Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 05, May-2016

Implementation of Wide Band FM Receiver on RTL-SDR

Jahnavi. S, Pooja C. B, Santro Leeona FM
Dept. of Electronics and Communications Engineering
Dr.T.Thimmaiah institute of Technology
Kolar Gold Fields-563117, Karnataka, India

Asst. Prof. Vijayageetha. R
Dept. of Electronics and Communications Engineering
Dr.T.Thimmaiah institute of Technology
Kolar Gold Fields-563117, Karnataka, India

***Abstract*— Signal Processing is found to be a very important area in communication field. Signal processing concepts in real time can be easily understood and demonstrated using Software Defined Radio. This paper focuses on GNU Radio, an open source software acting as a simulation tool to drive the receiver hardware. Here a wideband FM receiver is implemented with RTL-SDR on/using GNU Radio. A frequency of 92.7 MHz signal is received from the receiver is processed with a python script on GNU Radio. Other radio frequency signals can also be received by this receiver. This proposed method of using RTL-SDR for reception of signals is more significant and cheap compared to other hardware. RTL-SDR, python and GNU Radio combine together into low cost experimental kit with real radio signals for various sources.**

Keywords*— *Wideband FM, Software Defined Radio, RTL-SDR, GNU Radio, python Script, SWIG (Simplified Wrapper and Interface Generator).

I. INTRODUCTION

Communication most of the time requires much cabling and wiring, just like MOD-BUS, PROFI-BUS, CAN-BUS, Ethernet etc; which requires large cost for both installation and management. To overcome this we go for wireless communication like Wireless Ethernet etc which eliminates the cabling issue and decreases the cost of maintance. Once the decision between wired or wireless communication is made the decision making does not stop here, the decision to choose among various wireless technologies is a big decision. One among those wireless technologies is a Software Defined Radio (SDR). Over 93% of mobile infrastructure market utilizes SDR technology, and further growth to support mobile data demand will simply drive more SDR base stations. The key motivation behind this project is that with the use of SDR technology the circuit size can be reduced with less complexity and better SNR. The adaption of wireless communication also has few disadvantages like:

a) Commercial wireless network standards are continuously evolving from 2G to 2.5G/3G, 4G and so on. Each generation networks differ significantly in link layer protocol standards causing problem to subscribers.
b) Migration from one network generation to the other is costlier.
c) The Air-interface and link-layer protocol differ across geographies [1] i.e., the European wireless networks are TDMA based while the USA standards are CDMA based. This creates a problem for the subscriber during the deployment of global facilities.
d) Deployment issues occur not only during roaming but also during rolling-out new features.

SDR technology overcomes all these problems by enabling implementation of radio functions in networking infrastructure as software modules which runs on a generic hardware platform. This eases migration from one generation to other as well as during roaming.

In 2001, compared to earlier radios SDR became major change as the functionality was predominantly realized in software. During that time the SDR's hardware was predominantly computers and data buses rather than IF/RF. In 2005, a brief study of digital quadrature transformation for Software Defined Radio (SDR) systems was done and proposed two generalizes SDR receiver schemes. Among those two one can reduce AD sampling speed by 2 times and the other lowers both output data rate and AD sampling speed. In 2012, there was evolution of modern radio communication applications using SDR technology such as Radar, electronic warfare and signal intelligence. This made possible to implement SDR transceiver using Simulink, MATLAB and Xilinx. In 2013, wireless communication became the hottest area and SDR is revolutionizing it. Many open sources like USRP (Universal Software Radio Peripheral) and GNU Radio were used commonly to experiment on SDR. In 2015, GNU Radio was mainly focused and used as simulation tool for driving the SDR transceiver hardware.

This paper is grouped as follows: Section (2) having brief description of Software Defined Radio, followed by RTL-SDR in section (3), GNU Radio in section (4), Block diagram and its implementation in section (5) results in section (6) and finally conclusion in section (7).

II. SOFTWARE DEFINED RADIO

A Radio system in which all the signal processing or all physical layer functions are implemented using software is nothing but a Software Defined Radio (SDR) [2]. If the function of any physical layer has to be modified then the hardware has to be redesigned which is costlier, so we define the functions in software which are flexible and reconfigurable [3] [4]. The evolution of SDR goes from military to civilian environments. Speakeasy [5] was the first operational SDR that was developed by United States' Navy between 1991 and 1995. Both SDR software [6] and hardware [7] are available at low prices.

