

Disclaimer: This dissertation has been written by a student and is not an example of our professional work, which you can see examples of [here](#).

[View full disclaimer](#)[Home](#) / [Examples](#)

Share this:

ABSTRACT

The modern telecommunication age has seen the development and use of a number of schemes for sending, receiving and processing signals, namely Amplitude modulation (AM), Frequency Modulation (FM), Phase Modulation (PM), GSM, OFDM among many others.

Initially telecommunication techniques were performed using analogue devices only. But as time progressed the merits of digital communication became evident and more digital techniques were developed. Thus, now more functions of modern radio system and telecommunications is implemented in software rather than hardware. This led to the development of Software Defined Radio architecture. The SDR has the advantage of being multi-typed, being able to perform high speed wireless communication using inexpensive hardware, very efficiently.

GNU Radio is based on the Software-Defined Radio (SDR) and uses the USRP as the hardware to do the signal transmission and reception.

In this senior design project we have first learned about the SDR platform and GNU Radio. We then got our test platform ready by installing Linux Mint OS and GNU Radio. We then simulated, transmitted, received and viewed FM, GMSK, QPSK, 8PSK, 16PSK, OFDM and PAPR Reduction of OFDM. We have also conducted error analysis of QPSK. All the experiments were performed using the SDR test-bed of USRP via the GNU Radio Software platform.

Our work progress is summarized as follows:

We have at first built a FM Receiver which receives FM signal from the airwave. We then built a FM Transmitter using which we were able to transmit a pre-recorded MP3 audio file which could be heard on any FM radio or cellphone. We then constructed a flow-graph for QPSK. Using this flowgraph we had calculated the BER by adding a uniform noise to the modulated QPSK. We have also simulated 8PSK and 16PSK and have viewed their constellation. Then we had sent and received a JPEG image file using GMSK and viewed its FFT and histogram plot. As a further example of GMSK application we had sent and received an audio file. Later we had modulated a signal using OFDM using QAM64 scheme and viewed its respective FFT, histogram and constellation plots. And finally we performed PAPR reduction of an OFDM signal using convolutional coding.

All the experiments mentioned above has been successfully run.

INDEX OF ACRONYMS

ADC Analog to Digital Converter

AGC Automatic Gain Control

AWGN Additive White Gaussian Noise

BW Band Width

BER Bit Error Rate

DAC Digital to Analog Converter

DSP Digital Signal Processing

DVB-T Digital Video Broadcasting-Terrestrial

DBPSK Differential Binary Phase Shift Keying

FIR Finite Impulse Response

FFT Fast Fourier Transform

FPGA Field-Programmable Gate Array

GPS Global Positioning System

GRC GNU Radio Companion

IFFT Inverse Fast Fourier Transform

ICI Inter-Carrier Interference

ISI Inter-Symbol Interference

LFSR Linear Frequency Shift Register

LTE Long Term Evolution

MC Multi-Carrier

MIMO Multiple-Input and Multiple-Output

OFDM Orthogonal Frequency Division Multiplexing

OS Operating System

PAPR Peak-to-Average Power Ratio

PSK Phase Shift Keying

PN Pseudorandom Noise

QAM Quadrature Amplitude Modulation

SDR Software Defined Radio

SNR Signal to Noise Ratio

USRP Universal Software Radio Peripheral

VCO Voltage-controlled Oscillator

CHAPTER 1

INTRODUCTION

1.1 Background

Radio communication in the modern telecommunication system plays an important role. Radio is widely used in commercial, medical, meteorological, military and civilian fields. Communication systems constantly transmit from analog systems to digital systems, a number of digital RF receiver appeared in this trend. Despite these receivers can cover multiple bands, their range is limited and their function is relatively small with lack of flexibility and scalability. They are not fully interoperable between different types of stations thus are unable to meet the modern communications requirements.

Conventional radio systems are hardware-based for specific purposes and are urgently needed to be replaced by multiband, multimode, programmable, versatile radio systems. The Software Defined Radio (SDR) concept was put forward to solve these problems.

SDR has been use since the early nineties although not so popularly. Its initial purpose was the creation of a radio capable of working at different frequencies. It can also tune to any frequency band, transmit and receive different modulation and different physical parameters across a large frequency spectrum by using a programmable hardware and powerful software. An alternative definition for SDR is a collection of hardware and software technologies that enable reconfigurable system architectures for wireless networks and user terminals.[1] The central idea is:

Construct an open, standardized, modular common hardware platform, and the various functions, such as working frequency, modulation and demodulation types, data formats, encryption mode communication protocol software to accomplish, and to A/D and D/A converter as close to the antenna, in order to develop a high degree of flexibility, openness and a new generation of wireless communication systems.

USRP (Universal Software Radio Peripheral) is used in conjunction with computers to design SDR platforms. It acts as digital baseband and Intermediate Frequency (IF) section in radio communication systems. In the software end we use the open source software GNU Radio. GNU Radio is a free software and is used along with USRP as software defined radio. GNU Radio mainly uses Python programming language to write. But its core signal processing module built in C++ in a microprocessor. Thus, developers can quickly and easily build real-time, high-capacity wireless communication systems.

1.2 Objectives

We have conducted a literature study on Software Defined Radio, the working principle of GNU Radio and the USRP hardware. We have carried out experiments to obtain results for analysis. The aim of this research is briefly listed below:

1. Broadly understanding the working principle of the SDR platform;
2. Installing and understanding the working principle of USRP;
3. Setting up GNU radio to implement different modulation schemes;
4. Creating flow-graphs and performing analysis of the following schemes:
 1. FM Receiver and Transmitter
 2. GMSK Audio and Image Receiver and Transmitter
 3. QPSK, 8PSK and 16PSK
 4. OFDM using QAM64
 5. PAPR Reduction of OFDM using Convolutional coding.

CHAPTER 2

LITERATURE REVIEW-I

2.1 Software Defined Radio (SDR)

Software Defined Radio was first proposed by Joseph Mitola in 1992 to create a device capable of emulating multiple radios operating at different frequencies. Instantly the research community saw the various other uses of the technology and the United States Navy was the first to use it between 1991 and 1995. But now it has a much broader use in the field of telecommunication and beyond.[2]

Software Defined radio (SDR) is a communication system where components that have been previously implemented by analogue devices such as mixers, filters, amplifiers, modulators, demodulators etc. are instead implemented digitally, using software.

A SDR processes digitally converted signals and data compared to analog signals in analogue radio. The data is digitized via an Analog to Digital Converter (ADC). The ideal receiver scheme contains an ADC to an antenna. A Digital Signal Processor (DSP) would read the converter, and then its software would transform the stream of data from the converter to any other form the application requires.

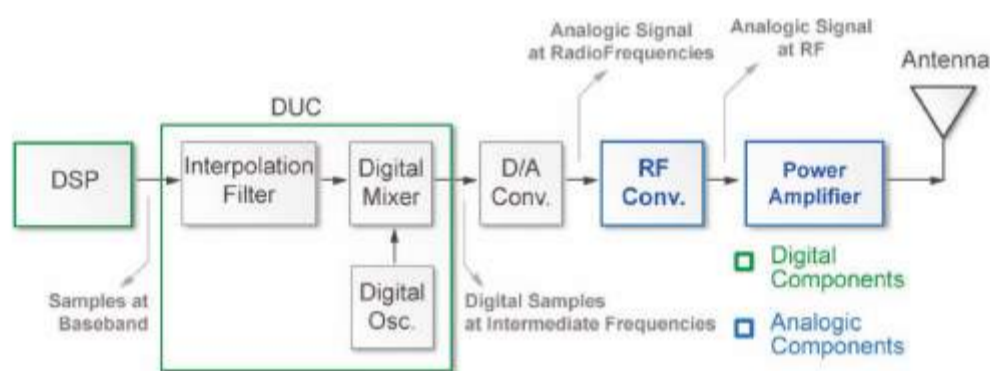


Fig 1: SDR Transmitter

To transmit signals through air it must be analog. The ideal transmitter would be similar to the receiver. A DSP generates a stream of numbers which would be sent to a Digital to Analog(DAC) converter connected to a radio antenna.

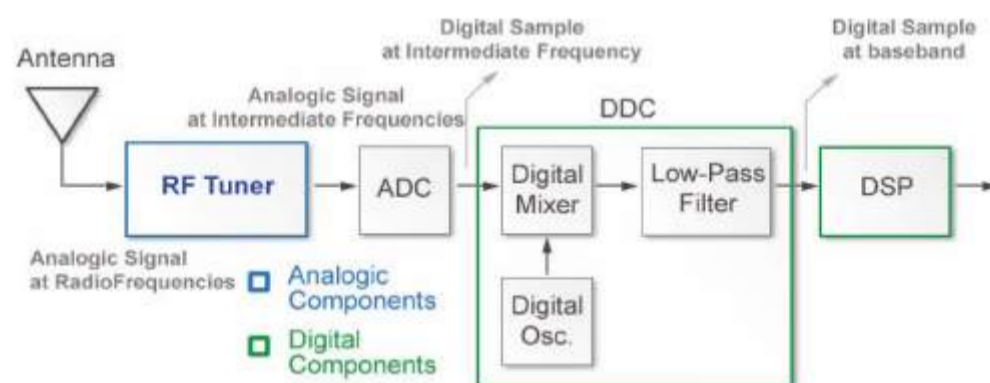


Fig 2: SDR Receiver

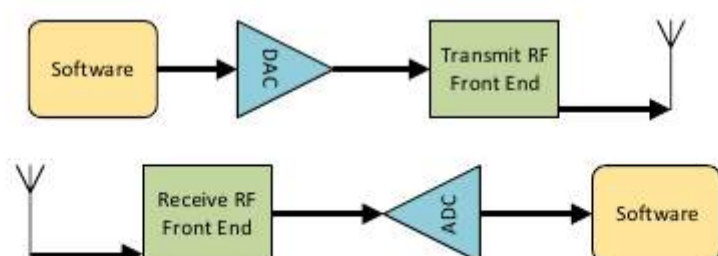


Fig 3: General Block diagram of SDR Transmitter and Receiver

There are three types of SDR:

1. Devices which support various different format e.g. GSM-CDMA dual mode cellphones. This is used so that devices do not become phased out when newer telecommunication schemes are introduced.
2. Devices based on Field-programmable gate-array (FPGA) and DSP. Usually these devices are vendor specific using specific hardware and software making them less scalable and limiting portability.
3. SDR in this category is used common hardware like ordinary PC and Open Source software. This category has the greatest advantages as common high-level languages (C or C++) can be used instead of proprietary software. Also in many case the device will be cheap and can be used for other purposes, making it more cost effective.

2.2 GNU Radio

There are around 30 different SDR software, of which, research oriented software are only four. They are:

- GNU Radio
- MATLAB RTL-SDR
- SDR Lab (NI Labview)
- Redhawk

Many of the SDR software are device-specific and are useless when used in conjunction with other unspecified devices. Of the aforementioned programs only GNU Radio and Redhawk is free of cost. Both MATLAB and SDR Lab are proprietary.

GNU Radio was launched in 2001 by philanthropist John Gillmore and Eric Blossom with a funding of \$320,000 from the former, and is distributed using the GNU General Public License.[3] It is today one of the most advanced open source project. GNU Radio's target users are researchers, academics and hobbyists. GNU Radio provides examples, reference systems and applications for FM, AM, PM, GMSK, Global System for Mobile communications (GSM), OFDM, High-definition television (HDTV) etc [4].

GNU Radio can be primarily used in Linux operating systems, with small support for Mac OS and Windows. In GNU Radio, the radio system can be represented by a flow-graph where graph vertices are known as signal processing blocks and edges indicate a connection between the two blocks. Data flows in one direction from an input signal source to one or more output signal sinks. For succeeding blocks to support inputs from preceding blocks both the blocks have to have the same data type. GNU Radio has a few data types which are color coded:

- Complex
- Floating point
- Byte
- Integer
- Short

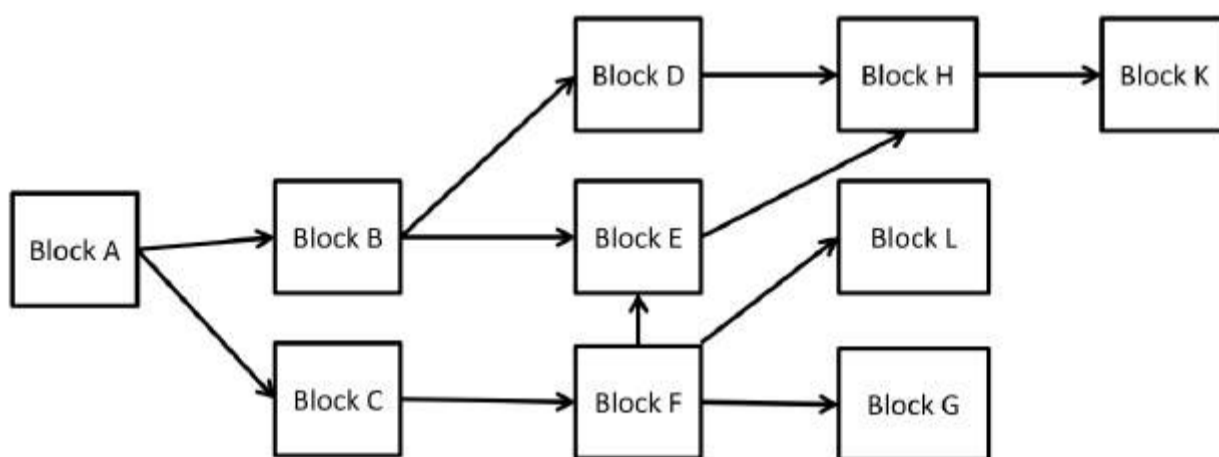


Fig 4: A typical GNU Radio block flow-graph

In GNU Radio, signal processing blocks are defined in C++, while the connections between the blocks for a given application are declared in Python. Use of a high-level language for development has the advantage of users simply being able to make connections between adjacent blocks, thus creating new applications, without having to write a single line of code. This gives agility of software development in a high level language without having to spend the previously required amount of time to write the entire application in C++. Thus debugging becomes easy and time is spent on making applications and not coding. Though there is room for coding whenever users need specialty applications which are not present in GNU Radio by default, thus providing flexibility.

2.3 GNU Radio Companion (GRC):

GRC is the graphical user interface (GUI) for the GNU Radio. GNU Radio can be used without GUI but then the user would have to do all the operations using Linux terminal commands which, to some people, would be a less intuitive way to designing applications. In GRC we can set up our application graphically using the aforementioned blocks. Each block parameters can be tweaked by going into properties. By changing the block parameters we are actually inputting value in the C++ source code which is finally executed. The GRC essentially acts as a wrapper for the GNU Radio tool.

