VLSI IMPLEMENTATION OF 2×2 MIMO-OFDM SYSTEM ON FPGA

By

PREMALATHA.R

Reg. No. 1020106014

of

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University of Technology, Coimbatore)

COIMBATORE – 641 049

A PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfilment of the requirements
for the award of the degree
of

MASTER OF ENGINEERING
in
APPLIED ELECTRONICS
APRIL 2012

BONAFIDE CERTIFICATE

Certified that this project report titled **"VLSI IMPLEMENTATION OF 2×2 MIMO-OFDM SYSTEM ON FPGA"** is the bonafide work of **Ms. PREMALATHA.R (Reg.No:1020106014)** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report for dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Project Guide	**Head of the Department**
Ms.M.Shanthi	Dr. Rajeswari Mariappan

The candidate with University Register No. 1020106014 was examined by us in Project Viva-Voce examination held on _ _ _ _ _ _ _ _ _

Internal Examiner	**External Examiner**

ACKNOWLEDGEMENT

First I would like to express my praise and gratitude to the Lord, who has showered his grace and blessing enabling me to complete this project in an excellent manner.

I express my sincere thanks to our beloved Director **Dr.J.Shanmugam Ph.D.,** Kumaraguru College of Technology, for his kind support and for providing necessary facilities to carry out the work.

At this pleasing moment of having successfully completed the project work, I wish to acknowledge my sincere gratitude and heartfelt thanks to our beloved principal **Dr.S.Ramachandran Ph.D.,** for having given me the adequate support and opportunity for completing this project work successfully.

I express my sincere thanks to **Dr.Rajeswari Mariappan Ph.D.,** the ever active, Head of the Department of Electronics and Communication Engineering, who rendering us all the time by helps throughout this project

In particular, I wish to thank and everlasting gratitude to the project coordinator **Ms.R.Hemalatha M.E.**, **Assistant Professor(SRG)** Department of Electronics and Communication Engineering for her expert counselling and guidance to make this project to a great deal of success. With her careful supervision and ensured me in the attaining perfection of work.

I extend my heartfelt thanks to my internal guide **Ms.M.Shanthi M.S.**, **Associate Professor** for her ideas and suggestion, which have been very helpful for the completion of this project work.

Last, but not the least, I would like to express my gratitude to my family members, friends and to all my staff members of Electronics and Communication Engineering department for their encouragement and support throughout the course of this project.

ABSTRACT

Multiple Input Multiple Output Orthogonal Frequency Division Multiplexing (MIMO-OFDM) technology is an attractive transmission technique for wireless communication systems with multiple antennas at transmitter and receiver. The core of this technology is that it divides one data stream to many. Hence, data rate, reliability and diversity can be increased along with the stability for multi-path signals. The aim of the project is to design MIMO-OFDM transceiver with good channel estimation method, efficient FFT/IFFT processor and better coding techniques. To reduce the hardware resources a multichannel MIMO-OFDM system is proposed using single FFT block that is shared across modulations for the system.

This project describes the efficient implementation of a Low-Power 64-point Pipeline FFT/IFFT processor adopting a single-path delay feedback style. The proposed architecture applies a reconfigurable complex multiplier and bit-parallel multipliers to achieve a ROM-less FFT/IFFT processor, thus consuming less power. Header-based channel estimation with maximum likelihood algorithm is chosen in consideration of hardware feature as well as communication theory for fast prototyping. The pipeline architecture here includes the simple logic of one adder and channel memories without redundancy. Thus reducing the complexity from O (n2) to O (1), it saves 43 percent of the hardware resources and achieves a better performance in the architecture.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

MIMO	Multiple Input Multiple Output
OFDM	Orthogonal Frequency Division Multiplexing
FPGA	Field-Programmable Gate Arrays
STBC	Space Time Block Code
STTC	Space Time Trellis Code
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
RAM	Random Access Memory
PE	Processing Element
DIF	Decimation In Frequency
DIT	Decimation In Time
SFG	Signal Flow Graph
ROM	Read Only Memory
SDF	Single-path Delay Feedback
MDC	Multiple-path Delay Commutator
DFT	Discrete Fourier Transform
DSP	Digital Signal Processing
QPSK	Quadrature Phase Shift Keying

CHAPTER 1

INTRODUCTION

Multiple Input Multiple Output (MIMO) antenna architecture has the ability to increase capacity and reliability of a wireless communication system. Space Time Coding and spatial multiplexing are two main categories of MIMO system. Space Time Coding is able to increase the reliability of the wireless communication link. While spatial multiplexing or layered space-time coding can linearly increase the data rate of the wireless system. Orthogonal Frequency Division Multiplexing (OFDM) is well-known for efficient high speed transmission and robustness to frequency selective channels. Hence, the integration of these two technologies has the potential to meet the ever growing demands of future communication systems. If space-time coding is used at the transmitter, the channel knowledge is required at the receiver to decode the transmitted symbols. Therefore, accurate channel estimation plays a key role in data detection especially in MIMO-OFDM system. MIMO-OFDM system is used in many applications like Wi-Max, Wi-Fi, WLANs, and many more applications.

1.1 MOTIVATION

In recent years the telecommunications industry has experienced phenomenal growth, particularly in the area of wireless communication. This growth has been fueled by the widespread popularity of mobile telephones and wireless computer networking. However, there are limits to growth, and the radio spectrum used for wireless communications is a finite resource. Therefore considerable effort has been invested in making more efficient use of it. Using the spectrum more efficiently caters for the ever increasing demand for faster communications since more bits per second can be transmitted using the same bandwidth. Recently a major research focus in this area has been the use of multiple antennas for transmitting and receiving instead of the traditional single antenna systems. It has been proposed that using multiple transmit and receive antennas, orthogonal frequency division multiplexing and associated coding techniques could increase the performance of wireless communication systems

1.2 OVERVIEW

MIMO-OFDM system consists of scrambler/descrambler, convolution encoder/decoder, interleaver/de-interleaver, modulator/ demodulator, or space time encoder/decoder, FFT/IFFT Processor, and channel estimation. This project aims at design and Source bit implementation of a simple and most efficient channel estimation method and a good modulation technique for increasing the channel capacity, bandwidth, increasing bit rates and eliminates inter-symbol interference.

1.3 SOFTWARE USED

- ModelSim XE 111 6.2g
- Xilinx ISE 9.2i

1.4 ORGANIZATION OF THE REPORT

- **Chapter 2** discusses about the introduction to MIMO-OFDM system
- **Chapter 3** discusses about Baseline MIMO-OFDM system
- **Chapter 4** discusses the low power 64-point pipeline FFT/IFFT Processor.
- **Chapter 5** discusses the channel estimation.
- **Chapter 6** discusses the Proposed MIMO-OFDM system
- **Chapter 7** discusses the simulation results and synthesis report.
- **Chapter 8** shows the conclusion and future scope of the project.

CHAPTER 2

INTRODUCTION TO MIMO-OFDM SYSTEM

2.1 MIMO

One of the drawbacks of wireless communication is fading. To improve the reliability of wireless communication and to reduce the effects of fading, diversity technique is employed. Diversity is the technique, in which the same information is transmitted across multiple channels to achieve higher reliability. It operates on the principle that it is unlikely that all of the channels used to transmit the redundant information will be experiencing deep fading at the same time. Even if one particular channel is unusable, the information may still be recovered from the redundant transmission over the other channels. Therefore the overall reliability of the communication system is improved at the cost of transmitting redundant information.

If multiple antennas are used at the transmitter or receiver, then there occur multiple transmission channels between the transmitter and receiver. In Figure 2.1, the potential channels in a 2×2 MIMO system is shown. These multiple channels can be used to exploit transmit and receive diversity.

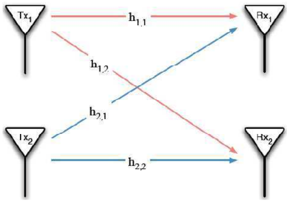


Figure 2.1 2x2 MIMO system

Receive diversity is a diversity technique in which the same information is received by different antennas. For instance the information sent from Tx1 is transmitted across channels $h_{1,1}$ and $h_{1,2}$, and received by both Rx1 and Rx2. In Transmit diversity, the same information is sent from multiple transmit antennas. One possible way to achieve this is to code across multiple symbols periods. For instance, at time t antenna Tx1 could transmit the symbol s then at time t+1 antenna Tx2 would transmit the same symbol, s. The Alamouti's scheme uses a method similar to this to obtain transmit diversity.

MIMO systems are able to achieve impressive improvements in reliability and capacity by exploiting the diversity offered by the multiple channels between the transmit and receive antennas. Different coding schemes vary in their exact approaches. However all seek to use the available channels to increase capacity and/or reliability.

2.2 OFDM

Frequency Division Multiplexing (FDM) transmits multiple signals simultaneously over a single path. Each signal has a unique frequency range (carrier). Orthogonal FDM (OFDM) is a special case of FDM where a single data stream is distributed over several lower rate sub-carriers. In other words, one signal is transmitted by multiple carriers. Sub-carriers are separated by given frequency ranges, to avoid cross-carrier interference. The benefit of orthogonality is that it gives a high spectral density (maximizing channel usage).OFDM is modulation method known for its capability to mitigate multipath.

In OFDM, the high speed data stream is divided into N_c narrowband data streams, where N_c corresponds to the subcarriers or sub channels, i.e. one OFDM symbol consists of N symbols modulated for example by QAM or PSK. As a result the symbol duration is N times longer than in a single carrier system with the same symbol rate. The symbol duration is made even longer by adding a cyclic prefix to each symbol. As long as the cyclic prefix is longer than the channel delay spread OFDM offers inter-symbol interference (ISI) free transmission. Delay spread is the time spread between the arrival of the first and last multipath signal seen by the receiver. In a digital system, the delay spread can lead to inter-symbol interference.

Another key advantage of OFDM is that it dramatically reduces equalization complexity by enabling equalization in the frequency domain. OFDM, implemented with IFFT at the transmitter and FFT at the receiver, converts the wideband signal, affected by frequency selective fading, into N narrowband flat fading signals thus the equalization can be performed in the frequency domain by a scalar division carrier-wise with the subcarrier related channel coefficients. The channel should be known or learned at the receiver.

