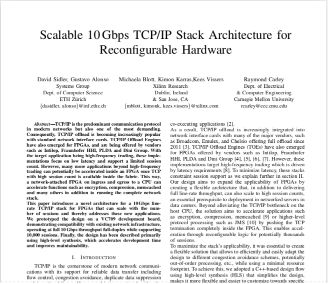
ETH zürich

Low-Latency TCP/IP Stack for Data Center Applications

David Sidler, Zsolt István, Gustavo Alonso
Systems Group, Dept. of Computer Science, ETH Zürich

Original Architecture [1]



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. Consequently, TCP/IP offload is becoming increasingly popular with standard network interface cards. TCP/IP Offload Engines have also emerged for FPGAs, and are being offered by vendors such as Intilop, Fraunhofer HHI, PLDA and Dini Group. With the target application being high-frequency trading, these implementations focus on low latency and support a limited session count. However, many more applications beyond high-frequency trading can potentially be accelerated inside an FPGA once TCP with high session count is available 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 running 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 addresses 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 using high-level synthesis, which accelerates development time and improves maintainability.

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

co-executing applications [2].

As a result, TCP/IP offload is increasingly integrated into network interface cards with many of the major vendors, such as Broadcom, Emulex, and Chelsio offering full offload since 2011 [3]. TCP/IP Offload Engines (TOEs) have also emerged for FPGAs offered by vendors such as Intilop, Fraunhofer HHI, PLDA and Dini Group [4], [5], [6], [7]. However, these implementations target high-frequency trading which is driven by latency requirements [8]. To minimize latency, these stacks constraint session support as we explain further in section II.

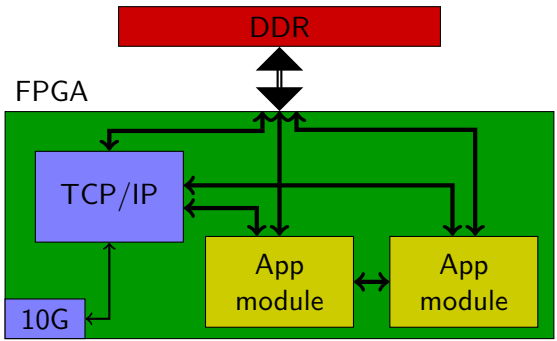
Our design aims to expand the applicability of FPGAs by creating a flexible architecture that, in addition to delivering full line-rate throughput, can also scale to high session counts, an essential prerequisite to deployment in networked servers in data centers. Beyond alleviating the TCP/IP bottleneck on the host CPU, the solution aims to accelerate applications such as encryption, compression, memcached [9] or 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 adopted a C++-based design flow using high-level synthesis (HLS) that simplifies the design, makes it more flexible and easier to customize towards specific

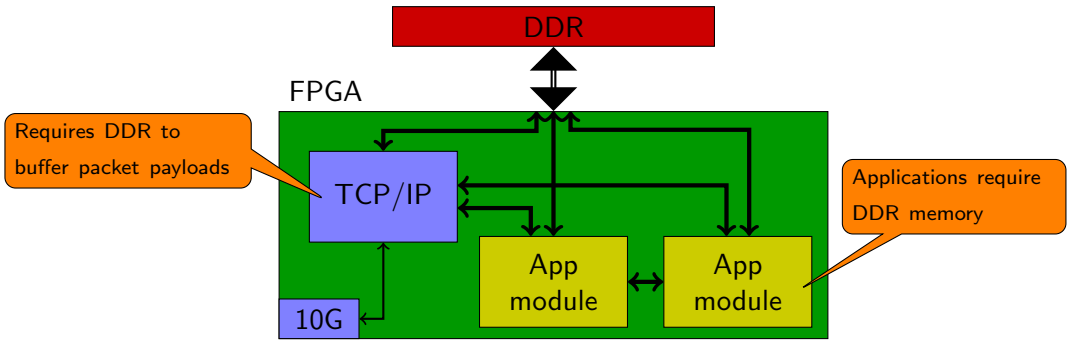
- 10 Gbps bandwidth TCP/IP stack
- Supporting thousands of concurrent connections
- Generic implementation as close to specification as possible
- Enables seamless integration of FPGA-based applications into existing networks

[1] Sidler et al., *Scalable 10 Gbps TCP/IP Stack Architecture for Reconfigurable Hardware*, FCCM'15, `http://github.com/dsidler/fpga-network-stack`

