"DESIGN AND IMPLEMENTATION OF HIGHLY RELIABLE, LOW POWER PHY BASEBAND PROCESSOR BASED ON IEEE 802.15.4"

A Major Project Report

Submitted in Partial Fulfillment of the Requirements for the Degree of

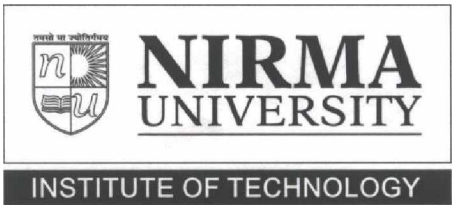
MASTER OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGG.

(VLSI Design)

By
Dhaval Shah
(03MEC018)



Department of Electronics & Communication Engineering
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY OF SCIENCE & TECHNOLOGY,
AHMEDABAD 382 481

MAY 2005

CERTIFICATE

This is to certify that the Major Project Report (Part-I-II) entitled "Design and Implementation of Highly Reliable, Low Power PHY Base Band Processor Based on IEEE 802.15.4" submitted by Mr. Dhaval Shah (Roll No.03MEC18) towards the partial fulfillment of the requirements for Semester III-IV of Master of Technology (Electronics & Communication Engg.) in the field of VLSI Design of Nirma University of Science and Technology is the record of work carried out by him under our supervision and guidance. The work submitted has in our opinion reached a level required for being accepted for examination. The results embodied in this major project work to the best of our knowledge have not been submitted to any other University or Institution for award of any degree or diploma.

Date:

Project Guide:
Mr. Sanjay Shah
CEO, Sancom Technologies

Facilitator at Institute:
Prof. N. P. Gajjar
Electronics & Comm. Engg. Dept.
Institute Of Technology,
Nirma University, Ahmedabad

HOD
Dr. M. D. Desai
EC – Department
NIRMA University, Ahmedabad

Director
Dr. H. V. Trivedi
NIRMA University, Ahmedabad

Signature of Examiners:

Abstract

The new IEEE standard 802.15.4 shows promise to bring ubiquitous networking into our lives, at least technically. Unlike other standard targeting high or moderate data rates applications, IEEE 802.15.4 standard is a global standard design for low data rates, low power consumption and low cost applications. This so called enabling standard will bring many simple, originally standalone devices into network and thus not only open the door to a numerous number of new application but also add values to many other existing application such as lighting control, automatic meter reading, wireless smoke detector, HVAC control, home security, medical sensing and monitoring.

There are a multitude of standards that address mid to high data rates for voice, PC LANs, video, etc. However, up till now there hasn't been a wireless network standard that meets the unique needs of sensors and control devices. Sensors and controls don't need high bandwidth but they do need low latency and very low energy consumption for long battery lives and for large device arrays.

There are a multitude of proprietary wireless systems manufactured today to solve a multitude of problems that also don't require high data rates but do require low cost and very low current drain. These proprietary systems were designed because there were no standards that met their requirements. These legacy systems are creating significant interoperability problems with each other and with newer technologies.

The Zigbee Alliance is not pushing a technology; rather it is providing a standardized base set of solutions for sensor and control systems. Zigbee is poised to become the global control/sensor network standard. The physical layer was designed to accommodate the need for a low cost yet allowing for high levels of integration. The use of direct sequence allows the analog circuitry to be very simple and very tolerant towards inexpensive implementations. The media access control (MAC) layer was designed to allow multiple topologies without complexity. The power management operation doesn't require multiple modes of operation. The MAC allows a reduced functionality device (RFD) that needn't have flash nor large amounts of ROM or RAM. The MAC was designed to handle large numbers of devices without requiring them to be "parked".

The network layer has been designed to allow the network to spatially grow without requiring high power transmitters. The network layer also can handle large amounts of nodes with relatively low latencies.

Zigbee has been designed to provide the following features:

Dual PHY (2.4GHz and 868/915 MHz), Data rates of 250 kbps (@2.4 GHz), 40 kbps (@ 915 MHz), and 20 kbps (@868 MHz), Optimized for low duty-cycle applications (<0.1%), CSMA-CA channel access yields high throughput and low latency for low duty cycle devices like sensors and controls, Low power (battery life multi-month to years), Multiple topologies: star, peer-to-peer, mesh, Addressing space of up to: 65,535 devices (16 bit extended address), optional guaranteed time slot for applications requiring low latency, Fully hand-shaked protocol for transfer reliability, Range: 50m typical (5-500m based on environment).

This project work involves the design and implementation of PHY base band processor for the Zigbee devices. The project work has been carried out in several stages. It started with the design of the architecture. Then a golden reference design has been done in the MATLAB and SIMULINK. This golden design was simulated in the MATLAB. The results of this simulation such as BER, PER etc. were compared with that given in the standard. Then the RTL design was done in VHDL. This design was then simulated in the Modelsim, synthesized using the Xilinx synthesis Tool (XST), and finally implemented using placing and routing Tool. Implemented design was then downloaded on the FPGA board. Some quantitative results from MATLAB simulation and logical synthesis have been discussed.

CONTENTS

ACKNOWLEDGEMENT

Firstly I would like to express my profound gratitude to my project guide, Mr. Sanjay Shah, CEO, Sancom Technologies, for his outstanding support and guidance during my stay at Sancom technologies, Gandhinagar. I would also like to thank Mr. Alpesh Shah, Chief Operating Officer, Sancom Technologies, Gandhinagar without whose support it would be impossible to complete this project successfully.

Again my sincere and deep gratefulness goes to my project guide, Prof, N.P. Gajjar and Course Coordinator Dr. N.M. Devashrayee for his kind support and marvelous guidance throughout my project work. His academic excellence continues to be a source of inspiration.

My heartiest thanks goes to my parents and my sister and my all friends Sharlik, Kaumil, Chirayu, Chirag, Vishal, Nirav, Ravi, Dishant and room mates, Manish, Vishal, Bharat, and Abhishek for their friendship, support, patience and collegial help, which was so essential during my project work. I also would like to thank Mr. Apurva (EC Dept., Nirma University) for helping me in final implementation.

Lastly I would I like to thank each and every person who has helped me directly or indirectly in accomplishing my project.

List of Figures

List of Tables

ABBREVIATIONS

BER	:	Bit Error Rate
CAP	:	Contention Access Period
CCA	:	Clear Channel Access
CFP	:	Contention Free Period
CSMA-CA	:	Carrier Sense Multiple Access - Collision Avoidance
DSSS	:	Direct Sequence Spread Spectrum
ED	:	Energy Detection
FFD	:	Full Functional Device
FPGA	:	Field Programmable Gate Array
GTS	:	Guaranteed Time Slot
IEEE	:	Institute Of Electrical
LLC	:	Logical Link Control
LQI	:	Link Quality Indication
LR-WPAN	:	Low Rate Wireless Personal Area Network
LUT	:	Look Up Table
MAC	:	Medium Access Control
MEMS	:	Micro Electronic Mechanical System
MLME	:	MAC Layer Management Entity
MPDU	:	MAC Protocol Data Unit
MUX	:	Multiplexer
O-QPSK	:	Orthogonal Quadrature Phase Shift Keying
PAN	:	Personal Area Network
PC-LAN	:	Personal Computer Local Area Network
PER	:	Packet Error Rate
PHR	:	PHY Header
PHY	:	Physical Layer
PLME	:	PHY Layer Management Entity
PPDU	:	PHY Protocol Data Unit
PSDU	:	PHY Service Data Unit
RAM	:	Random Access Memory
RFD	:	Reduced Functional Device
ROM	:	Read Only Memory
RTL	:	Register Transfer Logic
SAP	:	Service Access Point
SFD	:	Start Frame Delimiter
SHR	:	Synchronization Header
SSCS	:	Service Specific Convergence Sub layer
TV	:	Television
UWB	:	Ultra Wide Band
VHDL	:	Very high speed integrated circuit Hardware Description Language
WPAN	:	Wireless Personal Area Network
XST	:	Xilinx Synthesis Tool

Chapter – 1

Introduction

Up till now, the main focus in the wireless industry has been on wireless communication with higher data rates leaving out set of application, which require low data rate and latency requirement. So such need arises for a global standard, which offers low complexity, low data rate for the applications for which other global standards are inappropriate. IEEE 802.15.4 is a global standard designed for Low data rate, Low power consumption, Low cost applications. This standard will bring many simple and originally standalone devices into network.

In this chapter, first answer of some basic question like why such need arises for a new global standard?, what is the problem with other existing global standards? Will be given, next the application scenarios of IEEE 802.15.4 will be resented, then overview of the standard, which includes architecture of standard, different network topologies, Superframe structure, and two PHY specifications.

In chapter – 2, reviewed literature will be presented. In chapter – 3, proposed design and all its implementation issues will be discussed, in this chapter only core part of the design will be discussed like encoders and decoders. In chapter – 4 coding gain from the MATLAB plot and logical synthesis results and average power consumed by each component will be presented. In chapter – 5, conclusion will be drawn from the results discussed in chapter – 4, and future aspects of this project will be discussed. In Appendix – A to Appendix – D, only simulation and testing is covered. In Appendix – A Test Vectors are presented which is helpful in understanding of the figures in Appendix – D. In Appendix – B simulation waveform from the ModelSim waveform viewer. In Appendix – B Pin Interface diagram of each implemented components will be presented which helps in understanding the implemented design. And last in Appendix – D, testing strategy and testing waveform captured from Chipscope's waveform viewer will be presented.

1.1 WHY ANOTHER STANDARD?

The past several years have seen the rapid growth of wireless networking. So wireless networking has been mainly focused on high data rate and long-range applications. The effort

to increase the data rate can be clearly seen in the development of IEEE 802.11 standard series, from the initial 1-2 Mbps to as high as 54 Mbps in 802.11g. Bluetooth (IEEE 80.15.1) is the first well-known standard facing low data rate applications. The complexity of Bluetooth makes it expensive and inappropriate for some simple applications requiring low cost and low power consumption.

As more and more low cost high quality devices appear on the market and new application emerges every day, short range WPAN both low and high data rates are the horizon. Two major efforts of IEEE are underway to boost the development of WPAN. One is the specification of IEEE 802.15.3 also known as Ultra Wide Band (UWB) for high rate WPAN. Another one is IEEE 802.15.4 for low rate WPAN. As a matter of fact low data rates applications are closer to our daily lives than high data rates. If high data rates applications helps us in saving time then low data rate applications can helps us to save lives. The main obstacle of low data rate applications has been lack of feasible technique especially based on global standards. However with the release of IEEE 802.15.4 as well as advances in the field of embedded processors, MEMS, and radio technologies, low data rates applications are expected to play an important role in our lives.

1.2 APPLICATION SCENARIO

This IEEE Std. 802.15.4 (LR-WPAN, here after) supports application for which other existing global standards are inappropriate. List of application can benefit from this standard includes:

- Industrial and Home Automation
- Monitoring: safety, Environment
- Entertainment: Interactive games, PC peripherals

To elaborate on the potential extent to which 802.15.4 can save our daily lives. Here below are the few paragraphs, which elaborate some of the applications.

Life Saving: No matter how careful you are, it is still likely that at some point you will forget to turn off the gas stove or go to sleep without locking the door to your home. This oversight can cost you a lot. Science and Technology has improved our quality of life. There are many life-threatening scenarios such as fires, flood and earthquakes. Many life saving systems have proven to be ineffective. Some are too complicated and too expensive for pervasive

deployment, some stop working due to their rapid depletion of their batteries and others lack the ability to network, which is the key feature of life saving system. A global standard designed for low data rate, low power consumption and low cost applications, 802.15.4 has promise as a life saver. With the ability to connect simple devices such as sensors and actuators, an 802.15.4 system can monitor various events and automatically takes action when needed.

Home Automation: To free people from daily trivialities, home automation is one of the most attractive application of 802.15.4, regardless from the fact that the devices in your home come from the different manufacturers, they will be able to talk to each other and cooperate with each other. You can configure your home network in such a way that light intensity will automatically lowered when TV will on, or TV will mute when telephone rings or you pick up the phone or make a call. You can do even more than this. For example, everybody in your home can have a private electronic profile, which could be a tiny 802.15.4 compliant device and other device will react according to detecting this profile. Now the light, temperature, music, TV programs and web sites will automatically set according to your taste. You do not need to perform a complicated configuration for your profile to achieve above, since your profile can be built through some self learning procedures.

Precision Agriculture: Another challenging application for LR-WPAN is precision farming or Precision Agriculture. Precision agriculture is an environment-friendly system solution that optimize product quality and quantity while minimizing cost, human intervention and variation caused by unpredictable nature. Today agriculture is both user and environment demanding. It is mainly hardware oriented with manual and on-sight control using independent dumb machines, which produces unpredictable quality and quantity. With the new paradigm of precision agriculture, farming will become more information and software oriented, using automatic and remote controlled smart machine. This application requires large mesh-type networks consisting of thousands of LR-WPAN devices linked with sensors. These sensors will gather the information such as soil moisture content, nitrogen concentration and pH level. Weather sensors for measuring rainfall, temperature, humidity and barometric pressure will also provide the farmer with valuable information. Each sensor will pass the measured data to its corresponding LR-WPAN device, which in turn will pass it through the network to a central collection device. In order the sensor data to be useful, location awareness technology is necessary for correlating each sensor with its specific

location in the field. The combined information will give farmer an early alert to potential problem and allow him to achieve higher crop yield.

The precision agriculture application is at the low end of the LR-WPAN application range, requiring the transmission of few bits of data per day by each deployed devices. Data flow will be asynchronous in nature with minimum restriction on data latency. This combination of factors is advantageous for achieving long battery life. The challenge for this application is the topology of the network. Some nodes serve as repeater for others, relaying messages to the final destination. Since the application require mesh topology.

1.3 AN OVERVIEW OF 802.15.4

Objective behind giving overview is that the basics of the standard should be clear before understanding the proposed design.

1.3.1 ARCHITECTURE OF STANDARD

The LR-WPAN architecture consists if various layers. Each layer is responsible for one part of the standard and offers services to the higher layers. An LR-WPAN device comprises a PHY, which contains the RF transceiver along with its low level control mechanism, and a MAC sub layer that provides access to the physical channel for all types of transfer. The upper layers as shown in figure, consists of a network layer, which provides network configuration, manipulation, and message routing, and an application layer, which provides the intended function of the of the device. Fig – 1.1 shows the architecture of the standard.

