



Sandia
National
Laboratories

Offloading Data Management Services to SmartNICS

Craig Ulmer, Sandia National Labs, California

Jianshen Liu, Aldrin Montana, & Carlos Maltzahn, *University of California Santa Cruz*
Matthew L. Curry, Scott Levy, & Whit Schonbein, *Sandia National Labs, New Mexico*
John Shawger, *University of Wisconsin-Madison*



U.S. DEPARTMENT OF
ENERGY

Office of
Science

This material is based upon work supported
by the U.S. Department of Energy, Office of
Science, Office of Advanced Scientific
Computing Research under Field Work
Proposal Number 20-023266.

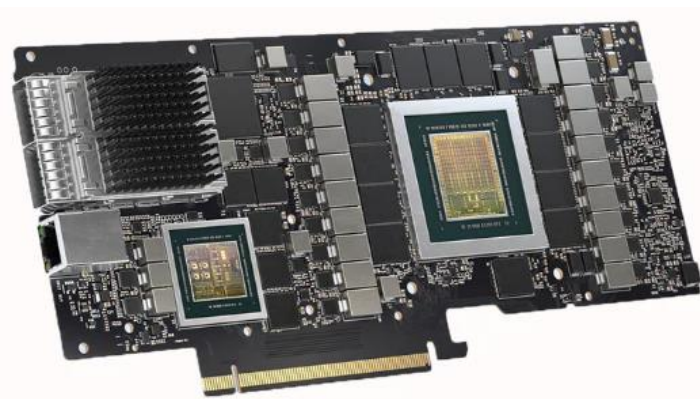
UC SANTA CRUZ
CROSS | CENTER FOR RESEARCH IN
OPEN SOURCE SOFTWARE



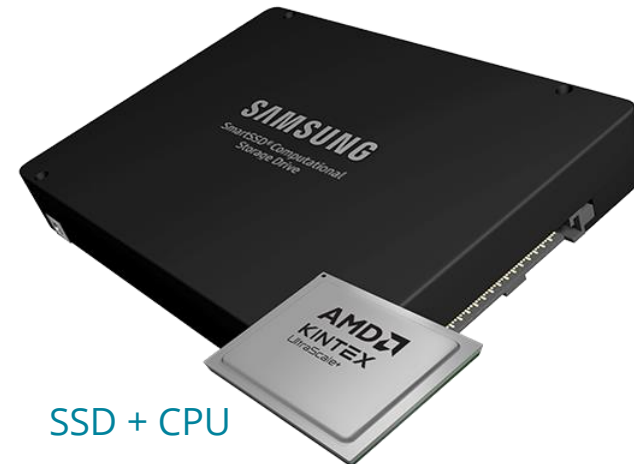
Sandia National Laboratories is a
multimission laboratory managed
and operated by National Technology
& Engineering Solutions of Sandia,
LLC, a wholly owned subsidiary of
Honeywell International Inc., for the
U.S. Department of Energy's National
Nuclear Security Administration under
contract DE-NA0003525.

SAND2024-03874PE

- Hardware vendors are adding *embedded processors* to network & storage products
 - Enables users to execute code on in-transit or in-storage data
 - Use cases: infrastructure security, resilience, embedded queries
- How can we leverage these devices in scientific computing workflows?
- DOE ASCR Project: Offloading Data Management Services to SmartNICs (FY20-23)
 - Focus on *data services* used for moving data in HPC workflows
 - Leverage *Apache Arrow* as a data standard, Sandia's *Faodel* for communication



Network Interface Card + CPU (+ GPU)



SSD + CPU

3



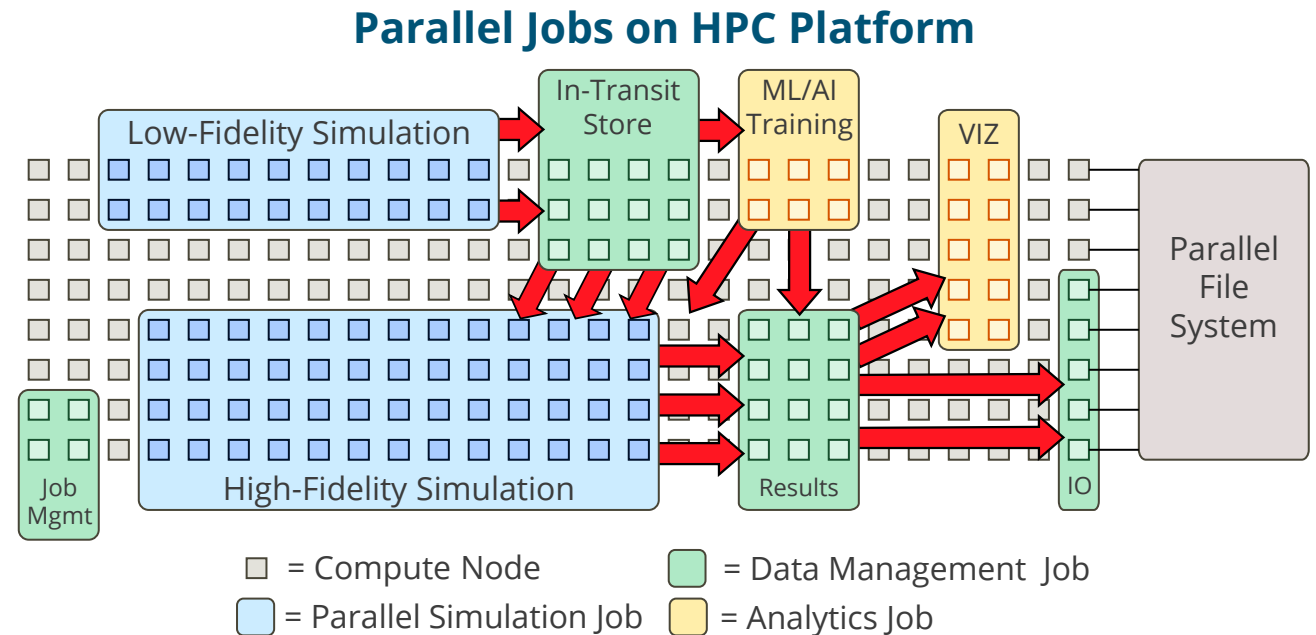
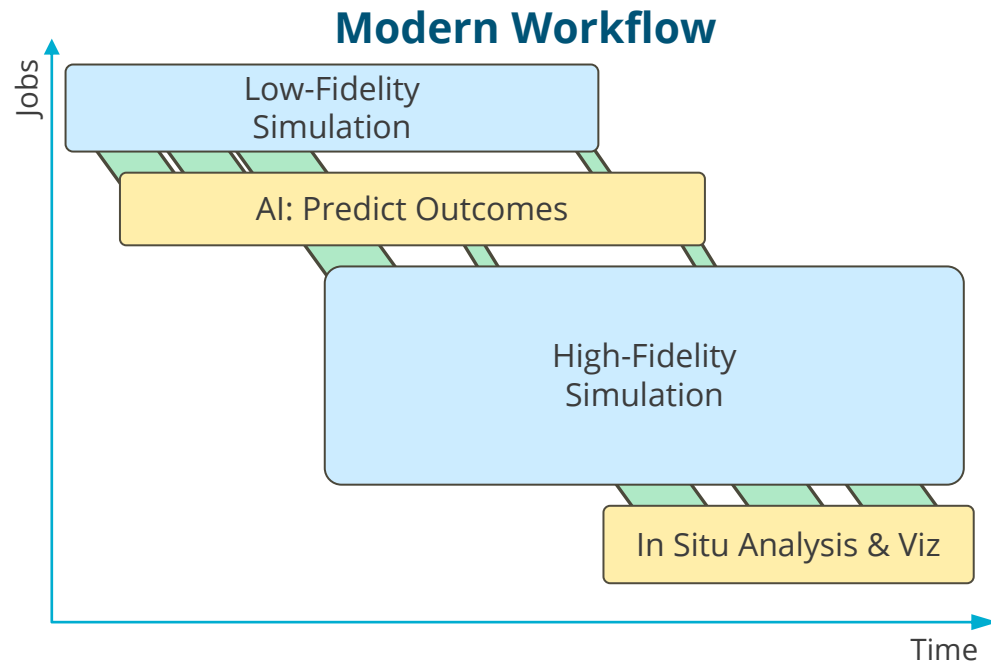
Motivation



High-Performance Computing Workflows



- ECP/ATDM: Advanced workflows involve many parallel tools
 - Use *data management services* to implement data flow between jobs, hide I/O costs
 - Multiple library options: DataSpaces, Mochi, Conduit, Faodel
- Problem: Data management services consume host resources