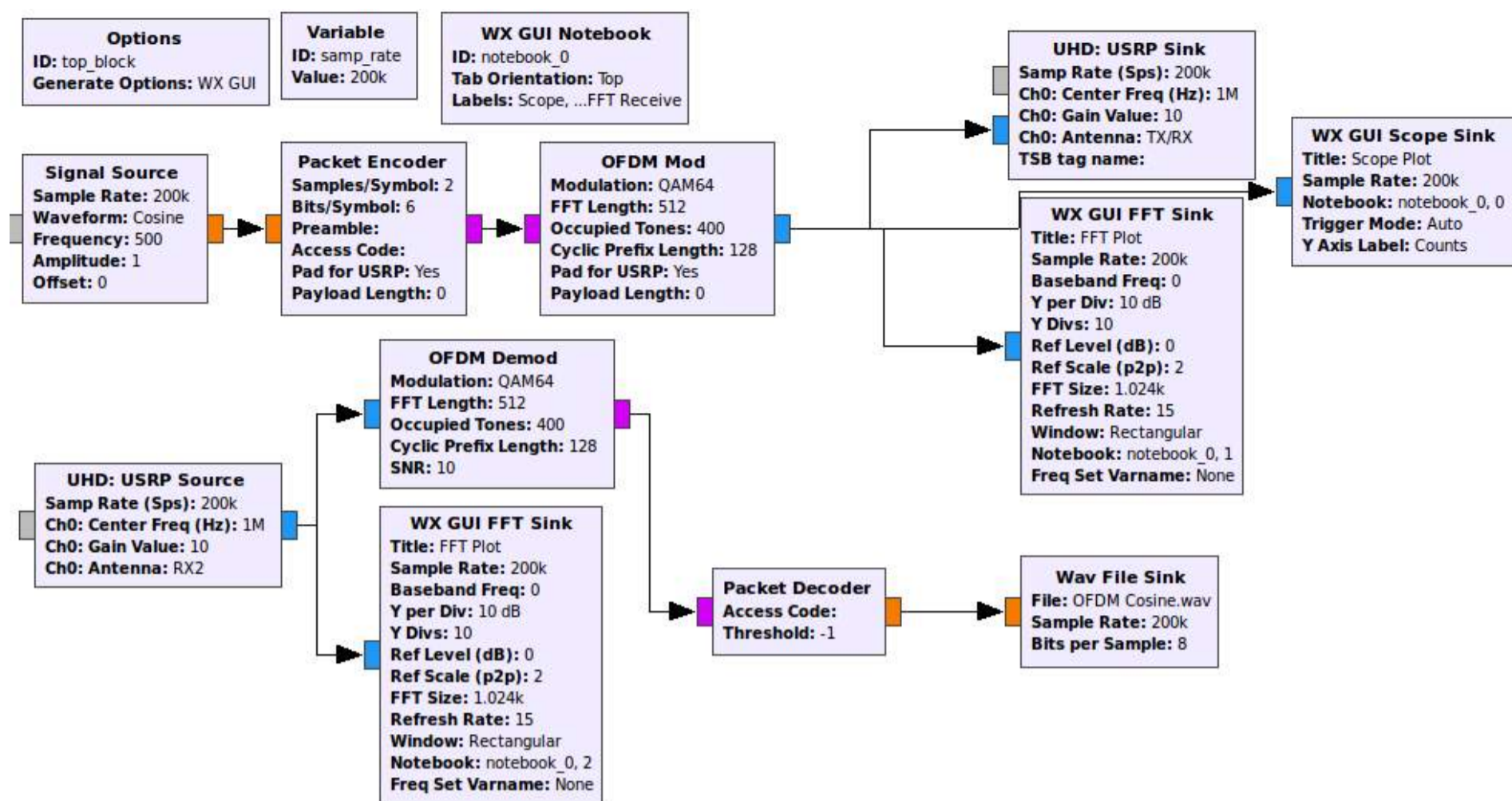


Fig 5: A typical GRC flowgraph with blocks

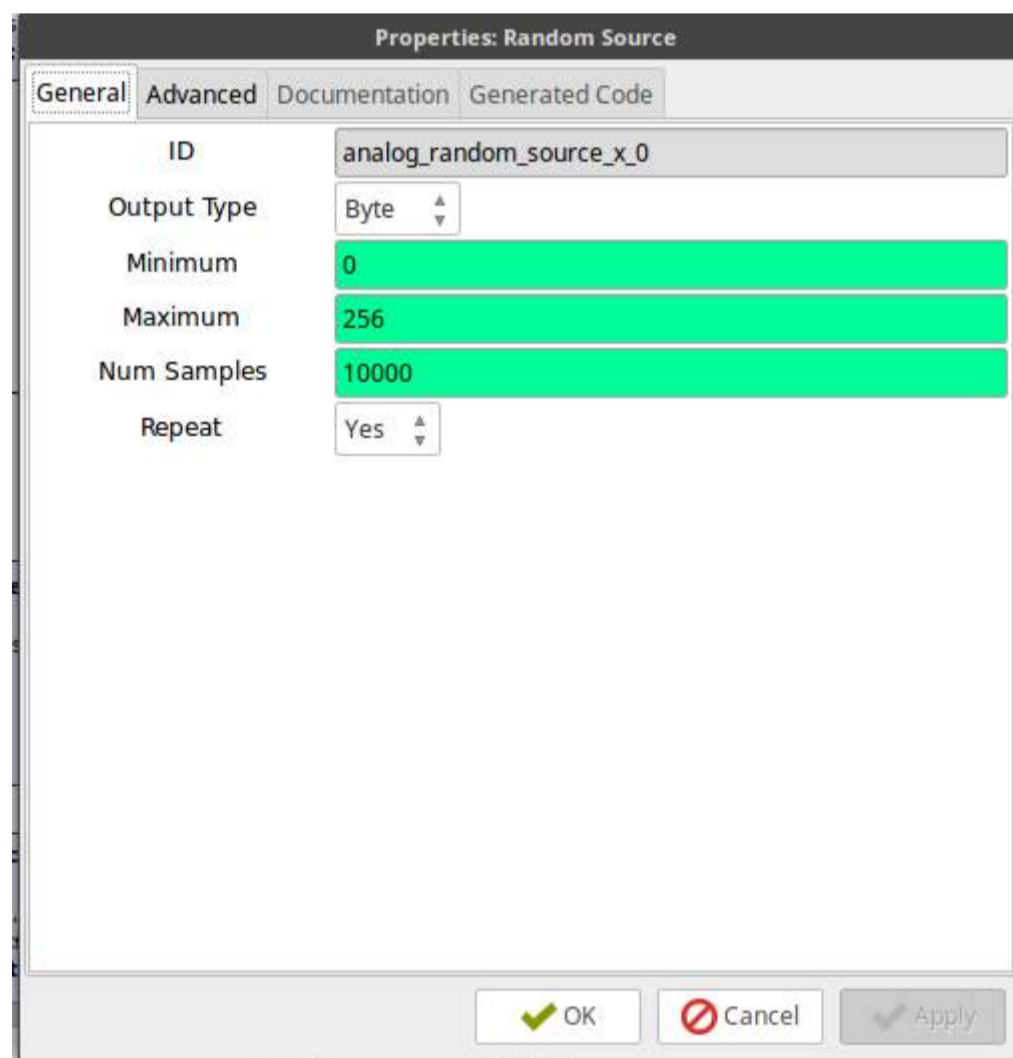


Fig 6: A typical Properties of a block

GRC mainly has four block types:

1. **Source blocks:** They are mainly used as the input source for the entire application being developed. There can be multiple signal sources in one application.
2. **Sink blocks:** They are used to store or output the processed signal into a file.
3. **Operation blocks:** They perform all the operation required for the application being developed.
4. **Visualization blocks:** They are used to view a graphical representation of the input or output signal.

A partial list of GNU Radio DSP blocks are given below:

- Mathematical operations such as, add, subtract, multiply, divide, power, logarithm and Logic Operations;
- FFT/IFFT blocks;
- Filters such as High pass filter, Low pass filter, Band pass filter, Band reject filter, Finite Impulse response(FIR) filter, Infinite Impulse response(IIR) filter;
- Modulation and demodulation schemes such as AM, FM, PSK, QAM, OFDM;

- Coding schemes such as Scrambler, Descrambler, Trellis coding, Viterbi decoding;

A partial list of GNU Radio signal source and sinks are given below:

- Constant source;
- Noise source;
- Pseudo random noise source;
- File source and sink;
- USRP source and sink;
- Audio source and sink;
- Graphical sinks such as Oscilloscope, FFT, Waterfall, Constellation and Histogram.

2.4.1 Universal Software Radio Peripheral (USRP)

USRP is a range of software defined radios designed and sold by Ettus Research and its parent company, National Instruments. Developed by a team led by Matt Ettus, the USRP product family is intended to be a comparatively inexpensive hardware platform for software radio, and is commonly used by research labs, universities, and hobbyists [5]. Most USRPs connect to a host computer through a high-speed link, which the host-based software uses to control the USRP hardware and transmit/receive data. Some USRP models also integrate the general functionality of a host computer with an embedded processor that allows the USRP device to operate in a stand-alone fashion.

For this project we have used N200 USRP. It falls under the highest performing class of hardware among the USRP family, which enables to rapidly design and implement powerful, flexible software radios systems [6].

SPECIFICATIONS					
Spec	Typ.	Unit	Spec	Typ.	Unit
POWER			RF PERFORMANCE (w/ WBX)		
DC Input	6	V	SSB/LO Suppression	35/50	dBc
Current Consumption	1.3	A	Phase Noise (1.8 GHz)		
w/ WBX Daughterboard	2.3	A	10 kHz	-80	dBc/Hz
CONVERSION PERFORMANCE AND CLOCKS			100 kHz	-100	dBc/Hz
ADC Sample Rate	100	MS/s	1 MHz	-137	dBc/Hz
ADC Resolution	14	bits	Power Output	15	dBm
ADC Wideband SFDR	88	dBc	IIP3	0	dBm
DAC Sample Rate	400	MS/s	Receive Noise Figure	5	dB
DAC Resolution	16	bits	PHYSICAL		
DAC Wideband SFDR	80	dBc	Operating Temperature	0 to 55°	C
Host Sample Rate (8b/16b)	50/25	MS/s	Dimensions (l x w x h)	22 x 16 x 5	cm
Frequency Accuracy	2.5	ppm	Weight	1.2	kg
w/ GPSDO Reference	0.01	ppm			

* All specifications are subject to change without notice.

Fig 7: Specifications of USRP N200

USRP is constructed out of the following components:

1. Gigabit Ethernet Interface to Host
2. ADC (Analog to Digital Converter)
3. DAC (Digital to Analog Converter)
4. PGA (Programmable Gain Amplifier)
5. Daughter boards
6. FPGA (Field Programmable Gate Array)

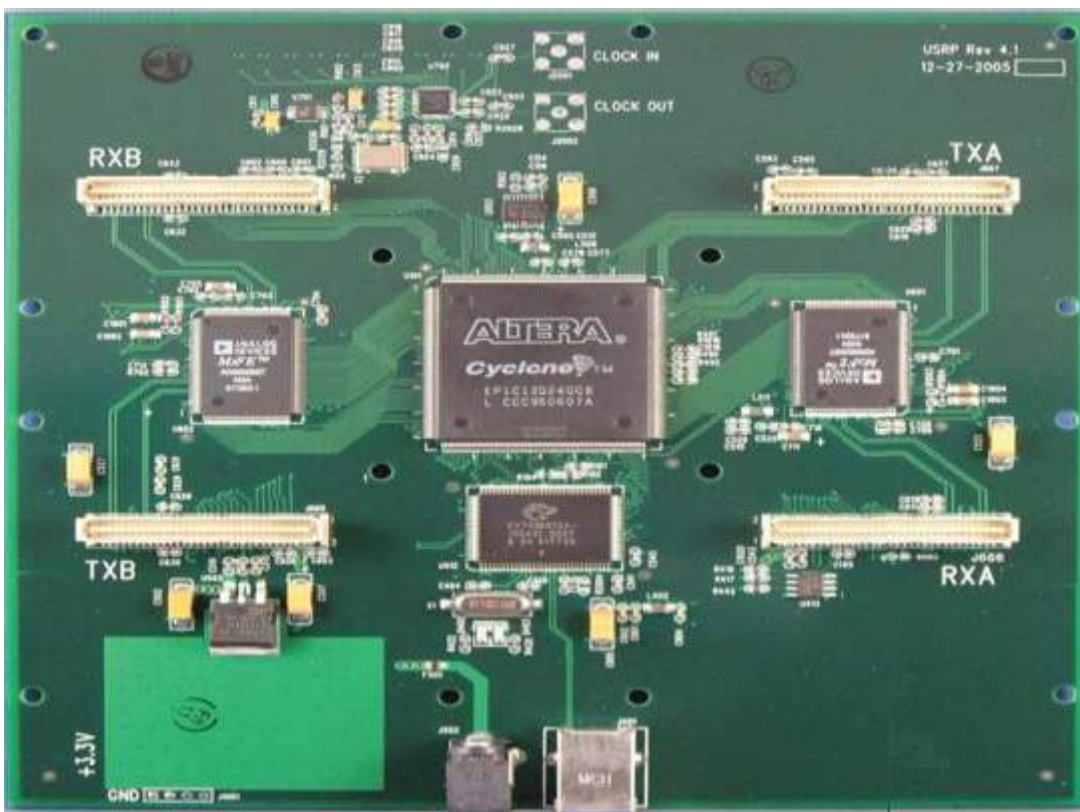


Fig 8: USRP motherboard and four daughter boards

2.4.2 MOTHERBOARD:

The USRP has 4 high speed analog to digital converts (ADCs), each at 12 bits per sample, 64MSamples/sec. There are also high speed digital to analog converters (DACs), each at 14bits per sample, 128MSamples/sec. These 4 input and 4 output channels are connected to an Altera Cyclone EP1C12 FPGA. The USRP connects to the computer via a high speed Gigabit Ethernet Interface only, and will not work with 100Mbps LAN. So in principle, we have 4 input and 4 output channels if we use real sampling. However, we can have more flexibility (and bandwidth) if we use complex (IQ) sampling. Then we have to pair them up, so we get 2 complex inputs and 2 complex outputs.[7] The Gigabit Ethernet Interface contains the firmware that defines its behavior. The firmware also takes care of loading the FPGA bit stream, The FPGA handles the high bandwidth computation and reduces the data rate to something we can send over the Gigabit Ethernet Interface. The Analog Device chip is a missed signal processor that takes care of the conversion between the analog and digital signals, digital up conversion in the transmit path and interpolation/decimation of the signals. The motherboard can have up to four daughterboard, two for receive and two for transmit to achieve wireless communication at different frequencies. They consist of the RF front end where the signal is up converted from the intermediate frequency to the carrier frequency or vice versa for the received signal. The following figure is a top view of the USRP.[7]



Fig 9: Top view of the USRP

2.4.3 DAUGHTER BOARD:

The daughterboard acts as the RF front-end of the SDR. In most of the daughterboard, the signal is already filtered, amplified and tuned to a baseband frequency dependent on the boards IF bandwidth and local oscillator frequency. There are also so called Basic Rx/Tx boards with no frequency conversion or filtering [9]. They only provide a direct RF connection to the motherboards.

FPGA:

In order to understand the functionality of each of the building blocks then FPGA is essential. What goes on in the USRP/USRP 2, FPGA is the most important part for the GNU Radio users. All the ADCs and DACs are connected to the FPGA. It performs high bandwidth math, and to reduce the data rated to something you can squirt over USN2.0 /GE ON USRP/USRP 2 respectively [7].

ADC Section:

There are 4 high-speed 12 bit ADC converters. The sampling rate is 64M samples per second. In principle, it could digitize a band wide as 32MHz[8]. For USRP2, there are two high-speed 14bit ADCs. There are two other auxiliary ADCs for each daughter board connector.

DAC Section:

At the transmitting path, there are also 4 high-speed 14 bit DA converters. The DAC clock frequency is 128 MS/s, so Nyquist frequency is 64MHz. For USRP 2, the DAC clock frequency is 400 MS/s, so the Nyquist frequency is 200 MHz. USRP 2 has main DAC (Dual of type AD9777 used at 400MS/s) and two auxiliary DACs (of type AD5623).[7]

CHAPTER 3

LITERATURE REVIEW-II

3.1.1 Modulation:

Modulation is defined as a process by virtue of which, some characteristic (amplitude, frequency or phase) of a high frequency sinusoidal wave is varied in accordance with the instantaneous amplitude of the baseband signal [10]. Here the baseband signal is known as the modulating signal and the carrier acts as the modulated signal. There are many different types of modulation of which some are listed below:

- Amplitude modulation (AM)
- Frequency modulation (FM)
- Phase modulation (PM)
- PSK (phase-shift keying)
- FSK (frequency-shift keying)
- ASK (amplitude-shift keying)
- QAM (quadrature amplitude modulation)

The general equation which defines the modulation of a high frequency carrier signal is,

As seen from the figure, GMSK's power spectrum drops much quicker than MSK's.

TIME DOMAIN RESPONSE:

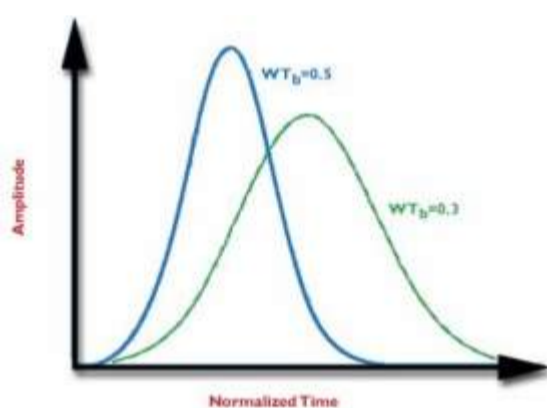


Fig 27: Time domain response of different Gaussian filters

Lower time-bandwidth products produce a faster power-spectrum. As a result lower time-bandwidths products have the pulse spread over a longer time, which can on the other hand cause inter symbol interference. Therefore, a compromise between spectral efficiency and time-domain performance is required. Thus, an intermediate time-bandwidth product must be chosen.

ADVANTAGES OF GMSK:

There are several advantages of GMSK modulation in radio communication system:

- It improves the spectral efficiency when compared to other phase shift keyed modes
- It can be amplified by a non-linear amplifier and remain undistorted.
- It is immune to amplitude variations and therefore more resilient to noise, than some other forms of modulation, because most noise is mainly amplitude based.
- Reduces sideband power
- ISI is tolerable
- Good BER performance
- Self-synchronizing capability

DISADVANTAGES OF GMSK:

- Higher power level than QPSK
- Requiring more complex channel equalization algorithms such as an adaptive equalizer at the receiver
- Probability of error is higher than MSK

PERFORMANCE CRITERIA:

The performance of a GMSK modem is generally quantified by measurement of the signal to noise ratio (SNR) versus BER. SNR is related to E_b/N_0 by

$$E_b/N_0 = S/R \cdot N_0 = S/N(B_n/R)$$

Where:

E_b - Energy/bit

S = Signal Power

R = Data rate in b/sec

N_0 = Noise

B_n = Noise bandwidth of IF filter

$B_n \times N_0$ = N noise power

APPLICATIONS:

- Most widely used in the Global System for Mobile Communications (GSM)
- Used it remote controlled devices i.e. cellular phones, Bluetooth headsets, etc.
- Used for GPRS & EDGE systems
- Used for CDPD (cellular digital packet data) overlay network.

3.4 Orthogonal Frequency Division Multiplexing (OFDM)

OFDM stands for Orthogonal Frequency Division Multiplexing. It is a multiplexing method, which means different data channels share the bandwidth available. OFDM is a multi-carrier modulation scheme, by reducing and eliminating the influence of inter-symbol interference to overcome the frequency selective fading in channel. It divides total available bandwidth into large number of closely-spaced orthogonal sub-carriers and simultaneously transfers signals on these sub-carriers with a low data rate. The data is divided into several parallel data streams or channels, each sub-carrier is modulated with a conventional scheme, such as quadrature modulation (QAM) or phase shift keying (PSK) at low symbol rate, maintaining total data rate similar to conventional single-carrier and 5 sub-carriers OFDM spectrum.

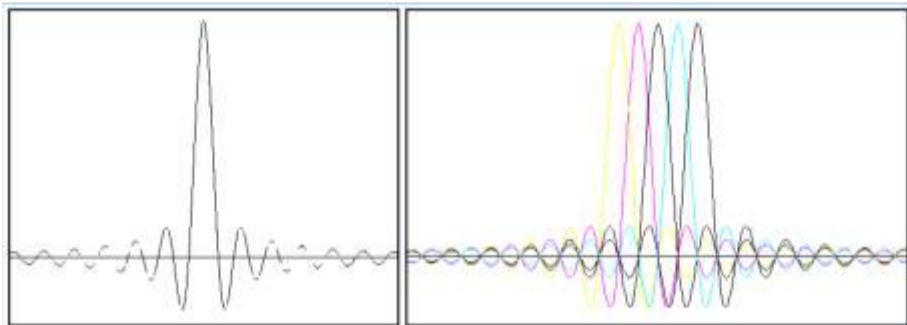


Fig 28: A sub-carrier channel (left) and 5 sub-carriers OFDM spectrum (right)

OFDM is one of the special cases of the Multi-Carrier Modulation (MCM) scheme which modulates data symbols in parallel on orthogonal subcarriers. The main idea of OFDM system is to separate the single high rate data stream into N parallel low data sub-streams that are modulated onto N orthogonal sub-carriers. This process is simply made in the discrete time domain through an N point Inverse Discrete Fourier Transform (IDFT) or Inverse Fast Fourier m (IFFT) element and the resulted signal is transmitted in sequence. The information at the receiver is retrieved by the performing a DFT/FFT unit. The block diagram of a conventional OFDM system is shown below.

