

OFDM over Optical Fiber Channel



By

Asmaa Ali Rayan

Alaa Amr Othman

Maha Emad Abd El-Maqsoud

Under the Supervision of

Dr. Hassan Mostafa

Dr. Tawfik Ismail

A Graduation Project Report Submitted to
the Faculty of Engineering at Cairo University
In Partial Fulfillment of the Requirements for the
Degree of
Bachelor of Science
in
Electronics and Communications Engineering

Faculty of Engineering, Cairo University

Giza, Egypt

July 2015

Contents

Table of Figures	VIII
Table of Tables.....	X
.....	2
Chapter 1: Introduction.....	2
1.1 Motivation & project definition	2
1.2 OFDM.....	3
1.2.1 Introduction.....	3
1.2.2. Brief History.....	4
1.2.1.1 OFDM Advantages.....	4
1.2.1.2 OFDM disadvantages.....	5
1.2.2 Orthogonality and OFDM	6
1.3 Block Diagram.....	6
Blocks in details:	7
1.3.1 Modulation and Demodulation of Data	7
1.3.2 Serial to Parallel Converter.....	8
1.3.3 IFFT/FFT	8
1.3.4 Guard Interval.....	8
1.3.5 Cyclic Prefix.....	9
Chapter2.1 Introduction to WI-FI.....	11
2.1.1 Introduction.....	11
2.1.2 Brief History.....	11
2.1.3 Frequency and Data Rates.....	11
2.1.4 Architecture of 802.11.....	12
2.1.5 Wireless LAN Components	12
2.1.6 Range	13
2.1.7 Benefits.....	14
Chapter3. System Explanation	16
3.1. Block Diagram.....	16
3.2. Transmitter Blocks Implementation.....	17
3.2.1. Scrambler.....	17
3.2.1.1 Function of Block	17

3.2.1.2 Structure.....	18
3.2.1.3 Description for VHDL Code flow chart.....	19
3.2.1.4 Post Layout Simulation.....	20
3.2.1.5 Performance measurement	20
3.2.2 Encoder.....	21
3.2.2.1 Introduction.....	21
3.2.2.2 Convolutional Codes.....	21
3.2.2.3 Puncturing Convolutional Codes	23
3.2.2.4 Function of the Block.....	25
3.2.2.5 Structure.....	25
3.2.2.6. The VHDL code description	26
3.2.2.7. Flow chart of the encoder	28
3.4.2.8. Post Layout Simulation.....	29
3.2.2.9 Performance measures	29
3.2.3 Interleaver	29
3.2.3.1. Function of block.....	29
3.2.3.3 Interleaver Implementation	31
3.2.3.5 Performance Measurement	32
3.2.4 Mapper	32
3.2.4.1. Function of the block.....	32
3.2.4.2 Block diagram	36
3.2.4.3 The VHDL Code description.....	37
3.2.4.4. The Calculated I and Q values in the mapper.....	38
3.2.4.5. Flow chart for the mapper	40
3.2.4.6 Post Layout Simulation.....	41
3.2.4.7 Performance Measurement	41
3.2.5. Framer	41
3.2.5.1. Function of the Block.....	41
3.2.5.3 VHDL Description.....	43
3.2.5.4. Post Layout Simulation.....	44
3.2.5.5. Performance measurement	44
3.2.6 FFT/IFFT	45

3.2.6.1 CFFT core	46
3.2.6.2 Description	46
3.2.6.3 Block diagram	47
3.2.6.4 Timing	47
3.2.6.5 Gain	48
3.2.6.6 Interfacing the CFFT core to the system	48
3.2.6.7 Structure:.....	49
3.2.6.8 Post Layout Simulation	50
3.2.6.9 Performance measurements:.....	51
3.7.2 Cyclic Extension	51
3.2.7.1 Function of Block	51
3.2.7.3 The VHDL code description	52
3.2.7.4. Flow chart of the cyclic extension block:.....	53
3.2.7.5 Post Layout Simulation	54
3.2.7.6 Performance measurement	54
3.2.8 Preamble	54
3.2.8.1 Function of the Block.....	54
3.2.8.2. Structure.....	55
3.2.8.3. Flow chart of the Preamble block	55
3.2.8.4 Post Layout Simulation	56
3.2.8.5 Performance measurement	56
3.3 Receiver Blocks Implementation	56
3.3.1 Packet divider	56
3.3.1.1 Structure.....	58
3.3.1.2 Operation:	59
Chapter4. Controller Implementation.....	61
4.1. Transmitter Controller Implementation.....	61
4.1.1 Introduction.....	61
4.1.2. Data Rates	61
4.1.3 Idea of Controller	61
4.1.4. Structure.....	62
4.1.5 VHDL Description using Flow Chart.....	63

4.1.6. Post layout simulation	64
4.2. Receiver Controller Implementation.....	65
4.2.2 Introduction.....	65
4.2.2 Structure.....	67
4.2.3 Post Layout Simulation	68
Chapter5. Synchronization	70
5.1 Timing Synchronization	70
5.1.1 Packet Detection	70
5.1.1.1 Double Sliding Window Packet Detection.....	71
5.1.1.2. Using the Structure of the Preamble for Packet Detection: (Delay and Correlate)	77
5.2 Residual Phase Error Correction Using CORDIC Algorithm	85
5.2.1 Introduction.....	85
5.2.2 Block Diagram for Our phase Correction Implementation	87
5.2.3 Flow Chart for Our Design Implementation	87
5.2.4 Simulation for the Phase Correction Implementation	89
5.2.5 Design performance	89
Chapter6. System Implementation	91
6.1 Design Flow using VHDL	91
6.1.1 System-level Verification.....	91
6.1.2 RTL design and test-bench creation	91
6.1.3 RTL verification (70% of design time at RTL)	91
6.1.4 Look-ahead Synthesis	92
6.2 Quartus II	92
6.2.1 Creating project:.....	93
6.2.2 Managing Timing Constraints:.....	94
6.2.3 Customizing Internal Memory IP Cores.....	96
6.3 Function simulation using ModelSim	98
1- Open the Design >> write in the Transcript Window these steps:	98
2-Create a Stimulus. First thing we need to do is to create a stimulus on our input >> right click on Add Wave >>click on Clock.	98
6.4 Verification on FPGA kit.....	99
6.6 Introduction to Nios Development Board and Stratix III FPGA:	102

Chapter7. Optical Fibers	105
7.1 introduction.....	105
7.2 Optical fiber advantages.....	105
7.3 Basic Fiber Optic Communication System.....	107
7.4 How Fiber Works	107
7.5 How to Choose Optical Fiber Single-Mode Fiber Performance Characteristics.....	109
7.6 Single-Mode Propagation.....	110
7.6.1 Single-Mode Characteristics.....	111
7.6.1.1 Mode Field Diameter (MFD) and Spot Size	111
7.6.1.2 Field Strength at the Fiber End.....	111
7.6.1.3 Bend Loss in Single-Mode Fiber	112
7.6.1.4 Cutoff Wavelength	112
Chapter8. OFDM Over Optical Access Network.....	114
8.1 Introduction.....	114
8.2 Light Fundamentals	115
8.3 Wave Division Multiplexing Technique (WDM)	116
8.4. Direct Detection OFDM	117
8.5. Effective Wavelengths In Optical Transmissions.....	118
8.6 Difference Between RF OFDM and OFDM Over Optical Channel	119
8.7. FUTURE WORK.....	119
8.7.1 Principle of Coherent Optical OFDM	119
8.8 difference between OFDM with optical fiber channel and CO-OFDM.....	120
Chapter9. Opti-System	123
9.1. Introduction.....	123
9.2 Simulation Main Blocks	124
9.2.1 User defined Bit Sequence Block:.....	124
9.2.2 MATLAB Block:	125
9.2.3 M-ary Pulse Generator:	125
9.2.4 Mach-Zender Modulator:.....	126
9.2.5 Optical Fiber:	127
9.2.6 Photo-Detector:.....	127
9.2.7 Low Pass Bessel Filter:	127

9.2.7 Electrical amplifiers:	127
9.2.9 DC Offset Block and Electrical Adder:.....	127
9.2.10 M-ary Threshold Detector:.....	128
9.3. Schematic	128
9.3.1 Back To Back testing.....	128
9.3.1.1. 1 st step: Testing the System back to back:	128
9.3.1.2. Results	128
9.3.2. 2 nd , Testing the system with 100 m fibers length.....	129
9.3.2.1. System Schematic.....	129
9.3.2.2. Results	130
Data sent from transmitter bock VS data received at the receiver block:.....	131
1-Opti-System Block visualizer at 30km 128 bit sequence length	XI
• Before channel vs before receiver	XI
Simulations at 80 db.....	XI
2-Opti-System Block visualizer at 90km 1024 bit sequence length	XI
3-Opti-System Block visualizer at 100km 1024 bit sequence length	XII
At 100 db gain.....	XIII
1-Opti-System Block visualizer at 100km 1024 bit sequence length	XIII
2-Opti-System Block visualizer at 140km 1024 bit sequence length	XIV
3-Opti-System Block visualizer at 150km 1024 bit sequence length	XV
4-Opti-System Block visualizer at 170km 1024 bit sequence length	XVI
Sample Opti-system Matlab code	XVII
References.....	XX

Table of Figures

Figure 1 OFDM block diagram	6
Figure 2 parallel to series converter.....	7
Figure 3 16-QAM Constellation	7
Figure 4 IFFT description	8
Figure 5 OFDM Frame with Guard Time.....	9
Figure 6 Inter-Carrier Interference (ICI)	9
Figure 7 Adding Cyclic Extension	9
Figure 8 Block diagram of OFDM.....	16
Figure 9 Block diagram of WIFI.....	17
Figure 10 Data scrambler	18
Figure 11	18
Figure 12.....	19
Figure 13.....	20
Figure 14.....	22
Figure 15 Puncturing patterns of IEEE.....	24
Figure 16 Convolutional encoder (K=7).....	25
Figure 17.....	25
Figure 18 An example of the bit stealing and bit insertion procedure ($r=3/4, 2/3$).....	27
Figure 19.....	28
Figure 20.....	29
Figure 21 Rate dependent parameters	31
Figure 22.....	31
Figure 23 BPSK, QPSK, 16-QAM, and 64-QAM constellation bit encoding	34
Figure 24.....	36
Figure 25.....	40
Figure 26.....	41
Figure 27 The location of the pilots in the frame.....	42
Figure 28.....	42
Figure 29 Inputs and outputs of the IFFT block in the 802.11a WLAN standard	43
Figure 30.....	44
Figure 31 conversion from frequency domain to time domain in transmitter	45
Figure 32 conversion from time domain to frequency domain in receiver	45
Figure 33 output of the two operations back to back.....	45
Figure 34 CFFT Block Diagram	47
Figure 35 Timing of input and output of CFFT core.....	47
Figure 36.....	48
Figure 37.....	49
Figure 38 post layout simulation of CFFT_Control component	50
Figure 39 Input Zoom in	50
Figure 40 Output Zoom in	50

Figure 41 Matlab output for the same input.....	51
Figure 42 OFDM symbol after cyclic extension	52
Figure 43.....	52
Figure 44.....	53
Figure 45.....	54
Figure 46.....	55
Figure 47.....	55
Figure 48.....	56
Figure 49 Formation of WiFi Packet	57
Figure 50 OFDM training structure	57
Figure 51.....	58
Figure 52.....	62
Figure 53.....	63
Figure 54.....	64
Figure 55.....	64
Figure 56.....	67
Figure 57 post layout simulation of receiver controller.....	68
Figure 58.....	68
Figure 59 The response of the double sliding window packet detection algorithm.....	71
Figure 60.....	72
Figure 61.....	74
Figure 62.....	75
Figure 63.....	77
Figure 64 Signal flow structure of the delay and correlate algorithm	78
Figure 65.....	79
Figure 66.....	80
Figure 67.....	81
Figure 68.....	82
Figure 69.....	83
Figure 70.....	84
Figure 71 OFDM receiver front end and synchronization architecture	86
Figure 72 Block Diagram for the phase correction Implementation using CORDIC.....	87
Figure 73.....	88
Figure 74.....	92
Figure 75.....	93
Figure 76.....	94
Figure 77 Warning of Constrain file.....	94
Figure 78 Compilation Report Window.....	95
Figure 79 Creating Clock.....	96
Figure 80.....	97
Figure 81.....	97
Figure 82.....	98

Figure 83.....	99
Figure 84 Stratix III kit.....	102
Figure 85 Data transmission in fibers.....	107
Figure 86 Total reflection in optical fibers.....	108
Figure 87 Attenuation in fibers.....	109
Figure 88 Signal amplitude as distance increases in fibers.....	109
Figure 89 Dispersion in fibers.....	110
Figure 90 Fiber operating modes.....	111
Figure 91 Single fiber bend loss.....	112
Figure 92 Optical region (RCDD).....	115
Figure 93 Light reflection and refraction (htt31).....	116
Figure 94 Wavelength division multiplexing technique.....	116
Figure 95 Power distribution in WDM.....	117
Figure 96 Wave lengths in optical transmission.....	118
Figure 97 up-converter/down-converter.....	120
Figure 98 User defined binary sequence block configuration.....	124
Figure 99 MATLAB block configuration.....	125
Figure 100. Mach-Zehnder modulator.....	126
Figure 101 Back to black testing.....	128
Figure 102 Back to back BER results.....	129
Figure 103 100 m test- Transmitter blocks.....	129
Figure 104 100m test-receiver blocks.....	130
Figure 105 100m BER results.....	130
Figure 106 Data sent before channel and after.....	131
Figure 107.....	XI
Figure 108.....	XII
Figure 109.....	XIII
Figure 110.....	XIV
Figure 111.....	XV
Figure 112.....	XVI
Figure 113.....	XVII

Table of Tables

Table 3. 1.....	24
Table 3. 2 Modulation-dependent normalization factor KMOD.....	33
Table 3. 3 BPSK encoding table.....	35
Table 3. 4 QPSK encoding table 5 QPSK encoding table.....	35
Table 3. 5 16-QAM encoding table.....	35
Table 3. 6 64-QAM encoding table.....	36

Table 4. 1 61

Table 5. 1 illustration of the probability of correct detection for different 76

Table6. 1 100

CHAPTER 1

Chapter 1: Introduction

1.1 Motivation & project definition

OFDM over optical access network, OFDM implemented on FPGA over optical access network rather than wireless network. As The combination of radio over fiber (RoF) and orthogonal frequency division multiplexing (OFDM) techniques has resulted in a high-data-rate at lower cost in the last mile of wireless networks.

The purpose of implementing this system is to increase data rate as the wireless system OFDM increased the data rate to the range of 1Gbps, and as the demand for high speed data rate and high capacity of bandwidth has increased due to recent advances in technology in the access networks bandwidth, so a new way of thinking started. These trials aims to increase the data rate in the range of tens or hundreds of Gbps for data transmission, regardless the transmission technique used with the best modulation scheme to achieve the highest possible signal to noise ratio (SNR) and a higher accuracy.

Increasing data rate to higher range than the range of 4G communication systems which operating on the range of 1G, makes it possible to reach to the fifth generation of communication systems 5G .

The integration of fiber optics and wireless communication systems has many advantages. Such as, increasing data rate and lower power consumption with higher bandwidth than that achieved with wireless communications. Also, it helps to send high data rates to long distances without the need to add repeaters or to regenerate the signal (Fuentes, 2005-03-22).

1.2 OFDM

Orthogonal Frequency Division Multiplexing (OFDM) is a multi-carrier modulation technique which is common used nowadays in wireless network such as 4G mobile communications, it's also used for digital television and audio broadcasting. Long distance communication is improved using OFDM, cause of its ability to cope with severe channel conditions without complex equalization filters and its ability to eliminate the inter-symbol interference (ISI).

1.2.1 Introduction

Orthogonal frequency-division multiplexing (OFDM) is a method of digital modulation in which a signal is split into several narrowband channels at different frequencies. Recently, a worldwide convergence has occurred for the use of Orthogonal Frequency Division Multiplexing (OFDM) in wireless communication applications as an emerging technology for high data rates.

Wireless communication is having the fastest growth phase because of unprecedented evolution in the field. Orthogonal frequency-division multiplexing (OFDM) is a method of digital modulation in which a signal is split into several narrowband channels at different frequencies. OFDM has been adopted by several technologies such as Asymmetric Digital Subscriber Line (ADSL) services, IEEE 802.11a/g, IEEE 802.16a, Digital Audio Broadcast (DAB), and digital terrestrial television broadcast: DVD in Europe, ISDB in Japan 4G, IEEE 802.11n, IEEE 802.16, and IEEE 802.20.

The principle of work of OFDM that it converts a frequency-selective channel into a parallel collection of frequency flat sub channels which makes it robust against large delay spreads. OFDM provides many advantages over this conventional technique. The subcarrier frequencies of OFDM are chosen so that the signals are mathematically orthogonal over one symbol period. So the effect of inter-channel interference is eliminated.

1.2.2. Brief History

The IEEE initiated the 802.11 project in 1990 with a scope “to develop a Medium Access Control (MAC) and Physical Layer (PHY) specification for wireless connectivity for fixed, portable, and moving stations within an area.”

In 1997, IEEE first approved the 802.11 international interoperability standard. In 1999, the IEEE ratified the 802.11a and the 802.11b wireless networking communication standards. The goal was to create a standards-based technology that could span multiple physical encoding types, frequencies, and applications. The 802.11a standard uses orthogonal frequency division multiplexing (OFDM) to reduce interference. This technology uses the 5 GHz frequency spectrum and can process data at up to 54 Mbps.

1.2.1.1 OFDM Advantages

Because using wireless environment fading usually impairs signals and multipath delay .In such wireless channels extreme fading of the signal amplitude occurs and Inter Symbol Interference (ISI), multipath effects due to the frequency selectivity of the channel appears at the receiver side, a considerable delay and losses will lead to a high probability of errors so the system’s overall performance becomes very poor.

OFDM advantages are its robustness against channel dispersion and the ease of phase and channel estimation in a time-varying environment. One of the major strengths of the orthogonal frequency-division multiplexing (OFDM) modulation format is its rich variation and ease of adaption to a wide range of applications. OFDM has although the ability to control the signal as hard as it is possible. As it;

- Makes efficient use of the spectrum by allowing overlap.
- By dividing the channel into narrowband flat fading sub-channels, OFDM is more resistant to frequency selective fading than single carrier systems are.

- Eliminates ISI and IFI through use of a cyclic prefix.
- Using adequate channel coding and interleaving one can recover symbols lost due to the frequency selectivity of the channel.
- Channel equalization becomes simpler than by using adaptive equalization techniques with single carrier systems.
- It is possible to use maximum likelihood decoding with reasonable complexity.
- OFDM is computationally efficient by using FFT techniques to implement the modulation and demodulation functions.
- Is less sensitive to sample timing offsets than single carrier systems are.
- Provides good protection against channel interference and impulsive parasitic noise.

1.2.1.2 OFDM disadvantages

The essential disadvantages of OFDM are its high peak-to-average power ratio (PAPR) and sensitivity to frequency and phase noise, the main reason of system sensitivity as it requires high frequency and time synchronization accuracy between transmitter and receiver. OFDM is very sensitive for the error of frequency and time synchronization, which means frequency and time offset.

- The OFDM signal has a noise like amplitude with a very large dynamic range, therefore it requires RF power amplifiers with a high peak to average power ratio.
- It is more sensitive to carrier frequency offset and drift than single carrier systems are due to leakage of the DFT.

1.2.2 Orthogonality and OFDM

As the orthogonality is the principle of work of OFDM, it is important to define the orthogonality concept used in OFDM. If the FDM system has the ability to use a set of subcarriers that are orthogonal to each other, a higher level of spectral efficiency could have been achieved using FDM (frequency division multiplexing). The use of orthogonal subcarriers would allow the subcarriers' spectra to overlap, thus increasing the spectral efficiency. Orthogonality means: If two random processes are uncorrelated, then they are orthogonal. The transmitted signal at the transmitter is divided into several signals to be carried on OFDM sub-carriers which are mathematically orthogonal. Then at the receiver side, the signals from the subcarriers are then combined to form an estimate of the source signal from the transmitter. The orthogonal and uncorrelated nature of the subcarriers is exploited in OFDM with powerful results.

1.3 Block Diagram

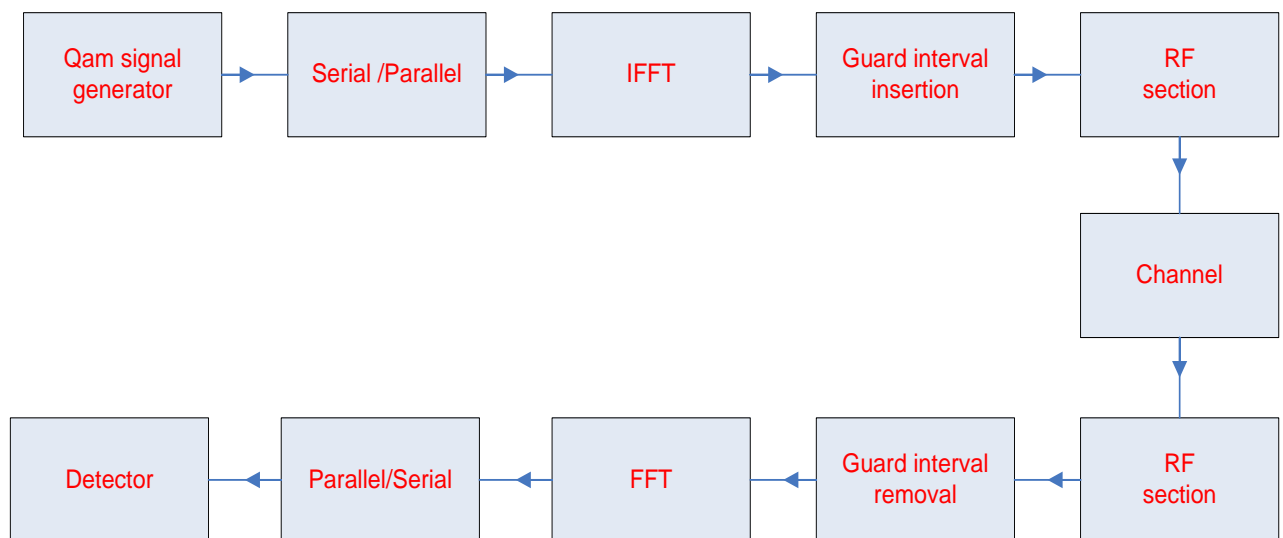


Figure 1 OFDM block diagram

Blocks in details:

1.3.1 Modulation and Demodulation of Data

Modulation is a process of mixing a signal which contains information data with a sinusoid to produce a new signal. This new signal, will have certain benefits over an un-modulated signal. The process by which information signals, analog or digital, are transformed into waveforms suitable for transmission across the channel.

After bit interleaving, the data bits are entered serially to the constellation mapper. The data bits shall be modulated by using 16-QAM modulation; the encoded and interleaved binary serial input data shall be divided into groups of NBPSK (4) bits and converted into complex numbers representing 16-QAM constellation points.

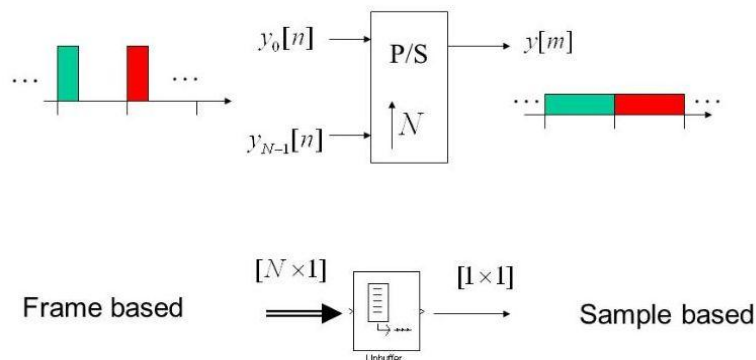


Figure 2 parallel to series converter

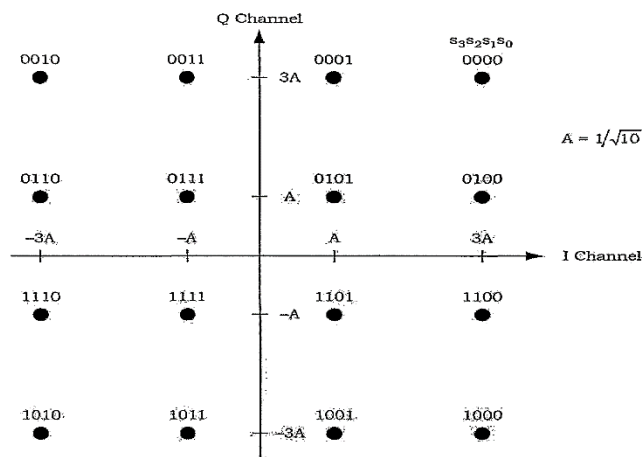


Figure 3 16-QAM Constellation

1.3.2 Serial to Parallel Converter

The input serial data stream is formatted into the word size required for transmission, e.g. 4 bits/word for QAM, and shifted into a parallel format. The data is then transmitted in parallel by corresponding each data word to one carrier in the transmission.

1.3.3 IFFT/FFT

Since the OFDM signal is in the time domain, so IFFT is used in the transmitter, which can convert frequency domain samples to time domain samples. The IFFT is useful for OFDM because it generates samples of a waveform with frequency components satisfying orthogonality conditions. Then, the parallel to serial block creates the OFDM signal by sequentially outputting the time domain samples. This operation ensures that sub-carriers do not interfere with one another.

Receiver performs the inverse of the transmitter.

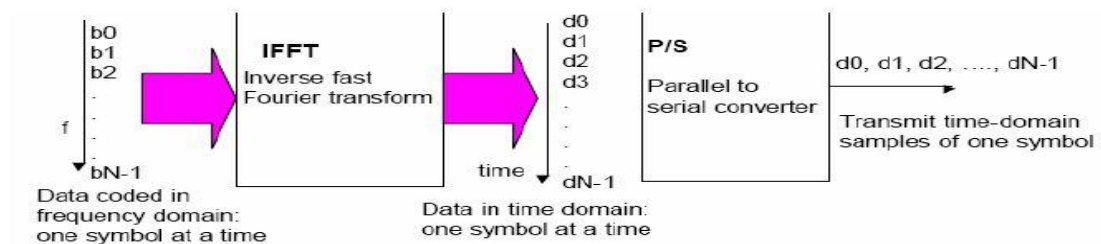


Figure 4 IFFT description

1.3.4 Guard Interval

The guard interval is used to eliminate inter-symbol and inter-carrier interference (ISI) completely.

Guard time is chosen larger than the expected delay spread such that the multipath component for one symbol can't interfere with the next symbol.

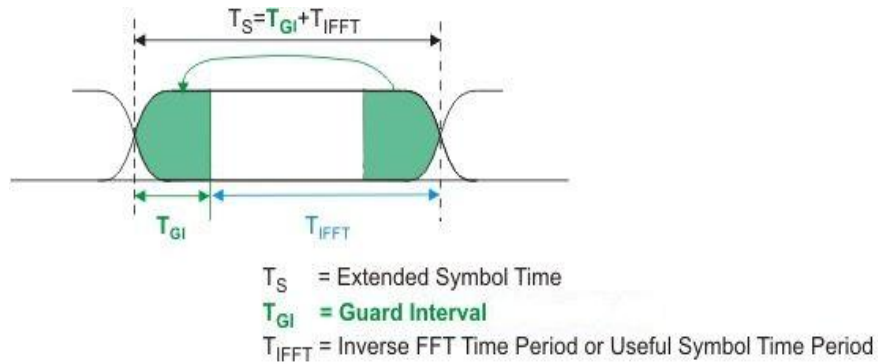


Figure 5 OFDM Frame with Guard Time

1.3.5 Cyclic Prefix

To eliminate ICI (Inter-Carrier Interference) which is the cross talk between different subcarriers which means they are no longer orthogonal.

OFDM symbols are cyclically extended in the guard time. This ensures that delayed replicas of the OFDM symbol always have an integer number of cycles within the FFT interval.

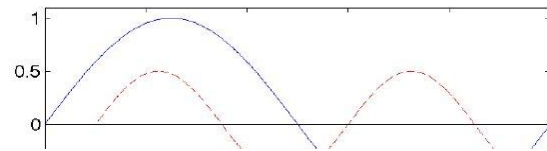


Figure 6 Inter-Carrier Interference (ICI)

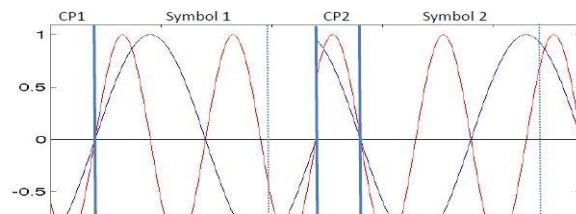


Figure 7 Adding Cyclic Extension

Chapter 2

CHAPTER 2

Chapter 2.1 Introduction to Wi-Fi

2.1.1 Introduction

WLAN technology and the WLAN industry date back to the mid-1980s when the Federal Communications Commission (FCC) first made the RF spectrum available to industry. During the 1980s and early 1990s, growth was relatively slow. Today, however, WLAN technology is experiencing tremendous growth. The key reason for this growth is the increased bandwidth made possible by the IEEE 802.11 standard. A brief introduction to the 802.11 and WLAN technology follows.

2.1.2 Brief History

The IEEE initiated the 802.11 project in 1990 with a scope “to develop a Medium Access Control (MAC) and Physical Layer (PHY) specification for wireless connectivity for fixed, portable, and moving stations within an area.” In 1997, IEEE first approved the 802.11 international interoperability standard. In 1999, the IEEE ratified the 802.11a and the 802.11b wireless networking communication standards. The goal was to create a standards-based technology that could span multiple physical encoding types, frequencies, and applications. The 802.11a standard uses orthogonal frequency division multiplexing (OFDM) to reduce interference. This technology uses the 5 GHz frequency spectrum and can process data at up to 54 Mbps.

2.1.3 Frequency and Data Rates

IEEE developed the 802.11 standards to provide wireless networking technology like the wired Ethernet that has been available for many years. The IEEE 802.11a standard is the most widely adopted member of the 802.11 WLAN family. It operates in the licensed 5 GHz band using OFDM technology. The popular 802.11b standard operates in the unlicensed 2.4 GHz-2.5 GHz Industrial, Scientific, and Medical (ISM) frequency band using a direct sequence spread-spectrum technology. The ISM band has become popular for wireless communications because it is available worldwide. The

802.11b WLAN technology permits transmission speeds of up to 11 Mbits/second.

2.1.4 Architecture of 802.11

The IEEE 802.11 standard permits devices to establish either peer-to-peer (P2P) networks or networks based on fixed access points (AP) with which mobile nodes can communicate. Hence, the standard defines two basic network topologies: the infrastructure network and the ad hoc network. The infrastructure network is meant to extend the range of the wired LAN to wireless cells. A laptop or mobile device may move from cell to cell (from AP to AP) while maintaining access to the resources of the LAN. A cell is the area covered by an AP and is called a “basic service set” (BSS). The collection of all cells of an infrastructure network is called an extended service set (ESS). This first topology is useful for providing wireless coverage of large areas. By deploying multiple APs with overlapping coverage areas, organizations can achieve broad network coverage. WLAN technology can be used to replace wired LANs totally and to extend LAN infrastructure. A WLAN environment has wireless client stations that use radio modems to communicate to an AP.

Client stations are generally equipped with a wireless network interface card (NIC) that consists of the radio transceiver and the logic to interact with the client machine and software. An AP comprises essentially a radio transceiver on one side and a bridge to the wired backbone on the other. The AP, a stationary device that is part of the wired infrastructure, is analogous to a cell-site (base station) in cellular communications. All communications between the client stations and between clients and the wired network go through the AP.

2.1.5 Wireless LAN Components

A WLAN comprises two types of equipment: a wireless station and an access point. A station, or client, is typically a laptop or notebook personal computer (PC) with a wireless NIC. A WLAN client may also be a desktop or handheld device, or equipment within a kiosk on a manufacturing floor or

other publicly accessed area. Wireless laptops and notebooks - “wireless enabled” - are identical to laptops and notebooks except that they use wireless NICs to connect to access points in the network. The wireless NIC is commonly inserted in the client's Personal Computer Memory Card International Association (PCMCIA) slot or Universal Serial Bus (USB) port. The NICs use radio signals to establish connections to the WLAN. The AP, which acts as a bridge between the wireless and wired networks, typically comprises a radio, a wired network interface such as 802.3, and bridging software. The AP functions as a base station for the wireless network, aggregating multiple wireless stations onto the wired network.

2.1.6 Range

The reliable coverage range for 802.11 WLANs depends on several factors, including data rate required and capacity, sources of RF interference, physical area and characteristics, power, connectivity, and antenna usage. Theoretical ranges are from 29 meters (for 11 Mbps) in a closed office area to 485 meters (for 1 Mbps) in an open area. However, through empirical analysis, the typical range for connectivity of 802.11 equipment is approximately 100 meters (about 320 ft.) indoors. Special high-gain antennas can increase the range to several miles make WLAN the ideal technology for many applications. APs may also provide a “bridging” function. Bridging connects two or more networks together and allows them to communicate - to exchange network traffic. Bridging involves either a point-to-point or a multipoint configuration. In a point-to-point architecture, two LANs are connected to each other via the LANs’ respective APs. In multipoint bridging, one subnet on a LAN is connected to several other subnets on another LAN via each subnet AP.

For example, if a computer on Subnet A needed to connect to computers on Subnets B, C, and D, Subnet A’s AP would connect to B’s, C’s, and D’s respective APs. Enterprises may use bridging to connect LANs between different buildings on corporate campuses. Bridging AP devices are typically placed on top of buildings to achieve greater antenna

reception. The typical distance over which one AP can be connected wirelessly to another by means of bridging is approximately 2 miles. This distance may vary depending on several factors including the specific receiver or transceiver being used

2.1.7 Benefits

User Mobility: Users can access files, network resources, and the Internet without having to physically connect to the network with wires. Users can be mobile yet retain high-speed, real time access to the enterprise LAN.

Rapid Installation: The time required for installation is reduced because network connections can be made without moving or adding wires, or pulling them through walls or ceilings, or making modifications to the infrastructure cable plant.

Flexibility: Organizations can enjoy the flexibility of installing and taking down WLANs in locations as necessary. Users can quickly install a small WLAN for temporary needs such as a conference, trade show, or standards meeting.

Scalability: WLAN network topologies can easily be configured to meet specific application and installation needs and to scale from small peer-to-peer networks to very large enterprise networks that enable roaming over a broad area.

Because of these fundamental benefits, the WLAN market has been increasing steadily over the past several years, and WLANs are still gaining in popularity. WLANs are now becoming a viable alternative to traditional wired solutions.

Chapter 3

CHAPTER 3

Chapter3. System Explanation

3.1. Block Diagram

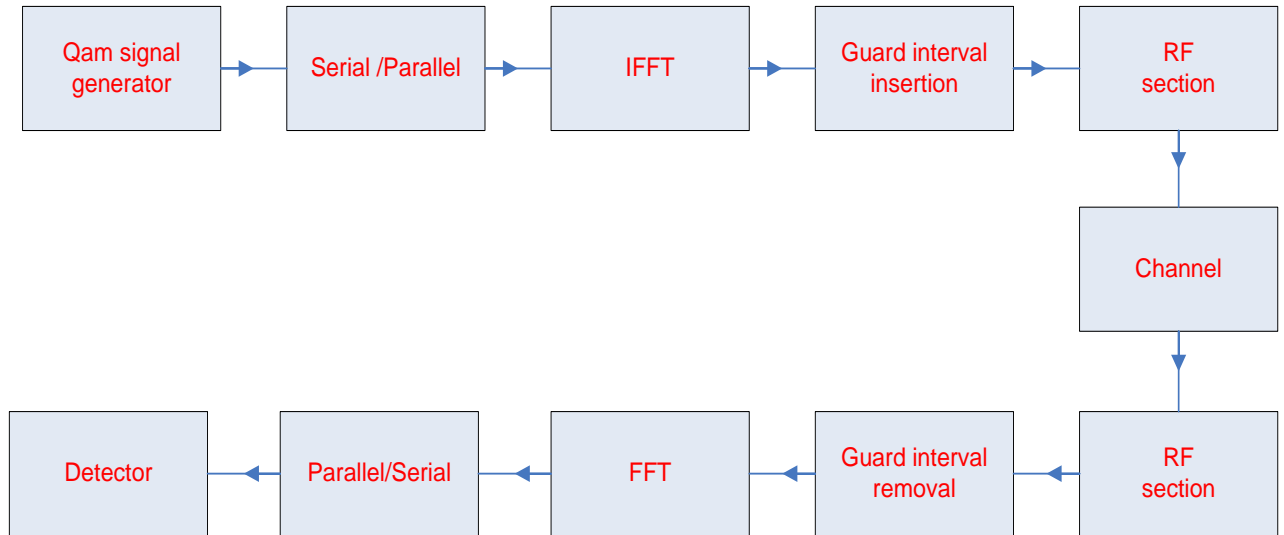


Figure 8 Block diagram of OFDM

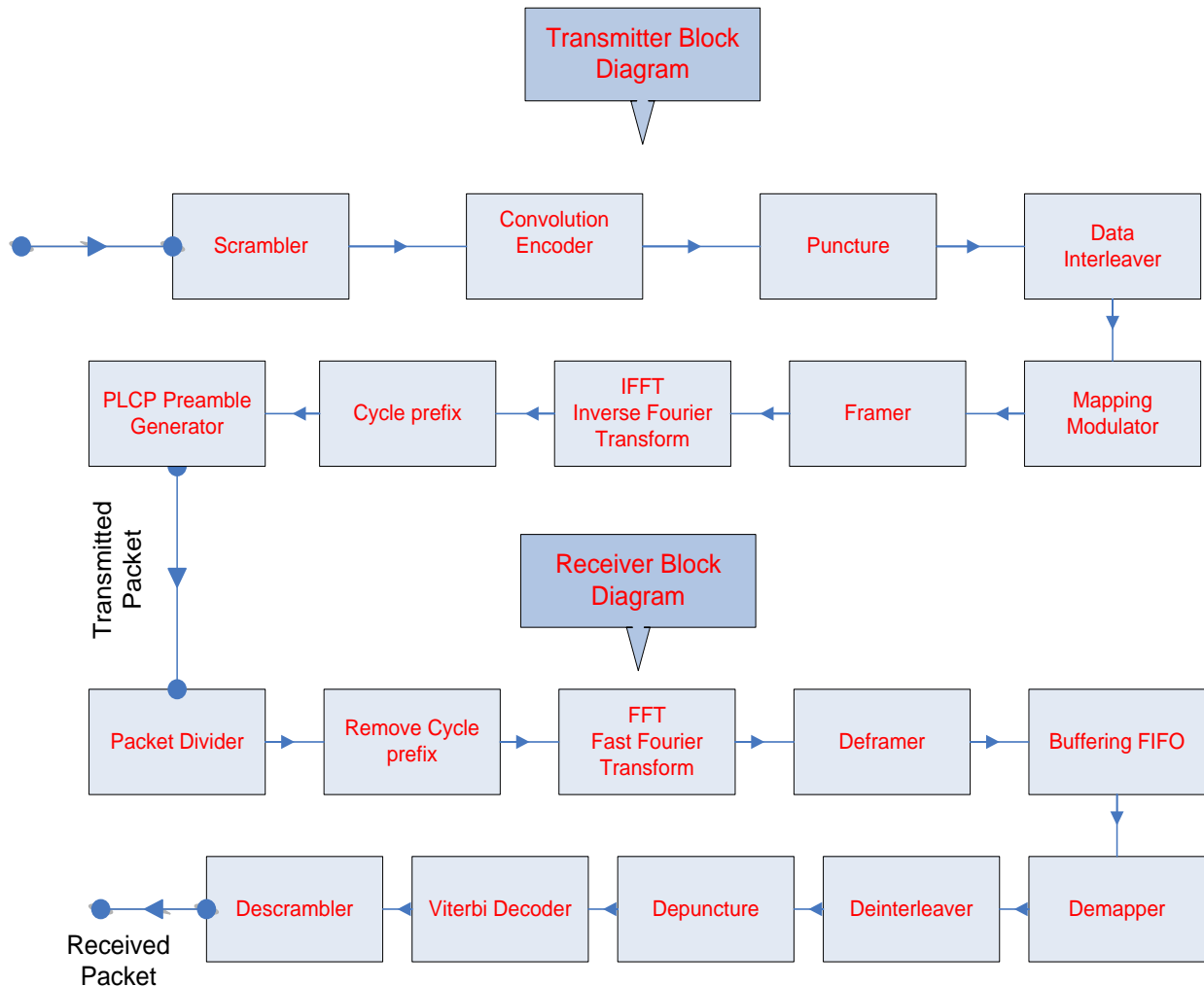


Figure 9 Block diagram of WIFI

3.2. Transmitter Blocks Implementation

All the transmitter and receiver block are synthesized and tested (post layout simulation) on STRATIX II EP2S60 (class, 2013).

