Demonstration of GNU Radio Compatibility with a NASA Space Communications Network Modem (GRCON 2019)

David T. Miller
Dave.Todd.Miller@gmail.com
NASA Contractor, Ashburn, VA 20147 USA

Abstract

This paper presents the results of testing that the author recently conducted to demonstrate that a GNU Radio Software Defined Radio (SDR) receiver is compatible with a typical National Aeronautics and Space Administration (NASA) satellite ground station vendor modem and NASA waveforms within the scope of this initial testing phase. The author implemented the GNU Radio SDR receiver for this testing by installing the open source GNU Radio software on a Dell laptop and using a Commercial Off-The-Shelf (COTS) RTL-SDR hardware dongle. During the demonstration testing, the NASA vendor modem transmitted a repeating bit pattern to the GNU Radio SDR receiver using Binary Phase Shift Keying (BPSK) modulation for one test case and Quadrature Phase Shift Keying (QPSK) modulation for a second test case. The GNU Radio SDR received and demodulated the signal in order to recover the transmitted bit stream. This paper provides the specific GNU Radio SDR receiver design developed and implemented for this testing, the detailed demonstration test configurations, and the demonstration test results. This paper also concludes by listing many possible functions that the GNU Radio SDR could provide in a NASA satellite ground station.

1. Introduction

The National Aeronautics and Space Administration (NASA) Space Network (SN) provides communications services at S-band, Ku-band, and Ka-band to a variety of Low Earth Orbit (LEO) science spacecraft via many Geosynchronous Earth Orbit (GEO) Tracking and Data Relay Satellite (TDRS) spacecraft and the ground station assets located in New Mexico and other locations worldwide. Similarly, NASA's Near Earth Network (NEN) provides communications services at S-band, X-band, and Ka-band to LEO science spacecraft via direct links to ground stations located worldwide. Both the SN and NEN use S-band to provide low data rate (LDR) communications services typically at data rates of ≤3.0 Mbps. The SN and NEN currently use X-band, Ku-band, and Ka-band to provide high data rate (HDR) communications services typically at data rates ≤300 Mbps.

NASA uses several different vendor modems in their ground stations with an Intermediate Frequency (IF) set at either 70 MHz, 370 MHz, or 1.2 GHz.

By demonstrating via actual testing that the GNU Radio Software Defined Radio (SDR) is compatible with a typical NASA modem and typical NASA waveforms, NASA could then consider employing the GNU Radio SDR in ground stations to provide many development, operational, and testing functions that this paper discusses in detail at the end of this paper in the conclusions section.

2. Demonstration Test Objective

The objective of this demonstration test activity was to determine whether a GNU Radio SDR receiver can be compatible with a typical NASA vendor modem and NASA waveforms within the scope of this initial testing phase.

3. Scope of Demonstration Test

As Figure 1 depicts, this demonstration activity used a back-to-back test configuration with a 50 ohm coaxial cable between the GNU Radio SDR receiver and the NASA vendor modem modulator. The author conducted the demonstration tests with a 1.2 GHz IF.

Proceedings of the 9th GNU Radio Conference, Copyright 2019 by the author.

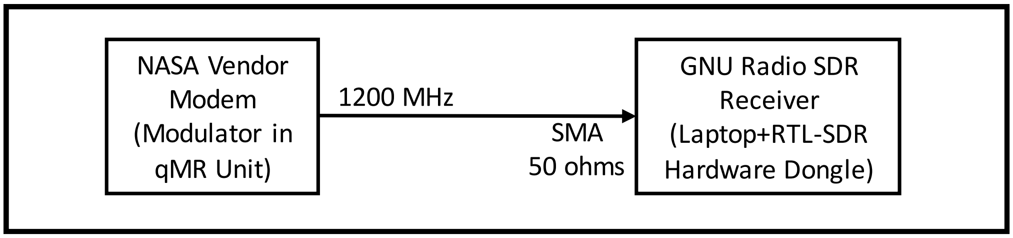

Figure 1: NASA Modem-To-GNU Radio Demonstration Back-To-Back Test Configuration

NASA often uses the Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK) modulation waveforms for communications services. Therefore, the author conducted BPSK and QPSK test cases.

