Audio File Transmission using GNU RADIO and USRP

EEC 687 - Mobile Computing (Fall 2016)

Final Project Report

Raag Kharadi	CSUID: 2644321
Harshil Mehta	CSUID: 2643874

Electrical and Computer Engineering, Cleveland State University

Introduction

In radio engineering radio engineers were facing problem related to chip computing power so they had to think again to redesign of radio receivers. Since, from many decades' hardware for radio transceiver consist of signal were down converted to intermediate frequency then filtered and then demodulated after converting to baseband signal. After few years around 1980's digital signal processor leads to development of digital transceivers. This kind of digital transceivers requires hardware for just for limiting frequency to narrow band frequency and then ADC and other portion of architecture consist of software interface so this kind of transceivers leads to so-called Software Define Radios(SDR's). In SDR there is general purpose processors and other devices which perform signal processing are integrated in FPGA.

Software Defined Radio

In SDR there are certain stages in receiving and transmission chain where signal is digitized and using software techniques computation is done on digital radio signal. The main aim of this SDR is to convert almost all hardware system problem into digital domain problems so it can be easily modified and problems can be solved easily. In general, SDR consist of Antennas, an ADC and subsystem defined in software domain.

Thus, there are certain conditions to be followed to implement software define radios which are as follows:

1. Antennas which are used for any specific system should be capable for all radio signal of interest that to be operated.
2. So design for ADC and DAS would be such that its sampling rate must be grated than twice the frequency than that for signal of interest.
3. Thus, the unit which does task for processing should be capable of enough processing power to process signal of interest.

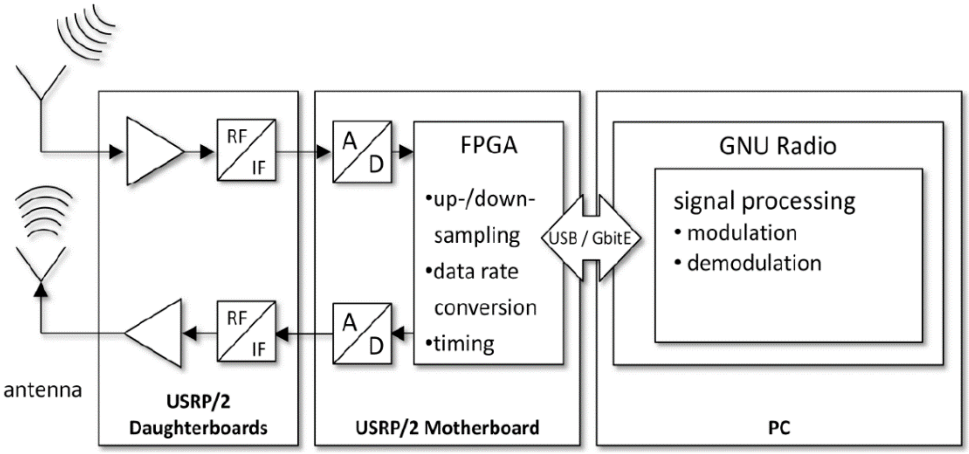


Figure 1. Software Defined Radio Block Diagram

Universal Software Radio Peripheral (USRP)

The USRP is developed by Ettus Research LLC. who are involved in developing of SDR platform form general purpose computers. The USRP we use here knows as USRP 1 in which motherboard consist of four ADCs/DACs which are capable for fast and effective performance these are connected to radio front end daughterboard. There is programmable FPGA Chip on motherboard which is connected to PC through USB. Figure 2 show USRP's motherboard and daughterboard which are used in our project also there are other list of daughterboard that are used for different frequency band.

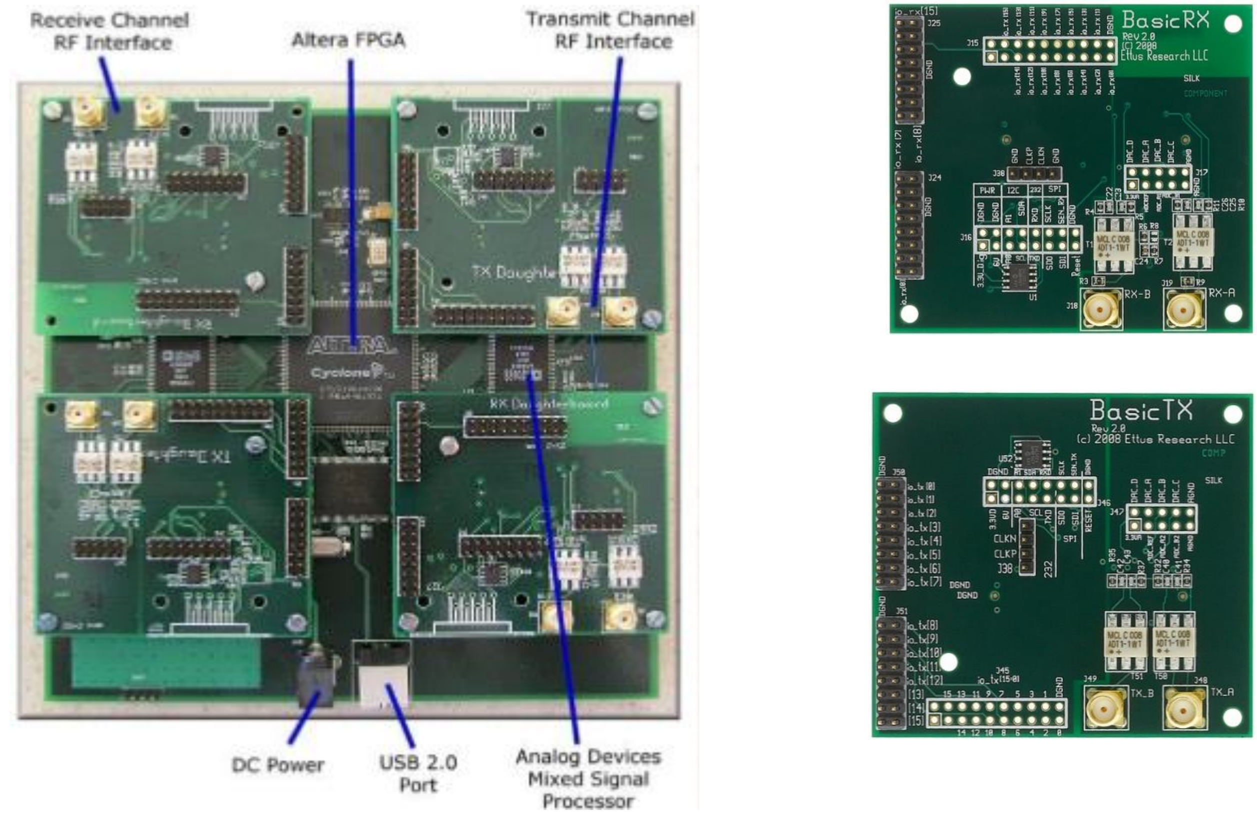

Receive Channel RF Interface
Altera FPGA
Transmit Channel RF Interface
DC Power
USB 2.0 Port
Analog Devices Mixed Signal Processor
BasicRX
Rev 2.0
(c) 2008
Ettus Research LLC
RX-B
RX-A
BasicTX
Rev 2.0
(c) 2008 Ettus Research LLC
TX_B
TX_A

Figure 2. USRP Main Board and Basic TX and Basic RX Daughterboard 1-250 MHz Tx/Rx

Types of Daughterboard.

Following are list of Daughterboard with its band range and cost

1. Basic TX and Basic RX Daughterboard 1-250 MHz Tx/Rx - $140
2. LFTX -- DC-30 MHz Transmitter and Receiver - $154
3. TVRX2 50-860 MHz Rx x 2 - $205
4. DBSRX -- 800 MHz to 2.4 GHz Receiver - $155
5. RFX400 -- 400-500 MHz Transceiver - $275
6. RFX900 -- 800-1000MHz Transceiver - $275
7. RFX1200 -- 1150 MHz - 1450 MHz Transceiver - $275
8. RFX1800 -- 1.5-2.1 GHz Transceiver - $275
9. RFX2400 -- 2.3-2.9 GHz Transceiver, 20+mW output - $ 275

GNU Radio

GNU radio provide software environment for developing and which is open source and free of cost software and also has inbuilt signal processing blocks for implementation of software radios. For creating SDR it provides less cost RF hardware and it also provides stimulation like user interface which does not required physical hardware. It is widely used in real radio systems and for research related work for wireless communication.

Implementation

For implementation in GNU Radio their certain blocks which are pre-defined in gr library, this blocks are been created in C++ as low level programing while python is used to build graphs as high level programming and as interface between them (Simplified Wrapper and Implemented Generator) SWIG is used.

