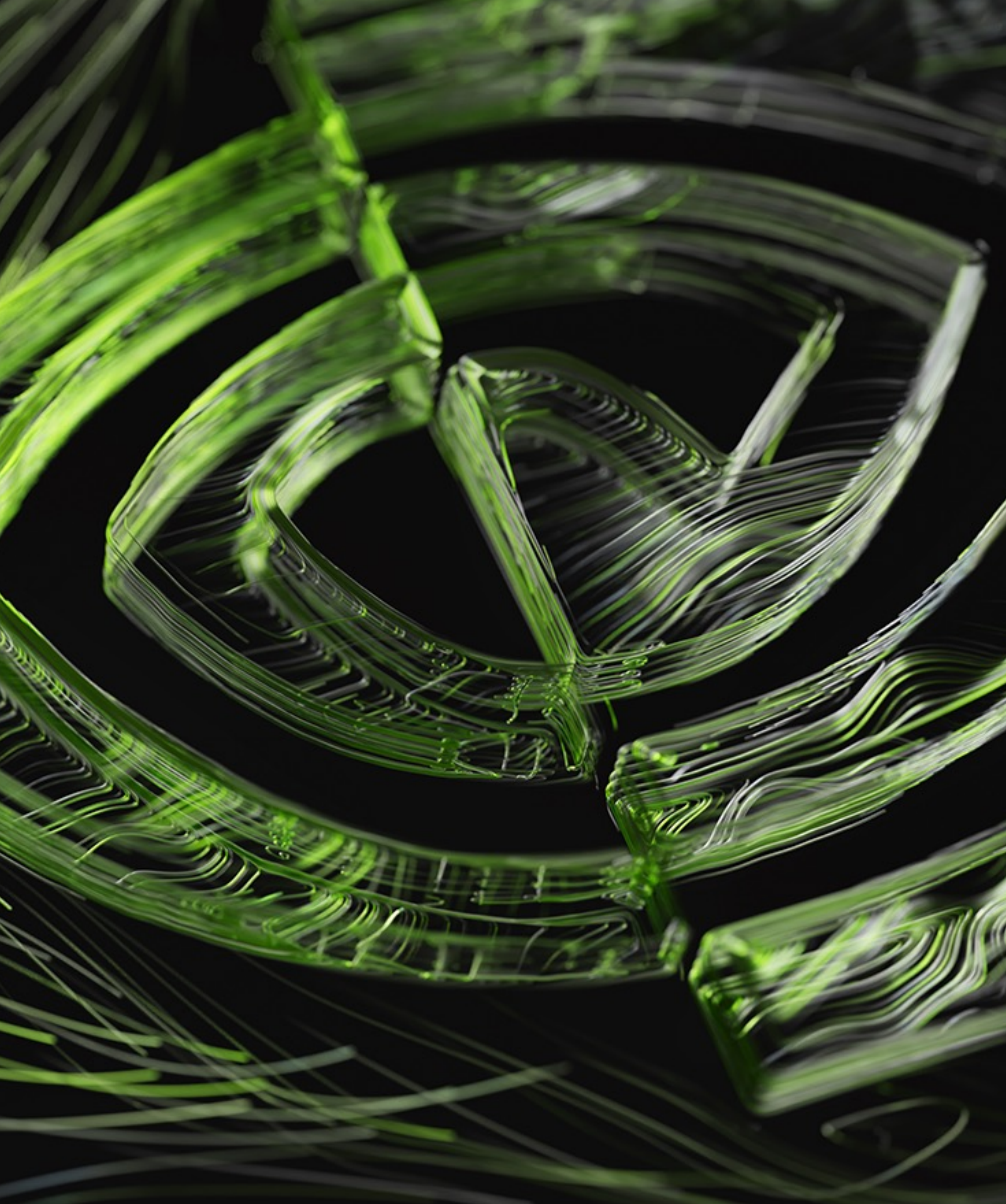




Mixed-precision scientific computing with Tensor Cores on NVIDIA GPUs: Exceeding the performance characteristics of single precision while maintaining numerical accuracy

Harun Bayraktar, Director of Engineering

HPC on Heterogenous Hardware (H3) Workshop @ ISC23 – May 2023




Agenda

- Part 1: Heterogeneous Computing

- Part 2: Mixed-Precision scientific computing with Tensor Cores

- Closing Remarks

The background features a dark, almost black, space filled with numerous thin, glowing green lines that create a sense of motion and depth. Some lines are straight and parallel, while others curve or form larger, more complex shapes. The overall effect is reminiscent of a data visualization or a network of connections.

Part 1: Heterogeneous Computing

Heterogeneous Hardware

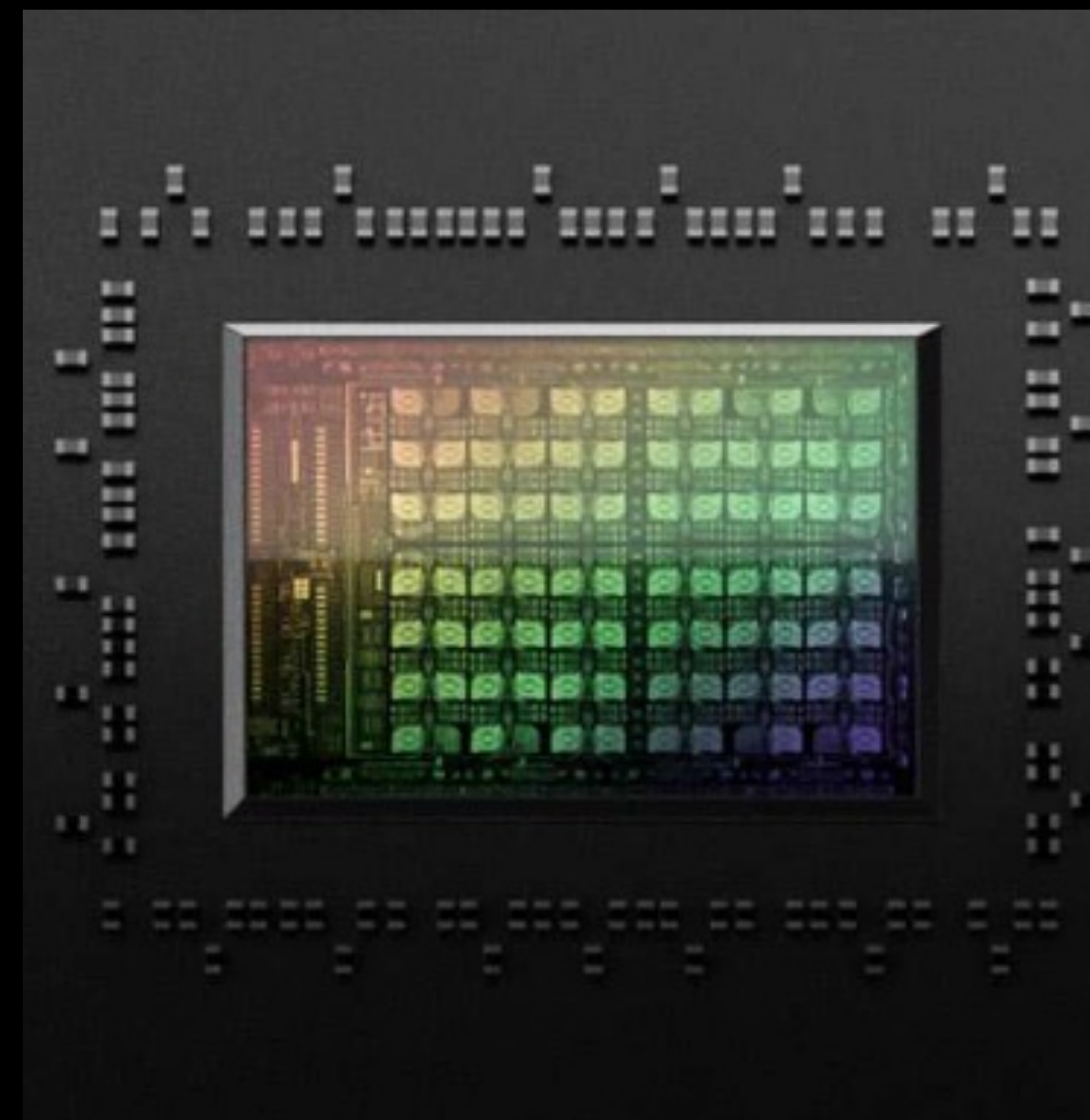
Specialization of hardware components

- heterogeneous
- adjective US: /,het̩.ə.roʊ'dʒiː.ni.əs/ UK: /,het.ər.ə'dʒiː.ni.əs/
- consisting of parts or things that are very different from each other

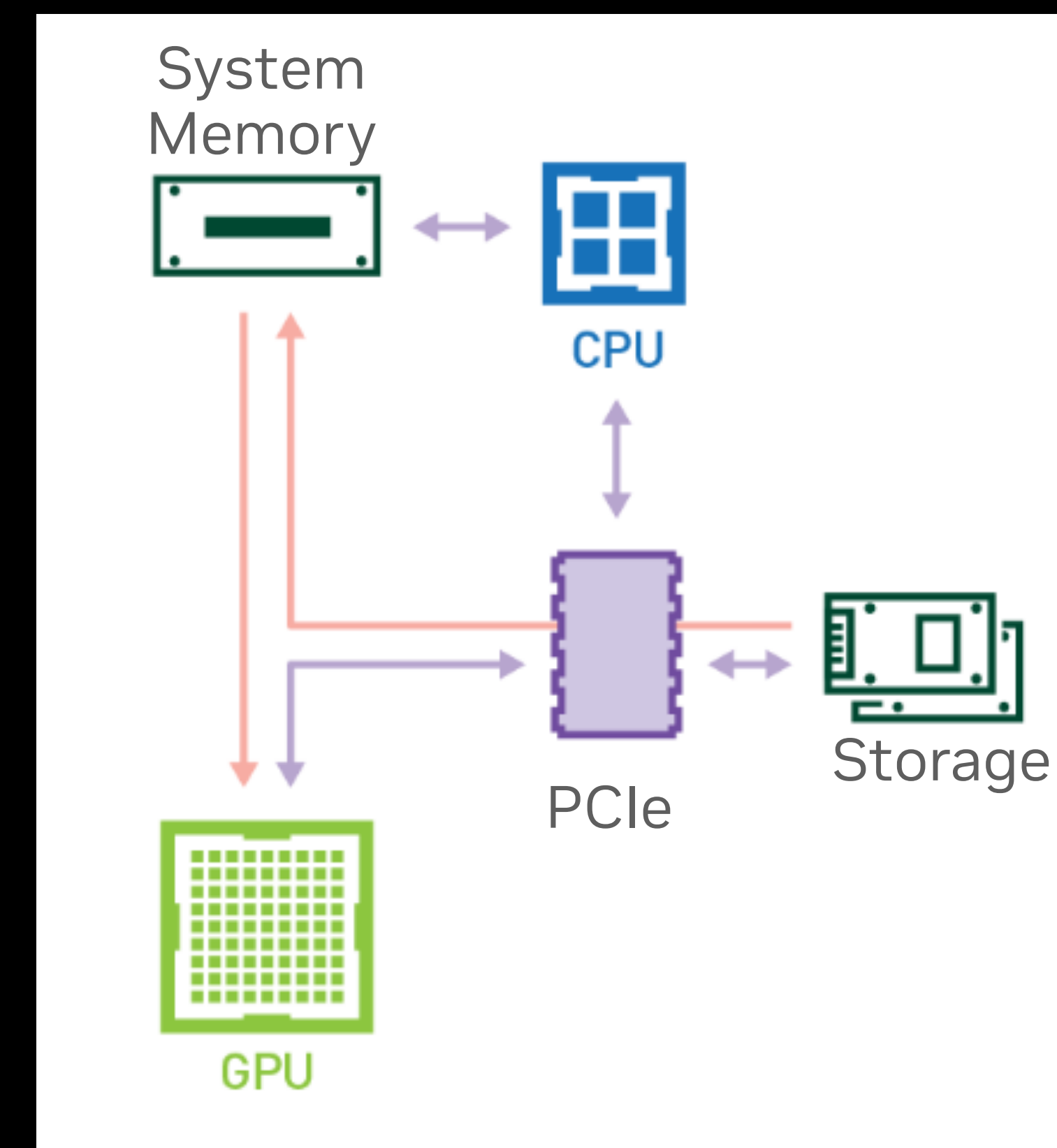
GPU



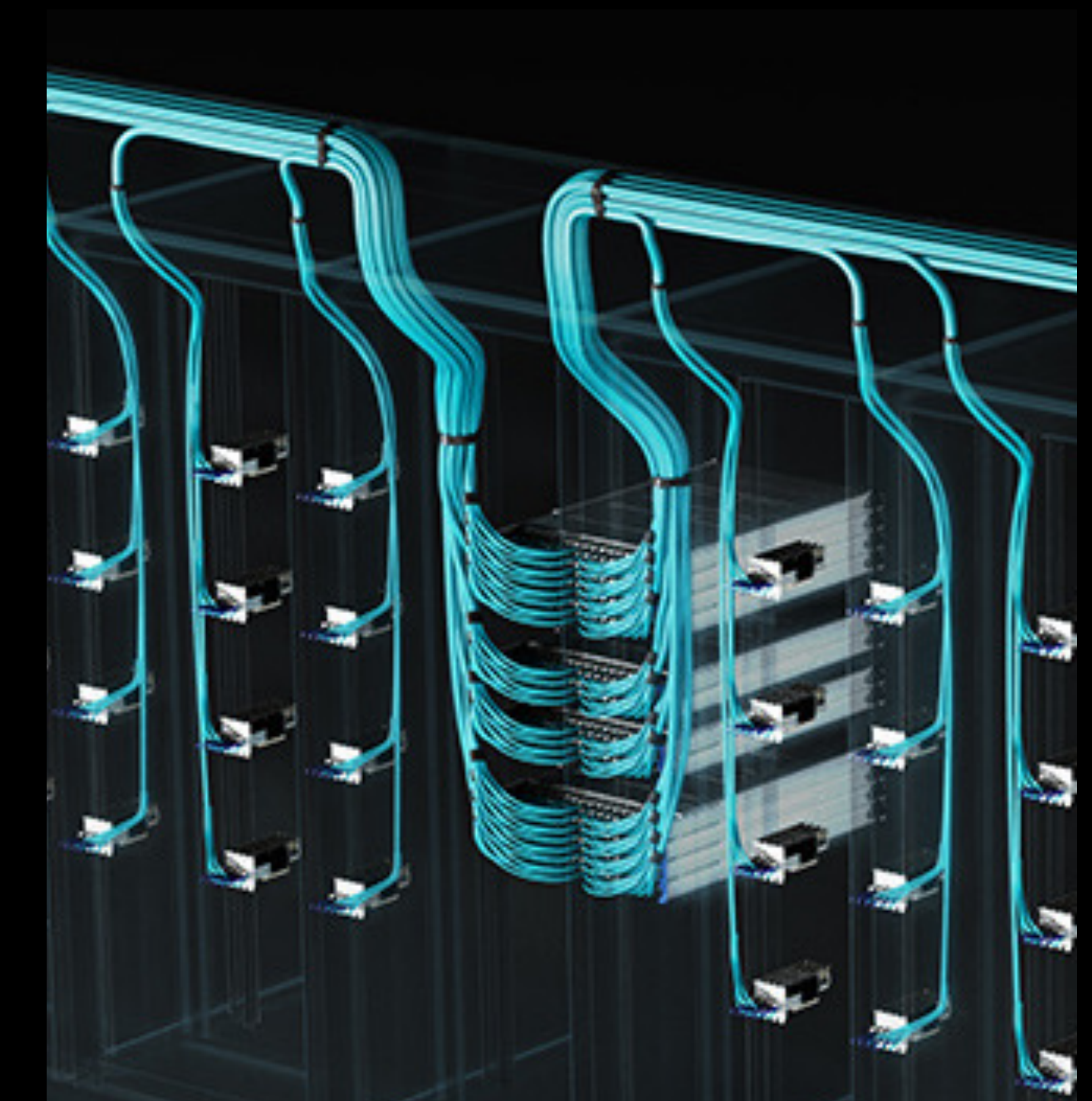
CPU



Memory



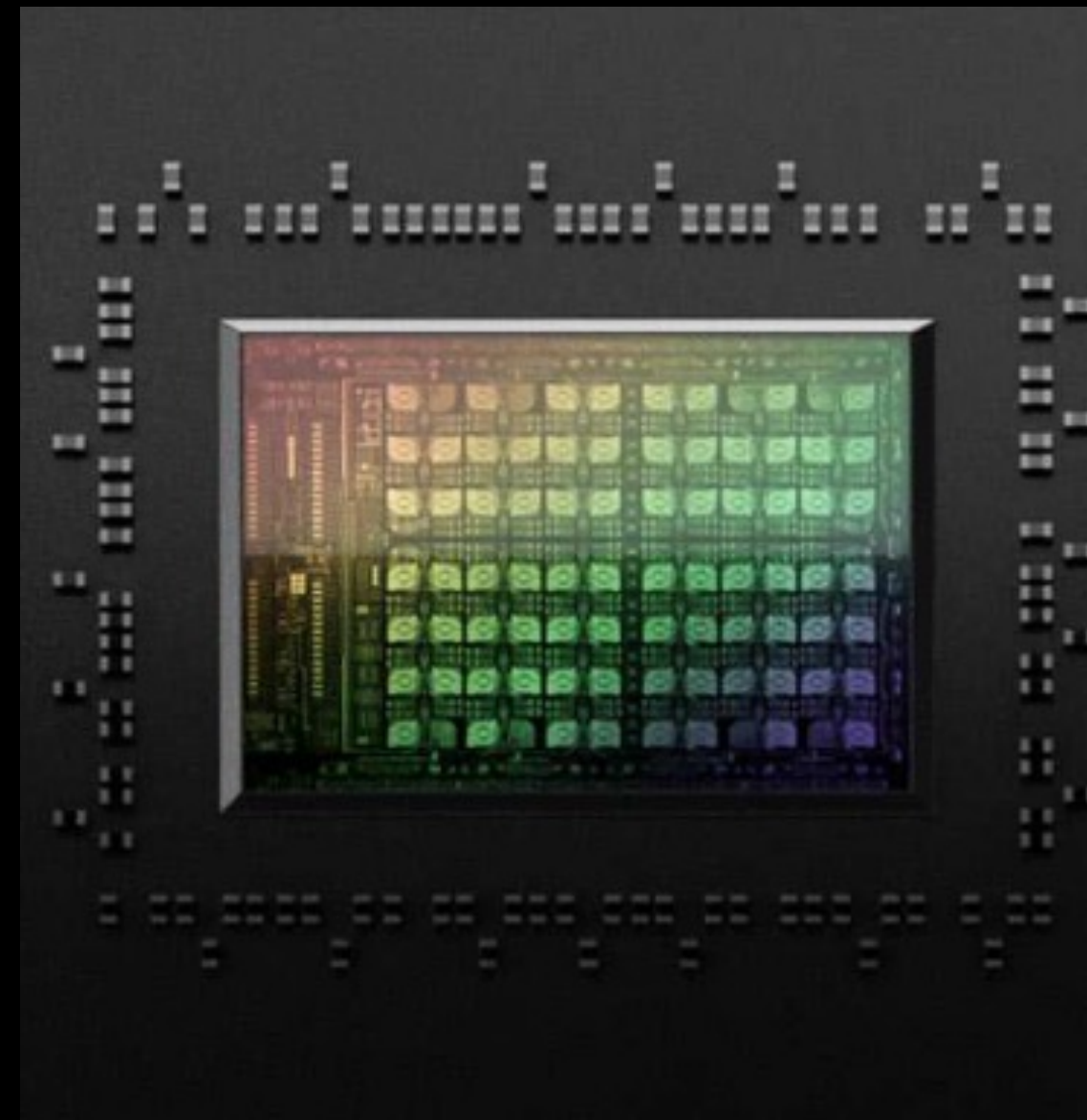
Communication



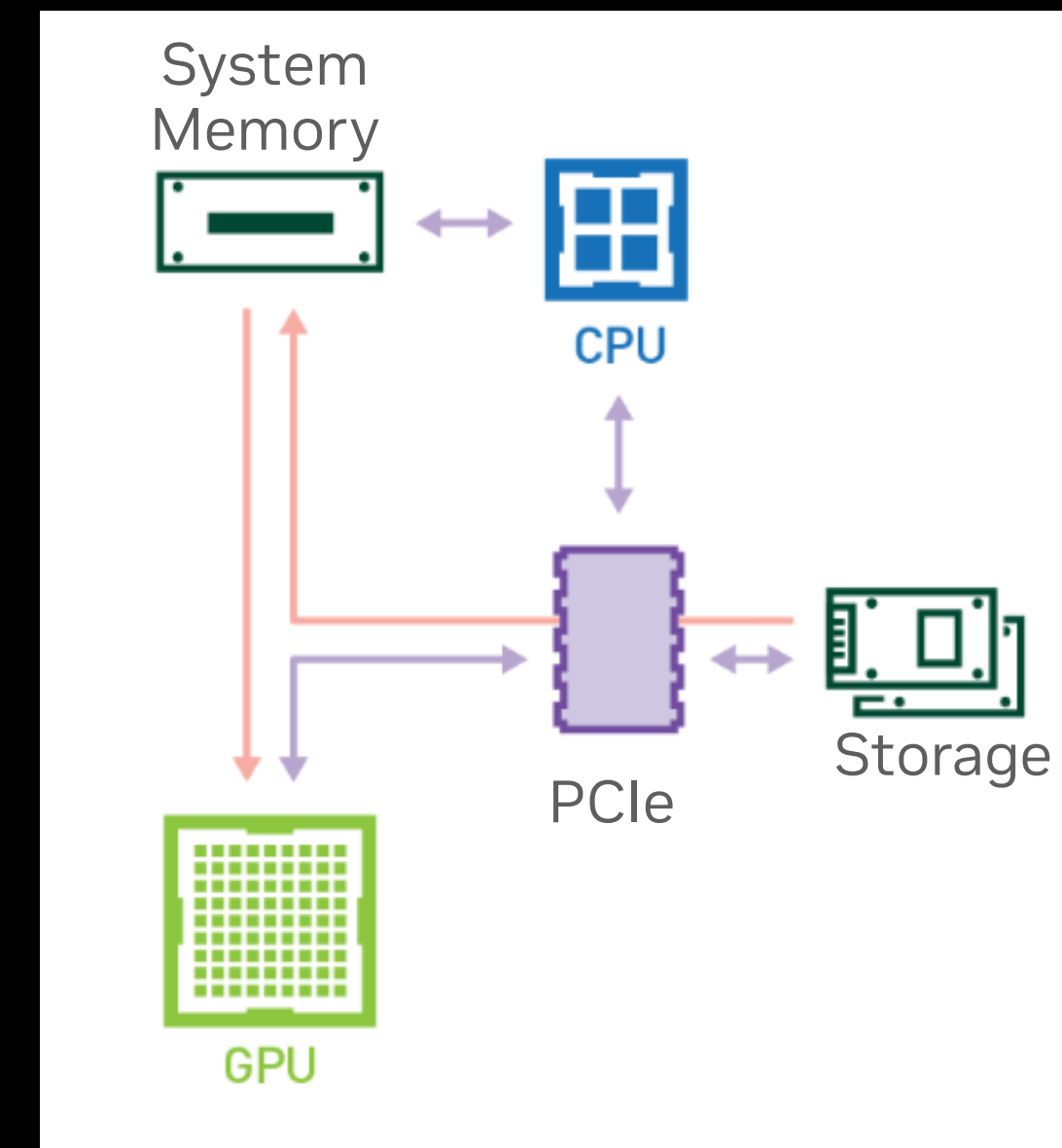
GPU



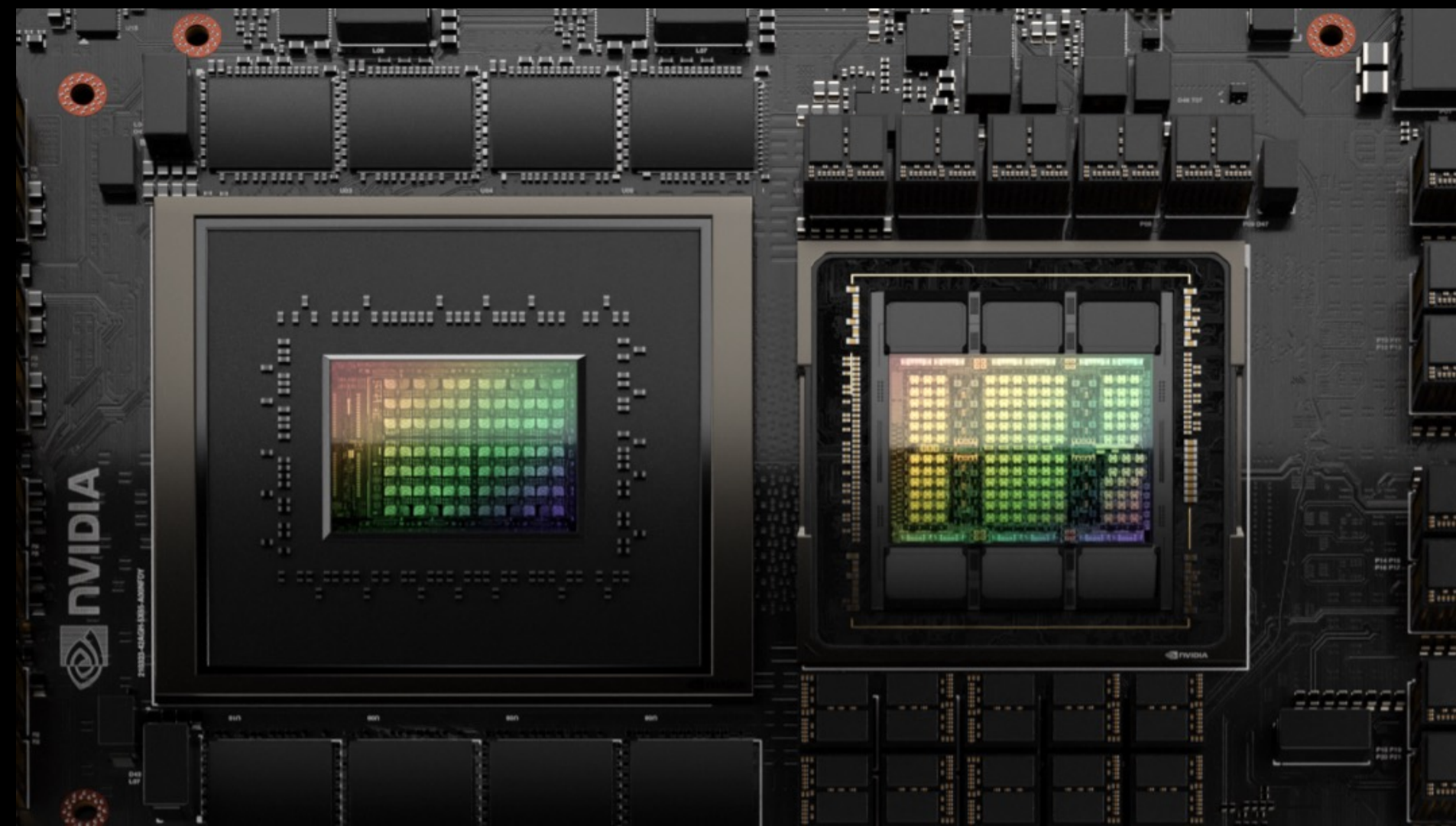
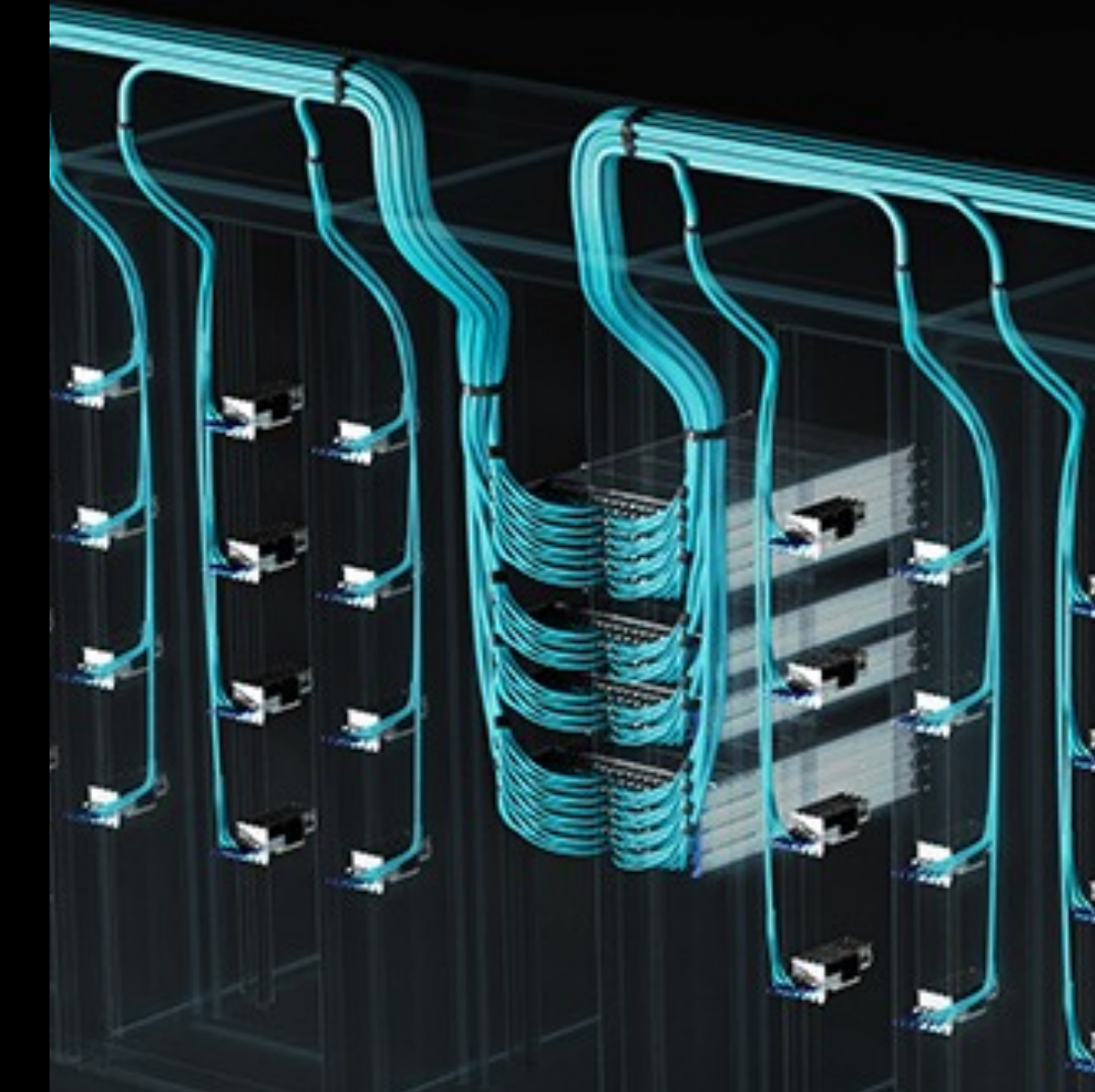
CPU



