SDR and Error Correction using Convolution Encoding with Viterbi Decoding

Shriram K. Vasudevan
VIT/Embedded Systems, Vellore, India
Email: shriramkv@rocketmail.com

Siva Janakiraman and Subashri Vasudevan
SASTRA/Communication, Tanjore, India
SASTRA/Computer Science, Tanjore, India
Email: sivajanakiraman@gmail.com,vrsuba@gmail.com

Abstract— *The aim of the paper is to build a software based radio capable of demodulating Frequency Modulated signals with the ability to receive upto four stations simultaneously. In addition convolution encoding along with Viterbi decoding is also implemented to aid in error correction capability of digital transmission systems. The initial phase is aimed at software based demodulation of multiple channels of Frequency Modulated signals. The second phase is directed towards the Error correction using convolution encoding with Viterbi decoding for streaming data encountered in digital broadcast systems.*

Software Radio is a way of designing software very close to the antenna. The basic feature of software radio is that software will define the transmitted waveforms, and software will demodulate the received waveforms. This paper is developed with the help of free software based radio toolkit given by a community named GNU radio. Channel coding schemes need to be used extensively in the case of transmission of digital data over the air or through any other medium. These coding schemes also called FEC (Forward Error Correction) are used in cases where the re-transmission of the data is not feasible or possible. The channel coding schemes comes to the rescue in such cases where redundant data are sent over the transmission medium along with the message. In the receiver side, this channel coded signal is decoded to get back the original data even if the channel coded signal undergoes some interference from the noise in the transmission medium. Though the channel coding has a downside of requirement to transmit additional data over the transmission medium, the advantages offered by error detection and correction mechanisms to the receiver outweighs the disadvantage of additional data transfer rate and bandwidth requirements.

Index Terms—**SDR, Viterbi Decoder, Convolution Encoder**

I. Introduction

Software-Defined Radio (SDR) is an evolving technology that has received enormous recognition. In the past few years, analog radio systems are replaced by digital radio systems for various radio applications in civilian, military and commercial spaces. In addition to this, programmable hardware modules are playing a vital role in digital radio systems at different functional levels. SDR totally aims to take benefits of these programmable hardware modules to get open-architecture based radio system software.

SDR technology supports implementation of some of the functional modules in a radio system such as modulation/demodulation, signal generation, coding and link-layer protocols in software. This assists in getting reconfigurable software radio systems where dynamic selection of parameters for each of the above-mentioned functional modules is possible. A Complete hardware based system has many limitations.

Before getting into what software radio does, it is good to review the design of a traditional analog, hardware-based radio (Figure.1). In wireless communications, information is encoded into radio waves. These are collected (or transmitted) from (to) the air by the antenna. The received signal is then passed to a series of components that extract the useful information and convert it into the output of the radio. The basic design is the same whether the radio signal is destined for a cell phone, microwave repeater, or AM/FM car radio. Traditional radios are based on the super heterodyne (superhet) receiver circuit.

Manuscript received August 2, 2009; revised August 20, 2009;

doi:10.4304/jetwi.2.2.122-130

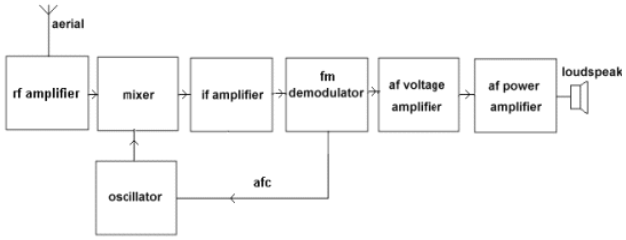


Figure 1. Generic FM receiver

In the superhet receiver, the incoming signal, which is Radio Frequency (RF), is first down-converted to a lower intermediate frequency (IF). The IF is then filtered for noise and amplified before being demodulated to produce the baseband signal that represents the desired information. This analog baseband signal may be passed directly to further downstage processing, or it may first be digitized and subjected to additional signal processing. There are several important reasons for down converting to a lower and standardized IF:

(a) It is easier and hence less expensive, to build filters and amplifiers – especially linear amplifiers – for a lower frequency signal.

(b) Use of a common IF enables standardization in the design of radio components.

Traditional designs for implementing the super heterodyne receiver architecture were optimized for specific frequencies and applications and each of the stages were implemented in hardware that was closely coupled. This was due largely to the difficulties inherent in and limitations in the state of the art in the design of analog signal processing components. Analog processing is much more complicated than digital processing, which is one of the key reasons why the transition to digital signals is so important.

