nKV: Near-Data Processing with KV-Stores on Native Computational Storage





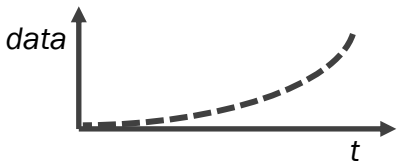
Tobias Vinçon, Arthur Bernhardt, Ilia Petrov
Data Management Lab
Reutlingen University, Germany

Lukas Weber, Andreas Koch
Embedded Systems and Applications Group
TU Darmstadt, Germany

Motivation

High Insertation
+
Update Intensive Workload

High Throughput
+
Analytical Workload



We are „living in exponential world“
Gray et al. [1]

DBMS

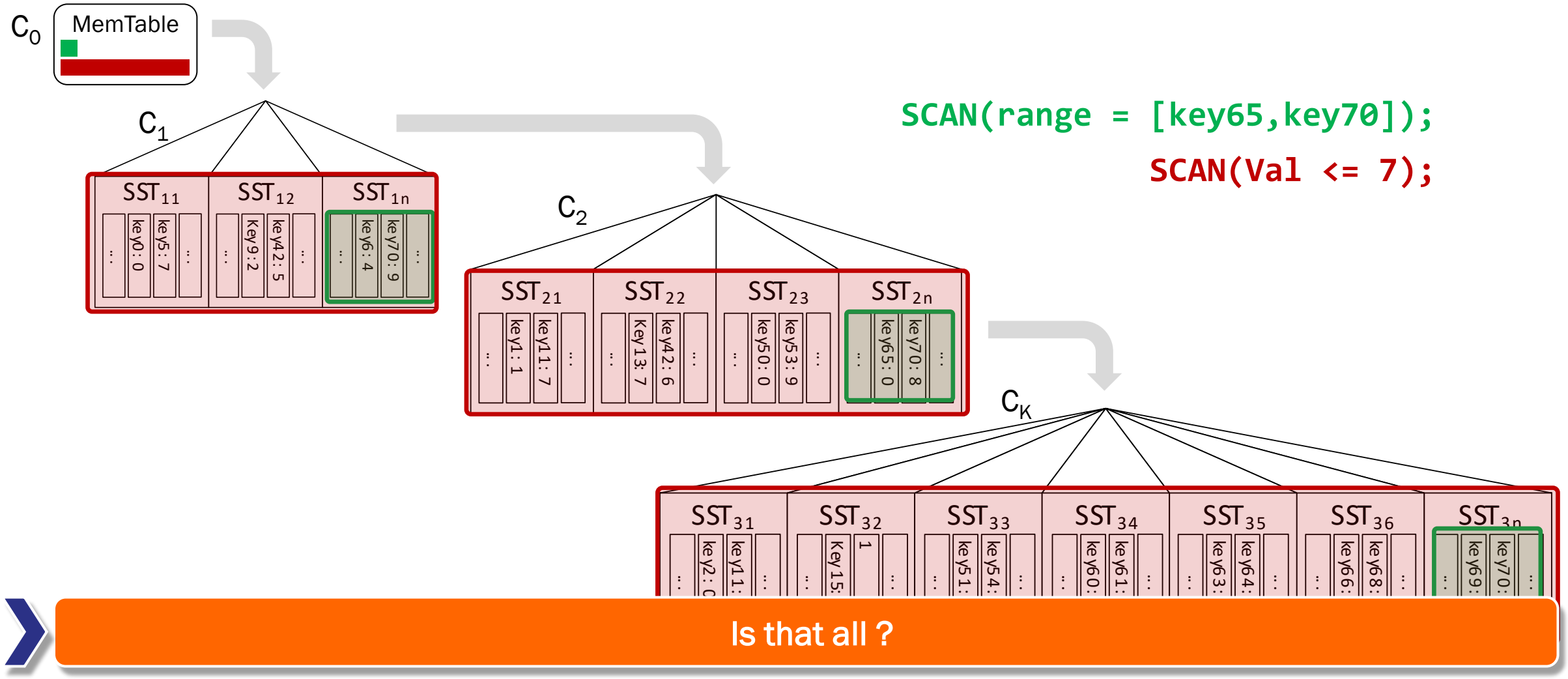
Storage

- Increase in data transfers
- Impair performance/scalability
- Hurt resource-/energy-efficiency

But why ?

