**A VLSI Implementation of
an Adaptation Algorithm for a Pre-Emphasis
in a Backplane Transceiver**

By

Lei Lin

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Applied Science

Department of Electronics

Carleton University

Ottawa, Ontario, Canada

December 2003

© 2003 Lei Lin

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-89850-4

Our file Notre référence

ISBN: 0-612-89850-4

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this dissertation.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de ce manuscrit.

While these forms may be included in the document page count, their removal does not represent any loss of content from the dissertation.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

This thesis uses an adaptive pre-emphasis technique to compensate for channel inter-symbol interference (ISI) on a typical copper backplane. Functional models and Matlab code have been developed to verify the effectiveness and efficiency of the adaptation and equalization process using a sign-sign block least-mean-square (LMS) adaptive algorithm. The pre-emphasis structure covered in the simulation is a linear transversal type. The pre-emphasis and the adaptive algorithm are modeled using various simulation modules covering three different channel models. Both 2-level pulse amplitude modulation (PAM) and 4-level PAM are used in simulation. Using 4-PAM reduces the symbol rate by half compared to the conventional 2-PAM system. This symbol rate reduction relaxes the requirement for the channel bandwidth by a factor of two and allows the on-chip clock frequency to be reduced to half of the original.

Two different hardware structures of a sign-sign block LMS algorithm for an adaptive pre-emphasis in a copper backplane transceiver have been implemented in Verilog targeting the TSMC 0.18 μ m CMOS technology. One structure is the round-robin adaptation engine that updates tap coefficients every four blocks of data. This structure was proposed by Accelerant Networks in 2002. The other is the proposed structure, called the parallel adaptation engine. It updates tap coefficients every block of data. Simulation (or Synthesis) results show that digital standard cells can be directly applied to both structures to achieve 625 MHz timing constraints. A custom circuit is not needed to implement the digital adaptation algorithm for an analog adaptive pre-emphasis up to 625 MHz.

This research work has shown that the transmit adaptive pre-emphasis technique used here is a simple and cost-effective way to achieve a high data rate (up to 5 Gb/s) on typical copper backplanes as described above using 4-PAM modulation.

Acknowledgements

First and foremost, I would like to thank my supervisor Prof. Tad Kwasniewski for providing an interesting topic, constant direction, valuable advice, and most of all, for providing me the opportunity. He has appreciation and tremendous respect.

Thanks also to Prof. John Knight, Mr. Peter Noel, Mr. Gord Allan and Dr. Yuming Tao for their time and effort. Their efforts have substantially improved the quality of this thesis.

Thanks to all of my colleagues in the electronics research group. It is not without considerable regret that I refrain from attempting to list them all. However, there have been so many people who have contributed in so many different ways that a list somehow seems inappropriate. I would also like to gratefully acknowledge the support of Altera Corporation and CMC.

Great personal thanks goes to my mom who has provided me with much needed and appreciated support. Finally, I would like to thank my husband Rong for his great support and encouragement.

The information used in this thesis comes in part from the research program of Dr. Tad Kwasniewski and his associates in the VLSI in Communications group. The research results appearing in this thesis represent an integral part of the ongoing research program. All research results in this thesis including tables, graphs and figures but excluding the narrative portions of the thesis are effectively incorporated into the research program and can be used by Dr. Kwasniewski and his associates for educational and research purposes, including publication in open literature with appropriate credits. The matters of intellectual property may be pursued cooperatively with Carleton University and Dr. Kwasniewski, where and when appropriate.

Table of Contents

Chapter 1: Introduction.....	1
1.1 Motivation.....	1
1.2 Objective	4
1.3 Contributions	4
1.4 Thesis organization	5
Chapter 2: Background Theory	7
2.1 Introduction.....	7
2.2 Introduction to the Serial Link	7
2.3 Transmission Medium Limitations	9
2.3.1 Skin Effect	10
2.3.2 Dielectric Loss	12
2.4 Equalization Techniques	12
2.4.1 Transmit Pre-Emphasis.....	13
2.4.2 Receive Equalization	15
2.4.3 Transmit Pre-Emphasis and Receive equalization Combo	16
2.4.4 Comparison.....	16
2.5 Pulse Amplitude Modulation	17
2.6 Eye Diagram	19
2.7 Overview of Adaptive Filters	24
2.7.1 Introduction.....	24
2.7.2 The Least-Mean-Square (LMS) Algorithm	25
2.7.3 Channel Equalization	29
Chapter 3: Functional Simulation.....	30

3.1	Introduction.....	30
3.2	Circuit Description.....	30
3.2.1	Transceiver Block Diagram	30
3.2.2	Equalization Process	34
3.3	Sign-Sign Block LMS Algorithm	35
3.4	Matlab Simulation Model	37
3.4.1	Random Data Generator	37
3.4.2	Adaptive Pre-Emphasis.....	38
3.4.3	Coefficient Digital-to-Analog Convertors (CDACs).....	39
3.4.4	Channel Model.....	40
3.4.4.1	Channel Model One	40
3.4.4.2	Channel Model Two.....	40
3.4.4.3	Channel Model Three	41
3.4.5	AWGN Generator	41
3.4.6	Adaptation Engine	41
3.5	Simulation Results	45
3.5.1	Performance of the Adaptive Pre-Emphasis over Channel One	45
3.5.2	Performance of the Adaptive Pre-Emphasis over Channel Two	46
3.5.3	Performance of the Adaptive Pre-Emphasis over Channel Three	47
3.5.4	Effect of the Adaptation Step Size	48
3.5.5	Choose Data Block Size	48
3.6	Conclusions.....	67
Chapter 4: Custom Adaptation Engine Design		68
4.1	Introduction.....	68
4.2	Design Specification	68

4.3	Verilog HDL Implementation	69
4.3.1	Adaptation Algorithm	69
4.3.2	Digital Round-Robin Adaptation Engine	70
4.3.2.1	8-Bit Shift Register	73
4.3.2.2	4:1 Multiplexer	73
4.3.2.3	Sequencer	73
4.3.2.4	Up/Down Counter	75
4.3.2.5	Latch Bank	75
4.3.2.6	Message Generator	75
4.3.3	Digital Parallel Adaptation Engine	76
4.3.3.1	Tap Sequencer	79
4.4	Design Process	79
4.4.1	Register-Transfer Level (RTL) Model and Simulation	80
4.4.1.1	RTL Model	80
4.4.1.2	RTL Simulation	80
4.4.2	Logic Synthesis	81
4.4.2.1	Design Compiler - Synthesis	81
4.4.2.2	Test Compiler - Scan Insertion	81
4.4.2.3	Gate-level Simulation	82
4.4.3	Physical Design	83
4.4.3.1	Floorplanning, Placement, and Routing	83
4.4.3.2	Physical Verification	84
4.4.4	Comparison of the Adaptation Engines	88

Chapter 5: Conclusions and Future Work.....	90
5.1 Conclusions.....	90
5.2 Future Work	91
References.....	92
Appendix A: Schematics Views of Top Modules	95
Appendix B: Pre-Layout Timing Reports	98
Appendix C: Post-Layout Timing Reports.....	101

List of Tables

TABLE 4.1: Comparison of Synthesis Results	88
--	----

List of Figures

Figure 2.1:	Serial Link Components	8
Figure 2.2:	Lumped RLGC model of a transmission line	10
Figure 2.3:	Frequency response of the copper cable	13
Figure 2.4:	Equalization Techniques	13
Figure 2.5:	400ps Pulse response of a 1m long, 1.2mm wide stripline channel	14
Figure 2.6:	Block implementation of pre-emphasis circuit.....	15
Figure 2.7:	The conditions for modulation selection	19
Figure 2.8:	(a) Distorted binary wave with noise, but no ISI, (b) Eye pattern	20
Figure 2.9:	(a) Distorted binary wave with noise and ISI, (b) Eye Pattern	20
Figure 2.10:	Eye pattern features	20
Figure 2.11:	Eye diagram for an ideal binary system.....	23
Figure 2.12:	Eye diagram for an ideal 4-PAM system	23
Figure 2.13:	Eye diagrams with channel distortion.....	24
Figure 2.14:	Typical adaptive filter as a two-input, two-output system.....	25
Figure 2.15:	N-tap transversal adaptive filter.....	26
Figure 3.1:	Simplified transceiver block diagram	31
Figure 3.2:	4-PAM signal slicing.....	32
Figure 3.3:	Pre-emphasis architecture with CDACs	33
Figure 3.4:	Equalization Path for startup and normal equalization	35
Figure 3.5:	Simulation block diagram	37
Figure 3.6:	Functional block diagram of an adaptive pre-emphasis filter.....	39
Figure 3.7:	Impulse response of the first channel	42
Figure 3.8:	Channel loss of the first channel.....	42

Figure 3.9:	Impulse response of the second channel.....	43
Figure 3.10:	Frequency response of the second channel.....	43
Figure 3.11:	Impulse response of the third channel	44
Figure 3.12:	Frequency response of the third channel	44
Figure 3.13:	Design flowchart for the first channel model	45
Figure 3.14:	Design flowchart for the second/third channel model	47
Figure 3.15:	Eye diagram without pre-emphasis at a rate of 2.5GBd/s over channel one .	50
Figure 3.16:	Eye diagram with pre-emphasis at a rate of 2.5GBd/s over channel one ..	51
Figure 3.17:	Optimal coefficients of pre-emphasis at a rate of 2.5GBd/s over channel one	52
Figure 3.18:	Mean square error vs. number of iterations at a rate of 2.5GBd/s over channel one	53
Figure 3.19:	Eye diagram without pre-emphasis at a rate of 2.5GHz over channel two	54
Figure 3.20:	Eye diagram with pre-emphasis at a rate of 2.5GBd/s over channel two ..	55
Figure 3.21:	Optimal coefficients of pre-emphasis at a rate of 2.5GBd/s over channel two	56
Figure 3.22:	Mean square error vs. number of iterations at a rate of 2.5GHz over channel two	57
Figure 3.23:	Eye diagram without pre-emphasis at a rate of 5GBd/s over channel three ..	58
Figure 3.24:	Eye diagram with pre-emphasis at a rate of 5GBd/s over channel three ...	59
Figure 3.25:	Optimal coefficients of pre-emphasis at a rate of 5GBd/s over channel three	60
Figure 3.26:	Mean square error vs. number of iterations at a rate of 5GBd/s over channel three	61

Figure 3.27:	200ns Pulse response for the channel three	62
Figure 3.28:	Eye diagram for adaptation step size = 0.01	62
Figure 3.29:	MSE for adaptation step size = 0.01	63
Figure 3.30:	Eye diagram for adaptation step size = 0.02.....	63
Figure 3.31:	MSE for adaptation step size = 0.02.....	64
Figure 3.32:	Eye diagram for adaptation step size = 0.1	64
Figure 3.33:	MSE for adaptation step size = 0.1	65
Figure 3.34:	Coefficients vs. data block size for the first channel	65
Figure 3.35:	Coefficients vs. data block size for the second channel.....	66
Figure 3.36:	Coefficients vs. data block size for the third channel	66
Figure 4.1:	Block diagram of round-robin adaptation engine	72
Figure 4.2:	Block diagram of the 8-bit shift register	73
Figure 4.3:	Timing diagram of the tap sequencer sub-block.....	74
Figure 4.4:	Block diagram of message generator.....	76
Figure 4.5:	Block diagram of parallel adaptation engine	78
Figure 4.6:	Digital IC Design Flow.....	79
Figure 4.7:	RTL simulation result of the round-robin adaptation engine.....	85
Figure 4.8:	Gate-level simulation result of the parallel adaptation engine	86
Figure 4.9:	Gold-netlist simulation result of the parallel adaptation engine	87
Figure 4.10:	Layouts of two adaptation engines	88

List of Abbreviations

AC	Alternating Current
ASIC	Application Specific Integrated Circuit
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CDAC	Coefficient Digital to Analog Conversion
CMC	Canadian Microelectronics Corporation
CML	Current Model Logic
CMOS	Complementary Metal Oxide Semiconductor
CMOSP18	0.18um CMOS Process provided by TSMC
CTGen	Clock Tree Generator
DAC	Digital to Analog Conversion
DC	Direct Current
DEF	Design Exchange Format
DFII	Design Framework II
DFT	Design For Testability
DRC	Design Rule Check
FF	Flip Flop
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
FR-4	Flame retardant version of the old G10 epoxy laminate with glass fibre reinforcement

FTF	Fast Transversal Filters
GDSII	Graphic Design System II
GTECH	A set of generic logic cells used as the target for the first pass of Synopsys' logic synthesis program
HDL	Hardware Description Language
HSBI	High Speed Backplane Initiative
ISI	Inter-Symbol Interference
LMS	Least Mean Square
LVS	Layout Versus Schematic
MMSE	Minimum Mean Square Error
MSB	Most Significant Bit
MSE	Mean Square Error
Mux	Multiplexer
NRZ	Non Return Zero
PAM	Pulse Amplitude Modulation
2-PAM	2-Level Pulse Amplitude Modulation
4-PAM	4-Level Pulse Amplitude Modulation
PCB	Printed Circuit Board
PDP	Physical Design Planner - a Cadence tool for digital layout
PLL	Phase-Locked Loop
PRBS	Pseudo Random Binary Sequence
RLS	Recursive Least Square

RTL	Register Transfer Level
SD-LMS	Sign Data Least Mean Square
SE-LMS	Sign Error Least Mean Square
SERDES	Serializer/Deserializer
SNR	Signal to Noise Ratio
SS-LMS	Sign-Sign Least Mean Square
TSMC	Taiwan Semiconductor Manufacturing Company
UTP	Unshielded Twisted Pair
VLSI	Very Large Scale Integration
XOR	Exclusive OR Gate
ZF	Zero Forcing

1.1 Motivation

The ever-increasing need for high data rate in communication systems is driving data transmission within networking equipment to a rate of multi-gigabit per second. Copper backplane based serial transmission is one of the most commonly used techniques in high-speed transmission due to its lower cost, reduced complexity and high reliability.

