ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.14, No.15, 1–12

Acquisition accuracy enhancement of high-speed storage interface signals

Hyunwoo Kim[a)] **and Yong Ho Song**

Department of Electronics and Computer Engineering, Hanyang University, 222 Wangsimni-ro, Seongdong-gu, Seoul 04763, Korea

a) *hwkim@enc.hanyang.ac.kr*

Abstract: As storage interfaces have begun to employ high-speed signals and complex protocols, it has become increasingly difficult to ensure correct interactions between hosts and their storage. Correctness verification often requires the acquisition and thorough inspection of signals running at the interface. However, increases in interface signaling frequency may aggravate misalignment between sampling clocks and signals as well as among multiple signals, rendering signal acquisition and inspection difficult. To address this problem, this paper proposes a dynamic phase alignment scheme that can be used within a signal acquisition system. The proposed scheme was implemented on a Field-Programmable Gate Array (FPGA) board and was verified to successfully capture interface signals.

Keywords: storage interface, signal acquisition, phase alignment, mobile storage, eMMC, oversampling clock data recovery, acquisition accuracy

Classification: Integrated circuits

References

[1] K. Lee and Y. Won: "Smart layers and dumb result: IO characterization of an android-based smartphone," EMSOFT (2012) 23 (DOI: 10.1145/2380356.2380367).

[2] H. Kim and D. Shin: "Optimizing storage performance of android smartphone," ICUIMC (2013) 95 (DOI: 10.1145/2448556.2448651).

[3] N. Aparna, *et al.*: "Design of SD/eMMC protocol compliance solutions," IJAIS **7** (2014) 33 (DOI: 10.5120/ijais14-451215).

[4] Teledyne LeCroy Frontline: ComProbe SD 2.0 (2011) http://www.fte.com.

[5] Prodigy Technovations: PGY-SSM (2014) http://www.prodigytechno.com.

[6] J. M. Lin, *et al.*: "A 2.5-Gb/s DLL-based burst-mode clock and data recovery circuit with 4× oversampling," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **23** (2015) 791 (DOI: 10.1109/TVLSI.2014.2316553).

[7] A. Agrawal, *et al.*: "An 8 × 5 Gb/s parallel receiver with collaborative timing recovery," IEEE J Solid-State Circuits **44** (2009) 3120 (DOI: 10.1109/JSSC.2009.2033399).

[8] R. Ginosar: "Metastability and synchronizers: A tutorial," IEEE Des. Test Comput. **28** (2011) 23 (DOI: 10.1109/MDT.2011.113).

[9] Hardkernel: ODROID-XU4 (2015) http://www.hardkernel.com.

© IEICE 2017

DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

[10] Joint Electron Device Engineering Council: JESD84-B50 (2013) http://www.jedec.org.

1 Introduction

As performance demands with regard to storage systems have increased, high-speed interface protocols have been widely employed between hosts and their storage. Accordingly, it has become important to verify the correctness of storage operations. When storage operations are suspected to be faults, the host runs test programs to compare the results with what are expected [1, 2]. However, to precisely identify the root of a problem, it is sometimes necessary to thoroughly analyze signals exchanged at the storage interface. For this analysis, a series of interface signals were captured over a certain period and stored in permanent storage.

A variety of methods exist with regard to acquiring interface signals. One popular method uses commercial measuring instruments such as a logic analyzer or oscilloscope. These devices can capture and send interface signals to a PC for further analysis [3]. However, interface signals may experience different delays during transmission due to uneven electrical and mechanical characteristics of the signal paths to the measuring instruments. The instruments are thus unable to accurately acquire interface signals due to a lack of signal alignment. Another method involves using a *signal acquisition system* (*SAS*) tailored for specific storage interface signals [4]. In this approach, it is possible to capture interface signals and identify the compliance to protocol requirements. However, the *SAS* still suffers from alignment problems and thus it should perform phase alignment to correctly sense the signals, but it is difficult for the *SAS* to align the signals.

There are a few reasons for the aforementioned difficulties. First, signal alignment is usually performed between the host system and its storage device, not the *SAS*. In fact, the *SAS* taps interface signals in the middle of the physical communication path between the host and its storage (Fig. 1(a)). Interface protocols (e.g., the Embedded Multi Media Card (eMMC) 5.0) often provide command sequences, which are used to drive predefined data patterns on the interface; it allows the host to measure signal delays and negotiate signal phases (or delays) with the device. However, the *SAS* is not allowed to participate in such active negotiations because it should remain transparent at the interface. The *SAS* thus usually prepares its own *static phase alignment* to eliminate them [5]. During the *static phase alignment*, signal delays are separately measured and the *SAS* aligns the signals based on delays prior to signal acquisition. Second, the *SAS* should be able to deal with all possible delays or phase variants. Signal delays may differ in each host-device combination. Signal phases may fluctuate during runtime due to a variety of reasons, such as a change in the agent transmitting the signals or a speed mode change. Thus, the *SAS* needs to be able to *dynamically* adjust delays during signal acquisition. However, *static phase alignment* is fixed to a host-device combination and during runtime; therefore, it cannot cope with such variants. The clock and data recovery (CDR) schemes [6, 7] are worth considering, however they are usually