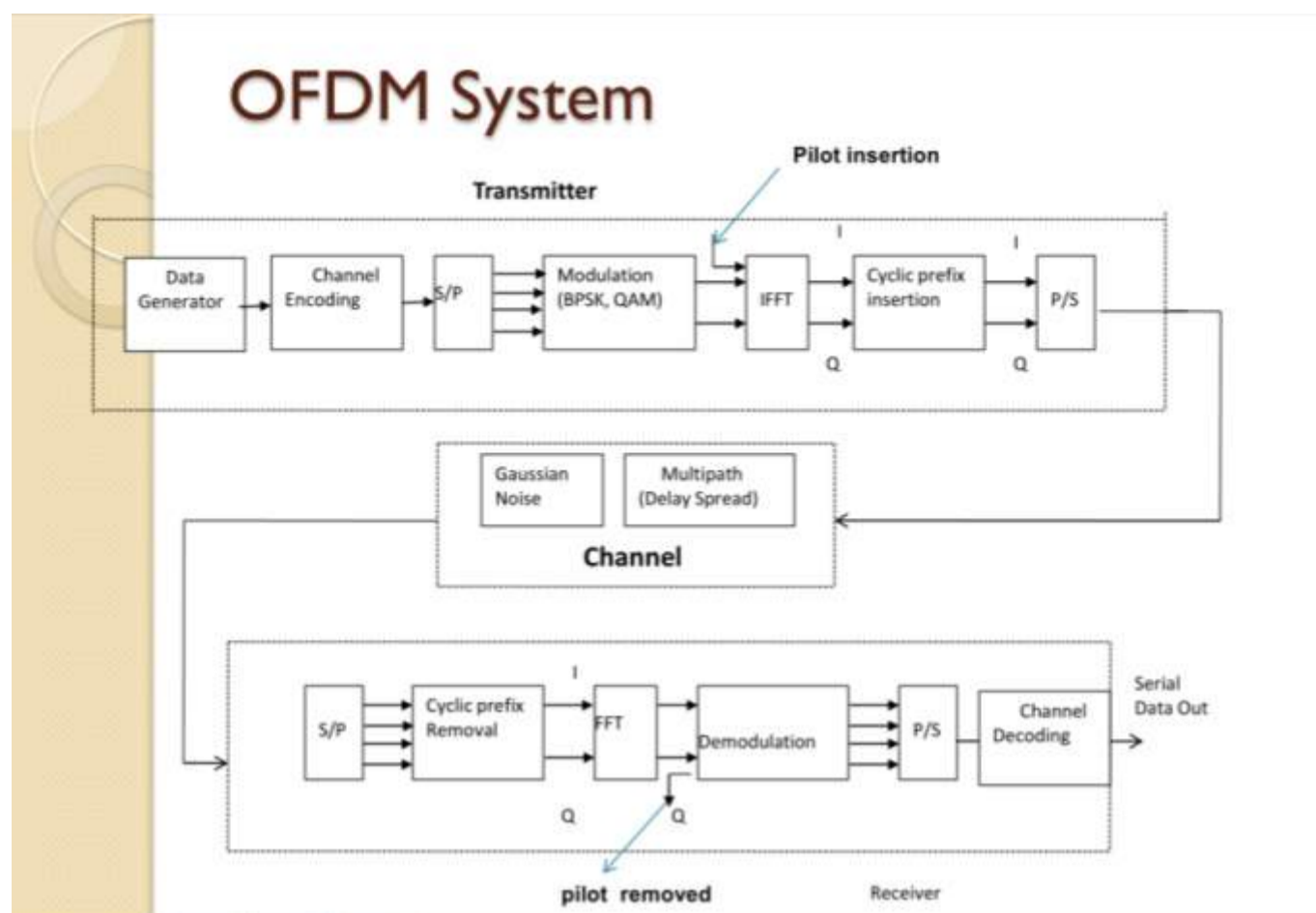


Fig 29: Block Diagram of OFDM system

Briefly, the OFDM transmitter works as follows:

- The data from the data generator is fed to channel encoding, where the bit stream is broken into groups of K bits.
- Symbol mapping block maps each group of K bits to the corresponding complex symbol (the modulated symbol)
- The IFFT is then computed
- Then the cyclic prefix is added to the data. The Cyclic Prefix is a periodic extension of the last part of an OFDM symbol that is added to the front of the symbol in the transmitter, and is removed at the receiver before demodulation. The Cyclic Prefix has two important benefits:
 1. The Cyclic Prefix act as a guard space between successive OFDM symbols and therefore prevents Inter-symbol Interference (ISI), as long as the length of the CP is longer than the impulse response of the channel.
 2. The Cyclic Prefix ensures orthogonality between the sub-carriers by keeping the OFDM symbol periodic over the extended symbol duration, and therefore avoiding Inter-carrier Interference (ICI).
- Next the signal is fed with Gaussian noise and Delay spread.

Briefly, the OFDM receiver works as follows:

- Now, this signal is first converted from series to parallel and then cyclic prefix is removed
- Next, Fast Fourier Transform is computed
- After that the signal is demodulated and is converted from parallel to series
- Then de-mapping is done to revert the original signal

DELAY SPREAD AND CYCLIC PREFIX:

In wireless communication systems the received signal will always be received many times due to the multipath propagation, this effect gives as a result in the receiver a number of signal with different amounts of delay respect the first multipath signal, which usually corresponds with the line of sight. The difference of delay between the first of the multipath components and the last one is called delay spread. The effect of delay spread is specially present in urban environments, in which the number of multipath components is higher than in rural environments, but also in environments where sender or receiver are moving at high speeds. There are two basic problems regarding delay spread [4].

1) First of all the different echoes that arrive at different times can come with a different phase in respect to the main signal and they can cause some distortion in the main component.

2) The second problem is the effect that the delayed echoes can cause in the next transmitted symbol. This is called Inter-symbol interference (ISI). The figure below shows the effect that multipath echoes can cause on a transmitted signal. To eliminate ISI completely a guard item is introduced for each OFDM signal as shown in the next figure. The guard time (GT) is chosen larger than the expected delay spread such that multipath component for one symbol cannot interfere with next symbol. The guard time consists of no signal. However the problem of Inter-Carrier Interference (ICI) would arise. ICI is the cross talk between different subcarriers which means they are no longer orthogonal. To eliminate ICI, OFDM symbols are cyclically extended in the guard time as shown in the third figure. This ensures that delayed replicas of the OFDM symbol always have an integer number of cycles within the FFT interval. And that we call Cyclic Prefix.

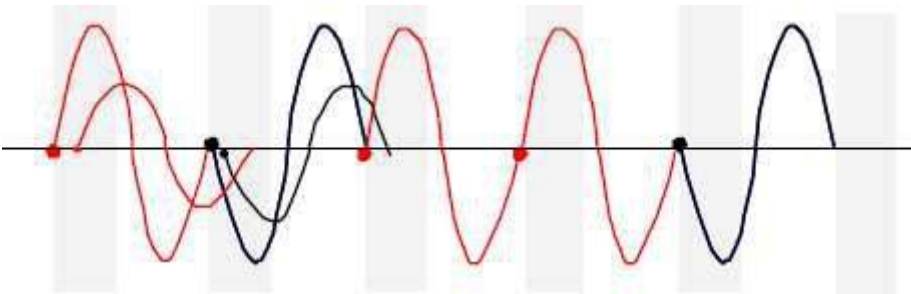


Fig 30: Signal in the presence of ISI

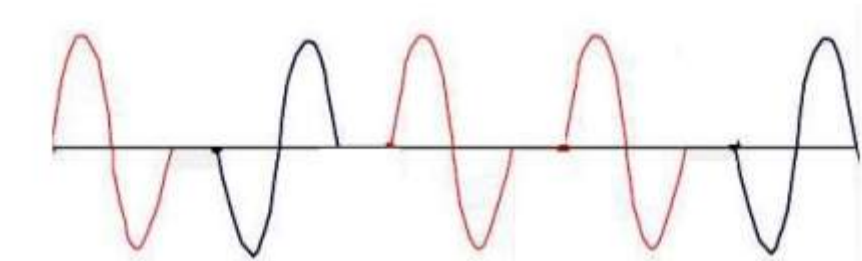


Fig 31: Signal in the presence of GT

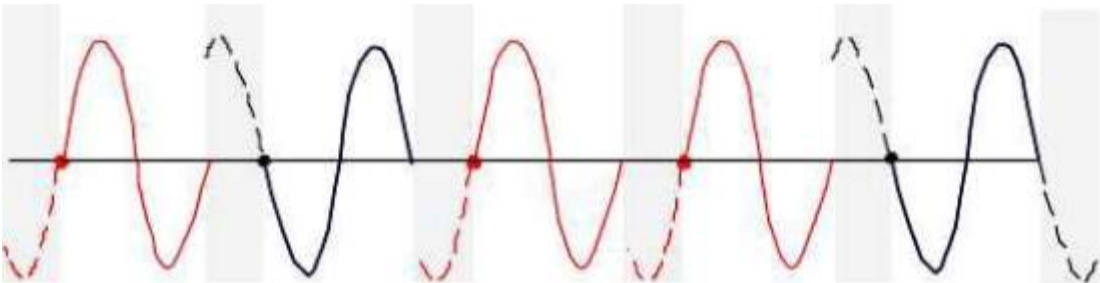


Fig 32: The effect of adding a cyclic prefix to the signal

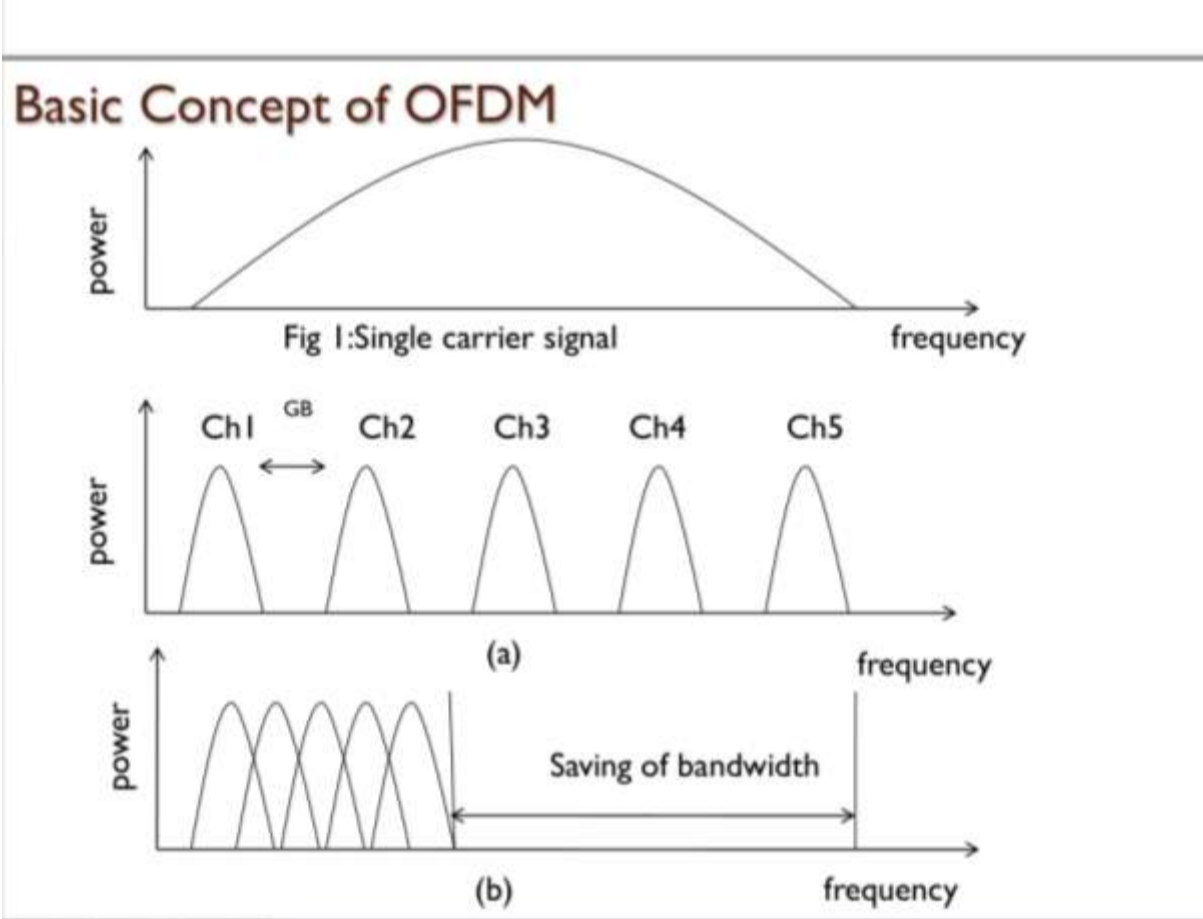


Fig 33: Basic concept of OFDM

MATHEMATICAL MODEL:

- $f_t = \sin m\omega t \cdot \sin n\omega t$

$$= \frac{1}{2} \cos(m-n)\omega t - \frac{1}{2} \cos(m+n)\omega t$$

- $\int_0^{2\pi} f_t dt = \int_0^{2\pi} \frac{1}{2} \cos(m-n)\omega t - \int_0^{2\pi} \frac{1}{2} \cos(m+n)\omega t$

If f_0 is the fundamental frequency of a signal and n is an integer number, then harmonics of the fundamental $f_n = n f_0$

ADVANTAGES AND DISADVANTAGES OF OFDM:

Advantages	Disadvantages

<ul style="list-style-type: none"> • Efficient way to transmit at very high data rates over multipath fading channels. • Four slow channels, loading is easier than single carrier • OFDM is robust against narrow band interference, as it will affect only portion of subcarrier and may overcome it by using coding • OFDM system offers high spectral efficiency 	<ul style="list-style-type: none"> • More sensitive to frequency offset and phase noise • OFDM has large PAPR which results in reducing the power efficiency or the RF amplifier • Sensitive to Doppler shift • Need very linear Amplifiers with large dynamic range
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

APPLICATIONS OF OFDM:

- Digital Television European and Australian standard
- WIFI
- WiMax
- Long Term Evolution
- CO-OFDM

PAPR:

The transmitted signals in an orthogonal frequency-division multiplexing (OFDM) system can have high peak values in the time domain since many subcarrier components are added via an inverse fast Fourier transformation (IFFT) operation. As a result, OFDM systems are known to have a high peak-to-average power ratio (PAPR) when compared to single-carrier systems. In fact, the high PAPR is one of the most detrimental aspects in an OFDM system as it decreases the signal-to-quantization noise ratio (SQNR) of the analog-digital convertor (ADC) and digital-analog convertor (DAC) while degrading the efficiency of the power amplifier in the transmitter. As a side note, the PAPR problem is more of a concern in the uplink since the efficiency of the power amplifier is critical due to the limited battery power in a mobile terminal [18].

PAPR problems are an important problem to take care of in an OFDM system. The PAPR of a signal is expressed by the following formula [18]:

- $PAPR = P_{peak} / P_{average} = 10 \log_{10} (\max[|X_n|^2] / E[|X_n|^2])$

FACTORS INFLUENCING PAPR:

- Modulation Schemes
- Number of subcarriers
- Over Sampling Rate
- Code Rate

PAPR REDUCTION TECHNIQUES:

- Amplitude Clipping and Filtering
- Coding
- Selected Mapping (SLM)
- Companding
- Tone Reservation

CONVOLUTIONAL CODING:

The convolutional coding technique is used to encode and decode a continuous stream of bits. The basic concept behind the convolution is the overlapping of two signals to form the other one. Because of the nature of convolution coding technique the binary bit stream source is convolved by applying some binary operations on them. It is a memory based system, which means the output bit is dependent of the current bit being encoded as well as the previous bit stream stored in the memory [19]. A convolutional code is a type of error-correcting code that generates parity symbols via a Boolean polynomial function to a data stream.

Convolutional codes are often characterized by the base code rate and the depth (or memory) of the encoder [n,k,K]. The base code rate is typically given as n/k, where n is the input data rate and k is the output symbol rate. The depth is often called ‘constraint length’ “K”, where the output is a function of the current input as well as the previous K-1 inputs. The depth may also be given as the number of memory element ‘v’ in the polynomial or the maximum possible number of states of the encoder [20].

Convolutional codes are used extensively to achieve reliable data transfer in numerous applications, such as digital video, radio, mobile communications and satellite communications.

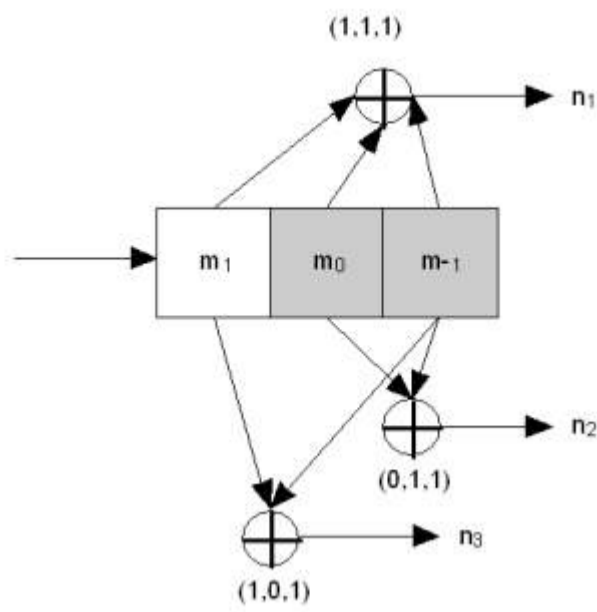


Fig 34: Rate 1/3 non-recursive, non-systematic convolutional encoder with constraint length 3.

BIT ERROR RATE (BER):

In telecommunication transmission, the bit error rate (BER) is the percentage of bits that have errors relative to the total number of bits received in a transmission, usually expressed as ten to a negative power. The BER is an indication of how often a packet or other data unit has to be retransmitted because of an error. Too high a BER may indicate that a slower data rate would actually improve overall transmission time for a given amount of transmitted data since the BER might be reduced, lowering the number of packets that had to be resent [21].

