

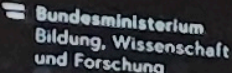
How to play MUSICA

Jan Zabloudil

*BOKU University
VSC Research Center, TU Wien, Austria*



Funded by
the European Union
NextGenerationEU



Bundesministerium
Bildung, Wissenschaft
und Forschung

QUANTUM AUSTRIA

MUSICA - multi-site computer austria

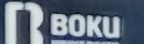
WIEN - LINZ - INNSBRUCK



universität
wien



TECHNISCHE
UNIVERSITÄT
WIEN



BOKU
UNIVERSITÄT
WIEN



TU
Graz



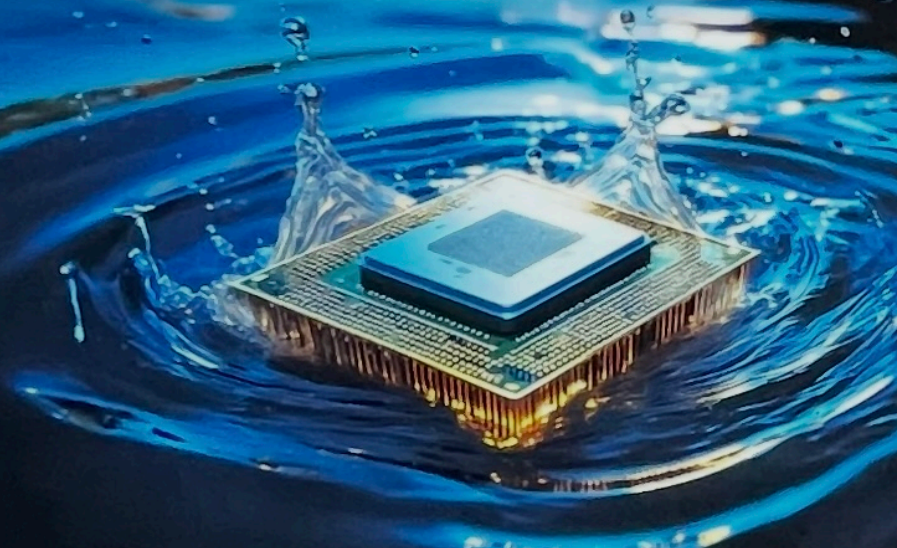
JKU
JOHANNES KEPLER
UNIVERSITÄT
LINZ



universität
innsbruck

Direkt heißwassergekühltes System
mit 1088 NVIDIA GPUs H100/94GB und 84480 CPU-Cores

Gesamtleistung 42 PFlop/s



designed by:



EDV-DESIGN
integration at work
Consulting • Systemintegration • Servertechnologie



Lenovo

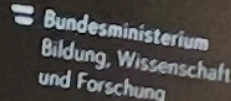
powered by:



NVIDIA



Funded by
the European Union
NextGenerationEU



Bundesministerium
Bildung, Wissenschaft
und Forschung

QUANTUM AUSTRIA

MUSICA - multi-site computer austria

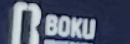
WIEN - LINZ - INNSBRUCK



universität
wien



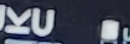
TECHNISCHE
UNIVERSITÄT
WIEN



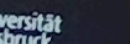
BOKU
UNIVERSITÄT
WIEN



TU
Graz



JKU
JOHANNES KEPLER
UNIVERSITÄT
LINZ



universität
innsbruck

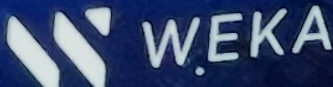
MEGWARE Storage

3 x 4 PB All-Flash WEKA® Data Platform





MEGWARE

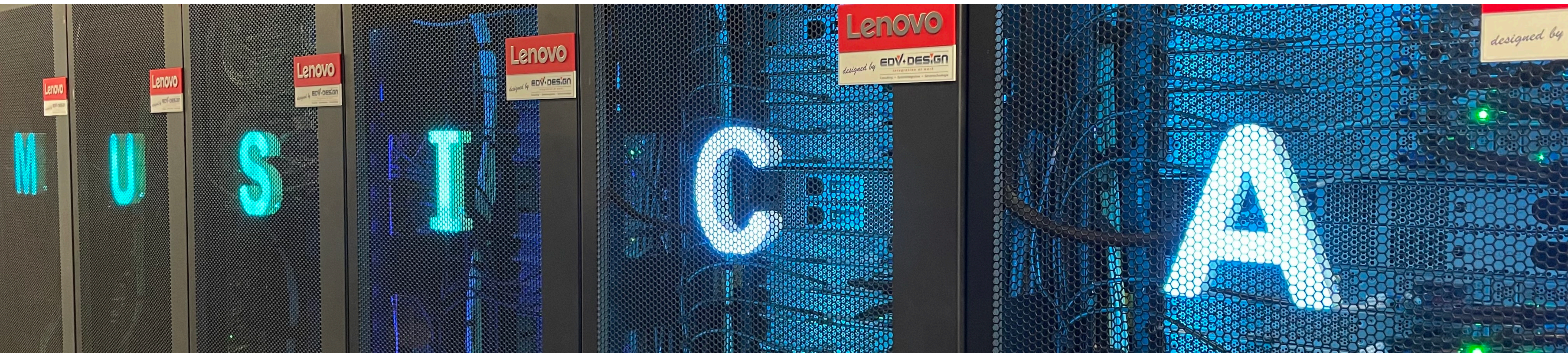


WEKA

Agenda:

- GPU Nodes Performance
- GPU Nodes Energy Efficiency
- GPU Nodes SW Architecture
- GPU Nodes Running Jobs

Performance



MUlti-Site Computer Austria (MUSICA)

272 GPU Nodes:

- * 2x AMD Epyc 9654
- * 4x SXM5 H100
- * NDR

168 CPU Nodes:

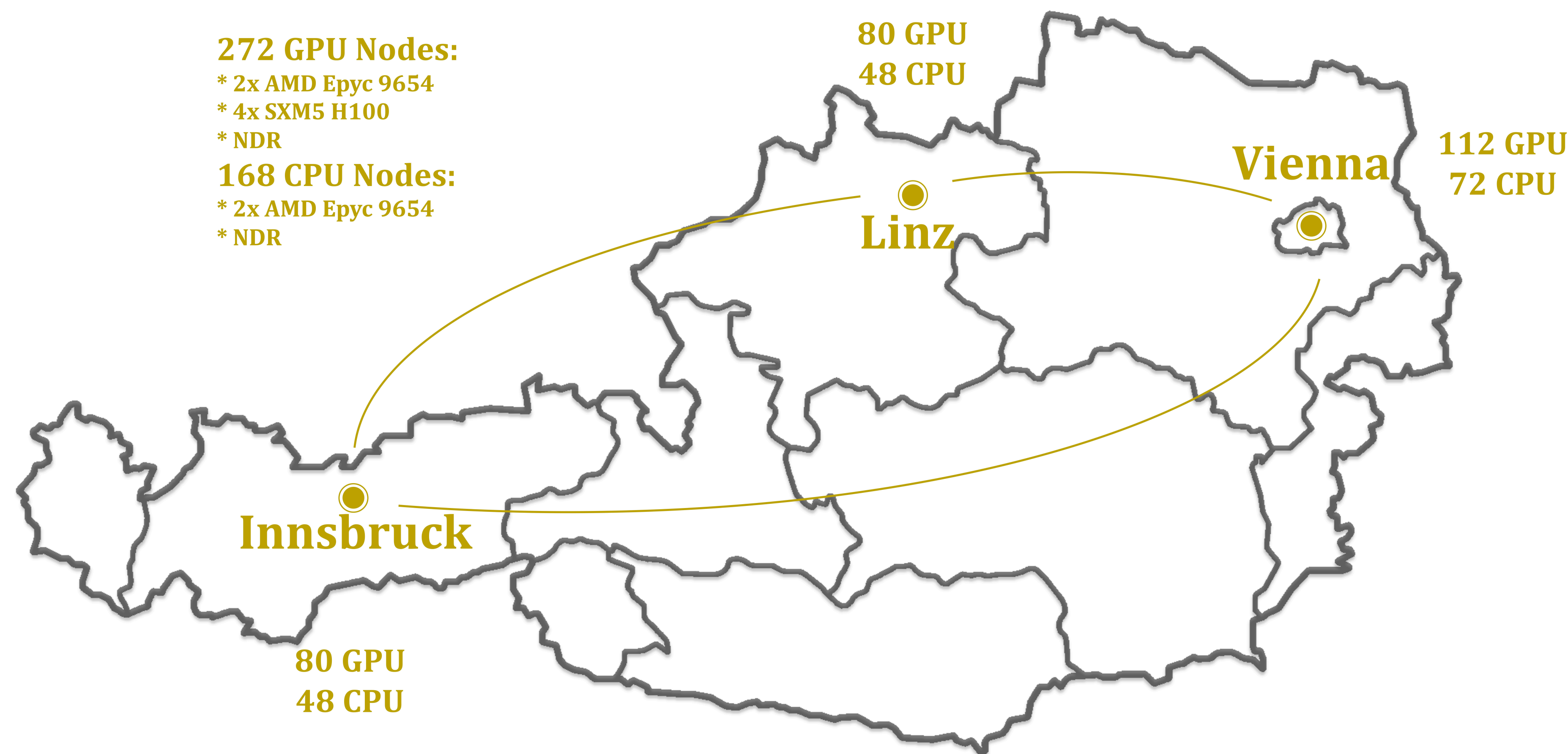
- * 2x AMD Epyc 9654
- * NDR

CPU Nodes

- 32256 CPU Cores
- Per Node Performance:
 - 7.3-11.3 TFlop/s
 - Measured: 8.9 TFlop/s