In this paper chapter 2 deals about the performance and power limitations, Chapter 3 talks about architecture of SDR. Chapter 4 is on signal flow, Chapter 5 is dealing with how to derive instantaneous frequency. Chapter 6 talks about De Emphasizer. Chapter 7 explains about multi channel FM reception. Chapter 8 will give details about the Forward Error Correction. Chapter 9 and 10 speaks on implementation and results. Chapter 11 is concluding the paper followed by references.

II. Performance and Power Limitations

The change from hardware to software radios has faced lot of problems. First, performance generally comes down in the shift from dedicated to general-purpose hardware.

Secondly, the transition from hardware to software processing results in a substantial increase in computation which ultimately results in increased power requirements. This reduces battery life and is one of the key reasons why software radios will not be deployed first in end-user devices such as cell phones, but rather in base stations which can take advantage of external power sources.

III. Architecture

We now take a view at the overview of a basic conventional digital radio system and then look at how SDR technology can be used to implement radio functions in software. This will be followed by the software architecture of SDR. The definition of the software radio system is not absolute and depends on where the Analog to digital conversion is performed. The question of where the A/D conversion is performed determines what radio functions can be moved into software and what types of hardware are required. At one extreme, we consider calling it software radio if software is used at any stage within the radio. This case typically involves the digitization at the baseband for signal processing. In this case the actual demodulation process is performed before the conversion to digital format. This setup is more of a digital radio than software radio. What process can be done using software is very much limited. Since the actual demodulation itself takes place with hardware, it cannot be used for demodulation of other kinds of signal and hence is very limited in its functionality.

At the other extreme, we might choose to use software radio for cases in which A/D conversion is performed right at the antenna with all radio functionality implemented in software running on general-purpose hardware. In this case the Digitization of the signal takes place right at the antenna and thus gives us the complete flexibility. This setup requires the use of a very high speed and wideband ADC, not economically feasible at this point in time. Figure 2 shows the basic software radio architecture.

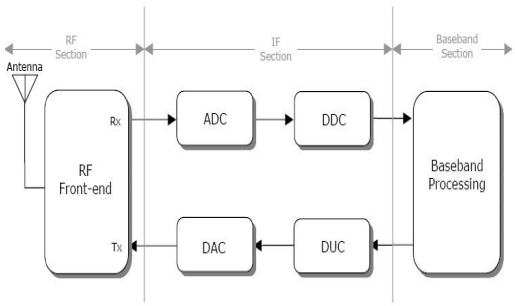


Figure.2 Basic Software Radio Architecture

In this paper, we take a stance that lies in between the two cases described above. The digitization of the signal is performed in the Intermediate frequency band, followed by the digital down conversion. The baseband processing is done in a general purpose computer. Though this setup does not give us a complete software control over the radio spectrum, it does offer us a great level of flexibility with the software since we can actually demodulate the signal using software. The architecture used in this project is indicated in Figure 3. This helps us to explore the part of radio spectrum which falls in the IF region that can be digitized.

The digital radio system consists of three main functional blocks: RF section, IF section and baseband section. The RF section consists of essentially analog hardware modules while IF and baseband sections contain digital hardware modules. The RF section (also called as RF front-end) is responsible for transmitting/receiving the radio frequency (RF) signal from the antenna via a coupler and converting the RF signal to an intermediate frequency (IF) signal. The RF front-end on the receive path performs RF amplification and analog down conversion from RF to IF. On the transmit path, RF front-end performs analog up conversion and RF power amplification.

The ADC/DAC blocks perform analog-to-digital conversion (on receive path) and digital-to analog conversion (on transmit path), respectively. DDC/DUC blocks perform digital-down conversion (on receive path) and digital-up-conversion (on transmit path), respectively. DUC/DDC blocks essentially perform modem operations, i.e., modulation of the signal on transmit path and demodulation of the signal on receive path.

For this paper work, the RF section and the IF section are handled by the hardware, followed by the baseband processing by Computer. The hardware used is known as the Universal Software Radio Peripheral (USRP).

III. Software based FM receiver

The real implementation of the project starts from here. The hardware used is a custom built board provided by the creators of the GNU Radio community, which is designed to run the free software toolkit provided by them. The main goal is to take the FM based demodulation one step further to receive up to four FM stations simultaneously.

What the USRP gives to the Computer is a Digital-Down converted, complex, quadrature signal in the Baseband. The remaining processing will be taken care by the software after getting the signal from the USRP board. Thus once the signal enters the computer, the demodulation of the FM signal consists of the following steps.

(a) Signal flow from the air to the computer (from real to complex)
(b) Getting the instantaneous frequency (from complex to real)
(c) De-emphasizer
(d) Audio FIR decimation filter
(e) Output to Soundcard/File

Thus each stage of this signal processing acts as a block with input and output ports. Each block receives the signal as input from the previous block, performs the required signal processing and gives the output to the next block in the chain.

