CDMA System: Simulation and Implementation

by

Anup Dhital (059BEX403)

Nilesh Man Shakya (059BEX419)

Sarvesh Agarwal (059BEX437)

Saurav Ratna Tuladhar (059BEX438)

Final year project report submitted in partial fulfillment of the requirements for the degree of Bachelors of Engineering in Electronics and Communication Engineering

Supervisors: **Asst. Prof. Dibakar Raj Panta**
Asst. Prof. Sanjeev Ghimire

Institute of Engineering
Department of Electronics and Computer Engineering
Nepal
December 2006

ACKNOWLEDGEMENT

This project was successfully accomplished within the stipulated duration. Our success would not have been possible without help from our supervisors, teachers, family and friends.

We would like to express our gratitude towards our supervisors **Asst. Prof. Dibakar Raj Panta** whose excellent suggestions with positive approach and **Asst. Prof. Sanjeev Ghimire**, whose guidance throughout the semester has been of great moral support for us.

We thank **Asst. Prof. Mr. Rajesh Sharma** for his ideas and genuine suggestions for the project. We are also greatly indebted to **Asst. Prof. Sanjeeb P. Pandey** for supporting us and allowing us to use the RF and Microwave lab for system testing.

We are grateful towards **Asst. Prof. Rajendra Lal Rajbhandari**, for supporting us with necessary logistics.

Finally we express our sincere appreciation for the unflinching support from our family and friends.

Thank you.

Anup Dhital (059BEX403)

Nilesh Man Shakya (059BEX419)

Sarvesh Agarwal (059BEX437)

Saurav Ratna Tuladhar (059BEX438)

December 2006

ABSTRACT

CDMA is a multiplexing technique on the basis of set of orthogonal codes. CDMA is complex compared to traditional multiplexing technique yet CDMA has become the preferred choice of multiple access air interfaces in mobile telephone. This project aims at implementing code division multiplexing in hardware level, during the initial phase. In the final phase a simulation based study of CDMA system was performed. This report presents the detailed description of work carried out within project duration and the results obtained. Implementation of Code Division Multiplexing in FGPA has been successfully accomplished and the report explains in depth, the system design, implementation in VHDL, synthesis onto an FPGA and verification process in detail. Similarly, the simulation of CDMA system in MATLAB has also been completed with some limitations. The report describes the simulation model used and the analysis of results obtained.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

CDMA : Code Division Multiple Access

TDMA : Time Division Multiple Access

FDMA : Frequency Division Multiple Access

SSMA : Spread Sprectrum Multiple Access

CDM :Code Division Multiplexing

IS-95: Interim Standard – 95

GSM: Global System for Mobile Communication

WCDMA : Wideband CDMA

BER: Bit Error Rate

CHAPTER 1
INTRODUCTION

1.1 Background

A robust multiplexing technique is imperative to serve multiple users over a common air interface channel in mobile communication. TDMA and FDMA are two traditional multiple access schemes and are used in GSM. A new multiple access technique based on identifying a unique user based on a set of orthogonal codes is called Code Division Multiple Access (CDMA).

CDMA uses code division multiplexing (CDM) technique which is based on mathematical concept of orthogonal codes. CDM allows all users to share a common time and frequency domain and yet be uniquely identified on the basis unique codes. CDM technique proves to be advantageous over TDMA and FDMA techniques is becoming the preferred choice of multiple choice air interfaces in future generations of mobile technology. The IS-95 cellular mobile system from Qualcomm was the first system to use CDMA as the multiple access technique in mobile telephony.

CDM is complex technique compared to TDMA and FDMA. Further, intuitive understanding of CDM is difficult due to mathematical theory involved. However this technique is proved to be more spectrally efficient, increased capacity and secure compared to prevalent technologies. The commercial popularity of CDM technology in mobile telephony has made CDMA an important technology.

This project aims at understanding this very multiplexing technology. In the initial phase operation of CDM was implemented at hardware level. In the final phase a theoretical simulation study of CDMA based communication system under different operational conditions was performed.

1.2 Objectives

This project aims at understanding CDMA technology and has two objectives.

- To implement Code Division Multiplexing in FPGA using VHDL.
- To simulate CDMA Communication System in MATLAB and analyze its performance under different noise conditions.

1.3 Methodology

The implementation of CDM in FPGA was taken up in following stages

- Design of logic circuit for CDM
    - This involved designing of PN sequence generators, spreading and despreading logic and detection logic.
- Implementation of the design in FPGA
    - This involved implementing the designed digital circuit in a XSA-50 prototyping board. VHDL was used as the description language for this purpose.
- Verify multiple access operation

The simulation of CDMA system involved development of theoretical model of CDMA system in MATLAB. The simulation was run with different input conditions and the results obtained were analyzed.

1.4 Structure of the Report

This report has been divided into six chapters. Chapter1 introduces the project background, objectives and methodology used.Chapter2 provides the related literature review.Chapter3 discusses in detail, the implementation of CDM system in FPGA. Chapter4 deals with the simulation of CDMA system in MATLAB. Chapter5 provides the results obtained from FPGA implementation and MATLAB simulation. Finally Chapter6 presents the concluding remarks and recommendation for further improvement

CHAPTER 2
LITERATURE REVIEW

2.1 Code Division Multiple Access

CDMA (Code Division Multiple Access) is a multiplexing scheme that discriminates multiple users by the unique codes assigned to them. Unlike Frequency Division Multiple Access (FDMA) and Time Division Multiple Access (TDMA), it does not use frequency and time as discriminating factor. All the users occupy a common RF spectrum simultaneously. This concept can be visualized with time, frequency and code in three axes as shown in figure 1.CDMA provides a new dimension in multiple access schemes.

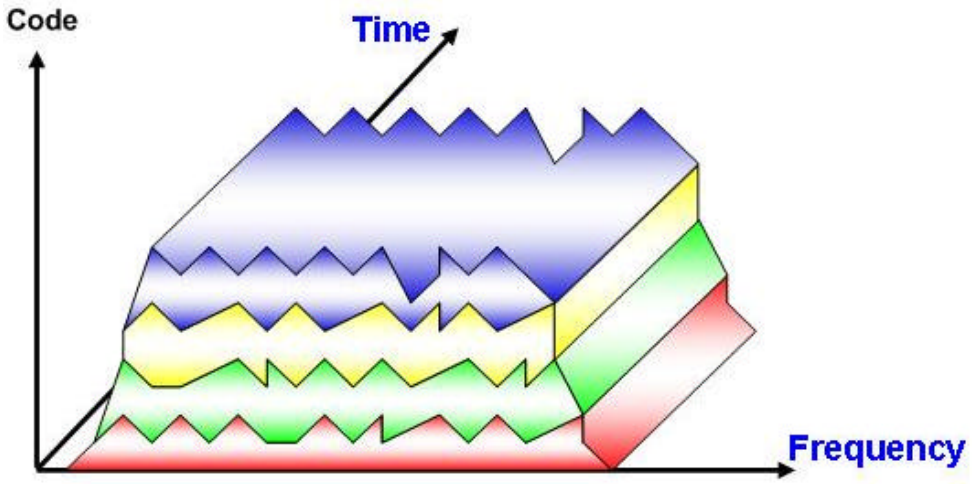

Figure 1. Code Division Multiplexing Scheme

CDMA is based on the core concept of orthogonality and spread spectrum. It is actually a kind of spread spectrum multiple access technique (SSMA). SSMA uses pseudo-noise (PN) sequence to convert a narrowband signal to a wideband noise like signal before transmission. The resulting signal has much greater bandwidth than required by the original message signal. Two commonly used spreading sequences in CDMA are

- Pseudorandom Codes/Pseudonoise(PN) Codes
- Walsh Codes.

PN sequence is preferred over Walsh codes for its property of randomness and deterministic generation.There are two types of SSMA techniques

Frequency Hopped Multiple Access (FHMA)

Direct Spread Multiple Access (DSMA).

2.1.1 Direct Sequence CDMA (DS-CDMA)

The direct spread multiple access technique is more commonly known as CDMA. In CDMA the narrowband message signal is multiplied by a very large bandwidth signal called the spreading signal. This spreading of a narrow band signal using a wideband signal is shown in figure 2. The spreading signal is a PN code sequence that has a chip rate[1] which is orders of magnitude greater than the data rate of the message. Each user in CDMA has its own pseudorandom code. The receiver performs a time correlation operation to detect only the specific desired codeword. All other code words appear as noise due to the decorrelation. Besides PN sequences, Walsh codes can also be used to uniquely identify each user and spread narrow band signal.

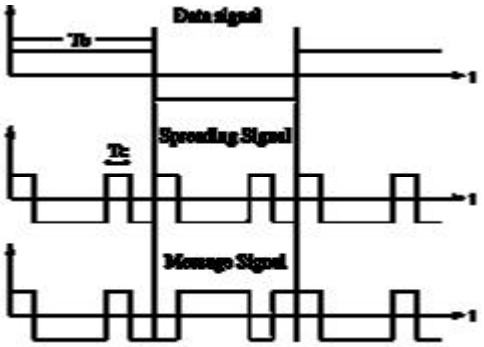

Figure 2. Direct sequence spreading

In practical implementation of CDMA the multiplication between the data and the spreading code is achieved by exclusive-ORing the data with the spreading code. At the receiver the data is recovered by again exclusive-ORing the multiplexed data with the spreading code of the particular user. Thus, exclusive-ORing the information bits with the spreading sequence twice (once prior to transmission and once after reception) reproduces the original information bits. The process of spreading and despreading is shown in figure 3.

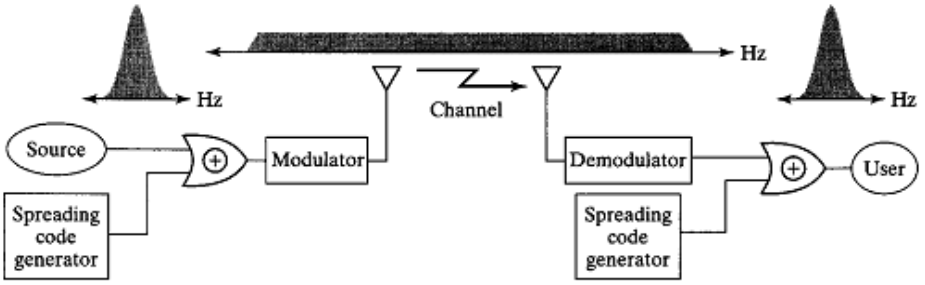

Figure 3. Spread Spectrum System

[1] Each bit of a spreading sequence is called a 'chip'.

2.1.2 Standard Communication System and CDMA System

In a standard communication system shown in figure4, a bit determines whether the transmit filter or its inverse is emitted every T seconds. The waveform then passes through the channel, and is corrupted by noise. The resulting signal is passes through a matched filter, and sampled every T seconds. T is known as the bit time and R = 1/T is known as the bit rate.

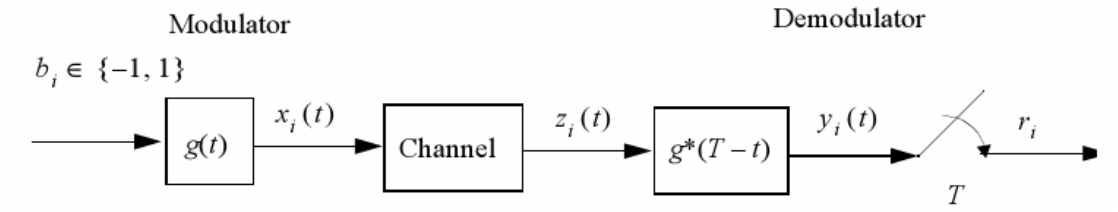

Figure 4. Standard Communication System

However, in a CDMA system shown in figure5, a bit still enters the system every T seconds, but it is now multiplied by a higher rate random sequence with period T_c. The result is sent through the channel, sampled and the sampled signal is multiplied by the corresponding random bit. The transmitted data sequence is then recovered by a correlator detector.

