





Diving into NVIDIA Grace Hopper and NVIDIA Grace CPU Superchips: capabilities and performance

Filippo Spiga | March 26th, 2024

The background of the slide is an abstract composition of curved, overlapping green shapes that create a sense of depth and movement. A solid vertical green bar runs down the right side of the image, partially overlapping the text area.

THIS INFORMATION IS INTENDED TO OUTLINE OUR GENERAL PRODUCT DIRECTION. MANY OF THE PRODUCTS AND FEATURES DESCRIBED HEREIN REMAIN IN VARIOUS STAGES AND WILL BE OFFERED ON A WHEN-AND-IF-AVAILABLE BASIS. THIS ROADMAP DOES NOT CONSTITUTE A COMMITMENT, PROMISE, OR LEGAL OBLIGATION AND IS SUBJECT TO CHANGE AT THE SOLE DISCRETION OF NVIDIA. THE DEVELOPMENT, RELEASE, AND TIMING OF ANY FEATURES OR FUNCTIONALITIES DESCRIBED FOR OUR PRODUCTS REMAINS AT THE SOLE DISCRETION OF NVIDIA. NVIDIA WILL HAVE NO LIABILITY FOR FAILURE TO DELIVER OR DELAY IN THE DELIVERY OF ANY OF THE PRODUCTS, FEATURES, OR FUNCTIONS SET FORTH IN THIS DOCUMENT.

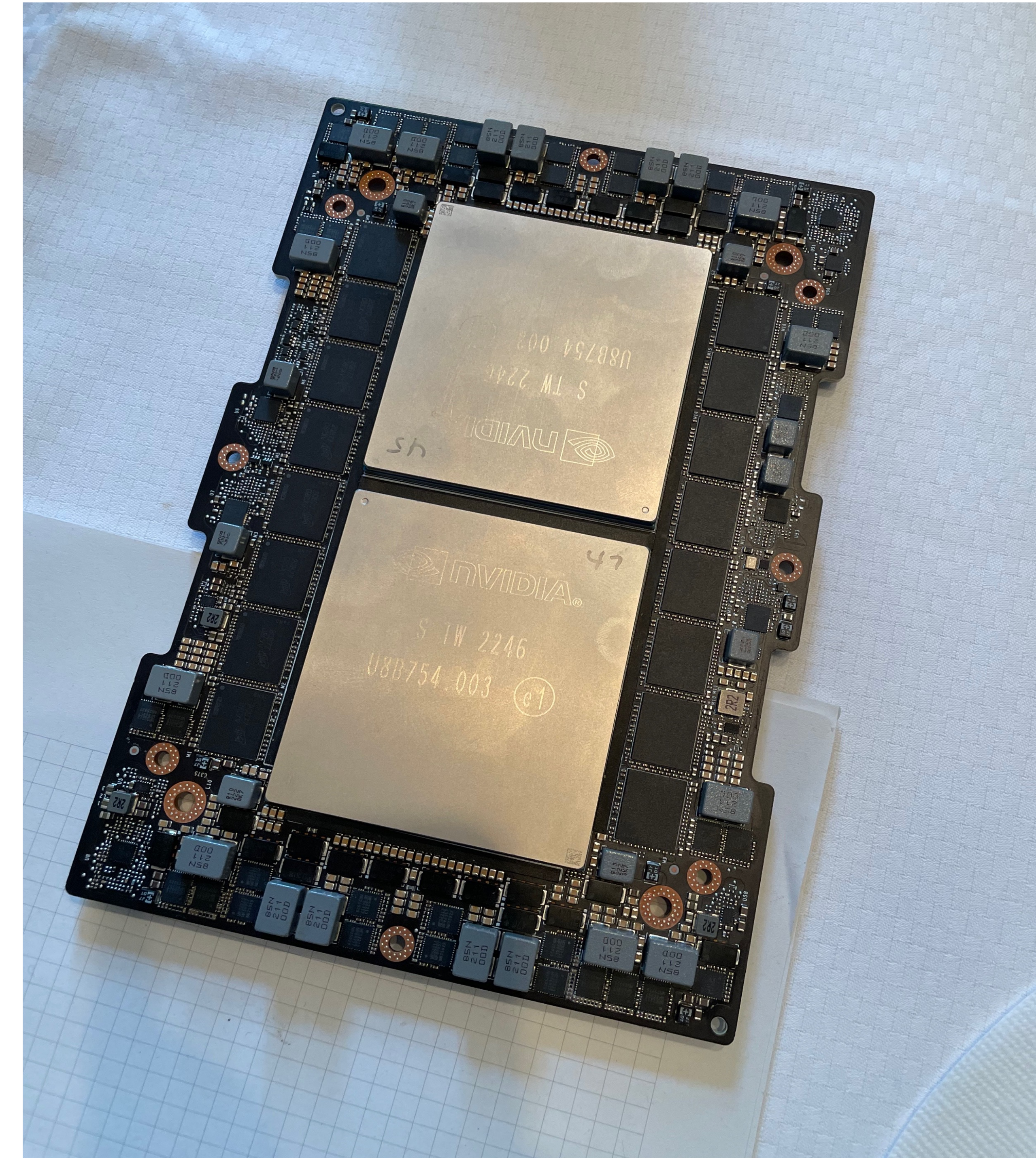
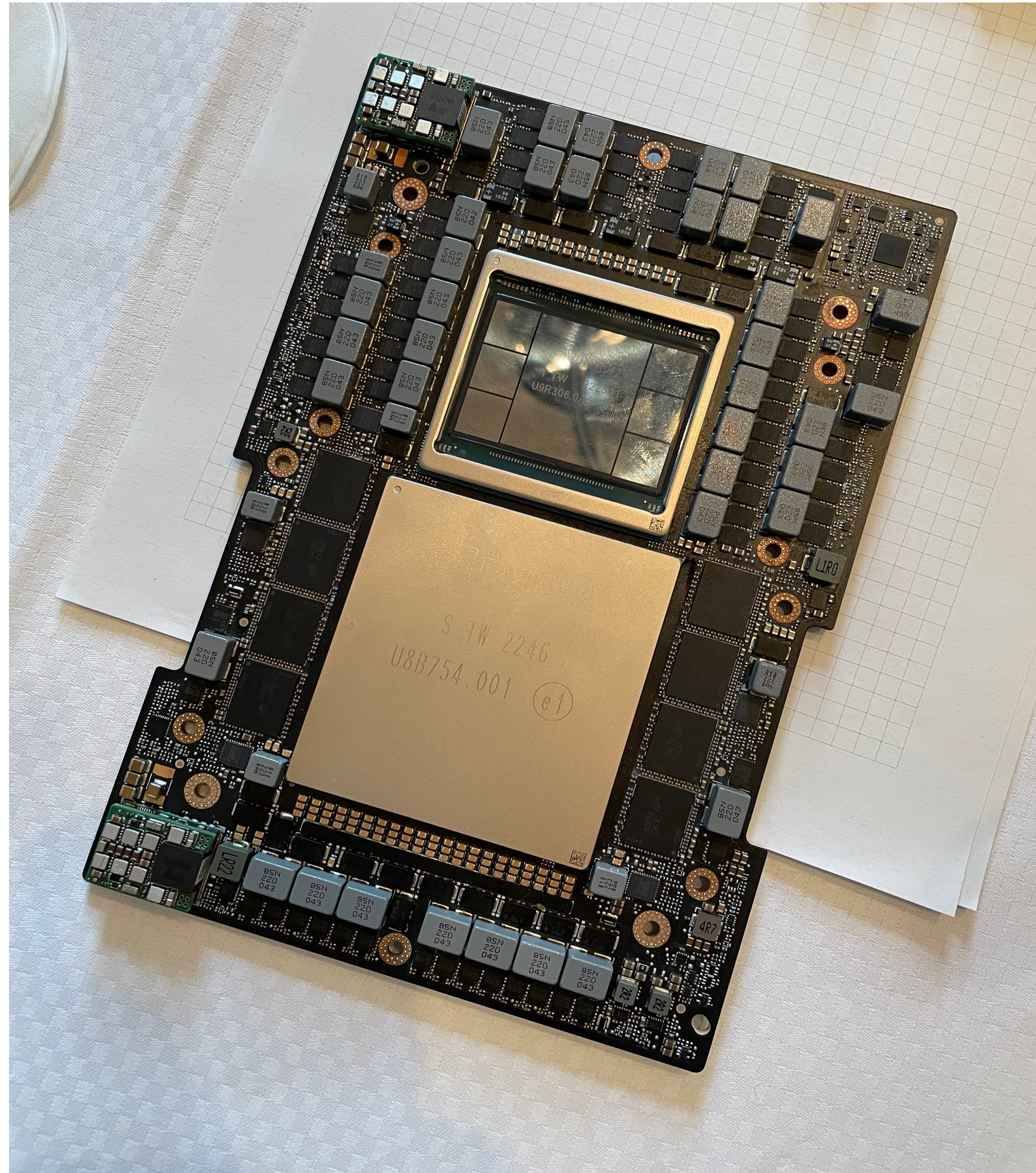
- 
- The background features a series of curved, overlapping bands in various shades of green, ranging from light lime to dark forest green. A solid, medium-green vertical bar runs down the right side of the image, separating the abstract graphic from the text.
- Products
 - Capabilities
 - Software
 - Use-Cases

The background features a series of overlapping, wavy, light green bands that create a sense of depth and movement. On the far left, there is a solid, vertical green bar. The overall aesthetic is clean, modern, and tech-oriented.

Platforms

NVIDIA Superchip Modules

Grace Hopper (GH200) on left || Grace CPU Superchip on right



NVIDIA Grace CPU Superchip

2X Performance at the Same Power for the Modern Data Center

High Performance Power Efficient Cores

144 flagship Arm Neoverse V2 Cores with
SVE2 4x128b SIMD per core

Fast On-Chip Fabric

3.2 TB/s of bi-section bandwidth connects
CPU cores, NVLink-C2C, memory, and system IO

High-Bandwidth Low-Power Memory

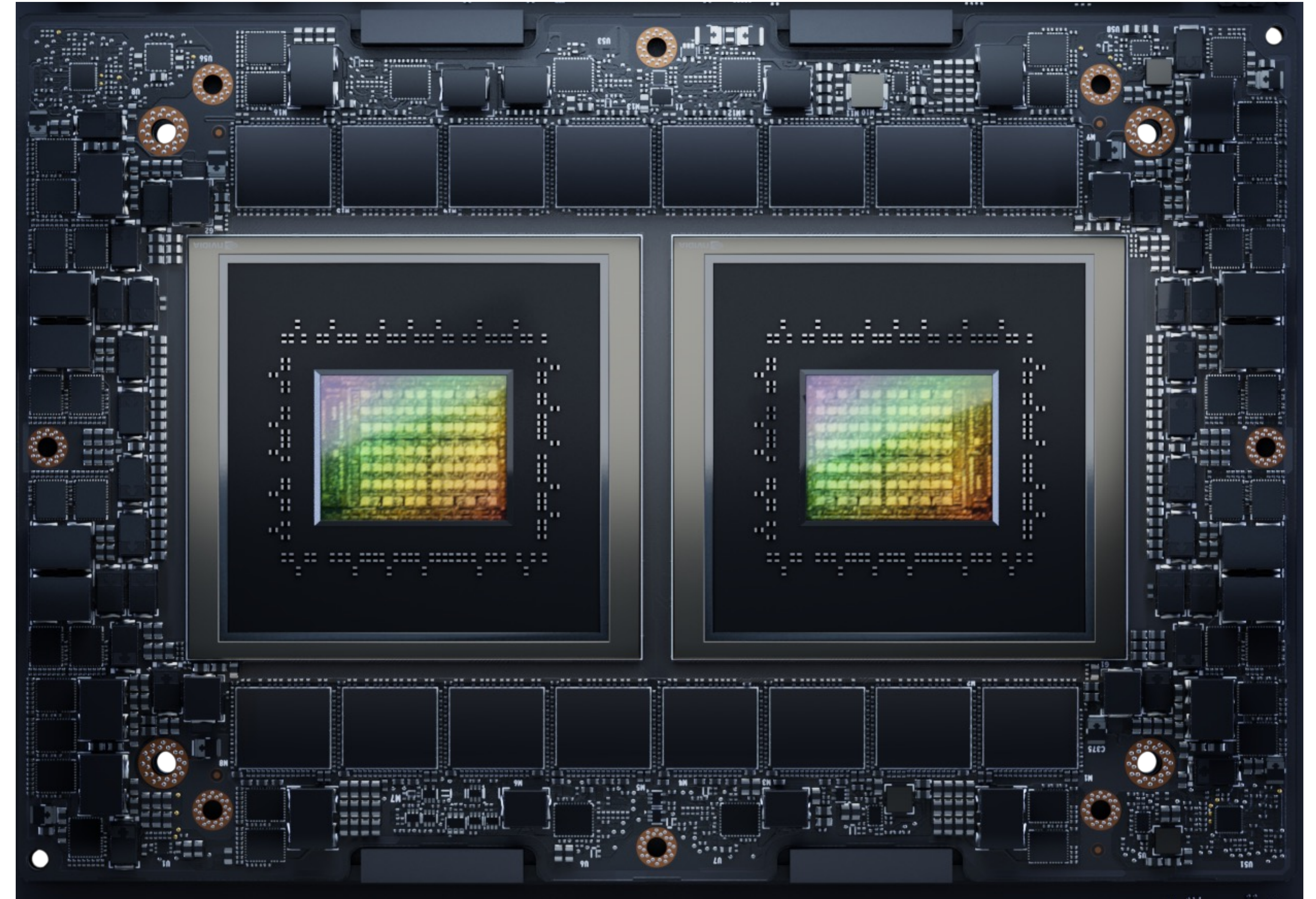
Up to 960GB of data center enhanced LPDDR5X Memory that
delivers up to 1TB/s of memory bandwidth

Fast and Flexible CPU IO

Up to 8x PCIe Gen5 x16 interface. PCIe Gen 5 up to 128GB/s
2X more bandwidth compared to PCIe Gen 4

