

Developing the Test Bench for a Next-Generation Bistatic
GNSS Reflectometry Receiver Instrument

Thesis

Presented in Partial Fulfillment of the Requirements for
Graduation with Honors Research Distinction in the
College of Engineering at The Ohio State University

By

Ryan Linnabary

Department of Electrical and Computer Engineering
The Ohio State University

2018

Thesis Committee:

Prof. Joel T. Johnson, Adviser

Dr. Andrew O'Brien

Prof. Inder J. Gupta

Copyright by Ryan Linnabary
2018

Abstract

Recently, scientists have started to exploit global navigational satellite system (GNSS) signals for geophysical remote sensing using a technique called “reflectometry” (GNSS-R). This involves an airborne or spaceborne receiver measuring direct GNSS satellite signals as well as the reflected signals off of the surface of the Earth. This enables scientists to measure properties of Earth’s surface at the reflection point. NASA launched a constellation of eight micro-satellites, called CYGNSS, in December of 2016 that carried GNSS-R receivers. CYGNSS tracks reflected GPS signals to measure ocean wind speed for hurricane forecasting. Use of GPS signals has several advantages in terms of cost, temporal coverage, and spatial coverage.

Researchers at The Ohio State University are involved in the CYGNSS Science Team and testing of first generation GNSS-R receivers. The success of the project so far has motivated a potential follow-on mission as researchers are identifying opportunities for improvement and expansion of reflectometry capabilities. A next-generation reflectometry receiver is being developed which should be capable of tracking more than the GPS L1 C/A-coded signal to increase the number of visible reflections. To evaluate the behavior of this new instrument, a test bench is implemented based on a test-system created for CYGNSS. The design and validation of the test bench is documented in this thesis.

The test bench developed for CYGNSS is not capable of simulating other GNSS bands, cannot generate long-duration signals, and does not incorporate GNSS meta-

data or navigational messages in its simulations. The purpose of this project is to improve the quality of the test bench hardware and software used to test GNSS-R instruments and add support for the new instrument's enhanced features. In this work we accomplish the following tasks: we assemble a transceiver from software defined radios; we modify the signal generator for parallel processing; we provide an efficient and simple user interface; we add support for the L1, L5, E1bc, and E5a bands; and we include actual satellite meta-data, including navigational messages and realistic timing information, in our simulations.

Deliverable test signals, which are generated and transmitted by the completed system, are validated to confirm an accurate simulation of the receiver's space environment. Analysis of time-domain samples verifies modulation of spread-spectrum codes and simulated doppler shift. An FFT of these time-domain samples visually confirms the frequency contents of these signals in spectrum plots. Signal acquisition is performed on the files to identify PRN codes and verify the presence of multiple satellites. RF lab equipment verifies the frequency content of the test-system output.

In the Summer of 2018, testing of the completed receiver will commence. We plan to subject the receiver to a 24-hour broadcast of a dual-band signal from the test bench according to the procedure outlined in a test plan developed by May 15th. We hope that this effort discloses any problems with the receiver prior to space deployment, at which time it will be impossible to make changes.

This research provides a valuable tool for testing reflectometry receivers in the future. It is easily adapted for expansion of features and to add support for alternative applications. The test bench supports development of a valuable reflectometry instrument, which may facilitate the University's involvement in a follow-on mission to CYGNSS.

Dedicated to the Free Software Foundation

Acknowledgements

I sincerely thank Dr. Andrew O'Brien, my mentor, for moral and technical support during my research. He motivated me to investigate the unknown and was available for guidance in challenging circumstances.

I also express gratitude to my faculty research advisor, Prof. Joel Johnson. Prof. Johnson helped me in coordinating this research project and guide my progress. He is an inspiring person and someone I seek to emulate in my own career.

I'd like to recognize Prof. Inder Gupta for supporting me in this effort as well. Prof. Gupta offered guidance and advice to me as I became involved in undergraduate research, motivating me in pursuit of my goals.

I also express thanks to my colleagues who observed my progress and were always available to help. This includes Eric Loria, Mohammad Al-Khaldi, Matthew Buchanan, Leo Tchorowski, and Justin Kuric.

Thanks to Dr. Chris Ball and Dr. Nima Ghalichechian, who facilitated my initial involvement at ElectroScience Lab and introduced me to engineering research.

This research was performed in the Navigation Research Lab at ElectroScience Lab and was made possible by my mentors, my adviser, Interim Director Robert Burkholder, and all faculty at this wonderful facility.

Thanks to everyone involved for the work that you do. It has enabled and contributed to my success in research as an undergraduate at the Ohio State University.

Vita

1989Born - Columbus, Ohio, USA
2008Diploma - Big Walnut High School - Sunbury, Ohio, USA
2013A.S. - Columbus State Community College - Columbus, OH, USA
2014-2018B.S.E.C.E. - The Ohio State University - Columbus, OH, USA

Fields of Study

Major Field: Department of Electrical and Computer Engineering

Field of Study: Computer Engineering

Research: Electromagnetics, Remote Sensing, and Digital Signal Processing

Contents

Abstract	ii
Acknowledgements	v
Vita	vi
List of Figures	ix
List of Tables	xi
List of Algorithms	xii
1 Introduction	1
1.1 Background	1
1.1.1 Global Navigational Satellite Systems (GNSS)	1
1.1.2 GNSS Reflectometry (GNSS-R)	3
1.1.3 Next Generation GNSS-R Receiver	6
1.2 Problem Statement	6
1.3 Research Goals and Significance	7
1.4 Thesis Overview	7
2 Methodology	9
2.1 Hardware Design and Configuration	9
2.1.1 Test Bench Setup	10
2.1.2 Software-Defined Radios	12
2.1.3 Signal Storage	14
2.1.4 Host PC	16
2.2 Software	18
2.2.1 User Interface	19
2.2.2 Signal Generator	21
2.3 Signal Validation	26
3 Results	28
3.1 Waveform Analysis	28
3.1.1 Band 1: L1 and E1bc	28

3.1.2	Band 2: L5 and E5a	32
3.2	Signal Acquisition	36
3.2.1	GPS L1	37
3.2.2	Galileo E1bc	39
3.2.3	GPS L5	41
3.2.4	Galileo E5a	43
3.3	Signal Tracking	45
3.4	RF Validation	46
3.5	Data Acquisition	48
4	Conclusion	49
4.1	Contributions	49
4.2	Additional Applications	49
4.3	Future Work	50
4.4	Summary	50
	Appendices	52
A	Significant GNSS Parameters	53
	References	55

List of Figures

1.1	GNSS Signal Components	3
1.2	GNSS Reflectometry Illustration	4
1.3	Map of CYGNSS Global Coverage	5
1.4	Simulated Typical Delay-Doppler Map	5
2.1	Diagram of Benchtop Test Setup	10
2.2	Photograph of Benchtop Hardware Test Setup	11
2.3	Photograph of Antenna Setup for Data Acquisition	12
2.4	USRP N210 Software Defined Radio	13
2.5	Octoclock Multi-Channel Clock Synchronization Module	13
2.6	Sequence of Sample Files	14
2.7	Waveform Generator Command Prompt Interface	16
2.8	File Writing Output for Debugging	17
2.9	Threaded File Playback Using Custom UHD Interface Program	18
2.10	User Interface and Working Software Development Directory Contents	19
2.11	Python Program	20
2.12	Primary C Program Functions	21
3.1	GSP L1 C/A-Code Samples	29
3.2	GSP L1 C/A-Code Spectrum	30
3.3	Galileo E1bc Samples	30
3.4	Galileo E1bc Spectrum	31
3.5	Band 1 Samples and Spectrum	31
3.6	Band 1 Spectrum	32
3.7	GPS L5 Samples	33
3.8	GPS L5 Spectrum	33
3.9	Galileo E5a Samples	34
3.10	Galileo E5a Spectrum	34
3.11	Band 2 Samples	35
3.12	Band 2 Spectrum	36
3.13	Visually Confirmed Initial L1 PRN Acquisitions	38
3.14	Visually Confirmed Initial E1bc PRN Acquisitions	40
3.15	Visually Confirmed Initial L5 PRN Acquisitions	42
3.16	Visually Confirmed Initial E5a PRN Acquisitions	44
3.17	L1 PRN 4 Acquisitions for First 15 Seconds	45

3.18 Band 1 Output at 1.57542 <i>GHz</i> with 25MHz Span	46
3.19 Band 2 Output at 1.17645 <i>GHz</i> with 25MHz Span	46
3.20 Dual-Band Output From Test Bench Transciever	47
3.21 Signals Acquired from Live-Sky SDR GNSS Transciever Recordings .	48

List of Tables

2.1	Waveform File Format	15
3.1	Initial DDM Parameters for Band 1 L1 Acquisitions	37
3.2	Initial DDM Parameters for Band 1 E1bc Acquisitions	39
3.3	Initial DDM Parameters for Band 2 L5 Acquisitions	41
3.4	Initial DDM Parameters for Band 2 E5a Acquisitions	43
A1	GNSS Signal Parameters	53

List of Algorithms

1	Signal Generator <code>main()</code> Function	23
2	Signal Generator <code>generate()</code> Function	24
3	Signal Generator <code>getSignal()</code> Function	25
4	Signal Generator <code>getSample()</code> Function	26

Chapter 1

Introduction

This chapter introduces Global Navigational Satellite System (GNSS) signals and their utility in remote sensing research. A general overview of GNSS signal parameters supports a discussion of the problem at hand and the goals of this project.

1.1 Background

1.1.1 Global Navigational Satellite Systems (GNSS)

GNSS receivers enable autonomous timing and positioning using radio-frequency (RF) signals from constellations of orbiting satellites (space vehicles). These constellations, some of which are still under development, include the United States GPS, Russian GLONASS, Chinese BeiDou Navigation Satellite System, Japan’s regional Quazi-Zenith Satellite System (QZSS), and the European Union’s Galileo system.

The work discussed here is focused on GPS and Galileo signals (specifically GPS L1, GPS L5, Galileo E1bc, and Galileo E5a). The first GPS satellite was launched in 1978 and the constellation currently contains 31 operating space vehicles [1]. All GPS space vehicles broadcast the L1 C/A-coded civilian “legacy” signal. The newest blocks of satellites (Blocks IIF and III, currently 12 space vehicles) broadcasts the third civilian “modernized” L5 signal [1]. Although the Galileo constellation is not

yet complete, 22 space vehicles are available as of February 2018 [2]. The Galileo program is scheduled to be fully operational by 2020 [2]. Table A1 in Appendix A summarizes other significant GPS and Galileo constellation parameters.

GPS and Galileo signals use code division multiple access (CDMA) for unique identification of signals from individual satellites. This is achieved by modulating the data bits with a spread-spectrum code. This code is called a pseudo-random noise (PRN) sequence for our purposes. Such a sequence is chosen for its low cross-correlation properties with other PRN sequences [3]. Spread-spectrum technology has additional advantages beyond signal identification; it allows a receiver to acquire and track low-power signals under the noise floor, and it enables derivation of signal delay which is proportional to signal path distance (travel time) [4]. The sequence is named pseudo-random noise in reference to the fact that it is predictable but statistically resembles noise. A GNSS receiver generates a replica of the code and acquires a signal by cross-correlation, where the replica is adjusted in delay and Doppler until the sequences align, resulting in a strong cross-correlation value.

There are several different signals broadcast by each GNSS satellite, and each signal typically implements a different method of code generation. Although each code is different, they all seek to approximate the characteristics of an infinite sequence of independent random binary variables [4]. Some finite-length sequences have similar cross-correlation characteristics to the infinite sequence and may be generated using a shift register. A separate low rate navigational message is modulated on top of the PRN sequence to form the complete GNSS signal. The navigational message allows a satellite to broadcast orbit parameters required by the user for positioning, as well as other auxiliary information. It includes parameters for satellite ephemeris, time, clock corrections, service, ionosphere, and almanacs [3]. The GPS L1 C/A and L5 codes are modulated with the PRN and navigational data through binary phase-shift keying (BPSK). This method of modulation is illustrated in Figure 1.1. Note that

the period of the codes are not displayed to scale.

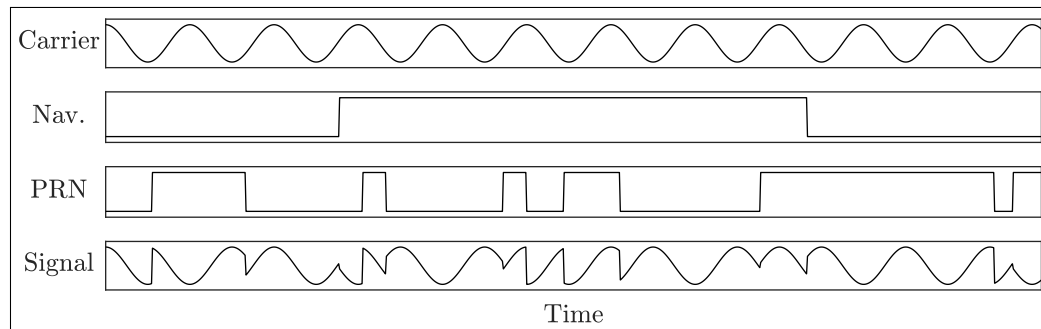


Figure 1.1: GNSS Signal Components

Figure 1.1 shows the result of modulating the bits of both the navigation message and the PRN code on a sinusoidal carrier signal. The phase of the carrier is shifted by 180° for each transition in the binary sequences. A receiver compares the modulated signal to a reference carrier to recover the binary information.

1.1.2 GNSS Reflectometry (GNSS-R)

Even though GNSS was originally designed for navigation and timing, it has been recently recognized that it provides a set of unique signals-of-opportunity for geophysical remote sensing. Remote sensing techniques take advantage of the precision timing of GNSS signals to make inferences about the properties of the Earth's surface by observing characteristics of the reflected signal. It has been demonstrated that these signals may be used to “passively” sense features of the Earth because satellites persistently broadcast for navigational purposes. Signal reflections can be accurately sensed by airborne and spaceborne GNSS receivers, and they primarily occur at a single point on the surface called the specular point [5]. Figure 1.2 depicts reflectometry as a bistatic radar configuration.