Although a copper backplane offers significant advantages in high-speed transmission, it does have some technical challenges, mainly because of distortions due to skin effect, dielectric loss, and reflections, which are frequency dependent. Overall, these distortions often cause severe signal degradation in copper backplanes. This results in “eye-closure” at the receive end. A smaller eye opening makes signal reception and recovery more difficult for the receiver, which usually results in higher bit-error-rate (BER). Higher BER causes more packet loss and reduces the effective bandwidth.

The simplest way to achieve higher data rates is to actively compensate for the channel transfer function. For a multi-gigabit per second transmission channel, analog equalization is an attractive alternative to digital approaches. The main advantages of analog equalization are lower power, less silicon area, and most importantly, capability of performing equalization at high speed.

Pre-emphasis is a unique signal improving technique that opens eye patterns at the far end of a transmission line. Pre-emphasis boosts the higher frequency components of the signal by increasing the output level for the first symbol in a series of the same symbols. Subsequent symbols in the sequence are reduced to lower power levels. Higher frequencies tend to attenuate faster than lower frequency components. The idea is to boost, right from the transmitter side, the high frequency components of the signal, knowing that it will attenuate more by the time it reaches the receiver. Pre-emphasis does not waste or burn any bit time, the bus is still 100% efficient for data transfer. Every bit sent is valid information.

Basically there are two types of pre-emphasis: fixed (tuning) and adaptive. If channel characteristics are known a priori, fixed pre-emphasis can be used. However, if channel characteristics change over time, such as varying printed-circuit-board (PCB) trace lengths and trace impedances unique to each individual channel in the backplane, adaptive pre-emphasis would be a good option. The ability to self-optimize pre-emphasis coefficients for each individual interconnect within any given systems results in significantly improved robustness, reduced characterization and calibration efforts, and high-speed operation.

An adaptive pre-emphasis is accompanied by an adaptation engine that performs the adaptation algorithm at receive side, where quality of the signal can be directly measured. Once an update is completed, adjustments for the taps are communicated back to the far end transmitter where they are applied via a slow transparent link. The coefficients are continually updated until the optimal performance is reached. An adaptive pre-emphasis compensates for not only the distortion caused by packages, connectors and backplane traces, but also the changes in environmental conditions, such as power supply drift, component aging, and temperature variations.

Although digital circuits in new process technologies will eventually reach very high speed, there will be a frequency limit beyond which analog adaptive equalization is more efficient. In order to keep analog circuit design as simple as possible, equalizers usually have a few adapted coefficients to adjust the amount of high frequency peaking in the filter transfer function. Since the connections are fixed, faster adaptation is generally not required so that the adaptive algorithm need not be implemented with high speed analog circuits [4]. The best combination is often to use analog circuits for high speed signal path, and slower digital circuits for adaptation.

Although several other algorithms with superior convergence properties exist, such as recursive-least-square (RLS) algorithm and fast-transversal-filter (FTF) algorithm, the most popular adaptive algorithm for high-speed integrated filters is the least-mean-square (LMS) algorithm. This is due to its straightforward and robust digital hardware implementation. Unfortunately implementation of the LMS algorithm in analog adaptive filters is challenging. DC offset on analog signals can prevent accurate convergence [9] [10] and significantly more complicated analog hardware may be required to obtain gradient signals [11]. In practice, an efficient and simple algorithm for adapting high-frequency filters is required. At very high data rates, a minimal number of calculations can be done in real time, therefore a sign-sign block LMS algorithm is an excellent choice for the adaptation engine of the adaptation and equalization process. This is because it can be implemented in hardware with relatively low complexity. Using the sign-sign block LMS approach, only the sign of the error, the difference between the filter output and the desired output, and the sign of the filter output are multiplied, allowing complicated digital multipliers to be replaced by XOR gates. A drawback of this algorithm is a longer convergence time than with the conventional LMS algorithm. This is not an issue for this design during system normal operation as transmission line characteristics change slowly with time. But at system initialization, we want the coefficient convergence speed to be as faster as possible.

1.2 Objective

The objectives of this thesis can be stated as follows:

1. Design a simulator using Matlab to verify the effectiveness and efficiency of the adaptation and equalization process. It consists of a pre-emphasis, a channel, an adaptation engine and a coefficient digital-to-analog convertor. The limitations, such as finite tap weight and resolution, and finite transmit and receive bandwidth, are considered in the simulator.

2. Implement two different hardware structures of the sign-sign block least-mean-square (LMS) algorithm for an adaptive pre-emphasis in a copper backplane transceiver using Verilog targeting the TSMC 0.18um CMOS technology. One structure is the round robin adaptation engine. The other is the parallel adaptation engine. The round-robin adaptation engine requires four blocks of data in order to update tap coefficients and the parallel adaptation engine only utilizes one block of data to update tap coefficients.

3. Verify that the use of standard cells in a 0.18um CMOS technology will allow a 500 MHz on-chip clock and compare two implementations.

1.3 Contributions

The contributions of this research are outlined as follows:

1. A Matlab based simulator has been developed with three different channel models. Simulated results show significant improvement in the eye openings using adaptive pre-emphasis technique. Also, the simulation proved that the sign-sign block LMS algorithm is a simple and efficient adaptation algorithm, which effectively compensates for inter-symbol interference (ISI) caused by a copper backplane channel.

2. Two different hardware structures for the adaptation algorithm circuit were implemented in Verilog HDL targeting the TSMC 0.18um CMOS technology. The design process included RTL coding, logic synthesis, scan chain insertion, floorplanning, place-

ment, clock tree generation, routing, timing analysis, DRC, LVS, etc. Results show that the parallel adaptation structure is four times faster than the round-robin adaptation structure in terms of coefficient update rate with comparable hardware. Both circuits prove that standcells from a CMOSP18 (0.18 μ m CMOS process) technology can be used to achieve 625 MHz timing constraints. This means that custom circuits are not required for the implementation of the digital adaptation engine for the analog adaptive pre-emphasis at rates up to 625 MHz. Both designs were submitted to CMC (Canadian Microelectronics Cooperation) for fabrication.

1.4 Thesis organization

The topics addressed by this thesis are outlined in the following paragraphs:

Chapter 1 is the introduction, which outlines the motivation and scope of this research work.

Chapter 2 aims to provide an overview of related topics in serial link backplane applications. This chapter covers serial link structure and limitations of the backplane environment. Introduction to equalization techniques, multi-level signalling and eye diagram are also given in the chapter. The last part of this chapter gives the fundamental concepts of the adaptive filters which will be used in the following chapters.

Chapter 3 presents the Matlab simulation model for the adaptive pre-emphasis. The circuit description and the equalization process are introduced. A detailed simulation model is described. Finally, the simulation results are given.

Chapter 4 details the implementation of two custom adaptation engine circuits targeting CMOSP18 technology provided by CMC. One is a round-robin adaptation engine which updates the tap coefficients every four blocks of data. Another is a parallel adaptation engine which updates the tap coefficients every block of data. The comparisons between two hardware structures are discussed.

Chapter 5 draws the conclusions of the overall research work and provides some directions for the future research work.

2.1 Introduction

This chapter provides some fundamental background theory necessary for a proper understanding and appreciation of the remaining chapters. A point-to-point serial link system is introduced in Section 2.2. The limitations of a copper transmission medium are illustrated in Section 2.3. Channel equalization techniques for reliable high speed data are addressed in Section 2.4. A multi-level modulation technique which transfers more data within the same bandwidth are discussed in Section 2.5. The concept of an eye diagram for measuring distortions in a transmission system is introduced in Section 2.6. The last section of this chapter reviews some important aspects of adaptive filtering and channel equalization.

2.2 Introduction to the Serial Link

Backplanes and transceivers are used in almost all networking systems, including switches, routers, cross connects, and optical networking equipment, etc. As interconnect speed increases, the challenges in designing reliable, high-performance backplane interconnections grow rapidly.

There are two dominant approaches for high-speed signalling: shared buses (multi-point) and point-to-point links. The shared bus concept describes a backplane in which a

signal has more than two connections, with each connection often able to transmit or receive signals. Shared buses have typically been used for short-range interconnections with rates up to 100-200Mbps, such as in multi-processor [12], [13], [14], processor-to-memory interfaces [15], [16], and network switches [17]. The point-to-point approach describes a backplane in which a signal has only two connections, i.e., a transmit and a receive connection. Point-to-point is often favored for high-speed design, as it increases the total bandwidth per channel and therefore the total chip throughput for a fixed package pin count.

A typical high-speed serial link is comprised of three primary components: a transmitter, a channel, and a receiver, as shown in Figure 2.1. A transmitter converts a digital bit-stream to an analog waveform, which is then sent with a needed timing reference on a physical channel. A channel is a medium on which data are propagated. This channel is terminated with its characteristic impedance at both ends. A receiver termination absorbs the incident wave, preventing any reflections. The source termination makes the system more tolerant to crosstalk and impedance discontinuities by absorbing reflected waves. The channel can attenuate the signal and introduce noise. To achieve very high data rates and reduce noise, a channel with low attenuation and distortion at high frequencies is required. A receiver is a circuit that amplifies and samples the received waveform. A timing-recovery circuit is used to properly place the sampling strobe.

Figure 2.1: Serial Link Components

Clock generation and timing recovery are very important to accurate signal transmission and reception. The jitter of the clock generator in the transmitter is directly transmitted to the output signal, where it degrades the transmitted waveform. The timing recovery circuit, which is often embedded in the receiver, adjusts the phase of the clock to sample the in-coming signal at the optimal position.

The maximum link distance for a copper medium link is determined by the medium attenuation. The link distance performance can be improved by employing transmit pre-emphasis and/or receive equalization. Both boost the gain at higher frequencies to compensate for medium attenuation.

2.3 Transmission Medium Limitations

As depicted in Figure 2.1, a channel is an electrical path between a transmitter and a receiver. PCB traces and cables are two types of transmission media commonly used, while connectors and vias can be thought of as transition mechanisms between segments of transmission media. At speeds of gigabit-per-second or higher, printed-circuit-board (PCB) traces and cables behave as transmission lines that store and propagate signal energy. An ideal transmission line has zero resistance, but a practical transmission line always has a small series resistance. The nonzero resistance of a practical transmission line causes both attenuation (loss) and distortion in propagating signals. Signal attenuation means that signals shrink progressively as they move along the transmission line. Signal distortion means signals of different frequencies are attenuated and phase-shifted by different amount as they propagate. A transmission line model including a series of lumped RLGC elements is depicted in Figure 2.2. The loss in a transmission line is primarily due to the serial resistive component of the copper (R) and parallel conductive component of the dielectric (G). The relationship among attenuation, phase shift and frequency at a point X inches from the head of the cable can be described by Equation (2.1) [5]:

$$H_X(\omega) = e^{-X\sqrt{(R+j\omega L)(G+j\omega C)}} \quad (2.1)$$

where R = transmission line resistance, ohms/inch

L = transmission line inductance, H/inch

C = transmission line capacitance, F/inch

G = transmission line conductance, mhos/inch

X = length of cable, inch

$H(\omega)$ = complex function specifying amplitude and phase response of
transmission line at frequency $\omega = 2\pi f$

This equation only applies to infinite or perfectly terminated transmission lines. The higher the frequency and the longer the PCB trace, the greater the signal loss. This is caused by the bandwidth limitation of the channel. When the data rate is higher than the bandwidth of the channel, a loss of the signal will occur, which causes eye-closure at the receive side.

Figure 2.2: Lumped RLGC model of a transmission line

2.3.1 Skin Effect

The frequency dependent losses across a backplane are primary due to skin effect and dielectric losses. Skin effect is a tendency for alternating current (AC) to flow mostly near the outer surface of a solid electrical conductor at frequencies above the audio range. The effect becomes more and more apparent as the frequency increases. At low frequency, the distribution of current density inside a conductor is uniform. Looking at a cross section

of a conducting wire, there is as much current flowing in the center as there is near the edge. At high frequencies the current density grows larger near the surface of the wire, and almost no current flows in the center. Current fills the wire evenly at low frequency and flows only near the surface at high frequencies. The higher the data rate, the narrower the current path and therefore, the higher the resistance of the trace. This effect can be modeled as an increase in series resistance of the transmission line as depicted in Figure 2.2.

At high frequencies, the average depth of current penetration, called the skin depth, is quite shallow. The average current depth (skin depth) is a function of frequency ω (in rad/s), which is defined as [5]:

$$\text{Skin depth} = \sqrt{\frac{2\rho}{\omega\mu}} \quad (2.2)$$

where μ is the magnetic permeability of the conductor, ρ is the volume resistivity. The conductor's apparent resistance is inversely proportional to the depth of current flow (the skin depth) and the skin depth is inversely proportional to the square root of frequency. Hence, the AC resistance of conductors grows proportionally to the square root of frequency.