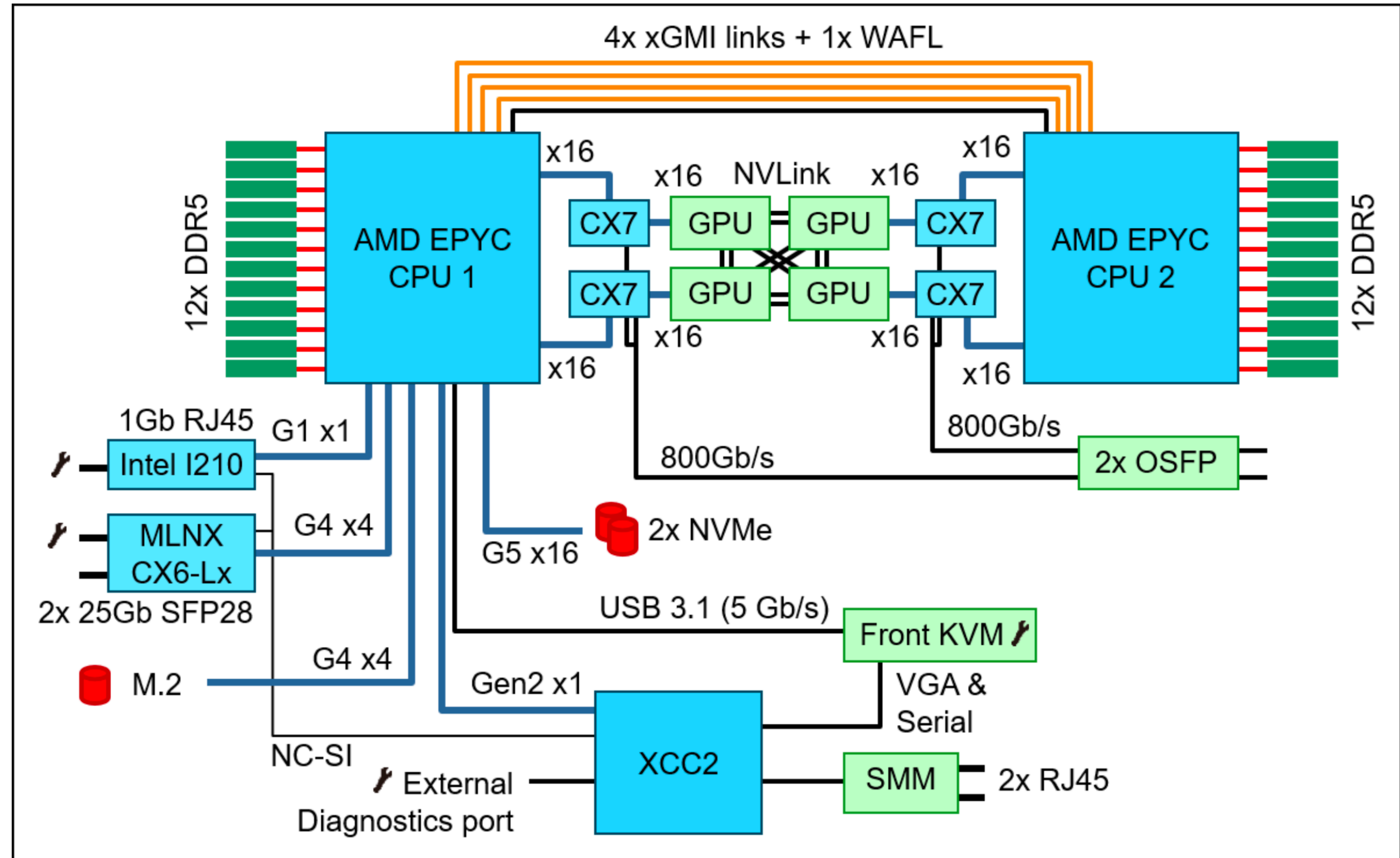
GPU Nodes

- 1088 GPUs
- 52224 CPU Cores
- Per GPU Performance:
 - FP64 Tensor Core 67 TFlop/s
 - Measured: 47.5 TFlops/s

