From MPI Semantics to Silicon

A Universal Distributed Compute Fabric for HPC/ MVAPICH User Group (MUG'26)

August 2026

Charles Archer, CTO, Cornelis Networks

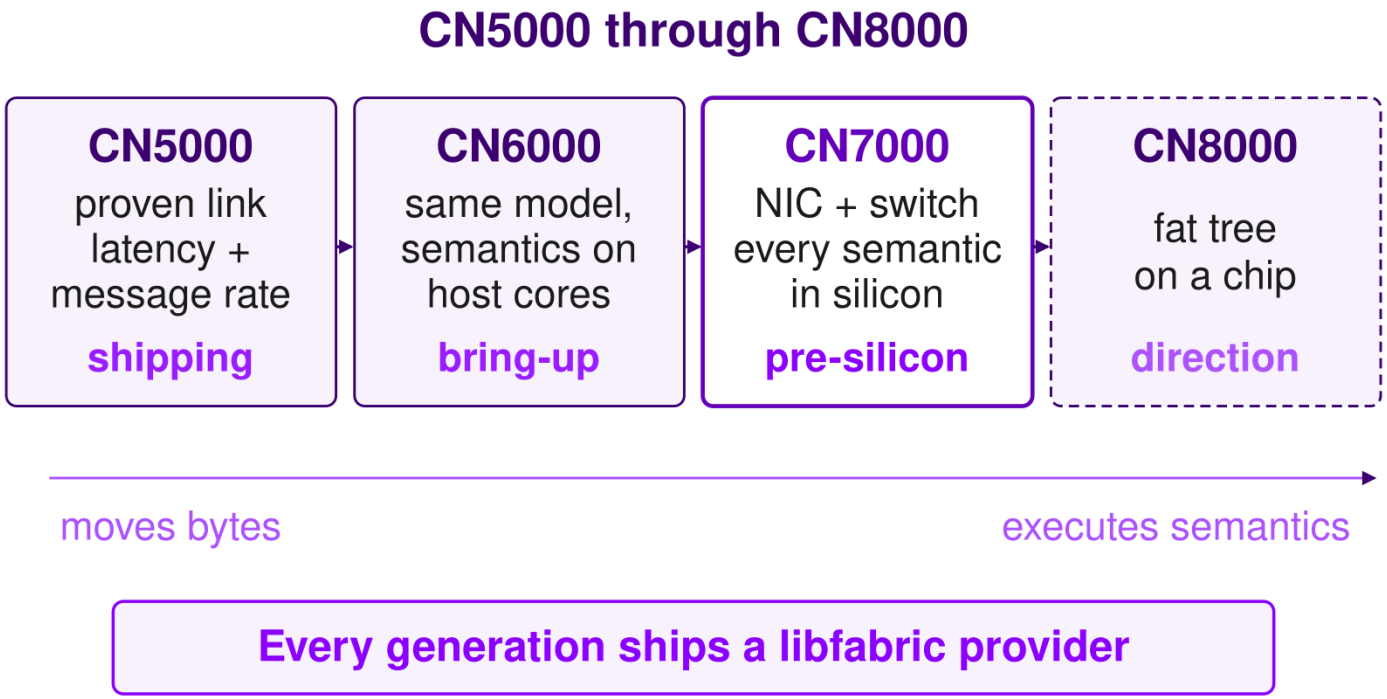
Cornelis Networks' Products

		Speed	Protocols	Key Capabilities
CN7000 2027	**Next-Generation Scale-Out & Scale-Up Network**	1.6T PCIe 7.0	Ultra Ethernet UALink & ESUN	UE Performance Enhancements In-Network Computing
CN6000 2H 2026	**Multi-Protocol SuperNIC** **Omni-Path End-to-End**	800G PCIe 6.0	Omni-Path SuperNIC and Switch RoCEv2 SuperNIC for 3rd Party Switches	RoCE Performance Optimization Ultra Ethernet Compliant PQC Root-of-Trust
CN5000 Shipping now	**Enterprise Class AI and HPC Network**	400G PCIe 5.0	Omni-Path	Adaptive Routing Advanced Congestion & Flow Control

CORNELIS™

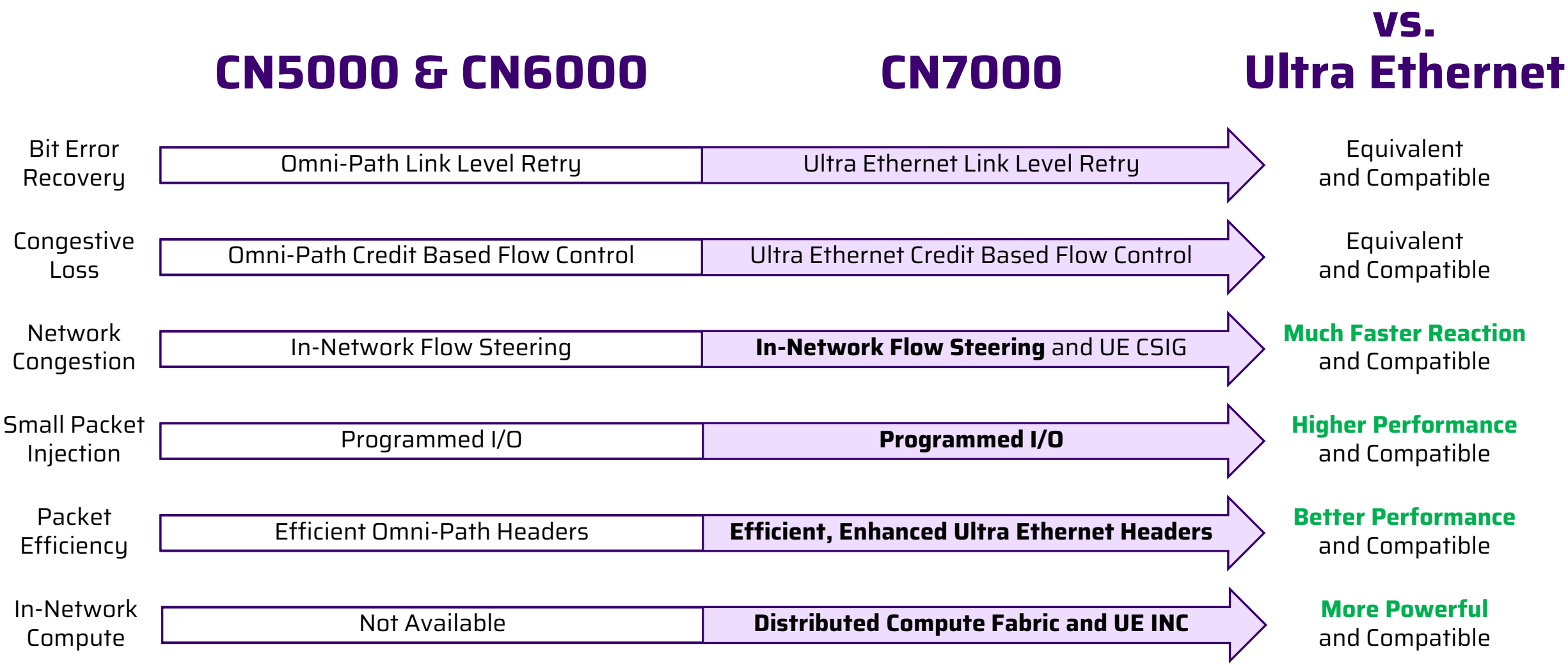
Cornelis Networks Fabric Roadmap

- Intel Omni-Path 100, Cornelis Networks' CN5000 and CN6000 are Omni-Path architecture: Focused on **reliable, efficient, link architecture**.
  - Credit Based Flow Control, Forward Error Correction, Link Level Replay, Pre-emption, Flit
  - Inherited from Intel, QLogic and Cray Technology
- CN7000 — extends the efficient link: **Accelerates application semantics**
- CN8000 extends the line with a leaf/spine hierarchical chiplet design and adds additional acceleration and physical link capabilities



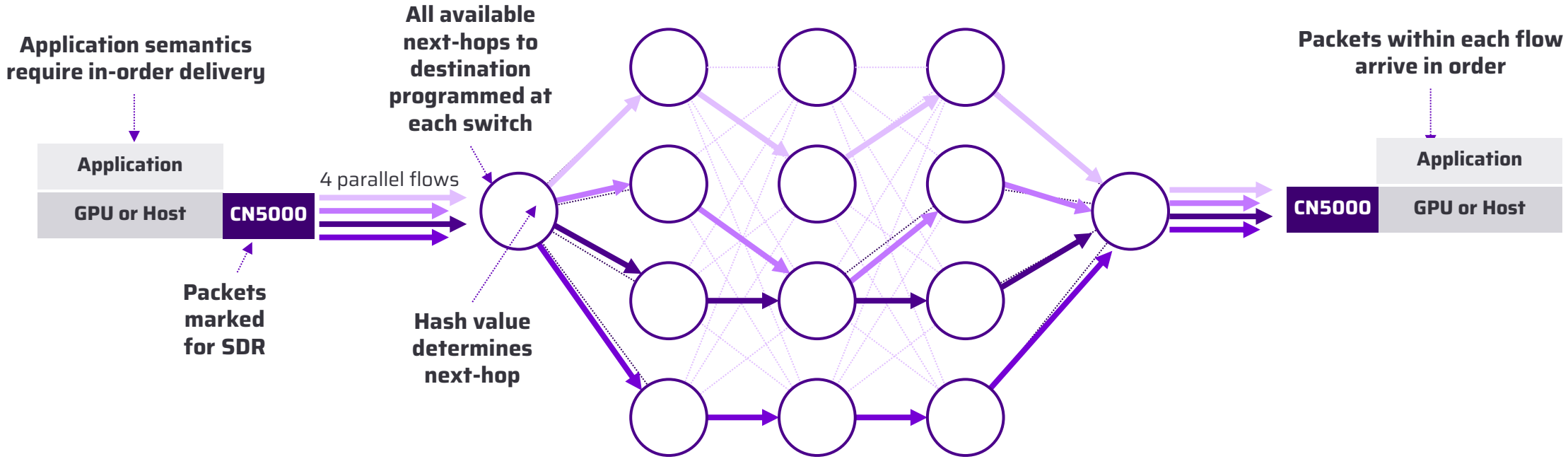
CORNELIS™

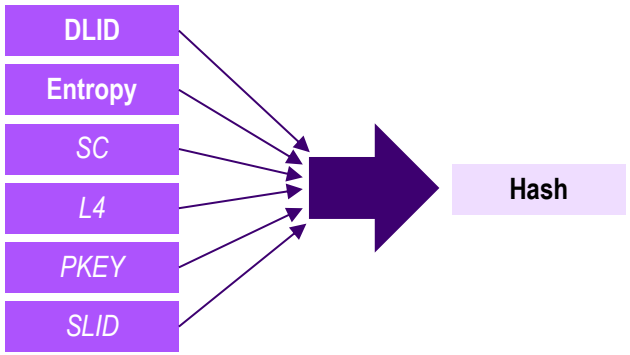
CN7000: Omni-Path to Ultra Ethernet Link



CORNELIS™

Static Dispersive Routing (SDR)





Programmable Hash Function

DLID – Destination LID
Entropy – **Software-controlled** field to identify related packets
(e.g. flows with ordering requirements)

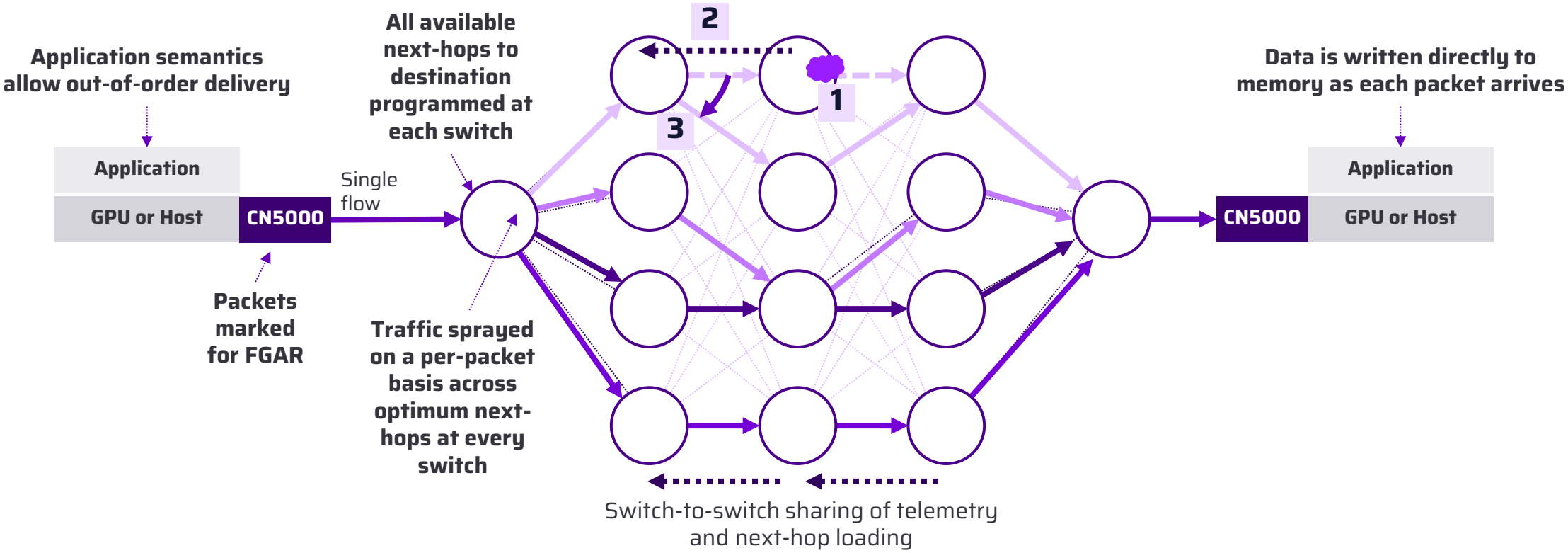
Optional

SC – Service Channel, combination of Virtual Lane and Traffic Class
L4 – Transport Mode
PKEY – Partition Key
SLID – Source LID