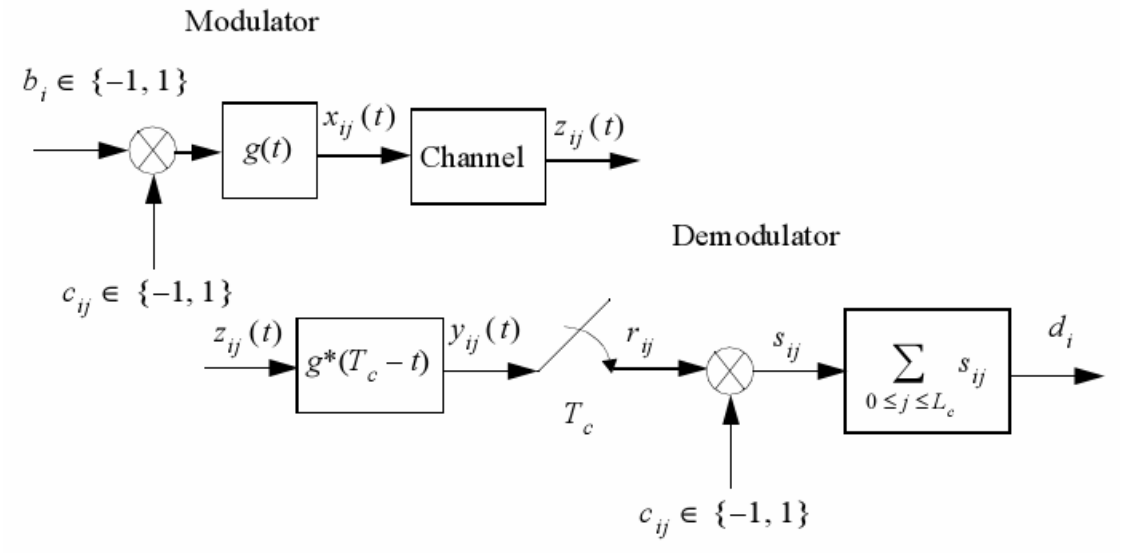

Figure 5 CDMA System

Each short symbol of the random sequence is called a chip, the time Tc is then known as a chip time, while $R_c = 1/T_c$ is known as the chip rate or spreading rate.

2.1.3 Multiple Access using CDMA

Spreading can be used as a multiplexing technique by developing a series of orthogonal spreading codes. These codes may be Walsh codes or PN codes. However PN codes are preferred over Walsh codes because of its one additional feature: If any two different orthogonal spreading codes are exclusive-ORed bit by bit, the resulting series of bits will itself be a PN code. Thus, if a signal is spread using one code and then despread using another orthogonal code, the result will just look like a PN code and will have a power spectral density similar to wideband white Gaussian noise.

Consider the system shown in Figure 6, with each of the three source pairs employing mutually orthogonal spreading codes to transmit information over a common channel. Source A uses one spreading code (Let's call it Spreading Code A) and transmits the spread spectrum signal shown in the figure. This signal is wideband and, from the viewpoint of the channel and any observer who does not know the code, the signal looks exactly like additive white Gaussian noise. Source B uses a second, orthogonal spreading code (Spreading Code B) and transmits another spread spectrum signal. Source C uses a third, orthogonal spreading code (Spreading Code C) to transmit a message across the channel. The sum of the signals, which has a power spectral density similar to wideband noise, carries across the channel and arrives at the receivers being employed by the three users. Some noise from the channel itself may also be added.

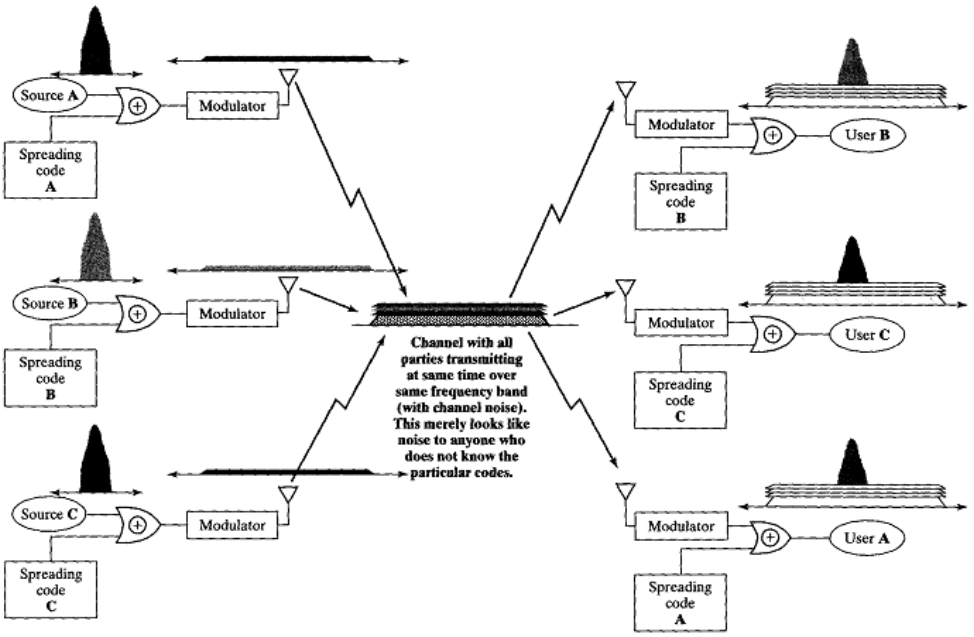

Figure 6. Multiple Access using CDMA

Now consider the receivers. The receiver associated with User B applies Spreading Code B to the total received signal. This despreads the portion of the signal transmitted from Source B, but leaves all other portions of the received signal with a wideband noise-like power spectral density. Using a narrow bandpass filter, User B may now extract the portion of the signal associated with Source B, with the channel noise and the interference from Users A and C significantly reduced. Similarly, User A and User C may use their respective spreading codes to extract their intended message.

The spreading operation minimizes the effects of narrowband interference and channel noise while minimally affecting the desired receiver signal. The despreading concentrates the desired signal's power into a bandwidth that is $1/G_p$ times the bandwidth of the spread signal, but despreading does not affect the power spectral density of the interfering signals from the other transmitters or the channel noise. Thus, after despreading and filtering, the SNR increases by a factor of G_p which is called the Processing Gain. This parameter gives an approximate measure of the interference rejectio n capability of the spread spectrum system. The processing gain is defined as

$$\mathbf{G_p = B_{ss}/B}$$

Where B is the original message bandwidth and B_{ss} is the spread spectrum signal bandwidth.

2.1.4 Frequency Hopped Multiple Access (FHMA)

In FHMA the carrier frequencies of the individual users are varied in a pseudorandom fashion within a wideband channel. At the receiver a locally generated PN code is used to synchronize the receiver's instantaneous frequency with that of the transmitter. This scheme of spread spectrum provides a high level of security and is generally used in Bluetooth and Home RF wireless technologies. The frequency hopping CDMA for four users is shown in figure7.

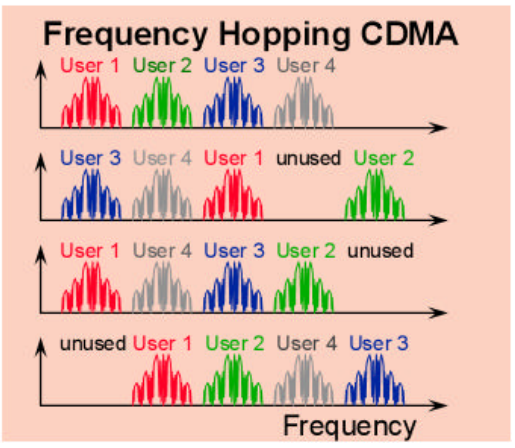

Figure 7. FHMA scheme for four users

2.2 Generation of Spreading Sequences (PN sequences)

An ideal spreading sequence would be random sequences of binary ones and zeros. However, because it is required that transmitter and receiver must have a copy of the random bits stream, a predictable way is needed to generate the same bit stream at transmitter and receiver and yet retain the desirable properties of a random bit stream. The requirement is met by a PN generator. A PN generator will produce a periodic sequence that eventually repeats but that appears to be random.

PN sequences are generated by an algorithm using some initial value called **seed**. The algorithm is deterministic and therefore produces sequence of numbers that are not statistically random. However, if the algorithm is good, the resulting sequences will pass many reasonable tests of randomness. Such numbers are often referred to as *pseudorandom numbers or pseudonoise sequence* (PN sequences). An important point is that unless the algorithm and the seed are known, it is impractical to predict the sequence.

Two important properties of PNs are randomness and unpredictability. The following two criteria are used to validate that a sequence of numbers is random

Uniform distribution: The distribution of numbers in the sequence should be uniform; that is, the frequency of occurrence of each of the numbers should be approximately the same. For a stream of binary digits, only two numbers ('1' and '0') are used, following two properties are desired

Balance property: In a long sequence the fraction of binary ones should approach ½.

Independence: No one value in the sequence can be inferred from the others.

In spread spectrum application, the correlation property should be such that *if a period of the sequence is compared term by term with any cycle shift of itself, the number of terms that are the same differs from those that are different by at most one.*

2.2.1 Linear Feedback Shift Register (LFSR) implementation

The PN generator for spread spectrum is usually implemented as a circuit consisting of exclusive-OR gates and a shift register, called a **Liner Feedback Shift Register (LFSR)**. The LFSR is a string of 1-bit storage devices. Each device has an output line, which indicates the value currently stored, and an input line. At discrete time instants, known as clock times, the value in the storage device is replaced by the value indicated by its input line. The entire LFSR is clocked simultaneously, causing a 1-bit shift along the entire register.

The circuit implementation of LFSR contains

- n registers (flip flops)
- 1 to (n-1) XOR gates

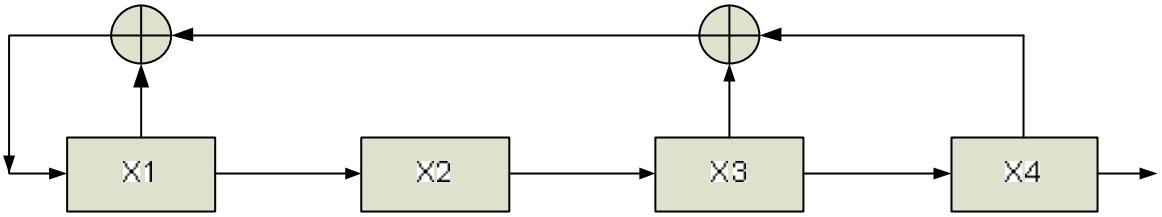

Figure 8. Liner Feed Back Shift Register (LFSR)

The linear shift-register sequence is specified by a primitive polynomial h(x) of degree *m*, where:

$$h(x) = h_m x^m + h_{m-1} x^{m-1} + h_{m-2} x^{m-2} + \ldots + h_1 x + h_0$$

The LFSR in figure 8 is specified by the generator polynomial $h(x) = x^4 + x^3 + x + 1$

The output of an LFSR is periodic with maximum period of $N = 2^n - 1$. The all zero sequence occurs only if the initial contents of the LFSR are all zero. A feedback configuration can always be found that gives a period of N; the resulting sequences are called *maximal-length or m-sequences.* The m-sequences are used in CDMA. The optimal tap points in LSFR for generating m-sequences are listed in Appendix B.

