A Novel Characterization Technique for High Speed I/O Mixed Signal Circuit Components Using Random Jitter Injection

Ji Hwan (Paul) Chun†‡
Intel Corporation†
Austin, TX 78746
paul.chun@intel.com

Jae Wook Lee‡ and Jacob A. Abraham‡
Computer Engineering Research Center‡
The University of Texas at Austin, Austin TX 78712
{jaelee11, jaa}@cerc.utexas.edu

Abstract— Timing problems in high-speed serial communications are mitigated with phase-interpolator (PI) circuitry. Linearity testing of PI has been challenging, even though PI is widely used in modern high speed I/O architectures. Previous research has focused on implementing additional built-in circuits to measure PI linearity. In this paper, we present a cost effective PI linearity measurement technique which requires no significant modification of existing I/O circuits. Our method uses jitter distributions obtained from random jitter injected into the data channel. Two distributions are separately obtained using undersampling and sampling using PI. The proposed algorithm calculates the differential nonlinearity (DNL) from the difference of these distributions. Simulation results show that the average prediction RMS error for the DNL calculation is 0.31 LSB.

I. Introduction

The growing demand for high bandwidth computing in modern computer technology drives the development of advanced and high speed design in I/O structures. In order to achieve the goal of multi-gigabit transfer rates, industrial I/O designs such as PCI Express, Serial ATA (SATA), QuickPath Interconnect (QPI), and HyperTransport (HT) have adopted a serial interconnect scheme to resolve data skew issues in parallel data transfer schemes. Due to their speed and architecture, however, testing the serial high speed interconnects becomes a significant challenge. The challenge becomes more obvious from the perspective of test cost effectiveness when high speed and high precision test equipment is used.

In order to resolve the issue of test costs, adoption of self test methods has become an attractive solution. Without relying on costly automated test equipment (ATE), self test methods provide effective solutions to test high speed serial links. One of the cost effective methods to test high speed I/Os is to use a loop-back scheme [1]. In this technique, the I/Os are configured in a loop-back manner, and then timing and voltage margining techniques are used to determine whether the actual signal eye meets or exceeds the eye mask specification. In general, timing margining capability is enabled by using phase interpolator (PI) circuitry. The phase interpolators are used to margin the timing of the data eye to identify the total jitter of a given set of data pattern and to screen defective parts if they exceed the allowed amount of jitter in the specification. As another application of PI, Casper et al. [2] implement an on-die oscilloscope to measure the timing aspects of the signal, and the PI is used to scan the signal boundary with respect to timing. In both cases, in order to ensure the validity of the measured data, the linearity of the phase interpolator should be fairly good. In real manufacturing cases, however, process, voltage and temperature (PVT) variation significantly affects the linearity of the PI in each die. The PVT impact becomes more severe in today's highly advanced process technologies since variation tends to increase as the size of device shrinks; therefore, it is necessary to find a way to test the linearity of phase interpolator itself.

Conventional methods for testing the linearity of typical analog mixers may include direct measurement of phase relationships for various phase configurations. However, this approach is difficult to apply to PIs in high speed I/O applications since the resolution of interpolated phases needs to be in the unit of the several picoseconds and measuring of the subtle difference is a significant challenge. This challenge becomes more obvious when using external probes since loading effect degrades signal integrity more at high speeds. To mitigate this issue, measuring linearities at lower speeds can substitute for high speed measurement. However, at-speed measurement is becoming more important, since at-speed measurement of linearity shows differences from the measurements at lower speeds.

There is some previous work regarding linearity test techniques for the PI. Provost [7] proposes a PI linearity measurement technique that requires an additional phase interpolator to determine whether each PI satisfies specifications in terms of linearity. Shi et al. [8] introduce self test circuitry which is composed of a phase detector, a phase-difference-to-voltage converter, an analog-to-digital converter (ADC) and control logic. While it is possible to measure the linearity of PIs using these techniques, both these techniques require large amounts of on chip real estate as compared to that for the PI, which raises yield concern since the probability of defects in the test logic becomes greater as the size of the logic increases.

We propose a new method to measure the linearity of PI code steps without using too much additional logic. First, we inject random jitter (RJ) into the data channel. The amount of the injected jitter needs to be adjusted such that it can barely close the data eye. The distribution of the random jitter is captured by two different methods, which are undersampling and sampling using phase interpolator code sweeping. Then, the two results are compared and predicted differential non-linearities

978-1-4244-5767-0/10/$26.00 2010 IEEE

(DNLs) are derived using a series of mathematical calculations. This method can be implemented in production test in a cost-effective manner and combined with timing margining tests, such that margining steps, which are equivalent to PI code steps, are accurately mapped to actual timing information. This information can then be used to margin the timing of the eye, thus reducing the need of excessive guardbanding based on the worst case variation scenarios.

This paper is organized as follows. In Section 2, we explain various high speed I/O schemes and basic operation of phase interpolator circuitry. The motivation of the proposed technique is also described in the section. In Section 3, the proposed technique is explained in detail. Simulation results and conclusion follow in Sections 4 and 5 respectively.

II. Background

A. High Speed I/O Design and Phase Interpolator Basics

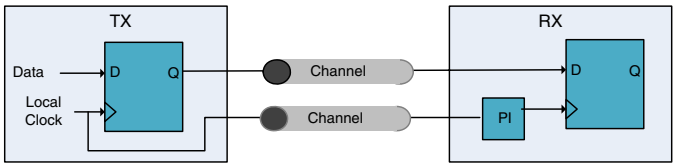

Fig. 1. Forwarded clock scheme

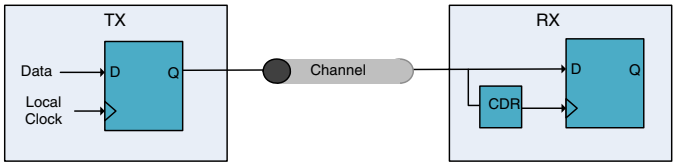

Fig. 2. Derived clock scheme

Typically, there are two architectural variations for I/O clocking schemes in high speed I/O design [3, 4]. High level block diagrams of both high speed I/O configurations are illustrated in Figures 1 and 2. The forwarded clock scheme shown in Figure 1 is used in QuickPath Interconnect (QPI), Fully Buffered Dual In-line Memory Module (FBD, FB-DIMM), etc. In this scheme, the clock signal is forwarded from transmitter to receiver which provides inherent tracking between clock and data. PI is implemented to eliminate phase skew between clock and data lanes created by board level channel length differences. Figure 2 shows the derived clock scheme which is being used in PCI express, Serial ATA, etc. In this scheme, the clock signal is derived from data transitions in data signal using clock and data recovery (CDR) circuitry. While the derived clock scheme uses fewer pins since it does not need a clock channel, the data signal is required to provide a sufficient number of transitions to guarantee proper clock signal generation at the receiver side. 8b/10b encoding is a popular scheme to provide necessary signal transitions. Although typical CDRs are designed using a structure similar to a PLL, PI based CDR scheme is gaining popularity due to its fast acquisition time as well as lower area overhead and better stability control.

Figure 3 describes a circuit level diagram of a typical phase interpolator circuit implementation [5]. The output voltage

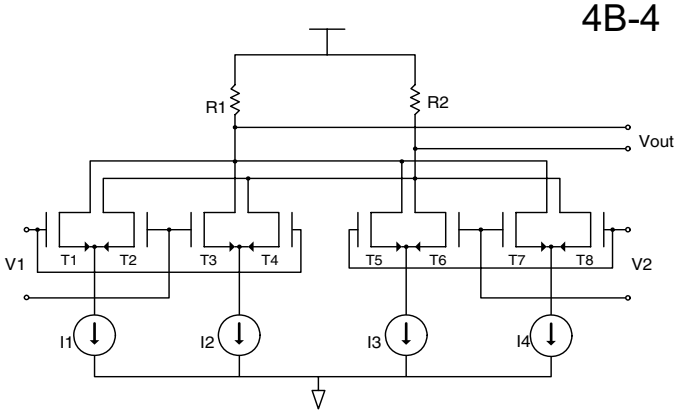

Fig. 3. Phase interpolator schematic

Index	Code	Phase Degree
0	0000	0°
1	0001	10°
2	0010	20°
⋮	⋮	⋮
8	1000	80°

TABLE I
Example Code of Phase Interpolator Encoding

V_{out} achieved by this circuitry can be expressed as follows.

$$V_{out} = a_1 V_1 + a_2 V_2 \tag{1}$$

where a_1 and a_2 are weighted factors which can be realized by adjustable current sources (I1 through I4). If V_1 and V_2 are identical sinusoidal signals with the phase difference of $90°$, Equation 1 can be rewritten as follows. Using trigonometric identities, we obtain:

$$\begin{aligned} V_{out} &= sin(\omega t)cos(\alpha) + cos(\omega t)sin(\alpha) \\ &= sin(\omega t + \alpha) \end{aligned} \tag{2}$$

where

$$\begin{aligned} V_1 &= sin(\omega t),\ V_2 = cos(\omega t) = sin(\omega t + 90°) \\ a_1 &= cos(\alpha),\ a_2 = sin(\alpha) \end{aligned}$$

Thus, it is shown that the phase of V_{out} is shifted by α.

In many phase interpolator designs, the adjustable current sources that determine the coefficients can be digitally controlled so that the possible set of interpolation can be uniformly distributed. Each digitally encoded code is mapped to certain amount of current from the current sources; thus it achieves programmed degree of interpolation. An example encoding of PI is shown in Table I.

Some phase interpolation designs use multiple phase interpolators to implement high resolution PI circuitry [10]. In one scheme, instead of using just one PI with in-phase and quadrature phase inputs to create uniformly distributed middle phase outputs, a network consisting of a coarse resolution interpolation circuit and multiple fine resolution PIs is used. Although this scheme can increase the resolution of the PI by reducing the burden of fine tuning of adjustable current sources, it increases the size of the circuitry, thus becoming more probable to be defective or more sensitive to process variation in the manufacturing process.

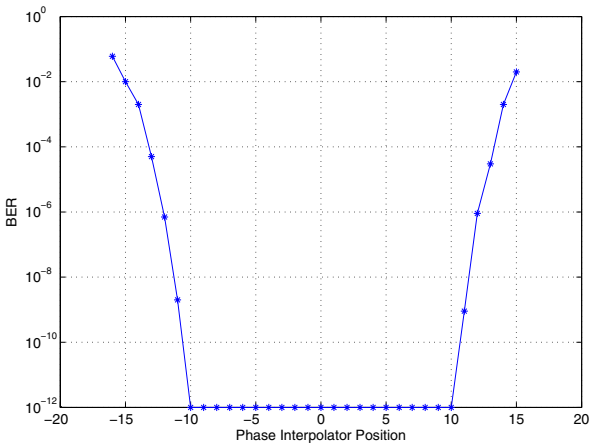

Fig. 4. Example bathtub curve of receiver

B. Impact of Nonlinearity of PI

It may be a question whether several picoseconds of nonlinearity in PIs would make a significant difference in high speed I/O performance. To illustrate the point, we use an example bathtub curve that is captured from a high speed serial link. Figure 4 is drawn based on measurement data from [6]. The curve on the left side shows that it takes about one step of the phase interpolator's position to change bit error rate (BER) from 10^{-12} to 10^{-9}. This can be translated to $\frac{1}{32}$ UI or 3.125 psec; thus 3.125 psec of jitter can degrade BER from 10^{-12} to 10^{-9}.

Addition of a small amount of total jitter can degrade the BER significantly due to the sharp slope of the bathtub curve which results from characterization of most multi gigahertz high speed I/O. If the differential nonlinearity (DNL) of the PI in the region of the eye boundary is δ and the measured total jitter using a timing margining technique is J_{total}, the worst case real jitter can be given as $\delta + J_{total}$, which could exceed specification limits depending on the test conditions, with the risk of defective part escapes.

III. Overview of The Proposed Technique

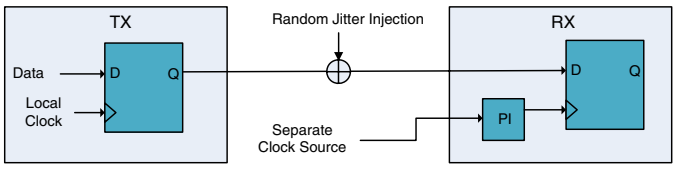

Fig. 5. Proposed configuration for forwarded clock scheme

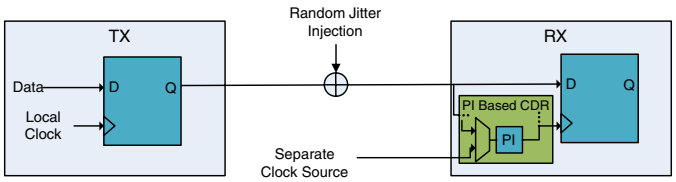

Fig. 6. Proposed configuration for derived clock scheme

The configurations for our proposed technique to measure nonlinearity of PI are described in Figures 5 and 6. We configure the transmitter (TX) and receiver (RX) such that the data channels are connected together. This can be either through a loop-back configuration on a single device or by connecting two identical devices by pairing up TX and RX. A separate clock source is introduced to undersample the data. The quality and the cost of the clock source are not of concern as high quality and low jitter clock source can be implemented on-board with low cost [11]. For the forwarded clock scheme, the clock channel is directly connected to the undersampling clock source as shown in Figure 5. For the derived clock scheme, we use a multiplexer to reach the PI in the CDR block as shown in Figure 6. With this configuration, we follow the following steps to predict the PI linearity.

First, the data port sends an alternating data pattern, i.e., the 1010 sequence to the channel. The jitter injection shown in the figure does not occur at this step. Depending on the clocking scheme of the high speed serial link, we undersample the data by injecting a clock signal with the period of $T + \Delta T$, where T is a data signal period. When we undersample the signal, we select ΔT to be the same as the phase resolution or code step of the PI. Since the PI is connected to the clock signal path, we program the PI code such that PI does not shift the phase of the signal while undersampling the signal. The collected undersampled data is then stored as the expected bit sequence from undersampling. Then we supply a normal clock signal with the period of T and use the PI to sample the alternating data pattern. We collect the sampled data for each PI code index and store it as the expected bit sequence from PI sampling for the code index i. We repeat this procedure until we sweep all combinations of PI codes and collect the sampled data.

Next, we inject random jitter into the data channel with a fair amount of variance to barely close the data eye. Since random jitter injection occurs in the data channel, jitter injection capability in the system or load board is required. Then, we undersample and PI-sample the jitter injected data signal with the same period and steps.

In order to construct the random jitter distribution vector from the injected random jitter in the data channel, error bit sequences EV_{us} and $EV_{ps,i}$ for undersampling and PI-sampling, respectively, are created by comparing the expected bit sequence with the jitter injected bit sequence. Since we reduce other interference factors by comparing bits from the same data source, random jitter injection is the only difference, and it is ensured that the distribution from the error bit sequence will be closer to a random Gaussian distribution. EV_{us} and $EV_{ps,i}$ are re-bucketed based on sampling positions of the data eye, then these distributions are stored in vector D_{pos} and D'_{pos}, respectively, after normalization. The distribution vector D_{pos} represents a random jitter distribution sampled with ΔT resolution. If we use an ideal PI to characterize the jitter, the PI sampled distribution, D'_{pos} will be identical to D_{pos} under one condition that the PI's resolution is also ΔT. Since, in reality, the PI is not ideal and contains nonlinearities, the distribution collected using PI code sweeping, D'_{pos} will be different from D_{pos}. Therefore, we can mathematically derive DNLs from the difference between D_{pos} and D'_{pos}. The details of the mathematical algorithm are explained in the following subsections.

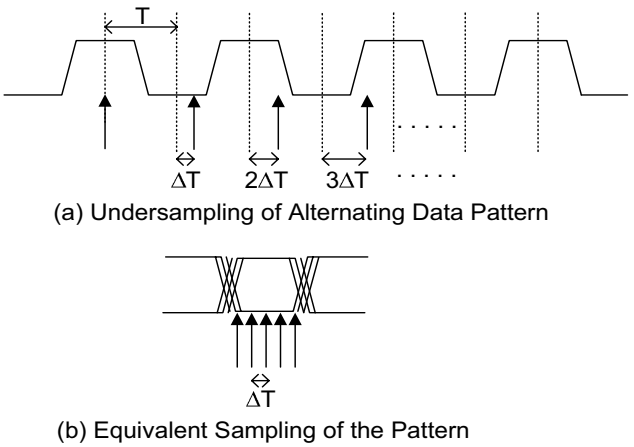

Fig. 7. Undersampling technique

A. Distribution Vector Creation Using Undersampling

Undersampling techniques have been used to measure jitter of high speed applications [11]. The concept of undersampling technique is illustrated in Figure 7. In Figure 7 (a), the period of the data signal is given as T and the period of the sampling clock is given as $T + \Delta T$. Since the sampling clock period is delayed by ΔT, every cycle of data signal is sampled with the delay of ΔT. After we collect all the sampled data and construct the eye diagram, the equivalent sampling points of the eye diagram are shown in Figure 7 (b).

By comparing the sampled jittery bit sequence with the expected bit sequence, we can derive an error vector EV_{us} and $EV_{ps,i}$ where i represents PI code index. The normalized distribution vectors can be derived by the following equations.

$$\text{For all integers } i \text{ such that } 0 \le i < \frac{T}{\Delta T},$$

$$D_{pos}(i) = \frac{\sum_{k=0}^{l}(EV_{us}(i + k\frac{T}{\Delta T}))}{\sum_{k=0}^{l} EV_{us}(k)} \quad (3)$$

$$D'_{pos}(i) = \frac{\sum_{k=0}^{l} EV_{ps,i}(k)}{\sum_{i,k=0}^{l} EV_{ps,i}(k)} \quad (4)$$

where k in the form of $EV_{us/ps,i}(k)$ represents the kth bit in the vectors and l is the largest number that makes the vector index become the maximum number.

B. Calculation of Predicted DNL

Based on the jitter distribution, D_{pos}, which is derived from the undersampling technique, we can calculate DNLs of the PI. We consider D_{pos} as a golden reference here since it can be assumed to be the identical distribution when PI DNLs are all zero. To ease the mapping of distribution and the DNLs, we use a piecewise cubic polynomial interpolation technique [12]. Piecewise cubic polynomials of D_{pos} are given as follows.

$$\text{For } i = 0, 1, 2, \ldots, n-1 \text{ and } x_i \le x \le x_{i+1},$$

$$S_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \quad (5)$$

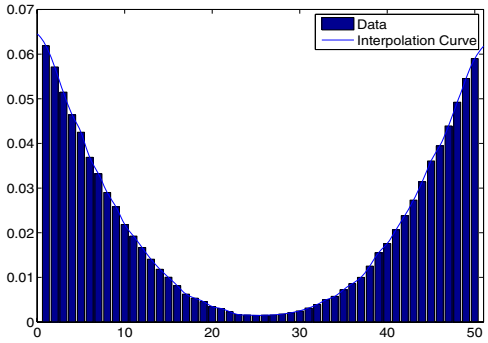

Fig. 8. Piecewise cubic polynomial interpolation of D_{pos}

For Equation 5, we need to determine $4n$ conditions to have an analytic solution for the equation. The conditions are given as follows.

$$\begin{aligned}
S_i(x_i) &= D_{pos}(i), \text{ for all } i = 0, 1, 2, \ldots, n, \\
S_i(x_{i+1}) &= D_{pos}(i+1), \text{ for all } i = 1, 2, 3, \ldots, n-1, \\
S'_{i-1}(x_i) &= S'_i(x_i), \text{ for all } i = 1, 2, 3, \ldots, n-1, \\
S''_{i-1}(x_i) &= S''_i(x_i), \text{ for all } i = 1, 2, 3, \ldots, n-1, \\
S''_0(x_0) &= S''_{n-1}(x_n) = 0. \quad (6)
\end{aligned}$$

An example of the piecewise cubic polynomial interpolation result is shown in Figure 8. Once we derive the piecewise polynomials from D_{pos}, we can create a vector that contains the estimated positions of sampling points of PI. Given $D'_{pos}(i)$, a piecewise polynomial exists so that $S_{k-1}(k-1) \le D'_{pos}(i) \le S_k(k)$. Since there exists more than one piecewise polynomial that satisfies the condition, i.e., the distribution is not monotonic but more like a U-shape, we need to divide the region so that the search space for the solution is always monotonic. In our algorithm, we divide the search space into two regions: $0 \le i \le \frac{T}{2\Delta T}$ and $\frac{T}{2\Delta T} < i < \frac{T}{\Delta T}$. Then we use the polynomial in an appropriate region to find a solution of t_i which results in $S_{k-1}(t_i) = D'_{pos}(i)$ where $k-1 \le t_i \le k$. Here, t_i represents the predicted position of sampling point for PI code index i. The predicted DNL for each PI code index i is determined by the following equation.

$$DNL_i = t_{i+1} - t_i \quad (7)$$

IV. Simulation Results

A. Simulation Configuration

MATLAB simulation was performed to validate the proposed technique. First, we configured a 10 Gbps forwarded clock serial link simulation model where the eye size for each data bit was 100 psec. Phase interpolator resolution was set to 2 psec to provide enough resolution for 10 Gbps serial links. Then we randomly generated DNLs in a uniform distribution with the maximum value of 3 LSB and injected each one of them to each code position of the PI. The proposed algorithm was implemented and the predicted DNLs were captured per PI code basis. An example simulation result from a single execution is shown in Figure 9. As shown in the figure, the random injection of DNLs and the predicted DNLs are tracking each other very well, except for the center of the PI sampling code.

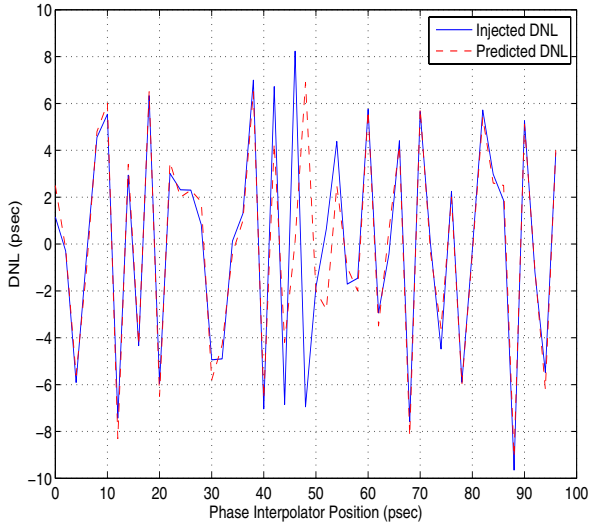

Fig. 9. Injected DNL vs predicted DNL

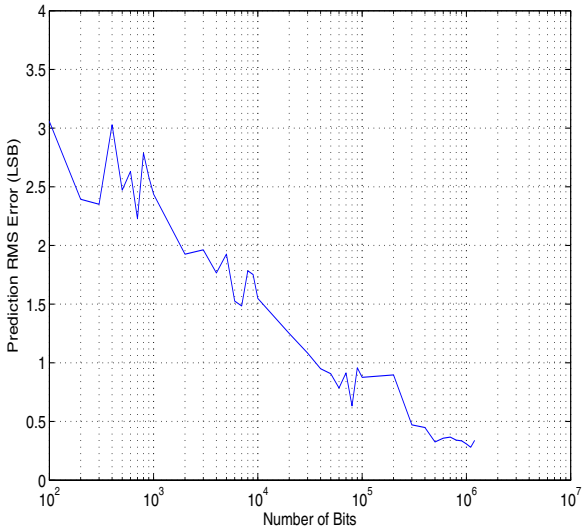

Fig. 11. Number of bits in alternating data sequence vs prediction error

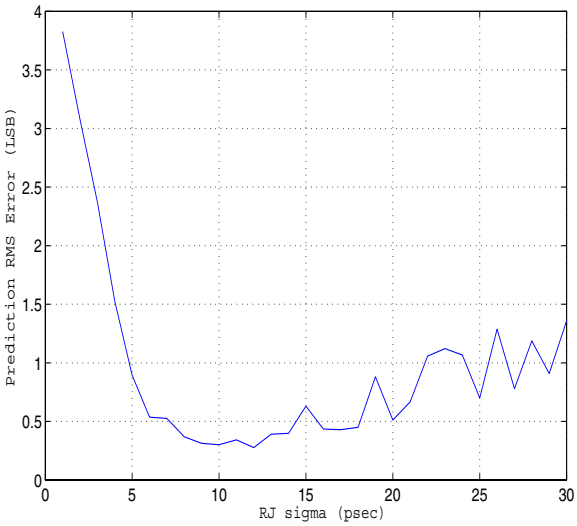

Fig. 10. Injected random jitter vs prediction error

The prediction error in the central region can be explained as follows. As the sampling position for the PI code converges to the center of the data eye, it has a lower number of the histogram samples due to the lower probability of the occurrence, which results in lower sharpness in the constructed piecewise cubic polynomial curves. This increases the prediction error rate since a sharper curve determines a smaller range of values in x axis for a given range of values in y axis, as compared to a gradual one. Although the predicted DNLs are less accurate at the central region, our proposed scheme does not seem to be flawed since the linearity of phase interpolator is more important in the region of the signal eye boundary, where BER is determined by several picoseconds of jitter. Therefore, we have configured our scheme such that we ignore the prediction error in the center of the PI code and we obtain sufficient prediction accuracy with a fair number of transmitted bit sequences.

B. Simulation Results

To determine the optimal conditions for the simulation, we performed the following experiments. First, we experimentally studied the impact of amount of RJ to the prediction accuracy. We increased the amount of the root mean square (RMS) value of the RJ's standard deviation (σ) from 1 psec RMS to 30 psec RMS, and repeated the simulation for each RJ σ value to analyze the impact. Figure 10 shows the simulation results. As can be seen, the prediction accuracy increased when the RJ σ value increased, and when it reached 7 psec the prediction error reached the lowest level and stayed at this level until the value became 13 psec. From 13 psec to 30 psec, the prediction error started increasing, showing more uncertainty for the simulation. This is because two Gaussian distributions of random jitter at the left and right sides of the signal eye are convolved together if excessive random jitter exists. This effect reduces the sharpness of the slope for the distribution, and degrades the prediction performance. Since it did not show dependency on the amount of RJ σ for the range of 7 psec to 13 psec, we selected 10 psec RMS as a simulation set point.

Next, we experimented on the relationship between the number of transmitted bits and prediction accuracy. The result in Figure 11 shows that the prediction error decreases as the number of transmitted bits increases. Once it reached 500,000 bits, the prediction accuracy became stabilized. We selected 1 million bit sequences as a simulation set point to obtain an accurate result with reasonable simulation performance.

Using the selected simulation conditions, we performed a Monte-Carlo simulation with randomly generated ensembles to determine the repeatability of our scheme. 100 iterations of this simulation set were performed. Figure 12 describes the result of the simulation. Simulation conditions as well as resultant mean and standard deviation of the prediction RMS error are summarized in Table II. As shown in the results, our scheme can predict the DNLs with a mean prediction error of 0.31 LSB or 0.62 psec. 99.7% of the measurements will be within the error of $0.31 + 3 \times STD$ or 0.67 LSB if we assume that the measurement error distribution is Gaussian.

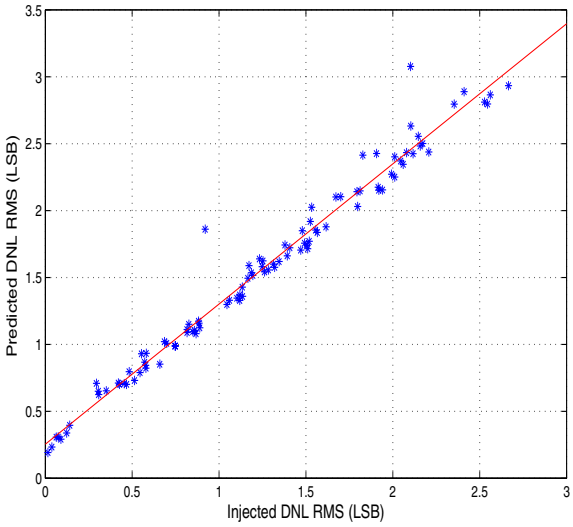

Fig. 12. Monte Carlo simulation of the proposed technique

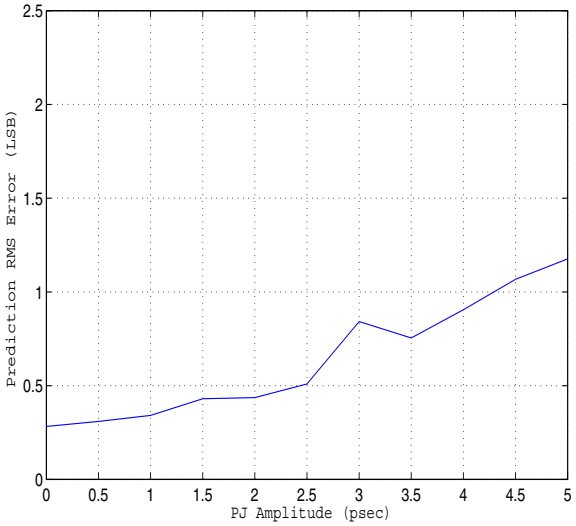

Fig. 13. Injected periodic jitter vs prediction error

Description	Value
Link Speed	10 Gbps
Num of Bits	1000,000
Injected RJ σ	10 psec RMS
Prediction Error Mean	0.31 LSB
Prediction Error STD	0.12
99.7% (3σ) Max Error	0.67 LSB

