High-Performance Computing Architecture: Fundamental Overview

NRIS

Norwegian research infrastructure services

Hicham Agueny, PhD
Scientific Computing Group
Info. Tech. Department, UiB/NRIS

Agenda

- Short introduction to HPC
- Compute Node **Architecture**
- CPU & GPU **Architectures**
- High-speed Network **Architecture**
- Communication Framework **Architecture**
- File System & Storage **Architecture**
- Optimization Strategies: GPUDirect Technology

Motivation

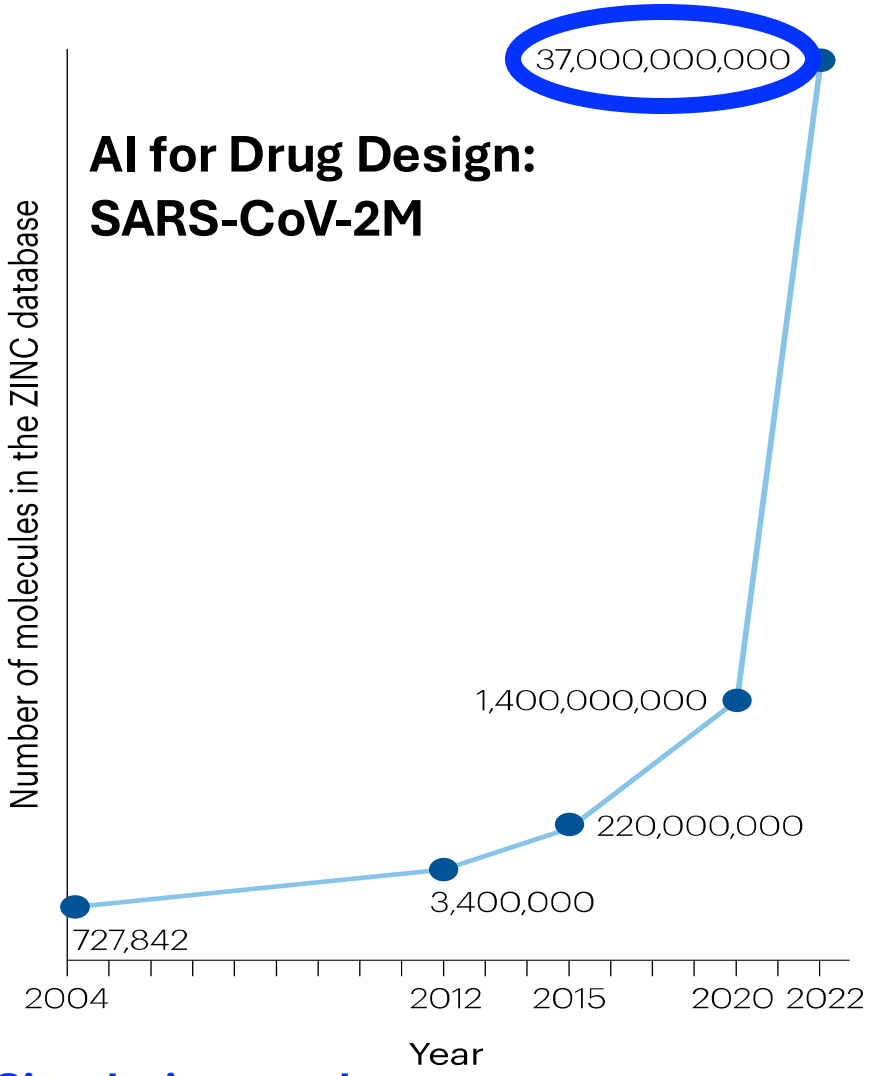
TOP4 Powerful Supercomputers as of Nov 2024

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**El Capitan** - HPE Cray EX255a, AMD 4th Gen EPYC 24C 1.8GHz, AMD Instinct MI300A, Slingshot-11, TOSS, HPE DOE/NNSA/LLNL United States	11,039,616	1,742.00	2,746.38	29,581
2	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Cray OS, HPE DOE/SC/Oak Ridge National Laboratory United States	9,066,176	1,353.00	2,055.72	24,607
3	**Aurora** - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel DOE/SC/Argonne National Laboratory United States	9,264,128	1,012.00	1,980.01	38,698
4	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	

Advance nuclear weapon science and scientific discovery.

US$600 million

Motivation



Simulations took:

90 days on **250 GPUs & 640 cpu-core**

1 day on **27000 GPU** Summit supercomputer

Nature Reviews Drug Discovery **23**, 141–155 (2024)

TOP4 Powerful Supercomputers as of Nov 2024

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**El Capitan** - HPE Cray EX255a, AMD 4th Gen EPYC 24C 1.8GHz, AMD Instinct MI300A, Slingshot-11, TOSS, HPE DOE/NNSA/LLNL United States	11,039,616	1,742.00	2,746.38	29,581
		76	1,353.00	2,055.72	24,607
	Max, Slingshot-11, Intel DOE/SC/Argonne National Laboratory United States				
4	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	

Advance nuclear weapon science and scientific discovery.

US$600 million

nature reviews drug discovery

Perspective | Published: 08 December 2023

Integrating QSAR modelling and deep learning in drug discovery: the emergence of deep QSAR

Why Care About HPC Architecture ?

- **Performance Optimization** – HPC systems rely on parallel processing. Knowing the architecture helps users optimize their code for better performance.

  And take full advantage of the system.

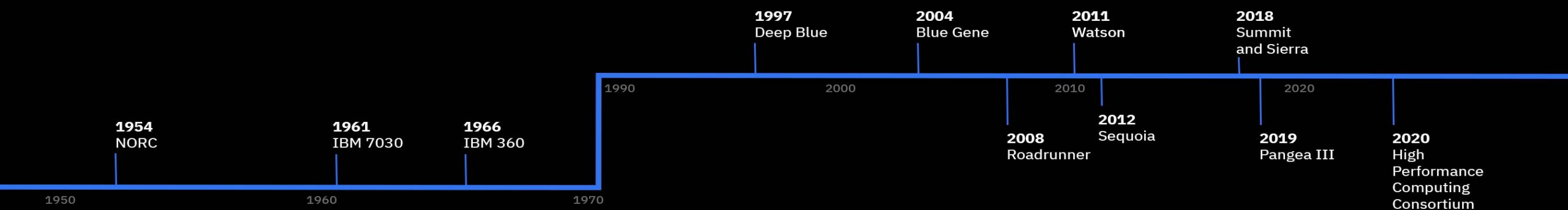
  Maximize efficiency in computations by utilizing e.g new features. (memory hierarchy, cpu-affinity, RDMA capability)

- **Resource Utilization** – Efficient use of CPU, GPU & memory reduce computational time and costs.

- **Troubleshooting & Diagnostics** – When things go wrong, understanding the architecture helps users **diagnose issues**

◆ **Just Curious about HPC architecture**
It is a JOY to understand how things work!

IBM Supercomputing timeline



1954
The Naval Ordnance Research Calculator helped forecast weather and performed other complex calculations.

1961
The **IBM 7030** was capable of 2 million operations per second.

MegaFLOPS
Million operation/s

1966
The **IBM 360** and its successors helped power NASA's Apollo program.

1997
Deep Blue wins its match with chess grandmaster Garry Kasparov.

2004
Blue Gene ushers in a new era of high-performance computing as it helps biologists explore gene development.

2008
Built for Los Alamos National Laboratory, **Roadrunner** is the first super-computer in the world to reach petaflop speed.

A new era of HPC
PetaFLOPS
Quadrillion

2011
Watson beats human competitors on Jeopardy!, earning a million-dollar jackpot for charity.

2012
Sequoia, the third-generation **Blue Gene** system, reaches speeds of 16.32 petaflops.

16 PetaFLOPS

2018
Summit begins work at Oak Ridge National Laboratory; a sister machine, **Sierra,** launches at Lawrence Livermore National Laboratory.

2019
IBM builds **Pangea III,** the world's most powerful commercial super-computer, for Total to accurately locate new energy resources.

2020
IBM helps launch the COVID-19 **High Performance Computing Consortium** to research the COVID-19 virus and its potential cures.

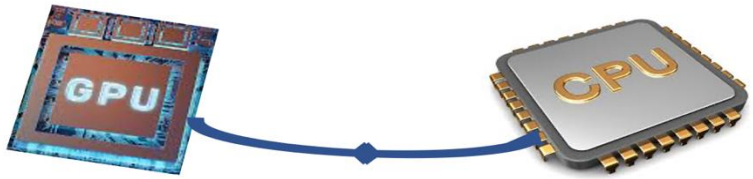
~500 PetaFLOPS

https://www.ibm.com/think/topics/supercomputing

Terminology

Heterogenous system is a system (HPC system) composed of:

- Different type of hardware and software that are connected e.g. CPUs & GPUs.



- Increasing performance & Scalability.

- **Cluster:** Collection of computers (nodes) that are just connected.

- **Server or compute node**: a single unconnected node.

- **HPC system**: Collection of compute nodes connected via a **high-speed network** called interconnect & forming a **single system**.
  teraFLOPS-petaFLOPS: HPC Cluster
  petaFLOPS-exaFLOPs: Supercomputer

Cluster

HPC cluster

Performance of a computer

- **The performance of a processor is measured by the quantity:**

FLOPS (**F**loiting-**P**oint of **O**pertaions **P**er **S**econd).

It is a measure of the speed of a computer to perform arithmetic operations.

For a single processor:

FLOPS = (Clock speed)x(cores)x(FLOPs/cycle)

=Peak performance

FLOP is a way of encoding real numbers (i.e. FP64 or FP32, FP16…)

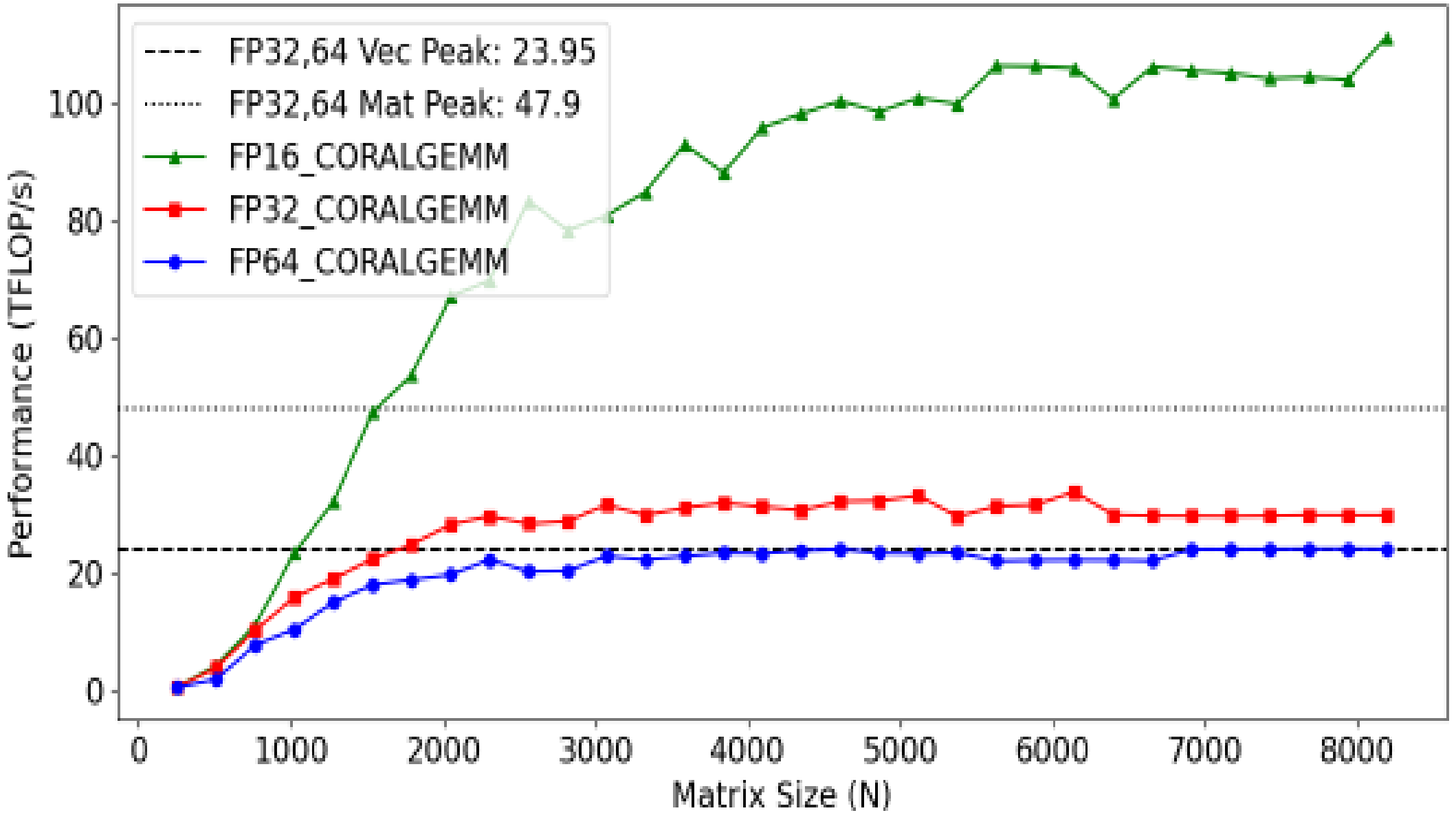
- 1 TeraFLOPS = 10^12 calculations per second.
- 1 PetaFLOPS = 10^15 calculations per second.
- **1 ExaFLOPS** = 10^18 calculations per second

(quintillion floating point operations per second)

1.000.000.000.000.000.000

- **1 GigaFLOPS**: processor can handel **billion floating-point (64 bit) operations every second**.
- **For matching: 1GigaFLOPS ~performing one calculation every second for 31.69 years**.

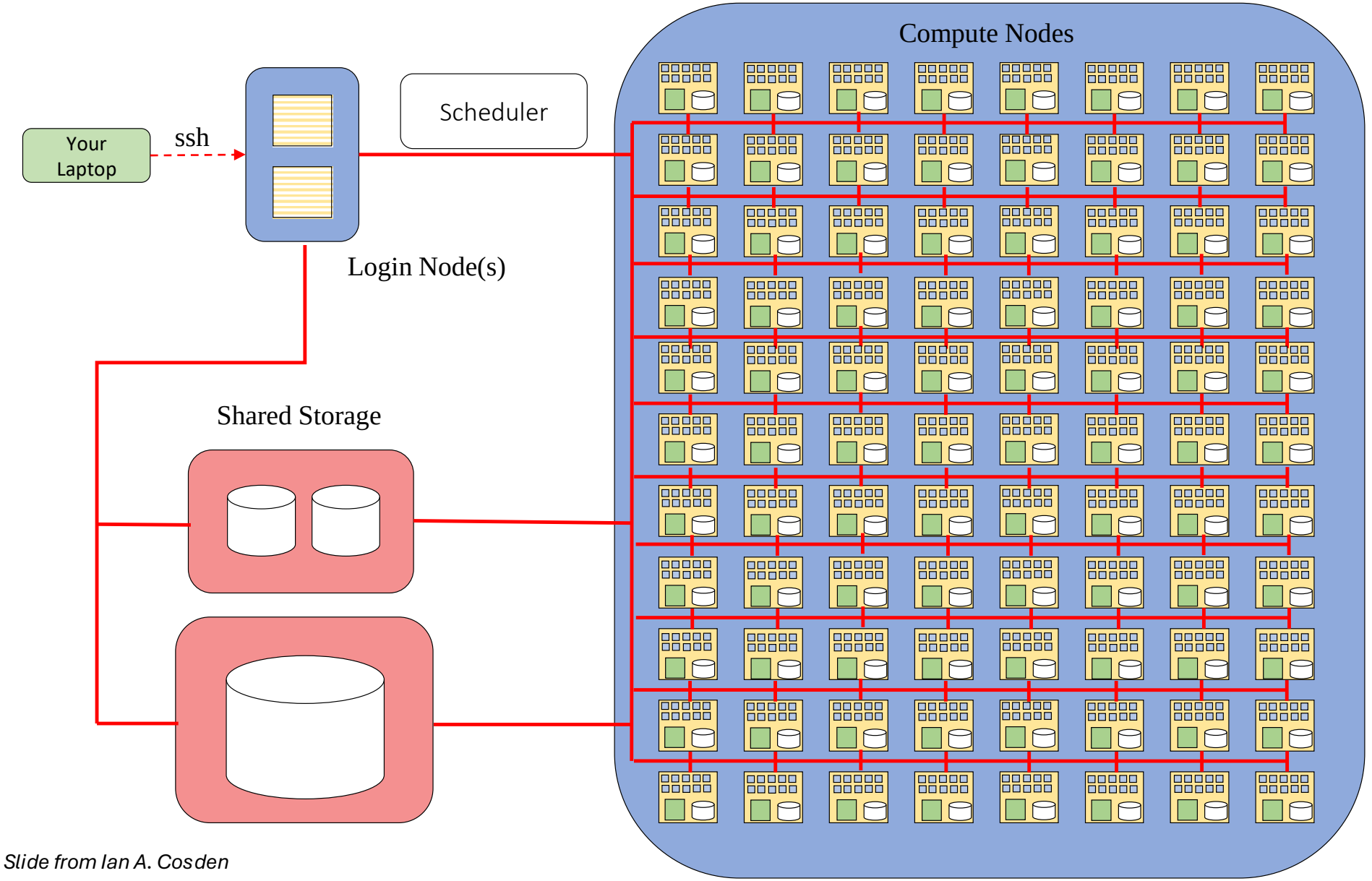
Comparison of peak performance: FP64 vs FP32 vs FP16