CORNELIS™

Fine-Grained Adaptive Routing (FGAR)

1. Heavy load on switch ports
2. Congestion information shared with neighbor switches
3. New set of optimum next-hops selected based on local and remote congestion



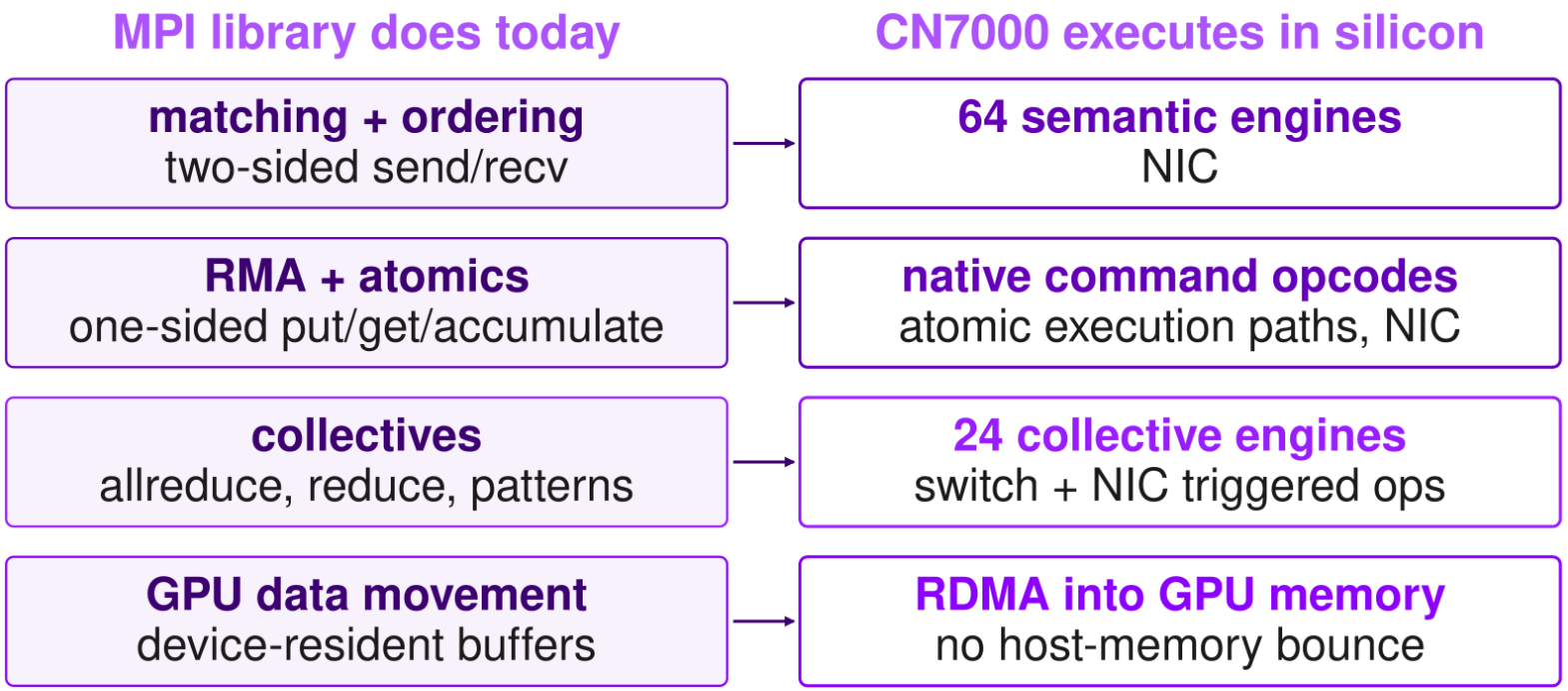
Consistent bandwidth performance for AI and storage applications

CORNELIS™

But how can we accelerate applications?

- Applications include traditional HPC and AI, heavily skewed toward RDMA and Collectives
- We did this comprehensive mapping of MPI and PGAS with **libfabric**! Ultra Ethernet has chosen this as its primary API binding
- **Wire protocols** and **link architecture** deliver the solid foundation.
- **Programming model** and **application mapping** delivers the efficiency.

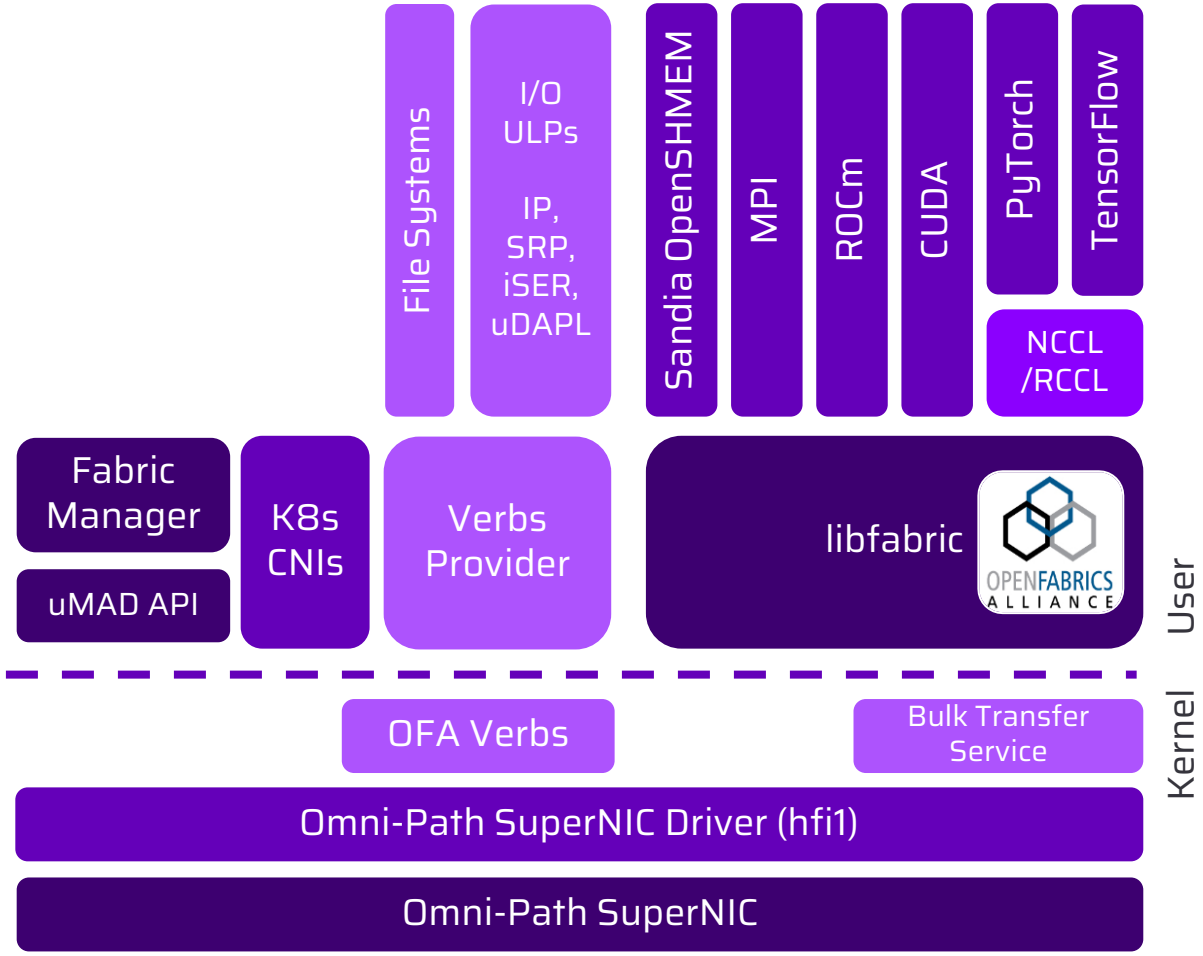
MPI semantics and their hardware addresses



Host-core work moves into fabric silicon.

CORNELIS™

Start with the Application Stack Optimized and Open Software



Optimized Performance

Tight Software-Hardware Integration and Co-Design

Embrace All APIs

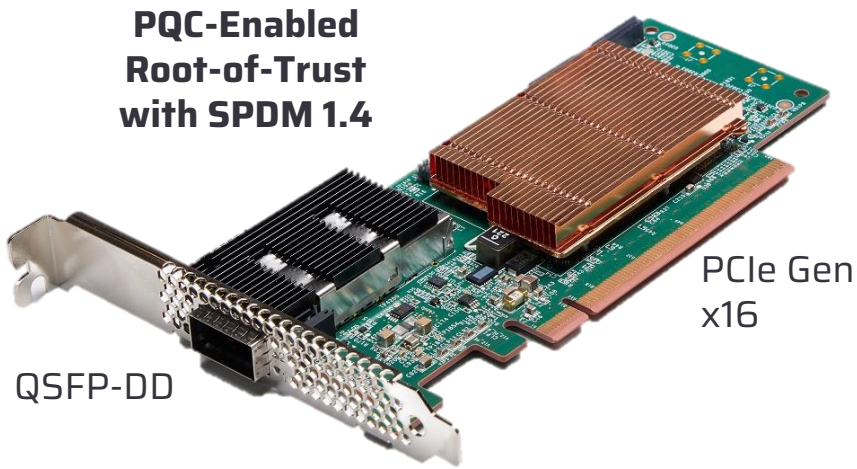
Adoption of Open Standards and Full Stack Support