2.3 MIMO-OFDM

MIMO-OFDM combines OFDM and MIMO techniques thereby achieving spectral efficiency and increased throughput. A MIMO-OFDM system transmits independent OFDM modulated data from multiple antennas simultaneously. At the receiver, after OFDM demodulation, MIMO decoding on each of the sub-channels extracts the data from all the transmit antennas on all the sub-channels. MIMO-OFDM is the corner stone of future broadband wireless access.

2.3.1 ADVANTAGES OF MIMO-OFDM

- Less interference
- Diversity gain
- Increase data capacity
- Power efficiency
- Bandwidth gain

CHAPTER-3

BASELINE MIMO-OFDM SYSTEM

The architecture of the transmitter and the receiver is illustrated in Figure 3.1 and Figure3.2 respectively. Binary data is scrambled and then encoded by conventional encoder. Coded bit stream mapped to a constellation by digital QPSK modulator, interleaved and encoded by STBC encoder. Each of parallel output symbol stream corresponding to a certain transmitter antenna follows the same Transmit process:

- ➢ Insertion of pilot symbols (synchronization)
- ➢ Modulation by inverse FFT
- ➢ Attachment of CP and Preamble

Finally, data frame is transferred for transmitter antenna.

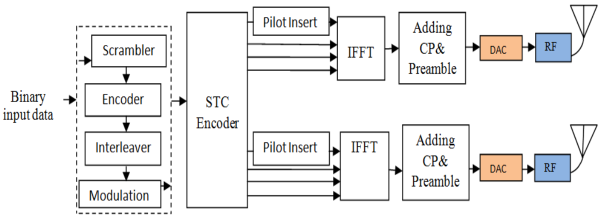


Figure 3.1 System architecture of the transmitter

The received symbol streams from different received antennas are first synchronized. Preambles and CPs are extracted from receiver symbol stream. Remaining OFDM symbols are demodulated by FFT. Frequency pilots are extracted from the demodulated OFDM symbols, and are used for channel estimation. Estimated channel matrix aids the MIMO decoder. Estimated transmitted symbols are demodulated and decoded. Preamble is used for synchronization.

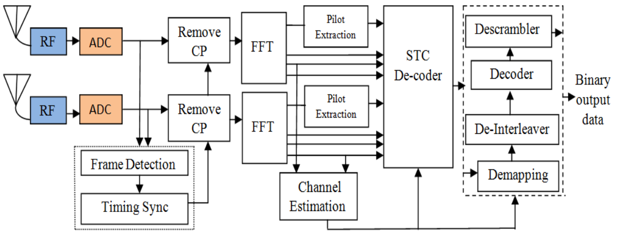


Figure 3.2 System architecture of the receiver

3.1 CYCLIC PREFIX

Cyclic Prefix (CP) is inserted between two successive symbols, which not only mitigates Inter Symbol Interference (ISI), but also converts the linear convolution between the transmitted OFDM symbol and channel impulse response to a circular one. At the receiver, the CP corrupted by ISI is generally discarded and the ISI free part of the OFDM symbol is used for channel estimation and data detection. Cyclic prefix is nothing but preappending the last part of the signal to the beginning of the signal. Figure 3.3 shows the structure of cyclic prefix.

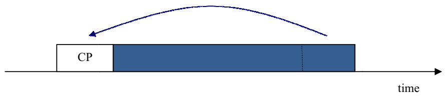


Figure 3.3 Cyclic prefix

Passing a sinusoidal through a multipath channel does not affect the frequency of the sinusoidal. As the multipath channel does not change the frequency of the sinusoidal, multipath channel does not affect orthogonality of the subcarriers. Transmission of cyclic prefix reduces the data rate. Hence the cyclic prefix duration should not be much more than the duration of the maximum expected multipath channel.

3.2 QPSK MODULATOR

Phase shift keying is a digital modulation technique, perhaps one of the most used in digital communication systems. In PSK modulation scheme, the phase of the carrier is altered in accordance with the input binary coded information.

Modulation is very important block in communication system, used

- to transmit the data through channel without loss of data
- to reduce size of antenna incase of wireless communication
- to reduce the channel distortion & to use in RF communication

In QPSK, the data bits to be modulated are grouped into symbols, each containing two bits, and each symbol can take on one of four possible values: 00, 01, 10, or 11. During each symbol interval, the modulator shifts the carrier to one of four possible phases corresponding to the four possible values of the input symbol. In the ideal case, the phases are each 90 degrees apart, and these phases are usually selected such that the signal constellation matches the configuration shown in Figure 3.4. Inputs and outputs of QPSK modulator in Table 3.1.

Input Bits	I-Out	Q-Out
00	-1	-1
01	-1	1
10	1	-1
11	1	1

Table 3.1 Inputs and outputs of QPSK modulator

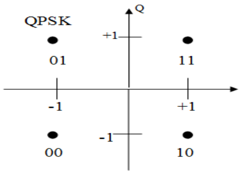


Figure 3.4 QPSK constellation mapping

3.3 SCRAMBLER/DESCRAMBLER

Scramblers transform the input data stream by applying a pseudo-random binary sequence (PRBS). A pre-calculated PRBS stored in the Read-only memory is used, but more often it is generated by a linear feedback shift register (LFSR). The generated data stream is reversed by a descrambler at the receiving side to get the original input data stream.

There are two main reasons for using scrambling:

- It facilitates the work of a timing recovery circuit, an automatic gain control and other adaptive circuits of the receiver (eliminating long sequences consisting of '0' or '1' only).
- It eliminates the dependence of a signal's power spectrum upon the actual transmitted data, making it more dispersed to meet maximum power spectral density requirements.

3.4 INTERLEAVER/DEINTERLEAVER

Interleaver arranges the symbols into a group of transmitted code words so that adjacent symbols in the data stream are not from the same codeword. Deinterleaver reassembles the code words when the deinterleaver processes the data stream.

A convolutional interleaver consists of N rows of shift registers, with different delay in each row. In general, each successive row has a delay which is J symbols duration higher than the previous row. The code word symbol from the encoder is fed into the array of shift registers, one code symbol to each row. With each new code word symbol the commutator switches to a new register and the new code symbol is shifted out to the channel. The i-th ($1 \leq i \leq N-1$) shift register has a length of (i-1)J stages where J = M/N and the last row has M-1 numbers of delay elements. The convolutional deinterleaver performs the inverse operation of the interleaver and differs in structure of the arrangement of delay elements. Zeroth row of interleaver becomes the N-1 row in the deinterleaver. 1st row of the former becomes N-2 row of later and so on. . This process of interleaving and deinterleaving data reduces the effects of burst errors (multiple errors in a row) as it rearranges codeword symbols and spreads the errors among multiple code words.

3.5 PILOT TONE

A pilot tone is a reference data which is multiplexed along with the data stream for channel estimation. These tones are known to the receiver. The idea behind this method is to exploit the knowledge of the transmitted pilot symbols at the receiver to estimate the channel.

There are two types of pilot arrangement:

- Block pilot arrangement
- Comb pilot arrangement

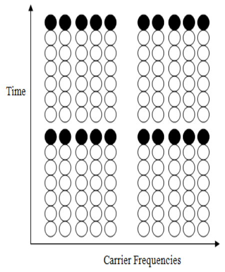


Figure 3.5 Block Pilot

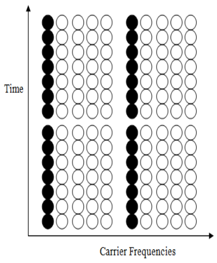


Figure 3.6 Comb Pilot

For block fading channel, when the channel is constant over the few OFDM symbols, the pilots are transmitted on all sub-carriers in periodic intervals of all OFDM blocks. This type of pilot arrangement, depicted in Figure 3.5, is called the block type arrangement. For fast fading channel, when the channel changes between adjacent OFDM symbols, the pilots are transmitted at all times but with an even spacing on all sub-carriers, representing a comb type pilot arrangement, Figure3.6. The channel estimates from the pilot sub-carriers are interpolated to estimate the channel at the data sub-carriers.

The model adopted for the channel estimation is the block - type pilot insertion in which OFDM channel estimation symbols are transmitted regularly and all sub-carriers are employed as pilots. If the channel is invariable during the block, there will be no error in the channel estimation as the pilots are sent at all carriers.

3.6 SPACE TIME CODE

A Space–Time Code (STC) is a method employed to improve the reliability of data transmission in wireless communication systems using multiple transmit antennas. Jointly encodes the data streams over antennas, and therefore aims to maximize diversity gain. The Space-Time refers to coding across space and time. Coding across space by using multiple transmit and receive antennas, and across time by using multiple symbol periods.

There are two main space-time coding schemes:

- STTC obtains coding and diversity gain due to coding in ST dimension.
- STBC is based on orthogonal design and obtains full diversity gain with low decoding complexity.

STC may be further subdivided according to whether the receiver knows the channel impairments. In coherent STC, the receiver knows the channel impairments through training or some other form of estimation. These codes have been studied more widely because they are less complex than their non-coherent counterparts. In non-coherent STC the receiver does not know the channel impairments but knows the statistics of the channel.

Space–time trellis codes (STTCs) are a type of space–time code used in multiple-antenna wireless communications. This scheme transmits multiple, redundant copies of a trellis (or convolution) code distributed over time and a number of antennas ('space'). These multiple, 'diverse' copies of the data are used by the receiver to attempt to reconstruct the actual transmitted data. For an STC to be used, there must necessarily be multiple transmit antennas, but only a single receive antennas is required; nevertheless multiple receive antennas are often used since the performance of the system is improved by so doing.

In contrast to space–time block codes (STBCs), they are able to provide both coding gain and diversity gain and have a better bit-error rate performance. However, being based on trellis codes, they are more complex than STBCs to encode and decode; they rely on a viterbi decoder at the receiver where STBCs need only linear processing. Coding gain is the measure in the difference between the signal to noise ratio levels between the un-coded system and coded system required to reach the same bit error rate levels when used with the error correcting code. Diversity gain is the increase in signal-to-interference ratio due to some diversity scheme, or how much the transmission power can be reduced when a diversity scheme is introduced, without a performance loss. Diversity gain is usually expressed in decibel, and sometimes as a power ratio.