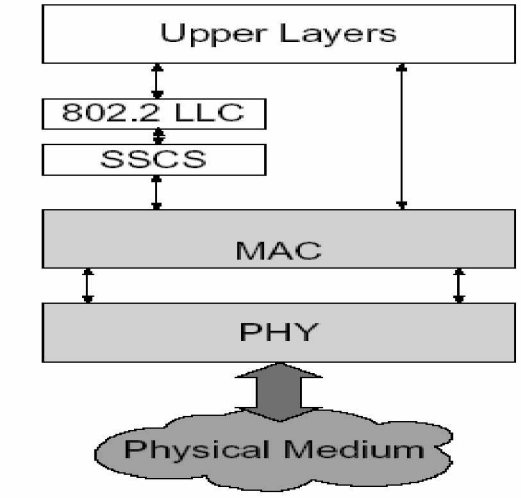


Figure 1.1 Architecture

PHY: - The PHY provides two services: the PHY data service and the PHY management service interfacing to the physical layer management entity (PLME). The PHY data service enables the transmission and reception of PHY protocol data units (PPDUs) across the physical radio channel. The features of the PHY are activation and deactivation of the radio transceiver, ED, LQI, channel selection, channel assessment (CCA), and transmitting as well as receiving packets across the physical medium.

MAC: - The MAC sub layer provides two services: the MAC data service and the MAC management service interfacing to the MAC sub layer management entity service access point (MLME-SAP). The MAC data service enables the transmission and reception of MAC protocol data units (MPDUs) across thy physical data service. The features of the MAC sub layer are beacon management, channel access, GTS management, frame validation, acknowledgement frame delivery, association, disassociations and controlling security mechanisms.

1.3.2 NETWORK TOPOLOGIES

The LR-WPANS may operate in either of two topologies: the star topology or the peer-to-peer topology. In the star topology the communication is established between the devices and a single central controller called the PAN coordinator. All devices operating on a network of either topology shall have unique 64 bit extended addresses. This address can be used for direct communication within the PAN, or it can be exchanged for a short address allocated by the PAN coordinator when the device associates.

The peer-to-peer topology also has a PAN coordinator; however, it differs from the star topology in that any device can communicate with any other device as long they are in range of one another.

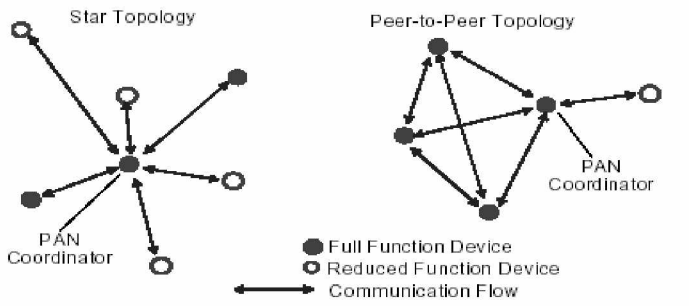


Figure 1.2 Network Topologies

1.3.3 Super frame structure

The format of the super frame is typically as shown below. The beacon frame is transmitted in the first slot of each super frame by the coordinator. The beacons are used to synchronize the attached devices, to identify the PAN, and to describe the structure of the super frames.

If a coordinator does not wish to use a super frame structure, it any turn off the beacon transmissions. Any device wishing to communicate during the contention access period (CAP) between two beacons shall compete with other devices using a slotted CSMA-CA mechanism.

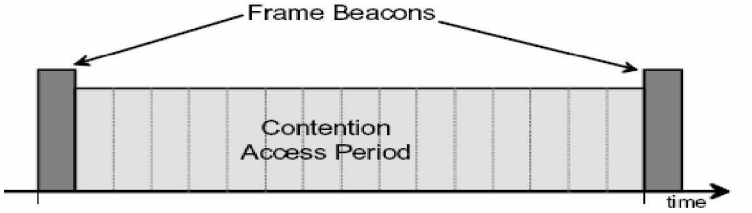


Figure 1.3 Superframe Structure without GTSs

For some low latency applications the PAN coordinator may dedicate portions of the active super frame to that application. These dedicated slots are known as guaranteed time slots (GTS). The GTSS form the contention free period (CFP), which always appears at the end of the active super frame.

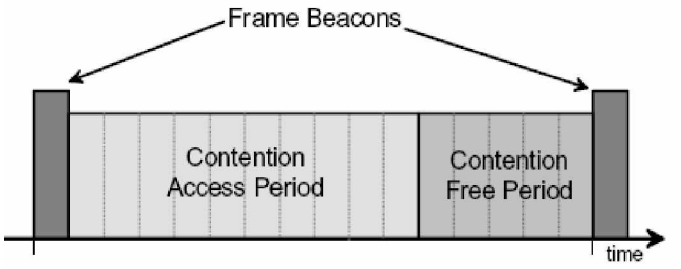


Figure 1.4 Superframe Structure with GTSs

1.3.4 PHY Specifications

The PHY is responsible for the following tasks:

— Activation and deactivation of the radio transceiver

— ED within the current channel

— LQI for received packets

— CCA for CSMA-CA

— Channel frequency selection

— Data transmission and reception

Operating frequency range

A compliant device shall operate in one or several frequency bands using the modulation and spreading formats summarized in table below:

Table – 1 Operating Frequency range

PHY (MHz)	Frequency Band (MHz)	Spreading parameters		Data parameters		
		Chip rate (kchip/s)	Modulation	Bit rate (kb/s)	Symbol rate (ks/s)	Symbols
868/915	868-868.6	300	BPSK	20	20	Binary
	902-928	600	BPSK	40	40	Binary
2450	2400-2483.5	2000	O-QPSK	250	62.5	16-ary Orthogonal

Channel assignments and numbering

A total of 27 channels, numbered 0 to 26, are available across the three frequency bands. 16 channels are available in the 2450 MHz band, 10 in the 915 MHz band, and 1 in the 868 MHz band. The center frequency of these channels is defined as follows:

$Fc = 868.3$ in MHz, for $k = 0$

$Fc = 906 + 2\,(k - 1)$ in MHz, for $k = 1, 2 \ldots 10$

And $Fc = 2405 + 5\,(k - 11)$ in MHz, for $k = 11, 12 \ldots 26$

Where k is the channel number.

For each PHY supported, a compliant device shall support all channels allowed by regulations for the region in which the device operates.

PPDU format

This clause specifies the format of the PPDU packet. The PPDU packet structure is presented so that the leftmost field as written in this standard shall be transmitted or received first. All multiple octet fields shall be transmitted or received least significant octet first and each octet shall be transmitted or received least significant bit (LSB) first. The same transmission order should apply to data fields transferred between the PHY and MAC sub layer.

Each PPDU packet consists of the following basic components:

— A SHR, which allows a receiving device to synchronize and lock onto the bit stream.
— A PHR, which contains frame length information.
— A variable length payload, which carries the MAC sub layer frame.

General packet format

The PPDU packet structure shall be formatted as illustrated

<table>
<tr><td>Octets: 4</td><td>1</td><td colspan="2">1</td><td>variable</td></tr>
<tr><td>Preamble</td><td>SFD</td><td>Frame Length
(7 Bits)</td><td>Reserved
(1 Bit)</td><td>PSDU</td></tr>
<tr><td colspan="2">SHR</td><td colspan="2">PHR</td><td>PHY Payload</td></tr>
</table>

Figure 1.5 General Packet Format

SHR

The preamble field is used by the transceiver to obtain chip and symbol synchronization with an incoming message. The preamble field shall be composed of 32 binary zeros.

The SFD is an 8 bit field indicating the end of the synchronization (preamble) field and the start of the packet data. The SFD shall be formatted as illustrated

Bit - 0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1	1	0	0	1	0	1

Figure 1.6 SDF Bit Pattern

PHR

The frame length field is 7 bits in length and specifies the total number of octets contained in the PSDU (i.e., PHY payload). It is a value between 0 and 127

PHY Payload

The PSDU field has a variable length and carries the data of the PHY packet. For all packet types of length five octets or greater than seven octets, the PSDU contains the MAC sub layer frame (i.e., MPDU).

1.3.5 PHY specifications – 2450 MHz

The requirements for the 2450 MHz PHY are specified below:

Data rate

The data rate of the IEEE 802.15.4 (2450 MHz) PHY shall be 250 kb/s.

Modulation and spreading

The 2450 MHz PHY employs a 16-ary quasi-orthogonal modulation technique. During each data symbol period, four information bits are used to select one of 16 nearly orthogonal Pseudo-random Noise (PN) sequences to be transmitted. The PN sequences for successive data symbols are concatenated, and the aggregate chip sequence is modulated onto the carrier using Offset Quadrature Phase-Shift Keying (O-QPSK).

Reference modulator diagram

The functional block diagram in Figure 1.7 is provided as a reference for specifying the 2450 MHz PHY modulation and spreading functions.

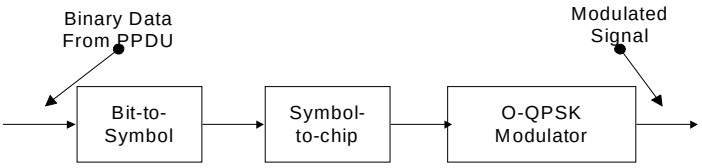


Figure 1.7 Reference Modulator Diagram

Bit-to-symbol mapping

All binary data contained in the PPDU shall be encoded using the modulation and spreading functions shown in Figure 1.7. This sub clause describes how binary information is mapped into data symbols. The 4 LSBs ($b0$, $b1$, $b2$, $b3$) of each octet shall map into one data symbol, and the 4 MSBs ($b4$, $b5$, $b6$, $b7$) of each octet shall map into the next data symbol. Each octet of the PPDU is processed through the modulation and spreading functions (see Figure 1.7) sequentially, beginning with the preamble field and ending with the last octet of the PSDU. Within each octet, the least significant symbol ($b0$, $b1$, $b2$, $b3$) is processed first and the most significant symbol ($b4$, $b5$, $b6$, $b7$) is processed second.

Symbol-to-chip mapping

Each data symbol shall be mapped into a 32-chip PN sequence as specified below. The PN sequences are related to each other through cyclic shifts and/or conjugation.

Data Symbols (decimal)	Data Symbols (binary) (b_0,b_1,b_2,b_3)	Chip Values ($c_0,c_1,\ldots,c_{30},c_{31}$)
0	0000	11011001110000110101001000101110
1	0001	11101101100111000011010100100010
2	0010	00101110110110011100001101010010
3	0011	00100010111011011001110000110101
4	0100	01010010000101110110110011100011
5	0101	00110101001000101110110110011100
6	0110	11000011010100100010111011011001
7	0111	10011100001101010010001011101101
8	1000	10001100100101100000011101111011
9	1001	10111000110010010110000001110111
10	1010	01111011100011001001011000000111
11	1011	01110111101110001100100101100000
12	1100	00000111011110111000110010010110
13	1101	01100000011101111011100011001001
14	1110	10010110000001110111101110001100
15	1111	11001001011000000111011110111000

Figure 1.8 PN Sequence Diagram

O-QPSK modulation

The chip sequences representing each data symbol are modulated onto the carrier using O-QPSK with half sine pulse shaping. Even-indexed chips are modulated onto the in-phase (I) carrier and odd-indexed chips are modulated onto the quadrature-phase (Q) carrier. Because each data symbol is represented by a 32-chip sequence, the chip rate (nominally 2.0 Mchips/s) is 32 times the symbol rate. To form the offset between I phase
and Q-phase chip modulation, the Q-phase chips shall be delayed by *Tc* with respect to the I-phase chips (see Figure 19), where *Tc* is the inverse of the chip rate.

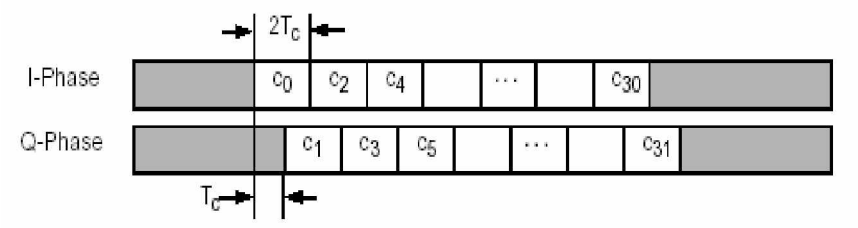


Figure 1.9 O-QPSK Modulator

1.3.6 PHY specifications - 868/915 MHz

The requirements for the 868/915 MHz band PHY are specified below.

868/915 MHz band data rates

The data rate of the 868/915 MHz band PHY shall be 20 kb/s when operating in the 868 MHz band and 40 kb/s when operating in the 915 MHz band.

Modulation and spreading

The 868/915 MHz PHY shall employ direct sequence spread spectrum (DSSS) with binary phase-shift keying (BPSK) used for chip modulation and differential encoding used for data symbol encoding.

Reference modulator diagram

The functional block diagram in Figure 21 is provided as a reference for specifying the 868/915 MHz band PHY modulation and spreading functions. The number in each block refers to the sub clause that describes that function. Each bit in the PPDU shall be processed through the differential encoding, bit-to-chip mapping and modulation functions in octet-wise order, beginning with the preamble field and ending with
The last octet of the PSDU. Within each octet, the LSB, b0, is processed first and the MSB, b7, is processed last.

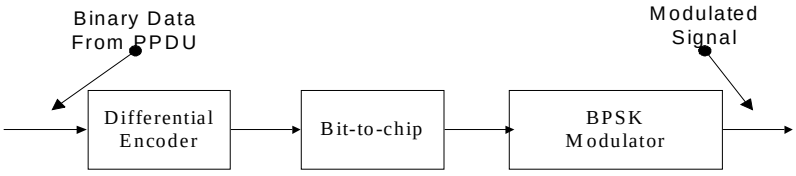


Figure 1.10 Reference Modulator Diagram

Differential encoding

Differential encoding is the modulo-2 addition (exclusive or) of a raw data bit with the previous encoded bit. This is performed by the transmitter and can be described by Equation below:

$$E_n = R_n \oplus E_{n-1}$$

Where,

R_n is the raw data bit being encoded,

E_n is the corresponding differentially encoded bit,

E_{n-1} is the previous differentially encoded bit.

For each packet transmitted, R_1 is the first raw data bit to be encoded and E_0 is assumed to be zero. Conversely, the decoding process, as performed at the receiver, can be described by Equation:

$$R_n = E_n \oplus E_{n-1}$$

For each packet received, E_1 is the first bit to be decoded, and E_0 is assumed to be zero.

Bit-to-chip mapping

Each input bit shall be mapped into a 15-chip PN sequence as specified below:

Input bits	**Chip values (c_0,c_1,…,c_{14})**
0	111101011001000
1	000010100110111

Figure 1.11 PN Sequence

BPSK modulation

The chip sequences are modulated onto the carrier using BPSK with raised cosine pulse shaping (roll-off factor = 1). The chip rate is 300 kchip/s for the 868 MHz band and 600 Kchip/s in the 915 MHz band.

Chip transmission order

During each symbol period, the least significant chip, *c*0, is transmitted first, and the most significant chip, *c*14, is transmitted last.

