Decoding Satellite Signals Using Software Defined Radios

Wyatt Raymond*

*Department of Electrical and Computer Engineering

Old Dominion University, Norfolk, VA 23529 USA

Abstract

The rapid advancement of telecommunications and satellite communications technology has driven the need for more adaptable communication systems. This report explores the use of Software Defined Radios (SDRs) for decoding satellite signals, leveraging software flexibility to process and interpret signals across diverse frequencies and standards. Unlike traditional hardware-based radios, SDRs perform signal processing tasks through software, enabling quick adaptation to new protocols without physical modifications. This adaptability is crucial in the dynamic satellite communication environment, where various signal types and frequencies are encountered. This report demonstrates the practical implementation and experimentation with SDRs for receiving and decoding signals from weather and educational satellites, showcasing the versatility and effectiveness of SDR technology using affordable hardware like the RTL-SDR dongle and sophisticated software tools. The findings highlight the potential of SDRs to manage complex satellite communications with minimal investment, providing insights into the evolution, integration, and real-world applications of SDR technology.

Index Terms

Software Defined Radio (SDR), Satellite Communication, RTL-SDR, NOAA Satellites, APT System, Signal Processing, BEESAT Satellites

I. Introduction

The rapid advancement of technology in the field of telecommunications and satellite communications has driven the need for more flexible and adaptable communication systems. This report delves into the application of Software Defined Radios (SDRs) for decoding satellite signals, a cutting-edge approach

Manuscript received August 1, 2024; ... Corresponding author: W. Raymond (email: wraym002@odu.edu).

that leverages the flexibility of software to process and interpret signals across a wide range of frequencies and standards.

Software Defined Radios have emerged as a pivotal technology, transforming the landscape of satellite communications. Unlike traditional hardware-based radios, SDRs perform signal processing tasks through software, enabling swift adaptation to new protocols and standards without the need for physical modifications. This adaptability is particularly crucial in the dynamic and diverse environment of satellite communications, where multiple signal types and frequencies are encountered.

The focus of this report is on the practical implementation and experimentation with SDRs for receiving and decoding signals from various satellites. The aim is to demonstrate the versatility and effectiveness of SDR technology in real-world applications, including weather monitoring and telemetry from environmental and educational satellites. By utilizing affordable and accessible SDR hardware, such as the RTL-SDR dongle, alongside sophisticated software tools, we illustrate how complex satellite communications can be managed with minimal investment.

The report is structured to provide a comprehensive overview of the history and evolution of SDR technology, its integration with satellite communications, and the specific methodologies employed in the experiments. It also includes a detailed examination of the hardware and software utilized, the setup and execution of the experiments, and the analysis of the results obtained.

II. Literature Review

A. History Of Software Defined Radio

The development and evolution of Software Defined Radio (SDR) technology has had a profound impact on the field of telecommunications and satellite communications. The concept of SDR emerged in the late 20th century, driven by the need for more flexible and adaptable communication systems that could operate across a wide range of frequencies and standards without the need for extensive hardware modifications.

The early 2000s saw significant advancements in SDR technology, particularly in the field of Global Navigation Satellite Systems (GNSS). One of the pioneering works in this era was the application of SDR to GNSS signal processing. Kovář and Vejražka highlighted the benefits of using SDR for digitalizing and processing satellite navigation signals. This early research demonstrated the potential of SDR to handle the processing procedures of new GNSS signals such as GPS L2 C/A, GPS L5, and Galileo signals [1].

As SDR technology matured, its applications expanded significantly. Sharawi and Korniyenko demonstrated the flexibility of SDR by implementing a non-real-time GPS software receiver capable of decoding raw GPS data and the civil navigation message. This work underscored the adaptability of SDR in handling different satellite communication standards with minimal hardware changes [2].

The integration of SDR with modern processing technologies further enhanced its capabilities. In recent years, the use of graphics processing units (GPUs) for real-time SDR operations has been explored. Peters and Benson developed an SDR receiver design that utilizes GPUs for online Doppler search and demodulation of satellite signals, highlighting the advanced computational power now available for SDR applications [3].

In the context of satellite communications, SDR has proven to be a versatile and cost-effective solution. The development of SDR-based systems for satellite telemetry and image reception, as demonstrated by Peralta et al., shows the practical applications of SDR in capturing and decoding signals from various types of satellites, including NOAA and picosatellites [4].

Recent advancements also include the implementation of multicore SDR architectures for GNSS receivers, which allow for processing signals from multiple satellites simultaneously and improving navigation accuracy. Hurskainen et al. presented such an architecture, demonstrating its potential for both flexibility and mobile applications [5].

The history of SDR is marked by continuous innovation and expansion into various fields of communication technology. From its early application in GNSS to its modern integration with GPUs and multicore architectures, SDR has evolved into a critical technology for flexible and adaptable communication systems, particularly in satellite communications. The ongoing research and development in this field promise even greater advancements and applications in the future.

B. Evolution and Architecture of SDR

SDR architectures have significantly evolved from traditional hardware-centric designs to sophisticated software-centric models, primarily driven by advancements in digital signal processing (DSP) technology. Early SDRs followed a superheterodyne architecture, where a high-frequency RF signal was downconverted to a lower intermediate frequency (IF) for easier processing. This downconversion process typically involved a series of mixers, local oscillators, and filters, which were all implemented in hardware. The signal at the IF stage was then digitized using an analog-to-digital converter (ADC) and processed digitally to extract the desired information [6].

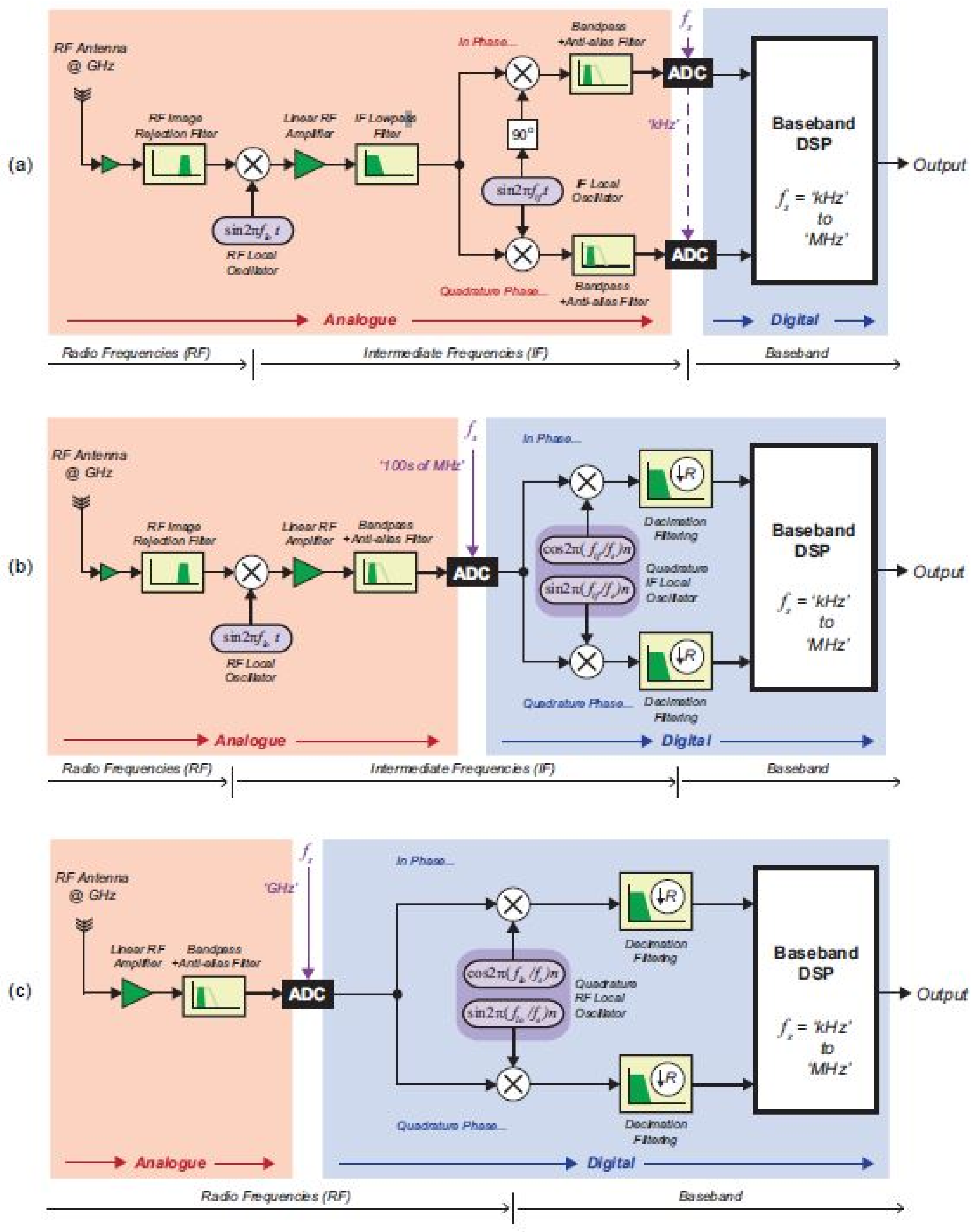


Fig. 1. Evolution of SDR; The sampling rates of ADC increase and move closer to antenna, starting with (a) Baseband Digital Radio, to (b) IF Digital Radio and finally (c) RF Digital Radio. [6]

As digital technology advanced, particularly with the development of high-speed ADCs and digital-to-analog converters (DACs), SDR architectures began to move away from the traditional IF downconversion model. Modern SDR systems often employ direct conversion or zero-IF (ZIF) architectures, where the RF signal is directly converted to baseband in one step. This is achieved by mixing the incoming RF signal with a local oscillator signal that is at or very close to the carrier frequency, effectively eliminating the need for multiple stages of downconversion.

In a direct conversion architecture, the baseband signal, which is typically a complex signal with in-phase (I) and quadrature (Q) components, is directly digitized by the ADC. This approach simplifies the signal chain, reduces the number of analog components, and minimizes issues such as image frequency and phase noise that are common in superheterodyne receivers. The digitized I/Q data is then processed by a DSP or a field-programmable gate array (FPGA) to perform various signal processing tasks such as filtering, demodulation, and decoding. The use of DSP and FPGA technology in SDRs allows for highly flexible and reconfigurable systems that can be updated or modified through software, enabling support for multiple communication standards and protocols [6].

C. Use of SDR to Communicate with Satellites

The integration of Software Defined Radio (SDR) technology with satellite communication systems has been a significant advancement in both fields, enabling more flexible, efficient, and adaptive communication solutions. The history of satellites in relation to SDR provides insight into the technological evolution and the increasing capabilities of modern satellite systems.

Satellite communication began with basic radio frequency transmissions, which evolved rapidly with advancements in technology. The initial successes in radio communications, such as those achieved by Marconi with low and very low-frequency waves, laid the foundation for satellite communications. This evolution continued through the mid-20th century, characterized by the launch of communication satellites with high capacity and reliability [7].

The concept of Software Defined Radio (SDR) was developed in the late 20th century, initially driven by the need for more adaptable and flexible communication systems. Early SDR implementations in satellite communications focused on digitalizing and processing radio signals using programmable digital circuits, such as high power digital signal processors (DSPs) and field programmable gate arrays (FPGAs) [1].

One of the pioneering uses of SDR in satellite communications was its application in small satellite missions, such as CubeSats. These small satellites, with their limited power and size, benefited significantly from SDR's ability to support various modulation schemes and data rates without requiring hardware modifications. Grayver et al. demonstrated the use of a 915 MHz SDR based on a Zynq processor for CubeSats, highlighting its capability to maximize downlink throughput by adapting coding and modulation in real-time [3].

SDR technology has also been instrumental in supporting multi-satellite communication systems. Maheshwarappa proposed an SDR architecture that combines FPGA System-on-Chip (SoC) with RF programmable transceivers to handle signals from multiple satellites, addressing the challenges of bandwidth, data rate, and processing requirements [4].

Recent developments have seen SDRs being used for more complex satellite missions and telemetry applications. For instance, Peralta et al. utilized SDR for receiving telemetry and imagery from various types of satellites, including NOAA class satellites, demonstrating the flexibility of SDR in different space communication scenarios [5].

Furthermore, the evolution of SDR has led to advanced cognitive radio systems capable of automatically adjusting parameters such as frequency and modulation to optimize communication under varying conditions. These cognitive SDRs are now being explored for their potential to enhance satellite communications by providing more robust and adaptable solutions [7].

The history of satellites in relation to SDR reflects a continuous advancement in technology, leading to more flexible and efficient communication systems. From the early days of basic radio transmissions to the modern use of cognitive SDRs, the integration of these technologies has significantly enhanced the capabilities of satellite communications, promising even greater advancements in the future.

III. Operation of Software Defined Radios

In this section, the detailed technical aspects of Software Defined Radio (SDR) operation are reviewed, focusing on the key principles and mathematical foundations that govern their functionality. Understanding these fundamentals is crucial for effectively leveraging SDRs in both research and practical applications.

A. Signal Representation in SDRs

SDRs are designed to handle various types of signals, primarily focusing on bandpass signals, which are signals with a spectrum concentrated around a specific carrier frequency, f_c. A bandpass signal, $s(t)$, can be mathematically represented as:

$$s(t) = \mathrm{Re}\{s_I(t)\cos(2\pi f_c t) - s_Q(t)\sin(2\pi f_c t)\} \tag{1}$$

Where $s_I(t)$ and $s_Q(t)$ are the in-phase and quadrature components of the signal, respectively. These components are crucial in SDR operations as they allow the separation of the signal into two orthogonal components, facilitating easier manipulation and analysis in the digital domain.

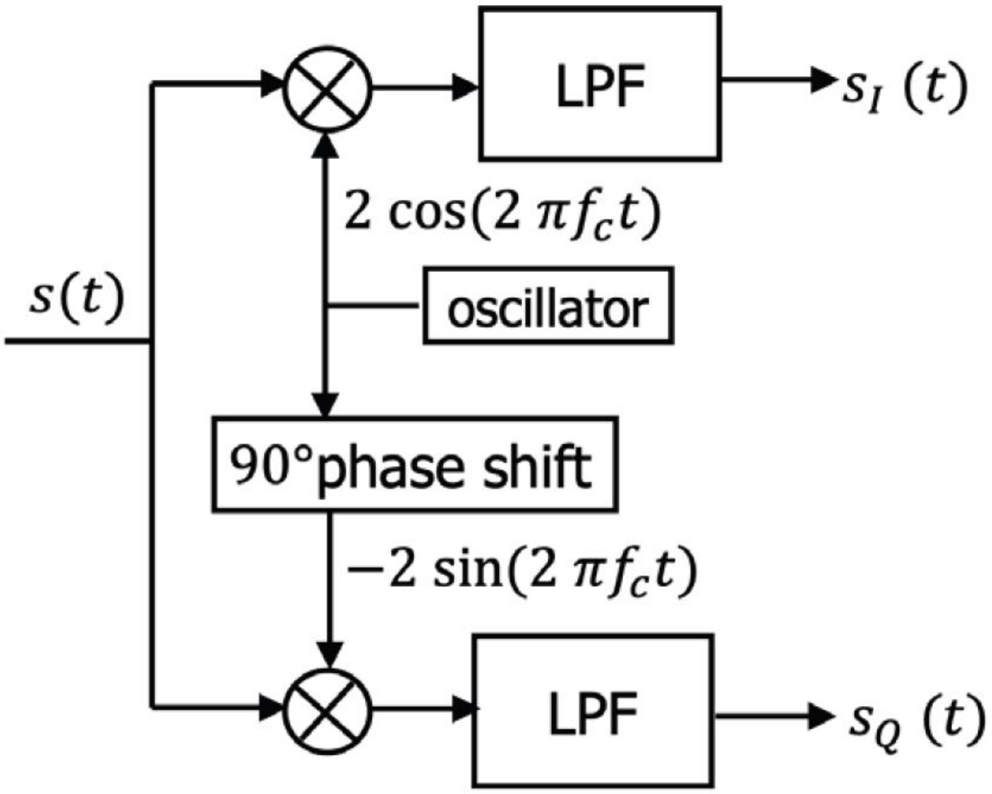


Fig. 2. Obtaining IQ components from band pass signal [8]

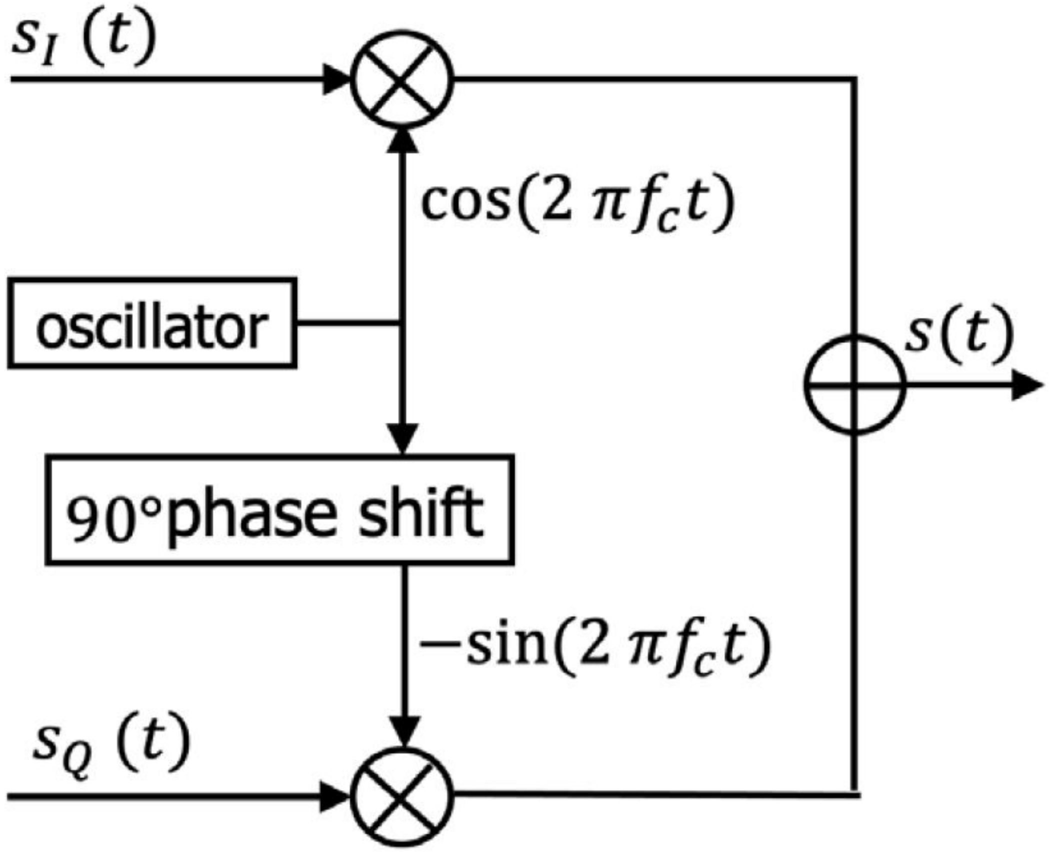


Fig. 3. Synthesizing IQ components from band pass signal [8]

The mathematical operations in an SDR involve converting these bandpass signals into their baseband equivalents using a process known as downconversion. The downconverted baseband signal, $s_{bb}(t)$, is represented as:

$$s_{bb}(t) = s_I(t) + js_Q(t) \tag{2}$$

This complex signal allows the SDR to process both amplitude and phase information digitally. [8]

B. SDR Hardware Architecture

The ideal SDR architecture, involves direct conversion of the RF signal to digital form at the antenna, with all subsequent processing occurring in the digital domain. This approach, depicted in Figure 4, would theoretically allow the SDR to handle signals across a wide range of frequencies with minimal analog components. However, due to current technological limitations, particularly in the performance of analog-to-digital converters (ADC) and digital-to-analog converters (DAC), most SDRs employ an intermediate frequency (IF) stage.

