse import ArgumentParser
from gnuradio.eng_arg import eng_float, intx
from gnuradio import eng_notation
import osmosdr
import time
from gnuradio import qtgui

class FM_receiver(gr.top_block, Qt.QWidget):

    def __init__(self):
        gr.top_block.__init__(self, "Simple FM Decoder")
        Qt.QWidget.__init__(self)
        self.setWindowTitle("Simple FM Decoder")
        qtgui.util.check_set_qss()
        try:
            self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))
        except:
            pass
        self.top_scroll_layout = Qt.QVBoxLayout()
        self.setLayout(self.top_scroll_layout)
        self.top_scroll = Qt.QScrollArea()
        self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)
```

```
        self.top_scroll_layout.addWidget(self.top_scroll)
        self.top_scroll.setWidgetResizable(True)
        self.top_widget = Qt.QWidget()
        self.top_scroll.setWidget(self.top_widget)
        self.top_layout = Qt.QVBoxLayout(self.top_widget)
        self.top_grid_layout = Qt.QGridLayout()
        self.top_layout.addLayout(self.top_grid_layout)

        self.settings = Qt.QSettings("GNU Radio", "FM_receiver")

        try:
            if StrictVersion(Qt.qVersion()) < StrictVersion("5.0.0"):
                self.restoreGeometry(self.settings.value("geometry").toByteArray())
            else:
                self.restoreGeometry(self.settings.value("geometry"))
        except:
            pass

        ##################################################
        # Variables
        ##################################################
        self.samp_rate = samp_rate = 2e6

        ##################################################
        # Blocks
        ##################################################
        self.rtlsdr_source_0 = osmosdr.source(
            args="numchan=" + str(1) + " " + "rtl=0"
        )
        self.rtlsdr_source_0.set_time_unknown_pps(osmosdr.time_spec_t())
        self.rtlsdr_source_0.set_sample_rate(samp_rate)
        self.rtlsdr_source_0.set_center_freq(95e6, 0)
        self.rtlsdr_source_0.set_freq_corr(0, 0)
        self.rtlsdr_source_0.set_gain(10, 0)
        self.rtlsdr_source_0.set_if_gain(20, 0)
        self.rtlsdr_source_0.set_bb_gain(20, 0)
        self.rtlsdr_source_0.set_antenna('', 0)
        self.rtlsdr_source_0.set_bandwidth(0, 0)
        self.rational_resampler_xxx_1_0 = filter.rational_resampler_fff(
                interpolation=48,
                decimation=50,
                taps=None,
                fractional_bw=None)
        self.rational_resampler_xxx_0_0 = filter.rational_resampler_ccc(
                interpolation=1,
                decimation=4,
```

```python
                taps=None,
                fractional_bw=None)
        self.qtgui_sink_x_0 = qtgui.sink_c(
            1024, #fftsize
            firdes.WIN_BLACKMAN_hARRIS, #wintype
            95e6, #fc
            samp_rate, #bw
            "", #name
            True, #plotfreq
            True, #plotwaterfall
            True, #plottime
            True #plotconst
        )
        self.qtgui_sink_x_0.set_update_time(1.0/10)
        self._qtgui_sink_x_0_win = sip.wrapinstance(self.qtgui_sink_x_0.pyqwidget(),
        ↪  Qt.QWidget)

        self.qtgui_sink_x_0.enable_rf_freq(False)

        self.top_grid_layout.addWidget(self._qtgui_sink_x_0_win)
        self.low_pass_filter_0_0 = filter.fir_filter_ccf(
            1,
            firdes.low_pass(
                1,
                2e6,
                100e3,
                1e6,
                firdes.WIN_HAMMING,
                6.76))
        self.audio_sink_0 = audio.sink(48000, '', True)
        self.analog_wfm_rcv_0_0_0 = analog.wfm_rcv(
                quad_rate=500e3,
                audio_decimation=10,
        )



        ##################################################
        # Connections
        ##################################################
        self.connect((self.analog_wfm_rcv_0_0_0, 0),
        ↪  (self.rational_resampler_xxx_1_0, 0))
        self.connect((self.low_pass_filter_0_0, 0), (self.analog_wfm_rcv_0_0_0, 0))
        self.connect((self.rational_resampler_xxx_0_0, 0), (self.low_pass_filter_0_0,
        ↪  0))
        self.connect((self.rational_resampler_xxx_1_0, 0), (self.audio_sink_0, 0))
```

```
        self.connect((self.rtlsdr_source_0, 0), (self.qtgui_sink_x_0, 0))
        self.connect((self.rtlsdr_source_0, 0), (self.rational_resampler_xxx_0_0, 0))

    def closeEvent(self, event):
        self.settings = Qt.QSettings("GNU Radio", "FM_receiver")
        self.settings.setValue("geometry", self.saveGeometry())
        event.accept()

    def get_samp_rate(self):
        return self.samp_rate

    def set_samp_rate(self, samp_rate):
        self.samp_rate = samp_rate
        self.qtgui_sink_x_0.set_frequency_range(95e6, self.samp_rate)
        self.rtlsdr_source_0.set_sample_rate(self.samp_rate)



def main(top_block_cls=FM_receiver, options=None):

    if StrictVersion("4.5.0") <= StrictVersion(Qt.qVersion()) <
    ↪  StrictVersion("5.0.0"):
        style = gr.prefs().get_string('qtgui', 'style', 'raster')
        Qt.QApplication.setGraphicsSystem(style)
    qapp = Qt.QApplication(sys.argv)

    tb = top_block_cls()
    tb.start()
    tb.show()

    def sig_handler(sig=None, frame=None):
        Qt.QApplication.quit()

    signal.signal(signal.SIGINT, sig_handler)
    signal.signal(signal.SIGTERM, sig_handler)

    timer = Qt.QTimer()
    timer.start(500)
    timer.timeout.connect(lambda: None)

    def quitting():
        tb.stop()
        tb.wait()
    qapp.aboutToQuit.connect(quitting)
    qapp.exec_()

```