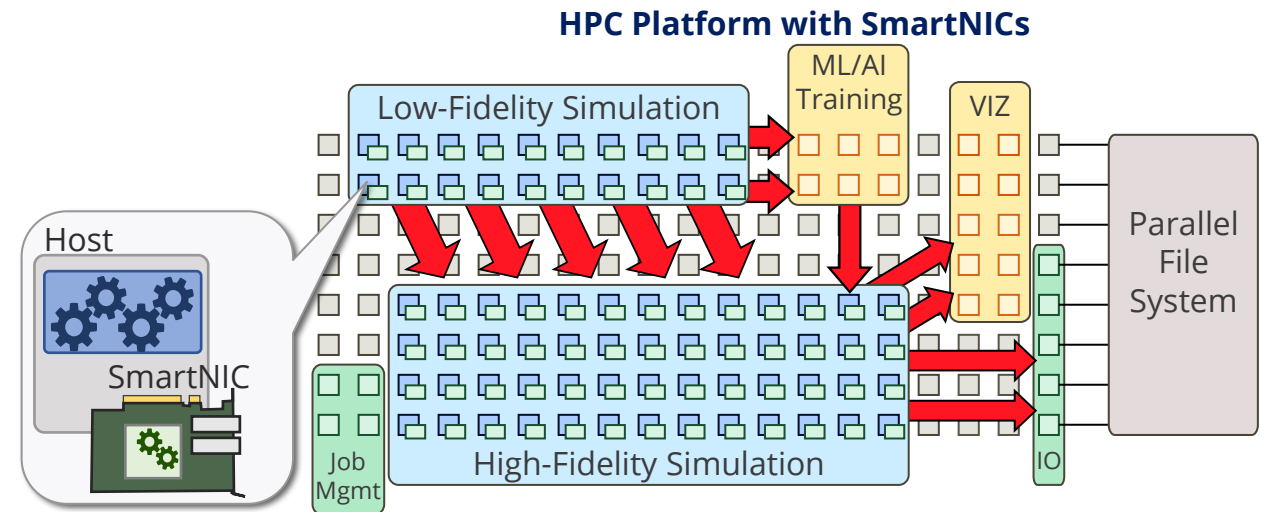
Smart Network Interface Cards (SmartNICs)



- Network vendors now offer SmartNICs with *user-programmable* resources
 - Examples: NVIDIA BlueField-2 DPU, Intel IPU, and FPGA
 - Embedded processors provide isolated space for caching and processing in-transit data
- Emerging HPC platforms include SmartNICs
 - How do we make an environment for hosting data services in SmartNICs?



BlueField-2 DPU SmartNIC



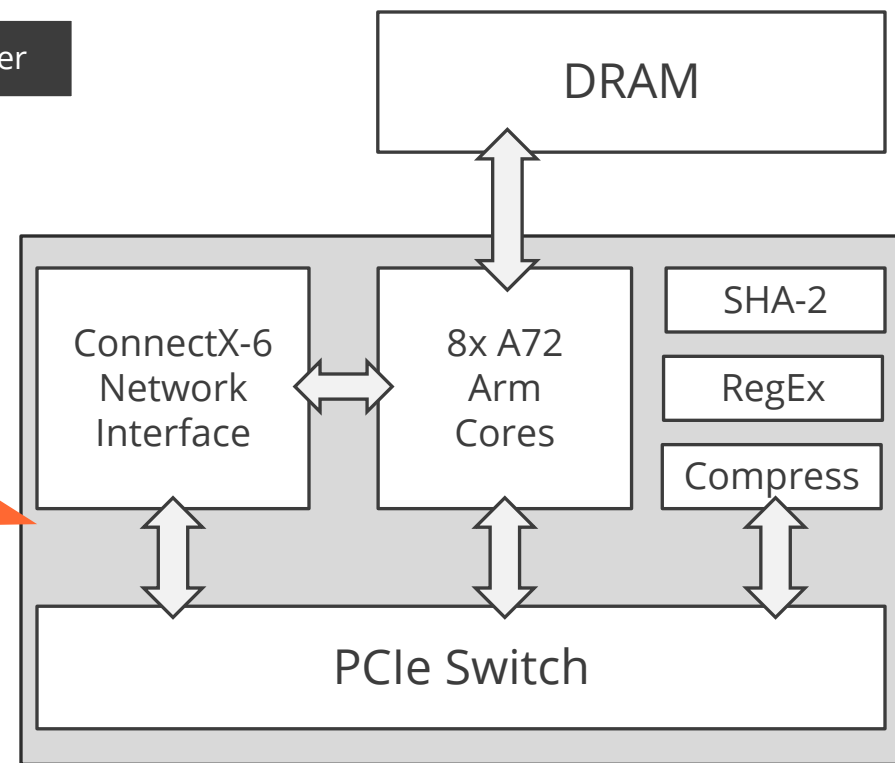
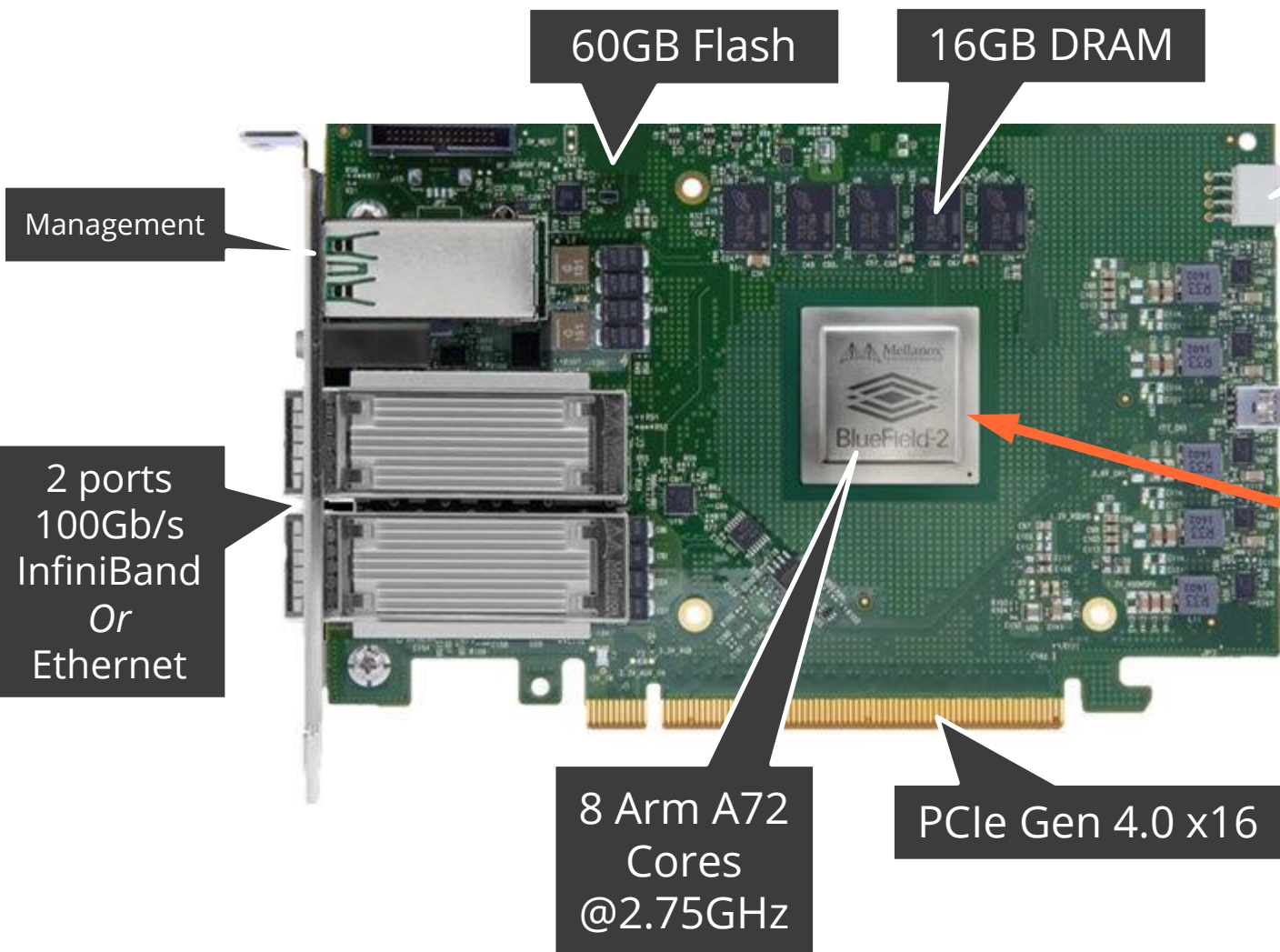
 = Compute Node with a *SmartNIC* for offloading data services



