FPGA IMPELMENTATION OF BASEBAND PROCESSING MODULE FOR TELECOMMAND RECEIVER

Submitted by:

Salman S/O Sadruddin

2011-NUST-MS Phd-Elec(Comm-N)-14

Supervisor:

Dr. Arshad Aziz

Thesis Submitted:

In Partial Fulfillment of the Requirements for the Degree of Master of Science in Electrical Engineering with Specialization in Communications

at the Department of Electronic and Power Engineering

Pakistan Navy Engineering College, Karachi

National University of Sciences and Technology

H-12, Islamabad, Pakistan

December 2013

DEDICATED TO MY PARENTS

Acknowledgement

All praise is for Almighty Allah alone. I would like to express immense gratitude to my supervisor Dr. Arshad Aziz for his valuable guidance and support which enabled me to complete my research. I am also grateful to my GEC members; Dr. Pervez Akhtar, Dr. Khawaja Bilal Ahmed Mahmood and Dr. Sameer Hashmat Qazi.

I am also thankful to Mr. Muhammad Kashan Mobeen and Mr. Zeeshan Jawaid of Pakistan Space and Upper Atmosphere Research Commission (SUPARCO) for their thoughtful suggestions which helped me to improve my work.

I am extremely grateful to my beloved parents for their continuous support throughout my work.

Finally, I offer my sincere thanks to my friends and colleagues who motivated me during my work.

Abstract

This work presents a resource efficient baseband module implementation of a LEO satellite telecommand receiver using FPGA. The adopted scheme uses a digital Costas loop for carrier recovery and an improved early late gate timing recovery algorithm for bit synchronization. Loop filter is designed and implemented without using embedded multipliers. The Bit Error Rate (BER) performance of the designed receiver is almost identical to theoretical values with negligible difference due to implementation losses. The optimized receiver module has BER performance identical to theoretical, with minor degradation due to implementation losses.

A new method for software defined radiation hardening of a baseband module for a LEO satellite telecommand receiver is proposed. FPGAs in space are subject to single event upsets (SEUs) due to high radiation environment. Traditionally, triple modular redundancy (TMR) is used for mitigating Single Event Upsets (SEUs). The drawback of using TMR is that it consumes a lot of hardware resources and requires more power. Reduced precision redundancy (RPR) can be a viable alternative of TMR in digital systems for arithmetic operations. This work uses the combination of RPR and TMR for mitigating SEUs.

The designed system consumes less resources when compared to a BPSK receiver having same specification. It compensates frequency shifts up to ± 200 KHz due to Doppler effect. The hybrid software defined radiation hardening technique consumes 26% less area than a customary TMR protected receiver.

Table of Content

List of Figures

List of Tables

CHAPTER 1 INTRODUCTION

1.1 Background

The Satellite industry has progressed dramatically in the last five decades. Satellites in the early ages were big in size and heavier in weight. But after the birth of microcontrollers, the trend of size and mass of satellites splits into two categories: large satellites and small satellites. Small satellites are recently gaining more interest and attention all over the world due to their attractive applications. Their fast and cost effective development process makes them a suitable platform for technology evaluation and demonstration missions. But Small space craft increase the constraints of limited available power, size and mass for the satellite payload. Field programmable gate arrays offer this solution to these limitations. There small size, light weight and high computational capabilities makes them a preferred choice over other digital systems. FPGA's also provide adaptability and reconfigurability which are the trending feature of modern space technology. The ability to remotely reconfigure FPGA with an updated functionality reduces the hardware requirement in space craft [1].

Communication modem is one of the key subsystems of a satellite. It establishes the communication channel between satellite and the control center on the earth. Its function is to transmit the telemetry data of the satellite to ground station and to receive the telecommand data from ground station. It operates during all the phases of mission. Conventionally, telemetry and telecommand unit is made using discrete electronic components. This makes the unit large and bulky which is well suited for large satellites. But when it comes to resource constraint environment of small satellites, its size and weight becomes a serious concern. FPGA based transceivers consume less space and have light weight. They also offer the flexibility and last minute modification freedom which is not possible in discrete hardware based transceivers.

However, FPGA's face some severe problems in the space environment. Space contains high energy particles and ionizing radiations which can cause malfunctioning in integrated circuits. The effects of these subatomic particles on integrated circuit are referred as Single Event Effect (SEE). The high energy particles in space may interact with the memory cells within an integrated circuit which can change their logic state [2]. This alteration may disrupt the operation of a digital system defined by memory cells. This phenomenon is called as Single Event Upset (SEU). FPGAs contain large array of memory cells which makes them

more susceptible to SEUs. In the past, radiation effects have been treated as a part of hardware problem and they are mitigated by shielding and radiation hardened processors. Shielding makes the module heavy and doesn't provide protection against high energy particles such as heavy ions. Radiation hardened processors require large lead time, consume more power and are very costly.

Software defined radiation hardening techniques are proving to be a viable solution to these problems. This approach enables the use of COTS hardware in space. There exist a huge performance gap between commercial and space grade hardware. COTS will provide a drastic increase in performance capability while also reducing the cost by many folds.

1.2 Survey of Related Work

In Z. Zhao., et al., [3] the BPSK Modem is implemented as a part of software defined radio. The authors of [3] have used BPSK modulation and recovered the modulated carrier using Costas loop. The modulation and demodulation are simulated on MATLAB. But there design did not have bit synchronization. Maya, J.A., et al., [4] presents the BPSK receiver for high data rates and high dynamic applications. The Carrier recovery is achieved by using Costas loop. Loop filters were realized by analog to discrete time conversion. Gardner algorithm is used for timing recovery of the demodulated signal.

Traditionally, Triple Modular Redundancy (TMR) has been used for SEU mitigation. The drawback of using TMR is that it consumes a lot of hardware resources and requires more power [5]. Thus; there has been a constant effort to find an alternative to the TMR technique. Shim, et al. [6] introduced Reduced Precision Redundancy (RPR) as part of a power-reduction technique for ASIC-based systems, Snodgrass [7] demonstrated variation of RPR on FPGA to limit high magnitude errors of arithmetic operations in high radiation environment. Pratt, B., et al [8] has presented the hybrid approach using RPR and TMR for FPGA based communication systems.

1.3 Aim

This thesis presents a highly efficient design of a software defined radiation tolerant module for a LEO satellite telecommand receiver. During literature review, it was found that the BPSK receiver of Maya, J.A., et al [4] and our design has same technical specifications. So it was taken as a bench mark for fair resource utilization. The designed module consumes less

resources and its Bit Error Rate (BER) is approaching to 10-6. The designed module uses the combination of RPR and TMR for SEU mitigation. To the best of author's knowledge, this hybrid approach is not being implemented to a satellite telecommand receiver module using Binary Phase Shift Keying (BPSK) modulation. The research work also evaluates the effect of SEUs on the BER performance of a telecommand receiver.

1.4 Thesis Outline

The rest of the thesis report is organized as follows:

- Chapter 2 deals with the introduction to satellite architecture and its communication system
- Chapter 3 describes the detailed architecture of telecommand receiver
- Chapter 4 presents the advantages of FPGA and SEU mitigation technique
- Chapter 5 explains the design methodology adopted
- Chapter 6 gives the implementation of the system using high level tools
- Chapter 7 presents the Hardware Co-simulation and SEU mitigation results of designed receiver
- Chapter 8 concludes the report and suggests future work

CHAPTER 2 SATELLITE COMMUNICATION

2.1 Introduction

A satellite is an object that revolves or orbits around another object. For example, the Moon revolves around the earth so it's a natural satellite of Earth, in the same way Earth is a natural satellite of the Sun. In context of space flight, satellites are manmade objects that are positioned in the earth orbit on purposely. They are also called as artificial satellites in order to differentiate them from natural satellites. They are launched from earth in a Satellite launching vehicle (SLV) and are placed in orbit at a predetermined location according to the mission requirements. There are hundreds of satellites currently in orbiting the earth.

Satellite size and shapes vary according to their scope of application. The first manmade satellite was launched by former Soviet Union in the year 1957 called "*Sputnik*" and was the size of a basketball. Its purpose was simply to send a Morse code signal repeatedly. In contrast, modern day satellites can transmit, receive and process thousands of signals at the same time, from simple digital data to complex television programs. Nowadays, satellites are being used in wide variety of applications, such as Internet communications, television broadcasting, radio communications, Global Positioning Systems (GPS) and weather forecasting.

2.2 Types of satellite

2.2.1 Navigation

Navigation satellites are the artificial satellites in space that are used for navigation purpose. In modern day, navigation has become one of the most important parameter. In daily life, it is used by people to reach up to their destinations. Navigation is the backbone of aerospace industry, which uses navigation parameters to locate their current position. The most common navigation systems are GPS (Global Positioning System) launched by USA and GLONASS launched by Russia.

2.2.2 Weather

The first weather satellite was launched in the year 1960. Several satellite missions were launched during the 1970s and 1980s from which different meteorological observation have been made. The purpose of the weather satellites is to analyze the current state of atmosphere

and provide this valuable information to ground. This information is then used by the scientists to observe the changes in the global atmosphere such as global warming etc. and is also used by for weather forecasting. Different sensors in the weather satellites allow the scientists to estimate the moisture, cloud cover, wind speed and direction which are then used in different fields of science.

2.2.3 Communication

A communications satellite is a manmade satellite placed in space for the purposes of telecommunications. Today's communication satellites are stationed in geosynchronous orbits, Molniya orbits or low Earth orbits. Major applications of communication satellites are Telephony, Television, radio, mobile satellite technology and satellite broad band.

2.2.4 Earth Observation

Earth observation satellites are used for understanding and analyzing the global environment conditions which are essential for safety and quality of life. The data gathered by these satellites enables us to understand the processes and changes in land masses, oceans, and atmosphere. It allows monitoring the Earth's natural resources and to predict or detect the environmental disasters in a timely manner. The data acquired is used for different applications such as agriculture, geology, forestry, cartography, risk management, defense and environmental studies.

2.3 Communication subsystem

A communication sub-system is a very critical part of the satellite. It is the only link between ground and the satellite. It has three primary functions (1) to transmit a tracking signal, (2) download telemetry to a ground station and (3) to receive telecommand from an earth station. The communication subsystem is also referred as a TT&C (Tracking, Telemetry and Command) system because of its functionality. The communication between the satellite and the ground station is known as data link, is a two way communication channel where the uplink is the data commands transmitted from the ground station to the satellite and the downlink is the telemetry data or the beacon transmitted from the satellite to the ground station [9].

The Figure 2.1 depicts the block diagram of the communication subsystem and its communication with the ground station. The modem converts the signals received from the

ground station and forwards the data to the OBDH (On-Board Data Handling subsystem). The satellite sends the telemetry signal to earth station via its Telemetry transmitter. The earth station receives these signals and in return issue some telecommand signals which are received by the telecommand signal on the satellite. This thesis deals with the efficient design of telecommand receiver.

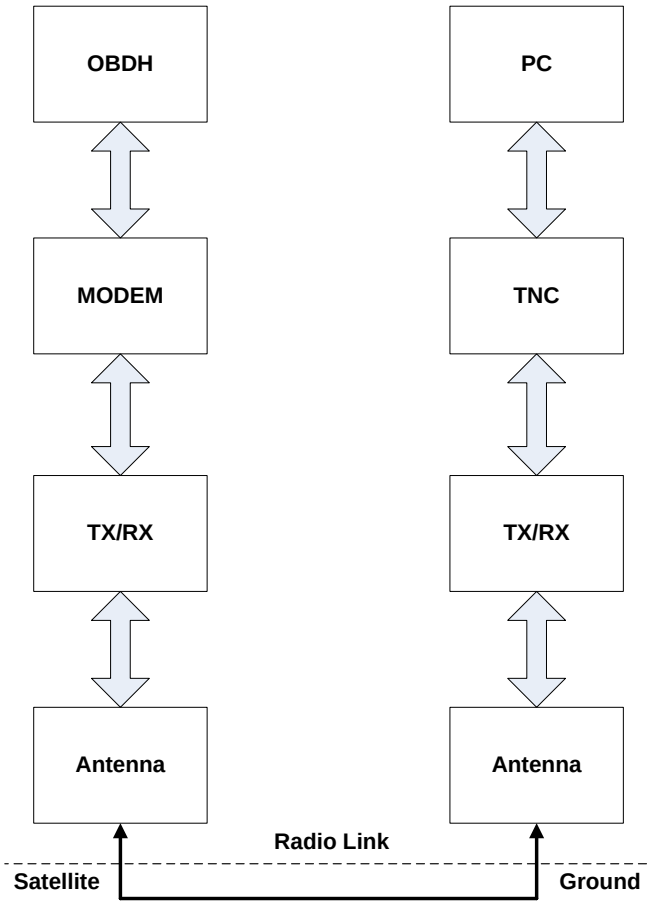

Figure 2.1: A general overview of the communication subsystem

2.3.1 Modem

Modem is a combination of modulator and demodulator. At the transmitting end, Modem acts as a device that accepts binary data from a data source which is modulated to create a signal that is suitable for transmission and at the receiving end it acts as complementary for transmitting end [10]. Modems are classified based on the amount of data transferring capability which is measured in terms of bits per second (bps) or baud rate. In order to encode and decode the data we need to have a proper modulation and demodulation technique. The designing of modem which can establish a good communication link between the Satellite and ground station with the available transmitter power is constrained by these above

mentioned power and size factors [11]. In this thesis we would design a resource efficient design and implementation of a telecommand receiver on FPGA which is the main module in the communication subsystem.

2.4 Modulation and Demodulation techniques

Modulation is process of imposing the properties of a message signal onto a high frequency carrier signal. Modulation schemes are mainly classified into Analog modulation methods and Digital modulation methods. Analog modulation techniques are further classified based on amplitude, frequency and phase modulation techniques. Digital modulation schemes are further classified based on type of keying. The elementary digital modulation techniques are Amplitude shift keying (ASK), Frequency shift keying (FSK), Phase shift keying (PSK) and Quadrature amplitude modulation (QAM). Demodulation is process of retrieving back the original information from the modulated signal. In this thesis we are going to use BPSK (Binary Phase Shift Keying) modulation and AFSK (Audio Frequency Shift Keying) demodulation.

2.4.1 Binary Phase Shift Keying (BPSK)

BPSK is one of three binary modulation techniques. BPSK modulation is a technique where the phase of a carrier sinusoidal signal changes abruptly by 180° or phase reversal occurs for every transition of modulating binary sequence (input bit) [12]. The general form of the BPSK signal is based on the equation (2.1). The binary data is represented by two signals with different phases in BPSK. $S_1(t)$ and $S_2(t)$ are the two signals at point of time t.

$$S_i(t) = \begin{matrix} S_1(t) = -Asin(2\pi f_c t),\ if 0_T \\ S_2(t) = +Asin(2\pi f_c t),\ if 1_T \end{matrix} \quad (2.1)$$

Where:

- A is the amplitude

- f_c is the frequency of the carrier and

- T is the time.

The phase of the transmitted signal remains the same if a “1” was transmitted and is shifted by 180° if a “0” is transmitted [13].

Advantages

- BPSK has fine spectrum efficiency, strong anti-interference performance, good spectral characteristics and has high transfer rates [3].
- Due to its robustness it is extensively used in satellite communication systems.

Disadvantages

- It is simple to implement, but it is inefficient in terms of using available bandwidth.

2.4.2 Audio Frequency Shift Keying (AFSK)

AFSK is a modulation scheme in which the data is represented by changes in the frequency of an audio tone. The changes in the frequency are between mark and space frequencies represented by binary zero and one respectively.

Advantages

- AFSK encoded signals pass through AC-coupled links that are included in most devices designed to carry music or speech.
- Implementing an audio modulation scheme allows us operate on many digital modes that have been developed by amateurs.

Disadvantages

- AFSK is inefficient in terms of using available bandwidth and power.

2.4.3 Quadrature Phase Shift Keying (QPSK)

In QPSK, the data bits to be modulated are grouped into symbols, each containing two bits, and each symbol can take on one of four possible values: 00, 01, 10, or 11. During each symbol interval, the modulator shifts the carrier to one of four possible phases corresponding to the four possible values of the input symbol. In the ideal case, the phases are each 90 degrees apart, and these phases are usually selected such that the signal constellation matches the configuration shown in Figure 2.2.

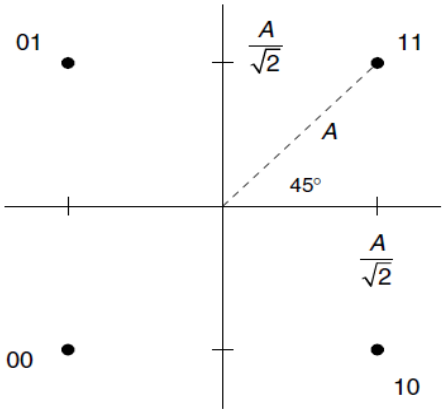

Figure 2.2 : Ideal QPSK constellation

Advantages

- QPSK has excellent spectral efficiency and supports high data rates as compared to BPSK.

Disadvantages

- QPSK is Susceptible to Phase disturbances.

CHAPTER 3 TELECOMMAND RECEIVER DESIGN

The telecommand data link for the proper control of various functions of the satellite is established between the ground station transmitter and on-board telecommand receiver. This receiver demodulates the telecommand data upon reception from the ground station and sends it to OBDH unit which collects this data and issues command to various satellite sub-systems. This Chapter explains the design and the functionality of a LEO satellite telecommand receiver.

3.1 Design Specifications

The design input parameters are taken from a practical communication channel link of satellite. The parameters are listed in Table 3.1:

Table 3.1: Design input parameters of Telecommand receiver

Carrier Frequency	4 MHz
Data Rate	1 Mbps
Modulation Type	BPSK
Bit Error Rate	10^{-6}
Signal to Noise ratio (SNR)	48 dB
Maximum Doppler Shift Compensation	± 200 kHz

3.2 Possible Design Approaches

In PSK modulated transmission systems, the incoming modulated data pulses to the radio receiver can be demodulated either coherently (synchronously) or non-coherently (differentially). Based on this difference there are two basic types of receivers:

1. Coherent Receiver
2. Non-coherent Receiver

3.2.1 Coherent Receiver

In a coherent receiver the received RF signal is coherently demodulated by first multiplying it with a local oscillator (LO) signal which exactly has the same frequency and phase as the transmitted carrier signal. The information about the carrier frequency and phase is acquired by recovering the carrier signal from the received modulated RF signal. This process is

commonly referred to as **"Carrier Recovery"** or **"Carrier Synchronization"**. After the carrier is recovered, it is multiplied with the modulated RF signal by a product demodulator to convert it directly to baseband. The baseband signal is then applied to an optimum binary detector for timing synchronization and finally, data recovery. A generic block diagram of a coherent receiver is shown in Figure 3.1.

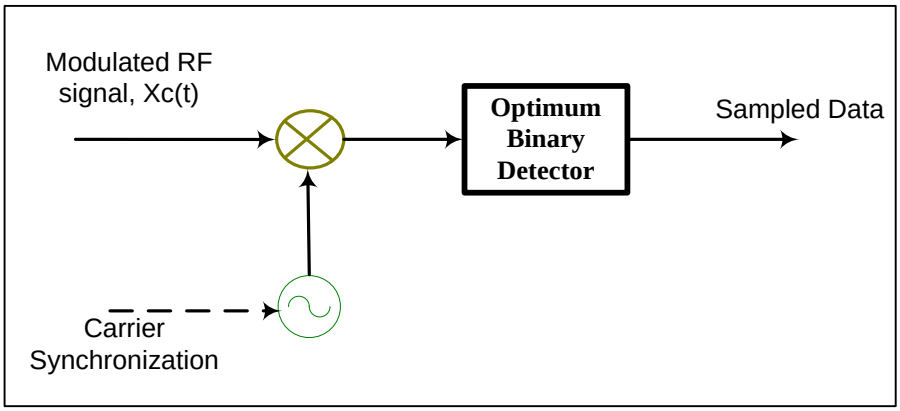

Figure 3.1: Block diagram of a coherent receiver

Theoretically, coherent detection can be achieved by product demodulation. In practice it can be quite difficult to implement. The crux of the problem is synchronization- synchronizing an oscillator to a sinusoid that is not even present in the incoming signal if the carrier is suppressed (as is the case in BPSK modulation). To facilitate the matter, suppressed-carrier systems may have a small amount of carrier reinserted in the modulated signal at the transmitter. This pilot carrier is picked off at the receiver by a narrow band pass filter, amplified, and used to synchronize the receiver's LO. Another approach is to recover the carrier from the received modulated signal by employing techniques such as phase-locked loops (PLL), Costas loops, squaring loops etc. In both cases, it adds complexity to the overall system. Furthermore, perfect synchronism is rarely achieved and some degree of asynchronism must be expected in synchronous detectors. Sometimes, even this small amount of asynchronism can lead to a large degradation in the Bit Error Rate (BER).

3.2.2 Non-Coherent Receiver

In a non-coherent receiver there is no need to synchronize the locally generated carrier with the transmitted carrier. Instead, the data pulses at the transmitter end are differentially encoded prior to PSK modulation. The receiver thus accepts differentially encoded, PSK modulated data centered at the carrier frequency. At the receiver end the data is differentially demodulated using a delay-multiply loop. Thus this type of receiver is referred to as a "Non-

coherent PSK" or "Differentially coherent Phase Shift Keying (DPSK)" receiver. The block diagram of a DBPSK receiver is shown in Figure 3.2.

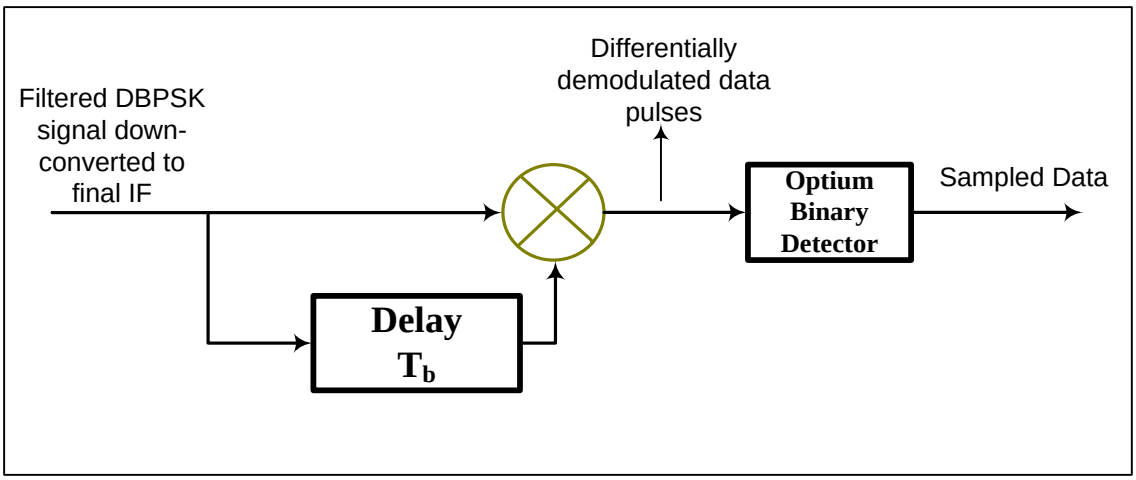

Figure 3.2: Block diagram of a DBPSK receiver

3.3 Reasons for selecting the approach

We selected the coherent approach for the telecommand receiver. Although, the coherent receiver is more complex in design as compared to Non-coherent, but it uses PSK instead of DPSK which offers various valuable advantages. PSK have low probability of bit error (P_b) performance than DPSK receivers at a same E_b/N_0, which is a very critical parameter in satellite and other outer space applications. A typical plot of P_b vs. E_b/N_0 for different modulations is shown in Figure 3.3.