Full NVIDIA Software Stack

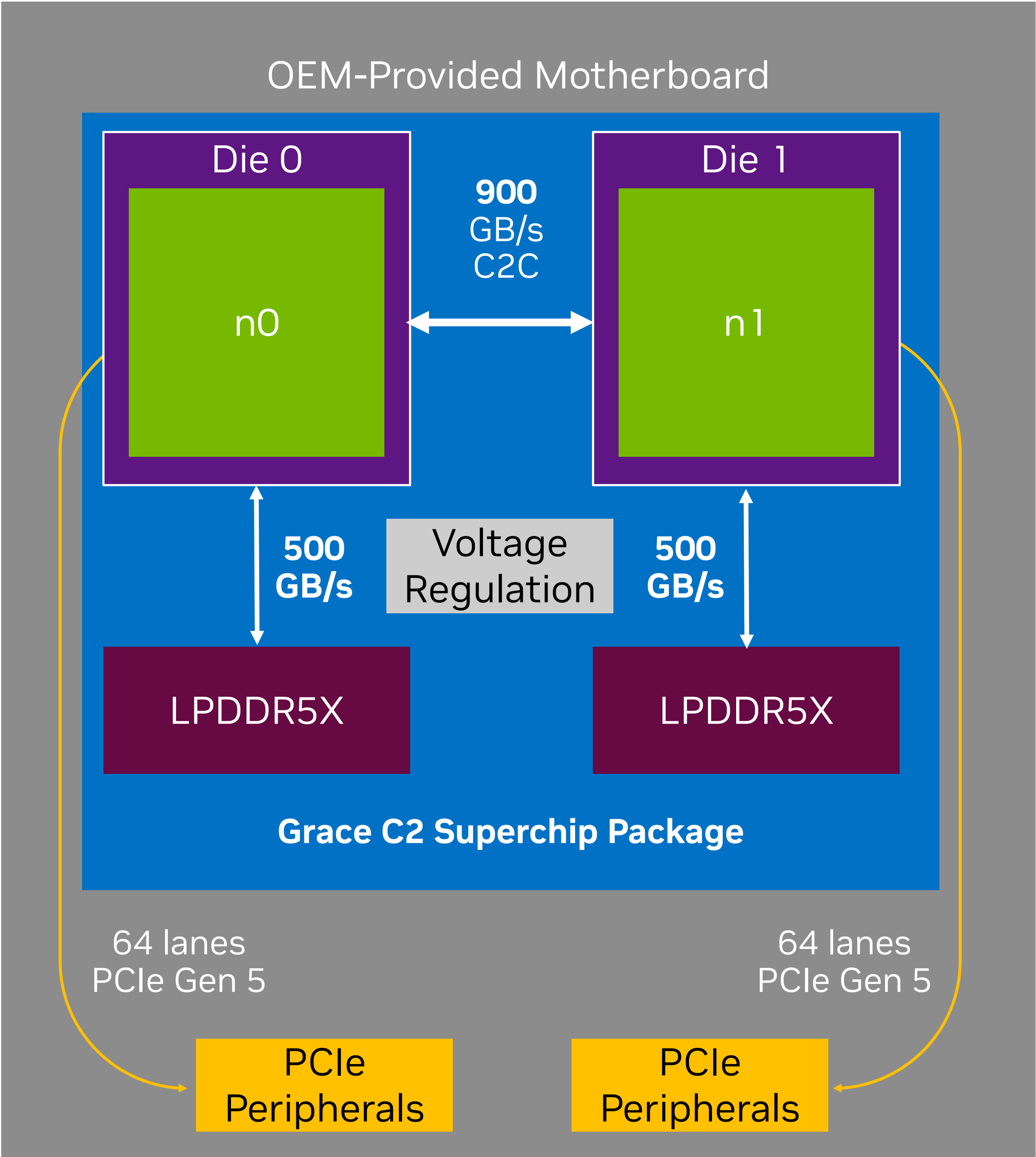
AI, Omniverse



Grace Simplifies System Design and Workload Optimization

Reduces NUMA & sub-NUMA Bottlenecks

Grace Server
Grace C2 Superchip



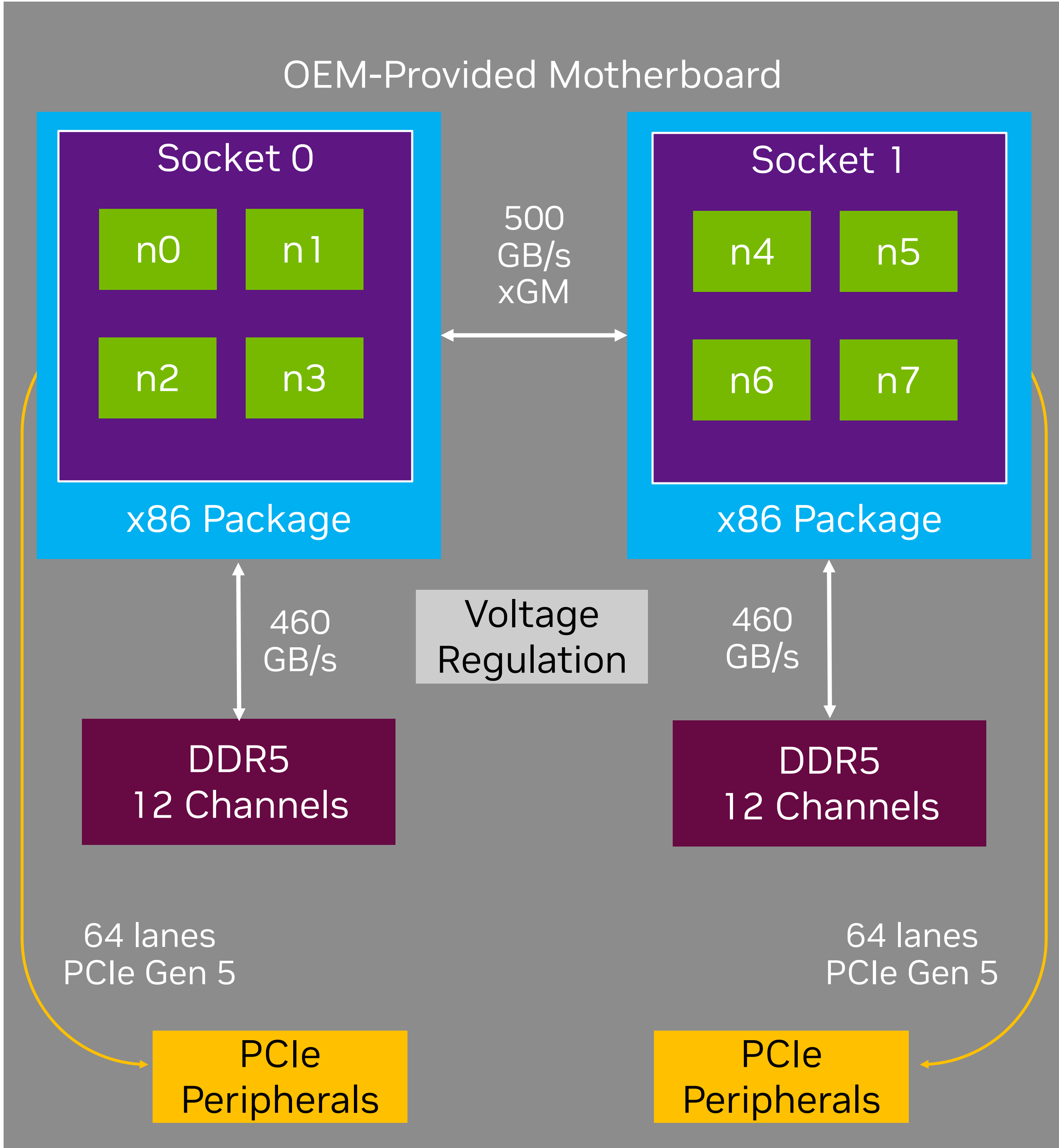
2 NUMA Nodes

2 Compute Dies

500 Watts (CPU + MEM)

900 GB/s worst-case n to n

Conventional 2-Socket Server
Example: 2x AMD Genoa, Native NPS=4



8 NUMA Nodes

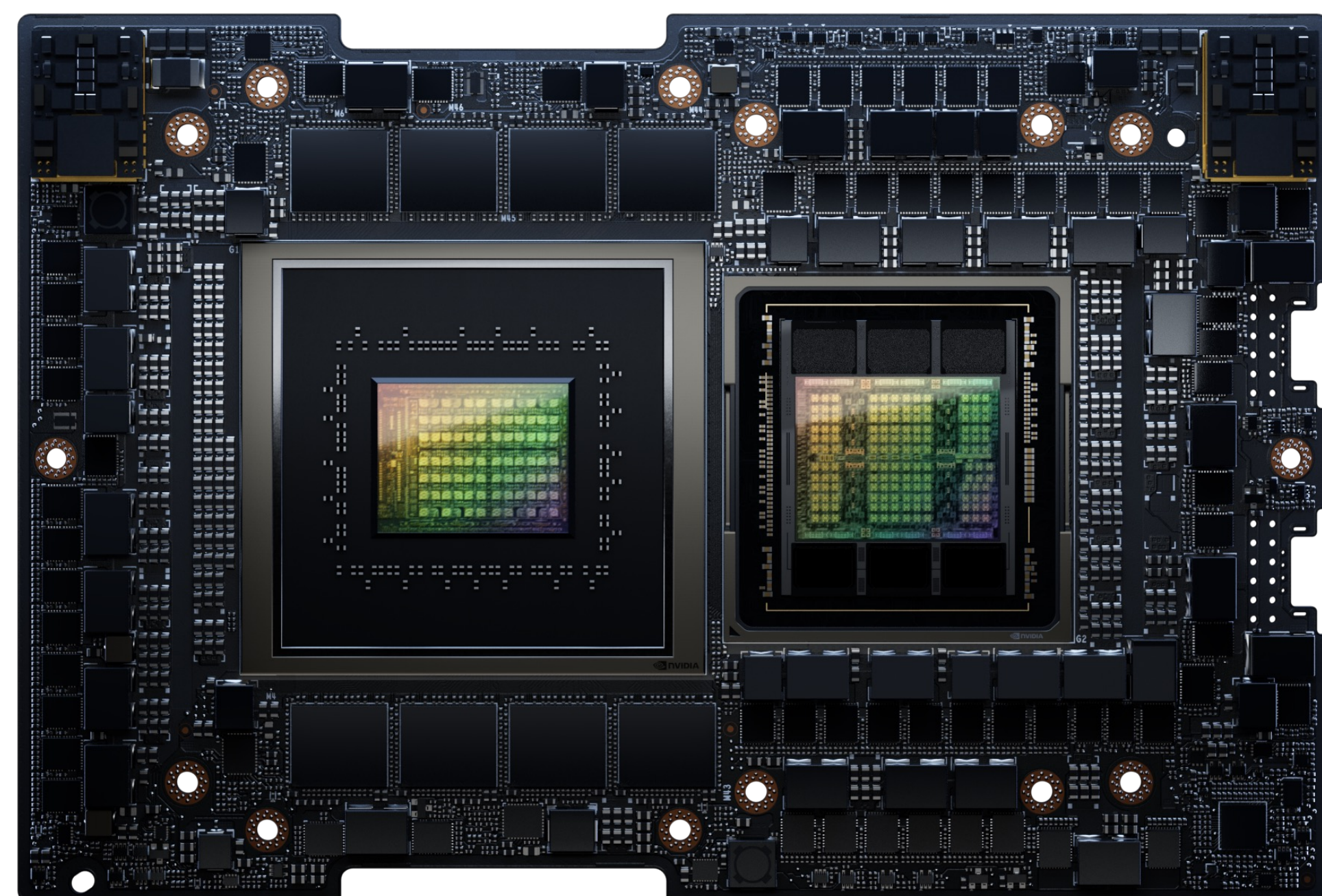
24 Compute Chiplets

900+ Watts (CPU + MEM)

500 GB/s worst-case n to n

NVIDIA GH200 Grace Hopper Superchip

Processor For The Era of Accelerated Computing And Generative AI

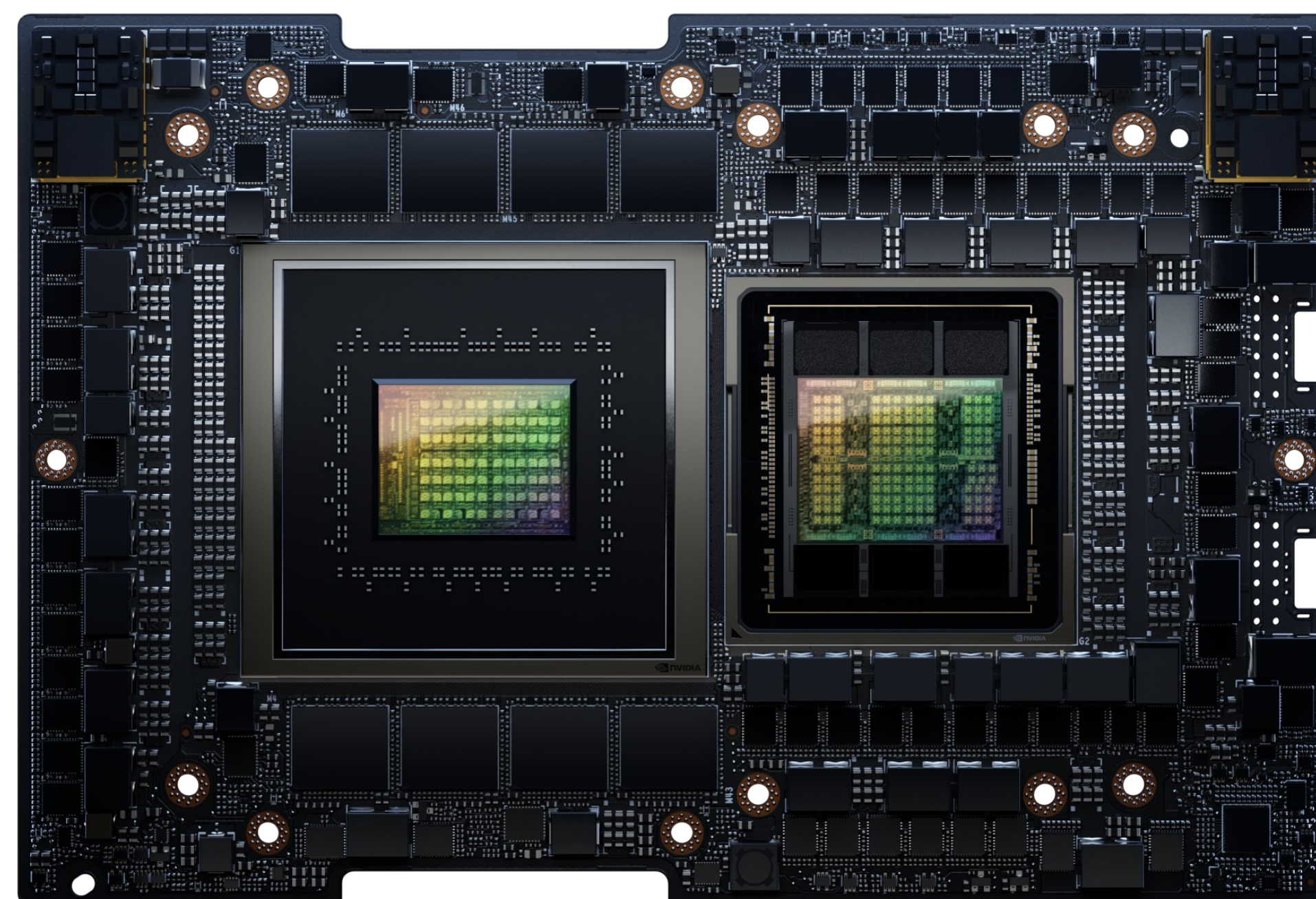


72 Core Grace CPU | 4 PFLOPS Hopper GPU
96 GB HBM3 | 4 TB/s | 900 GB/s NVLink-C2C

- 7X bandwidth to GPU vs PCIe Gen 5
- Combined 576 GB of fast memory
- 1.2x capacity and bandwidth vs H100
- Full NVIDIA Compute Stack

GH200 with HBM3

Available for order

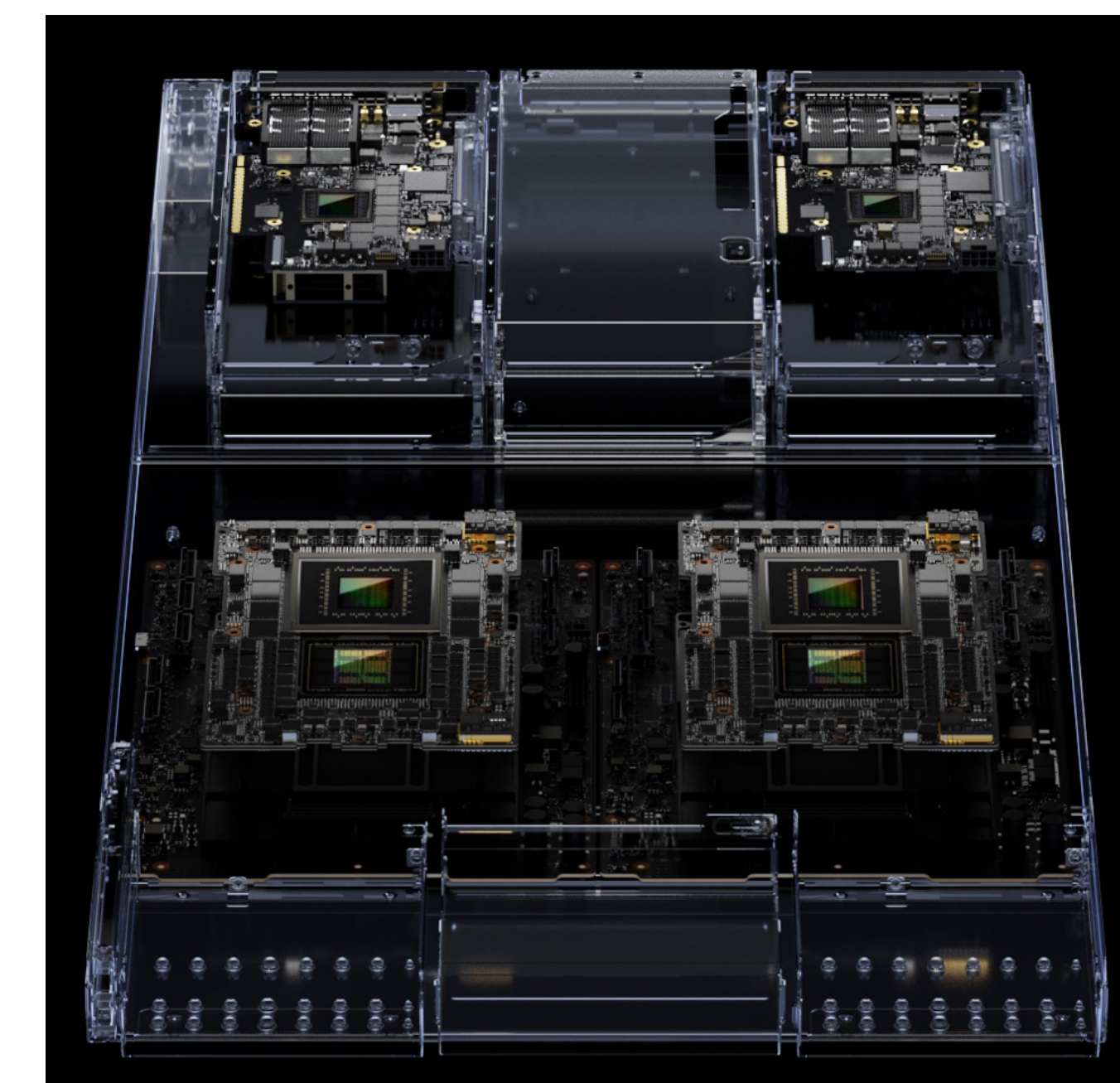


72 Core Grace CPU | 4 PFLOPS Hopper GPU
144 GB HBM3e | 5 TB/s | 900 GB/s NVLink-C2C

- World's first HBM3e GPU
- Combined 624 GB of fast memory
- 1.7x capacity and 1.5x bandwidth vs H100
- Full NVIDIA Compute Stack

