FPGA IMPLEMENTATION OF MIMO SYSTEM FOR SYMBOL-WAVELENGTH-SPACED ANTENNAS

by

Harshal Desai

B.E., Mumbai University, 2003

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science in Engineering

In the Graduate Academic Unit of Electrical and Computer Engineering

Supervisors:	Mary E. Kaye, M.Eng., Department of Electrical and Computer Engineering Brent R. Petersen Ph.D., Department of Electrical and Computer Engineering
Examining Board:	Christopher P. Diduch, Ph.D., Department of Electrical and Computer Engineering Bruce G. Colpitts, Ph.D., Department of Electrical and Computer Engineering, Chair Yevgen Biletskiy, Ph.D., Department of Electrical and Computer Engineering
External Examiner:	Przemyslaw R. Pochec, Ph.D., Faculty of Computer Science

This thesis is accepted

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

May, 2007

To the advancement of science and technology.

Abstract

For Line-of-Sight (LOS) radio channels, recent research shows that in order to improve effectively the performance of a Multi-input Multi-output (MIMO) system, the antenna elements at the receiver must be separated on the scale of a symbol wavelength ([speed of light]/[symbol rate]). The main focus of this thesis was to design, implement and test a two-by-two (2X2) system based on that separation. The system was implemented on a single Field Programmable Gate Array (FPGA) board. The adaptive Space-Time (ST) receiver in the system includes a Least Mean Square (LMS) algorithm to adapt the coefficients. The system was developed using Altera Quartus II software and Verilog HDL was used as the coding language. The system was debugged and tested for convergence using MATLAB and the Altera SignalTap II Logic Analyzer software. The results indicate that the system converges successfully. The system's converged coefficients show selected ranges of zeros or small values.

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Prof. Mary Kaye and Prof. Brent Petersen for their guidance and for keeping me focused in my research. This thesis would not have been possible without their kind support, trenchant critiques, and remarkable patience.

I would like to thank the administrative staff at the ECE office; Denise Burke, Shelley Cormier and Karen Annett for their kindness and support. My special thanks to Troy Lavigne for providing valuable support and knowledge of FPGA technology and software. I would also like to thank Nagesh Polu for his valuable suggestions and help.

This research was generously funded by the Atlantic Innovation Fund from the Atlantic Canada Opportunities Agency, and by Bell Aliant, our industrial partner. Last but not the least, I would like to thank my parents, Jyotsana and Rajendra Desai; my sister Sapna Desai; Shivani and my other friends for their love and support.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

$\uparrow 4$	Up-sample by Factor 4
$\downarrow 4$	Down Sample by Factor 4
1X1	One-by-One
2X2	Two-by-Two
BER	Bit Error Rate
DS	Direct Sequence
DSP	Digital Signal Processing
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
FSE	Fractionally Spaced Equalizer
HDL	Hardware Description Language
IO	Input Output
LAN	Local Area Networks
LMS	Least Mean Square
LOS	Line of Sight
LRS	Linear Recursive Sequence
LRSG	Linear Recursive Sequence Generator
MAI	Multiple Access Interference
MIMO	Multi-input Multi-output
MLSE	Maximum Likelihood Sequence Estimation

MMSE	Minimum Mean Square Error
MSE	Mean Square Error
MSym/s	Mega Symbols Per Second
NLMS	Normalized Least Mean Square
OFDM	Orthogonal Frequency Division Multiplexing
PN	Pseudo Noise
SISO	Single-input Single-output
S2U	Signed to Unsigned
SMA	SubMiniature Version A
ST	Space Time
SWAP	Signalling Wavelength Antenna Placement
U2S	Unsigned to Signed

Chapter 1

Introduction

With the increasing use of wireless communication for data applications such as Internet access and multimedia, the demand for reliable high-data-rate services is increasing rapidly. Wireless channels introduce a variety of impairments in the transmitted signals due to fading, intermittent interference and multi-user interference [4]. The use of MIMO technology can exploit multi-path propagation to mitigate these impairments. MIMO technology uses multiple forms of diversity by using multiple antennas at the transmitter and the receiver. For applications such as wireless Local Area Networks (LAN) and cellular telephony, it is required that a single base station must communicate with multiple devices. This makes the study of MIMO multi-user systems an important topic of research. With multiple devices communicating with a single base station at the same time, Multiple Access Interference (MAI) exists in a MIMO multi-user system. In order to combat MAI and to identify the users at the receiver, a suitable MIMO multi-user detection technique is required [5, 6]. Several receiver architectures have been proposed for MIMO multi-user wireless systems. For exploiting the advantages of MIMO channels we need a suitable MIMO equalizer. A linear Minimum Mean Square Error (MMSE) receiver using an adaptive algorithm such as LMS or Normalized Least Mean Square (NLMS) can be used

for this purpose [7]. Furthermore, for LOS scenarios the value of MMSE decreases when the separation between the receiver antenna elements is greater than the symbol wavelength [8]. While it is possible to study the performance of MIMO systems in simulations, the results from the simulations may not match those of a practical real-world system. In order to investigate fully the performance of MIMO systems, it is desired to test these systems on a hardware platform. With the vast development in the field of FPGA technology, Digital Signal Processing (DSP) algorithms can be effectively and efficiently implemented in FPGAs. Thus FPGAs provide a suitable hardware platform for implementing wireless communication systems. Also, the use of FPGAs for implementing communication systems with adaptive filtering techniques provides a high-level of overall system performance, resource utilization and low power consumption along with the advantage of reprogrammability [9]. An FPGA-based implementation of a multi-antenna system, exploiting the benefits of separating the antennas on the scale of a symbol wavelength, can help in investigating the benefits of MIMO systems in real-world scenarios. The goal of this thesis is to design and implement on an FPGA, a MIMO system with two users and a receiver with two antenna elements based on symbol-wavelength separation of antenna elements. This thesis will be useful for practical investigation of MIMO systems and for channel equalization and estimation. It also exploits the benefits of the FPGA rapid prototyping platform.

1.1 Background

The objective of this section is to provide a brief understanding of MIMO technology and the FPGA rapid prototyping platform.

1.1.1 MIMO Technology

In recent years, MIMO technology has shown great potential in wireless communication systems [4]. MIMO communication systems have the capability to achieve higher throughput compared to Single-input Single-output (SISO) systems at the same bandwidth and transmit power. Wireless MIMO systems send and receive information over two or more antennas often shared among many users. The signals reflect off of objects in the environment causing multiple paths. In conventional systems, these multi-paths cause interference and fading. However, MIMO systems combine the multiple fading paths and users' signals to overcome multi-user interference and fading, and thereby increase data throughput and reduce Bit Error Rate (BER) as compared to SISO systems. Use of multiple antennas is a diversity technique. There are many other diversity techniques available, using frequency, polarization, time and space. Space diversity is a commonly used and relatively simple technique. It is relatively easy to implement the space diversity techniques by using multi-antennas at the transmitter or the receiver. However, due to space constraints this technique is usually employed at the base station. If multiple antennas are used, multiple channels may exist between the transmitter and receiver [10, 11, 12].

1.1.2 FPGA Technology

An FPGA is a large-scale integrated circuit that can be programmed after it is manufactured rather than being limited to a predetermined unchangeable hardware function. FPGA technology is widely used in wireless communication systems. It combines the speed of dedicated, application-optimized hardware and reprogrammability of microprocessors which makes it suitable for high speed implementation of adaptive filters [13]. A programmable FPGA consists of the following elements:

Programmable Logic Cells which provide the functional elements for construction of the user's logic,

Programmable Input Output (IO) Blocks which provide the interface between the logic cells and the package pins, and

Programmable Interconnects which provide the routing paths to connect the input and output of logic cells and the IO blocks.

Modern FPGAs provide high-level arithmetic and control structures, such as multipliers, counters, multiply accumulate units, memory resources and processor cores. These resources provide high performance, low power consumption and are highly suitable for applications such as DSP [14].

The behavior of an FPGA can be defined by using a hardware description language (HDL) such as VHDL or Verilog or by arranging blocks of existing functions using a schematic-oriented design tool. The design is compiled and synthesized using proprietary FPGA place-and-route tools. The compilation and synthesis process generates a bit file which can then be downloaded on the FPGA [15].

1.2 Literature Review

Wireless communication systems with multiple inputs and multiple outputs in which many antennas are used for transmission and reception provide spatial diversity, and improve the performance of wireless systems by mitigating the effects of multipath fading [12]. Yanikomeroglu et al. [16] proposed that in order to achieve increased antenna gain against interference in addition to the diversity gain, the separation of the antenna elements in a spread spectrum system must be many times greater than the chiplength = [speed of light] / [chip rate] of the spreading code. Zhu et al. [8] introduced a new constraint with respect to the signalling wavelength for

the separation of antenna elements. For Direct Sequence (DS) systems this is equal to the chiplength and for non-spread systems it is equal to the symbol wavelength = [speed of light] / [symbol rate]. He showed that this constraint improves the performance of multi-antenna systems and named this improvement the Signalling Wavelength Antenna Placement (SWAP) gain. Polu et al. [17] investigated the measurements for the MMSE gain in a multi-antenna system. They concluded that the MMSE decreases when the separation between the antenna elements is increased and if the antenna separation is increased more than a symbol wavelength there is no change in the MMSE. Harriman [18] demonstrated a software-defined radio implementation of a four-channel transceiver testbed with signalling-wavelength-spaced antennas. He suggested the use of an LMS-algorithm-based receiver to provide user detection by channel equalization and thereby reduce distortion and compensate for the impairments in the channels.

In order to exploit the benefits of multi-antenna systems, a suitable MIMO linear combiner or equalizer with an LMS-algorithm-based receiver is needed [7]. Atiniramit [9] implemented a single-user adaptive filter receiver on an FPGA-based configurable computing platform called Giga Ops G900. He used an LMS algorithm as the adaptive filtering algorithm to alleviate multiple access interference and the near-far problem. Although the NLMS algorithm has fast convergence, he suggested the use of the LMS algorithm as the NLMS algorithm requires a division operation and its steady state Mean Square Error (MSE) is higher. He further noted that the NLMS algorithm does not converge faster for small step size and both algorithms display similar behavior. He also pointed out the practical disadvantages of using DSP processors and suggested that the use of an FPGA-based computing platform can increase throughput and reduce output latency.

Lin [3] presented a comparison between DSP processors and FPGAs for implementing adaptive-filter-based receivers. He suggested the use of FPGAs over DSP

processors for implementing adaptive filtering algorithms. He also showed that as the filter order increases, the performance of an FPGA-based system deteriorates due to increase in the longest register-to-register delay. To overcome this effect, he suggested the use of pipelining techniques.

Thus far there has been much theoretical research in the field of MIMO systems. However, relatively few practical systems have been demonstrated [19]. Christian et al. [20] demonstrated a scalable rapid prototyping system for real-time MIMO Orthogonal Frequency Division Multiplexing (OFDM) transmissions. However, their implemented system uses a combination of DSP processors and FPGAs. Also, it does not incorporate the benefits of SWAP gain. Thus, a MIMO system exploiting the advantages of SWAP gain and the rapid prototyping capabilities of FPGAs has been implemented in this thesis.

1.3 Thesis Contributions

A real-time MIMO multi-user system with two users and one receiver with two antenna elements has been implemented on an FPGA. The system incorporates the benefits of separating the antenna elements on a scale of symbol wavelength and thereby provides an FPGA based real-time system to investigate the effects of SWAP gain in a real-world environment. The system includes an ST receiver, which distinguishes amongst the users and provides diversity gain. It also minimizes MAI and other impairments introduced by the channels. Verification and testing of the system has been done to ensure convergence of the system. The pathological case shown by Zhu [8] has been investigated and the failure of a 2X2 system in a pathological case has also been confirmed. A MATLAB simulation has been developed for debugging and testing purposes. This MATLAB simulation can be further modified and used for expansion of the system.

This system provides a hardware platform to investigate the performance of MIMO systems in real time. With the channel equalization capability of the system, it identifies possibilities for receiver complexity reduction. The hardware utilized for implementing the adaptive transversal filters in the system can be reduced by placing the coefficients only where needed and setting the rest of the coefficients to zero. The system was designed and implemented incrementally. The components of the implementation can be replicated to expand the system to incorporate more users and receiving antennas and to increase the size of the Finite Impulse Response (FIR) filters using more FPGA boards. A pipelined architecture is used for the multipliers. The system can be incorporated with Harriman's [18] work by replacing the simulated channels with the radio channel testbed developed by Harriman to provide a complete FPGA-based MIMO system testbed exploiting the benefits of SWAP gain.

1.4 Thesis Structure

- Chapter 2 gives a brief explanation of the system model and its components. Details of the LMS algorithm used in the ST receiver are also covered in this chapter.

- Chapter 3 describes the implementation specific details of the system. A detailed explanation of the implementation of various components of the system is covered. This chapter also gives the description of the MATLAB simulation used for comparison and debugging and illustrates the results of the simulation. Specifications of the Altera Stratix EP1S80 Board used for implementation are also listed in this chapter.

- Chapter 4 describes the debugging and verification process adapted in this thesis and illustrates the results of the implementation. It also gives a brief

explanation of the various issues encountered during the course of this thesis and the describes the steps taken to resolve these issues.

- Chapter 5 summarizes the work completed and possible research in the future.

Chapter 2

System Model

The objective of this chapter is to give a brief explanation of the implemented system model and its components. Figure 2.1 shows a block diagram of the implemented 2X2 system. The main components of the system are the transmitters, simulated channels and the receiver. A multi-antenna system with two users and two receiving antenna elements is considered. The data is generated and transmitted. This data also serves as the training sequences for the adaptive ST receiver. Coaxial cables are used as the communication link between transmitter and receiver. Pure-delay LOS channels are considered in this thesis. These channels are simulated on the FPGA itself. The receiver includes a linear combiner and uses the LMS algorithm and transversal filters to minimize the MSE. The entire system is implemented on a single FPGA board.

2.1 Transmitter

In order to design a 2X2 communication system, it is necessary to design two transmitters which will model the two information signals to transmit. These signals serve as the training data or the desired signals which are known to the receiver. These two transmitters represent two users that will continuously generate

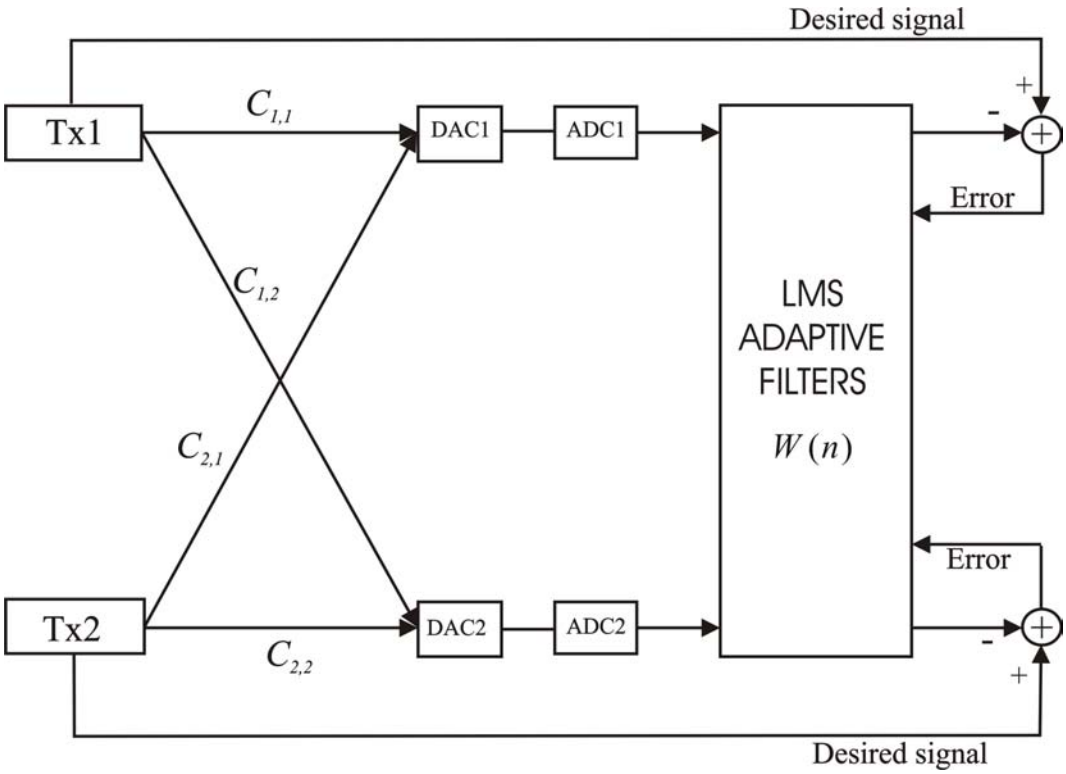

Figure 2.1: Block diagram of implemented system

and transmit the training data to the receiver. The generation and transmission of the data is done simultaneously. The main focus of this thesis is to investigate the implementation of a MIMO multiuser system for symbol-wavelength-spaced antennas. The separation of the antenna elements on a scale of symbol wavelength, instead of carrier wavelengths, in LOS channels increases the performance of the multi-antenna multi-user system [17, 8]. A symbol rate of 5 Mega-symbols per second (MSym/s) was selected. The bit rate is also the symbol rate since there is one bit per symbol. Thus, the symbol wavelength, λ_g, can be defined as

$$\begin{aligned} \lambda_g &= \frac{c}{f_g} \\ &= \frac{3 \times 10^8}{5 \times 10^6} \\ &= 60 \text{ m}, \end{aligned} \tag{2.1}$$

where c is the velocity of light and f_g is the symbol rate.

In order to take advantage of the SWAP gain, the antenna elements have to be spaced at a distance equal to or greater than this symbol wavelength. The generation of the data is done in the FPGA itself. Two Linear Recursive Sequences (LRS) of length 4095 were used as the transmitted signals to facilitate testing and debugging.

2.2 Channel Model

For a MIMO multiuser model, multiple channels exist by the use of multiple antennas. Figure 2.2 shows the possible MIMO channels for M users each with a single transmitting antenna and one receiver with L receiving antennas.