[1] A. Szalay, J. Gray; Science in an exponential world

Read Amplification of LSM-Trees

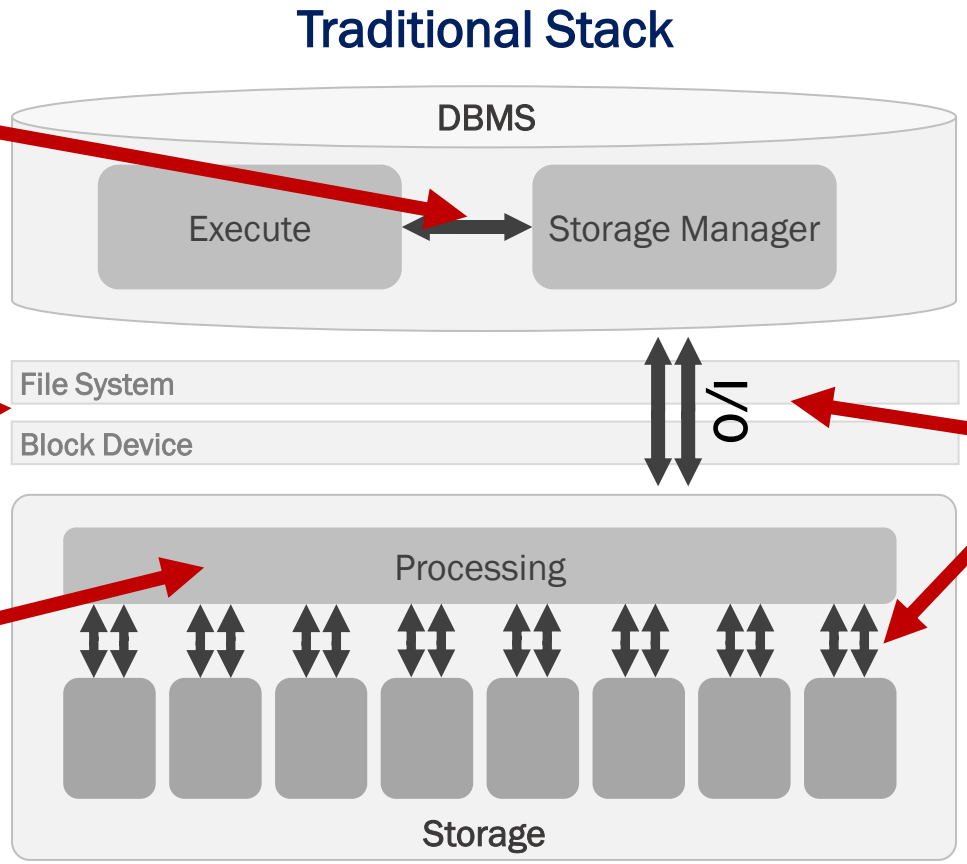


System Drawbacks

System Paradigm:
only „Data-to-Code“

Intermediate Layers:
introduce levels of indirection

Resources:
used for compatibility



Bandwidth:
Device ↔ Host ≪ On-Device

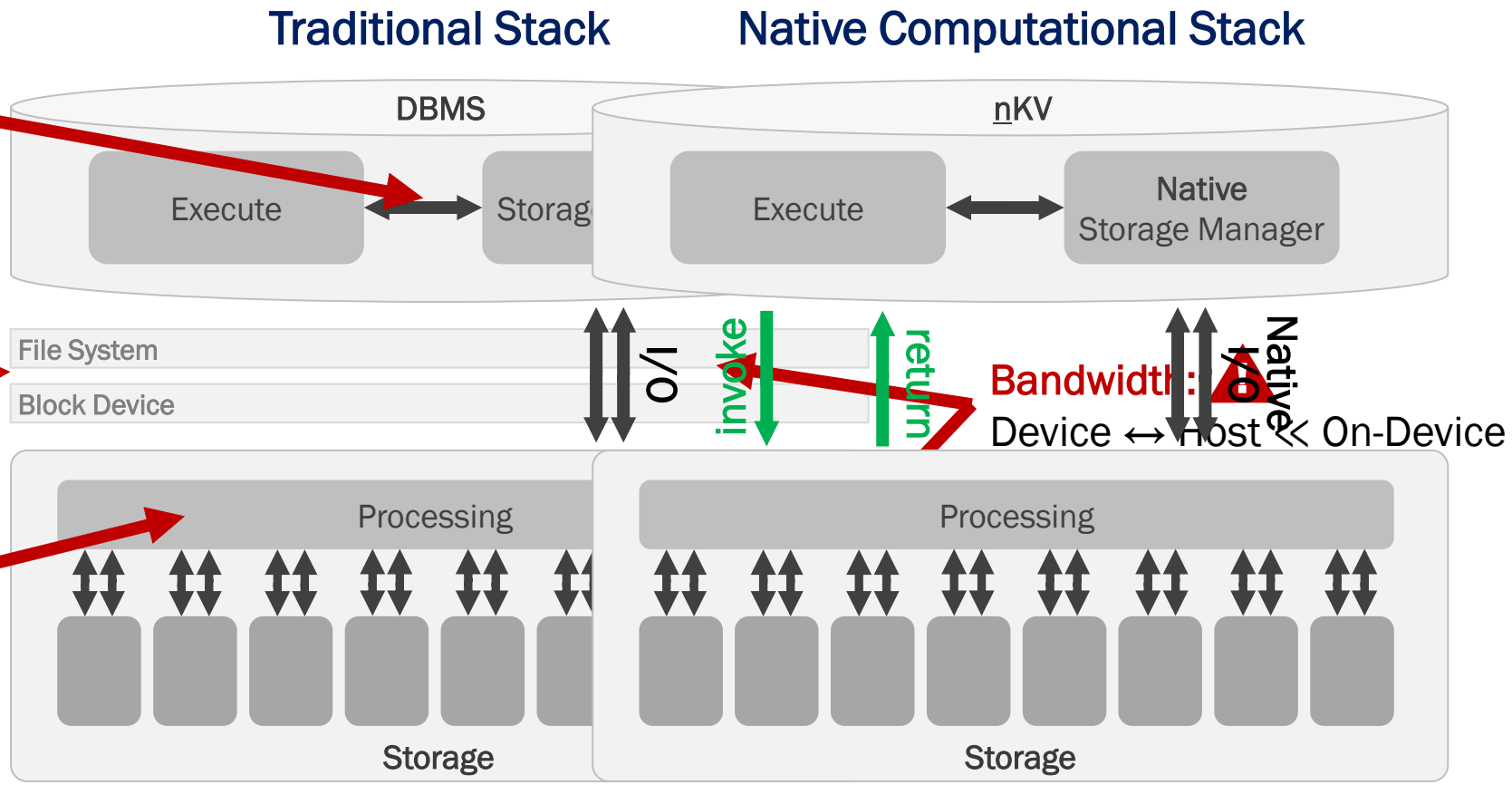
Several elements impair performance – how do we address these?

