An efficient approach to evaluate PCIe DMA design and DMA performance for Common Readout Unit(CRU)

S. Mukherjee[1],[*] F. Costa[2], R. Paul[3], A. Chakrabarti.[3], S.A. Khan[4], J. Mitra[4], and T. Nayak[4]

[1] *Centre for Astroparticle Physics and Space Science, Bose Institute, Kolkata -700091,India*
[2] *Organisation Europeenne Pour la Recherche Nucleaire,CERN,Switzerland*
[3] *University of Calcutta, Kolkata-700091,India and*
[4] *Variable Energy Cyclotron Centre, Kolkata-700091,India*

1.Introduction

A Large Ion Collider Experiment (ALICE) is the detector system at CERN LHC dedicated particularly to study the properties of QGP (Quark Gluon Plasma). The detector system of ALICE will undergo a major upgrade during the upcoming Long Shutdown 2 (LS2) which is forseen to start in 2018. Most of detector sub-systems will upgrade their detector hardware and associated electronics to support higher collision rate and luminosity. A common read out unit (CRU) is proposed which will be used by all the upgraded detector systems to read out detector data at higher rate. PCIe-DMA (Peripheral Component Interconnect Express-Direct Memory Access) is one of the modules of CRU. It is the interface between Online-Offline (O^2) computing system and CRU. In this paper we introduce the motivation behind the evaluation of PCIe-DMA and then brief about its design evaluation process and the related results.

2.Motivation behind PCIe-DMA evaluation

The functionality of PCIe-DMA in CRU can be grouped into three roles, as explained below:

The large amount of data coming from detector side as well as monitoring points to the CRU are to be forwarded to O^2 after processing inside CRU, if needed.

The data for detector configuration are to be uploaded to the CRU by DCS (Detector Control System).The volume of configuration data are relatively lower than the data coming from detector side.

The internal CRU module specific registers are to be configured properly prior data acquisition.

The large amount of bi-directional data flow from CRU to O^2 can be achieved with PCIe based DMA (Direct Memory Access) protocol as PCIe offers large bandwidth. PCIe Base Address Register (BAR) based register access could be the solution to configure registers of internal modules of CRU. The points discussed above guided us for the preliminary feasibility study of PCIe-DMA.

3.PCIe-DMA design overview and limitation

Among various designs available with Altera, we have selected one PCIe based DMA design [1], shown in FIG 1 as it provides register based interface for quick development. The design involves loopback flow of data over PCIe interface using DMA engines.

The design has three main blocks - DMA modules attached with PCIe protocol stack, Descriptor controller and Card memory. The working principle of DMA is based on descriptor tables. The source, destination addresses and transfer size for data movement are specified in that descriptor table.As per the design, Read DMA module will copy the descriptor tables from system memory to the card side First in First Out (FIFO) memory. Descriptor controller instructs DMA module as per the descriptor table to move the data from source memory to destination memory. When

[*]Electronic address: `sanjoym@jcbose.ac.in,sanjoy.mukherjee@cern.ch`

all the descriptors get executed, DMA module updates the system status memory by writing the value 1. Data and control information are communicated through PCIe packet based protocol.

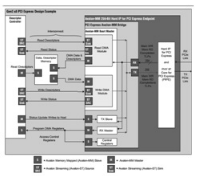

FIG. 1: Altera PCIe-DMA design

Next,we brief the limitations of target design and simulation model which can be considered as challenges in the evaluation process. The loopback data flow of the design restricts us to push the external data. The limited card memory was one of the major constraints for large data movement. The descriptor table and DMA register configuration used in simulation model were not properly configured. We managed to overcome the shortcomings of the design, as discussed in the next section, during evaluation process.

4.PCIe-DMA design evaluation process and result

Inter-connectivity among the modules was being re-arranged to push external data and break the loop. One data emulator module was also being developed to generate different external data patterns.

The Root Complex Simulation Model that emulates the PC side PCIe was not properly configured for evaluation. We made it ready for evaluation by creating user defined descriptor tables, modifying the value of different DMA registers and other user registers and re-writing the data flow model.

The module specific register configuration through PCIe-BAR was being evaluated successfully with updated design and simulation model. We used the same logic also for configuring the DMA registers.

The entire 32KB card memory was moved successfully to the system memory of the simulation model using various lengths of descriptor table and different transfer sizes. Later, we fixed one optimum length of table and transfer size that gave us better DMA performance.

Repetitive 32KB data movement is known as DMA rolling that was achieved by means of status memory polling. Register configuration involves PCIe protocol layers delay that in turn, creates dead time between two DMA data movement and lower the performance.

We reduced the dead time and increased the performance by looping over descriptor tables such that there will be pipelined flow of descriptors that means continuous DMA data movement. The final DMA performance was achieved by looping over entire (128) descriptor table entries was 50 Gb/s in Gen3 x8 lane mode.

There are two ways through which we can upload DCS data in CRU via PCIe interface. One is using DMA engines and another is writing an array of registers. We used the later. It took 1131 us to write 8K, 32 bit data which was sufficient to fulfill our requirement.

The logic used at the time of simulation was validated using Altera Arria X development kit having x8 lane. We mounted the card in the PCIe slot of server having Scientific Linux and executed the software application developed in 'C' programming language based on the same logic.

We have embedded small piece of code in the software application for debugging and validating the data received over PCIe interface. Unnecessary loading in status checking loop affects the DMA performance as the software runs sequentially and the firmware runs in parallel.

5.Summary and outlooks

The evaluation process and the corresponding results will be presented at symposium. As evaluation result was compliant with our needs, we decided to develop further user logic based on this design. We have validated our logic using PCIe card having x8 lanes.Logic development and validation with PCIe40 card having x16 lanes are yet to be done.

References

[1] https://www.altera.com/products/reference-designs/ip/interface/m-pci-express-refdesigns.html