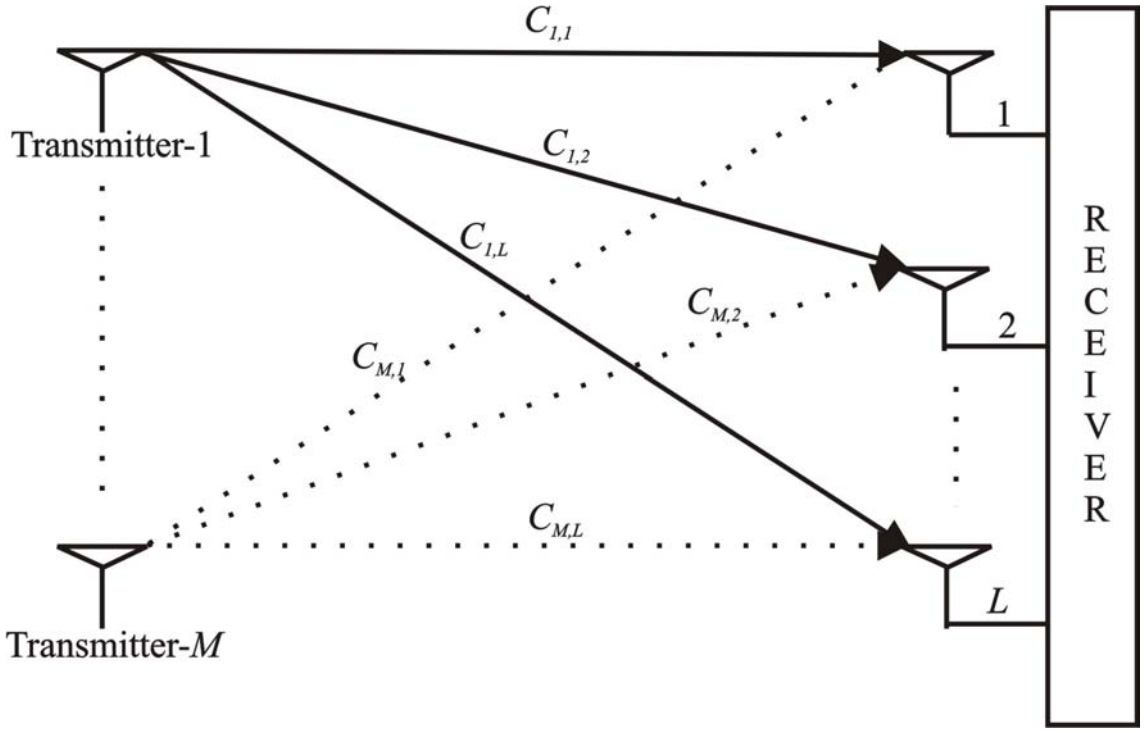

Figure 2.2: MIMO channels

LOS channels for a MIMO system with two users and one receiver with two

antenna elements were considered in this thesis. In order to effectively improve the performance of a multi-antenna system, the antenna elements must be separated properly. Recent research shows that the separation can be on the scale of a symbol wavelength [8]. Increasing separation between the antenna elements results in significant performance improvement until a maximum separation of one symbol wavelength, after which there is negligible improvement with increasing separation [17]. Figure 2.3 shows the top view of the communication model between two users and one receiver with two antenna elements, which are separated by one symbol wavelength. This case will be referred as the general case in this thesis. The pathological case illustrated by Zhu [8] is shown in the Figure 2.4. With a symbol rate of 5 MSym/s, the separation, d, between the antenna elements at the receiver would be equal to the symbol wavelength λ_g,

$$\begin{aligned} d &= \lambda_g \\ &= 60 \text{ m}. \end{aligned} \tag{2.2}$$

LOS channels can easily be simulated by using FIR filters. Since pure-delay LOS channels have no multi-path, all the coefficients of these FIR filters are zero, except for one coefficient and its value is equal to one. Thus, the signal transmitted by a user reaches the receiving antenna with a delay based on its distance from the receiving antenna. The delays can be calculated for each channel according to the geometry in Figures 2.3 and 2.4. Tables 2.1 and 2.2 give the calculated delays for the general case and the pathological case respectively.

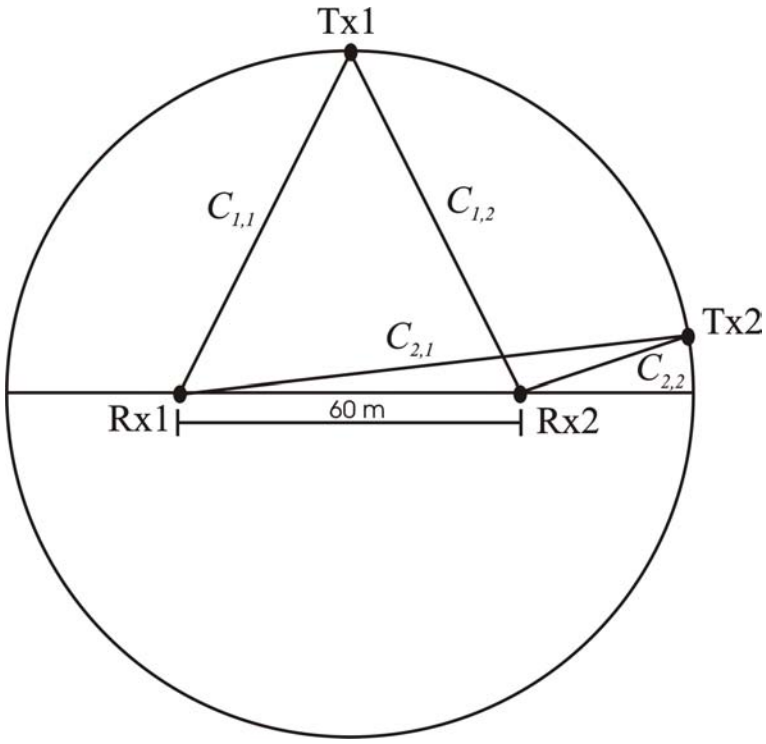

Figure 2.3: Communication model between two users and one receiver with two antenna elements – General case

Table 2.1: Delays for MIMO Channels - General Case

MIMO channel	Delay
$C_{1,1}$	224 ns
$C_{1,2}$	224 ns
$C_{2,2}$	103 ns
$C_{2,1}$	299 ns

Table 2.2: Delays for MIMO Channels - Pathological Case

MIMO channel	Delay
$C_{1,1}$	280 ns
$C_{1,2}$	147 ns
$C_{2,2}$	147 ns
$C_{2,1}$	280 ns

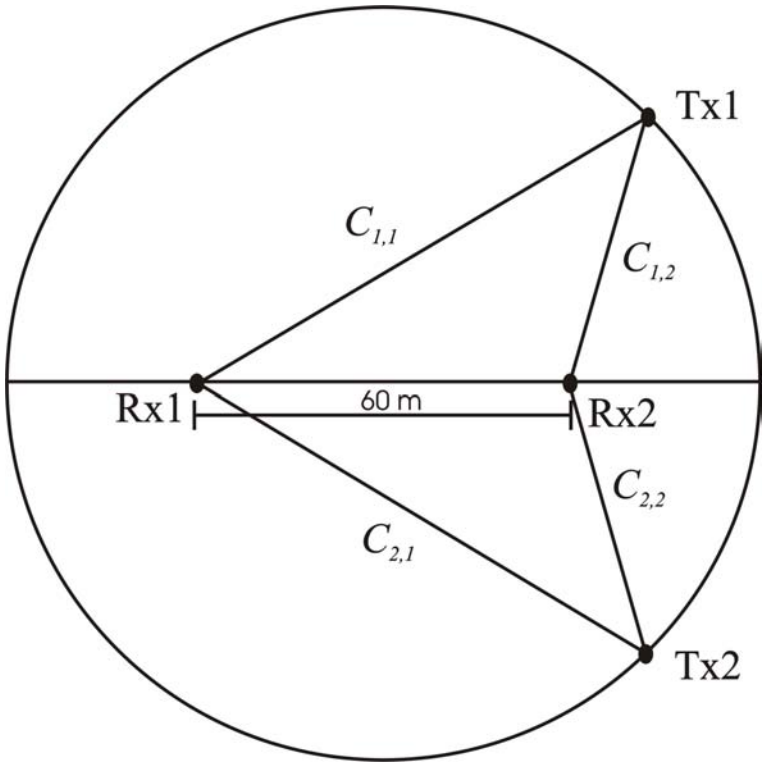

Figure 2.4: Communication model between two users and one receiver with two antenna elements – Pathological case

2.3 Receiver

For known channels, MIMO multi-user receivers are mainly divided into two categories: the Optimal Maximum Likelihood Sequence Estimation (MLSE) receiver and sub-optimal receivers [21, 22]. The optimal receiver is relatively complex and expensive to implement [23, 24, 25]. A sub-optimal receiver is less complex. There are two types of sub-optimal receivers, linear and non-linear sub-optimal. A linear sub-optimal MMSE-based receiver, also known as a linear combiner, that suppresses cross-channel interference and MAI interference is used in this thesis. This ST receiver may use an LMS adaptive algorithm to optimize the coefficients to equalize the channel, reduce interference, and recover each user's transmission. Since there are two users and two antennas at the receiver, the receiver consists of two sets of filters. Each set of filters has two adaptive filters. One particular set of filters separates one

user in the system. The outputs of the adaptive filters in one set are then combined to average the signals from both the antennas. This averaged signal is used to calculate the MSE. Figure 2.5 shows an ST receiver with M users and L receiving antenna elements and Figure 2.6 shows the detailed structure of the transversal filter.

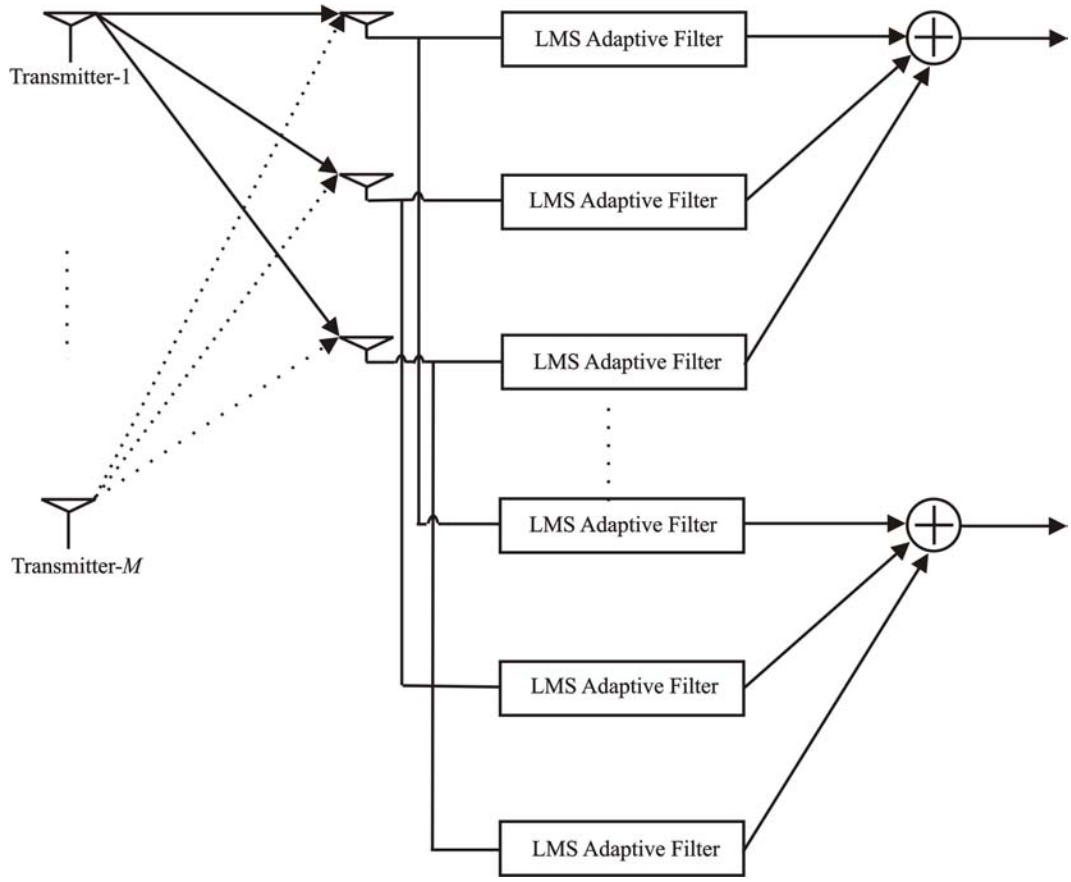

Figure 2.5: Block diagram of ST receiver for M users

The error signal is obtained by comparing the signal at the output of each set of adaptive filters and the training sequence, which is the known transmitted data. The coefficients of the transversal filters are updated to minimize this MSE. This can be achieved by three different techniques. They are, taking the gradient, completing the square and statistical orthogonality. When using the gradient method, the gradient of the MSE with respect to the filter coefficients is taken and equated to zero to obtain a global minimum. This is called the Wiener solution. Alternative

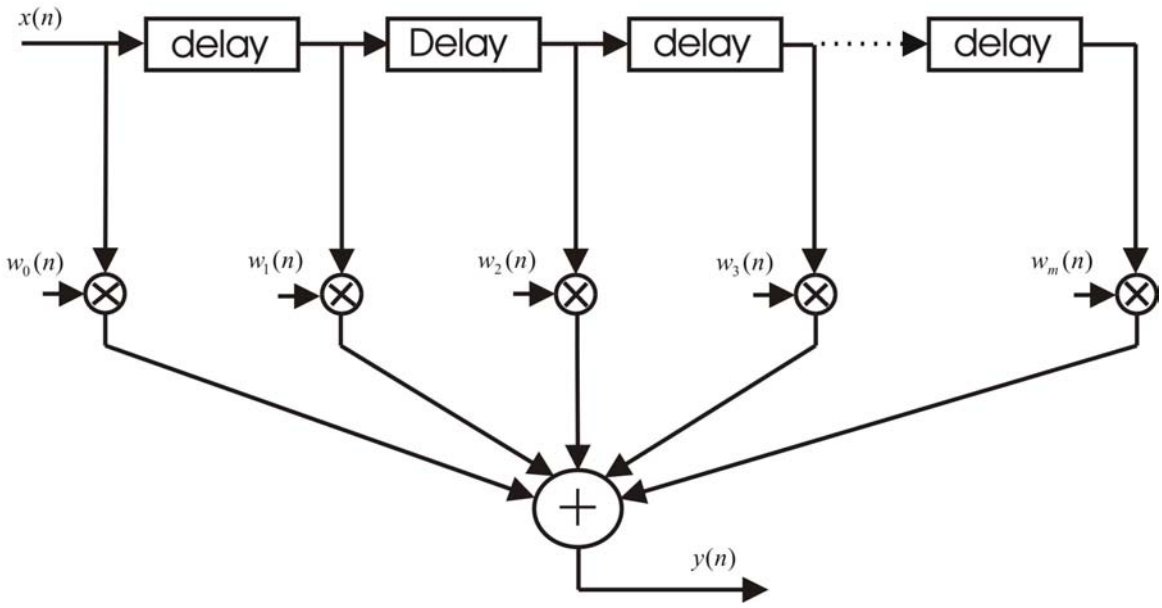

Figure 2.6: Detailed structure of transversal filter

methods to the Wiener solution are the steepest decent algorithm and stochastic gradient algorithm, also known as the LMS algorithm. In this thesis we use the LMS algorithm.

2.3.1 LMS Algorithm

The adaptive LMS algorithm, also known as Widrow-Hoff Learning Algorithm, is one of the most widely used adaptive filtering algorithms. This algorithm is based on an approximation of the steepest decent procedure. In channel equalization, the LMS algorithm adapts the filter coefficients of an FIR filter driven by the received signal. The algorithm updates the coefficients of the filter to minimize the MSE. The error signal is formed by the difference of a training signal and the output of the filter [26, 27]. The LMS algorithm is simple and requires little computation. It updates the tap weights every sample so it continuously adapts the filter. Also, it tracks well slow changes in the signal strength which makes it suitable for adaptive filtering applications [28]. Figure 2.7 shows a block diagram of an adaptive filter

system using the LMS algorithm.

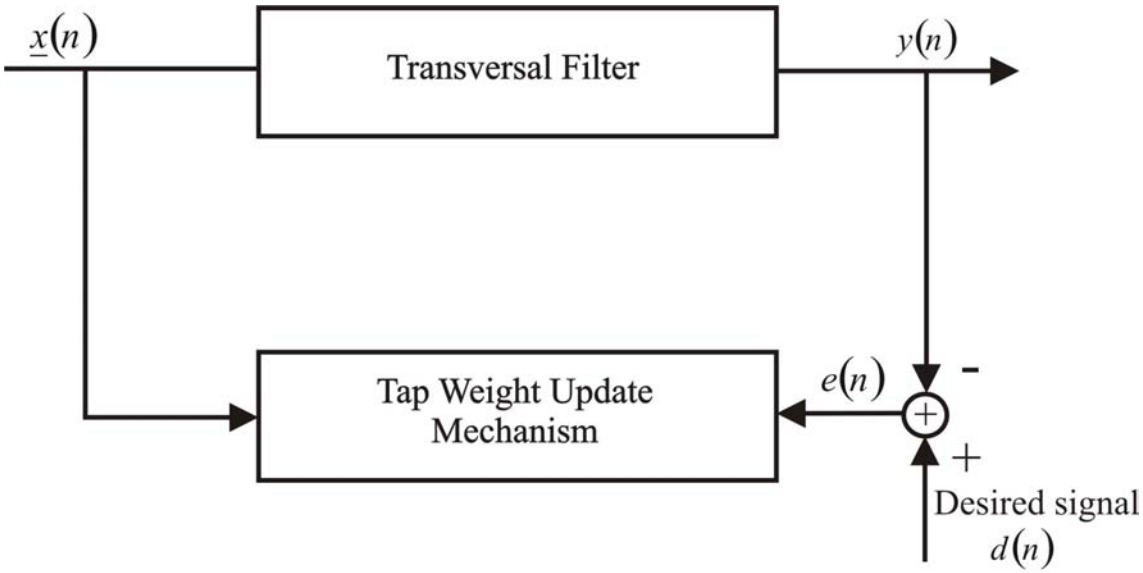

Figure 2.7: Block diagram of LMS algorithm

Assuming that $x(n), x(n-1), x(n-2), \ldots, x(n-N+1)$ forms the receiver input vector $\underline{x}(n)$ where N is the number of taps in the FIR filter, n is the time index and $w_0, w_1, w_2, \ldots, w_{N-1}$ forms the tap weight vector $\underline{w}$, then

$$y(n) = \underline{w}^H(n)\,\underline{x}(n) \tag{2.3}$$

$$e(n) = d(n) - y(n) \tag{2.4}$$

$$R = E[\underline{x}(n)\,\underline{x}^H(n)] \tag{2.5}$$

$$\underline{p} = E[\underline{x}(n)\,d(n)] \tag{2.6}$$

where H denotes the Hermitian transpose, $y(n)$ is the output of the filter, $d(n)$ is the desired output, R is the autocorrelation matrix of the filter input, $\underline{p}$ is the cross-correlation vector betwen $\underline{x}(n)$ and $d(n)$, and $e(n)$ is the error given by the difference between the desired output and the output of the filter which is used to update the tap weights.

The MSE performance function ξ is given by,

$$\begin{aligned} \xi &= E[e^2(n)] \\ &= E[d^2(n)] - 2\underline{w}^H \underline{p} + \underline{w}^H R \underline{w}. \end{aligned} \tag{2.7}$$

The performance function ξ is a quadratic function of the tap-weight vector $\underline{w}$. ξ has a single global minimum obtained by solving the Wiener-Hopf equation

$$R\,\underline{w} = \underline{p}\,, \tag{2.8}$$

if R and $\underline{p}$ are available. An iterative method can be employed for solving equation 2.8 starting with an initial value for $\underline{w}$, say $\underline{w}(0) = w_0(0), w_1(0), w_2(0), \ldots, w_{N-1}(0)$.

The gradient of ξ is given by

$$\nabla\,\xi = 2\,R\,\underline{w} - 2\,\underline{p}\,. \tag{2.9}$$

With an initial value of $\underline{w}(0)$ at $n = 0$, the tap-weight vector at the k-th iteration can be denoted as $\underline{w}(k)$. This tap-weight vector can then be updated by the steepest descent algorithm equation,

$$\underline{w}(k+1) = \underline{w}(k) - \frac{\mu}{2}\,\nabla_k \xi\,, \tag{2.10}$$

where $\mu > 0$ is called the step size and $\nabla_k \xi$ denotes the gradient vector $\nabla\,\xi$ evaluated at the point $\underline{w} = \underline{w}(k)$.

However, in practice the performance function ξ is difficult to compute and is approximated to its instantaneous estimate $\hat{\xi} = e^2(n)$. From Figure 2.6 the output $y(n)$ of the transversal filter can be given as

$$y(n) = \sum_{i=0}^{N-1} w_i(n)\, x(n-i)\,. \tag{2.11}$$

Substituting the value of $\hat{\xi}$ in equation 2.10, we get

$$\underline{w}(n+1) = \underline{w}(n) - \frac{\mu}{2}\nabla\, e^2(n) \tag{2.12}$$

where $\underline{w}(n) = w_0(n), w_1(n), w_2(n), \ldots, w_{N-1}(n)$ and

$$\nabla = \Big[\frac{\partial}{\partial w_0} \quad \frac{\partial}{\partial w_1} \quad \ldots \quad \frac{\partial}{\partial w_{N-1}}\Big].$$

The i-th element of the gradient vector $\nabla e^2(n)$ is

$$\begin{aligned}\frac{\partial e^2(n)}{\partial w_i} &= 2\, e(n)\, \frac{\partial e(n)}{\partial w_i} \\ &= -2\, e(n)\, \frac{\partial y(n)}{\partial w_i} \\ &= -2\, e(n)\, x(n-i).\end{aligned}$$

Then $\nabla e^2(n) = -2\, e(n)\, \underline{x}(n)$. Substituting the value of $\nabla e^2(n)$ in equation 2.12, we get the equation for LMS algorithm as

$$\underline{w}(n+1) = \underline{w}(n) + \mu\, e(n)\, \underline{x}(n)\ [27]. \tag{2.13}$$

The step-size μ has to be chosen properly for the system to converge. If μ is very small, the system convergence rate is slow, but the MSE obtained after convergence is smaller. However, if μ is large, the rate of convergence is fast, but the MSE obtained after convergence is high. If μ is too high, the system is unstable. Thus, the value of μ must be chosen such that the convergence rate is not too fast and not too slow. At the same time the MSE obtained after convergence should be acceptable.