GH200 with HBM3e

Available late Q2 2024



144 Core Grace CPU | 8 PFLOPS Hopper GPU
288 GB HBM3e | 10 TB/s | 900 GB/s NVLink-C2C

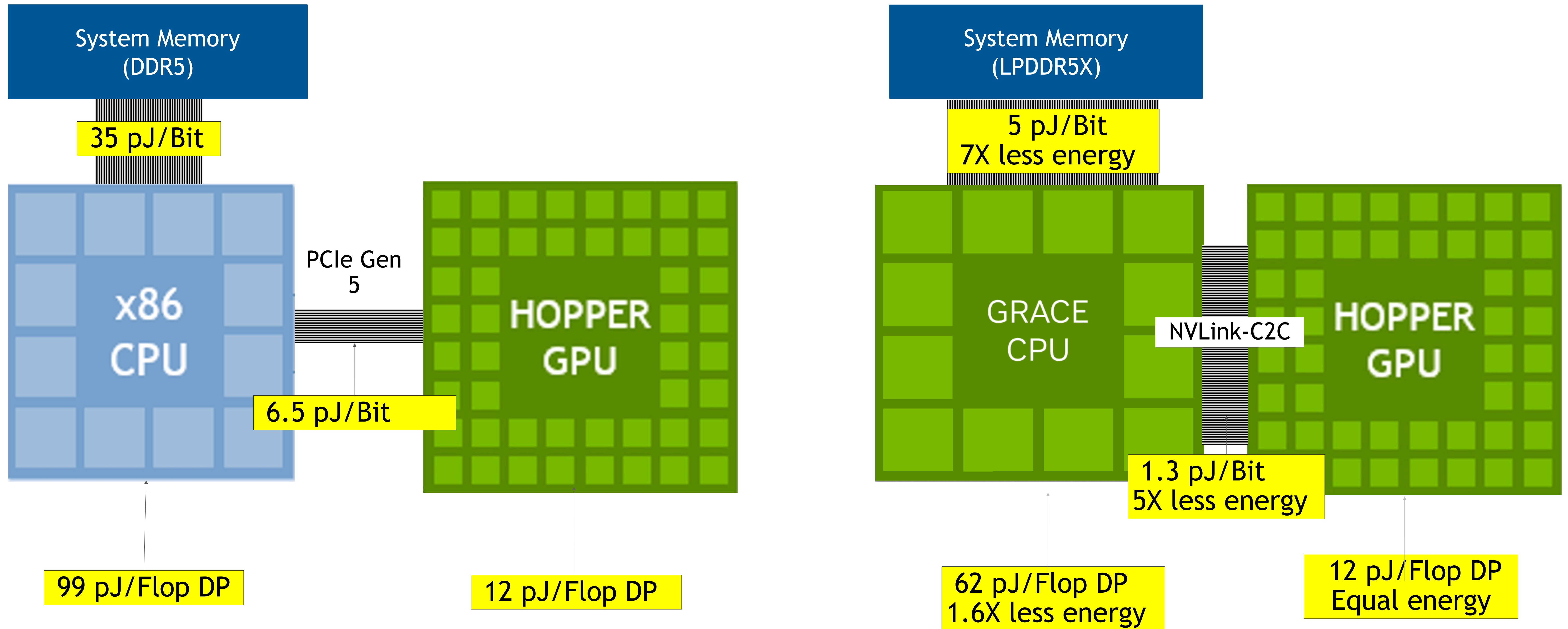
- Simple to deploy MGX-compatible design
- Combined 1.2 TB fast memory
- 3.5x capacity and 3x bandwidth vs H100
- Full NVIDIA Compute Stack

NVLink Dual GH200 System

Available late Q2 2024

Energy Efficient Design

More Efficient Computation and Data Movement



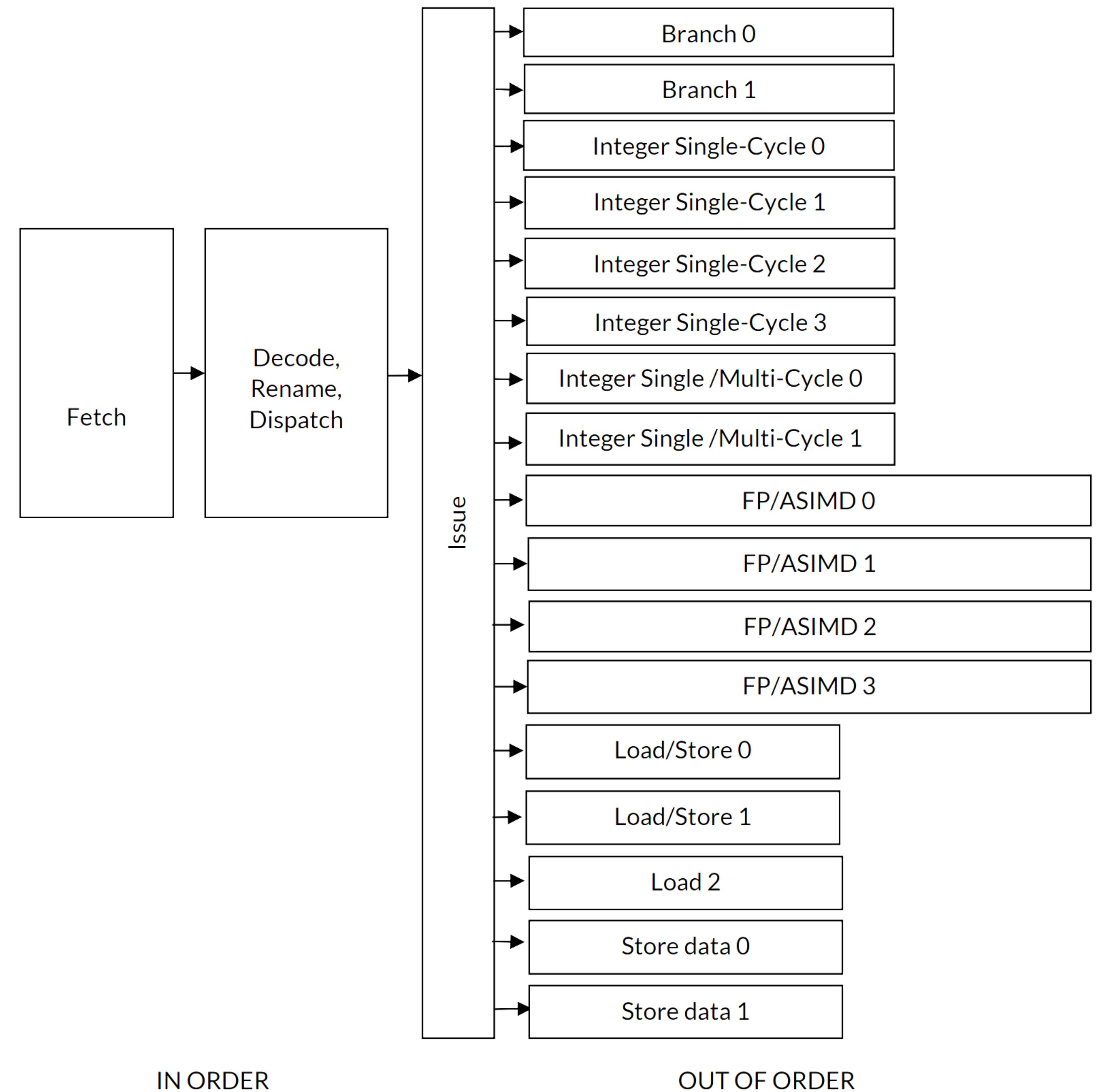
The background features a series of parallel, wavy lines in various shades of green, creating a sense of depth and movement. A solid green vertical bar is positioned on the far left side of the image.

Capabilities

NVIDIA Grace SIMD Highlights

Neoverse V2 Arm IP core

- 4x128b SIMD units = 512b SIMD vector bandwidth total
- Each SIMD unit can retire NEON or SVE2 instructions
- On this architecture, **SVE2 and NEON have the same peak performance**
- SVE2 can vectorize more complex codes and supports more data types than NEON.
- NEON doesn't require predicate calculation
 - *Neither does VLS SVE*

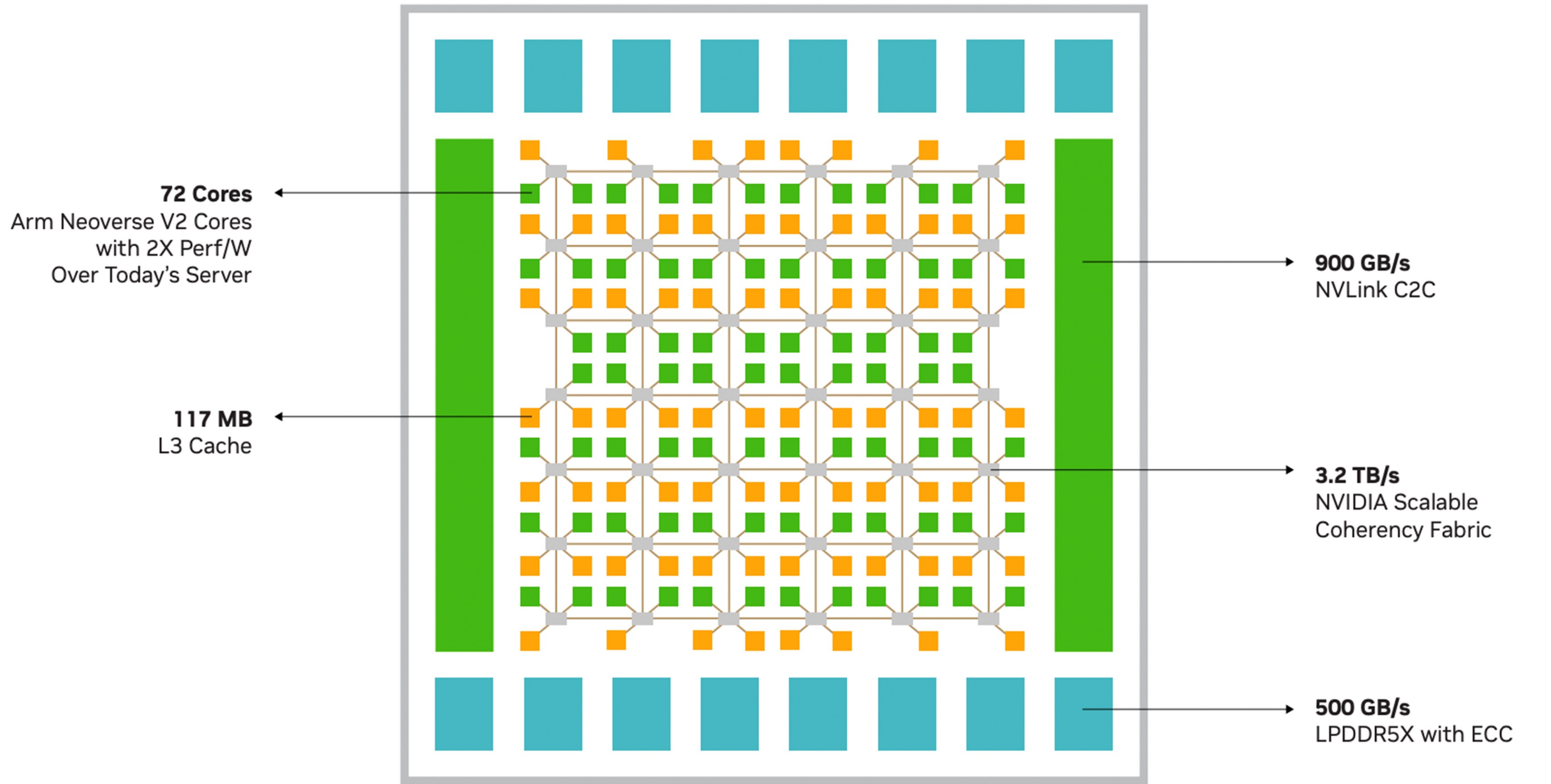


For more in-depth core u-arch details: [Arm® Neoverse™ V2 Core Technical Reference Manual](#)

Grace SoC Memory Subsystem

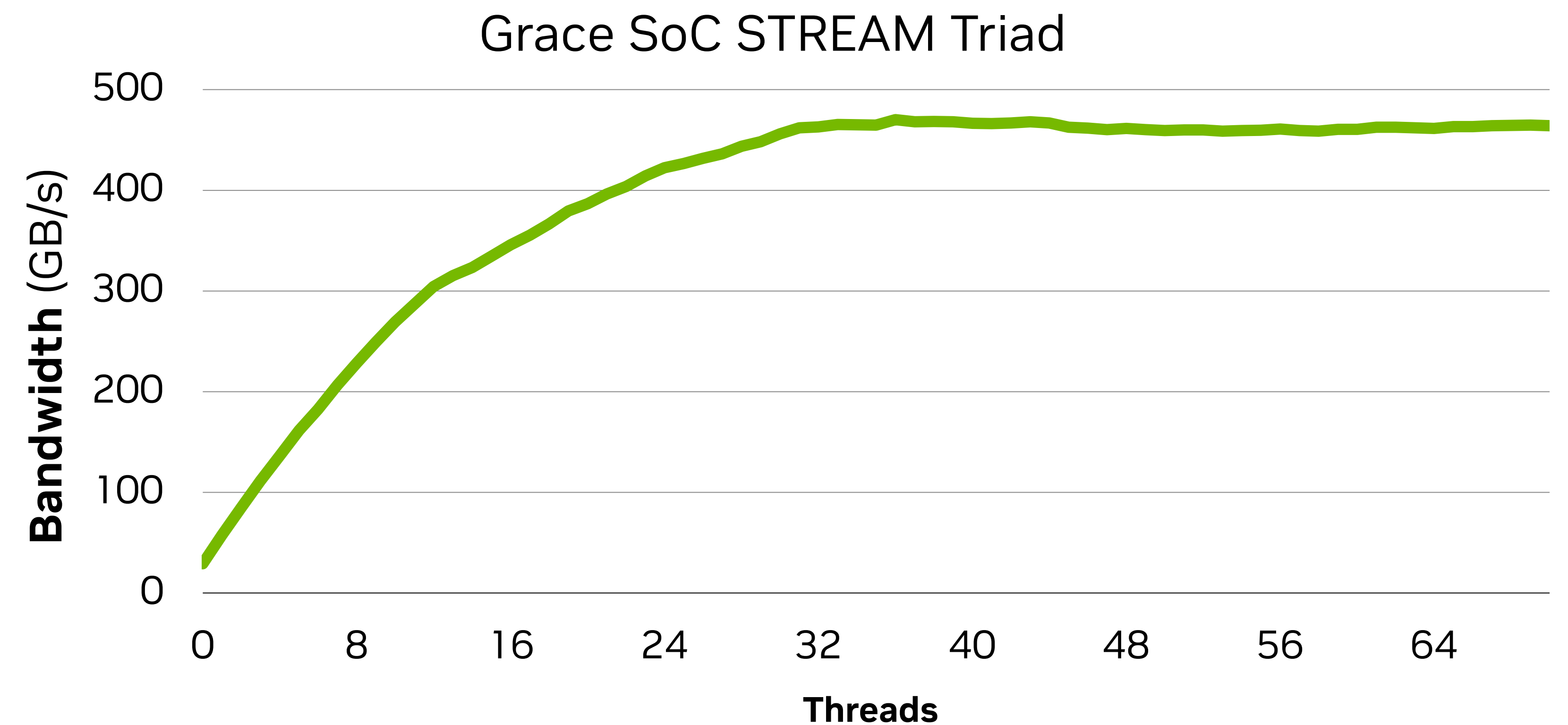
Compute and Data-mover Architecture

- Separate L1 data and instruction caches per core
 - L1 64KB, 4-way set associative, 64B cache line
- Private, unified data and instruction L2 cache per core
 - 1MB , 8-way set associative
- Scalable Coherency Fabric
 - Shared, uniform 117MB of L3 cache for entire chip.