Application Integration

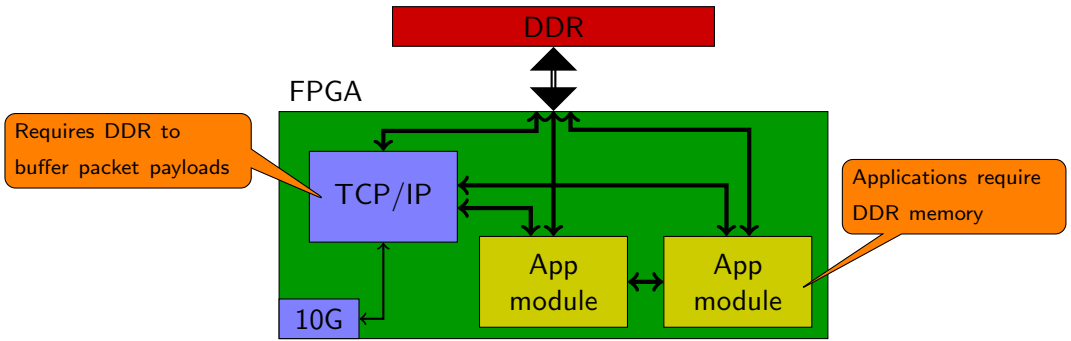


Application Integration



- Memory bandwidth is shared among multiple modules → potential bottleneck

Application Integration



- Memory bandwidth is shared among multiple modules → potential bottleneck
- Distributed systems rely on very low latency → to guarantee latency bounds to clients

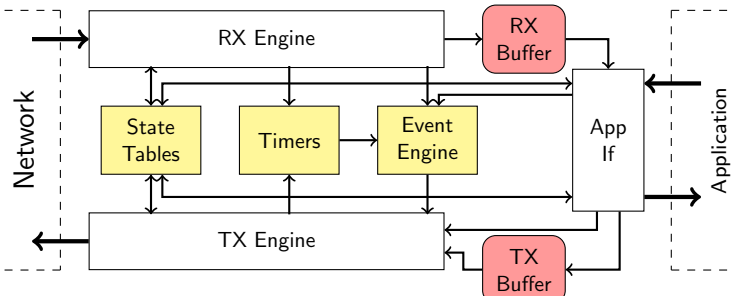
Assumptions

- Application
  - Client requests fit into an MTU (maximum transfer unit)
  - Synchronous clients
  - Application logic consumes data at line-rate

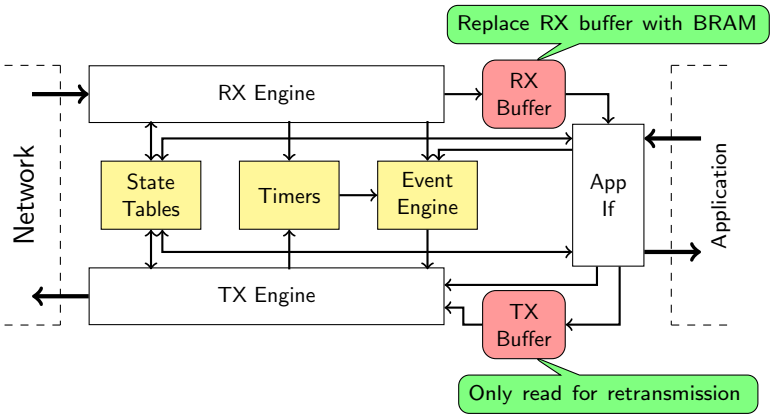
Assumptions

- Application
  - Client requests fit into an MTU (maximum transfer unit)
  - Synchronous clients
  - Application logic consumes data at line-rate
- Data center network
  - High reliability and structured topology
  - Data loss less common $\rightarrow$ fewer retransmission
  - Packets are rarely reordered

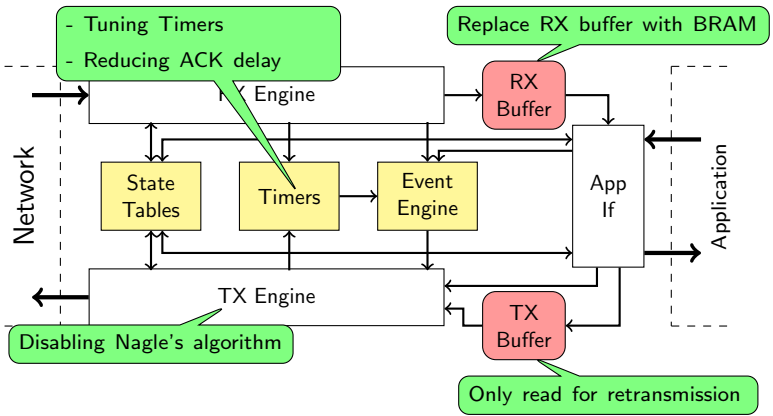
Optimizations for Data Center Applications



