8. October 2017

100 GBIT/S END-TO-END COMMUNICATION:
FLEXIBLE PROTOCOL PROCESSING ON MANYCORE NICS

Steffen Büchner, Jörg Nolte, Lukasz Lopacinski, Rolf Kraemer

Project End2End100
Desired throughput 100 Gbit/s and data-packet size of 1500 Byte:

⇒ We have to handle one data-packet every 120 ns.
⇒ The main memory access latency for a 64 Byte cache-line is $\sim 96.6$ ns$^1$.
⇒ The protocol processing has to be parallelized and offloaded into a manycore NIC!

---

To be able to utilize a manycore processor for the protocol processing ...

- ... we need easy to parallelize protocols.
- ... we need a way to manage the parallel protocol processing.
- ... we have to adapt the protocol processing to the available processing power.
A SOFT REAL-TIME STREAM PROCESSING
DESIGN PROCESS

One design process . . .

• . . . without paradigm changes,
• . . . that allows us to adapt the protocol for different scenarios,
• . . . scale the protocol for different data-rates,
• . . . and map it on the manycore NIC.
DESIGN
AN EASY TO PARALLELIZE PROTOCOL STRUCTURE

Frame Structure:

Preamble | Header |
-----------|--------|

Sub-Packets

nr | payload | FEC/CRC |
---|---------|---------|

Acknowledgement:

Preamble | Header |
-----------|--------|

Aggregated acknowledgement coded as a bitmap of a Data Chunk

We have to combine . . .

• . . . large frames with small packets,
• . . . to minimize the protocol overhead as well as possible packet loss.
A communication protocol is represented as a . . .

- . . . Protocol Processing Engine, which is composed of
- . . . Protocol Processing Stages, that fulfill the individual processing tasks.
• Each Protocol Processing Stage has soft real-time requirements that state how often this Protocol Processing Stage has to be executed.
• The hardware capacities state how often a Protocol Processing Stage can be executed on a single CPU.
Stream-Operators are used to . . .

- . . . adapt the soft real-time requirements to the hardware’s capacities
- . . . and to alter the protocol’s behavior!
Several Data-Chunks are transmitted in parallel:

- Robust against packet loss
- Protocol overhead and retransmissions hidden by parallel pipelines
- Throughput expected to increase accordingly to the number of channels
- Latency is expected to stay constant
ADAPTATION
PROTOCOL PROCESSING ENGINE FOR STABLE THROUGHPUT

Requirements:
- Data-Chunk: 73.33/s
- Trigger: 146.66/s
- Sub-Packet: 505977.00/s
- Frame: 126494.26/s

Capacities:
- Data-Chunk: 1207.43/s
- Trigger: 55*10^6/s
- Sub-Packet: 627352.57/s
- Frame: 126518.00/s

Many Core FPGA

10 GbE Interface
CRC/FEC

Host

Data-Chunk

Sub-Packet Generator

Stream Split

Sub-Packet

Ack Processor

Frame

Sub-Packet Aggregator

Sub-Packet

Frame

Sub-Packet

Stream Join

Sub-Packet Aggregator

Sub-Packet

Frame

Sub-Packet

Stream Join

Sub-Packet Aggregator

Sub-Packet

Frame

Sub-Packet

Stream Split

Sub-Packet

Ack Processor

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet Ack.

Trigger

Ack.

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame

Sub-Packet

Sub-Packet

Frame
One Data-Chunk is transmitted in parallel:

- Throughput depends on low packet loss
- Protocol management overhead is not hidden
- Latency is expected to decrease accordingly to the number of interfaces
ADAPTATION
PROTOCOL PROCESSING ENGINE FOR LOW LATENCY

Requirements:
Capacities:

Host

PHY-Layer

Data-
Chunk

Sub-
Packet

Generator

Stream

Split

Trigger

Ack.

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Aggregator

Sub-
Packet

Ack.

1173.24/s

55*10^6/s

586.62/s

8286/s

586.62/s

579710.14/s

586.62/s

579710.14/s

586.62/s

8286/s

1207.43/s

627352.57/s

102796.06/s

102796.06/s

102796.06/s

102796.06/s
Evaluation System

- 2x TILEncore Gx72 Manycore – high level protocol processing
- Xilinx Virtex-7 FPGA – offloading of compute intensive protocol parts
- Connected by 10 GbE Interfaces
EVALUATION - THROUGHPUT
NO FEC

Stable Throughput

- 78.315 Gbit/s out of theoretical max. 80 Gbit/s
- Processing/Management Overhead:
  - min. 0.495% - max. 0.927%
  - 0.050 Gbit/s (0.50%) 1x 10GbE and
  - 0.444 Gbit/s (0.55%) 8x 10GbE

Low Latency

- 76.73 Gbit/s out of theoretical max. 80 Gbit/s
- Processing/Management Overhead:
  - min. 0.56% - max. 0.92%
  - 0.056 Gbit/s (0.56%) 1x 10GbE and
  - 2.054 Gbit/s (2.57%) 8x 10GbE

Offloaded FEC

- 2.45% lower throughput due to offloading
EVALUATION - LATENCY PER 16MB DATA-CHUNK
NO FEC

Stable Throughput

- Almost constant latency

Low Latency

- Latency decreases according to the number of interfaces
- Latency:
  - 13.714 ms (1x GbE)
  - 1.748 ms (8x GbE)
⇒ ~34 µs off the optimal scaling

Offloaded FEC

- 2.52% higher latency due to offloading
Modeling communication protocols as a soft real-time stream processing problem helps us to . . .

- . . . design highly scalable protocols,
- . . . analyze a protocol’s soft real-time requirements,
- . . . adapt a protocol to fit the hardware’s capacities,
- . . . and implement the protocol without paradigm changes.
Thank you for your attention

The project End2End100 is a joint project of IHP/BTU and is part to the DFG (German Research Foundation) priority program "100 Gbit/s Wireless And Beyond". ²

²DFG Schwerpunktprogram SPP 1655 Drahtlose Ultrahochgeschwindigkeitskommunikation für den mobilen Internetzugriff