Test of Phase Interpolators in High Speed I/Os Using a Sliding Window Search

Ji Hwan (Paul) Chun†‡, Siew Mooi Lim†, Shao Chee Ong†, Jae Wook Lee†‡ and Jacob A. Abraham‡

Intel Corporation†
Computer Engineering Research Center, the University of Texas at Austin‡
paul.chun@intel.com

***Abstract*— Conventional test for high speed serial links requires expensive test equipment to meet the standard $< 10^{-12}$ bit error rate (BER) requirement. Although timing margining loop-back tests are cost effective, phase interpolator (PI) circuitry needs to be tested for test completeness. Our method provides an efficient linearity test capability for the PI circuitry. In the proposed scheme, a sliding window search algorithm is used to extract differential nonlinearity (DNL) and integral nonlinearity (INL), based on a jitter distribution obtained from undersampling. Various simulations were performed to evaluate the accuracy and robustness of the method. They indicate that the proposed algorithm provides an accurate estimation of linearities of the PI. We also implemented our algorithm in a conventional low cost high volume manufacturing (HVM) tester platform to show feasibility and validity of the proposed technique.**

keywords - High Speed I/O, Phase Interpolator, Mixed Signal Test, HVM

I. Introduction

The growing demand for high performance computing has been driving the development of high speed I/O interfaces. These interfaces such as Serial ATA^{TM}(S-ATA^{TM}), PCI Express^{TM}, QuickPath Interconnect^{TM}(QPI^{TM}) and Fully Buffered DIMM^{TM}(FBD^{TM}) use a serial scheme which is known to be better in aligning timing skews and reducing the pin count. The serial interface architecture adopts schemes in which clock signals are either recovered from data or separately forwarded through another lane to align data and clock signals. Such an architecture, also known as serializer-deserializer (SerDes), places a unique challenge for production test of the interface; first, as the speed of interface increases, the cost of the automated test equipment (ATE) also increases. Second, signal integrity issues on the load board are more severe at higher speed, hence developing a reliable connectivity between a device under test (DUT) and the ATE is difficult and costly.

Moreover, process technology evolves to shrink device sizes, making devices more susceptible to process, voltage and temperature (PVT) variations. Due to this fact, the probability of having defects on the serial interface circuit continues to increase, which poses good return of investment (ROI) on the development of accurate and cost effective serial interface test methods.

Quality of the serial link is normally determined by the bit error rate (BER) metric. Most serial link applications require a BER of 10^{-12} or lower. Since measuring BER directly on the link requires long test time, research has focused on developing indirect measurement methods to estimate the BER at the 10^{-12} level. Various methods to estimate BER based on jitter statistics are widely used [1]–[4].

Loop-back based timing margining test [5] has been gaining popularity as a cost effective test methodology for high speed serial links. In this method, the transmitter (TX) output is wired to the receiver (RX) input and the timing of the recovered clock is either advanced or retarded with respect to the data signal to measure the margin of the data eye. Most timing margining test implementations re-use the phase interpolator (PI) circuitry which already exists in the SerDes architecture to align data timing. By enabling programmable PI capability, which can be achieved by multiplexing logic and access mechanisms, a timing margining feature can be implemented without major modifications to the SerDes architecture.

However, a timing margining test enabled by PI has its own limitation. Since the margining is required to move by an even distance, non-uniform steps in the PI could result in an incorrect assessment of the timing margin [6]. This can be translated to either false pass which results in test escape and increases defective parts per million (DPPM), or false fail which decreases yield by sacrificing good devices. There are a few previous papers in this area to mitigate the risk [7], [8]; most of them require either additional circuitry or major circuit modifications to measure the linearity of the PI. Chun et al. [6] propose a PI linearity test technique using random jitter injection. This method provides a PI linearity test capability without major circuit modifications; however, it still requires a random jitter injection source on the load board to properly characterize the PI linearity.