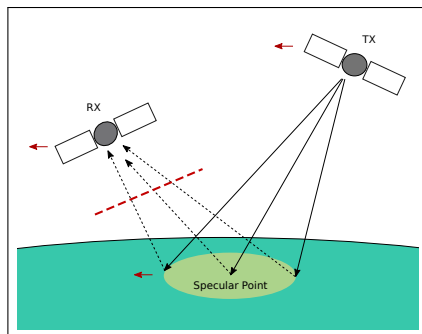


Figure 1.2: GNSS Reflectometry Illustration

Figure 1.2 shows a GNSS transmitter broadcasting a signal toward the Earth’s surface. A reflectometry receiver observes the reflection of this signal with a downward-pointing antenna. The reflection appears to the receiver as a spatial distribution of scattered signal power within what is called the “glistening zone”, centered at the “specular point”. Each element in this configuration exhibits a velocity relative to the Earth’s surface indicated by red arrows in the diagram. Considering the curvature of the earth and the relative positions and speeds of the satellites, interesting scattering geometries arise in the course of data acquisition.

The geometry gives rise to Doppler shifts and delays for the reflections from different satellites. Thus, the geometry determines where in delay and Doppler space the receiver will find the reflection information. The scattered power depends on the surface’s roughness. As first verified by the UK-DMC receiver in 2006 [6], the surface area of the zone around the specular point (glistening zone) is proportional to ocean surface roughness and therefore also to wind-speed. These observations motivated NASA’s Cyclone Global Navigational Satellite System (CYGNSS).

CYGNSS is a constellation of eight small satellites launched in December of 2016 to measure wind speeds over Earth’s oceans for hurricane monitoring [7]. The system observes latitudes from $\pm 38^\circ$, revisiting every 5-6 hours. These eight satellites each track four GPS L1 C/A-coded signals for a total of 32 simultaneous measurements. Figure 1.3 displays the spatial coverage of CYGNSS satellites.

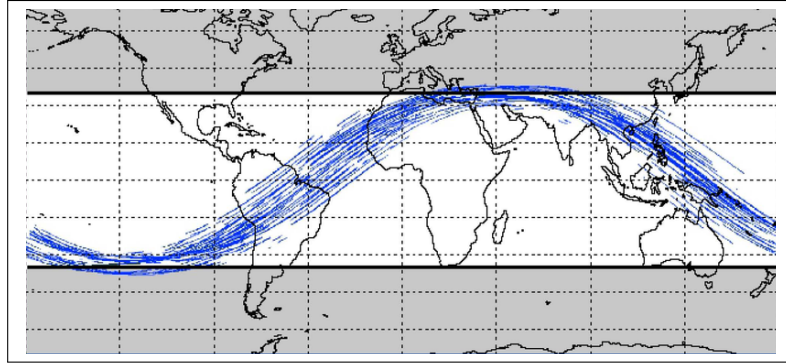


Figure 1.3: Map of CYGNSS Global Coverage [8]

GNSS-R receivers produce a delay-doppler map (DDM) from the acquired signal reflections. A typical, simulated ocean DDM is illustrated in Figure 1.4.

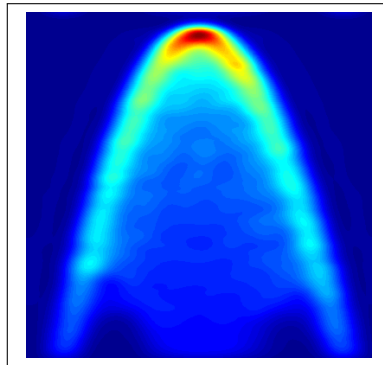


Figure 1.4: Simulated Typical Delay-Doppler Map

This image illustrates a two dimensional function of delay and Doppler frequency. Generally the delay is plotted increasing from top to bottom, in units *chips* (number of PRN code samples) and is proportional to signal path distance. Doppler frequency is expressed in kHz and is caused by the changing relative velocities of the satellites. The DDM illustrates power distribution, so the peak power corresponds to the red region near the top of Figure 1.4. Peak power is received from the center of a specular point. At this location the signal path is shortest and signal scattering is maximum. The tails of the DDM display reflected power from points further from the center of the glistening zone. Surface roughness is responsible for long tails in a DDM; a smooth surface would concentrate the reflections at the center of the specular point

and produce a narrower DDM.

1.1.3 Next Generation GNSS-R Receiver

Researchers at The Ohio State University and colleagues at University of Michigan, are motivated by the success of NASA's first CYGNSS mission and seek to improve the capabilities of the next generation of instruments. A new GNSS-R receiver is currently being developed which will improve reflectometry techniques by tracking additional GNSS constellations. The team believes that a receiver capable of tracking multiple types of GNSS signals will greatly increase spatial coverage for remote sensing. This effort improves temporal coverage as well, an important implication for cyclone wind speed monitoring. Modernized wide-band signals, including the GPS L5 and Galileo E5a, provide higher resolution measurements than the L1 C/A code received by CYGNSS. Higher resolution measurements enable further research in altimetry, soil-moisture content, and vegetation monitoring.

1.2 Problem Statement

The new instrument that is currently being developed will need to undergo benchtop testing. To support this, bench top hardware and software need to be assembled for injecting simulated Earth reflected signals into the flight hardware. Since OSU was involved in development of the first NASA CYGNSS receiver, that system's test software is available for re-purposing. Though the existing test platform is not capable of comprehensively testing of the next-generation receiver, it provides a foundation of software to modify for our current demands.

The benchtop setup will need to simulate the signals as they would appear to the instrument on orbit. Figure 1.2 shows a red dashed line that defines a boundary between the receiver and the reflected and direct path signals. This line effectively defines the interface connecting this test bench system to the receiver; the test bench

mimics the receiver's spaceborne environment by simulating a field of reflections. In practice, many transmitters and specular points are visible at a time.

1.3 Research Goals and Significance

This research effort seeks to accomplish four main goals. First, we will assemble a software-defined radio (SDR) based hardware system in the lab. Second, existing test software will be modified to handle additional signals (GPS L5, GAL E1bc, and GAL E5a) and include satellite metadata. Third, an efficient and user-friendly user interface will be written for access to parallel signal generation and operation of the transceiver. Finally, we verify the quality of signals generated by performing signal acquisition and spectral analysis.

The test bench to be created is an important component in the development of future and more capable receivers for GNSS reflectometry. This research field is expanding as a cost-effective means of efficient and accurate geophysical remote sensing; therefore, the demand for GNSS-R development efforts is on the rise. Our research will produce a valuable tool for the test process. Such testing must be comprehensive and prove correct operation of the instrument, including the receiver's ability to measure multiple GNSS bands through a 24 hour period. The test system must also simulate GNSS signals as accurately as possible. A successful test system will enable the development team to characterize through tests how well the improved instrument will process such signals in space, and may allow developers to address any problems with the new receiver instrument before its future space deployment.

1.4 Thesis Overview

The remaining chapters of this thesis document our design methodology, results of our work, and a conclusion of this research and its impact in the field of remote sensing.

Chapter 2 describes the design and implementation of a multi-band GNSS simulator and transceiver (the test bench). A discussion of hardware details follows to highlight the system's capabilities and then transitions to a discussion of the software implementation, including the user interface and how the system manages waveform generation, storage, and playback. It also discusses the validation of simulated GNSS signals by reading the waveforms in software to acquire and identify individual satellite signals.

Chapter 3 presents example output from the completed benchtop testing. First the time-domain and frequency-domain representations of each signal are analyzed and compared to expected results. Then the process of signal combination is described. The results of signal validation are presented to illustrate the presence of many satellite signals among the generated waveform files. A basic tracking system is described which is written to simulate and visualize the actions of the receiver.

Chapter 4 concludes with a brief summary of the design and implementation, as well as a discussion of additional applications and future work.

Chapter 2

Methodology

Development of this test bench involves engineering research, design, implementation, and verification of results in the lab. We began by establishing an understanding of GNSS systems and the utility of GNSS signals in remote sensing research. In Chapter 1 we specified requirements for the test system related to its evaluation of the device under test, including which signals to simulate and to what extent to test the receiver. Possible implementations of the test bench were considered based on equipment at our disposal and available funding. This chapter discusses the selection of hardware components and software design decisions made to complete the remaining engineering design tasks, including prototype development and testing to verify fulfillment of the requirements specification.

2.1 Hardware Design and Configuration

Physical components of the benchtop system are selected primarily from equipment at hand, with the exception of the VeraPhase 6000 antenna which was purchased mainly to support development of the receiver. A hardware solution was established and documented before assembling the components.

2.1.1 Test Bench Setup

A diagram of the proposed test configuration is shown in Figure 2.1. The primary hardware components include the host computer and an array of 6 software-defined radios (SDRs), labeled “USRP-X”, which form a custom GNSS transceiver system. An Ethernet connection establishes communication between the host and SDRs. The USRPs are synchronized by a clock distribution module called an “Octoclock” [9].

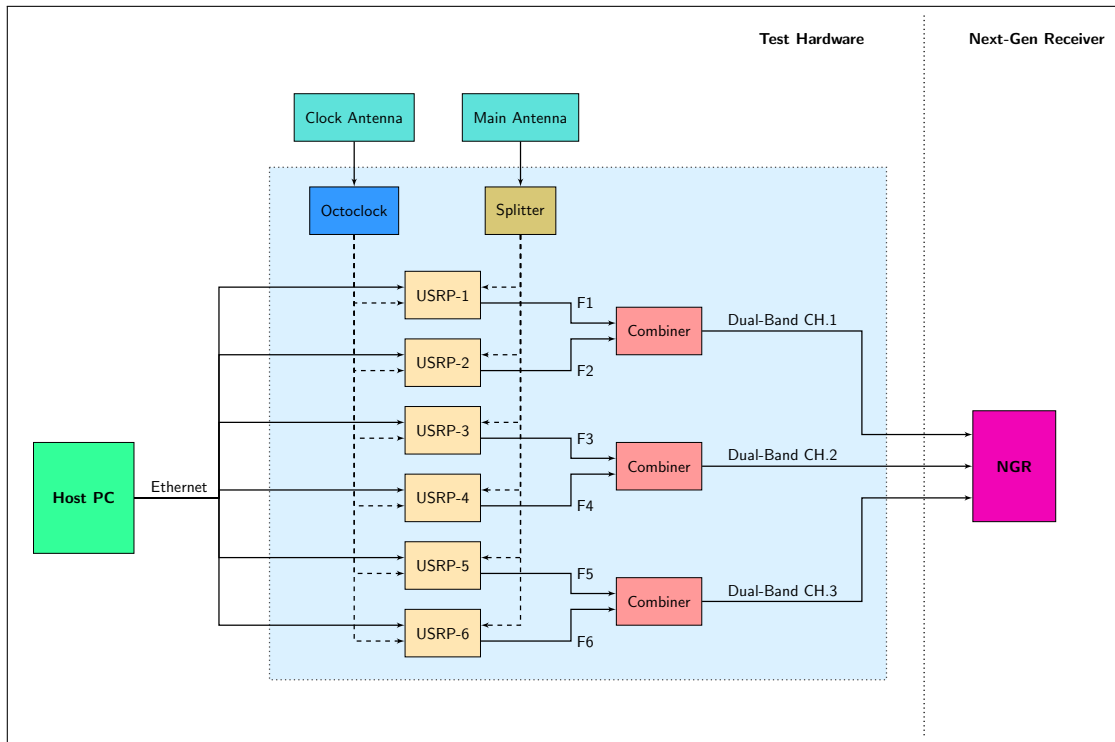


Figure 2.1: Diagram of Benchtop Test Setup

For preliminary data collection, the inputs of the transceiver’s USRPs are connected to a splitter which distributes the signal collected by the VeraPhase antenna (labeled “Main Antenna”). The Octoclock synchronizes its internal clock with orbiting GNSS clocks using the received signals from a small GNSS antenna labeled “clock antenna”.

For testing purposes, the transceiver’s 6 output channels are paired by RF combiners to produce 3 dual-band RF channels. These channels are connected to antenna ports on the receiver. Software on the Host PC configures the USRPs and coordinates

the transfer and modulation of waveform samples at appropriate carrier frequencies.

A photograph in Figure 2.2 shows the completed test setup. Components are oriented in this image as in the diagram of Figure 2.1. Here, a spectrum analyzer is substituting the NGR instrument for verification purposes.

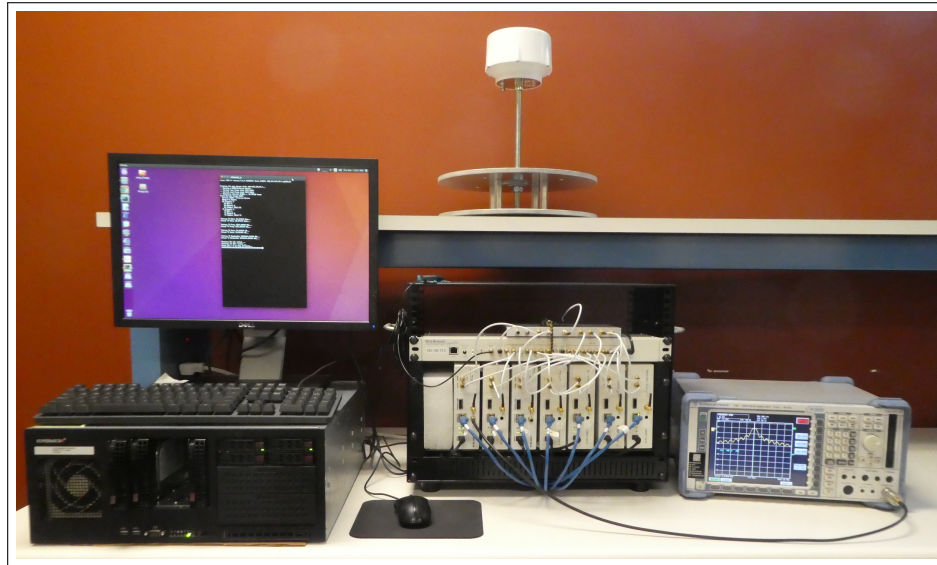


Figure 2.2: Photograph of Benchtop Hardware Test Setup

In this photo, a terminal on the computer displays the output of a USRP driver executing the `tx_samps_threaded` program. Visible on the spectrum analyzer screen is the L1 C/A-coded signal.

The custom GNSS transceiver system was also designed to receive multiple GNSS bands simultaneously. Precise data acquisition is made possible not only by the Octoclock but also a Tallysman VeraPhase 6000 GNSS antenna, shown at the top of Figure 2.2. For all GNSS bands this antenna provides high receive gain, low axial ratios from horizon to horizon, exceptional front-to-back ratios, high efficiency ($> 70\%$), less than 1mm phase center variation, 35mA current at 5V, and multi-path rejection [10]. It is powered by a standard bench-top power supply through its coaxial connection.

This antenna was used to obtain live-sky recordings of GNSS signals to support

receiver development. During the Summer in 2017, a recording was obtained from the ground and another from the roof of the ElectroScience Lab. A photograph of the VeraPhase in use is shown in Figure 2.3.