Operating frequency range

The 868/915 MHz PHY operates in the 868.0–868.6 MHz frequency band and in the 902–928 MHz frequency band.

So after being familiar with the standard, main objective is to implement PHY baseband processor base upon IEEE 802.15.4. Here below is the proposed design for baseband processing. PHY consist of two section, one is Transmitter and another one is Receiver.

In the Transmitter section, PHY accepts PSDU Length and PSDU parallely. Maximum 127 octet of PSDU is accepted [1], in the PHY Transmitter, we have used multiple channel coders with interleaver. Therefore, PSDU data is first formatted [1] into PPDU. PPDU is then passed through outer channel coder, which is Linear Block coder, Interleaver, and through inner coder which is convolution coder. Lastly, the output of inner coder is spreaded using the spreading sequence specified in the standard. In the Receiver section, exactly reverse procedure to decode data. First despreading, then through inner decoder, deinterleaver, and outer coder. Then extracting PSDU from PPDU [1]. Here below are the block diagram of transmitter and receiver.

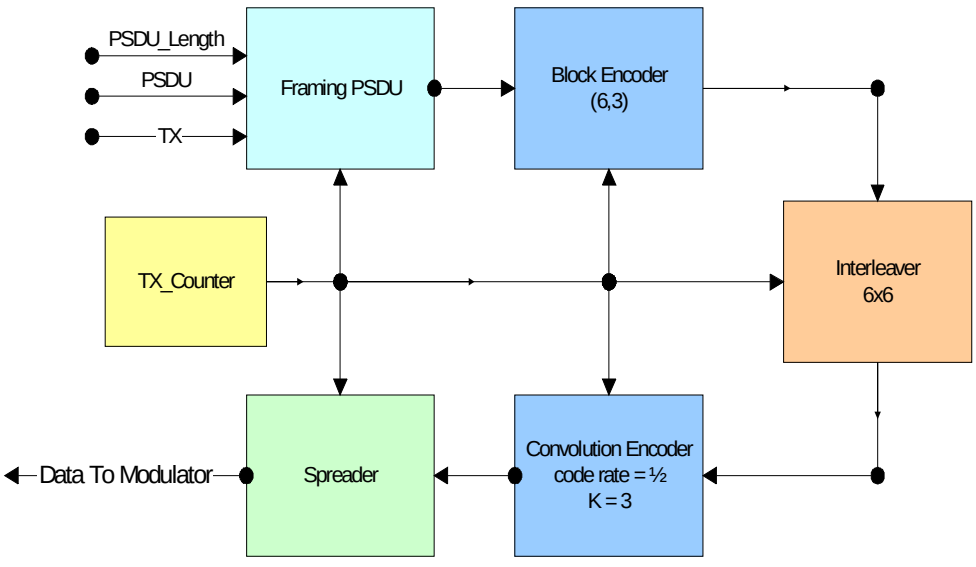


Figure 1.12 Transmitter Block Diagram

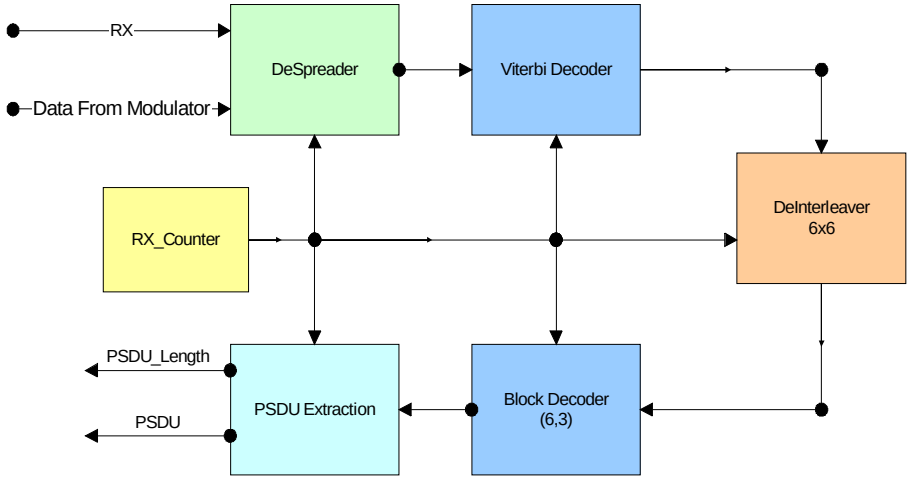


Figure 1.13 Receiver Block Diagram

Functionality and implementation issue of each block will be discussed in chapter-3.

Chapter – 2

Review Of Literature

Most of the literature used in this project was taken from IEEE Std. 802.15.4-2003, Wireless medium access control (MAC) and physical layer (PHY) specifications for low rate wireless personal area network (LR-WPAN). This standard describes the specifications of the physical layer (PHY), PPDU format, PHY features, and the functional details of the physical layer.

Other literature reviewed during the project was from some IEEE papers, VHDL language reference manual, and some textbooks. Though these papers were quite useful in understanding the various applications and performance appraisal of the PHY layer. But none of them were helpful in the design issues. And most of the content was similar to the standard [1].

1. Jianliang zheng and Myung J. Lee [2] discusses few application scenarios to show the potential extent to, which this new standard can affect, our lives and gives overview of standard.
2. Jose A. Gutierrez, Marco Naeve [3] discusses about the issues like what was the problem with the other existing standards?, this article also discusses the technical consideration necessary when implementing low cost, low power WPAN, this article also shows calculation regarding duty cycle and related current requirement.
3. Chris Evens [4] discusses about why such need arises for such a new standard?, he also discusses some applications like home automation, wireless meter reading and medical sensors. He also compare Zigbee with Bluetooth in terms of working of protocol, he also discuss about Zigbee Alliance and at last he made some prediction about the acceptance of the Zigbee.
4. Ed callway, Paul Gorday [5] discusses IEEE 802.15.4 standard and its home networking applications, they also discuss about different features like network flexibility, Low cost, low power consumption.

Other than these above mentioned IEEE papers, some books had also been refer.

1. Bernard Sklar [6] is used to study the implementation issues of encoders and decoders will be discussed in chapter – 3.
2. Whole system is implemented using VHDL, for that different books [7], [8], [10] were used. Kevin Skahill [7] discusses about the how good VHDL programming can be done and also discusses about synthesis issues in detail.
3. VHDL Golden reference guide contains syntax of all VHDL keywords alphabetically and also mention that it is supported by most common synthesis tools or not.
4. Regarding the Low power design using VHDL, refer to [12].

Chapter – 3

Proposed Design

As mention in chapter - 1 PHY offers two services, one is Data service and other one is PLME. Former one deals with transmission and reception of data and later one deals with the services like Link Quality Indication, Energy Detection, performing channel selection algorithm and managing its attributes. In this project we implement the base band part of data services.

In this chapter, core design and its implementation issues will be discussed and its simulation, synthesis, and testing results will be presented in Appendix – B, chapter – 4 and Appendix – D respectively.

So deem a system, which operate reliably with high performance in span of 10m to 15m typically. Here below are the block diagrams of transmitter and receiver in figure 3.1 and 3.2 respectively.

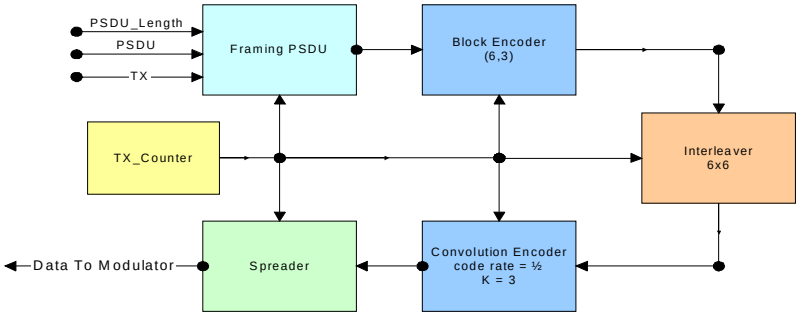


Figure 3.1 Transmitter Block Diagram

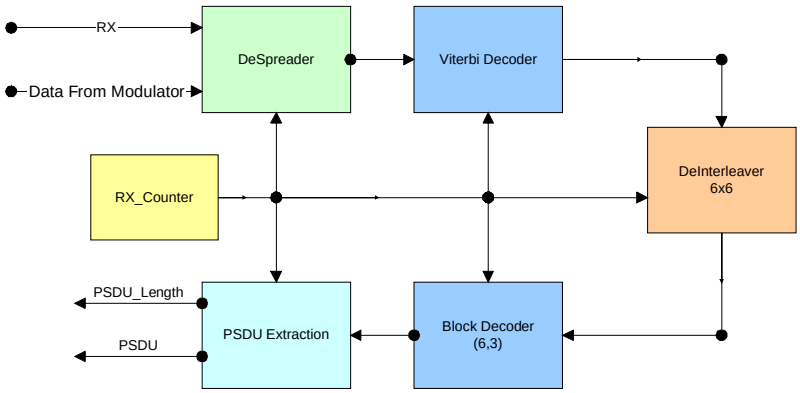


Figure 3.2 Receiver Block Diagram

Keep in mind that one to two bit of error correction codes are suffice for the span of 10m.

3.1 PPDU Formatting

This block is the first block of the transmitter. It takes PSDU_LEN (6:0) and PSDU (7:0) as the input and gives data (2:0) as the output to the next block, block coder. This block forms the PPDU from the PSDU given by MAC. The block diagram of this block is shown below. Write counter is used to count number of octets written in the buffer. Read counter is used to count the bits read by the next block.

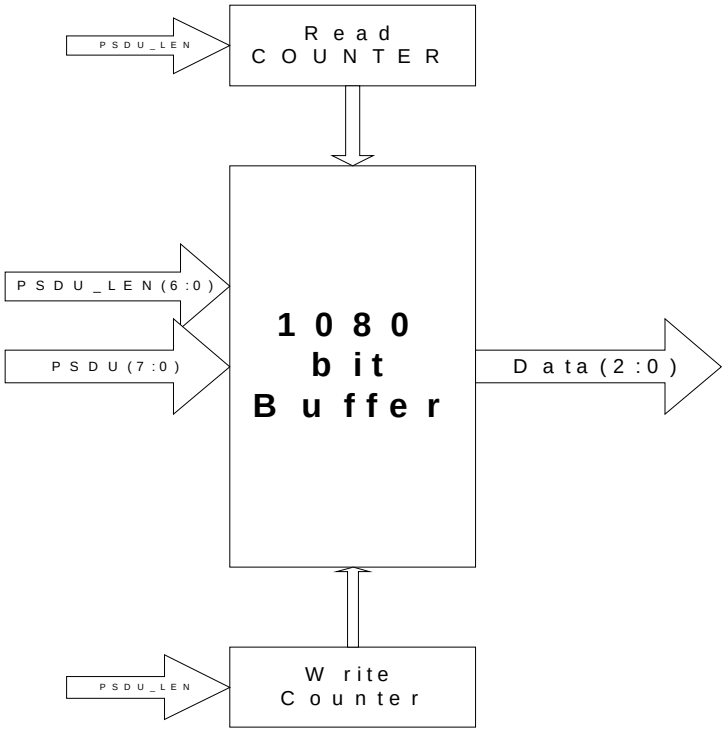


Figure 3.3 PPDU Formatting

3.2 Linear Block Coder

Linear block code is a class of parity check codes that can be characterized by (n, k) notation. Encoder transform k message bits into n output bits. k message bits forms 2^k distinct message sequence and at output is of n bits so, there are 2^n distinct code word of length 6. But out of 2^n only 2^k are the true codeword.

Out of n bits first k bits are same as message sequence and remaining n-k bits are linear combination of k message bits. Let say 'c' is output sequence of length n, and 'v' is input sequence of length k

$$c_n = v_k$$

$$c_{n-1} = v_{k-1}$$

$$c_{n-2} = v_{k-2}$$

……..

$$c_{n-k} = v_0$$

$$c_{n-k-1} = h_{11}v_k \oplus h_{12}v_{k-1} \oplus h_{13}v_{k-2} \oplus \ldots\ldots\ldots \oplus h_{1k}v_0$$

$$c_{n-k-2} = h_{21}v_k \oplus h_{22}v_{k-1} \oplus h_{23}v_{k-2} \oplus \ldots\ldots\ldots \oplus h_{2k}v_0$$

$$c_{n-k-3} = h_{31}v_k \oplus h_{32}v_{k-1} \oplus h_{33}v_{k-2} \oplus \ldots\ldots\ldots \oplus h_{3k}v_0$$

……

$$c_0 = h_{n1}v_k \oplus h_{n2}v_{k-1} \oplus h_{n3}v_{k-2} \oplus \ldots\ldots\ldots \oplus h_{nk}v_0$$

From this we can derive the generator matrix of size kxn as below:

$$\begin{bmatrix} 1\,0\,0\ldots\ldots\ldots 0 & h_{11} & h_{12} & \ldots\ldots\ldots & h_{1k} \\ 0\,1\,0\ldots\ldots\ldots 0 & h_{21} & h_{22} & \ldots\ldots\ldots & h_{2k} \\ \ldots\ldots\ldots\ldots & \ldots\ldots\ldots & & & \\ 0\,0\,0\ldots\ldots\ldots 1 & h_{n1} & h_{n2} & \ldots\ldots\ldots & h_{nk} \end{bmatrix}$$

So, Generator matrix of size 3x6 is

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Looking at the generator matrix we can say that if message sequence contains '1' at last bit position then output can be obtained by xor operation between last and second last message bits. We take LSB as first bit until further notice. Similarly if message sequence contains '1' on second position then output is xor between first and last bit of message and if '1' on first position then output is xor between first and second bit.

We will take you on the tour of how block encoding is done using an illustrative example. Let say message is "110". Output can be obtained by modulo-2 addition between first and second row of generator matrix. So we need only three two input xor gates.

= 100110 $\oplus$ 010101

= 110011, this is the true codeword.

This (6, 3) code can correct all one bit of error and one two bit of error [2].