In this paper, a novel PI test technique is presented, which is an extension of the method described in [6]. Using the proposed algorithm utilizing intrinsic jitter, our new method does not require an additional random jitter injection. We demonstrate that our method works both in simulation and on a low cost high volume manufacturing (HVM) tester environment.

This paper is organized as follows. Section 2 reviews the basic theory behind the proposed method. The proposed

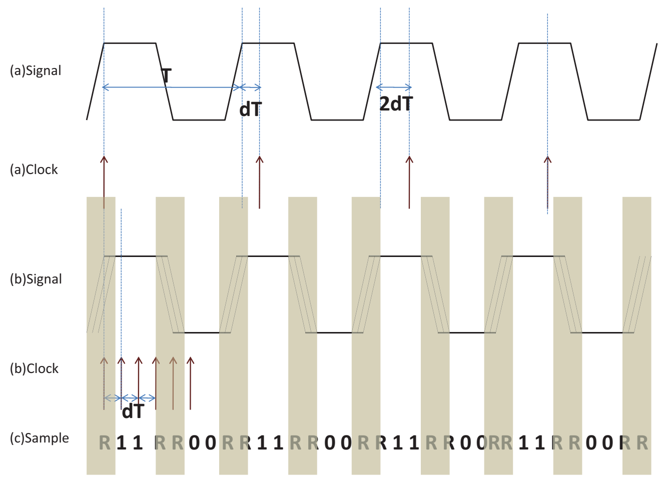
(a)Signal
T
dT
2dT
(a)Clock
(b)Signal
(b)Clock
dT
(c)Sample
R 1 1 R R 0 0 R R 1 1 R R 0 0 R R 1 1 R R 0 0 R R 1 1 R R 0 0 R R

Fig. 1. Undersampling Technique Concept

technique is described in Section 3. Experimental results are discussed in Section 4 for both simulation and hardware validation. In Section 5, we conclude the paper and discuss possible future work.

II. Preliminaries

A. Undersampling Technique Basics

For serial link testing, selection of the sampling frequency for the measured signal is challenging since the speed of the link is already high, and hence it is difficult to further increase the sampling frequency. Due to this challenge, undersampling is widely used in both ATE based [9] and on-chip based [10] methods. The concept of the undersampling technique is illustrated in Figure 1.

For a clock-like signal, e.g., a 1010 signal (measured signal), let T be the period of the measured signal. dT is selected so that the new sampling period $T + dT$ would result in a slightly lower sampling frequency as compared to the original one as shown in Figure 1 (a). Since the measured signal is periodic, the coherent undersampling results in strobe scanning of the measured signal with an effective resolution of dT as illustrated in Figure 1 (b). Due to the jitter present in the signal edge, the undersampling of the measured signal's jittery region (i.e. transition region) yields random 0s and 1s, which is described in Figure 1 (c). In the figure, 'R' denotes a binary value which can be either 0 or 1.

B. Jitter and BER

The bits sampled from the transition region create a stochastic distribution which can be denoted as a probability density function (PDF), $f_J(t)$. BER can be described as a function of f_J as follows.

$$BER(t_s) = \int_{t_s}^{\infty} f_J(t)dt \tag{1}$$

The signal edge can be considered to occur at t_s when $BER(t_s) = 0.5$ as long as the jitter aliasing is minimal. Now

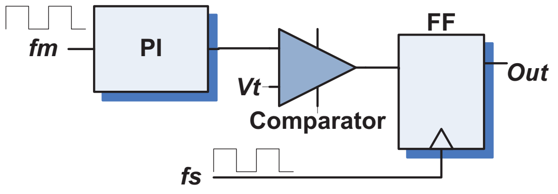

Fig. 2. Circuit Configuration Concept

for the undersampled bit sequence, $BER(t_s)$ can be re-written as a discrete function:

$$BER(t_n) = \sum_{k=t_n}^{\infty} f_J(k) \tag{2}$$

III. Proposed Technique

A. Test Procedure

The PI circuit's input/outputs are re-configured as conceptually depicted in Figure 2. An internal clock signal, f_m of period T, is supplied to the PI and delayed by the PI step configuration. The f_s signal is supplied, which has the period $T + dT$, externally to undersample the shifted signal f_m. A comparator input voltage, V_t is properly set so that the edge transition voltage is picked and sampled at the flip-flop (FF). The comparator and the FF need to be properly designed to operate at high frequencies. In order to find the estimated edge location, the sliding window search algorithm which we describe in the next subsection is used. After obtaining the estimated edge location, the PI step is advanced by one and the proposed procedure and algorithm are repeated to construct the estimated edge location array. From the estimation, the differential nonlinearity (DNL) and integral nonlinearity (INL) of each PI step are calculated.

B. Jitter Aliasing Reduction Algorithm Using Sliding Window Search

There exist related approaches to measure jitter and delays using undersampling. Before presenting our method, the difference among the related approaches is discussed. Huang and Cheng [11] propose a time interval measurement technique based on undersampling. Since the method aggregates the distribution in entire transition regions, jitter components are aliased, thus deteriorating the measurements. The authors indicate the jitter in sampling clock needs to be under a certain limit to obtain a desired accuracy on the time interval measurement.

The jitter aliasing problem between the measured signal and the sampling clock signal can be alleviated by various techniques. In order to measure jitter accurately, Sunter et al. [10] propose a median alignment technique to reduce the low frequency jitter induced by the undersampling clock. In this technique, rather than aggregating all transition regions, each transition region's median is first determined, then a jitter distribution is created based on the median aligned aggregation of the distributions. This technique requires either additional on-chip circuitry to determine the median value of each transition region, or post-processing of the entire bit stream to aggregate the distribution with the median alignment, which may be computationally expensive, when implemented in a low cost HVM ATE environment. Hong et al. [12] propose a method based on a root mean square calculation of individual standard deviations (σs) on the transition region. This method provides good accuracy on jitter measurement since the σs are first calculated based on each transition region; however it is mainly applicable to random jitter measurement rather than measuring linearity.

The aforementioned approaches focus on the individual transition regions to obtain an accurate jitter distribution by limiting the low frequency jitter aliasing effect. Our method focuses on the fact that adjacent samples are less susceptible to low frequency jitter. This translates to the idea that if the scope of edge location search is limited in a certain window, whose size can be either larger than the size of an individual transition region or smaller than that, it will result in a distribution which suffers less from low frequency jitter. We define a window for the calculation of the edge location as follows. For a bit position n in an integer, $b(n)$ is defined as

$$\begin{aligned} b(n) &= 1, \quad \text{for bit 1} \\ &= -1, \quad \text{for bit 0} \end{aligned} \tag{3}$$

A transition density function can be defined as follows.

$$F_{td}(n) = \sum_{i=-m}^{m} b(n+i) \tag{4}$$

where, m in integer is an empirical parameter to determine the window size. If the amount of jitter on the sampling clock is excessive, the window size can be decreased to avoid excessive aliasing. With the defined summation window, n is searched so that it satisfies $F_{td}(n) = 0$, which is an equivalent point of $BER = 0.5$. The proposed method is referred to as the sliding window search algorithm, since it works as if the summation window slides towards the right side as n increases when the undersampled bit stream is denoted from the left to the right.

Since both rising edge and falling edge can yield $F_{td}(n) = 0$, differential values were checked so that the solutions corresponding to rising edges are used for our technique. In fact, either rising or falling edges can be used as long as only one is used for consistency. This algorithm is simple and easy to implement in a low cost HVM tester environment where the size of the response capture memory is limited.

