Instituto Tecnológico
y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática
Maestría en Diseño Electrónico

Design and Optimization of a Multi-Stage Receiver for a PCI Express Gen6.0 Link

TESIS que para obtener el **GRADO** de
MAESTRO EN DISEÑO ELECTRÓNICO

Presenta: **KARLA GABRIELA LOPEZ ARAIZA**

Director **FRANCISCO ELIAS RANGEL PATIÑO**

Tlaquepaque, Jalisco. 13 de Junio del 2025.

Summary

The escalating demand for greater bandwidth, driven by the emergence of new applications has led to the development of the latest Peripheral Component Interconnect Express (PCIe) known as Gen6. This advancement achieves data rates of 64 giga-transfers per second (GT/s) and adopts the Pulse Amplitude Modulation 4-level (PAM4) signaling scheme. While PAM4 effectively addresses the need for increased bandwidth, it introduces new challenges in the design of the physical channel. Specifically, PAM4 is more susceptible to errors arising from various noise sources attributed to reduced voltage and timing ranges, resulting in a higher Bit Error Rate (BER). This phenomenon also brings additional complexities in areas such as slicers, transition jitter, and equalizers [1]; the inherent 1/3 eye amplitudes of PAM4 also result in a signal-to-noise ratio (SNR) penalty and transitions between non-adjacent levels diminish the width of the eye openings. Additionally, many undesired channel effects worsen with higher data rates making of equalization (EQ) a critical process for PAM4 signaling. In this thesis, a multi-stage continuous-time linear equalizer (CTLE) design with high-band, mid-band, and low-band frequency boost stages and a low frequency equalizer (LFEQ) is proposed to meet the requirements of a wide frequency range and channel losses. Due to the intricate nature of equalization for multi-level signals, optimization methods are employed including an efficient optimization of the transmitter finite impulse response (FIR) filter and the receiver CTLE tuning. Equalization settings are optimized to maximize the eye diagram for best link performance. The design verification as well as the optimization method proposal are evaluated through MATLAB SerDes Toolbox.

The research conducted in this thesis resulted in a paper that was published in the IEEE Latin American Electron Devices Conference. The objective of this thesis is to provide a more detailed explanation of the findings presented in the previously published paper titled "A Multi-Stage CTLE Design and Optimization for PCI Express Gen6.0 Link Equalization".

Keywords—channel, CTLE, equalization, eye-diagram, FIR, ISI, jitter, optimization, PAM4, PCIe, receiver, transmitter.

Resumen

La creciente demanda de ancho de banda por parte de nuevas aplicaciones ha llevado al desarrollo de la nueva generación (Gen6) de la interconexión exprés de componentes periféricos (PCIe), alcanzando velocidades de transferencia de datos de hasta 64 gigatransfers por segundo (GT/s) y adoptando el esquema de señalización de modulación por amplitud de pulso de 4 niveles (PAM4). Si bien PAM4 resuelve los requisitos de ancho de banda, plantea nuevos desafíos para el diseño del canal físico. PAM4 es más susceptible a errores debido a diversas fuentes de ruido causadas por la reducción de voltajes y rangos de tiempo, lo que produce una tasa de error de bits (BER) más alta. También introduce nuevos desafíos en los slicers, el jitter por transición y los ecualizadores [1]; las amplitudes intrínsecas de 1/3 del diagrama ojo de PAM4 también conllevan una penalización en la relación señal/ruido (SNR), y las transiciones entre niveles no adyacentes reducen las aperturas horizontales del diagrama de ojo. Además, muchos efectos de canal no deseados empeoran con velocidades de datos más altas, lo que convierte a la ecualización (EQ) en un proceso crítico para la señalización PAM4. En esta tesis, proponemos un diseño de ecualizador lineal en tiempo continuo (CTLE) de múltiples etapas con etapas de frecuencias de banda alta, banda media y banda baja y un ecualizador de baja frecuencia (LFEQ) para cumplir con los requisitos de un amplio rango de frecuencias y pérdidas de canal. Dada la complejidad de la EQ de las señales de varios niveles, se utilizan técnicas de optimización, incluida una optimización eficiente del filtro de respuesta de impulso finito (FIR) del transmisor y la sintonización del CTLE en el receptor. La configuración de ecualización es optimizada para maximizar el diagrama de ojo y lograr el mejor rendimiento del enlace de comunicación. La verificación del diseño, así como la propuesta del método de optimización se evalúan a través de MATLAB SerDes Toolbox.

La investigación realizada en esta tesis dio como resultado un artículo que fue publicado en la Conferencia Latinoamericana de Dispositivos Electrónicos de la IEEE. El propósito de esta tesis es proporcionar una explicación más detallada de los hallazgos presentados en el artículo previamente publicado titulado "A Multi-Stage CTLE Design and Optimization for PCI Express Gen6.0 Link Equalization".

Content

Introduction

The escalating bandwidth requirements of emerging applications have necessitated the implementation of Peripheral Component Interconnect Express (PCIe) Gen6, achieving data rates of 64 giga-transfers per second (GT/s) and adopting the pulse amplitude modulation 4-level (PAM4) signaling scheme. In contrast to conventional non-return-zero (NRZ) signaling scheme, the introduction of PAM4 transceivers poses numerous novel challenges for the analysis and design of the physical channel. The inherent 1/3 eye amplitudes in PAM4 result in a signal-to-noise ratio (SNR) penalty, while transitions between non-adjacent levels with finite rise and fall times diminish the horizontal eye openings. Furthermore, various undesired channel effects exacerbate at higher data rates, including noise and attenuation in the received signal [1].

There is currently a significant industry focus on advancing the development of PAM4 receiver (Rx) architectures featuring high bandwidth, high gain, low noise, and exceptional linearity [2]. Furthermore, equalizers are employed to cancel several undesired effects in the physical channel, such as inter-symbol interference (ISI), thereby increasing the complexity of PAM4 equalization (EQ) [3]. The common approach to address inter-symbol interference (ISI) involves utilizing a combination of continuous-time linear equalizer (CTLE) and decision feedback equalization (DFE). However, with the increased transmission rates, the traditional CTLE is no longer capable of meeting the necessary criteria across a broad spectrum of channel losses [4]-[5].

An intent to apply the PAM-4 signaling to a previous PCIe generation such as PCIe Gen 4.0 is presented in [6]. However, as previously mentioned the new PCIe applications demand a higher data transfer, and then PCIe has evolved to satisfy the requirements of new technologies. PCIe Gen 6.0 quadruples the bit rate of Gen 4.0 from 16GT/s to 64GT/s. Therefore, the challenge for the PAM4 wireline transceiver is to design an optimal Rx stage for a complex generation of PCIe.

A deep analysis of the PAM4 signaling method used in high-speed systems is presented in [7]. The work describes the advantages and disadvantages of using a multi-level signaling, the effect of impedance mismatch over data transmission as well as the impact of crosstalk. Although this is a very reach investigation, the design of a high-speed serial link in a transmission line becomes impractical because it involves hardware.

In [8] some key design techniques of clock and data recovery (CDR) are proposed for PAM4, including effective voltage level transitions selection, sampling location optimization and threshold adjustment. This work is an important research that may have great contribution in the development of the PAM4 wireline transceiver for Gen 6.0. It is imperative the CDR circuit works flawlessly otherwise the data received may not be reliable.

Transmitter (Tx) and receiver (Rx) equalizations continue to play an important role with PAM4 encoding. Even though de-emphasis is a well standardized approach for equalization in NRZ, two enhancement approaches for PAM4 de-emphasis are proposed in [9]. The equalization approaches presented in [9] may contribute for the design of PAM4 Wireline transceiver for PCIe Gen 6.0.

A new optimization technique to design a DFE for binary and multi-level signals is proposed in [10]. This work provides a detailed explanation of the main equalizations techniques and the challenges to overcome using multi-level modulation. The work published in [10] may contribute for the design of PAM4 Wireline transceiver for PCIe Gen 6.0 as well.

The objective of present work is to design an optimal Rx equalizer circuit to be used in a PAM4 Wireline Transceiver for PCIe Gen6. The optimization will be focused to find out the right number of stages that maximize the eye diagram at the receiver meeting the requirements per PCIe 6.0 industrial specification [11]. To maximize the eye diagrams a low-frequency equalizer (LFEQ) will be added. The uncompensated low-frequency loss causes nonnegligible long-term residual ISI that results in 0.42UI data dependent Jitter (DDJ) that is difficult to reduce further by enhancing a CTLE but can be reduced to 0.21UI by adding LFEQ [5].

This thesis introduces a three-stage Continuous-Time Linear Equalizer (CTLE) along with a Low-Frequency Equalizer (LFEQ), designed to address significant channel losses across high, mid and low-frequency bands. Additionally, an effective optimization approach is presented for determining the optimal coefficients for both the transmitter (Tx) feed-forward equalizer (FFE) and the Rx CTLE. The process involves creating a new objective function as a figure of merit (FOM) suitable for PAM4, and subsequently employing a numerical optimization technique through a combination of pattern search [12] and Nelder-Mead [13] methods.

The proposed methodology is validated using the MATLAB SerDes Toolbox with realistic parameters [1]. SerDes Toolbox provides a MATLAB and Simulink model library and a set of analysis tools and apps for the design and verification of serializer/deserializer (SerDes) systems or high-speed memory PHYs such as DDR5. This design toolbox is extremely powerful in terms of computation time as well as the replicating the real-world channel loss.

1. PCI Express

PCIe is a general-purpose high-performance interconnect architecture and one of the most complex high-speed input/output (HSIO) interfaces broadly used across diverse technological areas; and has a wide range of applications ranging from communications, data center, embedded, military and recently adopted by the automotive industry. These are just some of the vast applications this interface can support.

PCIe is the third generation HSIO bus used to interconnect peripheral devices, implements a dual-simplex link transmitting and receiving simultaneously on a lane [8]. A link is the physical connection between two devices and a lane consists in a pair of differential signals in each direction as shown in Fig. 1-1. PCIe has scalable widths meaning that it is possible to have up to 32 lanes in one system. The PCIe speed can also be scalable from 2.5 GT/s to 64 GT/s [11].

The PCIe architecture has incorporated beneficial features from previous generations while leveraging recent advancements. Its predecessors PCI and PCI-X buses were multidrop parallel interconnect buses meaning devices had to share one bus. PCI had a maximum data rate of 266 Mbytes/sec with a 32-bit architecture and PCI-X evolved to a 2131 Mbytes/sec with the last evolution in 2002. Table I-I shows the evolution of PCI.

In order to reduce system costs and continue evolving with newest technology on the market, PCI had to be re-designed. The new interface generation implemented a serial point-to-point type interconnection and the devices were connected via switches which implied a large number of devices connected together in one system. The advantage with this new architecture resulted in a smaller number of pins used to interconnect devices, improving chip and board design costs [14].

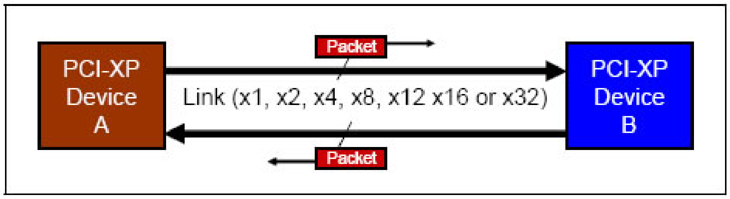


Fig. 1-1 Device A and Device B connected through a link. From [14].

TABLE I-I
PCIE PREDECESSORS. FROM [14].

Bus Type	Clock Frequency	Peak Bandwidth	Number of Card Slots per Bus
PCI 32-bit	33 MHz	133 Mbytes/sec	4-5
PCI 32-bit	66 MHz	266 Mbytes/sec	1-2
PCI-X 32-bit	66 MHz	266 Mbytes/sec	4
PCI-X 32-bit	133 MHz	533 Mbytes/sec	1-2
PCI-X 32-bit	266 MHz effective	1066 Mbyte/sec	1
PCI-X 32-bit	533 MHz effective	2131Mbyte/sec	1

As can be seen PCIe achieves the highest bandwidth per pin, this is due to the serial bus accomplished using a packet-based communication protocol. The packets are called transaction layer packets (TLPs), and there are four types of transactions in PCIe: 1) Memory, 2) IO, 3) configuration and 4) message transactions as shown in Table I-II. Type 1, 2 and 3 are already supported in PCIe predecessors but type 4 is new to PCIe [14].

TABLE I-II
TLP PACKET TYPES. FROM [14].

TLP Packet types	Abbreviated Name
Memory Read Request	MRd
Memory Rea Request-Locked access	MRdLk
Memory Write Request	MWr
IO Read	IORd
IO Write	IOWr
Configuration Read (Type 0 and Type 1)	CfgRd0, CfgRd1
Configuration Write (Type 0 and Type 1)	CfgWr0, CfgWr1
Message request without Data	Msg
Message Request with Data	MsgD
Completion	Cpl
Completion with Data	CplD
Completion without Data- associated with Requests	CplLk
Completion with Data- associated with Requests	CplDLk

1.1. PCIe Transactions

Signals received in high-speed links experience attenuation and dispersion resulting from factors such as transmission line losses, dielectric losses, reflections, and radiation-induced energy dissipation. Consequently, interconnect bandwidth has emerged as a bottleneck in contemporary high-speed systems. To accommodate higher data rates, specialized circuitry and signal processing techniques have proven effective in mitigating the frequency-dependent attenuation encountered in serial data transmission applications. These types of transactions shown in Table I-II are optimized for best performance.

1.2. PCIe Device layers

The PCIe layers consist of a transaction layer, a data link layer and a physical layer. These layers have two portions, a transmit portion that processes departing information and a receive portion which processes incoming information. Fig. 1-2 illustrates the distribution of the layers in a PCIe device [14].

1.2.1 Transmit portion

Packet contents are generated in the transaction layer with information from the device core and application, this packet is stored until is transmitted to the next layers; then the data link layer adds the information required for error detection; and finally, in the physical layer the packet is encoded and is transmitted through the link in a differential way [14].

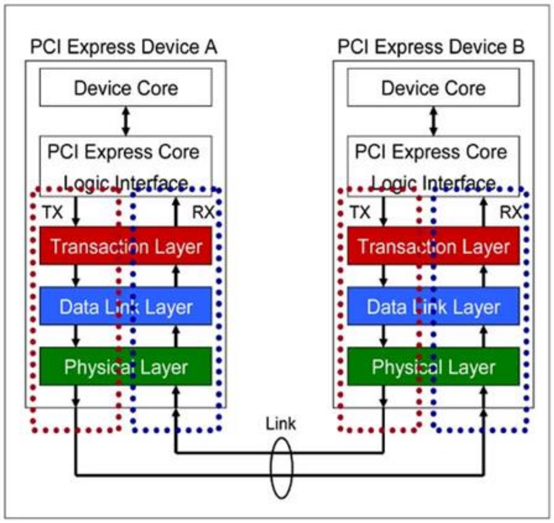


Fig. 1-2 PCIe Device layers and TX/RX portions. From [14].

1.2.2 Receiver portion

The receiver portion decodes the packet in the Physical layer and sends this information to the upper layers, then data link layer checks for any error in the incoming packet. If there are no errors this packet goes directly to the transaction layer where the information is converted to a representation that can be processed by the device core and application [14].

1.3. Packet construction

The software layer or device core sends to the transaction layer the information required to assemble the core of the packet which are the header and the data. As shown in PCIe transaction section some packets do not contain data. Here, it is also added an optional End-to-End cyclic redundancy check (ECRC) used by the completer to check CRC errors in the core information of the packet [14].

The core information is now transmitted to the data link layer which adds a sequence identification (ID) and Link CRC (LCR) field. LCRC fields are used by the completer to check if there is any error in the core section plus the sequence ID, the resultant packet is now forwarded to the physical layer which concatenates a start and end framing character of 1 byte. Then the packet is encoded and differentially transmitted on the link [14].

The received packet is decoded in the physical layer and strips the start and end bytes from the TLP, then the remaining information is passed to the data link layer which checks for any errors in the packet and strips the sequence number and LCRC field. If there is not LCRC errors the packet continues to the transaction layer, if the receiver device is a switch with the header address it can be known the path that needs to follow to arrive to the destination [14].

The targeted device of this TLP checks for ECRC errors and this field is stripped, leaving the header and data section. This information is forwarded to the device core / software layer.

There are other types of packets that can be transmitted in a PCIe link. Data link layer packets (DLLPs) are created in the data link layer of the transmitter and finishes at the data link layer of the receiver device. These types of packets are used for link management functions [14].

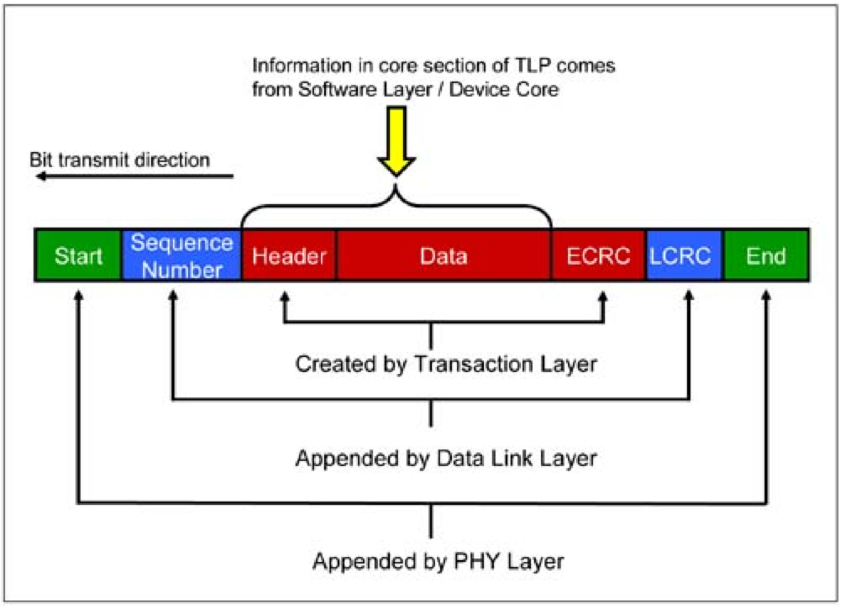

Information in core section of TLP comes
from Software Layer / Device Core
Bit transmit direction
Start
Sequence Number
Header
Data
ECRC
LCRC
End
Created by Transaction Layer
Appended by Data Link Layer
Appended by PHY Layer

Fig. 1-3 TLP Assembly. From [14].

Physical layer packets (PLPs) are packets generated in the physical layer of the transmitter and terminated in the physical layer of the receiver and are used for link training. Fig.1-3 shows the TLP assembly passing through each one of the layers.

Although PCIe Gen5 is already supporting high speed applications and satisfies the current demand in technology with 32 GT/s, every day technology advances with innovative applications in communications, data center, military and even in the automotive industry. It is very accurate to infer that 32 GT/s will not be fast enough in the near future. Therefore, although PCIe generations continue to double the data rate, bandwidth starts to become the bottle neck. Thus, there will be the need of a drastic change in the signaling methodology used to transmit data. Pulse Amplitude Modulation 4 Level (PAM4) is a well-known signaling method that is starting to take more relevance in communications, and it will be introduced in the next chapter.

2. Signaling Methodology

Non-Return to Zero (NRZ) is a signaling methodology used nowadays to transmit data in high-speeds protocols. NRZ also known as Pulse Amplitude Modulation 2 Level (PAM2) consists of 1's and 0's where each amplitude level contains 1 bit of information.

NRZ has gradually evolved over the last 50 years improving speed from 110 bits per second (b/s) to 100 Gb/s [15]. PCIe Gen6 predecessors continue using NRZ as a signaling method, however new advanced applications demand an increase of bandwidth, as the data rate increases beyond 32 GT/s the bandwidth becomes the bottleneck of high-speed interfaces, which are affected by the channel and package losses by using the conventional NRZ signaling method [16].

In order to overcome this problem, PCIe Gen6 adopted PAM4 signaling, which has already been implemented by other networking standards.