Characterizing the BlueField-2 SmartNIC



NVIDIA BlueField-2 VPI SmartNICs



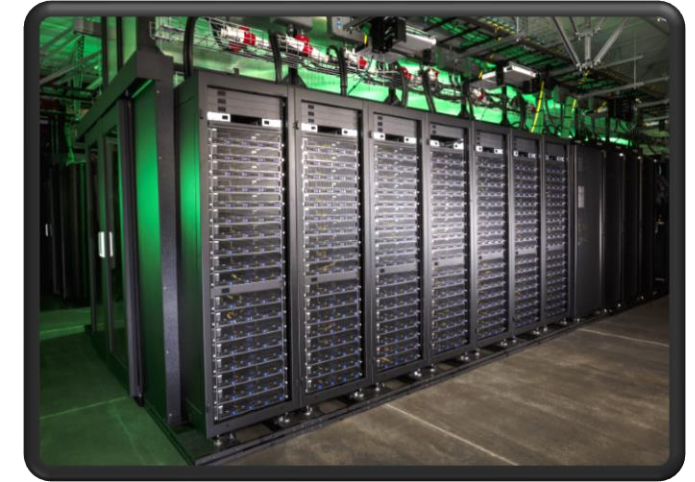
Cost: ~\$2,000
Power: 30W idle, 42W active, 63W max
OS: Linux (Ubuntu, RHEL, etc)

Sandia's Glinda Cluster (2021)



- Heterogenous architecture for Data Analytics
 - BlueField-2 DPU, AMD Zen 3 CPUs, NVIDIA A100
 - 126 Nodes
- BlueField-2's Embedded processors
 - 4-10x slower, but better power efficiency

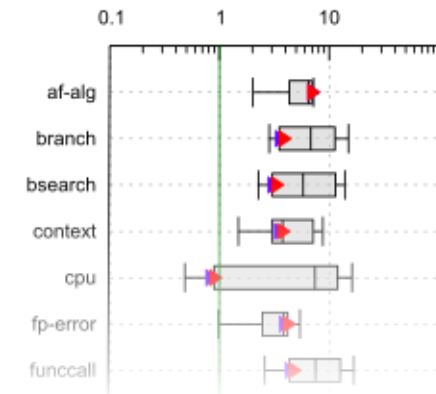
	SmartNIC	Host CPU	GPU
Processor	Arm A72	AMD EPYC 7543P	Ampere A100
Cores	8	32	108 SMs
Clock	2.75GHz	2.8GHz	0.765 – 1.41GHz
L1 Cache	256KB	1MB	192KB
L2 Cache	6MB	256MB	-
Memory Capacity	16GB	512GB	40GB
Memory Bandwidth	25GB/s	204GB/s	1,555GB/s
TDP	63W	225W	250W



Glinda

stress-ng microbenchmarks

- BF2 vs 12 servers in 200 tests
- BF2 lower, but within order of magnitude



"Performance Characteristics of the BlueField-2 SmartNIC"



Creating an Environment for Data Services on SmartNICs

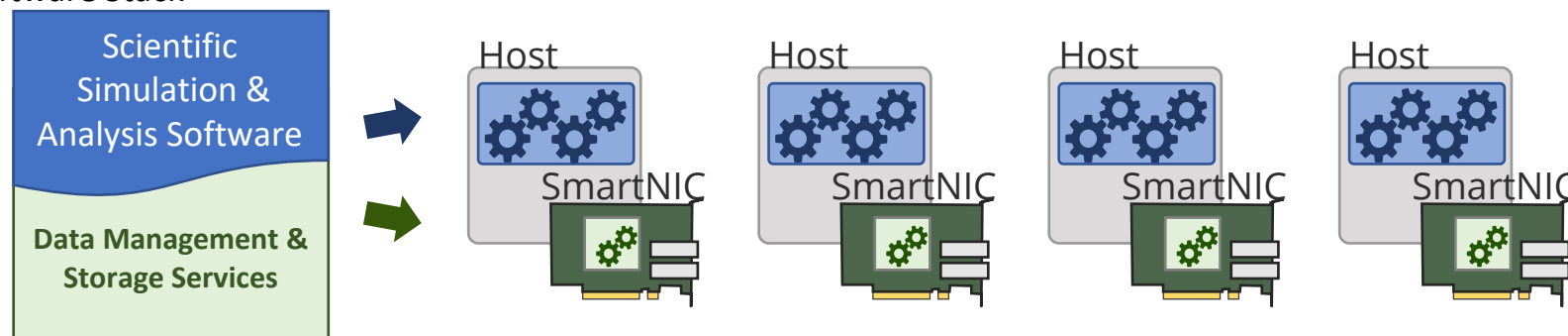


Create an Environment for Hosting Data Services on SmartNICs



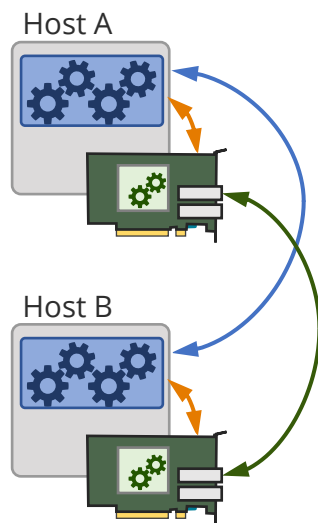
- We define five requirements (R1-R5) for creating this environment
 - Three communication, Two computation
- Leverage existing libraries as much as possible
- Prototype environment
 - Communication via **Faodel**: C++ library with distributed-memory Key/Blob API built on RDMA
 - Computation via **Apache Arrow**: C++ library for processing in-memory tabular data

Software Stack



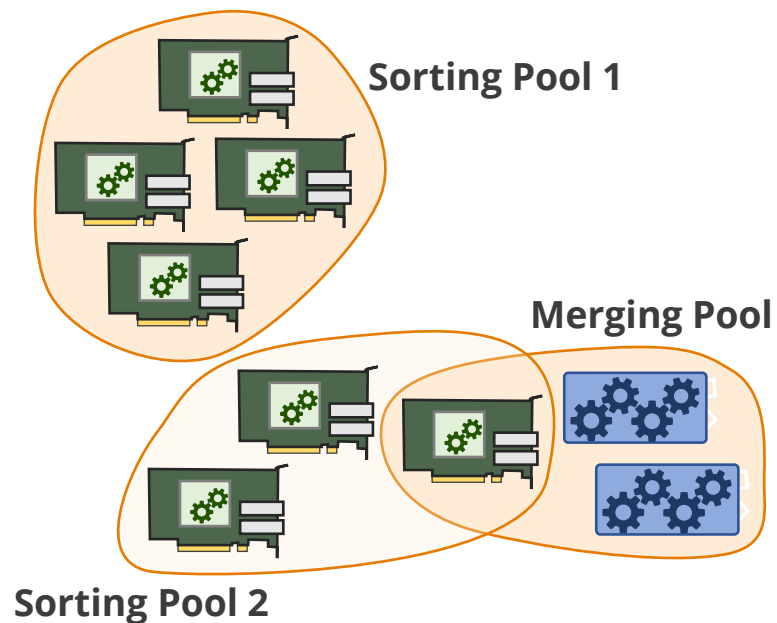
R1: Any-to-Any Transfers

- Faodel has globally accessible endpoints
- Host and SmartNICs can be endpoints
- Put/Get remote objects
- RDMA for point-to-point transfers



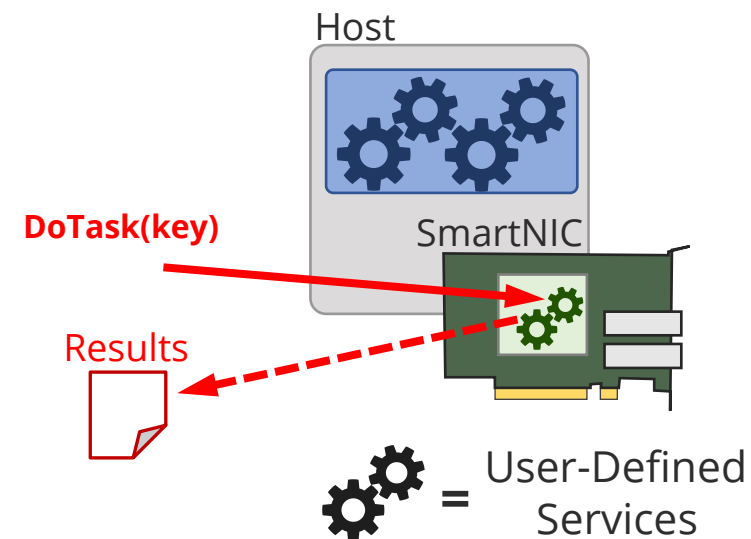
R2: Group Resources

- Faodel uses a Pool abstraction
- Pool has endpoints & distribution policy
- Work mapped to resources at run time



R3: Dispatch Computations

- Faodel primarily moves data
- Invoke remote operation on object
- Local main can also make decisions





R4: Common Data Representation

- Arrow provides robust data structures for 2D data
- Efficient in-memory storage
- Built-in functions to serialize



ID	Time	Pos _{xyz}	Vel _{xyz}
100	7	XYZ	XYZ
714	7	XYZ	XYZ
867	7	XYZ	XYZ
943	7	XYZ	XYZ
483	7	XYZ	XYZ
...			

Simulation

ID	Time	Pos _{xyz}	Vel _{xyz}
100	7	XYZ	XYZ
714	7	XYZ	XYZ
867	7	XYZ	XYZ
943	7	XYZ	XYZ
483	7	XYZ	XYZ
...			