Memory



Communication



Multi-die
Grace Hopper
CPU+GPU

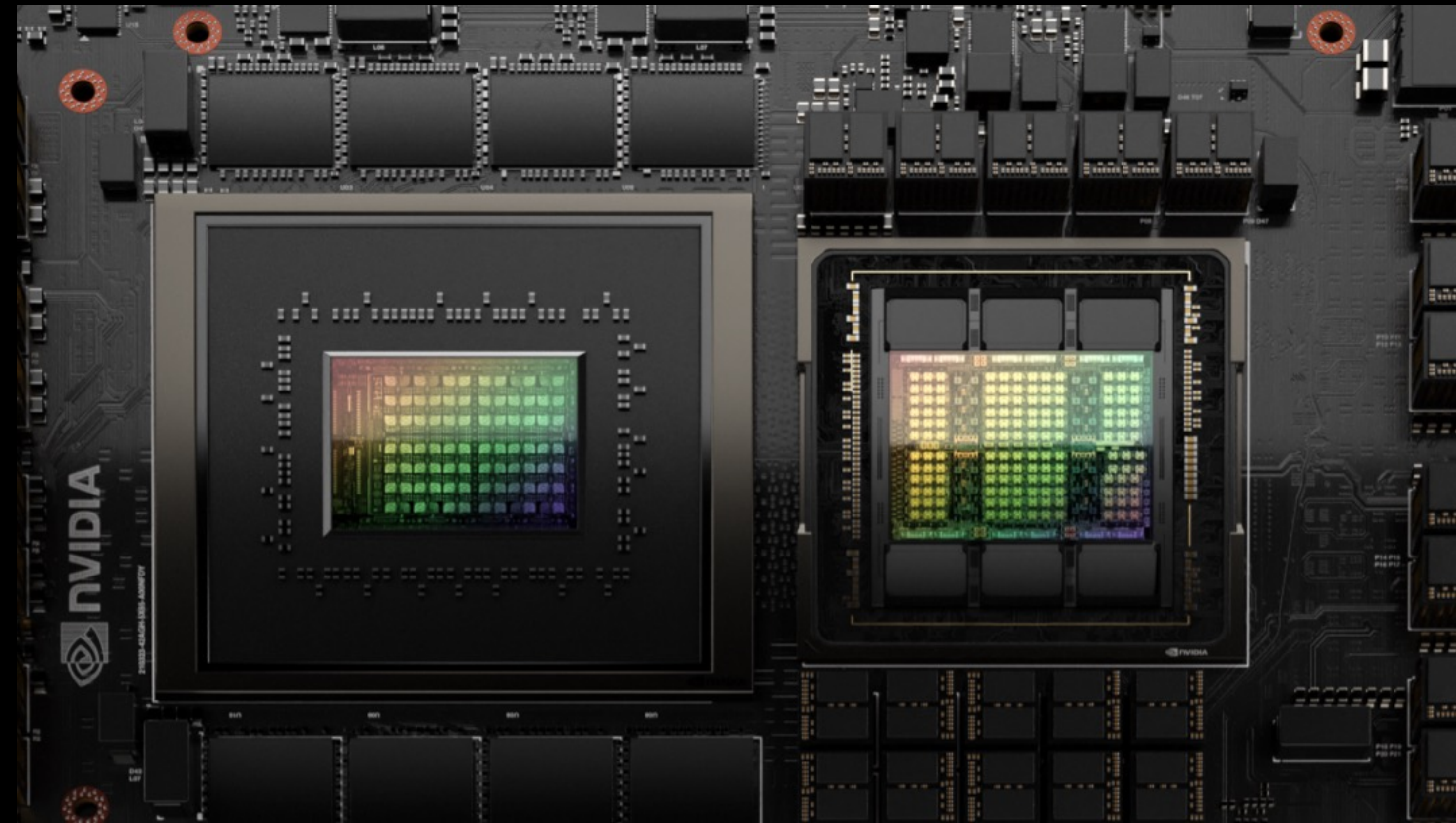


Multi-chip
NVLink/NVSwitch

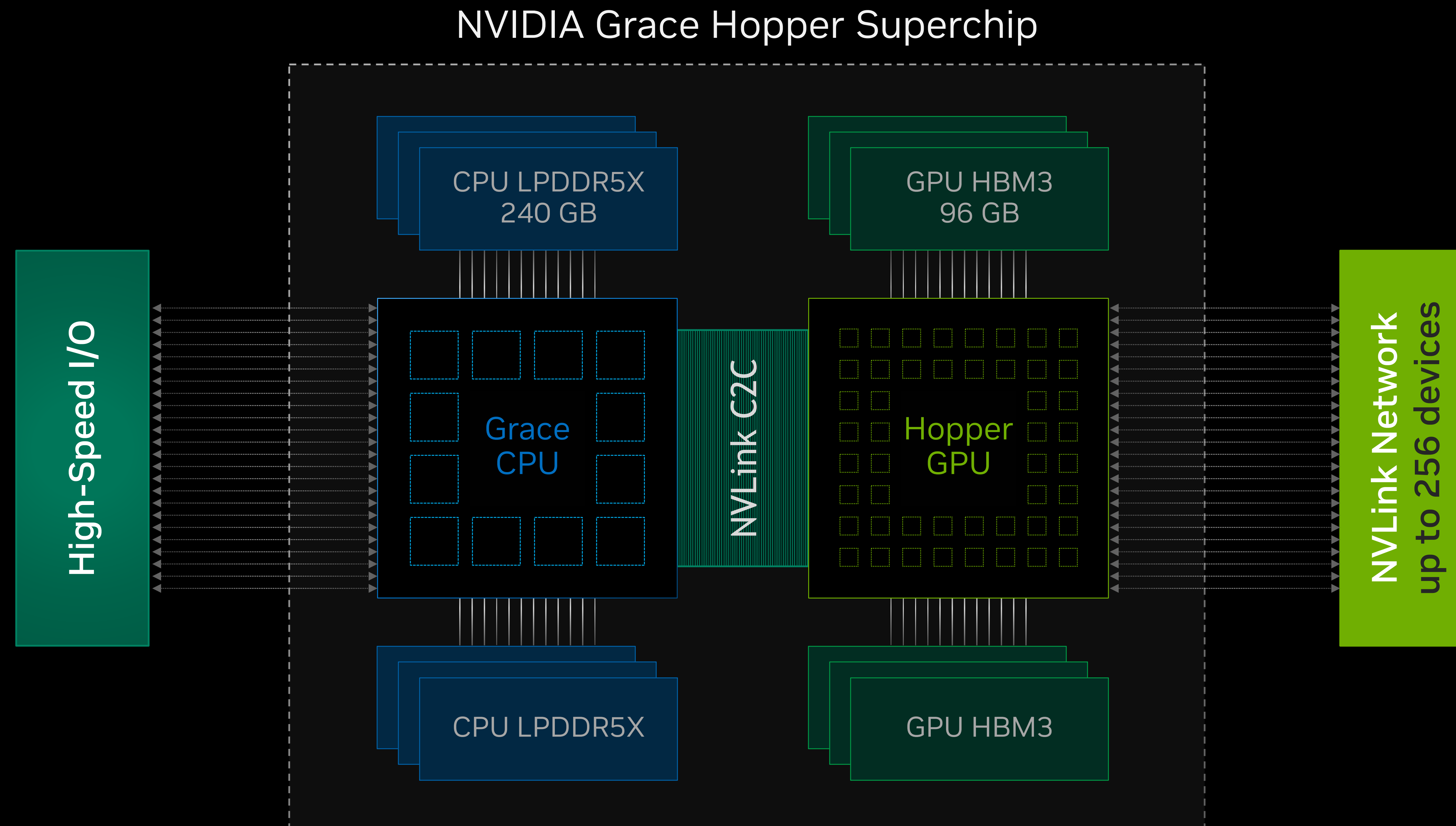


Multi-node
NVLink/NVSwitch
Infiniband

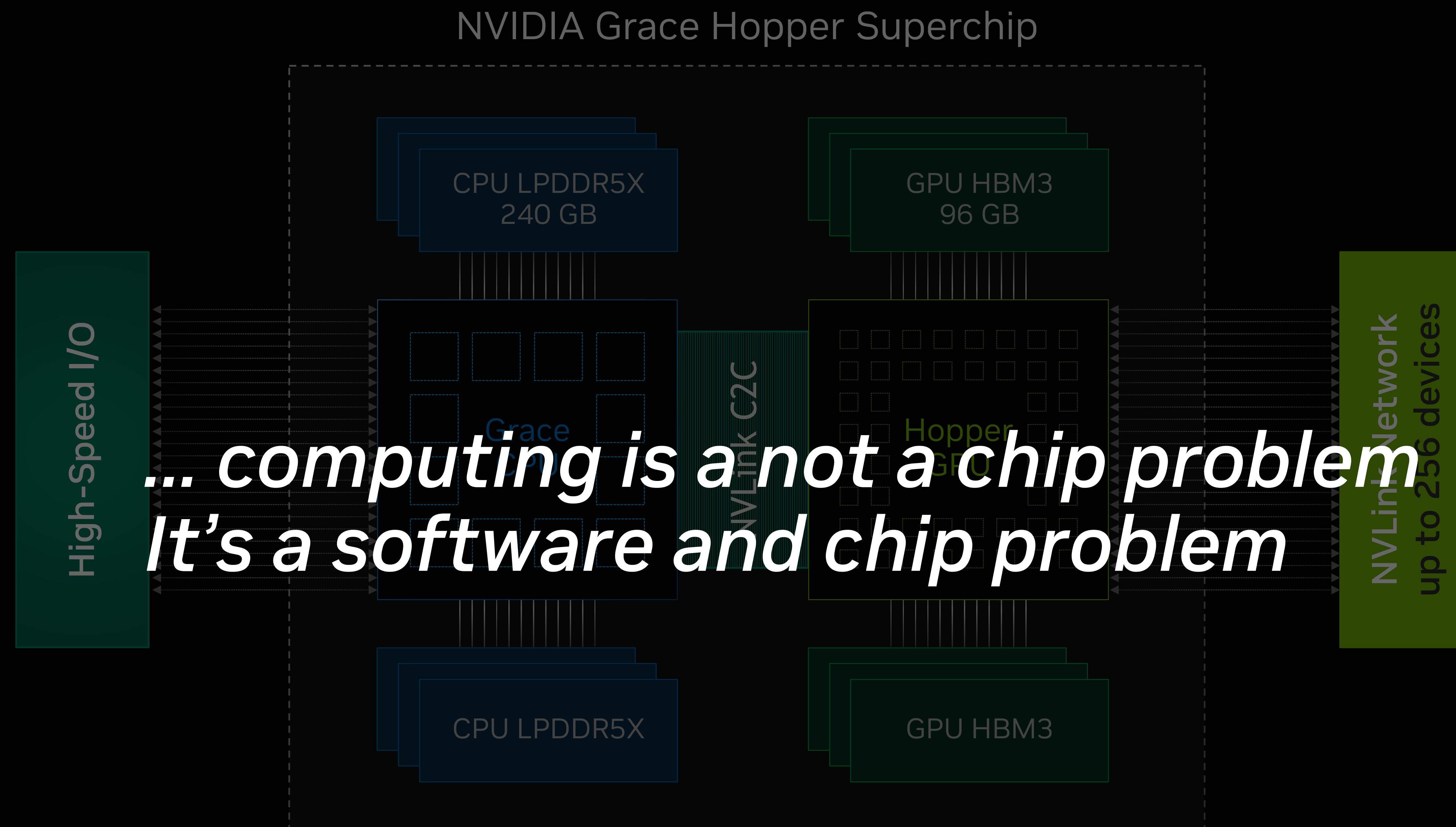
Grace/Hopper Superchip



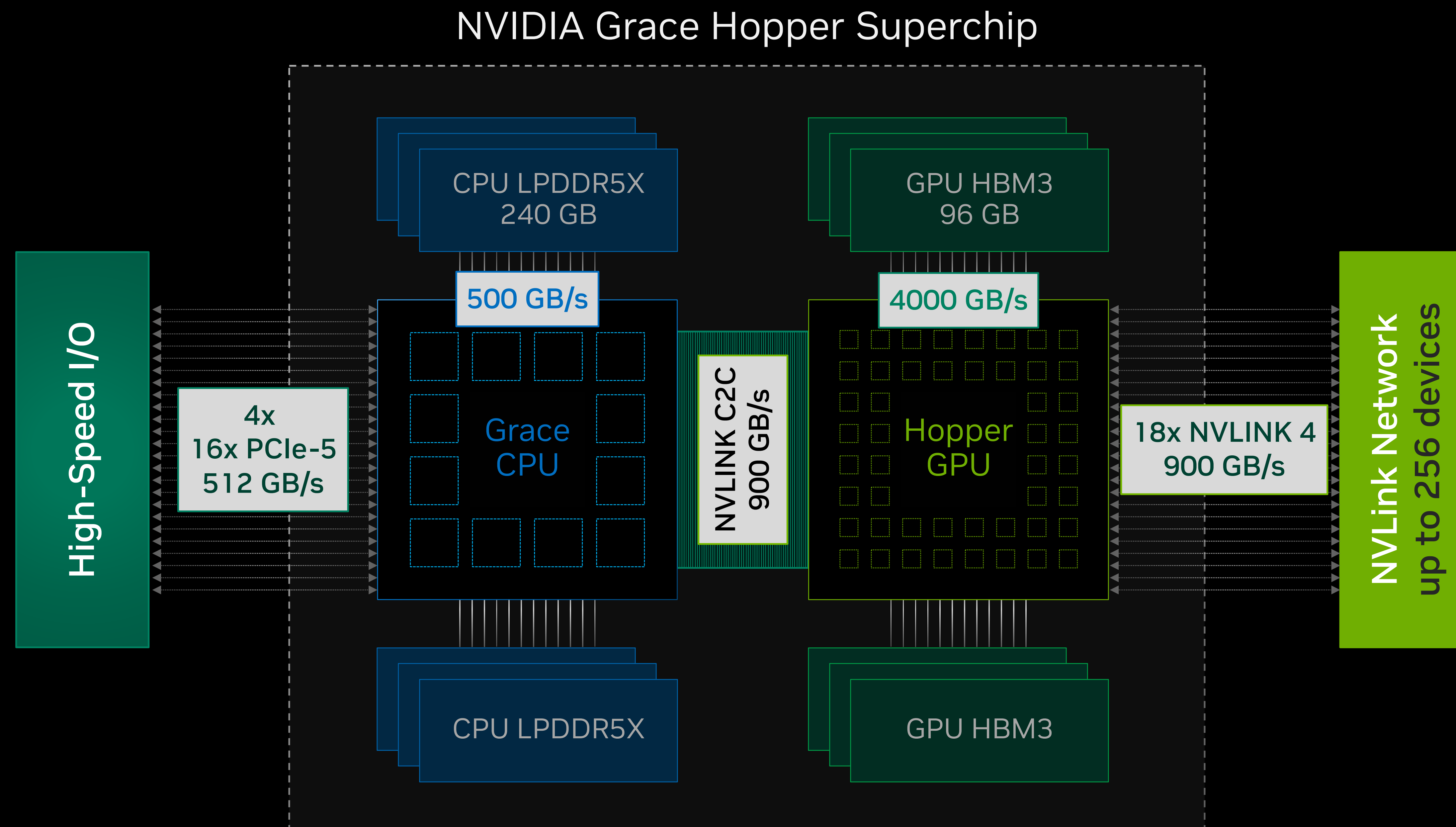
Grace/Hopper Superchip



Grace/Hopper Superchip

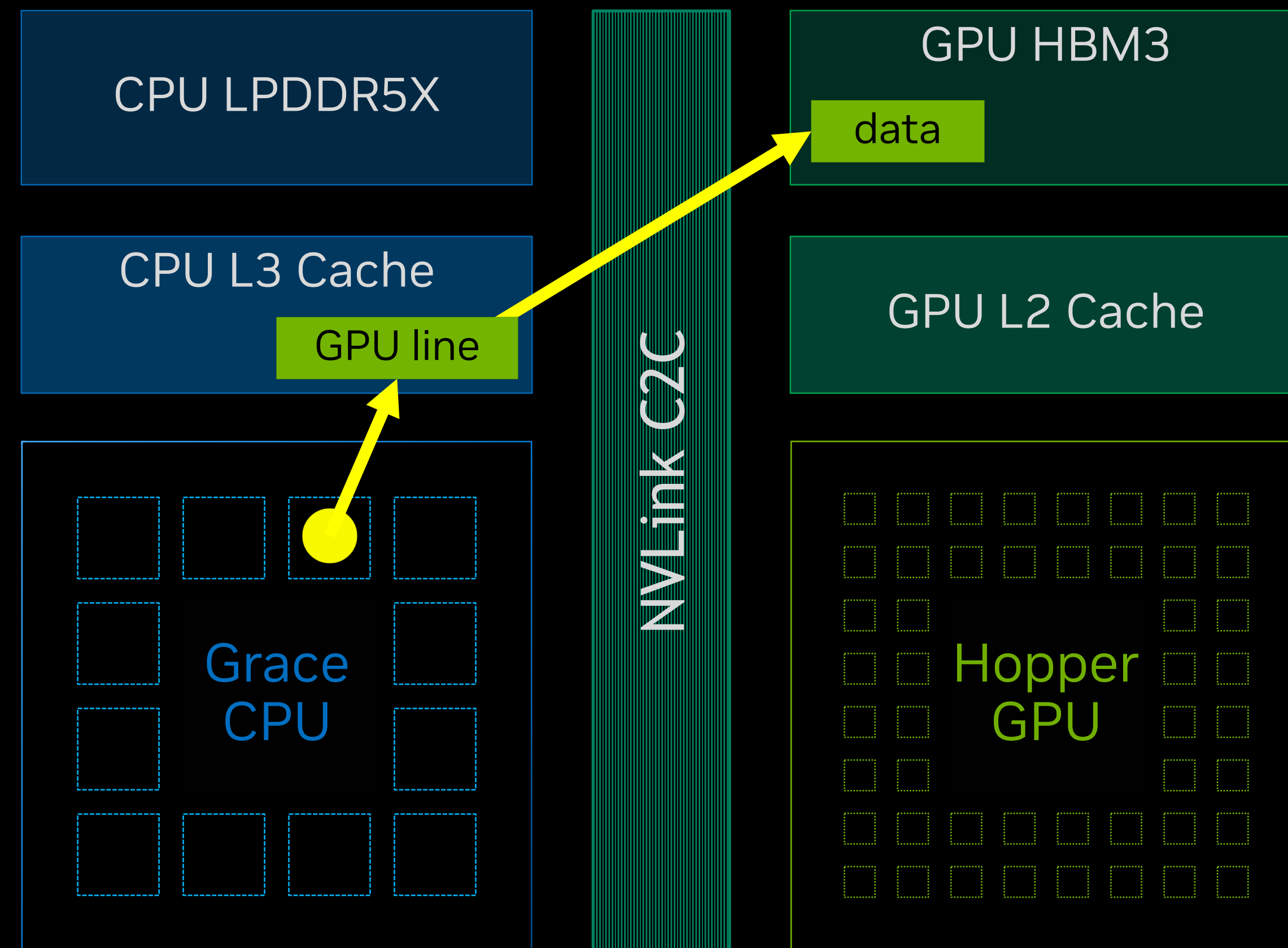


Grace/Hopper Superchip



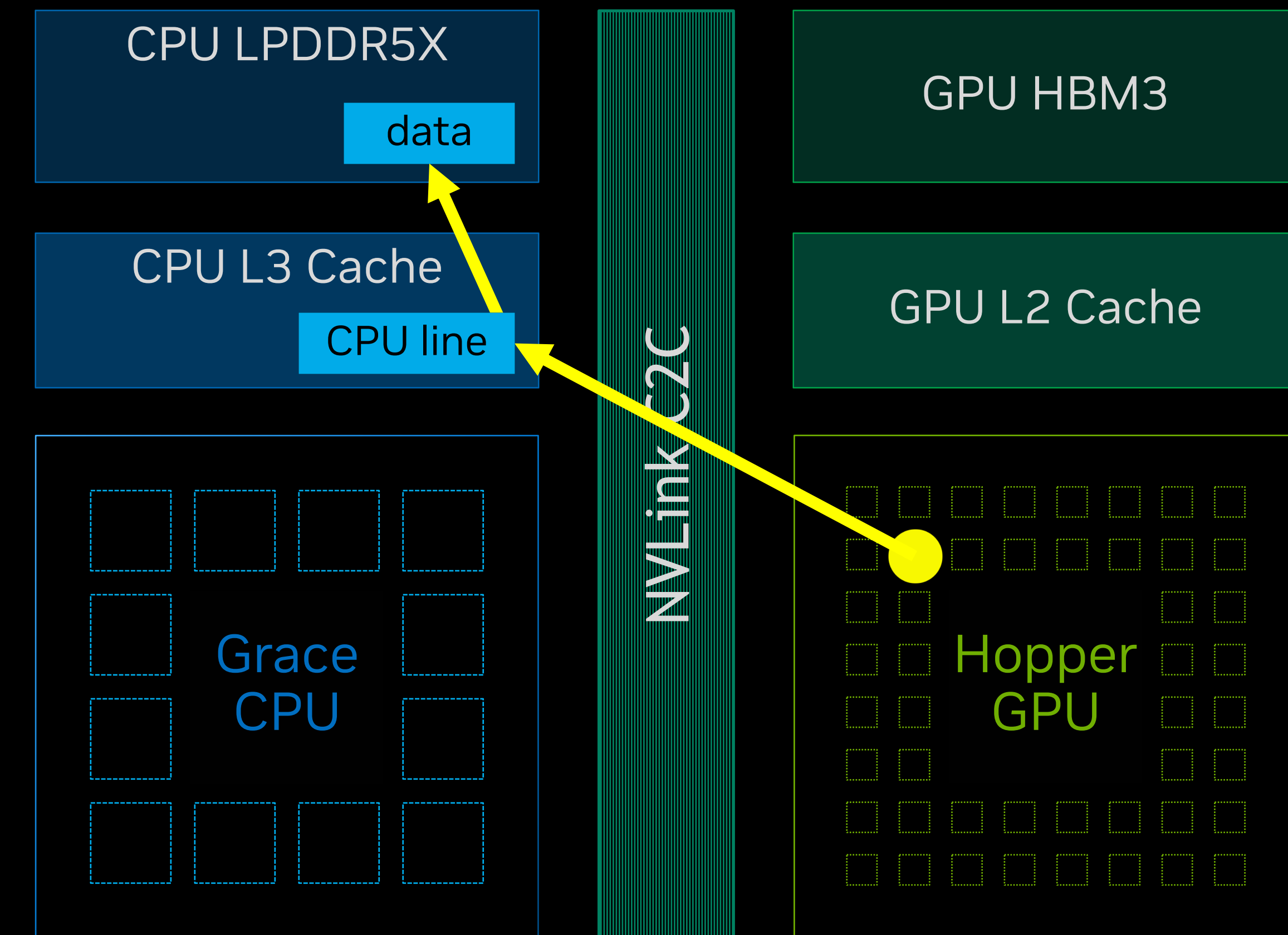
Global Access to All Data

Cache-coherent access via NVLink C2C from either processor to either physical memory



Grace directly reading Hopper's memory

CPU fetches GPU data into CPU L3 cache
Cache remains **coherent** with GPU memory
Changes to GPU memory **evict** cache line