For many communication applications, the 0, 1 sequence is changed to a ±1 sequence by representing a binary 1 with +1 and binary 0 with -1. A related function, also important in the spread spectrum context, is the cross correlation function. The cross-correlation between two sources, A and B, is defined as

$$R_{A,B}(t) = \frac{1}{N} \sum_{k=1}^{N} A_k B_{k-t}$$

The cross correlation between two different m-sequences is low, and this property is useful for CDMA applications because it enables a receiver to discriminate among spread spectrum signals generated by different m-sequences.

2.2.2 Orthogonality of PN sequence [6]

The correlation of two random variables x(t) and y(t), is a time-shift comparison which expresses the degree of similarity or the degree of likeness between the two variables. The Auto-Correlation function R, provides the degree of similarity between a random variable

x(t) and a time-shifted version of itself. Likewise, the cross-correlation function provides the degree of similarity between a random variable x(t) and time-shifted version of another random variable y(t). To get the average value of the auto-correlation or cross-correlation, normalization by the sequence length L is required.

Consider Ci(t) and the time-shifted version of itself, say Ci(t-1)

Ci(t) = 1 0 0 1 1 1 0

Ci(t-1) = 0 0 1 1 1 0 1

When corresponding bits from the two sequences have the same parity (or match each other), we call the match an agreement "A". Likewise, when corresponding bits from the two sequences do not have the same Parity (do not match each other), we call the mismatch a disagreement "D" (see figure 4). By counting all the agreements and all the disagreements over the full length L of the sequence, a measure of correlation can be estimated as follows. Correlation = Total number of "A" - Total number of "D"

Now, consider the reference PN code Ci(t) and its time-shifted versions as shown below. Now let us compute the correlation of Ci(t) and Ci(t-t1), for all suitable values of t1 (here from 0 to 7).

Time Shifts	**Shifted Sequence**	**Correlation**
Reference	1 0 0 1 1 1 0	
t0	1 0 0 1 1 1 0	+7
t1	0 0 1 1 1 0 1	-1
t2	0 1 1 1 0 1 0	-1
t3	1 1 1 0 1 0 0	-1
t4	1 1 0 1 0 0 1	-1
t5	1 0 1 0 0 1 1	-1
t6	0 1 0 0 1 1 1	-1
t7	1 0 0 1 1 1 0	+7

In general, it can be shown that the full-length auto-correlation function (R) of PN codes or PN sequences is characterized by a large positive number equal to the length of the PN sequence ($R=2^n$ -1) when time shift=0, and -1 for all time-shifts equal or greater than the duration of one chip. So when normalized by the length, the auto-correlation function is equal to 1 at time-shift zero and is very small (-1/L) for all values of time shifts equal or greater than one chip.

Two PN sequences Ci(t) and Cj(t) are said to be orthogonal if and only if their respective normalized correlation function is equal to 1 at a time-shift of zero and their cross-correlation function is equal to zero for all time-shift values. As shown above, averaged over the code length, the cross-correlation function of PN sequences is not zero. As a result, PN sequences are not perfectly orthogonal.

2.3 Walsh Hadamard Codes

Walsh Hadamard codes or simply Walsh codes are the most common orthogonal codes used in CDMA application. A set of Walsh codes of length n consists of n x n Walsh matrix. The matrix is defined by a recursive relation as follows

$$\mathbf{W_1 = (0)}$$

$$W_{2n} = \begin{pmatrix} W_n & W_n \\ W_n & \overline{W_n} \end{pmatrix}$$

where n is a power of two and the $\overline{W_n}$ denotes the logical complement . Walsh matrix of 2nd and 4th order are shown below

$$W_2 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad W_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

The Walsh matrix has the property that every row is perfectly orthogonal to every other row and to the logical NOT of every other row. To compute the cross correlation, we replace 1 with +1 and 0 with -1. In case of W_4 consider the cross correlation between codes on row 2 and row 3. R = [-1 1 -1 1] .x [-1 -1 1 1] = -1x-1 + 1x-1+-1x1 + 1x1 = 1+-1+-1+1 = 0

Since the cross correlation between two codes is zero, the two codes are orthogonal to each other.

One of the main advantages of Walsh-Hadamard codes is that the perfect orthogonal property and eliminates the effects of multi-access interference. In CDMA mobile communication system Walsh functions are used for two purposes. In the forward link, they are used as orthogonal cover to create independent transmission channels. In the reverse link, they are used as orthogonal modulation.

2.4 CDMA in Mobile Communication Air Interface [7]

A number of different terms are used to refer to CDMA implementations. The original standard spearheaded by Qualcomm was known as IS-95, the IS referring to an Interim Standard of the Telecommunications Industry Association (TIA). IS-95 is often referred to as 2G or second generation cellular. The Qualcomm brand name **cdmaOne** may also be used to refer to the 2G CDMA standard. After a couple of revisions, IS-95 was superseded by the IS-2000 standard. This standard was introduced to meet some of the criteria laid out in the IMT-2000 specification for 3G, or third generation, cellular. It is also referred to as **1xRTT** which simply means "1 times Radio Transmission Technology" and indicates that IS-2000 uses the same 1.25-MHz shared channel as the original IS-95 standard. A related scheme called 3xRTT uses three 1.25-MHz carriers for a 3.75-MHz bandwidth that would allow higher data burst rates for an individual user, but the 3xRTT scheme has not been commercially deployed. More recently, Qualcomm has led the creation of a new CDMA-based technology called **1xEV-DO**, or IS-856, which provides the higher packet data transmission rates required by IMT-2000 and desired by wireless network operators.

The Qualcomm CDMA system includes highly accurate time signals (usually referenced to a GPS receiver in the cell base station), so cell phone CDMA-based clocks are an increasingly popular type of radio clock for use in computer networks. The main advantage of using CDMA cell phone signals for reference clock purposes is that they work better inside buildings, thus often eliminating the need to mount a GPS antenna on the outside of a building. Also frequently confused with CDMA is W-CDMA. The CDMA technique is used as the principle of the W-CDMA air interface, and the W-CDMA air interface is used in the global 3G standard UMTS and the Japanese 3G standard FOMA, by NTT DoCoMo and Vodafone; however, the CDMA family of standards (including cdmaOne and CDMA2000) are not compatible with the W-CDMA family of standards.

CHAPTER 3
IMPLEMENTATION OF CDM SYSTEM IN FPGA

3.1 System Design

The Code Division Multiplexed (CDM) system consists of a two digital data sources, a multiplexing transmitter and a de-multiplexing receiver. The system block diagram is shown in figure 9.

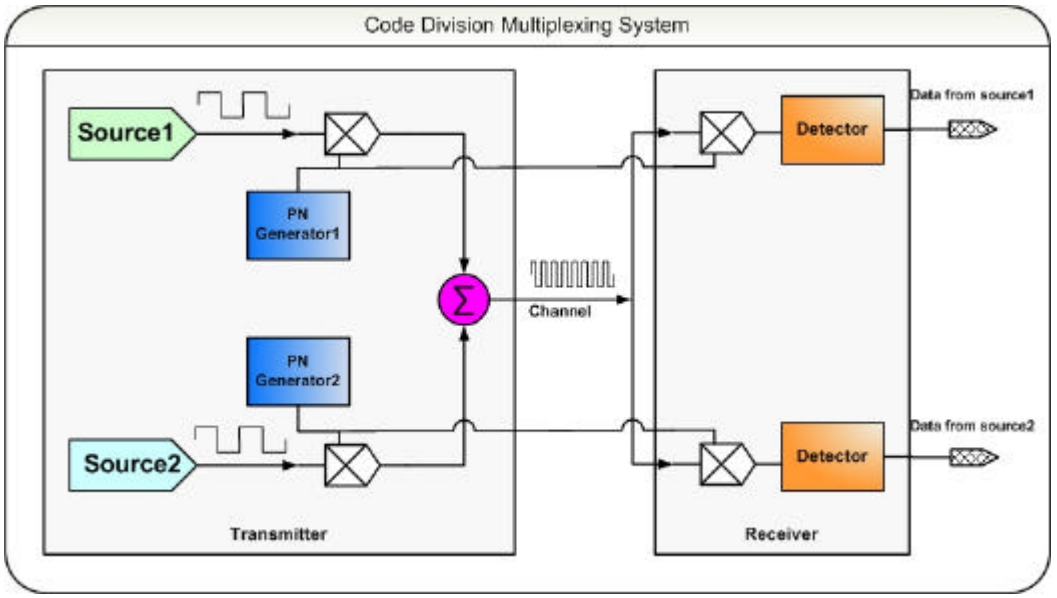

Figure 9. Code Division Multiplexing System Block Diagram

At the transmitter end, two data sources generate digital data sequences which are individually multiplied using two orthogonal code sequences. Each user is identified by its unique code. The orthogonal code sequences are generated by periodically repeating orthogonal codes at a rate much greater than the source data rate. The modulated sequence hence has wider frequency spectrum than the original source sequence and is called spread spectrum data sequence. It is imperative that the two codes are orthogonal to each other for the multiplexing operation to work. PN codes or Walsh Codes can be used as the orthogonal codes. The PN code sequence can be easily generated using a Linear Feedback Shift Register (LFSR) and is preferred over Walsh code for randomness and better spectrum spreading property. The two spread spectrum data sequences are finally summed up and transmitted over a common channel.

At the receiver end, spread spectrum data sequence received over the common channel is separately multiplied by orthogonal code sequences corresponding to each source on the

transmitter side. The detectors then use the multiplied sequence to determine whether the received data is a one or a zero.

In the above described system, **time synchronization** between transmitter and receiver is essential for proper detection. To ensure time synchronization, a single master clock generator is used as reference clock for the entire system. The orthogonal code generators should operate at master clock rate and the date source should operate at N fraction of master clock rate, where N is the length of orthogonal code.

3.2 System Implementation

The Code Division Multiplexing System was implemented completely using VHDL (**V**HSIC **H**ardware **D**escription **L**anguage) and synthesized onto **XC2S50 Spatran2 FGPA** on XSA prototyping board from XESS Corporation. The prototype board has an on board regulated power supply, 100MHz programmable oscillator (DS1075Z-100), PS/2 interface, parallel port, DIP switches, and a single digit display, fabricated onto a multilayer PCB. The prototype board hence provides a robust hardware platform for implementing the CDM system. VHDL was chosen as the hardware description language for reasons of previous experience. **Xilinx ISE 7.1i Project Navigator** was used for implementation and synthesis of VHDL codes.

The top level design was implemented in schematics and all components in the top level schematics were implemented in VHDL. The top level components are described next, and codes of important components are listed in Appendix C.

3.3 Top Level Components

3.3.1 Keyboard Interface (kbrd.vhd)

This component provides interface to PS/2 keyboard. It was implemented by modifying the interface entity described in application note from XESS. (See [11]). The component was designed to read only the number keys from 0 to 9 on the keyboard. The component was modified to generate 8-bit binary data corresponding to the number key pressed on the keyboard [figure 10]. The top level design uses two keyboard interface components. One of the keyboards was connected through the on board PS/2 port whereas second keyboard was connected through prototyping pins.