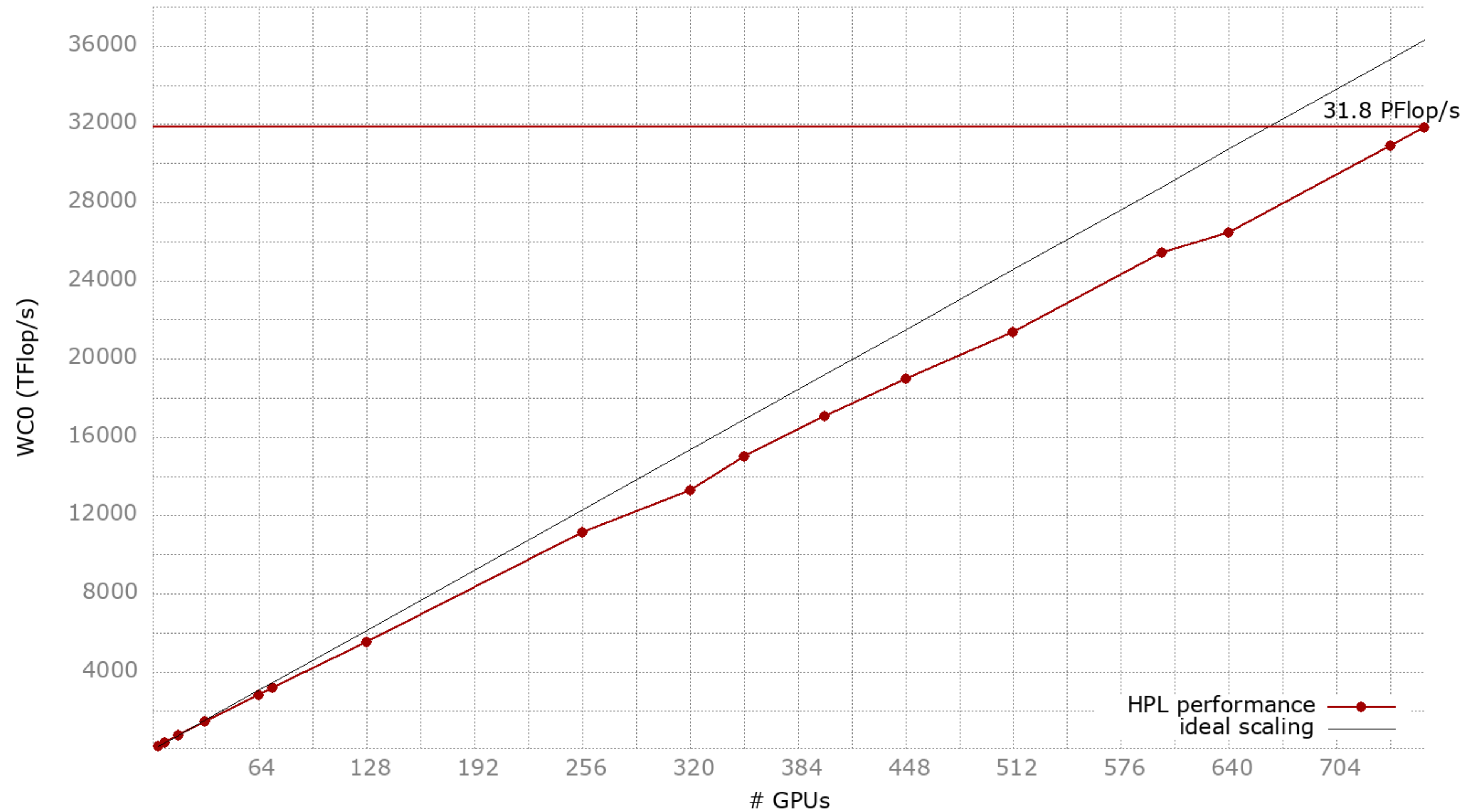


SD665-N v3

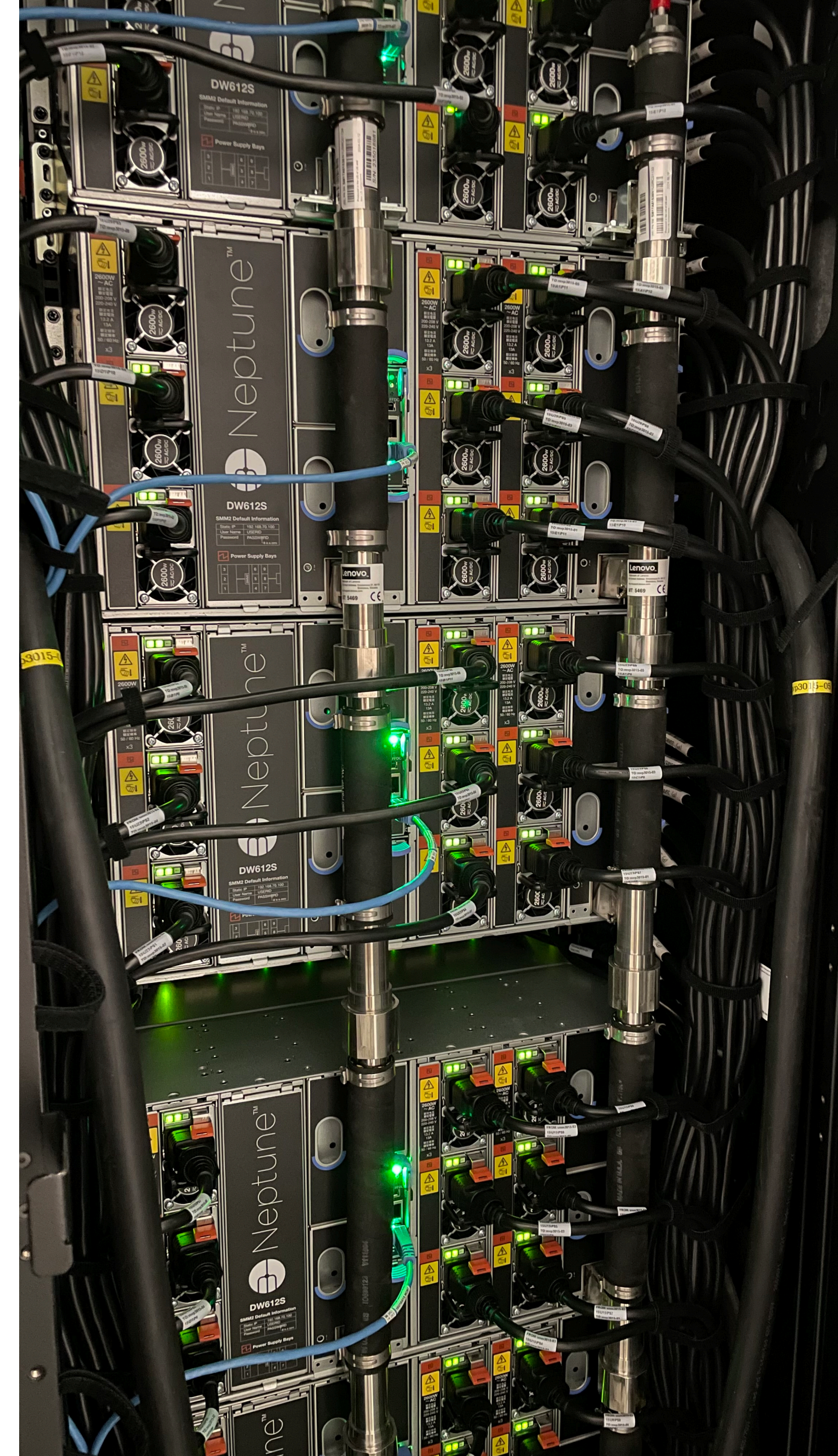
- 2x AMD Epyc 9654, 96c
- SXM5 H100 94GB 700W
- 768 GB RAM
- 4x NDR200



MUSICA GPU Linpack Performance (T_{inlet} 39C, PL 700W)



