Analysis and Optimization of Multi Gb/s Chip-to-Chip Communication

Raheem Alhamdani, M. Lucas Loero, Ben Meakin, Jordan Kemp, Bryson Kent, Ken Stevens

***Abstract*— This paper presents a worst case and statistical analysis of a high speed chip-to-chip communication link. Implemented in graphical user interface this tool will read data from an outside source and output the resulting performance diagram, pass/fail, and bit error rate on a user-defined basis. The software tool will use mathematical models to characterize the performance of a high speed chip-to-chip interconnect. This software tool will include multiple sources of co-channel interference and transmitter/receiver jitter. The goal is to produce an output of simulated results that match real world applications to help understand if a design is valid.**

I. Introduction

In modern high performance computing systems, memory access is a major bottleneck. A single access to main memory can take hundreds of clock cycles. To a degree, modern cache and out-of-order processor architectures get around this by exploiting locality and instruction-level parallelism. However, an access to off-chip memory is inevitable. Designing fast chip-to-chip interconnects is critical in reducing this bottleneck.

High-frequency chip-to-chip communication introduces signal degradation and limits the maximum achievable data-rates. Some of the issues encountered with chip to chip communication are jitter, signal noise, and reference voltage noise. Many of these problems are caused by channel impact on the analog signals used to represent bit values at high frequencies. Other problems result from timing uncertainty in clocking which makes sampling correct bit values more difficult. There are several methods used for dealing with these problems. This project will focus on calculating the worst possible outcome and a statistical approach to verification of interconnect designs. The work done in this project is largely based on previous written work listed in the references. It is also recognized that a previous attempt has been made in creating a software tool and is known as Stateye. This program has many limiting factors and usability factors.

To ensure that these issues do not result in intolerable bit error rates, it is important to calculate all the possible sources of error and add them together in an attempt to validate a working design before fabrication. This can be done by running a model of the design many times and plotting the results over one period of time. This plot is called an eye-diagram. Running a model of the design many times can be done in a variety of simulation products. This type of analysis is called a worst-case scenario. The worst-case scenario is a condition that may occur but will not reflect the systems generalized performance. Identifying this condition requires running the system model exhaustively and takes over 10 days. Thus it is our goal to calculate this condition using mathematical models and pulse based analysis.

An eye-diagram can also be generated using a statistical approach. Using a statistical analysis it is possible to calculate the bit error rate under normal operating conditions. The statistical analysis computes the probability density function (PDF) and cumulative distribution function (CDF) of the output data of a pulse sent down the channel. This data is then used to show a bit error rate (BER) eye diagram of the channel, which shows the probability of error given a sample voltage and a sample time. This is useful because it shows the probability of error for a user-specified voltage and time, rather than just a pass/fail output as calculated in the worst case analysis.

The purpose of the software tool is to use these techniques to assist the validation of interconnect designs. The tool is cross platform and utilizes a graphical user interface to allow for maximum usability. The tool allows the user to select a statistical or worst-case analysis of the signals and output the corresponding pass/fail or bit error rate. The tool will then display the resulting eye diagram and related information. The tool also allows the user to define timing jitter and co-channel distortion as desired. The mathematical computations done in this project are based on previously written documents referenced at the end of this paper.

II. Worst Case Analysis

This section describes the computations used to generate a worst case analysis. This project is based on pulse based analysis which is the convolution of a transmitter symbol response and the channel impulse response. It should also be noted that this work is based on previous work and should be consulted for further explanation [1]. This will give as a new pulse than contains all needed information about the system, this pulse is labeled y(t). The maximum signal degradation due to ISI is computed by summing all multiples of the period. This is given in equation (1) from [1]. This will give us the bottom half of the eye diagram while the top half is given by equation (2) from [1].

$$S_0(t) = \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} y(t-kt)|_{y(t-kt)>0} \quad (1)$$

$$S_1(t) = y(t) + \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} y(t-kt)|_{y(t-kt)<0} \quad (2)$$