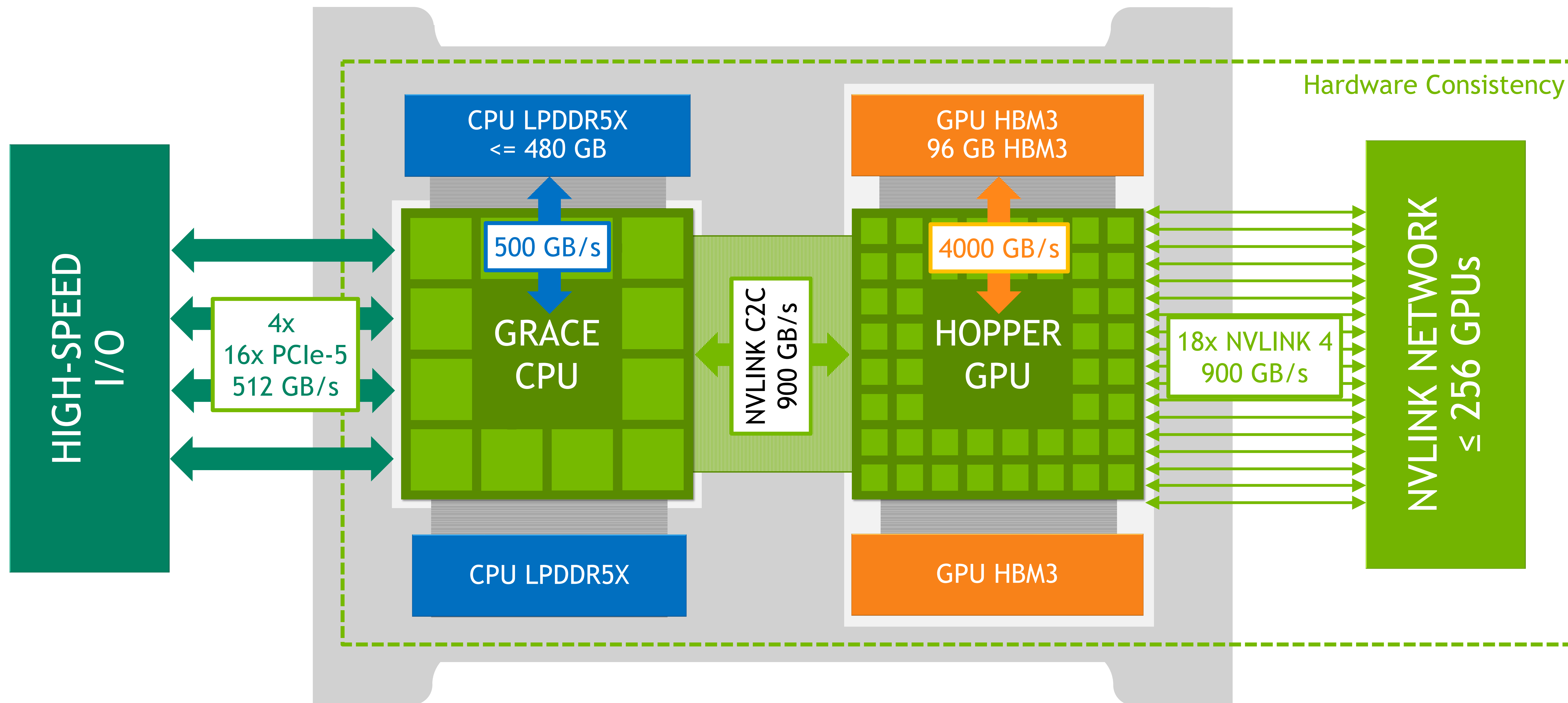
Superchip	Capacity (GB)	OMP_NUM_THREADS	Expected TRIAD Bandwidth
Grace-Hopper	120	72	450+
Grace-Hopper	480	72	340+
Grace CPU	240	144	900+
Grace CPU	480	144	900+
Grace CPU	960	144	680+

Source: <https://docs.nvidia.com/grace-performance-tuning-guide.pdf>



Grace Hopper Superchip

GPU can access CPU memory at CPU memory speeds



GPU Memory is Visible to the Operating System

Standard operating system commands work on the GPU

```
nvidia@localhost:~$ numactl -H
available: 9 nodes (0-8)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 5
2 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71
node 0 size: 490310 MB
node 0 free: 475425 MB
node 1 cpus:
node 1 size: 96768 MB
node 1 free: 96767 MB
node 2 cpus:
node 2 size: 0 MB
node 2 free: 0 MB
node 3 cpus:
node 3 size: 0 MB
node 3 free: 0 MB
node 4 cpus:
node 4 size: 0 MB
node 4 free: 0 MB
node 5 cpus:
node 5 size: 0 MB
node 5 free: 0 MB
node 6 cpus:
node 6 size: 0 MB
node 6 free: 0 MB
node 7 cpus:
node 7 size: 0 MB
node 7 free: 0 MB
node 8 cpus:
node 8 size: 0 MB
node 8 free: 0 MB
node distances:
node  0  1  2  3  4  5  6  7  8
  0: 10  80  80  80  80  80  80  80  80
  1: 80  10 255 255 255 255 255 255 255
  2: 80 255 10 255 255 255 255 255 255
  3: 80 255 255 10 255 255 255 255 255
  4: 80 255 255 255 10 255 255 255 255
  5: 80 255 255 255 255 10 255 255 255
  6: 80 255 255 255 255 255 10 255 255
  7: 80 255 255 255 255 255 255 10 255
  8: 80 255 255 255 255 255 255 255 10
nvidia@localhost:~$ free -g
              total        used        free      shared  buff/cache   available
Mem:           573           11          558           1           3          541
Swap:            0             0             0
```

CPU

GPU

MIG

```
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 5
2 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71
node 0 size: 490310 MB
node 0 free: 475425 MB
node 1 cpus:
node 1 size: 96768 MB
node 1 free: 96767 MB
```

Hopper GPU appears to the OS as a NUMA node with no CPU cores

```
nvidia@localhost:~$ free -g
              total        used        free      shared  buff/cache   available
Mem:           573           11          558           1           3          541
Swap:            0             0             0
```

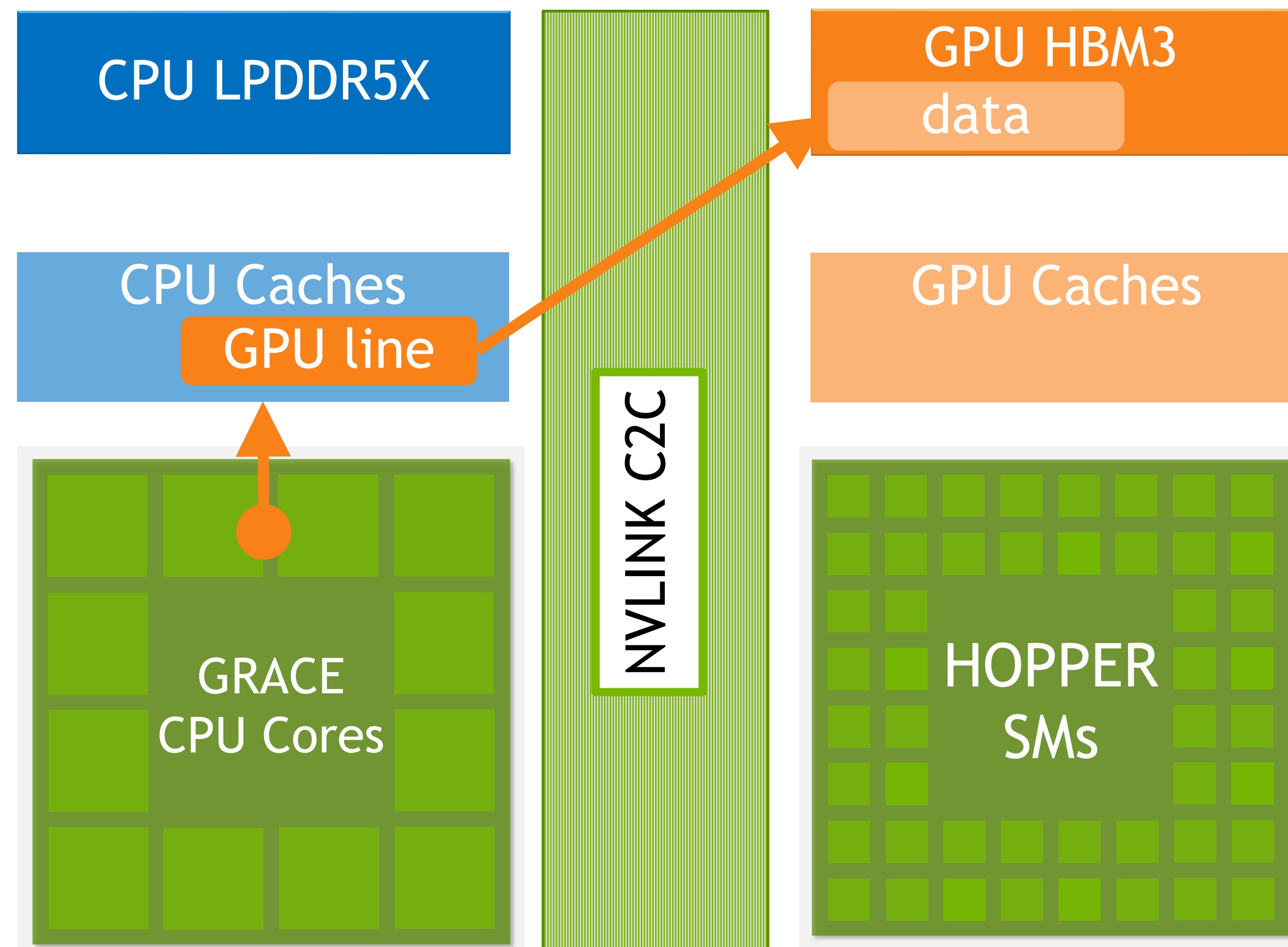
Total system memory capacity is CPU (480GB) + GPU (96GB)

```
nvidia@localhost:/home/nvidia/jlinford/mt-dgemm/src$ numactl -m1 ./mt-dgemm.nvpl 5000 1 1 1 0 1 1
Matrix size input by command line: 5000
Repeat multiply 1 times.
```

Can use numactl to put CPU application data in GPU memory

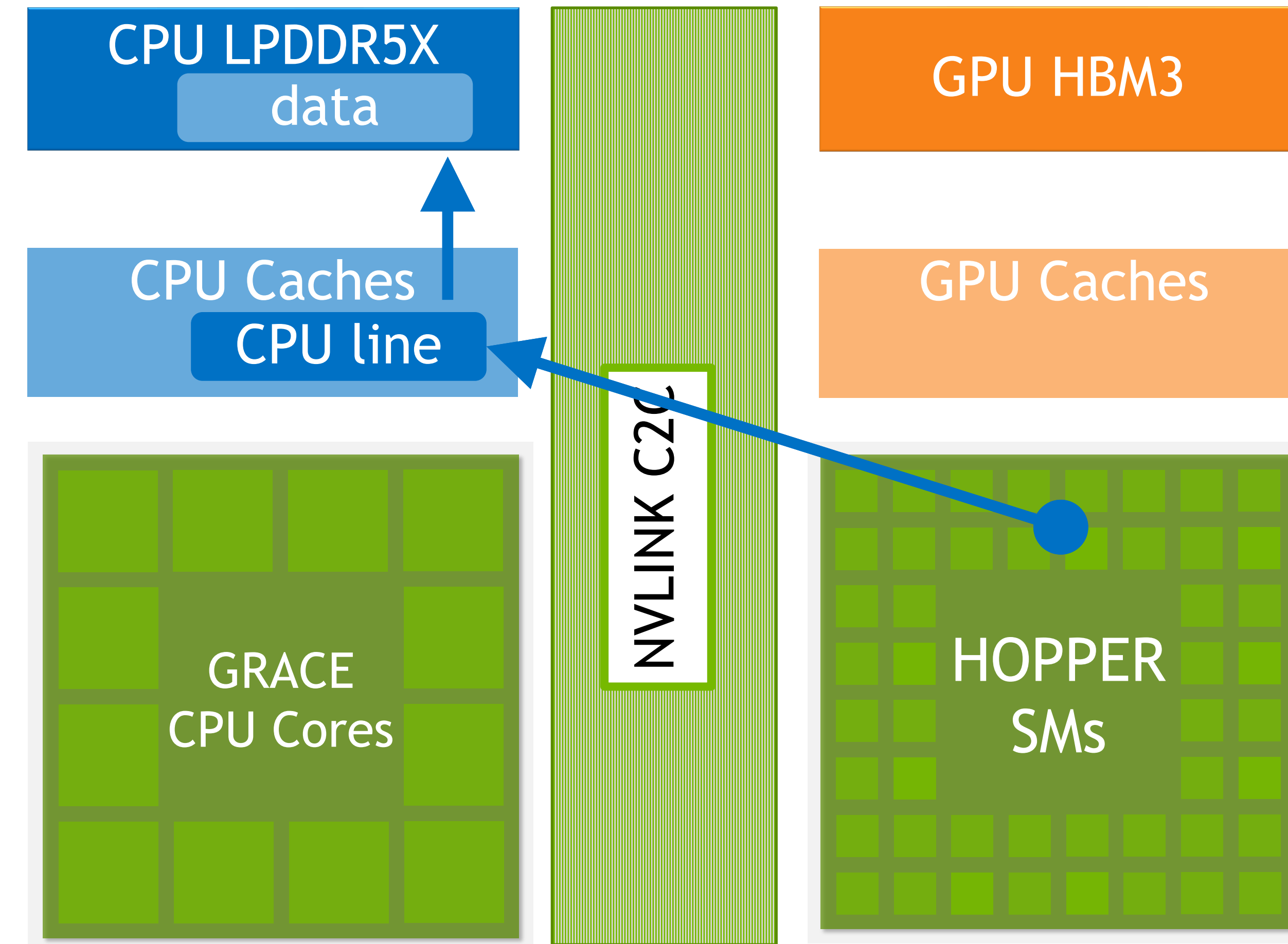
Global Access to All Data

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

The background features a series of overlapping, wavy, light green bands that create a sense of depth and movement. On the far left, there is a solid, vertical green bar. The overall aesthetic is clean, modern, and tech-oriented.

Software

Compilers

- Use a compiler that supports Neoverse V2
- Check and update your compiler flags
- Use **-mcpu=native**
 - Can also use **-mcpu=neoverse-v2**, but **-mcpu=native** will “port forward”
- If possible use **-Ofast**
 - If fast math optimizations are not acceptable, use **-O3 -ffp-contract=fast**
 - For even more accuracy, use **-ffp-contract=off** to disable floating point operation contraction (e.g. FMA)
- Use **-flto** to enable link-time optimization
 - The benefits of link-time optimization vary from code to code, but can be significant
 - See e.g. <https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html> for details
- Apps may need **-fsigned-char** or **-funsigned-char** depending on the developer’s assumption
- Fortran may benefit from **-fno-stack-arrays**
- Remember to check your dependencies
 - This is also a good opportunity to check for newer version with improved Arm Neoverse V2 / Grace support

Compiler	Version \geq
GCC	12.2
LLVM (Clang)	16
NVIDIA HPC	23.3
Arm Compiler	23.04

Porting Applications that use Math Libraries (MKL, OpenBLAS, etc.)