C. Interpolation Technique to Overcome Finite Resolution

There may be the case that the solution that yields $F_{td}(n) = 0$ does not exist. In this case, the closest solution to determine the edge location is the integer value n, which makes $abs(F_{td}(n))$ to be minimum. This is because the function is discrete in terms of the effective resolution dT. By reducing the value of dT, we can increase the resolution, and subsequently makes the transition density function more continuous. However, it has a tradeoff in that test time increases in lieu of the resolution improvement, since the number of samples increases. Physical hardware limitations might also exist for the undersampling clock generation in that the supported frequency of the signal generator cannot provide a finer effective resolution. Piecewise cubic polynomials can be used to fit the distribution into a continuous function. This interpolation technique provides benefits to overcome the finite resolution problem, while keeping the number of samples consistent. The piecewise cubic polynomial based on $F_{td}(n)$ is defined as

$$\text{For } i = 0, 1, 2, \ldots, n-1 \text{ and } t_i \le t \le t_{i+1},$$

$$F_i(t) = a_i(t-t_i)^3 + b_i(t-t_i)^2 + c_i(t-t_i) + t_i \tag{5}$$

For Equation 5, $4n$ conditions can be determined to have an analytic solution. The conditions are given as follows.

$$\begin{aligned} F_i(t_i) &= F_{td}(i), \quad \text{for all } i = 0, 1, 2, \ldots, n, \\ F_i(t_{i+1}) &= F_{td}(i+1), \quad \text{for all } i = 1, 2, 3, \ldots, n-1, \\ F'_{i-1}(t_i) &= F'_i(t_i), \quad \text{for all } i = 1, 2, 3, \ldots, n-1, \\ F''_{i-1}(t_i) &= F''_i(t_i), \quad \text{for all } i = 1, 2, 3, \ldots, n-1, \\ F''_0(t_0) &= F''_{n-1}(t_n) = 0. \end{aligned} \tag{6}$$

Due to non-monotonicity of the $F_{td}(n)$ as well as $F_i(t)$, there may exist multiple solutions that satisfy the conditions. Let $t_1, t_2, \ldots, t_k$ be the multiple solutions that satisfy the given conditions, and the final signal transition position L can be derived by averaging the solution values and multiplying by dT as follows.

$$L = \sum_j \frac{t_j}{j} \times dT \tag{7}$$

Now since the L is determined for one PI step position, PI step can be advanced by one and repeat the whole process. Let us define L_i the estimated edge location for the PI step i, and the DNL and the INL can be derived for step i as follows.

$$DNL_i = L_{i+1} - L_i \tag{8}$$

$$INL_i = \textstyle\sum_{k=1}^{i} DNL_k \tag{9}$$

IV. Experimental Results

Our technique was validated both in simulation and through hardware measurements. We simulated the DNL estimation part only for convenience of analysis, since INL measurement error can be calculated as aggregation of DNL measurement error. Both INL and DNL were measured for the hardware validation.

A. Simulation Results

Numerical simulation using MATLAB® was performed to validate the proposed algorithm. The parameters used in the models are listed in Table I. Uniformly distributed random DNL values were generated and injected to the PI model and DNL estimations were calculated and error of the estimation was reported. For comparison purposes, we developed the models for the two cases; 1) undersampling method along with our proposed sliding window search algorithm, and 2) the plain undersampling method where nonlinearity is calculated by aggregating entire distributions from all transition regions [11]. Three parameters were varied to understand the

Description	Value
Link Speed	2 Gbps
PI step size	5 psec
Undersampling dT	2 psec
Injected RJ σ	5 psec RMS
Injected PJ Amplitude	2 psec
PJ Frequency	200 MHz

TABLE I
SUMMARY OF SIMULATION CONDITIONS

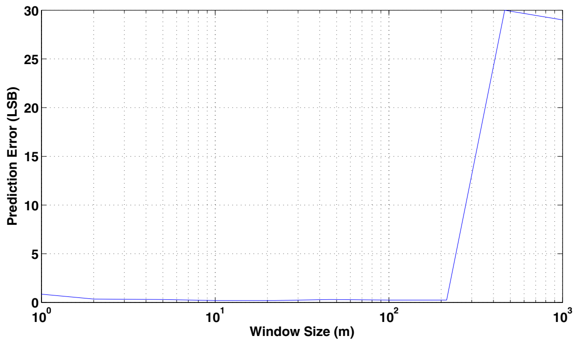

Fig. 3. Estimation Error vs. Size of Window

capability and limitations of our technique: the window size, m, number of sampled bits, amount of random jitter (RJ) in conjunction with periodic jitter (PJ). With the optimal values in each parameter, another simulation was performed with 100 iterations to understand repeatability.