A. SDR Transmitter

Most commonly SDR devices are used for reception purpose but it can also include transmission schemes.

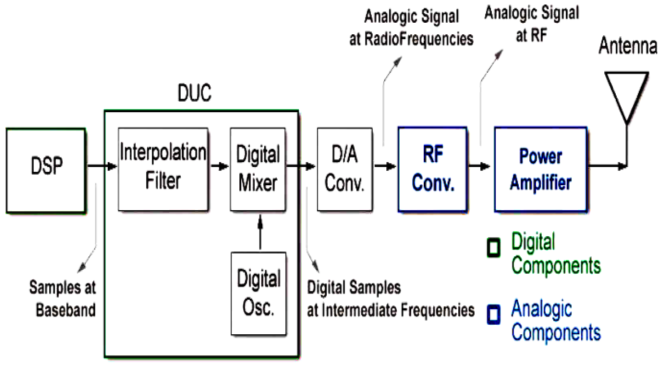


Fig.1. Block Diagram of SDR Transmitter

The input to the SDR transmitter is the baseband signal generated by a DSP block as given in the fig.1. DUC (Digital up Converter) is the first block which transfers baseband signals to IF followed by DAC that transforms the samples to analog domain. Now the signal is shifted towards higher frequency by RF converter. Finally the signal is amplified and given to the antenna for transmission. The interpolation filter in DUC is responsible for raising the baseband signal sampling rate to match the operating frequency. Finally the IF samples are shifted by digital mixer and local oscillator.

B. SDR Receiver

The SDR Receiver consists of the following blocks shown in the fig.2. Firstly, the signals captured by the receiving antenna is tuned and converted from analog signal to IF in RF tuner.

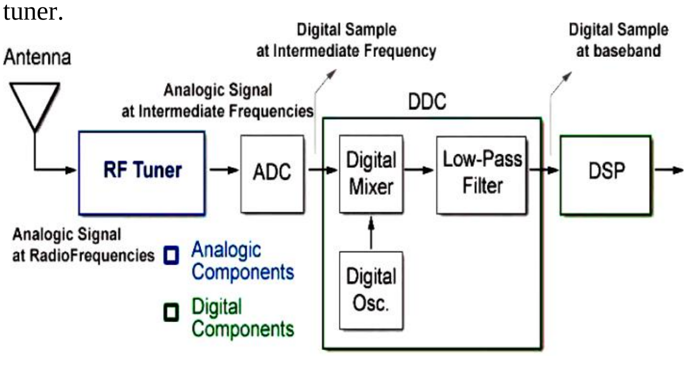


Fig.2. Block Diagram of SDR Receiver

Now these converted IF signal passes through the ADC converter which converts analog signals to digital samples. These digital samples are fed to DDC (Digital Down Converter) commonly known as monolithic chip. This DDC has three main functional blocks (i) a digital mixer, (ii) a digital local oscillator and (iii) a FIR low pass filter. The IF digital samples are shifted to baseband signal in digital mixer and local oscillator. The filter limits the bandwidth [8]. Decimation process can be performed to decrease the sampling rate. However, the new sampling rate can be twice the highest frequency [9] popularly known by Nyquist theorem. By various practical analysis the reduction in sampling rate can be done up to an extra 20%, which does not affect the quality of the result and given by:

$$f_{b2} = 0.8fb = fs/N \tag{1}$$

Where fb is the baseband frequency, fs is the sampling frequency, N is the decimator factor and f_{b2} is new calculated baseband frequency after decimation.

Lastly the baseband signals are passed through the DSP in which the demodulation and decoding process is done.

C. SDR Development and Development Tools

The initial waveform development tool was GNU Radio since it was open source in nature, very friendly to use. Also, GNU Radio Companion a graphical interface is very intuitive to use and provides readily available digital signal processing blocks.

