International Journal of Informatics and Communication Technology (IJ-ICT)
Vol. 4, No. 1, April 2015, pp. 7~12
ISSN: 2252-8776

Simulation of PCI Express™ Transaction Layer using Hardware Description Language

V. Sudheer Raja*, Dr. M. V. Raghavendra, G. Subbarao***
* Assistant professor, Adama Science and Technology University, Adama, Ethiopia
** Associate professor, Adama Science and Technology University, Adama, Ethiopia

Article Info

Article history:

Received Nov 12, 2014
Revised Feb 20, 2015
Accepted Mar 26, 2015

Keyword:

Active HDL
IP core
PCI express
Transaction layer
VHDL

ABSTRACT

PCI Express is a high-speed serial connection that operates more like a network than a bus. PCI Express will serve as a general purpose I/O interconnects for a wide variety of future computing and communications platforms. PCI Express (PCIe) is implemented with a split-transaction protocol that provides more bandwidth and is compatible with existing operating systems. PCI Express has three discrete logical layers: the Transaction Layer, the Data Link Layer, and the Physical Layer. This paper analyze and simulates the function of Transaction layer IP core in the System Level with top-down design method, wrote the codes to implement Transaction Layer using Very high speed hardware description language (VHDL) and provided the simulation results using Active HDL Simulation tool. The simulation result shows that the designed IP core meets the required protocol specifications for the proper functioning of PCI Express Transaction layer.

Corresponding Author:

V. Sudheer Raja,
Assistant professor,
Adama Science and Technology University,
Adama, Ethiopia.
Email: sudheerrajav@yahoo.com

1. INTRODUCTION

PCI Express is a high-speed serial connection that operates more like a network than a bus. PCI Express will serve as a general purpose I/O interconnects for a wide variety of future computing and communications platforms. Instead of one bus that handles data from multiple sources, PCIe has a switch that controls several point-to-point serial connections. Peripheral Component Interconnect (PCI) slots are such an integral part of a computer's architecture that most people take them for granted. For years, PCI has been a versatile, functional way to connect sound, video and network cards to a motherboard. Key PCI attributes, such as its usage model and software interfaces are maintained whereas its bandwidth-limiting, parallel bus implementation is replaced by a long-life, fully-serial interface. A split-transaction protocol is implemented with attributed packets that are prioritized and optimally delivered to their target.

The PCI Express Architecture is specified in three logical layers as shown in Figure 1. The PCI model with load-store architecture with a flat address space is maintained to provide compatibility to all existing applications and drivers. Transaction layer, Data link layer and physical layer form the basic architecture. Each of these layers proces the information or data being transmitted and received and are subdivided into two sections accordingly. PCI Express components communicate with each other through packets which carries the information. These Packets are generated in the Transaction and Data Link Layers. The physical layer transports packets between the link layers of two PCI Express agents.

The primary role of a link layer is to ensure reliable delivery of the packet across the PCI Express link. The transaction layer receives read and write requests from the software layer and creates request packets for transmission to the link layer. As the transmitted packets flow through the other layers, they are extended with additional information necessary to handle packets at those layers.

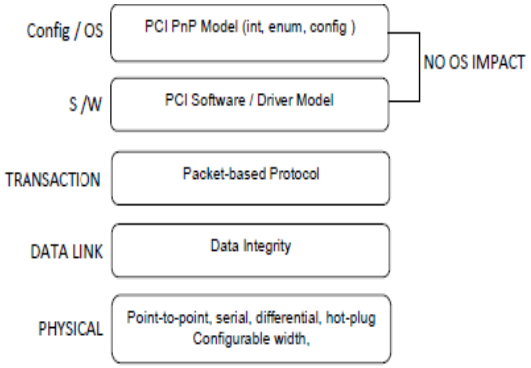

Figure 1. PCI Express™ Architecture showing logical layers

2. TRANSACTION LAYER

The transaction layer is the top Layer of PCI Express architecture. It is responsible for the assembly and disassembly of data Packets used for communication. The transaction layer receives read and write requests from the software layer and creates request packets for transmission to the link layer. These Data packets are called TLP's (Transaction layer packets). The Transaction Layer is also responsible for managing credit-based flow control for TLPs. All requests are implemented as split transactions and some of the request packets will need a response packet. Each packet has a unique identifier that enables response packets to be directed to the correct originator. TLP's supports either 32bit memory addressing or extended 64bit memory addressing.