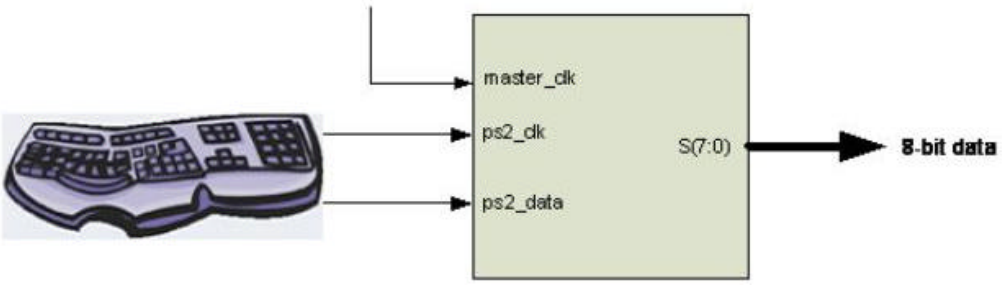

Figure 10. PS/2 keyboard interface component

3.3.2 Divide by seven (divide_by_seven.vhd)

This component is a 1/7 frequency divider whose output is a clock with 1/7 master clock frequency.

3.3.3 Data Source (src_Data.vhd)

This component reads in 8-bit data from keyboard interface and shifts out the data serially. It is composed of two entities **Divide by eight** (divclk8.vhd) and an **8-bit Parallel In Serial Out** (piso.vhd) This component is driven by clock of one seventh master clock rate and the output bit changes every 7^{th} master clock cycle. Thus the serial output is synchronized with PN sequence such that one data bit period equals seven PN code chips.

3.3.4 PN Generator (pngenerator.vhd, pngenerator2.vhd)

This component implements 7-bit PN code generator using three stages Linear Feedback Shift Register (LFSR). The logic circuit diagram of the 3 stage LFSR implemented in VHDL is shown in figure11.

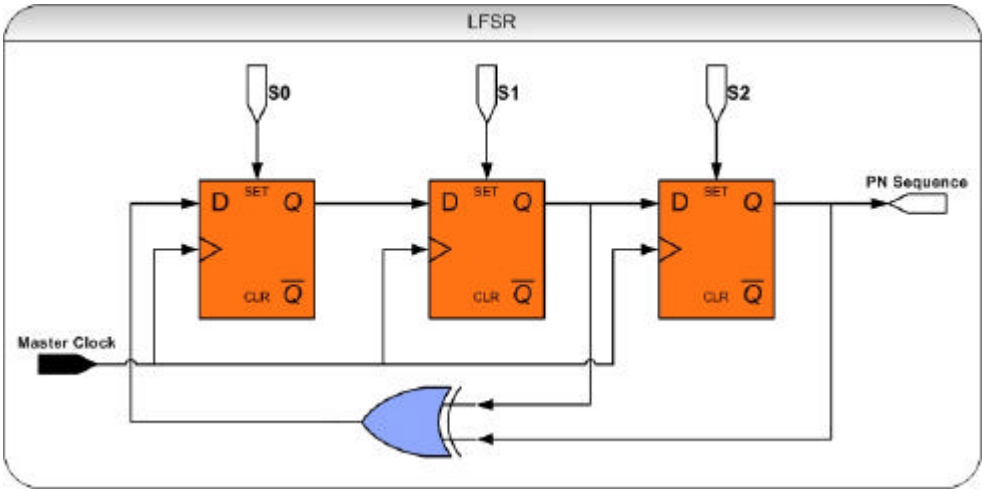

Figure 11. Three Stage LFSR

The Two separate PN generator entities were implemented, one each for two sources. The 3-bit initializer (seed) was set within the VHDL coding itself in each entity. The PN generator for Source 1 is initialized with (S2 S1 S0) = 111 and generates PN code **1110100** whereas the PN generator for Source 2 is initialized with (S2 S1 S0) = 110 and generates PN code **1101001**

3.3.5 Receiver (receiver.vhd)

This component implements the detector unit on the receiving side. The component is composed of a 7-bit comparator (comparator.vhd) and 7-bit Serial In Parallel Out (SIPO) register (sipo.vhd). The comparator operates at divide by seven clock and SIPO operates at master clock rate. The block diagram or receiver component is shown in figure 12.

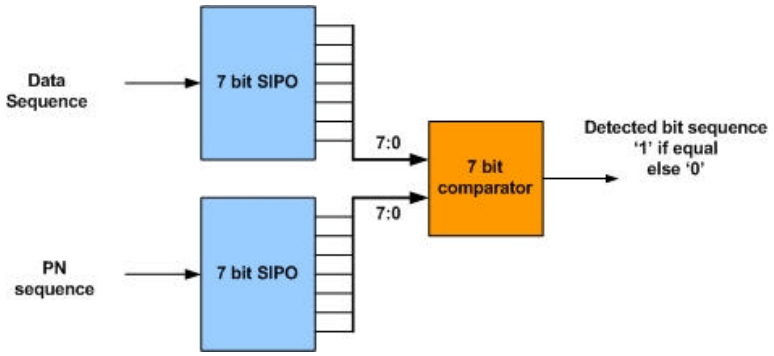

Figure 12. Receiver Block Diagram

3.3.6 Two Clock Cycle Counter (count2.vhd)

This is a special component which counts two clock cycles and then asserts its output port HIGH. The component provides the necessary synchronization at output section.

3.3.7 8-bit SIPO (sipo8.vhd)

This is a simple 8 bit shift register. The output from receiver sequentially stored in the register and every 7th master clock the 8-bit register content is latched out. This shift register starts shifting in data only after the Two Clock Counter asserts its output HIGH.

3.3.8 Seven segment decoder (ledDisp.vhd)

This component is simply a BCD to seven segment decoder implemented in VHDL. The component maps the input BDC to a 7-bit value corresponding to the configuration shown in figure13. The decoded data from each user is displayed on a seven segment display unit.

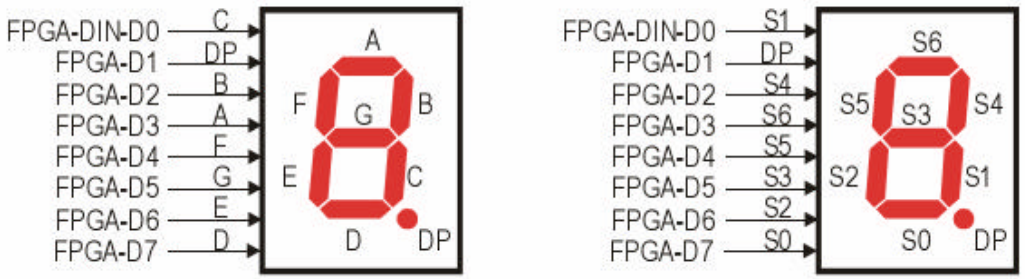

Figure 13. Seven Segment Single Display Pin Configurations

3.4 Synthesis

As mentioned earlier, the designed CDM system implemented in VHDL was synthesized onto XC2S50 FPGA using **Xilinx® Synthesis Technology (XST)** component in the Xilinx ISE 7.1i. When synthesized, the CDM system constituted of following elements

Elements	**Quantity**
ROMs	**2**
# 16x7 –bit ROM	2
Registers	**113**
# 1-bit register	96
# 14-bit register	2
# 4-bit register	5
# 5-bit register	2
# 7- bit register	4
# 8- bit register	4
Counters	**2**
# 4-bit up counter	2
Adders /Sub tractors	**7**
# 14-bit adder	2
# 4-bit adder	5
Comparators	**2**
# 7-bit comparators	2

The timing summary generated by XST stated

Minimum period: 10.778ns (Maximum Frequency: **92.782MHz**)

Minimum input arrival time before clock: 6.103ns

Maximum output required time after clock: 11.935ns

The timing summary after synthesis by Xilinx ISE, shows that the master clock frequency is constrained to 92.782 MHz. Although the XSA prototyping board had an onboard oscillator or 100MHz, the oscillator was reprogrammed to generate 50MHz clock to meet the constraints.

The onboard DIP switch was used for initializing the PN generators. The switches are in on state by default and keeps the registers in LFSR preset a fixed value. Only when toggled to off state, the LFSR starts generating the PN sequence.

3.5 System Operation

The CDM system was designed to operate on baseband data sequence. The onboard oscillator was used as the reference clock for the entire system. Two keyboards were interfaced to the XSA prototype board, which form two baseband data source. The keyboard interface is handled by the **keyboard interface.** The 8-bit data generated by keyboard interface is converted to serial data sequence by **data source** component.

The **pngenerator** components generate the respective PN sequences for two sources. The PN sequence is generated at a *chip rate* of seven times (7x) the serial data rate (1x). To ensure that seven chips in the PN sequence align exactly with one bit of serial data, the PN sequence is generated at the master clock rate i.e. master clock rate is assumed to be 7x. The serial data sequence is generated at one seventh rate of master clock. Let D1 and D2 represent the data bits from source1 and source2 respectively and C1 and C2 be the corresponding PN codes. Then in one bit duration of D_i, there are seven chips of C_i.

At the transmitting end, the serial data sequences are multiplied by their corresponding PN sequences. The multiplication operation is performed by logical AND operation on D_i and C_i. The output of **AND** operation is $D_i \cdot C_i$. Since the rate of C_i is 7x, the rate of the resulting product is same as that of C_i and hence the multiplication operation constitutes the **direct sequence spectrum spreading (DSSS)** operation. The spectrum of D_i has been spread by a factor of seven. The two spread spectrum signals $C1 \cdot D1$ and $C2 \cdot D2$ are summed up by logical **OR** operation. The output of OR operation is **SS = (C1·D1 + C2·D2)** which also has 7x data rate. This spread spectrum sequence is transmitted over a common channel.

At the receiving end, separate data recovery circuit (see figure 9) is designed for each of two sources. In each data recovery circuit, the received spread spectrum is multiplied by corresponding PN sequence. To ensure timing synchronization, the PN sequence from the PN generators at transmitting end is used. At this stage, the data sequence rate is still 7x. The **dispreading** and decision operation is based on following logic.

Consider the recovery of data sequence from source1. The result of AND operation at receiver is

Y =(C1·D1 + C2·D2)·C1

Y= C1·C1·D1 + C1·C2·D2

Y=C1·D1 + C1·C2·D2

If D1 = 0 and D2 = 0, then Y = 0

If D1 = 1 and D2 = 0, then Y = C1

If D1 = 0 and D2 = 1, then Y = C1·C2 ? C1

If D1 = 1 and D2 = 1, then Y = C1 + C1·C2 = C1·(1 + C2) = C1

The logic presented above shows that the whenever D1 = '1', the output of AND operation is same as the PN code from source1 (C1) and when D1 = '0', the output is not equal to the PN code. Similar logic can be drawn for source2.

Thus after the AND operation, the resulting sequence is simply compared with the PN code of the corresponding source. If the two are equal then '1' is detected, else '0' is detected. It should be noted that, since one bit of D_i corresponds to seven chips of PN code, for detection of one data bit, the 7-bit sequence resulting after AND operation is to be compared with 7-bit PN code. Hence the use of 7-bit SIPO and comparators in the recovery circuit as shown in figure9. The comparison occurs every 7^{th} master clock cycle, hence the output detected bit sequence has data rate of 1x. This constitutes the **dispreading** operation.