Skin depth is a material property that varies with bulk conductivity of the conducting medium. It is not a function of conductor shape [5]. At low frequency when the skin depth is comparable to, or greater than, the wire radius, the current distribution fills the wire evenly, only the total DC resistance of the wire needs to be considered. When the skin depth is less than the wire radius, the resistance per inch goes up by the square root of the frequency.

Transmission loss, in decibels, is proportional to resistance as [5]:

$$\text{Loss per inch} = 4.34 \left[\frac{R}{\sqrt{L/C}} \right] dB \quad (2.3)$$

where R is the transmission line resistance (in ohms/inch), L is the transmission line inductance (in H/inch), C is the transmission line capacitance (in F/inch). Resistance is proportional to the square root of frequency. Therefore, the attenuation in decibels must be proportional to the square root of frequency.

2.3.2 Dielectric Loss

Dielectric loss is defined as a loss of energy which eventually produces a rise in temperature of a dielectric placed in an alternating electrical field. This loss can be modeled as a conductance G between the signal wire and ground as depicted in Figure 2.2. When a piece of glass epoxy printed circuit board material used as insulating dielectric of a transmission line, dielectric losses translate into signal attenuation. The higher the dielectric loss, the more attenuation.

Dielectric loss is a function of frequency. The glass epoxy material commonly used for printed circuit boards (FR-4) has negligible loss for digital applications below 1 GHz. At higher frequencies, however, the dielectric loss of FR-4 grows. The dielectric loss, G , typically increases linearly with frequency [3].

Both skin effect and dielectric loss can cause inter-symbol interference (ISI), because the attenuation of the signal prevents it from reaching the full strength within its symbol period, causing it to spread into the next symbol. A typical frequency response of a copper cable accounting for both skin effect and dielectric loss is depicted in Figure 2.3 [3]. At low frequencies, skin effect is the dominant loss. However, at frequencies higher than 2.5 GHz, dielectric losses begin to dominate.

2.4 Equalization Techniques

Data on a backplane are communicated by transmitting an electrical signal from one line card to another via copper traces. Backplane traces have a low-pass frequency characteristic that reduces the gain of the high-frequency signal. If we can design a circuit

that produces the reverse behavior of the channel frequency response to compensate for the attenuation, the signal can be restored. Both transmit and receive equalization have been proven to be effective in dealing with the loss associated with the channel. Choosing the right equalization technique is heavily dependent upon the application. In general, there are three possible ways to overcome transmission losses, as depicted in Figure 2.4. They are transmit pre-emphasis, receive equalization, or a combination of the two. All of these methods require extra circuitry either in the transmitter, the receiver or both, to compensate for high frequency losses.

Figure 2.3: Frequency response of the copper cable

Figure 2.4: Equalization Techniques

2.4.1 Transmit Pre-Emphasis

When the data rate is higher than the bandwidth of the channel, the binary data may not complete a transition within a symbol interval. It will spread into adjacent symbols and cause inter-symbol interference (ISI). Figure 2.5 depicts a time-domain response

of a 1m long, 1.2mm wide stripline to an 400ps ideal trapezoidal pulse. The pulse suffers more attenuation in its high frequency components (transitions) than in its low frequency components. The long tail of the pulse caused by group delay distortion and lower frequency attenuation spreads into several symbols and corrupts the following transmitted symbols. It reduces their effective width and amplitude.

Figure 2.5: 400ps Pulse response of a 1m long, 1.2mm wide stripline channel

Pre-emphasis operates by boosting the high frequency energy whenever there is a transition in the data. It adds additional output current during the transition of the bit. This tends to speed up the edge rate and also provides a bit of over-shoot to the signal at the driver output. Since high frequency components are attenuated more than lower frequency components, an equalized eye will be presented at the receiver side. This enables the receiver to recover the signal more easily.

One example of a pre-emphasis circuit block implementation is depicted in Figure 2.6. This circuitry acts as a two-tap Finite-Impulse Response (FIR) filter. This cir-

cuitry compares a previously-transmitted data bit with the current data. If the two bits are at the same level, then the current bit is transmitted at the normal level. Otherwise, the current bit is transmitted at a higher magnitude. To achieve optimized pre-emphasis effects, more than one programmable tap is required to further cancel the ISI. The best pre-emphasis effects are achieved by overcompensating for the ISI with the first tap, then using the second tap to cancel “the extra” compensation of the first tap.

Too much pre-emphasis can also be of concern. A large signal will generate more noise and crosstalk, and increase power dissipation. By using an adaptive pre-emphasis technique, signal quality can be measured at the receive side and the proper amount of pre-emphasis can be determined automatically. This research focuses on the digital adaptation algorithm hardware implementation.

Figure 2.6: Block implementation of pre-emphasis circuit

2.4.2 Receive Equalization

Receive equalization may also be used to open the eye of a backplane connection. The goal of the equalizer block is to provide an inverse transfer characteristics of the channel to fully recover the signal. Receive equalization provides this functionality in the receiver to help overcome high frequency signal losses of the transmission medium. In receive equalization, the signal is transmitted without any special processing, but compen-

sated at the receiver by using filtering techniques. In general, receive equalization is implemented using a high-pass filter. Receive equalization can be either analog or digital. Analog techniques use linear filter circuits, such as RC filters, to reverse the low pass effect of the channel with a high pass filter to recover and restore the original signal. Its net effects are similar to that of pre-emphasis in opening the eye. Digital techniques use digital filters and analog to digital conversion for equalization. A digital equalizer is equivalent to applying pre-emphasis techniques at the receive end of the channel.

2.4.3 Transmit Pre-Emphasis and Receive equalization Combo

The equalization techniques described above have a net effect of flattening the channel response, which means an eye-opening in time domain. In some cases, however, the system requires a combination of transmit pre-emphasis and receive equalization.

2.4.4 Comparison

The transmit pre-emphasis and the receive equalization essentially accomplish similar functions, but they are implemented differently. Choosing the right equalization technique depends on the application. Pre-emphasis is preferred over receive equalization if crosstalk is not a factor in the transmission line. Otherwise pre-emphasis will make crosstalk more pronounced due to amplified transmitted signals. Receive equalization is preferred over transmit pre-emphasis if the signal-to-noise ratio (SNR) is not a factor in the transmission line. Otherwise, receive equalization will amplify the noise transmitted across the transmission line at the receiver. The transmit pre-emphasis is simple and effective for very high data rate design. However, receive equalization also has an advantage over pre-emphasis, i.e., it does not require reliable feedback.

Receive equalization is well suited for certain applications, such as 100Base-T and Gigabit Ethernet over category 5 UTP (unshielded twisted pair) cables. Transmit pre-emphasis is a better option for backplane applications. This is because the data rates in

multi-gigabit backplane applications are much higher than the Fast Ethernet or Gigabit Ethernet and the transmission is on the PCB traces as opposed to UTP cable [20].

In backplane applications, transmit pre-emphasis offers many advantages over receive equalization, such as lower power consumption, better performance and better interoperability.

At multi-gigabit per second data rate, transmit pre-emphasis is typically implemented by current summation in the high-speed transmit current mode driver using Current Model Logic (CML) techniques. This method results in a small power overhead compared to a normal driver without pre-emphasis. Receive equalization however requires dedicated high-power circuits to perform equalization, which consumes additional power on top of the power dissipated in the current mode transmit driver.

In receive equalization, the receiver bandwidth needs to be significantly higher than the baud rate in order to equalize the signal adequately. This may require special process technologies with high cut-off frequencies. Using comparable CMOS process technologies for receive equalization will result in a lower level of equalization due to lack of higher cut-off frequency.

Most vendors of multi-gigabit SERDES (serializer/deserializer) components use the transmit pre-emphasis method. Therefore, transmit pre-emphasis will normally offer better interoperability for components from different vendors.

2.5 Pulse Amplitude Modulation

The ability to effectively double the data transmission rate for a given frequency by using 4-level signaling has proven to be an essential solution for applications where the transfer characteristics and signal-to-noise ratios drop off so dramatically at high frequency. As the name implies, M-level pulse amplitude modulation (PAM) conveys information by assigning $k = \log_2(M)$ bits to a set of M discrete amplitudes of a transmitted

signal. The simplest multi-level transmission scheme is M -level pulse amplitude modulation which is a one-dimensional signaling scheme described mathematically by

$$s_i(t) = a_i\phi(t), \quad i = 1, 2, \dots, M, \quad 0 \leq t \leq T \quad (2.4)$$

where

$$a_i = (2i - 1 - M)A, \quad i = 1, 2, \dots, M \quad (2.5)$$

$\phi(t)$ is a real-valued signal pulse whose shape influences the spectrum of the transmitted signal, and A is half of the distance between adjacent signal amplitude.

For a given data rate, M -PAM modulation reduces the effective symbol-rate by a factor of $k = \log_2(M)$ compared to a conventional binary (2-PAM) system. This is the rate at which changes occur in the amplitude of the carrier to reflect the transmission of new information. The symbol rate reduction reduces not only the ISI in the channel, but also the maximum required on-chip clock frequency [3]. The bandwidth of the transmission is predominantly determined by the symbol rate rather than the aggregate data throughput. This lowers the bandwidth requirements of the transmitter/receiver.

For the backplane applications, there remains the need for both 2-PAM and 4-PAM schemes. The use of 2-PAM (NRZ) has proven effective for backplane serial link transmission at frequency ranges of 1 to 3 Gbps [21]. However, when the backplane frequency range goes higher toward 5 to 10 Gbps, it may need 4-PAM on some channels, and 2-PAM on others. The criterion for determining whether to use 2-PAM or 4-PAM is based on the difference in signal attenuation at the two frequencies, as depicted in Figure 2.7 [21]. If the amplitude at frequency “ f ” is less than 3x that of “ $2f$ ”, the 2-PAM is the preferred signaling choice, otherwise 4-PAM is the correct choice.

In this research, both 2-PAM and 4-PAM signals were used for the functional simulation in Matlab. Actually, the adaptation algorithm used in this research is compatible with both bi-level and multi-level signaling.

Figure 2.7: The conditions for modulation selection

2.6 Eye Diagram

The eye pattern is an experimental tool to evaluate the combined effect of receiver noise and ISI on the overall system performance in an operational environment. It is defined as the synchronized superposition of all possible realizations of the signal of interest viewed with a particular signalling interval, as depicted in Figure 2.8 (b) and Figure 2.9 (b). An eye pattern can be viewed on an oscilloscope if the data rate is used to trigger the horizontal sweep while the bit stream is sampled and applied to the vertical input. The horizontal width of the lines on the scope include the jitter (phase noise) and the rise and fall times of the data pulses can be measured from the vertical "crossings". This pattern is called the eye as it appears like an eye between a pair of rails.

Figure 2.8: (a) Distorted binary wave with noise, but no ISI, (b) Eye pattern

Figure 2.9: (a) Distorted binary wave with noise and ISI, (b) Eye Pattern

Figure 2.10: Eye pattern features

An eye pattern provides a great deal of useful information about the performance of a data transmission system, as illustrated in Figure 2.10. The key features are:

- *Eye opening* - the minimum separation at the sampling times between two signals represents different adjacent values.
- *Noise margin* - another term used to quantify the eye opening. It is a measure of the shortest distance from an eye boundary, at the sampling time, to the midpoint between ideal received signal levels.
- *Eye width* - (the minimum width of the eye at the midpoint between ideal signal levels) the time interval over which the received signal can be sampled without error due to ISI.
- *Rate of closing of the eye* - the slope of the eye boundary in the vicinity of the optimum sampling time.
- *Zero-crossing distortion* - the amount of distortion in the vicinity of the zero crossing (width of the eye boundaries at the zero crossings).

The eye opening or noise margin indicates the sensibility of the regenerator to crosstalk and interference. It can be used to estimate the regenerator error rate. The better the design of the equalizer, the larger the noise margin. A wide open eye indicates that the amplitude and phase distortions introduced by the channel have been minimized.

While the noise margin indicates the margin against signal amplitude disturbances, the eye width indicates the regenerator margin for the timing variations. The optimum sampling instants are located where the eye opening is a maximum. Variations of the sampling instants from the optimum will reduce the noise margin and, as the edges of the eye are approached, the error rate will increase rapidly.

The sensitivity of the regenerator system to timing error is affected by the rate of closing of the eye. If the slope is large, then small timing variations can cause large reductions in the noise margin. If the rate of closure of the eye is low, small timing variations have negligible effect on the bit error rate.

Zero crossing distortion is the variation in the zero crossings of the eye. Zero crossings correspond to the time instants at which the eye waveforms pass through zero volts. Zero crossing distortion indicates a likely extent of variations at the output of the timing recovery circuits which set the sampling instants.

In order to optimize eye opening, the distortion at sampling rate and distortion of zero-crossing must be minimized.

In the case of an M -ary system, an eye pattern contains $(M-1)$ eye openings stacked up vertically one on the other, where M is the number of discrete amplitude levels used to construct the transmitted signal. In a strictly linear system with truly random data, all these eye openings would be identical. Figure 2.11 and Figure 2.12 show the eye diagrams for a baseband PAM transmission system using $M = 2$ and $M = 4$ respectively, under ideal conditions, i.e, no channel noise and no bandwidth limitation. The eye diagrams with channel distortion (both channel noise and bandwidth limitation) are depicted in Figure 2.13.

An open eye pattern corresponds to minimal signal distortion. Distortion of the signal waveform due to inter-symbol interference and noise appears as closure of the eye pattern. The symbol error probability of the system can be estimated from eye patterns.