Optimizations for Data Center Applications

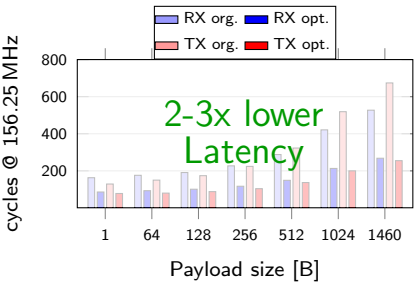


Optimizations for Data Center Applications

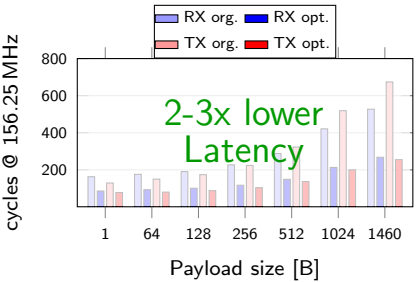


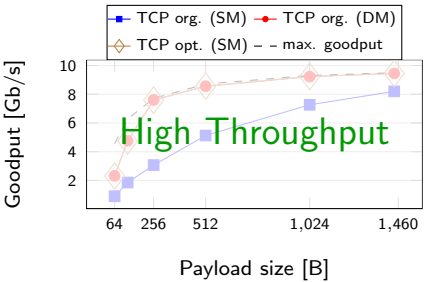
Results

Results

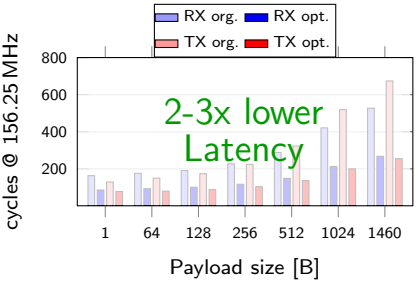

RX org.
RX opt.
TX org.
TX opt.
cycles @ 156.25 MHz
800
600
400
200
2-3x lower
Latency
1
64
128
256
512
1024
1460
Payload size [B]

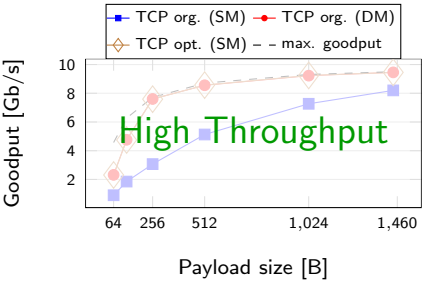
Results


RX org.
RX opt.
TX org.
TX opt.
cycles @ 156.25 MHz
800
600
400
200
2-3x lower
Latency
1
64
128
256
512
1024
1460
Payload size [B]


TCP org. (SM)
TCP org. (DM)
TCP opt. (SM)
max. goodput
Goodput [Gb/s]
10
8
6
4
2
High Throughput
64
256
512
1,024
1,460
Payload size [B]

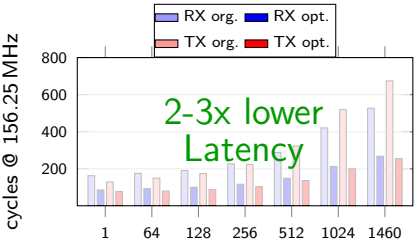
Results

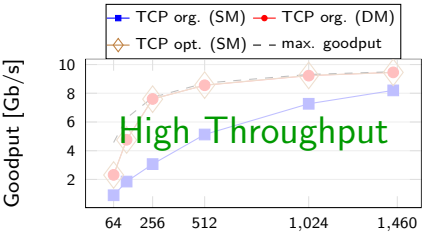




	Mem. allocated	Mem. bandwidth
TCP org.	1,300 MB	40 Gbps
TCP opt.	650 MB	10 Gbps
Diff	**-50%**	**-75%**

Results



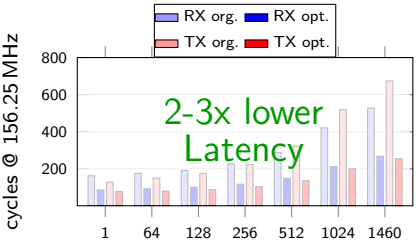


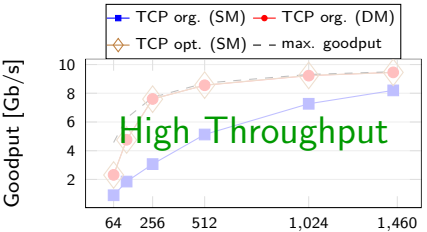
These results enabled a consistent distributed key-value store [2]

	Mem. allocated	Mem. bandwidth
TCP org.	1,300 MB	40 Gbps
TCP opt.	650 MB	10 Gbps
Diff	**-50%**	**-75%**

[2] István et al., *Consensus in a Box: Inexpensive Coordination in Hardware*, NSDI16

Results





Visit our poster for more results and details!
Find the source at: `http://github.com/dsidler/fpga-network-stack`

	Mem. allocated	Mem. bandwidth
TCP org.	1,300 MB	40 Gbps
TCP opt.	650 MB	10 Gbps
Diff	**-50%**	**-75%**

[2] István et al., *Consensus in a Box: Inexpensive Coordination in Hardware*, NSDI16