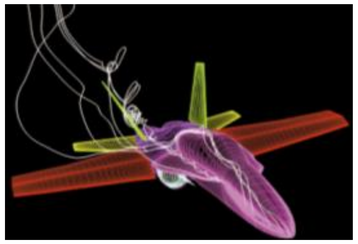
Developments for CFD Applications on GPUs and Arm CPUs

Stan Posey, Program Manager, CFD Domain, NVIDIA
John Linford, PhD, Principal Technical Product Manager, NVIDIA

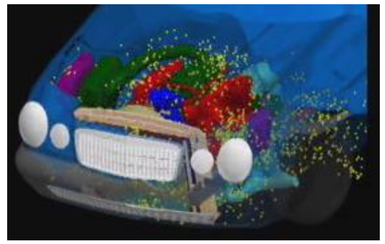
Advanced Modeling & Simulation (AMS) Seminar Series
NASA Ames Research Center,
September 7th, 2023

Drivers for CFD in Engineering Practice



Research CFD Objectives

- Motivation – Global advancement in areas of defense, energy, education, and others
- CFD Software – Need for high-performance compilers, development tools, and libraries
- Applications – Efficiency in parallel algorithms for advanced CFD capabilities



Industry CFD Objectives

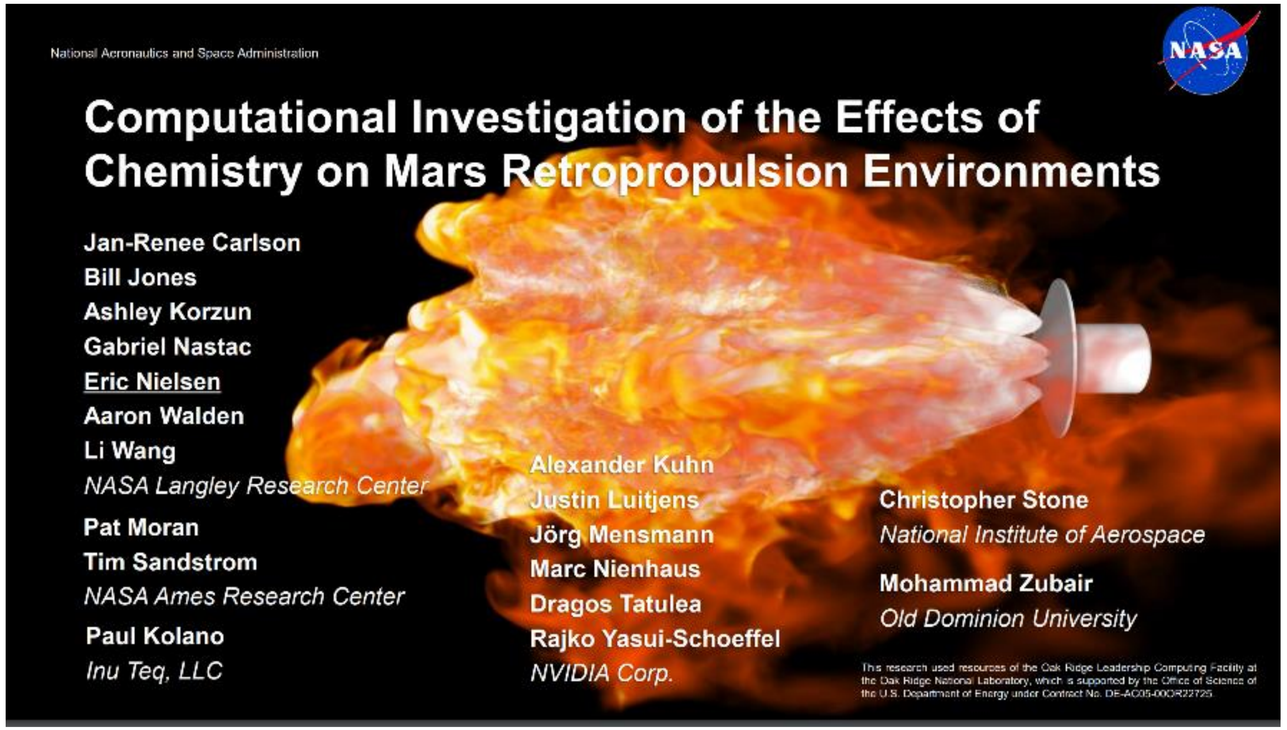
- Motivation – Competitive product developments that satisfy regulations and consumers
- CFD Software – Need for accuracy and high-performance from commercial ISV software
- Applications – Efficiency in job through-put to satisfy design optimization targets

Select Collaborations for GPU Accelerated CFD

	Org	Software	Method	GPU Features
Community:				
	NASA	FUN3D	CUDA	Full application, RG chemistry
	NASA	OVERFLOW	OpenACC	Central diff schemes, Euler flux, smoothing
	DoD NRL	JENRE	CUDA	Full application, high-order FE LES
Argonne NATIONAL LABORATORY	DOE ANL	NekRS	OCCA (CUDA)	Full application, high-order spectral element
OpenFOAM®	ESI-OpenCFD	OpenFOAM	CUDA Lib	Linear solver only using AmgX lib
Custom:				
BOEING	Boeing	BCFD	OpenACC	Full application, 2nd order FV RANS
	GE	GENESIS	CUDA	Full application, high-order LES
Commercial:				
Ansys	ANSYS	Fluent	CUDA	Full application, core features
SIEMENS	Siemens	STAR-CCM+	CUDA	Full application, core features
ALTAIR	Altair	ultraFluidX	CUDA	Full application LBM, core features
cadence	Cadence	CharLES	CUDA	Full application, core features

NVIDIA.

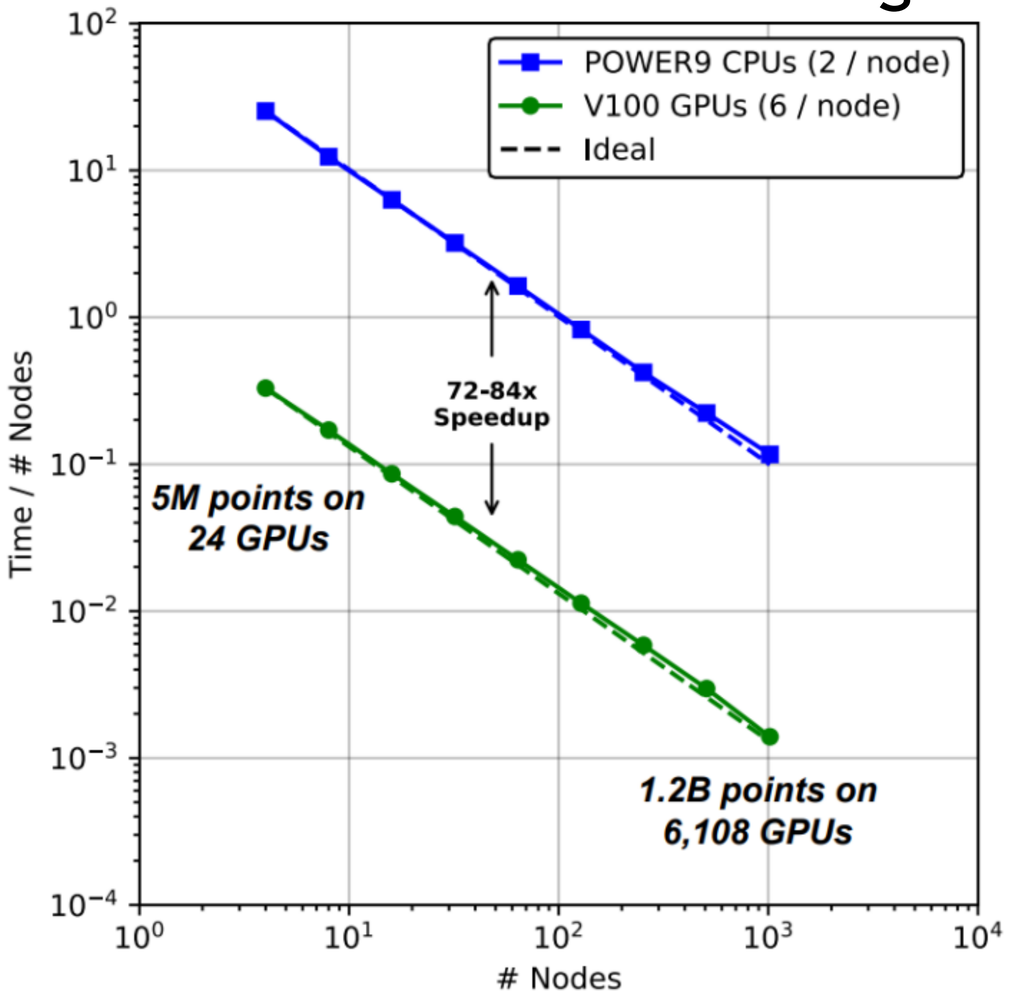
FUN3D Performance Results from GTC Mar 2022



2 x 64-core AMD 7742	1.0x
NVIDIA V100 32 GB	4.0x
NVIDIA A100 40 GB	7.0x

https://www.nvidia.com/en-us/on-demand/session/gtcspring22-s41286/

ORNL Summit Scaling



OVERFLOW GPU Performance on NVIDIA GPUs

Results for DPW6 WB CRM 25M

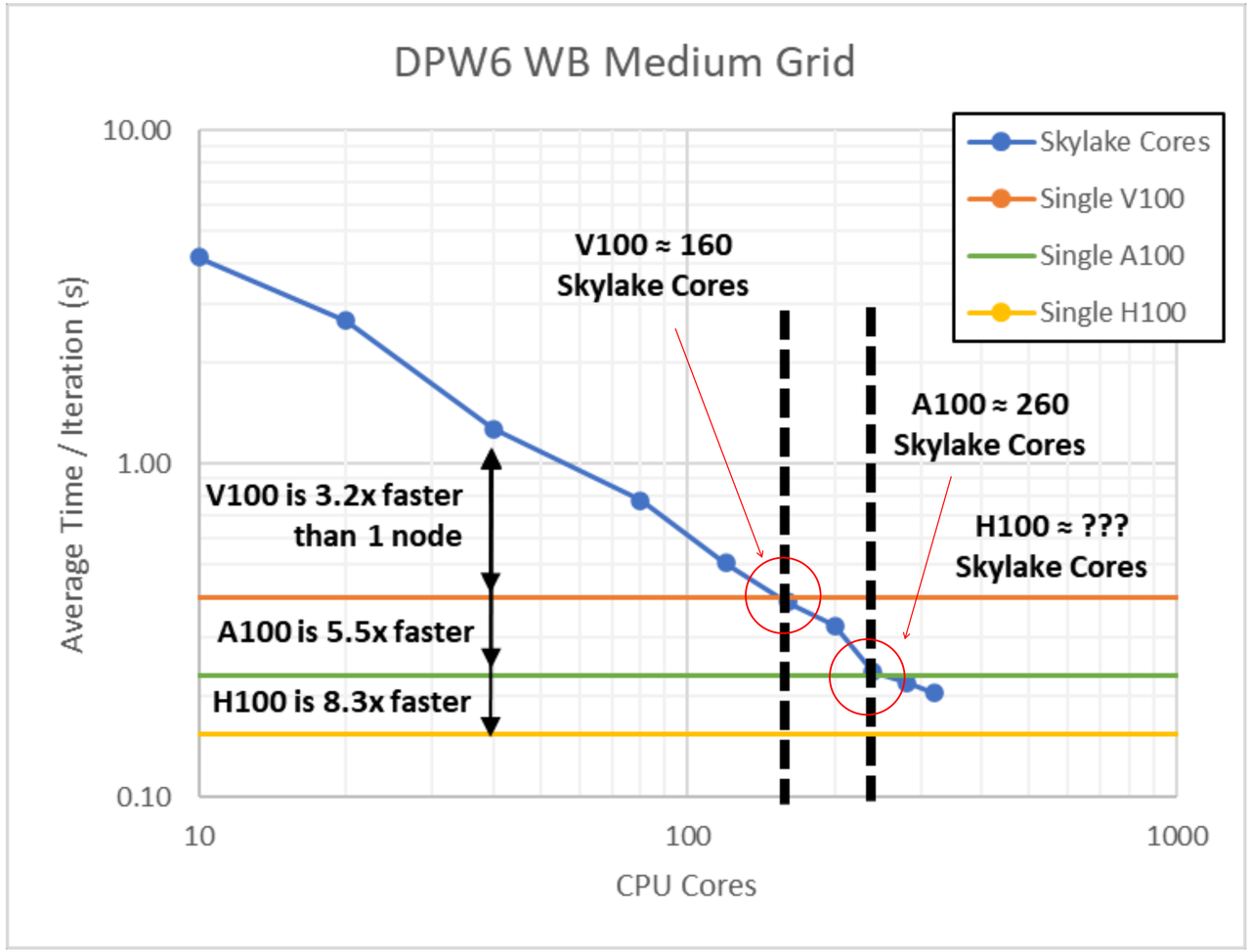
- NASA CRM Wing- Body configuration with prescribed static aeroelastic wing twistAdjusts font size/designation
- https://aiaa-dpw.larc.nasa.gov/

- 6.5 Skylake nodes (40 cores) measured on Pleiades to reach same performance as 1 x A100 GPU

- 10 Skylake nodes needed to match a single H100 (estimated)

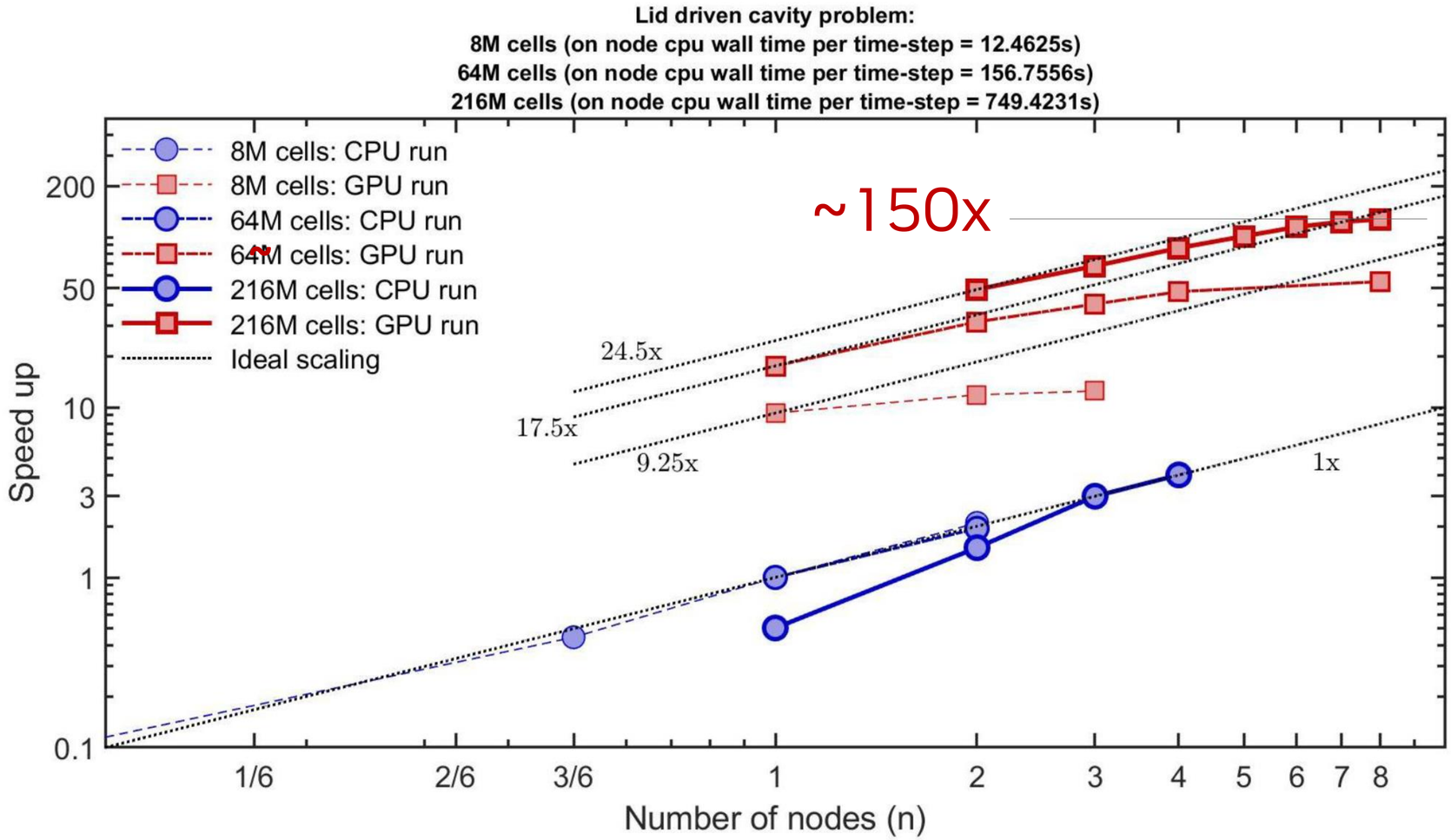
- Multi-GPU will experience *intra-node* scaling up to 8 GPUs→ better scaling than CPU experience.

- NASA LaRC PI Chip Jackson (YT video Nov 2022):
  Porting OVERFLOW CFD Code to GPUs To Hackathons and Beyond



OpenFOAM GPU Strong Scaling on ORNL Summit

Strong scaling on Summit



Running GPU-enabled OpenFOAM on Summit

FERMI Project

Dr. Arpan Sircar, Dr. Vittorio Badalassi

DOE Oak Ridge National Laboratory

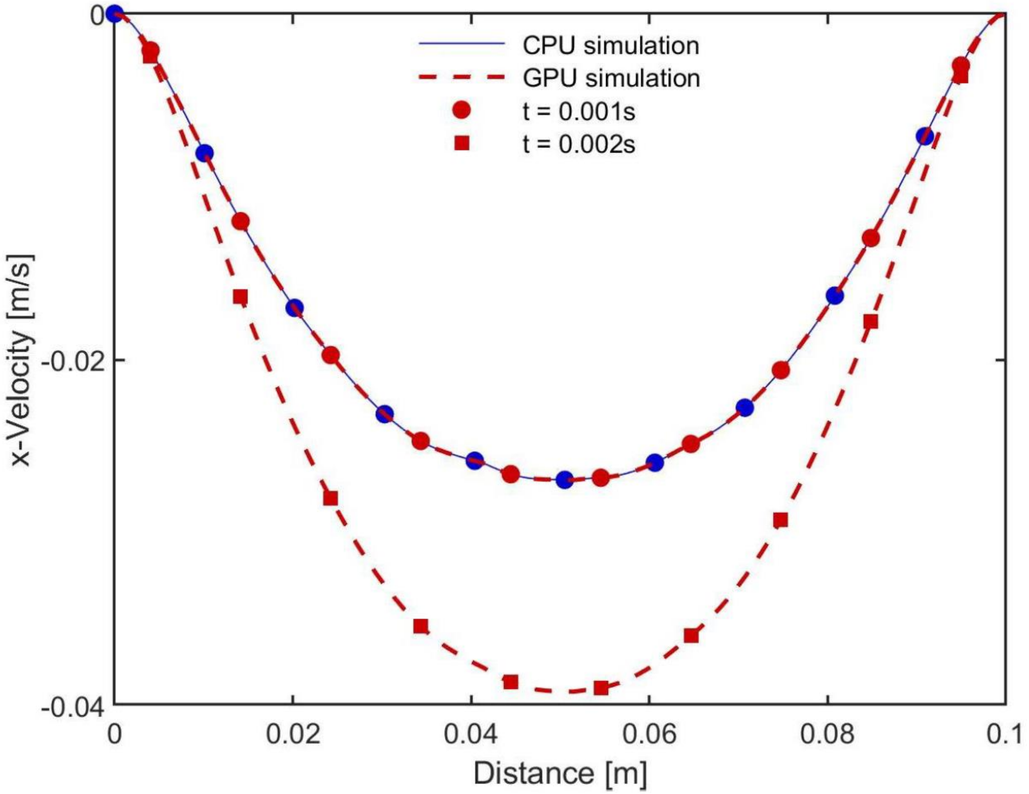
OAK RIDGE National Laboratory

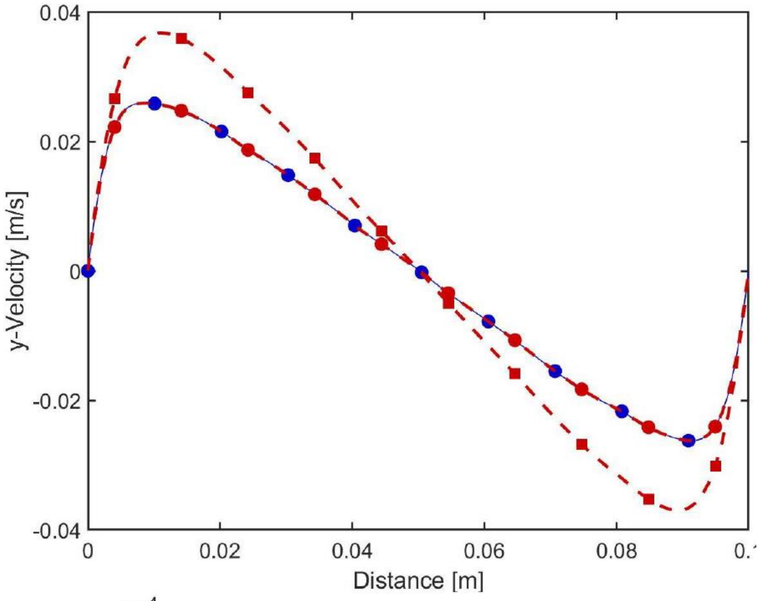
NVIDIA

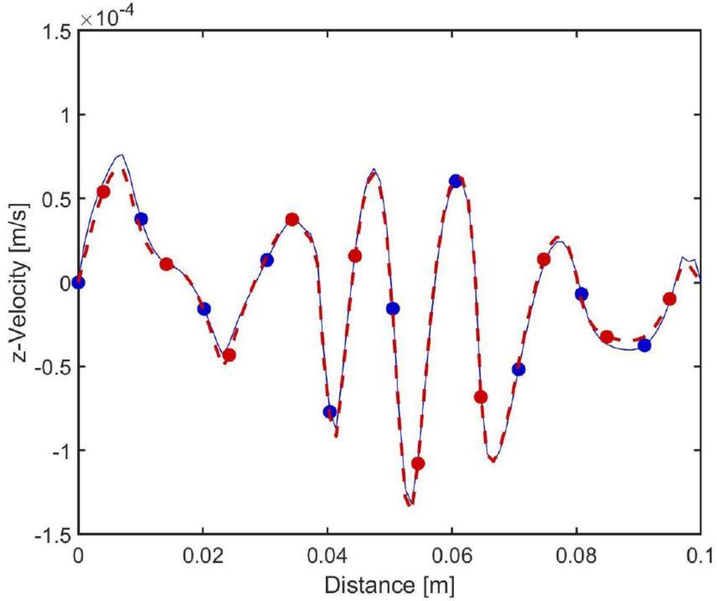
DOE ORNL OpenFOAM GPU Validation Study

Verifying results between CPU and GPU

(results extracted along x-axis on the mid-plane of a unit cube with unit top-plate velocity for 216M cell mesh)