The most relevant SDR dongles available in market is given below:

Table I: Most relevant SDR devices in market

Commercial Name	Min Freq (MHz)	Max freq (MHz)	Band width	Resolution of ADC	Transmit ?	Price in $
RTL-SDR	24	1766	3,2	8	No	10-20
Funcube Pro	64	1700	0,96	16	No	150
Funcube Pro+	410	2050	0,192	16	No	200
HackRF	30	600	20	8	Yes	300
BladeRF	300	3800	40	12	Yes	400-650
USRP 1	10	6000	64	12	Yes	700
MatchStiq	300	3800	28	12	Yes	4500

III RTL-SDR

A very low cost easy to use USB device which receives RF radio signals. These were actually designed to be a DVB-T (Digital Video Broadcast-Terrestrial) receiver, but later it was found that it could be used as generic SDRs simply by putting those USBs in different modes. These SDRs are capable of receiving any RF signal tuned by the tuner. The range of frequency that can be received by the USB differs from device to device depending upon the components used. But the common frequency range is from 25MHz to 1.75GHz.

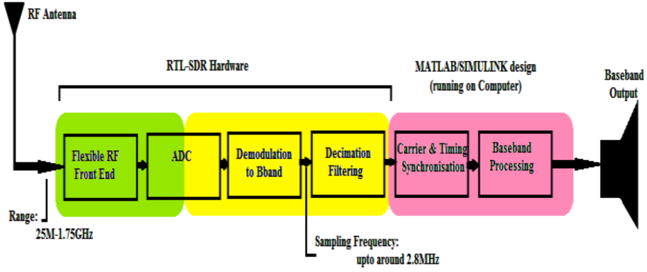


Fig.3.Block Diagram of RTL-SDR Receiver Chain

The antenna receives the RF signals and is quadrature down converted in RTL-SDR. These in-phase and quadrature phase samples are presented to the computer/PC running MATLAB. DSP algorithms are used to implement the receiver design and to demodulate the RF signal into baseband signal.

This RTL-SDR not only receives FM radio signals, but also capable of receiving UHF/DTV signals, Digital Audio Broadcast (DAB) radio, 2G 3G 4G cellular signals, GPS signals, scientific and medical (ISM) bands etc. In fact RTL-SDR can receive any desired signal in tuners range.

A. *RTL-SDR Technical Specifications*

1. 22-2200 MHz Tunable range (Depends on tuner model).
2. 3.2 MHz bandwidth (~ 2.8 MHz stable).
3. 8-bit ADC giving ~ 50dB dynamic range.
4. < 4.5dB noise power LNA.
5. 75 Ohm input impedance.

B. *RTL-SDR Bandwidth*

3.2 MHz is the maximum bandwidth of RTL-SDR, but the largest stable bandwidth is either 2.4 or 2.8 MHz depending on the PC. Setting large bandwidth will cause the samples to be lost giving choppy audio. Sometimes the bandwidth is referred to as sampling rate in RTL-SDR. Although these both are not same but setting the sampling rate to 2Msps (Mega Samples per Second) in RTL-SDR will give 2MHz of bandwidth.

C. *Input Impedance and Current Usage*

As these RTL-SDRs were designed for DVB-T their input impedance is 75 ohms. The RTL-SDR dongles have different current usages. The R20T dongle uses 300 mA of current while the E4000 requires 170 mA.

Table II: *RTL-SDR Compatible Dongles*

Tuner	Min freq. (MHz)	Max freq. (MHz)
R820T	24	1766
E4000	52	2200
FC0012	22	948.6
FC0013	22	1100
R828D	24	1766
FCI	146	438
FC2580	308	934

The most commonly purchased RTL-SDR is R820T. There are many other dongles available in market. RTL-SDR has many applications few are listed below:

- Decoding aircraft ACARS short messages.
- Sniffing GSM signals.
- Receiving NOAA weather satellite images.
- Radio astronomy.
- It is used as true random number generator.
- Receiving HF weather fax.
- Receiving wireless temperature sensors and wireless power meter sensors.
- Listening to VHF amateur radio.
- Receiving GPS signal and decoding them.
- Listening to satellite, ISS and unencrypted military communications.
- Listening to FM radio and DAB broadcast radio.
- Listening to amateur radio hams on SSB with LSB/USB modulation.

IV GNU RADIO

GNU Radio is an open source Software Defined Radio toolkit which has readily available signal processing blocks for implementing software radios. It can be used for readily available low cost RF hardware to create software-defined radios. It is most commonly used in academic and commercial environments to support wireless communication researchers as well as real-world radio systems [10]. Eric Blossom was the founder of first GNU Radio project. GNU Radio supports Linux and major Linux distributions pre-complies the packages. A port to windows is also developed with limited functionalities. GNU Radio has a library which includes signal processing blocks like modulators, demodulators, filters etc. which are used to construct a radio. GNU Radio is a two tire structure in which critical signal processing blocks are implemented in C++ [11], and higher level organizing, connecting and gluing the blocks together is done using python [12]. There is also a graphical environment available which is used to create a custom radio called as GNU Radio Companion (GRC).

