SAP SE

sap.com

#sap

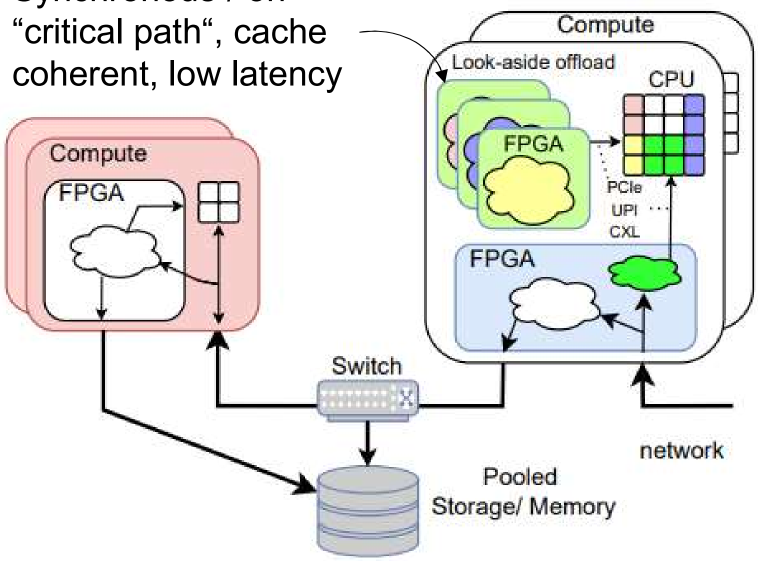
DASH : Asynchronous Hardware Data Processing Services

Norman May, Daniel Ritter, Andre Dossinger, Christian Färber, Suleyman Demirsoy

SAP SE

FPGA Compute
Topologies in the Cloud

Synchronous / on “critical path“, cache coherent, low latency



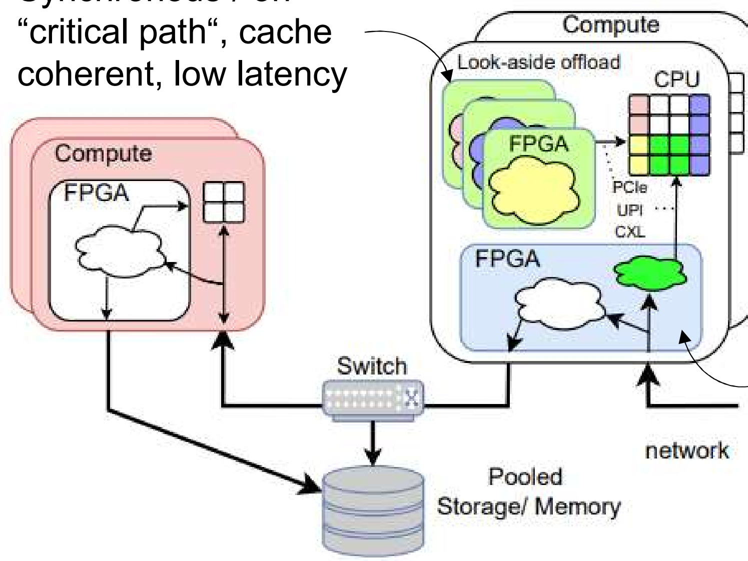
Motivation:

I. A lot of good work on query processing (green) → FPGAs more cost- and energy efficient, but no breakthrough for FPGA usage on “critical path“ in commercial databases (latency-centric)

- Fang, Jian, et al. "In-memory database acceleration on FPGAs: a survey." *The VLDB Journal* 29.1 (2020): 33-59.
- AWS AQUA: https://aws.amazon.com/blogs/aws/new-aqua-advanced-query-accelerator-for-amazon-redshift/, Nitro: https://aws.amazon.com/de/ec2/nitro/

FPGA Compute
Topologies in the Cloud

Synchronous / on "critical path", cache coherent, low latency



"Bump-in-the-wire"

Motivation:

I. A lot of good work on query processing (green) → FPGAs more cost- and energy efficient, but no breakthrough for FPGA usage on "critical path" in commercial databases (latency-centric)

