Configurable Design of PCI Express Data Link Layer Receiver

Bhavesh S. Motiwala
M.E. in VLSI & Embedded System Design
GTU PG School, Gujarat Technological University, Ahmedabad, Gujarat, India

Abstract—PCI Express is a modern, high-performance communication protocol implementing sophisticated features to meet today's performance demands. This paper looks at the success of the widely adopted PCI Express protocol and describes a higher performance. This paper presents the proposal of the implementation of the Data Link Layer Receiver of PCI-Express, in which the architecture contains the Receiver module which ensures the reliability conveying of the Transaction Layer Packets (TLPs) between two components using the PCI-Express protocol. The Receiver module will take part into the communication via receiving data from the Physical Layer and supplying it to the Transaction Layer with several communication parameters. Here High Level as well as Detailed Design for the Receiver section with the sub modules are presented. The attributes presented here are amongst positive/negative acknowledgements, flow control, storage and retry buffers (for failed packets), power management for successful communication.

Keywords: PCIe, DLL, DLLP, TLP, LCRC

I. INTRODUCTION

The PCI bus has been widely used for the last 10 years and it will be used for the next few years. However, todays and tomorrow's processors and I/O devices are demanding much higher I/O bandwidth than PCI can deliver [1]. PCI Express is a high performance, general purpose I/O interconnects defined for a wide variety of future computing and communication platforms. Key PCI attributes, such as its usage model, load-store architecture, and software interfaces, are maintained, whereas its parallel bus implementation is 5 replaced by a highly scalable, fully serial interface [1]. PCI Express takes advantage of recent advances in point-to-point interconnects, Switch-based technology, and packetized protocol to deliver new levels of performance and features. Power Management, Quality of Service (QoS), Hot- Plug/Hot-Swap support, Data Integrity, and Error Handling are among some of the advanced features supported by PCI Express. The PCI Express architects have carried forward the most beneficial features from previous generation bus architectures and have also taken advantages of new developments in computer architecture [1]. The predecessor Bus PCI has some limitations and to overcome those limitations PCI Express has been introduced, which will be analyzed here.

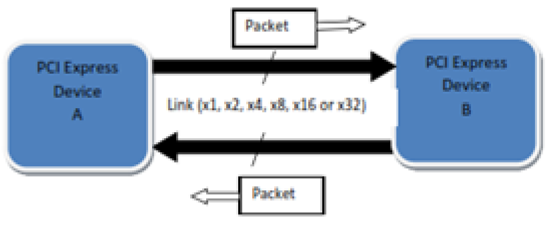


Fig. 1: PCI Express Link

As shown in Figure 1, a PCI Express interconnect consists of either a x1, x2, x4, x8,x12, x16 or x32 point-to-point Link. A PCI Express Link is the physical connection between two devices. A Lane consists of signal pairs in each direction. A x1 Link consists of 1 Lane or 1 differential signal pair in each direction for a total of 4 signals. A x32 Link consists of 32 Lanes or 32 signal pairs for each direction for a total of 128 signals. The Link supports a symmetric number of Lanes in each direction. During hardware initialization, the Link is initialized for Link width and frequency of operation automatically by the devices on opposite ends of the Link. No OS or firmware is involved during Link level initialization [6].

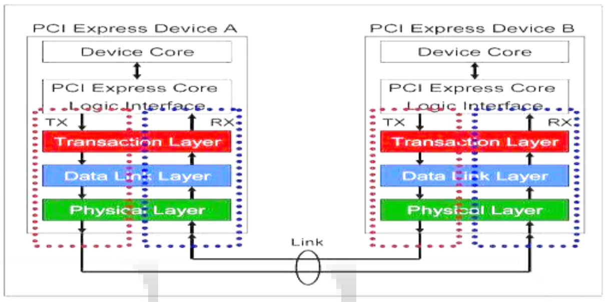


Fig. 2: PCI Express Device Layers

The PCI Express specification defines a layered architecture for device design as shown in Figure 2. The layers consist of a Transaction Layer, a Data Link Layer and a Physical layer. As shown in the figure, the packet communication takes place in all the layer. Transmitter side, data is inserted from the device core and further processed in the packet form. Other informations such packet number, LCRC, ECRC are appended by each layer separately for robust communication and at the receiver side, the reverse process takes place and at the end, the raw data is captured by the device end at the receiver by removing all indicating header and footers.

