HARDWARE IMPLEMENTATION OF A RELAY IN A WIRELESS LINK

By

Sadaf Naz
Sajjal Akram
Capt.Waqas Khan

Submitted to the Faculty of Electrical Engineering Department

National University of Sciences and Technology, Islamabad

in partial fulfillment for the requirements of a B.E Degree in

Telecom Engineering

JUNE 2013

It is thereby certified that the project has been duly completed by a group of EE department under the supervision of Lt Col. Dr. Adnan A.Khan

(Signature)
Name: Lt Col. Dr. Adnan A.Khan
Date:

ABSTRACT

This project is an implementation of Cooperative MIMO which is the advanced technology of MIMO systems. This system has multiple nodes which cooperate with each other to transmit the signal with improved quality.

For this purpose, a Relay has been introduced between transmitter and receiver, which works on the principle of "Decode-Forward Technique".

The parameters of BICM and OFDM have been used for better data rate and improved quality. The signals at the receiver are added through MRC (Maximal Ratio Combining), giving reduced interference and over all better signal strength.

Dedicated

to

Our Parents

Acknowledgements

Our heartiest gratitude to “Almighty Allah” whose blessings and exaltations flourished our thoughts and enabled us to be among those who make contributions.

Words are lacking to express our humblest obligations to our ever helpful and dedicated supervisor **Maj Dr. Adnan Ahmed Khan**, Assistant Professor, Department of Electrical Engineering, Military College of Signals, Rawalpindi, who put a lot of effort through valuable guidance, sympathetic attitude, keen interest and constructive criticism for the accomplishment of the present work.

Special thanks to **Maj.Dr. Rizwan Ghaffar** for his valuable advice, skillful suggestions, his inspiring guidance and precious advice.

We express our good feelings to our colleagues for their moral and valuable cooperation throughout the entire course of project. We would also like to acknowledge the lab attendants for their patience and help during this work.

Our deepest love and thanks to our parents whose heartiest cooperation, kindness, prayers and noble guidance was a great source of strength to us, who motivated us to complete our work.

Contents

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK	Acknowledgement
ADC	Analog to Digital Converter
AES	Advanced Encryption Standard
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CMIMO	Cooperative MIMO
CRC	Cyclic Redundancy Check
CSI	Channel State Information
DAC	Digital to Analog Converter
DDC	Digital Down Converter
DUC	Digital Up Converter
FER	Frame Error Rate
GF	Galois Field
GMSK	Gaussian Minimum Shift Keying
GUI	Graphical User Interface
ISI	Inter Symbol Interference
LNA	Low Noise Amplifier
LOS	Line of Sight
MAC	Medium Access Control
MIMO	Multiple Input Multiple Output
NACK	Negative Acknowledgement
NRZ	Non Return to Zero
PER	Packet Error Rate

PSK	Phase Shift Keying
SIMO	Single Input Multiple Output
SISO	Single Input Single Output
SNR	Signal to Noise Ratio
SPN	Substitution Permutation Network
USRP	Universal Software Radio Peripheral

Chapter No. 1

Introduction

1.1 Project Description

Wireless communications have seen a remarkably fast technological evolution. Although separated by only a few years, each new generation of wireless technology has brought significant improvements in terms of link communication speed, device size, battery life, applications, etc. In past some years the technological evolution has reached a point where researchers have begun to develop wireless network architectures that depart from the traditional idea of communicating on an individual point-to-point basis with a central controlling base station. Such is the case with wireless sensor networks, where the traditional hierarchy of a network has been relaxed to allow any node to help forward information from other nodes, thus establishing communication paths that involve multiple wireless hops.

One of the most appealing ideas within these new research paths is the implicit recognition that, contrary to being a point-to-point link, the wireless channel is broadcast by nature. This implies that any wireless transmission from an end-user, rather than being considered as interference, can be received and processed at other nodes for a performance gain. This recognition facilitates the development of new concepts on distributed communications and networking via cooperation.

The technological progress seen with wireless communications follows that of many underlying technologies such as integrated circuits, energy storage, antennas, etc. Digital signal processing is one of these underlying technologies contributing to the progress of wireless communications. One of the most important contributions to the progress in recent years has been the advent of MIMO (multiple-input multiple-output) technologies. MIMO technologies improve the received signal quality and increase the data communication speed by using digital signal processing techniques to shape and combine the transmitted signals from multiple wireless paths created by the use of multiple receive and transmit antennas.

Cooperative communications is a new paradigm that draws from the ideas of using the broadcast nature of the wireless channel to make communicating nodes help each other in implementing the communication process in a distribution fashion and of gaining the same advantages as those found in MIMO systems. This system has multiple nodes which cooperate with each other to transmit the signal with improved quality. For this purpose, a **Relay** has been introduced between transmitter and receiver, which works on the principle of **"Decode-Forward Technique".** The parameters of **BICM** and **OFDM** have been used for better data rate and improved quality. The signals at the receiver are added through **MRC (Maximal Ratio Combining),** giving reduced interference and over all better signal strength. The end result is a set of new tools that improves communication capacity, performance and bit-error rate; reduce battery consumption and extend network lifetime; increase the throughput and stability region for multiple access schemes; expand the transmission coverage area; and provide cooperation tradeoff beyond source–channel coding for multimedia communications.

1.2 Report Organization

The report comprises of three main parts; Chapter 2 and 3 deals with the theoretical aspects of cooperative communications and software defined radio (SDR). The chapters also focus on the typical communication models and software radios that have been considered for this project. Chapter 4 consists of the simulation of cooperative schemes and theoretical comparisons of cooperative MIMO with conventional SISO and collocated multi antenna array systems (SIMO).Chapter 5 describes in detail the installation of GNU Radio. Chapter 6 and 7 describes the theoretical aspects of OFDM and BICM which are the parameters of LTE. Chapter 8 and 9 provide the software radio models of the physical, data link and medium access control layers. They consist of flow graphs that define the implementation of radio characteristics in software. Chapter 10 underlines the physical platform (test bed) that is created to carry out testing of cooperative MIMO protocols whereas chapter 11 concludes the report by detailing the performance comparisons of CMIMO with different communication techniques.

Chapter No. 2

Cooperative Communications

Fading is the major problem in a wireless communication due to which line of sight communication is not possible everywhere. Wireless transmitted signals are affected by noise, attenuation, distortion and interference.

2.1 MIMO

2.1.1 What is MIMO?

MIMO stands for Multiple Input Multiple Output. In wireless communications the term is used to express systems that employ multiple antennas for transmission and reception. This wireless communication technique has delivered higher data rates with greater data fidelity giving rise to efficient radio systems in terms of power and spectrum.

2.1.2 Why MIMO?

Fading remains the prime problem in wireless communications and line of sight communication is not possible everywhere. MIMO is a technique used to fight off multi path fading effects by physically placing antennas to create independent channel realizations. These self-created multi paths along with tested receiver combining techniques give a resultant signal that is stronger than any of the multipath signals with single antenna transmission.

2.2 Cooperative \ Virtual MIMO

2.2.1 Motivation: Limitations of MIMO Systems

The ***wireless channel is, by nature, broadcasting***. Even directional transmission is in fact a kind of broadcast with fewer recipients limited to a certain region. This implies that many nodes/users can hear and receive transmissions from a source and can help relay information if it needed. The broadcasting nature, long considered as a significant waste of energy causing interference to others, is now regarded as a potential resource for possible assistance. It is well known that the wireless channel is quite bursty /having uneven spurts, i.e., when a channel is in a severe fading state or its too noisy, it is likely to stay in the state for a while. Therefore, when a source information cannot reach its destination due to severe noise and interference, it will not be of much help to keep trying by leveraging repeating transmission protocols such as ARQ. If a third party that receives data from the source could help via a channel that is independent from the source destination link, the chances for a successful transmission would be better, thus overall performance will be improved.

MIMO systems have delivered much and are continued to be used for obtaining higher data throughputs with reliability. But these systems come with their limitations. Virtual MIMO or Cooperative MIMO is a technique used to replicate the advantages of MIMO without installing multiple physical antennas at the transmitter.

The use of multiple antennas is limited by the physical size of the device and the power source available to it. Mobile communication using cell phones is an example where it may be difficult to deploy multiple antennas because of physical size limitation of mobile devices and limited power source availability.

A wireless network system is traditionally viewed as a set of nodes trying to communicate with each other. From another point of view, because of the broadcast nature of wireless channels, one may consider these nodes as a set of antennas distributed in the wireless system. Adopting this point of view, multiple nodes in the network may cooperate with each other for distributed transmission and processing of information.

A cooperating node can act as a relay node for a source node. As such, cooperative communications can generate independent MIMO-like channel links between a source and a destination via the introduction of relay channels.

2.3 Diversity in Wireless Channels

Fading wireless channels present the challenge of being changing over time. In communication systems designed around a single signal path between source and destination, a crippling fade on this path is a likely event that needs to be addressed with such techniques as increasing the error correcting capability of the channel coding block, reducing the transmission rate, using more elaborate detectors, etc. Nevertheless, these solutions may still fall short for many practical channel realizations.

Viewing the problem of communication through a fading channel with a different perspective, the overall reliability of the link can be significantly improved by providing more than one signal path between source and destination, each exhibiting a fading process as much independent from the others as possible.

In this way, the chance that there is at least one sufficiently strong path is improved. Those techniques that aim at providing multiple, ideally independent, signal paths are collectively known as *diversity techniques*. In its simplest form, akin to repetition coding (where signal redundancy is achieved by simply repeating the signal symbols multiple times), the multiple paths may carry multiple distorted copies of the original message. Nevertheless, better performance may be achieved by applying some kind of coding across the signals sent over the multiple paths and by combining in a constructive way the signals received through the multiple paths.

2.4 Cooperative Diversity

The advantages of the MIMO technique have been widely acknowledged. However, due to physical size, price, or hardware limitations, most wireless devices may not be able to support multiple transmission antennas. This problem had decreased the performance gains and was solved by a new spatial diversity form, referred to as cooperative diversity.

Cooperative communications [1-3] is a new communication paradigm which generates independent paths between the user and the base station by introducing a relay node/ channel. The relay channel can be thought of as an auxiliary channel to the direct channel between the source and destination. Since the relay node is usually several wavelengths distant from the source, the relay channel fades independently from the direct channel.

The relayed information is subsequently combined at a destination node so as to create the environment of spatial diversity, so that system gain could be increased. This creates a network that can be regarded as a system implementing a distributed multiple antenna where collaborating nodes create diverse signal paths for each other.

A classic example of the cooperative diversity (CD) technique can actually be traced back to the groundbreaking work on the relay channel in 1977 [4]. In this work, the author analyzed the capacity of the three-node network consisting of a transmitter, a relay and a receiver. It was assumed that all nodes operate in the same frequency band, so the system can be decomposed into a broadcast channel from the viewpoint of the source and a multiple access channel from the viewpoint of the destination, as shown in Figure 2-1.

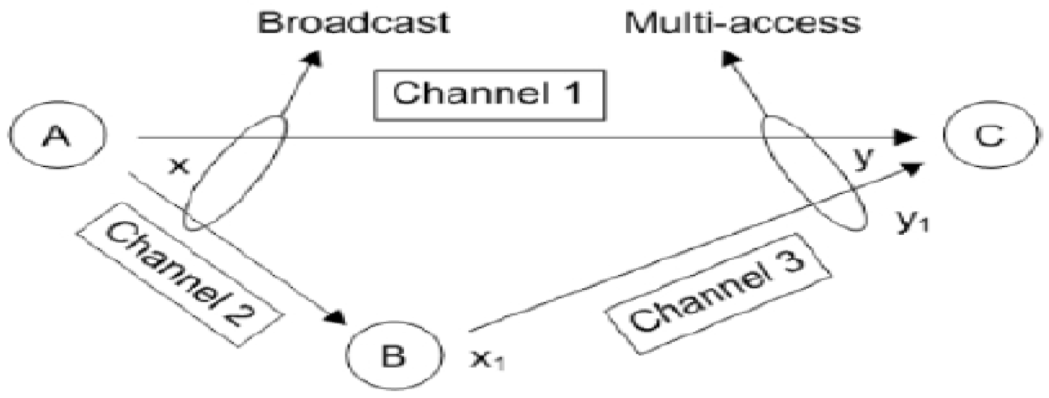


Figure 2-1 Channel Independence - Network nodes Operating in same Band

Where x and x1 are the transmitted signals from the source (A) and relay (B) nodes and y and y1 are the corresponding signals reaching the destination node.

2.4.1 Cooperative Diversity Techniques

There are number of cooperative diversity techniques proposed and mathematically modeled. The major difference in the diversity techniques is how the relay node relays the broadcast data of the source. The relay might have signal processing capabilities (***Decode and Forward***) or it might just be an amplifier (***Amplify and Forward***). Furthermore the techniques can be selective or incremental meaning that a user might want to relay selected data from the relay node or all of the data that the node receives. Generally in a channel with high probability of error the incremental techniques are favored.

Decode and Forward

- Simple and adaptable to channel condition (power allocation)
- If detection in relay node unsuccessful => detrimental for detection in receiver (adaptive algorithm can fix the problem)

- Receiver need Channel State Information between source and relay for optimum decoding

Amplify and Forward

- Achieve full diversity
- Performance is not better than decode-forward but better than direct transmission and decode-and-forward
- when number of relays tend to infinity, it achieves the capacity

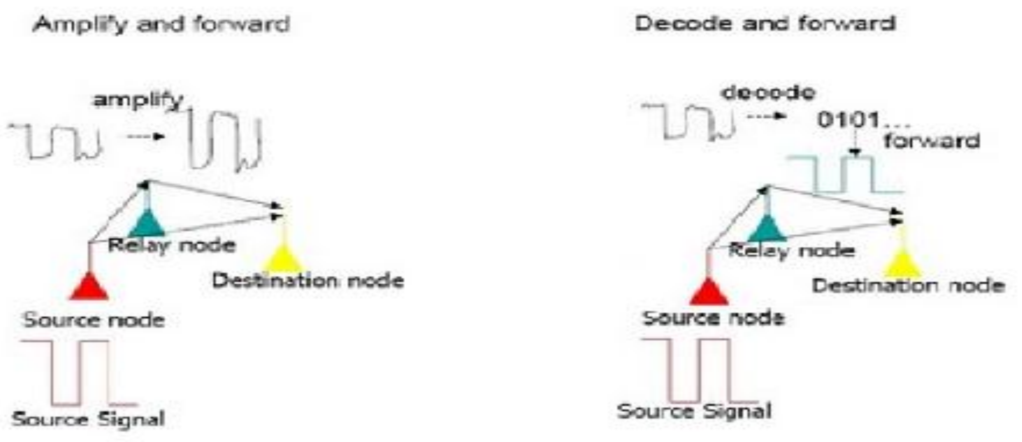


Figure 2-2 Cooperative Diversity Techniques

2.5 Diversity-Multiplexing Tradeoff

In MIMO systems, the multiple paths created between any pair of transmit-receive antennas can be used to obtain diversity gain. On the other hand, these paths can also be used to transmit independent messages from each transmit antenna, in which case it is possible to achieve an increase in transmit bit rate given by a multiplexing gain.

At the receiver, the MIMO configuration allows for separation of each data stream. It is readily apparent that there should be a tradeoff between diversity and multiplexing gains because the later is achieved at the expense of signal paths that otherwise could be used to increase the former. This project aims at exploiting the diversity gain provided by cooperative systems.

2.6 Wireless Fading Channels

In wireless communications, the presence of reflectors in the environment surrounding a transmitter and receiver creates multiple paths. As a result, the receiver will receive the scattered, reflected and diffracted signals from all directions. The receiver sees the superposition of multiple copies of the transmitted signal, which experienced differences in attenuation delay and phase shift while traversing a different path; such a phenomenon is called multipath fading. This can result in either amplifying or attenuating the signal and constructive or destructive interference, at the receiver. All simulations and practical performance on the SDR platform are considered to follow a Rayleigh distribution with slow fading.

2.6.1 Rayleigh Fading Channel

Rayleigh fading is the most applicable to model heavily built-up city centers where there is no line-of-sight (LOS) between the TX and RX, and many buildings and other objects diffract, refract, reflect and attenuate the signal. The Rayleigh distribution is frequently used to model multipath fading with no direct LOS path. The square root of a sum of two zero-mean identically distributed Gaussian random variables has a Rayleigh distribution. It can be assumed that the real and imaginary parts of the response are modeled by an independent and identically distributed zero mean Gaussian process, so the gain/amplitude of the response is the sum of two such processes. If r is defined as a Rayleigh distribution random variable, the probability density function (pdf) is given by

$$p_{(R)}(r) = \frac{r}{\sigma^2} e^{-r^2/2\sigma^2}, \qquad r \geqslant 0$$

where σ2 is the variance of two zero-mean identically distributed Gaussian random variables

2.6.2 Slow Fading Channels

A channel is generally referred to as introducing slow fading if *T*0 > *Ts* where *T*0 is the channel coherence time, and *Ts* is the time duration in which the channel behaves in a correlated manner is short compared with the time duration of a symbol. Here, the time duration in which the channel behaves in a correlated manner is long compared with the time duration of a transmission symbol. Thus, one can hope that the channel state to virtually remain unchanged during the time in which a symbol is transmitted. The primary degradation in a slow-fading channel, as with flat fading, is decrement in SNR.

2.6.3 Channel Considerations

The project for its simulation and the practical implementation on the SDR platform considers a Slow Fading Rayleigh Channel environment. Some important and useful terms to consider are the delay spread, Doppler spread, coherence time, and coherence distance and coherence bandwidth.

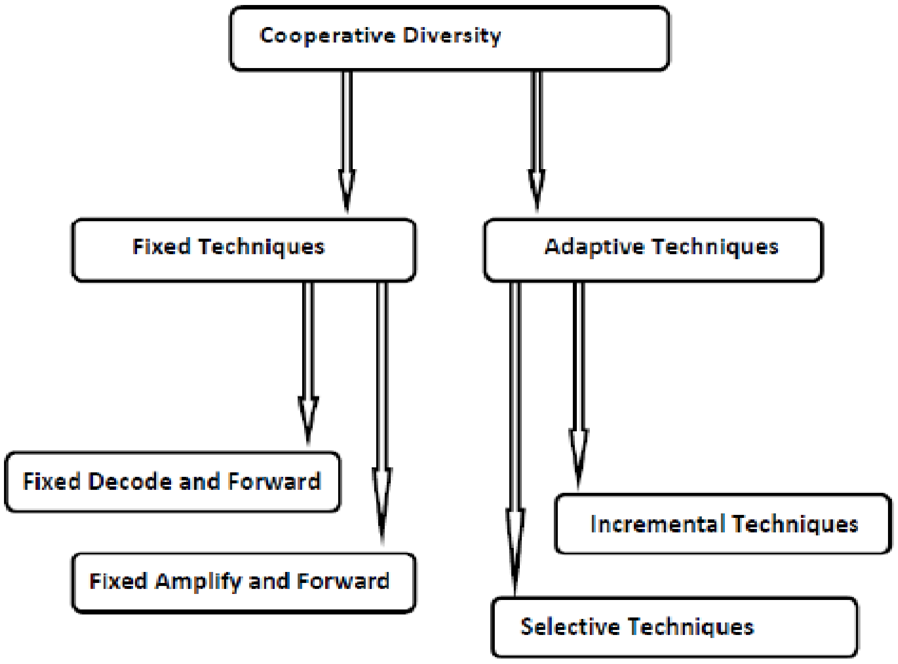
- Delay spread – time difference between first and last arriving signal copies
- Doppler spread – range of observed frequencies due to the movement of nodes
- Coherence time – time scale at which channel realizations are roughly independent
- Coherence distance – typical distance that a node must move to produce an independent channel realization

- Coherence bandwidth – the range of contiguous frequencies that experience essentially the same multipath fading effect

A minimum separation in successive frame transmissions must be equal to at least the coherence time. Coherence distance is the distance a node must travel to observe an independent channel realization. A common approximation of coherence distance is one quarter of the carrier wavelength. With the operational frequency of 2.45GHz, the coherence distance comes out to be approximately 3cm.

$$D_{coh} = \frac{\lambda}{4}$$

A modular breakdown of the cooperative diversity techniques is shown.



Chapter No. 3

Software Defined Radio (SDR)

The need of real-time experimental results was depicted to verify the systems parameters of throughput and reliability. For this purpose a flexible and reconfigurable platform was required. A platform capable of high speed processing and wireless transmission to take in the real channel effects of multi-path fading. These conditions made SDR the premium choice for the platform.