https://www.osti.gov/servlets/purl/2224161

Basic Cluster-Architecture

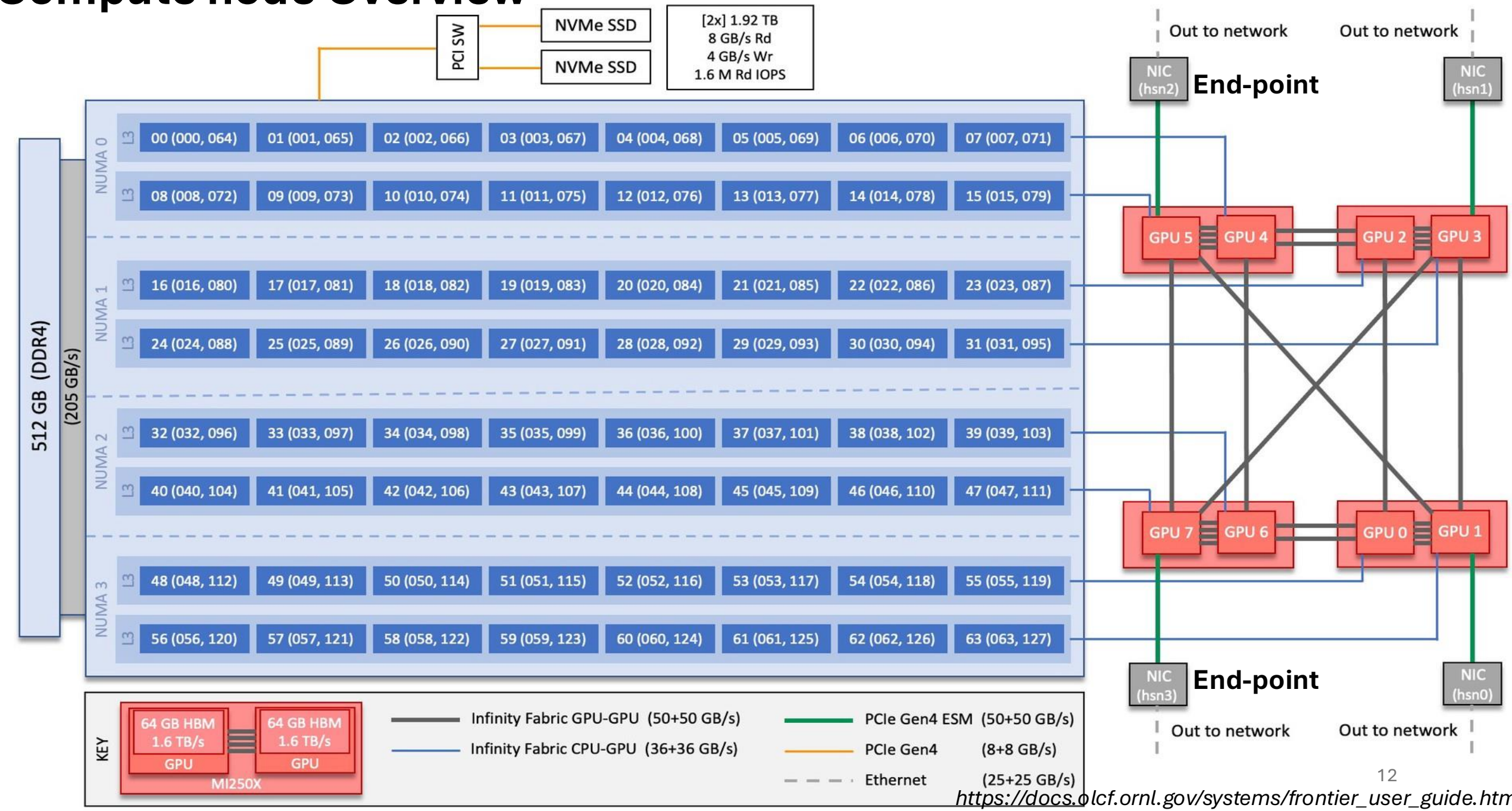
Basic Cluster-Architecture



Slide from Ian A. Cosden

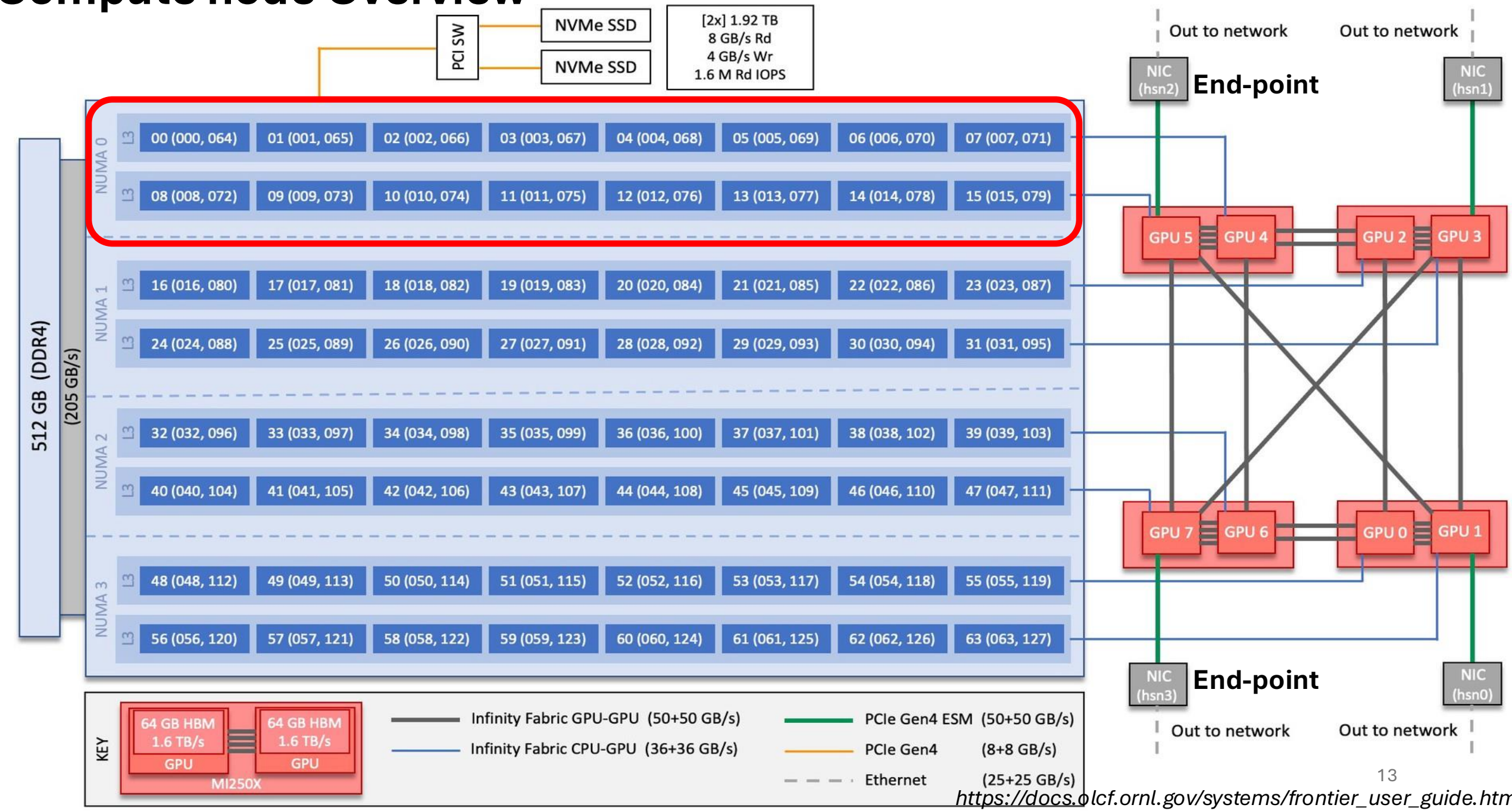
https://indico.cern.ch/event/814979/contributions/3401193/attachments/1831477/3105158/comp_arch_codas_2019.pdf

Compute node Overview



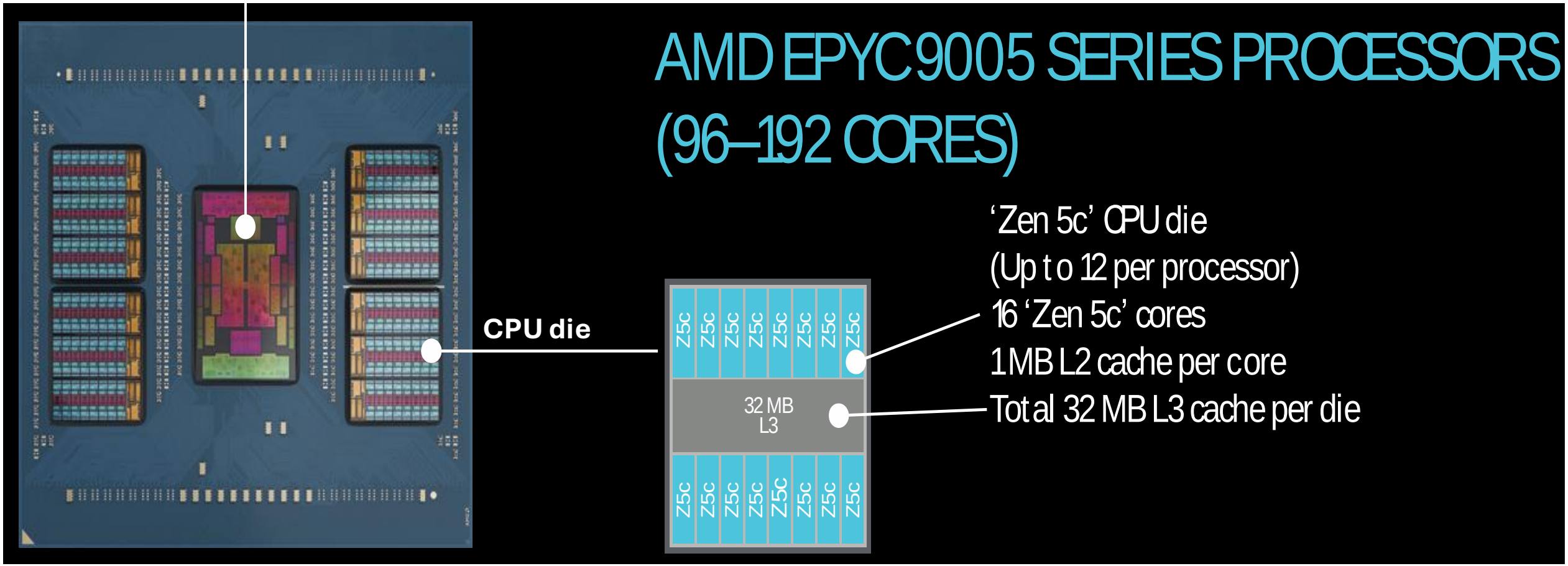
https://docs.olcf.ornl.gov/systems/frontier_user_guide.htm

Compute node Overview



https://docs.olcf.ornl.gov/systems/frontier_user_guide.htm

CPU Processor Architecture: 5th Gen AMD EPYC



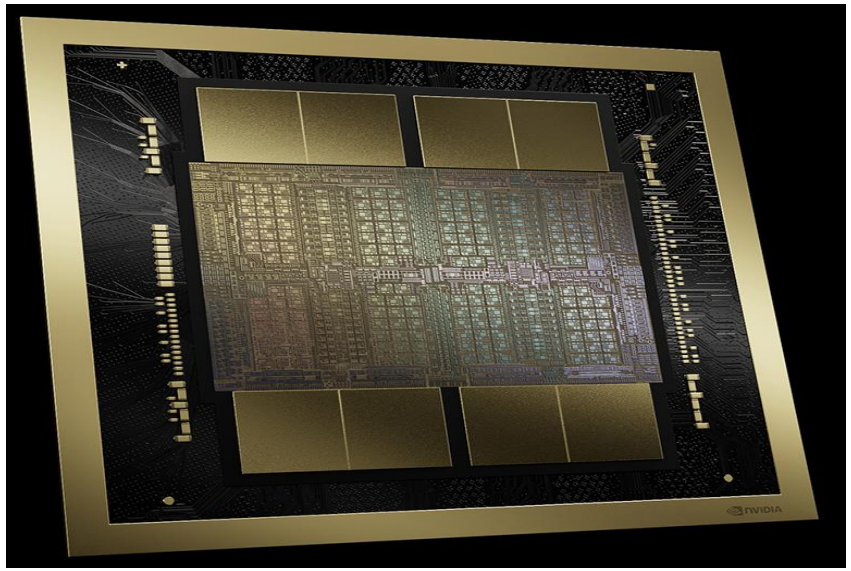
12 CPU die
Each die has 16 cores
16 cores share 32 M L3 cache
Max Memory Capacity **6 TB**

'Zen 5c' core is produced at **3nm**
I/O die remains at **6nm**
Memory channels / max persocket
theoretical bandwidth **12 / 614 GB/s**

Core density up **192 cores**
The highest of any x86- architecture
CPU available today

https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/5th-gen-amd-epyc-processor-architecture-white-paper.pdf

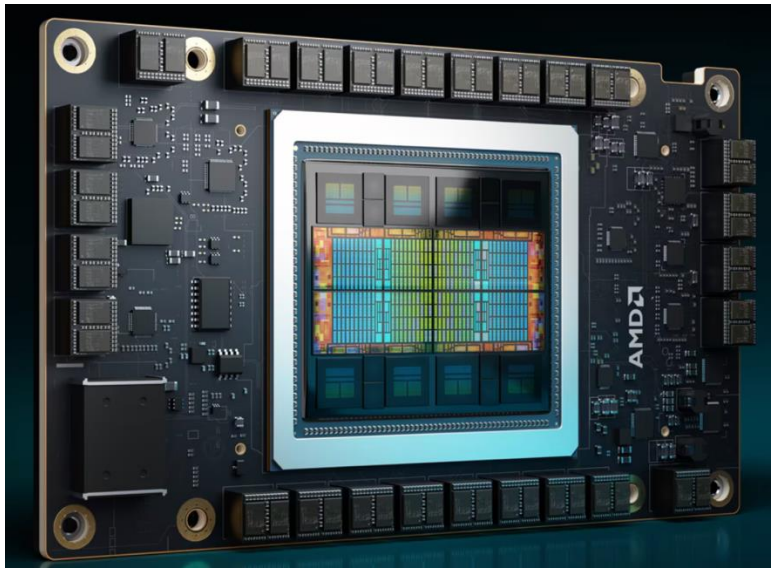
GPUs – Graphic Processing Units



NVIDIA BlackWell GB202 GPU

192 Streaming Multiprocessors (SMs)
24,576 CUDA-cores
92.2 Billion transistors
186GB
8 TB/s
80 TFLOPS (FP32)

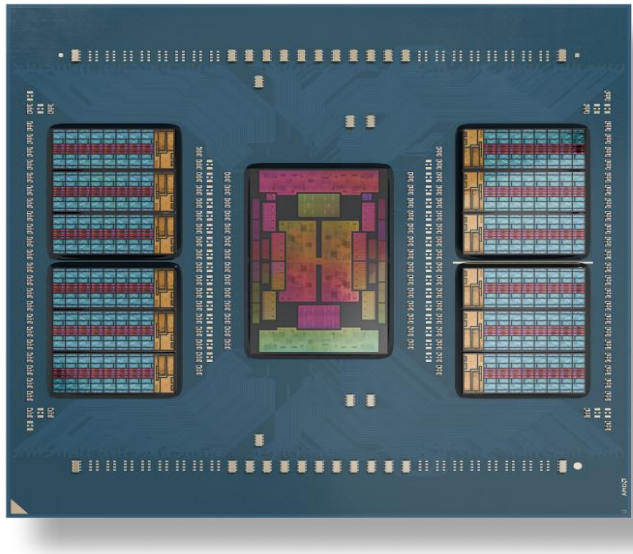
https://images.nvidia.com/aem-dam/Solutions/geforce/blackwell/nvidia-rtx-blackwell-gpu-architecture.pdf