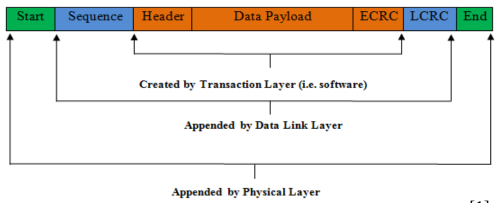
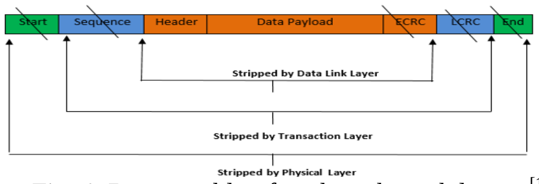

Start
Sequence
Header
Data Payload
ECRC
LCRC
End
Created by Transaction Layer (i.e. software)
Appended by Data Link Layer
Appended by Physical Layer

Fig. 3: Packet flow through the layers [1]

As discussed above, packet transmission is manipulated in all the layers as shown in the Figure 3. All we need to deal is to transmit the Data Payload from Transmitter to Receiver. As shown in the above figure, the assembly operation of the packet inside the PCI Express generation at different layers. The software layer/device core sends to the Transaction Layer the information required to assemble the core section of the TLP which is the header and data portion of the packet. Some TLPs do not contain a data section. An optional End-to-End CRC (ECRC) field is calculated and appended to the packet. The ECRC field is used by the ultimate targeted device of this packet to check for CRC errors in the header and data portion of the TLP [6].

A neighbouring receiver device receives the incoming TLP bit stream. As shown in Figure 4, the received TLP is decoded by the Physical Layer and the Start and End frame fields are stripped [6].


Start
Sequence
Header
Data Payload
ECRC
LCRC
End
Stripped by Data Link Layer
Stripped by Transaction Layer
Stripped by Physical Layer
Fig. 4: De-assembly of packets through layers [1]

So here the De-assembly mechanism has been presented which will be done at Receiver side.

II. Proposed Architecture Of Dll

The receiving Data Link Layer is responsible for checking the integrity of received TLPs and for submitting them to the Transaction Layer for further processing. Data Link Layer is responsible for reliably exchanging information with its counterpart on the opposite side of the Link. Initialization and power management services [7].

It is also responsible to accept power state Requests from the Transaction Layer and convey to the Physical Layer. It also conveys active/reset/disconnected/power managed state to the Transaction Layer i.e. Data protection, error checking, and retry services, CRC generation, Transmitted TLP storage for Data Link levels retry, Error checking, TLP acknowledgment and retry messages, Error indication for error reporting and logging [7]. All these attributes have been covered and designed here and shown in this paper with the architectural view.

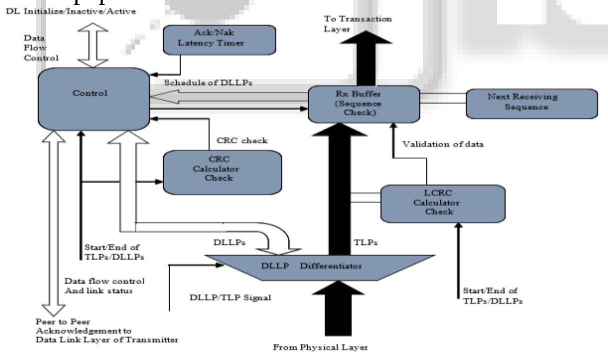


Fig. 5: Proposed architecture of PCIe Receiver

As shown in the above Figure 5, an architecture with all the sub modules included in the design at the receiver side which has been implemented. All the modules have been described as below with their functionalities.