The transaction layer supports four address spaces: it includes the three PCI address spaces memory, I/O, configuration and adds a Message Space to support all prior side-band signals, such as interrupts, power-management requests, resets, and so on, as in-band Messages. The basic use of each address space is shown in Table 1.

Table 1. PCI Express Address Space and Transaction Types

Address Space	Transaction Types	Purpose
Memory	Read, Write	Transfer data to or from a location in the system memorymap
IO	Read, Write	Transfer data to or from a location in the system IO map
Configuration	Read, Write	Transfer data to or from a location in the configuration space of a PCI-compatible device.
Message	Baseline, Vendor-specific	General in-band messaging and event reporting (without consuming memory or IO address resources)

2.1. Transaction Layer Packet Format

The Transaction Layer is responsible for managing credit-based flow control for TLPs. All requests are implemented as split transactions. Accesses to the four address spaces in PCI Express are accomplished using split-transaction requests and completions.

Three kinds of packets Posted, Non-Posted and Completion (Cpl) are responsible for transactions handled by the transaction layer. Each Transaction Layer Packet contains a three or four double word (12 or 16 byte) header. Figure 2 shows the complete view of a TLP Four double word format.

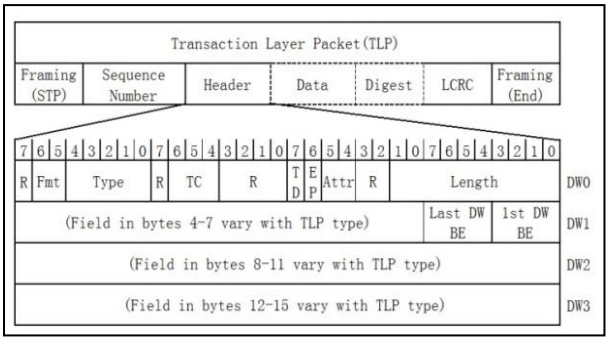

Figure 2. Generic 4DW Header TLP Format

Included in the 3DW or 4DW header are two fields, Type and Format (Fmt), which define the format of the remainder of the header and the routing method to be used on the entire TLP as it moves between devices in the PCI Express topology.

2.2. Overview of TLP Header Information

As TLPs arrive at an ingress port, they are first checked for errors at both the physical and data link layers of the receiver. Assuming there are no errors, TLP routing is performed; basic steps include:

a. The TLP header Type and Format fields in the first DWord are examined to determine the size and format of the remainder of the packet
b. Depending on the routing method associated with the packet, the device will determine if it is the intended recipient; if so, it will accept (consume) the TLP. If it is not the recipient, and it is a multi-port device, it will forward the TLP to the appropriate egress port--subject to the rules for ordering and flow control for that egress port
c. If it is neither the intended recipient nor a device in the path to it, it will generally reject the packet as an Unsupported Request (UR).

Table 2 below summarizes the encodings used in TLP header Type and Format fields. These two fields, used together, indicate TLP format and routing to the receiver.

Table 2. TLP Header Type and Format Field encoding

TLP	FMT[1:0]	TYPE [4:0]
Memory Read Request (MRd)	00 = 3DW, no data 01 = 4DW, no data	0 0000
Memory Read Lock Request (MRdLk)	00 = 3DW, no data 01 = 4DW, no data	0 0001
Memory Write Request (MWr)	10 = 3DW, w/ data 11 = 4DW, w/ data	0 0000
IO Read Request (IORd)	00 = 3DW, no data	00010
IO Write Request (IOWr)	10 = 3DW, w/ data	0 0010
Config Type 0 Read Request (CfgRd0)	00 = 3DW, no data	0 0100
Config Type 0 Write Request (CfgWr0)	10 = 3DW, w/ data	0 0100
Config Type 1 Read Request (CfgRd1)	00 = 3DW, no data	0 0101
Config Type 1 Write Request (CfgWr1)	10 = 3DW, w/ data	0 0101
Message Request (Msg)	01 = 4DW, no data	1 0 RRR* (for RRR, see routing subfield)
Message Request W/Data (MsgD)	11 = 4DW, w/ data	1 0 RRR* (for RRR, see routing subfield)
Completion (Cpl)	00 = 3DW, no data	0 1010
Completion W/Data (CplD)	10 = 3DW, w/ data	0 1010
Completion-Locked (CplLk)	00 = 3DW, no data	0 1011
Completion W/Data (CplDLk)	10 = 3DW, w/ data	0 1011