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

effective with regard to handling delays on only a single signal as opposed to multiple signals. Third, when the interface speed becomes faster, chances are that even a small delay may shift the signal to an incorrect cycle. If some parallel signals are aligned to neighboring cycles, data sensed during the same cycle can be interpreted incorrectly, although they are simultaneously driven from the host.

This study proposes a *dynamic phase alignment* (*DPA*) scheme to eliminate the delay differences in interface signals and to align signals to the same sampling clock edge for high-speed interfaces. The scheme dynamically aligns the phases during runtime; hence, the system can treat delay variations and phase changes. In fact, the scheme measures the delay of individual interface signals and adds an additional delay such that all the signals are well-aligned with the next sampling clock edge. This system was employed in the *SAS* for the automated interpretation of protocol flows and possesses three functional components: 1) *phase difference detection* to measure phase differences between the sampling clock and signals, 2) *signal phase adjustment* to shift the phase of each signal eye to the sampling clock edge based on measured phase differences, and 3) *cycle adjustment* to align the parallel signals to the same clock cycle. The *DPA* scheme was implemented in the hardware using an FPGA board and its operation was verified by capturing signals on a mobile platform with eMMC 5.0 storage.

2 Background

The *SASs* were used to obtain and analyze interface signals being transferred between the interface host and its device, as can be seen in Fig. 1(a). The source signals are driven simultaneously at the same interface clock edge (Fig. 1(b)). However, because the *SAS* typically uses its own internal sampling clock to acquire interface signals, the signal eyes could be poorly aligned with the sampling clocks. In this case, it would be difficult to guarantee setup/hold margins to the signals, making the signals sampled incorrectly at metastable states, as depicted in Fig. 1(c) [8]. Moreover, non-uniform delays may cause the signals to be aligned to different clock edges (Fig. 1(d)). When a signal frequency increases, any slight difference in phase or delay could be critical to the accuracy of signal capture.

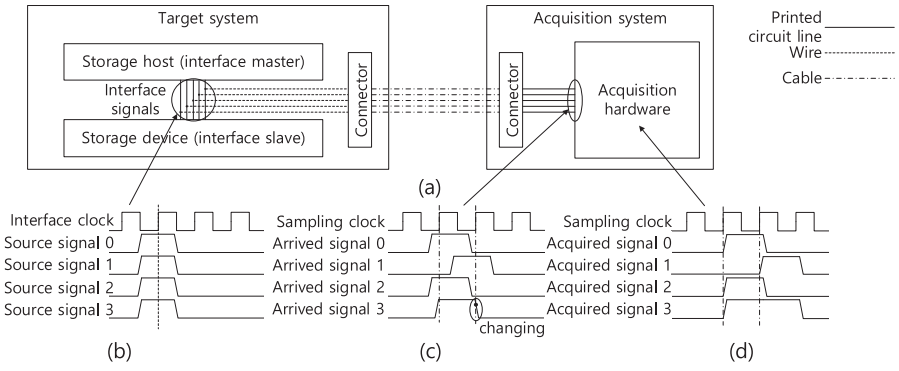

Fig. 1. (a) Signal acquisition system, (b) signal source, (c) arrived signal, and (d) acquired signal.

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

3 DPA for the signal acquisition system

Herein, a dynamic phase alignment scheme was proposed to enhance the acquisition accuracy of high-speed parallel signals. While conventional schemes require predefined data patterns to adjust signal phases or are fixed to a certain misalignment, the proposed scheme dynamically (1) measured phase differences, (2) aligned an individual signal to the sampling clock, and (3) synchronized signals transmitted in parallel. The system provided a solution to the alignment-related problems in *SASs*, which were due to the infeasibility of using signal alignment of protocols, delay variation among target systems, phase changes during runtime, and delay deviations among parallel signals.

3.1 *Phase difference detection*

In this step, the phase of signal edge was measured for each individual signal, representing the phase difference between the sampling clock edge and signal edge. To measure the phase of signal edge, the same signal was sampled using N different *phase-shifted sampling clocks* (*PSS clocks*) as illustrated in Fig. 2(a). Each *PSS clock* exhibited a phase shifted by $2\pi i/N$ rad ($i = 1, 2, \ldots, N$). Note that the *PSS clocks* were only used to measure differences. The *phase-shifted sampled signals* (*PSS signals*) were subsequently scanned to determine the phase in which the signal edge was present. If a signal transition took place during a certain clock period, it should be found within the *PSS signals* corresponding to that period, as can be seen in Fig. 2(b). The location of the signal edge was detected by scanning the *PSS signal* values in order of the smallest to largest shifted phases and by finding a change in the scanned value.