Running GPU-enabled OpenFOAM on Summit

FERMI Project

Dr. Arpan Sircar, Dr. Vittorio Badalassi

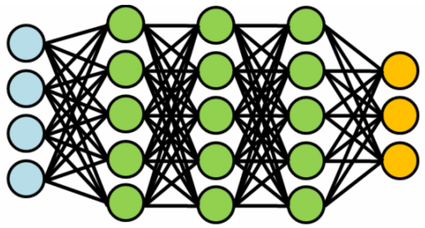
DOE Oak Ridge National Laboratory

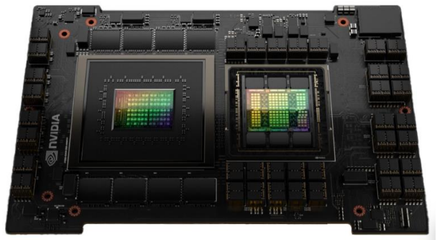
OAK RIDGE National Laboratory

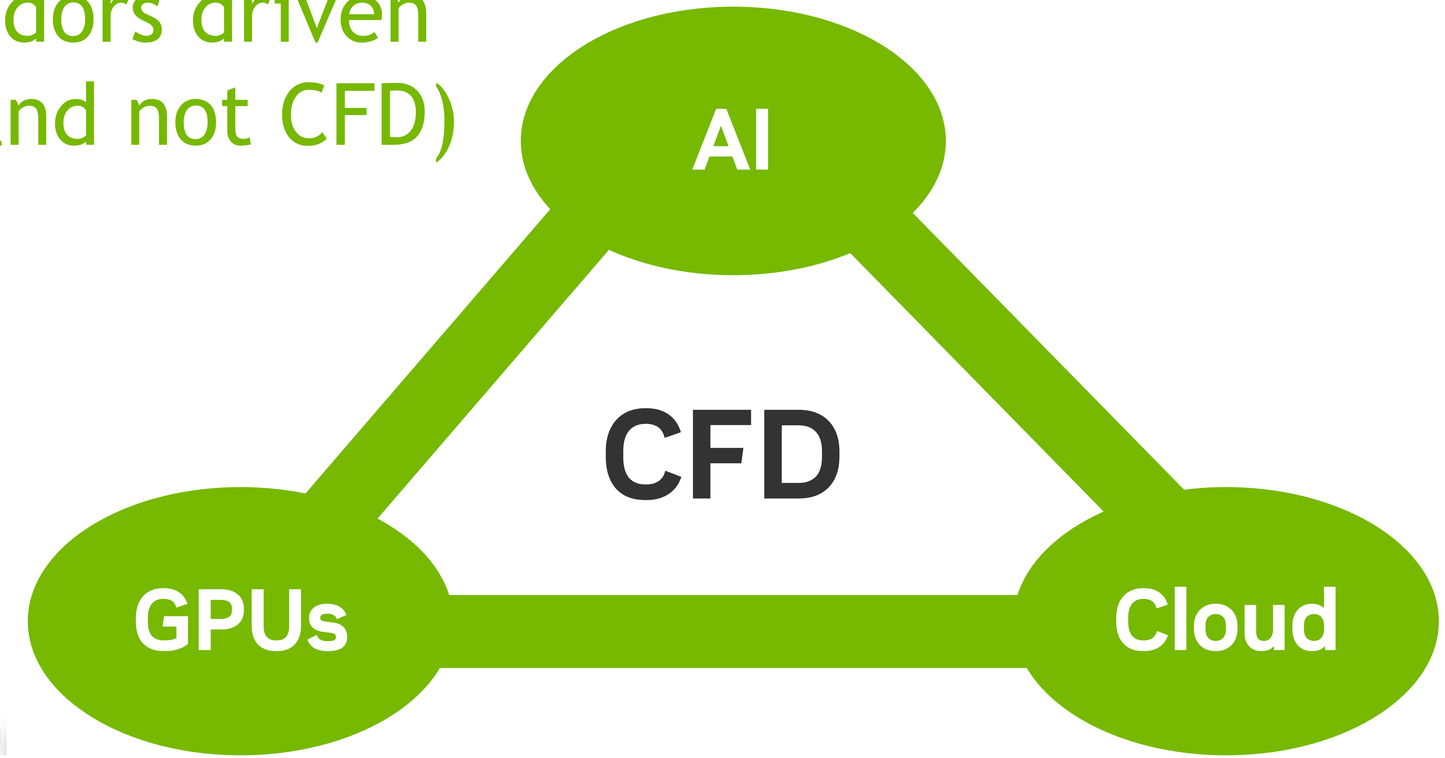
NVIDIA

HPC Technologies Driving Novel CFD Trends

Reality: HPC opportunity for system vendors driven by AI market (and not CFD)







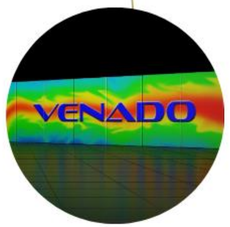


Exascale AI Systems also Run CFD Applications





NVIDIA (EOS)
18 EFLOPS AI Perf
Quantum-2 400Gb/s InfiniBand



LANL (Venado)
10 EFLOPS AI Perf



BSC (MareNostrum 5)
18 EFLOPS AI Perf
Quantum-2 400Gb/s InfiniBand



CSCS (ALPS)
20 EFLOPS AI Perf



CINECA (Leonardo)
10 EFLOPS



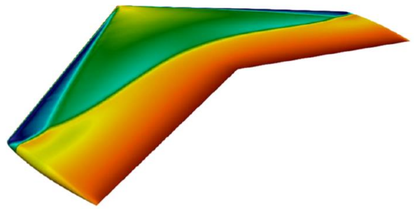
KAUST (Shaheen-III)
7 EFLOPS AI Perf

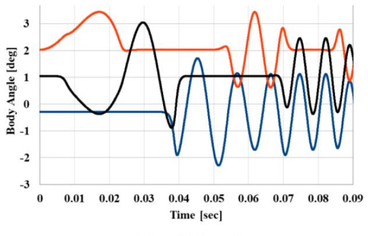
Select AI-Based CFD Applications in Aerospace

- Machine Learning-Based Surrogate Modeling for Aerodynamic Loads Predictions

  J Abras, T Tuckey, DR McDaniel, NS Hariharan

  AIAA SCITECH 2023 Forum, 0232

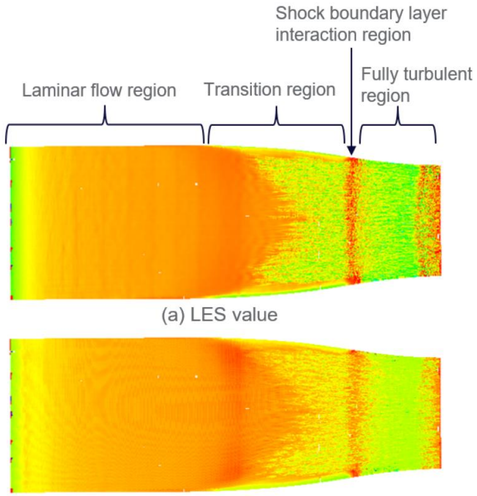




DOD HPC MODERNIZATION PROGRAM

CREATE Computational Research and Engineering Acquisition Tools and Environments

GE Research



OAK RIDGE National Laboratory

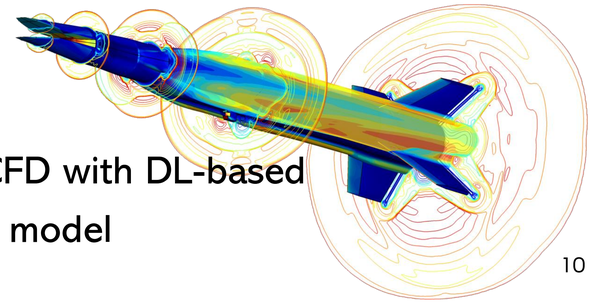
"Science-Guided Machine Learning for Wall-Modeled Large Eddy Simulation," R. Bhaskaran, R. Kannan, B. Barr and S. Priebe, *2021 IEEE Intl Conference on Big Data*, Orlando, FL, USA, 2021, pp. 1809-1816, doi: 10.1109/BigData52589.2021.9671436.

"Deep neural network based unsteady flamelet progress variable approach in a supersonic combustor

Sinan Demir, Prithwish Kundu, Austin C. Nunno, Sibendu Som, Robert A. Baurle and Tomasz G. Drozda. AIAA 2022-2073. AIAA SCITECH 2022 Forum. January 2022. doi: 10.2514/6.2022-2073

NASA

Argonne NATIONAL LABORATORY



VULCAN-CFD with DL-based chemistry model

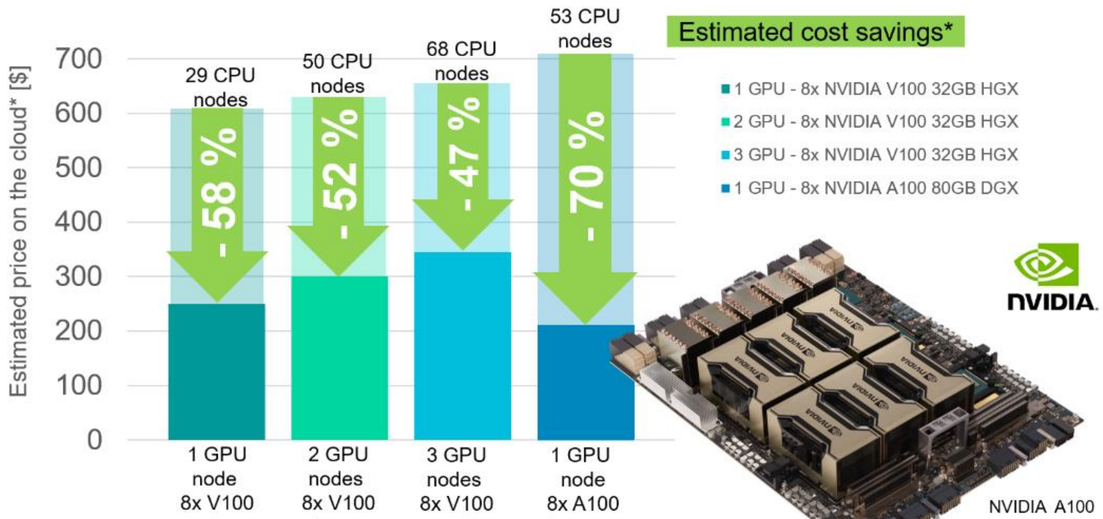
NVIDIA

Commercial CFD Software Towards GPUs and Cloud

SIEMENS Digital Industries Software

More bang for the buck with CFD on GPUs on the cloud

If we translate those numbers into the cost of running on the cloud we realize that on NVIDIA GPUs we get a significant cost reduction. An instance of GPUs 8x NVIDIA V100 costs approximately $25, 8x NVIDIA A100 are about $33, while an instance of CPU (1 dual-socket Xeon Gold node) costs $2.10 dollars.



*Cost saving may vary. CPU count chosen to match GPU runtime. Prices for GPU and CPU compute resource estimated from popular cloud vendors: 8x NVIDIA V100 32GB HGX $25/hr; 8x NVIDIA A100 80GB DGX $33/hr; Compute-optimized CPU instance dual-socket Xeon Gold nodes (40 cores per node) $2.10/hr

https://blogs.sw.siemens.com/simcenter/gpu-acceleration-for-cfd-simulation/

COUNTRIES & REGIONS | CONTACT US | CAREERS | STUDENTS AND ACADEMIC

Why Ansys | Products & Services | Learn | Ansys

ANSYS BLOG FEBRUARY 25, 2022

Unleashing the Power of Multiple GPUs for CFD Simulations

Computational fluids dynamics (CFD) engineers are keenly interested in accelerating their simulation throughput, whether that's by automating workflows, upgrading to newer/better methods, or using high-performance computing (HPC).

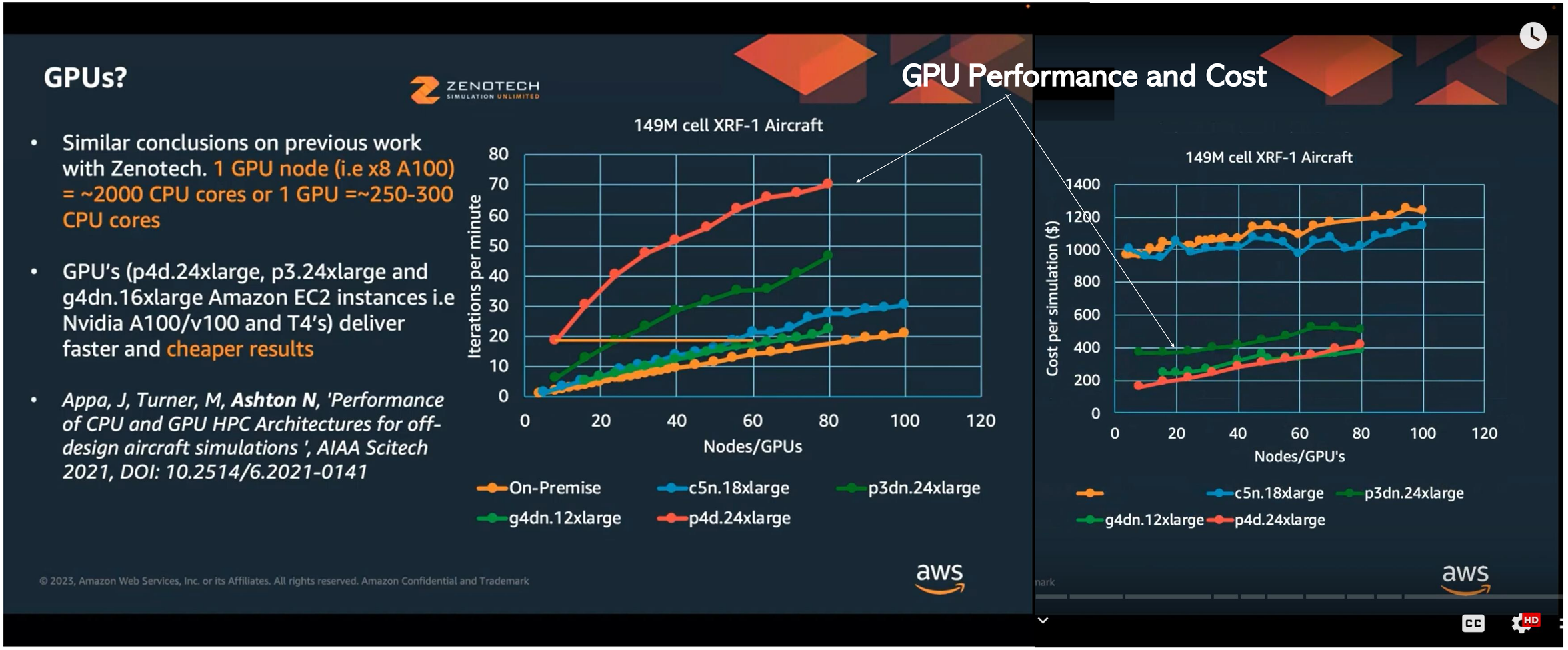
https://www.ansys.com/blog/unleashing-the-full-power-of-gpus-for-ansys-fluent

CFD startups promote latest HPC trends

- **FlexCompute:** www.flexcompute.com/
- **Luminary Cloud:** www.luminarycloud.com/
- **Volcano Platforms**
- **AI Engineering:** www.ai-eng.com/

AWS Performance-Cost Study for Cloud CFD

GPUs More Favorable Performance-Cost Profile



Source: 18th OpenFOAM Workshop, 11-14 July 2033 – Dr. Neil Ashton, AWS

Developments for CFD Applications on GPUs and Arm CPUs

Stan Posey, Program Manager, CFD Domain, NVIDIA
John Linford, PhD, Principal Technical Product Manager, NVIDIA

NVIDIA Developments for CFD Applications on GPUs and Arm CPUs

John Linford, Ph.D. | Principal Technical Product Manager

jlinford@nvidia.com



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.

NASA NAS AMS Seminar



Agenda

- Introduction to NVIDIA Grace Superchips
- NVIDIA Grace Software Ecosystem and Applications
- Developing for the NVIDIA Platform, and NVIDIA Grace
- Q&A

NASA NAS AMS Seminar

NVIDIA Grace Superchips

NASA NAS AMS Seminar

The NVIDIA Grace CPU

The building block of the superchip

High Performance Power Efficient Cores

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

Fast On-Chip Fabric

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

High-Bandwidth Low-Power Memory

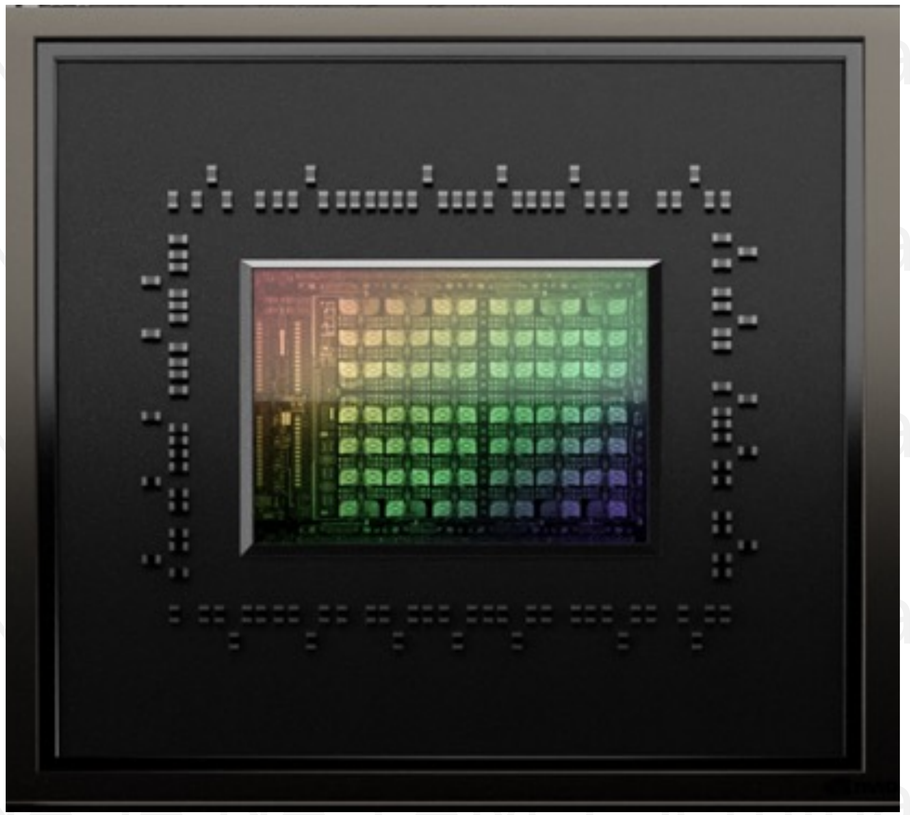
Up to 480 GB of data center enhanced LPDDR5X Memory that delivers up to 500 GB/s of memory bandwidth

Coherent Chip-to-Chip Connections

NVLink-C2C with 900 GB/s bandwidth for coherent connection to CPU or GPU

Industry Leading Performance Per Watt

Up to 2X perf / W over today's leading servers



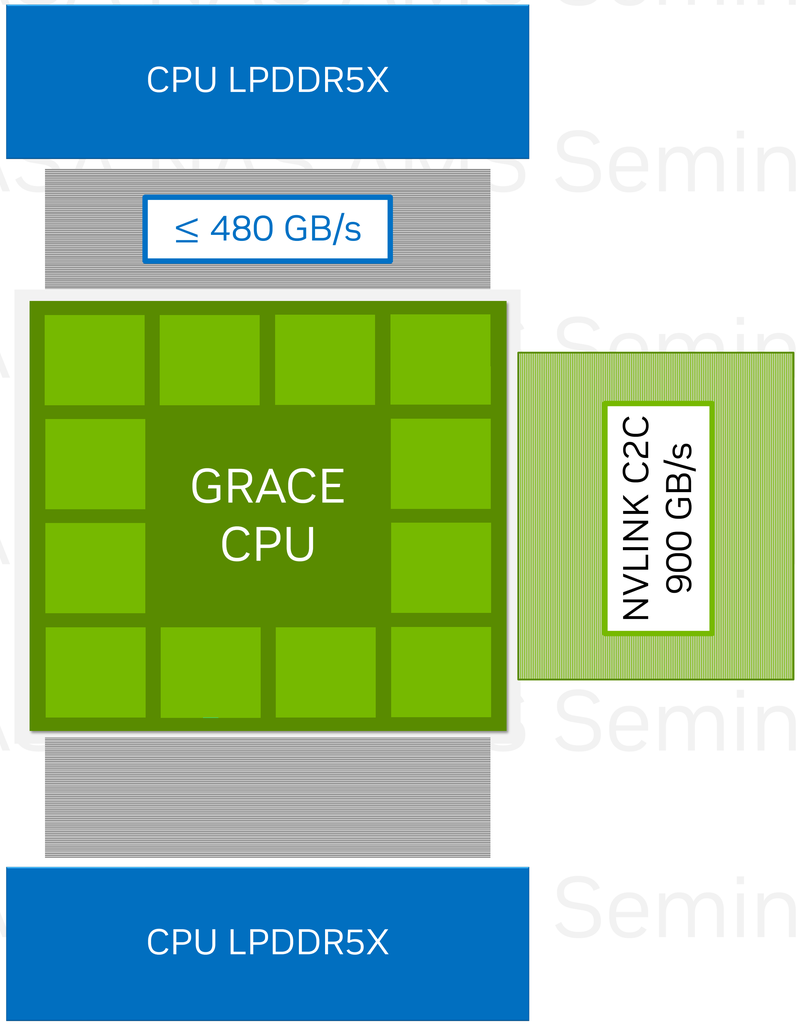
NVIDIA

NASA NAS AMS Seminar

NVLINK-C2C

High Speed Chip to Chip Interconnect

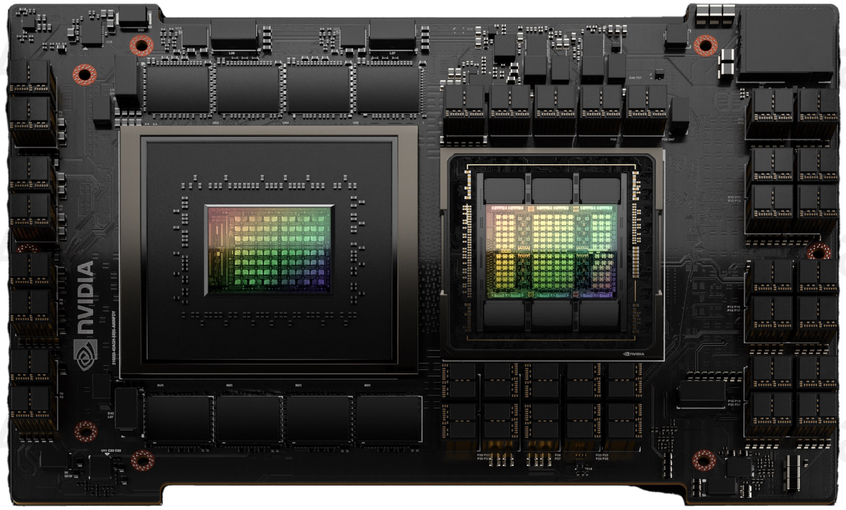
- Grace Hopper and Grace Superchips
- Removes the typical cross-socket bottlenecks
- Up to 900GB/s of raw bidirectional BW
  - Same BW as GPU to GPU NVLINK on Hopper
- Low power interface - 1.3 pJ/bit
  - More than 5x more power efficient than PCIe
- Enables coherency for both Grace and Grace Hopper superchips



One Powerful CPU – Two Superchip Configurations

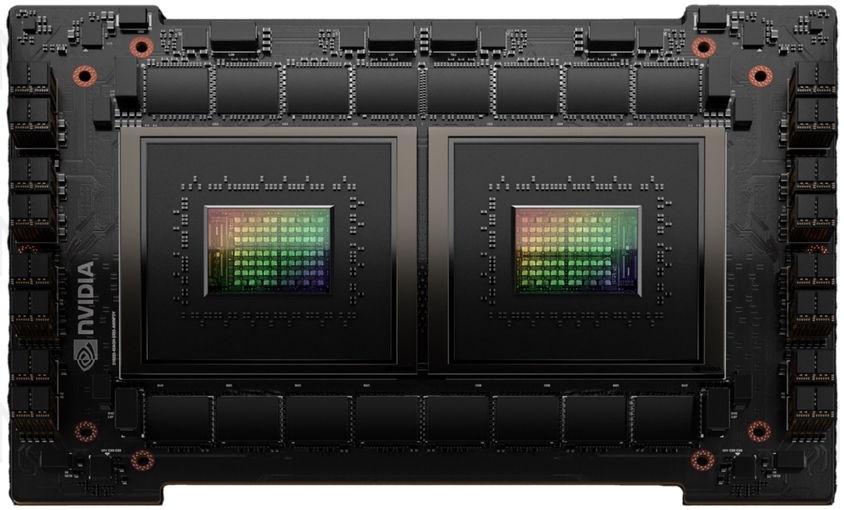
Grace Hopper Superchip (GH200)