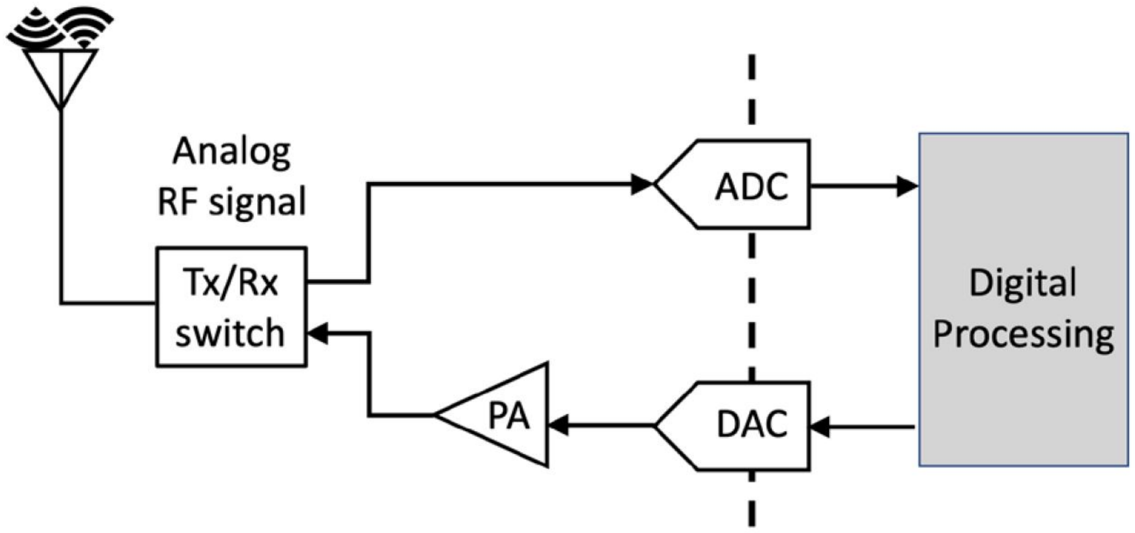


Fig. 4. Ideal SDR Architecture [8]

In practice, the architecture illustrated in Figure 5 is more commonly implemented. This architecture consists of two main stages: the RF front end and the digital processing stage, bridged by an IF stage where the ADC and DAC conversions take place. [8]

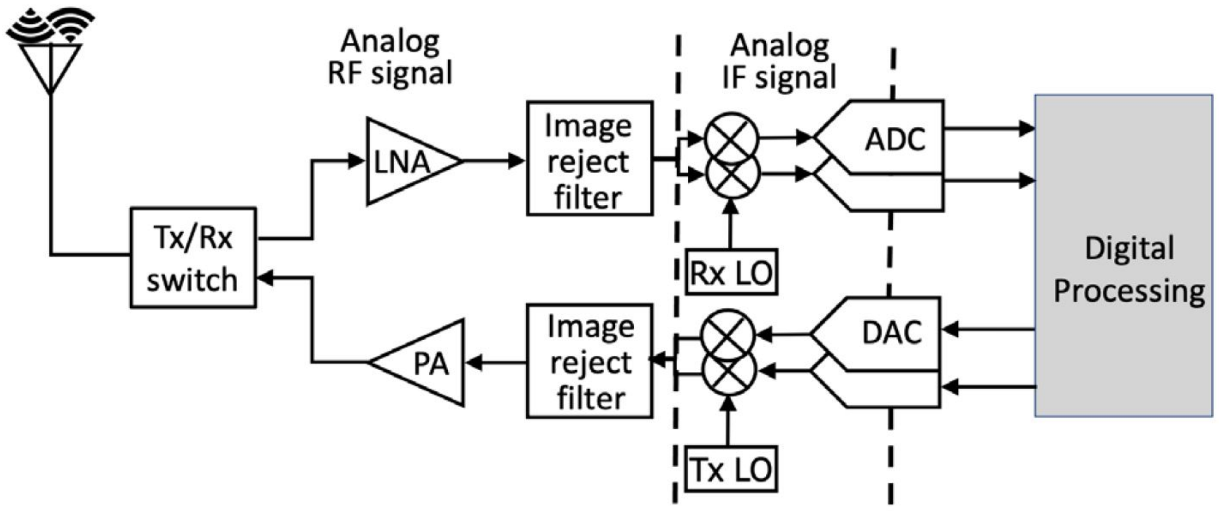


Fig. 5. SDR Architecture used in practice [8]

1) RF Front End and IF Stage: The RF front end is responsible for the initial reception and transmission of signals. On the receiver side, the incoming RF signal is first amplified using a low-noise amplifier (LNA), which enhances the signal strength while minimizing the noise figure. The signal is then downconverted to an intermediate frequency (IF) by mixing it with a local oscillator signal. This downconversion step is crucial because it allows the signal to be processed at a lower frequency, where ADCs can operate more efficiently. The IF stage thus serves as a bridge, enabling the digitization of high-frequency signals with existing ADC technology.

On the transmitter side, the process is reversed. The baseband signal, which is digitally synthesized, is first upconverted to the IF. After this, the signal undergoes digital-to-analog conversion, producing an analog IF signal. This IF signal is then further upconverted to the desired RF band using a power amplifier (PA) before being transmitted. The PA is critical for ensuring that the transmitted signal meets the required power levels for effective communication. [8]

2) Digital Front End and Baseband Processing: The digitized signal from the IF stage enters the digital processing section of the SDR, which is typically divided into the digital front end and the baseband processing stage. The digital front end performs two primary functions:

- **Sample Rate Conversion:** This adapts the sampling rate from the IF stage to the rate required for baseband processing. The sampling rate must be carefully chosen to balance between the available processing power and the resolution needed for accurate signal reconstruction.
- **Channelization:** This involves filtering the digitized signal to isolate specific frequency channels of interest. The digital front end often includes decimation filters to reduce the data rate, making it more manageable for subsequent baseband processing.

In the baseband processing stage, the SDR performs the core signal processing tasks, including modulation, demodulation, and error correction. These tasks are typically implemented using DSP algorithms, which can be reconfigured through software to accommodate different communication standards. This flexibility is one of the key advantages of SDRs, allowing them to adapt to various signal types and protocols without requiring changes to the hardware.

The digital front end is usually implemented on an FPGA (Field Programmable Gate Array), which offers the necessary computational power and flexibility to handle real-time signal processing tasks. In many modern SDR platforms, the FPGA is co-located with the RF front end and IF components on the same board, enabling compact and efficient designs. The baseband processing may be performed on the FPGA itself or offloaded to a general-purpose processor (GPP) in a host computer, depending on the complexity of the processing tasks and the capabilities of the FPGA. [8]

3) Integration and Implementation Challenges: In most SDR implementations, the RF front end, IF stage, and digital front end are integrated into a single chip or module, such as the AD936x transceiver chips used in Ettus Research's USRP series. This level of integration helps reduce the overall size, power consumption, and cost of the SDR platform, making it more accessible for various applications, from academic research to commercial deployment.

However, integrating all these components poses significant challenges, particularly in maintaining signal integrity across the different stages. Careful design considerations are necessary to ensure that the signal-to-noise ratio (SNR) is preserved, especially during the analog-to-digital and digital-to-analog conversion processes. Additionally, power management is critical, as the RF amplifiers and digital processing stages can consume significant amounts of power, particularly in high-performance SDRs.

In conclusion, while the ideal SDR architecture promises maximum flexibility and performance, practical limitations necessitate the use of an intermediate frequency stage in most current implementations. This architecture, as shown in Figure 5, strikes a balance between the theoretical ideal and the practical realities of current ADC and DAC technologies, enabling SDRs to operate effectively across a wide range of frequencies and applications. [8]

C. Mathematical Modeling of Noise in SDRs

Noise is an inherent aspect of all communication systems, including SDRs. It is typically modeled as an additive white Gaussian noise (AWGN) process. The received signal $r(t)$ in the presence of noise can be modeled as:

$$r(t) = s(t) + n(t) \tag{3}$$

Where $n(t)$ is the noise component, assumed to have a mean of zero and a power spectral density $N_0/2$. The performance of SDRs in the presence of noise is often evaluated using metrics such as the signal-to-noise ratio (SNR), which is defined as:

$$\text{SNR} = \frac{P_s}{P_n} \tag{4}$$

Where P_s and P_n are the power of the signal and noise, respectively. Understanding and mitigating the effects of noise is crucial in SDR design, particularly in applications involving weak signal detection. [8]

IV. Satellites of Interest

A. NOAA Satellites

The National Oceanic and Atmospheric Administration (NOAA) operates a series of environmental satellites that have been integral to weather forecasting, climate monitoring, and environmental observation for decades. These satellites include both geostationary and polar-orbiting types, each serving specific roles in data collection and analysis.

NOAA satellites have employed various technologies to enhance data collection and transmission capabilities. The Advanced Very High Resolution Radiometer (AVHRR) sensors, for instance, are used to acquire remote sensing data and broadcast Automatic Picture Transmission (APT) images. The APT system is a fundamental method used by NOAA satellites, to transmit low-resolution images of the Earth's surface. The APT system encodes satellite images into audio tones that can be received and decoded by ground stations equipped with appropriate software and hardware.

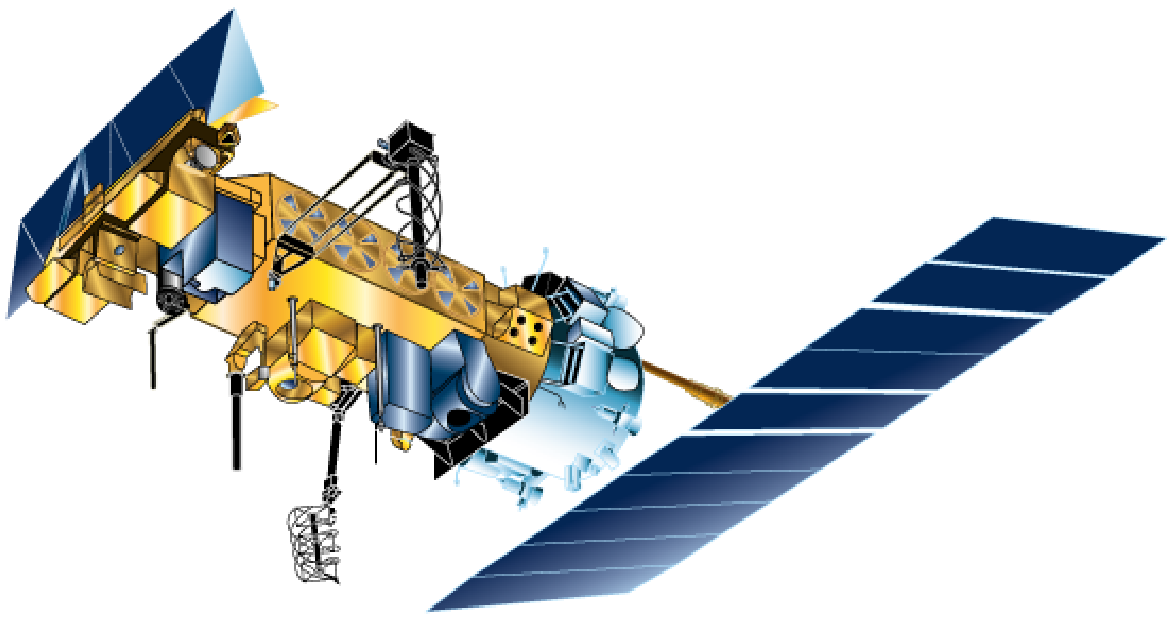


Fig. 6. NOAA 19 Satellite [9]

The APT encoding process begins with the satellite capturing images in multiple spectral bands, typically visible and infrared. These images are then converted into an analog signal. In the APT system, each image line is transformed into a series of frequency-modulated audio tones. The process involves translating the pixel intensity values into corresponding audio frequencies, which are then transmitted to the ground. This method utilizes frequency modulation (FM) to encode the pixel data into an audio signal that can be easily transmitted over long distances and received with simple radio equipment [10].

The APT signal is broadcast continuously as the satellite passes over the Earth. Ground stations equipped with VHF antennas and receivers tuned to the appropriate frequency bands (137-138 MHz) capture these signals. A typical setup for receiving APT images includes a software-defined radio (SDR) receiver, a computer, and specialized software capable of demodulating and decoding the received audio tones back into an image format [11].

Once the audio signal is received, the decoding software processes it to reconstruct the image. This involves several steps, the first being software demodulating the frequency-modulated signal to retrieve the original pixel intensity values encoded in the audio tones. this is followed by each line of the image is reconstructed from the series of audio frequencies received, with the timing of the lines synchronized to the satellite's transmission rate. Lastly the decoded lines are assembled to form a complete image. This process may also involve correcting for Doppler shifts, which occur due to the relative motion of the satellite and the ground station, affecting the received signal's frequency [11].

Recent advancements have focused on leveraging Software Defined Radio (SDR) technology to improve the reception and processing of these satellite signals. SDR provides flexibility and cost-effectiveness, enabling the use of inexpensive hardware such as the RTL-SDR dongle to receive NOAA satellite data [12]. Several projects have demonstrated the successful integration of SDR with NOAA satellite data reception. For example, Peralta et al. utilized SDR for receiving telemetry and imagery from NOAA satellites in Brazil, demonstrating its adaptability and cost-effectiveness for educational and outreach purposes [4]. Another study by Bósquez et al. analyzed the design of a land station using SDR to download meteorological images from NOAA satellites, highlighting the combination of SDR with adaptive antennas and server technologies to facilitate remote analysis and access to meteorological data [12].

These advancements in SDR technology have significantly enhanced the capabilities of NOAA satellites in data collection and dissemination, making it easier and more affordable to obtain critical environmental data for various applications, including disaster management and climate research [13].

B. BEESAT Satellites

The Berlin Experimental and Educational Satellite (BEESAT) project, developed by the Technical University of Berlin, is a series of pico-satellites aimed at demonstrating new space technologies and conducting scientific experiments in orbit. The primary objective of BEESAT is to validate the performance of miniaturized reaction wheels for attitude control in space [14].

The BEESAT series has incorporated SDR technology to enhance its communication capabilities. The use of SDR allows for flexible and reconfigurable communication systems that can adapt to various modulation schemes and data rates. This flexibility is particularly beneficial for small satellites like BEESAT, which have limited power and space for traditional communication hardware [15].

Fig. 7. BEESAT-1 Satellite [16]

The primary type of information encoded and transmitted by BEESAT satellites includes telemetry data, status reports, and experimental results. These data are typically encoded using binary phase-shift keying (BPSK) or quadrature phase-shift keying (QPSK) modulation schemes. BPSK is preferred for its robustness in noisy environments, while QPSK is utilized for higher data rate transmissions due to its ability to encode two bits per symbol [17]. Additionally, advanced modulation formats such as 8PSK and 16-QAM are considered for improving bandwidth efficiency, although they require higher power and more complex error correction techniques [18].

The modulation format determines how the digital data is mapped to the carrier wave, which is then transmitted by the satellite. For instance, in QPSK modulation, the data is split into two streams, each modulating a carrier signal that is 90 degrees out of phase with the other. This allows the transmission of two bits per symbol, effectively doubling the data rate compared to BPSK while maintaining the same bandwidth [19].

Decoding these signals on the ground involves several steps. First, the received signal must be demodulated to retrieve the encoded binary data. This process often involves techniques such as Viterbi decoding, which is particularly effective for convolutional codes used in satellite communications. Viterbi decoders operate by finding the most likely sequence of transmitted symbols, considering the entire received signal sequence to minimize error rates [17].

Error correction is also crucial in satellite communications to ensure data integrity. Turbo codes and Low-Density Parity-Check (LDPC) codes are commonly used due to their excellent error-correcting performance and relatively low decoding complexity. These codes work by adding redundant bits to the transmitted data, which the decoder uses to detect and correct errors introduced during transmission [20] [21].

For instance, turbo decoding involves iterative processing where the received signal is demodulated to produce soft decisions (probabilistic estimates of the transmitted bits) that are refined through multiple iterations of decoding and re-encoding. This iterative process significantly reduces the bit error rate (BER), enhancing the reliability of the communication link [22].

One notable achievement of the BEESAT project is the successful on-orbit verification of micro reaction wheels, which are critical for precise attitude control in pico-satellites. These wheels, developed in cooperation with industry partners, have demonstrated significant improvements in the maneuverability and stability of small satellites in space [14].

The integration of SDR with BEESAT's communication systems has enabled more efficient and reliable data transmission, facilitating better control and monitoring of satellite operations. This approach aligns with the broader trend of using SDR in space applications to achieve greater flexibility and performance in satellite communications [15].

V. Hardware

A. *RTL-SDR*

In the rapidly evolving field of satellite communication, commercially available Software Defined Radios (SDRs) have become indispensable tools for decoding satellite signals. This section explores various SDR models that are widely used for this purpose, highlighting their unique features, capabilities, and suitability for different satellite decoding applications. From high-performance devices offering extensive frequency ranges and advanced processing power to more budget-friendly options ideal for hobbyists and educational purposes, we will examine how these SDR models contribute to the efficient and effective reception and interpretation of satellite data.

The RTL-SDR is a versatile and cost-effective Software Defined Radio (SDR) that has gained significant popularity among both amateur radio enthusiasts and professionals. Originally designed as a DVB-T TV tuner, the RTL-SDR can be repurposed to function as a wideband SDR capable of receiving frequencies from approximately 500 kHz to 1.75 GHz. This wide frequency range allows users to explore a multitude of signals, including AM, FM, and various digital modes [1].

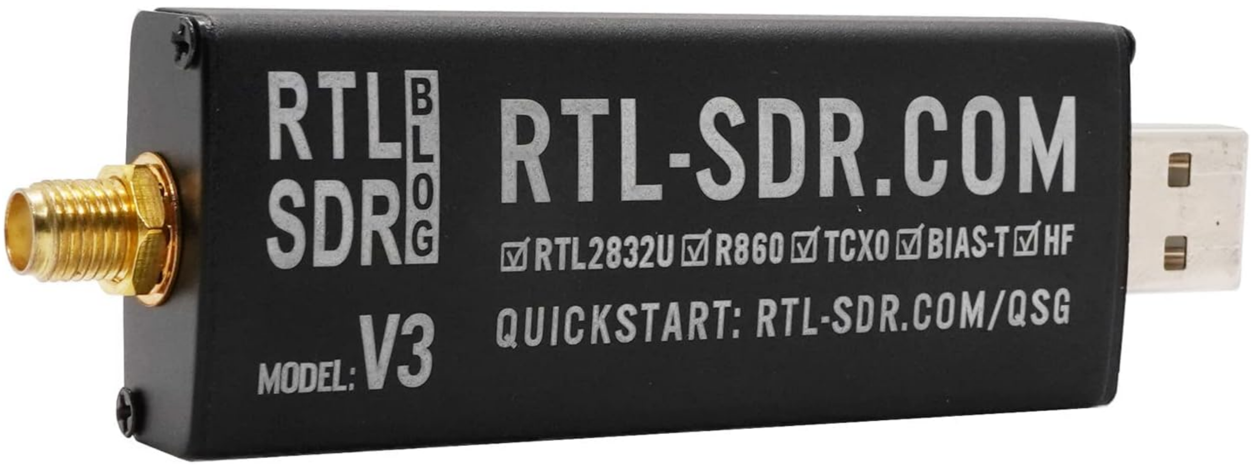

RTL SDR BLOG
RTL-SDR.COM
☑RTL2832U ☑R860 ☑TCXO ☑BIAS-T ☑HF
QUICKSTART: RTL-SDR.COM/QSG
MODEL: V3

Fig. 8. RTL-SDR Blog 3, a model of SDR used in this project [23]

One of the standout features of the RTL-SDR is its affordability, making advanced radio frequency technology accessible to a broader audience. Despite its low cost, the device does not compromise on functionality. It supports a variety of modulation schemes and protocols, making it suitable for diverse applications such as weather satellite decoding, aircraft tracking with ADS-B, and general radio scanning. Additionally, the RTL-SDR can be used for decoding digital modes like DMR, P25, and TETRA, as well as for listening to amateur radio bands [24].

The device's popularity is further enhanced by the extensive range of compatible software available, both free and paid. Popular software options include SDRSharp, which provides a user-friendly interface for Windows users, and GQRX, a versatile option for Linux and Mac OS users. These software tools enable users to customize their listening experience and extract maximum utility from their RTL-SDR devices [24].

Furthermore, the RTL-SDR community is robust and active, offering extensive resources, forums, and user-contributed projects that enhance the device's functionality and ease of use. This community support, combined with continuous updates and improvements to both the hardware and software, ensures that the RTL-SDR remains a relevant and powerful tool for radio frequency exploration [24].

In conclusion, the RTL-SDR stands out as a powerful, flexible, and affordable SDR option suitable for a wide range of applications. Its ability to receive a broad spectrum of frequencies, combined with extensive software support and a strong user community, makes it an invaluable tool for both beginners and experienced radio enthusiasts [24].

The RTL-SDR (Realtek Software Defined Radio) serves as a prime example of how SDR principles are applied in practice, particularly in low-cost, accessible systems. The RTL-SDR originally began as a digital video broadcasting (DVB-T) receiver, but due to its wide frequency range and general-purpose nature, it has been repurposed for SDR applications.

The RTL-SDR consists of several key components:

- **Rafael Micro R820T Tuner:** This tuner chip downconverts the RF signal to an intermediate frequency. It supports a wide range of frequencies, from 24 MHz to 1.76 GHz, allowing the RTL-SDR to receive a broad spectrum of signals, including FM radio, TV broadcasts, and various other communication signals.
- **RTL2832U Demodulator:** This chip is responsible for digitizing the IF signal. It includes a built-in ADC that converts the analog signal to a digital I/Q stream. The RTL2832U can process data rates up to 3.2 MS/s, which is sufficient for many SDR applications.
- **USB Interface:** The digitized I/Q data is transferred to a computer via a USB interface. The data can then be processed by software such as MATLAB, GNU Radio, or other SDR platforms. The flexibility of the RTL-SDR lies in its software-based processing, where users can implement custom signal processing algorithms, analyze different signal types, and experiment with various communication protocols [6].