Native Computational Stack

System Paradigm:
only „data-to-code“

Compatibility Layers:
introduce levels of indirection

Resources:
used for compatibility



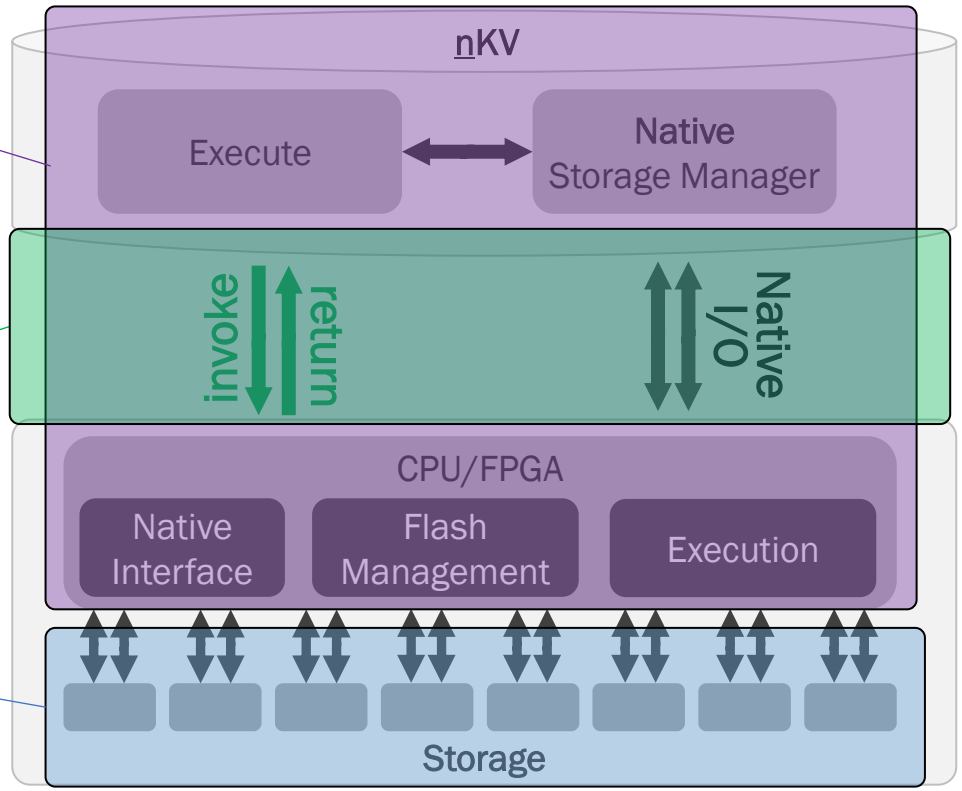
nKV uses NDP to utilise on-device resources and to reduce data transfers

Key Aspects for “Code-to-Data”

Computation Placement:
- Depending on the operation/workload computation can be placed in Host/on-Device CPU or FPGA

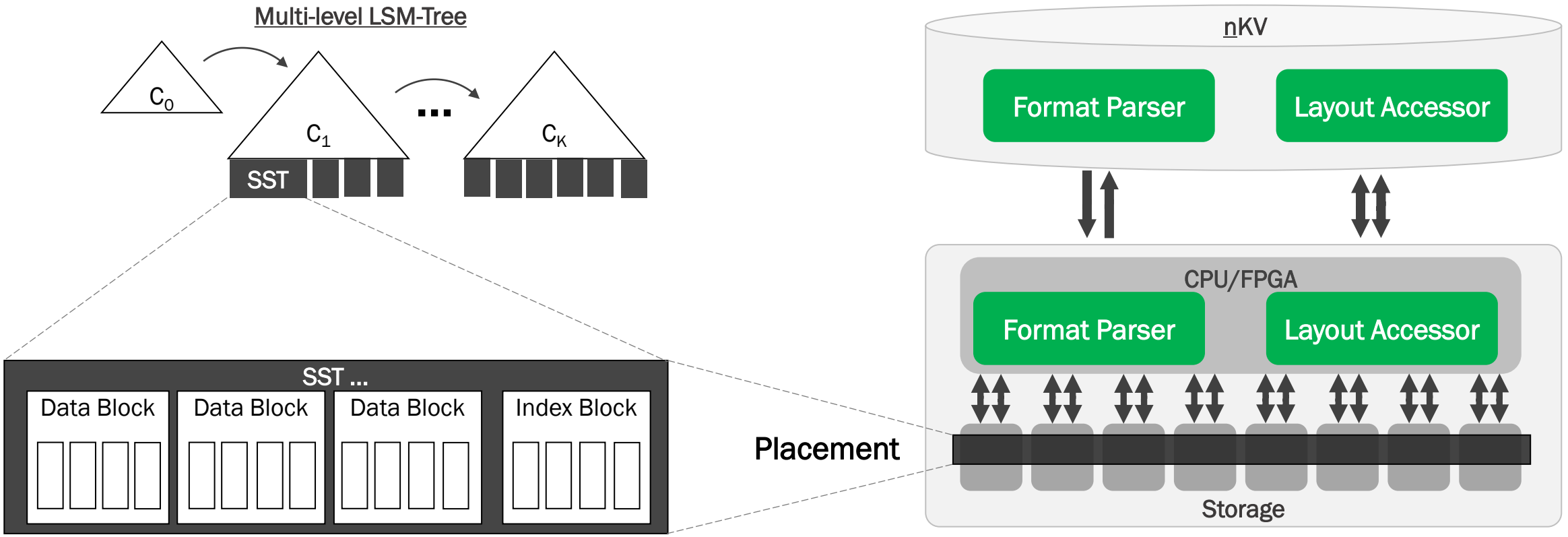
NDP Interface Extension:
- NVMe Support: Seamless integration in Database
- Synchronous and Asynchronous task scheduling