2.4 Fractionally Spaced Equalizer

The receiver structure discussed thus far is a symbol-spaced equalizer. In a symbol-spaced equalizer, the received signal and the receiver filter taps are spaced at the symbol period T. However, this equalizer is sensitive to channel delay distortion and sampling delay. On the other hand, an equalizer with a tap spacing smaller than T gives better system performance [29, 30]. Such an equalizer is called a Fractionally Spaced Equalizer (FSE). In this thesis we use an FSE. The operation of an FSE is similar to T-spaced equalizer. However, the tap spacing is reduced to KT/N where K and N are integers and $N > K$. To implement an FSE-based receiver, it is necessary to up-sample the data before it is fed to the input of the receiver. The adaptive filter coefficients will be spaced according to this new data rate. The output of the FSE must then be down sampled to the symbol rate before the error can be calculated.

Chapter 3

System Implementation

This chapter describes the implementation of the system. As discussed in Chapter 2 there are three main parts of the system, the transmit side, channel model and the receive side. The system was implemented on an Altera Stratix EP1S80 DSP development board and was developed using Altera Quartus II software. The HDL coding was done in Verilog. The system was debugged using Altera Signaltap II Logic Analyzer software and MATLAB.

3.1 MATLAB Simulation

In order to investigate the feasibility of the system, a MATLAB simulation was developed. The simulation uses two transmitters, which are simulated using two 31-bit long LRSs. The details of the LRSs will be discussed further in this chapter. It is necessary to use LRS generators to create Pseudo Noise (PN) sequences. This is done in order that the same sequences can be used in the implementation and the MATLAB simulation, to facilitate debugging the system. These PN sequences also represent the predetermined training sequences for the receiver. Based on the geometry shown in Figure 2.3 the MIMO channels were simulated using simple FIR filters. As mentioned in Chapter 2, for pure-delay LOS channels, the coefficients of

these FIR filters were all set to zero, except for one coefficient which was set to one. This coefficient was selected according to the delays calculated from the geometry in Figures 2.3 and 2.4 in Chapter 2. Tables 2.1 and 2.2 from Chapter 2 list the values of these delays for the general case and pathological case, respectively.

The receiver includes two sets of filters, each of which have two adaptive transversal filters and their coefficient update mechanism based on the LMS algorithm. A few modifications were required in the MATLAB simulation to match it to the FPGA implementation. This was done by modifying the output of the transmit side and the input to the receive side to match the data in the FPGA. Figure 3.1 and 3.2 shows the simulation results. From the simulation results it is observed that the system converges successfully in the general case. The coefficients of the FIR filters compensate for the LOS channels. From Figure 3.2, due to the high MSE, it is clear that the system fails to converge in the pathological case.

3.2 FPGA Development Board Features

The system was implemented on an Altera Stratix EP1S80 DSP development board. This board features the Altera EP1S80 chip. Table 3.1 gives the key specifications of this chip. The board has two 12-bit 125-MHz ADCs and two 14-bit 165-MHz DACs, which were used in implementing the system. These ADCs and DACs have SubMiniature Version A (SMA) connectors. External signals can be fed to the ADCs and internally generated signals can be put out through the DACs. The communication between the ADCs and the DACs was carried out using coxial cables. Figure 3.3 shows the Altera EP1S80 DSP development board.

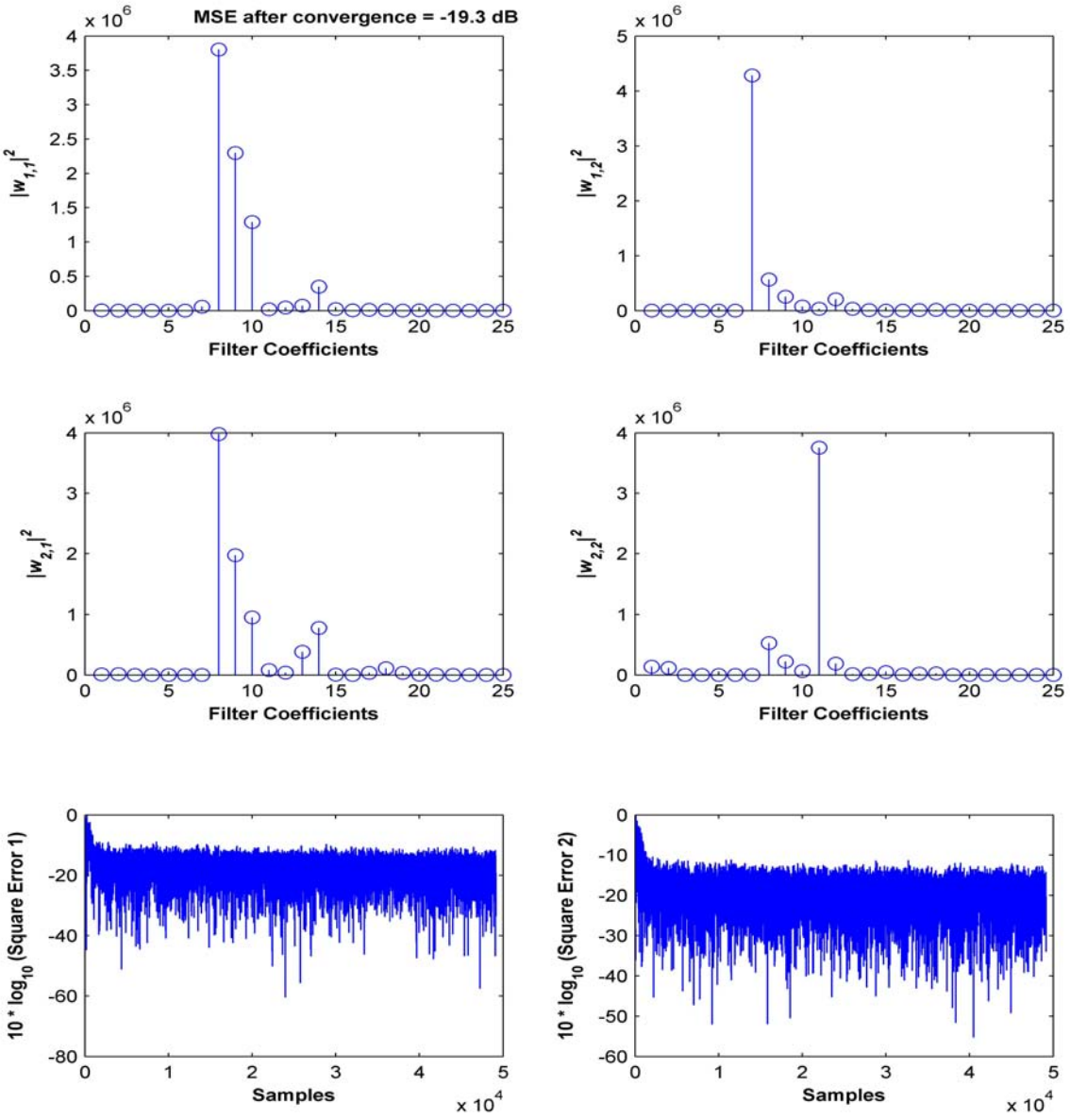

Figure 3.1: Linear combiner filter coefficients and the MSE learning curves obtained from MATLAB simulation – General case

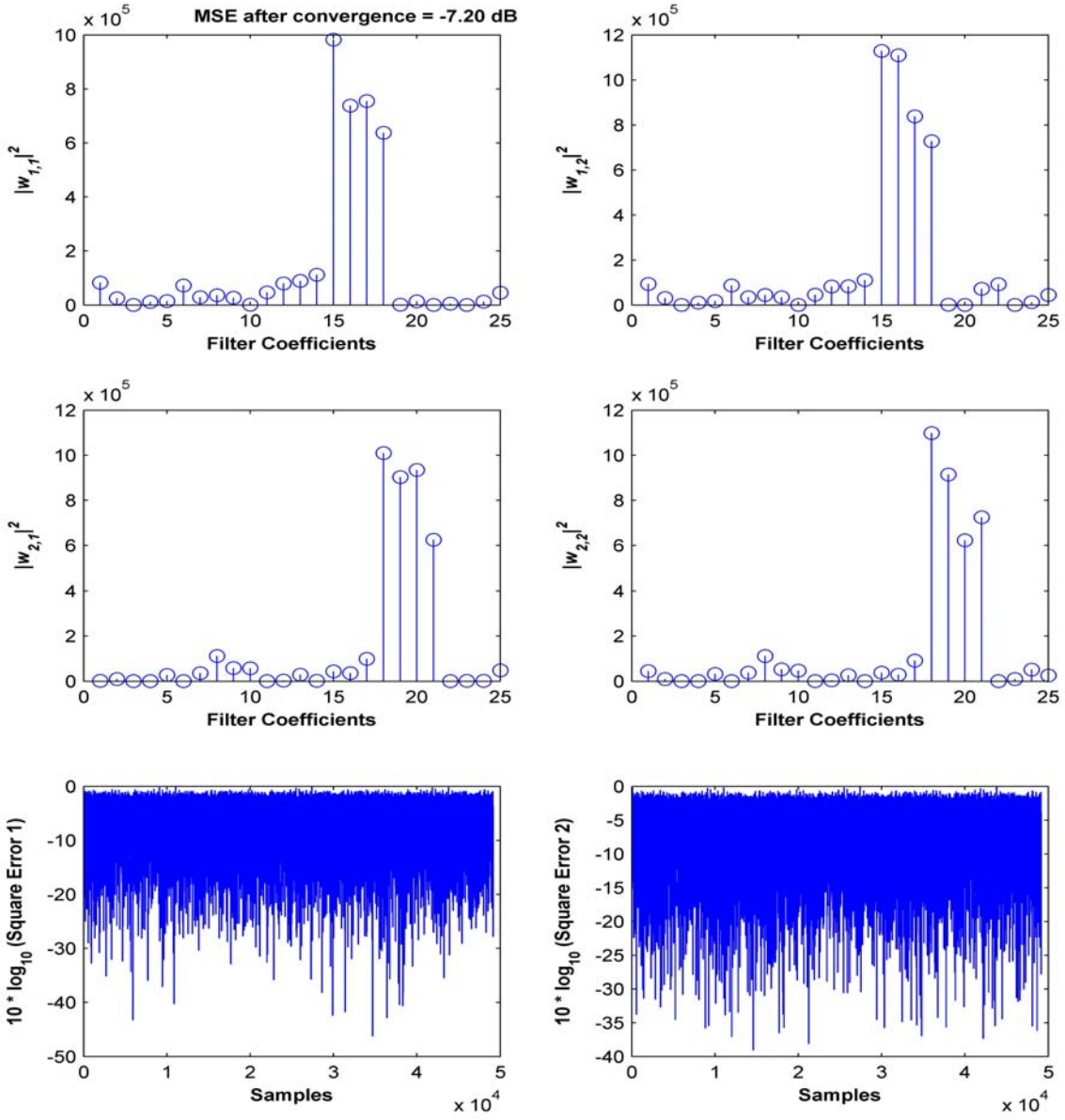

Figure 3.2: Linear combiner filter coefficients and the MSE learning curves obtained from MATLAB simulation – Pathological case

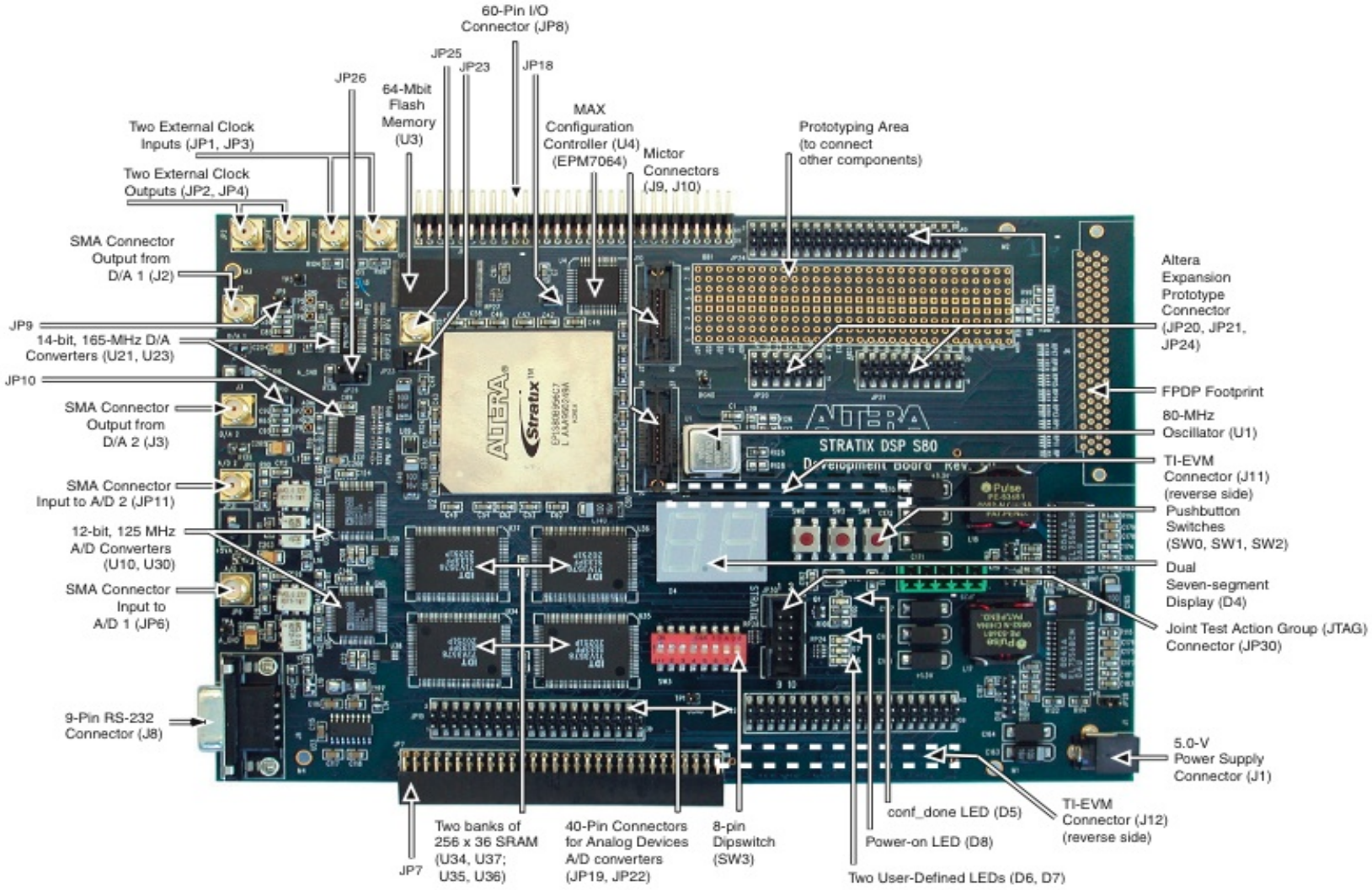

Figure 3.3: EP1S80 DSP development board [1]

Table 3.1: Key Features for Altera EP1S80 Chip [1]

Feature	EP1S80B95676
Logic Elements	79,040
M512 RAM Blocks (32 x 18 bits)	767
M4K RAM Blocks (128 x 36)	364
M-RAM Blocks	9
Total RAM bits	7,427,520
DSP Blocks	22
Embedded Multipliers (based on 9 x 9)	176
PLLs	12
Maximum user I/O pins	679
Package Type	956-pin BGA
Board Reference	U1
Voltage	1.5-V internal, 3.3-V I/O

3.3 Generation of Data for the Two Users

As discussed earlier, the first stage of developing the system was to generate the signals for transmission. The signals are generated using LRS generators. Two PN sequences are used to represent the training sequences for the two users. The LRS generators can be easily created using a shift register circuit and exclusive-OR gates. These sequences appear random in the short term. However, the output of any digital sequential circuit is deterministic. A shift register with N-stages can only take 2^N different states. Therefore these sequences are periodic and repeat at predefined intervals. In the ideal PN generator, the sequence is 2^N-1 bits long, where N is the number of flip flops. This is called maximal length sequence. A maximal length sequence can only be created by using a primitive prime polynomial [2]. For a polynomial $P(x)$ to give a maximal length sequence of length L, it must be a factor of $X^L + 1$ (and of no other smaller L). Such a prime polynomial is called a primitive prime polynomial. To ensure that the ST receiver would have enough time to converge, a polynomial of order 12 was used to give a 4095-bit sequence. Table 3.2 lists some of the polynomials giving maximal length sequences of 4095-

bits. A complete list of primitive prime polynomials which generate maximal length sequences can be found in work by Peterson et al. [2].

Table 3.2: Primitive Prime Generating Polynomials [2]

Polynomial	Tap Configuration
$x^{12}+x^6+x^4+x+1$	(12,6,4,1,0)
$x^{12}+x^{11}+x^9+x^8+x^7+x^5+x^2+x+1$	(12,11,9,8,7,5,2,1,0)
$x^{12}+x^{11}+x^{10}+x^8+x^6+x^4+x^3+x+1$	(12,11,10,8,6,4,3,1,0)
$x^{12}+x^{11}+x^{10}+x^5+x^2+x+1$	(12,11,10,5,2,1,0)
$x^{12}+x^9+x^3+x^2+1$	(12,9,3,2,0)
$x^{12}+x^{11}+x^6+x^4+x^2+x+1$	(12,11,6,4,2,1,0)

Polynomials $x^{12}+x^6+x^4+x+1$ and $x^{12}+x^{11}+x^9+x^8+x^7+x^5+x^2+x+1$ were used to generate the two training sequences in this thesis. Figures 3.4 and 3.5 show the circuit diagrams of LRS generators for these polynomials.

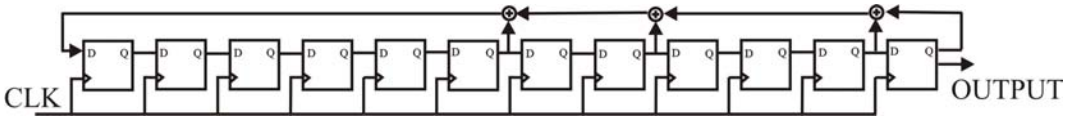

Figure 3.4: Circuit diagram for polynomial $x^{12}+x^6+x^4+x+1$

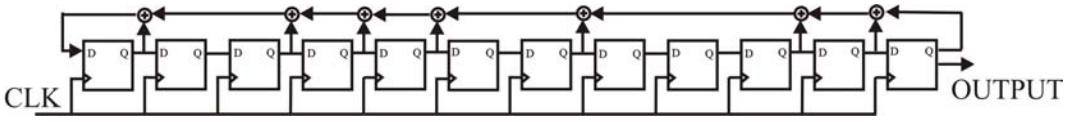

Figure 3.5: Circuit diagram for polynomial $x^{12}+x^{11}+x^9+x^8+x^7+x^5+x^2+x+1$

As discussed in Chapter 2, the data rate was chosen to be 5 MSym/s. In this thesis, an FSE is used in the receiver. Thus, the generated data is up-sampled to 20 MHz. This also increases the processing speed of the entire circuit. Since the data occupies 5 MHz bandwidth, it still satisfies the Nyquist's criterion. The FPGA board has an 80 MHz onboard oscillator. In order to generate the 5 MHz and 20 MHz clocks, a clock divider circuit was used. This circuit includes two separate

counters. These counters provide a division operation on the 80 MHz clock signal to give 5 MHz and 20 MHz clocks.