1) *GNU Radio Architecture*

Fig.4. shows the GNU radio software architecture which consists of complex flow graphs that has a module and low level algorithms. These modules and low level algorithms are structured in C++ providing basic signal processing functions (ex: filters, FFT, channel encoding etc ;). These blocks /modules are automatically generated into python modules with the assistance of python “wrapper” or interface i.e., SWIG (Simplified Wrapped and Interface Generator) which also compiles and allows the integration between C++ and python language.

The signal processing blocks are usually written in C++ and to tie these blocks together in order to form a flow graph a python scripting language is used. These blocks which are generated with the help of python are used to construct a flow graph model. The communication of data through module buffers creating a simple scheduler which runs the blocks in sequential order for single iteration is done through python framework [13].

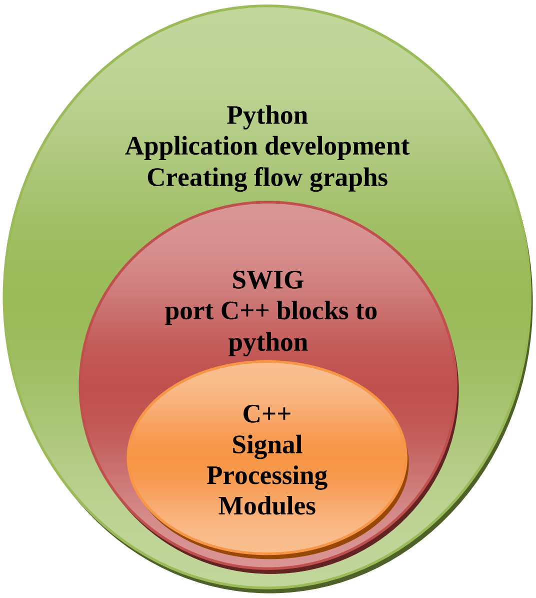


Fig.4. GNU Radio Software Architecture

GNU Radio typically consists of four elements [14]:

a) *SOURCE:* It is the start of the flow graph. Every flow graph has a signal source. Most common signal sources are file source, USRP source etc.

b) *SINK:* It is the end of the flow graph. Every flow graph is terminated with a single sink, USRP sink and file sinks are the commonly used sinks.

c) *FLOW GRAPH:* Every application depends on the flow graph where each flow graph consists of intermediate blocks containing a single source and sink blocks. A single application can have multiple flow graphs.

d) *SCHEDULER:* It is created based on data flow between the blocks to create a active flow graph which is responsible for transferring data through flow graphs. Input and output buffers are monitored by scheduler for sufficient data to trigger the processing functions for each blocks.

GNU Radio can run under several operating systems namely Linux, Mac OSX, NetBSD.

GNU Radio companion is a graphical interface that allows user to easily create GNU Radio applications which has a list of available modules that are inserted for each application by just double clicking or dragging it directly. Modules can be connected to each other very easily, once the blocks are dragged and connected GRC generates the corresponding python code automatically for the blocks and runs the application.

2) *Python*

Python is an object-oriented high level programming language which is used to implement the proposed frame work. The main cause for choosing python is the codes written in python is easier to understand by users and had only fewer lines of code. This code can also be written in other programming languages such as C++, C# and java. Python runs on broad range of platforms and its script is independent of operating system as they are executed on a python interpreter which runs on various operating systems ranging from desktop operating systems to mobile operating systems.

V. IMPLEMENTATION

Although commercial FM radio stations are broadcast all over the world the received signal at any particular location is poor. FM system removes noise easily compared to AM systems. When we tell “FM radio” we actually talk about its wideband FM (WBFM) which is the commercial standards for all the radio stations. This has a frequency deviation of 75KHz. FM signal contains infinite number of sidebands. These sidebands are located at every positive and negative multiple of the information signal located around the carrier [15].

For the purpose of our project, the WBFM receiver setup consists of:

- An RTL-SDR dongle and a stock antenna**.**
- A modern laptop computer running Linux.
- Supporting software packages, including

1. SDR – It is a PC based application to provide real-time radio functionality, data recording and it fully supports the RTL-SDR devices.