More than “Grace + Hopper”

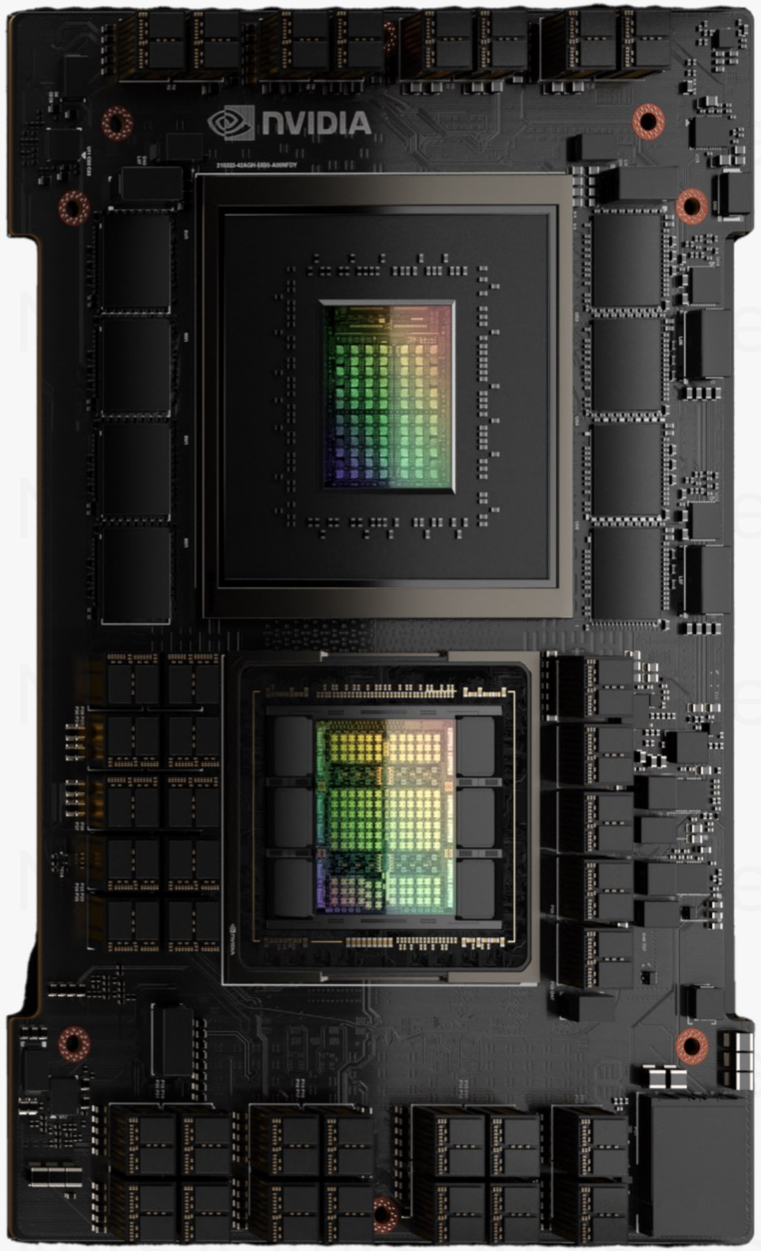


Grace CPU Superchip

More than “2 x Grace”



NASA NAS AMS Seminar

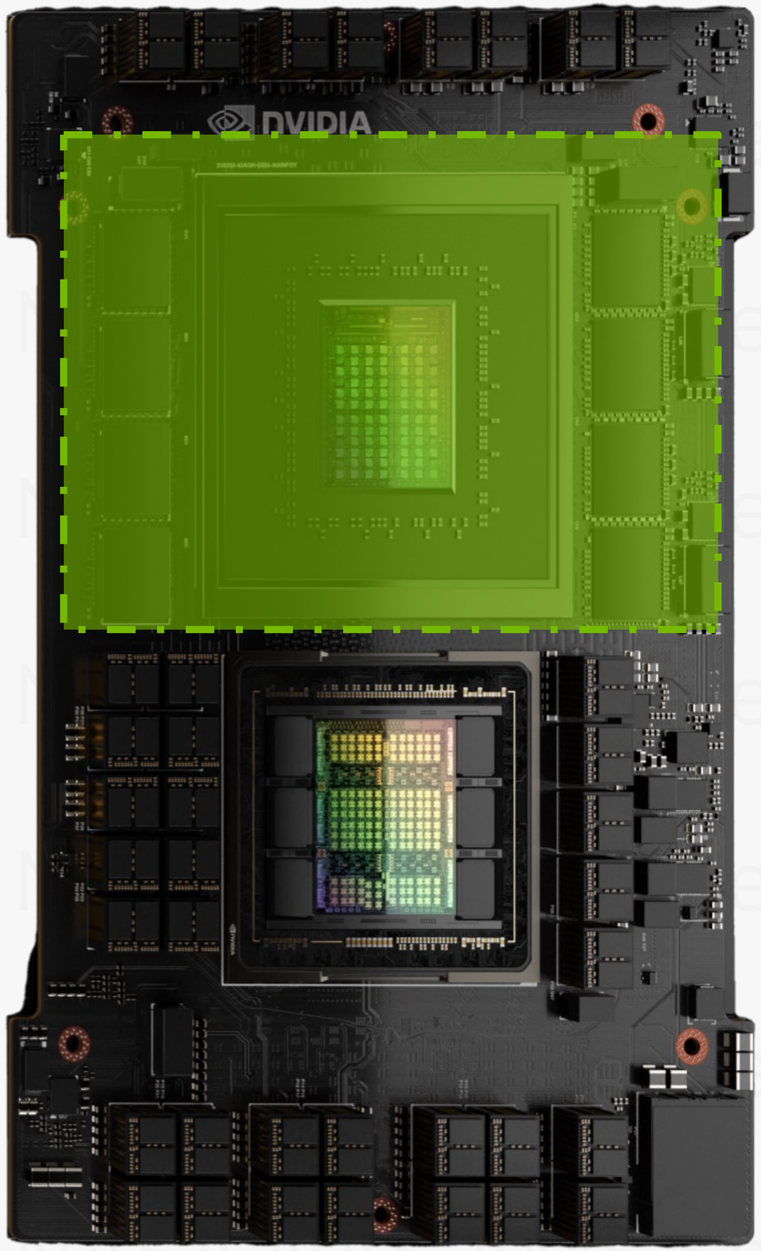


NVIDIA Grace Hopper Superchip

"Super" → more than a "chip"

NVIDIA CPU + NVIDIA GPU w/o compromises

NASA NAS AMS Seminar



NVIDIA Grace Hopper Superchip

"Super" → more than a "chip"

NVIDIA CPU + NVIDIA GPU w/o compromises

- **NVIDIA Grace CPU + LPDDR5 Memory**
  - 72 Arm-v9 Neoverse V2 CPU cores with SVE2.
    - →Efficiency: 62pJ/DFMA (x86: ~99); 1.6x more efficient
    - →Performance: 3.6 FP64 TFLOP/s
  - Memory:
    - → High capacity: ≤ **480 GB** LPDDR5X (5pJ/bit vs 36 DDR)
    - → High bandwidth: ≤ **500 GB/s**
    - → Low latency: less than competitors at peak bandwidth

NASA NAS AMS Seminar



NVIDIA Grace Hopper Superchip

"Super" → more than a "chip"

NVIDIA CPU + NVIDIA GPU w/o compromises

- **NVIDIA Grace CPU + LPDDR5 Memory**
  - 72 Arm-v9 Neoverse V2 CPU cores with SVE2.
    - →Efficiency: 62pJ/DFMA (x86: ~99); 1.6x more efficient
    - →Performance: 3.6 FP64 TFLOP/s
  - Memory:
    - → High capacity: ≤ **480 GB** LPDDR5X (5pJ/bit vs 36 DDR)
    - → High bandwidth: ≤ **500 GB/s**
    - → Low latency: less than competitors at peak bandwidth
- **NVIDIA Hopper GPU**
  - →High performance: 60 FP64 TC TFLOP/s
  - Memory:
    - → High capacity: 96 GB HBM3
    - → Extreme bandwidth ≤ 4000 GB/s
  - → Threads are threads (not SIMD lanes)

NASA NAS AMS Seminar



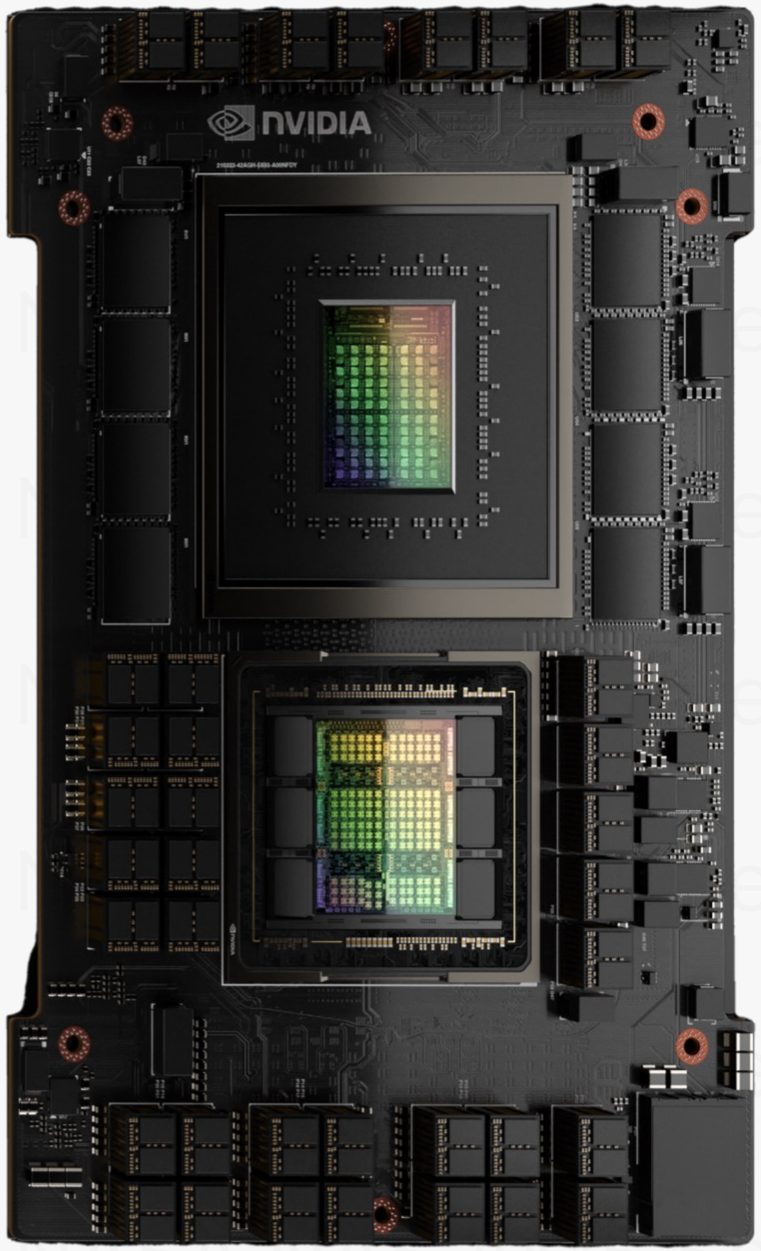
NVIDIA Grace Hopper Superchip

Soul is the new **NVLink-C2C** CPU ←→ GPU interconnect

- **Memory coherency**: ease of use
  - →*All* threads – GPU and CPU – access system memory: C++ `new`, `malloc`, `mmap`'ed files, atomics, …
  - → Fast automatic page migrations HBM3←→LPDDR5X.
  - → Threads cache peer memory → Less migrations.
- **High-bandwidth**: 900 GB/s (same as peer NVLink 4)
  - → GPU reads or writes local/peer LPDDR5X at ~peak BW
- **Low-latency**: GPU→HBM latency
  - →GPU reads or writes LPDDR5X at ~HBM3 latency

For all threads in the system
memory is memory
expected behavior + latency + bandwidth.

NASA NAS AMS Seminar



NVIDIA Grace Hopper Superchip

Made 💚 for any programming model

Portable ISO C++, ISO Fortran, Python

- **Simplifies parallelization:** less SW changes

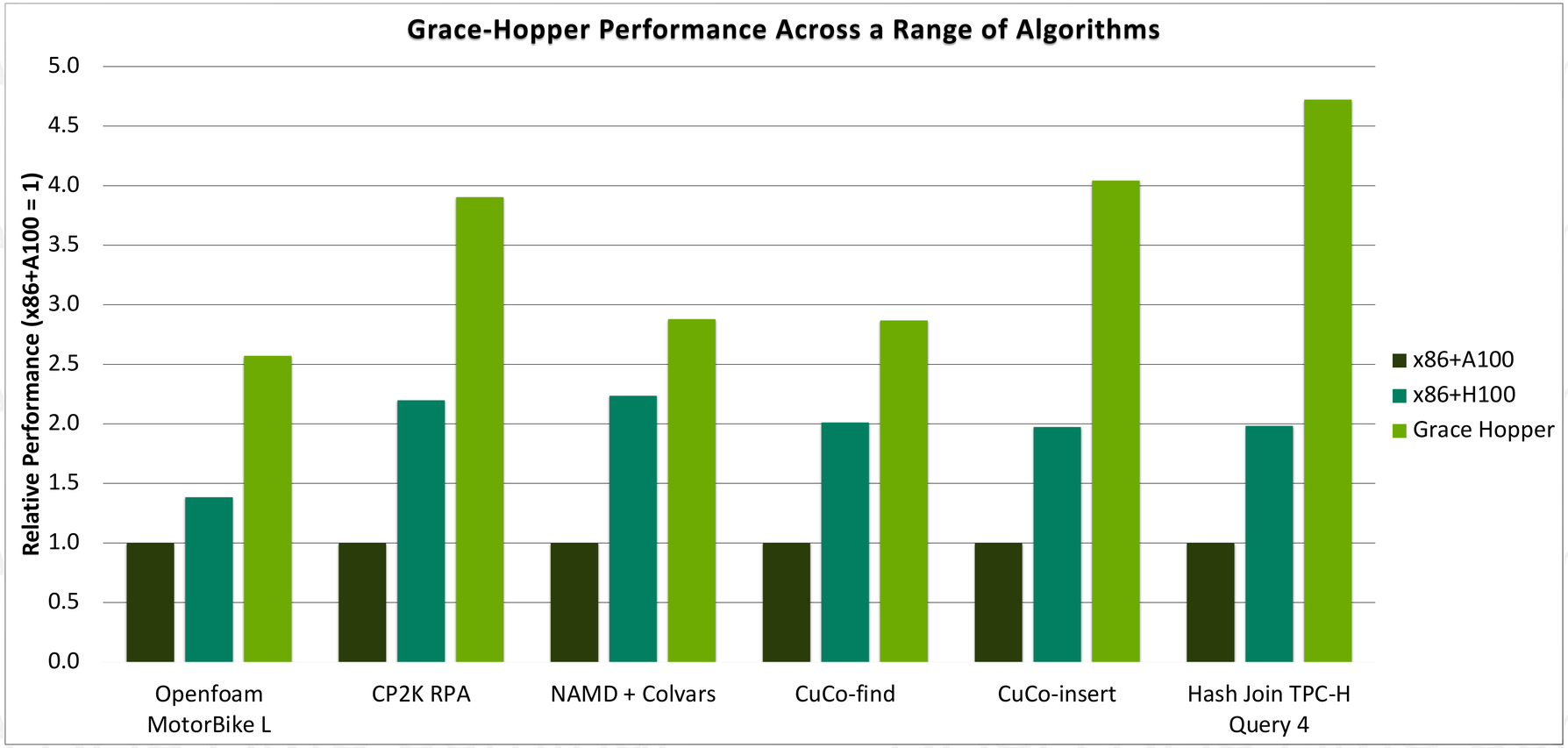
  → **ISO C++, ISO Fortran, Python**: Threads are “threads” (!SIMD), memory consistency, automatic memory management, …

  → **Applications**: complex code stays on CPU, infrequently used memory stays on DDR, large GPU memory capacity (600 GB).

- **Easiest system to:**
  → teach & learn heterogeneous programming
  → parallelize applications
  → use the right HW for each algorithm

NASA NAS AMS Seminar

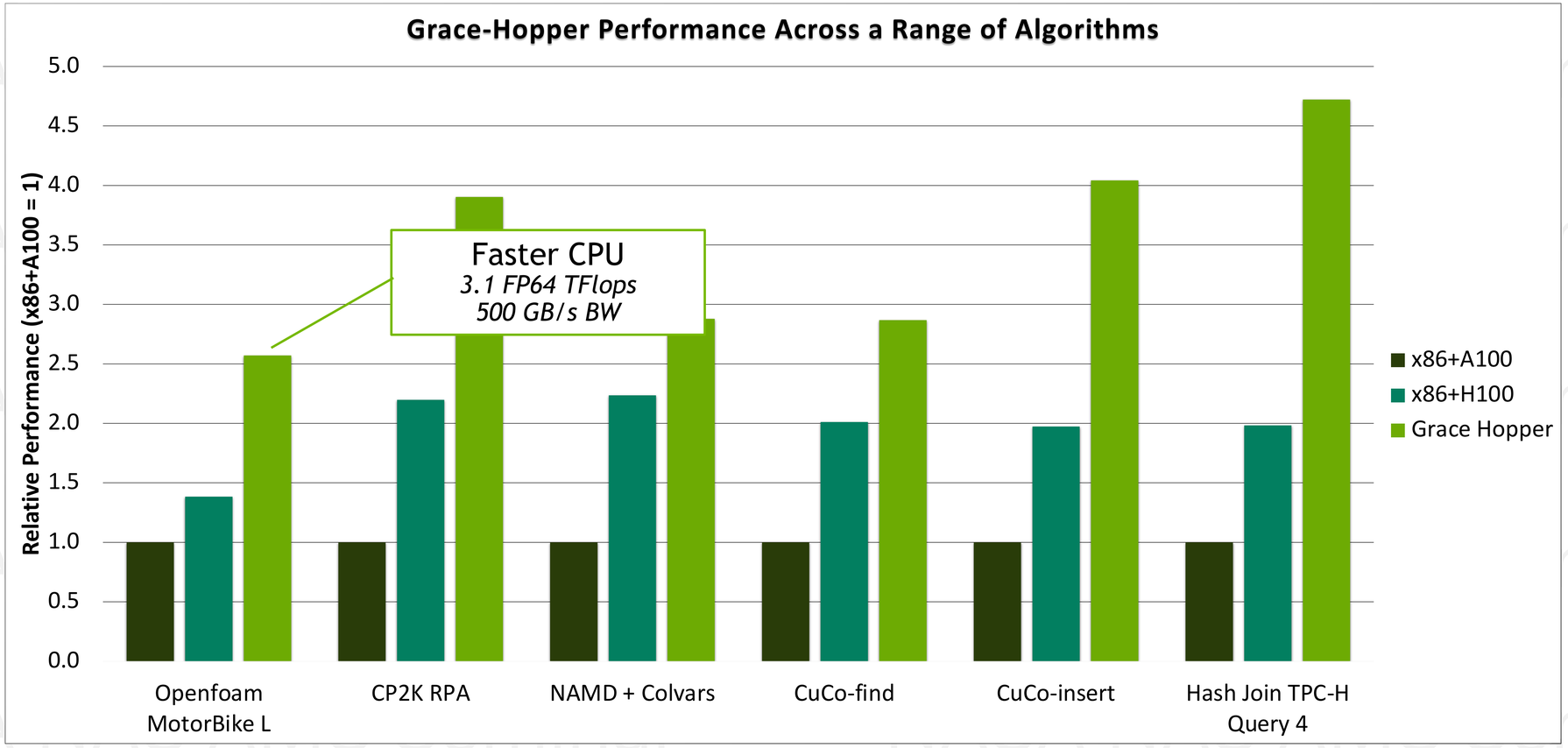
Grace-Hopper Superchip Workload Performance



NASA NAS AMS Seminar

NASA NAS AMS Seminar

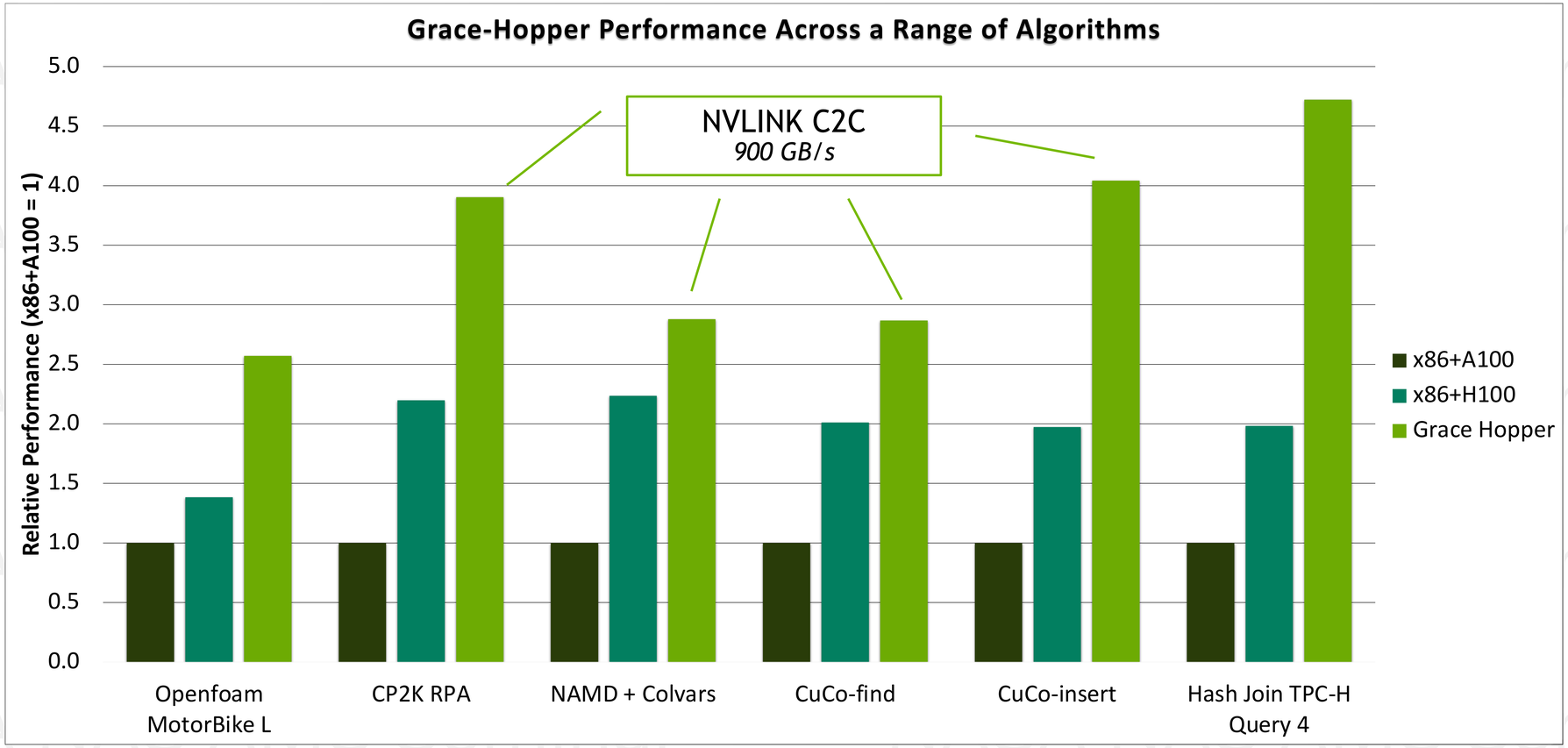
Grace-Hopper Superchip Workload Performance



NVIDIA.