3.1 Software Defined Radio: Definition

A Software-Defined Radio system is a radio communication system where the components that have typically been implemented in hardware are instead implemented using software on a personal computer or embedded computing devices. The concept of SDR is not new, the rapidly emerging capabilities of digital electronics are making practically many processes possible that were once only theoretically possible.

A basic general purpose processor, rather than special purpose hardware is required for signal processing. Such design produces a radio that can receive and transmit widely different radio protocols (sometimes referred to as waveforms) based solely on the software used.

3.2 SDR: Architecture

As the name implies, the Software Defined Radio should be completely software based, but that‘s not the whole story. A generic SDR platform has two parts which are stated as follows

1. Software Part

2. Hardware Part

Figure 3-1 shows the basic architecture of a generic Software Defined Radio

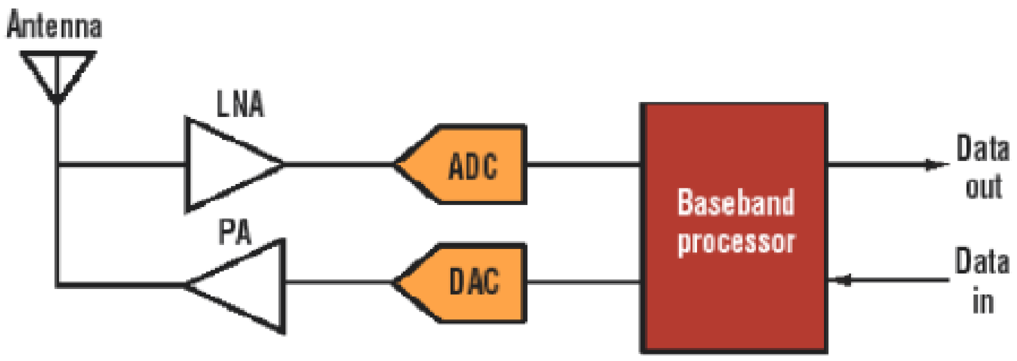


Figure 3-1 SDR Architecture

3.2.1 SDR: Software

The software part of the SDR basically handles the baseband processing components of a communication system shown in the Figure 3-1. Generic SDR software should support the following four functions

3.2.1.1 Radio Control Interface

The radio control interfaces provide a set of control functions to read and write various settings in the hardware. Furthermore, this interface should be useful to control the hardware parameters vital for a communication system like the gain settings and carrier frequency.

3.2.1.2 Main Program

The main program defines the basic architecture of the radio communication system. Core processing units are designed and implemented in the main program so the SDR software provides the platform for to code the main program.

3.2.1.3 Device Object

The device object implements the framework control interfaces and the data processing threads defined in main program for the transmit and receive paths. This deals with the data handling issues required for efficient use of the PC resources.

3.2.1.4 Port Implementation

The port implementation code contains the software to support the control and data interface ports. For example, the audio port and the USB port.

3.2.2 SDR: Hardware

The software part of the SDR basically handles the RF signal processing for wireless transmission and reception. The general components required for a SDR software compatible hardware are as follows

3.2.2.1 Modulators & Demodulators

These are required to mount the baseband signal onto the operating frequency.

3.2.2.2 Analog to Digital Converters & Digital to Analog Converters

ADCs are used to sample the received analog signal for bit determination are data recovery on the other hand DACs are used in the opposite direction to convert the digital data into analog signal in a way to produce the phase shift keying or frequency shift keying.

3.2.2.3 Digital Down Converter & Digital Up Converter

These are used to handle data inflow and outflow and to create and remove redundancy in the data. Up-converters are used on the transmission path to add extra samples in the signal to ensure better detection of the signal while at the reception part the Down-converters removes this added redundancy for saving the memory resources.

3.2.2.4 Local Oscillator

In order to demodulate a signal we need to regenerate the carrier frequency at the receiver side. For this purpose a local oscillator is needed.

3.2.2.5 Power Amplifiers

These are used at the transmission path to amplify the signal to be transmitted.

3.2.2.6 Low Noise Amplifiers

These are used at the reception side for noise eradication and better signal recovery.

3.2.2.7 FPGA

An FPGA is the central unit of the hardware that handles the control and signaling issues related with the RF hardware. This also acts as a dedicated processing unit since the main program designed in the software has some of its part relating to the baseband components burned in to the FPGA for handling the real-time aspect of the SDR platform

Figure 3-2 clearly explains a generic SDR architecture

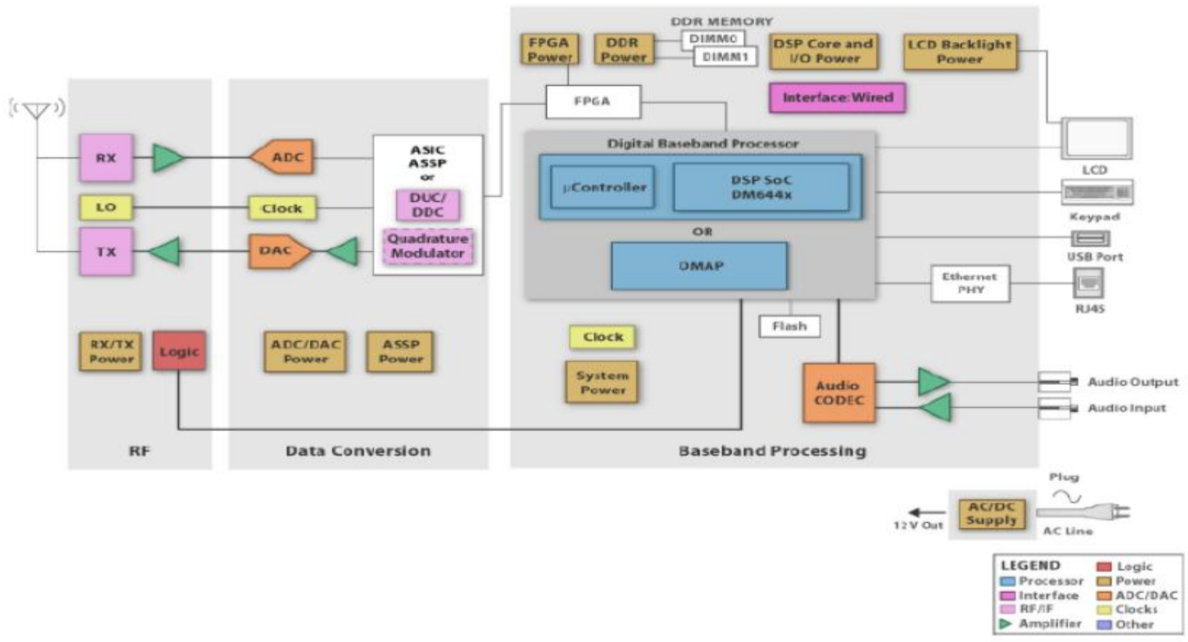


Figure 3-2 Detailed SDR Schematics

Considering the functions defined for the software and the hardware for SDR, we selected GNURADIO as the SDR software and GNURADIO compatible hardware for SDR is Universal Software Radio Peripheral (USRP).

3.3 UNIVERSAL SOFTWARE RADIO PERPIHERAL

The Universal Software Radio Peripheral or USRP is designed to allow general purpose computers to function as high bandwidth software radios. It serves as a digital baseband section and IF section of a radio communication system.

The basic design philosophy behind the USRP has been to do all of the waveform-specific processing, like modulation/demodulation, on the host CPU. All of the high-speed operations like digital down and up conversion, interpolation and decimation are done on the FPGA.

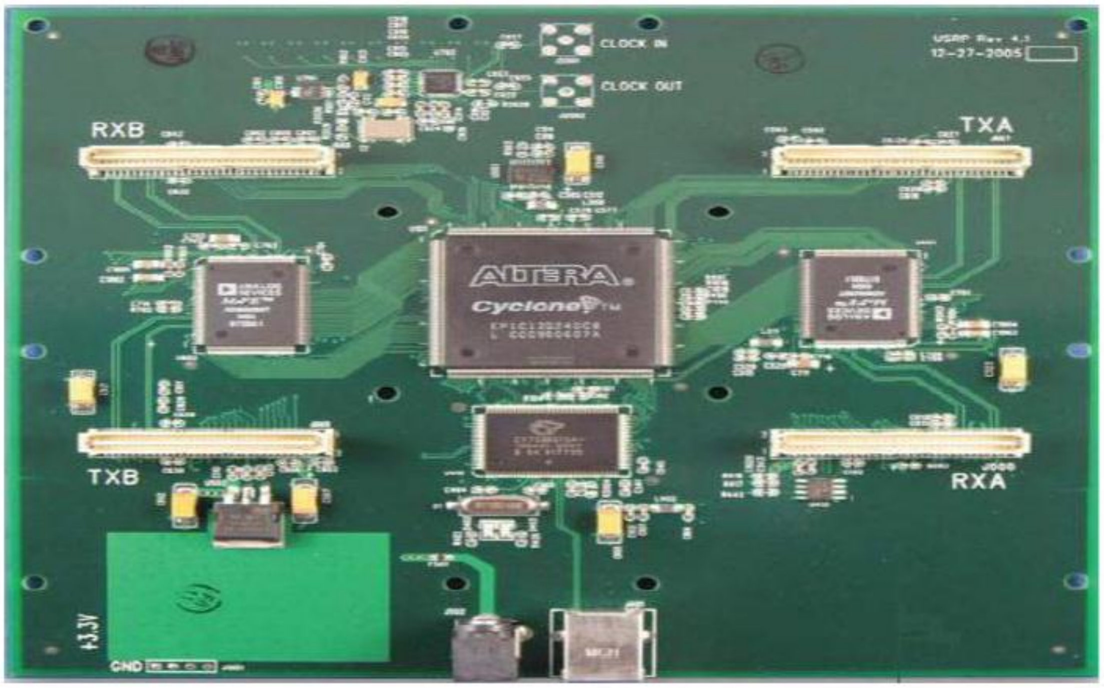


Figure 3-3 ETTUS USRP

The USRP has 4 high-speed analog to digital converters (ADCs), 64MSamples/sec, each at 12 bits per sample. There are also 4 high-speed digital to analog converters (DACs), 128MSamples/sec, each at 14 bits per sample. These 4 input and 4 output channels are connected to FPGA Altera Cyclone EP1C12. The FPGA connects to a USB2 interface chip, the Cypress FX2, and to the computer. The USRP connects to the computer through a high speed USB2 interface only, and it will not work with USB1.1.

3.3.1 ADC Section

There are 4 high-speed 12-bit AD converters. Its sampling rate is 64M samples per second. It could digitize a band as wide as 32MHz. These AD converters can band pass-sample signals of up to about 200MHz. The full range of the ADCs is 2V p-p, and the input is 50 ohms differential.

3.3.2 DAC Section

At the transmitting path, there are 4 high-speed 14-bit DA converters. Its clock frequency is 128 MS/s, so Nyquist frequency is 64MHz. The DACs can supply 1V peak to a 50 ohm differential load, or 10mW (10dBm). There is also PGA used after the DAC, which provides the gain up to 20dB.

3.3.3 Auxiliary Input/output Analog Channels:

There are 8 Auxiliary analog input channels connected to low speed 10 bit ADC inputs which can be read from software.ADCs can convert up to 1.25MSPS and they have a bandwidth of around 200 KHz. These analog channels are useful for sensing the RSSI signal levels, temperatures, bias levels etc. Additionally, there are 8 analog output channels connected low-speed 8bit DAC outputs. These DACs can be used for supplying various control voltages such as external variable gain amplifiers control.

3.3.4 Auxiliary Digital I/O Ports:

The USRP motherboard has high speed 64 bit digital I/O ports. These are divided in two groups (32 bit for IO_RX and 32 bit for IO_TX).

These digital Input/output pins are connected to the daughterboards interface connecters (RxA, TxA, RxB and TxB).

3.3.5 FPGA

Probably understanding what goes on the USRP FPGA is the most important part for the GNU Radio users. As shown in the Figure 3-4, all the ADCs and DACs are connected to the FPGA.

This piece of FPGA plays a key role in the USRP system. Basically it perform high bandwidth math. The FPGA connects to a USB2 interface chip. Everything (FPGA circuitry and USB Microcontroller) is programmable over the USB2 bus.

3.3.6 Digital Down Converter

The standard FPGA configuration includes digital down converters (DDC) implemented with 4 stages cascaded integrator-comb (CIC) filters. Cascaded integrator-comb filters are very high-performance filters using only adds and delays.

The standard FPGA configuration implements 2 complete digital down converters (DDC). Also there is an image with 4 DDCs but without half band filters. This allows 1, 2 or 4 separate RX channels.

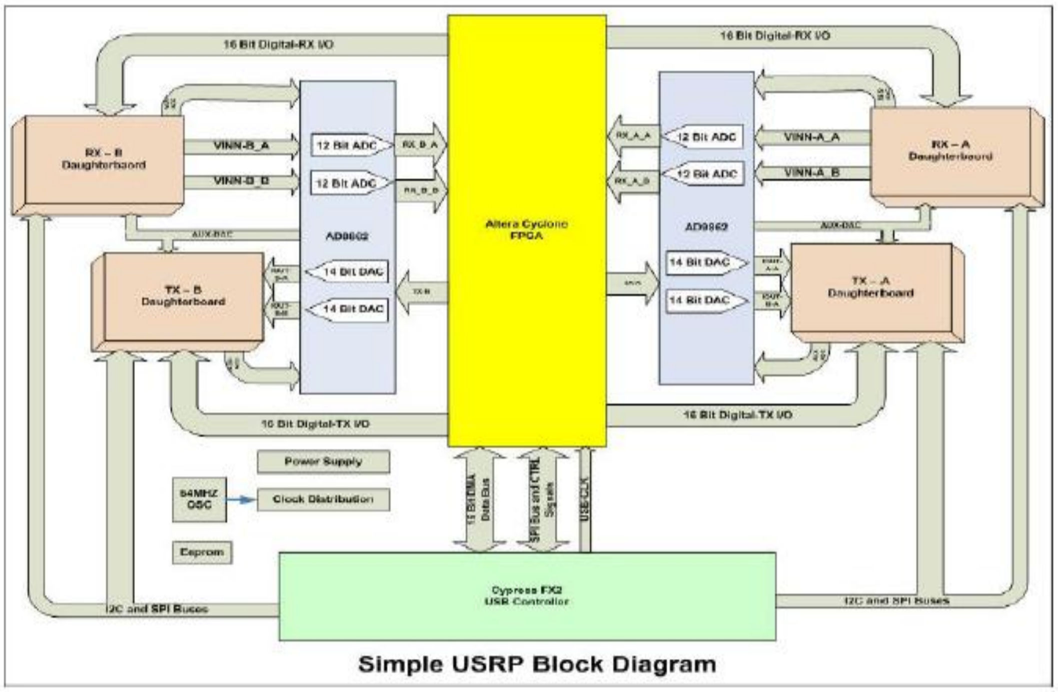


Figure 3-4 USRP Circuit Diagram

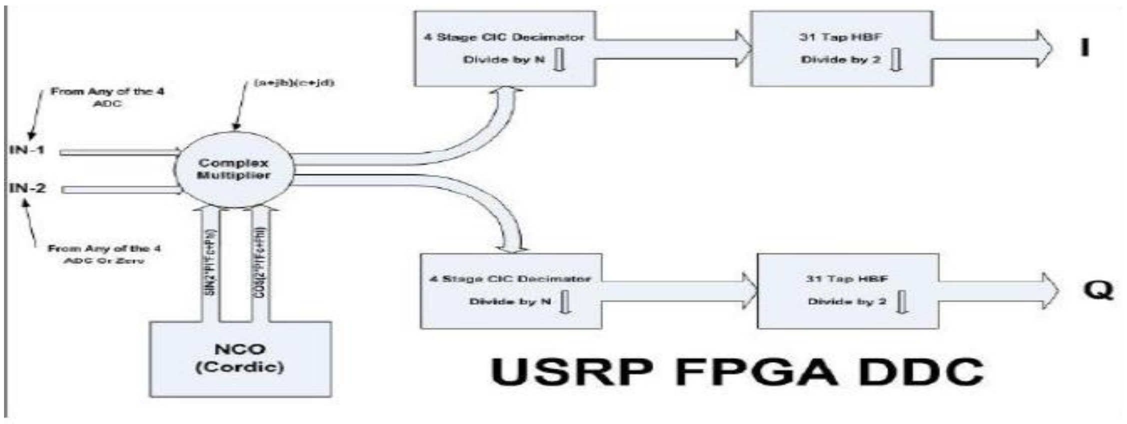


Figure 3- 13 Digital Down Converter

Figure 3-5 Digital Down Converter

3.3.7 Digital Up Converter:

The DUC down converts the signal from the IF band to the base band. Then it decimates the signal so that the data rate can be adapted by the USB 2.0 interface and is reasonable for the computers' computing capability. The complex input signal (IF) is multiplied by the exponential signal (usually also the IF) of constant frequency. The resulting signal is also complex and centered at 0. The decimator can be treated as a low pass filter followed by a down sampler.

At the TX path, the story is much the same, but it happens reversely. We required to send a baseband I/Q complex signal to the USRP board. The digital up converter (DUC) will interpolate the signal, then up convert it to the IF band and finally send it through the DAC.

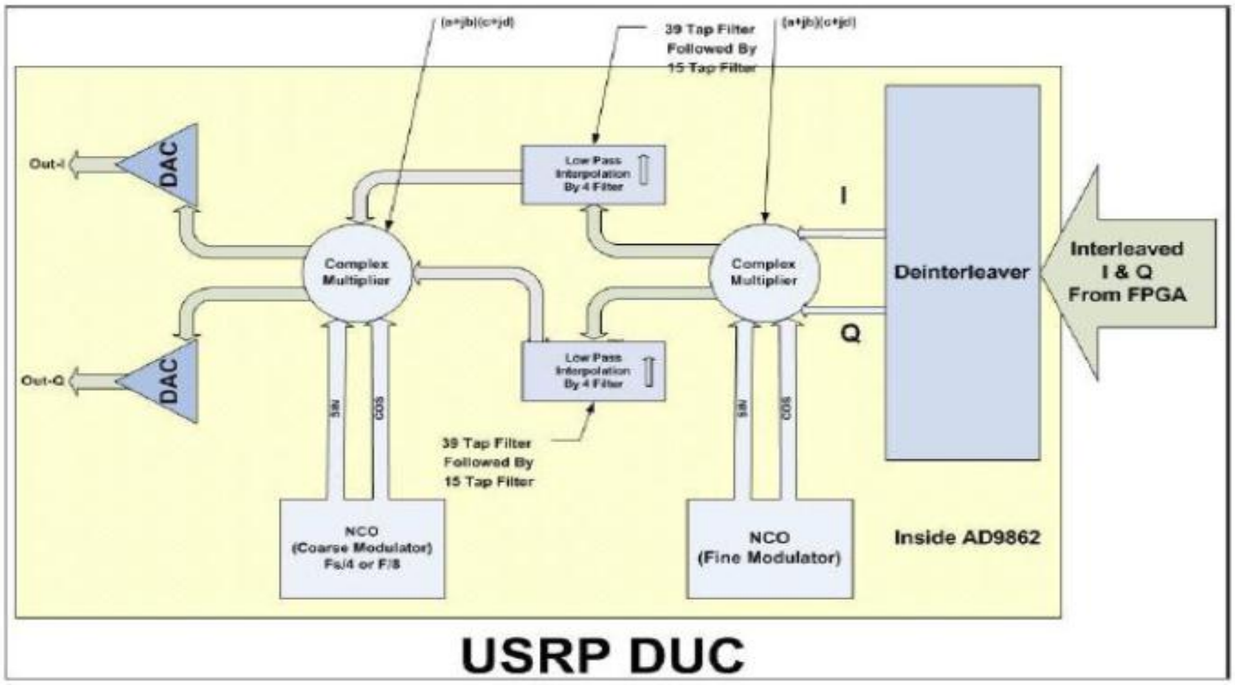

(a+jb)(c+jd)
39 Tap Filter Followed By 15 Tap Filter
(a+jb)(c+jd)
Out-I
DAC
Low Pass Interpolation By 4 Filter
I
Complex Multiplier
Complex Multiplier
Deinterleaver
Interleaved I & Q From FPGA
Q
Low Pass Interpolation By 4 Filter
Out-Q
DAC
SIN
COS
39 Tap Filter Followed By 15 Tap Filter
SIN
COS
NCO (Coarse Modulator) Fs/4 or F/8
NCO (Fine Modulator)
Inside AD9862
USRP DUC

Figure 3-6 Digital Up Converter

The USRP can operate in full duplex mode. In duplex mode, transmit and receive sides are completely independent of one another. The only point of consideration is that the combined data rate over the bus must be 32 Megabytes per second or less.