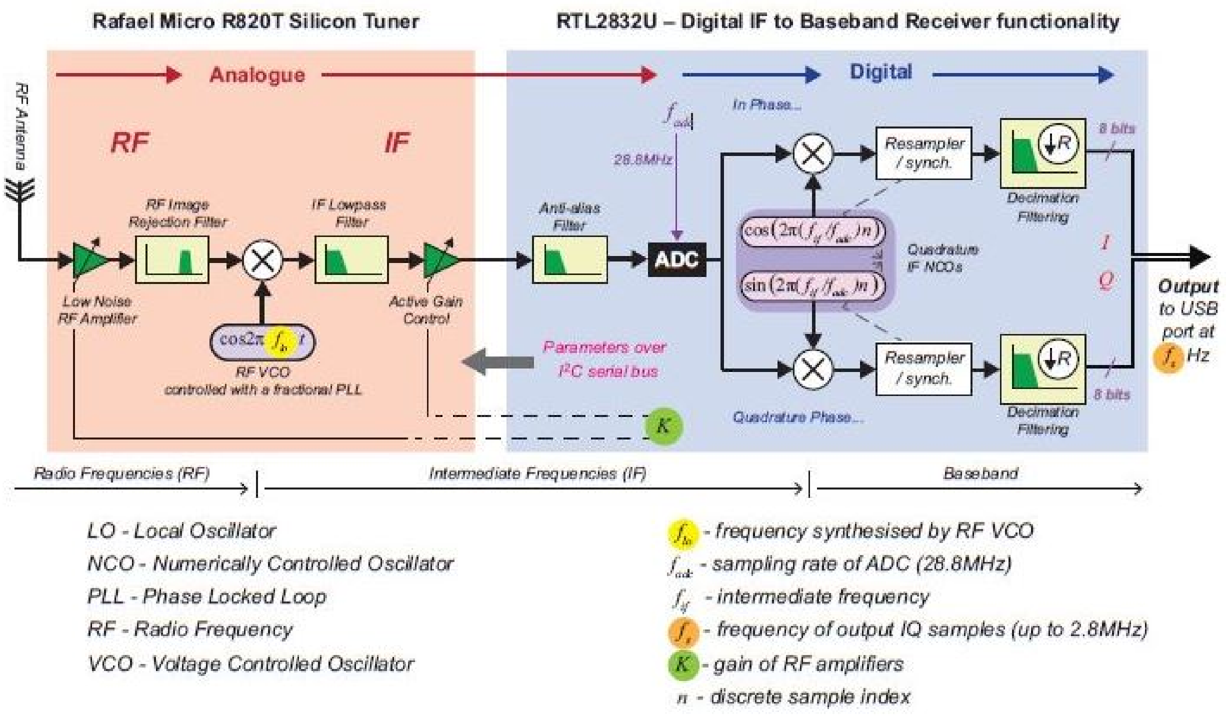

Rafael Micro R820T Silicon Tuner
RTL2832U – Digital IF to Baseband Receiver functionality
Analogue
Digital
RF Antenna
RF
IF
RF Image Rejection Filter
IF Lowpass Filter
Low Noise RF Amplifier
Active Gain Control
RF VCO controlled with a fractional PLL
Anti-alias Filter
28.8MHz
ADC
In Phase...
Quadrature Phase...
Quadrature IF NCOs
Resampler / synch.
Decimation Filtering
8 bits
Output to USB port at Hz
Parameters over I²C serial bus
Radio Frequencies (RF)
Intermediate Frequencies (IF)
Baseband
LO - Local Oscillator
NCO - Numerically Controlled Oscillator
PLL - Phase Locked Loop
RF - Radio Frequency
VCO - Voltage Controlled Oscillator
- frequency synthesised by RF VCO
- sampling rate of ADC (28.8MHz)
- intermediate frequency
- frequency of output IQ samples (up to 2.8MHz)
- gain of RF amplifiers
- discrete sample index

Fig. 9. RTL-SDR Block Diagram [6]

Figure 9 provides a detailed block diagram of a typical SDR system. It clearly illustrates the flow of signals from the RF front end, through the ADC, into the digital processing domain, and finally, back to the analog domain (if necessary) for transmission. The figure highlights the key stages in the SDR architecture, including the tuner, ADC, DSP, and DAC components. Each of these stages plays a crucial role in ensuring that the SDR system can flexibly adapt to different signal environments, process a wide range of frequencies, and implement various communication protocols through software [6].

B. Antennas

Software-Defined Radio (SDR) technology necessitates flexible and high-performance antenna systems to leverage its potential fully. Antennas used in SDR must handle a wide range of frequencies and provide reconfigurability to adapt to various communication standards and environments.

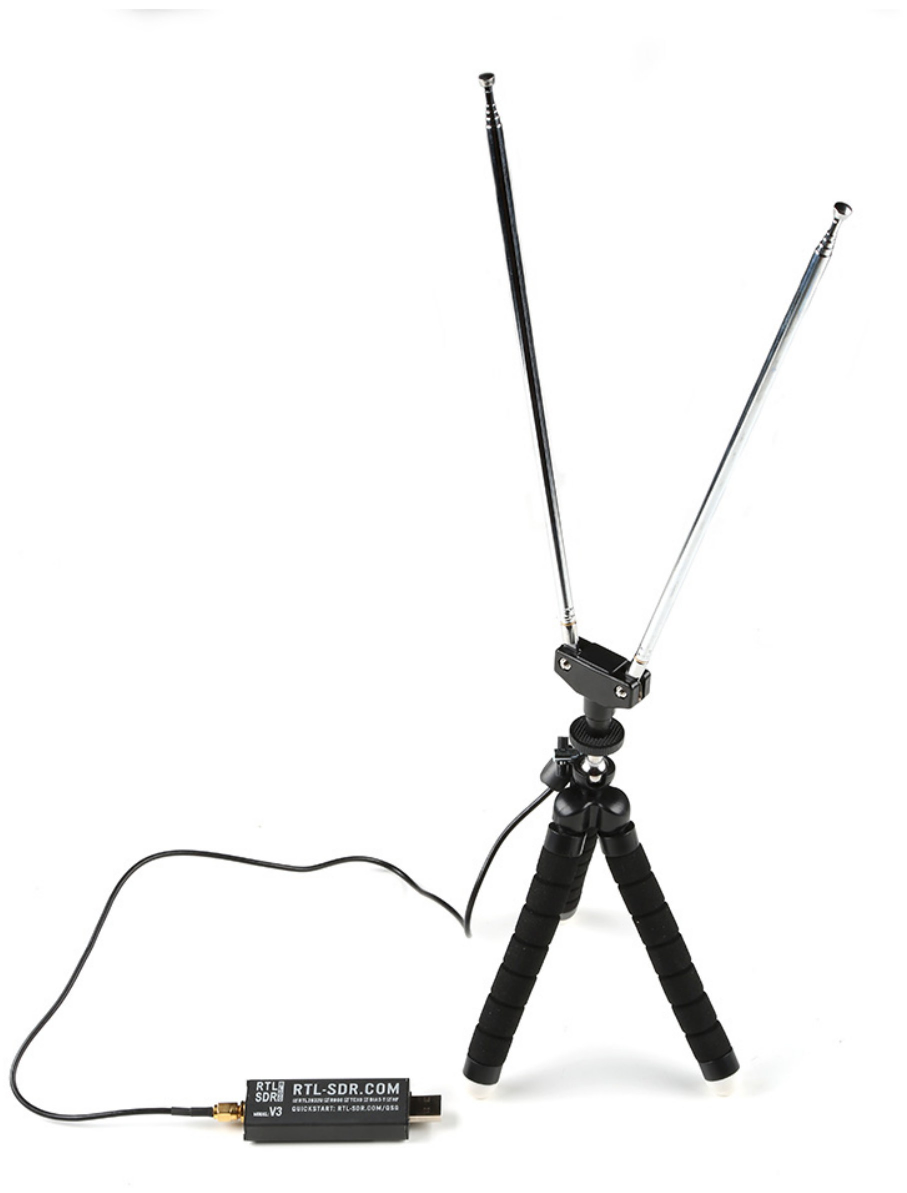


Fig. 10. A portable dipole Antenna connected to an RTL-SDR [23]

1) Di-Pole: Dipole antennas are one of the simplest and most commonly used types of antennas in SDR applications. They consist of two conductive elements, usually rods, which are fed by an alternating current. This configuration creates an oscillating electromagnetic field, enabling the antenna to transmit and receive signals effectively. Dipole antennas are appreciated for their simplicity, ease of construction, and relatively broad bandwidth, which makes them suitable for various SDR applications. Dipole antennas can be adapted into more complex structures to support multiple communication protocols with linear and circular polarizations [25].

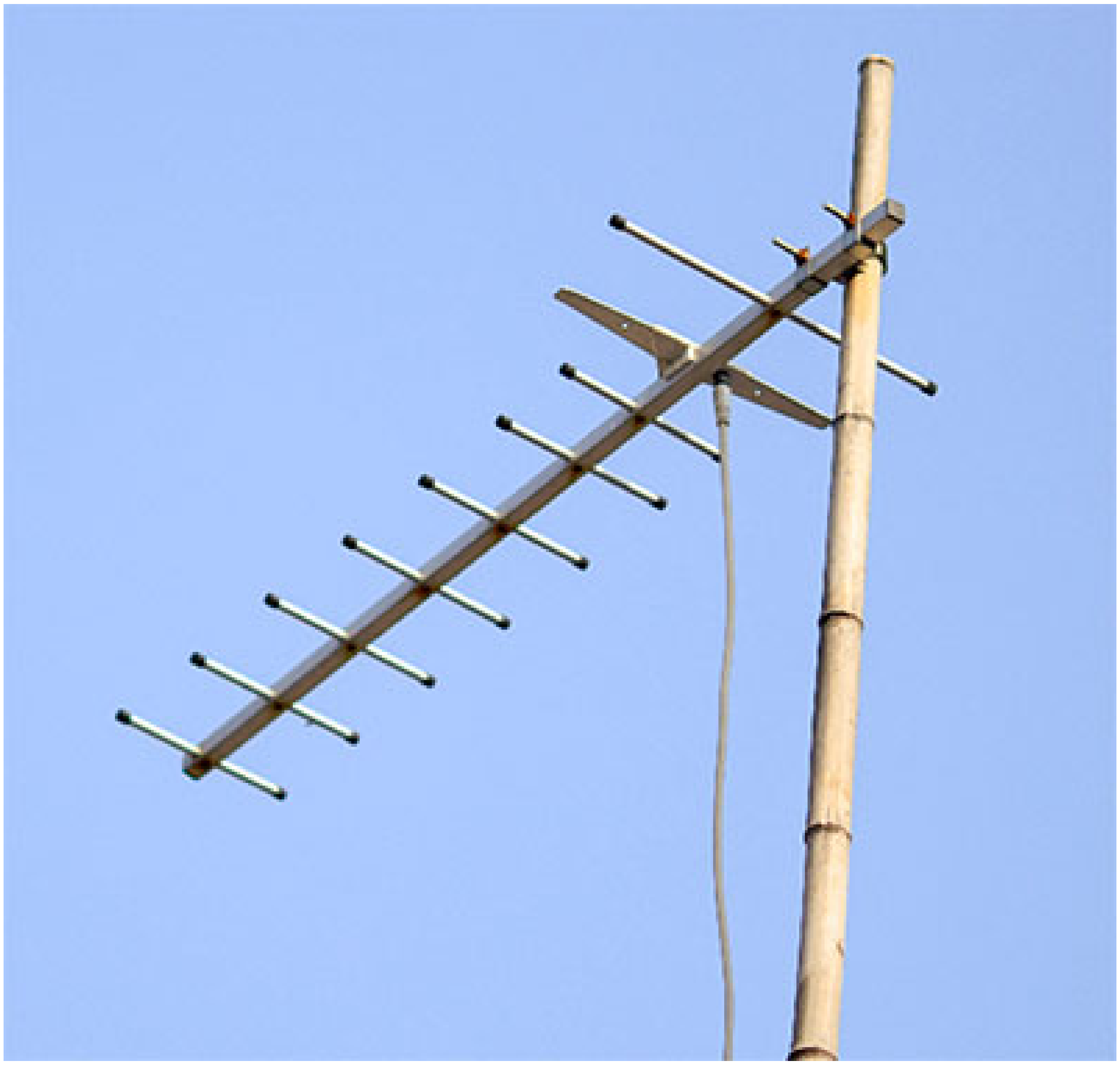


Fig. 11. A Yagi antenna [26]

2) Yagi: Yagi antennas, also known as Yagi-Uda antennas, are highly directional antennas consisting of multiple parallel elements in a line, typically one driven element, a reflector, and one or more directors. This configuration enhances the antenna's directional gain, making it highly effective for applications requiring long-range communication and focused signal reception. Yagi antennas are particularly useful in scenarios where SDR systems need to communicate over large distances or in environments with significant interference. The design and application of Yagi antennas in SDR have been explored by various researchers, highlighting their adaptability and performance in SDR systems [27].

VI. Software

A. *Windows vs Linux for SDR*

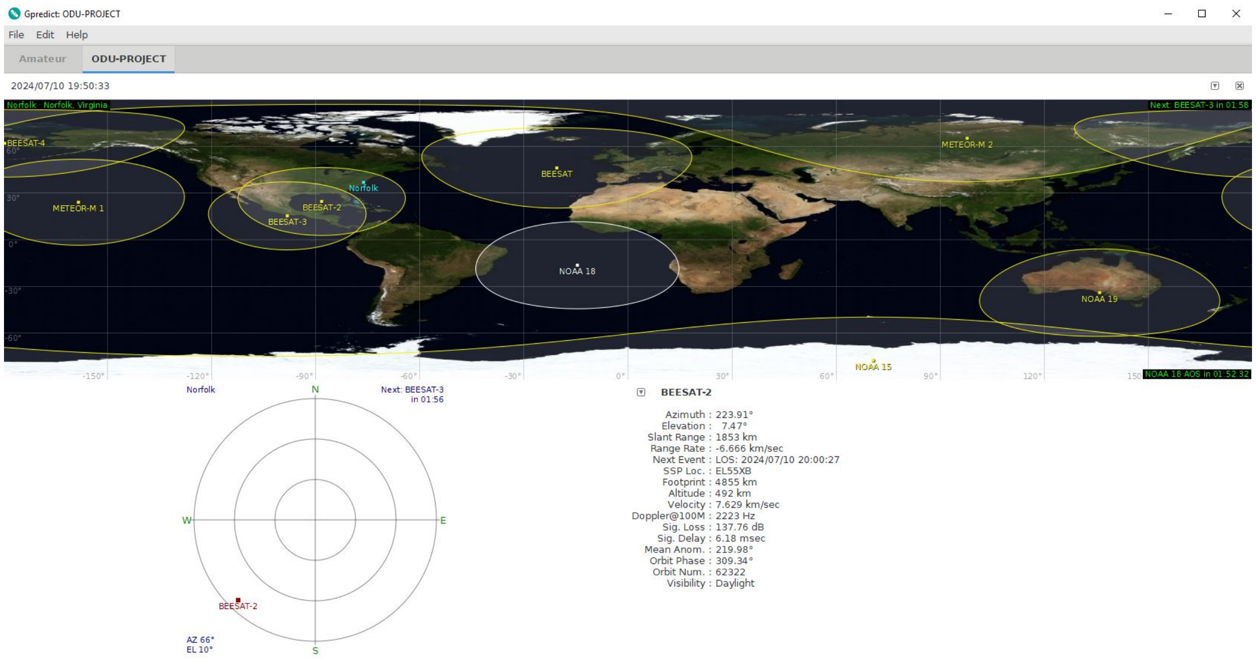
When comparing Windows and Linux for software-defined radio (SDR) applications, both platforms offer distinct advantages and challenges. Linux is often favored for its stability and security, which are crucial for maintaining uninterrupted SDR operations. Furthermore, Linux provides robust support for real-time performance optimization, including real-time scheduling policies and fine-grained system timers, which are essential for minimizing latency in SDR systems [28].

In contrast, Windows is recognized for its user-friendly interface, making it accessible for users who may not have extensive experience with command-line operations. This accessibility can be particularly beneficial for educational settings or rapid development environments where ease of use and integration with existing tools are priorities [29].

A key advantage of Linux in SDR applications is its compatibility with a wide range of open-source software, such as GNU Radio, which facilitates flexible and powerful signal processing capabilities. This open-source ecosystem enables users to customize and optimize their SDR setups without significant additional costs [30]. On the other hand, Windows can run Linux applications through compatibility layers like Services for Unix or virtual machines, allowing users to leverage Linux-based SDR tools within a Windows environment [31].

In conclusion, the choice between Windows and Linux for SDR applications largely depends on the user's requirements for stability, security, and ease of use. Linux's stability and support for real-time performance make it ideal for high-reliability applications, while Windows' user-friendly interface and compatibility options can simplify the development process.

B. *Gpredict*



```
BEESAT-2

     Azimuth : 223.91°
   Elevation :   7.47°
 Slant Range : 1853 km
  Range Rate : -6.666 km/sec
  Next Event : LOS: 2024/07/10 20:00:27
   SSP Loc. : EL55XB
   Footprint : 4855 km
    Altitude : 492 km
    Velocity : 7.629 km/sec
Doppler@100M : 2223 Hz
   Sig. Loss : 137.76 dB
  Sig. Delay : 6.18 msec
 Mean Anom. : 219.98°
 Orbit Phase : 309.34°
 Orbit Num. : 62322
  Visibility : Daylight
```

Fig. 12. Gpredict Software

Gpredict is a sophisticated software tool designed for real-time satellite tracking and orbit prediction. It enables users to monitor multiple satellites simultaneously, providing visual data through various formats such as maps, tables, and polar plots. This functionality is particularly advantageous for amateur radio operators and professionals who use Software-Defined Radio (SDR) for satellite communication and data acquisition. Gpredict allows for the creation of customizable modules that can be configured to track satellites based on different observer locations, which enhances its utility for SDR users by ensuring precise frequency and timing adjustments for optimal signal reception [32].

The integration of Gpredict with SDR systems can significantly improve the efficiency of satellite communication operations. By offering accurate predictions of satellite passes and providing detailed pass information, Gpredict helps users optimize their tracking setups. This is crucial for capturing high-quality data and maintaining stable communication links with satellites. The software's open-source nature, licensed under the GNU General Public License, encourages customization and adaptation to meet specific user requirements, further enhancing its appeal to the SDR community [32].

Gpredict is a powerful tool that enhances the functionality of Software-Defined Radio (SDR) setups and physical antenna controllers. When used as an SDR controller, Gpredict provides real-time satellite tracking and precise frequency adjustments, ensuring that the SDR is tuned accurately to receive signals from satellites at the correct times. This optimization is crucial for achieving high-quality data reception and maintaining stable communication links.

As a physical antenna controller, Gpredict can interface with rotor hardware to automatically adjust the antenna's position based on the satellite's trajectory. This automation allows for continuous tracking of moving satellites, ensuring that the antenna is always pointed at the optimal angle for signal acquisition. This feature is particularly beneficial for users who need to track multiple satellites or conduct long-duration communication sessions without manual intervention [32].

By combining these capabilities, Gpredict serves as a comprehensive solution for satellite communication enthusiasts, providing seamless integration of SDR and antenna control functionalities. This integration allows for more efficient and effective satellite tracking and communication, making Gpredict an invaluable tool for both amateur and professional users in the field.

C. GQRX

Gqrx is an open-source software defined radio (SDR) receiver powered by the GNU Radio and Qt graphical toolkit, designed for use with a variety of SDR hardware. It operates as a general-purpose radio receiver that can demodulate AM, FM, SSB, and CW signals. Gqrx provides a user-friendly graphical interface that allows users to visualize the spectrum and waterfall displays, facilitating the exploration of the radio frequency (RF) spectrum. The software supports various SDR hardware, including the widely used RTL-SDR dongles, HackRF, USRP devices, and more. This versatility makes Gqrx a popular choice among hobbyists, researchers, and radio enthusiasts for activities such as listening to commercial radio broadcasts, amateur radio communications, and even tracking aircraft and ships through ADS-B and AIS signals [33].

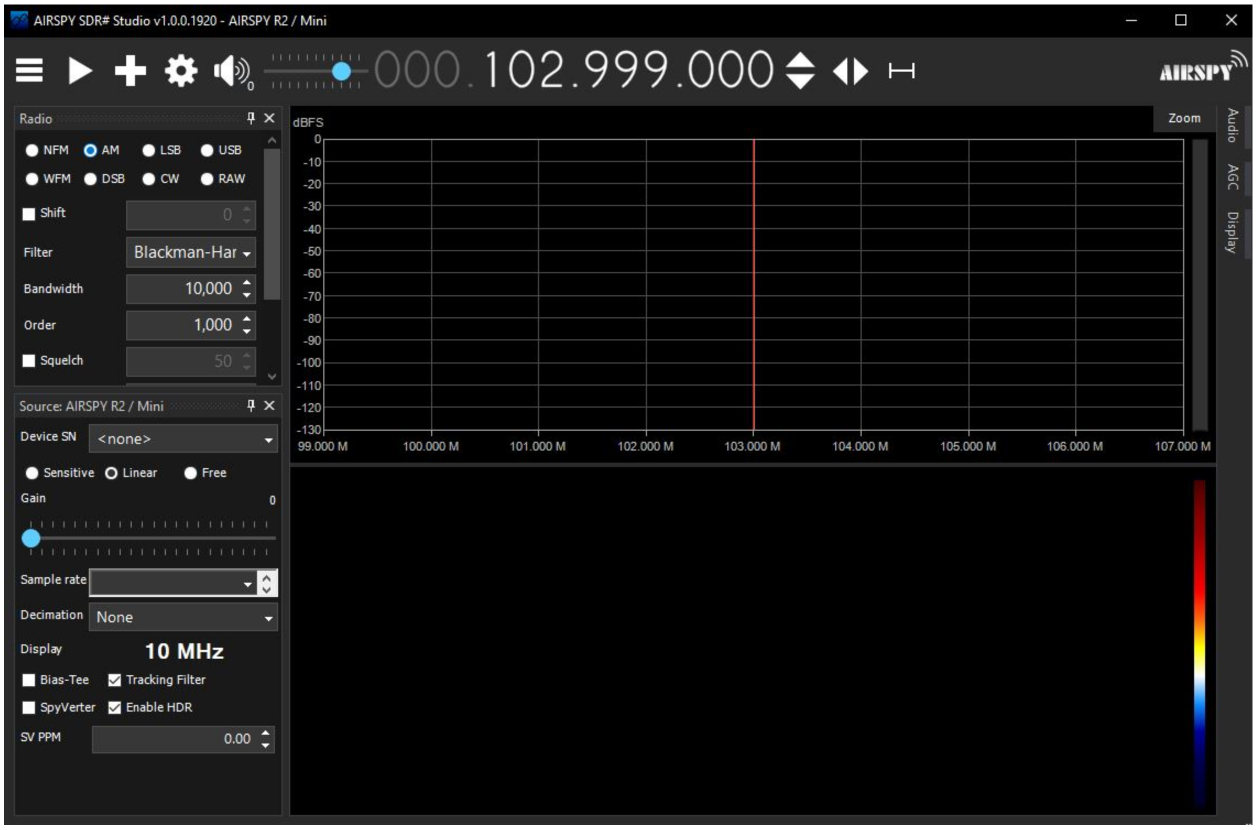
One of the key features of Gqrx is its ability to handle real-time RF signal processing, which is crucial for SDR applications. The software includes a built-in audio spectrum analyzer and a waterfall display that provides a visual representation of the signal's frequency domain. This feature is particularly useful for identifying and analyzing different types of signals and interference patterns. Additionally, Gqrx supports remote operation through network streaming, allowing users to access and control their SDR receivers from different locations. This capability is enhanced by the software's compatibility with various network protocols, making it an ideal tool for remote sensing and monitoring applications [33].