Native Storage:
- Physical Data Placement
- Leverage internal Flash parallelism e.g. Channels/Luns



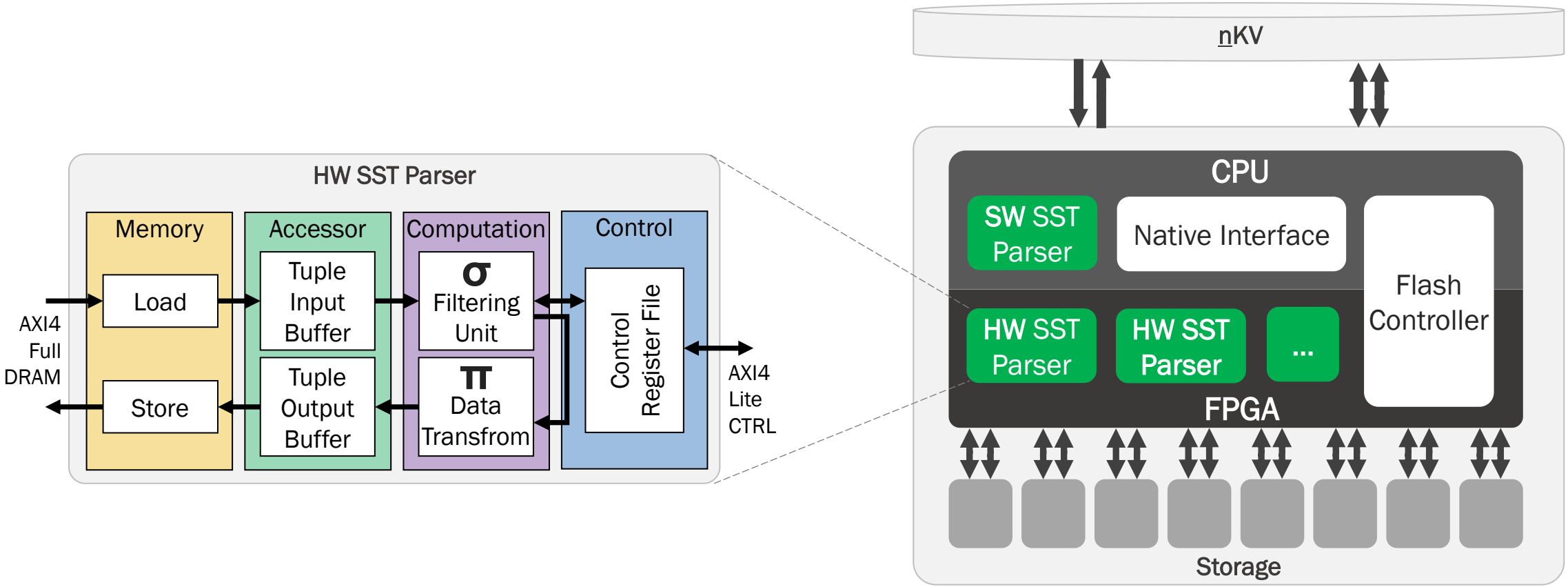
Native Storage, NDP interface extension and computation placement are essential parts of nKV