NASA NAS AMS Seminar

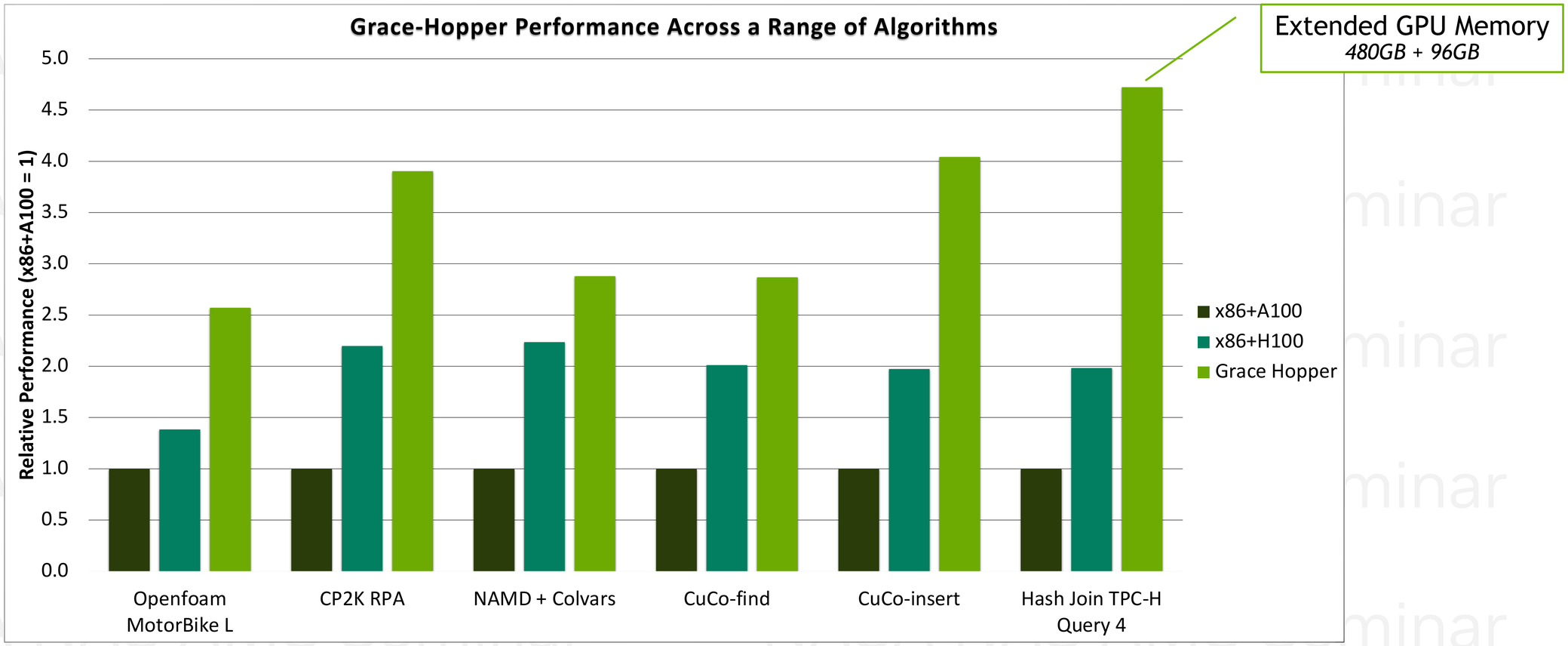
Grace-Hopper Superchip Workload Performance



NVIDIA.

NASA NAS AMS Seminar

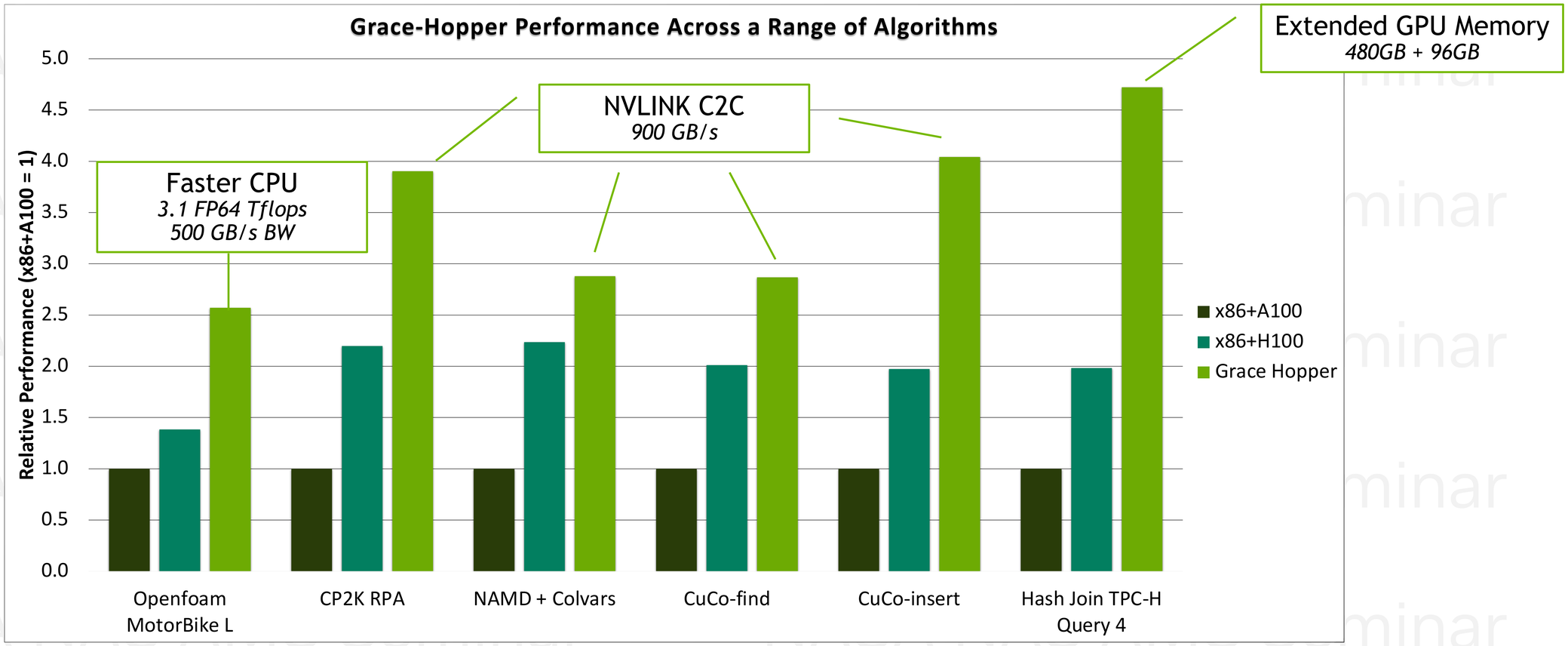
Grace-Hopper Superchip Workload Performance



NVIDIA.

NASA NAS AMS Seminar

Grace-Hopper Superchip Workload Performance

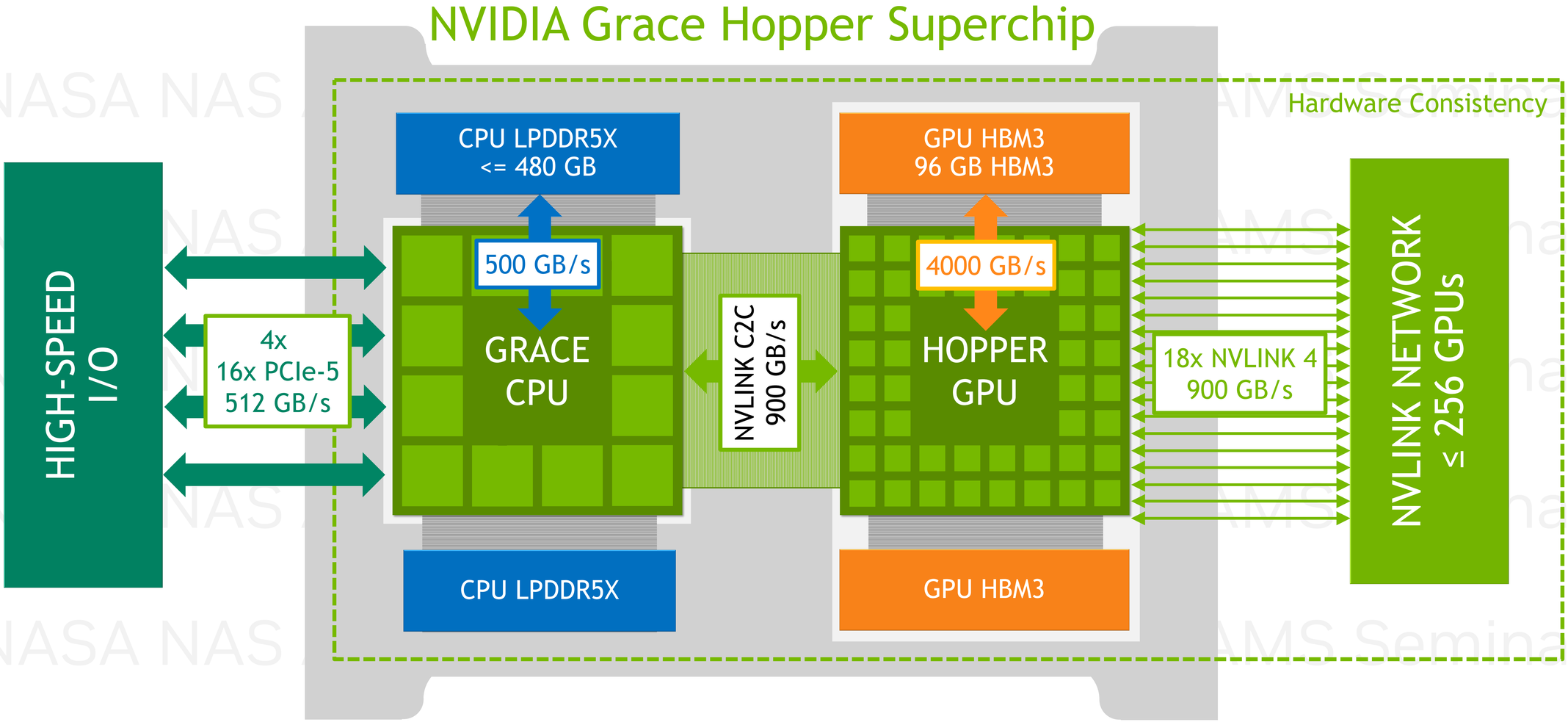


NVIDIA

NASA NAS AMS Seminar

Grace Hopper Superchip I/O

PCIe-5 and NVLINK 4 enable versatile scale-out platform design



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

NVIDIA

NASA NAS AMS Seminar NASA NAS AMS Seminar

Grace Hopper Superchip I/O

PCIe-5 and NVLINK 4 enable versatile scale-out platform design

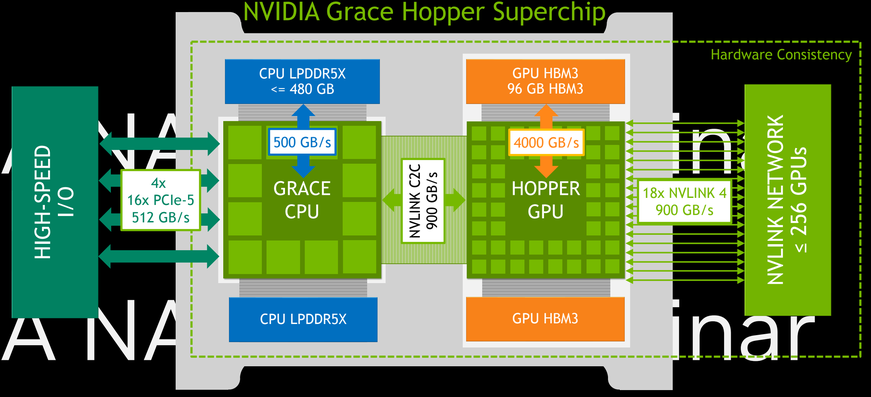
NASA NAS AMS Seminar NASA NAS AMS Seminar

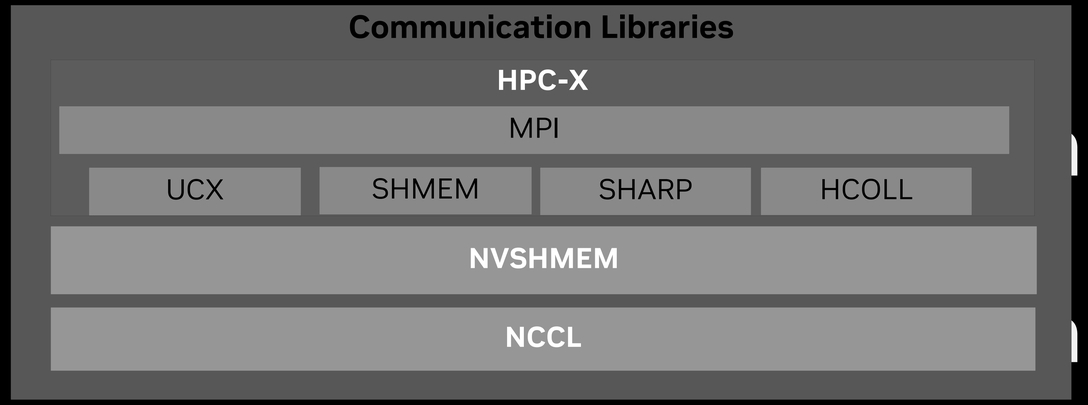
DGX-GH200 - 256 Grace Hopper Superchips with MNNVL interconnect

1EFLOPS AI Performance | **144TB** unified fast memory
36 L2 NVLink switches | **900 GB/s** GPU-to-GPU bandwidth | **128 TB/s** bisection bandwidth



NASA NAS AMS Seminar NASA NAS AMS Seminar





NASA NAS AMS Seminar NASA NAS AMS Seminar

Grace CPU Superchip

Grace CPU Superchip	
Architecture	Arm Neoverse V2 Armv9.0-A, **SVE2 4x128b SIMD**
Cores / Speed	144 cores, up to 3.2GHz
Memory	LPDDR5x soldered down, 1TB/s BW Up to 480GB per superchip
Cache	L1: 64KB i$ + 64KB d$ per core L2: 1MB per core L3: 234MB per superchip
Power	**500W including LPDDR5x memory**
Interfaces	Up to 8x PCIe Gen5 x16 HS interface
Process Node	TSMC 4N
Availability	Q3 2023



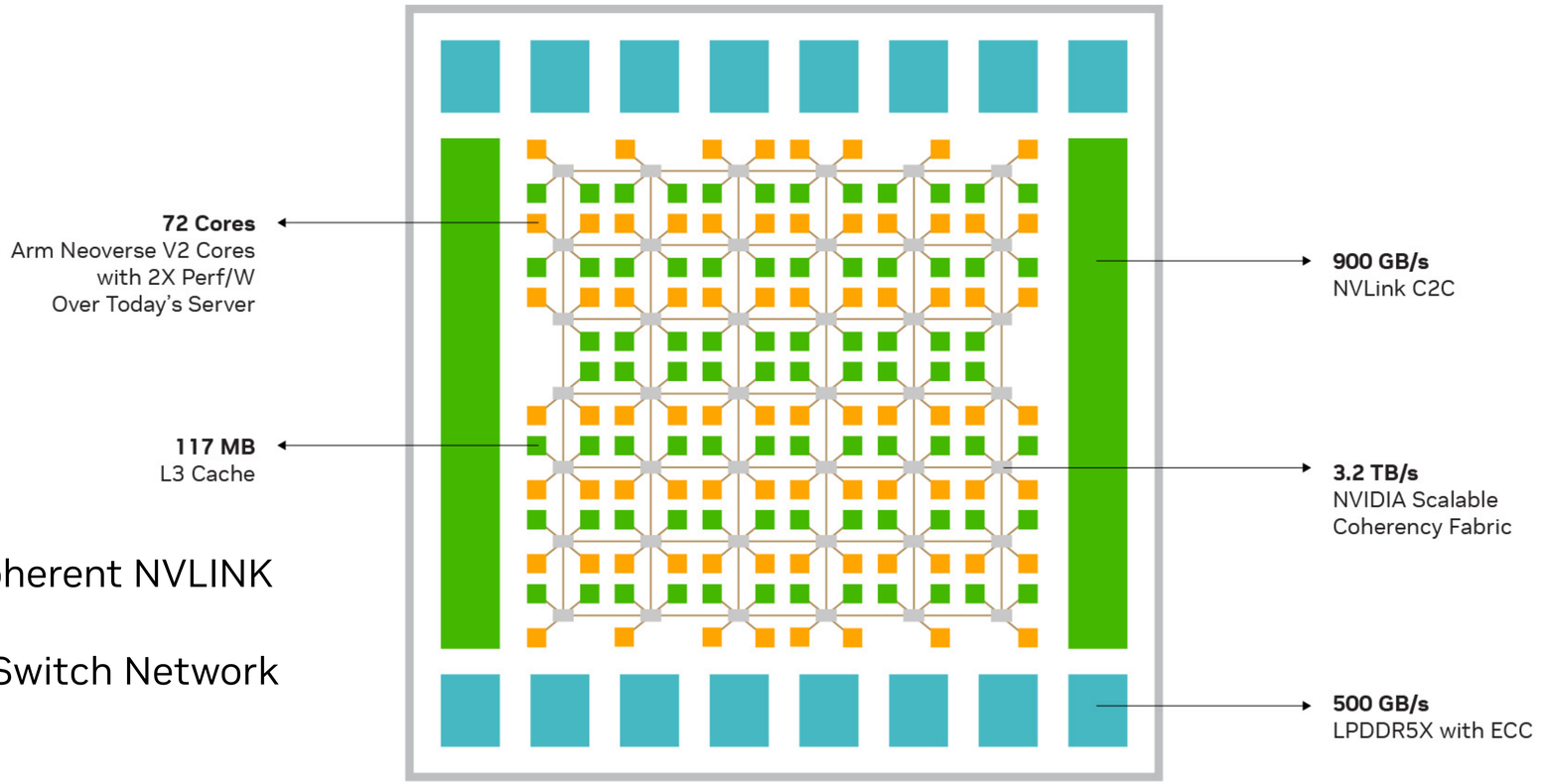
NASA NAS AMS Seminar

NASA NAS AMS Seminar

Grace is a Compute and Data Movement Architecture

NVIDIA Scalable Coherency Fabric (SCF) and distributed cache design

- **Single Die:** More efficient use of power
- 3,225.6 GB/s Bi-section BW
- 117MB of L3 cache
- Scalable to 72+ cores per die
- Local caching of remote die memory
- Supports up to 4-die coherency over Coherent NVLINK
- Background data movement via Cache Switch Network



Example possible fabric topology for illustrative purposes

NASA NAS AMS Seminar

Grace Simplifies System Design and Workload Optimization

A high-performance server on a single superchip package

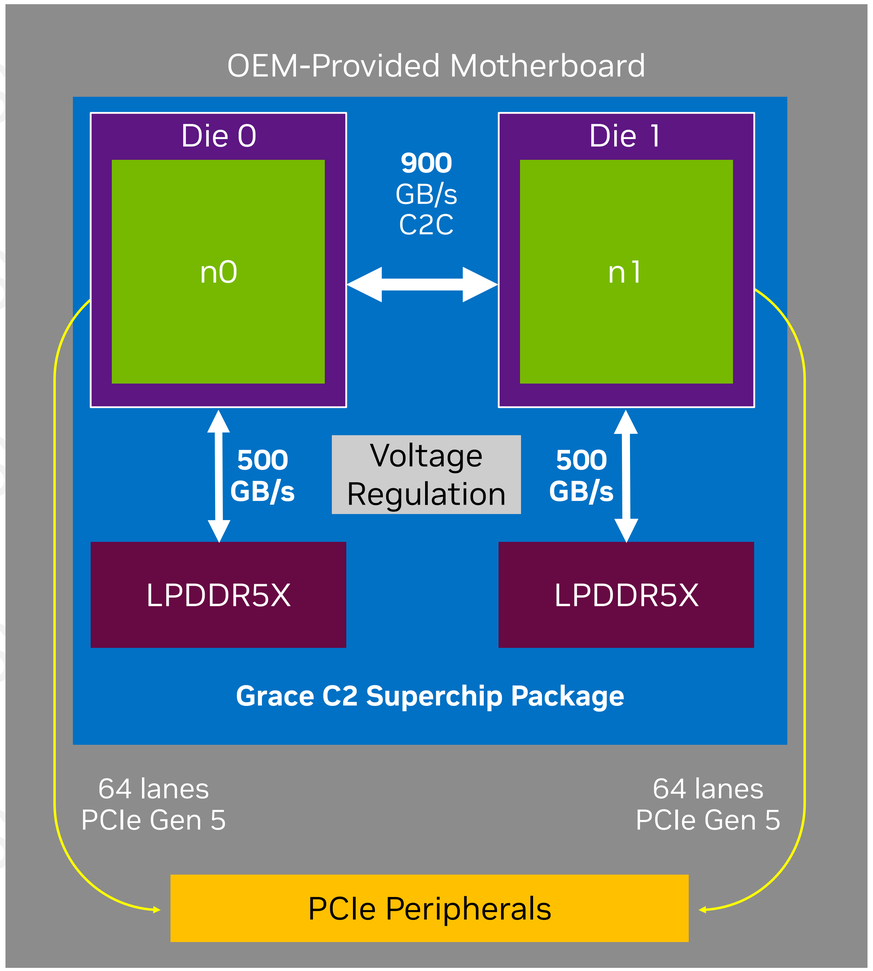
2 NUMA Nodes

2 Compute Dies

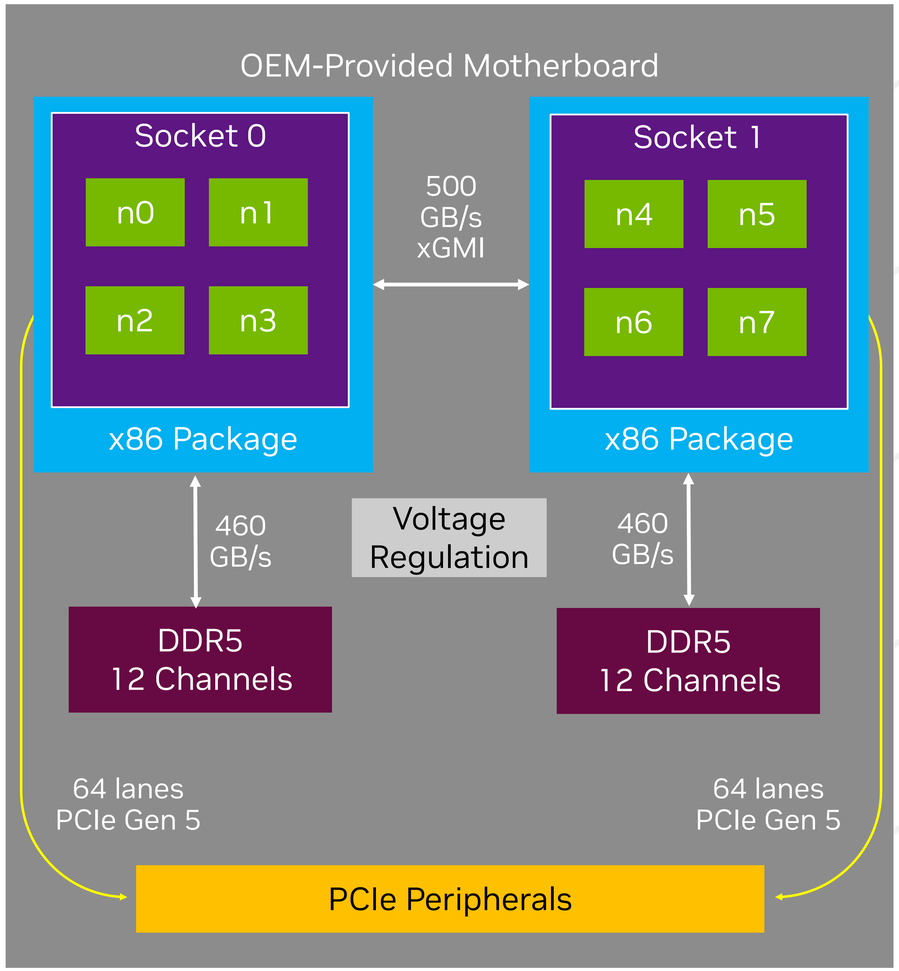
500 Watts (CPU + MEM)

900 GB/s worst-case n to n

Grace Server
Single Grace C2 Superchip



Conventional 2-Socket Server
Dual-socket x86, NPS=4



8 NUMA Nodes

24 Compute Chiplets

900+ Watts (CPU + MEM)

500 GB/s worst-case n to n

NVIDIA

NASA NAS AMS Seminar

Transforming HPC and AI Around the World

Argonne NATIONAL LABORATORY

BERKELEY LAB
Lawrence Berkeley National Laboratory

CINECA

Los Alamos NATIONAL LABORATORY

FLATIRON INSTITUTE
Division of Simons Foundation

LINKÖPING UNIVERSITY

JÜLICH Forschungszentrum

MAX PLANCK GESELLSCHAFT

NCSA

NREL
NATIONAL RENEWABLE ENERGY LABORATORY

TACC
TEXAS ADVANCED COMPUTING CENTER

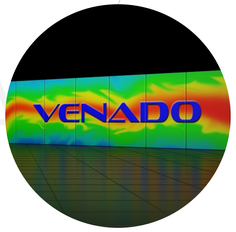
Tokyo Tech

UNIVERSITY OF CAMBRIDGE

UF UNIVERSITY of FLORIDA

筑波大学
University of Tsukuba





LANL (Venado)
Grace Hopper
10 EFLOPS AI Perf



Univ.of Bristol (Isambard 3)
Grace CPU
2 PFLOPS HPC Perf



BSC (MareNostrum 5)
Grace CPU
2 PFLOPS HPC Perf



CSCS (ALPS)
Grace Hopper
20 EFLOPS AI Perf



KAUST (Shaheen-III)
Grace Hopper
7 EFLOPS AI Perf

NVIDIA

Announcing University of Bristol's *Isambard 3*

Powered by NVIDIA's Grace CPU Superchip

- Over 55,000 Arm Neoverse V2 cores
- 2.7 PF of HPC Performance, 270 kW of power
- 6X more performance and energy efficiency vs. predecessor
- Built by HPE
- Enabling breakthroughs in Climate Science, Drug Discovery, and Industrial HPC.



