DESIGNING OF LOW POWER HIGH SPEED CURRENT MODE LOGIC SERDES

M.Tech. Thesis

By

MOHIT SINGH CHOUDHARY

(1402102008)

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JULY, 2016

DESIGNING OF LOW POWER HIGH SPEED CURRENT MODE LOGIC SERDES

A THESIS

Submitted in partial fulfillment of the requirements for the award of the degree

of

Master of Technology

by

MOHIT SINGH CHOUDHARY

(1402102008)

DISCIPLINE OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JULY, 2016

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **DESIGNING OF LOW POWER HIGH SPEED CURRENT MODE LOGIC SERDES** in the partial fulfillment of the requirements for the award of the degree of **MASTER OF TECHNOLOGY** and submitted in the **DISCIPLINE OF ELECTRICAL ENGINEERING, Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from July, 2014 to July, 2016 under the supervision of Dr. S.K. Vishvakarma, Associate Professor, Discipline of Electrical Engineering, IIT Indore.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

Signature of the student with date
MOHIT SINGH CHOUDHARY

This is to certify that the above statement made by the candidate is correct to the best of my/our knowledge.

Signature of the Supervisor of
M.Tech. thesis (with date)
Dr. S.K. Vishvakarma

MOHIT SINGH CHOUDHARY has successfully given his M.Tech. Oral Examination held on **29TH June 2016**.

Signature(s) of Supervisor(s) of M.Tech. thesis
Date:

Convener, DPGC
Date:

Signature of PSPC Member
Dr. Ram Bilas Pachori
Date:

Signature of PSPC Member
Dr. Anirban Sengupta
Date:

Acknowledgments

Firstly, I would like to express my sincere gratitude to my advisor Dr. S. K. Vishvakarma for the continuous support, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I would like to thank all the faculty members of electrical department for their teaching and support throughout these two years.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr.Srivathsan Vasudevan, Dr.R.B. Pachori, and Dr.Anirben Sen Gupta, for their insightful comments and encouragement, but also for the hard question which helped me to widen my knowledge from various perspectives.

I must express my very profound gratitude to my fellow lab mates Bhupendra Renewal, Poran Singh, Vishal Sharma, Pramod k. Bharti, and Dr. N.K. Yadav for their support and valuable suggestions on my work. I specially wants to thank Mahesh Kumawat for his constant help in my project.

I also place on record, my sense of gratitude to all my friends, who have been with me, supported me, for all the fun we had and made my stay in IIT Indore delightful.

I also thank my family for the unceasing encouragement, support and love. Thank you.

Abstract

Transmission of data from a source to destination with error protection is the main aim of communication. Communication can be established in a parallel way or serial way. In parallel communication each bit of data require a separate line for transmission. But as according to the Moore's Law number of transistor density in increasing resulting in the growth of pin interconnect density. Thus the unrivaled solution is to use SerDes. SerDes uses serial communication for data transfer. The SerDes is used not only in backplane drivers but also in high speed IO interface such as chip-to-chip, chip-to-backplane and chip-to-memory. SerDes is becoming a common building block for ASICs. SerDes will become pervasive IO solution but also the characteristic and optimization of these links present new challenges.

With the increase in speed of communication and complexity of circuits more fast and compact SerDes are required. SerDes itself have its design concerns such as error correction, load matching at the receiver, area constants, speed and power consumption. In this work, our main aim is to reduce the power with optimal speed. Current Mode Logic is used because it has a power advantage over CMOS at high speed. It has great error immunity as compared to conventional methods.

Our work is divided in two parts. First to design a synchronous SerDes. Synchronous SerDes is designed using CML mux. A new parallel chain method is used at deserializer. This all help us to reach a speed of $16.64Gbps$ speed with only $9.29mW$. In second work, we designed a asynchronous SerDes, which was designed with the help of CML latch. Asynchronous since does not require clock, saves lot of power from PLLs and CDRs. The asynchronous circuit operates at $18.92Gbps$ consuming only $8.82mW$ power.

List of Publications

- Mohit S. Choudhary, Mahesh Kumawat, Pramod K. Bharti, and S. K. Vishvakarma, 16.64Gbps Synchronous CML SerDes Transceiver Design Technique with Process Corner Variations for Low Power Application, *IEEE VLSI Circuit System Letter*, Volume 2 Issue 1, Page 2-6, April 2016.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

In the modern society, communication plays an important role. Use of the computer is increasing which also enhances the need for data transfer. In Mobile communication, user needs are increasing day-by-day which results in development from 2G to 4G network. For various applications like voice call, video calls, Internet browsing, HD videos, Bio-Medical Signal processing all this require significant amount of data transfer. With the conventional parallel method of data transfer, require a large number of pins consuming more area and power. These all increases the research in the interconnects. The most suitable and adopted way of data transfer is serial communication. This growing demand has to be fulfilled by new circuit optimised for increased speed and reduced power.

1.2 Motivation

Increasing demand of interconnect speed has to be optimized for power consumption. Since interconnects are the physical medium, thus they have their limitations and losses. Interconnect IC's has to be designed by keeping all this constrains in mind.

- A interconnect IC must be able to communicate at higher speed
- Must have low power consumption.
- Must be able to minimise the distortion produced by channel and recover data properly.

- Must be backward compatible and most importantly must have a vast area of implementation.

Some of the examples where SerDes is used are HDMI, USB, Infiniband, MIPI, PCI Express, SATA, LTE etc.

The related work is done in [1], Sally *et. al.* using a three-level encoding technique they are able to achieve a data rate of $12Gbps$. They have designed a comparatively small phase detector which recovers data and clock from the received data. In [2], a modified three level coding is desirable to reduce the power consumption and decrease the error probability. They have designed a modified three-level encoder and decoder which is also able to recover the clock thus small circuit saves power and chip area. In [3] again they used half data rate self-timed three-level coding to increase the speed to $24Gbps$.

In [4] transformer coupled CML technique is used to reduce the power. In this technique, they have reduced the supply voltage in order to obtain the desirable power. Transformer coupled technique with enhanced drain-to-source voltage provides significant drain current per unit of width-to-length ratio, which provides large voltage swing reducing the effect of reduced power supply. In [5] and [6], Jri Lee and *et. al* used the NRZ and PAM4 encoding methods to increase the speed up to $56Gbps$. This NRZ and PAM4 are more error resistive and provide a better communication link.

Diego Tondo *et. al.* [7] used both CMOS and CML design methods to utilises the benefits of both CMOS at lower speed and CML at higher speed. CMOS perform well at low frequency therefore $16 \times n$ multiplexing is done using CMOS logic. Where as CML has advantages over CMOS at high frequency thus $n \times 1$ multiplexing is done by using CML. Since half the circuit is built using the CMOS approach it has comparatively smaller area. Asynchronous Serialization approach was explained by Bui Chinh Hien in [8]. They used conventional shift registers method to serialize and deserialize data. Since it is an asynchronous method, it has lesser power consumption as compared to synchronous method. They chain of tri-state inverters separated by the Delay Element to design SerDes.The Delay element is used to maintain proper bit width in asynchronous design methods. They are able to reach up to a speed of $3.9Gbps$.

In [9], Ashok Jaiswal uses a CML asynchronous method for SerDes designing. They have used CML-MUX for serializer/deserializer. By doing so, they gain a speed of $12.67Gbps$ at $14.3mW$ power consumption. Rostislan Dobkin in [10] used asynchronous method with LEDR encoder to reach up to a speed of $67Gbps$.

1.3 Summary of Contributions

Our main focus was on to design the circuit which consumes less power so that they can easily be used in mobile devices. Significant contributions of this thesis are listed as following

- Current Mode Logic Synchronous SerDes is designed with reduced power as compared with previous methods.
- CML based asynchronous SerDes is designed which has much less power consumption because of the asynchronous design method.

1.4 Organization of the Thesis

The thesis is organized as follows

- Chapter 2 give the detailed study of Current Mode Logic. Since in whole of our work we have used CML method, therefore starting from basic and advantages, few CML basic components are also discussed.
- Chapter 3 give the study of why we need SerDes. What are its advantages, disadvantages? Some of the application are given.
- Chapter 4 has a detailed view of Serializer and Deserializer. Their types i.e. synchronous and asynchronous SerDes. The basic component of a SerDes transceiver. Serialization and Deserialization methods. . Moreover, previous studies were done in the field of SerDes are explained.
- Chapter 5 tells about the various protocol which defines the working of SerDes.
- Chapter 6 gives the proposed CML Synchronous SerDes. In this chapter, our work on synchronous SerDes is represented with the block diagram of proposed serialize and deserializer. Timing diagram to explain serialization is also given. Circuit designed in Cadence Virtuoso is also given.
- Chapter 7 tells about the proposed CML asynchronous SerDes. CML-latch is used to design SerDes. The control block is also explained in detail whose function is to control the working of asynchronous SerDes.

- Chapter 8 gives the results of proposed synchronous and asynchronous methods in chapter 6 and chapter 7.
- Chapter 9 gives the conclusion and future work of the proposed methods.

1.5 Publication List

- Mohit S. Choudhary, Mahesh Kumawat, Pramod K. Bharti, and S. K. Vishvakarma, 16.64Gbps Synchronous CML SerDes Transceiver Design Technique with Process Corner Variations for Low Power Application, IEEE VLSI Circuit System Letter, Volume 2 Issue 1, Page 2-6, April 2016.

Chapter 2

Current Mode Logic

2.1 Overview

In this chapter, Current Mode Logic (CML) is explained with its comparison of conventional design techniques. Their implementation using CMOS technology is discussed. We will see the designing of the basic inverter, mux and latch using the CML.

2.2 Current Mode Logic

2.2.1 Basics of CML

Until now most convenient approach adopted for circuit design is CMOS. CMOS has many advantages such as zero static power dissipation, high level of integration, comparative easy circuit design. However, as the speed of communication increases, CMOS cannot parallely cope with the higher speed and low power requirement. By increasing W/L ratio of CMOS, we can increase the speed but along side, it also increases the power consumption. Additionally, at higher frequencies CMOS produces the inexorable amount of noise, thus integration of sensitive analog and digital circuit on the same chip become difficult [11].

MOS Current Mode Logic is a fully differential logic style which overcomes high-speed limitations of CMOS. CML has much less power consumption at high-speed as compared with the CMOS. CML is widely accepted in high speed circuit design applications such as optical communication. [12] CML circuits as compared with CMOS has varies characteristics such as low supply noise generation, high noise immunity and low power consumption at high frequencies enabling merger of analog and digital circuits on the same

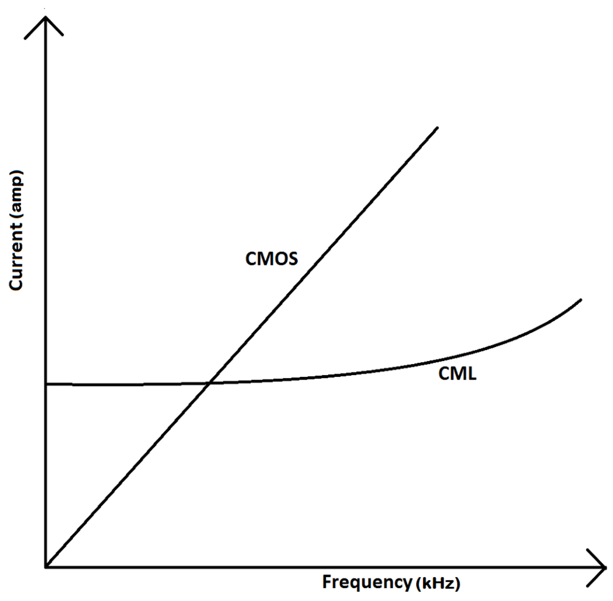

Figure 2.1: Current in CMOS and CML for increasing frequency.

chip [13].

As shown in figure 2.1 CMOS current is directly proportional to the frequency; it consumes very less power at low frequencies and power consumption increases with frequency. Whereas as shown in figure 2.1 [13]. CML current changes very less with frequency making it ideal to use at high frequencies [12]. Table 2.1 shown the tabular comparison of CMOS, CML and Bipolar CML [13].

Table 2.1: Tabular comparison of CMOS and CML

Logic type	*Noise performance*	*Maximum speed*	*Power Dissipation*
CMOS	Bad	Moderate	Low at low frequency, high at high frequency
CML	Good	High	High at low frequency, Low at high frequency
Bipolar CML	Excellent	Very High	High at low frequency, Low at high frequency

2.2.2 Advantages and Disadvantages of CML

Benefits of CML over CMOS are as follows:

- Fully Differential working.
- Low Voltage swing is making it suitable for high-speed circuits.
- The weak dependence of propagation delay on fanout load capacitance.
- High noise immunity as compared with CMOS.
- CML draws constant current from power source due to Current source.
- Switching noise is very less.
- Power dissipation is very less as compared with CMOS, especially at high frequency.

Disadvantages of CML over CMOS

- CML has a current source, thus has a static power dissipation.
- More complicated design process.
- Comparably more design parameters than CMOS.
- PMOS (resistor) require large circuit area.
- A large number of interconnect (usually double than CMOS) due to the differential method.

2.3 CML Circuit Design

2.3.1 Basic Component

CML consist of basic 3 circuit component as shown in figure 2.2. They are

1. Load resistance
2. Pull down network
3. Constant current source.

Load resistance can be designed using an active PMOS. PMOS source-gate voltage equal to V_{DD} and a much smaller source-drain voltage, thereby forcing PMOS in triode region in which it can act as resistance. The output voltage is taken across these load resistance. The pull-down network consists of logics created with

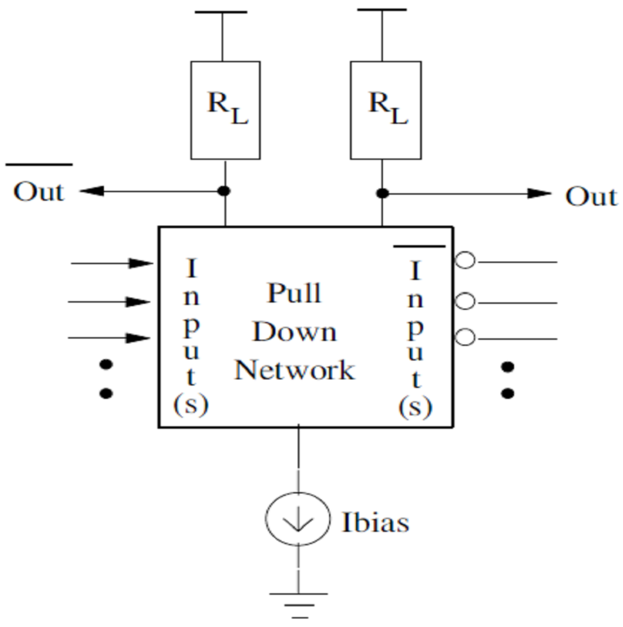

Figure 2.2: Basic CML circuit component.

NMOS. Since CML is a fully differential network, logic circuit and its complement circuit are designed in pull-down network (PDN) using only NMOS. Because of this circuit area increases but differential circuit become more immune to noise [14].

The I_{bias} is a constant current source provided with the help of NMOS current mirror circuit. For simplicity in the circuit diagram, a current mirror is represented by a simple current source in this report. Current from this source depending on the logic provided flows through on of the branches of PDN. Since the voltage swing is less in CML, it has less dynamic power consumption, and its speed is fast as it only uses NMOS transistors [14].

2.3.2 CML Inverter

The circuit diagram of CML Inverter is shown in figure 2.3. It consists of a constant current source I_{ss}, PMOS in always ON condition and a pull-down network designed for inverter operation.