The software architecture is implemented in two layers. The bottom layer is implemented in C++, which satisfies the performance requirement of the demodulation and signal processing. The top layer is implemented in python, which acts as a mask layer, interconnecting the bottom C++ layers which perform the bulk of the signal processing. This two layered architecture gives us the advantage or reusing the signal processing blocks and makes it easier to customize the signal processing flow necessary. Thus connecting or removing a processing block from the software demodulation chain is much easier. The entire software layer is built and run on a free UNIX port for windows called cygwin. Cygwin offers a UNIX platform on windows based system

IV. Signal Flow from the air to the computer (from real to complex).

Basically what the USRP does is to select the part of the spectrum we are interested in and decimate the digital sequence by some factor N. The resulting signal is complex with I/Q two channels. Thus USRP gives out a `complex' signal, with a data rate 256k samples per second, called `quadrature rate' – or quad rate because the complex signal has I/Q quadrature components (Figure. 3)

.

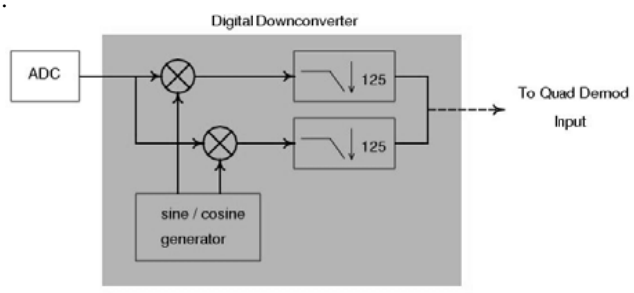


Figure.3 Digital down Converter

V. GETTING THE INSTANTANEOUS FREQUENCY (FROM COMPLEX TO REAL).

For FM, the instantaneous frequency of the transmitted waveform is varied as a function of the input signal. The instantaneous frequency at any time is given by the following formula:

$$f(t) = k * m(t) + f_c$$

m(t) is the input signal, k is a constant that controls the frequency sensitivity and f_c is the frequency of the carrier (for example, 100.1MHz). So to recover m(t), two steps are needed. First we need to remove the carrier f_c, then we're left with a baseband signal that has an instantaneous frequency proportional to the original message m(t). The second step is to compute the instantaneous frequency of the baseband signal.

Removing the carrier is taken care by the USRP, via the digital down converter (DDC). The resulting signal coming into the Computer has already become a baseband signal and the remaining task is to calculate its instantaneous frequency. If we integrate frequency, we get phase, or angle. Conversely, differentiating phase with respect to time gives frequency. These are the key insights we use to build the receiver. The angle between two subsequent samples can be determined by multiplying one by the complex conjugate of the other and then taking the arc tangent of the product.

Thus Arctangent of the product gives the phase difference between adjacent samples, if we divide it by the sample interval or multiply the data rate, we get the radian frequency ω, which gives the instantaneous frequency (f) if further divided by 2ω. This process of recovering the instantaneous frequency from the quadrature signal from the USRP is called the "Quadrature Demodulation" (Figure. 4).

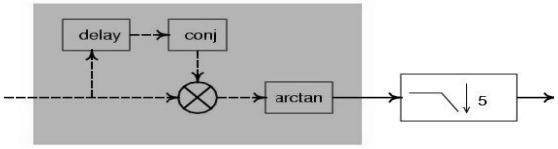


Figure.4 Quadrature Demodulation

VI. DE-EMPLASIZER

It has been theoretically proved that, in FM detector, the power of the output noise increases with the frequency quadratically. However, for most practical signals, such as human voice and music, the power of the signal decreases significantly as frequency increases. As a result, the "Signal to Noise Ratio" (SNR) at the high frequency end usually becomes unbearable. To circumvent this effect, `pre-emphasis' and `de-emphasis' were introduced into the FM system. At the transmitter, proper pre-emphasis circuits are used to manually amplify the high frequency components, and the converse operations are done at the receiver to recover the original power distribution of the signal. As a result, the SNR is improved effectively. In the analog world, a simple first order RLC circuit usually suffices for pre-emphasis and de-emphasis. In the case of the Software defined Radio, a first order IIR filter is implemented to get the magnitude of amplified, high frequency signal down to the magnitude of normal lower frequency signal, thereby improving the SNR.