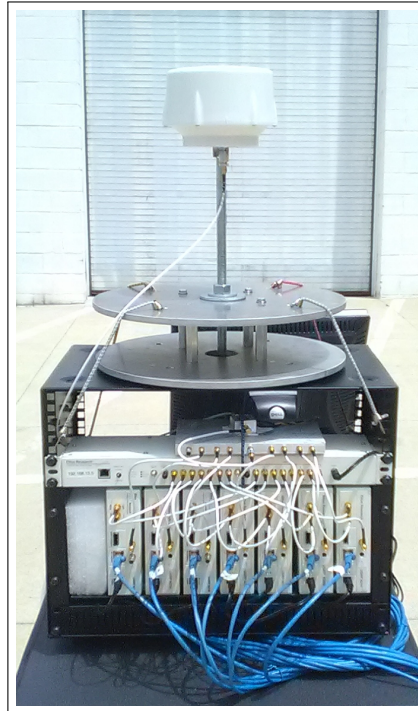


Figure 2.3: Photograph of Antenna Setup for Data Acquisition

2.1.2 Software-Defined Radios

The radios used are Ettus Research model USRP N210 [11]. The N210 was selected for the wide bandwidth and high dynamic range required to produce GNSS signals. The following connections are established to each radio: an ethernet connection to receive waveform samples from the host computer, an input coaxial connection from the clock distribution module for a synchronized 10MHz clock reference and pulse-per-second (PPS) trigger, an input coaxial connection from the receiving antenna, and an output coaxial connection to a combiner. Figure 2.4 contains a photo of the device.



Figure 2.4: USRP N210 Software Defined Radio [11]

Ettus Research provides an extensive library of open-source hardware drivers for the USRP. This platform allows us to customize USRP’s behavior beyond temporary configuration parameters. Modifying the C++ source and compiling the driver library is made simple by the operating system we use, and permitted by Ettus Research through the MIT license [12].

Each USRP contains a daughterboard for receiving or transmitting RF signals. Four USRPs contain the DBSRX2 Receiver [13], which operates from 800MHz to 2300MHz. The other two USRPs contain the WBX Transceiver [14] and are capable of transmitting or receiving from 50MHz to 2200MHz.

The transceiver contains 6 USRPs, turned sideways and stacked in a row in a small rack-mount case. This enables easy access to the front-facing SMA connectors, ethernet ports, and power ports. SMA inputs “clock” and “pps” are connected to the Octoclock which is mounted above the array, and depicted in Figure 2.5.



Figure 2.5: Octoclock Multi-Channel Clock Synchronization Module[9]

2.1.3 Signal Storage

Waveforms are stored as samples in binary files in the host computer's secondary storage. The threaded playback script points the UHD transmit drivers to the `signals` directory to obtain output samples. Two subdirectories contain sequences of files for the two main output bands. Band 1 contains the L1 and E1bc signals and is broadcast at 1.57542GHz. Band 2 contains L5 and E5a signals and is broadcast at 1.17645GHz. Figure 2.6 illustrates the signals directory and file contents.

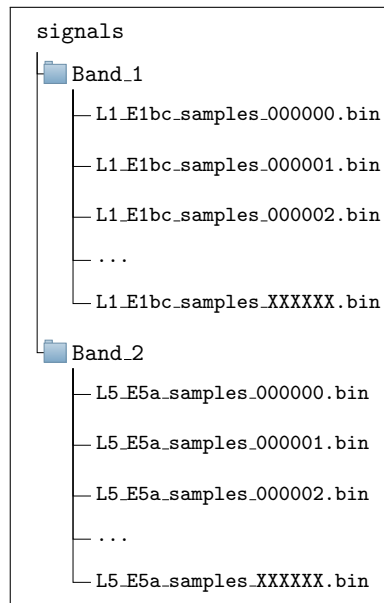


Figure 2.6: Sequence of Sample Files

Note that the files have the same prefix appended with an index (a six-digit string of integers). Threaded waveform playback depends on this sequence format, which follows a pattern of increasing index starting from “0”.

Though only two sequences of test-files exist, they contain a large amount of compressed information that will be combined at output into a single dual-band channel. A receiver connected to this channel that is able to receive both GNSS bands can search for and acquire all 44 PRNs, regardless of signal type.

Samples stored without the associated signal parameters are meaningless, so a

header prepends each binary file to carry this information. This “metadata” enables a USRP or post-processing software to interpret the data meaningfully as a time-valued waveform. Binary samples become meaningful when the sample size (bits) and sampling rate (sps) are known. Binary files become meaningful when the file length (in sampled time) and number of samples is known. A USRP is able to stream samples and modulate them on a carrier with this information, as well as the power factor (gain) with which to scale the samples. Table 2.1 contains a description of the header format.

Waveform File		
Byte Offset	Type	Description
0	uint32	File Type
4	uint8	Sample Type
5	uint32	Sampling Rate
9	uint32	File Length (ms)
13	double	Power Conversion Factor
21	Sample Type	I-value 0
	Sample Type	Q-value 0
	Sample Type	I-value 1
	Sample Type	Q-value 1
	Sample Type	...
	Sample Type	...
	Sample Type	I-value X
	Sample Type	Q-value X

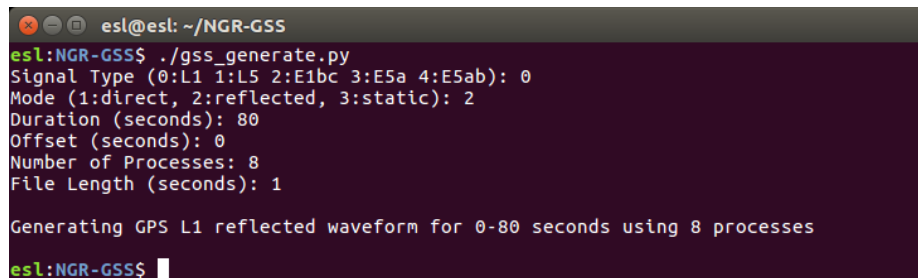
Table 2.1: Waveform File Format

As discussed in Chapter 1, all signals generated by this test bench are BPSK signals. This method indicates the presence of two phases in the digital signal, the in-phase and quadrature-phase components. In this application, each component is sampled individually so the waveforms contain a sequence of alternating I- and Q-samples (each pair sampled at the same instant in time).

2.1.4 Host PC

A host computer is assembled using the Supermicro X9DAi motherboard. It provides 8 processing cores, 10 ethernet ports, 32GB of RAM, and supports RAID storage. Currently it is equipped with a 2TB hard-drive and boots Linux Ubuntu 16.04. The USRP hardware drivers discussed in the previous section are installed and modified for our purposes. The University provides a license to Matlab, which we use to analyze and validate the waveform files. Software development is accomplished using the default editor, C and C++ compilers, and Python interpreter.

A user interacts with the test bench through the terminal. Several Python scripts were written to facilitate signal generation and SDR waveform playback. Figure 2.7 is a screenshot of prompts in the terminal used to poll the user for unique waveform parameters following the command `./gss_generate.py`.



```
esl@esl: ~/NGR-GSS
esl:NGR-GSS$ ./gss_generate.py
Signal Type (0:L1 1:L5 2:E1bc 3:E5a 4:E5ab): 0
Mode (1:direct, 2:reflected, 3:static): 2
Duration (seconds): 80
Offset (seconds): 0
Number of Processes: 8
File Length (seconds): 1

Generating GPS L1 reflected waveform for 0-80 seconds using 8 processes
esl:NGR-GSS$
```

Figure 2.7: Waveform Generator Command Prompt Interface

For the example in Figure 2.7, 80 files are written by 8 processes. The operating system automatically assigns each process to an available CPU. The files are written in parallel so that the time to completion is reduced. Figure 2.8 shows 2 of the 8 processes opened in separate terminals. The processes are independently writing sequences of different files which will be played back in a different order during testing.

```

Starting Process #1
OPEN "../signals/GPS_L1_samples_000000.bin" ...
>>> including prn # 4
>>> including prn # 5
>>> including prn # 12
>>> including prn # 18
>>> including prn # 20
>>> including prn # 21
>>> including prn # 24
>>> including prn # 25
>>> including prn # 26
>>> including prn # 29
>>> including prn # 31
CLOSE "../signals/GPS_L1_samples_000000.bin"
OPEN "../signals/GPS_L1_samples_000008.bin" ...
CLOSE "../signals/GPS_L1_samples_000008.bin" ...
OPEN "../signals/GPS_L1_samples_000016.bin" ...
CLOSE "../signals/GPS_L1_samples_000016.bin" ...
OPEN "../signals/GPS_L1_samples_000024.bin" ...
CLOSE "../signals/GPS_L1_samples_000024.bin" ...
OPEN "../signals/GPS_L1_samples_000032.bin" ...
CLOSE "../signals/GPS_L1_samples_000032.bin" ...
OPEN "../signals/GPS_L1_samples_000040.bin" ...
CLOSE "../signals/GPS_L1_samples_000040.bin" ...
OPEN "../signals/GPS_L1_samples_000048.bin" ...
CLOSE "../signals/GPS_L1_samples_000048.bin" ...
OPEN "../signals/GPS_L1_samples_000056.bin" ...
CLOSE "../signals/GPS_L1_samples_000056.bin" ...
OPEN "../signals/GPS_L1_samples_000064.bin" ...
CLOSE "../signals/GPS_L1_samples_000064.bin" ...
OPEN "../signals/GPS_L1_samples_000072.bin" ...
CLOSE "../signals/GPS_L1_samples_000072.bin" ...
CLOSE "../signals/GPS_L1_samples_000072.bin" ...
Process #1 Done, Returned: 0
press any key to continue ...

Starting Process #2
OPEN "../signals/GPS_L1_samples_000001.bin" ...
>>> including prn # 4
>>> including prn # 5
>>> including prn # 12
>>> including prn # 18
>>> including prn # 20
>>> including prn # 21
>>> including prn # 24
>>> including prn # 25
>>> including prn # 26
>>> including prn # 29
>>> including prn # 31
CLOSE "../signals/GPS_L1_samples_000001.bin"
OPEN "../signals/GPS_L1_samples_000009.bin" ...
CLOSE "../signals/GPS_L1_samples_000009.bin" ...
OPEN "../signals/GPS_L1_samples_000017.bin" ...
CLOSE "../signals/GPS_L1_samples_000017.bin" ...
OPEN "../signals/GPS_L1_samples_000025.bin" ...
CLOSE "../signals/GPS_L1_samples_000025.bin" ...
OPEN "../signals/GPS_L1_samples_000033.bin" ...
CLOSE "../signals/GPS_L1_samples_000033.bin" ...
OPEN "../signals/GPS_L1_samples_000041.bin" ...
CLOSE "../signals/GPS_L1_samples_000041.bin" ...
OPEN "../signals/GPS_L1_samples_000049.bin" ...
CLOSE "../signals/GPS_L1_samples_000049.bin" ...
OPEN "../signals/GPS_L1_samples_000057.bin" ...
CLOSE "../signals/GPS_L1_samples_000057.bin" ...
OPEN "../signals/GPS_L1_samples_000065.bin" ...
CLOSE "../signals/GPS_L1_samples_000065.bin" ...
OPEN "../signals/GPS_L1_samples_000073.bin" ...
CLOSE "../signals/GPS_L1_samples_000073.bin" ...
CLOSE "../signals/GPS_L1_samples_000073.bin" ...
Process #2 Done, Returned: 0
press any key to continue ...

```

(a) Process 1

(b) Process 2

Figure 2.8: File Writing Output for Debugging

Though this work could be completed without output to the user, the development and debugging process is made simpler by a stream of significant job responsibilities. The terminals display which files are written by which processes, which PRN codes are included in the waveforms, and indicates when errors occur during execution. Figure 2.8 shows that Process 1 is writing every eighth file starting from 0 (0,8,16,...) and Process 2 is writing every eighth file starting from 1 (1,9,17,...).

The Python interface also facilitates waveform playback. For example, to modulate a waveform stored as a sequence of binary files the user executes the command `./gss_mono_TX_threaded.py <Band #>`. The script communicates with the USRP drivers for SDR configuration and to establish a channel for transferring waveform samples. “Threaded” file playback is made possible by modifications to USRP hard-

ware driver software, causing the program to read from a numbered sequence of files instead of only one. A screen shot of terminal output is shown in Figure 2.9 where an 80 second sequence of waveform samples is being modulated by a single USRP.



```
Reading file into buffer: signals/GPS_L1_samples_000067.bin.
Reading file into buffer: signals/GPS_L1_samples_000068.bin.
Reading file into buffer: signals/GPS_L1_samples_000069.bin.
Reading file into buffer: signals/GPS_L1_samples_000070.bin.
Reading file into buffer: signals/GPS_L1_samples_000071.bin.
Reading file into buffer: signals/GPS_L1_samples_000072.bin.
Reading file into buffer: signals/GPS_L1_samples_000073.bin.
Reading file into buffer: signals/GPS_L1_samples_000074.bin.
Reading file into buffer: signals/GPS_L1_samples_000075.bin.
Reading file into buffer: signals/GPS_L1_samples_000076.bin.
Reading file into buffer: signals/GPS_L1_samples_000077.bin.
Reading file into buffer: signals/GPS_L1_samples_000078.bin.
Reading file into buffer: signals/GPS_L1_samples_000079.bin.
Reading file into buffer: signals/GPS_L1_samples_000001.bin.
Reading file into buffer: signals/GPS_L1_samples_000002.bin.
Reading file into buffer: signals/GPS_L1_samples_000003.bin.
Reading file into buffer: signals/GPS_L1_samples_000004.bin.
Reading file into buffer: signals/GPS_L1_samples_000005.bin.
Reading file into buffer: signals/GPS_L1_samples_000006.bin.
Reading file into buffer: signals/GPS_L1_samples_000007.bin.
Reading file into buffer: signals/GPS_L1_samples_000008.bin.
Reading file into buffer: signals/GPS_L1_samples_000009.bin.
Reading file into buffer: signals/GPS_L1_samples_000010.bin.
Reading file into buffer: signals/GPS_L1_samples_000011.bin.
Reading file into buffer: signals/GPS_L1_samples_000012.bin.
Reading file into buffer: signals/GPS_L1_samples_000013.bin.
Reading file into buffer: signals/GPS_L1_samples_000014.bin.
Reading file into buffer: signals/GPS_L1_samples_000015.bin.
Reading file into buffer: signals/GPS_L1_samples_000016.bin.
Reading file into buffer: signals/GPS_L1_samples_000017.bin.
Reading file into buffer: signals/GPS_L1_samples_000018.bin.
Reading file into buffer: signals/GPS_L1_samples_000019.bin.
^CFinished reading file.
Finished transmitting

Done!
press any key to continue ...
```

Figure 2.9: Threaded File Playback Using Custom UHD Interface Program

Though direct access to parameters passed to the hardware driver are abstracted from the user by the Python script, the parameters may be changed by editing the script with no need to compile the hardware driver. For instance, the value passed for the driver parameter “-repeat” can be changed to zero in the script before executing it again to avoid starting back at the beginning of the sequence.