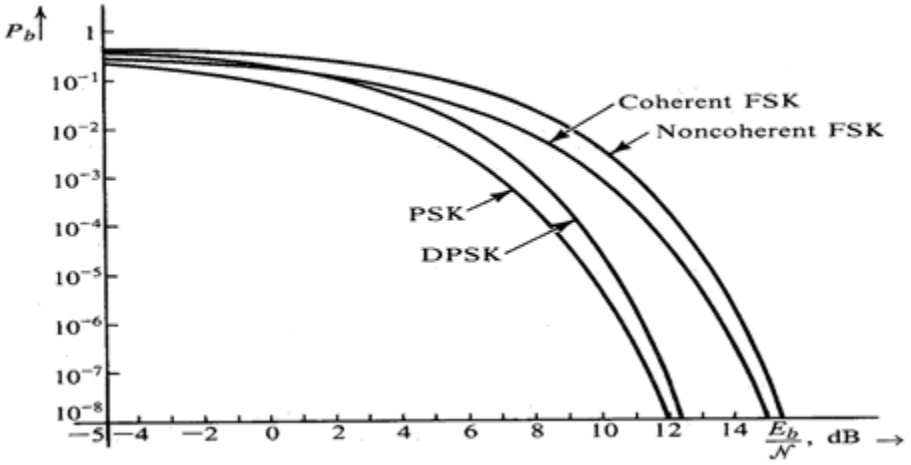

Figure 3.3: Error probability of coherent PSK, DPSK, and coherent FSK

It is also evident from the graph, that the PSK have good BER as compared to DPSK.

3.4 Telecommand Receiver

The architecture of the telecommand receiver is shown in Figure 3.4. It consists of carrier Recovery, Integrator, Bit Synchronizer and Decision blocks. The description and working of each block is presented in the following section.

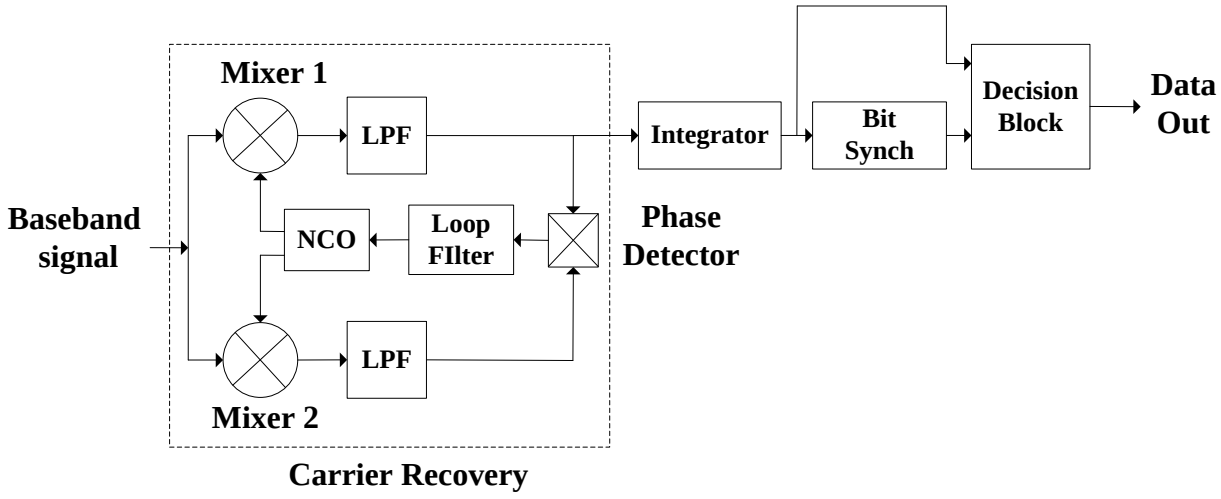

Figure 3.4: Telecommand Receiver Architecture

3.4.1 Carrier Recovery

Coherent detection and demodulation needs synchronization measures for extracting the carrier phase and frequency from the receivers signal. The synchronization schemes such as Phase locked Loops (PLLs) use phase and frequency to extract the information from the received signal. By adding few additional components in the PLL, we can directly obtain the demodulated signal from it. But in suppressed carrier modulation technique, PLL suffers a problem. In the absence of carrier, PLLs are not able to synchronize and track the incoming signal [14]. Therefore, Costas Loop is used instead of PLL for demodulated suppressed carrier modulated signal.

The Costas loop was first proposed by John Costas in 1956 to accomplish phase acquisition, signal tracking, synchronization and demodulation of double-sideband suppressed-carrier (DSB-SC) AM signals [15]. Although the original goal of the Costas loop was to track and demodulate DSB-SC AM signals, it can also be used to demodulate various other suppressed-carrier modulation. Costas loop is also readily used to demodulate the BPSK signals. This can be done due to the fact that BPSK signals can be articulated and demodulated as a DSB-SC AM signals [14].

Costas loop is a variation of phase locked loop and is used for carrier recovery. In actual, it is a combination of two PLL's operating in phase orthogonal to each other. The recovered carrier is used to demodulate and extract the data from the received BPSK signal. It provides an excellent performance for BPSK and is one of the most efficient binary data modulation schemes in terms of noise immunity per unit bandwidth [16], [17].

The traditional Costas loop is severely affected by the phase imbalance between the in-phase branch and the quadrature branch. There are some inherent problems of analog Costas loop for instance direct current zero excursion and complexity to debug [18]. These shortcomings of analog Costas loop can be overcome by using digital implementation of Costas loop [19]. The digital design of the system is realized by the transformation of analog components of the Costas loop to its corresponding discrete time and digital domain [20], [21]. The Costas loop used for BPSK demodulation is shown in Figure 3.5.

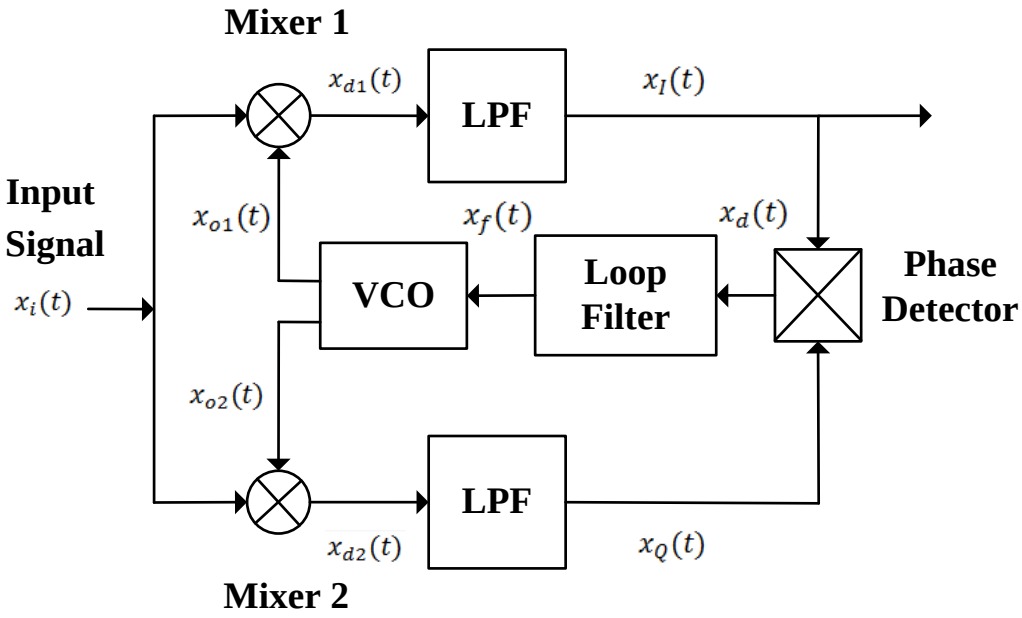

Figure 3.5: Costas Loop

It is composed of mixers (i.e. multipliers), Low Pass Filters (LPF), Voltage Controlled Oscillator (VCO), Loop Filter (LF) and a Phase Detector (PD). The incoming BPSK modulated signal is applied to the two mixers of the upper branch termed as in-phase channel or I-Channel and the lower branch termed as quadrature phase channel or Q-Channel. The VCO generates two orthogonal sinusoid signals (i.e. Sine and Cosine) for the mixers of I-Channel and Q-Channel. The multiplier outputs of I and Q-channel are pass through a low pass filters. The filtered signals are applied to the phase detector which calculates the phase difference termed as "Error signal" between the both channels. The loop filter smoothens the

error signal by filtering its high frequency components which is then used to control the VCO's output phase and frequency. Once the VCO's output frequency and phase becomes equal to the incoming signal's phase and frequency, the demodulated signal is obtained from the I-channel.

The input to the system shown in Figure 1 is $x_i(t)$ defined in equation (3.1) as a BPSK sinusoidal signal with amplitude A_i and a digital modulating signal $m(t)$. The sine wave has a frequency ω_i and phase θ_i. $x_{o1}(t)$ as shown in equation (3.2) is the first output signal of VCO that is in-phase with the incoming signal, whereas $x_{o2}(t)$ as shown in equation (3.3) is in quadrature-phase to the incoming signal. $x_i(t)$ is multiplied to $x_{o1}(t)$ and $x_{o2}(t)$ at the upper and lower-side mixers, respectively.

$$x_i(t) = A_i m(t) \sin(\omega_i t + \theta_i) \tag{3.1}$$

$$x_{o1}(t) = A_0 \, sin(\omega_i t + \theta_0) \tag{3.2}$$

$$x_{o2}(t) = A_0 \cos(\omega_i t + \theta_0) \tag{3.3}$$

The signals after multiplications can be obtained by using trigonometric identities as shown in equation (3.4) and (3.5).

$$x_{d1}(t) = \frac{A_i A_0}{2} m(t) \cos(\theta_i - \theta_0) - \frac{A_i A_0}{2} m(t) \sin(2\omega_i t + \theta_i + \theta_0) \tag{3.4}$$

$$x_{d2}(t) = \frac{A_i A_0}{2} m(t) \sin(\theta_i - \theta_0) + \frac{A_i A_0}{2} m(t) \sin(2\omega_i t + \theta_i + \theta_0) \tag{3.5}$$

The high frequency components of equations (3.4) and (3.5) are effectively removed by the low pass filters of I and Q channel to get the signals $x_I(t)$ and $x_Q(t)$ as shown in equation (3.6) and (3.7) respectively, which are multiplied to obtain $x_d(t)$ as mentioned in equation (3.8).

$$x_I(t) = \frac{A_i A_0}{2} m(t) \cos(\theta_i - \theta_0) \tag{3.6}$$

$$x_Q(t) = \frac{A_i A_0}{2} m(t) \sin(\theta_i - \theta_0) \tag{3.7}$$

$$x_d(t) = \frac{[A_i A_0 m(t)]^2}{8} \sin[2(\theta_i - \theta_0)] \tag{3.8}$$

The signal $x_d(t)$ is passed through the loop filter to get the error signal $x_f(t)$ as shown in equation (3.9).

$$x_f(t) = \frac{[A_i A_0]^2}{8} \sin[2(\theta_i - \theta_0)] \tag{3.9}$$

Replacing, $[A_i A_0]^2/8$ with k_d , the phase detector gain, and using small signal approximation equation (3.9) can be re-written as

$$x_f(t) = k_d(\theta_i - \theta_0) \tag{3.10}$$

Equation (3.10) implies that the Costas loop will successfully track the phase error when it is at or very close to the lock state. If all parameters are correctly estimated, the loop will be in locked state and the phase error will approach to zero. Then equation (3.6) and equation (3.7) becomes equation (3.11) and equation (3.12).

$$x_I(t) = \frac{A_i A_0}{2} m(t) \cos(0)$$

$$x_I(t) = \frac{A_i A_0}{2} m(t) \tag{3.11}$$

$$x_Q(t) = \frac{A_i A_0}{2} m(t) \sin(0)$$

$$x_Q(t) = 0 \tag{3.12}$$

The equations above show, when the Costas loop is in the locked state, the required modulating signal $m(t)$ can be acquired from the I- Channel.

3.4.2 Integrator

The demodulated output from the I-Channel is applied to the Integrator module as shown in Figure 3.4. The integrator integrates the signal over one bit duration. This converts the demodulated signal into a triangular waveform. This waveform is then used by the bit synchronizer and sampling module for recovering the data bits.

3.4.3 Bit Synchronizer

A digital communication system requires various timing control mechanisms for specific purposes. For example, timing information is needed to identify the rate at which bits are transmitted. It is also needed to identify the start and end instants of an information-bearing symbol or a sequence of symbols. We have used Early-Late-Gate synchronizer due to its simplicity in design and less sensitivity towards DC offset [22].

It takes three samples of signal with delays termed as early, late and present. The algorithm used to determine the correct instant for sampling is as follows:

if $((|E| < |P|)$ AND $(|L| < |P|)$ AND $(|P| \geq T_h))$

Peak detect=1

Different instances of sampling are shown in Figure 3.6. After sampling the bit, it is compared with the pre-stored preamble bits. If the incoming bit sequence is matched with the pre-stored preamble sequence, the data sampler gets enabled.

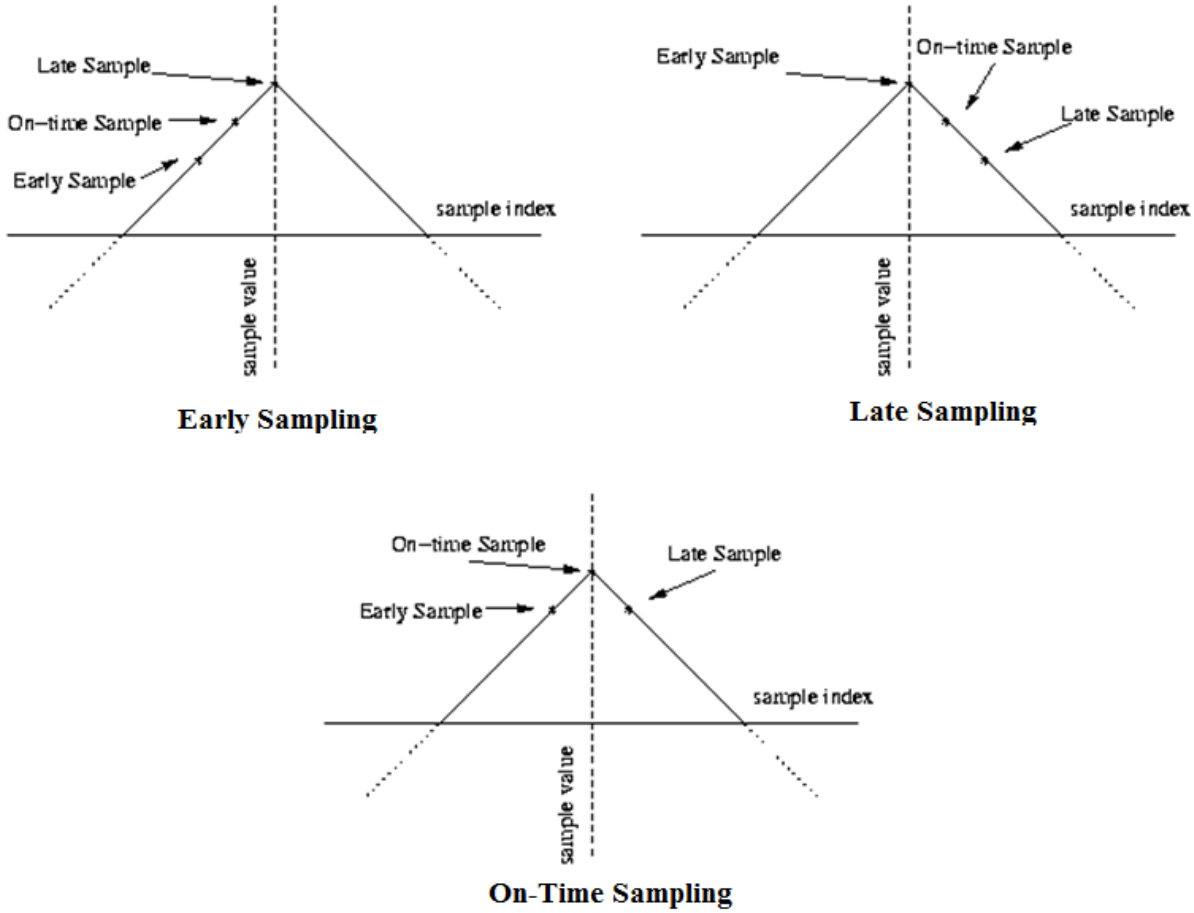

Figure 3.6: Sampling Instances for Early late gate Synchronization

3.4.4 Data Sampler

The data sampler, samples the demodulated signal of carrier recovery on the basis of the sampling instant provided by the bit-synchronizer.

CHAPTER 4 FIELD PROGRAMMABLE GATE ARRAYS

4.1 Introduction

A Field Programmable Gate Array (FPGA) is a digital programmable logic chip. FPGA comprises of two dimensional arrays of logic cells and switches. These logic cells and switched are connected through programmable interconnects. A logic cell is programmed to perform certain actions via a Hardware descriptive Language (HDL). Once the desired logic or function is written in HDL, it is dumped into FPGA by a simple adaptor provided by the vendor. Verilog and VHDL (Very high speed integrated circuit HDL) are the most commonly used hardware description languages. The FPGA are a cost effective solutions as compared to ASICS. Unlike ASICS, FPGA can be updated with a new functionality by reprogramming. The new design or algorithm is tested and validated very easily as compared to ASICS which requires a long fabrication process.

4.2 Functional Overview

FPGA is composed of programmable logic component called “logic blocks” which are joined together in a fabric by reconfigurable interconnects. These logic blocks can be programmed according to the design methodology to realize a variety of hardware functions. These logic blocks also consist of memory elements, which can be a simple flip-flop or more complete memory blocks such as Block RAMs or DSP slices [23].

4.3 FPGA vs. DSP

The implementation of a modem is a digital signal processing issue. Two types of programmable platform could be used to realize the modem, i.e., a Digital Signal Processor (DSP) or an FPGA.

The telecommand receiver is implemented in baseband domain using various Digital Signal Processing (DSP) techniques. There are two types of devices that can be used for realizing DSP algorithms. One is the Digital Signal Processor and other is the FPGA.

FPGA has an advantage of parallel processing when compared to a DSP which is a dedicated processor. As a result of this multi-processing the performance of an FPGA is more than a DSP. FPGA’s are enormously faster, flexible and less expensive.

4.4 FPGA in Space

Reconfigurability and adaptability are one of the most desirable features of modern space technology. FPGA provides this flexibility along with good performance. They have become an integral part of satellite systems for over a decade. Their high computational capacity combined with small size and light weight makes them a preferable choice over other digital systems. The ability to reconfigure FPGA with an updated functionality reduces the hardware requirement in space craft [1].

However, FPGA's face some severe problems in the space environment. The high energy particles in space may interact with memory cells within an integrated circuit and can change their logic state [2]. This alteration may disrupt the operation of a digital system defined by memory cells. FPGAs contain large array of memory cells which makes them more susceptible to single event upsets (SEUs).

In SRAM based FPGAs, a large area is composed of memory cells. These memory cells contain both user data and circuit configuration data that defines the functionality of a system. When high energy charged particles such as neutrons and alpha particles present in the space environment interact with SRAM cells, they occasionally invert there logic state. This phenomenon is called as SEU [2]. The inverted logic state can be both of user data or configuration data and can cause unpredictability in systems behavior. The SEU directly affects the bit error rate performance of a communication receiver. Four general classes of SEUs are identified according to their effect on BER [8].

- In class 1 SEUs, lower order bits of arithmetic operations (such as output of accumulator or coefficient of a filter) are affected. They are 30%-77% of the total SEUs.
- In class 2 SEUs, middle order bits of arithmetic operations are affected. They are 17%-64% of the total SEUs.
- In class 3 SEUs, higher order bits of arithmetic operations are affected. They cause severe degradation in circuit performance and are unacceptable. They constitute 3%-4% of the total SEUs.
- In class 4 SEU, clock distribution, global reset signals MSB of filter or threshold comparator are affected. They are termed as "catastrophic" and reduced BER to ½. They constitute 2%-4% of the total SEUs.

So class 3 and 4 SEUs are more critical and need to be mitigated proficiently. Various techniques have been used in the past to mitigate class 3 and 4 SEUs, the most popular being the TMR technique [8]. In TMR, three replicas of the same circuit are made and they are connected to a voter block which selects the correct output among them. TMR, however, consumes a lot of resources and power. On the other hand a resource efficient alternative to TMR for arithmetic operations is "RPR". In this thesis we have applied RPR to the arithmetic operations involved in the design of a telecommand receiver.

4.5 Reduced Precision Redundancy

In RPR, the full precision (FP) module to be protected is replicated twice with reduced precision (RP) as shown in Figure 4.1. The decision block uses the output of RP modules to determine the error in FP module as follows:

$$\text{if } ((|\mathrm{FP_{out}} - \mathrm{RP1_{out}}| > \mathrm{T_h}) \text{ AND } (\mathrm{RP1_{out}} = \mathrm{RP2_{out}}))$$

$$\text{output} = \mathrm{RP2_{out}}$$

$$\text{else}$$

$$\text{output} = \mathrm{FP_{out}}$$

Threshold (T_h) value is a very critical parameter in RPR. If T_h is very small, false error detection will occur and if T_h value is high, error will not be detected. In order to avoid this problem, the T_h value is set equal to the difference between the FP and RP modules' outputs as shown in equation (4.1) when there is no error.

$$\mathrm{T_h} = |\ \mathrm{FP_{out}} - \mathrm{RP_{out}}\ | \tag{4.1}$$

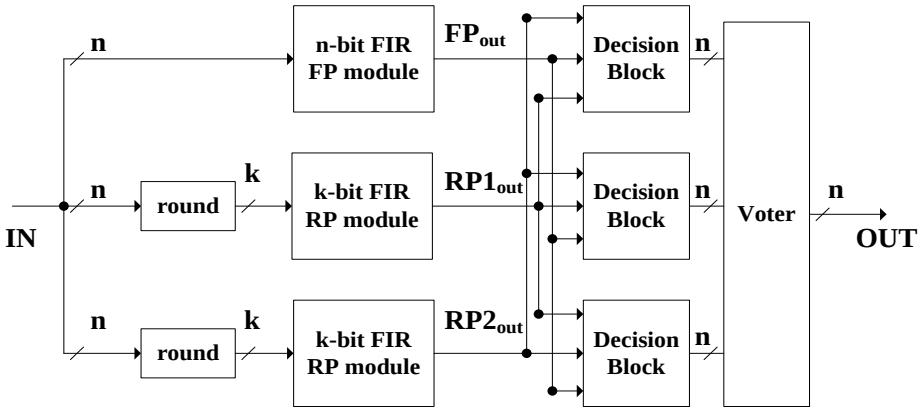

Figure 4.1: Block diagram of an n- bit FIR filter protected with k-bit RP modules

The reduced precision redundancy factor (k) is a tradeoff between mitigation cost and SEU performance [8]. Therefore, the size of RP modules and decision block must be chosen in such a way that they consume less resource than TMR while mitigating SEUs. RPR can be applied to arithmetic operations of any size and complexity. Whether, it is a simple FIR filter or a complex receiver. Unlike TMR, RPR is not suited for every application. It is only applicable to those arithmetic operations that can be approximated with a reduced precision.

4.6 Xilinx FPGA

The architecture of a FPGA is shown in Figure 4.2 below. In our thesis we have used two FPGA's. The first one is Spartan 3E, which is used for the efficient implementation of telecommand receiver. The second FPGA is Virtex 4 which is used for the implementation of the radiation hardened version of the telecommand receiver. Brief overview of both the FPGA families is presented in the next section.