Let us now see about Audio FIR decimation filter:

After passing the de-emphasizer, it is a real signal with a data rate of 256kHz.It is a baseband signal, containing all the frequency components of a FM station. The bandwidth of a FM station is usually around 2 * 100kHz. A sample rate of 256kHz is suitable for the 200kHz bandwidth, without losing any spectrum information

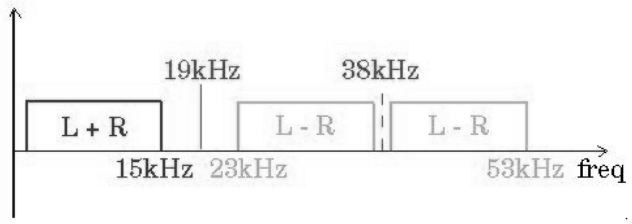


Figure.5 Typical FM Band

The FM signal band is spread out with various band of information (Figure. 5). From 0 to about 16 kHz is the left plus right (L + R) audio. The peak at 19 kHz is the stereo pilot tone. The left minus right (L - R) stereo information is centered at 2x the pilot (38kHz) and is AM-modulated on top of the FM. Additional sub carriers are sometimes found in the region of 57kHz - 96kHz. Thus for the reception of a mono FM station, a Low pass FIR filter with a pass band frequency of 15Khz and a transition band with 1Khz should suffice. Thus, once the signal comes out of the FIR decimation filter block, it is now ready to be played into the soundcard.

Output to Soundcard/File:

Once the signal passes through the FIR Decimation filter, it is ready to be played by the soundcard or alternatively stored in a File. The signal from the FIR decimation filter is in a raw bit format and can be converted to other formats using any suitable compression algorithms.

VII. Forward Error Correction

Forward error correction [FEC] is a system of error control for data transmission, whereby the sender adds redundant data to its messages, which allows the receiver to detect and correct errors (within some bound) without the need to ask the sender for additional data. The advantage of forward error correction is that retransmission of data can often be avoided, at the cost of higher bandwidth requirements on average, and is therefore applied in situations where retransmissions are relatively costly or impossible.

Such a typical situation where the receiver is not able to make a request for the re-transmission of message is FM reception. FEC devices are often located close to the receiver of an analog signal, in the first stage of digital processing after a signal has been received. That is, FEC circuits are often an integral part of the analog-to-digital conversion process.

Forward Error Correction Implemented:

The Forward error correction algorithm consists of two parts namely Encoding and Decoding. The Encoding is implemented using a Convolution algorithm and the Decoding implemented is the Viterbi algorithm. The convolution encoder and the corresponding Viterbi decoder are implemented in C programming language.

Motivation for Forward Error Correction

The situation in which the receiver is not able to make a request to the sender for re-transmission of message forms the basis for all forward error correction techniques. A typical radio receiver is a classic example in which the receiver has no provision to communicate with the sender. Under such circumstances, if an error occurred in the signal transmitted, there will be no way for the receiver to detect the error unless some error control mechanisms are incorporated into the signal before transmission itself. This process of adding error control to the signal before transmission is called Forward Error Correction.

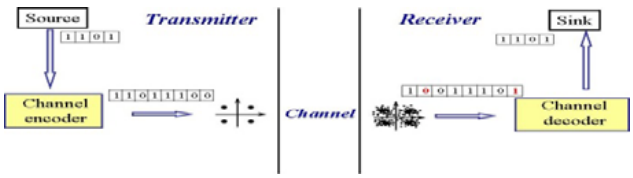


Figure 6. Typical wireless communication channel

A typical communication channel in which the data to be transmitted is encoded using some encoding algorithm by adding some redundant bits (Figure. 6). Thus the output of the encoder is called Channel Symbols. These channel symbols are then transmitted over the wireless channel.

Thus it can be seen that unless the error correction mechanisms are introduced there is no way for the receiver to recover the original information correctly. Hence Forward Error Correction mechanisms are essential if the receiver is to recover the original information. There are numerous forward error correction mechanisms each applicable to particular type of communication. The factors which govern which error correction algorithm is used for a particular communication type are

- Bandwidth of the spectrum
- Depth of error correction required
- Rate of encoding
- Processing capacity of Sender/Receiver

Introduction to Convolution Encoding

Convolution Encoding typically involves encoding of stream of data bits by using previous bits to perform a logical operation, the output of which is the encoded channel symbols. Thus depending upon the type of convolution encoding, a single data bit can have its influence on two or more adjacent bits thereby providing the required redundancy to the data. This influence which a bit has on other bits is what enables the receiver to identify any bit errors that occurred during the data transmission.

A convolution encoder is called so because it performs a convolution of the input stream with encoder's impulse responses. It can be represented mathematically as

$$y_i^j = \sum_{k=0}^{\infty} h_k^j x_{i-k},$$

where x is an input sequence, y^j is a sequence from output j and h^j is an impulse response for output j. A convolution encoder is a discrete linear time-invariant system. Every output of an encoder can be described by its own transfer function, which is closely related to a generator polynomial.