The definition of bit error rate can be translated into a simple formula:

BER=

Number of errors / Total number of bits sent

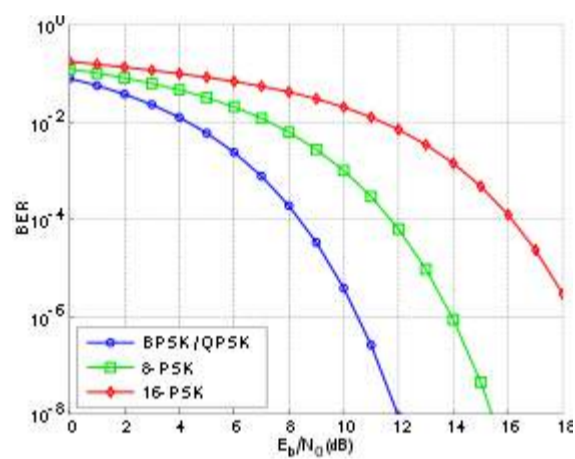


Fig 35: Bit error rate curves for BPSK, QPSK, 8-PSK and 16-PSK, AWGN channel

Factors affecting BER:

- Interference
- Increase transmitter power
- Lower order modulation
- Reduced bandwidth

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 Hardware and Software used:

Hardware:

- ASUS A43E
- DELL 3421
- NI Ettus USRP N200 x2
- Antenna x2
- LAN and power Cable x2

Software:

- Linux OS: Linux Mint 18
- GNU Radio Companion(GRC) 3.7.11.1

4.2 FM Receiver

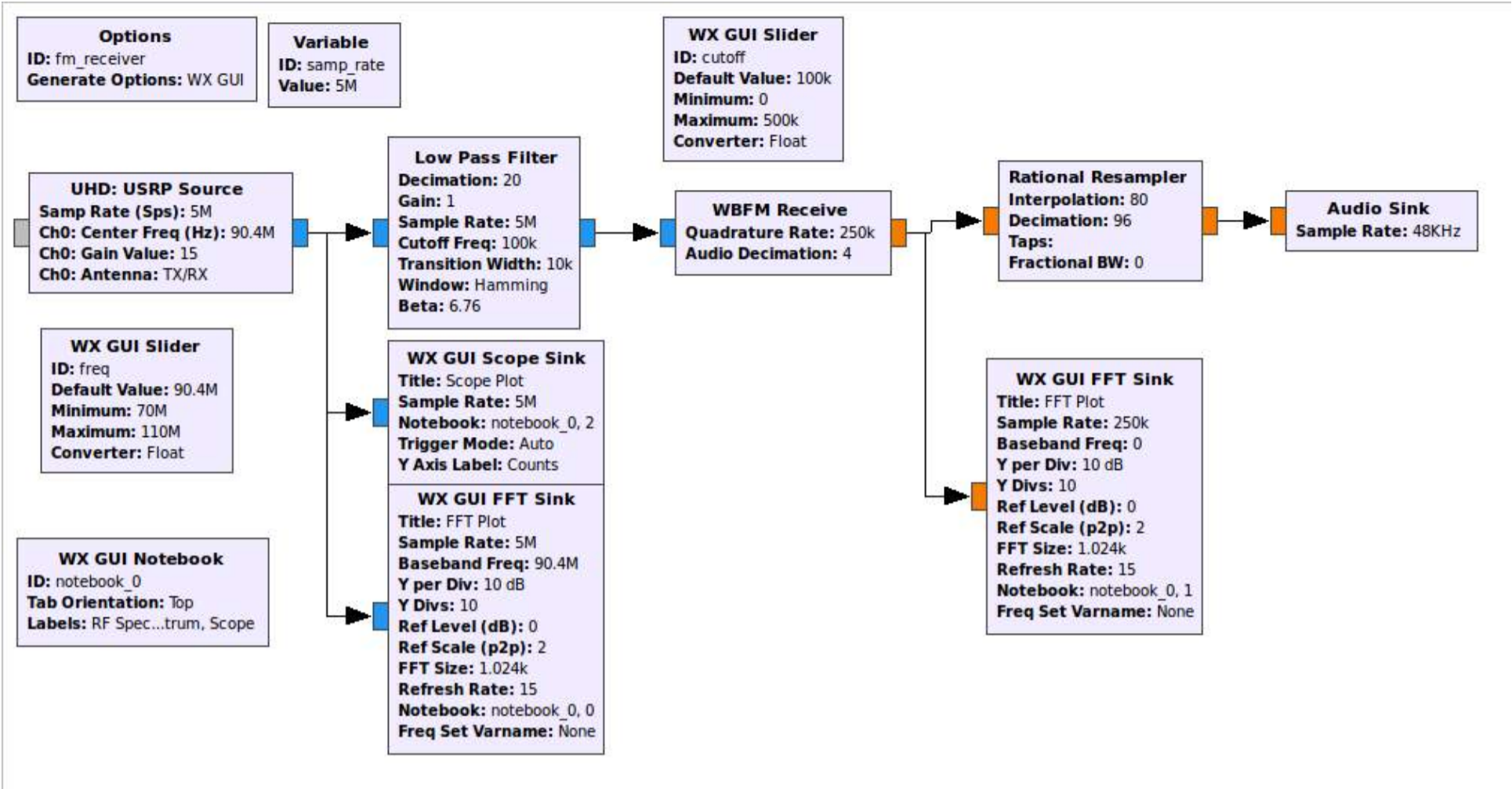


Fig 36: FM Receiver flow-graph

As our first GNU Radio application we have tested and received FM radio channels broadcasted in Dhaka. In Dhaka the occupied FM frequency range is between 80.0 MHz and 105.0 MHz. The FM signal is received from the airwave via the USRP board and is then processed by the FPGA converting the analog FM signal into digital signal by the ADC. The digitized signal is sent to the computer where it is analyzed and audio is sent to the speaker.

Here the following blocks have been used:

- Sample Rate:** The sampling rate has been kept at 5 MHz.
- USRP Source:** Receives the digitized FM signal from the USRP board. Here the center frequency of the USRP has been kept at 90.4 MHz as that was the most clearest in the Lab room.
- Low Pass Filter:** The incoming FM signal is very noisy thus we have used a LPF to reduce noise.
- WBFM Receive:** The Wide band FM Receiver discriminates only the FM signal from all the other signals and demodulates the FM signal.
- Rational Resampler:** Resamples the incoming demodulated signal into 48 KHz so that the laptop's speaker can output the signal as sound wave.
- WX GUI FFT Sink:** Show the graphical representation of the FFT of the incoming FM signal and the demodulated signal.
- Audio Sink:** Sends the demodulated and resampled signal to the laptop's speaker.



Fig 37: FM Receiver FFT

4.3 FM Transmitter

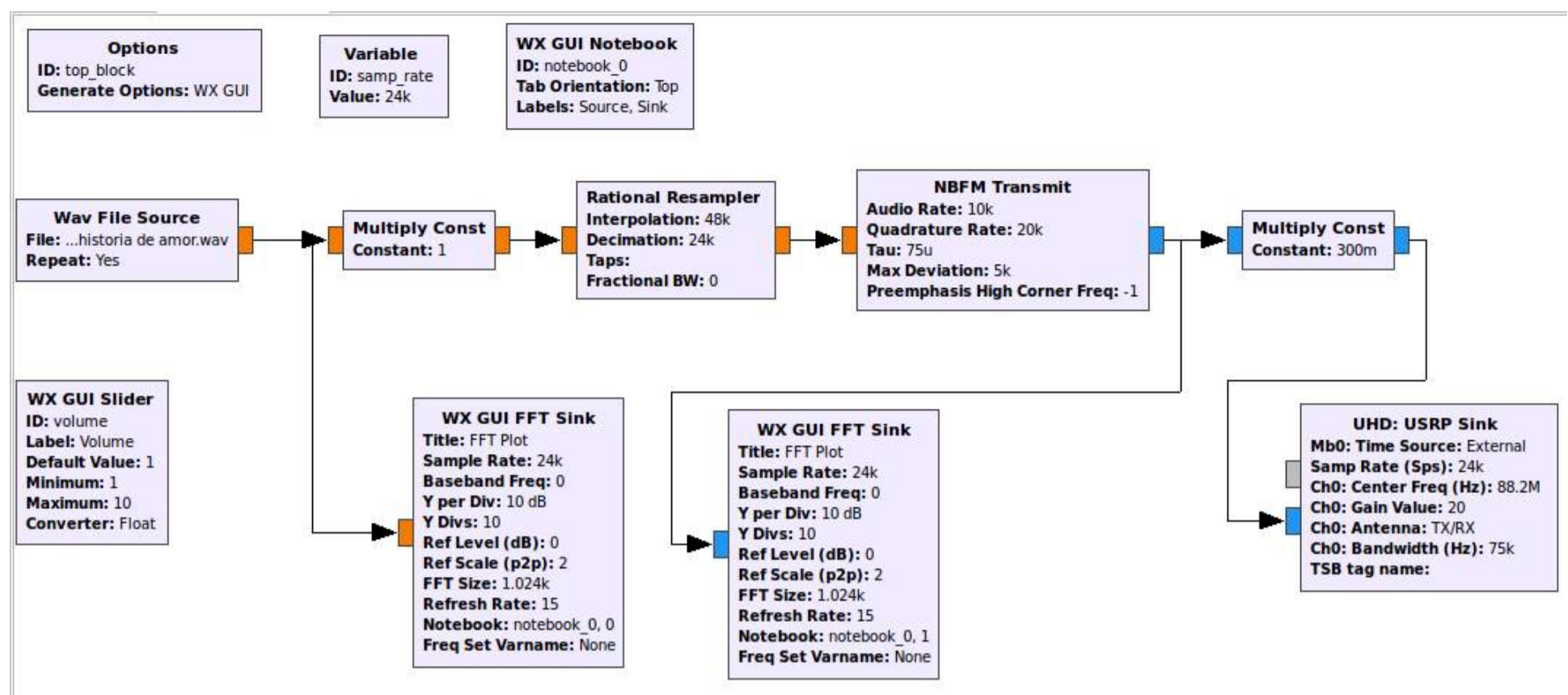


Fig 38: FM Transmitter flowgraph

Here we have made a FM transmitter i.e. an FM channel of our own, broadcasting any pre-recorded or real-time audio.

Wav File Source: The wav file source acts as the file sink. We use the wav source to input an audio file in wav format, to be transmitted.

Multiply Const: This first multiply constant uses a multiplier of 1. This is used to increase the gain and thus the volume of the output signal.

Rational Resampler: Takes the 48 KHz input audio and resamples it into a lower rate of 24 KHz using the decimation option.

NBFM transmit: The Narrow band FM Transmitter receives the 24 KHz audio and Frequency modulates the signal.

USRP Sink: Takes the FM signal and interfaces it to the USRP board. The antenna in the USRP radiates the signal into space. The center frequency of the USRP is set at 88.2 MHz. Thus our signal masks and occupies the 88.2 MHz frequency and tuning to that frequency using a cellphone we can hear the audio file.

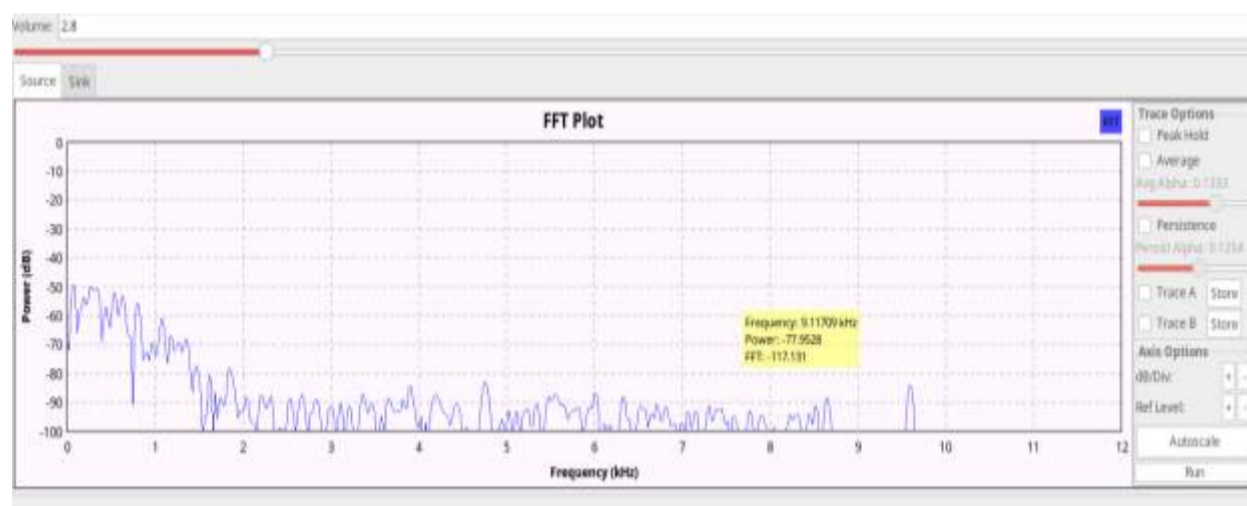


Fig 39: FM Transmitter audio input FFT

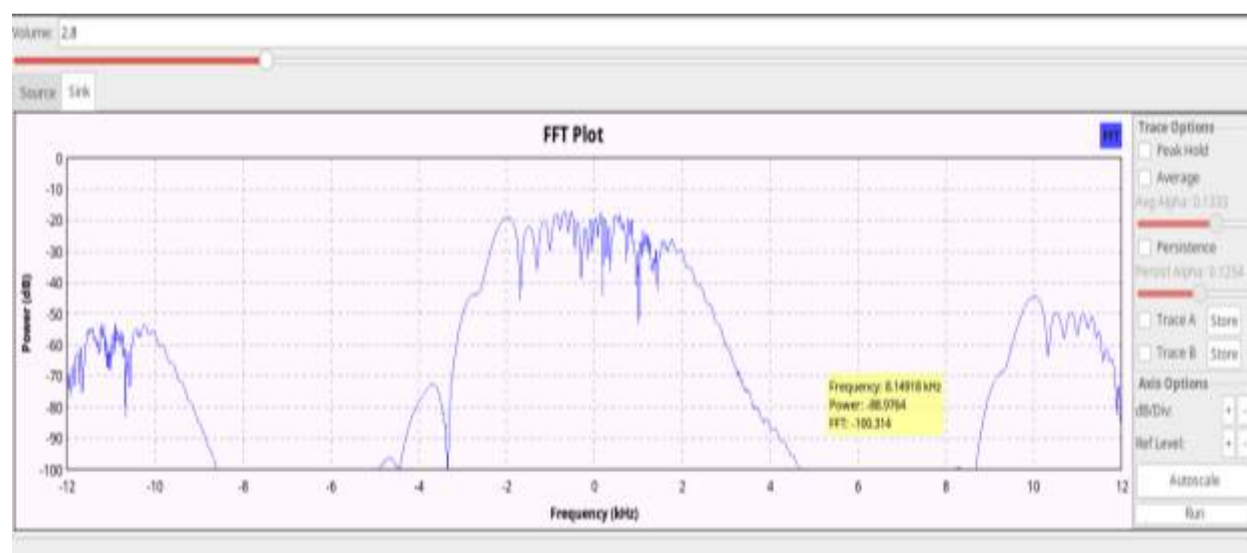


Fig 40: FM Transmitter audio output FFT

4.4 QPSK Modulator and Demodulator

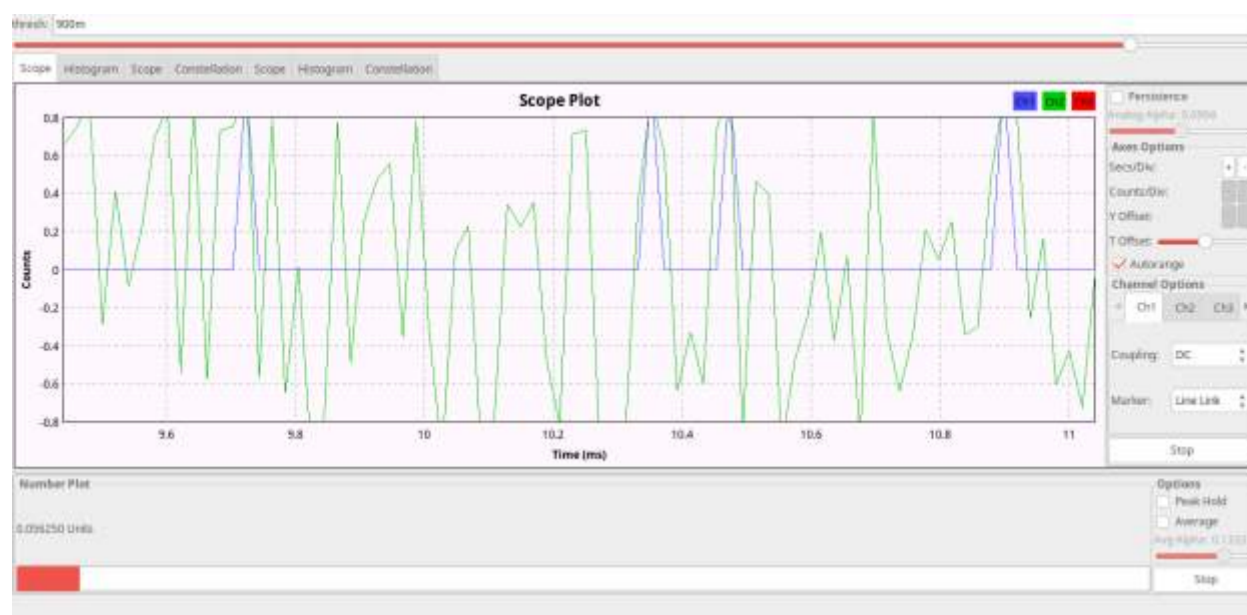


Fig 42: QPSK oscilloscope plot

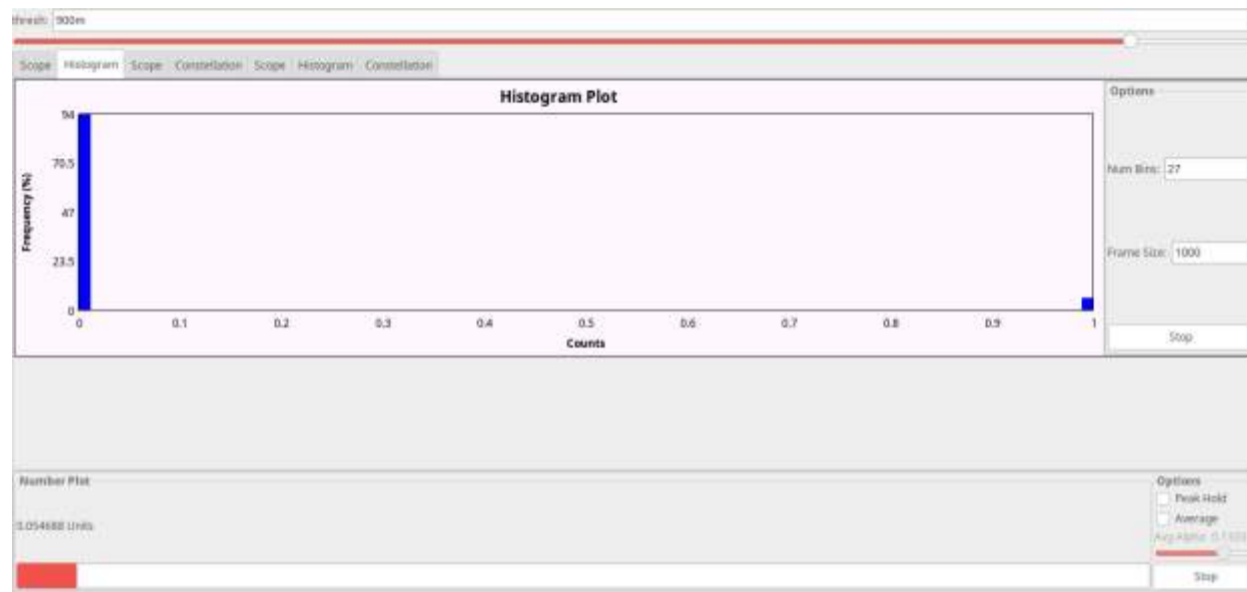


Fig 43: QPSK Histogram plot

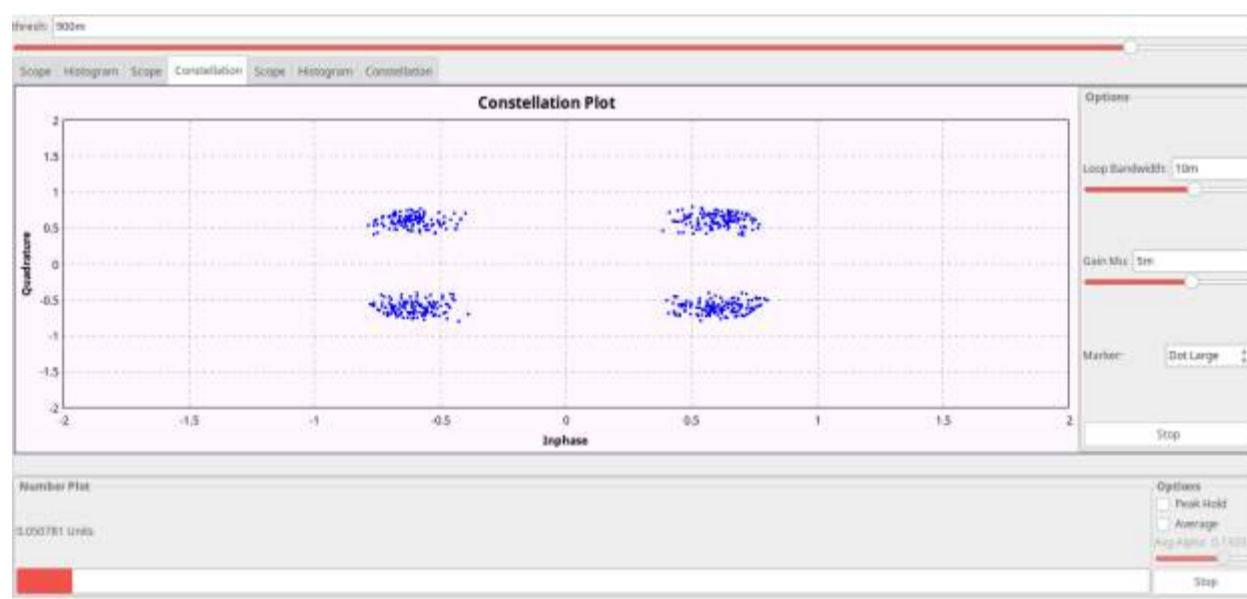


Fig 44: QPSK Constellation plot

The curves of the QPSK modulated random signal is as expected by QPSK model. The oscilloscope curve does not show any clipping as there is no noise.