3.2.1. Scrambler

3.2.1.1 Function of Block

It is used to scramble the transmitted data , by using a generator polynomial $S(x)$ where:

$$S(x) = x^7 + x^4 + 1$$

Which will generate the 127-bit sequence generated repeatedly by the scrambler shall be (leftmost used first), 00001110 11110010 11001001 00000010 00100110 00101110 10110110 00001100 11010100 11100111 10110100 00101010 11111010 01010001 10111000 11111111, when the “all ones” initial state is used.

And this sequence is XOR with the transmitted data bit as seen in figure 10.

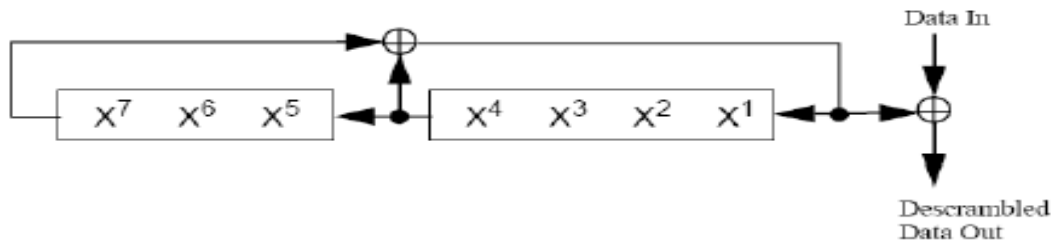


Figure 10 Data scrambler

3.2.1.2 Structure

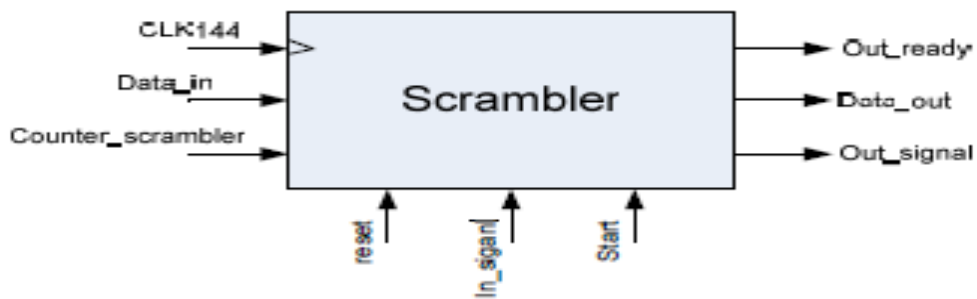


Figure 11

3.2.1.3 Description for VHDL Code flow chart

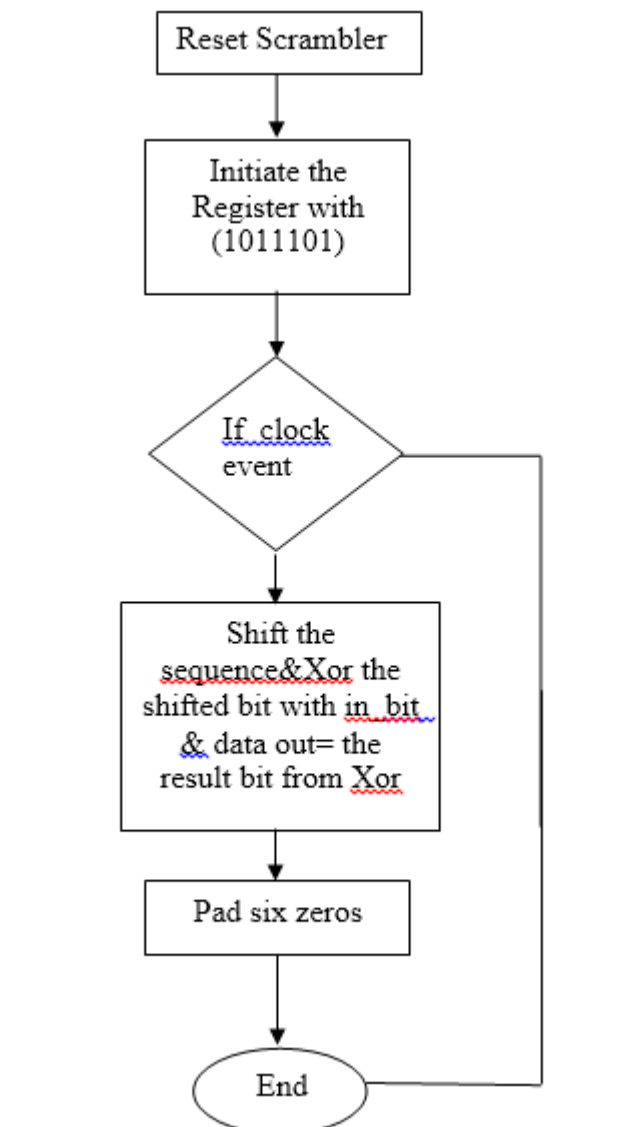


Figure 12

Padding

According to the standard there are six zeros follow the data, those six scrambled "zero" bits following the "data" are replaced with six non-scrambled "zero" bits. Those bits return the convolutional encoder to the "zero state" and are denoted as "tail bits". The PPDU tail bit field shall be six bits of "0," which are required to return the convolutional encoder to the "zero state." This procedure improves the error probability of the convolutional decoder, which relies on future bits when decoding and

which may be not be available past the end of the message. The PLCP tail bit field shall be produced by replacing six scrambled “zero” bits following the message end with six non-scrambled “zero” bits.

3.2.1.4 Post Layout Simulation

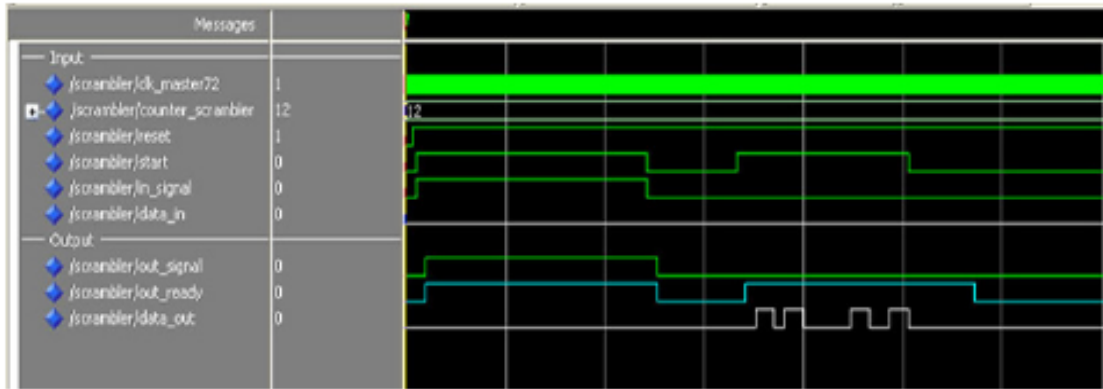


Figure 13

3.2.1.5 Performance measurement

Logic utilization	< 1%
Combinational ALUTs	112 / 48,352 (< 1 %)
Dedicated logic registers	31 / 48,352 (< 1 %)
Total registers	31
Total block memory bits	0 / 2,544,192 (0 %)
Max freq	250.69Mhz

3.2.2 Encoder

3.2.2.1 Introduction

Channel codes are a very important component of any modern digital communication system, and they make today's effective and reliable wireless communications possible.

The basic measure of channel coding performance is coding gain, which is usually measured in dBs as the reduction of required to achieve a certain bit error rate in AWGN channel. Hence, coding gain is the measure in the difference between the signal to noise ratio (SNR) levels between the uncoded system and coded system required to reach the same bit error rate (BER) levels when used with the error correcting code (ECC).

3.2.2.2 Convolutional Codes

Convolutional codes are one of the mostly widely used channel codes in today's systems; all the major cellular systems (GSM, IS-95) in use today use convolutional channel codes. IEEE 802.11a and HiperLAN/2 WLAN standards also use convolutional error correcting code, and IEEE 802.11b includes an optional mode that uses them. Convolutional codes owe their popularity to good performance and flexibility to achieve different coding rates.

A convolutional code is defined by a set of connections between stages of one or more shift registers and the output bits of the encoder. The number of shift registers k is equal to the nominator of the code rate and the number of output bits n is equal to the denominator. The rate of a convolutional code is generally expressed as k/n . The number of connections required to describe a code is equal to the product of k and n . For each output bit there are k connections that define how the value of the output bit is calculated from the state of the shift registers. For example, Figure (14) shows the convolutional encoder used in IEEE 802.11a. This is a rate code $1/2$ with connections 1338 and 1718. The connections are defined as octal numbers, the binary representations are 001 011 0112 and 001 111 0012. The octal notation is used to shorten the expressions,

when connections for different convolutional codes are tabulated. From the binary notation, the structure of the encoder is easily constructed. The connections are aligned to the end of the shift register, and a value of 1 means that the shift register stage output is connected to one of the output bits of the encoder using a binary XOR operation. In Figure (14) the connection 1338 that defines values the even indexed bits b_{2n} and the connection 1718 defines the values of the odd indexed bits b_{2n+1} . Figure (14) state rate 1/2 convolutional encoder used in IEEE 802.11a.

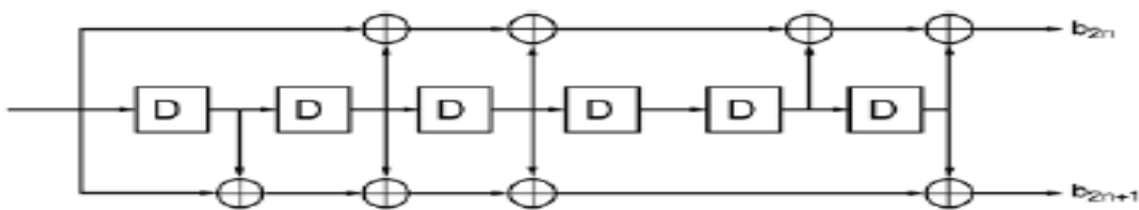


Figure 14

The number of shift register elements determines how large a coding gain the convolutional code can achieve. The longer the shift registers, the more powerful the code is; unfortunately, the decoding complexity of the maximum likelihood Viterbi algorithm grows exponentially with the number of shift register elements. This complexity growth limits the currently used convolutional codes to eight shift register elements, and IEEE 802.11a uses only six, due to its very high speed data rates. The performance of a convolutional code is determined by the minimum free distance of the code. Free distance is defined using the Hamming distance that is equal to the number of position in which two code words are different. Free distance of a convolutional code is the minimum Hamming distance between two different code words. Before we can calculate the free distance of our encoder we may see first the different representations of the convolutional encoder.

3.2.2.3 Puncturing Convolutional Codes

Usually communications systems provide a set of possible data rates; for example, IEEE 802.11a has eight different data rates: 6, 9, 12, 18, 24, 36, 48, and 54 Mbits/s. Now if the system could only change the data rate by adjusting the constellation size, and not the code rate, a very large number of different rates would be difficult to achieve as the number of constellations and the number of points in the largest constellation would grow very quickly. Another solution would be to implement several different convolutional encoders with different rates and change both the convolutional code rate and constellation. However this approach has problems in the receiver that would have to implement several different decoders for all the codes used. Puncturing is a very useful technique to generate additional rates from a single convolutional code. Puncturing was first discovered by Cain, Clark, and Geist, and subsequently the technique was improved by Hagenauer. The basic idea behind puncturing is not to transmit some of the bits output by the convolutional encoder, thus increasing the rate of the code. This increase in rate decreases the free distance of the code, but usually the resulting free distance is very close to the optimum one that is achieved by specifically designing a convolutional code for the punctured rate. The receiver inserts dummy bits to replace the punctured bits in the receiver, hence only one encoder/decoder pair is needed to generate several different code rates.

The bits that are not transmitted are defined by a puncturing pattern. Puncturing pattern is simply a set of bits that are not transmitted within a certain period of bits. Figure 28 shows the two different puncturing patterns of IEEE 802.11a. The pattern (a) is used to generate rate 3/4 code from the rate 1/2 mother convolutional code. This puncturing pattern has a period of six bits, and bits 3 and 4 are punctured (not transmitted) from each period.

The puncturing rate is equal to $4/6=2/3$ and the overall code rate is equal to $1/2*(2/3) = 3/4$, since $2/3$ of the original encoded bits are output from the puncture.

The other punctured code rate of IEEE 802.11a is a rate $2/3$ code. The puncturing pattern is shown in Figure (15) has a period of 4 bits, and the fourth bit is punctured, the puncturing rate is, $3/4$ hence the overall code rate is $1/2*(3/4) = 2/3$.

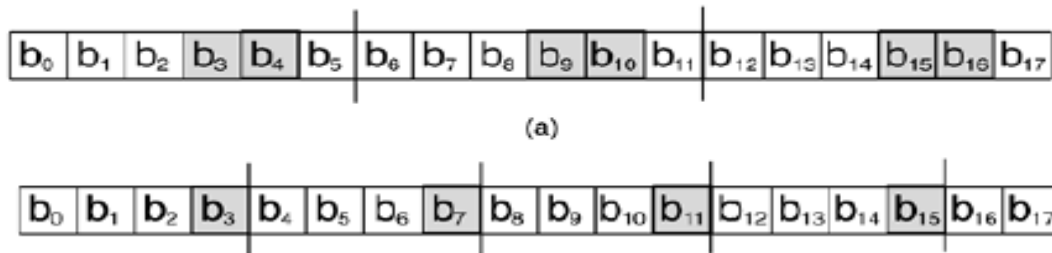


Figure 15 Puncturing patterns of IEEE

Table 3.1 shows the free distances and the asymptotic coding gains of the three code rates used in IEEE 802.11a. The table also shows the optimum rate $3/4$ and $2/3$ codes; as you can see, the performance loss due to using punctured codes, instead of the optimum ones, is very small. The rate $1/2$ is naturally the optimum code, because the original code is $1/2$ a rate code. Therefore the table does not show the punctured free distance and coding gain values for this rate.

Table 3.4. Free Ddistances of the 64 State Convolutional Codes Used in IEEE 802.11a							
Code Rates	Punctured Distance	Free	Punctured Coding	Optimum	Free	Optimum	Coding
			Gain	Distance		Gain	Gain
$\frac{1}{2}$	-	-		10		7.0 dB	
$\frac{2}{3}$	6		6.0 dB	7		6.7 dB	
$\frac{3}{4}$	5		5.7 dB	6		6.5 dB	

Table 3. 1

3.2.2.4 Function of the Block

The IEEE 802.11a/b/g specification offers support for a variety of other modulation and coding alternatives. For example, the standard allows engineers to combine BPSK, QPSK, and 16-QAM modulations with convolution encoding ($R = 1/2$ and constraint length seven) to generate data rates of 6, 12, and 24 Mbps. All other combinations of encoding rates, including $R = 2/3$ and $R = 3/4$ combined with 64-QAM, are used to generate rates up to 54 Mbps, which are optional in the standard.

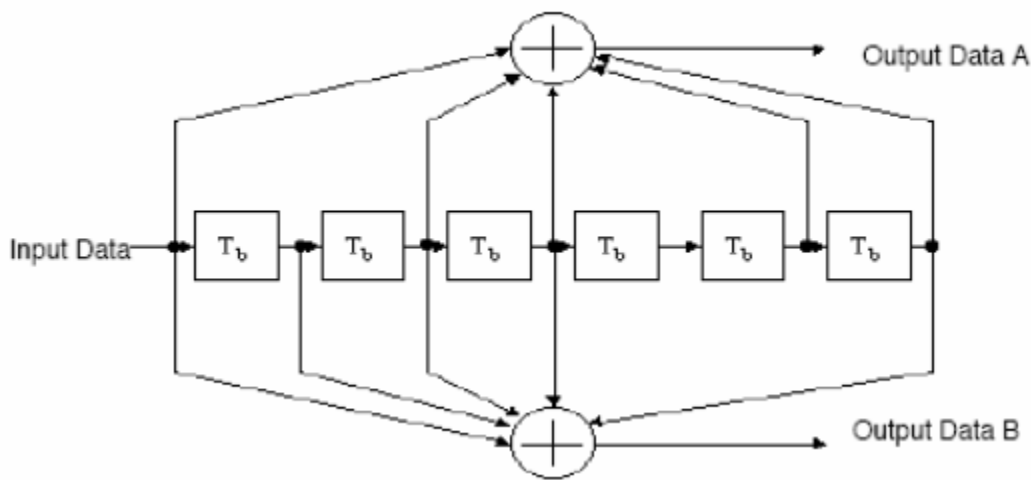


Figure 16 Convolutional encoder (K=7)

3.2.2.5 Structure

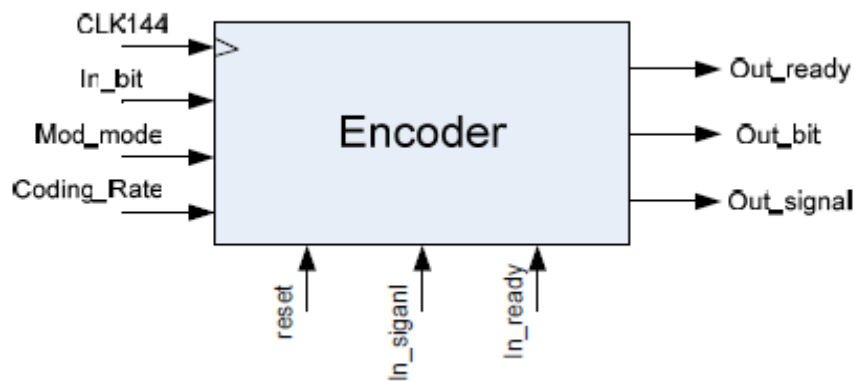
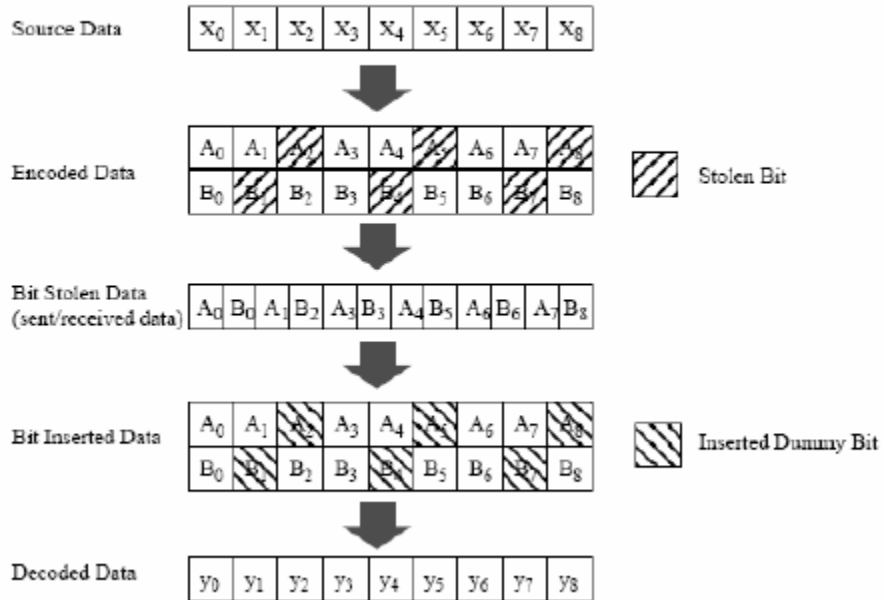


Figure 17

3.2.2.6. The VHDL code description

The encoder is formed by 6 flip flops where the input to the encoder is the input to the first flip flop and the output is A & B where, $A \leq \text{inbit} \text{ xor } \text{srg}(4) \text{ xor } \text{srg}(3) \text{ xor } \text{srg}(1) \text{ xor } \text{srg}(0)$; $B \leq \text{inbit} \text{ xor } \text{srg}(5) \text{ xor } \text{srg}(4) \text{ xor } \text{srg}(3) \text{ xor } \text{srg}(0)$; We will save the input to the encoder in a register with a variable size according to the coding rate: At rate 1/2 we don't need to use this register, at each clock of the in_clk we will have a new inbit we will compute A and B at the same time and will out A with the 1st out_clk and B will out with 2nd out_clk. At rate 2/3 we will use a register with a size =4 bits ,we will have a new inbit each in_clk then compute A and B and save them in the register then out A of the 1st bit with the 1st out_clk , out B of the same bit at the 2nd out_clk ,out A of the 2nd bit at the 3rd out_clk and ignore B of this bit. At rate 3/4 we will use a register with a size =6 bits ,we will have a new inbit each in_clk then compute A and B and save them in the register then out A of the 1st bit with the 1st out_clk , out B of the same bit at the 2nd out_clk ,out A of the 2nd bit at the 3rd out_clk and ignore B of this bit then out B of the 3rd bit at the 4th out_clk and ignore A of this bit.

Punctured Coding ($r = 3/4$)



Punctured Coding ($r = 2/3$)

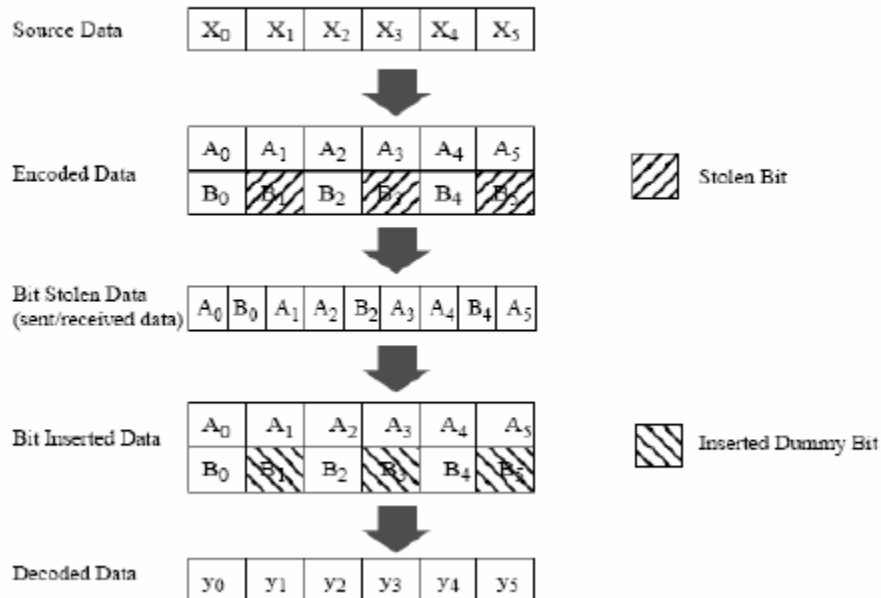


Figure 18 An example of the bit stealing and bit insertion procedure ($r=3/4, 2/3$)

3.2.2.7. Flow chart of the encoder

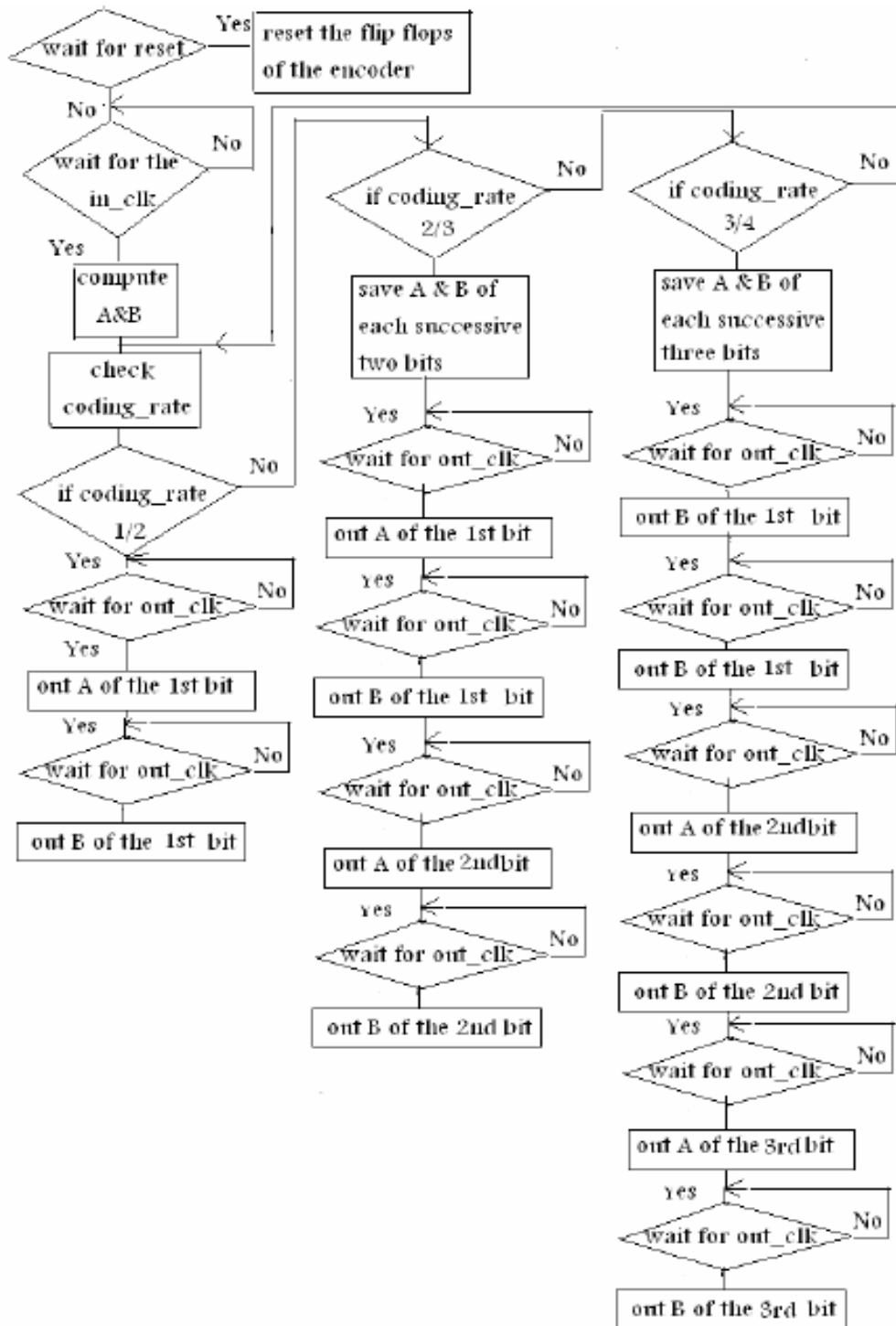


Figure 19

3.4.2.8. Post Layout Simulation

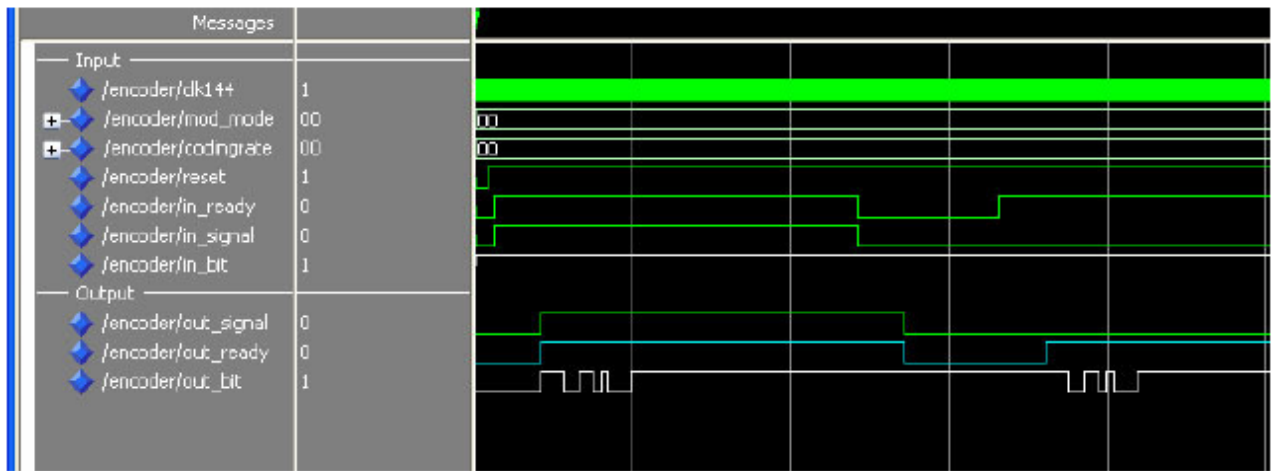


Figure 20

3.2.2.9 Performance measures

Logic utilization	< 1%
Combinational ALUTs	117 / 48,352 (< 1 %)
Dedicated logic registers	40 / 48,352 (< 1 %)
Total registers	40
Total block memory bits	0 / 2,544,192 (0 %)
Max freq	229.57 MHz

3.2.3 Interleaver

3.2.3.1. Function of block

Interleaving aims to distribute transmitted bits in time or frequency or both to achieve desirable bit error distribution after demodulation. What constitutes a desirable error distribution depends on the used FEC code. WLAN systems generally assume a very slowly fading channel, also called quasi-stationary, that does not change during one packet. WLAN systems are wide bandwidth systems, and therefore usually experience frequency

selective fading channel. OFDM technology is well suited for communication over slow frequency selective fading channels. Interleaving necessarily introduces delay into the system because bits are not received in the same order as the information source transmits them. The overall communication system usually dictates some maximum delay the system can tolerate, hence restricting the amount of interleaving that can be used.

All encoded data bits shall be interleaved by a block interleaver with a block size corresponding to the number of bits in a single OFDM symbol, NCBPS. The interleaver is defined by a two-step permutation. The first permutation ensures that adjacent coded bits are mapped onto nonadjacent subcarriers. The second ensures that adjacent coded bits are mapped alternately onto less and more significant bits of the constellation and, thereby, long runs of low reliability (LSB) bits are avoided. We shall denote by k the index of the coded bit before the first permutation; i shall be the index after the first and before the second permutation, and j shall be the index after the second permutation, just prior to modulation mapping. The first permutation is defined by the rule

$$i = (\text{NCBPS}/16) (k \bmod 16) + \text{floor}(k/16) \quad k = 0,1,\dots,\text{NCBPS} - 1$$

The function $\text{floor}(\cdot)$ denotes the largest integer not exceeding the parameter. The second permutation is defined by the rule

$$j = s \times \text{floor}(i/s) + (i + \text{NCBPS} - \text{floor}(16 \times i/\text{NCBPS})) \bmod s$$

$$i = 0,1,\dots, \text{NCBPS} - 1$$

The value of s is determined by the number of coded bits per subcarrier, NBPSC, according to

$$s = \max(\text{NBPSC}/2,1)$$

3.2.3.3 Interleaver Implementation

The implementation is based on the equations mentioned above, after simulating these equations on Matlab the interleaving addresses is generated by VHDL conditions ,these addresses is used to save the coming data (original arranged data) into LPM RAMs by the interleaved addresses, then reading this data out and sending them using the linear address of the RAM.

<i>Data rate (Mbits/s)</i>	<i>Modulation</i>	<i>Coding rate (R)</i>	<i>Coded bits per subcarrier (NBPS)</i>	<i>Coded bits per OFDM symbol (NCBPS)</i>	<i>Data bit per OFDM symbol (NDBPS)</i>
6	BPSK	$\frac{1}{2}$	1	48	24
9	BPSK	$\frac{3}{4}$	1	48	36
12	QPSK	$\frac{1}{2}$	2	96	48
18	QPSK	$\frac{3}{4}$	2	96	72
24	16-QAM	$\frac{1}{2}$	4	192	96
36	16-QAM	$\frac{3}{4}$	4	192	144
48	64-QAM	$\frac{2}{3}$	6	288	192
54	64-QAM	$\frac{3}{4}$	6	288	216

Figure 21 Rate dependent parameters

3.2.3.4 Post Layout Simulation

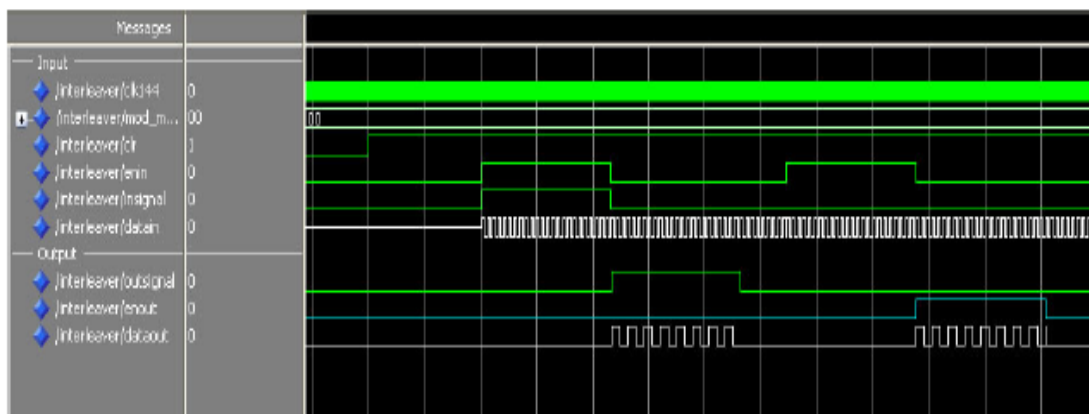


Figure 22

3.2.3.5 Performance Measurement

Logic utilization	< 1%
Combinational ALUTs	224 / 48,352 (< 1 %)
Dedicated logic registers	52 / 48,352 (< 1 %)
Total registers	52
Total block memory bits	512 / 2,544,192 (< 1 %)
Max freq	201.25 MHz

3.2.4 Mapper

3.2.4.1. Function of the block

The OFDM subcarriers shall be modulated by using BPSK, QPSK, 16-QAM, or 64-QAM modulation, depending on the RATE requested. The encoded and interleaved binary serial input data shall be divided into groups of NBPSC (1, 2, 4, or 6) bits and converted into complex numbers representing BPSK, QPSK, 16-QAM, or 64-QAM constellation points. The conversion shall be performed according to Gray-coded constellation mappings, illustrated in Figure 116, with the input bit, b_0 , being the earliest in the stream. The output values, d , are formed by multiplying the resulting $(I+jQ)$ value by a normalization factor $KMOD$, as described in this equation.

$$d = (I + jQ) \times KMOD$$

The normalization factor, $KMOD$, depends on the base modulation mode, as prescribed in Table 81. Note that the modulation type can be different from the start to the end of the transmission, as the signal changes from SIGNAL to DATA, as shown in Figure 107. The purpose of the normalization factor is to achieve the same average power for all mappings. In practical implementations, an approximate value of the normalization factor can be used, as long as the device conforms with the modulation accuracy requirements.

Modulation	K_{MOD}
BPSK	1
QPSK	$1/\sqrt{2}$
16-QAM	$1/\sqrt{10}$
64-QAM	$1/\sqrt{42}$

Table 3. 2 Modulation-dependent normalization factor K_{MOD}

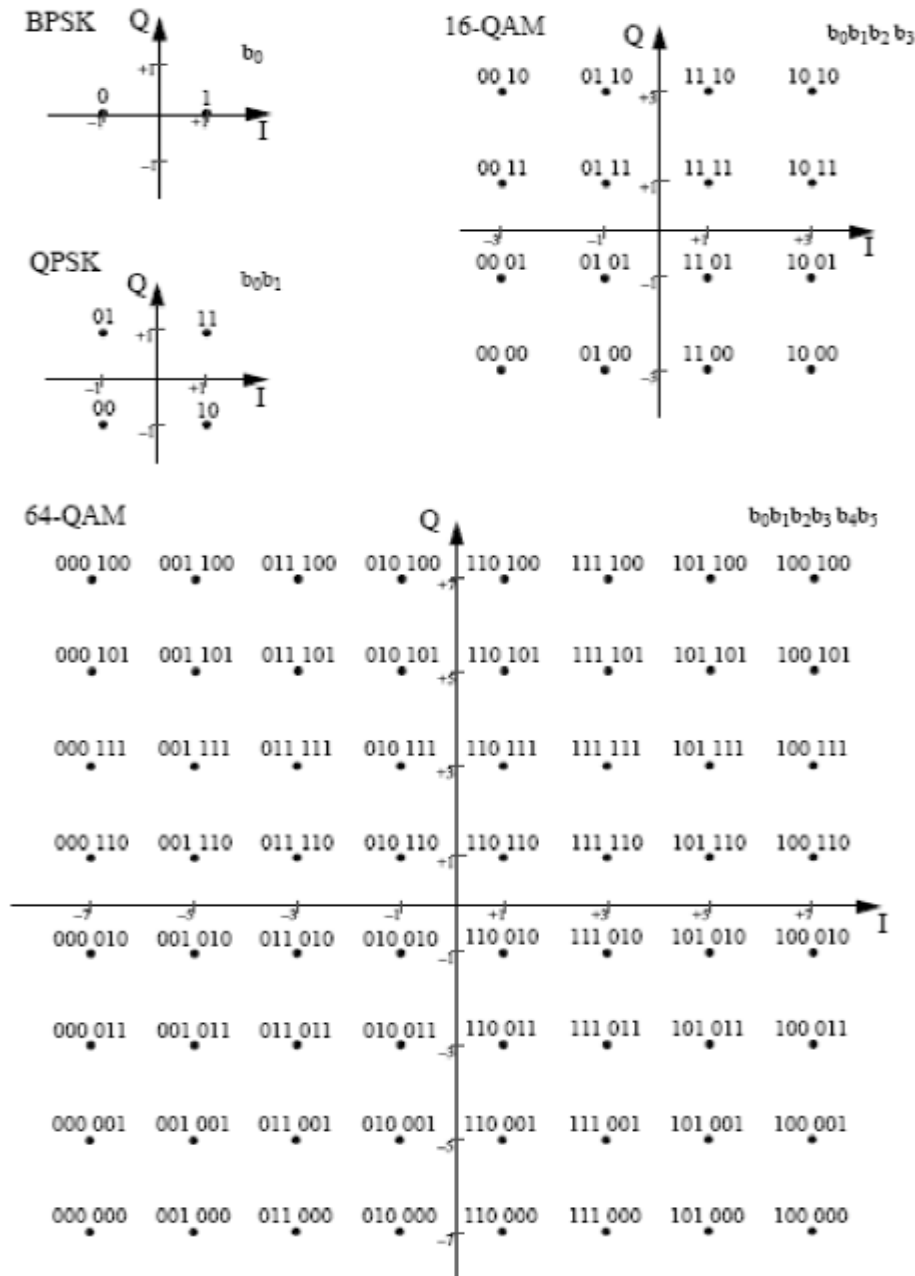


Figure 23 BPSK, QPSK, 16-QAM, and 64-QAM constellation bit encoding

For BPSK, b_0 determines the I value, as illustrated in Table 3.3. For QPSK, b_0 determines the I value and b_1 determines the Q value, as illustrated in Table 3.4. For 16-QAM, b_0b_1 determines the I value and b_2b_3 determines the Q value, as illustrated in Table 3.5. For 64-QAM, $b_0b_1b_2b_3$ determines the I value and b_4b_5 determines the Q value, as illustrated in Table 3.6.