2.1. What is PAM4?

In numerous applications, the conventional baseband NRZ signal modulation is being substituted with the more bandwidth-efficient PAM4. PAM4 reduces the required bandwidth for a specific data rate by transmitting two bits in each symbol and does not return to zero after each symbol as NRZ signaling scheme [17]. PAM4 technologies range from 50 Gb/s to 400 Gb/s, achieving the highest data rates by combining multiple lanes of lower-rate signals. The electrical

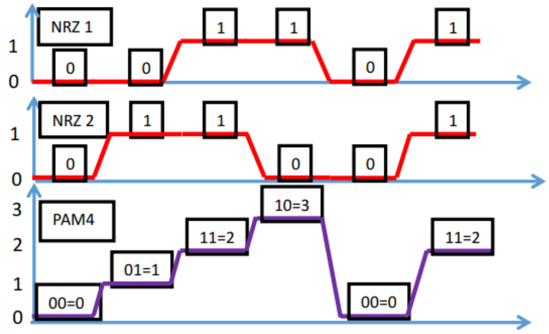


Fig. 2-1 Signaling methodologies. From [7].

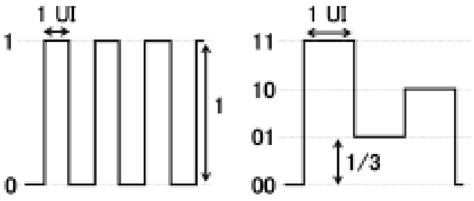


Fig. 2-2 Unit Interval and amplitude difference comparison.

specifications include multi-lane configurations, hot plug capability, low voltage, balanced differential pairs with embedded clocking, and either transmitter, receiver equalization, or both.

2.2. Comparison of PAM4 and NRZ coding

PAM4 is an alternative coding scheme to binary modulation NRZ in high-speed systems especially for signaling speeds beyond 25 Gb/s. PAM4 encodes two bits into one symbol 11, 10, 01 and 00, using gray coding. It is possible to achieve the same data rate using half the bandwidth compared to NRZ, in Fig. 2-1 both signaling methodologies are shown [18].

Table II-I summarizes the principal differences between these two signaling methodologies.

The PAM4 Unit Interval (UI) or symbol length will be larger than NRZ UI, however there are some disadvantages. PAM4 signal has 1/3 the amplitude of NRZ signal (see Fig. 2-2), consequently PAM4 has a worse signal-to-noise ratio (SNR), hence is more susceptible to noise [19]. As a consequence, although UI is larger in PAM4 we will have an additional 33% eye width loss due to jitter (switching between adjacent and non-adjacent levels).

PAM4 offers the benefits of halving the Nyquist frequency and doubling the throughput at the same baud rate as shown in Fig. 2-3.

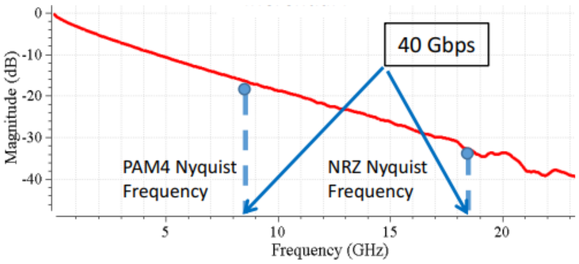


Fig. 2-3 Nyquist frequency difference. From [7].

TABLE II-I
COMPARISSON OF PAM2 AND PAM4 SIGNAL CHARACTERISTICS. FROM [17].

	PAM4	PAM2
Bits per symbol	2	1
Symbol levels	4	2
Rising/Falling edges	6	2
Distinct transitions	12	2
Eye diagrams per UI	3	1
Average transition density	75%	25%

2.3. Gray coding

Instead of having one eye in the NRZ coding scheme, with PAM4 there are three eyes due to the four voltage levels. Two types of mapping methods are available to represent the PAM4 voltage levels, linear coding or gray coding. The difference between these two mapping methods is shown in Table II-II.

As each PAM4 symbol represents two bits, a single symbol error may lead to either one- or two-bit errors. The implementation of gray coding aims to elevate the likelihood that symbol errors induced by amplitude noise will manifest as one-bit errors rather than two, thereby mitigating potential issues. In this way, each symbol represents two bits: 11, 10, 01 and 00 [17].

Gray coding differs only by one bit in each symbol as a result the probability of false reading is reduced minimizing Bit Error Rate (BER) by 25% [20].

There are 3 naming conventions for PAM4 levels generally used in the different literature as shown in Fig. 2-4.

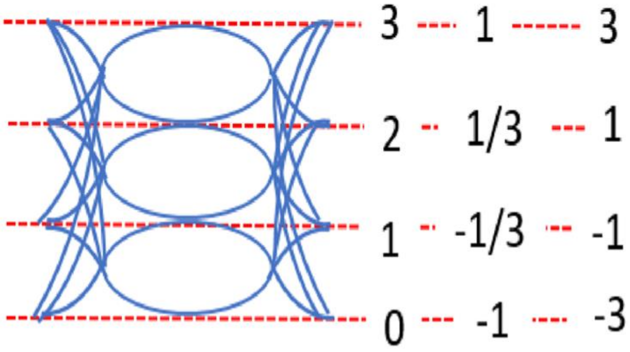


Fig. 2-4 Naming conventions of PAM-4 levels. From [19].

2.4. PAM4 Eye Diagram Characteristics

The PAM4 configuration includes a sole “outer eye” with three inter-dependent “inner eye” diagrams. These inner eye diagrams, labeled as low, middle, and upper, exhibit interdependence since transitions from one symbol to another can impact more than one eye, as shown in Fig. 2-5. Analysis of each of the three eyes follows the same approach used for analyzing PAM2 eyes. In other words, we can independently measure jitter, noise, e_w (eye width), and e_h (eye height) separately for the lower, middle, and upper eye diagrams [17].

When using NRZ, the e_h and e_w are measured from the biggest opening of an eye. However, this is not the case for PAM4’s e_h and e_w [19]. In Fig. 2-6, EH6 represents the e_h and EW6 represents the e_w at a BER of 10^{-6}. e_h is the vertical separation between two points of intersection on 10^{-6} contour ring within an eye diagram. From the Fig. 2-6, observe that the e_w does not represent the widest opening. The imbalance between the upper and lower portions of the eye results in the widest section to be off-center [19].

2.5. PAM4 Design challenges

Newly developed technology is required to accomplish implementation of PAM4, and more complex Tx and Rx circuit designs are required to address the PAM4 challenges. The PAM4 eye height has 1/3 the amplitude of a NRZ symbol making PAM4 more susceptible to noise. Finite rise times affecting various transition amplitudes inherently introduce ISI and complicate clock recovery. The transition time of the PAM4 data signal can lead to notable horizontal eye closure caused by switching jitter, which relies on the signal's rise and fall times.

TABLE II-II
LINEAR AND GRAY CODING. FROM [19].

Linear coding	Gray coding
11	10
10	11
01	01
00	00

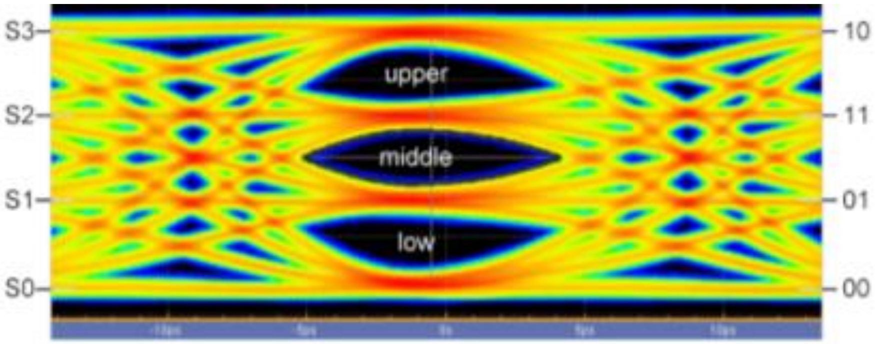


Fig. 2-5 PAM4 Eye diagram. From [17].

Many challenges exist in the PAM4 Clock Data Recovery (CDR) design with ~9dB lower SNR and four voltage levels, especially at the data rate over 50 Gb/s. A variety of level transitions make the sampling clock hard to align with data zero-crossings, which lead to larger jitter [8].

Equalization techniques, including Tx de-emphasis, pre-emphasis, Rx CTLE, and DFE, are extensively employed in high-speed serial links to enhance the opening of the eye diagram [11], and they continue to be used for PAM4. The equalization configuration at the Rx may involve a combination of a CTLE and a DFE. The CTLE is a straightforward circuit with a single coefficient that operates continuously in time, providing high-frequency gain boost. Its transfer function can effectively compensate or equalize the channel response [16]. However, for PCIe Gen6 recent literature states that several CTLE stages may be needed. The decreased eye height in PAM4 imposes a stricter constraint on the sensitivity of the decision circuitry. Additionally, a minimum of three slicers is required to make decisions based on three separate thresholds. [21].

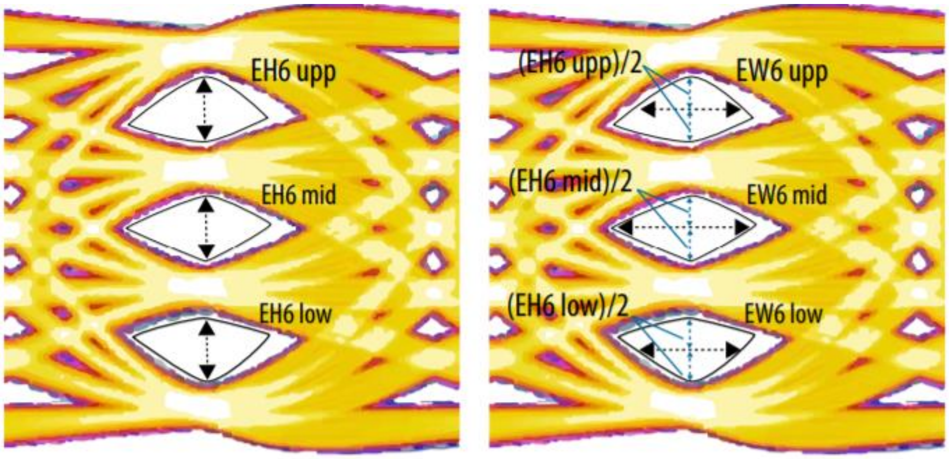


Fig. 2-6 PAM4 Eye diagram features. From [19].

As a result, the power consumption and loading introduced by the slicers are significant considerations in the design of PAM4 receivers. Moreover, integrating a DFE into a PAM4 receiver substantially increases the power and area requirements due to the exponential rise in the number of slicers needed for unrolling the loop [22]. Discovering the most effective tap coefficients has the potential to decrease the required number of DFE taps, thereby lowering the cost of the equalizer.

3. Equalization

3.1. Inter-symbol interference

ISI is a phenomenon in which one symbol interferes with subsequent symbols causing interference between them to the degree that the Rx cannot distinguish between changes of state.

It is not common to have the same impedance through the entire channel, this results in multiply reflected versions of the signal, referred also as “delay spread”. ISI is caused by the nonuniform frequency response of the system which modifies the pulse shape of different bits in a signal, this nonuniform frequency response is due to the geometry of the circuit, the medium from which is composed and the voltage swing of the signal [18].

A communication channel has the typical behavior of a low pass filter, this means that in high frequencies the convolution of the signal transmitted and the channel response results in spread bits causing ISI. Narrow filtering at the Tx and Rx, or channel distortion, can result in the spreading out of the waveform representing a symbol into subsequent symbol periods [23] as shown in Fig. 3-1.

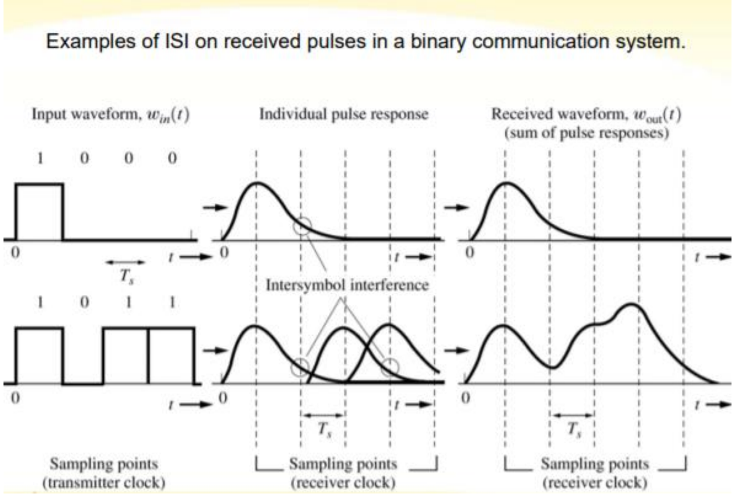


Fig.3-1 Examples of ISI. From [11].

Since the factors that causes the ISI are already determined before data transmission and do not have great fluctuations, the ISI can be corrected. EQ techniques provide the possibility to recover the original signal of the received signal.

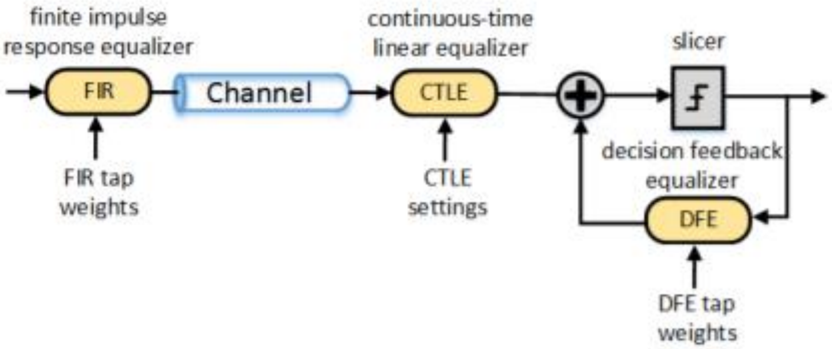

finite impulse response equalizer
FIR
Channel
FIR tap weights
continuous-time linear equalizer
CTLE
CTLE settings
slicer
decision feedback equalizer
DFE
DFE tap weights

Fig. 3-2 Equalization scheme for a PCIe device. From [24].

EQ has many forms but is fundamentally a signal correction scheme. Deconvolution also known as EQ is a compensation that takes place in a signal or waveform that is distorted due to a convolution. In communication systems, this convolution occurs during signal propagation through specific channels, whether wired or wireless [25].

To cancel channel effects due to the constant increase in communication speeds, PCIe specification defines requirements to perform EQ at the Tx and the Rx. A representation of EQ at Tx and Rx is shown in Fig. 3-2.

The main function of EQ is to invert the contribution of the communication channel. A transceiver for PCIe Gen6 implementing PAM4 is expected to be more complex and consume higher power than a transceiver supporting NRZ because of the need for more advanced EQ [11].

3.2. Equalization at the transmitter

There are two types of signal conditioning in EQ also known as emphasis, the first one is the pre-emphasis which consists of amplifying high frequency content of the transmitted signal and the second one is called de-emphasis that consists in decreasing the low frequency content of the transmitted signal.

Tx EQ comes in the form of de-emphasis and preshoot. Fig. 3-3 in section (a) shows the effect of the signal conditioning in EQ for PCIe Gen3-Gen5, section (b) displays signal conditioning for PCIe Gen6.

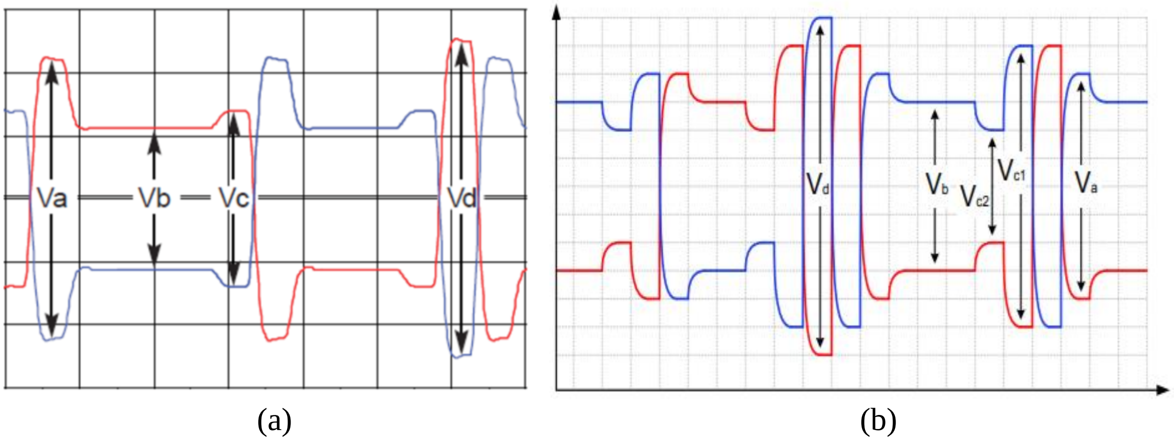


Fig. 3-3 Transmitter equalization for (a) PCIe Gen3-Gen5 from [26]. (b) PCIe Gen6 from [11].

PCIe Gen3-Gen5 signal conditioning functions are identified in (3-1), (3-2) and (3-3).

$$\text{De-emphasis} = -20\, log_{10} \frac{Va}{Vb} \tag{3-1}$$

$$\text{Preshoot} = 20\, log_{10} \frac{Vc}{Vb} \tag{3-2}$$

$$\text{Maximum boost} = 20\, log_{10} \frac{Vd}{Vb} \tag{3-3}$$

PCIe Gen6 signal conditioning functions are listed on (3-4), (3-5), (3-6) and (3-7).

$$\text{De-emphasis} = 20\, log_{10} \frac{Vb}{Va} \tag{3-4}$$

$$\text{Preshoot 1} = 20\, log_{10} \frac{Vc1}{Vb} \tag{3-5}$$

$$\text{Preshoot 2} = 20\, log_{10} \frac{Vc2}{Vb} \tag{3-6}$$

$$\text{Maximum boost} = 20\, log_{10} \frac{Vd}{Vb} \tag{3-7}$$

Where Va refers to the boost after the polarity inversion, Vb is constant voltage when polarity maintains, Vc, $Vc1$ and $Vc2$ refers to the boost before the polarity inversion and Vd is a major boost when there is polarity inversion for one bit interval.

Tx EQ uses 3-tap finite impulse response (FIR) filter also known as feed-forward equalizer (FFE). This type of equalizer compensates for ISI with adjustable tap-coefficients. The three filter taps coefficients are represented by C_{-1}, C_0 and C_{+1} also known as precursor, cursor, and post cursor. C_0 represents a tap coefficient working in the actual signal whereas C_{-1} and C_{+1} represents if the taps work on an advanced or delayed signal with respect to time. These delays are implemented with flipflops controlled by the tap coefficients [18].