Several library options to choose from

- Prefer Netlib BLAS/LAPACK and FFTW interfaces
 - Building on these interfaces enables compatibility

- **NVPL**

- gcc **-DUSE_CBLAS** -ffast-math -mcpu=native -O3 \
-I/PATH/T0/nvpl/include \
-L/PATH/T0/nvpl/lib \
-o mt-dgemm.nvpl mt-dgemm.c \
-lnvpl_blas_lp64_gomp

- **ArmPL**

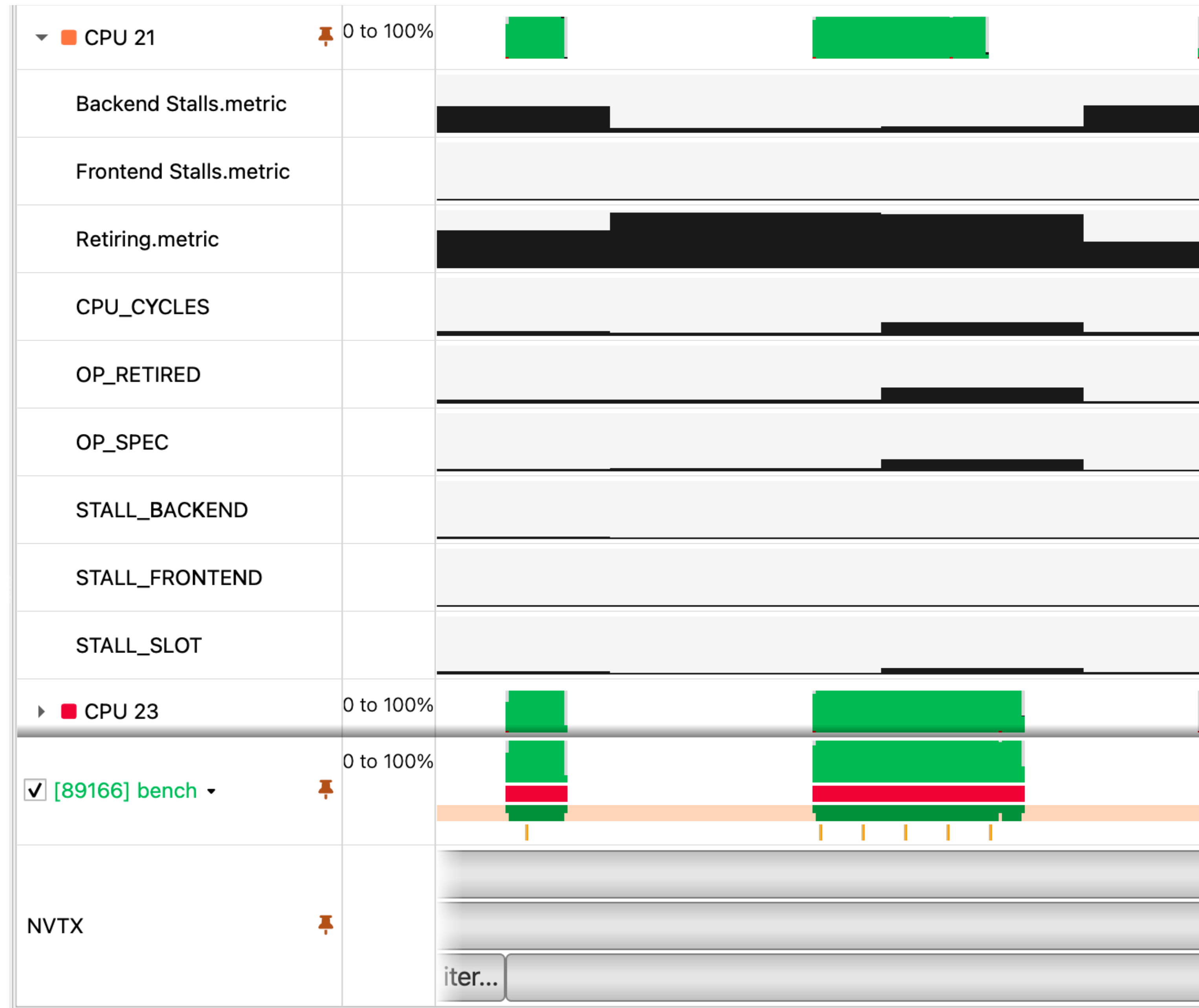
- gcc -DUSE_CBLAS -ffast-math -mcpu=native -O3 \
-I/opt/arm/armpl-23.10.0_Ubuntu-22.04_gcc/include \
-L/opt/arm/armpl-23.10.0_Ubuntu-22.04_gcc/lib \
-o mt-dgemm.armpl mt-dgemm.c \
-larmpl_lp64

```
libnvpl_blas_ilp64_gomp.so  
libnvpl_blas_ilp64_seq.so  
libnvpl_blas_lp64_gomp.so  
libnvpl_blas_lp64_seq.so  
libnvpl_fftw.so  
libnvpl_lapack_ilp64_gomp.so  
libnvpl_lapack_ilp64_seq.so  
libnvpl_lapack_lp64_gomp.so  
libnvpl_lapack_lp64_seq.so  
libnvpl_rand_mt.so  
libnvpl_rand.so  
libnvpl_scalapack_ilp64.so  
libnvpl_scalapack_lp64.so  
libnvpl_sparse.so  
libnvpl_tensor.so
```

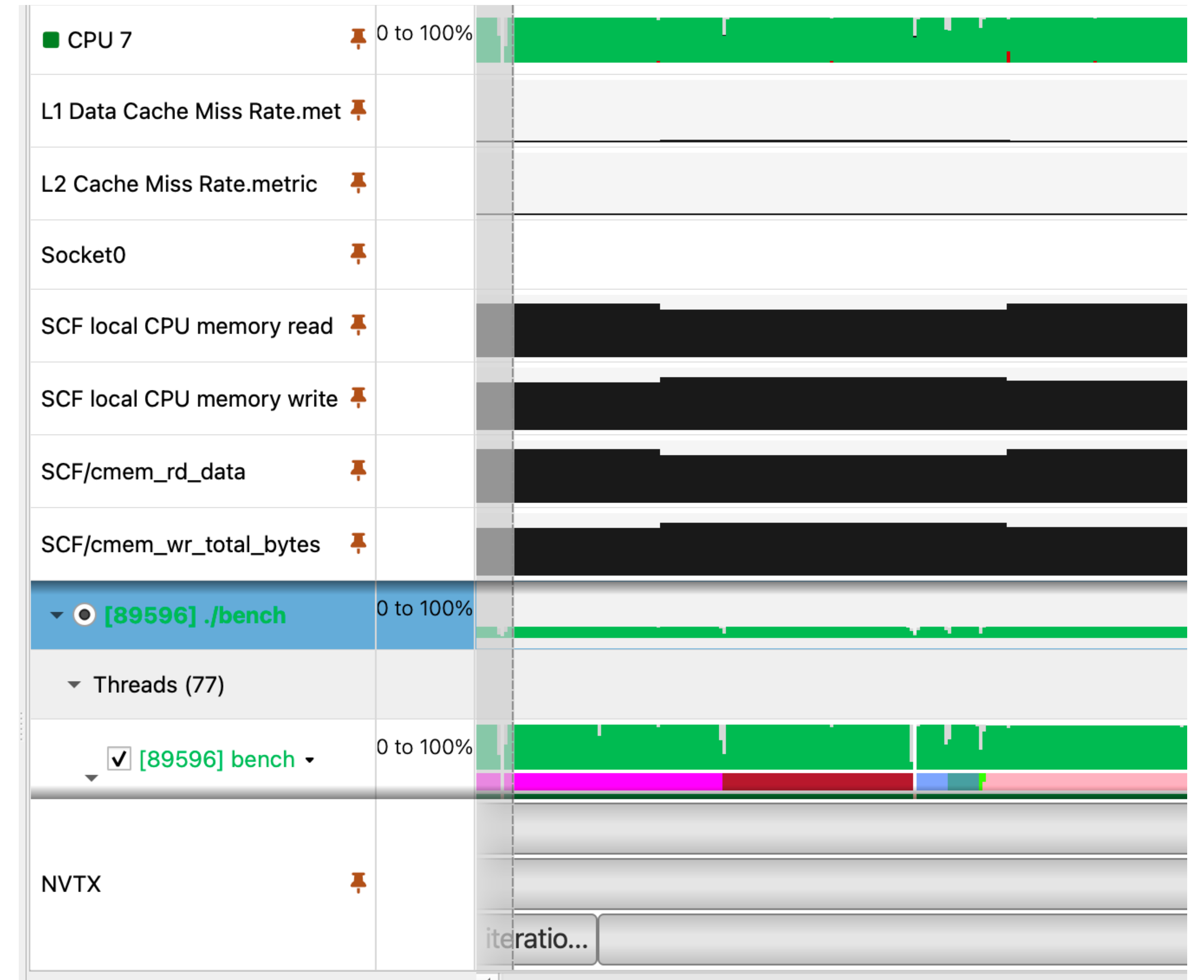
- **ATLAS**, **OpenBLAS**, **BLIS**, ... Community supported with some optimizations for Neoverse V2.
 - Works on Grace, but unlikely to outperform NVPL and ArmPL. A good compatibility option.

Nsight Systems – Core/Uncore metrics

Collect core/uncore performance metrics



`nsys profile --cpu-core-metrics=help`



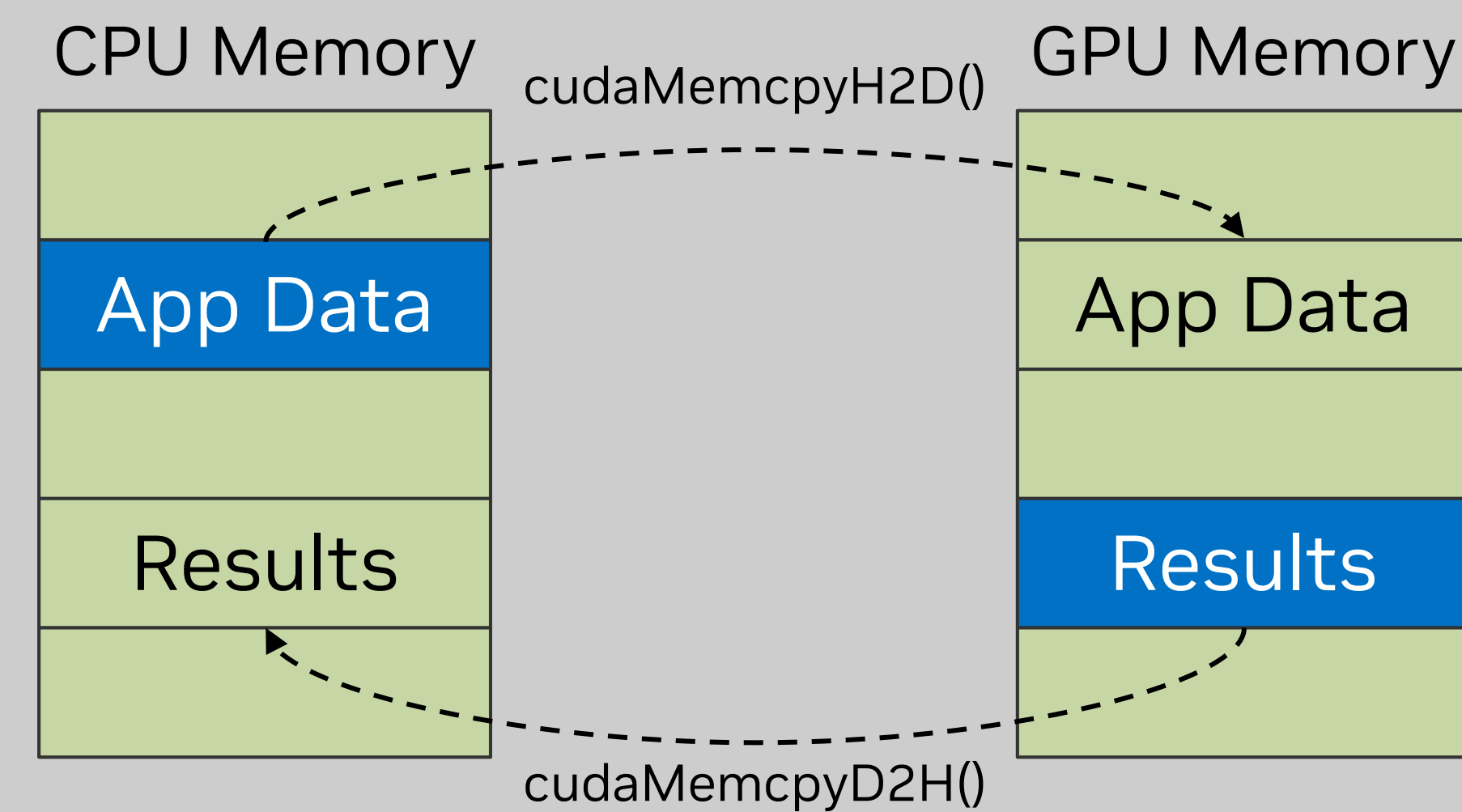
`nsys profile --cpu-socket-metrics=help`