Serialized Data

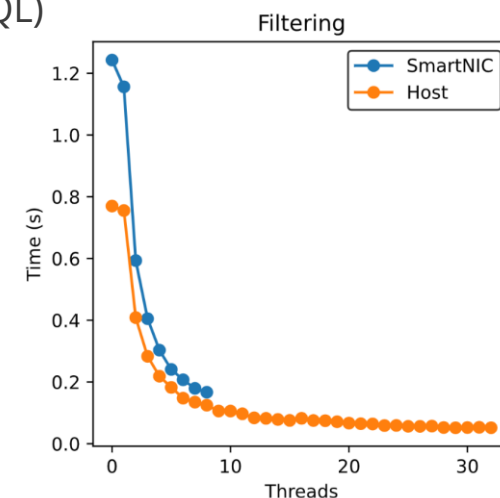
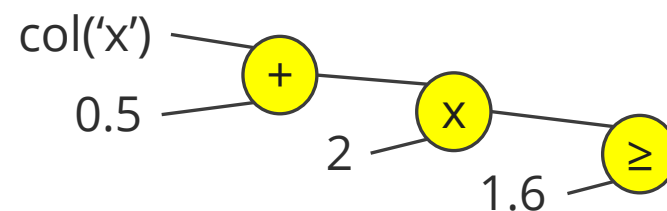


ID	Time	Pos _{xyz}	Vel _{xyz}
100	7	XYZ	XYZ
714	7	XYZ	XYZ
867	7	XYZ	XYZ
943	7	XYZ	XYZ
483	7	XYZ	XYZ
...			

Analysis

R5: Data-Parallel Computations

- Arrow includes compute functions for tables
- Target for higher-level languages (SQL)
- Thread- and SIMD-Aware



```
// 2 * (0.5 + x) >= 1.6
auto filter_expression = arrow::compute::greater_equal(
    arrow::compute::call(
        "multiply",
        {arrow::compute::literal(2),
         arrow::compute::call("add_checked", {arrow::compute::literal(0.5),
                                                arrow::compute::field_ref("x")})}),
    arrow::compute::literal(1.6));
```



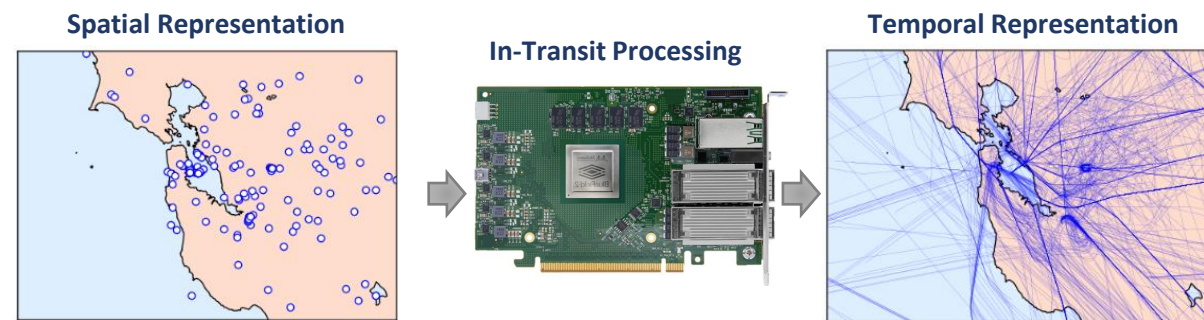
Particle Sifting Example



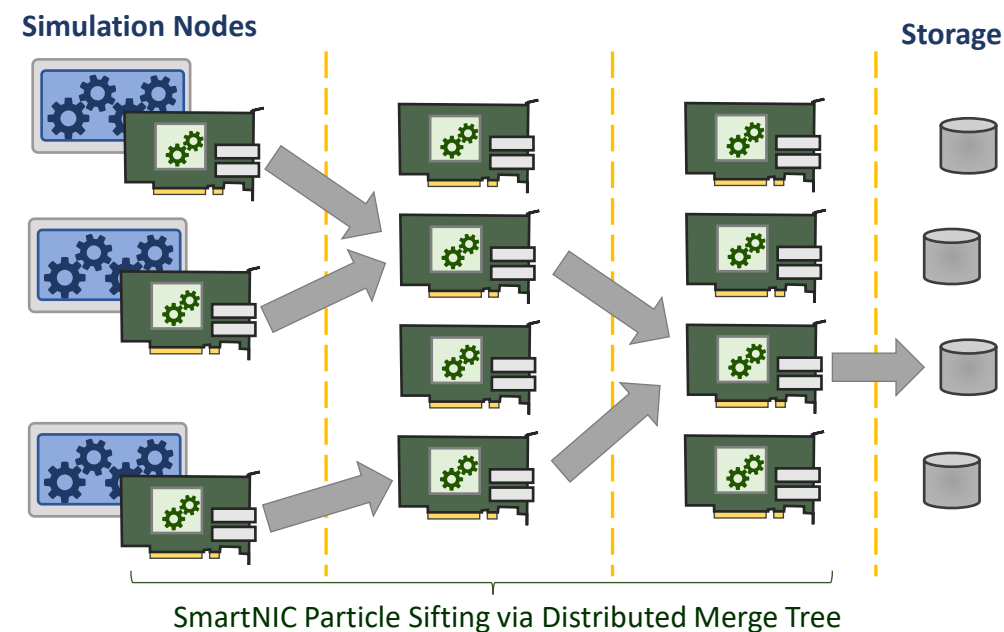
Example: Reorganizing Particle Simulation Results



- Particle simulations track billions of particles
- Mismatch between producers/consumers
 - Simulations: Sorted by position and time
 - Analytics: Sorted by ID and time
- Particle sifting service
 - Periodically sample current data
 - Use distributed SmartNICs to reorganize
 - Distributed merge tree sorts data by ID
- Implementation
 - Faodel Pools/Keys to control data flow
 - Arrow compute to split data
- Experiments on 100-node Cluster w/ BlueField-2 DPUs



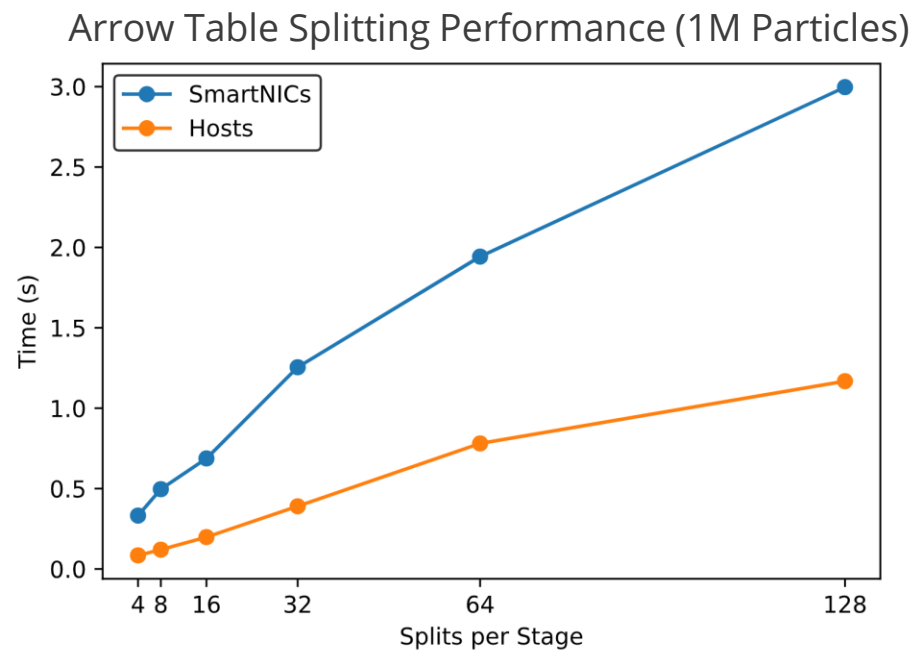
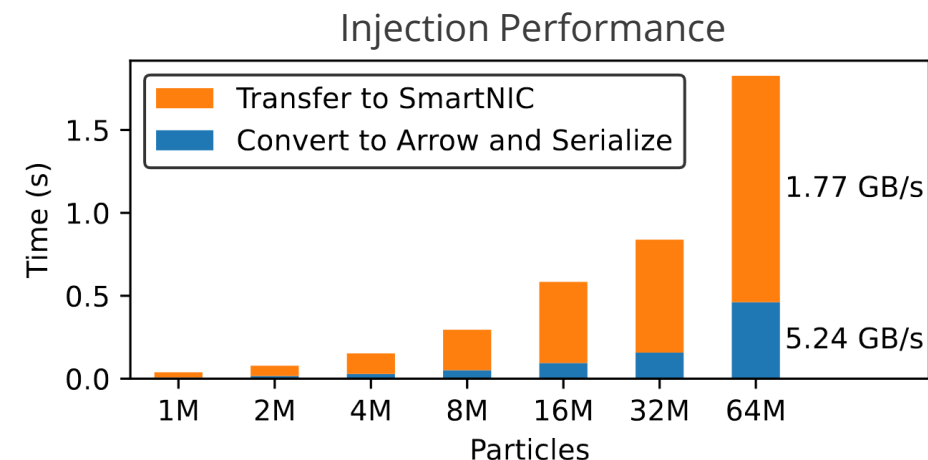
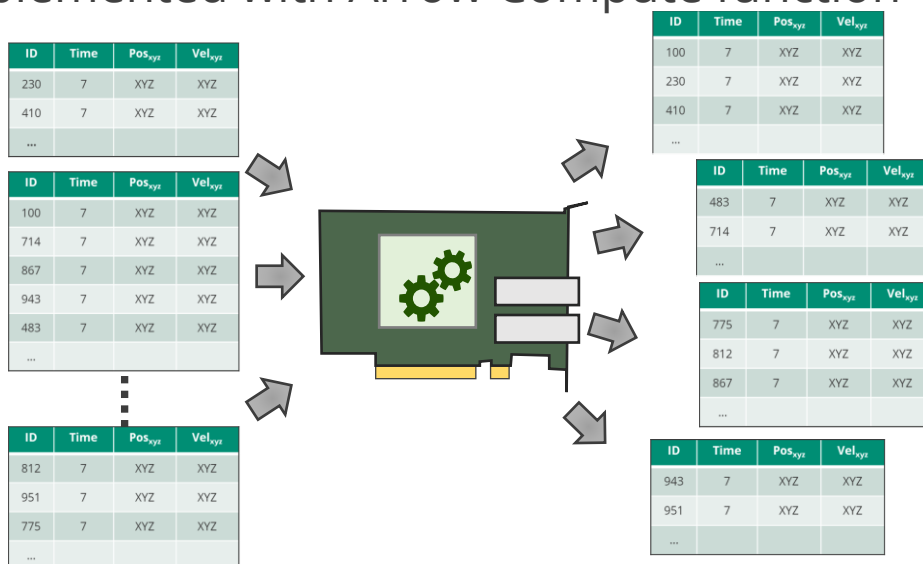
SmartNICs enable simulation results to be transformed while in transit to storage.



