DESIGNCON® 2024
WHERE THE CHIP MEETS THE BOARD

Low latency speculative error correction using simplified ML detector for 64Gbps wireline transceiver

Ehud Nir,
Mansi Mehrotra,
Amin Karami,
Chris Holdenried,
Robert Wang

Cadence Design Systems, Toronto, Canada

Abstract

Wireline interconnect data-rates have been doubling consistently every 3-4 years due to increasing demand for higher connectivity speed in datacenters. With limited improvement of the electrical channels, MLSE has emerged as a key digital technique for boosting signal power in wireline links, at the price of increased latency and die area.

This paper presents an ultra-low latency speculative error correction (SEC) engine embedded within the DFE. It uses a simplified ML detector to correct single symbol errors before they propagate into a burst. Significant reduction of the trellis size is achieved by using soft information together with assumptions on the error distribution. The design was customized for an existing Gen6 PCIe PHY in 5nm CMOS technology. Simulated and measured SNR gain exhibits comparable performance at a fraction of the latency and area of full MLSE.

Authors Biography

Ehud Nir

Ehud Nir received the B.Sc. and M.Sc. degrees in Electronics Engineering from Tel Aviv University, Tel Aviv, Israel, in 2001 and 2013, respectively. He has over 25 years of experience in custom digital and DSP circuit design. He is currently a director of digital engineering at Cadence Design Systems Inc., developing high-speed digital circuits for wireline communication.

Mansi Mehrotra

Mansi Mehrotra received the Bachelor of Engineering degree in Electronics Engineering from Indian Institute of Technology, BHU (IIT BHU), India, in 2012. She has over 11 years of experience in digital design of Controllers and PHY Subsystems. She is currently working in high-speed Serdes group at Cadence Design Systems Inc., Noida, India

Amin Karami

Amin Karami received the B.Sc. degree in electrical engineering from Sharif University of Technology, Tehran, Iran in 2013, the M.Sc. degree from Iran University of Science and Technology, Tehran, Iran in 2016, and the Ph.D. degree in electrical engineering from University of Alberta, Edmonton, AB, Canada. He is currently working in high-speed Serdes group at Cadence Design Systems Inc., Toronto, ON, Canada.

Chris Holdenried

Chris Holdenried received the B.Sc. and Ph.D. degrees in electrical engineering from the University of Calgary in 2000 and 2005 respectively. He is currently a director in the high-speed Serdes group at Cadence Design Systems Inc., Toronto, ON, Canada.

Robert Wang

Robert worked at Intel before joining Snowbush in Toronto, Canada as an analog design engineer. He then became a founding team member and SerDes architect at VSemiconductor in 2008. Following VSemiconductor's acquisition by Intel, Robert took on various lP and serdes related roles. In 2017, he co-founded AnalogX, where he served as CEO until the company was successfully acquired by Rambus Inc. Robert holds a MASc in Electrical Engineering from University of Toronto and an BASc from University of Waterloo.

Introduction

Maximum-likelihood sequence estimation (MLSE) [4], is an equalization scheme that looks for the most likely transmitted sequence rather than taking symbol-by-symbol decisions . Compared with FFE-DFE receiver which only exploits the main cursor in the equalization, MLSE improves the SNR by taking advantage of postcursor ISI energy as well. Its main disadvantage is its computational complexity, which grows exponentially with the number of bits per symbol and the partially equalized channel memory length. Several methods were proposed to reduce full MLSE implementation complexity, including: Reduced State Sequence Estimation (RSSE) which reduces the area, but the latency is still block size dependent[5]; Null-Zone DFE (NZDFE) which incorporates ML detector in the feedback and approaches full MLSE performance at the cost of significantly higher DFE complexity[6]; and Dual Decision Feedback Equalizer (DDFE) with reduced feedback complexity only when deployed in baud rate analog DFEs[7]. The speculative error correction (SEC) engine proposed hereunder extends the parallel DFE loop unrolling implementation method to include a ML detector. This method doesn't increase the feedback complexity, and the resulting performance is approaching that of a full MLSE with significantly lower added latency and area.

The SEC includes 3 parts: 1. Erasure zone check for DFE loop unrolling candidates 2. Simplified ML detector, and 3. Candidate error correction. The number of candidates is reduced from 4 to 2, by deploying partially unrolled (PUDFE) architecture, in which soft information from an auxiliary FFE and slicer are used to rule out two unlikely decisions [8]. The SEC reuses the definition of 'erasure zone' from [7], as the candidate's probability density function (PDF) section bounded by ±ε around the threshold levels. Candidates crossing into this zone are more likely to have an error than other areas. Such candidates are marked within the pre-compute slicer block. Next, the marked candidates and their possible error corrected value are expanded into the future by using a short loop unrolling DFE multiplexer. ML detector is used to select between the two sequences and possibly point out an error in the original pre-compute stage. Candidate error, if found, is then fixed and the subsequent logic is identical to that of regular PUDFE. The sequence length and the erasure zone width are selected to maximize the number of errors corrected by the SEC and minimize latency. The design was customized for a 64Gbps Gen6 PCIe transceiver and synthesized in 5nm CMOS technology. SNR gain performance was simulated and measured by post processing transceiver ADC samples, reaching 0.03dB from full MLSE. The resulting circuit adds ~1ns (~1cycle) to existing DFE and has synthesis cell area of $4500\ \mu m^2$.

The rest of the paper is organized as follows: In Section II, we give background and analyze the implementation of conventional MLSE. In Section III we explain the rationale behind DDFE and PUDFE. Section IV outlines the proposed method to co-design the PUDFE with ML detector, resulting in the SEC. The algorithm, simulated performance and circuit level implementation are described. Section V presents the offline processing setup used to measure the SEC performance in PCIe Gen6 PHY and the collected results. The paper ends with a summary of findings and conclusions.

Conventional MLSE complexity analysis

1. The MLSE Algorithm

A simplified model of an ADC-DSP SerDes link which includes the transmitter, channel, noise source, and receiver is shown in Figure 1. The transmitter is emitting a PAM4 modulated sequence $v_k \in \{-3, -1, +1, +3\}$. The SerDes channel is represented as a time dispersive channel with additive white gaussian noise (AWGN) which is denoted by $\{w_k\}$. The receiver is processing the channel output symbols $\{r_k\}$, in a data path that includes ADC, FFE and an option to select between DFE or MLSE [1][2].

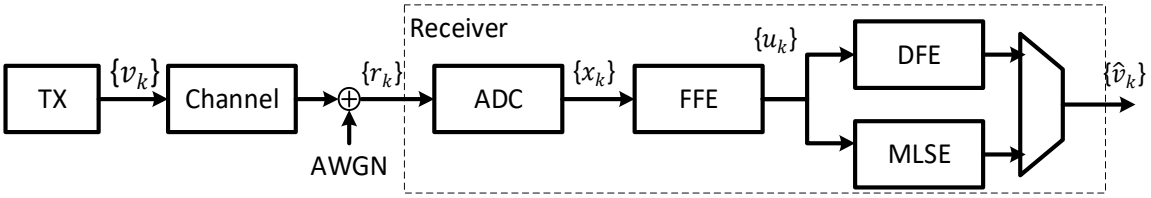

Figure 1: ADC-DSP SerDes link model

The FFE is assumed to remove all ISI except the 1st post cursor, resulting in an equivalent $1 + \alpha D$ channel with additive white gaussian noise w_k:

Eq (1)

$$u_k = v_k + \alpha v_{k-1} + w_k$$

where α is positive and equal to the magnitude of the post cursor ISI. When the DFE is selected, a symbol-by-symbol decision is taken to recover the transmitted signal from u_k, by eliminating the remaining ISI using feedback from previous decision $\hat{v}_{k-1}$. The decision is made by the DFE slicer, which input y_k is equal to:

Eq (2)

$$y_k = v_k + \alpha(v_{k-1} - \hat{v}_{k-1}) + w_k$$

Assuming previous symbol decision was correct $(v_{k-1} - \hat{v}_{k-1} = 0)$, a random error will occur when a high w_k drives y_k beyond the slicer's threshold level, and lead to a decision error in the DFE output:

Eq (3)

$$\hat{v}_k = v_k + e_k$$

where $e_k = \pm 2$ is a single random error. As shown in [3], in a DFE such error may propagate to future decision(s) as well and create a burst (Assuming only single level errors):

Eq (4)

$$y_{k+1} = v_{k+1} + \alpha(v_k - \hat{v}_k) + w_{k+1} \xrightarrow{v_k - \hat{v}_k = -e_k} v_{k+1} - \alpha e_k + w_{k+1}$$

$$e_{k+1} = \begin{cases} -2 \; if \; (w_{k+1} - \alpha e_k) < -1 \; and \; v_{k+1} \in \{-1, +1, +3\} \\ +2 \; if \; (w_{k+1} - \alpha e_k) > +1 \; and \; v_{k+1} \in \{-3, -1, +1\} \\ 0 \; otherwise \end{cases}$$

The error propagation is the main drawback of the DFE.

In contrast to the symbol-by-symbol operation of the DFE, the MLSE algorithm is searching for the most likely series $\{\hat{v}_k\}$ to lead to the FFE output sequence $\{u_k\}$. Given that the symbols at the transmitter are independently and identically distributed, and that w_k is AWGN, the most likely series will bring to minimum the sum of Euclidean square distance between $\{u_k\}$ and $\{\hat{v}_k\}$ convoluted with the channel response [9]:

Eq (5)

$$[\hat{v}_0, \hat{v}_1, \cdots \hat{v}_k] = \arg\min \sum_{k=0}^{\infty} \|u_k - \hat{v}_k - \alpha * \hat{v}_{k-1}\|^2$$

The MLSE SNR is higher than that of the DFE since it is using the post cursor energy as well. It is less sensitive to error propagation as seen in Eq (4), but its complexity is significantly higher - for the PAM4 signal sequence of length k, the MLSE needs to compare 4^k variations of the sequence to select the most likely one.

2. The Viterbi Algorithm

VA is a well-known method to implement the MLSE algorithm with significantly reduced computational complexity. The VA is operating on a trellis diagram as shown in Figure 2 (a).

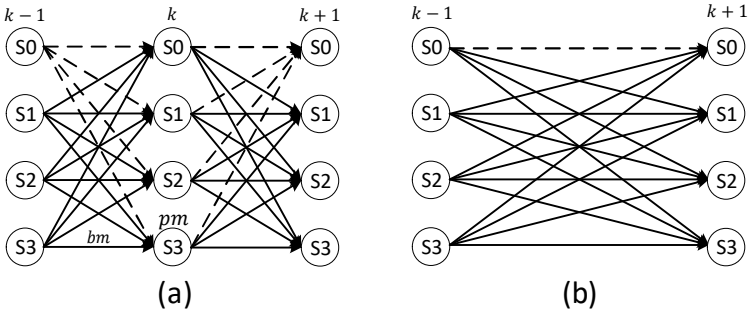

Figure 2: (a) 4 state VA Trellis diagram and (b) processed by one look ahead layer.

The vertical nodes s_k represent the states at time k. The number of states for PAM4 modulated signal and $1 + \alpha D$ channel is equal to 4. Each transition from a state at time $k-1$ (s_{k-1}) to a state at time k (s_k) is called a branch and is represented by a branch metric (bm), which is defined as:

Eq (6)

$$bm(s_{k-1}, s_k, k) = \|u_k - v'_k - \alpha * v'_{k-1}\|^2 = (u_k - v'_k - \alpha * v'_{k-1})^2$$

bm is the Euclidian distance between the FFE output and a speculation for the symbol that was transmitted at time k. Each state is associated with a path metric (pm) , equal to the minimum sum of bm 's leading to this state, and calculated as:

Eq (7)

$$pm(s_{k-1}, k) = \min [pm(S0, k-1) + bm(S0, s_k, k),$$

$$pm(S1, k-1) + bm(S1, s_k, k),$$
$$pm(S2, k-1) + bm(S2, s_k, k),$$
$$pm(S3, k-1) + bm(S3, s_k, k)]$$

The path metric calculation is named add, compare, select (ACS). Each time it is calculated, the previous state s_{k-1} resulting in the minimum sum assigned to current state's pm is stored. One of the properties of the Viterbi decoder is that starting from each state at the beginning or at the end of the trellis and tracing inwards, all paths converge. The depth of convergence is typically $5v$, where v is the encoder depth [10]. In the case of $1 + \alpha D$ channel, $v = 1$ and the depth of convergence is 5. The trellis depth should be sufficiently long to allow all the paths to converge. Under this consideration, the path ending at the state with the smallest pm at time k can be used to recover the transmitted sequence until time $\leq k - 5$ by tracing back the stored states along this path. This adheres to MLSE equation number Eq (5) . The number of sequences considered by the VA is reduced from 4^k to $(k-1) * 4^2$.

3. High throughput Viterbi decoder

High speed SerDes processes M symbols in parallel each cycle. This is achieved by heavily pipelining the DSP logic. Considering the path convergence property mentioned above, there is no need to accumulate the entire history of the path metric in Eq (7). As described in [11], M symbols block can be processed independently through the VA if it's padded with *L* symbols before (named sync) and after (named traceback), where *L* is greater than the depth of convergence. This will guarantee the extraction of the maximum likelihood path in the trellis for the inner M processed symbols. Since blocks are processed in pipeline, the sync padding can be replaced with the pm's of the last state in the previous block [12]. Likewise, the traceback padding can be replaced with the result of the traceback operation of the next cycle (either are using $M > 5$ symbols hence depth of convergence is guaranteed). This architecture is named sliding block Viterbi decoder (SBVD).

The pipelining of the Viterbi algorithm is enabled by *look ahead* operation. Each cycle, every other unit interval states are looked over and replaced with branch metrics going from time *k-1* to *k+1*. 16 ACS units are needed per unit interval. Each one of them is creating a single branch metric out of a pair, which sum is the minimum among all four possible pairs, starting and ending at the same states.:

Eq (8)

$$bm(s_{k-1}, s_{k+1}, k+1) = \min [bm(s_{k-1}, S0, k) + bm(S0, s_{k+1}, k+1),$$
$$bm(s_{k-1}, S1, k) + bm(S1, s_{k+1}, k+1),$$
$$bm(s_{k-1}, S2, k) + bm(S2, s_{k+1}, k+1),$$
$$bm(s_{k-1}, S3, k) + bm(S3, s_{k+1}, k+1)]$$

This result is illustrated in transition from Figure 2(a) to Figure 2(b). Once two unit intervals are left, a single ACS block is used to calculate the four path metrics of the block, based on previous block path metrics. Figure 3 illustrates this architecture.

Assuming an ACS operation takes one cycle to complete in advance technologies, the overall latency of the SBVD for $1 + \alpha D$ channel will be (including 1 cycle for the BM calculation):

Eq (9)

$$L_{LA\_VD} = M + 1$$

We will use the number of 2 input adders and comparators in the ACS units as a measure for the arithmetic complexity throughout this paper. The ACS in equation Eq (7) requires 4 adders and 3 comparators per state. Each look ahead time slot (first M-1 time slots in Figure 3) requires a total of 16 such ACS units. Each unit is used to eliminate ("*look ahead*" over) intermediate state at time *k* by comparing the 4 possibilities to reach from one state in time *k-1* to another state in time *k+1*. One example is highlighted using dashed lines in the transition from Figure 2(a) to (b).

The final path metric calculation requires another 4 ACS blocks. The overall complexity of the look ahead VD is:

Eq (10)

$$\mathrm{C}_{LA\_VD} = 7 * (16 * (M - 1) + 4)$$

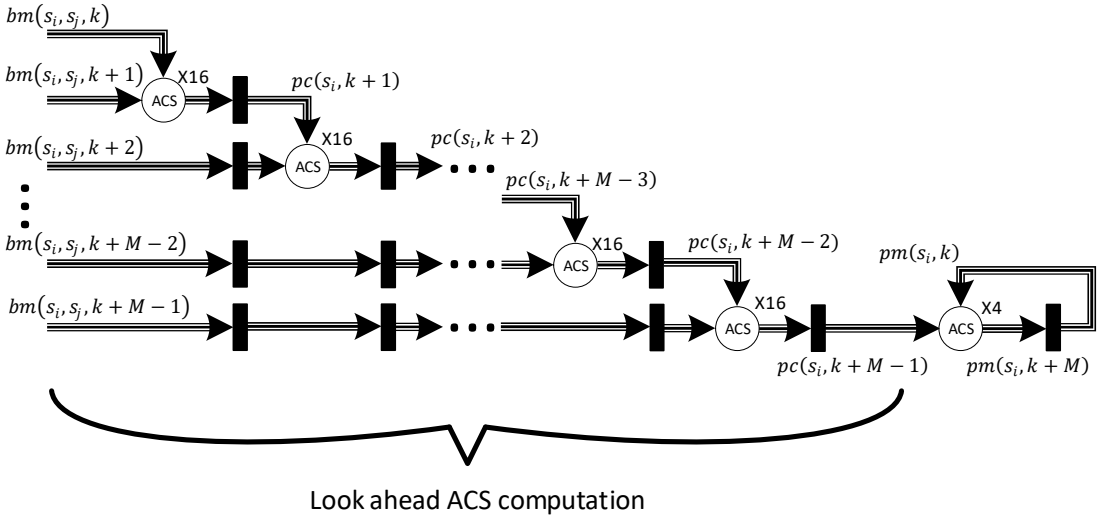

Figure 3: Parallel processing in VD using look ahead ACS

As shown in [12], this implementation can be further improved by applying the look ahead units on the entire block simultaneously. The resulting *layered* look ahead architecture is illustrated in Figure 4.

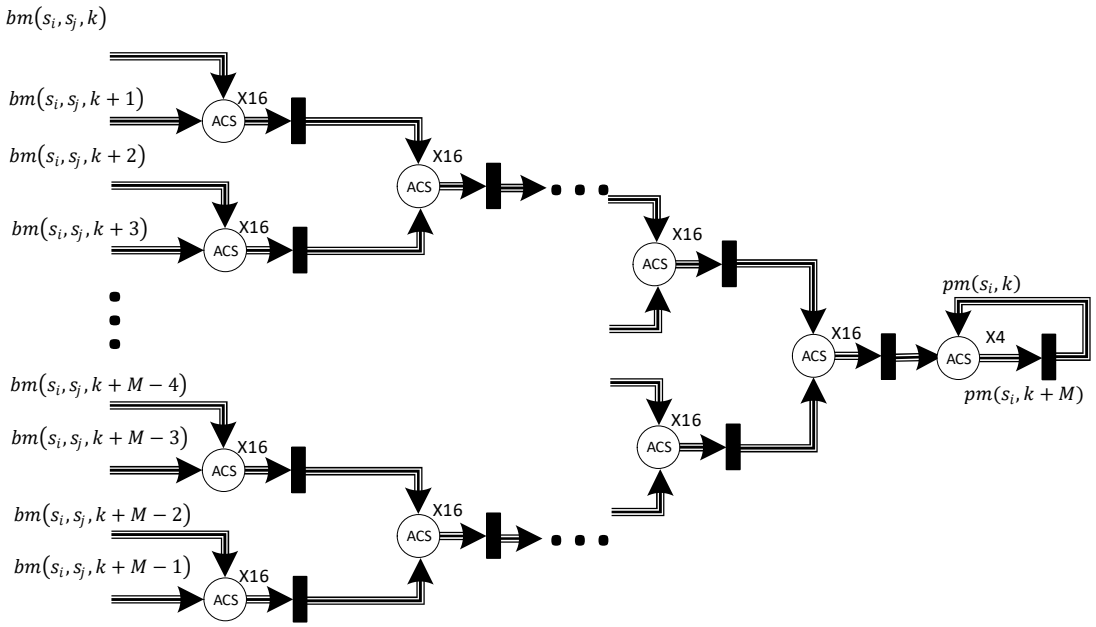

Figure 4: Layered look ahead Viterbi decode architecture.

The layered look ahead (LLA) architecture is reducing the VD latency, while the arithmetic complexity remains the same:

Eq (11)

$$L_{LLA\_VD} = \log M + 1$$

Eq (12)

$$C_{LLA\_VD} = 7 * (16 * (M - 1) + 4)$$

Reduced state sequence estimation architecture can be incorporated to significantly reduce the arithmetic complexity by applying set partitioning on the trellis [5]. Assuming single look ahead stage per cycle (set by the technology speed), the latency remains block size dependent, like the analysis above.

DFE architecture evolution

1. Dual Decision Feedback Equalizer

DDFE is processing the FFE output with two conventional DFEs in parallel[7]. They are completely identical except for the thresholds level feeding their slicers. As illustrated in Figure 5 (PAM4 version), the top DFE is using PAM4 thresholds skewed up by ε, while the bottom DFE uses PAM4 thresholds skewed down by − ε. ε range is [0,1] , where 1 is the +1 level amplitude. The zone [−ε, ε] around the thresholds is defined as erasure zone. Equalized signal falling within this zone are considered unreliable. If the additive noise is small, then the decisions of both DFEs will match the transmitted symbol. However, if the slicer input value falls within the erasure zone, then the decisions of the top and bottom DFEs will differ and an erasure period will be flagged. When that occurs, the thresholds of the two DFEs will switch to non-skewed values (ε=0). The error energy of each DFE is computed across erasure period (or detection

delay, δ). At the end of erasure period, the DFE associated with the smallest error energy delivers the final decision, and the register contents of the selected DFE are transferred to the other one to realign the two DFEs. From this point on, the search for erasure commences again.

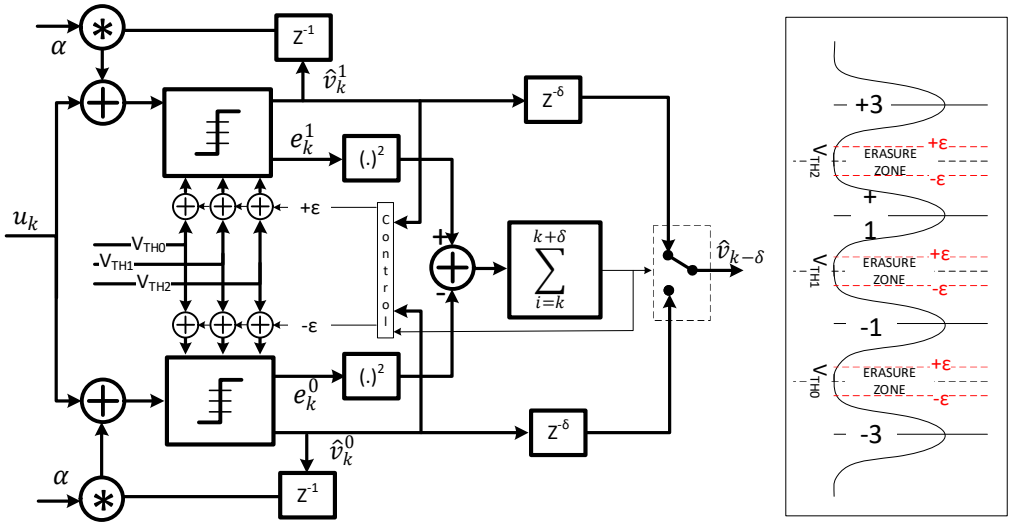

Figure 5: Dual Decision DFE and PAM 4 erasure zones

The DDFE scheme is designed to prevent error propagation by eliminating the initial symbol error decision from occurring. The error behavior of the DDFE is analyzed for time $k = k_0$ where the entire contents of the shift registers are error free and identical. The possibilities below exist for $\hat{v}_k^0$and $\hat{v}_k^1$:

- Decisions $\hat{v}_k^0$and $\hat{v}_k^1$ are both correct, DDFE is error free.
- Either decision is wrong, and an erasure period commences. At the end of the period, the DDFE will return to an error free state with respect to time k_0, with these exceptions:
  - The DFE that produces the correct decision $\hat{v}_k^0$, had one or more decision errors within the erasure period – error type I.
  - The final decision taken at the end of the erasure period is incorrect – error type II.
- Decisions $\hat{v}_k^0$and $\hat{v}_k^1$ are both wrong – error type III.

The respective probabilities of these error types were approximated in [7], and listed Table 1:

Table 1: DDFE error probabilities

Error Type	Error probability	Dependency
I	Eq (13)	p_I increases with ϵ and δ (Dominant when erasure zone is too wide, and detection delay is too long –

Error Type	Error probability	Dependency
	$p_I \propto \delta Q\left(\sqrt{\frac{1+(1-\epsilon)^2}{\sigma^2}}\right)$	probability to include more errors within the detection delay increases)
II	Eq (14) $p_{II} \propto Q\left(\sqrt{\frac{d_\delta^2}{\sigma^2}}\right)$	p_{II} decreases with δ (Dominant when detection delay is too short). p_{II} minimum value is reached when δ is equal to the depth of convergence, *5* (This also governs the performance of the optimum detector)
III	Eq (15) $p_{III} \propto Q\left(\sqrt{\frac{(1+\epsilon)^2}{\sigma^2}}\right)$	p_{III} decreases with ϵ (Dominant when erasure zone is too narrow - probability to "miss" more errors increase)

Where $\epsilon \in [0,1]$, d_δ^2 is the minimum Euclidian distance between any two admissible data sequences across the memory span δ, and $Q(x)$ is the Q-function, equal to the probability that a standard normal random variable will take a value higher than x (monotonically decreasing):

Eq (16)

$$Q(x) \triangleq \frac{1}{\sqrt{2\pi}} \int_x^\infty exp\left\{\frac{-y^2}{2}\right\} dy$$

The DDFE is best suited for analog baud rate implementation. The control logic imposes a constraint on the final decision hence not suited to be included as is with a look ahead parallel type of implementation. Furthermore, the serial implementation has a tradeoff between the p_I and p_{II} with respect to the memory span δ. As δ is increased to minimize p_{II}, p_I is increasing and the other decision errors occurring within the erasure period are not inspected. This becomes more significant as ϵ is increased to minimize p_{III}.

2. Partially unrolled DFE

The architecture of 1 tap Partially unrolled DFE (PUDFE), illustrated in Figure 6 , was originally introduced in [8]. A conventional loop-unrolled DFE requires computation of all the possible equalized values for the symbol at time k, which involves 4 possible choices for the symbol at time $k-1$. PUDFE is eliminating 2 unlikely choices out of the 4, based on soft information from a parallel FFE-Slicer block performing only partial equalization (to meet the timing requirements and save power). This parallel slicer

decision is used to define 3 regions at the output of the parallel FFE, shown in Figure 7. Regions are bounded by a distance of h_0 from the slicer thresholds. As shown in [8], the probability of a symbol residing in one region to result from a transmitted symbol in a different region is significantly lower than the target BER, hence neglected.

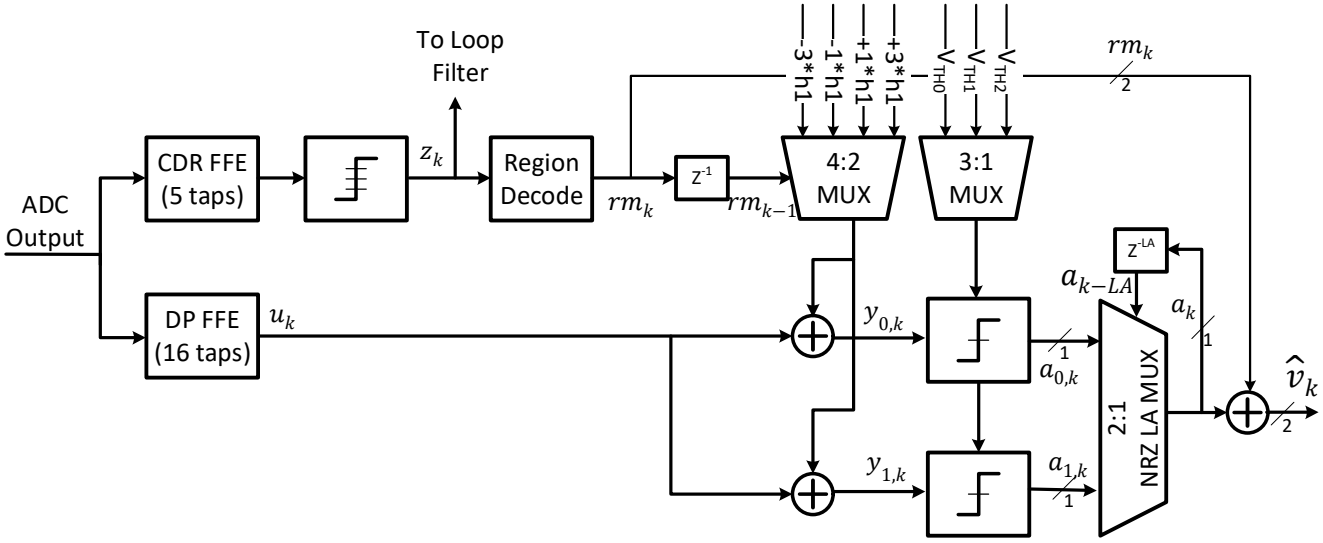

Figure 6: Partially unrolled DFE

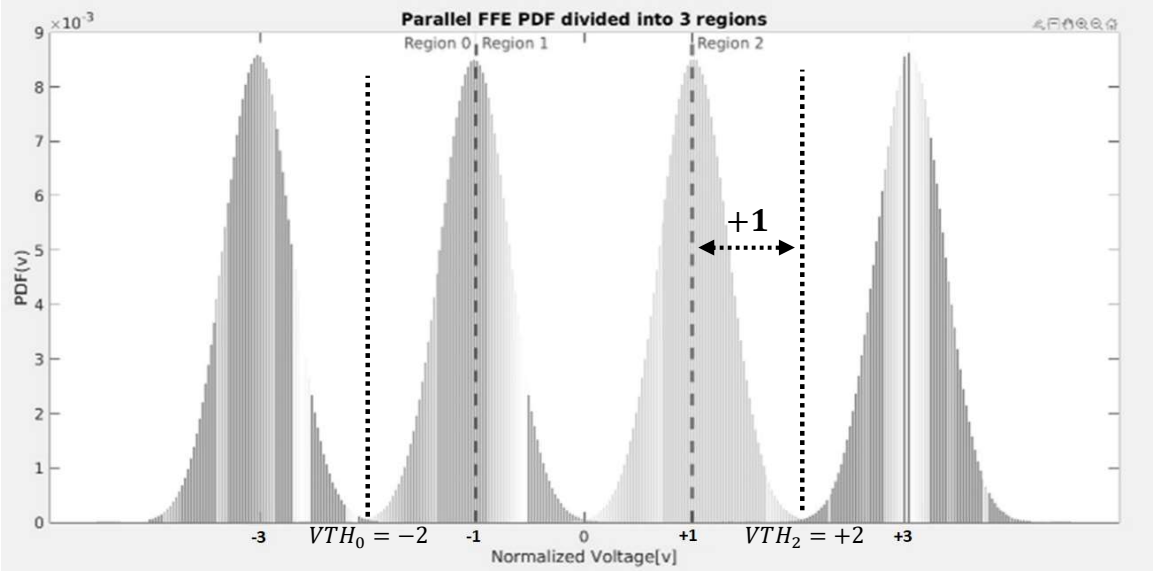

Figure 7: Parallel FFE PDF divided into 3 regions

A slightly modified PUDFE was integrated into the PCIe transceiver used in present work. The CDR digital loop FFE and slicer were reused as the PUDFE parallel FFE and slicer to save area and power. The CDR FFE constraints didn't change the PDF and BER significantly, hence previous statistical assumptions were still valid. On top of the elimination of the 2 unlikely choices for $k-1$ symbol, the CDR slicer output is also

used to select the threshold for current equalized signal candidates ($y_{i,k}$). This is done based on the same conclusion drawn from the parallel FFE PDF above. The selected threshold is the one in the middle of the region in which the parallel FFE output resides in. This change reduces the number of slicers required by a factor of 3. Simulations and lab measurements indicated that this modification and overall PUDFE architecture does not limit the performance at least until the channel loss exceeds 36 dB. Bathtub curve measurements guarantee that for much higher bit error rates than what exhibit in 36 dB channel. The third modification is the definition of region minimum (rm_k) as a marker for the region's lower symbol (Assigned values are $0/\ 1/\ 2$). This allows the output of the PUDFE slicers to be decoded into a single bit and the PUDFE look ahead mux (LAMUX) to be composed of single 2to1 multiplexers as in NRZ signaling (Instead of 2X 2to1 multiplexers for 2 bits candidate in [8]). The resulting simplified PUDFE slicer inputs and outputs values are defined by Eq (17) and Eq (18):

Eq (17)

$$y_{i,k} = u_k - \alpha * (2 * (rm_k + i) - 3)$$

Eq (18)

$$a_{i,k} = \begin{cases} 0 \ if \ y_{i,k} < V_{TH<rm_k>} \\ 1 \qquad\qquad o.w. \end{cases}$$

The output of the LAMUX, a_k, is summed with rm_k to construct the PUDFE output $\hat{v}_k$.

ML assisted Speculative Error Correction

1. MLSE simplification

The proposed SEC block is applying a simplified maximum likelihood algorithm on erasure symbols within the loop unrolling logic of the DFE. Thus, any potential error on that symbol can be corrected before propagating and creating an error burst. By fixing the error in previous symbol $k-1$, e_{k-1} will be driven to 0 in Eq (4) and the current symbol's slicer's input will be equal to:

Eq (19)

$$y_k = v_k + w_k$$

Which is a normal distribution around v_k ,identical to v_{k-1}. The probability of a burst is no longer dependent on the previous symbol and is significantly lower than the target BER, as w_k and w_{k-1} are independetly distributed.

The ML detector simplification process is illustrated on a trellis diagram shown in Figure 8a to c.

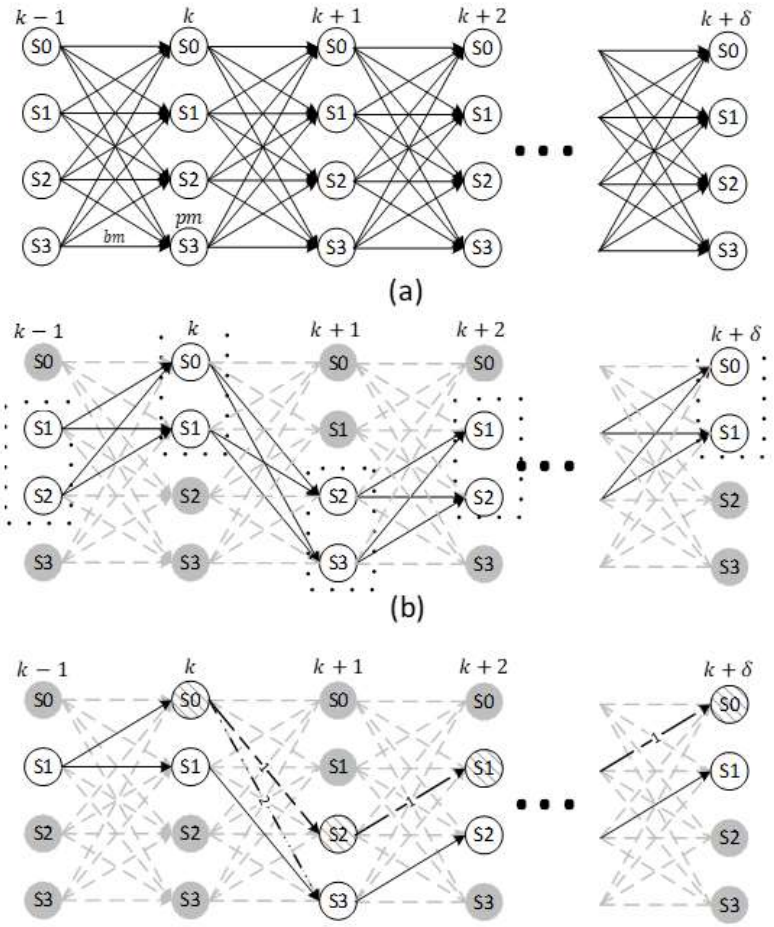

Figure 8: MLSE simplification in the SEC

Figure 8a depicts a full trellis, with synchronization (not shown) and traceback $\delta \geq 5$. In Figure 8b, the number of states is reduced from 4 to 2 and the number of branches from 16 to 4, by adopting the usage of the parallel FFE soft information as shown in the PUDFE architecture. The assumption of a single symbol DFE error at time k (with or without burst following it), is applied in Figure 8c, reducing the number of likely paths to 2. As the previous DFE decision is assumed to be correct, there is no need for synchronization. The two remaining paths are the result of the DFE equalization applied on symbols $k \rightarrow k + \delta$, given the two possible options of the erasure symbol at time k. Figure 8c illustrates two examples (marked '1' and '2') for the path passing through the state corresponding with the alternative DFE output at time k. Path '1' converges with the DFE output at time k+1 and path '2' doesn't converge after δ symbols. Only one of these two, or a different convergence pattern (after $1 < b < \delta$ states) will occur for the path passing through the alternative DFE output at time k .

Assuming these simplifications, the likelihood criteria $Vsum$ for the symbol at time k becomes:

Eq (20)

$$Vsum = \sum_{l=0}^{\delta} \left(bm_1(s_{k+l-1}^1, s_{k+l}^1, k) - bm_0(s_{k+l-1}^0, s_{k+l}^0, k) \right)$$

Where s_{k+l}^0, s_{k+l}^1 are the states at time $k+l; l = 0..\delta$ for paths 0,1 respectively and $s_{k-1}^1 = s_{k-1}^0$. Path 0 (Upper index) relates to the path passing through the PUDFE slicer output of the erasure symbol at time k, and 1 through the alternative for it within the time k PUDFE region. A negative $Vsum$ indicates that path 1 is more likely than path 0 hence the candidate at time k should be inverted (Inverting this value results in the sliced value on the other side of the PUDFE region, which is the optional error corrected sliced value for this symbol). This criterion is equal to the one depicted in Figure 5 for the DDFE.

The error types, probabilities, and tradeoffs of the DDFE (Table 1) are applicable for this simplification as well. Error type I is illustrated in Figure 9, where the correct path passes through S0 at time k, but it is marked as erasure because of large noise. The two sequences considered differ by one symbol. On time $k+1$ a real error occurred. This results in sub optimum ML detection as the most likely path is not considered (broken line). Although such scenario can lead to a false detection of an error in symbol k below, the SEC is executing independent check on all symbols, hence it can correct the subsequent error at time $k+1$. This was not possible in the DDFE implementation in which the threshold skew is turned off during the erasure period.

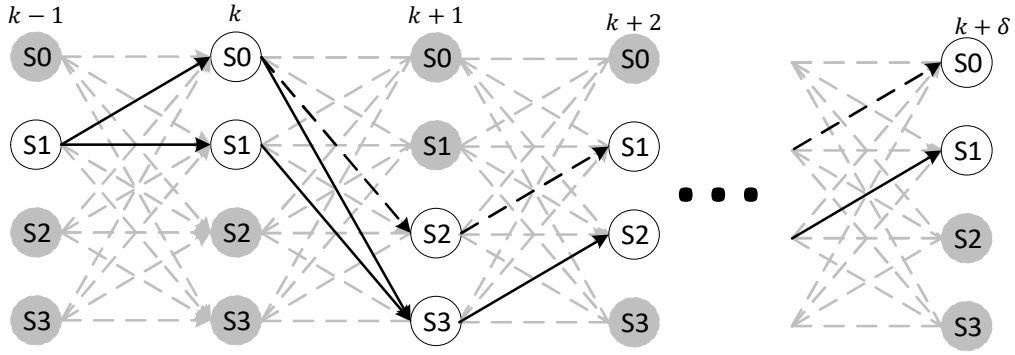

Figure 9: Simplified MLSE error scenario

2. The SEC algorithm

The baud rate SEC operation is presented in Table 2.

Table 2: SEC algorithm

Algorithm 1: Speculative error correction (SEC) for Partially Unrolled DFE at time $\boldsymbol{k_0}$

Inputs: unrolling slicer inputs at time k_0: $y_{i,k_0}, i \in [0,1]$
unrolling slicer outputs (candidates) at times $[k_0, k_0+\delta]$: $a_{i,k_0 \to k_0+\delta}$
PUDFE regions minimum at times $[k_0, k_0+\delta]$: $\mathrm{rm}_{k_0 \to k_0+\delta}$
FFE outputs at times $[k_0, k_0+\delta]$: $\mathrm{u}_{k_0 \to k_0+\delta}$
PAM4 slicer thresholds: $vth_j, j \in [0,2]$
Output: candidate inversion (correction) signal at time k_0: c_{i,k_0}

for $\boldsymbol{i}$=0 to 1 **do** *// Loop through the two candidates*
Initializations
$\mathrm{m}_{\mathrm{i,k_0}} = 0$
$\mathrm{c}_{\mathrm{i,k_0}} = 0$
$\mathrm{v}_{\mathrm{sum}} = 0$
if $\left|\mathrm{y}_{\mathrm{i,k_0}} - \mathrm{vth}_{\mathrm{rm_{k_0}}}\right| < \epsilon$ **do** *// Erasure zone marking*
$m_{i,k_0} = 1$
end if
if $\mathrm{m}_{\mathrm{i,k_0}} == 1$ **do**
// Create two sequences of length $\delta + 2$: $q^0_{l=0..\delta+1}$, $q^1_{l=0..\delta+1}$
$\mathrm{q}^0_{\mathrm{l=0}}$=i *// Initialize the previous symbol sliced value for both sequences 0 location.*
$\mathrm{q}^1_{\mathrm{l=0}}$=i
$\mathrm{q}^0_{\mathrm{l=1}} = \mathrm{a}_{\mathrm{i,k_0}}$ *// Assign the two options for the candidate i at time* k_0
$\mathrm{q}^1_{\mathrm{l=1}} = \sim \mathrm{a}_{\mathrm{i,k_0}}$
// Apply simplified ML detection.
for l=1 to δ+1 **do**
$\mathrm{bm}_{0,\mathrm{l}} = \left(\mathrm{u}_{0,\mathrm{k_0+l-1}} - (\mathrm{rm}_{\mathrm{i,k_0+l-1}} + \mathrm{q}^0_{\mathrm{l}}) - \alpha * (\mathrm{rm}_{\mathrm{i,k_0+l}} + \mathrm{q}^0_{\mathrm{l-1}})\right)^2$
$\mathrm{bm}_{1,\mathrm{l}} = \left(\mathrm{u}_{1,\mathrm{k_0+l-1}} - (\mathrm{rm}_{\mathrm{i,k_0+l-1}} + \mathrm{q}^1_{\mathrm{l}}) - \alpha * (\mathrm{rm}_{\mathrm{i,k_0+l}} + \mathrm{q}^1_{\mathrm{l-1}})\right)^2$
$\mathrm{Vsum} = \mathrm{Vsum} + \mathrm{bm}_{1,\mathrm{l}} - \mathrm{bm}_{0,\mathrm{l}}$
if $\mathrm{l} \le \delta$ **do** *// Apply DFE mux to extract next symbol sliced values*
$\mathrm{q}^0_{\mathrm{l+1}} = \mathrm{a}_{\mathrm{i=q^0_l,l+1}}$
$\mathrm{q}^1_{\mathrm{l+1}} = \mathrm{a}_{\mathrm{i=q^1_l,l+1}}$
end if
end for
if $\mathrm{Vsum} < 0$ **do**
$c_{i,k} = 1$
end if
end if
end for

It starts by inspecting whether either PUDFE candidates at time $k = k_0$ are within the erasure zone. Applicable candidates are marked by setting $m_{i,k}$ to 1. This portion of the algorithm can be efficiently implemented within the slicer (as detailed in the implementation section hereunder). The proposed simplified ML detector is executed on the marked candidates. First, two binary sequences of length $\delta + 2$ are defined, $\{q_l^0\}$ and $\{q_l^1\}$. $q_{l=0}^0$, $q_{l=0}^1$ are initialized with the $k_0 - 1$ PUDFE candidate, i (The index of a loop unrolling candidate at time k_0 is matching the previous sliced value that was used to calculate them). $q_{l=1}^0$, $q_{l=1}^1$ are assigned with the marked candidate at time k_0, and its inversion, respectively.

The rest of the processing is done iteratively. Each iteration is calculating the branch metrics, $bm_{0,l}$ and $bm_{1,l}$, to traverse from previous state to current state for each of the sequence 0 and 1 respectively. $bm_{0,l}$ is summed with negative sign and $bm_{1,l}$ with positive sign into the variable $Vsum$. The iteration ends with a DFE mux which selects the next sliced value for each sequence – $a_{0,l+1}$ or $a_{1,l+1}$, using current iteration sequence values – q_l^0 and q_l^1.

The two paths path metrics summation is completed after δ iterations. At this point $Vsum$ sign is inspected, and if negative the conclusion is that the path passing through the error corrected value is the more likely one. In such a scenario the candidate inversion signal c_{i,k_0} is set to 1.

Figure 10 illustrates how the above SEC algorithm is to be integrated within the PUDFE. Although the algorithm is applied on each UI/ candidate and may inverse many of them, most of the inversions are applied on candidates that will eventually not be selected by the DFE mux. For this reason, it is considered 'speculative'.

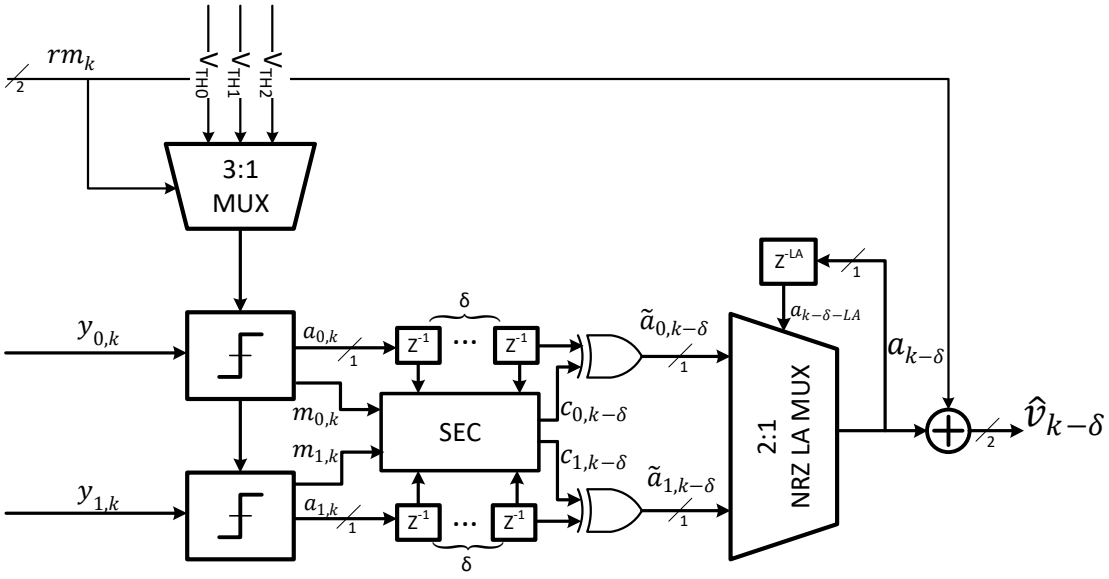

Figure 10: PUDFE loop unrolling with embedded speculative error correction block.

3. Simulation results

The SEC algorithm verification was done by adding a SEC-PUDFE integration as a 3rd equalization option in the Mathworks Simulink [14] system model of a 64Gbps PCIe Gen6 SerDes. A 36dB channel and α ranging from 0.35 to 0.7 (Combined response of channel, AFE and FFE) were used in the simulations. The resulting model is depicted in Figure 11.

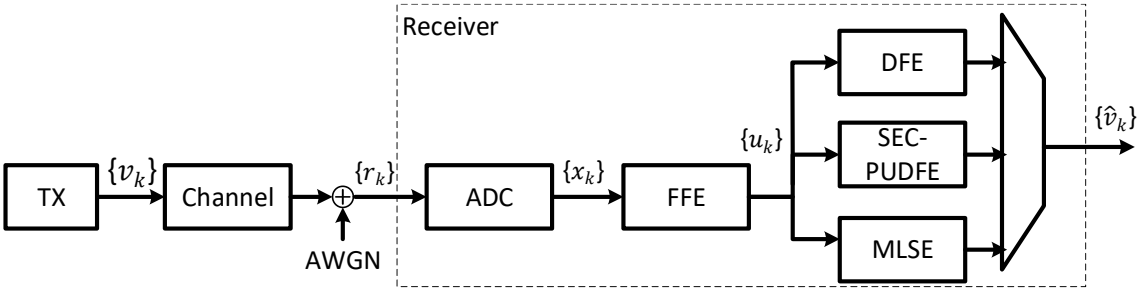

Figure 11: Gen6 PCIe SerDes link model with SEC-PUDFE block

First the statistical behavior of the residual SEC errors was validated using the analysis done for the DDFE (Table 1). This was achieved by simulating the dependency of the different error types in ε and δ. The SNR at the slicer input was 18.8dB with α =0.6, resulting in Symbol Error Rate (SER) of ~1e-5 at the MLSE output. As can be seen in Figure 12 (a) and (b), most of the errors are type II and converge to the MLSE error floor (147), when δ approaches the depth of convergence. Type I and III have opposite dependency on ε and reach minimum for ε=0.3. Type I has slight dependency on δ. These findings align with the DDFE error probability dependencies. Type I is significantly improved compared with the NRZ DDFE as the erasure zone checks for future UIs, within the detection period of a marked UI, are not disabled. The SEC total error count reaches a minimum (150) with δ=4 and ε=0.3.

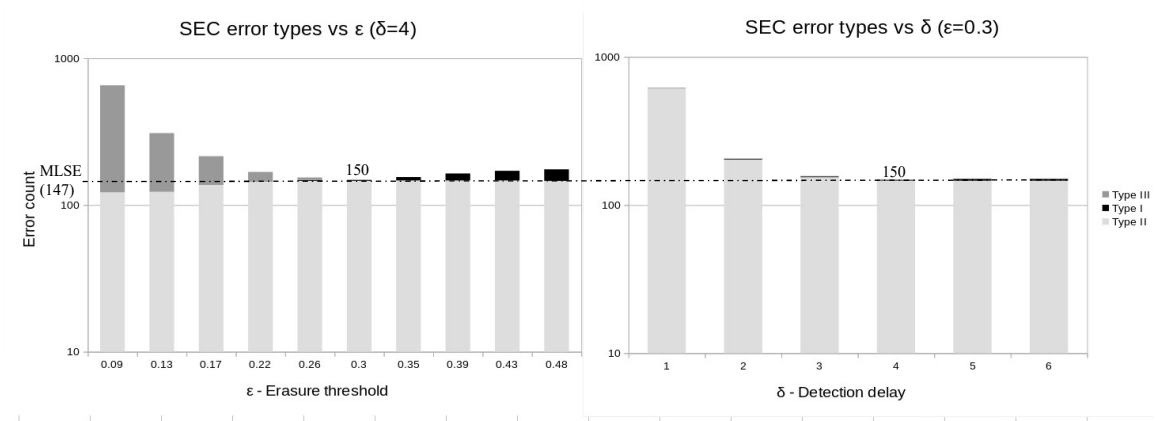

Figure 12: SEC error types as function of (a) ε and (b) δ

Next the various equalization schemes were simulated with increasing noise levels (Figure 13). Each SER datapoint was extracted from a >100M symbols simulation. As can be seen, there is a negligible difference between the SEC-PUDFE and the MLSE (≤0.03dB). The MLSE and SEC exhibit 1.3dB, 1.27dB of SNR gain for SER=1e-6 (PCIe

PHY target BER) respectively. The SEC-PUDFE is providing 15x to >100x SER improvement over the conventional DFE.

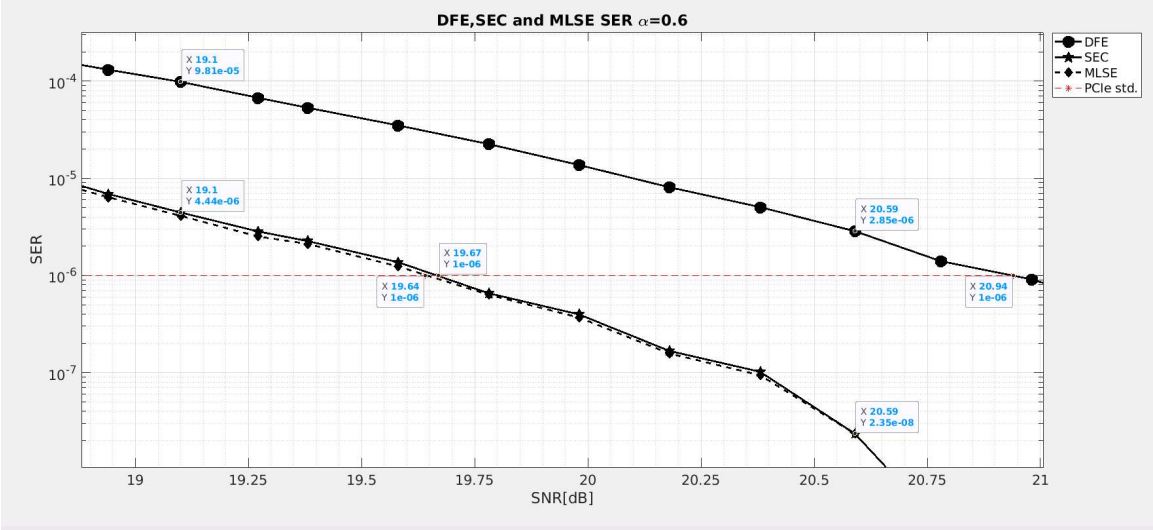

Figure 13: SER vs SNR DFE, SEC-PUDFE and MLSE (α=0.6)

4. SEC circuit implementation

The baud rate SEC algorithm in Table 2 was implemented within a pipelined parallel DSP. It enables concurrent SEC operation on M UIs per cycle. The notation to be used for parallel processing hereunder is:

Eq (21)

$$\{u_k\} \rightarrow \{u_{n,t}\} \rightarrow \{u_{n,0}, u_{n,1}, \dots, u_{n,M-1}\}$$

Where $k \triangleq n * M + t,\ t = 0..M-1$. For simplification, the cycle index n is omitted from signal names.

The algorithm starts with the marking of the erasure candidates by setting $m_{i,t}$ to 1, when:

Eq (22)

$$\left|y_{i,t} - vth_{\mathrm{rm}_t}\right| < \epsilon$$

Power and area optimization is achieved by reusing the slicer for this purpose. Note that the algorithm allows two markings $m_{[0,1],t}$ for each symbol corresponding to the two PUDFE candidates $a_{[0,1],t}$. However, the hardware implementation is optimized for a single SEC erasure candidate per symbol, imposing the following constraint:

Eq (23)

$$\alpha + \varepsilon \leq 1$$

When this constraint is maintained, if one PUDFE candidate slicer input is within erasure zone, the other one which is $2 * \alpha$ away cannot be. Hence, $m_{i,t}$ is assumed to be at most one hot encoded. The parallel implementation of the SEC algorithm is illustrated in Figure. It is processing M PUDFE candidate pairs each cycle.

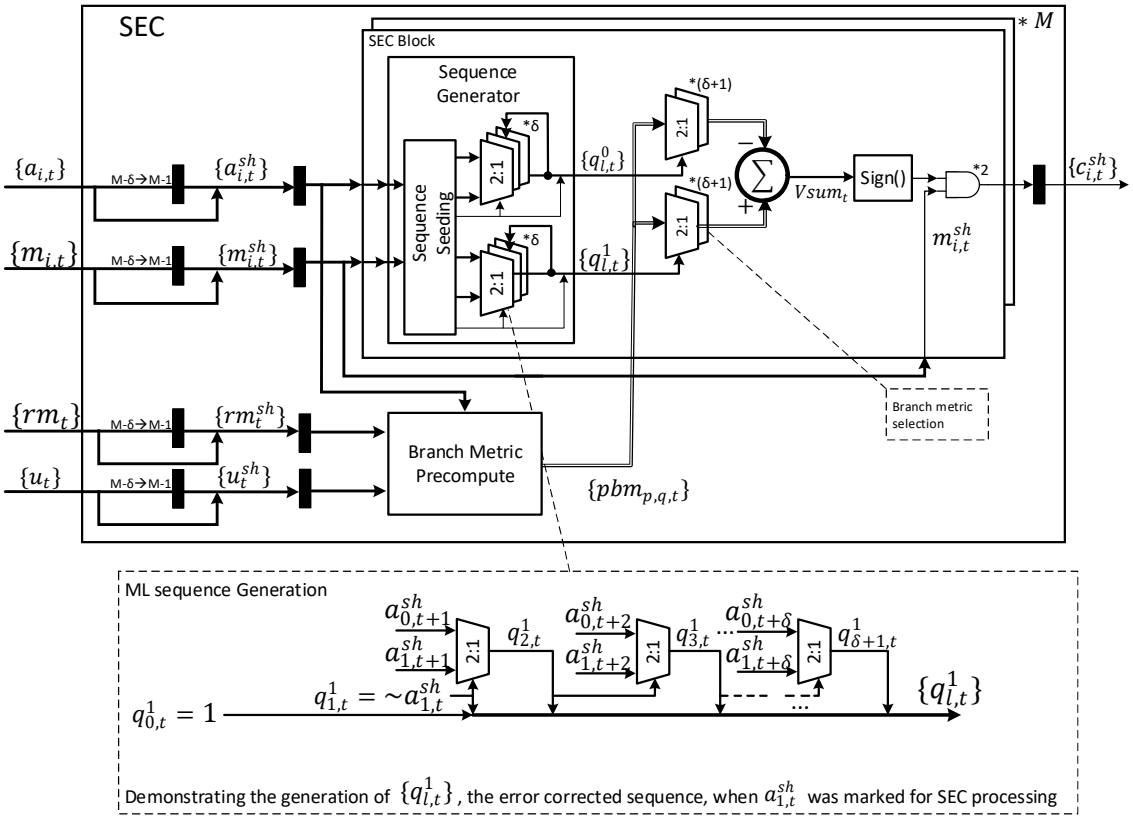

Figure 14: Parallel SEC circuit implementation

To facilitate the SEC processing on cycle end symbols $[M-\delta, M-1]$, all required SEC input busses are shifted by δ symbols and prefixed with the previous cycle history as follows:

Eq (24)

$$\{u_t^{sh}\} = \{u_{n-1,M-\delta-1}, \dots, u_{n-1,M-1}, u_{n,0}, u_{n,1}, \dots, u_{n,M-1}\}$$

Similar processing is applied to generate $\{a_{i,t}^{sh}\}, \{rm_t^{sh}\}$ and $\{m_{i,t}^{sh}\}$. The new bus width is $M+\delta$, but only the first M symbols are processed through the SEC each cycle.

To ease the critical path timing at the expense of slight increase in power consumption and area, the following two operations are done is parallel: (1) The two sequences to be compared for likelihood, $\{q_{l,t}^0\}$ and $\{q_{l,t}^1\}$, are constructed by the sequence generator , and (2) the branch metrics $pbm_{p,q,t}$ are precomputed.

The sequence generation starts with identifying which candidate requires the ML detection, (indexed by 'I'):

Eq (25)

$$I = \begin{cases} 0 \; if \; m_{0,t}^{sh} = 1 \\ 1 \qquad o.w. \end{cases}$$

Location 0 in both sequences is seeded with the previous UI symbol:

Eq (26)

$$q_{0,t}^{0} = q_{0,t}^{1} = I$$

Location 1 in original candidate sequence and alternative candidate sequence are seeded with the original candidate and its inversion respectively:

Eq (27)

$$q_{1,t}^{0} = a_{I,t}^{sh}$$
$$q_{1,t}^{1} = \sim a_{I,t}^{sh}$$

The remaining symbols in the sequences are selected using loop unrolling DFE multiplexer. The branch metrics $pbm_{p,q,t}$ are precomputed for all possible PUDFE state transitions from previous candidate to current:

Eq (28)

for t=0 to M-1 **do**
for p=0 to 1 **do**
for q=0 to 1 **do**

$$\mathrm{pbm_{p,q,t}} = \mathrm{u_t^{sh}} - \mathrm{h_1} * \left(\mathrm{rm_t^{sh}} + \mathrm{p}\right) - \mathrm{h_0} * \left(\mathrm{rm_t^{sh}} + \mathrm{q}\right)$$

end for
end for
end for

Next, the branch metrics for each sequence $\{bm_{0,l,t}\}$ and $\{bm_{1,l,t}\}$ are selected by each entry in the sequences $\{q_{l,t}^{0}\}$ and $\{q_{l,t}^{1}\}$ as follows.

Eq (29)

$$bm_{i,l,t} = pbm_{q_{l,t}^{0}, q_{l+1,t}^{0}, t}$$

As mentioned in Table 2, the selected branch metrics are summed up with negative and positive sign respectively, into $Vsum_t$. The correction enable signal $c_{i,t}$ is assigned with the sign of $Vsum_t$ and validated with the candidate's erasure marker**.**

The latency of the above implementation is not dependent on the block size – it only depends on the speed of the technology. As verified in synthesis, for $\delta = 4$ and M=32 it takes 1ns to complete the SEC processing in 5nm FinFET technology node, hence:

Eq (30)

$$L_{SEC} = 1 + \frac{\delta}{M} \xrightarrow[M=3]{\delta=4} = 1.13$$

Where $\frac{\delta}{M}$ was introduced by the shift discussed above. Using the same metric to measure the complexity as in Eq (12), given that each SEC block requires $2 * (\delta + 1)$ 2-input adders to calculate $Vsum$, the total amount of adders required to implement the SEC is:

Eq (31)

$$C_{SEC} = \mathrm{M} * 2 * (\delta + 1) \xrightarrow[M=32]{\delta=4} = 320$$

Table 3 compares this implementation to the Viterbi decoder implementation methods described previously.

Table 3: Area, latency and performance comparison of the SEC and the referenced MLSE implementation methods (M=32)

	Look ahead VD*	Layered look ahead VD**	**This work: SEC with δ=4**	Fully unrolled DFE
Technology	-	-	**5nm**	5nm
Data Rate	64Gbps	64Gbps	**64Gbps**	64Gbps
Modulation	PAM-4	PAM-4	**PAM-4**	PAM-4
Parallel Block Size (M)	32	32	**32**	32
Clock Frequency [GHz]	-	-	**1**	-
Added latency [1GHz cycles]	33	6	**1.13 (19%)**	-
Complexity metric [2 input adders count]* **	3500	3500	**320 (9%)**	-
Synthesis gate area , 5nm FinFET technology [μm^2]	-	-	**4.5K (SEC) 10K (PU-SEC)**	4.5K
SNR [dB] @ SER=1e-6	19.64	19.64	**19.67**	20.94

* As depicted in Figure 3. Latency and complexity calculated with Eq (9) and Eq (10) respectively.
** As depicted in Figure 4. Latency and complexity calculated with Eq (11) and Eq (12) respectively.
*** The complexity metric used in this paper and in [12] is the number of 2 input adders needed to implement the ACS unit of the VD

SEC validation in silicon: Horizontal SER Bathtub measurement

The SEC was validated in silicon using a PCIe Gen6 (64Gbps) wireline transceiver test-chip fabricated in 5nm FinFET technology. The overall block diagram of the system is shown in Figure 15. In the receiver, a Continuous Time Linear Equalizer (CTLE) equalizes the incoming signal, and a Variable Gain Amplifier (VGA) adjusts the signal power to the ADC full scale. A 32 GS/s time interleaved ADC with programmable full scale digitizes the VGA output and the data is then processed in the DSP. The time interleaved ADC consists of two stages of track and hold followed by 32 asynchronous 7-bit Successive-Approximation (SAR). The first sampling stage is clocked by 4X1UI pulses generated from the quadrature clock coming from the internal Phase-locked loop (PLL, not shown below).

The DSP is processing 32 ADC samples every 1GHz cycle. The data path consists of a 16-tap FFE and a 1-tap DFE. The DFE is implemented in two ways, fully unrolled (DFE) and partially unrolled (PUDFE). Each of them can be selected to drive the output bus. In the timing recovery path (DTL), a 7-tap FFE and slicer are followed by baud-rate phase detector which controls a Phase Interpolator (PI). The PI adjusts the quadrature sampling

clock phase and frequency, to compensate for the frequency difference between the received data and the internal PLL.

The SEC was not implemented in this test-chip. To test it, a replica of the receiver DSP was implemented in python (pDSP in the diagram below), and driven with samples from the on-chip ADC memory buffer. The Receiver LMS engine was used to train the pDSP coefficients.

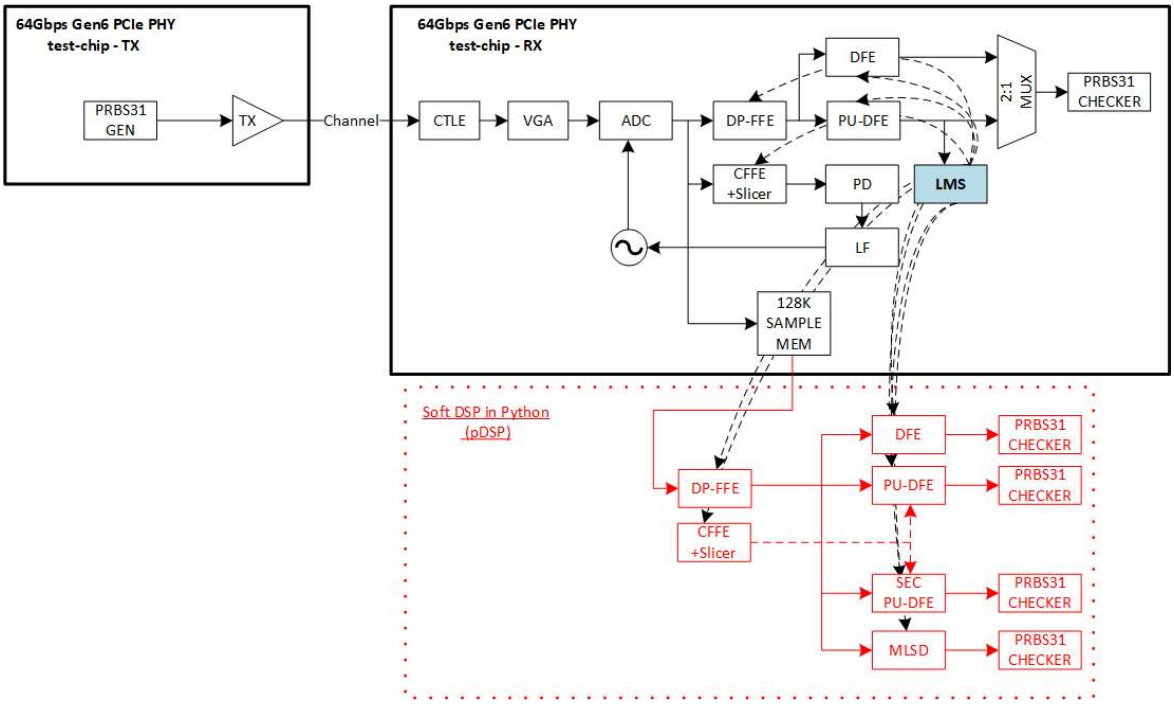

Figure 15: 64Gbps Gen6 PCIe PHY and offline processing logic

The measurement setup is depicted in Figure 16. The evaluation board is connected to a 31" channel on an ISI board. The total loss including the channel, connectors and package was measured to be 36dB at Nyquist.

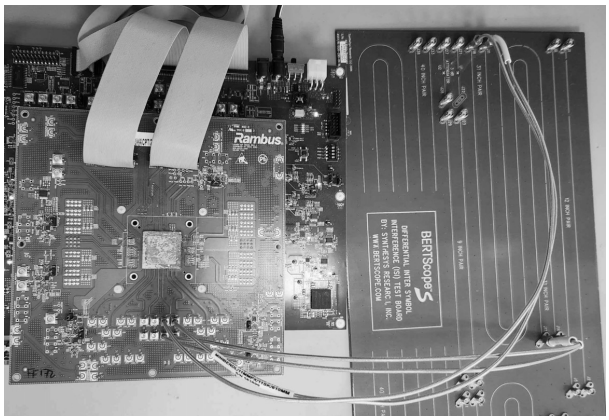

Figure 16: SEC SER measurement setup

The ADC samples were read out using a slow serial interface. To reduce the run time to 10 hours or less per datapoint, the measured MLSE/SEC SER was limited to

values higher than 1e-7. Since the lock point SER with a 36dB channel is lower than that, the SEC performance was validated on the edges of a SER bathtub curve. The bathtub measurement procedure is outlined in Table 4.

Table 4: SER Bathtub measurement procedure

SER Bathtub measurement procedure

1. Bring-up the PHY to mission mode, CDR locked to eye center**.**
2. Freeze the CDR
3. Decrement the PI code until timing loop SER = 1e-2
4. Run the below routine in the pDSP for each PI code until MLSE SER reaches below $\mathbf{1e-7}$:

Initialization

$\text{mlse\_bit\_errors} = 0;\ \text{sec\_bit\_errors} = 0;\ \text{dfe\_bit\_errors} = 0;$
$\text{iter\_count} = 0;\text{dir} = +1$
while $\text{iter\_count} < 2000$ and $\text{mlse\_err\_count} < 20$ do
$\text{ADC\_BUF} = \text{get\_chip\_samples}()$
$\text{pDSP\_COEFF} = \text{get\_rx\_coeff}()$
$\text{FFE\_BUF} = \text{ffe\_eq}(\text{ADC\_BUF}, \text{pDSP\_COEFF})$
$\text{RM\_BUF} = \text{get\_region\_min}(\text{cffe\_eq}(\text{ADC\_BUF}, \text{pDSP\_COEFF}))$ //Region Min
$\text{dfe\_bit\_errors} = \text{dfe\_bit\_errors} + \text{check\_prbs}(\text{dfe\_eq}(\text{FFE\_BUF}, \text{h0}, \text{h1}))$
$\text{sec\_bit\_errors} = \text{sec\_bit\_errors} + \text{check\_prbs}(\text{sec\_pudfe eq}(\text{FFE\_BUF}, \text{RM\_BUF}, \text{h0}, \text{h1}))$
$\text{mlse\_err} = \text{mlse\_err} + \text{check\_prbs}(\text{mlse\_eq}(\text{FFE\_BUF}, \text{h0}, \text{h1}))$
$\text{pi\_code} = \text{pi\_code} + \text{dir}$
$\text{iter\_count} = \text{iter\_count} + 1$
end while

5. Increment the PI code until timing loop SER = 1e-2, store intermediate DFE SER values
6. Run the routine in (4) with $\boldsymbol{dir = -1}$

The measured SER Bathtub is depicted in Figure 17. The SEC-PUDFE is providing 15x to 60x SER improvement over the conventional DFE when SER <*1e-4*. The number of errors at the SEC-DFE output are within $\pm 5\%$ compared with the MLSE.

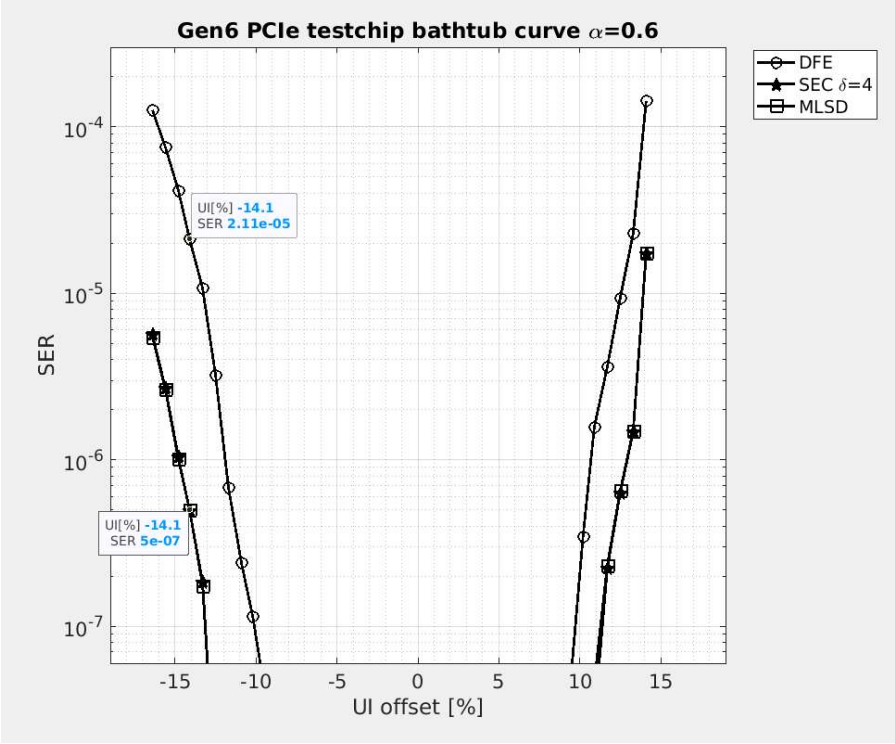

Figure 17: DFE/SEC/MLSE 36dB channel bathtub

Conclusions

The SEC circuit presented in this paper, is an improvement to partially unrolled DFE resulting in SER that approaches that of full MLSE implementation. It addresses the main culprit of the DFE, the error propagation phenomena. A simplified and very short maximum likelihood engine is applied on loop unrolling candidates that exhibits high noise levels. If it is more likely that a candidate has an error, the engine will correct it. By correcting most of the burst triggering random errors, the associated bursts are eliminated. For block size of 32 samples, the added latency of the circuit is ~1ns, and the synthesized area is 4.5K μm^2 in 5nm FinFet technology. These values are much lower than those required for full MLSE. The SEC was verified in simulation and validated by post processing ADC samples read from a PCIe Gen6 (64Gbps) wireline transceiver test-chip fabricated in 5nm FinFet technology.

Acknowledgements

The authors would like to thank Marc Loinaz and the Cadence HPP group for promoting innovation and supporting this research activity. Special thanks to the Toronto site team members who enabled and executed the offline validation of the SEC.

References

[1] J. Q. Wang, A. Tan, A. Iyer, A. Fan, A. Farhoodfar, B. Alnabulsi, B. Smith, C. Loi, C. R. Ho, D. Cartina, J. Riani, J. Casanova, , K. Raviprakash, L. Patra, L. Wang, M. Bachu, S. Ray, S. Chong, S. Dallaire, T. Nguyen, T.-F. Wu, V. Giridharan, V. Gurumoorthy, X. Ding, Y. Yin, Z. Sun, S. Jantzi and L. Tse, "A 2.69pJ/b 212Gb/s DSP-Based PAM-4 Transceiver for Optical Direct-Detect Application in 5nm FinFET", ISSCC 2024

[2] D. Pfaff, M. Nummer, N. Hai, P. Xia, K. G. Yang, M. M. Mohsenpour, M. A. LaCroix, B. Zamanlooy, T. Eeckelaert, D. Petrov, M. Haroun, C. Dick, A. Zaman, H. Mei, S. Moazzeni, T. Shakir, C. Carvalho, H. Huang, P. Kumari, R. Mason, F. Brishty and I. Jaffri , "A 224Gb/s 3pJ/b 40dB Insertion Loss Transceiver in 3nm FinFET CMOS", ISSCC 2024

[3] M. E. Meybodi , H. Gomez , Y. C. Lu, H. Shakiba and A. Sheikholeslami, "Design and Implementation of an On-Demand Maximum-Likelihood Sequence Estimation (MLSE)" , IEEE Open Journal of Circuits and Systems, Volume: 3, May 2022

[4] G. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," in IEEE Transactions on Information Theory, vol. 18, no. 3, pp. 363-378, May 1972

[5] H. Yueksel , M. Braendli, A. Burg, G. Cherubini, , R. D. Cideciyan, , P. A. Francese, , S. Furrer, M. Kossel, L. Kull, D. Luu, C. Menolfi, T. Morf and T. Toifl, "Design Techniques for High-Speed Multi-Level Viterbi Detectors and Trellis-Coded-Modulation Decoders", IEEE transactions on circuits and systems, Volume: 65, Issue: 10, October 2018

[6] R. Gitlin and E. Ho, "A Null-Zone Decision Feedback Equalizer Incorporating Maximum Likelihood Bit Detection", IEEE Transactions on Communications , Volume: 23, Issue: 11, November 1975

[7] J.W.M. Bergmans, J.O. Voorman and H.W. Wong-Lam, "Dual decision feedback equalizer", IEEE Transactions on Communications , Volume: 45, Issue: 5, May 1997

[8] S. Kiran, S. Cai, Y. Luo, S. Hoyos, and S. Palermo , "A 32 Gb/s ADC-Based PAM-4 Receiver with 2-bit/Stage SAR ADC and Partially-Unrolled DFE" , CICC 2018

[9] T. Xu , Z. Li, J. Peng, A. Tan, Y. Song, Y. Li, J. Chen and M. Wang., "Decoding of 10-G Optics-Based 50-Gb/s PAM-4 Signal Using Simplified MLSE," in IEEE Photonics Journal, vol. 10, no. 4, pp. 1-8, Aug. 2018

[10] G. C. Clark and J. B. Cain, Error-Correction Coding for Digital Communications. New York: Plenum, 1981, pp. 227–264.

[11] C. Xu, M. Lai, F. Lv, L. Xiao, Z. Luo, X. Qi and Y. Ou , "A low BER adaptive sequence detection method for highspeed NRZ data transmission", proceedings of ICICM 2022, pp. 692–697

[12] J. J. Kon, and K. K. Parhi, "Low-Latency Architectures for High Throughput Rate Viterbi Decoders", IEEE Trans. on VLSI Systems, Volume: 12, Issue: 6, June-2004.

[13] S. Kiran, S. Cai, Y. Luo, S. Hoyos, and S. Palermo, "A 52-Gb/s ADC-Based PAM-4 Receiver With Comparator-Assisted 2-bit/Stage SAR ADC and Partially Unrolled DFE in 65-nm CMOS", IEEE Journal of Solid-State Circuits, Volume: 54, Issue: 3, March 2019

[14] MATLAB and Simulink are registered trademarks of The MathWorks, Inc