Poster Abstract: Chaining of hardware accelerated Virtual Network Functions in PCIe Environments

Ralf Kundel
Multimedia Communications Lab, Technische Universität Darmstadt, Germany
ralf.kundel@kom.tu-darmstadt.de

Tim Burkert
Zoi TechCon GmbH,
Stuttgart, Germany
tim.burkert@zoi.de

Carsten Griwodz
Department of Informatics,
University of Oslo, Norway
griff@ifi.uio.no

Boris Koldehofe
Multimedia Communications Lab, Technische Universität Darmstadt, Germany
boris.koldehofe@kom.tu-darmstadt.de

Abstract

The growth of compute-intensive applications causes an increasing demand for computing resources in both data centers and underlying networks. To satisfy these computing and networking demands, hardware-accelerated computing with GPUs and FPGAs becomes more and more prevalent. However, current approaches of accelerated Virtual Network Functions (VNF), typically based on PCIe accelerator cards, suffer from many superfluous data transfers between the memory of hosts and accelerator devices.

In this work we propose "host bypassing" for chained hardware-accelerated network functions, which reduces memory copying to a minimum, which increases throughput and reduces latency.

Keywords VNF, PCIe, FPGA, GPU, hardware acceleration, offloading, network function chaining

ACM Reference Format:
Ralf Kundel, Tim Burkert, Carsten Griwodz, and Boris Koldehofe. 2019. Poster Abstract: Chaining of hardware accelerated Virtual Network Functions in PCIe Environments. In *20th International Middleware Conference Demos and Posters (Middleware Demos and Posters '19), December 9–13, 2019, Davis, CA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3366627.3368112

1 Motivation

Network Function Virtualization (NFV) has arisen in the past years in many networking domains. This paradigm describes the execution of network functions in virtualized software environments, typically based on standard x86-servers, instead of purpose-built devices[3]. Besides the independence from the vendors of these purpose-built devices, NFV enables a much more flexible provisioning and chaining of several network functions. An example for such a chain may consist of network address translation, firewall filtering and load balancing, which should be executed in sequence for each processed network packet. Today, such NFV chains are typically implemented on x86 processors, where on processor core is dedicated to each network function, in addition to two croes for sending and receiving packets. Executing these chains of virtualized software functions provides flexibility but consumes much more energy than purpose-built devices with fixed functionality implemented in hardware on Application Specific Integrated Circuits (ASIC).

The authors of [1] have determined the total energy consumption of the Internet to 1% of the overall energy consumption in 2007, risen to 7% in 2017. To combat this, programmable hardware acceleration, mainly GPUs and FPGAs, becomes more and more important, since this combines programmability with a good performance and energy efficiency similar to fixed-function ASICs.

The authors of [5] have shown that an execution of a network function on hardware accelerators (HWA) is possible and meaningful. Current approaches require user space communication between the network interface card (NIC) and the accelerator hardware, including various copy operations through host memory. For example, the system depicted in Figure 1 receives incoming network packets from the NIC and stores them in main memory. A first software thread initiates the transmission of the data to the GPU/FPGA where the processing of the hardware accelerated network function is performed. Finally, the packet is copied back to the main memory, before being copied to the NIC by a second CPU thead for sending. This example exhibits several wasteful copy operations to and from main memory. Furthermore, software threads without a functional contribution are required and the memory controller, which is a shared resource with the CPU, is utilized without any benefit.

With this work, we propose the idea of "host bypassing", a direct interconnect between NICs and hardware accelerators

Middleware Demos and Posters '19, December 9–13, 2019, Davis, CA, USA
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-7042-4/19/12.
https://doi.org/10.1145/3366627.3368112

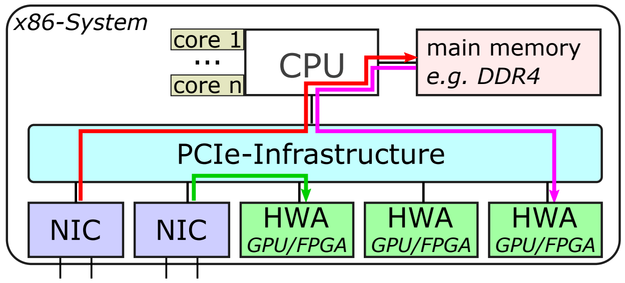

Figure 1. System concept of hardware accelerators in a x86-based host system. Red path + purple path illustrates the currently common data path and green depicts the proposed approach.

such as FPGAs and GPUs. A network function chain consisting of multiple hardware accelerated network functions would benefit strongly from this approach as the packets flow directly between the accelerator entities.

2 Related Work & State-of-the-Art

Current state-of-the-art hardware accelerators are integrated into server systems by the PCI Express (PCIe) bus system. Bittner *et al.* [2] have investigated PCIe peer-to-peer (P2P) communication and have shown the feasibility and a speedup from $60\mu s$ to $40\mu s$ compared to a host memory-based communication for a sample data transfer between GPU and FPGA in 2014. We expect current technologies to behave similarly in both cases. Remote Direct Memory Access (RDMA) enables kernel bypassing for high-performance data transmissions between the main memory of different computer systems via a network in order to minimize CPU overhead. NVIDIA *GPUDirect RDMA* enables the address-range mapping of other PCIe devices into the own CUDA application, forming the basis for P2P communication. In our previous work [4] we have shown the feasibility and benefits of direct communication between GPU and NVMe disc.

Intel I/O Acceleration Technology (I/OAT) offers multiple improvements regarding PCIe performance: 1) fast data copy without CPU interaction by dedicated chipset functionality, 2) direct processor cache access, allowing NICs and other devices to write data directly into the CPU cache without traversing the main memory, 3) TCP/IP stack acceleration by offloading TCP flow reassembling and 4) QoS-aware interrupt signaling for a higher efficiency.

The Data Plane Development Kit (DPDK) is a framework for building high performance VNFs with kernel bypassing. As it is an OpenSource project it can be easily extended and technical internals of NICs can be extracted.

3 Approach

PCIe P2P communication minimizes the number of memory copies, reduces the workload of the memory management unit, and improves the latency and throughput of a data path. For a given worklaod, energy consumption is reduced and CPU resources are freed. For that, DPDK programmable NICs can be used to forward packets directly to a hardware accelerator. The driver functionality of DPDK can be divided into two parts: (1) connection setup between NIC and accelerator and (2) sending and receiving packets. The initial connection setup has to be done only once and should be done by the x86 host as it is not performance-critical. Sending and receiving packets is managed directly by the hardware accelerator and the NIC. As a precondition for this, memory ring buffers for sending and receiving packets are implemented on the accelerator hardware, following the DPDK software implementation. Configuring the NIC to receive packet into the physical location of such a ring buffer on a hardware accelerator causes the packets to be transferred there directly, without passing through the main memory.

In contrast to existing work on inter-FPGA/GPU communication, this approach works with off-the-shelf DPDK-capable NICs. First tests with a low-priced server (Xeon-4110 CPU, Intel 82599 NIC) have shown promising results. PCIe P2P communication seems to be available on all tested architectures and the reconfiguration of the DPDK-NIC to use the ring buffers on an FPGA worked as well.

4 Future Work

The next steps are a fully-fledged implementation of a NIC to FPGA/GPU interconnect and DPDK integration. Following this, we plan to evaluate the performance characteristics and improvements in detail. Furthermore, we assume that P2P communication can benefit from dedicated PCIe switches on mainboards as no communication with the processor and main memory is required. Finally, this work can reduce the ratio of server CPU cores to hardware accelerators and by that costs and energy consumption can be reduced.

Acknowledgment

This work has been supported by the German Research Foundation (DFG) as part of the project C2 within the Collaborative Research Center (CRC) 1053 MAKI. We thank our colleagues and reviewers for their valuable feedback.

References

[1] J. Baliga, K. Hinton, and R. S. Tucker. 2007. Energy Consumption of the Internet. In *COIN-ACOFT 2007 - Joint International Conference on the Optical Internet and the 32nd Australian Conference on Optical Fibre Technology.* 1–3.

[2] R. Bittner, E. Ruf, and A. Forin. 2014. Direct GPU/FPGA communication Via PCI express. *Cluster Computing* 17, 2 (2014), 339–348.

[3] Y. Li and M. Chen. 2015. Software-Defined Network Function Virtualization: A Survey. *IEEE Access* 3 (2015), 2542–2553.

[4] J. Markussen, L.B. Kristiansen, H.K. Stensland, F. Seifert, C. Griwodz, and P. Halvorsen. 2018. Flexible Device Sharing in PCIe Clusters Using Device Lending. In *Proceedings of the 47th International Conference on Parallel Processing Companion (ICPP '18).* ACM, Article 48, 10 pages.

[5] H. Shojania, B. Li, and X. Wang. 2009. Nuclei: GPU-Accelerated Many-Core Network Coding. In *IEEE INFOCOM 2009.* 459–467.