Space Time Block Coding is a technique used in wireless communications to transmit multiple copies of a data stream across a number of antennas and to exploit the various received versions of the data to improve the reliability of data-transfer. The fact that the transmitted signal must traverse a potentially difficult environment with scattering, reflection, refraction and so on and may then be further corrupted by thermal noise in the receiver means that some of the received copies of the data will be 'better' than others. This redundancy results in a higher chance of being able to use one or more of the received copies to correctly decode the received signal. In fact, space–time coding combines *all* the copies of the received signal in an optimal way to extract as much information from each of them as possible.

STBCs can be described by a code matrix, which defines what is to be sent from the transmit antennas during transmission of a block. The code matrix is of dimension $N_t \times t_b$ where

N_t is the number of transmit antennas and t_b is the number of symbol periods used to transmit a block. So the rows of the matrix represent the transmit antennas, and the columns are the time (symbol) periods.

The space-time block code transmission matrix is

$$G_2 = \begin{pmatrix} x_1 & x_2 \\ -x_2^* & x_1^* \end{pmatrix}$$

The most common form of STBC coding, known as Alamouti's coding, is the foundation for the space-time coding used in the IEEE 802.11n draft. Like normal block codes the alamouti code operates on blocks of input bits, however rather than having 1 dimensional code vectors it has 2 dimensional code matrices. Using this space-time code, the first spatial stream transmits symbols: $x1$ and $x2$ (in time), and the second transmits: $-x_2^*$ and x_1^*. STBC is used in the system because it has less complex implementation terms than STTC.

CHAPTER-4

A LOW POWER 64 POINT FFT/IFFT PROCESSOR

Discrete Fourier Transform (DFT) is a very important technique in modern Digital Signal Processing (DSP) and telecommunications, especially for applications in Orthogonal Frequency Demodulation Multiplexing (OFDM) systems, such as IEEE 802.11a/g, Worldwide Interoperability for Microwave Access (WiMAX), Long Term Evolution(LTE), and Digital Video Broadcasting—Terrestrial(DVB-T). However, DFT is computational intensive and has a time complexity of O (N^2). The Fast Fourier Transform (FFT) was proposed by Cooley and Tukey to efficiently reduce the time complexity to O ($N\log_2 N$), where N denotes the FFT size.

The implementations can be mainly classified into memory-based and pipeline architecture styles. Memory-based architecture is widely adopted to design an FFT processor, also known as the single Processing Element (PE) approach. This deign style is usually composed of a main PE and several memory units, thus the hardware cost and the power consumption are both lower than the other architecture style. However, this kind of architecture style has long latency, low throughput, and cannot be parallelized. On the other hand, the pipeline architecture style can get rid off the disadvantages of the foregoing style, at the cost of an acceptable hardware overhead.

Generally, the pipeline FFT processors have two popular design types. One uses a Single-path Delay Feedback (SDF) pipeline architecture, and the other uses a Multiple-path Delay Commutator (MDC) pipeline architecture. The Single-path Delay Feedback (SDF) pipeline FFT is good in its requiring less memory space (about N-1 delay elements) and its multiplication computation utilization being less than 50%, as well as its control unit being easy to design. Such implementations are advantageous to low-power design, especially for applications in portable DSP devices. Based on these reasons, the SDF pipeline FFT is adopted in our work. However, the FFT computation often needs to multiply input signals with different twiddle factors for an outcome, which results in higher hardware cost because a large size of ROM is needed to store the wanted twiddle factors. Therefore, to throw off these ROM's for area-efficient consideration, proposed an efficient ROM-less FFT/IFFT processor. The complex multipliers used in the processor are realized with shift-and-add operations. Hence, the processor uses only a two-input digital multiplier and does not need any ROM for internal storage of coefficients. However, low speed and higher hardware cost caused by the proposed complex multiplier are the pay-off. A smart structure for ROM-size reduction to produce twiddle factors as well as to compact the chip area.

In order to further improve the power consumption and chip area of previous works, proposed efficient radix-2 pipeline architecture with low power consumption for the FFT/IFFT processor. The proposed architecture includes a reconfigurable complex constant multiplier and bit-parallel complex multipliers instead of using ROM's to store twiddle factors, which is suited for the power-of-2 radix style of FFT/IFFT processors.

4.1 FFT AND IFFT ALGORITHMS

The discrete Fourier transform (DFT) X_k of an N-point discrete-time signal x_n is defined by:

$$X_k = \sum_{n=0}^{N-1} x_n W_N^{nk} \quad , \quad 0 \leq k \leq N-1. \qquad (4.1)$$

where

$$. W_N^{nk} = e^{-j2\pi nk/N}$$

However, a straightforward implementation of this algorithm is obviously impractical due to the huge hardware required. Therefore, the fast Fourier transform (FFT) was developed to efficiently speed up its computation time and significantly reduce the hardware cost. Generally, FFT analyzes an input signal sequence by using a Decimation-In-Frequency (DIF) or Decimation-In-Time (DIT) decomposition to construct an efficiently computational Signal-Flow Graph (SFG). Here, employs a DIF decomposition because it matches the manipulation manner of single-path delay pipeline facility. An example of radix-2 DIF FFT SFG for $N = 16$ is depicted in Figure 4.1.

The radix-2 DIF FFT described above appears regularity in SFG and has less complex multipliers required. Thus, it is suited for hardware implementation, because some complex multiplications can be simplified to reduce the chip area.

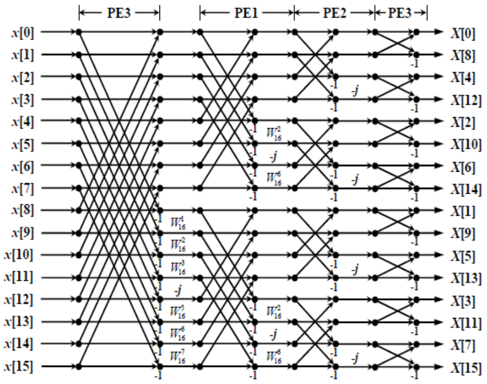


Figure 4.1 DIF FFT signal flow graph

For instance, an input signal multiplied by in Figure 4.1 can be expressed as:

$$- \qquad (4.2)$$

Where, (a+jb) denotes the discrete time signal complex form.

Similarly, the complex multiplication of is given by:

$$- \qquad (4.3)$$

Both these above equations will ease hardware implementation in the future, because they only need to calculate the multiplication by $-$ and two real additions, respectively. Especially, the multiplication by $-$ can be obtained easily. The inverse discrete Fourier transform (IDFT) of length N is given by:

$$- \qquad (4.4)$$

To reuse the same hardware core for reducing the chip area the above equation can be rewritten as:

$$x_n = \frac{1}{N}\left(\sum_{k=0}^{N-1} X_k^* W_N^{nk}\right)^* \ , \ 0 \leq n \leq N-1 \qquad (4.5)$$

where the star symbol * denotes a conjugate. This new form can be viewed as a general DFT. In other words, DFT and IDFT can reuse the same hardware core, while IDFT requires some extra computations. These extra computations include conjugating the input data X_k and the outcomes of DFT, as well as dividing the previous output by N. Obviously, this new reuse version of DFT/IDFT algorithm will also simplify the design effort of an DFT/ IDFT processor and thus reduce the chip area, if both the DFT and IDFT processors are activated alternatively, and not simultaneously.

4.2 PROPOSED ARCHITECTURE

Traditional hardware implementation of FFT/IFFT processors usually employs a ROM to look up the wanted twiddle factors, and then word length complex multipliers to perform FFT computing. However, this introduces more hardware cost, thus a bit-parallel complex constant Multiplication scheme is used.

Besides, since the twiddle factors have a symmetric property, the complex multiplications used in FFT computation can be one of the following three operation types:

$$\text{Type 1:} W_N^k.(a+jb) = W_N^{k-\left(\frac{N}{4}\right)}.(b-ja),\ N/4 < k < N/2 \qquad (4.6)$$

$$\text{Type 2:} W_N^k.(a+jb) = -W_N^{k-\left(\frac{N}{2}\right)}.(a+jb),\ N/2 < k < 3N/4 \qquad (4.7)$$

$$\text{Type 3:} W_N^k.(a+jb) = -W_N^{k-\left(\frac{3N}{4}\right)}.(b-ja),\ 3N/4 < k < N \qquad (4.8)$$

Arbitrary twiddle factor used in FFT can utilize these operation types to derive the wanted value, thus can significantly shorten the size of ROM used to store the twiddle factors. To further decrease the size of ROM, two extra operation types are included.

$$\text{Type 4:} W_N^k.(a+jb) = [W_N^{\left(\frac{N}{4}\right)-k}.(b+ja)]^*,\ 1 < k < N/4 \qquad (4.9)$$

$$\text{Type 5:} W_N^k.(a+jb) = [W_N^{\left(\frac{N}{2}\right)-k}.(b+ja)]^*,\ N/4 < k < N/2 \qquad (4.10)$$

A radix-2 64-point pipeline FFT/IFFT processor with low power consumption, as shown in Figure 4.2. The proposed architecture is composed of three different types of processing elements (PEs), a complex constant multiplier, delay-line (DL) buffers (as shown by a rectangle with a number inside), and some extra processing units for computing IFFT. Here, the conjugate for extra processing units is easy to implement, which only takes the 2's complement of the imaginary part of a complex value. The divided-by-64 module can be substituted with a barrel shifter. In addition, for a complex constant multiplier in Figure 4.2, a novel reconfigurable complex constant multiplier is used to eliminate the twiddle-factor ROM. This new multiplication structure thus becomes the key component in reducing the chip area and power consumption of proposed FFT/IFFT processor.

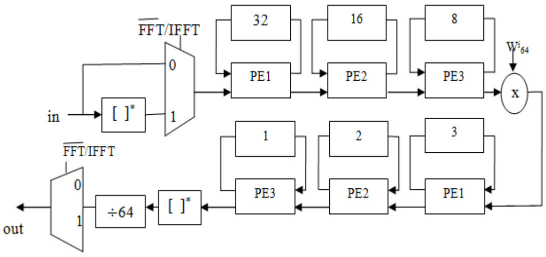


Figure 4.2 Radix-2 64 point pipeline FFT/IFFT processor

4.3 PROCESSING ELEMENTS

Based on the radix-2 FFT algorithm, the three types of processing elements (PE3, PE2, and PE1) used in our design are illustrated in Figure 4.3, Figure 4.4, and Figure 4.5, respectively. First, the PE3 stage is used to implement a simple radix-2 butterfly structure only, and serves as the submodules of the PE2 and PE1 stages. In the figure, I_{in} and I_{out} are the real parts of the input and output data, respectively. Q_{in} and Q_{out} denote the image parts of the input and output data, respectively. Similarly, DL_I_{in} and DL_I_{out} stand for the real parts of input and output of the DL buffers, and DL_Q_{in} and DL_Q_{out} are for the image parts, respectively.