1) Size of Window Sweep: Random DNL values were injected in the simulation and the corresponding estimated DNL values based on the proposed algorithm were calculated. The size of the window, m, was varied from 1 to 1,000 in log scale while we set the number of sampled bits as 10,000. Mean estimation errors of 10 iterations per data point were plotted with respect to the size of m as shown in Figure 3. As shown in the plot, the estimation error was minimum when m was between about 10 to 100 for the given simulation conditions. From this data, m was selected as 30 for the rest of simulations.

2) Number of Samples Sweep: In this simulation, the total number of samples was varied from 100 to 10,000 to determine the dependency on the parameter. For each iteration of the simulation, a DNL value was randomly generated then an estimated DNL was calculated using each algorithm. 20 iterations per each number of samples were made to obtain stable results. Figure 4 presents the estimation error for each data point of the number of bits. In general, both algorithm showed that the accuracy greatly increased once it sampled more than 1,000 bits. The proposed method based on the sliding window search yielded better accuracy as compared to the plain undersampling method.

3) Amount of Jitter Sweep: In this simulation, we set the amplitude of PJ to 10 psec across the simulation data point and varied RJ from 1 psec to 20 psec, which ensured to cover PJ and RJ component dominant cases. 10,000 bits were sampled and 20 iterations were made to select reliable data points. Figure 5 presents the result. Although there was a trend of a slight increase of estimation error when RJ was increased, the

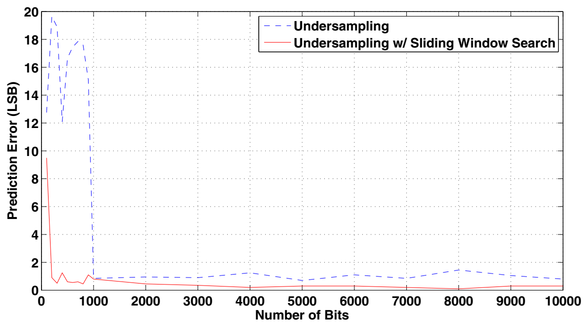

Fig. 4. Estimation Error vs. Number of Bits

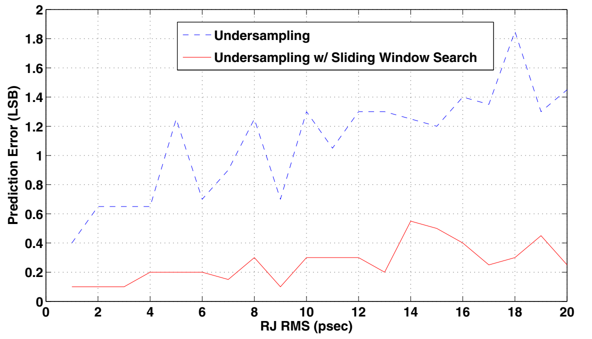

Fig. 5. Estimation Error vs. RJ σ

proposed algorithm again showed better accuracy as compared to the method that used undersampling only.

4) Repeatability Analysis: With the selected conditions, we ran the simulation 100 times to observe the repeatability of our method. Randomly generated DNL values were injected and corresponding estimated DNLs based on each algorithm were plotted to determine the repeatability. 10,000 bits were sampled to estimate the DNLs. Figure 6 presents the comparison between the undersampling method and our sliding window search algorithm. More outliers were observed in the plain undersampling method as compared to the proposed method since our method reduces the aliasing impact of injected jitters. Mean of DNL estimation errors and standard deviation (STD) of the errors are summarized in Table II. The results suggest that the sliding window search algorithm is more repeatable as compared to the plain undersampling method.