Performance Measurements



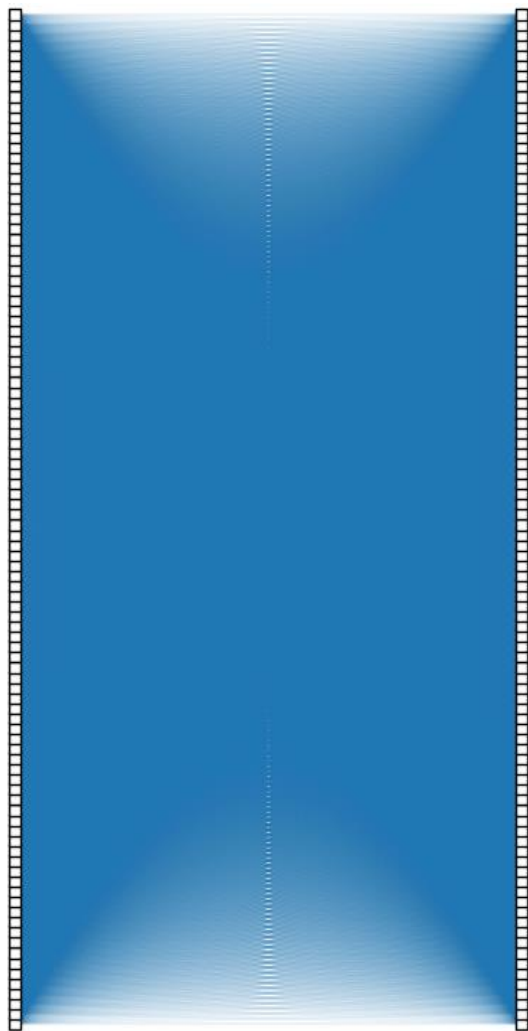
- Injection
 - Convert to Apache Arrow's serialized IPC format
 - Transfer to local SmartNIC
 - 1M-64M Particles (37MB-2.4GB), Overall: 1.32GB/s
 - **New Work:** 10GB/s for "Serialize-on-Transfer"
- Splitting Tables
 - Merge incoming tables and split based on particle IDs
 - Implemented with Arrow Compute function



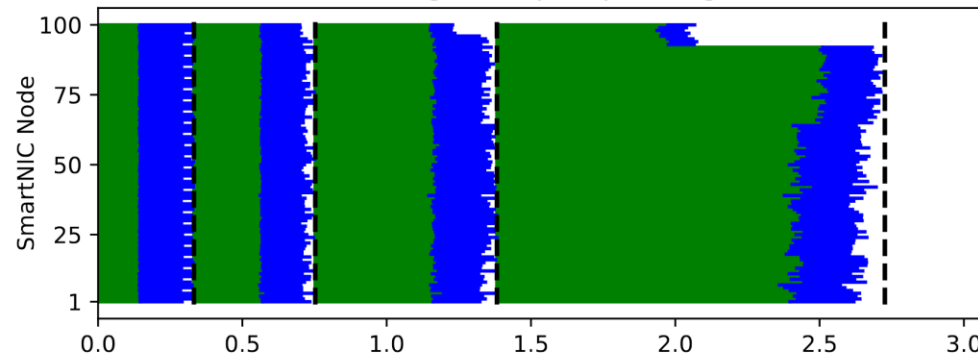
Overall Sifting Performance: 100M Particles on 100 SmartNICs



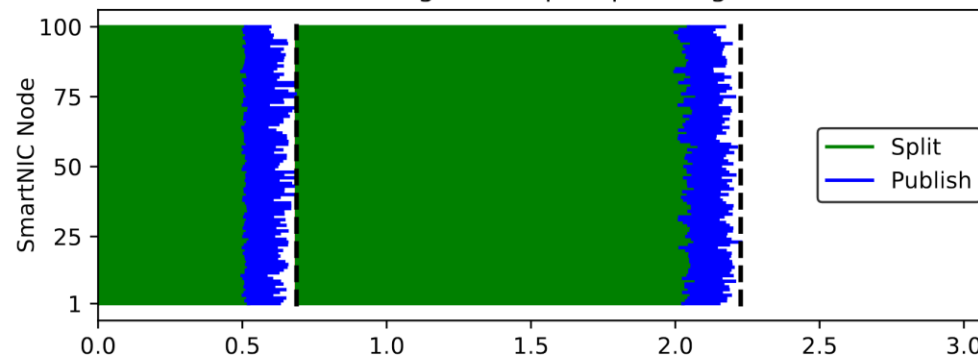
1-Stage, 128-Splits/Stage



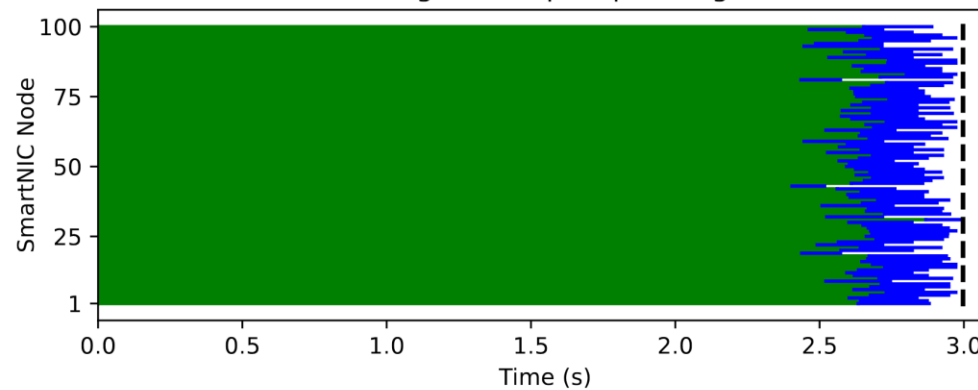
4 Stages, 4 Splits per Stage



2 Stages, 16 Splits per Stage



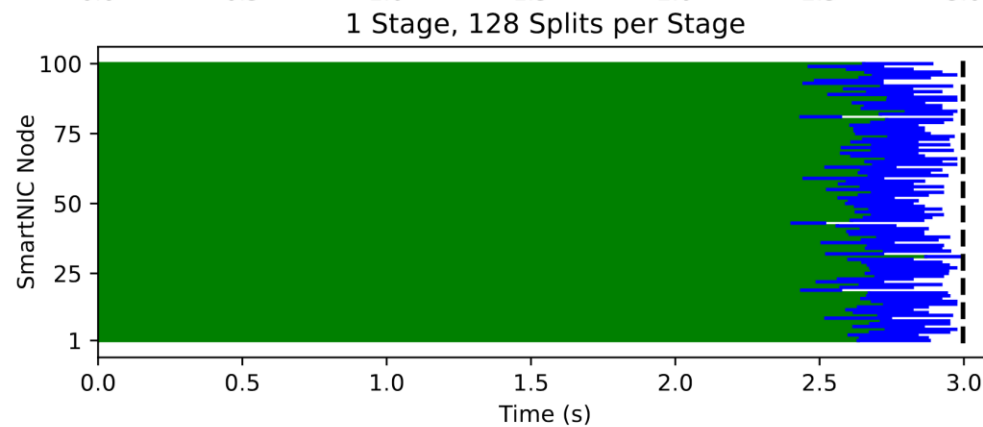
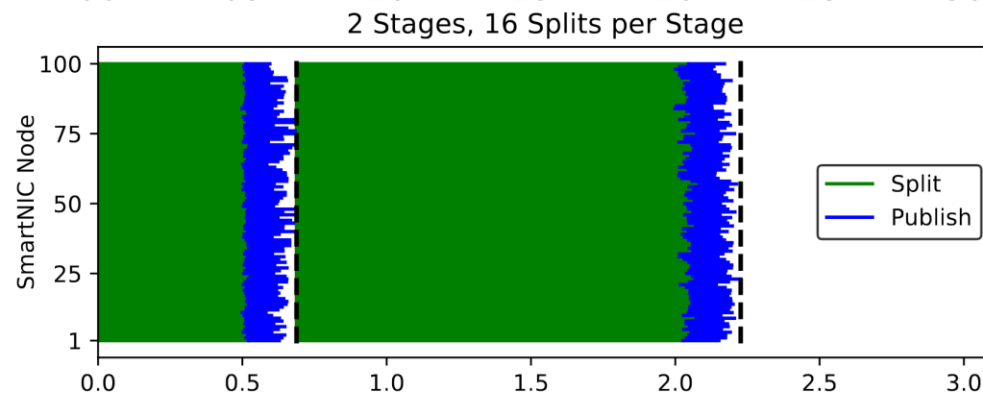
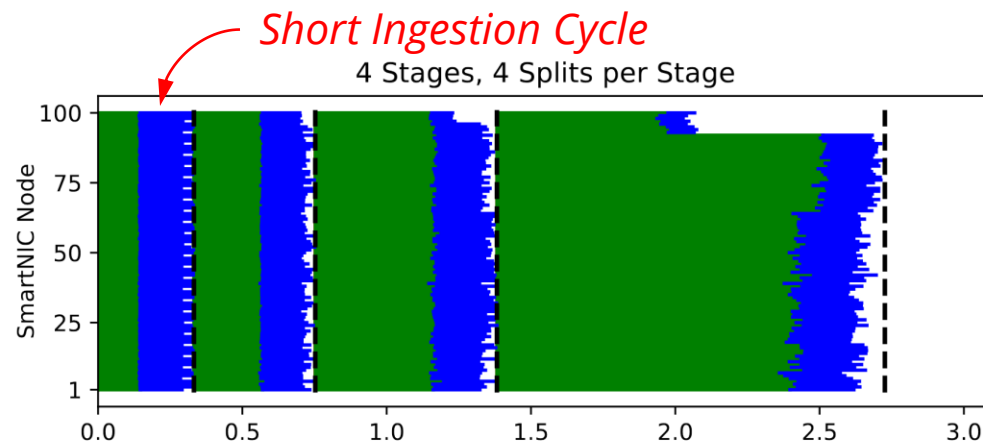
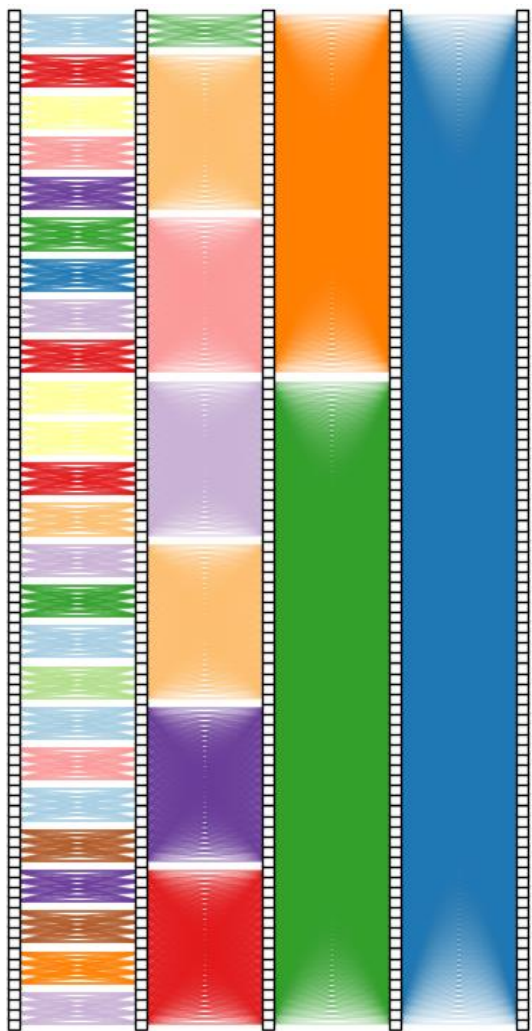
1 Stage, 128 Splits per Stage



Overall Sifting Performance: 100M Particles on 100 SmartNICs



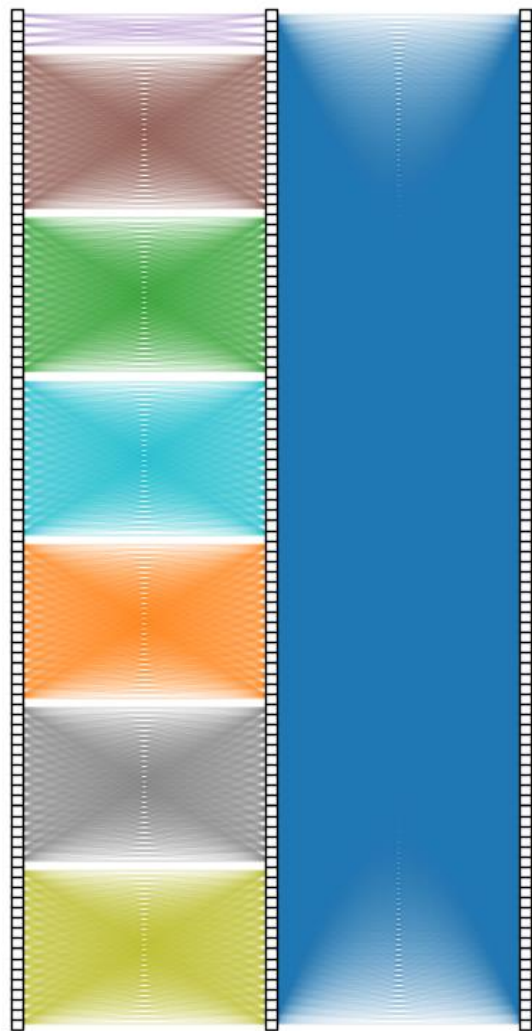
4-Stages, 4-Splits/Stage



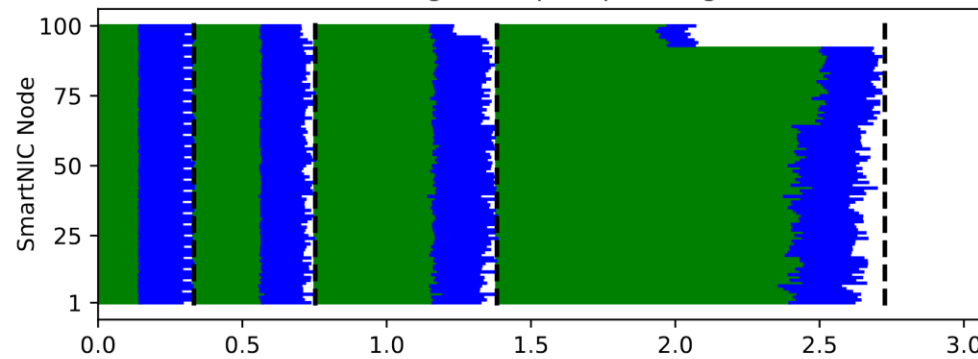
Overall Sifting Performance: 100M Particles on 100 SmartNICs



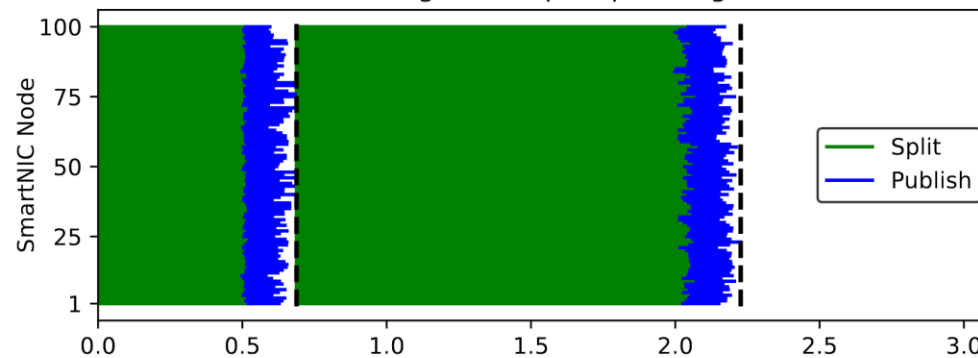
2-Stages, 16-Splits/Stage



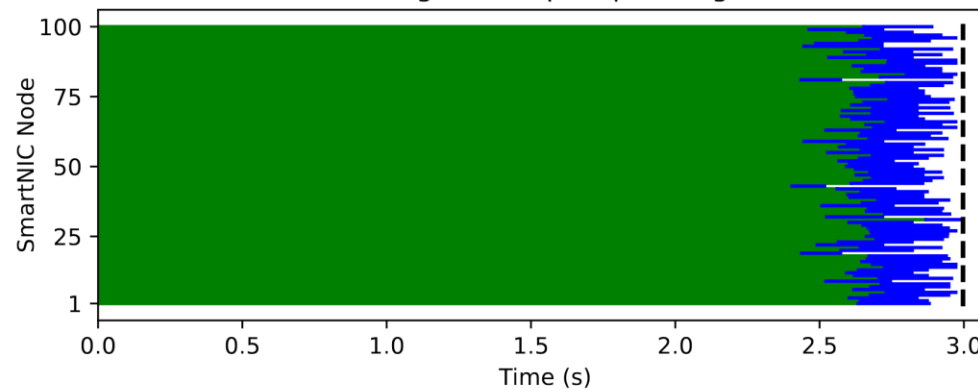
4 Stages, 4 Splits per Stage



2 Stages, 16 Splits per Stage



1 Stage, 128 Splits per Stage



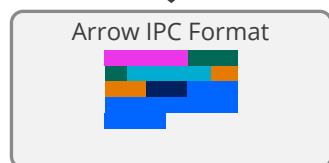
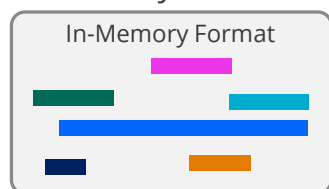
Additional Research Threads Explored



Injection Optimizations

- **Problem:** Serialization is expensive
- **Goal:** Minimize output time
- Defer serialization to SmartNIC
 - Reorganize on SmartNIC or
 - Align fragments on transfer

Host Memory



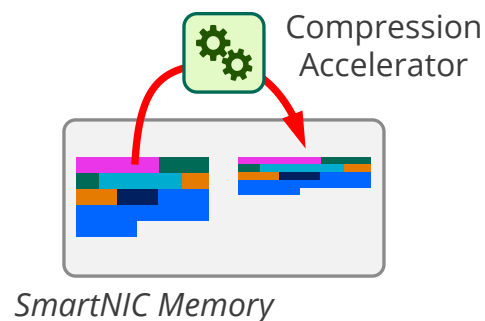
SmartNIC Memory

Scott Levy & Whit Schonbein

Leveraging high-performance data transfer to offload data management tasks to SmartNICs

Compression

- **Problem:** Compression is expensive
- **Goal:** Dynamic data compression
- Leverage BF2 compression unit
 - DEFLATE standard
 - Block or Streaming
 - 30x faster than host

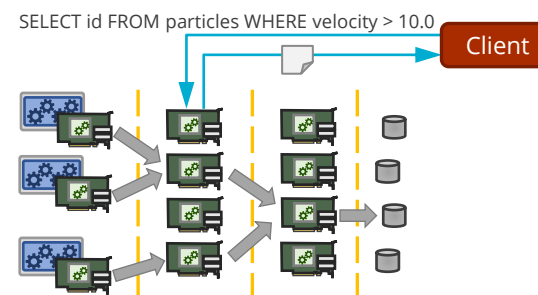


Jianshen Liu

Processing particle data flows with SmartNICs

Dynamic Data Queries

- **Problem:** Poor dataflow visibility
- **Goal:** Dynamic query in-transit data
- Dynamic queries w/ Arrow
 - User generates query plan
 - SmartNIC applies locally
 - Push-down vs Push-back

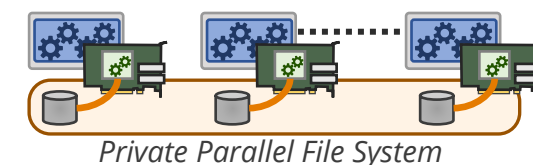


Jianshen Liu

Opportunistic query execution on SmartNICs for analyzing in-transit data

Node-Local Storage

- **Problem:** Platform storage is noisy
- **Goal:** Private, Parallel FS
- Implement PFS in SmartNICs
 - Attach to host's NVMe via NVMe-over Fabric
 - BeeGFS



Matthew L. Curry & John Shawger

Offloading node-local filesystems in high performance computing environments

Summary and Future Work



- SmartNICs offer a new space for hosting data management services
 - Positive: Isolated space for operations near producers, Cheap nodes
 - Negative: Host processors 4x faster, Vendor-specific libraries, extra costs (\$, power)
- Can build a functional environment for hosting services from existing libraries
 - Faodel and Arrow provided primitives we needed
- Opportunities
 - Scheduling use of network link to avoid contention
 - Connect with computational storage devices
 - Low-cost metric collection, resilience