```

if __name__ == '__main__':
    main()
```

Bibliography

Airspy R2 (2019).
URL: *https://airspy.com/airspy-r2/*

Arduino Uno rev3 (n.d.).
URL: *https://store.arduino.cc/usa/arduino-uno-rev3*

Csete, A. (2008), 'csete/gpredict: Gpredict satellite tracking application'.
URL: *https://github.com/csete/gpredict*

Csete, A. (2010), 'csete/gqrx: Software defined radio receiver powered by GNU Radio and Qt.'.
URL: *https://github.com/csete/gqrx*

Csete, A. (2017), 'AFSK 1200 decode - add TCP tocket to APRS-IS'.
URL: *https://github.com/csete/gqrx/issues/540*

Diamond Antennas (2019), 'Diamond MX72A'.
URL: *https://www.diamondantenna.net/mx72a.html*

Esteves, D. (n.d.), 'GitHub - daniestevez/gr-satellites'.
URL: *https://github.com/daniestevez/gr-satellites*

EverythingRF (n.d.), 'LNA-580 - RF Bay | RF Amplifier'.
URL: *https://www.everythingrf.com/products/microwave-rf-amplifiers/rf-bay/567-791-lna-580*

Fischer, M. & Scholtz, A. L. (2010), Design of a Multi-mission Satellite Ground Station for Education and Research, *in* '2010 Second International Conference on Advances in Satellite and Space Communications', IEEE, pp. 58–63.
URL: *http://ieeexplore.ieee.org/document/5502585/*

GlavSoft LLC (n.d.), 'TightVNC: VNC-Compatible Free Remote Control / Remote Desktop Software'.
URL: *https://www.tightvnc.com/*

GNU Radio Foundation (n.d.), 'GNU Radio - The Free & Open Source Radio Ecosystem · GNU Radio'.
URL: *https://www.gnuradio.org/*

Gongliang, Z., Weide, Y. & Du, D. (1044), *Wujing Zongyao*, Emperor Renzong of Song, Kaifeng.

Good, A. (n.d.), 'k3ng/k3ng_rotator_controller: K3NG Arduino Amateur Radio Rotator Controller'.
URL: *https://github.com/k3ng/k3ng_rotator_controller*

Hi-Tec Aerials (2013), 'Hi-Tec Aerials UHF Yagis - Professional Series - Specification'.
URL: *http://bit.ly/389Bd4I*

Hitefield, S., Leffke, Z., Fowler, M. & McGwier, R. W. (2016), *System Overview of the Virginia Tech Ground Station*, IEEE.

Hoots, F. R. & Roehrich, R. L. (1988), SPACETRACK REPORT NO. 3 Models for Propagation of NORAD Element Sets, Technical report, United States Department of Defence.

Kopanchuk, A. (n.d.), ‘UZ7HO Personal page - Packet-Radio - English version’.
URL: *http://uz7.ho.ua/packetradio.htm*

LabJack Corporation (n.d.), ‘U12 | LabJack’.
URL: *https://labjack.com/products/u12*

Langner, J. (2015), ‘GitHub - wb2osz/direwolf’.
URL: *https://github.com/wb2osz/direwolf*

M2Inc (2019), ‘2MCP14’.
URL: *https://www.m2inc.com/FG2MCP14*

Nel, F. & Barry, R. (2003), *University of Stellenbosch Satellite Ground Station - Operation and Future Plans*, IEEE.

pothosware (n.d.), ‘SoapySDR’.
URL: *https://github.com/pothosware/SoapySDR/*

Radio Amateur Satellite Corporation (2020), ‘AMSAT OSCAR Satellite Status’.
URL: *https://www.amsat.org/status/*

Raspberry Pi Foundation (2019), ‘Raspberry Pi R3’.
URL: *https://www.raspberrypi.org/products/raspberry-pi-3-model-b/*

Richards, S. (2019), *University of Auckland Ground Station One (PGDipSci Dissertation)*, The University of Auckland, Auckland.

RTL-SDR Blog V.3. Dongles User Guide (n.d.).
URL: *https://www.rtl-sdr.com/rtl-sdr-blog-v-3-dongles-user-guide/*

Rupprecht, M. (2020), ‘Status - DK3WN Satblog’.
URL: *http://www.dk3wn.info/p/?page_id=29535*

satellogic (2019), ‘Orbit-Predictor’.
URL: *https://github.com/satellogic/orbit-predictor*

Shamutally, F., Soreefan, Z., Suddhoo, A. & Momplé, J. M. (2018), Practical "low cost" LEO receiving ground station, Technical report, Mauritius Research Council.

Thirumalai, A. (2017), *CubeSat Ground Station Development - An Automated Tracking System for Amateur Satellites (MEngStud Thesis)*, The University of Auckland, Auckland.

Vallado, D. A., Crawford, P., Hujsak, R. & Kelso, T. S. (2006), Revisiting Spacetrack Report #3, Technical report, Colorado Center for Space Standards and Innovation.

Yaesu (2019), ‘Welcome to Yaesu.com’.
URL: *http://bit.ly/35o6j7g*