Trusted Deployment

Open-Source Libfabric Provider and Upstreamed Kernel Driver

CORNELIS™

First, a diversion: CN6000 800 Gbps SuperNIC



Omni-Path
Switch Fabric

3rd Party
Ethernet Fabric

Omni-Path Architecture for Ethernet

1.6 Billion Messages per Second*

Omni-Path, RoCEv2, and UET

Independently Configurable Ports

Same Software Stack Support

Open Ecosystem Embracing All Software Stacks

* 8B bidirectional

CORNELIS™

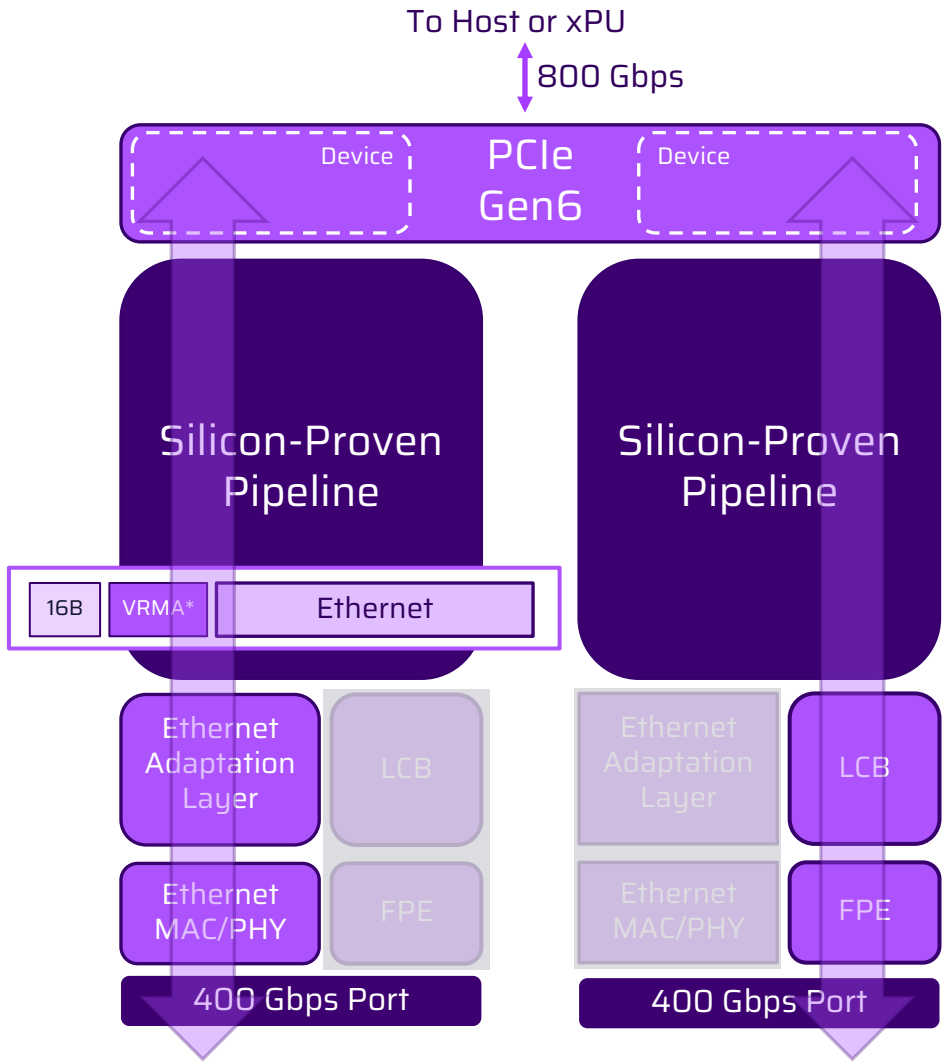
CN6000 SuperNIC Overview

Dual Independent 400 Gbps Ports

- PCIe Gen6 x16 Host Interface
- Production-Proven Pipeline Design

Ethernet Adaptation Layer

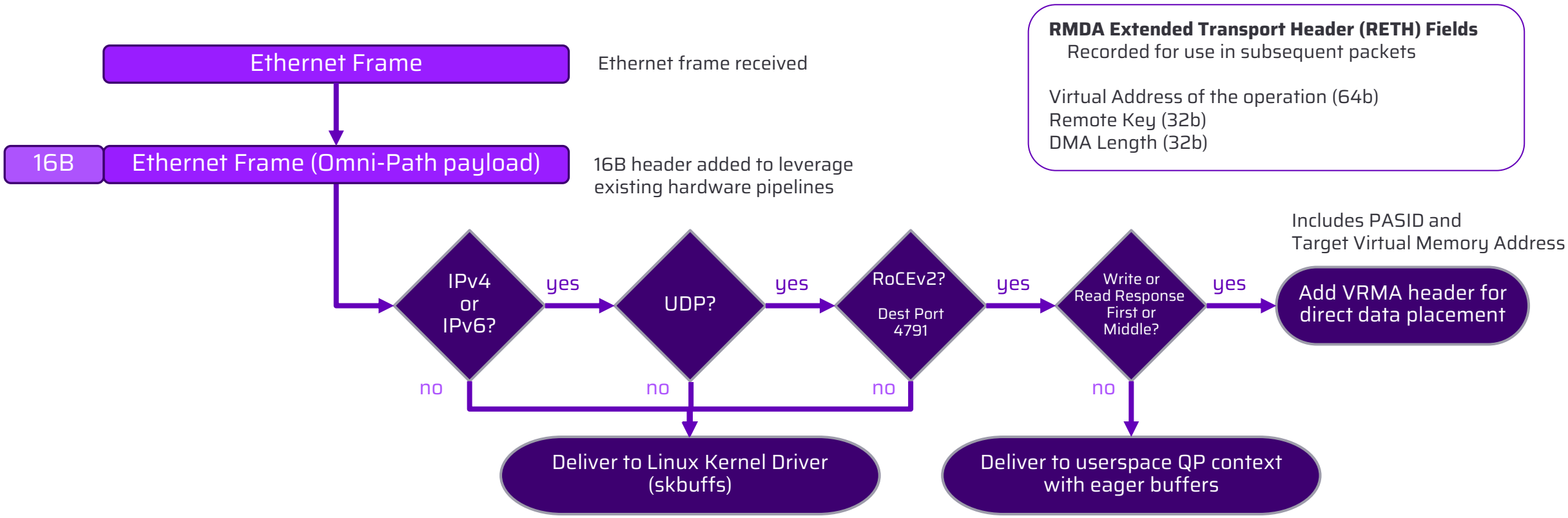
- Leverages Existing Omni-Path Acceleration Features
  - Increased number of hardware pipelines (contexts)
- Hardware-accelerated RDMA reads and writes
- Each port independently configurable between Ethernet and Omni-Path
- Ultra Ethernet software stack

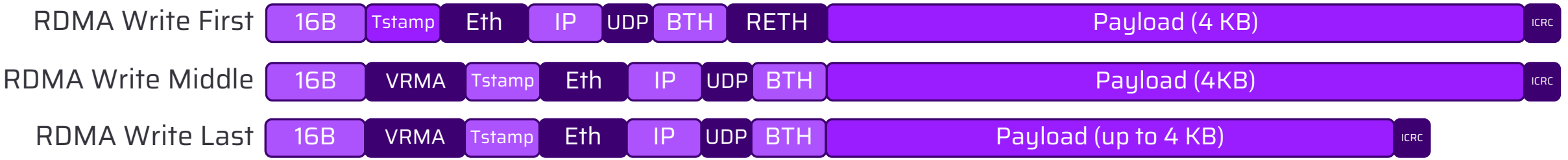


*Virtual Remote Memory Access

CORNELIS™

CN6000 RoCEv2 Hardware Acceleration

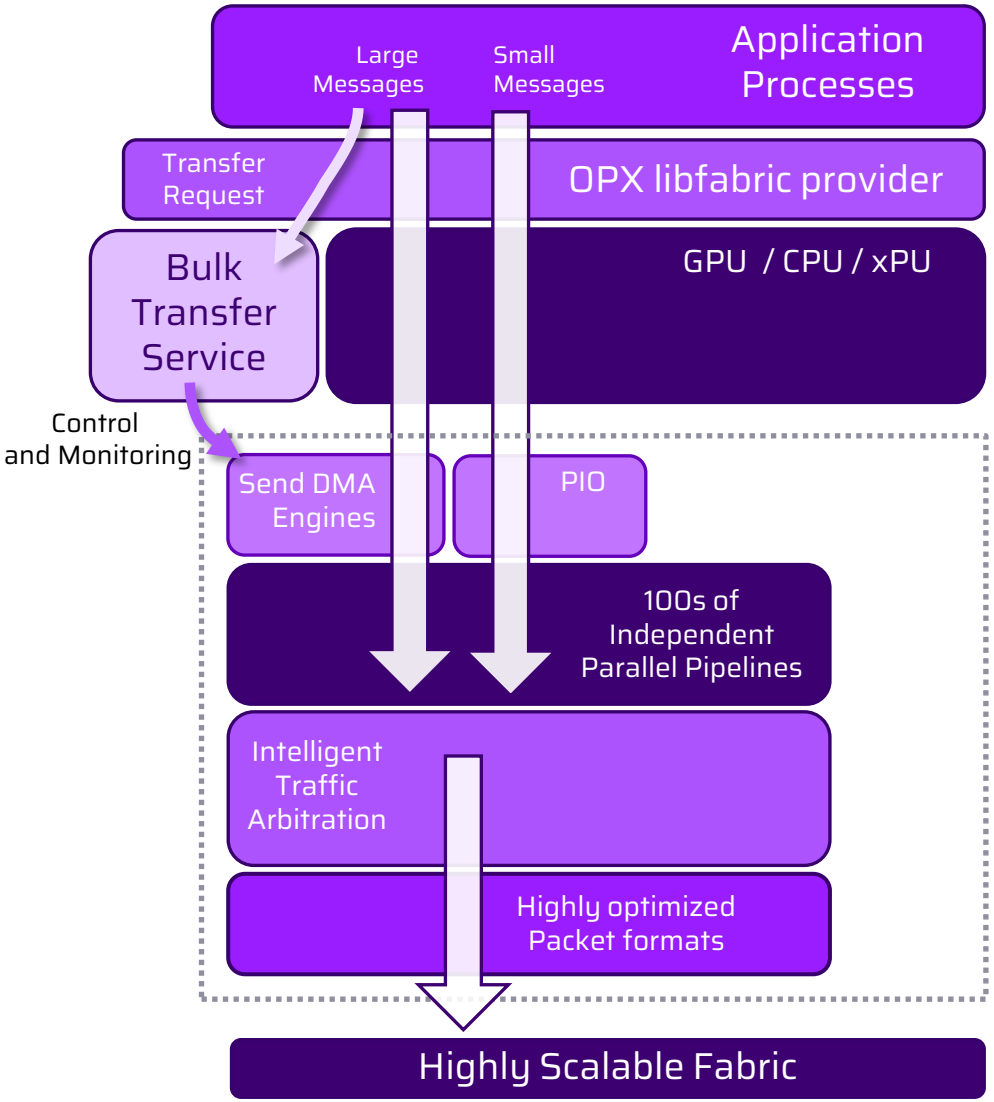




CORNELIS™

Software Architecture: Bulk Transfer Service

- Engage your DMAs! Just memory copy for me!
- Simple non-QP based DMA API (looks very similar to libfabric)
- Kernel bypass for all dma operations
- **Who is this for? You just said libfabric was perfect!**
- It's for libfabric and verbs implementors that require some kernel interactions, say with dma-bufs or memory region optimizations.
- It's almost 1-1 with libfabric rma apis. but has a kernel component.
- It provides an asynchronous ring buffer API with completions
- The ring buffer APIs are implemented in hardware for CN7000
- Library writers (libfabric, verbs, MPI direct) can write to CN5000 and CN6000 bulk service and won't see much difference when writing directly to CN7000 hardware.
- An analog could be MRC to Ultra Ethernet: **We just want something simple**.
- This is our framework today for accelerated offloads
- Old idea that worked well on BlueGene systems (good overlap and bandwidth, low CPU utilization)



CORNELIS™

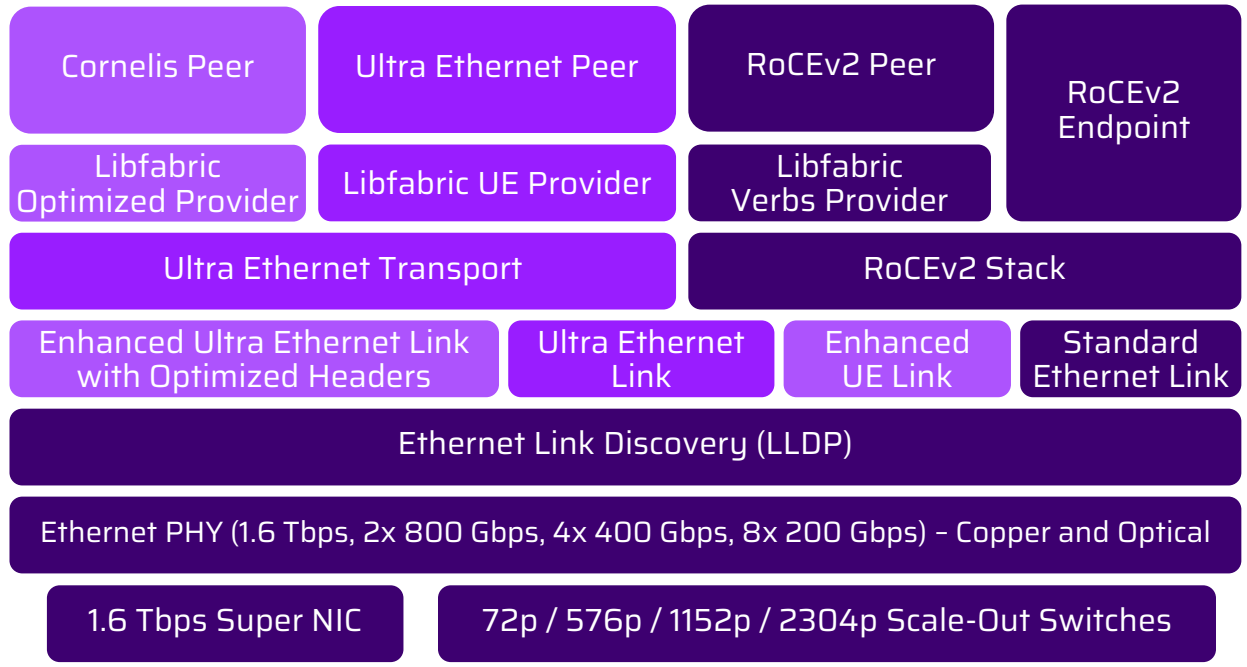
CN7000: Focused Performance for Scale-Out and Scale-Up

Universal Scale-Out Networking

GPUs, TPUs, FPGAs, Accelerators, CPUs,
Storage Systems, Quantum Processors, etc.