II. FPGAs promising for "Bump-in-the-wire" (blue); commercial products, e.g., AWS Aqua + Nitro (throughput-centric)

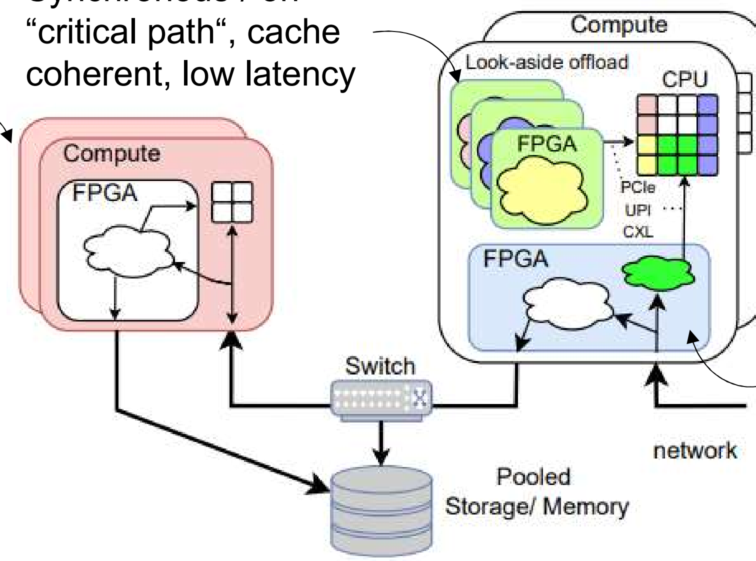
- Fang, Jian, et al. "In-memory database acceleration on FPGAs: a survey." *The VLDB Journal* 29.1 (2020): 33-59.
- AWS AQUA: https://aws.amazon.com/blogs/aws/new-aqua-advanced-query-accelerator-for-amazon-redshift/, Nitro: https://aws.amazon.com/de/ec2/nitro/

FPGA Compute

Topologies in the Cloud

Asyncronous, non-cache coherent, compute-intensive, throughput-optimized

Synchronous / on “critical path“, cache coherent, low latency



“Bump-in-the-wire“

Motivation:

I. A lot of good work on query processing (green) → FPGAs more cost- and energy efficient, but no breakthrough for FPGA usage on “critical path“ in commercial databases (latency-centric)

II. FPGAs promising for “Bump-in-the-wire“ (blue); commercial products, e.g., AWS Aqua + Nitro (throughput-centric)

III. Which options do we have for non-cache coherent, compute-intensive and throughput-centric workloads? Asynchronous compute acceleration (red)