Gqrx also excels in its capability to record and playback IQ (in-phase and quadrature) data, a feature that significantly enhances its utility for satellite communication and analysis. IQ data captures the raw signal received by the SDR, preserving all its properties for detailed post-processing and analysis. This feature is particularly beneficial for satellite enthusiasts and researchers who need to analyze signals from various satellite systems, including weather satellites, communication satellites, and navigation satellites.

Recording IQ data allows users to capture entire sessions of satellite transmissions, which can later be demodulated and decoded without the need for a live connection to the satellite. This flexibility is crucial for capturing fleeting or sporadic satellite signals, which might not be consistently available. For instance, weather satellite enthusiasts can record passes of NOAA satellites and later process the data to extract weather images and other telemetry. Similarly, those interested in amateur radio satellites (such as those in the AMSAT fleet) can record and playback transmissions to decode messages and experiment with various demodulation techniques [33].

Playback of recorded IQ data in Gqrx is straightforward, allowing users to select previously recorded files and process them as if they were live signals. This feature supports comprehensive analysis and experimentation, as users can apply different filters, demodulation methods, and decoding algorithms to the same dataset, comparing the outcomes. Additionally, this capability facilitates collaborative research and education, as recorded IQ data can be shared among users for collective analysis and learning [33].

D. SDR Sharp


AIRSPY SDR# Studio v1.0.0.1920 - AIRSPY R2 / Mini

000.102.999.000

Radio

NFM [✓ AM] LSB USB WFM DSB CW RAW

Shift: 0

Filter: Blackman-Har

Bandwidth: 10,000

Order: 1,000

Squelch: 50

Source: AIRSPY R2 / Mini

Device SN: <none>

Sensitive [✓ Linear] Free

Gain: 0

Sample rate:

Decimation: None

Display: 10 MHz

☐ Bias-Tee ☑ Tracking Filter

☐ SpyVerter ☑ Enable HDR

SV PPM: 0.00

Fig. 13. SDR Sharp Software

SDRSharp is a comprehensive and high-performance software defined radio (SDR) application tailored for use with RTL-SDR dongles and high-performance AIRSPY devices. This freeware software is distinguished by its continuous updates and extensive customization options, which include a wide range of plugins designed to enhance its functionality and user experience. SDRSharp's primary appeal lies in its accessibility, making it suitable even for users with minimal technical expertise, who can quickly start using the software without a traditional installation process. Users simply extract the zipped content to any directory and begin working with the included files and plugins, which are automatically recognized and integrated into the application [34].

SDRSharp is renowned for its versatility and the broad spectrum of features it offers. The software supports various devices and enables advanced functionalities such as synchronous demodulation, anti-fading, broadband noise filtering, and narrowband noise cancellation. These features are crucial for achieving high-quality signal reception and processing.

Additionally, SDRSharp's plugins, like the DF8RY Databridge and CSVUserlistBrowser, extend its capabilities by allowing users to manage and utilize numerous radio frequency databases, providing a user-friendly interface for tuning and tracking signals. The software's continuous evolution, marked by frequent updates and the incorporation of the latest algorithmic refinements, ensures that it remains at the forefront of SDR technology [34].

In summary, SDRSharp stands out as a powerful, user-friendly SDR application that caters to both novice and advanced users through its ease of use, extensive customization options, and ongoing development. Its ability to integrate a wide range of plugins and support multiple devices makes it a versatile tool in the field of software-defined radio.

E. Audacity

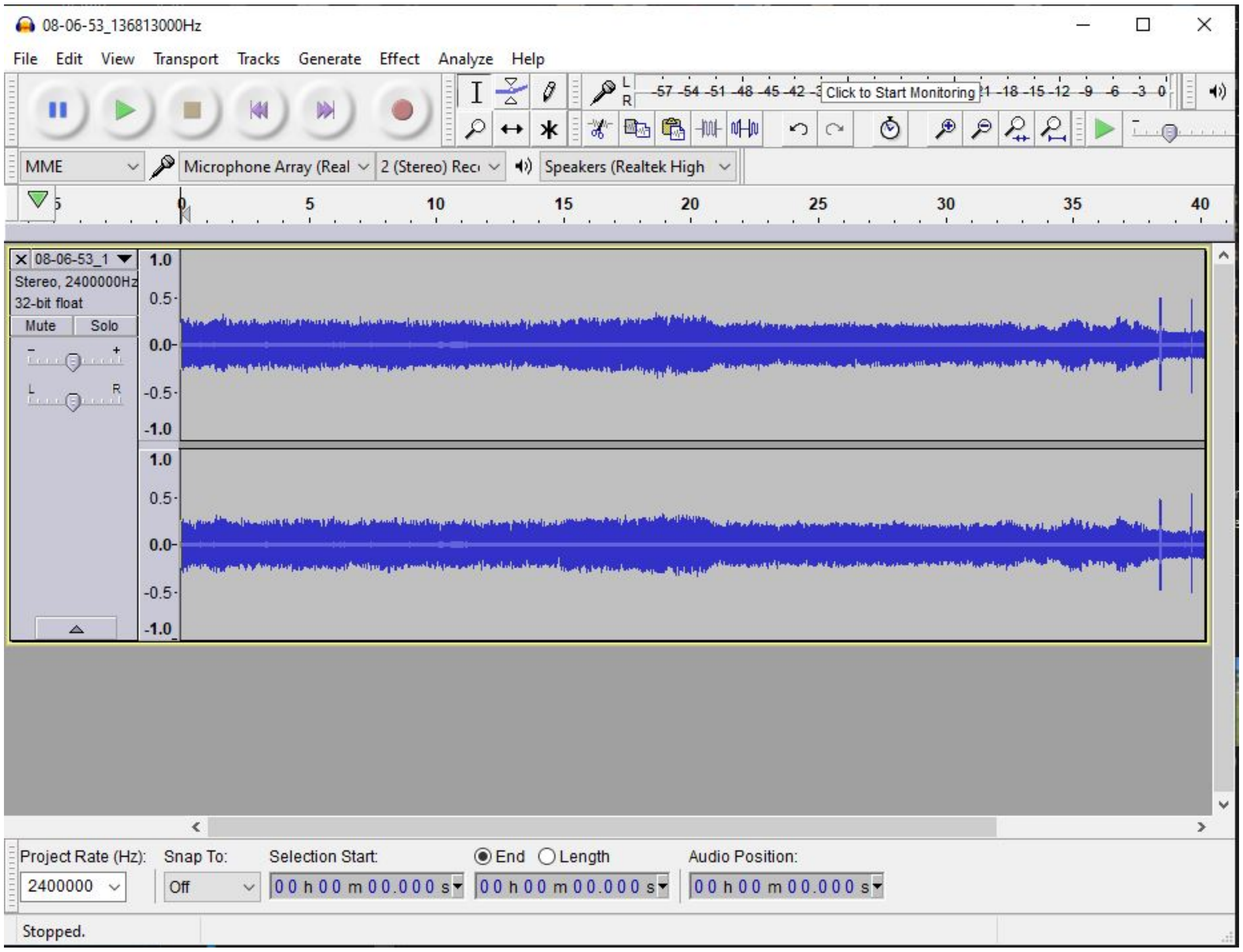

08-06-53_136813000Hz
File Edit View Transport Tracks Generate Effect Analyze Help
Click to Start Monitoring
MME
Microphone Array (Real
2 (Stereo) Reco
Speakers (Realtek High
08-06-53_1
Stereo, 2400000Hz
32-bit float
Mute Solo
Project Rate (Hz): 2400000
Snap To: Off
Selection Start: 00 h 00 m 00.000 s
End Length
00 h 00 m 00.000 s
Audio Position: 00 h 00 m 00.000 s
Stopped.

Fig. 14. Audacity Software

Audacity, an open-source audio editing software, is known for its extensive capabilities in audio manipulation and analysis, making it a useful tool in SDR applications, particularly in decoding satellite signals. The flexibility and broad range of features offered by Audacity allow users to record, filter, and analyze audio signals, which are essential steps in processing the radio frequencies received from satellites.

In the context of decoding satellite signals, Audacity can be integrated with SDR hardware to handle the audio output from SDR devices. For instance, signals from satellites such as the International Space Station (ISS) or National Oceanic and Atmospheric Administration (NOAA) satellites can be captured using SDR hardware like USRP or RTL-SDR, and then processed in Audacity. The software's ability to filter noise, adjust frequencies, and analyze spectrums facilitates the identification and extraction of useful data from the received signals [35].

Audacity's spectral analysis tools can be particularly beneficial in SDR applications where signal integrity and clarity are crucial. By using these tools, researchers can visually inspect the signal's frequency components and make necessary adjustments to enhance the signal-to-noise ratio. This capability is critical when dealing with the Doppler shift and other distortions inherent in satellite communications [3].

Additionally, the integration of Audacity in educational and research settings for SDR applications provides a cost-effective method for studying and prototyping new algorithms and techniques in satellite signal processing. This is especially relevant in environments where budget constraints limit the availability of high-end commercial software and hardware solutions [36].

F. WxToIMG

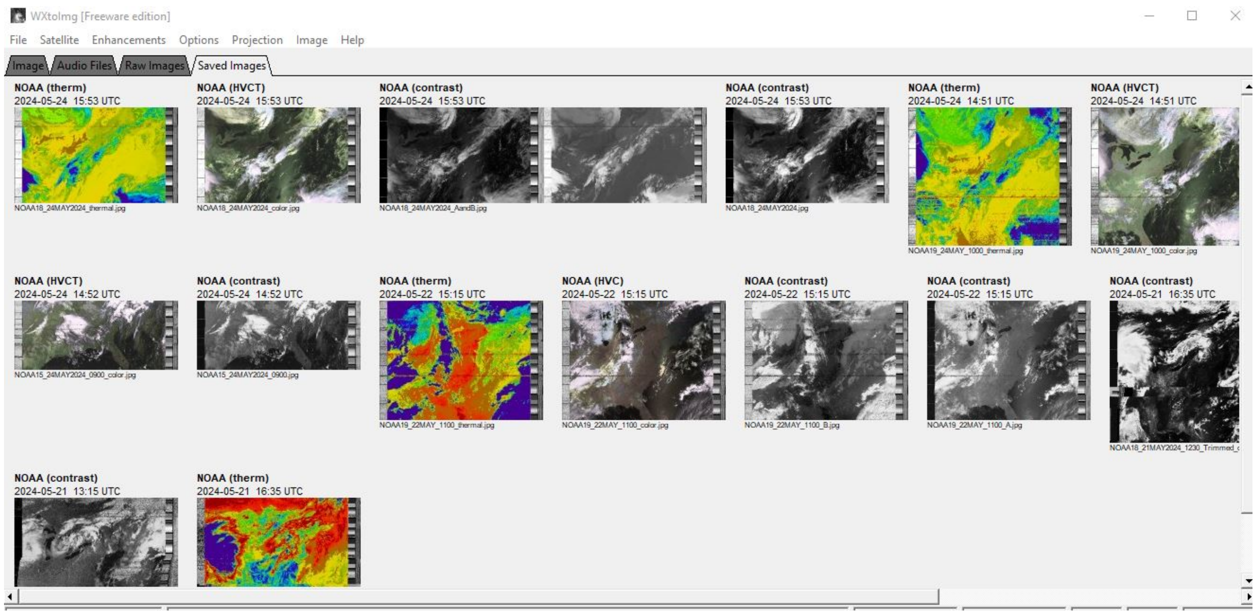

WXtoImg [Freeware edition]
File Satellite Enhancements Options Projection Image Help
Image Audio Files Raw Images Saved Images
NOAA (therm) 2024-05-24 15:53 UTC
NOAA18_24MAY2024_thermal.jpg
NOAA (HVCT) 2024-05-24 15:53 UTC
NOAA18_24MAY2024_color.jpg
NOAA (contrast) 2024-05-24 15:53 UTC
NOAA18_24MAY2024_AandB.jpg
NOAA (contrast) 2024-05-24 15:53 UTC
NOAA18_24MAY2024.jpg
NOAA (therm) 2024-05-24 14:51 UTC
NOAA19_24MAY_1000_thermal.jpg
NOAA (HVCT) 2024-05-24 14:51 UTC
NOAA19_24MAY_1000_color.jpg
NOAA (HVCT) 2024-05-24 14:52 UTC
NOAA15_24MAY2024_0900_color.jpg
NOAA (contrast) 2024-05-24 14:52 UTC
NOAA15_24MAY2024_0900.jpg
NOAA (therm) 2024-05-22 15:15 UTC
NOAA19_22MAY_1100_thermal.jpg
NOAA (HVC) 2024-05-22 15:15 UTC
NOAA19_22MAY_1100_color.jpg
NOAA (contrast) 2024-05-22 15:15 UTC
NOAA19_22MAY_1100_B.jpg
NOAA (contrast) 2024-05-22 15:15 UTC
NOAA19_22MAY_1100_A.jpg
NOAA (contrast) 2024-05-21 16:35 UTC
NOAA18_21MAY2024_1230_Trimmed_
NOAA (contrast) 2024-05-21 13:15 UTC
NOAA (therm) 2024-05-21 16:35 UTC

Fig. 15. WxToIMG Software

WXtoImg is a comprehensive software tool designed for decoding weather satellite signals. It offers a fully automated process for receiving, decoding, and displaying images from APT (Automatic Picture Transmission) and WEFAX (Weather Facsimile) signals. The software supports multiple operating systems including Windows, Linux, and macOS, and integrates features like real-time decoding, map overlays, color enhancements, and 3-D image generation. WXtoImg is particularly valued for its capability to transform raw satellite data into visually rich and informative weather images.

In the context of Software Defined Radio (SDR) applications, WXtoImg is often used to decode signals from NOAA weather satellites. SDR devices, such as RTL-SDR, can capture satellite signals, which are then processed and decoded using WXtoImg. This setup allows for real-time weather data acquisition, which is crucial for meteorological studies and disaster management. The software's ability to handle high-resolution images and perform advanced image corrections makes it an essential tool for academic research and practical applications in weather monitoring and analysis [37].

The software integrates real-time decoding, map overlays, color enhancements, and 3-D image generation, transforming raw satellite data into visually rich weather images. This is particularly effective when used with Software Defined Radio (SDR) devices like RTL-SDR, which capture satellite signals for WXtoImg to process and decode. SDR devices, such as RTL-SDR, capture the raw radio signals transmitted by NOAA satellites. These signals, typically at 137 MHz, contain APT data which is modulated in FM (Frequency Modulation) format. SDR hardware like RTL-SDR can sample these signals, converting them into a digital form that can be processed by software on a computer [38].

To decode these signals, the software takes the digitized signals from the SDR and demodulates them to extract the image data. This involves correcting for frequency shifts (Doppler effect) due to the satellite's motion and filtering out noise. Advanced algorithms within WXtoImg help in mitigating the Doppler effect and reducing noise in the images for better clarity [10]. Once the image data is extracted, WXtoImg processes it to generate weather images. This includes applying map overlays to show geographical information, enhancing colors to distinguish different weather patterns, and even generating 3-D visualizations of the data. These enhancements help in interpreting the satellite data effectively [37].

WXtoImg's integration with SDR technology provides a cost-effective solution for obtaining high-quality weather satellite images. Researchers have demonstrated the effectiveness of using inexpensive SDR setups combined with WXtoImg for receiving and processing signals from NOAA satellites. This approach not only reduces the cost of setting up satellite receiving stations but also simplifies the process of acquiring and analyzing satellite imagery. By utilizing WXtoImg, users can enhance the capabilities of their SDR systems to capture detailed and accurate weather data [39].

G. GNURadio

GNU Radio is a free and open-source software development toolkit that provides signal processing blocks to implement software-defined radios (SDR). It is utilized by academia, government, industry, hobbyists, and researchers for both prototyping and real-world radio systems. By leveraging the capabilities of GNU Radio, users can process real-time data, perform signal analysis, and build communication systems without the need for traditional, hardware-based radio systems.

GNU Radio's modular architecture allows users to create complex signal processing systems by connecting various blocks within a graphical interface or through Python scripting. This flexibility makes it an invaluable tool for developing and testing new communication protocols, performing wireless experimentation, and exploring concepts in radio frequency (RF) communications. Its ability to interface with various hardware, such as USRP devices, further extends its versatility in SDR applications [40].

Getting started with GNURadio to work with SDR and IQ data is relatively simple. With minimal effort, you can set up and display a waterfall plot from previously recorded NOAA IQ data. The necessary blocks to achieve this are shown below. These blocks allow you to process and visualize the IQ data, making it easy to analyze the recorded signals and gain insights into the captured radio frequencies.

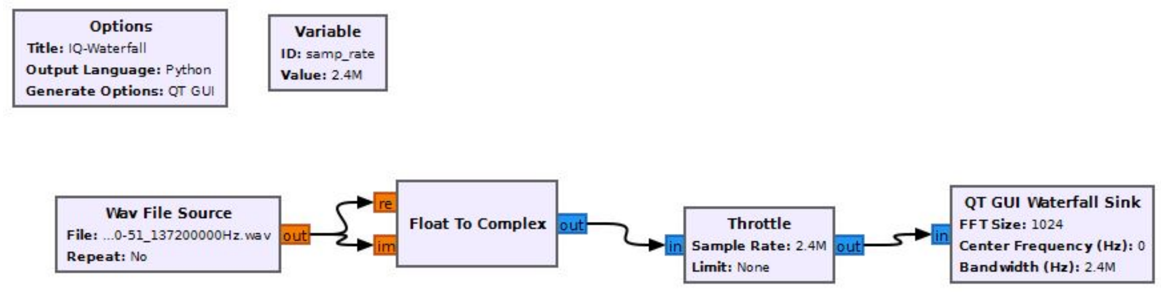


Fig. 16. GNU Radio Companion(GRC) file that generates a waterfall of pre-recorded IQ Data

The following waterfall display is just a small portion of what is possible for users of GNURadio. Whether you are a beginner or an experienced user, GNURadio's user-friendly interface and powerful capabilities streamline the process of working with SDR and IQ data.

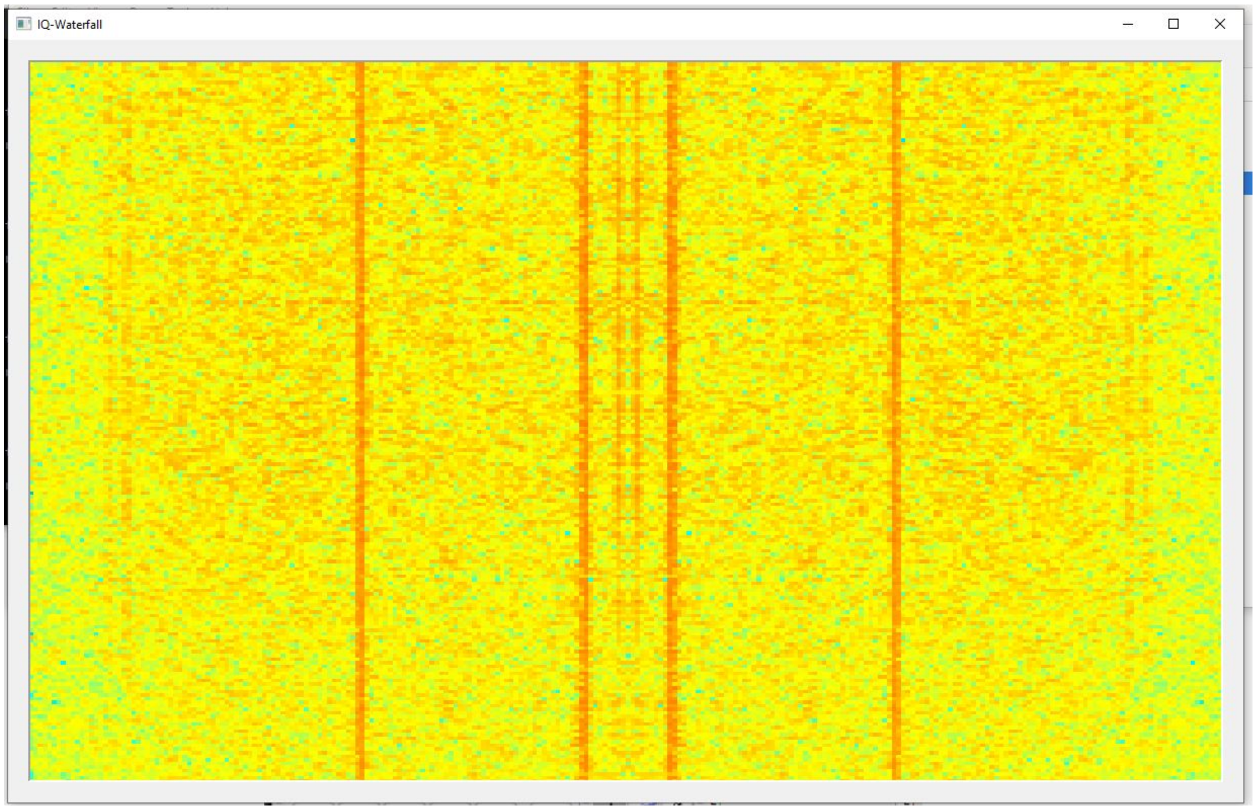


Fig. 17. NOAA Signal Waterfall generated with previous GRC

1) UbuntuVM: UbuntuVM is an invaluable tool for those starting with GNU Radio and Software-Defined Radio (SDR). This virtual machine environment offers a pre-configured platform that simplifies the often complex process of installing and using GNU Radio, which can be challenging due to various dependencies and potential compatibility issues.