Fastest Time-to-Results
For AI and HPC

Widest Interoperability



Scale-Up Networks

GPUs, TPUs, FPGAs,
Accelerators, Quantum Processors

UALink / ESUN

UALink and Ethernet PHYs

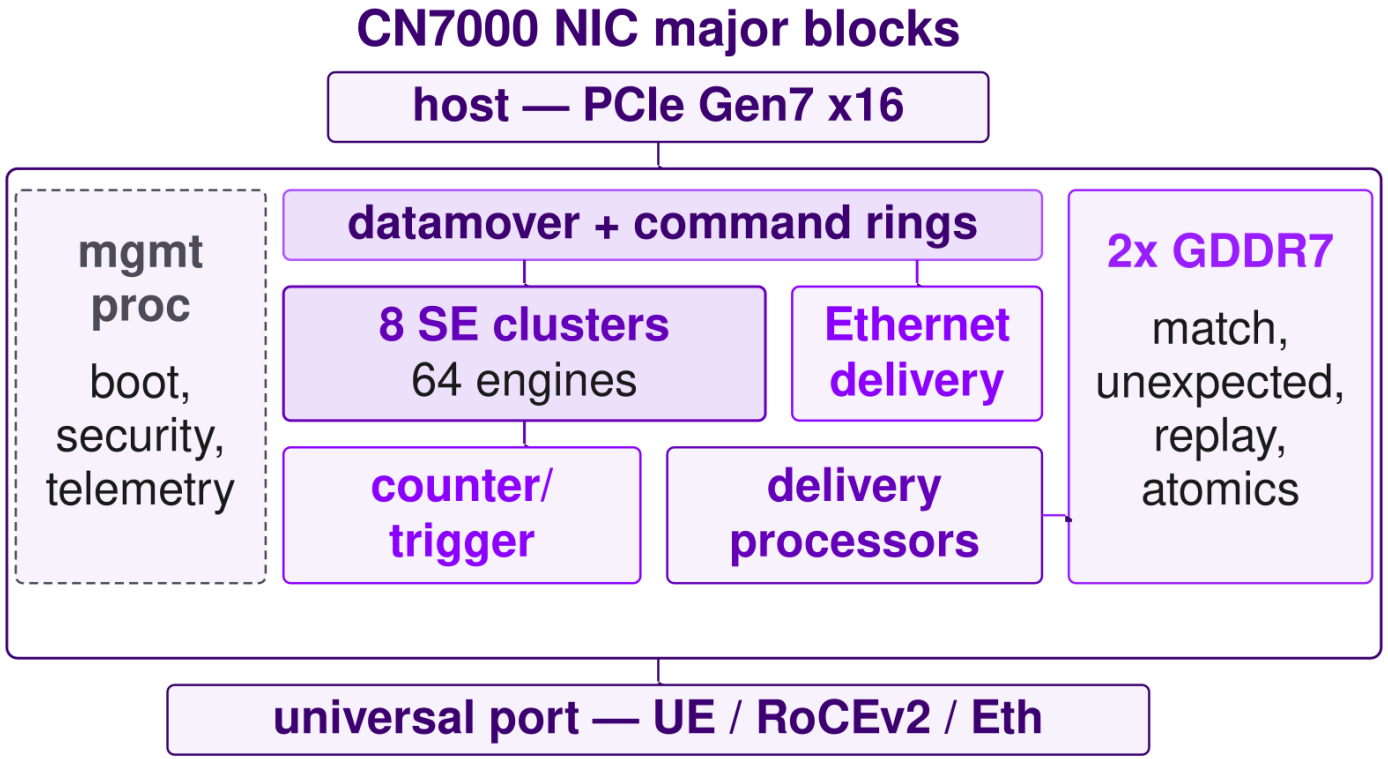
576p / 288p / 144p x 200 Gbps Scale-up Switches

- Enhanced Ultra Ethernet, Ultra Ethernet and Ethernet
- Distributed In-Network Compute in NIC and Switch
- 1.6 Tbps SuperNIC
- 72x 1.6 Tbps scale-out switch
  - 1x 1.6 Tbps, 2x 800 Gbps, 4x 400G Gbps, 8x 200 Gbps
- 576p / 1152p / 2304p Director Class Switches
- 576p / 288p / 144p x 200 Gbps scale-up switches
  - UALink with In-Network Compute

CORNELIS™

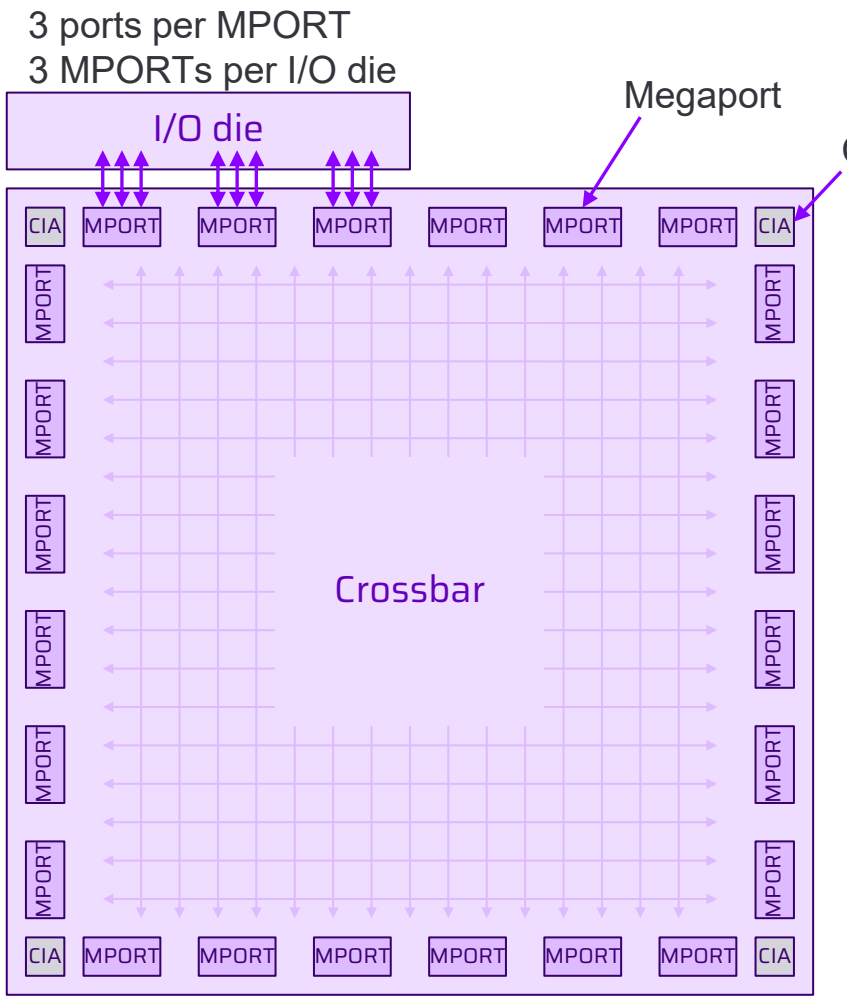
From MPI Semantics to Silicon

- CN7000 does not implement MPI *directly* in silicon. But we are getting closer. Hardware acceleration routes are via:
  - **Verbs (RoCE or MRC):** Simple apis with nasty ordering and wire protocol semantics
  - **Libfabric/UCX:** More complex, but much richer feature set and better application semantic match
  - **The switch:** provides the efficient delivery (packet spray, adaptive routing, collective acceleration)
  - Together, the NIC and switch implement **a premier libfabric accelerated fabric**.
- HPC mapping onto libfabric— MPI, SHMEM, *CCL, and PGAS map cleanly. HFI Service == CN7000 command ring
- Ultra Ethernet adopted libfabric as its primary interface, so accelerating libfabric accelerates the UE transport directly.



CORNELIS™

CN7000 Switch Architecture



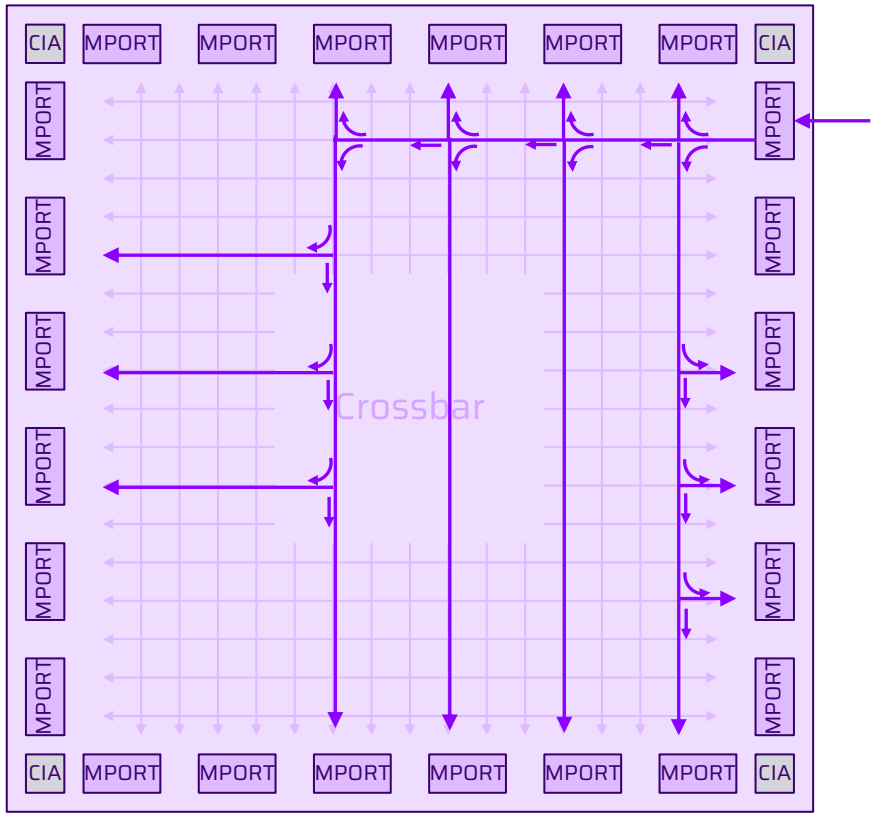
Fan-out/converge trees implement the crossbar
- Not a mesh

Megaports: groups of 3 ports
- Each 8 x 200Gbps
- 6/12/24 UALink links
- 3 Megaports attach to an I/O die (one shown)
- Each megaport
  - Implements receive buffering
  - Implements routing decisions
  - **Contains a Collective Engine (CE)**

Crossbar Input Arbiter (CIA)
- MPORTs consult the CIA for permission to send
- The CIA coordinates data flows for collectives
- The 4 CIAs work together. Proximity reduces latency

CORNELIS™

Multicast



Multicast is an inherent switch feature

- Duplicate at “turns” in the trees
- Works with multi-tier switching
- Same latency to all ports at once as unicast

Two steps

1. Credit grant within switch
2. Send packet with replication sideband

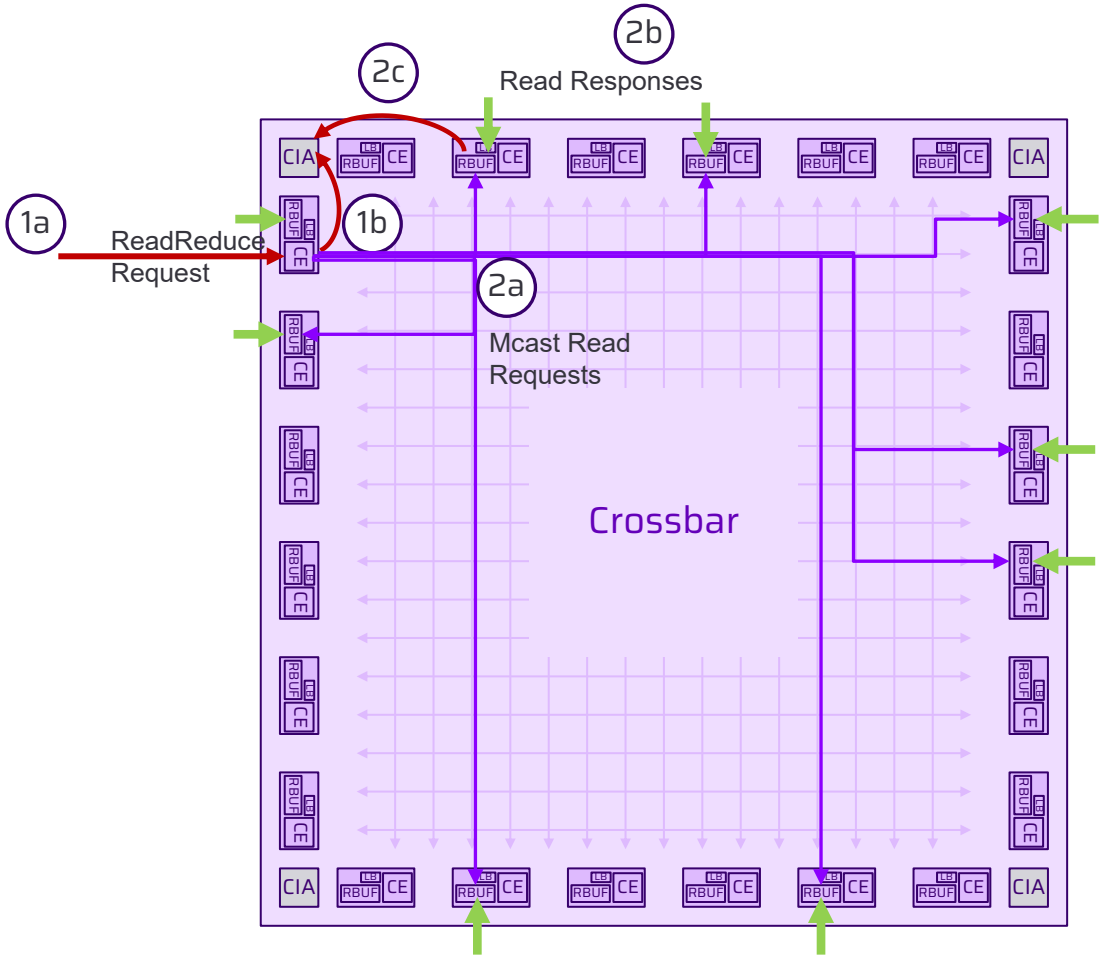
Multicast is broken into sub-steps if all credits cannot grant simultaneously