AMD MI325 X GPU

304 Compute Units (CUs)
19,456 Stream processors
153 Billion transistors
256 GB
6 TB/s
163.4 TFLOPs (FP32)

https://www.amd.com/en/products/accelerators/instinct/mi300/mi325x.html

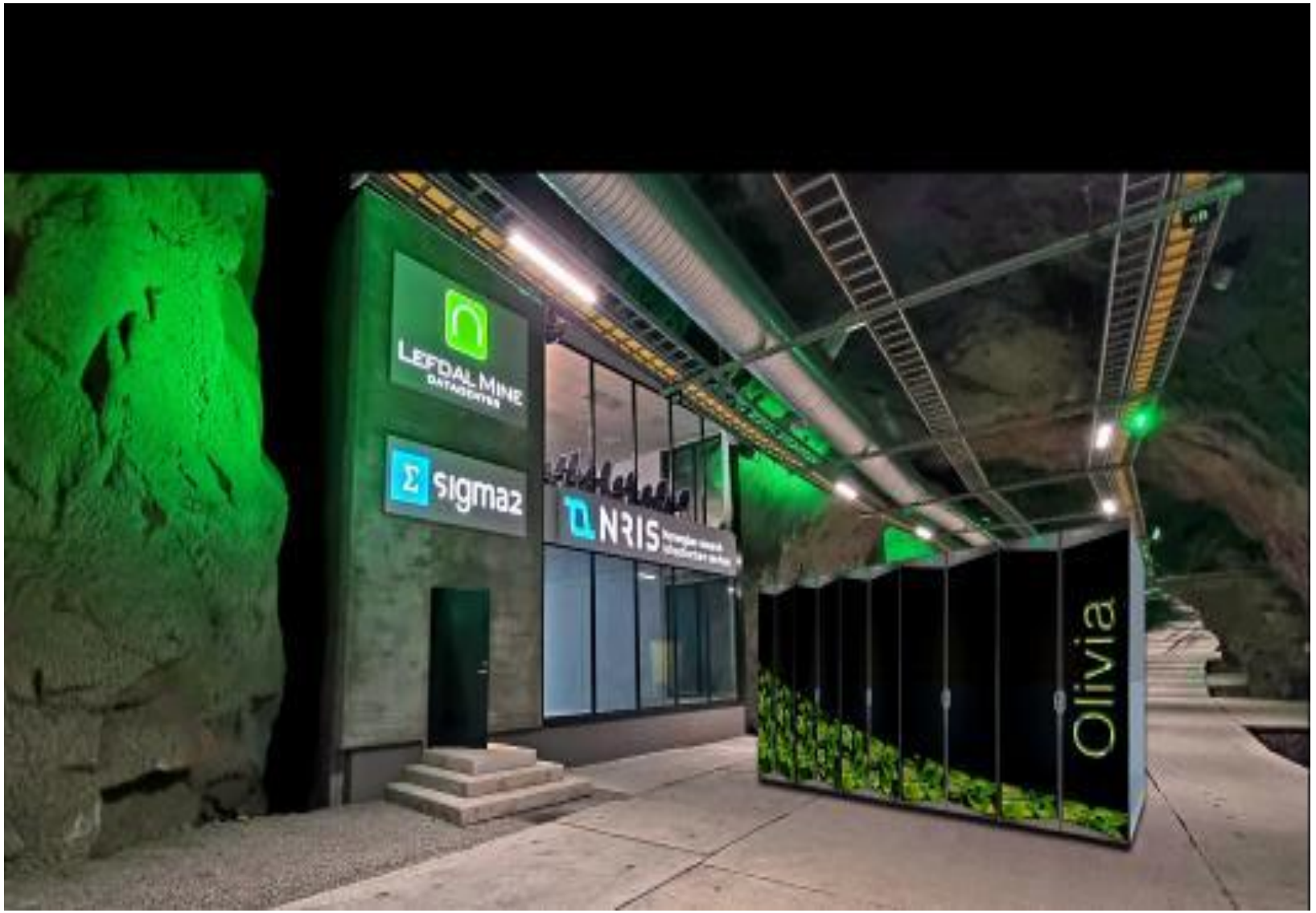


AMD EPYC 9005 CPU

12 CPU Dies
192 cores
6TB
614 GB/s

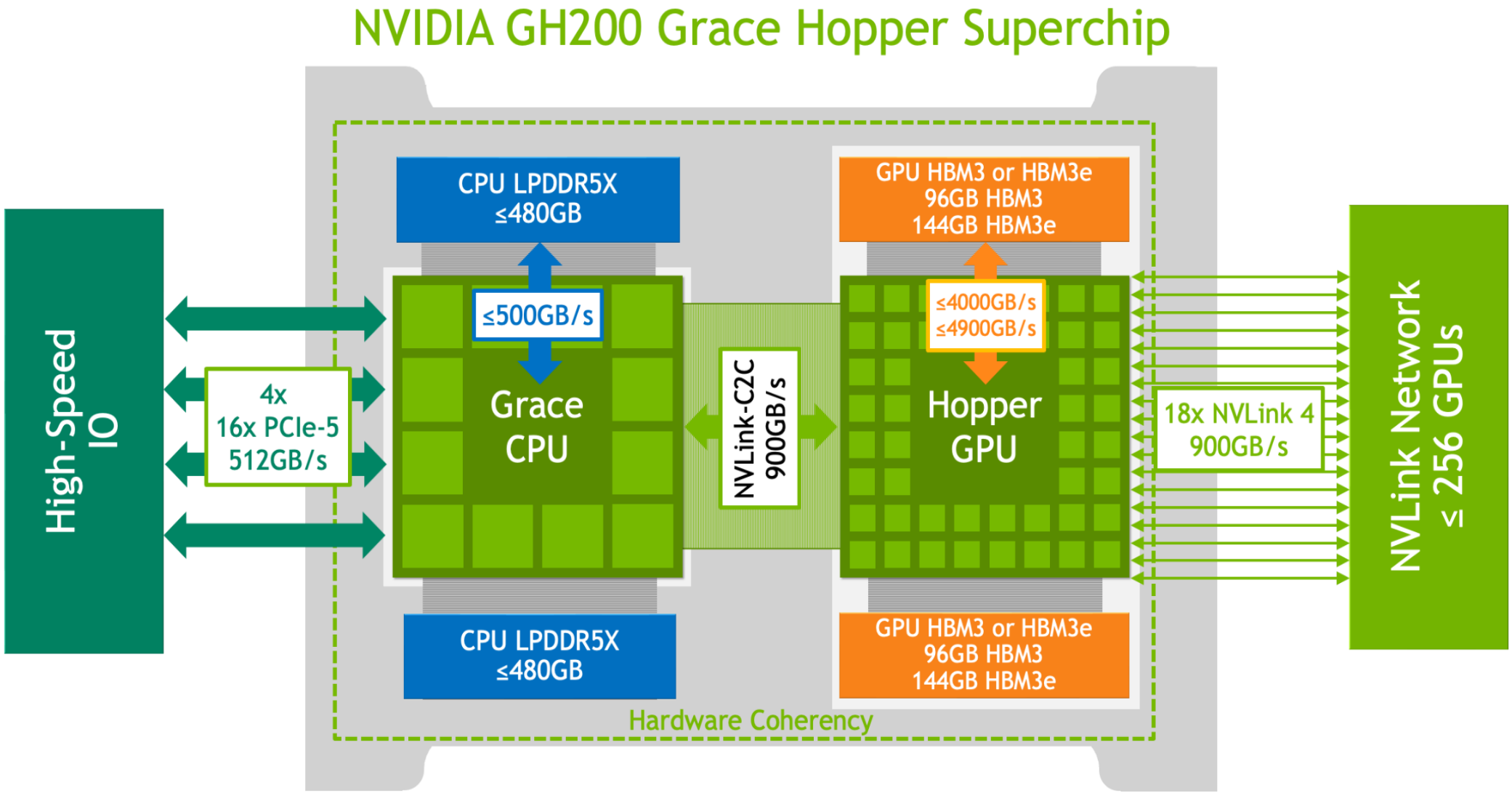
Norway’s Next Supercomputer: Olivia

- System: **HPE Cray** Supercomputing EX
- 76 GPU-nodes (**304** GPUs)
- NVIDIA **Grace Hopper Superchip** (**GH200** 96 GB)
- **HPE Slingshot Interconnect**



https://www.sigma2.no/news/2024/olivia-becomes-name-norways-next-supercomputer

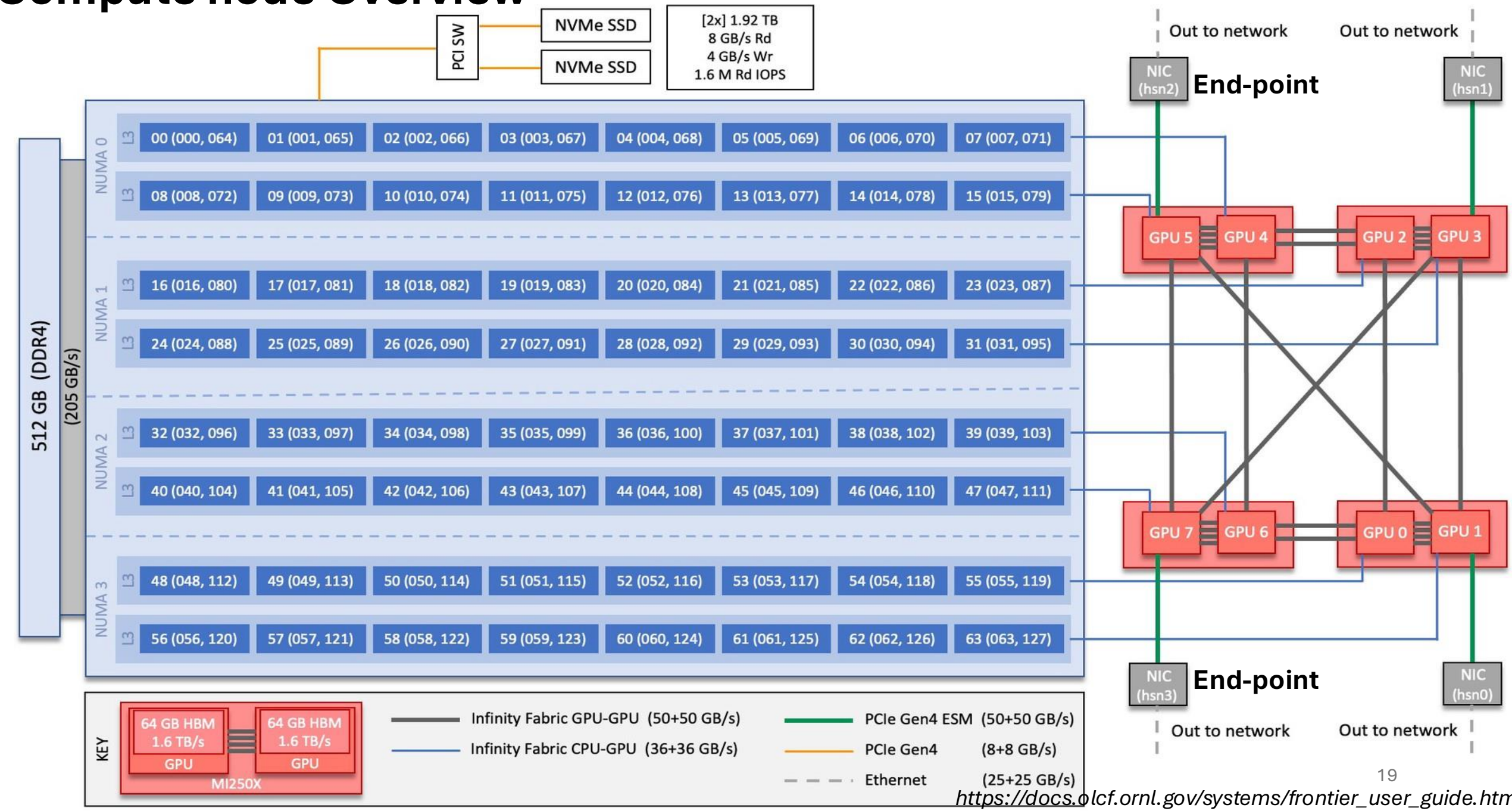
Overview of the Grace Hopper System



https://resources.nvidia.com/en-us-grace-cpu/nvidia-grace-hopper

Network Fabric Architecture

Compute node Overview



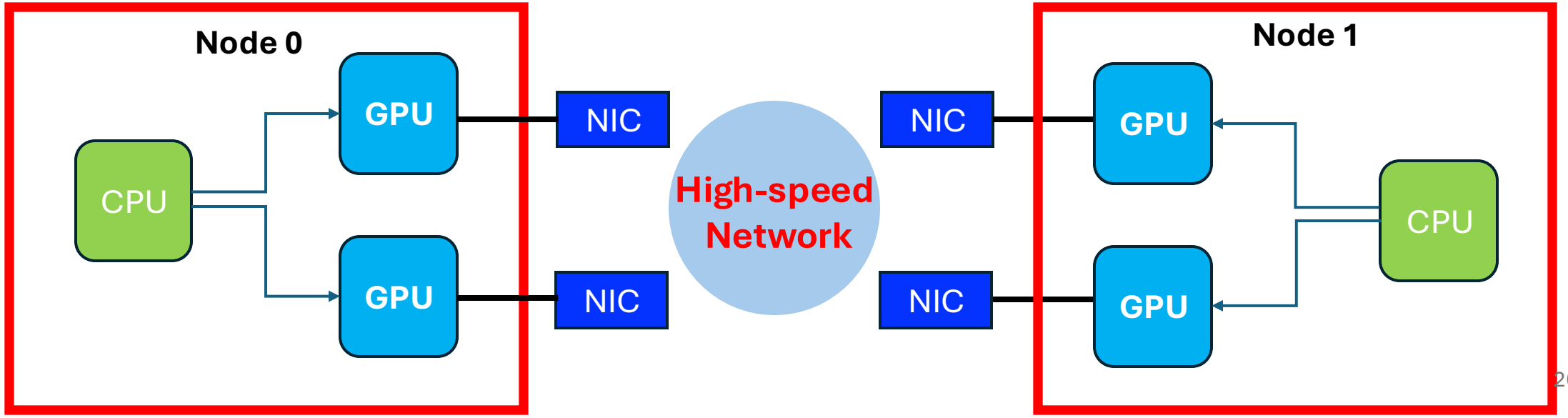
https://docs.olcf.ornl.gov/systems/frontier_user_guide.htm

Network Interface Card - NIC

- **NIC** provides a physical interface (**end-point**) between a compute node and a computer network.
- Network design allows each **NIC** to communicate with any other **NIC** in a system.
- **NIC** handles the processing of network protocol.
- **NIC** enables **RDMA** protocol between compute nodes.
- **RDMA** – *Remote Direct Memory Access*: direct memory-to-memory transfers over high-speed networks

without involving the CPU or OS kernel

- Processes in a node can use **NIC** to send/receive data to/from processes on other nodes in a system.