NASA NAS AMS Seminar

nvidia@localhost: ~

Work PC on MERCURY

```
Tasks: 41, 41 thr; 1 running
Avg:  .  % sys:  .  % loLoad average: 34.69 27.31 20.54
Uptime: 3 days, 07:48:21
Mem[||                7.41G/511G]
Swp[                     0K/0K]
  0[        0.0%]  73[        0.0%]
  1[        0.0%]  74[        0.0%]
  2[        0.0%]  75[        0.0%]
  3[        0.0%]  76[        0.0%]
  4[        0.0%]  77[        0.0%]
  5[        0.0%]  78[        0.0%]
  6[        0.0%]  79[        0.0%]
  7[        0.0%]  80[        0.0%]
  8[        0.0%]  81[        0.0%]
  9[        0.0%]  82[        0.0%]
 10[        0.0%]  83[        0.0%]
 11[        0.0%]  84[        0.0%]
 12[        0.0%]  85[        0.0%]
 13[        0.0%]  86[        0.0%]
 14[        0.0%]  87[        0.0%]
 15[        0.0%]  88[        0.0%]
 16[        0.0%]  89[        0.0%]
 17[        0.0%]  90[        0.0%]
 18[        0.0%]  91[        0.0%]
 19[        0.0%]  92[        0.0%]
 20[        0.0%]  93[        0.0%]
 21[        0.0%]  94[        0.0%]
 22[        0.0%]  95[        0.0%]
 23[        0.0%]  96[        0.0%]
F1Help  F2Setup  F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Q
```

```
Running the test with following options:
Number of threads: 144
Initializing random number generator from current time

Prime numbers limit: 10000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 554065.75

General statistics:
    total time:                          10.0017s
    total number of events:              5542682

Latency (ms):
         min:                                  0.26
         avg:                                  0.26
         max:                                 16.27
         95th percentile:                      0.26
         sum:                            1438171.20

Threads fairness:
    events (avg/stddev):           38490.8472/127.58
    execution time (avg/stddev):   9.9873/0.00

grace$
```

```
Avg:  .  % sys:  .  % low:  .  % vir:  .  %
  0[        0.0%]  56[        0.0%]
  1[        0.0%]  57[        0.0%]
  2[        0.0%]  58[        0.0%]
  3[        0.0%]  59[        0.0%]
  4[        0.0%]  60[        0.0%]
  5[        0.0%]  61[        0.0%]
  6[        0.0%]  62[        0.0%]
  7[        0.0%]  63[        0.0%]
  8[        0.0%]  64[        0.0%]
  9[        0.0%]  65[        0.0%]
 10[        0.0%]  66[        0.0%]
 11[        0.0%]  67[        0.0%]
 12[        0.0%]  68[        0.0%]
 13[        0.0%]  69[        0.0%]
 14[        0.0%]  70[        0.0%]
 15[        0.0%]  71[        0.0%]
 16[        0.0%]  72[        0.0%]
 17[        0.0%]  73[        0.0%]
 18[        0.0%]  74[        0.0%]
 19[        0.0%]  75[        0.0%]
 20[        0.0%]  76[        0.0%]
 21[        0.0%]  77[        0.0%]
 22[        0.0%]  78[        0.0%]
 23[        0.0%]  79[        0.0%]
F1Help  F2Setup  F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Q
```

```
Number of threads: 224
Initializing random number generator from current time

Prime numbers limit: 10000

Initializing worker threads...

Threads started!

CPU speed:
    events per second: 285429.18

General statistics:
    total time:                          10.0020s
    total number of events:              2855608

Latency (ms):
         min:                                  0.30
         avg:                                  0.78
         max:                                 76.91
         95th percentile:                      1.12
         sum:                            2233501.79

Threads fairness:
    events (avg/stddev):           12748.2500/352.75
    execution time (avg/stddev):   9.9710/0.03

x86$
```

```
[0] 0:ssh*                                              "localhost" 04:30 22-Apr-23
```

10.137.175.52:8081

Not secure | 10.137.175.52...

Sign in

x86 Server Power

NVIDIA Grace Server Power



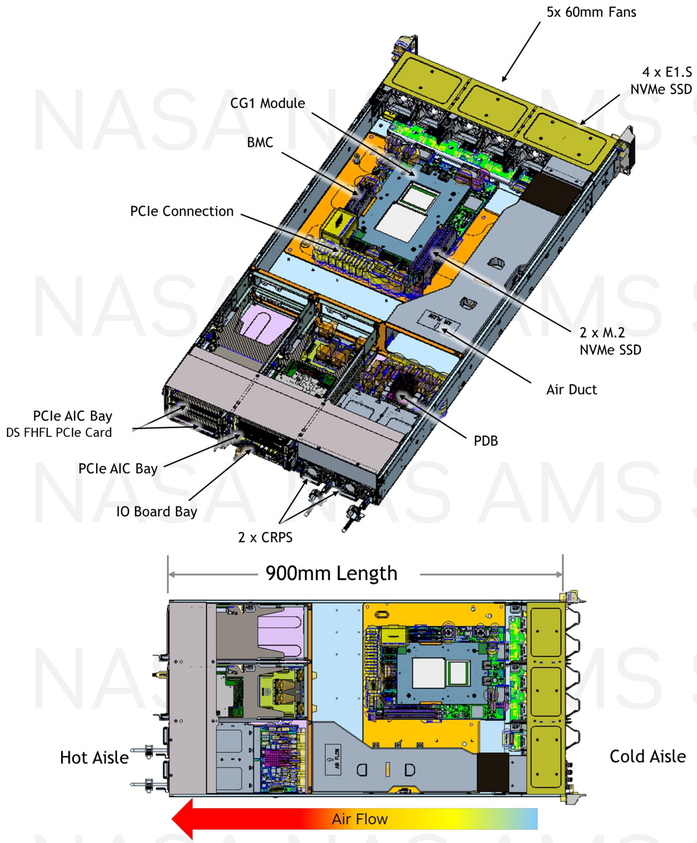
Search

NVIDIA

NASA NAS AMS Seminar

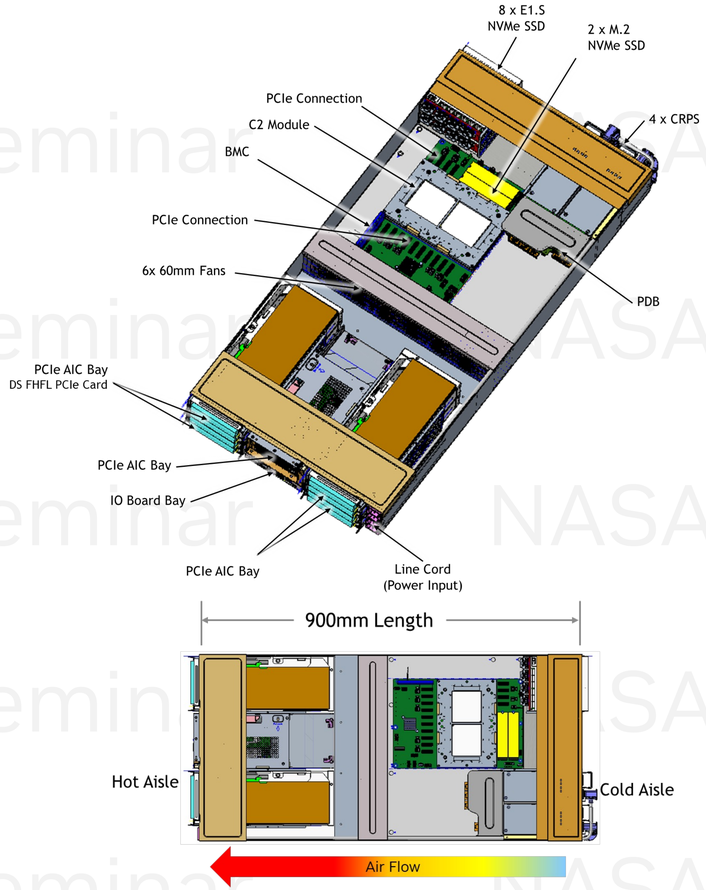
Grace Superchip Evaluation Systems

Contact your NVIDIA sales representative for ordering information



Grace Hopper Evaluation System

1x Grace-Hopper Superchip in a 2U server



Grace CPU Superchip Evaluation System

1x Grace CPU Superchip in a 2U server

- 1x Superchip (G-H or G-G)
- 2U 900mm chassis
- 2x 2000W power supply
- CX-7 InfiniBand
- PCIe AIC Bay, DS FHFL
- Air cooled

- **Note**: Orders will be handled directly via NVIDIA's partners. Exact delivery dates, price, technical details, etc. will come from them.

NASA NAS AMS Seminar

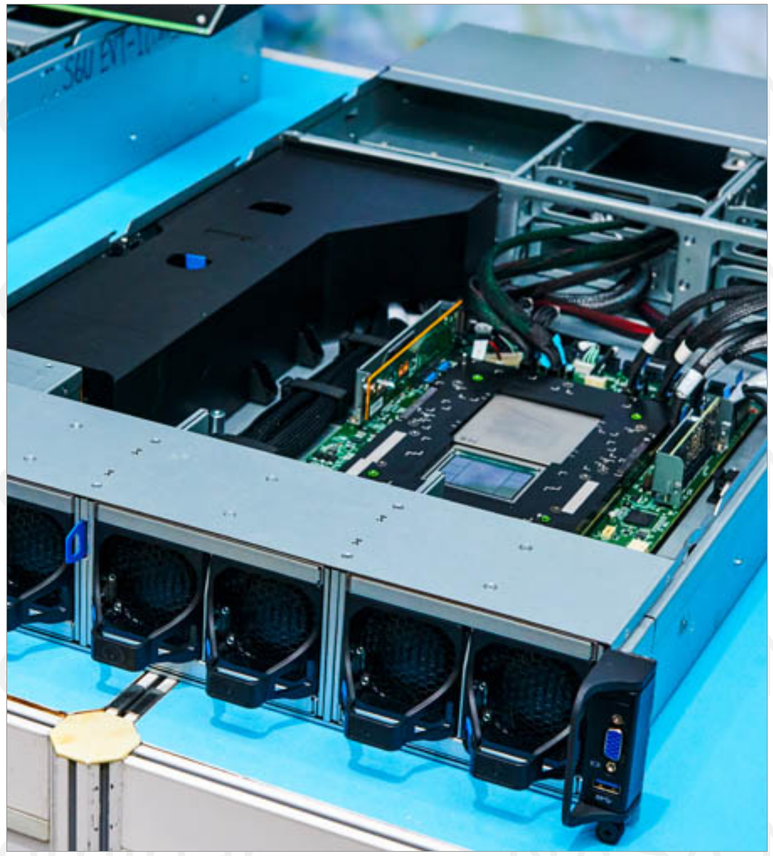
STH: *NVIDIA Grace Hopper MGX Platform QCT QuantaGrid S74G-2U*

https://www.servethehome.com/nvidia-grace-hopper-mgx-platform-qct-quantagrid-s74g-2u-at-computex-2023/



QCT QuantaGrid S74G 2U With NVIDIA Grace Hopper Installed

© 2009-2023 Loyolan Ventures, LLC



QCT QuantaGrid S74G 2U Front Lid Off Angle

© 2009-2023 Loyolan Ventures, LLC



QCT QuantaGrid S74G 2U Rear PCIe

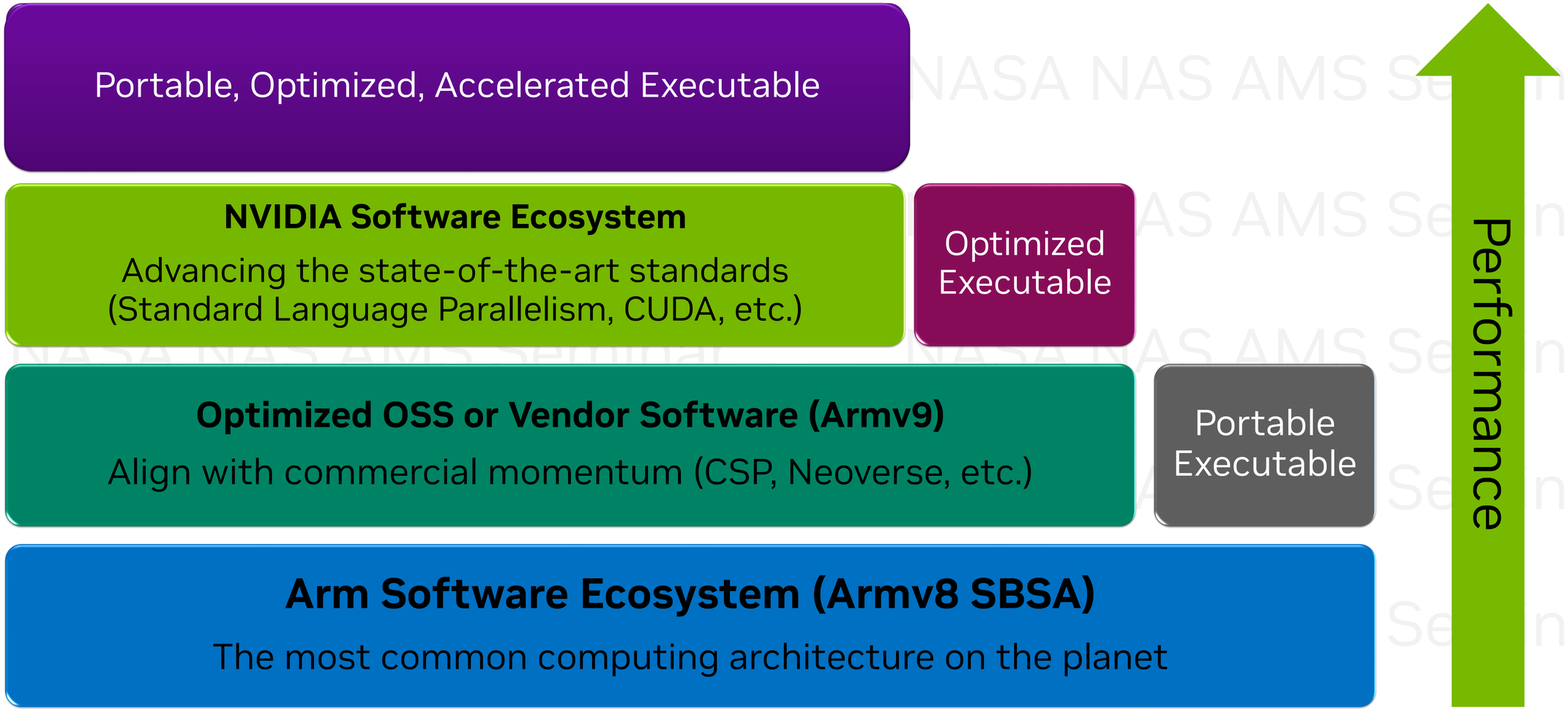
© 2009-2023 Loyolan Ventures, LLC

NASA NAS AMS Seminar

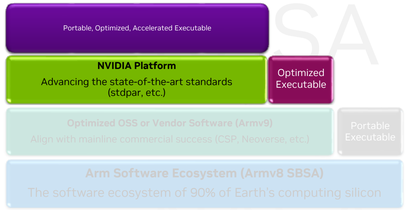
NVIDIA Grace Software Ecosystem and Applications

Grace Software Ecosystem is Built on Standards

The NVIDIA platform builds on optimized software from the broad Arm software ecosystem

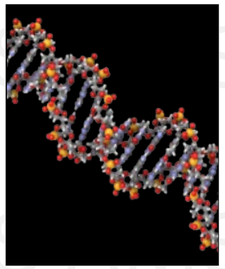


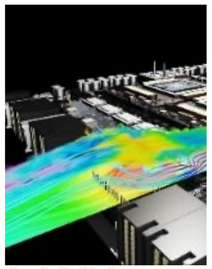
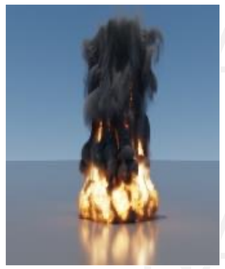
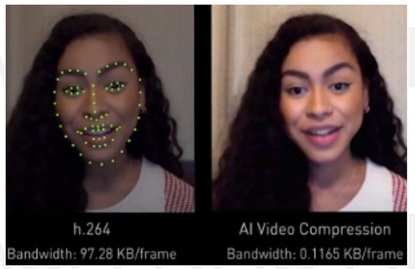
NASA NAS AMS Seminar



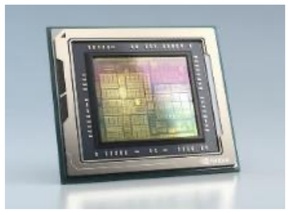
The Expanding NVIDIA Ecosystem

+3M Developers | 7,600 Startups

Omniverse	Isaac	DRIVE	Aerial	Clara	GFN	Metropolis	Riva	Maxine	Merlin
AI	HPC	RTX	RAPIDS	Triton	TRT	Rivermax	Nsight	JetPack	
CUDA					DOCA				








500+
GitHub Projects

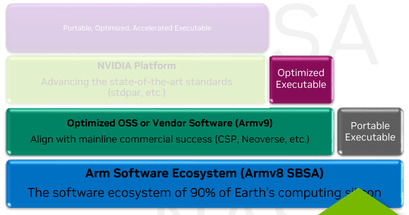
60+
GPU Accelerated SDKs

50+
Developer Libraries

20+
Developer Tools and Utilities

NASA NAS AMS Seminar

NASA NAS AMS Seminar



The Arm Software Ecosystem

Developed by NVIDIA, the OSS community, and other Arm partners

NVIDIA Software
Frameworks, Libraries, SDKs, Toolkits, etc.

Applications and Frameworks
Deep Learning | HPC | General Compute

Developer Tools
Open Source and Commercial

Communication Libraries
Open Source and Commercial

Compilers
Open Source and Commercial

Libraries
Open Source and Commercial

Filesystems
All major parallel filesystems

Admin Tools & Packaging

Job Schedulers
Open Source and Commercial

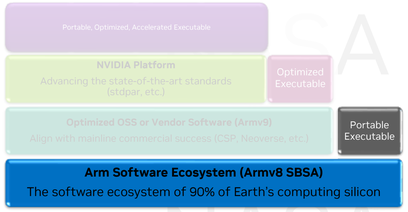
Cluster Management
NVIDIA Bright, Open Source, and Commercial

Containers & Orchestration
NVIDIA Container Toolkit and All Major Frameworks and Tools

OS
All Major Linux Distributions

Arm Server Ready Platform
Standard Firmware and RAS

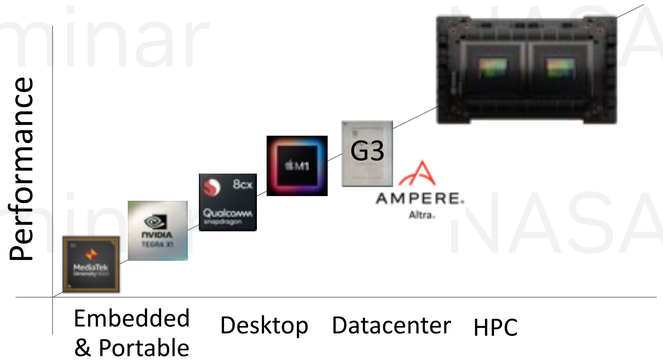
NVIDIA



Performance Backed by Arm Standards

Best-in-class performance with full compatibility





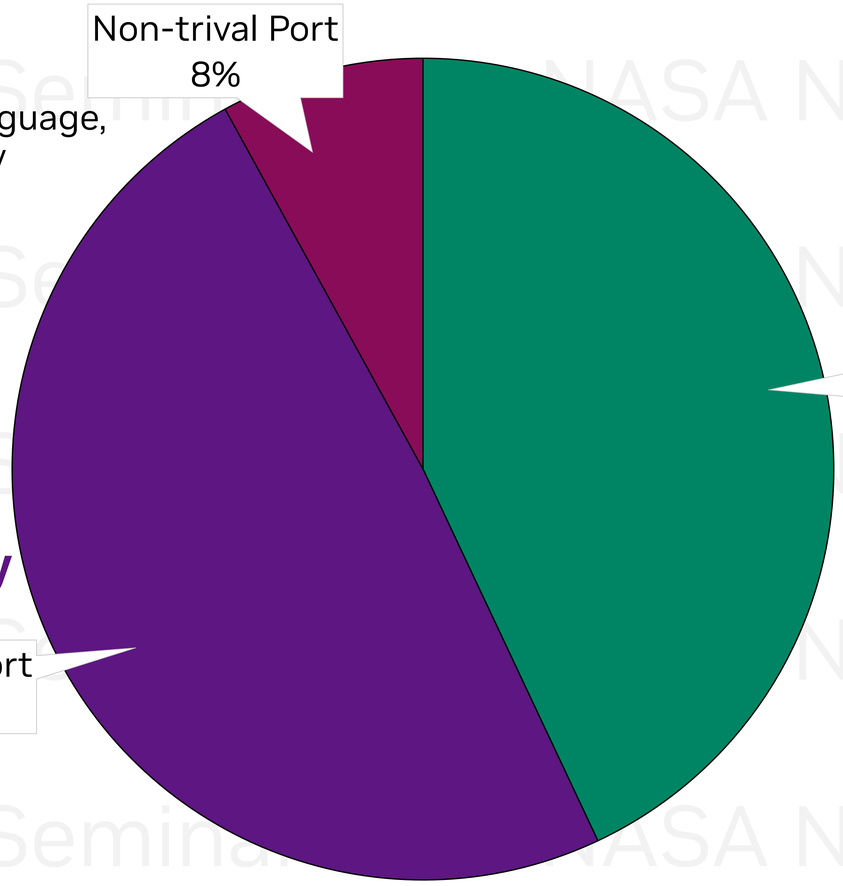
KVM docker Red Hat SUSE kubernetes

Standardized Arm Core	Leading Performance	Rich Ecosystem with Commercial Momentum
A rich technology that exists everywhere CPUs are found	Optimizations for Grace are an optimization for the entire ARM ecosystem, maximizing customer and partner ROI. Grace's simple design means less tuning	The same containers, operating systems, and application binaries that run on Graviton, Altra, etc. run on Grace, but *faster*

NASA NAS AMS Seminar

Workloads of a National Computing Center

Annual core hours



Job done!
Found on Arm at another HPC center

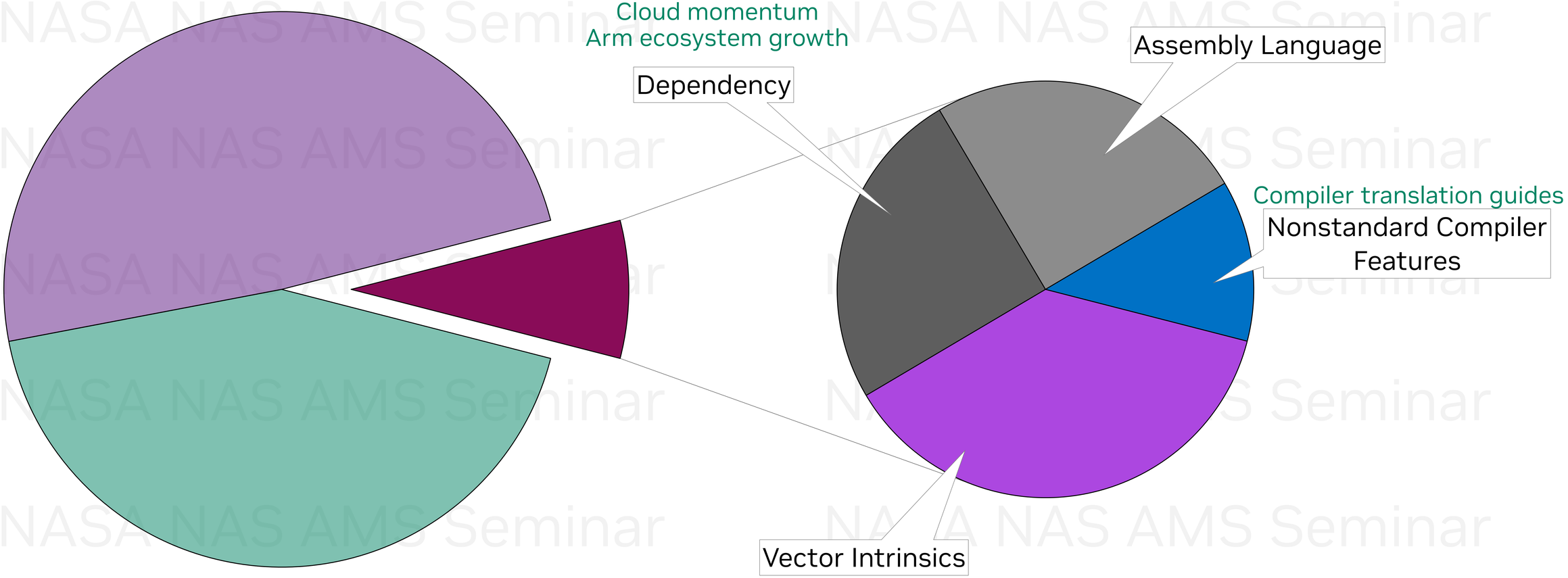
Straightforward, easy work < 1 day
Recompile and reconfigure runtime parameters

NASA NAS AMS Seminar

NVIDIA CONFIDENTIAL. DO NOT DISTRIBUTE.