UALink

- Requests can be multicast
- WriteMulticast, Read for ReadReduce

CORNELIS™

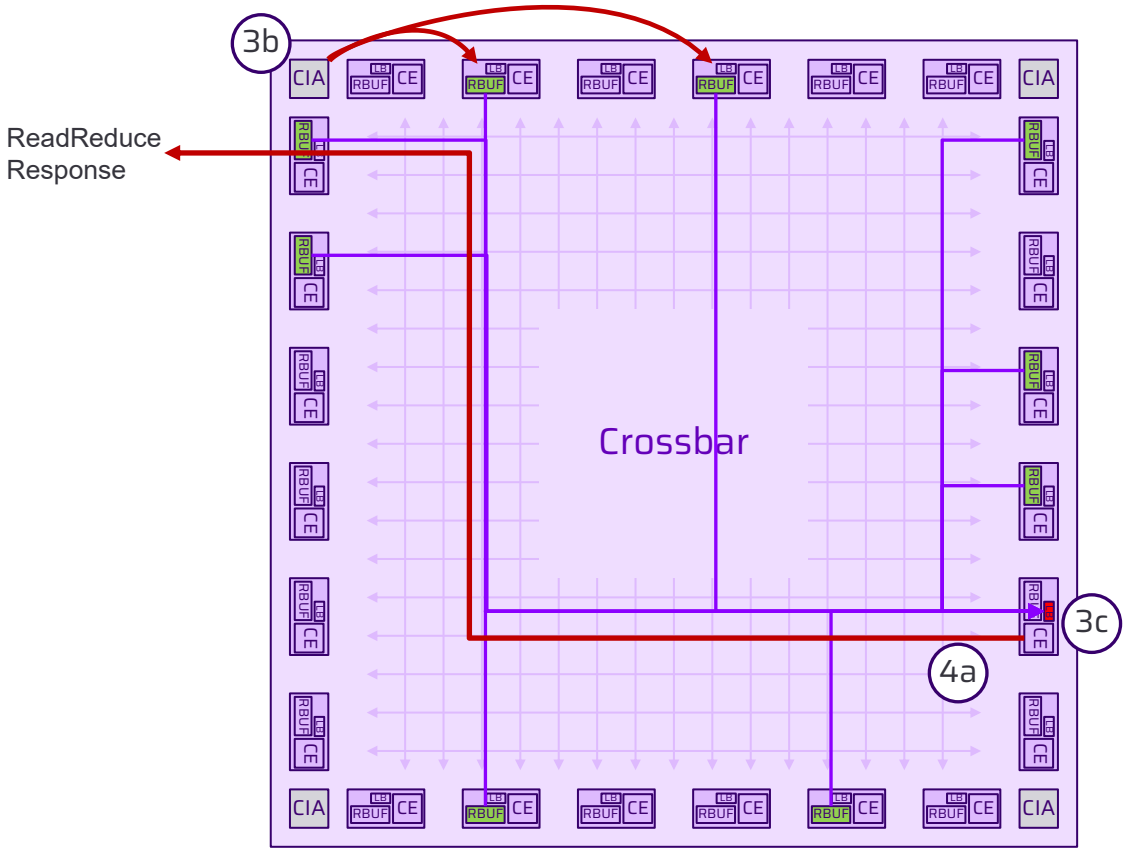
UALink: INC ReadReduce



1. Initiation
   a) ReadReduce is forwarded to a CE in the local MPORT
   b) CE requests a collective op from the CIA
   c) CIA grants collective-id back to CE
2. Data Pull
   a) CE multicasts a READ to the group (includes collective-id)
   b) GPUs return Read responses (RTT can be > 1us)
   c) Rpipe informs CIA of responses (one shown) Responses are held in RBUF waiting for grant
3. Data Push and Compute
4. Response

CORNELIS™

UALink: ReadReduce Steps (cont.)



1. Initiation
2. Data Pull
3. Data Push and Compute
   a) CIA counts all Read data is in RBUF and selects target CE (not shown)
   b) Issues grants in order to move data to CE LBUF
   c) CE fires event to reduce data
4. Response
   a) Reduced data forms ReadReduce response sent back to initiator

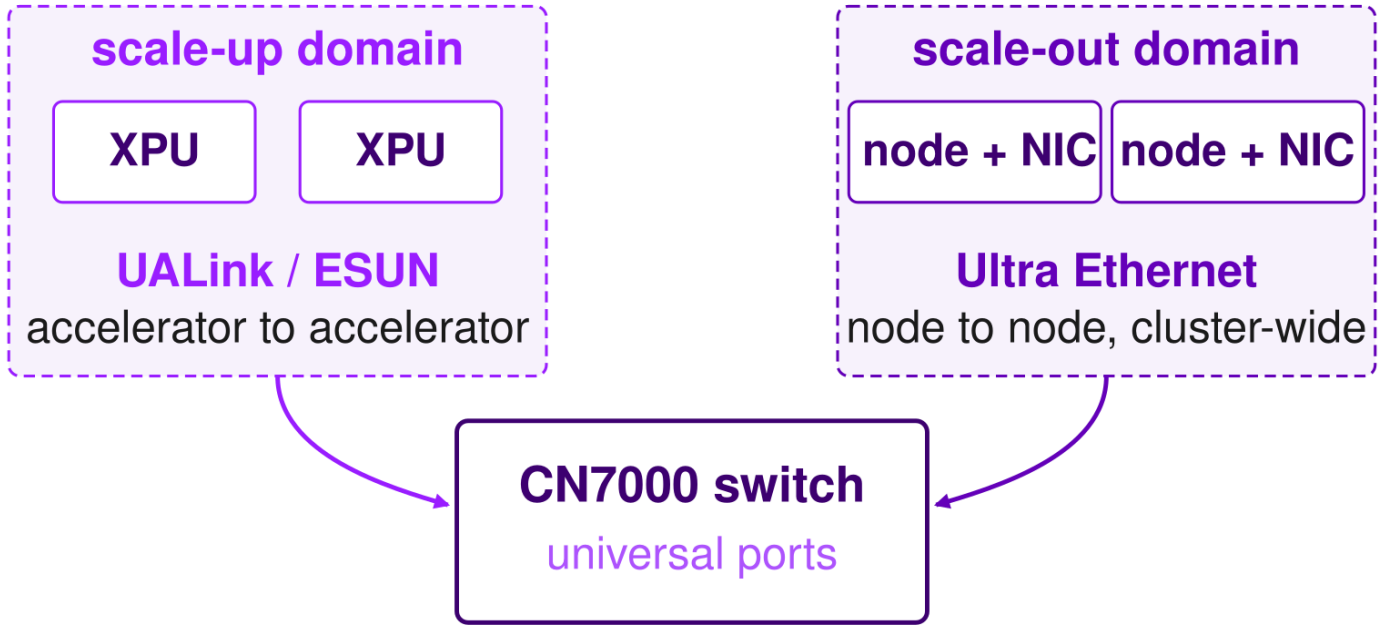
But why are you showing me UALink collectives at an MPI user group?

CORNELIS™

The Same Switch Builds Scale-Up and Scale-Out Fabrics

- A cluster runs two independent fabrics: a **scale-up fabric** (accelerator-to-accelerator UALink/ESUN) and a **scale-out fabric** (node-to-node Ultra Ethernet), connected through the nodes.
- The same CN7000 switch design builds both, port by port — one switch part, one management model, instead of a different vendor and stack per fabric.

Two independent fabrics, built from the same switch

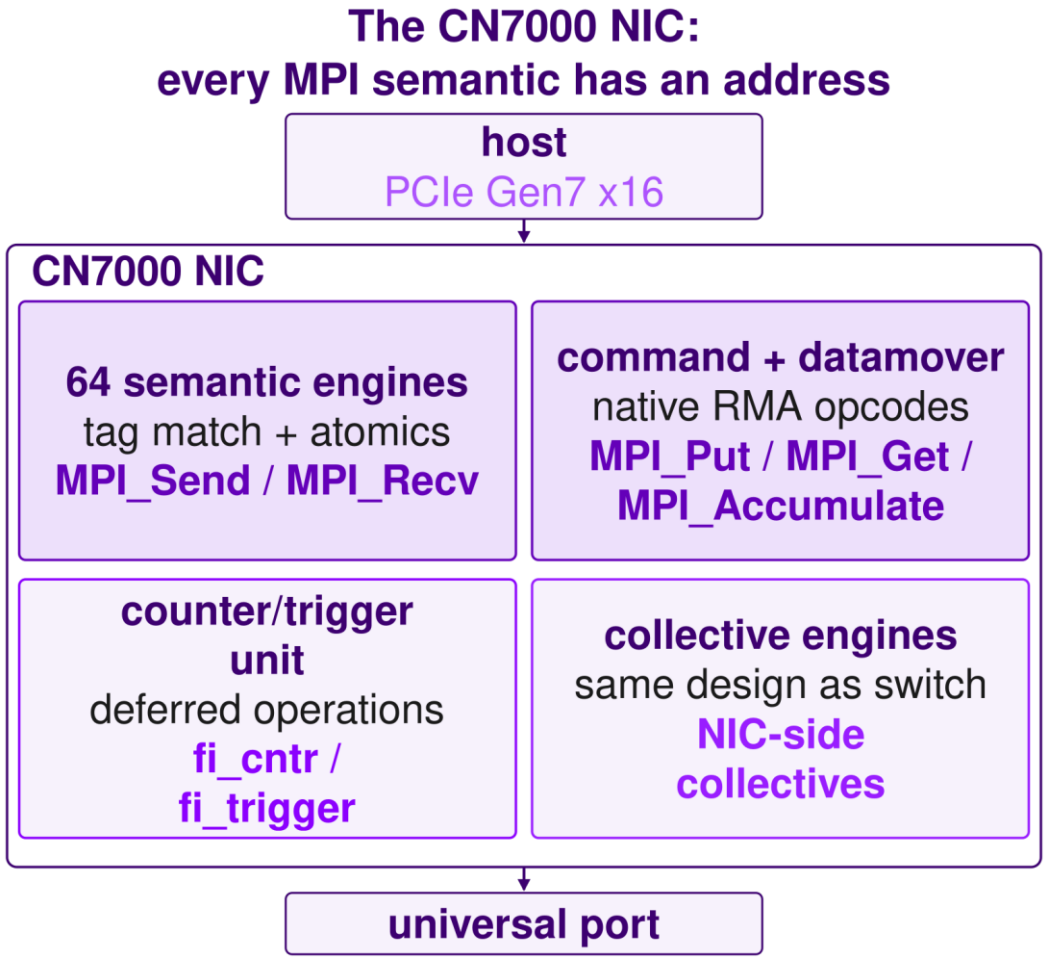


The same switch part serves both fabrics.