2. GNU Radio – It is accessible from a bootable Ubuntu Linux Live USB flash drive.

In order to design a real-time DSP device, computer laptop is the real-time processor. A number of software packages can be used to access the real-time data stream from the RTL-SDR. A variety of visualization tools are available in GRC like oscilloscope display, waterfall and FFT displays as well as audio sinks that interface to the PC soundcard. RTL-SDR is used here as it is very low cost device and its tunable frequency range lies in the range of FM signals. Hence it is used for the reception purpose.

i. *Hardware Requirements*

a. A laptop containing following specifications
    - Core i3 – processor or above
    - 4GB RAM
    - Hard disk – 250GB (minimum)
    - At least first generation PC

b. RTL-SDR dongle with an antenna (RTL382U R820T)

ii. *Software Requirements*

a. Linux OS: Ubuntu 14.04
b. GNU Radio 3.6.4.1

The hardware set up of a wideband FM receiver is shown in the fig.5.

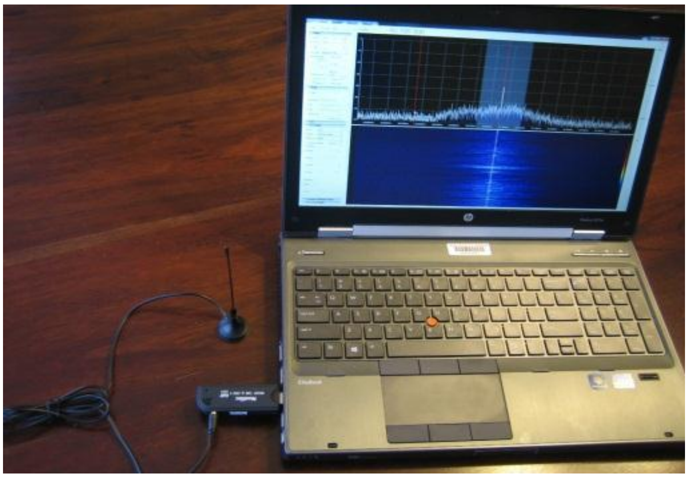
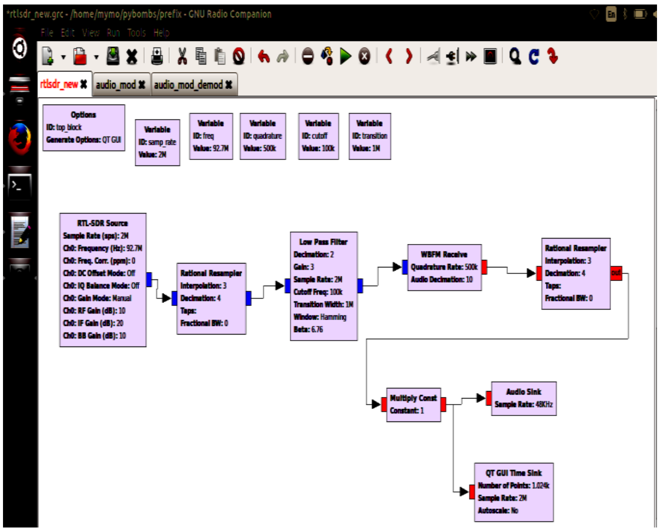


Fig.5. The RTL-SDR Based WBFM Receiver Setup

The WBFM receiver setup consists of R820T dongle connected to an antenna which in turn is connected to the PC installed with GNU Radio companion. The antenna captures the FM signal directly from the air, processes in PC and the signal is demodulated. Sound card is used to output this demodulated signal.

FM receivers are constructed using hardware which is fabricated in a plant which demonstrates the power of Software Defined Radio. Fig.6. gives the GRC flowgraph for FM radio receiver. The FM receiver consists of following blocks:

- RTL-SDR source
- Rational Resampler
- Low pass Filter
- Wideband FM receive/demodulator
- Audio sink



Options
ID: top_block
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 2M

Variable
ID: freq
Value: 92.7M

Variable
ID: quadrature
Value: 500k

Variable
ID: cutoff
Value: 100k

Variable
ID: transition
Value: 1M

RTL-SDR Source
Sample Rate (sps): 2M
Ch0: Frequency (Hz): 92.7M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 10
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 10