CHAPTER 4
CDMA SYSTEM SIMULATION

4.1 Simulation Background

The CDMA system simulation attempts to emulate a simple CDMA communication link. It uses Direct Sequence Spread Spectrum (DSSS) technique and includes effects of Additive White Gaussian Noise (AWGN) and Multi User Access Interference (MAI). The simulation uses N data sources of which only one is the *desired user* whose data is to be recovered at the receiver and the remaining N-1 users are considered as *interferers*. Walsh codes are used for user identification as well as for spreading the data sequence.

The simulator estimates BER (Bit Error Rate) as a function of channel signal to noise ratio (SNR), Number of Interferers (NoI) and spreading factor (SF). The block diagram of the simulator is shown in figure 14.

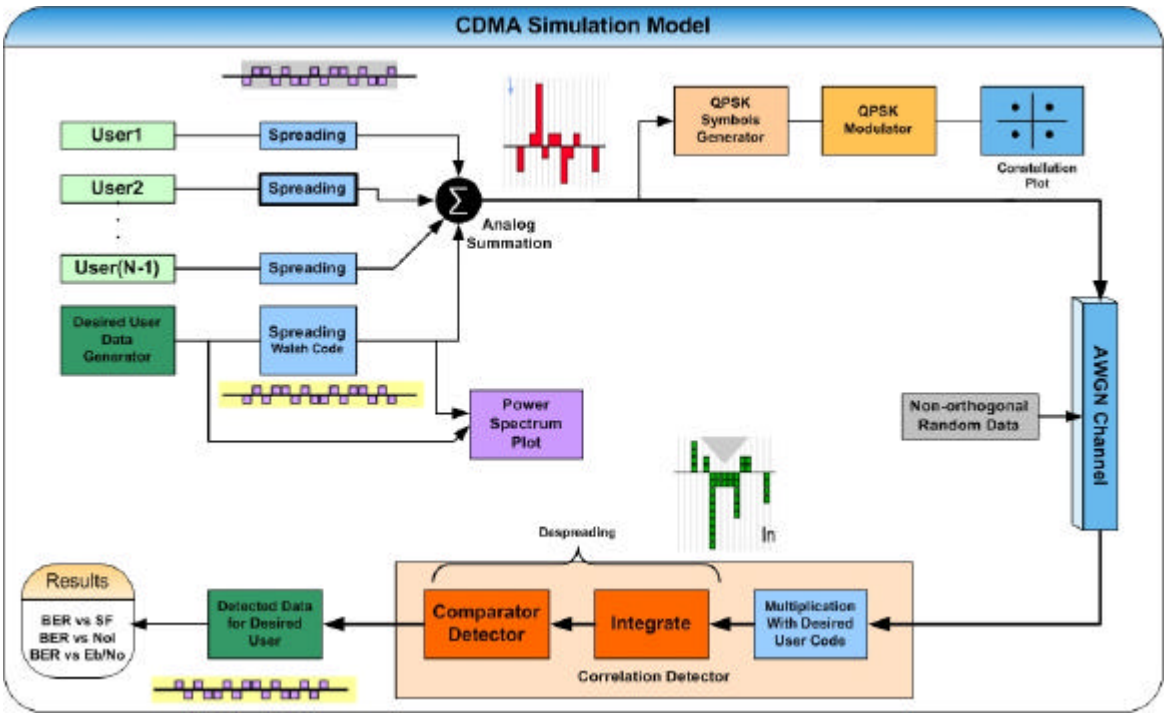

Figure 14. CDMA Simulator Block Diagram

In order to simplify the simulation some assumption were made. The simulation does not use multipath fading channel and hence the receiver is a simple correlator receiver and not a RAKE receiver. Baseband data was used for transmission over the channel at it is computationally less extensive compared to pass band simulation. No equalization or

matched filter was used. These assumptions simplify the simulation work however the results may deviate from real systems.

4.2 Simulation System Description

The simulation was implemented in MATLAB. The components implemented in the simulation are described below.

4.2.1 User Data Generation and Spreading

Digital bipolar user data of fixed length are generated randomly for given number of users. The built in MATLAB function *randsrc()* is for this purpose. Each symbol represented by these bipolar data is then over sampled by a factor equal to the Walsh code length. This is done to simulate the relative frequencies difference between the Walsh code and the user symbols. Each user's data is then spread by multiplying each symbol with the corresponding Walsh code. The Walsh matrix generator was implemented in *walsh.m*. The user data generation was implemented in *multiUser.m* and the spreading operation was implemented in *spreadData.m*.

4.2.2 Channel

Channel is the medium through which the signals are transmitted. A communication channel can be generally characterized as liner filters and signals transmitted over the channel suffer distortion due to channel response and noise.

In CDMA, since all user channels operate simultaneously in the same RF band, the interference due to the multiple users in same RF channel become more important factor and it plays vital role in deciding the performance of the communication system. A mobile channel includes effects of multipath fading and noise. In the simulation however, only Additive White Gaussian Noise (AWGN) channel has been used, which is the simplest channel models, from analysis point of view. A built in command *awgn()* was used to model AWGN channel.

In the simulation communication link has been modeled as a forward channel where each user data is uniquely identified and spread using Walsh codes. As mentioned earlier, the perfect orthogonality of Walsh codes significantly reduces the probability of error at receiver. In order to simulate bit errors due to multipath effect, intercellular interferers, random non orthogonal data were added onto the AWGN channel.

These interfering signals are not orthogonal with the Walsh codes used for the channels in the concerned cell and thus causes substantial amount of error. However in order to simulate error due to the multipath interferences from the users operating in the same cell or the

interferences coming from users in other cells, random data was added in the channel that may not be exactly orthogonal to the Walsh codes.

4.2.3 Correlation Detector

A correlation detector was used for the reception of the transmitted signal. The correlator detector has been implemented in *corrDetector.m* according to block diagram in figure 15.

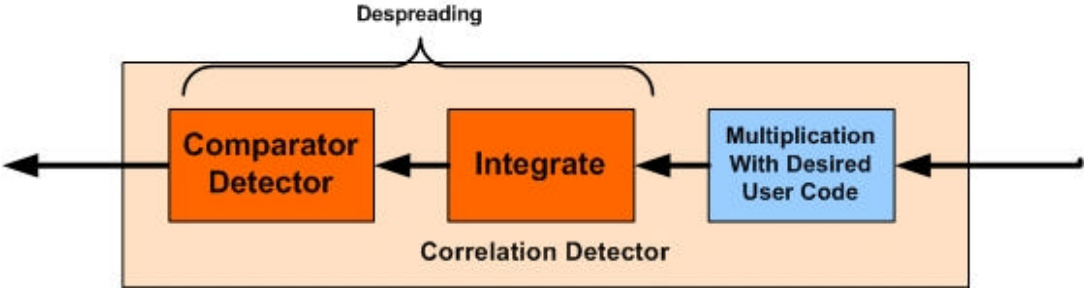

Figure 15. Correlation Detector Block Diagram

The received channel signal is despread by multiplying with the Walsh code of the desired user. In ideal case the despreading operation completely suppresses the effect of signals from other users, due to perfect orthogonality of Walsh code. But the noise and interference present in the real channel inhibits complete suppression of interfering signals and also distorts the desired user's data resulting in some error at reception.

Thus despread data is then summed up by an integrator. The sum is finally compared with a threshold set at zero and the decision is made in favor of bit '1'if the sum is positive or bit '0' if the sum is negative.

Initially, QPSK modulation was used for transmission of source data over the channel. The built in MATLAB functions *pskmod()* and *pskdemod()* was used. However these commands require the input data to be in a specific format and our user data sequence did not match with the required format. Numerous operations were performed to transform the used data sequence to the required data format. However, these transformation operations had undesired effects in the entire simulation resulting in unrealistic results. Thus the modulation demodulation operation was discarded and simple base band transmission was used.

The complete simulation was integrated in the m-file *cdmaSim.m*.

CHAPTER 5
RESULTS AND DISCUSSION

5.1 FPGA Implementation Verification

PN generator is the most critical component in the CDM system. This was the first component to be designed, implemented and synthesized on the FPGA. The PN generator output for initializing code 111 as observed on the oscilloscope is shown in figure 16. In the figure a 7-bit PN code **1110010** can be seen aligned with the seven master clock cycles.

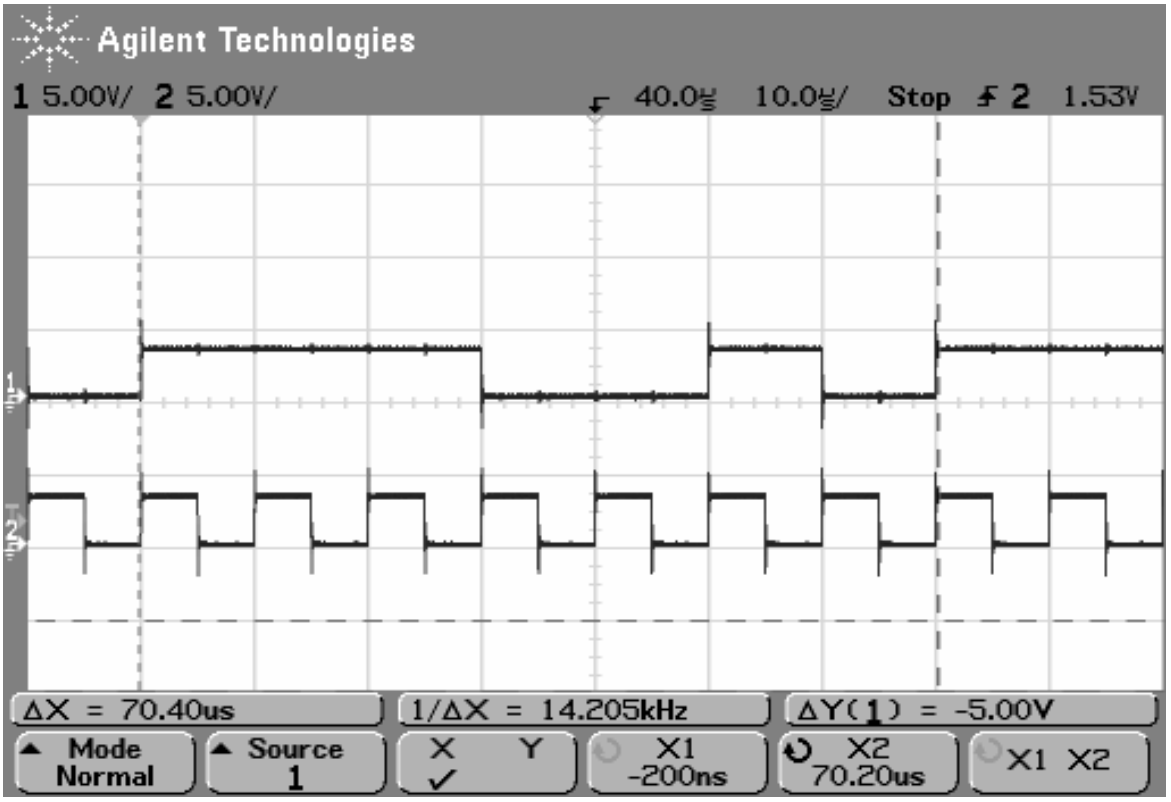

Figure 16. 7-bit PN sequence (1110010) observed on oscilloscope

Secondly, spreading and dispreading components were implemented with single data source. The system was tested with 50MHz master clock. The input was given through a keyboard and the decoded data at the receiving end were observed on onboard 7-segment display. The spreading and dispreading operations were correctly verified.

