An Exact Linear Time Algorithm for Computing
Worst Case Eye Diagrams for a Class of Non-Linear
High-Speed Signaling Systems

Aadithya V. Karthik, Sayak Ray, Robert Brayton, and Jaieet Roychowdhury
The Department of Electrical Engineering and Computer Sciences, The University of California, Berkeley, CA, USA

Abstract—We present a new computational technique for determining worst case (WC) eye diagrams in non-linear high-speed communications sub-systems – an important unsolved problem that frequently arises in Signal Integrity (SI) applications. Existing approaches to the problem, such as Peak Distortion Analysis (PDA), work only for linear and time-invariant (LTI) systems, and are incapable of handling correlated bits on the transmit side. By contrast, our approach is fully general, and capable of handling arbitrary bit correlations. Our technique works by expressing bit correlations using a Finite State Machine (FSM), and by reducing the WC eye diagram computation to a combinatorial optimization problem on the FSM, which is then solved efficiently using dynamic programming. Our solution only takes time linear in the size of the FSM, making it suitable for the analysis of cutting-edge high-speed signaling systems. Furthermore, our approach yields exact and provably accurate WC eye diagrams that are neither overly pessimistic (unlike PDA) nor unduly optimistic (unlike many existing simulation-based approaches). In this paper, we apply our method to analyze and produce WC eye diagrams for a (7,4)-Hamming encoded communications scheme, as well as an 8b/10b SERDES encoded system.

I. INTRODUCTION

High-speed signaling/communications modules are becoming increasingly prominent in today’s cutting-edge mobile chipsets, SoCs, and other high-performance hardware – including supercomputers, network components and switches used in data warehouses, high-frequency trading applications, etc. This is because such components now play a significant role in determining key system-level performance metrics, including speed, power consumption, reliability, etc. [1]–[4].

Indeed, over the past decade, the performance of high-speed signaling systems has improved dramatically – from a few Mb/s to multi-Gb/s throughput. This is largely due to the widespread adoption of increasingly sophisticated and innovative design/optimization techniques, such as transmit pre-emphasis/de-emphasis methods, new coding and modulation schemes, advanced equalization techniques to compensate for inter-symbol interference (ISI), improved PLLs, DLLs, and clock and data recovery (CDR) solutions, advances in transceiver/link design with improved noise/jitter mitigation, etc. [1]–[6].

However, with the advent of the above techniques, and the increased component-level and system-level complexity that they brought about, the modelling and analysis of high-speed signaling systems has become an extremely challenging problem [1]–[3].

For example, consider the problem of carrying out an end-to-end worst case (WC) eye diagram analysis of a modern high-speed signaling system – a problem that frequently arises in Signal Integrity (SI) applications [1], [2]. Given a mathematical model for each component in the system (e.g., a SPICE-level model, a reduced-order continuous model [7], or a Boolean model at a higher level of abstraction [8]), the purpose of this analysis is to predict the worst case bit patterns (and hence eye diagram shapes) at the receiver. This in turn yields a quick intuitive picture of end-to-end system-level performance – with important implications for noise margins and end-to-end BER analysis, choice of encoder/decoder architecture, design-space exploration for the receiver’s front-end, etc.

The main computational technique available today for WC eye diagram calculation is called Peak Distortion Analysis (PDA) [1], [3], [9]. As Fig. 1 shows, given a Linear and Time-invariant (LTI) analog channel, PDA can be used to quickly compute the WC eye diagram at the output of the channel, assuming that the input bits are all uncorrelated (i.e., every bit sequence is possible at the input).

Unfortunately, PDA suffers from some serious limitations. For example, the assumption that the input bits are uncorrelated is rarely true in modern signaling systems [2]. Indeed, virtually any modern technique used for coding/modulation at the transmit side (e.g., 8b/10b encoding, error correcting codes such as the Hamming codes, etc.) is bound to introduce significant bit correlations at the input. Furthermore, these bit correlations are often designed to dramatically improve the eye opening at the output – a fact that is completely ignored by PDA. As a result, for modern signaling systems, the WC eye diagrams predicted by PDA are almost always overly pessimistic. As a result, high-speed link engineers and communications system architect often resort to extensive simulation as the only way to compute WC eye diagrams. This is not only tedious and time-consuming; it also opens up the possibility that important bit patterns and corner cases are missed by the analysis, especially if the system is being designed for very low BERs (e.g., of the order of $10^{-12}$), which is quite typical in practice. This in turn increases design time and debugging cost.