C.1. Convolution Encoding Algorithm in this paper

The convolution algorithm implemented in this project is a Half rate,(5,7) encoder with a constraint length of K=3, which is suitable for streaming data. .The convolution encoding typically involves the process of encoding data by using the bits of data to perform modulo 2 addition, thereby adding redundancy (Figure. 7).

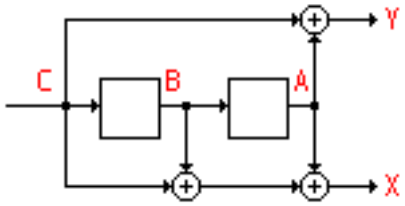


Figure 6. Basic convolution encoding

XIII. VITERBI ALGORITHM

The Viterbi algorithm implemented in this project is suitable for decoding data that has been encoded with a Half-rate,(7,5) convolution encoder with a constraint length of K=3.

IX. IMPLEMENTATION OF ALGORITHMS

The steps involved in simulating a communication channel using convolutional encoding and Viterbi decoding are as follows

1. Generate the data to be transmitted through the channel.
2. Convolutionally encode the data.
3. Introduce channel errors
4. Perform Viterbi decoding on the received channel symbols

X. RESULTS

The results of the project implemented are analyzed in the sections below.

Software Based Demodulation:

Forward Error Correction

The simulation setup consists of a channel encoder which generates random bit streams consisting of 0's and 1's which represents the message bits. This bit stream is then encoded. Then the encoded data is given to a channel error simulator to simulate bit errors. Then the message with errors is given as input to the Viterbi decoder and its error correction capability is analyzed.

Convolution Encoder

The length of the generated bit stream is taken as input from the user. Once the bit stream is generated, the convolution encoding is performed and the encoded data is written to a file. The encoder simulation is shown in figure.7

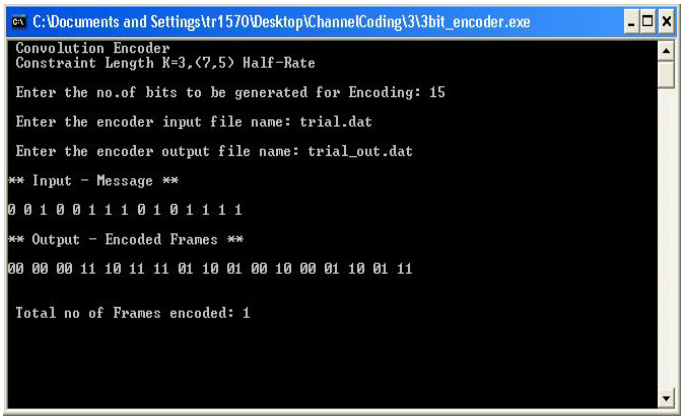


Figure.7 Simulation result of encoder

Since the encoding is typically performed on a stream of data, it must have a known length for the decoder to decode the message properly. Hence the encoder encodes the data in frames. The frame length used in this project is 15 message bits with 3 flushing bits for each frame. Hence the stream of data is broken up into frames of constant length and then encoded. Once the encoding process is completed, the encoded message is then written to a file.

Channel Error Simulation

In order to verify the error correction capability of the decoder, the encoded message must be included with random bit errors to simulate the message transmitted through a noisy channel. A model in which random bit errors occurs is created.

The number of bit errors to be introduced in each frame is taken as an input from the user and correspondingly random bits are inverted i.e. 0's changed to 1's and 1's changed to 0's which provides a reasonable model to simulate channel errors. The channel error simulation module is shown in figure 8

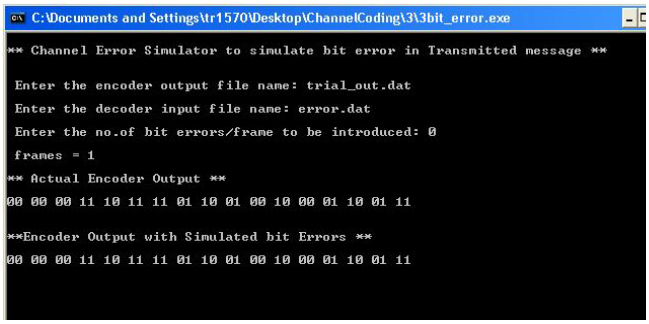


Figure.8 Channel error simulation

In this case, the number of bit errors introduced is zero and hence the encoder output and the received channel symbols are identical. The output of the channel error simulator is written to a file, which will then be taken as the input for the decoder.