3.4 Simulation of LOS Channels on FPGA

Based on the geometry in Figures 2.3 and 2.4 the MIMO pure-delay LOS channels can be modeled according to Table 2.1. Four FIR filters are used to implement these channels. These FIR filters delay the input by the period specified in Tables 2.1 and 2.2. This can be implemented easily by using FIR filters. Since all the coefficients of the FIR filters are zero except for one and its value is one, the channels can be implemented using only a series of delays as shown in the Figure 3.6. This way we can avoid the use of extra multipliers.

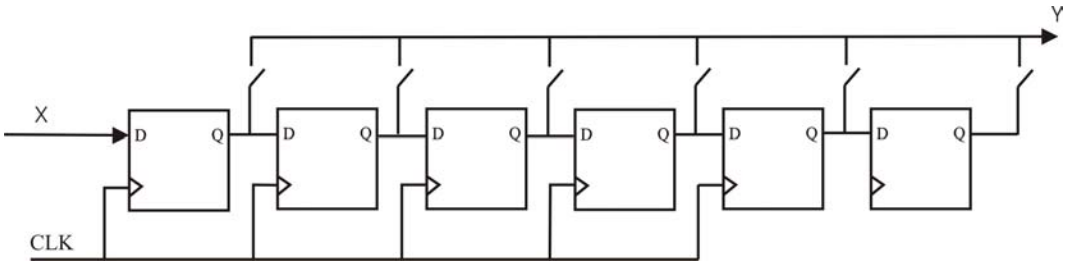

Figure 3.6: Generalized circuit diagram for channel model

Figure 3.7 shows a detailed diagram of the implemented transmitter side and the simulated LOS channels. The data is generated by the two Linear Recursive Sequence Generators (LRSG) at the rate of 5 MHz. The outputs of the LRSGs are one-bit unsigned binary numbers. In order to perform arithmetic operations using signed binary numbers, the one-bit unsigned outputs of the LRSGs are converted into three-bit signed binary numbers using the Unsigned to Signed (U2S) blocks. To accommodate for the FSE the data is then up-sampled to 20 MHz using the Up-sample by Factor 4 ($\uparrow 4$) blocks. This up-sampled data is then fed to the simulated LOS MIMO channels. The outputs of the LOS MIMO channels are the final data

for transmission via the DACs. Signals $S_1(m), S_2(m), S_3(n), P_{1,1}(n), P_{2,1}$ and $S_4(n)$ in Figure 3.8 show the sample signals at various stages of the transmitter. Since the DACs expect 14-bit unsigned binary data, the transmission data is converted from 3-bit signed binary numbers to 14-bit unsigned binary numbers using the Signed to Unsigned (S2U) blocks. The outputs of the DACs are analog signals which are transmitted to the receiver side via coaxial cables. These analog signals are external to the FPGA. The desired data, which is the known data for the receiver, is provided to the receiver side internally within the FPGA.

3.5 Adaptive Linear Combiner ST Receiver Implementation

The ST receiver consists of a linear combiner circuit. The data is received at the ADCs in 12-bit signed binary format. These ADCs are clocked at 80 MHz. The received data thus has to be down sampled to reduce the data rate back to 20 MHz using the Down Sample by Factor 4 ($\downarrow 4$) blocks. Figure 3.9 shows a detailed diagram of the implemented receiver side. The 20 MHz data is fed to the linear combiner circuit. The linear combiner used in this system is based on an FSE. Thus the output of the linear combiner circuit has to be down sampled by a factor of four to reduce the data rate from 20 MHz to 5 MHz. The error signal is then calculated as the difference between the output of the linear combiner circuit and the desired signal.

The adaptive transversal filters in the linear combiner circuit assume an FIR filter structure as shown in Figure 3.10. The main components of the filters consist of unit-delay registers and weight updates. These filters use 25 coefficient taps each. The unit-delay registers are simply D flip flops. The weight update components modify the filter coefficients according to the LMS equation presented in Chapter 2, Equation 2.13. Figure 3.11 shows a detailed diagram of a single coefficient tap with

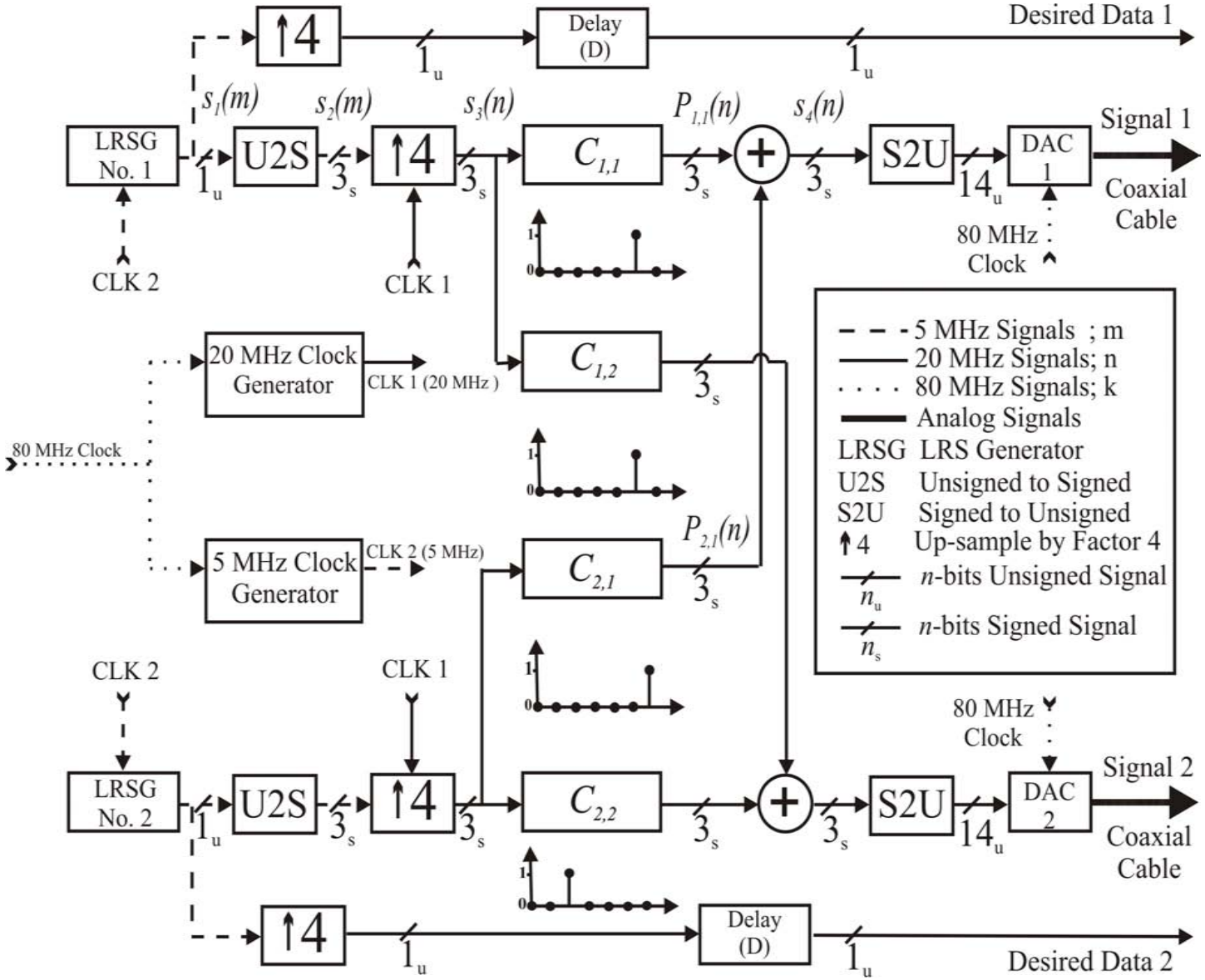

Figure 3.7: Detailed diagram of implemented transmitter side

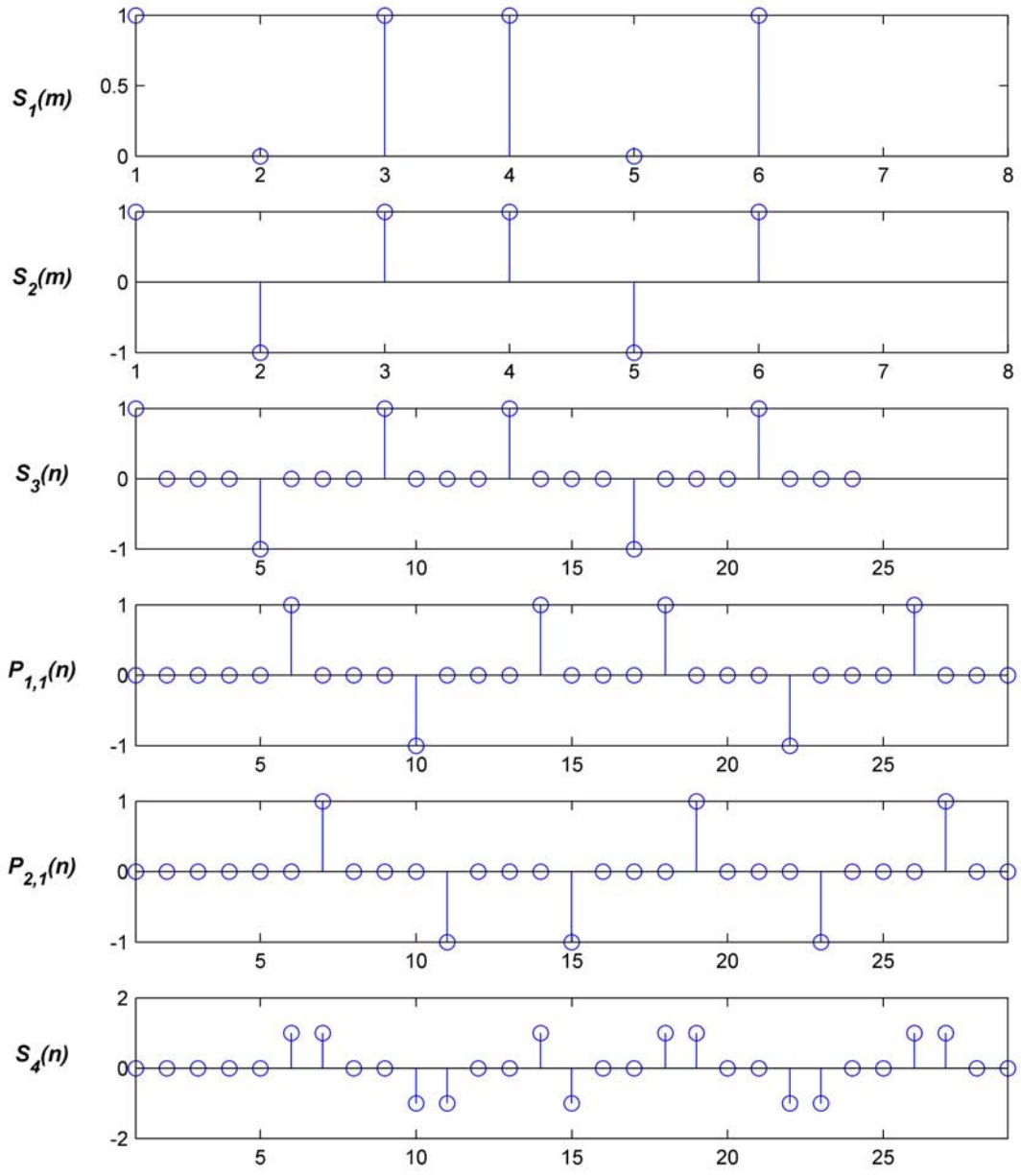

Figure 3.8: Sample signals from the transmitter side

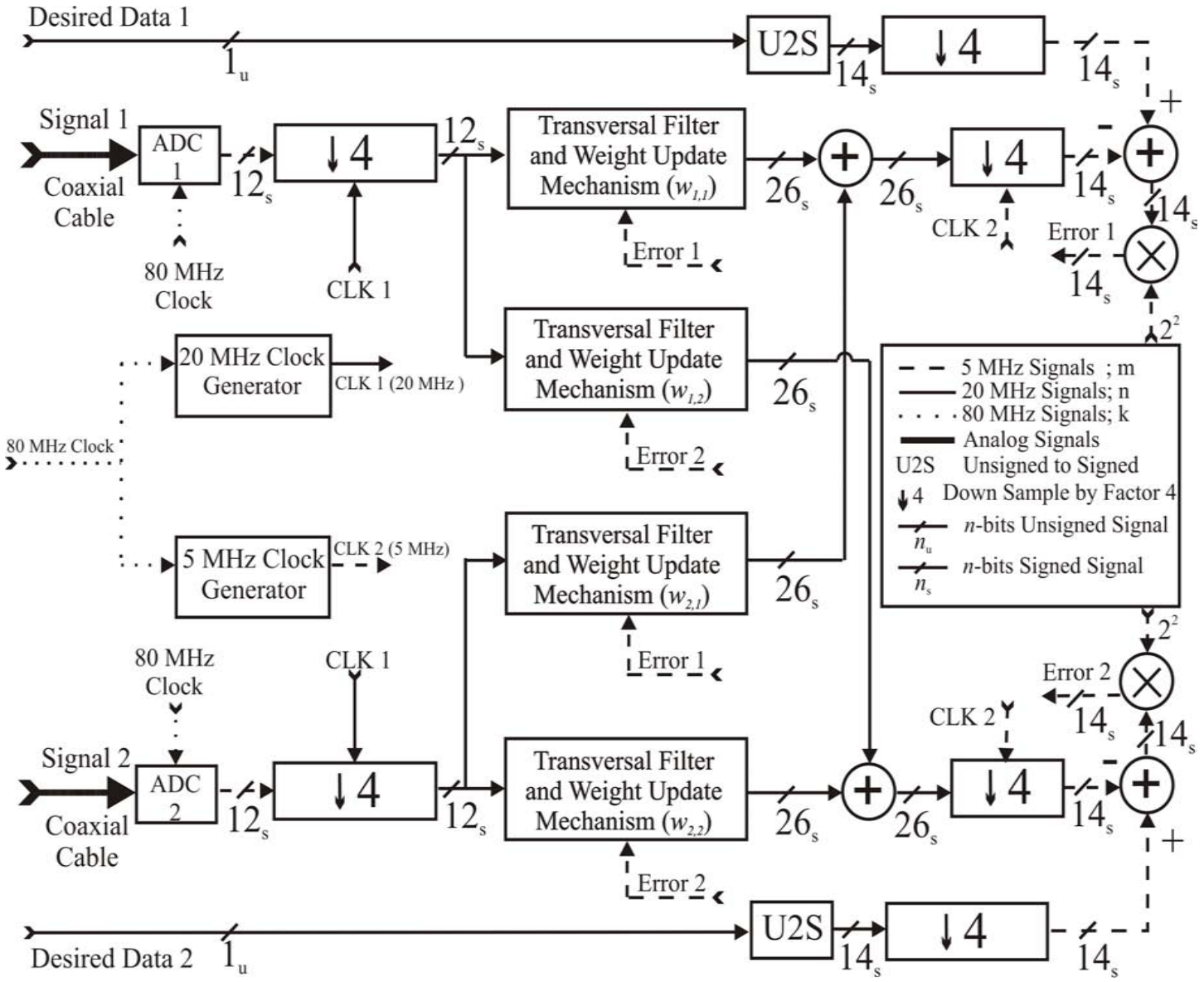

Figure 3.9: Detailed diagram of implemented receiver side

the weight update mechanism. The error signal, which is the difference between the recovered signal and the desired signal, is fed back to the weight update components to produce the next set of filter coefficients.

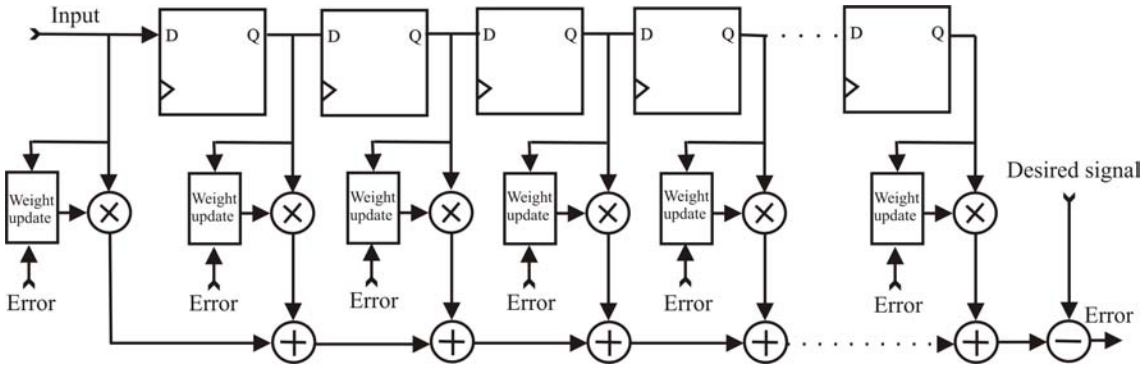

Figure 3.10: Block diagram of adaptive transversal filter [3]

3.6 Time-Alignment System

As the order of the filter increases, the longest register-to-register delay increases. First, this is due to the increase in the number of multipliers and adders from the first weight update component on the left to the error calculation circuit on the right. Second, this is due to the channel delays. This limits the speed of the system and also produces erroneous intermittent values. To overcome this, a time-alignment system was adopted, as suggested by Lin [3].

In the time-alignment system, the multipliers are pipelined. This means that the multiplier logic is broken into small elements and pipeline registers are introduced in between these elements. However, pipelining the multipliers introduces latencies in the output of the multipliers. These latencies are propagated through the system and are reflected in the calculated error. To compensate for this latency, delays are added in the desired signal to synchronize it with the output of the adaptive filters. Also, the delayed error signal is fed back to the weight update elements to

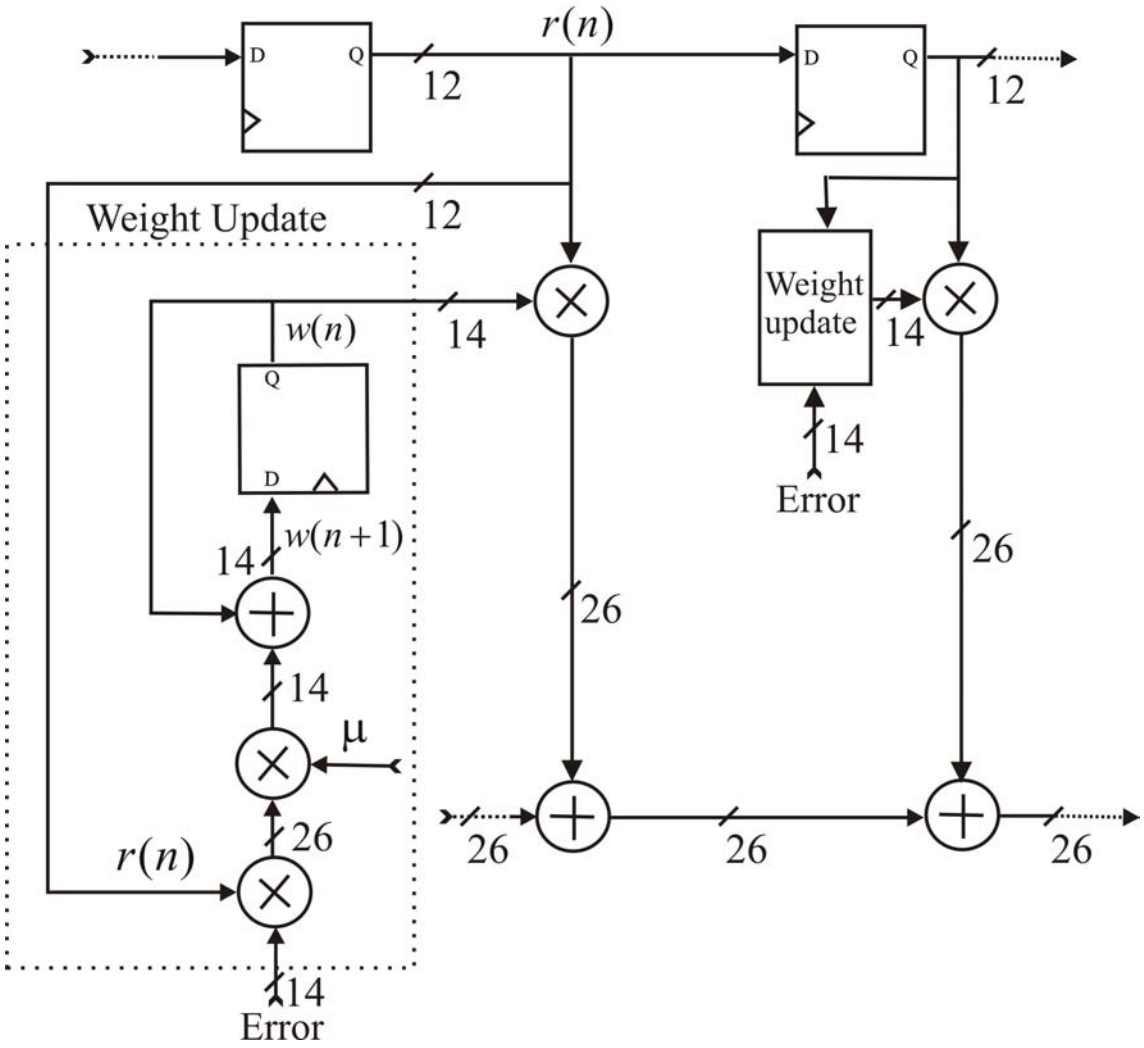

Figure 3.11: Detailed diagram of the LMS-implemented weight update mechanism

calculate the next set of coefficients. These weight update elements are therefore aligned using delayed filter taps. This modified system is shown in Figure 3.12. As a result, the system can be operated at a higher clock rate than a non-pipelined system. However, the system has a slower convergence rate [3]. The system also uses additional hardware in terms of the buffers introduced to compensate for the latencies.

