High-Speed Interface IP Validation based on Virtual Emulation Platform

Jaehun Lee
jaehunn.lee@samsung.com

DaeSeo Cha
dscha@samsung.com

Sungwook Moon
sw2013.moon@samsung.com

Samsung Electronics Co., Ltd.
1-1, Samsungjeonja-ro, Hwaseong-si, Gyeonggi-do, Korea

***Abstract* – As the complexity and the speed of HSI (High Speed Interface) IPs are rapidly increasing with every generation, it is becoming harder to functionally validate the HSI IPs within given schedules. To achieve this, various methodologies are utilized to completely validate the HSI IP. One of commonly used methods is Field Programmable Gate Array (FPGA). However, using an FPGA still has some challenges such as visibility, high effort to implement design and high Turn Around Time (TAT). This paper suggests virtual emulation platform to address the above problems. Virtual emulation is used to test the design under test (DUT) with OS-level applications on a virtual machine and has the advantages of full-blown emulation like full-visibility, real scenario and fast speed.**

Keywords - Emulation, Virtual Emulation, Validation, Verification, High-speed interface, HSI IP, SoC, Software, Virtual machine

I. Introduction

Current electronic devices require large data storage to store numerous data and perform necessary calculations quickly. Most of them communicate with other devices through various interfaces instead of standalone communication. This requirement of high-speeds leads to wide usage of HSI IPs in modern mobile Application Processor (AP) designs. The recent trend has seen HSI IPs being widely used in various system-on-chip (SoC) products such as automotive, artificial intelligence (AI), high performance computing (HPC) and Internet of Things (IoT) devices.

Therefore, to design competitive SoCs, the HSI IP needs to be completely validated prior to SoC design. As the complexity and the speed of HSI IP are rapidly increasing with every generation, it becomes harder to completely validate the functionality within a project schedule. Traditionally, FPGA is commonly used for HSI IP validation [1]. However, FPGA inherently has a challenge to achieve the real speed of HSI IP [2]. Also, FPGA-based validation platform requires complete implementation of the design and additional multi project wafer (MPW) test chips including analog circuitry of physical layer for HSI IP validation [1]. We can start doing the functional validation in parallel with developing software (S/W) codes only once MPW test chips are physically available. Those problems become the main bottlenecks for HSI IP validation in terms of project schedule and quality of product.

This paper suggests virtual emulation platform (VEP) to address the above problems. The concept of virtual emulation is to connect a hardware emulator and virtual machine using virtual interface and test the DUT with specific applications on a virtual machine. VEP can be used at a pre-silicon stage and it can develop target S/W codes and make the functional validation with OS-level scenarios by leveraging proven device driver running on the virtual machine. In this work, we demonstrate that the VEP approach can reduce TAT by 33.3% and achieve 19.7% higher functional coverage compared to traditional methods.

II. THE CONCEPT OF VIRTUAL EMULATION

Emulation is widely used for the system-level validation of modern SoC design at a pre-silicon stage. Many companies validate their SoC using emulation with firmware and, by extension, with a specific application like day of use (DOU) scenario. However, one of the biggest challenges of IP-level validation is how to take test with more realistic scenarios such as OS-level applications with device driver of that IP. The virtual emulation technology is targeted for the above challenges and needs a software adapter that enables user applications and OS drivers to establish a virtual protocol connection to hardware emulator. Figure 1 shows the concept of virtual emulation and the red box indicates the software adapter, which is the key solution of virtual emulation technology.

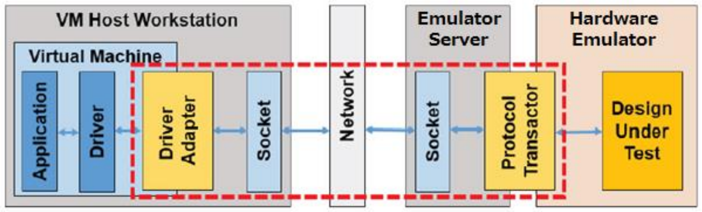


Figure 1. Concept of Virtual Emulation

Figure 2 shows the detailed architecture of the software adapter. The software adapter means the combination of the DEV1 and DEV2 on the left hand side and the DEV1 and DEV2 on the right hand side. An application in the virtual machine is used to interact with the driver to open the concerned virtual device and stream in the protocol transactions. And the driver which is proven in the OS communicates with the virtual device that is not an actual hardware but a virtual prototype. The DEV1 and DEV2 on the left hand side are called driver adapter, as specified in figure 1. This driver adapter intercepts the transactions going to the virtual device, and sends the data as packets to the DEV1 and DEV2 on the right hand side. The protocol transactor is composed of DEV1 and DEV2, C proxy and interface transactor (xtor) and it connects via the protocol to the DUT in the hardware emulator.