Input bit (b_0)	I-out	Q-out
0	-1	0
1	1	0

Table 3. 3 BPSK encoding table

Input bit (b_0)	I-out	Input bit (b_1)	Q-out
0	-1	0	-1
1	1	1	1

Table 3. 4 QPSK encoding table 5 QPSK encoding table

Input bits ($b_0 b_1$)	I-out	Input bits ($b_2 b_3$)	Q-out
00	-3	00	-3
01	-1	01	-1
11	1	11	1
10	3	10	3

Table 3. 5 16-QAM encoding table

Input bits ($b_0 b_1 b_2$)	I-out	Input bits ($b_3 b_4 b_5$)	Q-out
000	-7	000	-7
001	-5	001	-5
011	-3	011	-3
010	-1	010	-1
110	1	110	1
111	3	111	3
101	5	101	5
100	7	100	7

Table 3. 6 64-QAM encoding table

3.2.4.2 Block diagram

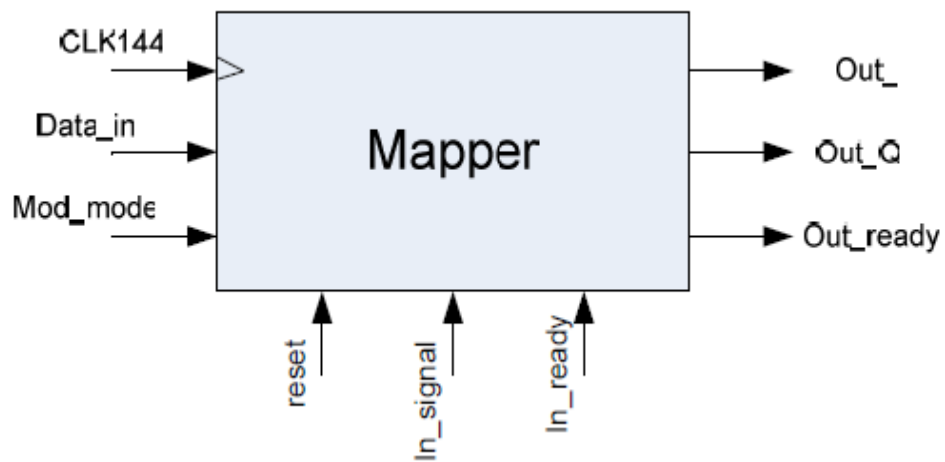


Figure 24

3.2.4.3 The VHDL Code description

The mapper will work when the data from previous block (interleaver) become ready, i.e. when `data_ready` signal become activated (high). - The mapper receives 1 bit at a time from the interleaver, so it should form a group of bits such that each group consists of (1,2,4 or 6) bits according to the modulation mode used and the mapper should convert each group into a complex number, therefore we've made in the code 6 states for the input clock (`clk1,clk2,clk3,clk4,clk5,clk6`), where at each i/p clock, 1 bit is input to the mapper, and we've made a latch (b) with size equals 6 bits (because the max # of bits/symbol=6 bits) to store the required group of bits, where I and Q will be calculated after the group of bits is formed. - And then I and Q will be outputted at an output clock (`clk_out_mapper`), and `out_ready` signal become activated. - If we press reset (i.e. `reset=0`), we will start processing beginning from the 1st state (`clk1`).

At the first input clock (at clk1):

The input bit is stored in b (0). -Then if the `modulation_mode` is BPSK , mapper'll check the value of the bit, and according to its value, I and Q are computed and `out_ready` become activated (high), then I and Q are outputted if `clk_out='1'` and then return again to the 1st state for receiving a new input bit. -But if the `modulation_mode` isn't BPSK, we'll enter the 2nd state (`clk2`).

At the 2nd input clock (at clk2):

At `clk2`, the 2nd input bit is stored in b (1). -And if `modulation_mode` is QPSK , mapper'll check the values of both `b(0)&b(1)` together and according to their values, I and Q are computed and then outputted if `clk_out='1'` and then return again to the 1st state for receiving a new bit. But if `modulation_mode` isn't QPSK, we'll enter the 3rd state (`clk3`).

At the 3rd input clock (at clk3):

-where at clk3, the 3rd input bit is stored in b (2) then we'll enter the 4th state(clk4).

At the 4th input clock (at clk4): -At clk4 the 4th input bit is stored in b (3). - And if modulation_mode is 16 QAM , b(0)&b(1) together will determine I value while b(2)&b(3) together will determine Q value as shown in the constellation mapping , and if clk_out = '1' , I and Q will be outputted. And then return again to 1st state.

-But if modulation_mode isn't 16 QAM, we'll enter 5th state (clk5).

At the 5th input clock (at clk5):

-In this state the 5th input bit is stored in b (4) then we'll enter 6th state (clk6).

At the 6th input clock (at clk6):

- At clk6 the 6th input bit is stored in b (5). - And if modulation_mode is 64 QAM , b(0)&b(1)&b(2) together will determine I value while b(3)&b(4)&b(5) together will determine Q value as seen from the signal constellation, then we'll enter 1st state(clk1) , and if clk_out = '1' , I and Q will be outputted, and the above steps will be similarly repeated again for the remaining incoming data bits.

3.2.4.4. The Calculated I and Q values in the mapper

9 bits will be needed to represent each of I and Q, where 2 bits for the integer # before the point and 7 bits for the fractions. So, we've calculate I and Q values in 9 bits according to the modulation mode used, where there is a modulation factor (Kmod) that is taken into consideration.

1)- Bpsk:

+1=010000000=80H

-1=110000000=180H

2)- Qpsk:

+1/√2= +0.7071=001011010=5AH

-1/√2= -0.7071=110100110 =1A6H

3)- 16Qam:

$$+1/\sqrt{10} = +0.3162 = 000101000 = 28H$$

$$-1/\sqrt{10} = -0.3162 = 111011000 = 1D8H$$

$$+3/\sqrt{10} = +0.9487 = 001111001 = 79H$$

$$-3/\sqrt{10} = -0.9487 = 110000111 = 187H$$

4)- 64Qam:

$$+1/\sqrt{42} = +0.1543 = 000010011 = 13H$$

$$-1/\sqrt{42} = -0.1543 = 111101101 = 1EDH$$

$$+3/\sqrt{42} = +0.4629 = 000111011 = 3BH$$

$$-3/\sqrt{42} = -0.4629 = 111000101 = 1C5H$$

$$+5/\sqrt{42} = +0.7715 = 001100010 = 62H$$

$$-5/\sqrt{42} = -0.7715 = 110011110 = 19EH$$

$$+7/\sqrt{42} = +1.0801 = 010001010 = 8AH$$

$$-7/\sqrt{42} = -1.0801 = 101110110 = 176H$$

3.2.4.5. Flow chart for the mapper

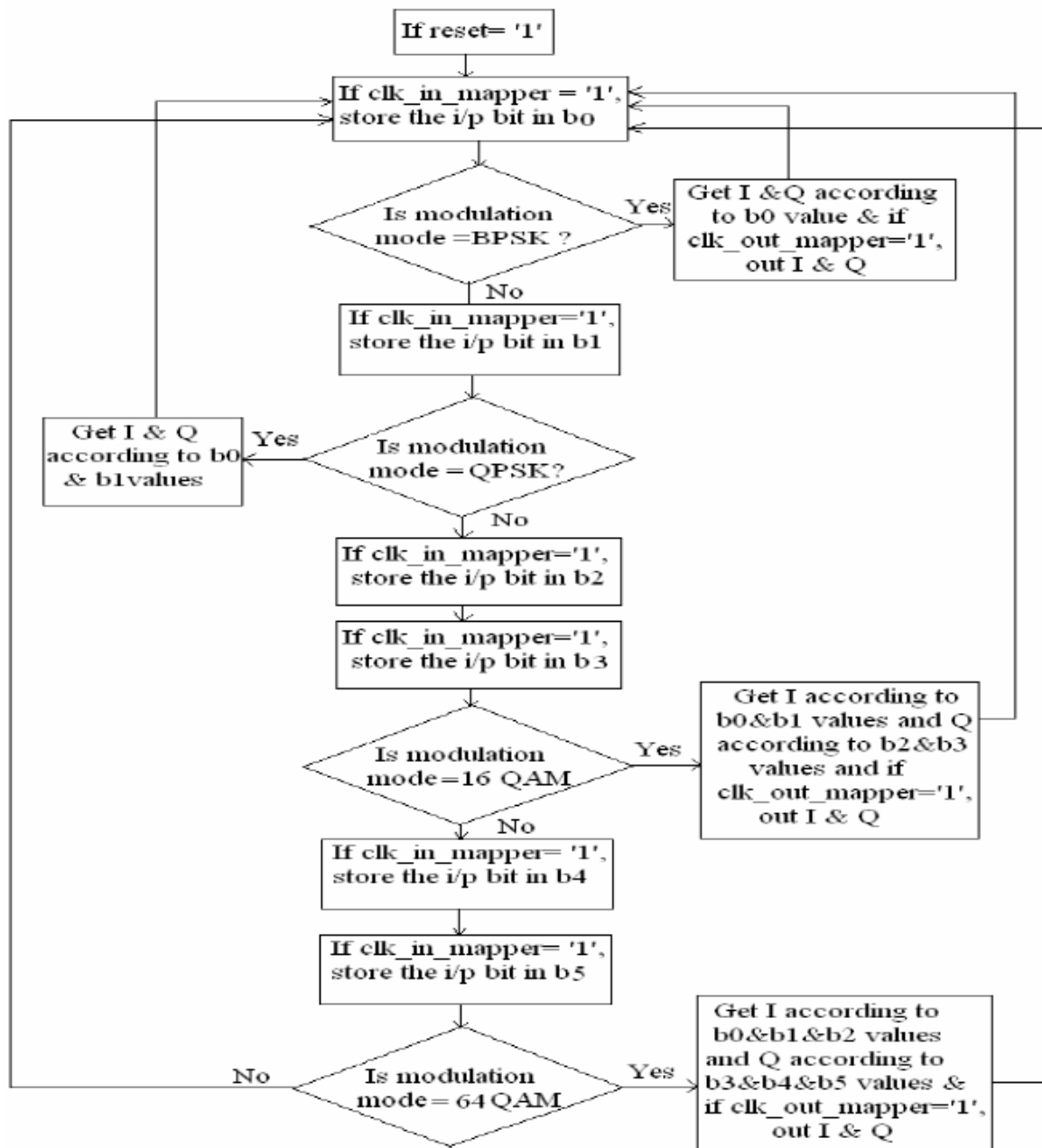


Figure 25

3.2.4.6 Post Layout Simulation

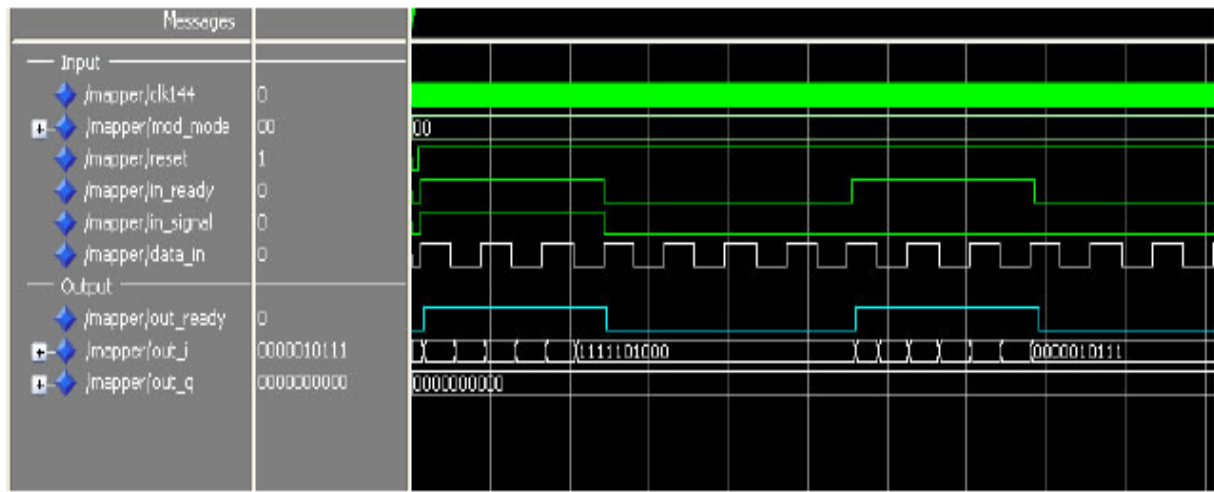


Figure 26

3.2.4.7 Performance Measurement

Logic utilization	< 1%
Combinational ALUTs	32 / 48,352 (< 1 %)
Dedicated logic registers	14 / 48,352 (< 1 %)
Total registers	14
Total block memory bits	0 / 2,544,192 (0 %)
Max freq	483.56Mhz

3.2.5. Framer

3.2.5.1. Function of the Block

In each OFDM symbol, four of the subcarriers are dedicated to pilot signals. These pilots are used for tracking the frequency and phase errors in the OFDM received signal. They also help in estimating and correcting the channel effect. These pilot signals shall be put in subcarriers -21 , -7 , 7 and 21 . The pilots shall be BPSK modulated by a pseudo binary sequence to prevent the generation of spectral lines.

3.2.5.3 VHDL Description

I_{in} and Q_{in} are the complex outputs (real and imaginary) from the mapper . The in_ready signal is also from the mapper to indicate the start of the transmission. The out_ready signal indicates that a valid output is ready in I_{out} and Q_{out} signals. This block receives 48 input (I & Q) from the mapper we insert zeros and pilots , Hence the output will be 64 sample including this zeros and pilots. Pilots are calculated from the output of the PN-Generator described above, which produce an output each 64 clock cycle. In the implementation of this block, we used double buffering in order to pipeline the operation. The output pattern is formed as the standard required for the IFFT input as shown in Figure (29).

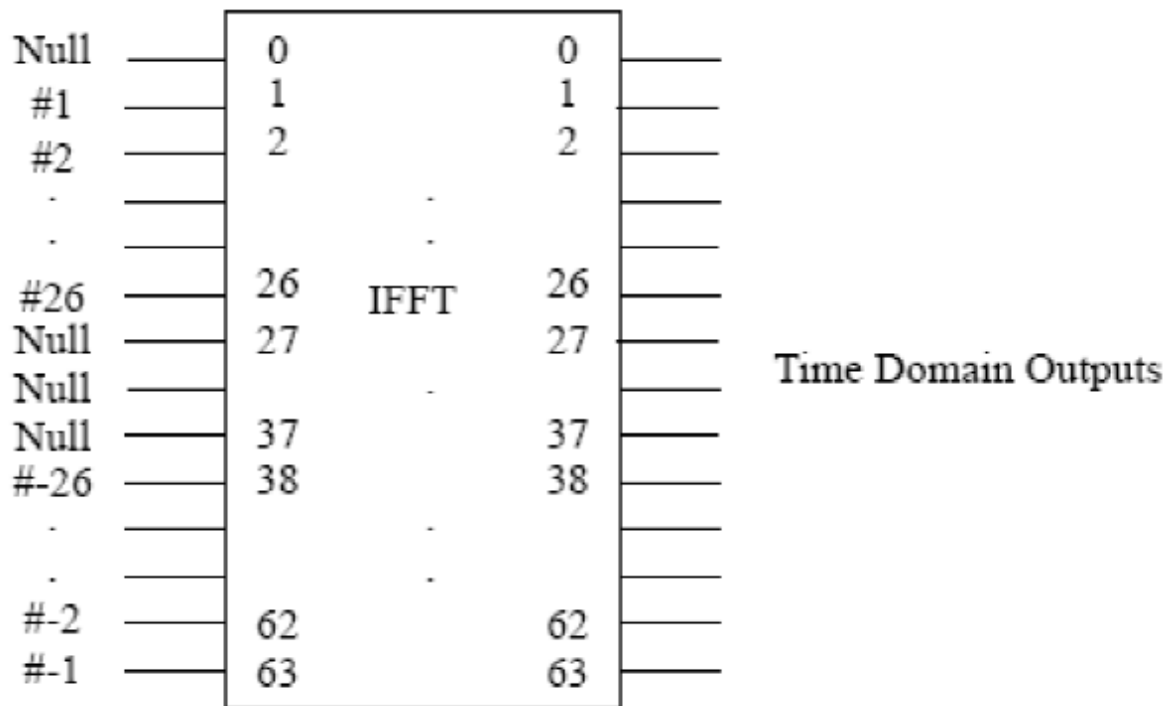


Figure 29 Inputs and outputs of the IFFT block in the 802.11a WLAN standard

This block has 1 symbol delay as it stores a complete symbol, output the symbols in the right order, insert pilots and zeros in their positions.

Theoretical notes: The negative part of the input OFDM symbol is transformed to the end of the symbol in the framer because the IFFT block has a range from 0 to 64, and this has nothing to do with the signal as the frequency domain of the DFT is periodic, therefore, this is theoretically right. The Zeros insertion in the middle of the output frame is actually in the high frequency part of the symbol (in the symbol, the max. freq. is at sample $64/2$ [from the DFT properties]).

3.2.5.4. Post Layout Simulation

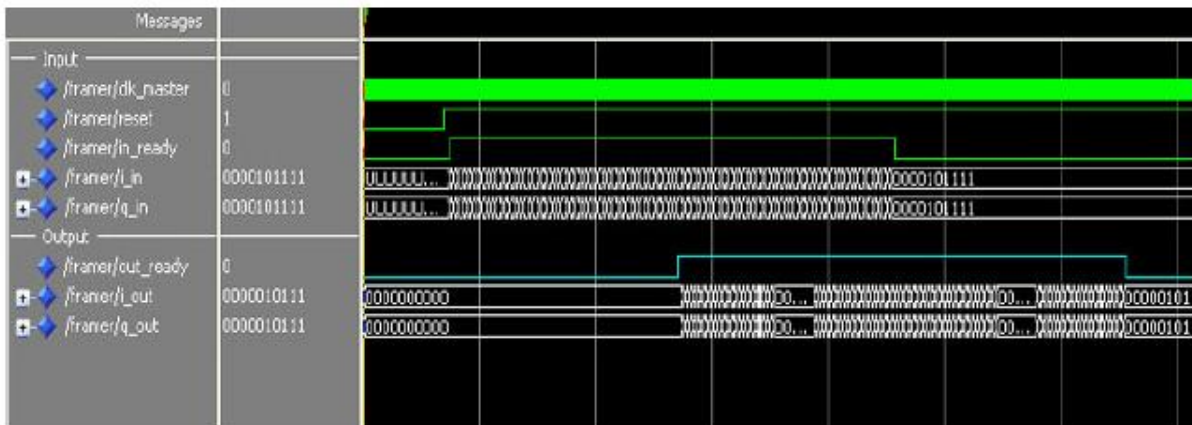


Figure 30

3.2.5.5. Performance measurement

Logic utilization	< 1%
Combinational ALUTs	77 / 48,352 (< 1 %)
Dedicated logic registers	50 / 48,352 (< 1 %)
Total registers	50
Total block memory bits	2560 / 2,544,192 (< 1 %)
Max freq	166.67Mhz

3.2.6 FFT/IFFT

The fast Fourier Transform and inverse fast Fourier Transform operations must be carried out in the transmitter and the receiver of the Wi-Fi system as they carry out the conversion of OFDM symbol from Frequency domain to time domain in transmitter and vice versa in receiver.

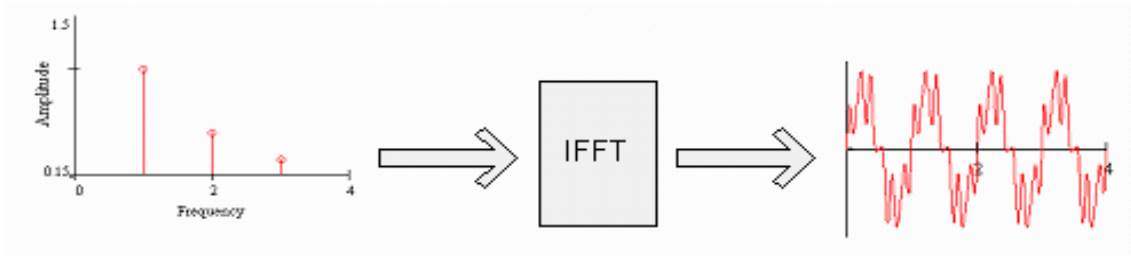


Figure 31 conversion from frequency domain to time domain in transmitter

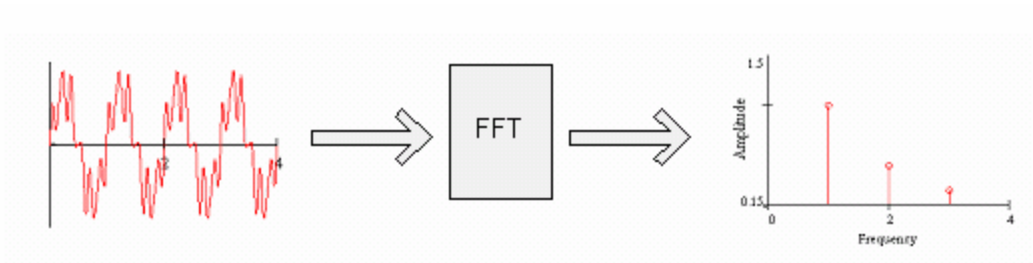


Figure 32 conversion from time domain to frequency domain in receiver

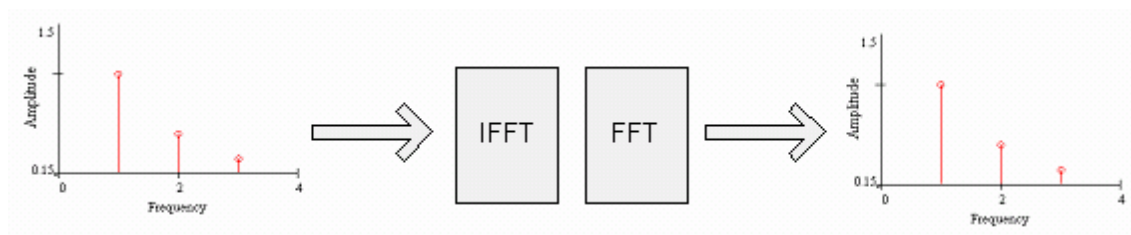


Figure 33 output of the two operations back to back

IFFT/FFT illustration

3.2.6.1 CFFT core

In our Wi-Fi Transceiver we used the CFFT core done by ZHAO Ming and available in the website opencores.org and we connected our interface to this component to control it and make it compatible with our design's handshaking system and compatible with the input and output data rates to it.

3.2.6.2 Description

CFFT is a radix-4 fast Fourier transform (FFT) core with configurable data width and a configurable number of sample points in the FFT. Twiddle factors are implemented using the CORDIC algorithm, causing the gain of the CFFT core to be different from the standard FFT algorithm. This variation in gain is not important for orthogonal frequency division modulation (OFDM) and demodulation. The gain can be corrected, to that of a conventional FFT, by applying a constant multiplying factor. The input of FFT core is ordered by radix 4 and the output is reverse ordered.

3.2.6.3 Block diagram

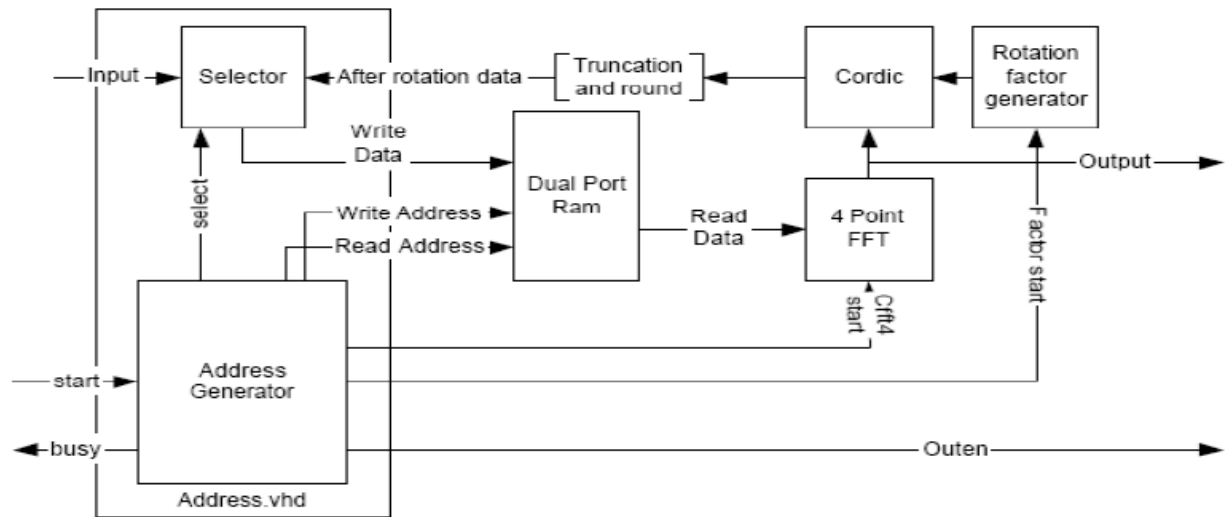
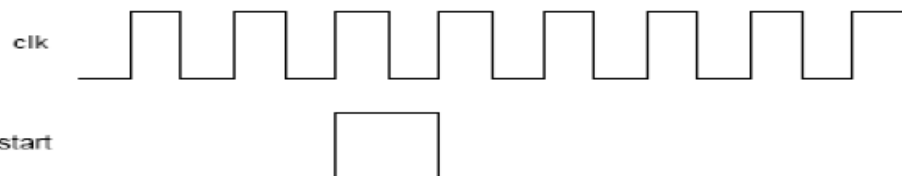


Figure 34 CFFT Block Diagram

3.2.6.4 Timing

A. input timing



B. output timing

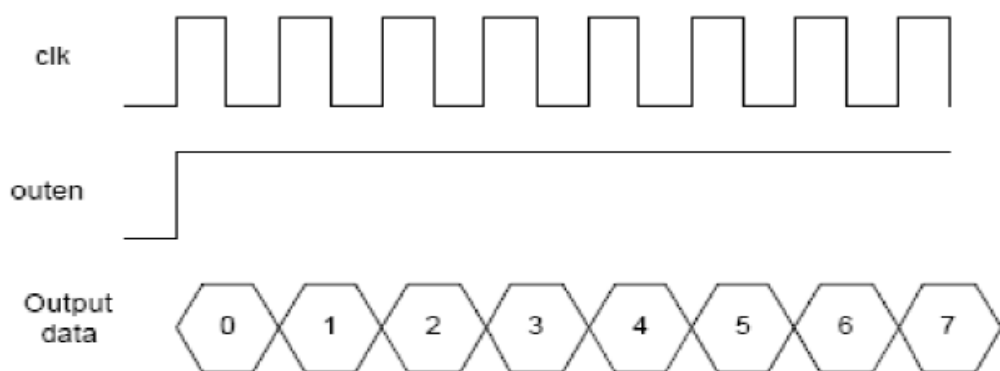


Figure 35 Timing of input and output of CFFT core

3.2.6.5 Gain

For 64 point IFFT operation gain = 10.84765. For 64 point FFT operation gain = 0.16943. Overall gain does not equal one so it is handled up in the Mapper and the demapper look up tables of in phase and quadrature values.

3.2.6.6 Interfacing the CFFT core to the system

The double buffering technique is used to interface the CFFT core to our system in order to assure pipelining and supporting a constant input and output rates of 16 mega HZ, the double buffering technique is depending on the existence of 4 memories, two memories to pipeline the input stream(input clock is 16 MHZ, output clock is 72 MHZ) and the other two to pipeline the output stream(input clock is 72 MHZ, output clock is 16 MHZ), the input and output rates is assured by the feeding clocks attached to the memories.

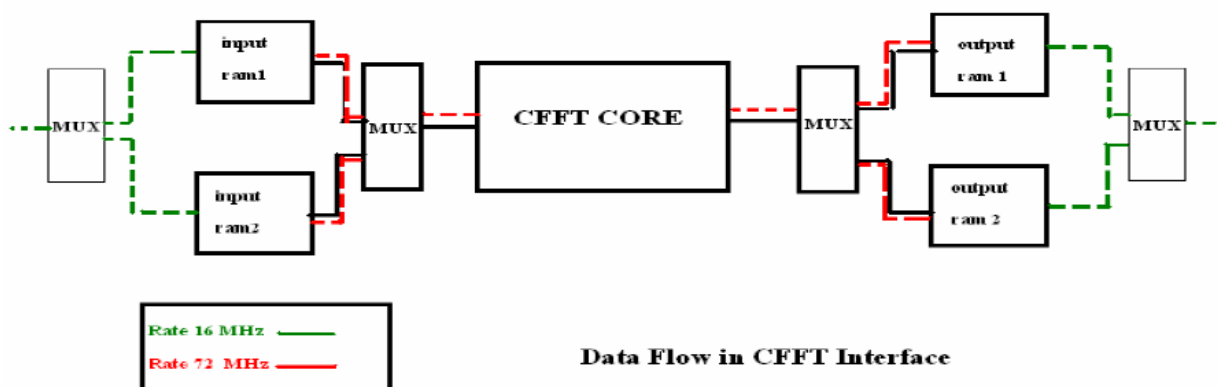


Figure 36

In the other hand the CFFT core is working in a higher clock of 72 mega Hz value to finish its work in the current input OFDM symbol before

the next one is buffered in the input memories with 16 mega Hz and be ready to be the new input to the CFFT core.

3.2.6.7 Structure:

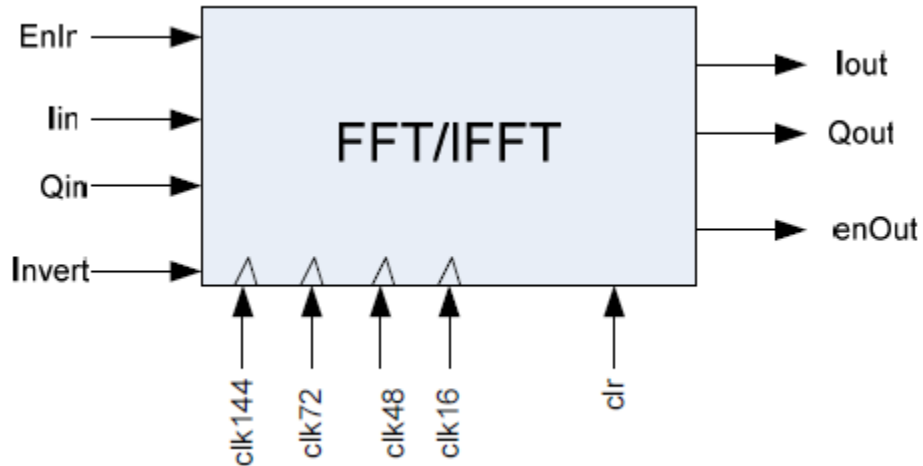


Figure 37

Inputs:

- En In: enable of the input data.
- CLR: to reset the block (active low).
- CLK144: input clock to the block 144 mega Hz.
- CLK72: input clock to the block 72 mega Hz.
- CLK48: input clock to the block 48 mega Hz.
- CLK16: input clock to the block 16 mega Hz.
- Invert: input flag to specify the operation (0 for FFT, 1 for IFFT).
- I in: input in phase component.
- Q in: input quadrature component.

Outputs:

- En out: handshaking flag starts and ends with the block output.

- I out: output in phase component.
- Q out: output quadrature component.

3.2.6.8 Post Layout Simulation

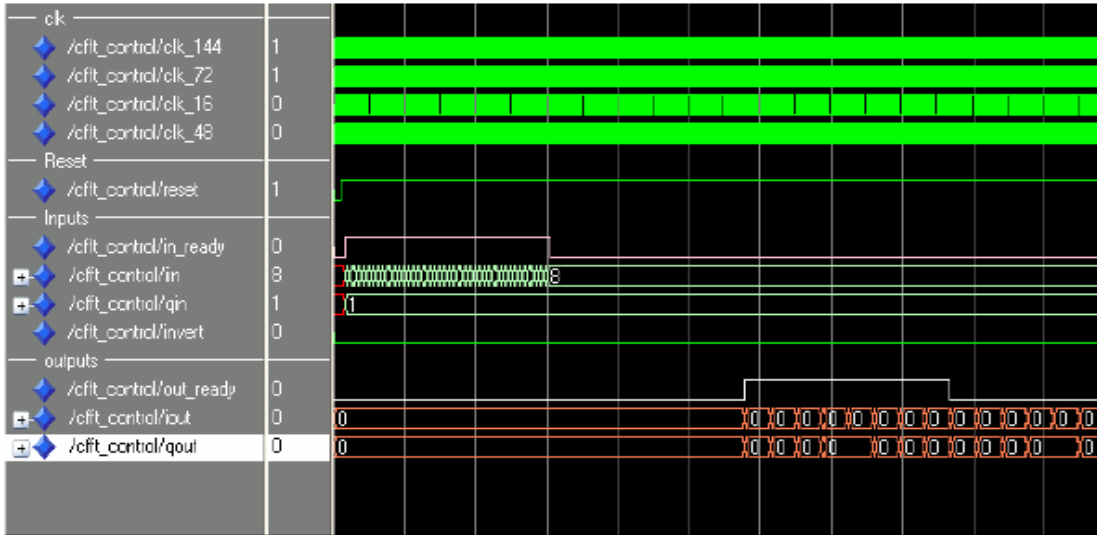


Figure 38 post layout simulation of CFFT_Control component

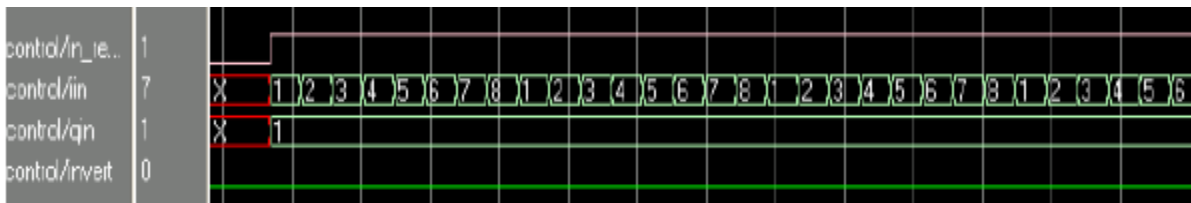


Figure 39 Input Zoom in

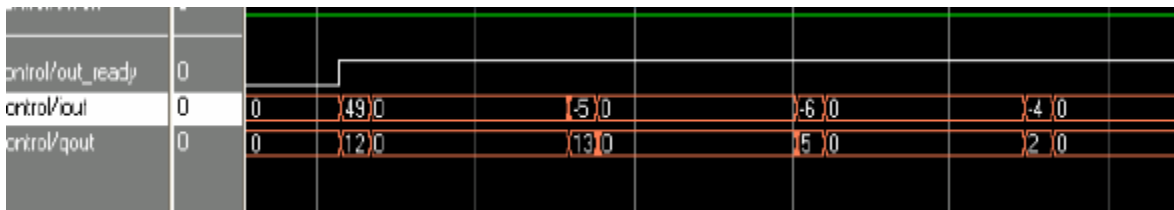


Figure 40 Output Zoom in

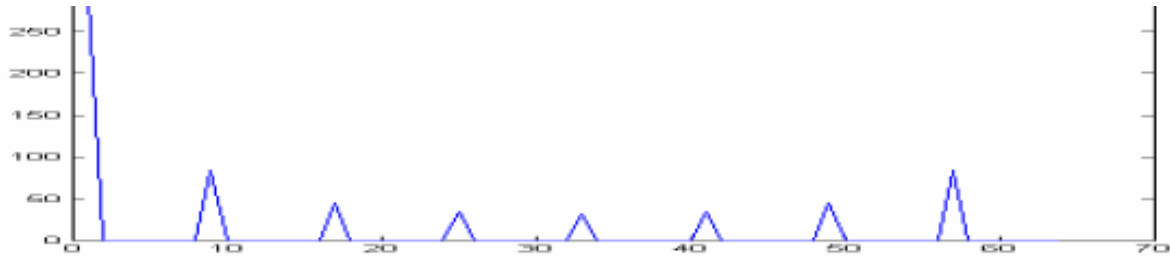


Figure 41 Matlab output for the same input

3.2.6.9 Performance measurements:

Logic utilization:	4 %.
Combinational ALUTs:	1,172 / 48,352 (2 %).
Dedicated logic registers:	1,511 / 48,352 (3 %)
Total registers:	1511
Max Frequency of clk144	500 MHz
Max Frequency of clk72	200 MHz
Max Frequency of clk48	475 MHz
Max Frequency of clk16	233 MHz

3.7.2 Cyclic Extension

3.2.7.1 Function of Block

Cyclic Prefix (CP) insertion: This block is used to introduce a prefix and a suffix and to window the OFDM symbols. In this standard, the prefix is 16 samples long. One of these prefix samples is used for the windowing and the remaining 15 are used to compensate for the channel delay spread. The whole OFDM symbol is therefore 80 samples long. The selection of the conforming pulse $w(n)$ is left to the designer. Nevertheless the standards suggest the window $w(n)$ being 0.5 for $n = 0, 80$ and 1 for $1 \leq n \leq 79$.

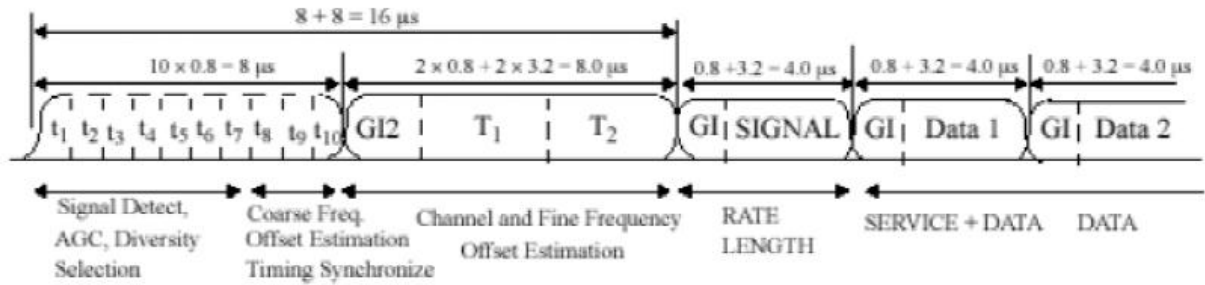


Figure 42 OFDM symbol after cyclic extension

The parts from t_1 to t_{10} are short training symbols, that are all identical and 16 samples long. GI is a 32-sample cyclic prefix that protects the long training symbols T1 and T2 from intersymbol interference (ISI) caused by the short training symbols. The long training symbols are identical 64 samples long OFDM symbols.