Figure 2.11: Eye diagram for an ideal binary system

Figure 2.12: Eye diagram for an ideal 4-PAM system

Figure 2.13: Eye diagrams with channel distortion

2.7 Overview of Adaptive Filters

2.7.1 Introduction

Filtering is a signal processing operation whose objective is to process a signal in order to manipulate the information contained in the signal. For time-invariant filters the internal parameters and the structure of the filter are fixed. When either the fixed specifications are unknown or the specification can not be satisfied by time-invariant filters, adaptive filters are an attractive option. The adaptive filters are time-varying since their parameters are continually changing in order to meet a performance requirement. At low speeds, adaptive filtering can be easily implemented using digital circuits. On the other hand, analog filters are preferred at high speeds because of their low power consumption, small silicon area, and capability of performing equalization at high speed.

An adaptive filter system consists of two distinct parts: a *filter* with adjustable coefficients and an *adaptive algorithm* which is used to adjust or modify the coefficients of the filter, as shown in Figure 2.14. n is the iteration number. $u(n)$ and $y(n)$ are the input and output of the filter. $d(n)$ and $e(n)$ are the desired and error signals, where $e(n) = d(n) - y(n)$. The error signal is then used to form a performance (or objective) function that is required by the adaptation algorithm in order to determine the appropriate update of the filter coefficients. The minimization of the objective function implies that the adaptive filter output signal is matching the desired signal in some sense.

Figure 2.14: Typical adaptive filter as a two-input, two-output system

An adaptive filter has the ability to self-update its coefficients. New coefficients are sent to the filter from a coefficients generator, an adaptive algorithm that modifies the coefficients in response to an incoming signal. The goal of the adaptive filter is to find a set of filter coefficients to minimize the power of the error signal, $e(n)$. In a LMS algorithm, the coefficient optimization is achieved through the minimization of the mean-square value of an error signal formed by taking the difference between the filter output and a desired output.

2.7.2 The Least-Mean-Square (LMS) Algorithm

Many computationally efficient algorithms for adaptive filtering have been developed in the past twenty years [23]. They are based on either a statistical gradient approach,

such as the least-mean-square (LMS) algorithm, or a deterministic approach, such as the recursive least-square (RLS) algorithm. The least-mean-square (LMS) algorithm, which was first proposed by Widrow and Hoff in 1960 [18], is the most popular algorithm for the adaptation of integrated filters because of its robustness and simplicity.

There are two basic processes involved in the least-mean-square (LMS) algorithm as a linear adaptive filtering method:

1. *A filtering process*, which consists of (a) computations of the output of a transversal filter produced by a set of tap inputs, and (b) generating an estimation error by comparing this output to a desired response.
2. *An adaptive process*, which performs automatic adjustments of the tap weights of the filter in accordance with the estimation error.

The filtering process is performed through a transversal filter, as illustrated in Figure 2.15 [23]. The input to the transversal filter is applied to an equally spaced tapped delay line. Each tap output is then applied to a variable gain and the results are summed together to form the filter output. The filter output is:

Figure 2.15: N-tap transversal adaptive filter

$$y(k) = \sum_{i=0}^{N-1} w_i(k)u(k-i) \quad (2.6)$$

where N is the number of taps and $w_0(k), w_1(k), \dots, w_{N-1}(k)$ are the tap coefficients. This filter can be implemented in either continuous or discrete time with analog circuits.

The estimation error $e(n)$, the difference between the desired response and the filter output ($e(k) = d(k) - y(k)$) and the tap-input $u(n)$ are then applied to the adaptation algorithm to compute the updated coefficients to minimize the estimation error. The filter tap weights are functions of the time index k . This indicates that in an adaptive filter, the tap weights are time varying.

The performance criterion used for LMS adaptation is the mean-squared-error (MSE) [23]:

$$\varepsilon(w) = E[(d-y)^2] = E[e^2] \quad (2.7)$$

where the operator $E[\cdot]$ is the expectation. w is the filter coefficients. In the conventional LMS algorithm, each coefficient w is updated using the equation [18]:

$$w_{k+1} = w_k + \mu e_k u_k \quad (2.8)$$

where μ is the step size, e_k is the difference between the desired response and the actual filter output at time k , and u_k is the input to the particular tap weight at time k . Implementation of this algorithm requires two multiplies as well as the generation of the slicer error in order to obtain the correction term which is then added to the current value of the coefficient. The time required for these multiplies increases the latency of the coefficient update which in simulation resulted in stability problems in the adaptation, requiring very small values of μ [24].

In order to simplify the implementation of the LMS algorithm, variants of this algorithm were proposed. Some of these include the sign-data LMS (SD-LMS) [25], the sign-error LMS (SE-LMS), and the sign-sign LMS (SS-LMS) [26].

Using the sign-data LMS algorithm, the mean square error is calculated by using the sign of the input data to change the filter coefficients. When the input data is positive or zero, the new coefficients are the previous coefficients plus the error multiplied by the step size μ . If the input data is negative, the new coefficients are the previous coefficients minus the error multiplied by the step size μ . The updating equation for the sign-data LMS algorithm is:

$$w_{k+1} = w_k + \mu e_k \text{sgn}\{u_k\} \quad (2.9)$$

where $\text{sign}(\cdot)$ is the signum function which is +1 or -1 for a positive or negative argument, respectively.

Using the sign-error LMS algorithm, the mean square error is calculated by using the sign of the error to modify the filter coefficients. When the error is positive or zero, the new coefficients are the previous coefficients plus the input data multiplied by the step size μ . If the error is negative, the new coefficients are the previous coefficients minus the input data multiplied by μ . The updating equation for the sign-error LMS algorithm is:

$$w_{k+1} = w_k + \mu \text{sgn}\{e_k\} u_k \quad (2.10)$$

Using the sign-sign LMS algorithm, the mean square error is calculated by using both the sign of the input data and sign of the error to change the filter coefficients. When the product of the sign of input data and sign of the error is positive, the new coefficients are the previous coefficients plus the step size μ . If the product is negative, the new coefficients are the previous coefficients minus the step size μ . The updating equation for the sign-sign LMS algorithm is

$$w_{k+1} = w_k + \mu \text{sgn}\{e_k\} \text{sgn}\{u_k\} \quad (2.11)$$

The coefficient updates for the SD-LMS, SE-LMS, and SS-LMS then reduce to:

$$w_{k+1} = w_k \pm \mu e_k \quad (2.12)$$

$$w_{k+1} = w_k \pm \mu u_k \quad (2.13)$$

$$w_{k+1} = w_k \pm \mu \quad (2.14)$$

The hardware advantages of these simplifications over conventional LMS are significant. A full multiplier is reduced to a one-bit multiplier for SD-LMS and SE-LMS, and a single XOR gate for the SS-LMS.

In this research, the sign-sign block LMS algorithm is used for the adaptation and equalization process. By using this algorithm, rather than continuously updating the parameters via Equation (2.14), one might keep them fixed for a block of L consecutive data samples. The detail of the sign-sign block LMS algorithm will be presented in next chapter.

2.7.3 Channel Equalization

Adaptive filters are widely used to provide channel equalization (inverse filtering) in modems that transmit data over a bandlimited channel. An adaptive equalizer is employed to compensate for the distortion caused by the transmission medium. In general, its operation involves a training mode followed by a tracking mode.

Initially, the equalizer is trained by transmitting a known test data sequence $u(n)$. By generating a synchronized version of the test signal in the receiver, the adaptive equalizer is supplied with a desired response $d(n)$. The equalizer output $y(n)$ is subtracted from this desired response to produce an estimation error, which is in turn used to adjust the coefficients of the equalizer to their optimum values. When the initial training period is completed, the adaptive equalizer tracks possible time variations in channel characteristics during transmission by using a receiver estimate of the transmitted sequence as a desired response. The receiver estimate is obtained by applying the equalizer output $y(n)$ to a decision device, slicer.

3.1 Introduction

An accurate simulation model provides proof of concept and facilitates the functional verification of proposed architecture before the actual adaptation circuit is implemented in hardware. A complete Matlab simulation model for adaptive pre-emphasis is discussed in detail in this chapter. This chapter starts with the description of the adaptive pre-emphasis system in Section 3.2. Detail of a sign-sign block LMS algorithm is provided in Section 3.3. A Matlab functional model for the adaptation process is presented in Section 3.4. This model includes the equalizer, channel, convergence algorithm, and coefficient digital-to-analog conversion (CDAC) which includes some limitations, such as finite tap weight range and resolution, and finite transmit and receive bandwidths. Section 3.5 gives simulation results and hardware implementation parameters.

3.2 Circuit Description

3.2.1 Transceiver Block Diagram

The simplified transceiver architecture is shown in Figure 3.1. Only blocks related to the adaptive pre-emphasis are included. The on-chip clock for the transceiver is 500MHz.

Figure 3.1: Simplified transceiver block diagram

In the transmitter section, a built-in Random Data Generator generates training data during system initialization. The N:1 Multiplexer reduces the on-chip frequency requirement by a factor of N and converts parallel data to serial pulses and combines them into a serial stream. The multiplexer appropriately advances and delays the symbol for the pre- and post-taps in addition to the rover tap for the pre-emphasis. A 4-PAM modulation scheme reduces symbol rate by a factor of two compared to a conventional 2-PAM system. This symbol rate reduction lowers both signal ISI in the channel and the maximum required on-chip clock frequency [3]. The pre-emphasis, a conventional transversal filter with configurable taps, generates the equalized output waveform. The transmitted signal is pre-distorted by boosting the high frequency energy as it propagates through the channel.

In the receiver front end, a 1:N demultiplexer relaxes the on-chip speed required at the receiver front end to convert low-swing analog signals to digital signals. A set of slicers captures incoming data and the sign of error for the received signal samples which are required by the adaptation algorithm. Each branch of slicers consists of 10 slicers, as shown in Figure 3.2 [2]. 4-PAM signaling requires three slicers at reference levels +2, 0, and -2, sampling at Φ_{middle} to resolve the incoming four signal levels. In order to get the

error signal, the difference between the desired received signal and the real received signal, four additional slicers at reference levels +3, +1, -1, and -3 are used to measure whether the sample is above or below the reference level to get the sign of the error. The sign of the received signal and the sign of the error will be used in the adaptation engine to update the coefficients of the pre-emphasis. Finally, three slicers sampling at Φ_{edge} generate phase information for the clock recovery loop that aligns the on-chip clocks to the incoming symbol stream. An adaptation engine performs adaptation processing. The tap updates are computed using sign-sign block LMS algorithm and the adjustments for the taps are sent back to the transmit side where they are applied. Detailed hardware structure and implementation will be provided in Chapter 4. The clock edges used to sample the middle and edge transitions (Φ_{middle} and Φ_{edge}) must be locally generated and aligned for robust data recovery. This is done by the receiver PLL. The PLL uses phase information from the three phase-detecting slicers to lock the ten 200ps outputs of the oscillator to the middle and edge transitions of the symbol stream.

Figure 3.2: 4-PAM signal slicing

Hardware implementation of the adaptation algorithm depends on the structure of the pre-emphasis. The block diagram of the pre-emphasis used in this research is shown in Figure 3.3. The first row of 2-bit DACs sums and generates the equalized 4-PAM output

waveform. The second of DACs creates the signal associated with the main symbol, which is weighted by the coefficients to adjust the level of the transmitted symbols. The first and third DACs produce drives that are weighted by the coefficient values and proportional to the symbols that lead and trail the main symbol to cancel the precursor and postcursor. The fourth DAC provides a drive that is weighted by a coefficient and proportional to a symbol that may have a latency of 2 to 7 [2] symbol intervals after the main symbol to cancel the reflection caused by the imperfect matching of the terminals. The position of this tap is chosen to maximize the correlation between the delayed symbol and the residual ISI, thus, canceling the greatest remaining error. The default position of this tap is placed right after the trailer tap which has the latency of 2 symbol intervals after the main symbol. The four taps are called leader tap, main tap, trailer tap, and rover tap, respectively.

Figure 3.3: Pre-emphasis architecture with CDACs

The second row of the DACs are coefficient DACs (CDACs), which control the amount of current steered by the associated 2-bit output DAC in the first row. The inputs to

each coefficient DAC are controlled by tap adjustments issued by the adaptation engine at far end of the receiver side, which is the main task of this research work.

3.2.2 *Equalization Process*

In this section, a high-level view of how adaptive equalization works will be described. The detailed hardware implementation will be provided in Chapter 4. In this research work, equalization is performed at the transmitter with a pre-emphasis. The updated tap adjustments are calculated in the adaptation engine at the receiver and sent back to the transmitter via a low speed link. The transmitted signal is predistorted by boosting the high-frequency portion of the signal spectrum to compensate for the attenuation caused by higher frequency energy as it propagates through the channel. The adaptive pre-emphasis automatically sets the most appropriate coefficients to the pre-emphasis by communicating information between the transmit and receive paths.

There are two distinct modes involved in the equalization process. The first mode is startup mode which performs equalization during system initialization using a training sequence. The second mode is normal mode which performs equalization during normal operation. Figure 3.4 [2] depicts the communication flows of the equalization paths for both system startup and normal equalization. The solid line describes the system startup equalization, while the dashed line represents the normal equalization.