3.3.8 Daughterboards

The daughter boards are used to hold the RF receiver interface or tuner and the RF transmitter. There are slots labeled as TXA and TXB, and 2 corresponding RX daughter boards' slots RXA and RXB.

This allows the host software to automatically set up the system properly based on the installed daughterboard. If this EEPROM is not programmed, a warning message is printed, when USRP software is run, every time.

There are a number of daughter boards available meant to function in various frequency bands. Basic TX/RX Daughterboards, Low Frequency TX/RX Daughterboards, TVRX Daughterboards, DBSRX Daughterboards and RFX Daughterboards.

3.3.8.1 RFX Daughterboards

The RFX family of daughter boards consists of boards capable of functioning in a number of frequency bands. The RFX 2400 is the choice for pursuing this project.

The RFX family of daughterboards is a complete RF transceiver system. They have Independent RF synthesizers (local oscillators) for both TX and RX which enables a split-frequency operation. It has a built-in T/R switching and signal TX and RX can be on same RF port (connector) or in case of RX only, we can use auxiliary RX port. Most boards have built-in analog RSSI measurement. Each board is fully synchronous design and MIMO capable.

3.3.8.2 Power

The USRP is powered by a 6V 4A AC/DC power converter. The converter can handle of 90-260VAC, 50/60 Hz operations. The USRP motherboard itself only needs 5V, but to supply its daughterboards a 6V supply is needed.

It draws about 1.6A with 2 daughterboards connected to it. The power can be checked to be connected to the USRP by seeing a blinking LED on it. If LED is not blinking then check for continuity in the power fuse and check all power connections.

3.4 GNU RADIO

GNU Radio is a free ***software development toolkit*** [5] that provides signal processing in runtime and processing blocks to implement software radios using readily-available, low-cost RF hardware and commodity processors such as a General Purpose Processor (GPP) or Field Programmable Gate Array (FPGA).

GNU Radio applications are primarily written using the programming language, that is ,**Python**, while the supplied, performance-critical signal processing path is implemented in C++ using processor floating point extensions where available.

Thus, the developer is able to implement high-throughput real-time, radio systems in a rapid-application-development, simple-to-use environment.

A block attributes include the number of input and output ports it has as well as the type of data that flows through each block. Some blocks have only output ports or input ports. These ports serve as data sources and sinks in the graph.

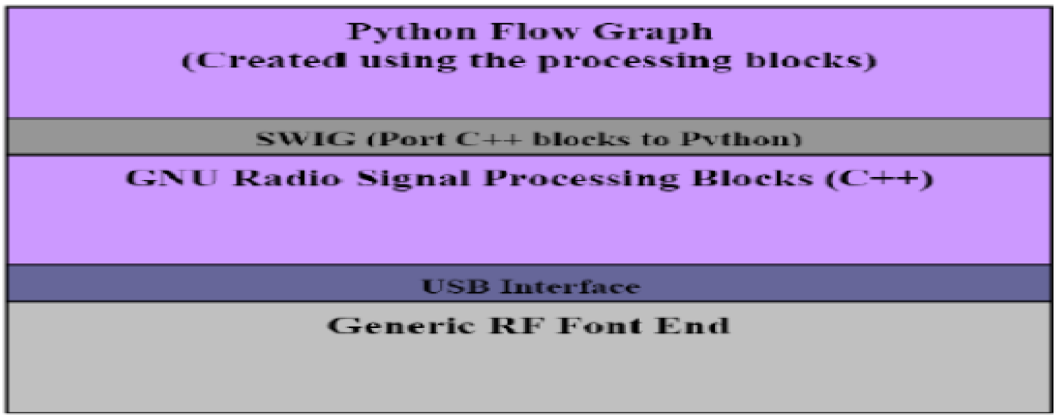

Python Flow Graph
(Created using the processing blocks)
SWIG (Port C++ blocks to Python)
GNU Radio Signal Processing Blocks (C++)
USB Interface
Generic RF Font End

Figure 3-7 Block Diagram of GNU Radio Components

The Universal Software Radio Peripheral or USRP2 is the recommended device for interfacing GNU Radio with the real world. The USRP kit has been developed especially for GNU Radio, and is available from ***Ettus Research.***. It is possible to use the USRP with software other than GNU Radio. It is possible to use GNU Radio with other I/O hardware

Before building the GNU Radio one must ensure that all the dependencies mentioned in [7] have been compiled. The installation and even a code run from GNU Radio is operating system specific. [6] Provides a list of supported operating systems. We opted for a Linux environment, the Ubuntu. The GNU Radio build guide for Ubuntu is available at [8]. A most helpful feature of GNU Radio is the GNU Radio mailing list [9].

3.5 GNU RADIO COMPANION (GRC)

This GUI termed GNU Radio Companion (GRC) allows users to interact with GNU Radio signal blocks in a manner similar to Lab view or Simulink. The entire interface is completely designed with GNU Radio in mind, and encompasses over 150 blocks from the GNU Radio Project. Blocks are integrated into GRC via descriptive python definitions. These definitions are very flexible, and allow multiple GNU Radio blocks to be grouped into a single GRC super-block.

The main components of the GRC are:

- Flow graph
- Signal blocks
- Parameters
- Sockets
- Connections

- Variables

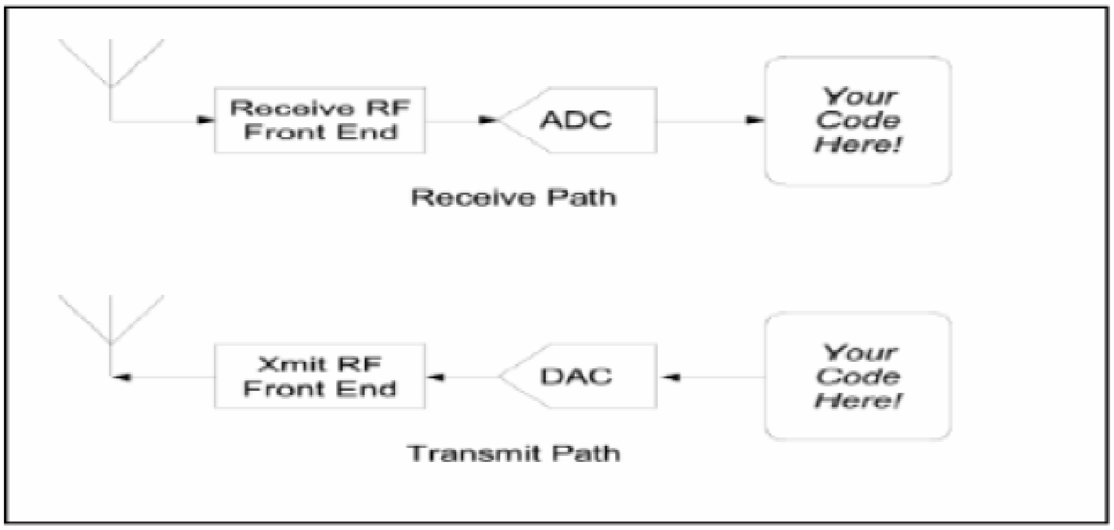


Figure 3-8 Transmit and Receive Paths

Chapter 4

Simulation of Cooperative Communication Systems

4.1 Why Simulation?

Simulation in general terms is defined as **"the act of imitating the behavior of some situation or some process by means of something suitably analogous (especially for the purpose of study or personnel training)".**

The choice of having a simulation platform to develop the schematic and proper concept of the proposed protocol was done due its obvious benefits.

4.1.1 Benefits of creating a simulation

There are numerous benefits of performing a simulation rather than actually building the design and testing it. Most of the time simulation testing is cheaper and faster than performing the multiple tests of the design each time. This also adds up to the advantage of controlling the expenditure.

Simulations take the building/rebuilding phase out of the loop by using the model already created in the design phase. Considering the university budget was also a major goal of the whole project, so as to make the whole communication setup as economically feasible as possible.

The second biggest advantage of a simulation is the level of detail that a person can get from a simulation. The benefit of simulation of providing a complete test bed platform was of pivotal importance to the whole project. A simulation could be set to run for as many time steps as a person desires and at any level of detail. The only restrictions in the simulation platform are the person‘s imagination, programming skills, and his CPU.

4.2 MATLAB Simulation

The basic simulation that was to be developed had to be similar to the undertaken project. It had to assume the same working environment and also had to follow the same protocol for data transmission and reception for making the results validate the practicality of the original project Simulation, as always, is the foundation step for the project. It had to prove the validity of the co-operative diversity protocol in an assumable constant working environment. The results also had to validate our proposed research protocol by proving the co-operative diversity protocol to be better than an ordinary SISO link that was the main aim of our practical scenario.

4.3 Success Criteria of the Simulated Project

Our basic aim was to practically prove the co-operative diversity protocol to be better than a normal SISO link. We also planned to prove the validity of this proposed protocol against a SIMO environment, to further validate the authenticity and the robustness of our proposed co-operative communication protocol.

In the simulation, much heed was paid to the PER vs. SNR graphs which were supposed to prove the feasibility of our research protocol, and also to prove its mark by showing better results than a SIMO scheme.

The intended results were achieved successfully and the resulting graphs, as shown, proved the co-operative diversity communication to be a much promising, and a better resulting scheme than a normal SIMO. With these promising results of the simulated platform, the basic motivation for going towards the practical implementation of the proposed protocol was achieved.

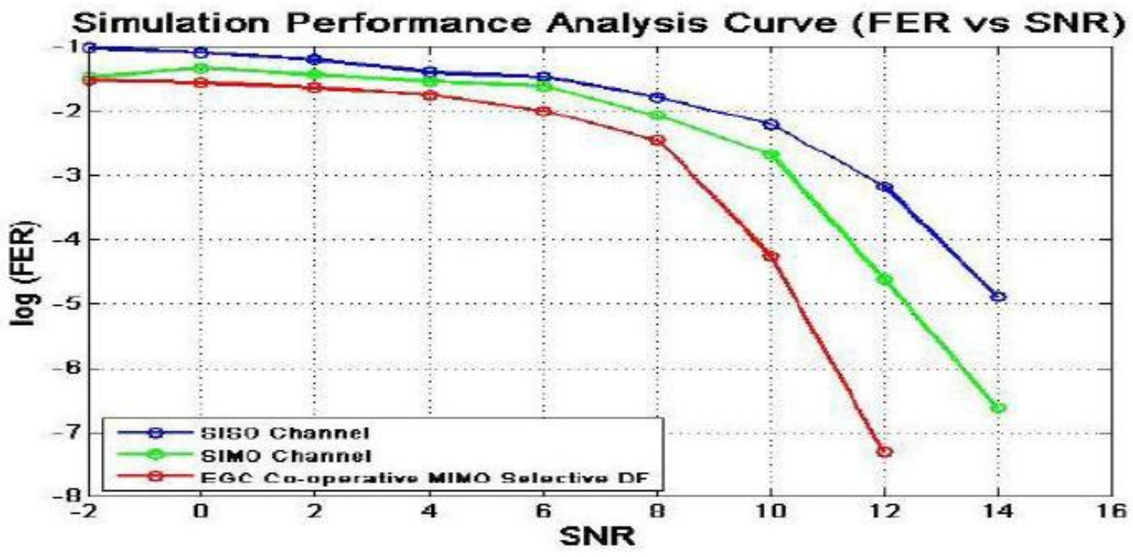


Figure 4- 1a Simulation Results for FER vs. SNR of different communication schemes

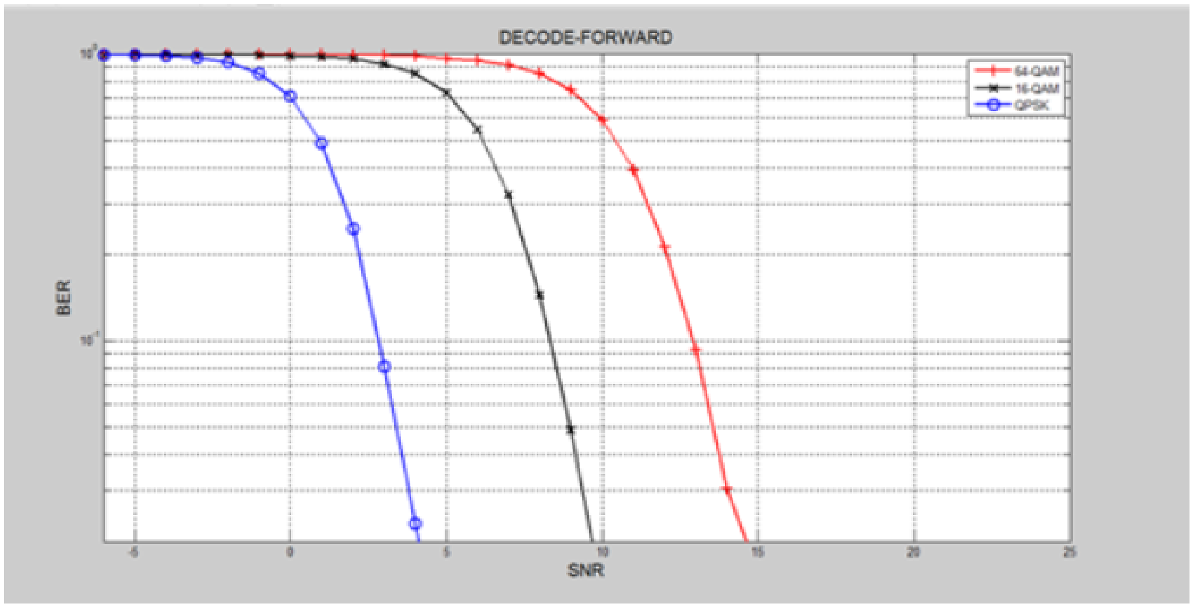
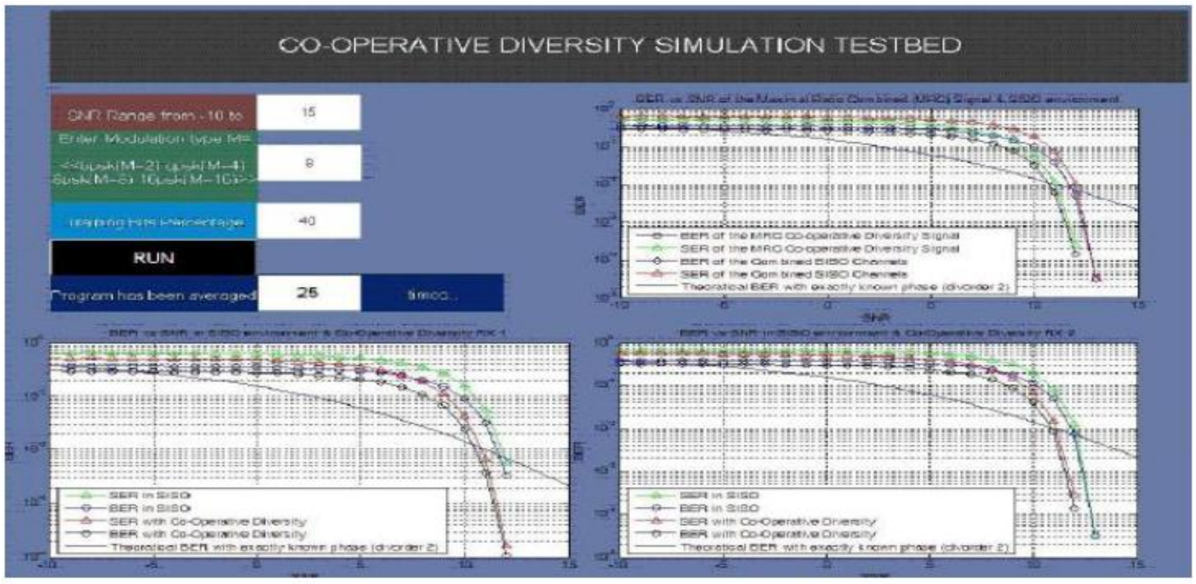


Figure 4- 1b Simulation Results for BER vs. SNR of different communication schemes Using Decode-Forward Technique

4.4 GUI of the Project

After the completion of the simulation a GUI was developed to make the whole simulation more interactive and also for easy testing of the simulated project. A picture of our GUI is shown below.

The GUI provided an easier way of inputting the varying parameters, and also with varying methodologies for transmission (modulation, training bits percentage etc.), to encompass as much real time variability as possible. This in turn provided the results to be generated in varying scenarios, thereby further authenticating the co-operative diversity scheme.


CO-OPERATIVE DIVERSITY SIMULATION TESTBED

SNR Range from -10 to: 15

Enter Modulation type M= <<bpsk(M=2) qpsk(M=4) 8psk(M=8) 16psk(M=16)>>: 8

Training Bits Percentage: 40

RUN

Program has been averaged 25 times...

BER vs SNR of the Maximal Ratio Combined (MRC) Signal & SISO environment
BER of the MRC Co-operative Diversity Signal
SER of the MRC Co-operative Diversity Signal
BER of the Combined SISO Channels
SER of the Combined SISO Channels
Theoretical BER with exactly known phase (divorder 2)
SNR
BER vs SNR in SISO environment & Co-Operative Diversity RX 1
BER vs SNR in SISO environment & Co-Operative Diversity RX 2
SER in SISO
BER in SISO
SER with Co-Operative Diversity
BER with Co-Operative Diversity
Theoretical BER with exactly known phase (divorder 2)
BER

Figure 4- 2 GUI of the Simulation for FER vs. SNR

4.5 Components of the Simulation

To make our simulation close to the practical scenario and also to make its results validate our proposed project, the whole simulation had to be designed in a way which would assume the real time channel scenarios and also the real time working of the transmission and reception sides, as close as possible. The main components of our simulation thus includes,

- Rayleigh Channel Simulator
- Gaussian Noise Adder
- Channel Estimator
- Equalizers
- MPSK Modulator & Demodulator
- CRC Generator
- CRC Detector And Error Evaluator

Each of above components is explained in detail below, along with their purpose and general functionality.

4.5.1 Rayleigh Channel Simulator

Rayleigh Channel is a statistical model for determining the effect on a radio signal of the propagation environment, such as that used by wireless devices.

Rayleigh fading models assume that the power/magnitude of a signal that has passed through a communications channel will vary randomly, or fade, according to a Rayleigh distribution — the radial component of the sum of two uncorrelated Gaussian random variables.

Rayleigh fading is viewed as a reasonable model for heavily built-up urban environments on radio signals, and is most applicable when there is no dominant propagation along a line of sight between the transmitter and receiver. The choice of having the Rayleigh channel simulator, create the basic traffic channel was done owing to its capability of encompassing the multi path scenarios. The channel generated by the Rayleigh distribution encompasses the varying real time parameters, thereby providing a viable option to simulate the real time channels.

4.5.2 Gaussian Noise Adder

This was implemented to further simulate the real time channel as close as possible to the actual practical scenario. Noise plays a vital role in any communication scheme, and is of pivotal importance in determining the feasibility of the communication platform.

This impediment was therefore also considered and additive white Gaussian noise was added to simulate the channel effects in a more pronounced way.

The following MATLAB command was used for this purpose

```
y = awgn (x, snr)
```

It adds white Gaussian noise to the vector signal x. The scalar value of SNR specifies the signal-to-noise ratio per sample, in dBs, if x is complex, AWGN adds complex noise.

4.5.3 Channel Estimator

Channel Estimation is the process of characterizing the effect of the physical medium on the input sequence. It is an important and necessary function for wireless systems. A receiver can gain insight into the data sent over by the transmitter, even with a limited knowledge of the wireless channel properties. The main goal of Channel Estimation is to measure the effects of the channel on known or partially known set of transmissions. The channel estimation technique which we implemented on our system was the training based channel estimation.

4.5.3.1 Training Based Channel Estimation:

Channel is estimated based on the training sequence which is known to both transmitter and receiver. The receiver can utilize the known training bits and the corresponding received samples for estimating the channel.