CORNELIS™

CN7000 NIC: Where Each MPI Operation Executes

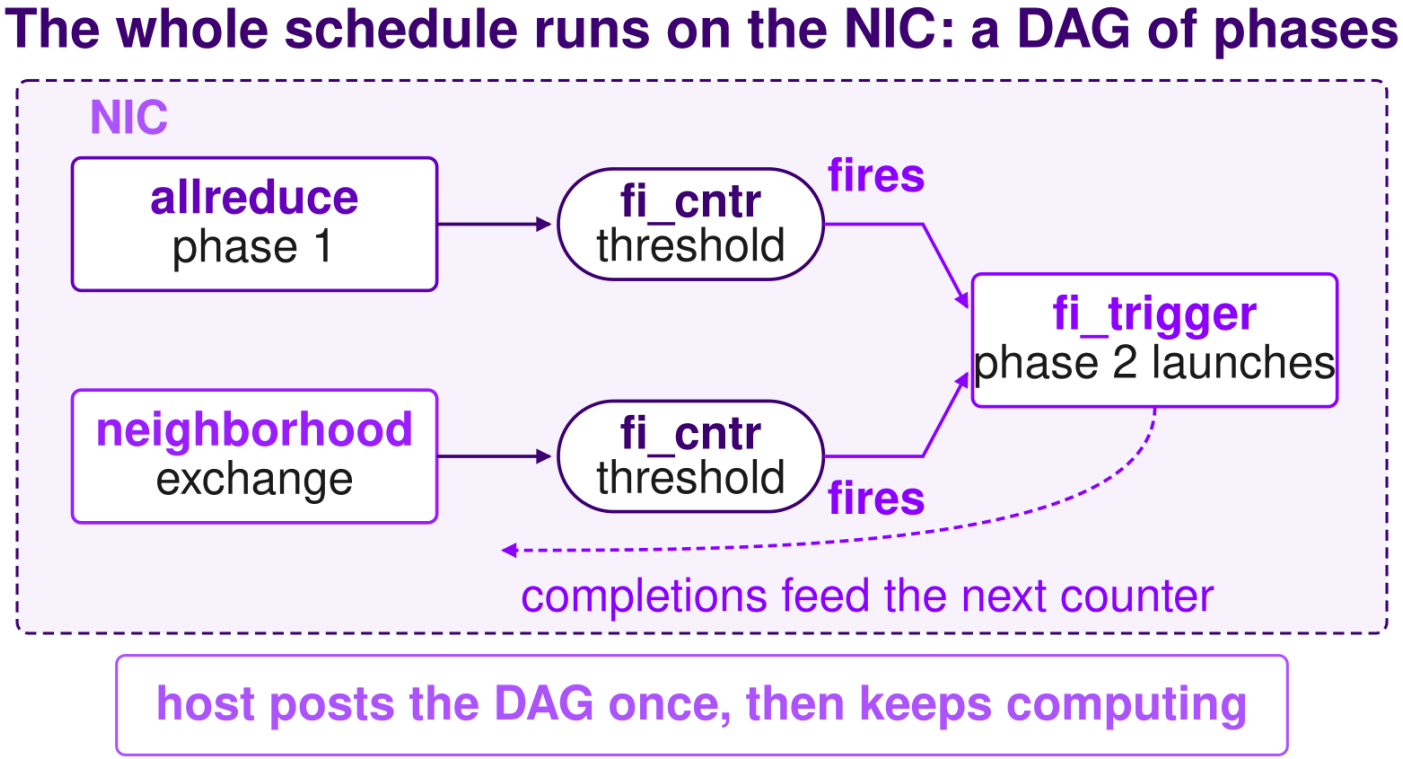
- **Commands enter on the libfabric command ring:**
  - User space writes descriptors into a per-endpoint command queue the NIC consumes directly
  - Completions are returned
  - Enough resources can be mapped to remain lock free
- **Every command has a hardware destination:**
  - Tag matching in the semantic engines,
  - RMA and atomics through the command interface and datamover,
  - Triggered dependency chains in the counter/trigger unit,
  - Small collectives to the CTU



CORNELIS™

RMA and Triggered Operations: MPI Without the Host CPU

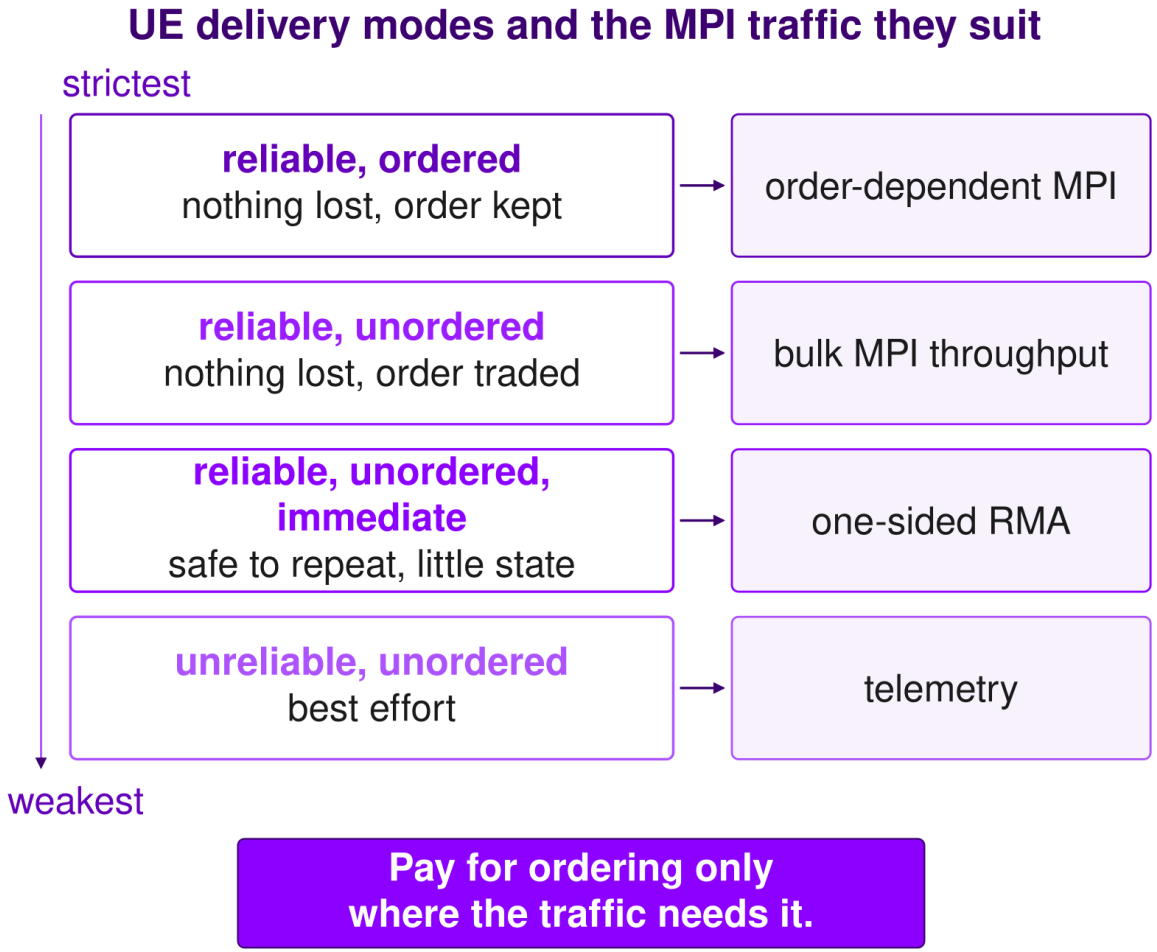
- One-sided MPI is native: RMA and atomic operations are first-class command opcodes with dedicated atomic execution paths, and an on-NIC AMO cache executes atomics against device-resident state — no host memory round trip per update.
- fi_cntr and fi_trigger turn phases into a DAG the NIC executes: one collective phase and a neighborhood exchange both complete, their counters fire, and the next phase launches — the host posts the graph once and keeps computing.



CORNELIS™

Matching UE Delivery Modes to MPI Traffic

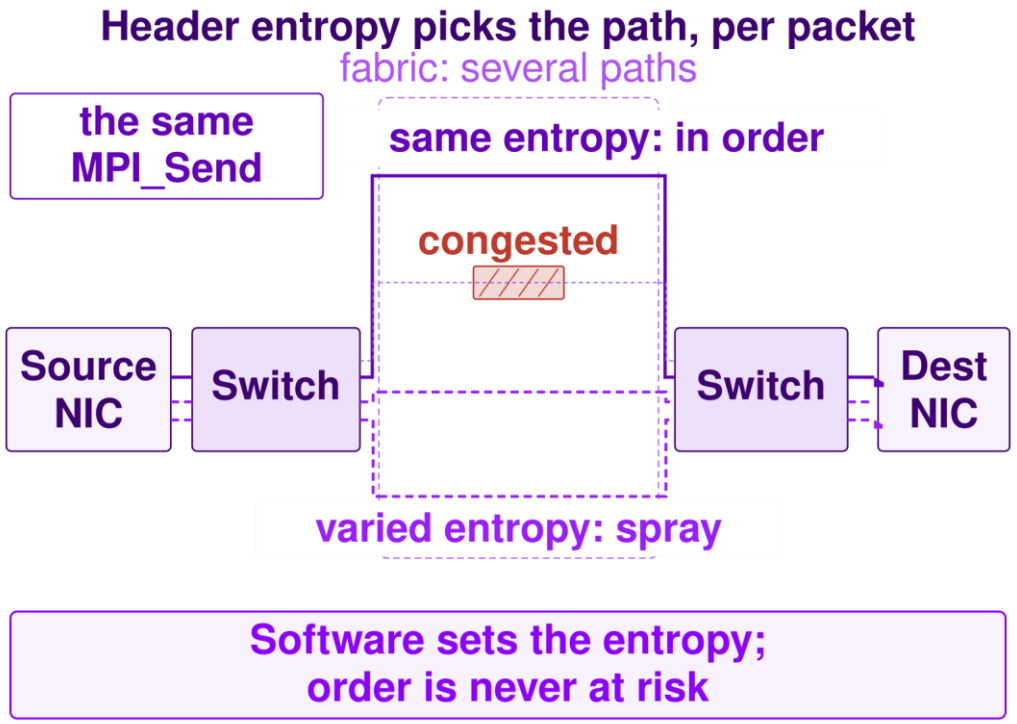
- Ultra Ethernet's layering is the NIC's layering: the semantic sublayer (matching, RMA, atomics) executes in the semantic engines, and the packet delivery sublayer beneath it executes in the delivery processors — the full UE stack in hardware.
- The delivery processors implement all four UE delivery modes; the provider picks the weakest mode each operation tolerates — ordering costs throughput, so paying for it everywhere is the wrong default.



CORNELIS™

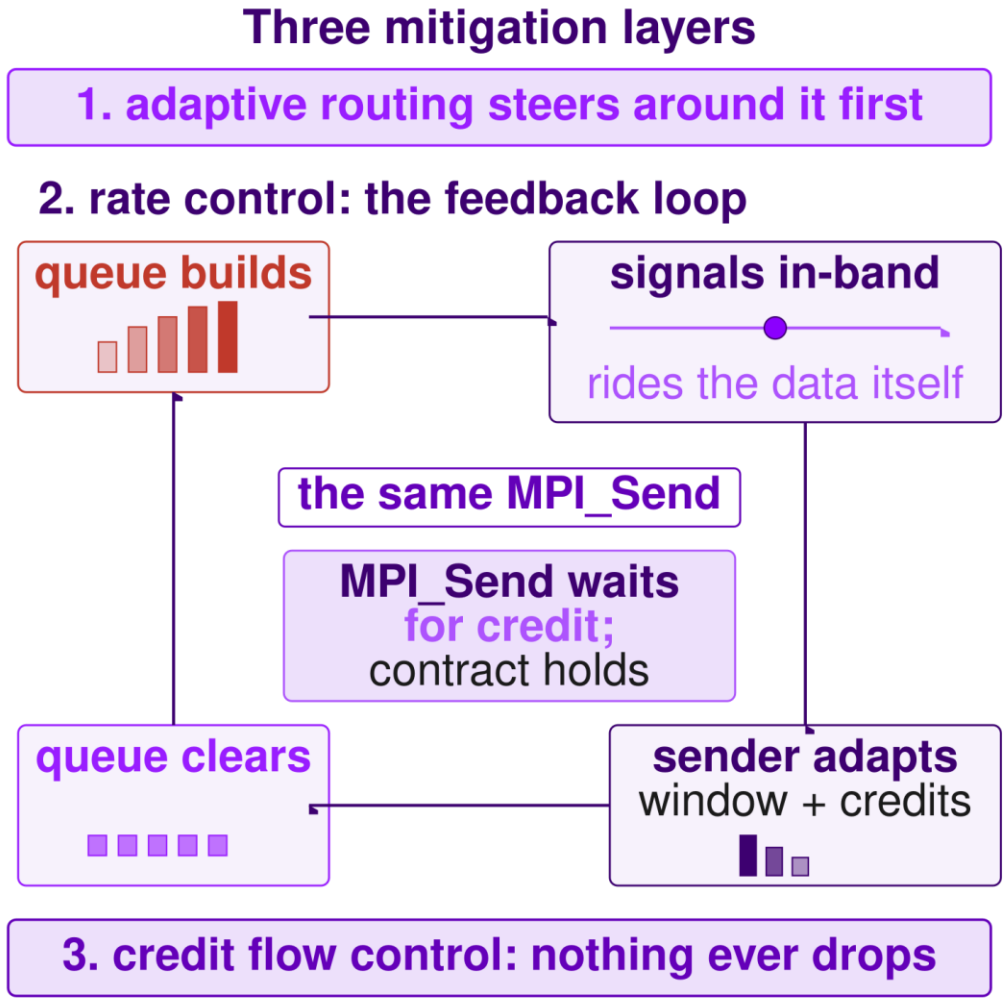
Multiple Paths and relaxing order. Once matched, ooo packet delivery is possible without resequence

- The path is a function of an entropy value software writes into each packet's headers: same entropy, same path, in order — different entropy, path diversity. Software can set it per packet, the programming model carried forward from CN5000/CN6000.
- The default is adaptive: the fabric steers around congestion — the routing lineage Omni-Path proved in production — and MRC adds host-controlled flow steering via SRv6 source routing.