Decoding received symbols

The input to the Viterbi decoder is the file containing channel errors, given as output by the channel error simulator. The decoder takes this file as input and then performs decoding and tries to recover the original message. The decoding simulation is shown in figure 9

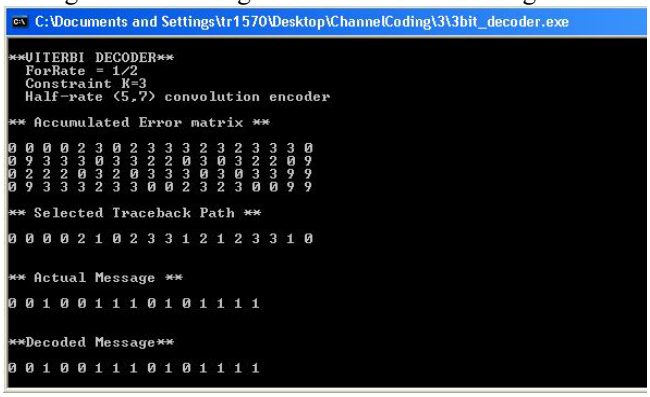


Figure.9 Decoding simulation result

While encoding messages are encoded in frame, so that decoder can start from a known state for each frame. Figure 9 shows decoding of channel symbols containing only one frame.

Performance of Viterbi Decoder

The Viterbi decoder implemented in this project is a Hard-decision decoder capable of decoding the channel symbols encoded by half-rate,(7,5) convolution encoder with a constraint length of 3. Various amounts of bit errors are simulated and the performance of the decoder is then analyzed.

The Test case taken up consists of a 60-bit message encoded into four frames by the encoder. This setup is shown in Figure

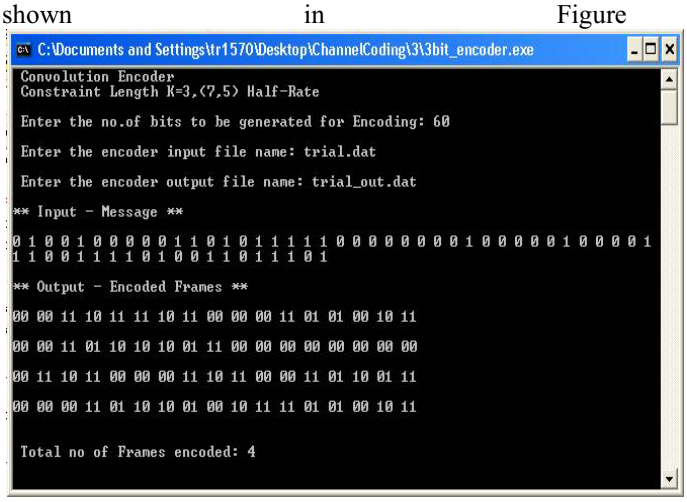
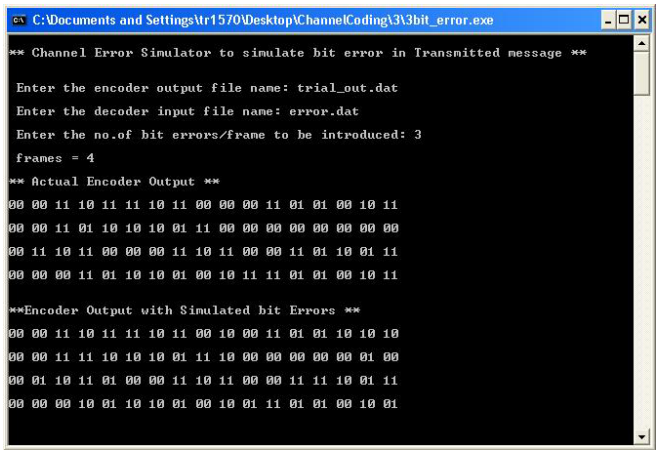
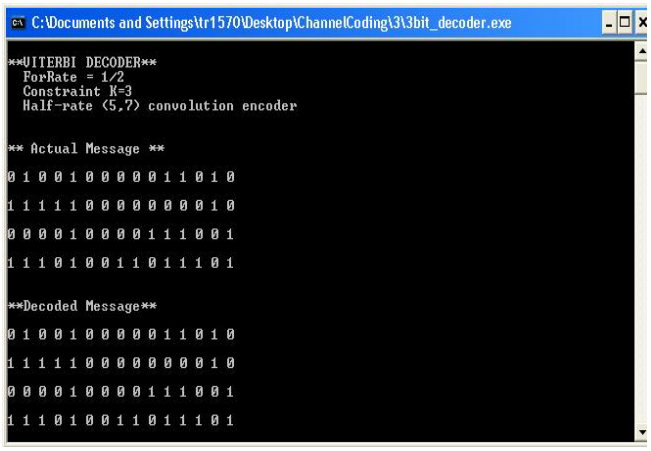


Figure 10. An example depiction

The message is encoded into four frames and is passed to the channel error simulator to introduce various amounts of bit errors and the decoder performance is analyzed.