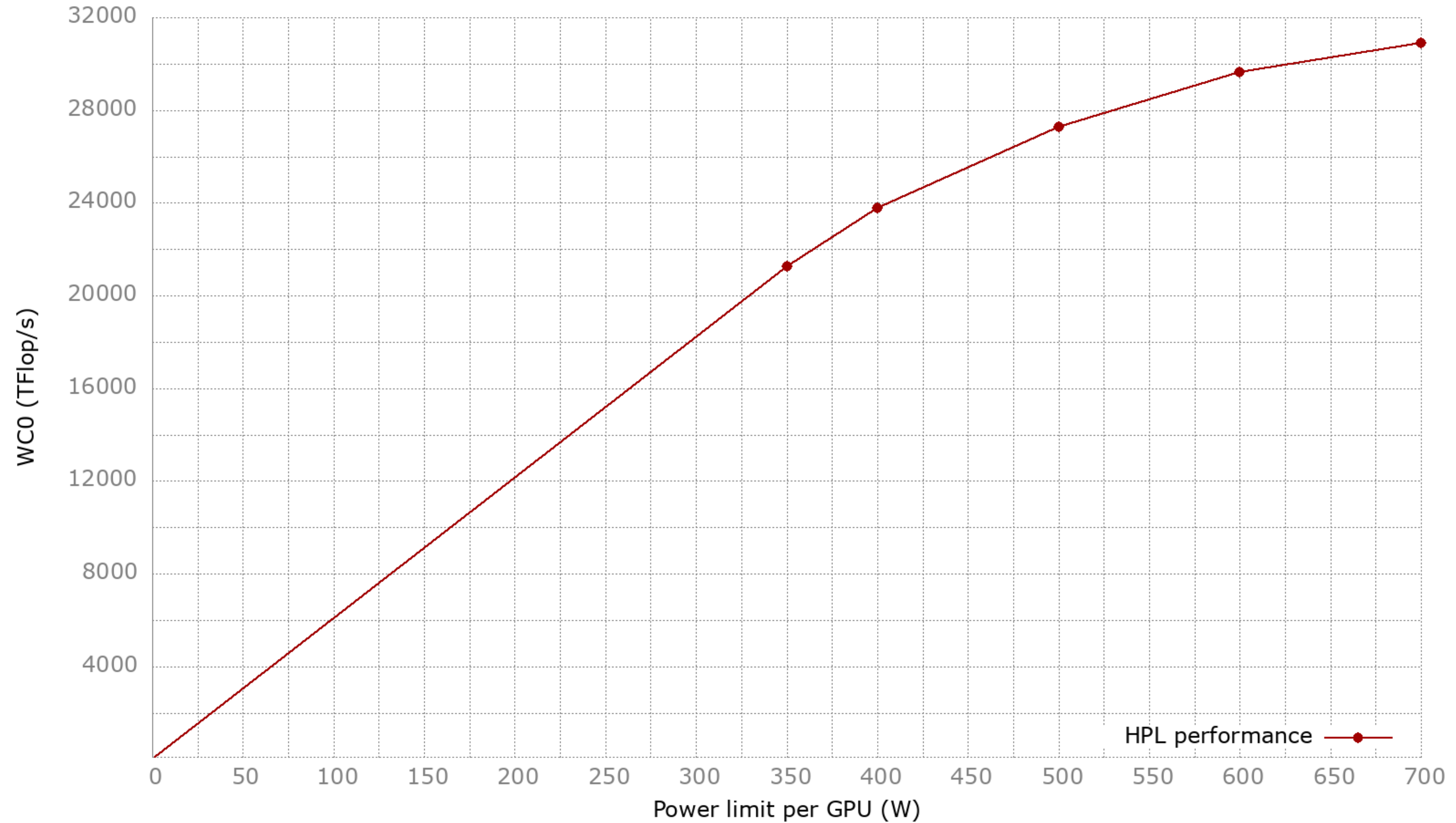
Energy Efficiency



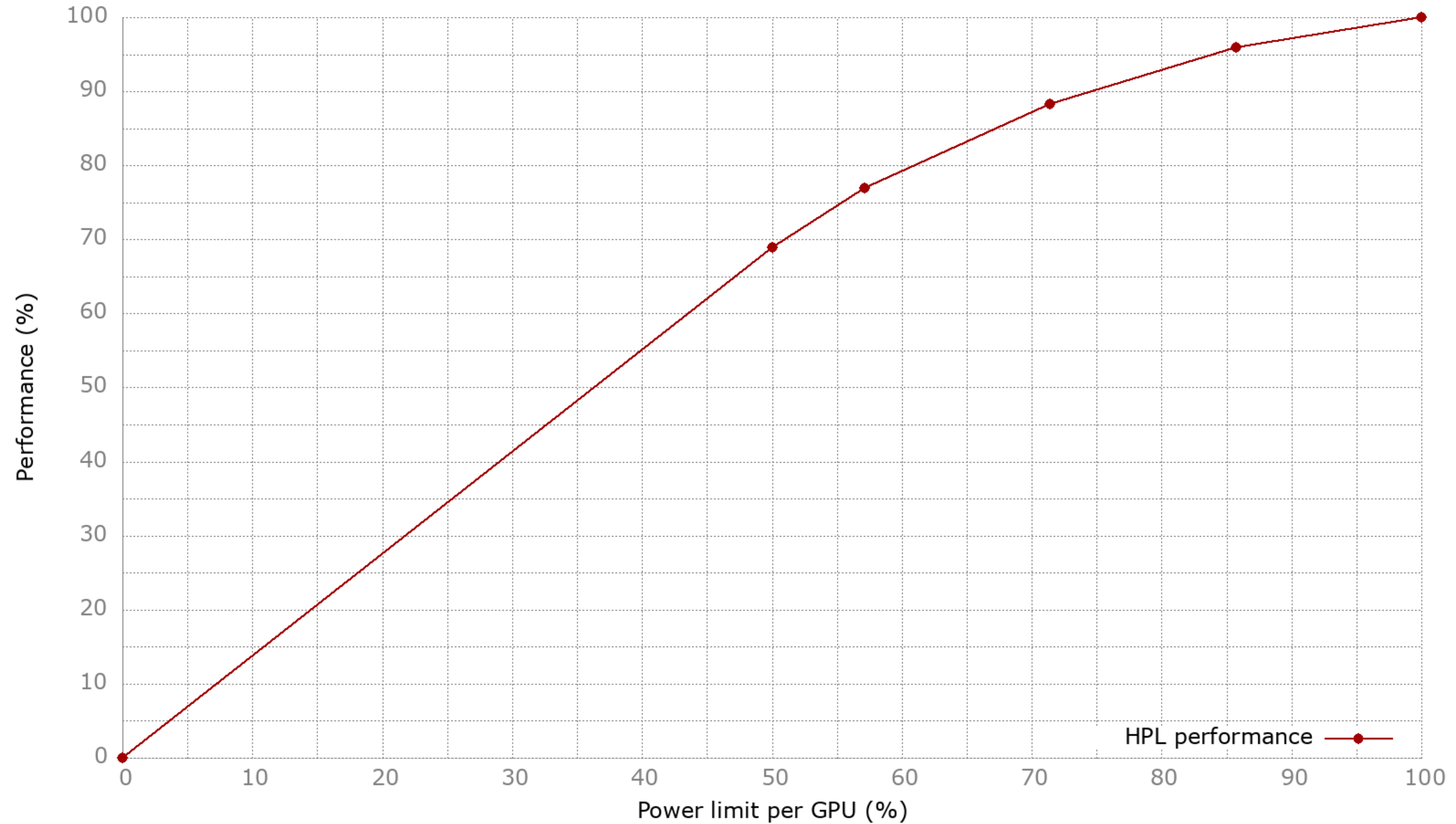
nvidia-smi

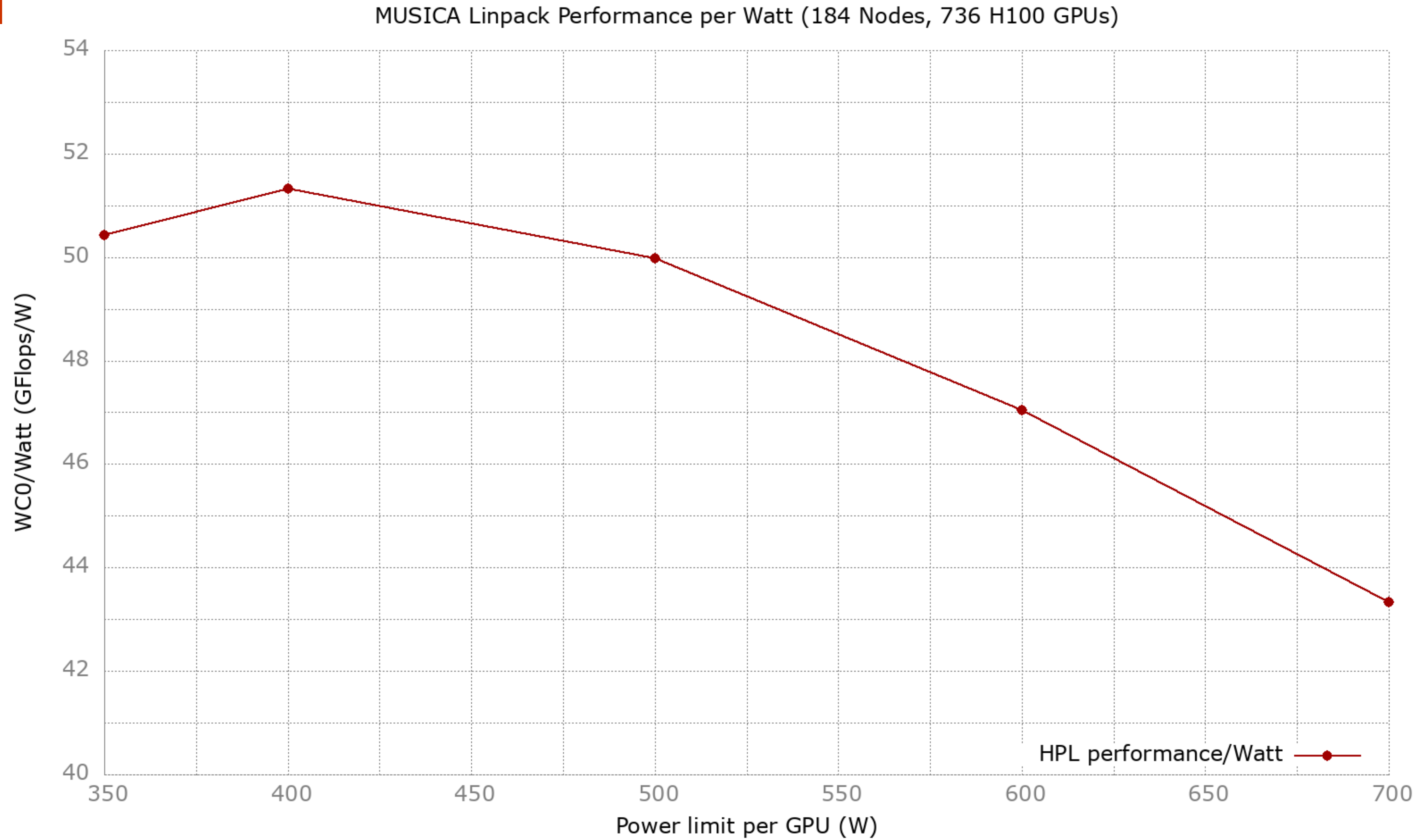
NVIDIA-SMI 560.35.05												Driver Version: 560.35.05				CUDA Version: 12.6											
GPU Name												Persistence-M				Bus-Id				Disp.A				Volatile Uncorr. ECC			
Fan Temp Perf												Pwr:Usage/Cap				Memory-Usage				GPU-Util				Compute M.			
																								MIG M.			
0 NVIDIA H100												On				00000000:06:00.0 Off								0			
N/A 41C P0												69W / 700W				1MiB / 95830MiB				0%				Default			
																								Disabled			
1 NVIDIA H100												On				00000000:26:00.0 Off								0			
N/A 41C P0												70W / 700W				1MiB / 95830MiB				0%				Default			
																								Disabled			
2 NVIDIA H100												On				00000000:A6:00.0 Off								0			
N/A 40C P0												69W / 700W				1MiB / 95830MiB				0%				Default			
																								Disabled			
3 NVIDIA H100												On				00000000:C6:00.0 Off								0			
N/A 40C P0												71W / 700W				1MiB / 95830MiB				0%				Default			
																								Disabled			
Processes:																											
GPU GI CI												PID				Type				Process name				GPU Memory			
																								Usage			
No running processes found																											

MUSICA GPU Linpack Performance (184 Nodes, 736 H100 GPUs)

