IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

DEVELOPMENT OF VIP FOR PCI EXPRESS PROTOCOL IN OPEN-POWER CORE PROCESSER BASED FABLESS SOC

[1] Mr. Manikanta Thella , [2] Mr. M. Sreedhar, [3] Dr. G. Mamatha

[1] M.Tech Scholar, [2] Assistant Professor Adhoc, [3] Assistant Professor

[1,2,3] MTech in VLSI-SD,

[1,2,3] JNTUA College of Engineering Ananthapuramu, Ananthapuramu, India

Abstract **-** The high-speed serial data bus standard named PCI Express stands for Peripheral Component Interconnect Express. PCIe offers a range of direct linkages that let several devices communicate with one other at fast data rates, with plenty of capacity, and across numerous lanes. A computer can be connected to one or more peripheral devices using the serial expansion bus standard known as Peripheral Component Interconnect Express (PCIe or PCI-E). Each hardware item that is linked to a motherboard via a PCIe link has a unique point-to-point connection. The test bench can be fitted with verification IP (VIP) blocks that can be used to replicate the design (either an IP or an SoC) and check its functionality. Due to the rising complexity of system-on-chip (SoC) designs, this is becoming more and more crucial. Multiple providers of a design project and stages of the design cycle might use VIP. sample of reusable intellectual property is verification IP, which may provide thorough tests to speed up SoC verification and broaden test coverage. It's normal practice to validate standard bus protocols using verification IP**.**

Index Terms **– System Verilog, UVM Testbench, PCIe.**

I.INTRODUCTION

The best aspects of previous-generation bus topologies have been carried over into PCI Express, and developments in computer architecture have also been utilized. The use model and load-store communication paradigm used by PCI Express are the same as those used by PCI and PCI-X. Common transactions including memory read/write, IO read/write, and configuration read/write transactions are permitted by PCI Express.Similar to PCI and PCI-X, the memory, IO, and configuration address space models are used. By maintaining the address space notion, existing operating systems and driver software will run without a hitch in a PCI Express machine. In other words, software created for PCI and PCI-X computers may run on PCI Express platforms. In reality, a PCI Express card can run an existing OS without requiring any changes to the drivers or software programs. The PCI/ACPI power management software will continue to work.

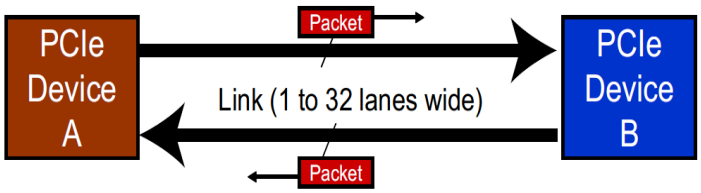

Figure – Dual Simplex Link

A Link, which consists of one or more transmit and receive pairs, is the name of this line of communication between the devices. A Link can be built up of 1, 2, 4, 8, 12, 16, or 32 of these pairs, each of which is referred to as a Lane. The Link Width, often regarded as the number of lanes, is denoted by the symbols x1, x2, x4, x8, x16, and x32. The trade-off between the number of lanes to be employed in a particular design is simple: more lanes enhance the Link's bandwidth but raise its price, take more space, and use more power.

II.PCI-E ARCHITECTURE OVERVIEW

1 Topology –

A CPU is located at the top of the figure. Here, it's important to note that the CPU is regarded as being situated at the very top of the PCIe hierarchy. Similar to PCI, PCIe only permits basic tree architectures; loops and other intricate topologies are not supported. This is done to preserve backward compatibility with PCI software, which did not handle diverse environments and utilized a straightforward setup approach to track the topology. Make sure your template is the appropriate size for your paper first. This template has been designed to print on A4-sized paper. Please dismiss this file and download the Microsoft Word, Letter file if you plan to publish on US letter-sized paper. Software must be able to produce configuration cycles in the same way as previously, and the bus topology must seem the same as it did before, to safeguard compatibility.