TABLE II
SUMMARY OF SIMULATION CONDITION AND RESULT

Another experiment was performed to observe the sensitivity of our scheme to periodic jitter (PJ). We modeled sinusoidal jitter with a frequency of 200 MHz and mixed it with an RJ σ of 10 psec RMS. With the same configuration of the serial link simulation, we plotted DNL prediction error for various amounts of PJ in amplitude while we fixed RJ σ at 10 psec RMS. Figure 13 shows the result; the prediction error increased slightly as more PJ was present. This result is expected as periodic jitter is a part of deterministic jitter which can be modeled as dual Dirac delta functions and degrades monotonicity of distribution D_{pos}. This consequently resulted in slight increase in the misprediction rate of the proposed algorithm.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed an efficient test technique for phase interpolators using random jitter injection. The proposed algorithm is cost effective in that it does not require hardware overhead in the case of forwarded clock scheme. For the derived clock scheme, we only need to implement one multiplexer in the clock lane to implement clock injection capability. The loadboard or system board only needs to contain a random jitter injector and a clock generator for undersampling purpose. The proposed algorithm accurately predicts the DNL of the phase interpolator. Simulation results show that our method accurately predicted nonlinearities of the PI. Since our method does not require significant circuit changes, it can be easily applied to various high speed I/O microarchitectures where PI is used.

We observed a slight increase in the misprediction rate when PJ is present. We will continue to improve our technique to be able to extract DNLs for complex jitter models.

ACKNOWLEDGEMENTS

We would like to thank Intel Corporation for tuition assistance for the first author's Ph.D. study.

REFERENCES

[1] Meixner, A.; Kakizawa, A.; Provost, B.; Bedwani, S., "External loopback testing experiences with high speed serial interfaces," *Test Conference 2008, IEEE International,* pp. 1-10, 28-30 Oct. 2008

[2] Casper, B.; Martin, A.; Jaussi, J.; Kennedy, J.; Mooney, R., "An 8-Gb/s simultaneous bidirectional link with on-die waveform capture," *Solid-state Circuits, IEEE Journal of,* vol. 38, no. 12, Dec. 2003

[3] Anderson, W., "High-speed I/O design," *High-Performance Energy-Efficient Microprocessor Design,* Ed. Vojin G. Oklobdzija and Ram K. Krishnamurthy, Springer US, pp. 289-309, 2006

[4] Li, M., "Jitter, noise, and signal integrity at high-speed," Prentice Hall, 2008

[5] Kreienkamp, R.; Langmann, U.; Zimmermann, C.; Aoyama, T.; Siedhoff, H., "A 10-Gb/s CMOS clock and data recovery circuit with an analog phase interpolator," *Solid-State Circuits, IEEE Journal of,* vol. 40, no. 3, pp. 736-743, Mar. 2005

[6] Bulzacchelli, J. et al., "A 10-Gb/s 5-Tap DFE/4-Tap FFE transceiver in 90-nm CMOS technology," *Solid-State Circuits, IEEE Journal of,* vol. 41, no. 12, Dec. 2006

[7] Provost, B., "Testability techniques for phase interpolators," US Patent Application no. 20090122849

[8] Shi, X.; Assaderaghi, F., "Phase linearity test circuit," US Patent Application no. 20070252735

[9] Benyahia, M.; Moulard, J.B.; Badets, F.; Mestassi, A.; Finateu, T.; Vogt, L.; Boissieres, F., "A digitally controlled 5 GHz analog phase interpolator with 10 GHz LC PLL," *Design & Technology of Integrated Systems in Nanoscale Era, 2007. DTIS. International Conference on* , pp.130-135, 2-5 Sep. 2007

[10] Jiang, Y.; Piovaccari, A., "A compact phase interpolator for 3.125G SerDes application," *Mixed Signal Design, Southwest Symposium on,* pp. 249-252, 23-25 Feb. 2003

[11] Sunter, S.; Roy, A., "On-chip digital jitter measurement, from megahertz to gigahertz," *Design and Test of Computers, IEEE,* vol. 21, no. 4, pp. 314-321, Jul.-Aug. 2004

[12] Dierckx, P., "Curve and surface fitting with splines," Oxford, England: Oxford University Press, 1993