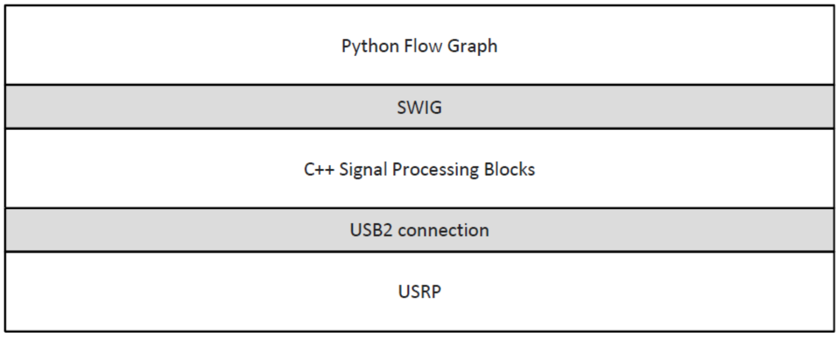


Figure 3. GNU radio framework structure

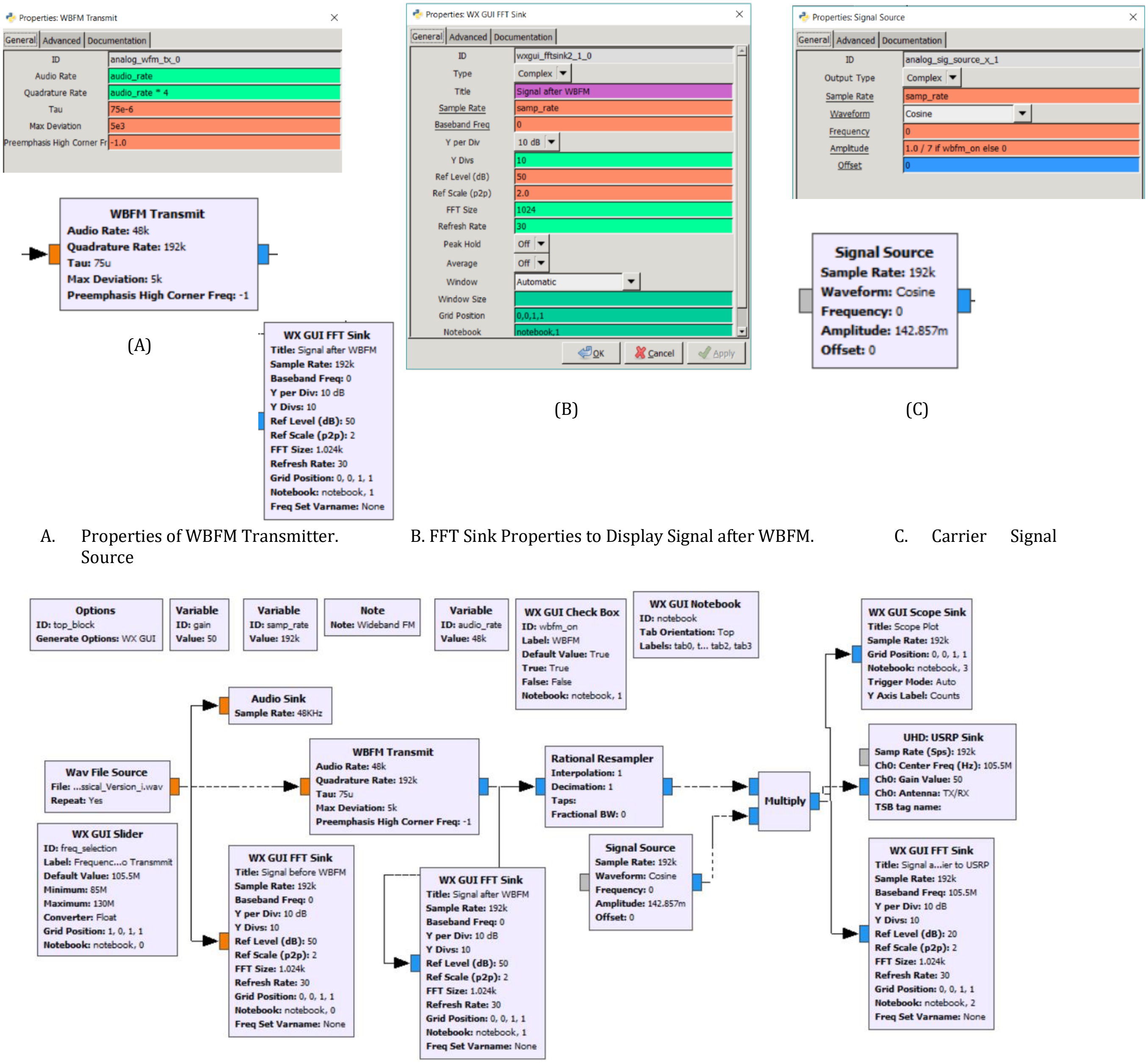
1. Practical Implementation using WBFM Modulation/Demodulation Scheme

FM Transmitter

In practical implementation the modulation technique used here is wideband frequency modulation. When the highest frequency component of a modulating signal is greater than the peak frequency deviation of modulated signal, then it is called as wide band frequency modulation.

Following are detail description for block for building WBFM

Figure 3a WBFM Transmit: The message signal is then passed through WBFM block which takes the input signal and gives the frequency modulated signal as an output. Audio rate and quadrature rate are the sampling rate of input and output signal. You can give any value to Max Deviation as long as it is greater than 20kHz (highest frequency component of a (sound) message signal). Also Figure 3b shows FFT sink properties of after WBFM block to see exactly how is signal look after Modulation.



Properties: WBFM Transmit

General | Advanced | Documentation

ID: analog_wfm_tx_0
Audio Rate: audio_rate
Quadrature Rate: audio_rate * 4
Tau: 75e-6
Max Deviation: 5e3
Preemphasis High Corner Fr: -1.0

WBFM Transmit
Audio Rate: 48k
Quadrature Rate: 192k
Tau: 75u
Max Deviation: 5k
Preemphasis High Corner Freq: -1

(A)

WX GUI FFT Sink
Title: Signal after WBFM
Sample Rate: 192k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Grid Position: 0, 0, 1, 1
Notebook: notebook, 1
Freq Set Varname: None

Properties: WX GUI FFT Sink

General | Advanced | Documentation

ID: wxgui_fftsink2_1_0
Type: Complex
Title: Signal after WBFM
Sample Rate: samp_rate
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2.0
FFT Size: 1024
Refresh Rate: 30
Peak Hold: Off
Average: Off
Window: Automatic
Window Size:
Grid Position: 0,0,1,1
Notebook: notebook,1

OK | Cancel | Apply

(B)

Properties: Signal Source

General | Advanced | Documentation

ID: analog_sig_source_x_1
Output Type: Complex
Sample Rate: samp_rate
Waveform: Cosine
Frequency: 0
Amplitude: 1.0 / 7 if wbfm_on else 0
Offset: 0

Signal Source
Sample Rate: 192k
Waveform: Cosine
Frequency: 0
Amplitude: 142.857m
Offset: 0

(C)

A. Properties of WBFM Transmitter. B. FFT Sink Properties to Display Signal after WBFM. C. Carrier Signal Source

Options
ID: top_block
Generate Options: WX GUI

Variable
ID: gain
Value: 50

Variable
ID: samp_rate
Value: 192k

Note
Note: Wideband FM

Variable
ID: audio_rate
Value: 48k

WX GUI Check Box
ID: wbfm_on
Label: WBFM
Default Value: True
True: True
False: False
Notebook: notebook, 1

WX GUI Notebook
ID: notebook
Tab Orientation: Top
Labels: tab0, t... tab2, tab3

Audio Sink
Sample Rate: 48KHz

Wav File Source
File: ...ssical_Version_i.wav
Repeat: Yes

WX GUI Slider
ID: freq_selection
Label: Frequenc...o Transmmit
Default Value: 105.5M
Minimum: 85M
Maximum: 130M
Converter: Float
Grid Position: 1, 0, 1, 1
Notebook: notebook, 0

WBFM Transmit
Audio Rate: 48k
Quadrature Rate: 192k
Tau: 75u
Max Deviation: 5k
Preemphasis High Corner Freq: -1

WX GUI FFT Sink
Title: Signal before WBFM
Sample Rate: 192k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Grid Position: 0, 0, 1, 1
Notebook: notebook, 0
Freq Set Varname: None

Rational Resampler
Interpolation: 1
Decimation: 1
Taps:
Fractional BW: 0

WX GUI FFT Sink
Title: Signal after WBFM
Sample Rate: 192k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 50
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Grid Position: 0, 0, 1, 1
Notebook: notebook, 1
Freq Set Varname: None

Signal Source
Sample Rate: 192k
Waveform: Cosine
Frequency: 0
Amplitude: 142.857m
Offset: 0

Multiply

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 192k
Grid Position: 0, 0, 1, 1
Notebook: notebook, 3
Trigger Mode: Auto
Y Axis Label: Counts