2.3. Split Transaction Protocol

Accesses to the four address spaces in PCI Express are accomplished using split-transaction requests and completions. The split transaction protocol is an improvement over earlier bus protocols (e.g. PCI) which made extensive use of bus wait-states or delayed transactions (retries) to deal with latencies in accessing targets.

In PCI Express, the completion following a request is initiated by the completer only when it has data and/or status ready for delivery. The fact that the completion is separated in time from the request which caused it also means that two separate TLPs are generated, with independent routing for the request TLP and the Completion TLP. Note that while a link is free for other activity in the time between a request and its

subsequent completion, a split-transaction protocol involves some additional overhead as two complete TLPs must be generated to carry out a single transaction.

2.4. Virtual Channel (VC) and Transaction ordering

The VC mechanism provides support for carrying, throughout the fabric, traffic that is differentiated using TC labels. The foundation of VCs is independent fabric resources (queues/buffers and associated control logic). These resources are used to move information across Links with fully independent flow-control between different VCs. Traffic is associated with VCs by mapping packets with particular TC labels to their corresponding VCs. Table 1 defines the ordering requirements for PCI Express Transactions. The rules defined in this table apply uniformly to all types of Transactions on PCI Express including Memory, I/O, Configuration, and Messages. The ordering rules defined in this table apply within a single Traffic Class (TC). There is no ordering requirement among transactions with different TC labels. Note that this also implies that there is no ordering required between traffic that flows through different Virtual Channels since transactions with the same TC label are not allowed to be mapped to multiple VCs on any PCI Express Link. For Table 3, the columns represent a first issued transaction and the rows represent a subsequently issued transaction. The table entry indicates the ordering relationship between the two transactions.

In order to obtain higher efficiency, Flow Control (FC) is used to prevent overflow of Receiver buffers and to enable Compliance with the ordering rules, there are six types of information tracked by Flow Control for each Virtual Channel: Posted Header (PH), Posted Data (PD), Non-Posted Header (NPH), Non-Posted Data(NPD), Completion Header (CplH) , Completion Data (CplD), Each Virtual Channel maintains an independent Flow Control credit pool. The unit of Flow Control credit is 4 DW for data.

Table 3. Ordering rules summary

Row Pass Column?		Posted Request	Non-Posted Request		Completion	
		Memory Write or Message Request (Col 2)	Read Request (Col 3)	I/O or Configuration Write Request (Col 4)	Read Completion (Col 5)	I/O or Configuration Write Completion (Col 6)
Posted Request	Memory Write or Message Request (Row A)	a) No b) Y/N	Yes	Yes	a) Y/N b) Yes	a) Y/N b) Yes
Non-Posted Request	Read Request (Row B)	No	Y/N	Y/N	Y/N	Y/N
	I/O or Configuration Write Request (Row C)	No	Y/N	Y/N	Y/N	Y/N
Completion	Read Completion (Row D)	a) No b)Y/N	Yes	Yes	a) Y/N b) No	Y/N
	I/O or Configuration Write Completion (Row E)	Y/N	Yes	Yes	Y/N	Y/N

3. SYSTEM ARCHITECTURE AND DESIGN

Receive layer, Configuration space and Transmit layer are the three modules that form the basic architecture of PCI express. Every receive layer consists of eight VC's VC_0- VC_7 among them VC_0 is necessary while others are optional. The basic architecture of PCI express is shown in Figure 3.

Each VC is sub divided into two modules: Receive buffer and Controller. Each receive buffer is subdivided into six parts in order to store PH, PD, NPH, NPD, CplH, and CplD respectively.

Header stores no more than 128 credits data stored no more than 2047 credits. The size of credits is indicated by FC credits. Controller implement the function of FIFO, FC and Sequencing (ordering).There must be sequence strictly to the packet of Posed and Cpl ,according to the rule of transmit ordering.