3.3 Interleaver

Interleaver is nothing but a Memory. It can be viewed as a rectangular array of storage locations. Its depth can specify interleaver. In this design, depth of interleaver is six. It means that there are six rows, and number of column depends upon output of previous block. So here, six columns required. Input is entered row wise. Output is taken column wise. The block diagram of interleaver is shown in the figure below:-

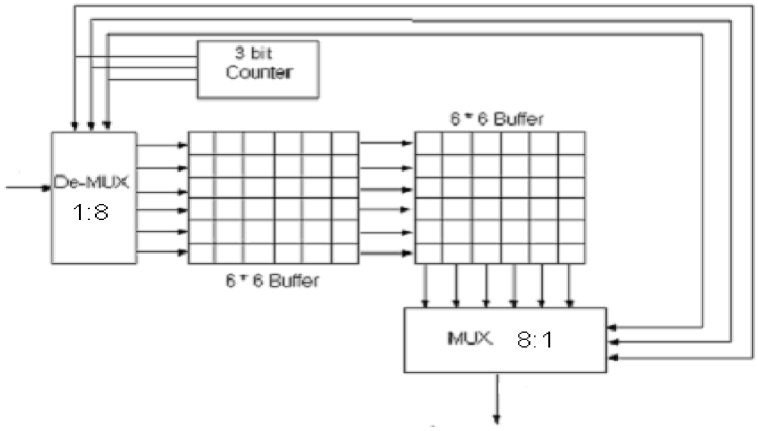


Figure 3.4: Block Diagram of 6*6 Interleaver

As shown in the block diagram, Components Required in the 6*6 Interleaver are

1. 6 * 6 bit buffer to store data row wise.
2. 6 * 6 bit buffer to read data column wise.
3. Counter : 3 bit binary synchronous up-counter with Asynchronous RESET.
4. De-MUX (8:1): to select the row of the buffer in which data is to be written.
5. MUX(8:1) : to select the column of the buffer from which data is to be read.
6.

The state diagram of the counter is as shown below: -

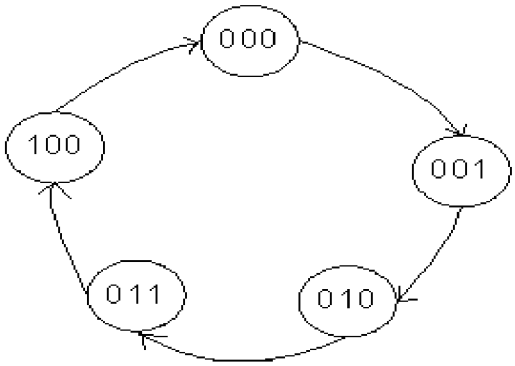


Figure 3.5 State Diagram of Counter

Main advantage of using interleaver-deinterleaver is that it helps us to fight against burst errors. Let say for example there are six consecutive bits are in error. Received data are entered column wise, so each row contain only one bit of error, these data are then inserted into decoder row wise, which can correct one bit of error. So like this interleaver and deinterleaver helps to fight against burst error. Disadvantage of using interleaver-deinterleaver is that it causes large amount of delay as depth increases. The reason is straight forward, until interleaver-deinterleaver is not fully filled, no further processing occurs.

3.4 Convolution Encoder

Convolutional codes, first introduced in 1955 by Elias, differ from block codes as follows: In a block codes, the block of n bits are generated by the encoder in any particular time unit depends only on the block of k input data bits within that time unit. In a convolution codes, block of n bits generated by the encoder in a particular time unit depends not only on block of k message bits within that time unit but also on previous K-1 (K= constraint length) bits in encoder.

Convolution coder can be specified by the constraint length (K) and code rate (r). Here in this design K = 3 and code rate = ½. Constraint length means no. of register in coder. In code rate is k/n in (n, k) code, where k is no. of message bits and n is no. of output bits. Here below is the structure of convolutional encoder.

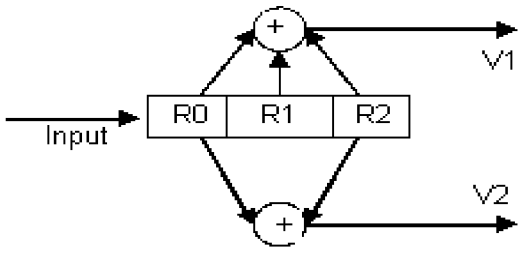


Figure 3.6: Convolutional Encoder

Message bits are stored in a temporary variable with two zero appended at end. Here below is the whole operation flow:

1. Before start encoding reset R0, R1 and R2.
2. First bit starting from LSB is inserted in i.e. R0 <= LSB.
3. Output is taken i.e. V1V2
4. Present content of registers are shifted right by one position.
5. Then insert next message bit.

Now, for remaining all the bits of temporary variable repeat steps from 3 to 5.

We require two basic components.

1. Component, which can shift right by one position.
2. Component, which computes output; that is performing XOR operation.

But keep in mind that before we start encoding we need to flush out previously stored data. In describing hardware, we call first component seven times because first call is to flush out old data and remaining six calls are for six message bits and second component six times.

3.5 Spreader

In this block, PN sequences are used to spread the incoming data. We do not have to bother about to generate PN sequences. It is already defined in standard itself as shown in the table 1 and table 2 below for OQPSK and BPSK modulation respectively. For BPSK modulation 15 chips are assigned to 1 bit, and in OQPSK 32, chips are assigned to 4 bit of symbol. To see the PN sequence, refer to standard [1].

Table - 2 PN Sequences

Data symbol (decimal)	Data symbol (binary) (b_0, b_1, b_2, b_3)	Chip values (c_0 c_1 ... c_{30} c_{31})
0	0000	11011001110000110101001000101110
1	1000	11101101100111000011010100100010
2	0100	00101110110110011100001101010010
3	1100	00100010111011011001110000110101
4	0010	01010010001011101101100111000011
5	1010	00110101001000101110110110011100
6	0110	11000011010100100010111011011001
7	1110	10011100001101010010001011101101
8	0001	10001100100101100000011101111011
9	1001	10111000110010010110000001110111
10	0101	01111011100011001001011000000111
11	1101	01110111101110001100100101100000
12	0011	00000111011110111000110010010110
13	1011	01100000011101111011100011001001
14	0111	10010110000001110111011100011100
15	1111	11001001011000000111011110111000

Input bits	**Chip values (c_0,c_1,…,c_{14})**
0	111101011001000
1	000010100110111

Figure 3.7 Functional Diagram of BPSK Spreading

In above diagram, we require a LuT (2x15) and 15 Mux for BPSK spreader. Here such 15 Mux are required. One Mux for each chip. Selection line of all Mux is common to all Mux.

Selection line is nothing but the one-bit input signal of spreader here it is presented as **a**.

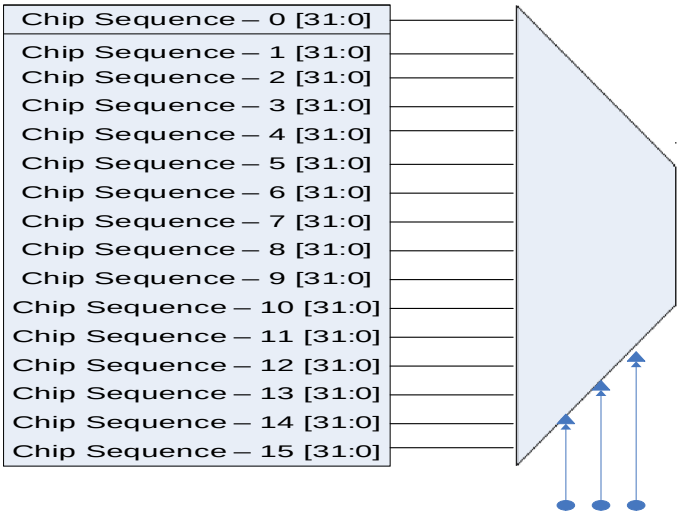


Figure 3.8 Functional Diagram of OQPSK Spreader

In above diagram, we require a LuT (16x32) and 32 multiplexers for OQPSK Spreader. Here such 32 Mux are required. One Mux for each chip. Selection line of all Mux is common to all Mux. Selection line is nothing but the four-bit input signal of spreader here it is presented as **a, b, c, d**.

3.6 Despreader

As shown in figure 3.2, demodulated data will face despreader first, same PN sequences are used to despread the received data. These PN sequences are stored in a look up table.

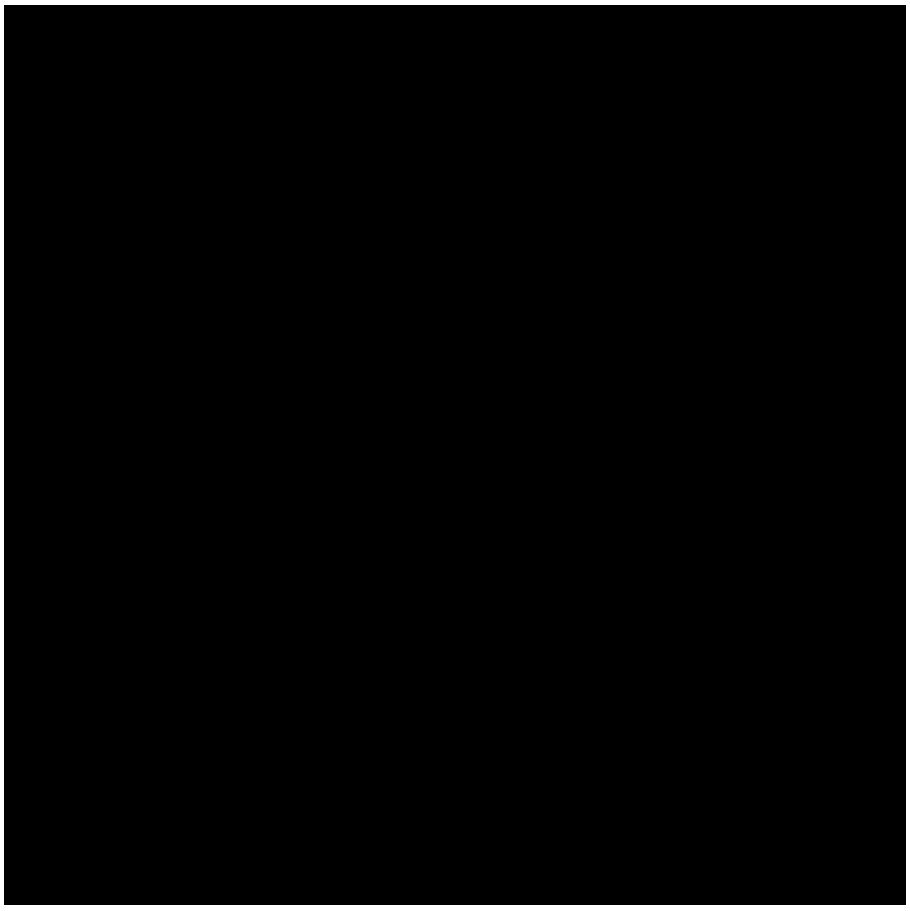


Figure 3.9 OQPSK Despreader Block Diagram

LuT: In above diagram, LuT contains 16 rows; each contains 32 chips PN sequence. Each row output is connected to respective hamming distance calculation block.

Hamming Distance Block: This block calculates the hamming distance between the received data and all the chip sequences in LuT. Output of this block is integer. So all 16-integer values are supplied to comparator to get the despreaded data.

Comparator: This comparator takes 16-hamming distance values, and gives 4-bit binary despreaded data. This comparator assign a 4-bit binary index to each integer value, it simplifies the job of comparator to despread the data. For ex. Value from hamming block-0 is assign an index of 0000b, similarly hamming block-15 is assign an index of 1111b.

There are two cases one is if any of the received hamming distances value is zero and second is all values are non-zero. In first case, output will be simply its index without any comparison. However, in later case all the 16 data are sorted in ascending order with their indexes in order to find the minimum distance value. This decoding scheme is called **maximum likelihood decoding**. For ex. value from hamming distance block-10 having minimum Value of all. In comparator it is assign an index 1010b, so after sorting this value is on top with its index, so output will be 1010b.

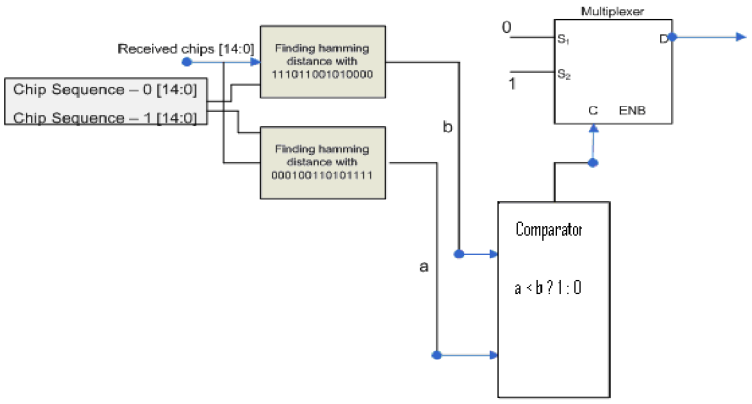


Figure 3.10 BPSK Despreader Block Diagram

Above diagram represents despreader in BPSK modulation scheme. It contains same components as of OQPSK mod. Scheme. Comparator issues logic 'high' if a < b (as shown in figure) otherwise logic 'low' to selection line of 2x1 mux.

3.7 Viterbi Decoding

Among various decoding methods for convolutional codes, viterbi's maximum likelihood algorithm is one of the best techniques yet evolved for digital communication where energy efficiency dominates in performance.

What is maximum-likelihood decoding?

The maximum-likelihood receiver implies selecting a codeword closest to the received word, because there are 2^k (k is no. of input bits) code words. This calculation is extremely difficult for large k and whole result in an overly complex decoder.

How maximum-likelihood decoding is applicable to viterbi's algorithm?

A major simplification was made by Viterbi in the maximum likelihood calculations by noting that each of the four nodes in figure-3.11 a, b, c and d has only two predecessors; that is each node can be reached through two nodes only, and the path that agrees most with the received sequence (minimum-distance path) need be retained for each node.