Co-channel interference is the induced voltage on the line and thus behaves like ISI. To include co-channel interference a similar approach was taken. A time shift is added to the co-channel pulse to center it over the ISI pulse, this gives maximum distortion. This computation is done with equation (3) from [1] and can be added to equation (1) to form a worst case analysis. A similar equation can be written for a new S1(t).

$$\sum_{i=1}^{n} \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} y^i(t-kt-t_i)|_{y^i(t-kt-t_i)>0} \quad (3)$$

Using equation (3) also allowed for multiple sources of co-channel interference. It is also possible to add more terms of distortion to the above equations to produce a total worst case analysis. The final equations implemented included terms for transmitter and receiver jitter. The equations were first implemented and verified in Matlab. The computations were later converted into Java. Matlab simulations were done over a 6 inch channel at 4Gbps and provided the eye in figure 1. Co-channel sources were then added to the simulation and closed the eye as expected. The results in figure 1 can be compared to figure 2 for verification. Figure 2 is the system performance with no co-channel interference.

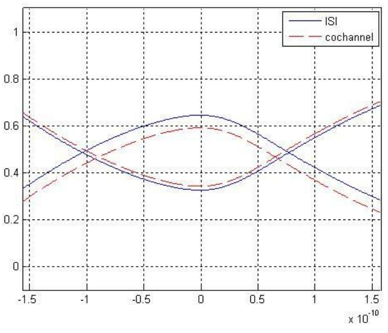


Figure 1: Worst case ISI and co-channel interference

The results in figure 1 can be compared to figure 2 for verification. Figure 2 is the system performance with no co-channel interference.

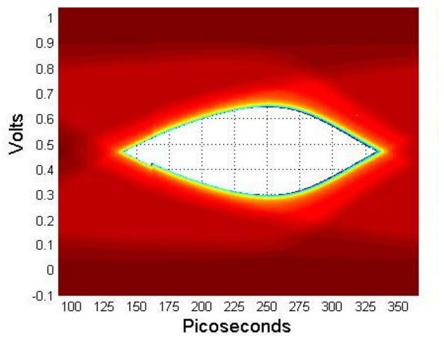


Figure 2: Worst case ISI for 6 in channel

III. STATISTICAL ANALYSIS

The statistical analysis is similar to the worst-case analysis, but there are some key differences. The worst-case analysis sums up all negative Inter-Symbol Interference (ISI) values to bring down the top of the eye diagram, and sums up all the positive ISI values to bring up the bottom of the eye diagram. This says that any possible interference that could happen will happen. The resulting eye diagram is an eye that gives two regions: one region where errors will occur, one region where errors will not occur. The statistical analysis takes into account the probability of the interference happening. The resulting diagram is a Bit-

Error Rate (BER) eye diagram of sample voltage versus sample time versus probability of error, shown in Figure 3 below. This Figure is for the same channel as used in the worst case analysis.

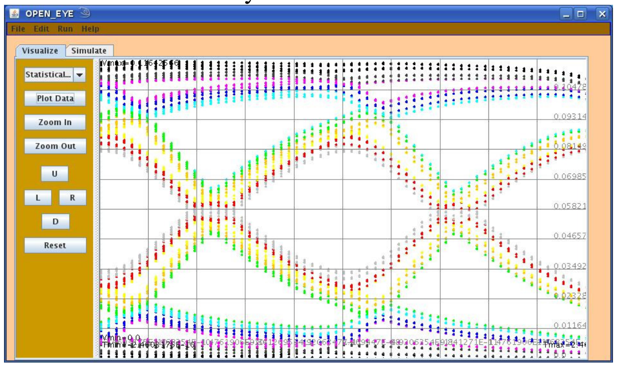


Figure 3: BER eye diagram (color represents probability of error)

The statistical analysis is computed by first convolving the channel impulse response, p(t), and the transmitter symbol response, c(t), to get the simulated output of the channel, y(t), which is then used to find the Probability Density Function (PDF) of the data using equation (4) from [1].

$$Z_{k+1}(\tau,t) = \begin{cases} \frac{\delta(\tau)+\delta(\tau-y(t-kT))}{2} \otimes Z_k(\tau,t) & k \neq 0 \\ Z_k(\tau,t) & k = 0 \end{cases} \quad (4)$$

The Cumulative Distribution Function (CDF) is then computed from the resulting PDF assuming equal probability of 0's and 1's.

$$\text{Probability of Error}[\lim_{\text{voltage}\to-\infty}] = \text{Probability of Error}[\lim_{\text{voltage}\to+\infty}] = 0.5$$

The center of the eye in the BER eye diagram is taken as the peak of the channel output. The convolution of the values at integer multiples of the symbol period and the peak value itself result in the probability of error versus sample voltage for that specific time. This computation is done for all values of time in the range of 1 Unit Interval (UI) to the left of the peak value and 1 UI to the right of the peak value. This computation results in the BER eye diagram.

The statistical analysis was implemented in the project by adding ISI values and concatenating as shown in equation (5) below, rather than convolving delta functions.

$$\mathbf{[0, -0.01] \otimes [0, 0.59] = [(0+0), (-.01+0), (0+.59), (-.01+.59)] = [-.01, 0, .58, .59]} \quad (5)$$

From this equation, it is shown that the convolution of a delta function at zero and a sampled ISI value with another delta function at zero and a second sampled ISI value is just the addition of these values concatenated with the original sampled ISI. The end result of all the additions is then plotted against the height of each of the delta functions. This method is much quicker than traditional convolution methods, it is easy to implement, and it has infinite precision (in theory), because it takes the actual values at the sample points, rather than a vector with a specified precision, and impulse values at specific index values within the vector. It is also noted that to add co-channel and jitter distortion we follow the same convolution process.

IV. Jitter

One of the main goals of this project was to generate a worst-case eye diagram for transmitter jitter and receiver jitter added to the worst case eye. The receiver jitter was analyzed using the equation (6) neglecting transmitter jitter [1]. The first part of equation (6) models the channel output of the system, where $a[n] = d[n] - d[n-1]$ is the transition sequence, and $d[n]$ is the independent and identically distributed transmitted bits sequence. The term $s[n]$ is the step response of the channel, which is a convolution of the step function and the channel impulse response $h[n]$. The step function represents the clock that gives the width for the transmitted bits and the channel impulse response represents the modulation of the channel to the signal when it is transmitted. This part of the equation represents a system with no jitter.

The second part of equation (6) represents the sampled channel output with receiver clock jitter. The term $j_{rx}[n]$ represents the receiver clock jitter due to various noise sources (intrinsic device and power supply noise. In this second part, the transmitted bits are convoluted with the channel impulse response of the system and then modulated by the receiver jitter.

$$y[n] = a[n] \otimes s[n] + (a[n] \otimes h[n]) * j_{rx}[n] \ldots (6)$$

One of requirements for this project was to model the worst-case eye diagram for the receiver jitter. This was accomplished using equation (7), and the worst-case inter-symbol-interference (ISI) of the system derived by equation (8). Equation (7) is the second part of equation (1), where $\hat{a}[n]$ is the worst case ISI distortion data derived from equation (8).

$$Receiver\ Jitter\ Noise = (\hat{a}[n] \otimes h[n]) * \max(j_{rx}[n]) \ldots (7)$$

$$Worstcase\ ISI\ Noise = \sum_{k=-\infty}^{\infty} y(t-kT)|_{y(t-kT)>0,k\neq 0} - \sum_{k=-\infty}^{\infty} y(t-kT)|_{y(t-kT)<0,k\neq 0} \dots (8)$$

We now present the simulation results obtained from Matlab. The following step were taken to obtain the worse-case eye diagram for receiver jitter; first, worst-case ISI eye diagram was calculated using equation (8), second, the receiver jitter noise was generated using equation (7), and finally the receiver jitter worse-case eye diagram was obtained by subtracting the jitter noise from the worse-case ISI eye as seen in figure 4.

The transmitter jitter was then analyzed. The same framework as the receiver jitter was used to accomplish this analysis. Taking in consideration that the receiver jitter was neglected, the first part of equation (9) is output of the channel without jitter affecting the system as explained in receiver jitter section. The second part of this equation models the degradation of the noise margin at the detector input because the transmitter clock jitter causes suboptimal sampling at the receiver due to the limited tracking bandwidth of the timing-recovery loop [4]

$$y[n] = a[n] \otimes s[n] + ((a[n] * j_{tx}[n]) \otimes h[n]) \dots (9)$$

The transmitter jitter worst-case was obtained using equation (10), where $\hat{a}[n]$ is the worst-case ISI distortion data derived from equation (8). The worst-case eye for the transmitter jitter was generated in the same form described as in receiver jitter worst-case eye. This result can be seen in figure 1.

$$Transmitter\ Jitter\ Noise = (|\hat{a}[n]| * \max(j_{rx}[n])) \otimes h[n] \dots (10)$$

We now show how both analysis can be combined into one equation. This is essential because often the effects of transmitter jitter and receiver jitter appeared together. Equation (11) shows both jitter combined, where the first part is the jitter free channel output, the second part is the receiver jitter and the third part is the transmitter jitter.

$$y[n] = a[n] \otimes s[n] + (a[n] \otimes h[n]) * j_{rx}[n] + (a[n] * j_{tx}[n]) \otimes h[n] \dots (11)$$

The worst-case eye diagram for the total jitter was simulated in same form described as in receiver jitter worst-case eye. This result can be seen in figure 4.

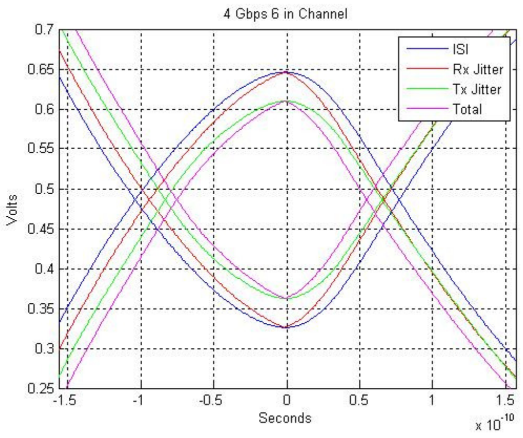


Figure 4: Jitter

V. APPLICATION DEVELOPMENT

These methods for analysis of chip-to-chip signaling systems have been incorporated into a computer-aided design tool to make their functionality accessible to interconnect engineers. This tool is called "Open Eye." Open Eye is a formal verification tool for high speed interconnects and provides the software infrastructure and functionality for an effective verification platform. It is open source and will be freely available. It provides a cross-platform graphical user interface built on the Java J2EE runtime environment, a flexible I/O system, data visualization, and a correct implementation of worst-case and statistical link analysis.

A. *Open Eye Architecture*

Open Eye is written in the Java programming language. It is object oriented, thus providing modularity and an efficient code organization. The system is built on a model-view-control (MVC) architecture as shown in figure 5. In an MVC architecture the model is the underlying software that implements all of the core functionality of the application and maintains data structures, the view is code that provides the user interface, and the control is the code that handles interaction between the model and view. The features of each of these layers are described throughout this section.

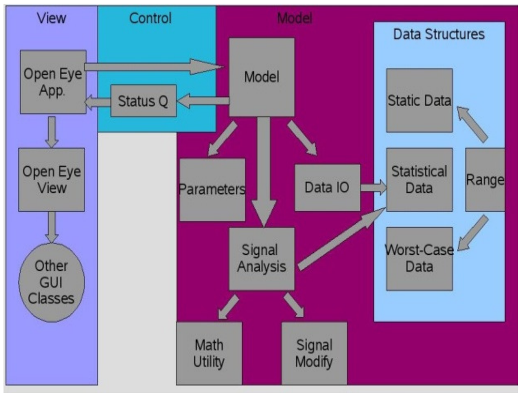


Figure 5: Software architecture

B. Simulation Types

At the heart of this software organization is the signal analysis class. It implements three simulation types which include static, worst-case, and statistical. A static simulation generates an eye diagram from a bit stream by superimposing the periods of the signal. It is so called because it does not rely on any formal methods for verification of the signaling, it only provides a naive solution based on the input data. This simulation type is included for comparison to the more robust methods.

The worst-case method is included and is based on the techniques previously described for calculating inter-symbol interference, co-channel interference, and jitter. It sums all of the negative effects on an input signal and provides the user with a worst-case eye and pass/fail result based on a user defined sampling window. It's implementation relies on a discrete time convolution and a variety of other mathematical functions provided by the math utility class. The statistical analysis method is also included and provides the user with a scatter plot of a BER diagram. Much of the code is shared across these two functions.

C. Data Structures

The core data structures of the application are abstracted into three objects, each corresponding to one simulation type; static, statistical, and worst-case. Each of these objects group pre- and post-simulation data vectors, an assortment of global variables, and functions for saving the state of these objects to the file system. Their class diagrams are given in figure 6.

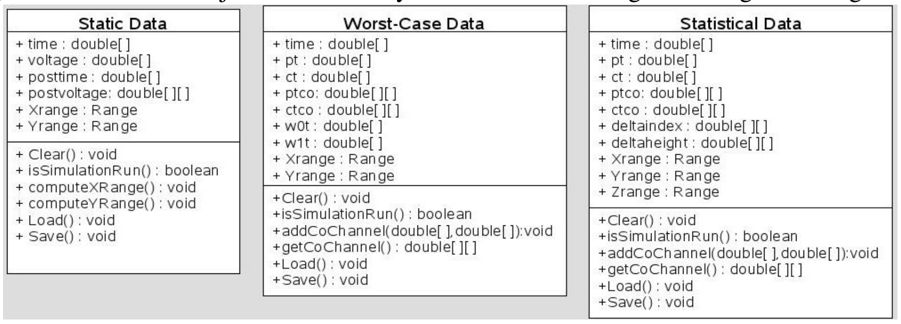


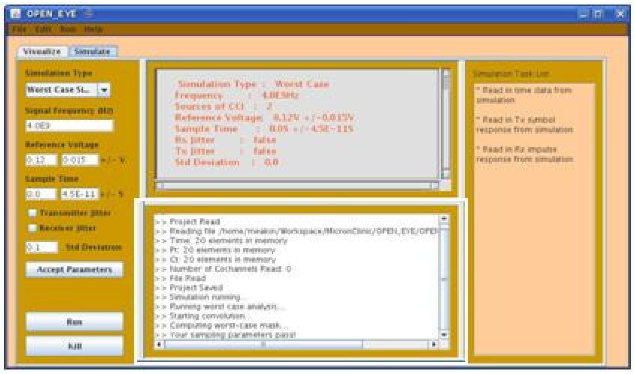
Figure 6: Class diagrams

D. I/O System

The input/output system is designed to be compatible with SPICE and Matlab text files. As input, the user enters the names of their SPICE variables and reads the data associated with those variables from a column formatted text file. Input for a given simulation can be spread across several files. After a simulation is run the post-simulation data can be persisted by saving the state of the appropriate data structure or by saving to a Matlab formated text file. Further analysis and visualization functionality will then be made available to the user through Matlab.