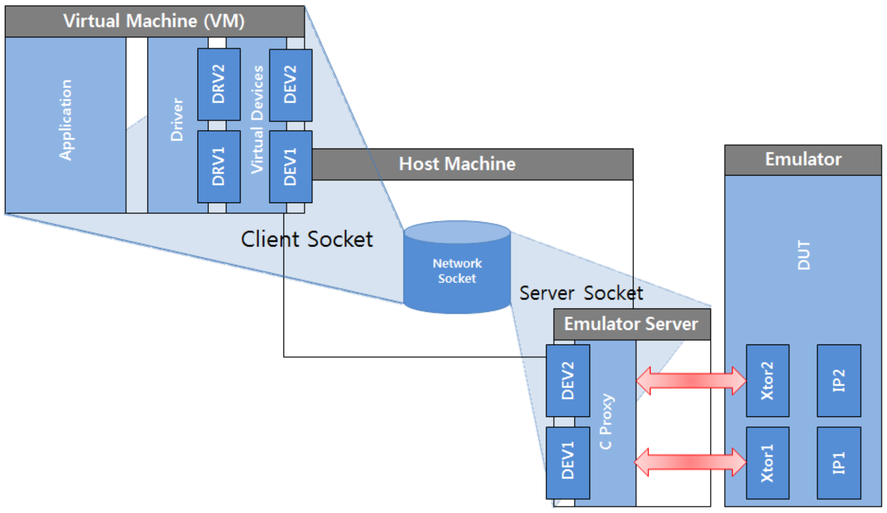


Figure 2. Virtual Emulation software adapter architecture

III. HIGH SPEED INTERFACE IP VALIDATION FLOW

A. *Existing HSI IP validation flow*

Figure 3 shows an example of HSI IP validation flow based on FPGA with a MPW platform. The platform is configured with mini-SoC having a PCIe Link controller on the FPGA side, and a physical layer (PHY) in a MPW test chip. The mini-SoC is optimized for executing target S/W codes for HSI IP and consists of an ARM CPU, system interconnect, memory and peripherals like a UART interface for S/W debugging. Physical media attachment sub-layer (PMA) in PHY has many analog circuits and hence, PHY is provided in a separate MPW test chip connected through physical coding sub-layer (PCS)/PMA interface with FPGA. For validating PCIe Endpoint with this platform, back-to-back connection is required. This means that through the back-to-back connection, checking compatibility of PCIe protocol is impossible.

In this approach, a MPW test chip has to be fabricated and it usually takes about 10 weeks, which is critical to meet a time-to-market requirement. In the meantime, simulations can be used to partially replace the method. However, it cannot manipulate all of the features due to the speed limitation of simulations. Also, the platform has a difficulty in achieving the full speed of PCIe Gen3 by the nature of FPGA.