Many Non-Trivial Cases Really Are Trivial

Vector intrinsics, dependencies, and nonstandard features are easily ported

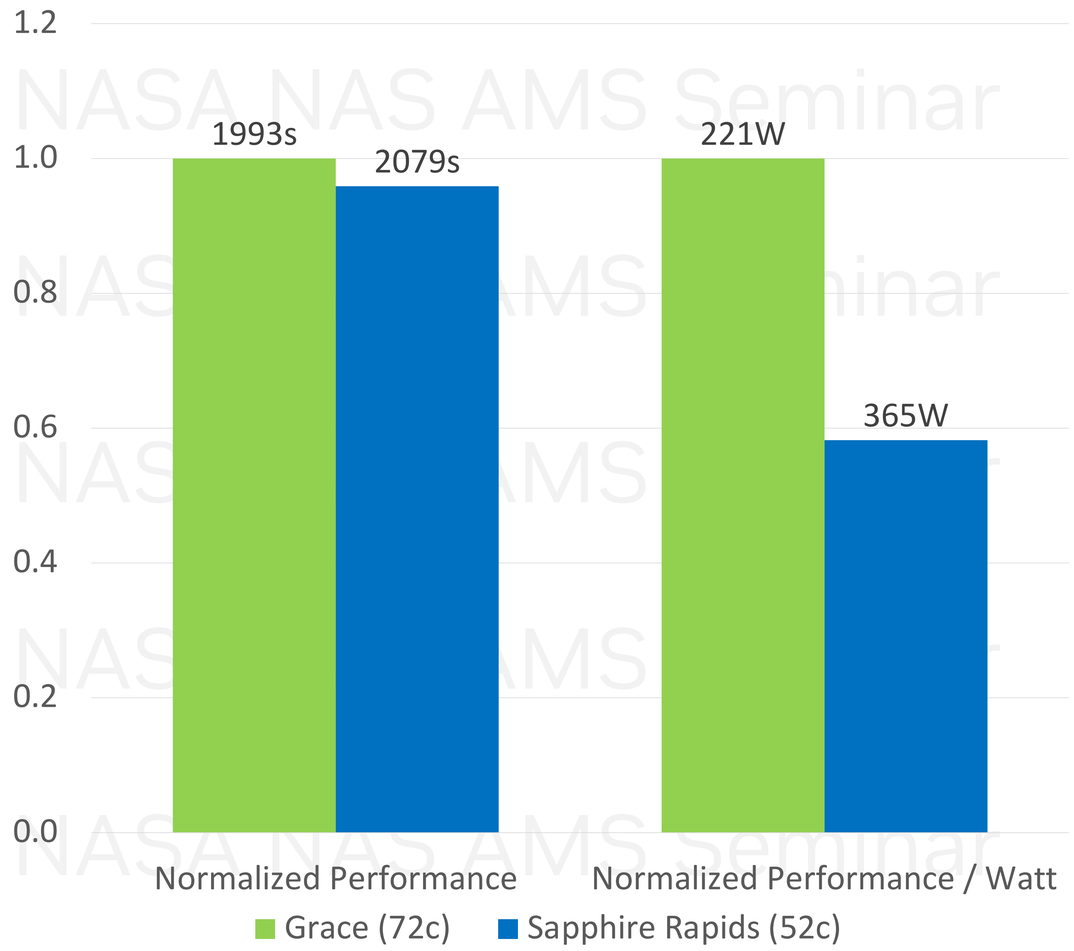


NASA NAS AMS Seminar

NVIDIA CONFIDENTIAL. DO NOT DISTRIBUTE.

“Non-Trivial” Example: BWA-MEM2

You’re not porting to Arm. You’re porting *away* from vendor-specific tools and libraries.



Scope of “porting” work, no optimization done:

~3 hours of developer time to investigate and add:
#include: AVX -> Vendor Nonspecific SIMD Wrapper
- https://github.com/simd-everywhere/simde

Tools:
- Compilers: Clang 16 (NVIDIA)
- Compile options: GCC 12 and SIMD-E

Comparison:
- Precompiled binaries on x86
- HG002 dataset from Illumina paired-end sequencers
- Complete human genome at 30x coverage

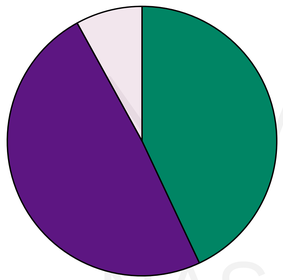
Run configuration:

Similar configuration overhead to moving between Intel & AMD
- Grace: jemalloc + transparent huge pages
- Intel: AVX512 intel-optimized version on SPR

NASA NAS AMS Seminar

NVIDIA Grace Works Like Any Other HPC System

Most applications recompile easily and work out of the box



+90%

Reuse
- NGC
- EasyBuild
- Spack

Recompile
- NVIDIA Compiler
- Arm Compiler for Linux
- GCC
- LLVM

Run
- High core count
- Threads-per-process
- Update tests

Optimize
- NVIDIA Nsight
- Arm Forge
- Perf, PAPI, TAU, Score-P, …

Quick Launch

- NGC containerized applications, frameworks, and toolkits
- EasyBuild and Spack can quickly build applications and common dependencies from source

Compilation Tips

- Most compiler flags are the same:
  - Use `-mcpu=native`
  - Don't use `-march` or `–mtune`
  - You may need `–fsigned-char`
- Update your unit tests:
  - Aarch64 floating point is as accurate as all other platforms

NASA NAS AMS Seminar

NASA NAS AMS Seminar

OpenFOAM 2206 Compilation with GCC and OpenMPI via Spack

- Use Spack to install OpenFOAM and all dependencies
- `spack install openfoam%gcc@12.2.0`
- Spack is convenient but it installs many extra dependencies.
- For a more streamlined build with NVIDIA HPC SDK:

```
# Load NVIDIA HPC SDK
module load nvhpc

export BUILD_DIR=$PWD/OpenFOAM
mkdir -p $BUILD_DIR
cd $BUILD_DIR

# Download source
wget https://sourceforge.net/projects/openfoam/files/v2306/OpenFOAM-v2306.tgz
wget https://sourceforge.net/projects/openfoam/files/v2306/ThirdParty-v2306.tgz

# Unpack source
tar xvzf OpenFOAM-v2306.tgz
tar xvzf ThirdParty-v2306.tgz
cd OpenFOAM-v2306

# Initialize build environment with NVIDIA HPC SDK compilers
source etc/bashrc WM_COMPILER=Nvidia

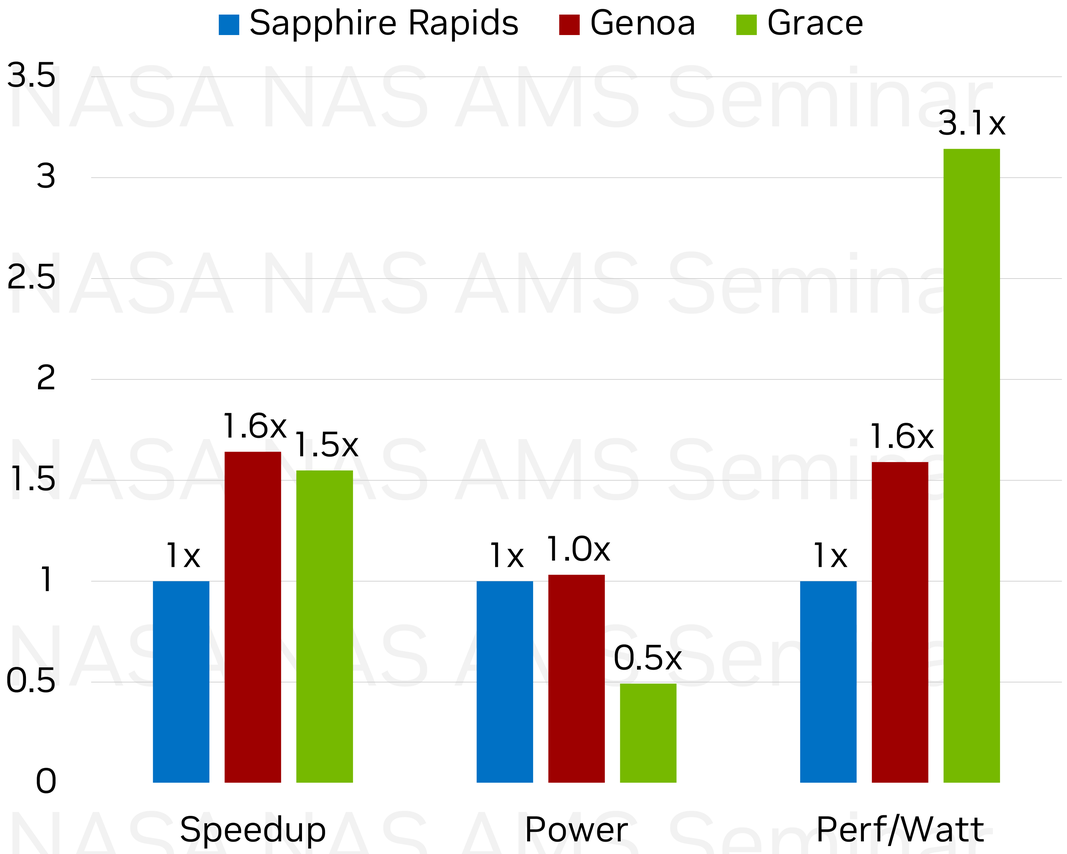
# Compile WRF and all third party packages
./Allwmake -j -s -q -l
```

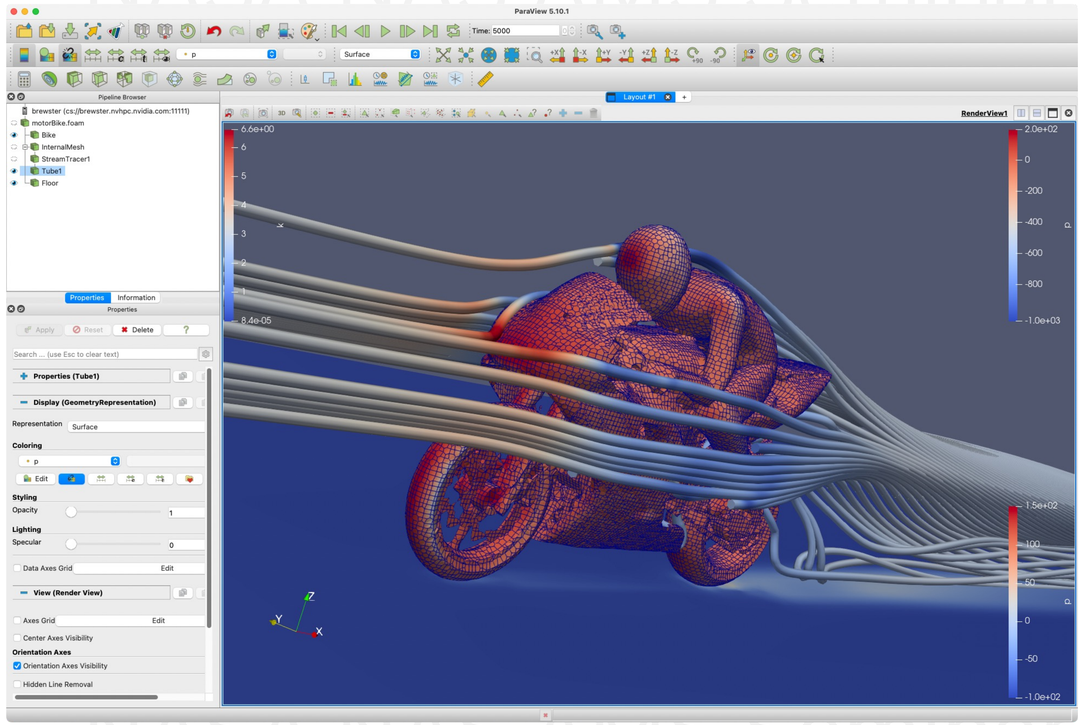


NVIDIA.

OpenFOAM 2206

MotorBike 5M





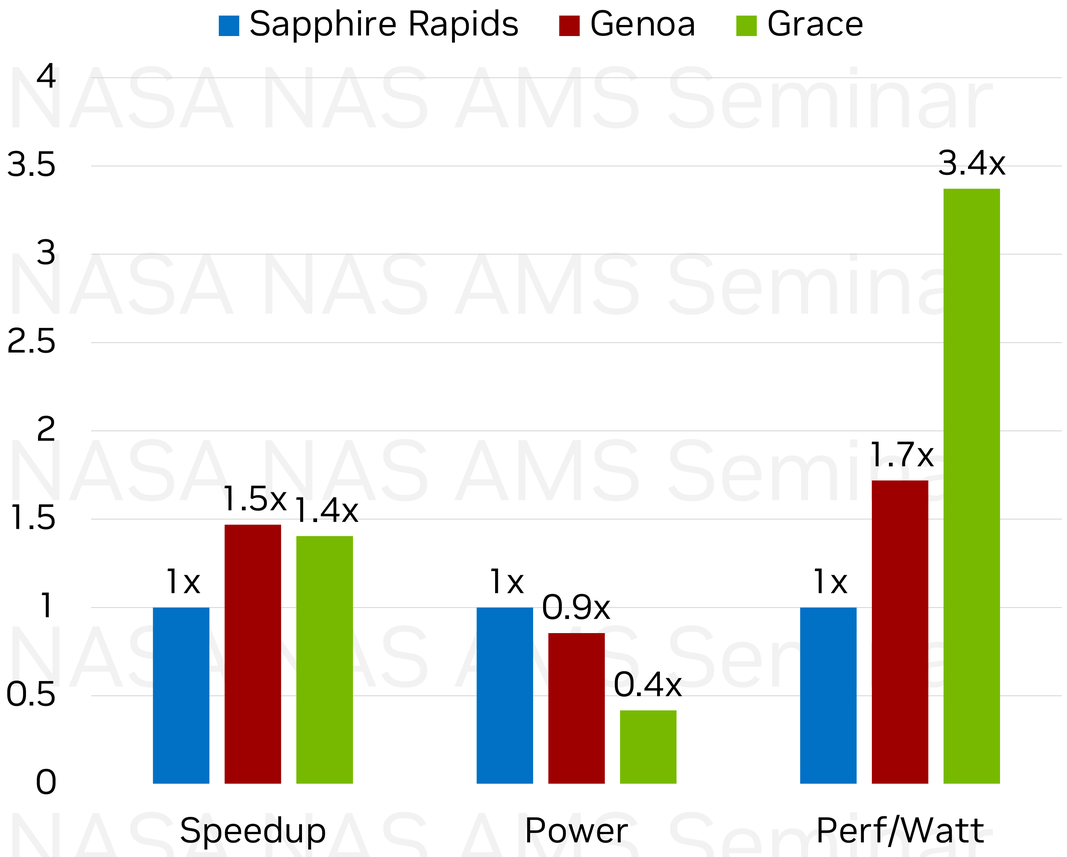
Sapphire Rapids: Intel Xeon Platinum 8470Q, 52c @ 2.1GHz - 3.8GHz
Genoa: AMD EPYC 9654, 96c @ 1.5GHz - 3.7GHz
Grace: NVIDIA Engineering Sample, 72c @ 3.2GHz
Best single socket time using best compilers (GCC, ICC, AOCC) and best rank/thread decomposition

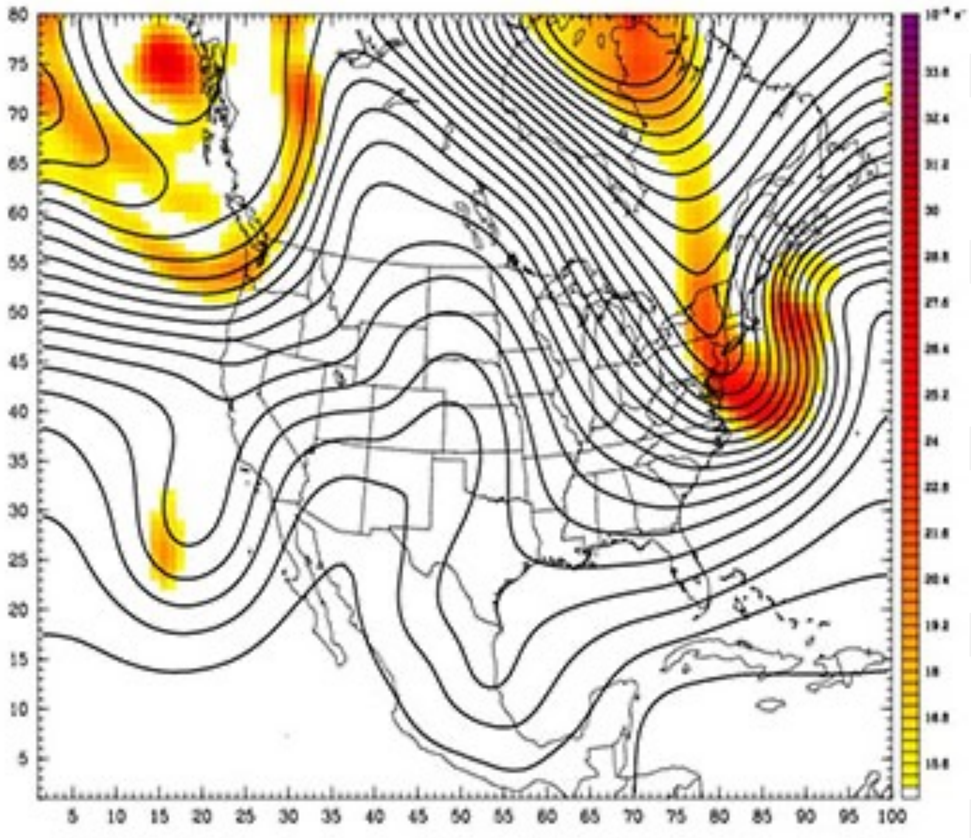
Grace numbers based on engineering sample measurements

NASA NAS AMS Seminar

WRF 4.4.2

CONUS12, 24hr simulation time





Sapphire Rapids: Intel Xeon Platinum 8470Q, 52c @ 2.1GHz - 3.8GHz
Genoa: AMD EPYC 9654, 96c @ 1.5GHz - 3.7GHz
Grace: NVIDIA Engineering Sample, 72c @ 3.2GHz
Best single socket time using best compilers (GCC, ICC, AOCC) and best rank/thread decomposition

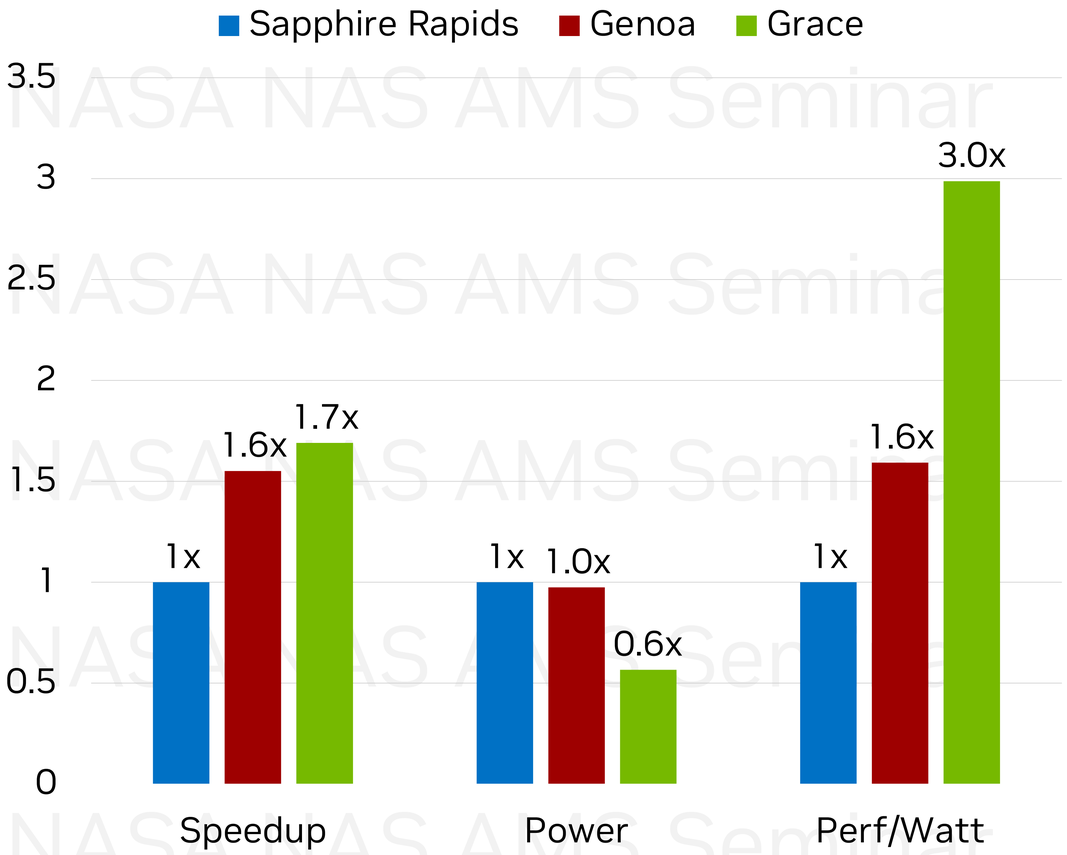
Grace numbers based on engineering sample measurements