On-device Data Parsing



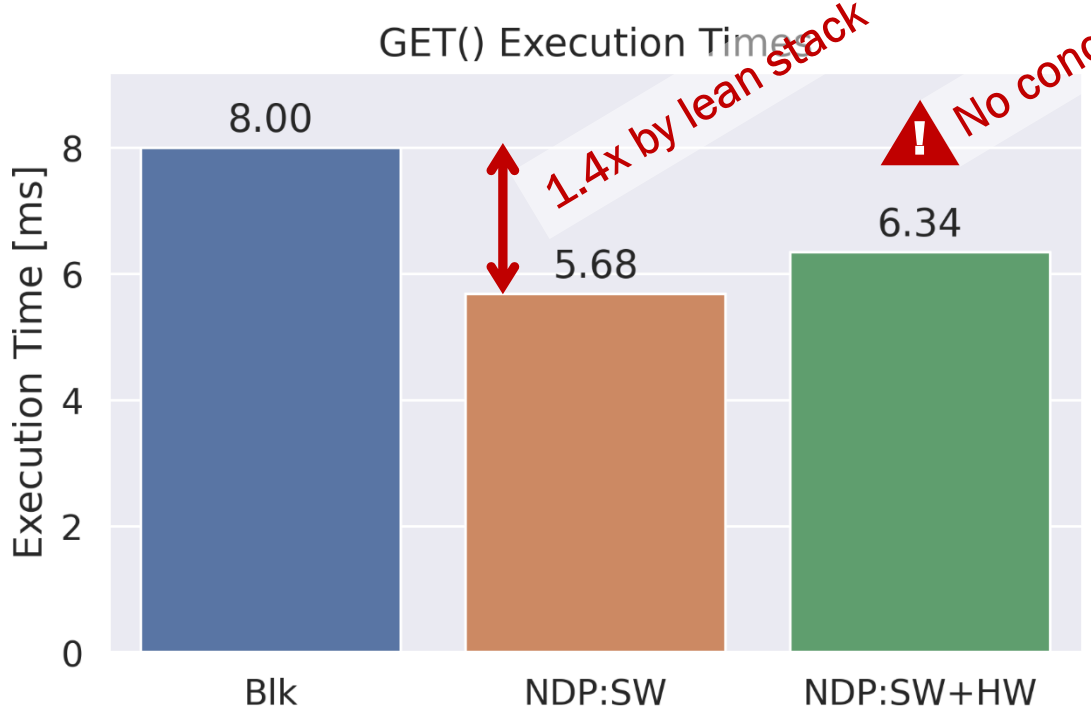
Accessors and parsers must be available in the host and on-Device for nKV