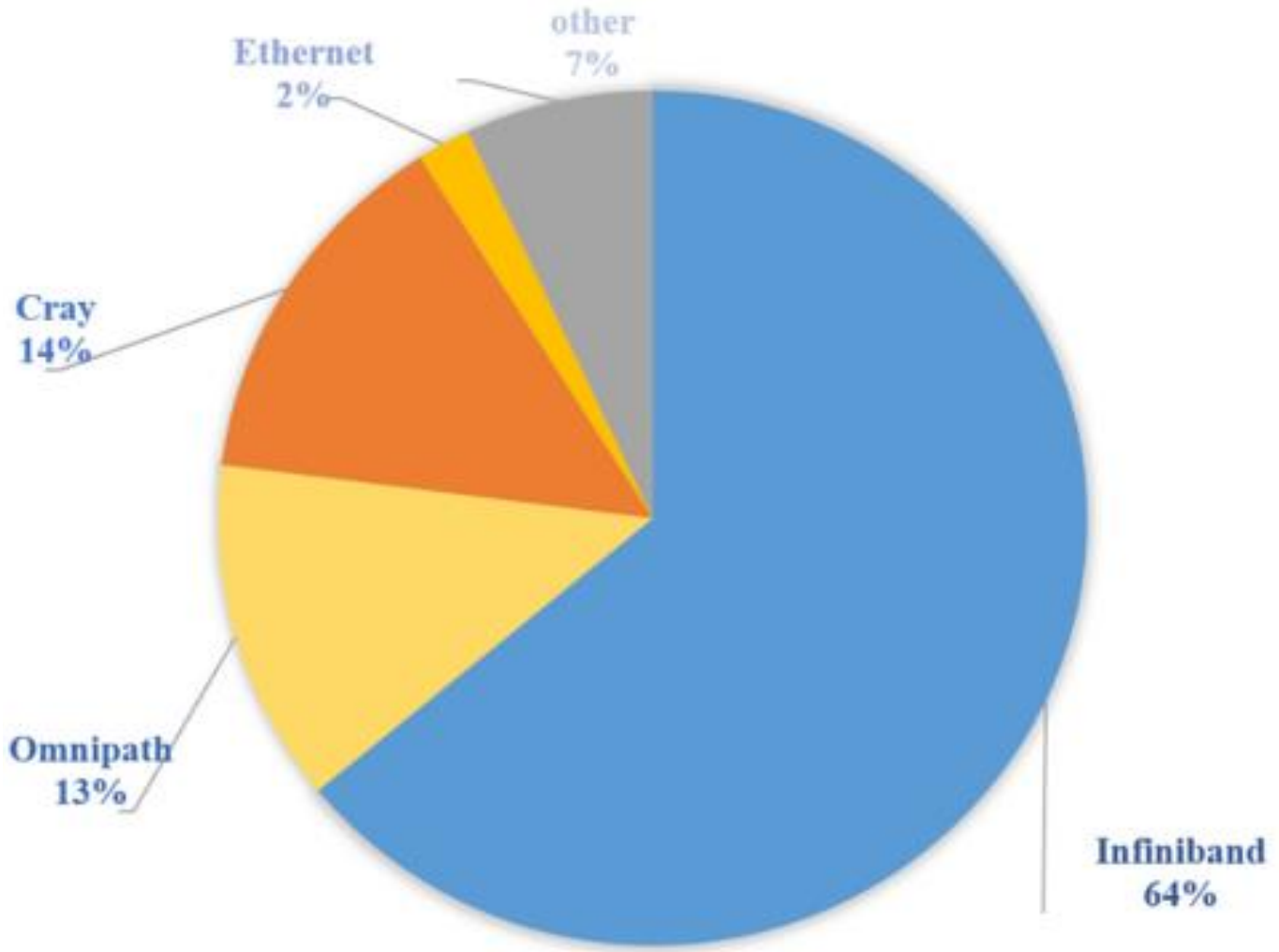
How does a compute node connect to thousands of other compute nodes in a large HPC system ??

Challenge: Design a high-speed network architecture that:

- **Maximizes bandwidth**: Enable fast data transfer.
- **Minimizes latency**: Achieve near-instantaneous communication.
- **Ensures scalability:** Supporting hundreds of thousands of endpoints without significant performance loss.

Requires High-Performance Interconnects

high-performance interconnects - HPIs

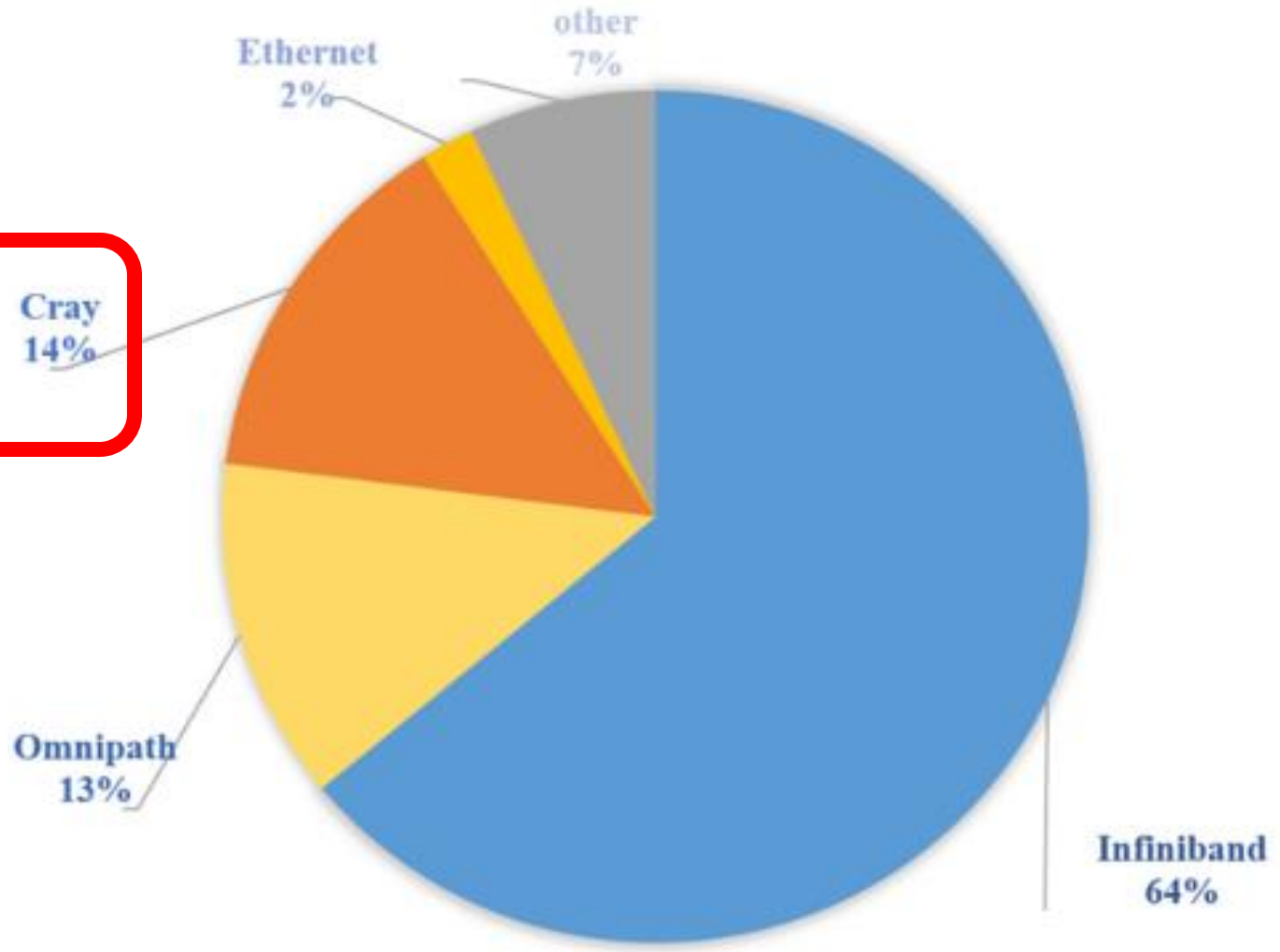


Distribution map of HPIs in the Top100 as of Nov 2021

Electronics 2022, **11**, 1369 10.3390/electronics11091369

high-performance interconnects - HPIs

HPE Slingshot Network



Distribution map of HPIs in the Top100 as of Nov 2021

Electronics 2022, **11**, 1369 10.3390/electronics11091369

HPE Slingshot interconnects

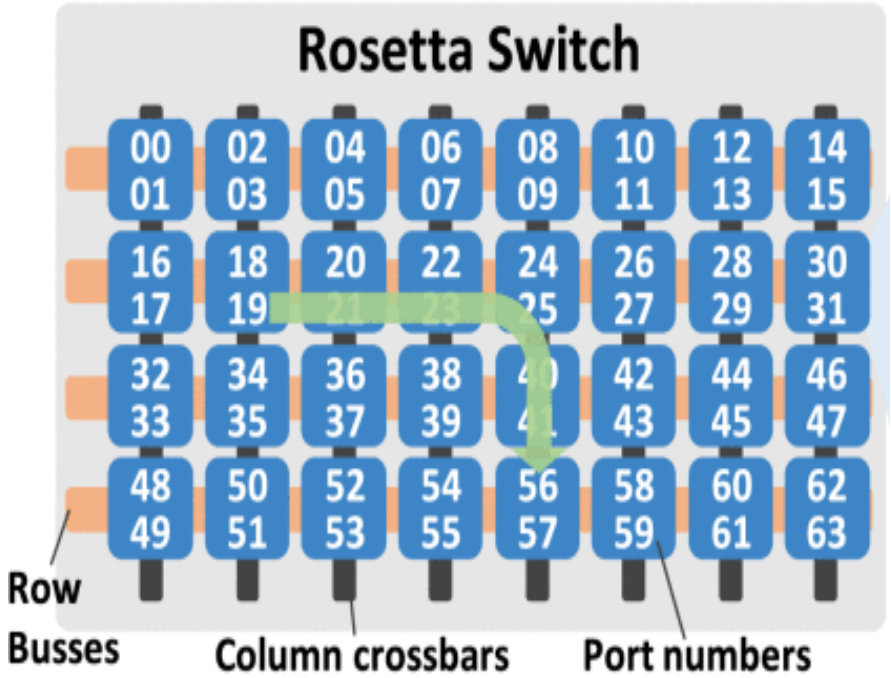
- Interconnect is one of the most critical components in large scale computing systems
- **HPE Slingshot networks** are constructed from two components:
  - **Cassini – a** PCIe Gen4 Network Interface cards (NIC)
  - **Rosetta – a** 64-port **200 Gbps Ethernet switch** (Slingshot Switch)
- These components are designed for **HPC (High-Performance Computing) & AI workloads.**

HPE Slingshot interconnects

- Interconnect is one of the most critical components in large scale computing systems
- **HPE Slingshot networks** are constructed from two components:
  - **Cassini – a** PCIe Gen4 Network Interface cards (NIC)
  - **Rosetta – a** 64-port **200 Gbps Ethernet switch** (Slingshot Switch)
- These components are designed for **HPC & AI workloads.**

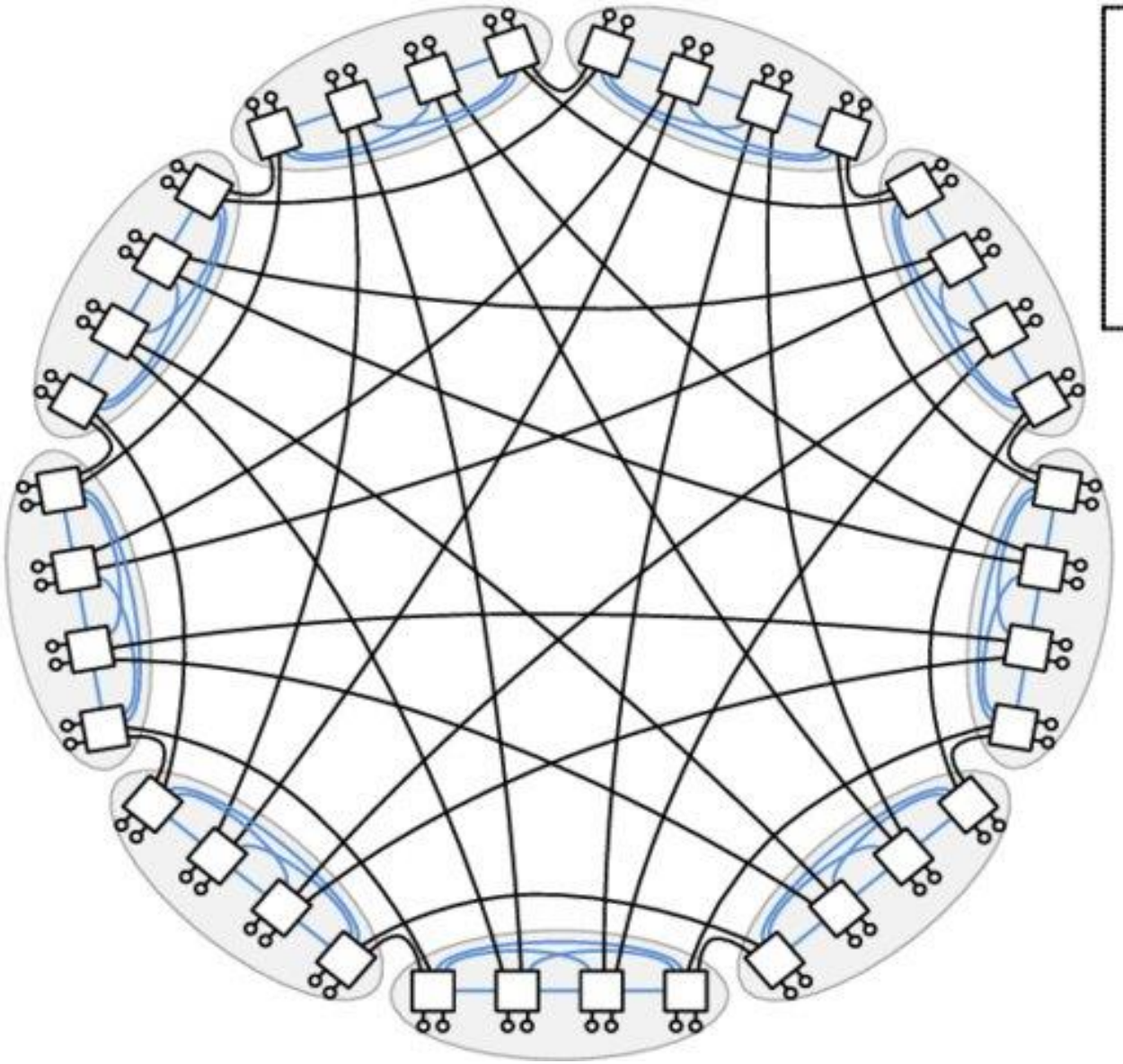
Role of Rosetta Switch

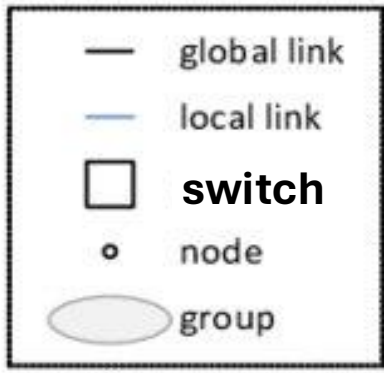
- Compute nodes are organized in a **network topology** (a structured way of connecting nodes via switches).
- If a node wants to send data to another node, it will go through **a series of network switches.**
- HPE Slingshot uses **Dragonfly topology** to ensure**:**
  - High-bandwidth
  - Low latency
  - Scalability
  - Cost effectiveness



http://ww.w.unixer.de/publications/img/sensi-slingshot.pdf

Dragonfly topology: Example of 72 end-points (compute nodes)





Dragonflies are organized as groups of switches.

Switches form a network that allows any node to reach any other node.