One of the primary benefits of using UbuntuVM is that it comes with GNU Radio pre-installed. This feature allows users, particularly beginners, to bypass the cumbersome installation and configuration processes that can be daunting for those unfamiliar with software dependencies and system setup. Additionally, running UbuntuVM through VirtualBox enables users to utilize GNU Radio on various operating systems, including Windows, macOS, and other Linux distributions, thus providing flexibility beyond the limitations of a native OS.

The virtual machine environment of UbuntuVM also offers the advantage of isolation, which helps maintain the stability of the host system. This separation ensures that any configuration issues or software conflicts that may arise during experimentation with SDR tools do not affect the host system. Moreover, UbuntuVM's user-friendly interface, especially when used with VirtualBox, makes managing the virtual machine straightforward. Tasks such as starting and stopping the VM, allocating system resources, and interacting with GNU Radio are simplified, making the setup accessible even to those with limited experience in virtual environments.

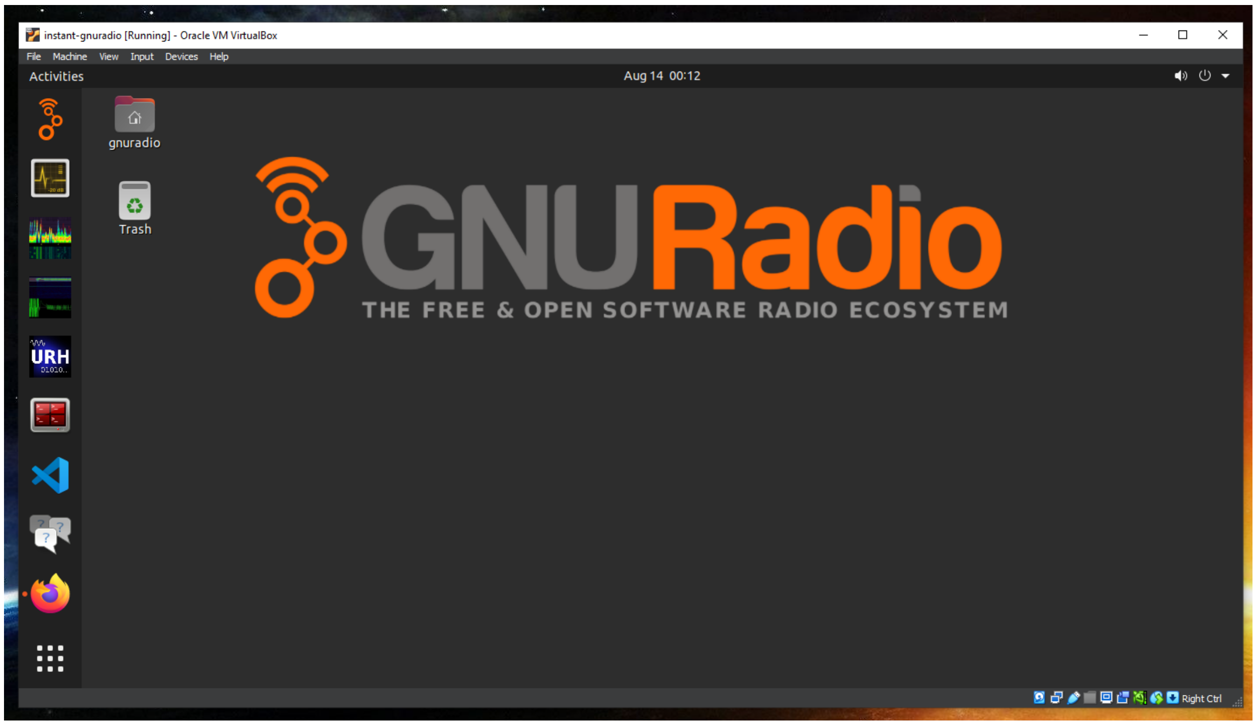

instant-gnuradio [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Aug 14 00:12
gnuradio
Trash
GNU Radio
THE FREE & OPEN SOFTWARE RADIO ECOSYSTEM
URH
Right Ctrl

Fig. 18. Using the imported UbuntuVM in VirtualBox

Setting up UbuntuVM involves installing VirtualBox on the host system, downloading and importing the UbuntuVM image from the GNU Radio Wiki, and configuring the system settings to optimize performance. The process is straightforward and well-documented, allowing users to quickly get started with GNU Radio. After launching the virtual machine, users can immediately begin working with GNU Radio to experiment with SDR applications, design flowgraphs, and decode signals using the tools provided.

UbuntuVM provides a robust and accessible environment for both beginners and experienced users in the field of software-defined radio. Its pre-configured setup, combined with the flexibility and stability of a virtual machine, makes it an ideal tool for exploring the capabilities of GNU Radio. By following the setup guidelines provided by the GNU Radio community, users can efficiently dive into SDR applications and take full advantage of the powerful features that GNU Radio offers. [41].

VII. Experimental Results

A. RTL-SDR and WxToIMG

This portion of the experiment was conducted in the Riverpointe Neighborhood located in Portsmouth Virginia. The location of where the satellite signals were received are depicted below. The open field in this area was selected for its minimal obstructions, providing optimal conditions for satellite signal reception. It is crucial that the ground station location is free from trees and buildings to minimize interference and ensure the readability of the satellite signals.

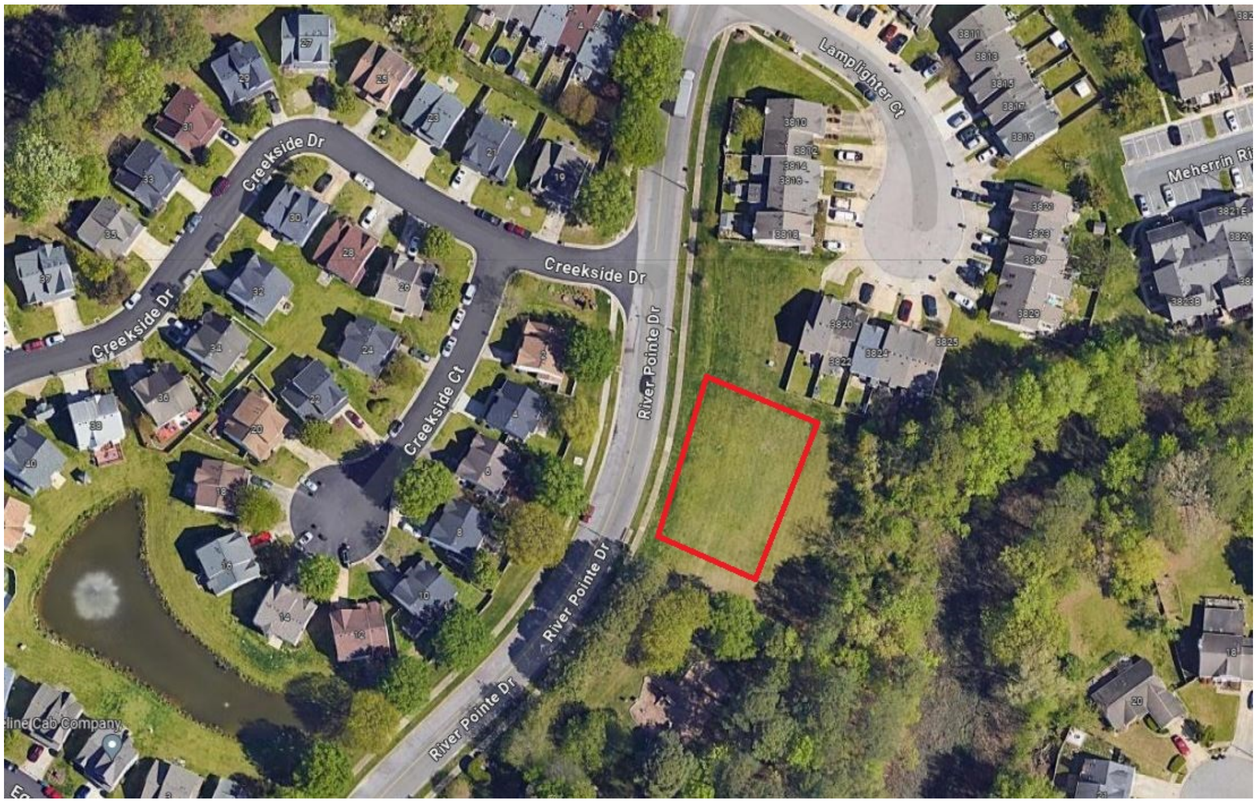


Fig. 19. Location of Ground Station marked in red.

The hardware setup for this experiment was intentionally kept simple to demonstrate the minimal equipment required to start receiving satellite signals with Software-Defined Radio (SDR). NOAA satellite signals were captured using a Simple Dipole Antenna connected to an RTL-SDR Blog 3. This RTL-SDR was linked to a Windows PC running SDRSharp, which was used to process and record the received signals.

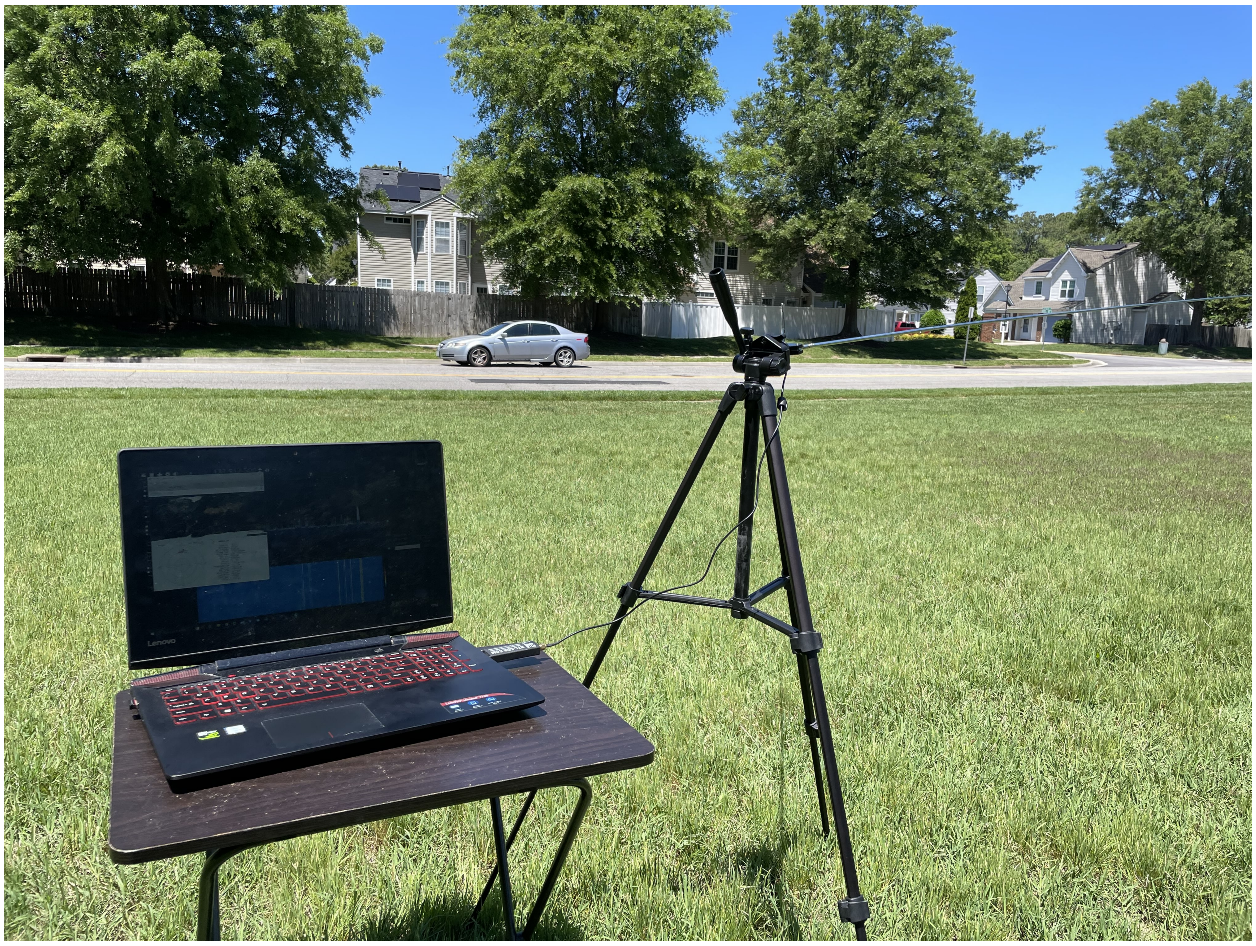
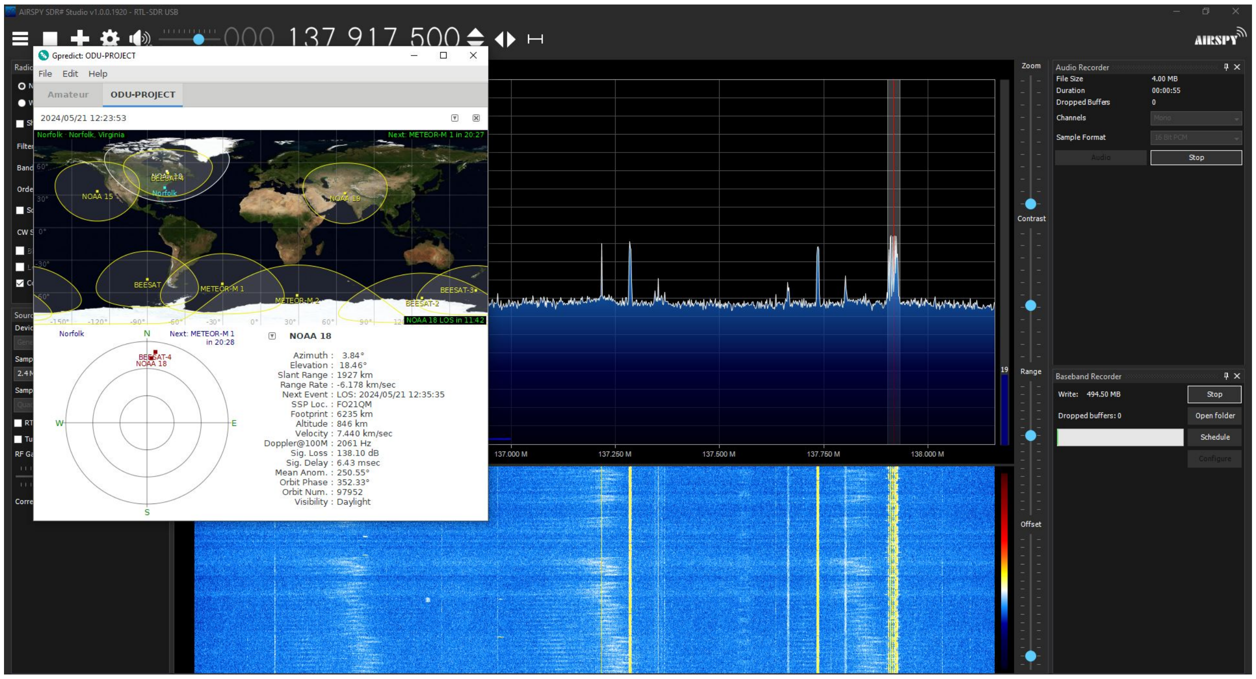


Fig. 20. Ground Station Setup.

Gpredict software was utilized to track NOAA satellites and provide necessary predictions, ensuring the system was ready to receive signals at the appropriate times. The setup process began at least five to ten minutes before each satellite pass to properly configure the antenna, SDR, and software. The image below shows this software configuration.



NOAA 18

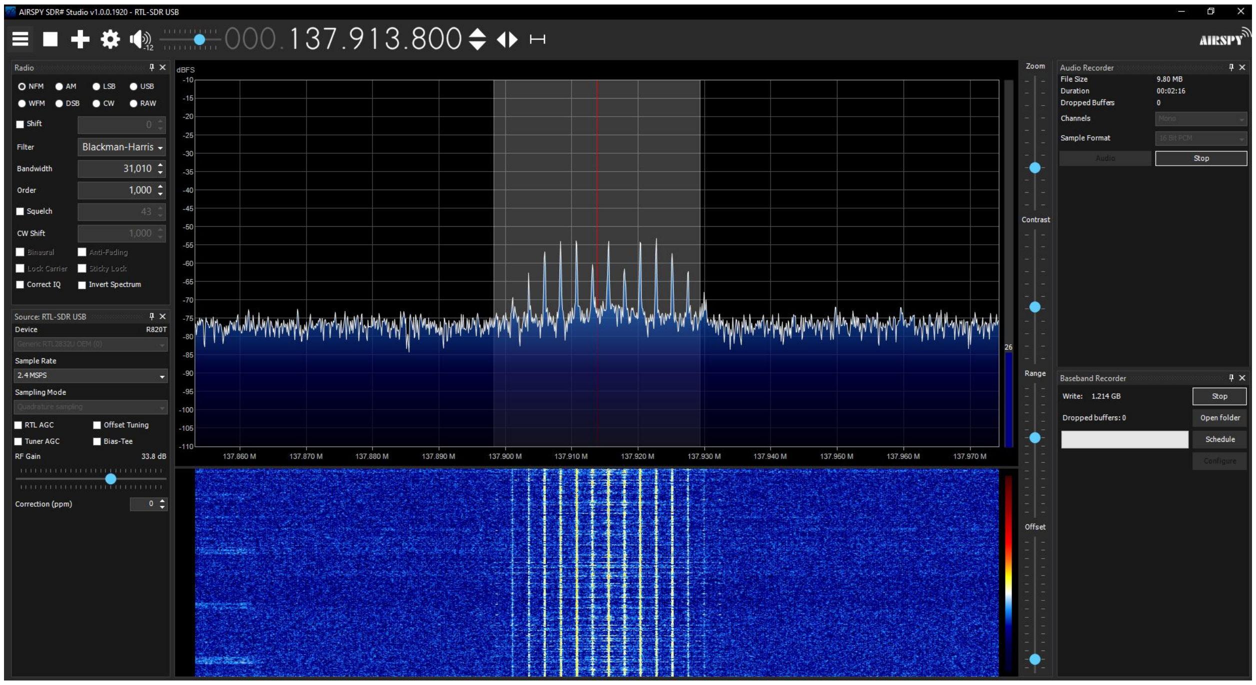
```
      Azimuth :  3.84°
    Elevation :  18.46°
  Slant Range : 1927 km
   Range Rate : -6.178 km/sec
   Next Event : LOS: 2024/05/21 12:35:35
    SSP Loc. : FO21QM
    Footprint : 6235 km
     Altitude : 846 km
     Velocity : 7.440 km/sec
Doppler@100M : 2061 Hz
    Sig. Loss : 138.10 dB
   Sig. Delay : 6.43 msec
  Mean Anom. : 250.55°
  Orbit Phase : 352.33°
   Orbit Num. : 97952
   Visibility : Daylight
```

```
File Size        4.00 MB
Duration         00:00:55
Dropped Buffers  0
```

```
Write:   494.50 MB
Dropped buffers: 0
```

Fig. 21. Using Gpredict to track NOAA Satellites.

Once the satellite reached an appropriate elevation, typically greater than 10-15 degrees, a clear signal was visible and audible via SDRSharp software. At this point, both sound and IQ recorders were activated to capture the raw data from each NOAA satellite. Below are images of SDRSharp receiving signals from NOAA 18 and NOAA 19, respectively.