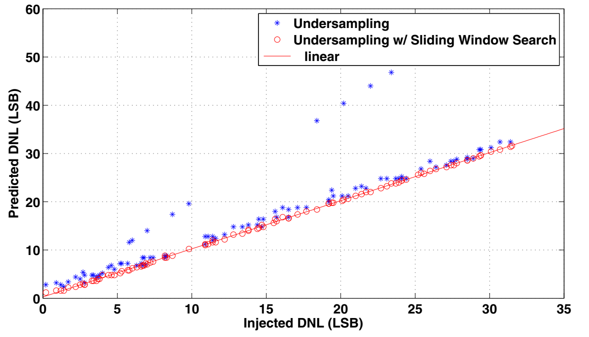

Fig. 6. Estimated DNL vs. Injected DNL

	Undersampling	w/ Sliding Window Search
DNL Mean Error	4.90	0.226
DNL Error STD	14.79	0.20

TABLE II
REPEATABILITY ANALYSIS RESULTS (LSB)

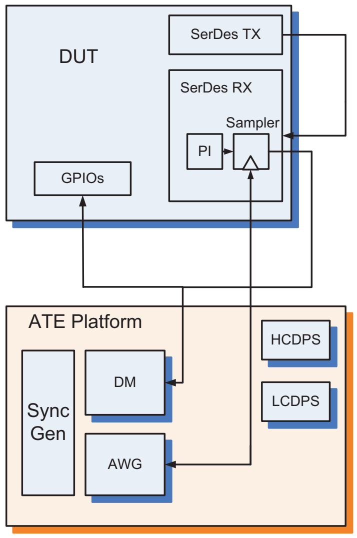

Fig. 7. Hardware Validation Configuration

B. Hardware Validation

Hardware measurement using the proposed method was performed using silicon with a 1.25 Gbps serial link circuit. The serial link was implemented with a PI based clock and data recovery (CDR) circuit which had 32 programmable PI steps. A design for testability (DFT) feature was implemented to enable the proposed technique.

The general HVM test configuration for the serial link was based on the loop-back test scheme in that the TX output is connected to the RX input to enable the timing margining based loop-back. During the PI test mode, multiplexers and test mode signals reconfigured the PI based CDR circuitry to have a proper undersampling setup, which is conceptually depicted in Figure 2.

Test content was developed using the proposed technique in a low cost production tester environment. The ATE configuration included a standard digital module that supports up to 667 Mbps input data rate and 500 Mbps output sampling rate for general production test purposes. An arbitrary waveform generator (AWG) was installed to enable the proposed technique by driving a proper clock signal with the period of $T + dT$ into the DUT. Figure 7 illustrates the hardware validation environment. The sampler circuit was implemented in a way that it operates in high speed and latches 1 if the input voltage is more than the threshold voltage, which is considered as the signal transition voltage. The digital module denoted as ‘DM’ was connected to general purpose I/Os (GPIOs) as well as the output of the sampler. The AWG drove the undersampling clock for the sampler. When creating the undersampling clock

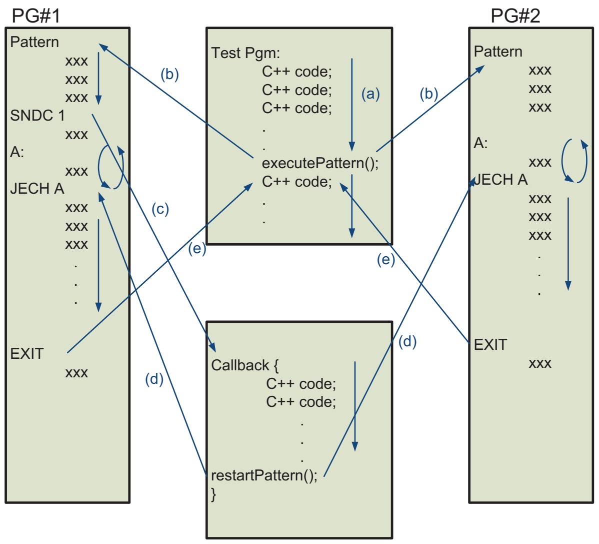

Fig. 8. Tester Pattern and Test Program Synchronization

waveform, up to 5th harmonics were used to generate a low jitter square waveform. Clock domains between two modules were synchronized by a sync generation module.