Rational Resampler
Interpolation: 3
Decimation: 4
Taps:
Fractional BW: 0

Low Pass Filter
Decimation: 2
Gain: 3
Sample Rate: 2M
Cutoff Freq: 100k
Transition Width: 1M
Window: Hamming
Beta: 6.76

WBFM Receive
Quadrature Rate: 500k
Audio Decimation: 10

Rational Resampler
Interpolation: 3
Decimation: 4
Taps:
Fractional BW: 0

Multiply Const
Constant: 1

Audio Sink
Sample Rate: 48KHz

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 2M
Autoscale: No

Fig.6. FM Radio Receiver GRC Flowgraph

A. *RTL-SDR source*

It is a source block that is used to capture the desired frequency signals with the help of an antenna. The sampling rate of the receiver is set to 2MHz and the desired frequency can be captured by setting the CH0. And CH0 here is set to 92.7 MHz in the RTL-SDR block. This value can be changed by adding an extra variable block [16].

B. *Rational Resampler*

Rational Resampler usually performs interpolation and decimation function to vary/adjust the sampling rate. The received signal from the RTL-SDR source block is decimated by four times. The decimation is a way to get more bits out of ADC i.e. the sampling rate decreases from 2Msps to 0.5Msps, by doing so we gain an extra bit out of ADC. Decimation has to be done carefully else wrong decimation leads to overlap of signals of interest with its neighboring signals.

C. *Low Pass Filter*

It performs filtering operation. Only the signal of interest is allowed through LPF. After decimation in Rational Resampler we use a low pass filter to select the WBFM signal of interest. The typical bandwidth of WBFM radio station is about 200 KHz, by decimating it by four the bandwidth is multiplied by four times i.e., 800 KHz. Since LPF cutoff frequency is measured for only one side of the bandwidth, we set the cutoff frequency to 100 KHz. The transition width of LPF defines the sharpness of our filter, smaller the transition width greater the CPU time consumed for doing calculations. If the transition width is 100e3 GNU Radio uses approximately 33% CPU and with the transition width of 10e3 it uses 45% CPU. Here the transition width is set to 1e6 which almost utilizes 100% of the CPU and the filter functions faster. The sampling rate is set to 2MHz.

D. *WBFM Receive*

In this block demodulation operation is performed on the received signal. Quadrature rate is varied to indicate the rate at which the demodulated signals are passed out. The quadrature rate is set to 500 KHz because 0.5 Msps is our sampling rate after decimation. The audio decimation is also set to 10 as it decreases the 0.5 Msps decimated signal to 50 KHz audio output signal which can be easily converted to 48 KHz which is used in our soundcards. We first down convert the signal to 12500 Hz by decimating it by 4 and then again interpolating it by 3 in rational resampler. Finally we get a sampling rate of 3x12500 = 37.3 KHz (approximately ~ 48 KHz).

E. *Audio Sink*

It is the block which terminated the flowgraph in GNU Radio that acts as an output. Here the tuned FM station can be listened at 48 KHz.

VI. RESULTS and DISCUSSION

In this section the results obtained by the above implementation are given. The output obtained at each and every stage is given here. These results are projected in the form of time plot. Time plot gives the variation of amplitude with time. The fig.7, fig 8, fig 9, fig 10, fig 11, fig 12 shows the output of WBFM at every stage in time plot.