UHD: USRP Sink
Samp Rate (Sps): 192k
Ch0: Center Freq (Hz): 105.5M
Ch0: Gain Value: 50
Ch0: Antenna: TX/RX
TSB tag name:

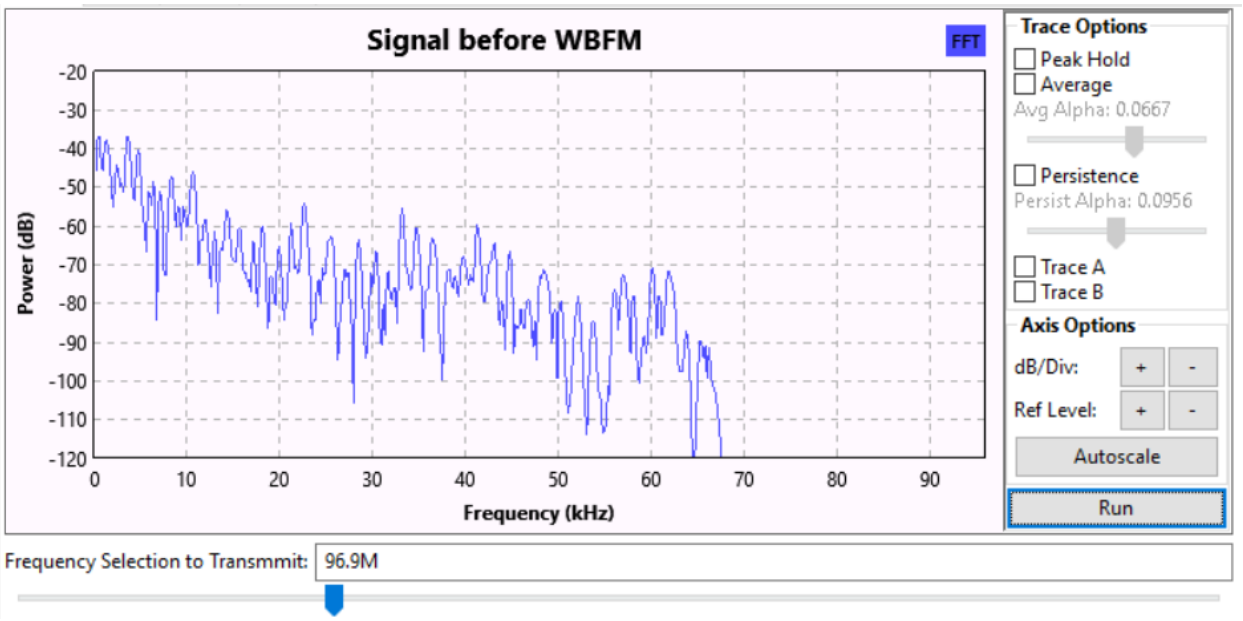
WX GUI FFT Sink
Title: Signal a...ier to USRP
Sample Rate: 192k
Baseband Freq: 105.5M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 20
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 30
Grid Position: 0, 0, 1, 1
Notebook: notebook, 2
Freq Set Varname: None

D. Flow Graph of WBFM Transmitter

Figure 4. WBFM Transmitter

Figure-4d shows the detail flowchart of the FM transmitter. In the implementation of the FM transmitter, we sampled the audio at 48 kHz, the rate at which it was stored, then converted it to 192 kHz using WBFM. After that it sent to the Rational Resampler for some interpolation and decimation. Then we multiply the cosine signal to reduce the effect of noise at high frequencies, and passed it to the USRP. The USRP up converted the FM signal to 96.9MHz and transmitted it.

1. Here our main aim is to transmit some audio file to the destination so we are going to use a file source (here wav file source) and not the USRP. In the Wav file source we have to choose the input file source where we store our audio file.
2. Figure-5 shows the graph for our Wav file before the WBFM block.

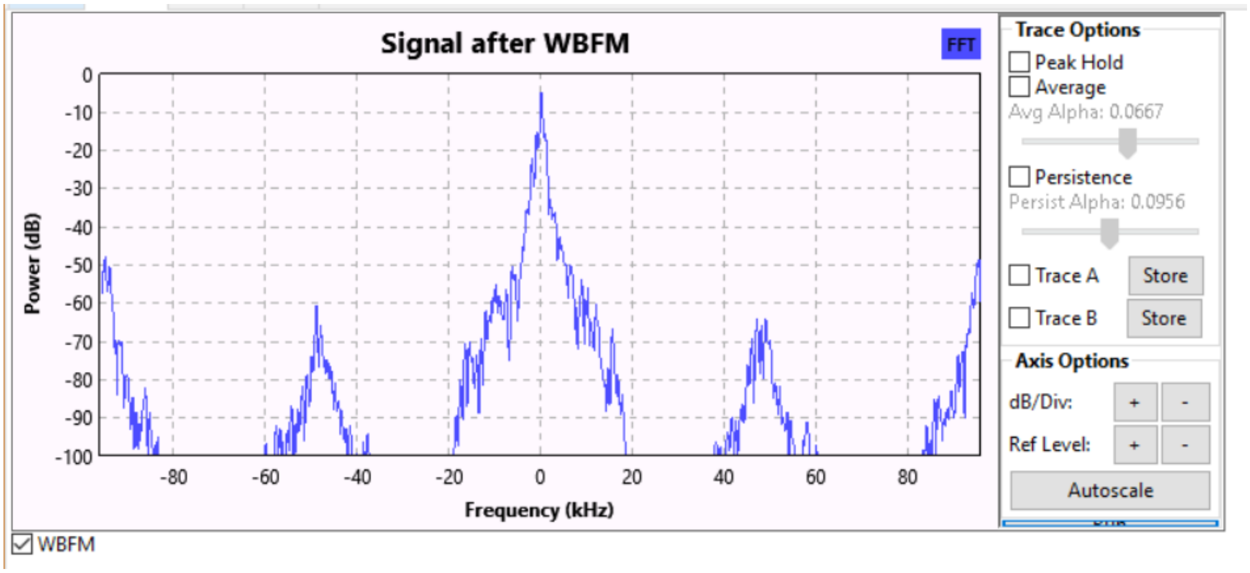


Frequency Selection to Transmmit: 96.9M

Figure 5. Original Wav file signal

3. Then this input file is sent the WBFM block where audio rate is predefined. Here quadrature rate is set in the multiple of 48 and this rate is the outgoing sample rate whereas 48kHz is incoming sample rate. The modulated signal is shown in Figure-6 below.

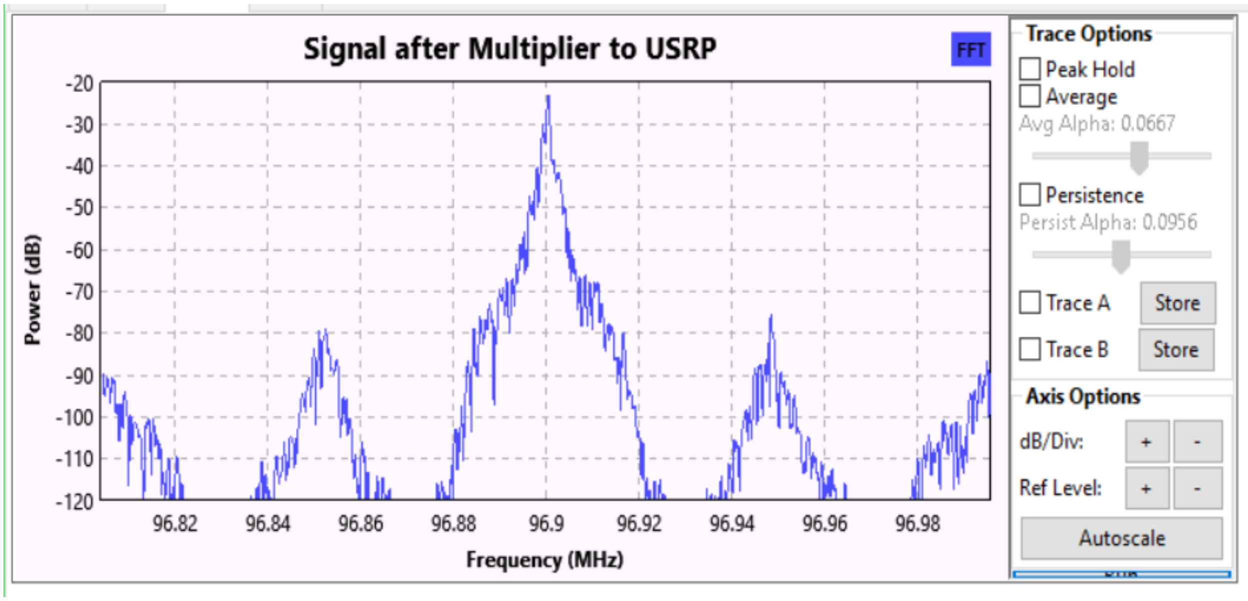
    Audio rate: 48 kHz
    Quadrature Rate: 48 kHz * 4 = 192 kHz
    Tua: 75u



☑ WBFM

Figure 6. Signal after modulation

4. After that this sampled signal is sent to the rational Resampler. Where we are doing some frequency conversion. For example, the Audio Sink needs lower frequency and USRP block need higher then this block converts the frequency by doing some interpolation or decimation as per the requirements.
5. Now output of the rational resampler is feed to one of the input of the Multiply block. Second input of the multiply block is Cosine signal with the 192 kHz sample rate. Here this carrier signal is used for proper transmission.
6. Figure-7 shows resulting signal after multiplier.