A. DLLP Differentiator

This module differentiates between TLPs and DLLPs through the indication of the Physical Layer about the kind of data that is passing. This block receives 8 bit Data and has two outgoing paths, one to send the DLLPs to the CRC Calculator Check and another one to send TLPs to LCRC Calculator Check, where the validation of data will be done. Let's we consider that the data that comes from the Physical Layer is a TLP.

B. LCRC Calculator Check

This block performs additionally to the calculation of the LCRC (32 bits). 32 bits LCRC is generated from the payload data bits and other header information received from the Transmitter. Last 32 bits received (LCRC bits generated from Transmitter) are not taken into account. The LCRC Calculator Check compares the LCRC value of the TLP with the LCRC value calculated. After this comparison, the LCRC Calculator check sends a signal to the R_x Buffer that the LCRC verification is correct or fails. For LCRC module, the divisor used is 04C11D57.

C. Receive Buffer

It stores temporally the TLPs that are incoming from the Physical Layer. When the TLPs arrive from Physical Layer, the LCRC Calculator Check verifies that the TLP is not corrupt. At the same time the TLP is been stored in the Rx Buffer. The buffer sixe has been kept of 2048 bytes which is same as it is kept in T_x buffer. It stores the TLP (i.e. TLPs with the sequence number and LCRC appended by the Data Link Layer Transmitter) packets on the temporary basis. Sequence number (12 bits) and LCRC (32 bits) are discarded after their respectively. Only TLPs with payload Transaction Layer header are sent to the Transaction Layer. After the Physical Layer asking for the Transaction request, on the basis of that Receive Buffer responds with Buffer Credits available telling the Physical Layer about the number of vacant positions in Receive buffer. For every Negative Acknowledgement, the address pointer gets updated to the sequence number with the corrupt packet.

D. CRC Calculator Check

This block compares the CRC value (16 bits) of the DLLP with the calculated CRC value in the CRC Calculator Check. If the DLLP is correct, then the CRC Calculator check indicates to the Control module that the DLLP was received without error, and the DLLP is processed. If it is not the case, the DLLP is ignored by the Data Link Layer without a further action.

E. Next Receiving Sequence

Next Receive Sequence (nrs) shows the next expected sequence of the TLP to be received from the Transmitter side. For every LCRC pass or Fail, this module will update its next sequence value, which it is expecting from Transmitter to be received.

– If **tlp=nrs**, it would get incremented.
– If **tlp<nrs or tlp>nrs**; it would not get incremented and keep its value as it was.

It is a 12 bit counter counting from 0-4095 and rolls over after reaching to maximum value. It sends Negative Acknowledgement when either LCRC or Sequence check got failed and sends Positive Acknowledgement when LCRC, Sequence number check are approved successfully.

F. Sequence Number Check

After successful check of LCRC error done, sequence number check is to be done afterwards to validate the sequence number, which is expected to be arrived. The comparison is performed between the Next Receive Sequence and TLP Sequence number.

– If TLP Sequence number = Next Receive Sequence Number; TLPs are not said to be corrupted and Acknowledgement is sent for this sequence number packet to the Transmitter.
– If TLP Sequence number < Next Receive Sequence Number; TLPs are duplicated i.e. we have already sent the acknowledgement for this particular packet. Hence

Acknowledgement is sent for this sequence number packet to the Transmitter.

– If TLP Sequence number > Next Receive Sequence Number; then the sequence number we expected has been skipped and TLPs is said to be corrupted and hence we need to send the Negative Acknowledgement to the Transmitter.