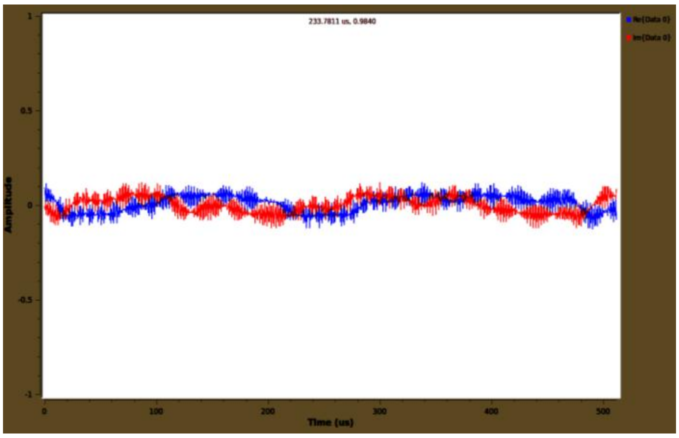


Fig.7. Time Plot of Input signal captured by RTL-SDR

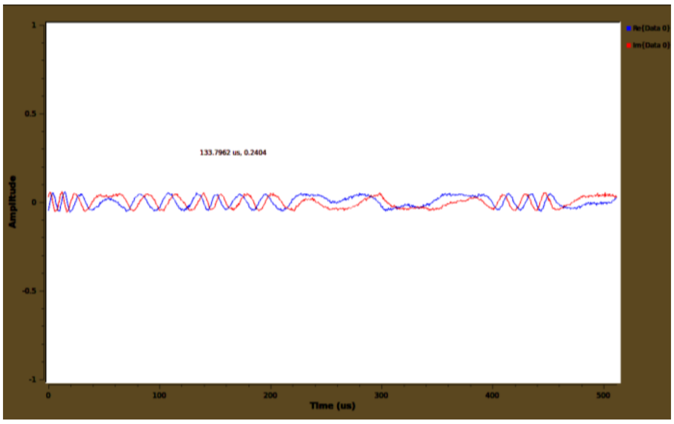


Fig.8. Time Plot across Rational Resampler

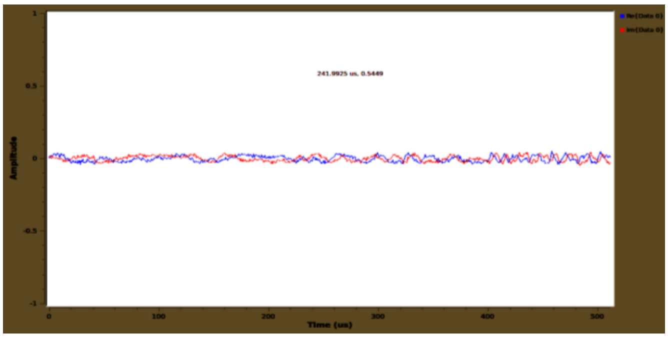


Fig.9. Time Plot across LPF (Preceding WBFM Receive)

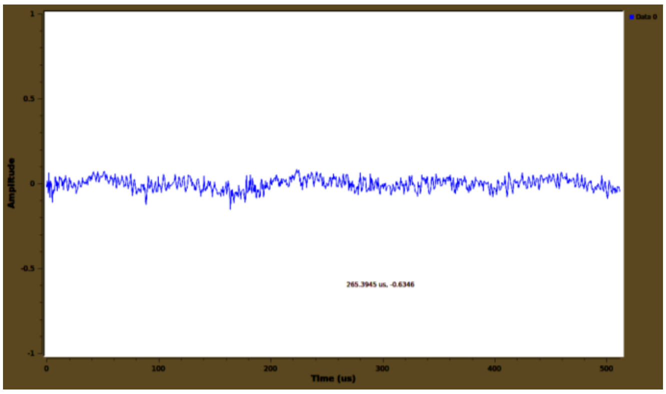


Fig.10. Time Plot across WBFM Receiver

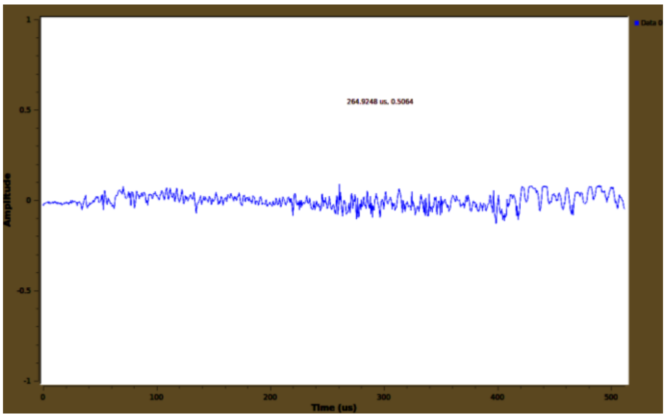


Fig.11. Time Plot across Rational Resampler (exceeding WBFM Receive)