In order to develop tester patterns that fit in pattern memory requirements and properly synchronize the digital module with the AWG, the following pattern and test program handshaking architecture was proposed as illustrated in Figure 8. Each module had its own pattern generator which is denoted as PG#1 or PG#2. The actual sequence is enumerated from (a) through (e). Test program which was implemented in C++ had an initialization routine that was executed at the beginning of the test (a). The ‘execute pattern’ function in the test program invoked patterns in both domains and ran pre-conditioning patterns to enable the test mode on the DUT (b). A call back function (c) was introduced to handle tasks such as tester conditioning and initial result comparison that needed to be executed after the pre-conditioning patterns. While the call back function was being executed, the pattern from PG#2 was running in an infinite loop to wait for the call back function to complete. The call back function contained a ‘pattern restart’ function (d) that properly synchronized the two domains again to collect undersampled bit stream. The undersampled bit stream from the DUT was stored in vector memory and after the patterns were executed, it returned to the test program to post-process the data to extract the PI step edge location (e). The program repeated the sequence to construct the full PI step edge locations and extracted DNLs and INLs. Note that for the hardware validation, the resolution interpolation technique was not implemented to create a simpler test program.

At a nominal device supply voltage and room temperature condition, we performed 12 measurements on 1 DUT and plotted the PI step edge locations. A normalized graph with respect to the 32 locations is described in Figure 9. As shown in the figure, depending on the initial starting point of the undersampling, the estimated PI step location starts from a different location and the value wrap around when it reaches the maximum value. This fact does not impact the stability

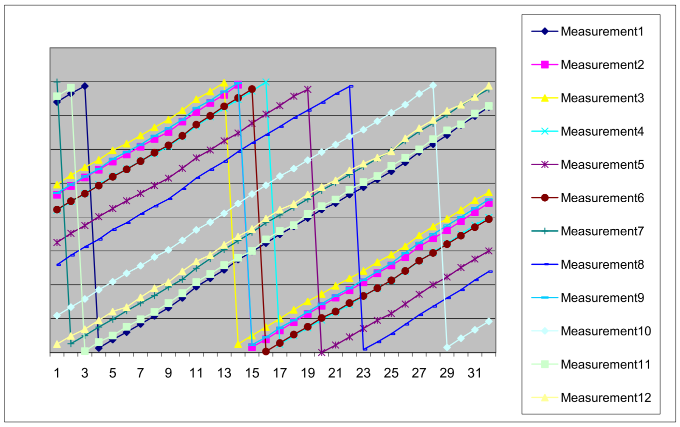

Fig. 9. PI Step Location Plot for 32 Positions

DUT ID	A	B	C	D
DNL Max Error	0.21	0.57	0.35	0.47
INL Max Error	2.90	1.16	0.89	0.66
DNL Error STD	0.16	0.24	0.20	0.17
INL Error STD	0.94	0.51	0.61	0.48

TABLE III
HARDWARE VALIDATION RESULTS (LSB)

of our algorithm, since DNL values are calculated based on differential values of the locations. For the cases where the edge location values were wrapped around, proper handling was implemented to yield correct DNL and subsequently INL estimations.

After collecting the 12 measurements, measured values were averaged to calculate the final DNLs and INLs for the device. The same sequence was repeated on 4 DUTs. The initial data collection on 4 DUTs suggested that the synchronization delay induced between ATE dual pattern generation operation and AWG instrument clock degraded the accuracy of the linearity test technique. By characterizing the offset delta induced by this delay and properly compensating the measured data by applying the offset, we obtained repeatable results which are summarized in Table III in a normalized manner.