CML operation can be explained by assuming them in the saturation region. Current i_{D1} and i_{D2} along

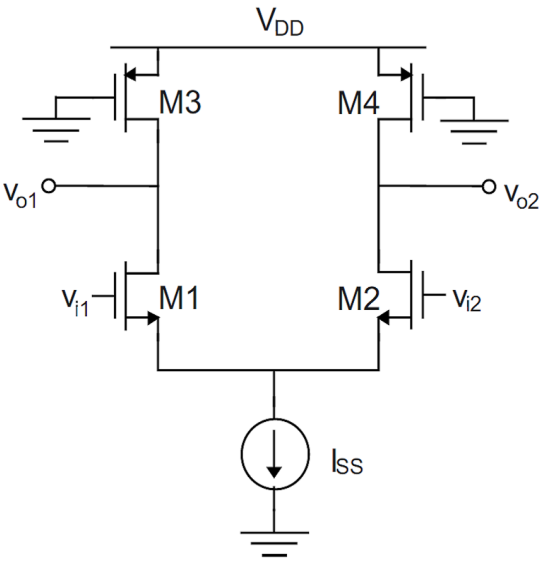

Figure 2.3: Current Mode Inverter.

transistor $M1$ and $M2$ can be expressed as the function of the differential input voltage $v_i = v_{i1} - v_{i2}$ as

$$i_{D1}(v_i) = \begin{cases} 0 & \text{if } v_i < -\sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{Ln}}} \\ \frac{I_{ss}}{2} + \frac{v_i}{2}\sqrt{\mu_n C_{OX}\frac{W_n}{Ln}I_{ss} - \left(\mu_n C_{OX}\frac{W_n}{Ln}\frac{v_i}{2}\right)^2} & \text{if } |v_i| \le \sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{Ln}}} \\ I_{ss} & \text{if } v_i > \sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{Ln}}} \end{cases} \tag{2.1}$$

$$i_{D2}(v_i) = I_{ss} - i_{D1}(v_i) \tag{2.2}$$

where W_n and L_n are the effective CMOS channel dimensions, C_{OX} oxide capacitance per area, μ_n carrier mobility of NMOS [15].

From eq. 2.6 it is clear that current I_{ss} steer through one of the branches whose input voltage is v_i is greater then $\sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{Ln}}}$. The branch which is OFF gives V_{DD} as output and branch where I_{ss} steers gives V_{DD} - $I_{ss}R_D$ as output, where R_D is the resistance offered by the active PMOS. Here active PMOS act as a current-to-voltage converter. BSIM3v3 MOSFET model can evaluate R_D of PMOS. Drain current for PMOS by BSIM3v3 is given by

$$i_D = \frac{I_{DAST0}}{1 + R_{DS}\frac{I_{DAST0}}{V_{SD}}} \tag{2.3}$$

where $R_{DS} = \frac{R_{DSW}\times 10^{-6}}{W}$ depends on model parameter R_{DSW} which is source/drain parasitic resistance [15]. in eq. 2.3 I_{DAST0} can be calculated by solving current equation given by BSIM3v3 model and assuming V_{SD} to be small, then I_{DAST0} is given by

$$I_{DSAT0} = \frac{V_{SD}}{R_{int}} \tag{2.4}$$

here R_{int} is intrinsic resistance given by

$$R_{int} = \frac{1}{\mu_{eff,p} C_{OX} \frac{W_p}{L_p}(V_{DD} - |V_{T,p}|)} \tag{2.5}$$

eq. 2.3 can be expanded by Taylor's series and neglecting higher order terms we get

$$i_D = I_{DSAT0}\left(1 - \frac{R_{DS}}{R_{int}}\right) \tag{2.6}$$

the equivalent resistance of the PMOS transistors is $R_D = V_{SD}/i_D$, from eq. 2.4 and eq. 2.6 R_D will be

$$R_D = \frac{R_{int}}{1 - \frac{R_{DS}}{R_{int}}} \tag{2.7}$$

Thus the differential output voltage across R_D will be

$$v_o = v_{o1} - v_{o2} = -R_D(i_{D1} - i_{D2}) \tag{2.8}$$

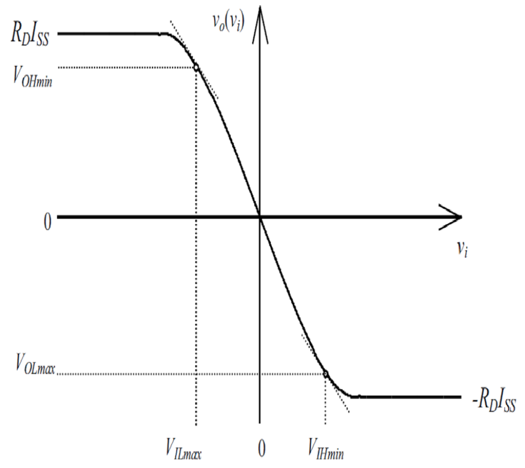

Figure 2.4: Transfer characteristics of CML Inverter.

From transistor current in eq 2.1 and substituting them in eq. 2.7

$$v_o(v_i) = \begin{cases} R_D I_{ss} & \text{if } v_i < -\sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{L_n}}} \\ -v_t R_D I_{ss}\sqrt{\frac{\mu_{eff,n} C_{OX}}{I_{ss}}\frac{W_n}{L_n} - \left(\frac{\mu_{eff,n} C_{OX}}{2I_{ss}}\frac{W_n}{L_n} v_i\right)^2} & \text{if } |v_i| \le \sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{L_n}}} \\ -R_D I_{ss} & \text{if } v_i > \sqrt{\frac{2I_{ss}}{\mu_n C_{OX}\frac{W_n}{L_n}}} \end{cases} \tag{2.9}$$

Eq. 2.9 shows that

$V_{OL} = -R_D I_{ss}$ and $V_{OH} = -R_D I_{ss}$ thus the output swing is given by

$$V_{SWING} = V_{OH} - V_{OL} = 2R_D I_{ss} \tag{2.10}$$

In CML inverter figure 2.3 NMOS transistor $M1 - M2$ are kept out from triode region by maintaining gate-drain voltage lower than the threshold voltage.

$$V_{GD} = V_{DD} - [V_{DD} - R_D I_{ss}] = R_D I_{ss} \leq V_{T,n} \tag{2.11}$$

Figure 2.4 shows the transfer characteristics of a CML inverter [15].

2.3.3 CML Buffer

The circuit of CML buffer is same as that of an inverter in figure 2.3; the only difference is that the outputs are taken from the opposite terminals. CML buffer is used to boost the signal swings if their level degrades and also used in designing of Delay Element used in the asynchronous SerDes explained later in next chapters.

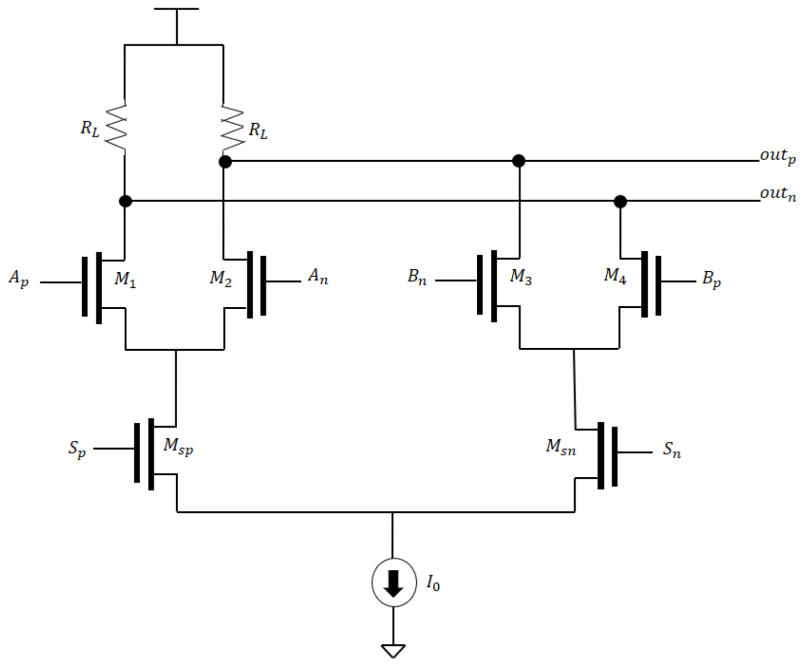

Figure 2.5: Current Mode Logic MUX (2×1).

2.3.4 CML Multiplexer (2×1)

The multiplexer is a combinational circuit which can be easily designed using current mode logic. Figure 2.5 shows the circuit diagram of CML MUX [9]. In the circuit, M_1 and M_2 are used as differential input for one of the MUX's inputs A_p and A_n and M_3 and M_4 as differential input for another input B_p and B_n. NMOS M_{sp} and M_{sn} are used as a differential select line for MUX.

The working operation can clearly be seen in the circuit that the current will steer through one of the differential branch depending on the value of the signal at M_{sp} and M_{sn}. When M_{sp} is high, and M_{sn} is low , M_1 and M_2 will be ON and output will follow the values at A_p and A_n. Similarly, when M_{sp} is low and M_{sn} is high, current will steer through M_3 and M_4 making output follow B_p and B_n. Current and voltage equations will be same as in eq. 2.1 and eq. 2.9.

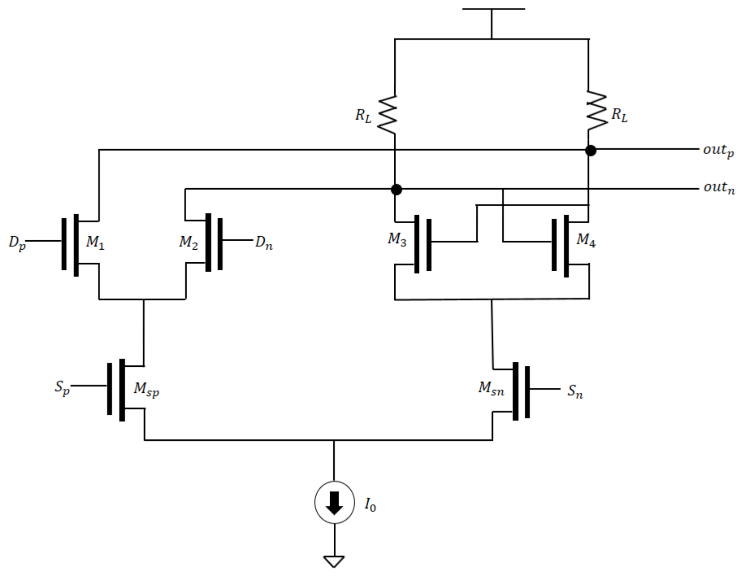

Figure 2.6: Current Mode Logic Latch.

2.3.5 CML Latch

The latch is a sequential circuit that can be designed by current mode logic using feedback. Circuit diagram for the latch is shown in figure 2.6 [9]. Its structure is somehow similar to that of CML Mux, only the second differential branch is feedback to the first. As shown in the figure 2.6 NMOS M_1 and M_2 are used as input and M_3 and M_4 maintains the output when the input clock is low. Here S_p and S_n are used as clock here. When S_p is high, and S_n is low, out_p and out_n will follow the input. However when S_p goes low and S_n

goes high, the second differential pair maintains the output to the same previous output. Current and voltage equations will be same as in eq. 2.1 and eq. 2.9.

During the positive clock cycle, i.e. S_p is high, and S_n is low output signal tracks the input signal. However when clock cycle becomes negative i.e. S_p is low, and S_n is high, CML latch enters the latching state and output will maintain the last output without affected by current changes in the input value.

In [16] and [17] new CML latch is designed since the conventional CML latch prone to function at high frequency. On cascading conventional CML latch during latching operation it may become incontinent to drive the load, and the voltage swing reduces to the undesired amount. To prevent such limitation, a novel latch is designed in [16] using a NMOS in parallel with the constant current source. Circuit for novel CML latch is shown in figure 2.7. In figure 2.7 NMOS M_7 is connected in parallel with the constant current

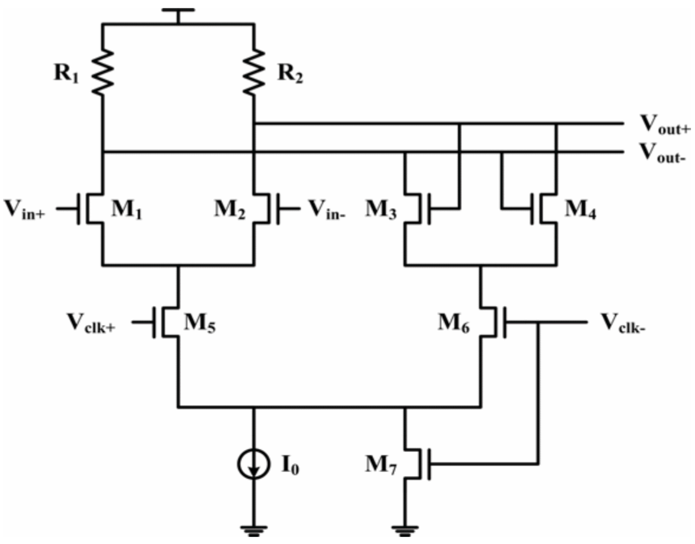

Figure 2.7: Novel CML Latch.

source I_0 and its gate is controlled by a negative part of the clock signal. When the clock goes low (V_{clk-} goes high), M_7 becomes ON and value of current increases to I(I_0) + I(M_7). This results in increasing the gain of latching branch and latch can work at higher frequencies. This novel CML latch is used to design asynchronous SerDes, which will be explained in the next chapters.

Chapter 3

Need of SerDes

3.1 Overview

This chapter gives the detail explanation of why SerDes is needed, its advantages, disadvantages, and applications.

3.2 Need for Serializer and Deserializer

When chip processes the data, that data need to be transferred from chip-chip or chip-device. The simplest way to transfer these data is the parallel connection, connect as many lines between chips as the size of data. In figure, 3.1 $chip_1$ is connected to $chip_2$ with n number of interconnects. Synchronized data is transferred between them as they are controlled by the same clock. This was the simplest way to transfer data between two chips. However, there are two problems with such type of connection. The first problem arises due to fast pace of the technological advancement of silicon technology. Interconnect pin density is not able to

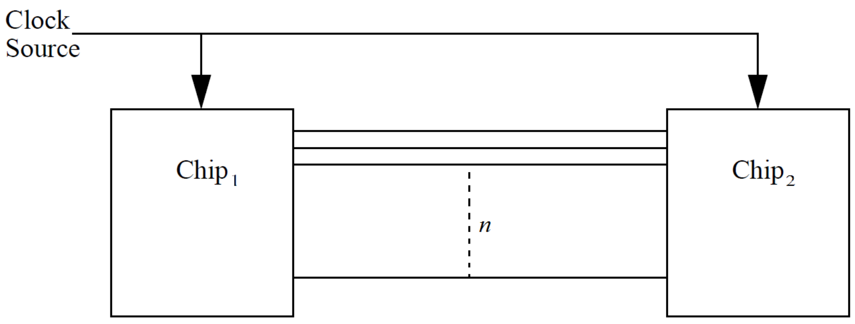

Figure 3.1: Parallel connection between two chips.

cope-up with the silicon technology. Therefore, interconnect pins are more expensive than silicon circuits. Thus dedicating this lines for parallel data is not acceptable for most application [18].

The second problem arises in meeting timing requirements. Data from $chip_1$ is transmitted synchronously and is received by $chip_2$ synchronously. Data at $chip_2$ must meet hold and setup times relative to the clock input of the chip. These hold and setup times should be calculated with permissible margin to allow for differences in the delay of the clock distribution path to the two chips. The delay may depend on chip process, voltage and temperature conditions, and an appropriate margin should be added to make communication possible. At high frequencies design such parallel interfaces become very difficult.