4.5.3.2 Functionality of the channel estimator

In our simulation, we developed a function for providing the basic channel estimation. It worked on the concept of *training based estimation* and utilized the received and the transmitted training sequences to estimate the effects of wireless propagation on the channel and also on the transmitted payload.

The function divided the distorted training sequence at the receiver and the known training sequence from the transmitter to work out the channel. The sequence that was transmitted contains both the effect of the communication channel and also the noise on it. When this distorted sequence is divided with the knowing training sequence at the reception side, it provided a rough estimate of the channel which is prevailing between the transmission and reception sides.

4.5.4 Equalizers

Equalization or **equalization** is the process of using passive or active electronic elements or digital algorithms for the purpose of altering (originally flattening) the frequency response characteristics of a system. The term generally implies amplitude correction or modification relative to frequency, but any frequency-dependent response characteristic can be equalized. Phase and time-delay methods can be used in equalization.

In our simulation the equalizer applied adjusted, or equalized the channel effects on the transmitted payload. The channel and phase error that were estimated through the estimator functions were nullified at the receiving side to get the original payload back.

4.5.5 MPSK Modulator & Demodulator

The choice of the modulation scheme for the simulation was quite vital, as it had a direct correspondence to the practical approach of the project. As the simulation was built to test the proposed protocol in varying environments, thus variable PSK schemes were implemented to be the modulation choice.

4.5.5.1 MPSK Modulator

M-PSK modulation was achieved through a built function which modulated the data according to the **Eigen Function** properties. The simulation was made capable of modulating the data in the following PSK forms

- BPSK
- QPSK
- 8PSK
- 16PSK

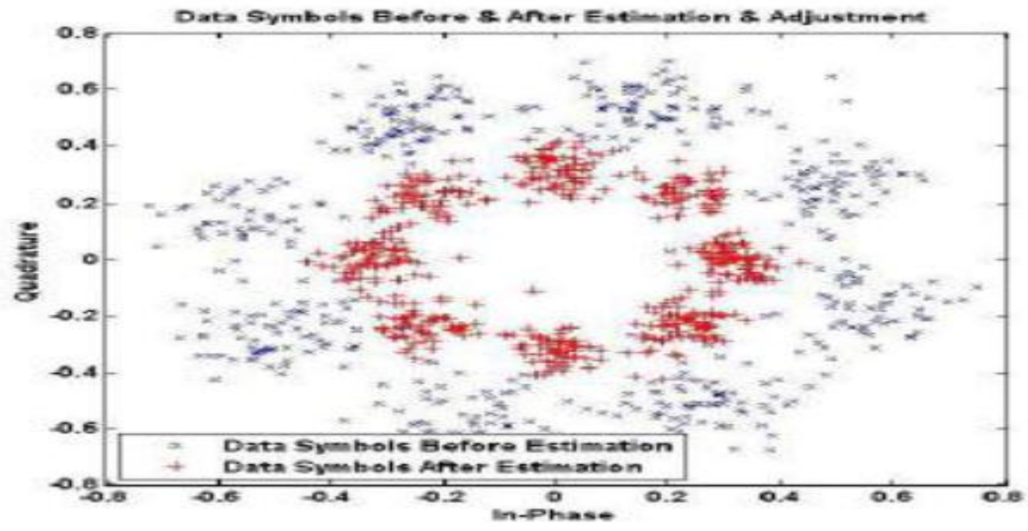


Figure 4-3 Equalization Results of 8PSK

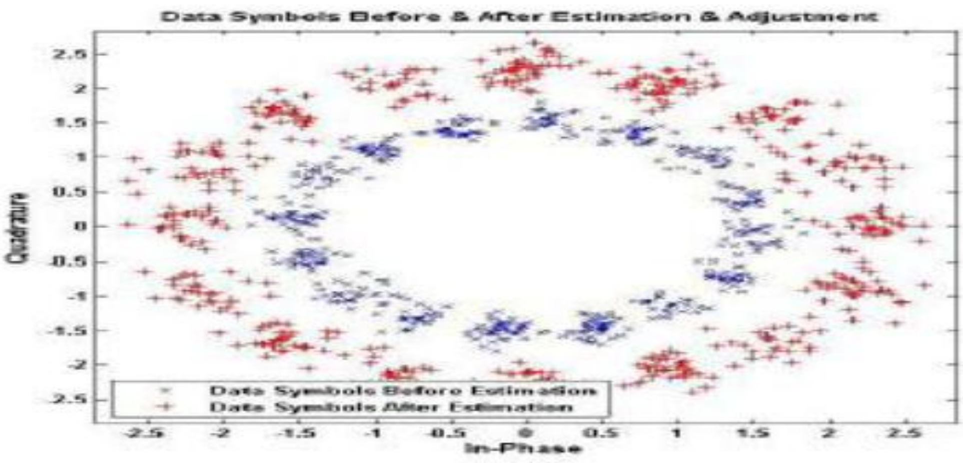


Figure 4-4 Equalization Results of 16PSK

4.5.5.2 MPSK Demodulator

The demodulator function was also built which brought about the demodulation process according to the type of PSK modulation, and the imaginary and the real values from the modulated data were evaluated.

The step sizes were computed according to the type of PSK scheme for both modulation and de-modulation.

4.5.6 CRC GENERATOR

The co-operative diversity scheme that was to be implemented in the project was chosen to be ***selective relaying***. In selective relaying, the relay node performs decode-and-forward operation on the message only if the received packet is correct. On the other hand, if the channel between the source and the relay has severe fading such that the received packets are corrupted, the relay idles.

The check that was placed was the **CRC Checksum**. The transmitted packets were appended with CRC at the transmission side and the CRC was checked at the receiver as an indication to whether the packet received was correct or not.

The following MATLAB command was used for this purpose

h = crc.generator (polynomial)

It constructs a CRC generator object H defined by the generator polynomial POLYNOMIAL.

Generator Method

Encoded = generate (h, msg) generates a CRC checksum for an input message using the CRC generator object H. It adds/appends the checksum to the end of MSG. The MSG, that is binary-valued, can be either a matrix or a column vector. If it is a matrix, then each column is considered to be a separate channel.

4.5.7 CRC Detector and Error Evaluator

At the relay and reception side, the CRC appended with the data packets is separated and the CRC checksum is evaluated using the same generator polynomial to check whether the packet is correct or not.

4.5.7.1 CRC Detection Evaluation at Relay

In the selective relaying the decision power relies with the relay. The relay has to decide which packets to forward and which should be discarded owing to being erroneous. The relay is to forward only those packets that have been received correctly.

The CRC is checked at the relay and the packets received are checked for errors. If the packet is erroneous, owing to the impairments in the simulated communication path, then the relay idles. Otherwise if the CRC is properly evaluated and matched, then the relay transmits the packets towards the reception side.

4.5.7.2 CRC Detection Evaluation at Receiver

The CRC checksum is also done at the reception side to check the packets for errors. The corrupted packets are kept track of and are added in the corrupted packets, which in the end provides the PER of the whole transmission.

The following MATLAB command is used for CRC evaluation and error detection

h= crc.detector (polynomial)

It constructs a CRC detector object H defined by the generator polynomial POLYNOMIAL

4.5.7.3 Detection Method

[OUTDATA ERROR] = DETECT (H, INDATA) detects transmission errors in the encoded input message INDATA by regenerating a CRC checksum using the CRC detector object H. The detector object H then compares the regenerated checksum with the checksum appended/added to INDATA. The binary-valued INDATA can be either a matrix or a column vector. If it is a matrix, each column is considered to be a separate channel. INDATA is identical to the output message OUTDATA, except that output message has the CRC checksum stripped off. ERROR is a 1xC logical vector, where C is the number of channels in INDATA, indicating if the encoded message INDATA has errors. An ERROR value of a 1 indicates errors and value of 0 indicates no errors.

Chapter 5

GNU Radio Installation

The list of available packages contains:

1. [gnuradio-core] The main library, includes the underlying runtime system and most of the hardware independent signal processing blocks.

2. [gnuradio-examples] Simple examples that exercise GNU Radio.

3. [gr-audio-alsa] Support for sound cards using the ALSA (Preferred on GNU/Linux).

4. [gr-audio-jack] Support for sound cards using the JACK (experimental: GNU/Linux).

5. [gr-audio-oss] Support for sound cards using the Open Sound System.

6. [gr-audio-portaudio] Preliminary support for sound cards using the portaudio V19 library (work-in-progress).

7. [gr-audio-osx] Support for OSX audio

8. [gr-audio-windows] Support for audio-sink in windows.

9. [gr-wxgui] GUI framework built on wxPython, including blocks for displaying realtime FFT and oscilloscope.

10. [gr-gsm-fr-vocoder] GSM 06.10 13kbit/sec voice encoder/decoder.

11. [gr-radio-astronomy] Radio astronomy application files

12. [gr-trellis] Implementation of trellis-based encoding and decoding algorithms

13. [gr-howto-write-a-block] Documentation and examples of how to write a new signal processing block in GNU Radio. Can be downloaded from ftp://ftp.gnu.org/gnu/gnuradio/gr-howtowrite- a-block-3.0.2.tar.gz

14. [usrp] The non-GNU Radio specific part of the USRP code base, containing the host libs, firmware and FPGA code.

15. [gr-usrp] The glue that ties the usrp library into GNU Radio.

5.1 Building GNU Radio

External dependencies are listed below:

With the exception of SDCC, the following GNU/Linux distributions are known to come with all required dependencies pre-packaged: SuSE 10.0 , Ubuntu 6.06, Fedora Core 2, 3, 4, 5, and 6. The required packages may be contained on your installation CD/DVD, or may be loaded over the Internet.

The specifics vary depending on particular GNU/Linux distributions. On systems using pkgsrc (e.g. NetBSD/Dragonfly), ―build metapackages/ gnuradio‖, will build a previous release and force installation of the dependencies. Then pkg_delete the usrp packages and gnuradio packages, which will leave the dependencies.

See the wiki at **http://gnuradio.org/trac/wiki**

Installing from the SVN repository:

The new repository organization simplifies a lot of the build system. You no longer need to go into the individual directories and compile separately.

To check out the latest code from the trunk, enter this on the command line:

```
$ svn co http://gnuradio.org/svn/gnuradio/trunk gnuradio
```

To instead checkout the latest stable release code, type this on the command line:

$ svn co http://gnuradio.org/svn/gnuradio/branches/releases/3.0 gnuradio

After ensuring that the dependencies specified in the top-level README are met. Most GNU/Linux systems come packaged with the earlier listed dependencies. You may need to install them off your install CD/DVD or over the Internet.

To compile, there are 5 steps. Start by cd'ing to the gnuradio directory, and then complete the following commands:

```
$ ./bootstrap # Do NOT perform this step if you are building from a tarball.

$ ./configure

$ make

$ make check

$ sudo make install
```

This will perform all configuration checks and select for test, build, and installation of all components that pass.

Some instructions for installing the tarball packages in Fedora Core:

[gnuradio-core]:

Prerequisites:

1. The "autotools"

autoconf 2.57 or later

automake 1.7.4 or later

libtool 1.5 or later

There's a good chance it also has automake-1.7, if your system has automake-1.4, or try:

```
$ man update-alternatives
```

for info on how some distributions support multiple versions.

2. pkgconfig 0.15.0 or later

Visit http://www.freedesktop.org/Software/pkgconfig

From the web site: pkgconfig is a system for managing library compile/link flags that works with automake and autoconf. It replaces the ubiquitous *-config scripts you may have seen with a single tool.

3. FFTW 3.0 or later (http://www.fftw.org)

When building FFTW, you must use the –enable-shared and –enable-single configure options. You should also use either the --enable-3dnow or --enable-sse options if you're on an Athlon or Pentium respectively.

4. Python 2.3 or later, visit http://www.python.org

Python 2.3 or later is now required. If your distribution distributes the python into a bunch of separate RPMS including libpython or python-devel, you'll most likely need those too.

```
./configure --enable-unicode=ucs4

make clean

make
```

make install

5. Numeric python library http://numeric.scipy.org

This library provides a high performance array type for Python.

http://sourceforge.net/project/showfiles.php?group_id=1369&package_id=1351

6. The Boost C++ Libraries http://www.boost.org

GNU Radio uses the Smart Pointer library. Fedore Core 2 has a package for this library, boost-devel-1.31.0-7. Otherwise download the source and follow the build instructions, which are a bit different from the normal ./configure && make

7. cppunit 1.9.14 or later.(http://cppunit.sourceforge.net)

Unit testing framework for C++.

Some of the other facilities which may be required are:

8. SWIG – Simplified Wrapper and Interface Generator

http://www.swig.org/

These versions are known to work: 1.3.23, 1.3.24, 1.3.25, 1.3.27, 1.3.28, 1.3.29

9. SDCC – Small Device C Compiler

http://sdcc.sourceforge.net/

If you don't have a USRP, ignore this.

Optional, but nice to have:

10. wxPython. Python binding for the wxWidgets GUI framework.

Use version 2.5.2.7 or later. As a last option, build it from source, although it is not recommended.

http://www.wxpython.org

Also, as noted on the website, GNU Radio experiences bugs in certain versions of g++ 3.3.x on the x86 platform. If you are using g++ 3.3 and make check fails, please either downgrade to 3.2or upgrade to 3.4. Both are known to work.

[gnuradio-examples]

Set your PYTHONPATH environment variable so that the GNU Radio toolkit and optional packages can be found by python. PYTHONPATH should include the path of the local site-packages directory.

If the above packages were installed using the default prefix (/usr/local) and you are using python 2.3, this may work:

```
$ export PYTHONPATH=/usr/local/lib/python2.4/site-packages
```

You may want to add this to your ~/.bash_profile or similar file.

Once PYTHONPATH is set, you should be able to run any of the examples for which you have the required i/o devices.

To ensure that your setup is sane, try the following command:

```
$ python
>>> from gnuradio import gr
```

If this works, your PYTHONPATH is set correctly.

[gr-audio-alsa] & [gr-audio-oss]

These two packages are audio packages which are needed to interface with the audio device on your computer.

[gr-howto-write-a-block]:

The documentation with descriptions of how to write signal processing blocks for GNU Radio.

If you have gotten doxygen installed and provided the --enable-doxygen configure option, the build process creates documentation for the class hierarchy.

Point your browser at gnuradio-core/html/index.html

The online version can be found at :

http://www.gnu.org/software/gnuradio/doc/howto-write-a-block.html.

The online documentation for GNU Radio with descriptions of all of the modules, class hierarchy and file list can be found at :

http://www.gnu.org/software/gnuradio/doc/

[gr-wxgui]:

GUI framework built on wxPython, includes blocks for displaying realtime FFT and oscilloscope. The modules Numerical Python and wxPython also need to be installed.

http://wiki.wxpython.org/index.cgi/Getting_Started is a good place to look for information about wxPython and its installation guidelines.

A short summary of the instructions is explained below:

a. **Install Python**

(http://www.python.org/download/releases/2.4.4/) (This is the version I used)

b. **Install wxPython:** (requires glib and gtk+ libraries installed -

http://www.wxpython.org/download.php#prerequisites)

Download the source code of the last wxPython release: wxPython website. The website has listings for different platforms. There is no separate version for Fedora Core 3, but instead use the version for Fedora Core 2 and it works.

The default installation happens in ―/usr/lib/python<version>/sitepackages/‖.

But the Python installation happens in ―/usr/local/lib/python<version>/site-packages/‖. This may prevent the ―wx‖ module from being accessed correctly.

To solve the wx module access problem, in ―/usr/local/lib/python<version>/site-packages/‖ create the path file ―wx.pth‖ and in that file give the full path to the wxPython installation i.e

―/usr/lib/python<version>/site-packages/wx-<unicode-version>/‖. This will enable python to find the wx module.

c. **To test the installation with a small program in python**:

After importing wxPython GUI, we instantiate a new wxFrame and a new wxPySimpleApp. A frame in wxPython is a window with its reduction, titlebar, and close buttons, etc... We make this Frame appear by "showing" it. Eventually, we start with the MainLoop of application whose role is to handle the events.

d. **Install Numerical Python**: Numerical Python adds a fast array facility to the Python language (http://numeric.scipy.org/).

Download it from **http://sourceforge.net/projects/numpy** The install requires no changes to the path file as it happens within the python installation directory,.

To verify, look under ―/usr/local/lib/python<version>/site-packages/‖ and ensure that the files ―Numeric.pth‖ and ―wx.pth‖ exist and contain valid paths.

http://www.pfdubois.com/numpy/html2/numpy.html gives a nice reference for programming with Numerical Python.

e. **Install NumArray:**

http://www.stsci.edu/resources/software_hardware/numarray

Download it from Sourceforge Numarray Download Page.

[usrp]:

The non-GNU Radio specific part of the USRP code base, which contains the host libs, firmware and FPGA code.

[gr-usrp]:

The glue that ties the usrp library into GNU Radio. The USRP hardware needs to be setup and checked for correct operation.

http://comsec.com/wiki?UsrpInstall

gives a quick walkthrough to check the working of USRP board.

USRP associated files and hardware schematics can be obtained with the following command:

$ svn co http://gnuradio.org/svn/usrp-hw/trunk usrp-hw.

5.2 Running Examples

To run examples, ―cd‖ into the directory ―gnuradioexamples/ python/audio‖.

You should find some *.py files which can be run as executables. Running ―./dialtone.py‖, should produce a dial tone of frequency 32KHz.

Also in the "usrp" subdirectory. Running ./usrp_oscope.py, should bring up something that looks like an oscilloscope. Resize it so you can read the labels on the buttons. It looks pretty dull until you get the triggering working -- set it from "Pos" to "Auto". Then you'll start seeing a bunch of noise on the screen -- a red line and a green line. If you attach a piece of wire to the inner conductor of your "RX-A" input on your "RXA" daughterboard, the green line will start to wiggle a lot.

Chapter No. 6

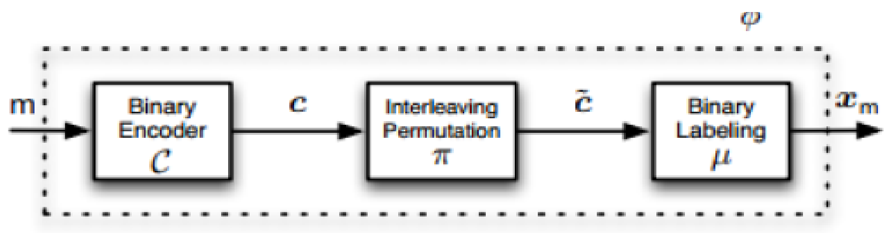
Bit Interleaving Coded Modulation

BICM has been regarded as a pragmatic yet powerful scheme to achieve high data rates with general signal constellations. Nowadays, BICM is employed in a wide range of practical communications systems, such as DVB-S2, Wireless LANs, DSL, WiMax, the future generation of high data rate cellular systems (the so-called 4th generation). BICM has become the de-facto standard for coding over the Gaussian channel in modern systems.

6.1 Abstract

To ensure data fidelity, a number of random error correction codes (ECCs) have been developed. ECC is, however, not efficient in combating bursts of errors, i.e., a group of consecutive (in one-dimensional (1-D) case) or connected (in two- and three-dimensional (2- D and 3-D) case) erroneous code symbols owing to the bursty nature of errors. Interleaving is a process to rearrange code symbols so as to spread bursts of errors over multiple code-words that can be corrected by ECCs. By converting bursts of errors into random like errors, interleaving thus becomes an effective means to combat error bursts. In this article, we first illustrate the philosophy of interleaving by introducing a 1-D block interleaving technique. Then multi-dimensional (M-D) bursts of errors and optimality of interleaving are defined. The fundamentals and algorithms of the state of the art of M-D interleaving—the t-interleaved array approach by Blaum, Bruck and Vardy and the successive packing approach by Shi and Zhang— are presented and analyzed. In essence, a t-interleaved array is constructed by closely tiling a building

block, which is solely determined by the burst size t. Therefore, the algorithm needs to be implemented each time for a different burst size in order to maintain either the error burst correction capability or optimality. Since the size of error bursts is usually not known in advance, the application of the technique is somewhat limited. The successive packing algorithm, based on the concept of 2×2 basis array, only needs to be implemented once for a given square 2-D array, and yet it remains optimal for a set of bursts of errors having different sizes. The performance comparison between different approaches is made. Future research on the successive packing approach is discussed. Finally, applications of 2-D/3-D successive packing interleaving in enhancing the robustness of image/video data hiding are presented as examples of practical utilization of interleaving.