Correlation activities were performed between system and tester result outliers. A few DUTs randomly failed when the proposed method was used in an HVM tester, although the same parts showed good linearities on a system board. The failure was observed as an abrupt jump of the calculated PI step location values for certain PI steps. Although the effort to minimize the configuration difference between the system board and the tester such as matching the board termination resistance did not yield the positive results, experiments showed that increasing the common mode voltage for the undersampling clock signal improved the correlation results. Further investigation identified that instrument noise from the tester pin electronics as well as the AWG degraded signal integrity of the undersampling clock and the resultant clock signal was marginal for certain DUTs in a random manner. Increasing common mode voltage improved signal to noise ratio (SNR) of the undersampling clock signal and resulted in elimination of the phase divergence issue. Table IV summarizes the difference between outlier results and corrected results after increasing the common mode voltage.

	Before	After
DNL Error	18.97	0.09
INL Error	53.48	0.20

TABLE IV
BEFORE & AFTER VOLTAGE CORRECTION RESULTS (LSB)

V. CONCLUSIONS AND FUTURE WORK

In this paper, a cost effective phase interpolator test technique is proposed. We demonstrate that the proposed method yields accurate measurement results in both simulation and HVM ATE environment. The results based on hardware validation show good correlation with actual system PI characteristics as well as good repeatability. The proposed nonlinearity estimation algorithm is efficient and concise in that it can be implemented in a conventional HVM tester. For future work, we will further validate the method in a larger number of DUTs to assess production worthiness of our methodology. We will also further evaluate the proposed method on higher frequency SerDes circuits to demonstrate the effectiveness of the method.

REFERENCES

[1] M. Li, *Jitter, noise, and signal integrity at high-speed*. Upper Saddle River, NJ, USA: Prentice Hall Press, 2007.
[2] Y. Cai, S. A. Werner, G. J. Zhang, M. J. Olsen, and R. D. Brink, “Jitter testing for multi-Gigabit backplane SerDes - techniques to decompose and combine various types of jitter,” in *Proceedings of the 2002 IEEE International Test Conference*, ser. ITC ’02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 700–709.
[3] D. Hong, C.-K. Ong, and K.-T. Cheng, “BER estimation for serial links based on jitter spectrum and clock recovery characteristics,” in *Proceedings of the International Test Conference on International Test Conference*, ser. ITC ’04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 1138–1147.
[4] T. Yamaguchi, M. Soma, M. Ishida, T. Watanabe, and T. Ohmi, “Extraction of instantaneous and rms sinusoidal jitter using an analytic signal method,” *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 50, no. 6, pp. 288 – 298, Jun. 2003.
[5] A. Meixner, A. Kakizawa, B. Provost, and S. Bedwani, “External loopback testing experiences with high speed serial interfaces,” in *Test Conference, 2008. ITC 2008. IEEE International*, Oct. 2008, pp. 1–10.
[6] J. H. Chun, J. W. Lee, and J. A. Abraham, “A novel characterization technique for high speed I/O mixed signal circuit components using random jitter injection,” in *Proceedings of the 2010 Asia and South Pacific Design Automation Conference*, ser. ASPDAC ’10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 312–317.
[7] B. Provost, “Testability technique for phase interpolators,” in *US Patent no. 7,907,661*, 2011.
[8] X. Shi and F. Assaderaghi, “Phase linearity test circuit,” in *US Patent no. 7,307,560*, 2007.
[9] W. Dalal and D. A. Rosenthal, “Measuring jitter of high speed data channels using undersampling techniques,” in *Proceedings of the 1998 IEEE International Test Conference*, ser. ITC ’98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 814–818.
[10] S. Sunter and A. Roy, “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.
[11] J.-L. Huang and K.-T. Cheng, “An on-chip short-time interval measurement technique for testing high-speed communication links,” in *Proceedings of the 19th IEEE VLSI Test Symposium*, ser. VTS ’01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 380–385.
[12] D. Hong, C. Dryden, and G. Saksena, “An efficient random jitter measurement technique using fast comparator sampling,” in *Proceedings of the 23rd IEEE Symposium on VLSI Test*, ser. VTS ’05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 123–130.