G. *Ack Nak Latency Timer*

This timer has a value, which is approximately 1/3rd of Transmitter Replay timer. When the timer expires, the receiver schedules an Ack DLLP with the sequence number of a last good unacknowledged TLP. It is used to send Ack/Nak DLLP for the received TLP before the Transmitter Replay timer expires. It resets when either Ack/Nak is scheduled or Data Link Layer is in Inactive State.

$$\mathbf{LatencytimerValue} = \left[\left(\frac{\mathbf{MaxpayloadSize + TLPOverhead}}{\mathbf{LinkWidth}}\right) * \mathbf{AckFactor}\right] + \mathbf{InternalDelay}$$

		Link Operating Width						
		x1	x2	x4	x8	x12	x16	x32
Max_Payload_Size (bytes)	128	333 AF = 1.4	224 AF = 1.4	169 AF = 1.4	163 AF = 2.5	154 AF = 3.0	144 AF = 3.0	129 AF = 3.0
	256	512 AF = 1.4	313 AF = 1.4	214 AF = 1.4	203 AF = 2.5	186 AF = 3.0	168 AF = 3.0	141 AF = 3.0
	512	655 AF = 1.0	385 AF = 1.0	250 AF = 1.0	182 AF = 1.0	205 AF = 2.0	182 AF = 2.0	148 AF = 2.0
	1024	1167 AF = 1.0	641 AF = 1.0	378 AF = 1.0	246 AF = 1.0	290 AF = 2.0	246 AF = 2.0	180 AF = 2.0
	2048	2191 AF = 1.0	1153 AF = 1.0	634 AF = 1.0	374 AF = 1.0	461 AF = 2.0	374 AF = 2.0	244 AF = 2.0
	4096	4239 AF = 1.0	2177 AF = 1.0	1146 AF = 1.0	630 AF = 1.0	802 AF = 2.0	630 AF = 2.0	372 AF = 2.0

Table 1 Latency timer limit according to Payload size and Link width [1]

H. *FSM Design DLL Controller*

The Finite State Machine designs have been done for LCRC check and Receive buffer which clears the big picture of the logic being implemented inside the Data Link Layer.

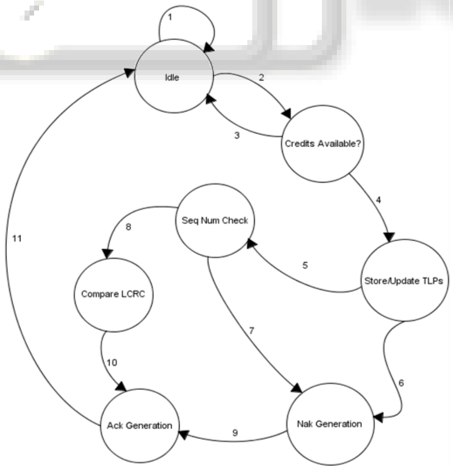


Fig. 6: State Transmissions for PCIe Receiver

First of all, TLP and DLLP packets are received via DLLP differentiator. The state is in Idle mode that time, It will then check for the credit availability and then update the TLPs in the receive buffer. Sequence Number check and LCRC comparison takes place then. At the end Positive Acknowledgement (Ack) or negative Acknowledgement (Nak) is generated and sent to Transmitter.

Trans. No.	Present State	Next State	Input signal causing this Transition
1	Idle	Idle	phy_req=1
2	Idle	Credits Available	phy_req=0
3	Credits Available	Idle	buff_full=1
4	Credits Available	Store/Update TLPs	buff_full=0
5	Store/Update TLPs	Seq Num Check	rx_lcrc_pass=1
6	Store/Update TLPs	Nak Generation	lcrc_fail=1
7	Seq Num Check	Nak Generation	tlp>nrs
8	Seq Num Check	Ack Generation	tlp=nrs tlp<nrs
9	Nak Genartion	Send DLLP with seq	nak_flag=1
10	Ack Generation	Send DLLP with seq	ack_flag=1
11	Send DLLP with seq	Idle	phy_req=0

Table 2 State Transitions for Receive Buffer Check

III. Simulation Results

All the modules design has been coded in Verilog language and for results, the simulation has been done in *Modelsim PE Student edition* simulator. After designing the PCI-Express Data Link Layer Receiver, Linting has been done over it using *HDL companion* tool to check the modularity of the design code. After Linting, Synthesis has been completed using *Xilinx Vivado* software and verified timing, power and utility report.