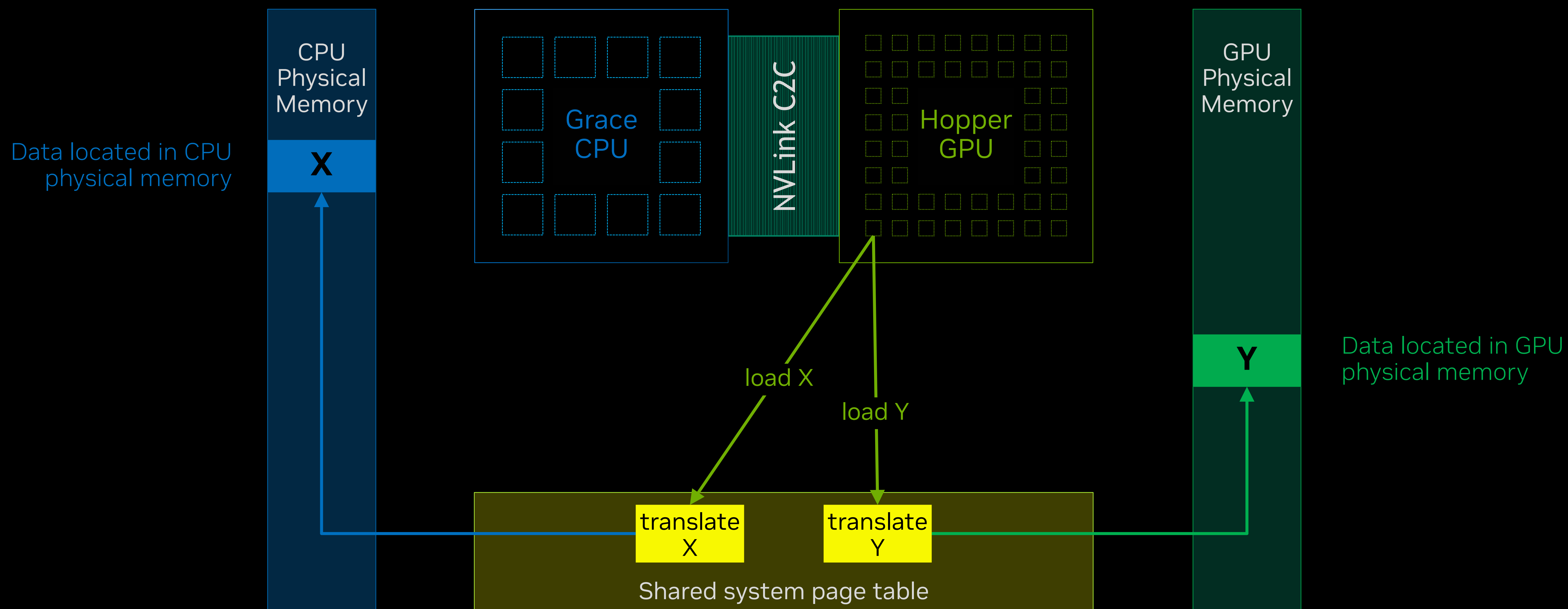


Hopper directly reading Grace's memory

GPU loads CPU data via CPU L3 cache
CPU and GPU **can both hit** on cached data
Changes to CPU memory **update** cache line

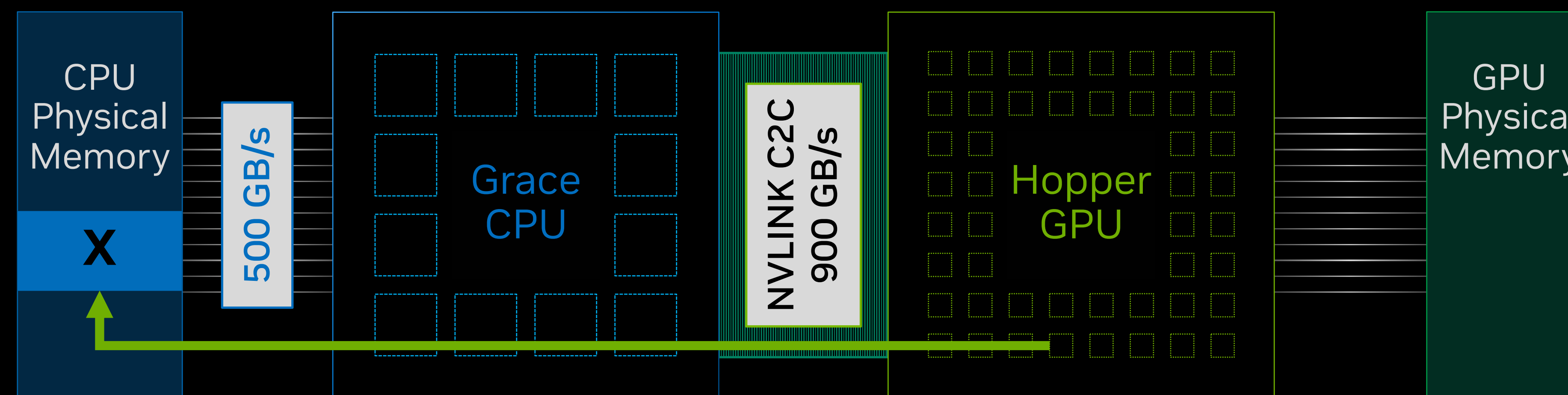
Grace/Hopper Unified Memory

Address Translation Service (ATS) allows full access to all CPU & GPU allocations

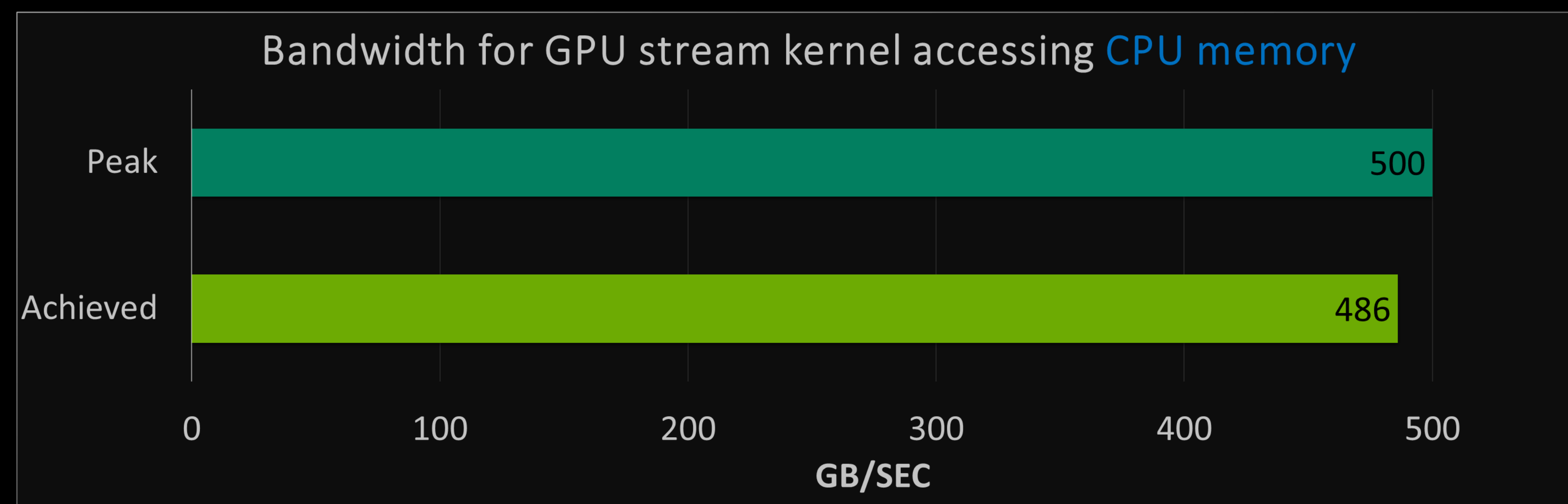


ATS creates a single page table for the whole system
NVLink C2C allows access to all physical memory **without migration**

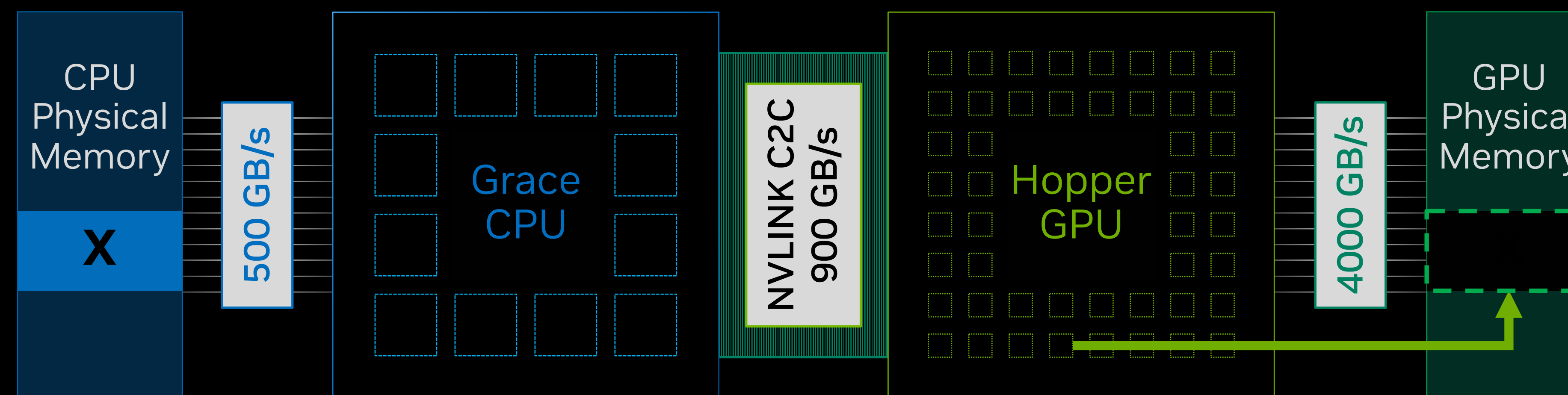
High Bandwidth Memory Access & Automatic Data Migration



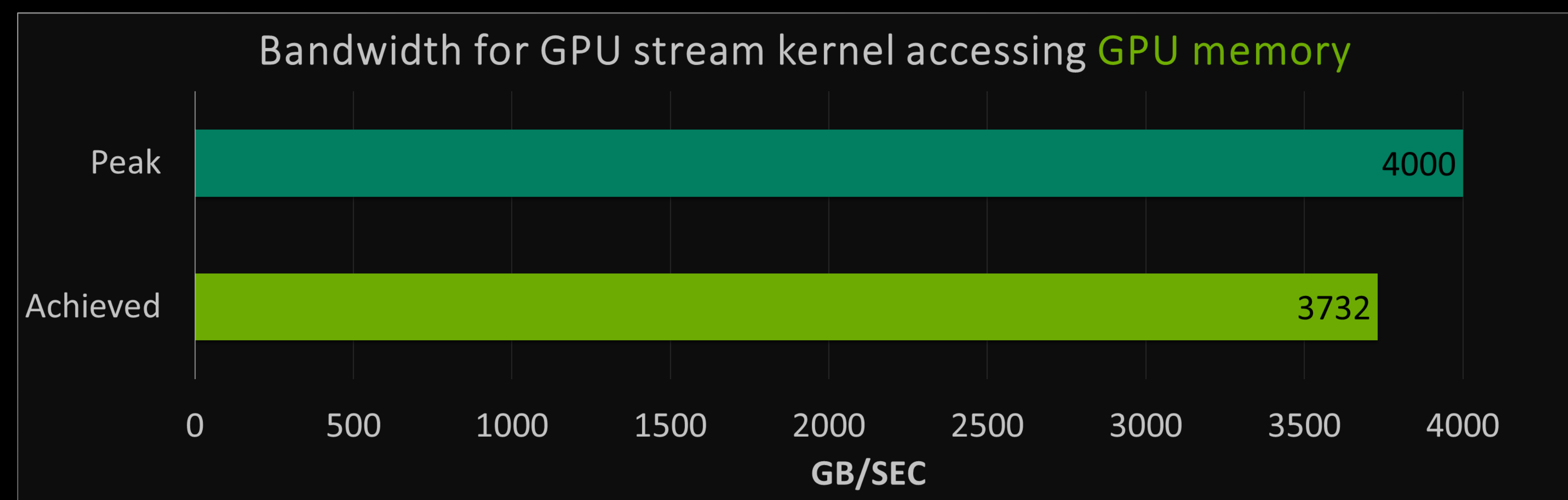
Hopper can access Grace memory at **full CPU memory speed** of 500 GB/sec



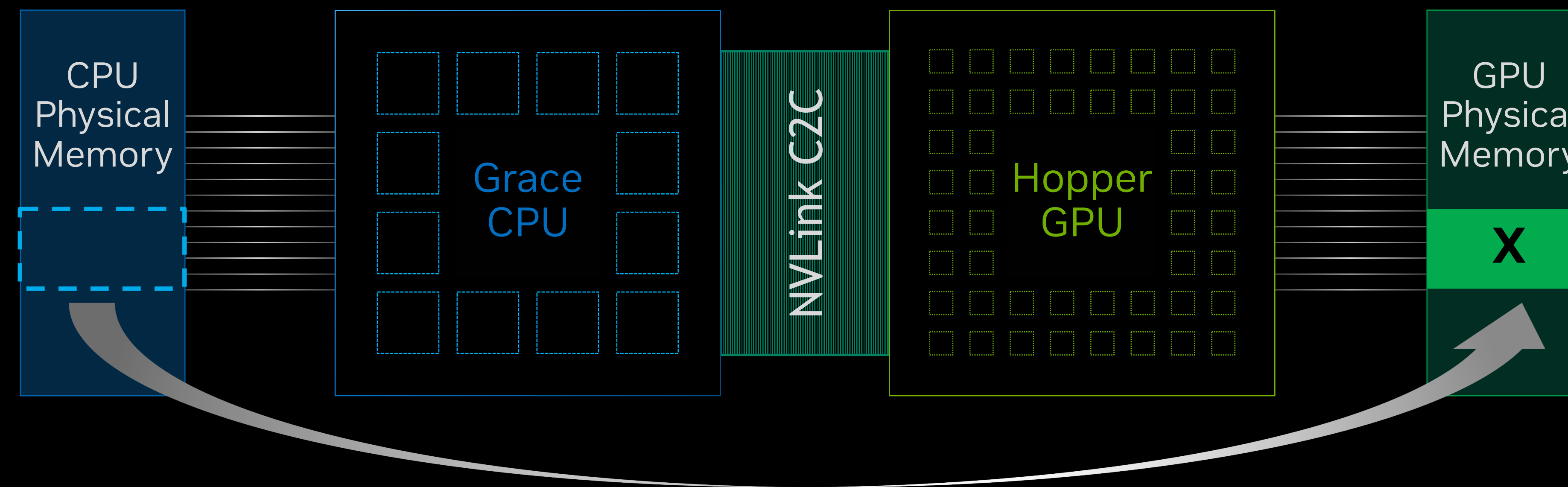
High Bandwidth Memory Access & Automatic Data Migration



But Hopper can access its own memory at **full HBM speed** of 4000 GB/sec

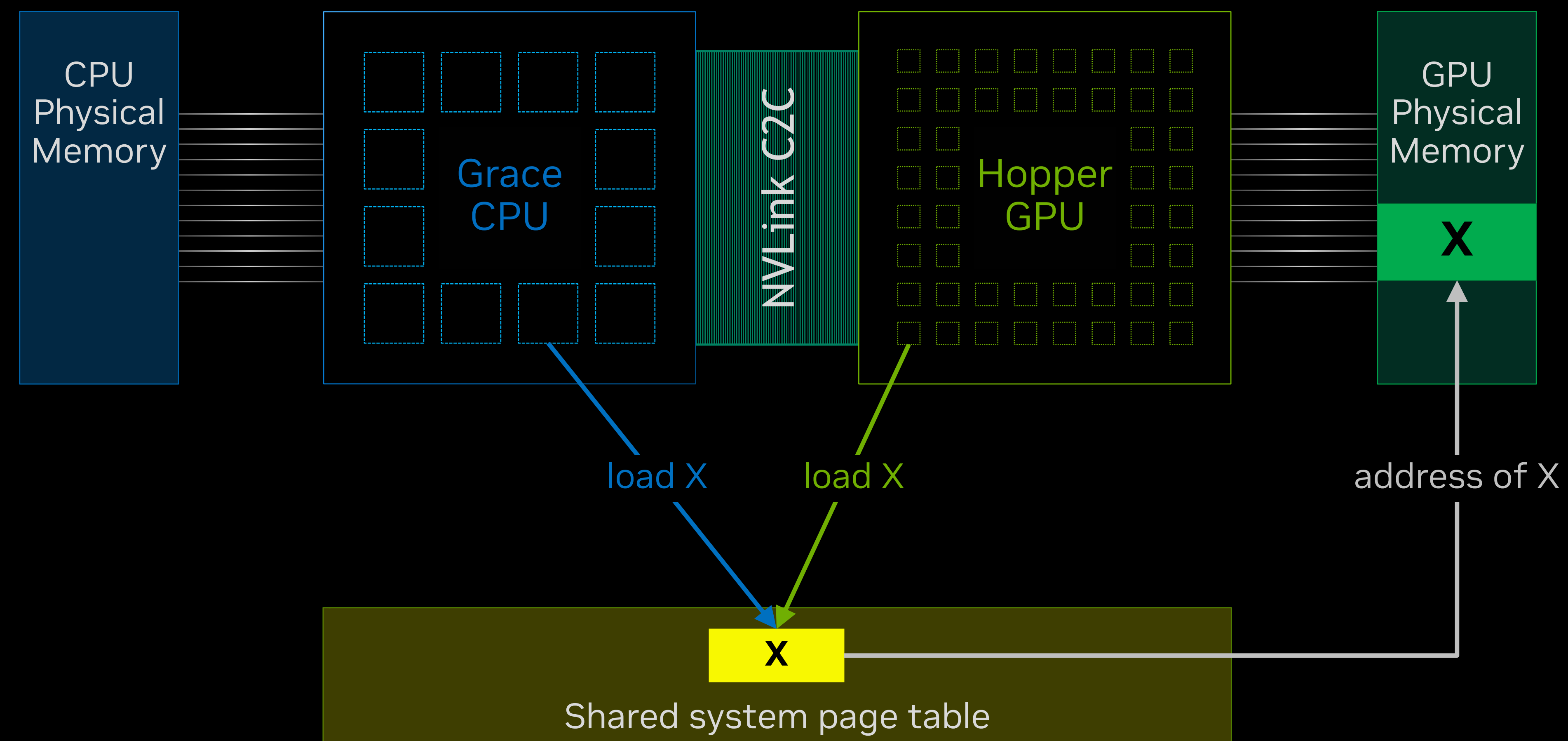


High Bandwidth Memory Access & Automatic Data Migration



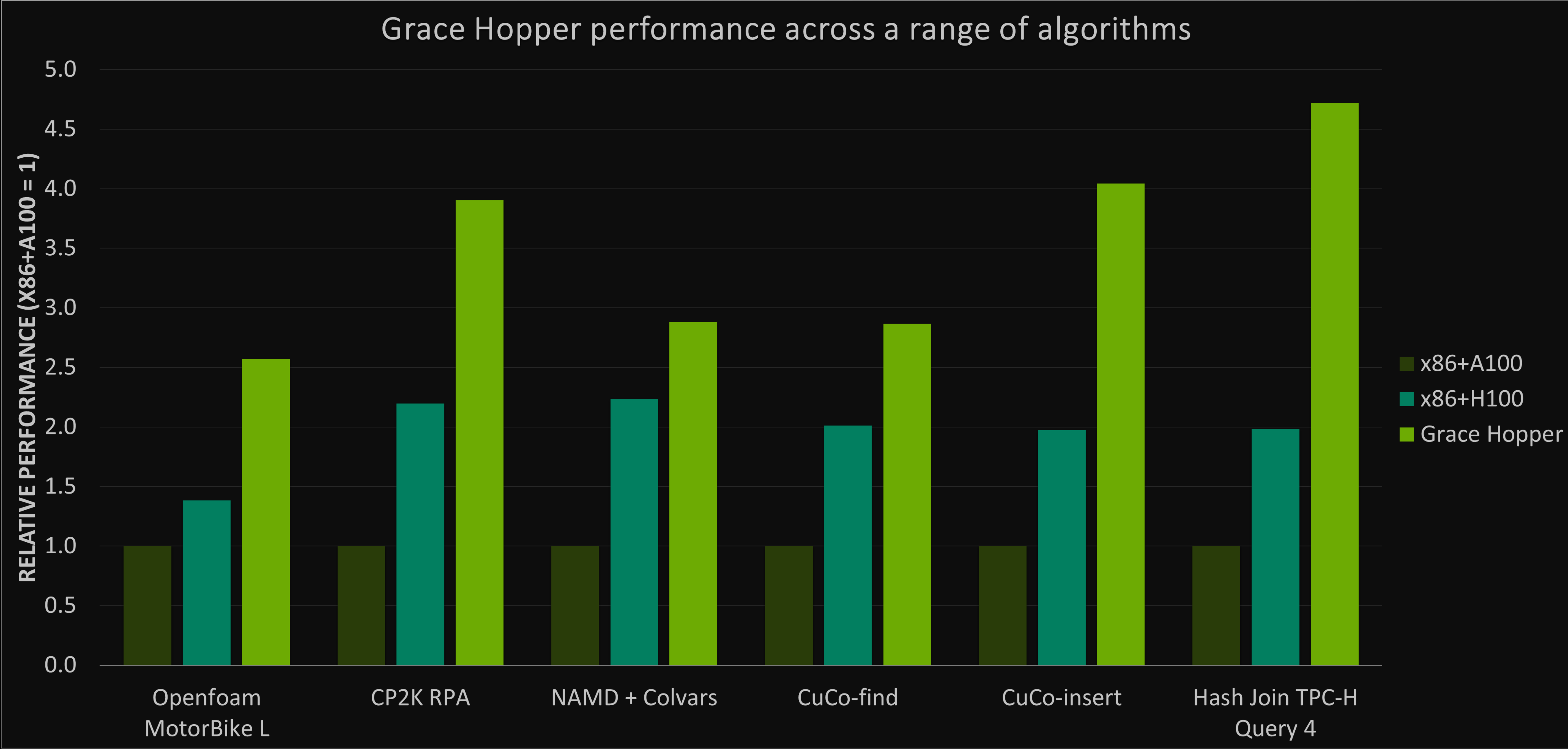
The system can **automatically** migrate both managed and CPU-allocated memory in order to optimize access speed

High Bandwidth Memory Access & Automatic Data Migration

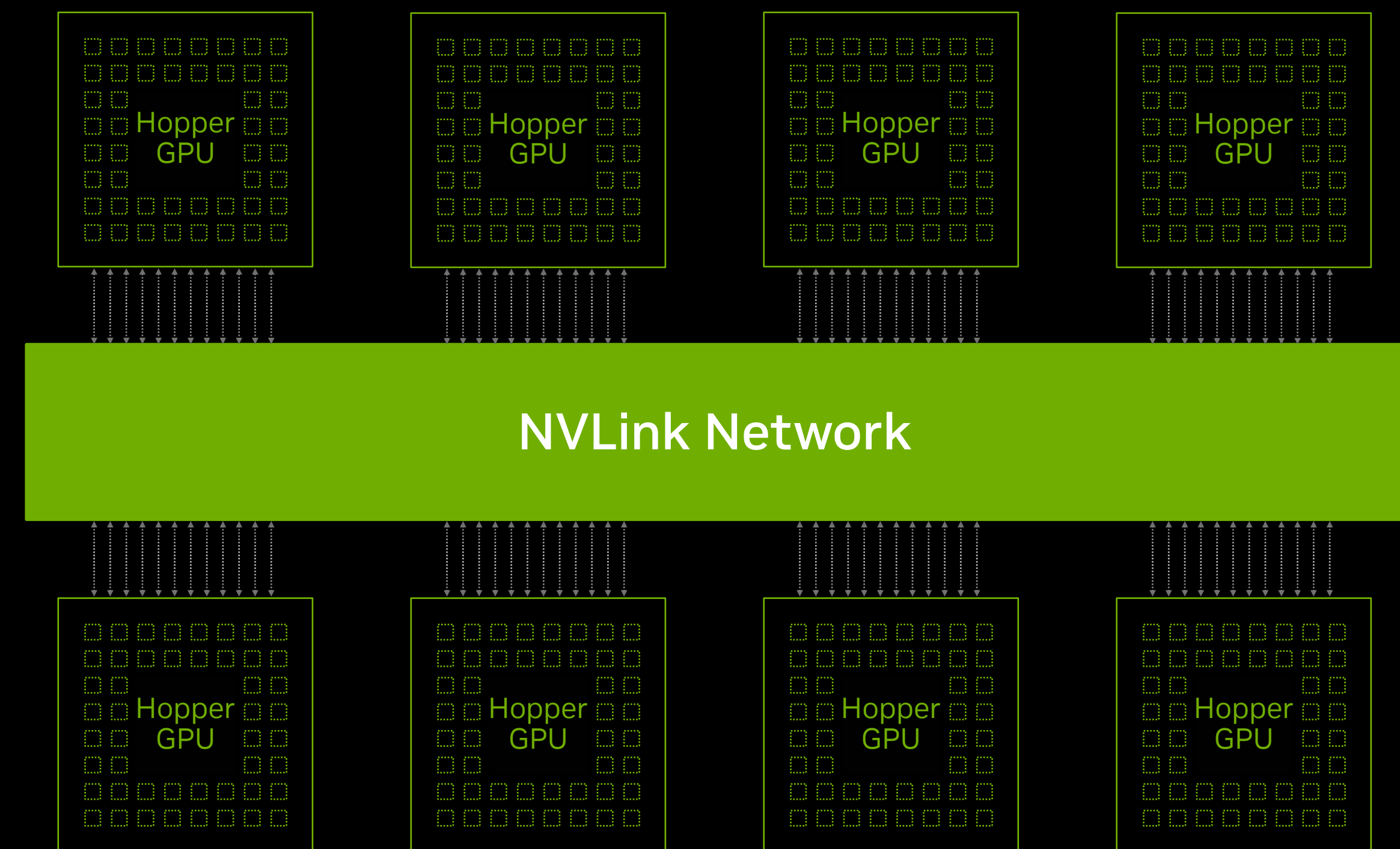
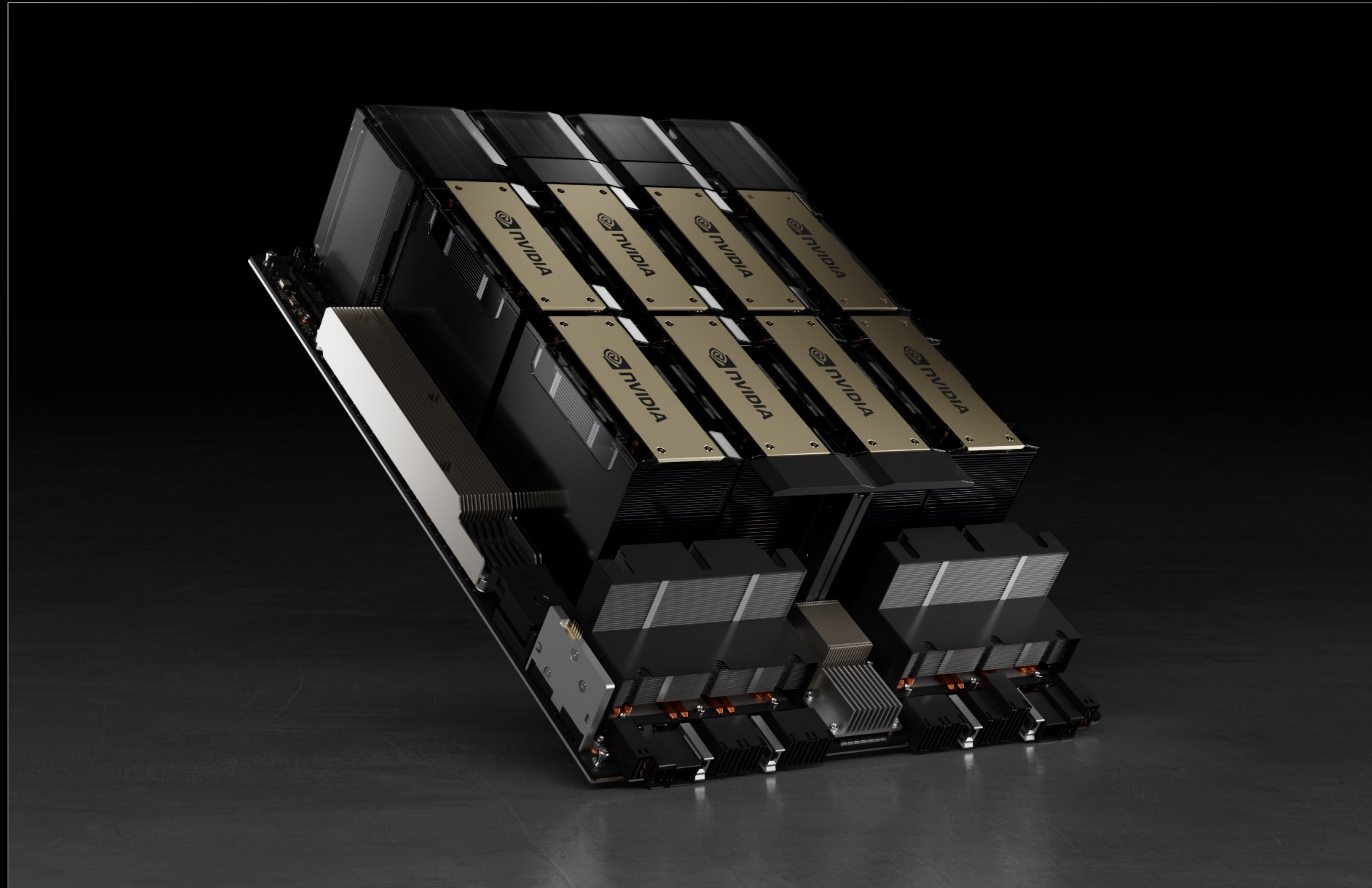