3.7 Power-of-Two FPGA Arithmetic

The filter coefficients are updated based on the LMS algorithm, Equation 2.13. These mathematical equations include multiplication and subtraction. In general, the step size μ is a real number and its value is often less than one. Multiplication with a fractional number of value less than one is equivalent to division by the reciprocal of that fractional number. In order to avoid the complexity of multiplication of floating point numbers, the power-of-two scheme was used. This scheme is also known as the Arithmetic Shift Right operation. It operates on a two's complement integer, by shifting the number n-bits towards the right. It is necessary to preserve the sign bit, the most significant bit, when using signed numbers. This is equivalent to multiplying the number by 2^{-n}, where n is a positive integer.

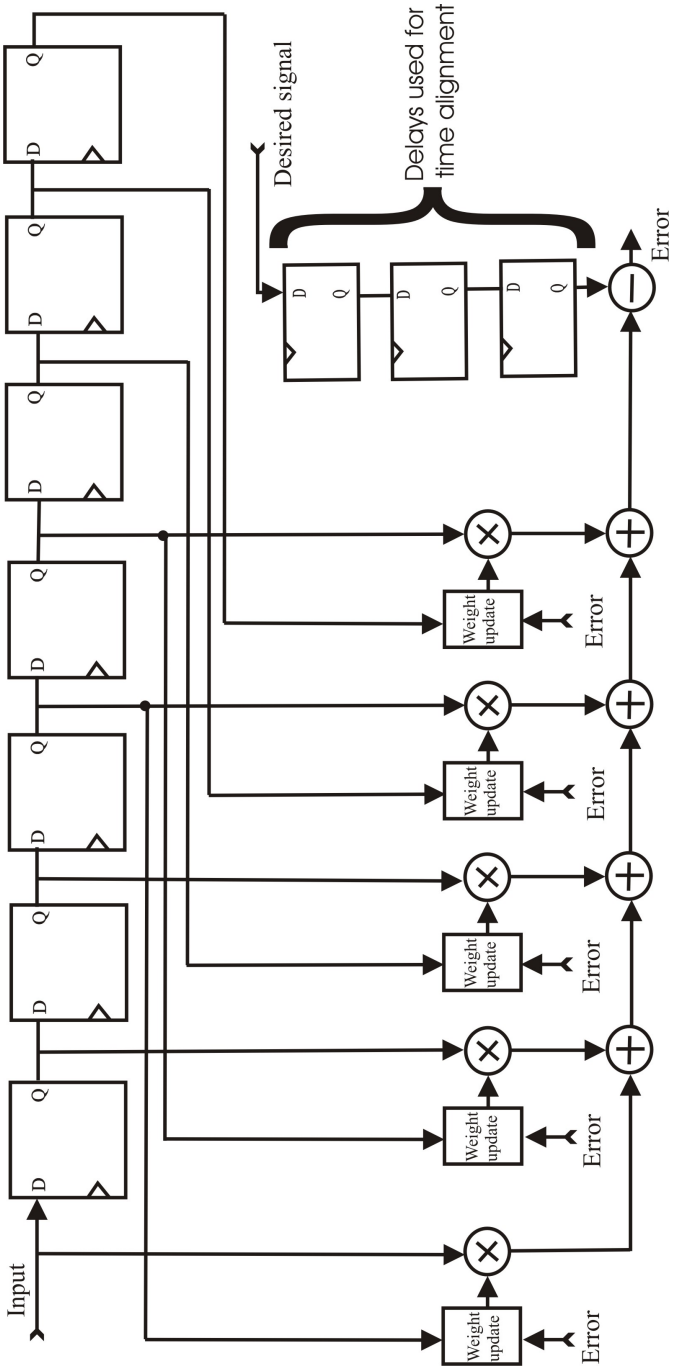

Figure 3.12: Time-alignment system block diagram [3]

Chapter 4

System Debugging and Results

The purpose of this chapter is to give a detailed explanation of the debugging and verification process, and to illustrate the results obtained from the implementation.

4.1 System Development and Debugging Process

In order to reduce the complexity of the debugging and verification process, the system was implemented in various stages. The first stage was to implement a communication system with a single user and a receiver with a single receiving antenna. The system incorporated an adaptive transversal filter having a single filter-tap. The next step was to modify this one-by-one (1X1) system and replace the transversal filter having a single-tap filter by a 25-tap filter. This being achieved, it was relatively easy to add the second user and the second receiving antenna. At every stage in the development of the system, the design was simulated in MATLAB to verify its feasibility. An iterative approach was used to implement the design in the FPGA board. The system components were developed using Verilog and were subsequently programmed into the FPGA board. The Altera Quartus II software package was used to compile the verilog-based design and Altera SignalTap II soft-

ware was used to record the system variables over a defined period of time. Once this process was completed, the recorded real-time signals were exported from the Altera SignalTapII software to the MATLAB workspace. This is a convenient method to examine the recorded variables off-line and to debug the system. Also, since the design was simulated in MATLAB, these real-time variables can be compared to the simulated values to help the debugging process.

The use of MATLAB along with SignalTap was found to be effective for design and debugging of the system. It was realized that as more and more components were added to the system, the size of the system and hence the required processing, increased significantly. This resulted in long compilation times. With the available hardware, the longest compilation time was 127 minutes. The Altera Quartus II software package synthesizes the hardware and generates a programmable bit file. Every change in the system requires the developer to run the compilation process before the design can be downloaded in the FPGA board. While debugging the system, this results in time spent waiting for the compilation process to finish. However, if the system is previously simulated in MATLAB, the developer has a better understanding of the design and requires fewer compiles of the hardware to achieve the desired results. Also, it takes less time to run the MATLAB simulation. Hence, most of the debugging can be done in MATLAB before making a change in the hardware. This can cause a significant saving of time.

MATLAB and SignalTap were the only tools used for debugging and simulation. The use of a combination of MATLAB and SignalTap tools gives a convenient and suitable method for debugging. Also, there are other tools available in the market. For example, ModelSim SE from Mentor Graphics is an effective tool for simulating and debugging FPGA designs. However, ModelSim SE helps in debugging the system based on off-line data and does not use the real-time signals for analysis.

4.2 FPGA Board Testing

In order to verify the functionality of the board, a simple design was downloaded to the FPGA. This design tested the three push button switches and two seven-segment displays available on the board. This also verified that the design can be downloaded correctly on the FPGA board using the JTAG cable and the Altera Quartus II software package. The next step was to test the ADCs and DACs on the board. For this purpose a test signal was generated using a function generator. Via coaxial cables, this signal was fed to the two ADCs on the board. The resulting signal was then examined using the SignalTap II software. This also helped to determine the range of values obtained at the ADC for different input voltages. To test the DACs, a test signal was generated within the FPGA. This signal was put out through the two DACs on the board. Via coaxial cables, these signals were then observed on the oscilloscope. This also helped in determining the range of voltage levels obtained for different output values at the DACs. It was demonstrated that the DACs required unsigned binary data to be output, whereas, the ADCs returned signed binary values in SignalTap. Since all the mathematical operations were done using signed binary numbers, it was required to convert the data being transmitted from signed to unsigned before putting it through the DACs. As a final test, the signal generated in the FPGA was put through one of the DACs and was fed back into one of the ADCs. The resulting signal was examined in the SignalTap II software and also on the oscilloscope by putting the received data through the second DAC. This process verified the functionality of the ADCs and DACs. For developers having little or no experience in implementing DSP designs on FPGA boards, this process can be a good starting point to gain familiarity and understanding of implementation practices.

4.3 Implementation Results

As mentioned earlier in this chapter, the design was synthesized and downloaded to the FPGA board using the Altera Quartus II software. A 2X2 system with antenna elements separated by a distance of one symbol wavelength was considered. The error signals generated from the difference between the recorded and estimated data were examined in the SignalTap software. The position of the transmitters was assumed as shown in the Figure 2.3. The pathological case shown in the Figure 2.4 was also considered. However, it is difficult to analyze the results in the SignalTap view. Figure 4.1 shows an example of the SignalTap software view for the resulting error signals in a 2X2 case with antenna elements separated by one symbol wavelength.

4.4 Hardware Implementation Results Examined in MATLAB

To analyze the results more efficiently, the recorded data in SignalTap was exported to MATLAB. There is no direct method to export the data from SignalTap to MATLAB. In order to get the data into MATLAB, the recorded data must be converted into a text (.txt) file. This is done by using the 'Create SignalTap II list file' option. This file must then be modified in any word processing software such as Microsoft Word or Notepad and the lines of header information must be deleted. This modified file can then be imported into MATLAB by using the 'textread' command. The imported data is in a matrix form. By examining the signal legend in the header of the .txt file, individual variables of interest can be separated from the matrix. It must be noted that SignalTap gives the data in hexadecimal representation by default. These must be converted into signed or unsigned decimal numbers

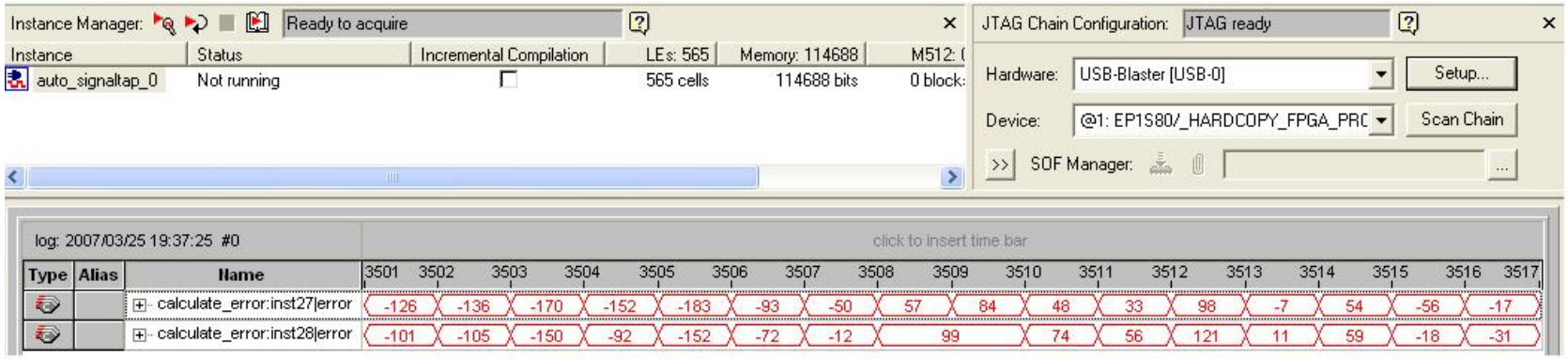

Figure 4.1: SignalTap view of the error signals

before creating the text file. The SignalTap software has an option to change the representation of the data from hexadecimal to signed or unsigned decimal. The results of the implementation can then be plotted in MATLAB. In order to plot the MSE values in dB, it is required to convert the zero error values to some positive value to avoid the undefined $\log_{10}(0)$ operation. This is done by converting the zero error values to an integer value of 1, the smallest non-zero value of the error in a fixed-point representation of the error signals.

4.4.1 Learning Curves for the System in General Case

The learning curves for the LMS algorithm and the adapted linear combiner coefficients of the system with antenna elements separated by one symbol wavelength are shown in the Figure 4.2. 4000 samples were collected from SignalTap and plotted in MATLAB.

From the two learning curves in Figure 4.2 it can be seen that the system has converged and the MSE after convergence is -15.7 dB, averaged from both curves. After convergence, the coefficients of the adaptive transversal filters provide an estimation of the simulated LOS MIMO channels. Since the coefficients are adapted using the training sequence, the global minima is approached due to the error signal having the properties of white noise. Also, it is observed that the learning curves show quantization effects. This is due to the rounding-off of the error values to the nearest integer value. The results indicate that after convergence the values of some of the coefficients are near zero. This estimation can be used to place the coefficients of the adaptive transversal filters only where the energy of the coefficients is concentrated in the estimation and set the near-zero coefficients to zero. This can help in reducing the hardware required to implement the adaptive transversal filters.

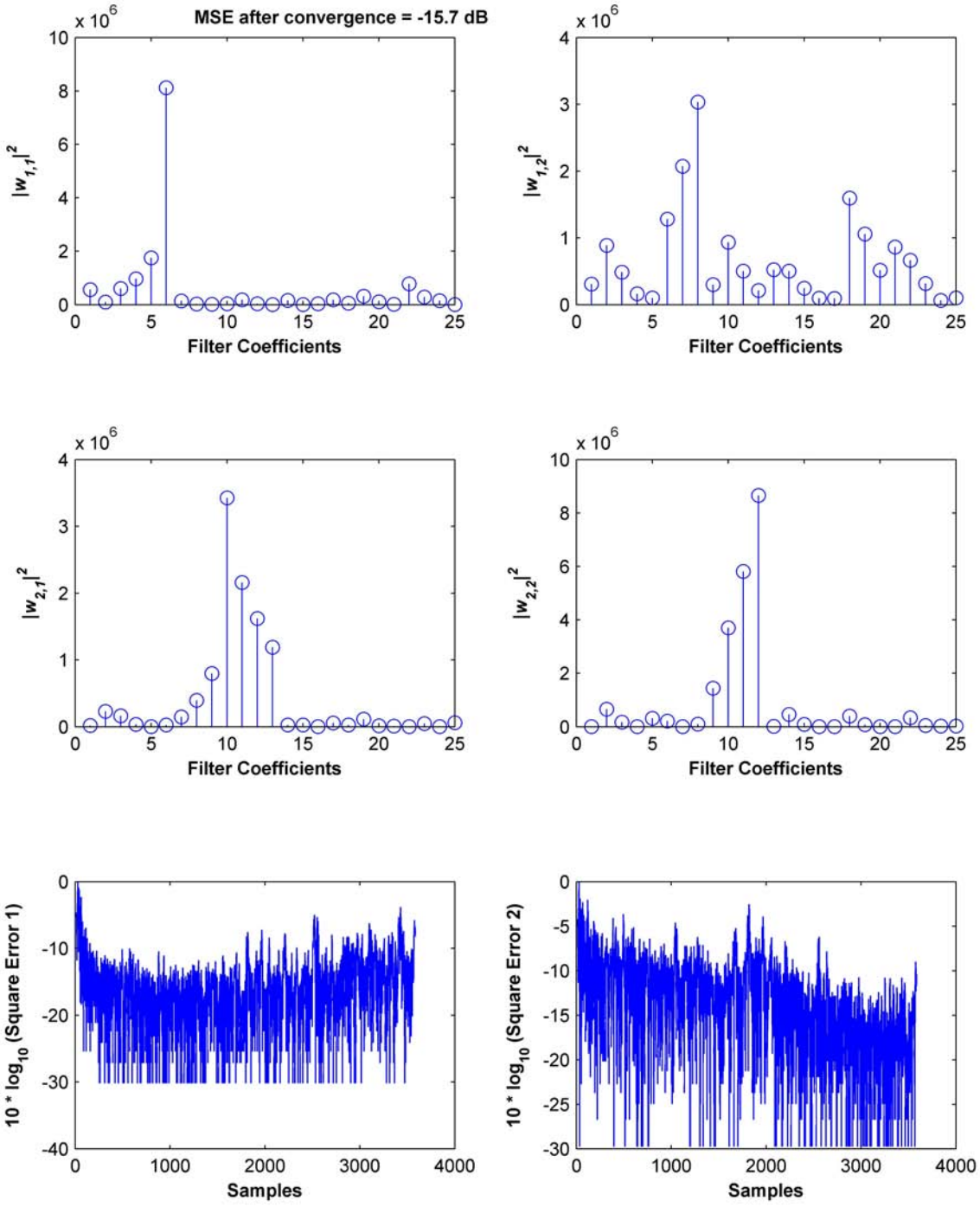

Figure 4.2: Linear combiner filter coefficients and the MSE learning curves obtained from the FPGA implementation – General case

4.4.2 Learning Curves for the System in Pathological Case

The pathological case showed by Zhu [8] was investigated. The learning curves and the adaptive linear combiner coefficients for the pathological case are shown in Figure 4.3. From the figure, due to the high MSE, it is clear that the system fails to converge when the transmitters are positioned in a pathological case. This is in accordance with the work of Zhu [8] and Polu [17]. The value of MSE for the pathological case is -5.03 dB.

4.5 MSE Learning Curves due to Time Variations

Figure 4.4 shows a comparison of the MSE learning curves obtained from the MATLAB simulation and the FPGA implementation for the system in the general case. The top two curves in Figure 4.4 are the MSE learning curves obtained from the MATLAB simulation. These curves are also shown in Figure 3.1. The bottom two curves in Figure 4.4 are the MSE learning curves obtained from the FPGA implementation. These curves are also shown in Figure 4.2. The implementation results are as predicted in the simulation. However, in the hardware implementation there is a considerable amount of noise. Some thermal noise is introduced because of the cables and the losses in the signal due to connection mismatching.

The system uses 25 taps in each of the four transversal filters. In order to verify if a sufficient number of taps are used, this system was tested by increasing the number of taps to 35. However, the system showed little or no performance improvement with the increase in the number of taps.