What Happens to MPI_Send When the Fabric Is Congested?

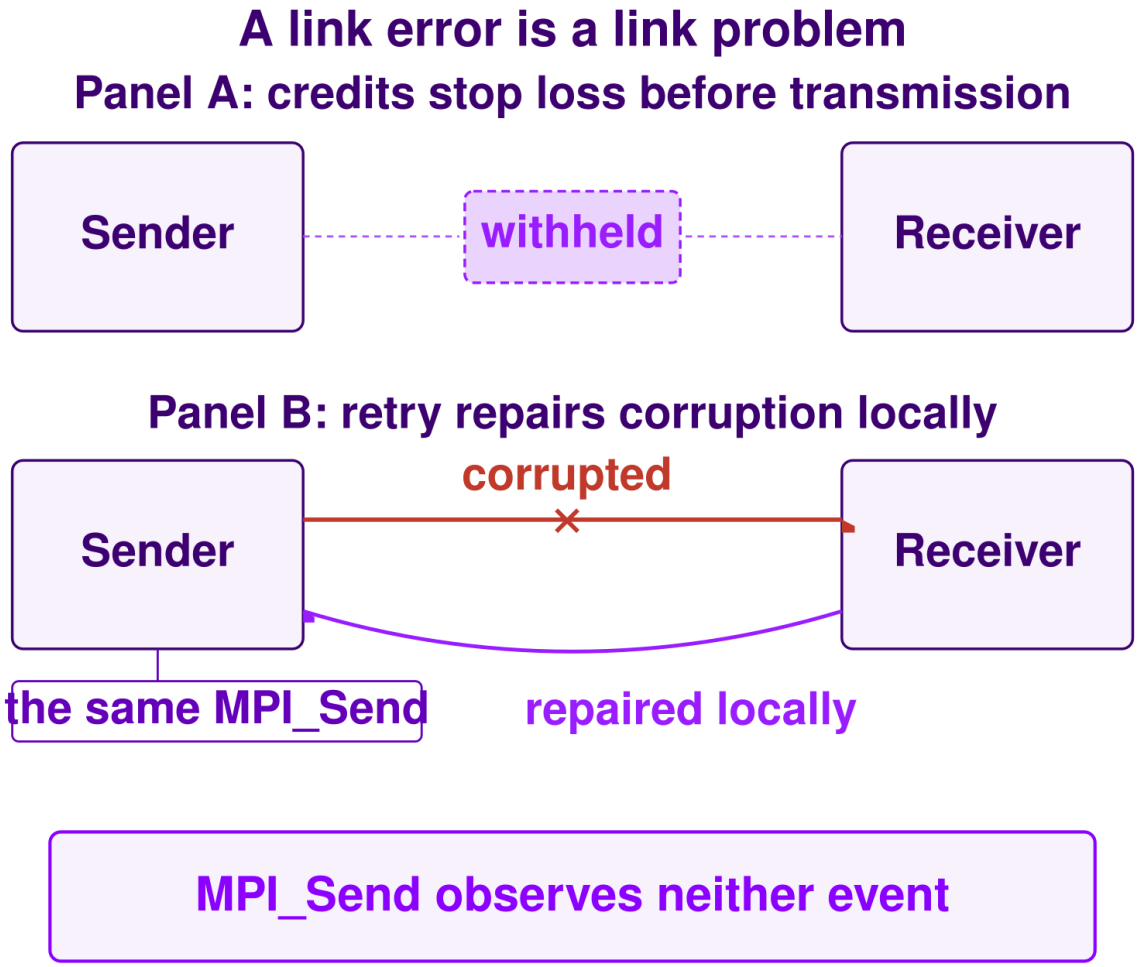
- **Mitigation is layered:** adaptive routing steers new packets around the hotspot, sender rate control (window-based NSCC) and receiver credits (RCCC) shrink what enters it, and credit-based flow control guarantees nothing is dropped while it drains.
- **The switch marks congestion in-band** — ECN single-bit today, standards-track multi-bit in development; MPI_Send may briefly stall, but the delivery mode's guarantees hold.



CORNELIS™

Why a Link Error Does Not Imply End to End replay (long latency tails)

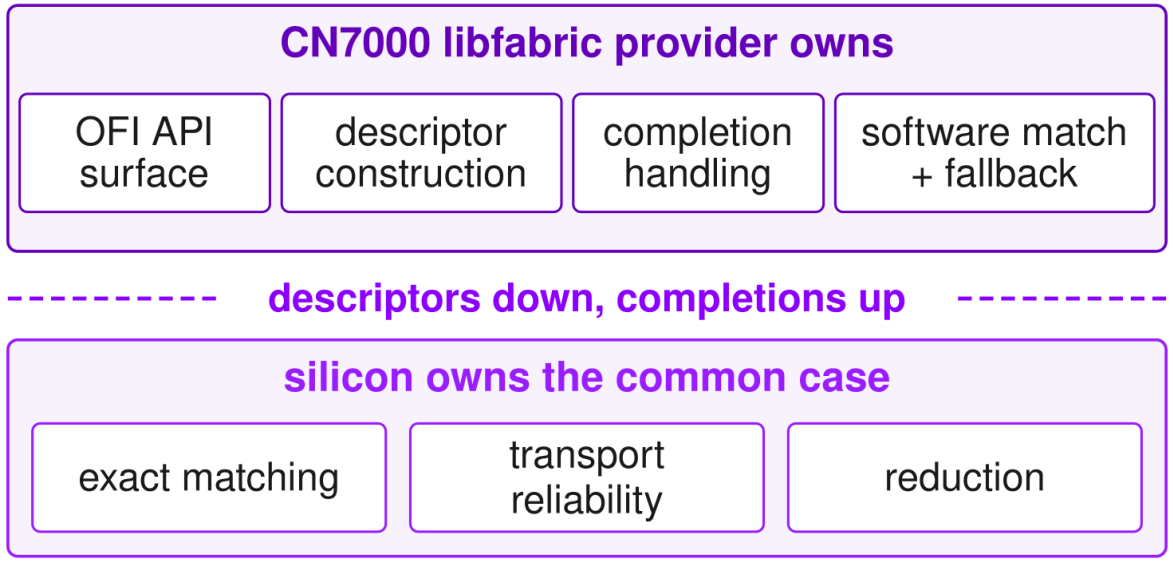
- Three layers of link hardware: FEC corrects most bit errors in place, link-level retry replays the uncorrectable remainder hop by hop, and credit-based flow control prevents loss from overrun in the first place.
- Endpoints never participate — CBFC and LLR are both carried from Omni-Path into Ultra Ethernet.



Is there anything left for software to do?

- The CN7000 libfabric provider implements the Ultra Ethernet transport: API surface, descriptor construction, completion handling, and the software matching and fallback paths already described. **This layer is very thin due to the nice semantic match. The MPI mapping should be thin to libfabric too.**
- Silicon executes the common-case operations — exact matching, transport reliability, and reduction; the provider handles everything else.

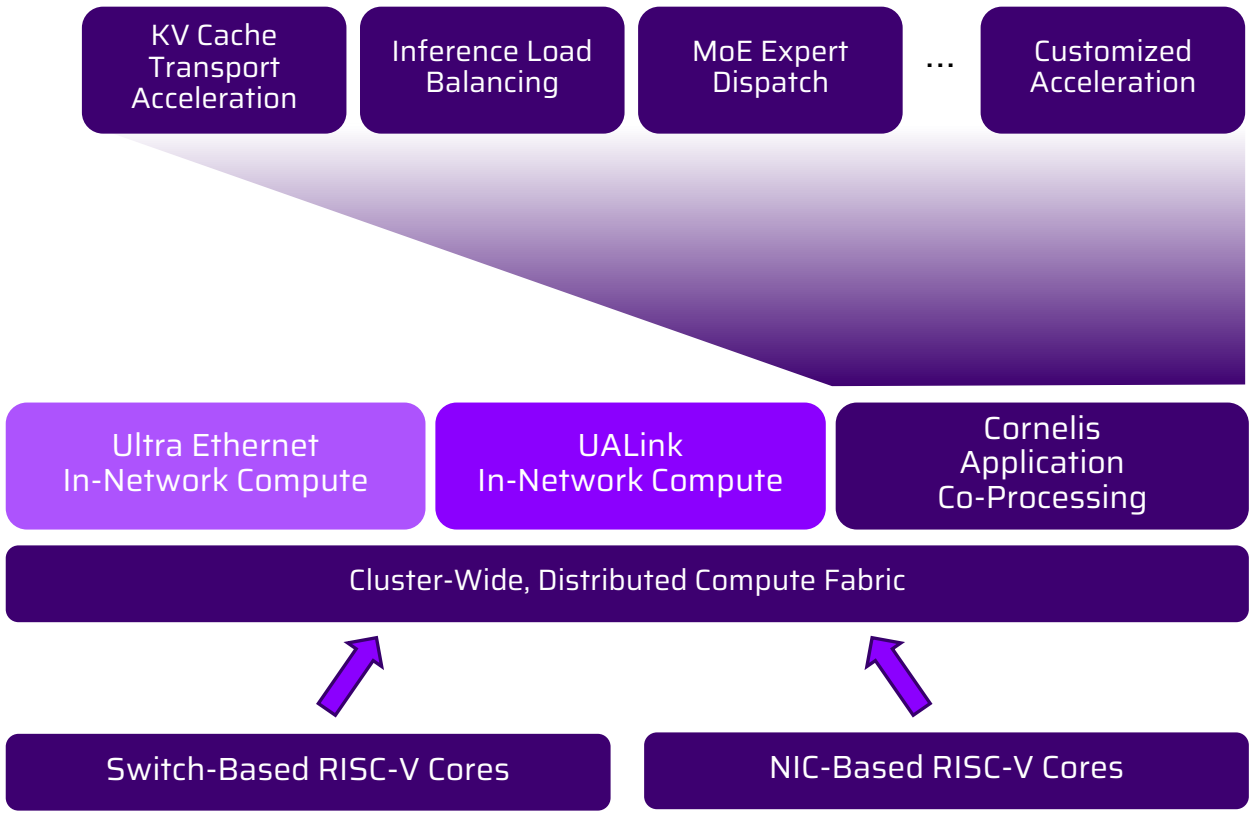
Provider/silicon division of labor



The common case runs in silicon; the provider stays thin.