The Grace Hopper Advantage

Full CUDA support with additional Grace memory extensions

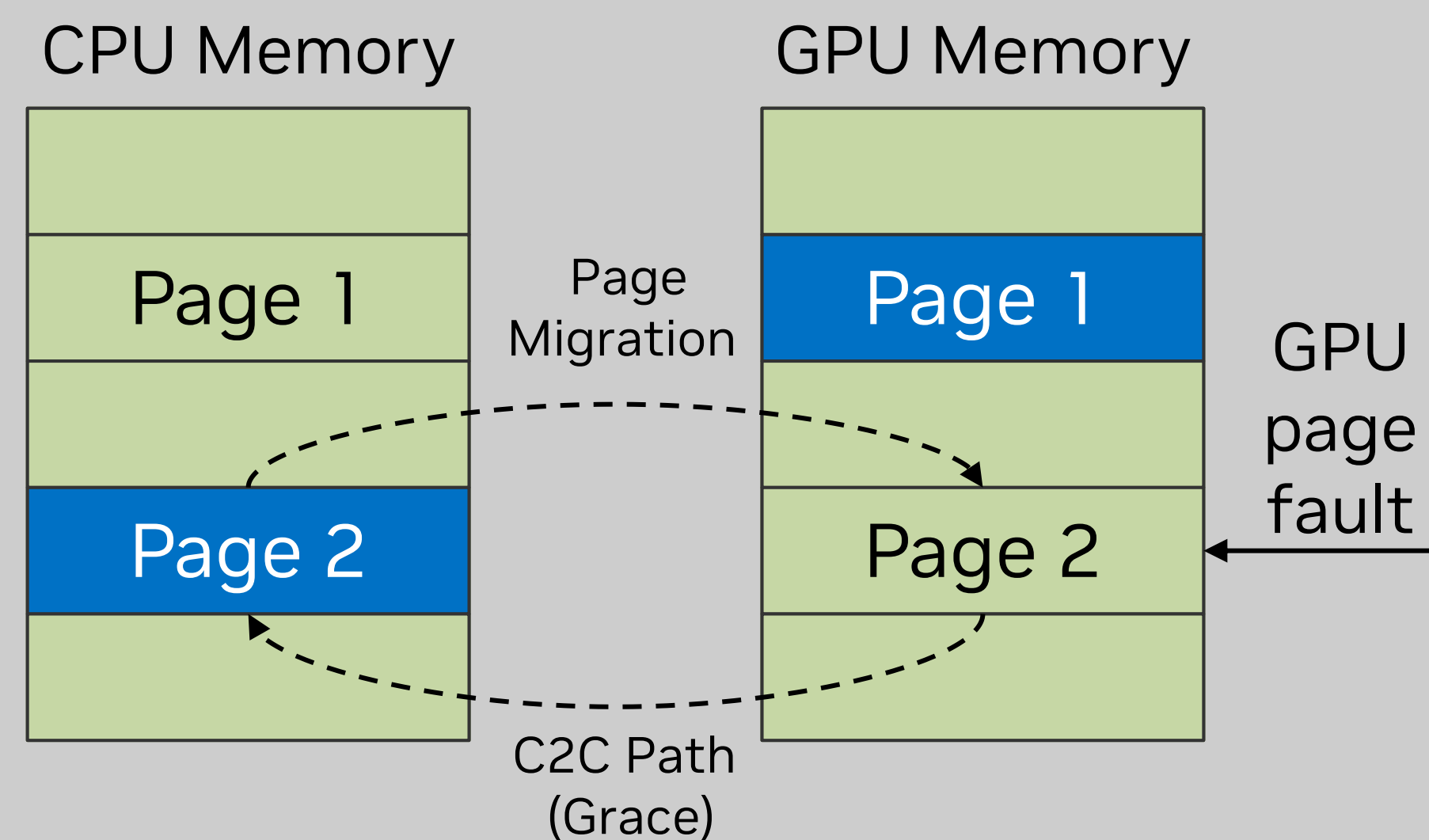
Explicit Copy

Application explicitly moves data between CPU & GPU as needed



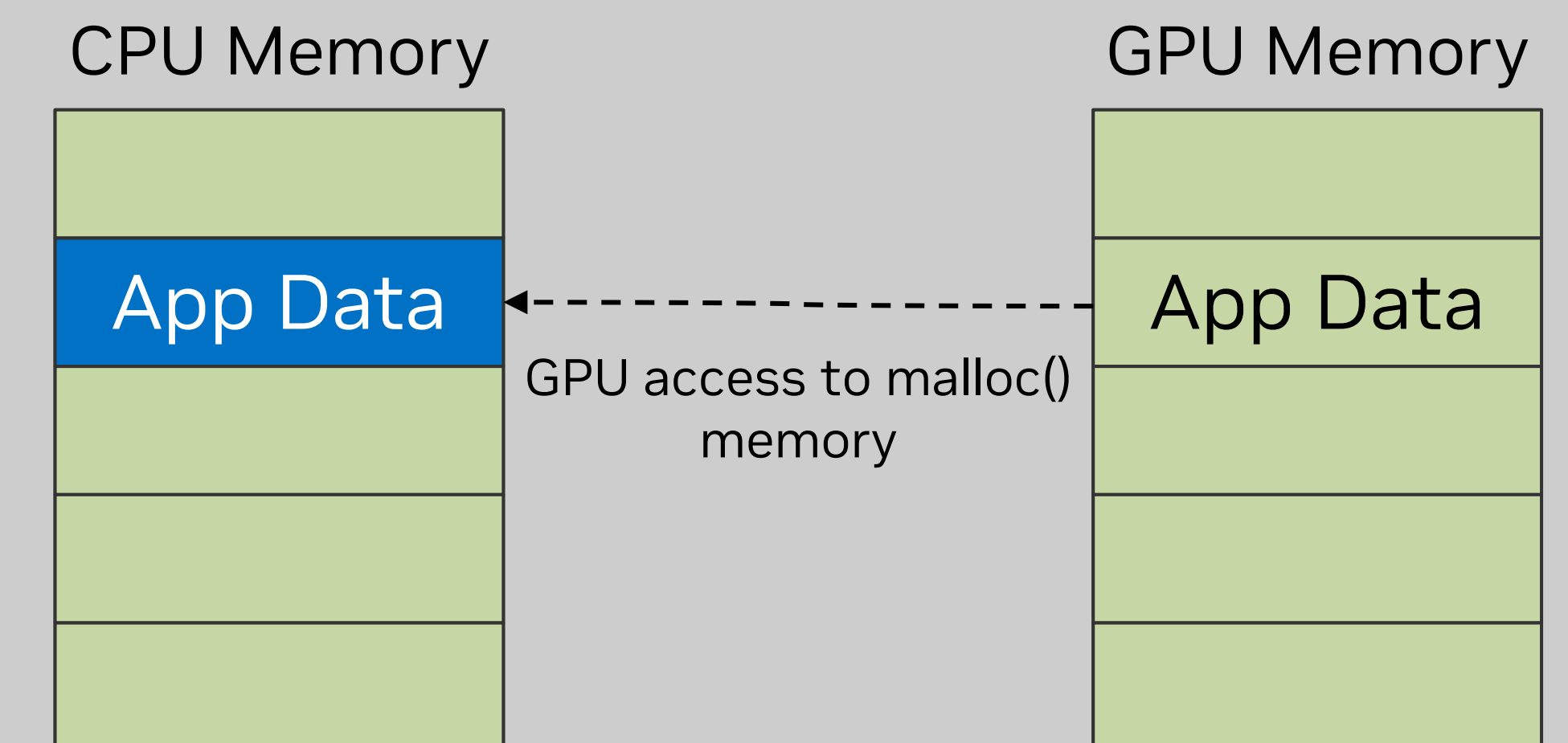
Managed Memory

CPU and GPU can access memory on-demand and data migrated locally for higher BW access



System Allocated

GPU can access memory allocated from malloc(), mmap(), etc.



HGX

~60 GB/s PCIe Gen5 transfers (H2D/D2H)

Requires migration to GPU

Access possible with explicit call to `cudaHostRegister()` at PCIe speeds
Requires HMM patch in Linux Kernel

G+H

7x faster transfers, up to 450 GB/s (NVLink C2C)

Migrations not required and faster migrations when they happen at NVLink C2C speed

`cudaHostRegister()` not needed; access at NVLink C2C speeds

Use-cases: NEMO on GH200

(Presented at GTC24 - S62337)

NEMO Ocean Model

A partially accelerated case utilizing unified memory on Grace-Hopper

The "Nucleus for European Modelling of the Ocean" (NEMO) is a state-of-the-art modelling framework, used for research activities and forecasting services in ocean and climate sciences.

- **Setup (NEMO v4.2.0)**

- **GYRE_PISCES** benchmark

- Scaling factor for grid resolution: **nn_GYRE = 25**
 - ~ORCA ½ grid
 - ~80 GB RAM, fits on single GPU

- **MPI-only**, single core to every MPI process for CPU runs

- **Incremental porting** on Grace-Hopper (**480GB**) using unified memory and access-counter based migrations

- Memory management left to runtime – **system-allocated memory with automatic migrations**

- compile with `-gpu=unified,nomanaged`

- Simply offloading loops to GPU using **OpenACC**, in 3 steps:

- **Horizontal (lateral) diffusion,**
- **Advection,**
- **Vertical diffusion and time-filtering,**

for both “active” (**TRA**) and “passive” (**TRC**) tracer transport

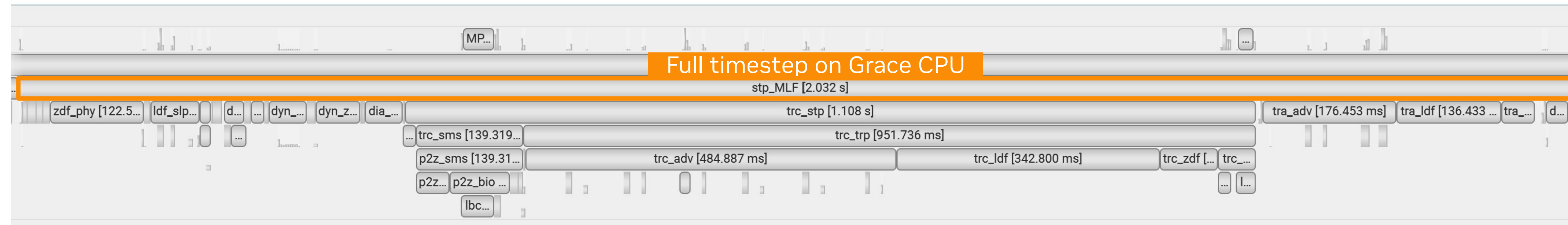


Image source:

[NEMO User Guide — NEMO release-4.2.2 documentation \(nemo-ocean.io\)](https://nemo-ocean.io/documentation)

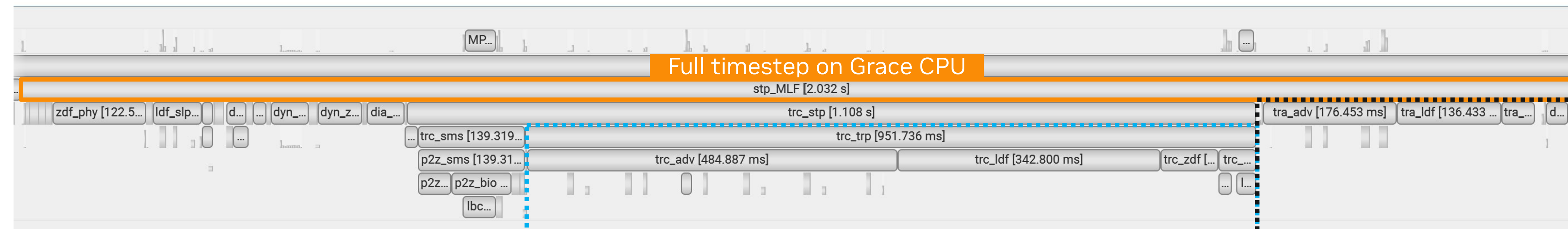
Porting NEMO to Grace-Hopper using Unified Memory

Incremental porting, zooming in to a single timestep ...



Porting NEMO to Grace-Hopper using Unified Memory

Incremental porting, zooming in to a single timestep ...

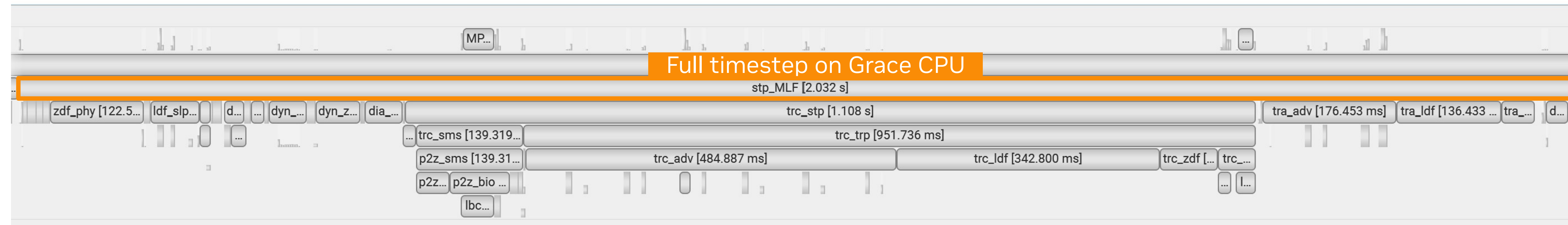


“Passive” tracer transport
(TRC)

“Active” tracer transport
(TRA)

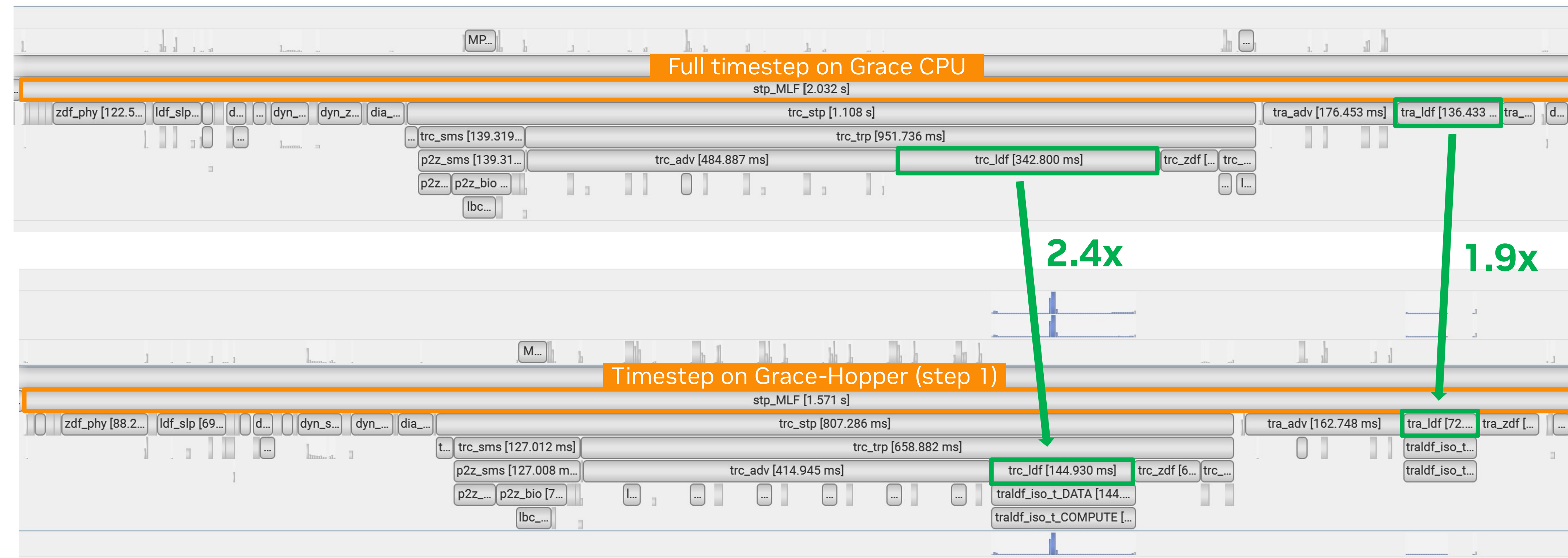
Porting NEMO to Grace-Hopper using Unified Memory

Incremental porting, zooming in to a single timestep ...



Porting NEMO to Grace-Hopper using Unified Memory

Incremental porting, zooming in to a single timestep ...



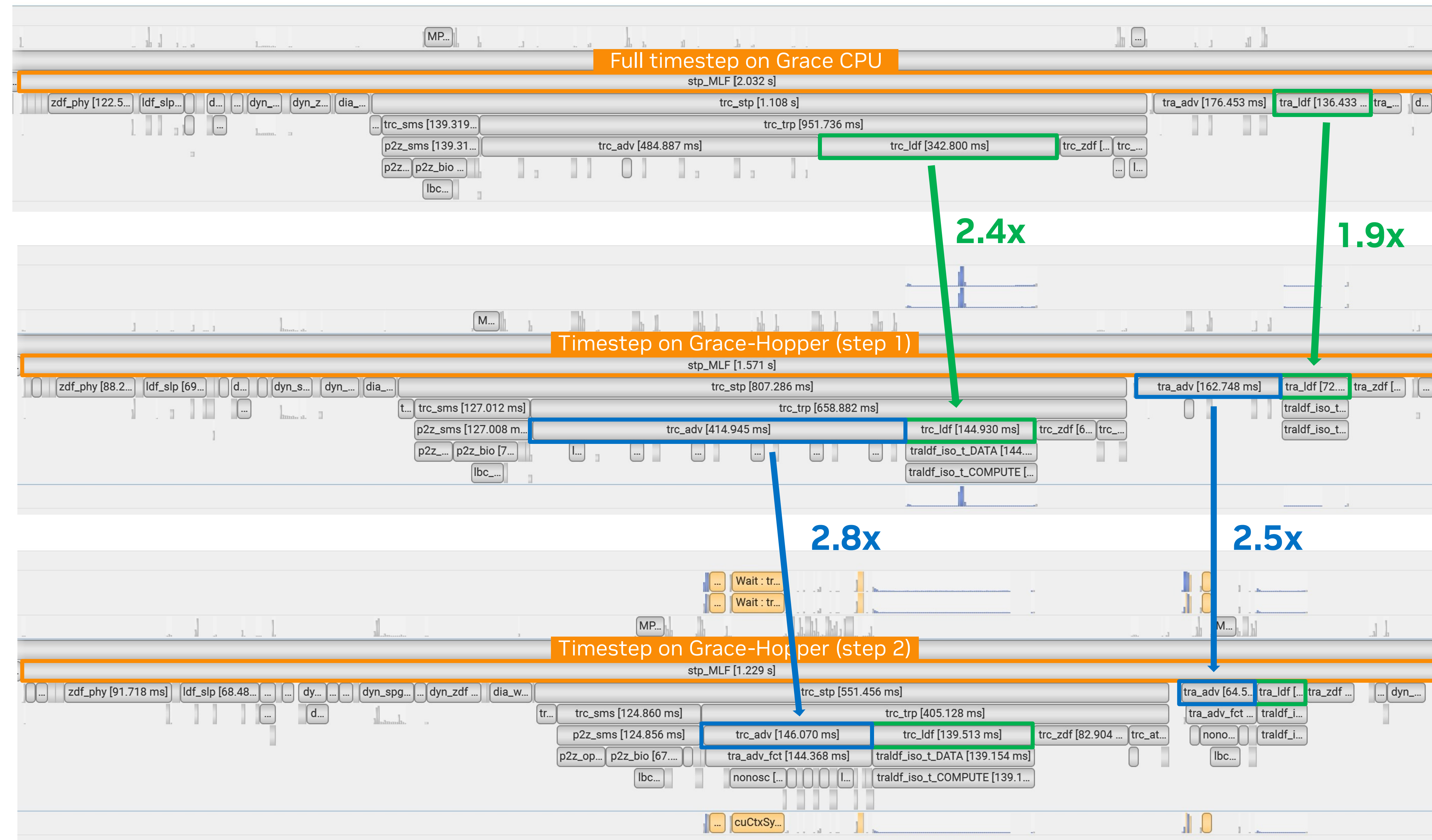
Ported to GPU:

- Horizontal diffusion

We run multiple (i.e. 40) MPI processes on **CPU and GPU** using **MPS**, and use “**migratable**” system allocated memory

Porting NEMO to Grace-Hopper using Unified Memory

Incremental porting, zooming in to a single timestep ...