The filter response is a multiplication of three consecutive received pulses called V_{n-1}, V_n and V_{n+1} with the correspondent tap coefficient as shown in (3-8). The output voltage then can be altered by tap coefficients as shown in the next equation [11].

$$V_{out} = V_{n-1}C_{-1} + V_nC_0 + V_{n+1}C_{+1} \tag{3-8}$$

Although this EQ was standardized in NRZ (PCIe Gen3-Gen5), it is needed to be enhanced for PAM4 (PCIe Gen6). Tx EQ coefficients for 64 GT/s are based on a FIR filter of 4 tap coefficients, EQ coefficients are subject to constrains limiting their maximum swing to $\pm$unity with C_{-2} being zero or positive, C_{-1} and C_{+1} being zero or negative [11]. The output voltage for PCIe Gen6 is aligned with equation (3-9)

$$V_{out} = V_{n-2}C_{-2} + V_{n-1}C_{-1} + V_nC_0 + V_{n+1}C_{+1} \tag{3-9}$$

3.2.1 Coefficient matrix

Coefficients must comply with the following constraints for PCIe Gen3-Gen5:

$$|C_{-1}| + |C_0| + |C_{+1}| = 1 \tag{3-10}$$

$$C_0 > 0, C_{-1} \leq 0, C_{+1} \leq 0 \tag{3-11}$$

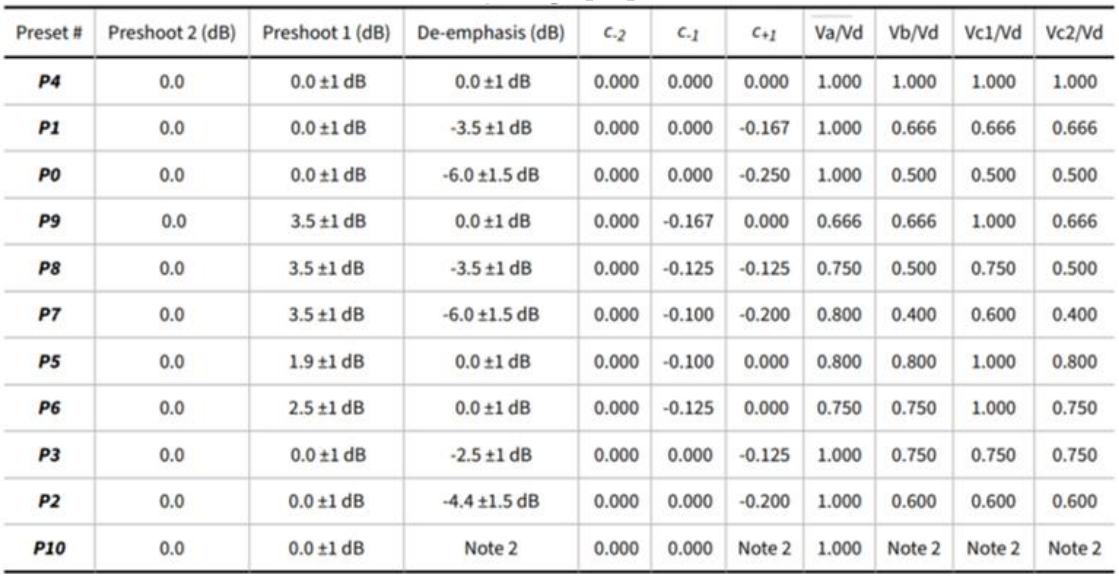
Knowing C_{-1} and $C\ _{+1}$ values, C_0 can be easily estimated. For PCIe Gen6 the constrains have an extra coefficient to consider: In equation (3-12 and 3-13) these constrains are displayed.

$$|C_{-2}| + |C_{-1}| + |C_0| + |C_{+1}| = 1 \tag{3-12}$$

$$C_{-2} \geq 0, C_{-1} \leq 0, C_{+1} \leq 0 \tag{3-13}$$

These coefficients are attached to the PCIe specification released by the Peripheral Component Interconnect Special Interest Group (PCI-SIG) [11].

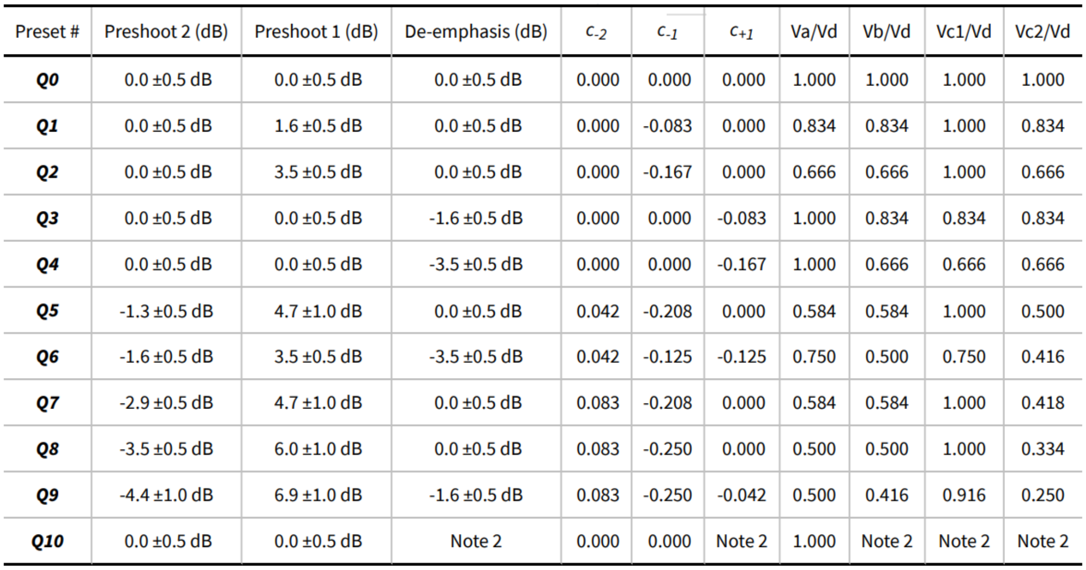
TABLE III-I
PRESETS AND CORRESPONDING COEFFICIENT VALUES FOR PCIE GEN3-GEN5. FROM [11].



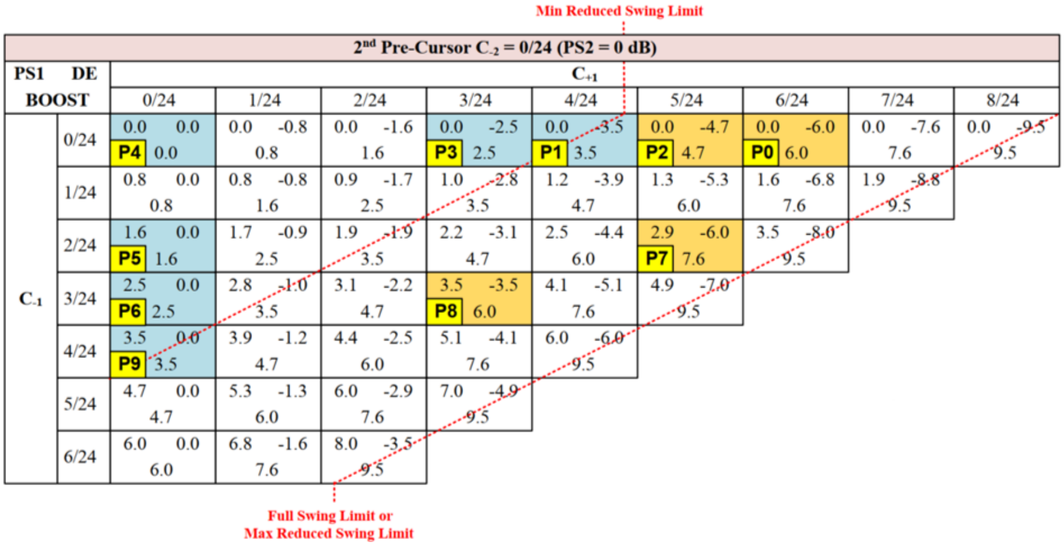
Preset #	Preshoot 2 (dB)	Preshoot 1 (dB)	De-emphasis (dB)	c_{-2}	c_{-1}	c_{+1}	Va/Vd	Vb/Vd	Vc1/Vd	Vc2/Vd
P4	0.0	0.0 ±1 dB	0.0 ±1 dB	0.000	0.000	0.000	1.000	1.000	1.000	1.000
P1	0.0	0.0 ±1 dB	-3.5 ±1 dB	0.000	0.000	-0.167	1.000	0.666	0.666	0.666
P0	0.0	0.0 ±1 dB	-6.0 ±1.5 dB	0.000	0.000	-0.250	1.000	0.500	0.500	0.500
P9	0.0	3.5 ±1 dB	0.0 ±1 dB	0.000	-0.167	0.000	0.666	0.666	1.000	0.666
P8	0.0	3.5 ±1 dB	-3.5 ±1 dB	0.000	-0.125	-0.125	0.750	0.500	0.750	0.500
P7	0.0	3.5 ±1 dB	-6.0 ±1.5 dB	0.000	-0.100	-0.200	0.800	0.400	0.600	0.400
P5	0.0	1.9 ±1 dB	0.0 ±1 dB	0.000	-0.100	0.000	0.800	0.800	1.000	0.800
P6	0.0	2.5 ±1 dB	0.0 ±1 dB	0.000	-0.125	0.000	0.750	0.750	1.000	0.750
P3	0.0	0.0 ±1 dB	-2.5 ±1 dB	0.000	0.000	-0.125	1.000	0.750	0.750	0.750
P2	0.0	0.0 ±1 dB	-4.4 ±1.5 dB	0.000	0.000	-0.200	1.000	0.600	0.600	0.600
P10	0.0	0.0 ±1 dB	Note 2	0.000	0.000	Note 2	1.000	Note 2	Note 2	Note 2

The values used for taps coefficients can be either from its registers and are called "presets" defined by the PCI-SIG specification or could use values sent by the receiver which are calculated by a state machine referred as link training and status machine (LTSSM) also known as "dynamic link equalization". In Table III-I it is observed a set of Presets for PCIe Gen3-Gen5, whereas in Table III-II Presets for PCIe Gen6 are given.

TABLE III-II
PRESETS AND CORRESPONDING COEFFICIENT VALUES FOR PCIE GEN6. FROM [11].



Preset #	Preshoot 2 (dB)	Preshoot 1 (dB)	De-emphasis (dB)	c_{-2}	c_{-1}	c_{+1}	Va/Vd	Vb/Vd	Vc1/Vd	Vc2/Vd
Q0	0.0 ±0.5 dB	0.0 ±0.5 dB	0.0 ±0.5 dB	0.000	0.000	0.000	1.000	1.000	1.000	1.000
Q1	0.0 ±0.5 dB	1.6 ±0.5 dB	0.0 ±0.5 dB	0.000	-0.083	0.000	0.834	0.834	1.000	0.834
Q2	0.0 ±0.5 dB	3.5 ±0.5 dB	0.0 ±0.5 dB	0.000	-0.167	0.000	0.666	0.666	1.000	0.666
Q3	0.0 ±0.5 dB	0.0 ±0.5 dB	-1.6 ±0.5 dB	0.000	0.000	-0.083	1.000	0.834	0.834	0.834
Q4	0.0 ±0.5 dB	0.0 ±0.5 dB	-3.5 ±0.5 dB	0.000	0.000	-0.167	1.000	0.666	0.666	0.666
Q5	-1.3 ±0.5 dB	4.7 ±1.0 dB	0.0 ±0.5 dB	0.042	-0.208	0.000	0.584	0.584	1.000	0.500
Q6	-1.6 ±0.5 dB	3.5 ±0.5 dB	-3.5 ±0.5 dB	0.042	-0.125	-0.125	0.750	0.500	0.750	0.416
Q7	-2.9 ±0.5 dB	4.7 ±1.0 dB	0.0 ±0.5 dB	0.083	-0.208	0.000	0.584	0.584	1.000	0.418
Q8	-3.5 ±0.5 dB	6.0 ±1.0 dB	0.0 ±0.5 dB	0.083	-0.250	0.000	0.500	0.500	1.000	0.334
Q9	-4.4 ±1.0 dB	6.9 ±1.0 dB	-1.6 ±0.5 dB	0.083	-0.250	-0.042	0.500	0.416	0.916	0.250
Q10	0.0 ±0.5 dB	0.0 ±0.5 dB	Note 2	0.000	0.000	Note 2	1.000	Note 2	Note 2	Note 2


Min Reduced Swing Limit
Full Swing Limit or
Max Reduced Swing Limit

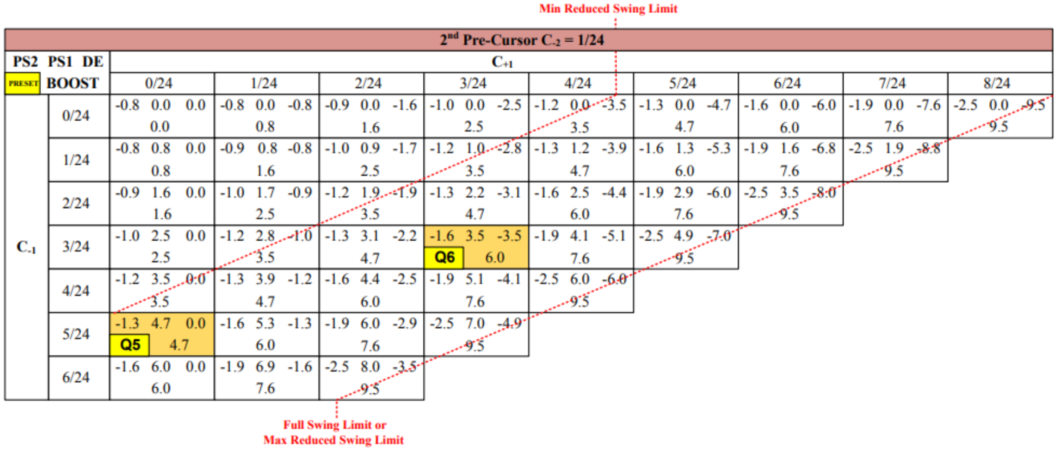
2nd Pre-Cursor C_{-2} = 0/24 (PS2 = 0 dB)										
PS1 DE		C_{+1}								
BOOST		0/24	1/24	2/24	3/24	4/24	5/24	6/24	7/24	8/24
C_{-1}	0/24	0.0 0.0 P4 0.0	0.0 -0.8 0.8	0.0 -1.6 1.6	0.0 -2.5 P3 2.5	0.0 -3.5 P1 3.5	0.0 -4.7 P2 4.7	0.0 -6.0 P0 6.0	0.0 -7.6 7.6	0.0 -9.5 9.5
	1/24	0.8 0.0 0.8	0.8 -0.8 1.6	0.9 -1.7 2.5	1.0 -2.8 3.5	1.2 -3.9 4.7	1.3 -5.3 6.0	1.6 -6.8 7.6	1.9 -8.8 9.5	
	2/24	1.6 0.0 P5 1.6	1.7 -0.9 2.5	1.9 -1.9 3.5	2.2 -3.1 4.7	2.5 -4.4 6.0	2.9 -6.0 P7 7.6	3.5 -8.0 9.5		
	3/24	2.5 0.0 P6 2.5	2.8 -1.0 3.5	3.1 -2.2 4.7	3.5 -3.5 P8 6.0	4.1 -5.1 7.6	4.9 -7.0 9.5			
	4/24	3.5 0.0 P9 3.5	3.9 -1.2 4.7	4.4 -2.5 6.0	5.1 -4.1 7.6	6.0 -6.0 9.5				
	5/24	4.7 0.0 4.7	5.3 -1.3 6.0	6.0 -2.9 7.6	7.0 -4.9 9.5					
	6/24	6.0 0.0 6.0	6.8 -1.6 7.6	8.0 -3.5 9.5						

Fig. 3-4 Transmit triangular coefficient matrix for PCIe Gen3-Gen5. From [11].

There is a triangular coefficient matrix that helps to graphically identify the best combination of coefficients in Fig. 3-4 it is shown an example of a triangular matrix with pre-shoot and calculated values for PCIe Gen3-Gen5 and in Fig. 3-5 PCIe Gen6 coefficient matrix is observed.

3.3. Equalization at the receiver

At higher data rates, various equalization techniques can be employed to counteract ISI impairments, aiming to optimize the eye diagram before the Rx sampling process fails to meet the


Min Reduced Swing Limit
Full Swing Limit or
Max Reduced Swing Limit

2nd Pre-Cursor C_{-2} = 1/24										
PS2 PS1 DE		C_{+1}								
PRESET BOOST		0/24	1/24	2/24	3/24	4/24	5/24	6/24	7/24	8/24
C_{-1}	0/24	-0.8 0.0 0.0 0.0	-0.8 0.0 -0.8 0.8	-0.9 0.0 -1.6 1.6	-1.0 0.0 -2.5 2.5	-1.2 0.0 -3.5 3.5	-1.3 0.0 -4.7 4.7	-1.6 0.0 -6.0 6.0	-1.9 0.0 -7.6 7.6	-2.5 0.0 -9.5 9.5
	1/24	-0.8 0.8 0.0 0.8	-0.9 0.8 -0.8 1.6	-1.0 0.9 -1.7 2.5	-1.2 1.0 -2.8 3.5	-1.3 1.2 -3.9 4.7	-1.6 1.3 -5.3 6.0	-1.9 1.6 -6.8 7.6	-2.5 1.9 -8.8 9.5	
	2/24	-0.9 1.6 0.0 1.6	-1.0 1.7 -0.9 2.5	-1.2 1.9 -1.9 3.5	-1.3 2.2 -3.1 4.7	-1.6 2.5 -4.4 6.0	-1.9 2.9 -6.0 7.6	-2.5 3.5 -8.0 9.5		
	3/24	-1.0 2.5 0.0 2.5	-1.2 2.8 -1.0 3.5	-1.3 3.1 -2.2 4.7	-1.6 3.5 -3.5 Q6 6.0	-1.9 4.1 -5.1 7.6	-2.5 4.9 -7.0 9.5			
	4/24	-1.2 3.5 0.0 3.5	-1.3 3.9 -1.2 4.7	-1.6 4.4 -2.5 6.0	-1.9 5.1 -4.1 7.6	-2.5 6.0 -6.0 9.5				
	5/24	-1.3 4.7 0.0 Q5 4.7	-1.6 5.3 -1.3 6.0	-1.9 6.0 -2.9 7.6	-2.5 7.0 -4.9 9.5					
	6/24	-1.6 6.0 0.0 6.0	-1.9 6.9 -1.6 7.6	-2.5 8.0 -3.5 9.5						

Fig. 3-5 Transmit triangular coefficient matrix for PCIe Gen6. From [11].

specified BER. Tx pre-emphasis encounters constraints related to peak power, while the performance of Rx equalizers is constrained by amplifier bandwidth. Hence, there is a need for strategic design compromises between the implementations of Tx and Rx or a combined approach. Nonetheless, in many instances, achieving perfect channel state information is unattainable and it may fluctuate due to factors such as the PCB manufacturing process, voltage variations, and temperature conditions. To address these challenges, continuous-time adaptive equalizers can be employed [1].

Signals received in high-speed links experience attenuation and dispersion resulting from factors such as transmission line losses, dielectric losses, reflections, and radiation-induced energy dissipation. Consequently, interconnect bandwidth has emerged as a bottleneck in contemporary high-speed systems. To accommodate higher data rates, specialized circuitry and signal processing techniques have proven effective in mitigating the frequency-dependent attenuation encountered in serial data transmission applications [27].

There are two types of equalizers used in the Rx to achieve the mitigation of channel effects, one of them is a discrete time equalizer referred as DFE which is based on samples directly dependent of the clock recovery circuit. The CTLE is not dependent of a clock to work thus this type of equalizer can help to ease ISI.

CTLE is a circuit which transfer function is defined by DC gain, zero frequency and can reach the maximum order adding RC arrays [28]. The CTLE aims to enhance higher frequencies at the receiver, ensuring uniform amplitudes across all frequency components of the signal. This, in turn, enhances jitter and eye-diagram performance [29]. CTLEs can enhance eye openings by compensating for trace losses in PCBs, thereby increasing the testability of the associated data channels.

Equalizer filters can be designed using standard filter design techniques employing either passive or active components. Those incorporating active components offer the advantage of potentially achieving gains surpassing unity and can be seamlessly integrated into silicon designs. One common type of CTLE is the source-coupled differential pair with source degeneration as shown in Fig. 3-6. High-frequency boosting is accomplished by introducing a real zero through the parallel resistor and capacitor network as shown in Fig. 3-6 [30]. As CTLE amplifies jitter,

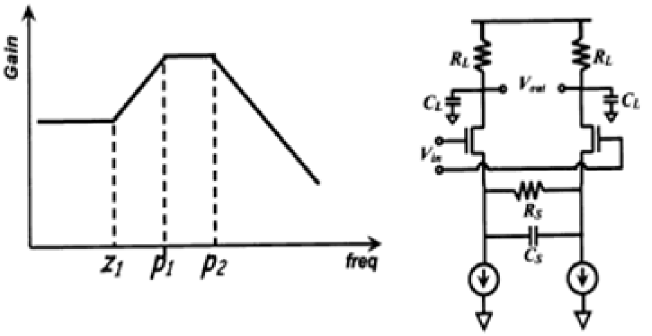


Fig. 3-6 CTLE basic circuit. From [27].