The system starts up with no information about the channel characteristics and the pre-emphasis coefficients are preset to nominal values which may not be optimal. A master and slave relationship is set between two interconnected systems and the coefficients of the equalizer in the slave are selected to converge first. During initialization, the pseudo-random-binary-sequence (PRBS) data is generated in the slave and is transmitted to the master. The receiver of the master compares the received and distorted PRBS data with locally generated or a delayed version of the PRBS to generate the error. During the normal operation, the error is generated by using a receiver estimate of the transmitted

sequence as a desired response. The receiver estimate is obtained by applying the equalizer output $y(n)$ to a decision device or slicer. Whenever the data and error are obtained, the signed data and signed error are provided to the adaptation engine where tap updates for the slave FIR filter coefficients are computed using the sign-sign block LMS algorithm. Tap updates are transmitted back to the far end slave at low speed which can be reliably received without equalization. Several iterations through this loop enable the slave equalizer taps to converge to the values that equalize the distortion introduced by the channel. Once this is completed for slave equalizer, the master and slave reverse roles. The master equalizer coefficients are adjusted with a similar process.

Figure 3.4: Equalization Path for startup and normal

After system startup, adaptive equalization continues working during normal operation. The adaptation engine continues to compute the tap update information and sends them back to the transmitter through an encoder which embeds an extremely low-rate control stream without any user bandwidth penalty. The receiver decodes this low-rate control signal and drives the coefficient DACs to provide the new coefficients to the equalizer. This communication flow works continuously and is bidirectional.

3.3 Sign-Sign Block LMS Algorithm

Choosing the right criterion and algorithm is very important for the optimization and convergence of the equalizer taps. The two most common criterion are the zero-forcing (ZF) criterion and the minimum mean square error (MMSE) criterion. A zero-forcing equalizer maximizes the contribution from the symbol of interest and removes all inter-symbol interference from other symbols. Such an equalizer is not suitable for noisy channels because it could enhance the additive noise in the process of eliminating ISI. An MMSE equalizer minimizes the mean square difference between the desired signal and transmitted signal without noise enhancement.

In this research work, a simple and effective sign-sign block least-mean-squares (LMS) algorithm is used for the adaptation process for the pre-emphasis. The tap update equations for the leader, main, trailer, and rover taps of block j are given by:

$$C_L(j+1) = C_L(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_{i+1}) \right] \quad (3.1)$$

$$C_M(j+1) = C_M(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_i) \right] \quad (3.2)$$

$$C_T(j+1) = C_T(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_{i-1}) \right] \quad (3.3)$$

$$C_R(j+1) = C_R(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_{i-M}) \right] \quad (3.4)$$

where the subscripts on the error (E) and data (S) represent their position within the data block j , L is the length of the block of data, and the signum function is defined as:

$$\text{sgn}(x) = \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases} \quad (3.5)$$

C_L , C_M , C_T and C_R are the inputs for the coefficient DACs at the transmit side shown in Figure 3.3. The outputs of these coefficient DACs provide the current to the 2-bit output DAC to form the 4-PAM output waveform. By using the sign-sign block LMS algorithm, these coefficients are fixed for a block of L consecutive data samples.

3.4 Matlab Simulation Model

A Matlab model of the adaptive pre-emphasis facilitates the functional verification of the proposed architecture before the actual adaptation circuitry is implemented in hardware. A complete Matlab simulation code for adaptive pre-emphasis simulation is developed. A Matlab simulation model used for this research is shown in Figure 3.5.

Figure 3.5: Simulation block diagram

This simulation model includes the following functions: random data generator, adaptive pre-emphasis, coefficient digital-to-analog converter (CDAC), channel model, AWGN generator and adaptation engine.

3.4.1 Random Data Generator

Unlike in real hardware design, a random data sequence was used in the simulation instead of using pseudo-random-binary-sequence (PRBS). In order to verify whether or not the sign-sign block LMS algorithm is suitable for both 2-PAM and 4-PAM modulation schemes, two different random data sequences were generated. One is the random 4-PAM data sequence which is created using the *rand* and *ceil* functions. The other is the random 2-PAM sequence which is created using only the *rand* function. The generated random data sequence is provided as input data to the adaptive pre-emphasis. The delayed version of the random data sequence is used as the desired signal to create error signal and provides input to the adaptation engine in the receiver.

3.4.2 Adaptive Pre-Emphasis

The adaptive pre-emphasis is modeled as an adaptive FIR filter with four configurable taps, as shown in Figure 3.6. This filter is a conventional transversal filter and retains the digital programmability of fully digital FIR equalizers by using coefficient digital-to-analog converters (CDAC's) for the coefficients. The output of the FIR filter, $y(n)$, can be expressed as:

$$y(n) = w_L u(n+1) + w_M u(n) + w_T u(n-1) + w_R u(n-M) \quad (3.6)$$

where w_L is the analog coefficient value for the symbol that leads the main symbol.

w_M is the analog coefficient value for the main symbol.

w_T is the analog coefficient value for the symbol that trails the main symbol.

w_R is the analog coefficient value for the symbol that may have the latency of 2 ~ 7 symbol intervals after the main symbol.

The leader (w_L) and trailer (w_T) components are used to cancel the inter-symbol interference (ISI), the main (w_M) component is used to adjust the amplitude of the transmitted signal to maintain the received signal within a desired range, and the rover (w_R)

component is used to cancel the reflections due to discontinuities and imperfect terminations of the transmission lines. The value of each coefficient is provided by a appropriately sized coefficient DAC in the Digital-to-Analog Conversion block.

Figure 3.6: Functional block diagram of an adaptive pre-emphasis

The coefficient w_R may have a latency of 2 ~ 7 symbol intervals after the main symbol because of the propagation through FR-4 (fire-resistant epoxy-bonded fiber-glass board material). The typical propagation through FR4 is about 170ps/inch, and given that a PAM-4 symbol is about 2.4 inches long, and that the equalizer can see round-trip reflections of about 16.8 inches (8.4 inches one way), the total distance range in length from 2 to 7 PAM symbol. This is adequate in most situations, since as the reflection source gets further away, the loss in the line tends to attenuate the reflected energy into the noise floor [2]. Choosing the right position of the rover tap can maximize the correlation between the delayed symbol and residual ISI, thus, cancel the worst remaining error. The default position of the rover tap is right after the trailer tap, which means the default value of the rover tap position M is 2.

3.4.3 Coefficient Digital-to-Analog Convertors (CDACs)

Digital-to-analog conversion (DAC) is the process of changing discrete digital data into a continuously varying signal in relation to a standard or reference. The inputs of the

CDACs are controlled by the adaptation engine at the receiver side. The outputs of the CDACs control the amount of current required by the adaptive pre-emphasis. There are four CDACs in this block corresponding to the four taps. Resolutions for the leader, main, trailer and rover taps used in simulation are 5-bit, 6-bit, 6-bit and 4-bit, respectively.

3.4.4 Channel Model

The complexity and performance of the channel depends strongly on the application. In this research, three different channel models were used in the simulation to check the performance of the adaptive pre-emphasis.

3.4.4.1 Channel Model One

The first channel model corresponds to an FIR filter that approximates a copper channel in the range from 12-inch to 30-inch. The channel attenuation is characterized by the following equation provided by Altera Corporation:

$$loss(f) = (A + B \cdot \sqrt{f} + C \cdot f) \times l \quad (3.7)$$

where f is the frequency in Hz, $A = 0.0453$ dB/inch, $B = 2.3447e-6$ dB/inch, $C = 1.2314e^{-10}$ dB/inch, l is the length of the PCB trace in inch, $loss$ is the channel loss at length l . The impulse response and the attenuation characteristics for a 26-inch channel at sampling rate 5 GHz are shown in Figure 3.7 and Figure 3.8, respectively. By using this channel model, only channel attenuation is considered in the channel compensation. No phase distortion is compensated.

3.4.4.2 Channel Model Two

The second channel model corresponds to an 1m long, 1.2mm wide stripline [27]. The impulse response and the frequency response of the second channel are shown in Figure 3.9 and Figure 3.10, respectively.

3.4.4.3 Channel Model Three

The third channel model corresponds to a backplane (FR4) circuit model from Altera Corporation. This channel includes termination resistors and the 30-inch FR4 channel. The impulse response and the frequency response of this channel are shown in Figure 3.11 and Figure 3.12, respectively.

3.4.5 AWGN Generator

At the output of the channel, a noise sequence is added. The noise is assumed to be Additive White Gaussian Noise (AWGN) with variance $\sigma_n^2 = 0.001$. It is modeled using Matlab function *randn*. The sum of the channel output and the noise sequence forms the received signal.

3.4.6 Adaptation Engine

The adaptation engine implements the sign-sign block LMS adaptation algorithm and passes the updated coefficients to the input of the coefficient DACs. Two different structure adaptation engines were designed. The first adaptation engine is called the round-robin adaptation engine which only changes one tap coefficient after each data block. It requires four data blocks to update all tap coefficients. These coefficients are then sent back to the transmitter. The second adaptation engine is called the parallel adaptation engine which updates all tap coefficients in one data block and then sends them back to the transmitter side. So the parallel adaptation engine is four time faster than the round-robin adaptation engine in terms of coefficient update rate. The block diagrams of both structures will be presented in Chapter 4.

Figure 3.7: Impulse response of the first channel

Figure 3.8: Channel loss of the first channel

Figure 3.9: Impulse response of the second channel

Figure 3.10: Frequency response of the second channel

Figure 3.11: Impulse response of the third channel

Figure 3.12: Frequency response of the third channel

3.5 Simulation Results

In this section, we present simulation results that show the performance of the adaptive pre-emphasis. The performance will be evaluated for both 2-PAM and 4-PAM modulation schemes covering the three different channel models at a rate of 2.5GBd/s or 5GBd/s using two design flows. For all simulations, the data block size is implicitly chosen to be 1024 symbols.

3.5.1 Performance of the Adaptive Pre-Emphasis over Channel One

The performance of the adaptive pre-emphasis over channel one was evaluated using flowchart shown in Figure 3.13.

Figure 3.13: Design flowchart for the first channel model

In this flow, only channel attenuation is considered. The channel is modeled as an FIR filter during simulation. At each iteration, a block of random data is generated, then it is passed through the adaptive pre-emphasis and the channel. The output sequence of the channel is compared with the delayed version of the original data to generate an error sequence. The sign of data and the sign of error are passed to the sign-sign block LMS algorithm to calculate the tap adjustments, and then to update the tap coefficients of the pre-emphasis. At a rate of 2.5GBd/s, Figure 3.15 shows the eye diagram without pre-emphasis. Figure 3.16 shows eye diagram with pre-emphasis. Figure 3.17 shows the optimal coefficients of the pre-emphasis. Figure 3.18 shows the MSE vs. the number of iterations.

3.5.2 Performance of the Adaptive Pre-Emphasis over Channel Two

The performance of the adaptive pre-emphasis over channel two and three was evaluated using flowchart shown in Figure 3.14.

In this flow, a block of random data was generated at each iteration. These data are passed through the pre-emphasis. The output of the pre-emphasis is upsampled 5 times and passed through the channel. The output sequence of the channel is down-sampled 5 times. It is then compared with the delayed version of the original data to generate the error signal. The sign of data and the sign of the error are passed to the sign-sign block LMS algorithm to calculate tap adjustments, and then the tap coefficients of the pre-emphasis are updated. At a rate of 2.5GBd/s, Figure 3.19 shows the eye diagram without pre-emphasis. Figure 3.20 shows the eye diagram with pre-emphasis. Figure 3.21 shows the optimal coefficients of the pre-emphasis. Figure 3.22 shows the MSE vs. the number of iterations.

3.5.3 Performance of the Adaptive Pre-Emphasis over Channel Three

The same flowchart shown in Figure 3.14 is used for the channel three simulation. At 5GHz baud rate, Figure 3.23 shows the eye diagram without pre-emphasis. Figure 3.24 shows eye diagram with pre-emphasis. Figure 3.25 shows the optimal coefficients of the pre-emphasis. Figure 3.26 shows the MSE vs. the number of iterations.

Figure 3.14: Design flowchart for the second/third channel model

The performance of the adaptive pre-emphasis can also be verified via passing a pulse through the channel. One example of pulse response for the third channel model is shown in Figure 3.27. The sampling times are marked on the x-axis. Note that near -zero

ISI is achieved. It is clear that skin effect and dielectric loss cause significant ISI if channel compensation is not utilized.

3.5.4 Effect of the Adaptation Step Size

The adaptation step size controls the convergence speed as well as the final mean square error of the equalizer. A smaller adaptation step size takes a longer time to convergence but provides smaller variations in the mean square error and a larger eye-opening. The choice of step size involves a trade-off between rapid convergence and the desire to keep the variance of the steady state error small. The effects of the adaptation step size are shown from Figure 3.28 to Figure 3.33. Figure 3.28, Figure 3.30 and Figure 3.32 show eye-openings for adaptation step size 0.01, 0.02 and 0.1, respectively. Figure 3.29, Figure 3.31 and Figure 3.33 show corresponding MSEs. Adaptation step size determines the convergence speed. It also determines the tap resolution of the coefficient digital-to-analog conversion (CDACs). As we can see from these results, there is no significant difference in the eye-opening when adaptation step size is chosen to be 0.01 or 0.02, but convergence speed is faster using step size 0.02. Choosing 0.1 as a step size will however result in significantly smaller eye-opening. Therefore, the adaptation step size is chosen to be 0.02 in this design. With the selected tap resolutions, 5-bit, 6-bit, 6-bit, and 4-bit for leader, main, trailer, and rover taps, respectively, the bias current tuning ranges for these taps are 0.64, 1.28, 1.28, and 0.32, respectively. If the wider tuning ranges are required to cover a larger range of channel lengths in practice, larger tap resolutions are needed or bias currents are required to be adjusted.