The NASA vendor modem that served as the transmitter for this demonstration as shown in Figure 1 was a Quantum Mission Receiver (qMR) unit manufactured by the company RT Logic. The qMR unit has both receiver and transmitter functions. NASA installed the qMR unit in one of their ground stations.

4. GNU Radio SDR Receiver Design

This section describes the details of the GNU Radio SDR design and implementation.

The author implemented the inexpensive GNU Radio SDR receiver with a Dell laptop, the free open source Linux/Ubuntu operating system, the free open source GNU Radio software (version 3.7.11), and an inexpensive $28.00 Commercial Off-The-Shelf (COTS) RTL-SDR hardware dongle (RTL2832U). The RTL-SDR dongle has a Universal Serial Bus (USB) interface on one side for connection to the Dell laptop. On its other side, the RTL-SDR dongle has a 50 ohm SubMiniature version A (SMA) interface for connection to the IF signal.

Please refer to (section 1.6 of Stewart, R., 2017) for a detailed description of the RTL-SDR hardware dongle functions and design.

Figure 2 depicts the GNU Radio Companion Flowgraph Graphical User Interface (GUI) for a BPSK test demonstration. Figure 3 depicts the GNU Radio Companion Flowgraph GUI for a QPSK test demonstration. The author implemented the GNU Radio SDR receiver with GNU Radio blocks that were already available in the GNU Radio Block library.

The RTL-SDR Source block shown in Figure 2 and Figure 3 controls the RTL-SDR dongle according to the configurable settings of the block. For example, the RTL-SDR dongle tunes to the IF frequency that the operator sets using the RTL-SDR Source block in Figure 2 and Figure 3. The RTL-SDR dongle also downconverts the IF signal to a Digital IF (complex I/Q baseband) and then sends the digital IF complex sample stream to the Dell laptop via the USB interface. The Costas Loop block provides carrier synchronization and the Polyphase Clock Sync block provides symbol synchronization. According to the gnuradio.org website, the Polyphase Clock Sync block implementation is based on the algorithm described in (Harris F., 2001).

The Time Sink blocks and the Frequency Sink block provide oscilloscope and Spectrum Analyzer functions, respectively, so that an operator can monitor the receiver performance in real-time. The I/Q Constellation Sink blocks provide real-time displays to verify carrier synchronization. Each File Sink block passes a recovered bit stream to a file for storage and post-test analysis. For the QPSK flowgraph in Figure 3, complex multiply blocks provide a convenient I/Q constellation rotation function to resolve carrier lock phase ambiguity (one of the saved files will have the correct transmitted bit pattern). Therefore, one can verify that the GNU Radio receiver can recover the correct bit pattern by just reviewing the saved files during the post-test analysis.

5. Demonstration Test Approach

The author conducted the following specific demonstration test cases with the test configuration of Figure 1:

- Test Case #1: BPSK at 1.024 Mbps
- Test Case #2: QPSK at 2.048 Mbps

Table 1 lists the driving GNU Radio block parameter settings for the BPSK and QPSK test cases.