AIRSPY SDR# Studio v1.0.0.1920 - RTL-SDR USB

000.137.913.800

Radio: NFM

Filter: Blackman-Harris

Bandwidth: 31,010

Order: 1,000

Squelch: 43

CW Shift: 1,000

Source: RTL-SDR USB

Device: R820T

Generic RTL2832U OEM (0)

Sample Rate: 2.4 MSPS

Sampling Mode: Quadrature sampling

RF Gain: 33.8 dB

Correction (ppm): 0

Audio Recorder

File Size: 9.80 MB

Duration: 00:02:16

Dropped Buffers: 0

Channels: Mono

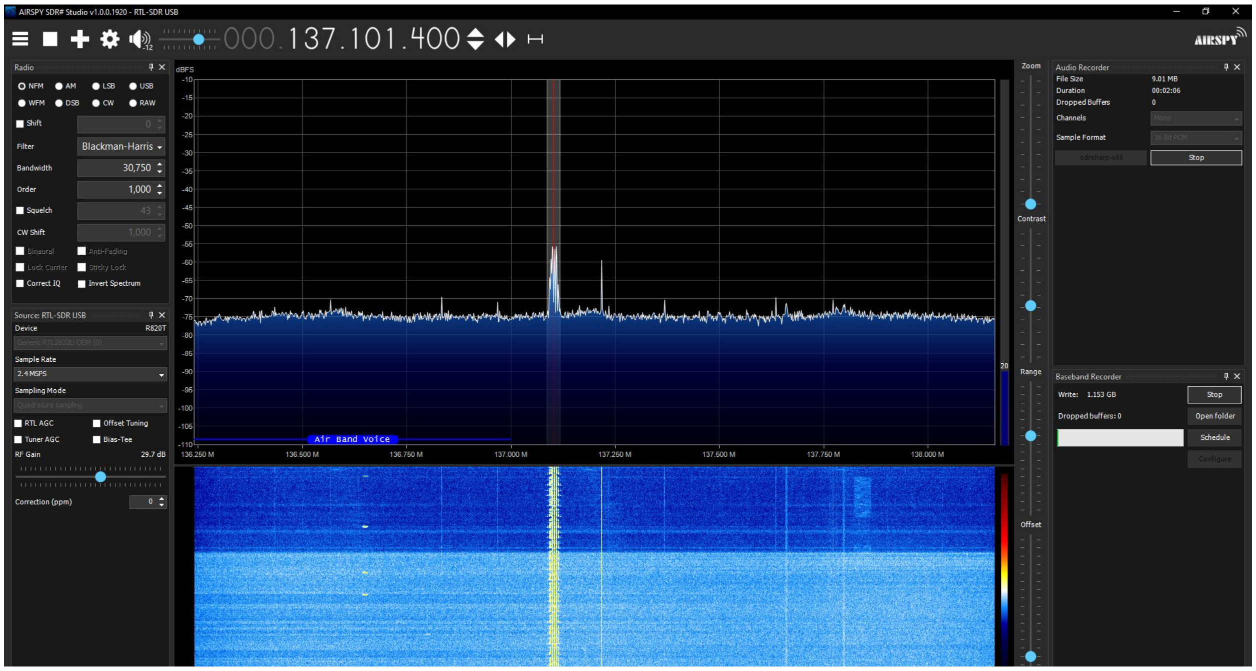
Sample Format: 16 Bit PCM

Baseband Recorder

Write: 1.214 GB

Dropped buffers: 0

Fig. 22. NOAA 18 signal depicted in SDRSharp zoomed in to focus on signal


AIRSPY SDR# Studio v1.0.0.1920 - RTL-SDR USB

000.137.101.400

Radio: NFM

Filter: Blackman-Harris

Bandwidth: 30,750

Order: 1,000

Squelch: 43

CW Shift: 1,000

Source: RTL-SDR USB

Device: R820T

Generic RTL2832U OEM (0)

Sample Rate: 2.4 MSPS

Sampling Mode: Quadrature sampling

RF Gain: 29.7 dB

Correction (ppm) 0

Audio Recorder

File Size: 9.01 MB

Duration: 00:02:06

Dropped Buffers: 0

Channels: Mono

Sample Format: 16 Bit PCM

Baseband Recorder

Write: 1.153 GB

Dropped buffers: 0

Fig. 23. NOAA 19 signal depicted in SDRSharp, zoomed out

After the satellites completed their passes, the audio and IQ files were saved for further processing. A straightforward pipeline was used to decode weather imagery from the captured data, employing

Audacity and WXtoIMG programs. Each audio recording was initially processed in Audacity, where it was normalized and resampled to 11025 Hz for compatibility with WXtoIMG. Excess noise at the beginning and end of the audio files was trimmed to reduce noise and static in the final images.

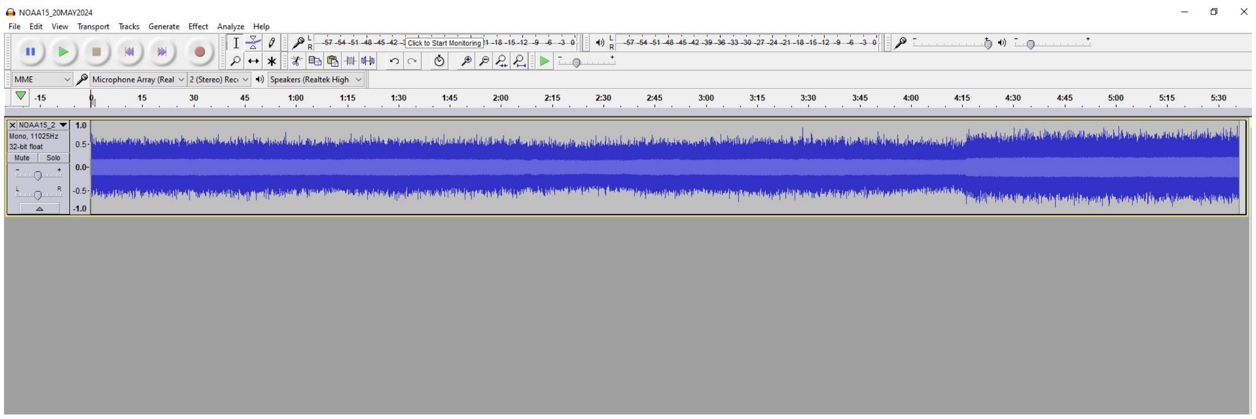


Fig. 24. NOAA 15 audio being processed in Audacity

The processed audio files were then loaded into WXtoIMG, which decoded the raw signals and generated weather imagery. WXtoIMG offers various overlay options, including color and thermal overlays, to display different types of weather data.

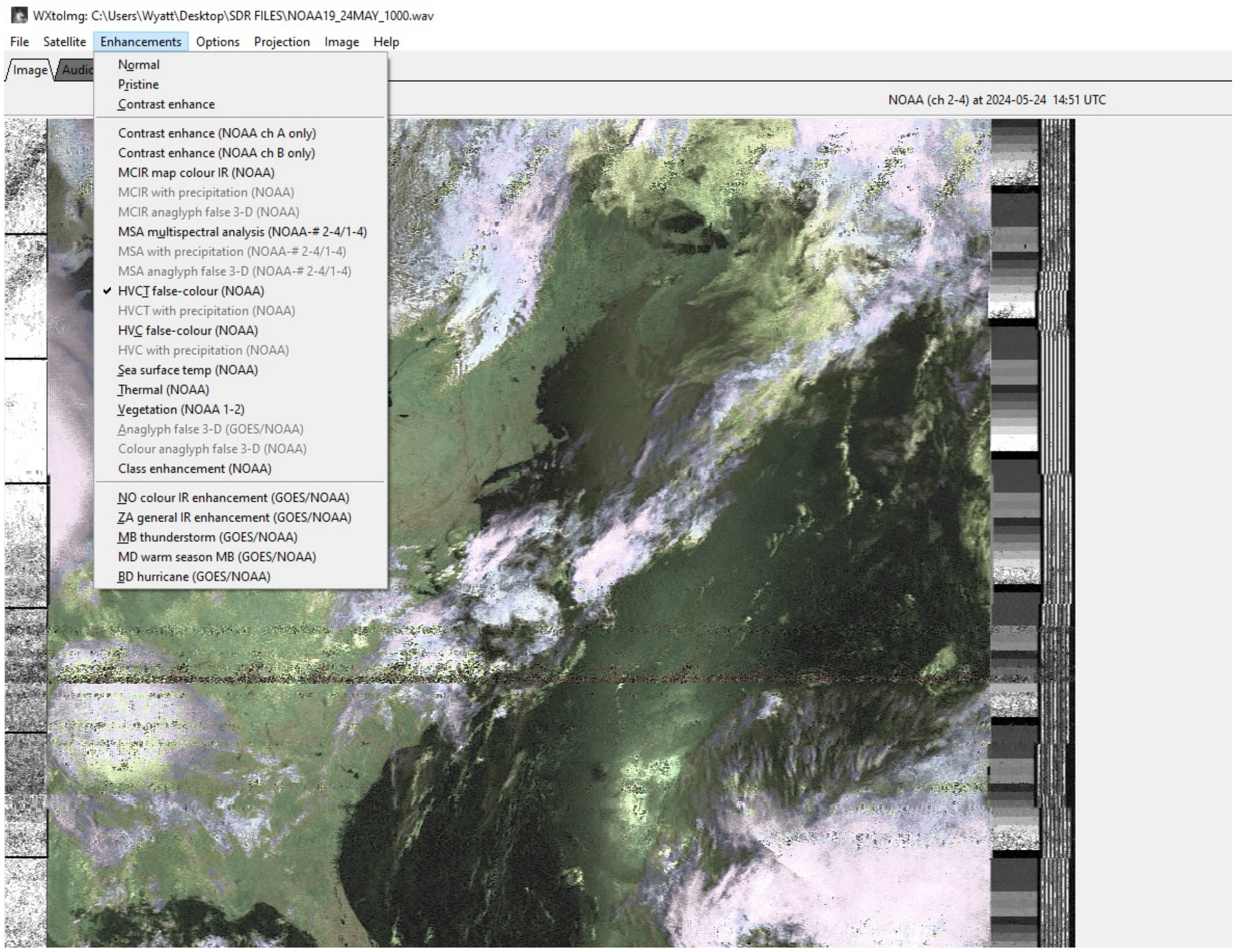

WXtoImg: C:\Users\Wyatt\Desktop\SDR FILES\NOAA19_24MAY_1000.wav
File
Satellite
Enhancements
Options
Projection
Image
Help
Image
Audio
NOAA (ch 2-4) at 2024-05-24 14:51 UTC
Normal
Pristine
Contrast enhance
Contrast enhance (NOAA ch A only)
Contrast enhance (NOAA ch B only)
MCIR map colour IR (NOAA)
MCIR with precipitation (NOAA)
MCIR anaglyph false 3-D (NOAA)
MSA multispectral analysis (NOAA-# 2-4/1-4)
MSA with precipitation (NOAA-# 2-4/1-4)
MSA anaglyph false 3-D (NOAA-# 2-4/1-4)
✓ HVCT false-colour (NOAA)
HVCT with precipitation (NOAA)
HVC false-colour (NOAA)
HVC with precipitation (NOAA)
Sea surface temp (NOAA)
Thermal (NOAA)
Vegetation (NOAA 1-2)
Anaglyph false 3-D (GOES/NOAA)
Colour anaglyph false 3-D (NOAA)
Class enhancement (NOAA)
NO colour IR enhancement (GOES/NOAA)
ZA general IR enhancement (GOES/NOAA)
MB thunderstorm (GOES/NOAA)
MD warm season MB (GOES/NOAA)
BD hurricane (GOES/NOAA)

Fig. 25. Processed Audio converted to image with WXtoIMG

The following subsections display the results from the three active NOAA Satellites, NOAA 15, NOAA 18 and NOAA 19.

1) NOAA 15: Below are the results from decoding NOAA 15 signals.

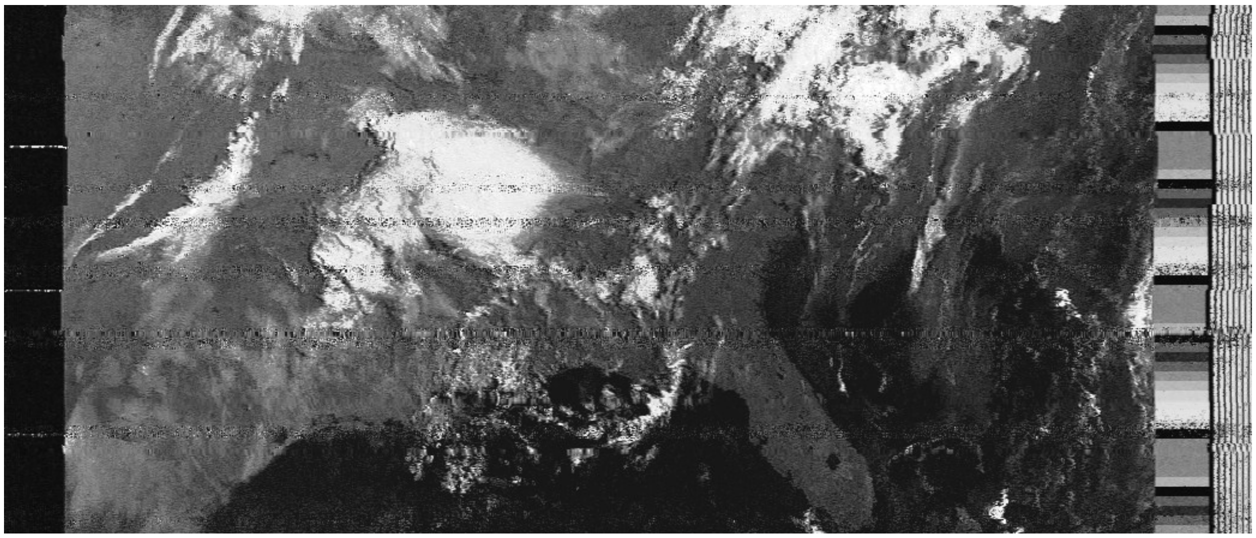


Fig. 26. NOAA 15 Decoded Image

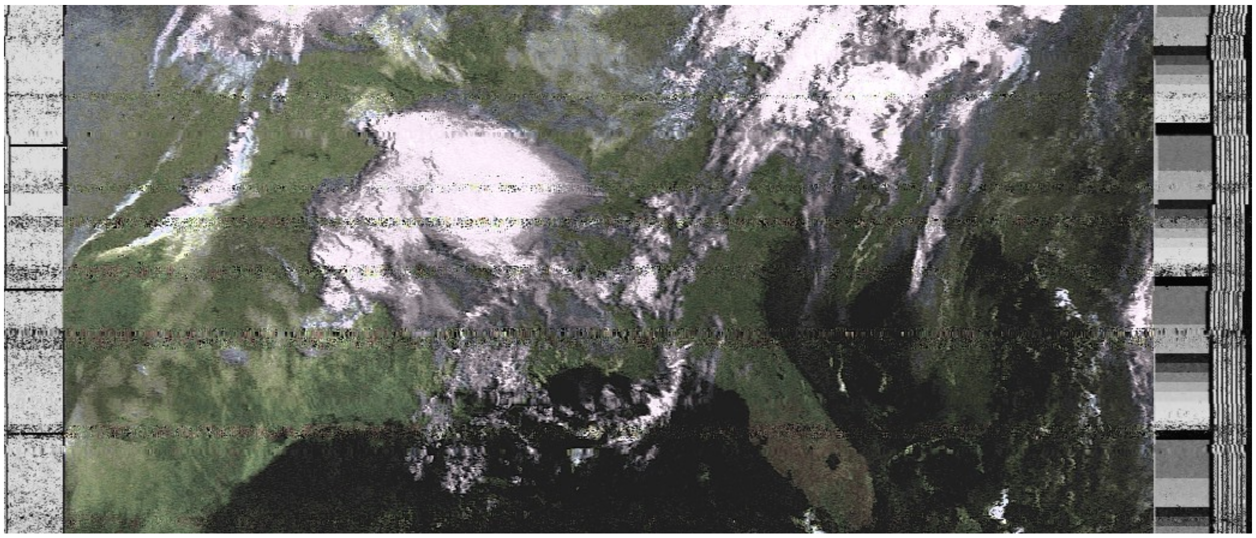


Fig. 27. NOAA 15 Decoded Image with colored overlay

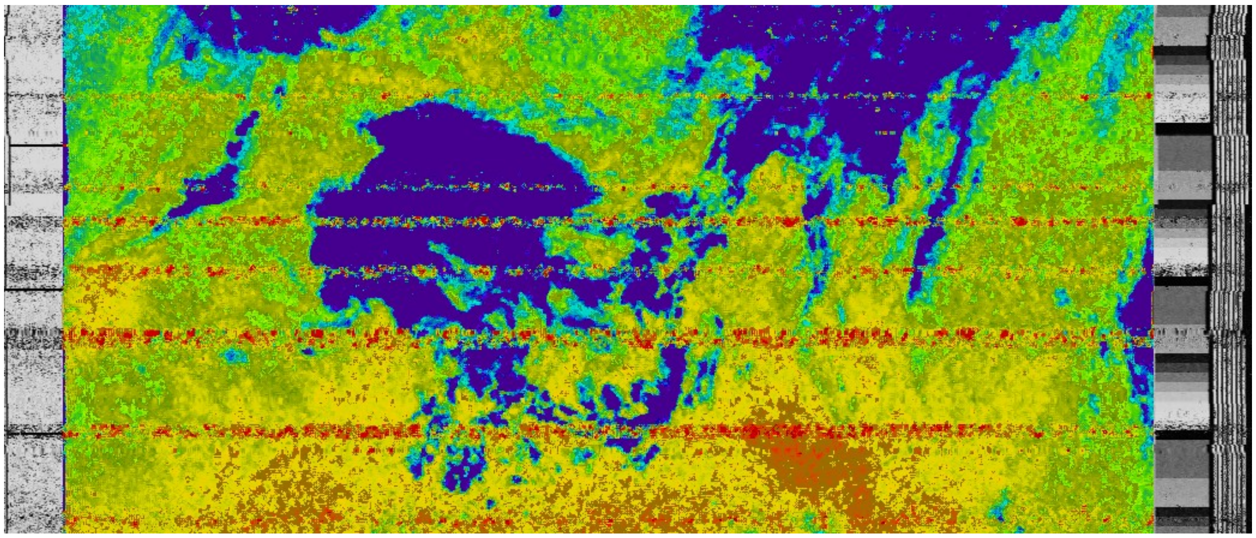


Fig. 28. NOAA 15 Decoded Image with thermal overlay

2) NOAA 18: Below are the results from decoding NOAA 18 signals.