NASA NAS AMS Seminar

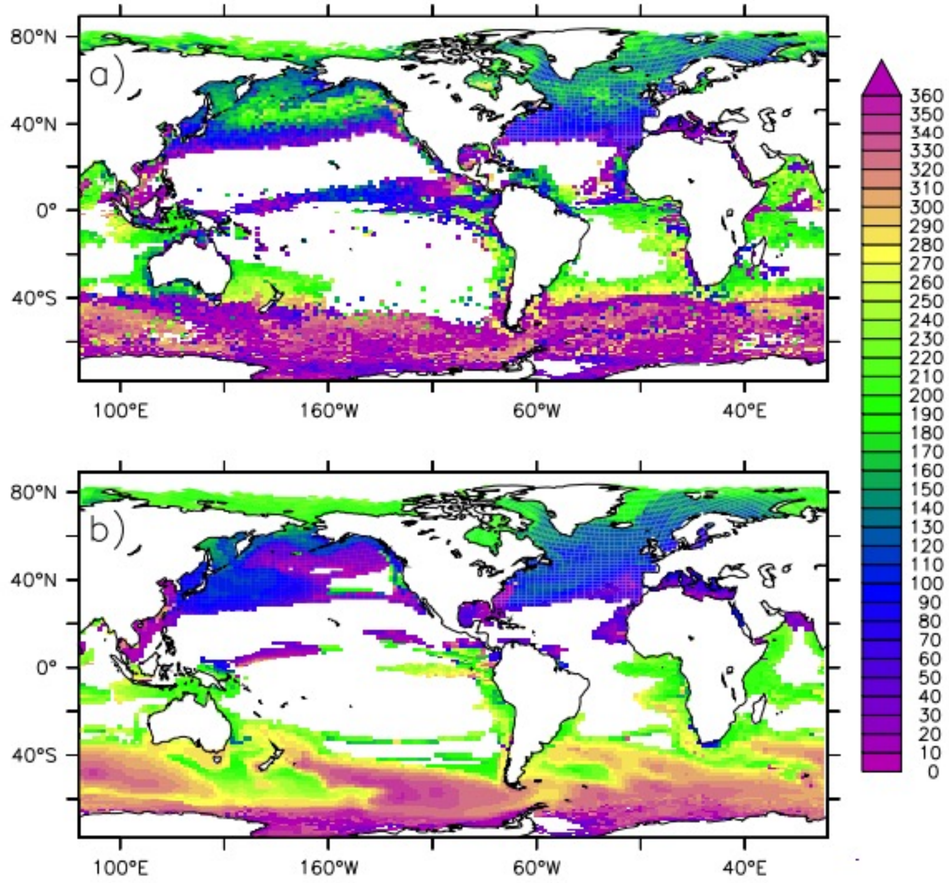
NASA NAS AMS Seminar

NEMO 4.2.0

GYRE_PISCES, nn_GYRE=25, nsteps=50



Sapphire Rapids: Intel Xeon Platinum 8470Q, 52c @ 2.1GHz - 3.8GHz
Genoa: AMD EPYC 9654, 96c @ 1.5GHz - 3.7GHz
Grace: NVIDIA Engineering Sample, 72c @ 3.2GHz
Best single socket time using best compilers (GCC, ICC, AOCC) and best rank/thread decomposition



Aumont, O., Ethé, C., Tagliabue, A., Bopp, L., and Gehlen, M.: PISCES-v2: an ocean biogeochemical model for carbon and ecosystem studies, Geosci. Model Dev., 8, 2465–2513, https://doi.org/10.5194/gmd-8-2465-2015, 2015.

Grace numbers based on engineering sample measurements

NVIDIA.

Developing for the NVIDIA Platform, and NVIDIA Grace

Conclusions First

- Existing GPU applications require no changes for Grace Hopper
  - No new APIs
  - No restructuring
  - No new programming model
  - Developers who choose to can optimize for the Grace Hopper platform
- Existing GPU applications (fully or partially ported) will run better on Grace Hopper
  - Data migration no longer required, may still be a performance optimization
  - When data migrations happen, they happen faster due to C2C interconnect
  - CPU code will benefit from higher bandwidth memory, high thread performance, coherent accesses
  - Existing, stable Unified Memory APIs may be used for performance optimization
- Non-GPU applications will run unmodified and benefit from Grace architecture
- Porting from CPU to GPU is made simpler by Grace Hopper
  - Coherent Memory Subsystem
  - C2C interconnect
  - Programming model choice
- Some new capabilities may be unlocked
  - Larger data sets
  - Workflows that utilize both halves



NASA NAS AMS Seminar

NASA NAS AMS Seminar

Programming the NVIDIA Platform

Unmatched Developer Flexibility

On PCs



On Prem



At the Edge



In the Public Cloud



Parallelism in Standard Languages

C++

Fortran

python

Directives For Existing Apps

OpenACC

OpenMP

Peak Performance

NVIDIA CUDA

C++ | Fortran | Python

Acceleration Libraries

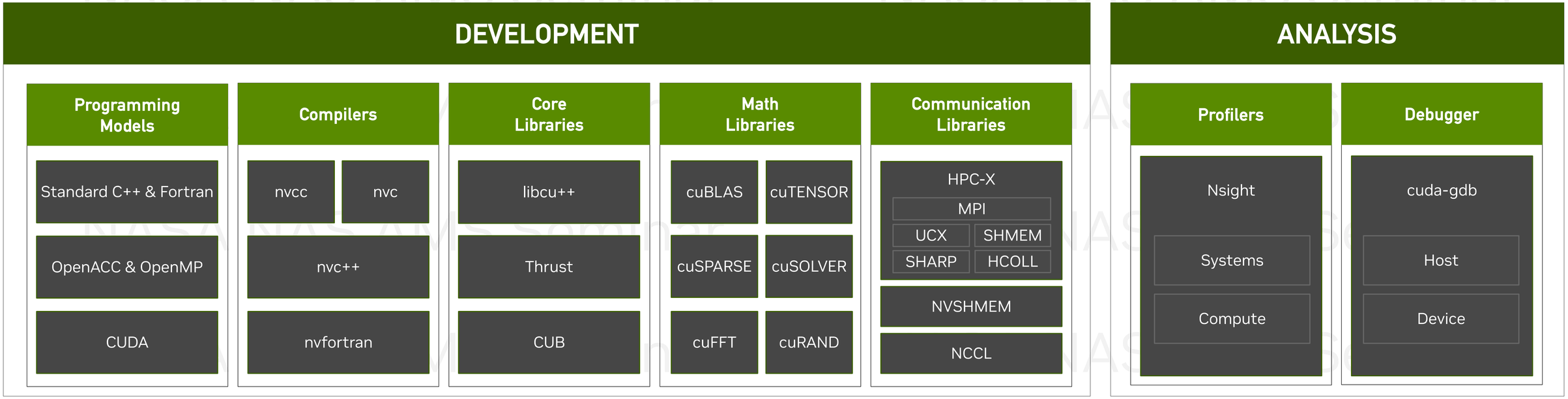
(AI, Data Analytics, Algebra, Quantum, Communication)

NASA NAS AMS Seminar

NASA NAS AMS Seminar

NVIDIA HPC SDK

Available at developer.nvidia.com/hpc-sdk, on NGC, via Spack, and in the Cloud



Develop for the NVIDIA Platform: GPU, CPU and Interconnect

Libraries | Accelerated C++ and Fortran | Directives | CUDA

x86_64 | Arm | OpenPOWER

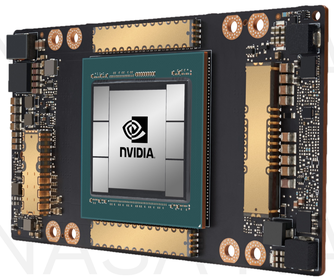
7-8 Releases Per Year | Freely Available

NVIDIA

NASA NAS AMS Seminar

HPC Compilers

NVC | NVC++ | NVFORTRAN

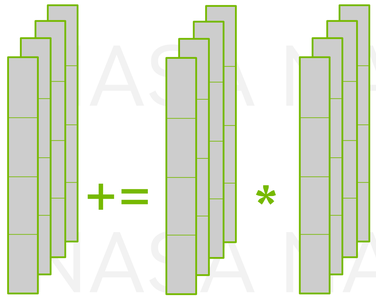


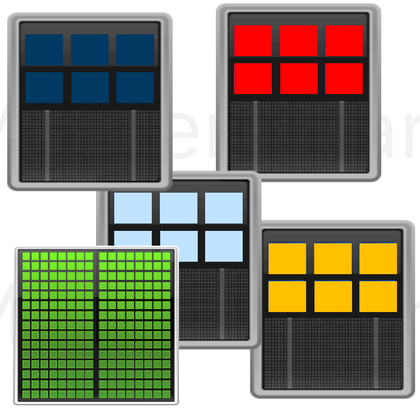
Fortran

NVIDIA CUDA

OpenACC
More Science, Less Programming

OpenMP





Accelerated

H100
Automatic

Programmable

Standard Languages
Directives
CUDA

CPU Optimized

Directives
Vectorization

Multi-Platform

x86_64
AArch64
OpenPOWER

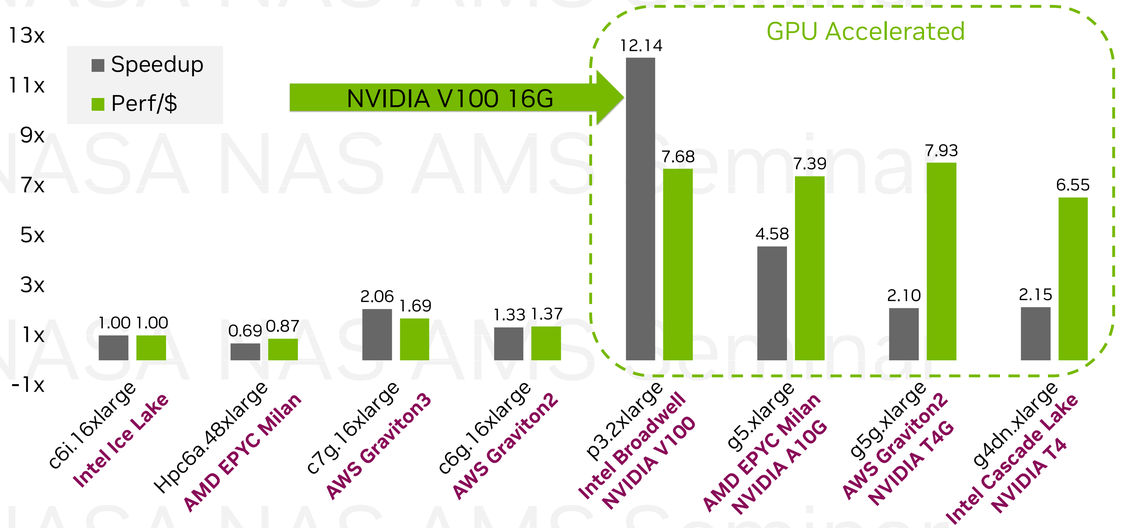
NVIDIA

NASA NAS AMS Seminar

Portable Computing with NVIDIA HPC SDK

developer.nvidia.com/blog/accelerating-nvidia-hpc-software-with-sve-on-aws-graviton3

- NVIDIA HPC SDK fully supports Arm CPUs
- Auto-vectorization for SVE and NEON
- See NVIDIA technical blog for details





Technical Blog

Subscribe

News

Sep 19, 2022 English

Accelerating NVIDIA HPC Software with SVE on AWS Graviton3

By John Linford, Scott Manley and Graham Lopez

Discuss (2) 0 Like

Tags: AWS, featured, HPC / Supercomputing, HPC SDK, News

The latest NVIDIA HPC SDK update expands portability and now supports the Arm-based AWS Graviton3 processor. In this post, you learn how to enable Scalable Vector Extension (SVE) auto-vectorization with the NVIDIA compilers to maximize the performance of HPC applications running on the AWS Graviton3 CPU.

NASA NAS AMS Seminar

Choosing A Programming Model

There can be ~~only~~ ***more than*** one

Libraries	Standard Languages	Compiler Directives	CUDA Languages
• Accelerate common operations with little/no code changes. • Expert-tuned performance. • Forward support guarantees.	• Strong cross-platform support. • Single source code for multiple platforms. • Reduced learning curve.	• High cross-platform support. • Single source code for multiple platforms. • Reduced learning curve. • Additional programmer control.	• Exposes full GPU capabilities. • Trades portability for performance. • Distinct GPU/CPU code paths. • Full programmer control.
Programmer Productivity			Programmer Control

By design these approaches are interoperable so developers can choose the right balance for their needs

NASA NAS AMS Seminar

NASA NAS AMS Seminar

HPC Programming in ISO Fortran

Examples of Patterns Accelerated in NVFORTRAN (by CUTENSOR)

```
d = 2.5 * ceil(transpose(a)) + 3.0 * abs(transpose(b))
d = 2.5 * ceil(transpose(a)) + 3.0 * abs(b)
d = reshape(a,shape=[ni,nj,nk])
d = reshape(a,shape=[ni,nk,nj])
d = 2.5 * sqrt(reshape(a,shape=[ni,nk,nj],order=[1,3,2]))
d = alpha * conjg(reshape(a,shape=[ni,nk,nj],order=[1,3,2]))
d = reshape(a,shape=[ni,nk,nj],order=[1,3,2])
d = reshape(a,shape=[nk,ni,nj],order=[2,3,1])
d = reshape(a,shape=[ni*nj,nk])
d = reshape(a,shape=[nk,ni*nj],order=[2,1])
d = reshape(a,shape=[64,2,16,16,64],order=[5,2,3,4,1])
d = abs(reshape(a,shape=[64,2,16,16,64],order=[5,2,3,4,1]))
c = matmul(a,b)
c = matmul(transpose(a),b)
c = matmul(reshape(a,shape=[m,k],order=[2,1]),b)
c = matmul(a,transpose(b))
c = matmul(a,reshape(b,shape=[k,n],order=[2,1]))
```

```
c = matmul(transpose(a),transpose(b))
c = matmul(transpose(a),reshape(b,shape=[k,n],order=[2,1]))
d = spread(a,dim=3,ncopies=nk)
d = spread(a,dim=1,ncopies=ni)
d = spread(a,dim=2,ncopies=nx)
d = alpha * abs(spread(a,dim=2,ncopies=nx))
d = alpha * spread(a,dim=2,ncopies=nx)
d = abs(spread(a,dim=2,ncopies=nx))
d = transpose(a)
d = alpha * transpose(a)
d = alpha * ceil(transpose(a))
d = alpha * conjg(transpose(a))
c = c + matmul(a,b)
c = c - matmul(a,b)
c = c + alpha * matmul(a,b)
d = alpha * matmul(a,b) + c
d = alpha * matmul(a,b) + beta * c
```

NVIDIA

Automatically Offloading BLAS, LAPACK, and FFT

- NVBLAS
  - Drop-in replacement library for a subset of the BLAS interface built on top of cuBLAS using only the cuBLASXt API
  - Simply link NVBLAS before the CPU BLAS library
    - e.g., `-lnvblas -lblas`
  - Setup the `nvblas.config` file
    - Comment out `NVBLAS_AUTOPIN_MEM_ENABLED` so that NVBLAS works with managed data pointers
    - Tune the value of `NVBLAS_TILE_DIM`

- NVLAMATH
  - Drop-in replacement library for a subset of the LAPACK interface built on top of cuSOLVER
  - Simply add `–gpu=nvlamath` to the compile line and `-cudalib=nvlamath` to the link line

- cuFFTW
  - A subset of the FFTW3 API to facilitate GPU acceleration of existing FFTW applications
  - Simply link cuFFTW library
    - e.g., `-lcufft -lcufftw`
  - If the application is using the legacy Fortran interface of FFTW, we need to write a wrapper

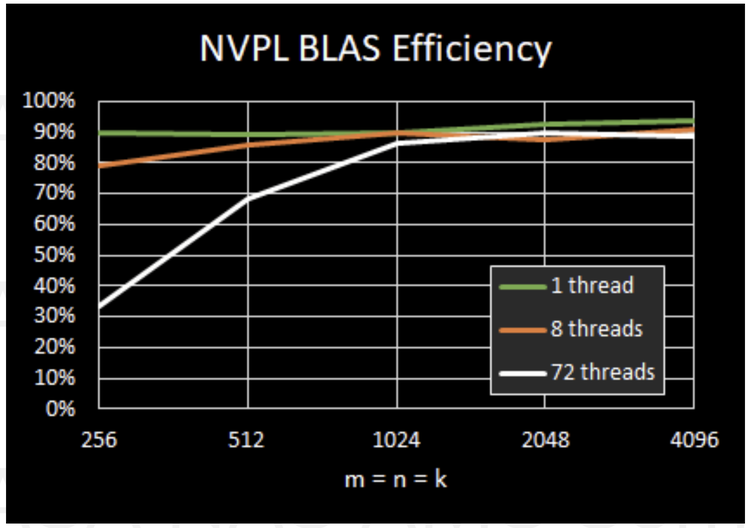
NASA NAS AMS Seminar

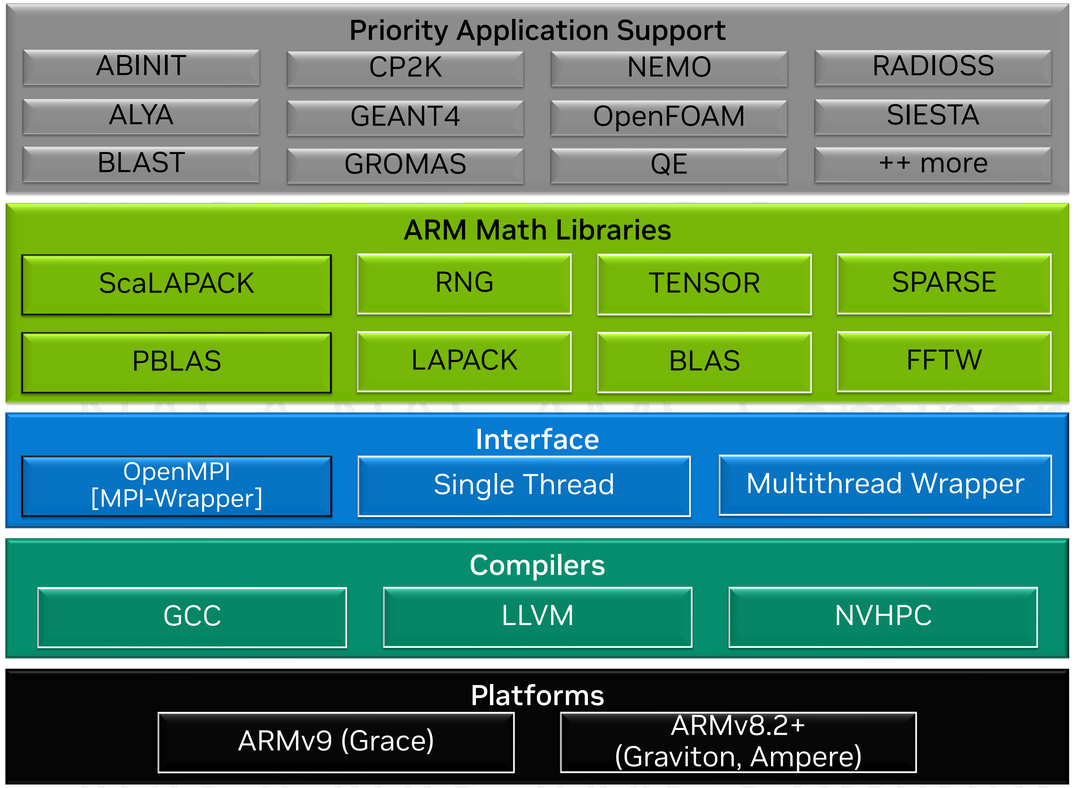
NASA NAS AMS Seminar

NVIDIA Performance Libraries (NVPL)

Math Libraries Optimized for Arm CPUs

- Easily port applications to NVIDIA Grace CPU
- Industry-leading performance and efficiency
- De-facto standard interfaces (e.g., BLAS, FFTW)
- New interfaces (e.g., SPARSE, TENSOR)





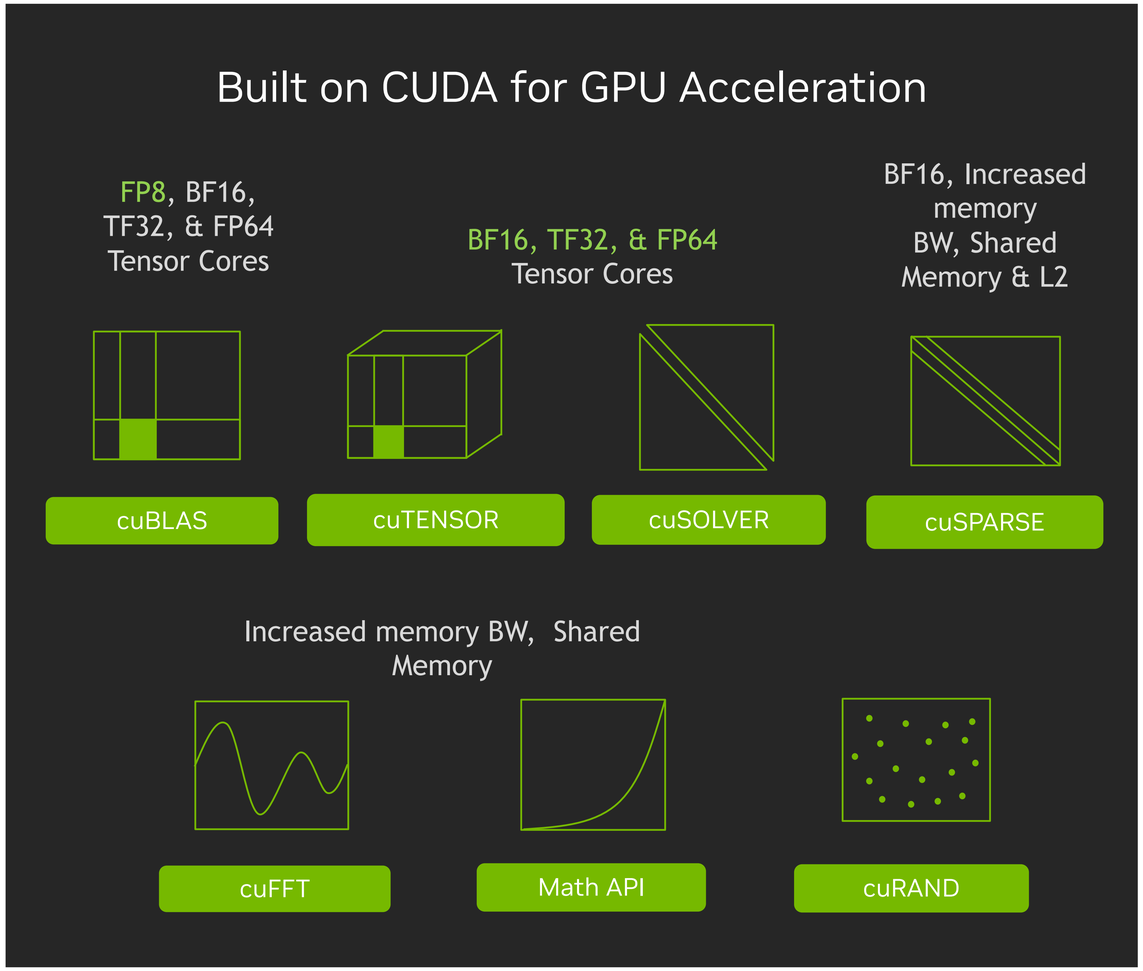
NVIDIA

NASA NAS AMS Seminar

NVIDIA Math Libraries for a Single Node

Optimized for Hopper (any host CPU) and Grace Hopper

- Applications using NVPL for CPU code can use NVIDIA Math Libraries for GPU acceleration and unleash full Grace Hopper potential
- Shipped with CUDA Toolkit and HPC-SDK
- Libraries targeting dense and sparse matrix multiplication in deep learning:
  - cuBLASLt: Dense matmuls with activation fusions:
    - Added support for FP8 (CUDA 11.8)
  - cuSPARSELt: Sparse x dense matmuls with activation fusions:
    - Support for FP8 and H100 in upcoming release
- MathDx - Suite of Device extensions to Math Libraries
  - Enables fusion by inlining optimized math functions (Access)
  - cuFFTDx in early access
  - Will add support for Grace Hopper platforms