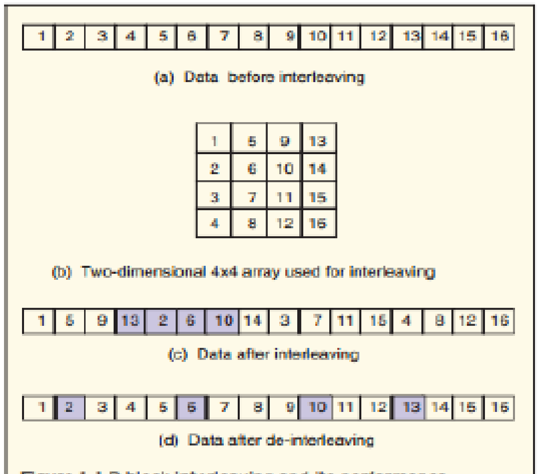


6.2 Introduction

In data manipulation and transmission, errors may be caused by a variety of factors including noise corruption, limited channel bandwidth, and interference between channels and sources. It is well known that many error correction codes (ECCs) have been developed to correct errors in order to ensure data fidelity. In doing so, redundancy has been added to ECC, resulting in what is known as random error correction codes. There are basically two different types of ECCs. One is known as block codes, the other as convolutional codes. The frequently used block codes are often denoted by a pair of two integers, i.e., (n, k), and one block code is completely defined by 2k binary sequences, each is an n-tuple of bits, known as code-word.

6.3 Philosophy of interleaving

The 1-D interleaving techniques have been well documented in error correction coding texts. The main idea is to mix up the code symbols from different code-words so that when the code-words are reconstructed at the receiving end error bursts encountered in the transmission are spread across multiple code words. Consequently, the errors occurred within one code-word may be small enough to be corrected by using a simple random error correction code. This can be seen clearly from a simple example. Consider a code in which each code-word contains four code symbols. Furthermore, the code possesses the random error



(a) Data before interleaving
(b) Two-dimensional 4x4 array used for interleaving
(c) Data after interleaving
(d) Data after de-interleaving

Figure 1. 1-D block interleaving and its performance.

correction capability. Without loss of generality, we assume the one-random-error-correction capability, i.e., any single code symbol error occurred in one code-word can be corrected. Suppose there are 16 symbols, which correspond to four code-words. That is, code symbols from 1 to 4 form a code-word, from 5 to 8 another code word and so on.

One of the 1-D interleaving procedures, known as block interleaving, first creates a 4×4 2-D array, called block interleaver as shown in Figure 1. The 16 code symbols are read into the 2-D array in a column-by-column (or row-by-row) manner. The interleaved code symbols are obtained by writing the code symbols out of the 2-D array in a row-by-row (or column by-column) fashion. This process has been depicted in Figure 1 (a), (b), and (c). Now take a look at how this interleaving technique can correct error bursts. Assume a burst of errors involving four consecutive symbols as shown in Figure 1 (c) with shades. After de-interleaving as shown in Figure 1 (d), the error burst is effectively spread among four code-words, resulting in only one code symbol in error for each of the four code-words. With the one-random-error-correction capability, it is obvious that no decoding error will result from the presence of such an error burst. This simple example demonstrates the effectiveness of 1-D interleaving technique in combating 1-D bursts of errors, i.e., how the interleaving spreads code symbols over multiple code words so as to convert a burst of errors occurred with the interleaved array into random-like errors in the deinterleaved array. In other words, the pair of interleaving and de-interleaving can equivalently convert a bursty channel into a random-like channel. Consequently, random error correction codes can be used efficiently to correct bursts of errors.

Chapter 7

Multi-carrier (OFDM) Physical Layer Design

OFDM is a frequency-division multiplexing (FDM) scheme utilized as a digital multi-carrier modulation method. A large number of closely-spaced orthogonal sub-carriers are used to carry data. The data is divided into several parallel data streams or channels, one for each sub-carrier. Each sub-carrier is modulated with a conventional modulation scheme (such as quadrature amplitude modulation or phase shift keying) at a low symbol rate, maintaining total data rates similar to conventional single-carrier modulation schemes in the same bandwidth. OFDM has developed into a popular scheme for wideband digital communication.

The primary advantage of OFDM over single-carrier schemes is its ability to cope with severe channel conditions — for example, attenuation of high frequencies in a long copper wire, narrow band interference and frequency-selective fading due to multipath — without complex equalization filters.

7.1 Transmitter:

Like the single carrier system, OFDM transceiver will also be explained in two parts, transmitter and receiver. The figure 7.1 shows the flow graph of an OFDM transmitter. In the subsequent sections the details of each block will be explained.

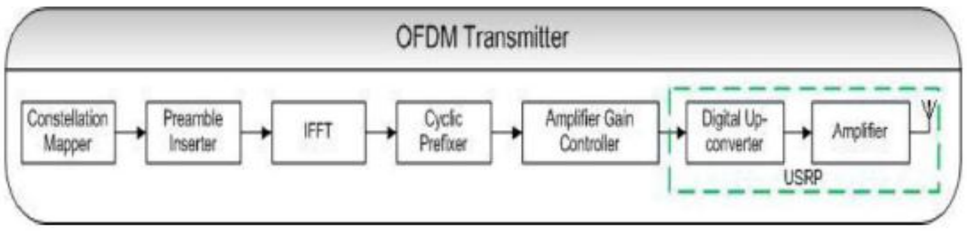
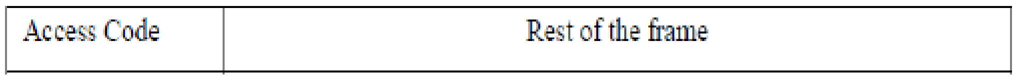


Figure 7.1 The OFDM transmit thread's flowgraph

7.1.1 Bits to Symbol Mapping:

For the time-being, let‘s assume that the data source is a file. This file can be a text file, image file, or even the handler to the drivers of a TUN/TAP device which will be explained in Chapter 10. Further assume that after reading several bytes of data from file, the proper framing has been done. Before a frame is handed over to physical layer all the bits in it are recognized as a python string. The framing structure will be explained in Chapter 8 but we need to emphasis one field of the frame for now i.e. ***access code***.



Access Code	Rest of the frame

The access code is a 8 bytes random bit pattern which indicates the beginning of the frame. At the receiver we will need to look for this bit stream in order to synchronize every frame.

In GNU radio:

When a frame is handed over to physical layer as a string it is converted into a new data type, i.e. GNU radio message, using function ***gr.message_from_string***. GNU radio message is not different from python string in anyway except that GNU radio recognizes the string frame as one message and all GNU radio blocks processes the message as a whole.

The block ***ofdm_mapper*** is used to map all the bits in the message (frame) onto complex constellation points. This block has two outputs:

- The first output is a stream of vectors of size equal to that of FFT size. This preprocessing helps in the operation of IFFT block where the input must be a vector of this format.
- The second output is a control signal, which consists of a stream of python shorts, each corresponding to one vector at first output. The value of short is 1 for the first constellation vector of the frame and 0 for all other vectors. This control signal is required for the following block i.e. preamble inserter.

7.1.2 OFDM preamble generator:

In OFDM systems, a preamble is inserted before every frame. This preamble is used to perform frame synchronization at receiver. The preamble is nothing more than a ***known*** constellation vector, making up a known OFDM symbol. So after preamble insertion an OFDM frame will be mapped onto constellation vectors like this :

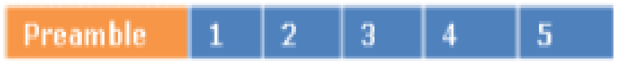


Figure 7.2 Five data constellation vectors following preamble vector

In GNU radio:

In GNU radio ***gr.ofdm_insert_preamble*** is used to perform this task. The details of this block are given below:

Block Name	**ofdm_insert_preamble ()**
Description	This block inserts "pre-modulated" preamble symbols before each frame. It has two input ports and two output ports. Input 1: stream of vectors of type gr_complex [fft_length]. These are the modulated symbols of the payload. Input 2: stream of characters/shorts. The LSB indicates whether the corresponding symbol on input 1 is the first symbol of the payload or not. It's a 1 if the corresponding symbol is the first symbol; otherwise 0. Output 1: stream of vectors of gr_complex [fft_length] These include the preamble symbols and the frame symbols. Output 2: stream of char. The LSB indicates whether the corresponding symbol on input 1 is the first symbol of a packet (i.e., the first symbol of the preamble.) It's a 1 if the corresponding symbol is the first symbol, otherwise 0.
Usage	**gr.ofdm_insert_preamble(fft_length, preamble)**
Parameters	fft_length : FFT length preamble : vector of complex vectors

7.1.3 IFFT:

The next step is to feed these constellation vectors to the input of IFFT block. The figure 7.3 shows the IFFT process. The input from xo to xN−1represents the constellation vector where N is the FFT size. The same way si represents the *i*th sample of the IFFT output.

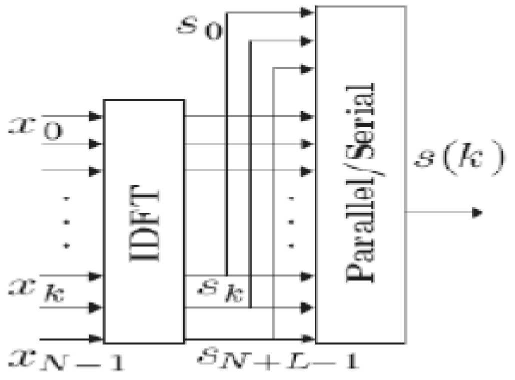


Figure 7.3 Process of IFFT and cyclic prefix

GNU Radio Block:

Block Name	**fft_vcc ( )**
Description	This block computes both forward or reverse FFT, complex vector in / complex vector out.
Usage	**gr.fft_vcc (fft_size, forward, window, shift=false)**
Parameters	fft_size : integer forward : bool True for forward FFT, False for inverse FFT window : window vector, shift : bool True or false

7.1.4 Cyclic Prefix insertion:

Because wireless communications systems are susceptible to multi-path channel reflections, a cyclic prefix is added to reduce ISI. A cyclic prefix is a repetition of the first section of a symbol that is appended to the end of the symbol. In addition, it is important because it enables multi-path representations of the original signal to fade so that they do not interfere with the subsequent symbol.

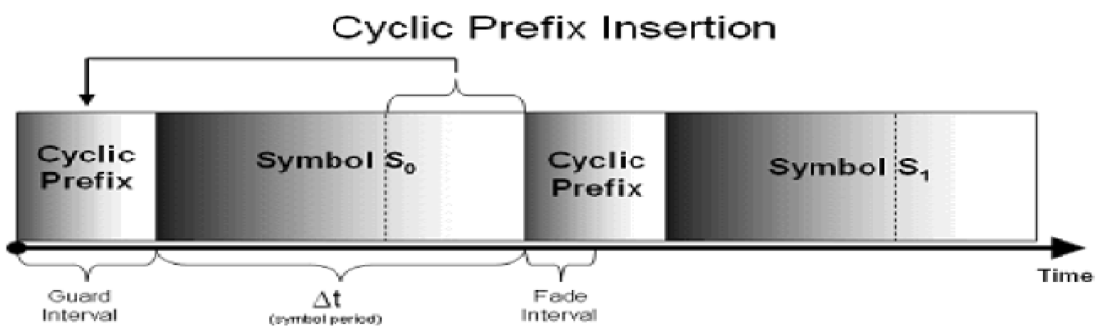


Figure 7.4 Cyclic Prefix Insertion

Cyclic prefixes are also used in OFDM in order to combat multipath by making channel estimation easy. However the process of channel estimation and equalization will be explained later.

In GNU radio:

The block ***ofdm_cyclic_prefixer*** adds a cyclic prefix vector to an FFT size long OFDM symbol (vector) and converts vector to a stream of **FFT size + cyclic prefix** length vector.

7.1.6 Digital to Analog Conversion and Up-conversion:

Digital to Analog conversion and up-conversion in OFDM systems is performed exactly the same way as it was performed in single carrier systems.

7.2 Receiver:

The figure below shows the block diagram of an OFDM receiver. The USRP portion behaves exactly the same way as explained for single carrier system however rest of the design is different.

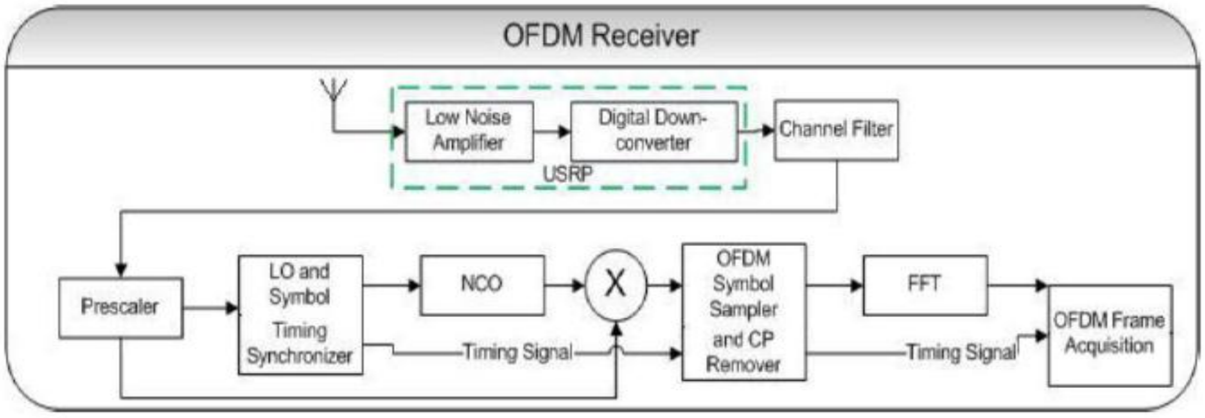


Figure 7.5 OFDM Receiver flow graph

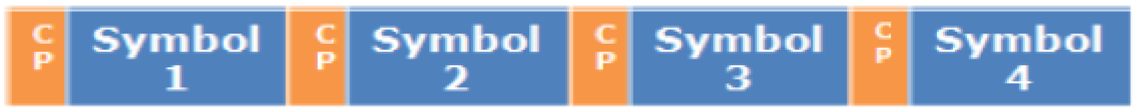
7.2.1 Down-conversion and Decimation:

Down-conversion and Decimation have been explained in section 6.2.1. Once the signal has been down-converted and decimated by FPGA in USRP, the received baseband signal is fed to CPU via USB. We use ***usrp.source_c*** block to get the baseband signal from USRP. The immediate task to accomplish is to perform synchronization of the received signal.

7.2.2 OFDM Synchronization:

There are two major aspects of synchronization in OFDM systems:

- One problem is the unknown time instant to start sampling a new OFDM symbol. Sensitivity to a timing offset is higher in multi-carrier systems than in single-carrier systems. When an OFDM frame is received there must be some mechanism to determine from where the frame starts in the received signal and from where every OFDM symbol starts in every frame. We will exploit the cyclic prefix of every symbol to accomplish this task.



- A second problem is the mismatch of the oscillators in the transmitter and receiver. The demodulation of a signal with an offset in the carrier frequency can cause large bit errors and may degrade the performance of a symbol synchronizer. It is therefore important to estimate this offset and minimize its impact .

In order to resolve the above two issues, we must perform timing and frequency synchronization. In this project, ML Estimation of Time and Frequency Offset in OFDM Systems by J van de Beeks has been used. This method is explained in the flow graph of Figure 7.6.

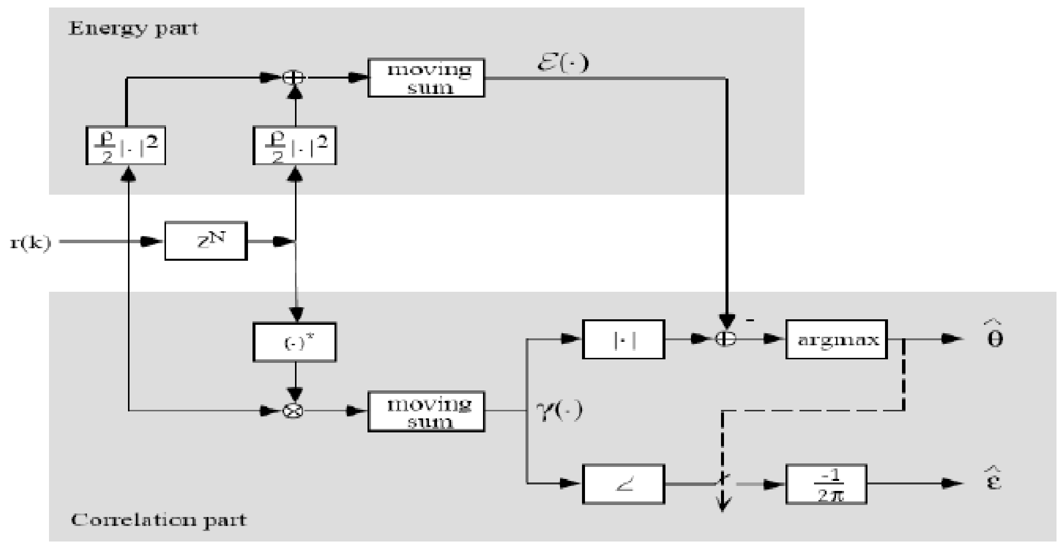


Figure 7.6 Block scheme of the ML Estimator.

We now consider two uncertainties in the receiver of this OFDM symbol. The uncertainty in the arrival time of the OFDM symbol is modeled as a delay in the channel impulse response, i.e., ±(k-Θ), where Θ is the integer-valued unknown arrival time of a symbol. The uncertainty in carrier frequency, which is due to a difference in the local oscillators in the transmitter and receiver, gives rise to a shift in the frequency domain.

Such behavior is modeled as a complex multiplicative distortion in the time domain,

$ej2\pi$ἑk/N, of the received data, where $\acute{\varepsilon}$ denotes the difference in the transmitter and receiver oscillators as a fraction of the inter-carrier spacing, N is the FFT size. Hence, the received data is

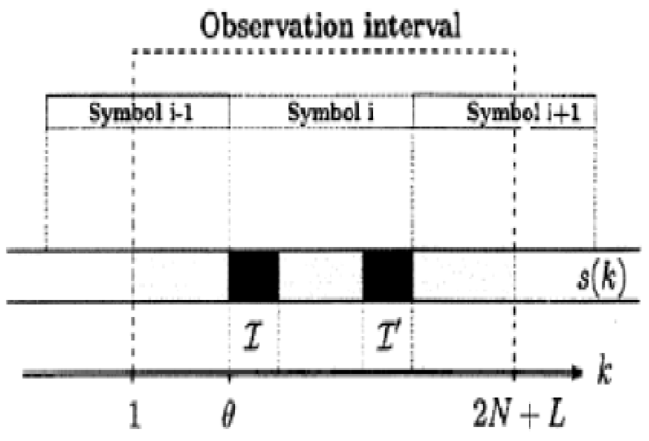


Figure 7.7 Structure of OFDM symbols with cyclically extended symbols, this 2N+L duration of sample contains unknown OFDM symbol.

Assume that we observe *2N +L* consecutive samples of (k), Figure 7.2, and that these samples contain one complete (N +L) sample OFDM symbol where *L* being CP length. The position of this symbol within the observed block of samples, however, is unknown because the channel delay Θ is unknown to the receiver.