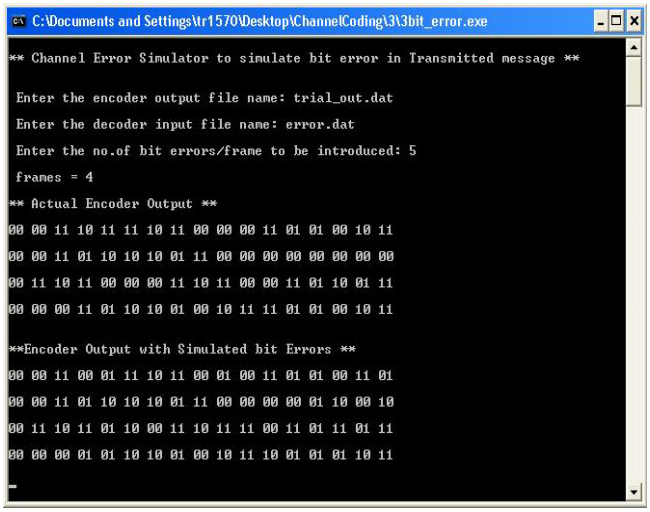


In the first case, three random bit errors are introduced in each frame. The performance of the decoder for this test case is shown if following Screen Shot.

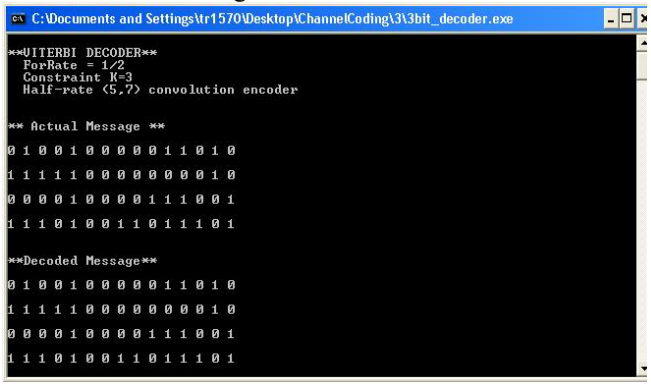


It is found that the decoder recovers the original message when three random bit errors are produced in each frame of the received channel symbols, offering 100% error correction capability for 3 random bit errors.

In the next test case, Five bit errors are introduced in each frame as shown in screen shot as follows.

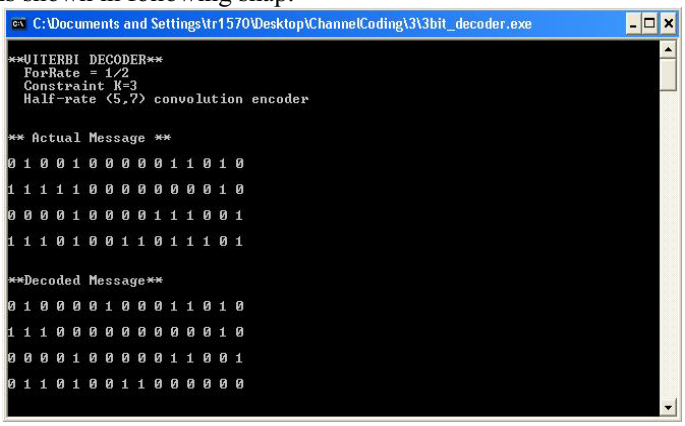


The performance of the decoder for this test case is shown in the following screen shot



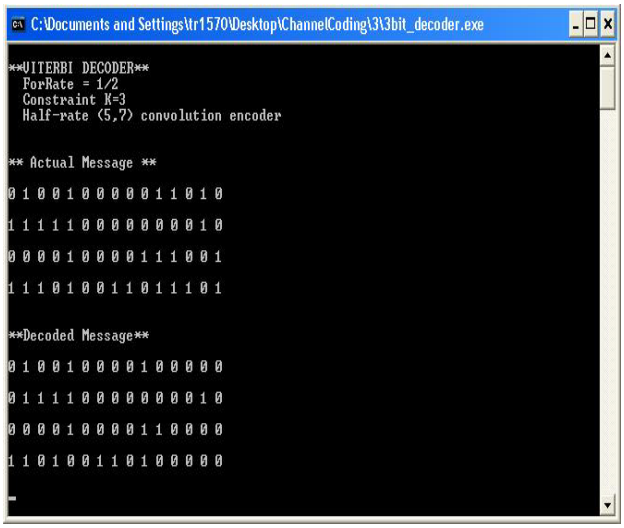
It is found that the Viterbi decoder is able to recover the original message in most of the cases when 5 random bit errors are introduced in each frame, thus offering almost 100% error correction when there is 33% error in the received channel symbols.

The next test case involves introducing seven bit errors per frame and the performance of the decoder in this case is shown in following snap.