Simulation results of some of the important modules are given below.

Fig. 7: Result for Parallel LCRC generation

Figure 7 shows the generation of the LCRC which consist of 32 bits for every received packet from Transmitter. IT calculates LCRC for whole the packet except the last packet which contains the generated LCRC as it has to compare the Generated LCRC at the receiver side with received LCRC from Transmitter side to make the further transaction happen.

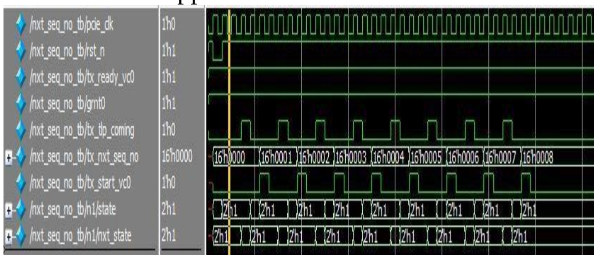

/nxt_seq_no_tb/pcie_clk
1'h0
/nxt_seq_no_tb/rst_n
1'h1
/nxt_seq_no_tb/tx_ready_vc0
1'h1
/nxt_seq_no_tb/grnt0
1'h1
/nxt_seq_no_tb/tx_tlp_coming
1'h0
/nxt_seq_no_tb/tx_nxt_seq_no
16'h0000
16'h0000 16'h0001 16'h0002 16'h0003 16'h0004 16'h0005 16'h0006 16'h0007 16'h0008
/nxt_seq_no_tb/tx_start_vc0
1'h0
/nxt_seq_no_tb/n1/state
2'h1
/nxt_seq_no_tb/n1/nxt_state
2'h1

Fig. 8: Result for Next Receive Sequence generation

Figure 8 shows the expected receiving packet which is of 32 bits and expected to be received from Transmitter. As shown in figure, it gets incremented for the successful reception of TLP packets.

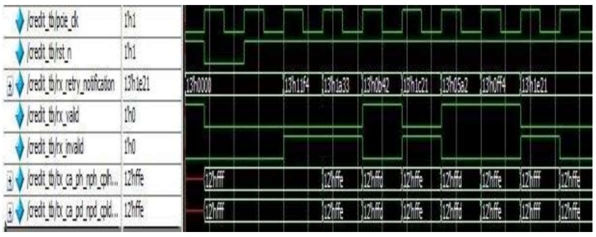

credit_tb/pcie_clk
credit_tb/rst_n
credit_tb/rx_retry_notification
credit_tb/rx_valid
credit_tb/rx_invalid
credit_tb/tx_ca_ph_nph_cph...
credit_tb/tx_ca_pd_npd_cpd...

Fig. 9: Results for Credit Count for Receive buffer

It shows the available credits in the receive buffer, so that other layers can start communicating after getting acknowledgement of this. If the buffer doesn't have enough credits, communication cannot be started.

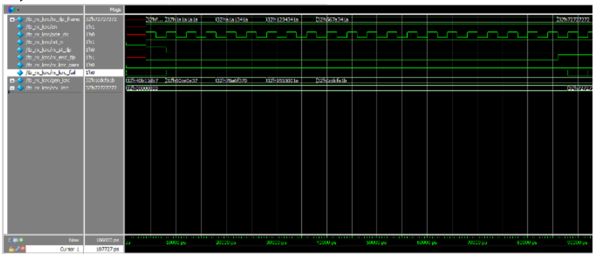


Fig. 10 Results for FSM Generation

Figure 10 shows the FSM result for LCRC calculation and comparison. We can see that last packet which has been received from Transmitter which is generated LCRC is not taken into account while calculating it in Receiver side. Here after receiving whole the packet with LCRC generated from transmitter side, at receiver LCRC can be compared so that any corruption in data can be fetched over here indicating LCRC pass or fail.