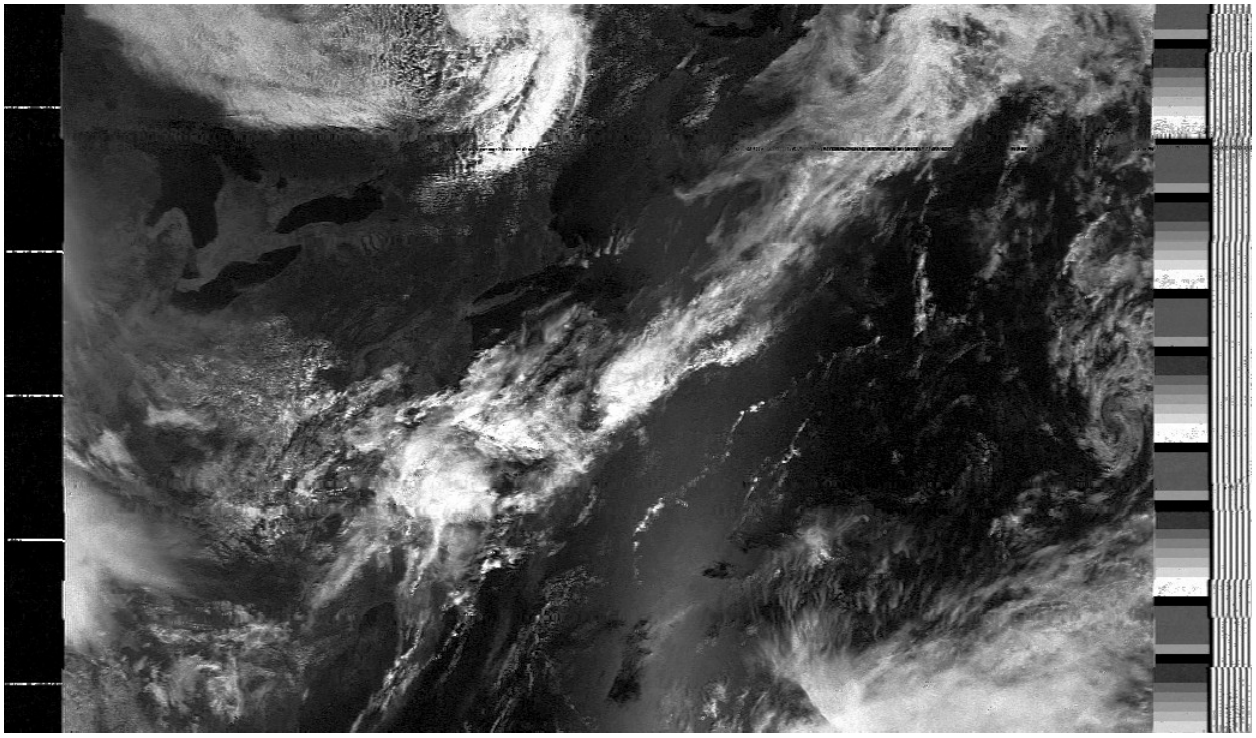


Fig. 29. NOAA 18 Decoded Image

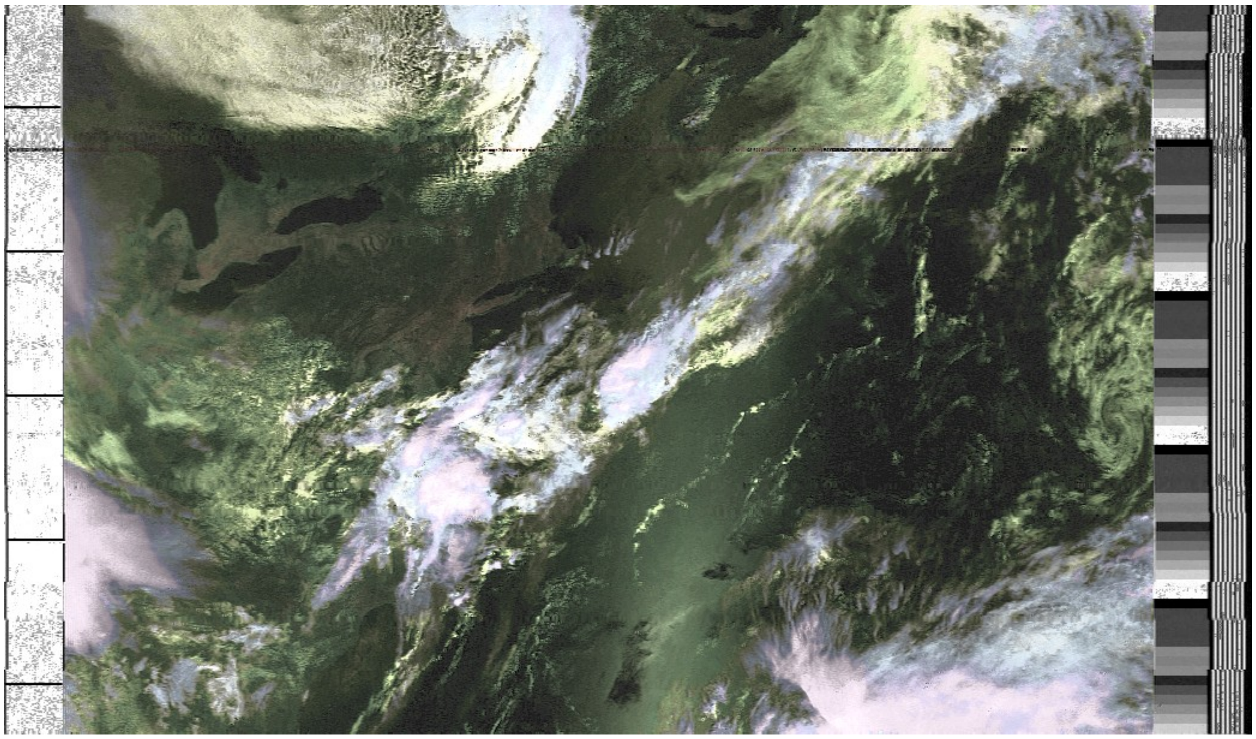


Fig. 30. NOAA 18 Decoded Image with colored overlay

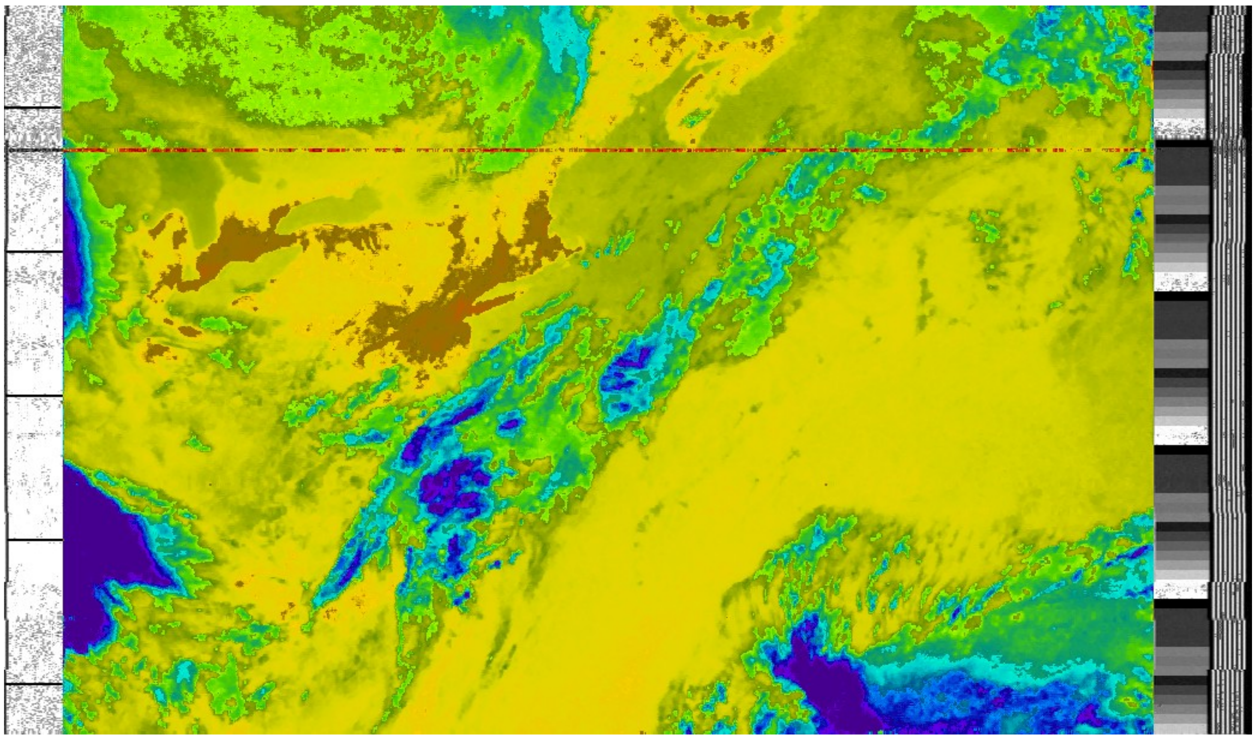


Fig. 31. NOAA 18 Decoded Image with thermal overlay

3) NOAA 19: Below are the results from decoding NOAA 19 signals.

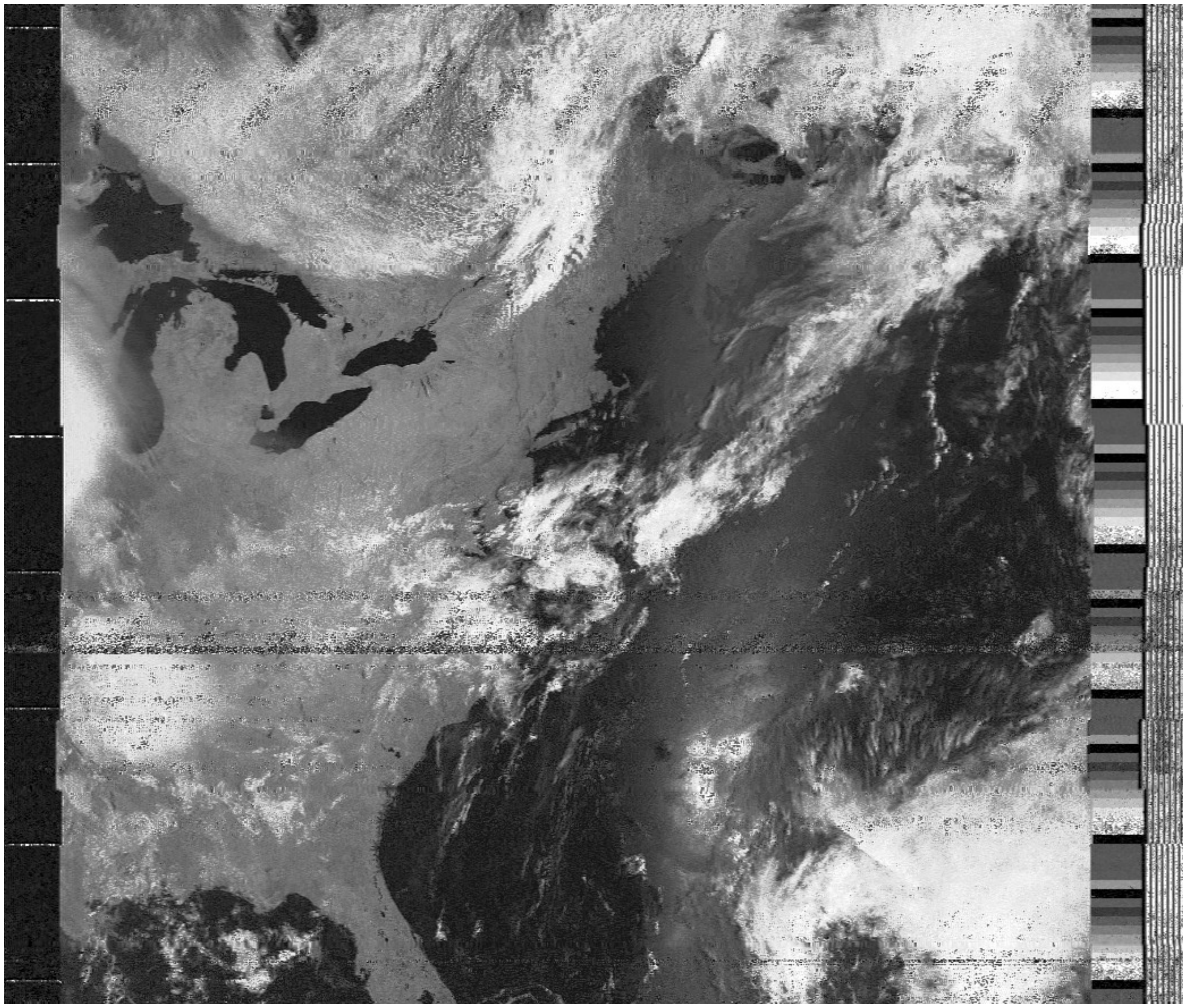


Fig. 32. NOAA 19 Decoded Image

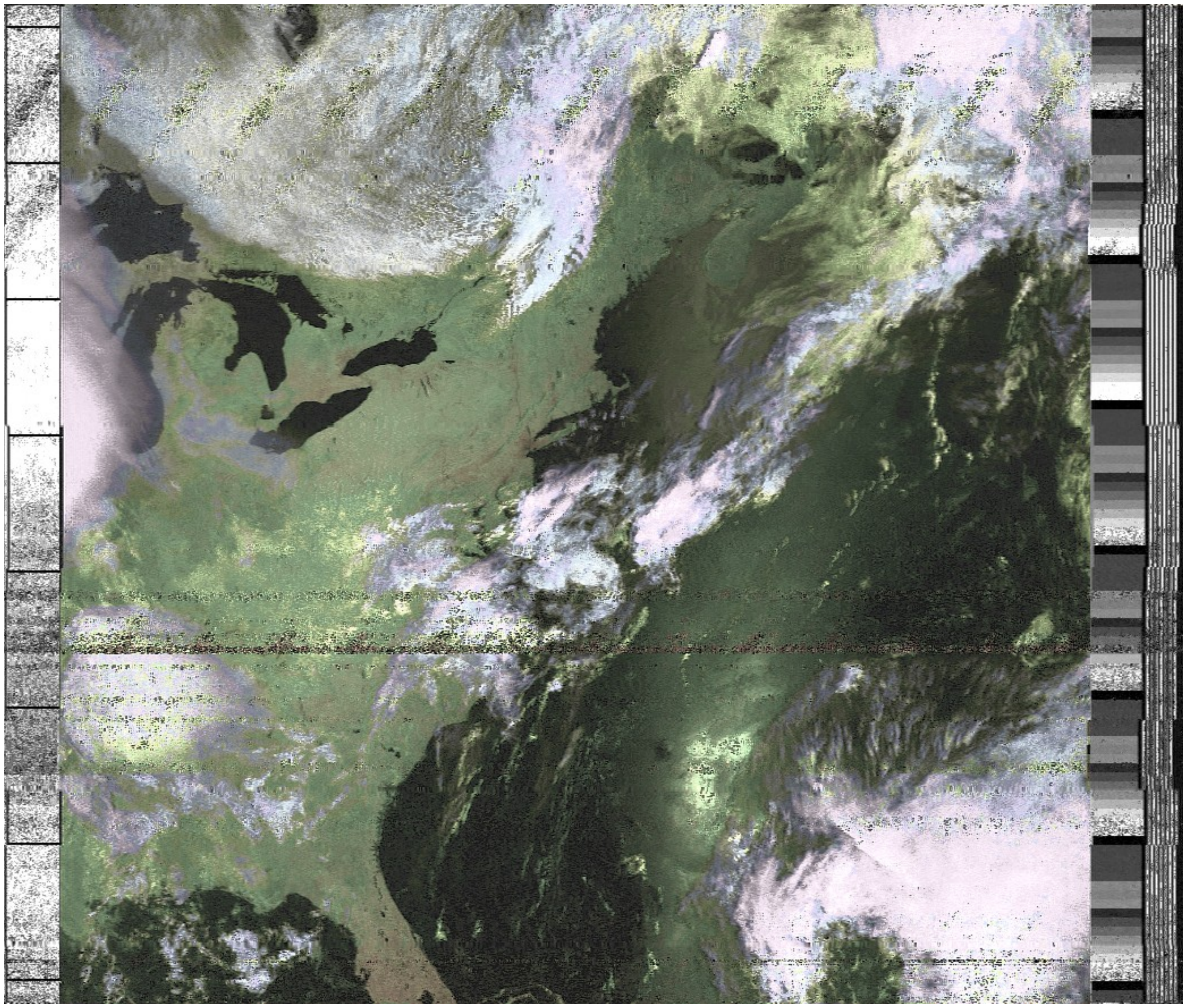


Fig. 33. NOAA 19 Decoded Image with colored overlay

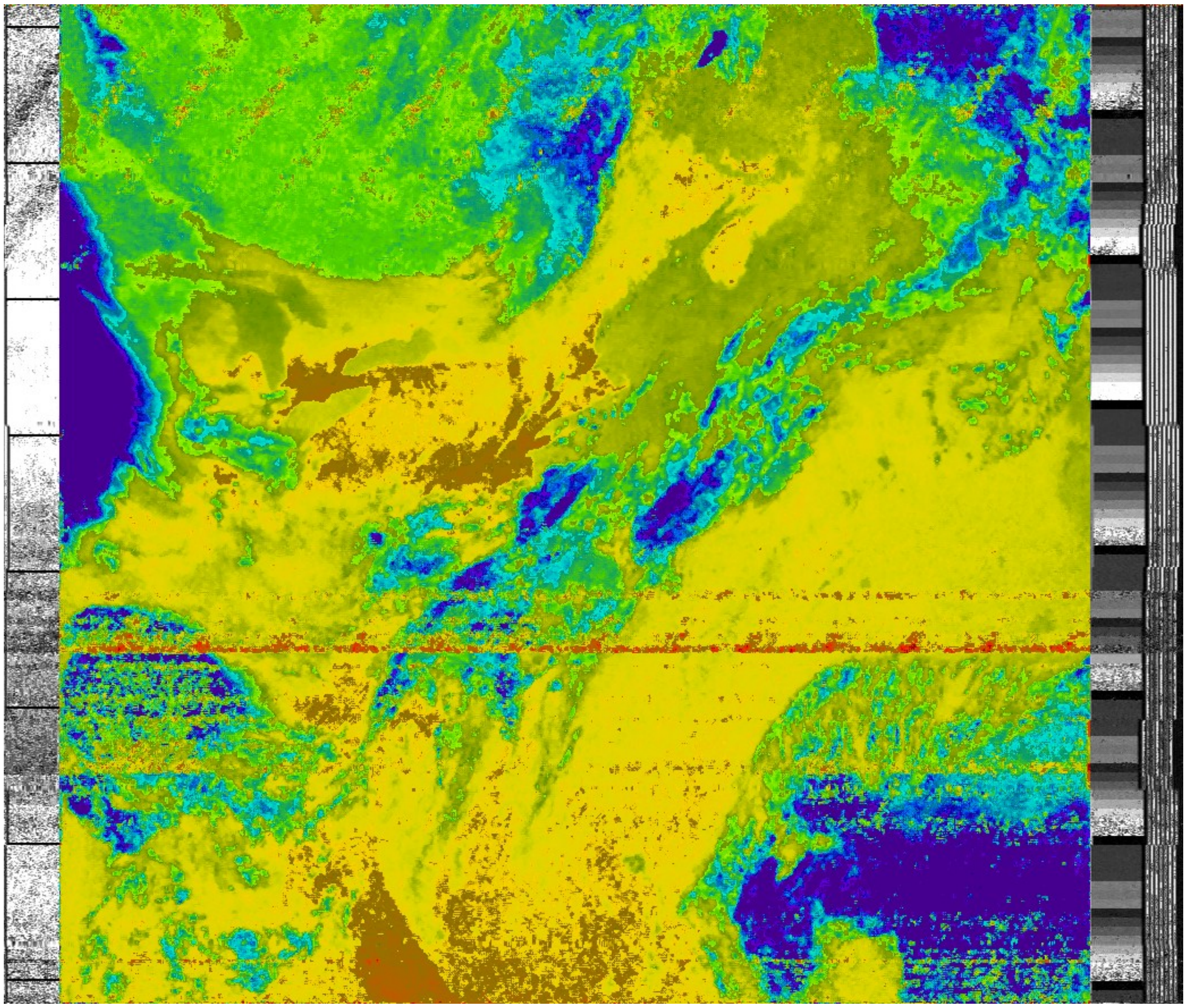
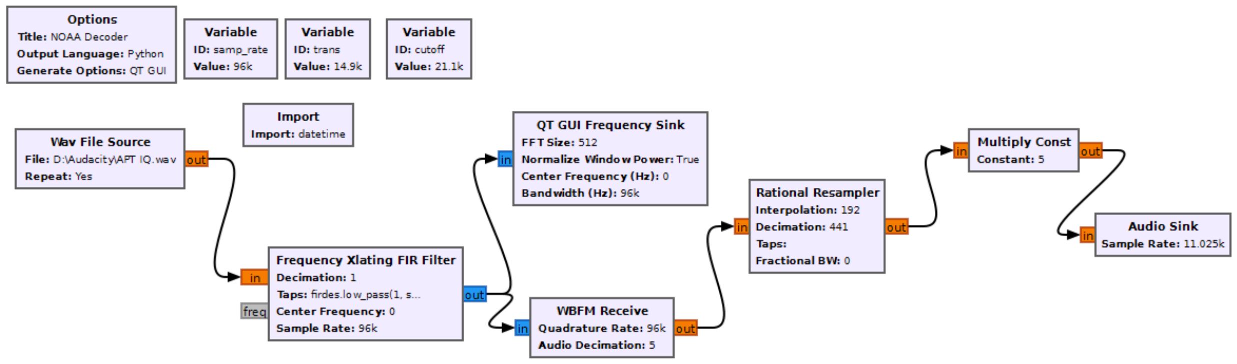


Fig. 34. NOAA 19 Decoded Image with thermal overlay

B. *NOAA GNU Radio Decoder*

This section explores the implementation of a GNU Radio-based NOAA satellite signal decoder using recorded IQ data from previous experiments. The implementation demonstrates the processing and decoding of satellite signals using an advanced software-defined radio (SDR) framework. The main objective of this subsection is to illustrate how the provided GNU Radio code can be utilized to decode NOAA satellite signals from previously recorded IQ data, showcasing the flexibility and power of GNU Radio in handling complex signal processing tasks.



Options
Title: NOAA Decoder
Output Language: Python
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 96k

Variable
ID: trans
Value: 14.9k

Variable
ID: cutoff
Value: 21.1k

Import
Import: datetime

Wav File Source
File: D:\Audacity\APT IQ.wav
Repeat: Yes

Frequency Xlating FIR Filter
Decimation: 1
Taps: firdes.low_pass(1, s...
Center Frequency: 0
Sample Rate: 96k

QT GUI Frequency Sink
FFT Size: 512
Normalize Window Power: True
Center Frequency (Hz): 0
Bandwidth (Hz): 96k

WBFM Receive
Quadrature Rate: 96k
Audio Decimation: 5

Rational Resampler
Interpolation: 192
Decimation: 441
Taps:
Fractional BW: 0

Multiply Const
Constant: 5

Audio Sink
Sample Rate: 11.025k

Fig. 35. NOAA Decoder GNU Radio flow-graph

This GNU Radio flow-graph is designed to process and decode audio signals from NOAA satellites using several key components. The flowgraph begins by reading IQ data from a WAV file that contains a recorded signal from one of the previously mentioned satellite passes. A frequency translating FIR filter is applied to the incoming signal, shifting the frequency to baseband and isolating the desired signal components through a low-pass filter. The wideband FM receiver block demodulates the frequency-modulated (FM) signal, converting it into an audio signal suitable for further processing. A rational re-sampler adjusts the sample rate of the audio signal to match the requirements of the audio sink, ensuring smooth playback and processing, while the final audio output is sent to the audio sink for real-time listening and analysis of the decoded signal. Additionally, a frequency sink is included to visualize the signal's spectrum, providing insights into the signal characteristics and aiding in debugging and analysis.