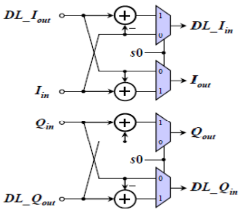


Figure 4.3 Circuit diagram of PE3 stage

As for the PE2 stage, it is required to compute the multiplication by *–j* or 1. Note that the multiplication by -1 in Figure 4.4 is practically to take the 2's complement of its input value. In the PE1 stage, the calculation is more complex than the PE2 stage, which is responsible for

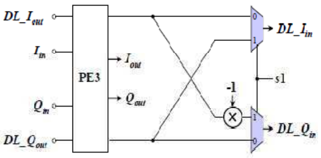


Figure 4.4 Circuit diagram of PE2 stage

computing the multiplications by *–j*, and , respectively. Since = - , itcan be given by either the multiplication by first and then the multiplication by *–j* or the reverse of the previous calculation. Hence, the designed hardware utilizes this kind of cascaded

calculation and multiplexers to realize all the necessary calculations of the PE1 stage. This manner can also save a bit-parallel multiplier for computing , which further forms a low-cost hardware.

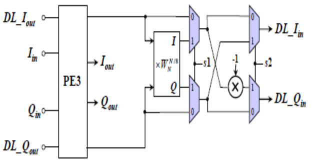


Figure 4.5 Circuit diagram of PE1 stage

4.4 BIT PARALLEL MULTIPLIERS

In FFT and IFFT algorithm, the multiplication by ‾ can employ a bitparallel multiplier to replace the wordlength multiplier and square root evaluation for chip area reduction.

The bit-parallel operation in terms of power of 2 is given by :

— = in × ((4.11)

If a straightforward implementation for the above equation is adopted, it will introduce a poor precision due to the truncation error, and will spend more hardware cost. Therefore, to improve the precision and hardware cost the above equation can be rewritten as

— (4.12)

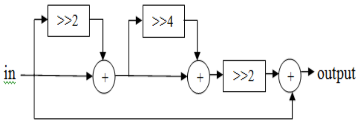


Figure 4.6 Circuit diagram for the bit-parallel multiplication by ‾

According to equation (4.12), the circuit diagram of the bit-parallel multiplier is illustrated in Figure 4.6. The resulting circuit uses three additions and three barrel shift operations. The realization of complex multiplication by using a radix-2 butterfly structure with its both outputs commonly multiplied by ‾, is shown in Figure 4.7. This circuit has just been used in the PE1 stage.

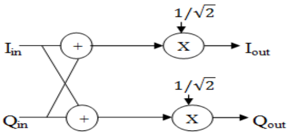


Figure 4.7 Circuit diagram of the multiplication by $W_N^{N/8}$

4.5 RECONFIGUREABLE COMPLEX CONSTANT MULTIPLIERS

Based on Equations (4.6)-(4.10), a reconfigurable low-complexity complex constant multiplier for computing is proposed, as shown in Figure 4.7 and Figure 4.8. This structure of this complex multiplier also adopts a cascaded scheme to achieve low-cost hardware. Here, the meaning of two input signals (I_{in} and I_{out}) and two output signals (Q_{in} and Q_{out}) are the same as the signals in the PE1 stage.

In Figure 4.8, this circuit is responsible for the computation of multiplication by twiddle factor in Figure 4.7, which is also an important circuit of FFT/IFFT processor. The word length multiplier used in Figure 4.8 adopts a low-error fixed-width booth multiplier for hardware cost reduction. Besides, there is no need to use bit-parallel multipliers to replace the word length one for two reasons. One is on the operation rate.

If bit-parallel multipliers are used, the clock rate is decreased due to the many cascaded adders. The other reason is the introduction of high wiring complexity because many bit-parallel multipliers are required to be switched for performing multiplication operations with different

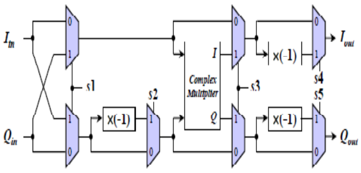


Figure 4.8 Proposed reconfigurable complex constant multiplier

twiddle factors Based on the above two reasons, the word length multiplier is still adopted to implement our complex constant multiplier under the consideration of operation speed and chip area. Note that our proposed complex constant multiplier will not introduce the issue of high hardware cost.

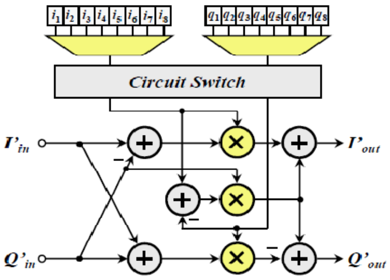


Figure 4.9 Complex multiplier used in Figure 4.8

A novel ROM-less and low-power pipeline 64-point FFT/IFFT processor is used for OFDM applications. Considering the symmetric property of twiddle factors in FFT, a reconfigurable complex constant multiplier such that the size of twiddle factor ROM is significantly shrunk, especially no ROM is needed. Thus the design owns lower hardware cost and power consumption. Of course, the proposed scheme can also be adapted to high-point FFT applications, with a lower size of twiddle-factor ROM's. The design can serve as a powerful FFT/IFFT processor in many other wireless communication systems.

CHAPTER-5

CHANNEL ESTIMATION

Channel estimation algorithms have a key role in signal detection in MIMO-OFDM systems. In this system, the number of channel components which need to be estimated is much more than conventional SISO wireless systems. Consequently, the computational process of channel estimation is highly intensive. In addition, the high performance channel estimation algorithms mostly suffer from high computational complexity. In the other words, the system undergoes intensive computations if high performance efficiency is desired. Channel estimation block finds a channel value that reflects the real channel. The channel can be modeled as a filter that has a frequency response. Channel estimation block requires a number of operations and should occupy many hardware resources. This block needs to be implemented efficiently with consideration of hardware area and performance.

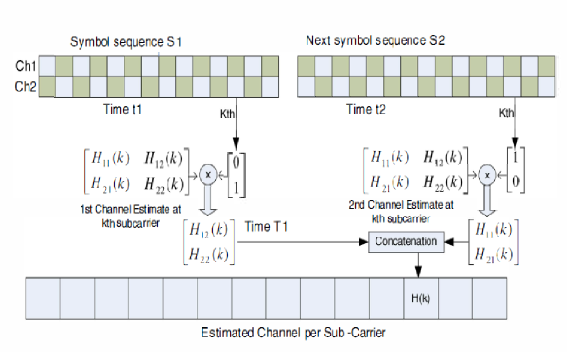

Symbol sequence S1
Next symbol sequence S2
Ch1
Ch2
Time t1
Time t2
Kth
Kth
$\begin{bmatrix} H_{11}(k) & H_{12}(k) \\ H_{21}(k) & H_{22}(k) \end{bmatrix}$
$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$
x
1st Channel Estimate at kth subcarrier
2nd Channel Estimate at kth subcarrier
$\begin{bmatrix} H_{12}(k) \\ H_{22}(k) \end{bmatrix}$
Time T1
Concatenation
$\begin{bmatrix} H_{11}(k) \\ H_{21}(k) \end{bmatrix}$
H(k)
Estimated Channel per Sub-Carrier

Figure 5.1 Channel estimation for each sub-carrier using long header

To de-multiplex the data transmitted from multiple transmitters, an accurate knowledge of the radio channel between different pairs of transmitter and receiver is required at receiver. The first requirement for reducing hardware resource is to select proper algorithm. The channel estimator uses header-based Maximum Likelihood (ML) method, in this special training symbols are transmitted before data packets are transmitted. Once training symbols are received and detected in receiver, the received data are modulated and preprocessed in frequency domain.

Hence, the channel frequency response is obtained in frequency domain, which is helpful to reduce computational complexity. Based on the selected algorithm, an effort is made to minimize hardware resource by using the pipelined architecture.

5.1 HEADER-BASED CHANNEL ESTIMATION WITH MAXIMUM LIKELI-HOOD ALGORITHM

Channel estimation is the task of estimating the frequency response of the radio channel on which the transmitted signal travels before reaching the receiver. The impulse response of a time varying radio channel is usually represented as a discrete time FIR filter. WLAN applications generally assume that channel is quasi-stationary. That is, the channel does not change during the data packet. Channel is estimated for every sub-carrier and each spatial path between transmitter and receiver antenna pairs, ($H_{ij}(k)$).

Header-based channel estimation with Maximum Likeli-hood algorithm is chosen to satisfy both the performance and less hardware resource utilization. Suppose X is a matrix of symbols being transmitted on a sub-carrier where the channel matrix is H. Then the received matrix of symbols R is given by

$$R=HX+W \qquad (5.1)$$

Where W is the white Gaussian noise vector at receiver. If we know X and R, the ML estimate is given by equation (7)

$$\hat{H}_{ML} = \arg\min_H \|R - HX\| \qquad (5.2)$$

If the matrix XX^+ is full-ranked matrix, the solution of the equation (2) is given by equation (3).

(5.3)

Select X such that where X is a unitary matrix in equation (4).

(5.4)

This can be modified to be performed on each subcarrier as,

$1 \leq k \leq N_c$ (5.5)

Thus, training symbols are selected on frequency domain as follows.

(5.6)

Hence, the channel estimation at receiver takes much less complicated operations after converting the received data to frequency domain. It makes hardware implementation simpler and hardware utilization smaller.

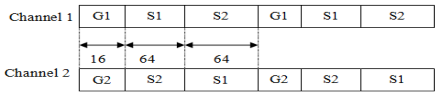


Figure 5.2 Structure of training symbols for channel estimation

Structure of channel estimation training symbols for 2 x 2 MIMO-OFDM is presented in Figure 5.2, where S1 and S2 indicate tone sequence 1 and tone sequence 2. Gl and G2 are the guard band of S1 and S2, respectively. Guard band is added for making cyclic extension of FFT symbols to avoid Inter Symbol Interference (ISI). S1, S2, G1, and G2 are repeated once again. Extra 3dB combining gain can be obtained by sending the same symbol two times.