ATS shared page table means that both CPU and GPU automatically access X in its new location after migration

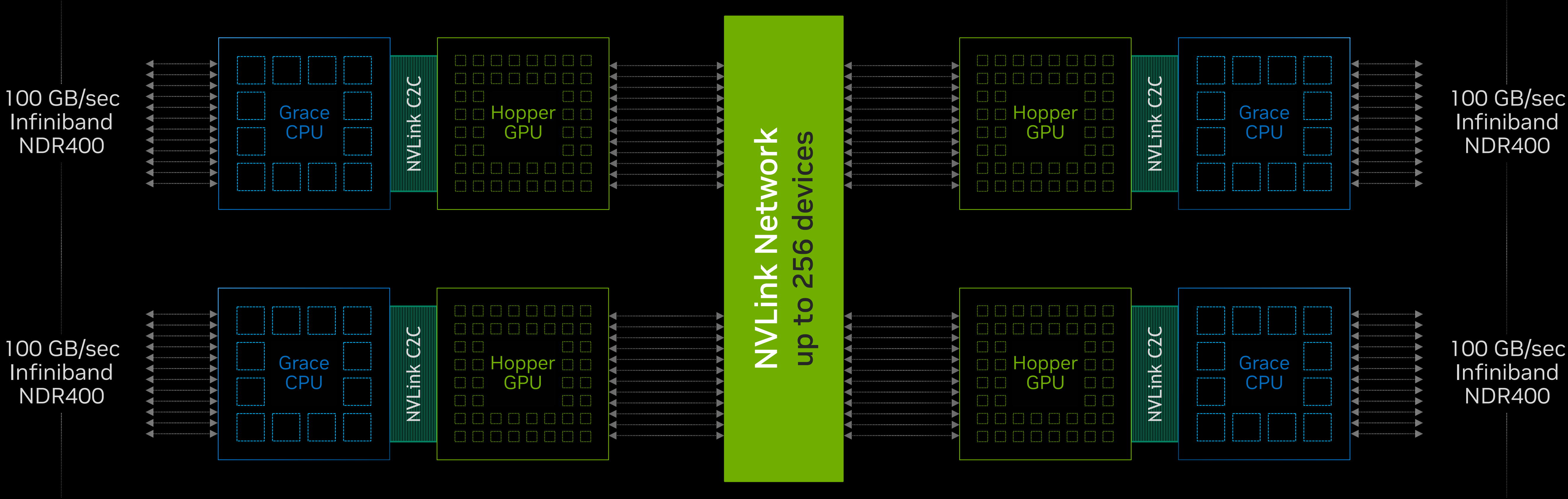
Grace/Hopper SuperChip Performance On Different Workloads



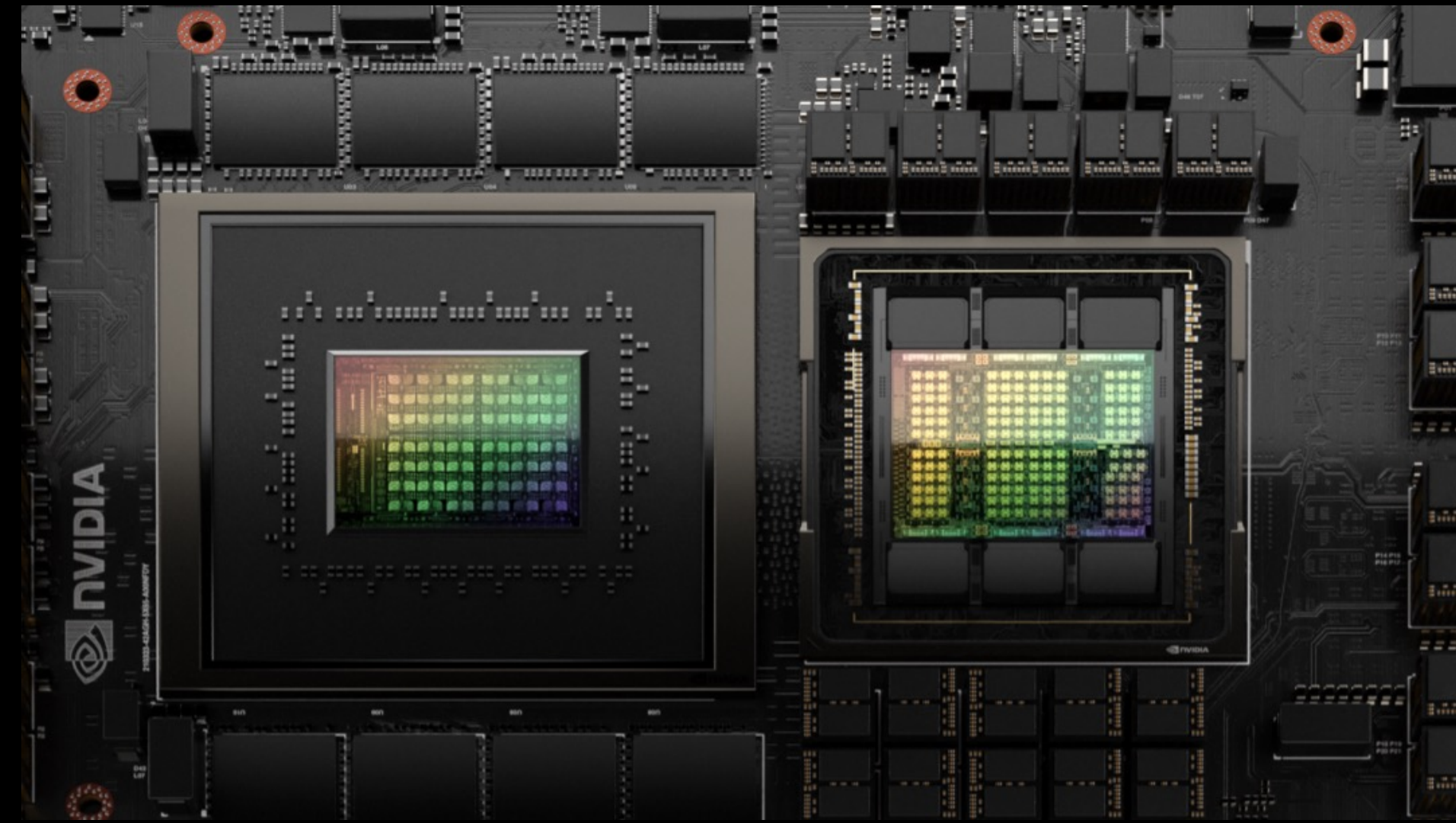
Multi-Chip Systems



NVLink Connects Up To 256 Superchips



<https://www.nvidia.com/en-us/data-center/nvlink/>
<https://resources.nvidia.com/en-us-tensor-core/gtc22-whitepaper-hopper>
<https://resources.nvidia.com/en-us-grace-cpu/nvidia-grace-hopper>



Multi-die
NVLINK C2C



Multi-chip
NVLINK+NVSWITCH

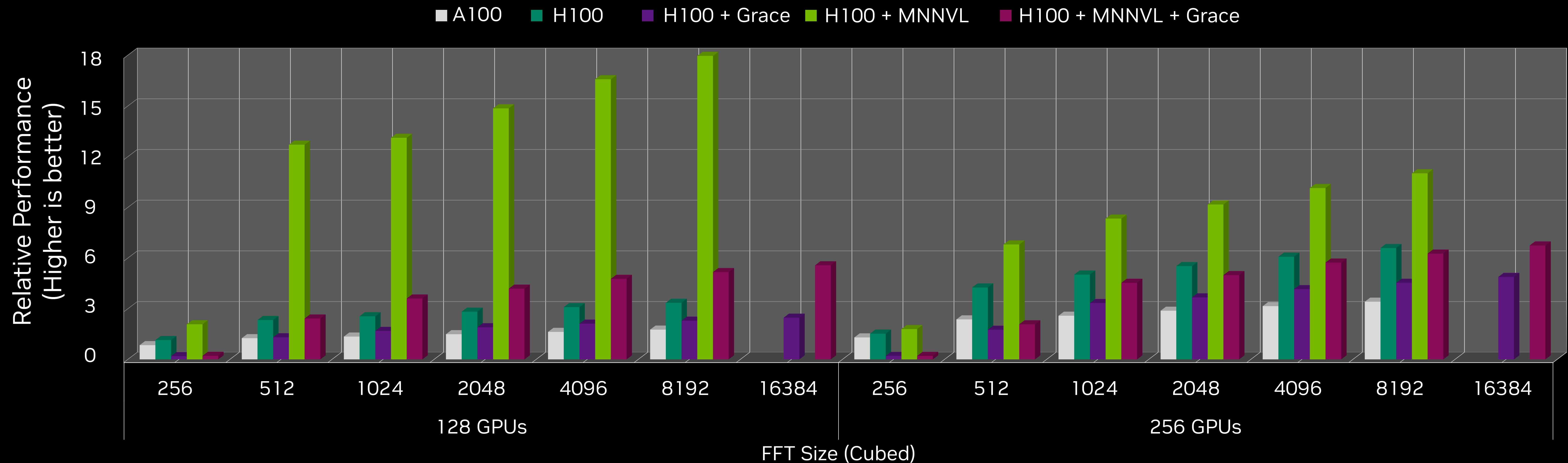


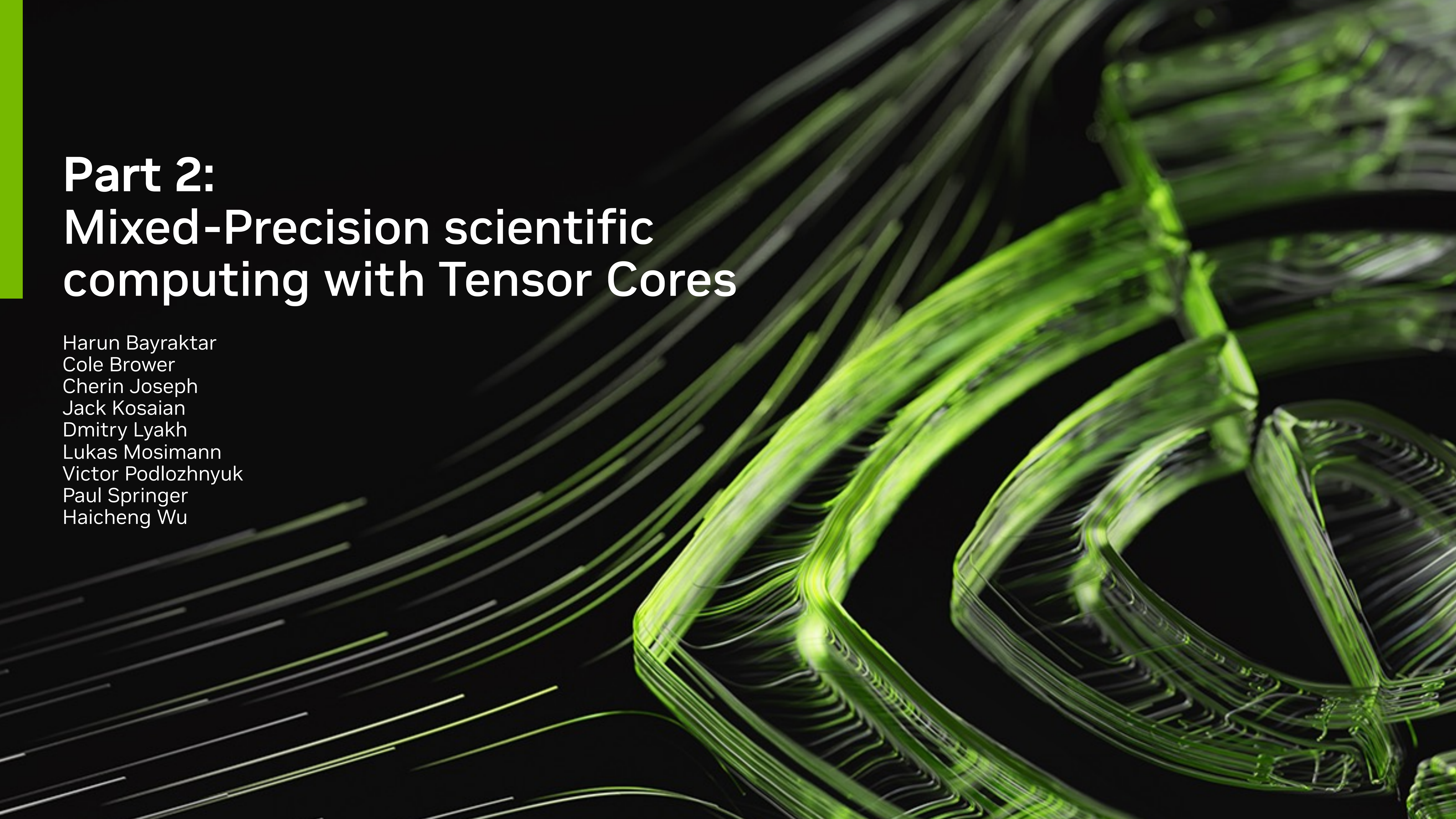
Multi-node (MNNVL)
NVLINK+NVSWITCH up to 256 GPUs
+ Infiniband

Multi-Node NVLink cuFFTMp

Performance Projections

- MNNVL offers significant performance on MGMN FFTs
- Drop in performance between 128 and 256 GPU, because IB is used (left to right – Green bars)
- Using CPU memory on Grace Hopper allows for bigger sizes, 32k³ starting at 1024 GPUs

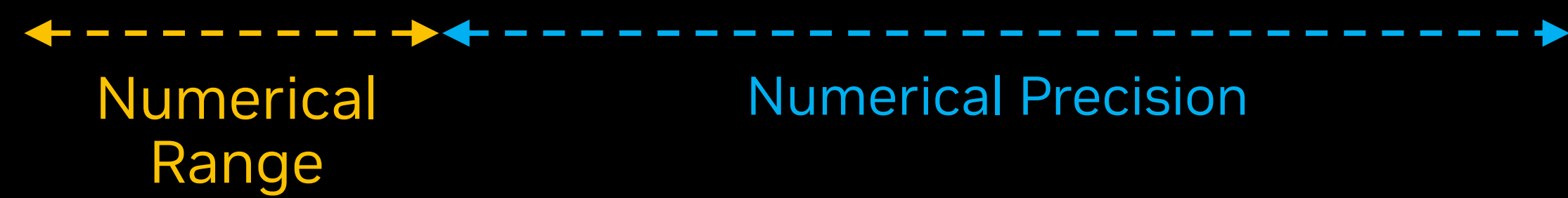
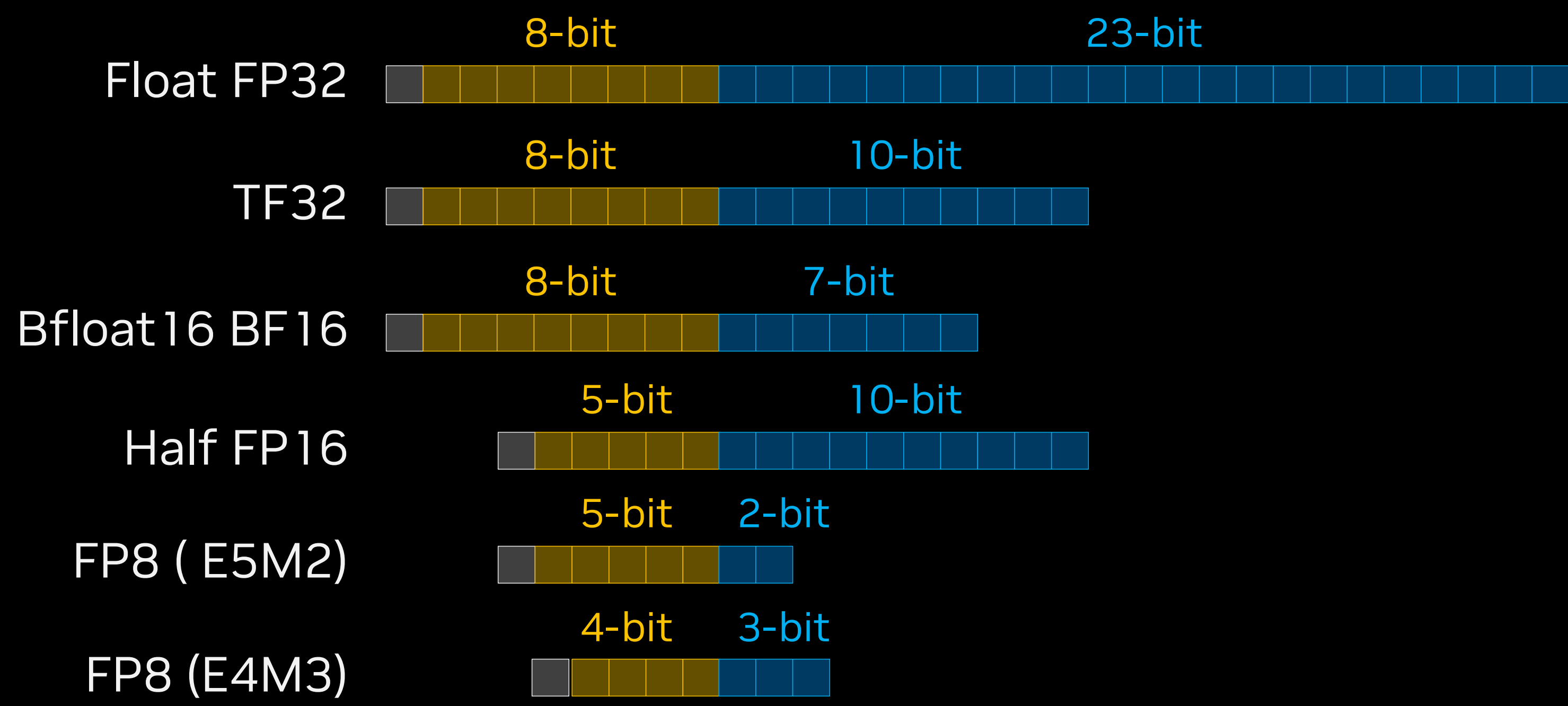


The background features a dark, almost black, space filled with vibrant green light trails and geometric patterns. On the left, a solid green vertical bar is partially visible. The main area is dominated by a series of parallel, slightly curved green lines that create a sense of depth and motion. In the foreground, there are several overlapping, glowing green rectangular shapes that resemble stylized architectural elements or data structures, possibly representing tensor cores or data flow paths. The overall aesthetic is futuristic and technical.

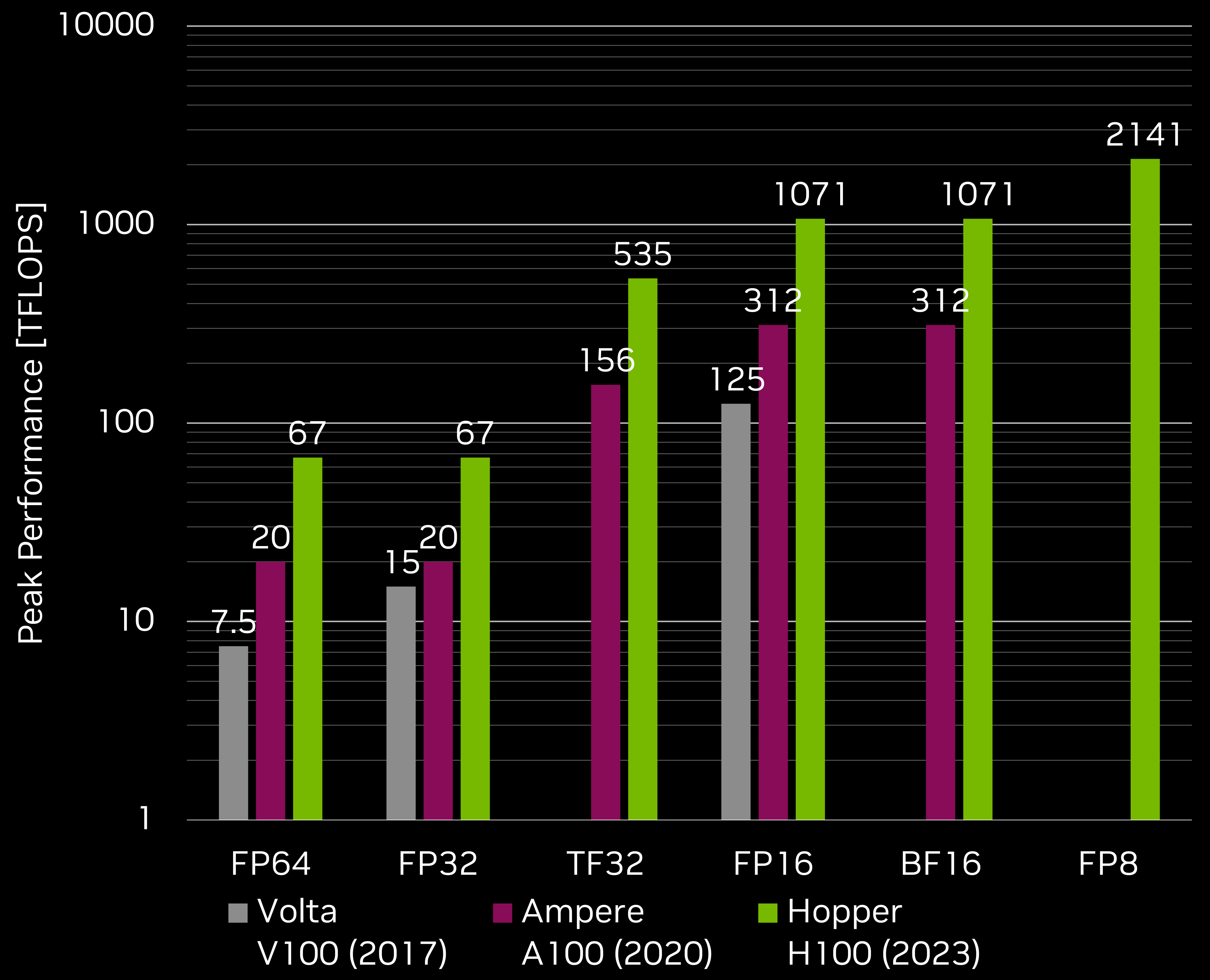
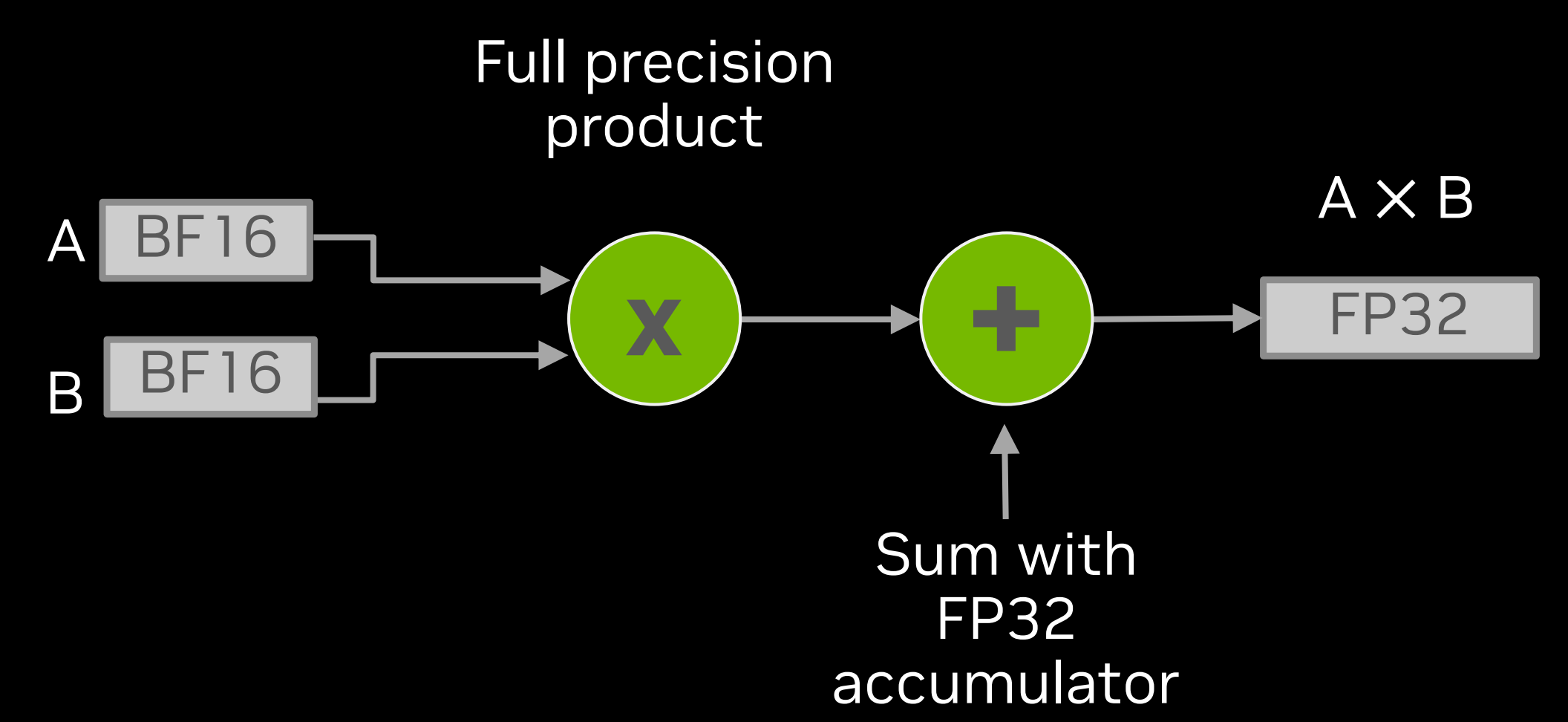
Part 2: Mixed-Precision scientific computing with Tensor Cores

Harun Bayraktar
Cole Brower
Cherin Joseph
Jack Kosaian
Dmitry Lyakh
Lukas Mosimann
Victor Podlozhnyuk
Paul Springer
Haicheng Wu

Tensor Core Performance Across GPU Generations



$$\text{value} = (-1)^{\text{sign}} \times 2^{\text{exponent}} \times (1 + \text{mantissa})$$



What types of things can we do to take advantage of Tensor Cores?