3.2.7.2. Structure

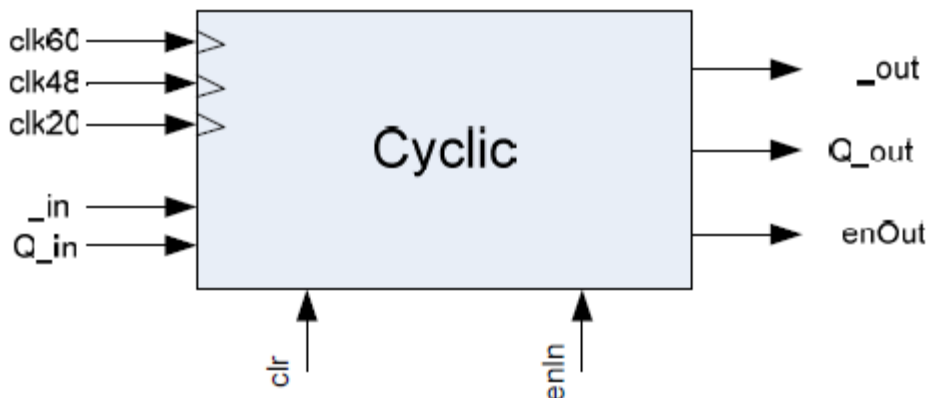


Figure 43

3.2.7.3 The VHDL code description

We have two ALTSyn rams .

We receive each clock (clk_in) one bit and store it in the first ram till this ram is full with the 64 bits of one OFDM symbol. - At the same time we

read each clock (clk_out) from the second ram the last 16 bits and then the whole 64 bits so now we extended the data to 80 bits. - And after finishing reading from the 2nd ram and writing in the 1st ram we write in the 2nd ram and read from the 1st ram and we do so not to waste any data during reading from one ram. - The first time we will read from the 2nd ram that we haven't write anything in it yet so we will ignore the output at the first time only that will be done by the controller.

3.2.7.4. Flow chart of the cyclic extension block:

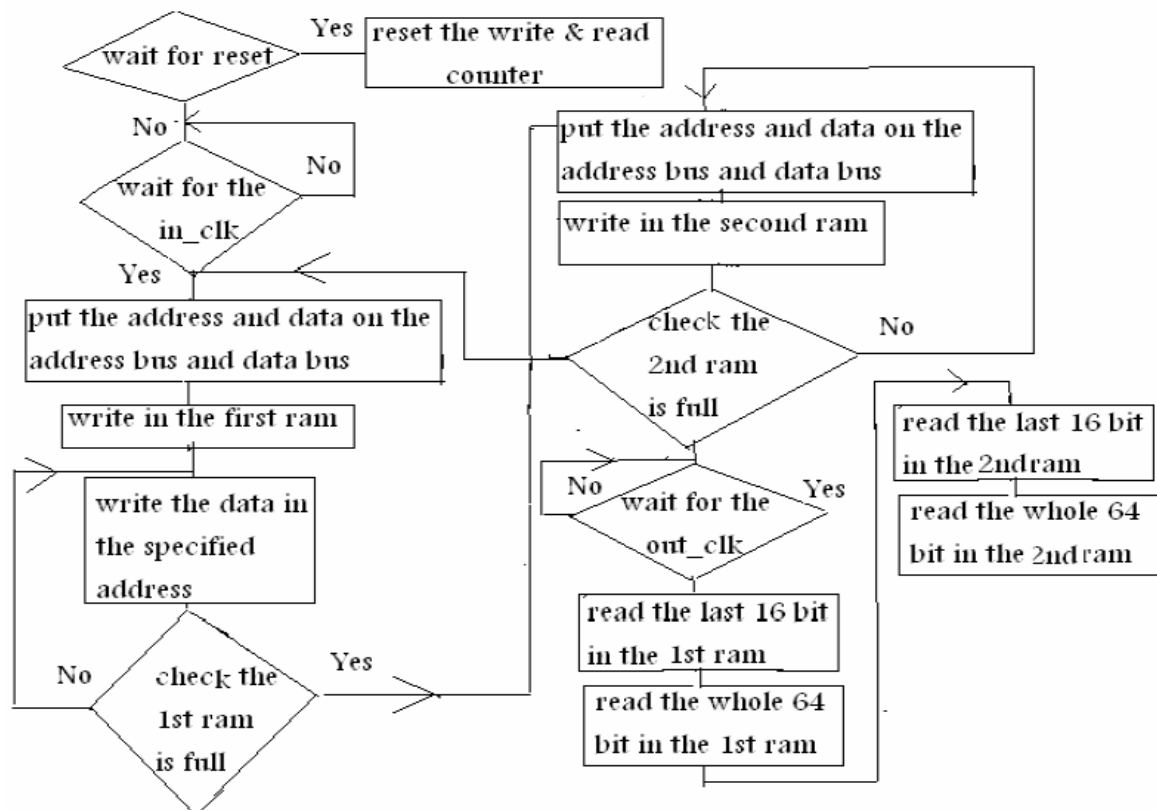


Figure 44

3.2.7.5 Post Layout Simulation

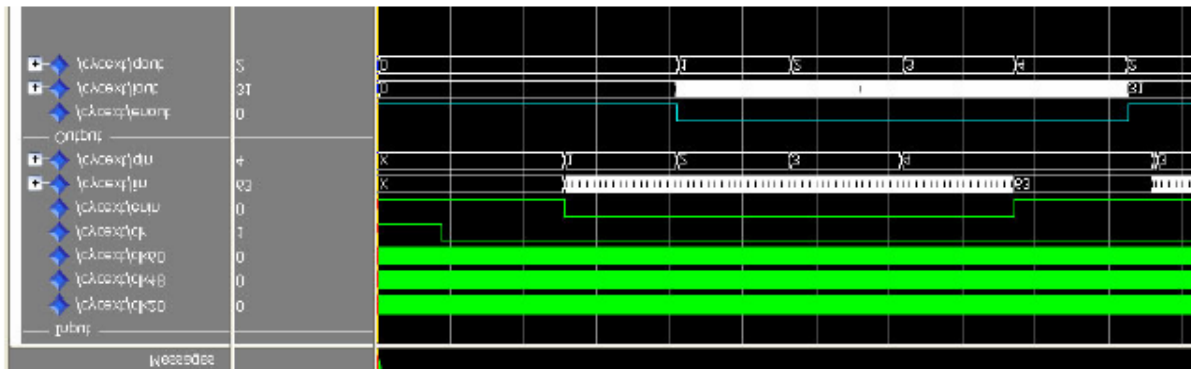


Figure 45

3.2.7.6 Performance measurement

Logic utilization	< 1%
Combinational ALUTs	89 / 48,352 (< 1 %)
Dedicated logic registers	112 / 48,352 (< 1 %)
Total registers	112
Total block memory bits	2560 / 2,544,192 (< 1 %)
Max freq	329.92Mhz

3.2.8 Preamble

3.2.8.1 Function of the Block

To output the preamble (training) sequence in time domain appended by the signal then the data.

3.2.8.2. Structure

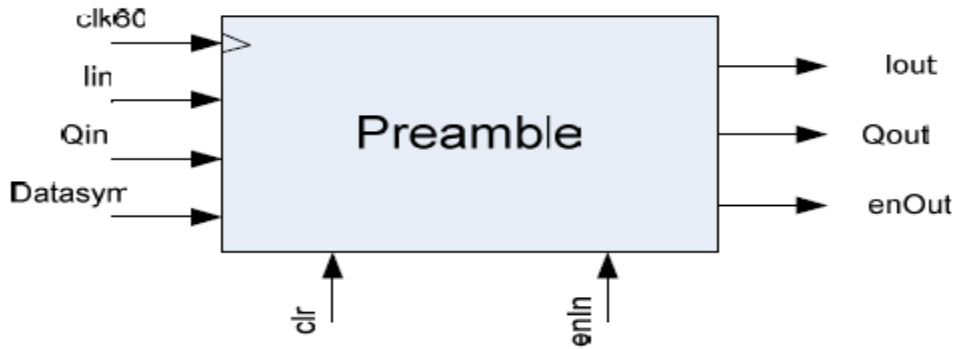


Figure 46

3.2.8.3. Flow chart of the Preamble block

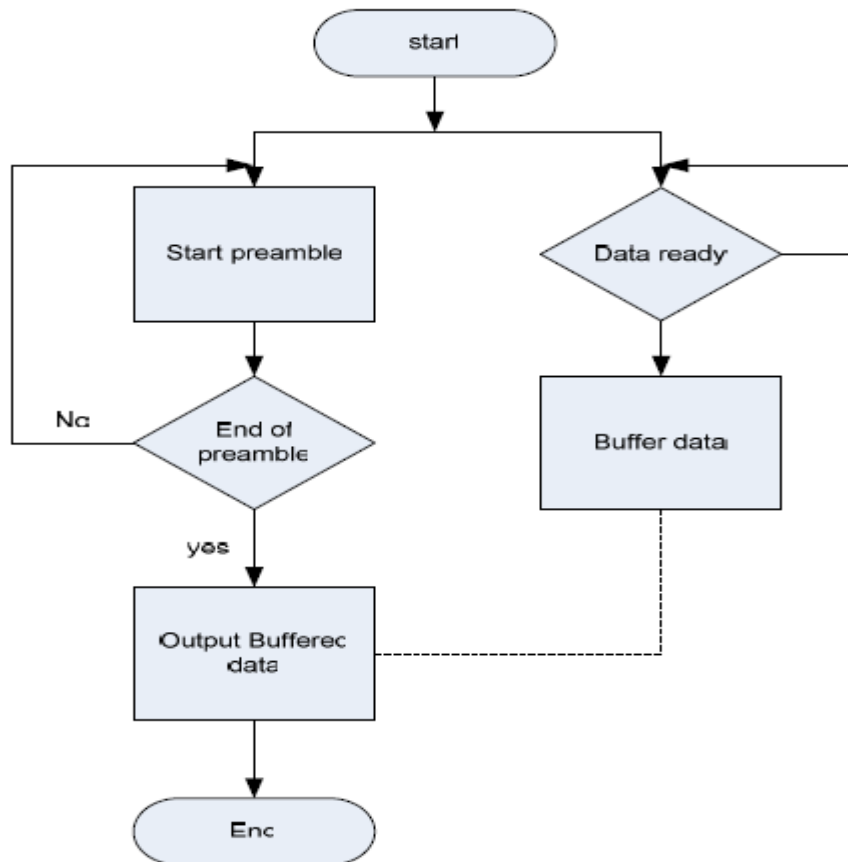


Figure 47

3.2.8.4 Post Layout Simulation

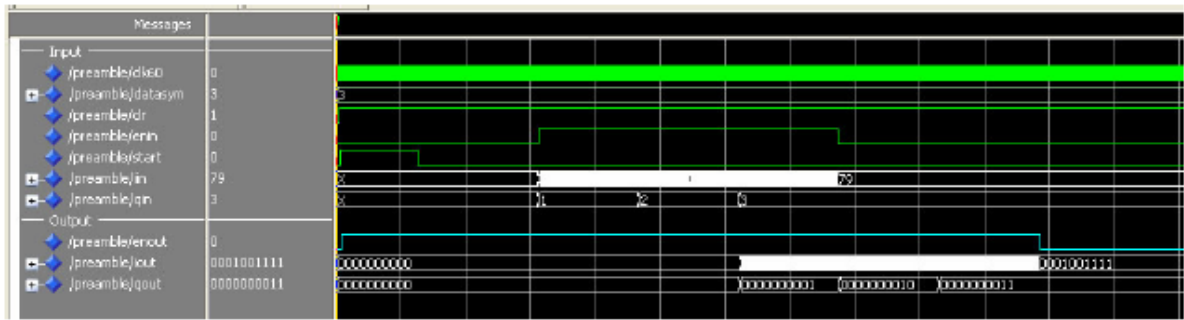


Figure 48

3.2.8.5 Performance measurement

Logic utilization	< 1%
Combinational ALUTs	274 / 48,352 (< 1 %)
Dedicated logic registers	211 / 48,352 (< 1 %)
Total registers	211
Total block memory bits	23040 / 2,544,192 (< 1 %)
Max freq	134.44Mhz

3.3 Receiver Blocks Implementation

3.3.1 Packet divider

This is the first block in the receiver, and its main target is to divide the input packet to the receiver into three parts:

4. Preambles (short and long training sequence 4 OFDM symbols).
5. Signal symbol.
6. Data symbols.

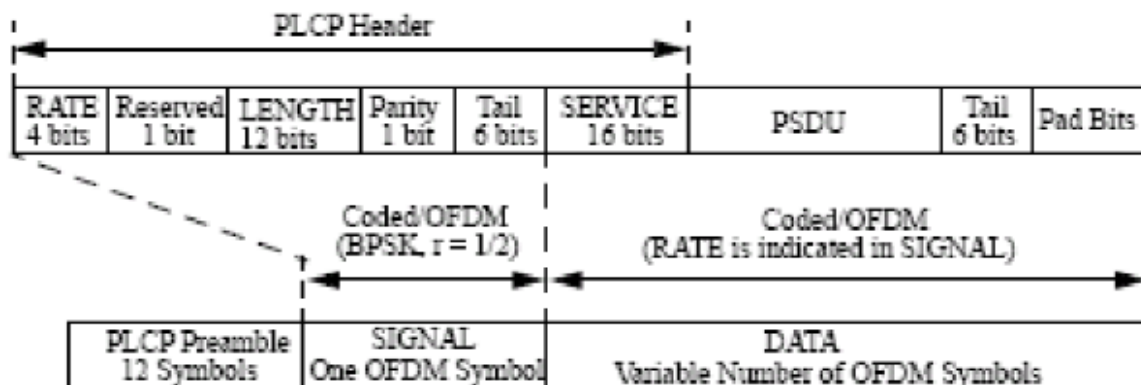


Figure 49 Formation of WiFi Packet

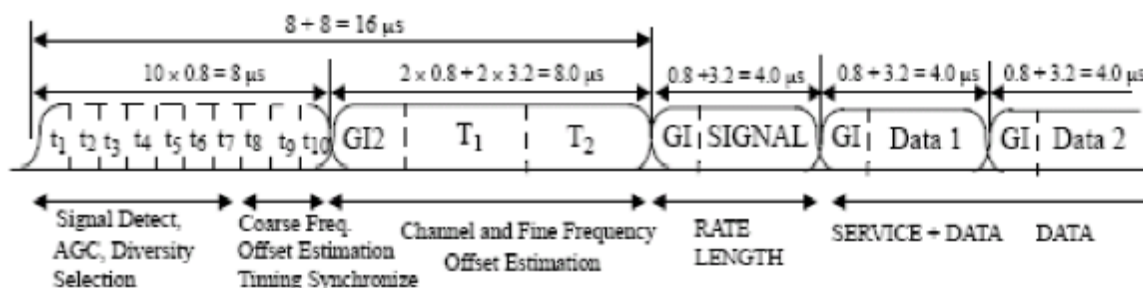


Figure 50 OFDM training structure

The PLCP preamble field is used for synchronization. It consists of 10 short symbols and two long symbols; the Packet divider block isolates the preambles and gives it a special handshaking flag starts and ends with it and deliver the preamble to the synchronization blocks.

Also the packet divider block give the Signal OFDM symbol a special flag starts and ends with it, this signal OFDM symbol goes through the whole receiver blocks (with modulation mode BPSK and coding rate $\frac{1}{2}$) except the descrambler as it is not scrambled at the transmitter and the signal OFDM symbol contains a Rate field (4 bits) to understand the packet

coding rate and modulation mode and it contains a data length field (12 bits) to know how many data symbols transmitted through this packet.

The final task of the Packet divider block is to give a special handshaking flag starts and ends with the data OFDM symbols related to the current received packet and deliver the data symbols with its handshaking flag to the next block which is the remove cyclic extension block.

3.3.1.1 Structure

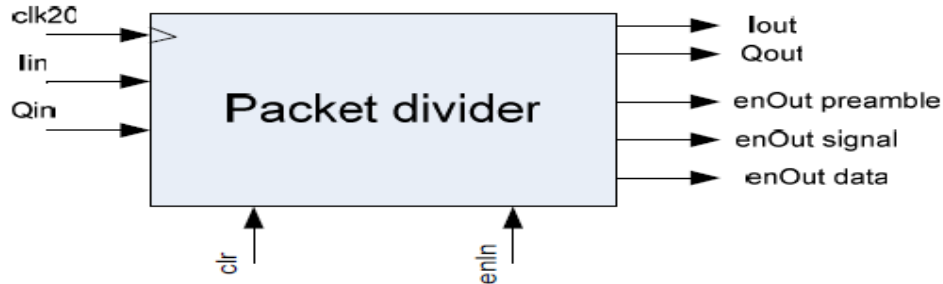


Figure 51

Inputs:

- En In: enable of the input data.
- CLR: to reset the block (active low).
- CLK_20: input clock to the block 20 mega Hz.
- I in: input in phase component.
- Q in: input quadrature component.

Outputs:

- En out Preamble: handshaking flag starts and ends with PLCP preambles.
- En out Signal: handshaking flag starts and ends with signal symbol.

- En out Data: handshaking flag starts and ends with Data symbols in the current input packet.
- I out: output in phase component.
- Q out: output quadrature component.

3.3.1.2 Operation:

1-the block counts 321 counts at first in the preamble case (4 OFDM symbols and one bit extra due to overlapping), after ending this counting the block steps into signal case. 2-the block counts another 80 counts in the signal case so the whole counts now = 401 then data case is the next one coming. 3-the block uses another counter (cnt80) to loop from 0 to 79, after each loop you have received a data symbol so the block increases number of received symbols by one. 4-the block keeps receiving data symbols until the input packet ends. 5-After finishing the data symbols the block is ready to a new packet.

Chapter 4

CHAPTER 4

Chapter4. Controller Implementation

4.1. Transmitter Controller Implementation

4.1.1 Introduction

To achieve this data rate for each block we have two solution : First solution : we can use the frequency synthesizer to achieve all this Clocks (6,9,12,16,18,20,24 Mbps) using counters but this solution is bad solution in FPGA as it will produce gated clocks which will have a large Skew delay Second solution : is based on how to minimize the number of frequencies and in the same time achieve the standard rate for OFDM system . we need clocks 6,9,12,16,18,20,24 Mbps so we will try to find the highest clocks which is divisible by this rates ,we used clocks 144, 72, 48, 60, 20, 16 MHz.

4.1.2. Data Rates

The following table includes the data rates required for each Block in the transmitter of OFDM according to the standard (802.11.a/b/g)

Modulation Mode	Rate	Rate of Scrambler		Rate of Encoder		Rate of Interleaver		Rate of Mapper		Rate of Framing		Rate of IFFT		Rate of Cyclic	
		in	out	in	out	in	out	in	out	in	out	in	out	in	out
Bpsk	1/2	6	6	6	12	12	12	12	12	12	16	16	16	16	20
Bpsk	3/4	9	9	9	12	12	12	12	12	12	16	16	16	16	20
Qpsk	1/2	12	12	12	24	24	24	24	12	12	16	16	16	16	20
Qpsk	3/4	18	18	18	24	24	24	24	12	12	16	16	16	16	20

Table 4. 1

4.1.3 Idea of Controller

Transmitter controller is the master of data flow in the transmitter and is represents the interface with the Mac layer as it hand checks with the Mac to feed the transmitter blocks with the required information about

the transmitted data to guarantee accurate flow of data with the required modulation mode and rate.

First, the Mac layer gives a start signal to the controller to inform it that there is data to be transmitted and gives it the number of data symbols, data length and the data rate. Second, the controller form the signal symbol containing the number of symbols, data length and the data rate then gives the modulation mode and data rate for all transmitter blocks and propagates this signal in the transmitter Finally, after waiting for one symbol the data propagates in the transmitter bocks with the required modulation mode and rate.

4.1.4. Structure

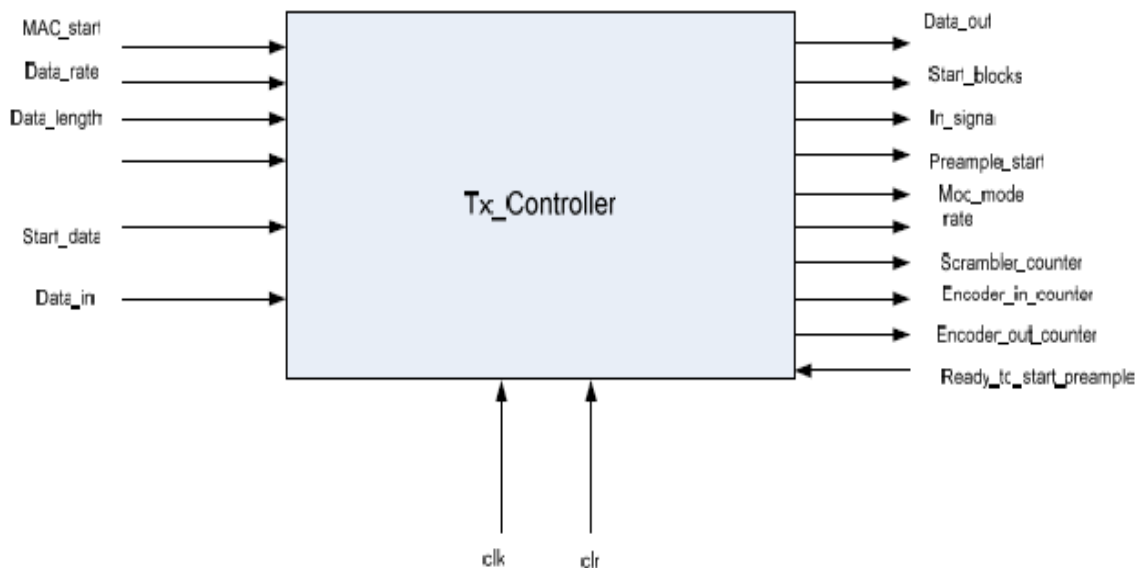


Figure 52

4.1.5 VHDL Description using Flow Chart

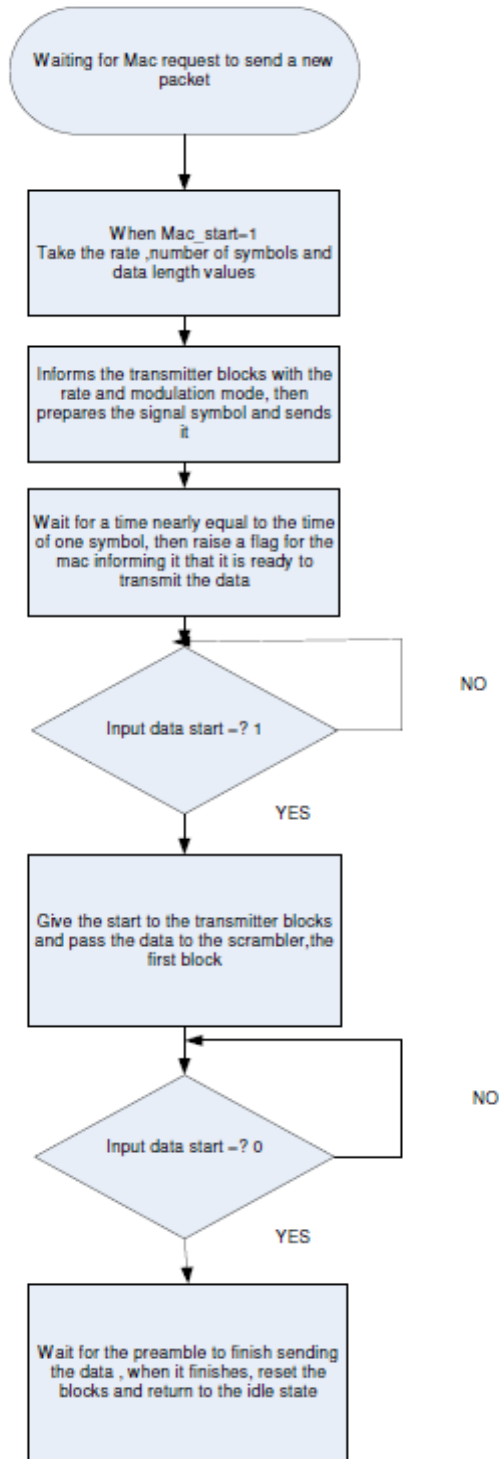


Figure 53

4.2. Receiver Controller Implementation

4.2.2 Introduction

The main function of the receiver controller is assuring that the signal symbol (BPSK and coding rate $\frac{1}{2}$) is well understood and decoded as it contains the modulation mode of the coming packet and the coding rate and the length of data samples related to this packet; so the blocks which depend on the modulation mode and the rate in their operation like demapper, de-interleaver, Viterbi decoder and descrambler will not work until the controller figure out the rate and modulation mode from the signal, so the data symbols are buffered in a FIFO buffer after the deframer until the signal pass all its way through the demapper, deinterleaver and viterbi decoder (BPSK and coding rate $\frac{1}{2}$) and it is not scrambled so it does not go through the descrambler, now the controller know the modulation mode and the coding rate of the packet and the data samples can be now released from the FIFO buffer and go through the rest of its way. The finite state machine which controls these steps consists of three states:

1. Idle state:

- The controller is in the signal mode so the deframer output is connected to the input of the demapper.
- The descrambler is not connected to the viterbi output (signal is not scrambled).
- Modulation mode is BPSK and coding rate is $\frac{1}{2}$.
- Waiting for the start of the output of the deframer and start a counter to calculate the signal period.
- At the end of the signal symbol coming out of the deframer jump to the next state to buffer data symbols until the signal is decoded.

2. Buffering data and decoding signal:

- Connect the output of the deframer to the input of the FIFO buffer to buffer the data symbols and give the FIFO buffer a write request.
- Wait for the start of the viterbi decoder output which is the decoded signal and from the first four bits get the rate and modulation mode, and also figure out the data length which is separated from the first four bits with a parity bit, the data length encoded in 12 bits and the total number is in bytes.
- After the Viterbi decoder finishes outputting the decoded signal the modulation mode and the coding rate of the packet is supported to the demapper, viterbi, deinterleaver and descrambler to be ready for the data mode.
- Jump to the next state to unbuffer data symbols from FIFO buffer and support them to the remaining blocks.

3. Unbuffering data and passing the output:

- In order to unbuffer data from the FIFO buffer the output of it is connected to the demapper input and a read request is given to the FIFO buffer.
- The Viterbi decoder is given a reset between the signal symbol and data symbols for proper operation.
- The output of the Viterbi decoder is connected to descrambler and the output of the descrambler is passed as the output of the receiver.
- If the output of the deframer ends the write request to the FIFO buffer ends too.

- when the output of the descrambler ends that means that the packet is finished so the controller reset all the receiver blocks and jump to the idle state to be ready for the next packet.

4.2.2 Structure

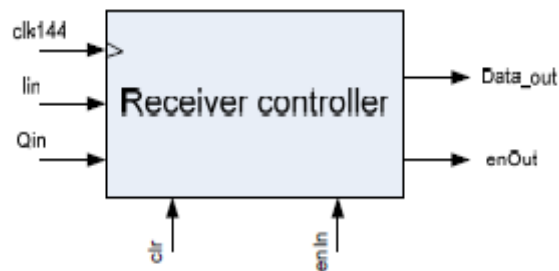


Figure 56

Inputs:

- En In: enable of the input data passed to the packet divider block.
- CLR: to reset the block (active low).
- CLK144: input clock to the block 144 mega Hz.
- I in: input in phase component.
- Q in: input quadrature component.

Outputs:

- En out: handshaking flag starts and ends with the block output.
- Data out: output data from the descrambler (class, 2013).

4.2.3 Post Layout Simulation

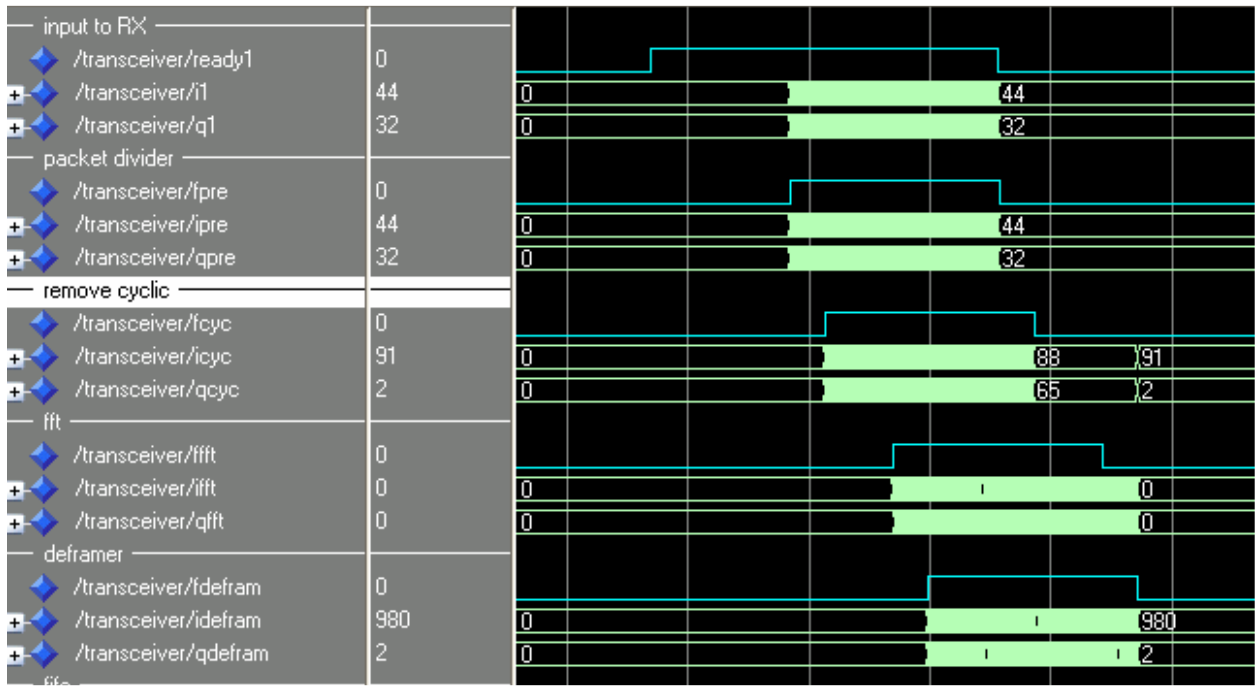


Figure 57 post layout simulation of receiver controller

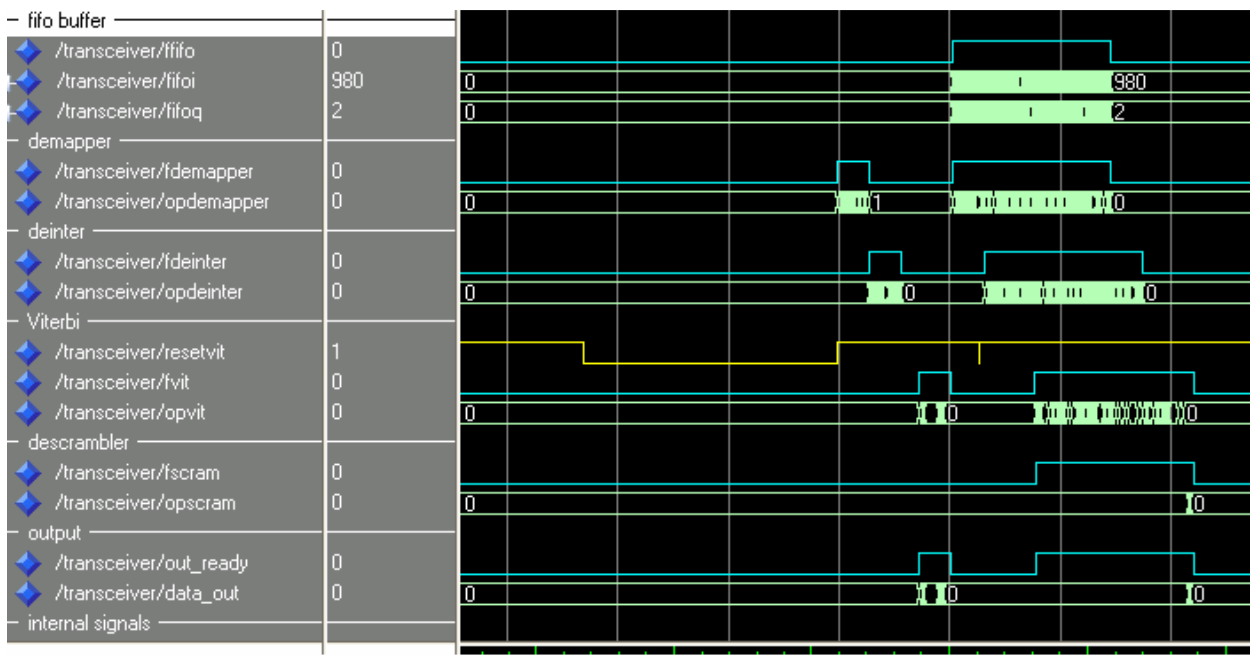


Figure 58

Chapter 5

Chapter 5. Synchronization

5.1 Timing Synchronization

- Timing estimation consists of two main tasks: packet synchronization and symbol synchronization.
- The IEEE 802.11 MAC protocol is essentially a random access network, so the receiver does not know exactly when a packet starts. The first task of the receiver is to detect the start of an incoming packet.

5.1.1 Packet Detection

- Packet detection is the task of finding an approximate estimate of the start of the preamble of an incoming data packet.
- In the packet detection test, it is required to assert whether a packet is present or not.
- The test is usually of the form that tests whether a decision variable exceeds a predefined threshold Th .
- If $mn < Th$ _ Packet not present
- If $mn \geq Th$ _ Packet present
 - The performance of the packet detection algorithm can be summarized with two probabilities:
 - a) Probability of detection PD (The probability of detecting a packet when it is truly present).
 - b) Probability of false alarm PFA (The probability that the test incorrectly decides that a packet is present, when actually there is none).
- Thus high PD is a desirable quality for the test and PFA should be as small as possible.
- In general, increasing PD increases PFA and decreasing PFA decreases PD, hence the algorithm designer must settle for some balanced compromise between the two conflicting goals.
- Generally it can be said that a false alarm is a less severe error than not detecting a packet at all.

- The reason is that after a false alarm, the receiver will try to synchronize to non-existent packet and will detect its error at the first received data integrity check.
- On the other hand, not detecting a packet always results in lost data.
- A false alarm can also result in lost data, in case an actual data packet starts during the time the receiver has not yet detected its mistake, the probability of this occurring is usually small.
- Thus, a little higher PFA can be tolerated to guarantee good PD.
- In Our implementation , 2 different packet detection algorithms are used:

5.1.1.1 Double Sliding Window Packet Detection

- The double sliding window packet detection algorithm calculates two consecutive sliding windows of the received energy.
- The decision variable m_n is the ratio between the total energy contained inside the first and second window. Figure 59 shows the windows A and B and the response of m_n to a received packet.

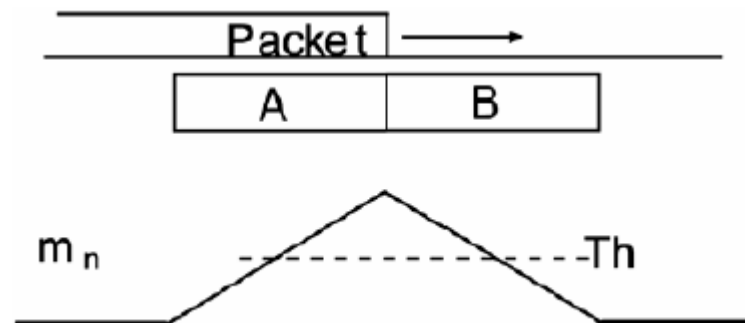


Figure 59 The response of the double sliding window packet detection algorithm

Thus, the decision variable m_n is given by:

$$m_n = \frac{a_n}{b_n}$$

$$\text{where: } a_n = \sum_{k=0}^{M-1} r_{n-k} r_{n-k}^* = \sum_{k=0}^{M-1} |r_{n-k}|^2$$

$$\text{and } b_n = \sum_{l=1}^L r_{n+l} r_{n+l}^* = \sum_{l=1}^L |r_{n+l}|^2$$

The advantage of this algorithm is that the value of m_n does not depend on the total received power, another benefit of this approach is that at the peak point of m_n , the value of m_n is related to the SNR as follows:

$$m_{peak} = \frac{a_{peak}}{b_{peak}} = \frac{S + N}{N} = \frac{S}{N} + 1$$

$$\widehat{SNR} = m_{peak} - 1$$

Our Implementation of Double Sliding Window Packet Detection

a) Block diagram

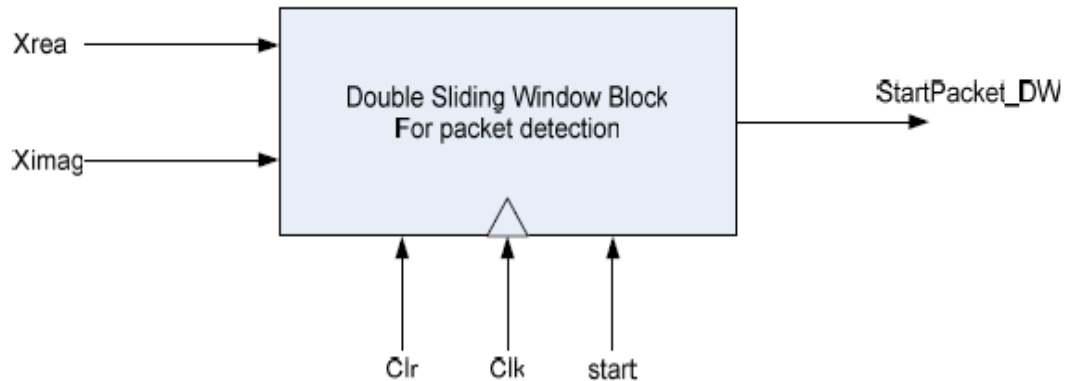


Figure 60

Where:

- `_ Xreal` , `Ximag` are the real and imaginary parts of the input sample.
- `_ Clk` : the operating clock of the block which equal 20 MHz ,the rate of the Incoming data samples.
- `_ start_pkt` : is high when a packet is detected.

b) How it works

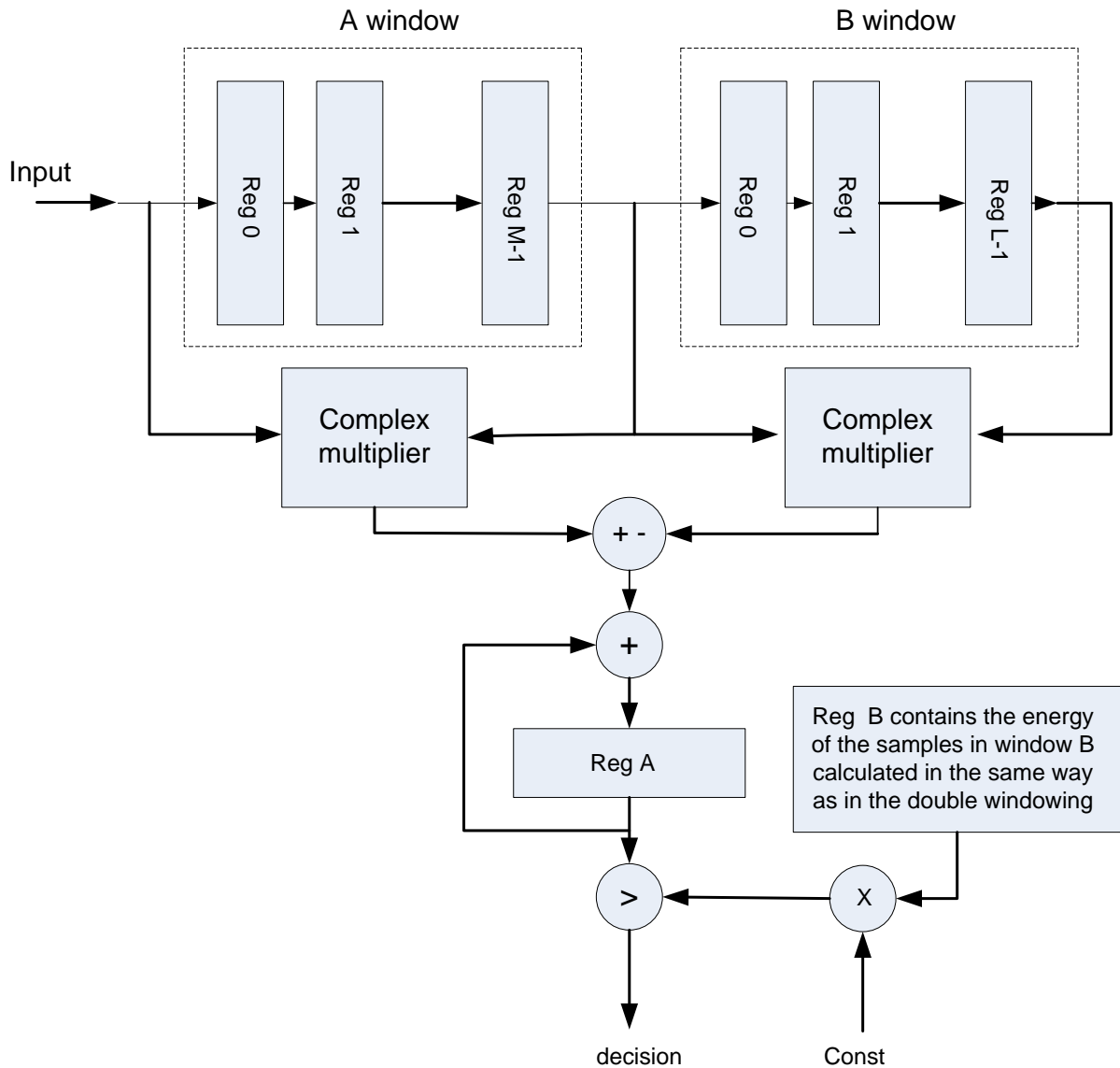


Figure 61

In designing the packet detection, the double sliding window is applied. As shown in the above figure ,after squaring the incoming sample ,two successive blocks of registers are used .The first consists of L registers ,while the other has M registers. The second block is fed from the end of the first block. Each time, the energy of each.