2.2 Software

Test bench software fulfills three primary functions, the user interface, waveform playback, and waveform assembly. Though a description of these systems is included in Subsection 2.1.4 a detailed analysis is developed here, including software depen-

dencies, implementation details, and algorithms. The user interface is created from scratch to facilitate access to the other two systems. Waveform playback is made possible by modifications to hardware driver software. Waveform assembly is carried out by an enhanced version of an executable program created in 2014. Important files and folders in the working software directory are depicted in Figure 2.10.

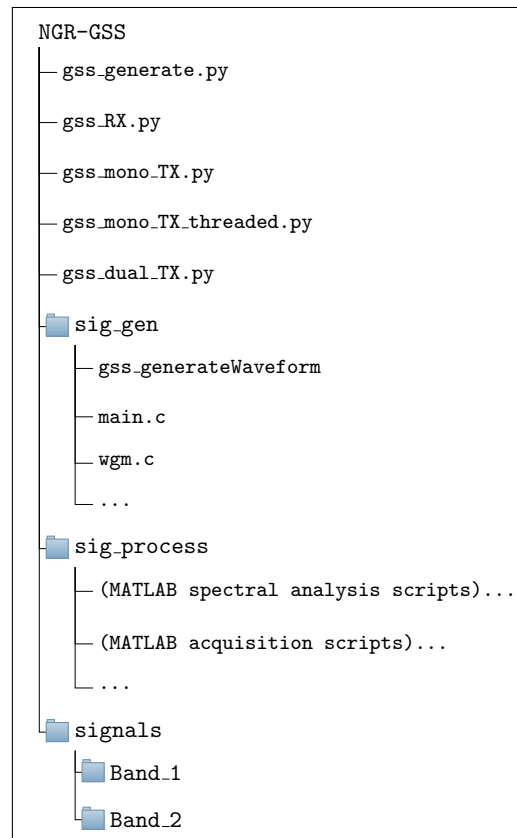


Figure 2.10: User Interface and Working Software Development Directory Contents

2.2.1 User Interface

A user interacts with the test bench by navigating to the `NGR-GSS` directory in Figure 2.10 and executing scripts from the command line in the following format:

```
$ ./<script_name>.py <args>
```

The signal generator script prompts the user for raw input to establish the signal type (L1, L5, E1bc, or E5a), mode of operation (direct, reflected, or static), duration

of sequence (seconds), offset from beginning of sequence (seconds), the number of parallel processes, and the length per file (seconds). Based on the signal type, this script reads from a dictionary (or map) the output file path and sampling rate, calculates the total number of files, and opens a separate terminal to execute each process simultaneously. A detailed description of the program executed by this script follows in section 2.2.2. Figure 2.11 contains a flowchart signal generator interface behavior.

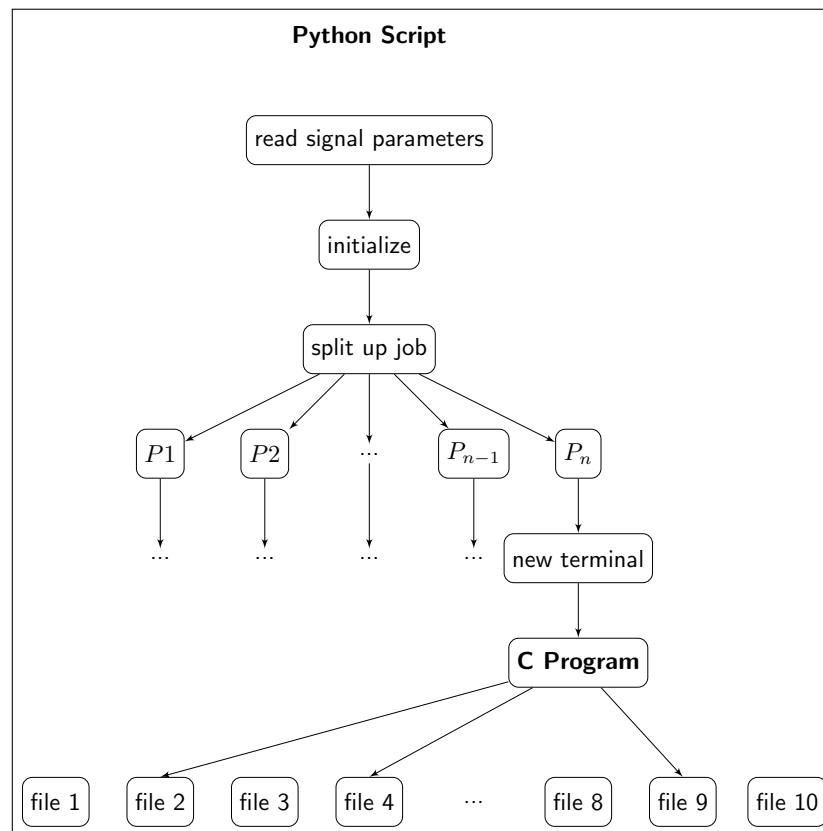


Figure 2.11: Python Program

Several scripts are written for various tasks required of the transceiver. The receiver script identifies the USRPs by IP address and configures them to receive different signals. Currently the receiver script is set up to receive the following bands: GLONASS bands G1 and G2; BeiDou B2; GPS L1, L2, and L5; and Galileo E6. It does so by assigning each USRP a frequency, gain, sampling rate, bandwidth, file path, and recording duration. A separate terminal is opened for each USRP

where a modified version of the UHD program `rx.samples.to.file` is executed. Our modifications cause each process to “agree” to wait until the same arbitrary future time to begin recording so that the incoming samples are synchronized.

Transmitter scripts configure USRPs in the same way, assigning each USRP a frequency, gain, sampling rate, bandwidth, and file path based on the signal type passed as an argument. Those USRPs with transmit cards have configuration parameters stored in dictionaries (maps) which are either selected or populated before being passed as arguments to a UHD program.

2.2.2 Signal Generator

A C program is developed to calculate and store time-domain samples of simulated GNSS signals. It is based on software written for the first NASA CYGNSS mission testing, and it has been enhanced with new features including support for parallel processing, inclusion of realistic satellite metadata, and the ability to generate additional GNSS bands. The compiled program is named `gss_generateWaveform` and is created from a makefile in the `sig_gen` directory. Main components of the program are shown in Figure 2.12 below. Each component’s behavior is described as an algorithm later in this section.

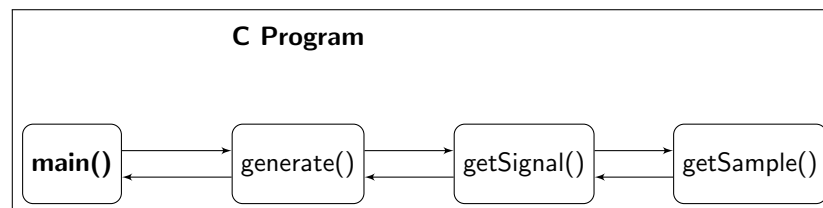


Figure 2.12: Primary C Program Functions

One executing thread of this program can generate a sequence of waveform files for a single signal type (L1, E5, etc.) containing up to eleven PRN codes. Further processing is required to combine sequences and a hardware combiner is required for multi-band simulation. However, we will begin by discussing the assembly of

waveforms at a low level and in the time domain.

The time-domain description of a general GPS C/A-coded signal can be expressed as

$$s(t) = \sqrt{2P}D(t)x(t)\cos(2\pi ft + \theta) \quad (2.1)$$

where $\sqrt{2P}$ is the amplitude, $x(t)$ is the spread spectrum PRN code, $D(t)$ is navigation data, and $\cos(2\pi ft + \theta)$ is the RF carrier at frequency f Hz and phase θ° [15]. The PRN code and navigation data take on binary values and are modulated by phase-shift keying, as described in Section 1.1.1. Other GNSS signals have alternative forms, with GPS L5 containing both I and Q components in a QPSK modulation and Galileo E1bc and E5 being BOC modulated.

The signal generator relies on input files which contain pre-generated PRN codes, navigation data, position and velocity data for the satellites and specular points, and other metadata for generating phase and doppler offset. The PRN codes and position and velocity data were produced by members of this research team, and the navigation message and metadata are realistic simulations provided by Navsys. PRN and navigation sequences are repeated so we only need to load one period into main memory, but the metadata used here is from a real twenty-five hour recording from a satellite and is too large to be stored in RAM. This data is buffered regularly during computation of the samples instead.

A significant portion of software development targets file-writing efficiency of the signal generator. We are limited by the data transfer speeds of the secondary storage devices as well as the steps involved in computing each sample value in code. Our discussion of the algorithm has yet to occur, but it is worth noting here that it takes much more than one second in real-time to generate “one-second” of waveform samples. In fact, the smallest time ratio achieved was 18.45:1. So generating a 1-

second long sequence of samples requires approximately 19 seconds to complete using all 8 CPU cores. It has been concluded that the main “bottleneck” of signal generation is writing to secondary storage.

A discussion of the waveform generator’s main functions follows, outlined as algorithms in pseudo-code for brevity. Each algorithm is implemented as a function in C to create the waveform generator program depicted in Figure 2.12. The compiled program `gss_generateWaveform` also includes other auxiliary functions not discussed in detail in this section. The program is executed through the Python interface.

The entry-point of program execution is the function `main()`. Its purpose is to interpret arguments passed from the user interface and initialize waveform generation. Algorithm 1 provides an outline of `main()` function execution.

Algorithm 1 Signal Generator `main()` Function

```

1: procedure MAIN(void)
2:   Parse Python Arguments
3:   Initialize Variables
4:   Initialize Positions and Velocities
5:   if (mode = static) then
6:     Allocate PV Buffers for Receiver, Specular Point, and GPS Transmitter
7:     Fill Buffers with Constant Cartesian Values
8:   else
9:     Allocate PV Buffers for Receiver, Specular Point, and GPS Transmitter
10:    Read Positions and Velocities ← PV File
11:   Check for Errors
12:   Call generate()
13:   return

```

The first step is parsing input arguments. These include signal type, duration, the number of processes, the current process, mode of receiver operation, a sampling rate, and file paths and names. This information is used to initialize the parameters that `main()` will pass to `generate()`. Based on the mode of operation, a geometry is initialized to define the positions and velocity vectors of the receiver, GNSS transmitter, and the specular point. Geometries are defined in three-dimensional Cartesian coordinates. Then an error check is performed to do a sanity check on variables and

to ensure that input files exist and can be opened before the variables are passed to `generate()`.

At this point the program has established which signal is being generated, the geometry of the satellites, and the duration of the simulation. The call to `generate()` begins computation of waveform samples and writing to output files. This function is outlined in Algorithm 2. It coordinates the assembly of a time-domain signal, expressed in Equation 2.1, by generating and writing samples in “blocks”.

Algorithm 2 Signal Generator `generate()` Function

```

1: procedure GENERATE(...)
2:   Initialize Variables
3:   Calculate # Blocks / Samples
4:   Assemble File Header
5:   Load Chip Tables
6:   Obtain DDM Pattern
7:   Allocate Memory
8:   Set Carrier Frequency & Chip Rate
9:   Check for Errors
10:  Calculate Velocity from Position
11:  Calculate Delay & Doppler from Transmitter & Receiver
12:  for (each block) do                                ▷ Each Second
13:    Update CAC Data
14:    Track Doppler for Skipping Samples                ▷ Separate File Assignments
15:    if (responsible for file) then
16:      if (beginning of second) then
17:        Open Output File
18:        Write Header
19:        for (# samples per file) do
20:          Interpolate Delay-Doppler Offset to Each Sample
21:          Allocate a Signal Buffer
22:          call getSignal()                            ▷ (A Block of Samples at a Time)
23:          Write Samples to Output File
24:        else
25:          Close Output File
26:    Free Memory
27:  return

```

This function first initializes the environment (Lines 2 through 11), then remains in a loop for the remainder of its execution to generate and write the samples. Initializa-

tion involves calculating number of samples and blocks, allocating memory, opening files for reading in auxiliary data, and calculating radar geometries.

In the main loop (Line 12) the function writes one block of samples at a time to an output file. For those files the process is not responsible for, the program skips writing those samples and keeps track of the changing delay and Doppler offset over the period of skipped samples (Line 14). For files the process must write, it ensures that the appropriate file stream is open, writes a header to the file, calculates one block of samples with a call to `getSignal()`, and writes the samples to the file. Upon completion, the function ensures all files are closed and frees all allocated memory.

To obtain a block of samples, a call to `getSignal()` is made. This function is outlined in Algorithm 3.

Algorithm 3 Signal Generator `getSignal()` Function

```

1: procedure GETSIGNAL(...)
2:   Clear Signal Buffer
3:   Latch # Space-Vehicles  $\leftarrow$  CAC File
4:   for (each space vehicle) do                                      $\triangleright$  Each PRN
5:     Latch Other Metadata  $\leftarrow$  CAC File
6:     print included PRNs
7:     Track Hot-Spot Accumulated Phase
8:     for (# samples in block) do
9:       Convert CAC Data into Delay and Phase Information
10:      Call getSample()
11:   Free Memory
12:   return

```

This function begins by filling the signal buffer with zeros, then it loops over the number of space vehicles. For each one it incorporates the vehicle's associated metadata as delay and phase information for that sample and calls `getSample()`. Note that the signal buffer is not cleared for each iteration of the loop for space vehicles. This effectively sums the individual contributions of each reflection to the received power in the signal (amplitude).

The function `getSample()` is outlined in Algorithm 4. This function calculates

both the in-phase and quadrature-phase components for a single waveform sample. For a given signal type (Line 4, 6, 8, or 10), the components are calculated based on delay and phase information as well as the current value of the chipping sequence (obtained from an array of chips loaded at the beginning of `generate()`). The I and Q values are then added to the existing value in the signal buffer, which accounts for the contributions of other space vehicles as well.