Non-Exhaustive Short List

- Tensor Cores have provided mixed-precision algorithms
algorithm developers increased opportunities and motivation [1]
- Algorithms that facilitate drop-in replacement for common functions
 - Matrix-multiply implementations
 - **Emulate full range and accuracy**
 - **Works for all use cases and can be made default**
 - Emulate partial range and accuracy [2]
 - Works for some use cases but cannot be made default
 - FFTs [3]
- New algorithms that use mixed-precision
 - Iterative refinement linear system solver with fallbacks [4]
 - Can be drop-in for LAPACK <T>GETRF/S

C = A · B algorithm using TF32 tensor cores for FP32 GEMMs [2]



$$A_{\text{low}} \leftarrow \text{toLow}(A_{\text{FP32}})$$

$$\Delta A_{\text{low}} \leftarrow \text{toLow}(A_{\text{FP32}} - \text{toF32}(A_{\text{low}})) \times 2^{-11}$$

$$B_{\text{low}} \leftarrow \text{toLow}(B_{\text{FP32}})$$

$$\Delta B_{\text{low}} \leftarrow \text{toLow}(B_{\text{FP32}} - \text{toF32}(B_{\text{low}})) \times 2^{-11}$$

$$C_{\text{F32}} \approx A_{\text{low}} \cdot B_{\text{low}}$$

$$+ (\Delta A_{\text{low}} \cdot B_{\text{low}} + A_{\text{low}} \cdot \Delta B_{\text{low}}) \times 2^{-11}$$

$$+ \Delta A_{\text{low}} \cdot \Delta B_{\text{low}} \times 2^{-22} \quad \leftarrow \text{Can this term be ignored?}$$

[1] Abdelfattah A, Anzt H, Boman EG, et al. A survey of numerical linear algebra methods utilizing mixed-precision arithmetic. The International Journal of High Performance Computing Applications. 2021;35(4):344-369. doi:10.1177/10943420211003313

[2] Hiroyuki Ootomo and Rio Yokota, Recovering single precision accuracy from Tensor Cores while surpassing the FP32 theoretical peak performance, 2022 <https://arxiv.org/pdf/2203.03341.pdf>

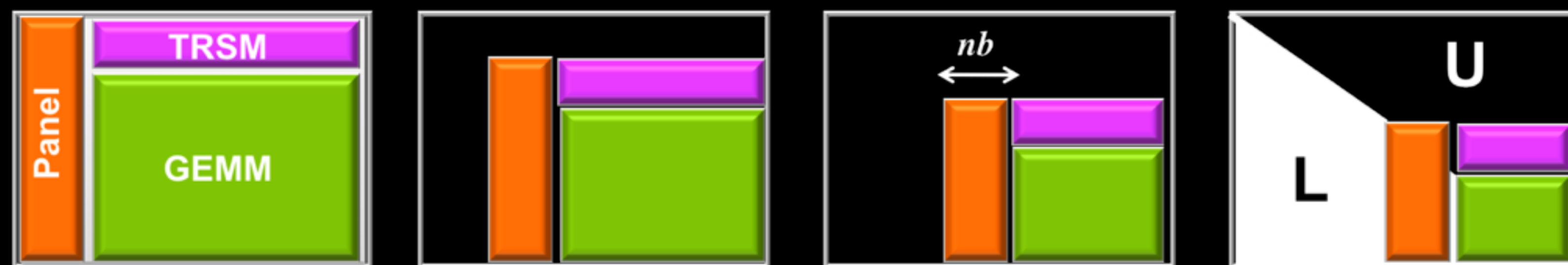
[3] L. Pisha and Ł. Ligowski, "Accelerating non-power-of-2 size Fourier transforms with GPU Tensor Cores," 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Portland, OR, USA, 2021, pp. 507-516, doi: 10.1109/IPDPS49936.2021.00059.

[4] Haidar Azzam, Bayraktar Harun, Tomov Stanimire, Dongarra Jack and Higham Nicholas J. 2020Mixed-precision iterative refinement using tensor cores on GPUs to accelerate solution of linear systems, Proc. R. Soc. A.4762020011020200110 <https://royalsocietypublishing.org/doi/10.1098/rspa.2020.0110>

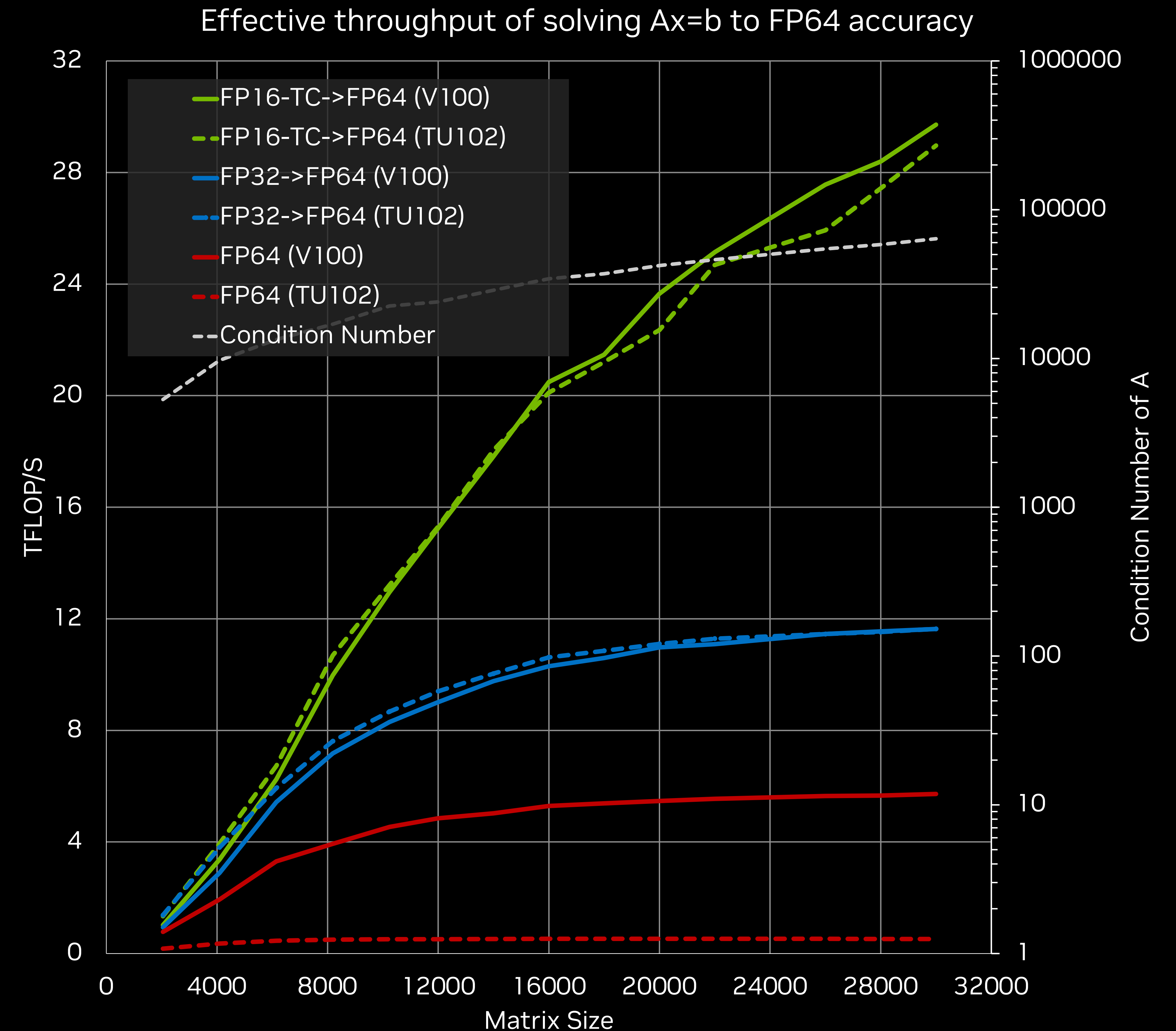
Tensor Core Accelerated Iterative Refinement Solver

FP64 accuracy linear system solution

- Iterative refinement method [1]
 - Dense Linear Solver for $Ax=b$
 - Can substitute $\langle T \rangle$ GETRF/S from LAPACK
 - Main idea:
 - Move $O(n^3)$ operations during LU factorization to mixed-precision Tensor Cores

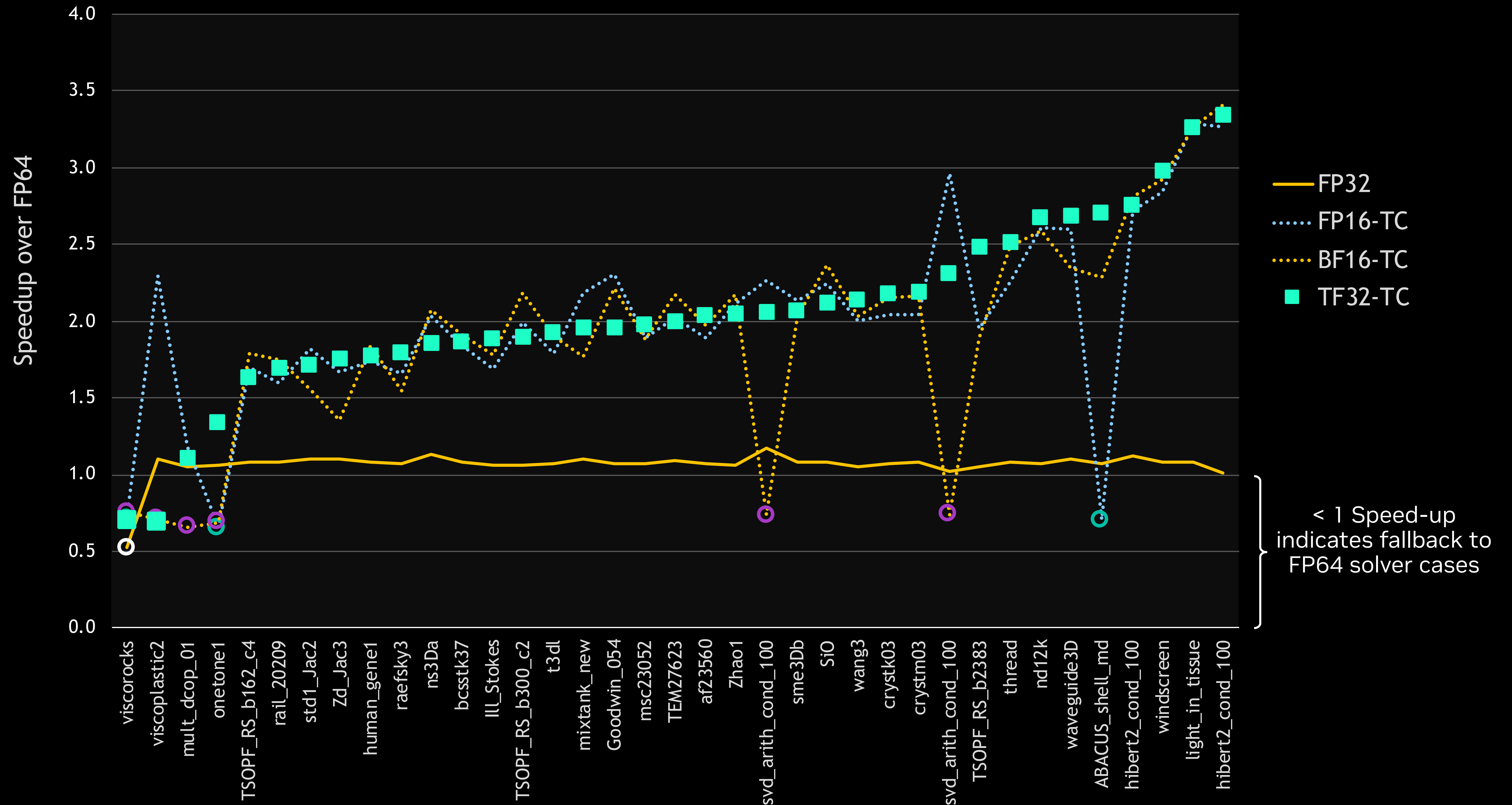


- Use the factor as a preconditioner for an iterative solver at FP64 where operations have $O(n^2)$ complexity
- **Reduces reliance on FP64 compute throughput** while delivering better performance
- Top500 benchmark HPL-MxP (formerly HPL-AI) uses this approach



Tensor Core Accelerated Iterative Refinement Solver

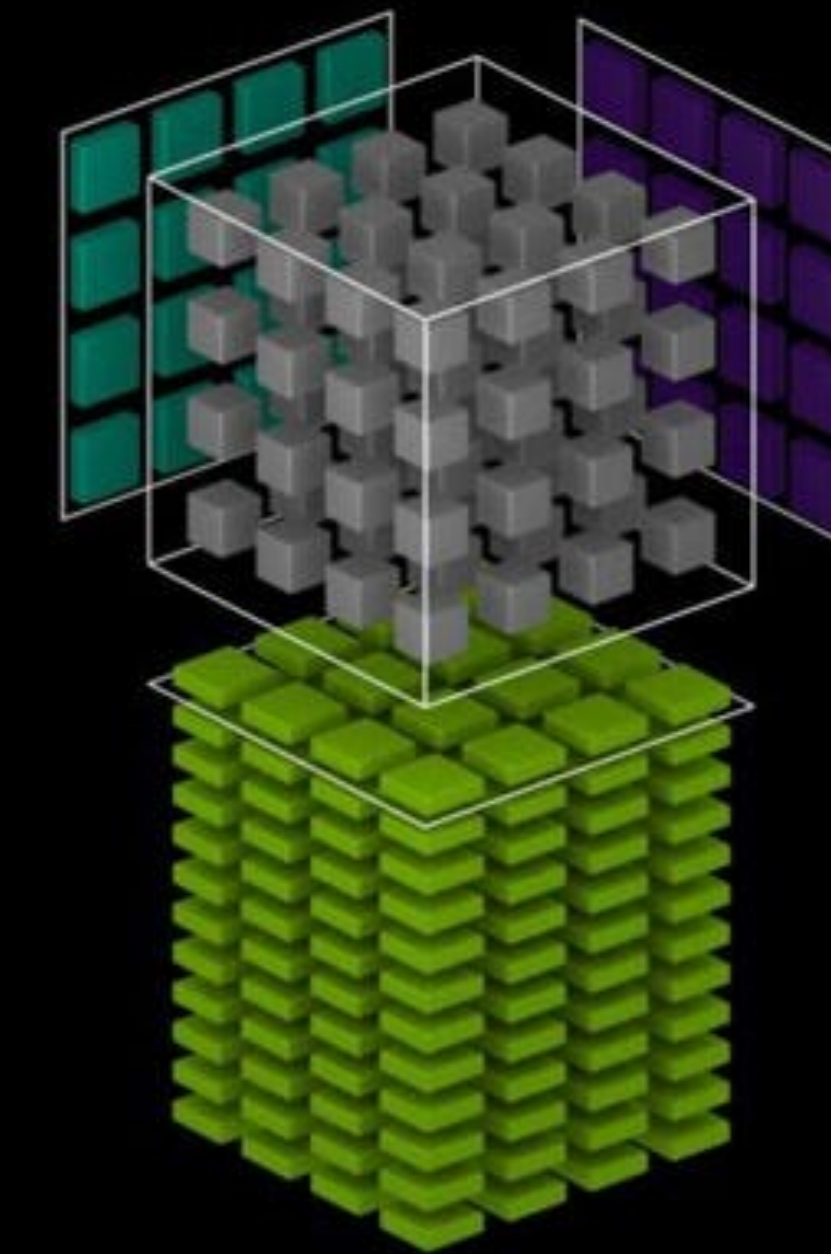
Speed-ups relative to full FP64 baseline solver



Goal

Accelerate single precision (FP32) matrix multiplies without any loss of range or accuracy

- Algorithm that uses BF16 Tensor Cores
- Study accuracy of implementation for corner cases
- Prototype implementation in cuBLAS and cuTENSOR
- Validate accuracy with real world applications

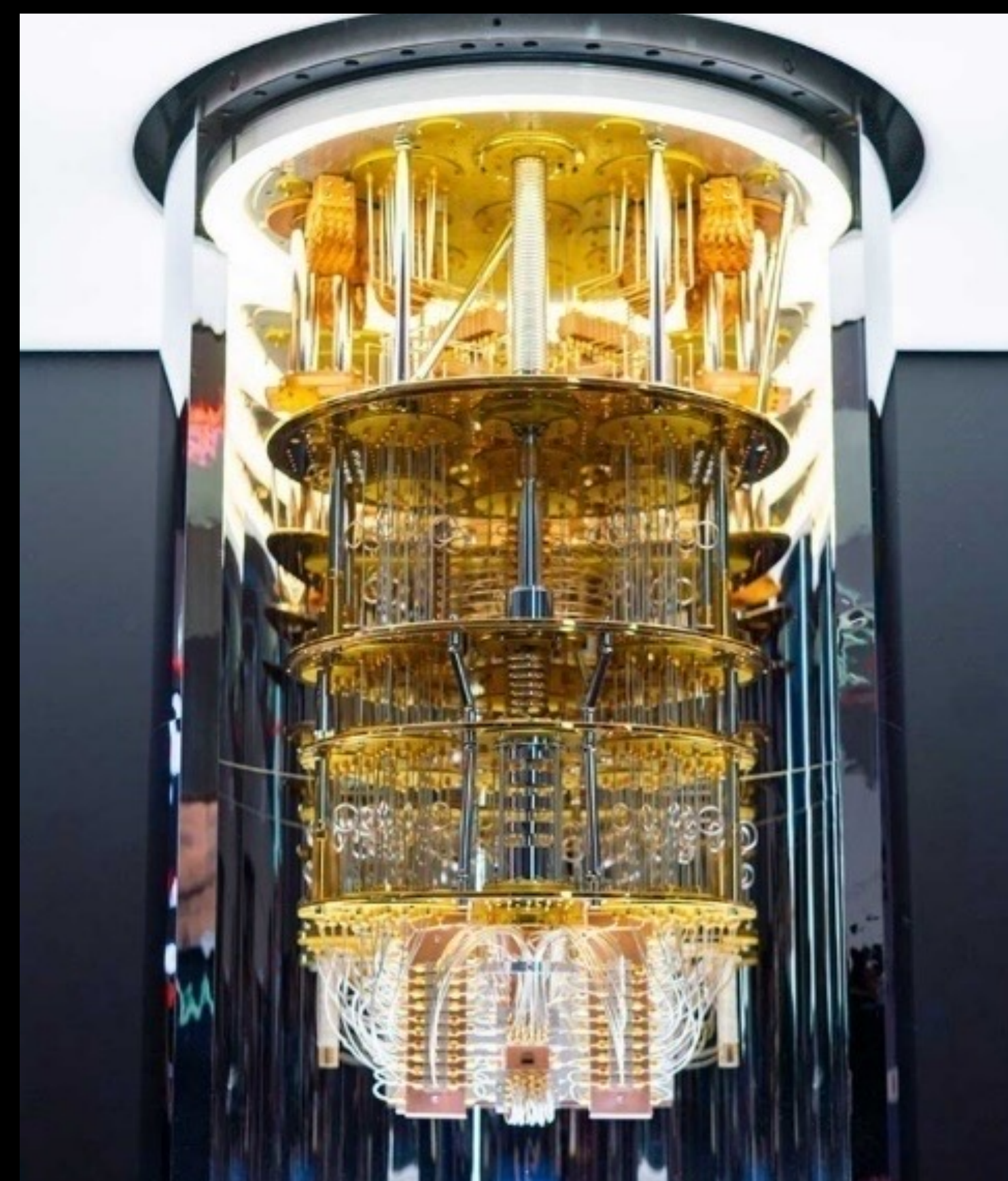


Baseline implementations in cuBLAS and cuTENSOR use single precision IEEE754 FMA instructions

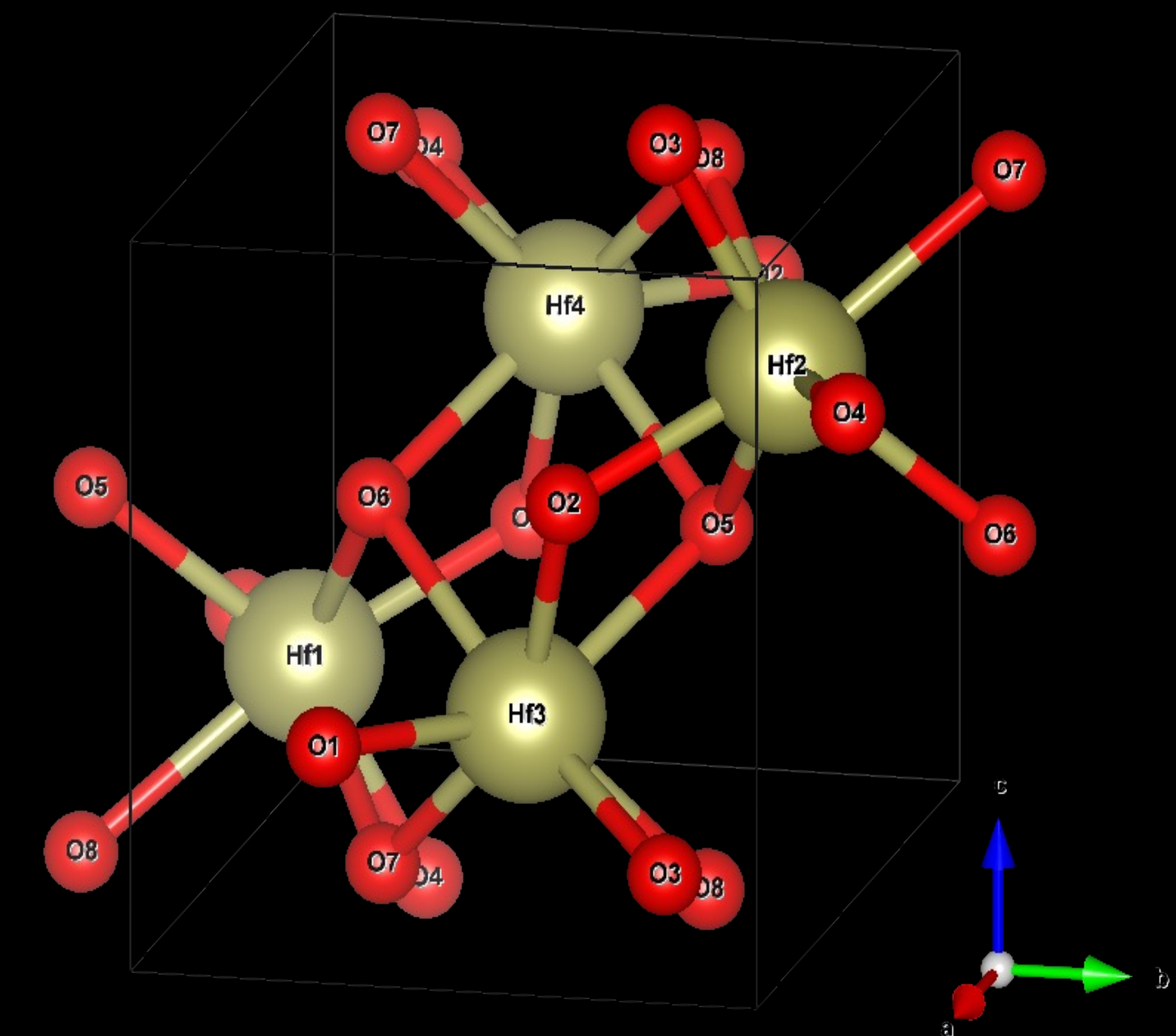
Weather Forecasting



Quantum Computing



Condensed Matter Physics



Algorithm Description

BF16/9 Algorithm

- The FP32 inputs are decomposed into 3 scaled BF16 components

$$a = a_0 + 2^{-8} \cdot a_1 + 2^{-16} \cdot a_2$$

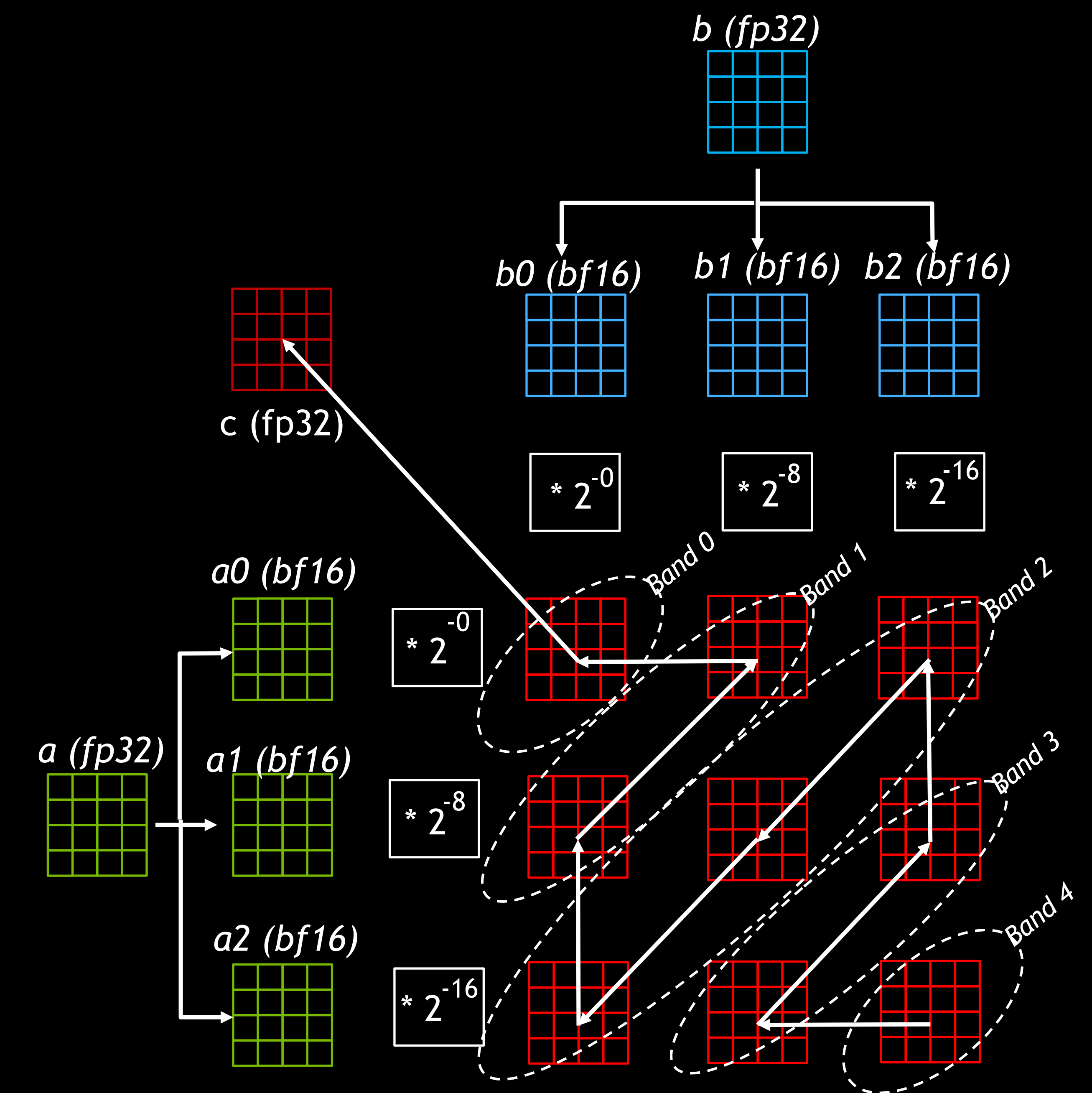
$$b = b_0 + 2^{-8} \cdot b_1 + 2^{-16} \cdot b_2$$