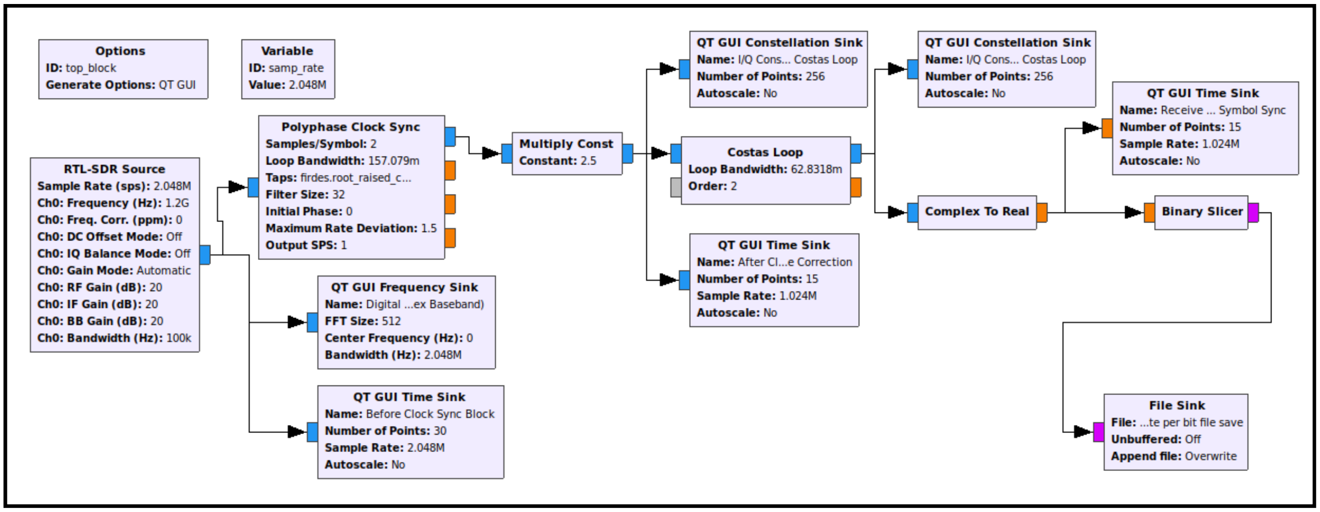

Figure 2: GNU Radio Companion GUI Flowgraph for BPSK Test Case #1

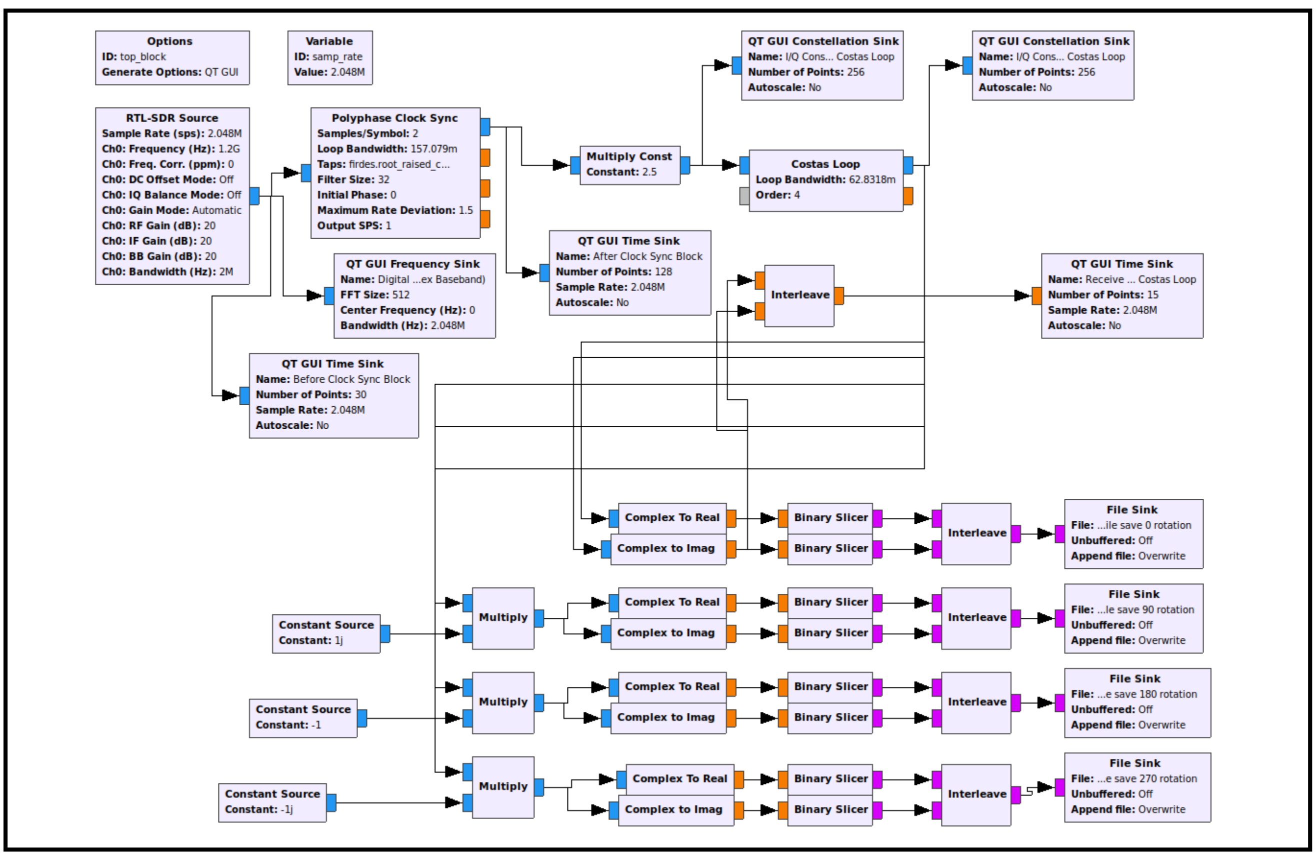

Figure 3: GNU Radio Companion GUI Flowgraph for QPSK Test Case #2

Table 1: GNU Radio Companion GUI Parameter Settings for Test Cases

GNU Radio Companion Block	BPSK (1.024 Mbps) Test Case #1	QPSK (2.048 Mbps) Test Case #2
RTL-SDR Dongle Source Block:		
Center Frequency	1.2 GHz	1.2 GHz
Sample Rate	2.048 Msps	2.048 Msps
Costas Loop Block:		
Order	2	4
Symbol Synchronizer Block:		
Input samples per symbol	2	2
Output samples per symbol	1	1
RRC Roll-Off Factor	0.5	0.5

The author configured both the GNU Radio and NASA vendor modem to operate at a 1.2 GHz IF.

The NASA Modem transmitted a short repeating bit pattern of 15 bits in length, specifically, a Pseudo Random Bit Stream (PRBS) 4 pattern. For both the BPSK and QPSK test cases (1.024 Mbps and 2.048 Mbps), the RTL-SDR source block set the RTL-SDR hardware dongle for only two samples per symbol (2.048 Msps) to achieve the demonstrated data rates.

The Polyphase Clock Synchronizer block also supported Root Raised Cosine (RRC) filtering. For these demonstrations, the author set both the GNU Radio receiver and NASA modem modulator for a RRC roll-off factor of 0.5.

The Binary Slicer block translated each bit into one byte (8 bits) for convenient file storage and post-test display with a COTS hexadecimal reader application.

6. Demonstration Test Results

The following performance occurred during both test cases (BPSK and QPSK):

- The GNU Radio SDR successfully recovered the transmitted bit stream with the correct repeating bit pattern (PRBS4).