Transmit Layer added VC Arbitration and Digest module to compare with Receive layer. Transmit Layer is not only ordering for Posed and Cpl packet, but also detect Flow Control.

a. *Interface Signal to Software Layer and handshake Sequence rules:* desc_n[127:0] indicate the VCn transmit header and data; data_n[127:0] indicate the VCn Receive header and data. Rules of Handshake in Header File transmit and receive: the sender set req_n to indicate RTS (Request To Send), the Receiver receives the request then set ack_n; Rules of Handshake in Data file transmit and receive: the sender set dfr_n to indicate the beginning of data transmit, and set ws_n to inform Receiver wait when Receiver is busy in the data transmit process, next data will be send until ws_n clear, it is time to clear dfr_n which inform receiver data send finished when the last 1DW data was sent.
b. *Interface Signal to Data Link Layer and handshake Sequence rules:* desc_data [31:0] are TLP signal which come from Data Link Layer, The corresponding part of Frame represent frame header. BlockT

indicate Replay state of Data Link Layer, and also indicate to pause the transmit TLP. FC credits n indicate the FC credits value which includes cred_alloc_p, cred_alloc_np, cred_alloc_cpl, cl_p, cl_np, cl_cpl that needed transmit or refresh. vc_reqn is set when VCn needed transmit data to DLL, vc_ack_n is set when Arbitrator permit this request, when VC received vc_ack_n , then set vc_get_n . When last data was sent vc_get_n will be cleared at the same time, and then inform Arbitrator to end the VC permission. Its configure register include conf_addr, conf_data_i, conf_data_o, conf_wr, conf_rd are configure address, data in , data out, read and write signal respectively,all of them are used to configure Arbitrator configure register, such as to configure VC arbitrator mode.

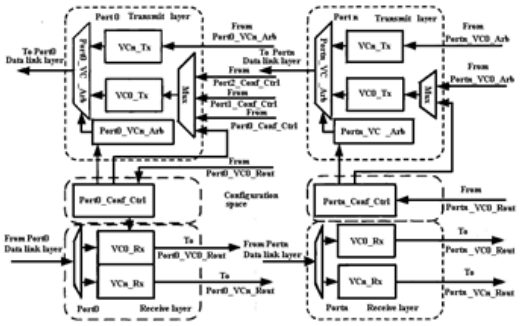

Figure 3. Transaction layer architecture

4. TRANSMITTER AND RECEIVER FLOW CHART

The transmitter flowchart is shown in Figure 4. The only different between transmitter flowchart and receiver flowchart is the former needed to add in flow control, so pnpc_rd also depends on the flow control. if c_ph[7] is 1shows the other side is already full.cl_p stands for CredAlloc of P; and cc_p stands for Credits Consumed of P.Flowchart (d) is virtual channel flow. In the initial state, if the data link layer is not block and vc_req is valid, now_vc value will update to next_vc value, and choose next channel.

The selection of channel is decision by arbitrate mode. Unnecessary vc may be chosen owing to the arbitrator support look-up table, so we must to judge whether the vc_req_n of this channel is 1 or not at this moment vc_get_n will clear when the virtual channel data transmission the last 1DW, and let vc_arb return to initial state. Before the arbitration we should configure VC registers in order to choose the mode of arbitration.