Algorithm 4 Signal Generator `getSample()` Function

```

1: procedure GETSAMPLE(...)
2:   Initialize Variables
3:   switch (signal) do
4:     case (L1)
5:       Accumulate I and Q Components           ▷ L1 Signal Plan
6:     case (L5)
7:       Accumulate I and Q Components           ▷ L5 Signal Plan
8:     case (E1bc)
9:       Accumulate I and Q Components           ▷ E1bc Signal Plan
10:    case (E5)
11:      Accumulate I and Q Components           ▷ E5a Signal Plan
12:    Add I and Q Components to Signal Buffer
13:  return

```

2.3 Signal Validation

Several techniques are developed to verify the correctness of test bench signals. This is done to prevent damage to the receiver when tested and to ensure that the signals accurately simulate the receiver's future space environment. Analysis tools are developed in Matlab which read samples from the files, plot the time-domain samples, and calculate the discrete Fourier transform to obtain spectrum plots as well. Matlab is also used to perform signal acquisition, a technique also used by the receiver to identify the individual satellite signals. Acquisition is performed in validation to visually and quantitatively verify space vehicle dynamics and proper code-modulation. Finally, a spectrum analyzer is used to obtain frequency-domain

plots of USRP output. These plots are compared to those generated in Matlab and used to confirm each band's carrier frequency.

Chapter 3

Results

This chapter discusses the collection of preliminary data and also show the contents of the simulated GNSS signal files to prove that our test bench design produces signals appropriate for injection into the receiver for testing. An analysis of the waveform samples in the time and frequency domains is included in Section 3.1, then verification of individual satellite signals through PRN acquisition and tracking of a signal over a significant duration of time. Finally the output spectrum is visualized for RF validation.

3.1 Waveform Analysis

The two main GNSS bands are analyzed in this section. Band 1 contains GPS L1 and Galileo E1bc signals and Band 2 contains GPS L5 and Galileo E5a signals. Each component of a band is analyzed alone first before signal combination. The signals in this section involve a single space vehicle so that the time-domain plots are recognizable.

3.1.1 Band 1: L1 and E1bc

Figure 3.1 shows the first 50 μs of samples for a solitary GPS L1 satellite signal. Both the in-phase and quadrature-phase components start at a non-zero initial am-

plitude, indicating some delay in time (chips). A sinusoidal envelope indicates the presence of Doppler shift. Though the navigation modulation is not visible, the PRN sequence is clearly indicated by 180° phase shifts about $1\mu s$ apart. To ensure that this minimum period (maximum chip frequency) approximates the expected frequency in Appendix A,

$$f_{chip} = \frac{1}{T} = \frac{1}{1\mu s} = 1MHz \approx 1.023MHz. \quad (3.1)$$

It also appears that the $50\mu s$ domain spans slightly less than one-half a full period of the Doppler envelope. Verifying a reasonable Doppler shift,

$$\frac{1}{2} \times T \approx 45\mu s \rightarrow f_{doppler} \approx 9kHz. \quad (3.2)$$

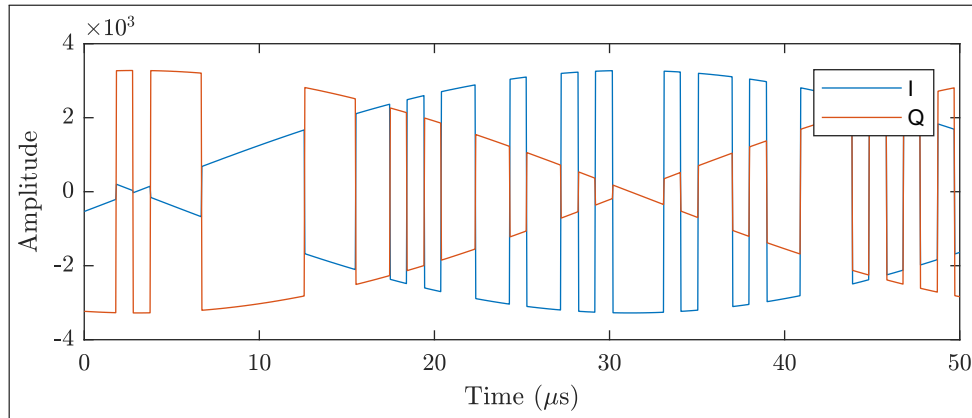


Figure 3.1: GSP L1 C/A-Code Samples

The domain in Figure 3.1 is not sufficiently large to observe a change in doppler, nor is it practical to observe a delay. This analysis is postponed to Subsections 3.2 and 3.3 where the samples are analyzed over a larger time-span and the beginning of the code sequence is identified.

Figure 3.2 shows a normalized frequency plot obtained by calculating the FFT. It clearly illustrates the L1 C/A code spectrum, with side lobes near -13dB.

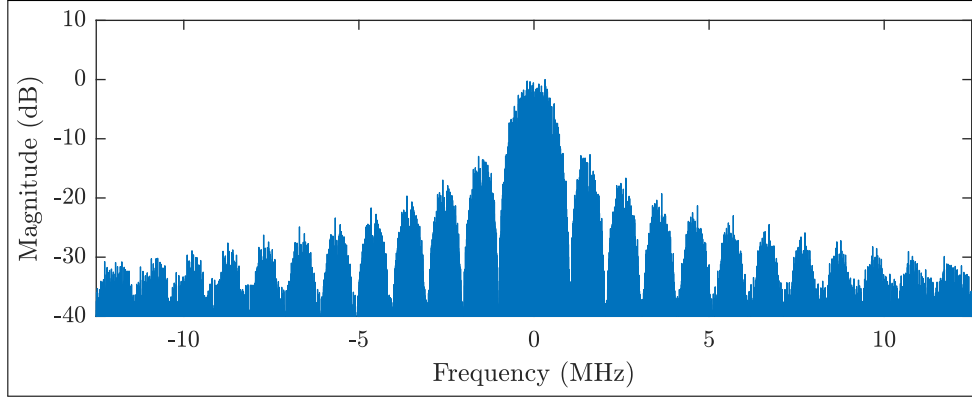


Figure 3.2: GSP L1 C/A-Code Spectrum

Figure 3.3 contains a plot of Galileo E1bc samples. It shows 180° phase separation between the I and Q components and also the existence of a lower magnitude, higher frequency sub-carrier pilot signal. Verifying the low-frequency rate,

$$f_1 = \frac{1}{T_1} = \frac{1}{1\mu s} = 1MHz \approx 1.023MHz. \quad (3.3)$$

The high-frequency switching appears to be five to ten time faster. So estimating,

$$f_{chip} = \frac{1}{T} = \frac{1}{(1/7)\mu s} = 7MHz \approx 6.138MHz. \quad (3.4)$$

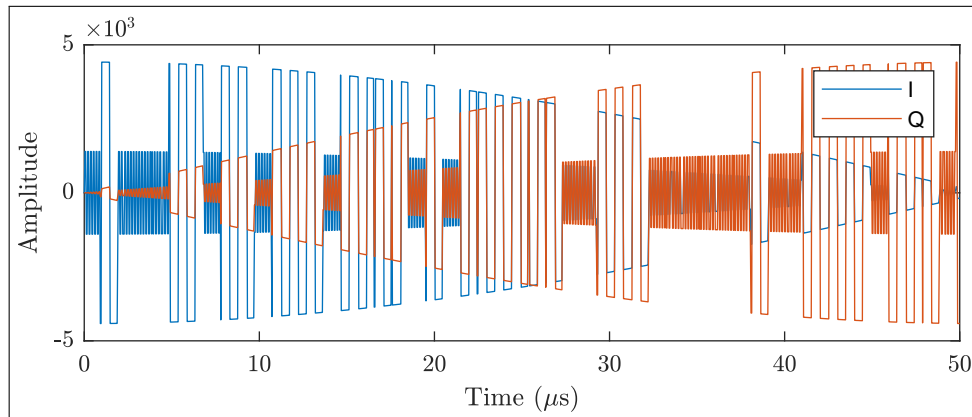


Figure 3.3: Galileo E1bc Samples

The frequency domain plot in Figure 3.4 shows two peaks, as expected for the

CBOC chipping sequence. First side lobes are nearly -13dB.

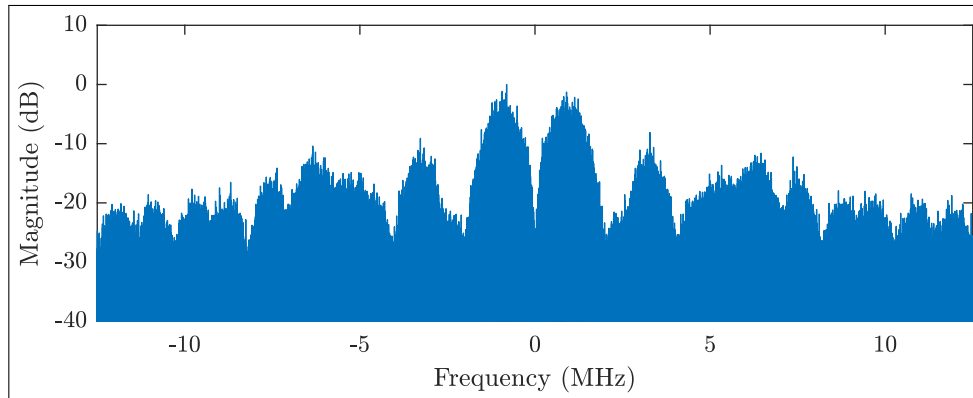


Figure 3.4: Galileo E1bc Spectrum

To combine GPS L1 and Galileo E1bc, the time domain samples are added together. That is, since the waveform files are stored at the same sampling rate we can iterate the files and add each sample together element-wise. This eliminates the need to combine L1 and E1bc signals with a hardware RF combiner.

A plot of the combined samples is shown in Figure 3.5. It is less obvious from this plot what the individual chipping rates are, but it is possible to observe the 180° phase shift between I and Q components. Also, the maximum value of the samples has increased because some samples have added constructively. We also observe Band 1 samples with very small absolute values where the summation was destructive.

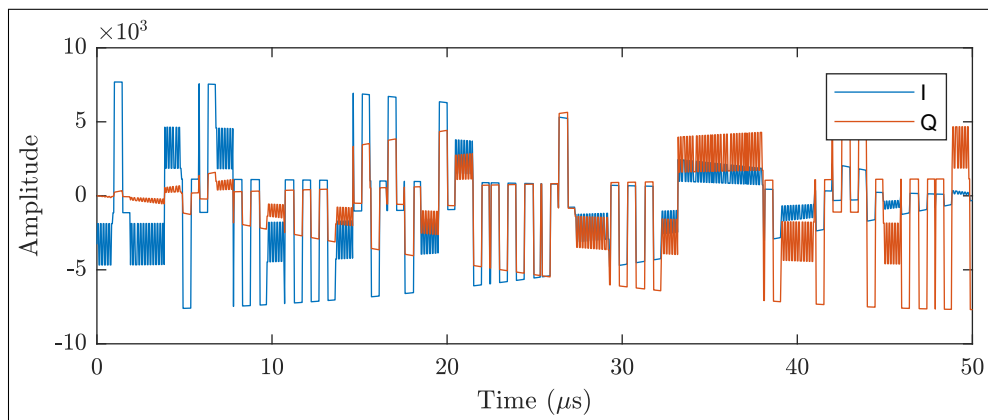


Figure 3.5: Band 1 Samples and Spectrum

By linearity, the DFT of time-domain samples is also the sum of each signal's DFT. This is verified in Figure 3.6, a plot of Band 1 in the frequency-domain.

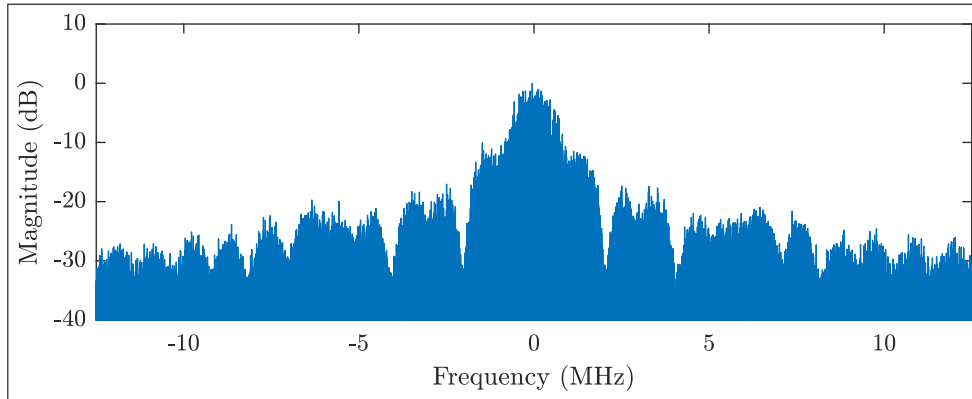


Figure 3.6: Band 1 Spectrum

3.1.2 Band 2: L5 and E5a

Figure 3.7 shows the first $50 \mu s$ of time-domain samples for the GPS L5 signal. We expect a chipping rate of 10.23 MHz , and observe that nearly one-hundred phase shifts occur in a $10 \mu s$ time-span. Verifying the chipping rate,

$$f_1 = \frac{1}{T_1} = \frac{100}{10 \mu s} = 10 \text{ MHz} \approx 10.23 \text{ MHz}. \quad (3.5)$$

I and Q components are 180° out of phase and $50 \mu s$ spans just less than one half a period of the Doppler envelope, so,

$$\frac{1}{2} \times T \approx 45 \mu s \rightarrow f_{\text{doppler}} \approx 9 \text{ kHz}. \quad (3.6)$$

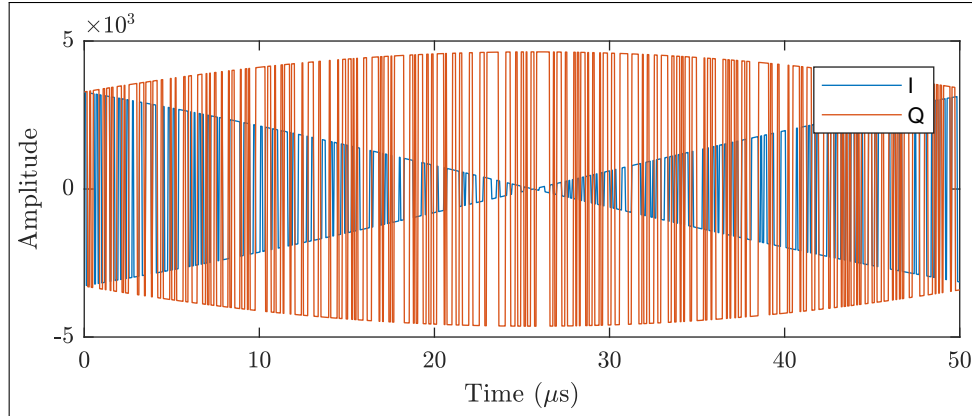


Figure 3.7: GPS L5 Samples

Figure 3.8 is a plot of GPS L5 in the frequency-domain. We observe a main lobe that is much wider than L1 in Figure 3.2 or E1bc in Figure 3.4. This is due to frequency-domain expansion brought on by a narrower chipping pulse. Figure 3.8 shows a main lobe that is approximately 20MHz-wide, ten times greater than the width of L1's main lobe in Figure 3.2 (approximately 2MHz wide).