- The GNU Radio displayed the expected frequency spectrums for BPSK and QPSK at the tested data rates.
- The GNU Radio SDR successfully achieved Symbol Synchronizer Lock.
- The GNU Radio SDR successfully achieved Carrier Synchronization Lock.

The author conducted this initial GNU Radio SDR demonstration test phase without adding noise, therefore, as one would expect, perfect Bit Error Rate (BER) performance occurred. The author verified perfect BER performance by analyzing the collected bit stream data in the saved files as a post-test activity. Also, the author resolved QPSK carrier lock phase ambiguity by examining the file data.

The author used the spectrum analyzer and oscilloscope blocks to monitor the GNU radio performance for any anomalies in real-time. The author did not observe any anomalies during the testing.

For example, Figure 4 depicts the GNU Radio spectrum analyzer (Frequency Sink) block display. Specifically, Figure 4 depicts the expected BPSK spectrum for a 1.024 Mbps data rate with a repeating 15 bit pattern.

Figure 5 depicts the GNU Radio Time Sink block oscilloscope located after the Costas Loop that depicts the recovered repeating PRBS4 bit pattern that occurred in real-time during BPSK Test Case #1. The bits in Figure 5 are actually a flipped version of the PRBS4 pattern because of the phase onto which the Costas Loop locked during the test case. The author also used the Time Sink block display in Figure 5 to verify in real-time that symbol synchronizer slips were not occurring. For example, the author observed the repeating bit pattern over a period of time in order to verify in real-time that bit flips were not occurring during