DFE is a nonlinear system specifically designed to avoid noise amplification in Fig. 3-7 a DFE system is shown [31].

The DFE depends on past symbol levels (high/low) to rectify the current symbol. This enables the DFE to address distortion in the current symbol induced by preceding symbols, with ISI directly subtracted from the incoming signal via a feedback FIR filter [32].

3.4. Adaptive equalization

Process, voltage, and temperature (PVT) conditions vary significantly and can cause the channel to change, this makes any EQ performed before inadequate at that time. Techniques such as continuous-time adaptive equalization can adaptively determine equalizer settings required to compensate PVT variations [33].

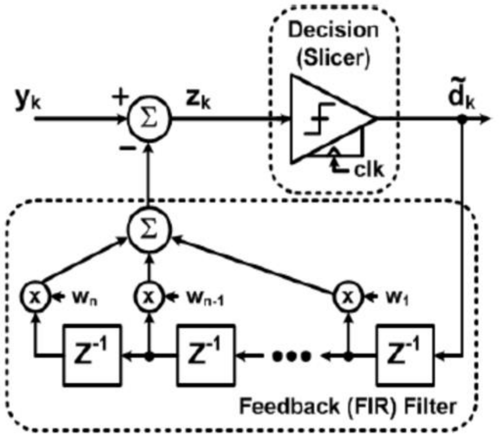


Fig. 3-7 DFE system diagram. From [34].

Equalization

The PCIe specification defines an interactive back-channel equalization (EQ) protocol, which enables link partners to exchange information and allocates a time window for each receiver to adjust the transmitter settings of its link partner. A current approach to implement adaptive link equalization involves conducting transmitter optimization based on receiver eye margin testing. This process systematically evaluates each equalization setting of the Tx while assessing the Rx margin using time margin and/or voltage margin design for test (DFT) functions. Upon exhaustively testing all available Tx EQ settings of the link partner, the optimization procedure selects the Tx EQ configuration that yields the highest margin observed during testing. Typically initiated by the BIOS during system startup, this eye margin test-driven Tx optimization can also be integrated into a PCIe controller or system agent, often referred to as "software equalization." [5].

4. CTLE Design

4.1. Continues Time Linear Equalization

A CTLE is a circuit that enhances high-frequency gain, and its transfer function is designed to flatten the frequency response of the channel. One common type of CTLE is a source-coupled differential-pair circuit with source degeneration, as illustrated in the basic topology shown in Fig. 3-6 [28]. The source resistor of the differential pair attenuates low-frequency signals, and the source capacitor facilitates the transmission of high-frequency signal content, leading to an increase in high-frequency gain [30].

The transfer function of this circuit can be represented by one zero and two poles, where the zero provides +20dB/decade slope and a pole gives −20dB/decade giving a total of −40dB/decade [1]. This topology can be modeled by

$$H(s) = w_{p2}\frac{s + w_{z1}}{(s + w_{p1})(s + w_{p2})} \tag{4-1}$$

where: $w_{z1} = w_{p1}\mathrm{ADC}$; $w_{p1} = 2\pi F_{p1}$; $w_{p2} = 2\pi F_{p2}$, with $w_{z\#}$ representing the location of the zeros, $w_{p\#}$ representing the location of the poles, $F_{p\#}$ the frequencies of the poles, and ADC is the DC gain. By placing $w_{p2} > w_{p1} > w_{z1}$ the CTLE provides high-frequency gain boosting [35] [1].

The PCIe Gen6 specification [11] defines the requirements to support 64 GT/s, defining a CTLE with six poles and three zeros, and an adjustable DC gain, so the system transfer function can be modeled as

$$H(s) = \sigma\frac{(s + w_{z1})(s + w_{p2}ADC)(s + w_{z3})}{(s + w_{p1})(s + w_{p2})(s + w_{p3})(s + w_{p4})(s + w_{p5})(s + w_{p6})} \tag{4-2}$$

where σ is defined by,

$$\sigma = \frac{w_{p1}w_{p3}w_{p4}w_{p5}w_{p6}}{w_{z1}w_{z3}} \tag{4-3}$$

The behavioral loss curves of the required CTLE solution for PCIe Gen6 are shown in Fig. 4-1.

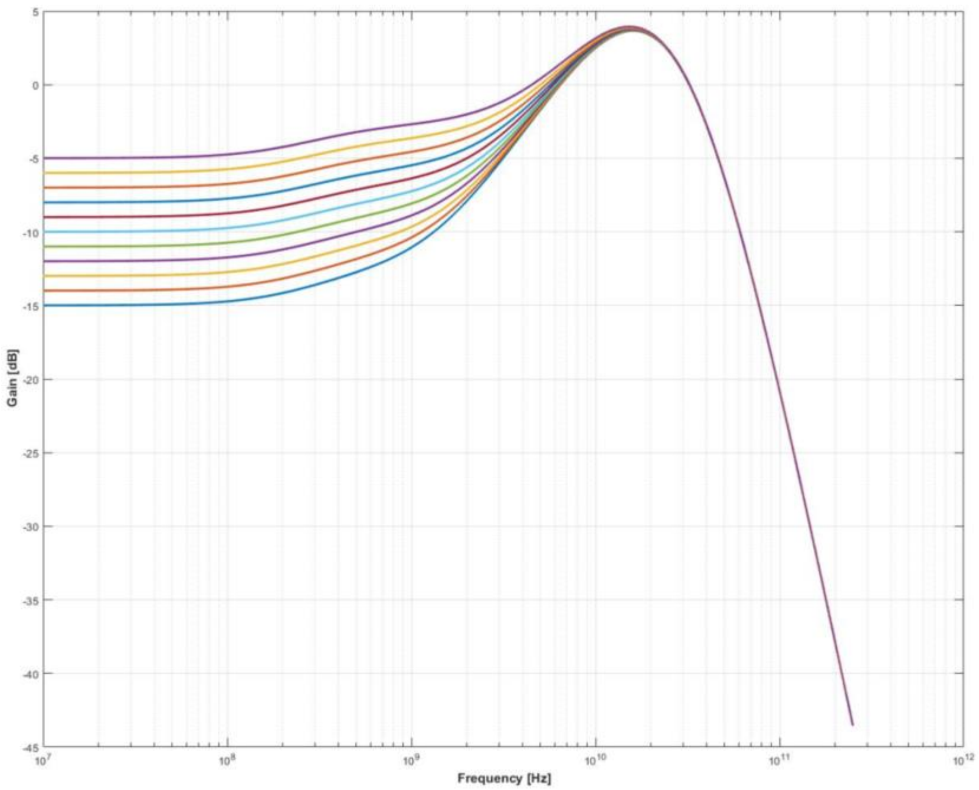


Fig. 4-1 Loss Curves for 64 GT/s behavioral CTLE.

Considering that the CTLE must support a wide frequency range of channel loss, the proposed CTLE consists of three stages to cover the overall transfer function at low-, mid-, and high-frequency ranges, respectively [1]. Henceforth (4-2) can be described as

$$H(s) = \sigma G_1(s)G_2(s)G_3(s) \tag{4-4}$$

where,

$$G_1(s) = \frac{s + w_{z1}}{(s + w_{p1})(s + w_{p6})} \tag{4-5}$$

$$G_2(s) = \frac{s + w_{p2}ADC}{(s + w_{p2})(s + w_{p4})} \tag{4-6}$$

$$G_3(s) = \frac{s + w_{z3}}{(s + w_{p3})(s + w_{p5})} \tag{4-7}$$

Consequently, the EQ topology at the Rx is a combination of a 3-stage CTLE and a DFE, as shown in Fig. 4-2.

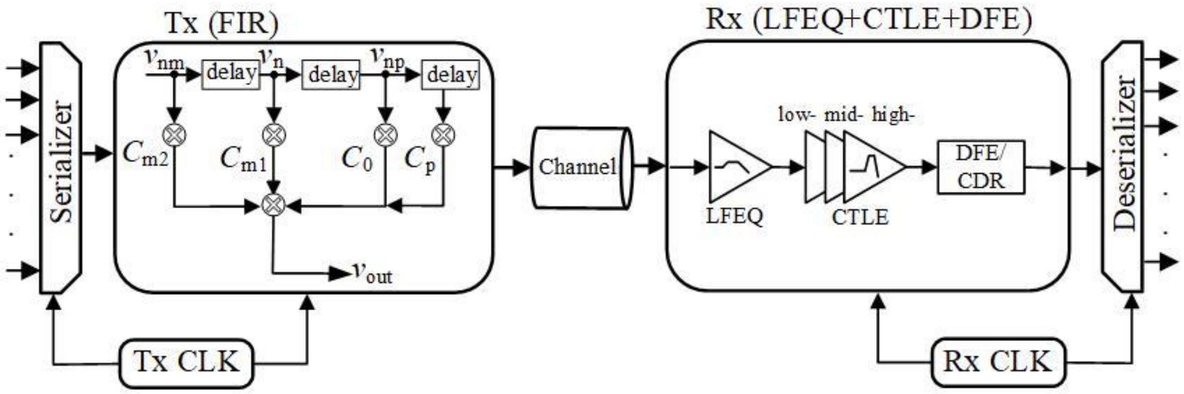


Fig. 4-2 Block diagram of the PCIe Gen6 serial link transceiver. From [1].

4.2. Low Frequency Equalizer

A typical CTLE is not capable of compensating minor low-frequency channel losses as its main purpose is to compensate high-frequency channel losses [16]. Given that the slope of the low-frequency loss is quite small (<3dB/dec), an additional circuit is needed [1].

The unaddressed loss in low-frequency signals leads to significant enduring ISI, contributing to data-dependent jitter (DDJ). Further enhancement of a CTLE alone becomes challenging in reducing this DDJ unless complemented by the incorporation of a low-frequency equalizer (LFEQ) [35]. The LFEQ is based on negative feedback. The objective is to minimize the small slope of low-frequency loss by placing together w_{z1} and w_{p1} pairs to achieve a small amount of low frequency gain (0 to 4dB) [30]. The transfer function of the LFEQ is defined by (4-8), where w_{p1} is tuned to provide the expected DC gain in the low frequency range [1].

$$H(s) = \frac{s + w_{z1}}{(s + w_{p1})(s + w_{p2})} \tag{4-8}$$

where: $w_{z1} = 2\pi F_{z1}$; $w_{p1} = 2\pi F_{p1}$; $w_{p2} = 2\pi F_{p2}$; F_{z1} = 200 MHz; F_{p1} = 320 MHz; F_{p2} = 35 GHz

LFEQ behavioral transfer function implementing the selected poles and zero location for this work is shown in Fig. 4-3. This plot denotes how the gain region extends over an extensive variety of frequencies.

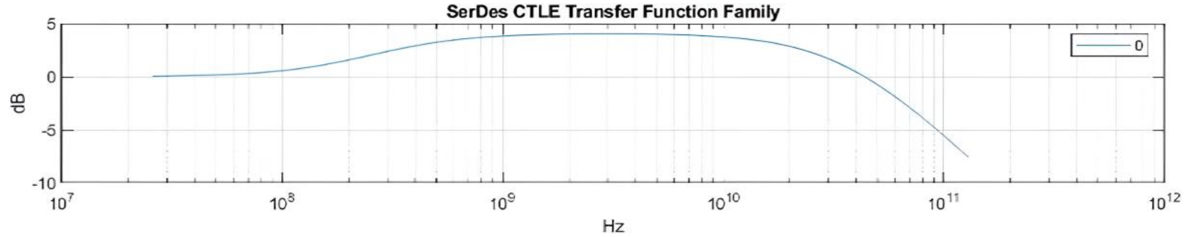


Fig. 4-3 Loss Curve for behavioral LFEQ.

For this thesis advantages of LFEQ and CTLE where combined to cover a wide range of channel and jitter losses and at the same time cover the CTLE curves defined in PCIe Gen6 Spec.

4.3. Implementation

The implementation of the CTLE design started with the evaluation of the three stages defined in chapter 4.1. As it was important to cover a wide frequency range, the first step taken was to ensure low-, mid- and high-frequency bands were supported by testing each stage separately. This evaluation was performed in MATLAB SerDes Toolbox [36].

Low-frequency stage transfer function defined by (4-5) and PCIe base specification [11] is shown in Fig. 4-4.

Where: $w_{z1} = 2\pi F_{z1}$; $w_{p1} = 2\pi F_{p1}$; $w_{p6} = 2\pi F_{p6}$; F_{z1} = 250 MHz; F_{p1} = 325 MHz; F_{p6} = 32 GHz.

Mid-frequency stage is defined by (4-6). Fig. 4-5 displays the adjustable DC gain from -5dB to -15dB to generate the family of curves detailed in PCIe base Specification [11].

Where: $w_{z2} = 2\pi F_{z2}$; $w_{p2} = 2\pi F_{p2}$; $w_{p4}=2\pi F_{p4}$; F_{z2} = mag(DC gain)F_{p2}; F_{p2} = 7.7 GHz; F_{p4} = 28 GHz.

Figure 4-6 shows behavioral transfer function of high-frequency stage described by (4-7), this is the last stage of CTLE design.

Where: $w_{z3} = 2\pi F_{z3}$; $w_{p3} = 2\pi F_{p3}$; $w_{p5} = 2\pi F_{p5}$; F_{z3} = 7.7 GHz; F_{p3} = 22 GHz; F_{p5} = 32 GHz.

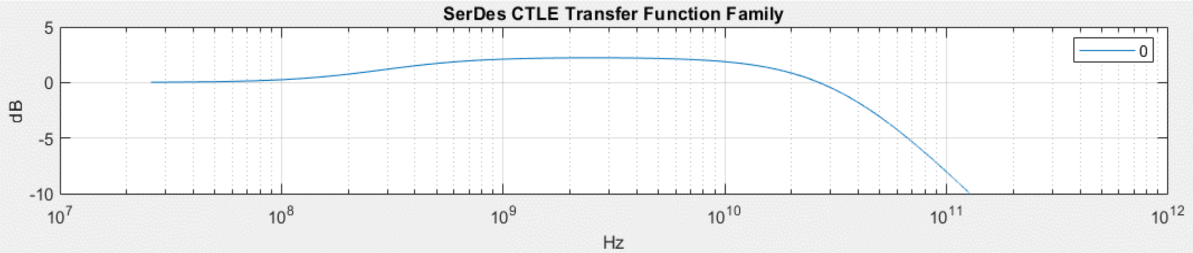


Fig. 4-4 Behavioral transfer function of low-frequency stage.

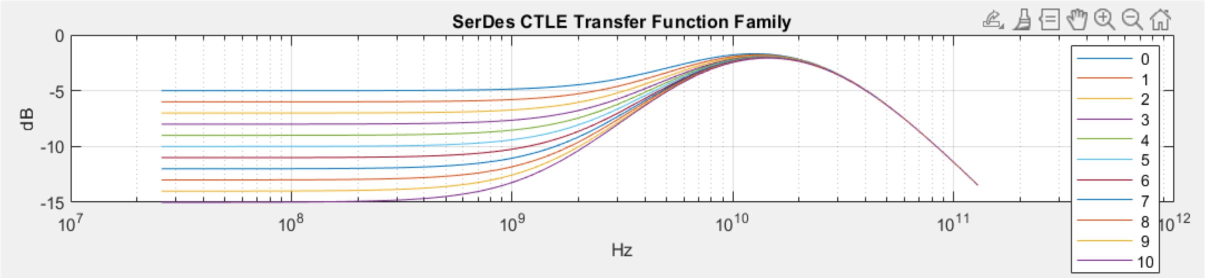


Fig. 4-5 Behavioral transfer function of mid-frequency stage.

With this analysis it can be concluded that each stage of the implemented design is working as expected. While this analysis holds great importance for the investigation, conducting simulation tests using MATLAB SerDes toolbox is crucial to ascertain the efficacy of the implemented solution.

4.3.1 MATLAB SerDes Toolbox

The MATLAB SerDes ToolBox is an add-on package from MathWorks that specializes in supporting design, analysis, and verification of serializer and deserializer (SerDes) systems and high memory physical layers (PHYs), this powerful tool encompasses various modulation schemes, such as NRZ, PAM3, PAM4, PAM8, and PAM16 [36].

Utilizing MATLAB SerDes Toolbox based building blocks like the CTLE, DFE, FFE, and CDR, the chosen architecture can be described using behavioral transfer functions. Simulations can be conducted employing statistical analysis. The application found within the SerDes Toolbox provides parameterized models and algorithms that facilitate an investigation and experimentation of various equalizer configurations, as depicted in Fig. 4-7.

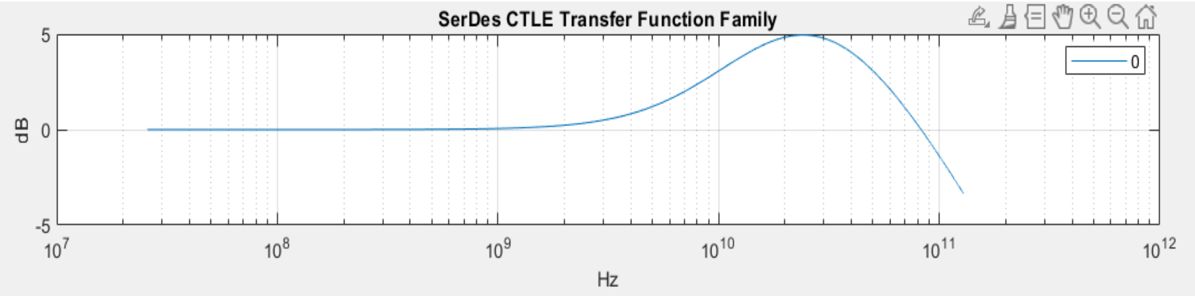


Fig. 4-6 Behavioral transfer function of high-frequency stage

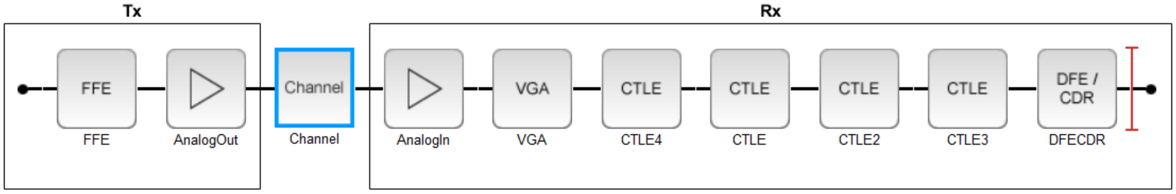


Fig. 4-7 PCIe SerDes design implementation. LFEQ is represented by CTLE4 block and complete CTLE design is exemplified with the following blocks. Low-frequency stage is embedded in CTLE block; Mid-frequency stage is represented by CTLE2 block; High-frequency stage symbolized with the CTLE3 block.

Through the utilization of this tool, important metrics such as the eye diagram, eye linearity, vertical eye closure, bathtub curve, and channel operating margin (COM) can be evaluated using the reporting system as can be seen on Table IIII-I. Furthermore, the assessment process can incorporate considerations for the effects of jitter and crosstalk.

The metrics utilized for evaluation and optimization are vertical eye closure (*VEC*), eye linearity ($e_{\text{linearity}}$), eye height *(*e_{h}) and eye width (e_{w}); *VEC* is a measure of the ratio of the ideal eye opening (separation between the average levels surrounding the eye) to the measured eye height reported in dB. (4-9) demonstrates the variables needed to calculate *VEC*, where the AVupp, AVmid and AVlow are the averages of the eye amplitudes.

$$VEC = 20\,log(\,min(\frac{AVlow}{e_h low}, \frac{AVmid}{e_h mid}, \frac{AVupp}{e_h upp})) \tag{4-9}$$

In the other hand $e_{\text{linearity}}$ is the ratio of minimum eye amplitudes to maximum eye amplitudes (in Volts). Equation (4-10) denotes the eye linearity function and Fig. 4-9 shows a PAM4 comparison between a good linearity and a bad linearity case.

TABLE IIII-I
PAM4 Signaling Metrics. From [36].