The histogram shows two values, 0 and 1. The 0 value is for signal without any error and 1 for signal with error. Here we can see that a maximum of the signal is at the 0 mark, showing that a maximum of the signal is error free.

The constellation diagram shows four distinct regions occupied by an ideal QPSK signal, without any region of intersection.

Graphical Curves after Noise Addition

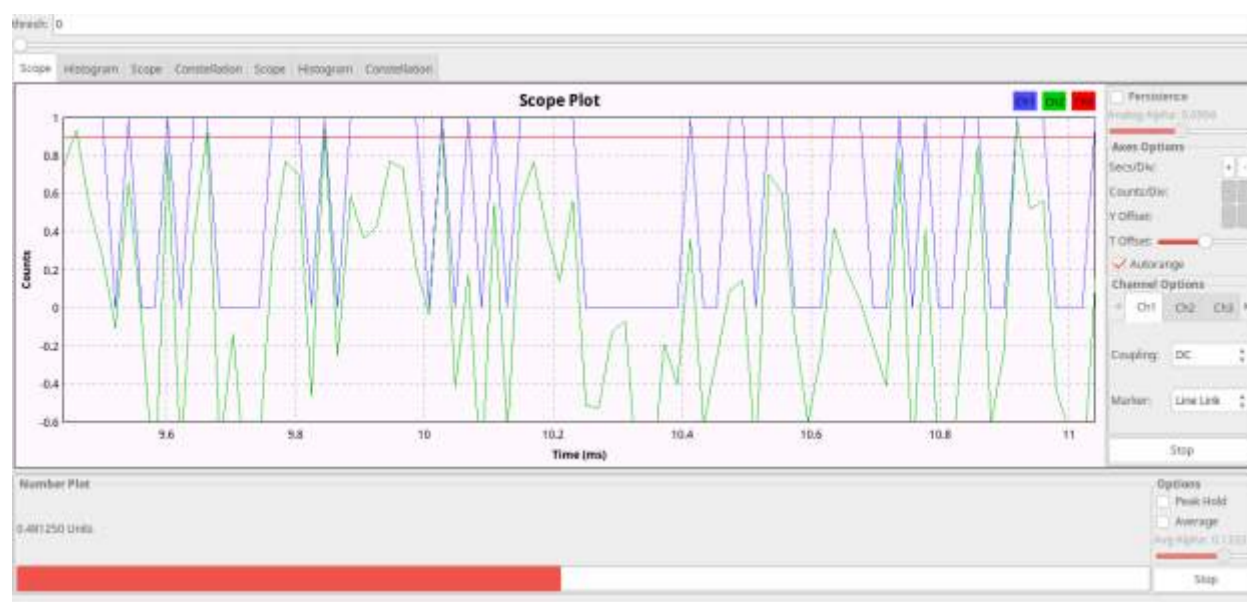


Fig 45: QPSK Oscilloscope noise plot



Fig 46: QPSK Histogram noise plot

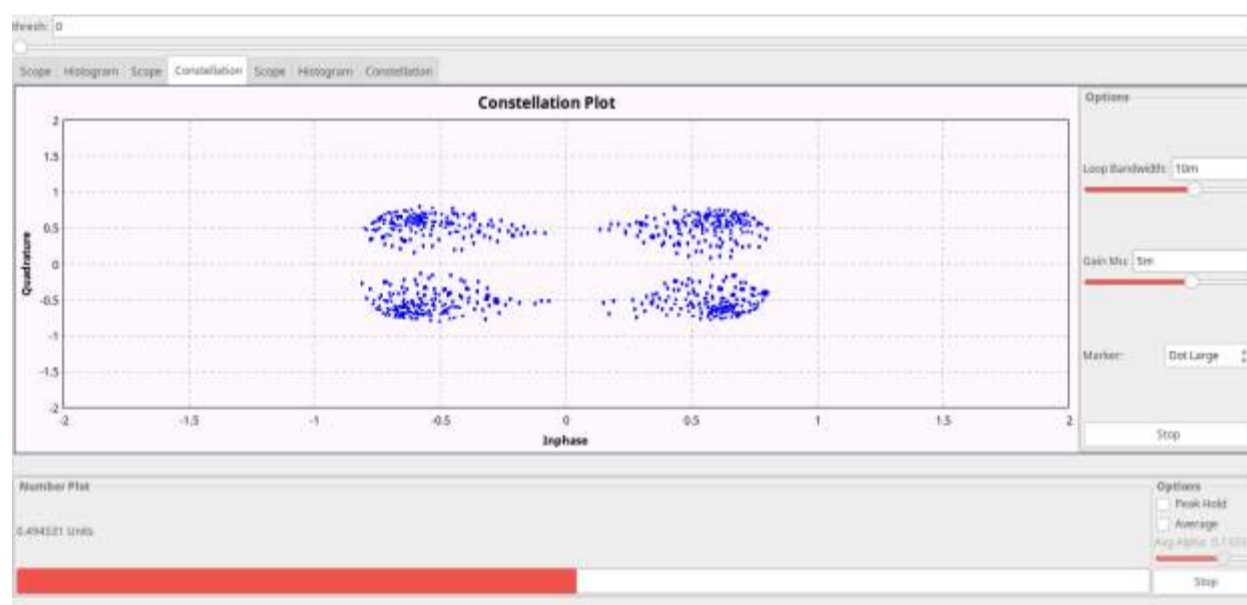


Fig 47: QPSK Constellation noise plot

Here is noise is added by sliding the threshold slider. We can see the effect of noise on the QPSK signal's oscilloscope. The maximum amplitude of the signal above 900m is clipped and the signal is generally noisier.

We can see the BER value of signal as 0.50 in the number slider.

In the histogram plot the effect of noise is more apparent. The plot shows an equal distribution in both the 0 and 1 mark. This means that the probability of the signal being the original signal to the noisy error signal is 0.5 i.e. 50% chance.

The effect of noise on the constellation diagram is also very prominent. In the no-noise scenario we had seen four distinct non-intersecting regions in the constellation plot. But now we can see that the four regions are still visible but they are intersecting each other. Thus we know that the phases have intersected and much of the signal is lost due to this.

4.5.1 8PSK

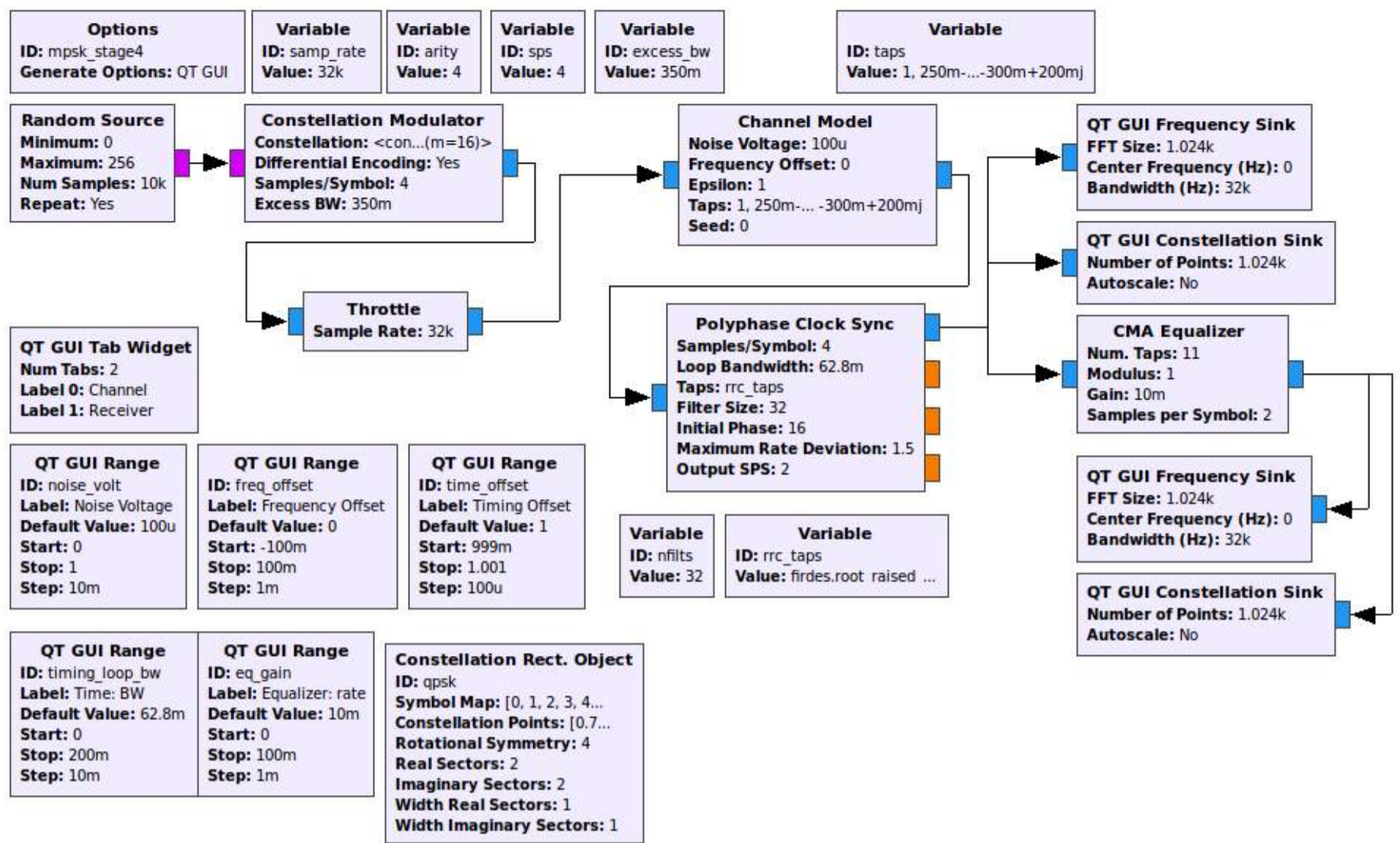
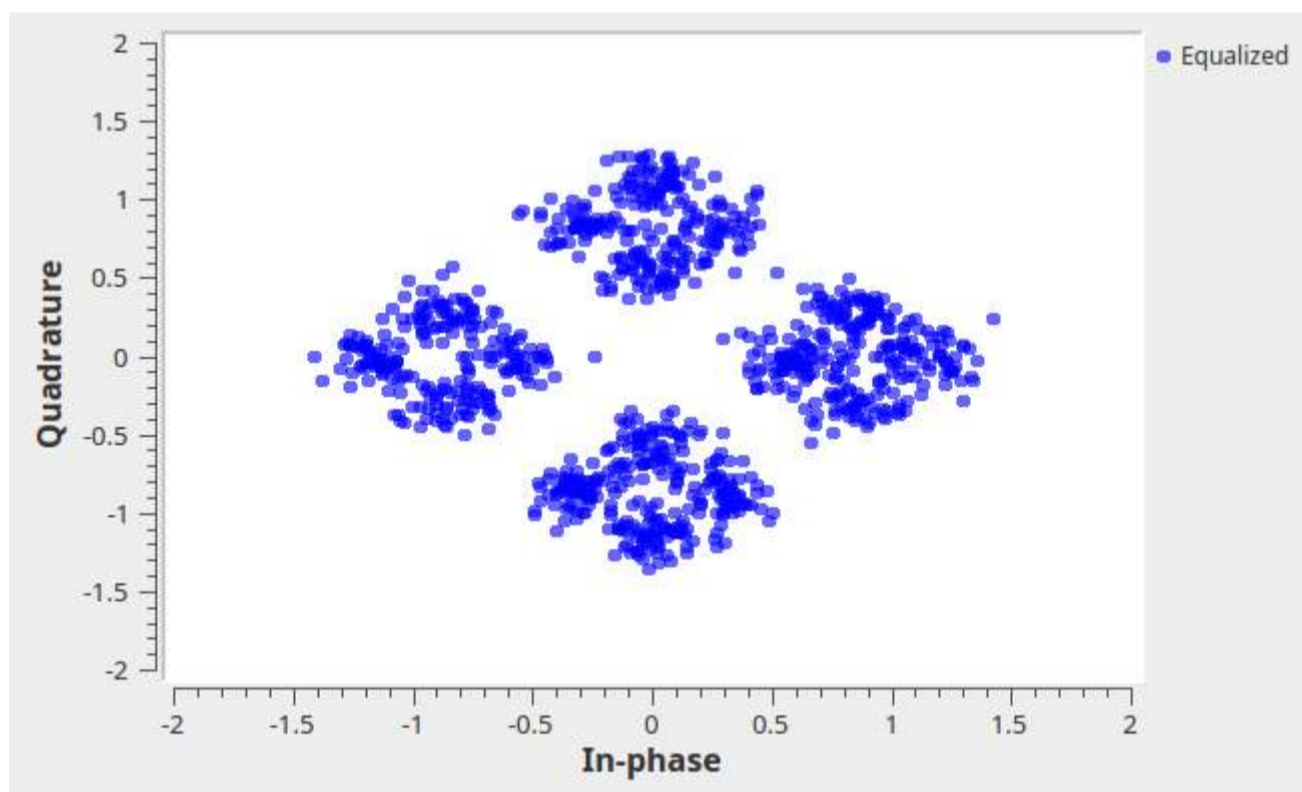


Fig 50: 16PSK flowgraph

The flowgraph for the 16PSK signal is the same as the 8PSK signal with the difference of 16 constellation points in the modulator block.



16PSK Constellation plot

Here we can see sixteen distinct regions in the constellation diagram of the 16PSK signal.