<https://github.com/sandialabs/faodel>

<https://github.com/apache/arrow>





Backup Slides

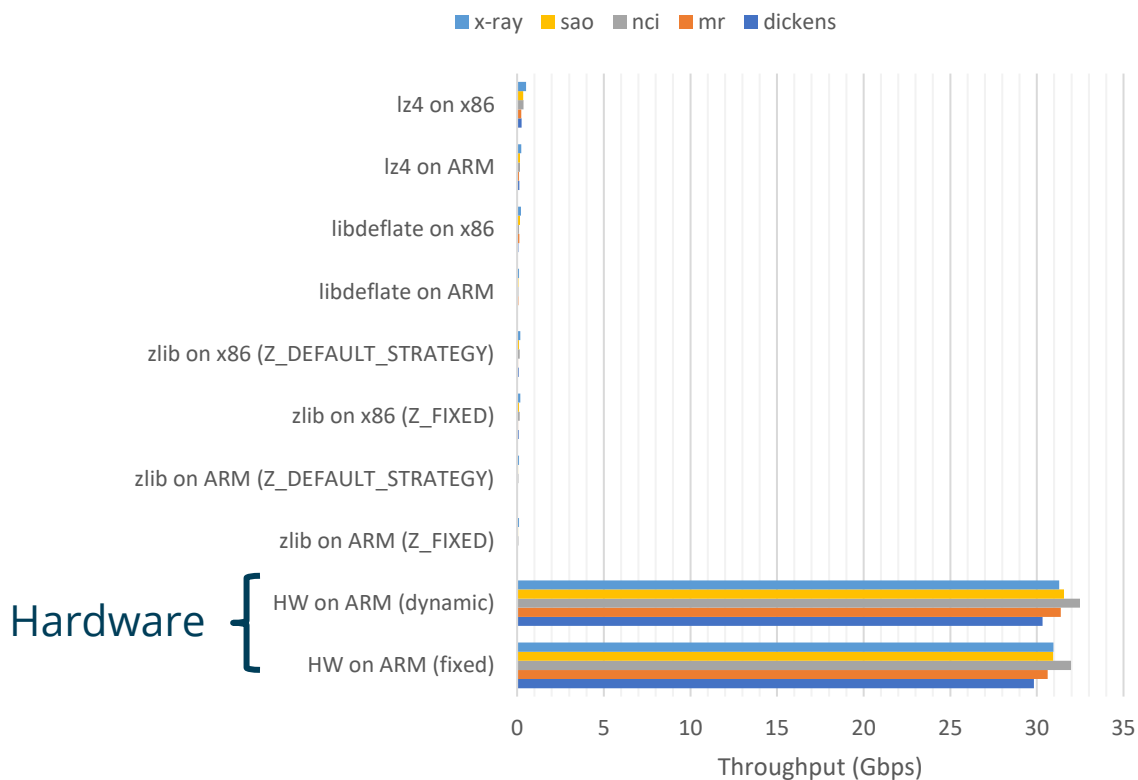


BlueField-2's Compression is Significantly Faster than Host

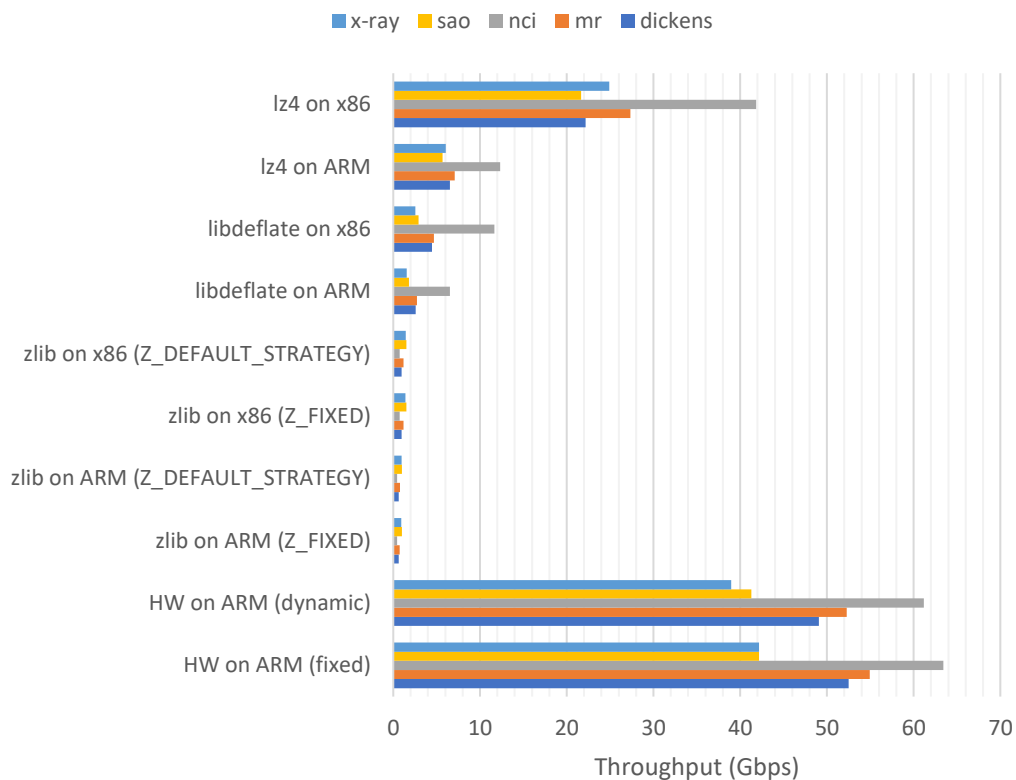


- BlueField-2 features compression hardware for the DEFLATE algorithm (e.g., gzip)

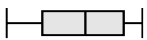


Compression Throughput

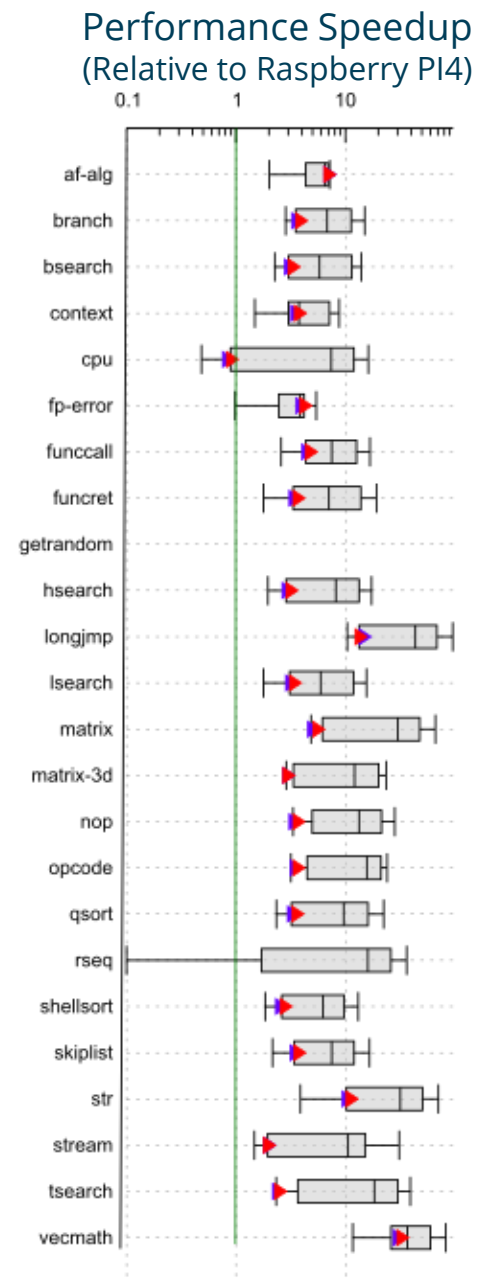


Decompression Throughput



Microbenchmarks

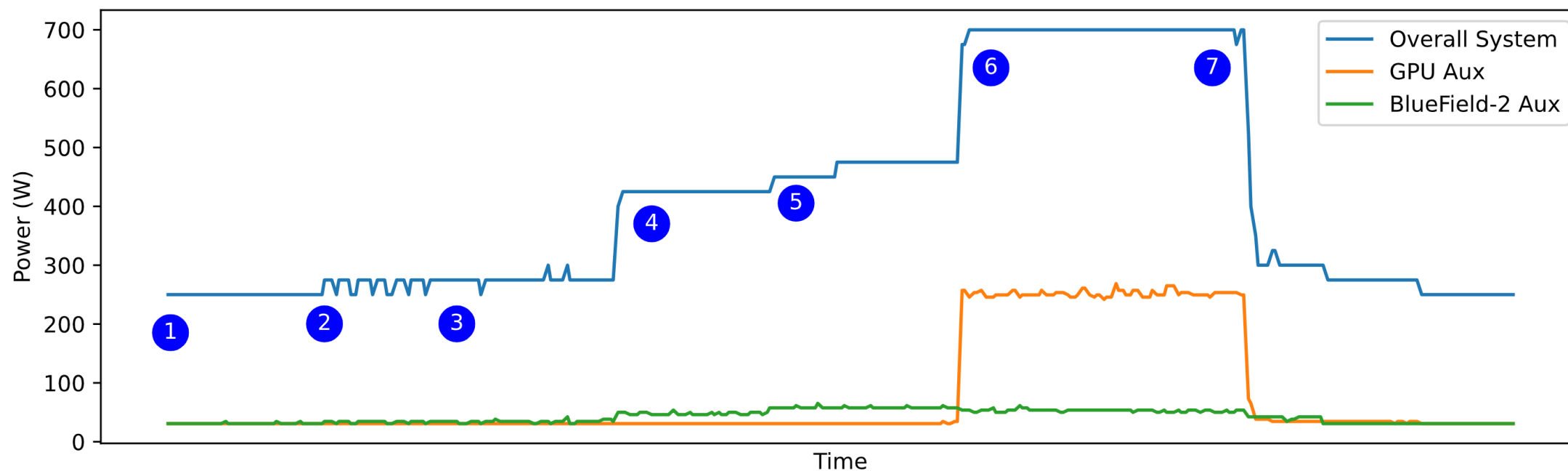
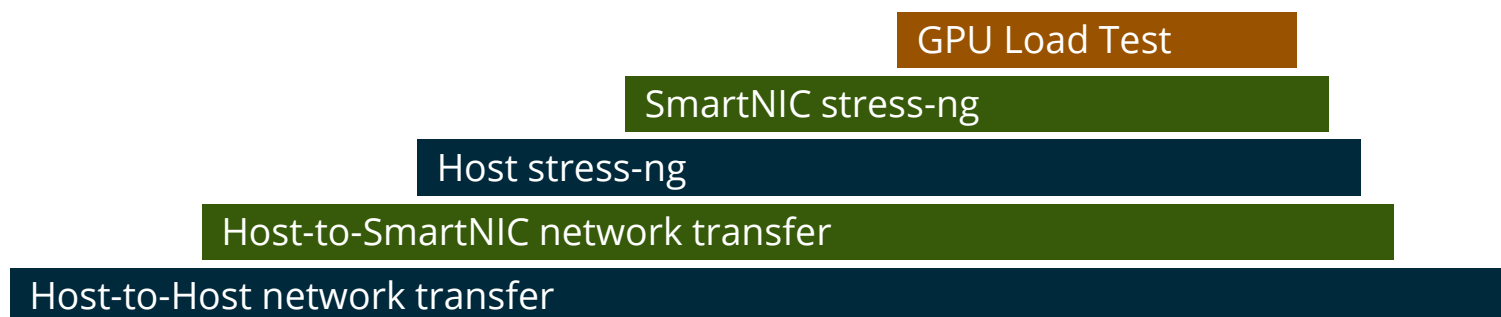
- **stress-ng** from Colin Ian King (Canonical)
 - Suite of 218 stressors for kernel burn-in
 - Normalize results to Raspberry Pi 4B
- Comparison of BlueField-2 to 12 servers
 -  Performance range of different servers
 -  BlueField-2 2.50 GHz (Ethernet)
 -  BlueField-2 2.75 GHz (Ethernet/InfiniBand)
- BlueField-2's Arms order of magnitude slower than hosts
 - *...but still good enough for data management tasks*



Glinda Power Use in Stress Tests



Internal power measurements for a Glinda node while different stress tests are launched on the host, SmartNIC, and GPU.



Evolution of Network Interface Cards



- NIC hardware is often surprisingly simple
 - FIFO queues to move packets between CPU and Wire
- 1990s: Golden Age of programmable NICs
 - Caltech Mosaic → Myricom, Intel IXP, I2O
 - User-programmable CPU at NIC to control messaging
 - Customize to HPC, multicast, collectives, route to PCI devices
- 2000s: InfiniBand, back to (better) hardware queues
- 2020s: Multiple vendors developing SmartNICs
 - Mellanox/NVIDIA BlueField, Intel IPU, Chelsio
 - Processor for **On-Path** or Off-Path processing