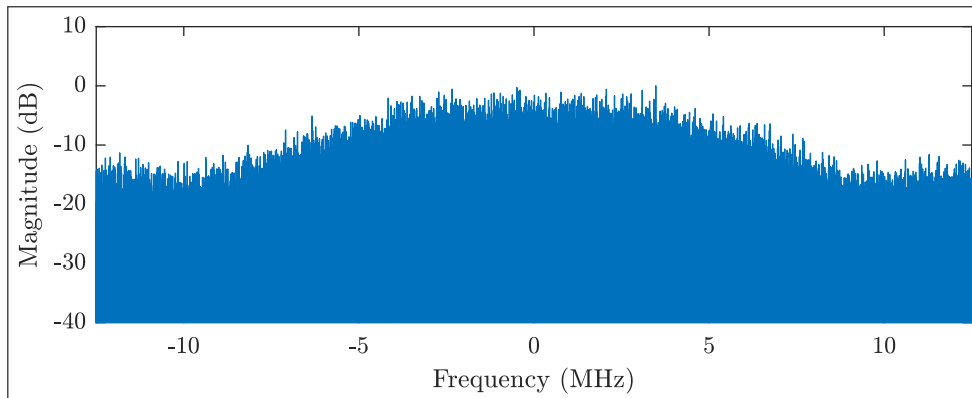


Figure 3.8: GPS L5 Spectrum

The samples of the E5a signal are shown in Figure 3.9. The plot is nearly indistinguishable from Figure 3.7 because the signals are very similar. As shown in Appendix A, their properties are nearly identical except for orbit parameters and the method of code modulation. E5a uses AltBOC and L5 uses BPSK.

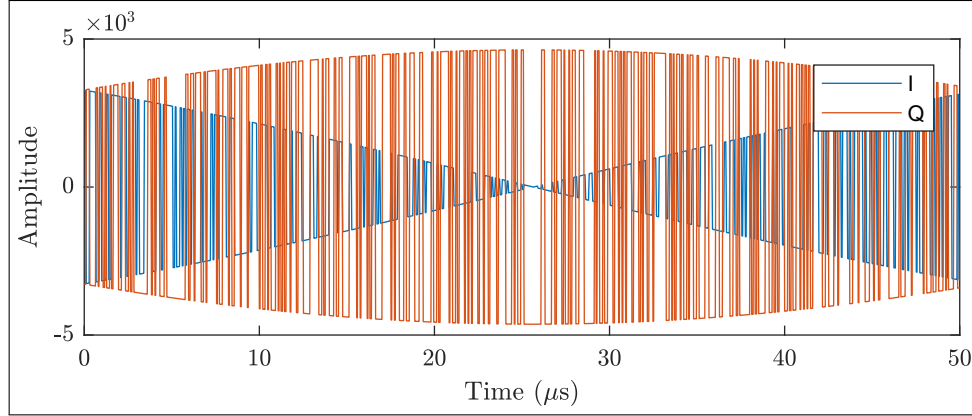


Figure 3.9: Galileo E5a Samples

Approximating the chip frequency, we estimate 100 chips per $10\mu s$,

$$f_{chip} = \frac{1}{T} = \frac{100}{10\mu s} = 10MHz \approx 10.23MHz. \quad (3.7)$$

I and Q components are 180° out of phase and $50\mu s$ spans just less than one half a period of the Doppler envelope, so,

$$\frac{1}{2} \times T \approx 45\mu s \rightarrow f_{doppler} \approx 9kHz. \quad (3.8)$$

The frequency domain is expectedly similar to L5 as well. It is plotted in Figure 3.10.

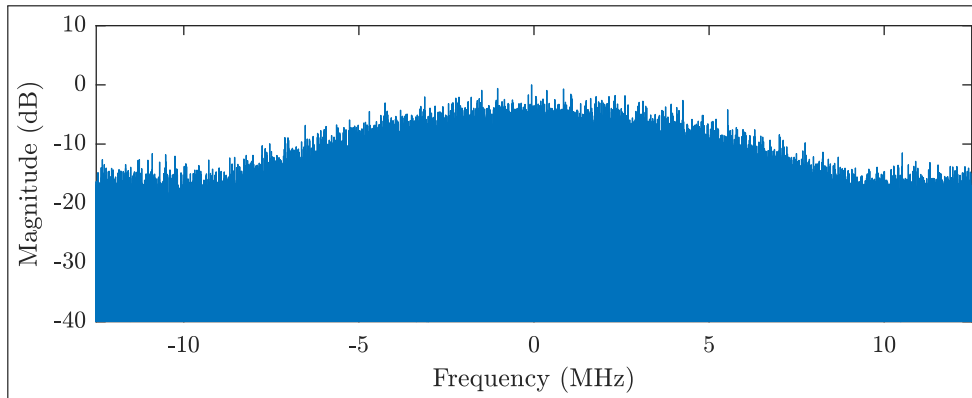


Figure 3.10: Galileo E5a Spectrum

GPS L5 and Galileo E5a signals are combined to create Band 2 using the same

method described in Subsection 3.1.1, with the element-wise summation of samples. A time-domain plot of this band is in Figure 3.11. We observe that some samples add constructively, resulting in a large magnitude. Others add destructively and sum to zero.

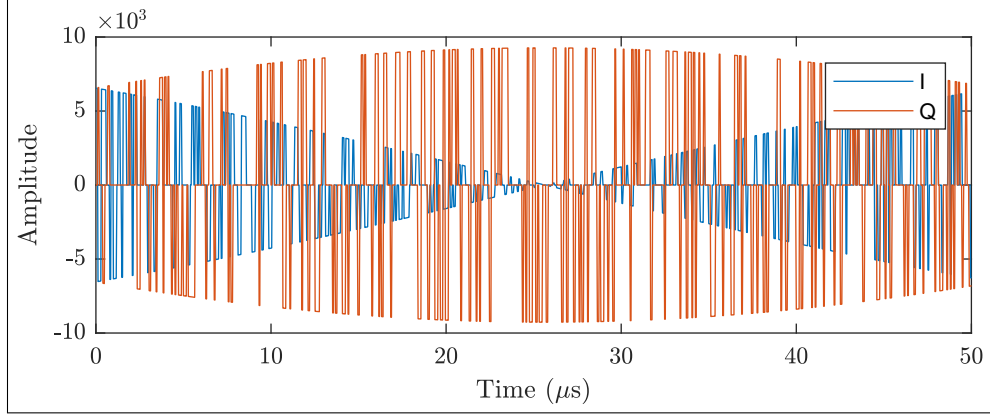


Figure 3.11: Band 2 Samples

Estimating the chip frequency,

$$f_{chip} = \frac{1}{T} = \frac{100}{10\mu s} = 10MHz \approx 10.23MHz. \quad (3.9)$$

The sample plot has maintained its frequency characteristics, with in-phase and quadrature-phase components separated by 180° . The doppler shift is also still apparent,

$$\frac{1}{2} \times T \approx 45\mu s \rightarrow f_{doppler} \approx 9kHz. \quad (3.10)$$

A frequency-domain plot is shown in Figure 3.12. It is visually indistinguishable from its component signals L5 and E5a, maintaining a 20MHz-wide main lobe and side-lobes at -13dB.

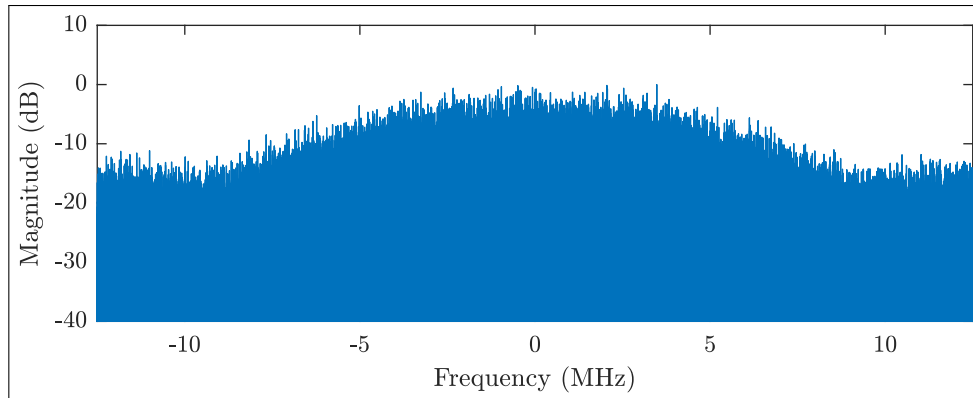


Figure 3.12: Band 2 Spectrum

3.2 Signal Acquisition

To verify that individual satellite signals are present in the simulations, signal acquisition was performed in Matlab to search for each PRN. These acquisitions were carried out on deliverable Band-1 and Band-2 signals: reflected mode, full-power, and containing eleven PRNs from each of their component signals. Only the first file was processed from each sequence, so the results in this section are representative of test bench initial conditions.

3.2.1 GPS L1

Initial delay, Doppler offset, and peak-to-mean ratio for all GPS L1 PRNs found through acquisitions are shown in Table 3.1 and Figure 3.13.

Band 1 L1 Acquisitions			
PRN	Doppler(kHz)	Delay(Chips)	Peak to Mean Ratio(dB)
4	8.9	141.8	20.8
5	13.5	46.4	20.6
12	19.1	745.0	20.5
18	14.8	8.9	20.3
20	19.0	894.6	20.4
21	-0.2	1018.3	20.9
24	16.5	270.6	20.2
25	9.0	565.1	21.0
26	-11.5	954.5	20.6
29	-3.3	201.0	20.7
31	-10.9	908.1	20.3

Table 3.1: Initial DDM Parameters for Band 1 L1 Acquisitions

These values indicate the presence of eleven GPS satellites broadcasting L1 C/A-coded signals. The satellites are a variety of distances from the receiver and are traveling at various relative speeds. The magnitudes of Doppler shifts exist in the range -20kHz to 20kHz, delay expectedly varies between 0 and 1023 chips, and the peak-to-mean ratio varies between 20dB and 21dB. Scaled-color images are shown in Figure 3.13. They show a windowed region of the acquisition space around the peak of the ambiguity function obtained for each PRN.

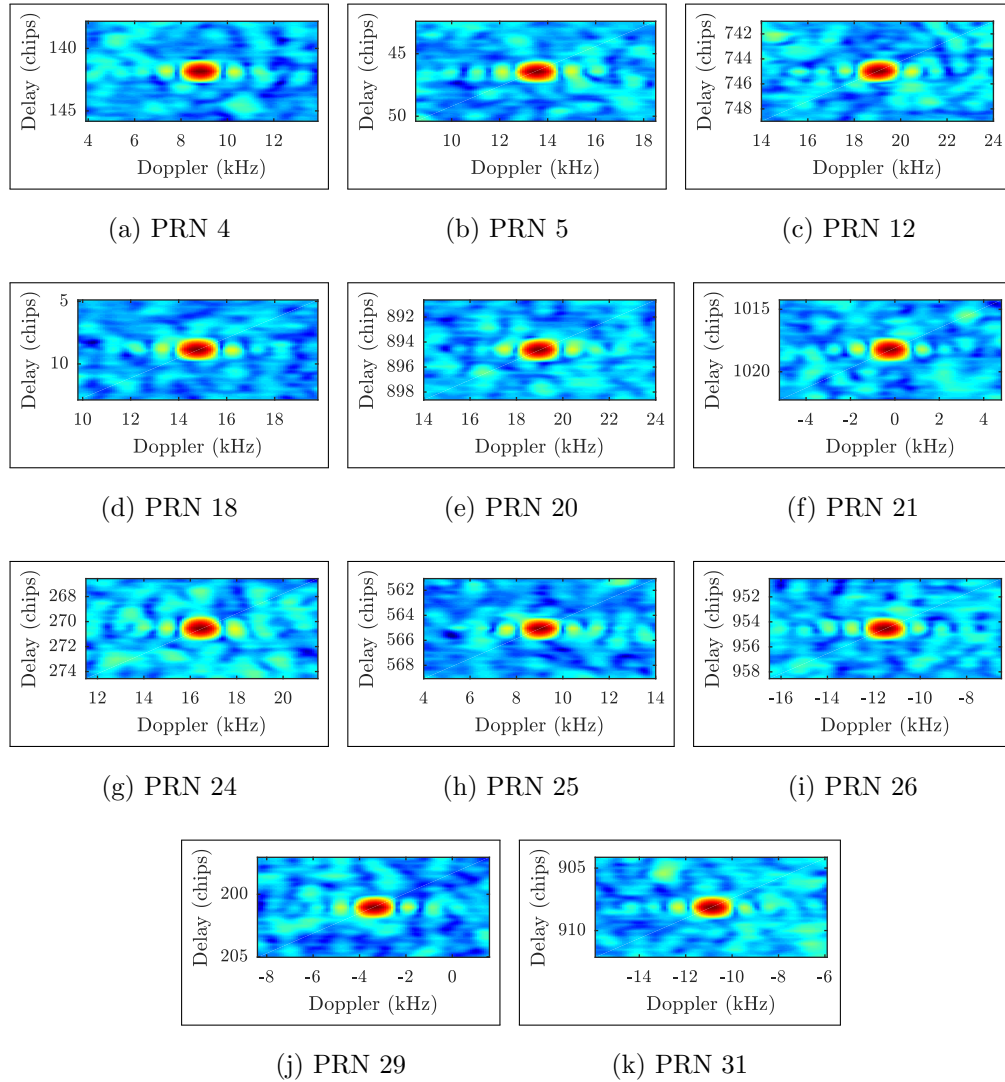


Figure 3.13: Visually Confirmed Initial L1 PRN Acquisitions

The color-scale images in this section show large values in red and small values in blue. Figure 3.13 confirms the successful acquisition of all eleven PRNs for this signal.

3.2.2 Galileo E1bc

Table 3.2 contains acquisition data for Galileo E1bc. Doppler shift magnitudes are less than 30kHz, delays are between 30 chips and 1000 chips, and the peak-to-mean ratio varies.