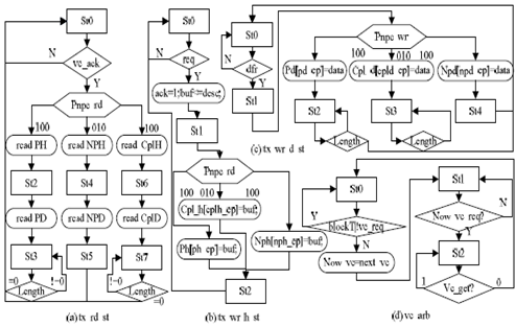

Figure 4. Transmitter Flow Chart

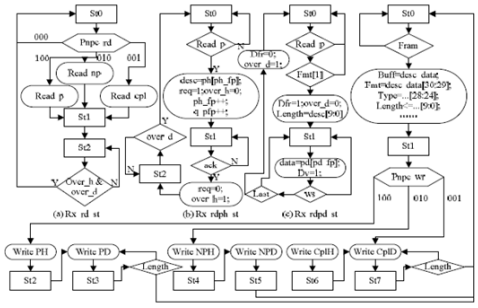

Figure 5. Receiver Flow Chart

Figure 5 shows Receiver flowchart, there are four sub-flowcharts: (a) is read flowchart, in this flowchart it is first to judge pnpc_rd is read_P, read_NP or read_Cpl TLPs at initial state, if pnpc_rd is 3'b100 then set read_p to start flowchart (b) and (c) which transmit PH and PD.Length register will record the remainder data in order to judge whether the data completely transmit. When flow chart (b) and (c) are finished, Rx_rd_st will return to initial state. When Data Link Layer transmit packets, Frame Synchronous signals will send, flowchart (d) start to receive data at the same time, St1 will judge next state is P, NP or Cpl

TLP by judge pnpc_wr. Then the header and data shall be written. Length Register is to judge whether data are completely transmission. Write NP, Cpl TLPs and P TLPs roughly as the same.

5. SIMULATION RESULTS

Transmitter and Receiver flowcharts are simulated for their functional verification using Active HDL Simulation tool.The figure 6 and figure 7 shows the simulation results for receiver and transmitter virtual channels respectively.

In figure 6 the different transactions made by the receiver virtual channel is divided by a vertical line at every 50ns of time,for example 0 to 50ns line indicates the initialization of various buffers and the line from 350ns to 400ns indicates the transaction of receiving data from data link layer.

The Figure 7 shows various transactions made by the transmitter virtual channel. For convenience every transaction is separated with a vertical line at every multiple of 50ns.Transactions include header and data read for posted, Non-posted and completion transactions.

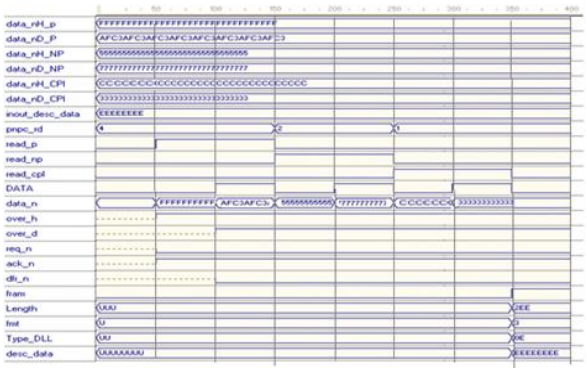

Figure 6. Simulation diagram of Receiver Virtual channel

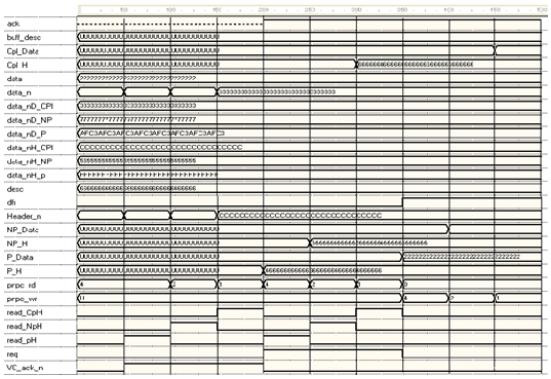

Figure 7. Simulation diagram of Transmitter Virtual channel

6. CONCLUSION

In this paper we presented the method of implementing the widely adopted higher performance PCI Express interconnect transaction layer with a top down design method, and wrote the VHDL codes to implement it. The simulation results show that the design can achieve the basic function of PCI Express Transaction Layer that meet the protocol of PCI Express™ Base Specification Revision 2.0. The same basic design methods can be used to improve the equipment transmission bandwidth by further implementing the PCI Express IP Core.

REFERENCES

[1] The Intel Developer Web site http://www.intel.com/technology/3gio
[2] The PCI-SIG Web site http://www.pcisig.com
[3] PCI Express system architecture by Ravi Budruck, Don Anderson, Tom shanley Mind share international publications
[4] PCI SIG. PCI-Express™ base specification Rev 2.0. PCI SIG. 2006.12.20
[5] PLD Applications, PCI Express Expert Core Reference Manual, 2006-02
[6] V. Bhaskar, VHDL primer, Prentice Hall/Pearson, 2005.01