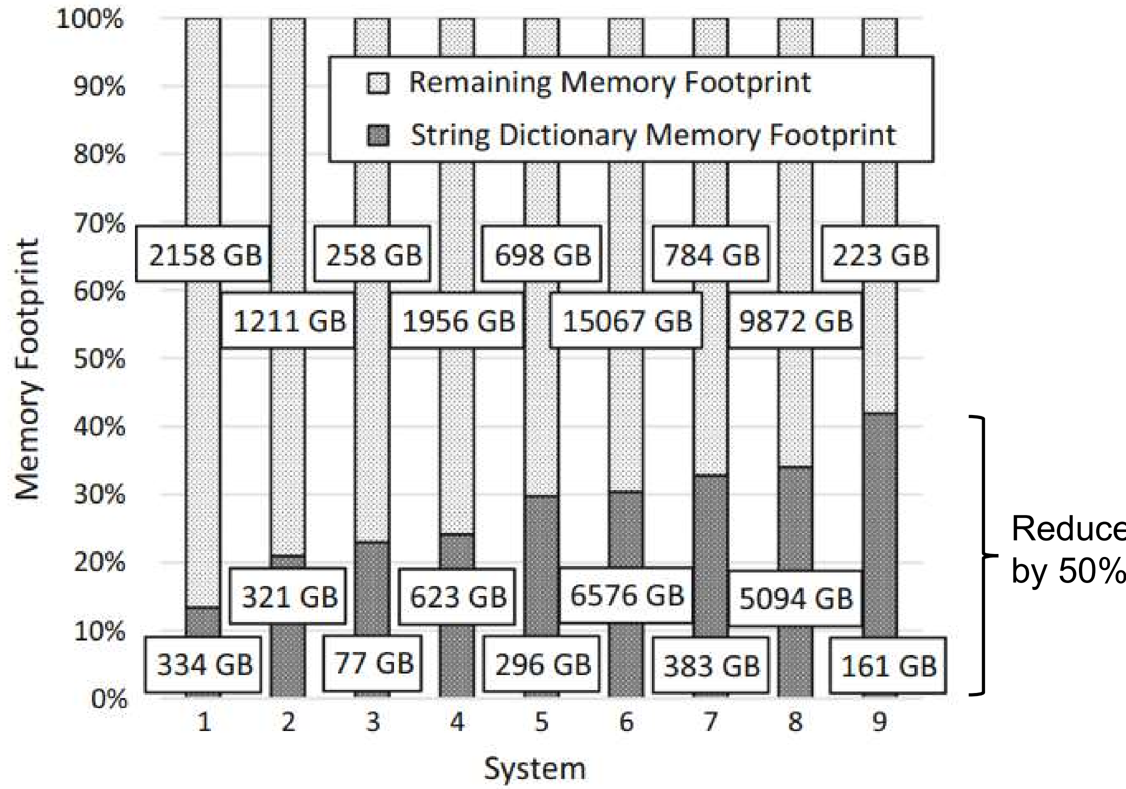
- Fang, Jian, et al. "In-memory database acceleration on FPGAs: a survey." *The VLDB Journal* 29.1 (2020): 33-59.
- AWS AQUA: https://aws.amazon.com/blogs/aws/new-aqua-advanced-query-accelerator-for-amazon-redshift/, Nitro: https://aws.amazon.com/de/ec2/nitro/

SAP SE

Use Case Example: String Dictionary Compression

High-density memory database instance:

- → Real-world ERP systems use >15% of memory for string dictionaries
- → Reduce memory consumption by 50% (Re-Pair)
- → Allows for more data to be loaded or less costs due to smaller instance
- → However, strong compression too slow for putting it on "critical path", stronger architecture coupling
- → FPGAs better throughput; Lower cost, energy consumption; FPGA shared by several instances



- Lasch, Robert, et al. "Faster & strong: string dictionary compression using sampling and fast vectorized decompression." The VLDB Journal 29.6 (2020): 1263-1285.
- Lasch, Robert, et al. "Accelerating re-pair compression using FPGAs." *Proceedings of the 16th International Workshop on Data Management on New Hardware*. 2020.

Why use FPGAs for Compute-intensive and Throughput-centric Workloads?

Benefits:

→ Competitive performance through data flow / pipelining for certain use cases

→ Efficient

  → compute with instructions tailored to the specific case

  → adaptable memory access

→ Cost and energy efficient (compared to CPU, GPU)

→ FPGAs still more improvement potential compared to CPUs (e.g., Moore's law)