Band 1 E1bc Acquisitions			
PRN	Doppler(kHz)	Delay(Chips)	Peak to Mean Ratio(dB)
4	6.7	991.0	8.7
5	18.7	732.5	14.3
12	24.3	408.0	13.5
18	19.9	695.0	13.5
20	-25.5	442.0	8.7
21	20.6	32.0	8.8
24	-0.7	434.0	8.6
25	14.2	228.0	14.1
26	-6.3	617.5	12.0
29	28.5	734.5	8.7
31	-5.8	571.0	13.8

Table 3.2: Initial DDM Parameters for Band 1 E1bc Acquisitions

The peak-to-mean ratio lies in two distinct ranges, indicating several failed acquisitions. Since for PRN 4, 20, 21, 24, and 29, the peak-to-mean ratio is nearly 9dB, the peak is actually near noise level. We do not consider this a successful acquisition. However, six of the space vehicles were identified and can also be visually verified in Figure 3.14.

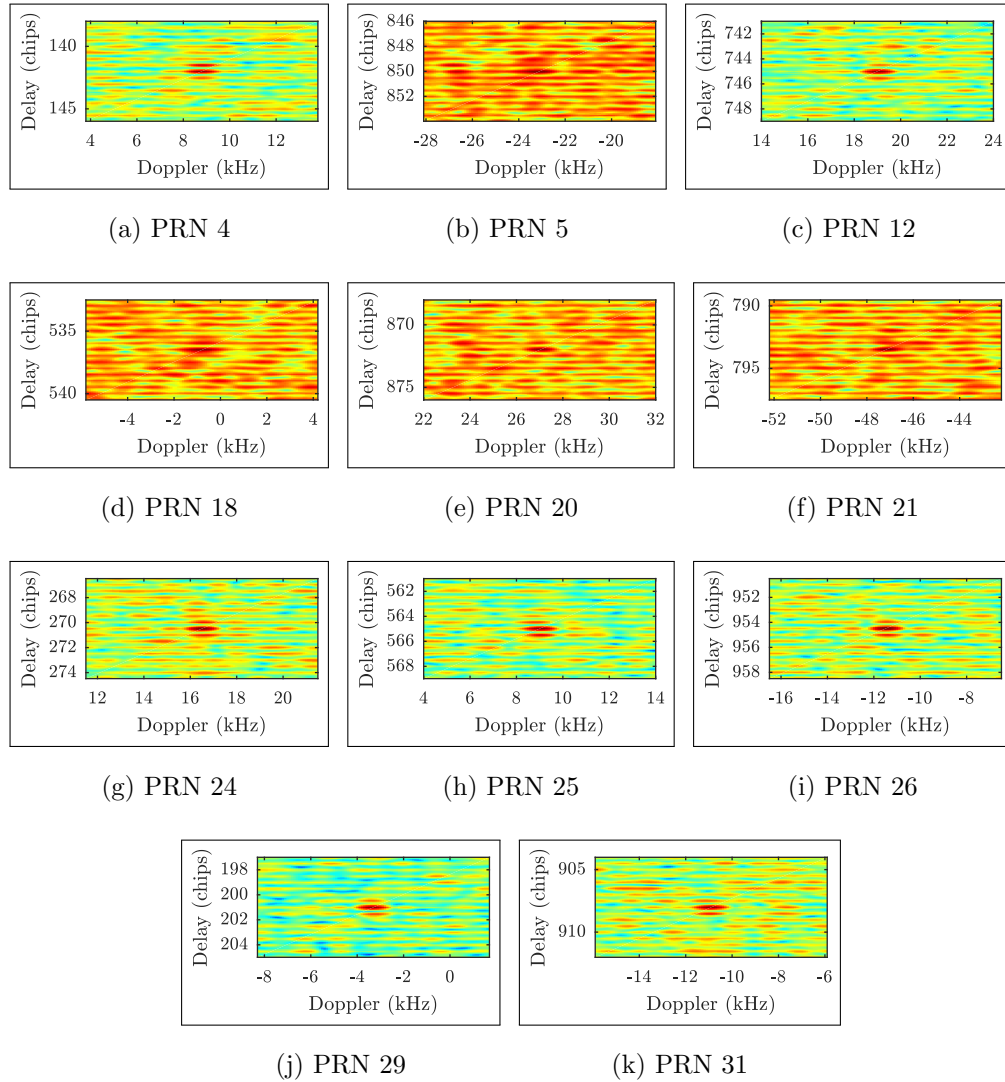


Figure 3.14: Visually Confirmed Initial E1bc PRN Acquisitions

Plots 3.14(b), 3.14(d), 3.14(e), and 3.14(f) show more red data-points because the peak in the ambiguity function was found very near the noise level. For these PRNs, the peak-to-mean ratio is approximately 8.5dB. It is reasonable to assume that failures in acquisition occur because errors exist in the acquisition script, rather than the C program. Failed PRN modulation in the signal generator would likely prevent even a single successful acquisition in this analysis.

3.2.3 GPS L5

Initial DDM parameters for GPS L5 acquisitions are in Table 3.3. Magnitudes for Doppler offset are below 20kHz, delay values range from 80 chips to 811 chips, and the peak-to-mean ratio ranges from 16dB to 21dB. Though this range of peak-to-mean ratio values is slightly larger than the range for L1 PRNs, these are considered successful acquisitions.

Band 2 L5 Acquisitions			
PRN	Doppler(kHz)	Delay(Chips)	Peak to Mean Ratio(dB)
4	8.9	681.6	20.5
5	13.5	586.2	18.9
12	-19.0	261.7	16.7
18	14.8	548.7	18.2
20	19.0	411.4	16.9
21	0.0	535.0	22.3
24	16.5	810.3	18.3
25	9.0	81.8	20.7
26	-11.5	471.3	16.3
29	3.4	740.8	19.2
31	-10.9	424.9	16.7

Table 3.3: Initial DDM Parameters for Band 2 L5 Acquisitions

These acquisitions are visually confirmed in Figure 3.15 as well. Note that the delay axis is expanded to accentuate the shape of the peak, which has a finer delay resolution due to the signal's higher chipping frequency.

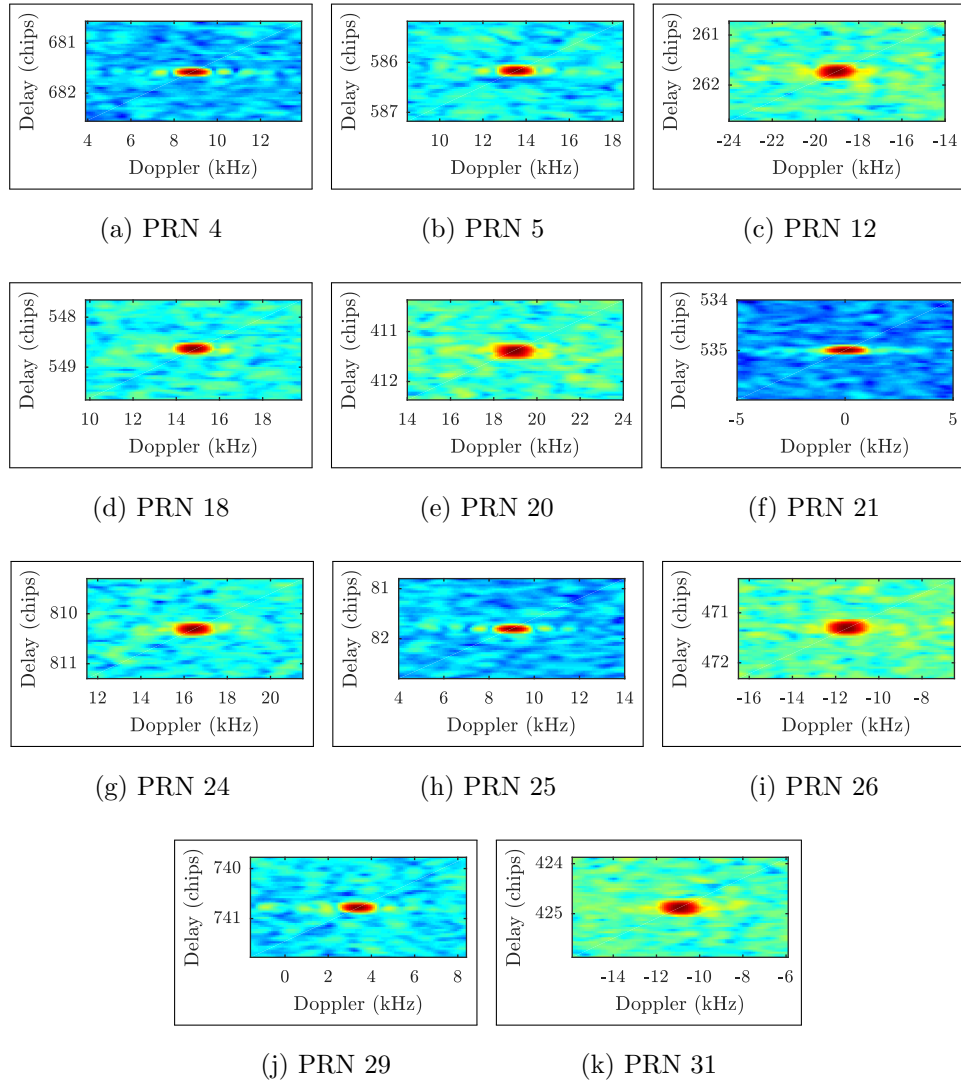


Figure 3.15: Visually Confirmed Initial L5 PRN Acquisitions

3.2.4 Galileo E5a

Initial DDM parameters for E5a acquisition are shown in Table 3.4. Doppler shift magnitudes lie in the range -12kHz to 30kHz. Delays range from 80 chips to 700 chips.

Band 2 E5a Acquisitions			
PRN	Doppler(kHz)	Delay(Chips)	Peak to Mean Ratio(dB)
4	29.2	125.8	6.4
5	-13.5	586.1	17.1
12	19.1	261.7	14.6
18	14.8	548.7	16.4
20	-4.2	662.6	5.7
21	0.0	535.0	6.9
24	-28.3	449.3	5.7
25	-9.0	81.8	18.4
26	-11.4	471.3	14.2
29	17.4	942.6	5.9
31	10.9	424.9	14.5

Table 3.4: Initial DDM Parameters for Band 2 E5a Acquisitions

The peak-to-mean ratio varies, but lies in distinct ranges due to several failed acquisitions. Successful acquisitions have a peak-to-mean ratio greater than 14dB. These are verified visually in Figure 3.16.

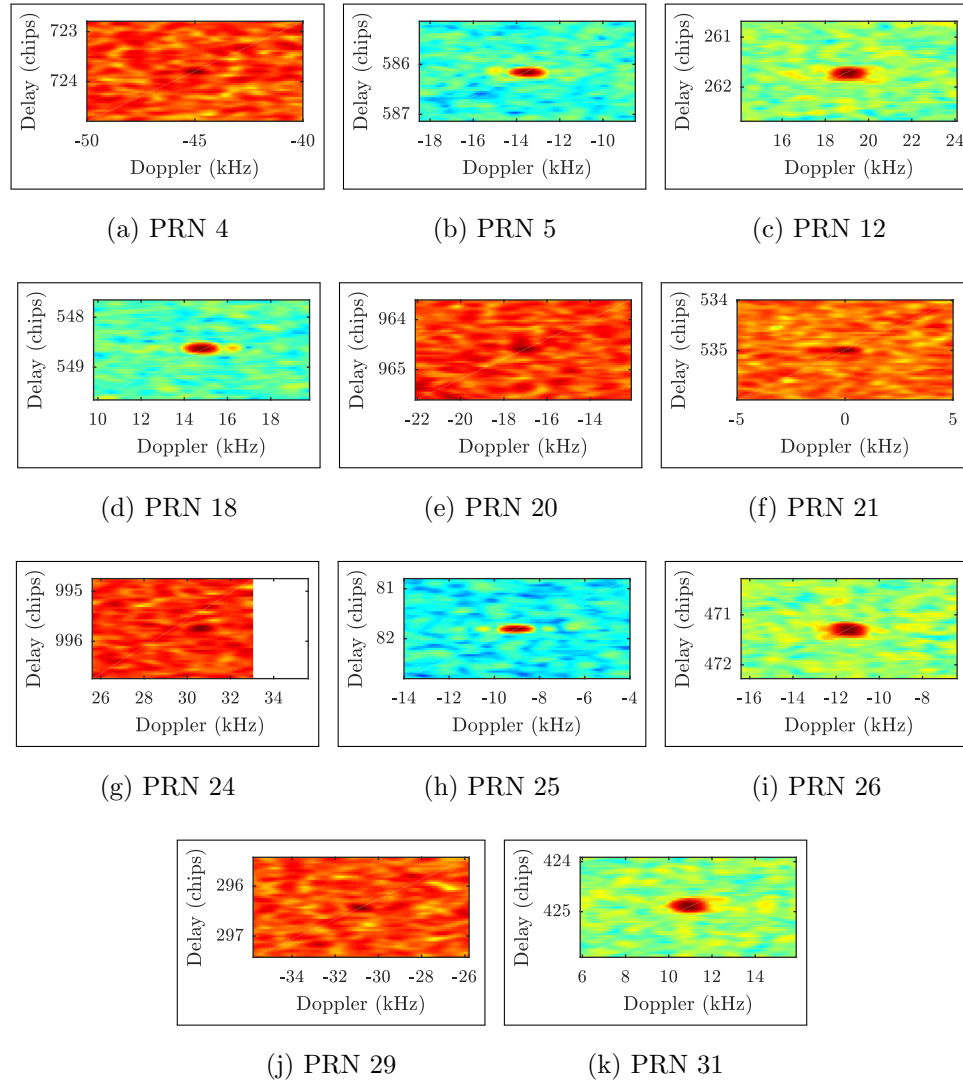


Figure 3.16: Visually Confirmed Initial E5a PRN Acquisitions

Figures 3.16(a), 3.16(e), 3.16(f), 3.16(g), and 3.16(j) show failed acquisitions which may be attributed to the same phenomena discussed in subsection 3.2.2. These peaks are not actually successful code acquisitions.

3.3 Signal Tracking

To analyze space-vehicle behavior as time progresses, the acquisition tool was modified to read in the sequence of waveform files and “track” a PRN for several seconds. This was performed on Band 1 samples, searching for GPS L1 PRN 4 over a period of 15 seconds. The results of this analysis are in Figure 3.17, containing separate stem plots for delay and Doppler as functions of time.