- The Inputs are decomposed using the CUDA cores

- The multiply-add operation is computed as a sum of 9 scaled partial products

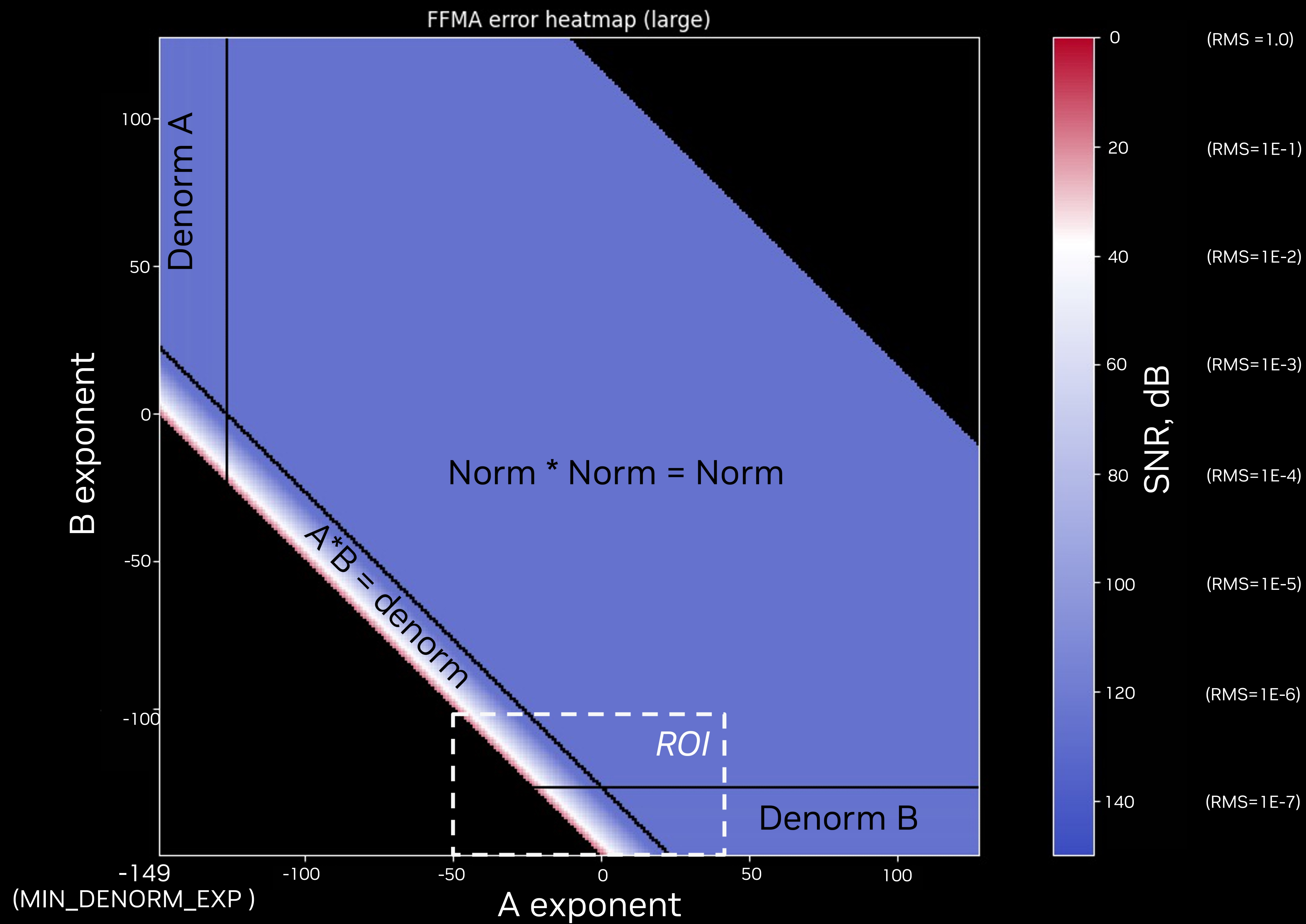
$$\begin{aligned} a * b + c = & a_0 \cdot b_0 + 2^{-8} \cdot a_0 \cdot b_1 + 2^{-16} \cdot a_0 \cdot b_2 \\ & + 2^{-8} \cdot a_1 \cdot b_0 + 2^{-16} \cdot a_1 \cdot b_1 + 2^{-24} \cdot a_1 \cdot b_2 \\ & + 2^{-16} \cdot a_2 \cdot b_0 + 2^{-24} \cdot a_2 \cdot b_1 + 2^{-32} \cdot a_2 \cdot b_2 + c \end{aligned}$$

- The partial products are computed in the BF16 Tensor cores
- The partial products are scaled appropriately in the CUDA cores
- The tensor cores and CUDA cores work in parallel
- The effective FP32 FLOPs is 1/9th that of the BF16 tensor core FLOPs
 - On H100 119 vs 67 TFLOP/s → **~1.8X maximum speed-up**



Measuring Accuracy

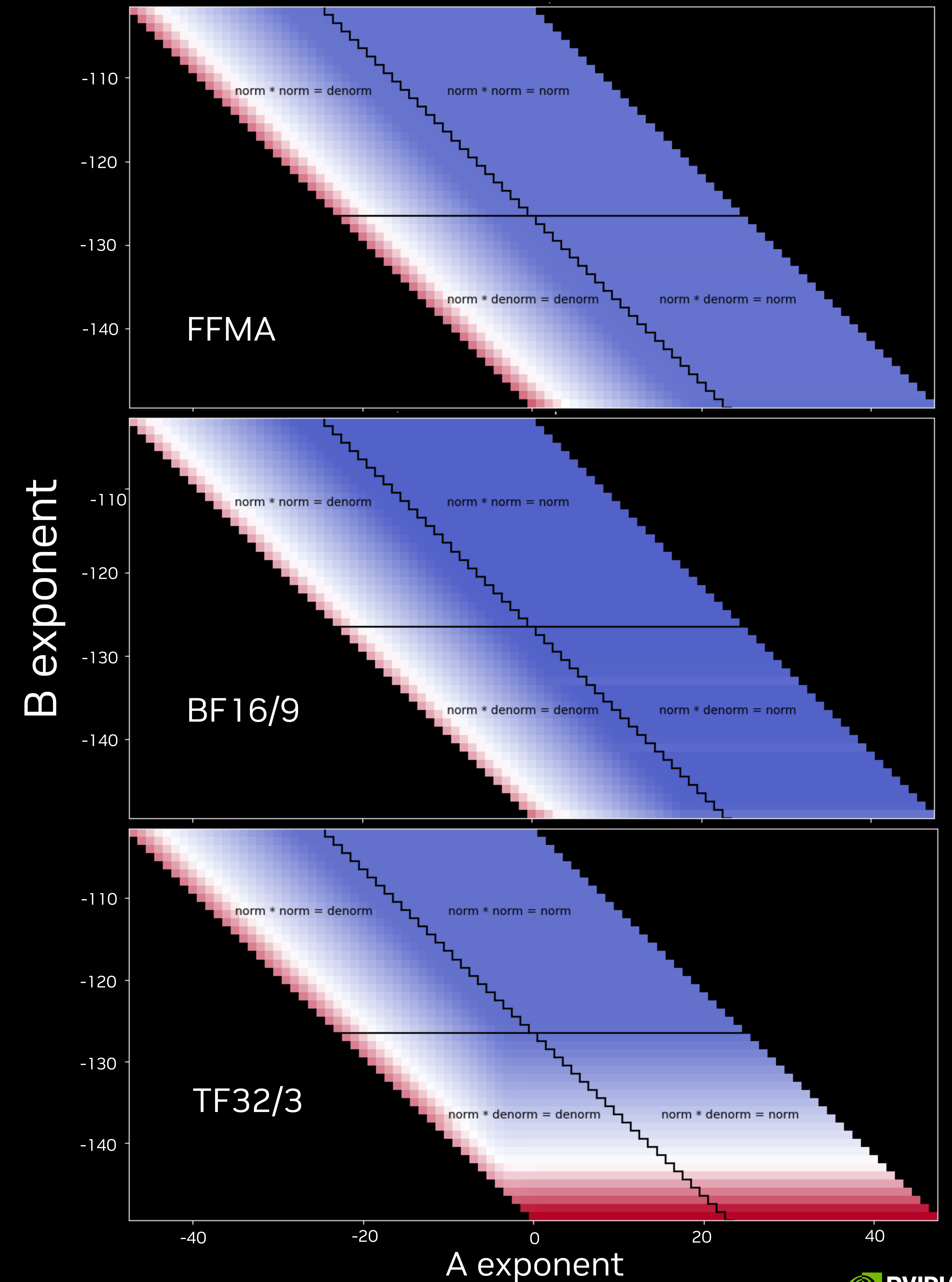
Smoke test: testing different uniform-exponent combinations



Matrix mul dimensions: [512 x 2048] = [512 x 1024] * [1024 x 2048]

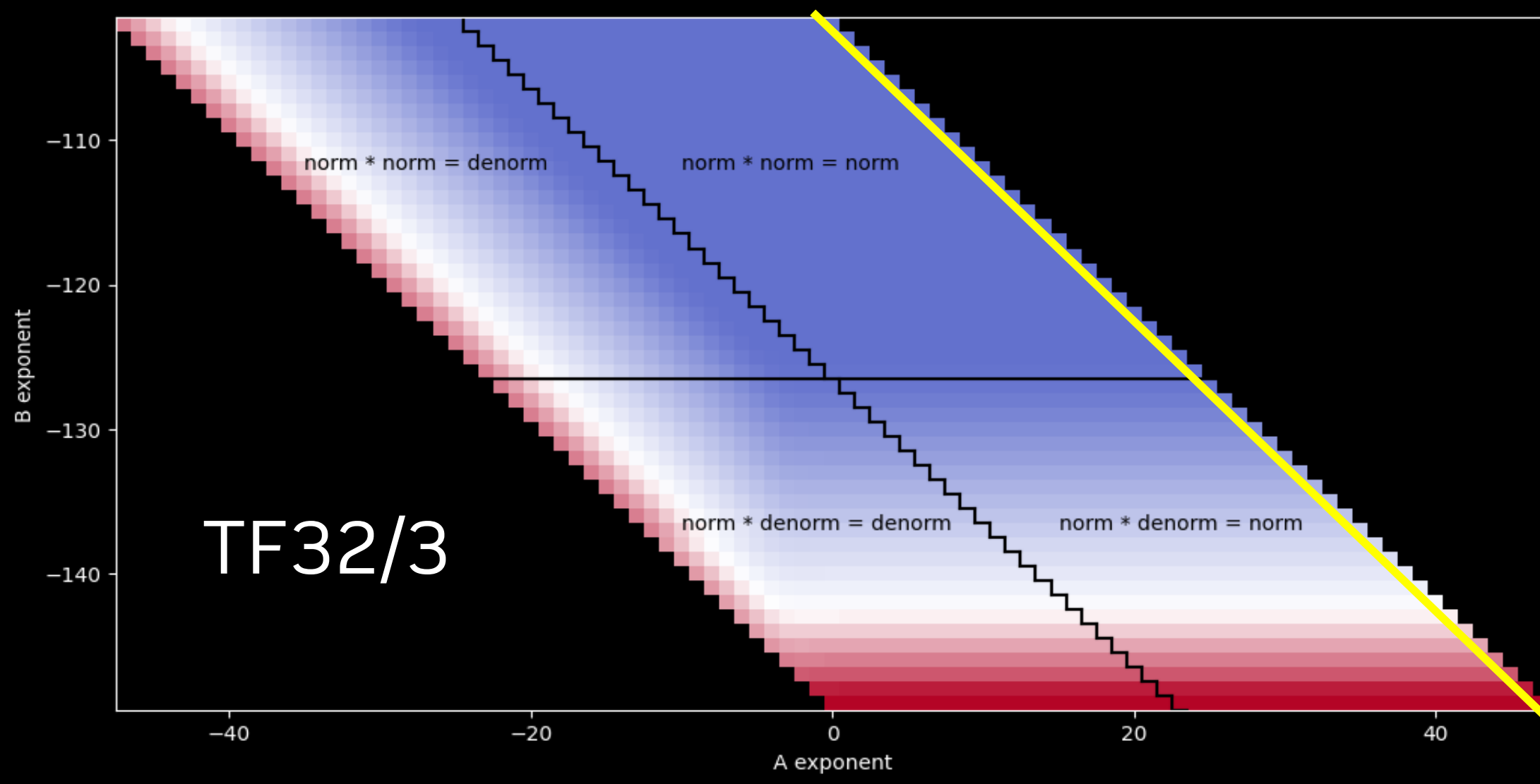
$$RMS = \sqrt{\frac{\sum_{r,c}\{Result_{r,c} - Result_{fp64_{r,c}}\}^2}{\sum_{r,c}\{Result_{fp64_{r,c}}\}^2}}$$

$$SNR = -20.0 * \log_{10} \sqrt{\frac{P_{err}}{P_{ref}}} = -20.0 * \log_{10} RMS$$

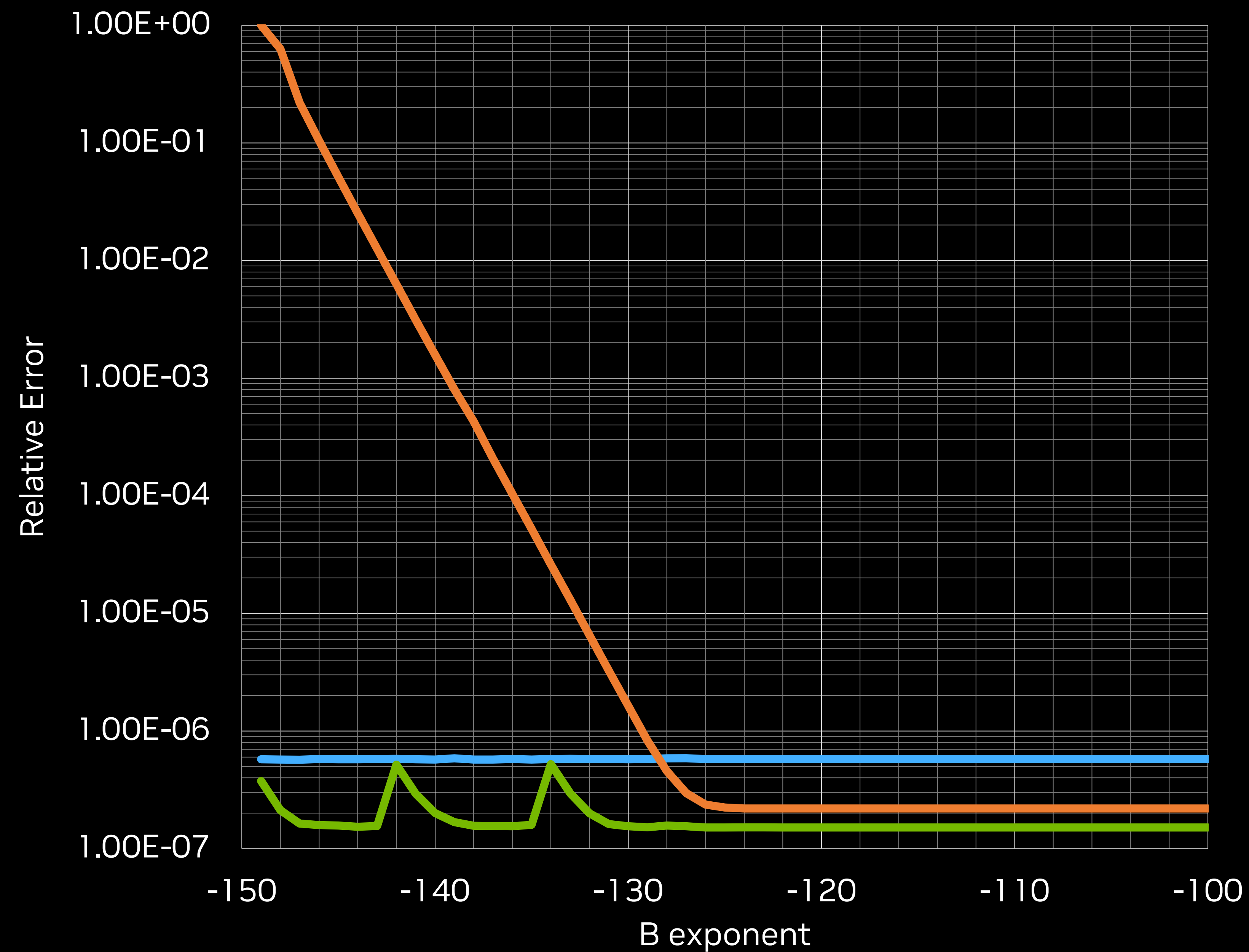


Numerical Accuracy Study

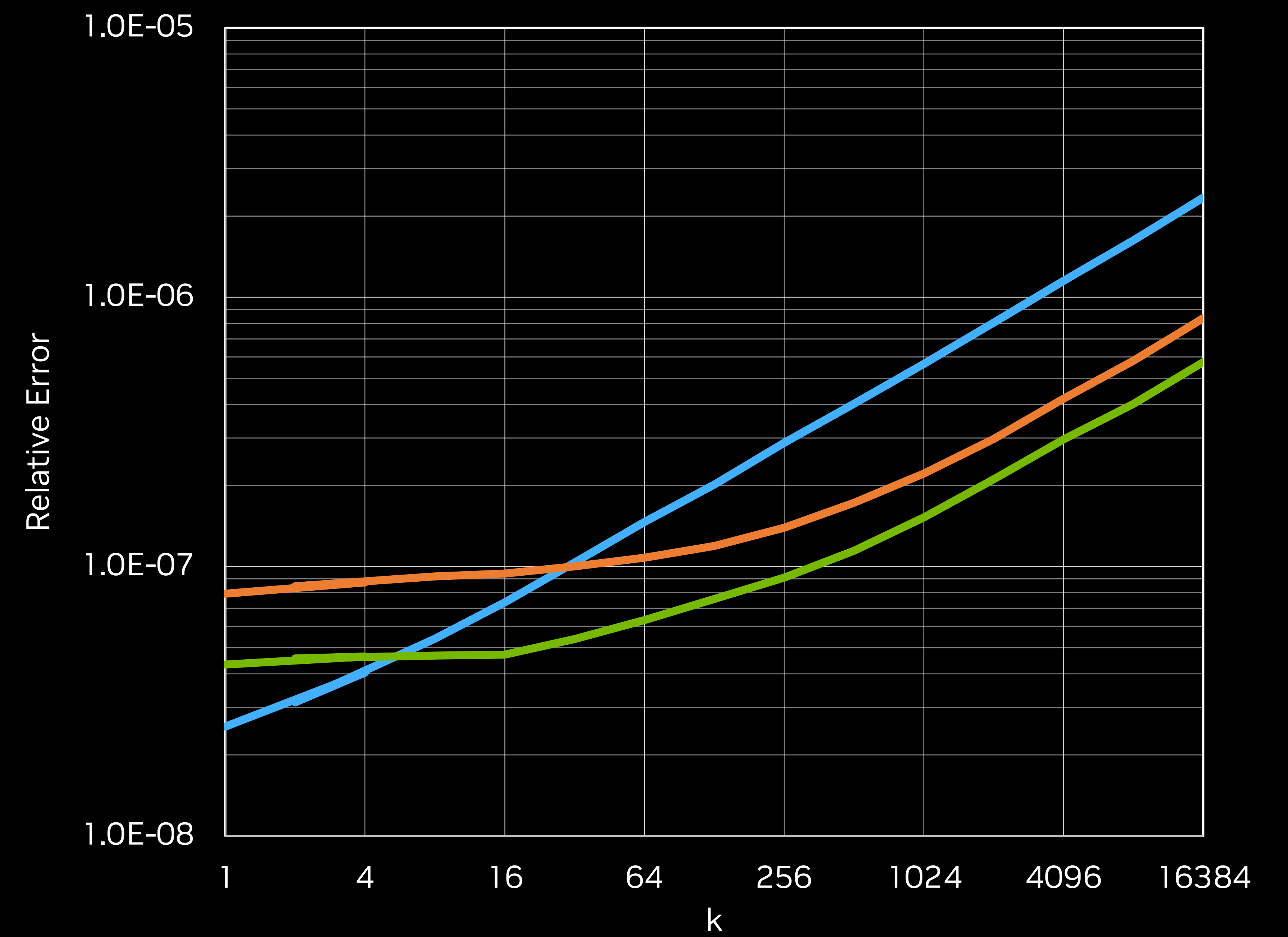
Results for two different data sets



ROI subset: $A_exp + B_exp = -102$ as 1D graph, (K = 1024)



Normal Distribution Dataset [1]

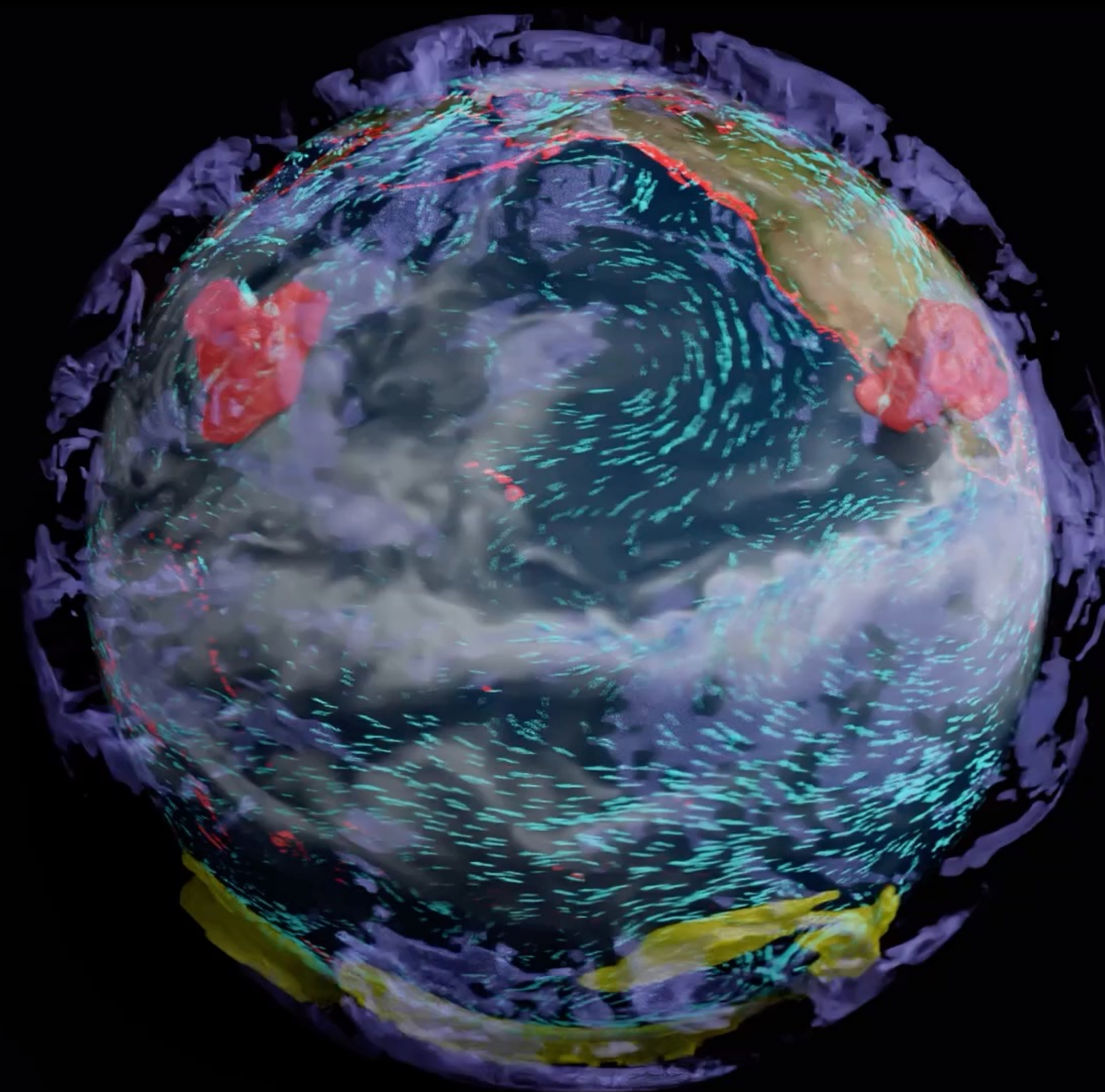


[1] Hiroyuki Ootomo and Rio Yokota, Recovering single precision accuracy from Tensor Cores while surpassing the FP32 theoretical peak performance, 2022 <https://arxiv.org/pdf/2203.03341.pdf>

Weather Forecast Simulation

Accelerate Spectral Transforms

- IFS: Integrated Forecast System from ECMWF
- Weather models moving to higher and higher resolution (e.g. TCo3999 ~ 2.5km global resolution). Spectral transform is a major bottleneck (>50% of the cost)
 - Spectral transform = All2All + FFT + All2All + GEMM
→ Scaling Bottlenecks are All2All and GEMMs
- Mid-term we aim to run weather models at a resolution of ~1 km because that would allow explicitly resolving convection
- Ectrans is the spectral transform library extracted from the full model. <https://github.com/ecmwf-ifs/ectrans>
- We use Ectrans to check accuracy of our emulated FP32 matrix multiply implementation