The ADCs have a lower cut-off frequency of 1 MHz. The adaptive receiver tries to compensate for the 1 MHz lower cut-off frequency. This affects in the enhancement of the noise. It was also observed that the MSE learning curves show a variation with samples, or time, when 16000 samples were collected instead of 4000

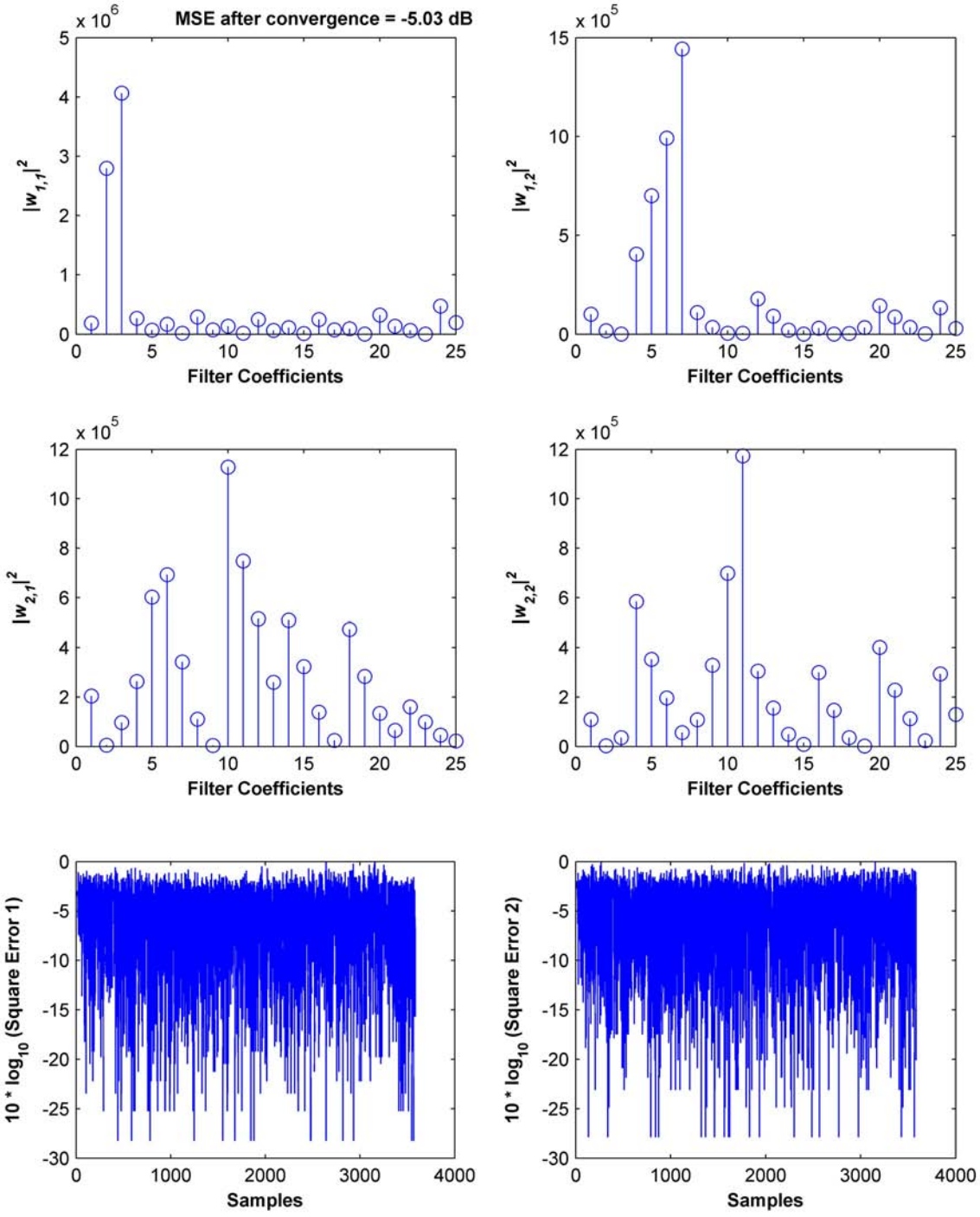

Figure 4.3: Linear combiner filter coefficients and the MSE learning curves obtained from the FPGA implementation – Pathological case

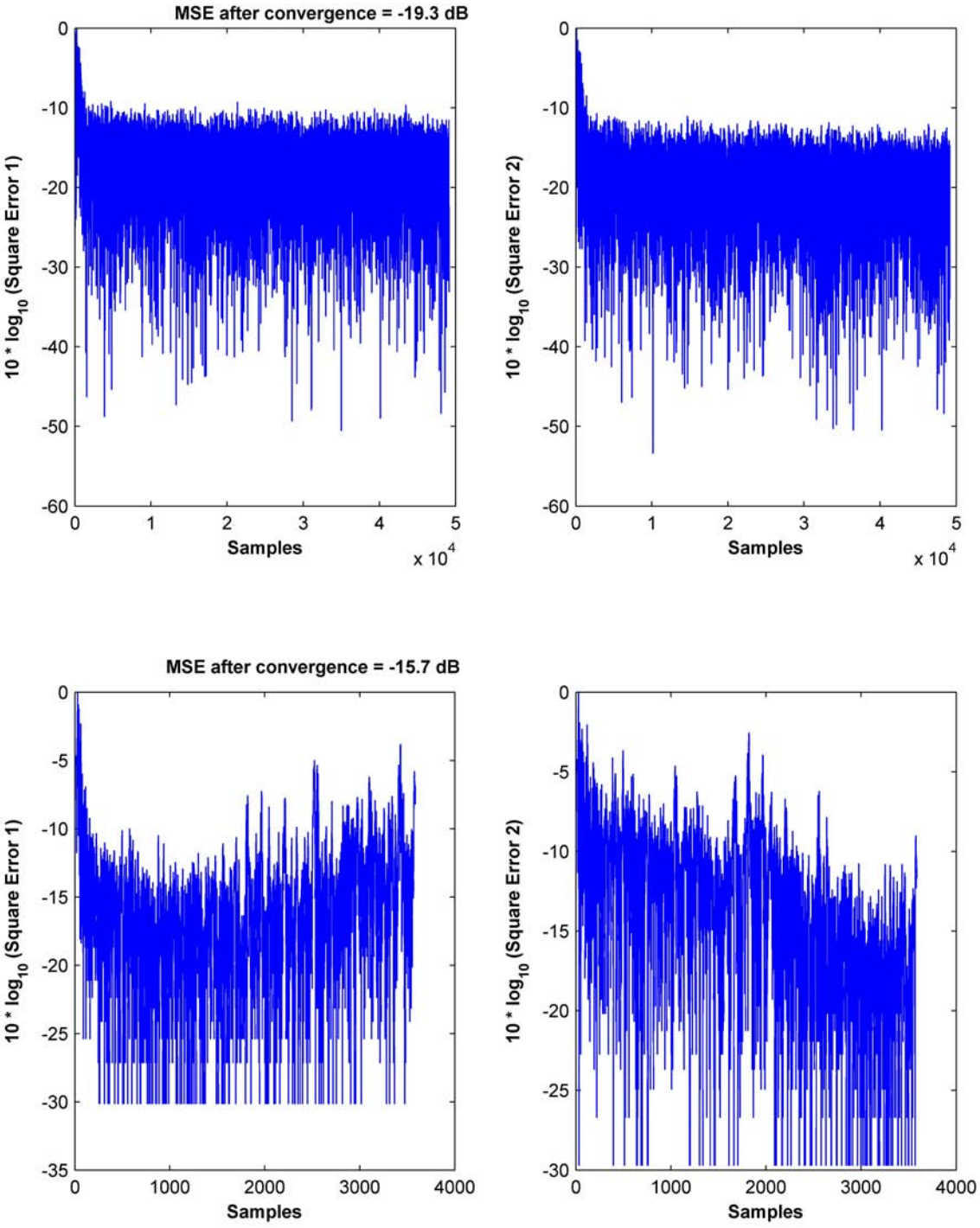

Figure 4.4: MSE learning curves obtained from the MATLAB simulation and the FPGA implementation for the general case

samples. Figure 4.5 shows these variations in the MSE learning curves.

These variations might be an effect of the ADCs, DACs, or clock frequency variations. To investigate this, the system was modified by replacing the hardware connection between the transmitter and the receiver by a software connection within the FPGA. These results are shown in the Figure 4.6. It can be seen from the results that system performance is significantly improved and the variations in the learning curves are no longer seen. The value of MSE after convergence for this case is -28.2 dB.

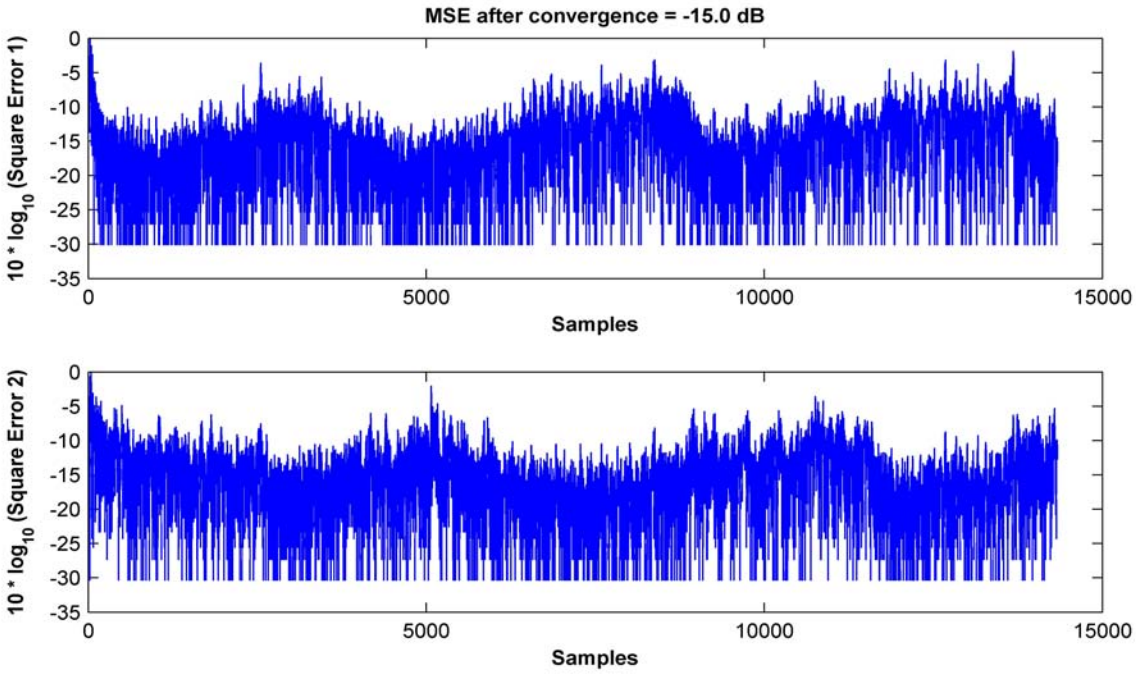

Figure 4.5: Variation in the learning curves with time obtained from the FPGA implementation for general case

Based on the findings of Zhu [8] and Polu [17], the performance of the system improves as the separation between the antenna elements is increased to one symbol wavelength. To investigate this, the system was tested with the receiving antenna element separation of 1/4 of a symbol wavelength in a general case scenario. The system performance showed no significant change when the test was performed using

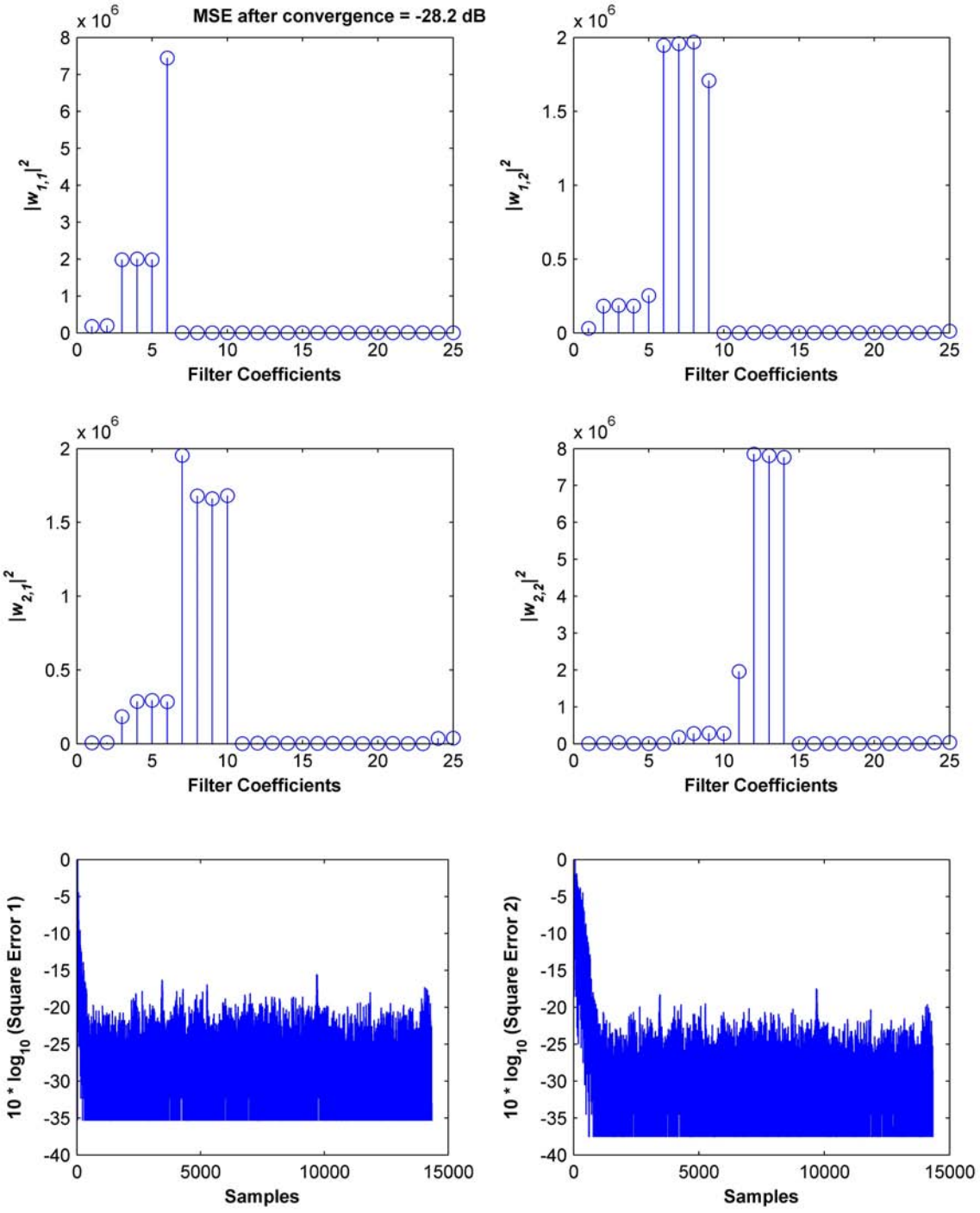

Figure 4.6: Linear combiner filter coefficients and MSE learning curves obtained from the FPGA implementation for general case – No ADCs and DACs

a hardware connection between the transmitter side and the receiver side. However, when the same test was performed with no ADCs and DACs using a software connection between the transmitter side and the receiver side, the MSE after convergence gets poorer, from -28.1 dB to -25.6 dB, as the separation between the antennas is reduced from one symbol wavelength to 1/4 of a symbol wavelength. Figure 4.7 shows the MSE learning curves for this test.

To investigate the effect of length of training sequences, LRSs of 31-bits were used instead of 4095-bit sequences. Figure 4.8 shows the MSE learning curves obtained from the FPGA implementation for a 2X2 system in the general case with 31-bit LRSs. The results show that with the use of 31-bit LRSs, periodic spikes are introduced in the learning curves. These spikes are spaced 31-bits apart. This is due to the recursive nature of the LRSs. To avoid this, use of longer LRSs is suggested.

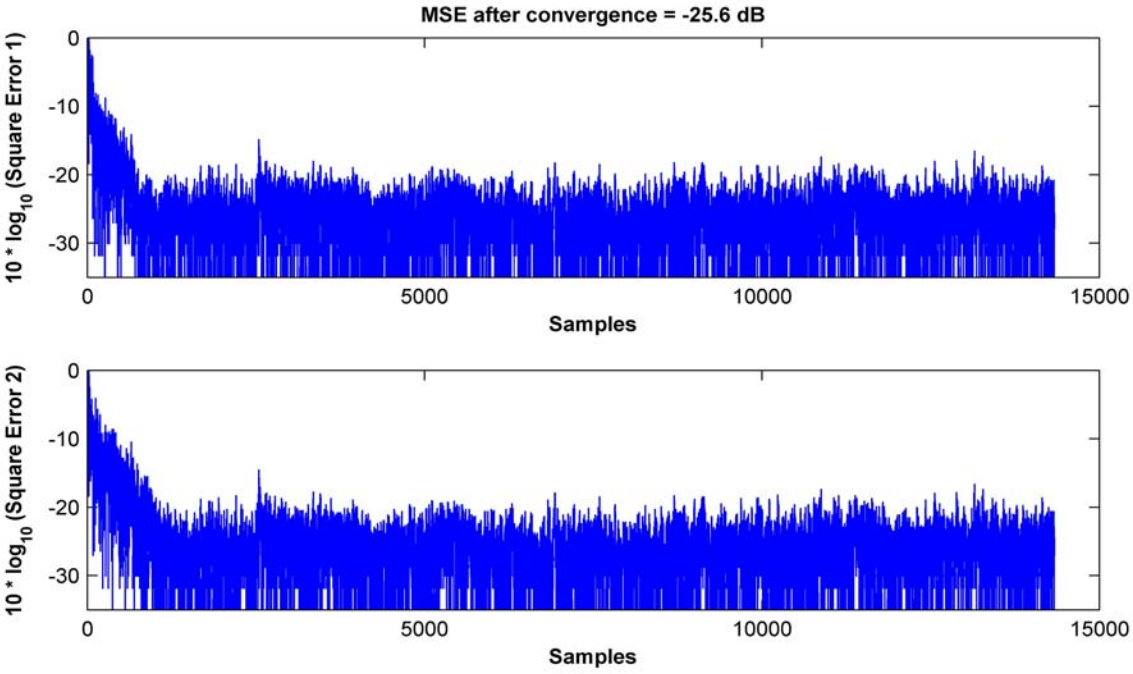

Figure 4.7: MSE learning curves obtained from the FPGA implementation – General case with antenna elements separated by 1/4 of a symbol wavelength – No ADCs and DACs

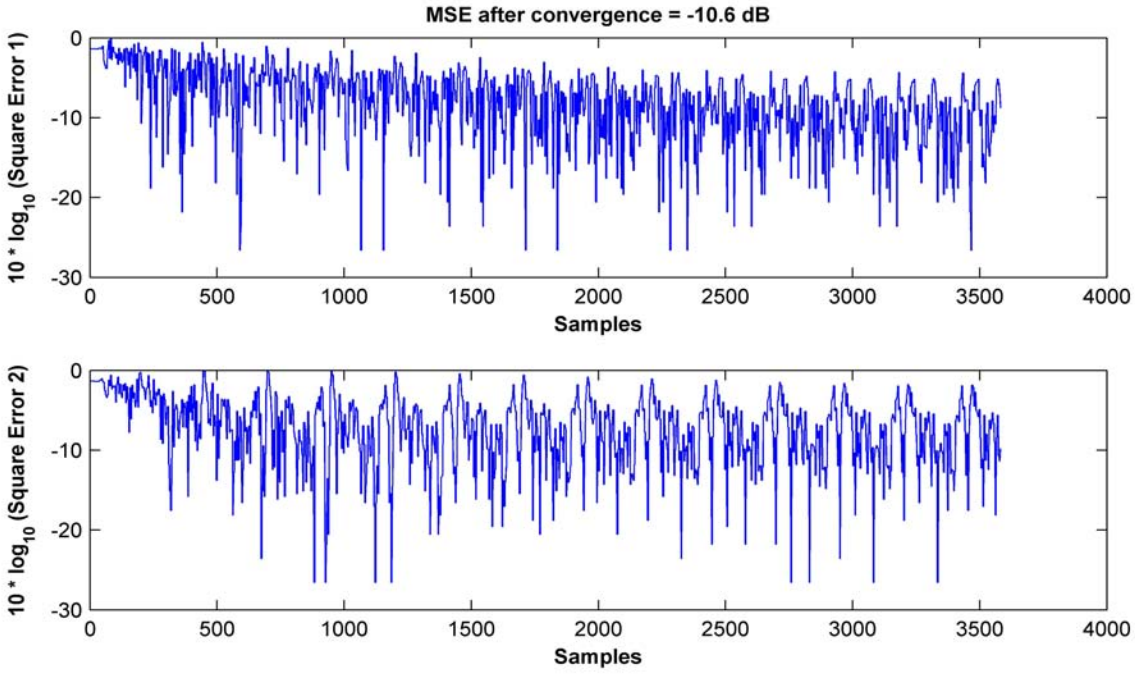

Figure 4.8: Variation in learning curves for the general case with the use of 31-bit sequences

4.6 Hardware Implementation Issues

During the development of the system, many issues were encountered. This section gives a brief description of these issues. The steps taken to resolve these issues; their possible solutions are also listed.