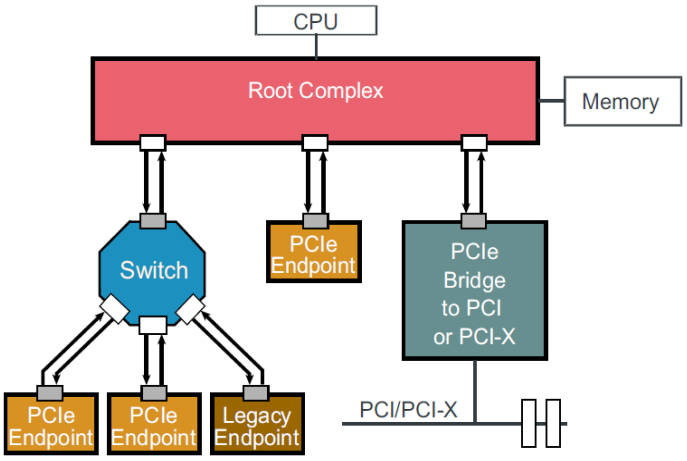

Figure – PCIe Topology

1.1 Root Complex:

The processor interface, Memory interface, and other components, in addition to possibly numerous chips, may be installed at the PCIe bus-to-CPU interface. This group is referred to as the Root Complex collectively (RC or Root). The RC performs on behalf of the CPU to interact with the rest of the system and is located at the "root" of the PCI inverted tree topology.

1.2 Switches and Bridges:

More units can be hooked up to a single PCIe Channel thanks to switches' fanout or aggregation capabilities. They assume the function of packet routers by establishing the routing information, for instance, an address, that a specific packet needs to follow. Bridges act as a connection point to the particulates bus, including PCI, PCI-X, and even another PCIe bus. An old PCI device can be connected to a new PCIe card and use the opposite configuration, or "backward bridge."

1.3 Native PCIe Endpoints and Legacy PCIe Endpoints:

Devices in a PCIe configuration labeled as endpoints serve as both transaction creators and completers on the bus and don't act as gateways or roads. They exclusively implement one Upstream Port and are located at the base of the tree's branches (facing toward the Root). They employ features like IO space and compatibility for IO exchanges or Locked requests that are not allowed in much more recent PCIe systems. As opposed to retrofitting existing PCI device designs with a PCIe interface, "Native PCIe Endpoints" would be PCIe electronics created from the base up. Memory-mapped devices are Native PCIe Endpoints devices.

2 Device Layers –

Because each layer has a transmit side for outbound material and a receive side for incoming traffic, the layers may be thought of as being logically divided into two sections that work separately. For hardware designers, the layered method offers certain perks since, if the logic is correctly partitioned, it may be simpler to upgrade to new versions of the standard by modifying only one layer of an original system and leaving the rest untouched. However, it's crucial to remember that the layers just specify interface duties, and styling is not needed to be segmented into sections in full compliance with the layers.

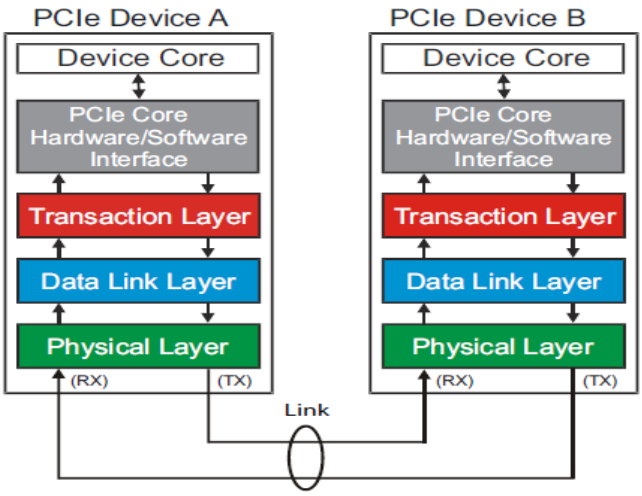

Figure – PCIe Device Layers

2.1 Device Core / Software Layer:

Such as a network interface or hard disc controller, this is the device's prime objective. The PCIe spec will not specify this as a layer, but it may be deemed one because it is located above the Transaction Layer and will either be the source or destination of all Requests. It sends requests to the Transaction Layer's submit side that contain details about the type of transaction, the address, the volume of data to be sent, and other things. Additionally, when incoming packets have been received, it serves as the location for information that has been forwarded up from the Transaction Layer.

