Scalable 10 Gbps TCP/IP Stack Architecture for Reconfigurable Hardware

David Sidler, Gustavo Alonso
Systems Group
Dept. of Computer Science
ETH Zürich
{dasidler, alonso}@inf.ethz.ch

Michaela Blott, Kimon Karras,Kees Vissers
Xilinx Research
Dublin, Ireland
& San Jose, CA
{mblott, kimonk, kees.vissers}@xilinx.com

Raymond Carley
Dept. of Electrical
& Computer Engineering
Carnegie Mellon University
rcarley@ece.cmu.edu

***Abstract*—TCP/IP is the predominant communication protocol in modern networks but also one of the most demanding. Foong [1] stipulates that 1 Hertz (Hz) of CPU processing is required to send or receive 1 bps of TCP/IP. Consequently, TCP/IP offload is becoming increasingly popular with standard network interface cards with many of the major vendors offering full offload since 2011 [2]. TCP/IP Offload Engines (TOEs) have also emerged for FPGAs offered by vendors such as Intilop, Fraunhofer HHI, PLDA and Dini Group [3], [4], [5], [6]. With the driving application being high-frequency trading (HFT), these implementations are focused on low latency and are typically constrained in their session count support. However, many more applications beyond HFT can potentially be accelerated inside an FPGA once TCP with high session count is terminated inside the fabric. This way, a network-attached FPGA on ingress and egress to a CPU, can accelerate functions such as encryption, compression, memcached and many others in addition to the complete network stack.**
This paper introduces a novel architecture for a 10 Gbps line-rate TCP/IP stack for FPGAs that can scale with the number of sessions and thereby address these new applications. We prototyped the design on a VC709 development board, demonstrating compatibility with existing network infrastructure, operating at full 10 Gbps throughput full-duplex while supporting 10'000 sessions. Finally, the design has been described primarily in C++ using High-level Synthesis (HLS), which provides greater flexibility and maintainability. This flexibility is highly advantageous to efficiently adapt the network stack to support additional network features and increased customization.

I. Introduction

TCP/IP is the cornerstone of modern network communications with its support for reliable data transfer including flow control, congestion avoidance, duplicate data suppression and in-order delivery. However, this is associated with substantial complexity. Foong [1] stipulates that 1 Hertz of CPU processing is required to send or receive 1 bps of TCP/IP which equates to 8 cores clocked at 1.25 GHz for 10 Gbps line-rate processing. The reasons for this are manyfold and well understood. First of all, as TCP is connection-oriented, the implemented network stack needs to keep state for every connection, something which naturally becomes more complex with the number of open sessions. Secondly, to ensure reliable data transfer, data has to be buffered until an acknowledgment has been received. Additionally, segmentation and reassembly are needed together with out-of-order processing to packetize incoming and outgoing data streams from the application layer. Finally, TCP is interrupt-intensive in nature, as for example every time a packet is received or a transmission timeout occurs, an interrupt is triggered. Together with its large footprint, which exceeds the capacity of standard instruction caches and therefore causes a high miss rate, this results in poor branch-predictability on standard x86 execution and disrupts co-executing applications [7].

As a result, TCP/IP offload is increasingly integrated with standard network interface cards with many of the major vendors, such as Broadcom, Emulex, and Chelsio offering full offload since 2011 [2]. TCP/IP Offload Engines (TOEs) have also emerged for FPGAs offered by vendors such as Intilop, Fraunhofer HHI, PLDA and Dini Group [3], [4], [5], [6]. However, these implementations target high-frequency trading (HFT) which is driven by latency requirements [8]. To minimize latency, these stacks constraint session support as we explain further in section II.

Our design aims to expand the applicability of FPGAs by creating a flexible architecture that, in addition to delivering full line-rate throughput, can also scale to high session counts, an essential prerequisite to deployment in networked servers in data centers. Beyond alleviating the TCP/IP bottleneck on the host CPU, the solution aims to accelerate applications such as encryption, compression, memcached [9], higher-level protocol processing such as JMS [10] by pushing the TCP termination completely inside the FPGA. This enables acceleration through reconfigurable logic for potentially thousands of sessions.

To maximize the stack's applicability, it was essential to create a flexible solution that allows to efficiently and easily adapt the design to different congestion avoidance schemes, potentially out-of-order processing, etc., while using a minimal resource footprint. To achieve this, we adppted a C++-based design flow using High-Level Synthesis (HLS) that simplifies the design, makes it more flexible and easier to customize towards specific requirements. As an added bonus, the HLS-based approach makes the design automatically portable to any device supported by the tool and significantly increases productivity.

In more detail, the key contributions of this paper are as follows:

- Sustained 10 Gbps bandwidth TCP/IP stack through dataflow architecture
- Support for 10'000 concurrent connections with external data buffers and hash-based session lookup and linear scalability
- Design flexibility through C/C++ design using Vivado HLS
- Control flow features and out-of-order segment processing