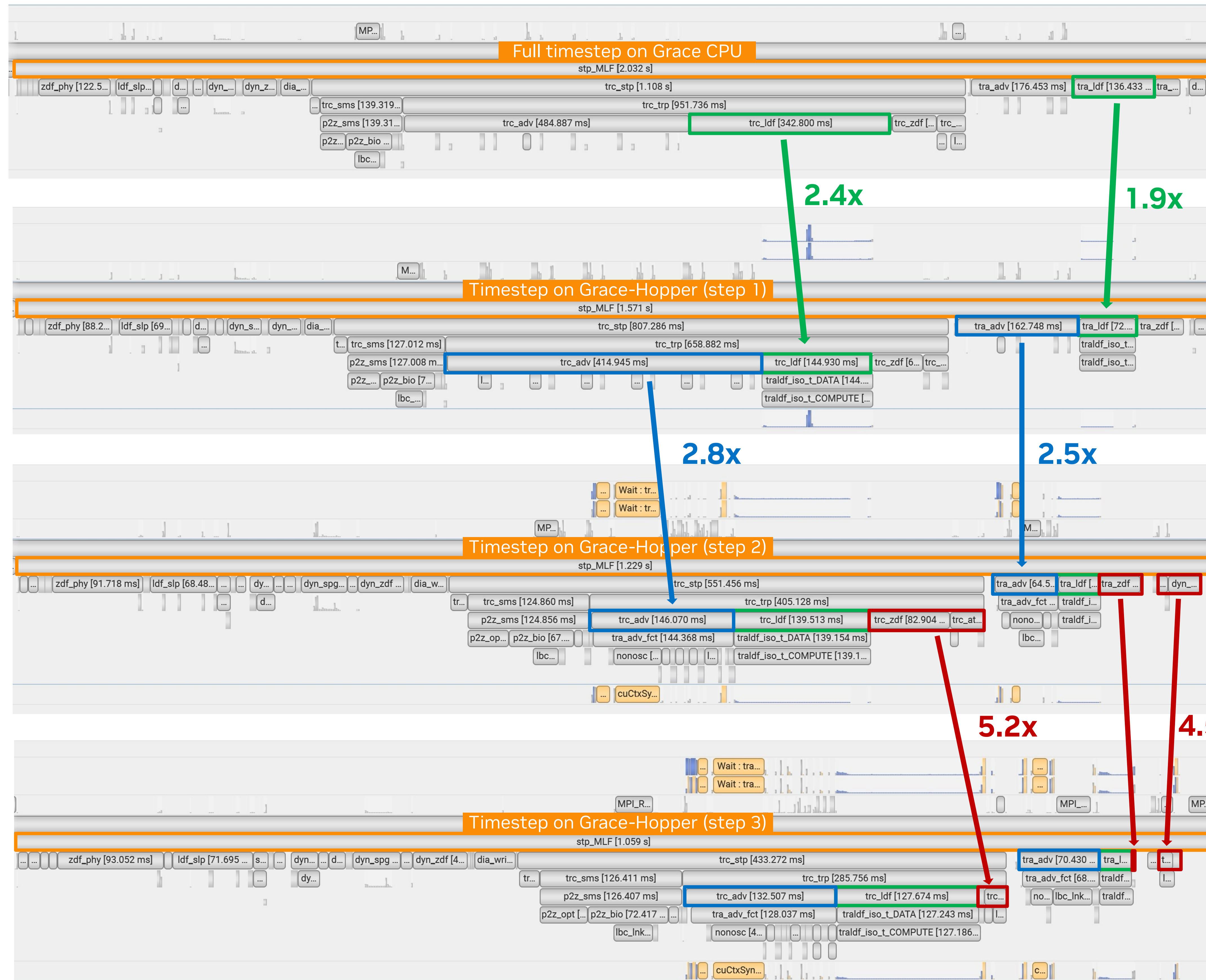
Ported to GPU:

- Horizontal diffusion
- Advection

We run multiple (i.e. 40) MPI processes on **CPU and GPU** using **MPS**, and use “**migratable**” system allocated memory

Porting NEMO to Grace-Hopper using Unified Memory

Incremental porting, zooming in to a single timestep ...



Ported to GPU:

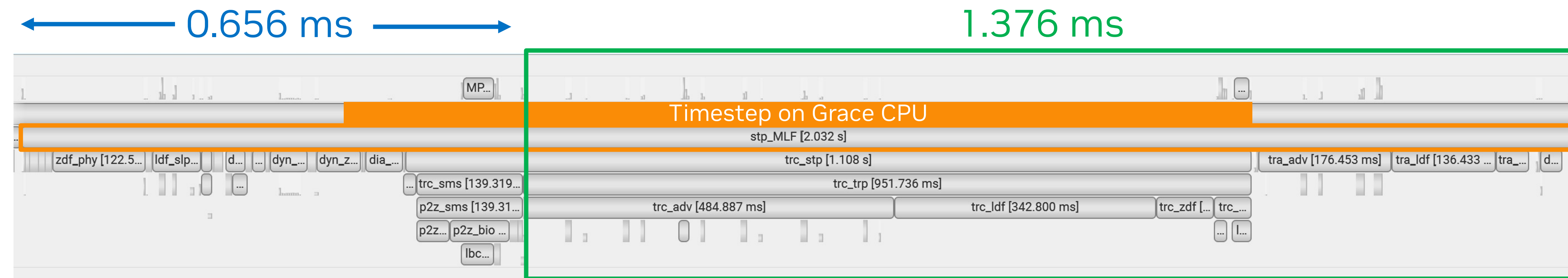
- Horizontal diffusion
- Advection
- Vertical diffusion and time-filtering

We run multiple (i.e. 40) MPI processes on **CPU** and **GPU** using **MPS**, and use “migratable” system allocated memory

Porting NEMO to Grace-Hopper using Unified Memory

A deeper look into the effect of access-counter based migrations on the partially accelerated port

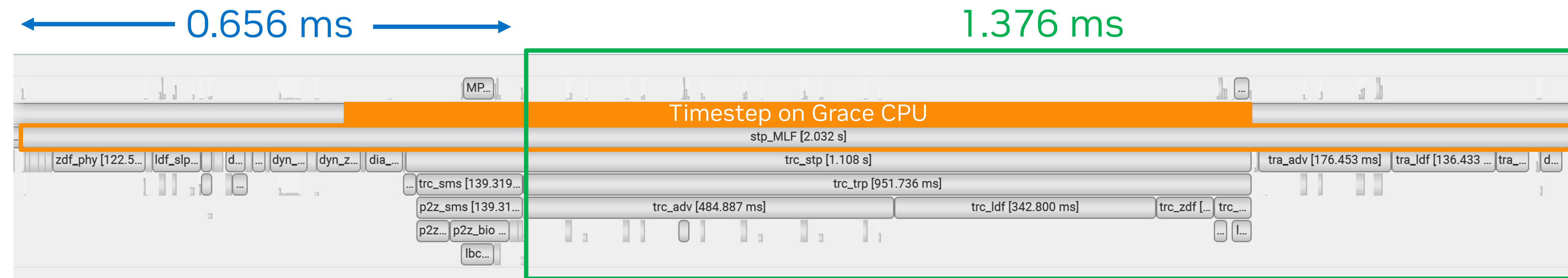
CPU only run



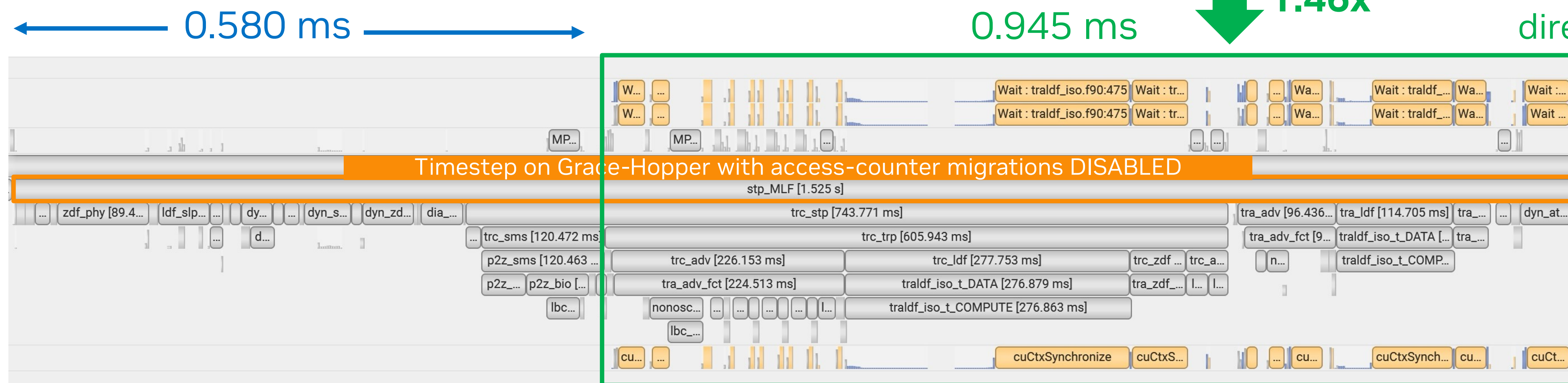
Porting NEMO to Grace-Hopper using Unified Memory

A deeper look into the effect of access-counter based migrations on the partially accelerated port

CPU only run



Tracer transport on GPU with migrations disabled (buffers with first-touch on CPU will never migrate to GPU memory)



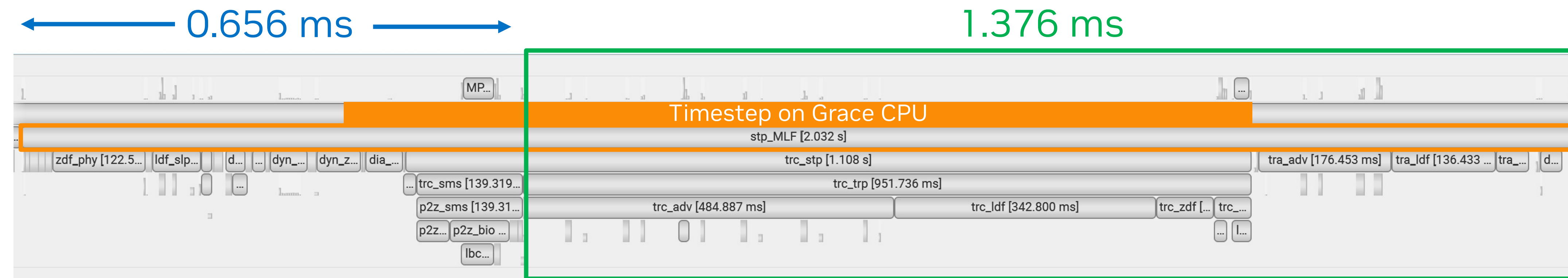
1.33x

1.44x

Porting NEMO to Grace-Hopper using Unified Memory

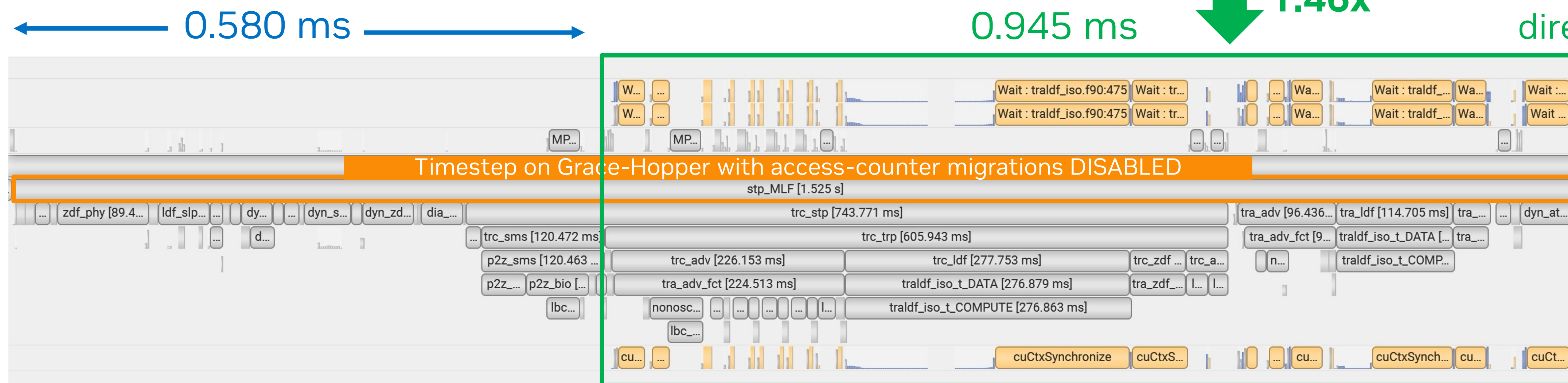
A deeper look into the effect of access-counter based migrations on the partially accelerated port

CPU only run



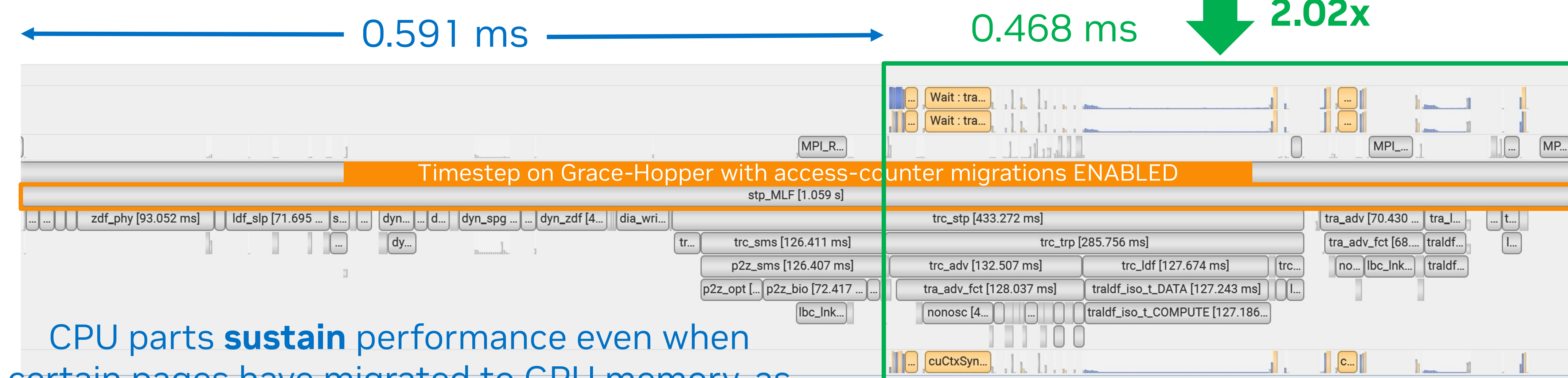
1.33x

Tracer transport on GPU with migrations disabled (buffers with first-touch on CPU will never migrate to GPU memory)



1.44x

Enabling automatic page migrations from CPU to GPU ("hot" pages migrate to GPU)



CPU parts **sustain** performance even when certain pages have migrated to GPU memory, as long as we use enough processes to saturate C2C BW

GPU kernels become faster as more and more pages migrate to GPU

Use-cases: Petrobras SolverBR

(Presented at GTC24 - S62529)


SolverBR




- The SolverBR is a **high-performance CPU sparse linear solver** specialized for reservoir simulation applications developed in collaboration with academic institutions.
- It combines excellence in parallel computing with the most advanced algorithms of sparse linear algebra for flow and geomechanics problems.