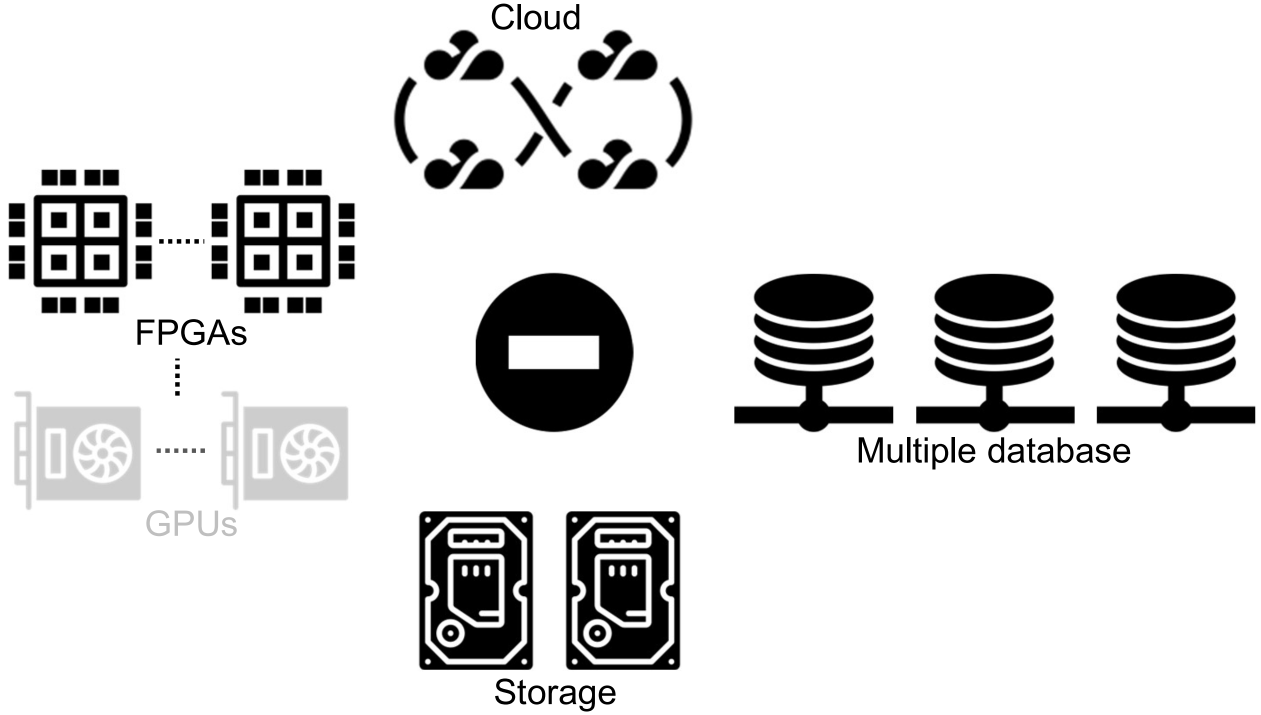
• Dann, Jonas, Daniel Ritter, and Holger Fröning. "Non-Relational Databases on FPGAs: Survey, Design Decisions, Challenges." ACM Computing Surveys (2020). [FPGA compute-intensive, throughput-centric examples on NRDS]

SAP SE

SAP SE

Disaggregated, Heterogeneous Accelerator-as-a-Service

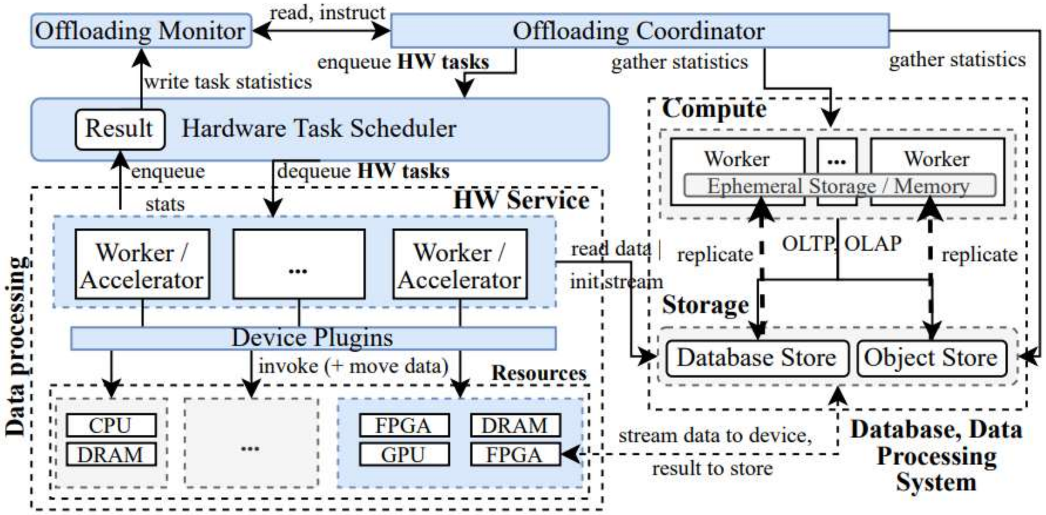
- → "FPGA-as-a-Service" for compute-intensive and throughput-centric, asynchronous offload, acceleration
- → Leverage cloud computing and next generation re-configurable HW
- → Loose architecture coupling
- → Not limited to FPGAs (e.g., GPUs, TPUs)



sap.com #sap

SAP SE

Asynchronous Hardware Data Processing Services



Core Building Blocks:

