ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THBPL06

HIGH PERFORMANCE RDMA-BASED DAQ PLATFORM OVER PCIE ROUTABLE NETWORK

W. Mansour†, N. Janvier, P. Fajardo, ESRF, Grenoble, France

Abstract

Current and upcoming 2D X-ray detectors for synchrotron radiation applications are capable of producing data rates in the range of 1-100 GB/s. Existing industrial protocols do not provide suitable data acquisition solutions for handling efficiently such a high-throughput data streams. A generic and scalable RDMA-based data acquisition platform called RASHPA, designed to address these detector needs, is introduced in this paper. The FPGA implementation as well as the Linux-based software stack is detailed. Finally, the paper presents a demonstrator integrating RASHPA over a routable PCIe network.

INTRODUCTION

The improvements of integrated circuits manufacturing technologies and processes, applied to the last generations of 2D X-ray detectors, result in a significant increase in the produced data rates. The ESRF has undertaken the implementation of a generic and scalable data acquisition framework as one of key essential components for the development of new advanced high performance detectors for scientific applications [1].

One of the key and specific features this new framework is the use of remote direct memory access (RDMA) for fast data transfer. RDMA consists on the transfer of data from the memory of one host or device into that of another one without any intervention of the CPU. This permits high-throughput, low-latency networking. Companies are investing more and more into this feature, already applied to high performance computing, by integrating it into their network cards and communication adapters. Some of the available technical solutions are Infiniband [2], RDMA over Converged Ethernet (RoCE) [3] and internet Wide Area RDMA Protocol (iWARP) [4], to name few.

The Peripheral Component Interconnect Express (PCIe) [5] bus is a high-speed serial computer expansion bus standard, which uses shared parallel bus architecture, in which the PCI host and all devices share a common set of addresses. In other words, PCIe have a direct access to the memory of the system. For RDMA based applications, PCIe is an ideal option due to its reliability, scalability, low latency and native integration. In fact, over the years, PCIe has become the default peripheral interconnect of x86 based platforms, it has a built-in support for control flow, data integrity and packet ordering, thus no need for additional protocol layers. In addition to that, its bandwidth can be adjusted depending on the number of used lanes. Many initiatives have been launched to start using the PCIe-based RDMA to accelerate communications in data centres [6-8].

Despite these benefits, the limited availability of PCIe over cable products and the lack of standardization of optical cabling form is still an issue [1].

RASHPA (RDMA-based Acquisition System for High Performance Applications) is the generic framework for detector data acquisition currently under development at the ESRF. It is optimised for the transfer of 2D detector data, i.e images, and it relies completely on RDMA mechanisms. When implemented over a low latency PCIe over cable network, RASHPA is able to push data, at very high speed into the address space of one or several backend computers. The scheme provides a high standardization level in the data transmission pipeline from the detector up to the software application for further processing, visualization or storage.

This paper details the FPGA implementation of RASHPA at the detector side, as well as its carefully designed Linux software stack. In addition to that, a demonstrator integrating RASHPA over a routable PCIe over cable network is also presented.

The paper is organised as follows: Section 2 introduces the RASHPA concept and architecture. Section 3, presents the hardware and software implementation of RASHPA and the interaction between both. Section 4 shows the RASHPA prototype and experimental results. Conclusions and future perspectives are discussed in section 5.

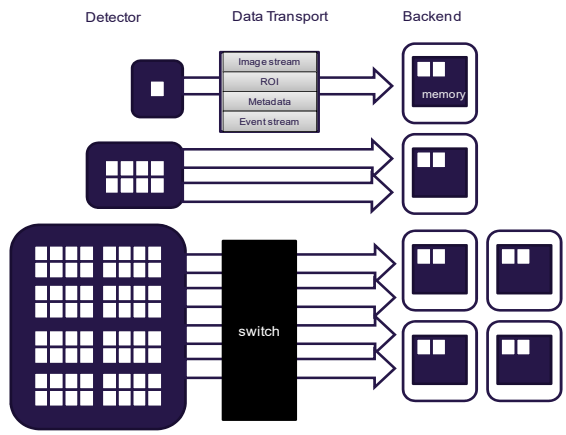

Figure 1: Block diagram of a RASHPA network.