□ Peak Hold
□ Average
Avg Alpha: 0.0667
□ Persistence
Persist Alpha: 0.0956
□ Trace A
□ Trace B
dB/Div:
Ref Level:

Figure 7. Multiplication of carrier signal and modulated signal

7. Now these multiplied signal is sent to the USRP sink from there signal is send to the destination using wireless medium. Here we use 96.9 MHz as center frequency but it can change using freq_selection slider and antenna is connected to TX/RX daughterboard.

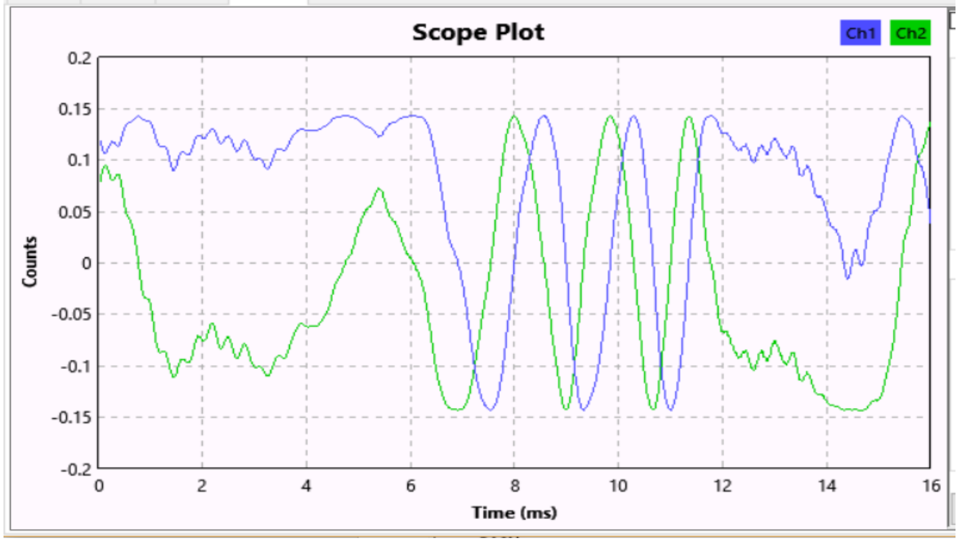
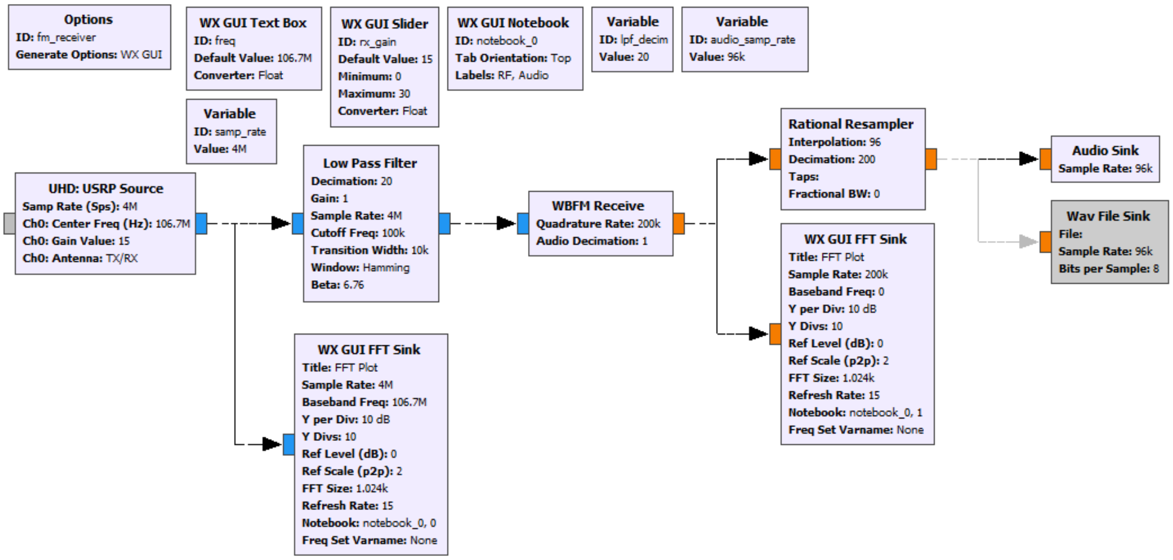


Figure 8. Multiplication of carrier signal and modulated signal in time domain

FM Receiver

1. Here we are trying to receive FM channels using USRP that is why our input source is going to be USRP block with centre frequency of 106.7MHz which can be adjusted with the help of Text Box. Here sampling rate is 4MHz and other properties are as below:
   Gain Value: It is managed by rf_gain slider with min=0 and max=30.
   Antenna: Connected to TX/RX in daughter board



Options
ID: fm_receiver
Generate Options: WX GUI

WX GUI Text Box
ID: freq
Default Value: 106.7M
Converter: Float

WX GUI Slider
ID: rx_gain
Default Value: 15
Minimum: 0
Maximum: 30
Converter: Float

WX GUI Notebook
ID: notebook_0
Tab Orientation: Top
Labels: RF, Audio

Variable
ID: lpf_decim
Value: 20

Variable
ID: audio_samp_rate
Value: 96k

Variable
ID: samp_rate
Value: 4M

UHD: USRP Source
Samp Rate (Sps): 4M
Ch0: Center Freq (Hz): 106.7M
Ch0: Gain Value: 15
Ch0: Antenna: TX/RX

Low Pass Filter
Decimation: 20
Gain: 1
Sample Rate: 4M
Cutoff Freq: 100k
Transition Width: 10k
Window: Hamming
Beta: 6.76

WBFM Receive
Quadrature Rate: 200k
Audio Decimation: 1

Rational Resampler
Interpolation: 96
Decimation: 200
Taps:
Fractional BW: 0

Audio Sink
Sample Rate: 96k

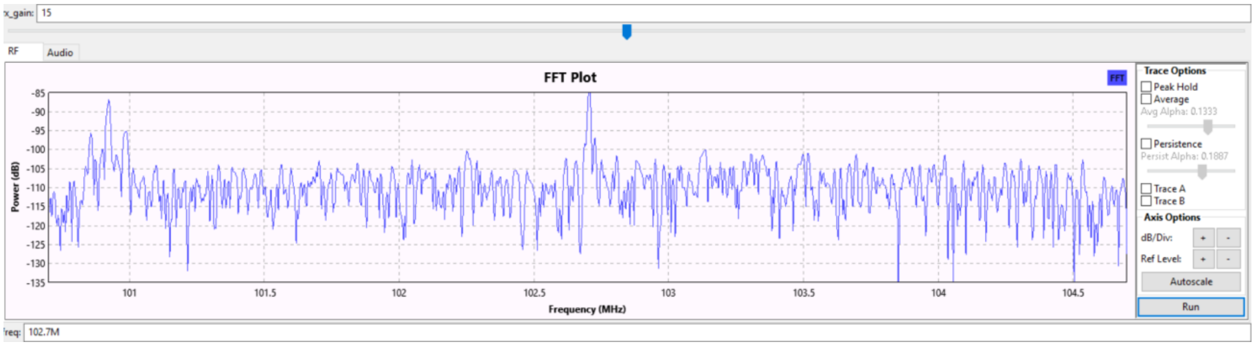
Wav File Sink
File:
Sample Rate: 96k
Bits per Sample: 8

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 200k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Notebook: notebook_0, 1
Freq Set Varname: None

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 4M
Baseband Freq: 106.7M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 1.024k
Refresh Rate: 15
Notebook: notebook_0, 0
Freq Set Varname: None

Figure 9. Modified FM Receiver

2. Now the output of the USRP block is going to the low pass filter. Here filter passes the signal whose frequency is below the cutoff frequency (here it is 100kHz) and also it deduces the signal with higher frequency.
3. Figure 10 shows the graph of the received signal at the input of USRP.