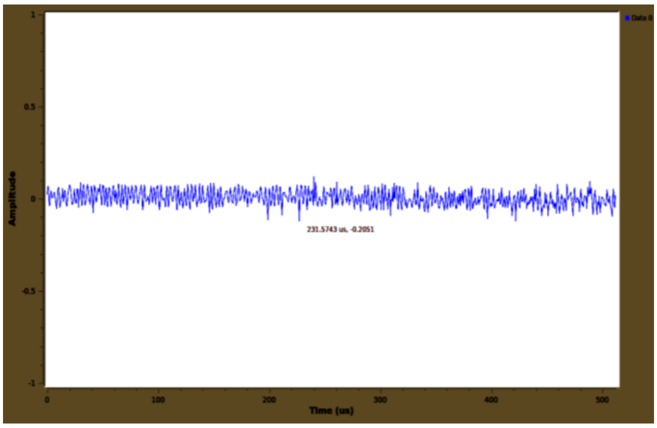


Fig.12. Time Plot of Audio Sink

From the results we can notice that the signal is complex in nature with both real and imaginary parts. The blue signal in the result plot represents the real part of the signal and the red represents the imaginary part. Fig.7. represents the captured FM signal from the receiving antenna which has amplitude of 0.9843 at the time 233.7811µs. when this signal is passed through the rational resampler the amplitude reduces to 0.2404 at the time interval 133.7962µs which is shown in fig.8. This reduction is due to interpolation and decimation operation performed. When this signal is passed through a LPF the amplitude becomes 0.5449 at the time 241.9925µs and showed in fig.9. In Fig.10. shows the output across WBFM receive block where actually demodulation operation is performed, here the signal amplitude is -0.6346 at the time 265.3945µs. at the audio sink the demodulated output is obtained with an amplitude of -0.2051 at the time 231.5743µs.

VII CONCLUSION

Software Defined Radio is bound to bring about the technological revolution for the current wireless communication system. GNU Radio saves time, low cost and is a powerful Software Defined Radio platform to create digital signal processing chain. In our work we use SDR as a new concept by combining it with popular frequency demodulation technique in wireless communication systems. An RTL-SDR dongle seems to be the most suitable hardware choice, which provides a sufficient frequency range and BW for educational purpose. Python is used as a programming environment because of its ease and scientific computing libraries. RTL-SDR is more versatile than other techniques. There is no need of writing a single line code to build a receiver which is the advantage of this project. SDR facilitates us to examine the spectrum, interference detection, to identify the intruders in the spectrum and characterizes the noise by bands. This continuous growth of SDR with receiving points throughout the world shows the formation of vast network worldwide in which there is possibility to listen a radio broadcast anywhere in the world with the help of internet.

REFERENCES

[1] "WIPRO", 21st January 2012 [online] Available: http//:www.broadcastpapers.com /broadband/WiproSDRadio.pdf.[Accessed 25 april 2013].
[2] A. L. Reis, A. F. Selva, K. G. Lenzi, S. E. Barbin and L. G. Meloni, "Software Defined Radio on Digital Communications: A new Teaching Tool," in Wireless and Microwave technology conference, 2012 IEEE 13th Annual. IEEE, 2012, pp.1-8.
[3] Rao Arvind Malleri, Khyati Vachhani, 2015 IEEE "Experimental Study on Wideband FM Receiver using GNU Radio and RTL-SDR".
[4] J. Mitola III, "Software Radios. Survey, Critical Evaluation and Future Directions", IEEE National Telesystems Conference, pp. 13-15, 1992.
[5] R. J. Lackey and D. W. Upmal, "Speakeasy: The Military Software Radio", IEEE Communications Magazine, vol.33, pp.56-61, 1995.
[6] (August 2014). SDR-Sharp official Website. Available: http://sdrdharp.com
[7] (August 2014). Amazon. Venta de Hardware SDR. Available: http://www.amazon.com
[8] V. Giannini, J. Craninckx, and A. Baschirotto, "Baseband Analog Circuits for Software Defined Radio", ed:Springer, 2008.
[9] B. Sklar, "Digital Communicatinos Fundamentals and Applications, ed: prentice Hall, 2001.
[10] GNU Radio official website. http://gnuradio.org
[11] C++ language tutorial. http://www.cplusplus.com/doc/tutorial
[12] Python. Website: http://www.python.org
[13] Alex Verduin, "GNU Radio Wireless Protocol Analysis Approach", October 2008.
[14] BBN Technologies Corp. "GNU Radio Architectural Changes".
[15] S. Haykin, John Wiely & Sons, " Communication Systems", 2008.
[16] V310c1r4pt0r.tk, "v310c1r4pt0r.tk using gnu radio companion simple fm radio tutorial," 2015. Available: http://v310c1r4pt0r.tk/2013/11/01/using-gnu-radio-companion-simple-fm-radio-tutorial