RASHPA CONCEPT

As a framework, RASHPA defines a set of functional concepts as well as the hardware and software interfaces. It also implements a generic, non hardware specific middleware running on the backend computers. For practical

† mansour@esrf.fr

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THBPL06

implementation, the ESRF is also developing a set of hardware blocks that allow building and integrating the required functionality in the in-house developed detectors in conformity with the RASHPA specifications.

RASHPA allows detectors to push data (images, regions of interest (ROI), metadata, events etc...) directly into one or more backend computers as depicted in Figure 1. RASHPA's main properties are its scalability, flexibility and high performance. It is intended to have an adjustable bandwidth that can be compatible with any backend computer. Although the examples in this paper are based on PCIe, RASPHA is in principle independent of a particular type of data link, as long as the data link supports RDMA and routability over a network to dispatch the data to multiple destinations.

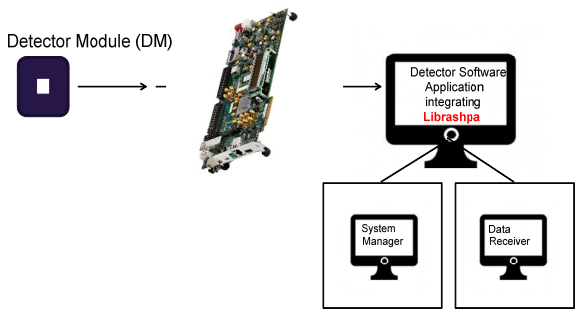

Figure 2: RASHPA behavioral diagram.

RASHPA allows Detector Modules (DM) to push data into Backend Computers (BC). Although the usual data destinations are RAM system memory buffers, other possible destinations under investigation are Graphical Processing Units (GPU), coprocessors and disk controllers. BC's receiving data from the DM are called Data Receivers (DR). BC's configuring and initializing RASHPA are called System Managers (SM) which can act as a DR as well. *librashpa*, a software library is responsible of providing to the detector software application the required functions to configure initialize and control RASHPA hardware. *librashpa* should be integrated in all the DRs and the SM. It does not provide any kind of inter-communications between multiple DRs and does not have any link with the DM. This behaviour is illustrated in Figure 2.

Figure 3 is a conceptual diagram of how the RASHPA framework fits the data transmission pipeline of a basic detector system consisting of a unique DM and a single BC. From a hardware point of view, the RASHPA controller consists of specific logic interfacing the detector readout electronics as well as a set of hardware blocks handling data transmission. These blocks are known as channels. Two types of configurable channels can be identified in RASHPA: data and event channels. Each data channel, for which there can be multiple instances in a single RASHPA controller, is responsible of transferring detector data to a pre-configured address space within one or several DRs. A single event channel should acknowledge the data receiver or system manager about any event occurring at the RASHPA controller. Typical asynchronous events are errors, end of transmission conditions, etc.

The main RASHPA components in the detector head are:

- A scheduler in charge of generating and dispatching memory transfer requests (data and event channels).
- DMA engines to access workstation memory.
- An embedded CPU in charge of handling configuration and control requests from the workstation.

In addition to that, RASHPA defines and implement a *librashpa* on the DR that initialises and manages the data transfer process and provides a standard programming interface to client applications.

RASHPA IMPLEMENTATION

Two types of implementation should be identified: hardware and software implementations.

From the hardware point of view, the FPGA implementation of RASHPA is independent of the type of high speed data link used to perform the data acquisition process.