3.5.5 Choose Data Block Size

In order to choose the data block size for hardware implementation, optimal coefficients of the adaptive pre-emphasis versus data block size for three channel models are shown in Figure 3.34 to Figure 3.36. As we can see from these figures, optimal coefficients of the adaptive pre-emphasis vary very little when data block size changes between

100 and 1500. A block size of 1024 symbols was chosen for the hardware implementation in order to use 10-bit PRBS in real design to cover enough frequency components.

(a) 2-PAM

(b) 4-PAM

Figure 3.15: Eye diagram without pre-emphasis at a rate of 2.5GBd/s over channel one

(a) 2-PAM

(b) 4-PAM

Figure 3.16: Eye diagram with pre-emphasis at a rate of 2.5GBd/s over channel one

(a) 2-PAM

(b) 4-PAM

Figure 3.17: Optimal coefficients of pre-emphasis at a rate of 2.5GBd/s over channel one

(a) 2-PAM

(b) 4-PAM

Figure 3.18: Mean square error vs. number of iterations at a rate of 2.5GBd/s over channel one

(a) 2-PAM

(b) 4-PAM

Figure 3.19: Eye diagram without pre-emphasis at a rate of 2.5GHz over channel two

(a) 2-PAM

(b) 4-PAM

Figure 3.20: Eye diagram with pre-emphasis at a rate of 2.5GBd/s over channel two

(a) 2-PAM

(b) 4-PAM

Figure 3.21: Optimal coefficients of pre-emphasis at a rate of 2.5GBd/s over channel two

(a) 2-PAM

(b) 4-PAM

Figure 3.22: Mean square error vs. number of iterations at a rate of 2.5GHz over channel two

(a) 2-PAM

(b) 4-PAM

Figure 3.23: Eye diagram without pre-emphasis at a rate of 5GBd/s over channel three

(a) 2-PAM

(b) 4-PAM

Figure 3.24: Eye diagram with pre-emphasis at a rate of 5GBd/s over channel three

(a) 2-PAM

(b) 4-PAM

Figure 3.25: Optimal coefficients of pre-emphasis at a rate of 5GBd/s over channel three

(a) 2-PAM

(b) 4-PAM

Figure 3.26: Mean square error vs. number of iterations at a rate of 5GBd/s over channel three

Figure 3.27: 200ns Pulse response for the channel three

Figure 3.28: Eye diagram for adaptation step size = 0.01

Figure 3.29: MSE for adaptation step size = 0.01

Figure 3.30: Eye diagram for adaptation step size = 0.02

Figure 3.31: MSE for adaptation step size = 0.02

Figure 3.32: Eye diagram for adaptation step size = 0.1

Figure 3.33: MSE for adaptation step size = 0.1

Figure 3.34: Coefficients vs. data block size for the first channel

Figure 3.35: Coefficients vs. data block size for the second channel

Figure 3.36: Coefficients vs. data block size for the third channel

3.6 Conclusions

Eye diagrams in Section 3.5 show significant improvement in the eye openings using adaptive pre-emphasis. The simulation proved that the sign-sign block LMS algorithm is a simple and efficient adaptation algorithm. It effectively compensates for ISI caused by copper backplane channels. The sign-sign block adaptation algorithm is compatible with both binary (2-PAM) and 4-PAM modulation, which means that at the same data baud rate, the optimal coefficients of the pre-emphasis are the same for both 2-PAM and 4-PAM modulations. With limited tap weight range and resolutions, the bias current settings are extremely important for the CDACs during simulation. A data block size of 1024 was used for hardware simulation.

Custom Adaptation Engine Design

4.1 Introduction

In Chapter 3, the functional simulation of an adaptive pre-emphasis using the sign-sign least-mean-square (LMS) algorithm was presented. Two different hardware structures of a sign-sign block LMS algorithm implemented in Verilog targeting the TSMC 0.18um technology will be discussed in detail in this chapter. Designs are partitioned into a few building blocks. RTL code is written for each individual building block and verification was performed on the top module of the adaptation engine. The comparison between these two structures is discussed at the end of this chapter.

4.2 Design Specification

The goal of this research is to implement two different hardware structures for a sign-sign block LMS algorithm for an adaptive pre-emphasis in a copper backplane transceiver following the digital IC design flow supported by CMC. One is the round-robin adaptation engine, the other is the parallel adaptation engine. Based on the simulation results obtained from Chapter 3, design assumptions and specification for both circuits are:

- Assume the sign of data and the sign of error are synchronized.
- The designs should meet a 500MHz timing constraint.

- The size of a data block is 1024 (2^{10}).
- The resolutions for the leader, main, trailer and rover taps are 5-bit, 6-bit, 6-bit, 4-bit, respectively.
- The latency of the symbol which is weighted by the rover tap can be programmed in the range of 2 to 7. The default value is 2, which means the rover tap is just next to the trailer tap.
- The design should have over 98% fault coverage using a multiplexed flip-flop scan-based test system.
- Power consumption should be less than 100 mW.

4.3 Verilog HDL Implementation

4.3.1 Adaptation Algorithm

As described in Chapter 3, the adaptation and equalization process is driven by a simple and effective sign-sign block least-mean-square (LMS) algorithm. For convenience, the tap update equations for the leader, main, trailer, and rover taps for the block j are represented as following:

$$C_L(j+1) = C_L(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_{i+1}) \right] \quad (4.1)$$

$$C_M(j+1) = C_M(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_i) \right] \quad (4.2)$$

$$C_T(j+1) = C_T(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_{i-1}) \right] \quad (4.3)$$

$$C_R(j+1) = C_R(j) + \text{sgn} \left[\sum_{i=0}^{L-1} \text{sgn}(E_i) \text{sgn}(S_{i-M}) \right] \quad (4.4)$$

where the subscripts on the error (E) and data (S) represent their position within the data block, L is the length of the block of data, and the signum function is defined as:

$$\text{sgn}(x) = \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases} \quad (4.5)$$

C_L , C_M , C_T and C_R are the inputs for the CDACs at transmit side shown in Figure 3.3. The outputs of these coefficient DACs provide the current to a 2-bit output DAC to form a 4-PAM output waveform. By using a sign-sign block LMS algorithm, these coefficients are fixed for a block of L consecutive data samples.

Once a sufficient number of terms have been summed, the tap adjustments are computed and sent back to the transmitter side where they are applied. The continuous adaptation process allows the equalizer to compensate for distortions, such as the impairments of packages, connectors, backplane traces, power-supply drift, component aging, and temperature variations. A block size of 1024 symbols was used for hardware implementation.

4.3.2 Digital Round-Robin Adaptation Engine

The first adaptation algorithm circuitry structure is the round-robin adaptation engine, which was proposed by Acclerant Networks, as shown in Figure 4.1. The 8-bit shift register and the associated 4:1 multiplexer are used to select the right “sign of the data” for each tap. A single flip-flop is used to provide a delayed version of the “sign of error”, which is required in Equation (4.1) to Equation (4.4). The multiplication of the sign of data and the sign of error is realized by using an XOR gate. The consecutive XOR outputs are accumulated in an up/down counter and the sign of the result is captured in a latch bank. The sequencer is the controller of the adaptation engine. It consists of a counter which generates a summation period control signal which is sent to the up/down counter. This control unit also changes the tap position after each block of data. When all

tap adjustments have been determined, a control signal generated by the sequencer is sent to the message generator to create the digital tap coefficients which are fed back to the coefficient DACs in the transmitter.

Figure 4.1: Block diagram of round-robin adaptation engine

4.3.2.1 8-Bit Shift Register

The block diagram of the 8-bit shift register is shown in Figure 4.2. The sign of data is shifted in and stored in the registers. The input $\text{rover}[2:0]$ is used to select the proper sign of data term for the rover tap. The range of this input is 2 to 7, with the default input being 2. The input sign of the data combined with outputs of this sub-block (Q_2 , Q_1 , and Q_0) are the proper sign of the data terms for the leader, main, trailer and rover taps, which are required by the sign-sign block LMS adaptation algorithm.

Figure 4.2: Block diagram of the 8-bit shift register

4.3.2.2 4:1 Multiplexer

The inputs to the 4:1 multiplexer are the sign of data for four taps. The output is selected based on the current tap which is controlled by the tap sequencer block.

4.3.2.3 Sequencer

A sequencer is the control unit of the round-robin adaptation engine. The timing diagram of the sequencer is shown in Figure 4.3.

Figure 4.3: Timing diagram of the tap sequencer sub-block

The tap sequencer consists of a 10-bit counter. When all outputs of this 10-bit counter are high, it generates a summation period control signal (*blk_cntr*) for the up/down counter and tap position control signal (*tap_pos*) for the 4:1 multiplexer and the latch bank. When the tap position control signal is 11 and all outputs of 10-bit counter are high, a message generation control signal (*gen_msg*) is created for the message generator sub-block.

4.3.2.4 Up/Down Counter

An 11-bit up/down counter is designed to accumulate the product of the sign of data and the sign of error. The most significant bit (MSB) of the up/down counter is used for the sign of the accumulated result. When the summation period control signal (*blk_cntr*) is high, the inverted MSB is sent out and captured in the latch bank with the proper tap position.

4.3.2.5 Latch Bank

The latch bank is a set of registers which are designed to store tap update information (up or down) for the proper tap position as selected by the tap sequencer block.

4.3.2.6 Message Generator

Message generator consists of four up/down counters and one parallel to serial block. These four up/down counters correspond to the four taps. The number of bits for each up/down counter is determined by the corresponding tap weight resolution. The number of bits for leader, main, trailer and rover up/down counters are 5, 6, 6 and 4, respectively. The outputs of these up/down counters are initially set to high except the most significant bit (MSB) at the beginning of the adaptation process. If the output reaches the up or low bound (outputs are all 1 or all 0) and the tap update is 1 or 0, the output will keep the previous value. The output of the message generator is triggered by the message generation control signal which is created by the sequencer block. The outputs of the message

generator is sent back to the transmitter. The block diagram of message generator is shown in Figure 4.4.

Figure 4.4: Block diagram of message generator

This block diagram is only used for this research work. In the actual design, the four up/down counters are located in the transmitter, only the tap adjustments (up/down) will be sent to the transmitter to control the four up/down counters. The output of the message generator is the input to the coefficient DACs.

4.3.3 Digital Parallel Adaptation Engine

In order to speed up coefficient update rate and to relax the critical path delay from the output of the 8-bit shift register to the input of the up/down counter, a parallel adaptation engine is proposed, as shown in Figure 4.5. In this structure, all tap coefficients are updated at the end of each block of data. All blocks in Figure 4.5 have the exact same functionality as in the round-robin adaptation engine except that the sequencer block is much simpler. A single sequencer provides simultaneous control for all four up/down counters. At the end of each block of data, a summation control signal is sent to all up/

down counters to trigger all updated tap adjustments and a message control signal is sent to message generator to create digital tap coefficients.

Figure 4.5: Block diagram of parallel adaptation engine

4.3.3.1 Tap Sequencer

As mentioned above, the tap sequencer block of the parallel adaptation engine is much simpler than that in the round-robin adaptation engine. This tap sequencer acts as a 10-bit counter. Whenever the outputs of this 10-bit counter are all 1s, a summation period control signal is sent to all the up/down counters. At next the clock cycle, a control signal is sent to the message generator block to trigger the up/down counters and parallel-to-serial conversion.

The schematic views of all modules are provided in Appendix A.

4.4 Design Process

Modified digital IC design flow supported by CMC was used for the adaptation engine VLSI implementations. Figure 4.6 depicts the design flow with the tools used in the flow.

Figure 4.6: Digital IC Design Flow

4.4.1 Register-Transfer Level (RTL) Model and Simulation

4.4.1.1 RTL Model

The most important aspect of implementing an efficient design is to properly model the design. The RTL models the design in detail. Only one clock is used for the design. Both designs are fully synchronous off the rising edge of the clock. No latches or three-state devices were used. The synchronous reset input will place the entire design in a known state. It is more time efficient to design in RTL than in gate-level. By using an RTL model, the full-power of synthesis and optimization tools can be used to generate a final gate level design.

The hierarchical design of both adaptation engines was captured using Verilog HDL and debugged and validated at the register-transfer level. In the development phase, both the behavioral and structural descriptions were applied in the Verilog HDL code.

4.4.1.2 RTL Simulation

Verification is extremely important in an ASIC/FPGA design flow. In this work, verification was first performed on each individual building block. A testbench for each block was written and simulation results were verified. Finally, the verification of the top level design containing an integration of all individual blocks was carried out using the Cadence *Verilog-XL* simulator.

For functional verification, test vectors (input vectors and expected outputs) are fed into the circuit. In the top level testbench, the functional vectors were generated by Matlab simulation. 100 arbitrary blocks of 1024 symbols and the corresponding blocks of 1024 errors were generated, with the expected tap adjustments. One of the expected results was altered so that the test bench would flag errors starting from the altered results. The designs were proven to agree with the design intent. Figure 4.7 shows an example of an RTL simulation result for the round-robin adaptation engine. A pipeline technique is used

in both designs to solve the critical data path delay from the output of the 80bit shift register to the 11-bit un/down counter(s). After four blocks of data and a several clock cycles of delay, the outputs of the up/down counters in the message generator are updated based on the tap adjustment information and sent out through the parallel-to-serial module.