The synchronization is performed by computing normalized gamma parameter given by:

$$\theta = arg\ max\ (\theta) - \rho \in (\theta)$$

Where,

$$\gamma(m) = \sum_{k=m}^{m+L-1} r(k)\, r^*(k+N)$$

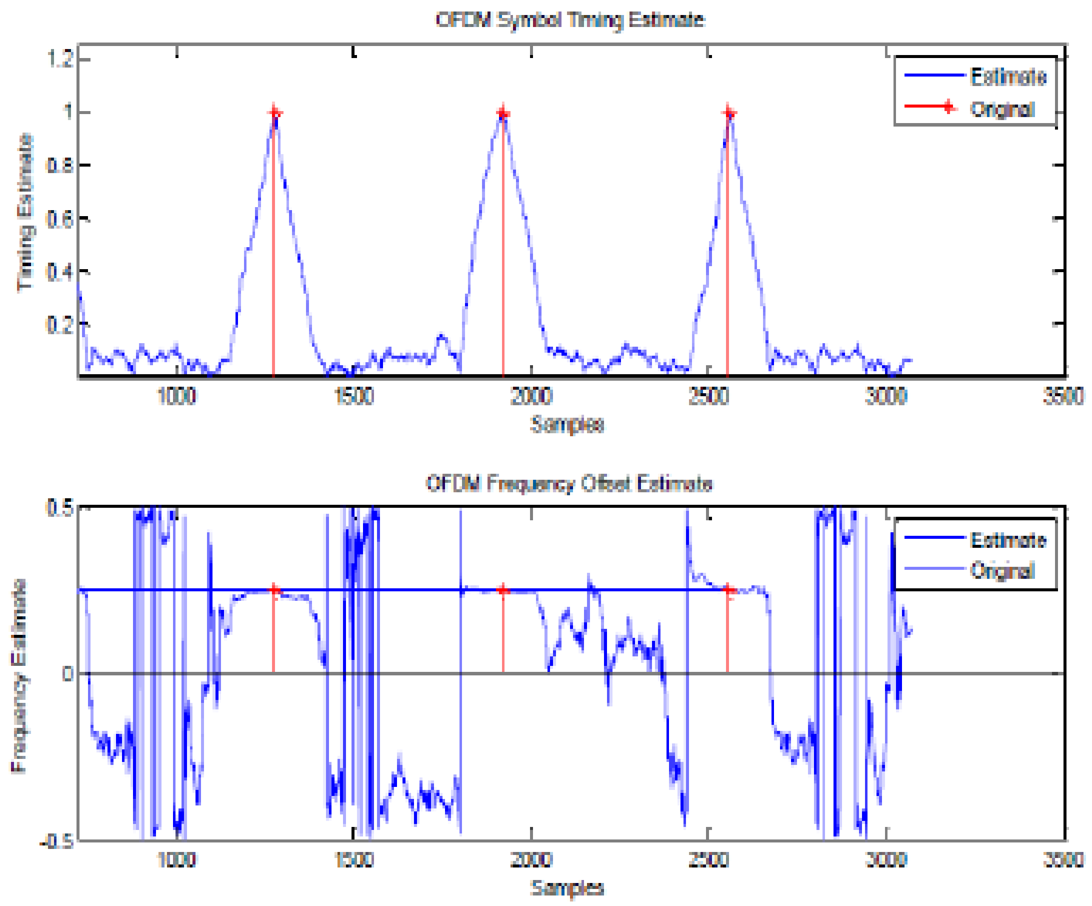


Figure 7.8 MATLAB simulation of ML estimator illustrating Timing and Frequency estimates.

The third aspect of synchronization is determining the beginning of an OFDM frame. Remember, on transmitting side a known OFDM symbol, called preamble, was sent at the beginning of every frame. At receiver, a preamble correlator will be used along with a peak detector at its output to generate the timing signal as shown in Figure 7.5.

7.2.3 Cyclic Prefix Removal

Cyclic prefix is inserted at the transmitter side and removed at receiver side. This removal makes the OFDM symbol become circularly convolved with channel and also removes the Inter-Block Interference (IBI). OFDM symbol size is limited to FFT size, so that FFT can be taken.

In GNU radio:

ofdm_sampler block samples the received signal at the beginning of every frame into vectors of symbol size. This block also removes the cyclic prefix (CP) at the beginning of every symbol and generates vectors of FFT size at its output which correspond to OFDM symbols in time domain with CP removed.

7.2.4 FFT:

Now we have OFDM symbols in time domain with CP removed and ready to be fed into FFT bock. In GNU radio the block ***fft_vcc*** is used to compute both FFT and IFFT. The output of this block is a frequency domain OFDM symbol consisting of a vector of FFT size whose elements are the constellation points.

7.2.5 Frequency Domain Equalization

Cyclic prefixes are also used in OFDM in order to combat multipath by making channel estimation easy. As an example, consider an OFDM system which has subcarriers.The OFDM symbol is constructed by taking the inverse discrete Fourier transform (IDFT) of the message symbol, followed by a cyclic prefixing. Let the symbol obtained by the IDFT be denoted by

$$\mathbf{x_0} = [x[0], x[1], \dots x[N-1]]^T$$

Prefixing it with a cyclic prefix of length $L-1$, the OFDM symbol obtained is:

$$\mathbf{x} = [x[N-L+1], \dots x[N-2], x[N-1], x[0], x[1], \dots x[N-1]]^T$$

Assume that the channel is represented using

$$\mathbf{h} = [h_0, h_1, \dots h_{L-1}]^T$$

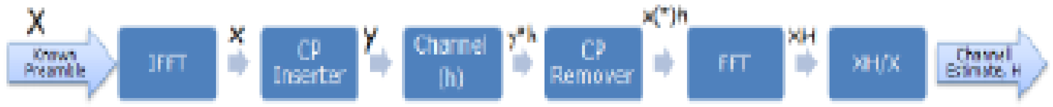
Then, after convolution with the channel, which happens as

$$y[m] = \sum_{l=0}^{L-1} h_l x[m-l] \quad 0 \le m \le N-1$$

which is circular convolution, as $x[m-k]$ becomes $x[(m-k) \mod N]$. So, taking the Discrete Fourier Transform, we get

$$Y[k] = H[k] \cdot X[k]$$

where $X[k]$ is the discrete Fourier transform of $\mathbf{x}$. Thus, a multipath channel is converted into scalar parallel sub-channels in frequency domain, thereby simplifying the receiver design considerably. The task of channel estimation is simplified, as we just need to estimate the scalar coefficients $H[k]$ for each sub-channel and once the values of $\{H[k]\}$ are estimated, for the duration in which the channel does not vary significantly, merely multiplying the received demodulated symbols by the inverse of $H[k]$ yields the estimates of $\{X[k]\}$ and finally an inverse discrete Fourier transform leads to the estimate of actual symbols $[d_0, d_1, \dots d_{N-1}]^T$



Chapter No. 8

Physical Layer Design for Cooperative Communications on SDR

8.1 Transmitter Modules

8.1.1 Data packetizing

The scheduler accepts string type data from a file. After reading several bytes of data (the number of bytes per packet can be varied) all the bits in it are recognized as python strings. A two byte packet number is appended to the string data for packet identification. It is then passed on to a packetizing function from the GNU radio library ―packet_utils.make_packet‖ that pads the data and appends an access code to it. The access code is an 8 bytes random bit pattern which indicates the beginning of the frame. At the receiver we will need to look for this bit stream in order to synchronize every frame.

8.1.2 USRP data

The packetizing function passes on the packed string data on to the USRP data formatter. This function converts the string data into a format that is understood by the SDR platform, the USRP. GNU radio message is not different from python string in anyway except that GNU radio recognizes the string frame as one message and all GNU radio blocks processes the message as a whole. For the purpose it uses the GNU radio function **―gr.message_from_string**.

8.1.3 Message source

There is a block named ***―gr.message_source”*** in GNU radio which converts the messages fed to it into bit streams at its output. The module outputs data at a constant rate. The module acts as a buffer that has a constant output rate irrespective of the processor resources at hand. This helps in elimination of data preprocessing variations in rate due to processor loading etc.

8.1.4 Differential Encoder

In digital communications, differential coding is a technique used to make data to be transmitted to depend not only from the current bit (or symbol), but also from the previous one. Differential encoder works on following principle

$$y_i = y_{i-1} \oplus x_i$$

Assuming that xi is a bit intended for transmission and yi is a bit actually transmitted (differentially encoded). Differential encoding is used to deal with 90 degrees local carrier phase ambiguity which remains there even after carrier synchronization. ―diff_encoder‖ function from the GNU radio provides this functionality.

8.1.5 Waveform Generation

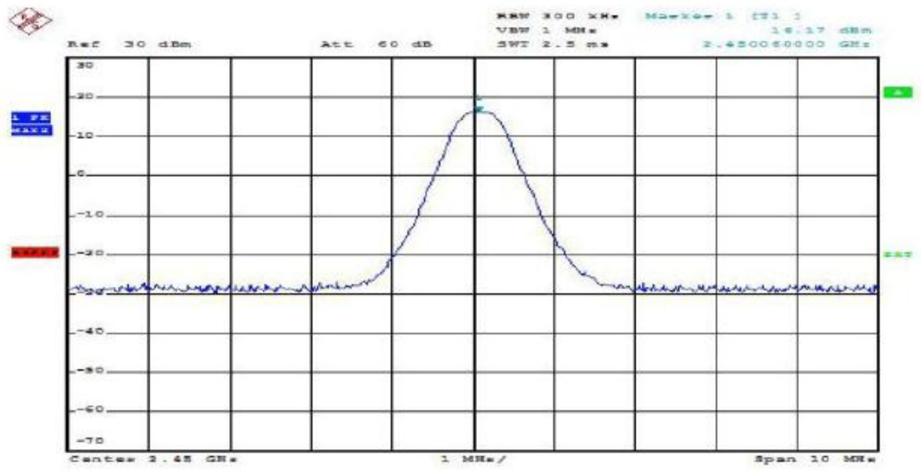
The differentially encoded data is then passed on to the waveform generation function which line codes the data according to NRZ (non return to zero) coding method.

8.1.6 Filtering and Modulation

The NRZ waveform is Gaussian filtered for transmission. In digital modulation due to its ability to minimize inter-symbol interference (ISI) the waveform is generally filtered. This block pulse shapes the mapped symbols. The Gaussian filtered waveform is gmsk modulated. Figure 8.1 shows a constellation diagram of the transmitted signal for gmsk which is the selected modulation scheme for the developed cooperative network.

8.1.7 Amplifier Gain Control

In USRP transmit thread there is a 14-bit digital to analog converter. In order to use the full range of our output amplifier we use amplifier gain control. The amplifier gain control amplifies input signal by multiplying the signal with suitable number or value (usually ¾ of maximum). This value dictates also the output power of transmitted signal.



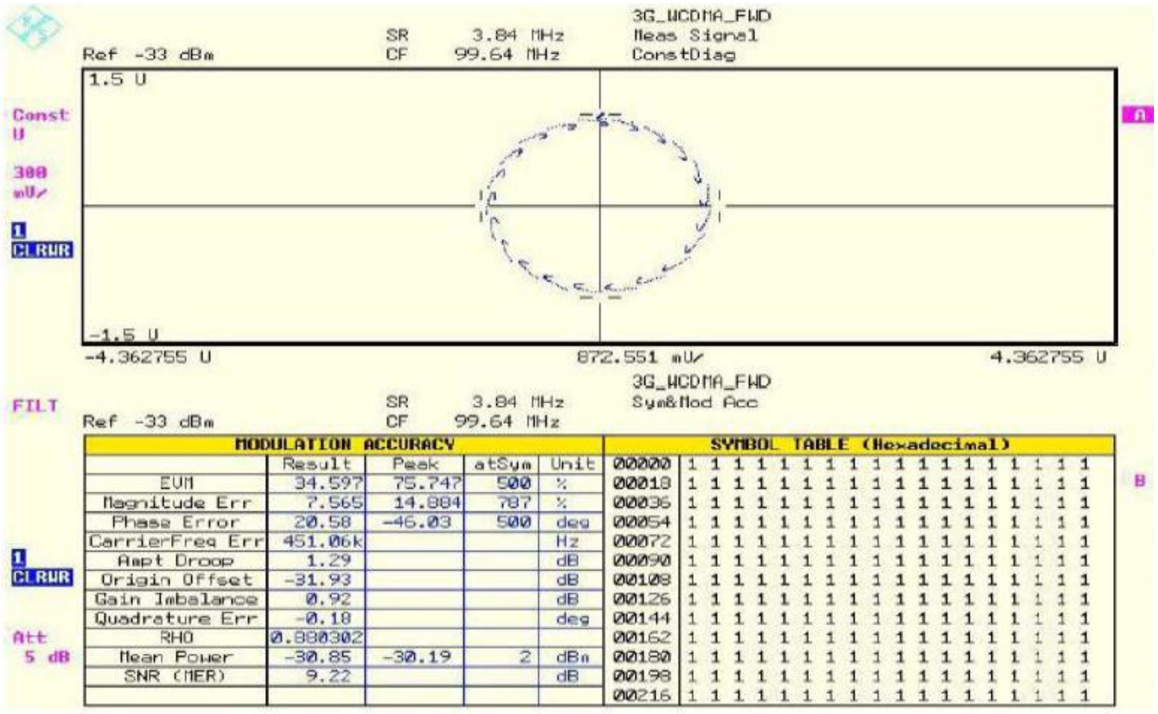
67

Figure 8-1 GMSK Modulated Input Signal Constellation as seen on a Spectrum Analyzer

"multiply_const" block takes the signal from its input port and multiplies it against the given number and makes the signal scaled at output port.

Connect to the output of Amplifier Gain Controller the ***gr.usrp_sink*** block with appropriate arguments, the baseband signal will be DAC_ed , up-converted, amplified and transmitted by USRP. ***usrp.sink_c*** block is used to port the baseband signal to USRP. This function takes in arguments the daughter card to port the signal to, interpolation rate of DAC and block size of the data chunks to be passed between CPU and USRP via USB.

A typical signal that is transmitted by the USRP is shown in Figure 8.2 with an RF end capable of working in the ISM (2400MHz) band. The signal is centered at 2.45GHz with a spectral bandwidth of around 3MHZ. The power transmitted is displayed in the upper right corner which is around 16dB.



MODULATION ACCURACY				
	Result	Peak	atSym	Unit
EVM	34.597	75.747	500	%
Magnitude Err	7.565	14.884	787	%
Phase Error	20.58	-46.03	500	deg
CarrierFreq Err	451.06k			Hz
Ampt Droop	1.29			dB
Origin Offset	-31.93			dB
Gain Imbalance	0.92			dB
Quadrature Err	-0.18			deg
RHO	0.880302			
Mean Power	-30.85	-30.19	2	dBm
SNR (MER)	9.22			dB

SYMBOL TABLE (Hexadecimal)																		
00000	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00018	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00036	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00054	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00072	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00090	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00108	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00126	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00144	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00162	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00180	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00198	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
00216	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 8-2 Input Signal- Power and Bandwidth Characteristics

8.2 Receiver Modules

8.2.1 Amplifier

Low- noise amplifier (LNA) is a special type of electronic amplifier or amplifier used in communication systems to amplify very weak signals that are captured by an antenna. LNA is usually located very close to the antenna, making the losses in the feed line less critical. It is necessary for an LNA to boost the desired signal power while adding as little noise and distortion as possible so that the retrieval of this signal is possible in the later stages in the system.

8.2.2 Combining

The destination hosts multiple antennas to create a MIMO effect at the receiver. The antennas are hosted on a single radio platform making it a collocated antenna receiver system. The signal streams received at the antennas are combined using the equal gain combining (EGC) for MIMO systems. The resultant signal is the weighted averaged signal of the two streams at the antennas. Due to the distributed nature the amount of spatial gains provided by cooperative MIMO systems is much higher than conventional collocated antenna MIMO systems. Thus cooperative systems are a means of providing high spatial diversity gains to communication systems.

The receiver accepts any one of the two similar packets reaching via any of the two independent channels after combining the signal streams. The signal steams are weighted with a constant ratio. The ratio should represent the average channel quality and therefore should not take account of temporary influences on the channel due to fading or other effects. The EGC can be expressed as

$$y_d[n] = \sum_{i=1,2..k} w_i \cdot y_i[n]$$

where di denotes weighting of the incoming signal yi. Using one relay station, the equation simplifies to

$$y_d[n] = w_{s,d} \cdot y_{s,d}[n] + w_{s,r,d} \cdot y_{r,d}[n]$$

8.2.3 Down Conversion and Decimation

In digital signal processing, a digital down-converter (DDC) converts a signal centered at an intermediate frequency (IF) to a baseband complex signal centered at zero frequency. In addition to down conversion, DDC‗s typically decimate to a lower sampling rate, which allows the follow-on signal processing by lower speed processors. A DDC consists of three subcomponents; a Direct Digital Synthesizer (DDS) which generates a complex sinusoid at the intermediate frequency, a pair of multipliers to convert (in quadrature) from IF to baseband, and a pair of low-pass filters and decimators.

8.2.4 Pre-scaler

When the signal at the input of ADC uses its full input range the output digitized signal has maximum amplitude of 16384. Scaling at this stage is necessary to normalize the full range of input signal with this maximum values of that the detection process is easy. The block used for pre- scaler is ***multiply_const*** which performs the adequate scaling of input stream so that signal is brought back in to range.

8.2.5 Symbol Decoder

Once the synchronization has been performed, the only job left is decision making on soft symbols. In symbol detection the constellation ***de-mapper*** takes received constellation points as an input, and outputs the corresponding bits.

8.2.6 Access Code Correlator

After decoding all the symbols, a bit stream is obtained. The next task is to extract frames out of this bit stream. This is done by using the access code correlator which keeps on looking for the access-code bit-stream with certain threshold (about 80-90%).

8.2.7 Demodulation and Differential Decoding

The data is passed on to the demodulator block. This is done using the gmsk demodulator block. In digital communications, differential coding is a technique used to make data to be transmitted to depend not only from the current bit (or symbol), but also from the previous one this helps in increased reliability of received data.

$$x_{i+1} = y_{i+1} \oplus y_i$$

8.2.8 Payload Conversion

The data from the demodulator block is passed on to the payload conversion block that transforms the packed data to string data. This string data is the actual data that is actually transmitted from the transmitter module.

8.3 Relay Module

The relay module is composed of basically the transmitter and receiver modules. The relay transceiver switches between transmit and receive modes.

An account of the network node functionality models created in software radios in presented in the form of flow graphs in Figures 8-3 to 8-5.

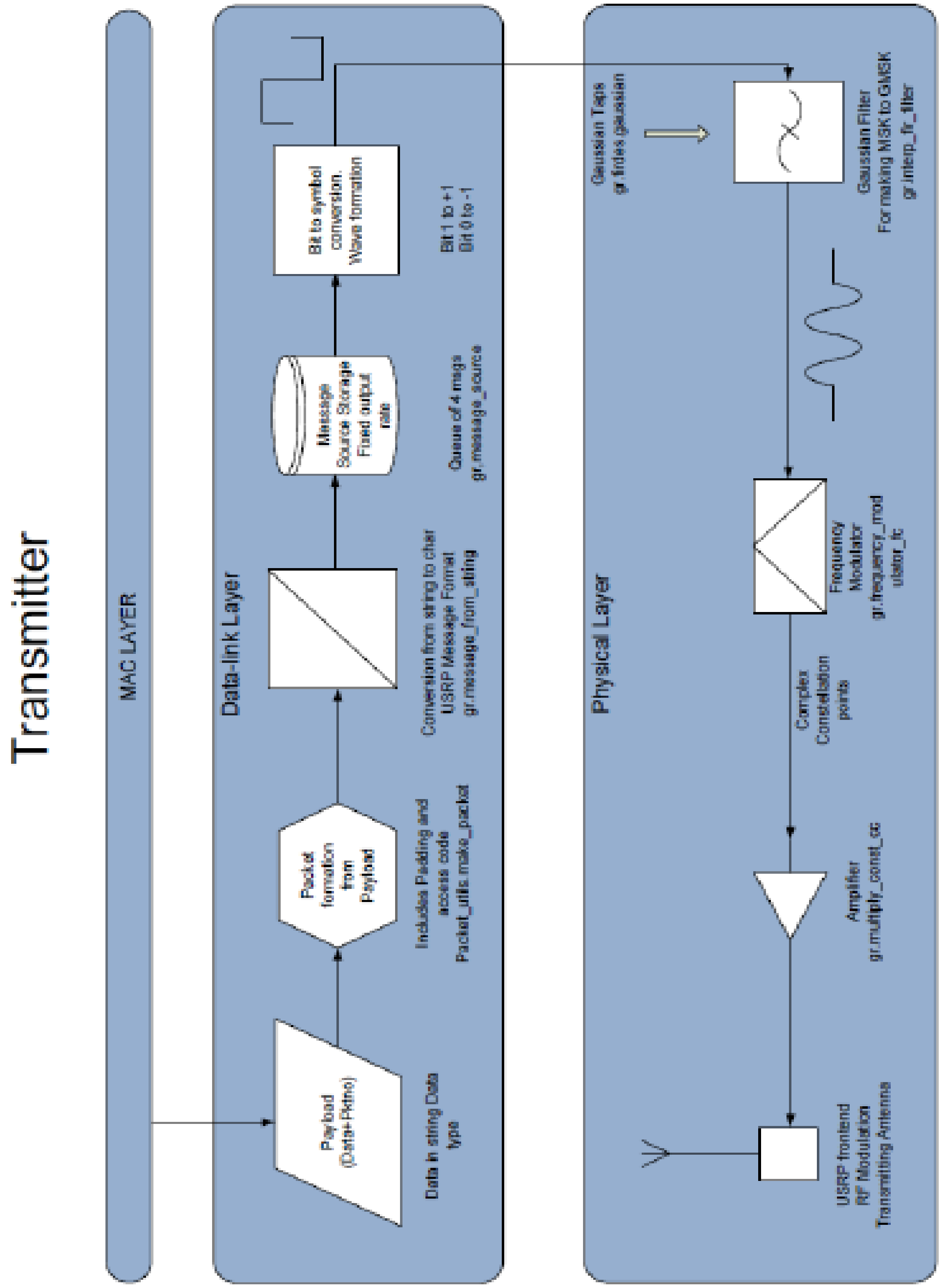

Transmitter
MAC LAYER
Data-link Layer
Payload (Data+Pktno)
Data in string Data type
Packet formation from Payload
Includes Padding and access code Packet_utils.make_packet
Conversion from string to char USRP Message Format gr.message_from_string
Message Source Storage Fixed output rate
Queue of 4 msgs gr.message_source
Bit to symbol conversion. Wave formation
Bit 1 to +1
Bit 0 to -1
Physical Layer
Gaussian Taps gr.firdes.gaussian
Gaussian Filter For making MSK to GMSK gr.interp_fir_filter
Frequency Modulator gr.frequency_modulator_fc
Complex Constellation points
Amplifier gr.multiply_const_cc
USRP frontend RF Modulation Transmitting Antenna

Figure 8-3 Transmitter Module Designed for SDR/USRP

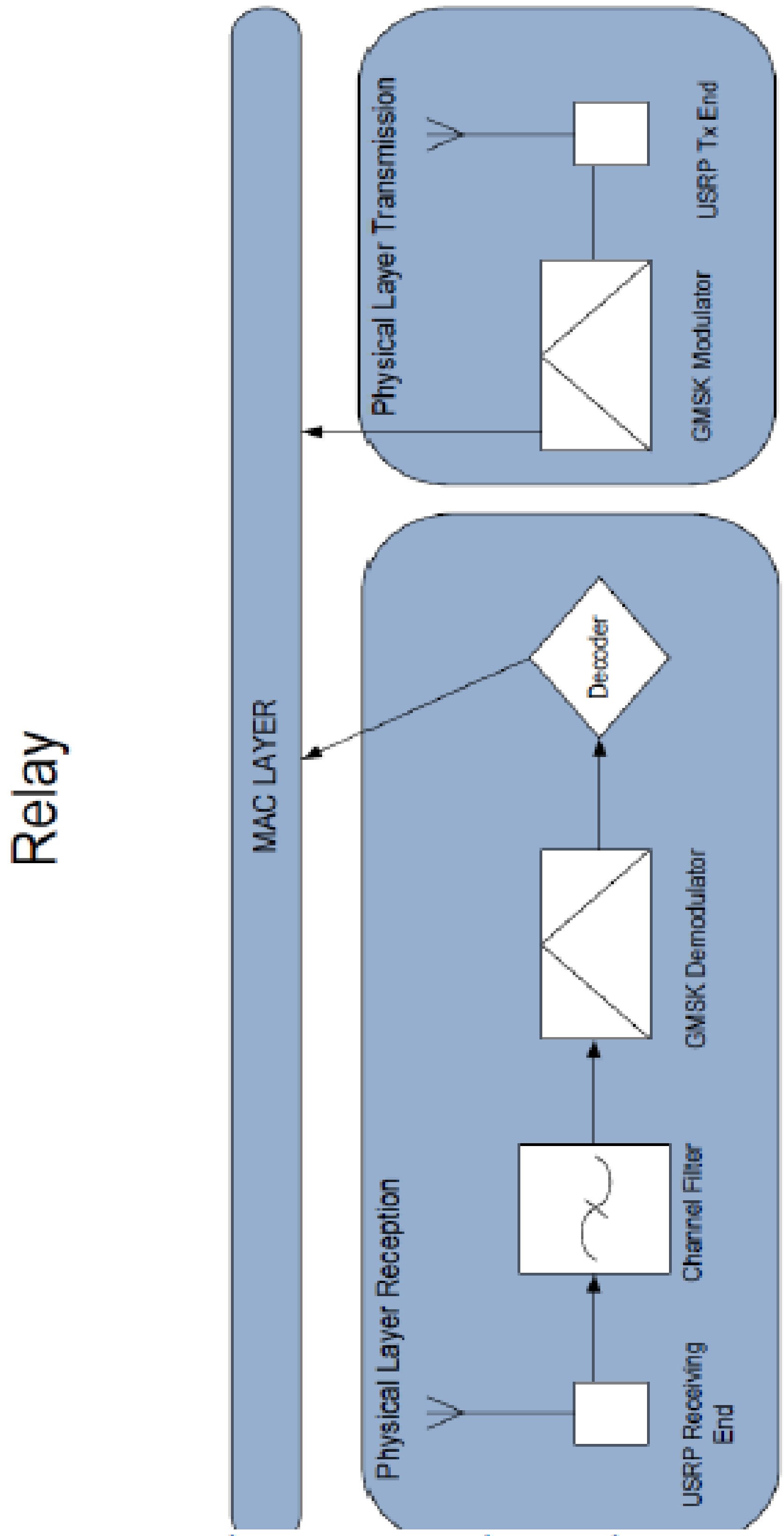


Figure 8-4 Relay Module Designed for SDR/USRP

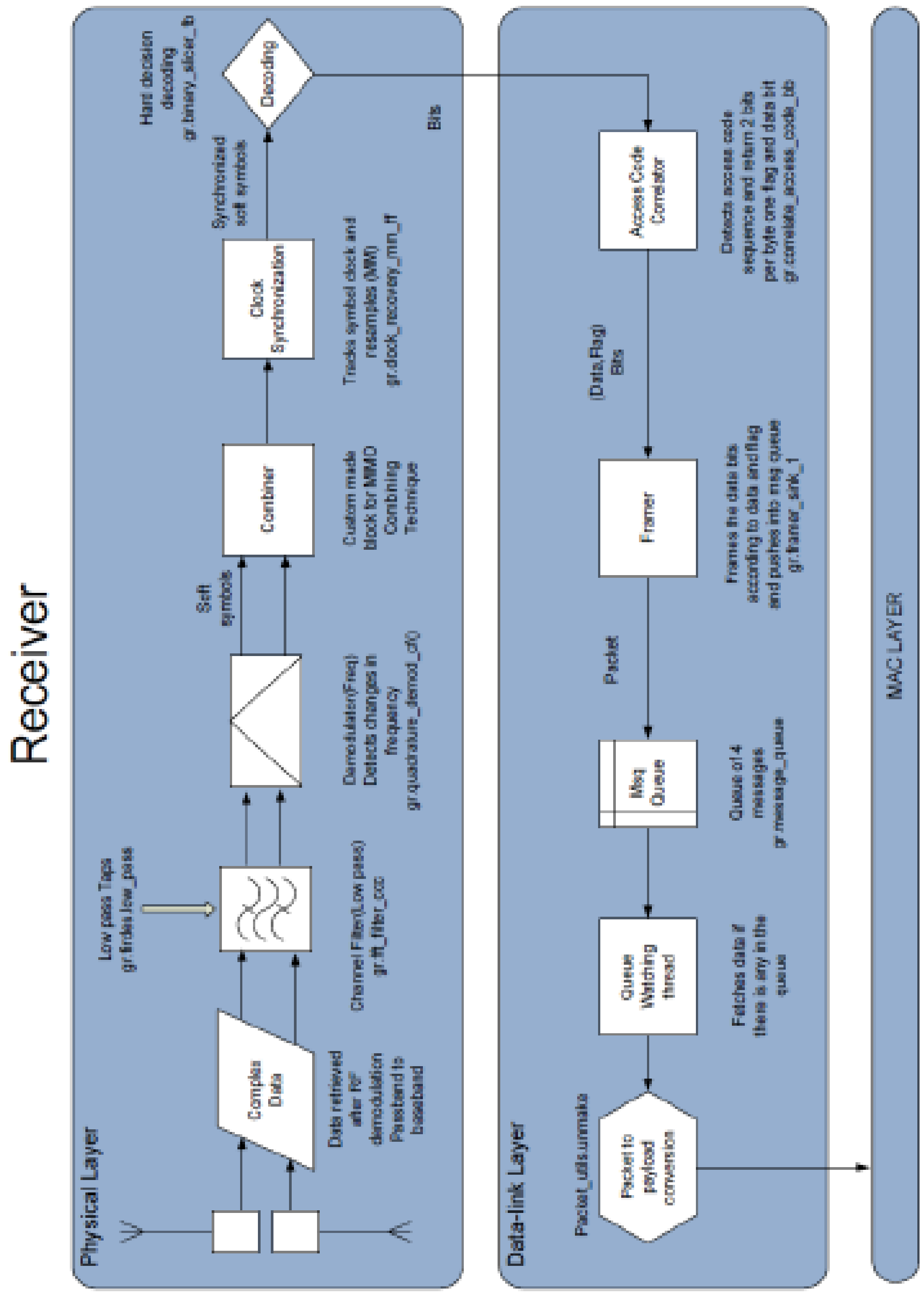

Receiver
Physical Layer
Complex Data
Data retrieved after RF demodulation Passband to baseband
Low pass Taps gr.firdes.low_pass
Channel Filter(Low pass) gr.fft_filter_ccc
Demodulator(Freq) Detects changes in frequency gr.quadrature_demod_cf()
Soft symbols
Combiner
Custom made block for MIMO Combining Technique
Clock Synchronization
Tracks symbol clock and resamples (MM) gr.clock_recovery_mm_ff
Synchronized soft symbols
Decoding
Hard decision decoding gr.binary_slicer_fb
Bits
Data-link Layer
Access Code Correlator
Detects access code sequence and return 2 bits per byte one flag and data bit gr.correlate_access_code_bb
(Data,Flag) Bits
Framer
Frames the data bits according to data and flag and pushes into msg queue gr.framer_sink_1
Packet
Msg Queue
Queue of 4 messages gr.message_queue
Queue Watching thread
Fetches data if there is any in the queue
Packet_utils.unmake
Packet to payload conversion
MAC LAYER

Figure 8-5 Receiver Module Designed for SDR/USRP

Chapter No. 9

Data Link Layer and Medium Access Control

9.1 System Model

A cooperative system implementation requires at least three network nodes, a source, a relay and a destination as shown in Figure 9-1. The nodes in the network represent the USRP in our experimental setup with the RFX-2400 daughter boards acting as front ends. The transmitted signal is broadcast and heard by the destination and the relay. The relay then retransmits this signal to the destination. This provides the destination with independent multiple copies of the transmitted signal which by application of a combining technique yields data with smaller error rates.

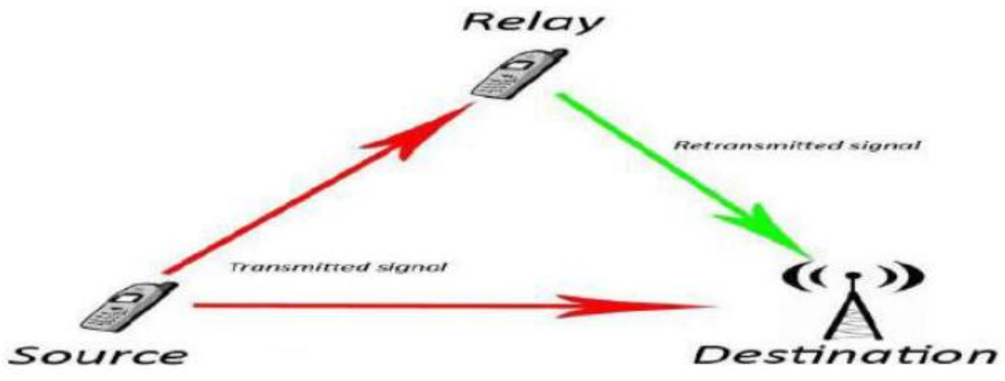


Figure 9-1 Requirement of at least Three Network Nodes

Due to multiple transmitting stations in the network there is a need for a medium access control such that the data is not lost due to collision.

9.2 Medium Access Control (MAC)

The medium access control is a special sub layer of the data link that deals with multiple access and medium access. It provides channel access control mechanisms that make it possible for several terminals or network nodes to communicate within a multi-point network.

The network nodes, the source, relay and the destination are individual distributed entities within the network. Each has the capability to transmit and receive (transceiver operation) through the use of individual threads in software for the two operations. The relay transmits only the information/data that it receives from the transmitter making it a passive relay (relay that does not generate its own data). Even though it does not generate data of its own there is a high probability that collision of data streams from the transmitter and relay takes place at the receiver. Thus there is a need of some kind of medium access control scheme.

Medium access can be controlled by creating orthogonality in data. This can be achieved in time, frequency, code or space or a combination of two or more parameters.

For this purpose a ***time division scheme*** as suggested in [10] is adopted. The transmitter and relay take turns to transmit. The transmitter transmits a data packet which is received by both the relay and the receiver and then goes into a sleep mode for at least the coherence time during which the relay can successfully forward the data packet. A minimum of ***coherence time*** is required for this TDMA scheme without any guard interval so that data collision has a very small probability of occurrence. For our experimental setup though, we used a guard interval between packet transmissions from the transmitter to be sure of collision free reception at the receiver.

Cooperative relaying systems have a minimum requirement of three network nodes, a source, a relay and a destination. The systems can though, have more than one relaying nodes. The presence of multiple relays complicates the network layer protocol but at the same time provides a greater amount of spatial diversity gain. To account for the number of relays that a cooperative network can accommodate with time orthogonality, the delay time between frame transmissions is of paramount importance. The transmission time of a frame can be calculated as

$$T_f = \frac{N_b}{R}$$

where Nb is the number of bits per frame and R is the bit rate of the transmission link. Greater the frame size, greater the frame transmission. A bit rate of 500k is used with the frame size of 256 bits including overheads. The frame transmission time then comes out to be Tf = 256/500k = 512 microseconds.

The number of relays that the network can support without collision of frames from source and relays can be determined using the delay time for frame transmission.

$$N_{relays} = \frac{delay_{T_f}}{T_f}$$

where N relays is the maximum number of relays the system can accommodate with the corresponding delay and frame transmission time.

Greater the amount of delay between successive frames, greater is the number of relays that can be accommodated. For the cooperative platform developed a frame delay time of 5ms which constitutes the guard time is used. This corresponds to Nrelays = 5ms/Tf which comes out to be approximately 10 relays. The above mentioned calculations are made for the experimental analysis of cooperative MIMO communications over USRP.

9.3 Data Link Layer

To ensure seamless communication between the network nodes it was necessary that the nodes could communicate with one another and acknowledge data after reception.

9.3.1 Data Link for SISO Link

The development of the data link layer for the ultimate cooperative network started off with the development of the layer for SISO link. Initially when communication was established between nodes it was noted that the receiver sometimes did not receive the complete file even when the transmitter transmitted all of the data. So a simple ARQ protocol was suggested and implemented that generated ACK for a correctly received packet and a NACK for a corrupt packet.

Still it was observed that sometimes the transmitter froze and did not transmit the entire file. The problem was that the channel sometimes did not let the acknowledgement (ACK/NACK) packet to reach the transmitter, thus the transmitter had no way to know that the packet it last sent was received or not and should it transmit a new packet. For this a timing overflow was incorporated to the layer which told the transmitter to retransmit a packet if its acknowledgement failed to reach the transmitter within a certain time period. This ensured complete and reliable data transmission in a SISO link.

9.3.2 Data Link for Cooperative Network

The development of data link for cooperative network was a much more complex task than the development of the data link for SISO link since the network consisted of decentralized nodes that needed to convey there state of operation to one another, whether they had received the required packet or awaiting reception. Initially a layer that performed the same task as the layer for SISO link communication was developed. All the nodes exchanged information on the current packet number and the transmitter retransmitted if the packet failed to reach either the relay or the receiver.

Though the communication was flawless, it was observed that the transmitter was overloaded with acknowledgement data from the receiver and relay whereas the relay was overloaded with data forwarding and receiver acknowledgements. The options were to enhance processing capabilities or come up with a simpler layer design.

The later was considered a cost effective solution and adopted for implementation. A close match of the sliding window protocol was developed, which noted the last correct packet received and generated a NACK only if the packet was corrupted from both the source and relay. The packet was accepted in case of correct reception from anyone of the two and the packet number was buffered without generation of an acknowledgement to save processing at the transmitter and relay. A NACK received at the source (through either of the two links) meant retransmission of the packet. This proved effective and reliable transmission without loading the processor.

Chapter No. 10

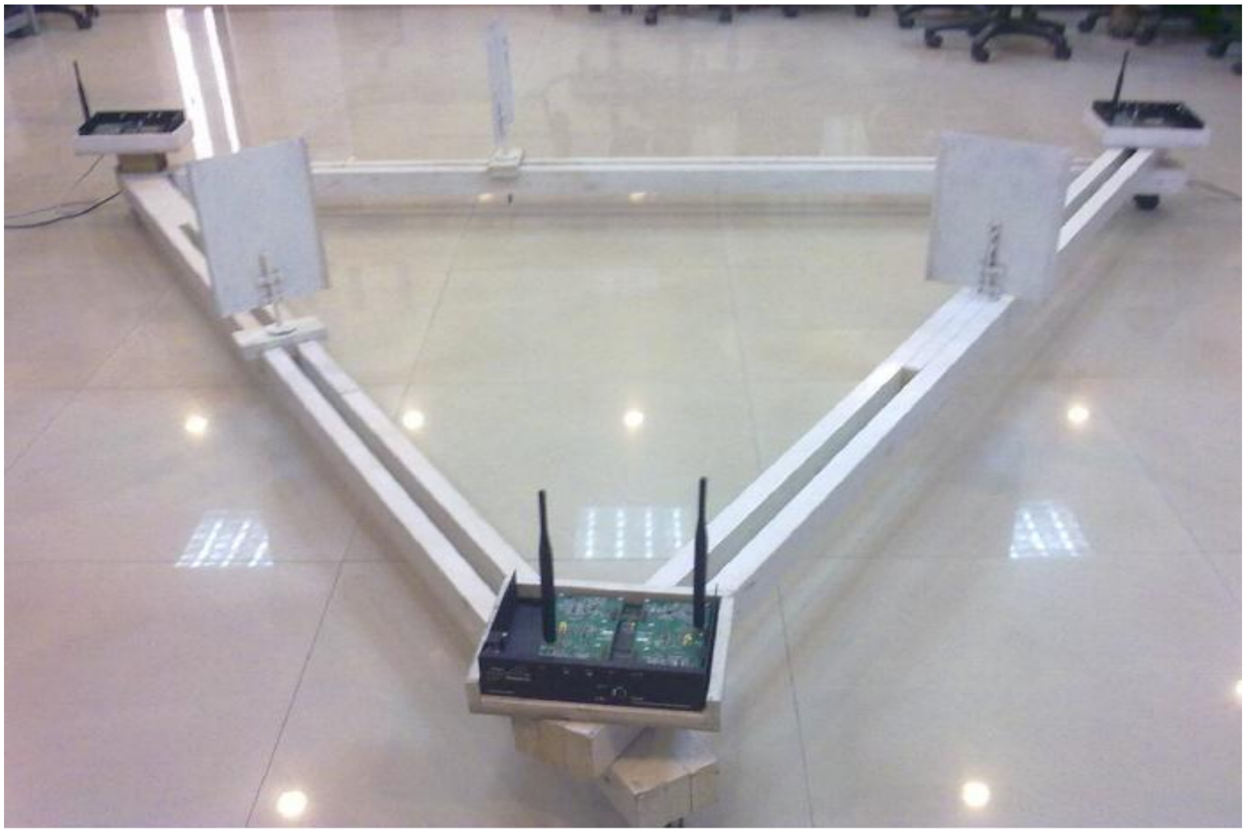
Experimental Setup and Calibration

For simplicity, the network was comprised of three nodes: a source node, a relay node, and a destination node. The relay fully decoded the source‘s message before forwarding it on to the destination which does not guarantee to provide full diversity, is one of the simplest schemes that are possible. This section will describe this simple network‘s geometry and system parameters, steps taken for characterization of the network and calibration of the nodes.