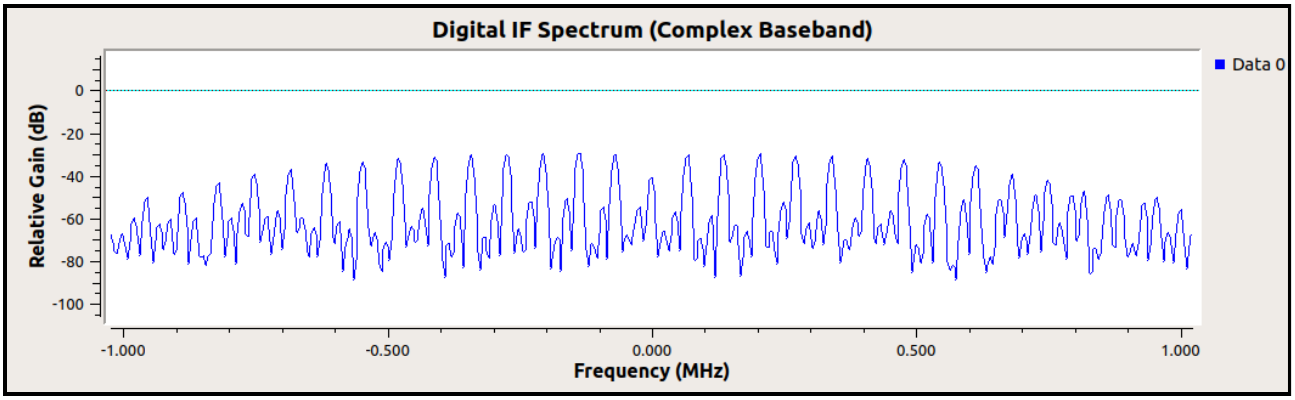

Figure 4: GNU Radio Companion Spectrum Analyzer Display for BPSK (Test Case #1)

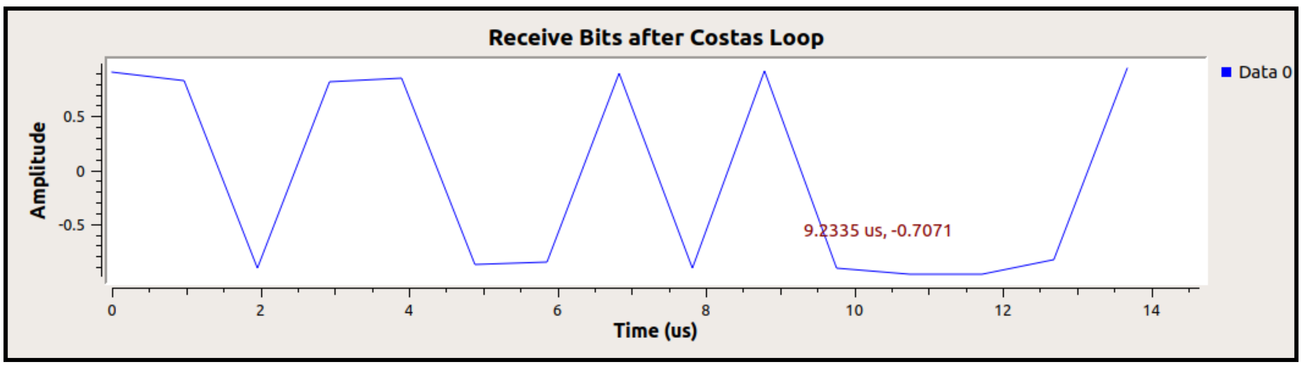

Figure 5: GNU Radio Companion Recovered Bit Pattern Display for BPSK (Test Case #1)

the testing.

Figure 6 depicts the display of the I/Q Constellation Sink block located before the Costas Loop block for Test Case #1 (BPSK). Specifically, Figure 6 depicts the rotating I/Q Constellation that occurred during test case #1 (BPSK) before carrier synchronization. For example, Figure 6 depicts the rotating I/Q Constellation that occurred during test case #1 before the Costas Loop block removed the residual frequency offset between the Vendor Modem and the RTL-SDR dongle.

Figure 7 depicts the display of the I/Q Constellation Sink block located after the Costas Loop for Test Case #1 (BPSK). Specifically, Figure 7 depicts the I/Q Constellation without a rotating I/Q Constellation that occurred during test case #1 after the Costas Loop block removed the residual frequency offset between the Vendor Modem and the RTL-SDR hardware dongle. Therefore, Figure 7 depicts the I/Q Constellation after successful carrier synchronization lock.

For example, equation (1) depicts the complex baseband signal with a residual frequency rotation that the RTL-SDR hardware dongle provided to the GNU Radio.

$$z_{IN}(t) = e^{j\theta t} \bullet e^{j[(2\pi f_\Delta)t + \theta_\Delta]} \qquad (1)$$

Where;

For BPSK, θ = 0 degrees or 180 degrees.
For QPSK, θ = 45 degrees, 135 degrees, 225 degrees, or 315 degrees.