Block is calculated using only one subtractor that subtracts the value of the last register inside the block from the value of the first register input. The subtraction output is accumulated using two dedicated registers (Reg A and Reg B), the second block energy (represented in Reg B) is multiplied by a

constant that is required threshold .Then the output from the multiplication is compared to the energy of the first block (represented in Reg A). The output of the comparison is actually the true decision.

In our design: we take the size of the windows = 16

c) Simulation results

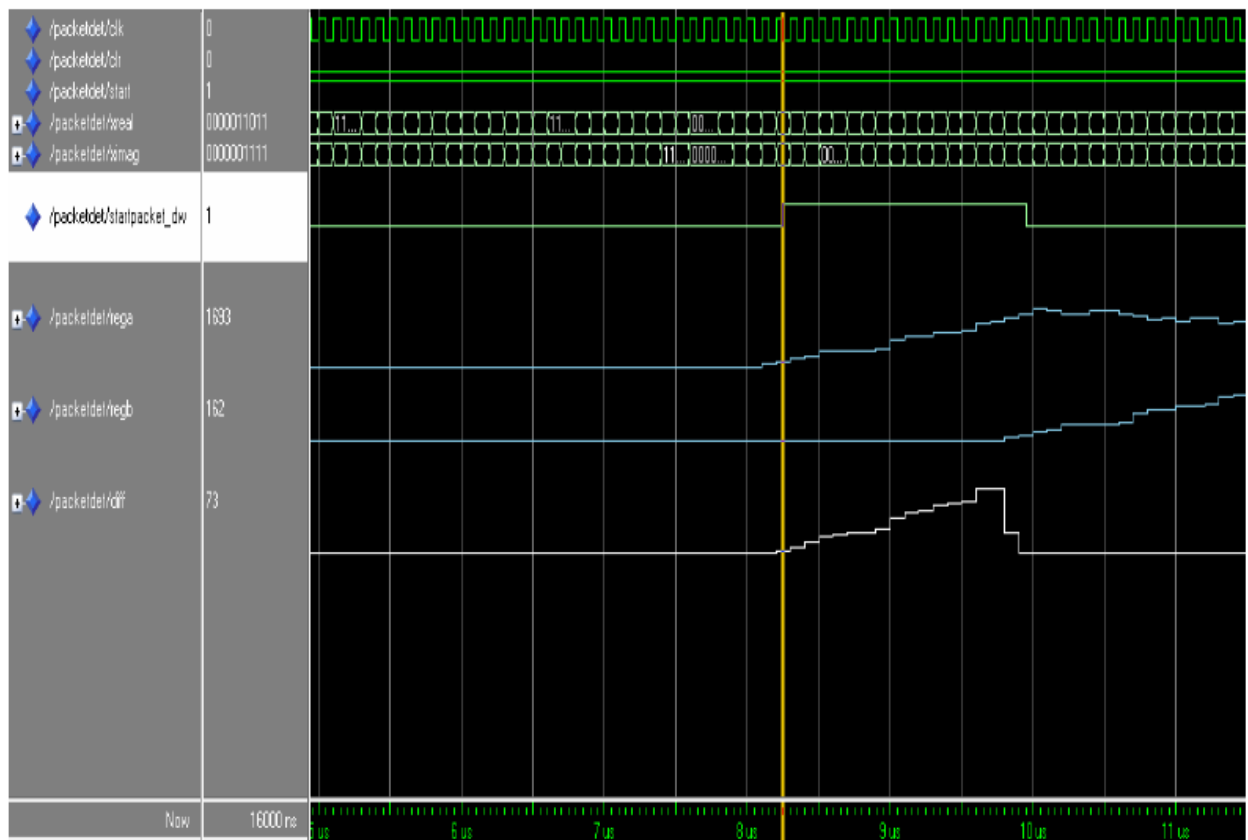


Figure 62

Simulation of double windowing algorithm, the packet starts at 8 us
 Choosing the appropriate threshold:

To choose the threshold, we make a Matlab simulations on the 802,11a model,
 We simulated the double windowing algorithm with different thresholds and got the probability of correct detection in different signal to noise ratios.

The following table illustrate the probability of correct detection for different thresholds and SNRs.

SNR Threshold	10	11	12	13	14	15	16	17	18	19	20
4	0.01	0.02	0.04	0.03	0.04	0.01	0.03	0.04	0.02	0.03	0.02
5	0.09	0.11	0.14	0.11	0.12	0.12	0.1	0.09	0.09	0.1	0.08
6	0.32	0.26	0.24	0.28	0.26	0.29	0.21	0.18	0.24	0.2	0.19
7	0.5	0.43	0.47	0.42	0.45	0.46	0.48	0.43	0.39	0.41	0.4
8	0.67	0.66	0.65	0.58	0.62	0.59	0.63	0.61	0.58	0.57	0.54
9	0.72	0.81	0.73	0.72	0.77	0.74	0.74	0.71	0.69	0.71	0.71
10	0.71	0.84	0.77	0.84	0.86	0.85	0.83	0.83	0.81	0.82	0.81
11	0.61	0.8	0.8	0.84	0.88	0.89	0.85	0.87	0.88	0.87	0.87
12	0.45	0.67	0.7	0.86	0.84	0.92	0.94	0.92	0.93	0.91	0.91
13	0.26	0.51	0.62	0.75	0.86	0.91	0.94	0.93	0.96	0.95	0.96
14	0.12	0.35	0.5	0.64	0.73	0.89	0.92	0.98	0.98	0.98	0.98
15	0.1	0.13	0.27	0.48	0.58	0.75	0.82	0.96	0.99	0.99	0.99
16	0.06	0.04	0.15	0.32	0.45	0.58	0.74	0.88	0.93	0.95	1
17	0.01	0	0.09	0.17	0.26	0.39	0.61	0.71	0.88	0.92	0.98
18	0	0	0.03	0.07	0.16	0.19	0.48	0.58	0.77	0.86	0.91
19	0	0	0	0.02	0.06	0.14	0.24	0.44	0.58	0.76	0.9
20	0	0	0	0	0.02	0.06	0.14	0.3	0.41	0.59	0.81
21	0	0	0	0	0	0.01	0.04	0.13	0.24	0.38	0.7

Table 5. 1 illustration of the probability of correct detection for different

We take threshold=10 as it gives a nearly equal probabilities of correct detection for different SNRs.

At SNR=10dB, the thresholds vs. the probability of correct detection is:

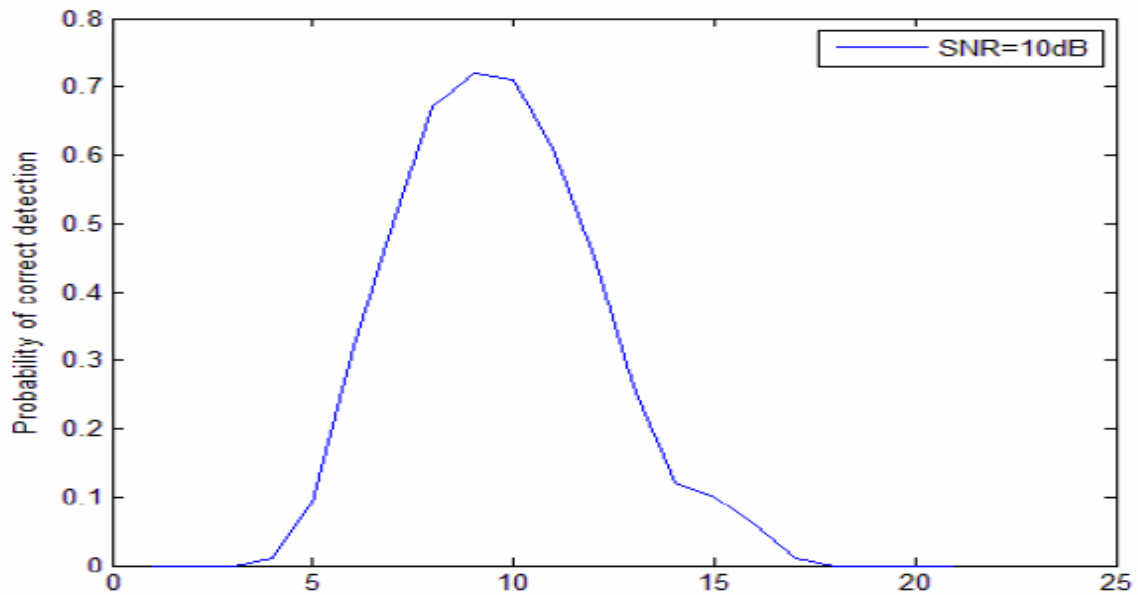


Figure 63

5.1.1.2. Using the Structure of the Preamble for Packet Detection: (Delay and Correlate)

A general communications system engineering principle is that the receiver should use all the available information to its advantage. In the packet detection algorithm, this means that the known structure of the preamble should be incorporated into the algorithm.

The structure of the WLAN preamble enables the receiver to use a very simple and efficient algorithm to detect the packet; it takes advantage of the periodicity of the short training symbols at the start of the preamble. This approach is called the delay and correlate algorithm.

Figure 64: shows the signal flow structure of the delay and correlate algorithm.

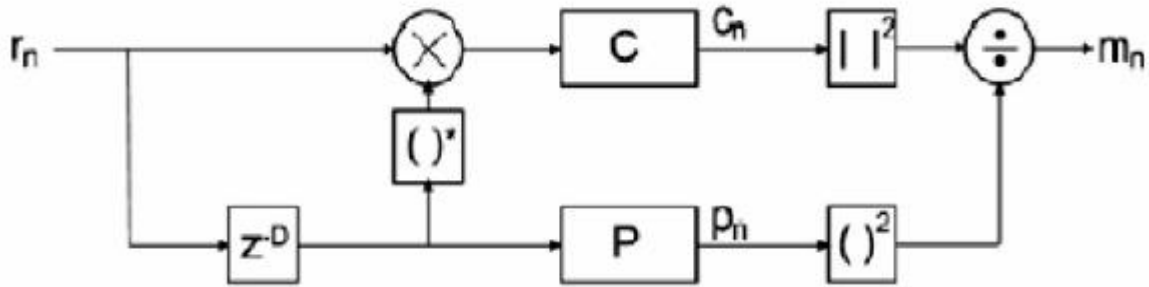


Figure 64 Signal flow structure of the delay and correlate algorithm

The Figure shows two sliding windows C and P . The C window is a cross-correlation between the received signal and a delayed version of the received signal, hence the name delay and correlate. The delay z^{-D} is equal to the period of the start of the preamble; for IEEE 802.11a: $D = 16$, the period of the short training symbols.

The P window calculates the received signal energy during the cross-correlation window.

The value of the P window is used to normalize the decision statistic, so that it is not dependent on absolute received power level.

Thus, the decision variable m_n is given by:

$$m_n = \frac{|c_n|^2}{(p_n)^2};$$

$$c_n = \sum_{k=0}^{L-1} r_{n+k} r_{n+k+D}^*$$

$$p_n = \sum_{k=0}^{L-1} r_{n+k+D} r_{n+k+D}^* = \sum_{k=0}^{L-1} |r_{n+k+D}|^2$$

When the received signal consists of only noise, the output c_n of the delayed crosscorrelation is zero-mean random variable, since the cross-correlation of noise samples is zero, thus m_n will be low.

Once the start of the packet is received, cn is a cross-correlation of the identical short training symbols, thus mn increases quickly to its maximum value thus exceeding the threshold. This gives a good estimate of the start of the packet.

Our implementation of using the Structure of the Preamble for Packet Detection: (Delay and Correlate)

a) Block diagram

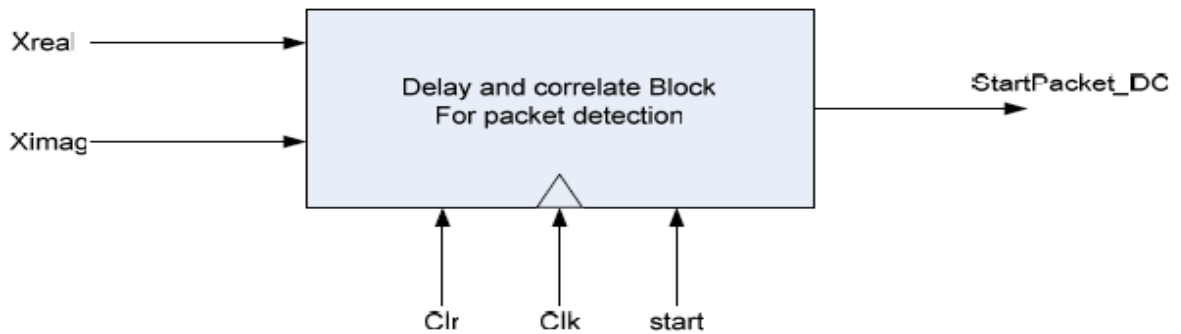


Figure 65

b) How it works

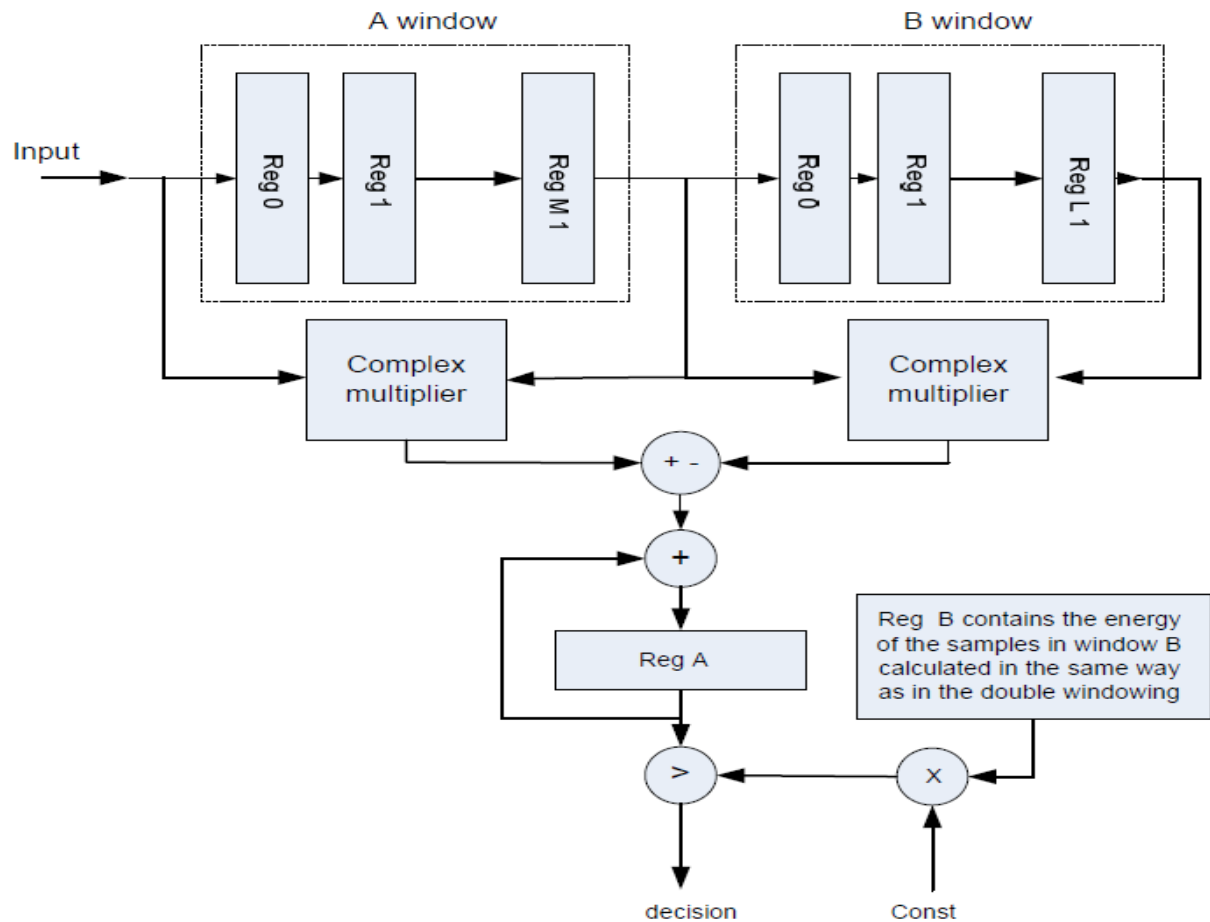


Figure 66

In the delay and correlate, as shown in the above figure ,two successive blocks of registers are used .The first consists of L registers ,while the other has M registers .

The second block is fed from the end of the first block. Each time, the correlation between the samples int the two registers are calculated using two complex multipliers and only one subtractor that subtracts the value of the multiplication result of the last samples from the multiplication result of the first samples in the two registers . The subtraction output is accumulated using one dedicated registers (Reg A) while RegB contains the accumulated

energy of register B as in the dounble windowing algorithm , the value in Reg B is multiplied by a constant that is the required threshold .Then the

output from the multiplication is compared to the energy of the first block (represented in Reg A).The output of the comparison is actually the true decision.

In our design: we take the size of the windows = 16

c) Simulation results

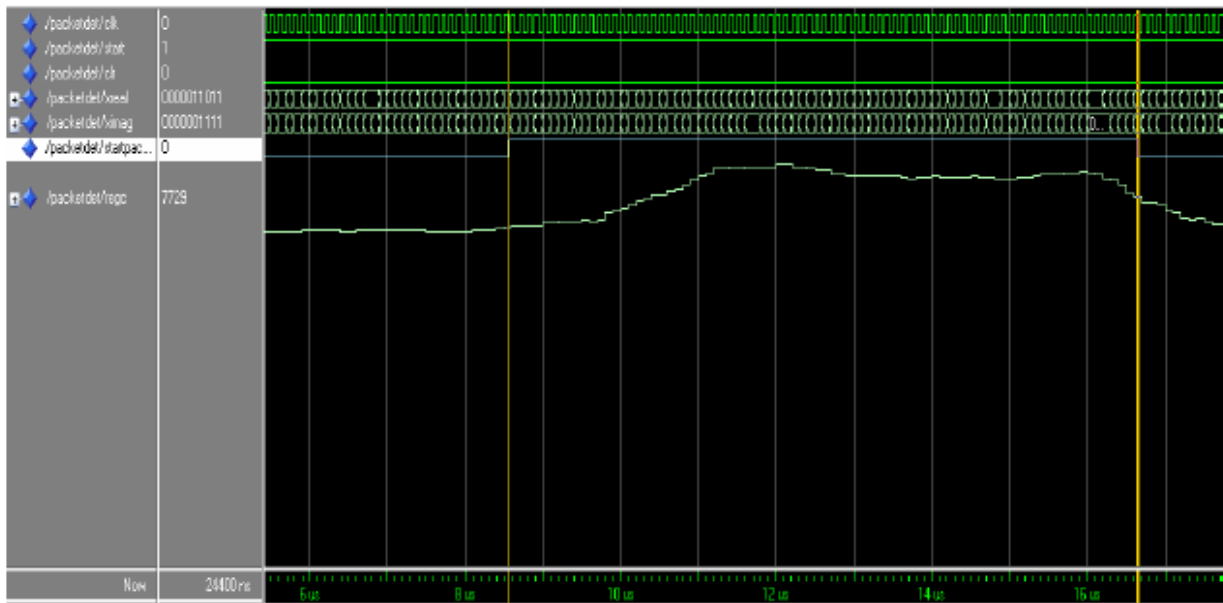


Figure 67

Simulations of the delay and correlate algorithm implementation , in simulation the short training sequence -80 sample - is input with SNR = 10 dB from the time 8 us to 16us,we note that the start_dc is high during this period.

Conclusion

The packet detection block is a combination of the two previous algorithms for better probability of correct detection, we take the (Oring) of the two alarms.

The packet detection block performance

Logic utilization: 5%

Combinational ALUTs: 1,05/48,352(2%)

Dedicated logic registers: 1,58/48,352(3%)

Total block memory bits: 0%

Max. Frequency: 74.22 MHz.

5.1.2 Symbol Timing

Symbol timing refers to the task of finding the precise moment of when individual OFDM symbols start and end.

The symbol timing result defines the DFT window; i.e., the set of samples used to calculate DFT of each received OFDM symbol. The DFT result is then used to demodulate the subcarriers of the symbol.

Packet detection provides an estimate of the start of the packet, and then the symbol timing algorithm uses a simple cross-correlation based algorithm to estimate the start of OFDM symbols.

The algorithm calculates the cross-correlation between the received signal r_n and a known reference t_k such as the end of the short training symbols and the start of the long training symbols.

Our implementation of Symbol Timing:

a) Block diagram

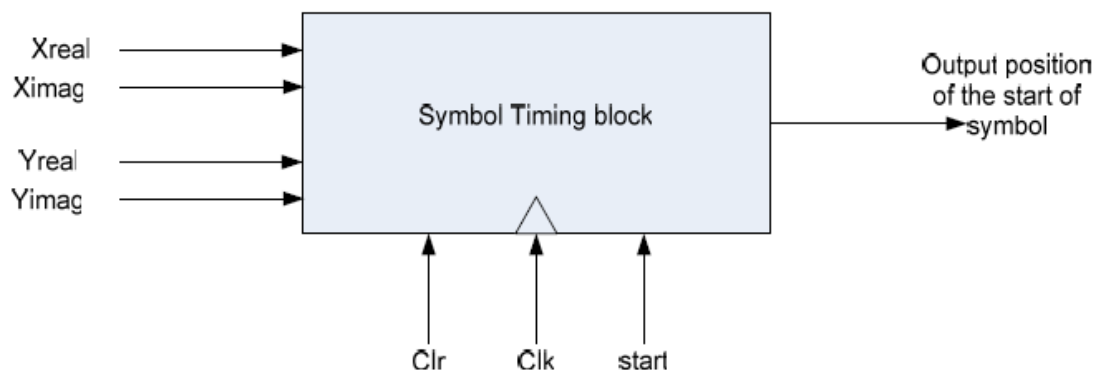


Figure 68

b) How it works

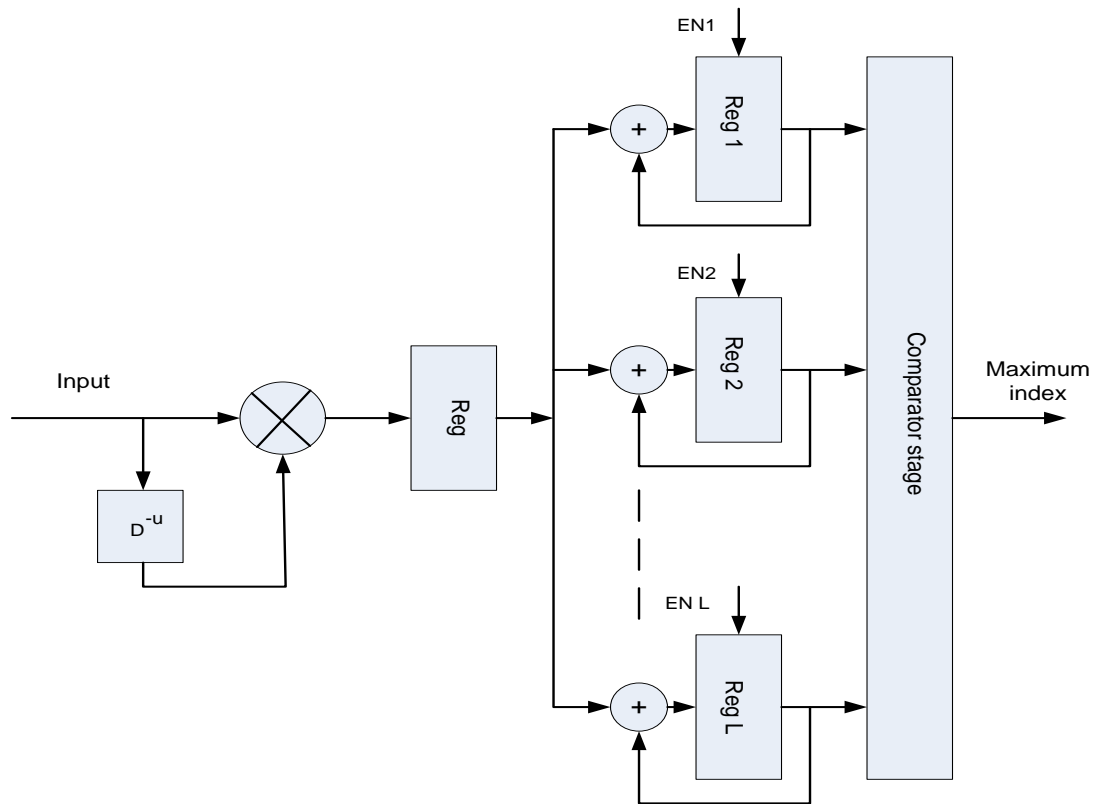


Figure 69

As shown in the figure, the proposed hardware consists of a multiplier, Ladders, L+1 registers, and one comparator stage. Two samples that have a delay of $U = \text{FFT length} - \text{CP length} = 16$

In between are multiplied together by a complex multiplier. Then the output is registered as a pipeline stage. The registered output is accumulated in L adders that are fired in sequence. The first adder is register, reg1 is enabled in the first clock cycle while the second register, reg2 is enabled at the second clock cycle and so on. After L clock cycles, all registers work. At the end of the cyclic prefix, the reverse order for enabling the registers is done. The first register is halted first and the second is halted in the following clock cycle and so on. After halting all the registers, the stored values are compared through the comparator stage. This comparator stage is a serial one; it accepts one input per clock and

performs the comparison then generates the index of the register that has the maximum value.

Note that the index corresponds to the register, reg1, is not the start of the packet generated from the packet detection block ,Actually , a small margin of safety is used, the packet start is assumed 5 samples advanced with respect to the start generated from the packet detection. In our design , we take L=10

c) Simulation results

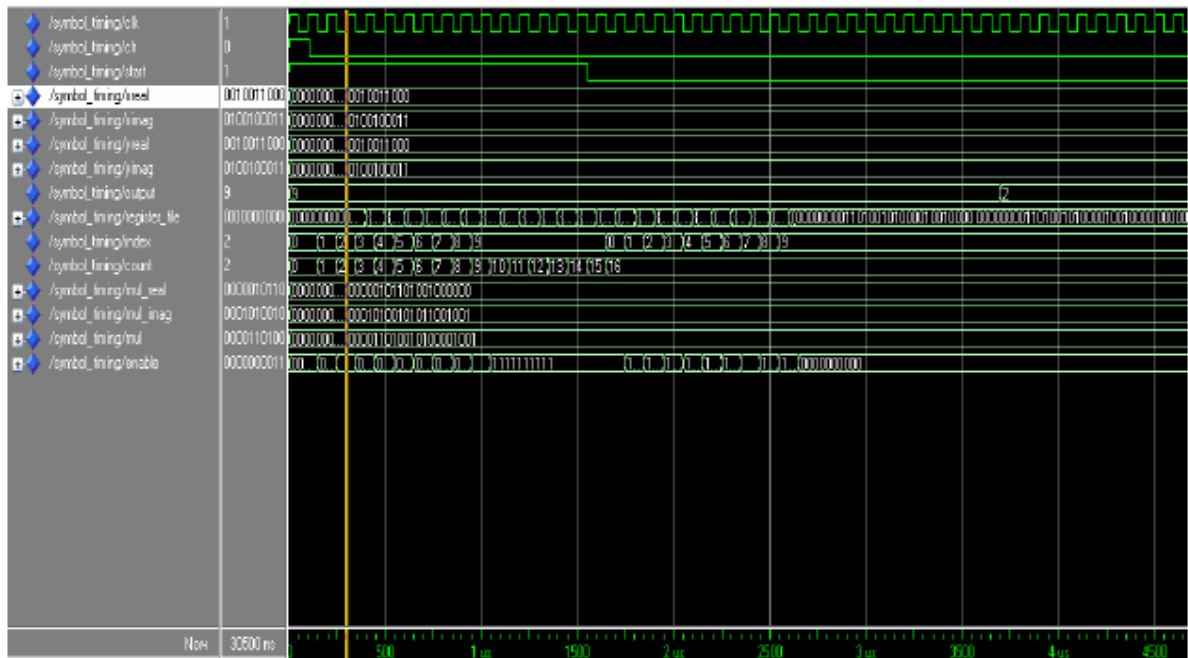


Figure 70

Simulation of the symbol timing algorithm, the data is input at the second instant and the algorithm successfully found that.

d) Block performance

Logic utilization: 2%
 Combinational ALUTs: 978/48,352(2%)
 Dedicated logic registers: 374/48,352(<1%)
 Total block memory bits: 0%
 Max. Frequency: 165.73 MHz.

5.2 Residual Phase Error Correction Using CORDIC

Algorithm

5.2.1 Introduction

OFDM signals are very much sensitive to the synchronizer performance, mainly because the different sub-carriers overlap their respective spectra. The synchronizer is the block responsible for detecting the incoming frame and to estimate and correct for possible frequency offsets. It also identifies the starting point from which on the different OFDM symbols will be fed into the FFT block. To carry out all the operations mentioned above, the standard IEEE 802.11a defines the so-called preamble symbols, which have a very specific structure to simplify the estimation procedures.

Suitable synchronizer architecture for this standard was presented by the authors in. Nevertheless, the synchronizer cannot compensate for some undesirable effects prompted by the RF down conversion and the analog-to-digital converters. Furthermore, the synchronizer itself will introduce some estimation errors. All these impairments are visible inside an OFDM symbol as a linear phase.

Pilot sub-carriers inside an OFDM symbol may help to easily estimate the remaining linear phase if they were used during the channel estimation, because this phase would be seen as a part of the channel itself. For the IEEE 802.11a standard, the pilots are not intended for channel estimation, i.e. the remaining linear phases should be explicitly estimated using the pilots that is called residual phase correction mechanism for IEEE 802.11a standard.

According to the IEEE 802.11a standard, the residual phase correction is carried out assuming that the residual phase is constant due to the non-perfect carrier frequency offset (CFO) synchronization and timing correction.

This phase could be corrected using pilots in each frame coming from the FFT block after the channel estimation process is done. The residual phase extraction technique uses the summation of pilots' subcarriers to minimize the noise effect in each subcarrier, this summation inputs to the

phase extraction block according to **CORDIC** algorithm using its vectoring mode to extract the rotating phase from the pilots. The following block diagram illustrates the flow of the synchronization process.

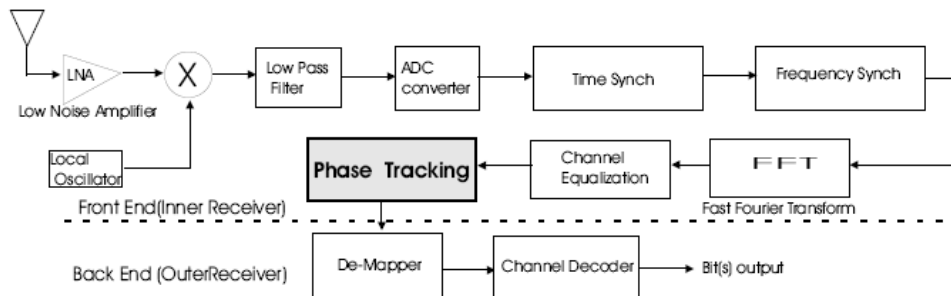


Figure 71 OFDM receiver front end and synchronization architecture

5.2.2 Block Diagram for Our phase Correction Implementation

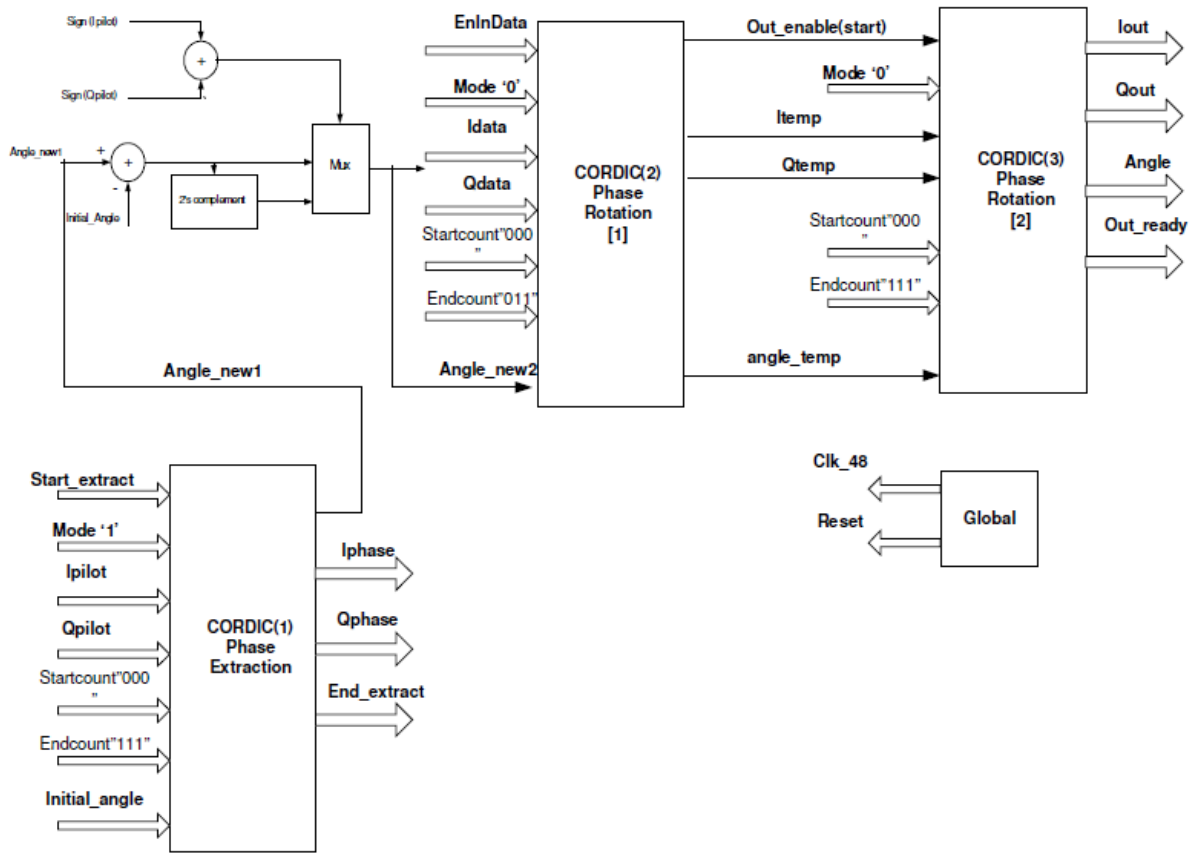


Figure 72 Block Diagram for the phase correction Implementation using CORDIC

As shown before in the block diagram, we use the pipelining concept to support the required rate coming from the deframer block, as the data input to the phase correction comes with 12 Mbps and our **CORDIC** design needs 8 clock cycles for each symbol to perform its iterative operation to rotate the incoming symbol with the required phase, so we use double **CORDIC** blocks with 48 Mega clock for that reason.

5.2.3 Flow Chart for Our Design Implementation

The following flow chart shows the hardware implementation algorithm for this approach (class, 2013).