The implemented system has four adaptive transversal filters. These adaptive transversal filters have 25 taps each. For the implementation of each tap two 14-bit-by-14-bit multipliers are required. The Altera Stratix EP1S80 FPGA chip has 176 9-bit-by-9-bit embedded multipliers. These are the dedicated hardware multipliers that are more efficient in terms of speed compared with general logic. Since the system has four transversal filters with 25 taps in each filter, the total required number of 9-bit-by-9-bit multipliers is 400. Thus, the embedded multipliers could not be used. This issue was resolved by using a logic element implementation of the multipliers. This method implements the multipliers in terms of logic elements

instead of the dedicated hardware multiplier circuitry on the chip. The implemented system uses 62842 logic elements from the available 79040 logic elements on the Altera EP1S80 FPGA board. To be able to take advantage of the high-performance embedded multipliers, an FPGA device which has a greater number of embedded multipliers is suggested. One such device is the Altera Stratix II EP2S180 FPGA chip. Also, as explained in Chapter 2, the performance of the multipliers deteriorates with the increase in the number of taps. This is due to the increase in the longest register-to-register delay. This issue was resolved by pipelining the multiplier stages.

Since the data rate is selected to be 5 Mbps, a sampling frequency of 10 MHz is high enough to satisfy the Nyquist's criterion. However, raising the sampling frequency would require more taps for successful convergence of the system. If more filter taps are used in the system, more processing, in terms of mathematical operations would be required. This increases the hardware requirement and subsequently increases the size of the system. To avoid this, the data was up-sampled to 20 Mbps. Thus the system requires fewer filter taps to converge and the processing speed of the system is also increased.

When implementing DSP algorithms in an FPGA, it is a common problem that the number of bits representing the data increases unnecessarily. This tendency is more evident when the result of an arithmetic operation is fed back to calculate the next set of results. In this thesis, the error signal generated at the output of the adaptive transversal filters, is fed back to calculate the next set of filter-coefficients. The extra bits used to represent the data, increases the processing in the FPGA and also the size of the system. These unnecessary bits were minimized using the MATLAB simulation; the MATLAB simulation was used to determine the dynamic range of the signals of interest and subsequently determine the number of bits required to represent the signals.

The step size μ is a real number and its value is always small enough to

maintain stability and achieve convergence. To represent this fractional number, the floating-point number system can be used. The floating-point number system offers a wide dynamic range with maximum precision. However, floating-point numbers consist of multiple parts and in an arithmetic operation each part needs to be differently treated. This leads to reduction in the operational speed and increases complexity [31]. To avoid the complexity of mathematical operations on floating-point numbers, the power-of-two scheme was used to represent the value of step size μ. This scheme is described in detail in Chapter 3. The value of μ is now limited to 2^N, where N is a positive integer. Due to this, certain values of μ could not be used, which might have given slightly better results. Also, the values of various signals within the FPGA are rounded off to the nearest integer value. The LMS algorithm may become unstable, giving arbitrarily large errors and coefficient values, due to the finite precision effect of the round-off errors. In order to ensure the stability of the algorithm, the error signals and the adaptive filter coefficients must be conditioned and initialized appropriately [32].

Chapter 5

Summary and Future Work

5.1 Summary of Work Completed

A MIMO multi-user system was implemented on an FPGA. The system consists of two users and an adaptive ST receiver with two antenna elements. The adaptive receiver uses an LMS algorithm to update the filter coefficients. The system incorporates the benefits of separating the antenna elements on a scale of a symbol wavelength. Furthermore, the thesis demonstrates the symbol-wavelength effect on the coefficients and adaptation of a linear combiner which is implemented on an FPGA. This system will be useful for real-time investigation of MIMO multi-user systems and the effect of SWAP gain in real-world environment. The design and implementation process of the system is presented in detail.

The results of the implementation indicate that the system successfully converges and provides successful channel equalization and user detection. The pathological case shown by Zhu [8] was investigated and the failure of the 2X2 system for the pathological case was confirmed from the results. Some coefficients of the adaptive transversal filters are near zero. These near-zero coefficients can be set to zero and the coefficient taps can be placed only where the energy of the coefficients

is concentrated. This can help in reducing the hardware required to implement the transversal adaptive filters.

A MATLAB simulation was also developed for testing and debugging purposes. Various issues encountered during the course of the implementation have been discussed. The steps taken to resolve these issues are listed or possible suggestions to solve these issues have been made.

5.2 Future Work

In this thesis, the implemented system has two users and two receiving antennas. This work can be extended to incorporate more users and receiving antennas. The effect of additional users and receiving antennas on the system performance can be investigated. This can be done by building upon this thesis and additional FPGA boards.

Possibilities for receiver complexity reduction, using the coefficients of the adaptive transversal filters after convergence, have been identified. This work can be extended by setting the near-zero coefficients to zero and placing the filter taps only where needed.

The system can be further developed by replacing the simulated LOS channels with commercial off-the-shelf radio transceivers and antennas to provide a complete real-time testbed for making radio channel measurements.

The transmitted data can be further modified and a raised cosine filter can be used to provide pulse shaping. The receiver used in this thesis is a linear combiner ST receiver. This receiver could be developed into a non-linear combiner receiver by using techniques such as, a decision feedback equalizer.

Bibliography

[1] *Altera Stratix EP1S80 DSP Development Board Manual*, Altera Corporation, December 2004, ver. 1.3.

[2] R. L. Peterson, R. E. Ziemer, and D. E. Borth, *An Introduction to Spread-Spectrum Communications*, 1st ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995.

[3] A. Y. Lin, "Implementation considerations for FPGA-based adaptive transversal filter designs," Master's thesis, University of Florida, Gainesville, FL, USA, 2003.

[4] D. Gesbert, M. Shafi, D. Shiu, P. J. Smith, and A. Naguib, "From theory to practice: An overview of MIMO space–time coded wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, April 2003.

[5] S. Verdú, *Multiuser Detection*, 1st ed. New York, NY, USA: Cambridge University Press, 1998.

[6] Q. H. Spencer, C. B. Peel, A. L. Swindlehurst, and M. Haardt, "An introduction to the multi-user MIMO downlink," *IEEE Communications Magazine*, vol. 42, no. 10, pp. 60–67, October 2004.

[7] V. Bale and S. Weiss, "Equalization of broadband MIMO channels by subband adaptive identification and analytic inversion(poster)," in *Proceedings of International ITG/IEEE Workshop on Smart Antennas (WSA)*, Duisburg, Germany, April 2005.

[8] G. Zhu, B. G. Colpitts, and B. R. Petersen, "Signalling wavelength in an antenna array for space-time wireless over LOS channels," in *Third Annual Communication Networks and Services Research Conference (CNSR) 2005*, Halifax, Nova Scotia, Canada, May 2005, pp. 69 – 73.

[9] P. Atiniramati, "Design and implementation of an FPGA-based adaptive filter single-user receiver," Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, USA, 1999.

[10] J. H. Winters, "The impact of antenna diversity on the capacity of wireless communication systems," *IEEE Transactions on Communications*, vol. 42, no. 4, pp. 1740–1751, April 1994.

[11] A. R. Calderbank, G. Pottie, and N. Seshadri, "Cochannel interference suppression through time/space diversity," *IEEE Transactions on Information Theory*, vol. 46, no. 3, pp. 922–932, May 2000.

[12] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*, 1st ed. 40 West 20th Street, New York, NY, USA: Cambridge University Press, 2003.

[13] L.-K. Ting, "Algorithms and FPGA implementations of adaptive LMS-based predictors for radar pulse identification," Ph.D. dissertation, Queen's University Belfast, Northern Ireland, UK, 2001.

[14] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*, 2nd ed., ser. Signals and Communication Technology Series. New York, USA: Springer-Verlag, February 2007.

[15] D. Pellerin and S. Thibault, *Practical FPGA programming in C*. Englewood Cliffs, N.J., USA: Prentice-Hall, 2005.

[16] H. Yanikomeroglu and E. S. Sousa, "Antenna gain against interference in CDMA macrodiversity systems," *IEEE Transactions on Communications*, vol. 50, no. 8, pp. 1356 – 1371, Aug. 2002.

[17] V. V. S. N. Polu, B. G. Colpitts, and B. R. Petersen, "Symbol-wavelength MMSE gain in a multi-antenna UWB system," in *Fourth Annual Communication Networks and Services Research Conference (CNSR) 2006*, Moncton, New Brunswick, Canada, May 2006, pp. 95 – 99.

[18] J. A. Harriman, "A reconfigurable four-channel transceiver testbed with signalling-wavelength-spaced antennas," Master's thesis, University of New Brunswick, Fredericton, New Brunswick, Canada, 2006.

[19] R. M. Rao, W. Zhu, S. Lang, C. Oberli, D. Browne, J. Bhatia, J. F. Frigon, J. Wang, P. Gupta, H. Lee, D. N. Liu, S. G. Wong, M. Fitz, B. Daneshrad, and O. Takeshita, "Multi-antenna testbeds for research and education in wireless communications," *IEEE Communications Magazine*, vol. 42, no. 12, pp. 72–81, December 2004.

[20] C. Mehlfuhrer, F. Kaltenberger, M. Rupp, and G. Humer, "A scalable rapid prototyping system for real-time MIMO OFDM transmissions," in *The 2nd IEE/EURASIP Conference on DSP enabled Radio*, Southampton, UK, September 2005, p. 7.

[21] S. Verdú, "Optimum multiuser asymptotic efficiency," *IEEE Transactions on Communications*, vol. COM-34, no. 9, pp. 890–897, September 1986.

[22] N. Seshadri, "Joint data and channel estimation using blind trellis search techniques," *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 1000–1011, February/March/April 1994.

[23] V. Tarokh, N. Seshadri, and A. R. Calderbank, "Space-time codes for high data rate wireless communications: Performance criterion and code construction." *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 744–765, march 1998.

[24] H. Li, X. Lu, and G. B. Giannakis, "Capon multiuser receiver for CDMA systems with space-time coding," *IEEE Transactions on Signal Processing*, vol. 50, no. 5, pp. 1193–1204, May 2002.

[25] Y. Rong, S. A. Vorobyov, and A. B. Gershman, "Robust linear receivers for multiaccess space–time block-coded MIMO systems: A probabilistically constrained approach," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1560–1570, August 2006.

[26] W. F. Gabriel, "Adaptive arrays - an introduction," in *Proceedings of the IEEE*, vol. 64, no. 2, February 1976, pp. 239 – 272.

[27] S. S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ, USA: Prentice Hall, 1996.

[28] J. R. Treichler, C. R. Johnson, and M. G. Larimore, *Theory and Design of Adaptive Filters*. New York, NY, USA: Prentice Hall, 2001.

[29] S. U. H. Qureshi, "Adaptive equalization," *Proceedings of the IEEE*, vol. 73, no. 2, pp. 1349– 1387, 1985.

[30] G. Ungerboeck, "Fractional tap-spacing equalizer and consequences for clock recovery in data modems," *IEEE Transactions on Communications*, vol. 24, no. 8, pp. 856–864, August 1976.

[31] R. Hashemian and B. Sreedharan, "A hybrid number system and its application in FPGA-DSP technology," in *Proceedings of the International Conference on Information Technology: Coding and Computing*, vol. 2, Las Vegas, Nevada, USA, April 2004, pp. 342–346.

[32] M. L. Honig and D. G. Messerschmitt, *Adaptive Filters: Structures, Algorithms, and Applications*. Norwell, MA, USA: Kluwer Academic Publishers, September 1984.

Appendix A

Verilog Design Source Code

A.1 Clock Generator Module

```
////////////////////////////////////////////////////////////////////////////////////
//
// This verilog module generates the 5 MHz and 20 MHz clocks using
// the on-board 80 MHz clock.
//
// Author: Harshal Desai
//
// Last modified : April 2007
//
////////////////////////////////////////////////////////////////////////////////////

module clkdiv(inclk, outclk);

input inclk; 80 MHz clock
output outclk; 20 MHz or 5 MHz clock

reg outclk;
reg [3:0]cnt;

   always @ (posedge inclk)

       if (cnt == 4'b0000)
```

```
          begin

           outclk <= ~ outclk;
           cnt <= 4'b0111; // For generation of 5 MHz clock
         // cnt <= 4'b0001; // For generation of 20 MHz clock

          end

       else

            cnt <= cnt - 4'b0001;

endmodule
```

A.2 Linear Recursive Sequence Generator

```
////////////////////////////////////////////////////////////////////////////////////////////
//
// Verilog module for LRS generators. This code also performs the upsampling
// ofdata from 5 MHz to 20 MHz and conversion of 1-bit unsigned binary data
// to 3-bit signed binary data.
//
// Author: Harshal Desai
//
//
// Last modified : April 2007
//
//
////////////////////////////////////////////////////////////////////////////////////////////


module bitstream ( dn,clk, clockdiv, reset, data_one);

input clk; // 20 MHz clock
input clockdiv; 5 MHz clock
input reset;

output   dn; // 1-bit desired data
output [2:0] data_one; // 3-bit signed binary upsampled data

reg  dn;
reg [2:0] data_one;
reg data;
reg [11:0]pi;
reg [3:0] cnt;

// generation of 1-bit signed binary data using shift register and
// exclusive OR gates.

always @(posedge clockdiv or negedge reset)
  if(!reset)
    begin

pi <= 12'b000000000001;
dn <= 1'b0;
```

```
    end
  else
    begin

// polynomial : x12+x6+x4+x+1
pi[11] <= pi[6]^pi[4]^pi[1]^pi[0];

// polynomial : x12+x11+x9+x8+x7+x5+x2+x+1
// pi[11] <= pi[11]^pi[9]^pi[8]^pi[7]^pi[5]^pi[2]^pi[1]^pi[0];
pi[10:0] <= pi[11:1];
data <= pi[0];
dn <= pi[0];

    end

// This code also performs the upsampling of
// data from 5 MHz to 20 MHz and conversion of
// 1-bit unsigned binary data to 3-bit signed binary data.

always @ (posedge clk or negedge reset)
if(!reset)
     begin

     data_one <= 3'b000;

     end

   else
begin

if (cnt == 0)
         begin

if (data == 1)
begin

data_one <= 3'b001;

end
```

```
else

begin

data_one <= 3'b111;

end

cnt <= 4'b0011;

       end
     else
                begin

                     data_one <= 3'b000;
                     cnt <= cnt - 4'b0001;

                 end

end

endmodule
```

A.3 Signed Binary Number to Unsigned Binary Number

```
////////////////////////////////////////////////////////////////////////////////////////
//
// This verilog module is used to convert 3-bit signed binary data to 14-bit
// unsigned binary data for the DACs.
//
//
// Author: Harshal Desai
//
//
// Last modified : April 2007
//
//
////////////////////////////////////////////////////////////////////////////////////////

module busconvert (in,out,clk);

input [2:0] in;
input clk;

output [13:0] out;   // Comment these lines to use the system with
reg [13:0] out;      // no ADCs and DACs (software connection)

//output [11:0] out; // Uncomment this to use the system with
//reg [11:0] out;    // no ADCs and DACs (software connection)

always @ (in)

begin
case(in)

3'b010: out <= 14'b11111111111111; // Comment these
3'b001: out <= 14'b10111111111111; // lines to use
3'b000: out <= 14'b01111111111111; // the system
3'b111: out <= 14'b00111111111111; // with no ADCs
3'b110: out <= 14'b00000000000000; // and DACs.
```

```
//3'b010: out <= 12'h299; // Uncomment these
//3'b001: out <= 12'h14C; // lines to use
//3'b000: out <= 12'h000; // the system
//3'b111: out <= 12'hEB3; // with no ADCs
//3'b110: out <= 12'hD67; // and DACs.


endcase


end
endmodule
```

Appendix B

MATLAB® Simulation Source Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This MATLAB code simulates the system implemented on the FPGA board.
%
% Author: Harshal Desai
%
% Last modified : April 2007
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear functions;
clear all;
clear;
clc;


 % (-), LMS adaptation constant
mu= 2^(-12);

 % (-), number of points in w
Npoints_w  = 25 ;

% transmitter data

% 31-bits sequence 1
%polynomial0 = [0 1 0 0 1];
```

```
% 31-bits sequence 2
%polynomial1 = [0 0 1 0 1];

% 4095-bits sequence 1
polynomial0 = [0 0 0 0 0 1 0 1 0 0 1 1];

% 4095-bits sequence 2
polynomial1 = [1 0 1 1 1 0 1 0 0 1 1 1];


% The function bp_shift_register_sequence calculates the output of a linear feedback shift
% register with initial states all ones, and polynomial0 and polynomial1 describing the
% the connections of the linear feedback shift register
% References:
% Michel C. Jeruchim, Philip Balaban and K. Sam Shanmugan, Simulation of
% Communication systems. New York, NY, USA: Plenum Press, 1992, p.285.
% ISBN 0-306-43989-1. LCCN 91-48369.
%
% Shu Lin and Daniel J. Costello, Jr., Error Control Coding:
% Fundamentals and Applications. Englewood Cliffs, NJ, USA:
% Prentice-Hall, Inc., 1980.
%
% W. Wesley Peterson and E. J. Weldon, Jr., Error-Correcting
% Codes. Cambridge, MA, USA: The MIT Press, 2nd ed., 1972.
%

sequence0 = bp_shift_register_sequence ( polynomial0 );
sequence1 = bp_shift_register_sequence ( polynomial1 );


Nsim = 3 ;
padd0 = [] ;
for isim = 1 : Nsim
    padd0 = [ padd0 sequence0 ] ;
end
d0 = padd0 ;

padd1 = [] ;
for isim = 1 : Nsim
    padd1 = [ padd1 sequence1 ] ;
```

```
end
d1 = padd1 ;

% Conversion to signed data
d0  =  2 * d0 - 1 ;
d1  =  2 * d1 - 1 ;


b0 =  zeros( 1 , length(d0) * 3 ) ;
b1 =  zeros( 1 , length(d1) * 3 ) ;

% Upsampling the data

x=1;
for i = 1:length(d1),
    b0hat(x) = d0(i);
    b1(x) = d1(i);
    x = x + 1;

    for j = 1:3,
        b0hat(x) = 0 ;
        b1(x) = 0 ;
        x = x + 1;
    end
end
b0 = b0hat(1:length(b1));

% Simulation of LOS channels

c11 = [0 0 0 0 0 b0 0 0];
c21 = [0 0 0 0 0 0 b1 0];
c22 = [0 0 b1 0 0 0 0 0];
c12 = [0 0 0 0 0 b0 0 0];


% Output of the transmitter side

r0 = c11 + c21;
r1 = c12 + c22;

% generating desired data with delays
```

```
bdn1 = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 b0];
bdn2 = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 b1];

% Modifying the received data to simulate real-time environment and match
% the simulation to the FPGA implementation. The preceding zeros are used
% to simulate the delay in the received data at the ADCs.


dac0 =  341 * [0 0 0 0 0 0 0 0 r0];
dac1 = 341 *  [0 0 0 0 0 0 0 0 r1];

% Adding noise to the received data

sigman1    = 10^( 2 ) ;

n1  = sqrt(sigman1) * randn(1,length(dac0)) ;
n2  = sqrt(sigman1) * randn(1,length(dac1)) ;

dac0 = dac0 + n1;
dac1 = dac1 + n2;


% Initializing the adaptive transversal filters in the receiver.

w11   =  zeros(1,Npoints_w);
w22   =  zeros(1,Npoints_w);
w12   =  zeros(1,Npoints_w);
w21   =  zeros(1,Npoints_w);

rn11  =  zeros(1,Npoints_w) ;
rn22  =  zeros(1,Npoints_w) ;
rn12  =  zeros(1,Npoints_w) ;
rn21  =  zeros(1,Npoints_w) ;