Recorded IQ data from the previous section was fed into this GNU Radio flow-graph to demonstrate the decoding process. The Wav File Source block was configured to read IQ data from the specified file path, allowing the flow-graph to access previously recorded satellite signals. The Analog WFM block effectively demodulated FM signals from NOAA satellites, extracting audio information embedded within the signals. The resampled audio was output through the audio sink, providing a real-time audio representation of the satellite transmission, while the QT GUI Frequency Sink provided a visual representation of the frequency spectrum, enabling the identification of key signal components and facilitating further analysis.

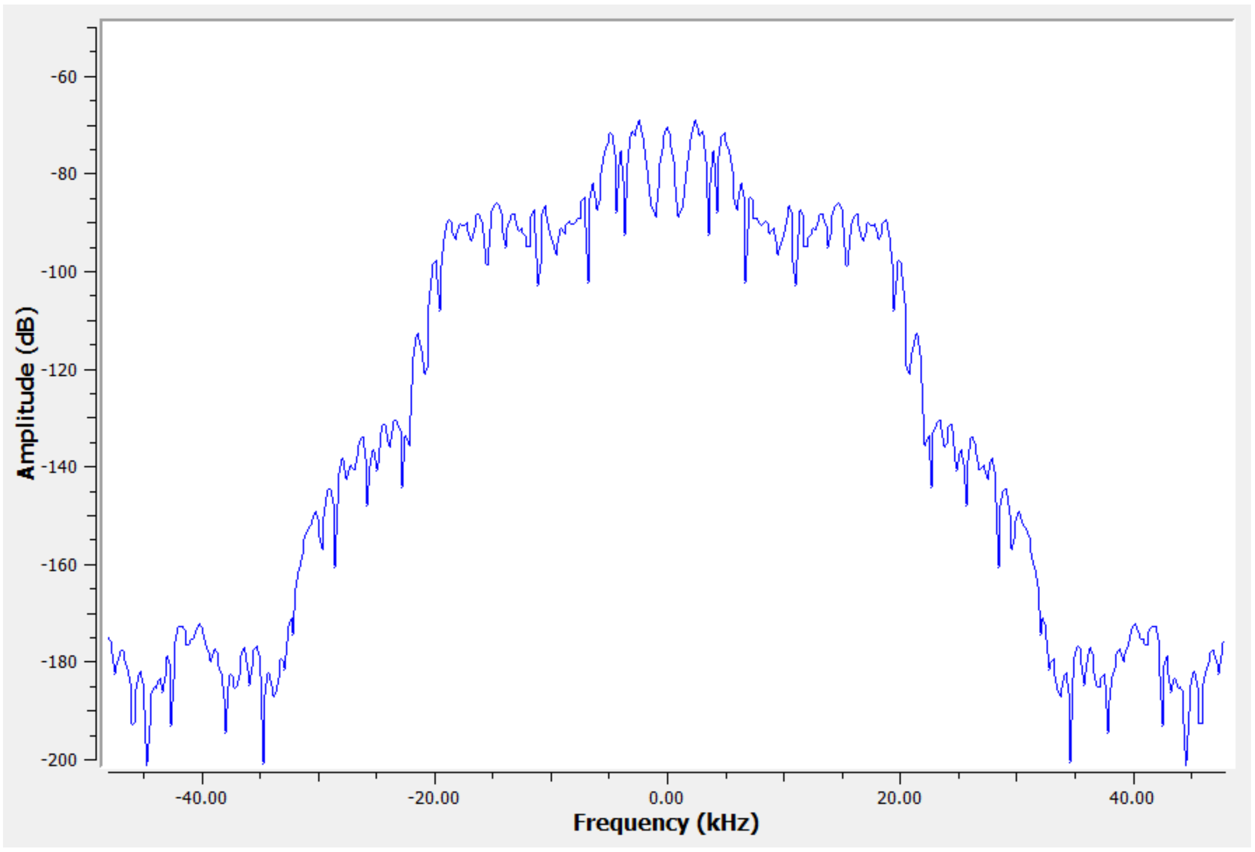


Fig. 36. NOAA Signal being displayed via QT GUI Frequency Sink

The use of GNU Radio for decoding satellite signals from recorded IQ data highlights the flexibility and robustness of SDR technology. By leveraging the capabilities of GNU Radio, the process of decoding and analyzing NOAA satellite transmissions was successfully demonstrated. This approach reinforces the utility of GNU Radio in satellite communication applications and emphasizes its potential for educational and research purposes.

C. BEESAT GNU Radio Investigation

The BEESAT satellite series, developed by TU Berlin, operates using various modulation techniques and protocols, most notably GMSK (Gaussian Minimum Shift Keying) at baud rates of 1200, 2400, 4800, and 9600 bps. GNU Radio, an open-source signal processing toolkit, provides the necessary tools for demodulating and processing signals from the BEESAT satellites. To facilitate this, TU Berlin hosts a repository containing out-of-tree (OOT) modules designed for GNU Radio. These modules are available for BEESAT-1 through BEESAT-9, along with other satellites such as TechnoSat and NanoFF, and can be used to demodulate raw data frames transmitted via GMSK.

The GNU Radio blocks provided by TU Berlin allow the demodulation of GMSK signals and use the Mobitex data-link layer protocol for further processing. The demodulation process is critical for decoding telemetry data transmitted by the BEESAT satellites. For instance, BEESAT-5 through BEESAT-8 transmit signals at baud rates of 1200, 2400, and 4800 bps, using GMSK modulation with a BT of 0.5, and their telemetry data can be decoded using the appropriate GNU Radio flowgraphs. These flowgraphs, such as tnc-b1.grc for BEESAT-1 and tnc-nx.grc for BEESAT-2 through BEESAT-9, can be found within the TU Berlin repository. [0]

In this investigation, several challenges arose when attempting to install the necessary GNU Radio modules for BEESAT signal demodulation. Initially, the most recent BEESAT GNU Radio repository was targeted, which is written for GNU Radio version 3.8. However, the current version of GNU Radio in use during the project was 3.10, creating compatibility issues between the two versions. The first attempt was to install GNU Radio 3.8 within a virtual machine (VM) using VirtualBox. A pre-configured Ubuntu VM with GNU Radio 3.8 was used to circumvent issues with compiling the necessary dependencies on modern systems. Despite this approach, the installation of the BEESAT OOT modules encountered failures during the execution of installation commands, which are responsible for compiling and installing the external modules required by GNU Radio.

The installation failures stemmed from missing dependencies and incompatibility between the modules designed for version 3.8 and the host system. As a result, the OOT modules, which are essential for running the BEESAT GNU Radio flowgraphs, could not be properly installed. This failure rendered the repository's .grc files, which are used for demodulating and decoding the satellite signals, unusable in this environment.

To work around these compatibility issues, an attempt was made to revert to an older version of GNU Radio (3.7), as a separate repository exists that supports BEESAT files for this version. A fresh installation of Ubuntu was performed within another VM, and the guide provided by TU Berlin was followed to downgrade and compile GNU Radio 3.7. Unfortunately, while attempting to compile the software, the process consistently failed around 83 percent completion. This repeated failure, despite multiple attempts, indicated a deeper issue with compatibility between the GNU Radio 3.7 configuration and modern Linux distributions.

It was ultimately determined that the BEESAT GNU Radio repository has not been updated since June 2022, and the modules are no longer compatible with the most recent GNU Radio versions. This limitation highlights the challenge of maintaining long-term compatibility for satellite communication projects, especially as new software versions are released and older ones are deprecated.

VIII. Conclusion

The research and experiments presented in this report underscore the growing relevance and utility of Software Defined Radios in modern satellite communication. SDR technology, with its ability to shift traditionally hardware-bound tasks into flexible, software-driven processes, opens up new avenues for both research and practical applications in satellite signal decoding. This project aimed to explore how SDRs could be used effectively to receive, process, and decode signals from various satellite platforms, including NOAA weather satellites and the BEESAT pico-satellites.

The experiments conducted with NOAA satellites demonstrated that SDRs, in combination with inexpensive and accessible hardware like the RTL-SDR dongle, can reliably capture and decode weather data transmitted from orbit. Tools such as GNU Radio and WXtoIMG were pivotal in transforming raw satellite signals into usable weather images, highlighting the importance of choosing the right software pipeline for specific satellite communication tasks. This capability empowers hobbyists, educators, and professionals alike to set up cost-effective ground stations capable of real-time data acquisition and analysis without the need for proprietary or expensive equipment.

In parallel, the investigation into BEESAT signals brought to light both the potential and challenges associated with the use of SDRs in space research. GNU Radio's flexibility allowed for experimentation with custom demodulation schemes, such as GMSK, to decode telemetry from the BEESAT satellite series. However, the compatibility issues encountered during the installation of GNU Radio modules for BEESAT on newer software versions underscored a broader challenge: the importance of maintaining up-to-date and backward-compatible software tools in fast-evolving fields like satellite communications. The inability to install certain out-of-tree (OOT) modules due to outdated dependencies reflects a common problem in long-term research projects, where maintaining software compatibility is as critical as advancing the technology itself.

Despite these challenges, the project successfully demonstrated that SDRs are not only capable of decoding satellite telemetry but can also be adapted for use with a variety of satellite systems with minimal hardware modifications. This adaptability makes SDRs particularly valuable for research and educational purposes, where users may need to handle multiple signal types, modulation schemes, and frequency bands across different satellite platforms. The integration of SDRs with open-source software also fosters innovation, allowing for continuous improvements in signal processing techniques and the development of new applications in satellite communications.

Looking ahead, the future of SDRs in satellite communication appears promising. As software-defined solutions become more sophisticated, SDRs will likely play an increasingly important role in space exploration, environmental monitoring, and telecommunications. For example, the use of cognitive radios, which can automatically adjust their parameters to optimize communication performance, represents the next frontier for SDR technology. In satellite systems, this could translate into more efficient and resilient communications, capable of adjusting to changing conditions in real-time without human intervention.

In conclusion, this research highlights the versatility, affordability, and effectiveness of SDRs in satellite signal decoding and telemetry. From the success of decoding NOAA weather data to the exploration of BEESAT telemetry, SDR technology proves to be a robust tool in both professional and amateur satellite communication environments. Despite some technical challenges, SDRs hold immense potential for advancing satellite communication technologies and making them more accessible to a broader audience. As the technology continues to evolve, SDRs will likely remain at the forefront of innovation in satellite communications, driving advancements in data reception, processing, and analysis for years to come.

References

[1] P. Kovář and F. Vejražka, "Software radio and its applications in GNSS," in *Proceedings. Elmar-2004. 46th International Symposium on Electronics in Marine*, 2004, pp. 16–21. [Online]. Available: https://www.researchgate.net/publication/4104243_Software_radio_and_its_applications_in_GNSS

[2] M. Sharawi and O. Korniyenko, "Software Defined Radios: A Software GPS Receiver Example," in *2007 IEEE/ACS International Conference on Computer Systems and Applications*, 2007, pp. 562–565. [Online]. Available: https://www.researchgate.net/publication/4252849_Software_Defined_Radios_A_Software_GPS_Receiver_Example

[3] E. G. Peters and C. Benson, "A Doppler Correcting Software Defined Radio Receiver Design for Satellite Communications," *IEEE Aerospace and Electronic Systems Magazine*, vol. 35, pp. 38–48, 2020. [Online]. Available: https://www.researchgate.net/publication/340572253_A_Doppler_Correcting_Software_Defined_Radio_Receiver_Design_for_Satellite_Communications

[4] D. J. M. Peralta, D. S. D. Santos, A. Tikami, W. A. D. Santos, and E. W. R. Pereira, "Satellite Telemetry and Image Reception with Software Defined Radio Applied to Space Outreach Projects in Brazil," *Anais da Academia Brasileira de Ciências*, vol. 90, no. 3, pp. 3175–3184, 2018. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/30304244/

[5] H. Hurskainen, J. Raasakka, T. Ahonen, and J. Nurmi, "Multicore Software-Defined Radio Architecture for GNSS Receiver Signal Processing," *EURASIP Journal on Embedded Systems*, vol. 2009, pp. 1–10, 2009. [Online]. Available: https://www.semanticscholar.org/paper/Multicore-Software-Defined-Radio-Architecture-for-Hurskainen-Raasakka/84a9ea560575bd91b37afe79b549c85426e48efd?utm_source=direct_link

[6] R. W. Stewart, K. W. Barlee, D. S. W. Atkinson, and L. H. Crockett, *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. Glasgow: Strathclyde Academic Media, 2015.

[7] T. Ulversoy, "Software Defined Radio: Challenges and Opportunities," *IEEE Communications Surveys & Tutorials*, vol. 12, pp. 531–550, 2010. [Online]. Available: https://www.researchgate.net/publication/220250216_Software_Defined_Radio_Challenges_and_Opportunities

[8] D. C. Popescu and R. Vida, "A Primer on Software Defined Radios," *Infocommunications Journal*, vol. 14, no. 3, pp. 16–27, 2022. [Online]. Available: https://doi.org/10.36244/ICJ.2022.3.3

[9] NASA Global Precipitation Measurement (GPM), “NOAA-19 Satellite Information,” [Online]. Available: https://gpm.nasa.gov/science-team/resources/noaa19-satellite. [Accessed: 02-Jul-2024]. [Online]. Available: https://gpm.nasa.gov/science-team/resources/noaa19-satellite

[10] S.-S. Lin and Y.-Z. Song, “A doppler effect correction method for apt weather satellite image reception,” in *2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, 2021, pp. 1–2.

[11] M. R. Islam and R. H. B. Exell, “Navigation of NOAA APT images,” *International Journal of Remote Sensing*, vol. 16, no. 13, p. 2311–2315, 1995. [Online]. Available: https://doi.org/10.1080/01431169508954560

[12] C. Bósquez, A. Ramos, and L. Noboa, “System for receiving NOAA meteorological satellite images using software defined radio,” in *2016 IEEE ANDESCON*, 2016, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/7836233

[13] J. Peña-Becerra, E. P. Estupiñán-Cuesta, and J. C. Martínez-Quintero, “NOAA Satellite Weather Stations: State of the Art, Perspective and Future Projection,” *Visión electrónica*, 2021. [Online]. Available: https://www.researchgate.net/publication/371095355_NOAA_Satellite_Weather_Stations_State_of_the_Art_Perspective_and_Future_Projection

[14] H. Kayal, F. Baumann, K. Brieß, and S. Montenegro, “BEESAT: A Pico Satellite for the On Orbit Verification of Micro Wheels,” in *2007 3rd International Conference on Recent Advances in Space Technologies*, 2007, pp. 497–502. [Online]. Available: https://www.researchgate.net/publication/4266386_BEESAT_A_pico_satellite_for_the_on_orbit_verification_of_micro_wheels

[15] M. R. Maheshwarappa, M. D. J. Bowyer, and C. Bridges, “Software Defined Radio (SDR) architecture to support multi-satellite communications,” in *2015 IEEE Aerospace Conference*, 2015, pp. 1–10. [Online]. Available: https://ieeexplore.ieee.org/document/7119186

[16] EO Portal, “BEESAT-1: Berlin Experimental Educational Satellite-1,” [Online]. Available: https://www.eoportal.org/satellite-missions/beesat-1beesat-1-berlin-experimental-educational-satellite-1. [Accessed: 02-Jul-2024]. [Online]. Available: https://www.eoportal.org/satellite-missions/beesat-1#beesat-1-berlin-experimental-educational-satellite-1

[17] J. Heller and I. Jacobs, “Viterbi decoding for satellite and space communication,” *IEEE Transactions on Communication Technology*, vol. 19, no. 5, pp. 835–848, 1971.

[18] E. Biglieri, “High-level modulation and coding for nonlinear satellite channels,” *IEEE Transactions on Communications*, vol. 32, no. 5, pp. 616–626, 1984.

[19] R. Morelos-Zaragoza, O. Takeshita, H. Imai, M. Fossorier, and S. Lin, “Coded modulation for satellite broadcasting,” in *Proceedings of GLOBECOM’96. 1996 IEEE Global Telecommunications Conference*, 1996, pp. 31–35.

[20] C. Fragouli and A. Polydoros, “Serially concatenated coding for broadcasting s-umts applications,” in *IEEE Seventh International Symposium on Spread Spectrum Techniques and Applications,*, vol. 3, 2002, pp. 697–701 vol.3.

[21] S. Morosi, R. Fantacci, E. Del Re, and R. Suffritti, “Soft demapping and iterative decoding for satellite communications,” in *2007 IEEE International Conference on Communications*, 2007, pp. 4432–4437.

[22] H. El Gamal, B. Beidas, and S. Kay, “Turbo decoding for high spectral efficiency satellite communications,” in *2000 IEEE International Conference on Communications. ICC 2000. Global Convergence Through Communications. Conference Record*, vol. 1, 2000, pp. 440–445 vol.1.

[23] “RTL-SDR BLOG V3 USB Dongle with Dipole Antenna Kit,” [Online]. Available: https://www.sparkfun.com/products/22957. [Accessed: 02-Jul-2024], 2024, SparkFun Electronics. [Online]. Available: https://www.sparkfun.com/products/22957

[24] RTL-SDR Blog, “About RTL-SDR.” [Online]. Available: https://www.rtl-sdr.com/about-rtl-sdr/

[25] E. Surducan, V. Surducan, D. Iancu, and C. Glossner, “Multiband Antennas for SDR Applications,” *Int. J. Digit. Multim.*

Broadcast., vol. 2009, pp. 460 143:1–460 143:9, 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1155/2009/460143

[26] "The Yagi-Uda Antenna," [Online]. Available: https://www.digikey.com/en/blog/the-yagi-antenna. [Accessed: 02-Jul-2024], 2024, DigiKey. [Online]. Available: https://www.digikey.com/en/blog/the-yagi-antenna

[27] K. Boyle, P. Steeneken, Z. Liu, Y. Sun, A. Simin, T. Huang, E. Spits, O. Kuijken, T. Roedle, and F. V. Straten, "Reconfigurable Antennas for SDR and Cognitive Radio," in *IC.2007.1549*, 2007, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/4458674

[28] Z. Wang, L. Xiao, X. Su, Q. Xin, and X. Xu, "On the Optimization of Real Time Performance of Software Defined Radio on Linux OS," *Communications and Network*, vol. 5, pp. 292–297, 2013. [Online]. Available: https://www.scirp.org/journal/paperinformation?paperid=39380

[29] J. Tosch, A. Aiken, and S. Dasson, "Software Defined Radio," 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/9780470742020.ch2

[30] M. Sruthi, M. Abirami, A. Manikkoth, R. Gandhiraj, and K. P. Soman, "Low cost digital transceiver design for Software Defined Radio using RTL-SDR," in *2013 International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s)*, 2013, pp. 852–855. [Online]. Available: https://ieeexplore.ieee.org/document/6526525

[31] Y. J. Wang, "Operating Linux Softwares on Windows Platform," *Applied Mechanics and Materials*, vol. 394, pp. 495–498, 2013. [Online]. Available: https://www.scientific.net/AMM.394.495

[32] "Gpredict: Free, Real-Time Satellite Tracking and Orbit Prediction Software." [Online]. Available: https://oz9aec.dk/gpredict/

[33] Blaine, "Gqrx SDR," [Online]. Available: https://www.gqrx.dk/. [Accessed: 02-Jul-2024], 2023, Gqrx Software Defined Radio. [Online]. Available: https://www.gqrx.dk/

[34] P. Romani, "SDRsharp Big Book 2024," [Online]. Available: https://airspy.com/?ddownload=5928, 2024. [Online]. Available: https://airspy.com/?ddownload=5928

[35] A. D. Kulkarni, R. Bhadade, A. A. Naik, and V. Gohokar, "Real-time Decoding of Satellite Signals," in *2022 IEEE North Karnataka Subsection Flagship International Conference (NKCon)*, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/10126774

[36] L. Presti, E. Falletti, M. Nicola, and M. Gamba, "Software Defined Radio technology for GNSS receivers," in *2014 IEEE Metrology for Aerospace (MetroAeroSpace)*, 2014, pp. 314–319. [Online]. Available: https://ieeexplore.ieee.org/document/6865941

[37] S. Mahmood, M. T. Mushtaq, and G. Jaffer, "Cost efficient design approach for receiving the NOAA weather satellites data," in *2016 IEEE Aerospace Conference*, 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7500854

[38] S. S, K. M, K. M, and G. P, "Establishment of pre-processing station for denoising noaa satellite images using legendre fenchel transformation method," in *2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, 2015, pp. 1–6.

[39] R. Wiryadinata, M. Khoirussolih, N. Rohanah, I. Muttakin, and T. Firmansyah, "Image Data Acquisition for NOAA 18 and NOAA 19 Weather Satellites Using QFH Antenna and RTL-SDR," *MATEC Web of Conferences*, vol. 218, p. 02002, 2018. [Online]. Available: https://www.matec-conferences.org/articles/matecconf/abs/2018/77/matecconf_iciee2018_02002/matecconf_iciee2018_02002.html

[40] GNU Radio, "GNU Radio," [Online]. Available: https://www.gnuradio.org. [Accessed: 24-Jul-2024], 2024. [Online]. Available: https://www.gnuradio.org

[41] ——, “UbuntuVM Installing the VirtualBox Extension Pack,” [Online]. Available: https://wiki.gnuradio.org/index.php?title=UbuntuVMInstalling$_the_V$
$02 - Jul - 2024], 2024.[Online].Available$:

T. U. Berlin, “Amateur Radio - Technische Universität Berlin,” [Online]. Available: https://www.tu.berlin/en/raumfahrttechnik
radio. [Accessed: 02-Jul-2024], 2024. [Online]. Available: https://www.tu.berlin/en/raumfahrttechnik/institute/
amateur-radio