- Asynchronous data processing
  - `Offloading Coordinator` (cf. ADM), singleton > e.g., reduce consumed memory, storage; enqueues actions as `HW Tasks`
- Scheduling and observing HW services
  - `Hardware Task Scheduler` > flow control, priorization
  - `Offloading Monitor` > feedback loop
- Disaggreated, Elastic Compute
  - `HW Services` with attached resources (e.g., FPGAs) via `Device Plugins`
  - Each `HW Service` with several `Worker` / `Accelerator` components
  - `Workers` match their capabilities to `HW task specifications` > fitting worker-resource pair

sap.com

#sap

SAP SE

by example of Compression-as-a-Service (CaaS)

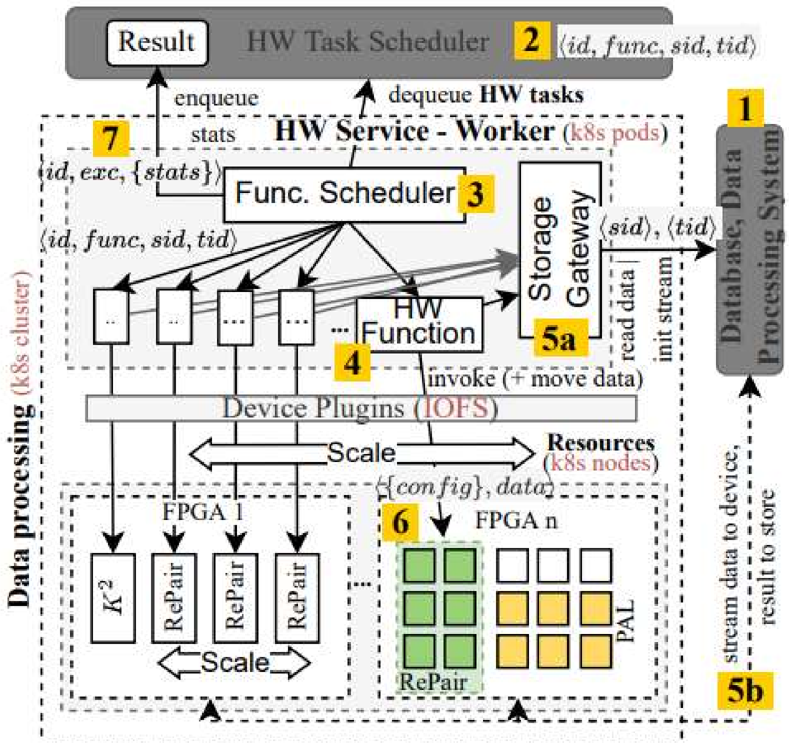
~ /kaːs > “cheese“ in Dutch

sap.com

#sap

SAP SE

Prototype



Instantiation of concept:

- using Re-Pair to compress string dictionaries in HANA Cloud; HANA's Elastic Compute Nodes compress string dictionaries using front coding
- HW Service Worker
  - Task specification with task ID, function ID, source and target data Ids
  - Multi-cloud Kubernetes on Gardener
  - Scale dimensions:
    1. Configurable logic on FPGA <> `HW Function` (1:1)
    2. Increase / decrease #`HW Function` through adding / removing FPGAs; current data center rack limit 8-10 FPGAs per `HW Service Worker`
    3. Attach / Detach `HW Service Worker` components
- Execution Flow

- Intel IOFS: https://www.intel.com/content/www/us/en/products/details/fpga/platforms/pac/open-fpga-stack.html

SAP SE

Business Case

1) Potential cost reduction with one FPGA:

→ Re-pair compression ratio ~50%

→ AWS F1 FPGA (f1.2xlarge) costs <1,000 USD / month (1.06 USD/h, 730h usage per month)