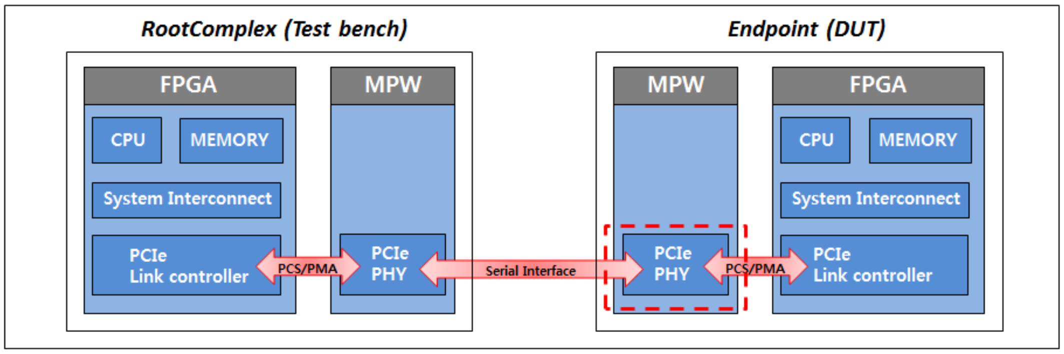


Figure 3. Previous Work: FPGA with a MPW platform

B. *Proposed HSI IP validation flow*

The proposed VEP also has mini-SoC on a hardware side and virtual machine with Linux OS including virtual device on a virtual side. Unlike the existing platform, the synthesizable PHY model covering the PMA is used instead of a MPW test chip and it can communicate with the PCIe Link controller. However, the synthesizable PHY cannot handle analog features like equalization and lane margining. Additionally, a protocol transactor is newly added in the hardware side for communicating with the virtual device through a virtual interface. Figure 4 shows the proposed VEP for validating a PCIe Gen3 IP.

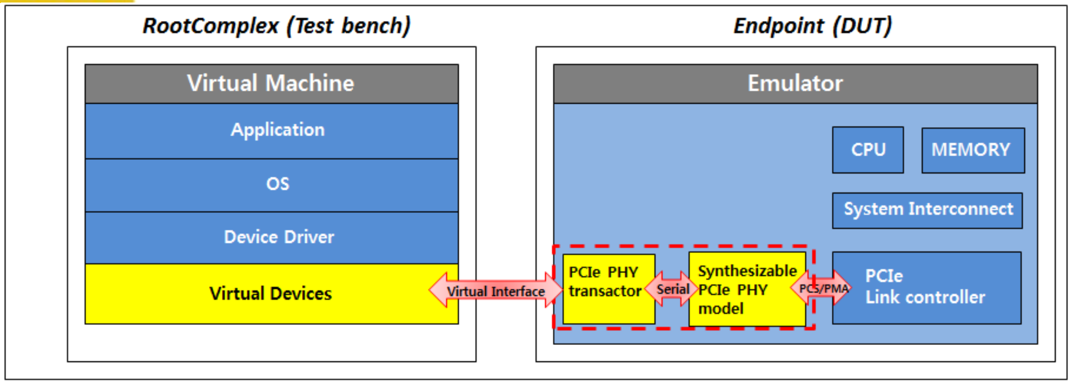


Figure 4. Proposed Work: Virtual Emulation Platform

The proposed VEP has three advantages compared with the existing approach. 1) VEP does not require the MPW test chip. Thus the VEP enables us to develop target S/W codes early because the VEP is available at a pre-silicon stage. In addition, we can efficiently make target S/W codes more robust by leveraging a real device driver on a virtual machine. 2) VEP can reinforce the functional verification by performing the applications on OS and accelerating the simulation speed using hardware emulator. Through the VEP approach, we can test many of the scenarios of interest combined with inbound/outbound transactions including massive data, power management and aging tests which cannot be achieved in simulation. 3) VEP can measure the performance of an HSI IP in a real benchmark program such as IOmeter with the full speed of a PCIe Gen3 IP. The existing platform cannot run at-speed of the HSI and will always be slower than target speed. However, the VEP has no such limitations and can run at the required speed of a HSI IP. Due to this functionality, we can check whether it complies with the effective bandwidth of PCI Express Base specification Revision 3.0 8.0GT/s. In addition to above advantages, VEP has flexibility to debug DUT and S/W with fully static environment which means there are no timeout constraints. Thus, S/W engineers can freely stop the DUT whenever they want. Lastly, the VEP environment can be easily replicated to other projects, which enables us to shorten TAT of similar projects.

IV. METHOD TO ACCELERATE PRE-SILICON VALIDATION

Once the VEP environment is ready, we can proceed with functional validation and develop target S/W codes to meet the target HSI IP speeds. If the main purpose of the VEP is to validate our PCIe Gen3 IP, then the interface between the virtual machine and the DUT should use the serial interface through the physical layer. However, if the main purpose of the VEP is S/W development, there is another method to accelerate the emulation performance which is detailed below.