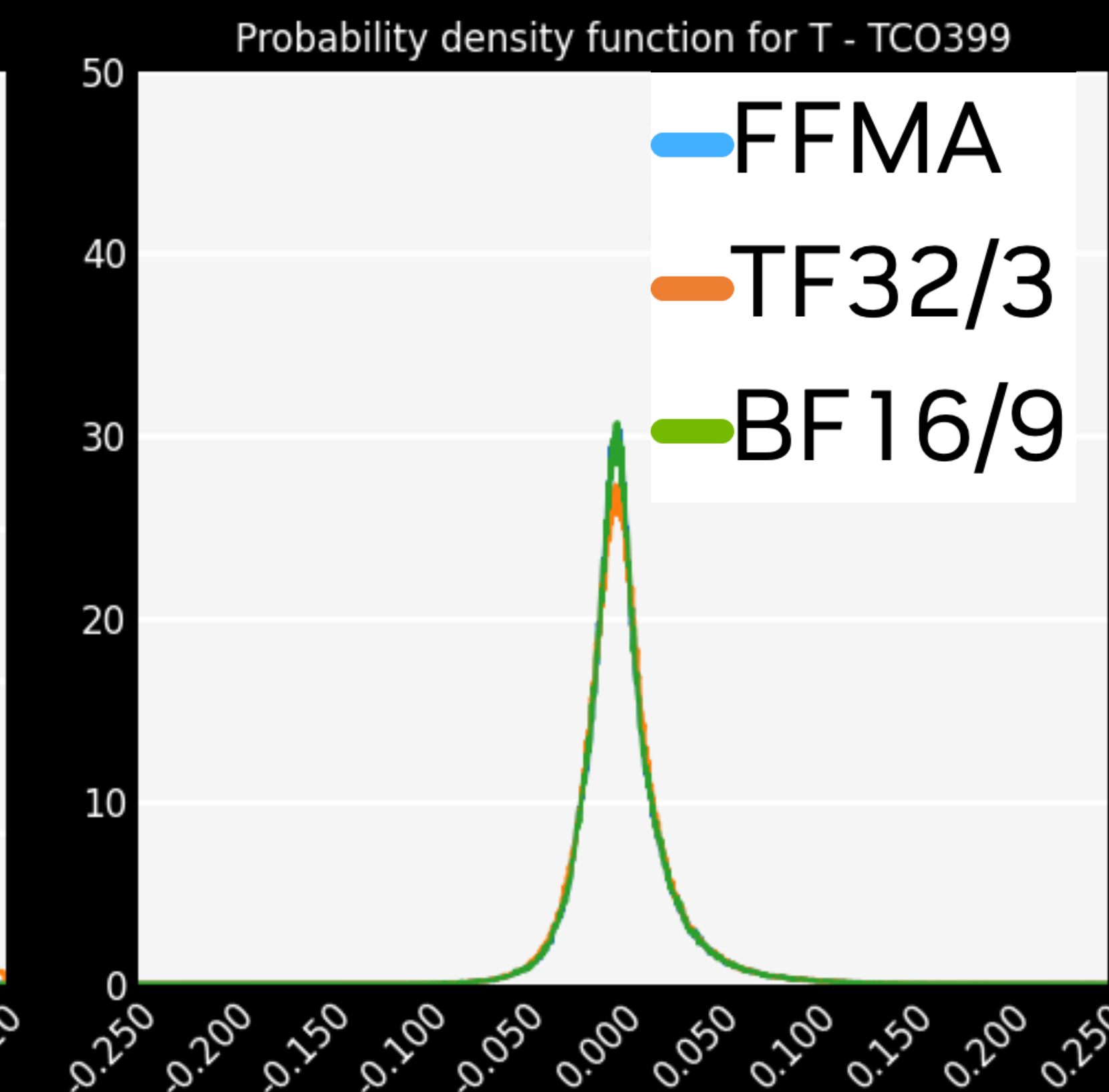
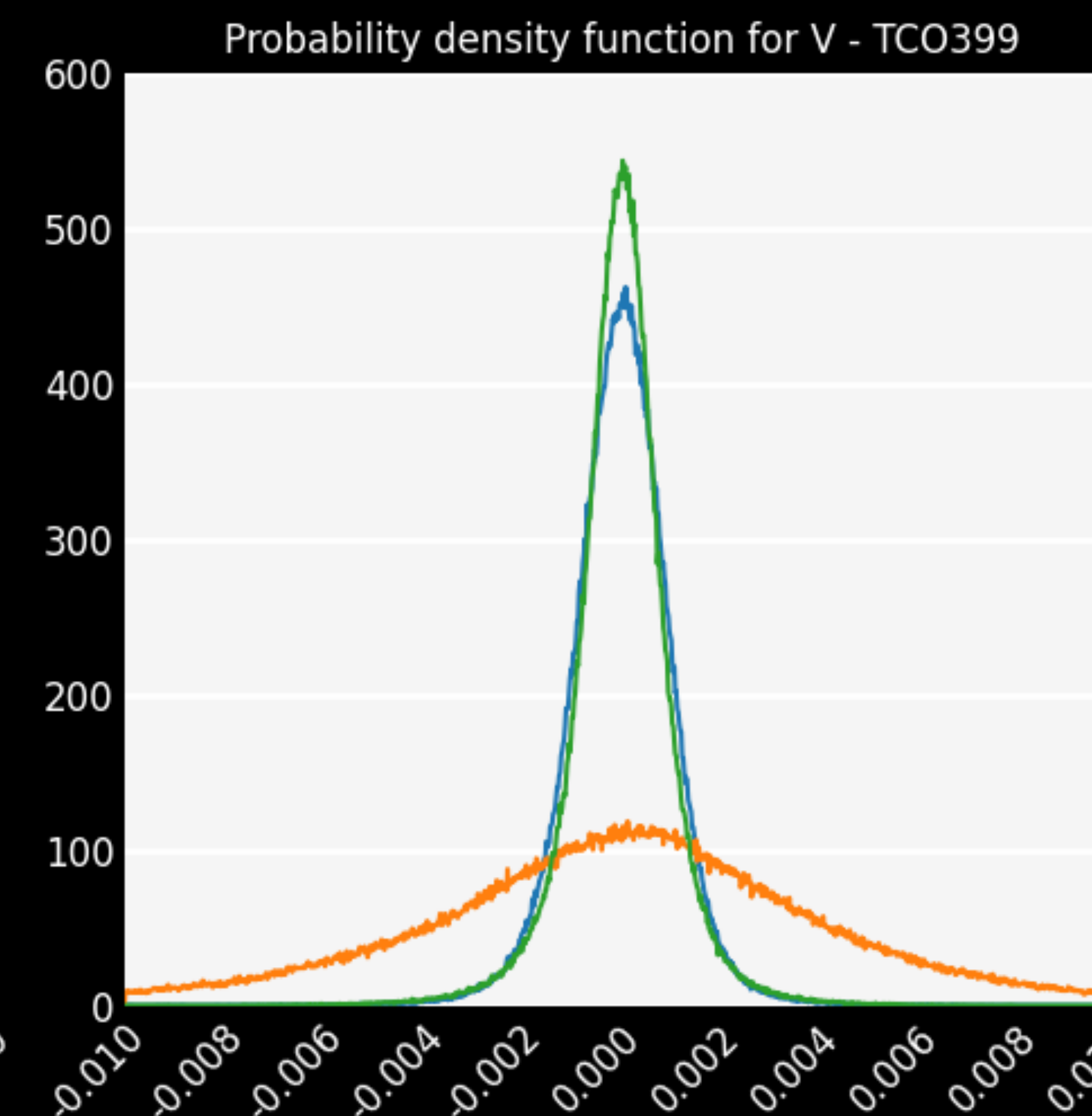
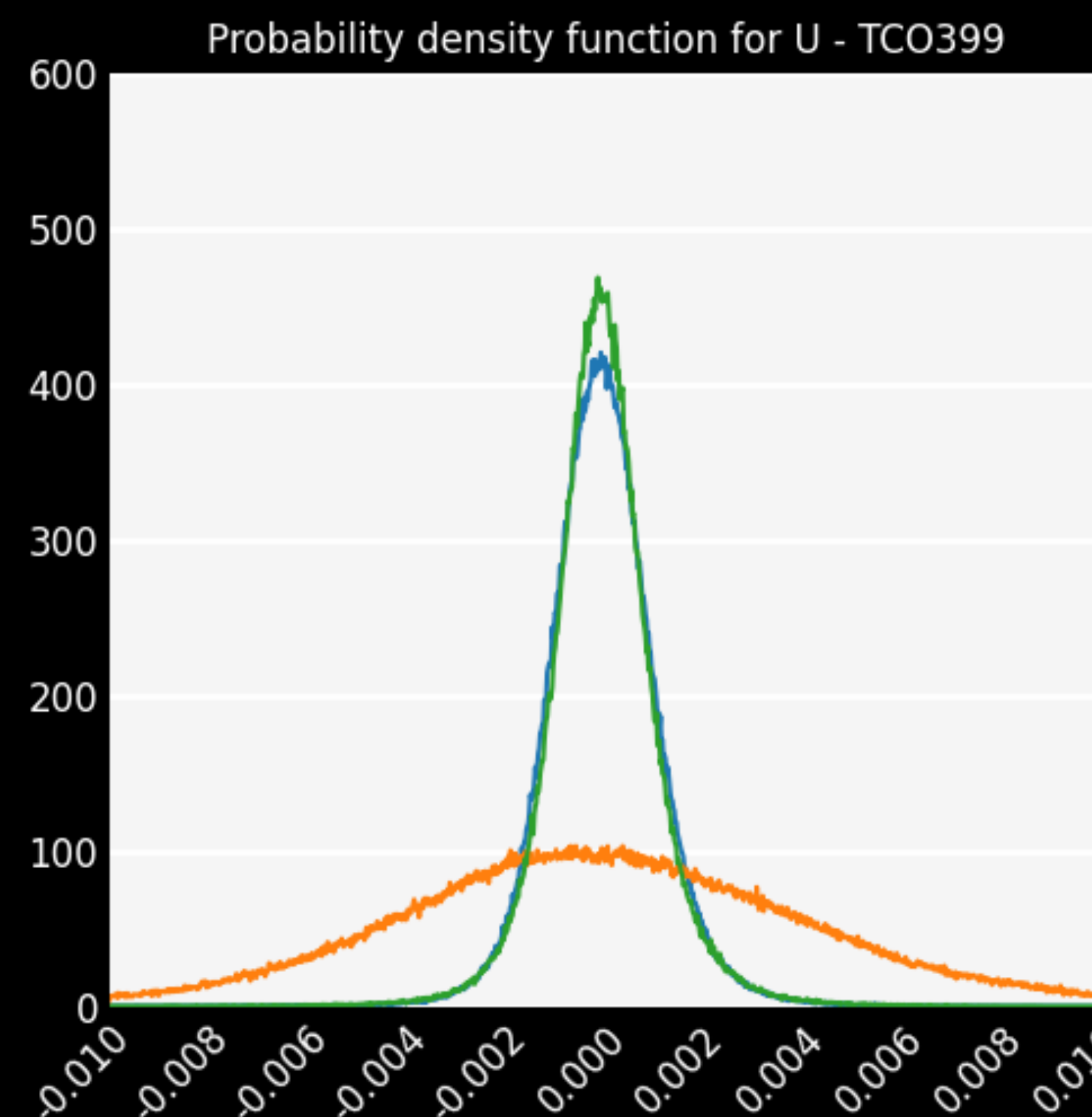
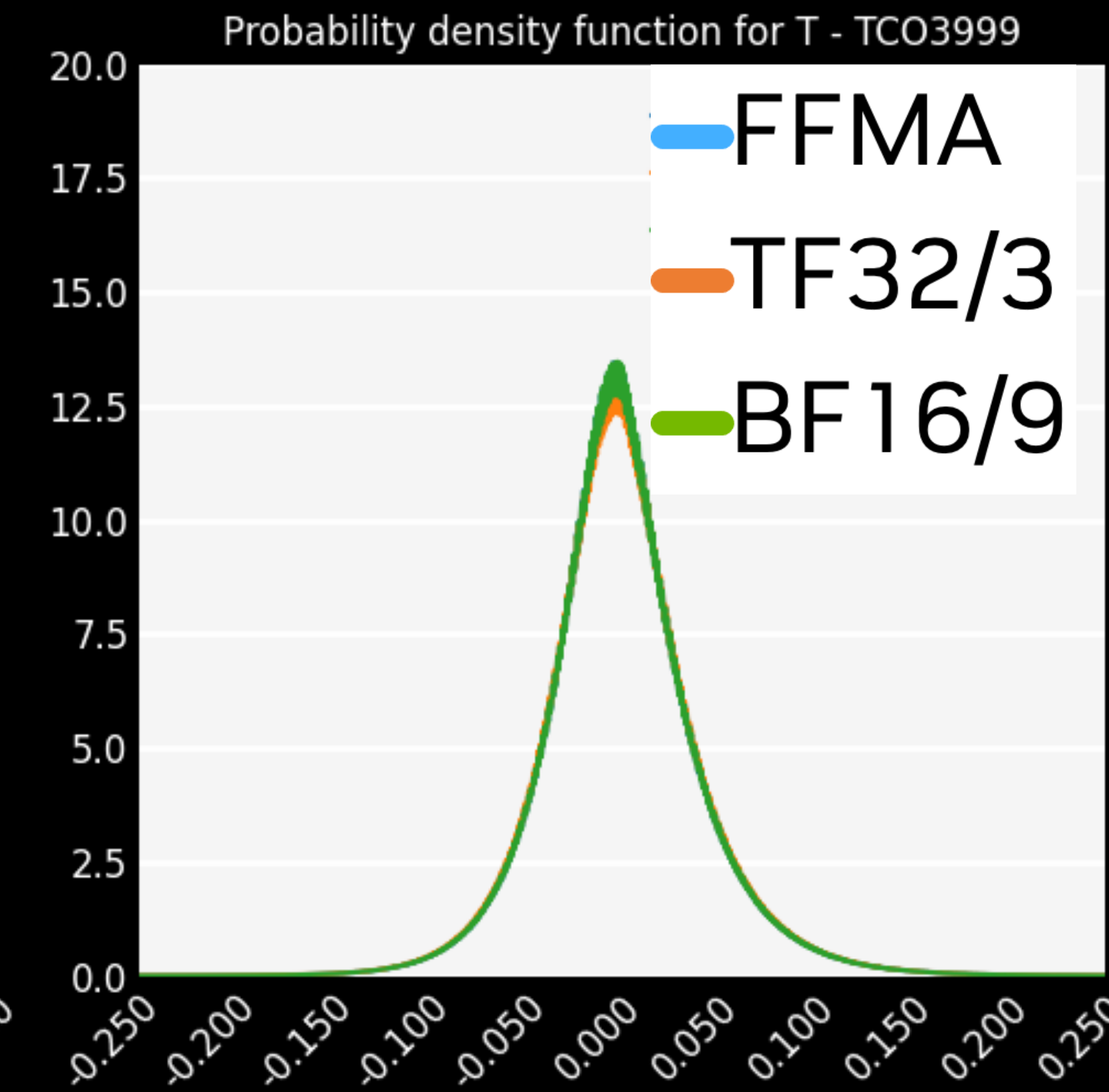
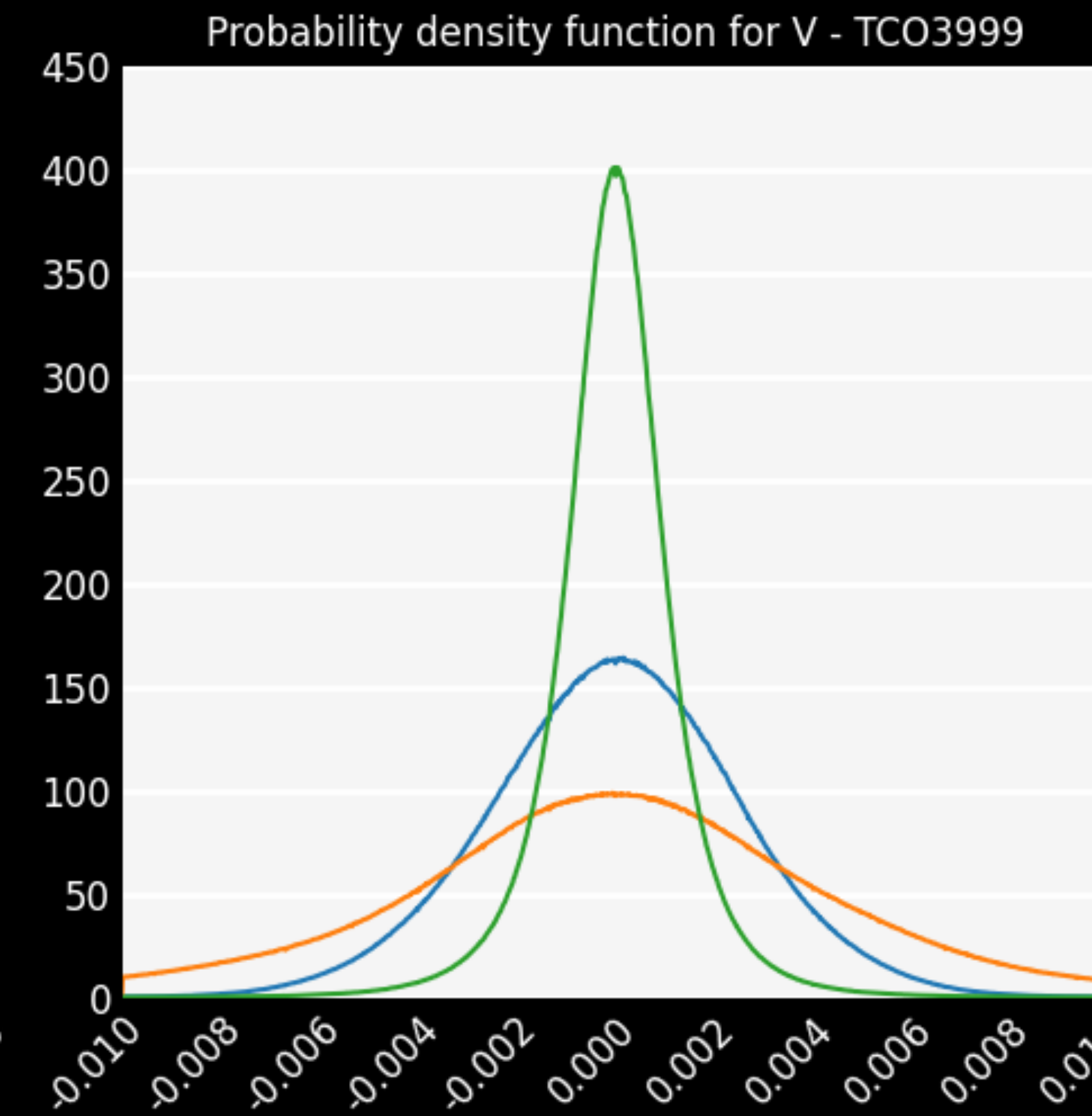
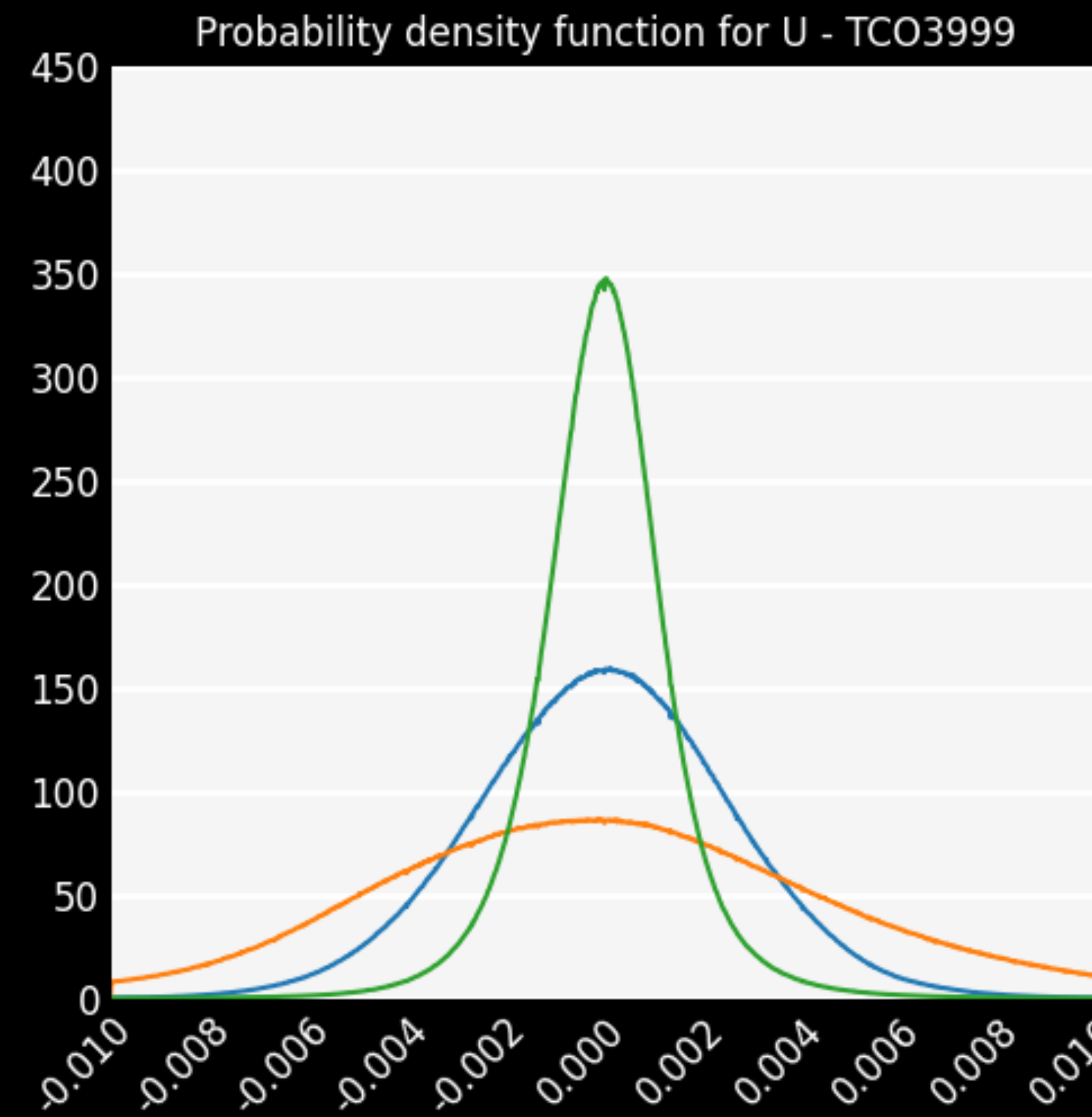
Weather Forecast Simulation

Accelerate Spectral Transforms

- Absolute deviations after 1000 iterations of forward and backward spherical transforms
- Ground truth is the original input
- In all cases, BF16X9 gives at least similar, for some variables significantly superior results
- Velocities can behave different compared to temperature because they have an additional conversion from u/v (in grid-point space) to divergence/vorticity (in spectral space)
- TF32 has significant (too large) deviations for temperature and explodes for velocities
 - TF32/3 implementation here is without scaling

Velocities: U, V

Temperature

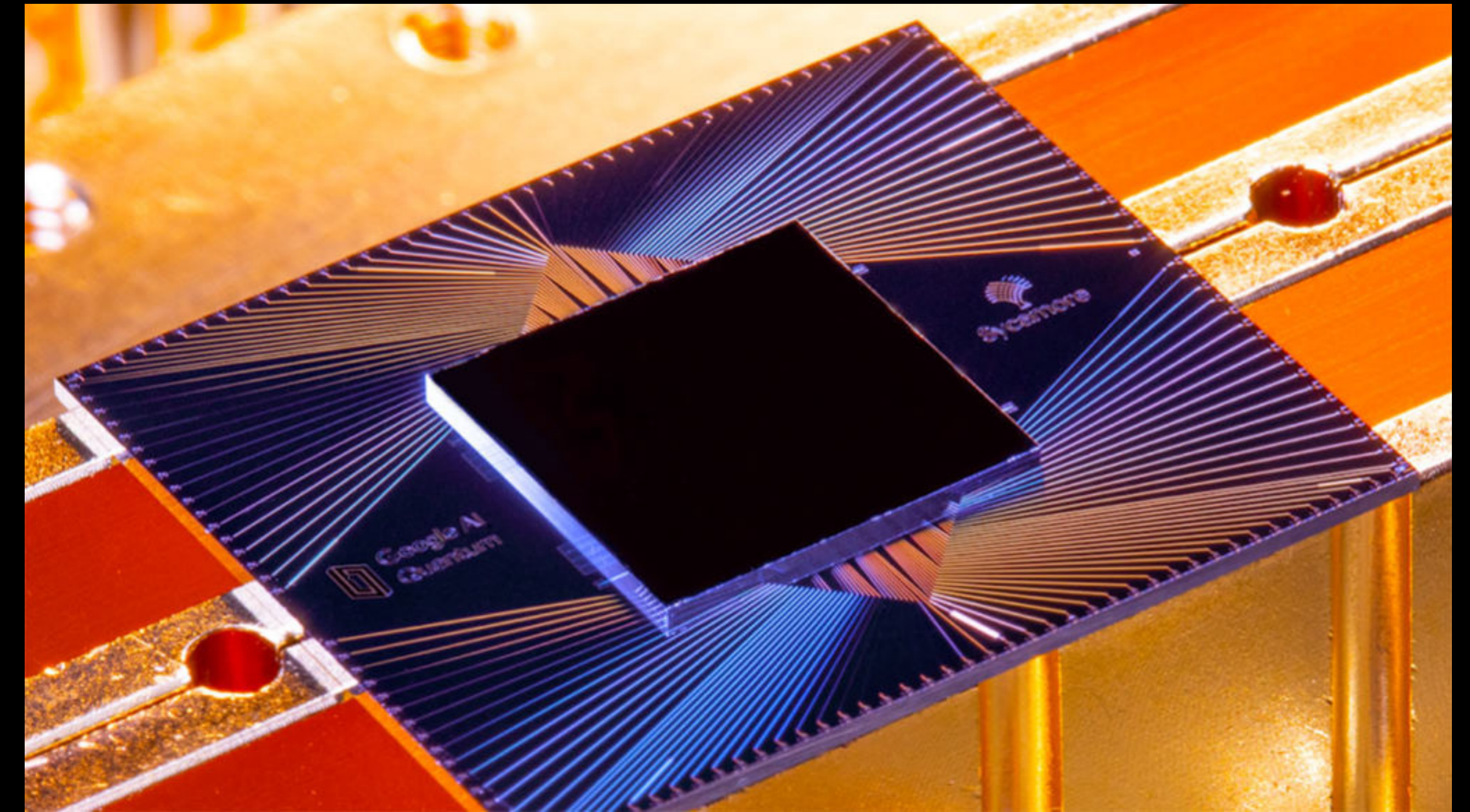


Quantum Computing Simulations

Accelerate Tensor Network Contractions

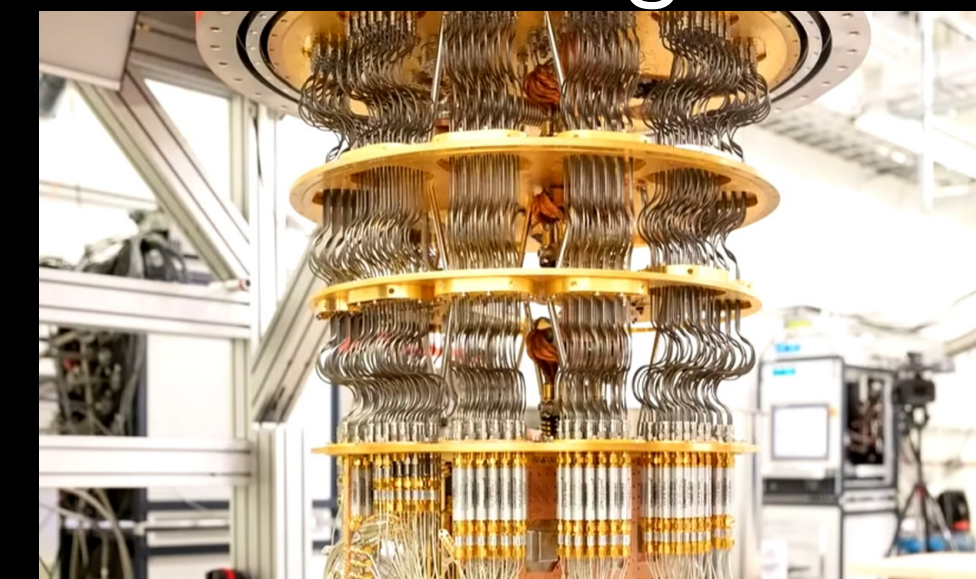
Google's Sycamore Quantum Chip (53 qubits)

- Validation of a quantum processor requires cross-entropy benchmarking against a simulation of a random quantum circuit of specific depth
- Simulation of a quantum circuit can be reduced to the contraction of a **tensor network** representing the circuit
- **cuTensorNet** library from **cuQuantum SDK** accelerates tensor network contractions on **NVIDIA GPUs**
- Tensor network contraction is expressed as a sequence of pairwise tensor contractions executed by the cuTensor library
- Algorithm development and validation of modern quantum chips is an extremely computationally demanding task that significantly benefits from GPU acceleration

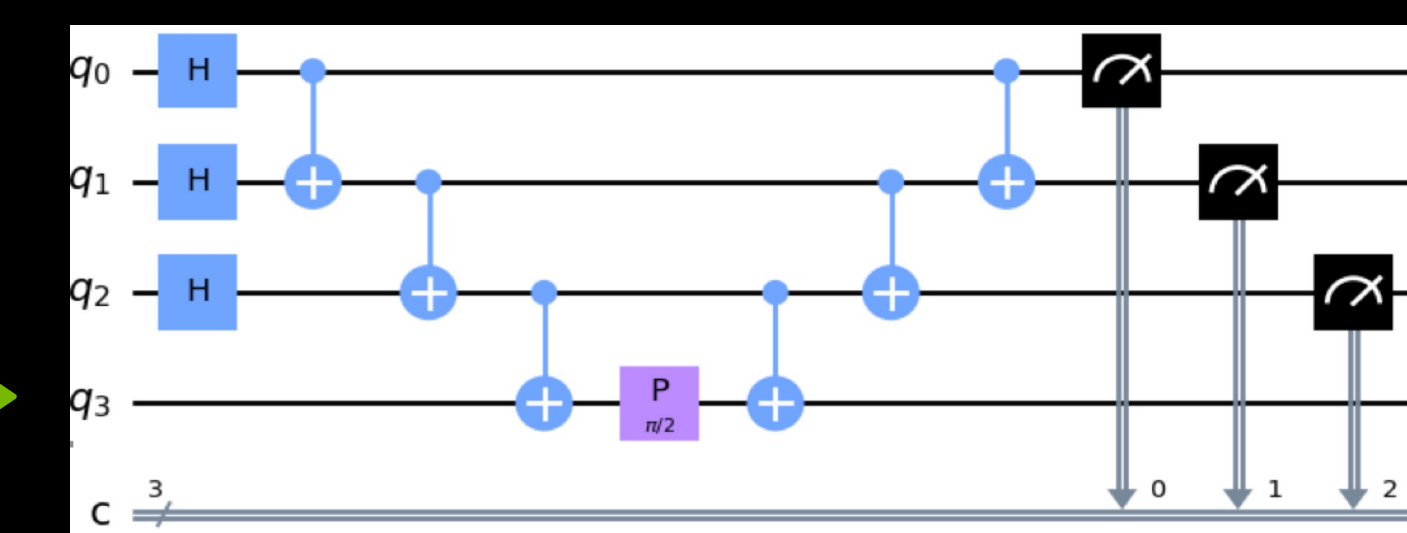


Arute, F., Arya, K., Babbush, R. *et al.* Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).