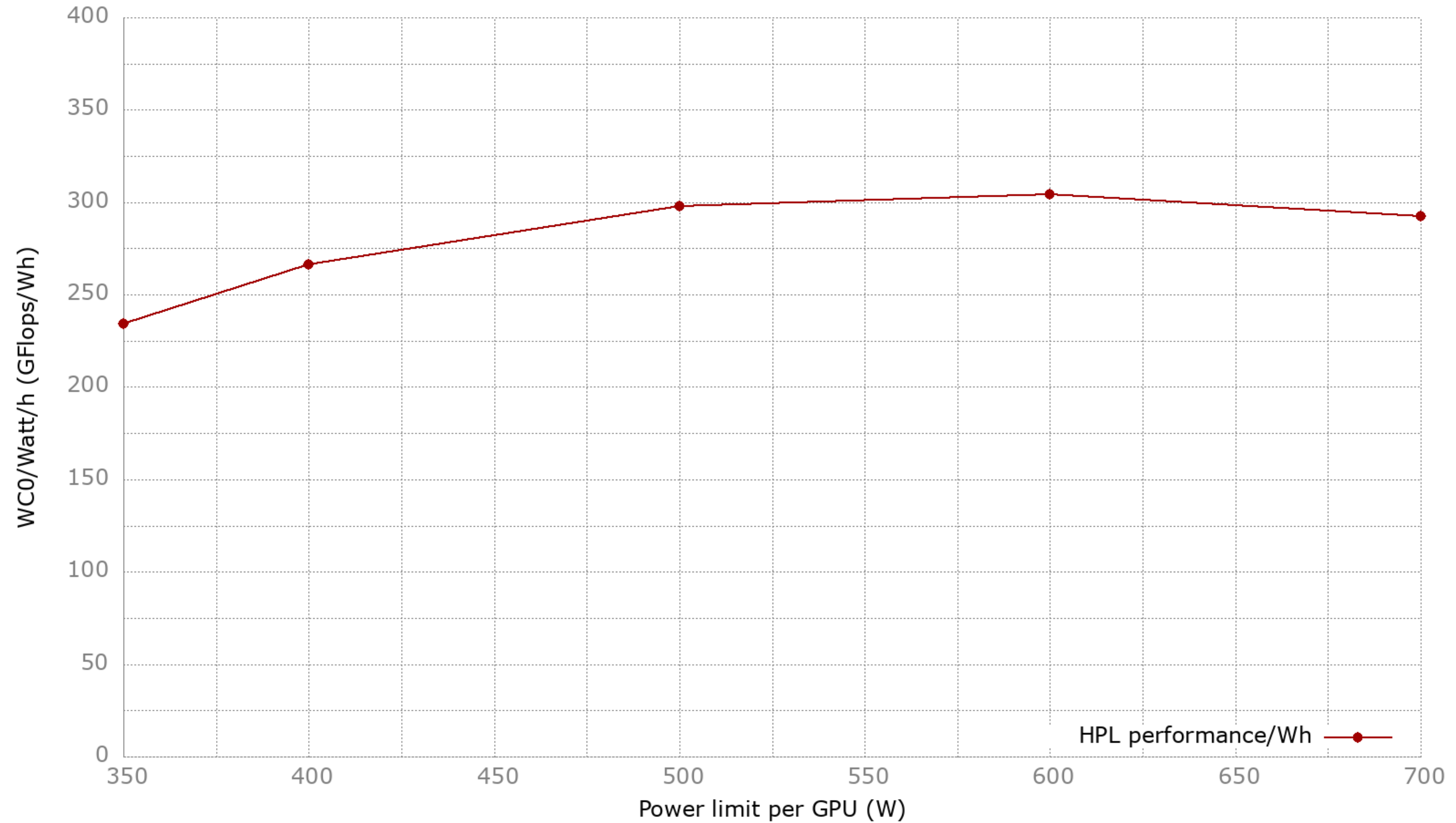


MUSICA GPU Linpack Performance (184 Nodes, 736 H100 GPUs)



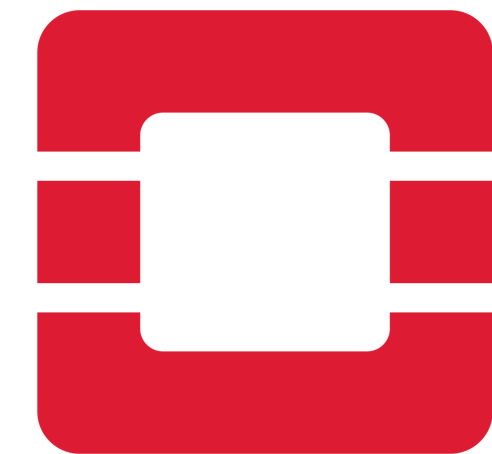


MUSICA Linpack Performance per Wh (184 Nodes, 736 H100 GPUs)

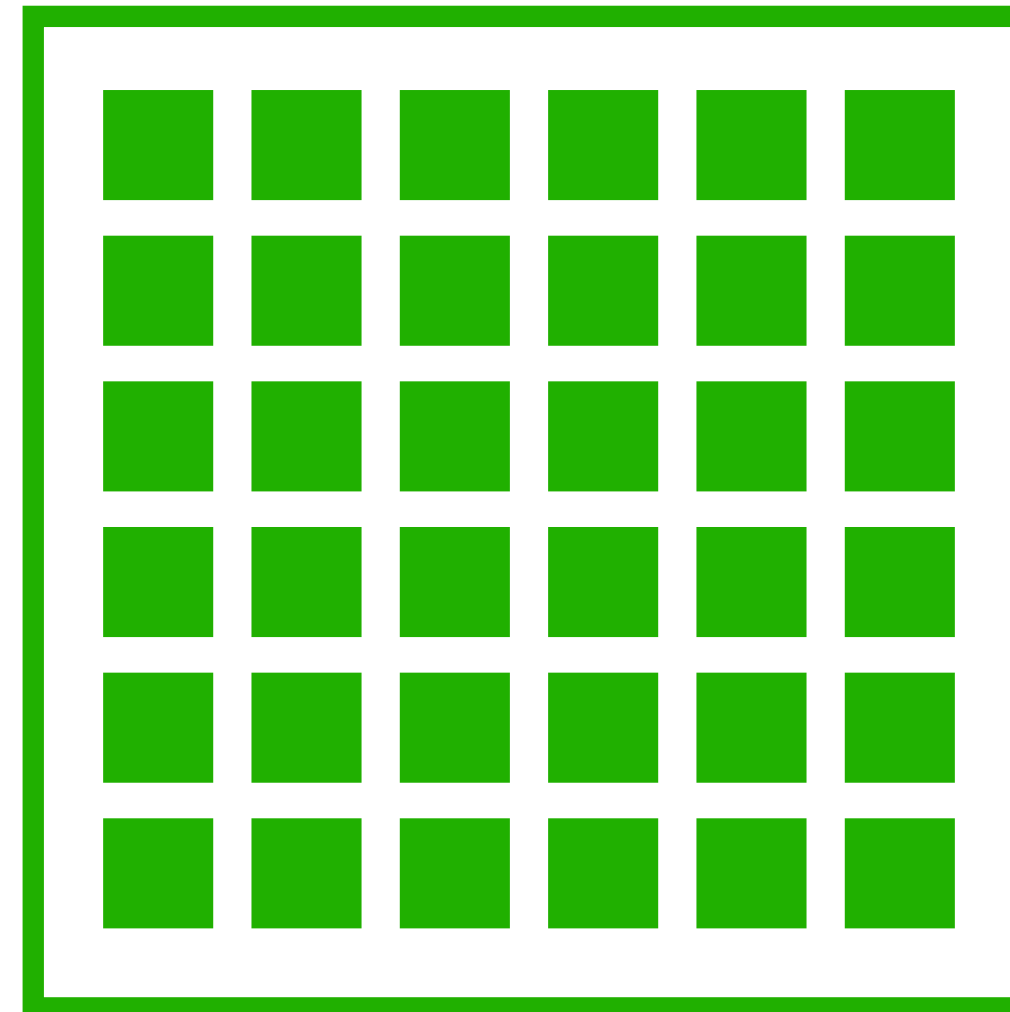




SW architecture

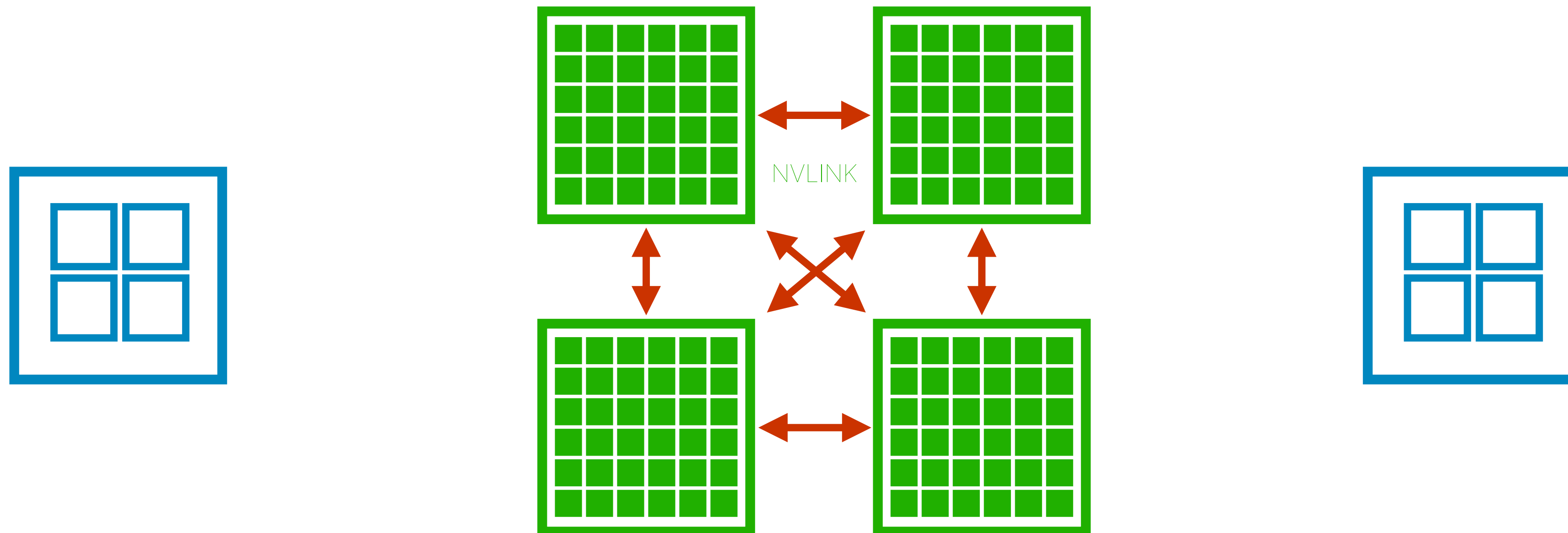


CUDA Driver



CUDA driver is essential for **enabling, managing, and optimizing** the execution of parallel workloads on NVIDIA GPUs, serving as the backbone for CUDA-enabled applications