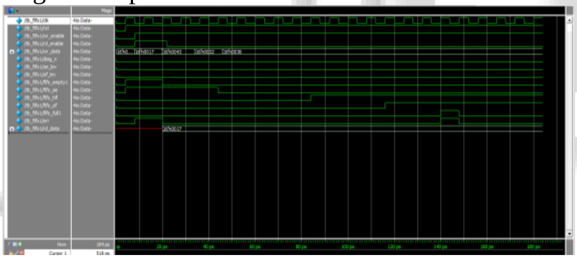


Fig. 11: Results for R_x Buffer behavior with error

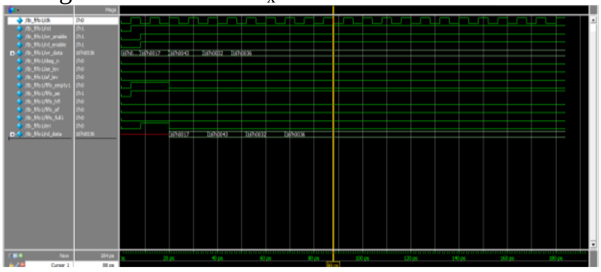


Fig. 12: Results for R_x Buffer behavior with no error

As we can see from above Figure 11 & 12, we have the data stored in the Receive buffer after Ack or Nak. In figure, the Error Indication signal such as Buffer Full, Buffer Empty etc. also used to deal with the communication with the upper and lower layers. First figure shows the behavior with error indication signals and the second one indicates the behavior without any error.

IV. Scope And Conclusion

Hence from all the above review of papers and published specifications, it can be said that PCI Express is the next generation communication protocol and it can be laid to mobile, network and other endpoints too. It has definite advantages as it has some advanced attributes like point to point protocol, enhanced error control, enhanced flow control, faster access times, data clock embed in the serial bits itself. Designing a Data Link Layer with these attributes for Transmitter and Receiver will result into customized IP core for Data Link Layer. We are able to add and enhance the features for flow and error control too as per the application specific requirement. We can overcome the drawbacks of fixed buffer size at the receiver side with the replay timer configuration to make the efficient communication with the further layers of PCI Express.

I have completed the project specifications, High level Design and Detailed Design of the project. Coding and Synthesis part is done too with the above specifications. An efficient design have been made which is highly customizable and can be linked to other IPs. The design is made on modular bases, so future extension can be done very easily with higher standards of the protocol. It has forward as well as backward compatibility too.

Acknowledgement

The analysis and results in this paper are from the work which was conducted as a Master Thesis by me in the field of VLSI & Embedded System Design, in co-operation with Centre for Development Advanced Computing (C-DAC), Pune. I take the opportunity to heartily thank our project guide respected **Mr. Santosh S. Jagtap** for his valuable guidance and touch of inspiration and motivation throughout the project. I am very grateful for those who supported and helped me during conducting this work.

References

[1] "PCI Express Base Specification Revision 3.0", PCI-SIG, Retrieved 10th November, 2010
[2] Eugin Hyun and Kwang-Su Seong, "The design of PCI Express for future communication platform", International Conference on Electrical and Electronics Engineering, Mexico, Year: 2009
[3] Hu Li, Yuan'an Liu, Dongming Yuan and Hefei Hu, "A Wrapper of PCI Express with FIFO Interfaces based on FPGA", IEEE Symposium on High Performance Interconnects, Korea, Year: 2012
[4] Li Jun and Wang Wei, "PCI Express Interface Design and Verification", International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE), China, Year: 2013
[5] Eugin Hyun and Kwang-Su Seong, "Design and verification for PCI Express controller", International Conference on Electrical and Electronics Engineering, Mexico, Year: 2012
[6] M. Dineshkumar, M.Tech. Thesis, "Implementation of PCS of Physical Layer for PCI Express", National Institute of Technology, Rourkela, 2009
[7] Ravi Budrak / Tom Shanley / Don Anderson, "PCI system Architecture", 4th edition, MINDSHARE, INC. pp 55-101
[8] "The PCI Express Architecture and Advanced Switching", Intel white paper, Year: 2003