4.4.2 Logic Synthesis

Logic synthesis is the process of converting the design from RTL code into gate-level code, generating timing constraints to be met by the placement and routing tools, and creating scan-based test and test vectors. Logic synthesis involves in three steps: translation, optimization, and mapping. First it translates the RTL code into generic boolean (GTECH), then it optimizes and maps the GTECH into the target technology based on the provided constraints. GTECH components have no timing or load characteristics, and are only used by Synopsys *Design Compiler* through *Design Analyzer* graphical interface.

4.4.2.1 Design Compiler - Synthesis

Design Compiler is the core of the Synopsys synthesis software products. It comprises tools that synthesize the Verilog HDL code of the adaptation engine circuits into optimized technology-dependent, gate-level designs. Timing and area goals for the design were applied before the design was optimized. The timing constraints are the most important parameters for the design. The clock period for the design is 2ns, which implies 500MHz clock frequency. The clock skew type is propagation skew. A 0.05ns uncertainty “window” surrounding each clock edge was defined.

4.4.2.2 Test Compiler - Scan Insertion

Today, with gate counts pushing into multi-millions, ASIC design requires managing the complex and difficult test issues. Design-For-Testability (DFT) can save time and money, particularly since it has been estimated that the cost of manufacturing test can represent as much as 30% of the total manufacturing cost [19]. Scan-based DFT is used to

allow the chip to be easily tested after manufacture to ensure there are no manufacturing errors, or subsequence problems (bonding problems for example) with the chip.

In this work, DFT means building the design in such way that all sequential elements (FF's) can be replaced with multiplexed equivalents, which are connected into a shift register. When in "scan mode" this long shift register (or scan chain) is used to pre-load and sample values of internal nodes of the circuit. When scan based testing is not used, the multiplexed FF's are placed in non-scan mode, and the circuit operates normally. In order to use scan-based test, scan in and scan enable were added to the design. The scan clock is the same pin as the regular clock, and scan out is the same pin as the data out.

Synopsys *Test Compiler* was used to substitute for all sequential devices (FF's) with scan equivalents and then connect them together to form a scan chain, to create a set of test vectors which can detect "stuck at 1" and "stuck at 0" faults in the chip. Full scan methodology was used to provide high fault coverage and improved diagnostic capabilities.

A multiplexed D flip-flop was selected to implement the full scan test. The multiplexed flip-flop implementation uses a multiplexed data input to provide serial shift capability. During the scan mode, the scan enable selects the scan data input which comes from either the scan input port or the scan output pin of the previous cell in the design chain. During the functional mode, the scan enable selects the system data input.

The timing reports of the worst paths after the scan-chain insertion for both circuits are in Appendix B.

4.4.2.3 Gate-level Simulation

At the end of logic synthesis, the gate level simulation was performed using the Cadence *Verilog-XL* simulator to confirm that the functionality is still the same as that of the original RTL code. This gate-level simulation does not count the effects of parasitic

and loading. A testbench for the RTL code was modified to match the gate-level netlist (signal naming conventions are different between RTL and gate-level netlists. In the RTL netlist, busses are written in the form A[7:0], while busses are expanded into all of their individual bits in the gate-level netlist). The same test vectors were used for gate-level simulation. The simulation results match the results of the original RTL simulation. Figure 4.8 gives an example of gate-level simulation result for the round-robin adaptation engine.

4.4.3 Physical Design

4.4.3.1 Floorplanning, Placement, and Routing

The purpose of the floorplanning is to create a floorplan for the design including a default group of cells, I/O ring connected by abutment, and defined placement sites for all cells. The *Physical Design Planner (PDP)* from Cadence was used to generate floorplan of the design before placement and routing. A detailed I/O ring floorplan which allows all of the I/O cells to be connected by abutment was prepared before using *PDP*.

Timing-driven placement of standard cells and clock tree generation were performed using the *First Encounter* tool from Cadence. The forward-annotated timing information was obtained from the Synopsys *Design Analyzer*. The balanced clock tree was generated by adding clock buffer cells and nets. The golden netlist for the design was generated and final verification was validated after clock tree insertion. Since the static timing analysis will be performed by *Pearl* in *Silicon Ensemble* after routing. The Verilog simulation is only to confirm that the functionality of the netlist remains the same as it was before the clock tree was inserted. The golden netlist is functionally equivalent to the original gate level netlist. This golden netlist will be used as the schematic when the final layout is verified with automatic Layout vs. Schematic (LVS) at the end of the design cycle. The simulation result for the gold-netlist is shown in Figure 4.9.

Silicon Ensemble was used to route the power, clock, and regular nets of the design using existing timing constraints. *Pearl* was then used to analyze the timing of the design based on the extracted parasitics of the routed nets. All timing goals were met after routing. The post-layout timing reports for both circuits are shown in Appendix C. The design DEF (Design Exchange Format) was exported from *Silicon Ensemble* for importing into the *Cadence DFII* (Design Framework II) environment for LVS (Layout Versus Schematic) and DRC (Design Rule Check) verification, metal slotting, adding a logo, and stream out into a GDSII database.

4.4.3.2 Physical Verification

Two important checks were performed in Cadence *DFII* environment after physical design: DRC (Design Rule Check) and LVS (Layout Versus Schematic). DRC checks for CMOS18 technology design rule violations that may have occurred during place & route. LVS verifies that the physical (Placed & Routed) version of the design contains the same instances, nets, and connectivity as the verified golden netlist. Diva LVS in Cadence *DFII* was used to compare the final layout in the form of a DEF (Design Exchange Format) file created from *Silicon Ensemble* to the golden Verilog netlist which was created after clock tree synthesis and subsequent functional verification.

Since Diva cannot work directly with a Verilog netlist, the netlist was first converted to a schematic view. Also the DEF file was translated into a layout view for the purpose of layout extraction.

Once all verification was completed, stream files for both designs were generated from the layout files and were submitted to CMC for calibre DRC checking. Final stream files for both designs were submitted to CMC for circuit fabrication. The final chip die size is 1122um x 1222um, as in Figure 4.10.

Figure 4.7: RTL simulation result of the round-robin adaptation engine

Waveform 1 - SimVision

Lei Lin
Carlson University
Gate-Level simulation for the round-robin convergence

Figure 4.8: Gate-level simulation result of the parallel adaptation engine

Waveform 1 – SimVision

Lei Lin
Carleton University
Gold-Netlist simulation for the round-robin convergence

Figure 4.9: Gold-netlist simulation result of the parallel adaptation engine

Figure 4.10: Layouts of two adaptation engines

4.4.4 Comparison of the Adaptation Engines

The comparisons between the two adaptation engines from the synthesis results are listed in Table 4.1. The comparisons show that the two structure are comparable in terms of hardware requirements.

TABLE 4.1: Comparison of Synthesis Results

Item	Round-Robin Adaptation Engine	Parallel Adaptation Engine
On-chip clock	500MHz	500MHz
Total Cell Area	113608 μm^2	122719 μm^2
Total Dynamic Power	49.3mW	56.8mW
Cell Leakage Power	1.02 μW	1.50 μW
Sequential Cells	111	126
Fault Coverage	99.28%	99.92%

The total cell area required for logic is slightly less for the round-robin but this is not an advantage as the die is pad limited. This results in the overall die size being the same for both implementations.

The parallel adaptation engine is four times faster in terms of block update rate, due to accumulating all tap adjustments in parallel. Also, the control block for the parallel

adaptation engine is much simpler than that of the round-robin adaptation engine. It is easier to expand more taps by using the parallel adaptation engine without changing the control unit, sequencer module.

Conclusions and Future Work

5.1 Conclusions

The main purpose of this thesis is two fold: to verify the effectiveness of the sign-sign block LMS algorithm at a high data rate (up to 5Gb/s) with Matlab simulation and to implement two different hardware structures of a sign-sign block LMS algorithm for an adaptive pre-emphasis in a copper backplane transceiver.

In the Chapter 2 of this thesis, the basic theory necessary for the proper understanding of a serial link backplane application was reviewed. The serial link communication structure and limitations of the backplane environment are provided. Equalization techniques, multi-level signalling and eye diagrams are introduced. The fundamental concepts of adaptive filters and channel equalization are also discussed.

The key contributions of this research work were presented in Chapter 3 and Chapter 4. In Chapter 3, a detailed study of an adaptive pre-emphasis scheme to improve signal quality (eye-opening) for high speed data transmission on copper backplane was presented. Functional models and Matlab code were developed to simulate both structures over three different channel models. Some of simulation results are included in this thesis, which show that the sign-sign block LMS algorithm is a simple and effective algorithm for the adaptation and equalization process.

In Chapter 4, two different hardware adaptation engine structures implemented in TSMC 0.18um digital CMOS technology were explored. The parallel adaptation engine is four times faster in terms of convergence speed compared to the round-robin adaptation engine (one block of data vs. four blocks of data) with comparable hardware requirements. Also, the control block for the parallel adaptation engine is much simpler than that of the round-robin adaptation engine. In order to reduce the path delay of the critical path, from the outputs of the 8-bit shift register to the input(s) of the 11-bit up/down counter, pipeline technique was used during hardware design. Several of flip-flops were added in both data and clock paths, as shown in Appendix A.

One objective of this research is to determine if CMOS18 standard cells can be used to achieve the imposed 500MHz timing constraints. The post-layout timing reports, as presented in Appendix C, show that the worst setup slack for both circuits is 0.55ns, which means that the timing constraint for both circuits can be pushed to as high as 625MHz without timing violation. A custom circuit is not needed when implementing the digital adaptation algorithm for the analog adaptive pre-emphasis up to 625MHz.

5.2 Future Work

As we can see from the post-layout timing reports shown in Appendix C, the 10 worst setup paths are all in the message generator and 11-bit up/down counter sub-blocks. If a higher on-chip clock is desired, 0.13um CMOS technology or custom design up/down counters can be an option. Advanced research could probe for a more efficient adaptation algorithm for optimizing coefficients.

As both designs were submitted for fabrication, we need to design a test plan to verify them as well.

References

1. Songgtag, J.; Stonick, J.; Gorecki, J.; Beale, B.; Check, B.; Xue-Mei Gong; Guiliano, J.; Kyong Lee; Lefferts, B.; Martin, D.; Un-Ku Moon; Sengir, A.; Titus, S.; Gu-Yeon Wei; Weinlader, D.; Yaohua Yang; "An Adaptive PAM-4 5 Gb/s Backplane Transceiver in 0.25 um CMOS", Custom Integrated Circuits Conference, 2002. Proceedings of the IEEE 2002, 12-15 May 2002, page(s):363-366.
2. Stonick, J.T.; Gu-Yeon Wei; Sonntag, J.L.; Weinlader, D.K.; "An Adaptive PAM-4 5-Gb/s Backplane Transceiver in 0.250um CMOS", Solid-State Circuit, IEEE Journal of, Volume: 38 Issue: 3, March 2003, page(s) 436-443.
3. Ramin Farjad-Rad, "A CMOS 4-PAM Multi-Gbps Serial Link Transceiver", PhD thesis, August 2002, Stanford University.
4. Anthony Chan Carusone, "Digital Algorithm For Analog Adaptive Filters", PhD thesis, 2002, University of Toronto.
5. Howard Johnson, Martin Graham, "High-Speed Digital Design", Prentice Hall PTR, 1993.
6. William J. Dally and John W. Poulton, "Digital System Engineering", Cambridge University Press, 1998.
7. R.W. Lucky, "Techniques for adaptive equalization of digital communication systems", Bell Syst. Tech. J., vol. 45, page(s) 225-286, Feb. 1996.
8. J. Zhang and Z. Wong, "White Paper on Transmit Pre-Emphasis and Receive Equalization", Mindspeed Technologies - A Conexant Business, Oct. 2002.

9. C.-J. Tzeng, "An Adaptive Offset Cancellation Technique for Adaptive Filters," IEEE Trans. on Acoust, Speech, Signal Proc., page(s) 799-803, May 1990.
10. D.A Johns; W.M. Snelgrove, and A.S. Sedra, "Continuous-Time LMS Adaptive Recursive Filters", IEEE Trans. Circuit Syst., page(s): 169-178, July 1991.
11. K.A. Kozma, D.A. Johns, A.S. Sedra, "Automatic Tuning of Continuous-Time Integrated Filters Using an Adaptive Filter Technique, IEEE Trans. Circuit Syst., page(s): 1241-1248, Nov. 1991.
12. R. Mooney et al., "A 900 Mb/s bidirectional signaling scheme", IEEE Journal of Solid-State Circuits, vol.30, no.12, page(s):1538-1543, Dec. 1995.
13. M. Galles et al., "Spider: a high-speed network interconnect:, IEEE Micro, vol.17, no.1, page(s): 34-39, an.-Feb. 1997.
14. J. Kuskin et al., "The Stanford FLASH Multiprocessor", Proceeding of the 21st International Symposium on Computer Architecture, Chicago, IL, page(s) 302-313.
15. N. Kushiyama et al., "A 500-megabyte/s data-rate 4.5 M DRAM", IEEE Journal of Solid-State Circuits, vol.28, no.4, page(s): 490-498, Apr. 1994
16. R.C. Foss et al., "Memory-fast interfaced for DRAMs", IEEE Spectrum, vol.29, no.10, page(s):54-57, Oct. 1992.
17. K.Y. Chang et al., "A 2-Gb/s Asymmetric Link for High-Bandwidth Packet Switches", Hot Interconnects V Symposium Record, Digest of technical papers, page(s): 171-179, Aug. 1997.
18. P.S.R. Diniz, "Adaptive Filtering Algorithms and Practical Implementation", second edition, Kluwer Academic Publishers, 2002
19. T. Peng, "VLSI Implementation of High-Speed Cellular Automate Encryption Algorithm for Data Communications", Master thesis, March 1999, Carleton University.
20. K. Dalmia, "High Performance Backplane Design Using the Marvel AlaskaTM X Quad 3.125 Gb/s SERDES", Marvel white paper, Nov. 2001.
21. K. Donnelly, "A highly flexible 10 Gbps serial link cell that extends the life of the copper backplane", Rambus Inc., Nov. 2002.
22. A. Carusone and D.A. Johns, "Analogue adaptive filters: past and present", Circuits, Devices and Systems, IEE Proceedings-vol. 147, pp. 82-90, Feb. 2000.