During an ideal case in which a signal immediately rose or fell with no slope, *PSS signals* before the signal edge read the same value, while those after the edge possessed an opposite value (see *PSS signals* 0–4 in Fig. 2(b)). However, it may take some time in the transition, and therefore signal metastability may have been observed near the signal edge. *PSS signals* captured near the signal edge were sometimes randomly read as either 0 or 1, as depicted in Fig. 2(c). This meant that the change in sampled values may have not correctly represented the signal edge. To correctly estimate the phase of the signal edge, the sampled values were listed as the clock phase increased and were scanned from both ends to locate the phase where the sampled values began to read differently ($0 \rightarrow 1$ or $1 \rightarrow 0$). Two such phases could exist from each scan direction. In this case, to estimate the phase of

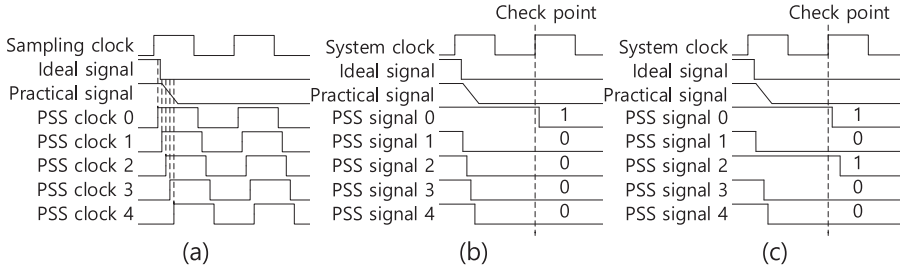

Fig. 2. (a) Sampling with phase-shifted sampling clocks, (b) acquired samples of an ideal signal, (c) acquired samples of a practical signal.

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

the correct signal edge, two phases were averaged to determine the median value. The measured phase of signal edge was delivered to the phase adjustment steps.

3.2 *Signal phase adjustment*

This step adjusted the signal eyes to the sampling clock edges by adding an extra delay to the signal. For this adjustment, the signal edges needed to be located π and $\pi/2$ rad from the sampling clock edge for Single Data Rate (SDR) and Double Data Rate (DDR) sampling, respectively. (Figs. 3(a) and 3(b)). To determine the necessary signal delay to adjust the signal edges, i.e., phase-adjustment value (P_a), the following simple equation, Eq. (1), was employed. Here, the phase of signal edge (P_s) measured during the *phase difference detection* step was used. Note that the phase of signal edge represented distance from the sampling clock edge to signal edge.

$$P_a = \lambda - P_s \tag{1}$$

In Eq. (1), λ is π and 3π rad when P_s falls into the range of $[0, \pi)$ and $[\pi, 2\pi)$, respectively, for SDR sampling. Likewise, λ is $\pi/2$ and $5\pi/2$ rad when P_s belongs to the range of $[0, \pi/2)$ and $[\pi/2, 2\pi)$, respectively, for DDR sampling.

After this adjustment, the signal eyes were subsequently placed at the right edge of the sampling clocks when the phase of signal edge was under π rad, and were placed at the next clock edge if the phase of signal edge was over π rad during SDR sampling. In other words, signals whose edges were in $[-\pi, \pi]$ were automatically adjusted to the same edge of the sampling clock. Signals whose phase differences were within a single cycle were matched with each other, i.e., cycle-aligned; however, signals shifted by more than one cycle were placed at the next clock edge, i.e., cycle-misaligned, as can be seen in Fig. 4(a).

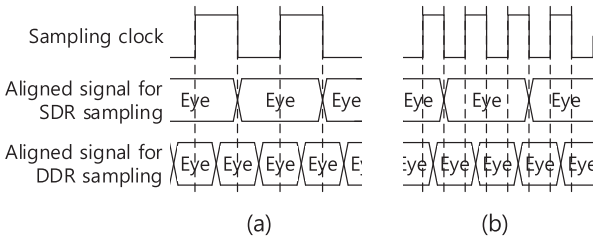

Fig. 3. Signal aligned to (a) 1x sampling clock and (b) 2x sampling clock.