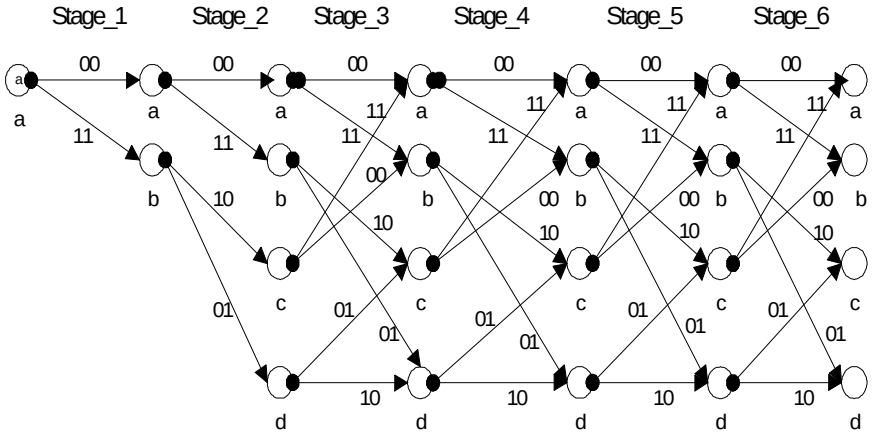
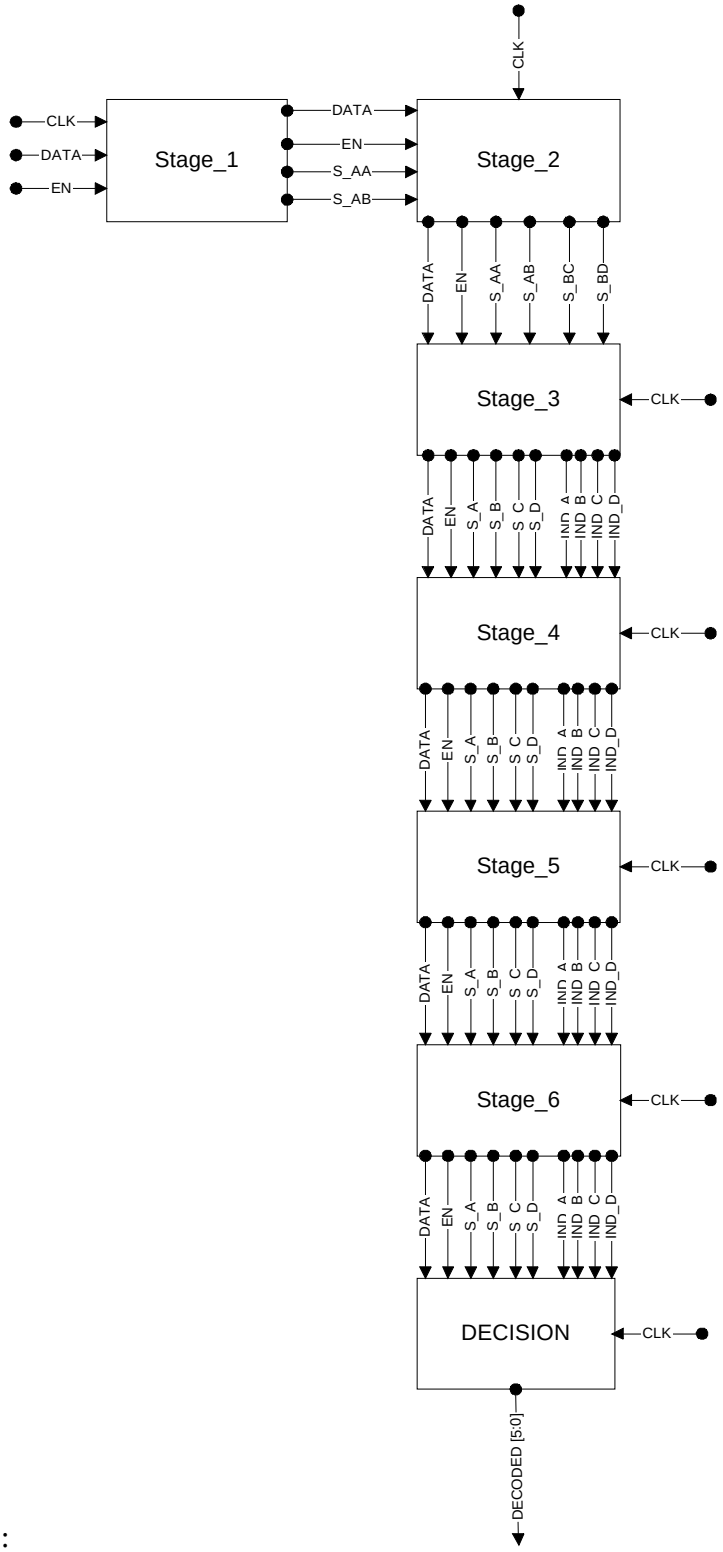


Figure 3.11 Trellis Diagram

Here below is the proposed architecture:



:

Figure 3.12 : Block Diagram of Viterbi Decoder

As shown in trellis diagram, there are six stages; that is it takes 12 bits of input and 6 bits of output. Stage_1, Stage_2, Stage_3, Stage_4, Stage_5 and Stage_6 of trellis diagram is Stage_1 to Stage_6 in Block diagram. And decision stage decides which address to select from available one of the four minimum hamming distances from stage_6.

Now we will see some more detail of each stage and proposed architectures of each:

Stage_1:

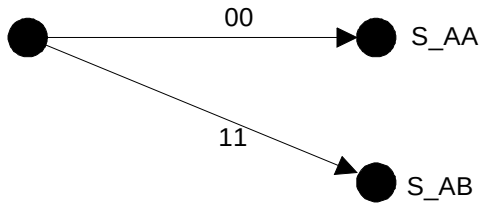


Figure 3.13 Stage - 1 trellis

In this stage, first two received bits are compared with “00” and “11”. Output of this comparison is integer, which is stored in s1_aa and s1_ab respectively. These integers represent the hamming distance.

Figure 3.14 Stage 1 Architecture

Fig. – 3.14 is detailed diagram of stage_1. Block ‘Hamming’ find the hamming distance of the received bits with states **a** (“00”) and **d** (“11”).

Stage_2:

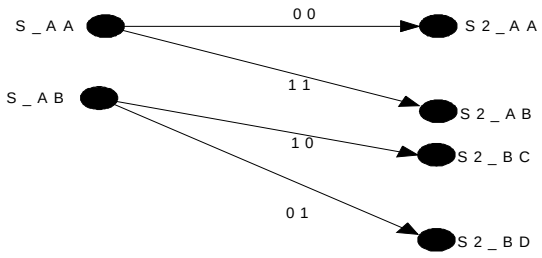


Figure 3.15 Stage 2 trellis

In this stage, second two received bits are compared with “00”,“11”,”10” and “01”. Output of this comparison is integer, which is S2_AA,S2_AB,S2_BC and S2_BD respectively. These integers represent the hamming distance. Fig – 3.16 is the detailed diagram of stage_2. Recall that ‘s1_aa’ and ‘s1_ab’ are from previous stage, which are added to hamming distance of this stage.

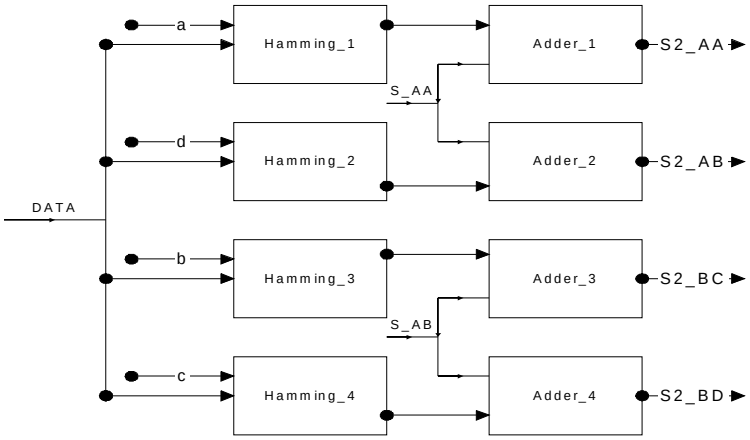


Figure 3.16 : Stage 2 architecture

Here below are the mathematical equations of how hamming distance is calculated in this stage.

S2_AA = S_AA + ham(data,”00”)

S2_AB = S_AA + ham(data,”11”)

S2_BC = S_AB + ham(data,”10”)

S2_BD = S_AB + ham(data,”01”)

‘Ham’ is the function, which calculates the hamming distance from the supplied data. Same thing is presented in detailed diagram in block ‘Hamming’. Block ‘Hamming’ contains two XOR gates and based on this output pattern of XOR gates hamming distance value is assign.

Stage_3:

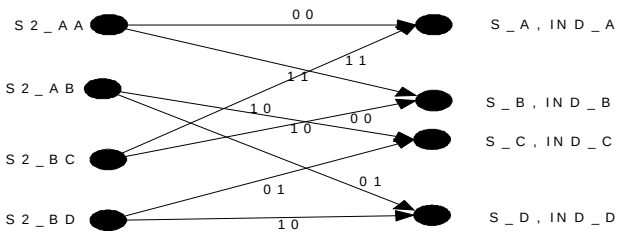


Figure 3.17 Stage 3 trellis

In this stage, next two received bits are compared with all the states. But in this stage situation is different from the previous stages. Each node on right hand side having two path inputted, out of which we select the maximum-likelihood path. Note that 'S2_AA', 'S2_AB', 'S2_BC' and 'S2_BD' are from second stage. Note that each node on right side in above diagram has two input path, so total of 8 paths. Now we assign address to each path as below.

Here total of 8 paths, so 3 bit of address starting from "000" to "111". "000" to "011" is assign to upper path of each node. And "100" and "111" are to lower path of each node starting from first node.

After getting all the eight hamming distances and its corresponding addresses, we have to select one of the maximum-likelihood path; that is minimum-distance path from each node named 'S_A', 'S_B', 'S_C' and 'S_D'. Along with that select corresponding address named 'IND_A', 'IND_B', 'IND_C' and 'IND_D'. Here below is the detailed diagram of stage_3.

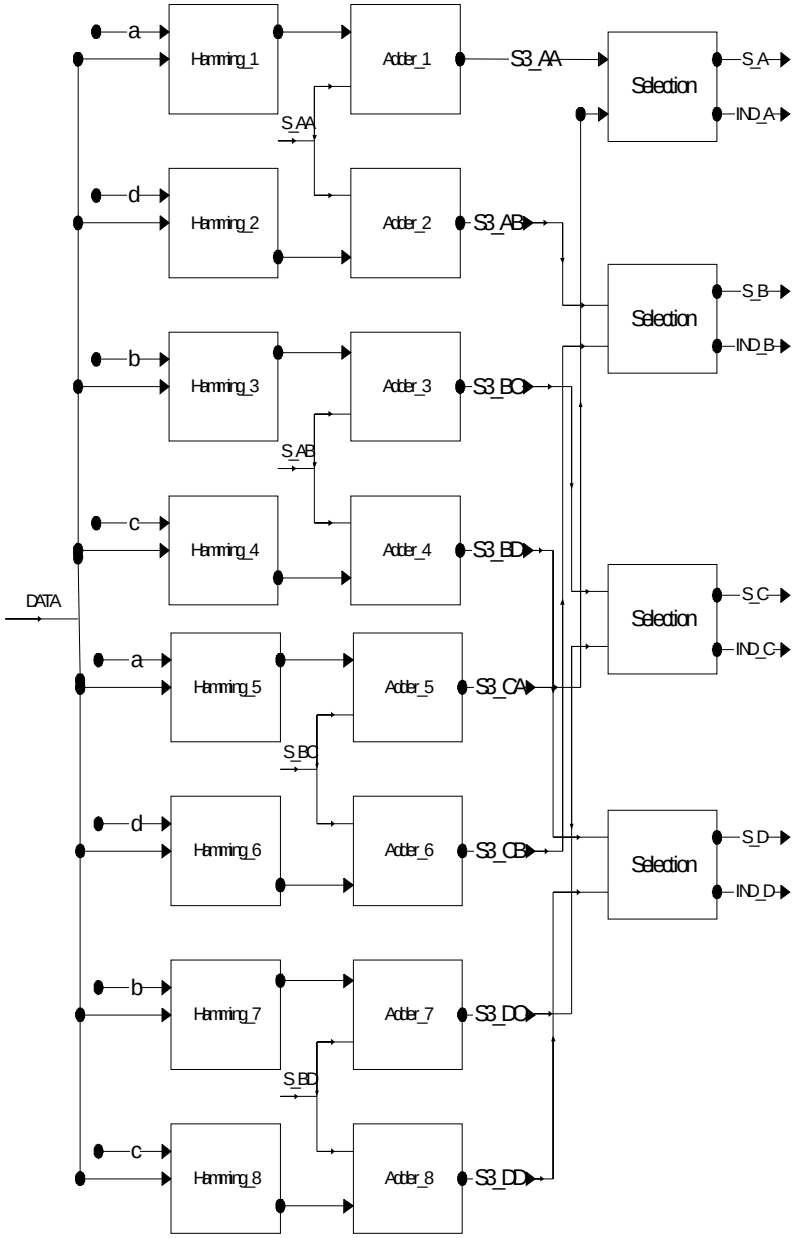


Figure 3.18 Stage 3 Architecture

As shown above in detailed diagram, we require eight hamming blocks because there are 8 paths. Then selection block, which select the optimum distance and corresponding address. Same mathematical equation can be applied to this stage with change in variable name.

As mention above, this is pipelined Viterbi decoder. From the trellis diagram it is clear that stage_4, stage_5 and stage_6 are same as stage_3, but the difference is that in each stage address value is modified i.e address from Stage_3 is of 3 bits and in Stage_4 it gets modified to 4 bits It can be represented by same block diagram and equations. As shown in trellis diagram, stage_4, stage_5 and stage_6 are same in terms of hardware. Last is the decision stage, input to this block is four hamming distance values and four address values of length 6

bits from stage_6. So decision is taken based upon minimum hamming distance from stage_6, in decision one of the four index values are selected.

This is pipelined Viterbi decoder. As shown in block diagram (figure 3.12) input data is passed to subsequent stages for decoding. This decoder starts giving output after 24 clocks of latency after it is enabled, after this much latency it puts decoded data on bus on each clock.

3.8 Deinterleaver

Deinterleaver performs exactly reverse operation to that of interleaver. Deinterleaver with the same specification is used at the rear end also. However, the difference is that here the received data is entered column wise and read row wise. Main advantage of using deinterleaver is that it spread out the burst errors in the received bits. So it can be easily handled. Diagram remains same, as that of interleaver as shown below: -

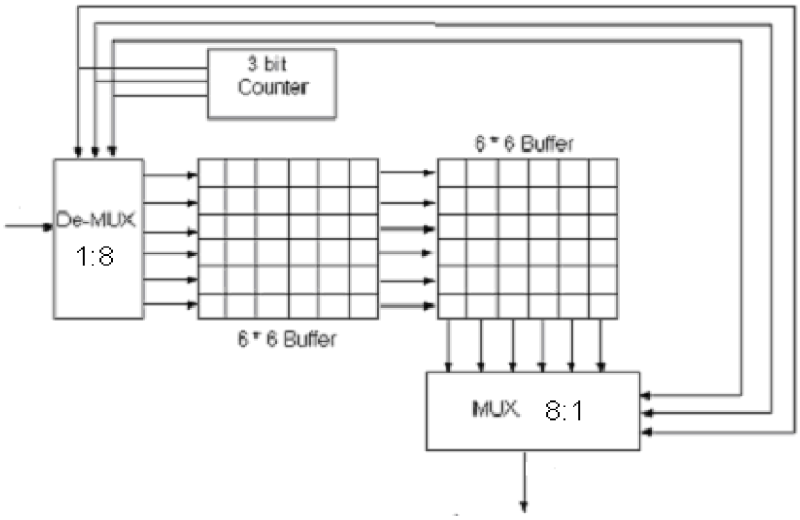


Figure 3.19 6*6 Deinterleaver

3.9 Linear Block Decoder

This linear block decoding is characterized by same notation as encoding. This is (6, 3) block decoding. Block diagram of this component is drawn below.