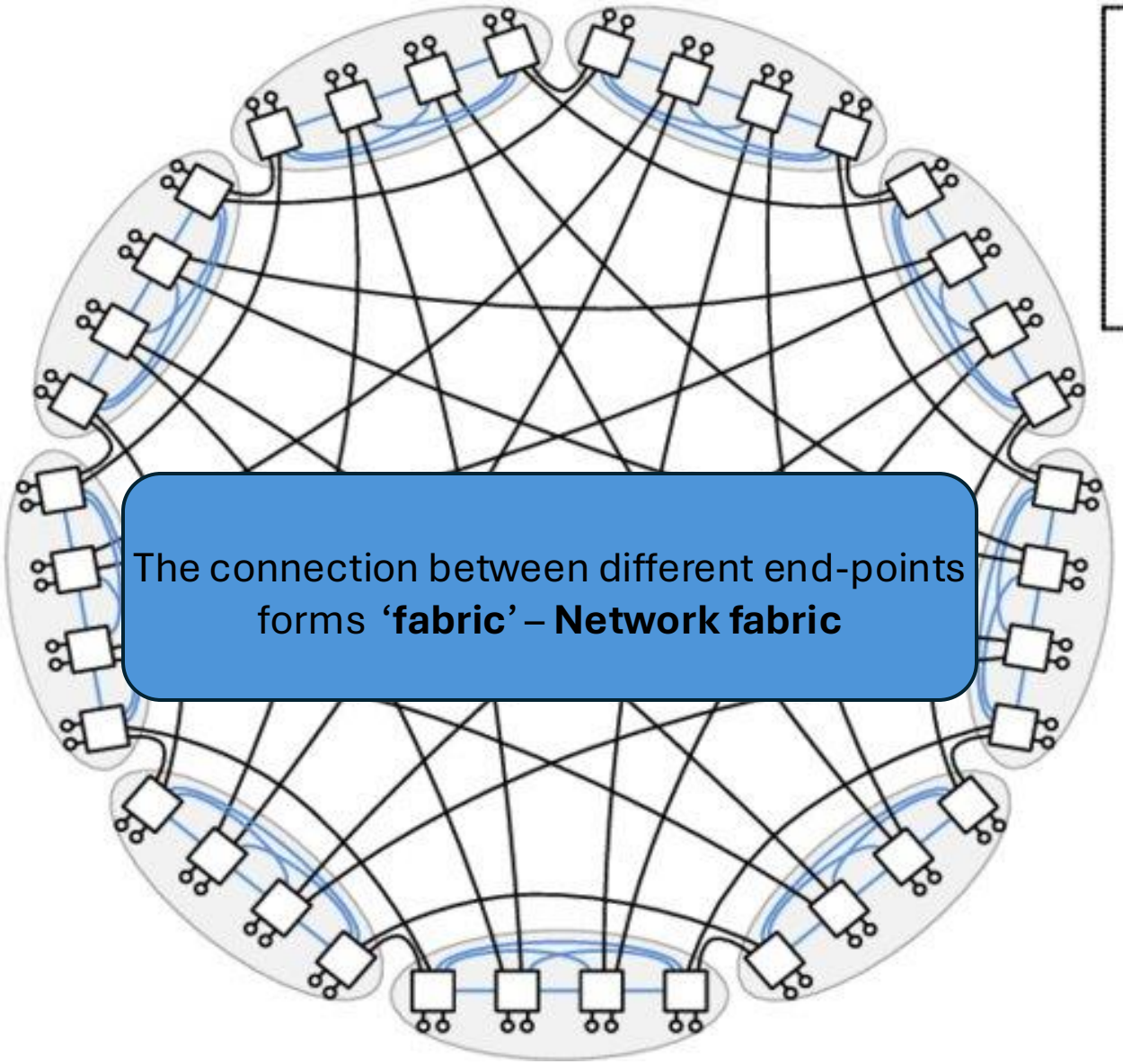
Ex. 9 groups
Each has 4 switches
Each switch connects 2 compute nodes
Total of 9x4x2 = **72 end-points (compute nodes)**

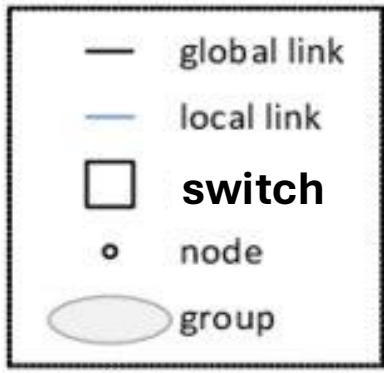
A compute node can communicate with any other node in at most 3 hops (steps)

1. **Node A** (Source) → **Switch 1** (1st hop)
2. **Switch 1** → **Switch 2** (2nd hop)
3. **Switch 2** → **Switch 3** (3rd hop)
4. **Switch 3** → **Node B** (Destination)

Figure adapted from: https://www.researchgate.net/publication/261313973_On-the-Fly_Adaptive_Routing_in_High-Radix_Hierarchical_Networks

Dragonfly topology: Example of 72 end-points (compute nodes)





Dragonflies are organized as groups of switches.

Switches form a network that allows any node to reach any other node.

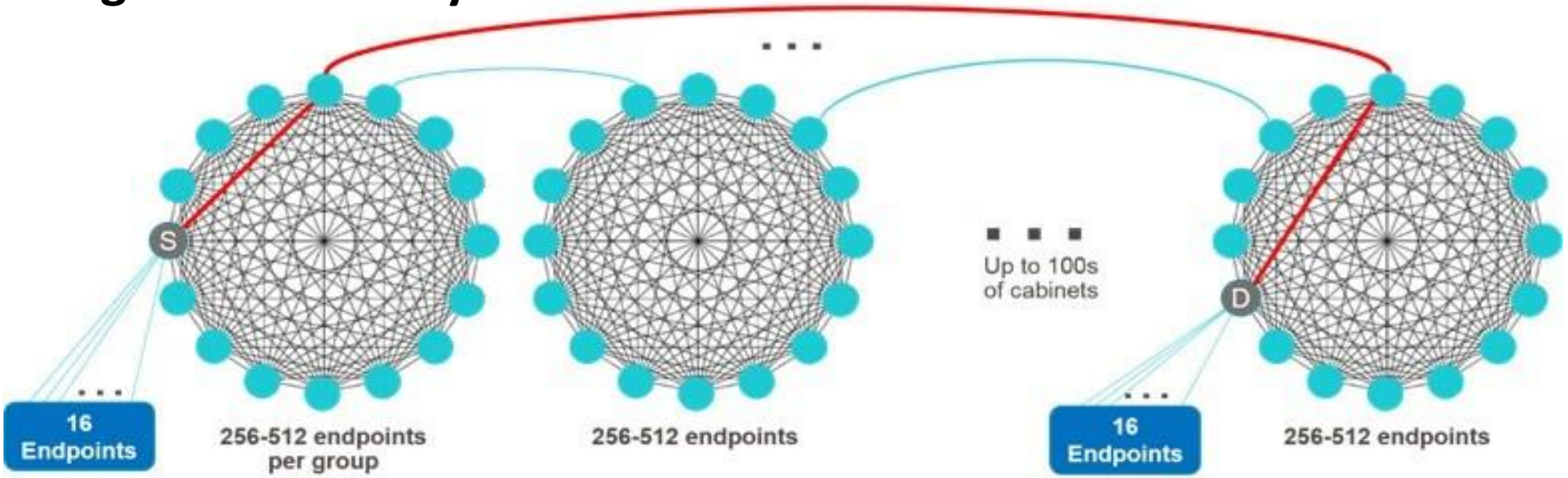
Ex. 9 groups
Each has 4 switches
Each switch connects 2 compute nodes
Total of 9x4x2 = **72 end-points (compute nodes)**

A compute node can communicate with any other node in at most 3 hops (steps)

1. **Node A** (Source) → **Switch 1** (1st hop)
2. **Switch 1** → **Switch 2** (2nd hop)
3. **Switch 2** → **Switch 3** (3rd hop)
4. **Switch 3** → **Node B** (Destination)

Figure adapted from: https://www.researchgate.net/publication/261313973_On-the-Fly_Adaptive_Routing_in_High-Radix_Hierarchical_Networks

HPE Slingshot scalability



Each group: **32 switches** each switch has **64 physical ports**

16 ports of each switch are used to connect to compute nodes

32x16 = 512 nodes on each group

31 ports are used to connect switches to each others

17 ports to connect to other groups

How many groups: **32 switches x 17 = 545 groups**

A system having **512 nodes x 545 groups** = **279 040** endpoints

HPE Slingshot Launched into Network Space: https://cug.org/proceedings/cug2022_proceedings/includes/files/pap121s2-file1.pdf
An In-Depth Analysis of the Slingshot Interconnect: http://ww.w.unixer.de/publications/img/sensi-slingshot.pdf

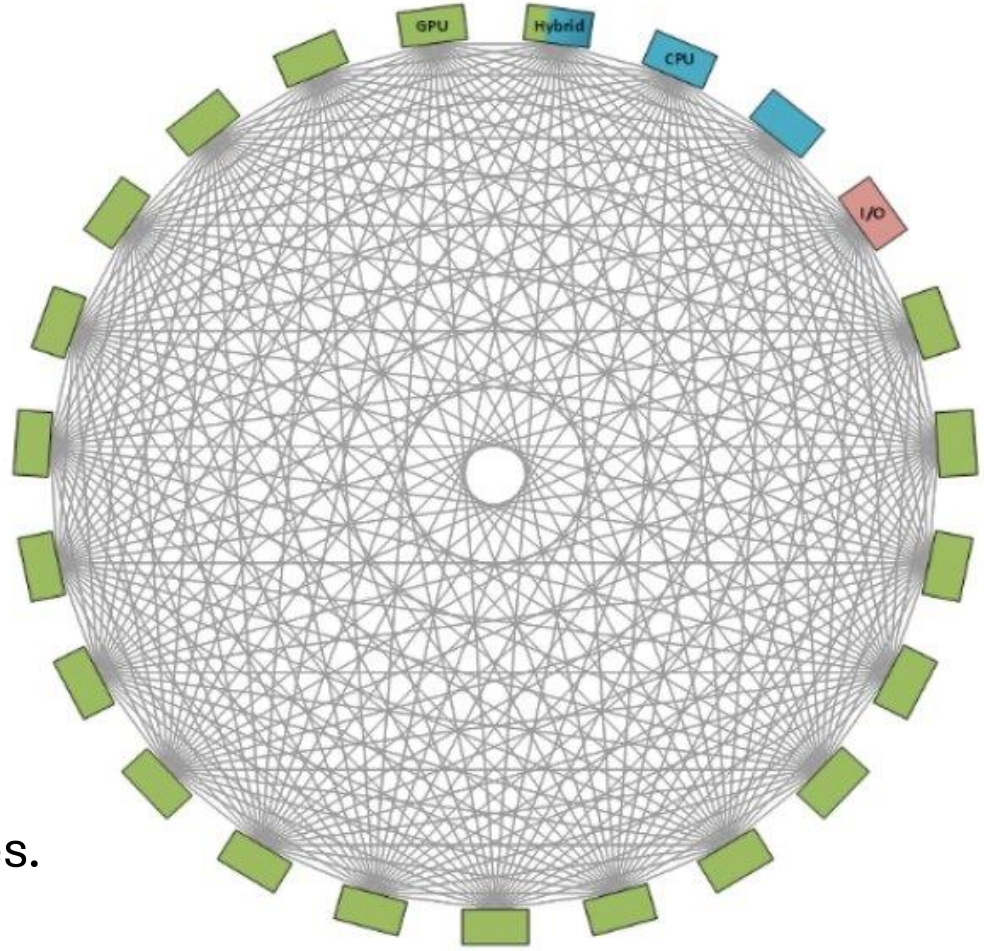
Complexity of Network Fabric

- Designing high-performance networks comes at a cost:
- Networks become more complex & challenging

Congestion - Traffic Bottleneck- is like a traffic jam in a network

Congestion occurs due to many reasons:

- Multiple packet flows directed to a single port.
- The destination node processing data too slowly, causing pauses.
- The destination port operating at lower bandwidth than others.



https://jlsrf.org/index.php/lsf/article/view/186

HPE Slingshot Features

Image from: https://www.olcf.ornl.gov/wp-content/uploads/2019/11/Intro-to-Frontier-public-20191008.pdf

Congestion - Traffic Bottleneck- is like a traffic jam in a network

Congestion occurs due to many reasons:

- Multiple packet flows directed to a single port.
- The destination node processing data too slowly, causing pauses.
- The destination port operating at lower bandwidth than others.

Advancement of Slingshot

Advanced congestion management

- Prevent congestion network
- Reduce packet loss
- Ensure **efficient data flow**
- Manage the bandwidth of flows

Adaptive routing to optimize network traffic

- Change a packet's path dynamically to less congested paths.
- Select the best path based on real-time **network load** and **queue lengths**.

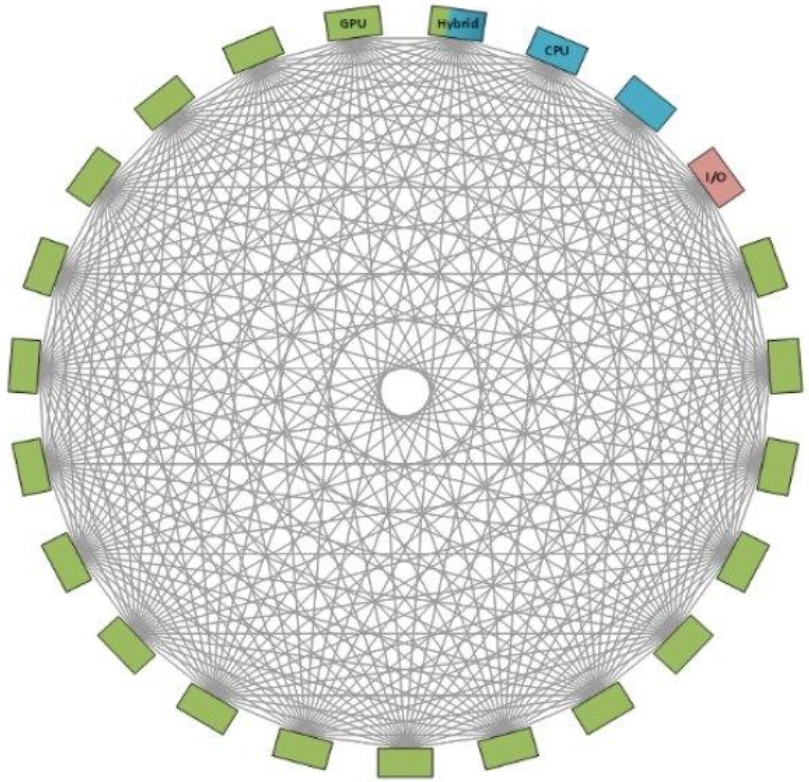
Quality of Service (QoS) for diverse workloads

- Set up policies to prioritize important traffic

HPE Slingshot Launched into Network Space: https://cug.org/proceedings/cug2022_proceedings/includes/files/pap121s2-file1.pdf