GPUDirect P2P



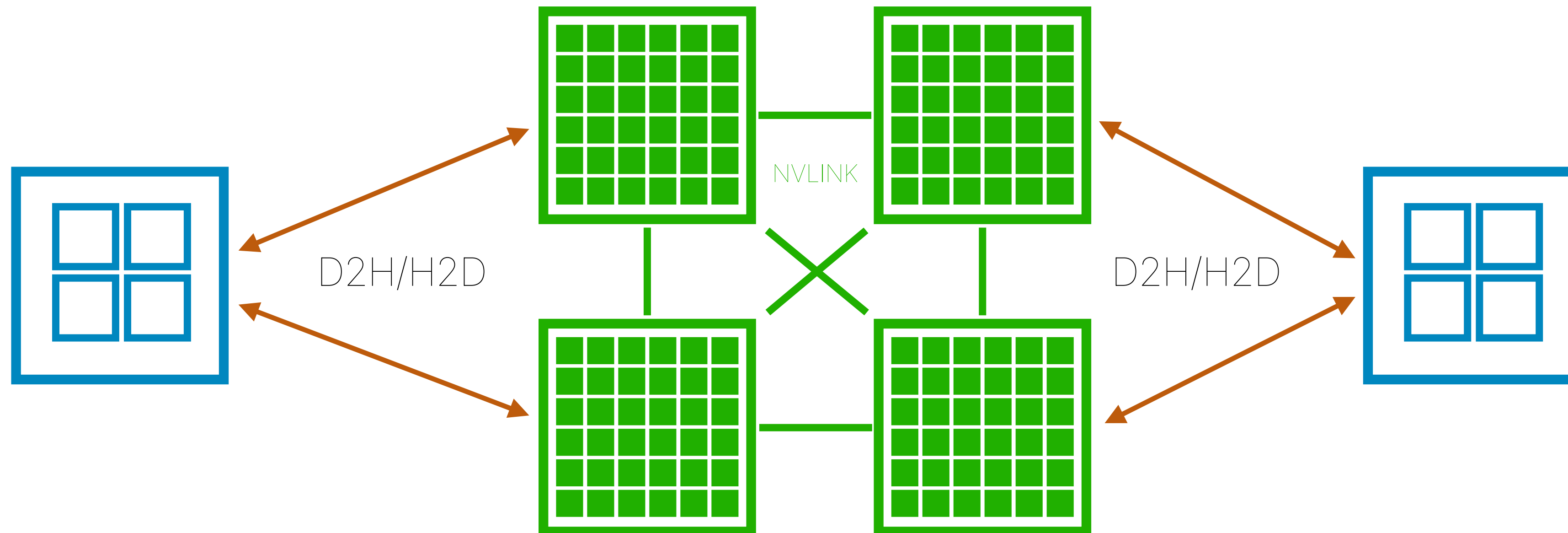
GPUDirect P2P

There is no need to install a separate kernel module for basic P2P functionality between GPUs.

Prerequisites:

- You need a system with two or more compatible NVIDIA GPUs.
- CUDA Toolkit and drivers should be installed.
- GPUs must be on the same PCIe root complex or connected via NVLink.

GDR COPY

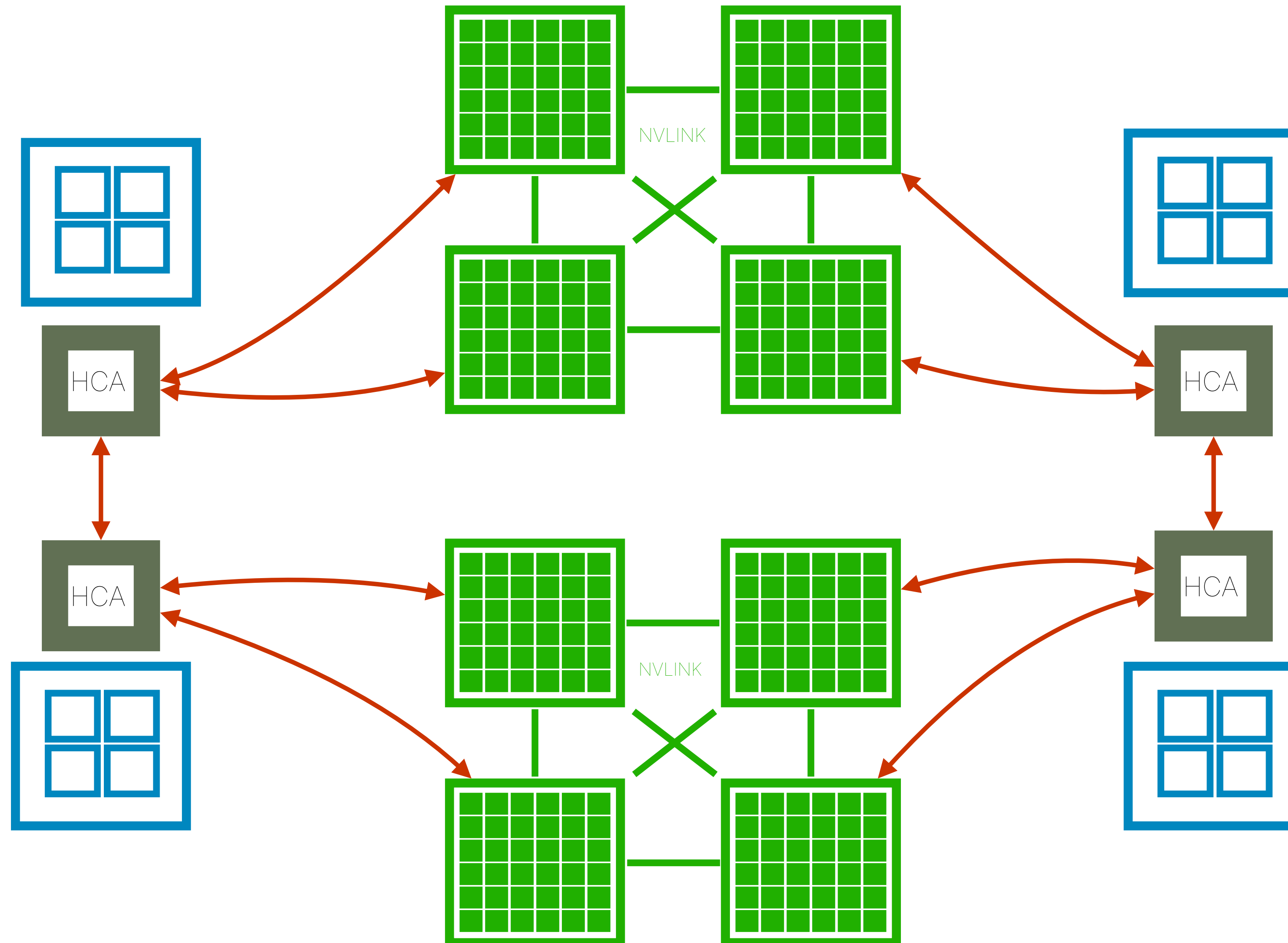


GDR COPY

... is a fast copy library used to **transfer files between the host and the GPU**.
Communication libraries (OpenMPI, UCX) can take advantage of GDRCopy

Prerequisite: DKMS (Dynamic Kernel Module Support) for NVIDIA allows the NVIDIA driver to automatically rebuild its kernel modules when the kernel is updated