2.2 Transaction Layer:

The Transaction Layer produces export packets in reaction to demands again from the Application Server. For non-posted transactions, it supports the split session paradigm and links an incoming Completion to an earlier sent exiting non-posted Request. The four request patterns that this layer may handle utilizing TLPs are as follows:

1. Memory
2. IO
3. Configuration
4. Messages

However, messages are a new kind for PCIe, while the initial three of them were originally supported by PCI and PCI-X. The mixture of a Request packet, which conveys an instruction to a target computer, and any Finishing packets the destination host sends back in response constitutes a Transaction.

2.3 Data Link Layer:

Between the Data Link Layers of the two neighboring devices on a Link, DLLPs are transmitted. **DLLP Assembly**- A DLLP is assembled at the transmitter's layer of the data link and swallowed by the receiver's data link layer. To monitor for mistakes at the receiver, a 16-bit CRC is introduced to the DLLP Core. The Physical Layer will receive the DLLP data. **Disassembly of DLLP**- whenever the Physical Layer will receive a DLLP. The remainder of the packet is delivered to the Data Link Layer, which analyses it for CRC mistakes before concluding what to do with it. The DLLP is not forwarded up to the Gain More insight since it is intended again for the Data Link Layer.

2.4 Physical Layer:

To make it easier for the receiver to understand the packet margins, Start and End elements are added to the TLPs and DLLPs first from Data Link Layer before being clocked into a queue in the Physical Layer. The Start and End elements are often known as "framing" characters given that they are located on both sides of a packet. In this layer, a procedure known as "byte striping" divides up any packet's bytes across each of the lanes employed by the Link. Each lane essentially functions as a separate serial connection over the Link, and at the receiver, their contents are all combined.

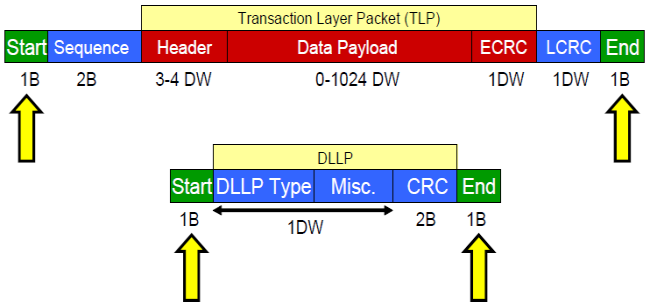

Figure – TLP and DLLP Structure at Physical Layer

III. UVM TESTBENCH ARCHITECTURE

UVM is a way of performing functional System Verilog verification that comes with a library of developed composite Verilog code. The Universal Verification Methodology is shortened storm. Accelerant developed UVM predicated on the OVM (Open Verification Methodology) Using emulation mostly, UVM is an approach for testing the performance of digital hardware. Verilog, System Verilog, VHDL, or System C are typically used to represent the hardware or environment to be evaluated at any suitable abstraction level. This could be at the gate level, register transfer level, or behavioral level. Although UVM is specifically three – a dimensional computer, it may also be utilized with additional hardware, emulation, or assertion-based verification.

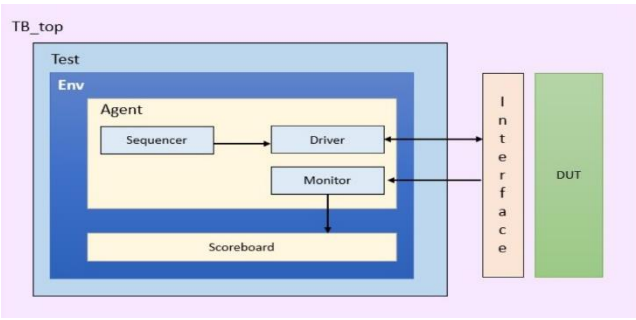

Figure – UVM Testbench.