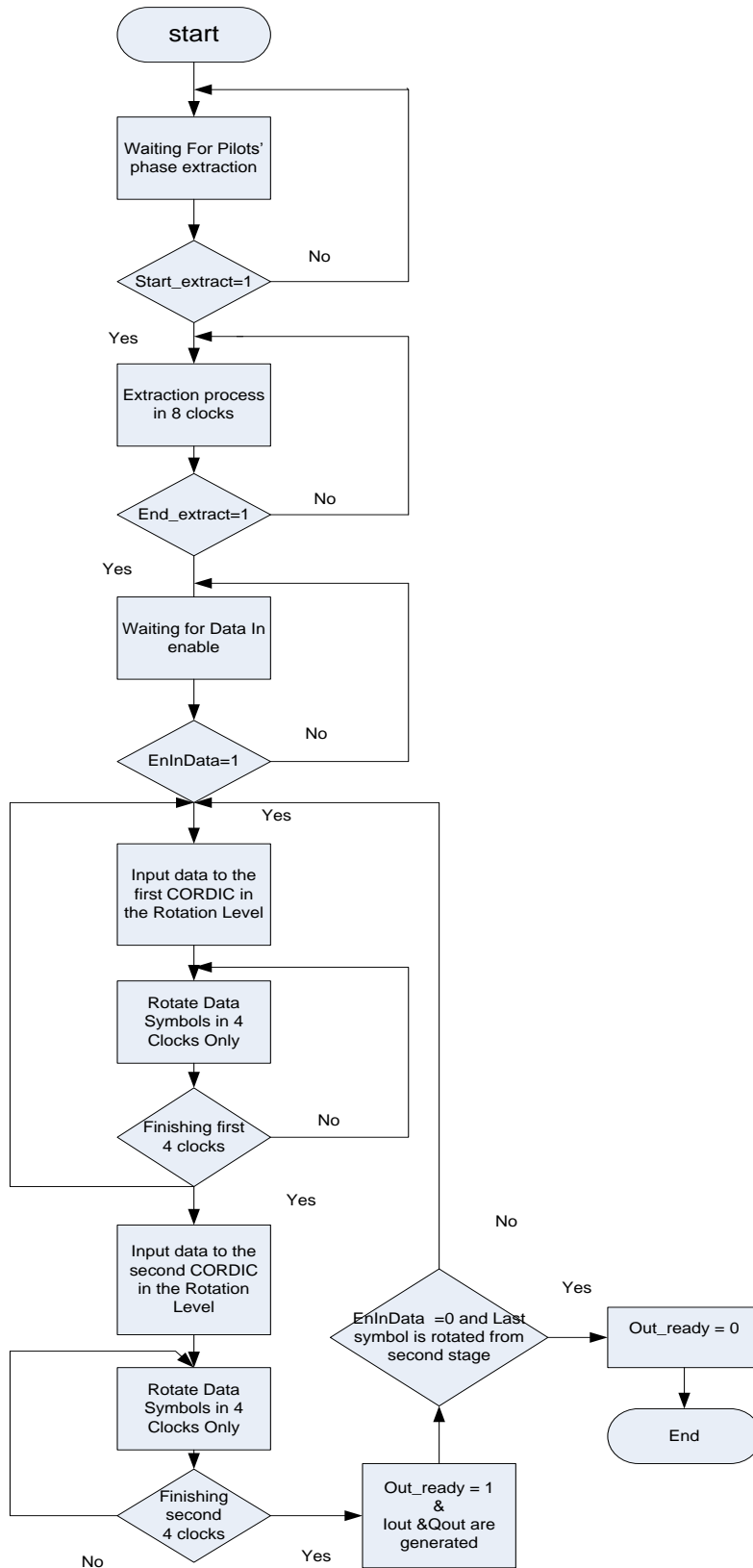


Figure 73

5.2.4 Simulation for the Phase Correction Implementation

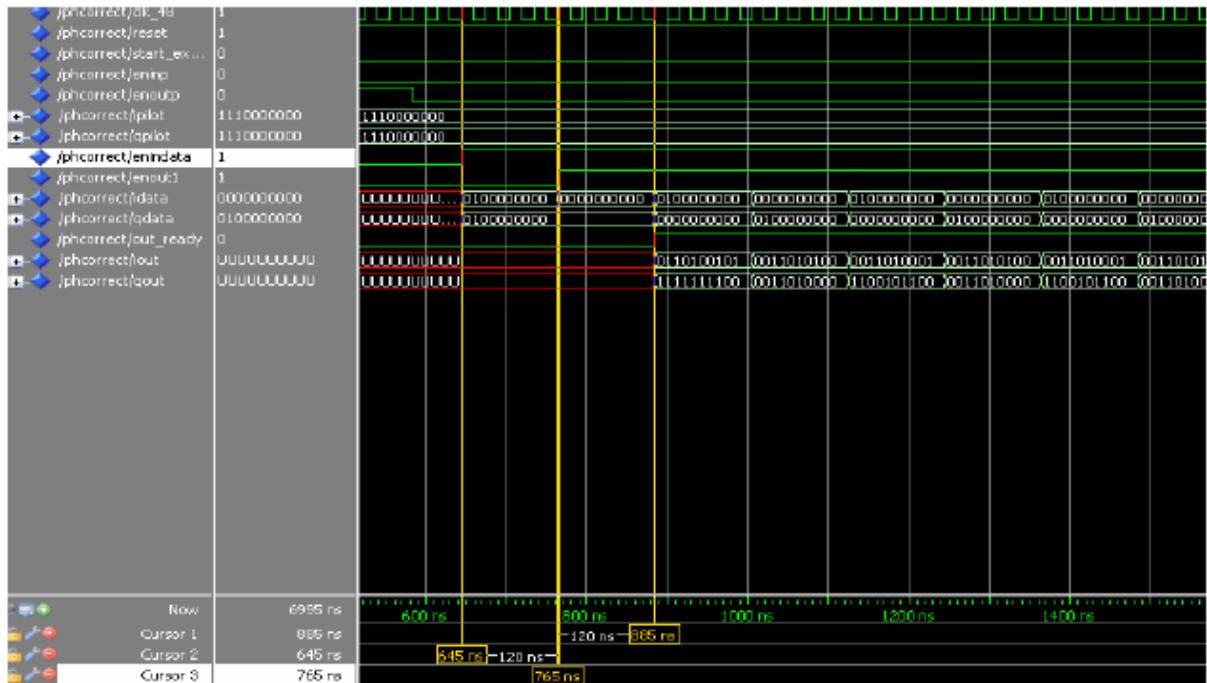


Figure 5.14

As shown in the figure the rotation process for the incoming data after the extraction of the phase from the pilots, the first **CORDIC** blocks rotates the symbols for only 4 iteration (4 clock cycles) after finishing them the second **CORDIC** starts its work and continues the rotation process for the remaining 4 iterations (4 clock cycles), during that the first **CORDIC** is working with the first 4 iterations for the next symbol to support the required rate (pipelining concept) and etc....

Note that, the incoming rate for the phase correction block from the deframer block is (12 Mbps) and the **CORDIC** block works with clock (48 Mega) [4 iteration means 120ns from the used clock]

5.2.5 Design performance

Logic utilization	< 1 %
Combinational ALUTs	321 / 48,352 (< 1 %)
Dedicated logic registers	181 / 48,352 (< 1 %)
Total registers	181
Total block memory bits	0 / 2,544,192 (0 %)

Chapter 6

Chapter 6. System Implementation

6.1 Design Flow using VHDL

The diagram below summarizes the high level design flow for an ASIC (ie. gate array, standard cell) or FPGA. In a practical design situation, each step described in the following sections may be split into several smaller steps, and parts of the design flow will be iterated as errors are uncovered.

6.1.1 System-level Verification

As a first step, VHDL may be used to model and simulate aspects of the complete system containing one or more devices. This may be a fully functional description of the system allowing the FPGA/ASIC specification to be validated prior to commencing detailed design. Alternatively, this may be a partial description that abstracts certain properties of the system, such as a performance model to detect system performance bottle-necks.

6.1.2 RTL design and test-bench creation

Once the overall system architecture and partitioning is stable, the detailed design of each FPGA/ASIC can commence. This starts by capturing the design in VHDL at the register transfer level, and capturing a set of test cases in VHDL. These two tasks are complementary, and are sometimes performed by different design teams in isolation to ensure that the specification is correctly interpreted. The RTL VHDL should be synthesizable if automatic logic synthesis is to be used. Test case generation is a major task that requires a disciplined approach and much engineering ingenuity: the quality of the final FPGA/ASIC depends on the coverage of these test cases.

6.1.3 RTL verification (70% of design time at RTL)

The RTL VHDL is then simulated to validate the functionality against the specification. RTL simulation is usually one or two orders of magnitude faster than gate level simulation, and experience has shown that this speed-up is best exploited by doing more simulation, not spending less time on simulation.

In practice it is common to spend 70-80% of the design cycle writing and simulating VHDL at and above the register transfer level, and 20-30% of the time synthesizing and verifying the gates.

6.1.4 Look-ahead Synthesis

Although some exploratory synthesis will be done early on in the design process, to provide accurate speed and area data to aid in the evaluation of architectural decisions and to check the engineer's understanding of how the VHDL will be synthesized, the main synthesis production run is deferred until functional simulation is complete. It is pointless to invest a lot of time and effort in synthesis until the functionality of the design is validated.

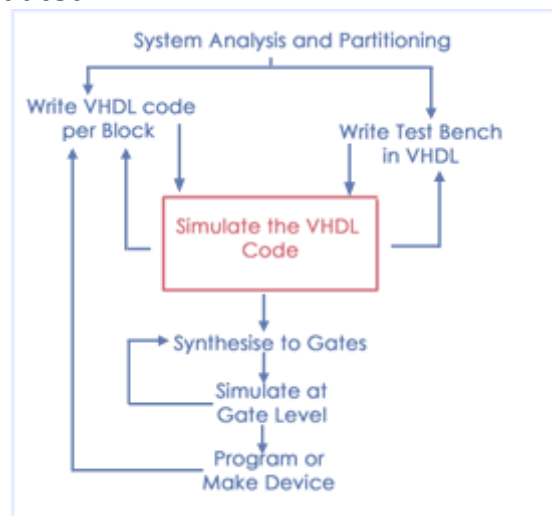


Figure 74

6.2 Quartus II

The Altera® Quartus® II design software provides a complete design environment that compiles, synthesis, fit, and perform timing analysis for the proposed HDL designs. It can also be used to download, test, debug, and verify the final design on Altera® FPGA's. The Quartus® II software provides many useful tools that can be used to optimize the design and get the best out of it.

Altera® Quartus® II software generates the simulation files that can be passed to Modelsim to perform Post Layout Simulations which gives in

depth knowledge of the real system performance and debug it if needed. Also, it contains many other tools that help testing and verifying the design even after downloading on chip. We will talk in some details about the Quartus II tools to help improvements in the project later.

- In the coming few pages we will quickly describe some steps to deal with Quartus II software.

6.2.1 Creating project:

- 1) **Open** Quartus II program.
- 2) **Choose** New Project Wizard (File menu). The new project wizard appears.
- 3) **Type the directory** name in the working directory box, or select the directory with Browse (...).
- 4) Type a name for the project in the project name box.
- 5) **Click next**; the second page of the New Project Wizard appears.
- 6) **Add all** and use **the up button** to put the top hierarchy at first and you have to add the VHD files of your design to use Quartus II as a synthesizer.
- 7) Click **next**; the third page of the **New Project Wizard** appears as shown below. Here you can **choose** the device family which you will download your design on >> **choose** (pin count, Speed grade) from the kit which will use it. We use **Stratix III EP3SL150F1152C2**.

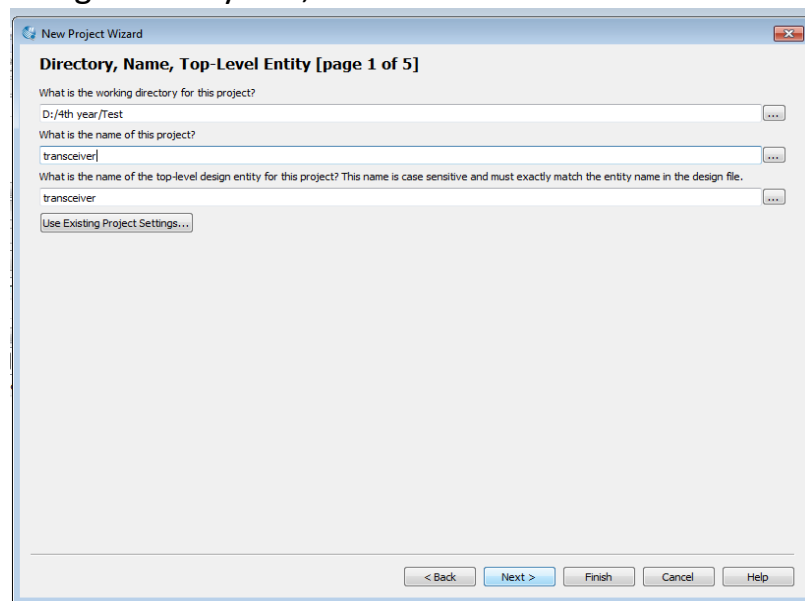


Figure 75

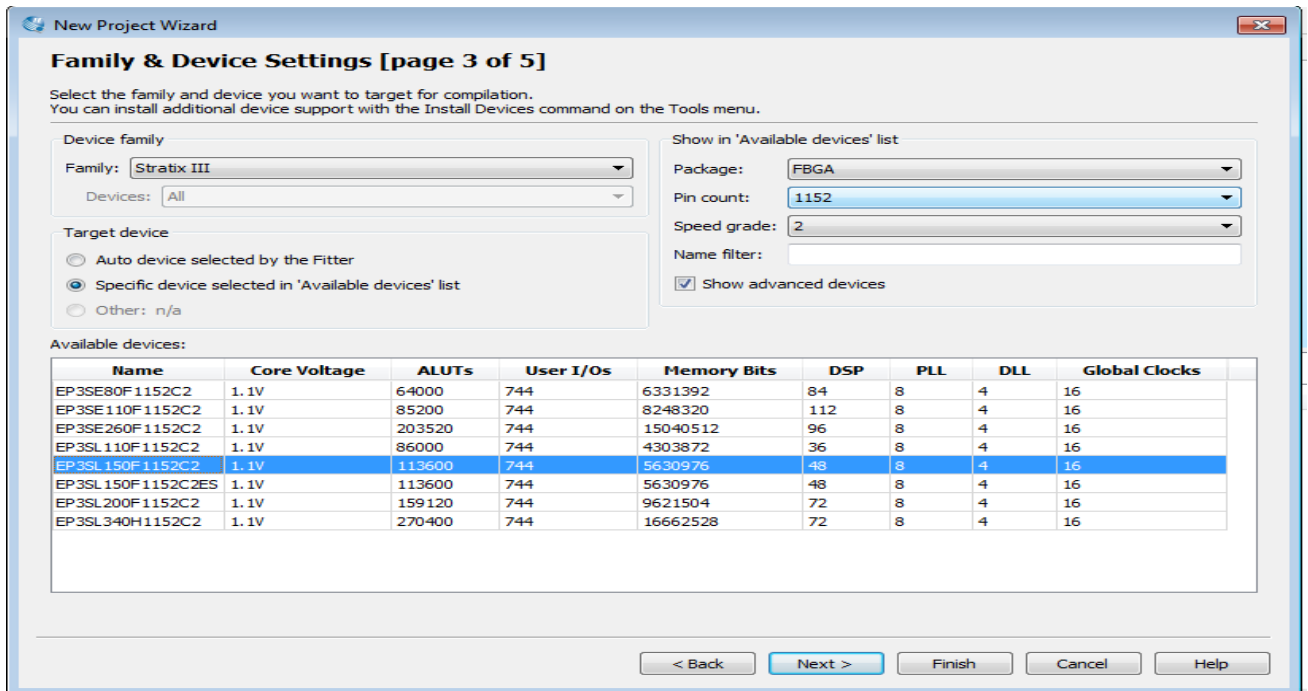


Figure 76

8) Click: next>> next >> finish.

6.2.2 Managing Timing Constraints:

It Views basic information about your project in the Project Navigator, Report panel, and Messages window.

Apply appropriate timing constraints to correctly optimize fitting and analyze timing for your design. The fitter optimizes the placement of logic in the device to meet your specified timing and routing constraints.

Without defined critical warning will appear in message box at critical warning as shown:

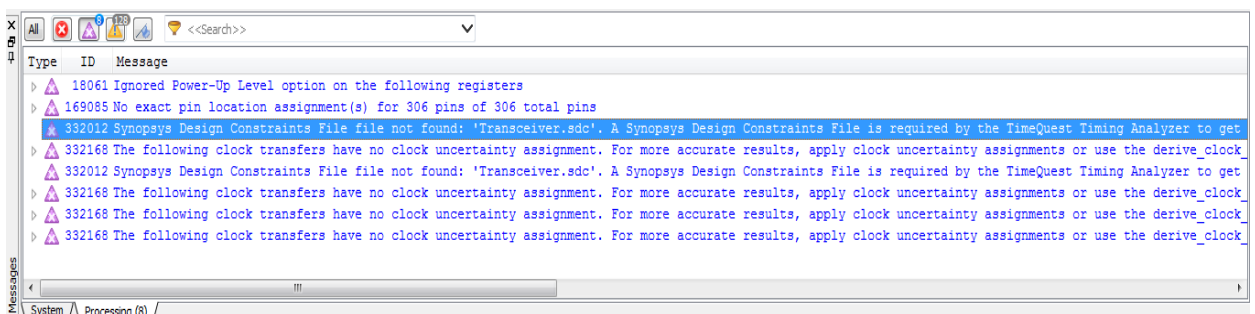


Figure 77 Warning of Constrain file

To specify it we have 2 ways:

First:

1-from tools >> TimeQuest Timing Analyzer >>File >> new SDC file .

For example:

create_clock -name {clk_master16} -period 62.500 -waveform {0.000 31.250}

This clock with frequency =16mega, negative clk , rising edge at 0.00 n falling edge at 31.250 n

2-Specify constraints for clock characteristics, timing exceptions, and external signal setup and hold times before running analysis. TimeQuest reports the detailed information about the performance of your design compared with constraints in the Compilation Report panel.

3-Save the constraints you specify in the GUI in Synopsys Design Constraints File (design name. sdc).

Second :

1- From Compilation Report of your design >> TimeQuest>>Clocks

2- Right click on the clock which appear in the table >> choose Edit Clock Constrains as shown :

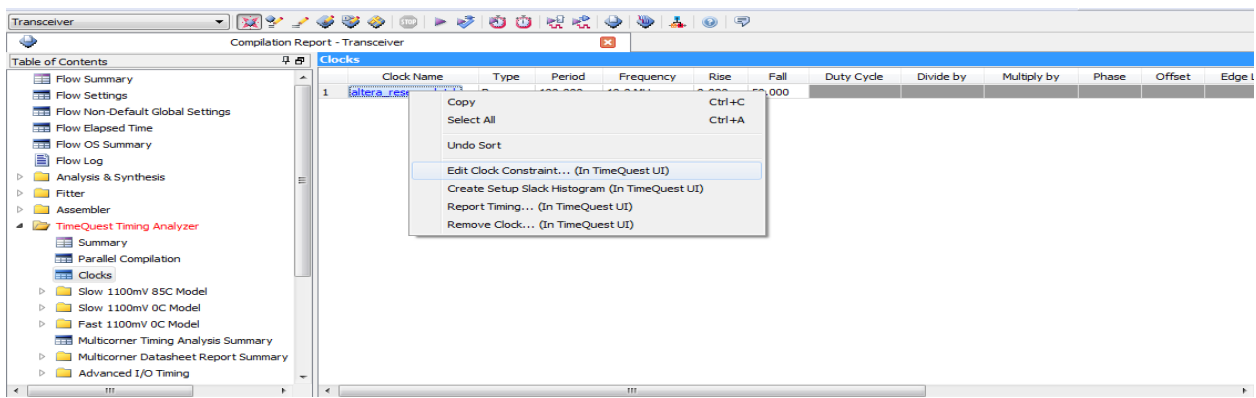


Figure 78 Compilation Report Window

- 3- Write clock name as in your design then define its period as shown in figure below, it will be created new SDC command in the TCL file.
- 4- Creat all clocks in your design >> close the Timing Analyzing widow >> save.

6.2.3 Customizing Internal Memory IP Cores

As long as OFDM contains memories which are used in as RAM or RAM or FIFO. So it should be defined it

We have two type of used IP Cores in the project:

- 1- Altera **megafunction** IP cores such as RAM: 2-Port—Dual-port RAMS.
- 2- **OpenCores** like CFFT Core done by ZHAO Ming and available in the website opencores.org

So each type has different way to include memory in the project.

Steps of including IP for Altera megafunction IP cores

1-Tools >>Choose >>Mega Wizard plug-In manager, for creating memory ram with 2port input and output.

2- **Choose** >>Create a new custom function Megafunction.

3-choose type we want as shown and write the name of file.vhd in the directory as shown.

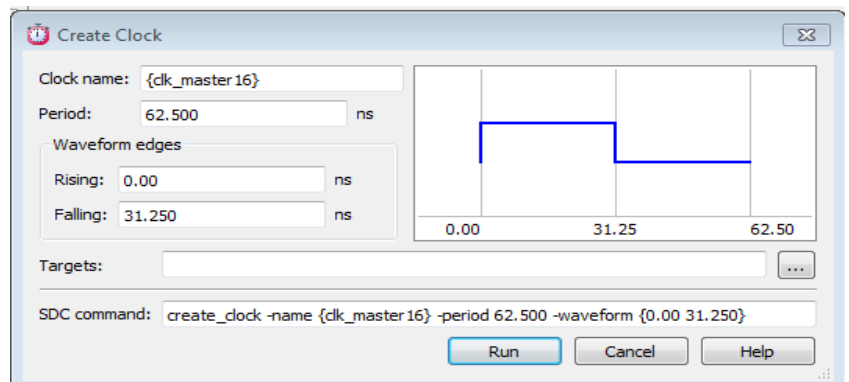


Figure 79 Creating Clock

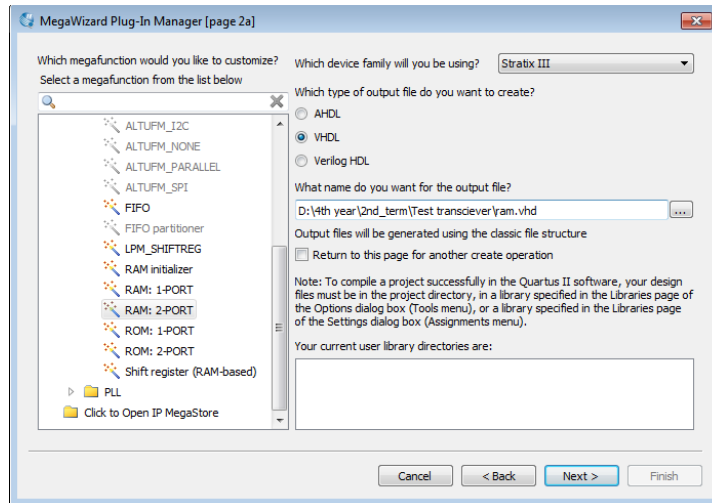


Figure 80

4-Choose for Specifying input size as a number of bits.

5-**Define** >> number of bits memory to combatable to memory ports you want, for example:

data bus=20 , waddress=6 so num. of bits = $2^6 * 20 = 1280$ bits, **check** >> Auto.

6- **Check** enable if it exist in your memory >> Next <<Next.

7- Specify the initial Content of the memory using **.hex** if exist as shown.

8-Next >>Finish.

After these steps <file>.qip will be created and after synthesis the IP Component of <file> will be added.

Steps of including OpenCores IP :

1-create a new project and add <file>.vhd of your IP core .

2- Right click on the file, choose "**set as top level entity**" and t-run analysis and synthesis.

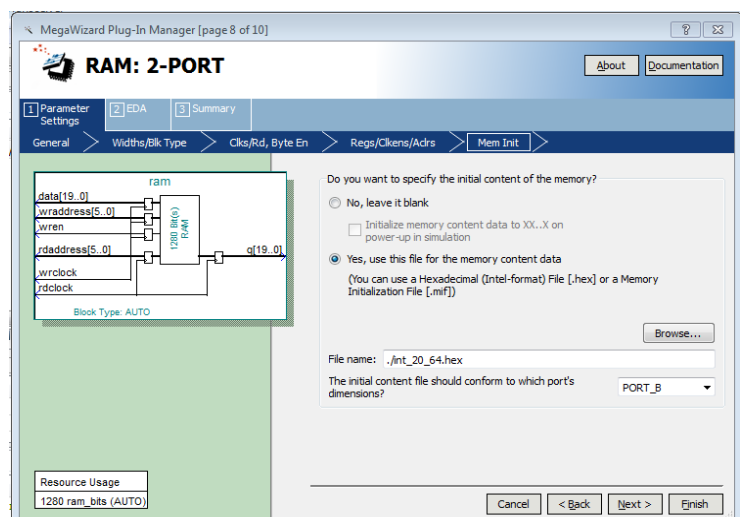


Figure 81

3- While without errors, go to **Project menu** >> **export design partition file**

4- Here you should generate file with an extension something like **.qxp**

5- Finally, you can use the generated partition file as an IP CORE in your design by adding it into project.

- These steps are very important to define all the memory in the project in order not to appear signals never read in design as a warning, and it may cause problem in the function simulation.

6.3 Function simulation using ModelSim

1- Open the Design >> write in the Transcript Window these steps:

```
>> cd {put directory of your project} ##change directory from default
>> vlib work ##create work library
>> vcom Transceiver.vhd ##compile this file
>> vsim -novopt Transceiver ##simulation
```

2- Create a Stimulus. First thing we need to do is to create a stimulus on our input >> right click on **Add Wave** >> click on **Clock**.

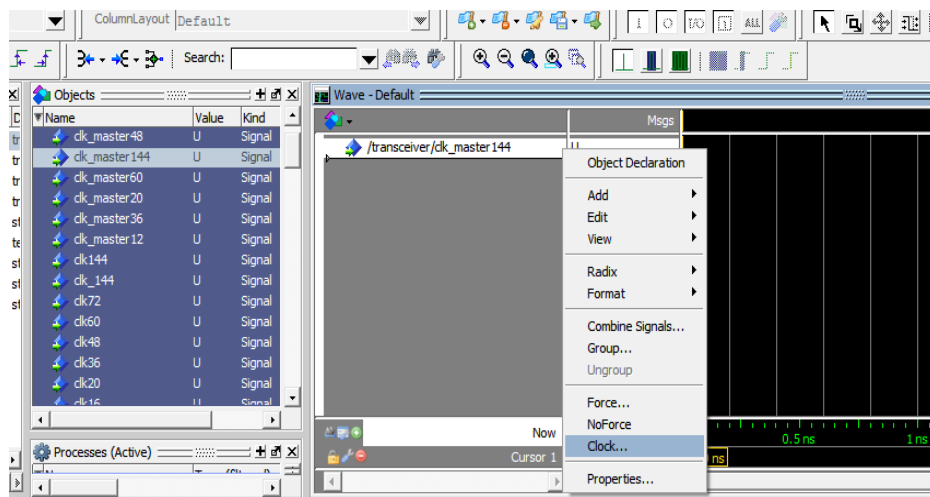


Figure 82

- 3- Define Wave as Clock as for example: “clock_master144” its frequency 144 Mega so Period 6.944 ns , make sure to keep the lowercase “ns” or ModelSim will complain.
- 4- After adding all waves and forcing value, write >>run 40000 in transcript .

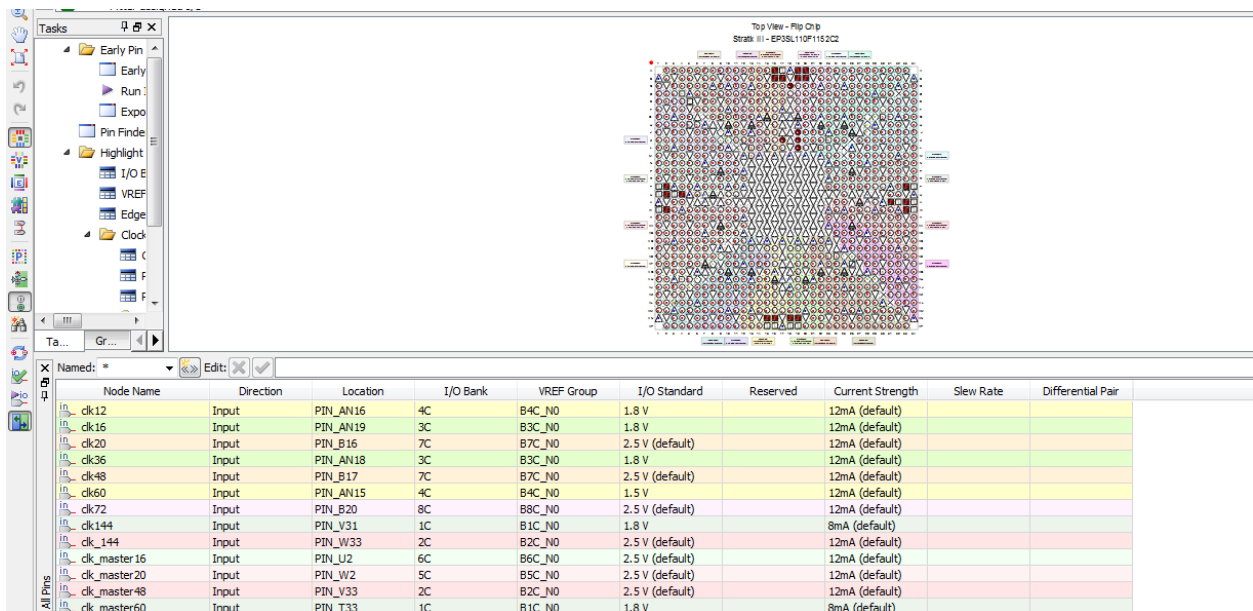
6.4 Verification on FPGA kit

➤ **Pin Assignment on the Stratix III kit:** to do that goes these steps

1-Open Quartus II.

2- Compile your Code.

3-Now we need to assign signals we use in our design to the FPGA pins and this can be done as follow. Click **Assignments >> choose pin planner** ,will appear this window ,Now begin to assign your signals to the suitable pins .



The screenshot shows the Pin Planner window in Quartus II. The top part displays a grid of pins with various colors indicating their status. The bottom part shows a table of assignments with the following columns: Node Name, Direction, Location, I/O Bank, VREF Group, I/O Standard, Reserved, Current Strength, Slew Rate, and Differential Pair.

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair
in_ ck12	Input	PIN_AN16	4C	B4C_N0	1.8 V		12mA (default)		
in_ ck16	Input	PIN_AN19	3C	B3C_N0	1.8 V		12mA (default)		
in_ ck20	Input	PIN_B16	7C	B7C_N0	2.5 V (default)		12mA (default)		
in_ ck36	Input	PIN_AN18	3C	B3C_N0	1.8 V		12mA (default)		
in_ ck48	Input	PIN_B17	7C	B7C_N0	2.5 V (default)		12mA (default)		
in_ ck60	Input	PIN_AN15	4C	B4C_N0	1.5 V		12mA (default)		
in_ ck72	Input	PIN_B20	8C	B8C_N0	2.5 V (default)		12mA (default)		
in_ ck144	Input	PIN_V31	1C	B1C_N0	1.8 V		8mA (default)		
in_ ck_144	Input	PIN_W33	2C	B2C_N0	2.5 V (default)		12mA (default)		
in_ ck_master16	Input	PIN_U2	6C	B6C_N0	2.5 V (default)		12mA (default)		
in_ ck_master20	Input	PIN_W2	5C	B5C_N0	2.5 V (default)		12mA (default)		
in_ ck_master48	Input	PIN_V33	2C	B2C_N0	2.5 V (default)		12mA (default)		
in_ ck_master60	Input	PIN_T33	1C	B1C_N0	1.8 V		8mA (default)		

Figure 83

➤ **Note:** Your signals may be assigned to different types of pins like FPGA pins LEDs pins , Switches pins we will show it.

4- Compile the project again

5-Now you are ready to download the design on the FPGA

Click **Tools >> Programmer** then click **Add File** and choose the **.sof file**
 Connect the JTAG to the kit and the USB Blaster to your PC.
 Check the **program/configure** box then click start.

	tatu	From	To	Assignment Name	Value	Enabled
1	✓		in- clk_master72	Location	PIN_T2	Yes
2	✓		in- clk_master16	Location	PIN_U2	Yes
3	✓		in- clk_master144	Location	PIN_U4	Yes
4	✓		in- clk_master20	Location	PIN_W2	Yes
5	✓		in- clk_master48	Location	PIN_V33	Yes
6	✓		in- clk_master60	Location	PIN_T33	Yes
7	✓		in- clk_master60	I/O Standard	1.8 V	Yes
8	✓		in- clk_144	Location	PIN_W33	Yes
9	✓		in- clk144	Location	PIN_V31	Yes
10	✓		in- clk144	I/O Standard	1.8 V	Yes
11	✓		in- clk72	Location	PIN_B20	Yes
12	✓		in- clk16	Location	PIN_AN19	Yes
13	✓		in- clk20	Location	PIN_B16	Yes
14	✓		in- clk48	Location	PIN_B17	Yes
15	✓		in- clk60	Location	PIN_AN15	Yes
16	✓		in- clk36	Location	PIN_AN18	Yes
17	✓		in- clk12	Location	PIN_AN16	Yes
18	✓		in- data_rate[0]	Location	PIN_B19	Yes
19	✓		in- data_rate[1]	Location	PIN_A19	Yes
20	✓		in- data_rate[2]	Location	PIN_C18	Yes
21	✓		in- data_rate[3]	Location	PIN_A20	Yes
22	✓		in- Mac_start	Location	PIN_K17	Yes
23	✓		in- input_..._start	Location	PIN_A16	Yes
24	✓		in- In_signal_ex	Location	PIN_K19	Yes

Table6. 1

The following steps describe the basic design flow for using the Quartus II GUI:

1. To create a new project and specify a target device or device family, on the File menu, click New Project Wizard.
2. Use the Text Editor to create a Verilog HDL, VHDL, or Altera Hardware Description Language (AHDL) design.

3. Use the Block Editor to create a block diagram with symbols that represent other design files, or to create a schematic.
4. Use the MegaWizard® Plug-In Manager to generate custom variations of megafunction and IP functions to instantiate in your design, or create a system-level design by using SOPC Builder or DSP Builder.
5. Specify any initial design constraints using the Assignment Editor, the Pin Planner, the Settings dialog box, the Device dialog box, the Chip Planner, the Design Partitions window, or the Design Partition Planner.
6. (Optional) Perform an early timing estimate to generate early estimates of timing results before fitting.
7. Synthesize the design with Analysis & Synthesis.
8. (Optional) If your design contains partitions and you are not performing a full compilation, merge the partitions with partition merge.
9. (Optional) Generate a functional simulation netlist for your design and perform a functional simulation with an EDA simulation tool.
10. Place and route the design with the Fitter.
11. Perform a power estimation and analysis with the PowerPlay Power Analyzer.
12. Use an EDA simulation tool to perform timing simulation for the design.
13. Use the TimeQuest Timing Analyzer to analyze the timing of your design.

14. (Optional) Use physical synthesis, the Chip Planner, LogicLock™ regions, and the Assignment Editor to correct timing problems.

15. Create programming files for your design with the Assembler, and then program the device with the Programmer and Altera programming hardware.

6.6 Introduction to Nios Development Board and Stratix III FPGA:

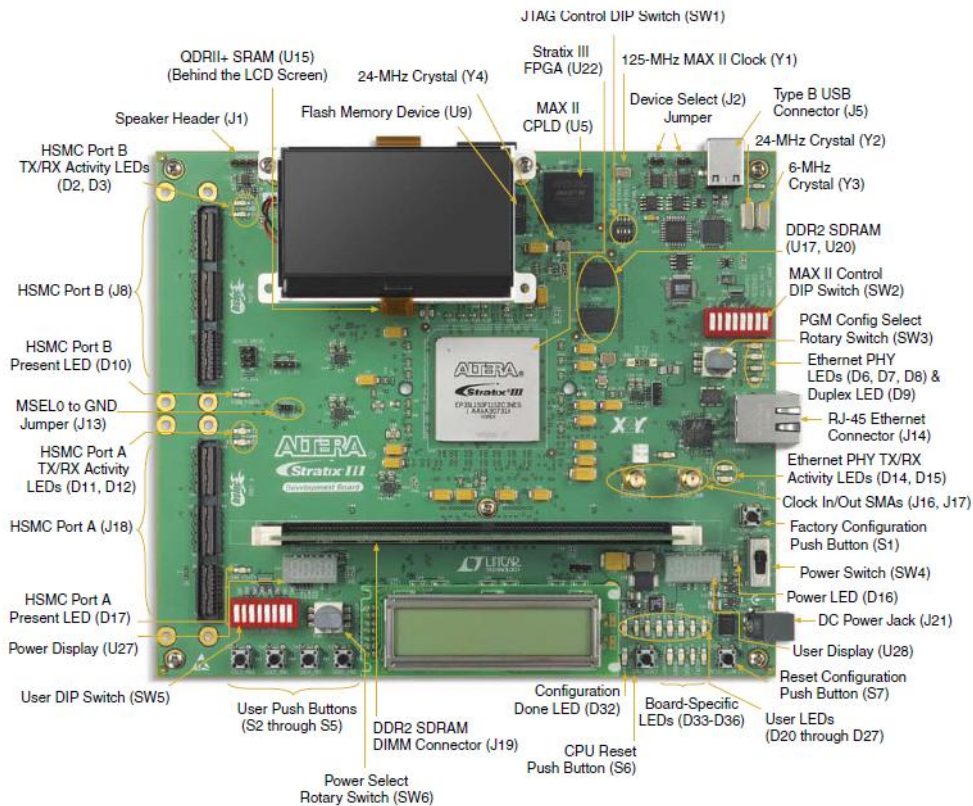


Figure 84 Stratix III kit

>> The Nios Development Board, Stratix II Edition, provides a hardware platform for developing embedded systems based on Altera Stratix II devices. The Nios Development Board, Stratix III Edition provides the following features:

- Single-ended, non-voltage-referenced, and voltage-referenced I/O standards
- Low-voltage differential signaling (LVDS), reduced swing

differential signal (RSDS), mini-LVDS, high-speed transceiver logic (HSTL), and stub series terminated logic (SSTL)

- Single data rate (SDR) and half data rate (HDR—half frequency and twice data width of SDR) input and output options
- Up to 132-full duplex 1.6-Gbps true LVDS channels (132 Tx + 132 Rx) on the row I/O banks
- Hard dynamic phase alignment (DPA) block with serializer /deserializer (SERDES)
- De-skew, read and write leveling, and clock-domain crossing functionality
- Programmable output current strength
- Programmable slew rate
- Programmable delay
- Programmable bus-hold
- Programmable pull-up resistor
- Open-drain output
 - Serial, parallel, and dynamic on-chip termination (OCT)
- Differential OCT
- Programmable pre-emphasis
- Programmable differential output voltage (VOD)

CHAPTER 7

Chapter 7. Optical Fibers

Mostly the predominant use of optical technology is as very fast “electric wire”. Optical fibers replace electric wire in communications systems and nothing much else changes, but in our case we replace the wireless channel with optical fiber channel to increase the data rate.

7.1 introduction

Optical fiber is an excellent physical medium of high-speed transmissions. An optical fiber provides low-attenuation transmission over a huge frequency range which allows signals to be transmitted over long distances at high speeds without being amplified or regenerated. Due to these properties, optical fibers are favored over traditional transmission media such as copper or free space. Development of fibers and devices for optical communications began in the early 1960s and continues strongly today. But the real change came in the 1980s. During this decade optical communication in public communication networks developed from the status of a curiosity into being the dominant technology. It is important to realize that optical communications is not like electronic communications. While it seems that light travels in a fiber much like electricity does in a wire this is very misleading. Light is an electromagnetic wave and optical fiber is a waveguide.

7.2 Optical fiber advantages

► The Optical communication has many advantages:

- **Material Cost**

Fiber cable cost is significantly less than copper cable for the same transmission capacity.

- **Information Capacity**

In 1998, the data rate of fiber systems in use is generally hundreds of Mbps on a single fiber. Now, the data rate in range of tens of Gbps. This is a very high rate in digital transmission.

- **No Electrical Connection**

In electrical systems there is always the possibility of “ground loops” causing a serious problem, especially in the LAN or computer channel environment. As often there is a voltage potential difference between “grounds” at different locations.

- **Long-distance Signal Transmission**

The low attenuation and superior signal integrity found in optical systems allow much longer intervals of signal transmission than metallic-based systems.

- **Large Bandwidth, Light Weight, and Small Diameter**

Today’s communication applications require an ever-increasing amount of bandwidth, which make it possible to increase data rate. It is commonplace to install new cabling within existing duct systems or conduit. Fiber cables are smaller and lighter than electrical cables to do the same job. A large coaxial cable involves a cable of several inches in diameter which is heavier. A fiber cable could be less than one half an inch in diameter and lighter to do the same job. The relatively small diameter and light weight of optical cable make such installations easy and practical, saving valuable conduit space in these environments. This means that the cost of laying the cable is reduced.

- **Non-conductivity**

The dielectric nature of optical fibers is considered an important advantage. Since optical fiber has no metallic components, it can be installed in areas with electromagnetic interference (EMI), including radio frequency interference (RFI). Areas with high EMI include utility lines, power-carrying lines, and railroad tracks.