Performance Metric	Description
Eye Height (V)	Eye height at the center of the BER contour
Eye Width (ps)	Eye width of the BER contour
Eye Linearity	Measure of the variance of amplitude separation among different levels of PAM4
COM	Channel operating margin
VEC	Vertical eye closure

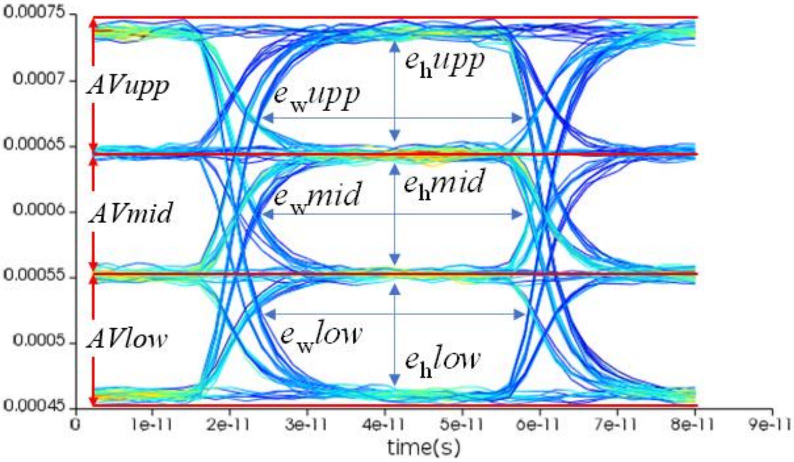


Fig. 4-8 Definition of eye center and nominal slice thresholds. From [37].

$$e_{linearity} = \frac{min(\,AVlow, AVmid, AVupp)}{max(\,AVlow, AVmid, AVupp)} \tag{4-10}$$

In Fig. 4-8 a visual representation of the variables needed to calculate the *VEC* and $e_{\text{linearity}}$ are shown.

In this implementation each block was modified to simulate a PCIe Gen6 model. The Rx section is constructed by various CTLE blocks; CTLE4 block holds the transfer function of designed LFEQ and subsequent CTLE's contains from low-frequency stage to high-frequency stage transfer functions.

4.4. Design results

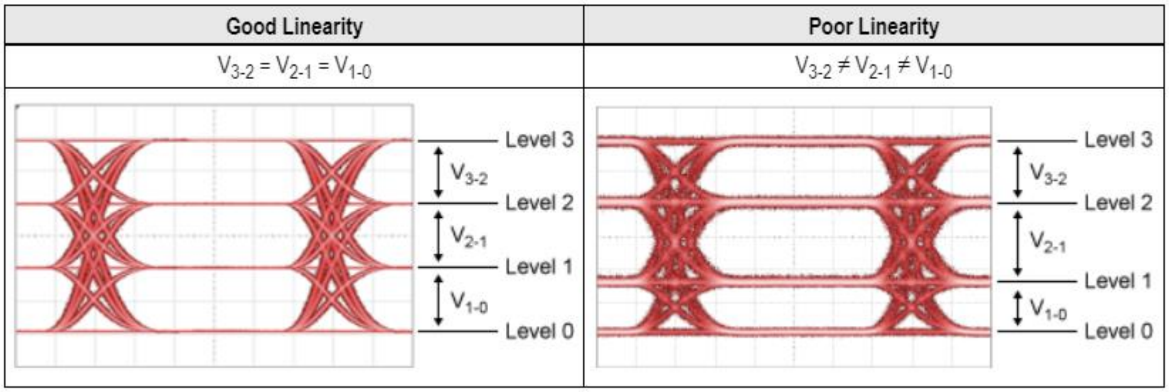


Fig. 4-9 Good eye linearity VS. poor eye linearity. From [38].

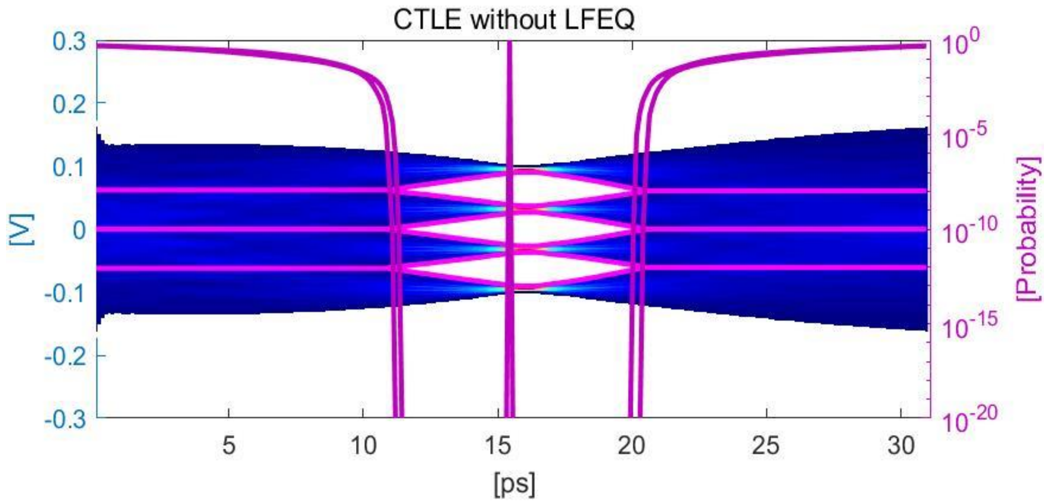


Fig. 4-10 CTLE performance without LFEQ. $e_h upp$ = 44mV, $e_w upp$ = 9.1ps, $e_h mid$ = 44mV, $e_w mid$ = 9.8ps, $e_h low$ = 44mV, $e_w low$ = 9.1ps

To verify our approach, the MATLAB SerDes Toolbox is utilized, evaluating short, medium, and long-reach channels (CEI-56G serial links) with losses of 10dB, 20dB, and 27dB respectively, within a 64 GT/s PCIe Gen6 link. The criteria for success or failure are determined based on a time-domain eye diagram at $\mathrm{BER} = 10^{-6}$. The link is simulated with the corresponding Tx jitter parameters (deterministic and sinusoidal) based on [11], and Rx jitter parameters from a common reference clock Rx architecture. The simulator generates an output containing the three statistical eye heights and widths [1].

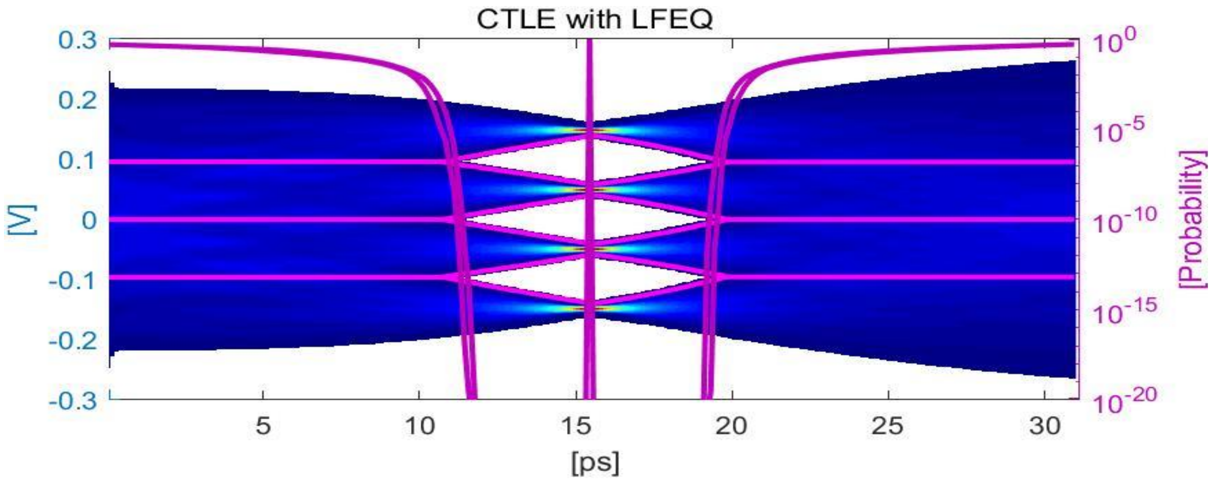


Fig. 4-11 CTLE performance without LFEQ. $e_h upp$ = 63mV, $e_w upp$ = 8.6ps, $e_h mid$ = 62mV, $e_w mid$ = 9ps, $e_h low$ = 63mV, $e_w low$ = 8.6ps

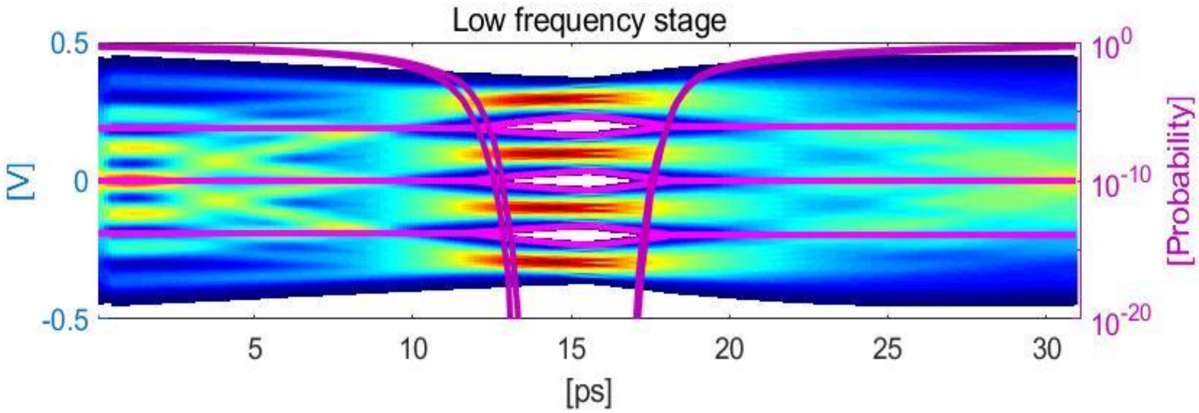


Fig. 4-12 Eye diagram at low frequency stage of the CTLE. $e_{\mathrm{h}}upp$ = 49mV, $e_{\mathrm{w}}upp$ = 5.9ps, $e_{\mathrm{h}}mid$ = 49mV, $e_{\mathrm{w}}mid$ = 6.3ps, $e_{\mathrm{h}}low$ = 49mV, $e_{\mathrm{w}}low$ = 5.9ps

The simulation outcomes for a medium-reach channel in Fig. 4-10 illustrate the performance without LFEQ compensation. However, Fig. 4-11 demonstrates how the combined use of LFEQ and CTLE enhances the overall performance in the Rx equalization scheme, resulting in a 35.3% improvement in eye area.

As explained before Nyquist frequency is continuously increasing with each PCIe generation, so it is needed more than one CTLE to cover EQ of frequencies in the lower to middle range. The low-frequency CTLE shown in Fig. 4-12 is designed to match better the gentle slope of the channel’s frequency response in the low frequency region [39].

Mid-Frequency stage in Fig. 4-13 is also referred as the long-tail ISI equalizer because it removes multiple post-cursors in the CTLE pulse-response [39], [40]. The high-frequency stage drives the pronounced peak in the higher frequency range.

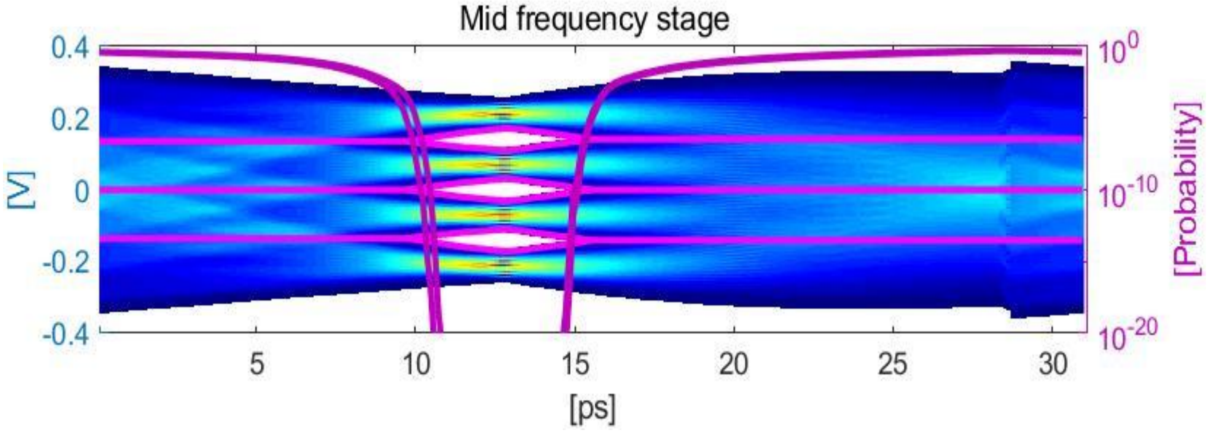


Fig. 4-13 Eye diagram at mid frequency stage of the CTLE. $e_h upp$ = 39mV, $e_w upp$ = 5.4ps, $e_h mid$ = 39mV, $e_w mid$ = 5.8ps, $e_h low$ = 39mV, $e_w low$ = 5.4ps

Figure 4-14 illustrates how each CTLE stage focuses on a specific frequency range to enhance the DC Gain. The high-frequency stage results in an improved eye opening [1].

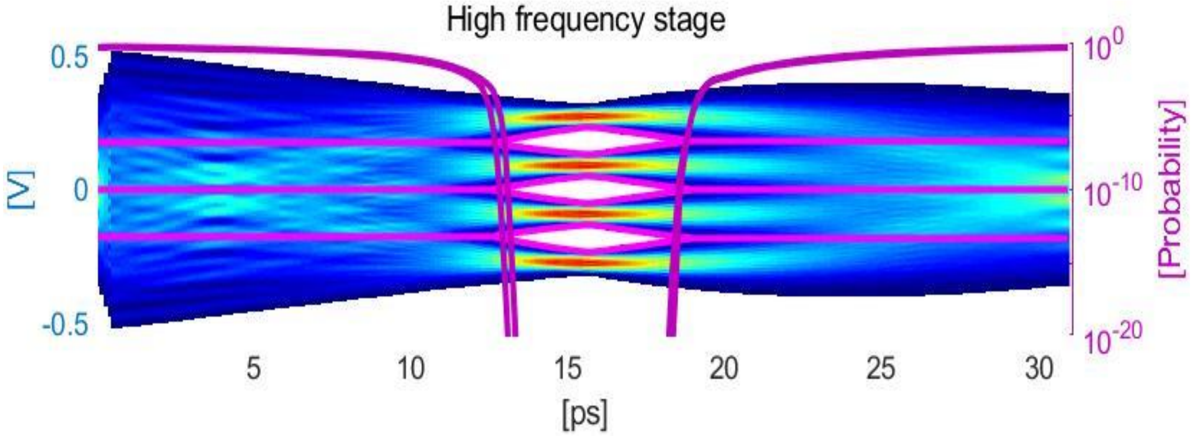


Fig. 4-14 Eye diagram at high frequency stage of the CTLE. $e_h upp$ = 68mV, $e_w upp$ = 6.1ps, $e_h mid$ = 68mV, $e_w mid$ = 6.4ps, $e_h low$ = 68mV, $e_w low$ = 6.1ps

5. PCIe Link Optimization

We aim at finding the optimal set of Tx and Rx EQ settings to maximize the eye diagram margins. Let $\boldsymbol{R}_m \in \Re^2$ denote the electrical system margins response,

$$\boldsymbol{R}_m = \boldsymbol{R}_m(\boldsymbol{x}) = [e_w(\boldsymbol{x}) \quad e_h(\boldsymbol{x})]^T \tag{5-1}$$

where $e_h \in \Re^1$ is the smallest of the three PAM4 eye height measurements and $e_w \in \Re^1$ is the smallest of the three PAM4 eye width measurements, which are functions of the Tx FFE and Rx CTLE EQ settings contained in vector $\boldsymbol{x}$ [1].

$$e_w(x) = \min(e_w low(x)\,, e_w mid(x)\,, e_w upp(x)) \tag{5-2}$$

$$e_h(x) = \min(e_h low(x)\,, e_h mid(x)\,, e_h upp(x)) \tag{5-3}$$

We need to ensure the optimal system margin response also meets an eye linearity, $e_{\text{linearity}}$, larger than 0.85, and a vertical eye closure (*VEC*) below 6 dB. An initial optimization problem can be defined through a constrained formulation,

$$u(\boldsymbol{x}) = -e_w(\boldsymbol{x})e_h(\boldsymbol{x}) \tag{5-4}$$

subject to $e_{\text{linearity}}(\boldsymbol{x}) > 0.85$ and $VEC(\boldsymbol{x}) < 6\text{dB}$, where $u(\boldsymbol{x})$ is the total area of the PAM4 eye diagram,

$$\boldsymbol{x}^* = arg \ \min_{\boldsymbol{x}} \ u(\boldsymbol{x}) \tag{5-5}$$

$e_{\text{linearity}}$ is the measure of the vertical linearity defined by the variance of amplitude separation among the different PAM4 levels, and *VEC* is the smallest of the ratios of voltage swing to eye height [1].

A more convenient unconstrained objective function is

$$u'(\boldsymbol{x}) = -w_1 u(\boldsymbol{x})\rho(\boldsymbol{x}) + w_2 \|\lambda(\boldsymbol{x})\|_2^2 \tag{5-6}$$

where $\rho(\boldsymbol{x})$ is a vertical eye closure penalty function defined as

$$\rho(\boldsymbol{x}) = 10^{-\frac{VEC(\boldsymbol{x})}{6}} \tag{5-7}$$

and $\lambda(\boldsymbol{x})$ is eye linearity penalty function defined as

$$\lambda(x) = max\{0{,}0.85 - e_{\text{linearity}}(\boldsymbol{x})\} \tag{5-8}$$

Both terms in (5-6) are scaled by weighting factors w_1, $w_2 \in \Re^1$ such that they become comparable [1]

$$w_1 = \frac{1}{u(x^{(0)})\rho(x^{(0)})} \tag{5-9}$$

$$w_2 = \frac{1}{\left\|0.85 - e_{linearity}(x)\right\|_2^2} \tag{5-10}$$

where $x^{(0)}$ is the starting point for optimization.

The initial unconstrained formulation can then be defined as

$$\boldsymbol{x}^* = arg \ \min_{\boldsymbol{x}} \ u'(\boldsymbol{x}) \tag{5-11}$$

Additionally, we need to ensure that the optimal system response lies within an appropriate region of the coefficient search space of the EQ map. Following the methodology from our previous work in [24] and [16], we redefine the corresponding objective function. As illustrated in Fig. 5-1, the four responses around $u'(\boldsymbol{x}^*)$ must be at least 80% of the value of $u'(\boldsymbol{x}^*)$. Here, $u'_{i,j}$ represents the objective function values as per equation (5-6) for the i-th C_{m1} and j-th C_{p} values [1].