Beyond HPE Slingshot features…

- Need of a **software stack** that enables **direct communication** between **endpoints** (compute nodes, storage, etc.)
- Providing a standard API for **network communication**.
- APIs to ensure the use of the **fastest Network protocol**. (when data travel across **multiple nodes**)
- Well-known Open-source APIs for **network communications**:
  - **Libfabric (OFI - Open Fabric Interfaces)**
  - **UCX – Unified Communication X**



https://jlsrf.org/index.php/lsf/article/view/186

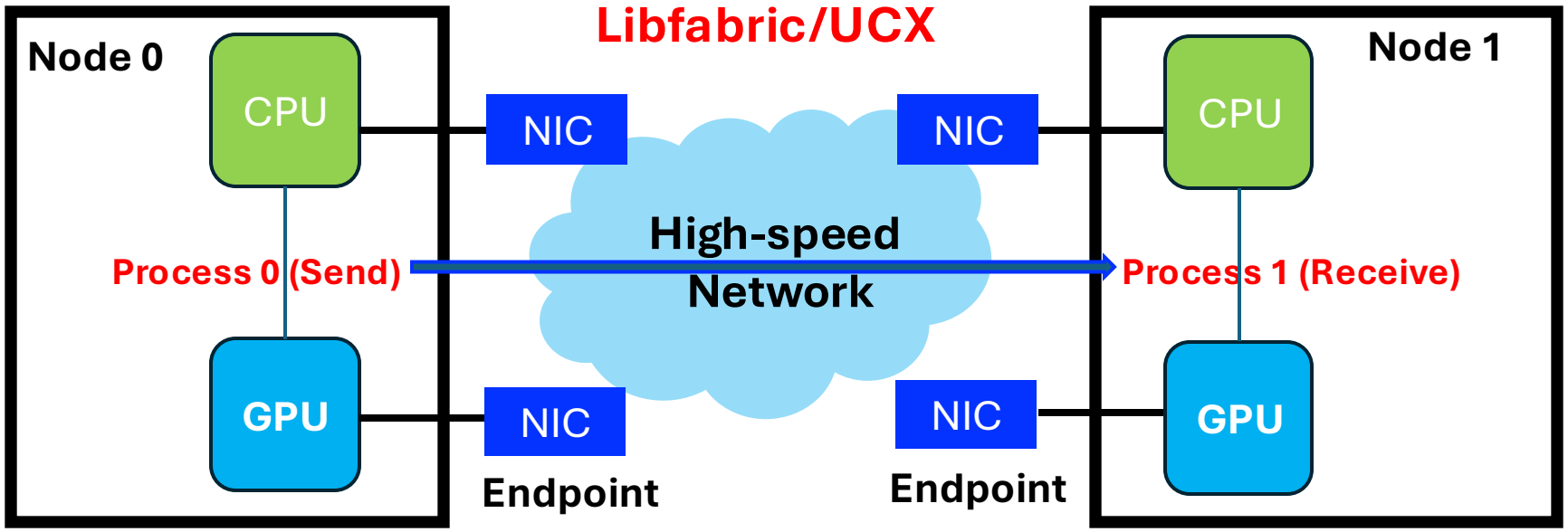
Abstract and optimize the complexity of Network Fabric

Libfabric & UCX

- **Libfabric** and **UCX** are both Open-source projects and serve the same goal -
- They are both collections of libraries that **abstract diverse networking technologies**.
- They are designed to enable **fast data transfer** between different **end-points** and with a **minimal delay**.
- Both leverage the high-performance **RDMA capabilities** where available (Network must support it).
- They offer APIs that allow applications to utilize various **high-performance interconnects** e.g. Slingshot, Infiniband, …

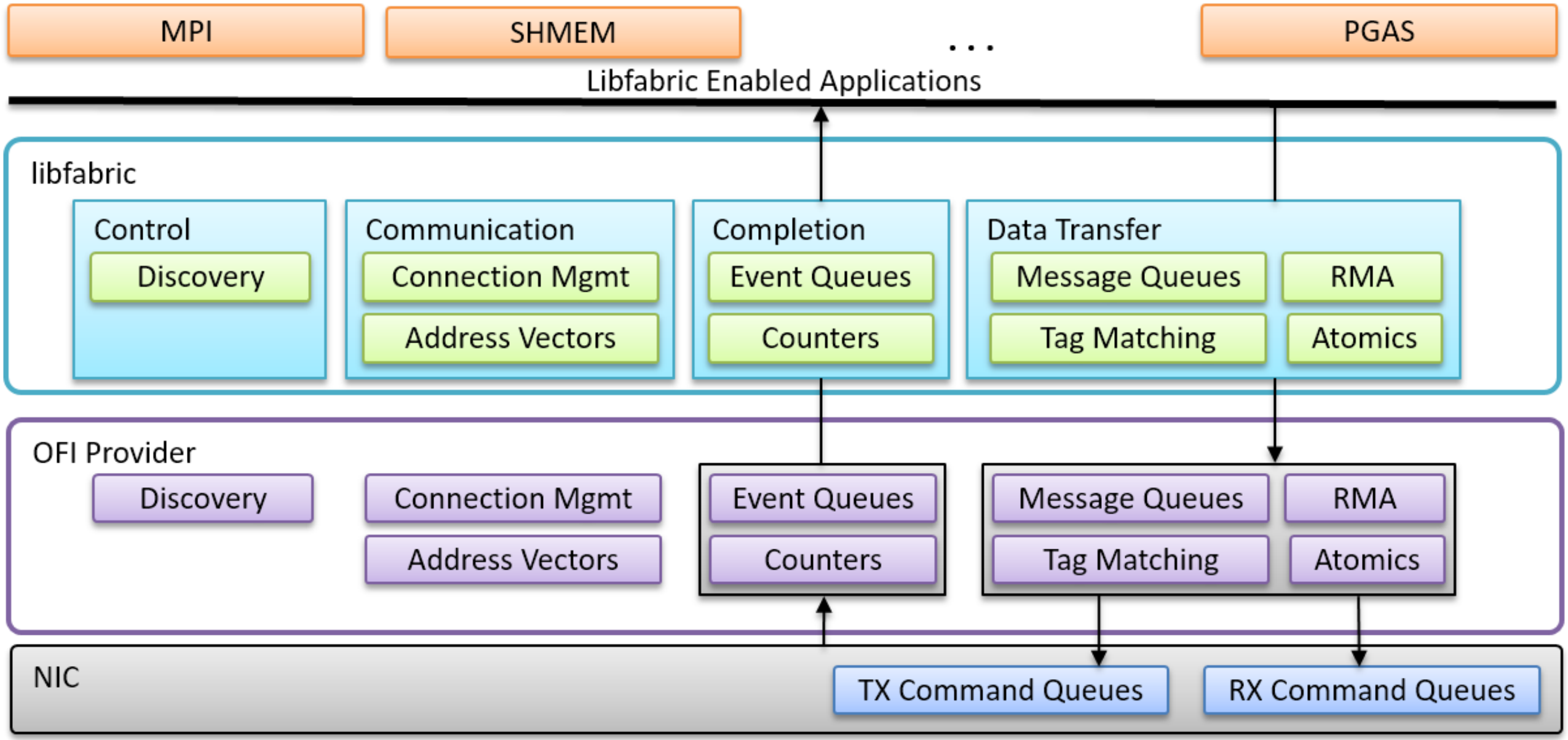
Slingshot 11 primarily utilizes the **Libfabric.**

So far, the **UCX** network module is **NOT supported with HPE CrayMPI on Slingshot 11** systems.



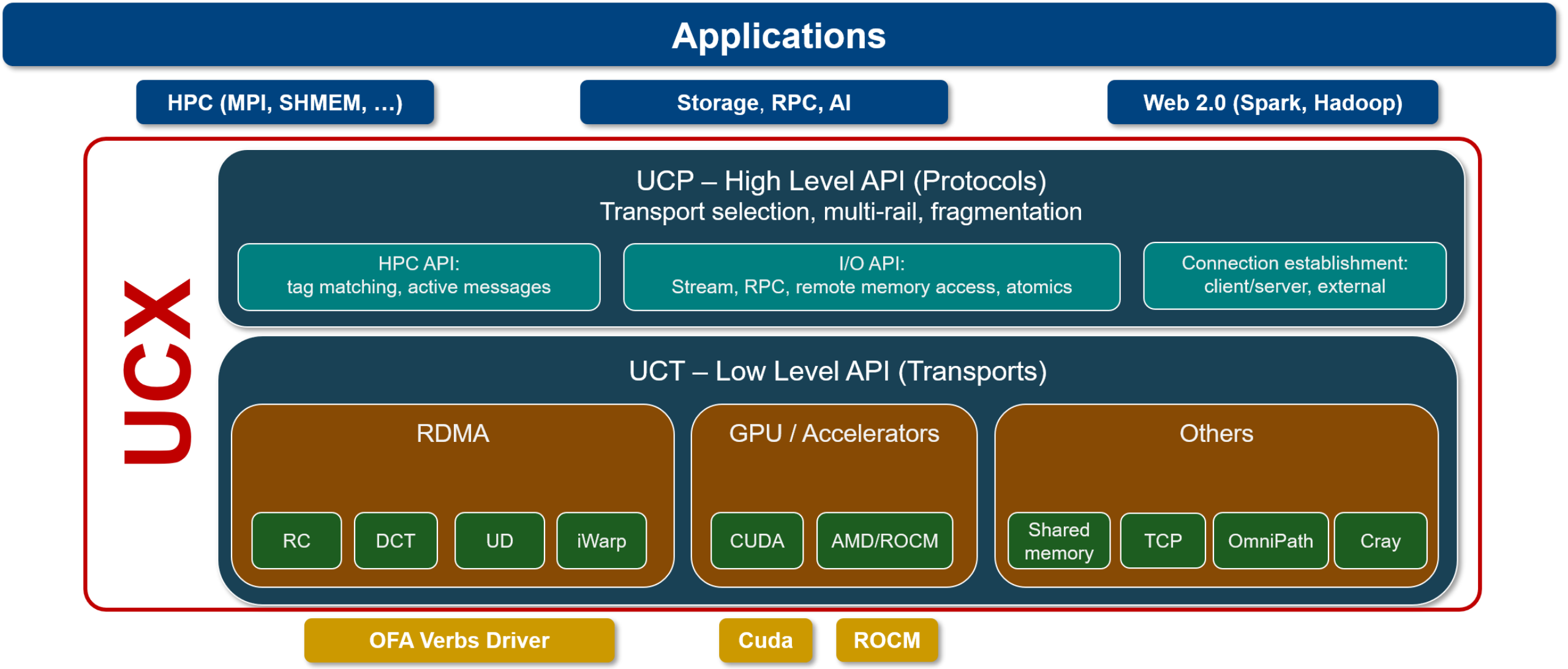
Libfabric Architecture

Libfabric Architecture



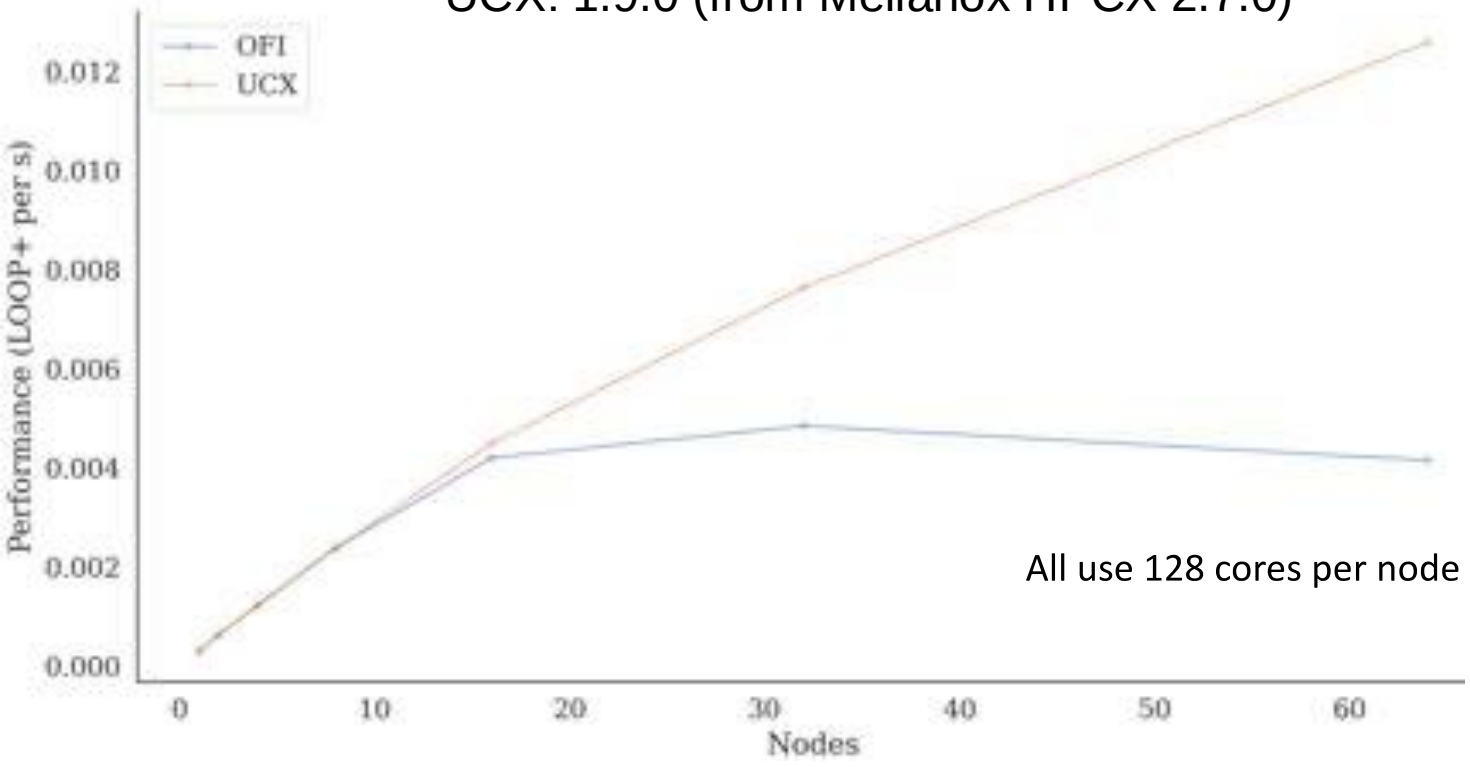
https://ofiwg.github.io/libfabric/

UCX Architecture



Benchmarking VAST: OFI vs UCX

- HPE Cray **Slingshot 10**
- HPE Cray MPICH 8.1.9
- OpenFabrics: 1.11.0.4.71 (libfabric)
- UCX: 1.9.0 (from Mellanox HPCX 2.7.0)



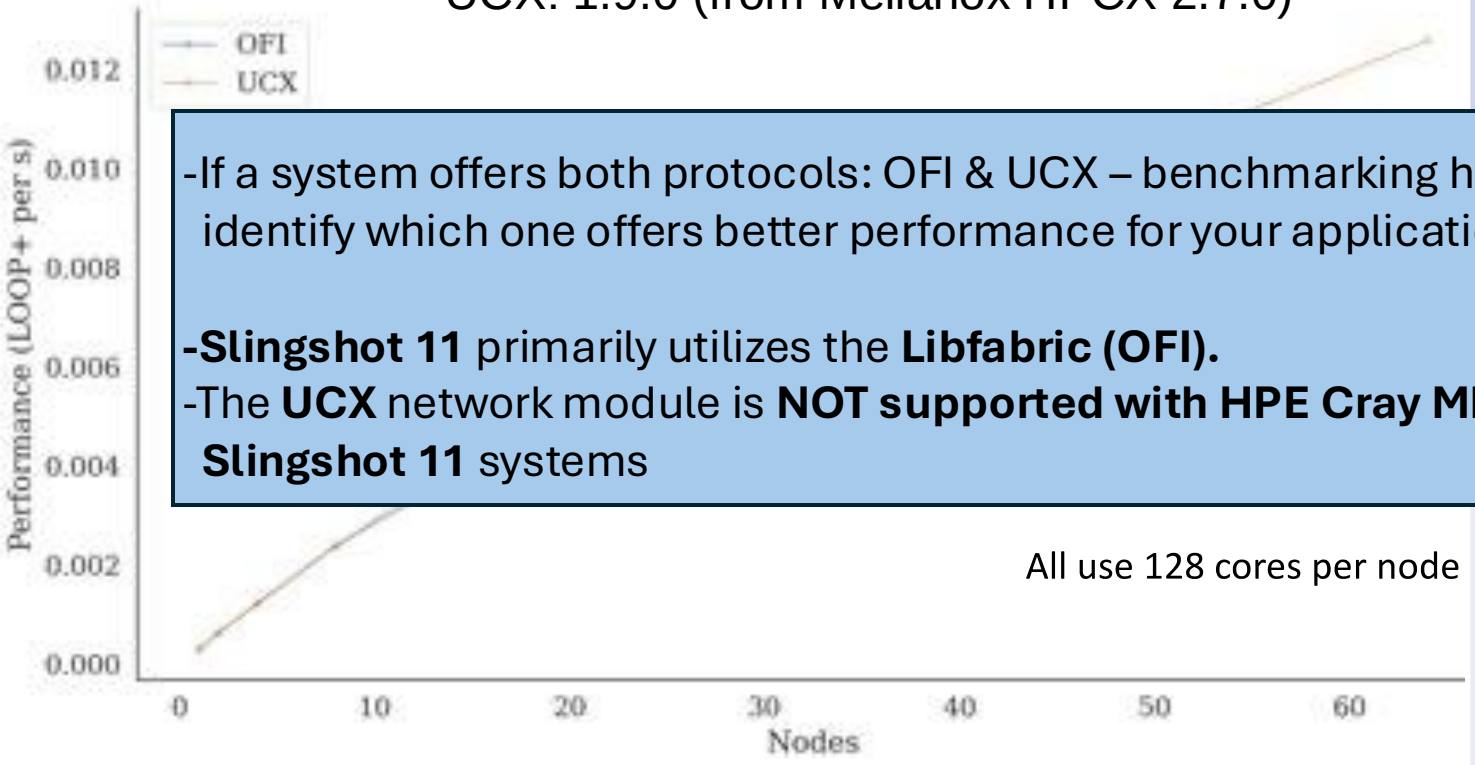
All use 128 cores per node

Nodes	OFI	UCX
8	LOOP+: 419.0s 27.7% MPI 8.3% Alltoallv 7.9% Bcast 5.1% Allreduce 4.6% Barrier	LOOP+: 421.0s 28.8% MPI 9.6% Bcast 5.2% Alltoallv 4.9% Barrier 4.6% Allreduce 4.5% Alltoall
64	LOOP+: 240.5s 60.5% MPI 43.2% Alltoallv 7.2% Bcast 5.6% Allreduce 4.0% Barrier	LOOP+: 79.4s 50.4% MPI 22.0% Bcast 10.2% Barrier 8.5% Allreduce 7.7% Alltoall 2.1% Alltoallv

https://github.com/ARCHER2-HPC/performance_ofi-ucx
OpenFabrics and UCX - Performance on the ARCHER2 HPE Cray EX system
Michael Bareford et al.