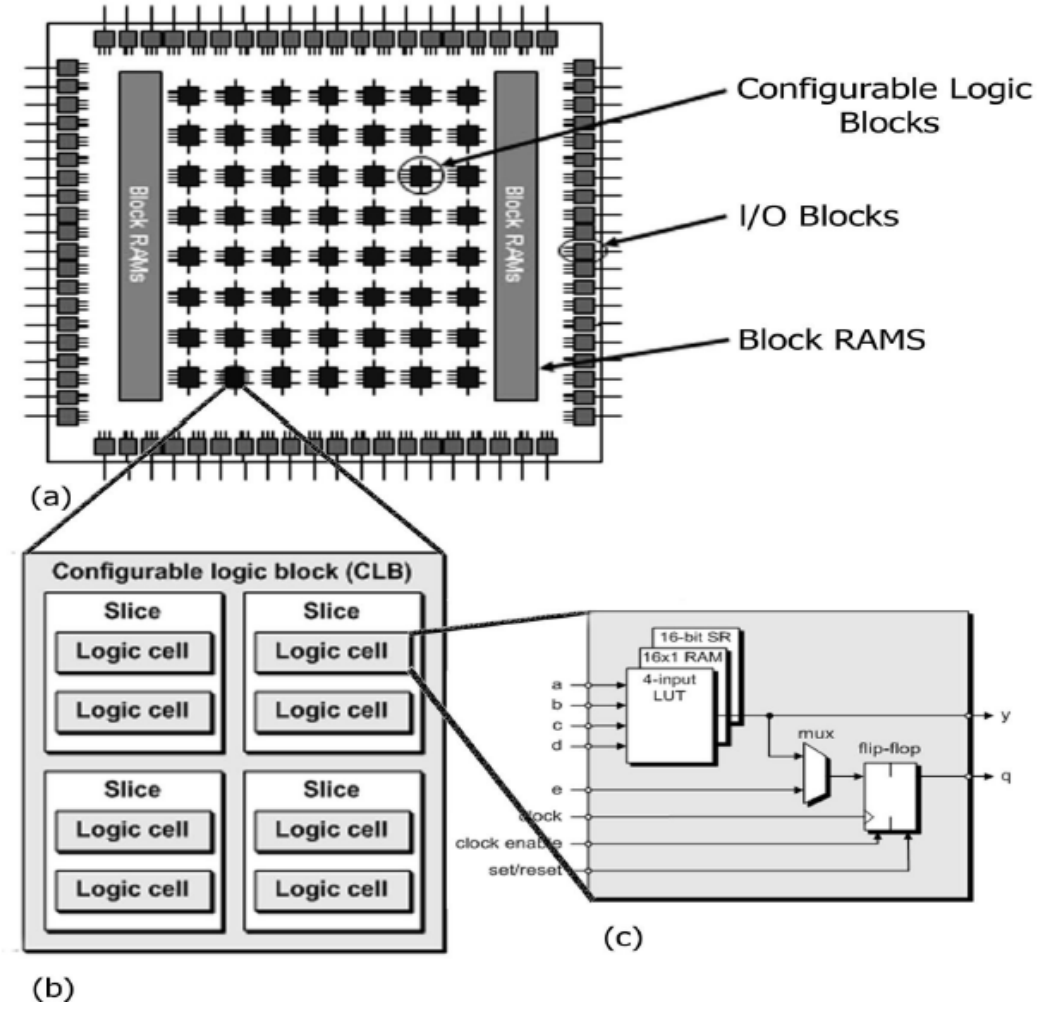

Figure 4.2: (a) Xilinx FPGA (b) Xilinx CLB (c) Simplified View of Xilinx Logic Cell

4.6.1 Overview of Spartan 3E Family

Spartan-3 family of FPGAs is particularly designed to meet the needs of high volume, cost sensitive consumer electronic applications. Since they are relatively cheap, Spartan-3 FPGAs are appropriate to a wide range of consumer electronics applications such as home networking, broadband access, digital television equipment and display/projection [24]. The Spartan-3 family architecture consists of five fundamental building blocks:

- Configurable Logic Blocks (CLBs)
- Input/output Blocks (IOBs)
- Block RAM (BRAM)
- Multiplier blocks
- Digital Clock Manager (DCM)

These logic blocks can be joined together by a programmable interconnect architecture. Spartan 3E FPGA is also consists of some dedicated functional blocks, such as Block Random Access Memories (BRAMs) and Digital Signal Processors (DSPs). These specialized blocks perform several flexible yet specific tasks, and provide a lot of ease to the programmer. The device selection for our platform is XCS3E500 from Spartan 3E family. The device nomenclature can be evaluated from the Figure 4.3: .

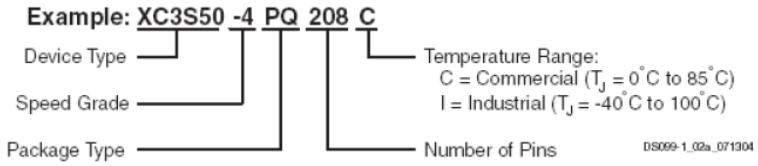

Figure 4.3: FPGA Nomenclature

The kit we have used for telecommand receiver implementation is Spartan 3E XCS3E500E-4FG320.

4.6.2 Overview of Virtex 4 Family

The Virtex-4 family from Xilinx greatly enhances programmable logic design capabilities as compared to previous Virtex and Spartan series and making it a powerful alternative to ASIC technology. Virtex-4 FPGAs offer three platform families—LX, FX, and SX—offering multiple feature choices and combinations to address all complex applications. The wide array of Virtex-4 FPGA hard-IP core blocks includes the PowerPC processors, tri-mode

Ethernet MACs, 622 Mb/s to 6.5 Gb/s serial transceivers, dedicated DSP slices, high-speed clock management circuitry, and source-synchronous interface blocks [25]. The software defined radiation hardened version of the telecommand receiver is implemented using Virtex 4 XC4VSX55-10FF1148.

CHAPTER 5 DESIGN METHODOLOGY

The approach used to design, simulate and verify the complete system is shown in Figure 5.1. First the system was designed in floating point environment using high level design tools (MATLAB/Simulink). After simulating and verifying the algorithms. The design was converted into fixed point. The fixed point system was simulated under system generator and then the Verilog code was generated. The code was synthesized and implemented on FPGA using Xilinx ISE.

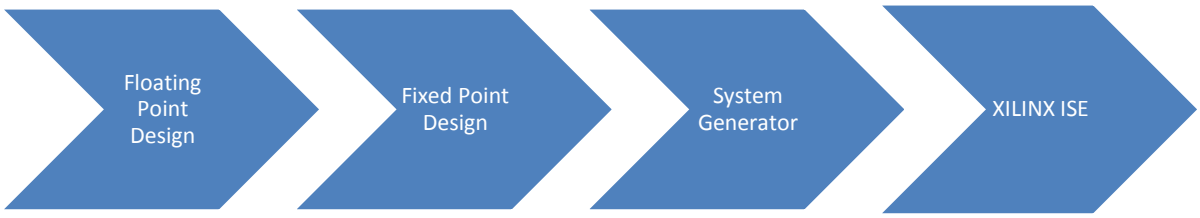

Figure 5.1: System design flow

5.1 High level Simulation

The concepts and algorithms used in the receiver system can be modeled using high level design tools (i.e. MATLAB/Simulink). These tools provide the flexibility to simulate, debug and analyze the functionality of each working block. Moreover, they accelerate the design process and assists in verifying the accuracy of the algorithms.

MATLAB is one of the most widely used software for Digital Signal Processing. It has become an integral part of DSP algorithm design and development. MATLAB/Simulink contains several toolboxes and functions for different applications such as aerospace, communications, image processing, signal processing and wavelet processing. Besides so many built-in functions present in MATLAB, the software package contains vector and array-based waveform data at the core of algorithms, which is very appropriate for applications such as image processing and wireless communications.

The Simulink provides a modular environment for multi-domain simulation and Model-Based Design. It enables parallel processing, automatic code generation, simulation and verification of DSP systems. Simulink contains graphical user interface tools, dedicated libraries, and different types of simulation solvers for modeling dynamic systems. It runs with MATLAB, which enables to integrate MATLAB's algorithms into Simulink design and then

transfer these results to MATLAB for additional analysis. In our work, we have used MATLAB/SIMULINK as a high level design and development tool.

Xilinx's System Generator is a system-level modeling tool that extends Simulink capability to provide FPGA hardware design. It offers high-level of abstractions (i.e. Xilinx System Generator block sets) that can be automatically compiled into an FPGA. System Generator can generate equivalent representations of the Simulink design, at the same or lower level of abstraction. For example, from the functional domain it can generate a structural representation, an HDL or NGC netlist, or a physical representation such as an FPGA configuration bitstream. It can also to generate an equivalent high-level module that performs a specific function in applications external to System Generator (ModelSim hardware co-simulation) [23].

System Generator bridges the gap between Digital Signal Processing algorithms and its FPGA realizations. System Generator is a very useful tool that enables the visualization of data flow and is ideal for modeling and simulating FPGA based DSP algorithms, and enables the designer to generate the VHDL code directly from the design model. It saves considerable time of a DSP developer from rewriting the complete DSP system in Verilog/VHDL.

System Generator automates the design process by enabling debugging, implementation and verification of the design on Xilinx-based FPGAs. It has built-in DSP libraries which enables high level simulation and code generation. It also provides HDL Co-simulation environment, system resource estimator and a hardware co-simulation interface for algorithm validation on FPGA hardware.

System Generator offers mechanisms to:

- To import HDL code into a design. A configuration wizard can be used to associate the HDL module to a Black Box block. The wizard creates an M-function that defines the interface, the implementation and the simulation behavior of the black box block it is associated with.
- To automatically generate an HDL testbench, including test vectors. Upon requested, System Generator generates a testbench that produces files to allow comparisons of simulation results between Simulink and ModelSim (HDL simulator). The testbench is a wrapper that feeds the stimuli to the HDL for the design and compares HDL results against expected ones.

- To perform hardware co-simulations, hardware run under the control of Simulink, bringing the power of MATLAB and Simulink to bear for data analysis and visualization. For hardware Co-Simulation, a bitstream is created and associated to a block. When the design is simulated in Simulink, results for the compiled portion are calculated in hardware.
- System Generator does not substitute hardware design, but permits to design less critical portions with System Generator blocks and then combined them with the other critical ones [24]. Therefore:
- Parts of the design are implemented using System Generator blocks. System Generator employs libraries of intellectual property (IP) to automatically map abstractions onto device primitives.
- Other parts are designed directly in the FPGA using basic functions (adders, registers, memories) and a HDL language. The developed code (VHDL, Verilog) can be imported using wrappers to create Black Boxes. Black boxes are wired into the design, participate in simulations, and are compiled into hardware.
- The complete design is a combination of all the parts (System Generator blocks and imported blocks) into a working whole.

5.2 Hardware Co-Simulation

Hardware co-simulation feature of system generator speed up the simulation and validation of the design on FPGA hardware. Hardware-in-the-loop co-simulation capability eases the design verification process by incorporating the processing power and analyzing tools of MATLAB and Simulink [26].

By using MATLAB/Simulink in conjunction with Xilinx System Generator and the Xilinx ISE synthesis and implementation tool, DSP designs can be implemented on FPGA. As a plug-in to the MATLAB/Simulink software, the Xilinx System Generator creates a precise model of FPGA circuits and automatically generates a synthesizable VHDL code along with the test bench. This synthesized VHDL designs can be used for implementing the designed system on the Xilinx's FPGAs platform. Figure 5.2 shows the implementation of the design process for Hardware Co-simulation.

When the Verilog code has been generated by the Xilinx System Generator software, it is first synthesized and optimized for better implementation results. During the process of

synthesis, the generated HDL code is converted into a logical or physical form that will take the place on FPGA. This implies, the operation of synthesis is to transform the design from Hardware Descriptive Language into gate level. The Verilog modules can be exported to the FPGA hardware by using Xilinx synthesis technology (XST) synthesis tool. The second step is to place and route the design in order to verify that this design will get realize on the FPGA or not. This is accomplished by using the Xilinx's ISE implementation tools. The place and route tool function is to place the synthesized modules into FPGA locations and makes necessary connections between these modules, so that they can operate as an integrated system.

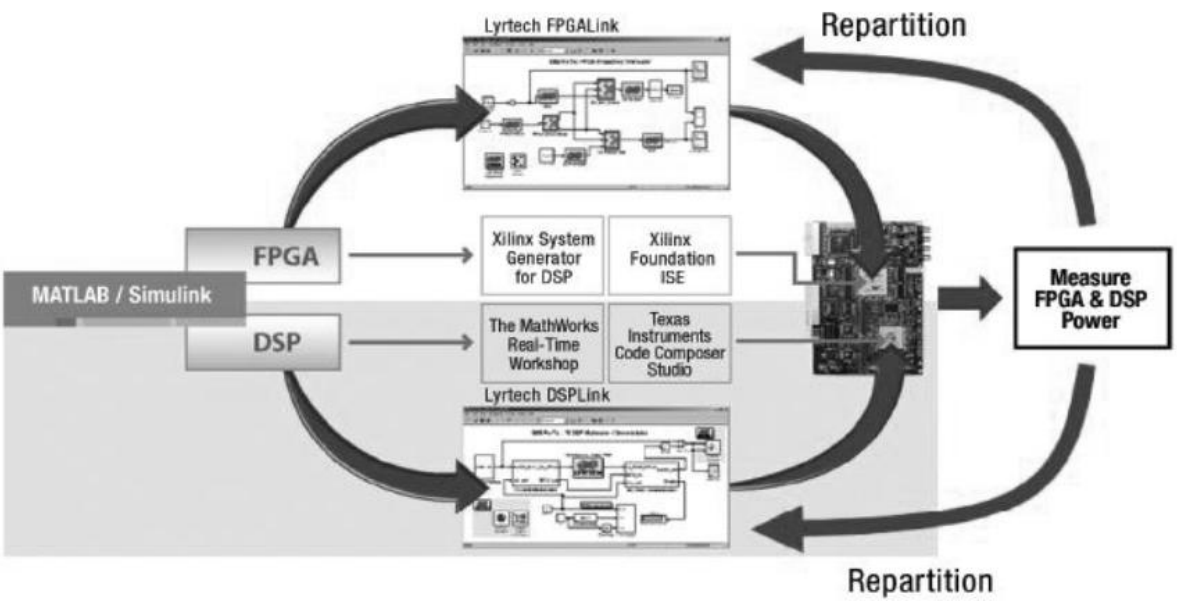

Figure 5.2: Implementation of the design process for Hardware Co-simulation

Placing and routing operation is then followed by hardware verification. The designed module is implemented on the FPGA. In hardware verification step, the module created in high level simulation is checked whether it would work well on the desired FPGA. Test signals are then used to check any difference between the simulation and the hardware implementation.

5.3 Simulation with Simulink and System Generator

In our thesis, the algorithm is first designed and simulated on Simulink and then implemented on Xilinx system generator. The designed receiver correctly demodulated the BPSK modulated signal applied to it. The complete system is designed using basic Simulink

functional blocks instead of its customized blocks, in order to ease the migration from Simulink to system generator.

CHAPTER 6 IMPLEMENTATION

This Chapter presents the Xilinx system generator realization of the Telecommand receiver and its SEU mitigation.

6.1 BPSK Modulator

In order to simulate and verify our receiver design, BPSK modulated signal is generated in MATLAB/SIMULINK. The BPSK modulation is achieved by multiplexing the two sinusoidal signals having 180 degree phase shift in between them using a switch. The binary data to be modulated is imported from workspace and is used as the select signal of switch as shown in Figure 6.1.

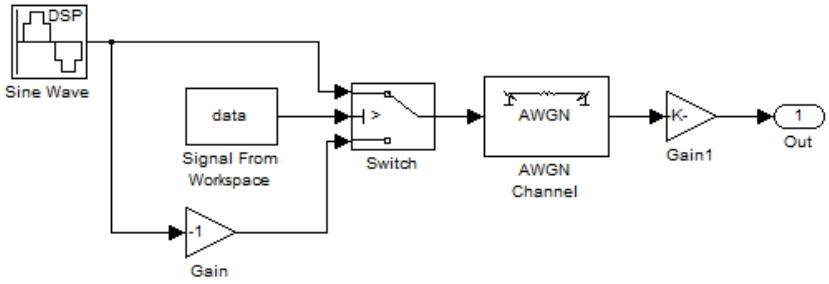

Figure 6.1: BPSK Modulator

The output BPSK modulated waveform along with other signals is shown in Figure 6.2.

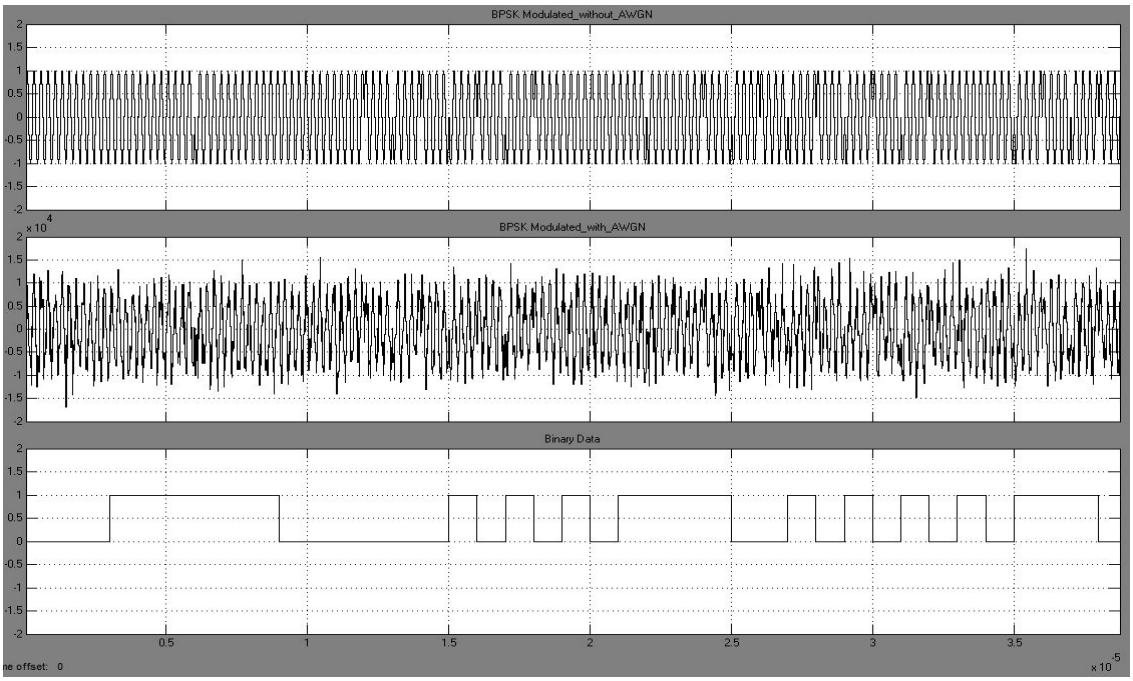

Figure 6.2: BSPK Modulator Output

The modulated signal is then passed through a AWGN Channel block to emulate the channel degradation effects. Finally the modulated signal is multiplied with a fixed gain to achieve the real voltage levels of ADC of the Spartan 3E kit.

6.2 Telecommand Receiver

6.2.1 Carrier Recovery

Carrier recovery is achieved by using Costas loop. Traditional analog Costas loop suffer from several problems such as imbalance between in-phase and quadrature branch, direct current zero excursion and difficulty to debug [18]. These problems can be avoided by using digital version of Costas loop [27] as shown in Figure 6.3.

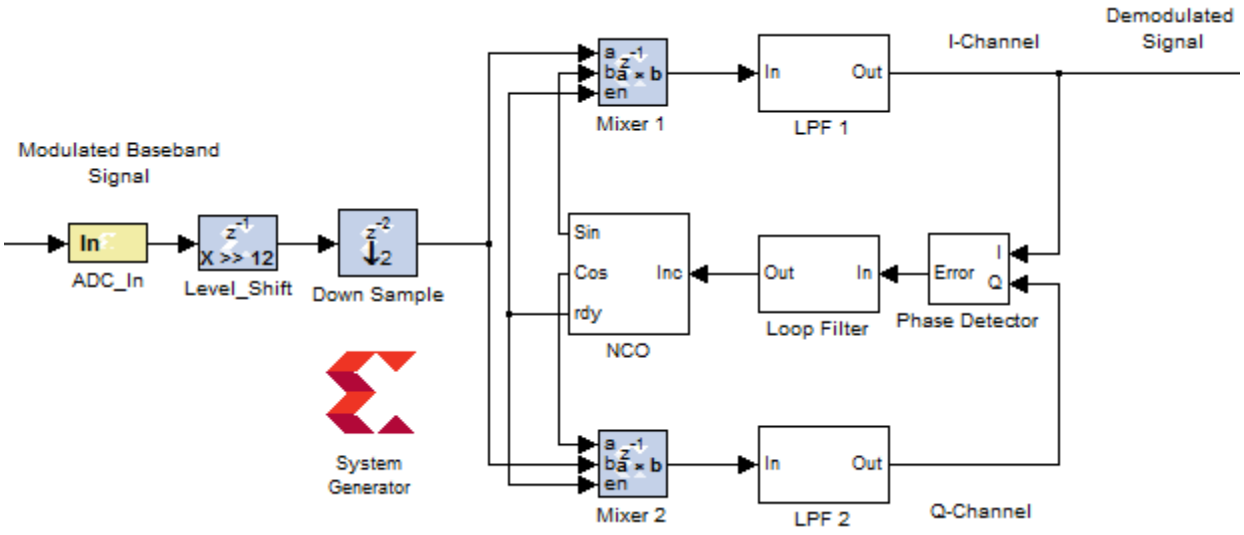

Figure 6.3: Costas Loop

The transmitted signal from Simulink is acquired in system generator environment by using gateway in block labeled as ADC_In. The received signal is of 14 bits and having frequency of 4MHz. It is sampled on 64 MHz and is level shifted to relax the computation requirements as shown in Figure 6.4.

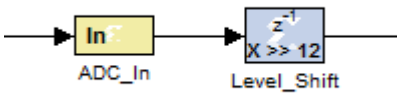

Figure 6.4: Input Interface of Telecommand receiver

The level shifted received signal as shown in Figure 6.5 is applied to the input of Costas loop for carrier recovery and demodulation. Modulated signal is down sampled by a factor of 2. This reduces the sampling frequency to half for the rest of the system thus drastically reduces the logic utilization for the complete design especially for filters.

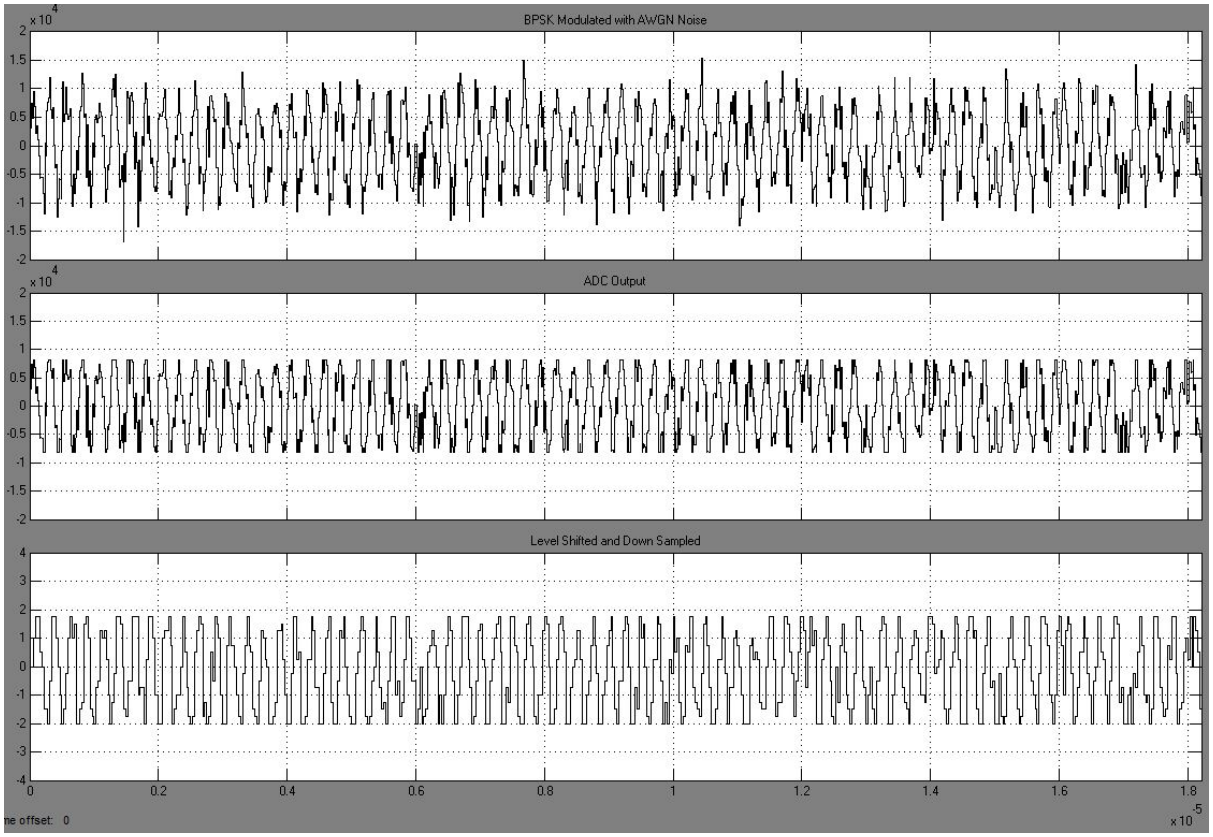

Figure 6.5: (a) BPSK Modulated signal with AWGN (b) ADC_in output (c) Level Shifted signal

The reference signals in the loop are generated by using Numerical Controlled Oscillator (NCO). The NCO is implemented using Direct Digital Synthesizer compiler 4.0 (DDS V4.0) which produces two orthogonal sinusoids. The architecture of NCO is shown in Figure 6.6 it has two distinct parts. First, a phase accumulator accumulates the phase increment and adds in the phase offset. In this stage, an optional internal dither signal can also be added. The NCO output is then calculated by quantizing the results of the phase accumulator section and using them to select values from a lookup table.

For a desired frequency F_0, the phase increment value can be calculated with the following equation

$$Phase\ Increment = \frac{(F_0 . 2^N)}{F_S}$$

Where N is the accumulator word length and

$$F_S = \frac{1}{T_S} = \frac{1}{Sample\ Time}$$

The frequency resolution of NCO is defined by

$$\Delta f = \frac{1}{T_s . 2^N} Hz$$

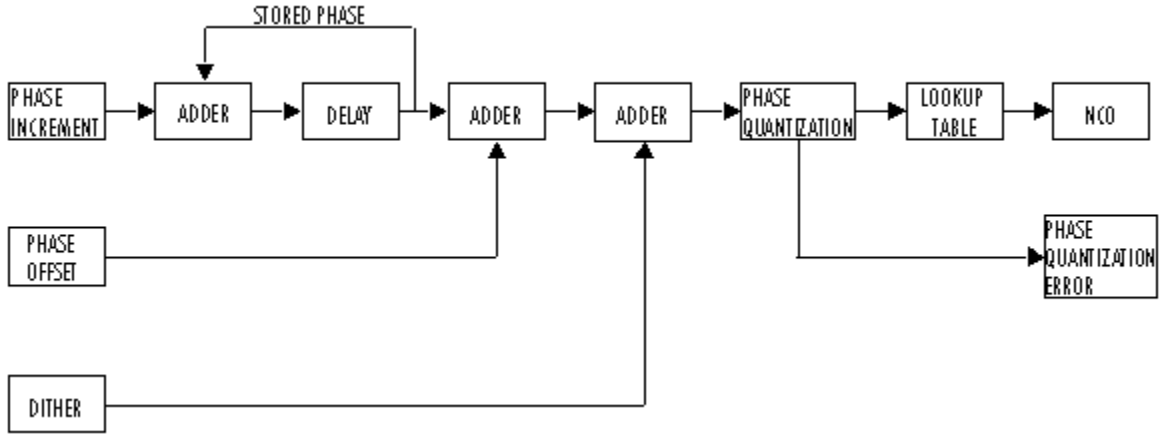

Figure 6.6: Architecture of NCO

The desired phase offset (in radians) can be set by following formula

$$Phase\ offset = \frac{2^N . desired\ phase\ offset}{2\pi}$$

The spurious free dynamic range (SFDR) is estimated as follows for a lookup table with 2^P entries, where P is the number of quantized accumulator bits:

$$SFDR = (6P)\ dB \qquad (Without\ Dither)$$

$$SFDR = (6P + 12)\ dB \qquad (With\ Dither)$$

This block uses a quarter-wave lookup table technique that stores table values from 0 to π/2. Our desired parameters for NCO calculations are:

- Desired output frequency: $F_0 = 4$ MHz
- Accumulator word length: N= 18
- Spurious free dynamic range: SFDR $\geq$ 90 dB
- Sample period: T_S= 1/32e6 s

- Desired phase offset: 0

By using above mentioned formula, phase increment comes out to be,

$$Phase\ increment = \frac{4e6 * 2^{18}}{32e6}$$

$$Phase\ increment = 32768$$

Calculating the number of quantized accumulator bits from the equation for spurious free dynamic range,

$$SFDR = (6P + 12)dB$$

$$96\ dB = (6P + 12)dB$$

$$P = 14$$

Now selecting the number of dither bits. In general, a good choice for the number of dither bits is the accumulator word length minus the number of quantized accumulator bits; in this case 4. After calculating all the parameters for NCO, we configured the DDS Compiler 4.0 of Xilinx system generator. In the basic tab of DDS as shown in Figure 6.7(a) system clock is set to be 32 MHz, Noise shaping is set to *"Phase_Dithering"*. In hardware parameter option, Phase width is set 18 bits and output signal width is set to be 14 bits. Output selection mode is configured to *"Sine_and_Cosine"*. In the implementation tab, the memory type is selected to be *"Block ROM"* because it is fast and optimized. The Optimization goal is set to area, in order to aim resource efficient implementation. Latency is selected to be 1. The optional pin *"rdy"* is checked because it provides the enable signal to the Mixer's when the DDS output is ready. Sample time for the DDS compiler is set to be 1/32e6 in explicit period. Since the frequency of the DDS will be determined by an external data coming from the loop, phase increment programmability is set to *"Programmable"* in the output frequency tab. The summary of configuration settings for DDS is shown in Figure 6.7(b).

The DDS compiler 4.0 data input range is from 0 to 1. Therefore the incoming data word is divided by 18 before applying it to DDS block as shown in Figure 6.8. The division operation is achieved by shift right operator. The write enable pin *"we"* is set to high.

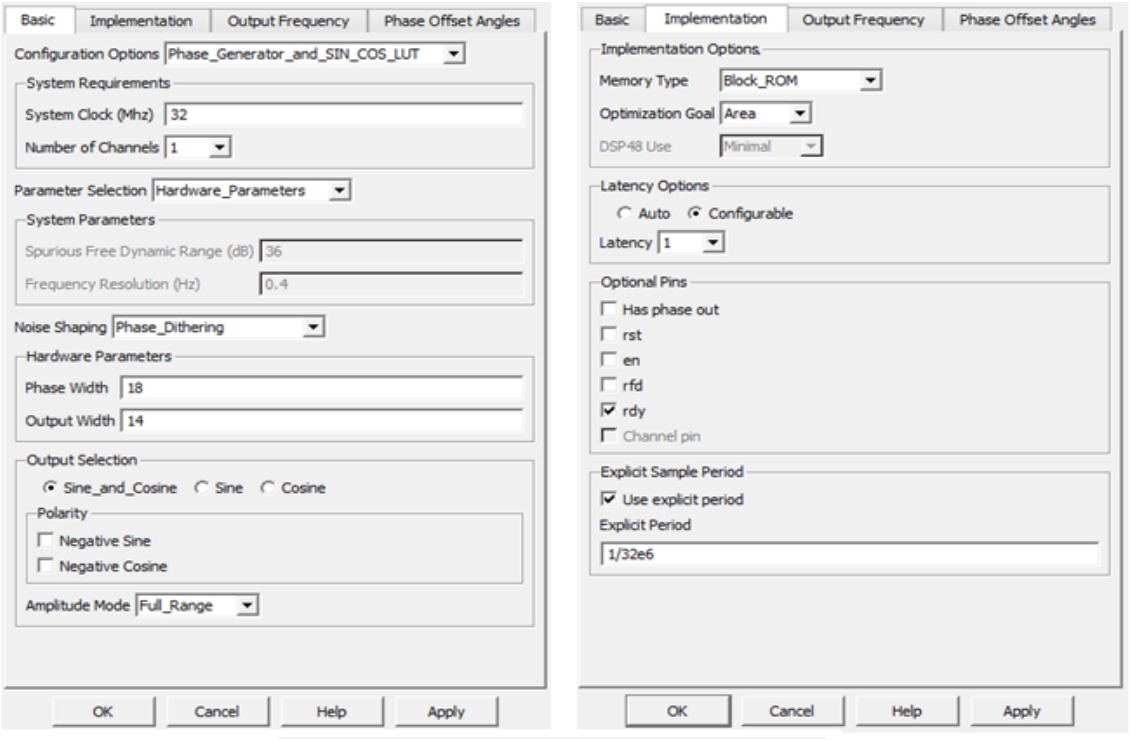

Figure 6.7: DDS Compiler internal Configuration a) Basic b) Implementation

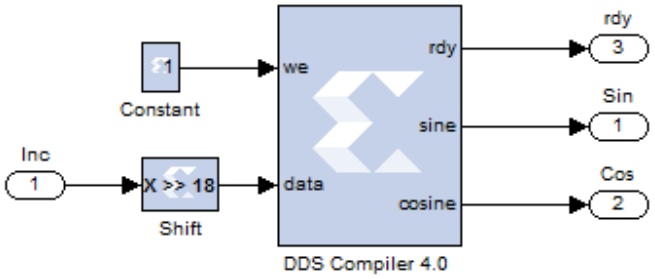

Figure 6.8: DDS compiler 4.0

The Sine, Cosine and ready signal as shown in Figure 6.9 are applied to the Mixer1 and Mixer 2 of both the arms of costas loop.

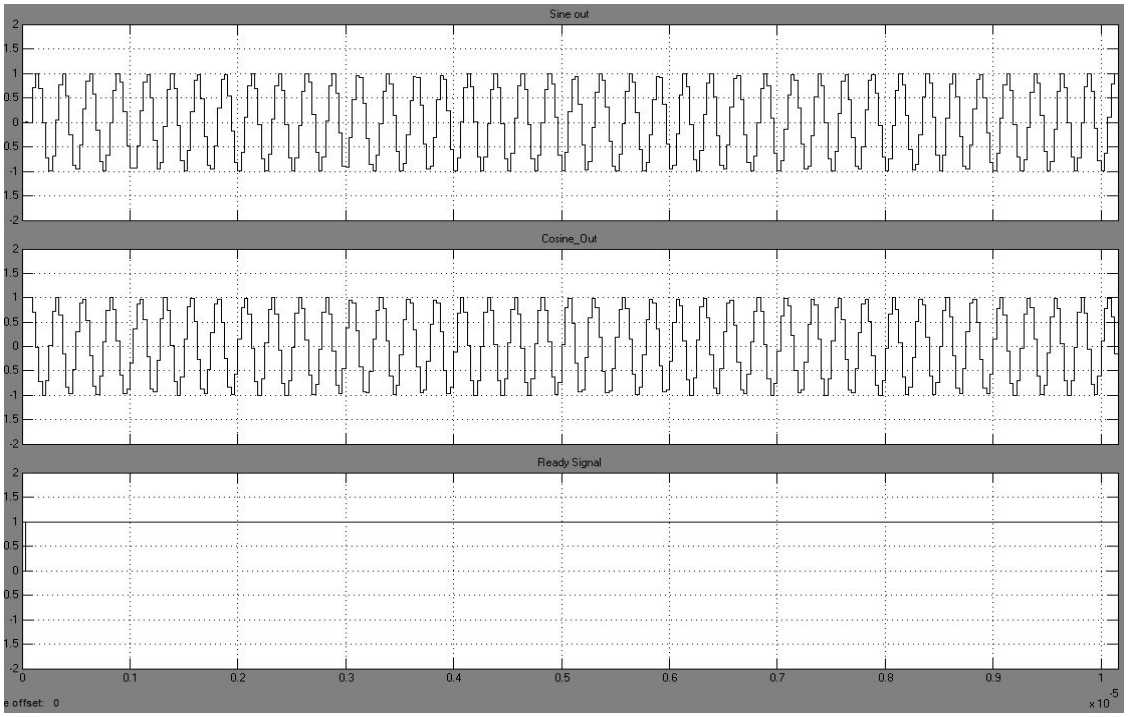

Figure 6.9: NCO Outputs (a) Sine wave (b) Cosine wave (c) ready signal

The arm in which sine wave is applied is called as in-phase (I) arm and where cosine wave is applied is called as quadrature arm (Q). The mixer's are realized by using 14 x 14 bit multipliers. After multiplication, two frequency components are generated. The high frequency components are filtererd by the fifth order Low Pass Filter (LPF). The filter is designed using Direct form symmetric architecture as shown in Figure 6.10. This architecture takes the advantage of symmetry in the coefficients of FIR filter and uses half the multipliers and adders than the conventional approach. The architecture is realized using discrete system generator blocks. This approach enables the in depth optimization of each block thus resulting a very efficient implementation. Moreover, one of the filters coefficients is implemented through binary shift operation which further reduces resource consumption. The low pass filters in each channel are designed wide enough to pass the data modulation without distortion [28]. All the adders used are of 14 bits and two coefficient implemented with *"cmult"* block are of 16 bits. Complete architecture of the filter is pipelined for maximum performance.

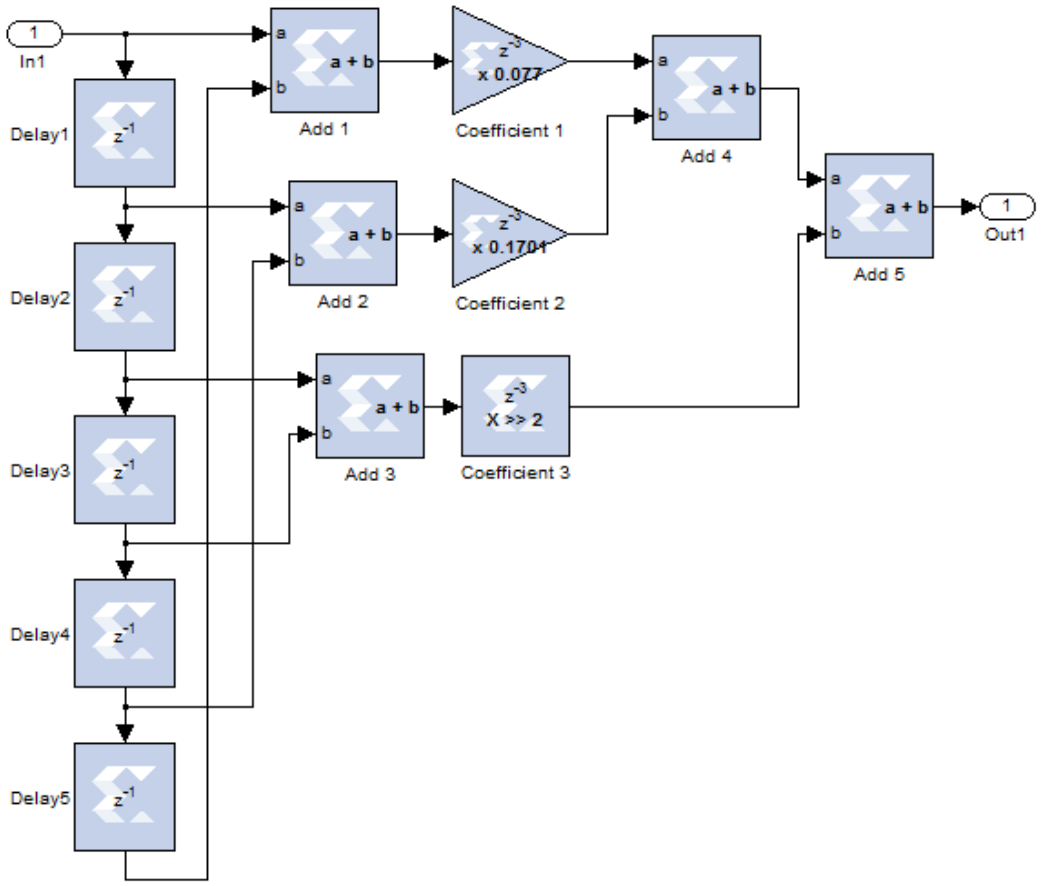

Figure 6.10: Low Pass Filter Architecture

The 16 bits output of I and Q channel filters are applied to the phase detector block. It is implemented by using a 18 x 18 bit multiplier block. It computes the phase error between the I and Q channel and generates the coresponding error signal. This error signal is applied at the input of loop filter. Loop filter removes the high frequency leakages of the phase detector. It provides a smooth and stable 18-bit control word to the NCO for modifying its output frequency and phase with respect to input signal as shown in Figure 6.11. The designing of loop filter is a very sensitive task as it determines the bandwidth of the loop and controls NCO's output. The loop filter is a first order Butterworth IIR filter and is entirely implemented by binary shift registers without using embedded multipliers as shown in Figure 6.12. The incoming error signal is first multiplied with a coefficient, this is achieved by shift left operation and then it is added to the reference word by a 18 bit adder.

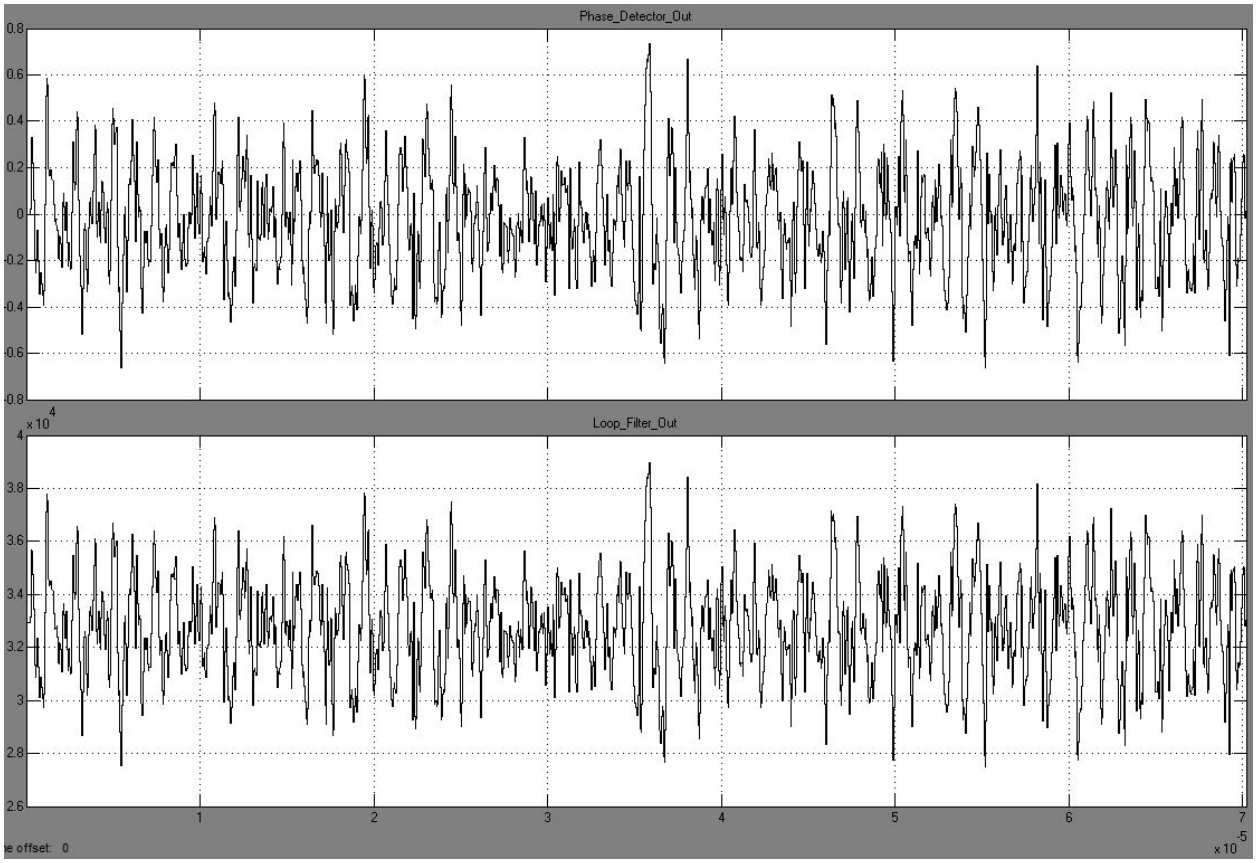

Figure 6.11: (a) Phase Detector Output (b) Loop Filter Output

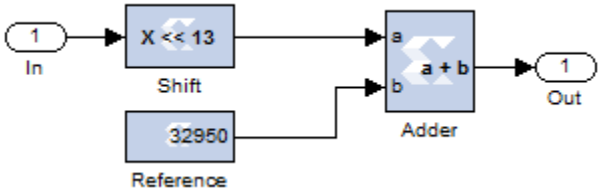

Figure 6.12: Loop Filter

NCO modifies its output frequency with respect to the provided data word. When the NCO's generated carrier frequency and phase gets synchronized with the incoming signals frequency and phase, the demodulated signal is produce at the I channel as shown in Figure 6.13. Our designed Costas loop can demodulate input signal with Doppler shifts up to 10 percent of the carrier frequency.

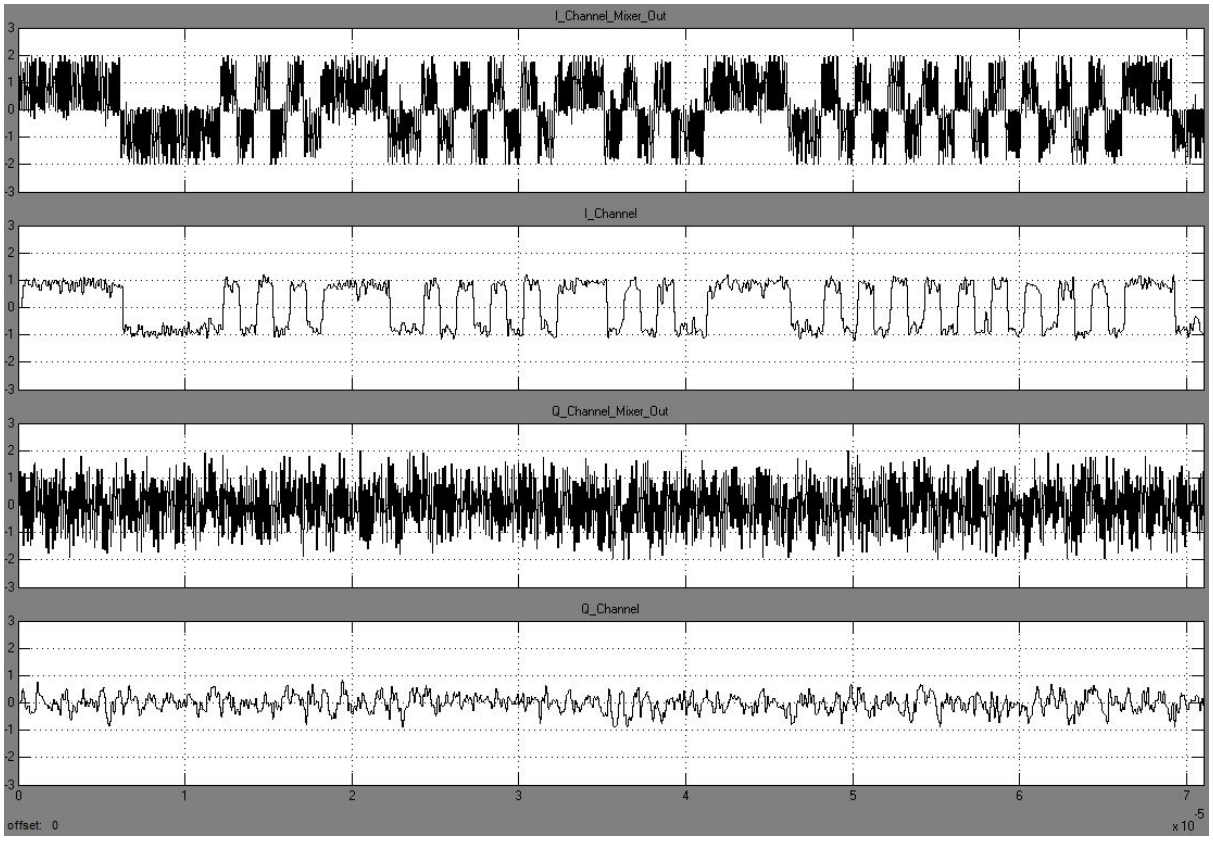

Figure 6.13: (a) I channel Mixer Out (b) I channel Out (c) Q channel mixer Out (d) Q channel Out

6.2.2 Integrator

The demodulated signal from the Costas loop is applied at the input of Integrator block as shown in Figure 6.14.

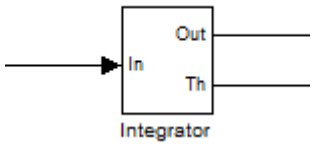

Figure 6.14: Integrator block

This block performs two tasks: it integrates the coming signal over one bit duration and it determines the threshold value which is used by later stages of the design. First, the input data type is converted from 16 bits to 10 bits as shown on Figure 6.15. This reduces the logic consumption of the proceeding blocks.

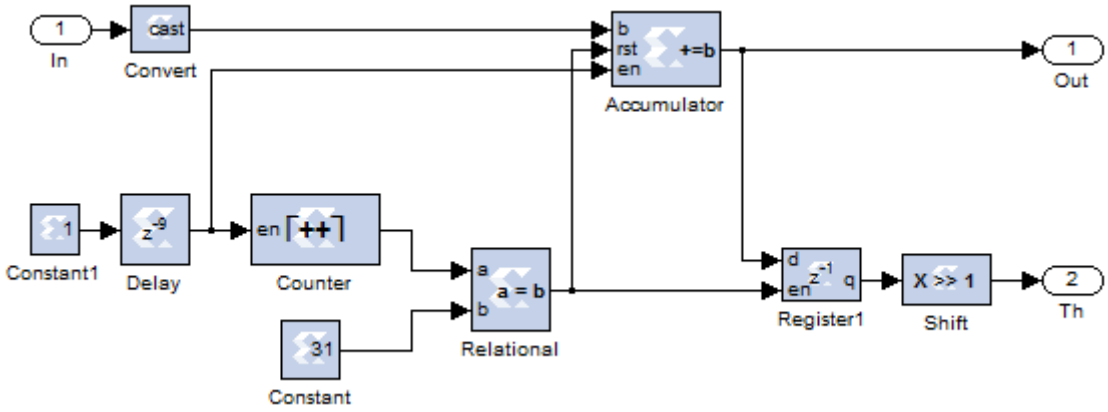

Figure 6.15: Internal logic of Integrator

The demodulated signal samples are added in an accumulator which gets reset after one bit duration. The accumulator operation is selected to add and the number bits are set to be 13. Optional ports of synchronous reset and enable are also checked. The internal configuration of the accumulator block is shown in Figure 6.16.

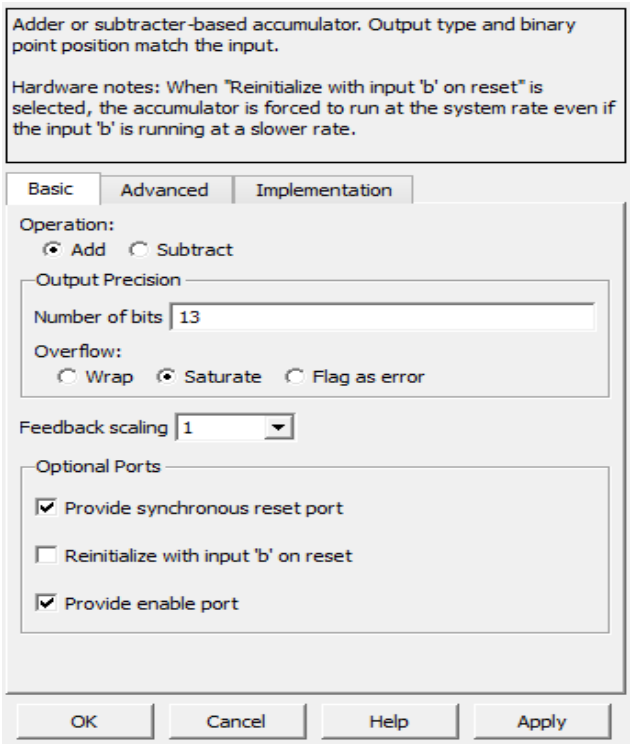

Figure 6.16: Accumulator Configuration

The resetting logic consists of a counter, a constant and a relational operator. There are 32 samples in one bit duration. The counter starts counting from 0 and when it value reaches up to 31, the relational operator equates it with the constant value and generates *"1"* at the its output. This *"1"* resets the accumulator backs to zero. In order to avoid synchronization

issues due to computational delays, the accumulator and the resetting logic is enables after the delays of 9 samples which is the processing time of carrier recovery loop. The resultant waveform is of triangular in shape. This reduces the Inter-Symbol-Interference (ISI) and provides to the peak detecting module.

Before the resetting of accumulator, the peak value is captured in a register. The peak value is divided by two by using shift right operation. The divided value is used as a "Threshold" value by decision blocks used in the later stages of the design. All the internal and external signals of integrator are shown in Figure 6. 17.

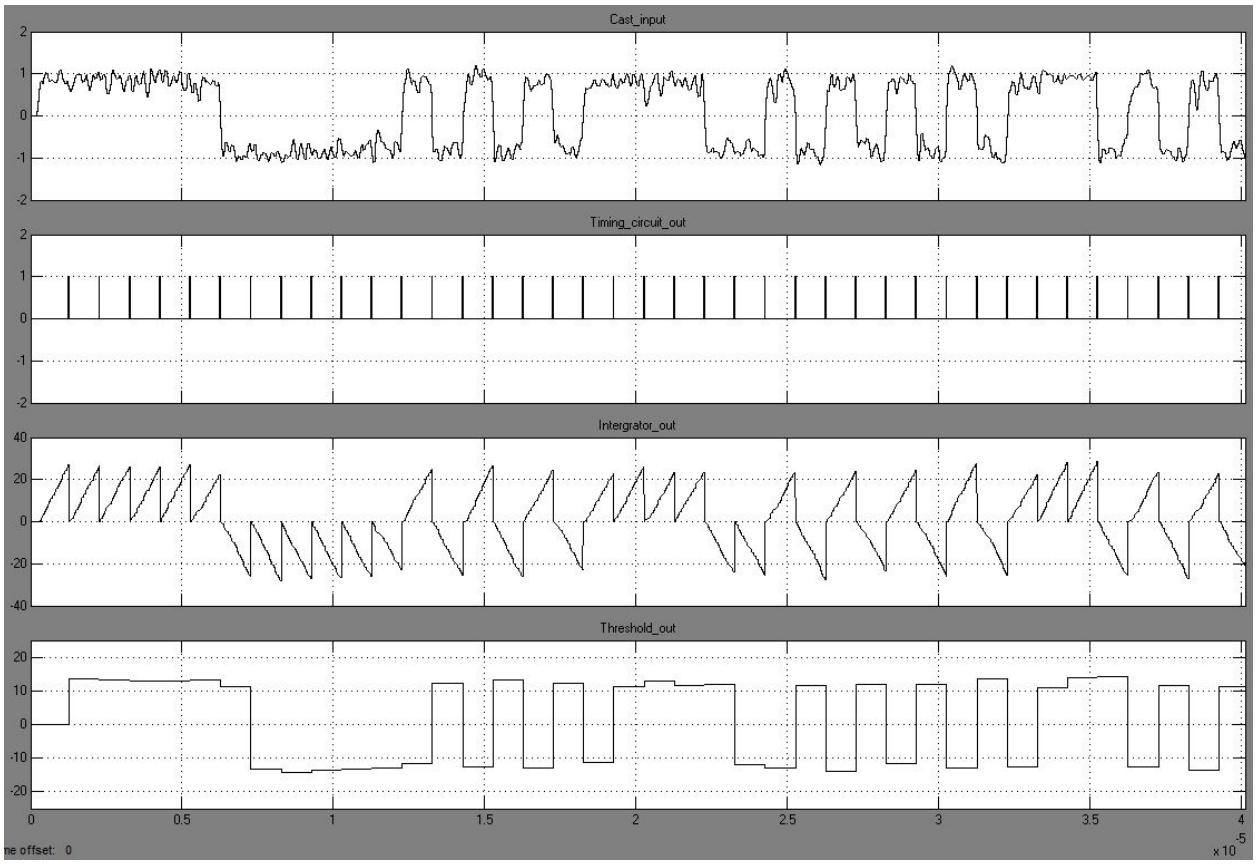

Figure 6. 17: (a) Cast input (b) Timing circuit output (c) Integrator Out (d) Threshold Out

6.2.3 Bit Synchronizer and Data Sampler

The bit synchronizer and data sampler module has two inputs i.e. integrated signal and threshold, and one output port i.e. Data as shown in Figure 6.18.

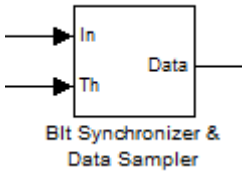

Figure 6.18: Bit Synchronizer and Data Sampler module

It performs three major tasks: peak detection, preamble matching and data sampling. As mentioned in the previous section, the signal from the integrator is a triangular waveform. The signal is first fed into the peak detect module. This block detects the sampling instant for the wave. After that, the signal is passed on to the preamble match block and data sampler block. Data sampler starts sampling the signal once it receives the peak detect *"1"*. The sampled data is sent to preamble match module, where data is compared with the pre-stored preamble sequence. If the recovered data sequence is matched with the pre-stored, peak detect and preamble match modules are disabled using *"Register1"* and *"inverter"* and also an enable signal is sent to the *"Register2"* which routes the sampled data to the output port. The detail design and functional description of each sub-module is presented in the following sections.

6.2.3.1 Peak Detector

The peak detector block determines the sampling instant for the integrated signal. In case of a triangular wave, it is the peak of signal. That's why this module is named as peak detector. The internal logic of the module is shown in Figure 6.19.

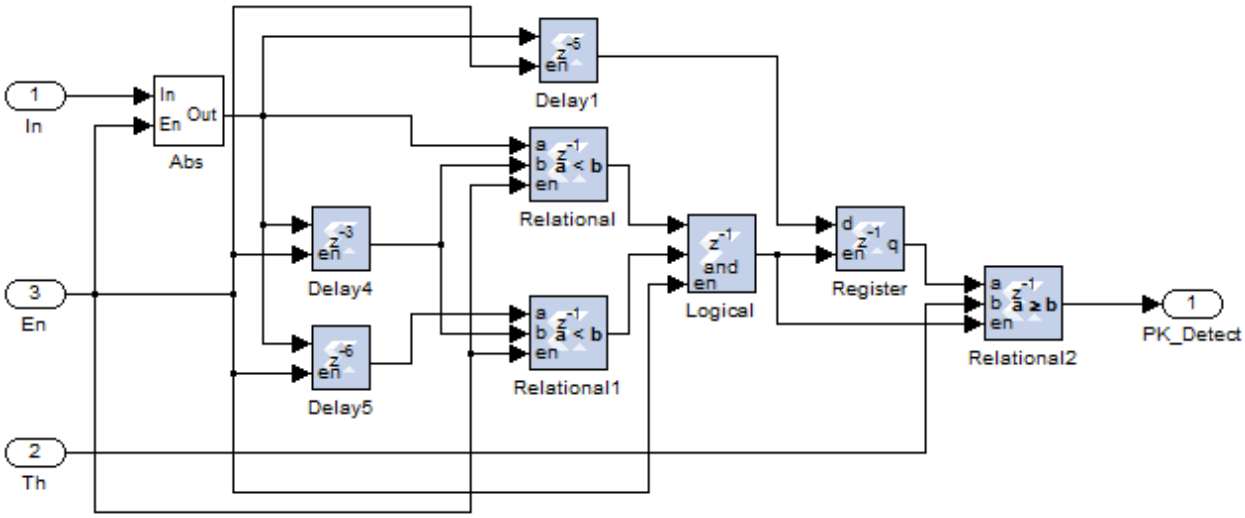

Figure 6.19: Peak detector

It uses the improved early late gate sampling algorithm for determining the peak of the signal. The algorithm is implemented by using relational operators rather than arithmetic operators in order to avoid any floating point arithmetic. This approach saves considerable FPGA resource logic. The incoming signal has a zero mean value i.e. it has a positive peak value and the negative peak value. So in order to apply this algorithm, absolute (abs) function is applied to the incoming signal values. Since the Xilinx system generator does not provide abs function block, we created our own functional module. The abs sub-module contains a multiplexer, a slice and a negate block as shown in Figure 6.20.

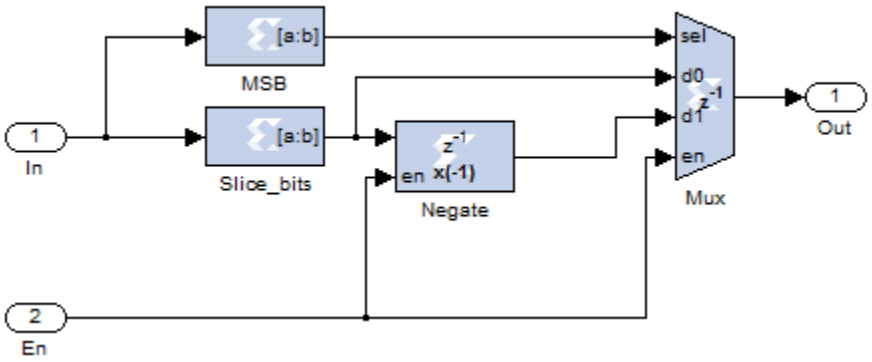

Figure 6.20: Absolute value computation

The Most Significant Bit (MSB) is sliced off the incoming signal and used as the select pin of Mux. The internal setting is shown in Figure 6.21. The incoming signal is of 13 bit wide having binary point on 6. In order to take the MSB, the width of slice bit is set to 1 and offset of bottom bit is set to 12. For the remaining bits, slice_bits setting is shown in Figure 6.22.

The width of slice bit is 7 and offset bottom bit is set to 6 because of the binary point position.

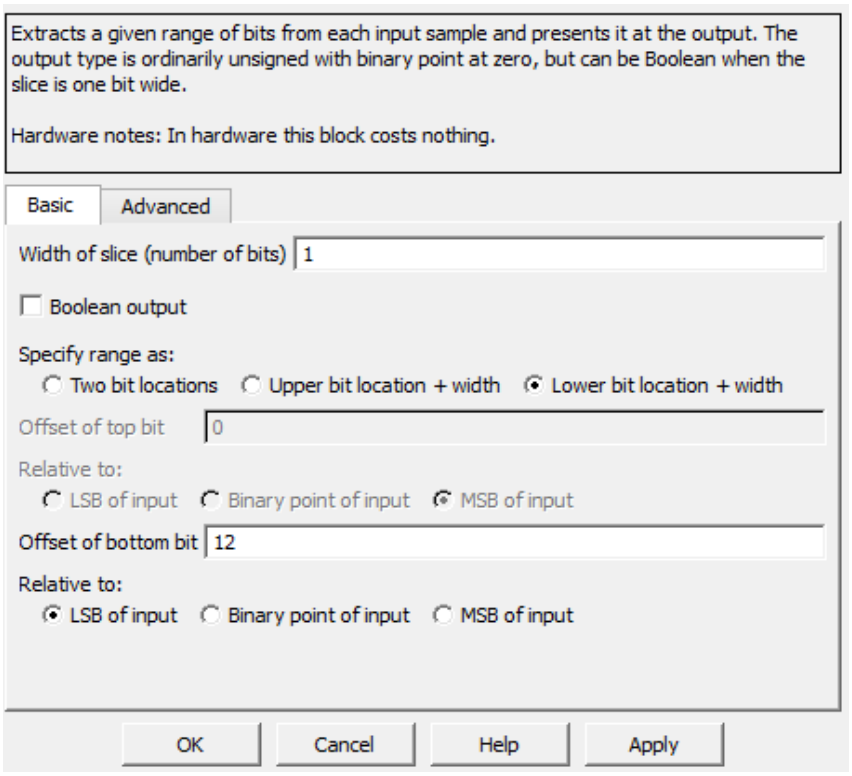

Figure 6.21: MSB slice

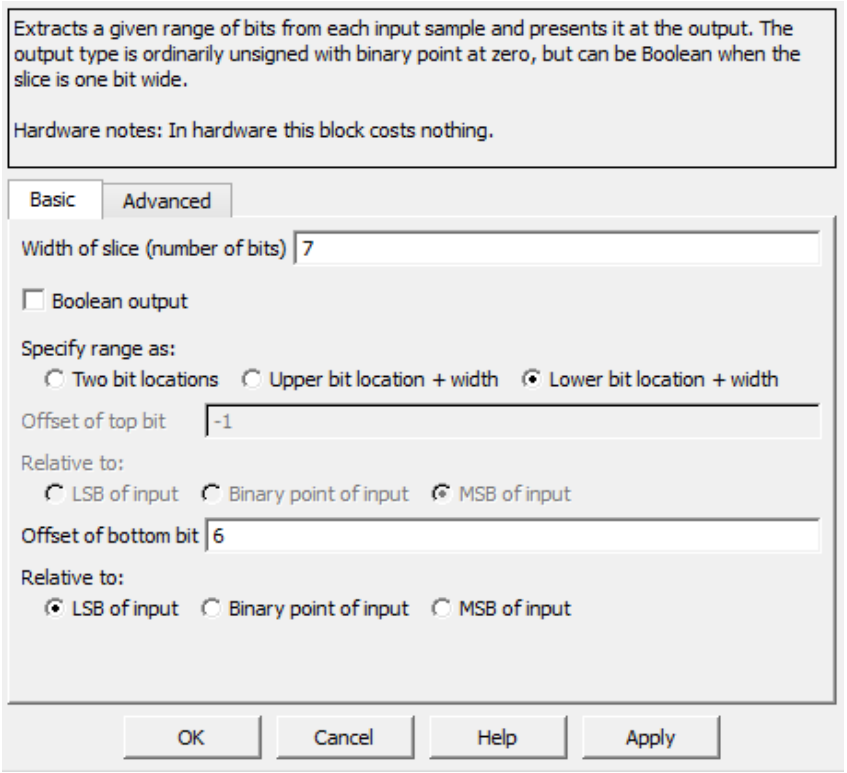

Figure 6.22: Bottom bits slice

The first input of the MUX *"d0"* is connected with sliced bits and the second input *"d1"* is connected with the negated sliced bits. If MSB is *"0"* means the incoming signal is positive, *"d0"* is routed to the output. If MSB is *"1"* means the incoming signal is negative, *"d1"* is routed to the output. The internal signals of abs block are shown in Figure 6.23.

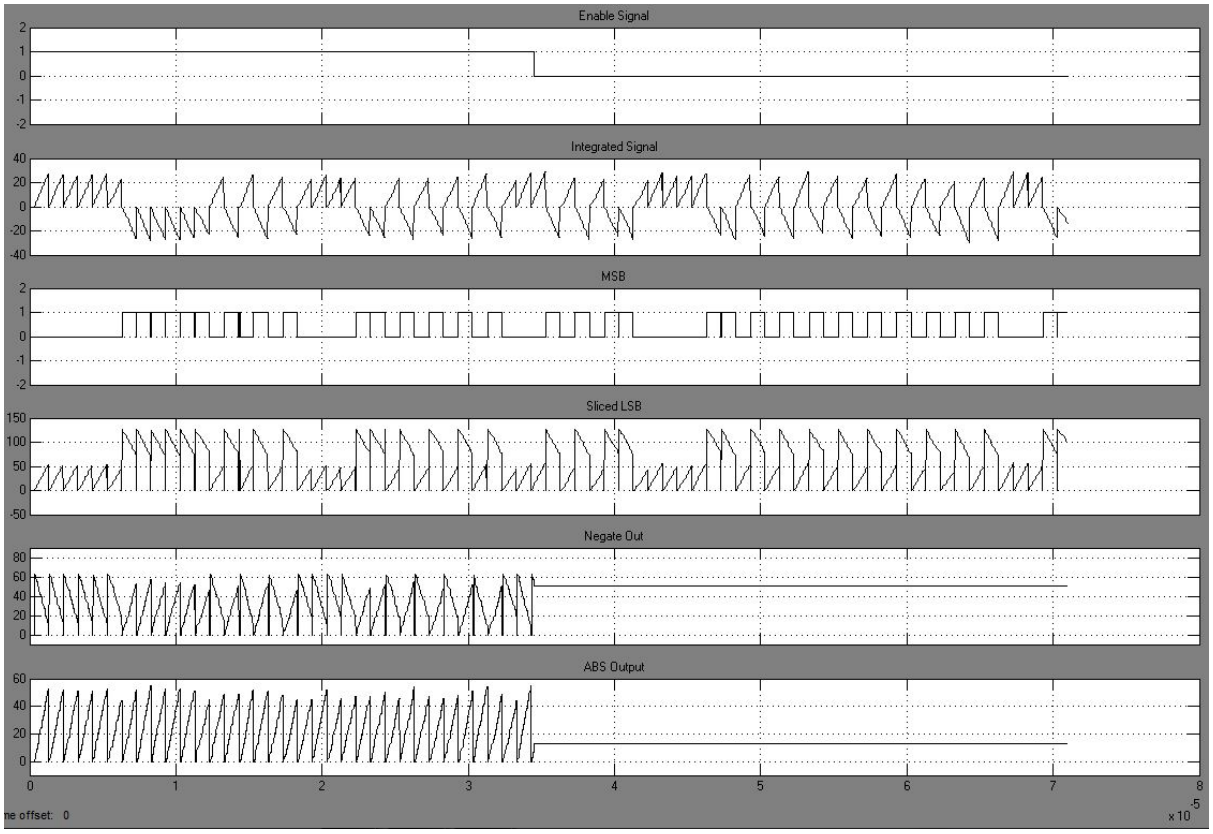

Figure 6.23: (a) Enable signal (b) Integrated Signal (c) MSB (d) Sliced LSB (e) Negate Out (f) ABS out

The output of abs block is then split into three samples as shown in Figure 6.19. They are termed as early sample (without delay), present sample (3 sample delay) and late sample (6 sample delay). They are applied to the inputs of two relational operators. These operators identify the peak among three samples. Once the peak is identified, its value is sampled and is compared with the threshold value. If it is greater than that, peak detect output goes high *"1"*, else it remains low *"0"*. The complete algorithm is presented as follow:

if ((|E| < |P|) AND (|L| < |P|) AND (|P| ≥ Th))

Peak detect=1

Else

Peak detect=0

The internal signals of peak detector and its output are shown in Figure 6.24.

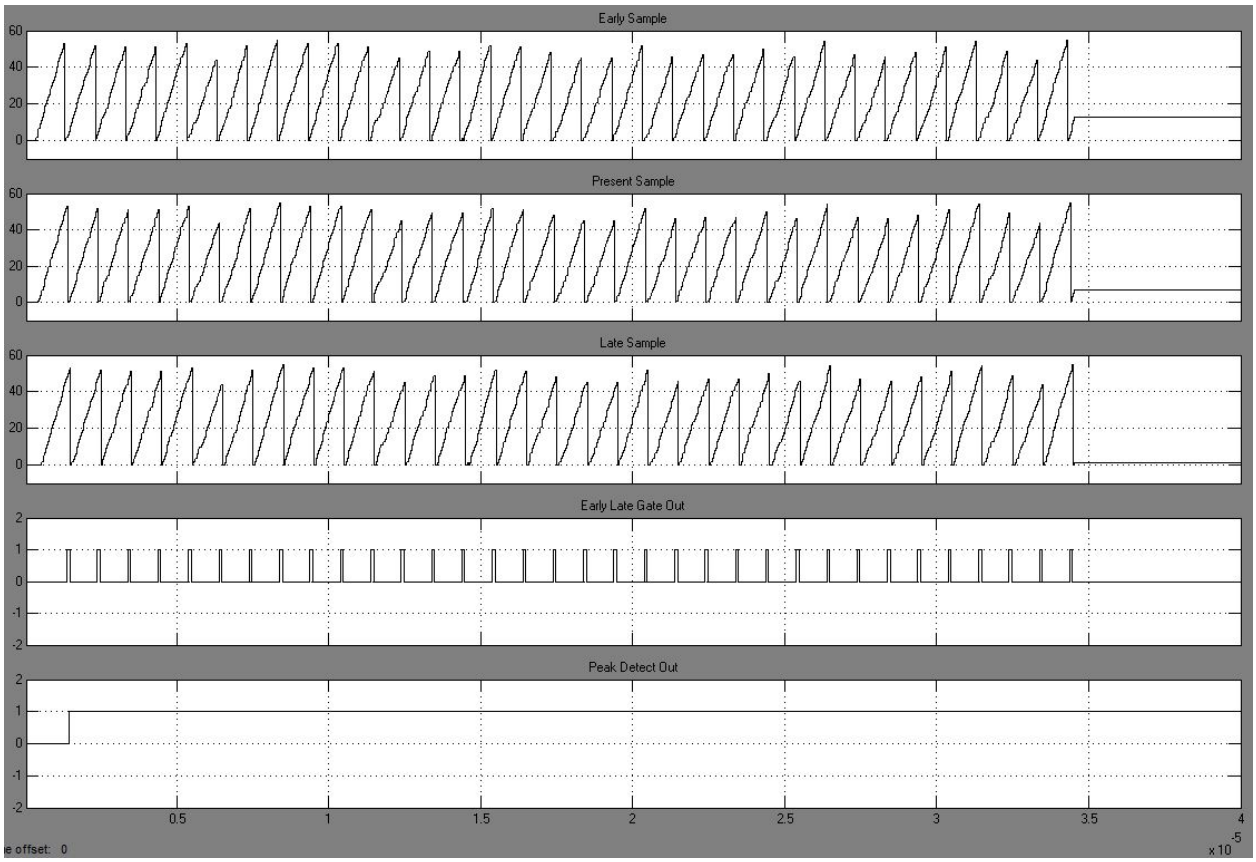

Figure 6.24: (a) Early Sample (b) Present sample (c) Late sample (e) Early late gate sampler out (f) Peak detect out

6.2.3.2 Data Sampler

The data sampler block has two inputs: one is the demodulated signal from the Costas loop and other is the peak detect signal from peak detect module. It consists of inverter, counter, registers and constant blocks as shown in Figure 6.25. The peak detect serve as an enable signal for the counter. Once the peak detect signal is asserted, counter starts counting and when it reaches the count 31, relational operator compares it with the no. of sample value and produce a *"1"* at the output. This *"1"* enables the register which captures the value and sends it to relational operator which compares it with *"0"*. If the value is greater than *"0"*, the output turned to *"1"* otherwise it remains *"0"*.

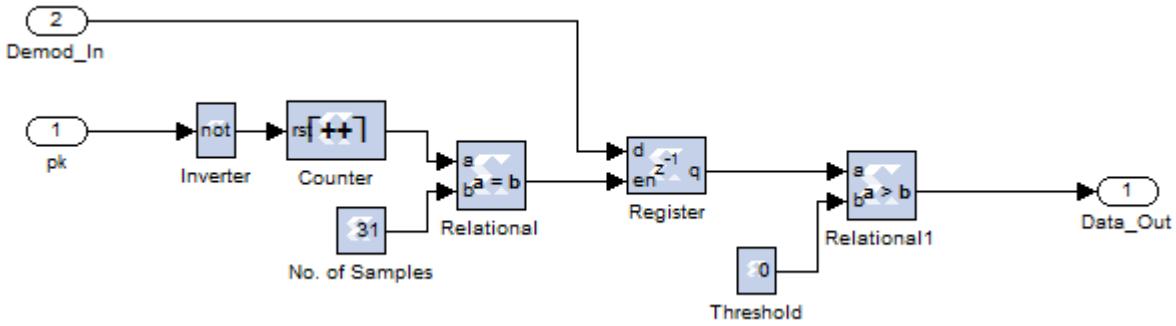

Figure 6.25: Data Sampler

The internal signals and there logic levels are shown in Figure 6.26.

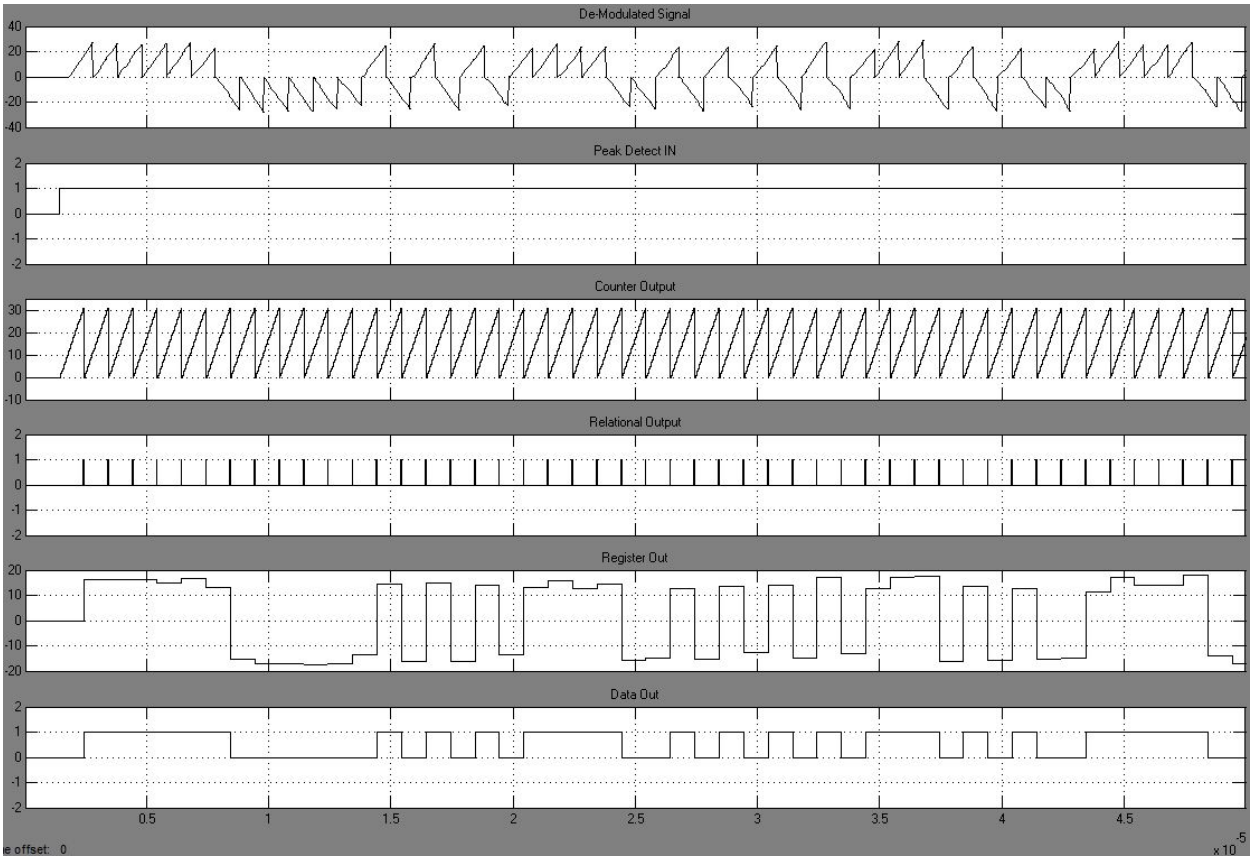

Figure 6.26: (a) Demodulated Signal (b) Peak detect in (c) Counter Out (d) Relational Output (d) Register Out

(e) Data Out

6.2.3.3 Preamble Match

This module has three input ports: In (which takes the peak detect signal from peak detector), en (which takes the enable input) and Sampled_In (which takes the sampled data bits from data sampler) as shown in Figure 6.27.

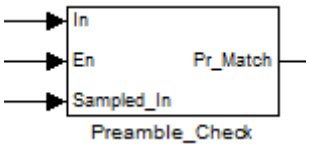

Figure 6.27: Preamble Check module

This module contains counters, registers, ROM, relational and logical operators as shown in Figure 6.28.

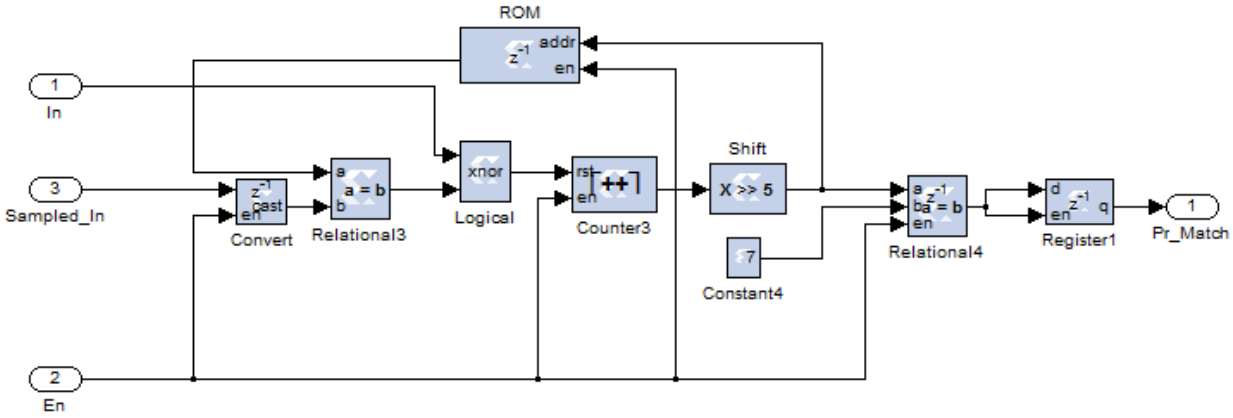

Figure 6.28: Preamble check internal logic

The enable signal is connected to the enable input of all the blocks. The sampled data bit *"Sampled_In"* is sent to relational operator *"Relational3"*. Since the relational operator does not accept Boolean data type, cast block is used to convert Boolean into unsigned data type. The relational block compares the sampled data bits with the pre-stored preamble bits in ROM. The preamble size is of 8 bits and memory for ROM is selected to be Block RAM as shown in Figure 6.29(a). The output of ROM is set to be unsigned type of 1 bit as shown in Figure 6.29(b).

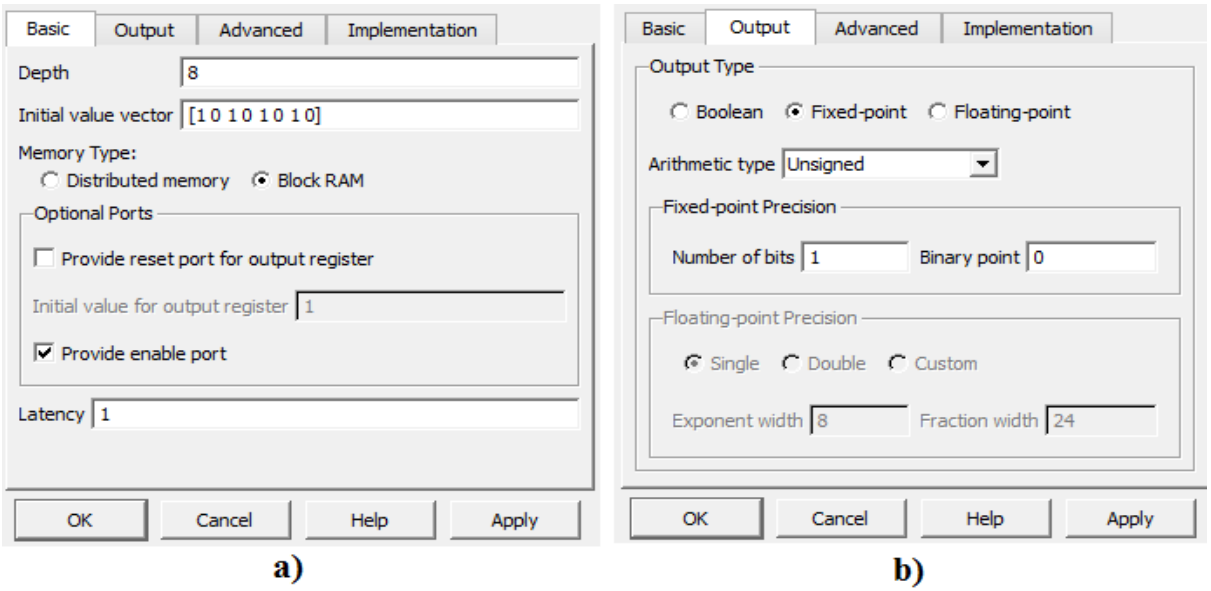

Figure 6.29: ROM configuration (a) Basic (b) Output

If the recovered data bit is equal to the output of ROM, *"Relational3"* output goes high *"1"* otherwise remains *"0"*. The output of *"Relational3"* is connected to the reset pin of counter which function's as an enable signal to it.

Initially counter needs to be hold to its initial position, it will count only when incoming bit is matched with the stored bit at the output of ROM. For this function there were two options, whether to use xor operation or xnor on *"Relational3"* output. Truth table of both the logical functions is given in Table 6.1.

Table 6.1: Truth Table

XOR			**XNOR**		
A	B	C	A	B	C
0	0	0	0	0	1
0	1	1	0	1	0
1	0	1	1	0	0
1	1	0	1	1	1

After the analysis, xnor operation is selected instead of xor because even if there is no peak detected and no data bit is matched, output of xor will give a *"0"* at counter reset port. This would start the counter which is not desired. So xnor operation is performed on the output of *"Relational3"* and *"In"* signal i.e. the peak detect signal. When both the inputs of xnor are same, output is *"1"* otherwise it is *"0"*. The output of xnor block is connected to the counter. The function of this counter is to provide address bits to the ROM for changing its preamble bit. Since preamble size is of 8-bits, 3 bit counter should have been used. But in order to avoid sampling time errors, 8 bit counter is used with max value of 224 as shown in figure 30. The output of counter is divided by 32 by using shift right operation. The divided output is connected to the address port of ROM and input of relational operator *"Relational4"*. It compares the divide output with a constant number *"7"*.

When the count reaches to 7 i.e. the complete preamble is matched with the recovered bit sequence, Relational operator will output 1 at *"Pr_match"* port or otherwise it will maintain *"0"*. The *"Pr_match"* is used as the enable signal for the register *"Register2"* as shown in Figure 6.30. When the "Pr_match" goes high "1", it enables the "Register2" which routes the recovered data bits of data sampler to the output port *"Data"*.

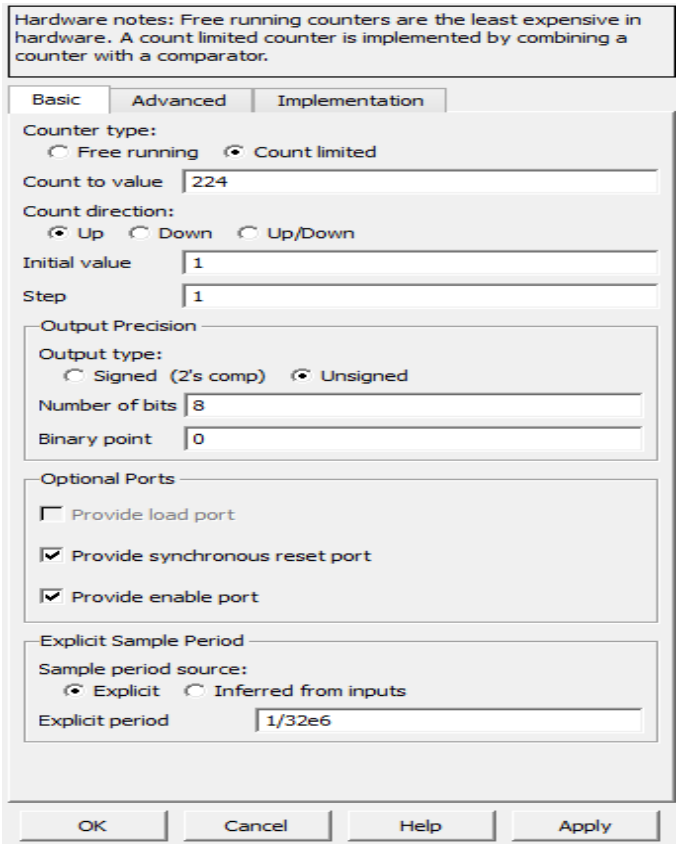

Figure 6.30: Counter configuration

The internal signals at different stages discussed and block output is shown in Figure 6.31.

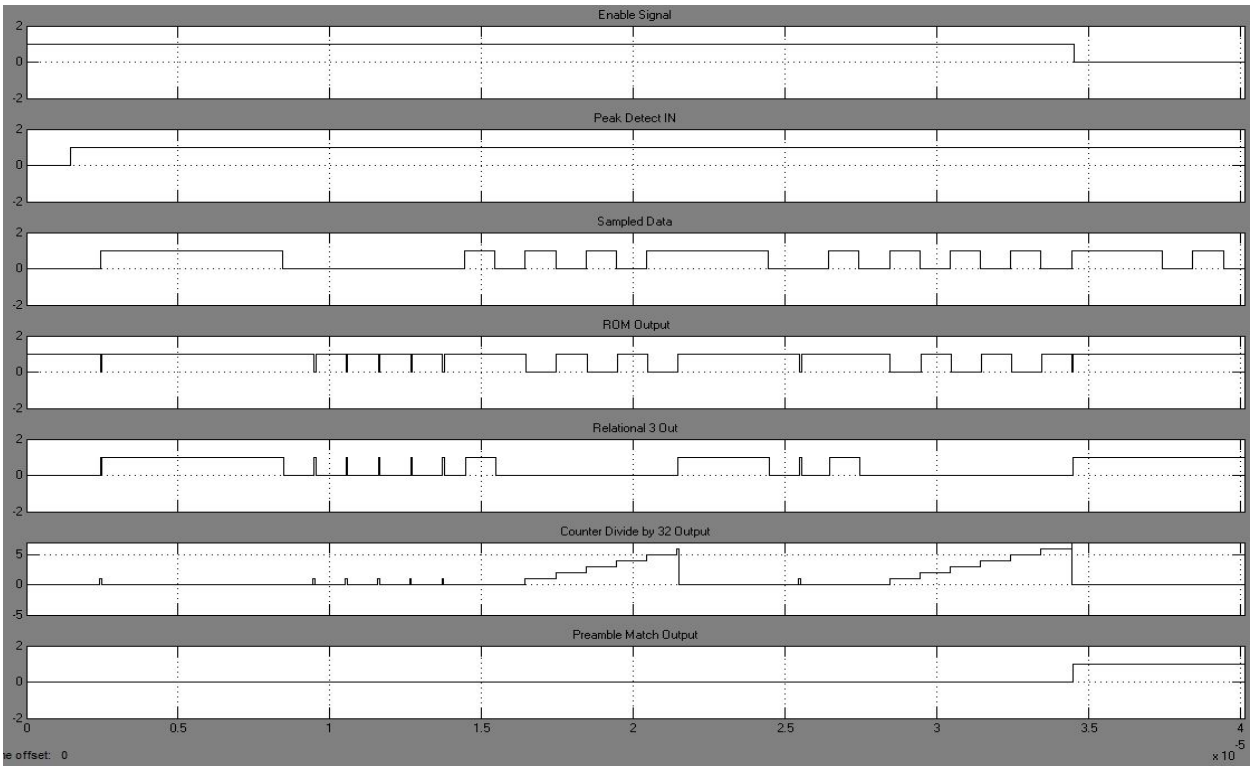

Figure 6.31: (a) Enable Signal (b) Peak Detect IN (c) Sampled Data (d) ROM Output (e) Relational Out (f) Counter Divide Out (f) Preamble match Out

The internal architecture of the bit synchronizer and data sampler block is shown in Figure 6.32. The different signals within it are shown in Figure 6.33

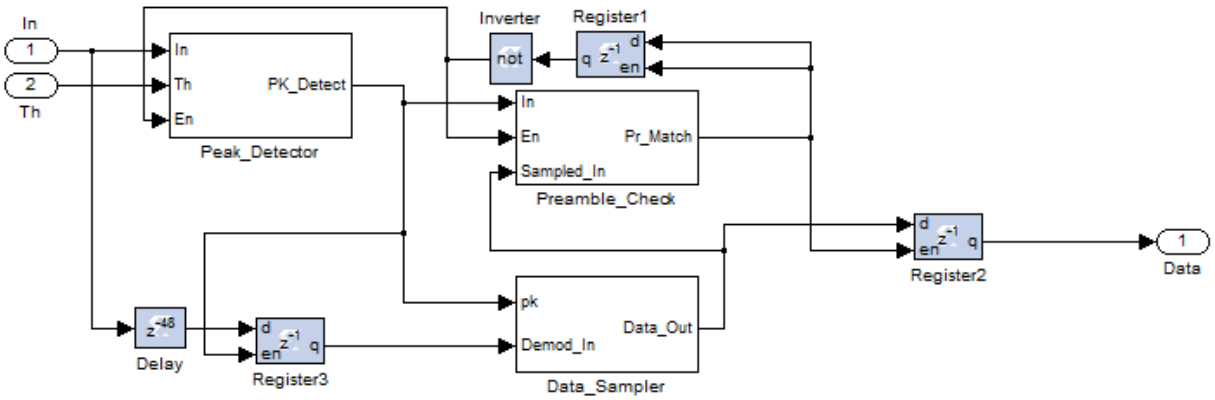

Figure 6.32: Bit Synchronizer and Data Sampler Sub-modules

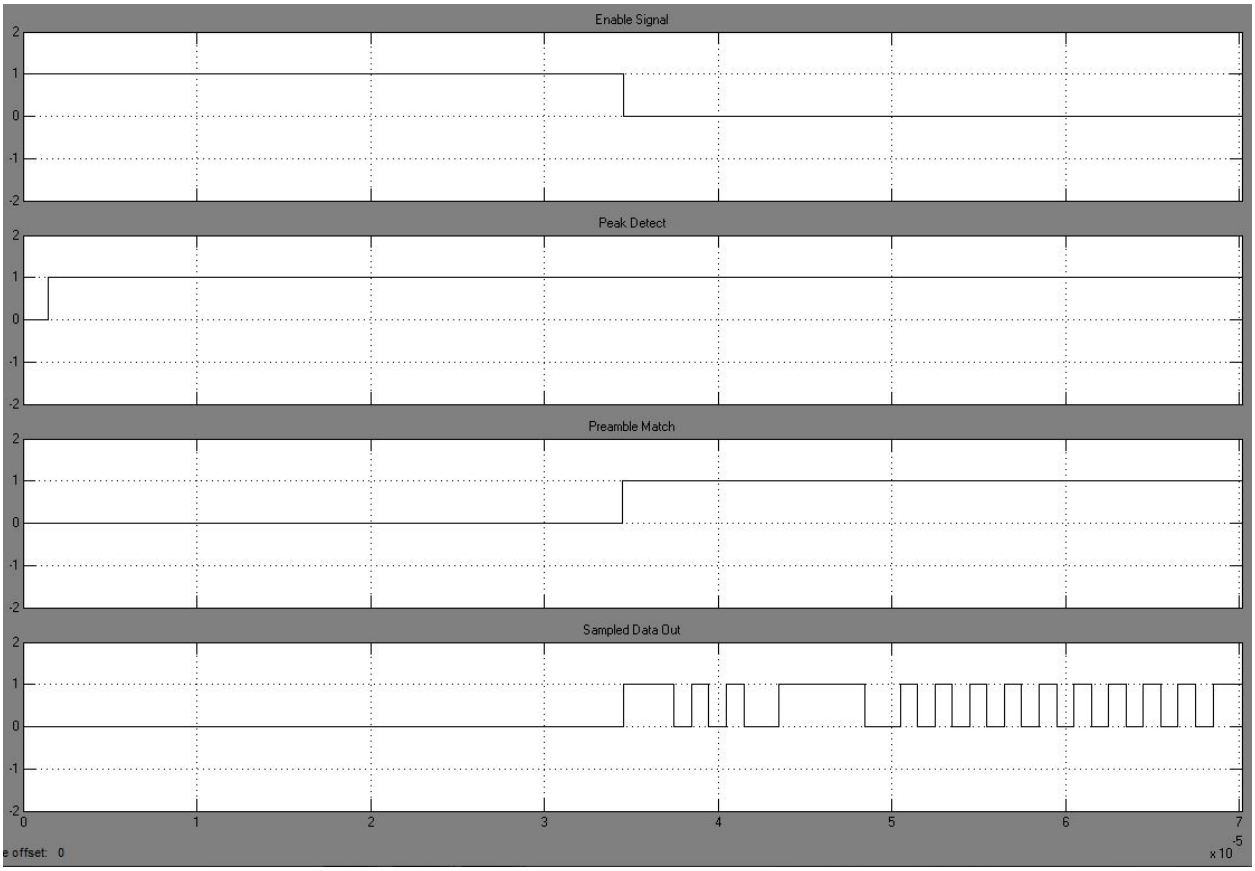

Figure 6.33: Bit Synchronizer and Data sampler internal signals

The complete telecommand receiver implementation of system generator is shown in Figure 6.34. The waveforms of various stages as discussed earlier along with the final recovered data bits are shown in Figure 6.35.

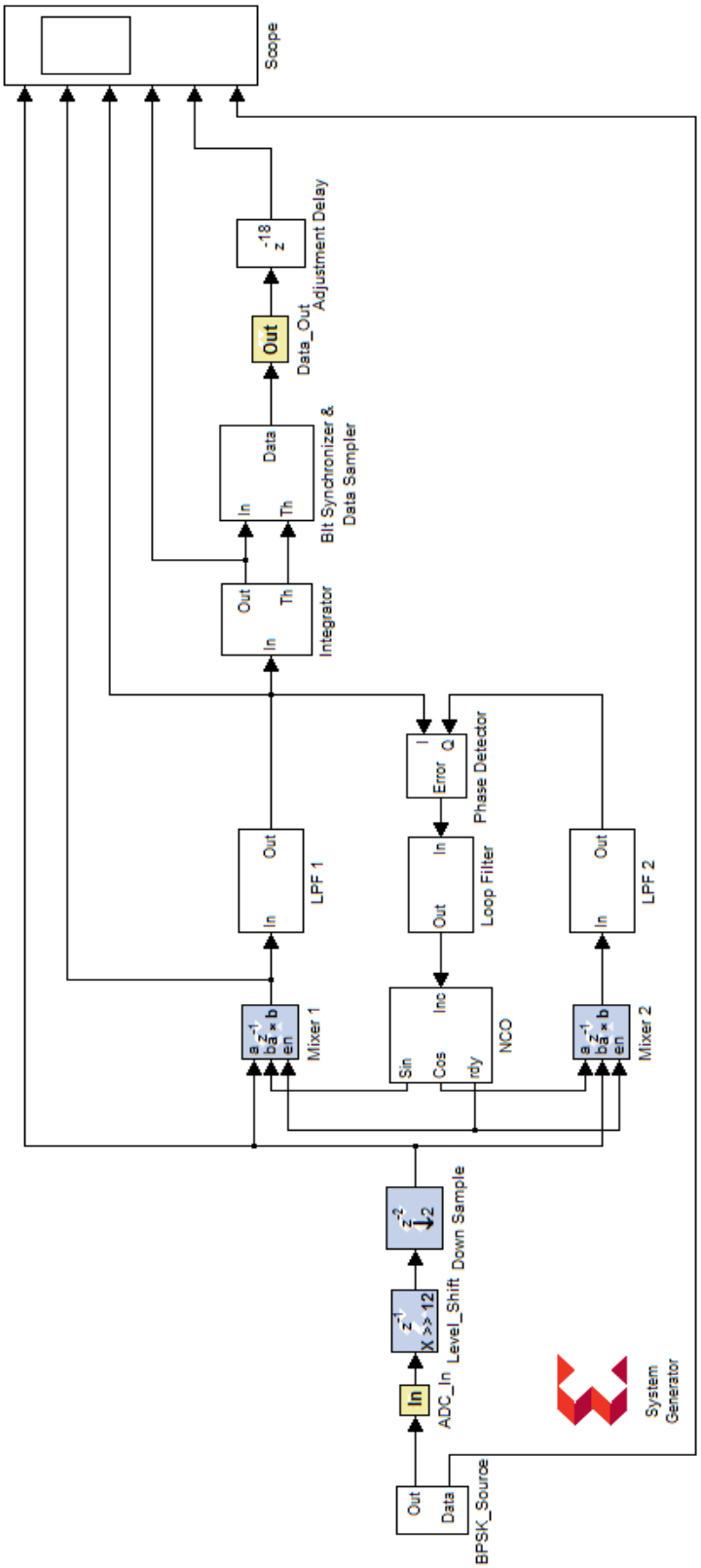

Figure 6.34: Implementation of Telecommand Receiver

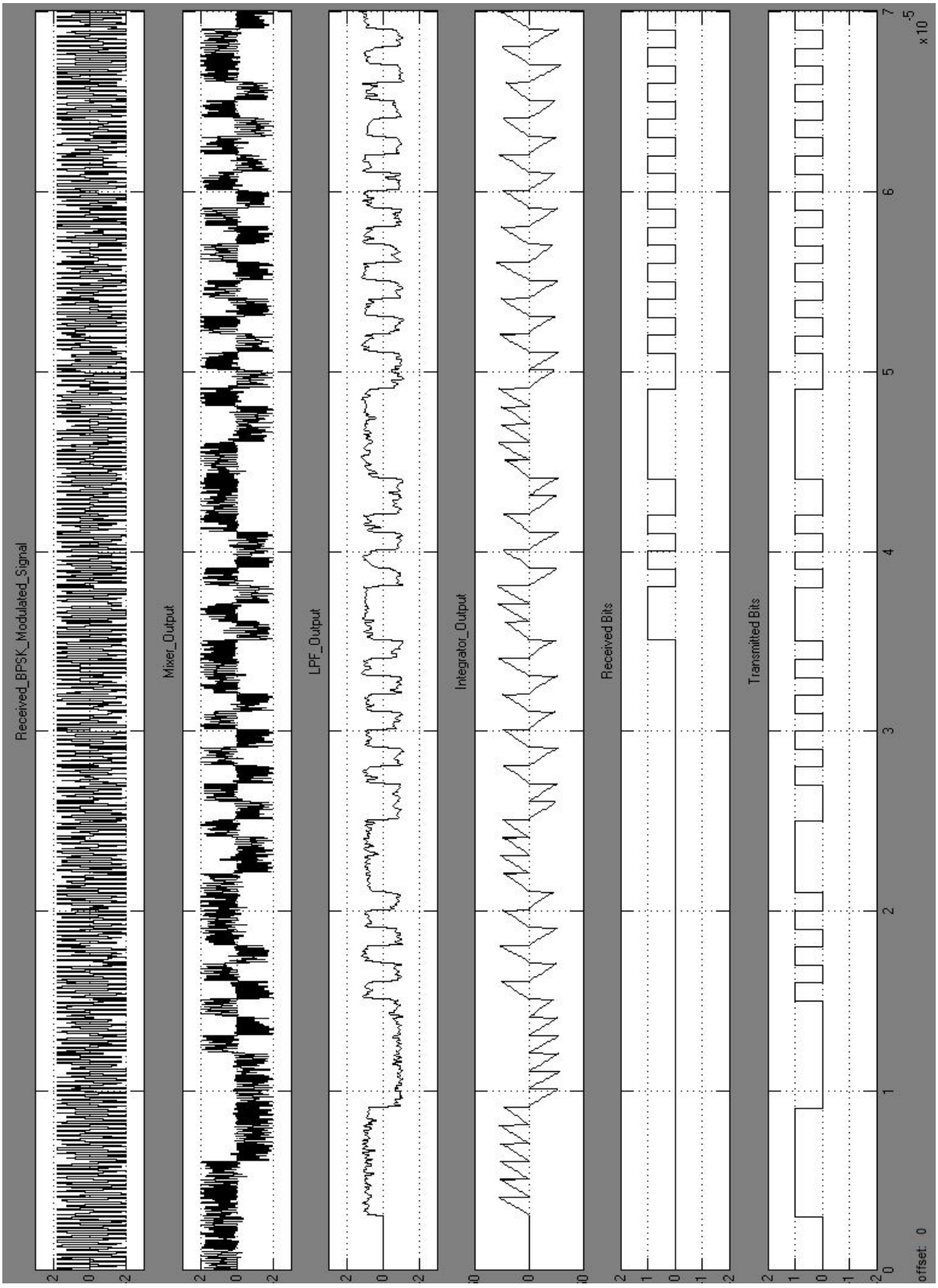

Figure 6.35: Simulation Results of Telecommand Receiver

6.3 SEU Mitigation

As mentioned earlier, RPR is not suited to all types of applications and designs. For applications comprising of arithmetic and non-arithmetic operations, combination of both RPR and TMR is the best approach for mitigating SEUs [8]. The baseband processing module of a telecommand receiver is composed of arithmetic and non- arithmetic operations.

The designed system was analyzed to identify the potential areas where RPR application would cause significant reduction in resource consumption for the complete system. It was observed that the low pass filters and mixers of I and Q channel consists bulk of arithmetic operations. In fact they constitute more than half of the total design resources. This makes them ideal contenders for RPR. Phase detector comprises of multiplication operation, which is better suited for RPR then TMR [29]. Loop filter is composed of binary shift registers which makes RPR ineffective. The decision block contains no arithmetic operation so it cannot be protected using RPR. Experimentally, it was determined that due to the high cost of RPR decision blocks, it is more efficient to apply TMR to NCO, integrator and to the bit synchronization module. The diagram of a telecommand receiver module is shown in Figure 6.36 with annotations indicating the type of mitigation technique applied to each system block.

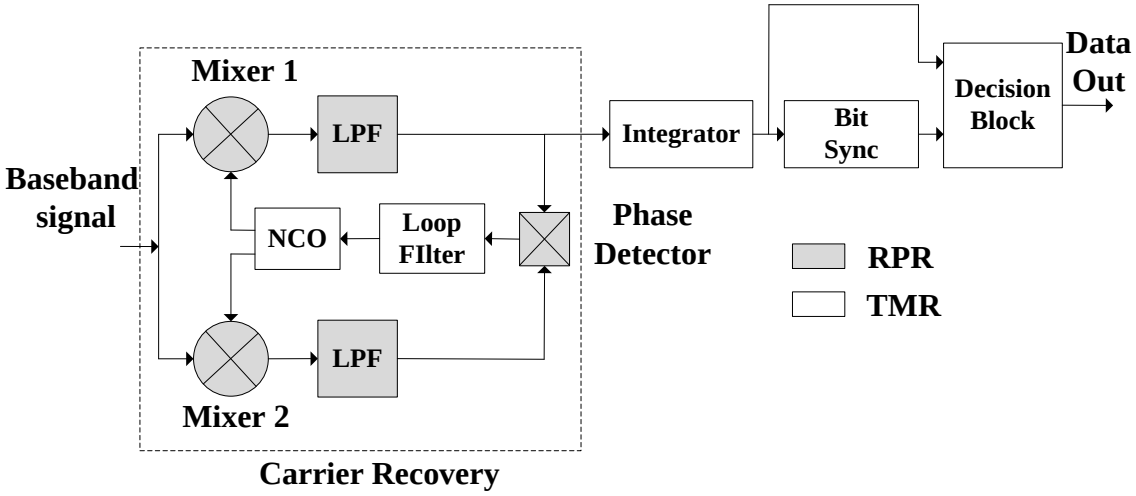

Figure 6.36: Telecommand Receiver annotated for RPR+TMR mitigation

The suitable value of RPR factor (k=7) is determined which reduces the size of RPR module while ensuring good SEU mitigation. The Threshold value (Th) is set to be the maximum difference between FP and RP modules. The RPR mitigated low pass filter with the internal implementation RPR algorithm is shown in Figure 6.37. The difference in the word length

between an unmitigated and mitigated low pass filter is shown in Figure 6.38. It can be observed from the figure, the precision of the LPF internal word length have been reduced. The TMR protected numerically controlled oscillator is shown in Figure 6.39.

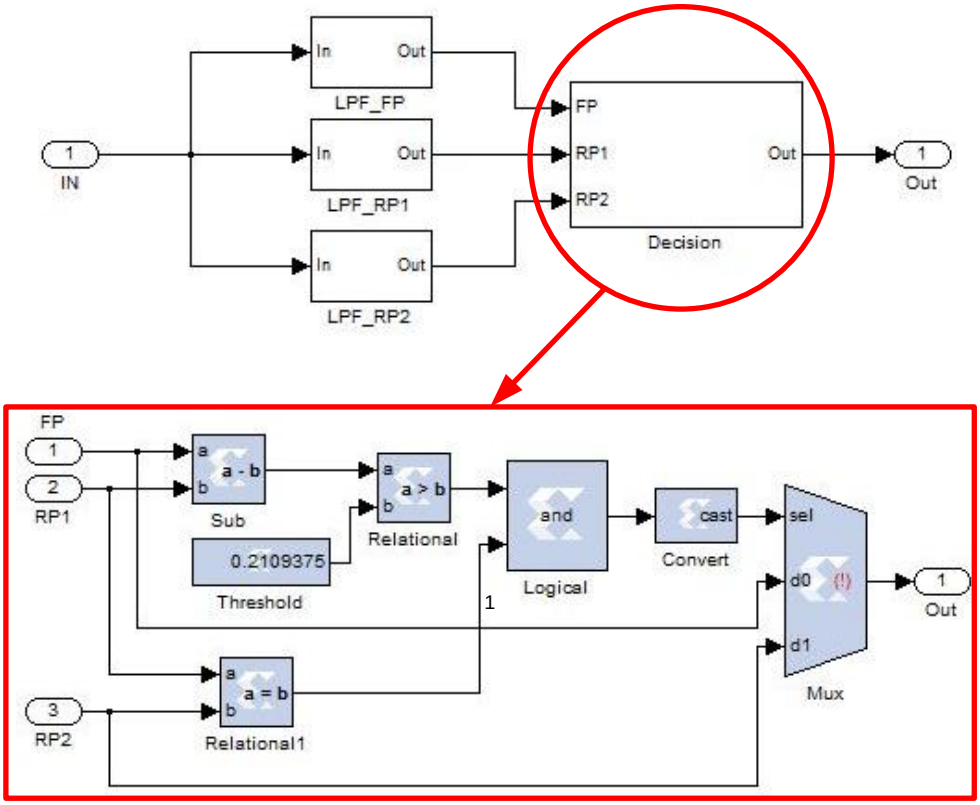

Figure 6.37: RPR Mitigated Low pass filter

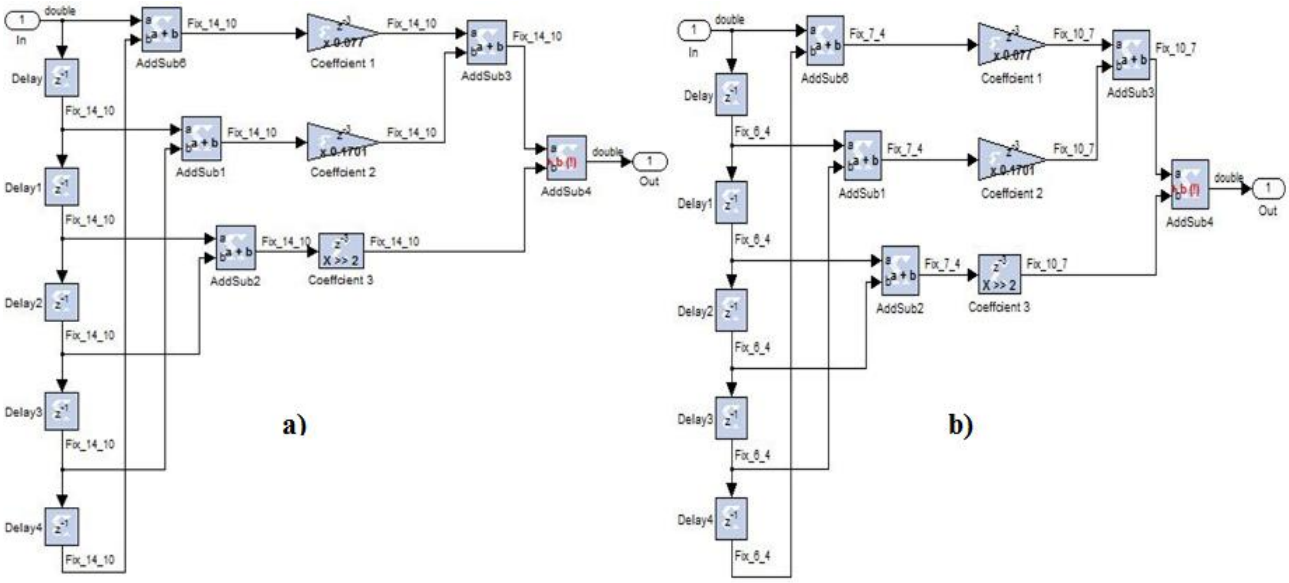

Figure 6.38: (a) Unmitigated LPF (b) Mitigated LPF

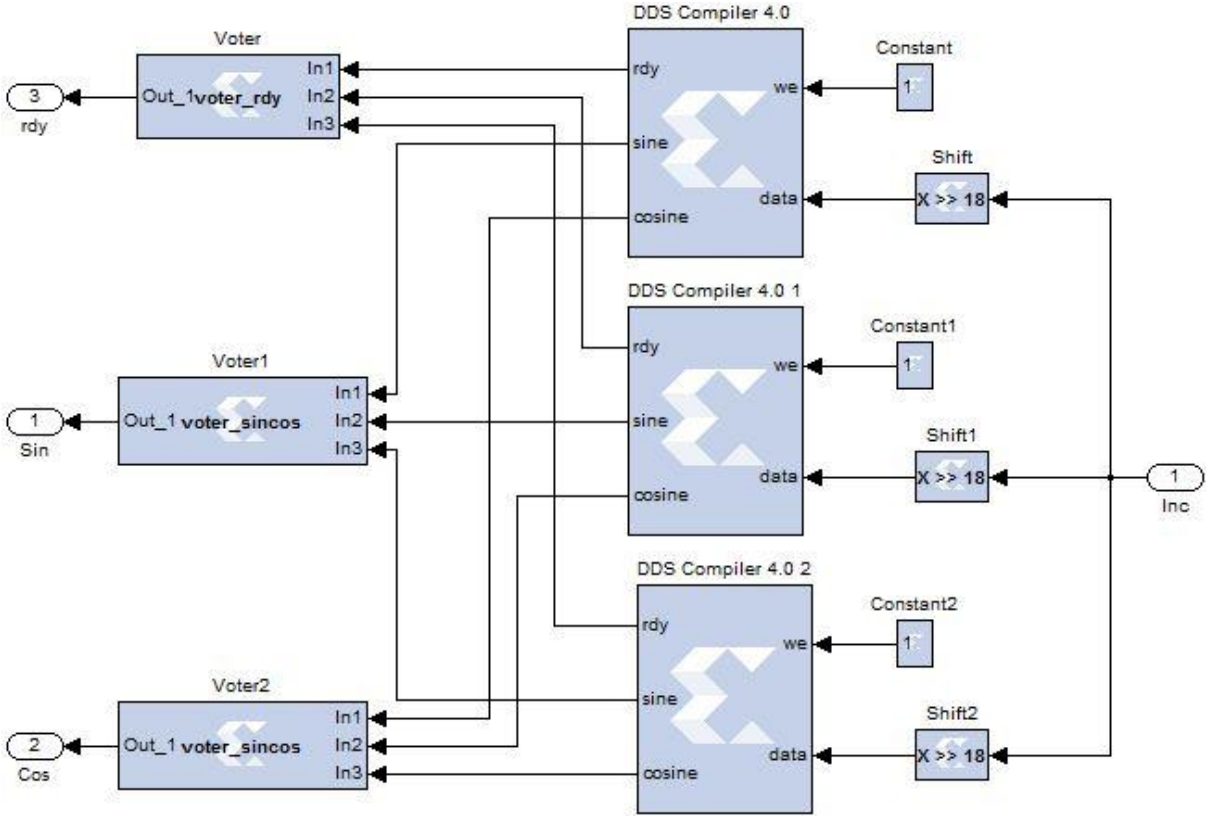

Figure 6.39: TMR Mitigated NCO

CHAPTER 7 RESULTS AND COMPARISON

This chapter presents the implementation results of telecommand receiver and impact of SEU mitigation on receivers BER performance.

7.1 Comparison

The proposed design of the telecommand receiver module is implemented on Xilinx System generator. The module was hardware co-simulated using Spartan 3E XCS3E500E-4FG320 FPGA as shown in Figure 7.1. Hardware Co-simulation incorporates FPGA hardware into the simulation and automates the data exchange process between hardware and software. The data is processed in FPGA and the results are displayed in System generator.

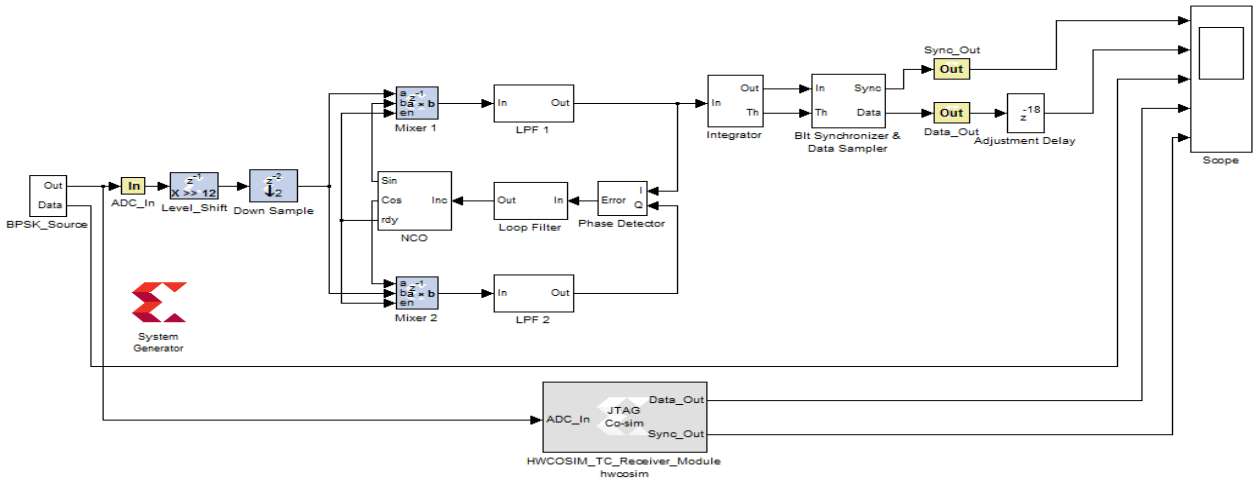

Figure 7.1: Hardware Co-simulated model of Telecommand Receiver

Results from graphs shown in Figure 7.2 confirms that, the Sync_lock signal (a) and received data bits (b) from Xilinx System generator simulation are identical to the Sync_lock signal (d) and data received bits (e) results from hardware co-simulation. The designed system was tested with frequency shifts up to ± 200 KHz. In first case, the Incoming signal's frequency was first shifted to 4.2 MHz, while the NCO was running on 4.0 MHz. In second case, the incoming signal was shifted to 3.8 MHz. In both the cases NCO successfully tracked the incoming signal frequency in less than 10 us as shown in Figure 7.3 and Figure 7.4.

BER analysis of the receiver module is performed using "bertool" provided in MATLAB. The BER performance of the overall designed system is calculated using Monte Carlo simulation. Figure 7.5 presents a comparison between the BER of the proposed system and the ideal BPSK receiver in AWGN channel. It can be seen that the proposed systems BER is

almost identical to the BER of the ideal receiver. The slight degradation in BER graph of the designed module is due to the implementation losses.

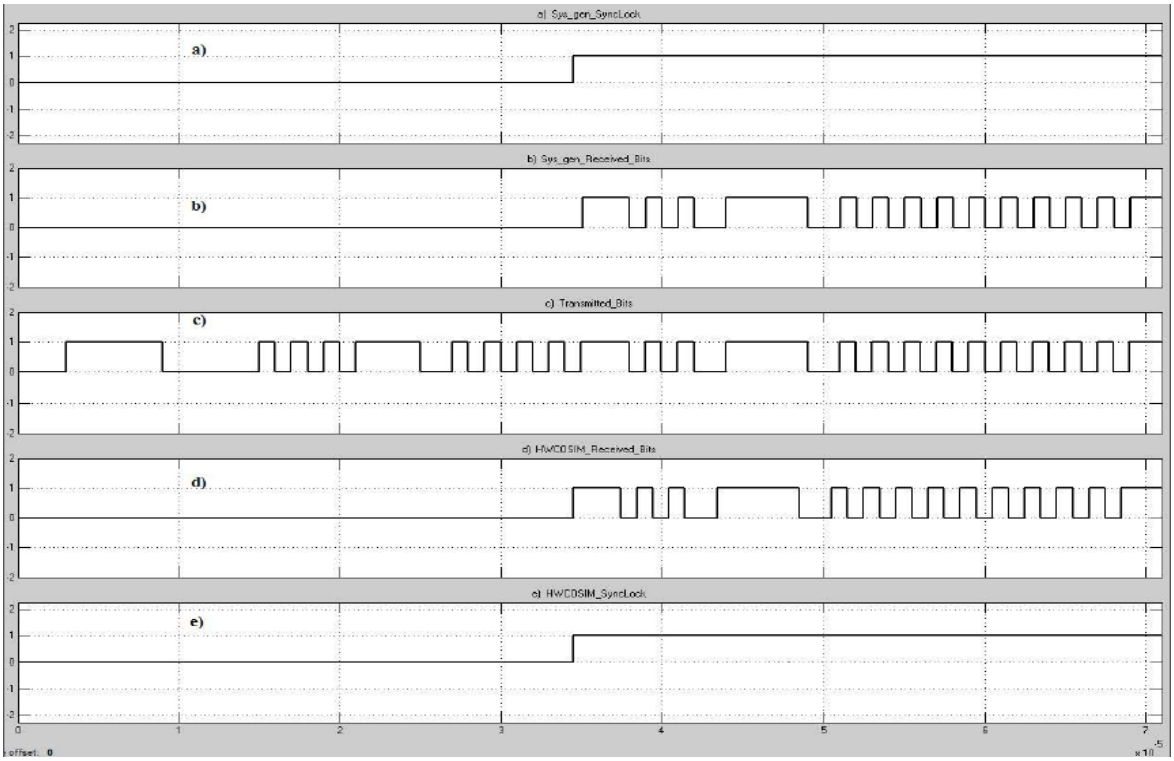

Figure 7.2: Output of Scope a) Sys_gen_SynLock b) Sys_gen_Received_Bit c) Transmitted_Bits d) HWCOSIM_Received_Bits e) HWCOSIM_SynLock

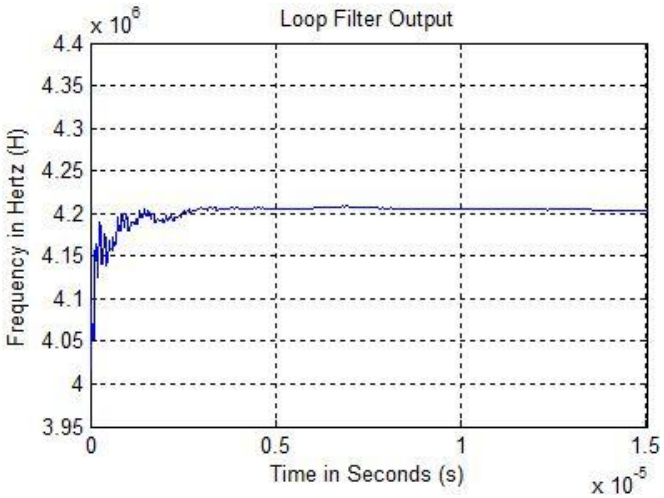

Figure 7.3: Loop Filter Output with incoming signal at 4.2 MHz

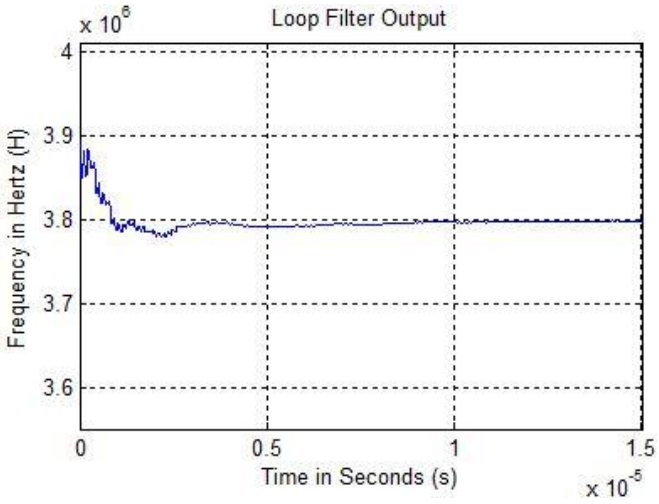

Figure 7.4: Loop Filter Output with incoming signal at 3.8 MHz

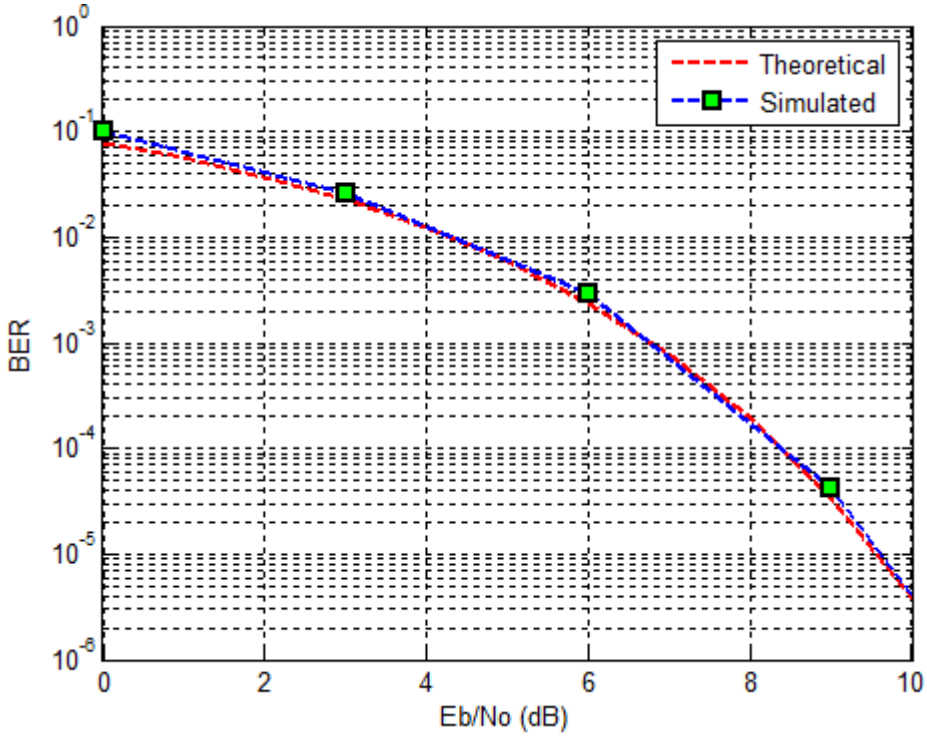

Figure 7.5: BER of Telecommand Receiver

The proposed module's logic utilization on FPGA and its comparison with Maya, J.A., et al [4] is presented in Table 7.1. It can be seen that the designed system consumes 50% less multipliers, 1% less slices and 5% less 4- input LUTS as compared to [4]. The timing recovery unit of the proposed system consumes almost 60% less slices as compared to [4] and uses no multiplier.

Table 7.1: Resource Comparison of Proposed Receiver Module and Maya, J et.al [4] on Spartan 3e

Logic Utilization	Carrier Recovery		Timing Recovery		Total Available	Utilization	
	Used		Used			Percentage %	
	This Work	[4]	**This Work**	[4]		**This Work**	[4]
Slice	**553**	607	**155**	387	4,656	**15**	16
Slice Flip flops	**759**	732	**158**	651	9,312	**9**	13
4-input LUTS	**677**	973	**136**	503	9,312	**8**	13
RAM16s	**1**	2	**1**	0	20	**10**	10
MULT18x18s	**3**	2	**0**	4	20	**15**	30
BUFGMUXs	**1**	1	**1**	1	24	**4**	8

7.2 SEU Mitigation

The SEU effect is emulated by inverting a bit in the design [30]. For this purpose, loop filter is selected because it plays a critical role of keeping the demodulator and receiver in the desired working area and an upset in it would have a major impact on receiver's performance. Class 1 SEU, is introduced by inverting the LSB of the loop filter. Class 2 SEU is simulated by flipping the middle order bit of the loop filter. Higher order bit is flipped for class 3 SEU. The MSB of the loop filter is inverted for class 4 SEU. The effect of SEU on BER performance with respect to different classes is shown in Figure 7.6.

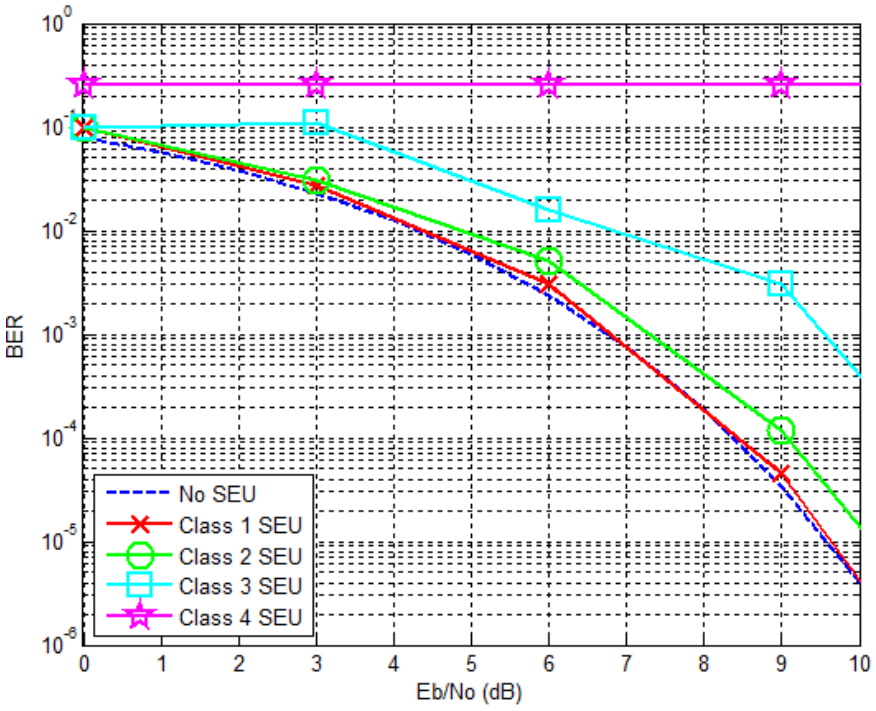

Figure 7.6: SEU effect on BER performance

It can be observed from Figure 7.6 that all classes of SEUs have different impact on BER performance. Class 1 and 2 SEUs cause minor degradation in BER and are not critical. These errors can be corrected using standard techniques. Class 3 and 4 SEUs have a devastating impact on BER; redundancy must be used in order to enhance the BER.

TMR and RPR application increases the overall size of the designed system by introducing its replica's. Therefore, a different FPGA platform was required that can meet the resource requirements. We decided to implement TMR and the combination of RPR and TMR using Virtex 4 XC4VSX55-10FF1148 FPGA as shown in Figure 7.7. Both methods mitigate SEUs successfully as shown in Figure 7.8. Their resource comparison is presented in Table 7.2. The results show that the hybrid approach is more efficient in terms of resources as compared to TMR. The combination of RPR and TMR consumes 26% less slices, 42% less slice flip-flops and almost 18% less 4-input LUTS as compared to TMR.

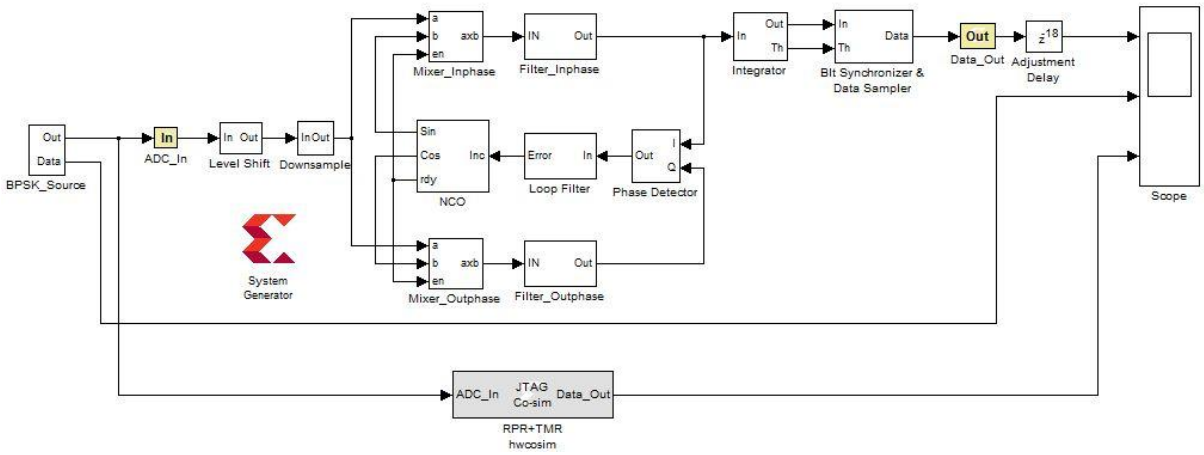

Figure 7.7: Hardware Co-simulated model of SEU Mitigation

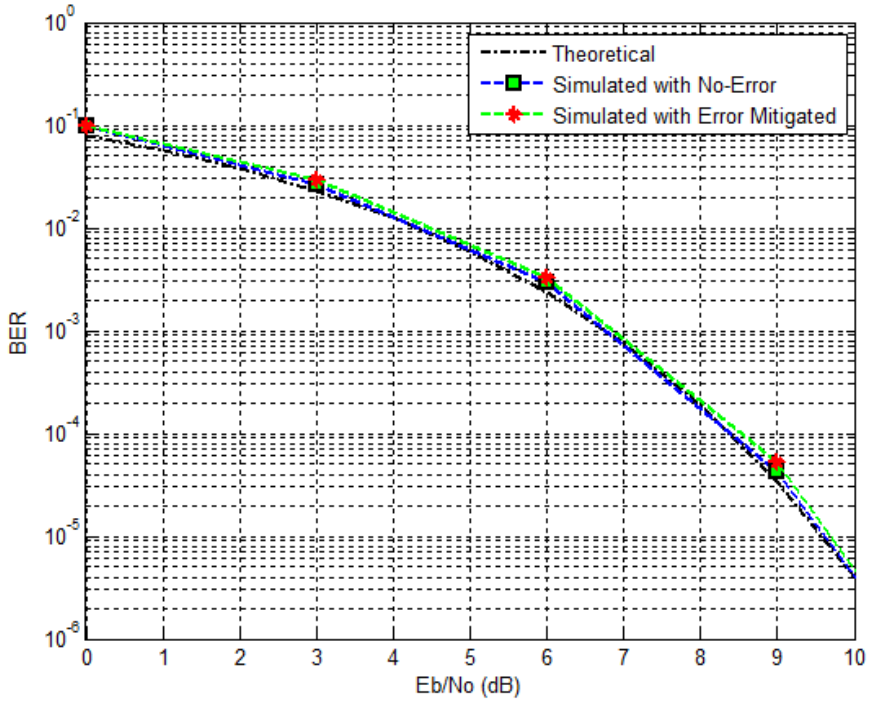

Figure 7.8: SEU mitigated using RPR+TMR

Table 7.2: Resource Comparison of TMR and RPR+TMR on Virtex 4

Logic Utilization	TMR	RPR + TMR	Available	Resource Reduction (%)
	Used	Used		
Slice	2,572	1,901	24,576	26.08
Slice Flip Flops	3,149	1,817	49,152	42.29
4-input LUTS	2,721	2,240	49,152	17.68

CHAPTER 8 CONCLUSION AND FUTURE WORK

8.1 Conclusion

In this research, we presented the resource efficient implementation of a BPSK satellite telecommand receiver using FPGA. The adopted scheme uses a resource efficient implementation of Costas loop for carrier recovery and early late gate sampling algorithm for timing recovery. The optimized receiver module has BER performance identical to theoretical, with minor degradation due to implementation losses. Our designed system consumes 50% less multipliers, 1% less slices and 5% less 4- input LUTS as compared to [4]. The timing recovery unit of the proposed system consumes almost 60% less slices as compared to [4] and uses no multiplier.

A new technique for software defined radiation tolerance of baseband module for a LEO satellite telecommand receiver is also implemented. The combination of RPR and TMR is used in the receiver module for SEU mitigation. This hybrid approach has shown to be very effective and consumes far less resource than a customary TMR protected receiver. It has been concluded that by focusing on targeted implementation of RPR in systems involving arithmetic operations, a lot of resources can be saved as compared to complete TMR system.

8.2 Future Work

As a Scope for future work, this design can be implemented using RF Front end connected with the FPGA hardware to perform real - time measurements. The RPR redundancy factor (k) can be changed to evaluate its impact on systems BER performance.

8.3 Publication

Salman Sadruddin and Arshad Aziz, “Reduced Precision Redundancy for Satellite Telecommand Receiver Module on FPGA”, Hindawi Publishing Corporation, Chinese Journal of Engineering, Article ID 453872, Volume 2013.

REFERENCES

[1] Caffrey, M., “A space-based reconfigurable radio,” Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA ’02), LasVegas, NV, pp. 49-53, June 2002.

[2] P.E. Dodd and L, W. Massengill, “Basic mechanisms and modeling of single-event upset in digital microelectronics,” IEEE Transactions on Nuclear Science, vol. 50, no. 3, pp. 583-602, June 2003.

[3] Z. Zhao, Y. Shen, and Y. Bai, “Design and Implementation of the BPSK Modem Based on Software Defined Radio,” in 2011 First International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC 2011), pp. 780–784, , Los Alamitos, CA, USA, 21-23 Oct. 2011.

[4] Maya, J.A., et al., “A high data rate BPSK receiver implementation in FPGA for high dynamics applications,” 2011 VII Southern Conference on Programmable Logic (SPL), Cordoba, Spain, pp. 233-238, April 2011.

[5] C. Carmichael, “Triple module redundancy design techniques for Virtex FPGAs,” Tech. Rep. XAPP197 (v1.0), Xilinx Corporation, 2001.

[6] Shim, B. and Shanbhag, N., “Reduced precision redundancy for low-power digital filtering,” Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers, vol. 1, Pacific Grove, CA, pp. 148-152, November 2001.

[7] Snodgrass, J. “Low-power fault tolerance for spacecraft FPGA-based numerical computing,” Ph.D. dissertation, Dept. of Electrical and Computer Engineering, Naval Postgraduate School, Monterey, CA, September 2006.

[8] Pratt B. et al., “Reduced-Precision Redundancy for Reliable FPGA Communications Systems in High-Radiation Environments,” IEEE Transaction on Aerospace and Electronic Systems vol. 49, no. 1, pp. 369-380, January 2013.

[9] J. L. Tresvig, “Design of a Prototype Communication System for the CubeSTAR Nano-satellite,” University of Oslo, Oslo, Norway.

[10] J. Bingham, B. Cohrssen, and C. H. Powell, “The theory and practice of modem design,” Recherche, vol. 67, pp. 02, 1988.

[11] W. Traussnig, “Design of a Communication and Navigation Subsystem for a CubeSat Mission,” Karl Franzens University of Graz, Graz, Austria.

[12] J. Proakis, Digital Communications, 4th ed. McGraw-Hill Science/Engineering/Math, 2000.

[13] S. O. Popescu, A. S. Gontean, and G. Budura, “Simulation and implementation of a BPSK modulator on FPGA,” in Applied Computational Intelligence and Informatics (SACI), 2011 6th IEEE International Symposium, pp. 459 –463, 2011.

[14] Shamla, B. and Gayathri Devi, K.G., “Design and Implementation of Costas loop for BPSK Demodulator”, Annual IEEE India Conference (INDICON) , Kochi, India, pp 785-789, December 2012.

[15] J. Costas, “Synchronous Communications,” Proceedings of the IEEE, vol. 44, p. 1713-1718, 1956.

[16] J. Proakis, Digital Communications, 4th ed. McGraw-Hill, pp. 356–357 2001.

[17] B. P. Lathi, Modern Digital and Analog Communication Systems, 3rd ed. Oxford University Press, pp. 187–188, 1998.

[18] Jeff Feigin, “Practical Costas loop design”, RF design, pp.2036, January 2002.

[19] Mitchell G. and Guichon T, “Digital Costas loop design for coherent microsatellite transponders”, IEEE Aerospace Conference Proceedings, vol.3, pp.1197-1209, 2002.

[20] Martin Kumm, Harald Klingbeil, and Peter Zipf, “An FPGA-Based Linear All-Digital Phase-Locked Loop”, IEEE Transactions on Circuits and Systems—I: Regular Papers, Vol. 57, No. 9, September 2010.

[21] “A z-Domain Model and Analysis of Phase-Domain All-Digital Phase-Locked Loops” by Stefan Mendel and Christian Vogel, Proceedings of the IEEE Norchip Conference, Aalborg (Denmark), 2007.

[22] “Design and Implementation of Early-Late Gate Bit Synchronizer for Satellite Communication” by P.N.Ravichandran, Satish Sharma, Sunil Kulkarni and P.Lakshminarsimhan, NCC2009, IIT Guwahati, India, January16-18 2009.

[23] Clive Maxfield, book, "The Design Warrior's Guide to FPGAs"; Published by Elsevier, 2004. ISBN 0750676043, 9780750676045.

[24] Spartan-3 FPGA Family Data Sheet, DS099, December 2009, available at http://www.xilinx.com/support/documentation/data_sheets/ds099.pdf

[25] Virtex4 FPGA Family Data Sheet, DS112, August 2010, available at http://www.xilinx.com/support/documentation/data_sheets/ds112.pdf

[26] Sanket Prakash Joshi, “Integrating FPGA with Multi-core SDR Development Platform to Design Wireless Communication System”, A graduate project submitted in partial fulfillment of the requirements for the degree of Masters of Science in Electrical Engineering., Dept. of Electrical Engineering, California State University, Northridge, May 2012.

[27] Mitchell G. and Guichon T, “Digital Costas loop design for coherent microsatellite transponders”, IEEE Aerospace Conference Proceedings, vol.3, pp.1197-1209, 2002.

[28] Liu Zhi, Jiang Zhou, Li Qing and Zeng Xiaoyang, “Efficient Carrier Recovery for High Order QAM,” International Conference on Consumer Electronics, pp.1-2, January 2007.

[29] Sullivan, M. A., “Reduced precision redundancy applied to arithmetic operations in field programmable gate arrays for satellite control and sensor systems”, Master’s thesis, Dept. of Mechanical and Astronautical Engineering and Dept. of Electrical and Computer Engineering, Naval Postgraduate School, Monterey, CA, December 2008.

[30] Salman Sadruddin and Arshad Aziz, “Reduced Precision Redundancy for Satellite Telecommand Receiver Module on FPGA”, Hindawi Publishing Corporation, Chinese Journal of Engineering, Article ID 453872, Volume 2013.