23. S. Haykin, "Adaptive Filter Theory", third edition, Prentice Hall, 1996.
24. G.Uehara, C.Wong, J.Rudell, & P.Gray, "A 50MHz 70mW 8-tap Adaptive Equalizer/Viterbi Sequence Detector in 1.2um CMOS", (HTML/MIF/PS) Custom Integrated Circuits Conference, June 1994.
25. J.L. Moschner, "Adaptive filter with clipped input data", Stanford Laboratory Report 6796-1, June 1970.
26. D. Hirsch and W. Wolf, "A simple adaptive equalizer for efficient data transmission", IEEE Trans. Commun., vol. COM-18, pages: 5, 1970.
27. S. Milijevic, "Eye-opening and Adaptive Eye-opening circuits", presentation, 2002.

Appendix A: Schematics Views of Top Modules

Appendix A.1: Schematic view of the top module with IO pads

Appendix A.2: Schematic view of the round-robin adaptation engine top module

Appendix A.3: Schematic view of the parallel adaptation engine top module

Appendix B: Pre-Layout Timing Reports

Following is the worst path timing report from Synopsys `design_analyzer` for the round-robin adaptation engine:

```
*****
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : thesischip
Version: 2001.08-SP2
Date   : Wed Jul 16 15:36:27 2003
*****
```

Operating Conditions: WCCOM Library: tpz973gwc
Wire Load Model Mode: segmented

```
Startpoint: top/tap_sequencer1_count_block_reg_4_
             (rising edge-triggered flip-flop clocked by clk)
Endpoint:   top/tap_sequencer1_state_reg_1_
             (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type:  max
```

Des/Clust/Port	Wire Load Model	Library
thesischip	TSMC16K_Conservative	tpz973gwc
topmodule	TSMC8K_Conservative	tpz973gwc

Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (propagated)	0.62	0.62
top/tap_sequencer1_count_block_reg_4_/CK (SDFRPQ1)	0.00	0.62 r
top/tap_sequencer1_count_block_reg_4_/Q (SDFRPQ1)	0.39	1.02 r
top/U150/Z (INVD2)	0.07	1.08 f
top/U69/Z (NOR2D2)	0.16	1.24 r
top/U285/Z (NAN3D4)	0.10	1.33 f
top/U353/Z (INVD4)	0.08	1.42 r
top/U362/Z (NAN2D4)	0.06	1.48 f
top/U326/Z (INVD2)	0.11	1.59 r
top/U532/Z (NAN2D4)	0.07	1.65 f
top/U528/Z (NOR2D2)	0.18	1.83 r
top/U529/Z (INVD1)	0.07	1.89 f
top/U572/Z (MUX2D2)	0.31	2.21 f
top/tap_sequencer1_state_reg_1_/D (SDFRPB1)	0.00	2.21 f
data arrival time		2.21
clock clk (rise edge)	2.00	2.00
clock network delay (propagated)	0.62	2.62
clock uncertainty	-0.05	2.57
top/tap_sequencer1_state_reg_1_/CK (SDFRPB1)	0.00	2.57 r
library setup time	-0.36	2.21

data required time	2.21
data required time	2.21
data arrival time	-2.21
slack (MET)	0.00

The worst path timing report from Synopsys **design_analyzer** for the parallel adaptation engine is following:

```
*****
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : thesischip
Version: 2001.08-SP2
Date   : Wed Jul 16 17:20:27 2003
*****
```

Operating Conditions: WCCOM Library: tpz973gwc
Wire Load Model Mode: segmented

```
Startpoint: top/msg_gen_counter1_dff1_data_out_tmp_reg
            (rising edge-triggered flip-flop clocked by clk)
Endpoint:  top/msg_gen_counter1_count_out_tmp_reg_3_
            (rising edge-triggered flip-flop clocked by clk)
Path Group: clk
Path Type: max
```

Des/Clust/Port	Wire Load Model	Library
thesischip	TSMC16K_Conservative	tpz973gwc
topmodule	TSMC8K_Conservative	tpz973gwc

Point	Incr	Path
clock clk (rise edge)	0.00	0.00
clock network delay (propagated)	0.65	0.65
top/msg_gen_counter1_dff1_data_out_tmp_reg/CK (SDFRPB1)	0.00	0.65 r
top/msg_gen_counter1_dff1_data_out_tmp_reg/Q (SDFRPB1)	0.52	1.17 f
top/U1018/Z (BUFD7)	0.18	1.34 f
top/U629/Z (NAN2D4)	0.09	1.43 r
top/U2/Z (INVD2)	0.07	1.50 f
top/U503/Z (NOR2D4)	0.07	1.57 r
top/U394/Z (EXNOR2D1)	0.30	1.87 r
top/U395/Z (INVD0)	0.06	1.93 f
top/U969/Z (MUX2D2)	0.29	2.22 f
top/msg_gen_counter1_count_out_tmp_reg_3_/D (SDFRSPQ1)	0.00	2.22 f
data arrival time		2.22
clock clk (rise edge)	2.00	2.00
clock network delay (propagated)	0.65	2.65
clock uncertainty	-0.05	2.60
top/msg_gen_counter1_count_out_tmp_reg_3_/CK (SDFRSPQ1)	0.00	2.60 r
library setup time	-0.38	2.22
data required time		2.22

data required time	2.22
data arrival time	-2.22
<hr/>	
slack (MET)	0.00

Appendix C: Post-Layout Timing Reports

File View Windows			Real Summary Timing only (50)		Help
Possibilities	Max Histogram	Min Histogram			
#	Constraint	Stack Reason			
1	Setup	0.55ns top/msg_gen_counter1_count_out_tmp_reg_4_ (SDFRPQ1 D A -> CK A)			
2	Setup	0.59ns top/msg_gen_counter3_count_out_tmp_reg_5_ (SDFRPB1 D A -> CK A)			
3	Setup	0.59ns top/msg_gen_counter3_count_out_tmp_reg_5_ (SDFRPB1 D V -> CK A)			
4	Setup	0.60ns top/msg_gen_counter1_count_out_tmp_reg_4_ (SDFRPQ1 D V -> CK A)			
5	Setup	0.60ns top/msg_gen_counter1_count_out_tmp_reg_3_ (SDFRSPQ1 D A -> CK A)			
6	Setup	0.61ns top/udcount_count_out_reg_10_ (SDFRPB1 D V -> CK A)			
7	Setup	0.61ns top/msg_gen_counter2_count_out_tmp_reg_5_ (SDFRPB1 D V -> CK A)			
8	Setup	0.61ns top/msg_gen_counter1_count_out_tmp_reg_3_ (SDFRSPQ1 D V -> CK A)			
9	Setup	0.61ns top/msg_gen_counter2_count_out_tmp_reg_5_ (SDFRPB1 D A -> CK A)			
10	Setup	0.64ns top/tap_sequencer1_state_reg_1_ (SDFRPB1 D V -> CK A)			
11	Hold	0.42ns top/tap_sequencer1_count_block_reg_0_ (SDFRPB2 D A -> CK A)			
12	Hold	0.42ns top/dff5_dff2_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
13	Hold	0.42ns top/tap_sequencer1_count_block_reg_0_ (SDFRPB2 D V -> CK A)			
14	Hold	0.43ns top/dff6_dff2_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
15	Hold	0.43ns top/dff2_dff3_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
16	Hold	0.43ns top/dff5_dff1_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
17	Hold	0.43ns top/dff4_dff3_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
18	Hold	0.43ns top/dff5_dff3_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
19	Hold	0.44ns top/msg_gen_dff1_dff2_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
20	Hold	0.44ns top/dff2_dff2_data_out_tmp_reg (SDFRPQ1 D A -> CK A)			
21	Width	0.60ns top/msg_gen_counter1_dff1_data_out_tmp_reg (SDFRPB2 CK low)			
22	Width	0.60ns top/msg_gen_counter2_dff1_data_out_tmp_reg (SDFRPB2 CK low)			
23	Width	0.60ns top/msg_gen_counter4_dff1_data_out_tmp_reg (SDFRPB2 CK low)			
24	Width	0.60ns top/msg_gen_counter3_dff1_data_out_tmp_reg (SDFRPB2 CK low)			
25	Width	0.60ns top/tap_sequencer1_count_block_reg_0_ (SDFRPB2 CK low)			
26	Width	0.60ns top/udcount_count_out_reg_10_ (SDFRPB1 CK low)			
27	Width	0.60ns top/udcount_count_out_reg_9_ (SDFRPB1 CK low)			
28	Width	0.60ns top/latches_tap_adjustments_tmp_reg_3_ (SDFRPB1 CK low)			
29	Width	0.60ns top/latches_tap_adjustments_tmp_reg_1_ (SDFRPB1 CK low)			
30	Width	0.60ns top/latches_tap_adjustments_tmp_reg_2_ (SDFRPB1 CK low)			

Appendix C.1: Timing report from Pearl for the round-robin adaptation engine

File View Windows			Pearl Summary: TimingVerity (30)	Help
Possibilities: Max Histogram Min Histogram				
#	Constraint	Block Reason		
1	Setup	0.58ns	top/udcount2_count_out_reg_9_ (SDFRPB1 D V -> CK ^)	
2	Setup	0.58ns	top/udcount3_count_out_reg_9_ (SDFRPQ1 D V -> CK ^)	
3	Setup	0.58ns	top/udcount3_count_out_reg_3_ (SDFRPQ2 D V -> CK ^)	
4	Setup	0.58ns	top/udcount3_count_out_reg_10_ (SDFRPB1 D V -> CK ^)	
5	Setup	0.59ns	top/msg_gen_counter1_count_out_tmp_reg_4_ (SDFRPQ1 D ^ -> CK ^)	
6	Setup	0.59ns	top/udcount2_count_out_reg_10_ (SDFRPB1 D V -> CK ^)	
7	Setup	0.59ns	top/msg_gen_counter2_count_out_tmp_reg_5_ (SDFRPB1 D V -> CK ^)	
8	Setup	0.59ns	top/udcount4_count_out_reg_9_ (SDFRPB1 D V -> CK ^)	
9	Setup	0.60ns	top/msg_gen_counter3_count_out_tmp_reg_5_ (SDFRPB1 D V -> CK ^)	
10	Setup	0.60ns	top/msg_gen_counter2_count_out_tmp_reg_5_ (SDFRPB1 D ^ -> CK ^)	
11	Hold	0.43ns	top/msg_gen_dff1_data_out_tmp_reg (SDFRPQ1 D ^ -> CK ^)	
12	Hold	0.45ns	top/tap_sequencer1_count_block_reg_0_ (SDFRPB1 D V -> CK ^)	
13	Hold	0.45ns	top/tap_sequencer1_count_block_reg_0_ (SDFRPB1 D ^ -> CK ^)	
14	Hold	0.46ns	top/sr1_data_out_tmp_reg_2_ (SDFRPQ1 D ^ -> CK ^)	
15	Hold	0.46ns	top/tap_sequencer1_count_block_reg_7_ (SDFRPB1 SD ^ -> CK ^)	
16	Hold	0.46ns	top/msg_gen_counter4_dff1_data_out_tmp_reg (SDFRPB2 SD ^ -> CK ^)	
17	Hold	0.46ns	top/udcount4_count_out_reg_0_ (SDFRPQ2 SD ^ -> CK ^)	
18	Hold	0.46ns	top/sr1_data_out_tmp_reg_1_ (SDFRPQ1 D ^ -> CK ^)	
19	Hold	0.46ns	top/dff7_data_out_tmp_reg (SDFRPQ1 D ^ -> CK ^)	
20	Hold	0.47ns	top/sr1_data_out_tmp_reg_4_ (SDFRPQ1 D ^ -> CK ^)	
21	Width	0.60ns	top/dff3_data_out_tmp_reg (SDFRPB2 CK low)	
22	Width	0.60ns	top/udcount1_count_out_reg_4_ (SDFRPB2 CK low)	
23	Width	0.60ns	top/dff2_data_out_tmp_reg (SDFRPB2 CK low)	
24	Width	0.60ns	top/dff5_data_out_tmp_reg (SDFRPB2 CK low)	
25	Width	0.60ns	top/dff4_data_out_tmp_reg (SDFRPB2 CK low)	
26	Width	0.60ns	top/msg_gen_counter4_dff1_data_out_tmp_reg (SDFRPB2 CK low)	
27	Width	0.60ns	top/msg_gen_counter3_dff1_data_out_tmp_reg (SDFRPB2 CK low)	
28	Width	0.60ns	top/msg_gen_counter2_dff1_data_out_tmp_reg (SDFRPB2 CK low)	
29	Width	0.60ns	top/udcount1_count_out_reg_7_ (SDFRPB1 CK low)	
30	Width	0.60ns	top/tap_sequencer1_count_block_reg_0_ (SDFRPB1 CK low)	

Appendix C.2: Timing report from Pearl for the parallel adaptation engine