- **Security**

Unlike metallic-based systems, the dielectric nature of optical fiber makes it impossible to remotely detect the signal being transmitted within the cable. The only way to do so is by accessing the optical fiber. Accessing the fiber requires intervention that is easily detectable by security surveillance. These circumstances make fiber extremely attractive to governmental bodies, banks, and others with major security concerns.

- **Designed For Future Applications Needs**

Fiber optics is affordable today, as electronics prices fall and optical cable pricing remains low. In many cases, optical fiber solutions are less costly than copper systems. As bandwidth demands increase rapidly with technological advances, fiber will continue to play a vital role in the long-term success of telecommunication.

7.3 Basic Fiber Optic Communication System

The transmission theory: Fiber optics is a medium for carrying

information from one point to another in the form of light. A basic fiber optic system consists of a transmitting device that converts an electrical signal into a light signal, an optical fiber cable that carries the light, and a receiver that accepts the light signal and converts it back into

an electrical signal.

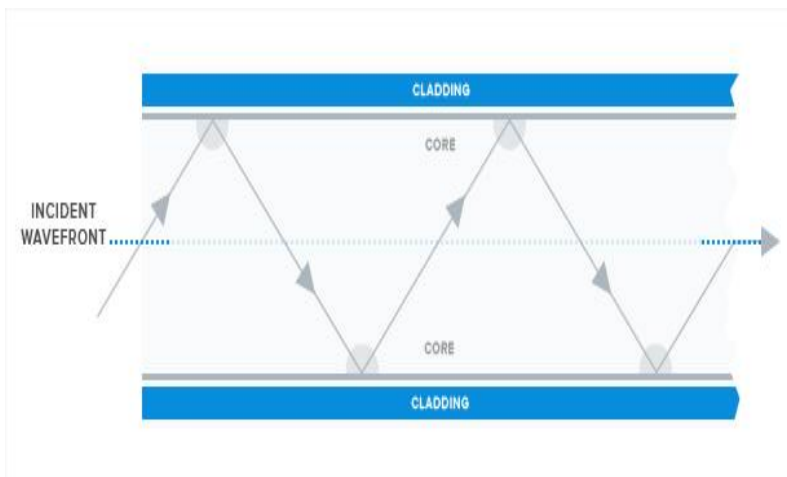


Figure 85 Data transmission in fibers

7.4 How Fiber Works

Total internal reflection is the principle of optical fiber work. Light reflects or refracts, depending on the angle at which it strikes a surface. One way of thinking about this concept is to envision a person looking at a lake. By looking down at a steep angle, the person will see fish, rocks, vegetation, or whatever is below the surface of the water (in a somewhat distorted location due to refraction), assuming that the water is relatively clear and calm. However, by casting a glance farther out, thus making the angle of sight less steep, the individual is likely to see a reflection of trees or other objects on an opposite shore. Because air and water have different indices of refraction, the angle at which a person looks into or across the water influences the image seen. This principle is at the heart of how optical fiber works. Controlling the angle at which the light waves are transmitted makes it possible to control how efficiently they reach their destination.

Light waves are guided through the core of the optical fiber in much the same way that radio frequency (RF) signals are guided through coaxial cable. The light

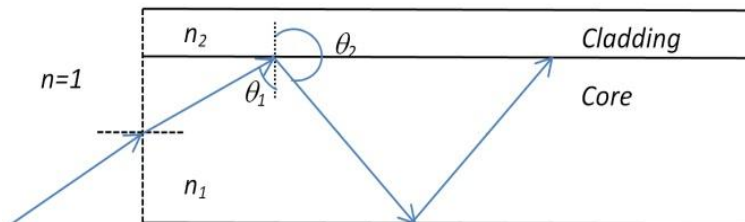


Figure 86 Total reflection in optical fibers

waves are guided to the other end of the fiber by being reflected within the core. The composition of the cladding glass relative to the core glass determines the fiber's ability to reflect light. That reflection is usually caused by creating a higher refractive index in the core of the glass than in the surrounding cladding glass, creating a "waveguide." The refractive index of the core is increased by slightly modifying the composition of the core glass, generally by adding small amounts of a dopant. Alternatively, the waveguide can be created by reducing the refractive index of the cladding using different dopants.

7.5 How to Choose Optical Fiber Single-Mode Fiber Performance Characteristics

The key optical performance parameters for single-mode fibers are attenuation, dispersion, and mode-field diameter. Optical fiber performance parameters can vary significantly among fibers from different manufacturers in ways that can affect the system's performance.

Attenuation : is the reduction of signal strength or light power over the length of the light-carrying medium. Fiber attenuation is measured in (dB/km). Optical fiber offers superior performance over other transmission media because it combines high bandwidth with low attenuation. This allows signals to be transmitted over longer distances while using fewer regenerators or amplifiers, thus reducing cost and improving signal reliability.

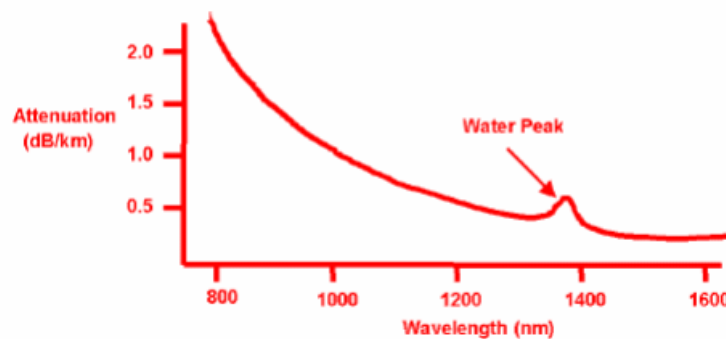


Figure 87 Attenuation in fibers

Attenuation of an optical signal varies as a function of wavelength. Attenuation is very low, as compared to other transmission media for standard single-mode fiber. Attenuation is caused by scattering and absorption of light.

Dispersion : is the time distortion of an optical signal that results from the time of flight differences of different components of that signal, typically resulting in pulse broadening (illustrated

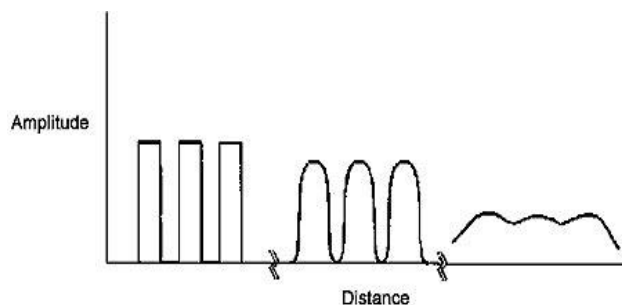


Figure 88 Signal amplitude as distance increases in fibers

in figure (88)). In digital transmission maximum data rate, the maximum distance, or the information-carrying capacity of a single-mode fiber link. In analog transmission, dispersion can cause a waveform to become significantly distorted.

Single-mode fiber dispersion varies with wavelength and is controlled by fiber design (illustrated in figure (89)). The wavelength at which dispersion equals zero is called the zero-dispersion wavelength. This is the wavelength at which fiber has its maximum information-carrying capacity.

Mode-Field Diameter: describes the size of the light-carrying portion of the fiber. For single-mode fibers, this region includes the fiber core as well as a small portion of the surrounding cladding glass. Mode-field diameter is an important parameter for determining a fiber's resistance to bend-induced loss and can affect splice loss as well.

Cutoff Wavelength: is the wavelength above which a single-mode fiber supports and propagates only one

mode of light. An optical fiber that is single-mode at

a particular wavelength may have two or more modes at wavelengths lower than the cutoff wavelength. The effective cutoff wavelength of a fiber is dependent on the length of fiber and its deployment and the longer the fiber, the lower the effective cutoff wavelength.

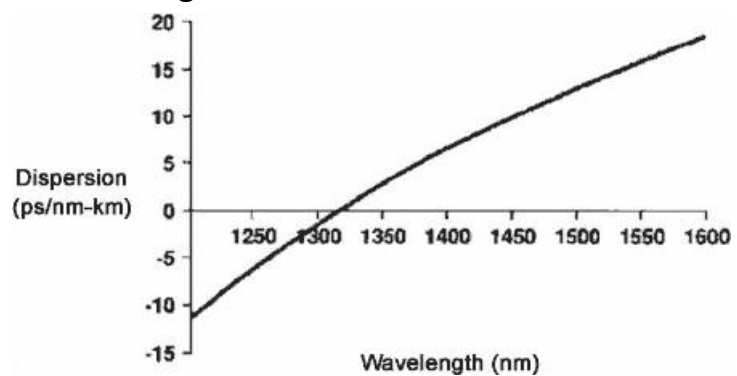


Figure 89 Dispersion in fibers

7.6 Single-Mode Propagation

In a single-mode, the fiber is so small compared with the wavelength of the light that the light is confined to go in one path. The light in case acts as an electromagnetic wave in a waveguide.

The light travelling down a single-mode fiber (illustrated in figure (90)). The important point here is that the light must be thought of as an

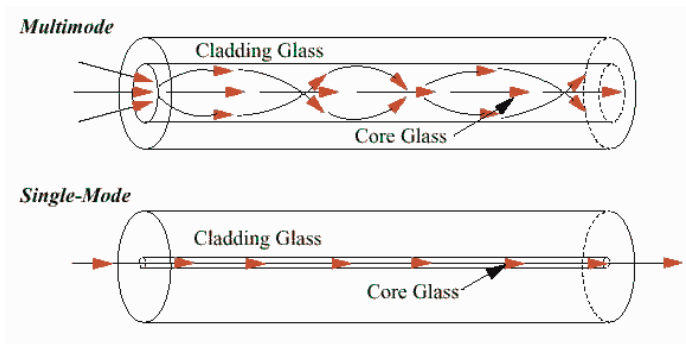


Figure 90 Fiber operating modes

electromagnetic wave. The electric and magnetic fields decrease exponentially as we move away from the axis of the fiber but nevertheless a significant amount of the optical power travels in the

cladding. The operation of single-mode fiber, one can

use the idea that only the axial mode propagates in the fiber. Single-mode fiber bandwidth is so large compared to that of the multimode fiber.

7.6.1 Single-Mode Characteristics

7.6.1.1 Mode Field Diameter (MFD) and Spot Size

In case of single-mode, optical power travels in both the core and the cladding. In single-mode fiber the core diameter is not sufficient. The mode field can be considered the effective core of the fiber although the real core size is typically somewhat smaller.

7.6.1.2 Field Strength at the Fiber End

Near Field: This is the electromagnetic field at the end face of the fiber itself. The fiber will be a profile of the bound mode in the single-mode but if the measurement is made close to the transmitter there may be cladding modes present as well.

Far Field: It is extremely difficult to measure the near field directly, so the far field is measured and used to calculate the characteristics of the near

field. It consists of a series of lobes spread out away from the axis of the fiber.

7.6.1.3 Bend Loss in Single-Mode Fiber

Considering light in single-mode fiber propagating along the fiber as a wave. A phase front moves along the fiber perpendicular to the direction of propagation.

The wave front must be in-phase with itself across the diameter of the field. As the phase front moves into a bend the light at the inner radius of the bend must move more slowly than the light at the outer radius.

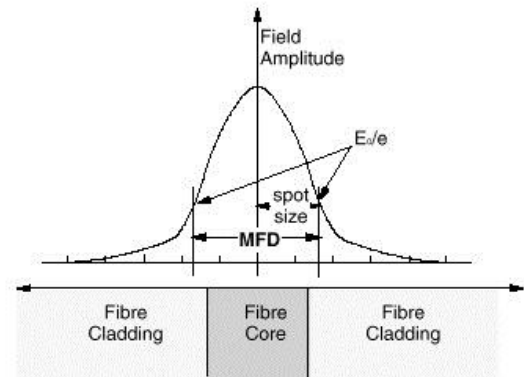


Figure 91 Single fiber bend loss

7.6.1.4 Cutoff Wavelength

The cutoff wavelength is the shortest wavelength at which the fiber will be single-mode. On the other hand wavelengths shorter than the cutoff will travel in multiple modes whereas wavelengths longer than the cutoff will travel in a single mode (Dutton, 1998).

CHAPTER 8

Chapter 8. OFDM Over Optical Access Network

8.1 Introduction

Optical access network is the network that involves light to send data from point to another using optical fibers. Optical access network is considered as a promising technique that would help a lot to satisfy the world's growing need to a wider bandwidth (htt33).

The ultimate goal of the optical signal transmission is to achieve the predetermined bit error ratio (BER) between any two nodes in an optical network. The optical transmission system has to be properly designed to provide reliable operation during its lifetime, which includes the management of key engineering parameters. When simulated results of Fiber are compared with AWGN channel, it is observed that the performance of optical fiber link even for short distances is better than that of the AWGN channel with SNR of 50 dB & above, this is due to the better transmission capabilities of Optical fiber.

The next generation wireless communications systems need to be of a higher standard so as to support various Broadband wireless services such as Video conferencing, mobile videophones, high-speed Internet access, computer network applications and potentially many multimedia applications at a significantly reduced cost in comparison with 3G networks, while the existing wireless systems can hardly provide transmission capacity of the order of few Mbps, to deliver multimedia or broadband signals at remotely distributed cells, as well as wireless transmission channels are no more able to fulfill the demands of higher bandwidth.

Optical fiber can provide data capacity of the order of tens and hundreds of Mbps. Because using wireless environment fading usually impairs signals and multipath delay, using RF OFDM the data rate will be, so OFDM with optical fiber channel is used rather than the wireless channel for higher SNR and data rate.

The optical network today is predominately based on point-to-point connections, and this is the evolution toward highly reconfigurable networks at the channel speed of 100 Gb/s. We aim to present the case that the time for OFDM over optical access network has come, in terms of the demand from today's ever-advancing optical networks and the availability of its underlying technologies.

The OFDM modulation scheme also leads to a high spectral efficiency because of its partially overlapping subcarriers. Moreover, the cyclic prefix code of the OFDM over optical network system makes the system more resistant to inter-symbol interference caused by chromatic dispersion and polarization mode dispersion.

First, we should define some light fundamentals that is used in optical before going through the OFDM over optical channel.

8.2 Light Fundamentals

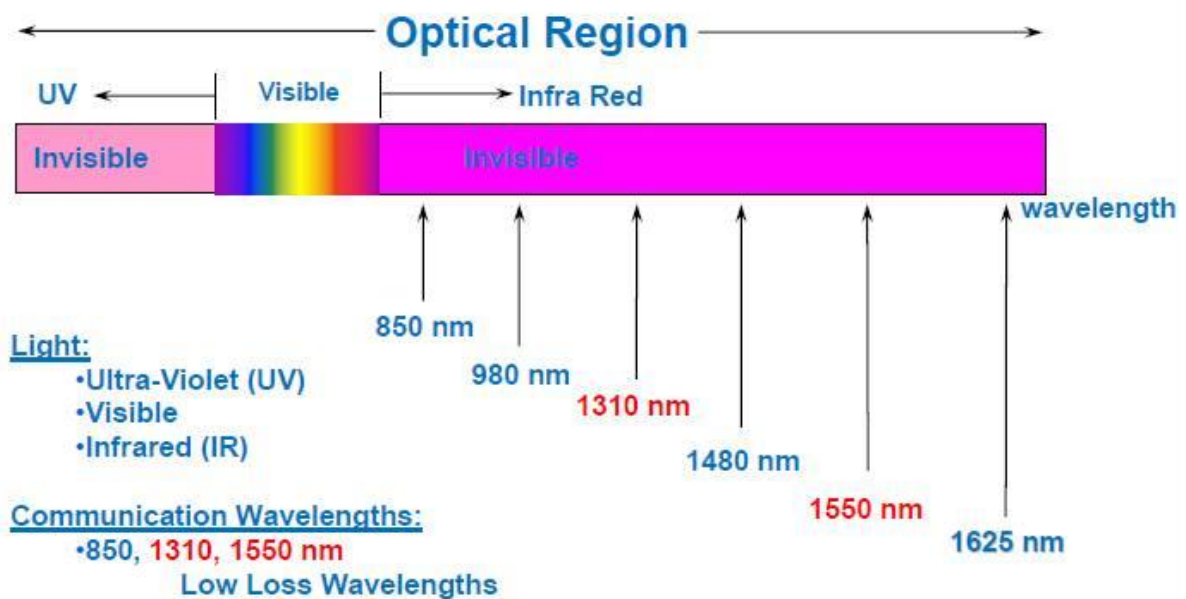


Figure 92 Optical region (RCDD)

The figure above shows the optical region and the wavelengths that can be used in communications. Data transmitting in fibers depends on two of light properties reflection and refraction (RCDD).

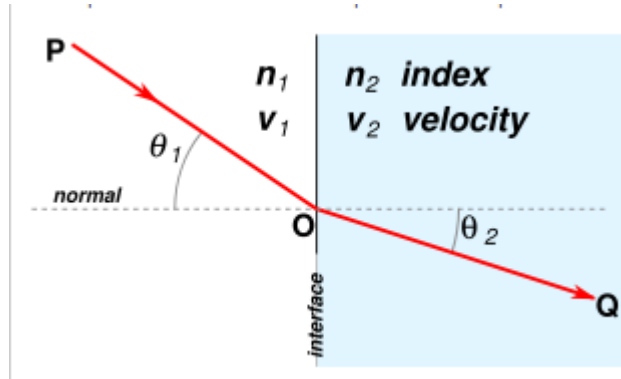


Figure 93 Light reflection and refraction (htt31)

8.3 Wave Division Multiplexing Technique (WDM)

Optical communications uses the wave division multiplexing technique which depends on the idea of combining signals with different wavelengths at the transmitter using a multiplexer and splitting them apart at the receiver. That should be done after ensuring that each system is sending a mono color, in other words only one wavelength to distinguish it at the receiver, the same as the prism's idea that uses the diffraction of light property to split the white light to 7 different colors according to the fact that every color

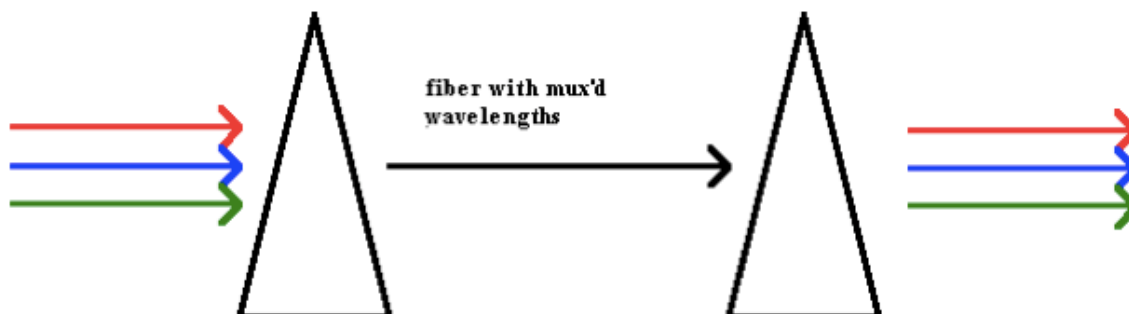
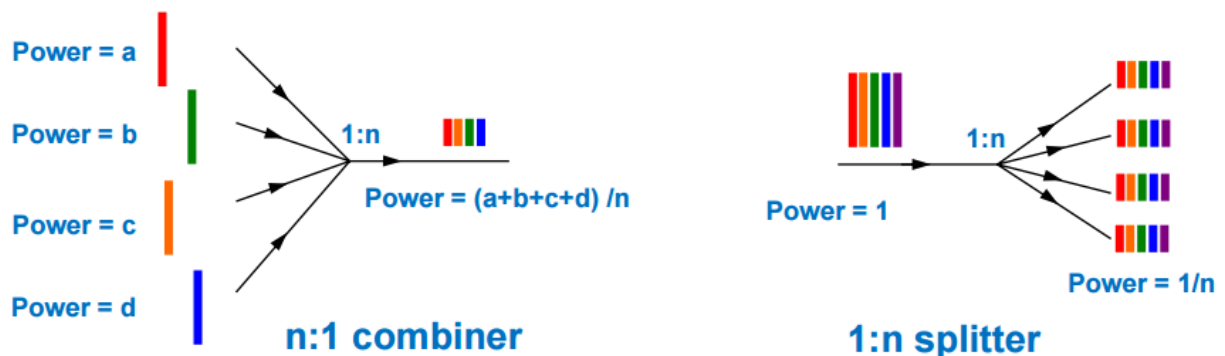


Figure 94 Wavelength division multiplexing technique

has a different wavelength and a different refractive index (RCDD).

The WDM technique allows communication companies to expand the capacity of their network without laying more fibers. Modern systems can handle up to 160 signals and can expand a basic 10 [Gbit/s](#) system over a single fiber pair to over 1.6 [Tbit/s](#) according



to Wikipedia website statistics.

Figure 95 Power distribution in WDM

The figure shown above illustrates how data power is combined at the transmitter, and how the receiver is distinguishing them. At the splitter data is sent to each receiver. Each receiver can receive any wavelength or multiple wavelengths according to its own configuration. Here, we can find that the power loss from the transmitter combiner is the same as the power loss resulted from the splitter.

8.4. Direct Detection OFDM

At our project, data is sent from one central to another one which is called a direct detection technique. Another technique is used is the coherent detection which sends data from the transmitter to different recipients which requires coherent receivers to synchronize with the data sent to each one.

At the direct detection, two dimension multiplexing is done. First one, is the frequency domain multiplexing made at the IFFT block of the ordinary OFDM system. Second, is the data sent in time domain with

different wavelengths that are combined together at the combiner. This way of sending data helps in increasing the bandwidth and rate of sent data using one optical fiber wire. At the splitter, data is sent to each receiver according to its wave length, then data is de-multiplexed again at the FFT block with each frequency sent. Direct detection OFDM can be used to support the FTTH (Fiber To The Home) idea, CATV, municipal networks, long data links such as utility grid management, supports mobile and internet companies providers to achieve better performance and higher capacities and some high speed LAN backbones.

8.5. Effective Wavelengths In Optical Transmissions

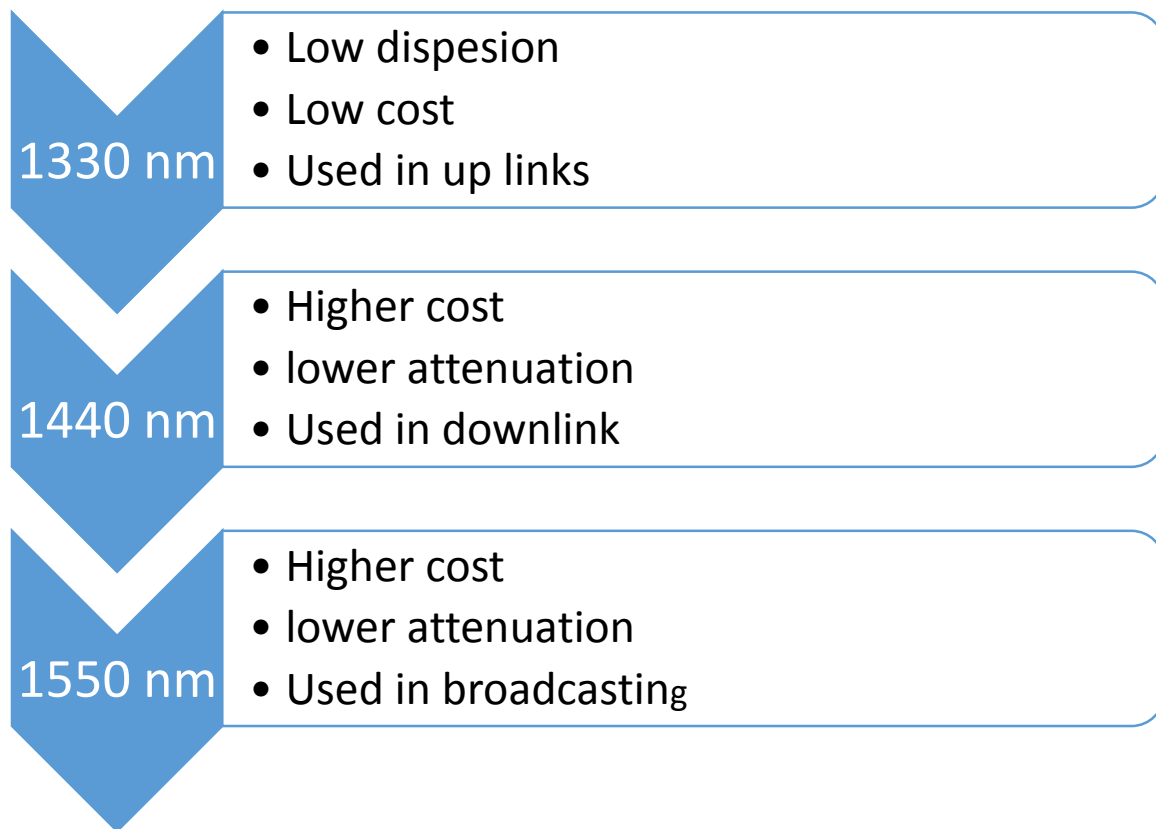


Figure 96 Wave lengths in optical transmission

8.6 Difference Between RF OFDM and OFDM Over Optical Channel

In wireless systems, OFDM has been incorporated in wireless local area network (LAN; IEEE 802.11a/g, better known as Wi-Fi), wireless WAN (IEEE 802.16e, better known as WiMAX), and digital radio/video systems (DAB/DVB) adopted in most areas of the world. In RF cable systems, OFDM has been incorporated in ADSL and VDSL broadband access via telephone copper wiring or power line. This rich variation results from the intrinsic advantages of OFDM modulation, including dispersion robustness, ease of dynamic channel estimation and mitigation, high spectral efficiency, and the capability of dynamic bit and power loading.

Despite the fact that OFDM has been extensively studied in the RF domain, it is rather surprising that the first report on OFDM with optical channel in the open literature appeared only in 1996. The lack of interest in optical OFDM in the past is largely due to the fact that the silicon signal processing power had not reached the point at which sophisticated OFDM signal processing could be performed in a CMOS integrated circuit (IC).

8.7. FUTURE WORK

8.7.1 Principle of Coherent Optical OFDM

Coherent Optical OFDM (CO-OFDM) is considered an enabling technology of the next generation optical communication system. As a coherent system, the CO-OFDM system maintains both signal amplitude and phase, thus increasing bandwidth utilization. The coherent optical communication system makes full compensation of chromatic dispersion, after optical/electrical conversion, possible.

OFDM brings coherent systems computation efficiency and ease of channel and phase estimation. The coherent systems bring OFDM a much needed linearity in RF-to-optical (RTO) up-conversion and optical-to-RF (OTR) down-conversion. CO-OFDM can be divided into five blocks:

1. RF OFDM transmitter.
2. RTO up-converter.
3. Optical channel.
4. OTR down-converter
5. RF OFDM receiver.

Each block is a structure of blocks and components dedicated for the function of the main block. The generated or recovered OFDM signals in either baseband or RF band. System linearity can be achieved by biasing the mach-zehnder modulators (MZM) ,as a linear conversion between RF signal and optical signal ,then a linear transformation from optical signal and electrical or RF signal can be achieved by using coherent detection ,so we can solve system non-linearity before and after the channel .The design of the optical transmitter and the design of optical receiver with RTO up-converter and OTR down-converter to solve system non-linearity , but it increases system complexity.

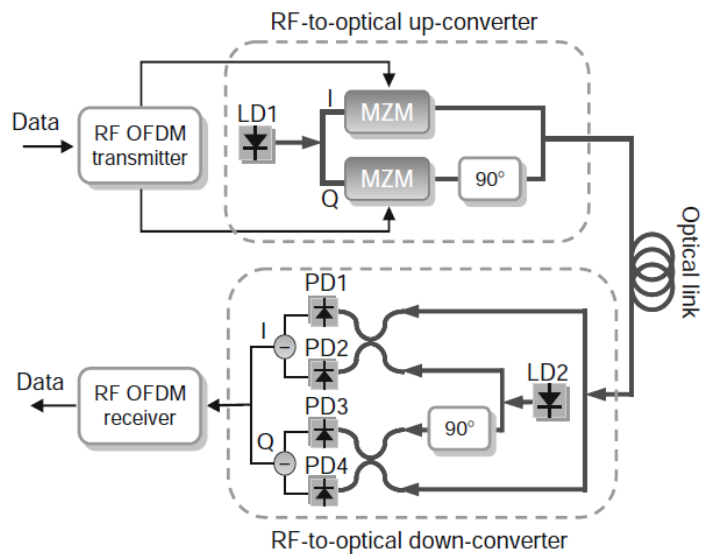


Figure 97 up-converter/down-converter

8.8 difference between OFDM with optical fiber channel and CO-OFDM

The large divergence in optical dispersion and mechanical stability makes it difficult for conventional modulation formats, such as direct-detection to meet the challenges for high-speed transmission at 100 Gb/s and beyond.

OFDM with optical channel combines the advantages of (direct detection) and orthogonal frequency-division multiplexing (OFDM

modulation), so it depends on simple hardware system that consists of an OFDM transmitter and receiver are connected to each other via a simple optical channel.

On the other hand CO-OFDM combines the advantages of (coherent detection) and orthogonal frequency-division multiplexing (OFDM modulation) and possess many merits that are critical for future high-speed fiber transmission systems, but it requires the highest complexity in transceiver design. It has recently emerged as an attractive modulation format for optical communications (Djordjevic, 2010).

CHAPTER 9

Chapter9. Opti-System

9.1. Introduction

Opti-system is a newly released tool in 2009 based on MATLAB which can be used to measure the BER between the data sent and data received via the optical channel since there is no other tool was developed for such a purpose.

Opti system simulations is done depending on 2014's class OFDM MATLAB codes. These simulations are done mainly to measure the effect of the optical channel on the OFDM performance and to calculate the BER rate reached when increasing the fiber length starting from 100m to 30 km.

9.2 Simulation Main Blocks

9.2.1 User defined Bit Sequence Block:

This block is used to send data offline to the MATLAB block. Find below, the configuration of the block, and values of sent data.

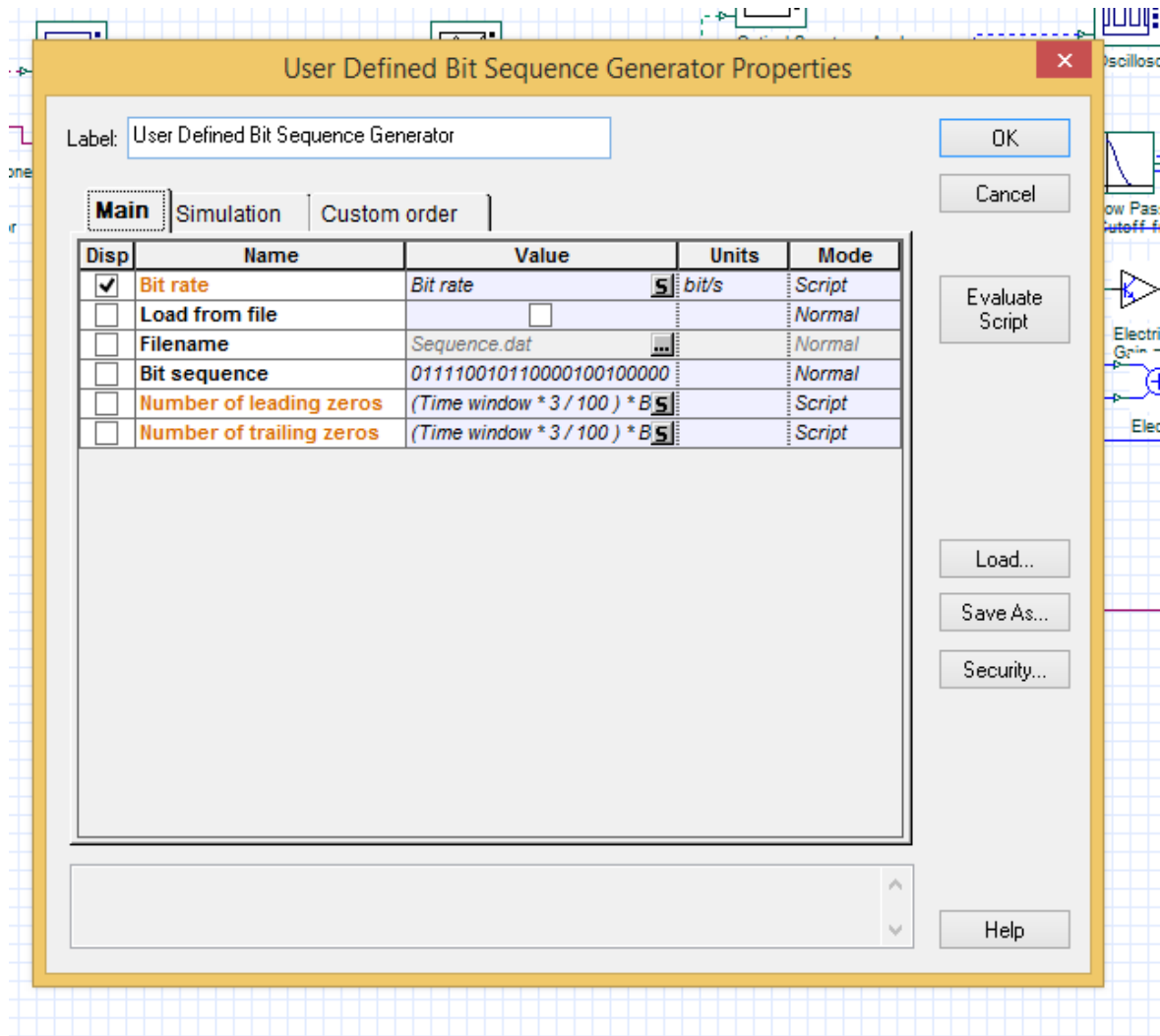


Figure 98 User defined binary sequence block configuration

9.2.2 MATLAB Block:

This block enables the utilization of components created in MATLAB.

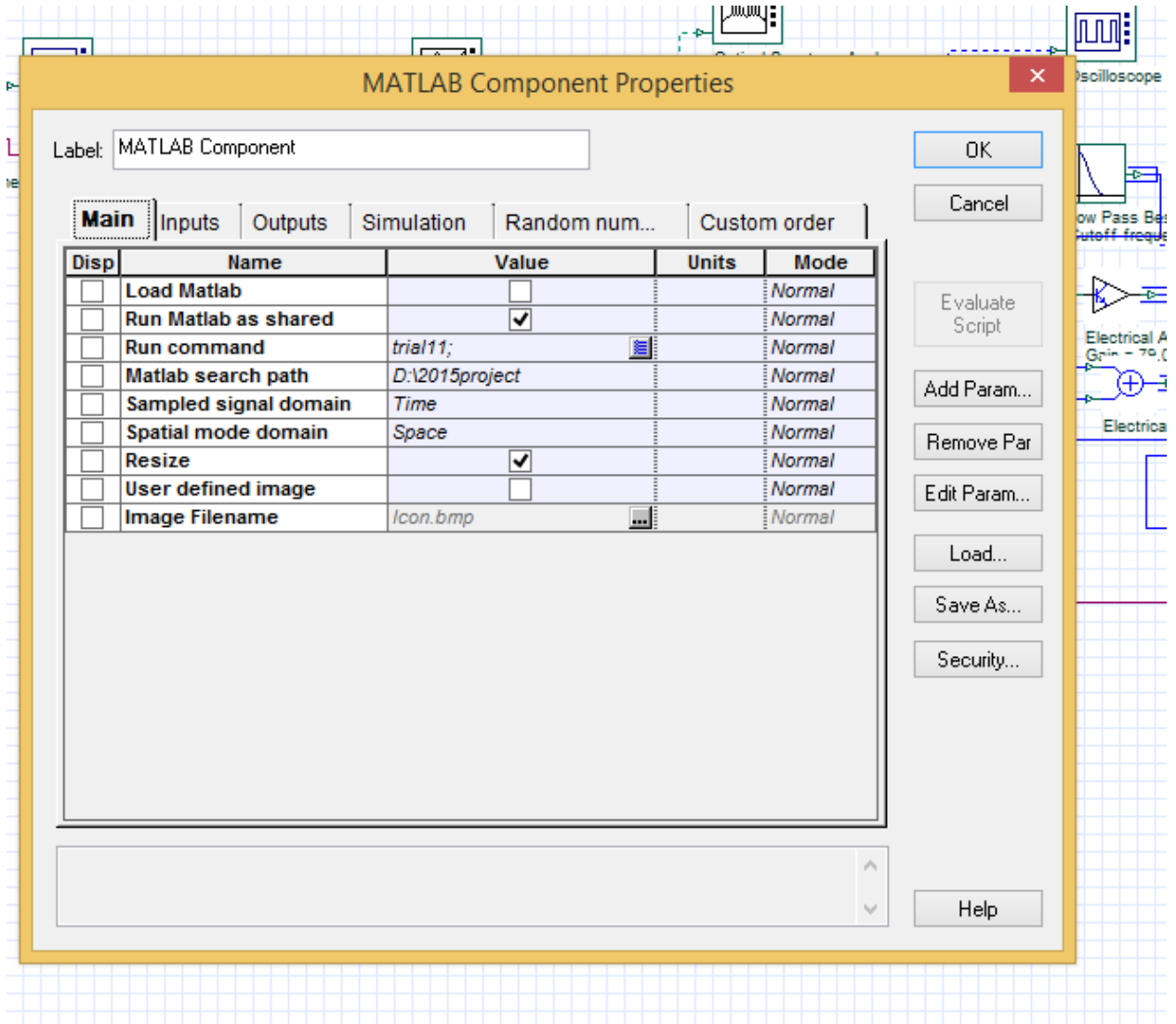


Figure 99 MATLAB block configuration

9.2.3 M-ary Pulse Generator:

Used to convert decimal output generated from MATLAB block to electrical signal divided into levels.

9.2.4 Mach-Zehnder Modulator:

Since we are going to use an optical fibers to send data, we are in need to convert the RF (radio frequency) signal sent from transmitter to a laser signal. Here, we used the Mach-zehnder modulator.

We have several types of Mach-zehnder modulators, depends on light property used in modulation. One can find the Intensity, power, phase, etc. modulator. Here we use an intensity Mach-zehnder modulator.

Mach-zehnder intensity modulator work idea is to send data through two waveguides by splitting the wave at the beginning of the waveguides as shown in figure (100). Depending on the equation:

$$v(\text{speed}) = \frac{d(\text{waveguide length})}{t(\text{time needed to pass the waveguide})}$$

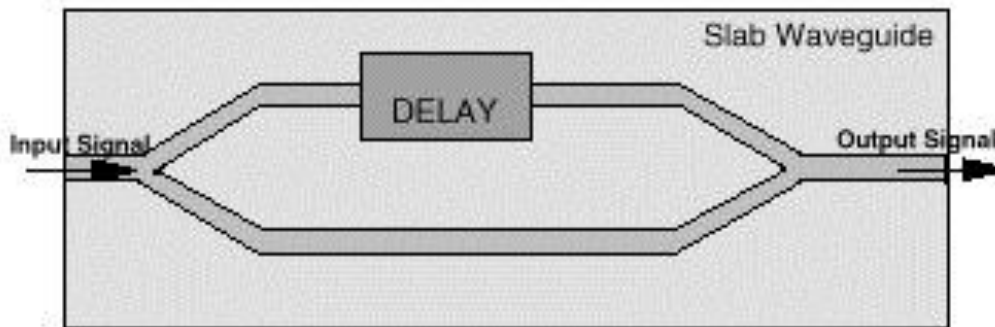


Figure 100. Mach-Zehnder modulator

Data can be sent via both of waveguides and combining them at the end of the waveguide. We can control the received wave amplitude value by applying a voltage drop to one waveguide or to both waveguides. Depending on the equation variable (v) to ensure a delay in one side of the wave guide.