$z_{IN}(t)$ = Complex signal from the RTL-SDR hardware dongle.

f_Δ = The residual frequency difference between the IF signal from the NASA vendor modem and the RTL-SDR hardware center frequency even when both were set to a 1.2 GHz center frequency.

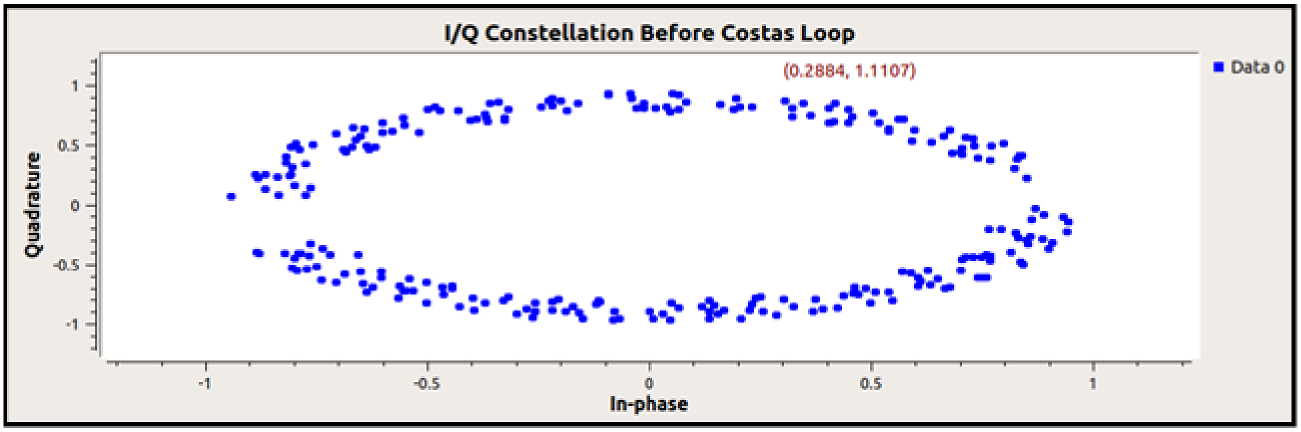

Figure 6: GNU Radio I/Q Constellation Before the Costas Loop for BPSK Test Case #1

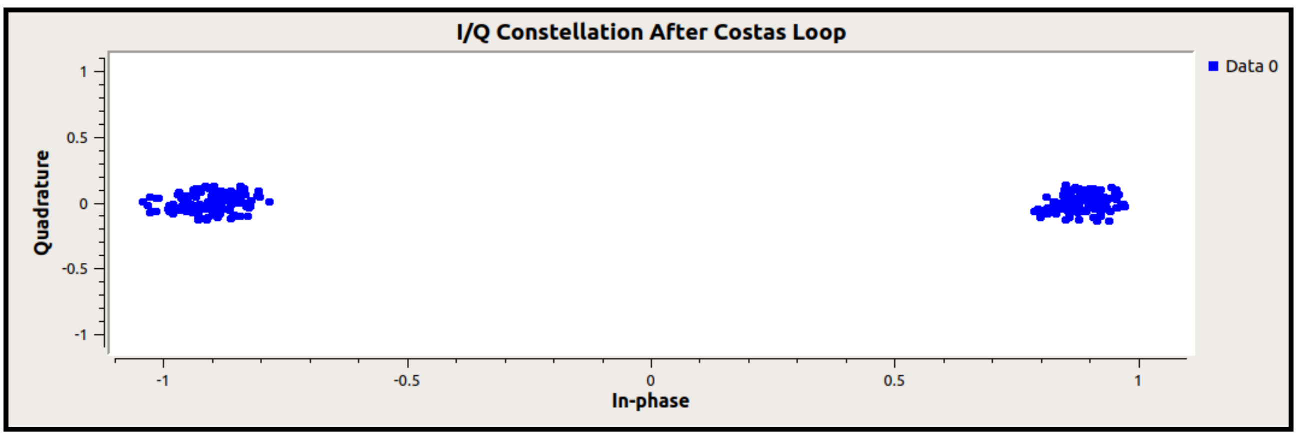

Figure 7: GNU Radio I/Q Constellation After the Costas Loop for BPSK Test Case #1

θ_Δ = The fixed residual phase difference at t=0 between the IF signal from the NASA vendor modem and the RTL-SDR hardware center frequency even when both were set to a 1.2 GHz center frequency.

t = time.