4.6.1 GMSK Image Send and Receive

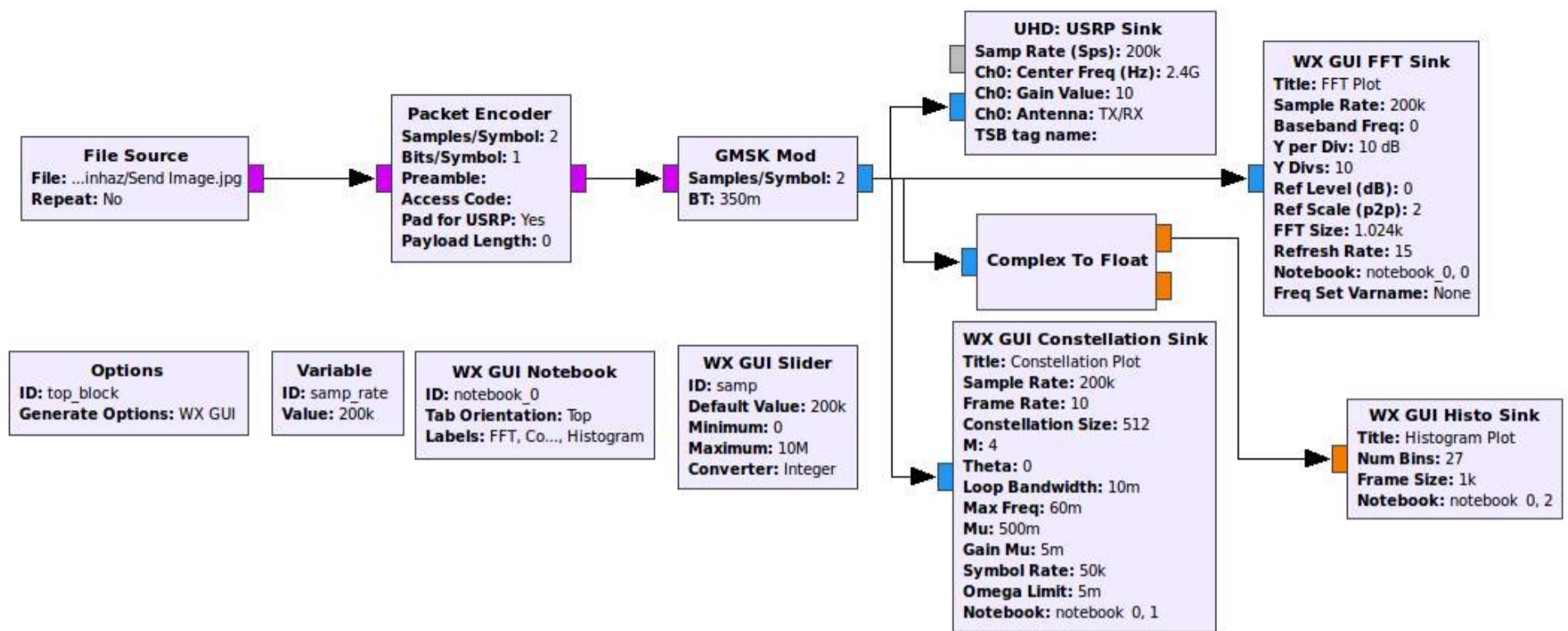


Fig 51: GMSK Image Send

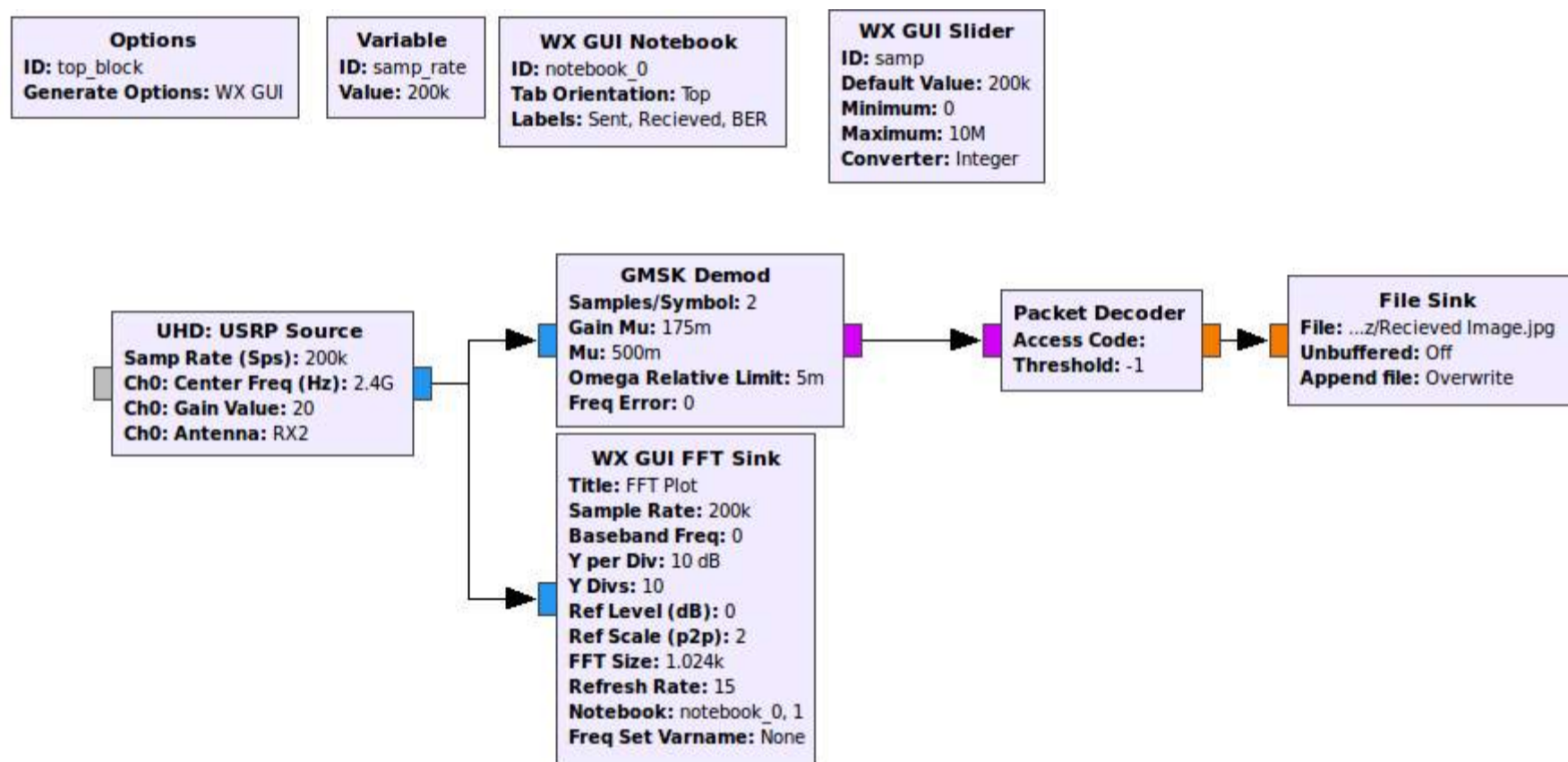


Fig 52: GMSK Image Receive

Here we have constructed a flowgraph which will send and receive a JPEG Image using GMSK. The original file size is 37,994 bytes and the size of the received image is 32,768 bytes. This shows that there was some data lost in the channel i.e. space, due to atmospheric noise.

The BER = $(37994 - 32768) \div 37994 = 0.137548034$

File Source: This block acts as the recipient of the JPEG image file. The source is nonrepeating thus after the file has been sent no more signal is sent.

Packet encoder: Is a block which packetizes data. Packet bytes go in, they get wrapped into a packet of a given payload length with a header, access code, and preamble.

GMSK Mod: This block modulates the encoded binary value of the image file using samples per symbol rate of 2.

GMSK Demod: This block demodulates the signal received by the USRP Source.

Packet decoder: The decoder looks for the access code with the number of available bits wrong. When it's found, it reads the header to get the payload length, extracts the payload, and outputs the payload.

File Sink: The decoded signal is sent to the file sink. This block writes the demodulated decoded signal to the disk as a JPEG file.

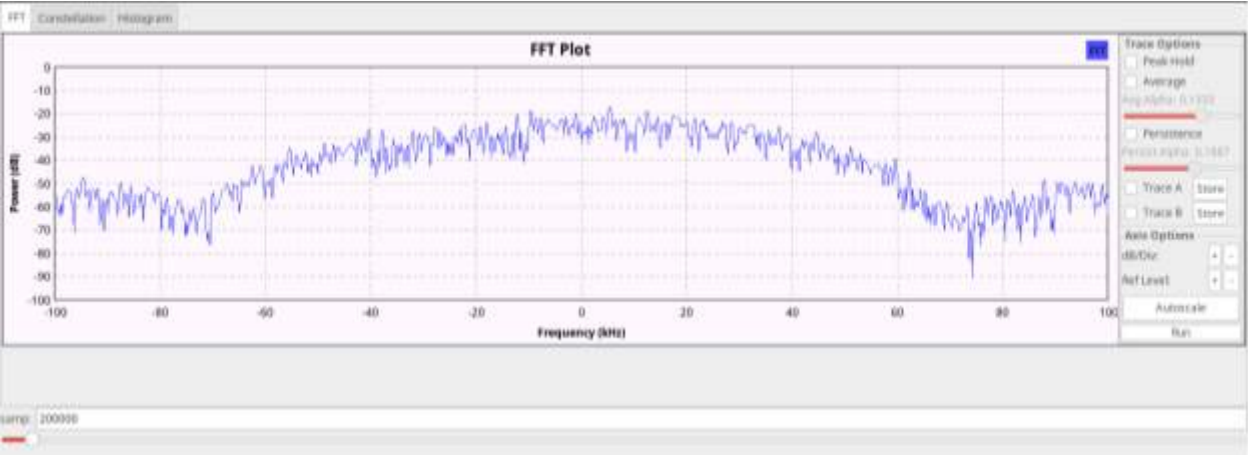


Fig 53: GMSK Image Send FFT

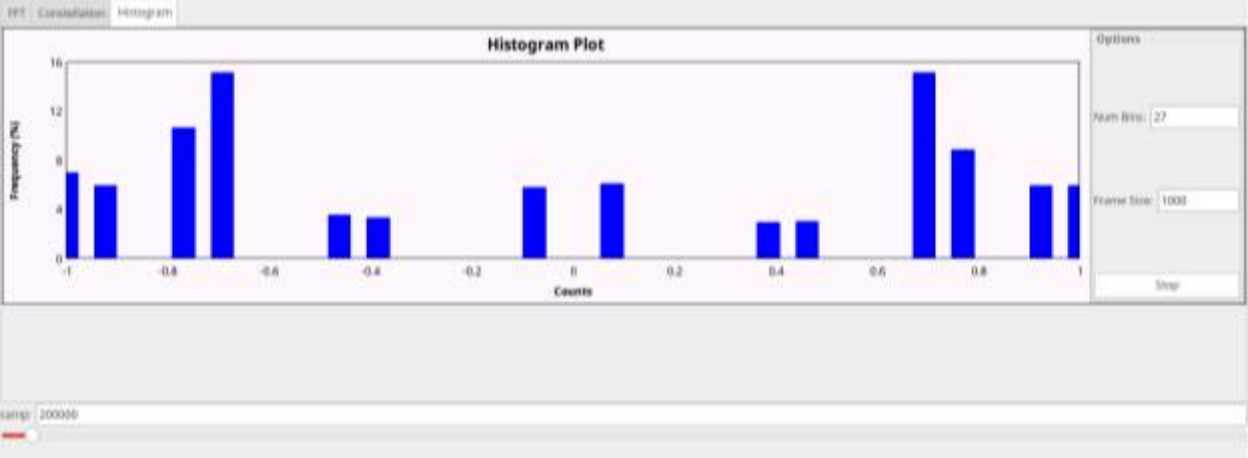


Fig 54: GMSK Image Send Histogram

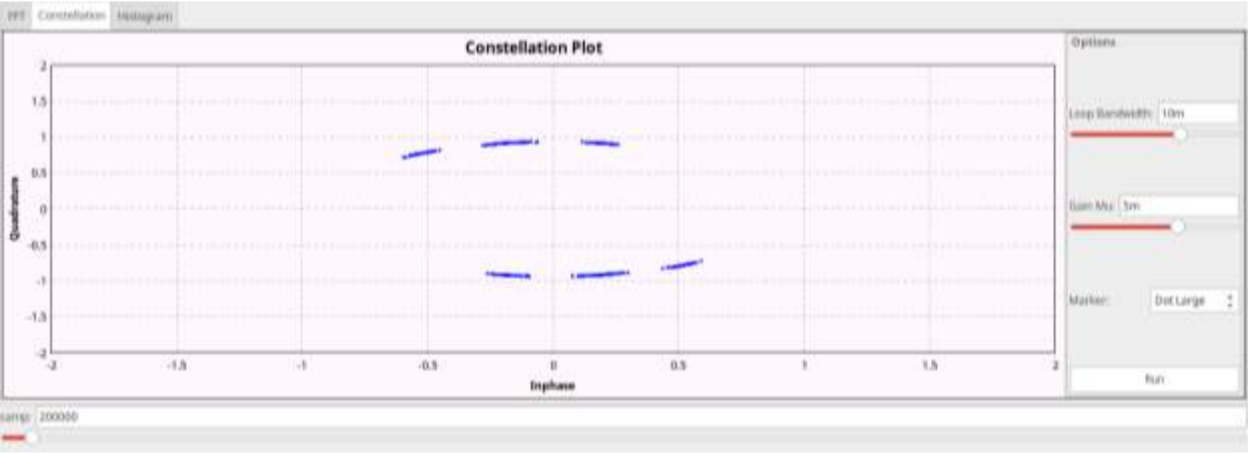


Fig 55: GMSK Image Send Constellation

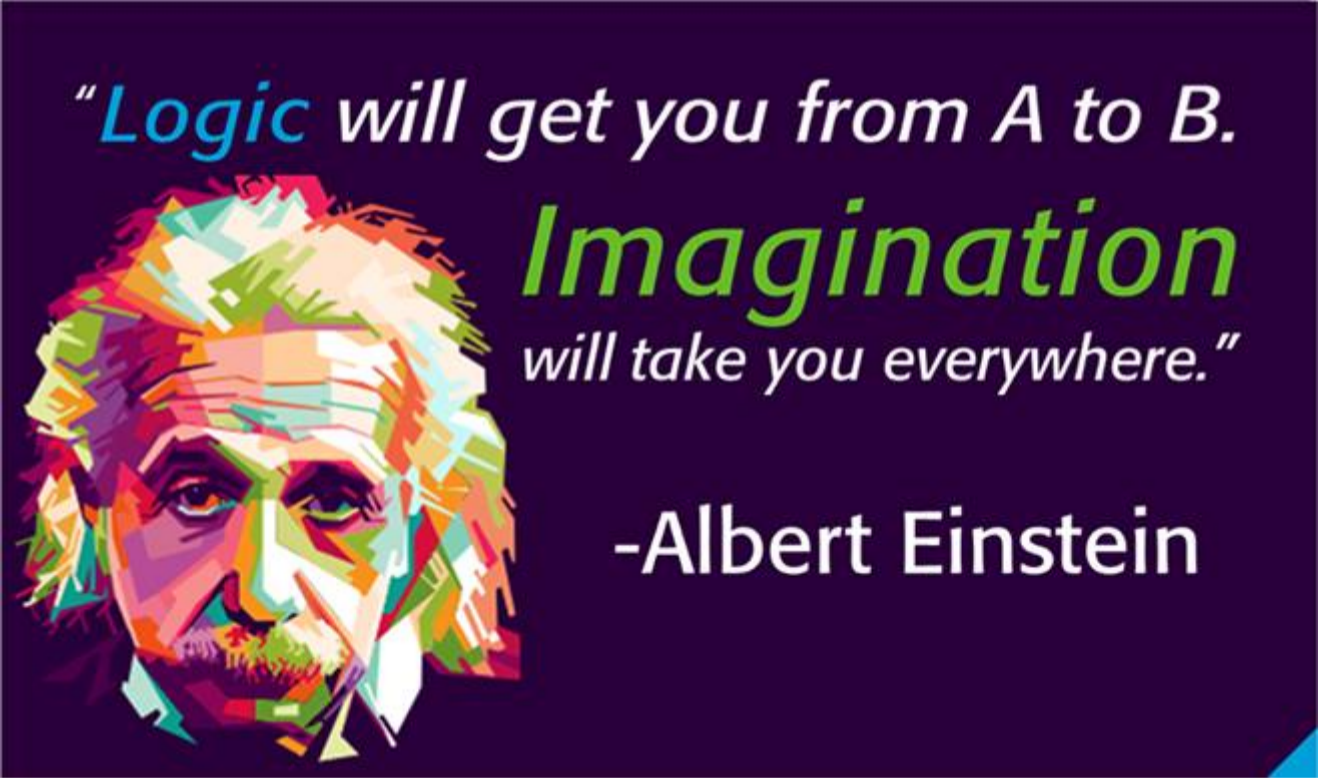


Fig 56: GMSK Sent Image

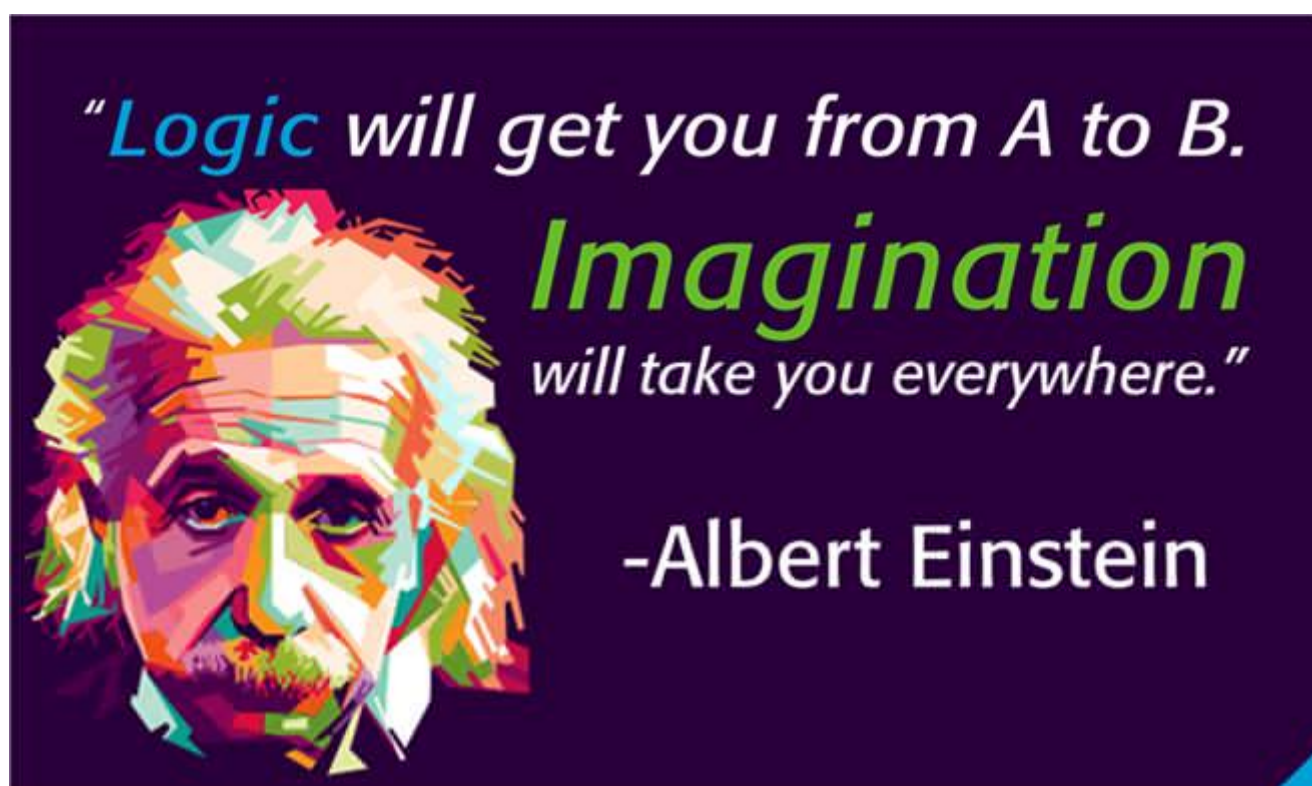


Fig 57: GMSK Received Image

4.6.2 GMSK Audio Send and Receive

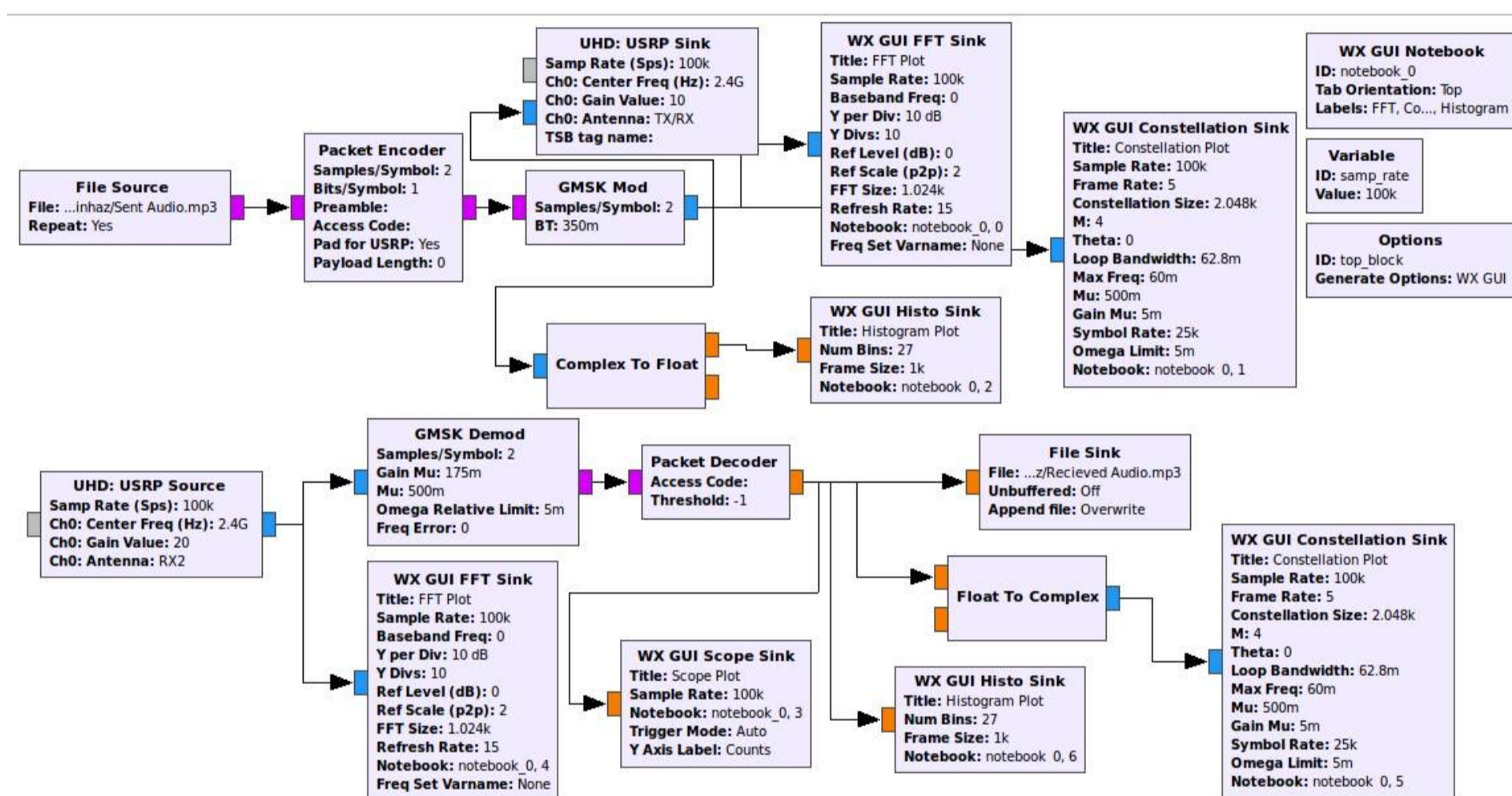


Fig 58: GMSK Audio Send and Receive

In this flow-graph we have sent and received an audio file in the form of an Mpeg-2 III, commonly known as, MP3 file. The file was input using the File Source block. The signal was then encoded and the modulated using the GMSK Mod block. The modulated signal was sent to the USRP Sink block to transmit the signal to the airspace. The output of the GMSK Mod block was also sent to the different waveform GUI's.