The best solution for these is to transmit data serially, i.e. the use of Serializer and Deserializer (SerDes) interface circuits. Some of the advantages of SerDes are: [19]

3.2.1 Advantages of SerDes

Maximum Data Flow

Let's show an example application that shows us how SerDes can help board designers system architecture

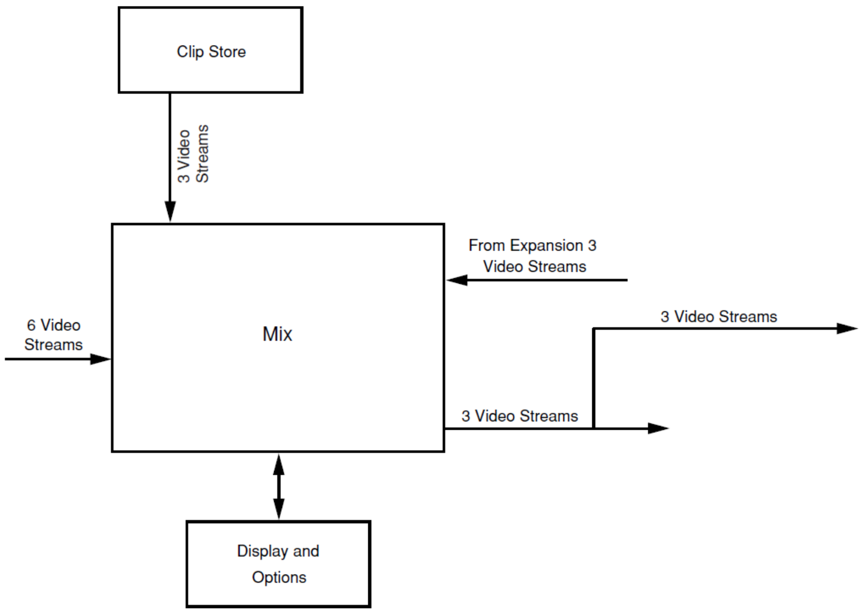

Figure 3.2: HD Video Mixer.

etc. In figure 3.2 HD video stream requires 1.5 Gb/s when transmitted in a uncompressed format. Figure 3.3 [19] shows building this system using deserializer and serializer for the serial video stream, and parallel interfaces for the expansion bus and chip store. Figure 3.4 [19] shown another method using gigabit transceivers inside the logic part to encode and decode the

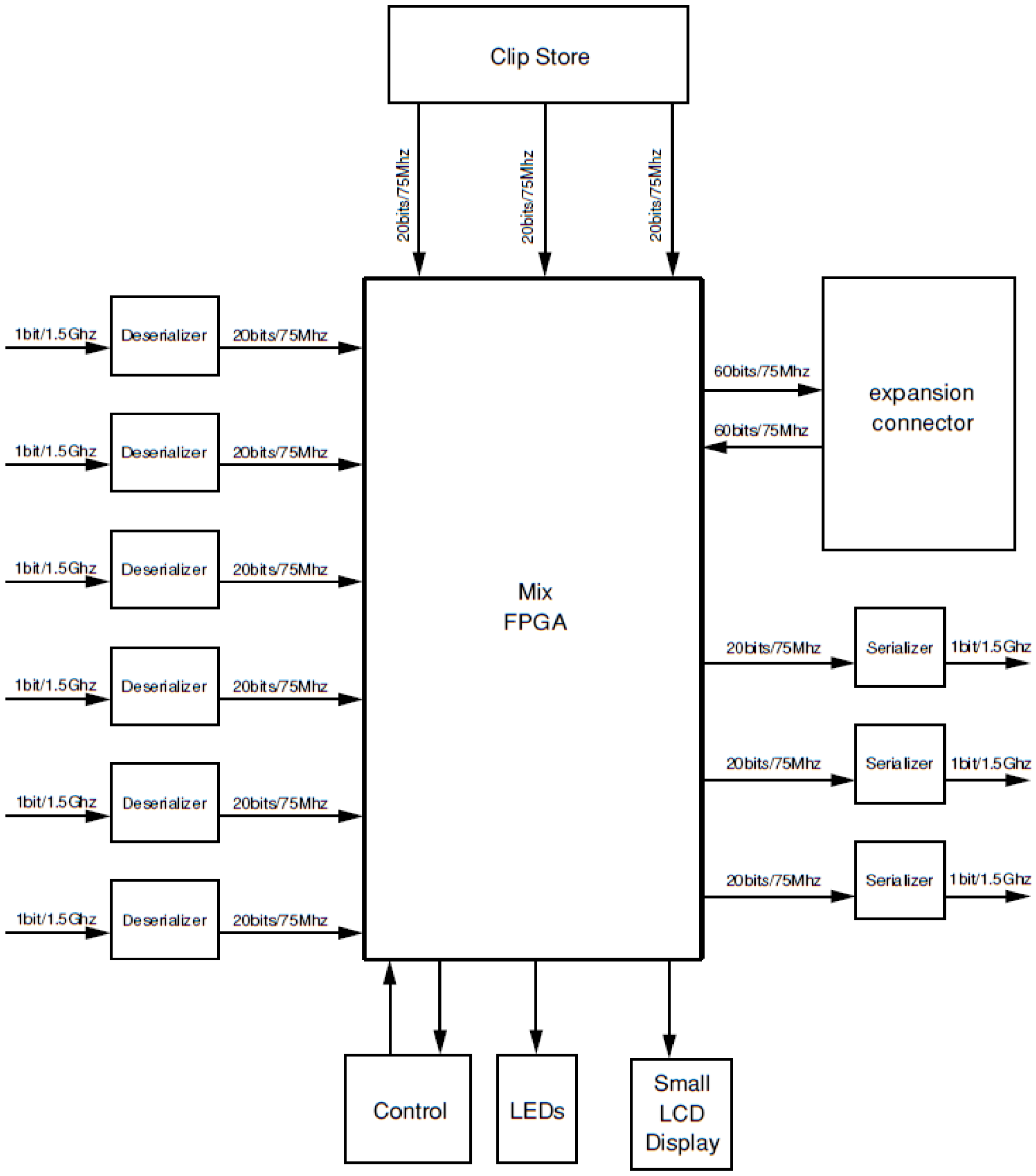

Figure 3.3: Deserializer/Serializer(parallel)

serial streams. The faster stream work as the interface to the expansion connectors and chip store.

Pin Count

Number of pins always a problem when significant data are transferred. The number of I/O pins are limited.

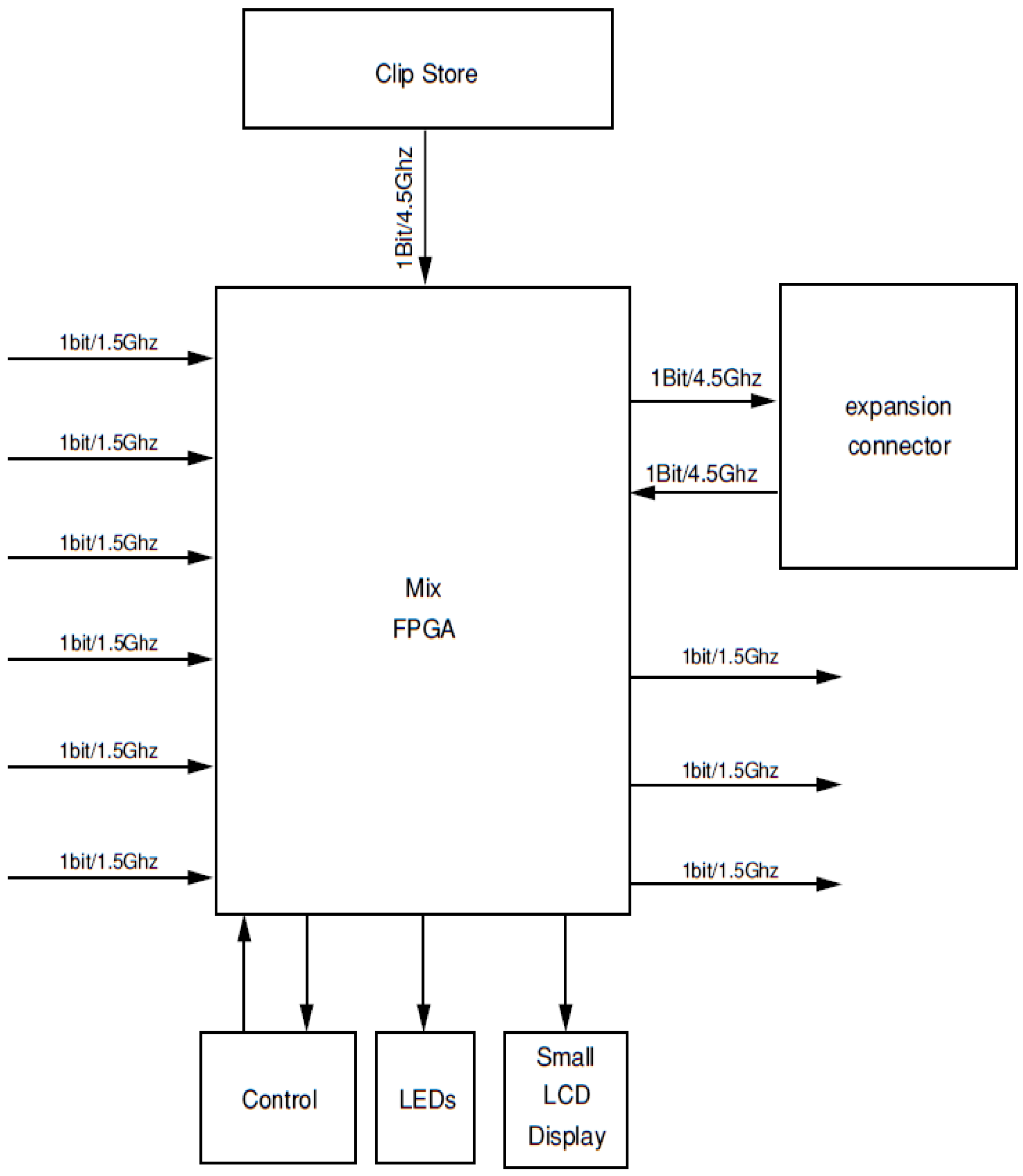

Figure 3.4: Gigabit Transceivers(SerDes)

Number of pins never cope up with the increasing demand arise due to ever increasing silicon technology.

It is evident from the table 3.1 [19] that a total number of pin required for the serial is much less than parallel. Also, cost reduces to a large extent when serial communication is used.

Table 3.1: Pin counts for figure.3.3 and figure.3.4

	Direction	*Parallel*	*Serial*
Inputs 1-6	IN	120	12
Clip store	IN	60	2
Expansion inputs	IN	60	2
Expansion outputs	OUT	60	2
Outputs 1-3	OUT	60	6
Control/ status In	IN	48	48
Control /status out	OUT	52	52
LEDs	OUT	12	12
LCD driver	OUT	48	48
Total		**520**	**184**

Simultaneous Switching Outputs When using parallel bus too many signals get toggle at the same time.If too many switches, ground bounce creates a lot of noise. Serial bus saves from this problem. [19]

EMI

As the frequency increases, emissions testing become more worse in parallel communication. however, serial link usually show very less radiation as compared to parallel links because of excellent signal integrity.

Cost

The serial link are much cheaper as compared to parallel links. A small, cheap package has less number of pins and board design will be simpler as well. Serial link also requires a fewer number of IC's.

3.2.2 Disadvantages of SerDes

To maintain the signal integrity requires to perform analog simulation and should use more complex bypassing schemes. Connectors and cables should be paid more for impedance matching and high-frequency operation. Since time base decreases more complication shows up.

3.2.3 Applications

Initially, the scope of SerDes was limited to the telecommunication industry. However as speed increases SerDes is accepted in the vast areas such as military, medical, video, communication, networking and much more. SerDes is used in chip-to-chip, chip-to-board, board-to-board/backplane and box-to-box communication. Some of the major industry standards where SerDes is used are:

- Fiber Channel
- Serial ATA
- PCI Express
- Rapid IO Serial
- Infiniband 1X,4X,12X
- Advanced Switching Interface
- 1-Gb Ethernet
- 10-Gb Ethernet (XAUI)

Chapter 4

Serializer and Deserializer

4.1 Overview

In this chapter, Serializer and Deserializer (SerDes) is explained with all its design parameters, application, and limitation. Types of SerDes which includes synchronous and asynchronous SerDes is explained. This chapter gives, a detail view about the working of SerDes. Previous work related to SerDes, synchronous and asynchronous SerDes.

4.2 SerDes

SerDes i.e. Serializer and Deserializer is an interface IC that converts n bit data into serial data at the transmitter and this serial data back to n it data at the receiver. Block diagram for SerDes is shown in figure 4.1 [18]. Transmitter side of the SerDes has Serializer, Equalizer, and line drivers. On the other side, receiver has Receiver, Clock and Data Recovery (CDR) and at last Deserializer. The internal structure of each block depends on the application where the SerDes is implemented. In many SerDes, an extra block of Line Coding is used before the serialization. This line coding is to ensure error detection and prevention. Thus at receiver, a line decoder will be used to recover coded data. Serializer block serializes the parallel data which may be encoded thus its input is in the multiple of 8 or 10. 10 input is for the 8b/10b encoder. These serialized data is then passed through an equalizer to prevent the losses caused by the transmission medium. Then these equalized serial data passed through line driver which help to increase the data power so it may easy pass through medium [18] [19].

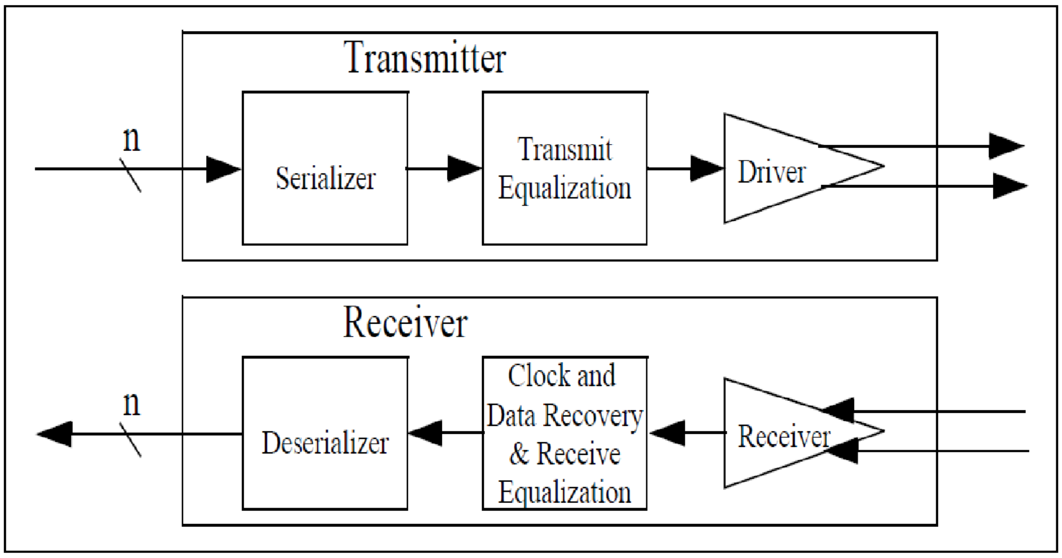

Figure 4.1: SerDes block diagram.

At the receiver clock and data recovery circuit is used to recover clock and data from the received stream of the bits. These data then passed through deserializer to recover back the parallel data.

Serializer

Serializer block in figure 4.1, converts n bit data in serial data which must be given as input to the equalizer. The value of n is in the multiples of 8 or 10. When the input is given in multiple of 8, it is useful for sending encoded data. When 1 bits are used it is a coded data such as 8B/10B coding.

In some SerDes first, the n bit data is multiplexed to k bit interconnect $(k < n)$ prior to the equalizer. Then after equalization these k bits are again serialized at driver stage [18]. In SerDes, the high clock is divided down to generate a clock for parallel data. This clock is used as a reference to serialize the data.

Equalizer

The channel between transmitter and receiver act as a filter at higher frequencies. Channel act as low pass filter and distort the signals at high frequency. Thus equalizer is used at transmitter or receiver to minimize the effect of the channel. Mathematically equalizer has the transfer function roughly equal to the inverse of the channel so that it can minimize the effect. Equalizer distorts the signal at the transmitter output such that the signal at receiver gets cleaned [18]. Mostly equalizer used in SerDes is FFE (Feed Forward Equalizer). The block diagram for a 3-tap Feed Forward Equalizer is shown in figure 4.2 [20] [21]. The serial data is delayed by each delay element and multiplied by each tap weight and final the summed if given as result

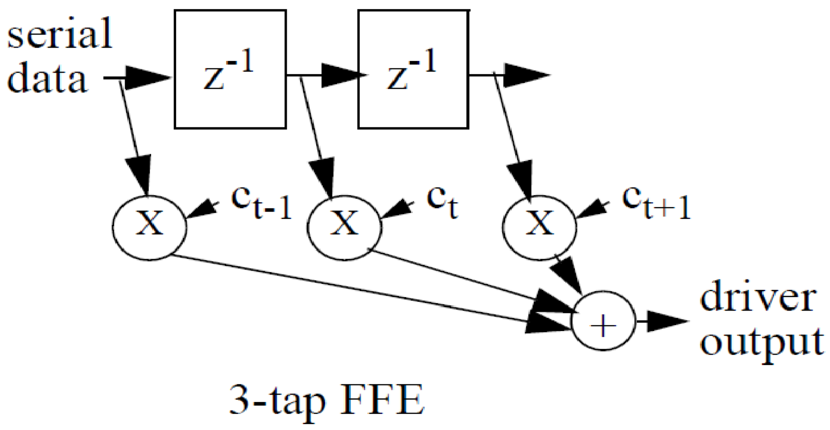

Figure 4.2: 3-tap Feed Forward Equalizer (FFE).