Journal of Computational Science 51 (2021) 101330

Contents lists available at [ScienceDirect](#)


 **Journal of Computational Science**

journal homepage: www.elsevier.com/locate/jocs



Hybrid parallel iterative sparse linear solver framework for reservoir geomechanical and flow simulation

Leonardo Gasparini^a, José R.P. Rodrigues^a, Douglas A. Augusto^b, Luiz M. Carvalho^{c,*}, Cesar Conopoima^d, Paulo Goldfeld^d, Jairo Panetta^e, João P. Ramirez^d, Michael Souza^f, Mateus O. Figueiredo^a, Victor M.D.M. Leite^g



Porting to Arm

• Build systems adjustments

- Added a new compile target for aarch64 architecture
 - *Mapped compatible compile variables*
- Initially used gcc for a “fair” and easier comparison

• Code porting adjustment

- Removed Intel Intrinsics
 - *Evaluated the usage of SIMD and SSE2NEON libraries as replacement*
- Addressed synchronization issues that led to precision errors in floating-point calculation
- Follow best practices

```

6  #ifndef __aarch64__
7  #include <xmmtrin.h>
8  #else
9  #include <utils/simd/sse2neon.h>
10 #endif

```

```

35 ALWAYS_INLINE void P2PNotify( const int task, volatile int * taskFinished )
36 {
37 #ifdef __aarch64__
38     OPENMP ( omp flush )
39     OPENMP ( omp atomic write )
40 #endif
41     taskFinished[ task ] = 1;
42 }

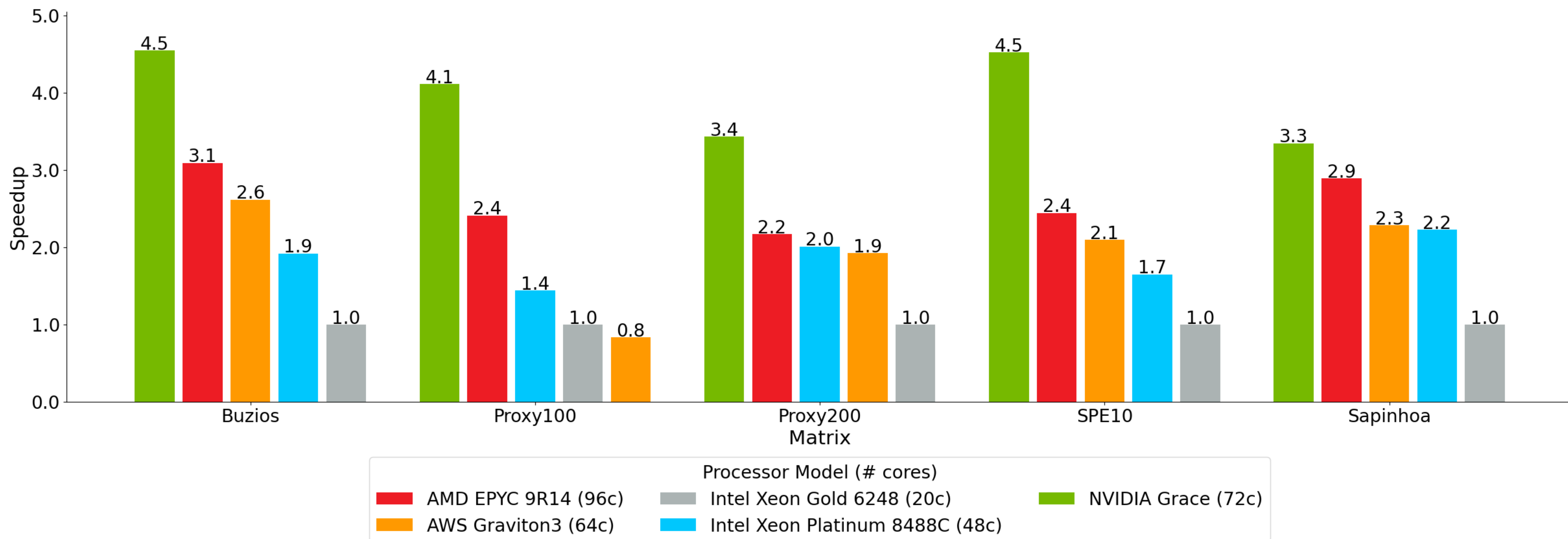
```

```

14 ALWAYS_INLINE void P2PWait( const int threadTask, const int * parentIndex,
15     const int * parents, const volatile int * taskFinished )
16 {
17     for ( int tIdx = parentIndex[ threadTask ]; tIdx < parentIndex[ threadTask + 1 ]; ++tIdx )
18     {
19 #ifdef __aarch64__
20         const int t = parents[ tIdx ];
21         int finished;
22         do
23         {
24             OPENMP ( omp flush )
25             OPENMP ( omp atomic read )
26             finished = taskFinished[ t ];
27         } while ( finished == 0 );
28 #else
29         const int t = parents[ tIdx ];
30         while ( taskFinished[ t ] == 0 ) PAUSE;
31 #endif
32     }
33 }

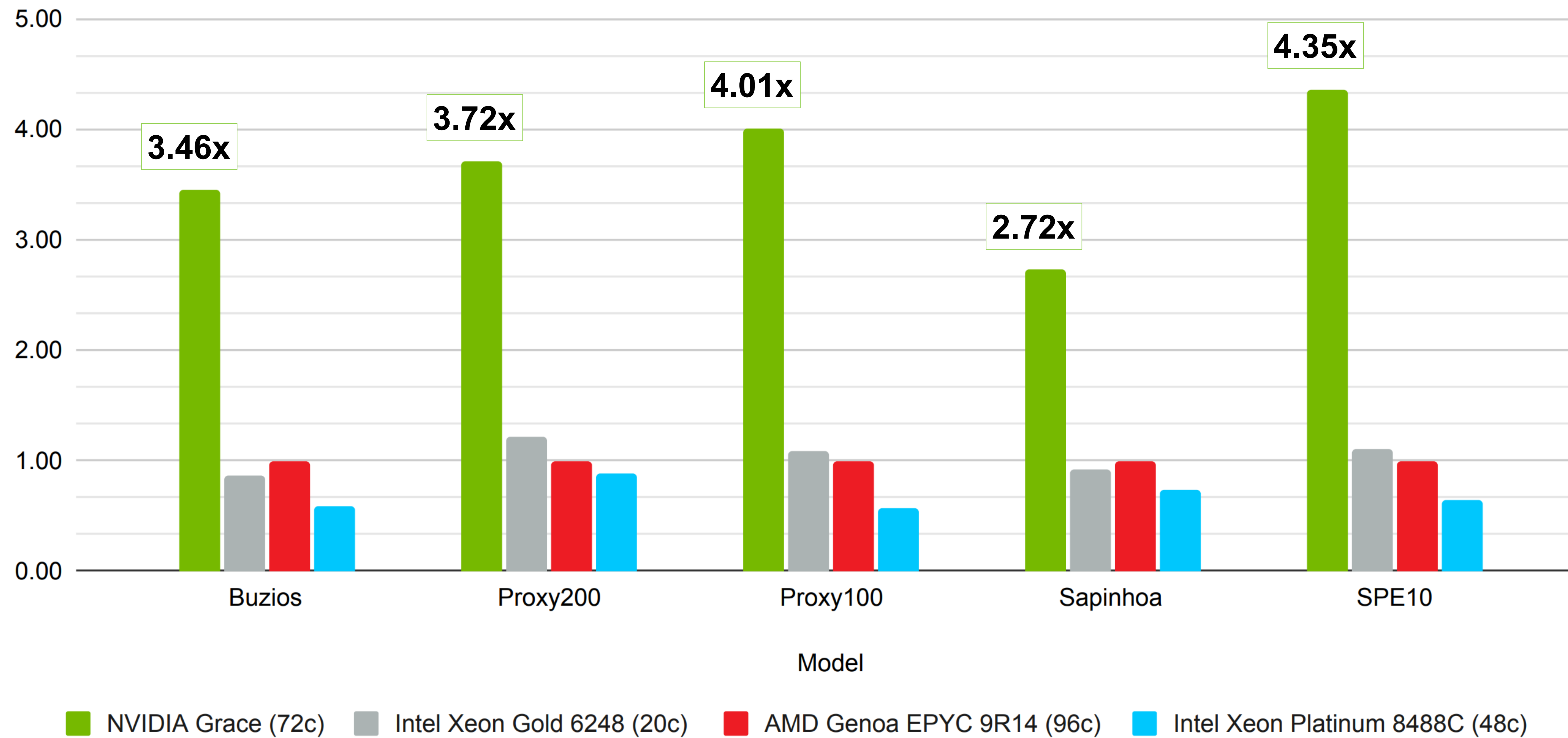
```

Single-socket Speedups (max core-count)



Average of 150 executions per matrix (considering 3 different timesteps)

Estimated Energy Efficiency at max load



Estimated are computed considering CPU max TDP at full load plus estimated memory consumption based on capacity and technology. We assume an average of 3W per 8 GB for DDR4 and 4.75W per 16 GB for DDR5. For Grace SoC, CPU plus memory is ~250W. Speed-ups are computed using AMD EPYC 9R14 as baseline.

To conclude...

E4 and NVIDIA Partnership

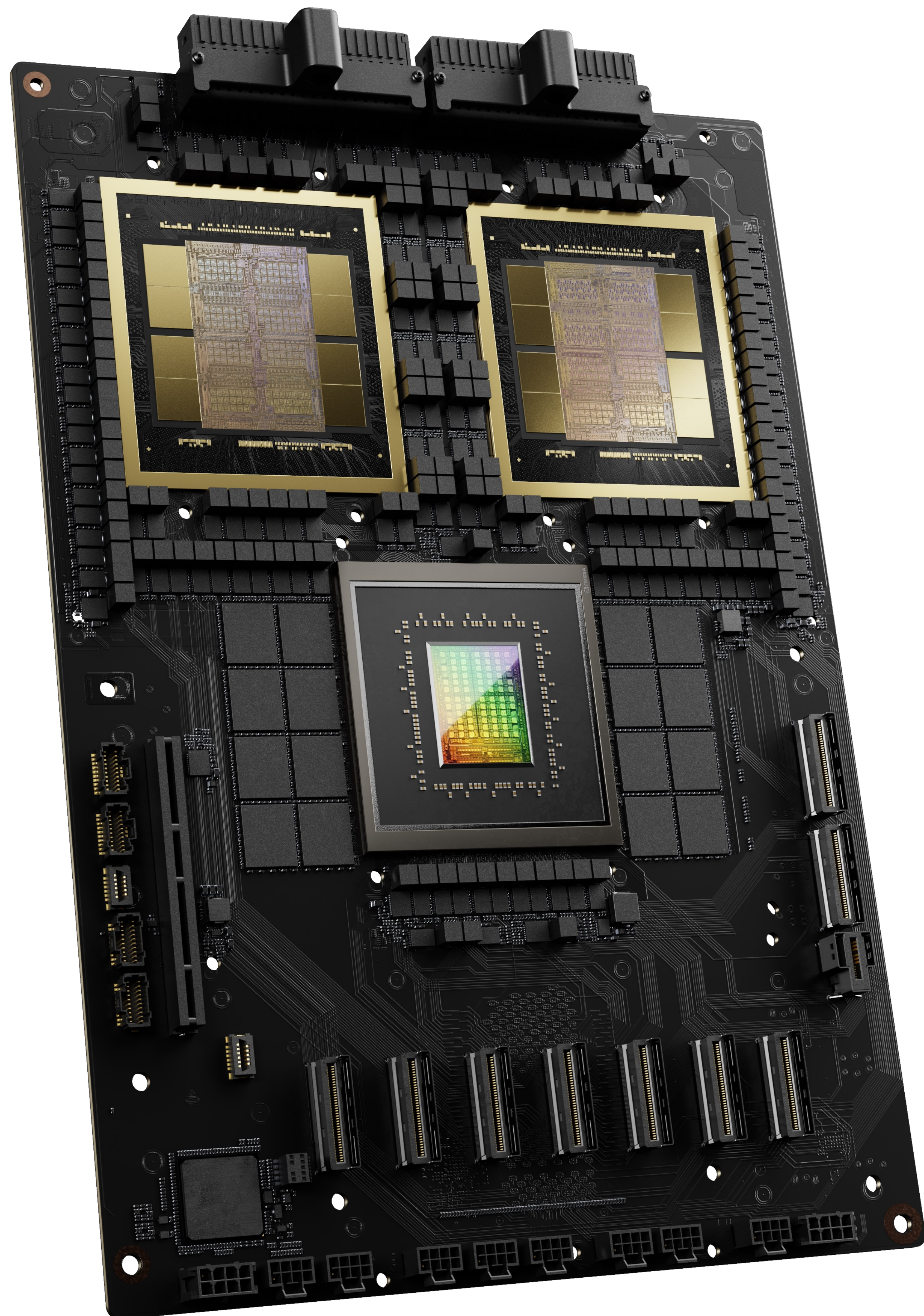
Relentless pursue of innovation and added value

- Remote access to GH200 and Grace Superchip
 - **CHIRON LAB:** <https://www.e4company.com/en/chiron-lab/>
- Systems from various OEM/ODM
 - Ability to run comparison across architectures
 - Ability to measure server power consumption
- E4 long-standing experience in Arm-based systems
- E4 expertise in system configuration and bring-up
- NVIDIA expertise in GPU application tuning

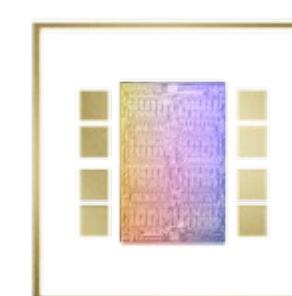
Contact: Marco Cicala (marco.cicala@e4company.com)



www.e4company.com



ANNOUNCING NVIDIA BLACKWELL PLATFORM FOR TRILLION-PARAMETER SCALE GENERATIVE AI



AI SUPERCHIP
208B Transistors



2nd GEN TRANSFORMER ENGINE
FP4/FP6 Tensor Core



5th GENERATION NVLINK
Scales to 576 GPUs



RAS ENGINE
100% In-System Self-Test



SECURE AI
Full Performance
Encryption & TEE



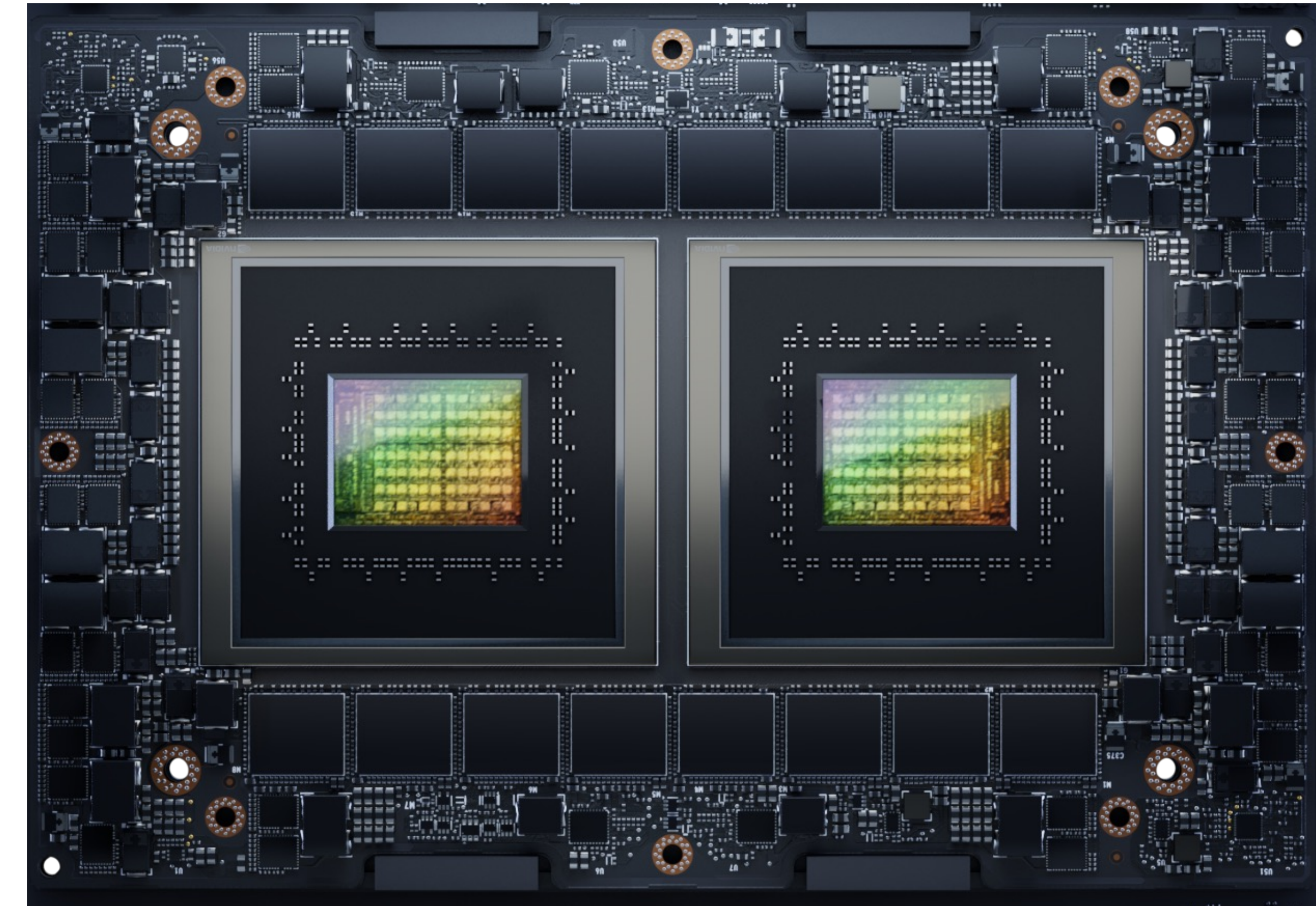
DECOMPRESSION ENGINE
800 GB/sec



Further Resources for Grace CPU and Grace Hopper

Grace CPU Superchip

- [Grace CPU Customer Deck](#)
- [Grace CPU Superchip Architecture Whitepaper](#)
- [Grace CPU Architecture In-Depth Blog](#)
- [Grace CPU Superchip Data Sheet](#)
- [Grace CPU Energy Efficiency Blog](#)
- [A Demonstration of AI and HPC Applications for NVIDIA Grace CPU \[S51880\]](#)
- [Grace CPU Power Efficiency Video](#)
- [Unlock the Power of NVIDIA Grace and Hopper with Foundational HPC Software](#)



GH200 Grace Hopper Superchip

- [GH200 Grace Hopper Customer Deck](#)
- [Grace Hopper Superchip Architecture Whitepaper](#)
- [Grace Hopper Architecture In-Depth Blog](#)
- [Grace Hopper Superchip Architecture Data Sheet](#)
- [Grace Hopper Recommender System Blog](#)
- [Programming Model and Applications for the Grace Hopper Superchip \[S51120\]](#)
- [Accelerating HPC applications with ISO C++ on Grace Hopper \[S51054\]](#)
- [Deploying RAG Applications on NVIDIA GH200](#)