3.3 *Cycle adjustment*

In this step, the signals were aligned to the same clock cycle with eyes individually phase-aligned to the sampling clock edge in the previous step. To achieve this, common bit patterns of the protocol were used. That is, when the host or interface device transmitted well-known start bits (or preamble bits) at the same time over all signal lines, the *SAS* individually detected the start bits and compared the detected bits with each other. In order to detect the start bits, phase-aligned signals were sampled in this step. Since the signals were phase-aligned to the edge of the sampling clock whose phase has not shifted, this sampling clock was used to sample signals. The position of the detected start bits over the signal wires was used for the

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

cycle adjustment: if there were start bits on the signals ahead of the others by one clock, the signals were intentionally delayed by one cycle, as depicted in Fig. 4(c). The start bits were transmitted simultaneously over the signal lines at the beginning of every packet transmission to indicate the packet start and synchronize the signals. Hence, it was reasonable to align parallel signals based on these start bits.

3.4 Example of dynamic phase alignment

A simple example of the proposed *DPA* is illustrated in Figs. 2(a), 2(c), 4(b), and 4(c). Fig. 2(a) depicts *phase difference detection*; note that only 5 of 16 *PSS clocks* are shown. The 16 *PSS clocks* were generated by delaying the sampling clock by increments of 22.5° (= 2π rad/16). The signal was sampled using *PSS clocks* and the sampled *PSS signals* were 1, 0, 1, 0, 0 for *PSS clocks* 0–4, respectively, as shown in Fig. 2(c). Since the signal exhibited a falling time spanning the rising edges of *PSS clocks* 2–4, as illustrated in Fig. 2(a), *PSS signals* 2–4 exhibited random values due to signal metastability. The sampled value began changing at *PSS signal* 2 and stopped changing at *PSS signal* 4; thus, the degree of phase shift of *PSS clock* 3, i.e., 67.5° (= 22.5° × 3), which was the median phase of *PSS clocks* 2 and 4, was presumed to be the phase of signal edge.

Fig. 4(b) illustrates the *signal phase adjustment* assuming the use of SDR sampling. Since the phase of signal edge was 67.5° and the sampling data rate was SDR, λ was 180° (= π rad). The phase-adjustment value was derived from Eq. (1) and was 112.5° (= 180° − 67.5). The signal eyes were shifted to the sampling clock edges by adding a delay equal to the calculated phase-adjustment value (112.5°), shifting the signal edge located at 67.5° to 180° from the sampling clock edge.

Fig. 4(c) demonstrates the misaligned state of signals from the *signal phase adjustment* step and their subsequent *cycle adjustment*. The start bits were assumed to be zeros. At the first rising edge of the sampling clock, the start bits were detected on signals 0 and 1 but not on signal 2. Thus, to align the three signals, signals 0 and 1 were delayed by one cycle until the start bit of signal 2 was detected.

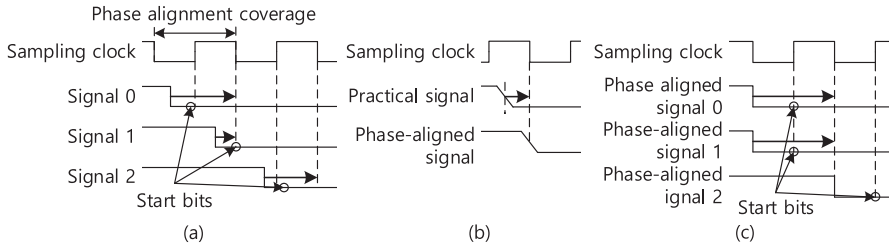

Fig. 4. For SDR sampling, (a) phase difference detection of signals, (b) signal phase adjustment, (c) cycle adjustment.

4 Hardware design of DPA

An acquisition system was designed to demonstrate the proposed *DPA* scheme. *DPA* implementation included *phase difference detectors* (*PDDs*), *signal phase adjusters* (*SPAs*), and a *cycle adjuster* (*CA*), as shown in Fig. 5(a). Each of the n parallel signals were fed into the associated *PDD* and *SPA*. The *clock phase shifter* generated 64 different *PSS clocks* by delaying the given sampling clock from $2\pi/64$ to 2π rad by increments of $2\pi/64$ rad. The *PSS clocks* were fed into each *PDD* and

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

were used to measure the phase of signal edge. The measured phases of signal edges were passed to the corresponding *SPAs* and were used to align the phases. The same system clock was used in both the *PDD* and *SPA*. The *PDD* used this clock to synchronize the *PSS signals*, while the *SPA* used it as a reference clock when delaying the signal. The n phase-aligned signals were passed to the *CA*, which latched the signals using the sampling clock and aligned the cycles of the latched signals.

The *DPA* was implemented on the FPGA platform. Delay modules embedded in the FPGA were used to shift the phases of the sampling clock and signals. Due to the specifications of the delay module, the number of phase-shifted sampling clocks used for the *PDD* was 64 and the delay step of *phase difference detection* and *signal phase adjustment* was 78 ps (i.e., 1/64 of a 5 ns period for a 200 MHz cycle). Additionally, the sampling frequency was set to 200 MHz due to the speed limit of the FPGA, and the range of the *CA* was constrained to two sampling clock cycles due to area constraints.