. Tap weights are selected to minimize the effect of the channel. Different algorithms are adopted to find the tap weight. In some SerDes, receiver equalizer is used in spite of transmitter equalizer. This receiver equalizer corrects the distortion produced by the channel [22]. Equalizer is generally of two types

- Passive Equalizer

  They consist of passive circuit element having transfer function inverse of the transmission line. They are considered as filters. [19]

- Active Equalizer

  They are frequency dependent amplifiers/attenuators. These are of two types- fixed pattern and self adjusting. Fixed pattern has same frequency response for all input frequencies whereas self-adjusting are adaptive and can adjust their tap-weight according to the input frequency. Self-adjusting area complicates to design and used for specific encoding schemes. [19]

Driver

The equalizer's output is provided to the driver which increases the power of the signal to minimize the jitters. Generally, differential drivers are used as they are superior to single-ended drivers. Different methods for differential signaling are - Low-Voltage Differential Signaling (LVDS), Low Voltage Pseudo Emitter-Coupled Logic(LVPECL), and Current Mode Logic (CML).

Receiver

Receiver receives the differential output of transmission line and gives it to the Clock and Data Recovery

(CDR) circuit. In many application receiver blocks also contain an error correcting logic, receiver equalizer (DFE) and linear amplifier.

Clock and Data Recovery (CDR)

When the output of the transmitter in one a single data wire or when the clock of serializer in accomplished with the data, but the receiver must process these data synchronously. Then CDR circuit is used to recover data, clock and timing information from the received stream of bits. Block diagram for basic CDR is shown in the figure 4.3. In figure 4.3 input noisy data is passed and in output, we get recovered data and recovered

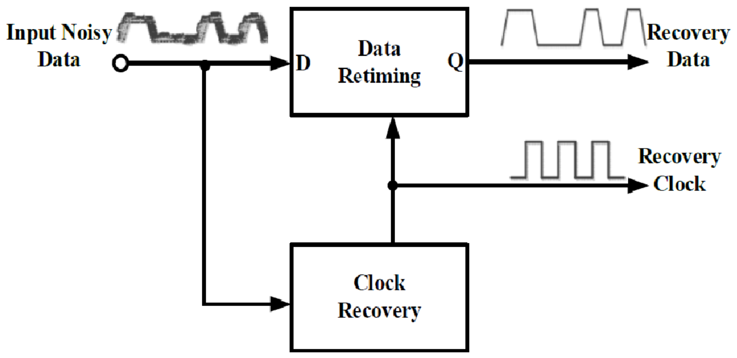

Figure 4.3: Block diagram of CDR

clock [23]. This recovered clock must have properties:

- The recovered clock frequency at the receiver must be equal to the frequency of the transmitter.
- Recovered clock should have minimum jitters.
- Recovered clock should have reasonable timing w.r.t the input data.

Deserializer

The function of Deserializer block is inverse of Serializer block. It convert back the serial data in to n bit parallel data. The clock for deserialization is generated by dividing down the internal high frequency clock and is supplied as an output for use by logic latching the parallel data.

4.3 Types of Serdes

On the basis of their working model SerDes are broadly classified in two categories i). Synchronous ii). Asynchronous

4.3.1 Synchronous SerDes

As the name suggest, synchronous SerDes require clock for their working. This reference clock is generated with the help of a Phase Loop Lock (PLL) which is a power consuming circuit. The basic block diagram for

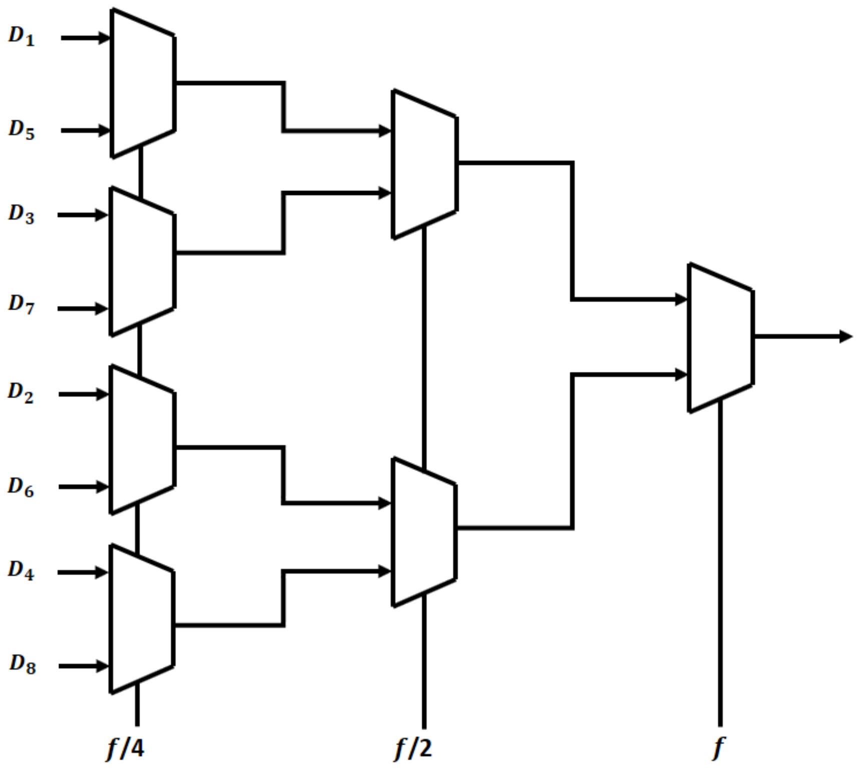

Figure 4.4: Block diagram of Synchronous Serializer

a synchronous serializer is shown in figure 4.4. This is a binary tree approach 2 bit are selected from one and so on the tree approaches. The main frequency f is divided internally by frequency divider to provide low frequencies to the previous stages of the tree. The blocks shown in figure 4.4 can be MUX, or a Double Edge-Triggered Flip-Flop depending on the circuit design.

Deserialization can be done with the help of block diagram shown in figure 4.5 and 4.6

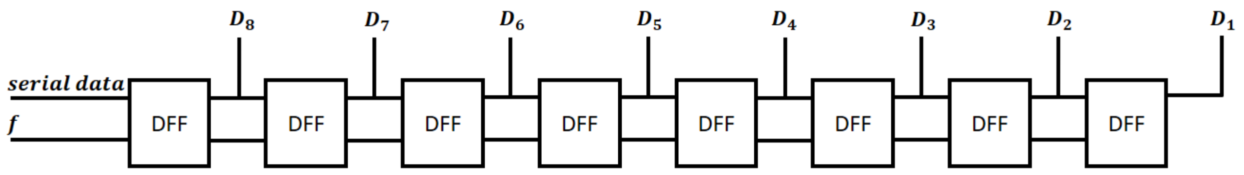

Figure 4.5: Block diagram of Synchronous Deserializer using single layer cascaded DFF

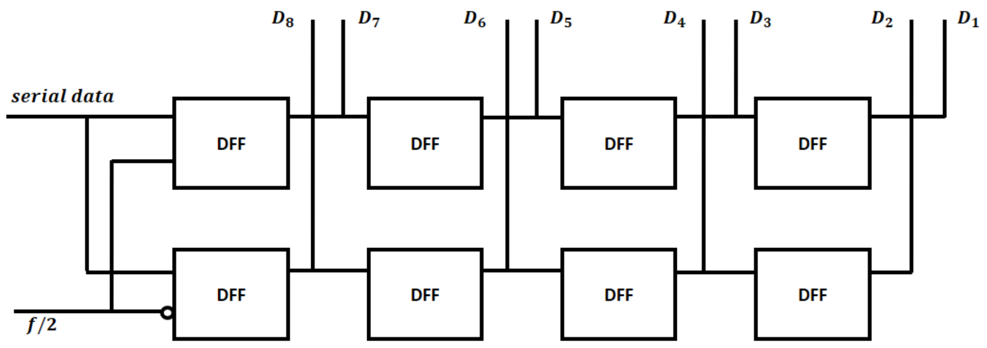

Figure 4.6: Block diagram of Synchronous Deserializer using double layer cascaded DFF

In figure, 4.5 D flip-flops are connected one after the other, the simplest way to convert serial data into parallel. Serial bits are passed through each shift registers with each clock and after eight clock cycle, parallel data is latched from their respective flip-flops. In this method, data is shifted at a rate of 1bit per cycle. The frequency used to operate this deserializer is f.

In the figure 4.6 shown another method to serialize the data. In this two parallel chains of D flip-flop are used to convert data to parallel. One branch is operational on rising edge of the clock and another branch of D flip-flop at the falling edge of the clock. Therefore bits are propagated through shift register at both rising edge and falling edge of the clock cycle. Frequency required for this operation is $f/2$.

These two are the basic methods for deserialization, other method such as inverse tree is also adopted for deserialization process.

In [1], synchronous method serialization is used. The Circuit is designed using C^2MOS logics. Serializer block diagram in [1] is similar to the above figure 4.4. In this paper, they have used double edge triggered D-flip flop, so that data can be transferred at both rising and falling edge of the clock. This paper proposed a new three-level encoding technique shown in figure 4.7. With the help of this coding technique, they are able to reduce the power and recover the clock at the receiver by using a simpler circuit.

This three-level encoding maintains the signal level at $V_{DD}/2$ independent of the data pattern. Because of this need of equalizer is eliminated. This technique require much smaller circuit as compares to CDR to recover clock at the receiver and also it is also insensitive to jitters. This circuit at $15.5mW$ power is providing a speed of $12Gbps$ in $65nm$ technology.

In [2] Hussein and et. al. used a modified three-level encoder and decoder to increase the speed. three-level coding is similar to that in [1]. The modified three-level encoder and decoder are shown in figure 4.8 and 4.9 [2]. This circuit is working on $16Gbps$ speed with $18.1mW$ power consumption.

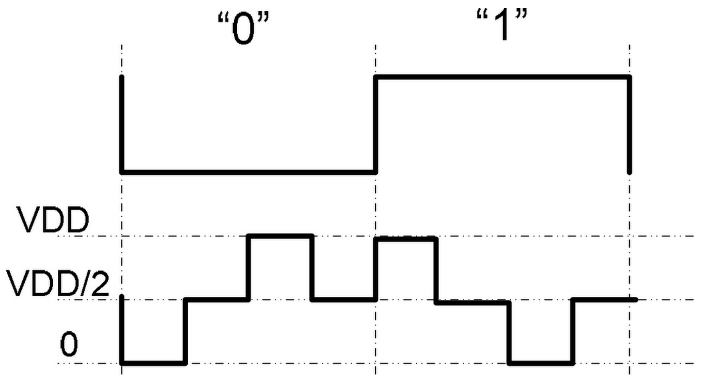

Figure 4.7: 3 level encoding technique

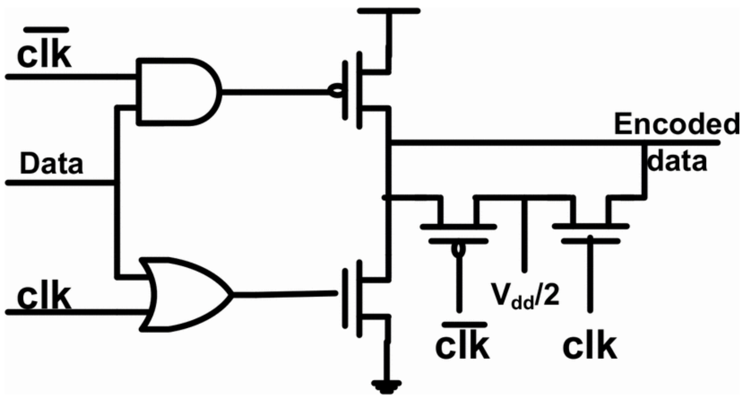

Figure 4.8: 3 level encoder

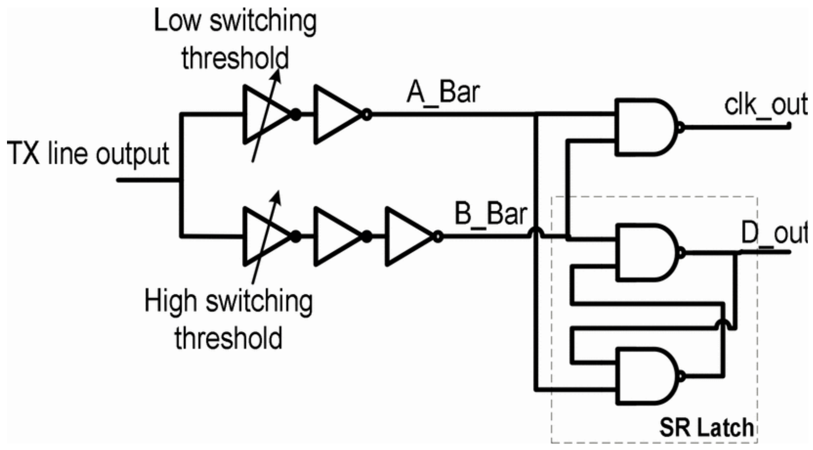

Figure 4.9: 3 level decoder

In [3] again a new three-level encoding scheme is used to increase the speed. This results in reduced dispersion and inter-symbol interference (ISI) because of shifted power spectrum. But this technique requires the frequency twice that of ordinary. Thus it is increasing the data rate to twice. Modified three-level encoding is shown in the figure 4.10. As shown in figure 4.10 data is divided into two 'A' and 'B' auxiliary

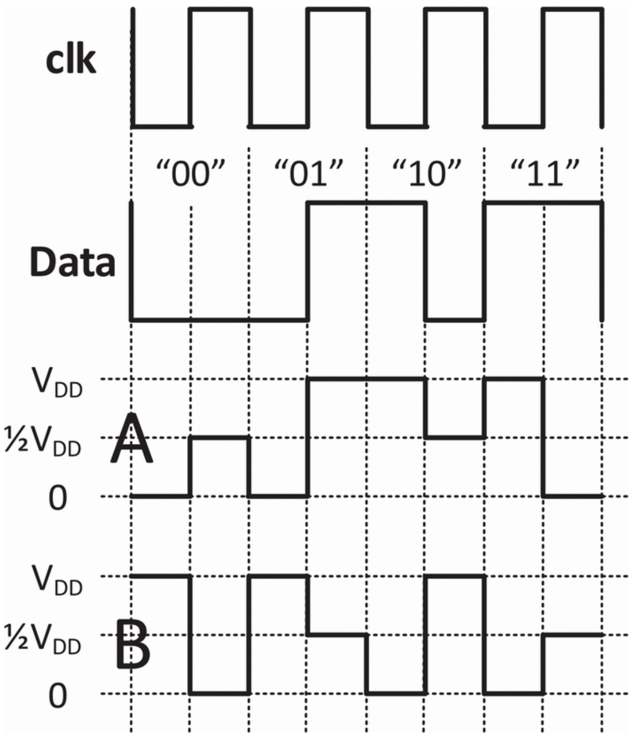

Figure 4.10: Modified 3 level coding

signals. Because of this auxiliary symbols, data rate of the circuit increases. Because of this speed of the circuit increases to $24Gbps$ consuming $109.6mW$ power.

In [24] and [25] 8B/10B encoding scheme is used for the serialization. It is a lane SerDes each lane working in a range from $1.065Gbps$ to $3.1875Gbps$. It is a part of $HX5000$ $Rad - Hard\ ASIC$ and can be used in box-to-box networks.

In [7] Tondo and et. al. used both CMOS and CML design technique to get the benefit of both. CMOS logic performs better at low frequency thus first data is multiplexed to $16 \times n$ using CMOS. Block diagram is shown in figure 4.11. Using CMOS, they get the advantages such as low static power dissipation, high noise margin, robustness and density, etc. CMOS is highly susceptible to environment noise such as supply and ground bounce and electromagnetic coupling. At high frequency its performance decreases. Thus in this circuit further $n \times 1$ multiplexing is done using CML logic which is fully differential and has advantages over CMOS at high frequency. It is 16×1 serializer which operates at $10Gbps$ speed.

In [4] Transformer Coupled (TC), the technique is used with CML to reduce the power consumption and increase speed. Transformer coupled id used to design latch and multiplexer for SerDes core. Transforming

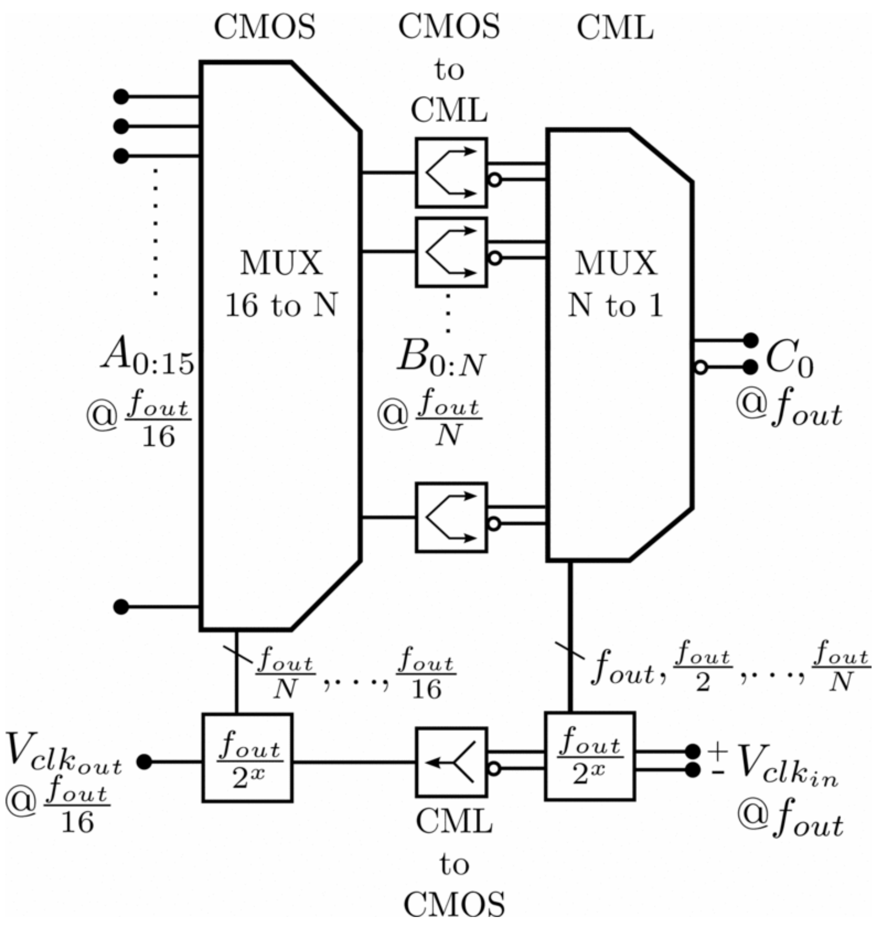
CMOS
CMOS to CML
CML
MUX 16 to N
MUX N to 1
$A_{0:15}$ @$\frac{f_{out}}{16}$
$B_{0:N}$ @$\frac{f_{out}}{N}$
C_0 @f_{out}
$\frac{f_{out}}{N}, \ldots, \frac{f_{out}}{16}$
$f_{out}, \frac{f_{out}}{2}, \ldots, \frac{f_{out}}{N}$
$V_{clk_{out}}$ @$\frac{f_{out}}{16}$
$\frac{f_{out}}{2^x}$
$\pm V_{clk_{in}}$ @f_{out}
CML to CMOS

Figure 4.11: Serializer architecture in [7]

coupling enhances the drain-to-source voltage. TC produces larger drain current per unit of width-to-length ratio. Thus large swing fight against the reduced voltage for saving power. Thus a small supply voltage is needed for this circuits operation. It is operating at $0.7V$ with $40Gbps$ of speed.

In [5] and [6] uses PAM4 and NRZ two data format for increasing speed. The PAM4 transmitter uses FFE, and its receiver uses DFE. It consists of an equalizer, three-level digitizer with DFE, a PAM4 decoder and CDR. DFE with 3 taps is used. The output we get at $28Gbps$ at two channel which is summed up at output. NRZ is designed with bandwidth and power optimisation to operate at $56Gbps$.

4.3.2 Asynchronous SerDes

Asynchronous as the name suggest does not require clock for their operations. In an asynchronous model communication is established on request. Thus it has to two main port: Active port and Passive port. Active port initiate communication and passive port wait for the communication to start. Or we can say that passive port make a request to Active port to start communication on a channel, Active port start communication. When the pasice port receives full data, it sends back the $acknowledgement$ [26].

The communication is control by $handshaking$ in asynchronous circuits. It consists of a $request$ signal and a $acknowledge$ signal. The $request$ signal is used to initiate a communication on a channel. After a complete data transfer $acknowledgement$ signal is generated specifying that complete data has reached the destination and now transmission can be stopped [26].

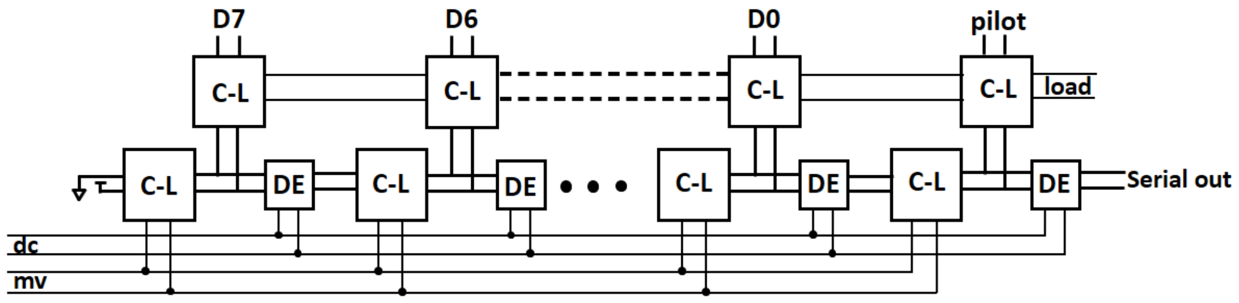

Figure 4.12: Asynchronous serializer block diagram

Figure 4.12 shown a block diagram for the serializer. It uses the chain of latches with a Delay Element in between. Data is fed into each latch using $load$ signals. Propagation of data is controlled by mv signal. A pilot is set at the starting of each word. This pilot bit is used to trigger the control block of deserializer.

Figure 4.13 shown the block diagram of deserializer. The structure of deserializer is same as of serializer so to have the signal symmetrically. The output of this deserializer is given to a control block which generates all the $load$ mv and dc signals used to control the circuit. The detailed explanation will be provided in this chapter later.

In [8] uses a chain of multiplexers and DFFs. The data propagate naturally through the chain. The bit width is maintained by the delay elements provided in between. Therefore delay should be correctly set.

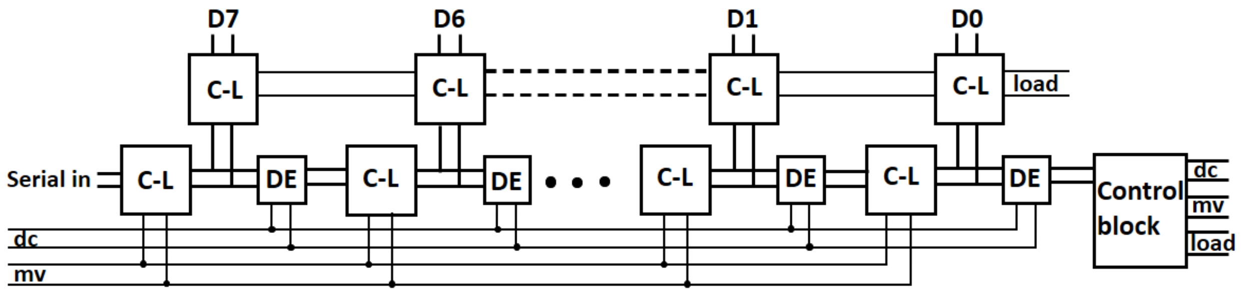

Figure 4.13: Asynchronous deserializer block diagram

A pilot signal is set at the starting of each word for data alignment at the receiver. The circuit operates at $3.9Gbps$ speed with $2.44mW$ power consumption.

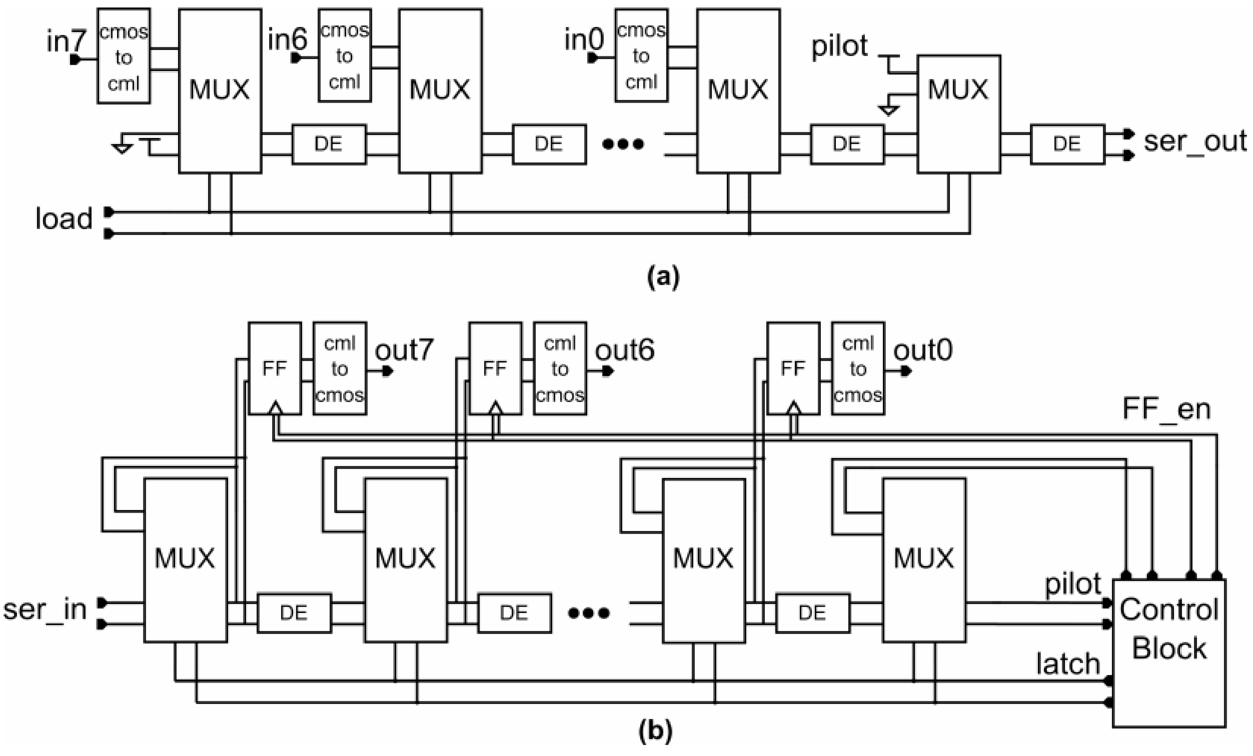

Figure 4.14: Structure of a) Serializer and b) Deserializer

In [27] SerDes is designed for implementation of Address-Event Representation Protocol (AER). It is based on Quasi-Delay Insensitive (QDI) synthesis. If data propagation is null, it consumes zero power because no clock is there to toggle. Serialization is done in $70ns$ with $2.34mW$ power consumption.

Dobkin and et. al. in [10], [28] and [29] uses NRZ Level Encoding Dual Rail (LEDR) asynchronous protocol. It also uses differential channel encoding so to have high error prevention probability. The shift registers are based on novel transition latches. After each word acknowledgement is provided to prevent slow down of the circuit. It achieve a speed of $67Gbps$.

In [9] asynchronous SerDes is implemented using the CML approach thus utilising the benefits of both asynchronous circuit and CML logic. It is 55% faster than conventional CMOS circuit, and CML circuit has less PVT variation as compared to CMOS. The circuit consumes $14.3mW$ power at $12.67Gbps$ speed. The block diagram for serializer is given in figure 4.14 (a) and deserializer in figure 4.14 (b).

Chapter 5

Protocols for SerDes

5.1 Overview

This includes the predefined protocol for SerDes. Place of SerDes in different layer of communication network.

5.2 Protocol Layers

In TCP/IP models the complete communication network from all software and hardware functions necessary for communication. Figure 5.1 shows the block diagram of TCP/IP model. Most serial link follows this model. The application layer is the software of communication network. It processes data in an application specific format and provides it to the lower layer. The transport layer is in control for the end-to-end data transfer by delivering data from an application to its remote peer. This layer also includes error control, fragmentation and flow control. Network layer acts as an interface to actual network hardware. It is also responsible for packet routeing. Data link layer is responsible for formatting data into a defined format for transmission. At last physical layer is the layer which has a connection with the communication channel. SerDes is the part of this physical layer, which helps to transmit data over channel [18].

Protocol defines the following items:

- Data Format
- Sub-channel
- Data Striping

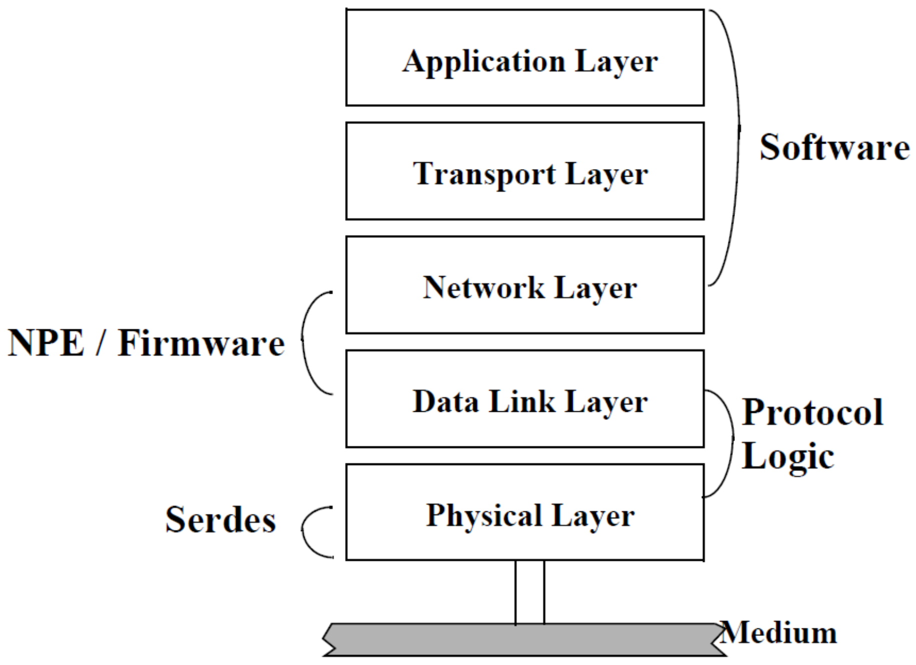

Figure 5.1: TCP/IP model

- Embedding
- Error detection and handling
- Flow Control
- Addressing/switching/forwarding
- Physical interface

Some of the standard protocols used for SerDes are [19]:

- **XAUI**: It is used for 10-Gigabit Ethernet with four channel interface $2.5Gbps$ payload, $3.125Gbps$ wire speed.
- **PCI Express**: Old parallel PCI updated it to a high-speed serial structure.
- **Serial RapidIO**: Another serial version of older parallel spec.
- **Fiber Channel**: Copper channel replaced by fibre optics cable.
- **Infiniband**: Box-to-box protocol uses either copper wire or fibre optics. These are high-speed interconnect for few meters of range.

- **Advanced Switching**: A switched fabric protocol for same as PCI Express.
- **ATCA**: Known as Advanced Telecom Computing Architecture or AdvancedTCA is for next generation telecommunication cabinets. Included are architectures for star-based backplanes, redundant star-based backplanes, and fully connected mesh architectures.
- **Aurora**: It only handle link layer and physical issues. It uses one or more high-speed links.

Chapter 6

Proposed CML Synchronous SerDes

6.1 Overview

In this chapter, the proposed technique for CML Synchronous SerDes is described. Both serializer and deserializer block are explained in detail with their benefits over previous technologies.

6.2 Proposed method

We have preferred CML over CMOS because of the advantages of CML described in chapter 2. It is fully differential and has low power consumption than CMOM at high speed. CML is also more immune to noise as compared with conventional CMOS. We have adopted the same binary tree technique to serialize the data shown in section 4.3.1. Figure 4.4 shows the block diagram for the serializer. In binary tree approach, we have used CMl-MUX for implementation. Deserializer block is similar to that shown in figure 4.6. It is implemented using CML D flip-flop.

6.3 Proposed CML Synchronous Serializer

The proposed serializer is implemented using CML technique. But the general data available is in CMOS logic. Thus first step is to convert this single ended CMOS logic into differential CML logic. The circuit for such conversion is shown in figure 6.1. CMOS to CML conversion is done with the simple differential pair, input and its inverted form are given to differential pair with a current source as soon in figure 6.1.

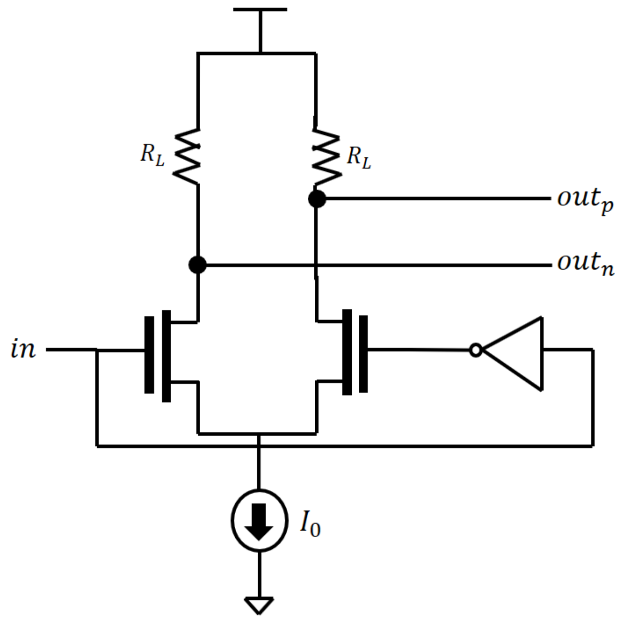

Figure 6.1: Circuit for CMOS to CML conversion

Similarly to interface with other logic the output of CML circuit must be converted back to CMOS logic. This conversion is done with the help of circuit diagram shown in figure 6.2.

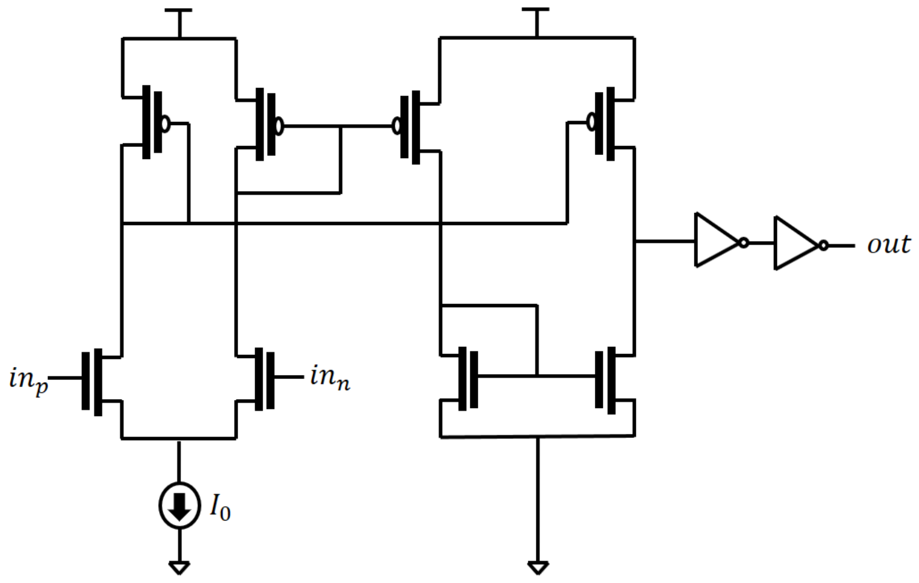

Figure 6.2: Circuit for CML to CMOS conversion

This CMOS to CML conversion is done with the help of a differential pair with an active load connected to a the second differential pair. The voltage of first differential pair is mirrored to the second differential circuit. Output we get from the second differential has low voltage swing thus two inverters are connected to buffer the output to required voltage levels.

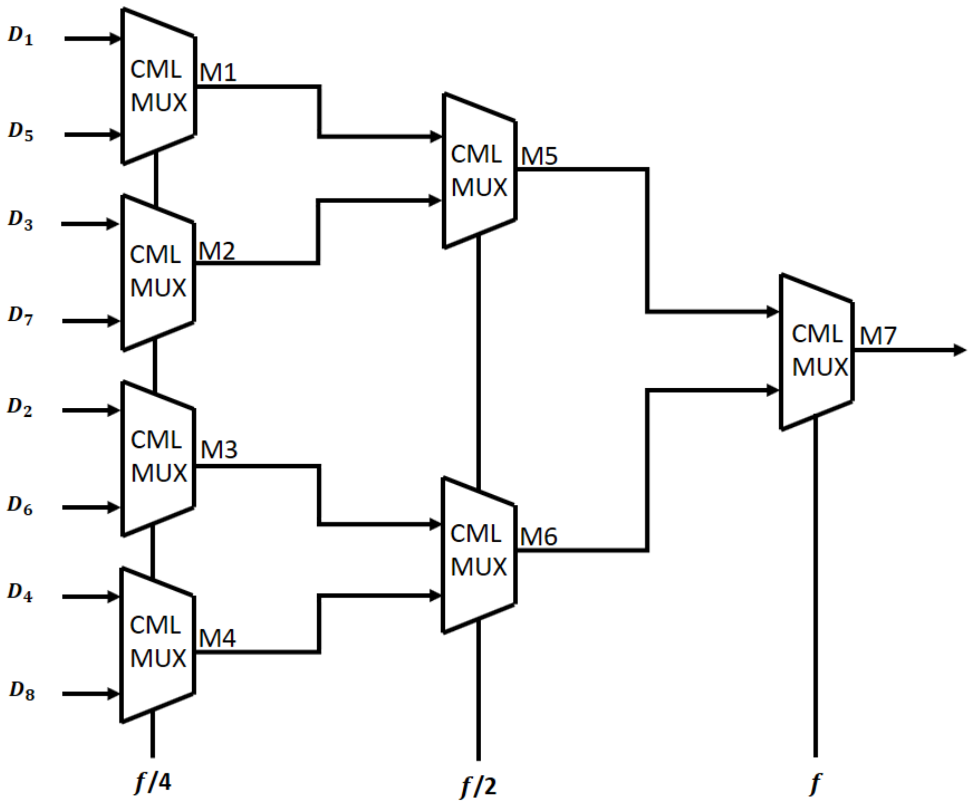

Figure 6.3: Block Diagram for proposed Serializer

The block diagram of proposed serializer is shown in figure 6.3. Its structure is same as shown in the section 4.3.1. Each block is designed using the CML-MUX whose working was explained in the section 2.3.4. Binary tree approach with three layers is adopted. The first layer is provided with the frequency $f/4$, then the second layer with the frequency $f/2$ and so the third layer with frequency f.

The input to first layer is so connected, that after serialization its comes in correct sequence. When $f/4$ goes high MUX1 will select D_1, MUX2 will select D_3; MUX3 will select D_2 and MUX4 will select D_7. Since the first stage has half the clock cycle that of the second stage. During positive half of the first stage, the second stage completes its whole cycle. During positive half of $f/2$, MUX5 will select the output of MUX1 i.e. D_1 and MUX6 will select the output of MUX3 i.e. D_2. Thus we get starting two bits at the input of the last stage. Since the last stage has again twice the frequency that of the second stage, it again toggles to full cycle during positive half of $f/2$. Thus we get D_1 and D_2 in output. The combination of the positive half cycle of first stage with the positive half cycle of the second stage and full cycle of the third stage gives

D_1 and D_2 in the output. Similarly, the positive half cycle of the first stage in combination with the negative half of the second stage and full cycle of the third stage give D_3 and D_4 in output. As discussed above, the negative half of the first stage give next four bits in output and we get all 8 bits serialized during one full cycle of first stage or four full cycles of third stage. The timing diagram for serializer is shown in figure 6.4. After serialization the data is passed through a channel. We have used a resistive channel of characteristic impedance 50Ω along with resistive termination.

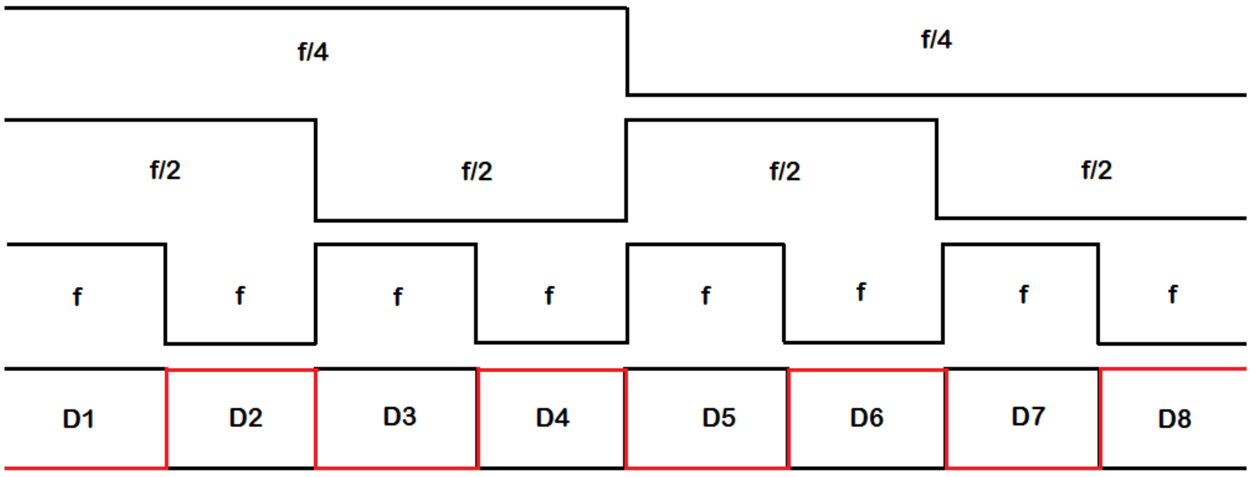

Figure 6.4: Timing Diagram for proposed Serializer

The circuit was designed in Cadence Virtuoso using United Microelectronics Corporation (UMC) 65nm technology. The cadence circuit diagram of serializer is shown in figure 6.5.

6.4 Proposed CML Synchronous Deserializer

Block diagram of proposed serializer is shown in figure 6.6. We have used CML D flip-flop to design deserializer. Deserializer has same structure as explained in section 4.3.1 and block diagram shown in figure 4.6. Two parallel branches of shift registers (D flip-flop) are connected. One branch is driven by positive cycle and another by negative cycle. When serial data is given to two branches, odd bits of data propagate through the upper branch and even bit of data propagate through the lower branch. Each bit shifted by one position at every clock pulse. After four clock cycle, each bit reached their respective registers, and parallel data is taken out from this shift registers. By using this circuit, we require a clock with half the frequency and also the loss in signal with propagating through 8 registers is also reduced.

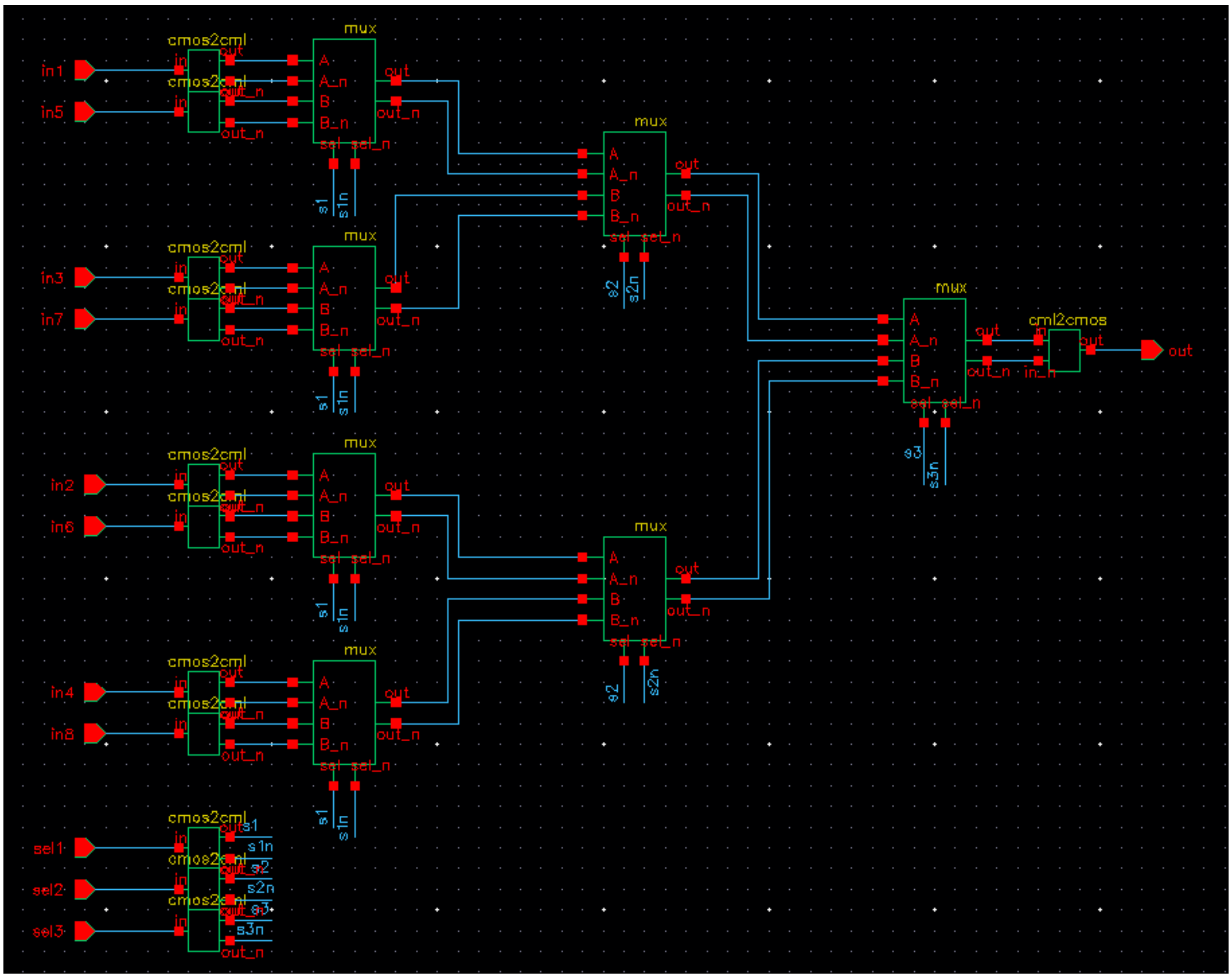

Figure 6.5: Circuit diagram of proposed Serializer in Cadence Virtuoso

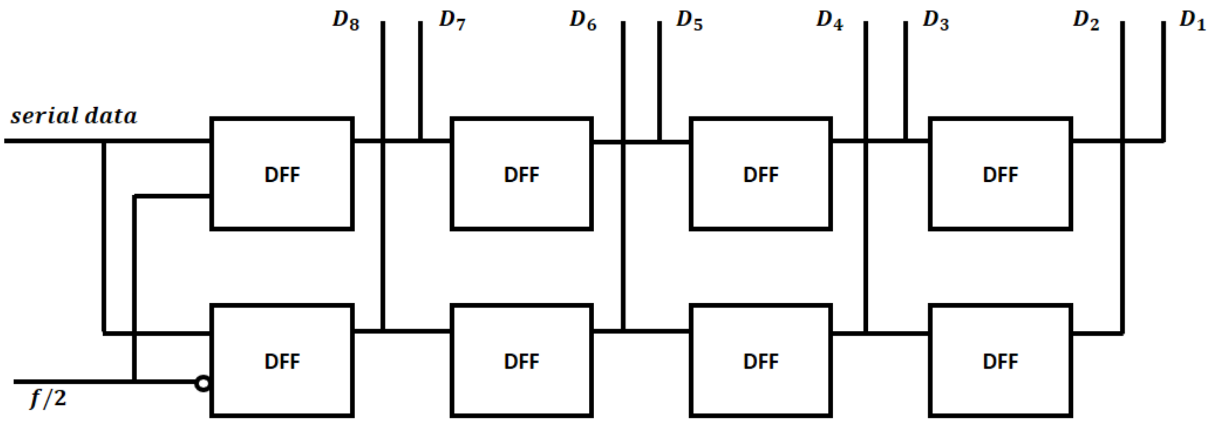

Figure 6.6: Block Diagram for proposed Deserializer

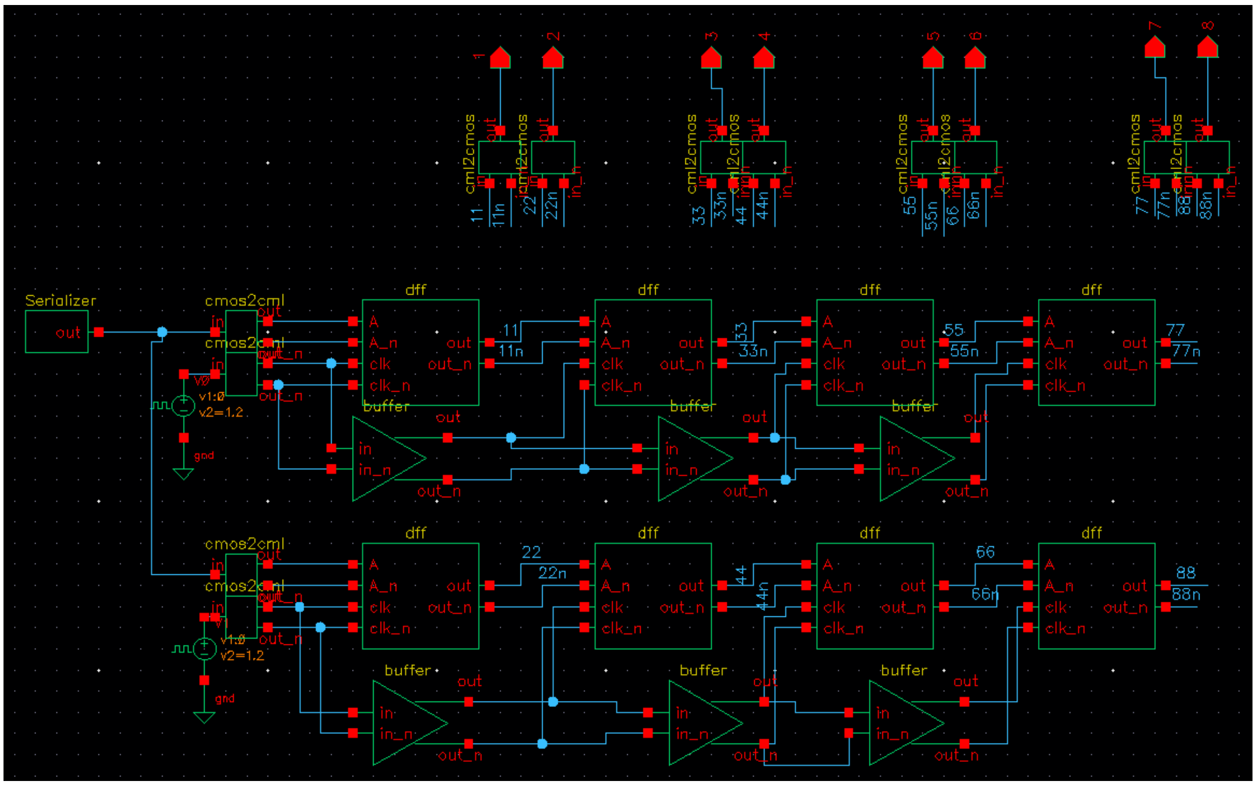

Figure 6.7: Circuit diagram of proposed Deserializer in Cadence Virtuoso

The Cadence circuit diagram is shown in figure 6.7. In this figure, CML buffers is used to propagate clock from one D flip-flop to another to maintain proper voltage level.

Chapter 7

Proposed CML Asynchronous SerDes

7.1 Overview

In this chapter, the proposed asynchronous CML technique is described in detail.

7.2 Proposed Asynchronous Serializer

Block diagram for the proposed asynchronous serializer is shown in figure 7.1. The circuit is implicated using the novel CML-latch explained in the section 2.3.5. Novel CML latch uses an extra nmos in parallel with the constant current source which is driven by negative clock cycle to add extra current during the latching branch. It helps vertical latches to maintain the input value during propagation.

The block diagram has vertical latches connected to D_7, D_6 .... D_0 and at has a $pilot$ bit prefixing all word. This $pilot$ bit can be zero or one. In our circuit, we have taken the pilot bit as one. There are horizontal latches connected to each other with Delay Element (DE). Loading of data through vertical latches

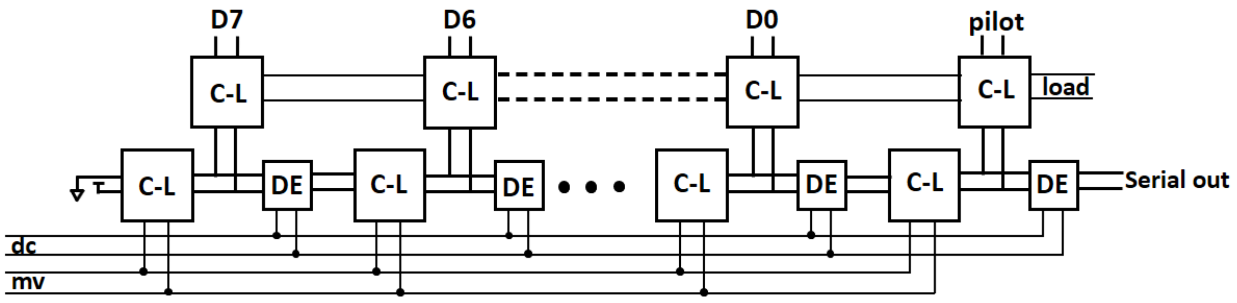

Figure 7.1: Block diagram of proposed Asynchronous serializer

are controlled by $load$ signal and propagation of data through horizontal latch are controlled by mv signal (movement signal). DE is most important part of the asynchronous circuit. Since this circuits does not have clock cycle to control the bit. The bit width is controlled by DE. DE decide the time for which the data remain at the input to CML-latch.

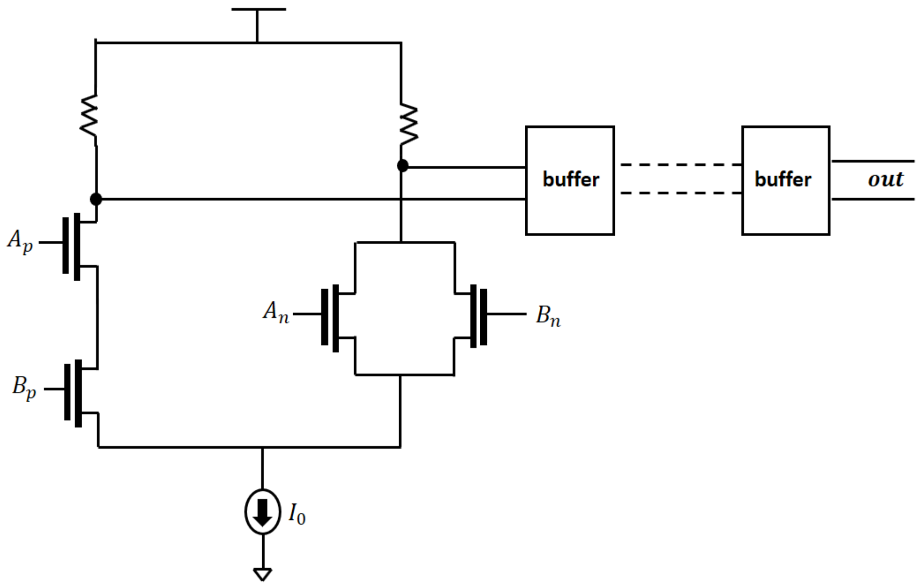

Figure 7.2: Circuit diagram of Delay Element(DE)

The circuit diagram for DE is shown in figure 7.2. We used a CML AND gate followed by CML buffers. DE element is controlled by the dc signal (delay control signal). When the dc signal is high, it passes on all the input data to out. But when the dc signal is low it gives '0' in the output. This will help to wipe out all the data from the serializer and deserializer when complete bits had transferred, and it is ready to take a new bit. Each buffer in the DE has a delay of $4.352ps$ and a total delay of $34.65ps$. So to prevent false triggering of control circuit all the data is wiped out using DE. All the control signal i.e. mv, $load$ and dc signals are generated by control block which will be explained later. In order load the parallel signal into the serializer $load$ signal goes high enabling parallel latch and mv signal does low disabling transmission. When this parallel data is loaded, $load$ signal goes low and mv signal goes high. Thus the vertical CML-latch maintains the parallel bit at the input of each DE. DE maintains the bit width and the parallel data shifts through the latch and get serialized. Last horizontal CML-latch is connected to zero voltage level in order to wipe the suffix of the parallel data. In this way, parallel data is converted into serial bit streams and transmitted over channel. This all serial data streams are prefixed by a pilot bit which is always logic '1'.

The serial data is 9 bit long, 1 *pilot* bit and rest data bit represented as $1XXXXXXXX$.

7.3 Proposed Asynchronous Deserializer

The receiver structure is shown in figure 7.3. As shown here the structure of the deserializer is same as that of the serializer, otherwise data will propagate differently. When the mv signal is high at the serializer, received data propagate through each horizontal CML-latch.

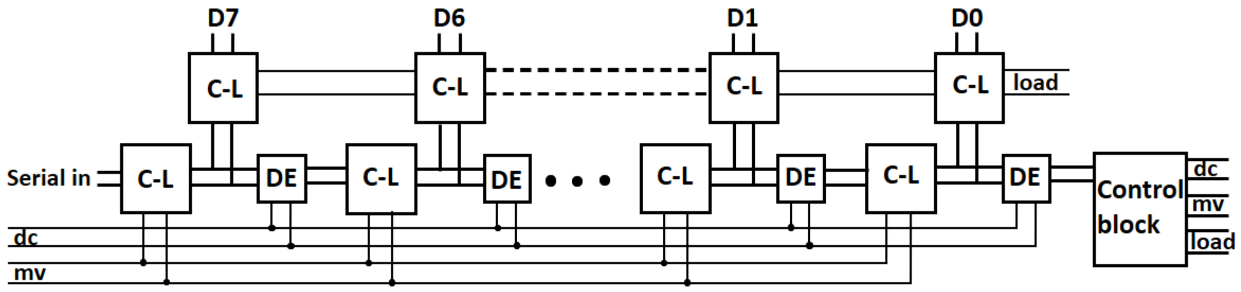

Figure 7.3: Block diagram of proposed Asynchronous deserializer

When the *pilot* bit reaches the Control Block, it signals to control block that all the bits have reached their respective CML-latch and control block generate a signal to ceases the transmission by making mv signal low. After this it makes the *load* signal high to latch the data out of the serializer through vertical latches. Control Block also generate the dc signal which wipes off all the data from input of CML-latch to prevent the false trigging of the control block. Block diagram of control block is shown in figure 7.4.

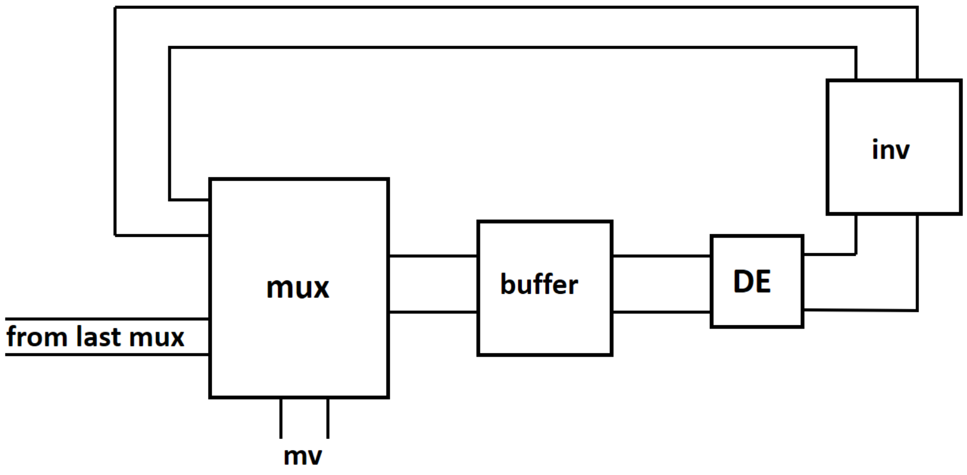

Figure 7.4: Block diagram of Control Block

7.4 Control Block

As soon in figure 7.4, control block consist of a mux, followed by buffer and DE and at last inverted signal is given back to second input of the MUX. When the mv signal is high data propagate in serializer and deserializer and in the control block 2^nd input of MUX waits for the $pilot$ bit. When the $pilot$ bit reaches control block, passing through the mux it goes to buffer. The output of buffer makes the mv to goes low. Thus in serializer and deserializer transmission stops and in control block 1^st input of MUX is selected. The output of buffer after passing through a DE generates the $load$ signal, enabling loading of data at the serializer and latching of data at deserializer.

The output of DE after inverting given as feedback to the 1^st input of MUX. Thus after a delay control block again starts transmission. Loading of data at serializer and latching of data at deserializer is done with the help of same $load$ signal, thus prevents time.

Chapter 8

Results

8.1 Overview

In this section, we will discuss the results of both the proposed technique Synchronous and Asynchronous SerDes with Process Conver Variation and comparison with previous works.

8.2 Result for CML Synchronous SerDes

The proposed method was designed using UMC 65nm technology. For this technology we have used a $1.2V$ power supply. We were able to achieve a data rate of $16.64Gbps$ at $9.29mW$ power consumption. Table 8.1 shows the PVT variation of the circuit. PVT various shown the circuit is quite stable with variation in PVT conditions. It reaches to the highest speed of $16.96Gbps$ for fast-fast and lowest speed of $15.36Gbps$ at slow-slow.

Table 8.1: PVT corners of proposed Synchronous Serdes

P	*V(V)*	*T(^0C)*	*Speed(Gbps)*
SS	1.08	125	15.36
TT	1.2	27	16.64
FF	1.32	-45	16.96

Table 8.2 shown the comparison result of proposed technique with previous techniques.

Table 8.2: Comparison of proposed technique with Various SerDes techniques

Architecture	*Technology(nm)*	*Speed(Gbps)*	*Power(mW)*
Self-timed	65	16	18.1
WP-CML	65	12.67	14.6
WP-CMOS	180	3.9	2.44
CMOS-CML	45/65	10	50/106
This Work	65	16.64	9.29

Figure 8.1 shows the output of the serializer. In the figure, it is clear that the CML output is little distorted but when it is converted back to CMOS, it gives clear result. This proves that CML is high immune to errors. The output we get is 11011000 with a delay.

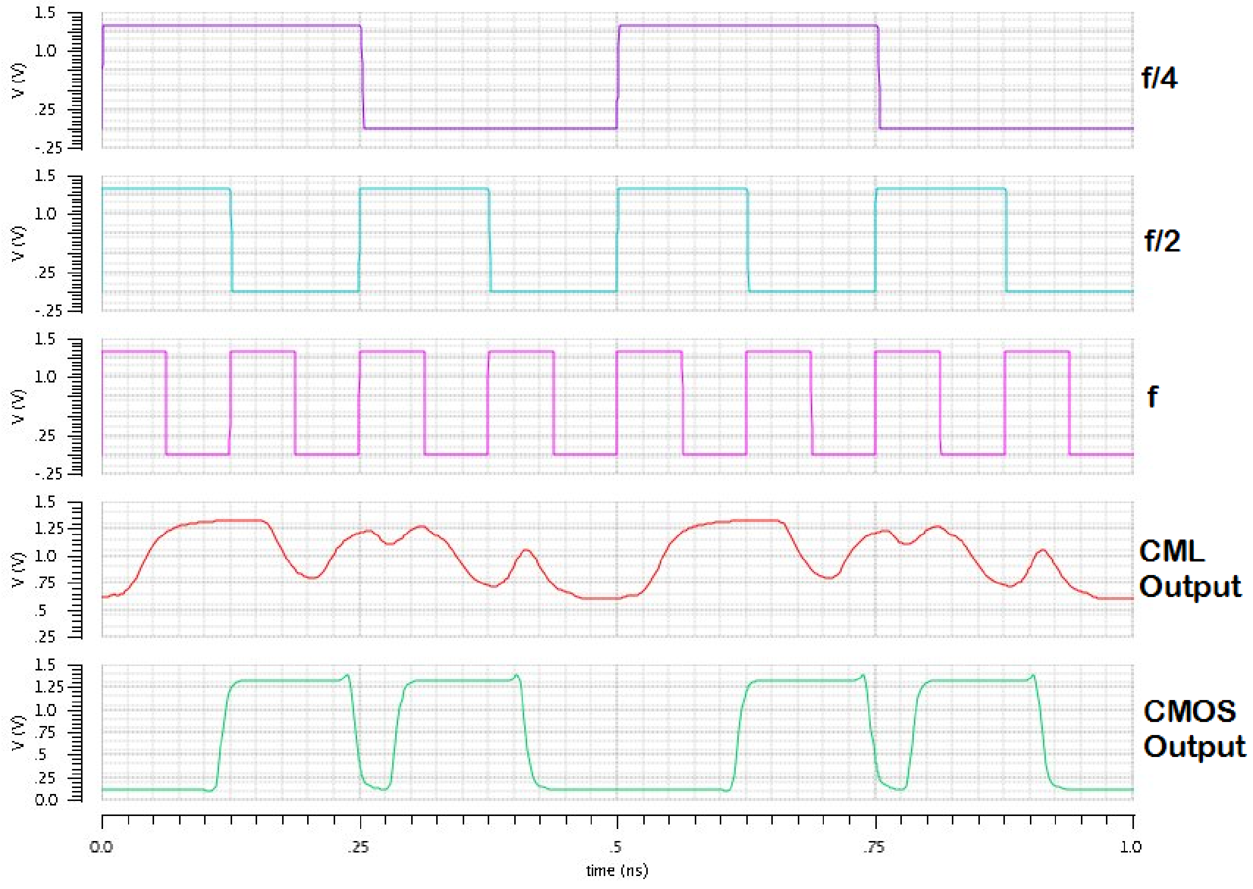

Figure 8.1: Output of Serializer

8.3 Result for CML Asynchronous SerDes

Asynchronous SerDes was also designed using UMC 65nm technology. The power supply was $1.2V$. Using asynchronous method, we were able to achieve a speed of $18.92Gbps$ at $8.82mW$ of power consumption. Table 8.3 shown the PVT varies of the circuit and next table 8.4 shown the comparison of our method with the previous methods.

Table 8.3: PVT corners of proposed Asynchronous Serdes

P	*V(V)*	*T(^0C)*	*Speed(Gbps)*
SS	1.08	125	18.02
TT	1.2	27	18.92
FF	1.32	-45	19.48

Table 8.4: Comparison of proposed asynchronous technique with Various SerDes techniques

Architecture	*Technology(nm)*	*Speed(Gbps)*	*Power(mW)*
Self-timed	65	12	15.5
Self-timed	65	16	18.1
WP-CML	65	12.67	14.6
WP-CMOS	180	3.9	2.44
CMOS-CML	45/65	10	50/106
WP-SR	65	67	175
This Work	65	18.92	8.82

In the next page figure 8.2 shows the output of serializer and figure 8.3 shows the output of deserializer.

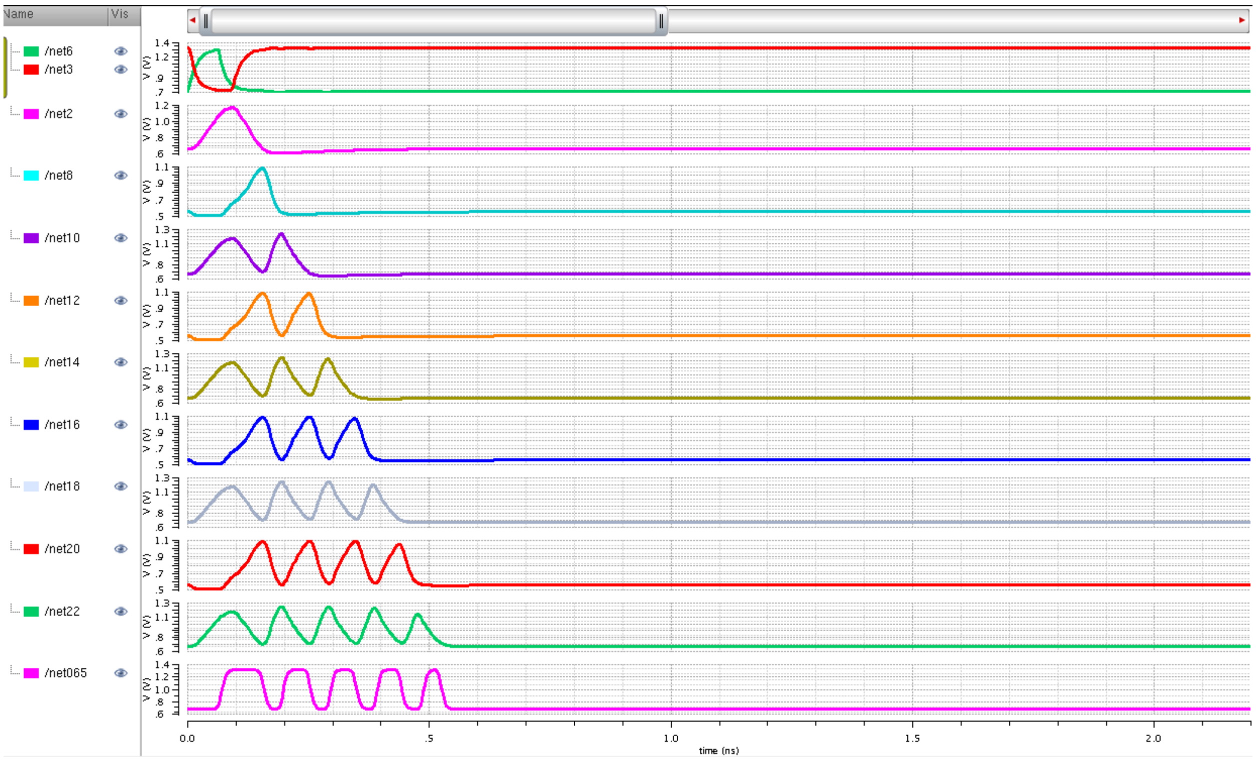

Figure 8.2: Output of Serializer

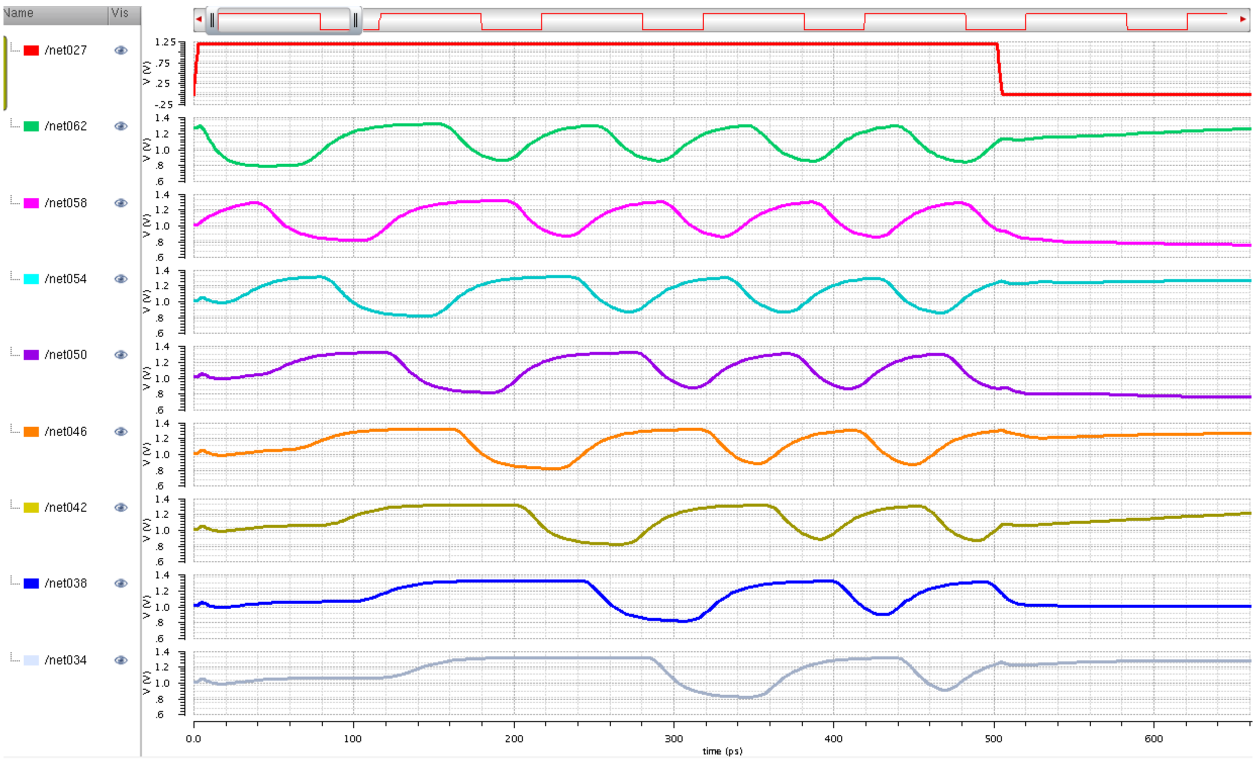

Figure 8.3: Output of Deserializer

Chapter 9

Conclusion & Future Work

9.1 Conclusion

This thesis represented a Synchronous and Asynchronous design of Serial Link transceivers. CML design technique is adopted, since it is fully differential it is more resistive to errors and also CML has low power consumption at high frequencies. All the circuit components are designed by CML logic like MUX, latch, inverter, buffer, etc.

In the first part Synchronous SerDes transceiver is implemented. Problems with the previous design related to power and speed are identified, and improvised technique is implemented and checked with simulation results and achieve $16.64Gbps$ with $9.29mW$. Proposed design has shown improvement in data rate by 32% approximately. The synchronous transceiver has the problem with clock and data recovery which is solved with the CML based synchronous transceiver design.

In order to get better performance in Asynchronous pair SerDes transceiver new pipelined method is proposed, which results in low power consumption of $8.82mW$ with the speed of $18.92Gbps$. Since asynchronous does not have PLL and CDR, it consumes less power. CML circuits design is used which more immune to errors, thus for on-chip communication data can be transferred directly.

Both the methods synchronous and asynchronous has very less variation at PVT corners.

9.2 Future Work

To increase further the performance of our circuit, we can use an inductive load CML logic. Due to an inductive load, it consumes less power and as high switching speed as compared to resistive load CML. For large distance transmission Equalizer can be added to decrease channel distortion. Source encoding and channel encoding methods can be used for error correction and security.

Bibliography

[1] S. Safwat, E. E. D. Hussein, M. Ghoneima, and Y. Ismail, "A 12gbps all digital low power serdes transceiver for on-chip networking," in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, May 2011, pp. 1419–1422.

[2] E. E. D. Hussein, S. Safwat, M. Ghoneima, and Y. Ismail, "A 16gbps low power self-timed serdes transceiver for multi-core communication," in *2012 IEEE International Symposium on Circuits and Systems*, May 2012, pp. 1660–1663.

[3] R. N. Tadros, A. H. Ahmed, M. Ghoneima, and Y. Ismail, "A 24 gbps serdes transceiver for on-chip networks using a new half-data-rate self-timed 3-level signaling scheme," in *Energy Aware Computing Systems Applications (ICEAC), 2015 International Conference on*, March 2015, pp. 1–4.

[4] F.-T. Chen, J.-M. Wu, and M.-C. F. Chang, "40-gb/s 0.7-v 2: 1 mux and 1: 2 demux with transformer-coupled technique for serdes interface," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 62, no. 4, pp. 1042–1051, 2015.

[5] J. Lee, P. C. Chiang, and C. C. Weng, "56gb/s pam4 and nrz serdes transceivers in 40nm cmos," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, June 2015, pp. C118–C119.

[6] J. Lee, P. C. Chiang, P. J. Peng, L. Y. Chen, and C. C. Weng, "Design of 56 gb/s nrz and pam4 serdes transceivers in cmos technologies," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 9, pp. 2061–2073, Sept 2015.

[7] D. F. Tondo and R. R. López, "A low-power, high-speed cmos/cml 16: 1 serializer," in *Micro-Nanoelectronics, Technology and Applications, 2009. EAMTA 2009. Argentine School of.* IEEE, 2009, pp. 81–86.

[8] B. C. Hien, S.-M. Kim, and K. Cho, “Design of a wave-pipelined serializer-deserializer with an asynchronous protocol for high speed interfaces,” in *Quality Electronic Design (ASQED), 2012 4th Asia Symposium on*, July 2012, pp. 265–268.

[9] A. Jaiswal, D. walk, Y. Fang, and K. Hofmann, “Low-power high-speed on-chip asynchronous wave-pipelined cml serdes,” in *2014 27th IEEE International System-on-Chip Conference (SOCC)*, Sept 2014, pp. 5–10.

[10] R. Dobkin, M. Moyal, A. Kolodny, and R. Ginosar, “Asynchronous current mode serial communication,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 7, pp. 1107–1117, July 2010.

[11] M. Yamashina and H. Yamada, “An mos current mode logic (mcml) circuit for low-power sub-ghz processors,” *IEICE Transactions on Electronics*, vol. 75, no. 10, pp. 1181–1187, 1992.

[12] H. Hassan, M. Anis, and M. Elmasry, “Mos current mode circuits: analysis, design, and variability,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 13, no. 8, pp. 885–898, 2005.

[13] J. W. Rogers, C. Plett, F. Dai *et al.*, *Integrated circuit design for high-speed frequency synthesis*. Artech House Boston, London, 2006.

[14] M. Usama and T. A. Kwasniewski, “Design and comparison of cmos current mode logic latches.” in *ISCAS (4)*, 2004, pp. 353–356.

[15] M. Alioto and G. Palumbo, *Model and design of Bipolar and MOS Current-Mode logic: CML, ECL and SCL Digital Circuits*. Springer Science & Business Media, 2006.

[16] X. Zhang, Y. Wang, S. Jia, G. Zhang, and X. Zhang, “A novel cml latch for ultra high speed applications,” in *Electron Devices and Solid-State Circuits (EDSSC), 2014 IEEE International Conference on*, June 2014, pp. 1–2.

[17] M. Usama and T. Kwasniewski, “New cml latch structure for high speed prescaler design,” in *Electrical and Computer Engineering, 2004. Canadian Conference on*, vol. 4, May 2004, pp. 1915–1918 Vol.4.

[18] D. R. Stauffer, J. Trinko-Mechler, M. A. Sorna, K. Dramstad, C. R. Ogilvie, A. Mohammad, and J. D. Rockrohr, *High speed serdes devices and applications*. Springer Science & Business Media, 2008.

[19] A. Athavale and C. Christensen, "High-speed serial i/o made simple," *Xilinx Inc*, vol. 4, 2005.

[20] F. Li, H. Yang, Y. Wang, and Q. Wu, "Current mode feed-forward gain control for 0.8v cmos hearing aid," in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, May 2011, pp. 793–796.

[21] R. A. Philpott, J. S. Humble, R. A. Kertis, K. E. Fritz, B. K. Gilbert, and E. S. Daniel, "A 20gb/s serdes transmitter with adjustable source impedance and 4-tap feed-forward equalization in 65nm bulk cmos," in *2008 IEEE Custom Integrated Circuits Conference*, Sept 2008, pp. 623–626.

[22] Y. Choi and Y. B. Kim, "A 10-gb/s receiver with a continuous-time linear equalizer and 1-tap decision-feedback equalizer," in *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2015, pp. 1–4.

[23] S. Waghela, "Phase locked loop (pll) based clock and data recovery circuits (cdr) using calibrated delay flip flop," 2014.

[24] G. Roosevelt, D. Bueno, J. Haque, W. Roper, and T. Romanko, "Rad-hard high speed serial communication using honeywell serdes macros," in *Aerospace conference, 2009 IEEE*. IEEE, 2009, pp. 1–10.

[25] G. Roosevelt, W. Roper, and T. Romanko, "Optimizing high speed serial communication using honeywell rad hard serdes," in *Adaptive Hardware and Systems (AHS), 2011 NASA/ESA Conference on*. IEEE, 2011, pp. 215–219.

[26] C. J. Myers, *Asynchronous circuit design*. John Wiley & Sons, 2004.

[27] G. Rovere, C. Bartolozzi, N. Imam, and R. Manohar, "Design of a qdi asynchronous aer serializer/deserializer link in 180nm for event-based sensors for robotic applications," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 2712–2715.

[28] R. R. Dobkin, Y. Perelman, T. Liran, R. Ginosar, and A. Kolodny, "High rate wave-pipelined asynchronous on-chip bit-serial data link," in *13th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'07)*, March 2007, pp. 3–14.

[29] R. Dobkin, R. Ginosar, and A. Kolodny, "Fast asynchronous shift register for bit-serial communication," in *12th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'06)*, March 2006, pp. 10 pp.–127.