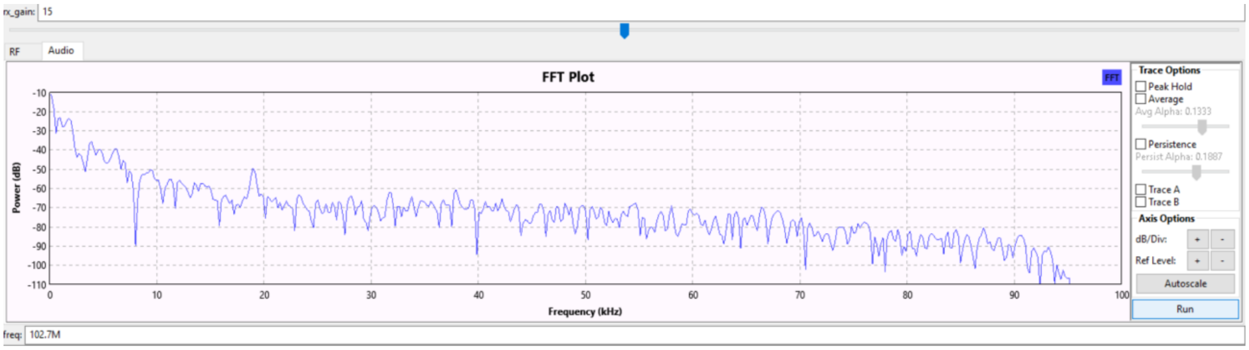
x_gain: 15

☐ Peak Hold
☐ Average
Avg Alpha: 0.1333
☐ Persistence
Persist Alpha: 0.1887
☐ Trace A
☐ Trace B

freq: 102.7M

Figure 10. Resulting Graph of Input Signal

4. Now filtered signal is given to the wide band FM block. This block demodulates the wide band frequency signal from data stream. So at the output we get the original data stream which sent by the sender.
5. Output of the WBFM block sent to the Rational Resampler. Here we decimate or interpolate the data stream according to the application to the desire rate. For example, the Audio Sink needs lower frequency and USRP block need higher then this block converts the frequency by doing some interpolation or decimation as per the requirements.



rx_gain: 15

freq: 102.7M

Figure 11. Resulting Graph of Low Passed Signal

6. Figure 11 is the graph of demodulated data stream in frequency domain.
7. Finally, Audio sink block is used to listen the sound of the received channel.

Combined Implementation

Figure 12 below shows the combine receiver and transmitter flow graph where we tune into our FM channel using GUI Text Box which is used to receive the FM channels. The USRP block takes this frequency as its center Frequency and shifts it to origin to make it a baseband signal. Thus changing the center frequency of the USRP, we can tune to various channels. In this implementation we use basic Rx so it can work only in certain frequency range i.e. 1-250MHz. Also the antenna has to be in the frequency range in which you want to receive the signal, so in the beginning, reception was not at all audible. After successfully receive the FM channel it is feedback to transmitter portion. Now with the use of this transmitter we transmit that channel to the another frequency (here it is 106.9 MHz). So, in general we receive an audio from one channel then play from laptop's soundcard and broadcast it from transmitter and listen to another channel using mobile phone.

1. Here we are trying to receive FM channels using USRP that is why our input source is going to be USRP block with center frequency of 91.1 MHz which can be adjusted with the help of Text Box. Here sampling rate is 4MHz and other properties are as below:
   Gain Value: It is managed by rf_gain slider with min=0 and max=30.
   Antenna: Connected to TX/RX in daughter board

2. Now the output of the USRP block is going to the low pass filter. Here filter passes the signal whose frequency is below the cutoff frequency (here it is 100kHz) and also it deduces the signal with higher frequency.

3. Now filtered signal is given to the wide band FM block. This block demodulates the wide band frequency signal from data stream. So at the output we get the original data stream which sent by the sender.

4. Output of the WBFM block sent to the Rational Resampler. Here we decimate or interpolate the data stream according to the application to the desire rate. For example, the Audio Sink needs lower frequency and USRP block need higher then this block converts the frequency by doing some interpolation or decimation as per the requirements.

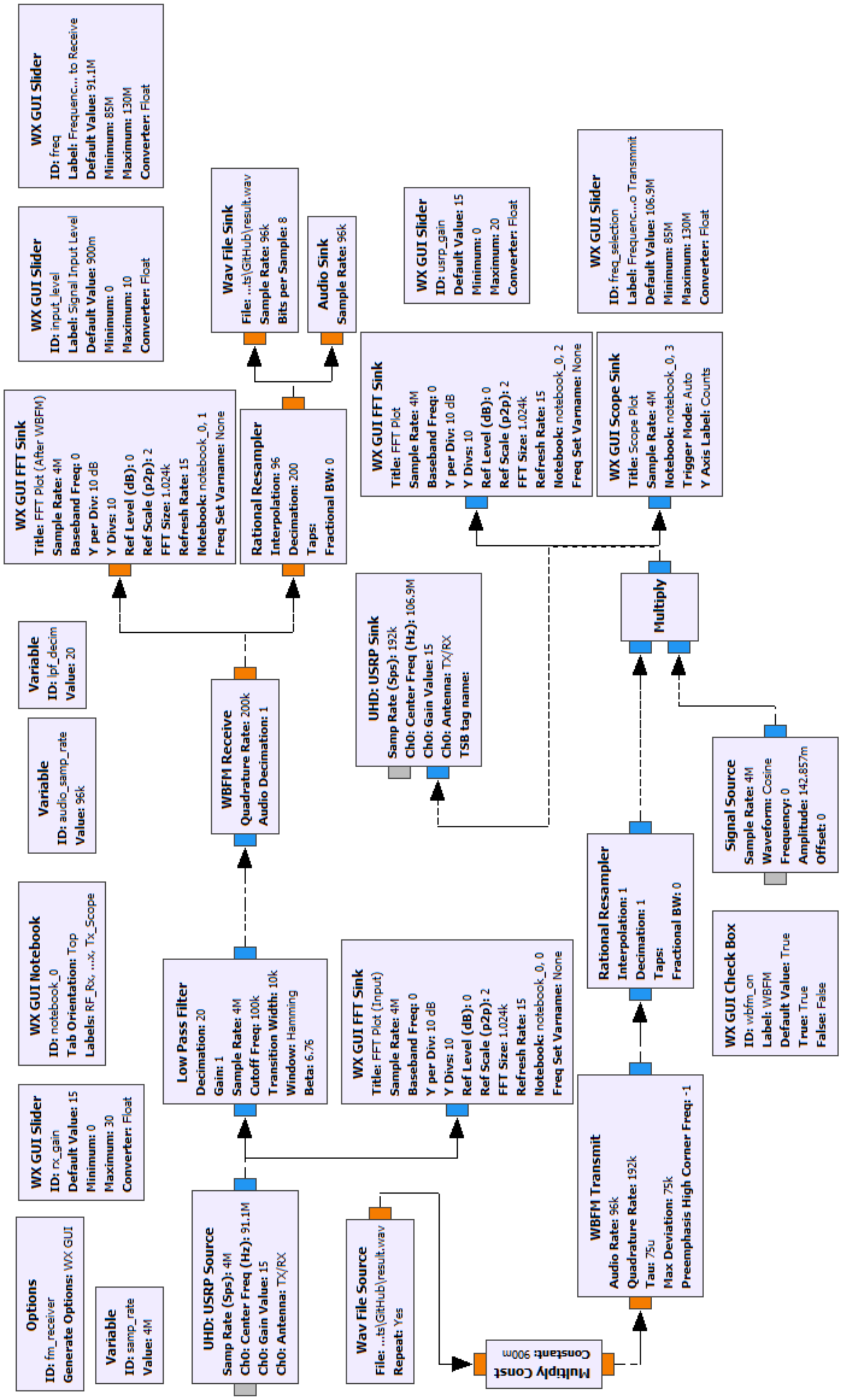


Figure 12. Architectural Flow Graph of FM Receiver and Transmitter using one USRP

5. Finally, Audio sink block is used to listen the sound of the received channel. Now as per our flow graph we are going to feed this back to transmitter for broadcasting. That's why it is used as input to the WBFM block.

6. Here in WBFM block quadrature rate is set in the multiple of 96 (i.e. 192 kHz) and this rate is the outgoing sample rate whereas 96kHz is incoming sample rate.
   Audio rate: 96 kHz
   Quadrature Rate: 96 kHz * 2 = 192 kHz
   Tua: 75u

7. After that this sampled signal is sent to the rational Resampler. Where we are doing some frequency conversion again.

8. Now output of the rational resampler is feed to one of the input of the Multiply block. Second input of the multiply block is Cosine signal with the 192 kHz sample rate. Here this carrier signal is used for proper transmission.

9. Now these multiplied signal is sent to the USRP sink from there signal is send to the destination using wireless medium. Here we use 106.9 MHz as center frequency but it can change and antenna is connected to TX/RX on daughterboard.

2. Implementation using GMSK Modulation/Demodulation Scheme

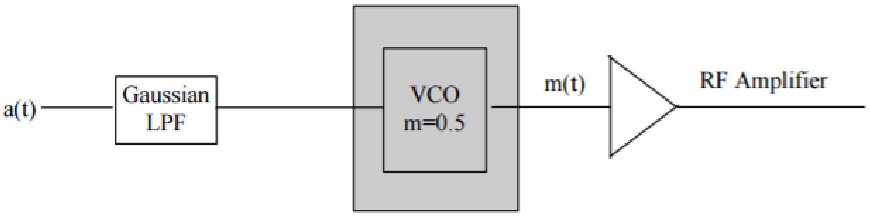
What is GMSK?