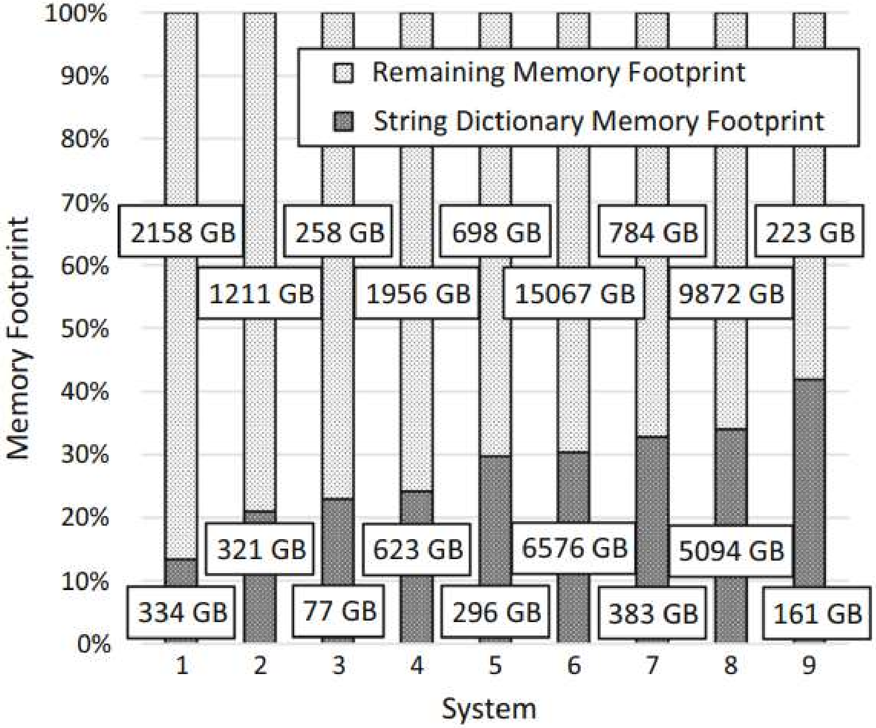
→ One FPGA can compress 8.6 TB/day of string dictionary (with CPU factor 17 less on Arria 10 / factor 34 on Stratix 10)

→ Save ~13,769 USD reduced DRAM with only one FPGA, used for several database instances

sap.com #sap

SAP intel

Business Case



1) Potential cost reduction with one FPGA:

→ Re-pair compression ratio ~50%

→ AWS F1 FPGA (f1.2xlarge) costs <1,000 USD / month (1.06 USD/h, 730h usage per month)

→ One FPGA can compress 8.6 TB/day of string dictionary (with CPU factor 17 less on Arria 10 / factor 34 on Stratix 10)

→ Save ~13,769 USD reduced DRAM with only one FPGA, used for several database instances

2) Real-world example from SAP ERP (system 9)

→ Memory footprint = 161 GB + 223 GB = 384 GB

→ Instance sizing with 384 GB and factor 2x overprosioning: 768GB DRAM => 12,592.5 CU => 10,000 USD / month (SAP HANA capacity estimator)

→ Re-Pair compression results in 80 + 223 = 303 GB => 10,073 CU => 8,060 USD / month

Research Challenges and Questions

Cloud Infrastructure and Operation

- → Missing FPGA resources in clouds / regions <> costs
- → "Out-of-hand": Operation / Monitoring, Security, Testing / Debugging
- → Scalability, Failures, HA etc.

Heterogeneous Compute

- → Joint workloads: FPGA, GPU, TPU
- → Combine "Bump-in-the-wire" with DASH
- → Further use cases (beyond)

…

Load Balancing and Data Management

- → Decentral, elastic scaling + used by several databases > scheduling strategies for long running tasks <> SLAs
- → Costs: scale-to-zero feasible?
- → `HW Task` chaining, FPGA2FPGA memory access (CXL) > more complex tasks

SAP SE

sap.com

#sap

Thank you!

Contact information:

Daniel Ritter
E-Mail: daniel.ritter@sap.com
DASH-Blog: SAP accelerates compression workload in POC with Intel® OFS - Intel Communities

Cloud Costs, Latencies