4.1 *Phase difference detector (PDD)*

To measure the phase of signal edge, the *PDD* sampled the interface signal using 64 *PSS clocks* and found the *PSS signal* representing the signal edge from 64 *PSS signals*. For this, the *PDD* included an array consisting of 64 flip-flops (FFs), a *synchronizer*, and a *phase encoder*, as depicted in Fig. 5(b).

The 64 FFs of the array were synchronized to the corresponding *PSS clocks* and were used to sample the signals. In order to scan the 64 FFs, the outputs of the FFs were transferred to the *synchronizer* and were synchronized to the system clock. The *synchronizer* possessed two FFs for each *PSS signal* to remove signal metastability by considering the mean time between failures [8]. The resulting 64 synchronized *PSS signals* were brought to the *phase encoder* in order to quantify the phase of the signal edge.

The *phase encoder* is depicted in Fig. 5(e) and detects the signal changes by XORing consecutive samples. The logical *high* of the XOR output meant that the signal was changed at the phase where the input sample was captured. Thus, by using the XOR results, the priority and reverse priority encoders could find the phases of the first and last signal changes, respectively. Afterwards, both the first and last edge phases were encoded to a 6-bit value ranging from 0 to 63 and were passed to an *intermediate value calculator*. The calculator then computed the median of the two values, which was the phase of signal edge.

4.2 *Signal phase adjuster (SPA)*

The *SPA* derived the phase-adjustment value and adjusted the signal by adding a delay, which is equal to the adjustment value, to the signal. To this end, the adjuster consisted of a *phase-adjustment value calculator* and a *signal delay adder*, as illustrated in Fig. 5(c).

The *phase-adjustment value calculator* computed the phase-adjustment value by using Eq. (1) with the 6-bit phase of signal edge received from the *PDD*. Since Eq. (1) is a simple subtraction process, the value could be calculated with a 7-bit subtractor; 4:1 and 2:1 multiplexors (MUX) were additionally required to select λ,

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

as shown in Fig. 5(f). The value of λ was determined by the sampling data rate and the range where the phase of signal edge was located. Therefore, λ was selected using a data rate (DR) signal, indicating the data rate, and the fifth and sixth bits of the phase of signal edge, distinguishing the range based on $\pi/2$ rad (encoded phase 16 (= 2^5)) and π rad (encoded phase 32 (= 2^6)), respectively. The resulting 6-bit adjustment value was then transferred to the *signal delay adder*.

The *signal delay adder* shifted the signal by the amount of the delivered adjustment value. Since the delay module that was used was only capable of delaying the signal from 0 to π rad, the *signal delay adder* contained two cascaded delay modules, a *front and rear phase delay module*, as depicted in Fig. 5(g), allowing signal delays up to 2π rad. The *delay compensator* preceding the *front phase delay module* consisted of several delay modules and could add additional signal delays to make up for the cycles consumed in the *PDD* and *SPA*. The *delay compensator* also offset the intrinsic delay of the *front and rear phase delay modules* by regulating the degree of added delay. The degree of added delay to both the *front and rear phase delay modules* as well as the *delay compensator* were obtained from a *delay amount generator*, making appropriate delay taps to the delay modules based on the phase-adjustment value.

4.3 *Cycle adjuster* (*CA*)

The goals of the *CA* were (1) to sample the phase-aligned signals, (2) to detect start bits from the sampled signals, and (3) to align the cycle-misaligned signals to the same clock edge, by selectively delaying the signals by one full cycle based on the detected start bits. As mentioned above, the extent of *CA* available for our design was up to two cycles and phase misalignment among signals within a single cycle was cycle-aligned in the *SPA*; hence, the *CA* only aligned signals by one additional cycle.

To this end, the *CA* had a *cycle selector* shared by all signals and both a *sampling FF* (*SFF*) and *delayed-sample FF* (*DSFF*) for each signal, as shown in Fig. 5(d). The *SFF* captured the phase-aligned signal using the sampling clock whose phase was not shifted and the *DSFF* stored the sample delayed by one cycle. The *cycle selector* detected the start bits of all signals by separately investigating the *SFF* and *DSFF* for each signal. When a start bit was detected in any of the signals, the *cycle selector* chose the delayed samples of the signal, *DSFF*, in which a start bit was detected and the non-delayed samples of the signal, *SFF*, in which no start bit was detected as a cycle-aligned signal. If the start bits were detected in any signal and if a signal misalignment was observed within one cycle, the start bits must be present in the next cycle for the remaining signals. Therefore, the *CA* could be performed by simply delaying the signals by one cycle, in which the start bits were detected.

To support any misalignments exceeding one cycle in the *CA*, *DSFFs* should be added proportionally to the number of required cycles, which would require additional logic. Therefore, our design only supported alignment within two clock cycles: one cycle at the *SPA* and another at the *CA*. In fact, this was reasonable considering the length of wire used for tapping and the signaling speed of the interface signals.

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

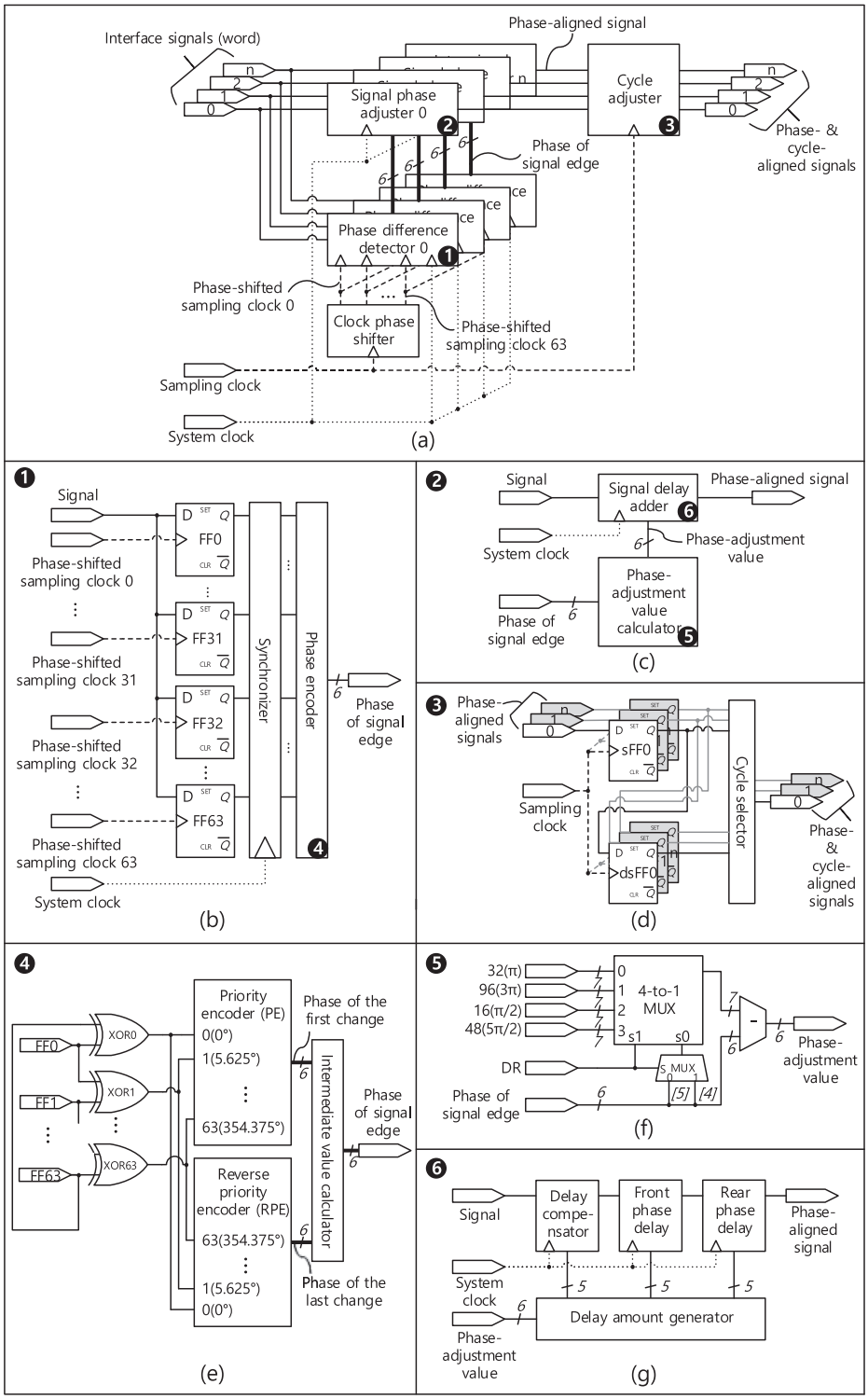

Fig. 5. (a) Overall *DPA* architecture, (b) phase difference detector, (c) signal phase adjuster, (d) cycle adjuster, (e) phase encoder, (f) phase-adjustment value calculator, (g) signal delay adder.

