PAPER • OPEN ACCESS

Latest generation interconnect technologies in APEnet+ networking infrastructure

To cite this article: Roberto Ammendola *et al* 2017 *J. Phys.: Conf. Ser.* **898** 082035

View the article online for updates and enhancements.

IOP Conf. Series: Journal of Physics: Conf. Series **898** (2017) 082035 doi:10.1088/1742-6596/898/8/082035

Latest generation interconnect technologies in APEnet+ networking infrastructure

Roberto Ammendola[1], Andrea Biagioni[2], Paolo Cretaro[2], Ottorino Frezza[2], Francesca Lo Cicero[2], Alessandro Lonardo[2], Michele Martinelli[2], Pier Stanislao Paolucci[2], Elena Pastorelli[2], Davide Rossetti[3], Francesco Simula[2] and Piero Vicini[2]

[1] INFN, Sezione di Roma Tor Vergata, Italy
[2] INFN, Sezione di Roma, Italy
[3] NVIDIA Corp, Santa Clara, California

E-mail: `michele.martinelli@roma1.infn.it`

Abstract. In this paper we present the status of the 3rd generation design of the APEnet board (V5) built upon the 28nm Altera Stratix V FPGA; it features a PCIe Gen3 x8 interface and enhanced embedded transceivers with a maximum capability of 12.5Gbps each. The network architecture is designed in accordance to the Remote DMA paradigm. The APEnet+ V5 prototype is built upon the Stratix V DevKit with the addition of a proprietary, third party IP core implementing multi-DMA engines. Support for zero-copy communication is assured by the possibility of DMA-accessing either host and GPU memory, offloading the CPU from the chore of data copying. The current implementation plateaus to a bandwidth for memory read of 4.8GB/s. Here we describe the hardware optimization to the memory write process which relies on the use of two independent DMA engines and an improved TLB.

1. Introduction

Hybrid systems are emerging as an efficient solution in the HPC arena, with an abundance of approaches for integration of accelerators into the system (*i.e.* GPU, FPGA). In this context, one of the most important features is being able to address the accelerators, whether they be local or off-node, on an equal footing compared to the CPU. Correct balancing in how the network supports this kind of transfers in inter-node traffic becomes tantamount to global system efficiency and high performance.

The goal of the APEnet project is the design and development of a point-to-point, low-latency and high-throughput interconnect adapter, to be employed in High Performance Computing clusters with a 3D toroidal network mesh. As regards the software flow, the board can be schematized as in figure 1; a more hardware component-oriented schematization is in figure 2. A first version of APEnet+ V5 was presented in [1], discussing the software and hardware design for the board and showing the improved results in comparison to the previous Gen2 version. We also showed that doubling the DMA engines responsible for transmission we succeeded in almost saturating the available bandwidth on the transmission side of the communication. In this paper we present the status of the 3rd generation design for the board (called V5) built around the 28 nm Altera Stratix V FPGA; it features a PCIe Gen3 x8 interface and enhanced embedded transceivers with a maximum capability of 12.5 Gbps each. The network

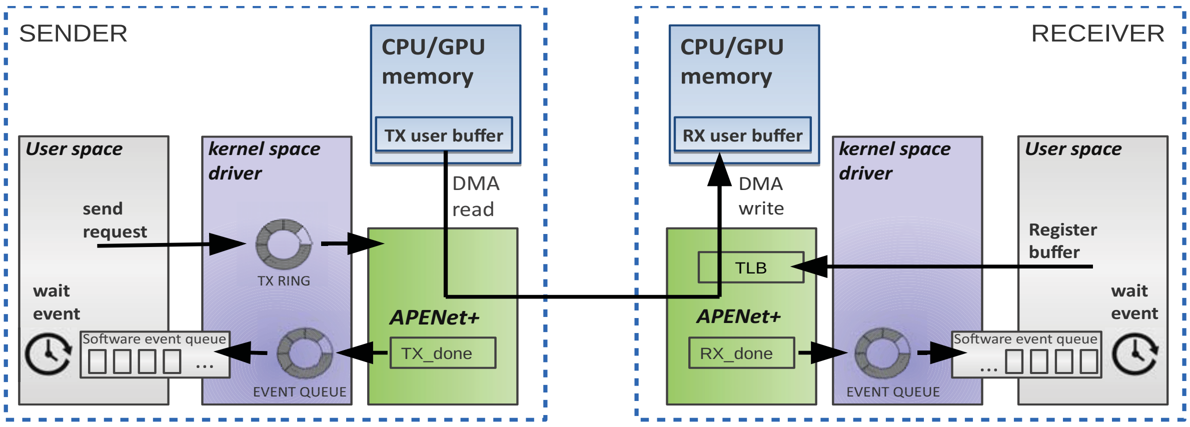

Figure 1. Software architecture of APEnet+ V5 NIC.

architecture was designed in accordance with the Remote DMA paradigm. APEnet implements the NVIDIA GPUDirect RDMA and V2 (peer-to-peer) protocols to directly access GPU memory, overcoming the bottleneck represented by transfers between GPU/host memory. The APEnet+ V5 prototype is built upon the Stratix V DevKit with the addition of XpressRICH3-AXI, a PLDA Soft IP core supporting multi-DMA engines. Zero-copy communication is assured by the possibility of DMA-accessing either host and GPU memory and offloading the CPU from the chore of data copying. The current implementation peaks at 4.8 GB/s for memory read bandwidth. Herein we describe the hardware optimization — relying on two independent DMA engines and an improved TLB (Translation Lookaside Buffer) — dedicated to memory write and the characterization of the software enhancements put in place to fully exploit such hardware capabilities. The APEnet+ V5 prototype offers three (X, Y and Z) APElink high performance channels; the X channel is implemented by bonding 4 lanes of the QSFP connector available on the board while the Y and Z channels were implemented onto the HSMC interface. In the following, details are given about the Transmission Control Logic (TCL), which manages the data flow by encapsulating packets into a light, low-level, word-stuffing proprietary protocol able to detect transmission errors via CRC. The current implementation of the APElink TCL is able to sustain the link bandwidth of about 5 GB/s at an operating frequency of $\sim$312 MHz. Latency and bandwidth measurements on host-to-host and GPU-to-GPU between two servers will be provided. The hardware upgrade to two independent DMA engines and an improved TLB to speed up memory writes will also be described. Finally, as regards future developments, we discuss work undertaken towards compliance with next generation FPGAs with hard IP processors on board.

2. Related work

Researchers are turning their attentions to the so called direct communication for GPU [2], FPGA [3], and all other processing units to improve and optimize the communication between these accelerators. Many technologies based on PCIe protocol have been proposed, such as the InfiniBand Architecture (IBA) — a fabric which is an industrial standard for high bandwidth/low latency communication — and other protocols also available for FPGA like Serial RapidIO, Fibre Channel, PCI Express Fabric [4] and many more. On the other hand, designing custom interconnection hardware allows the optimization for a specific workload or task. A natural extension of this idea would be using reconfigurable hardware like an FPGA — *e.g.* the TH2 Express interconnect [5], employed in the Tianhe-2 supercomputer currently holding second place in the TOP500 [6] —, simple encapsulation of other protocols in FPGA-to-FPGA interconnects like in UPI (Unified PHY Interface) system [7], or custom interconnects as in BlueLink [8] or in Data Acquisition Systems, see for example the NaNet [9] and Felix [10] projects. FPGAs allow for simple addition of hardware “modules”, *e.g.* the Translation Lookaside Buffer (TLB), whose description can be found in [11]. TLBs are often used for virtual-to-physical address translation, which is a common task in modern operating systems, thus integrated in processors Memory Management Units, as in [12]. In this paper, we updated the TLB module for bigger storage capacity while the main concepts of the IP were left untouched. Whatever task they are put to — whether it be providing a general purpose, modular NIC system component like in our case or a computing accelerator in its own right — FPGA-based devices also need high throughput towards their own hosts, often requiring non-trivial software infrastructure.

3. APEnet+ hardware

The architecture of APEnet+ V5, as depicted in figure 2, consists of three main blocks: *Network Interface*, *Router* and *Torus Link*.

The *Network Interface* manages the data flow between PCIe bus and APEnet+; on the TX side it collects data coming from the PCIe port and forwards it to the *Router* block while on the RX side provides hardware support for the RDMA programming model. The APEnet+ packets have 256 bits-wide header, footer and data words with variable size payload. The *Network Interface* uses a 3rd party device, a Gen3-compliant soft core by PLDA. The *Router* manages dynamic links between APEnet+ ports: it inspects the packet headers and translates their destination addresses into a proper path across the switch according to a dimension-ordered routing algorithm, managing conflicts on shared resources. The *Torus Link* allows point-to-point, full-duplex connections of each node with its neighbours, encapsulating the APEnet+ packets into a light, low-level, word-stuffing protocol able to detect transmission errors via CRC. It implements 2 virtual channels and proper flow-control logic on each RX link block to guarantee deadlock-free operations. APEnet+ V5 was developed on the Altera

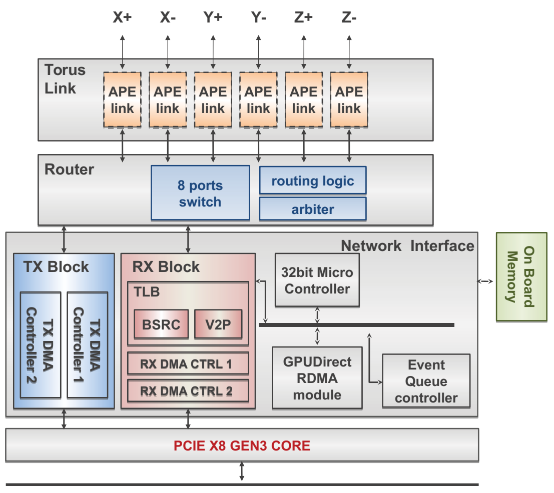

Figure 2. Outline of the main logic blocks of APEnet+ architecture.

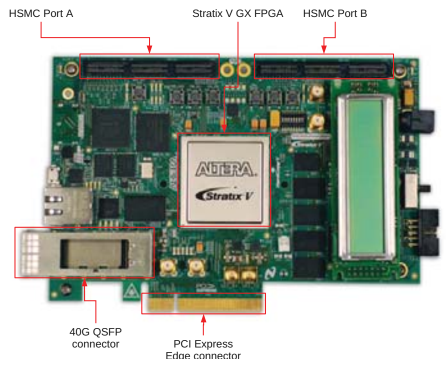

Figure 3. Altera DK-DEV-5SGXEA7N Development Board.

DK-DEV-5SGXEA7N development kit (see figure 3), a complete design environment featuring a 28 nm FPGA (*Stratix V GX 5SGXEA7K2F40C2N*). This setup features a PCIe connector tagged for Gen3 connection speeds of 8.0 Gbps/lane with the host machine and powering the board, a 40G QSFP connector and two HSMC ports which allow the interconnection between boards. We allotted four DMA channels arranged in a 2 RX/2 TX split to the task of increasing the bandwidth between GPU/CPU and APEnet+ across the PCIe interface. This choice was dictated by the need to mitigate the two latencies that incur first when setting up a DMA and, after the DMA is setup, waiting for data to start flowing. By means of issuing an alternating 'ready' signal between two channels when setting up the DMA, the achieved improvements were remarkable, either in transmission and reception, as shown in section 5. Another significant upgrade to our architecture which was made possible by the switch to a 28 nm FPGA was the up-sizing of the Look-Up Tables (LUTs) and the CAMs employed within the hardware implementation of the TLB; the effects of this upgrade are apparent in the bandwidth graphs (see section 5).

4. Hardware-Software interaction overview

In this section we provide a brief overview on the hardware-software interplay that better explains the test results in section 5. Let us assume, focusing on figure 1, that the user-space application needs to send a certain amount of data (stored in a local TX buffer) to a remote, already present, RX receiving buffer (irrespective to whether the source or the destination are in GPU memory, the behavior is the same). By calls to a provided library, the *send request* from user-space reaches the kernel-space driver where it is translated (according to the size of the sending buffer) into one or more *descriptors* that contain all required information for the data transfer by the APEnet NIC (*i.e.* source physical address, destination address etc.). Descriptors are then inserted in the so called *TX RING*, a circular list stored within the DMA-accessible host memory. The presence (and number) of new ready descriptors is then notified to the NIC by a doorbell register. Each descriptor matches a packet, with packets sized the same as *memory pages*; this is in order to better align with the abstraction of the virtual memory paradigm where a single message insisting on a large buffer (which is potentially scattered among many non-contiguous memory pages in memory) must be sent. The pages that make up the buffer simply translate to packets and corresponding descriptors. The data to be transferred is actually DMA-read by the APEnet hardware at the physical addresses given in the *descriptors.* After the *memory loading* phase, the hardware issues a *TX done* completion by DMA-writing it in the so-called *event queue* memory region and signaling an interrupt to the host that wakes up the device driver, which is in charge of acknowledging them and copying them into an user-space software queue. When a packet lands, APEnet inspects the destination virtual address and retrieves the associated physical destination address by means of a custom hardware TLB. The list of correspondences between the virtual and physical addresses is created during a *buffer registration* initialization phase, in which the user-space application uses the Nios II micro-controller embedded into the FPGA to instruct the TLB. The retrieved physical address is then used by the board to DMA-write the data into memory. The last step is a DMA-write of the *RX done* completion and assertion of a *new packet* interrupt line. Furthermore, we also implemented a polling mechanism to bypass all the interrupts while the driver keeps polling on the *next event* memory location.

5. Synthetic test results

All development were performed on the Test and Development (T&D) platform which is composed by two X10DRG-Q SuperMicro servers, connected by the cables as in table 1, populated with Haswell E5-2620@2.40 GHz CPUs, hosting the DK-DEV-5SGXEA7N Development Board as APEnet+ V5 and NVIDIA Tesla K40m GPUs; debugging was aided by a Tektronix TLA-7012 Gen3 Logic Analyzer. Both systems use the NVIDIA driver version 367.57 and CUDA version 7.5. We performed two sets of tests to measure latency and bandwidth.
In both tests the receiver starts by sending the *receiving buffer* virtual address to the sender using two alternatively different methods. The first is an out-of-band Ethernet connection (the

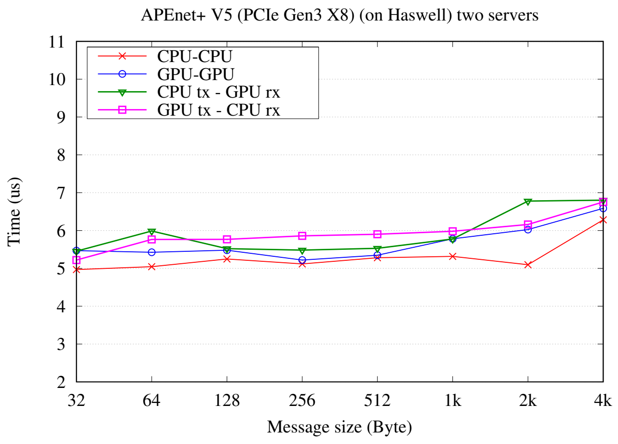

Figure 4. Latency for small packet is barely influenced by data size.

Table 1. APEnet+ BER measurements on Altera 28 nm FPGA.

Cable		BER	Data Rate
10 m Mellanox optical cable	$<$	2.36 E-14	11.3 Gbps
1 m Mellanox copper cable	$<$	1.10 E-13	10.0 Gbps

simpler way) while the second uses a list of ”streaming” buffers. In the latter option, the senders use the fictitious 0X00000000 virtual address as destination virtual address, then the receiving side put the new data into the first unused pre-registered buffer.

Latency tests are carried out by sending a packet in a ping-pong fashion: when a “ping” packet from the sender is received on the receiver side, a “pong” answer of fixed size is sent back. Time elapsed from the “sent” event to the “receive” one of the “pong” packet is measured by using the Intel x86 internal cycles counters accessed by the *Time Stamp Counter (TSC)* register. As expected, results (shown in figure 4) are barely influenced by data size for small packets.

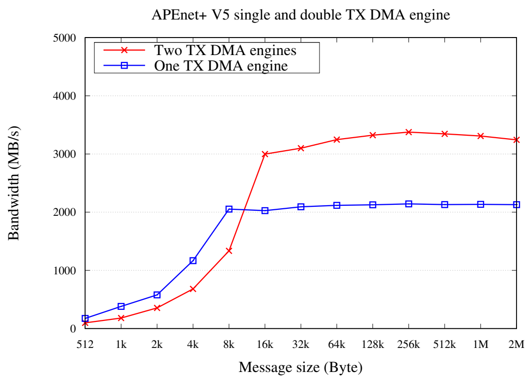

Figure 5. In this loop-back benchmark, physical addresses is used for both source and destination, bypassing the virtual-to-physical translation.

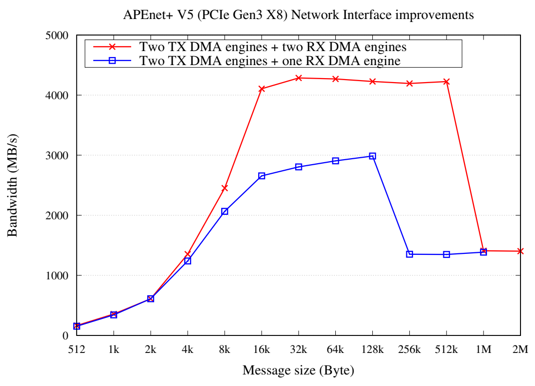

Figure 6. In this loop-back benchmark, we doubled the engines on the RX side and used virtual address, as can be seen by the TLB miss, which starts around 1M for the new TLB and around 256kB for the old version.

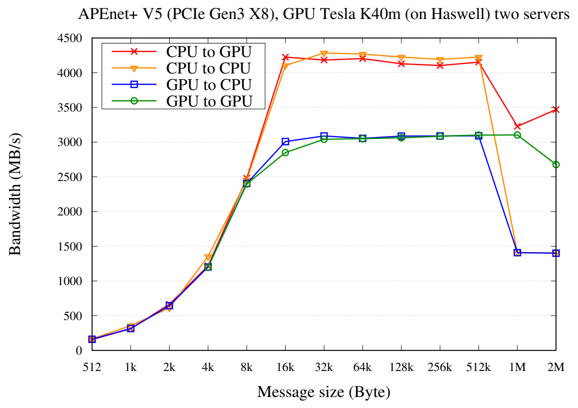

Figure 7. Bandwidth measured using the new TLB and double DMA engines on the RX side.

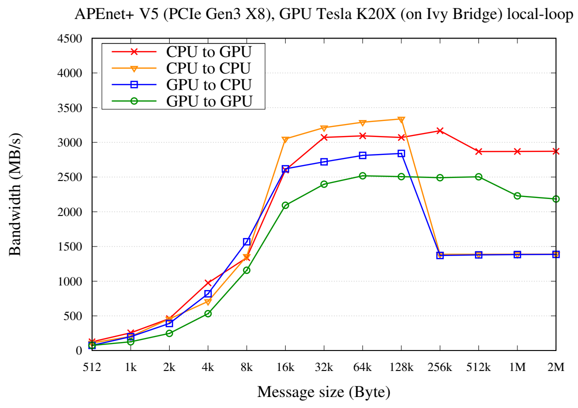

Figure 8. Bandwidth measured using the old version of the TLB. Miss-in-cache starts around 256kB for CPU communication. This effect is absent in the GPU plots because of the larger memory pages.

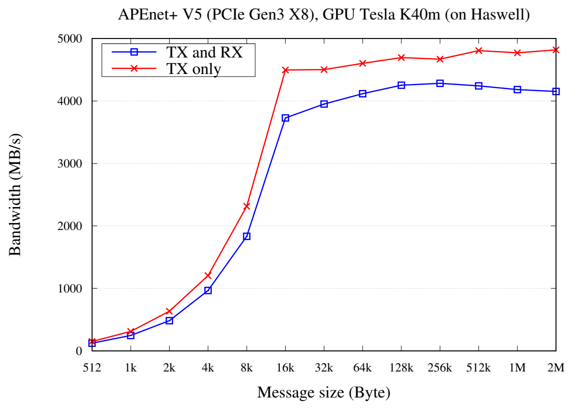

Figure 9. Performance gain flushing away the RX part of the communication spots the need for RX optimization.

In the bandwidth tests, multiple packets are sent at once, but only the last one is 'ACK'ed by the receiving side. Again, time from the "sent" to the "receive" event is measured. We made some preliminary tests on a single machine with one APEnet+ V5 in loop-back configuration to measure the improvement in performance gained by the use of two DMA engines. Results are shown in figure 5 and figure 6.

Furthermore, we used two different servers to take the same measurement; results for every combination of host/GPU memory and TX/RX modes are shown in figure 7. As it can be seen, the GPU TX path is slower than the host one, probably due to the DMA engines/IOMMU limitations, while best performance is achieved by using the host memory as source.

The comparison of the results to figure 8 shows an improvement in performance by using a second DMA engine, even if it only changes the point in which the application starts hitting the performance degradation. As regards the TLB improvements see figure 7 for and figure 8 for a bandwidth comparison between the old and the new version. We are working on the improvement of the receiving side since, as can be seen in figure 9, it currently represents the limiting factor of the overall communication performance.

6. Conclusion and future work

We have presented the hardware enhancements for the APEnet+ V5 board — doubling its DMA engines and integrating an extended TLB — showing encouraging results in the goal of pushing its PCIe channel saturation closer to the Gen3 x8 standard theoretical bandwidth. Furthermore, evaluations of the recently introduced SOC architecture, is strongly in progress in order to exploit the capabilities of the new hardened multi ARM cores embedded in the last generation FPGA.

References

[1] Ammendola R, Biagioni A, Frezza O, Cicero F L, Lonardo A, Martinelli M, Paolucci P S, Pastorelli E, Rossetti D, Simula F, Tosoratto L and Vicini P 2015 *Journal of Physics: Conference Series* **664** 092017 URL `http://iopscience.iop.org/article/10.1088/1742-6596/664/9/092017`

[2] Fröning H and Oden L 2013 *IEEE International Conference on Cluster Computing 2013* (Indianapolis, IN, USA) URL `http://www.ziti.uni-heidelberg.de/ziti/uploads/ce_group/2013cluster-ggas_public.pdf`

[3] Neuwirth S, Frey D, Nuessle M and Bruening U 2015 *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on* pp 627–638

[4] Mayhew D and Krishnan V 2003 *11th Symposium on High Performance Interconnects, 2003. Proceedings.* pp 21–29

[5] Pang Z, Xie M, Zhang J, Zheng Y, Wang G, Dong D and Suo G 2014 *Frontiers of Computer Science* **8** 357–366 ISSN 2095-2228 URL `http://dx.doi.org/10.1007/s11704-014-3500-9`

[6] `http://www.top500.org`

[7] An W, Jin X, Du X and Guo S 2016 *2016 18th International Conference on Advanced Communication Technology (ICACT)* pp 586–591

[8] Markettos A T, Fox P J, Moore S W and Moore A W 2014 *24th International Conference on Field Programmable Logic and Applications, FPL 2014, Munich, Germany, 2-4 September, 2014* pp 1–8 URL `http://dx.doi.org/10.1109/FPL.2014.6927472`

[9] Ammendola R, Biagioni A, Frezza O, Lamanna G, Cicero F L, Lonardo A, Martinelli M, Paolucci P S, Pastorelli E, Pontisso L, Rossetti D, Simula F, Sozzi M, Tosoratto L and Vicini P 2015 *Journal of Physics: Conference Series* **664** 092002 URL `http://stacks.iop.org/1742-6596/664/i=9/a=092002`

[10] Anderson J, Borga A, Boterenbrood H, Chen H, Chen K, Drake G, Donszelmann M, is D F, Gorini B, Guest D, Lanni F, Miotto G L, Levinson L, Narevicius J, Roich A, Ryu S, euder F S, Schumacher J, Vandelli W, Vermeulen J, Wu W and Zhang J 2016 *2016 IEEE-NPSS Real Time Conference (RT)* pp 1–2

[11] Ammendola R, Biagioni A, Frezza O, Lo Cicero F, Lonardo A, Paolucci P S, Rossetti D, Simula F, Tosoratto L and Vicini P 2013 *Field-Programmable Technology (FPT), 2013 International Conference on* pp 58–65

[12] Shamani F, Sevom V F, Ahonen T and Nurmi J 2016 *Microprocessors and Microsystems* ISSN 0141-9331 URL `//www.sciencedirect.com/science/article/pii/S0141933116303994`