E. Graphical User Interface

Probably the most important feature for an application which aims to be user friendly and provide visual feedback is a graphical user interface (GUI). Open Eye's GUI is shown in figure 7.



```
Simulation Type        Worst Case St...
Signal Frequency (Hz)  4.0E9
Reference Voltage      0.12  0.015  +/- V
Sample Time            0.0   4.5E-11 +/- S
[ ] Transmitter Jitter
[ ] Receiver Jitter
0.1   Std Deviation
```

```
Simulation Type  :  Worst Case
Frequency        :  4.0E9Hz
Sources of CCI   :  2
Reference Voltage:  0.12V +/-0.015V
Sample Time      :  0.0S +/-4.5E-11S
Rx Jitter        :  false
Tx Jitter        :  false
Std Deviation    :  0.0
```

```
>> Project Read
>> Reading file /home/mealon/Workspace/MicronClinic/OPEN_EYE/OPE
>> Time: 20 elements in memory
>> Pt: 20 elements in memory
>> Ct: 20 elements in memory
>> Number of Cochannels Read: 0
>> File Read
>> Project Saved
>> Simulation running...
>> Running worst case analysis...
>> Starting convolution...
>> Computing worst-case mask...
>> Your sampling parameters pass!
```

```
* Read in time data from simulation

* Read in Tx symbol response from simulation

* Read in Rx impulse response from simulation
```

Figure 7: Graphical user interface

Among many features it provides complete control over simulation parameters, a console for textual feedback, a simulation task list, and a plotting utility. The simulation parameters include the simulation type, signal frequency, and selectable transmitter, receiver jitter and their standard deviation. The GUI also allows the user to specify a reference voltage and sample time with tolerances for defining the sampling window. This is an important verification feature for worst-case analysis.

The console and task list are designed to help guide the user through a verification flow. There are many steps involved in preparing to run a simulation and these steps differ for each simulation type. The task list tells the user what must be done prior to running a simulation. Once the task list is completed the console provides the user with feedback concerning the status of the current simulation. It also reports errors that may have occurred. This is an important feature since simulations can take several minutes to complete.

The plotting utility is another vital part of Open Eye. It allows the user to visualize any data vectors that have been read in through file input or that have been generated through simulation. It supports zooming and provides grid lines and labels to annotate the plot. For worst-case simulations it will display the sampling window with the worst-case eye to indicate to the user how close the window is to being in a failure region. For statistical simulations the utility supports scatter plots, where the color of the (X, Y) data points represents the height of the delta functions. These features combine to create a useful and intuitive application.

VI. Conclusion

This paper has shown the design of a formal verification tool for high speed electrical signaling systems based on correct methods. It has been shown that the implementation of these methods is correct and capable of verification of industrial designs. A cross platform graphical application called Open Eye which encapsulates these methods has been presented. Open Eye provides the first correct implementation of these methods in the public domain. It enables easy verification of aggressively optimized designs and supplies engineers and researchers with a tool to aid in the development of solutions to the problem of off-chip interconnect bottlenecks.

Acknowledgments

We would like to thank, Neal Patwari and Tim Hollis for all the assistance in the creation and understanding of this project.

References

[1] B. K. Casper, M. Haycock, and R. Mooney, "An accurate and efficient analysis method for multi-Gb/s chip-to-chip signaling scheme", in Digest of Technical Papers from the IEEE Symposium on VLSI Circuits, June 2002, pp. 54–57.
[2] B. K. Casper , G. Balamurugan, J. E. Jaussi, J. Kennedy, M. Mansuri, "Future microprocessor interfaces: Analysis, design and optimization", in Proceedings of the IEEE Custom Integrated Circuits Conference, Sept. 2007, pp. 479-486.
[3] P. K. Hanumolu, B. K. Casper, R. Mooney, G. Y. Wei, and U. K. Moon, "Jitter in high-speed serial and parallel links", in Proceedings of the IEEE International Symposium on Circuits and Systems, May 2004, pp. 425–428.
[4] Pavan Kumar Hanumolu*, Bryan Casper, Randy Mooney,* Gu-Yeon Wei*, and Un-Ku Moon,* "Analysis of PLL Clock Jitter in High-Speed Serial Links", in IEEE Transactions on Circuits and Systems, November 2003, pp.879-886
[5] "Stateye" accessed on April 18 2009, www.stateye.org