Equation (2) depicts the complex baseband signal with the residual frequency and phase now removed (Costas Loop output with carrier synchronization lock).

$$z_{OUT}(t) = e^{j\theta t} \qquad (2)$$

Where;

$z_{OUT}(t)$ = Complex signal at output of the Costas Loop after carrier synchronization lock.

The test results demonstrated that within the scope of this initial testing phase, the implemented GNU Radio SDR is compatible with the NASA vendor modem and BPSK and QPSK waveforms that NASA networks typically use. However, note that the author did not test all NASA Network waveforms during this initial GNU Radio testing phase. For example, the NASA Phase Modulation (PM) with a Subcarrier waveform and also NASA coding would be a part of forward GNU Radio SDR work that this paper discusses in more detail in section 7.

7. Forward Work

In the near term, follow-on test activities should include conducting end-to-end link demonstrations through a NASA relay satellite with a GNU Radio SDR receiver rather than just the back-to-back modulator/receiver tests discussed in this paper.

Also, additional follow-on activities should include upgrading the GNU Radio as follows:

- New Waveform Blocks: Add another NASA Waveform Demodulator to this GNU Radio SDR: PM with a Subcarrier.
- Doppler Block: Add a coarse frequency offset tuning function so that the receiver can operate with a relatively large residual frequency offset between the incoming IF signal and the RTL-SDR hardware dongle selected center frequency. Large initial frequency offsets because of Doppler are common when providing communications services to LEO spacecraft.
- Add modulator functions to the GNU Radio SDR by procuring a hardware dongle that both transmits and receives.
- NRZ-M formatting.
- Add typical NASA framing and coding blocks.

Other areas of testing and GNU Radio receiver upgrades could include:

- For NASA space communications services waveforms, explore the maximum data rate capability of a GNU Radio SDR modem when using only a laptop Central Processor Unit (CPU).
- Consider developing blocks for higher order constellations like Digital Video Broadcasting Second Generation (DVB-S2) 16-ary Amplitude and Phase Shift Keying (16APSK). NASA is considering DVB-S2 16APSK in order to achieve higher data rates in the crowded S-Band frequency spectrum.

8. Conclusions

Within the scope of this initial demonstration testing phase, the GNU Radio is compatible with some NASA modems and NASA waveforms like BPSK and QPSK at NASA S-band relatively lower data rates. However, additional testing phases and expanded functionality will be needed in order to fully characterize the GNU Radio SDR performance with all NASA waveforms and modems that NASA employs for S-band communications services.

Also, based on this GNU Radio SDR design, development, and testing activity, NASA ground station operations and engineering personnel could employ a GNU Radio SDR as follows:

- The GNU Radio SDR could immediately be used as an inexpensive, compact, light, and mobile test tool with receiver, spectrum analyzer, and oscilloscope capabilities for conducting troubleshooting activities in the NASA ground stations.
- The GNU Radio SDR could immediately be used as an inexpensive modem for NASA engineering and operator team digital communications and communications theory training and education, including end-to-end link simulations.
- The GNU Radio SDR could immediately be used as an inexpensive and easy to reconfigure modem in a test bed, (including end-to-end testing) to explore new future network waveforms in a cost-effective manner.
- The GNU Radio SDR could immediately be used as an operational inexpensive digital IF recorder and baseband data recorder.
- After additional functions like PM demodulation, modulator, and coding are added, the GNU Radio SDR could be used as a very inexpensive and easy to configure spare modem for some operations to augment more expensive standard vendor modems.

References

Harris, F and Rice, M. Multirate Digital Filters for Symbol Timing Synchronization in Software Defined Radios. IEEE Selected Areas in Communications, Vol. 19, No. 12, December, 2001.

Stewart, R, Barlee, K, Atkinson, D, Crockett, L. Software Defined Radio Using MATLAB and Simulink and the RTL-SDR. University of Strathclyde Academic Media, 1st edition (revised), 2017.