In the demodulation side, the USRP Source receives the GMSK modulated audio signal which is then demodulated. The resulting signal is decoded via the packet decoder. The file is written to disk using the file sink and the various waveforms viewed using the GUI.

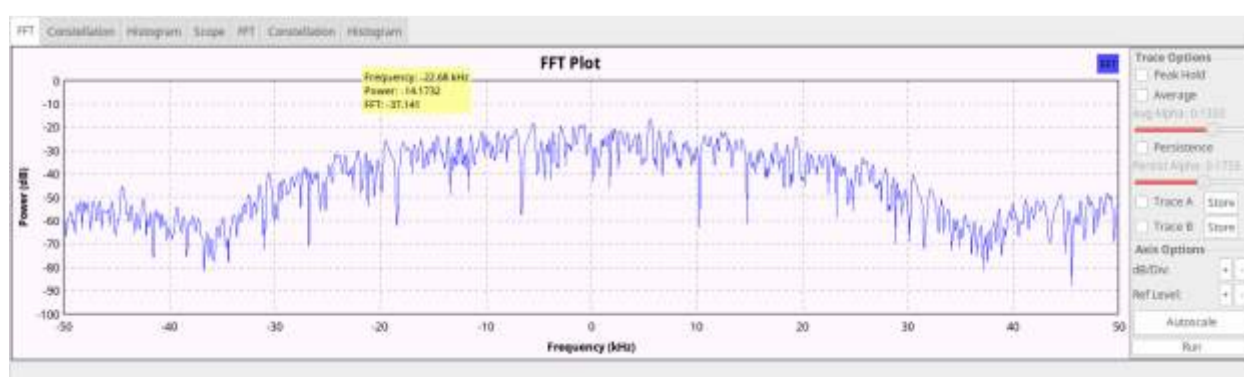


Fig 58: GMSK Sent Audio FFT

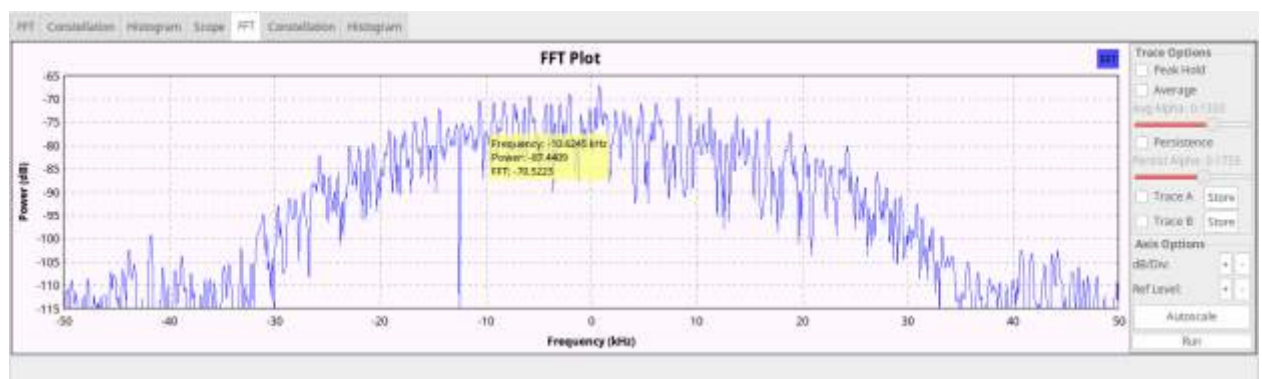


Fig 59: GMSK Received Audio FFT

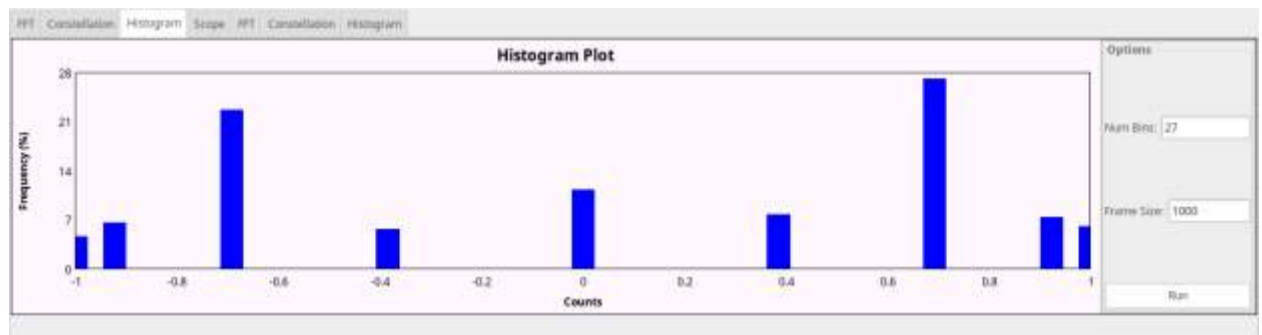


Fig 60: GMSK Sent Audio Histogram

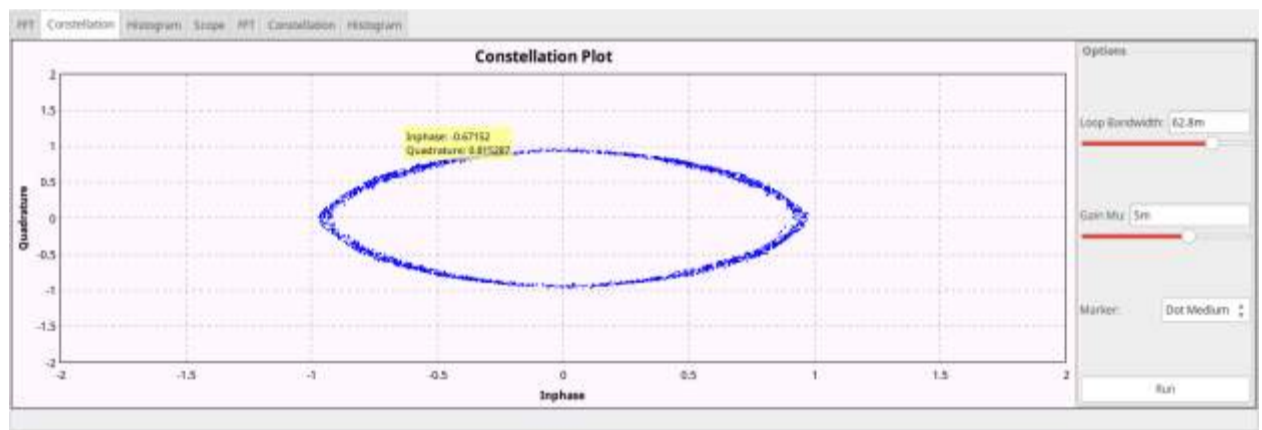


Fig 61: GMSK Sent Audio Constellation

4.7 OFDM Send and Receive

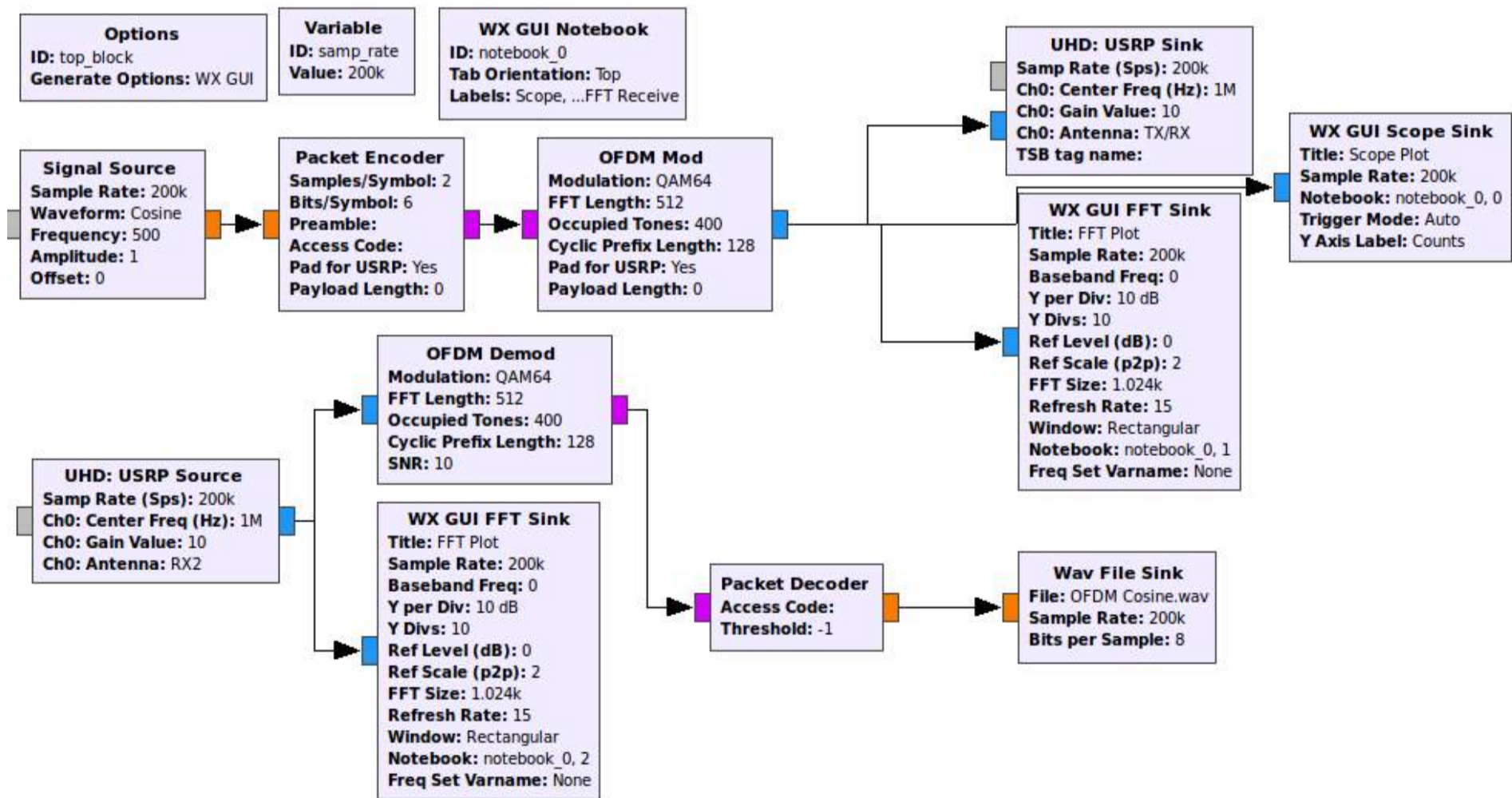


Fig 62: OFDM Send Receive

Here we have attempted to send and receive a sinusoidal signal using OFDM. Here we have used Quadrature amplitude modulation, QAM64 as the modulation scheme. The signal is modulated and then sent to the various waveform GUI's.

During demodulation the USRP source receives the OFDM signal and sends it to the demodulator. The demodulated signal is then saved to disk as a WAV audio file.

Signal Source: Here we have taken a cosine source with frequency 500 Hz.

OFDM Mod: Modulated the cosine source suing QAM64 modulation. We have set the cyclic prefix length at 128. We have also set the Occupied Tones at 400.

OFDM Demod: The modulated signal is demodulated by this block. The block parameters are the same as the modulators. The SNR is set at 10.

Wav File Sink: This file sink stores the incoming data to disk as a WAV audio file.

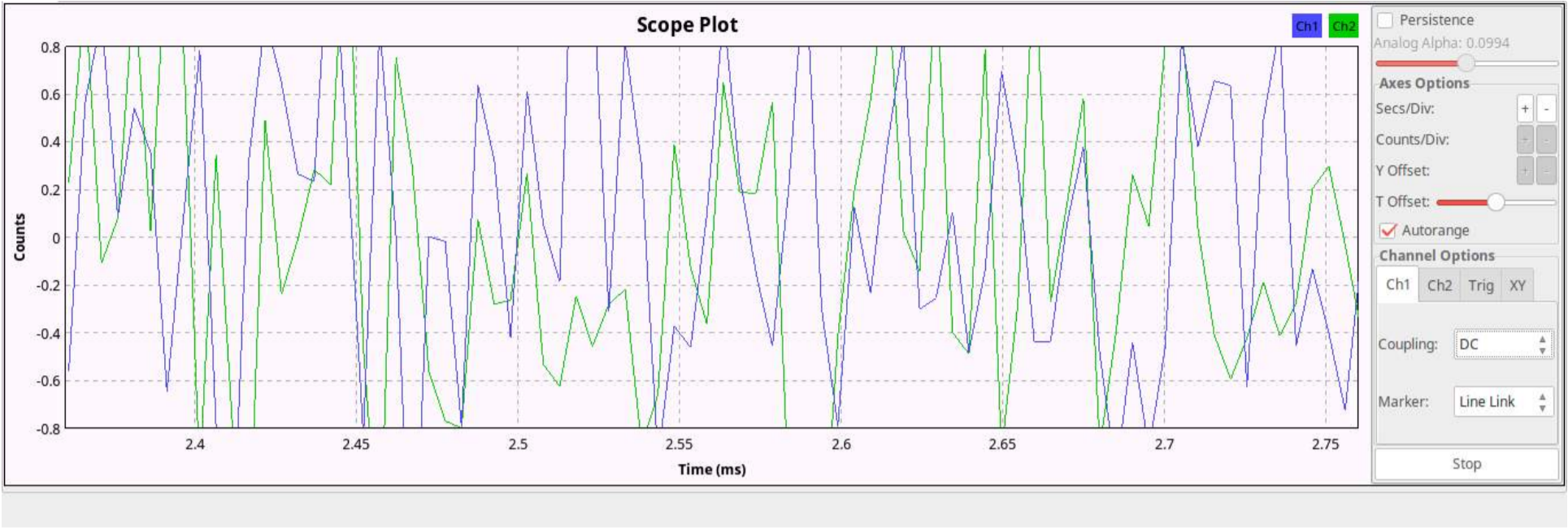


Fig 63: OFDM Sent Scope

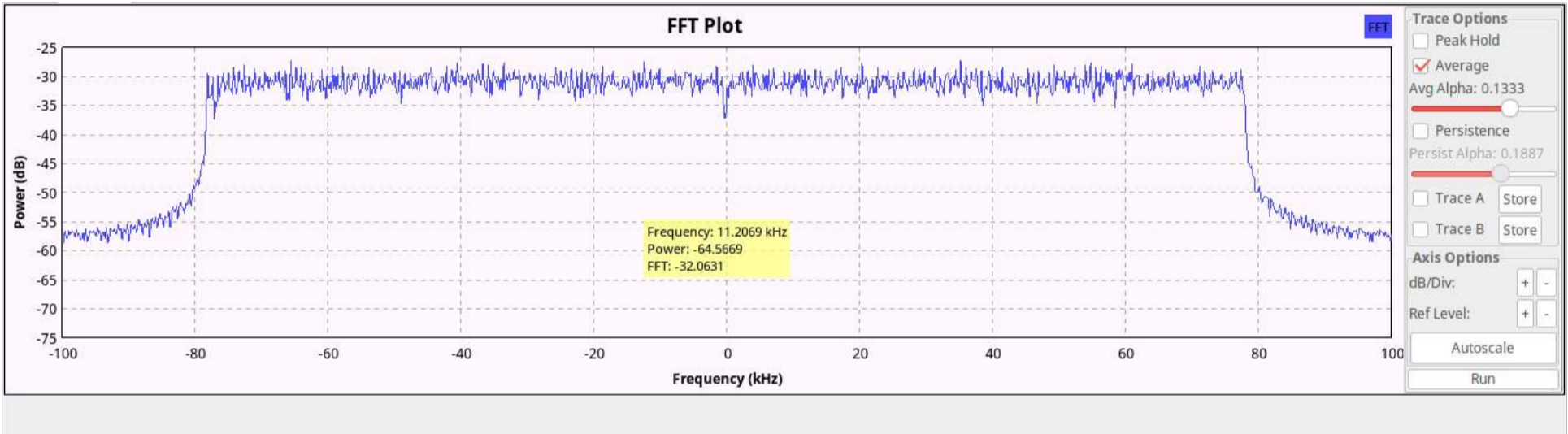


Fig 64: OFDM Send FFT

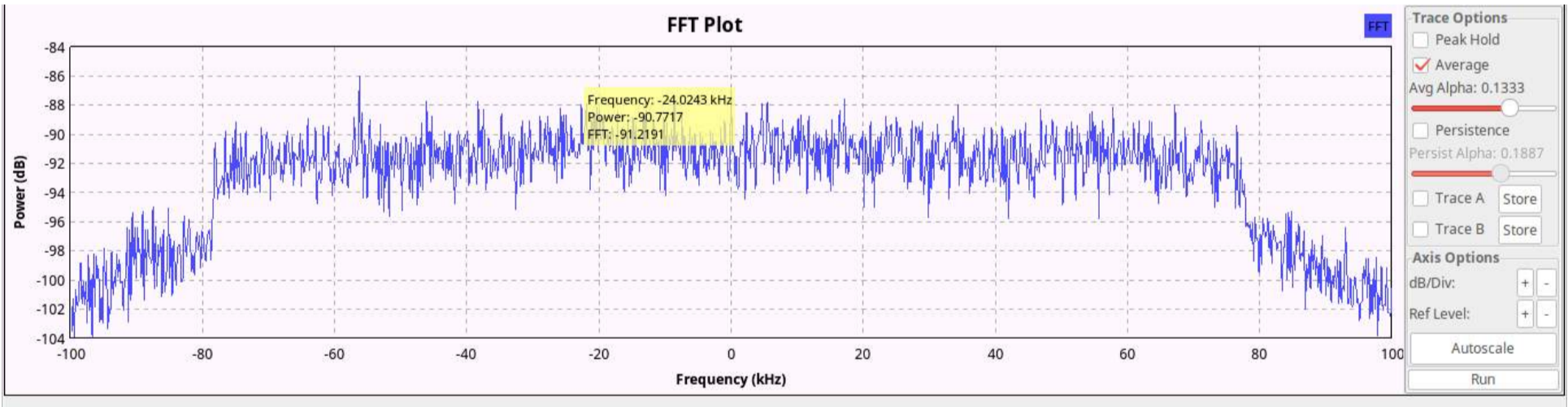


Fig 65: OFDM Received FFT

We have shown the FFT of the sent and received signal. Both the signals show that the PAPR is high, a characteristic disadvantage of OFDM. We will remedy this in the next experiment.