To the best of our knowledge, no efficient computational technique is known for computing WC eye diagrams in the presence of arbitrarily correlated bit sequences at the channel input.

Against this background, we propose a new algorithm that accurately and efficiently computes WC eye diagrams even when the input bits are correlated (and they can be correlated in an arbitrarily complex way).

Fig. 2 shows the utility of our new technique. In our approach, the correlations between input bits are expressed using a Finite State Machine (FSM). This is a very general method for representing bit correlations – virtually any encoder or error correcting scheme that can be implemented digitally (whether using combinational or sequential logic) can be expressed as an FSM (for more details, see §II-B) [10]. Furthermore, the channel can be any arbitrary LTI system, and it can be specified using SPICE netlists, or differential equations, or Fourier/Laplace domain
transfer functions, or measured data, or S-parameters, etc. Thus, our technique is completely general and widely applicable.

It must be noted that the introduction of the FSM above makes the overall system non-linear, which renders standard techniques like PDA inapplicable [2], [9]. Nevertheless, the approach that we propose is able to accurately and efficiently compute the required WC eye diagrams. Our technique works by first reducing the WC eye diagram computation to a combinatorial optimization problem, which is then solved efficiently using a dynamic programming algorithm [11]. The algorithm’s time complexity is only linear in the size of the given FSM, which makes our approach very attractive for practical use in the context of today’s sophisticated signaling systems. Furthermore, our algorithm is exact, i.e., we can mathematically prove that our algorithm indeed returns the WC eye diagram for the given system, without being either overly pessimistic (unlike PDA) or unduly optimistic (like many existing simulation-based approaches).

In this paper, we apply our technique to analyze signaling systems involving two different encoding schemes at the transmitter. The first (§IV-A) is a well-known error coding code, the (7,4)-Hamming code, with the output of the encoder feeding into an RLGC chain (a commonly used model for an I/O link/interconnect). The second system (§IV-B) is an 8b/10b SERDES encoder – representing a very effective and widely used encoding method that finds application in several modern standards, including Gigabit Ethernet, IEEE 1394b, PCI Express, SATA and SAS, USB 3.0, etc. [12]. In both these cases, we demonstrate that PDA makes overly pessimistic WC eye predictions, whereas our algorithm is accurate and exact in its prediction of the WC eye diagrams.

The rest of this paper is organized as follows. In §II, we cover preliminaries such as the standard PDA algorithm, the use of FSMs to represent bit correlations, etc. Then, in §III, we present the core algorithms and techniques behind our approach. We demonstrate our results (on the two systems described above) in §IV, and we conclude in §V.

II. PRELIMINARIES

In this section, we describe some key concepts that are required to understand our core technique and the algorithms covered in §III.

A. PDA for LTI systems driven by uncorrelated bits

As we mentioned in §I, PDA is a computational technique for determining the WC eye diagrams at the output of a Linear and Time-invariant (LTI) channel, when the input to the channel consists of a sequence of uncorrelated bits. We now describe how PDA works.

Given an LTI channel, consider the pulse input \( u(t, T) \) (where \( t \) denotes time and \( T \) is the pulse width) given by:

\[
u(t, T) = \begin{cases} 1 & \text{if } t \in [0, T) \\ 0 & \text{else} \end{cases}
\]

Let \( h(t, T) \) denote the channel’s response to the above input.

Suppose we apply a sequence of bits \( \{b_k\}_{k=-\infty}^{\infty} \) to the above system as input, with each bit being active for a period \( T \). It can be readily seen that the corresponding input waveform \( x_b(t, T) \) is given by:

\[
x_b(t, T) = \sum_{k=-\infty}^{\infty} b_k u(t-kT, T)
\]

The channel being an LTI system, its response \( y_b(t, T) \) to the above input is given by:

\[
y_b(t, T) = \sum_{k=-\infty}^{\infty} b_k h(t-kT, T)
\]

The key idea behind PDA is that we can formulate the WC eye computation at time \( \Delta \) as a pair of optimization problems:

\[
\text{WC0} (\Delta, T) = \max_{\{b\}} \sum_{k=-\infty}^{\infty} b_k h(\Delta-kT, T) \\
\text{subject to } b_0 = 0, \text{ and}
\]

\[
\text{WC1} (\Delta, T) = \min_{\{b\}} \sum_{k=-\infty}^{\infty} b_k h(\Delta-kT, T) \\
\text{subject to } b_0 = 1
\]

The intuition behind PDA is that, simply depending on the sign of the \( h(\cdot) \) term, we decide whether we want the corresponding bit to be a 0 or a 1. For example, if we are trying to maximize the output, and we find a \( h(\cdot) \) term that is positive, we must pick the bit that multiplies the \( h(\cdot) \) term to be a 1, and so on. This, of course, is possible only because the bits \( \{b_k\} \) are all uncorrelated and can be set independently of one another. If, however, the bits \( \{b_k\} \) are correlated (e.g., if they are produced by an FSM), then the entire method breaks down and more powerful techniques (such as the one we propose in §III) are needed.

B. FSMs as a way to model bit correlations

Our chief concern in this paper is the computation of WC eye diagrams in the presence of arbitrary bit correlations. Therefore, we first need to consider how to represent these bit correlations in a general way, so that they can be conveniently and efficiently analysed by our algorithms. In this context, Finite State Machines (FSMs) are powerful data structures that enable us to represent virtually any kind of bit correlation in a fast, convenient, and generally useful way.

Fig. 3. Example FSM for representing bit correlations: no two consecutive 1s.

Briefly, an FSM is a data structure that consists of finitely many discrete states (e.g., see Fig. 3). Each FSM represents an automaton, i.e., a machine that behaves according to a pre-defined logic. At the beginning, the machine is in one of a few specially designated states known as start states. For instance, the FSM of Fig. 3 has exactly one start state, labelled \( S_0 \). In addition, FSMs also have arcs (or edges) between states. These arcs are annotated with output bits, as shown in Fig. 3. The automaton operates in discrete time (e.g., at every uptick of a periodic clock signal). At each discrete time point, the machine transitions from its current state, along one of the arcs directed outward from its current state, and reaches a new state (wherever the arc leads to). During this time, the machine’s
output is said to be the bit along the arc that was just followed. For example, with reference to Fig. 3, if the machine is in state 50, it could transition to either S1 (producing a 1 as output), or it could remain in state 50 (producing a 0).

It is easy to see from the above example that the bit sequences produced by FSMs are correlated. Each bit cannot be set independently of the others. For example, the FSM of Fig. 3 can never produce two consecutive 1s at the output.

It can be shown that virtually any digitally implementable system, including complex communications protocols, encoders and decoders, error-correcting codes, etc., can be represented as an FSM. This enables us to use FSMs as a general way to specify arbitrary bit correlations for WC eye diagram analysis.

III. CORE TECHNIQUE: A NEW ALGORITHM FOR NON-LINEAR SI ANALYSIS

Suppose \( b = (\ldots, b_{-2}, b_{-1}, b_0, b_1, b_2, \ldots) \) is the bit sequence (with unit interval/period \( T \)) generated by the FSM on the transmit side (i.e., the input to the LTI channel as shown in Fig. 2). As discussed in Section II-A, the channel output \( y(t) \) upon applying the above input is given by:

\[
y(t) = \sum_{k=-\infty}^{\infty} b_k h(t - kT),
\]

where \( h(\cdot) \) denotes the pulse response of the channel.

Now, suppose a bit value \( b_0 \) is applied at the input of the LTI channel during the time interval \([0, T]\). As \( b_0 \) propagates to the output of the LTI channel, let us assume that we sample the output at time \( \Delta > 0 \). In other words, \( y(\Delta) \) gives the propagated value of \( b_0 \) at the output. We are interested in measuring how much perturbation the LTI channel could introduce to the value of \( b_0 \), in the worst case, depending on the other input bits. If \( b_0 = 0 \), the maximum possible value of \( y(\Delta) \) gives the worst case perturbation for logic value 0. We call this WC0. If \( \Gamma \) denotes the set of all possible bit sequences generated by the FSM, we may define WC0 as follows:

\[
WC0 = \max_{\Gamma} y(\Delta) \quad \text{subject to} \quad b_0 = 0
\]

Similarly, if \( b_0 = 1 \), the minimum value of \( y(\Delta) \) represents the worst case perturbation to logic value 1. We call this WC1, and it is given by:

\[
WC1 = \min_{\Gamma} y(\Delta) \quad \text{subject to} \quad b_0 = 1
\]

We approximate Eq. (5) by substituting \(-\infty\) with a large negative integer \(-N_1\). Thus, we obtain:

\[
WC0 = \max_{\Gamma} \sum_{k=-N_1}^{\Delta/T} b_k h(\Delta - kT) \quad \text{subject to} \quad b_0 = 0,
\]

and similarly for WC1.

Now, recall that the bit sequences in \( \Gamma \) are generated by an FSM. Some bit sequence \( x \in \Gamma \) would have two consecutive bits \( b_i \) and \( b_{i+1} \) iff there is a reachable sequence of transitions \( s_i \rightarrow s_{i+1} \rightarrow s_{i+2} \) in the FSM such that \( b_i \) and \( b_{i+1} \) are produced along transitions \( s_i \rightarrow s_{i+1} \) and \( s_{i+1} \rightarrow s_{i+2} \), respectively. We use \( b(s_i, s_{i+1}) \) to denote the bit that is generated when the FSM moves from state \( s_i \) to state \( s_{i+1} \).

The graph structure of the underlying FSM helps us solve the optimization problem of Eq. 7 very efficiently, using dynamic programming. The intuition is that, suppose we want to find the most optimal FSM path (i.e., the path that maximizes or minimizes the above objective function, as the case may be) that ends in state \( s_j \) at time \( k \), then we only have to consider the arcs leading into state \( s_j \). For example, if we have an arc \( (s_j, s_i) \), we must consider the optimal path that ends in state \( s_j \) at time \( k - 1 \). Picking out the most optimal path, from among the choices available for \( s_j \) will produce the desired result. This exposes the optimal sub-structure of the problem, and our algorithm for WC eye computation based on this idea is formalised in Algorithm 1.

Algorithm 1: Finding WC0, WC1 using dynamic programming

```plaintext
1: \( k \leftarrow -N_1 - 1; \ k_{\max} = \lfloor \Delta/T \rfloor; \)
2: forall \( v \in S \) do
3:   \( v \in S_0 \) ? \( \text{dmapmap}[v] \leftarrow 0 : \text{dmapmap}[v] \leftarrow \perp; \)
4: end
5: while \( k < k_{\max} \) do
6:   forall \( u \in S \) do
7:     \( \text{tmpmap}[u] \leftarrow \perp; \)
8:   end
9:   forall \( u \in S \) do
10:      \( \text{opt_node}_{\text{exists}} \leftarrow \text{false}; \ \text{opt_node} \leftarrow 0; \)
11:      forall \( v \in S \) such that \( v \rightarrow u \) do
12:         if \( k = 0 \) and \( b(v, u) \neq b_0 \) then continue;
13:      end
14:      if \( \text{dmapmap}[v] \neq \perp \) then
15:         \( \text{opt}_{\text{xy}} \leftarrow \text{dmapmap}[v]; \)
16:      else
17:         if \( b(v, u) = 0 \) then
18:            \( \text{opt_node}_{\text{cand}} \leftarrow \text{opt}_{\text{xy}}; \)
19:         else
20:            \( \text{opt_node}_{\text{cand}} \leftarrow \text{opt}_{\text{xy}} + h(\Delta - k \cdot T); \)
21:         end
22:      end
23:      if \( \text{opt_node}_{\text{exists}} = \text{false} \) then
24:         \( \text{opt_node} \leftarrow \text{opt_node}_{\text{cand}}; \)
25:      else
26:         if \( (2b_0 - 1) \cdot (\text{opt_node}_{\text{cand}} - \text{opt_node}) < 0 \) then
27:            \( \text{opt_node} \leftarrow \text{opt_node}_{\text{cand}}; \)
28:         end
29:      end
30:   end
31:   \( \text{dmapmap} \leftarrow \text{tmpmap}; \)
32:   \( k \leftarrow k + 1; \)
33: end
34: \( \text{return} \ b_0 = 0 ? \max(\text{dmapmap}[u]) : \min(\text{dmapmap}[u]); \)
```