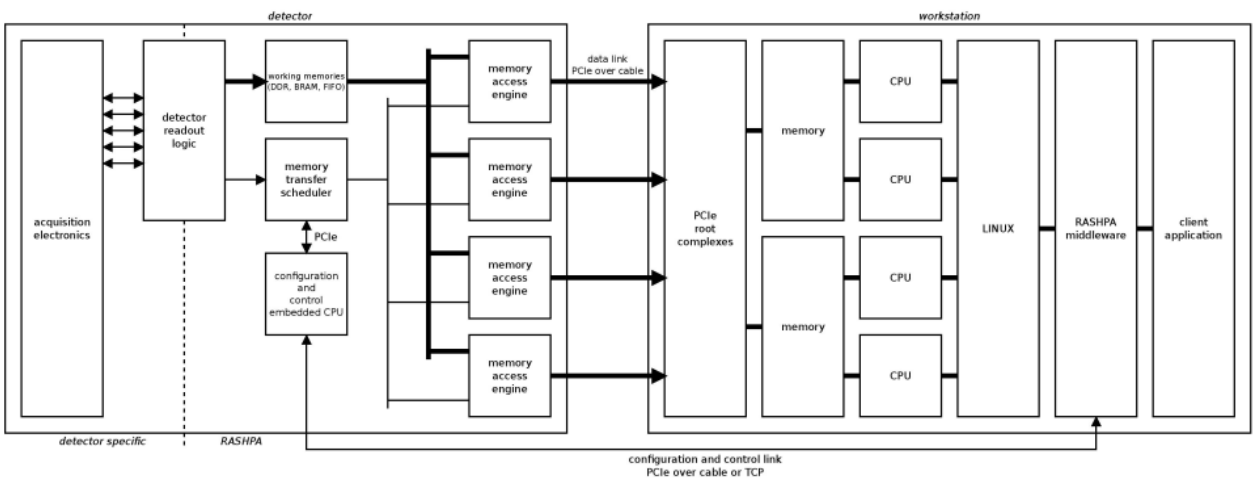

Figure 3: Architecture of a basic RASHPA system.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THBPL06

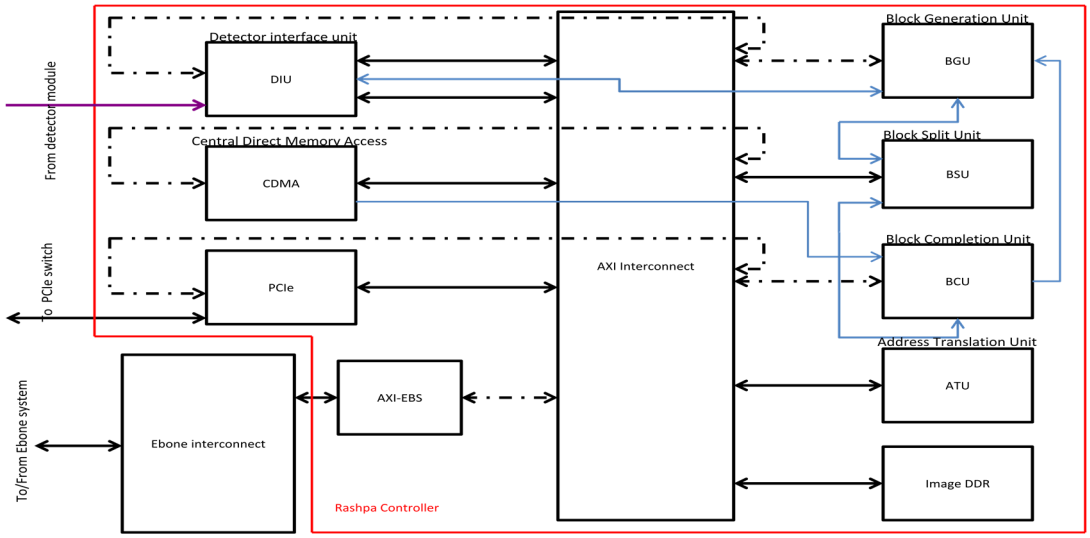

Figure 4: Architecture of a RASHPA FPGA implementation.

The FPGA implementation presented in the following paragraphs is based on an AXI subsystem implemented on a Xilinx KC705 development board [9]. It is considered as a separate intellectual property (IP) that can be easily integrated within the ESRF control systems. In this implementation, a PCIe over cable data link was chosen for the reasons mentioned in the introduction.

FPGA Implementation

The AXI subsystem implementing RASHPA on the KC705 development board, Figure 4, consists of several IPs, some from Xilinx and others designed ad-hoc:

- Detector Interface Unit (DIU): This unit reads the image data and store them in a DDR3 memory on board.
- Block Generation Unit (BGU): It contains all the data channels that are configured prior starting the data transfer process. This unit outputs 256-bit packets containing all the information about each transfer, such as source address in the DDR, destination address index of a physical address in an address translation unit, bytes to transfer, etc.
- Address Translation Unit (ATU): This is an internal memory containing the physical address of the available memory buffers on each DR.
- Block Split Unit (BSU): It is responsible of reading and analyzing BGU's output data, getting corresponding addresses from the ATU, and configuring the Central Direct Memory Access (CDMA), and the PCIe IPs.
- CDMA: a Xilinx built-in direct memory access IP.
- PCIe: a Xilinx built-in transfer layer IP.

All the previously described IPs are configured through the e-bone interconnect [10] which is the ESRF standard interconnect used for control applications.

PCIe over Cable

A RASHPA system must be able of sending data to multiple destinations, thus it requires a routable network. As this implementation has adopted PCIe for data transmission, one requires to use PCIe switches for packet routing. Such components are not so common, but one could find some of them such as the Dolphin IXS600 from dolphinics [11] and OSS-PCIe-1U-SW-x4-2.0 from one stop systems [12].

For the demonstration purpose, we made the decision to use the PXH810 board [13], presented in Figure 5 which integrates a PLX8749 PCIe switch.

The board comes with a Linux driver that is not compatible for RASHPA's application. We have then re-implemented the driver of the PLX switch in order to fit the RASHPA needs.

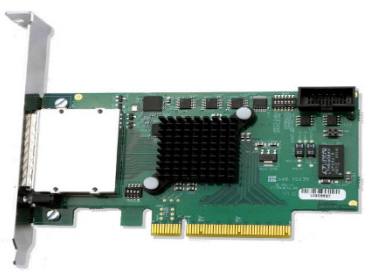

Figure 5: PXH810 board

The PXH810 board is plugged in a BC, with one PCIe cable adapter board. The adapter board links the current BC to the DM, whereas the PXH810 connects the current BC to another BC. This way, RASHPA can send data to two BCs via the PCIe switch.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THBPL06

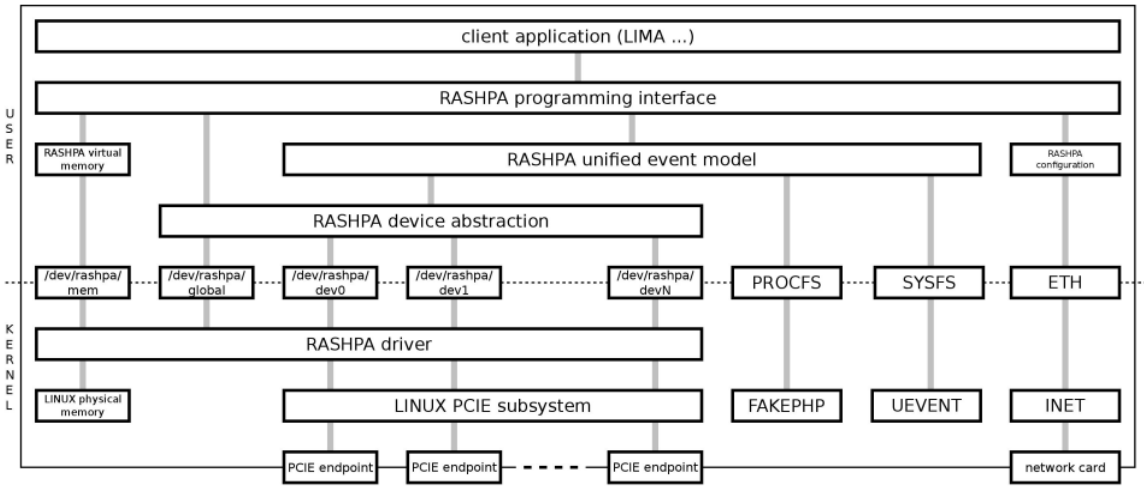

Figure 6: Structure of the RASHPA middleware.

Workstation Middleware

The workstation middleware called *librashpa*, a Linux library, is in charge of providing the client application with data transmission services through a well defined application programming interface (API). An example of client is the LIMA data acquisition and detector control library widely used at ESRF and other synchrotron radiation facilities [14]. The middleware layers depicted in Figure 6 support the following characteristics:

- Configuration: this layer is in charge of building description of the hardware such as when and where to transmit data.
- Device abstraction: Since RAHSPA is link independant, the PCIe endpoint discussed earlier can be replaced with any other data link such as Ethernet, Infiniband, etc... The device abstraction layer hides low-level details and provides the software with a generic interface to access the underlying devices. While most of the layer is implemented in user space, special operations such as interrupt handling require a thin driver.
- Memory management: RDMA transfer requires to work with the BC's physical address space, however typical client application buffers are allocated in the process virtual address space. The middleware thus provides a virtual memory allocator on top of an internally managed physical memory allocator.
- Unified Event Model: There are multiple sources of events that would make sequential programming difficult. For instance, data transmission completion is signalled by the detector using an interrupt along with auxiliary data; LINUX informs about PCIe device adding or removal using system notifications; Timers and callbacks may be registered by the client application itself... For those reasons, the middleware programming model is largely event based and provides software abstractions that unify the different event sources.

RASHPA ADVANCED PROTOTYPE

The first RASHPA prototype was developed in the frame of the European project CRISP [1]. In the first version, RASHPA supported the data transfer from a DM to one BC. It has been tested and validated using a data generator emulating the detector behaviour.

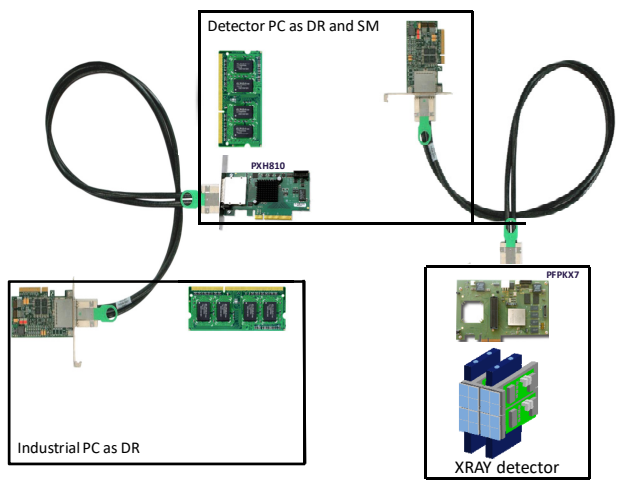

Figure 7: RASHPA advanced prototype.

In the current version, Figure 7, RASHPA has been integrated to the SMARTPIX, a Medipix3 based detector currently under development at the ESRF. In that perspective, the KC705 development board has been replaced with a commercial PFPKX7 board from Techway [15]. The new of prototype version also supports the multiple destinations feature thanks to the use of PCIe switches. Copper cables were used to build and test the routable network although fibre optics cabling is also available. In this implementation, two types of PC are

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THBPL06

used: a so-called detector PC having 64GB of internal DDR, and Gen3x16 PCIe endpoint, and an industrial PC with only 4GB on internal DDR and Gen1x1 PCIe endpoint. The PFPkx7 supports Gen2x4 PCIe, thus the link connecting the detector to the detector PC is negotiated to Gen2x4 whereas the virtual link connecting the detector to the industrial PC is limited to Gen1x1.

Due to the difficulty of producing real detector images with the current SMARTPIX hardware without an X-ray source, a Java image generator application was developed, where the ESRF logo was used as a reference image. A screen shot of an experiment is illustrated in Figure 8. In this experiment two data channels where configured to send the original image to the detector PC and a region of interest of that image to the industrial PC based on some configuration sent by the system manager to the RASHPA controller.

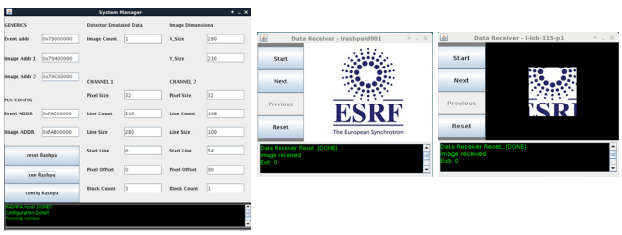

Figure 8: RASHPA results.

Table 1, shows the data throughput of the data acquisition process. Two ways were investigated to measure the data throughput. The first one measures the throughput internally within the FPGA, whereas the second one detects whenever data are available to the application.

Table 1: Measured Data throughput

	Bandwidth within the FPGA	Bandwidth when data available to the application
Detector PC Gen2x4	10.88Gbps 68%	8.18Gbps 51.15%
Industrial PC Gen1x1	1.44Gbps 71.99%	1.12Gbps 56.28%

Measured throughput when Gen2x4 PCIe endpoint is the target is 68% of the Gen2x4 maximum bandwidth when measured within the FPGA and 51.15% when measured at the application level. Similarly measured bandwidth at the Gen1x1 is 71.99% and 56.28% when measured at the FPGA and application level respectively. These losses in the data throughput are due to the configuration of the DMA which is imposed by RASHPA as well as the restriction of the PCIe packet size limited to 4Kbytes. Some other latency can be added when throughput is measured at the application level due to the processor execution time.

CONCLUSIONS AND FUTURE WORK

In this paper was described a generic and scalable RDMA-based data acquisition platform called RASHPA and its FPGA implementation as well as its carefully designed Linux-based software stack. A demonstrator integrating RASHPA with the ESRF SMARTPIX 2D X-ray detector over a routable PCIe network was presented.

The development and the realisation of the prototypes allowed to validate the basic RASHPA concepts and objectives. The obtained results showed the capability of RASHPA to send detector images as well as a region of interest of these images to multiple destinations simultaneously. Data throughput was around 53% when measuring the bandwidth at the application level whereas when measuring it at the FPGA level it reached 70%.

In future work is planned prototyping an RDMA-based protocol over 100G Ethernet network in order to replace the PCIe link to overcome the strong limitations of available commercial off-the-shelf hardware, in particular high performance switches. Another aspect under study is the application of the RDMA techniques in RASHPA to send data directly from the detector to GPUs in the data receivers.

ACKNOWLEDGEMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 654220.

REFERENCES

[1] F. Le Mentec *et al.*, *RASHPA : A data acquisition framework for 2D X-Ray detectors*, *ICALEPCS proceeding*, San Francisco, 2013

[2] Mellanox Technologies, *Introduction to Infiniband, white paper, Document Number 2003WP.*

[3] Mellanox Technologies, *RDMA/ROCE solutions,* https://community.mellanox.com/docs/DOC-2283.

[4] Intel corp., *Understanding iWARP : Delivering low latency Ethernet,* 2015.

[5] PCI-SIG, *PCI Express base specification revision 3.0, November* 10, 2010.

[6] Zang *et al.*, PROP: using PCIe-Based RDMA to accelerate rack-scale communications in data centers, *IEEE International conference on Parallel and Distributed Systems (ICPADS),*, January 2016, DOI:10.1109/ICPADS.2015.65

[7] Boffi *et al.*, PCIe-based network architectures over optical fiber links: an insight from the advent project, *18th Italian National Conference on Photonic technologies (Fotonica 2016)*, June 2016, Rome, Italy, DOI:10.1049/cp.2016.0883

[8] J.F. Zazo *et al.*, A PCIe DMA engine to support the virtualization of 40 Gbps FPGA-accelerated network appliances, *IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig),* Dec. 2015, Mexico City, Mexico, DOI:10.1109/ReConFig.7393334

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THBPL06

[9] https://www.xilinx.com/products/boards-and-kits/ek-k7-kc705-g.html

[10] https://www.ohwr.org/projects/e-bone

[11] http://www.dolphinics.com/products/IXS600.html

[12] https://www.onestopsystems.com/product/pcie-x4-gen2-10-port-switch

[13] http://www.dolphinics.com/products/PXH810.html

[14] A. Homs *et al.*, LIMA: Acquiring data with imaging detectors, in *Proc. ICALEPCS'11,* Grenoble, France, Oct. 2011, paper WEMAU011, pp. 676-679.

[15] http://www.techway.fr/fiche-detaillee?id_categorie=1&id_produit=118