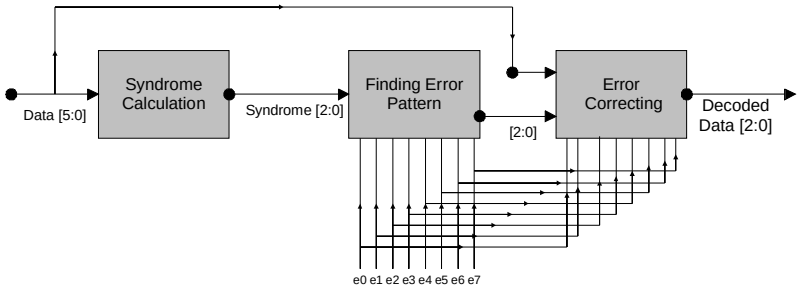


Figure 3.20: Block diagram of Block Decoder.

It also explains the flow for decoding.

1. Syndrome calculation
2. Finding error pattern
3. Error correcting

This flow simplifies our work of describing hardware. In syndrome calculation received codeword is multiplied with parity matrix, derived from generator matrix. Here below is the illustrative example of how syndrome is calculated.

$$= \begin{bmatrix} \overline{1} & 1 & 1 & 0 & 0 & \overline{1} \end{bmatrix} * \begin{bmatrix} \overline{1} & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \overline{1} \end{bmatrix}$$

$$= 0\ 0\ 1$$

If value of syndrome is zero then it indicates that received bits are true codeword else if its value is nonzero then received bits contain error. Keep in mind that by the value of syndrome we cannot predict, whether received bits contain one bit, two bit or all bits of error. This can be predicted by second block i.e. finding error pattern block.

In implementing this block, we need to perform matrix multiplication. But here parity matrix is constant of size 6x3. So any bit multiplied with zero is zero. So received bits are multiplied with each column of parity matrix. Then xor operation is performed within the multiplied bits. So we just perform xor operation between 5^{th}, 4^{th} and 2^{nd} bit of received bits for the MSB of syndrome, similarly xor between 5^{th}, 3^{rd} and 1^{st} bit of received bits for position of second MSB and xor between 4^{th}, 3^{rd} and 0^{th} bit of received bits for position of LSB. Like this we get 3 bit of syndrome.

After getting syndrome, we find respective error pattern from LUT. Error pattern length is same as code word length. Syndrome works as address for LuT. This error pattern is then supplied to error correcting block, which contains nothing but three XOR gates because in error correcting block modulo 2 addition takes place between received code and error pattern, and first three bits from MSB is the decoded data, so performing modulo 2 addition between these three bits only, like this we can save 3 xor gates!

Keep in mind that this decoder can correct all one bit of error and one two bit of error.

Chapter – 4

Results & Discussion

In this chapter, Synthesis result and average power consumed by different blocks of Transmitter and Receiver will be presented. Authors have first designed whole system in MATLAB, then in VHDL. Later in this chapter techniques used to reduce the power consumption will be discussed.

In presenting results, first is the MATLAB simulation result, which shows the achieved coding gain. Fig – 4.1 is the plot.

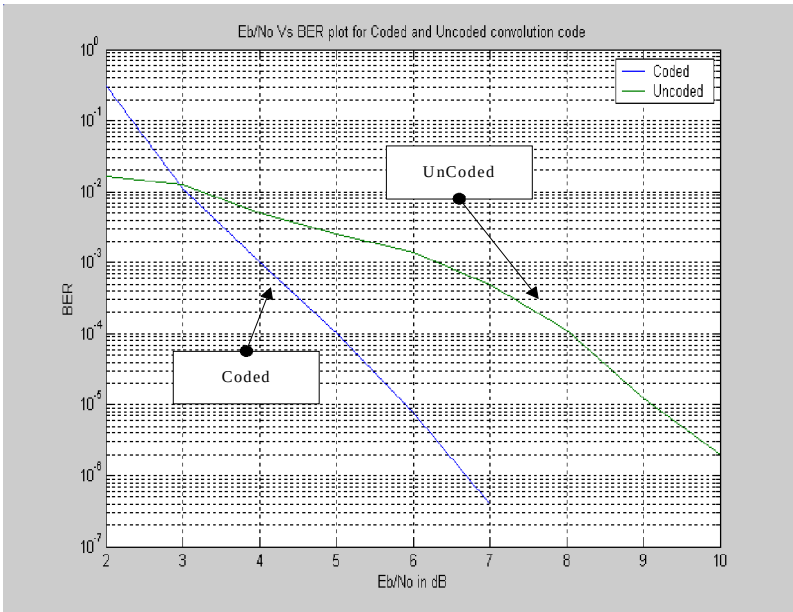


Fig. 4.1 E_b/N_o V_s BER plot for coded and uncoded convolution code.

From the plot, we can say that if we operate our system at BER of 10^{-5}, we achieve coding gain of appr. 3.1 dB.

Now logical synthesis results, set-up will be as follows:
Target device used is Xilinx Virtex 300 with speed grade –5 (xcv300pq240-5) offers 300K-gate count with 3072 slices. Here this mentioned FPGA is used just for prototyping. Here below are the logical synthesis and P_{avg} results in tabular form.

Component	**Slices Used**	**Gate Count**	**F_{max} (MHz)**	**P_{avg} (mW)***
Block Encoder	02	66	260	38.67
Interleaver	41	663	185	50.85
Convolution Encoder	05	150	158	36.9
Spreader	48	1311	200	189
Despreader	1254	14639	110	273.32
Viterbi Decoder	343	6500	120	187.8
Deinterleaver	41	663	185	50.85
Block Decoder	03	84	300	35.77

* P_{avg} is measured at 100 MHz clock frequency.

Below are some of the main techniques used to reduce average power consumption.

1. Gated Clock, means clock is gated with signal from a combinational circuit or gated with an enable signal. It means that clock will not be assign when the device is idle or not in use. This technique is very efficient because in case of interleaver P_{avg} without gates clock is 97.33 mW and with gated clock is 50.85 mW, so 46.48 mW of power saving!
2. Insert latches before the combinational part of component.

   Consider a decoder with two inputs and four outputs, if some combinational circuits drive these two inputs, so it may be possible that the combinational circuit is hazardous, so unwanted transitions occur on decoder input before steady state. And this unwanted transition consumes lots of power, if we put an enable signal these unwanted transitions can be avoided.

Another technique used to reduce power consumption is pipelining. As we know combinational circuits are hazardous. If combinational circuit is multi level then if glitch occurs at starting level then glitch propagates through remaining part of circuit, so lots of unwanted transition occurs and lots of power will be consumed. So solution to this problem is pipelining, in pipelining large combinational circuit is divided into small parts and registers are inserted between them. So now even glitches occur, inserted registers prevent it to propagate through whole design.

Chapter – 5

Conclusion & Future Scope

PHY design having discussed architecture has been simulated and synthesized and implemented in this project work. Following conclusions can be drawn from this project work.

1. BER in simulation is found to be 10^{-5} for E_b/N_0 = 5.5 dB. BER for some other values of E_b/N_0 is shown in the plot.

2. Coding gain through convolution coder is found to be appr. 3.1 @ BER of 10^{-5}.

3. The complete PHY design is synchronous and most of the synchronous components can be operated at a clock frequency of more than 100 MHz. So the design is targeted to operate at a clock frequency of 100 MHz.

Future Scope:

In this project PHY baseband processor is implemented, so future scope of this project is to implement MAC layer of LR-WPAN, and after implementing it, interface MAC and PHY. Currently an alliance of more than 80 companies named Zigbee Alliance is working on this standard. Job of this alliance is to specify the upper layer specifications in figure 1.1.

References

[1]. IEEE Std. 802.15.4- 2003, Wireless Medium Access Control (MAC) and Physical Layer (PHY) specification for Low Rate Wireless Personal Area Network (LR-WPAN).

[2]. "Will IEEE 802.15.4 make ubiquitous networking a reality? :- A discussion on potential Low-power, Low bit rate standard?" By Jianliang zheng and Myung J. Lee, The city college of Cuny, Communication magazine, IEEE, Vol. 42, issue 6, June – 2004, pp 140-146.

[3]. "IEEE 802.15.4 : A developing standard for low power, low cost wireless personal area network" By Jose A. Gutierrez, Ed Callway, Network, IEEE, Vol. 15, issue 5, sep-oct 2001, pp 12-19.

[4]. "Is the Zigbee wireless standard, promoted by an alliance of 25 firms, a big threat to Bluetooth" By Chris Evens – Pughe, IEEE review, Vol. 49, issue 3, March – 2003, pp 28-31.

[5]. "Home Networking with IEEE 802.15.4: A developing standard for Low Rate Wireless Personal Area networks" By Ed callway, Paul Gorday, Communication magazine, IEEE, Vol. 40, issue 8, Aug. 2002, pp 70-77.

[6]. "Digital Communication – Fundamentals and Applications" By Bernard Sklar, 2nd Ed., Pearson Education.

[7]. "VHDL for programmable devices" By Kevin Skahill, Cypress Semiconductor.

[8]. "Advanced VHDL techniques " - Doulos

[9]. "VHDL Language Course" By Clive Homes, Rutherford Appleton Laboratory, UK.

[10]. "VHDL Introduction from Simulation to Synthesis" By Sudhakar Yalamanchili, 2nd Ed., Pearson Education.

[11]. "VHDL Golden reference manual" By Doulos.

[11] www.zigbee.org

[12] www.actel.com

Appendix A Test Vectors

In this appendix we present the input & output data of each block of Transmitter and receiver for which the simulation and testing waveforms have been included in the Appendix B and Appendix D respectively. So that reader can understand the operation of the design.

As mention in chapter – 1, maximum no. of octets in PSDU is 127. Here for testing purpose only one octet is supplied. That will make the understanding easier. Octet supplied from MAC is "11111111". So PPDU become 00000000000000000000000000000000 11100101 10000000 11111111.

Input to the Block Encoder: -

'000', '000', '000', '000', '000', '000', '000', '000', '000', '000','001','110','010','110','000','000','111','111','110'.

Output of the Block Encoder or input to the Interleaver: -

'000000','000000','000000','000000','000000','000000',
'000000','000000','000000','000000','001011','110011',
'010101','110011','000000','000000','111000','111000',
'110011','110011','110011','110011','110011','110011'.

Output of the Interleaver or Input to the Convolution encoder: -

Interleaver – 1	Interleaver – 2	Interleaver – 3
000000	000000	010101
000000	000000	110011
000000	000000	000000
000000	000000	000000
000000	001011	111000
000000	110011	111000

Interleaver – 4

110011
110011
110011
110011
110011
110011

Data is entered row wise and taken out column wise and inputted to convolution encoder.

Output of the Convolution Encoder or Input to the Spreader: -

000000000000
000000000000
000000000000
000000000000
000000000000
000000000000
000000110111
000011011100
000000000000
000011101011
011111101011
101111101011
000011101011
110000000000
011100000000
101100000000
010101011011
010101011011
000000000000
000000000000
010101011011
010101011011

Spreaded Data to be stored in the TX Buffer: -

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111000100010111011011001110000110101100111000010101001000101110110
```

```
1101100111000011010100100010111001100000011101111011100011001001000001110111101110010110010
```

```
1101100111000011010100100010111011011001110000110101001000101110110011100001101010010001011101
```

```
1101100111000011010100100010111010010110000001110111101110001100011101111011000111001010010
```

```
1001110000110101001000101110110110010110000001110111101110001100011101111011000111001010010
```

```
0111011110111000110010010110000010010110000001110111101110001100011101111011000111001010010
```

```
1101100111000011010100100010111010010110000001110111101110001100011101111011000111001010010
```

```
0000011101111011100011001001011011011001110000110101001000101110110011100001101010010001011101
```

```
1001110000110101001000101110110111011001110000110101001000101110110110011100001101010010001011101101100111000
```

```
0111011110111000110010010110000011011001110000110101001000101110110110011100001101010010001011101101100111000
```

```
0011010100100010111011011001110000110101001000101110110110011100011101111011100011001001011000000
```

```
0011010100100010111011011001110000110101001000101110110110011100011101111011100011001001011000000
```

```
1101100111000011010100100010111011011001110000110101001000101110110110011100001101010010001011101101100111000
```

```
1101100111000011010100100010111011011001110000110101001000101110110110011100001101010010001011101101100111000
```

```
0011010100100010111011011001110000110101001000101110110110011100011101111011100011001001011000000
```