Gaussian Minimum Shift Keying (GMSK)

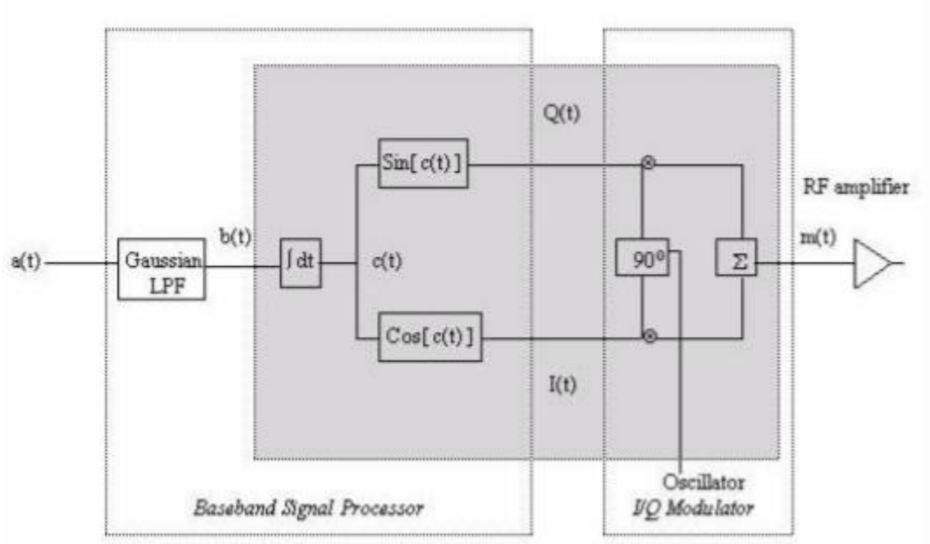
When we use half sinusoidal pulse instead of rectangular pulse then it creates the MSK (Minimum Shift Keying) signal. But when we use Gaussian Pulse shape rather than rectangular pulse then that type of modulation scheme is known as Gaussian Minimum Shift Keying (GMSK). Many studies states that GMSK digital modulation technique is revised adaptation of Minimum Shift Keying (MSK) in terms of bandwidth and spectral efficiency. In addition to that, GMSK has comparatively curtail side lobe level than the rectangular pulse and its main lobe is also confined than the QPSK and MSK. We can consider GMSK not only frequency modulation scheme but also we can view GMSK as phase modulation. Although changing of phase is restricted because of Gaussian response but advance or delay of 90˚ over bit period is not limited so beat shaping depends on Bandwidth Time Product (BT). Here in GMSK system bandwidth is totally depends on multiplication of filter bandwidth (before modulation) B and bit period T. Because of narrow bandwidth, GMSK suffers from Inter Symbol Interference (ISI) which decrease the signal power.

GMSK Modulation

Frequency Shift Keyed Modulation and Quadrature Phase shift keyed Modulation are the two main methods for generating the GMSK.



(A) GMSK using Frequency Shift Keying Modulation



(B) GMSK using Quadrature Phase Shift Keying Modulation

Figure (A) is simple technique but it is not suitable for demodulation because the modulation index is keep changing over time and temperature. So we have to use a method which keeps the modulation index to 0.5 (i.e. m=0.5). That is why implementation of Figure (B) is beneficial because it maintains the index m to 0.5. So, to understand the process of modulation let's consider the following input data stream {1,1, -1, -1, 1, 1, -1, -1, -1,1, -1, -1, 1,1, -1, -1,1, -1, 1,1, -1,1, -1, -1....}. The graphical form of this data stream is given in Figure 13.

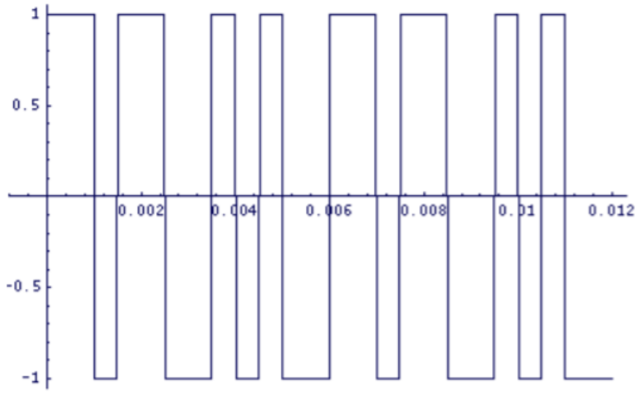


Figure 13. Input Data Stream

As we all know that in GMSK data is first pass through the Gaussian filter before going for modulation and there they are properly shaped using Gaussian response pulse but due to narrower bandwidth ISI is brought in because many bits entering the filter simultaneously. Since the data rate is 2T, the 3rd bit enter the filter when 2nd is only at half way, the fifth enters as third leaves and this process go on. The Figure 14 below shows the output of the Gaussian filter (Gaussian shaped pulses).

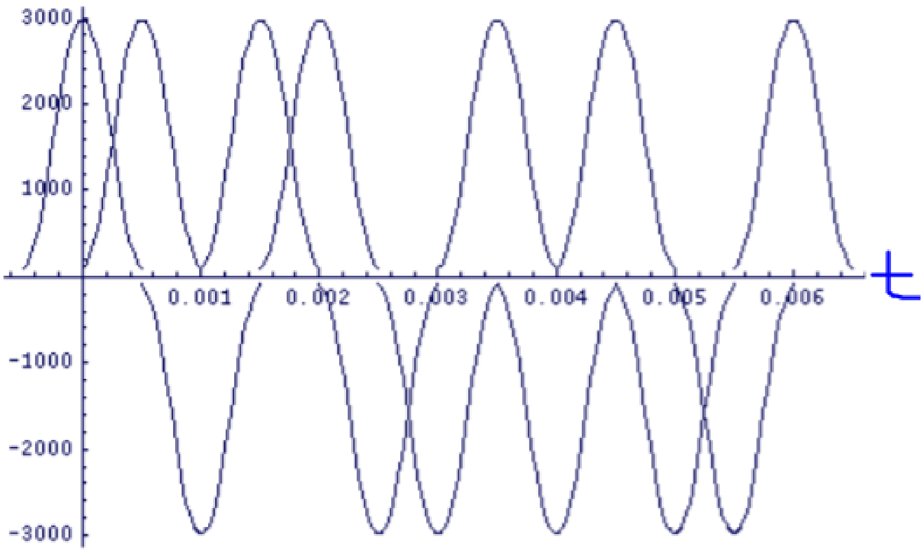


Figure 14. Gaussian Shaped Pulses

Now when we add these Gaussian Shaped Pulses together we get the following signal. Let denote this signal as b(t). After getting this b(t) we have to integrate this waveform with respect to time which is known as c(t) and this waveform is used to by forget the real part and imaginary part of the input data stream.

Once getting the c(t), we pass this signal from Sine and Cosine function which generates In-phase baseband signal I (real) and quadrature baseband signal Q (Imaginary).

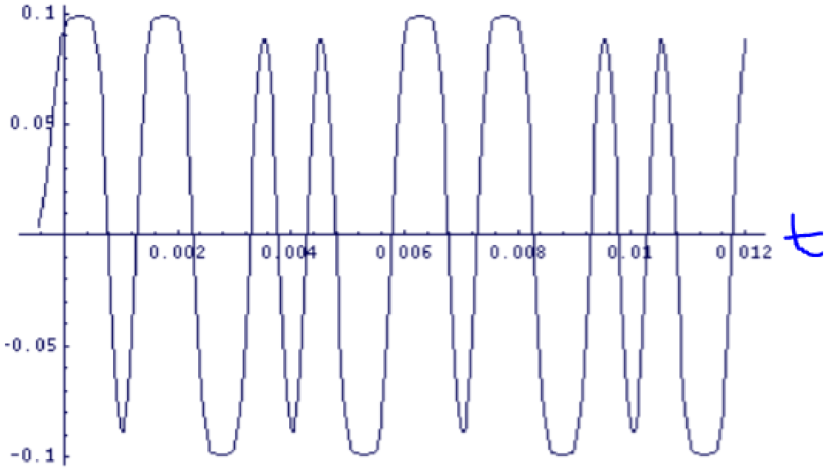


Figure 15. Summation of Shaped Pulses b(t)