To accelerate the emulation performance, using the PIPE (PHY interface for the PCI Express) interface has been chosen. The PIPE interface is intended to enable the development of functionally equivalent PCI Express PHY's. The PIPE specification defines a set of PHY functions which must be incorporated in a PIPE compliant PHY, and it defines a standard interface between such a PHY and link layer of PCI Express with Serializer/Deserializer (SerDes) PIPE. Thus, the PIPE interface uses parallel data transactions instead of the serial interface using Tx/Rx channel. With the real speed of HSI IP over 5Gbps, run time performance of emulation is degraded because the emulator is capturing events at every clock cycle. Using a fast clock inevitably causes capturing a lot of events. In case of PCIe Gen3 IP, the serial interface uses a clock frequency of 8GHz, whereas the PIPE interface uses a clock frequency of 250MHz, when data width is 32bit. That means the serial interface captures 32 times more data compared to the PIPE interface. The longer the scenarios, the gap of clock frequency makes a huge difference on their run time performance of emulation. Figure 5 shows the VEP with the PIPE interface

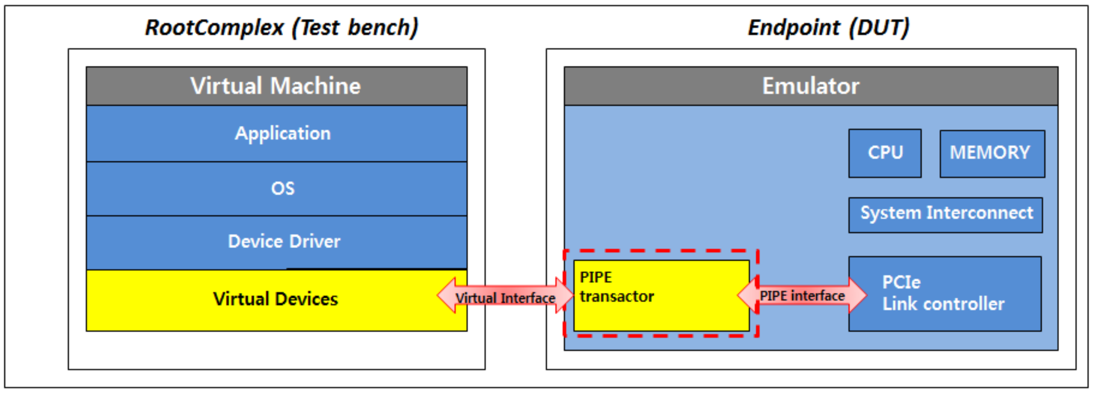


Figure 5. Virtual Emulation Platform with PIPE interface

Therefore this paper suggests the mixed interface methodology for S/W development. The first step is integrating the PIPE transactor instead of the PCIe PHY transactor. Using the PIPE interface does not require a synthesizable PHY model. This means building a VEP is faster using the PIPE interface compared to the serial interface. This makes it possible to develop basic firmware and device drivers. After the S/W is ready, the synthesizable PHY model and PCIe PHY transactor are integrated to the VEP to validate the IP with developed S/W including massive data transaction and power aging test. This methodology effectively enables us to reduce the TAT for HSI IP validation. Figure 6 shows the overall flow of the mixed interface methodology. This approach is requires higher engineering efforts. For using the mixed interface methodology, two emulation environments are required for validation. One is the DUT with PIPE transactor and the other is the DUT with PHY transactor so that the integrating method is different in its own environment.