5 Experiment

5.1 Experimental setup

To verify the effectiveness of the proposed *DPA*, an *SAS* (Fig. 6(a)) was implemented with the *DPA* hardware design using the Xilinx Zynq-7000 FPGA. The *SAS* could capture and process signal traces for further analysis. The FPGA device was mounted on a board (right in Fig. 6(b)) and connected to the host-device

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

interface wires for tapping interface signals as depicted in Fig. 6(b). The target system was an ODROID-XU4 [9] (left in Fig. 6(b)), including a Samsung Exynos 5422 application processor (AP) and a Toshiba eMMC 5.0 module. Although the eMMC 5.0 interface [10] supports signaling speeds up to 400 Mbps through the HS400 mode, our experiment ran at 333 Mbps due to restrictions on the IO PLL of the AP. The *SAS* aimed to tap the DAT0–DAT7 data lines between the AP and eMMC module. Although the eMMC standard specifies to sample the data lines using the DS (Data Strobe) or CLK (CLocK) signal, our system sampled the data lines using a self-contained sampling clock. This was because the DS and CLK are conditionally provided, and these signals are thus improper to be used as clock sources for capture logic. The data lines were sampled at both edges of the sampling clock, indicating DDR sampling.

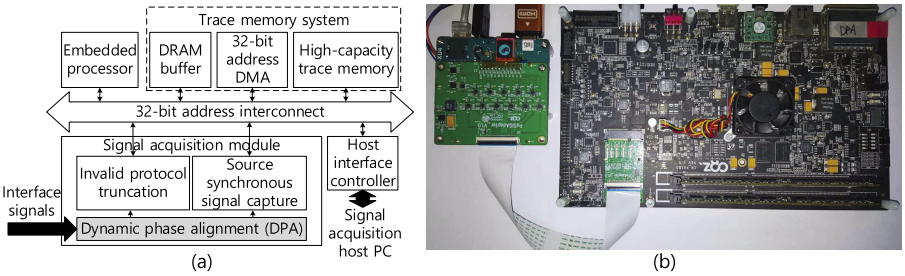

Fig. 6. (a) Signal acquisition system and (b) signal acquisition environment.

A synthetic workload was used in the experiment, which was generated by issuing specific combinations of eMMC commands using a modified MMC host test driver of a Linux kernel (v3.10) running on the AP. The workload consisted of six transaction types: single read & write, pre-defined read & write, and open-ended read & write, which were used in the HS400 mode. To transfer data on the signal lines at 333 Mbps, a checkerboard pattern consisting of alternating 0x55 and 0xAA was used as the data payload. Additionally, to model the occurrence of signal delays which could lead to misalignment between the signals and sampling clock as well as misalignment among parallel signals, three system models were configured by adding arbitrary extra delays at the FPGA input port where the signals are delivered, as can be seen in Table I.

Table I. Extra delays added to signals of the target system (ns).

System Model	DAT0	DAT1	DAT2	DAT3	DAT4	DAT5	DAT6	DAT7
Model 1	0	0.75	2.813	0.188	5.813	1.688	5.063	3.75
Model 2	2.438	2.625	0.563	4.688	1.688	3.563	0.938	2.625
Model 3	1.313	5.813	2.813	2.25	4.5	0	4.875	0.188

5.2 Experimental results

To assess the effectiveness of the proposed *DPA* on the acquisition accuracy, signal acquisition was performed on three system models with the proposed *DPA* turned on or off. The acquisition errors were detected by comparing transmitted data with the obtained signals. Fig. 7 shows the bit error rates (BERs) of signals obtained from the three models using this acquisition system. Without the *DPA*, BERs for

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

the *SAS* ranged from 37.77% to 79.24% with an average of 54.92%. Although the standard deviation of the BERs was relatively high, errors were frequently observed in all models. The read and write transactions exhibited different error trends because the agents transmitting data on the signal lines during read and write transactions are different. By contrast, the *DPA*-enabled *SAS* accurately acquired all signals for Models 1–3, yielding BERs of 0% for all transaction types and models.

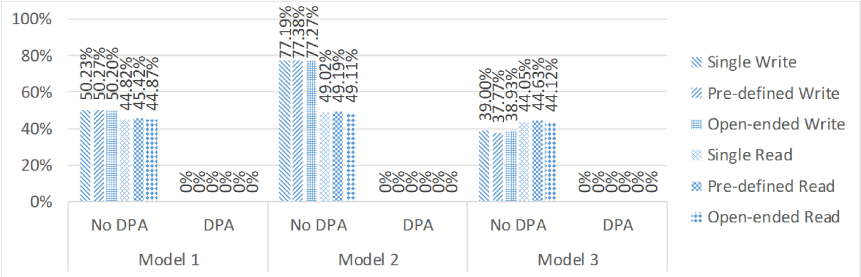

Fig. 7. BERs for signal acquisition with and without *DPA*.