The number of arithmetic operation per the whole channel estimation symbols in Figure 5.2 is

$$(64 \text{ multiplications} + 64 \text{ additions}) \times 4 \qquad (5.7)$$

where 4 is the number of channels in 2 x 2 MIMO-OFDM. Finally, 160 MOPS can be obtained from equation (5.7). This algorithm can be extended to support N x N MIMO-OFDM in the same way as 2 x 2 MIMO-OFDM. That is, when X is unitary matrix in N-dimensions, H_{ij} where i, j =0, 1, ... ,N can be obtained by using N sequences without decoding mixed channel.

5.2 ARCHITECTURE OF CHANNEL ESTIMATOR

General concept to implement channel estimation algorithm on hardware is to deploy each processing logic and memory onto hardware domain as shown in Figure 5.3. Each subcarrier delivers 1 bit signal as shown in equation (5.6), adder and multiplier required for equation (5.5) should become simpler. ROM Txl and Tx2 store sequence SI and S2, respectively. Switch block selects either ROM Txl or ROM Tx2 when either SI or S2 is received.

RAM is needed to store intermediate channel coefficients which are calculated over maximum 4 frames. However, the baseline architecture still consists of O (n^2) calculation blocks and memories in Figure 5.3. Instead, propose a pipelined architecture which uses only one block instead of O (n^2) redundant blocks in Figure 5.4. 4µsec is taken for a pipeline stage period between RAM1 and RAM2 because a transmission period per one group of modulated data is 4 µsec.

Consider a 2 x 2 MIMO-OFDM system, in this RAM la and RAM 1b store data received by Rxl and Rx2, respectively. Each data carried by each sub-carrier is read from RAM 1 in serial. When system clock frequency is 20 MHz, 80 cycles are taken to process one frame. Only one group of 64 calculations is required to cover two receivers per each frame because each Rx data includes 32 valid data. To reduce processing time taken by the calculation block, higher sampling frequency is used during channel estimation stage between RAM 1 and RAM 2 in Figure5.4. Dual-port RAM is deployed to reduce latency and support different data access.

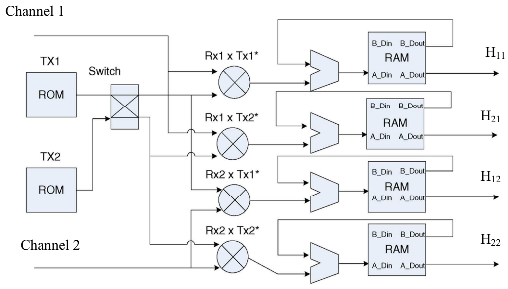


Figure 5.3 Architecture of baseline channel estimator

Input data stored in RAM la and RAM 1b are read every even clock and odd clock, respectively. Then, those data are processed through the calculation block, and stored into RAM 2 at 100 MHz while RAM 2 transfers estimated channels at 20 MHz 100 MHz is enough to

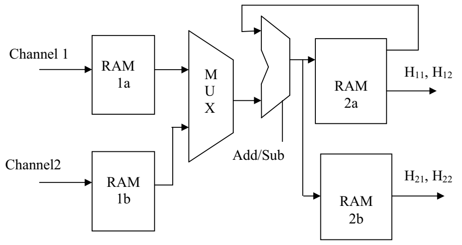


Figure 5.4 Architecture of new channel estimator

Adders can be used as subtractors. The calculation block is so simple that additional pipeline buffers are not needed. It means that critical path delay in channel estimator should be much shorter than system clock period. If two repeated sequences are used to improve performance, total 2 frames are taken to obtain channel estimation result. 3 x 3 MIMO-OFDM can be extended from 2 x 2 MIMO-OFDM in the same way. Only one group of 64 calculations is required to cover three receivers per each frame because each Rx data includes 21 or 22 valid data. Architecture in Figure 5.4 is also the same except the number of pipeline RAMs. Estimated channel coefficients can be obtained after processing data over 3 frames.

A header based maximum likeli-hood algorithm is chosen in consideration of hardware feature as well as communication theory for fast prototyping. Each channel is calculated in-dependently from other channels, so channel decoder block, based on the pipelined architecture which takes $0(n^2)$, could be removed. Since training symbols are unitary matrix, calculation logic consists of one adder and channel memories which corresponds to 0(1). Of course, ROM that stores training symbols is not necessary. All operations required for getting estimation result can be done by using the simple calculation logic in serial based on the pipelined architecture. New channel estimator architecture also supports more channels in MIMO-OFDM system.

CHAPTER 6
PROPOSED MIMO-OFDM SYSTEM

To design hardware-efficient MIMO-OFDM system is to reduce hardware resource used for Fast Fourier Transform (FFT) block. Multiple independent channels need multiple FFT operations. If N independent channels come at the same time, N parallel FFT blocks are needed. But, if each channel arrives at different time, resource sharing is possible between each channel. It means that the number of FFT blocks can be reduced. After all, as parallelism used for FFT processing is less, hardware area occupied by FFT block becomes smaller dramatically. If N FFT operations can be done by single FFT block without any parallelism, it can be regarded as a conceptually ideal system. a new pipelined architecture very close to the ideal single FFT system. It uses single radix-2 FFT block that is shared across modulations for MIMO-OFDM system. The baseline MIMO-OFDM system uses the same number of fast Fourier transform (FFT) blocks as antennas. The Proposed MIMO-OFDM system shows that at least 60 percent of the resources in the baseline MIMO-OFDM system can be saved using our proposed architecture, while achieving the same data rate.

6.1 PROPOSED ARCHITECTURE

The number of antennas that can be supported by the proposed pipelined architecture with one FFT. For example, 2-*Way* means that the proposed system achieves the same data rate as the baseline 2 × 2 MIMO-OFDM system with two FFT blocks.

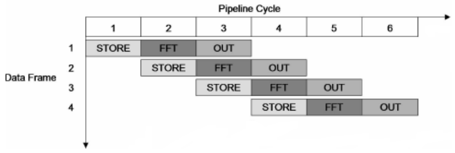


Figure 6.1 Pipeline flow

There are three stages in the proposed architecture: the STORE stage, FFT stage and OUT stage. These are shown in Figure 6.1. Since the FFT blocks of those designs also operate based on a pipelined architecture, pre-FFT (STORE) stages and post-FFT (OUT) stages are necessary.

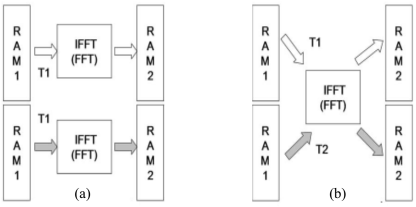


Figure 6.2 FFT operation of MIMO-OFDM

Figure 6.2-a illustrates the core of our design, and Figure 6.2-b illustrates the modulators of the baseline MIMO-OFDMs. The design in Figure 6.2-b uses approximately twice as much resources, compared with the design in Figure 6.2-a. T1 and T2 are the times at which the first and second data from RAM1 arrive at FFT block, respectively.

A general technique for using only one FFT block (as in Figure 6.2-a) is to increase the clock frequency for only the FFT stage. Therefore, the overall data flow given in Figure 6.3 needs to be divided into three stages. In Figure 6.3, the final output data from the STORE stage are stored into RAM1a and RAM1b.

The amount of resources (*RS*) used by the OFDM system is

$$RS = RD + RC, \qquad (6.1)$$

where *RC* and *RD* are the amount of resources consumed by logic blocks and data-path blocks respectively.

The amount of resources used by the baseline MIMO-OFDMs (R_m) is

$$R_m = R_C + N_A \times R_D$$
$$\cong N_A \times R_S \qquad (6.2)$$

where N_A is the number of antennas employed by the MIMO-OFDM system. Most of the data path blocks (R_D) used in OFDM such as the RAM and FFT block are simply extended to MIMO-OFDM.

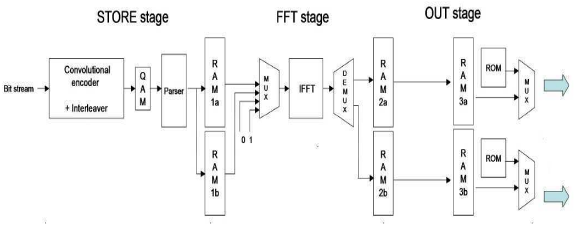


Figure 6.3 Transmitter data flow

.

The proposed MIMO-OFDM system occupies almost the same amount of resources as the OFDM system. Therefore, the ratio of resource used by the proposed architecture to the baseline MIMO-OFDM system ($n_{resource}$) can be expressed as

$$n_{resource} = R_{m2}/R_m \qquad (6.3)$$

The pipelined architecture supports high scalability as well as superior performance. The optimized MIMO-OFDM using pipelined architecture requires at least 60 percent fewer resources than baseline MIMO-OFDM systems, while maintaining the same data rate. As more channels are used, more resource can be saved by using proposed architecture.

CHAPTER 7
RESULTS & DISCUSSION

The simulation of this project has been done using ModelSim XE III 6.2g and Xilinx ISE 9.1i.

ModelSim is a simulation tool for programming {VLSI} {ASIC}s, {FPGA}s, {CPLD}s, and {SoC}s. Modelsim provides a comprehensive simulation and debug environment for complex ASIC and FPGA designs. Support is provided for multiple languages including Verilog, SystemVerilog, VHDL and SystemC. The following diagram shows the basic steps for simulating a design in ModelSim.

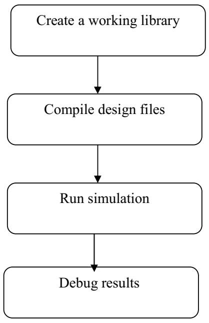


Figure 7.1 Basic steps of simulation in Modelsim

In ModelSim, all designs, be they VHDL, Verilog, or some combination thereof, are compiled into a library. We typically start a new simulation in ModelSim by creating a working library called "work". "Work" is the library name used by the compiler as the default destination for compiled design units. After creating the working library, we've to compile your design units into it. The ModelSim library format is compatible across all supported platforms. We can simulate our design on any platform without having to recompile design. With the design compiled, invoke the simulator on a top-level module (Verilog) or a configuration or entity/architecture pair (VHDL). Assuming the design loads successfully, the simulation time is set to zero, and enter a run command to begin simulation. If the results are not as expected , we can use ModelSim's robust debugging environment to track down the cause of the problem.

Xilinx was founded in 1984 by two semiconductor engineers, Ross Freeman and Bernard Vonderschmitt, who were both working for integrated circuit and solid-state device manufacturer Zilog Corp. The Virtex-II Pro, Virtex-4, Virtex-5, and Virtex-6 FPGA families are particularly focused on system-on-chip (SOC) designers because they include up to two embedded IBM PowerPC cores The ISE Design Suite is the central electronic design automation (EDA) product family sold by Xilinx. The ISE Design Suite features include design entry and synthesis supporting Verilog or VHDL, place-and-route (PAR), completed verification and debug using Chip Scope Pro tools, and creation of the bit files that are used to configure the chip .Xilinx is a synthesis tool which converts schematic/HDL design entry into functionally equivalent logic gates on Xilinx FPGA, with optimized speed & area. So, after specifying behavioral description for HDL, the designer merely has to select the library and specify optimization criteria; and Xilinx synthesis tool determines the net list to meet the specification; which is then converted into bit-file to be loaded onto FPGA PROM. Also, Xilinx tool generates post-process simulation model after every implementation step, which is used to functionally verify generated net list after processes, like map, place & route.

7.1 SIMULATION RESULTS

The following sections shows the simulation results of the MIMO-OFDM modules, Baseline MIMO-OFDM system, Proposed MIMO-OFDM system.

7.1.1 Simulation result of scrambler

Here in the waveform the clock is set and reset is set to 0. The input given is 1 and the scrambled output is 1101.

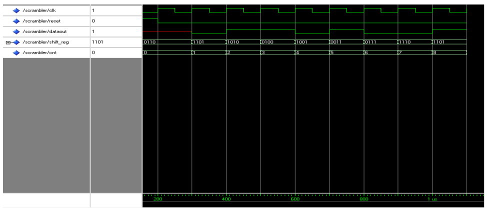


Figure 7.2 Simulation result of scrambler

7.1.2 Simulation result of descrambler

Here in the waveform the clock is set and reset is set to 0. The scrambled input is 1101 and the descrambled output is 1

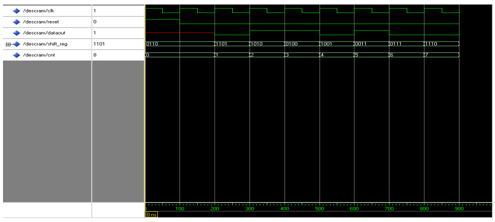


Figure 7.3 Simulation result of descrambler

7.1.3 Simulation result of interleaver

Here in the waveform the input data is 01101101 and the output is 01111001.

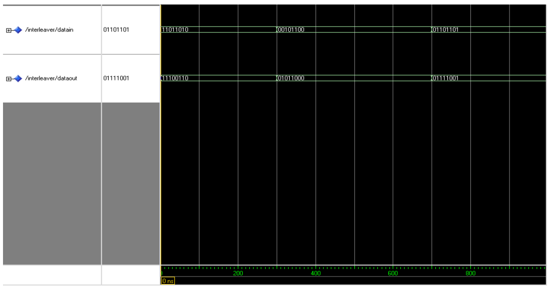
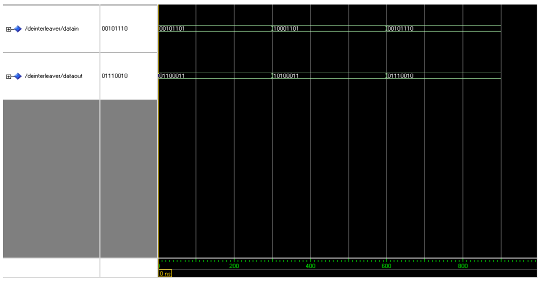


Figure 7.4 Simulation result of interleaver

7.1.4 Simulation result of deinterleaver

Here in the waveform the input data is 00101110 and the output is 01110010.



/deinterleaver/datain 00101110
/deinterleaver/dataout 01110010
00101101
10001101
00101110
01100011
10100011
01110010
200
400
600
800

Figure 7.5 Simulation result of deinterleaver

7.1.5 Simulation result of QPSK modulation

The Clock is set and the binary input is set to 11. The modulated inphase output is 1 and the Q-phase output is 1.

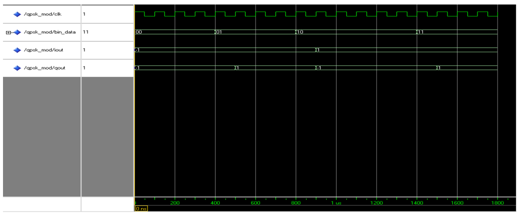


Figure 7.6 Simulation result of QPSK modulation

7.1.6 Simulation result of channel estimation

In the waveform the clock is set and binary input is 11. The channel matrix (H) is estimated h_{00}= 0, h_{01} = 0, h_{10} = -1, $h_{11\ =\ 1}$.

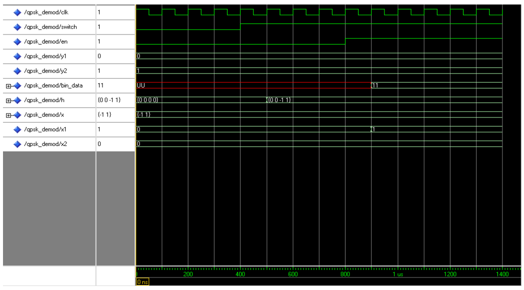
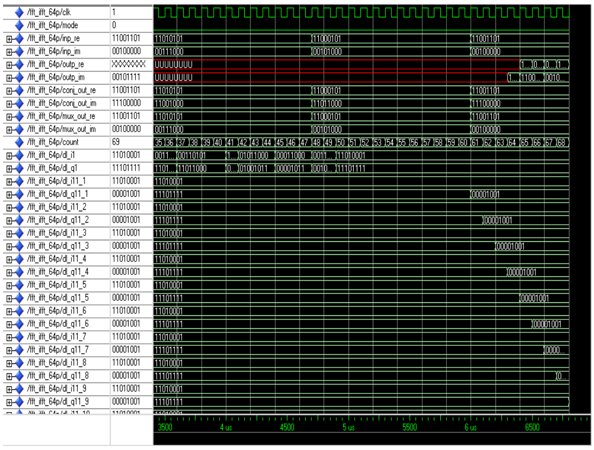


Figure 7.7 Simulation result of channel estimation

7.1.7 Simulation result of FFT/IFFT processor

In the waveform the clock is set and the mode is used to select the FFT/IFFT processor. If mode is 1 IFFT processor is selected and if the mode is 0 FFT processor is selected.

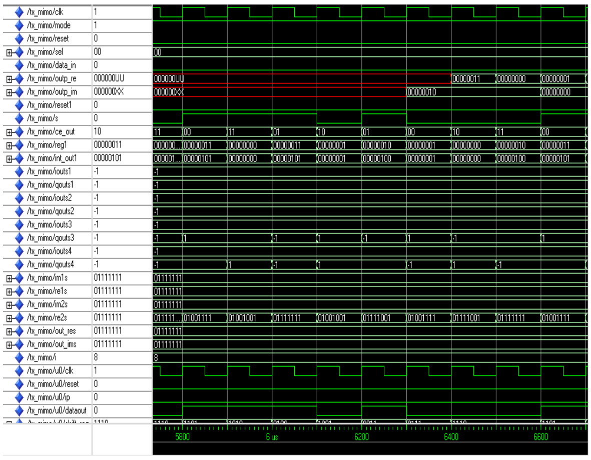


/fft_ifft_64p/clk 1
/fft_ifft_64p/mode 0
/fft_ifft_64p/inp_re 11001101
/fft_ifft_64p/inp_im 00100000
/fft_ifft_64p/outp_re
/fft_ifft_64p/outp_im 00101111
/fft_ifft_64p/conj_out_re 11001101
/fft_ifft_64p/conj_out_im 11100000
/fft_ifft_64p/mux_out_re 11001101
/fft_ifft_64p/mux_out_im 00100000
/fft_ifft_64p/count 69
3500
4 us
4500
5 us
5500
6 us
6500

Figure 7.8 Simulation result of FFT/IFFT processor

7.1.8 Simulation result of transmitter MIMO-OFDM system

In this waveform the clock is set and using the mode IFFT processor is chosen. The reset is set to 0. The output from the transmitter is given as input to the receiver.

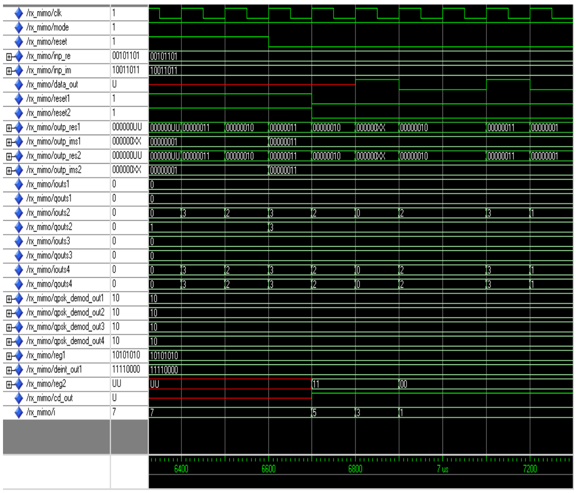


```
/tx_mimo/clk          1
/tx_mimo/mode         1
/tx_mimo/reset        0
/tx_mimo/sel          00
/tx_mimo/data_in      0
/tx_mimo/outp_re      000000UU
/tx_mimo/outp_im      000000XX
/tx_mimo/reset1       0
/tx_mimo/s            0
/tx_mimo/ce_out       10
/tx_mimo/reg1         00000011
/tx_mimo/int_out1     00000101
/tx_mimo/iouts1       -1
/tx_mimo/qouts1       -1
/tx_mimo/iouts2       -1
/tx_mimo/qouts2       -1
/tx_mimo/iouts3       -1
/tx_mimo/qouts3       -1
/tx_mimo/iouts4       -1
/tx_mimo/qouts4       -1
/tx_mimo/im1s         01111111
/tx_mimo/re1s         01111111
/tx_mimo/im2s         01111111
/tx_mimo/re2s         01111111
/tx_mimo/out_res      01111111
/tx_mimo/out_ims      01111111
/tx_mimo/i            8
/tx_mimo/u0/clk       1
/tx_mimo/u0/reset     0
/tx_mimo/u0/ip        0
/tx_mimo/u0/dataout   0
```

Figure 7.9 Simulation result of transmitter MIMO-OFDM

7.1.9 Simulation result of receiver MIMO-OFDM

In this waveform the clock is set and using the mode FFT processor is chosen. The output from the transmitter MIMO-OFDM is given as input to the receiver MIMO-OFDM.

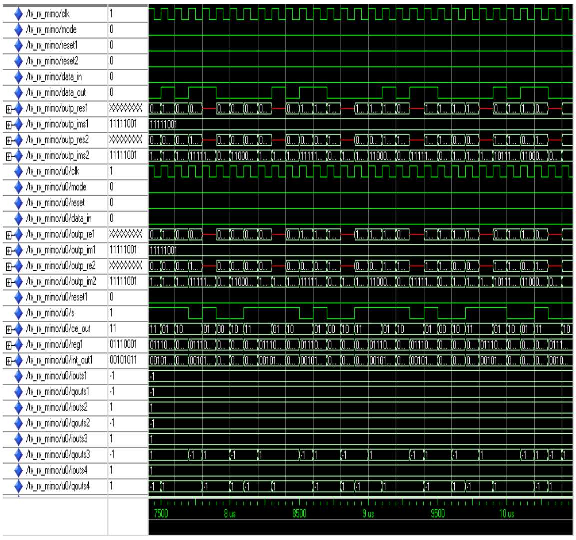


```
/rx_mimo/clk              1
/rx_mimo/mode             1
/rx_mimo/reset            1
/rx_mimo/inp_re           00101101
/rx_mimo/inp_im           10011011
/rx_mimo/data_out         U
/rx_mimo/reset1           1
/rx_mimo/reset2           1
/rx_mimo/outp_res1        000000UU
/rx_mimo/outp_ims1        000000XX
/rx_mimo/outp_res2        000000UU
/rx_mimo/outp_ims2        000000XX
/rx_mimo/iouts1           0
/rx_mimo/qouts1           0
/rx_mimo/iouts2           0
/rx_mimo/qouts2           0
/rx_mimo/iouts3           0
/rx_mimo/qouts3           0
/rx_mimo/iouts4           0
/rx_mimo/qouts4           0
/rx_mimo/qpsk_demod_out1  10
/rx_mimo/qpsk_demod_out2  10
/rx_mimo/qpsk_demod_out3  10
/rx_mimo/qpsk_demod_out4  10
/rx_mimo/reg1             10101010
/rx_mimo/deint_out1       11110000
/rx_mimo/reg2             UU
/rx_mimo/cd_out           U
/rx_mimo/i                7
```

Figure 7.10 Simulation result of receiver MIMO-OFDM

7.1.10 Simulation result of baseline MIMO-OFDM system

In this waveform the clock is set and reset is set to 0. Input data is transmitted and received in the system.

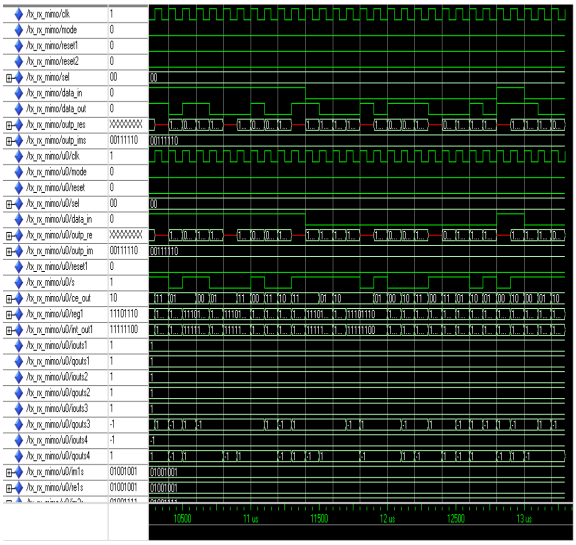


```
/tx_rx_mimo/clk            1
/tx_rx_mimo/mode           0
/tx_rx_mimo/reset1         0
/tx_rx_mimo/reset2         0
/tx_rx_mimo/data_in        0
/tx_rx_mimo/data_out       0
/tx_rx_mimo/outp_res1      XXXXXXXX
/tx_rx_mimo/outp_ims1      11111001
/tx_rx_mimo/outp_res2      XXXXXXXX
/tx_rx_mimo/outp_ims2      11111001
/tx_rx_mimo/u0/clk         1
/tx_rx_mimo/u0/mode        0
/tx_rx_mimo/u0/reset       0
/tx_rx_mimo/u0/data_in     0
/tx_rx_mimo/u0/outp_re1    XXXXXXXX
/tx_rx_mimo/u0/outp_im1    11111001
/tx_rx_mimo/u0/outp_re2    XXXXXXXX
/tx_rx_mimo/u0/outp_im2    11111001
/tx_rx_mimo/u0/reset1      0
/tx_rx_mimo/u0/s           1
/tx_rx_mimo/u0/ce_out      11
/tx_rx_mimo/u0/reg1        01110001
/tx_rx_mimo/u0/int_out1    00101011
/tx_rx_mimo/u0/iouts1      -1
/tx_rx_mimo/u0/qouts1      -1
/tx_rx_mimo/u0/iouts2      1
/tx_rx_mimo/u0/qouts2      -1
/tx_rx_mimo/u0/iouts3      1
/tx_rx_mimo/u0/qouts3      -1
/tx_rx_mimo/u0/iouts4      1
/tx_rx_mimo/u0/qouts4      1
```

Figure 7.11 Simulation result of baseline MIMO-OFDM system

7.1.11 Simulation result of proposed MIMO-OFDM system

In this waveform the clock is set to 1 and single FFT processor is shared across modulations for the system.



```
/tx_rx_mimo/clk            1
/tx_rx_mimo/mode           0
/tx_rx_mimo/reset1         0
/tx_rx_mimo/reset2         0
/tx_rx_mimo/sel            00
/tx_rx_mimo/data_in        0
/tx_rx_mimo/data_out       0
/tx_rx_mimo/outp_res       XXXXXXXX
/tx_rx_mimo/outp_ims       00111110
/tx_rx_mimo/u0/clk         1
/tx_rx_mimo/u0/mode        0
/tx_rx_mimo/u0/reset       0
/tx_rx_mimo/u0/sel         00
/tx_rx_mimo/u0/data_in     0
/tx_rx_mimo/u0/outp_re     XXXXXXXX
/tx_rx_mimo/u0/outp_im     00111110
/tx_rx_mimo/u0/reset1      0
/tx_rx_mimo/u0/s           1
/tx_rx_mimo/u0/ce_out      10
/tx_rx_mimo/u0/reg1        11101110
/tx_rx_mimo/u0/int_out1    11111100
/tx_rx_mimo/u0/iouts1      1
/tx_rx_mimo/u0/qouts1      1
/tx_rx_mimo/u0/iouts2      1
/tx_rx_mimo/u0/qouts2      1
/tx_rx_mimo/u0/iouts3      1
/tx_rx_mimo/u0/qouts3      -1
/tx_rx_mimo/u0/iouts4      -1
/tx_rx_mimo/u0/qouts4      1
/tx_rx_mimo/u0/im1s        01001001
/tx_rx_mimo/u0/re1s        01001001
```

Figure7.12 Simulation result of proposed MIMO-OFDM system

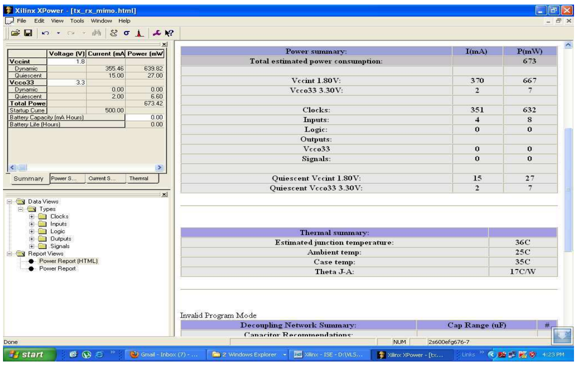
7.2 SYNTHESIS REPORT

This section describes the area and power reports of the MIMO-OFDM system.

7.2.1 Area report of baseline MIMO-OFDM system

```
Timing Summary:
---------------
Speed Grade: -7

Minimum period: 47.677ns (Maximum Frequency: 20.974MHz)
Minimum input arrival time before clock: 26.696ns
Maximum output required time after clock: 11.691ns
Maximum combinational path delay: 9.569ns

Design Summary
--------------
Number of errors:      0
Number of warnings:    3

Logic Utilization:
Total Number Slice Registers:   1,196 out of 13,824    8%
Number used as Flip Flops:                   1,194
Number used as Latches:                          2
Number of 4 input LUTs:         1,108 out of 13,824    8%

Logic Distribution:
Number of occupied Slices:                      1,121 out of  6,912   16%
Number of Slices containing only related logic:  1,121 out of  1,121  100%
Number of Slices containing unrelated logic:         0 out of  1,121    0%
*See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs:        1,219 out of 13,824    8%
Number used as logic:               1,108
Number used as a route-thru:          111
Number of bonded IOBs:          21 out of    510    4%
IOB Flip Flops:                       1
Number of GCLKs:                 1 out of      4   25%
Number of GCLKIOBs:              1 out of      4   25%

Total equivalent gate count for design:  18,118
Additional JTAG gate count for IOBs:  1,056
```

7.2.2 Power report of baseline MIMO-OFDM system



	Voltage (V)	Current (mA	Power (mW
Vccint	1.8		
Dynamic		355.46	639.82
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			673.42
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		673
Vccint 1.80V:	370	667
Vcco33 3.30V:	2	7
Clocks:	351	632
Inputs:	4	8
Logic:	0	0
Outputs:		
Vcco33	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

Thermal summary:	
Estimated junction temperature:	36C
Ambient temp:	25C
Case temp:	35C
Theta J-A:	17C/W

Invalid Program Mode

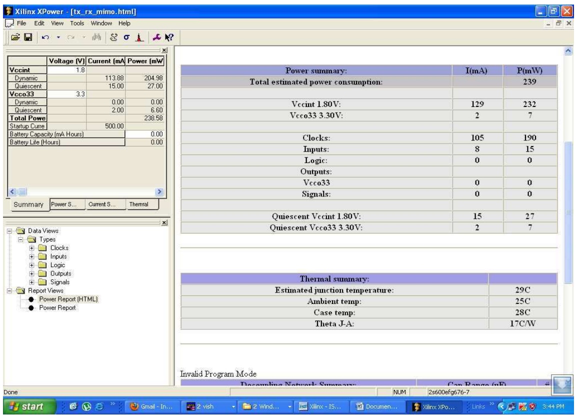
Decoupling Network Summary:	Cap Range (uF)	#
Capacitor Recommendations:		

Figure 7.13 Power report of baseline MIMO-OFDM system

7.2.3 Area report of proposed MIMO-OFDM system

```
Timing Summary:
 ---------------
Speed Grade: -7
Minimum period: 10.963ns (Maximum Frequency: 91.216MHz)
Minimum input arrival time before clock: 5.622ns
Maximum output required time after clock: 6.140ns
Maximum combinational path delay: No path found

Design Summary
--------------
Number of errors:      0
Number of warnings:    3

Logic Utilization:
Total Number Slice Registers:       74 out of 13,824    1%
 Number used as Flip Flops:                    72
 Number used as Latches:                        2
 Number of 4 input LUTs:            96 out of 13,824    1%

Logic Distribution:
 Number of occupied Slices:                        58 out of  6,912     1%
 Number of Slices containing only related logic:     58 out of     58  100%
 Number of Slices containing unrelated logic:         0 out of     58    0%
 *See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs:          101 out of 13,824    1%
 Number used as logic:                        96
 Number used as a route-thru:                  5
 Number of bonded IOBs:           2 out of    510    1%
 IOB Flip Flops:                   1
Number of GCLKs:                 1 out of      4   25%
Number of GCLKIOBs:              1 out of      4   25%

Total equivalent gate count for design:  1,625
Additional JTAG gate count for IOBs:  144
```

7.2.4 Power report of proposed MIMO-OFDM system



	Voltage (V)	Current (mA	Power (mW
Vccint	1.8		
Dynamic		113.88	204.98
Quiescent		15.00	27.00
Vcco33	3.3		
Dynamic		0.00	0.00
Quiescent		2.00	6.60
Total Powe			238.58
Startup Curre		500.00	
Battery Capacity (mA Hours)			0.00
Battery Life (Hours)			0.00

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		239
Vccint 1.80V:	129	232
Vcco33 3.30V:	2	7
Clocks:	105	190
Inputs:	8	15
Logic:	0	0
Outputs:		
Vcco33	0	0
Signals:	0	0
Quiescent Vccint 1.80V:	15	27
Quiescent Vcco33 3.30V:	2	7

Thermal summary:	
Estimated junction temperature:	29C
Ambient temp:	25C
Case temp:	28C
Theta J-A:	17C/W

Invalid Program Mode

Figure 7.14 Power report of proposed MIMO-OFDM system

The following table shows the comparison of logic utilization and logic distribution of baseline MIMO-OFDM and proposed MIMO-OFDM

Design summary		Baseline MIMO-OFDM system	Proposed MIMO-OFDM system
Logic Utilization	Number used as Flip flops	1,194	72
	Number of 4 input LUTs	1,219 out of 13,824	96 out of 13,824
Logic distribution	Total Number 4 input LUTs	2,178 out of 13,824	101 out of 13,824
	Total equivalent gate count	18,118	1623

Table 7.2.1 Logic utilization and distribution of MIMO-OFDM system

7.3 COMPARISON OF THE BASELINE AND THE PROPOSED MIMO-OFDM SYSTEM

Parameters	Baseline MIMO-OFDM	Proposed MIMO-OFDM
Total equivalent gate count for design	18,118	1623
Total Estimated Power Consumption	673mw	116mw

Table 7.3.1 Baseline Vs Proposed MIMO-OFDM system

CHAPTER 8
CONCLUSION

MIMO-OFDM Transceiver is designed using good channel estimation method, efficient FFT/IFFT processor, encoding and decoding of STBC and other better coding techniques. A modified MIMO-OFDM system is proposed using single Fast Fourier Transform that is shared across modulations for the system. Thus more resources are saved by using our proposed architecture. The design is simulated successfully in Modelsim. The design is synthesized and implemented on SPARTAN 2E FPGA board.

FUTURE SCOPE

The future scope of this project is to add the noise in the system and the estimation of channel using different channel models is to be carried. Channel estimation and compensation for different channel models for delays also to be implemented.

REFERENCES

[1] Jian Sun, Nan Cen, Dongfeng Yuan, “Implementation of a 2x2 MIMO-OFDM system on DSP/FPGA Platform ,” 3rd International Conference on Communications and Mobile Computing,2011.

[2] Jeoong Sung Park and Tokunbo Ogunfunmi, ”FPGA implementation of Channel Estimation for MIMO-OFDM,” 8th International Conference on Signal Processing,2011.

[3] Chu Yu, Mao-Hsu Yen, Pao-Ann Hsiung, and Sao-Jie Chen, “A Low-Power 64-point Pipeline FFT/IFFT Processor for OFDM Applications,” IEEE Transactions on Consumer Electronics, Vol. 57, No. 1, February 2011.

[4] Young-Ho Seo, Dong-Wook Kim,”A New VLSI Architecture of Parallel Multiplier–Accumulator Based on Radix-2 Modified Booth Algorithm,” IEEE TRANSACTIONS on VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 18, NO. 2, February 2010.

[5] J. S. Park, T. Ogunfunmi, "FPGA implementation of the MIMO OFDM Physical Layer using single FFT multiplexing," in Proceeding of IEEE International Symposium on Circuits and Systems (ISCAS),2010.

[6] K.Praveen Bagadi and S. Das, "MIMO-OFDM Channel Estimation Using Pilot Carries," International Journal of Computer Applications, vol. 2, pp. 81-88, May 2010.

[7] S.Tiiro, J. Y lioinas, M. MyllyHi, and M. Juntti, "Implementation of the least squares channel estimation algorithm for MIMO-OFDM systems," ITG Workshop on Smart Antennas, pp. 155-161, Feb. 2009.

[8] S. Yoshizawa and Y.Miyanaga, "VLSI Implementation of a 4x4 MIMO-OFDM transceiver with an 80-MHz channel bandwidth," Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on, vol., no., pp. 1743-1746,24-27 May 2009.

[9] S. Haene, D. Perels, and A. Burg "A Real-Time 4-Stream MIMO-OFDM Transceiver: System Design, FPGA Implementation, and Characterization," IEEE Journal on Selected Areas in Communications, vo1.26, no.6, pp.877-889, August 2008.

[10] En Zhou, Xiaolin Hou, Jianping Chen, “FPGA Implementation and Experimental Performances of a Novel Timing Synchronization Method in MIMO-OFDM Test bed,” Proc. of 14th Asia-Pacific Conference on Communications, APCC 2008. Oct. 2008.

[11] Nirav Shah, "MIMO-OFDM: Theory to implementation,", MS-Thesis, University of Southern California, 1435118,92 pages, Dec. 2006.

[12] Chin-Teng Lin, Yuan-Chu Yu, and Lan-Da Van, “A low-power 64-point FFT-IFFT design for IEEE 802.11a WLAN application,” in Proc. IEEE Int. Symposium. Circuits Syst. (ISCAS), May 2006, pp. 4523-4526.

[13] Changchuan Yin, Jingyu Li, Xiaolin Hou and Guangxin Yue, “Pilot Aided LS Channel Estimation in MIMO-OFDM Systems,”8th International Conference on Signal Processing,vol.3, pp.(16-20), 2006.

[14] George Tsoulos, “MIMO System Technology For Wireless Communications”, Revised Edition, CRC Publisher, 2006.

[15] Ian Griffiths, “FPGA Implementation Of MIMO Wireless Communications System,” University Of New Castle,Australia,1st November, 2005.

[16] G. L. Stuber, J. R. Barry, S. W. Mclaughlin, Y. Li, M. A. Ingram, and T. G. Pratt, “Broadband MIMO-OFDM wireless communications,” Proceedings of IEEE, vol. 92, no. 2, pp. 271-294, Feb. 2004.

[17] K. J. Cho, K. C. Lee, J. G. Chung, and K. K. Parhi, “Design of low-error width modified Booth multiplier,” IEEE Trans. Very Large Scale Integration Systems, vol. 12, no. 5, pp. 522–531, May 2004.

[18] Koushik Maharatna, Eckhard Grass, and Ulrich Jagdhold, “A 64-Point fourier transform chip for high-speed wireless LAN application using OFDM,” IEEE Journal of Solid-State Circuits, vol. 39, no. 3, pp. 484- 493, Mar. 2004.

[19] D. Gesbert, M. Shafi, Da shan Shiu, P. Smith, and A. Naguib, "From theory to practice: An overview of MIMO space-time coded wireless systems," IEEE Journal Selected Areas in Communications, vol. 21, no. 3: pp. 281-302, Apr. 2003.

[20] Y. Jung, H. Yoon, and J. Kim, “New efficient FFT algorithm and pipeline implementation results for OFDM/DMT applications*,”* IEEE Transactions on Consumer Electronics, vol. 49, no. 1, pp. 14-20, Feb. 2003.

[21] Sampath, H.,Talwar, S.,Tellado, J., “A fourth-generation MIMO-OFDM broadband wirelesssystem: design, performance, and field tria l results,” Communications Magazine, IEEE , Volume: 40 , Issue: 9, 2002.

[22] S. He and M. Torkelson, “Designing Pipeline FFT Processor for OFDM(de)Modulation,” in Proc. URSI Int. Symp. Signals, Systems, and Electronics, vol. 29, Oct.1998, pp. 257-262.

[23] G. J. Foschini, G. D. Golden, P. W. Wolnianshy and R. A. Valenzuela, “ Simplified processing for high spectral efficiency wireless communication employing multi-element arrays,” IEEE Journal of Selected Areas in Communications, vol. 17, no. 11, pp. 1841-1852, Nov. 1999.

[24] S. M. Alamouti, “A simple transmit diversity technique for wireless communications,” IEEE Journal on Selected Areas in Communication, vol. 16, pp. 1451-1458, Oct. 1998.

[25] H.L. Groginsky and G.A. Works, “A pipeline fast Fourier transform,” IEEE Transactions on Computers, vol. C-19, no. 11, pp. 1015-1019, Nov. 1970.