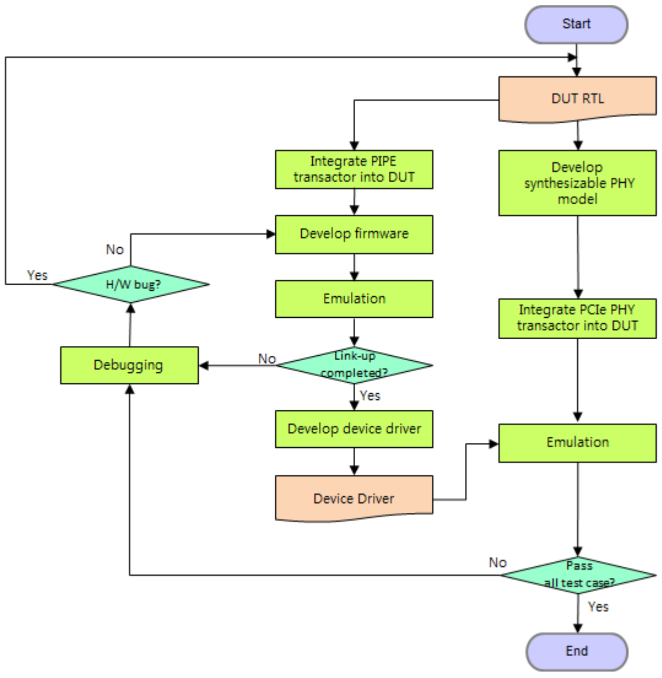


Figure 6. Mixed interface flow

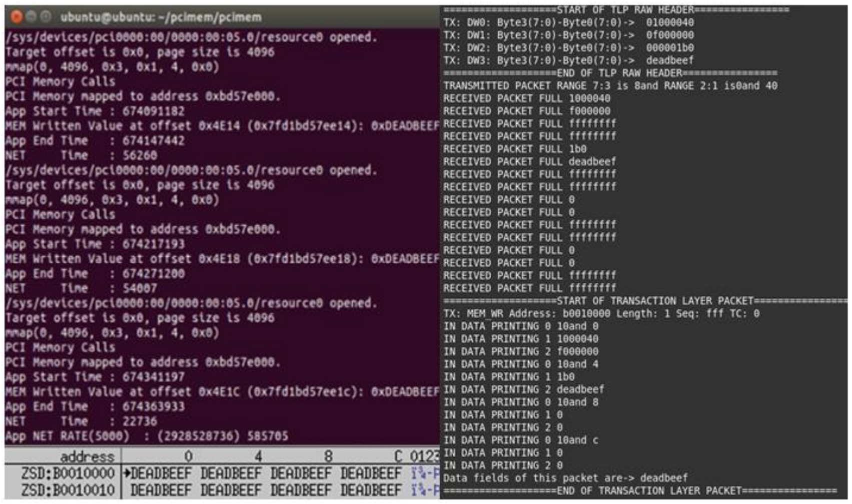
Table 1 compares the performance for the PCIe Gen3 link-up and loading the VM with 4096 bursts transactions, according to the type of interface. Regarding the scenario for the PCIe Gen3 link-up, as we estimated above, the result shows the PIPE interface reduced the simulation time by a factor of 31.35 times compared to the serial interface, and the wall clock time performs even faster than estimated. It can be helpful for debugging the link equalization process during the link-up at Gen3 speed, as well as testing the basic functions, and developing the firmware. Even if the reduction ratio of 4096 bursts transaction driven virtual machine is decreased in comparison with the Gen3 link-up, it is acceptable because the bottleneck is loading the OS in the virtual machine and not related to hardware.

Table 1. Performance comparison

Scenario / Interface	Gen3 Link-up		Loading VM and Transaction (4096 burst)	
	Simulation Time (ms)	Wall clock time (sec)	Simulation Time (ms)	Wall clock time (sec)
Serial Interface	62.7	3149.76	100	5006.75
PIPE Interface	2 x31.35	19.38 x162.5	18 x5.56	379.28 x13.2

V. HOW TO DEBUG IN VIRTUAL EMULATION PLATFORM

The VEP methodology has different debug methods based on the type of error (protocol error, data error) which help to ease the difficult task of developing target S/W or debugging complex IP. To analyze protocol-level transactions, the VEP can use log files generated by the socket on the server side. The log file dumps transaction layer packet (TLP) information of the data which contains the type of transaction, address, length, and data/header field of the TLP. Thus, S/W engineers can use the TLP information for debugging their code with the software debugger virtually connected to the hardware emulator through the virtual joint test action group (JTAG) interface.