```
0011010100100010111011011001110000110101001000101110110110011100011101111011100011001001011000000
```

Now at the receiver side, exactly reverse procedure to decode the encoded data. As mention earlier that transmitter and receiver are connected in loop back fashion to verify the operation.

Despreaded Data is same as input to spreader, viterbi's output is same as convolutional encoder's input, deinterleaver's output is same as input of interleaver and block decoder's output is same as actual input at the transmitter.

Appendix B Simulation Waveforms

This Appendix presents the Simulation waveforms captured from Modelsim waveform viewer. Output from each block has been presented independently as well as the transmitter and receiver simulation waveforms are also been presented. Test Vectors are taken from appendix A, so it is easy for the reader to understand.

1.Transmitter

Block Encoder: -

Figure B.1 Simulation of Block Encoder

Interleaver

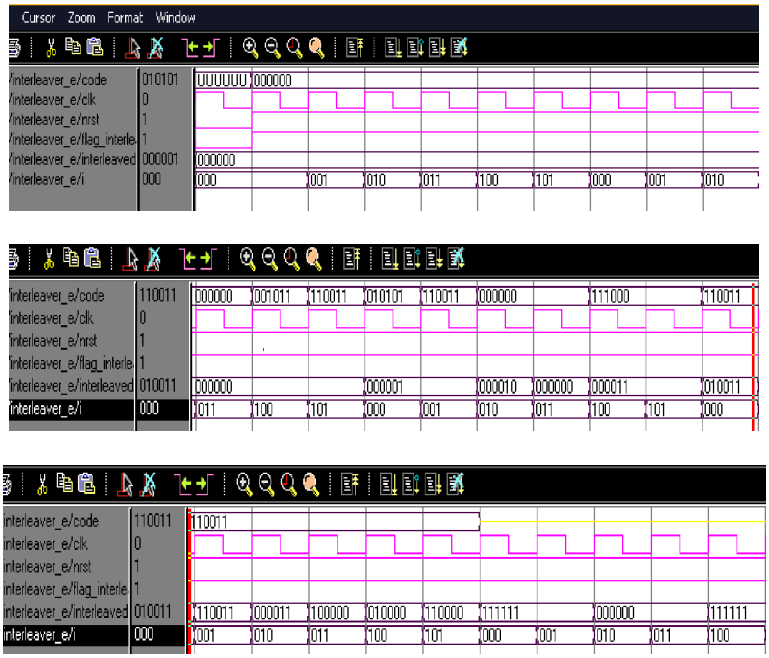


Figure B.2 Simulation of Interleaver

Convolution Encoder

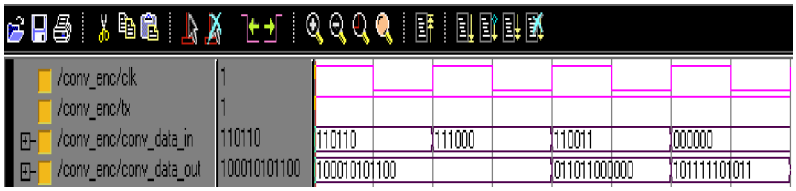
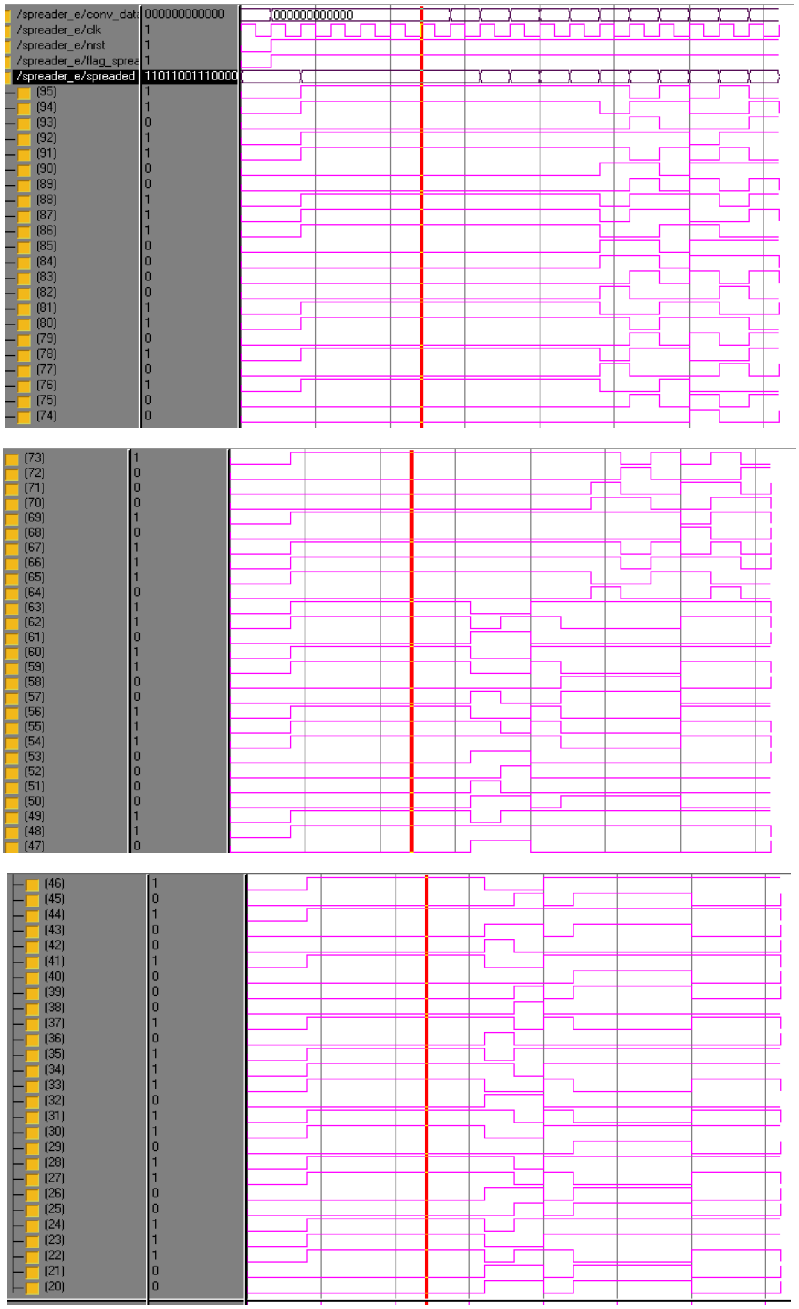


Figure B.3 Simulation of Convolution Encoder

Spreader



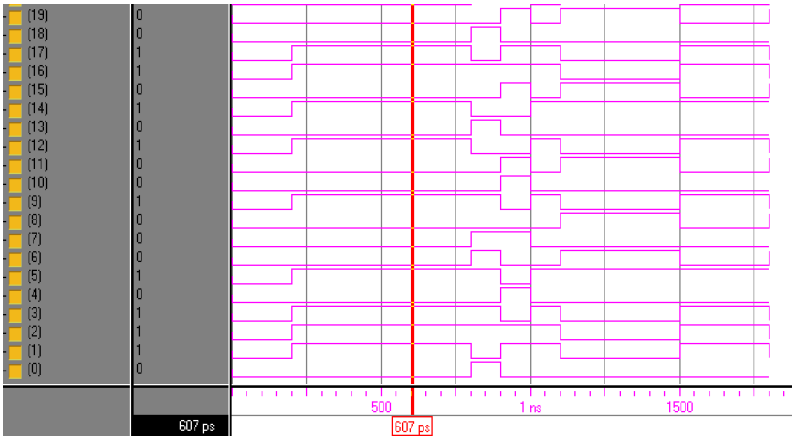


Figure B.4 Simulation of Spreader

Despreader

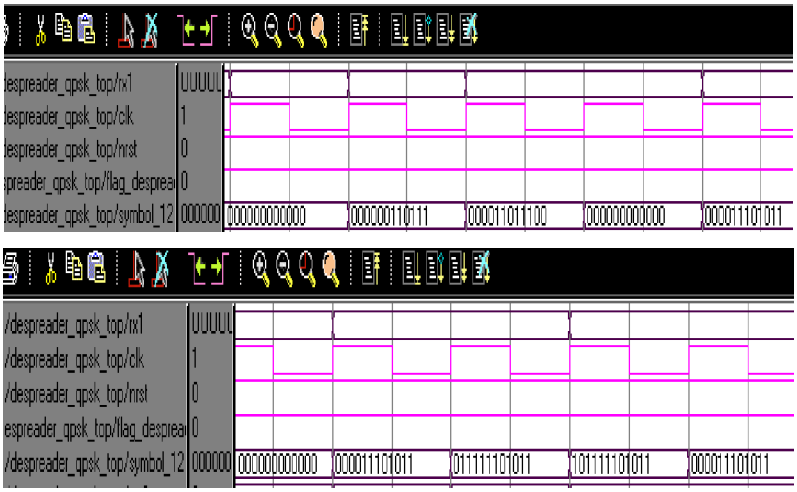


Figure B.5 Simulation of Despreader

Viterbi Decoder

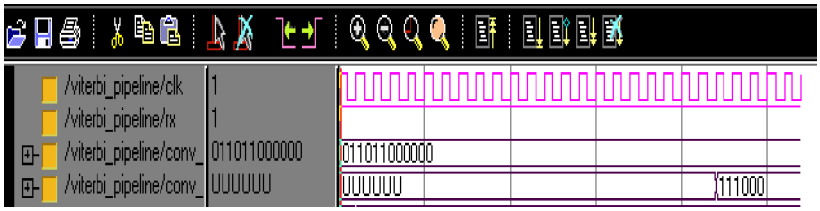


Figure B.6 Simulation of Viterbi Decoder

Deinterleaver

Figure B.7 Simulation of Deinterleaver

Block Decoder

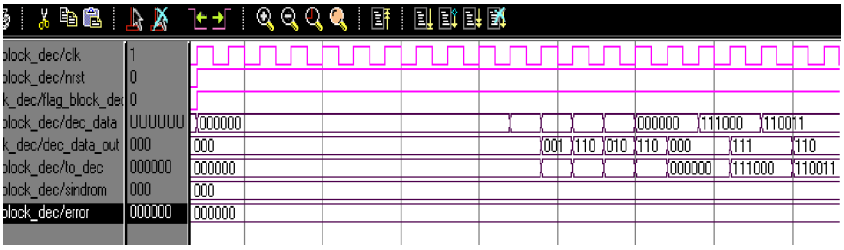


Figure B.8 Simulation of Block decoder

Appendix C Pin Interfaces

This Appendix presents the Pin Interface Schematic captured from the XST (Xilinx synthesis tool). Schematic of each and every component except buffers has been presented .

1. Transmitter

1.1 Block Encoder

Pin Interface:

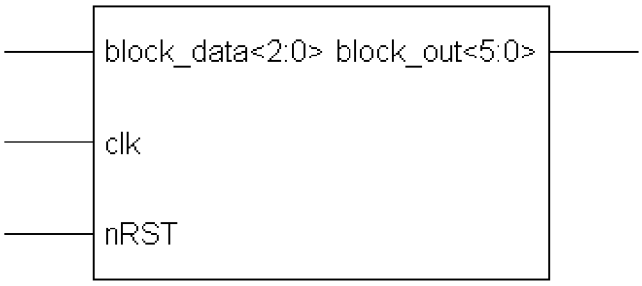


Figure C.1 Pin Interface of Block Encoder

1.2 Interleaver

Pin Interface:

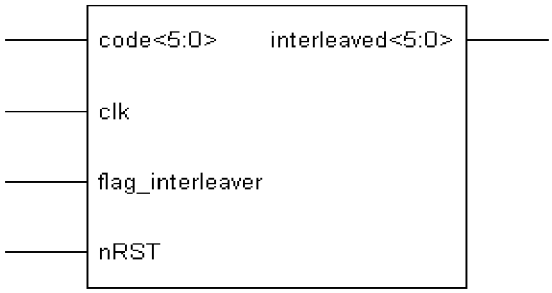


Figure C.2 Pin Interface of Interleaver

1.3 Convolution Encoder:

Pin Interface:

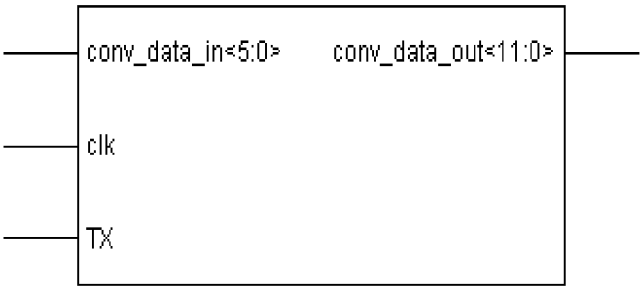


Figure C.3 Pin Interface of Convolution Encoder

1.4 Spreader :

Pin Interface:

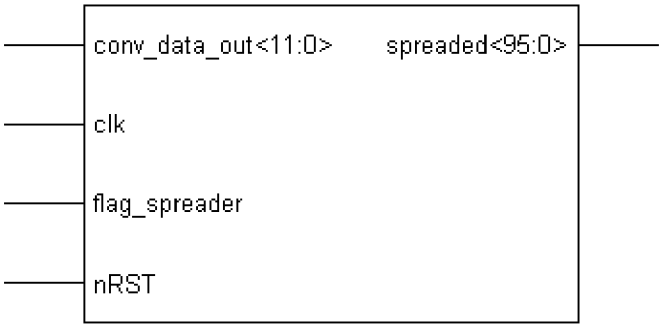


Figure C.4 Pin Interface of Spreader

1.5 Tx_counter:

Pin Interface:

Figure C.5 Pin Interface of TX_Counter

2. Receiver

2.1 Despreader

Pin Interface

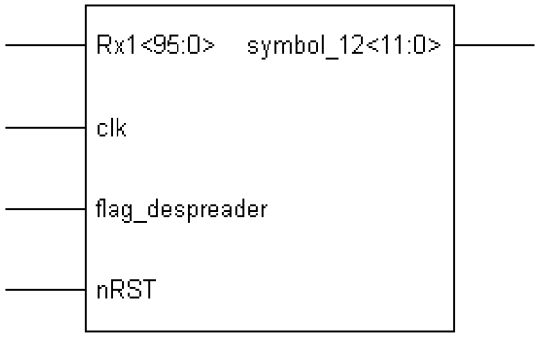


Figure C.6 Pin Interface of Despreader

2.2 Viterbi Decoder

Pin Interface:

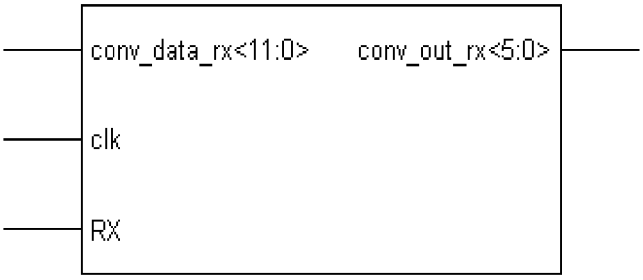


Figure C.7 Pin Interface of Viterbi Decoder

2.3 Deinterleaver

Pin Interface:

Figure C.8 Pin Interface of Deinterleaver