Finally, the complete CDM system was implemented with two data sources, the multiplexing transmitter, demultiplexing receiver and display units as shown in the figure17. The system was tested with two keyboards connected to the prototyping board acting as data source, onboard 7-segment display as output for source1 and an externally interfaced 7-segment

display as output for source2. The transmitter output (**SS_DATA**) was assigned to FPGA pin 79 and the receiver input (**SS_INPUT**) was assigned to pin 80. The two pins were connected with a single wire forming the common channel between the transmitter and the receiver. It was observed that, the keys pressed on the keyboard were successfully decoded and displayed on their respective 7-segment displays. When the common channel was removed, no data was received; this verified that the common channel was in fact carrying the signal between the transmitter and receiver. Thus the data from two sources multiplexed over a common channel and demultiplexed at the receiver was successfully verified.

The oscilloscope probes in the laboratory were designed conduct maximum of 10MHz sinusoidal signal. But master clock used was a 50MHz return to zero rectangular waveform, whose side lobes extended well beyond 100MHz. Thus attempts to observe the transmitted data sequence on oscilloscope showed only highly distorted signals. Only when master clock was set below 1MHz range, rectangular pulse waves could be observed on the oscilloscope.

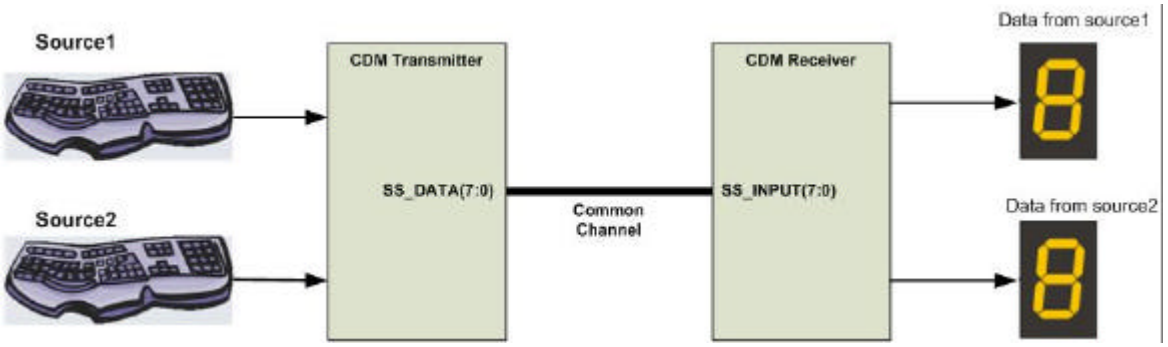

Figure 17. Implemented CDM Prototype

5.2 Simulation Results and Discussion

Initially the CDMA simulation was run to verify the operation of individual components such as Walsh code generator, user data generator, spreading operation. A 64-bit Walsh code generated by walsh.m is shown in figure 18.A 4-bit user data with 64 samples in each bit is shown in figure 19.

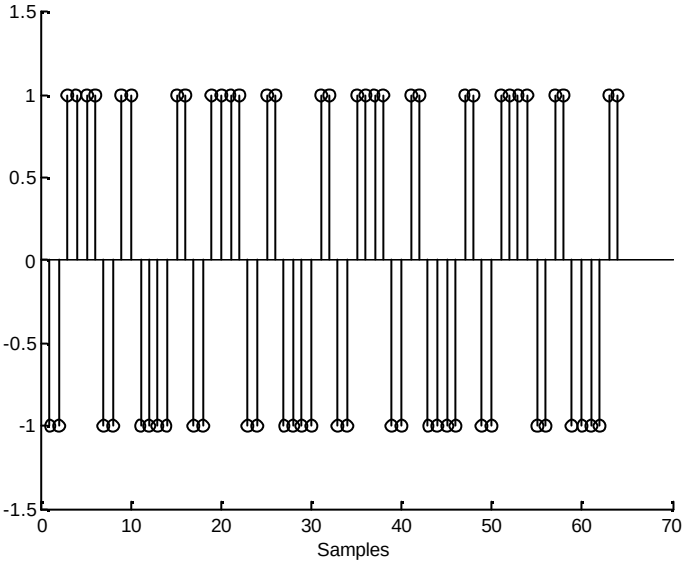

Figure 18. 64-bit Walsh Code generated using walsh.m

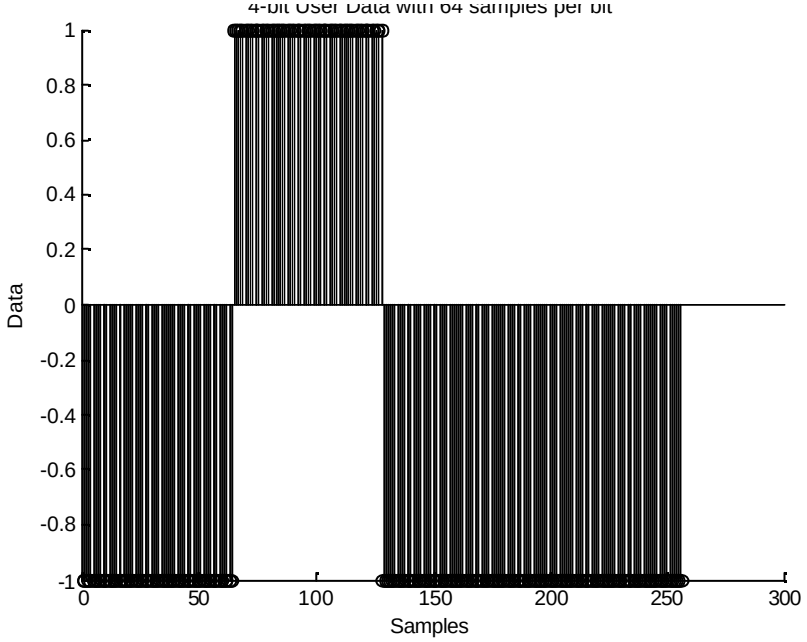

Figure 19. 4-bit User Data with 64 samples per bit

Similarly to verify the spreading operation, power spectrum density plots of user data sequence before and after spreading were observed. The resulting plots are shown in figure 20. It was seen that spread spectrum was wider bandwidth compared to the bandwidth of original user data.

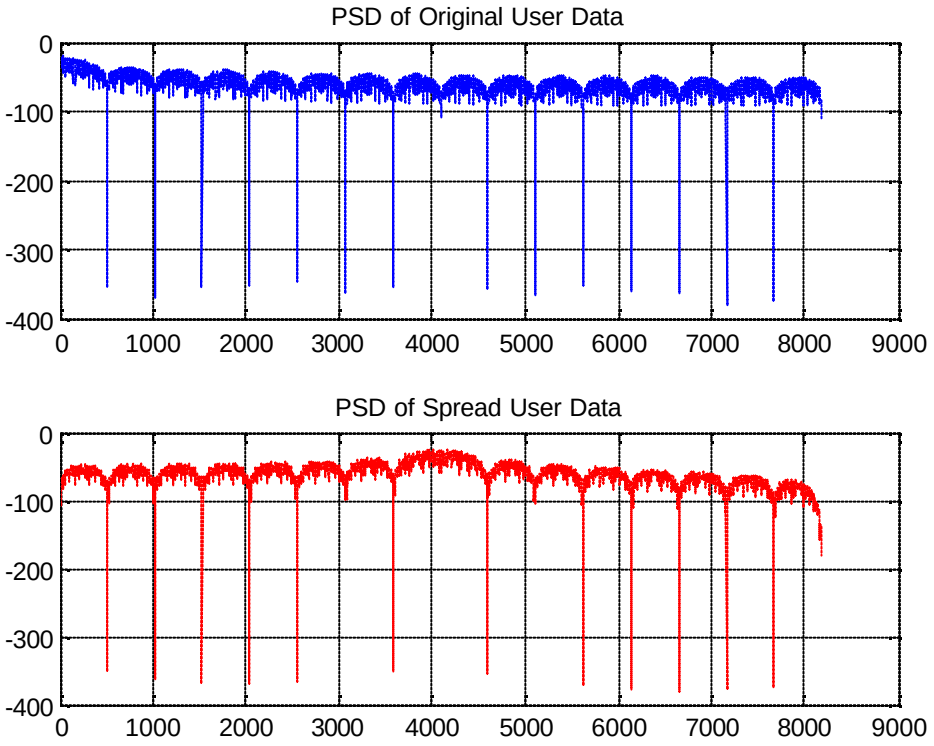

Figure 20. PSD plot of user data and spread data

Finally, the complete CDMA simulator was run under different test conditions and the corresponding results were analyzed to verify the correctness of the simulation. The tests performed and the results are explained below.

5.2.1 Multiplexing with orthogonal codes

The first test was to verify the need of orthogonal codes for code division multiplexing. In the simulation Walsh codes have been used to uniquely identify each user. However when non orthogonal codes were used to identify each user, the multiplexing operation resulted in unacceptably high error rates at receiver. Thus the need for orthogonal codes in code division multiplexing was verified.

5.2.2 BER vs NoI

In this test, the number of interfering users was varied for a fixed value of spreading factor and SNR. The resulting BER plot is shown below

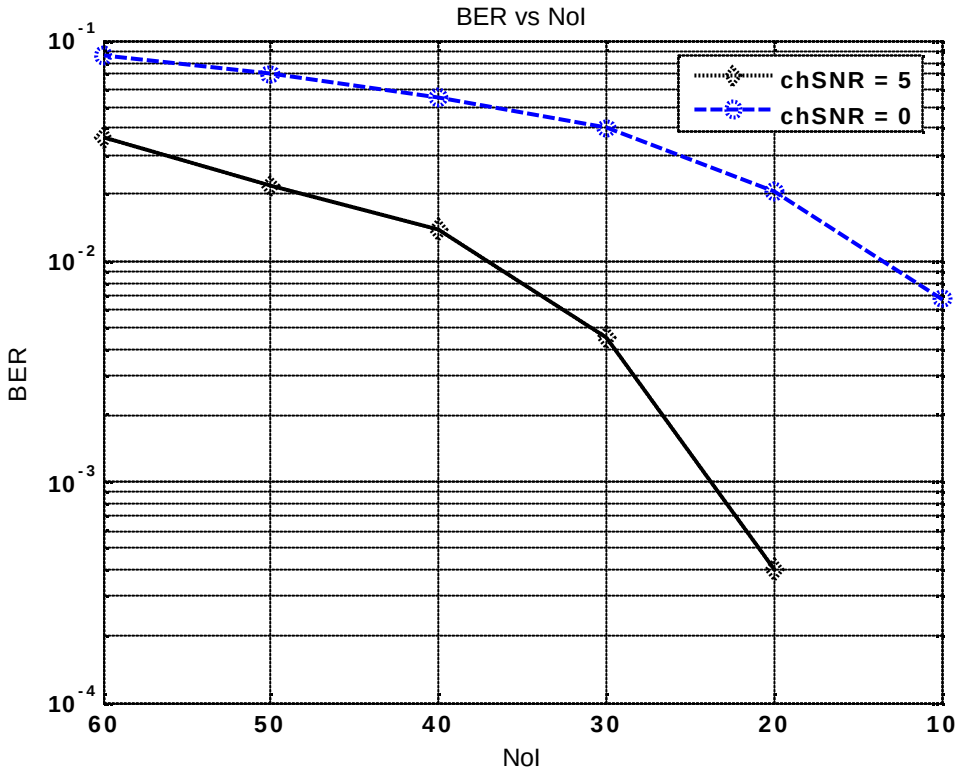

Figure 21. BER vs Number of Interferers

The resulting plot shows that BER decreases with decreasing number of interferers. Further the BER values corresponding to SNR = 5 was lower than the BER values corresponding to SNR= 0. The results are consistent with theory.