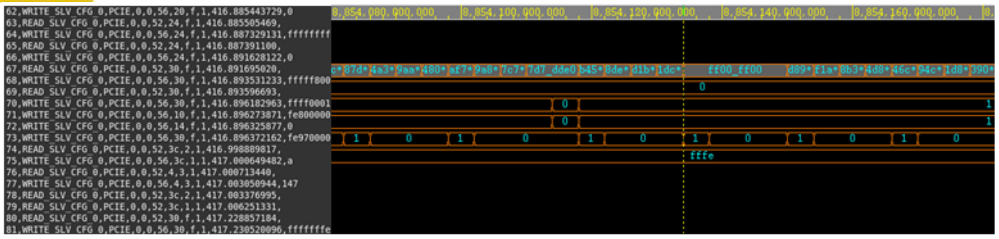
```
/sys/devices/pci0000:00/0000:00:05.0/resource0 opened.
Target offset is 0x0, page size is 4096
mmap(0, 4096, 0x3, 0x1, 4, 0x0)
PCI Memory Calls
PCI Memory mapped to address 0xbd57e000.
App Start Time : 674091182
MEM Written Value at offset 0x4E14 (0x7fd1bd57ee14): 0xDEADBEEF
App End Time    : 674147442
NET     Time    : 56260
/sys/devices/pci0000:00/0000:00:05.0/resource0 opened.
Target offset is 0x0, page size is 4096
mmap(0, 4096, 0x3, 0x1, 4, 0x0)
PCI Memory Calls
PCI Memory mapped to address 0xbd57e000.
App Start Time : 674217193
MEM Written Value at offset 0x4E18 (0x7fd1bd57ee18): 0xDEADBEEF
App End Time    : 674271200
NET     Time    : 54007
/sys/devices/pci0000:00/0000:00:05.0/resource0 opened.
Target offset is 0x0, page size is 4096
mmap(0, 4096, 0x3, 0x1, 4, 0x0)
PCI Memory Calls
PCI Memory mapped to address 0xbd57e000.
App Start Time : 674341197
MEM Written Value at offset 0x4E1C (0x7fd1bd57ee1c): 0xDEADBEEF
App End Time    : 674363933
NET     Time    : 22736
App NET RATE(5000)  : (2928528736) 585705
```

```
        address |    0        4        8        C  0123
 ZSD:B0010000 |→DEADBEEF DEADBEEF DEADBEEF DEADBEEF ï¾-F
 ZSD:B0010010 | DEADBEEF DEADBEEF DEADBEEF DEADBEEF ï¾-F
```

```
===================START OF TLP RAW HEADER=================
TX: DW0: Byte3(7:0)-Byte0(7:0)->  01000040
TX: DW1: Byte3(7:0)-Byte0(7:0)->  0f000000
TX: DW2: Byte3(7:0)-Byte0(7:0)->  000001b0
TX: DW3: Byte3(7:0)-Byte0(7:0)->  deadbeef
===================END OF TLP RAW HEADER=================
TRANSMITTED PACKET RANGE 7:3 is 8and RANGE 2:1 is0and 40
RECEIVED PACKET FULL 1000040
RECEIVED PACKET FULL f000000
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL 1b0
RECEIVED PACKET FULL deadbeef
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL 0
RECEIVED PACKET FULL 0
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL 0
RECEIVED PACKET FULL 0
RECEIVED PACKET FULL ffffffff
RECEIVED PACKET FULL ffffffff
===================START OF TRANSACTION LAYER PACKET=================
TX: MEM_WR Address: b0010000 Length: 1 Seq: fff TC: 0
IN DATA PRINTING 0 10and 0
IN DATA PRINTING 1 1000040
IN DATA PRINTING 2 f000000
IN DATA PRINTING 0 10and 4
IN DATA PRINTING 1 1b0
IN DATA PRINTING 2 deadbeef
IN DATA PRINTING 0 10and 8
IN DATA PRINTING 1 0
IN DATA PRINTING 2 0
IN DATA PRINTING 0 10and c
IN DATA PRINTING 1 0
IN DATA PRINTING 2 0
Data fields of this packet are-> deadbeef
===================END OF TRANSACTION LAYER PACKET=================
```