Benchmarking VAST: OFI vs UCX

- HPE Cray **Slingshot 10**
- HPE Cray MPICH 8.1.9
- OpenFabrics: 1.11.0.4.71 (libfabric)
- UCX: 1.9.0 (from Mellanox HPCX 2.7.0)



-If a system offers both protocols: OFI & UCX – benchmarking helps identify which one offers better performance for your application.

-**Slingshot 11** primarily utilizes the **Libfabric (OFI).**
-The **UCX** network module is **NOT supported with HPE Cray MPI on Slingshot 11** systems

All use 128 cores per node

Nodes	OFI	UCX
8	LOOP+: 419.0s 27.7% MPI 8.3% Alltoallv 7.9% Bcast 5.1% Allreduce 4.6% Barrier	LOOP+: 421.0s 28.8% MPI 9.6% Bcast 5.2% Alltoallv 4.9% Barrier 4.6% Allreduce 4.5% Alltoall
	LOOP+: 240.5s 60.5% MPI 43.2% Alltoallv 7.2% Bcast 5.6% Allreduce 4.0% Barrier	LOOP+: 79.4s 50.4% MPI 22.0% Bcast 10.2% Barrier 8.5% Allreduce 7.7% Alltoall 2.1% Alltoallv

https://github.com/ARCHER2-HPC/performance_ofi-ucx
OpenFabrics and UCX - Performance on the ARCHER2 HPE Cray EX system
Michael Bareford et al.

Parallel File System - LUSTRE

What's LUSTRE ?

- **Lustre** is parallel **filesystem** software platform designed for **HPC** systems.
- **Lustre** primarily focuses on the **management** of storage devices and file systems.
- Allows **multiple** clients (login nodes, compute nodes, …) to **read/write** data **simultaneously**.
- All clients see a **unified filesystem** (concurrent **coherent** access to files).
- **Lustre** differentiates between **servers that store data** and **servers that store metadata**

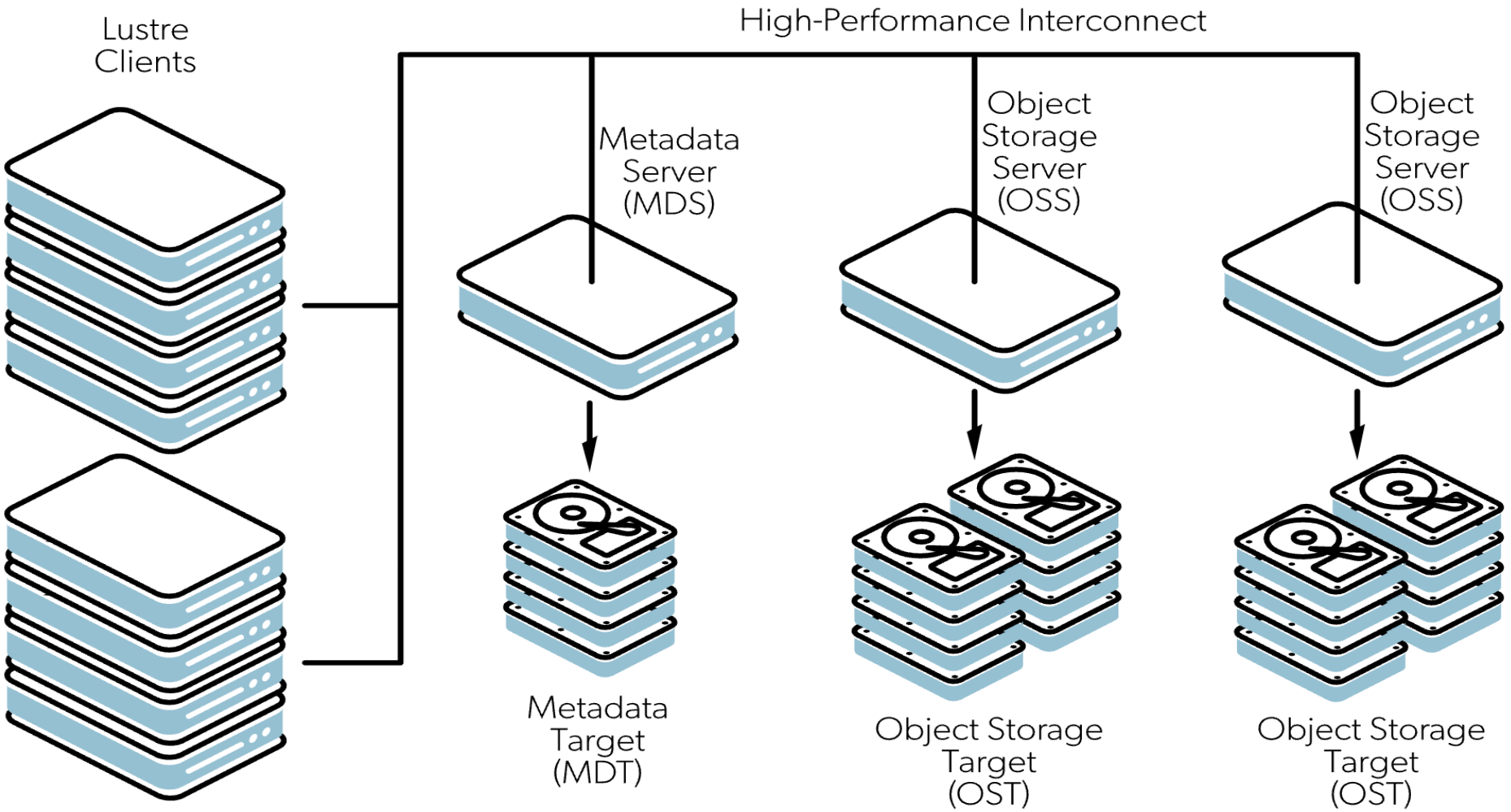
  (e.g. filename, directories, etc)
- **Key feature**: Ability to **scale** both **storage** capacity and **bandwidth** independently.

  Support hundred's of **PB of data storage** and hundreds of **GB/s** simultaneously.

https://wiki.lustre.org/Main_Page

Intel: https://www.lustre.org/wp-content/uploads/architecting-lustre-storage-white-paper.pdf

Architecture of Lustre



Metadata Server (MDS):
manages access to MDT
Handles metadata operations
e.g. file creation, deletion, etc
Each MDS is responsible of one or more MDTs

Metadata Targets (MDT):
stores metadata information
filesystem namespace information (directories, filenames, permissions etc.)

Object Storage Servers (OSS)

Object Storage Targets (OST):
hosts a local file system
stores the actual data
(file content).

Clients: Login nodes, compute nodes
(access global file system)

https://docs.lumi-supercomputer.eu/storage/parallel-filesystems/lustre/

Lustre Capabilities

Lustre is designed for large amounts of data

Storage System Requirements	Lustre File System Capabilities
Large file system	Up to 512 PB for one file system.
Large files	Up to 32 PB for one file.
Global name space	A consistent abstraction of all files allows users to access file system information heterogeneously.
High throughput	2 TB/s in a production system. Higher throughput being tested.
Many files	Up to 10 million files in one directory and 2 billion files in the file system. Virtually unlimited with Distributed Name Space.
Large number of clients accessing the file system in parallel	Up to 25,000+ clients in a production system.
High metadata operation rate	Support for 80,000/s create operations and 200,000/s metadata stat operations.

Intel: https://www.lustre.org/wp-content/uploads/architecting-lustre-storage-white-paper.pdf

Storage System Overview: Example supercomputer LUMI

Divide storage into several filesystems: **/scratch; /flash; /home; /project**; etc
Spaces can be physically or logically separated

Storage Type	Filesystem	Capacity	Bandwidth	Metadata Servers (MDSs)	Object Storage Targets (OSTs)	Storage Type
LUMI-P (Main Storage)	Lustre	20 PB	240 GB/s	1 MDS	32 OSTs	HDD (Spinning Disks)
LUMI-F (Flash Storage)	Lustre	8 PB	1,740 GB/s	2 MDSs	58 OSTs	SSD (Solid-State Drives)

filesystem summary

$ lfs osts

```
hiagueny@uan04:~> lfs df -h | grep 'filesystem_summary'
filesystem_summary:          8.5P        1.5P         6.9P  19% /pfs/lustref1
filesystem_summary:         18.2P        4.4P        13.6P  25% /pfs/lustrep1
filesystem_summary:         18.2P        4.6P        13.4P  26% /pfs/lustrep2
filesystem_summary:         18.2P        7.7P        10.3P  43% /pfs/lustrep3
filesystem_summary:         18.2P        4.7P        13.3P  26% /pfs/lustrep4
```

capacity used available use%

GPUDirect Technology

GPUDirect Technology

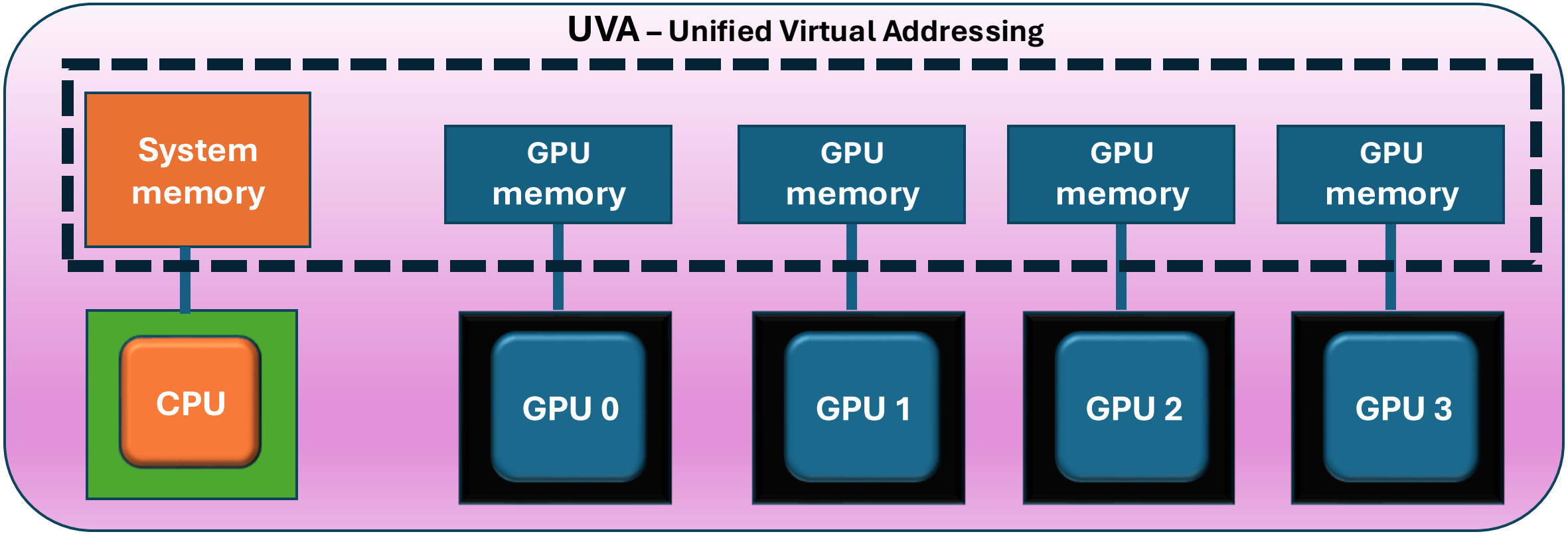
In GPU-aware MPI concept:

MPI library can directly access the GPU memory without necessarily passing by the CPU memory.

Takes advantage of GPUDirect Technology (RDMA and Peer-To-Peer)

Device pointers are passed as arguments to an MPI routine

Single node: MPI with GPU awareness

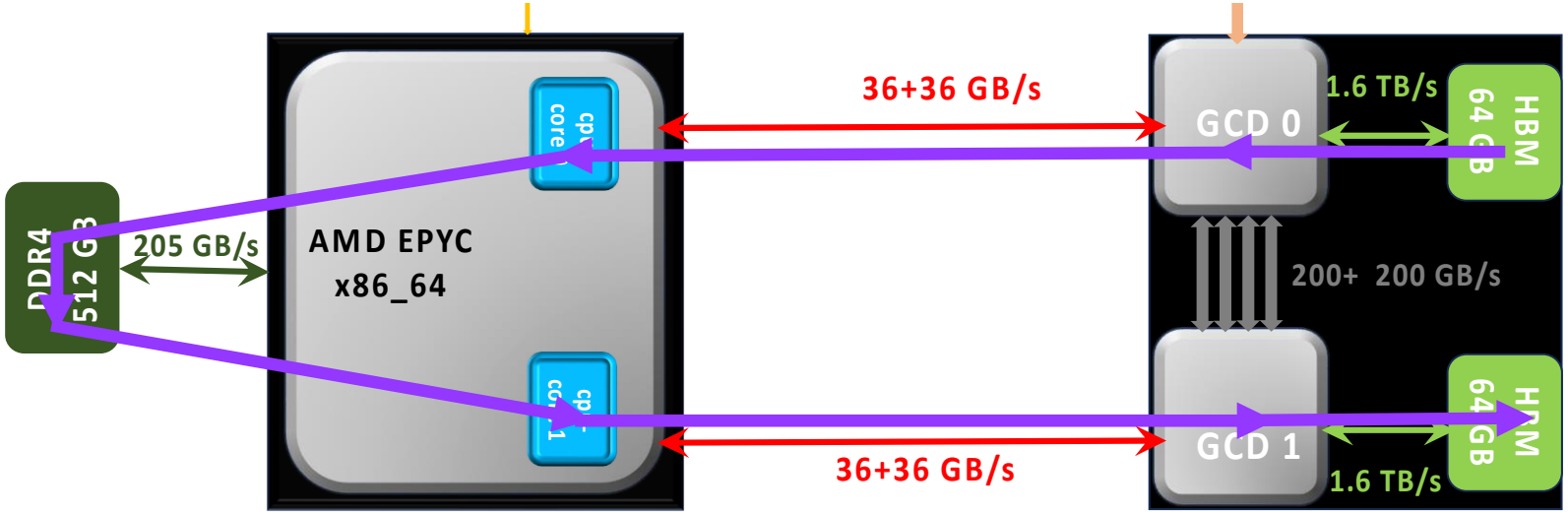


- Device pointers are passed as arguments to an MPI routine
- MPI library will detect that the pointer is a device pointer
- Unified Virtual Addressing (UVA): single virtual memory system for all memory (GPUs, C
- Direct GPU-to-GPU communication