The new optimization problem can thus be formulated with constraints, ensuring that the optimal set of coefficients maximizes the system response while maintaining a lower bound of 0.8u'(x*) in the surrounding area,

$$\boldsymbol{x}^* = arg \ \min_{\boldsymbol{x}} \ \mathrm{u}'(\mathbf{x}) \tag{5-12}$$

subject to $l_{11}(\boldsymbol{x}) \geq 0$, $l_{12}(\boldsymbol{x}) \geq 0$, $l_{21}(\boldsymbol{x}) \geq 0$, $l_{22}(\boldsymbol{x}) \geq 0$

with

$$\boldsymbol{l}(\boldsymbol{x}) = \begin{bmatrix} u(C_{\mathrm{m1}i*+1}, C_{\mathrm{ctle}}, C_{pj*}) & u(C_{\mathrm{m1}i*-1}, C_{\mathrm{ctle}}, C_{pj*}) \\ u(C_{\mathrm{m1}i*}, C_{\mathrm{ctle}}, C_{pj*+1}) & u(C_{\mathrm{m1}i*}, C_{\mathrm{ctle}}, C_{pj*-1}) \end{bmatrix} \tag{5-13}$$

$$-0.8u(C_{\mathrm{m1}i*}, C_{\mathrm{ctle}}, C_{pj*}) \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

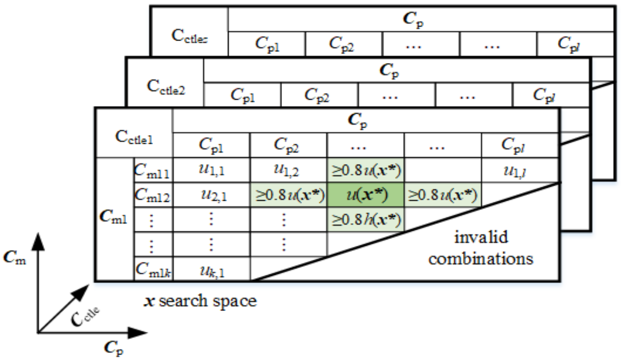


Fig. 5-1 EQ map coefficients search space for optimization. From [16].

where $C_{\mathrm{m}1i^*}$ and $C_{\mathrm{p}j^*}$ are the Tx set of coefficients that minimize (5-6) for each of the Rx CTLE setting values (C_{ctle}) [1].

Similarly, a more convenient unconstrained objective function can be defined by adding a penalty term,

$$U(\boldsymbol{x}) = u'(\boldsymbol{x}) + w_3||\boldsymbol{L}(\boldsymbol{x})||_F \tag{5-14}$$

where

$$\boldsymbol{L}(\boldsymbol{x}) = max\{0, \boldsymbol{l}(\boldsymbol{x})\} \tag{5-15}$$

and w_3 is a weighting factor [1].

$$w_3 = \frac{1}{|max\{l(x^{(0)})\}|^2} \tag{5-16}$$

Our final unconstrained formulation is

$$\boldsymbol{x}^* = arg\ \min_{\boldsymbol{x}}\ U(\boldsymbol{x}) \tag{5-17}$$

We find the optimal set of coefficients $\boldsymbol{x}^*$ by solving (5-17). To avoid the necessity of estimating gradients and recognizing the presence of numerous local minima in the objective function, we utilize a combination of pattern search [12] and Nelder-Mead [13] methods. We initiate the optimization with pattern search, exploring the design space until identifying a potential region where the global minimum may be located. Subsequently, the solution obtained through pattern search is employed as the initial seed for the Nelder-Mead method, which further minimizes the objective function for a more precise solution [1].

5.1. Optimization techniques

An optimization algorithm involves a repetitive process of evaluating different solutions until it identifies the best possible or a satisfactory solution. In numerous industrial design tasks, optimization is indirectly attained by assessing a select few design alternatives and selecting the most favorable one. A crucial element in the optimal design process involves expressing the design problem in a mathematical form that can be effectively utilized by an optimization algorithm. The theories mentioned earlier deal with either minimizing or maximizing objectives [41].

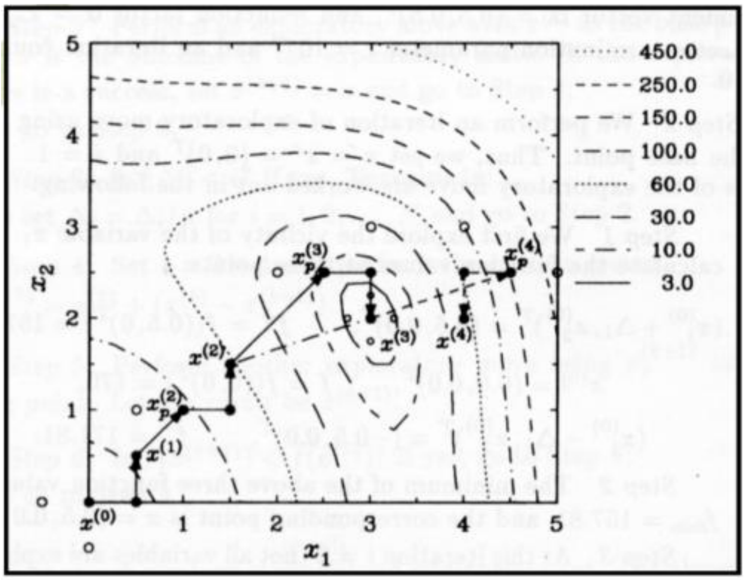


Fig. 5-2 Four iterations of Hooke-Jeeves search method. From [41].

Due to the increasing complexity of engineering design tasks, there is not a one-size-fits-all approach to efficiently address all optimization challenges. Consequently, a variety of optimization techniques have been created to address diverse types of optimization problems.

5.1.1 Hooke-Jeeves pattern search method

The pattern search method operates by systematically generating a series of search directions. These generated directions should effectively cover the entire search space, meaning that they must enable movement from any point within the search space to any other point, solely by following these prescribed search directions [41].

In the Hooke-Jeeves method, an exploratory step is taken systematically in the proximity of the current point to locate the optimal point within that vicinity. Subsequently, these two identified points are utilized to execute a pattern move as shown in Fig. 5-2.

However, given that the search primarily relies on movements along the coordinate directions during the exploratory phase, there is a risk of the algorithm converging prematurely to an incorrect solution, particularly when dealing with functions featuring significant nonlinear interactions among variables [41].

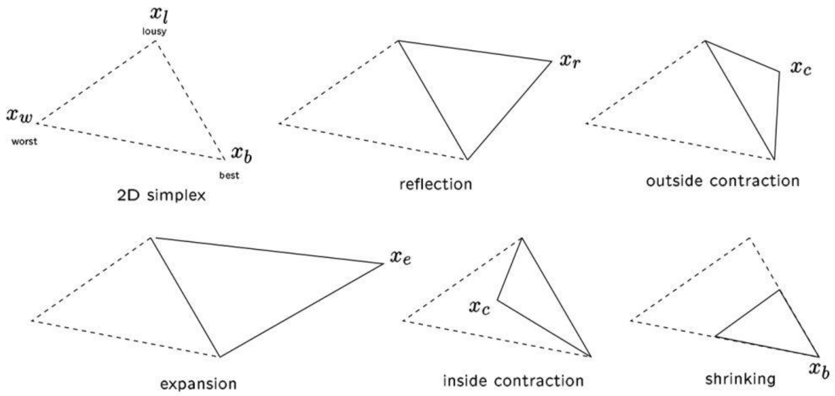


Fig. 5-3 Simplex search method. From [42].

5.1.2 Nelder-Mead method

The Nelder-Mead approach is a heuristic optimization method which aims to address the optimization problems by incorporating an element of randomness. Consequently, when applying the same algorithm to the same problem and initial conditions, one can expect varying outcomes on each run. These heuristic approaches explore the search space in a stochastic manner and maintain a record of the 'best' or most optimal solutions encountered during their exploration. It is important to note that this behavior is not entirely random but is governed by a set of predefined rules. It is crucial to keep in mind that heuristic techniques do not provide any guarantee of discovering the absolute optimal solution [43].

The Nelder-Mead method employs a geometric structure known as a simplex as its primary tool for exploring the problem domain as shown in Fig. 5-3, which is why it is also referred to as the simplex search method.

Algorithm

The Nelder-Mead method initiates its process with a randomly generated simplex. In each iteration, it systematically adjusts or relocates this simplex, vertex by vertex, in the direction of a more favorable area within the search space. At each step, the algorithm tests one or several modifications to the current simplex and selects the one that steers it closer to a 'superior' section of the domain. In an ideal scenario, the final iterations of this algorithm would entail the gradual contraction of the simplex towards the best point it contains [43].

Ultimately as shown in Fig. 5-4, the vertex within the simplex that produces the most optimal objective value is identified and returned.

This thesis did not cover other optimization techniques primarily due to the specific challenges and requirements of the design problem it addressed. The chosen optimization methods, Hooke-Jeeves pattern search and Nelder-Mead, were selected because they are well-suited for the type of optimization problem presented in the thesis. These methods are particularly effective for problems where estimating gradients is difficult and where there are numerous local minima, which is typical in the design of equalizers for high-speed data links like PCI Express Gen6.0.

The Hooke-Jeeves method systematically explores the search space and is beneficial for problems with significant nonlinear interactions among variables, which can lead to premature convergence in other methods. The Nelder-Mead method, on the other hand, is a heuristic approach that uses a simplex to explore the problem domain, making it effective for problems where the search space is complex and not easily navigable by gradient-based methods.

These methods were chosen to ensure that the optimization process could efficiently navigate the complex design space of the equalizer coefficients and achieve a robust solution that maximizes the eye diagram for optimal link performance. The combination of these methods allowed for a thorough exploration of the design space and the identification of optimal settings for the equalizer design, which was crucial given the intricate nature of equalization for multi-level signals like PAM4

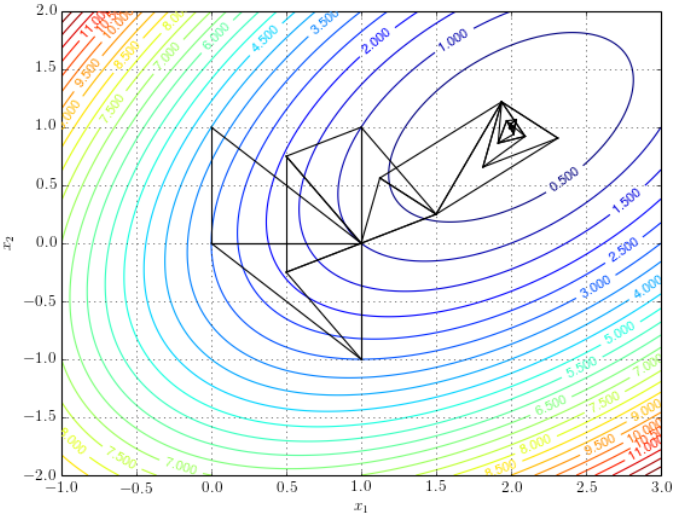


Fig. 5-4 Iterations of Nelder-Mead method. From [43].

5.2. Optimization results

To validate the proposed design within worst-case conditions, we added Tx and Rx deterministic (Tx Dj and Rx Dj) of 9e-13 and sinusoidal jitter (Tx Sj and Rx Sj) of 3e-13 to the system and proceeded to a link equalization optimization in a long-reach channel of 27 dB as

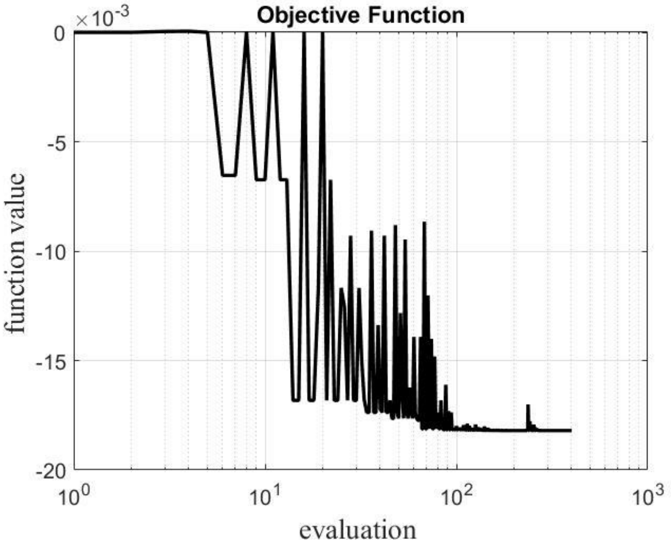


Fig.5-5 PCIe objective function values across iterations

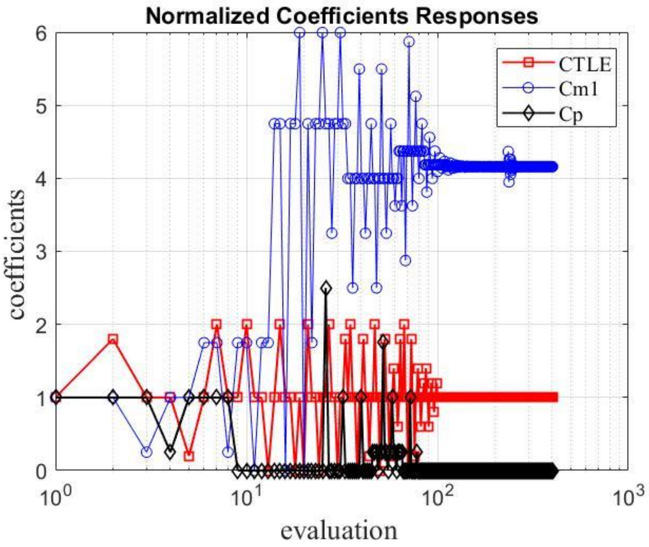


Fig. 5-6 PCIe normalized coefficients responses across iterations.

reference [1]. We aim at finding the optimal set of Tx and Rx EQ settings to maximize the eye diagram margins. We had to ensure that the optimal system response lies within an appropriate region of the coefficient search space of the EQ map and ensure at the same time to meet an eye linearity larger than 0.85, and a vertical eye closure below 6 dB.

Through the optimization process defined in the beginning of this chapter, a set of Tx and Rx optimal coefficients were found in just 401 iterations, as shown in Fig. 5-5. The initial optimization with pattern search finds a solution which was used as seed for the Nelder-Mead method to finalize the problem. Fig. 5-6 shows the evolution of the three coefficients during the optimization process.

The eye diagrams at the receiver, before (in Fig.5-7) and after (in Fig.5-8) applying the optimization process in Section 5 is shown. The resultant optimized value for the Rx was a CTLE value of 5 and the optimized results for the transmitter portion were C_{m1}=-0.1667 and C_p= 0.

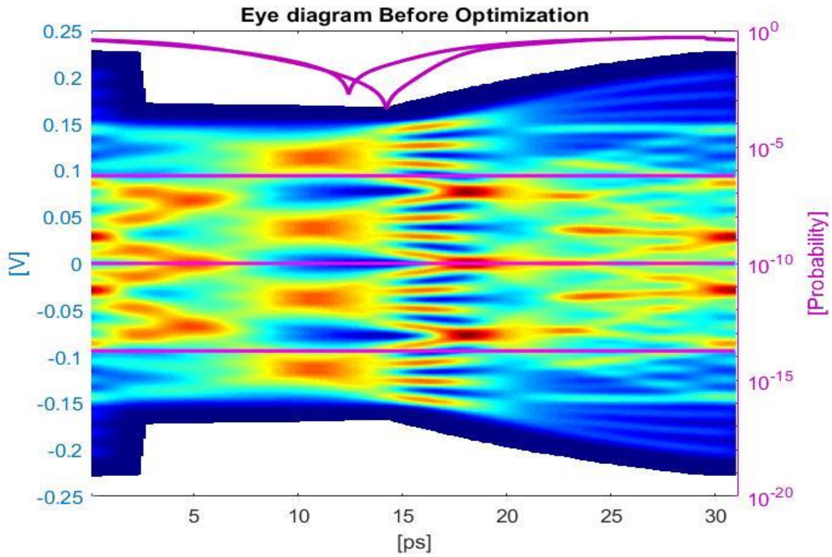


Fig. 5-7 Eye diagram before the optimization process.

Additionally, Table V-I confirms that the resultant top eye width and height amply satisfy the channel tolerancing eye mask defined in the PCIe Gen6 Spec [11]. The optimized eye-diagram, obtained under the most challenging channel conditions, validates the efficacy of the proposed optimization approach [1].

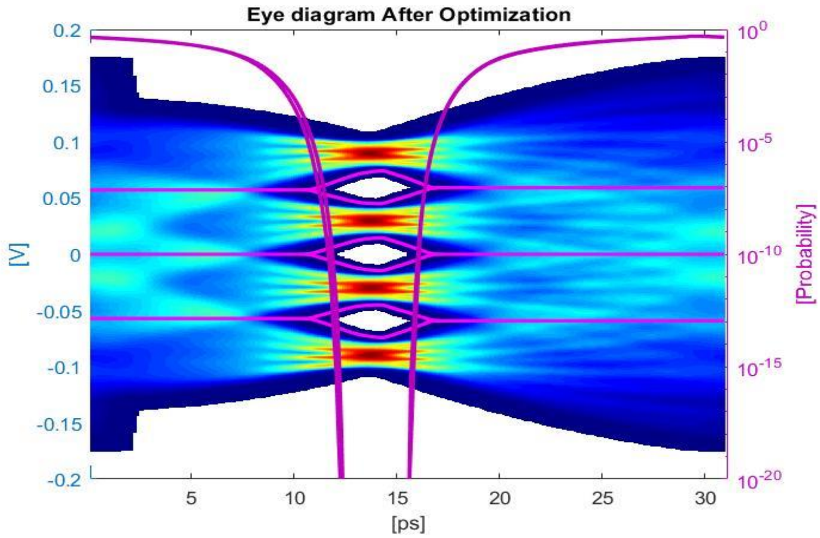


Fig. 5-8 Eye diagram after the optimization process. $e_{\mathrm{h}}upp$ = 20mV, $e_{\mathrm{w}}upp$ = 8.1ps, $e_{\mathrm{h}}mid$ = 20mV, $e_{\mathrm{w}}mid$ = 8.3ps, $e_{\mathrm{h}}low$ = 20mV, $e_{\mathrm{w}}low$ = 8.1ps, *VEC* = 5.84, $e_{\mathrm{linearity}}$ = 0.99.

TABLE V-I

64 GT/s EYE MARGINS. SPECIFICATION VERSUS SIMULATION. From [1].

Eye diagram parameter	PCIe Gen6 spec (min)	27dB channel simulation - worst-case Tx/Rx jitter parameters
top eye height	6.0 mV	20.0 mV
top eye width	0.1 UI	0.26 UI

6. Conclusion

This thesis introduces a 3-stage CTLE and a LFEQ design. This design compensates for PAM-4 PCIe Gen6 highly lossy channels considering high, mid and low frequency bands boosting stages with a 64GT/s data rate. The implementation was done in a MATLAB SerDes Toolbox environment utilizing customized blocks to implement a behavioral model derived from the investigation conducted in this thesis.

From the SerDes Toolbox behavioral design a transformation to MATLAB scripting was required to manually modify key variables. This allowed the ability to implement an optimization methodology to further improve the current design.

We additionally introduced a highly efficient optimization methodology for determining the optimal coefficients of the Tx FFE and Rx CTLE. One of the methodologies employed was Pattern search method which operates by systematically generating a series of search directions with the final goal of minimizing the defined objective function. The second methodology employed was Nelder-Mead method that utilizes a geometric framework called simplex as its primary instrument for investigating the problem domain. The optimization process started using Pattern search method, this allowed to navigate the design space until identifying a potential area containing the global minimum. Subsequently, the outcome obtained from the Pattern search method served as the initial point for the Nelder-Mead method, which proceeded to further minimize the objective function, aiming for a more precise solution.

The optimum values encountered in the optimization process fully yielded the minimization of the objective function, thus maximizing overall eye diagram area with different channel loss and jitter parameters configurations. The effectiveness of the optimized EQ coefficients was assessed by measuring the statistical eye diagrams at the receiver, demonstrating a notable enhancement in eye height, eye width, eye linearity, and vertical eye closure. We validated the proposed optimization method by using MATLAB SerDes Toolbox [1].

Appendix

A. MATLAB Code

This thesis extends the research conducted in "*PAM4 Transmitter and Receiver Equalizers Optimization for High-Speed Serial Links" thesis* [44].