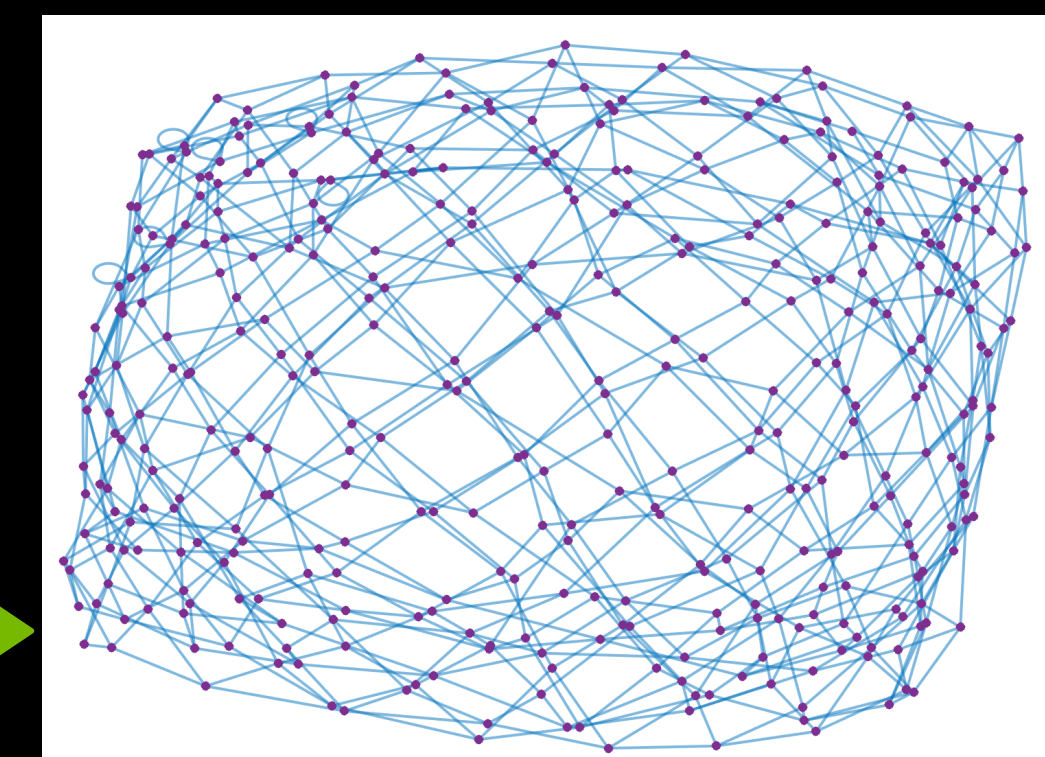
Quantum Algorithm



Quantum Circuit



Tensor Network



$$\beta = \langle \psi_r | U | \psi_0 \rangle$$

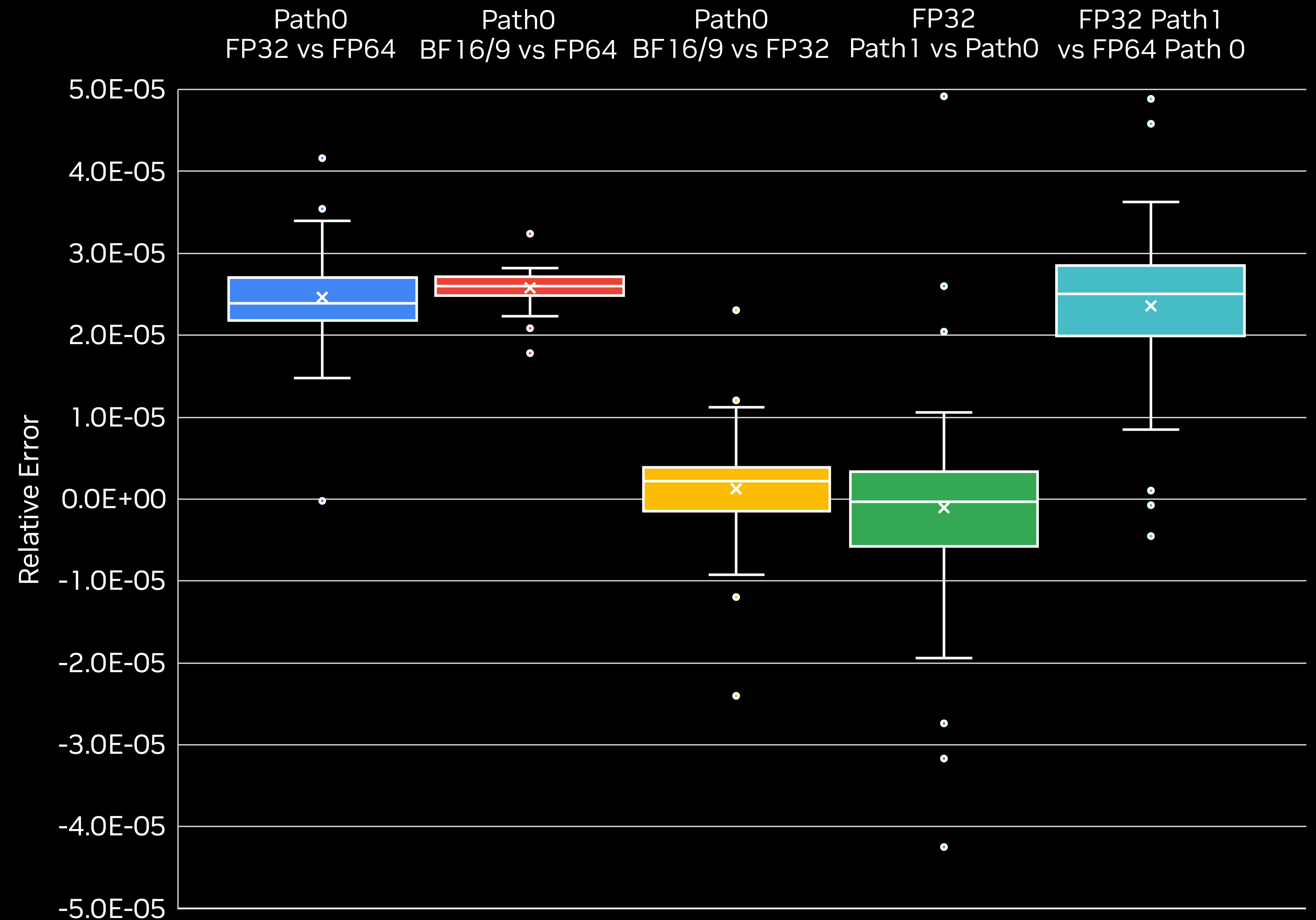
Quantum Computing Simulations

Accelerate Tensor Network Contractions

- We simulated the 53-qubit Sycamore quantum chip with 12 layers of random gates and computed probability amplitudes for 64 bit-strings

$$\beta = \langle \psi_r | U | \psi_0 \rangle$$

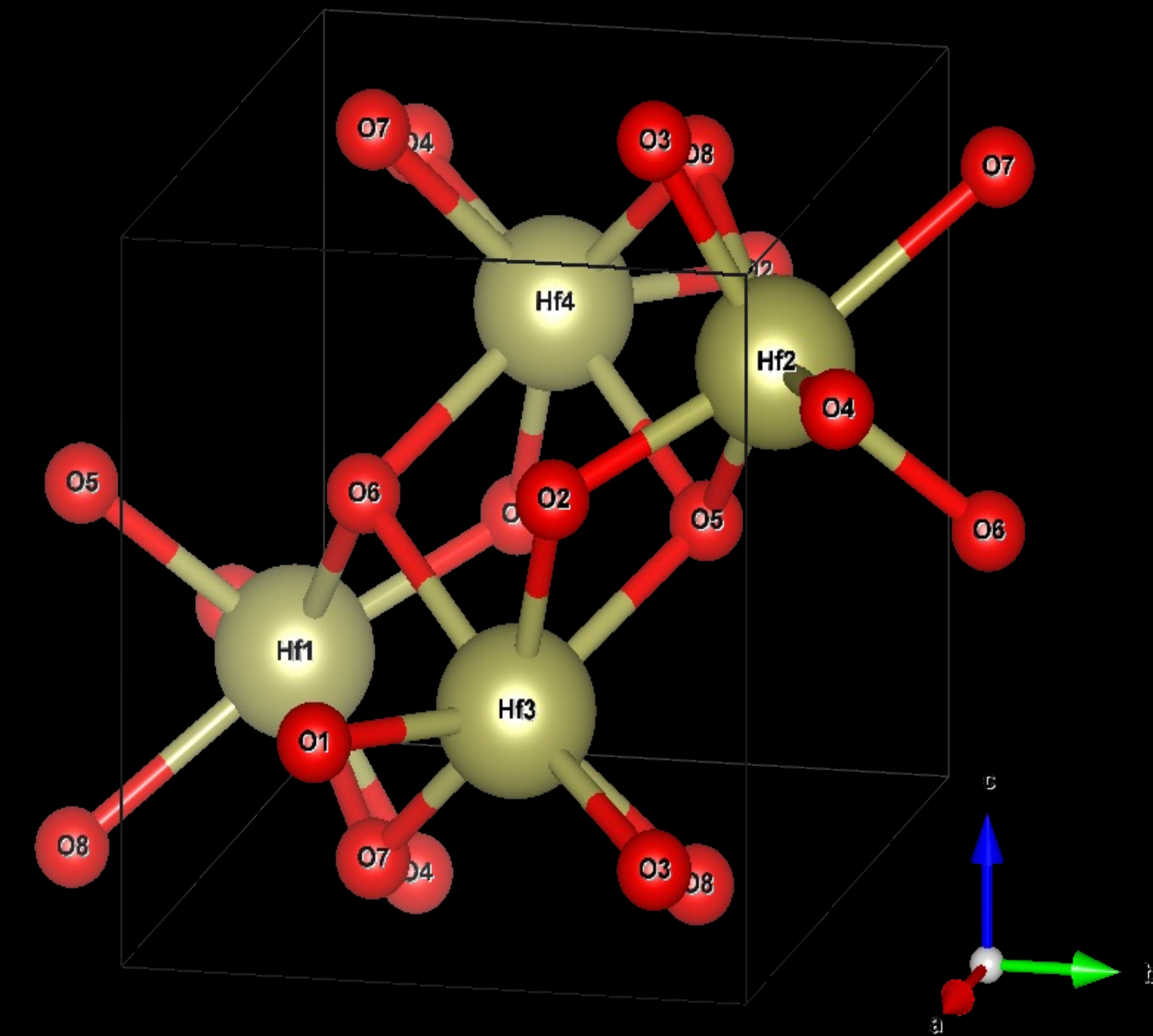
- 649216 total pairwise tensor contractions;
- 1.7% of the tensor contractions account for 95% of the total 0.83 PFLOPs (k-dim ≥ 16)
 - We offload these to **BF16/9**
- The relative error of the computed probability amplitudes with BF16/9 less than FP32 when compared to FP64 baseline
 - We offload these to **BF16/9**
- The variation of amplitude values due to the use of different tensor network contraction paths for FP32 compute introduces larger differences than BF16/9



Condensed Matter Physics Simulations

Accelerate Tensor Network Contractions

- Simulating **electronic structure of materials** is an extremely complex and computationally demanding task
- Widespread use of phenomenological models and Hamiltonians to reduce the complexity of the task
- The dimensionality of the corresponding linear Hilbert space grows **exponentially** with the number of simulated spins or electrons, thus mandating approximate solutions
- **Tensor network** theory provides a powerful systematic theoretical framework for approximating spin or electronic states of materials in highly-dimensional linear spaces
- The regular **linear-** and **eigen-**solvers can be reformulated in the language of tensor network theory, resulting in a drastic reduction of the computational cost due to the use of **tensor factorization**
- **Transverse-field Ising spin Hamiltonian** is used as a paradigmatic model for simulating a broad range of quantum phenomena



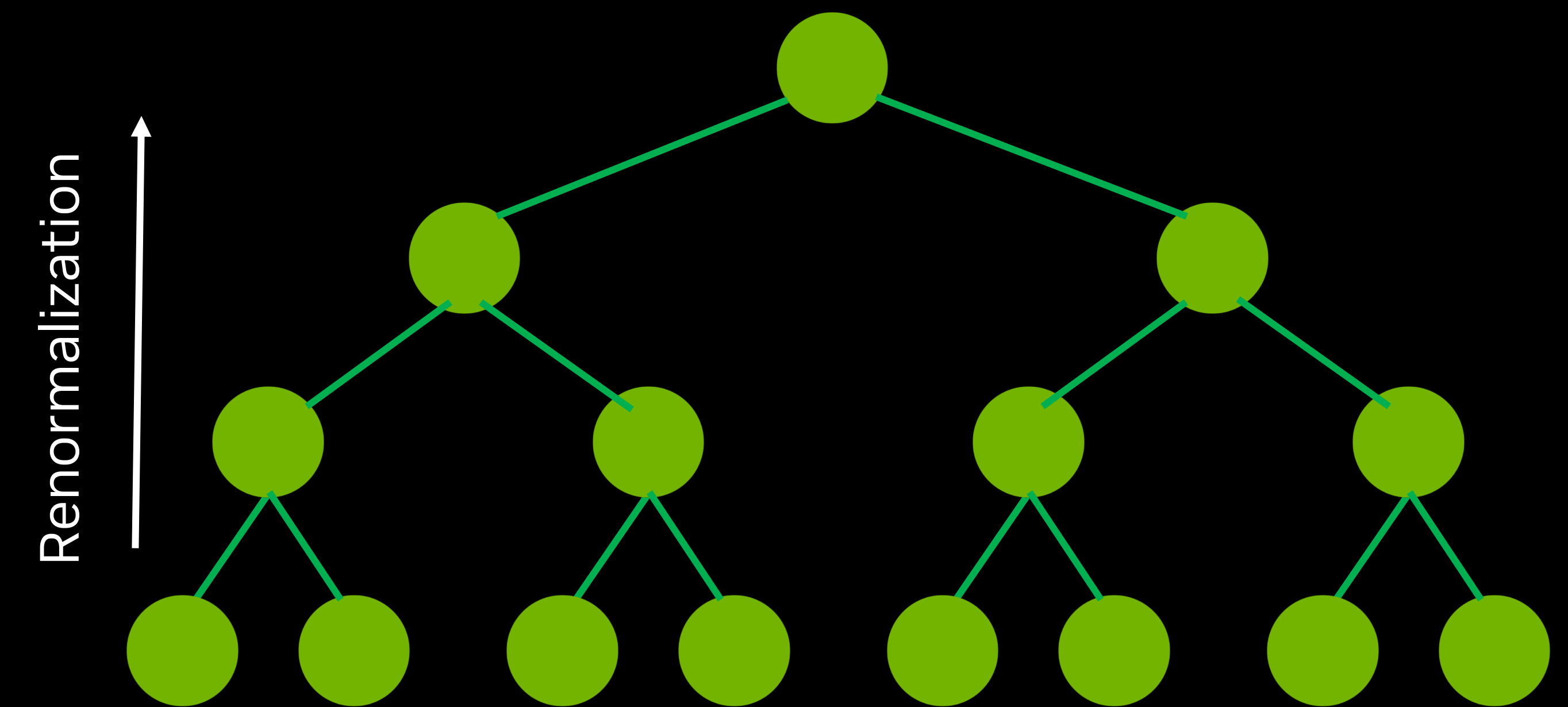
Transverse-field spin Ising Hamiltonian

$$H = -c \sum z_i z_j - g \sum x_i$$

Condensed Matter Physics Simulations

Accelerate Tensor Network Contractions

- We simulated the 16-site **transverse-field Ising Hamiltonian**
- The ground spin state is factorized as a binary **tensor tree** with maximal bond dimension of 16
- The variational optimization of the ground spin state involves two main numerical steps: (1) Tensor network contraction; (2) Modified Gram-Schmidt orthogonalization
- The Modified Gram-Schmidt orthogonalization step must be computed in **FP64**, otherwise the solver may diverge
- Most Flops are spent in **FP32** tensor network contractions --> offload to **BF16/9**
- The ground state energy is computed as a reduction over many terms --> Expect **cancellation** of individual term errors
- The resulting ground state energy is the same up to the 5th digit after decimal point: **FP32** and **BF16/9** produce about the same error (< 1e-5) as compared to full **FP64**



Individual spin sites (physical degrees of freedom)

Transverse-field spin Ising Hamiltonian

$$H = -c \sum z_i z_j - g \sum x_i$$

**Ground state energy of the 16-site Ising Hamiltonian
all converged to the same 2E-6 tolerance**

Precision	Energy	Error
FP64	-17.02418(9)	0
FP32	-17.02419(7)	8e-6
BF16/9	-17.02418(3)	6e-6

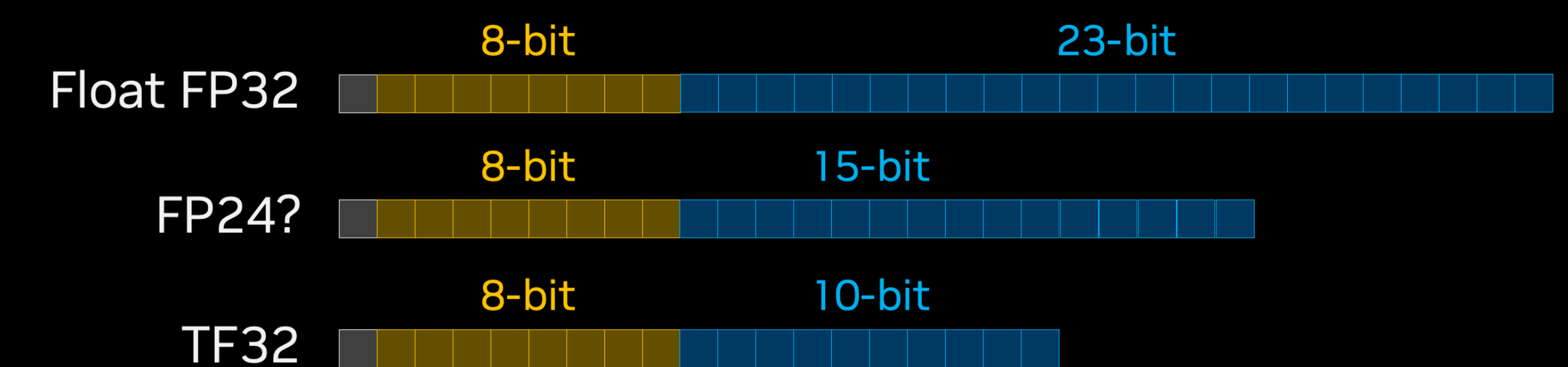
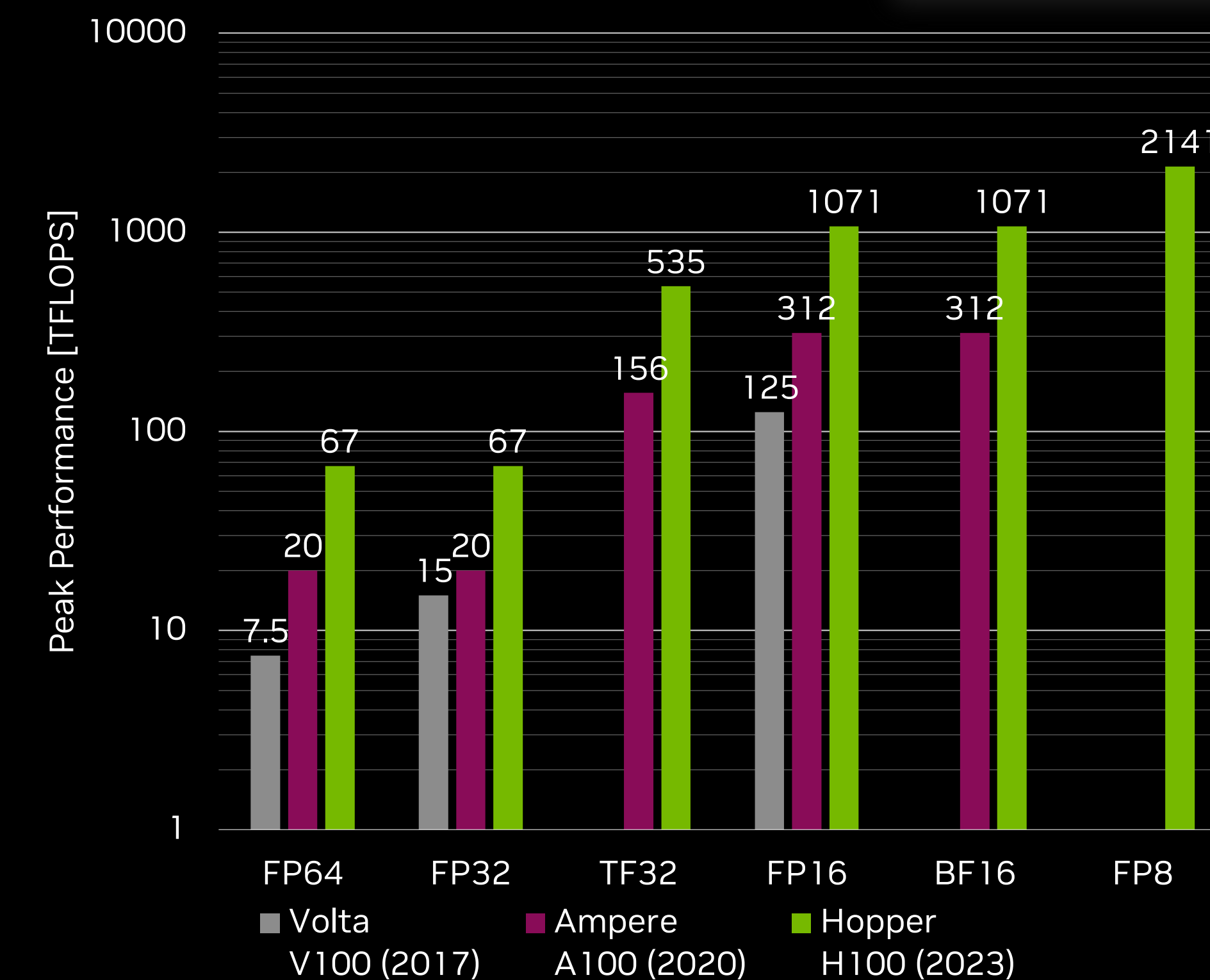
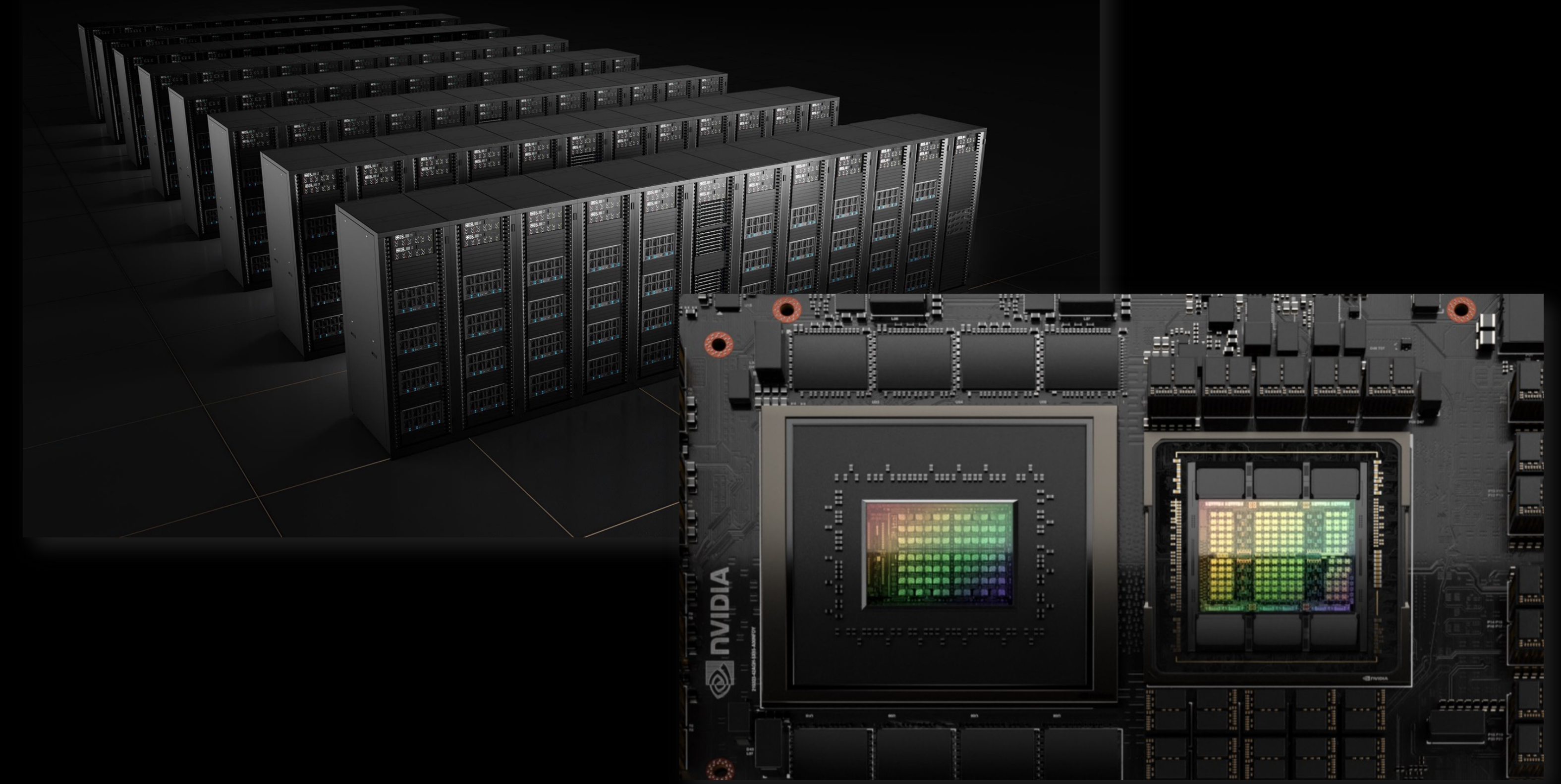
The background features a complex pattern of thin, overlapping lines in shades of green and white against a black background. The lines are arranged in a way that suggests depth and movement, with some lines appearing to curve and others to be straight. The overall effect is a dynamic, almost crystalline or fiber-like structure.

Closing Remarks

Concluding Remarks

Heterogeneous Computing At Multiple Levels

- Heterogeneity for HPC is reality at many different levels
 - Within a server and processor
 - Data storage, locality and access
 - Across the network that connects processor
 - Software stack
 - Within algorithms
- There are many challenges and opportunities for developing high-performance software
 - ... computing is a not a chip problem. It's a software and chip problem
- Tensor Cores that power AI can also be leveraged to transparently accelerate applications that require higher precision without any loss of accuracy
 - TF32/3, BF16/9, and similar algorithms can be extended to create new range and precision modes that can be tailored for applications for further acceleration



Are we ready to reap the benefits
of higher-performance non-IEEE
computations without sacrificing
accuracy?

It's also good for our planet!