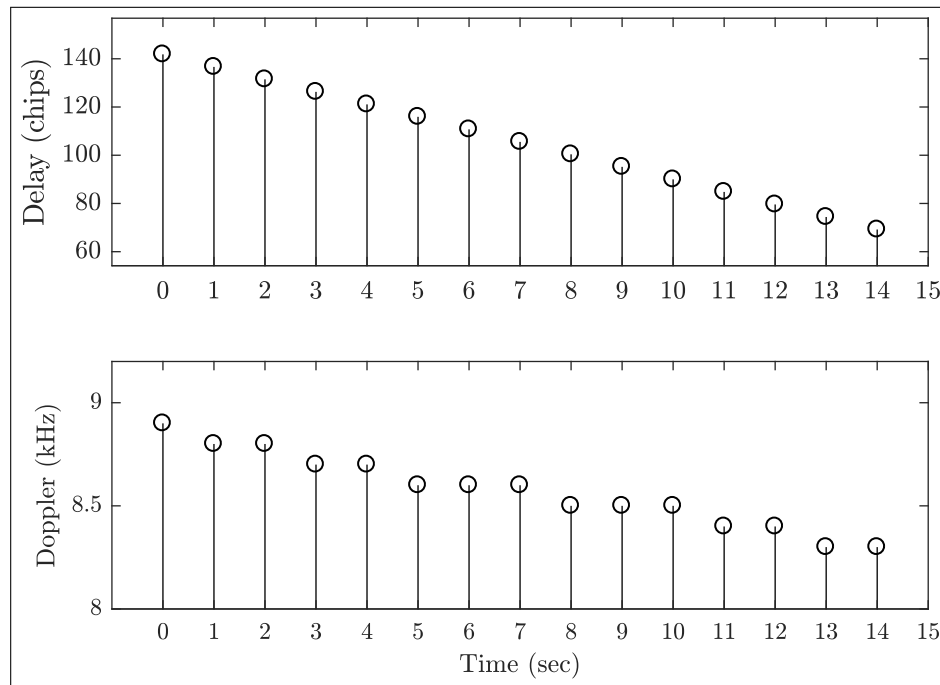


Figure 3.17: L1 PRN 4 Acquisitions for First 15 Seconds

This plot shows a linear trend in both delay and Doppler, indicating that the signal path distance is getting shorter at a constant rate and that the magnitude of the difference in velocities of the transmitter and receiver is decreasing at approximately a constant rate. Delay decreases from 140 chips to 80 chips at a rate of 4 chips/second. Doppler decreases from 8.9kHz to 8.3kHz at a rate of 40Hz/second.

3.4 RF Validation

One dual-band channel output from the test bench was connected to a spectrum analyzer to visualize the RF signal from the USRPs. Each band was transmitted from a single USRP and their outputs were combined in hardware. Signals in the following plots show time-averaged spectrums. Band 1 is depicted in Figure 3.18.

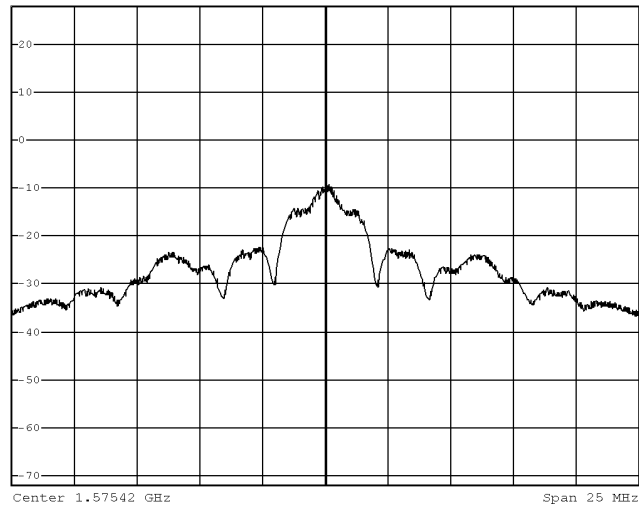


Figure 3.18: Band 1 Output at 1.57542 *GHz* with 25MHz Span

Band 1 output closely resembles Figure 3.6, verifying correct sample modulation by the USRP. Band 2, depicted in Figure 3.19, validates the spectrum in Figure 3.12.

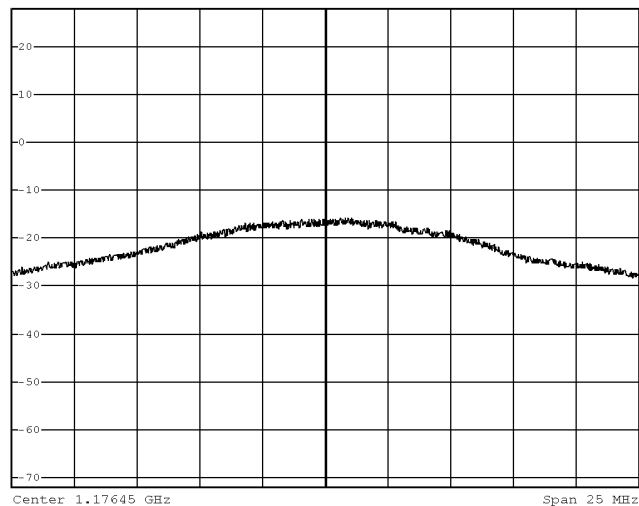


Figure 3.19: Band 2 Output at 1.17645 *GHz* with 25MHz Span

Finally, both bands are visualized on the spectrum analyzer by viewing a larger frequency span (600MHz) in Figure 3.20. The center frequency in this plot is approximately centered between the bands of interest at 1.3759GHz.

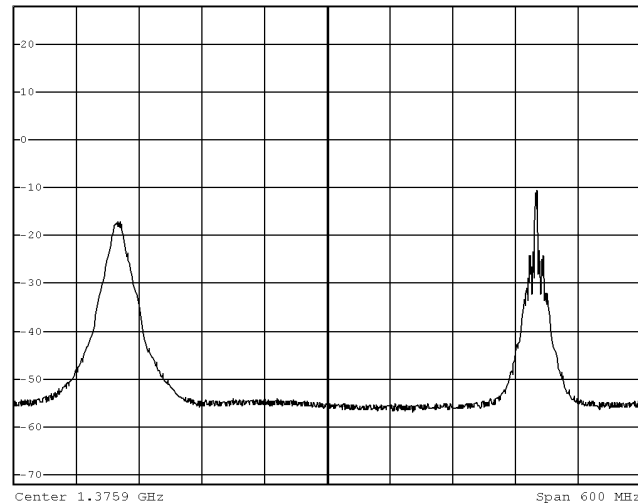


Figure 3.20: Dual-Band Output From Test Bench Transceiver

Figure 3.20 verifies output of the intended dual-band signal. It validates the RF signal at the interface to the receiver when it is complete and ready for testing, but does not indicate the presence of properly modulated PRN codes or navigation messages. We are limited to analyzing the digital contents of the signal in software on a computer.

3.5 Data Acquisition

GNSS signals recorded outdoors through the VeraPhase antenna are processed by a simple software implementation of the receiver. Figure 3.21 shows DDMs for three separate PRNs acquired.

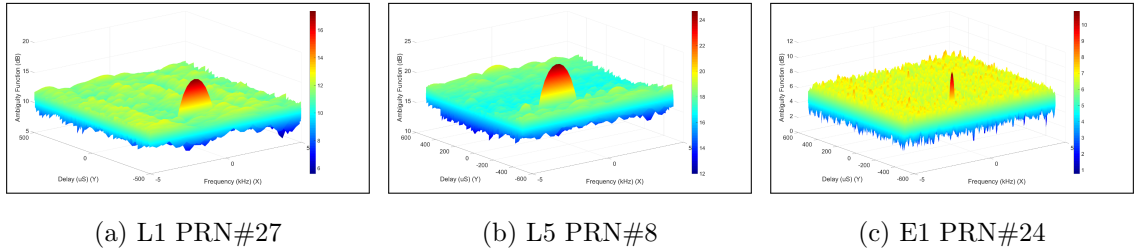


Figure 3.21: Signals Acquired from Live-Sky SDR GNSS Transceiver Recordings

These figures demonstrate that the device under test is capable of receiving both targeted GPS bands as well as Galileo E5a. Correlation peak shapes in these plots provide examples of ideal acquisition results from simulated files, so they are used for test-signal verification as well.

Chapter 4

Conclusion

4.1 Contributions

First, this research effort provides a valuable test system to researchers developing technologies for geophysical remote sensing. The system supplies the means to test spaceborne instruments before space deployment, reducing cost and risk of failure.

Second, it documents the utility and practicality of open-source software-defined radio systems for modulating arbitrary waveforms created by the user, specifically those devices created and supported by Ettus Research and National Instruments.

Third, this document expands the amount of knowledge accessible to GNSS researchers. It provides a high-level summary of relevant GNSS signal parameters and also a detailed explanation of signal construction.

4.2 Additional Applications

The test system was developed with the intent for future modification. GNSS-R research is continuing to grow in demand as researchers discover new practical uses for the data obtained by remote sensing instruments. GNSS systems in development, including QZSS and BeiDou, may become practical for reflectometry; our software may be modified to simulate these signals for supporting development of future in-

struments built to receive these signals.

The system can also transmit and receive simultaneous multi-channel RF communications in most of the VHF and UHF bands. It may be used to modulate arbitrary waveforms in these bands for testing a different receiver.

4.3 Future Work

Tests will be performed on the receiver when it is completed in Summer, 2018. At that time the plan is to transmit a 24-hour sequence of the dual-band signal discussed in this document, containing GPS L1 and L5 and Galileo E1bc and E5a, to one of the antenna ports of the receiver. Several tasks remain unsatisfied for accomplishing this goal.

Successful acquisition of all 44 PRNs must be verified. As of March, 2018, not all of the Galileo PRNs were found by the acquisition scripts. It is likely that errors exist in the acquisition script, because errors in the waveform generator would likely prevent acquisition of a single PRN.

Selection of appropriate transmit power is required. The receiver relies mainly on measurements of received power, so the selection of an appropriate gain from the USRPs is important for accurate simulation of reflections.

4.4 Summary

In summary, a test system has been developed which will be prepared for use on a GNSS reflectometry receiver currently in development. This thesis has documented the design procedure followed for implementing the hardware and software components of the test bench.

A review and description of GNSS signal parameters introduced and motivated the methods of waveform generation implemented in software. We discussed GPS

and Galileo systems and the components of the signals of interest, including L1, L5, E1bc, and E5a. These signals exhibit a variety of properties which we found useful in expanding the capabilities of GNSS reflectometry receivers.

In the design process, we discovered powerful and practical open-source hardware and software components to use. These include Linux operating system tools and the USRP software-defined radio with hardware drivers supplied by Ettus Research. Linux provides both a user interface and software development platform that includes editors and compilers/interpreters for Python, Matlab, C and C++. The hardware drivers are open-source, so a developer has the freedom to modify the code.

It has been demonstrated that this test bench accurately simulates the space environment for the receiver, including most of the intended signal reflections. Signal correctness was validated through analysis of time-domain waveform samples, spectrum analysis obtained by the FFT in Matlab, signal acquisition performed in Matlab to identify each satellite's spread-spectrum PRN code, and spectrum analysis of modulated output with RF test equipment in the lab.

Potential future GNSS reflectometry applications motivate the portability of the test-system design. It is easily modified for new purposes, including the addition of additional GNSS channels, improved performance, or even for alternative waveform simulations at different carrier frequencies in the VHF and UHF bands.

Appendices

A Significant GNSS Parameters

GNSS System Parameters						
GNSS System	GPS			Galileo		
Freq. Band	L1	L5		E1	E5	
Altitude (km)	20180			23222		
Period (hr)	14.08			11.97		
Elevation (°)	5			10		
Service Name	C/A	L5I	L5Q	E10S	E5a	
Carrier (MHz)	1575.42	1176.45		1575.42	1176.45	
Modulation	BPSK			CBOC	AltBOC	
Sub-Carrier Freq.	-	-	-	1.023 & 6.138	15.345	
Code Freq. (MHz)	1.023	10.23		1.023	10.23	
Sig. Component	Data		Pilot	Data	Pilot	Data
Prim. PRN Length	1023	10230		4092	10230	
Sec. PRN Length	-	10	20	-	25	20
Code Family	Gold	M	M	Random	M	
Data Rate (bps)	50		-	250	-	50

Table A1: GNSS Signal Parameters

References

- [1] U.S. Air Force. *GPS Overview*. Accessed: 2018-3-20. 2017. URL: <https://www.gps.gov/systems/gps/>.
- [2] European Commission. *European Radio Navigation Plan*. Accessed: 2018-3-20. 2018. URL: <http://ec.europa.eu/DocsRoom/documents/28325>.
- [3] Jose Ángel Ávila Rodríguez. *Galileo Signal Plan*. Accessed: 2018-2-27. 2011. URL: http://www.navipedia.net/index.php/Main_Page.
- [4] Ivan G. Petrovski and Tochiaki Tsujii. *Digital Satellite Navigation and Geophysics*. 1st ed. 2012. ISBN: 978-0-521-76054-6.
- [5] S. Gleason et al. “Detection and Processing of Bistatically Reflected GPS Signals From Low Earth Orbit for the Purpose of Ocean Remote Sensing”. In: *IEEE Transactions on Geoscience and Remote Sensing* 43 (2005), pp. 1229–1241.
- [6] Maria Paola Clarizia. “Investigating the Effect of Ocean Waves on GNSS-R Microwave Remote Sensing Measurements”. PhD thesis. University of Southampton, Oct. 2012.
- [7] Bob Allen. *CYGNSS Overview*. Accessed: 2017-08-24. May 2015. URL: www.nasa.gov/cygnss/overview/.
- [8] Razal Rose et al. “The CYGNSS flight segment; Mainstream science on a micro-budget”. In: 2015 (June 2015).
- [9] Ettus Research. *OctoClock CDA-2990*. Accessed: 2018-3-25. 2018. URL: <https://www.ettus.com/product/details/OctoClock-G>.
- [10] Tallysman Inc. *VeraPhase 6000 Family*. Accessed: 2018-3-25. 2018. URL: <http://www.tallysman.com/index.php/gnss/products/antennas-all-gnss-signals/vp6000/>.
- [11] Ettus Research. *USRPN210*. Accessed: 2018-2-23. 2018. URL: <https://www.ettus.com/product/details/UN210-KIT>.
- [12] Tamás Szelei. *License Terms for Humans*. Accessed: 2018-4-1. 2017. URL: https://files.ettus.com/manual/md_lib_deps_rpclib_LICENSE.html.
- [13] Ettus Research. *DBSRX2*. Accessed: 2018-3-20. 2018. URL: <https://www.ettus.com/product/details/DBSRX2>.

-
- [14] Ettus Research. *WBX*. Accessed: 2018-3-20. 2018. URL: <https://www.ettus.com/product/details/WBX>.
 - [15] Pratap Misra and Per Enge. *Global Positioning System Signals, Measurements, and Performance*. 2nd ed. 2006. ISBN: 0-9709544-1-7.