2.4 Block Decoder

Pin Interface:

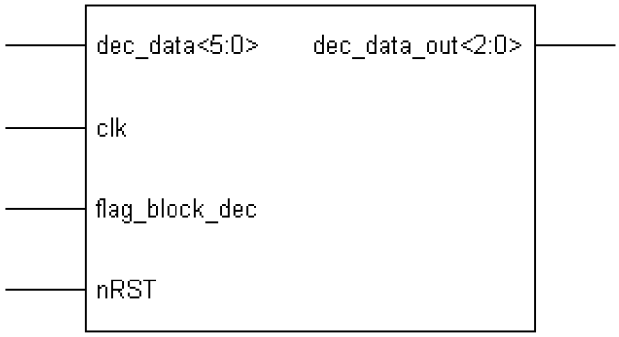


Figure C.9 Pin Interface of Block Decoder

2.5 Rx Counter

Pin Interface:

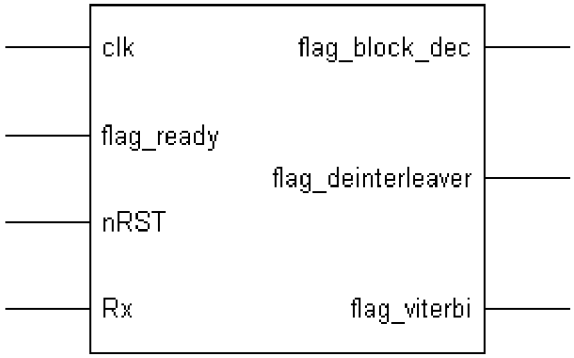


Figure C.10 Pin Interface of RX_counter

Appendix D Testing Waveforms

This appendix presents the testing waveforms captured from testing tool Chipscope's waveform viewer.

Given below is the strategy used for checking the functionality of each component.

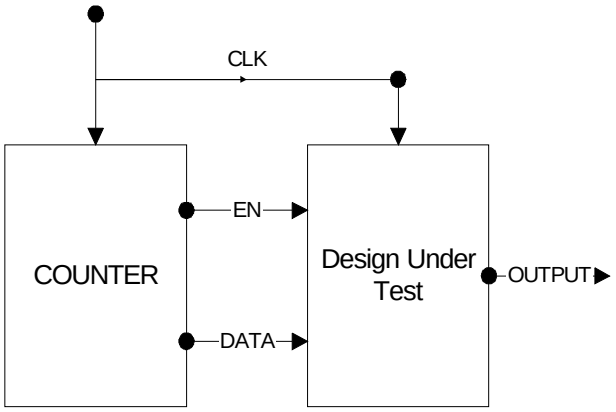


Figure D.1 Functionality Verification Strategy

Design Under Test is the component, which we have to verify its functionality. And 'COUNTER' is free running counter, which enables as well as gives stimulus to DUT on each clock rising edge.

Now below are snap shots, which proves that the system runs at real time. Here authors have divided this implementation in two parts. In first part transmitter is implemented and in second part receiver is implemented. To verify the whole design we connect transmitter and receiver in loop back fashion. So whatever the data is encoded in transmitter is inputted to receiver, so ultimately decoded data should be same as actual input data. So input to transmitter is in 3 bit form, '000', '000', '000', '000', '000', '000', '000', '000', '000', '001', '110', '010', '110', '000', '000', '111', '111', '110'. To verify the data of each block, reader can refer appendix A. Snap shots of some of the components are not mention here because that component's signals are not available in list of data ports.

Here below are the snapshots at transmitter side. First block is block encoder. First three MSB bits are same as input. So DataPort[87:89] are same as input.

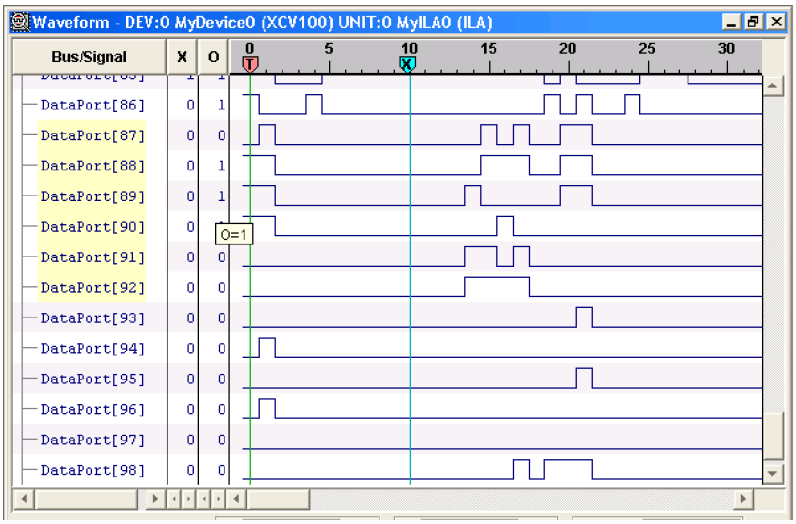


Figure D.2 Block Encoder Output

Here below is the interleaver snap shot.

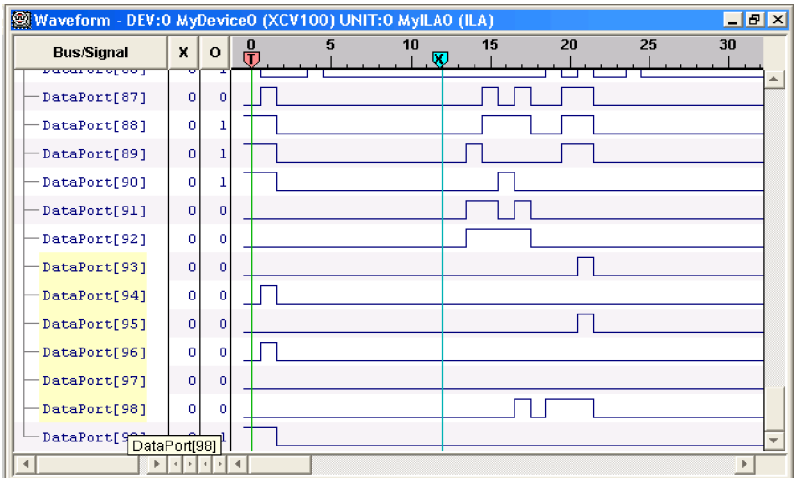


Figure D.3 Interleaver output

Below is the spreader snap shot.

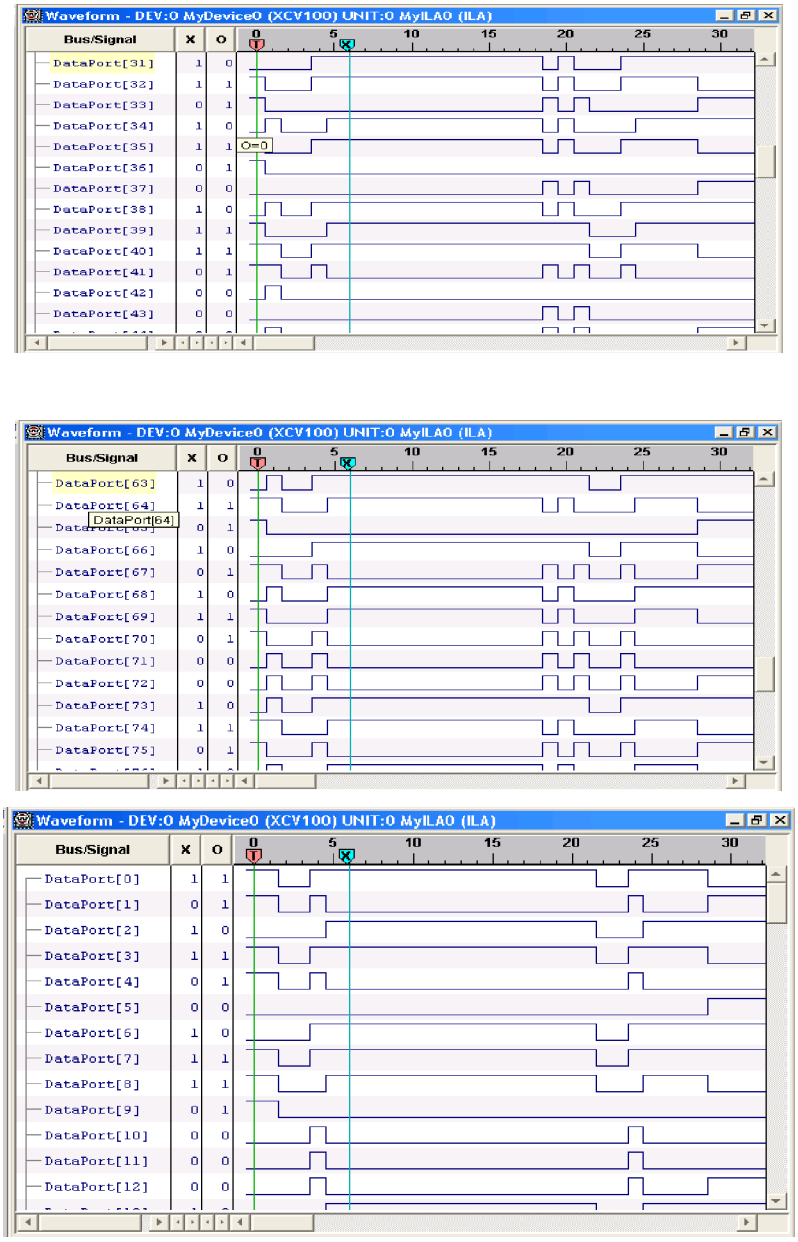


Figure D.4 Spreader Output

Below are the Viterbi decoder snapshots.

In this figure, reader can observe the data as in Appendix A.

First Figure of D.5 contains some of the decoded data and remaining are in second figure of D.5. Observe DataPort [0:5].

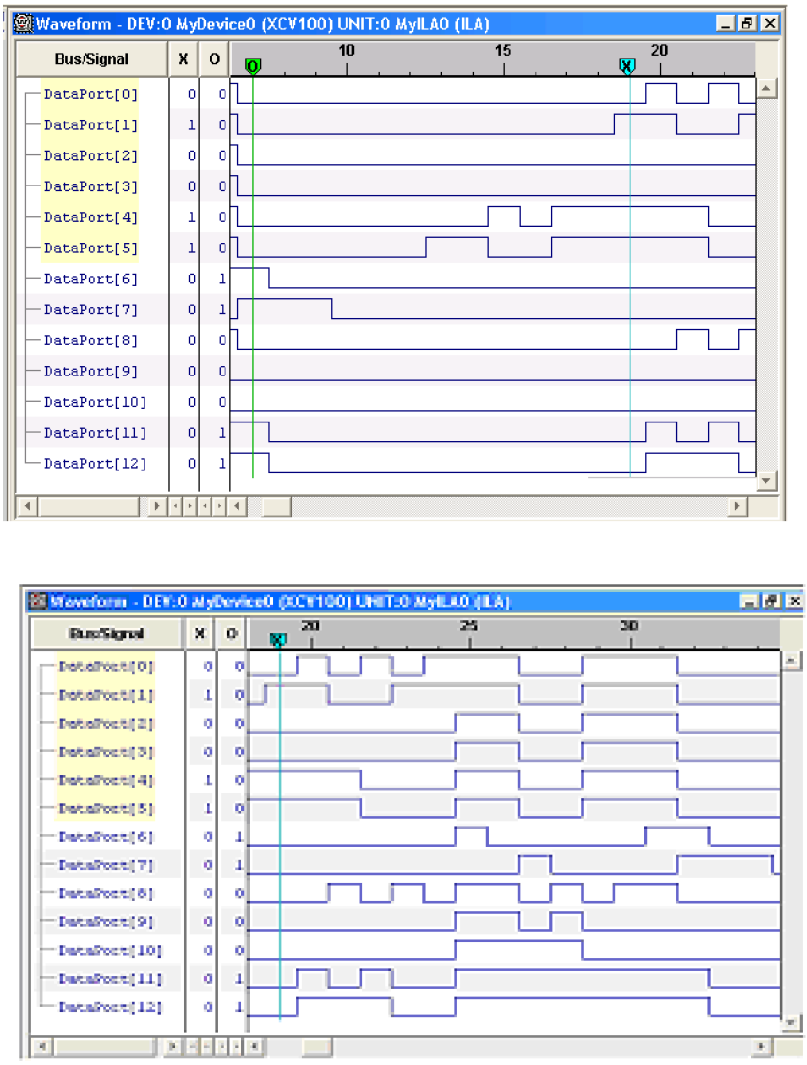


Figure D.5 Viterbi Decoder's Output

Deinterleaver

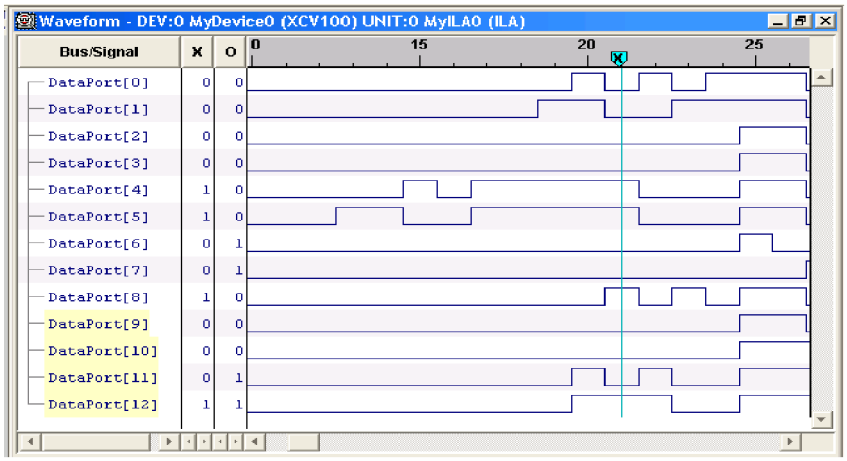


Figure D.6 Deinterleaver Output

Here below is the output of block decoder as well as final decoded data.

Here reader can observe that DataPort [0:2] is final decoded data.

Reader can verify this data with the actual input data at transmitter, which is same as input data.

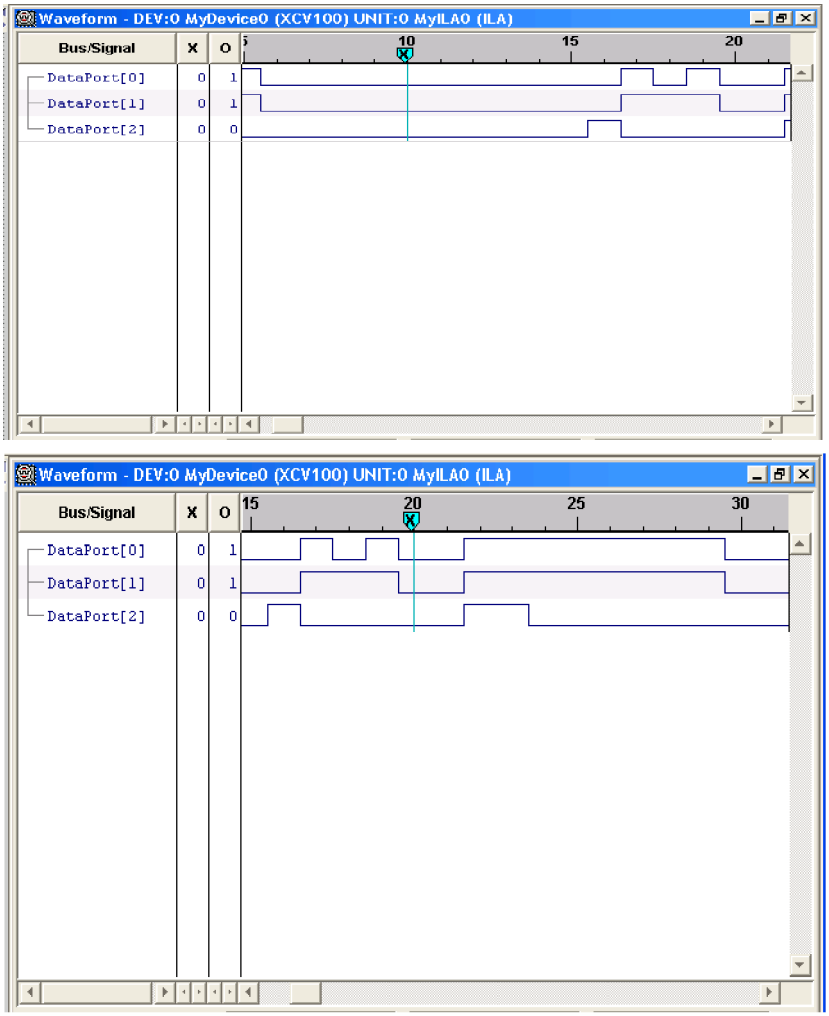


Figure D.7 Final Decoded Data