3.1 Sequence:

The stimulus is provided by the sequence and delivered to the driver through to the sequencer. Any combination of sequences can indeed be present in an agent. UVM sequences were thought up of several data components that may be combined in diverse manners to provide engaging situations.

3.2 Sequencer:

This sequence delivers this same information to the driver via the sequencer, who then receives it. An agent may certainly also include a mixture of sequences. UVM sequences have been developed using a variety of information elements that could be coupled in various ways and should provide interesting circumstances.

3.3 Agent:

A Sequencer, Driver, and Monitor are combined everything into an object by such an operator by inserting a new and coupling the parts through TLM connections. Considering UVM focuses on installation, an agent may contain assigning new choices, including the UVM agent kind (active/passive), switches to enable capabilities like functional coverage, and numerous other factors.

3.4 Driver:

Every UVM driver is indeed an intelligent creature that understands how and where to drive information to a certain graphical interface. For instance, the UVM driver specifies how well the messages should indeed be delayed in ensure to for objective protocol in becoming valid while driving a bus mechanism like Address bus.

3.5 Monitor:

The task of translating message behavior from the graphical interface into batch-processing data items that may be transmitted to these other elements is performed by UVM monitoring.

3.6 Scoreboard:

A UVM scoreboard seems to be a testing element that has markers and confirms a design's functioning. Through the TLM Analysis port, it receives additional batch processing objects that have been recorded out of a DUT's endpoints.

IV.VIP DEVELOPMENT

Verification A which was before the collection of code used mostly for verification is called IP (VIP). It might be a collection of assumptions to check a bus interface or a component created for use in conjunction with a specific verification technique, like UVM. In addition to other components related to a specific form block, like a USB connection, this frequently includes stimulus patterns, bus concepts, and methods, a set of checkers, coverage product lines, as well as other characteristics.

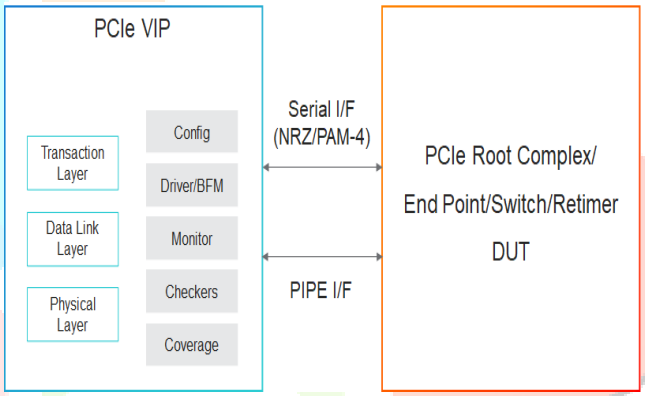

Figure – VIP for PCIe

In complement to particular PIPE as well as PIE, Verification IP (VIP) für PCI Express offers a full bus functional model (BFM) featuring billions of involves the use of computational protocol tests for all triple protocols levels (TL, DLL, PL). The VIP assists developers in reducing time toward first work, and accelerating verification completion while ensuring top when it comes. It is developed for simple integration with lead to a positive at IP, SoC, and the unit level.

V.RESULTS

That PCIe uses an Origin complicated mode to manage the data stream in the PCIe Adapter. In this instance, the root structure will serve as an Endpoints Manager VIP building. Utilizing random testing generated by the internet generating, the following findings are confirmed.