IV. RESULTS

We now apply the techniques discussed above to two systems that are of interest to high-speed communications system architects. Both these systems take the form shown in Fig. 2: an FSM that produces correlated bit sequences, followed by an analog LTI channel. In the first system (§IV-A), the FSM implements a \((7, 4)\)-Hamming encoder, and the LTI channel is a chain composed of RLGC units. In the second system (§IV-B), the FSM implements an \(8b/10b\) encoder, and the channel is a smooth behavioral multi-tap LTI filter. In each case, we show that PDA makes overly pessimistic predictions about the WC eye diagram at the channel output; by contrast, our approach predicts the WC eye opening both accurately and efficiently.

A. \((7, 4)\)-Hamming encoder followed by an RLGC chain

We now apply our technique to carry out WC eye diagram analysis of a \((7, 4)\)-Hamming encoded communications scheme (a very commonly
used parity-based error correcting code) over an LTI channel composed of a chain of RLGC units.

Fig. 4. A (7,4)-Hamming encoder that converts 4 parallel data bits into 7 serial Hamming encoded bits.

Fig. 4 depicts the (7,4)-Hamming encoder, which accepts 4 data bits \(d_1\) through \(d_4\) in parallel, and outputs a serial stream of 7 Hamming-encoded bits \((p_1\) through \(p_7)\). The relationship between the output bits \(\vec{p}\) and the data bits \(\vec{d}\) is given by the following equation:

\[
\begin{bmatrix}
    p_1 \\
p_2 \\
p_3 \\
p_4 \\
p_5 \\
p_6 \\
p_7 \\
\end{bmatrix} = \begin{bmatrix}
    1 & 1 & 0 & 1 & 1 & 0 & 1 \\
    1 & 0 & 1 & 1 & 0 & 0 & 0 \\
    0 & 1 & 1 & 1 & 1 & 0 & 0 \\
    0 & 1 & 0 & 0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 & 0 & 0 & 1 \\
    0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\end{bmatrix} \begin{bmatrix}
    d_1 \\
d_2 \\
d_3 \\
d_4 \\
d_5 \\
d_6 \\
d_7 \\
\end{bmatrix} \pmod{2}
\]

Equivalently, the above relationship can be tabulated as follows:

<table>
<thead>
<tr>
<th>(d_{1-4})</th>
<th>(p_{1-7})</th>
<th>(d_{1-4})</th>
<th>(p_{1-7})</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000000</td>
<td>1000</td>
<td>1110000</td>
</tr>
<tr>
<td>0001</td>
<td>1110101</td>
<td>1001</td>
<td>0011101</td>
</tr>
<tr>
<td>0010</td>
<td>0101010</td>
<td>1010</td>
<td>1011010</td>
</tr>
<tr>
<td>0011</td>
<td>1000011</td>
<td>1011</td>
<td>0110011</td>
</tr>
<tr>
<td>0100</td>
<td>1011100</td>
<td>1100</td>
<td>0111100</td>
</tr>
<tr>
<td>0101</td>
<td>0101011</td>
<td>1110</td>
<td>0111101</td>
</tr>
<tr>
<td>0110</td>
<td>1100010</td>
<td>1110</td>
<td>0011110</td>
</tr>
<tr>
<td>0111</td>
<td>0001111</td>
<td>1111</td>
<td>1111111</td>
</tr>
</tbody>
</table>

Table 1. Mapping \(\vec{d}\) to \(\vec{p}\) using the (7,4)-Hamming encoding.

We now construct an FSM that produces exactly the same bit sequences as the (7,4)-Hamming encoder above. To do so, we observe that not all 7-bit combinations are valid Hamming codewords. Indeed, although there are 128 possible combinations of 7 bits, only 16 of these represent valid codewords. These codewords can be represented using a suffix tree as shown in Fig. 5.

The root of the above suffix tree denotes the start state of our FSM. Each internal node in the suffix tree represents an FSM state, and the 16 leaves of the suffix tree (representing valid Hamming codewords) are looped back to the start state of the FSM (using the edges labelled reverse arcs in Fig. 5) in order to reset the encoder, so that it becomes ready to produce the next codeword. With that, it is readily seen that the FSM so constructed produces exactly the same bit patterns that a (7,4)-Hamming encoder is capable of generating. The full structure of this FSM (including all internal nodes and arcs) is depicted in Fig. 6.

Following the block diagram of Fig. 2, the correlated bits produced by the above FSM now feed into an analog LTI channel that consists of a chain of 10RLGC units, as shown in Fig. 7 below.

Fig. 7. A 10-unit RLGC chain used to model the analog channel following the (7,4)-Hamming decoder.

Given the system description above, we now determine the WC eye diagram at the output of the channel (i.e., at the receiver), using the dynamic programming algorithm presented in §III. In addition, we also validate the eye diagrams produced by our algorithm – by carrying out a large number of Monte Carlo simulations on the above FSM, and then using the results to construct an approximate eye diagram at the channel output. Furthermore, we also apply PDA to the above channel (assuming uncorrelated bits at the input), and plot the resulting eye diagram to demonstrate the overly pessimistic predictions of PDA.

The results are presented in Fig. 8. Note that each of the 7 bits produced by the encoder (per codeword) gives rise to a differently shaped eye opening, as seen from Fig. 8. This is a direct consequence of the correlated nature of the bits produced by the Hamming encoder – which PDA is unable to account for. Indeed, as seen from Fig. 8, the dynamic programming approach outlined in this paper is able to accurately reproduce the true shapes of the WC eye diagrams for the given system (closely matching the Monte Carlo simulation runs), unlike PDA, which often predicts overly pessimistic WC eye diagrams.

B. An 8b/10b encoded SERDES signaling system

We now apply our technique to an 8b/10b SERDES encoder, followed by a smooth behavioral multi-tap LTI channel.

In this system, the encoder accepts as input 8 data bits (A through H) in parallel, and produces as output 10 serialized bits (a, b, c, d, e, f, g, h, and j). The encoded bits have some very desirable properties that often result in significantly improved WC eye diagrams at the receiver; for example, it is guaranteed that no more than 5 consecutive 0s or 1s will occur at any time in the output bitstream. Due to these advantages, 8b/10b encoding is a very popular and widely used communications scheme, and it plays a role in several high-speed communications standards, including Gigabit ethernet, USB 3.0, SATA and SAS, PCI Express, etc.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
<th>abcdel</th>
<th>Inputs</th>
<th>Outputs</th>
<th>abcdel</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>010111</td>
<td>011000</td>
<td>01000</td>
<td>011011</td>
<td>100100</td>
</tr>
<tr>
<td>00001</td>
<td>011101</td>
<td>100010</td>
<td>01001</td>
<td>010011</td>
<td>010011</td>
</tr>
<tr>
<td>00010</td>
<td>101101</td>
<td>010110</td>
<td>01010</td>
<td>011011</td>
<td>011011</td>
</tr>
<tr>
<td>00011</td>
<td>110001</td>
<td>110001</td>
<td>10010</td>
<td>011101</td>
<td>100101</td>
</tr>
<tr>
<td>00100</td>
<td>110101</td>
<td>010110</td>
<td>10100</td>
<td>011011</td>
<td>101011</td>
</tr>
<tr>
<td>00101</td>
<td>101010</td>
<td>101010</td>
<td>10101</td>
<td>011101</td>
<td>101101</td>
</tr>
<tr>
<td>00110</td>
<td>011010</td>
<td>111010</td>
<td>10110</td>
<td>011101</td>
<td>101111</td>
</tr>
<tr>
<td>00111</td>
<td>111001</td>
<td>001010</td>
<td>11100</td>
<td>001101</td>
<td>111011</td>
</tr>
<tr>
<td>01000</td>
<td>111001</td>
<td>001110</td>
<td>11100</td>
<td>001111</td>
<td>111101</td>
</tr>
<tr>
<td>01001</td>
<td>100101</td>
<td>110011</td>
<td>11101</td>
<td>100111</td>
<td>111111</td>
</tr>
<tr>
<td>01010</td>
<td>010101</td>
<td>010101</td>
<td>11110</td>
<td>101101</td>
<td>101011</td>
</tr>
<tr>
<td>01011</td>
<td>110100</td>
<td>111010</td>
<td>11110</td>
<td>101111</td>
<td>101111</td>
</tr>
<tr>
<td>01100</td>
<td>011010</td>
<td>111100</td>
<td>11110</td>
<td>110000</td>
<td>101000</td>
</tr>
<tr>
<td>01101</td>
<td>101100</td>
<td>101100</td>
<td>11111</td>
<td>110001</td>
<td>101001</td>
</tr>
<tr>
<td>01110</td>
<td>011100</td>
<td>111110</td>
<td>11111</td>
<td>110010</td>
<td>101010</td>
</tr>
<tr>
<td>01111</td>
<td>011110</td>
<td>111111</td>
<td>11111</td>
<td>110100</td>
<td>101100</td>
</tr>
</tbody>
</table>

Table 2. Stage 1 of the 8b/10b encoder, i.e., a 5b/6b mapping.
Fig. 6. The full (7,4)-Hamming encoder FSM based on the 16 codeword suffix tree.

Fig. 8. Eye diagrams predicted by Monte Carlo simulation (various colors), by PDA (black), and by our dynamic programming technique (red) for the (7,4)-Hamming encoded communications scheme of §IV-A. It is clearly seen that the approach presented in this paper is able to produce exact and accurate WC eye diagrams for the given system, unlike the overly pessimistic eye diagrams predicted by PDA.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs fghj</th>
</tr>
</thead>
<tbody>
<tr>
<td>HGF</td>
<td>RD = −1</td>
</tr>
<tr>
<td>000</td>
<td>1011</td>
</tr>
<tr>
<td>001</td>
<td>1001</td>
</tr>
<tr>
<td>010</td>
<td>0101</td>
</tr>
<tr>
<td>011</td>
<td>1100</td>
</tr>
<tr>
<td>100</td>
<td>1101</td>
</tr>
<tr>
<td>101</td>
<td>1010</td>
</tr>
<tr>
<td>110</td>
<td>0110</td>
</tr>
<tr>
<td>111</td>
<td>1110</td>
</tr>
<tr>
<td></td>
<td>0111</td>
</tr>
</tbody>
</table>

Table 3. Stage 2 of the 8b/10b encoder, i.e., a 3b/4b mapping.

The encoder implementation works as follows. The 8b/10b encoding is actually carried out in two stages, a 5b/6b stage that encodes the bits EDCBA into a 6-bit word abcdel, followed by a 3b/4b stage that encodes the bits HGF into a 4-bit word fghj. In each of these two stages, the encoder keeps track of a quantity called the Running Disparity (RD), which is defined as the number of 1s minus the number of 0s in the bitstream produced thus far. The encoding used for the 5b/6b stage is dependent on the current value RD, which is always either −1 or +1. For each of these cases, the exact encoding is presented in Table 2. Following the 5b/6b stage, the RD is updated (again, it will be either −1 or +1), before applying the 3b/4b encoding of Table 3 (where the 3b/4b encoding for input 111 is chosen to avoid a streak of 5 consecutive 1s or 0s). The RD is again updated, and the process then repeats itself with the next 8 parallel incoming bits.

Fig. 9. Schematic for an 8b/10b encoder FSM based on 5b/6b and 3b/4b codes.
We see from Fig. 10 that our dynamic programming technique is able to accurately reproduce the true shapes of the WC eye diagrams for the given system, unlike the overly pessimistic eye diagrams predicted by PDA.

The FSM schematic for the 8b/10b SERDES encoder is shown in Fig. 9. This FSM has two start states, labelled Sa and Sb in the figure, corresponding to RD being −1 and +1 respectively. As Fig. 9 shows, the first stage (the 5b/6b encoding) is implemented using 64 FSM paths (32 emanating from Sa and 32 from Sb), each of which has 5 intermediate FSM states and a terminal state that is one of the 4 green states shown in the figure. This is followed by the 3b/4b encoding stage, which consists of 32 additional FSM paths (8 paths emanating from each of the 4 green FSM states), that eventually loop back to Sa or Sb after traversing 3 intermediate states each.

The channel model following the above FSM is a smoothed behavioral multi-lap LTI channel, whose unit pulse response \( h(t, T) \) is given by:

\[
    h(t, T) = \sum_{i=0}^{N-1} \frac{a_i}{2} \left[ \tanh \left( \frac{k_1}{T} (t - \delta - iT) \right) - \tanh \left( \frac{k_2}{T} (t - \delta - (i+1)T) \right) \right]
\]

Similar to our analysis of the (7,4)-Hamming encoder §IV-A, we now use our dynamic programming algorithm to determine the WC eye diagram for the 8b/10b SERDES system above. And just like the previous system, we also plot Monte Carlo simulation based eye diagrams, and the pessimistic WC eye diagrams predicted by PDA. And just like the previous case, we see from Fig. 10 that our dynamic programming technique is able to accurately reproduce the true shapes of the WC eye diagrams for the given system (closely matching the Monte Carlo simulation runs), whereas the predictions made by PDA are highly pessimistic. This demonstrates that our technique is indeed a viable and practical solution for predicting WC eye diagrams in modern high-speed signaling systems.

V. SUMMARY, CONCLUSIONS, AND FUTURE WORK

To summarise, we have developed and demonstrated a new algorithm and technique for computing WC eye diagrams in high-speed signaling systems, in the presence of arbitrary bit correlations. To the best of our knowledge, no such technique was known prior to this work. Our technique works by expressing bit correlations using FSMs, a general approach that can model virtually any encoder or error correcting scheme in existence. We then formulate the worst case eye diagram computation as a combinatorial optimization problem, which is then solved exactly and efficiently using dynamic programming. Thus, our approach offers several advantages. Firstly, it is fully general (it applies equally well to any kind of bit correlations, and any LTI channel model). Secondly, it is highly efficient: it only takes time linear in the size of the given FSM, which makes it suitable for practical use in the analysis of several modern high-speed signaling systems. Thirdly, our technique is exact: it is neither overly pessimistic (unlike PDA), nor unduly optimistic (unlike many existing simulation-based approaches); it predicts the WC eye provably and accurately.

In this paper, we applied our method to analyze two systems: a (7,4)-Hamming encoder, and an 8b/10b SERDES encoder (both represented as FSMs, and combined with channel models). In each case, we demonstrated that our approach is able to quickly and accurately predict worst case eye diagrams at the receiver (tallying well with Monte Carlo simulations), unlike PDA which makes overly pessimistic and predictions.

In future, we would like to apply our technique to many more systems of interest to high-speed link designers and communications system architects. In addition, we would like to explore algorithmic refinements that will enable our technique to directly analyze error-correcting encoders, communications protocols, and other Boolean systems (e.g., circuits expressed in Verilog, or as And Inverter Graphs or Binary Decision Diagrams), without having to first convert such systems into FSM form.

REFERENCES