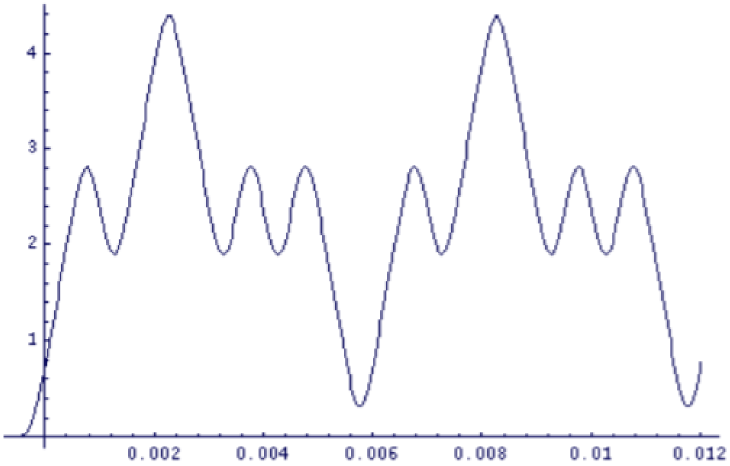
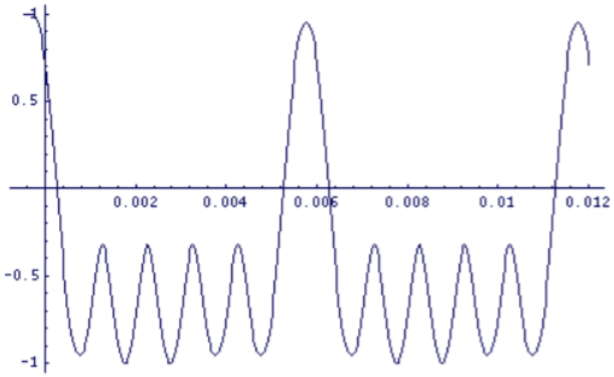


Figure 16. c(t) = Integration of signal b(t)

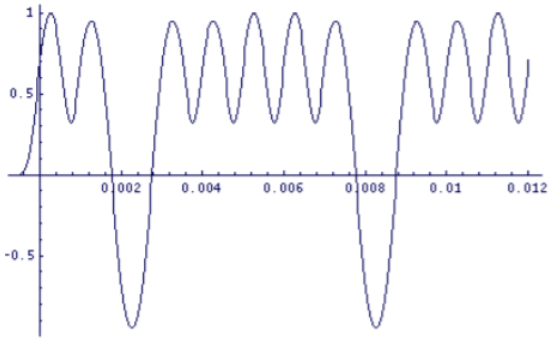
After that when we pass this c(t) from Cosine function it will produce In-phase signal I whereas passing through Sine function it will produce Quadrature Signal Q. i.e.

$$I(t)=Cos\{c(t)\}$$

$$Q(t)=Sin\{c(t)\}$$



I(t) baseband signal



Q(t) baseband signal

Now, our I/Q modulator plays major role in this modulation process. So, I(t) and Q(t) passed through this I/Q modulator which produce our output signal m(t).

$$M(t) = Sin\{(2*fc) * t\}\ I(t) + Cos\{(2*fc) * t\}\ Q(t)$$

Where f_c=carrier frequency

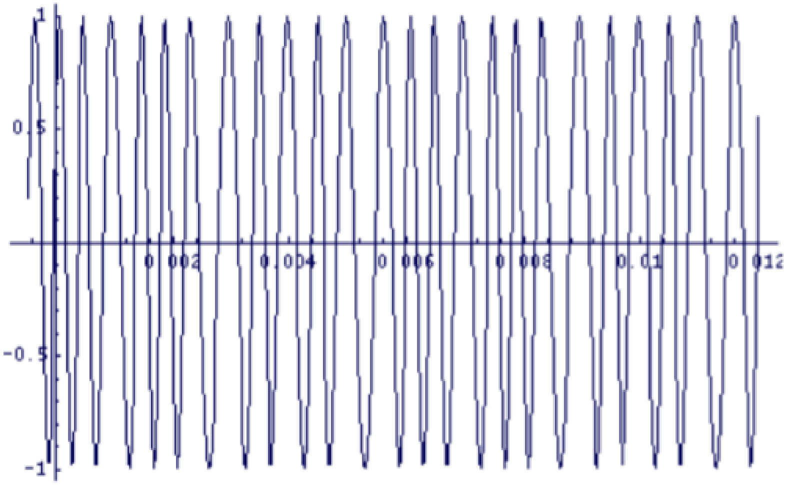
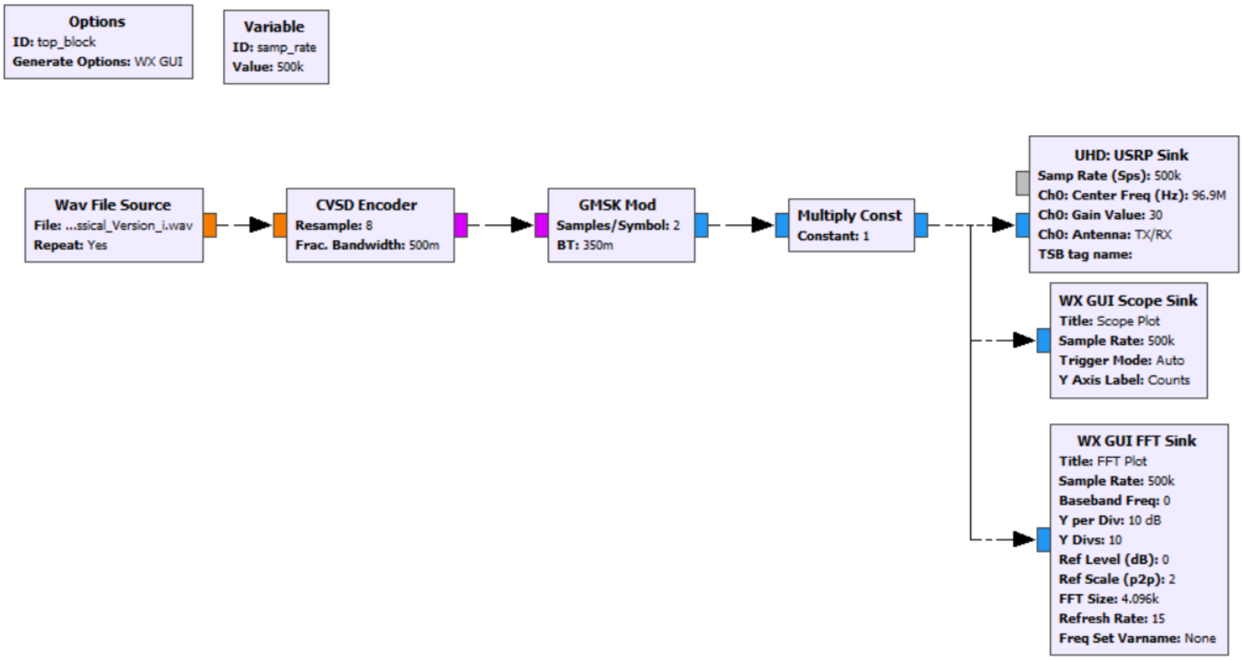


Figure 17: Resulting Modulating Signal m(t)

FM Transmitter



Options
ID: top_block
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 500k

Wav File Source
File: ...ssical_Version_i.wav
Repeat: Yes

CVSD Encoder
Resample: 8
Frac. Bandwidth: 500m

GMSK Mod
Samples/Symbol: 2
BT: 350m

Multiply Const
Constant: 1

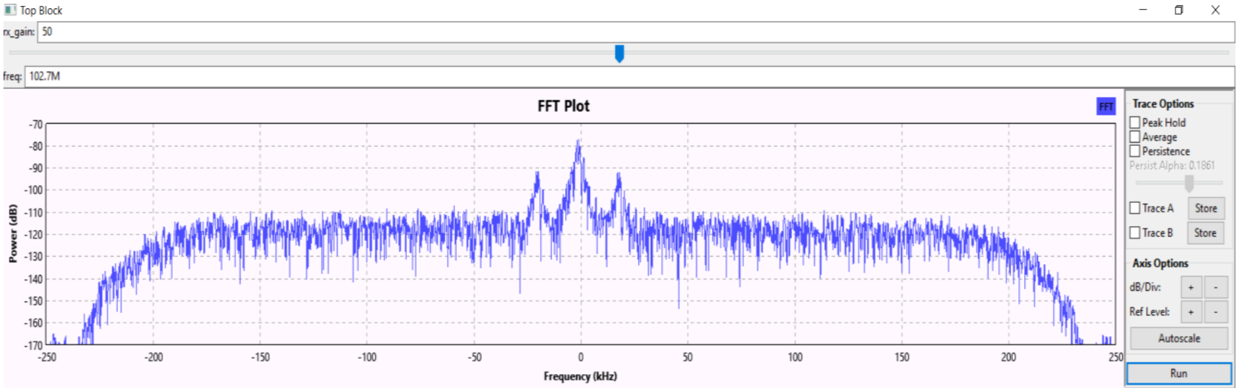
UHD: USRP Sink
Samp Rate (Sps): 500k
Ch0: Center Freq (Hz): 96.9M
Ch0: Gain Value: 30
Ch0: Antenna: TX/RX
TSB tag name:

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 500k
Trigger Mode: Auto
Y Axis Label: Counts

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 500k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 4.096k
Refresh Rate: 15
Freq Set Varname: None

Figure 18. FM Transmitter

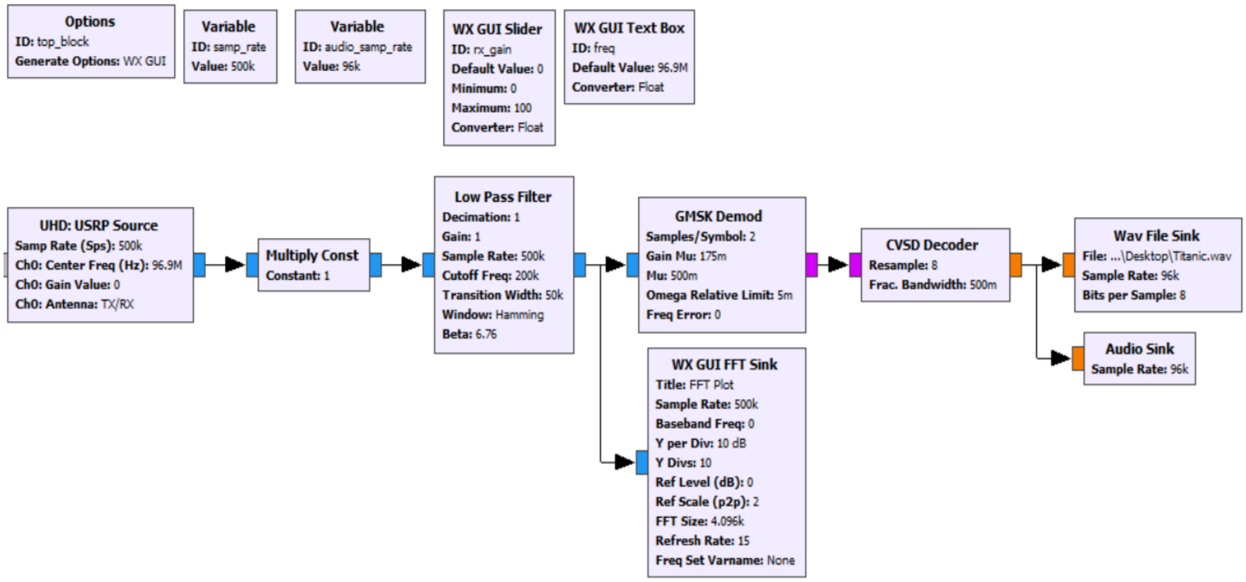
Resulting Graph


Top Block

rx_gain: 50

freq: 102.7M

FM Receiver


Options
ID: top_block
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 500k

Variable
ID: audio_samp_rate
Value: 96k

WX GUI Slider
ID: rx_gain
Default Value: 0
Minimum: 0
Maximum: 100
Converter: Float

WX GUI Text Box
ID: freq
Default Value: 96.9M
Converter: Float

UHD: USRP Source
Samp Rate (Sps): 500k
Ch0: Center Freq (Hz): 96.9M
Ch0: Gain Value: 0
Ch0: Antenna: TX/RX

Multiply Const
Constant: 1

Low Pass Filter
Decimation: 1
Gain: 1
Sample Rate: 500k
Cutoff Freq: 200k
Transition Width: 50k
Window: Hamming
Beta: 6.76

GMSK Demod
Samples/Symbol: 2
Gain Mu: 175m
Mu: 500m
Omega Relative Limit: 5m
Freq Error: 0

WX GUI FFT Sink
Title: FFT Plot
Sample Rate: 500k
Baseband Freq: 0
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 4.096k
Refresh Rate: 15
Freq Set Varname: None

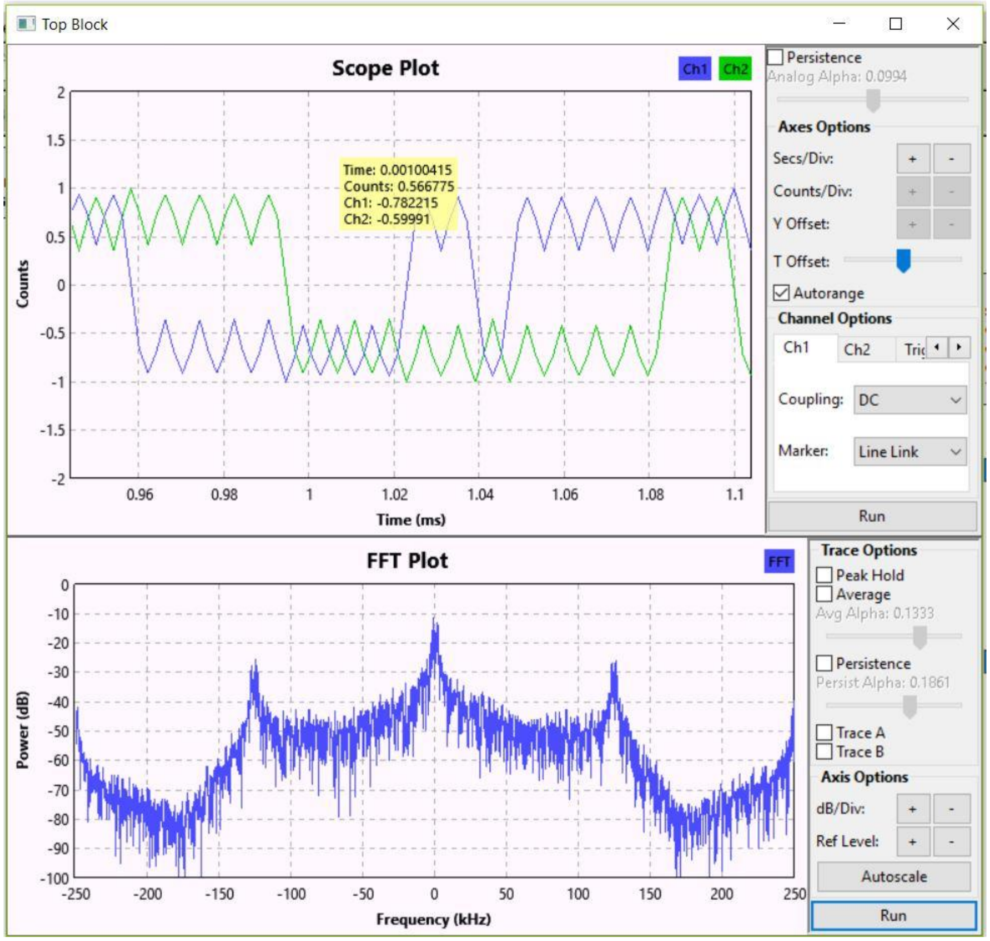
CVSD Decoder
Resample: 8
Frac. Bandwidth: 500m

Wav File Sink
File: ...\Desktop\Titanic.wav
Sample Rate: 96k
Bits per Sample: 8

Audio Sink
Sample Rate: 96k

Figure 19. FM Receiver

Resulting Graph



Time: 0.00100415
Counts: 0.566775
Ch1: -0.782215
Ch2: -0.59991

☐ Persistence
Analog Alpha: 0.0994
☑ Autorange
Coupling: DC
Marker: Line Link

☐ Peak Hold
☐ Average
Avg Alpha: 0.1333
☐ Persistence
Persist Alpha: 0.1861
☐ Trace A
☐ Trace B

Conclusion

In this project, we properly understand the concept of Software Radio and GNURadio with the help of USRP as hardware tool and Python as software tool. After that we simulated a communication system with GNU radio software with USRPs and we successfully transmit the audio file from source to destination. In addition, we also implement the same communication system using different modulation scheme such WBFM and GMSK and in both cases we successfully get desire result.

Reference

[1] E. Blossom, "GNURadio: Tools for Exploring the Radio Frequency Spectrum" Linux Journal, 2004, http://www.linuxjournal.com/article/7319

[2] Ettus Research LLC., http://www.ettus.com

[3] Firas Abbas Hamza, "The USRP under 1.5X Magnifying Lens", http://gnuradio.org/trac /attachment /wiki/UsrpFAQ/USRP Documentation.pdf

[4] GNUradio Official website, http://gnuradio.org/trac

[5] Danilo Valerio “Open Source Software-Defined Radio: A survey on GNU radio and its applications” ftw. Forschungszentrum Telekommunikation Wien Donaucitystrasse 1, 1220 Vienna, AUSTRIA

[6] Matthias Fähnle “Software-Defined Radio with GNU Radio and USRP/2 Hardware Frontend: Setup and FM/GSM Applications “

[7] http://www.emc.york.ac.uk/reports/linkpcp/appD.pdf

[8] http://techniajournal.com/attachments/article/128/Analog%20&%20digital%20modulation.Pdf