4.8 OFDM with convolutional coding and PAPR reduction

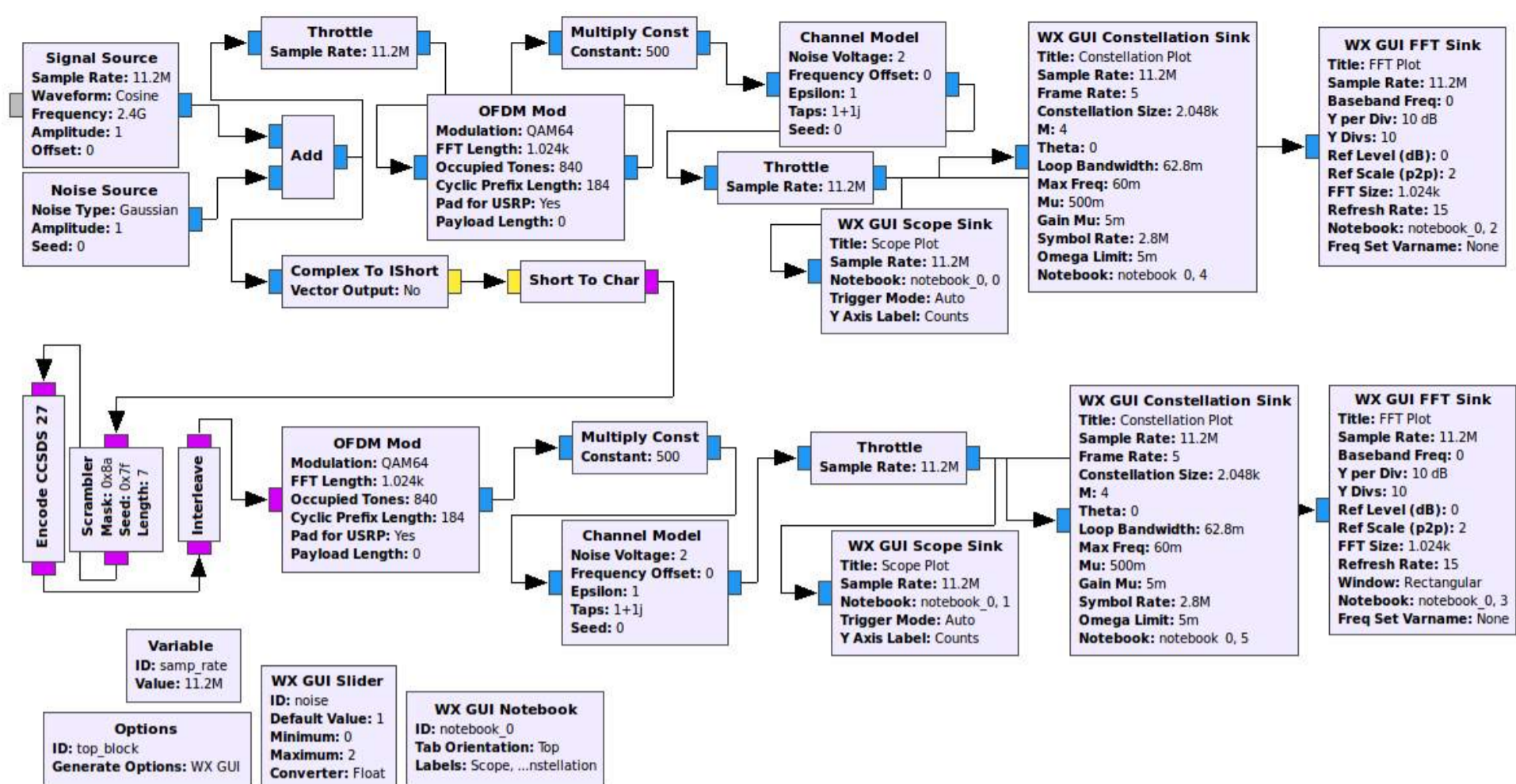


Fig 66: OFDM with convolutional coding and PAPR reduction

In this experiment we designed a flowgraph which takes a cosine signal, modulates it and is coded with convolutional coding. Both the non-coded and the coded signal's waveform and FFT are viewed.

Signal Source: A 2.4 GHz cosine wave is generated using this block.

Noise source: A Gaussian noise of amplitude 1 is generated.

Add: This block performs arithmetic addition of the cosine signal and the noise.

Throttle: Limits the computation of the program to 11.2 MHz.

OFDM Mod: Modulates the signal from the Add block.

Channel Model: The modulated signal is passed through an AWGN simulating channel.

The Blocks above form the original non-convolutional coded signal.

The Blocks below form the convolutional coded signal.

Scrambler: The noise added cosine signal from the Add block is sent to the Scrambler block. It is a form of LFSR. A scrambler is a function that is applied to the data sequence before transmitting with the goal of making this data more "random-like". For instance, scrambling avoids long runs of the bits 0 or 1 only, which may make the receiver loose synchronization or cause spectral concentration of the signal.

Encode CCSDS27: This block performs convolutional encoding using the CCSDS standard polynomial ("Voyager"). The input is an MSB first packed stream of bits. The output is a stream of symbols 0 or 1 representing the encoded data.

Interleave: This block interleaves blocks of samples. For each input connection, the samples are interleaved successively to the output connection. By default, the block interleaves a single sample from each input to the output unless block size is given in the constructor.



Fig 67: Oscilloscope plot

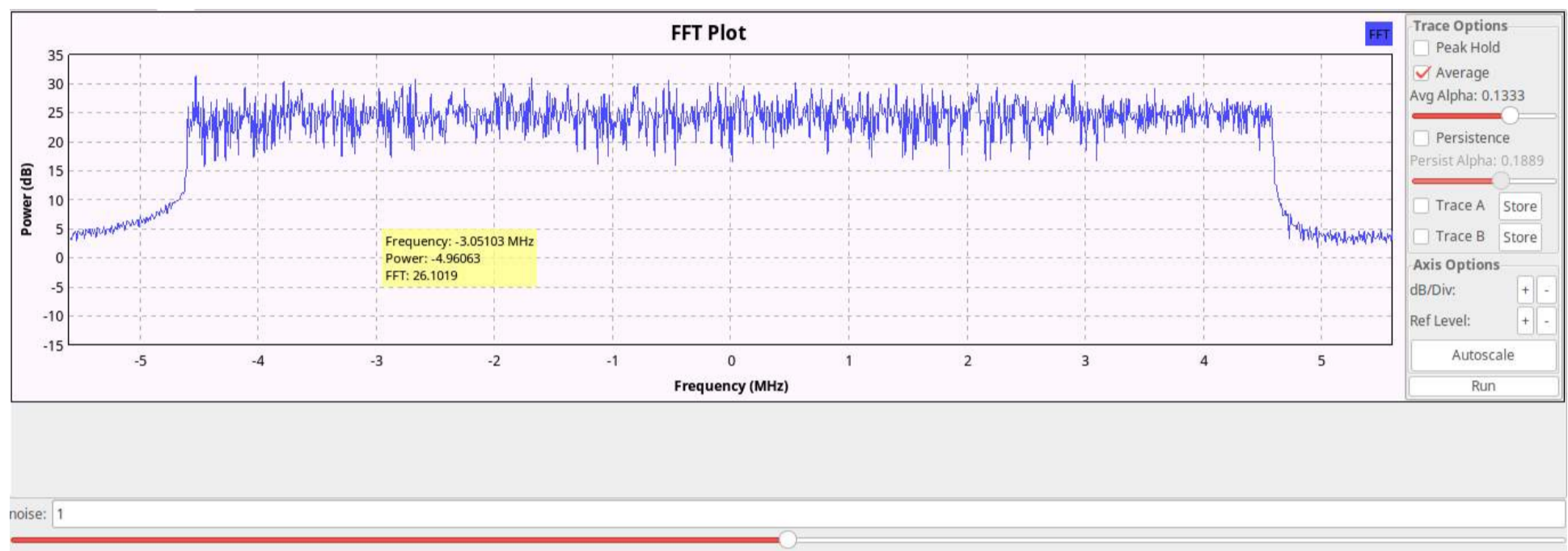


Fig 68: OFDM without convolutional coding

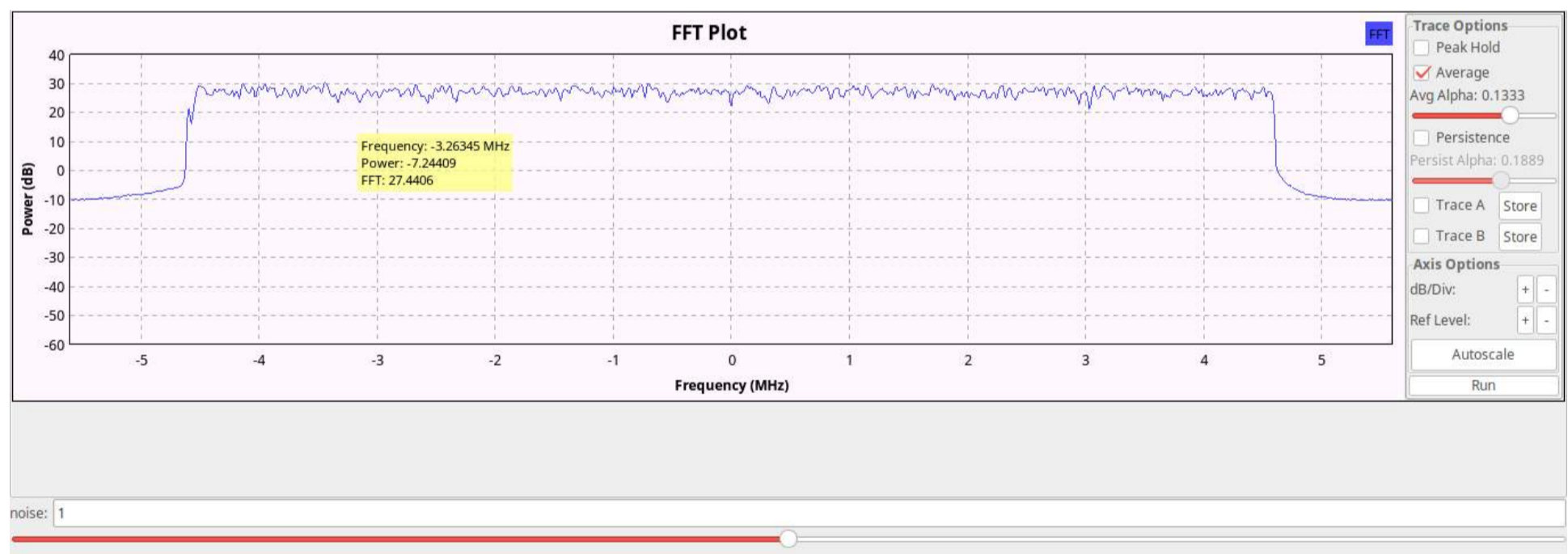


Fig 69: OFDM with convolutional coding and PAPR reduced

We have displayed the oscilloscope of the non-coded signal along with the FFT plot of the non-coded and the coded signal. Without convolutional coding we can see that the PAPR of the OFDM QAM64 modulated signal is very high, with many positive and negative peaks above the average curve. In the convolutional coded signal's FFT we can see the effects of coding very clearly. The overall shape of both the coded and non-coded curve remains the same. But the PAPR of the coded signal is very low and the overall modulated signal is very smooth. This shows that convolutional coding decreases PAPR drastically.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1 Conclusion

In this age we live in, rapid, reliable, robust and persistent communication is no longer a luxury but is now a necessity. No longer is telecommunication something for a select few but is a demand for everyone. To meet this technological demand we need newer, cheaper techniques and technology. And to tackle the problems present today and develop new schemes of telecommunication, Software Defined Radio is a must-have. The combination of the SDR platform, USRP N200 as the hardware and GNU Radio as the software, is a powerful and low cost combination to test and develop present and future schemes and applications.

We have through this research project, learned about the SDR platform. We have implemented a few of the uses of SDR which are summarized as follows: FM Receiver, FM Transmitter, QPSK modulator and Demodulator, 8PSK and 16PSK modulator and demodulator, Image and audio transmission via GMSK, OFDM modulator and demodulator and Convolutional coding of OFDM to reduce PAPR.

5.2 Future Work

More and more researchers at home and abroad are turning to SDR to test and develop their new applications and ideas. Though few problems are present with this platform, it is being rectified and improved upon rapidly. Some exciting and lucrative research areas for SDR includes:

- Cognitive Radio is an SDR which adapts to spectrum availability, protocols and has the capability to learn and react to the current needs of the user.
- The SDR will in future support the implementation of newer communication standards such as DVB-T, IEEE 802.11 a/g/p and LTE.
- The SDR platforms will support higher sampling rates and increased spectral efficiency and decreased BER using newer algorithms for channel coding.

REFERENCES

- [1] SDR forum website, <http://www.sdrforum.org>
- [2] <http://wb8nut.com/SDR/>
- [3] https://en.wikipedia.org/wiki/GNU_Radio
- [4] Marcos Majo Design and implementation of an OFDM-based communication system for the GNU Radio platform
- [5] https://en.wikipedia.org/wiki/Universal_Software_Radio_Peripheral
- [6] https://www.ettus.com/content/files/07495_Ettus_N200-210_DS_Flyer_HR_1.pdf
- [7] http://www.olifantasia.com/gnuradio/usrp/files/usrp_guide.pdf
- [8] <http://other.dreamrunner.org/2011/10/the-usrp-board/>
- [9] Matthias Föhnle Software-Defined Radio with GNU Radio and USRP/2 Hardware Frontend: Setup and FM/GSM Applications
- [10] <http://www.daenotes.com/electronics/communication-system/need-modulation>
- [11] <http://www.mmumullana.org/downloads/files/n5474524fb4134.pdf>
- [12] <http://www.radio-electronics.com/info/rf-technology-design/fm-frequency-modulation/what-is-fm-tutorial.php>
- [13] [https://en.wikipedia.org/wiki/Emphasis_\(telecommunications\)](https://en.wikipedia.org/wiki/Emphasis_(telecommunications))
- [14] https://en.wikipedia.org/wiki/Amplitude-shift_keying
- [15] http://www.eecs.umich.edu/courses/eecs455/eecs455_F_2004/lect11.pdf
- [16] https://www.sonoma.edu/users/f/farahman/sonoma/courses/ces540/lectures/PSK_Modulation_Dig.pdf
- [17] <http://www.radio-electronics.com/info/rf-technology-design/pm-phase-modulation/what-is-gmsk-gaussian-minimum-shift-keying-tutorial.php>
- [18] <https://www.nutaq.com/blog/understanding-papr-ofdm-systems>
- [19] Rinu Ann Baby Convolution coding and applications: A performance analysis under AWGN channel
- [20] https://en.wikipedia.org/wiki/Convolutional_code
- [21] <http://ajmalsi.blogspot.com/2015/01/understanding-ber-and-mer.html>

APPENDIX A

NI USRP N200 Datasheet



FEATURES:

- Use with GNU Radio, LabVIEW™ and Simulink™
- Modular Architecture: DC-6 GHz
- Dual 100 MS/s, 14-bit ADC
- Dual 400 MS/s, 16-bit DAC
- DDC/DUC with 25 mHz Resolution
- Up to 50 MS/s Gigabit Ethernet Streaming
- Fully-Coherent MIMO Capability
- Gigabit Ethernet Interface to Host
- 2 Gbps Expansion Interface
- Spartan 3A-DSP 1800 FPGA (N200)
- Spartan 3A-DSP 3400 FPGA (N210)
- 1 MB High-Speed SRAM
- Auxiliary Analog and Digital I/O
- 2.5 ppm TCXO Frequency Reference
- 0.01 ppm w/ GPSDO Option

N200/N210 PRODUCT OVERVIEW:

The Ettus Research™ USRP™ N200 and N210 are the highest performing class of hardware of the USRP™ (Universal Software Radio Peripheral) family of products, which enables engineers to rapidly design and implement powerful, flexible software radio systems. The N200 and N210 hardware is ideally suited for applications requiring high RF performance and great bandwidth. Such applications include physical layer prototyping, dynamic spectrum access and cognitive radio, spectrum monitoring, record and playback, and even networked sensor deployment.

The Networked Series products offers MIMO capability with high bandwidth and dynamic range. The Gigabit Ethernet interface serves as the connection between the N200/N210 and the host computer. This enables the user to realize 50 MS/s of real-time bandwidth in the receive and transmit directions, simultaneously (full duplex).

The Networked Series MIMO connection is located on the front panel of each unit. Two Networked Series units may be connected to realize a complete 2x2 MIMO configuration using the optional MIMO cable. External PPS and reference inputs can also be used to create larger multi-channel systems. The N200 and N210 are largely the same, except that the N210 features a larger FPGA for customers that intend to integrate custom FPGA functionality.

The USRP Hardware Driver™ is the official driver for all Ettus Research products. The USRP Hardware Driver supports Linux, Mac OSX, Windows.

APPENDIX B

Installing GNU Radio Companion (GRC)

Before installing GRC one needs to install a Linux OS. The Operating System of choice for us was Linux Mint. There is ample tutorial in the World Wide Web which shows how to install Linux Mint. We are giving a trusted source below:

<https://itsfoss.com/guide-install-linux-mint-16-dual-boot-windows/>

After installing Linux Mint one can install GRC using the steps listed below:

1. Open the Terminal window using menu or press Ctrl+Alt+T.
2. Copy the following command and paste in the terminal:

sudo apt update && apt upgrade && apt dist-upgrade

3. Follow the instruction given in the screen.

After the update is done type:

4. exit
5. mkdir ~/gnuradio
6. cd ~/gnuradio
7. wget http://www.sbrac.org/files/build-gnuradio && chmod a+x ./build-gnuradio && ./build-gnuradio -verbose

When the following message is prompted type the command in bold:

8. Proceed? **Y**
9. Sudo privileges? **Y**
10. After that it will automatically install the dependencies as well as the software packages from a live server. Just make sure the process is fully complete with no errors. Make sure there is continuous internet and power connection in your computer. It will take at least 4 hours to install. The installation will be complete gradually. If any error occurs just run the command starting from step 7 again.

Share this: