A 10 GbE TCP/IP Hardware Stack as part of a Protocol Acceleration Platform

Dipl.-Ing. U. Langenbach†, Dipl.-Ing. A. Berthe†, Dipl.-Ing. B. Traskov*, B.Sc. S. Weide†,
Prof. Dr.-Ing. K. Hofmann*, Prof. Dr.-Ing. P. Gregorius†
*Technische Universität Darmstadt
†Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute

***Abstract*—With the increasing number of Internet services, the flexible and reliable TCP/IP protocol suite has become a standard for network communication. TCP/IP is the underlying protocol used for ordered, connection-oriented data transfers over the Internet. Also in the application field of embedded systems TCP/IP has become the protocol suite of choice, because of interoperability across the internet. The next generation of embedded communication solutions will address 10 Gbps and more due to the growing demand for low round-trip times and high throughput. This paper presents a fully hardwired 10 GbE TCP/IP stack, designed a tightly coupled system integrating higher-level protocols and application-specific logic in order to build a fully integrated and accelerated communication stack as part of an FPGA or ASIC design. The advantages of using a hardware-based 10 GbE TCP/IP stack in terms of latency and throughput are illustrated in this paper by comparing its performance with that of state-of-the-art network interface cards in conjunction with software-based TCP/IP stacks.**

I. Background

The data volume processed by high-performance digital systems are ever increasing since the beginning of networking. Today appropriate physical transmission technologies and networks are available, but the protocol stack that links the desired application to the actual transmission technology imposes a huge bottleneck [1]. Most systems employ an increasing number of CPU cores, while the single core performance stagnates [2]. For many systems the increasing message rate leads to performance issues in the operating systems network stack, as the interrupt rate cannot be increased and single datastream processing cannot be generally parallelized. Although state-of-the-art TOE-based (TCP/IP Offload Engine) solutions are providing a certain amount of processing relief compared to classical NICs (network interface card), a huge portion of data processing is required from the main processor [3]. Due to sequential software flow, protocol processing consumes CPU time and resources creating a dependency between processor load and available throughput as well as latency [4]. This reveals a major drawback especially for embedded systems where resources are even more limited and CPU time is needed for application-specific tasks.

To overcome these system-dependent limitations in throughput and latency a complete TCP/IP stack has been implemented in hardware, see figure 1. The hardware to software interface is moved to the application layer or beyond. This TCP/IP stack approach can run as a standalone solution and does not need any CPU performance at all. Even parts of the application or higher-level protocols can be realized in hardware as customized solution. The main target of the proposed implementation is an optimized bandwidth-delay product (maximum throughput by lowest latency) to provide highly responsive data processing systems without sacrificing bandwidth.

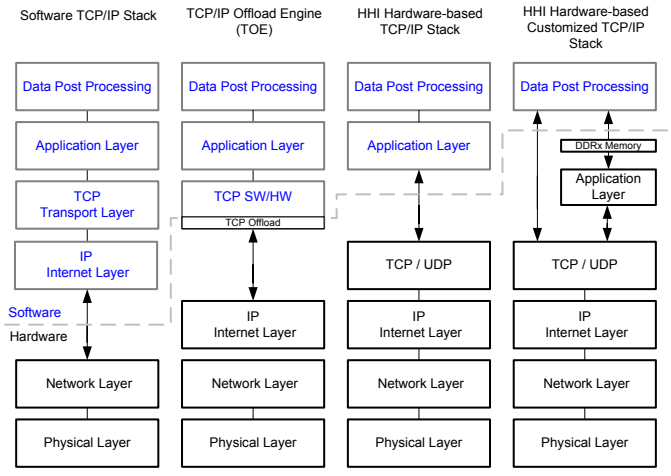

Fig. 1. Comparison of software and hardware-based TCP/IP stacks.

II. System Overview

The presented 10 GbE hardware-based TCP/IP stack can handle a single physical network interface and contains the IPv4, ICMPv4, UDP and TCP protocols as pictured in figure 2. The hardware-based TCP/IP stack features transparent handling of complete TCP/IP and UDP protocol tasks, e.g. packet encoding, packet decoding, acknowledge generation, link supervision, timeout detection, retransmissions and fault recovery. In addition all necessary sub protocols, i.e. ARP, IPv4 and ICMPv4 are implemented. The stack features complete connection control including tear up and tear down, transparent checksum generation and checksum checking.

III. Implementation Details

As shown in the high level block diagram in figure 2 the stack is implemented in subblocks each addressing a subfunction of a single protocol. Additionally these blocks are aggregated into wrappers for every single protocol. This

978-1-4799-1412-8/13/$31.00 ©2013 IEEE

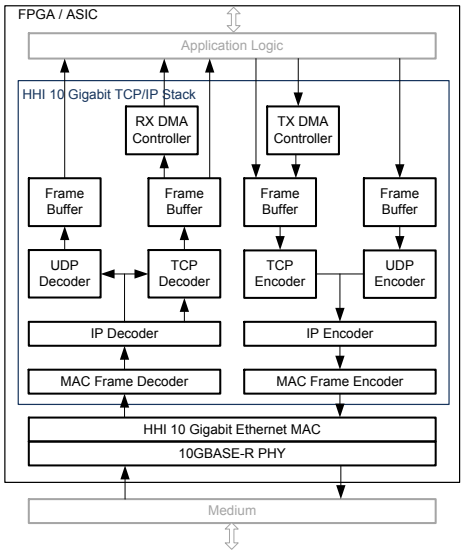

Fig. 2. System block diagram of the hardware-based TCP/IP stack.

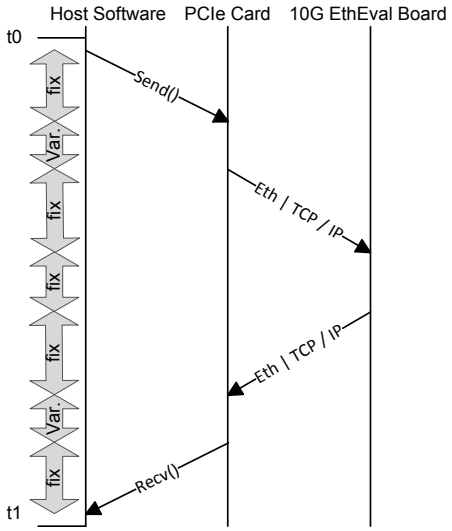

Fig. 3. Measurement procedure and latency breakdown

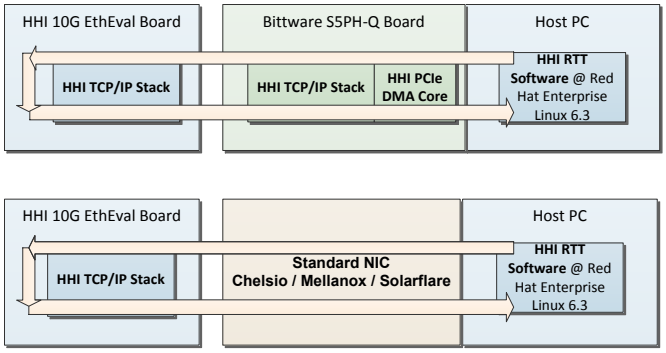

Fig. 4. Measurement Setup

enables straight forward configurability of the stack with respect to enable the usage of single protocols, e.g. optional IGMP. Additionally single protocol layers can be rearchitected, exchanged or added without massive impact on other layers. This comprises functional, logistical and physical implementation issues, e.g. timing closure, interactions between protocol layer implementations.

The implementation analysis follows the 'requirements for internet hosts', see RFC (request for comments) [5]. It is required that all unsupported protocol options are ignored on reception by IPv4 and TCP. All items, marked as must or should, are implemented in the stack.

The Fraunhofer HHI hardware TCP/IP stack provides an easy to use application interface for configuring a TCP session similar to software implementations. The application interface also provides functions for monitoring the session status. The implementation is tolerant to out-of-order segments and generates DUP ACKs to trigger fast retransmissions from the remote host.

Validation testing in the lab has been done against software TCP/IP stacks ranging from Linux kernels 2.6.18 to 2.6.32 and Windows 2000 - Windows 7, as well as the open onload stack ranging from version 20101111-u1 to 201210-u1. The Fraunhofer HHI 10 GbE hardware TCP/IP stack has been tested against itself in simulation and real world testing.

IV. Hardware Platforms

For validation, test and measurement of the 10 GbE TCP/IP stack multiple platforms facilitating a range of FPGA families from Altera and Xilinx are used. These platforms range from custom and in-house designed to COTS (commercial off-the-shelf) boards. Some platforms provide standardized interfaces such as PCIe cards, others are completely custom designed regarding form factor or interface, besides the network side. The stack is currently implemented on Xilinx Virtex 5, Virtex 6, Spartan 6 and Zynq platforms based on Artix 7 and Kintex 7 fabrics. Additionally Alteras Stratix IV and Stratix V, as well as Cyclone IV FPGAs are supported. The 10 GbE network interface can be connected to a third-party MAC or by the Fraunhofer HHI MAC. The network physical layer itself can be connected using the XAUI protocol layer for an external PHY or implemented directly inside the FPGA. The latter method implements a FPGA internal 10GBASE-R PCS/PMA using high speed transceivers. Legacy support is provided via an 1 GbE interface, which, connected to a 1 GbE MAC interfaces a PHY via SGMII, RGMII or GMII interfaces.

V. Measurement Results

A latency measurement setup enables a performance comparison of the Fraunhofer HHI 10 GbE hardware TCP/IP stack with state-of-the-art 10 GbE NICs. The standard NICs work in conjunction with software TCP/IP stacks. An FPGA design couples the Fraunhofer HHI 10 GbE hardware TCP/IP stack with a Fraunhofer HHI PCIe DMA Core. This approach preserves the application programming interface by providing a socket like API to the measurement software, similar to software stacks, while performing complete TCP/IP processing

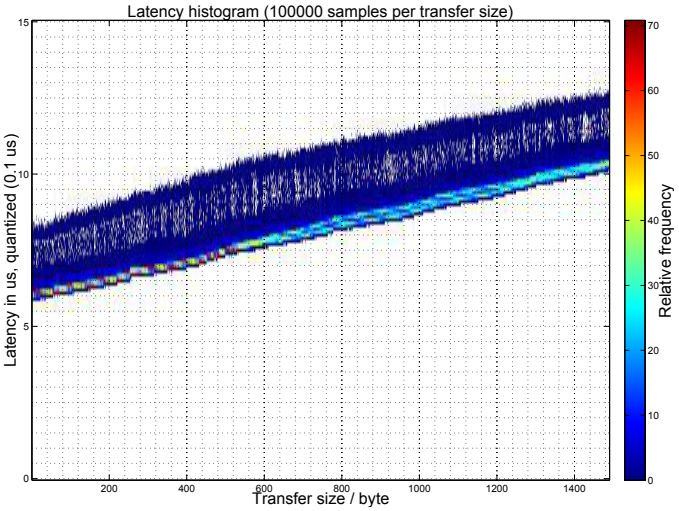

Fig. 5. Latency histogram - Solarflare NIC and OpenOnload TCP/IP stack (software TCP/IP stack, kernel-bypassing).

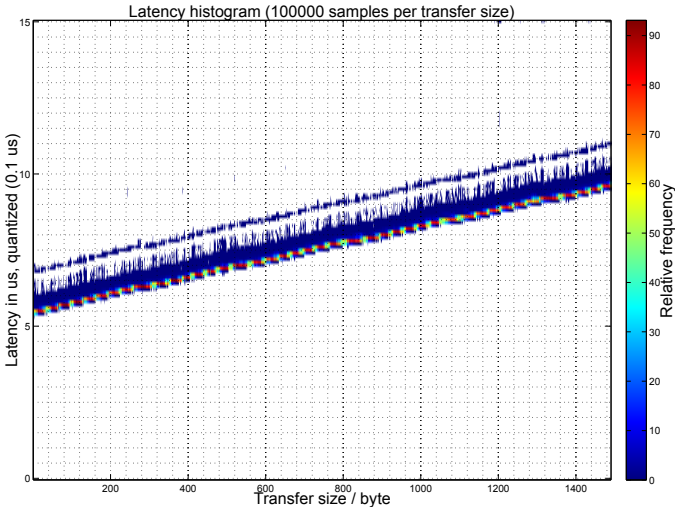

Fig. 6. Latency histogram - Fraunhofer HHI 10 GbE hardware TCP/IP stack (hardware TCP/IP stack, kernel-bypassing).

inside the chip.

The tests presented in this work use an Altera Stratix V based S5PH-Q Bittware PCIe board as a client and a custom Xilinx Virtex 5 based board (Fraunhofer HHI 10G EthEval Board) as a remote host. The system running the measurement software and containing the FPGA card and NICs is a standard Dell Precision T3600 workstation with an Intel XEON E5-1603 and 8 GB non-ECC memory. These cards are plugged into the graphics card slot. The measurement setup is shown in figure 4.

The test procedure is a simple ping pong test, focussing on latency measurement. A ping pong test always has only one packet in flight. This procedure was chosen to demonstrate the absolute maximum system performance in terms of latency. It does not load the host nor the network with additonal tasks or application level data processing. The ping pong test procedure is also consistently applicable to larger scale setups containing switches or routers, because it generally measures minimum latency values.

Figure 3 shows the breakdown of the overall latency into its fractions correlated to single steps of the measurement procedure. Because of the minimum load this setup puts onto the host, the latencies for the system calls can be assumed to be fixed. The latency of the completely passive network of this setup is also fix. The latency values include the software part of the system. Because the Fraunhofer HHI 10G EthEval Board implements a hardware TCP echo server, the latency is deterministic, even for full line rate TCP streams. The performance does not degrade with an increasing message rate.

To minimize the impact of the operating system several techniques such as CPU shielding have been used for all measurements. An extract of the measurement results is presented in form of latency histograms in figures 6 to 7. Each plot contains a total of 146 M single measurements, 100 k measurements per TCP payload size (column), which is sweeped from 1 to 1460 Byte. Every measurement is put into

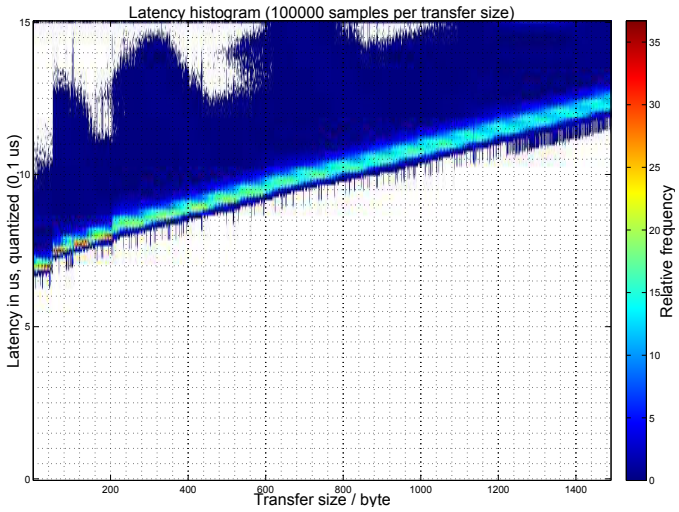

Fig. 7. Latency histogram - Mellanox NIC and standard Linux TCP/IP stack (software TCP/IP stack, no kernel-bypassing).

buckets of latencies with the size of 0.1 μs each, ranging from 0 μs to 15 μs absolute measured latency. All measurements above 15 μs are put into the 15 μs bucket. The color of each bucket encodes the percentage of measurements landed in the respective bucket with the same transfer size. This can also be interpreted as the probability for a latency of a specific transfer. Figure 5 shows the latency histogram when using a Solarflare SFN5122F NIC which offers several offloading features and the OpenOnload software stack. Even though the OpenOnload user space software stack uses kernel-bypassing the latency values are much more spread out than in figure 6. Figure 6 shows the latencies of the Fraunhofer hardware-based 10 GbE TCP/IP stack. The latency distribution shows a very small deviation with most of the values being very close to the minimum latency.A measurement in which the standard Linux TCP/IP software stack is used in combination with a Mellanox MCX354A-FCBT NIC is shown figure 7.

Comparing the histograms with respect to baseline latency the Fraunhofer HHI solution shows with 5.5 μs the lowest one. This is about 0.5 μs below the Solarflare and about 1.25 μs below the Mellanox NIC. The measurement of the Solarflare NIC with the onload stack shows a lower latency variance than the Mellanox NIC with the Linux kernel stack. The measurement of the Fraunhofer HHI hardware solution presents an even smaller latency spectrum.

The solarflare solution performs better with respect to latency than a solution using the standard Linux kernel TCP/IP stack, because of employing kernel bypass and a user space TCP/IP stack. The Fraunhofer HHI 10 GbE hardware TCP/IP stack perfoms even better, because it uses less CPU and OS bound ressources by offloading all protocol processing tasks to hardware and uses kernel bypass for exchanging data between hardware and measurement application.

VI. Application Scenarios

The high-performance networking provided by the hard-wired TCP/IP stack enables the reuse of the freed resources of the CPU for the application itself. This enables a more efficient use of the CPU cores and memory, which leads to a reduced number of machines needed to process a specific workload. The deterministic low-latency performance can be best applied in closed, short range network scenarios inside data centers, e.g. financial trading networks or cloud computing sites, due to the lower impact of other network equipment on latency and its distribution.

Load on the CPU can be additionally lowered by offloading parts of the application into the hardware using the proposed system architecture shown in figure 8. By offloading additional protocol processing units to the datapath in between the Fraunhofer HHI PCIe and TCP/IP cores, the overall application efficiency rises by using less cpu time. The measurements as described in the previous section uses the system as shown in figure 8, but without the custom protocol processing unit. As a result the latency shown for the presented protocol acceleration platform reflects the minimum latency for a protocol processing accelerator derived from this platform.

Using a partial reconfiguration flow for the FPGA based protocol acceleration platform enables real-time scheduling flexibility for custom protocols. This allows the overall system to flexibly adapt to changing application needs. The accelerated protocols can be switched by reconfiguring portions of the FPGA, while others, e.g. TCP/IP and PCIe, remain running and responsive and the hardware TCP/IP stack, e.g. still replies to echo requests (ping). The reconfiguration of the FPGA can be done on request of the application via PCIe using special driver functions and dedicated logic of the FPGA. For embedded systems it has been shown that this approach can be efficiently used to offload higher level protocol tasks off a CPU [6].

VII. Outlook

Measurements including networking equipment, e.g. switches and routers, will expand the view about the behavior of the presented solution as part of a network. These measurements additionally comprise bandwidth and latency under load test setups to improve the visibility of real world application effects.

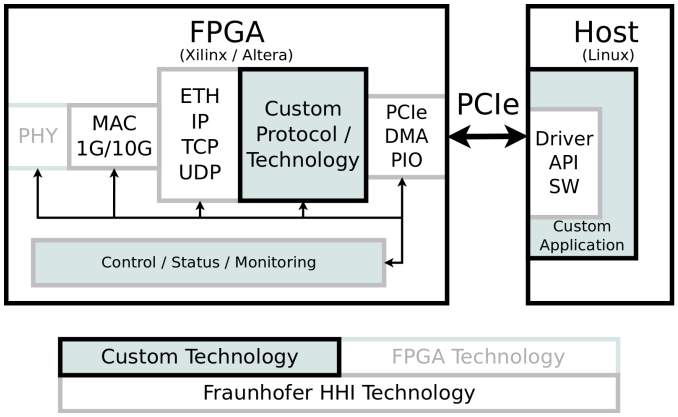

Fig. 8. Generic protocol acceleration platform

VIII. Conclusion

This work presents the new Fraunhofer HHI 10 GbE TCP/IP hardware stack showing lowest latencies with about 5.5 μs. The latency distribution histograms further on show the most deterministic behavior of this network stack. Based on this flexible platform high-performance accelerators of higher level protocols can be build. Such an accelerator is especially well suited for low latency applications in short range or tightly coupled networks.

Future embedded devices can benefit from adapting this presented FPGA based technology to an ASIC (application specific integrated circuit). This transition leads to better performance while increasing power efficiency. Creating an ASIC stand-alone TCP/IP stack enables high performance networking, low power SoCs (system on chip) for embedded devices.

References

[1] E. P. Markatos, "Speeding up tcp/ip: Faster processors are not enough," in *In 21st IEEE International Performance, Computing, and Communication Conference*. IEEE, 2002, pp. 341–345.

[2] P. E. McKenney, *Is Parallel Programming Hard, And, If So, What Can You Do About It?* Corvallis, OR, USA: kernel.org, 2011, available: http://kernel.org/pub/linux/kernel/people/paulmck/perfbook/perfbook.html.

[3] Z.-Z. Wu and H.-C. Chen, "Design and implementation of tcp/ip offload engine system over gigabit ethernet," in *Computer Communications and Networks, 2006. ICCCN 2006. Proceedings.15th International Conference on*, 2006, pp. 245–250.

[4] G. Regnier, D. Minturn, G. McAlpine, V. Saletore, and A. Foong, "Eta: experience with an intel reg; xeon trade; processor as a packet processing engine," in *High Performance Interconnects, 2003. Proceedings. 11th Symposium on*, 2003, pp. 76–82.

[5] R. Braden, "Requirements for Internet Hosts - Communication Layers," RFC 1122 (INTERNET STANDARD), Internet Engineering Task Force, Oct. 1989, updated by RFCs 1349, 4379, 5884, 6093, 6298, 6633, 6864. [Online]. Available: http://www.ietf.org/rfc/rfc1122.txt

[6] A. Salman, M. Rogawski, and J. Kaps, "Efficient hardware accelerator for ipsec based on partial reconfiguration on xilinx fpgas," in *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*, 2011, pp. 242–248.