Next, the BER performance with non orthogonal random data added in the channel was compared with the BER performance without the random data. It was observed that BER performance degraded when random data was added to the channel as shown in the figure 22

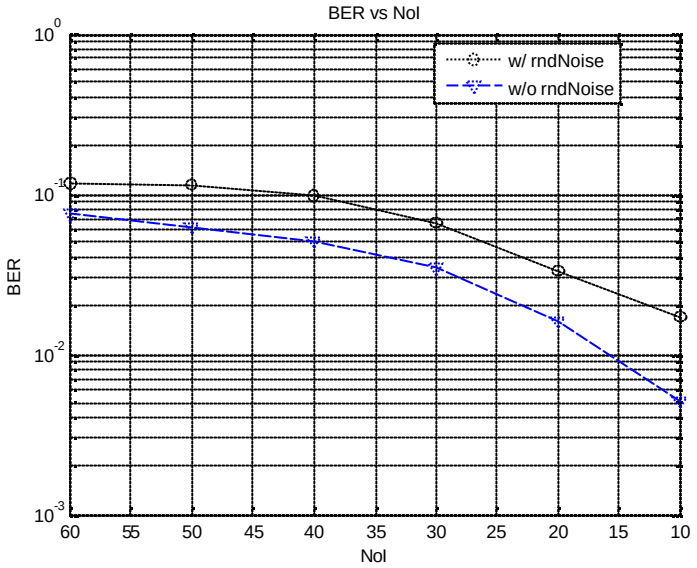

Figure 22. BER vs Number of Interferers with channel noise

5.2.3 BER vs Spreading Factor

In this test, BER performance for different values of spreading factor was studied. The resulting BER plot is shown in figure 23. Spreading factors 8, 16, 32, and 64 were used for two values of channel SNR. It was observed that the BER decreased with increasing spreading factor. The processing gain due to spreading of data compensates for the noise added in the channel and hence increasing spreading factor results in decreased BER

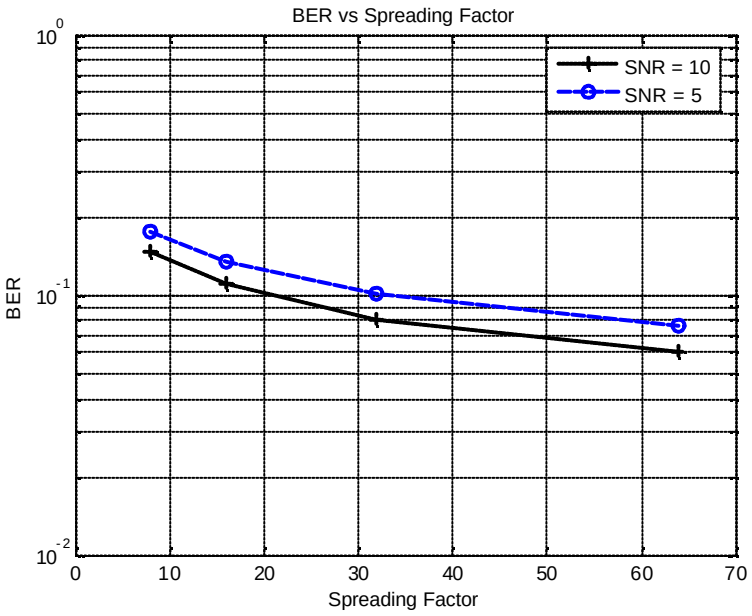

Figure 23. BER vs Spreading Factor

5.2.4 BER vs channel SNR

In this test BER performance for different values of the channel SNR was studied. The simulation as run for five values of SNR [0 2 4 6 8]dB with spreading factor of 32 and 32 user. The resulting BER plot is shown in the figure 24. It is clearly seen that BER decreases with increasing SNR which is consistent with the theory.

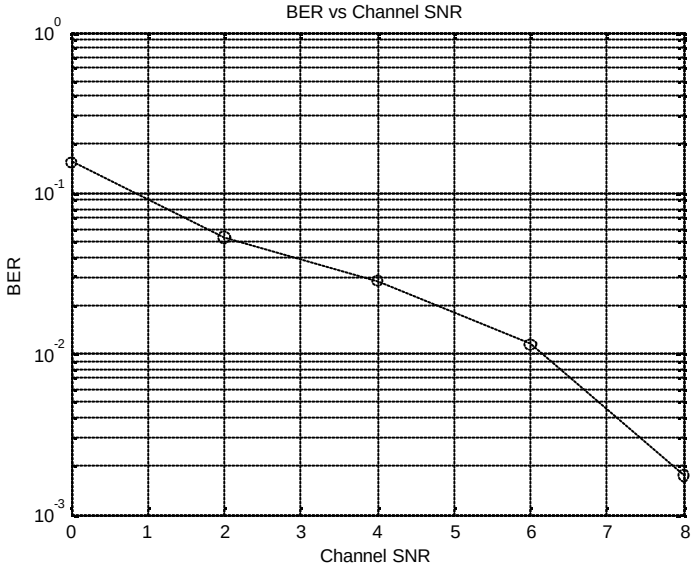

Figure 24. BER vs Channel SNR

5.2.5 Effects of AWGN

In order to study the effects of AWGN channel, the user data was QPSK modulated and passed through AWGN channel. It was observed that the constellation points were scattered around the ideal locations. The resulting constellation plot is shown in figure 25.

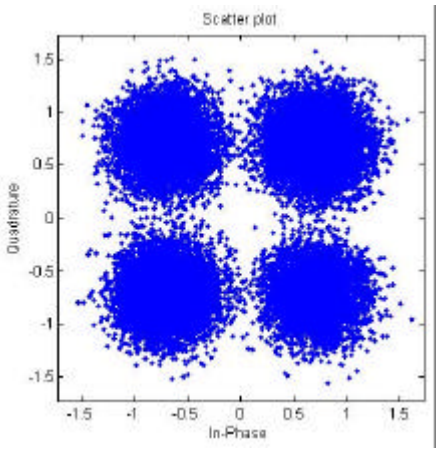

Figure 25. Constellation Diagram of QPSK Signal with AWGN

CHAPTER 6
CONCLUSION AND RECOMMENDATION

6.1 Accomplishments

This project was carried out with two major objectives of implementing a CDM system at hardware level and simulating a CDMA communication system in MATLAB. The first objective was successfully accomplished after extensive designing, coding and testing. A two-user CDM system prototype was implemented in VHDL and synthesized onto an FPGA prototyping board. The prototype system was kept simple to ease the design and testing process. The operation of the system was successfully verified.

Similarly in case of the second objective, MATLAB simulation of a CDMA communication link was successfully developed. A multiplexed channel with N users uniquely identified by Walsh codes was designed and simulated. The simulation was run to obtain system performance results under different conditions. However a number of assumptions were made while designing the simulation model which leaves a large area for further improvement.

6.2 Future Enhancements

The CDM system prototype in FPGA currently supports only two sources. The system can be easily upgraded to support multiple sources. Further this system can developed to be used for multiplexed serial data transmission over a common channel. Further, nested spreading can be implemented to achieve greater spreading factor and hence improve performance.

The CDMA system simulation can be developed to include more realistic channel models like Rayleigh and Rician fading channels. Further the simulation can be developed to realize the IS-95 Forward link modulation process.

In conclusion, the project work was finished within stipulated project duration and by working on this project, difficulties faced in implementing a communication system and challenges faced in simulating a real system were understood.

BIBLIOGRAPHY

[1] W. Stallings, *"Wireless Communications and Networking"*, p168-184,PHI, 2004.

[2] T.S. Rappaport, *"Wireless Communications Principles and Practice"*, p456-458,PHI 2005.

[3] F. Ansari, J. Kurtzweil, M. Shabib,*"Code Division Multiple Access Communication System"*,1999.

[4] Harold P.E. Stern, Samy A. Mahmoud, *"Theory of Code Division Multiplexing, Communication Systems: Analysis and Design"*, Prentice Hall PTR.

[5] S. Baxter, *"Technical Introduction to CDMA v4.0"*, 2005.

[6] Shah, *"Code Division Multiple Access : A Tutorial"*, p6-10,Wireless Communication, Rowan University,1999

[7] Wikipedia, *"Code Division Multiple Access"*, http://en.wikipedia.org/wiki/CDMA

[8] Wikipedia, *"Orthogonality"*, http://en.wikipedia.org/wiki/Orthogonality

[9] Wikipedia, *"Linear feedback shift register"*, http://en.wikipedia.org/wiki/Linear_feedback_shift_register.

[10] K. Skahill, *"VHDL for Programmable Logic"*, Addison-Wesley,1996

[11] XESS Corporation, *"XSA Board V1.1, V1.2 User Manual"*,2005

[12] D. Vanden Bout, *"PS/2 Keyboard Interface for the XSA Boards"*, XESS Corporation, 2004

APPENDIXES

Appendix A: Field Programmable Gate Array (FPGA)

A field programmable gate array (FPGA) is a semiconductor device containing programmable logic components and programmable interconnects. The programmable logic components can be programmed to duplicate the functionality of basic logic gates such as AND, OR, XOR, NOT or more complex combinational functions such as decoders or simple math functions. In most FPGAs, these programmable logic components (or logic blocks, in FPGA parlance) also include memory elements, which may be simple flip-flops or more complete blocks of memories.

A hierarchy of programmable interconnects allows the logic blocks of an FPGA to be interconnected as needed by the system designer, somewhat like a one-chip programmable breadboard. These logic blocks and interconnects can be programmed after the manufacturing process by the customer/designer (hence the term "field programmable") so that the FPGA can perform whatever logical function is needed.

Spartan™ II FPGA

The Spartan™-II 2.5V Field-Programmable Gate Array family gives users high performance, abundant logic resources and a rich feature set, all at an exceptionally low price. The six-member family offers densities ranging from 15,000 to 200,000 system gates, as shown in Table1. System performance is supported up to 200 MHz. Spartan-II devices deliver more gates, I/Os, and feature per dollar than other FPGAs by combining advanced process technology with a streamlined Virtex-based architecture. Features include block RAM (to 56K bits), distribute RAM (to 75,264 bits), 16 selectable I/O standards, and four DLLs. Fast, predictable interconnect means that successive design iterations continue to meet timing requirements. The Spartan-II family is a superior alternative to mask-programmed ASICs. The FPGA avoids the initial cost, lengthy development cycles, and inherent risk of conventional ASICs. Also, FPGA programmability permit design upgrades in the field with no hardware replacement necessary (impossible with ASICs).

XSA Board

XSA-50 Board contains the following components:

XC2S50 or XC2S100 Spartan-II FPGA: This is the main repository of programmable logic on the XSA Board. XC9572XL CPLD: This CPLD manages the interface between the PC parallel port and the rest of the XSA Board.

Oscillator: A programmable oscillator generates the master clock for the XSA Board. Flash: A 128 or 256-KByte Flash device provides non-volatile storage for data and configuration bits streams.

SDRAM: An 8 or 16-MByte SDRAM provides volatile data storage accessible by the FPGA.

LED: A seven-segment LED allows visible feedback as the XSA Board operates.

DIP switch: A four-position DIP switch passes settings to the XSA Board or controls the upper address bits of the Flash device.