Peerdirect/GPUDirect RDMA



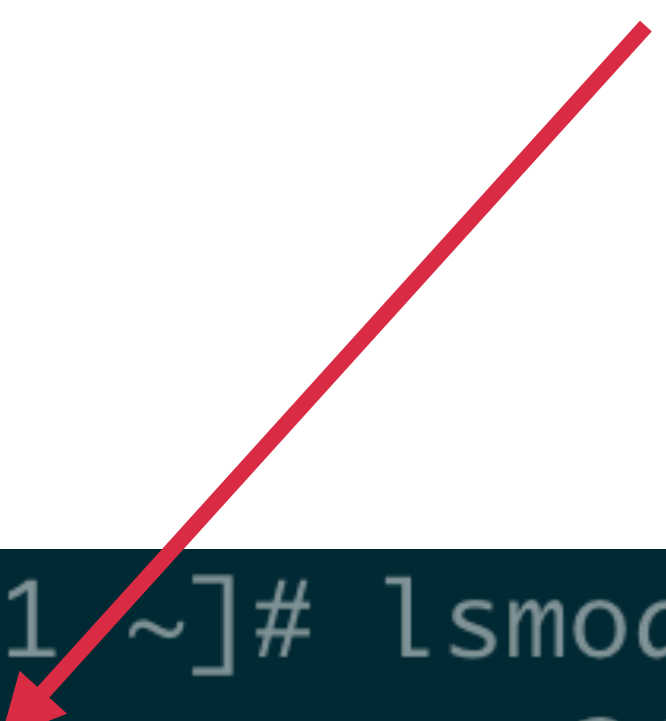
Peerdirect/GPUDirect RDMA

A technology to **improve data transfer efficiency** and **reduce latency** in HPC and data intensive applications.

Enables **direct communication between GPUs and other devices** like NICs, storage controllers, and other GPUs **without the need for CPU** intervention.

It encompasses **different variants** tailored for specific use cases

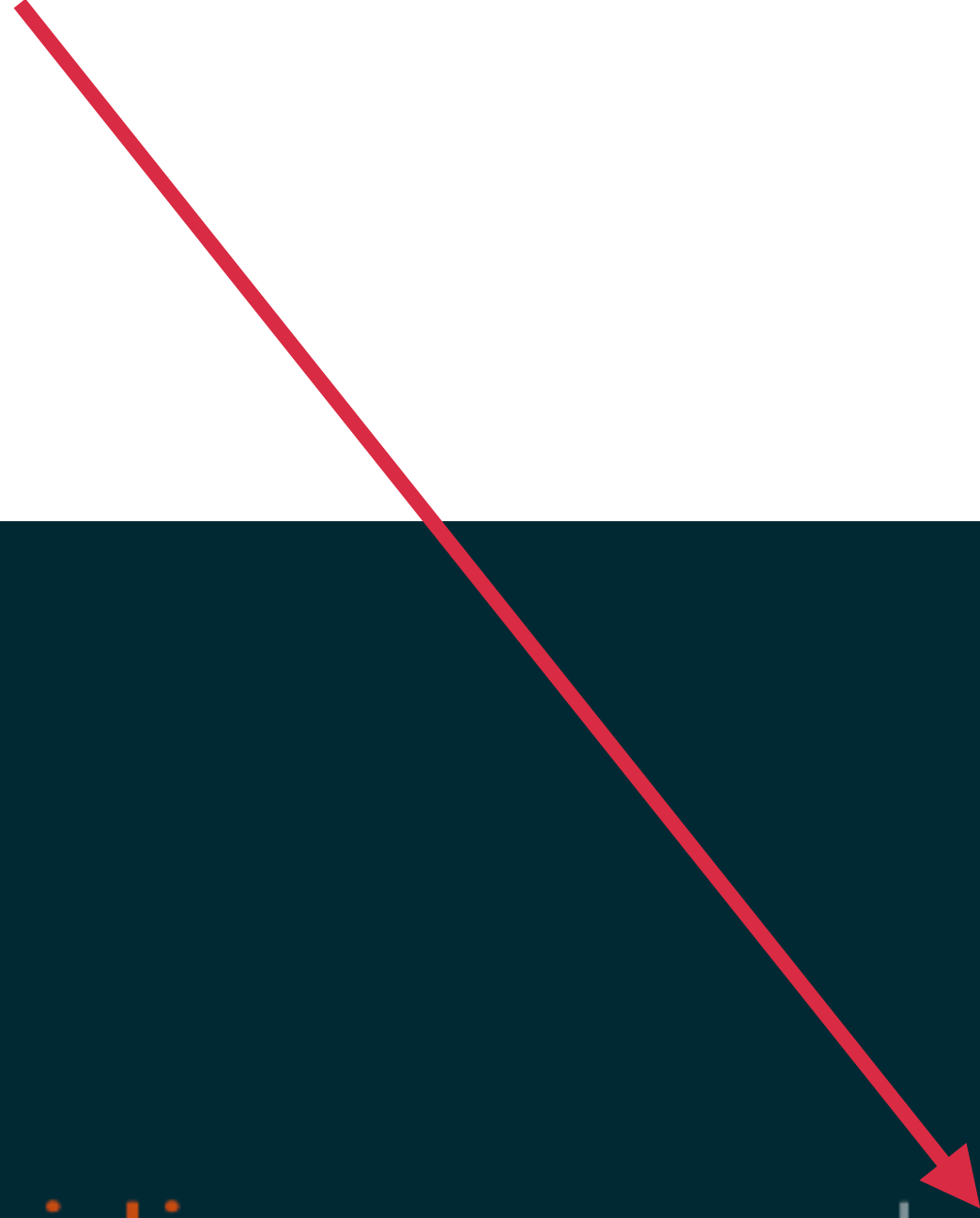
Peerdirect/GPUDirect RDMA



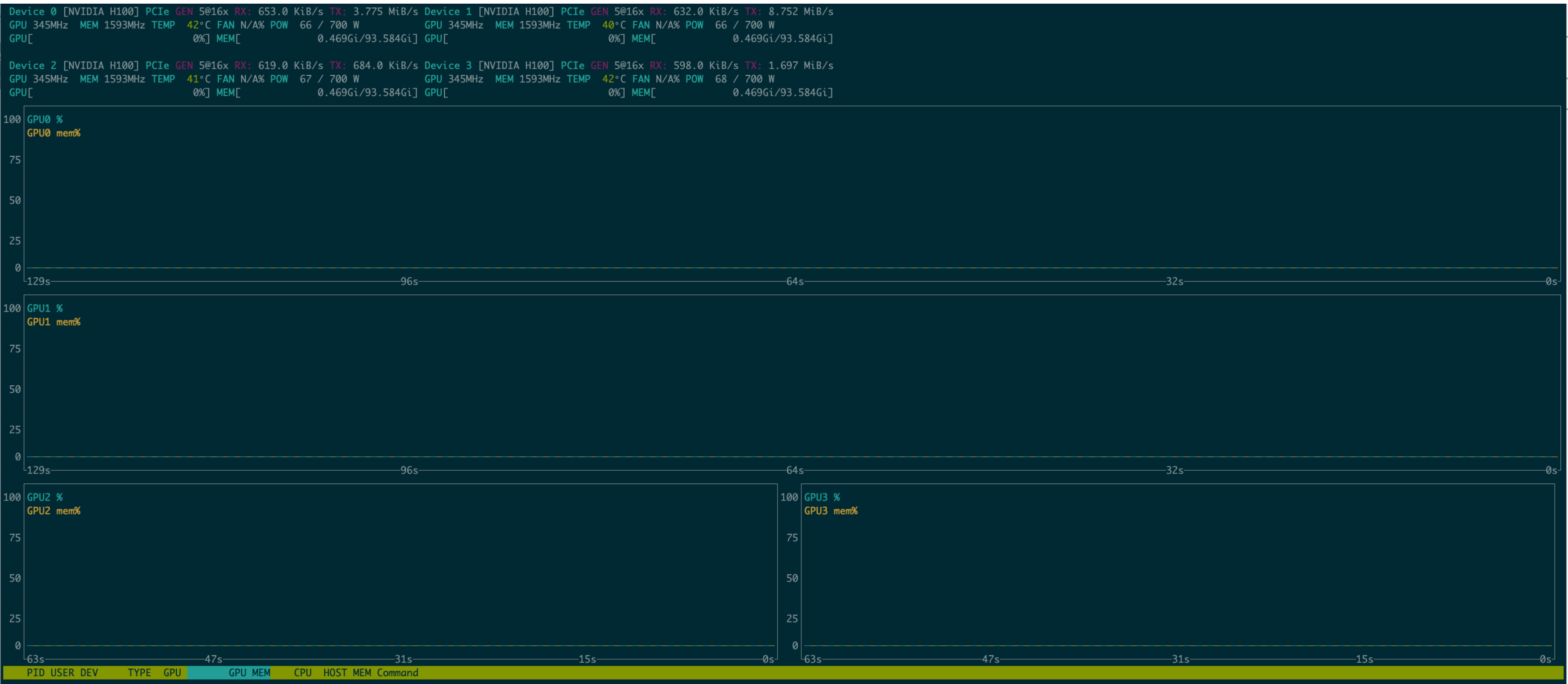
```
[root@n3010-001 ~]# lsmod | grep nvidia
nvidia_peermem      24576  0
nvidia_drm         131072  0
nvidia_modeset     1396736  1 nvidia_drm
nvidia_uvm         6774784  0
nvidia             65937408  4 nvidia_uvm,nvidia_peermem,gdrdrv,nvidia_modeset
```

The NVIDIA GPU driver package provides a kernel module, *nvidia_peermem*, which provides NVIDIA InfiniBand based HCAs **direct peer-to-peer read and write access** to the NVIDIA GPU's video memory. It allows GPUDirect RDMA-based applications to use GPU computing power with the RDMA interconnect **without needing to copy data to host memory**.

GDR Copy



```
[root@n3010-001 ~]# lsmod | grep nvidia
nvidia_peermem      24576  0
nvidia_drm          131072  0
nvidia_modeset      1396736  1 nvidia_drm
nvidia_uvm           6774784  0
nvidia              65937408  4 nvidia_uvm,nvidia_peermem,gdrdrv,nvidia_modeset
```

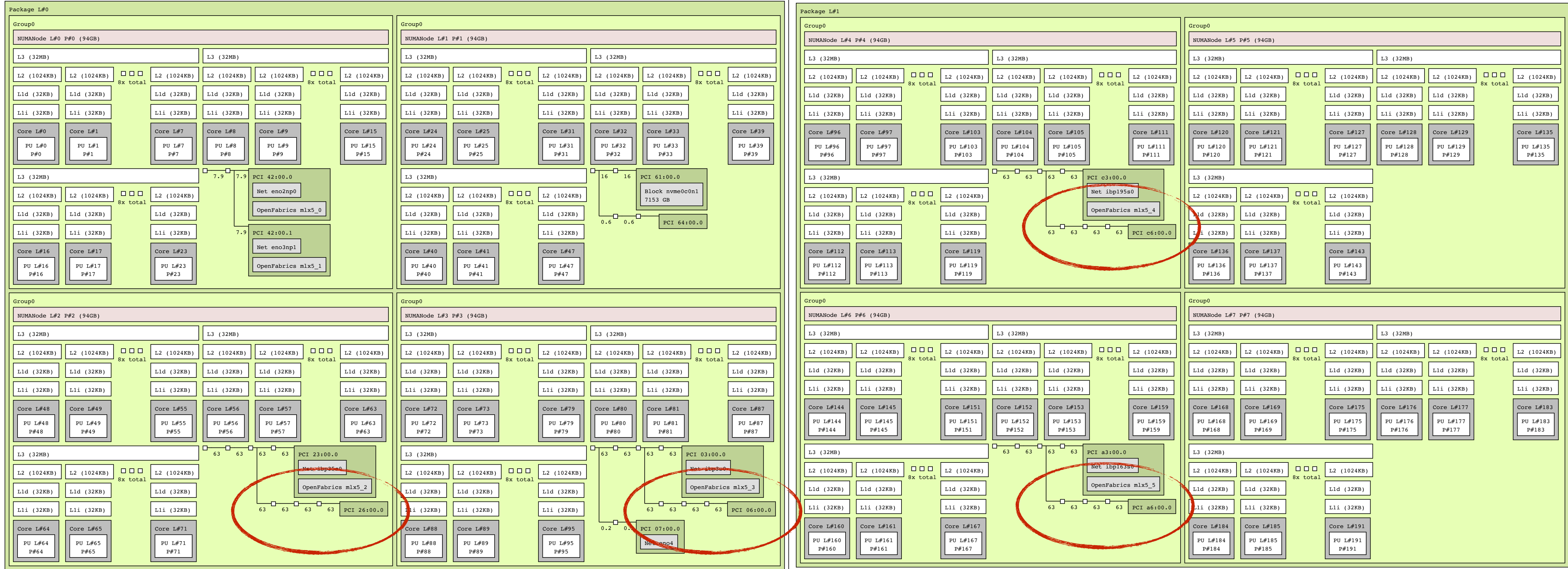



Running Jobs



NUMANode 4

Machine (756GB total)



NUMANode 2

NUMANode 3

NUMANode 6

```
NUMA node(s): 8
NUMA node0 CPU(s): 0-23,192-215
NUMA node1 CPU(s): 24-47,216-239
NUMA node2 CPU(s): 48-71,240-263
NUMA node3 CPU(s): 72-95,264-287
NUMA node4 CPU(s): 96-119,288-311
NUMA node5 CPU(s): 120-143,312-335
NUMA node6 CPU(s): 144-167,336-359
NUMA node7 CPU(s): 168-191,360-383
```

```
[rocky@n3001-001 ~]$ nvidia-smi topo --matrix
```

	<u>GPU0</u>	<u>GPU1</u>	<u>GPU2</u>	<u>GPU3</u>	<u>NIC0</u>	<u>NIC1</u>	<u>NIC2</u>	<u>NIC3</u>	<u>NIC4</u>	<u>NIC5</u>	<u>CPU Affinity</u>	<u>NUMA Affinity</u>	<u>GPU</u>	<u>NUMA</u>	<u>ID</u>
GPU0	X	NV6	NV6	NV6	SYS	SYS	SYS	PXB	SYS	SYS	72-95,264-287	3	N/A		
GPU1	NV6	X	NV6	NV6	SYS	SYS	PXB	SYS	SYS	SYS	48-71,240-263	2	N/A		
GPU2	NV6	NV6	X	NV6	SYS	SYS	SYS	SYS	SYS	PXB	144-167,336-359	6	N/A		
GPU3	NV6	NV6	NV6	X	SYS	SYS	SYS	SYS	PXB	SYS	96-119,288-311	4	N/A		
NIC0	SYS	SYS	SYS	SYS	X	PIX	SYS	SYS	SYS	SYS					
NIC1	SYS	SYS	SYS	SYS	PIX	X	SYS	SYS	SYS	SYS					
NIC2	SYS	SYS	SYS	SYS	SYS	SYS	X	SYS	SYS	SYS					
NIC3	PXB	SYS	SYS	SYS	SYS	SYS	SYS	X	SYS	SYS					
NIC4	SYS	SYS	SYS	PXB	SYS	SYS	SYS	SYS	X	SYS					
NIC5	SYS	SYS	PXB	SYS	SYS	SYS	SYS	SYS	SYS	X					

gres.conf

```
...
NodeName=n[001-100] Count=1 Cores=72-95 File=/dev/nvidia0 Name=gpu Type=h100
NodeName=n[001-100] Count=1 Cores=48-71 File=/dev/nvidia1 Name=gpu Type=h100
NodeName=n[001-100] Count=1 Cores=144-167 File=/dev/nvidia2 Name=gpu Type=h100
NodeName=n[001-100] Count=1 Cores=96-119 File=/dev/nvidia3 Name=gpu Type=h100
...
```

SLURM

```
#!/bin/bash
#SBATCH -N 184
#SBATCH --job-name "hpl.184N.736GPU"
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=24
#SBATCH --gres=gpu:4

export UCX_IB_GPU_DIRECT_RDMA=1
export UCX_TLS=rc,self,cma,rc_mlx5,cuda_copy,cuda_ipc,gdr_copy

CONTAINER=./hpc-benchmarks:24.09-hpl.sif
MOUNT=$PWD
SCRIPT=./hpl.sh

srun \
  --cpu-bind=none \
  --mpi=pmix singularity exec \
  --nv \
  -B ${MOUNT} ${CONTAINER} ${SCRIPT}
```

¹ <https://catalog.ngc.nvidia.com/orgs/nvidia/containers/hpc-benchmarks>

² <https://docs.nvidia.com/nvidia-hpc-benchmarks/overview.html>

³ <https://hpcadvisorycouncil.atlassian.net/wiki/spaces/HPCWORKS/pages/2791440385/GPUDirect+Benchmarking>

NUMA Node Bindings

```
case ${OMPI_COMM_WORLD_LOCAL_RANK} in
0)
export UCX_NET_DEVICES=mlx5_2:1,mlx5_3:1;
export CUDA_VISIBLE_DEVICES=0;
numactl --cpunodebind=3 --membind=3  $APP
;;
1)
export UCX_NET_DEVICES=mlx5_2:1,mlx5_3:1;
export CUDA_VISIBLE_DEVICES=1;
numactl --cpunodebind=2 --membind=2  $APP
;;
2)
export UCX_NET_DEVICES=mlx5_4:1,mlx5_5:1;
export CUDA_VISIBLE_DEVICES=2;
numactl --cpunodebind=6 --membind=6 $APP
;;
3)
export UCX_NET_DEVICES=mlx5_5:1,mlx5_4:1;
export CUDA_VISIBLE_DEVICES=3;
numactl --cpunodebind=6 --membind=6 $APP
;;
esac
```