CORNELIS™

CN7000: Beyond Traditional In-Network Compute

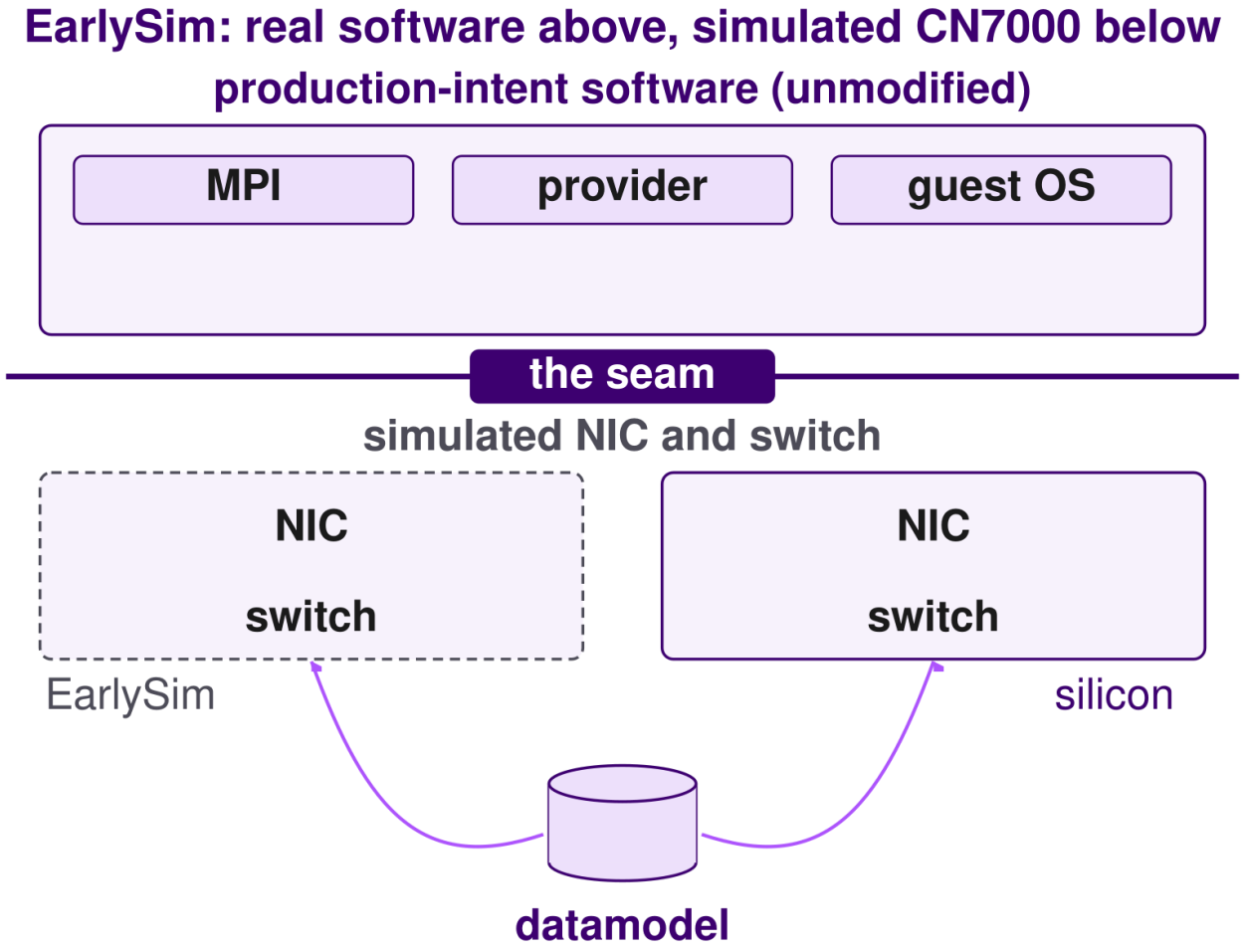


- Standards-compliant In-Network Compute
  - Ultra Ethernet and UALink INC
  - Accelerated collectives and triggered operations
- With programmable compute available at every port, the CN7000 Distributed Compute Fabric can be used as an application co-processor
- The Cornelis Distributed Compute Fabric (DCF) can deliver:
  - KV Cache Transport Acceleration
  - Inference Load Balancing
  - MoE Expert Dispatch
  - Speculative Data Prefetch
  - Gradient Compression
  - Sparse Data Aggregation
  - MPI Progress & Offload
  - PGAS and SHMEM Stack Offload
- SDK for technology partners
  - Customized, focused application co-processing
  - Leveraging distributed scalar and vector cores

CORNELIS™

Testing Before Silicon: EarlySim

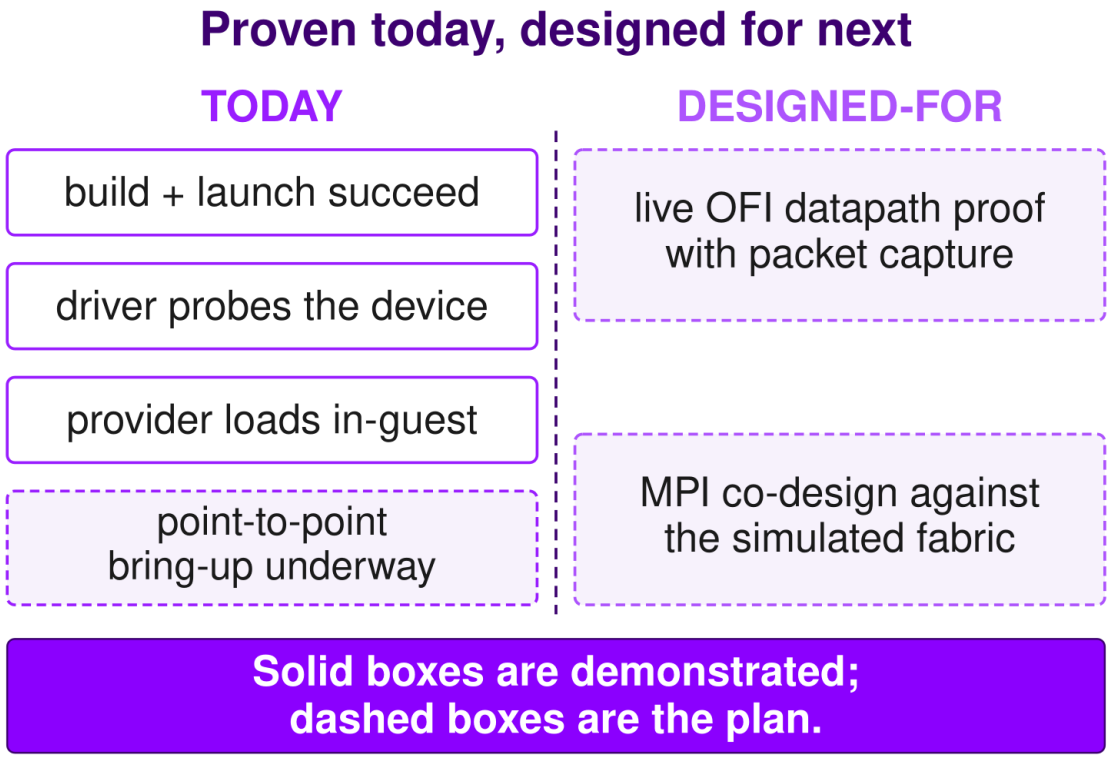
- EarlySim is a datamodel-driven cluster-on-a-host: above the seam runs the production-intent guest OS and provider stack, unmodified.
- Below that same seam, the CN7000 NIC and switch are simulated from the same machine-readable sources — descriptor formats, CSR definitions — that drive the silicon design.



CORNELIS™

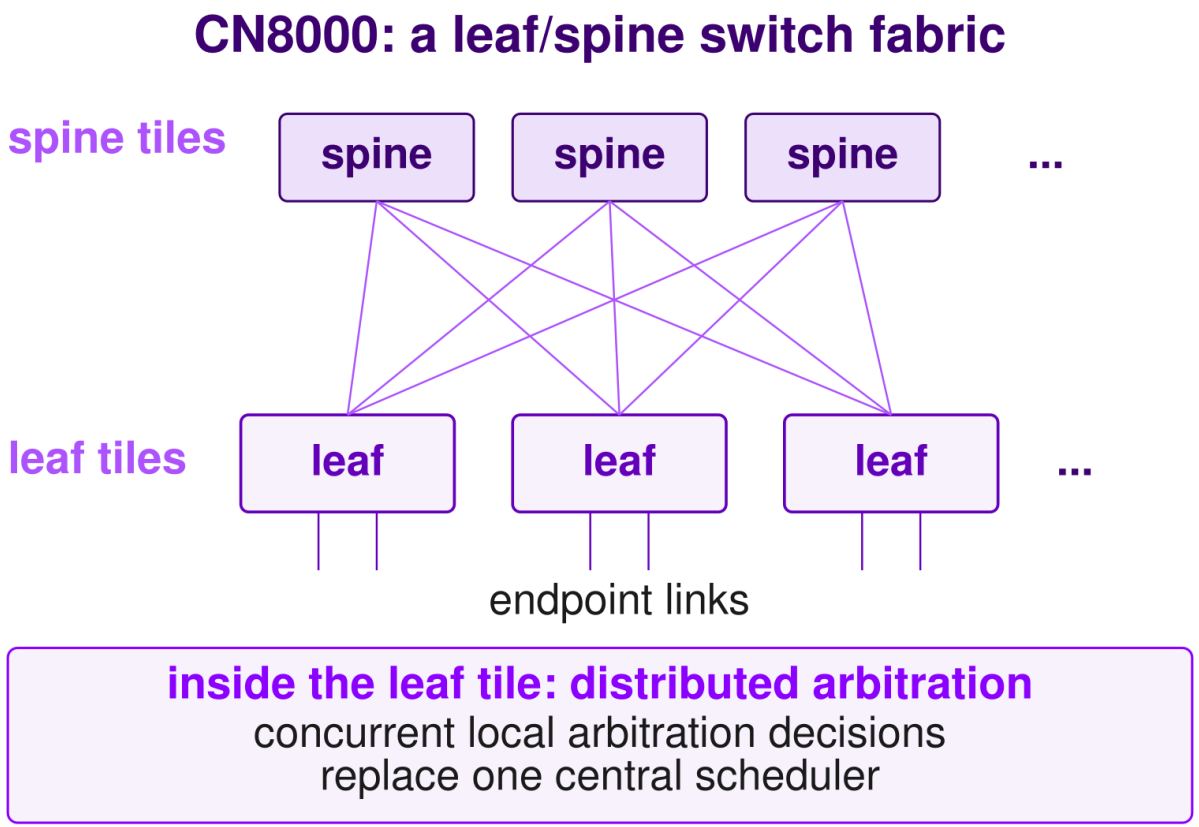
What We Can Run Today — and What Comes Next

- TODAY: build and launch succeed, the driver probes the device, and the provider loads in-guest; point-to-point bring-up with tag matching is underway. Hardware co-sim is being implemented
- DESIGNED-FOR: the upstream MPICH test suite, driven by MPICH's own runtests harness, running against the CN7000 provider in the simulated cluster — the same artifacts then drive real hardware.



CN8000: A Look Ahead to 480Tb/s

- CN8000 is a leaf/spine switch fabric built from switching tiles, multiplying fabric bandwidth over CN7000.
- Inside the leaf tile, many concurrent local arbitration decisions replace one central scheduler; this is architecture direction, not a committed product specification.

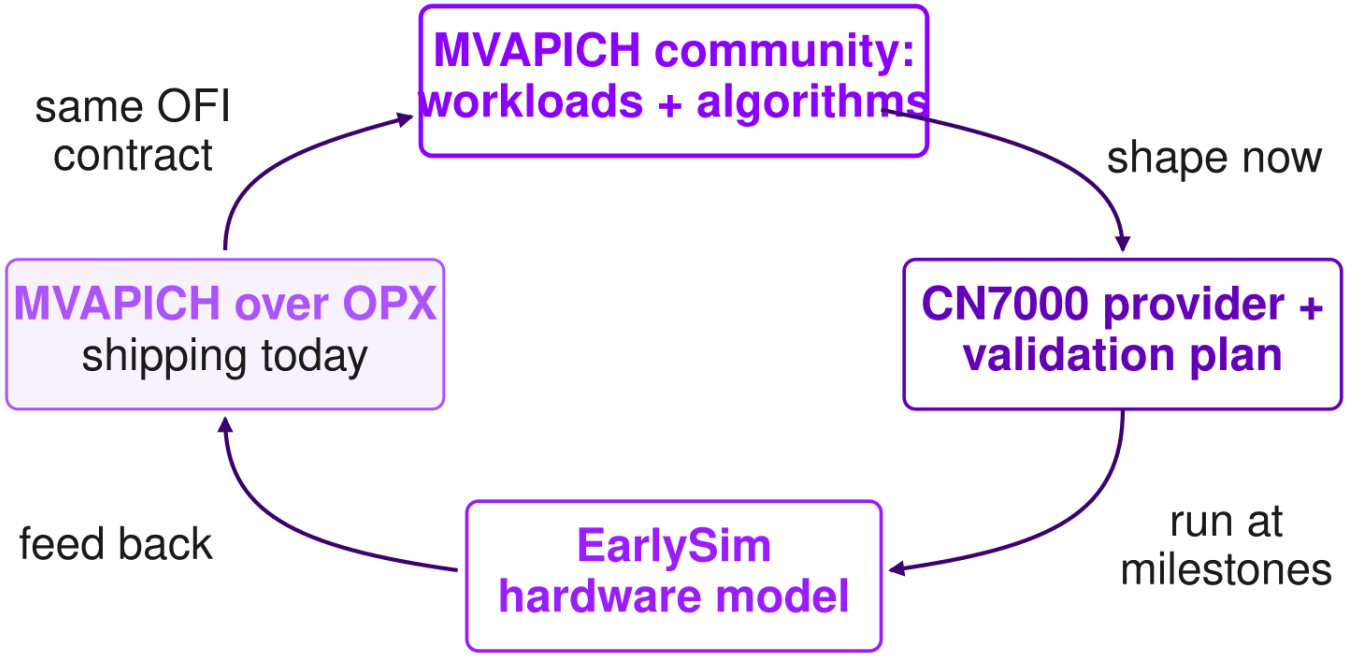


CORNELIS™

Collaboration Opportunities

- EarlySim: run MPI workloads against the simulated CN7000 cluster as the datapath milestones land — early feedback shapes the provider and the validation plan.
- Collective algorithms and frameworks: the programmable engines are open ground for algorithm work, and MVAPICH already runs over our OPX libfabric provider today.

Where community input feeds the design



CORNELIS™