Figure 7. S/W debugging method

For H/W engineers, the VEP records the transactions flowing between the virtual machine and the DUT using the CSV format, both on the server and client side. The CSV file has the type of transaction such as configuration, read/write and address/data of the transaction with timestamp. The timestamp helps the H/W engineer to locate points of interest in the waveform and also easily dump waveforms at the specified point of interest.



```
62,WRITE_SLV_CFG_0,PCIE,0,0,56,20,f,1,416.885443729,0
63,READ_SLV_CFG_0,PCIE,0,0,52,24,f,1,416.885505469,
64,WRITE_SLV_CFG_0,PCIE,0,0,56,24,f,1,416.887329131,ffffffff
65,READ_SLV_CFG_0,PCIE,0,0,52,24,f,1,416.887391100,
66,WRITE_SLV_CFG_0,PCIE,0,0,56,24,f,1,416.891628122,0
67,READ_SLV_CFG_0,PCIE,0,0,52,30,f,1,416.891695020,
68,WRITE_SLV_CFG_0,PCIE,0,0,56,30,f,1,416.893531233,fffff800
69,READ_SLV_CFG_0,PCIE,0,0,52,30,f,1,416.893596693,
70,WRITE_SLV_CFG_0,PCIE,0,0,56,30,f,1,416.896182963,ffff0001
71,WRITE_SLV_CFG_0,PCIE,0,0,56,10,f,1,416.896273871,fe800000
72,WRITE_SLV_CFG_0,PCIE,0,0,56,14,f,1,416.896325877,0
73,WRITE_SLV_CFG_0,PCIE,0,0,56,30,f,1,416.896372162,fe970000
74,READ_SLV_CFG_0,PCIE,0,0,52,3c,2,1,416.998889817,
75,WRITE_SLV_CFG_0,PCIE,0,0,56,3c,1,1,417.000649482,a
76,READ_SLV_CFG_0,PCIE,0,0,52,4,3,1,417.000713440,
77,WRITE_SLV_CFG_0,PCIE,0,0,56,4,3,1,417.003050944,147
78,READ_SLV_CFG_0,PCIE,0,0,52,3c,2,1,417.003376995,
79,READ_SLV_CFG_0,PCIE,0,0,52,3c,1,1,417.006251331,
80,READ_SLV_CFG_0,PCIE,0,0,52,30,f,1,417.228857184,
81,WRITE_SLV_CFG_0,PCIE,0,0,56,30,f,1,417.230520096,fffffffe
```

Figure 8. H/W debugging method

VI. RESULTS

The proposed VEP was applied to the validation of the latest PCIe Gen3 IP. We were able to achieve 19.7% coverage increase and 33.3% TAT reduction. Table 2 shows the coverage comparison between the previous platform and the proposed VEP. The previous platform covered only 68.1% of total validated space because it had many limitations to exercise all the PCIe Gen3 features. This meant that we had to verify the other features with simulation. But the VEP can almost cover the PCIe Gen3 features and its coverage up to 87.8%. The remaining area is an analog characteristic like link equalization and lane margining which the VEP cannot access. It can be only covered by a platform which uses FPGAs and a MPW.

Table 2. Coverage of PCIe Gen3 IP validation

Feature	Previous Work			Proposed Work		
	All	Completed	Coverage	All	Completed	Coverage
Normal operation (LTSSM, Speed Change, Interrupt, Bifurcation)	143	105	73.4%	143	120	83.9%
Transaction (Inbound, outbound, configuration, DMA)	38	23	60.5%	38	38	100%
Power Management (Aging, ASPM, PM)	33	11	33.3%	33	28	90.3%
Compliance	40	34	85%	40	37	92.5%
Total	254	173	68.1%	254	223	87.8%

Figure 9 shows the achieved coverage and time according to the platforms. The proposed approach is that both simulation and simulation acceleration were used for sanity checking in the beginning and then the VEP was used for early S/W development and validation for 6 weeks. Finally, FPGA with a MPW platform was applied to validate the remaining features with target S/W codes proven at the VEP and OS-level scenarios. As a result, we were able to successfully demonstrate 33.3% TAT reduction by relying on VEP.