CTLE main script

```
% CTLE_serdes_main
% Description: Main script that manages optimization functions.
% Company: Intel
% Authors: Karla Lopez, Francisco Rangel
% Date: 13/12/2022
% Version: 1.0
%==================================================
%                  DESIGN VARIABLES
%%% Define CTLE and FIR Filter Coefficients Vectors per PCIe Gen6 Spec
% Cm2 = 1/24
% x1: CTLE2 = [0 1 2 3 4 5 6 7 8 9 10]
% CTLE First stage of CTLE
% CTLE2 Second stage of CTLE
% CTLE3 Third stage of CTLE
% CTLE4 LFEQ
% x2: Cm1  = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24]
% x3: Cp   = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24 7/24 8/24]
%     C0   = 1 - |Cm1| - |Cp| - |Cm2|
% Define TapWeights Vector for SerDes ToolBox Simulator
% The Tx FFE block is set up for two pre-taps, one main tap, and one post-tap
by including four tap weights.
% The secuence of taps [0 Cm2 Cm1 C0 Cp] is defined per PCIe Gen6 Spec.
% txBlocks{1}.TapWeights = [0 Cm2 Cm1 C0 Cp]
%==================================================

Cm2 = 1/24; % Cm2 Rx coefficient
Cm1V = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24]; % Cm1 Tx coefficient   Cp
hasta 10*
CpV = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24 7/24 8/24]; %Cp Tx coefficient
Cp hasta 12*
CTLEV = [0 1 2 3 4 5 6 7 8 9 10]; %CTLE2

% ChannelLoss: Channel losses value
ChannelLoss = 10; %pg.1359 1365 max 28db
% SamplesPerSymbol: Number of data points per symbol.
% The Samples per symbol determine the acquisition bandwidth. PCIe Gen6 Data
Rate (Gb/s) = 64.0
SamplesPerSymbol = 64;
%  ModulationLevels: Number of logic levels in the modulation scheme: PAM4
ModulationLevels = 4;
% SymbolTime: Time it takes to send one symbol across the link.
% Per PCIe Base Specification for  PCIe Gen6: Unit Interval
% (UI(Tx))=31.246875 psec
SymbolTime = 31e-12;
```

```
% BERtarge: Target bit error rate. Target bit error rate used to generate
eye-contours,
% specified as a unitless real positive scalar. PAM4 requires higher BER at
the physical layer (~1e-6)
BERtarget = 1e-06;
% Jitter Configuration: Jitter parameters defined as Type UI:
Tx_Dj = .001e-12; % Tx Deterministic jitter pg 1350 1415
Tx_Sj = .001e-12; % Tx sinusoidal jitter pg 1350 1415
Rx_Dj = .001e-12; % Rx Deterministic jitter
Rx_Sj = .001e-12; % Rx sinusoidal jitter 1375,1377

% correr con 0 jitter....
% quitar LFEQ
%==================================================
%                     TEST TIME
% Clock inizialization to monitor optimization time
ti = clock;
clc %clears all the text from the Command Window
display(['Starting optimization at [HH:MM:SS]:'
datestr(now,'HH:MM:SS')])%prints the value of a variable or expression, X
fprintf('Running... \n') %command displays formated text

%==================================================
%                    DIRECT OPTIMIZATION
% Nelder-Mead Algorithm options configuration
MaxIter = 1e4;% Maximum Nelder mead evaluations
MaxFunEvals = 1e6; %Maximum evaluations for objective function U
tolx=1e-6; % Granularity
tolfun=1e-6; % Decimals
options =
optimset('MaxFunEvals',MaxFunEvals,'MaxIter',MaxIter,'TolX',tolx,'TolFun',tol
fun);
%3,-0.20833,-0.04167 5,-0.0417,-0.0417

%====================================================
%                   SEED definition
Xo = [5,-0.0417,-0.0417];% Seed: Xo = [CTLE2, Cm1, Cp] %try Seed should be in
range

%====================================================
%                   Variables initialization and C0 calculation
CTLE2 = Xo(1);
Cm1 = Xo(2);
Cp = Xo(3);

C0  = 1 - abs(Cm1) - abs(Cp) - abs(Cm2);

%===================================================
% u0 calculation with seed values

[u,EHj,EWj,eh6j,ew6j,VECj,ELj,PAM4_systems_Jitter] =
CTLE_serdes(Cm2,Cm1,Cp,CTLE2,C0,ChannelLoss,SamplesPerSymbol,ModulationLevels
,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
u0=u;
EHj0 = EHj;
```

```
EWj0 = EWj;
VECj0 = VECj;
ELj0 = ELj;
rho0 = 10^(-VECj0/6);
w1 = 1/abs(rho0);
w2 = 1/(norm(0.85-ELj0))^2;

% Eye plotting before optimization
figure(1)
plotStatEye(PAM4_systems_Jitter)
title('Eye diagram Before Optimization');

% fprintf('--------------------------------------------------\n')
% fprintf('PCIe Gen6 - PAM4 CTLE Optimization \n')
% fprintf('--------------------------------------------------\n')
% fprintf('\nChannel Loss [dB] ='),disp(ChannelLoss)
% fprintf('\nTx Deterministic jitter ='),disp(Tx_Dj)
% fprintf('\nTx sinusoidal jitter ='),disp(Tx_Sj)
% fprintf('\nRx Deterministic jitter ='),disp(Rx_Dj)
% fprintf('\nRx sinusoidal jitter ='),disp(Rx_Sj)
% fprintf('\nEye Height values before optimization [eh1 eh2 eh3]
='),disp(double([EHj0']))
% fprintf('\nEye Width values before optimization [ew1 ew2 ew3]
='),disp(double([EWj0']))
% fprintf('\nArea of the Eye Diagram value before optimization: AREA
='),disp(abs(u0))
% fprintf('\nVertical Eye Closure before optimization VEC ='),disp(VECj0)
% fprintf('\nEye Linearity before optimization ='),disp(ELj0)
% Count variables to monitor number of evaluations
global fevaluations U_i ctle_i Cm1_i Cp_i
fevaluations = 0;
U_i =0;
ctle_i=0;
Cm1_i=0;
Cp_i=0;

% Direct Optimization Procedure using Pattern Search Method
ObjectiveFunction = @(x)
CTLE_Fun_Opt(x,u0,w1,w2,Cm2,Cm1V,CpV,CTLEV,ChannelLoss,SamplesPerSymbol,Modul
ationLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
[x,fval] = patternsearch(ObjectiveFunction,Xo);
Xo(1) = x(1);
Xo(2) = x(2);
Xo(3) = x(3);

%====================================================
%               Optimization initializes
%               Nelder mead optimization


[Xopt,fval] =
fminsearch(@CTLE_Fun_Opt,Xo,options,u0,w1,w2,Cm2,Cm1V,CpV,CTLEV,ChannelLoss,S
amplesPerSymbol,ModulationLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj
);
```

```
% We defined a vector of values the variables could have, we need to make
% sure the otimization gives a valid value. Knobfilter helps to round the
% value to the nearest one.
[knob] = knobfilter(CTLEV',Xopt(1)); % Rounding Xo(1) to a valid value per
CTLE2 vector % "'"change vector format (from horizontal to vertical)
Xopt(1) = knob;
[knob] = knobfilter(Cm1V',Xopt(2)); % Rounding Xo(2) to a valid value per Cm
vector
Xopt(2) = knob;
[knob] = knobfilter(CpV',Xopt(3)); % Rounding Xo(3) to a valid value per Cp
vector
Xopt(3) = knob;


% We assign new optimized values to variables to recalculate eye diagram
% parameters

CTLEb = Xopt(1); % Optimal Rx CTLE coefficient values
Cm1b  = Xopt(2); % Optimal Tx Cm1 coefficient values
Cpb   = Xopt(3); % Optimal Tx Cp coefficient values
fsol  = fval; %returns in fval the value of the objective function fun at the
solution x.
C0opt = 1 - abs(Cm1b) - abs(Cpb) - abs(Cm2);
%=======================================================
% u0 calculation with optimized values

[u,EHj,EWj,eh6j,ew6j,VECj,ELj,PAM4_systems_Jitter] =
CTLE_serdes(Cm2,Cm1b,Cpb,CTLEb,C0opt,ChannelLoss,SamplesPerSymbol,ModulationL
evels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
Area_opt = abs(u);
EHj_opt = EHj;
EWj_opt = EWj;
VECj_opt = VECj;
ELj_opt = ELj;

%======================================================
%
figure(2)
plotStatEye(PAM4_systems_Jitter)
title('Eye diagram After Optimization');
%==================================================
%        Stopping clock
%                   Recording finishing time
tf = clock;
optimizationTime = etime(tf,ti)/60;
%===================================================

%                   Printing results
clc
fprintf('Optimization finished; time elapsed: %4.1f min\n',optimizationTime)
%===================================================
fprintf('--------------------------------------------------\n')
fprintf('PCIe Gen6 - PAM4 CTLE Optimization \n')
fprintf('--------------------------------------------------\n')
fprintf('\nChannel Loss [dB] ='),disp(ChannelLoss)
fprintf('\nTx Deterministic jitter ='),disp(Tx_Dj)
fprintf('\nTx sinusoidal jitter ='),disp(Tx_Sj)
fprintf('\nRx Deterministic jitter ='),disp(Rx_Dj)
fprintf('\nRx sinusoidal jitter ='),disp(Rx_Sj)
fprintf('\nEye Height values before optimization [eh1 eh2 eh3]
='),disp(double([EHj0']))
fprintf('\nEye Width values before optimization [ew1 ew2 ew3]
='),disp(double([EWj0']))
fprintf('\nArea of the Eye Diagram value before optimization: AREA
='),disp(abs(u0))
fprintf('\nVertical Eye Closure before optimization VEC ='),disp(VECj0)
fprintf('\nEye Linearity before optimization ='),disp(ELj0)
fprintf('\nSolution: \n')
fprintf('\nObjective function evaluations:'),disp(fevaluations)
fprintf('\nBest objective function : U ='),disp(fsol)
fprintf('\nOptimized Tx/Rx coefficients [CTLE Cm1 Cp] ='),disp(double([CTLEb
Cm1b Cpb]))
fprintf('\nEye Height values with optimal Tx/Rx Coefficients [eh1 eh2 eh3]
='),disp(double([EHj_opt']))
fprintf('\nEye Width values with optimal Tx/Rx Coefficients [ew1 ew2 ew3]
='),disp(double([EWj_opt']))
fprintf('\nArea of the Eye Diagram value with optimal Tx/Rx Coefficients:
AREA ='),disp(Area_opt)
fprintf('\nVertical Eye Closure with optimal Tx/Rx Coefficients: VEC
='),disp(VECj_opt)
fprintf('\nEye Linearity with optimal Tx/Rx Coefficients ='),disp(ELj_opt)
fprintf('\n\n')
%==================================================
%            PREPARING DATA FOR PLOT
        Iteration = 1:fevaluations;
        Ctle = ctle_i;
        Cm1 = Cm1_i;
        Cp = Cp_i;
        Function = U_i;
        if Ctle(1) == 0
           Ctle_norm = Ctle; %keep the same range with 1
        else
           Ctle_norm = Ctle/Ctle(1); %keep the same range with 1
        end
        if Cm1(1) == 0
           Cm1_norm = Cm1; %keep the same range with 1
        else
           Cm1_norm = Cm1/Cm1(1); %keep the same range with 1
        end
        if Cp(1) == 0
           Cp_norm = Cp; %keep the same range with 1
        else
           Cp_norm = Cp/Cp(1); %keep the same range with 1
        end

%==================================================
%          PLOTING RESULTS
    figure
    p1 = semilogx(Iteration,Ctle_norm,'r-s'); hold on
    P2 = semilogx(Iteration,Cm1_norm,'b-o'); hold on
    p3 = semilogx(Iteration,Cp_norm,'k-d');
    p1(1).LineWidth = 1;
    p2(1).LineWidth = 1;
    p3(1).LineWidth = 1;
    hold off
    set(gca,'fontsize',12)
    xlabel('evaluation ','FontName','Times','FontSize',16);
    ylabel('coefficients ','FontName','Times','FontSize',16);
    legend('\it\rmCTLE','\it\rmCm1','\it\rmCp','location','best');
    set(legend,'FontSize',12,'FontName','Times');
    title('Normalized Coefficients Responses');
    grid on;
    %
    figure
    p1 = semilogx(Iteration,Function,'k');
    p1(1).LineWidth = 2;
    set(gca,'fontsize',12)
    xlabel('evaluation ','FontName','Times','FontSize',16);
    ylabel('function value ','FontName','Times','FontSize',16);
    %legend('\it\bfC\rm_m','\it\bfC\rm_0','\it\bfC\rm_p','location','best');
    %set(legend,'FontSize',12,'FontName','Times');
    title('Objective Function');
    grid on;
%
%==================================================
%                   Equalization Map (Eye diagram Area)
%
fprintf('Running EQMap (sweeping Cm1 and Cp) at optimal CTLE value... \n')
%modify to only map the valid values on the EQ map
for C = 1:length(CpV)
        for R = 1:length(Cm1V)
           C0 = 1 - abs(Cm1V(R)) - abs(CpV(C)) - abs(Cm2);
           Eye_Diagram_Area(R,C)=
CTLE_serdes(Cm2,Cm1V(R),CpV(C),CTLEb,C0,ChannelLoss,SamplesPerSymbol,Modulati
onLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
        end
end
fprintf('Plotting EQMap... \n')
figure
contourf(CpV,Cm1V,Eye_Diagram_Area,20)
colormap(jet(100))
axis tight
grid on
colorbar
xlabel('Cp');  % add axis labels and plot title
ylabel('Cm1');
title('Eye Diagram Area Equalization Map');

figure
surf(CpV,Cm1V,Eye_Diagram_Area)

colormap(jet(100))
axis tight
grid on
colorbar
```

```
xlabel('Cp'); % add axis labels and plot title
ylabel('Cm1');
zlabel('Eye Diagram Area');
title('Eye Diagram Area Equalization Map');
```

CTLE serdes function

```
% CTLE_serdes.m
% Description: Optimization script for CTLE in PCIe gen6.
% Company: Intel
% Authors: Karla Lopez, Francisco Rangel
% Date: 13/12/2022
% Version: 1.0
% ------------------------------------
% MATLAB script to build SerDes System
% ------------------------------------
%==================================================
%                 DESIGN VARIABLES
%%% Define CTLE and FIR Filter Coefficients Vectors per PCIe Gen6 Spec
% Cm2 = 1/24
% x1: CTLE2 = [0 1 2 3 4 5 6 7 8 9 10]
% CTLE First stage of CTLE
% CTLE2 Second stage of CTLE
% CTLE3 Third stage of CTLE
% CTLE4 LFEQ
% x2: Cm1 = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24]
% x3: Cp  = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24 7/24 8/24]
%     C0  = 1 - |Cm1| - |Cp| - |Cm2|
% Define TapWeights Vector for SerDes ToolBox Simulator
% The Tx FFE block is set up for two pre-taps, one main tap, and one post-tap
by including four tap weights.
% The secuence of taps [0 Cm2 Cm1 C0 Cp] is defined per PCIe Gen6 Spec.
% txBlocks{1}.TapWeights = [0 Cm2 Cm1 C0 Cp]
%==================================================
% Cm2 = 0;
% Cm1 = 0;
% C0 = 1;
% Cp = 0;
% CTLE2 = 9;
%==================================================
%             Function definition
function[u,EHj,EWj,eh6j,ew6j,VECj,ELj,PAM4_systems_Jitter] =
CTLE_serdes(Cm2,Cm1,Cp,CTLE2,C0,ChannelLoss,SamplesPerSymbol,ModulationLevels
,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj)
%==================================================
% Build cell array of Tx blocks:
txBlocks{1} = serdes.FFE;
txBlocks{1}.BlockName = 'FFE';
txBlocks{1}.Mode = 1;
%txBlocks{1}.TapWeights = [0.0417 -0.1667 0.7916 0];
txBlocks{1}.TapWeights = [Cm2 Cm1 C0 Cp];
txBlocks{1}.Normalize = false;

% Build cell array of Rx blocks:
rxBlocks{1} = serdes.VGA;
```

```
rxBlocks{1}.BlockName = 'VGA';
rxBlocks{1}.Mode = 1;
rxBlocks{1}.Gain = 1;
rxBlocks{2} = serdes.CTLE;
rxBlocks{2}.BlockName = 'CTLE4'; %LFEQ
rxBlocks{2}.Mode = 2;
rxBlocks{2}.ConfigSelect = 0;
rxBlocks{2}.Specification = 'GPZ Matrix';
rxBlocks{2}.GPZ = [0 -320040000 -200000000 -35000000000];
rxBlocks{2}.PerformanceCriteria = 'SNR';
rxBlocks{2}.FilterMethod = 'PartialFraction';
rxBlocks{3} = serdes.CTLE;
rxBlocks{3}.BlockName = 'CTLE';
rxBlocks{3}.Mode = 2;
rxBlocks{3}.ConfigSelect = 0;
rxBlocks{3}.Specification = 'GPZ Matrix';
rxBlocks{3}.GPZ = [0 -325000000 -250000000 -32000000000];
rxBlocks{3}.PerformanceCriteria = 'SNR';
rxBlocks{3}.FilterMethod = 'PartialFraction';
rxBlocks{4} = serdes.CTLE;
rxBlocks{4}.BlockName = 'CTLE2';
rxBlocks{4}.Mode = 2;
rxBlocks{4}.ConfigSelect = 0;
%rxBlocks{4}.ConfigSelect = CTLE2; %We will use x1 to sweep CTLE vector
rxBlocks{4}.Specification = 'GPZ Matrix';
rxBlocks{4}.GPZ = [-5 -7700000000 -4330000000 -28000000000;-6 -7700000000 -
3860000000 -28000000000;-7 -7700000000 -3440000000 -28000000000;-8 -
7700000000 -3070000000 -28000000000;-9 -7700000000 -2730000000 -28000000000;-
10 -7700000000 -2430000000 -28000000000;-11 -7700000000 -2170000000 -
28000000000;-12 -7700000000 -1930000000 -28000000000;-13 -7700000000 -
1720000000 -28000000000;-14 -7700000000 -1540000000 -28000000000;-15 -
7700000000 -1370000000 -28000000000];
rxBlocks{4}.PerformanceCriteria = 'SNR';
rxBlocks{4}.FilterMethod = 'PartialFraction';
rxBlocks{5} = serdes.CTLE;
rxBlocks{5}.BlockName = 'CTLE3';
rxBlocks{5}.Mode = 2;
rxBlocks{5}.ConfigSelect = 0;
rxBlocks{5}.Specification = 'GPZ Matrix';
rxBlocks{5}.GPZ = [0 -22000000000 -7700000000 -32000000000];
rxBlocks{5}.PerformanceCriteria = 'SNR';
rxBlocks{5}.FilterMethod = 'PartialFraction';
rxBlocks{6} = serdes.DFECDR;
rxBlocks{6}.BlockName = 'DFECDR';
rxBlocks{6}.Mode = 2;
rxBlocks{6}.TapWeights = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
rxBlocks{6}.MinimumTap = [-0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -
0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7 -0.7];
rxBlocks{6}.MaximumTap = [0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7];
rxBlocks{6}.Taps2x = false;

% Build txModel:
txAnalogModel = AnalogModel( ...
      'R',50, ...
      'C',1.600000e-13);
tx = Transmitter( ...
      'Blocks',txBlocks, ...
      'AnalogModel',txAnalogModel, ...
      'RiseTime',2.905000e-12, ...
      'VoltageSwingIdeal',1, ...
      'Name','TX');

% Build rxModel:
rxAnalogModel = AnalogModel( ...
      'R',50, ...
      'C',1.600000e-13);
rx = Receiver( ...
      'Blocks',rxBlocks, ...
      'AnalogModel',rxAnalogModel, ...
      'Name','RX');

% Build ChannelData:
channel = ChannelData( ...
      'ChannelLossdB',ChannelLoss, ... %parameter used in serde_main
      'ChannelLossFreq',16000000000, ...
      'ChannelDifferentialImpedance',100);

% Build jitter parameters defined as Type UI:
% Tx_Dj = SimpleJitter('Value',0.000000000001000,'Include',true,'Type','UI');
% Tx_Sj = SimpleJitter('Value',0.000000000000600,'Include',true,'Type','UI');
% Rx_Dj = SimpleJitter('Value',0.000000000000800,'Include',true,'Type','UI');
% Rx_Sj = SimpleJitter('Value',0.000000000001400,'Include',true,'Type','UI');

% Build Jitter And Noise Object:
jitter = JitterAndNoise( ...
      'Tx_Dj',Tx_Dj,...
      'Tx_Sj',Tx_Sj,...
      'Rx_Dj',Rx_Dj,...
      'Rx_Sj',Rx_Sj,...
      'RxClockMode','clocked');

% Build SerDes System and system results
PAM4_systems_Jitter =
PAM4_sys_Jitter(tx,rx,channel,jitter,SymbolTime,SamplesPerSymbol,ModulationLe
vels,BERtarget);
vh = PAM4_systems_Jitter.Eye.Vh;
th = PAM4_systems_Jitter.Eye.Th;
stateye = PAM4_systems_Jitter.Eye.Stateye;

%Calculate Statistical Eye
[~,prefixstr2,Y2] = serdes.utilities.num2prefix(SymbolTime); %converts
numbers to their SI unit prefix.
th2 = th*SymbolTime*Y2;

   [~,~,contours,~,EH,~,~,~,~,EW,~,~,VEC,~,eyeAreas,~,COM] = ...
          serdes.utilities.calculatePAMnEye(ModulationLevels,BERtarget, ...
     th2(1),th2(length(th2)),vh(1),vh(length(vh)),stateye);
%======================================================================
%          Extract measurement variables
% Eye Diagram Measurements  with Jitter
EHj = EH; %Eye Height (mV)
EWj = EW; %Eye Width (ps)
VECj = max(VEC); % In a PAMn modulation, there are (n-1) inner eyes.
                % Each eye generates its own COM and VEC values. The app
reports the minimum of the generated COM values and the maximum of the
generated VEC values.
ELj = min(EHj)/max(EHj); % Linearity = Minimum amplitude of the different eye
levels/Maximum amplitude of the different eye levels

%    Calculate worst eye
eh6j = min(EHj); % EH6 = min(EH6low,EH6mid,EH6upp)
ew6j = min(EWj); % EW6 = min(EW6low,EW6mid,EW6upp)

%=====================================================================
%                Calculate u0 wth seed values
% The objective function is the area of the eye diagram (EW*EH). It will be
negative as we are looking to maximize the area of the eye diagram with
fminsearch
u = -eh6j*ew6j; %Eye Area (mV*ps)
end
```