HW Support

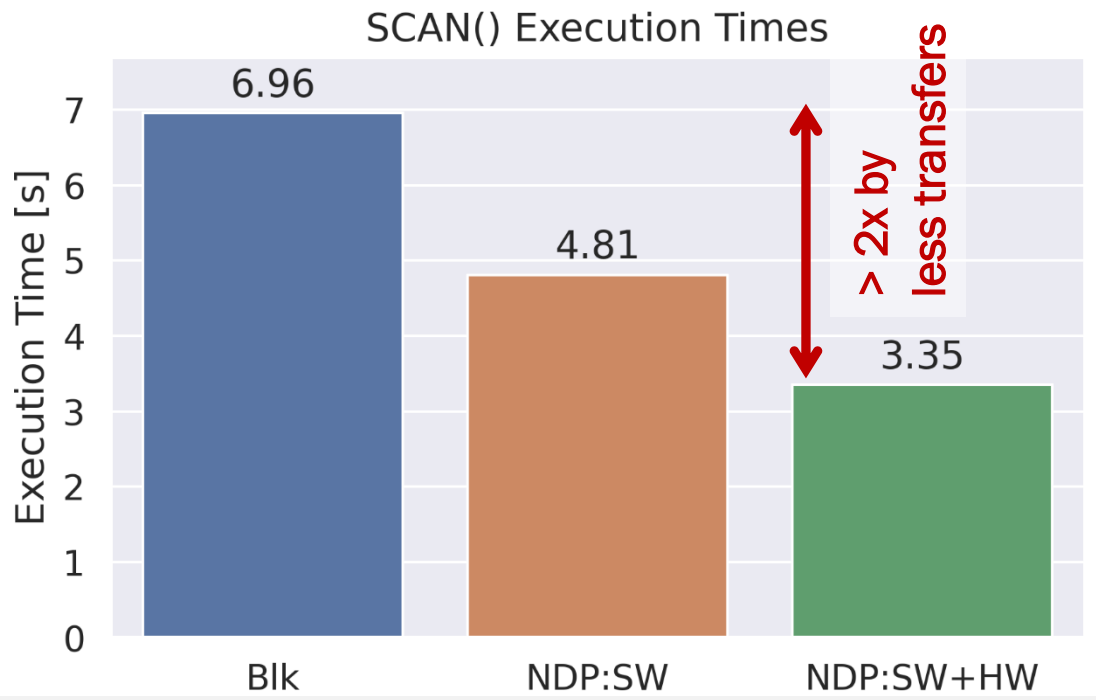


HW/SW Co-Design of nKV allows leveraging hardware resources efficiently

Experimental Evaluation



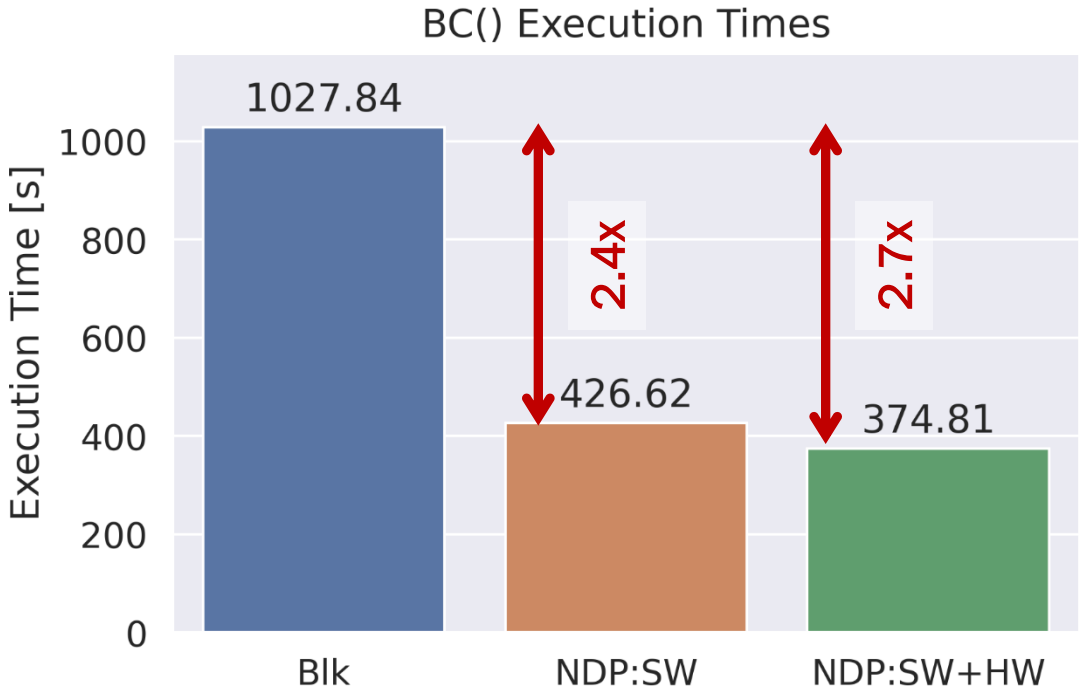
Experiment 1: Lean Native Stack

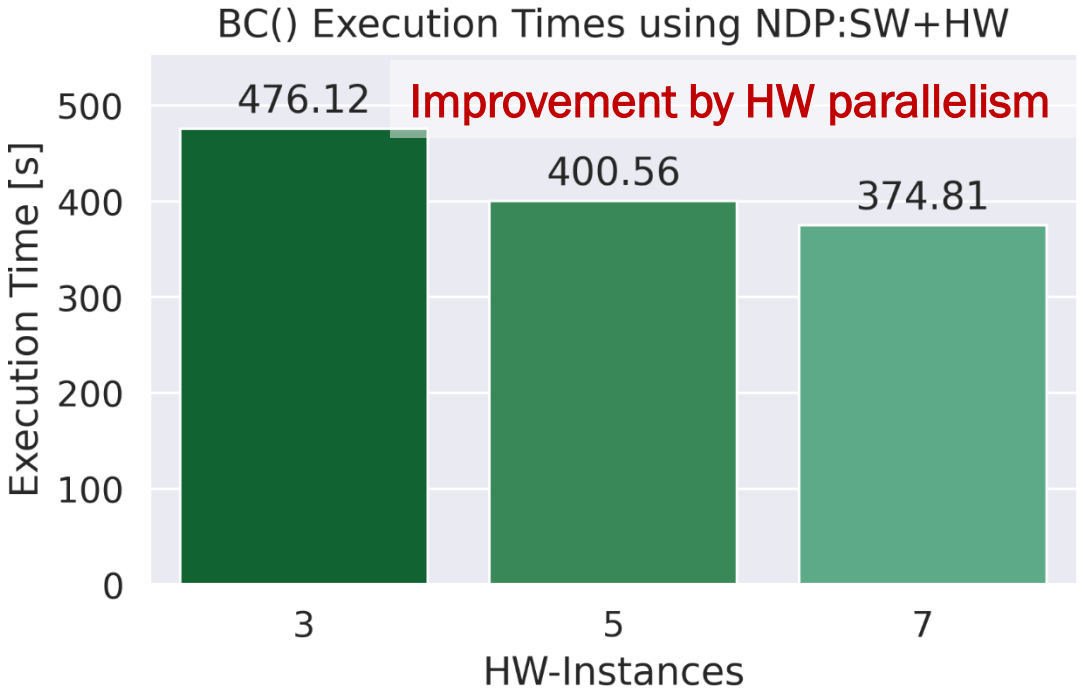


Experiment 2: Data Transfer Reduction

nKV's lean stack and less data transfers improve performance

Experimental Evaluation





Experiment 3: Lean Native Computational Storage

Experiment 4: Execution Parallelism

nKV speed up complex queries by a lean stack, less data transfers and leveraging hardware resources

Conclusion

nKV

native Computational Storage

- **Elimination of intermediary layers** for compatibility reasons (e.g. block device)
- **Physical data placement** according to storage device characteristics (e.g. Channels/LUNs)
- Seamlessly **embedded NVMe interface** with (a)/synchronous task scheduling

near-Data Processing

- **NDP interface extension** with a generic command
- **Data format parsers and accessors** in software and hardware
- **Flexible scheduling** across on-Device processing units (e.g. CPU/FPGA)

Evaluation

- **Lean stack** of simple lookups: **1.4x**
- **Data transfer reduction: >2x**
- **HW/SW Co-designed** of complex queries: **2.7x**

https://dblab.reutlingen-university.de
https://www.esa.informatik.tu-darmstadt.de

TECHNISCHE UNIVERSITÄT DARMSTADT

Hochschule Reutlingen
Reutlingen University

Embedded Systems & Applications

DBLab