10.1 Apparatus and Geometry

For the generation of BER curve data, the average received SNR for a given configuration must be known. This SNR is a factor of the path loss, and thus the distance that exists between two given nodes, and the transmit power. Estimating received SNR for a given channel realization at the destination is not helpful, as this includes the reduction in SNR resulting from attenuation of the signal by the multipath fading. There are two suitable ways to overcome this difficulty: keeping the distances between the nodes fixed while varying transmit power in a known way, or keeping the transmit power fixed while varying the distances between the nodes in a known way. In order to create simplicity, the former was adopted. The requirement to average over multiple realizations of the multipath channel gain means that the network nodes must also be moved within their environment.

These requirements led to the experimental apparatus seen in Fig. 8-1. The three nodes in the relay network are fixed to the corners of an equilateral triangle. The distance between nodes on the triangle is adjustable between 2 m and 3 m. Wheels are attached to the bottom of the triangle, and power and USB wires are routed on the sides to allow for easy movement of the experiment. The equilateral triangle results in equal average received SNR for the source-destination (S-D) link, the source-relay (S-R) link, and the relay-destination (R-D) link. In this setup, the average relay path (S-R-D) will not have an advantage over the direct path (S-D). It is desirable for the receive antennas to be located in the far-field antenna pattern of the transmitting antennas. In this far field region, signal strength is dependent solely on the distance from the transmitter and not on angular position.



A general rule-of-thumb [15] that can be used for determining where the far-field begins is

$$d_{ff} = \frac{\lambda}{2\pi}$$

where λ is the wavelength of the signal. For this experiment having a carrier frequency of 2400 MHz, λ was 0.125 m, giving a dff of 0.019 m. Again, a rough estimate [16] for the coherence distance is

$$d_c = \frac{\lambda}{4}$$

In order to obtain independent multipath gains, the nodes need to be moved by approximately this distance.

The coherence time is much greater than the transmission time of a packet, creating a slow fading environment in which all symbols within a packet experience approximately the same fading. Additionally, 1 ft x 2 ft metal sheets were placed in the middle of each radio link to attenuate the line-of-sight path. The advantage of attenuating the LOS is that diversity gains can be clearly identified.

Chapter No. 11

Performance Analysis of Cooperative MIMO System

Two of the most desired outputs of any communication system are

- High throughput rates
- Communication reliability

In the context of MIMO communication systems the above mentioned desirables are a result of spatial multiplexing gain and spatial diversity gain respectively. As mentioned in chapter 2 multiplexing and diversity gains are tradeoffs of one another. A communication network needs to be identified as having a requirement of high throughput or reliability. Much depends upon the environment in which the communication link is established. In case of a severe fading environment with no LOS path available it is sane to go for reliability through diversity rather than for high throughputs through multiplexing.

The project was aimed at achieving communication reliability through quantifying diversity gains provided by cooperative systems over conventional collocated antenna arrays and SISO communication. The system model described in chapter 6 provides explanation for shielding the LOS.

11.1 Communication Reliability through Diversity

Diversity is an attempt to increase transmission reliability in the fading environment via transmitting data from a source to a destination in multiple forms over, ideally, multiple independent fading realizations. The motivation for such an approach stems from the fact that, even if it is highly probable that one path is in a deep fade, it is less likely all paths will be severely impaired, providing robustness against multipath fading.

Spatial diversity comes in a variety of forms. Transmit diversity can be created by using multiple transmit antennas with a single receive antenna; receive diversity can be created by using multiple receive antennas with a single transmit antenna. A combination of the two results in a multiple-input multiple-output (MIMO) system. Cooperative MIMO systems create a virtual antenna array though cooperating distributed nodes of the network. For diversity gains, paths between different antenna pairs must be independent. This can usually be achieved by separating the collocated antennas by at least half of a wavelength of the carrier.

11.2 BER vs. SNR: Diversity Considerations

A common way of assessing performance for a given transmission scheme over the AWGN channel is to look at how quickly the bit error rate (BER) decreases with SNR. BER is known to decrease with the square of SNR for the AWGN channel [7]. In fading channels, it has been observed that the BER only decreases at a rate proportional to the inverse of SNR. The BER of diversity schemes can be approximated in the high SNR regime as [17]

$$\mathrm{BER} \approx (c \cdot \mathrm{SNR})^{-L}$$

where L is known as the diversity order and determines the rate at which the probability of error or BER decays at high SNR. Additionally, c is a coding gain obtained when codes more efficient than repetition coding are employed. Figure 11-1 compares the BER versus SNR curves for systems both with and without diversity by using a log-log scale.

The BER of the system with diversity can be observed to decay linearly at a rate of two orders of magnitude per decade of SNR. This steeper slope is characteristic of the diversity schemes. The rate at which the system BER decays is equal to the diversity order and is typically the number of independent paths in the system. Observing or proving the presence of diversity is thus usually done by doing BER curve analysis. This is a preferable choice since it not only indicates the presence of diversity, but is also a measure of the gain obtained therefrom.

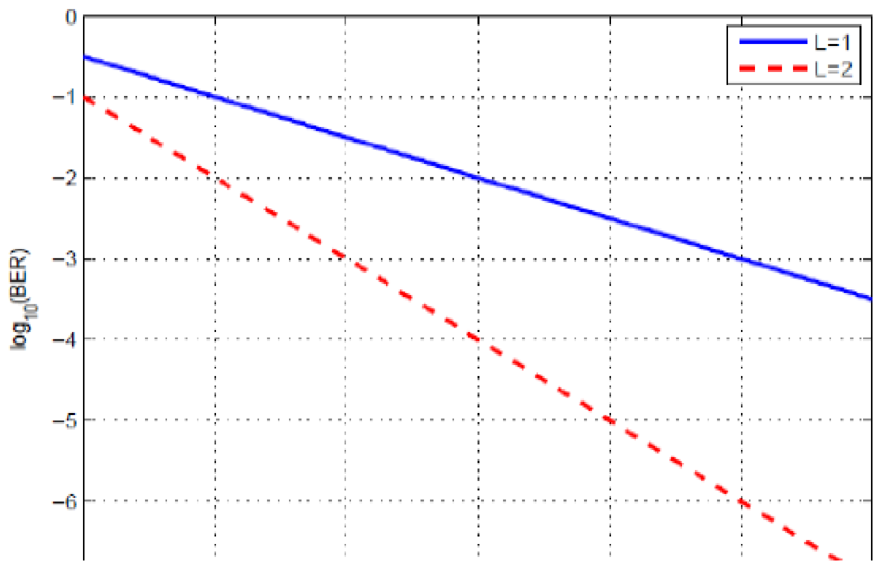


Figure 11-1 BER versus SNR in the high SNR Regime for no Diversity (L=1) and diversity order 2 (L=2) with coding gain c=1

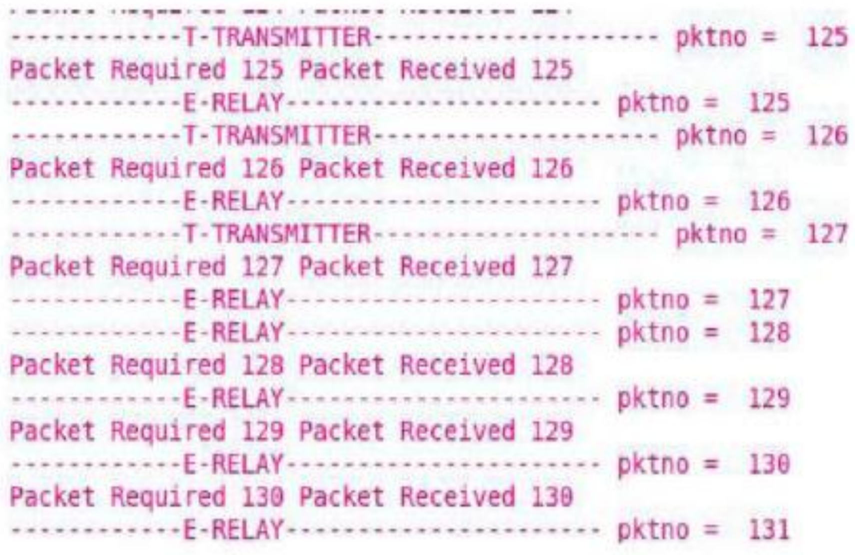
The simulation curves in chapter 4 provide a clear picture of the amount of diversity gain that the cooperative systems achieve over conventional systems. There are a large number of theoretical studies of cooperative networks which provide simulation results for cooperative networks. For a proof of concept of the gains provided by cooperative systems in theory, protocols established for a software radio test bed were used to compare performance of various communication systems.

11.3 Conventional Performance Parameters

A conventional performance parameter to analyze performance gains of cooperative systems is the outage capacity. When a source cannot reach its destination due to severe fading, so to keep trying by leveraging repeating transmission protocols such as ARQ will not be of much help. If a third party that receives the information from the source could help via a channel that is independent from the source-destination link, the chances for a successful transmission would be high, hence improving the overall performance. For cooperative networks there is path redundancy for signals via the relay hop.

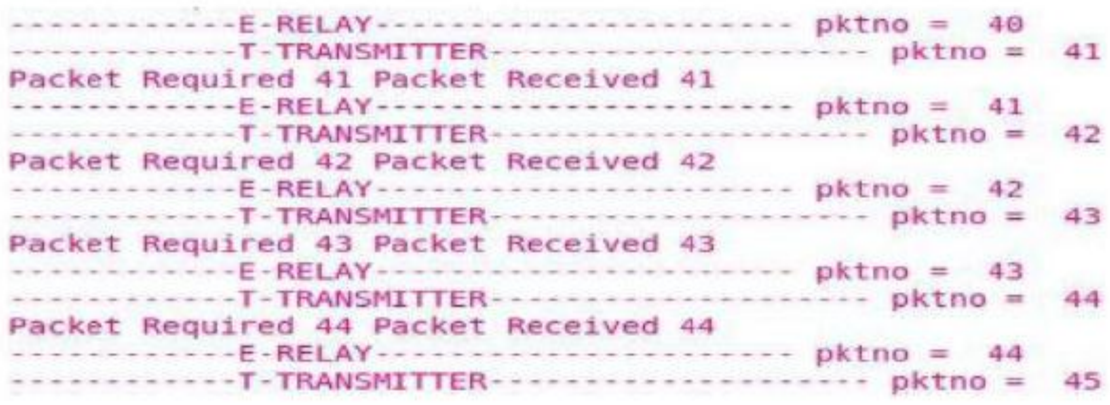
Figure 11-2 shows a screen shot of a receiver that is receiving signals from both the transmitter (T) and relay (E). Once the direct source-destination link goes under a deep fade the information still reaches the destination via the relay (E) link as in Figure 11-3, thus enhancing overall system capacity.

Figure 11-2 Path Redundancy-Signal from Independent Channels



```
------------T-TRANSMITTER------------------- pktno =  125
Packet Required 125 Packet Received 125
------------E-RELAY--------------------- pktno =  125
------------T-TRANSMITTER------------------- pktno =  126
Packet Required 126 Packet Received 126
------------E-RELAY--------------------- pktno =  126
------------T-TRANSMITTER------------------- pktno =  127
Packet Required 127 Packet Received 127
------------E-RELAY--------------------- pktno =  127
------------E-RELAY--------------------- pktno =  128
Packet Required 128 Packet Received 128
------------E-RELAY--------------------- pktno =  129
Packet Required 129 Packet Received 129
------------E-RELAY--------------------- pktno =  130
Packet Required 130 Packet Received 130
------------E-RELAY--------------------- pktno =  131
```

Figure 11-3 Information reaching Receiver via Relay (E) when Direct Link is under a deep fade



```
------------E-RELAY--------------------- pktno =  40
------------T-TRANSMITTER------------------- pktno =  41
Packet Required 41 Packet Received 41
------------E-RELAY--------------------- pktno =  41
------------T-TRANSMITTER------------------- pktno =  42
Packet Required 42 Packet Received 42
------------E-RELAY--------------------- pktno =  42
------------T-TRANSMITTER------------------- pktno =  43
Packet Required 43 Packet Received 43
------------E-RELAY--------------------- pktno =  43
------------T-TRANSMITTER------------------- pktno =  44
Packet Required 44 Packet Received 44
------------E-RELAY--------------------- pktno =  44
------------T-TRANSMITTER------------------- pktno =  45
```

Thus it may be said that path redundancy increases the overall system throughput capacity. Another conventional metric is the range extension for coverage achieved as a result of the multi hop in the network due to the presence of the relay node.

For cooperative relaying networks the application of the network is highly dependent upon the placement of the relay node. When placed within the primary coverage area, its major application is coverage hole elimination and providing path redundancy. A relay is placed at the very edge of the network coverage area to provide range extension. The application requirement is driven by the area where services are to be provided.

11.4 FER vs. SNR: Comparative Analysis of Communication Systems

SNR is proportional to the transmit power of the signal when distances between the nodes (single hop) are kept constant. When distances are kept constant the path-loss for all paths is the same and transmit power of the signal measured using a signal analyzer can be termed as the SNR at the receiver node. Figure 9.4 is a plot of the Frame Error Rates (FER) against the Signal to Noise Ratio (SNR) for different communication schemes

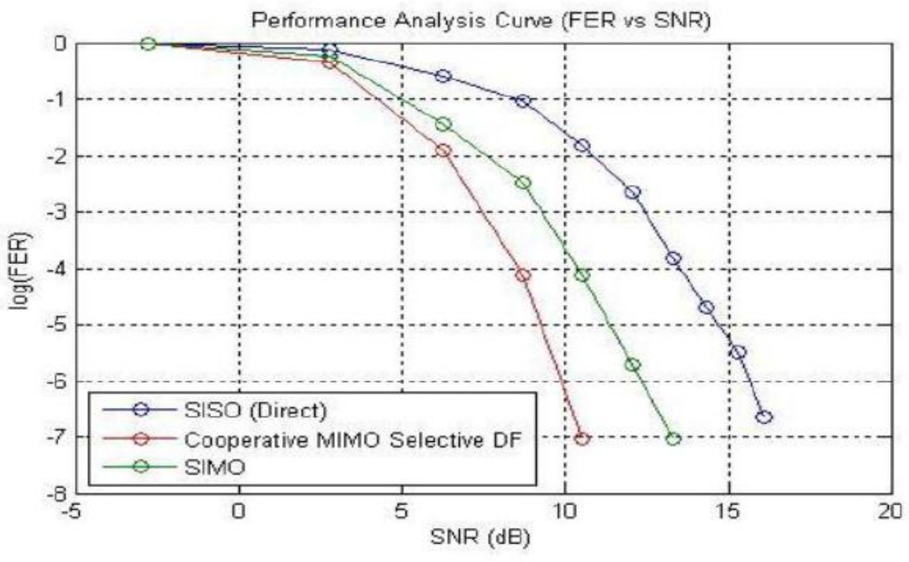


Figure 11-4 Performance Gain of Cooperative MIMO over SIMO (Direct) communication

In [18] simple and selective decode and forward techniques were analyzed in comparison with direct transmission. It was shown through experimental results that the simple DF did not provide discernible diversity gain. The selective DF however provided a measurable diversity gain which is here tested against various communication systems.

A node was selected out of the available number of nodes and its transmit power was measured on a signal analyzer for varying signal transmit characteristics in the control line of the software terminal. A snapshot of signal is shown in Figure 11-5 which depicts the maximum transmit power of the USRP which is around 16 dB. The Figure also shows the center frequency of the signal which is 2.45GHz with the signal bandwidth around 3MHz.

The diversity gain is quantified as the slope of the FER-BER curve. Steeper the curve, the better the gain provided by the system. The curves of direct (SISO), Single Input Multiple Output (SIMO) and Cooperative MIMO (CMIMO) are plotted, which clearly depict the supremacy of Cooperative systems. The Cooperative MIMO results compiled are for the selective decode and forward technique where the relay selectively forwards frames with the correct CRC.

The amount of diversity gain depends upon the diversity order of the system. This can be verified by mentioning that direct transmission is a 1x1 system, the SIMO system is a 2x1 system while CMIMO systems are 2x2 systems. Infact the theoretical transmit diversity gain provided by cooperative networks is even higher than conventional collocated antenna array systems (MIMO) because of the spatial delocation of the antennas.

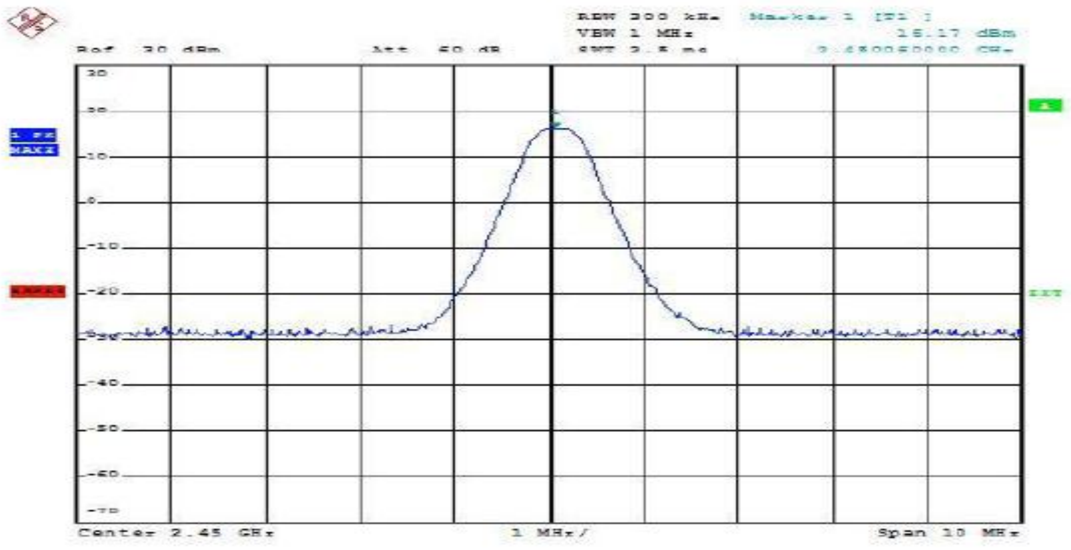


Figure 11-5 Input Signal – Power Bandwidth Characteristics

11.5 Conclusion

A framework to analyze spatial diversity gain for Cooperative Selective Decode and Forward (DF) technique is developed. The experimental results achieved suggest that Cooperative Selective DF provides reliable communication with lesser FER at lower SNR values thus improving system performance. Furthermore outage capacity is enhanced through path redundancy provided by relaying.

BIBLIOGRAPHY

[1] A. Sendonaris, E. Erkip and B. Aazhang, —*User cooperation diversity, part I: system description*‖ *IEEE Trans. Commun.*, vol. 51, no. 11, pp. 1927–1938, Nov. 2003.

[2] User cooperation diversity, part II: *Implementation aspects and performance analysis*‖ *IEEE Trans. Commun.*, vol. 51, no. 11, pp. 1939–1948, Nov. 2003.

[3] J. N. Laneman, D. N. C. Tse and G. W. Wornell, —*Cooperative diversity in wireless networks: efficient protocols and outage behavior*,‖ *IEEE Trans. Inf. Theory*, vol. 50, no. 12, pp. 3062–3080, Dec. 2004.

[4] T. Cover and A. E. Gamal, —*Capacity theorems for the relay channel*,‖ *IEEE Trans. Inf. Theory*, vol. IT-25, no. 5, pp. 572–584, Sep. 1979

[5] http://gnuradio.org/redmine/wiki/gnuradio

[6] http://gnuradio.org/redmine/wiki/gnuradio/BuildGuide

[7] http://gnuradio.org/redmine/wiki/gnuradio/UbuntuInstall

[8] http://gnuradio.org/redmine/wiki/gnuradio/MailingLists

[9] http://gnuradio.org/redmine/repositories/changes/gnuradio/README

[10] J.N. Laneman, G.W. Wornell, *"Distributed Space Time-Coded Protocols for Exploiting Cooperative Diversity in Wireless Networks*"

[11] www.wikipedia.com

[12] *Cryptography and Network Security* by *William Stallings*

[13] www.topbits.com

[14] http://pypi.python.org

[15] M. F. Iskander, *Electromagnetic Fields and Waves*. Waveland Press, Inc., 1992.

[16] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, June 2005.

[17] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, June 2005

[18] G.J. Bradford, J. Laneman, ―*An Experimental Framework for the Evaluation of Cooperative Diversity*"