Pushbutton: A single pushbutton sends momentary contact information to the FPGA.

Parallel Port: This is the main interface for passing configuration bit streams and data to and from the XSA Board.

PS/2 Port: A keyboard or mouse can interface to the XSA Board through this port.

VGA Port: The XSA Board can send signals to display graphics on a VGA monitor through this port.

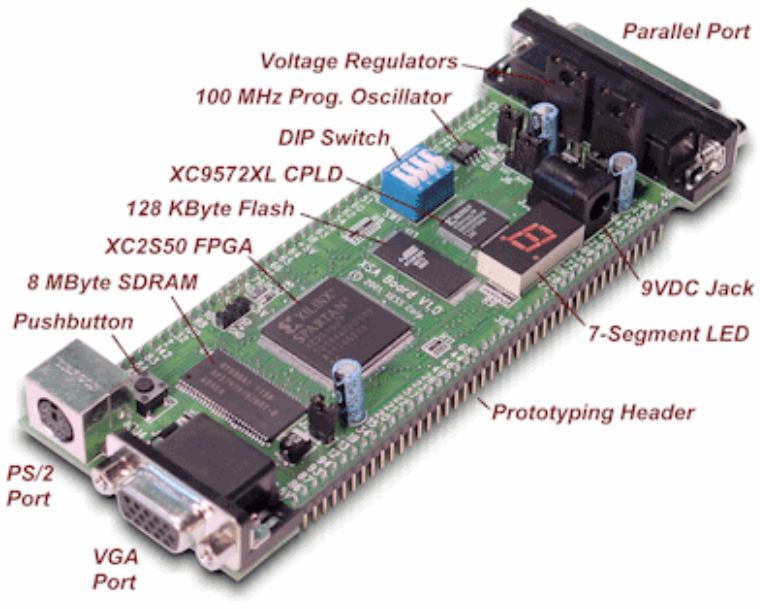

Figure A.1. XSA-50 Prototype Board

Appendix B: Optimum Tap Points for Maximal Length Sequence

The choice of which feedback taps to use determines how many values are included in a sequence of pseudo-random values before the sequence is repeated. Certain tap settings yield the N maximal length sequences of (2^n -1).

Some of the feedback connections for a given value of *m* are presented in Table1.

Table 1: m-sequence feedback connections

m	$N = 2^m - 1$	Feedback for *m*-sequences	Number of *m*-sequences
2	3	[2,1]	2
3	7	[3,1]	2
4	15	[4,1]	2
5	31	[5,3][5,4,3,2][5,4,2,1]	6
6	63	[6,1][6,5,2,1][6,5,3,2]	6
7	127	[7,1][7,3][7,3,2,1][7,4,3,2][7,6,4,2][7,6,3,1] [7,6,5,2][7,6,5,4,2,1][7,5,4,3,2,1]	18
8	255	[8,4,3,2][8,6,5,3][8,6,5,2][8,5,3,1][8,6,5,1] [8,7,6,1][8,7,6,5,2,1][8,6,4,3,2,1]	16
9	511	[9,4][9,6,4,3][9,8,5,4][9,8,4,1][9,5,3,2][9,8,6,5] [9,8,7,2][9,6,5,4,2,1][9,7,6,4,3,1][9,8,7,6,5,3]	48
10	1023	[10,3][10,8,3,2][10,4,3,1][10,8,5,1][10,8,5,4][10, 9,4,1] [10,8,4,3][10,5,3,2][10,5,2,1][10,9,4,2][10,6,5,3,2 ,1][10,9,8,6,3,2][10,9,7,6,4,1][10,7,6,4,2,1][10,9, 8,7,6,5,4,3] [10,8,7,6,5,4,3,1]	60
11	2047	[11,2][11,8,5,2][11,7,3,2][11,5,3,2][11,10,3,2][11 ,6,5,1][11,5,3,1][11,9,4,1][11,8,6,2][11,9,8,3][11, 10,9,8,3,1]	176

Appendix: C: VHDL Codes

PN Generator (pngenerator.vhd)

```
entity PNGenerator is
    Port ( CLOCK : in std_logic;
           INIT : in std_logic;--_vector (2 downto 0);
                    PNCode : out std_logic);
end PNGenerator;
architecture Behavioral of PNGenerator is
      component DFF
            Port ( CLK : in std_logic;
                D : in std_logic;
                         CLEAR : in std_logic;
                         PRESET : in std_logic;
             Q : out std_logic);
      end component DFF;
      component XOR2     --Using UNISIM Library
            port (O : out STD_LOGIC;
                        I0 : in STD_LOGIC;
                        I1 : in STD_LOGIC);
      end component;
signal Q0,Q1:std_logic;
signal Q2: std_logic := '0';
signal x : std_logic;
begin
-- DFF is D-type Flip Flop Entity
S0 : DFF port map (CLOCK,x,'0',INIT,Q0);
S1 : DFF port map (CLOCK,Q0, '0',INIT,Q1);
S2 : DFF port map (CLOCK,Q1, '0',INIT, Q2);
XX :  XOR2 port map (x,Q1,Q2);
process(CLOCK)
begin
if(rising_edge(CLOCK) and INIT = '0') then
PNCode <= Q2;
end if;
end process;
end Behavioral;
```

Receiver (receiver.vhd)

```
entity receiver is
    Port ( clk : in std_logic;
           clkdiv7 : in std_logic;
           datain : in std_logic;
           pncode : in std_logic;
           dataout : out std_logic);
end receiver;

architecture Behavioral of receiver is

component sipo
           Port ( clk : in std_logic;
           clkdiv7 : in std_logic;
           serialin : in std_logic;
           parrout : out std_logic_vector(6 downto 0));
 end component;

 component comparator
    Port ( clkdiv7 : in std_logic;
              parrdata:std_logic_vector(6 downto 0);
                    parrpn:std_logic_vector(6 downto 0);
           dataout : out std_logic);
end component;

signal parrdata: std_logic_vector(6 downto 0);
signal parrpn: std_logic_vector(6 downto 0);
signal compout : std_logic := '0';

begin
 sipo1 : sipo port map( clk,clkdiv7,datain,parrdata);
 sipo2 : sipo port map( clk,clkdiv7,pncode,parrpn);
 comp: comparator port map(clkdiv7,parrdata,parrpn,compout);
 dataout <= compout;
end Behavioral;
```

Appendix D: MATLAB Codes

Walsh Hadamard Matrix Generator(walsh.m)

```
function [ W ] = walsh( N )
% Walsh Code Matrix Generator
% W = Hadamard matrix of order N where N is a power of 2
% Bit convention: '0'=> +1 AND '1' => -1

[f,e] = log2(N);
k = find(f==1/2 & e>0);
if min(size(N)) > 1 || isempty(k)
   error('MATLAB:hadamard:InvalidInput',...
         'n must be an integer and n must be a power of 2.');
end
%-------------------------------------------------------------

W = -1;     %Initial value as bit 0
for i = 1:e-1,
    W = [W  W;
         W -W];
end
```

Correlator Detector (corrDetector.m)

```
function [ det_data ] = corrDetector(rx_data_sym,code,N,DL )
%Correlator Detector for CDMA
% ds_data -> Despread Data
% N -> Spreading Factor/ Correlation Value
% DL -> Data Length

%***********Multiplication
Operation************************************
ds_data = [];
 for i = 0:DL-1,
     ds_data = [ds_data rx_data_sym(1 + i * N : N + i * N) .* code];
 end

%**********Summation and Detection-
AD**********************************
det_data = [];
for i = 0:DL-1,
    Cor = sum(ds_data(1 + i*N :N + i * N));
    if Cor >= 0
        DR = 1;
    elseif Cor < 0
        DR = 0;
    %else
     %   DR = 1;  %Error bit is  1
    end
    det_data = [det_data DR];
end
%-------------------------------------------------------------------------
```

Multiple User Data Generator (multiUser.m)

```
function [userData_DL userData] = multiUser(NoI,N,DL)

%userData_DL -> User data bits
%userData -> Over sampled user data

userData_DL = zeros(NoI,DL); %Data matrix, Each row contains data for each

userData = zeros(NoI,DL*N);
for i = 1:NoI,
    userData_DL(i,:) = randsrc(1,DL);
end

for i = 1:NoI,
      userData(i,:) = rectpulse(userData_DL(i,:),N);
end

userData_DL = 0.5*(userData_DL + 1); %Convert to binary format
```

Main Simulation cdmaSim.m

```
%Initialization
 clear all;
 close all;

Simulation Constants
DL = 256; %Data Length
N = 64; %Spreading factor/ Walshcode length
M = 4;  %M-ary modulation
NoI = 63; %Number of users/interferers
SNR = 10;
U = 6;
%*************************************************************************
%User Data Generator (Uniform Distribution Random)-SA
[userData_DL userData] = multiUser(NoI,N,DL);
%userData_DL(3,:)

%************************TRANSMITTER
SECTION*******************************
%Walsh Code Generation
W = walsh(N);
code1 = W(U,:);
%*************************************************************************
%Spreading Operation
user_ss_data = zeros(NoI,N*DL);
for x = 1:NoI,
    user_ss_data(x,:) = spreadData(userData(x,:),N,DL,W(x,:));
end

%*************************************************************************
%% Modulation Quadrature Phase Shift Keying(QPSK) - SRT
[RR CC] = size(user_ss_data);
 bb_data = sum(user_ss_data);
%  bb_data_bin = de2bi(bb_data)';
% [R C] = size(bb_data_bin);
```

```
% bb_data_bin = reshape(bb_data_bin,1,R*C);
% mod_data = modulation(bb_data_bin,R,C,M);   %Modulation Command
% scatterplot(awgn(mod_data,SNR));
 tx_data = bb_data;
% tx_mp = zeros(1,length(tx_data) + 2)
%tx_mp = [tx_data 0 0 0 0]+ [0 rand*tx_data 0 0 0] + [0 0 0 rand*tx_data
0]; %Multipath
%******************Transmission Channel(Channel Modeling)-
NMS*************
%Transmit over channel
ch_data = awgn(tx_data,SNR,'measured') +
randint(1,length(tx_data),[round(-NoI/4) round(NoI/4)]);;    %AWGN
Transmission
%*************************************************************************
*
 %scatterplot(ch_data);           %Constellation Plot

%*********************RECEIVER SECTION***********************************
%Following four steps perform the necessary matrix manipulation to obtain
%the data in desired format
rx_data_sym = ch_data(1:DL*N);%bi2de(demod_data')' - 64;  % Sub
%***********************Despreading and Detection************************
det_data1 = corrDetector(rx_data_sym,code1,N,DL);
%det_data2 = corrDetector(rx_data_sym,code2,N,DL);

%*********************END of SIMULATION*********************************

[N R] = biterr(userData_DL(U,:),det_data1);
%**********Result Display*******************
ber = R;
display('Simulation Results')
display(['Bit Error Rate(BER) is ' num2str(R) 'No. of errors is'
num2str(N)])

%******SPECTRUM PLOTS***********************
% D = fft(userData(3,:));
% D = D(1:length(D)/2);
% SS = fft(user_ss_data(3,:));
% SS = SS(1:length(SS)/2);
% figure('Name','Spread Spectrum');
% title('Spread Spectrum');
% subplot(211);
% plot(10*log10(D.*conj(D)));
% grid on;
%     subplot(212);
%     plot(10*log10(SS.*conj(SS)),'r');
%     grid on;
%    hold off;
    %legend('Original','Spread');
```