When waves are sent without phase difference, they recombine at the end and power is almost the same with the less losses expected. When applying voltage difference causing a phase difference between the two waves, power is reduced at the end of the waveguide (htt32).

9.2.5 Optical Fiber:

Illustrated before in Chapter 7.

9.2.6 Photo-Detector:

In order to enable data to enter the receiver, we need to convert it from light beam to RF signal again. Then we used the photo detector which converts laser or light to electrical signals.

The photo detection idea is to detect light photons that affects a metal surface in Vacuum causing current to pass, depending on Planck's equation:

$$E = h\nu$$

(E) Stands for the energy resulted from photons, (h) for Planck's constant and (ν) for the frequency of radiation. As Planck's discovered that energy is radiated in small discrete units called quanta, which led to discover the photo-electric effect illustrated above. The photo-electric effect is the principle theory of light energy that's used in photo-detectors (htt30).

9.2.7 Low Pass Bessel Filter:

Filter with the Bessel frequency transfer function.

$$H(s) = \alpha \frac{d_0}{B_N(s)}$$

9.2.7 Electrical amplifiers:

Used to amplify the data after the channel as the power of the Mach-Zender is too small which resulted in decreasing the power of the electrical signal after conversion.

9.2.9 DC Offset Block and Electrical Adder:

Used to remove offset added to the signal while passing he channel.

9.2.10 M-ary Threshold Detector:

Used to detect the M-ary levels, and convert the signal to decimal values that MATLAB receiver block can deal with.

9.3. Schematic

9.3.1 Back To Back testing

9.3.1.1. 1st step: Testing the System back to back:

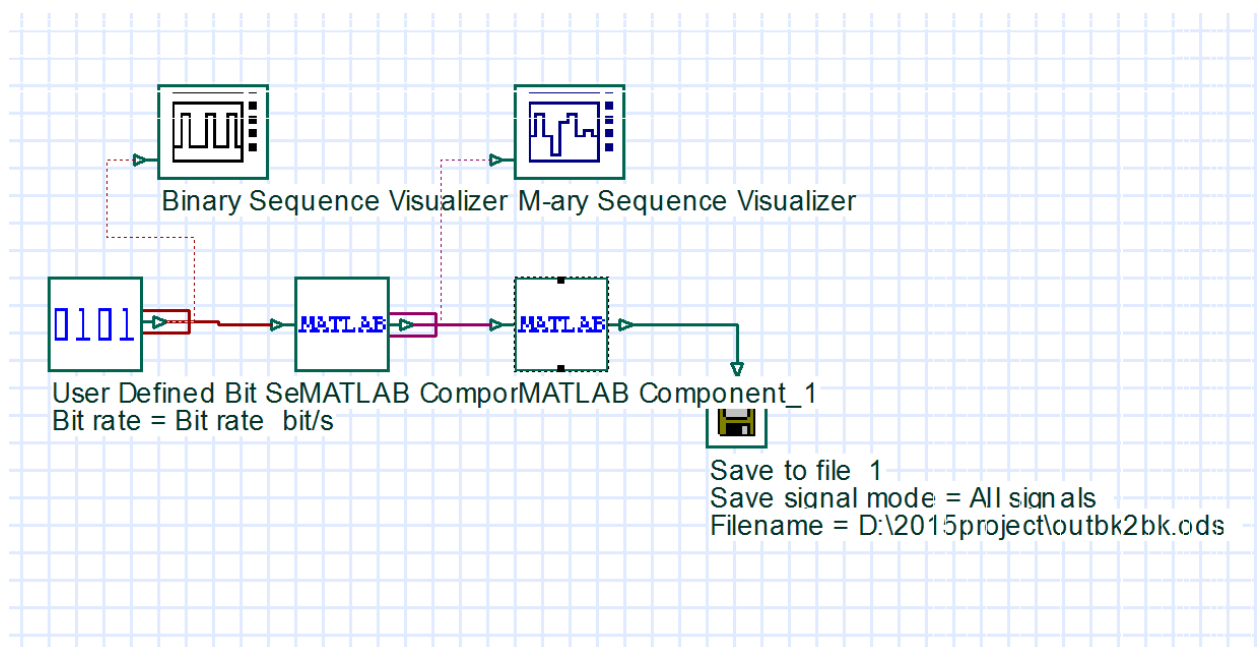


Figure 101 Back to black testing

9.3.1.2. Results

As shown here, the back to back system without channel, gives a bit error rate (BER) equals t zero.

BER				
1x1 double				
	1	2	3	4
1	0			

Figure 102 Back to back BER results

9.3.2. 2nd, Testing the system with 100 m fibers length

9.3.2.1. System Schematic

The following screenshots are for transmitter with channel and receiver respectively.

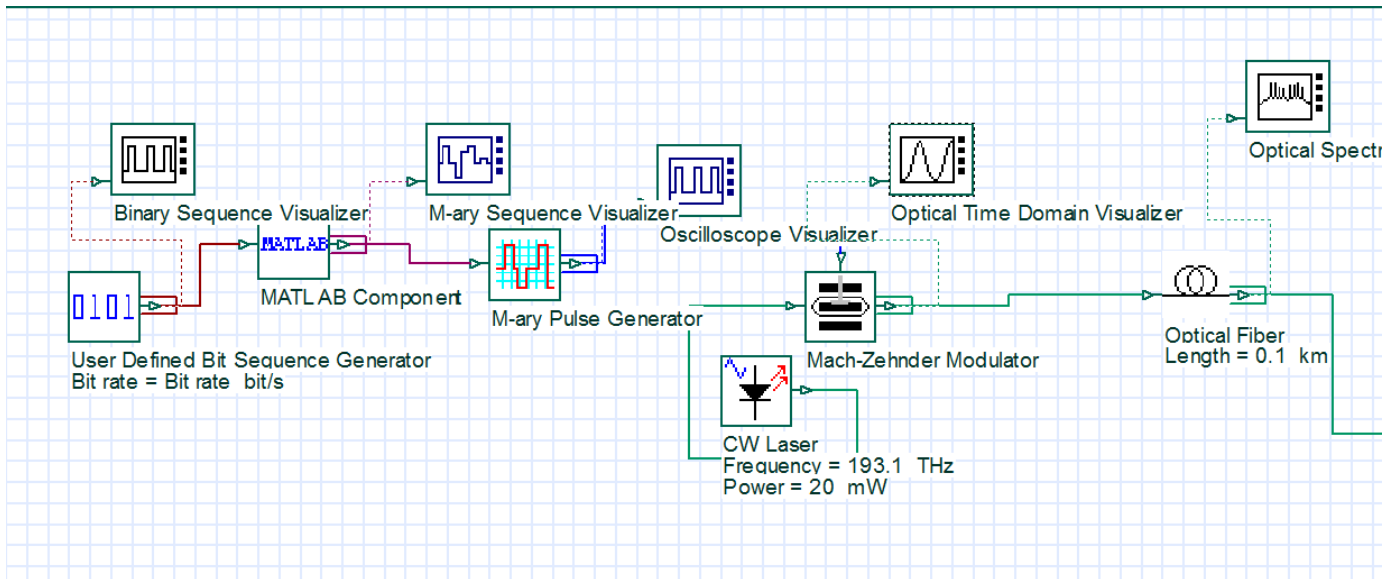


Figure 103 100 m test- Transmitter blocks

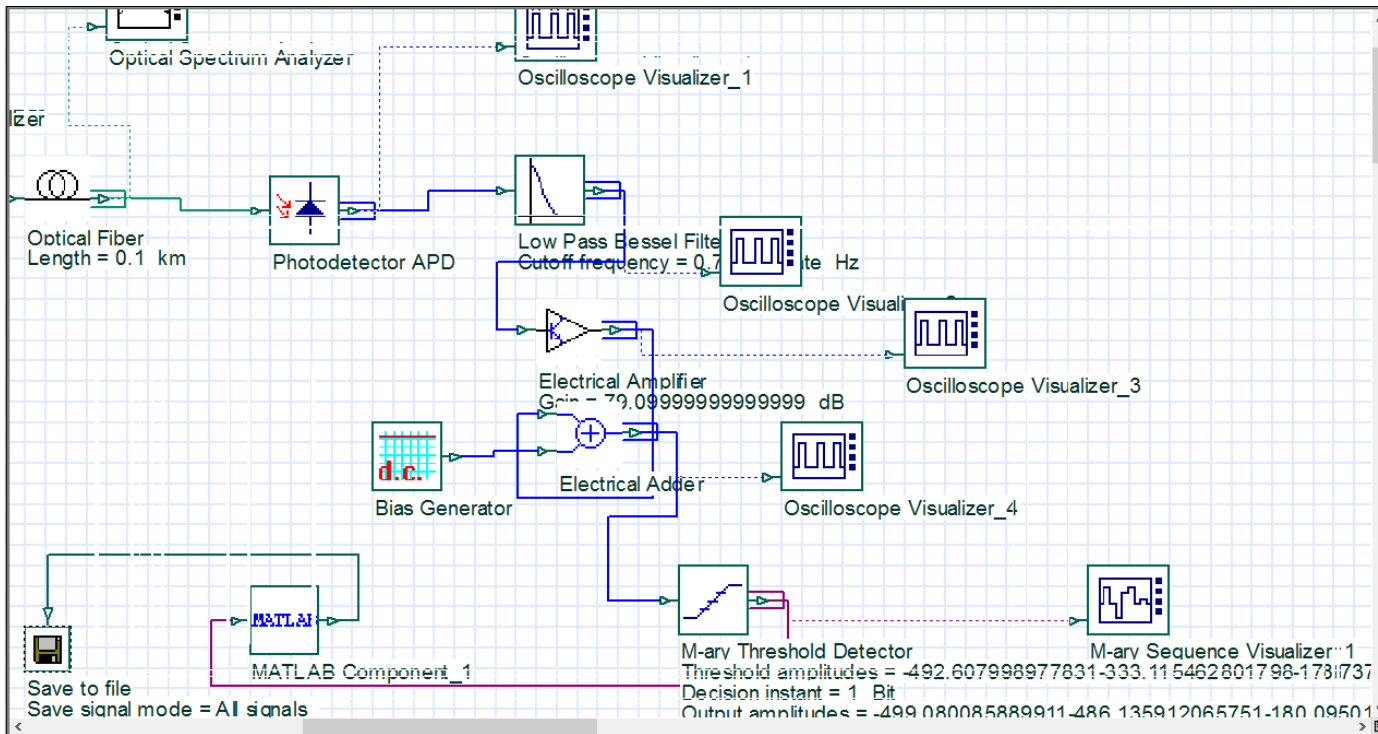


Figure 104 100m test-receiver blocks

9.3.2.2. Results

VARIABLE		SELECTION		
BER				
1x1 double				
	1	2	3	
1	0			

Figure 105 100m BER results

Data sent from transmitter block VS data received at the receiver block:

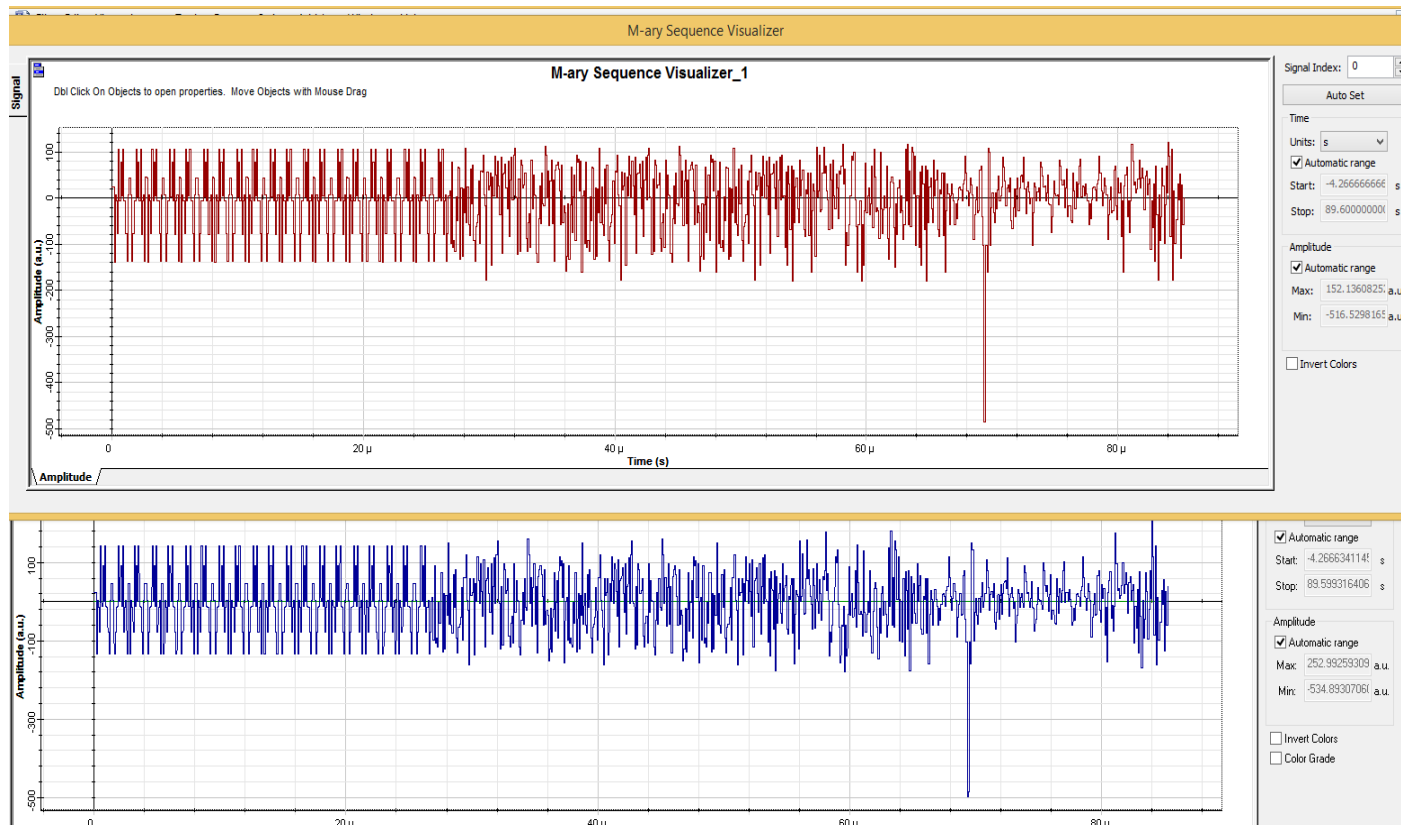


Figure 106 Data sent before channel and after

Opti-system simulations was done till 90km with BER equals to zero with amplifier gain value equals to 80 db.

To reach longer distance with the same BER amplifier gain was increased till 100db, that resulted in reaching 140 km with BER equals to zero.

More results to opti-system simulation results showing the increase in distance by step equals to 10km is shown at the appendix and some blocks graphs that show the effect of the optical channel.

Appendix

1-Opti-System Block visualizer at 30km 128 bit sequence length

- Before channel vs before receiver

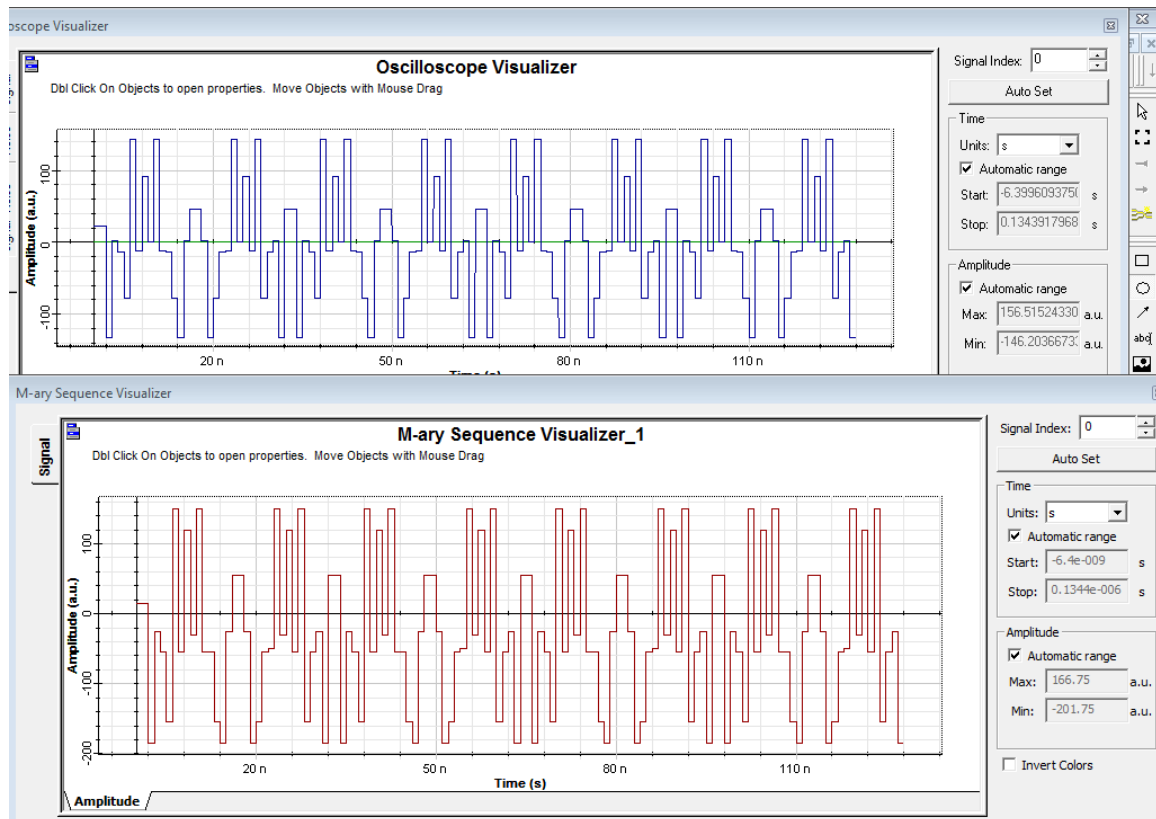


Figure 107

Simulations at 80 db

2-Opti-System Block visualizer at 90km 1024 bit sequence length

BER calculated=0

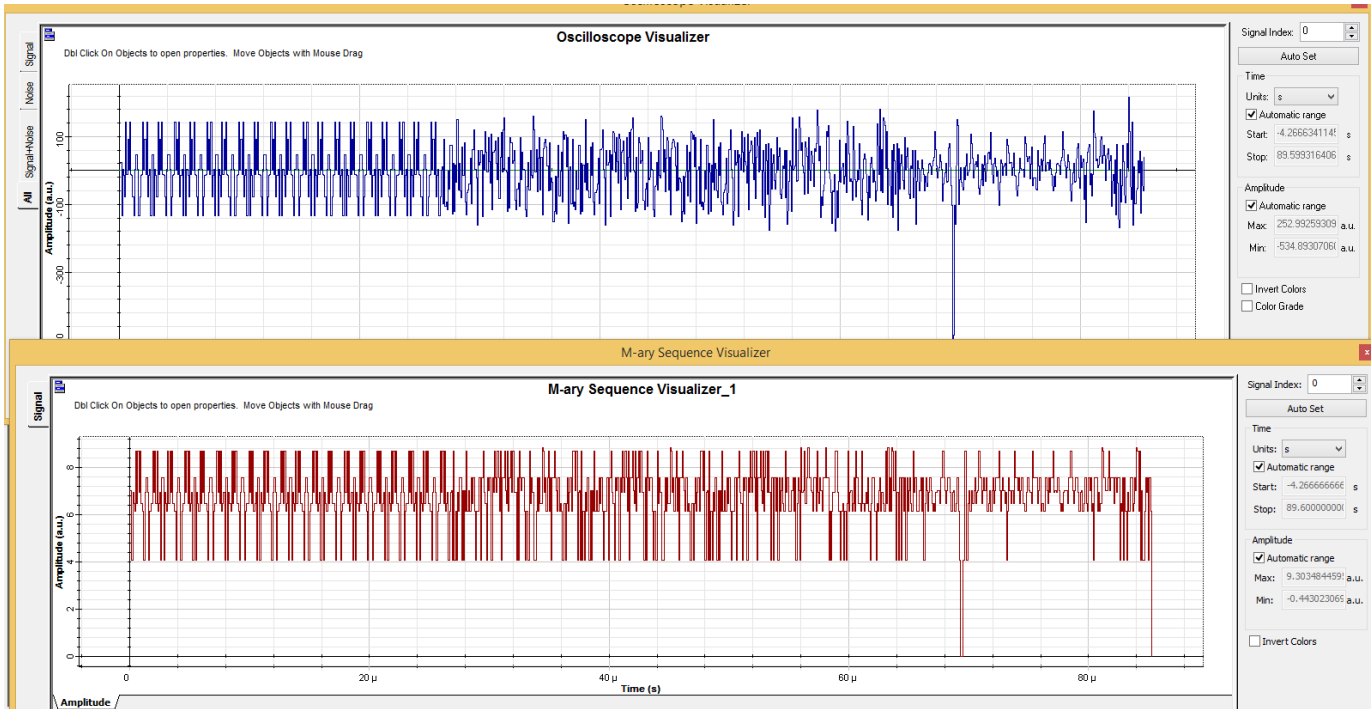


Figure 108

3-Opti-System Block visualizer at 100km 1024 bit sequence length
 As shown signal could not detected right at 80 db, which resulted in a BER= 0.2917

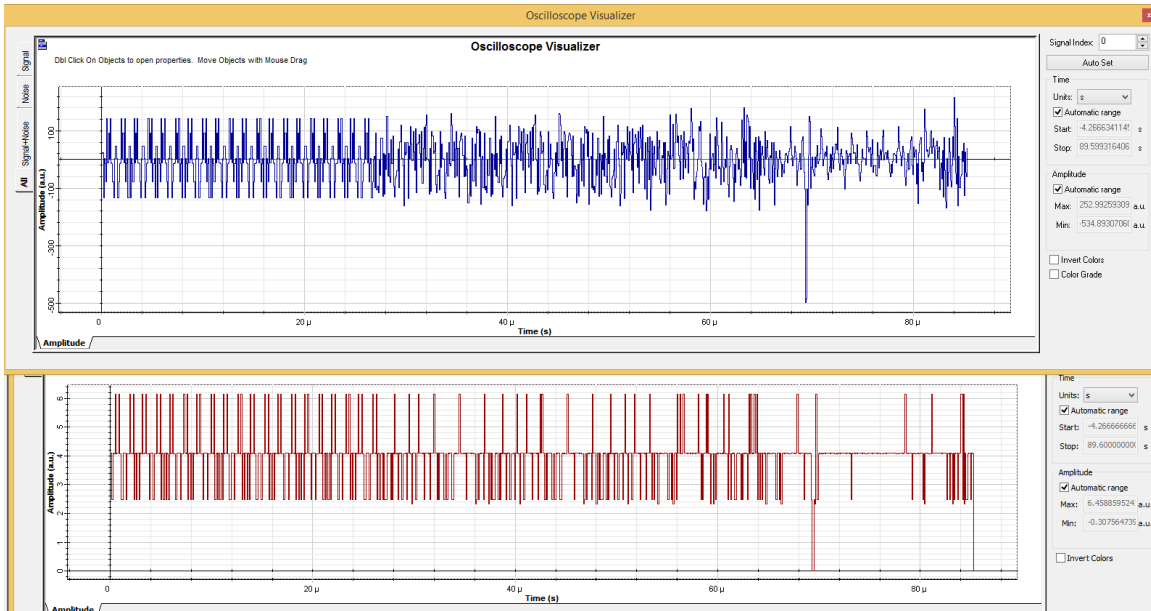


Figure 109

At 100 db gain

1-Opti-System Block visualizer at 100km 1024 bit sequence length

Here signal calculated BER=0;

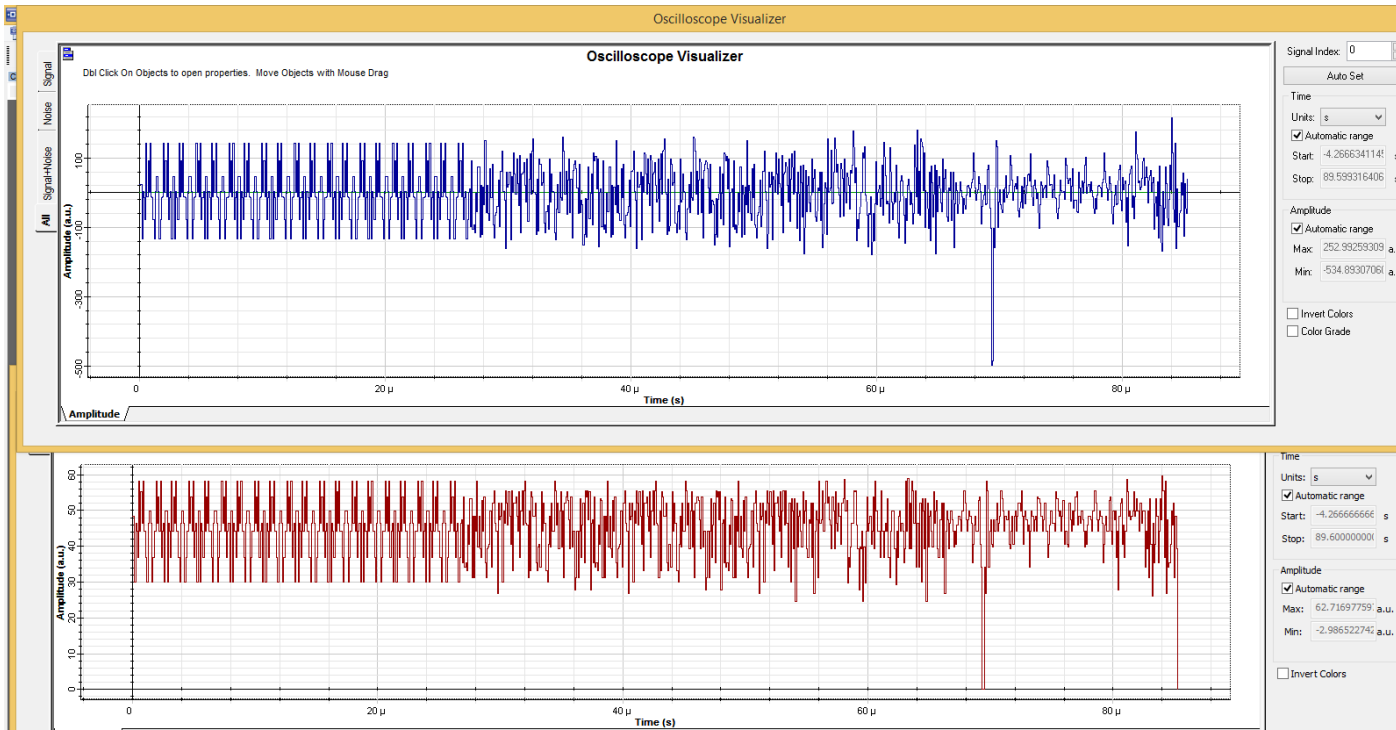


Figure 110

2-Opti-System Block visualizer at 140km 1024 bit sequence length
 Last detected BER=0;

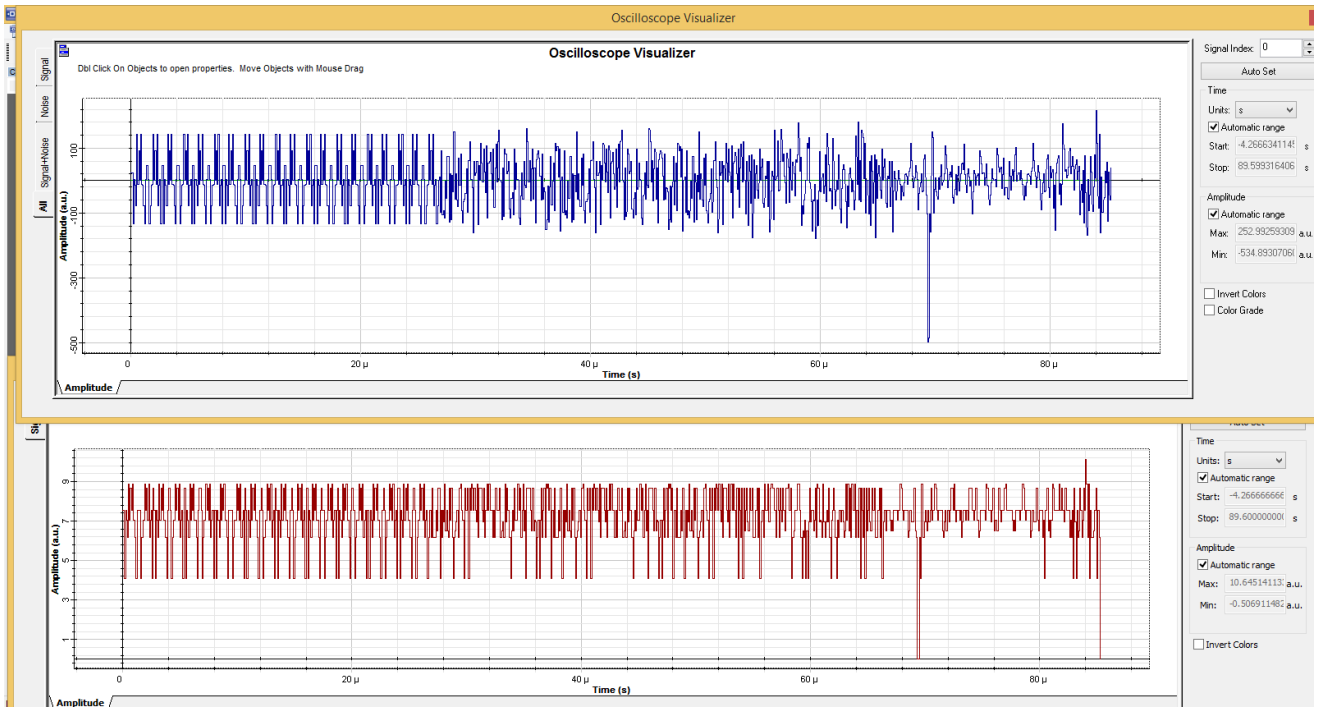


Figure 111

3-Opti-System Block visualizer at 150km 1024 bit sequence length
 BER =0.4063;

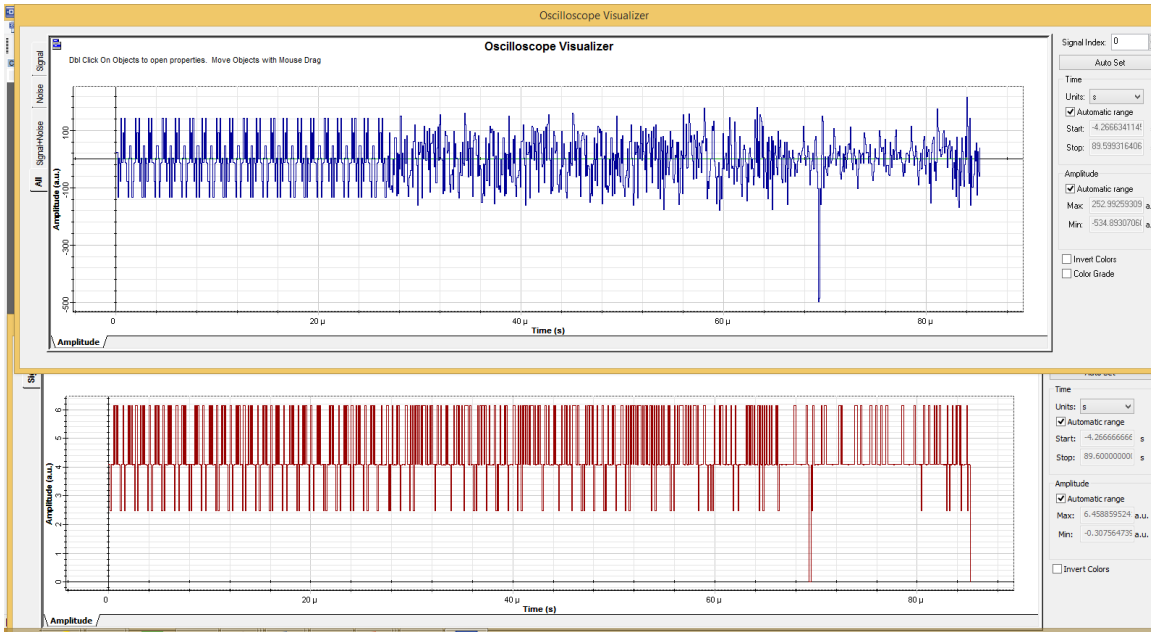


Figure 112

4-Opti-System Block visualizer at 170km 1024 bit sequence length
Signal could not detected right.

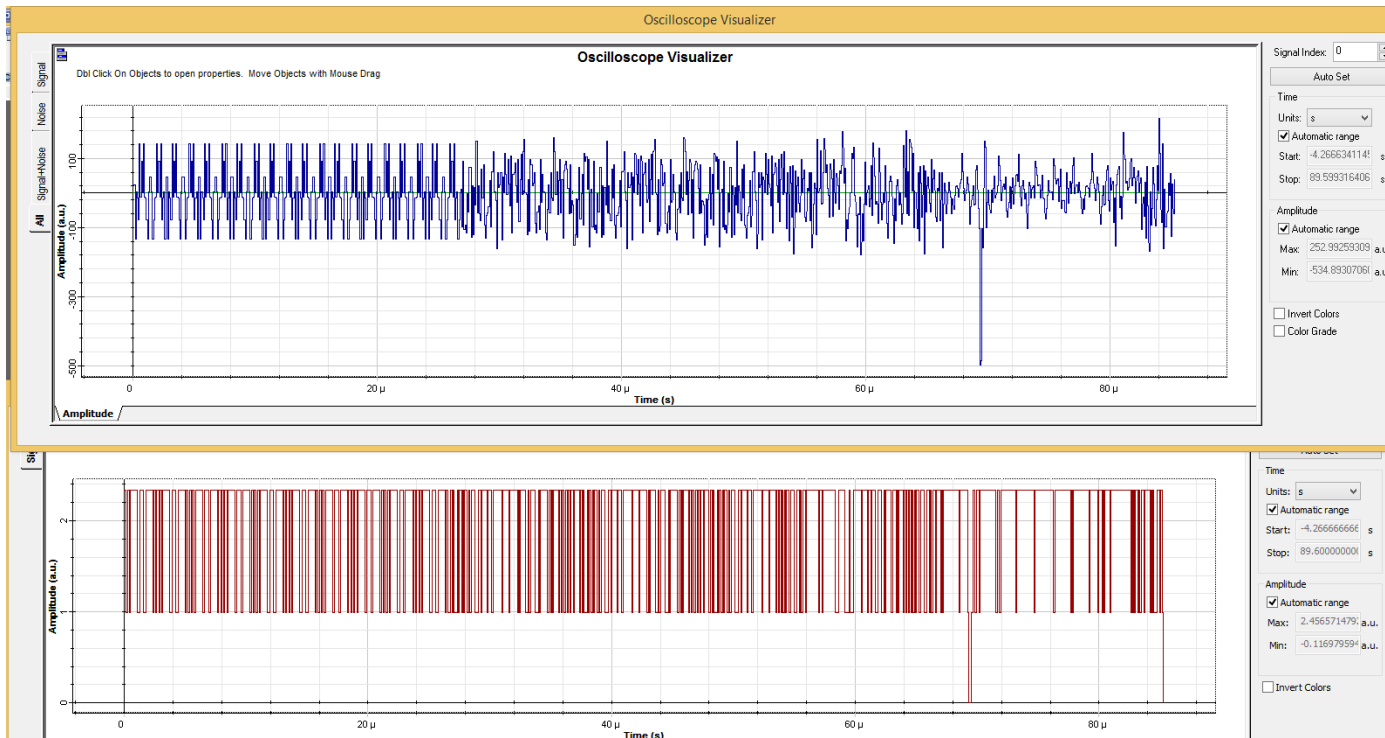


Figure 113

Sample Opti-system Matlab code

```

OutputPort1 = InputPort1;
format long
Data_Rate=12;
R=1/2;
Nbpsc=2;
Ncbps=96;
Ndbps=48;
Nfft=64;
Ts=50*10^-9;
modulation_scheme='QPSK';

cs = length(InputPort1.Sequence);

if( cs > 0 )
    PSDU=InputPort1.Sequence;
    DATA=data_framing(PSDU,Ndbps);           %%output DATA
    SIGNAL=signaling(PSDU,Data_Rate);         %%output signal
    encoded_signal=conv_encoderk7(SIGNAL,1/2); %%input
of viterbib---signal

```

```

        encoded_data=conv_encoderk7(DATA,R);          %%input
of viterbib-----data
        interleaved_signal=interleaver(encoded_signal,48,1); %%input of
dienterliever signal
        interleaved_data=interleaver(encoded_data,Ncbps,Nbpsc); %%input
of dienterliever data
        signal_symbols=modulator('BPSK',interleaved_signal,48); %%input
of demapper signal
        data_symbols=modulator(modulation_scheme,interleaved_data,Ncbps); %%input of demapper data

        mod_scheme = 'QPSK';
        received_PSDU=scramble(DATA,zeros(1,6));
        received_PSDU(1:length(PSDU)+17-1)
%       system_test;

        %plot(real(data_symbols),imag(data_symbols),'ro')
        carriers_symbols=insert_pilots(signal_symbols,data_symbols);
        ready_to_ifft=insert_padding(carriers_symbols);
        transmitted_symbols=perform_ifft(ready_to_ifft);
        transmitted_ofdm=add_cyclic(transmitted_symbols);

        preamble=form_preamble;
        transmitted_ofdm=[preamble transmitted_ofdm];
        transmitted_cont=DAC(transmitted_ofdm,10);
        [Real_Fixed,Imag_Fixed]=fixed([0.04*ones(1,18)+1j*0.04*ones(1,18)
transmitted_cont]);

        sent=transmitted_cont;
        %sent=0.1*ones(1,1000);
        for i=1:length(sent)
            data2(2*i-1)=real(sent(i));
            data2(2*i)=imag(sent(i));
        end
        data2;
        data3=1000.*data2;
        OutputPort1.Sequence= data3;
end

```


References

(n.d.). Retrieved from http://www.ele.uri.edu/courses/ele432/spring08/photo_detectors.pdf

(n.d.). Retrieved from BICSI South West Regional Meeting

(n.d.). Retrieved from <http://www.tutorvista.com/content/science/science-ii/refraction-light/refraction-light.php>

(n.d.). Retrieved from http://www.optics.rochester.edu/workgroups/lukishova/QuantumOpticsLab/2012/OPT_101/Quantum%20A%20Lab%20Presentation%202012-1.pdf

(n.d.). Retrieved from https://en.wikipedia.org/wiki/Access_network

class, 2. (2013). *OFDM Thesis*.

Djordjevic, W. S. (2010). *OFDM for Optical Communications*.

Dutton, H. J. (1998). *Understanding Optical Communications*.

Fuertes, L. Z. (2005-03-22). *OFDM PHY Layer Implementation based on the 802.11a Standard and System*.

RCDD, J. C. (n.d.). *Introduction to Wave Division Multiplexing*.

Special thanks to

Dr Hassan Mostafa

Dr Tawfik Ismail

Prof.: Serag El-Din Habib

Thanks for those who helped us:

Eng. Karim Ismail

Eng. Ahmed Sadek

Eng. Yasmin

Eng. Yahia Ramadan

And for every one helped us