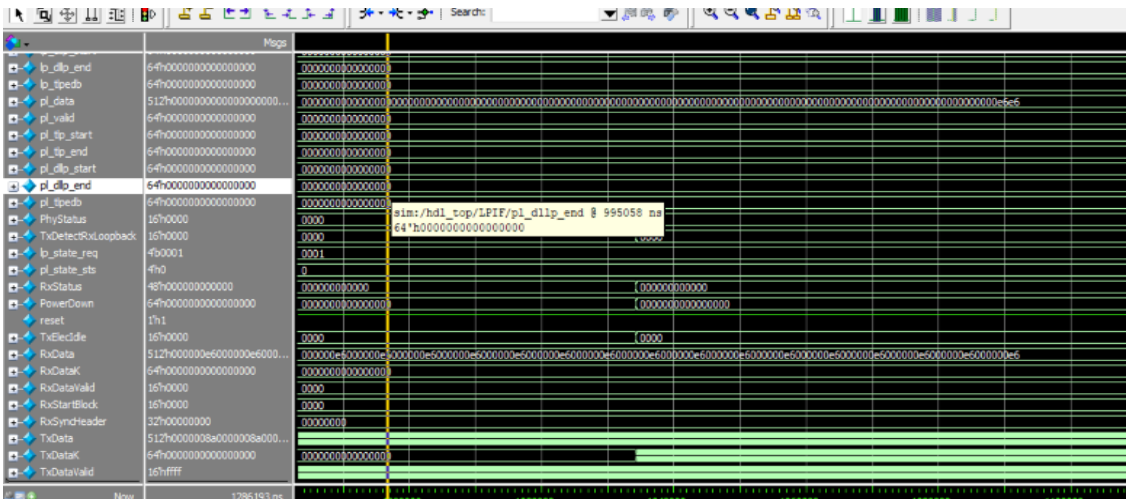

Figure – data transfer

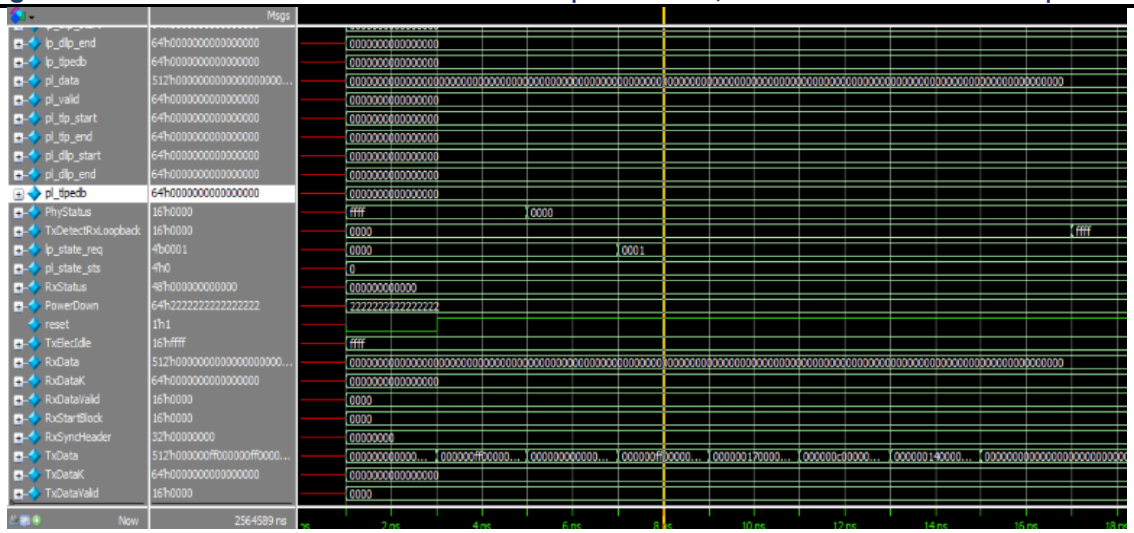

Figure – read and write transfer

VI.CONCLUSION

Employing Questasim, this creative work tested and evaluated the evolution of VIP over PCI Express in SV and UVM. The functional aspects are accomplished through PCI Express. Both speed and effectiveness of the data transport may be improved by this recommended architecture. The performance of data transfer might be improved by the suggested design. The writing and reading processes are simulated in Mentor Questa mostly using slave and maestro functionality functional testing.

VII.REFERENCES

[1]. PCI Express base Specification's revision 3.0.

[2]. Ultra-Scale+ Devices Integrated Block for PCI Express v1.3, Product Guide, Vivado Design Suite, April 29, 2021

[3]. Budruk, Ravi; Anderson, Don; Shanley, Tom (2003), Winkles, Joseph 'Joe', ed., PCI Express System Architecture, Mind share PC system architecture, Addison-Wesley, ISBN 978-0-321-15630-3 1120 pp.