CTLE function

```
%CTLE_Fun_Opt.m
% Description: Main script that manages optimization functions.
% Company: Intel
% Authors: Karla Lopez, Francisco Rangel
% Date: 13/12/2022
% Version: 1.0
%==================================================
%                  DESIGN VARIABLES
%%% Define CTLE and FIR Filter Coefficients Vectors per PCIe Gen6 Spec
% Cm2 = 1/24
% x1: CTLE = [0 1 2 3 4 5 6 7 8 9 10]
% x2: Cm1  = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24]
% x3: Cp   = -1*[0/24 1/24 2/24 3/24 4/24 5/24 6/24 7/24 8/24]
%     C0   = 1 - |Cm1| - |Cp| - |Cm2|
% Define TapWeights Vector for SerDes ToolBox Simulator
% txBlocks{1}.TapWeights = [Cm2 Cm1 C0 Cp]
%==================================================
function U=
CTLE_Fun_Opt(Xo,u0,w1,w2,Cm2,Cm1V,CpV,CTLEV,ChannelLoss,SamplesPerSymbol,Modu
lationLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj)
global fevaluations U_i ctle_i Cm1_i Cp_i
fevaluations = fevaluations + 1;

% Defining limits for Xo values based on CTLE, Cm1, and Cp vectors
if Xo(1) < CTLEV(1)
        Xo(1) = CTLEV(1);
    elseif Xo(1) > CTLEV(11)
        Xo(1) = CTLEV(11);
```

```
end
if Xo(2) > Cm1V(1)
        Xo(2) = Cm1V(1);
    elseif Xo(2) < Cm1V(7)
        Xo(2) = Cm1V(7);
end
 if Xo(3) > CpV(1)
        Xo(3) = CpV(1);
    elseif Xo(3) < CpV(9)
        Xo(3) = CpV(9);
 end

  if Xo(2)==0/24 && Xo(3)<-8/24
     U = 10000;
      return
  end

 if Xo(2)==-1/24 && Xo(3)<-7/24
     U = 10000;
      return
 end

 if Xo(2)==-2/24 && Xo(3)<-6/24
     U = 10000;
      return
 end

 if Xo(2)==-3/24 && Xo(3)<-5/24
     U = 10000;
      return
 end

 if Xo(2)==-4/24 && Xo(3)<-4/24
     U = 10000;
      return
 end

 if Xo(2)==-5/24 && Xo(3)<-3/24
     U = 10000;
      return
 end

 if Xo(2)==-6/24 && Xo(3)<-2/24
     U = 10000;
      return
 end

CTLE2 = round(Xo(1));% CTLE Gain Value
Cm1 = Xo(2);
Cp = Xo(3);
C0 = 1 - abs(Cm1) - abs(Cp) - abs(Cm2);

if C0 < 0.625
      U = 10000;
      return
```

```
end

% Compute "u" map center
[u,EHj,EWj,eh6j,ew6j,VECj,ELj] =
CTLE_serdes(Cm2,Cm1,Cp,CTLE2,C0,ChannelLoss,SamplesPerSymbol,ModulationLevels
,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
uc=u;
rho = 10^(-VECj/6);
lambda = max(0,0.85-ELj);
%
% Compute "u" up map
Cm1_up = (Cm1*24+1)/24;
C0_up = 1 - abs(Cm1_up) - abs(Cp) - abs(Cm2);
[u,EHj,EWj,eh6j,ew6j,VECj,ELj] =
CTLE_serdes(Cm2,Cm1_up,Cp,CTLE2,C0_up,ChannelLoss,SamplesPerSymbol,Modulation
Levels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
uy_p=u;
%
% Compute "u" down map
Cm1_down = (Cm1*24-1)/24;
C0_down = 1 - abs(Cm1_down) - abs(Cp) - abs(Cm2);
[u,EHj,EWj,eh6j,ew6j,VECj,ELj] =
CTLE_serdes(Cm2,Cm1_down,Cp,CTLE2,C0_down,ChannelLoss,SamplesPerSymbol,Modula
tionLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
uy_n=u;
%
% Compute "u" left map
Cp_left = (Cp*24-1)/24;
C0_left = 1 - abs(Cm1) - abs(Cp_left) - abs(Cm2);
[u,EHj,EWj,eh6j,ew6j,VECj,ELj] =
CTLE_serdes(Cm2,Cm1,Cp_left,CTLE2,C0_left,ChannelLoss,SamplesPerSymbol,Modula
tionLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
ux_n=u;
%
% Compute "u" right map
Cp_right = (Cp*24+1)/24;
C0_right = 1 - abs(Cm1) - abs(Cp_right) - abs(Cm2);
[u,EHj,EWj,eh6j,ew6j,VECj,ELj] =
CTLE_serdes(Cm2,Cm1,Cp_right,CTLE2,C0_right,ChannelLoss,SamplesPerSymbol,Modu
lationLevels,SymbolTime,BERtarget,Tx_Dj,Tx_Sj,Rx_Dj,Rx_Sj);
ux_p=u;

% Compute U(x)
l1 = abs(0.8*uc)-abs(uy_p);
l2 = abs(0.8*uc)-abs(uy_n);
l3 = abs(0.8*uc)-abs(ux_p);
l4 = abs(0.8*uc)-abs(ux_n);
L = [0,l1,l2,l3,l4]; %Define penalty vector
l = [l1,l2,l3,l4]; %% need to be static
%w3 = 1/(norm(max(l)))^2; % Compute penalty coefficient % need to be static
should evaluate with seed
%w1 = 1/abs(rho); %%new addition
%w2 = abs(uc)/(norm(0.85-ELjO))^2; %new aditions
%w3 = abs(uc)/(norm(max(l)))^2; % Compute penalty coefficient % need to be
static should evaluate with seed
```

```
%U = w1*uc*rho + w2*(norm(lambda))^2+(norm(max(L)))^2; % Compute and display
Objective %Primer termino esta en un rango de u, Segundo termino y tercer
termino
U = w1*uc*rho+abs(u0)*w2*(norm(lambda))^2+(norm(max(L)))^2;
%U1 = w1*uc*rho
%U2 =w2*(norm(lambda))^2
%U3 = w3*(norm(max(L)))^2
U_i(fevaluations) = U;
ctle_i(fevaluations) = CTLE2;
Cm1_i(fevaluations) = Cm1;
Cp_i(fevaluations) =  Cp;
%
end
```

PAM4 function

```
function[sys] =
PAM4_sys_Jitter(tx,rx,channel,jitter,SymbolTime,SamplesPerSymbol,ModulationLe
vels,BERtarget)

% SymbolTime = 31e-12;% PCIe6 UI
% SamplesPerSymbol = 16;
% ModulationLevels = 4;
% BERtarget = 1e-06;
sys = SerdesSystem(...
      'TxModel',tx,...
      'RxModel',rx,...
      'ChannelData',channel,...
    'JitterAndNoise',jitter,...
      'SymbolTime',SymbolTime, ...
      'SamplesPerSymbol',SamplesPerSymbol, ...
      'Modulation',ModulationLevels, ...
      'Signaling','Differential', ...
      'BERtarget',BERtarget);

End
```

Knobfilter function

```
function [knob] = knobfilter(s,x)

[~,ii] = min(bsxfun(@(x,y)abs(x-y),s(:).',x(:)),[],2);
knob = s(ii);

end
```

References

[1] Karla G. Lopez-Araiza, Francisco E. Rangel-Patiño, Jorge E. Ascencio-Blancarte, Edgar A. Vega-Ochoa, Jose E. Rayas-Sanchez and Omar Longoria-Gandara, “A Multi-Stage CTLE Design and Optimization for PCI Express Gen6.0 Link Equalization,” in *IEEE Latin American Elec. Devices Conf. (LAEDC)*, Puebla, Mexico, July 2023.

[2] H. Wang, Y. Chen, Y. Gao, N. Li, Z. Zhang, C. Guo and J. Li, “A quad linear 56Gbaud PAM4 transimpedance amplifier in 0.18 µm SiGe BiCMOS technology,” in *IEEE Int. System-on-Chip Conf. (SOCC 19)*, Singapore pp. 165-170, Sep. 2019.

[3] J. L. Zerbe, C. W. Werner, V. Stojanovic, F. Chen, J. Wei, G. Tsang, D. Kim, W. F. Stonecypher, A. Ho, T. P. Thrush, R. T. Kollipara, M. A. Horowitz and K. S. Donnelly, “Equalization and clock recovery for a 2.5-10-Gb/s 2-PAM/4-PAM backplane transceiver cell,” *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2121–2130, Dec. 2003.

[4] U. S. Patent 9,143,369 B2, Adaptive Backchannel Equalization (2014). Sept. 22, 2015, https://patentimages.storage.googleapis.com/d1/63/01/006c758fb4e522/US9143369.pdf.

[5] S. Parikh, T. Kao, Y. Hidaka, J. Jiang, A. Toda, S. Mcleod, W. Walker, Y. Koyanagi, T. Shibuya and J. Yamada, “A 32Gb/s wireline receiver with a low-frequency equalizer, CTLE and 2-tap DFE in 28nm CMOS,” in *IEEE Int. Solid-State Circ. Conf.*, CA, USA, pp. 28-29, Feb. 2013.

[6] N. Shamoon, “Simulation of pulse amplitude modulation 4-level (PAM-4) over PCIe gen 4.0 protocol”, California state Polytechnic University, Pomona, 2020.

[7] J. He, *High Speed serial link design with multi-level signaling and characteristic impedance extraction from transmission line with meshed ground planes*, Master Thesis, Missouri University of Science and Technology, Missouri, USA, 2017.

[8] Q. Liao et al., "The Design Techniques for High-Speed PAM4 Clock and Data Recovery," in IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA 18), Beijing, China, Nov. 2018.

[9] J. He, N. Dikhaminjia, M. Tsiklauri, J. Drewniak, A. Chada, and B. Mutnury, “Equalization enhancement approaches for PAM4 signaling for next generation speeds,” in IEEE 67th Electronic Components and Technology Conf. (ECTC 17), Orlando, FL, pp. 1874-1879, June 2017.

[10] N. Dikhaminjia, J. He, H. Deng, M. Tsiklauri, J. Drewniak, A. Chada and B. Mutnury, “Effect of improved optimization of DFE equalization on crosstalk and jitter in high speed links with multi-level signal”, in IEEE 68th Electronic Components and Technology Conference (ECTC), CA, USA, Aug. 2018.

[11] PCI-SIG, *PCI Express base specification Revision 6.1 Version 6.0 (2023).* July 24, 2023, https://members.pcisig.com/wg/PCI-SIG/document/19849.

[12] R. Hooke and T. A. Jeeves, “Direct search solution of numerical and statistical problems,” *J. of the ACM*, vol. 8, no. 2, pp. 212-229, April 1996.

[13] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, “Convergence properties of the Nelder-Mead simplex method in low dimensions,” *SIAM J. Opt.*, vol. 9, no. 1, pp. 112–147, Jan. 1998.

[14] Ravi Budruk, Don Anderson and Tom Shanley, *PCI Express System Architecture*. Sebastopol, CA: O'Reilly, 2003.

[15] Keysight, PAM-4 Design Challenges and the Implications on Test (2017). Dec. 1, 2017, https://assets-us-01.kc-usercontent.com/ecb176a6-5a2e-0000-8943-84491e5fc8d1/868a2e18-ec19-4feb-919c-56d07e120214/KS_PAM-4_Design_Challenges_5992-0527EN.pdf.

[16] R. J. Ruiz-Urbina, F. E. Rangel-Patiño, J. E. Rayas-Sánchez, E. A. Vega-Ochoa, and O. H. Longoria-Gandara, "Transmitter and Receiver Equalizers Optimization for PCI Express Gen6.0 based on PAM4", in *IEEE MTT-S Latin America Microw. Conf. (LAMC 21)*, Cali, Colombia, pp.1-4, May 2021.

[17] Tektronix, PAM-4 Signaling in High-Speed Serial Technology: Test, Analysis and Debug (2018). July 1, 2018, https://download.tek.com/document/PAM4-Signaling-in-High-Speed-Serial-Technology_55W-60273.pdf.

[18] IBM, *Comparison of PAM-4 and NRZ Signaling (2004).* March 10, 2004, https://www.ieee802.org/3/bladesg/public/mar04/anderson_01_0304.pdf.

[19] Intel, *AN 835: PAM4 Signaling Fundamentals (2019).* March 12, 2019, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/an/an835.pdf.

[20] K. Rautela and N. Belwal, "BER Performance of Gray-Coded PSK-Modulated and QA-Modulated MIMO systems," in *Second Int. on Advanced Computational and Communication Paradigms Conf. (ICACCP 19)*, Bhimtal, India, Feb. 2019.

[21] J. Lee, P. Chiang, P. Peng, L. Chen and 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.

[22] J. Im et al., "A 40-to-56 Gb/s PAM-4 Receiver with Ten-Tap Direct Decision-Feedback Equalization in 16-nm FinFET," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 12, pp. 3486-3502, Dec. 2017.

[23] Cambridge University Press, *Intersymbol interference and equalization (2012).* June 5, 2012, https://www.cambridge.org/core/books/abs/theory-and-design-of-digital-communication systems/intersymbol-interference-and-equalization/7FC40B1FBCC2AD8E234892E3F92481BE.

[24] F. E. Rangel-Patiño, J. E. Rayas-Sánchez, E. A. Vega-Ochoa, and N. Hakim, "Direct optimization of a PCI Express link equalization in industrial post-silicon validation," in *IEEE Latin American Test Symp. (LATS 18)*, Sao Paulo, Brazil, Mar. 2018, pp. 1-6.

[25] U. Spagnolini, *Statistical Signal Processing in Engineering*. Sebastopol, CA: O'Reilly, 2018.

[26] Logic Fruit Technologies, PCI Express 3.0 Equalization: The Mystery Unsolved (2020). August 27, 2020, https://www.logic-fruit.com/whitepaper/pci-express-3-0-equalization/.

[27] W. T. Beyene, "The design of continuous-time linear equalizers using model order reduction techniques," in *IEEE Elec. Perform. Electron. Packaging (EPEP 08)*, San Jose, CA, pp. 187-190, Oct. 2008.

[28] Keysight, *Continuous Time Linear Equalizer Operator (2014).* Aug. 1, 2014, https://rfmw.em.keysight.com/DigitalPhotonics/flexdca/UG/Content/Topics/Signal-Processing/Signal-Processing-Operators/ctle.htm.

[29] Teledyne Lecroy, *Continuous Time Linear Equalization (2018).* July 17, 2018, http://blog.teledynelecroy.com/2018/07/continuous-time-linear-equalization.html.

[30] R. Farjad-Rad and et aI., “0.622-8.0 Gbps 150 mW serial 10 macrocell with fully flexible preemphasis and equalization,” in *Symp. VISI Circuits Dig. Tech. Papers*, Kyoto, Japan, June 2003.

[31] S. Gondi and B. Razavi, "Equalization and clock and data recovery techniques for 10-Gb/s CMOS serial-link receiver", *Journal of SolidState Circuits*, vol 42, no. 9, Sep. 2007.

[32] Keysight, *Decision Feedback Equalizer Operator (2014)*. Aug. 1, 2014, https://rfmw.em.keysight.com/DigitalPhotonics/flexdca/UG/Content/Topics/Signal-Processing/Signal-Processing-Operators/dfe.htm#:~:text=The%20Decision%20Feedback%20Equalizer%20(DFE,caused%20by%20the%20previous%20symbols.

[33] F. Rangel-Patiño, *Transmitter and Receiver Equalizers Optimization Methodologies for High-Speed Links in Industrial Computer Platform Post-Silicon Validation*, Ph.D. Thesis, ITESO, Tlaquepaque, Mexico, 2018.

[34] Texas A&M University, Lecture 19: Rx DFE Equalization (2010). March 1, 2010, https://people.engr.tamu.edu/spalermo/ecen689/lecture19_ee689_rx_dfe_eq.pdf

[35] J. He, N. Qi, N. Yu, L. Wu, P. Yin Chiang, X. Xiao and N. Wu., “A 2nd-order CTLE in 130nm SiGe BiCMOS for a 50GBaud PAM4 Optical Driver,” in *IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA 18)*, Beijing, China, p.151, Nov. 2018.

[36] MathWorks, *SerDes Toolbox (1984)*. Dec. 7, 1984, https://www.mathworks.com/products/serdes.html.

[37] Ansys Optics, PAM4 Transceiver - INTERCONNECT Statistical Simulation (2024). Jan. 1, 2024, https://optics.ansys.com/hc/en-us/articles/360043451834-PAM4-Transceiver-INTERCONNECT-Statistical-Simulation

[38] Keysight, PAM4 Eye Linearity (2023). Nov. 1, 2023, https://helpfiles.keysight.com/scopes/FlexDCA-UG/Content/Topics/Eye-Mask-Mode/PAM-Measurements/pam_eye_linearity.htm.

[39] M. Aggarwal, *Low power analog front end design for 112 Gbps PAM-4 SERDES receiver*, Master Thesis, University of Toronto, Toronto, Canada, 2020.

[40] J. Liao, E. Ma, J. Chen and G. Zhang, “25G long reach cable link system equalization optimization,” in *DesignCon*, 2016.

[41] G. S. Kirgat and A. N. Surde, “Review of Hooke and Jeeves Direct Search Solution Method Analysis Applicable To Mechanical Design Engineering,” *International Journal of Innovations in Engineering Research and technology*, vol. 2, issue 2, pp. 1-10, Dec. 2014.

[42] A. Mohsin, Y. M. Alsmadi, A. Arshad, S. M. Gulfam, “A modified simplex based direct search optimization algorithm for adaptive transversal FIR filters,” Science in progress, April 2021.

[43] S. Joglekar, *Nelder-Mead Optimization (2016).* Jan. 16, 2016, https://codesachin.wordpress.com/2016/01/16/nelder-mead-optimization/.

[44] R. Ruiz, *PAM4 Transmitter and Receiver Equalizers Optimization for High-Speed Serial links*, Master Thesis, The Jesuit University of Guadalajara, Jalisco, Mexico, 2021.