It is found that the performance of the decoder deteriorates as the bit error increases to 7 errors per frame. The decoded messages differ from original message and on an average only 32% of the errors are corrected on a trial run with 10 samples.

When the number of bit errors per frame is further increased to 9 bit errors per frame, the performance of the decoder still goes down and the original message in not recovered by the decoder as indicated in next screen shot.



It is found that in case of nine bit errors per frame, the error recovery rate falls to as low as 8.2% on average run with 10 samples.

Thus overall, the performance of the Hard-decision Viterbi decoder falls with the increase in bit error rates and good level of error correction is available up to 5 bit errors per frame above which the message keeps getting deteriorated. This is indicated in the figure .11

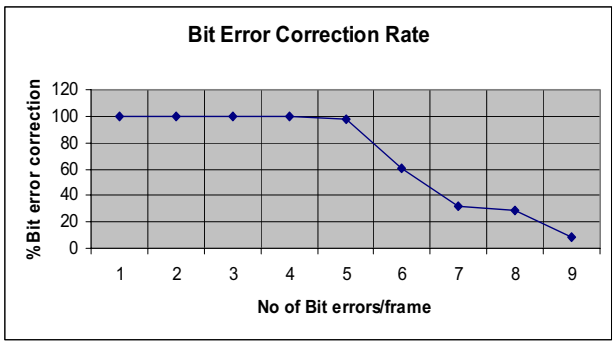


Figure 11. Analysis graph

XI. CONCLUSION

Thus the software based multi channel FM reception offers the capability to receive from one up to four FM stations at the same time. One station is streamed live to the speakers while the rest are stored in the disk in raw format. This offers the ability to listen to the recorded data at a later time thereby preventing the user from missing out on some important information.

Working with the FM band of the radio spectrum is the entrance to the Software based radio world which is capable of handling many parts of the radio spectrum. This project provided a good practical exposure to the problems involved in replacing a conventional radio with software based one and demonstrated the power and flexibility offered by using software in place of conventional hardware.

This project just touches the tip of the iceberg in the world of software defined radio, which has tremendous potential to be unleashed. With the aid of technology advancement, the software based radios will be able to explore the full radio spectrum very soon into the future.

With the move towards processing signal systems in digital format comes the problem of error detection and correction. Thankfully, this has been taken care by numerous error detection and correction algorithms, which were proposed long back, but were not implemented due to large computation requirements.

The convolution encoder and the Viterbi decoder implemented in this project offer a reasonable amount of error correction capability to a message signal.

These algorithms are the basic error correction algorithms on which many other complex error correction algorithms are built upon.

ACKNOWLEGDEMENT

We wish to thank all the co authors who supported this project for its success.

REFERENCES

[1] John G Proakis, Dimitris G Manolakis, "***Digital Signal Processing: Principles, Algorithms and Applications***", 1996, Prentice Hall, ISBN: 0-13-373762-4

[2] Todd K. Moon, "***Error correction coding: Mathematical methods and algorithms***",2002, John Wiley and sons, ISBN: 0-471-64800-0

Shriram K. Vasudevan was born at TamilNadu on 29-07-1983. He holds an M.Tech in Embedded Systems and B.E., in Electronics and Instrumentation. He has done his B.E., from Annamalai University, Chidambaram, India. He proceeded with his M.Tech at TIFAC-CORE of SASTRA University, Tanjore, India.

He has got overall 4 years of industrial experience in the field of Embedded Systems and Optical Networking. Currently he is holding the responsibility of Assistant Professor in the field of Embedded Systems in VIT University, Vellore, India. He was employed with Wipro Technologies for 2 years. He has visited USA, Dallas for his telecom training.

He has presented papers in National and International conferences in the areas of VLSI, Embedded and Networking. His research area focuses on Embedded Networking.

Siva Janakiraman was born at TamilNadu on 05-06-1982. He holds an M.Tech in Embedded Systems and B.E., in Electronics and Communication. He has done his B.E., from SASTRA University, Tanjore, India. He proceeded with his M.Tech at SASTRA University, Tanjore, India. He has got overall 5 years of teaching experience in the field of Embedded Systems. Currently he is holding the responsibility of Assistant Professor in the school of Electrical and Electronics Engineering in SASTRA University, Tanjore, India. His research area focuses on Embedded Systems and Cryptography.

Subashri Vasudevan, born at TamilNadu on 21-01-1989. She is currently doing her B.Tech and she is in her final year of her studies. She is graduating at SASTRA University. She has been consistently working on many projects and she is one of the few toppers in her department. She is currently doing project on Software Radio.

She has presented papers in the area of Software Radio and Wireless communications. Her research interest includes Wireless communications and Embedded Systems.