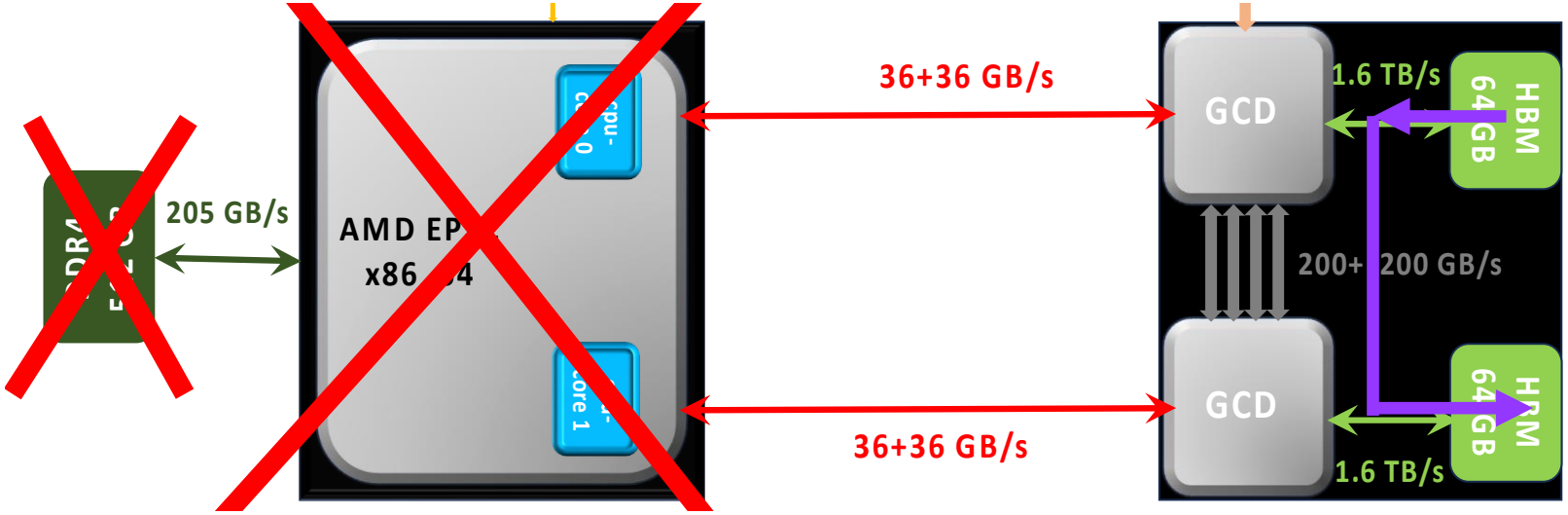
Adapted from: https://developer.nvidia.com/blog/introduction-cuda-aware-mpi/

GPU-aware feature

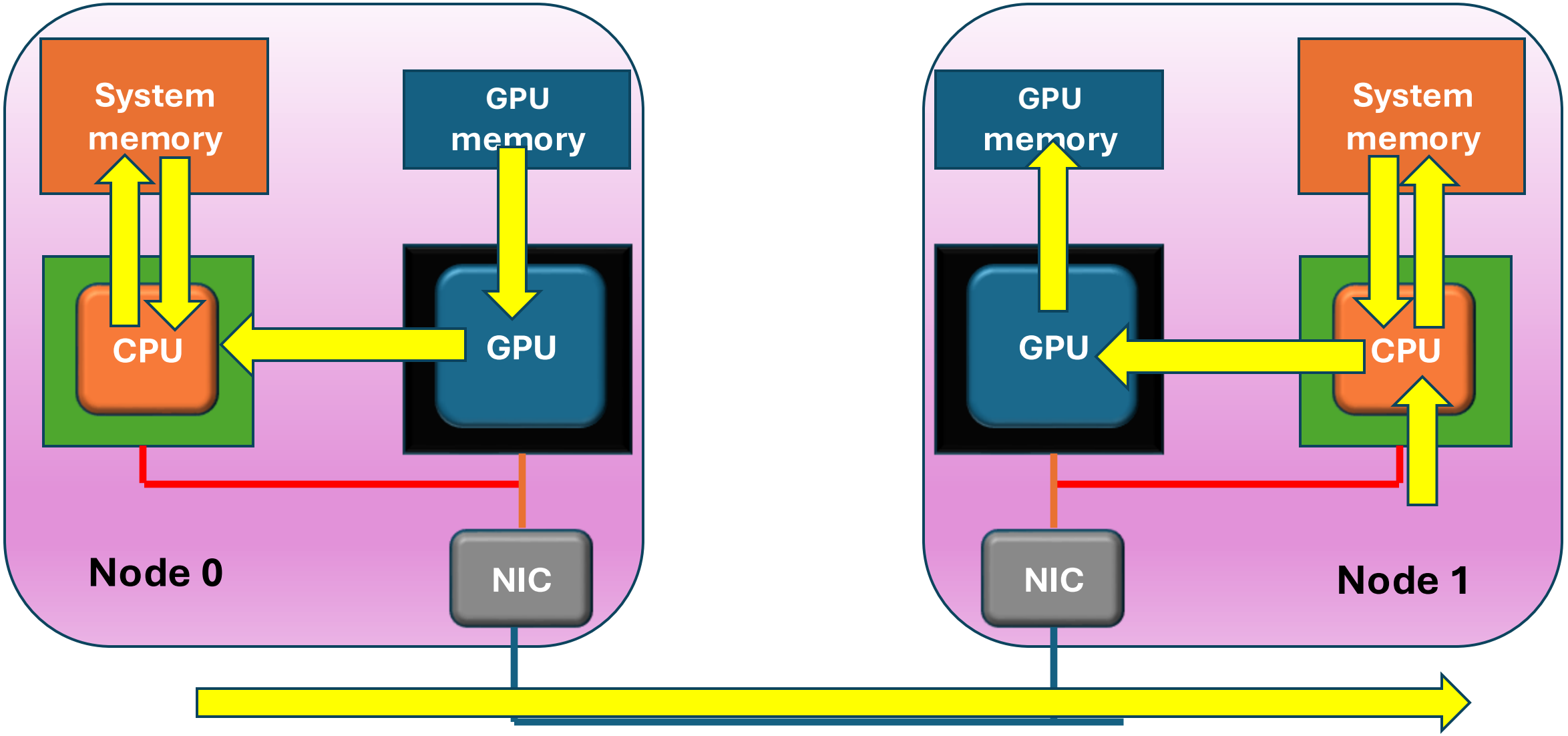
GPU-aware disabled



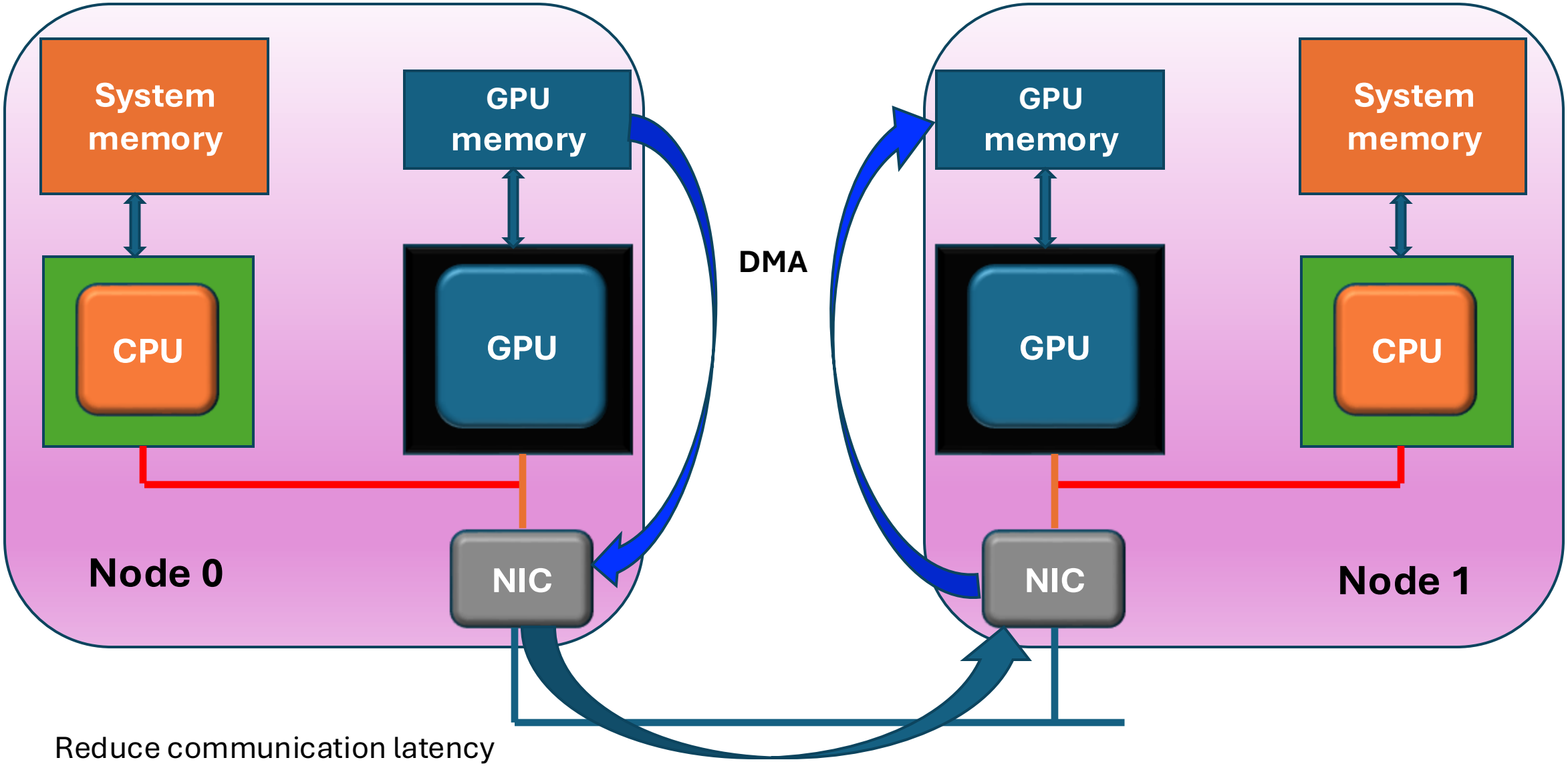
GPU-aware enabled



Multiple nodes: Traditional way of transferring data



Multiple nodes: GPUDirect RDMA (GDR)



Reduce communication latency
Improve performance

Measruing the bandwidth: 2 MPI processes and 2 GPUs, a single Node

GPU-aware feature disabled

```
--Time (s)          0.00002 Data size (B)          128 Bandwidth (GBps)          0.01187
--Time (s)          0.00002 Data size (B)          256 Bandwidth (GBps)          0.02398
--Time (s)          0.00003 Data size (B)          512 Bandwidth (GBps)          0.04033
--Time (s)          0.00003 Data size (B)         1024 Bandwidth (GBps)          0.08017
--Time (s)          0.00002 Data size (B)         2048 Bandwidth (GBps)          0.16789
--Time (s)          0.00003 Data size (B)         4096 Bandwidth (GBps)          0.27093
--Time (s)          0.00003 Data size (B)         8192 Bandwidth (GBps)          0.57978
--Time (s)          0.00003 Data size (B)        16384 Bandwidth (GBps)          1.05693
--Time (s)          0.00004 Data size (B)        32768 Bandwidth (GBps)          1.56283
--Time (s)          0.00006 Data size (B)        65536 Bandwidth (GBps)          2.12329
--Time (s)          0.00010 Data size (B)       131072 Bandwidth (GBps)          2.51498
--Time (s)          0.00019 Data size (B)       262144 Bandwidth (GBps)          2.78053
--Time (s)          0.00036 Data size (B)       524288 Bandwidth (GBps)          2.93066
--Time (s)          0.00066 Data size (B)      1048576 Bandwidth (GBps)          3.15746
--Time (s)          0.00112 Data size (B)      2097152 Bandwidth (GBps)          3.73963
--Time (s)          0.00203 Data size (B)      4194304 Bandwidth (GBps)          4.12282
--Time (s)          0.00438 Data size (B)      8388608 Bandwidth (GBps)          3.82907
--Time (s)          0.00733 Data size (B)     16777216 Bandwidth (GBps)          4.58024
--Time (s)          0.01343 Data size (B)     33554432 Bandwidth (GBps)          4.99637
--Time (s)          0.02581 Data size (B)     67108864 Bandwidth (GBps)          5.19988
--Time (s)          0.05028 Data size (B)    134217728 Bandwidth (GBps)          5.33868
--Time (s)          0.09886 Data size (B)    268435456 Bandwidth (GBps)          5.43059
--Time (s)          0.19354 Data size (B)    536870912 Bandwidth (GBps)          5.54795
--Time (s)          0.38534 Data size (B)   1073741824 Bandwidth (GBps)          5.57296
```

Speed of transferring 1 GB of data between 2 MPI-processes is about 6 GB/s .....Too slow!!

GPU-aware feature enabled

```
0.00001 Data size (B)          128 Bandwidth (GBps)          0.03195
0.00054 Data size (B)          256 Bandwidth (GBps)          0.00095
0.00001 Data size (B)          512 Bandwidth (GBps)          0.16521
0.00003 Data size (B)         1024 Bandwidth (GBps)          0.07636
0.00003 Data size (B)         2048 Bandwidth (GBps)          0.12525
0.00003 Data size (B)         4096 Bandwidth (GBps)          0.24760
0.00053 Data size (B)         8192 Bandwidth (GBps)          0.03119
0.00005 Data size (B)        16384 Bandwidth (GBps)          0.60414
0.00005 Data size (B)        32768 Bandwidth (GBps)          1.22875
0.00005 Data size (B)        65536 Bandwidth (GBps)          2.44771
0.00006 Data size (B)       131072 Bandwidth (GBps)          4.66266
0.00006 Data size (B)       262144 Bandwidth (GBps)          8.59824
0.00007 Data size (B)       524288 Bandwidth (GBps)         14.75897
0.00009 Data size (B)      1048576 Bandwidth (GBps)         23.20806
0.00013 Data size (B)      2097152 Bandwidth (GBps)         32.36915
0.00019 Data size (B)      4194304 Bandwidth (GBps)         45.31348
0.00030 Data size (B)      8388608 Bandwidth (GBps)         55.42490
0.00055 Data size (B)     16777216 Bandwidth (GBps)         61.15308
0.00098 Data size (B)     33554432 Bandwidth (GBps)         68.44087
0.00188 Data size (B)     67108864 Bandwidth (GBps)         71.25060
0.00383 Data size (B)    134217728 Bandwidth (GBps)         70.11301
0.00714 Data size (B)    268435456 Bandwidth (GBps)         75.14631
0.01384 Data size (B)    536870912 Bandwidth (GBps)         77.59176
0.02676 Data size (B)   1073741824 Bandwidth (GBps)         80.25726
```

Speed of transferring 1 GB of data between 2 MPI-processes is about 80 GB/s .....!!

Conclusion

- **Compute node architecture**
  NUMA node: share L3 cache with 8 threads
  CPU-affinity: selecting the closest GPU to a NUMA node for faster communication

- **CPU & GPU architectures**
  Memory hierarchy for optimizing communication latency
  GPUs offer high-throughput with 10000-20000 cores for heavy computing

- **Network Fabric architecture**
  Slingshot Network: Cassini NIC & Rosetta Switch
  NIC handles the processing of network protocol.
  Rosetta Switch uses DragonFly topology to communication between a large number of endpoints.
  Congestion: HPE Slingshot provides advanced congestion control to quickly **detect congestion**.
  RDMA capability to reduce communication latency
  Open-source APIs for **network communications: Libfabric & UCX**

- **Lustre architecture**
  Lustre focuses on the **management** of storage devices and file systems.
  Up to **512 PB** in one filesystem & up to **32 PB** for one file.
  Support of up to **25000+ clients** (access the file system in parallel).

General comments:

- In general: “**idle**” is associated with computation on either CPU or GPU
- But also to communication: **Network idle**
- Overlapping communication and computation
- CPU: Use **nonblocking** Sends and Recvs
- GPU: Use **asynchronous** programming via streams
- Further development of **new hardware features** – but require their use by programmers
- Design your code to take advantage of high-performance interconnects (**GPU-awareness feature**, RDMA)
- **Profiling** to identify bottelnecks
- If a system offers both protocols: **OFI & UCX** – benchmarking helps identify which one offers better performance for your application.
- High-speed Network is designed to handle **large amount of data**.
  It will struggle with small data transfers (the system is not optimized to handle lot of small files)

I stop HERE