The experimental results of Model 1 captured without the *DPA* were further analyzed to separately observe the influence of delays on each bit and the parallel signal acquisition. Table II shows the phase-error rates and cycle-error rates; the former were obtained by individually analyzing the samples acquired from each signal line and the latter were measured by comparing the appearance of the DAT*x* start bit and that of DAT0 because the start bit of DAT0 appeared first in Model 1. As shown in the table, many phase-errors occurred at DAT0 and DAT7. This was because these signals changed near the sampling clock edge, thus they were captured in a metastable state. The other signals were believed to escape such edges due to extra delays. On the other hand, cycle-errors were observed largely on DAT1, DAT2, and DAT5 since these were shifted from DAT0 and thus captured at the next cycle. Cycle-errors were barely measured on DAT4 and DAT6 although these signals were largely shifted from DAT0. This was because these signals were delayed by 2 cycles and were read the same due to the repetitiveness of the checkerboard pattern. If other non-regular patterns were used, the cycle-error rate would increase significantly for these signal lines. Although the analysis results are presented for Model 1, there were no significant differences observed for Models 2 and 3.

If more *PSS clocks* were employed, in other words, if the precision of *DPA* was increased, delays could be detected and controlled in lower granularities. In this vein, the effects of *DPA* precision with regard to acquisition accuracy were evaluated by performing signal acquisition for Models 1–3 while varying the degree of precision. The average BERs for Models 1–3 at various levels of precision can be seen in Fig. 8, in which the values listed on the *x*-axis represent the degree of precision as a fraction of the sampling clock cycle. When the degree of precision is high, the acquisition results are expected to be more accurate. The BER was 54.92% on average when the *DPA* was not applied (No *DPA*), which decreased to 11.7% on average for the precision level of $1/4$, i.e., $\pi/2$ rad. The BER was 0.0009% on average for the precision level of $1/8$ and 0% for all precision levels smaller than $1/8$, meaning that all errors were removed. From this experiment, it was concluded that signals transferred at a signaling speed of 333 Mbps

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017

could be obtained flawlessly at a precision level of 1/16; it could be expected that higher-speed signals could be accurately acquired with greater precision.

Table II. Error rate breakdown of each signal for the write transaction of Model 1 (%).

	Single write			Pre-defined write			Open-ended write		
	Phase-error	Cycle-error	Total	Phase-error	Cycle-error	Total	Phase-error	Cycle-error	Total
DAT0	11.9	0.0	11.9	14.8	0.0	14.8	11.9	0.0	11.9
DAT1	0.7	98.6	99.3	0.0	99.9	99.9	0.6	98.7	99.3
DAT2	0.0	100.0	100.0	0.0	100.0	100.0	0.0	100.0	100.0
DAT3	0.3	0.0	0.3	0.0	0.0	0.0	0.3	0.0	0.3
DAT4	0.1	0.1	0.2	0.1	0.1	0.2	0.1	0.1	0.2
DAT5	0.1	99.7	99.8	0.1	99.7	99.8	0.1	99.7	99.8
DAT6	0.1	0.1	0.2	0.1	0.1	0.2	0.1	0.1	0.2
DAT7	9.8	80.3	90.1	12.7	74.4	87.2	10.1	79.8	89.8

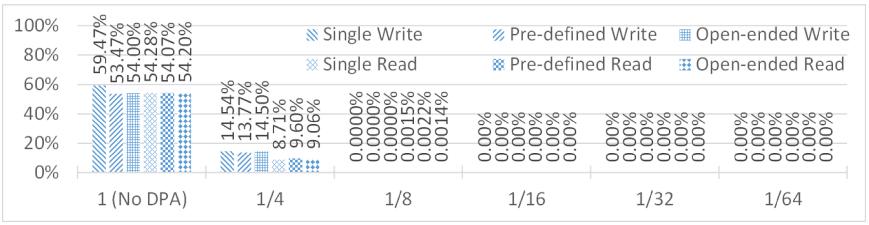

Fig. 8. Average BERs of Models 1–3 for various levels of *DPA* precision.

6 Conclusion

In this study, a *DPA* scheme to enhance the accuracy of acquiring high-speed parallel signals was proposed for signal acquisition systems. The proposed *DPA* improved accuracy by dynamically measuring the phase difference from interface signals, aligning signal eyes to the sampling clock edges, and aligning parallel signals based on the start bits. The hardware implementation of the proposed *DPA* scheme was also presented herein; the sampling system including the *DPA* hardware was implemented based on an FPGA. Experiments using the implemented acquisition system demonstrated 0% BER for all types of transactions and target system models when eMMC 5.0 interface signals transferred at 333 MHz were acquired, showing that the proposed scheme eliminated acquisition errors caused by phase misalignment. The acquisition accuracy drastically improved as the precision of the proposed *DPA* increased. It is expected that high-speed interface signals over 333 MHz can be acquired without errors by increasing the degree of precision.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science, ICT and Future Planning) (No. NRF-2015R1A2A1A01002716).

© IEICE 2017
DOI: 10.1587/elex.14.20170634
Received June 16, 2017
Accepted July 10, 2017
Publicized July 24, 2017
Copyedited August 10, 2017