%initializing vectors to record the value of error

errors1   =   0 * dac0 ;
errors2   =   0 * dac1 ;
e1 = 0;
e2 = 0;
```

```
j = 0;
value1=0;
value2=0;


for i  =  1 : length(dac0) - 1 ,

    % Regular FIR filtering

        rn11  =  [ dac0(i)  rn11( 1 : (Npoints_w-1) ) ] ;
        rn22  =  [ dac1(i)  rn22( 1 : (Npoints_w-1) ) ] ;
        rn12  =  [ dac0(i)  rn12( 1 : (Npoints_w-1) ) ] ;
        rn21  =  [ dac1(i)  rn21( 1 : (Npoints_w-1) ) ] ;



        u11  =  w11 * (rn11 .') ;
        u22  =  w22 * (rn22 .') ;
        u21  =  w21 * (rn21 .') ;
        u12  =  w12 * (rn12 .') ;

    % Combing the output of the filters

        rx1 = u11 + u21;
        rx2 = u22 + u12;

   % Simulating Downsampling at the receiver

    if   (rem(i,4) == 0 )  ,

    % Calculating the error using the desired data and the output of
    % the filters.


        if (bdn1(i) == 1) value1 = 2047;
            else if (bdn1(i) == -1) value1 = -2047;
                 end
            end
            if (bdn2(i) == 1) value2 = 2047;
            else if (bdn2(i) == -1) value2 = -2047;

                 end
```

```
            end
        e1  =   value1 - rx1 * (2^(-12));
        e2  =   value2 - rx2 * (2^(-12));

    end
        j = j + 1;


    % Recording the error signals
        errors1(j) = e1 ;
        errors2(j) = e2 ;

    % Updating the coefficients according to LMS algorithm using the error
    % signals


        w11   =   w11  +  mu *  e1  *  rn11 ;
        w12   =   w12  +  mu *  e2  *  rn12 ;
        w21   =   w21  +  mu *  e1  *  rn21 ;
        w22   =   w22  +  mu *  e2  *  rn22 ;


end

% Plotting the learning curves for the system and calculating the MSE
% after convergence

t1 = (errors1) / max(abs(errors1));
t2 = (errors2) / max(abs(errors2));
error1_db = 10 * log10(abs(t1));
error2_db = 10 * log10(abs(t2));

SE1=mean(error1_db((length(error1_db)-10000):(length(error1_db))));
SE2=mean(error2_db((length(error2_db)-10000):(length(error2_db))));
MSE= mean ([SE1 SE2]);
subplot(211);
plot(error1_db);
xlabel('Samples');
ylabel('10 * log_{10} (Square Error 1)')

title(['MSE after convergence = ', num2str(MSE),' dB'])
```

```
subplot(212);


plot(error2_db);


xlabel('Samples');
ylabel('10 * log_{10} (Square Error 2)')
```

Appendix C

Sample SignalTap® II List File

This chapter shows a sample of the SignalTap II list file obtained from the SignalTap II logic analyzer software. This file is created using the 'Create Signal-Tap II list file' option and it is used to export the values recorded in SignalTap to MATLAB workspace. The details of this procedure are described in Chapter 4. The recorded values are imported in the MATLAB workspace as a single matrix. The legend information in the list file can be used to separate the desired signals from the matrix. For instance, key 0 in the legend indicates the error signal. Thus, in the MATLAB workspace the second column of the matrix is the error signal. Thus, all the desired signals can be separated from the matrix in similar way.

```
Signal Legend:

Key        Signal Name

0        = calculate_error:inst27|error
1        = calculate_error:inst27|error[13]
2        = calculate_error:inst27|error[12]
3        = calculate_error:inst27|error[11]
4        = calculate_error:inst27|error[10]
5        = calculate_error:inst27|error[9]
6        = calculate_error:inst27|error[8]
7        = calculate_error:inst27|error[7]
```

```
8        = calculate_error:inst27|error[6]
9        = calculate_error:inst27|error[5]
10       = calculate_error:inst27|error[4]
11       = calculate_error:inst27|error[3]
12       = calculate_error:inst27|error[2]
13       = calculate_error:inst27|error[1]
14       = calculate_error:inst27|error[0]
15       = calculate_error:inst28|error
16       = calculate_error:inst28|error[13]
17       = calculate_error:inst28|error[12]
18       = calculate_error:inst28|error[11]
19       = calculate_error:inst28|error[10]
20       = calculate_error:inst28|error[9]
21       = calculate_error:inst28|error[8]
22       = calculate_error:inst28|error[7]
23       = calculate_error:inst28|error[6]
24       = calculate_error:inst28|error[5]
25       = calculate_error:inst28|error[4]
26       = calculate_error:inst28|error[3]
27       = calculate_error:inst28|error[2]
28       = calculate_error:inst28|error[1]
29       = calculate_error:inst28|error[0]

Data Table:

Signals-> 0    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15   16 17 18 19 20 21 22 23 24 25 26 27 28 29

sample

0        -333 1 1 1 1 1 0 1 0 1 1  0  0  1  1  -333 1  1  1  1  1  0  1  0  1  1  0  0  1  1
1        -333 1 1 1 1 1 0 1 0 1 1  0  0  1  1  -333 1  1  1  1  1  0  1  0  1  1  0  0  1  1
2        -333 1 1 1 1 1 0 1 0 1 1  0  0  1  1  -333 1  1  1  1  1  0  1  0  1  1  0  0  1  1
3        -333 1 1 1 1 1 0 1 0 1 1  0  0  1  1  -333 1  1  1  1  1  0  1  0  1  1  0  0  1  1
4        -332 1 1 1 1 1 0 1 0 1 1  0  1  0  0  -332 1  1  1  1  1  0  1  0  1  1  0  1  0  0
5        -332 1 1 1 1 1 0 1 0 1 1  0  1  0  0  -332 1  1  1  1  1  0  1  0  1  1  0  1  0  0
6        -333 1 1 1 1 1 0 1 0 1 1  0  0  1  1  -333 1  1  1  1  1  0  1  0  1  1  0  0  1  1
7        -333 1 1 1 1 1 0 1 0 1 1  0  0  1  1  -333 1  1  1  1  1  0  1  0  1  1  0  0  1  1
8        329  0 0 0 0 0 1 0 1 0 0  1  0  0  1  -336 1  1  1  1  1  0  1  0  1  1  0  0  0  0
9        -340 1 1 1 1 1 0 1 0 1 0  1  1  0  0  -340 1  1  1  1  1  0  1  0  1  0  1  1  0  0
10       -328 1 1 1 1 1 0 1 0 1 1  1  0  0  0  -328 1  1  1  1  1  0  1  0  1  1  1  0  0  0
11       -321 1 1 1 1 1 0 1 0 1 1  1  1  1  1  -321 1  1  1  1  1  0  1  0  1  1  1  1  1  1
```

12	-321	1	1	1	1	1	0	1	0	1	1	1	1	1	1	-343	1	1	1	1	1	0	1	0	1	0	1	0	0	1
13	295	0	0	0	0	0	1	0	0	1	0	0	1	1	1	-277	1	1	1	1	1	0	1	1	1	0	1	0	1	1
14	-281	1	1	1	1	1	0	1	1	1	0	0	1	1	1	-344	1	1	1	1	1	0	1	0	1	0	1	0	0	0
15	342	0	0	0	0	0	1	0	1	0	1	0	1	1	0	-237	1	1	1	1	1	1	0	0	0	1	0	0	1	1
16	-265	1	1	1	1	1	0	1	1	1	1	0	1	1	1	-333	1	1	1	1	1	0	1	0	1	1	0	0	1	1
17	265	0	0	0	0	0	1	0	0	0	0	1	0	0	1	-430	1	1	1	1	1	0	0	1	0	1	0	0	1	0
18	217	0	0	0	0	0	0	1	1	0	1	1	0	0	1	-486	1	1	1	1	1	0	0	0	0	1	1	0	1	0
19	153	0	0	0	0	0	0	0	1	0	0	1	1	0	0	1	-449	1	1	1	1	1	0	0	0	1	1	1	1	1
20	-206	1	1	1	1	1	1	0	0	1	1	0	0	1	0	-433	1	1	1	1	1	0	0	1	0	0	1	1	1	1
21	134	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	-595	1	1	1	1	0	1	1	0	1	0	1	1	0	1
22	284	0	0	0	0	0	1	0	0	0	1	1	1	0	0	-453	1	1	1	1	1	0	0	0	1	1	1	0	1	1
23	-244	1	1	1	1	1	1	0	0	0	0	1	1	0	0	-307	1	1	1	1	1	0	1	1	0	0	1	1	0	1
24	-203	1	1	1	1	1	1	0	0	1	1	0	1	0	1	-215	1	1	1	1	1	1	0	0	1	0	1	0	0	1
25	-35	1	1	1	1	1	1	1	1	0	1	1	1	0	1	-317	1	1	1	1	1	0	1	1	0	0	0	0	1	1
26	174	0	0	0	0	0	0	1	0	1	0	1	1	1	0	-304	1	1	1	1	1	0	1	1	0	1	0	0	0	0
27	210	0	0	0	0	0	0	1	1	0	1	0	0	1	0	-332	1	1	1	1	1	0	1	0	1	1	0	1	0	0
28	103	0	0	0	0	0	0	0	1	1	0	0	1	1	1	-441	1	1	1	1	1	0	0	1	0	0	0	1	1	1
29	141	0	0	0	0	0	0	1	0	0	0	1	1	0	1	-526	1	1	1	1	0	1	1	1	1	1	0	0	1	0
30	84	0	0	0	0	0	0	0	1	0	1	0	1	0	0	-509	1	1	1	1	1	0	0	0	0	0	0	0	1	1
31	-61	1	1	1	1	1	1	1	1	0	0	0	0	1	1	-473	1	1	1	1	1	0	0	0	1	0	0	1	1	1
32	-49	1	1	1	1	1	1	1	1	0	0	1	1	1	1	-432	1	1	1	1	1	0	0	1	0	1	0	0	0	0
33	177	0	0	0	0	0	0	1	0	1	1	0	0	0	1	-337	1	1	1	1	1	0	1	0	1	0	1	1	1	1
34	102	0	0	0	0	0	0	0	1	1	0	0	1	1	0	-286	1	1	1	1	1	0	1	1	1	0	0	0	1	0
35	-101	1	1	1	1	1	1	1	0	0	1	1	0	1	1	-287	1	1	1	1	1	0	1	1	1	0	0	0	0	1
36	115	0	0	0	0	0	0	0	1	1	1	0	0	1	1	-282	1	1	1	1	1	0	1	1	1	0	0	1	1	0
37	9	0	0	0	0	0	0	0	0	0	0	1	0	0	1	-293	1	1	1	1	1	0	1	1	0	1	1	0	1	1
38	-103	1	1	1	1	1	1	1	0	0	1	1	0	0	1	-340	1	1	1	1	1	0	1	0	1	0	1	1	0	0
39	114	0	0	0	0	0	0	0	1	1	1	0	0	1	0	-354	1	1	1	1	1	0	1	0	0	1	1	1	1	0
40	-73	1	1	1	1	1	1	1	0	1	1	0	1	1	1	-367	1	1	1	1	1	0	1	0	0	1	0	0	0	1
41	-113	1	1	1	1	1	1	1	0	0	0	1	1	1	1	-370	1	1	1	1	1	0	1	0	0	0	1	1	1	0
42	-71	1	1	1	1	1	1	1	0	1	1	1	0	0	1	-342	1	1	1	1	1	0	1	0	1	0	1	0	1	0
43	-93	1	1	1	1	1	1	1	0	1	0	0	0	1	1	-348	1	1	1	1	1	0	1	0	1	0	0	1	0	0
44	-72	1	1	1	1	1	1	1	0	1	1	1	0	0	0	-348	1	1	1	1	1	0	1	0	1	0	0	1	0	0
45	-88	1	1	1	1	1	1	1	0	1	0	1	0	0	0	-354	1	1	1	1	1	0	1	0	0	1	1	1	1	0
46	-84	1	1	1	1	1	1	1	0	1	0	1	1	0	0	-313	1	1	1	1	1	0	1	1	0	0	0	1	1	1
47	-78	1	1	1	1	1	1	1	0	1	1	0	0	1	0	-325	1	1	1	1	1	0	1	0	1	1	1	0	1	1
48	-42	1	1	1	1	1	1	1	1	0	1	0	1	1	0	-354	1	1	1	1	1	0	1	0	0	1	1	1	1	0
49	24	0	0	0	0	0	0	0	0	0	1	1	0	0	0	-339	1	1	1	1	1	0	1	0	1	0	1	1	0	1
50	-35	1	1	1	1	1	1	1	1	0	1	1	1	0	1	-346	1	1	1	1	1	0	1	0	1	0	0	1	1	0
51	-12	1	1	1	1	1	1	1	1	1	1	0	1	0	0	-288	1	1	1	1	1	0	1	1	1	0	0	0	0	0
52	41	0	0	0	0	0	0	0	0	1	0	1	0	0	1	-291	1	1	1	1	1	0	1	1	0	1	1	1	0	1

53	26	0	0	0	0	0	0	0	0	0	1	1	0	1	0	−315	1	1	1	1	1	0	1	1	0	0	0	1	0	1
54	−42	1	1	1	1	1	1	1	1	0	1	0	1	1	0	−330	1	1	1	1	1	0	1	0	1	1	0	1	1	0
55	−59	1	1	1	1	1	1	1	1	0	0	0	1	0	1	−368	1	1	1	1	1	0	1	0	0	1	0	0	0	0
56	−21	1	1	1	1	1	1	1	1	1	0	1	0	1	1	−318	1	1	1	1	1	0	1	1	0	0	0	0	1	0
57	53	0	0	0	0	0	0	0	0	1	1	0	1	0	1	−291	1	1	1	1	1	0	1	1	0	1	1	1	0	1
58	98	0	0	0	0	0	0	0	1	1	0	0	0	1	0	−285	1	1	1	1	1	0	1	1	1	0	0	0	1	1
59	108	0	0	0	0	0	0	0	1	1	0	1	1	0	0	−212	1	1	1	1	1	1	0	0	1	0	1	1	0	0
60	126	0	0	0	0	0	0	0	1	1	1	1	1	1	0	−142	1	1	1	1	1	1	0	1	1	1	0	0	1	0
61	85	0	0	0	0	0	0	0	1	0	1	0	1	0	1	−144	1	1	1	1	1	1	0	1	1	1	0	0	0	0
62	56	0	0	0	0	0	0	0	0	1	1	1	0	0	0	−278	1	1	1	1	1	0	1	1	1	0	1	0	1	0
63	94	0	0	0	0	0	0	0	1	0	1	1	1	1	0	−276	1	1	1	1	1	0	1	1	1	0	1	1	0	0
64	134	0	0	0	0	0	0	1	0	0	0	0	1	1	0	−260	1	1	1	1	1	0	1	1	1	1	1	1	0	0
65	90	0	0	0	0	0	0	0	1	0	1	1	0	1	0	−371	1	1	1	1	1	0	1	0	0	0	1	1	0	1
66	91	0	0	0	0	0	0	0	1	0	1	1	0	1	1	−362	1	1	1	1	1	0	1	0	0	1	0	1	1	0
67	88	0	0	0	0	0	0	0	1	0	1	1	0	0	0	−364	1	1	1	1	1	0	1	0	0	1	0	1	0	0
68	48	0	0	0	0	0	0	0	0	1	1	0	0	0	0	−332	1	1	1	1	1	0	1	0	1	1	0	1	0	0
69	69	0	0	0	0	0	0	0	1	0	0	0	1	0	1	−357	1	1	1	1	1	0	1	0	0	1	1	0	1	1
70	72	0	0	0	0	0	0	0	1	0	0	1	0	0	0	−509	1	1	1	1	1	0	0	0	0	0	0	0	1	1
71	17	0	0	0	0	0	0	0	0	0	1	0	0	0	1	−441	1	1	1	1	1	0	0	1	0	0	0	1	1	1
72	−18	1	1	1	1	1	1	1	1	1	0	1	1	1	0	−520	1	1	1	1	0	1	1	1	1	1	1	0	0	0
73	−43	1	1	1	1	1	1	1	1	0	1	0	1	0	1	−475	1	1	1	1	1	0	0	0	1	0	0	1	0	1
74	−30	1	1	1	1	1	1	1	1	1	0	0	0	1	0	−366	1	1	1	1	1	0	1	0	0	1	0	0	1	0
75	−53	1	1	1	1	1	1	1	1	0	0	1	0	1	1	−406	1	1	1	1	1	0	0	1	1	0	1	0	1	0
76	−26	1	1	1	1	1	1	1	1	1	0	0	1	1	0	−366	1	1	1	1	1	0	1	0	0	1	0	0	1	0
77	−57	1	1	1	1	1	1	1	1	0	0	0	1	1	1	−293	1	1	1	1	1	0	1	1	0	1	1	0	1	1
78	−22	1	1	1	1	1	1	1	1	1	0	1	0	1	0	−301	1	1	1	1	1	0	1	1	0	1	0	0	1	1
79	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−309	1	1	1	1	1	0	1	1	0	0	1	0	1	1
80	49	0	0	0	0	0	0	0	0	1	1	0	0	0	1	−318	1	1	1	1	1	0	1	1	0	0	0	0	1	0
81	4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	−320	1	1	1	1	1	0	1	1	0	0	0	0	0	0
82	33	0	0	0	0	0	0	0	0	1	0	0	0	0	1	−294	1	1	1	1	1	0	1	1	0	1	1	0	1	0
83	65	0	0	0	0	0	0	0	1	0	0	0	0	0	1	−296	1	1	1	1	1	0	1	1	0	1	1	0	0	0
84	35	0	0	0	0	0	0	0	0	1	0	0	0	1	1	−306	1	1	1	1	1	0	1	1	0	0	1	1	1	0
85	5	0	0	0	0	0	0	0	0	0	0	0	1	0	1	−327	1	1	1	1	1	0	1	0	1	1	1	0	0	1
86	−17	1	1	1	1	1	1	1	1	1	0	1	1	1	1	−331	1	1	1	1	1	0	1	0	1	1	0	1	0	1
87	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−308	1	1	1	1	1	0	1	1	0	0	1	1	0	0
88	57	0	0	0	0	0	0	0	0	1	1	1	0	0	1	−293	1	1	1	1	1	0	1	1	0	1	1	0	1	1
89	95	0	0	0	0	0	0	0	1	0	1	1	1	1	1	−303	1	1	1	1	1	0	1	1	0	1	0	0	0	1
90	113	0	0	0	0	0	0	0	1	1	1	0	0	0	1	−263	1	1	1	1	1	0	1	1	1	1	1	0	0	1
91	123	0	0	0	0	0	0	0	1	1	1	1	0	1	1	−218	1	1	1	1	1	1	0	0	1	0	0	1	1	0
92	82	0	0	0	0	0	0	0	1	0	1	0	0	1	0	−205	1	1	1	1	1	1	0	0	1	1	0	0	1	1
93	72	0	0	0	0	0	0	0	1	0	0	1	0	0	0	−302	1	1	1	1	1	0	1	1	0	1	0	0	1	0

94	108	0	0	0	0	0	0	0	1	1	0	1	1	0	0	-301	1	1	1	1	1	0	1	1	0	1	0	0	1	1
95	126	0	0	0	0	0	0	0	1	1	1	1	1	1	0	-269	1	1	1	1	1	0	1	1	1	1	0	0	1	1
96	91	0	0	0	0	0	0	0	1	0	1	1	0	1	1	-338	1	1	1	1	1	0	1	0	1	0	1	1	1	0
97	108	0	0	0	0	0	0	0	1	1	0	1	1	0	0	-345	1	1	1	1	1	0	1	0	1	0	0	1	1	1
98	83	0	0	0	0	0	0	0	1	0	1	0	0	1	1	-327	1	1	1	1	1	0	1	0	1	1	1	0	0	1
99	50	0	0	0	0	0	0	0	0	1	1	0	0	1	0	-309	1	1	1	1	1	0	1	1	0	0	1	0	1	1
100	83	0	0	0	0	0	0	0	1	0	1	0	0	1	1	-343	1	1	1	1	1	0	1	0	1	0	1	0	0	1

Vita

Candidate's full name: Harshal Desai

University attended: University of Mumbai, B.E., 2003