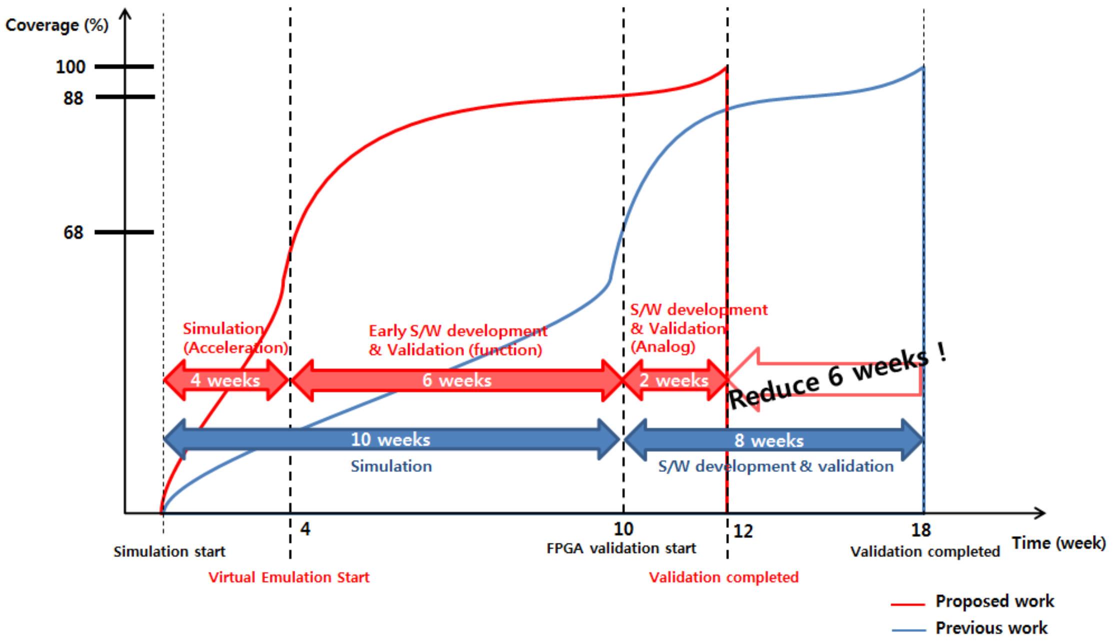


Figure 9. Time and Coverage of each Platform

Ⅴ. CONCLUSION AND FUTURE WORK

An innovative VEP (Virtual Emulation Platform) methodology is proposed for the validation of High-Speed Interface IPs in this paper and successfully demonstrated the achievement for all the desired objectives. Using the proposed approach enabled us to develop target S/W codes and take OS-level tests with the full speed of a HSI IP at a pre-silicon stage. Through the validation of the PCIe Gen3 IP, the VEP significantly improves finding design issues at early stage and achieves higher functional coverage up to 87.8%. Furthermore, it is expected to reduce the TAT of a HSI IP validation using the VEP. VEP has unique benefits such as flexibility, debug capability with static environment and accessibility at early stage. This approach will be complementary approach with using in-circuit emulation. In-circuit emulation can achieve high-performance test with real board-level environment. By combining those two solutions at each of design stage, it is expected to greatly improve ease of designing high quality and error-free products.

This proposed solution using VEP is still at the beginning stage to validate HSI IP. Many HSI IPs are required to verify their analog characteristic like equalization and lane margining in the validation stage. Therefore, functional verification and validation are focused on the simulation stage before silicon. However, the simulation cannot cover long test scenarios such as massive data transaction and power aging test because of the simulation speed [3]. If a functional error occurs in silicon, an engineering change order (ECO) is required but it is not easy to find whether the error is caused by H/W or S/W. By using VEP, the problem will be handled at a pre-silicon stage and error in silicon can be debugged [3]. Many engineers have depended on silicon or FPGA for HSI IP validation, but if this technology is utilized, VEP can solve most of their concerns at the early stage.

Even though this solution handles functional validation, it cannot handle analog characteristics of the IP which is also an important metric. If emulation environments can provide the analog handling technique or modeling, the tool can be even more powerful and improve coverage to almost 100%. To make this possible, close collaboration between electronic design automation (EDA) vendors and industry is required which can be that subject of future research on this area.

References

[1] M. K. Pandey, S. Shekhar, A. Sinha and A. Mishra, "An FPGA Based Ecosystem for USBPHY Validation, "*2014 15th International Microprocessor Test and Verification Workshop*, Austin, TX, 2014, pp. 44-48.

[2] J. Qingzhong and W. Xingdou, "Research on high-speed communication technology between DSP and FPGA," *2016 7th IEEE Control and System Graduate Research Colloquium*, Shah Alam, 2016, pp. 62-66.

[3] A. K. Sinha, A. Pal and A. Bisht, "DDR3: Bridging emulation to validation," *2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT)*, Ghaziabad, 2017, pp. 1-5.