NASA NAS AMS Seminar

NVIDIA Math Libraries for Full Scale Computing

Grace Hopper in Multi-Node

- NVIDIA Math MGMN (Multi-GPU, Multi-Node) Libraries from HPC-SDK (arm64-sbsa) optimized for H100
- In conjunction with NVIDIA Communication Libraries adapts to the scale-out configurations with InfiniBand or MNNVL interconnect

DGX-GH200 - 256 Grace Hopper Superchips with MNNVL interconnect



1EFLOPS AI Performance | **144TB** unified fast memory
36 L2 NVLink switches | **900 GB/s** GPU-to-GPU bandwidth | **128 TB/s** bisection bandwidth

NVIDIA MGMN Math Libraries for GPU (built on CUDA)

cuSOLVERMp
- Factorization
  - LU
  - LL^T (Cholesky)
  - QR
  - All linear solvers support multiple-RHS
- Symmetric Eigensolver
  - Standard
  - Generalized

cuFFTMp
- 2D and 3D FFTs
- Decompositions
  - Slab (1D)
  - Pencil (2D)

CuBLASMp
- COMING SOON
- First Release Q4'2023
  - Matrix Multiply Only

Communication Libraries

HPC-X

MPI

UCX | SHMEM | SHARP | HCOLL

NVSHMEM

NCCL

InfiniBand or MNNVL

Accelerated Standard Languages

Parallel performance for wherever your code runs

ISO C++

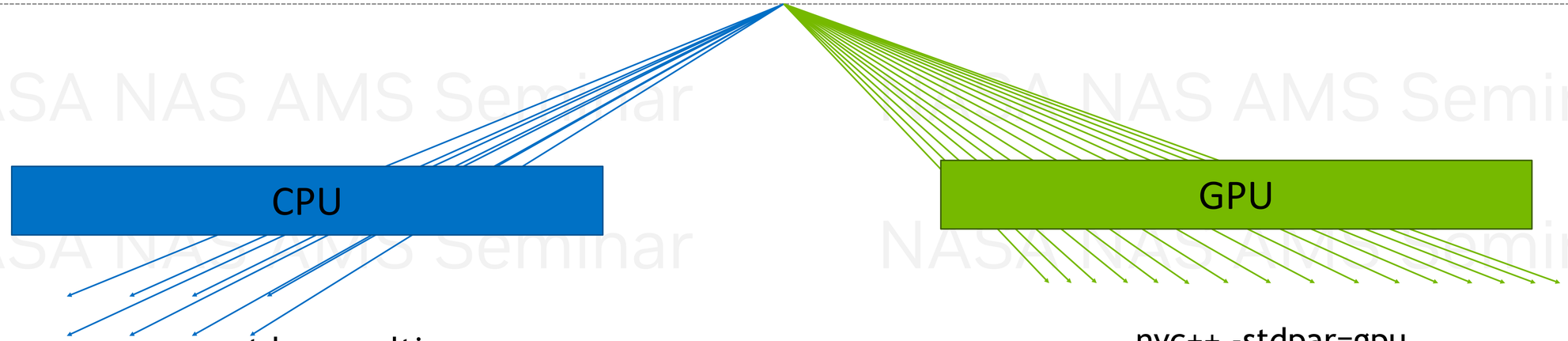
```
std::transform(par, x, x+n, y,
   y,[=](float x, float y){
      return y + a*x;
    }
);
```

ISO Fortran

```
do concurrent (i = 1:n)
   y(i) = y(i) + a*x(i)
enddo
```

Python

```
import cunumeric as np
…
def saxpy(a, x, y):
    y[:] += a*x
```



nvc++ -stdpar=multicore
nvfortran -stdpar=multicore
legate -cpus 16 saxpy.py

nvc++ -stdpar=gpu
nvfortran -stdpar=gpu
legate -gpus 1 saxpy.py

NASA NAS AMS Seminar

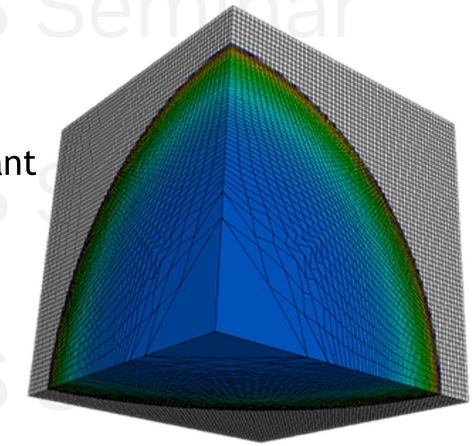
NASA NAS AMS Seminar

Lulesh with Standard C++

C++ Hydrodynamics Mini-app from LLNL

Rewritten from OpenMP to ISO C++

- More composable, compact, and elegant
- Easier to read and maintain
- ISO Standard
- Portable – nvc++, g++, icpc, MSVC, …



```cpp
static inline
void CalcHydroConstraintForElems(Domain &domain, Index_t length,
    Index_t *regElemlist, Real_t dvovmax, Real_t& dthydro)
{
#if _OPENMP
  const Index_t threads = omp_get_max_threads();
  Index_t hydro_elem_per_thread[threads];
  Real_t dthydro_per_thread[threads];
#else
  Index_t threads = 1;
  Index_t hydro_elem_per_thread[1];
  Real_t dthydro_per_thread[1];
#endif
#pragma omp parallel firstprivate(length, dvovmax)
  {
    Real_t dthydro_tmp = dthydro ;
    Index_t hydro_elem = -1 ;
#if _OPENMP
    Index_t thread_num = omp_get_thread_num();
#else
    Index_t thread_num = 0;
#endif
#pragma omp for
    for (Index_t i = 0 ; i < length ; ++i) {
      Index_t indx = regElemlist[i] ;

      if (domain.vdov(indx) != Real_t(0.)) {
        Real_t dtdvov = dvovmax / (FABS(domain.vdov(indx))+Real_t(1.e-20)) ;

        if ( dthydro_tmp > dtdvov ) {
          dthydro_tmp = dtdvov ;
          hydro_elem = indx ;
        }
      }
    }
    dthydro_per_thread[thread_num] = dthydro_tmp ;
    hydro_elem_per_thread[thread_num] = hydro_elem ;
  }
  for (Index_t i = 1; i < threads; ++i) {
    if(dthydro_per_thread[i] < dthydro_per_thread[0]) {
      dthydro_per_thread[0] = dthydro_per_thread[i];
      hydro_elem_per_thread[0] = hydro_elem_per_thread[i];
    }
  }
  if (hydro_elem_per_thread[0] != -1) {
    dthydro = dthydro_per_thread[0] ;
  }
  return ;
}
```

C++ with OpenMP

```cpp
static inline void CalcHydroConstraintForElems(Domain &domain, Index_t length,
                Index_t *regElemlist,
                Real_t dvovmax,
                Real_t &dthydro)
{
  dthydro = std::transform_reduce(
    std::execution::par, counting_iterator(0), counting_iterator(length),
    dthydro, [](Real_t a, Real_t b) { return a < b ? a : b; },
    [=, &domain](Index_t i)
  {
      Index_t indx = regElemlist[i];
      if (domain.vdov(indx) == Real_t(0.0)) {
        return std::numeric_limits<Real_t>::max();
      } else {
        return dvovmax / (std::abs(domain.vdov(indx)) + Real_t(1.e-20));
      }
  });
}
```

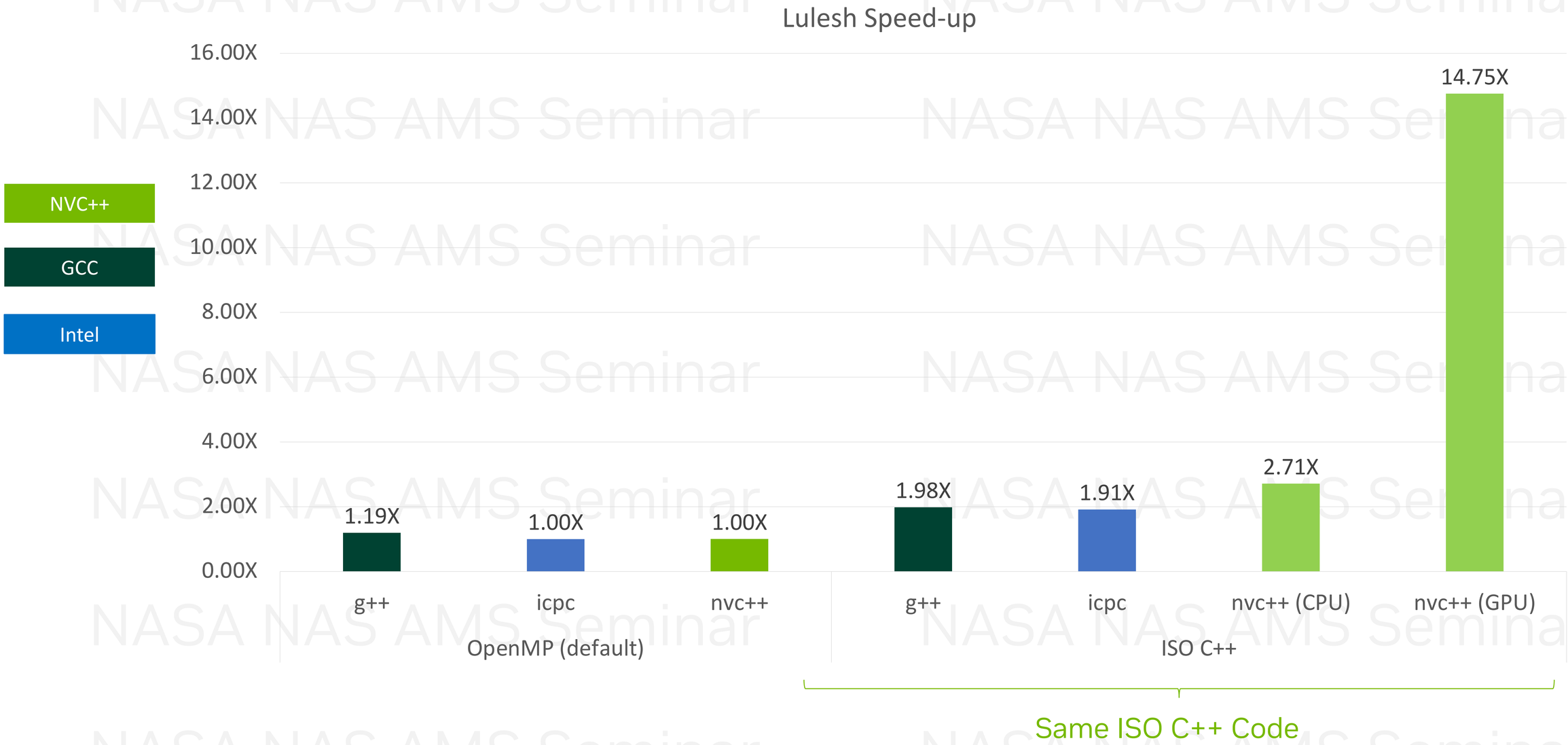
Standard C++

codesign.llnl.gov/lulesh

NVIDIA

C++ Standard Parallelism

Lulesh Performance



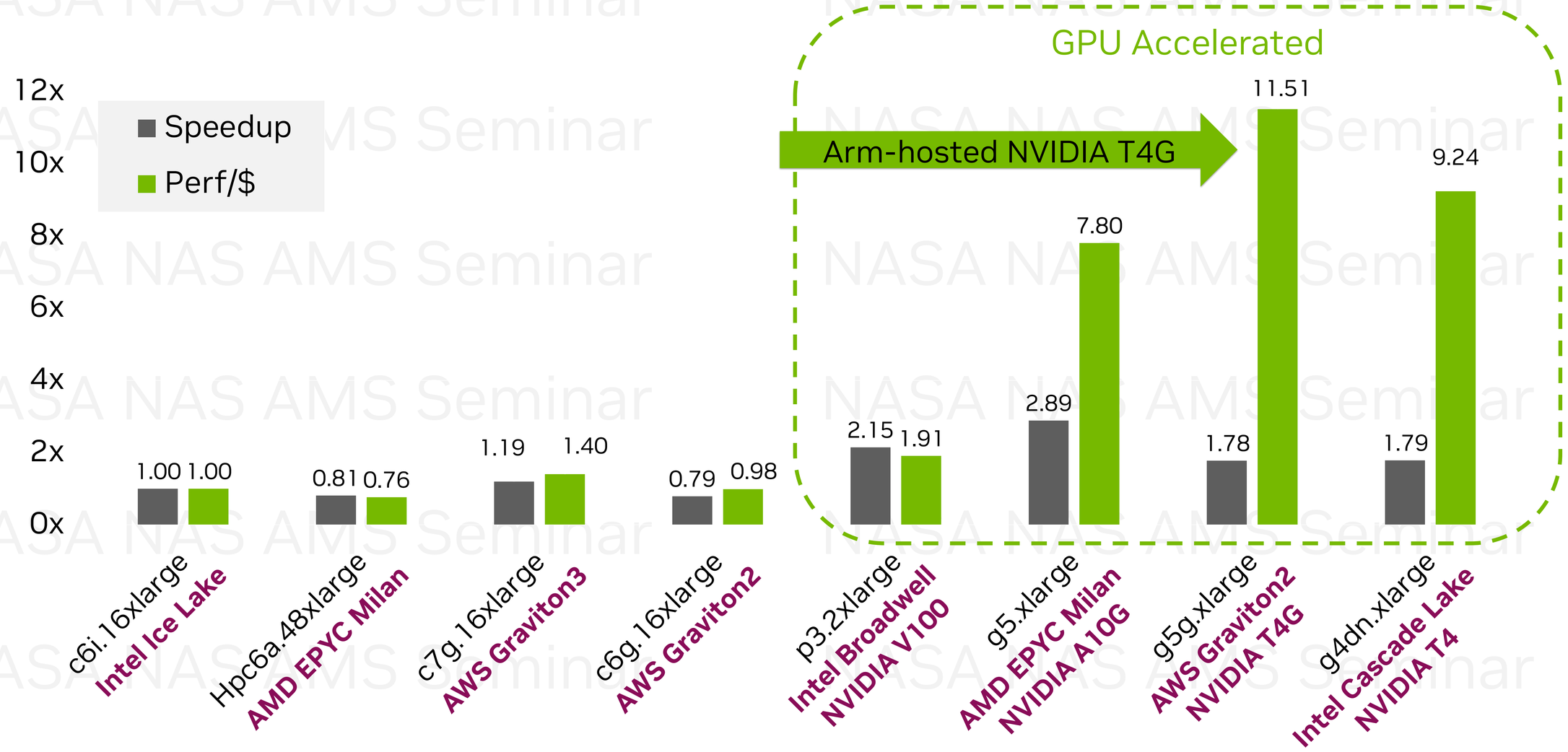
AMD EPYC 7742 CPU, NVIDIA A100 GPU. g++ version 10.3.0, icpc version 2021.5.0, nvc++ version 22.3

NASA NAS AMS Seminar

NASA NAS AMS Seminar

ISO C++ LULESH with NVIDIA HPC SDK

Standard Language Parallelism across all architectures



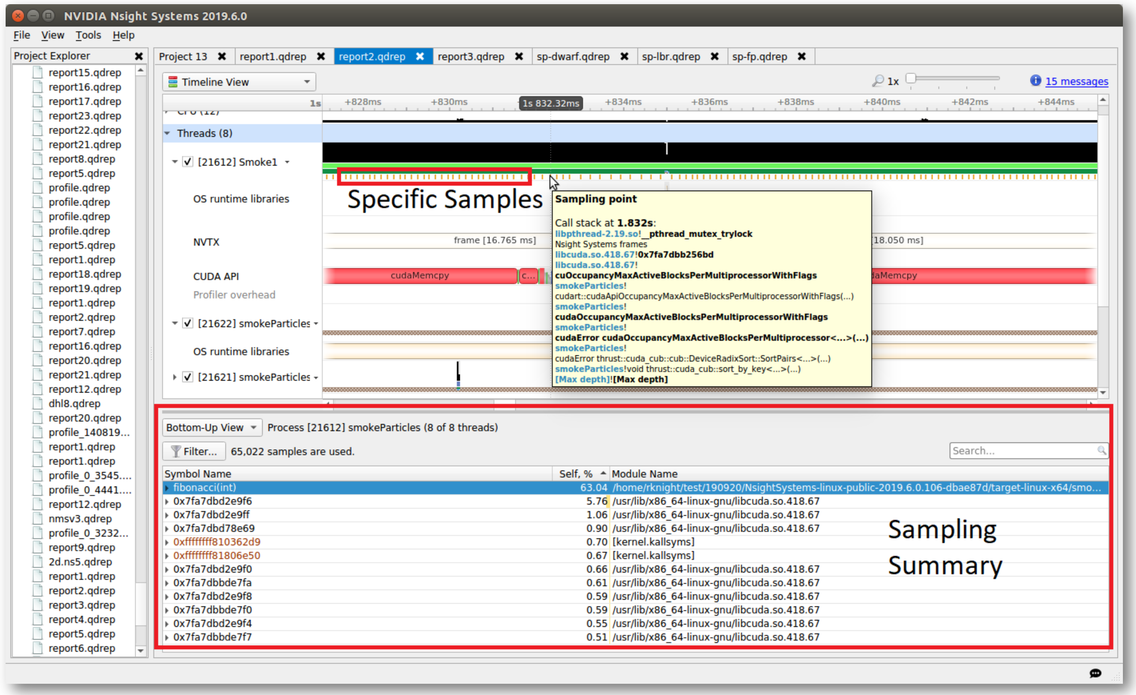
NVIDIA

NASA NAS AMS Seminar

Debuggers and Profilers

Full capability on Grace-Hopper

- Generally, all major debugging and profiling tools for x86+Hopper are available on Grace-Hopper
  - Free: gdb, lldb, perf, etc.
  - Commercial: Linaro Forge, Perforce Totalview
- Exception: vendor-specific tools, e.g. Intel Vtune
  - Similar capabilities are provided by other tools on Grace
- **NVIDIA Nsight tools** have full feature-parity between GH200 and x86+Hopper
  - Anything you can do with Nsight tools on x86+Hopper, you can do on GH200 with the same workflow
- GH200 has hundreds of performance counters (PMUs)
  - **Computational intensity, bandwidth, instruction mix…**
  - GH200 implements standard Arm **in-core** PMUs ➔ Profiling tools that support Arm support GH200
  - GH200 implements a full suite of **uncore** PMUs to monitor data movement, inter-core communication, etc.
  - Many community tools (PAPI, HPCToolkit, etc.) support GH200 and can read in-core and uncore PMUs



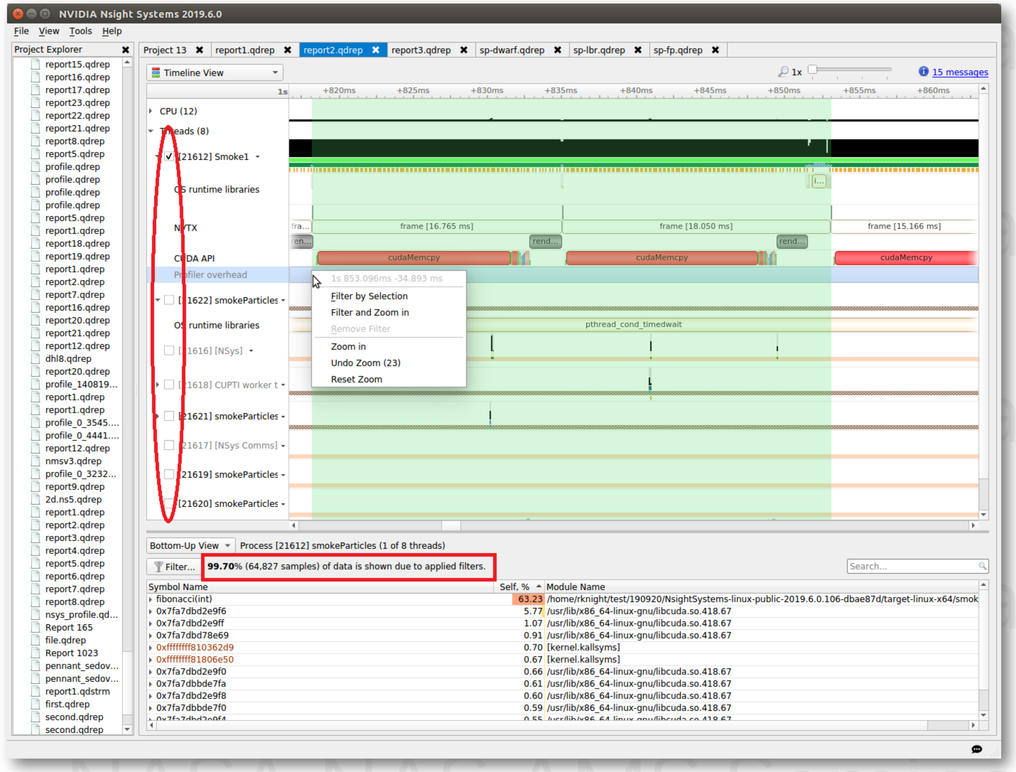
NVIDIA

NASA NAS AMS Seminar

NVIDIA Nsight Systems: Application Performance Analysis

Low overhead analysis tool for developers tuning and scaling software across entire systems

- GPU compute & graphics, NIC networking, and CPU on a unified timeline
- GPU performance metrics sampling
- GPU workload trace
- Built-in statistics and analyzers, as well as export for third-party tools
- Compute
  - CUDA, OpenACC, OpenMP, DirectML, cuBLAS, cuDNN, TensorRT, DALI, cuDF/RAPIDS/SPARK, cuFile, NVTX, …
  - GPU Direct RDMA, Storage, P2P, & CVV access sampling
- Graphics
  - OptiX, Vulkan, OpenGL, D3D12, D3D11, DXR, DXGI, PIX
  - Graphics multi-frame stutter analysis
- Networking
  - MPI, UCX, NCCL, NVSHMEM, OpenSHMEM
  - NVIDIA NIC & DPU Infiniband metrics sampling



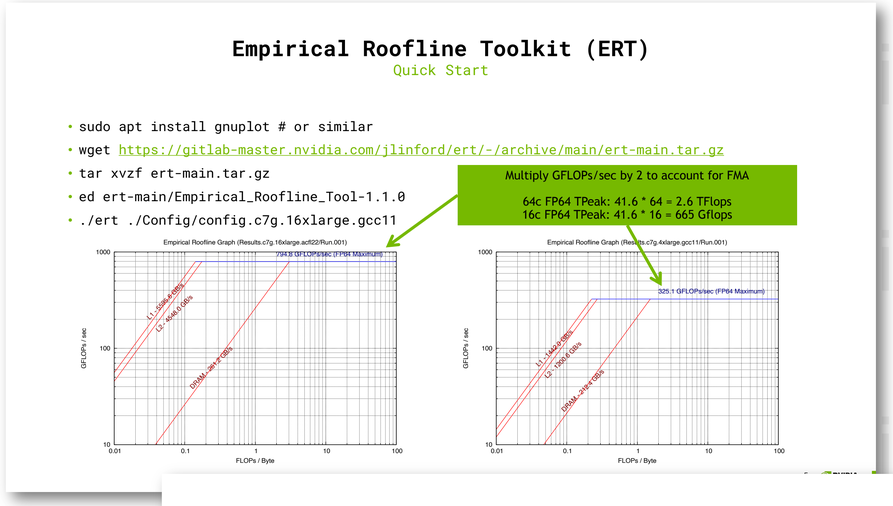
NVIDIA

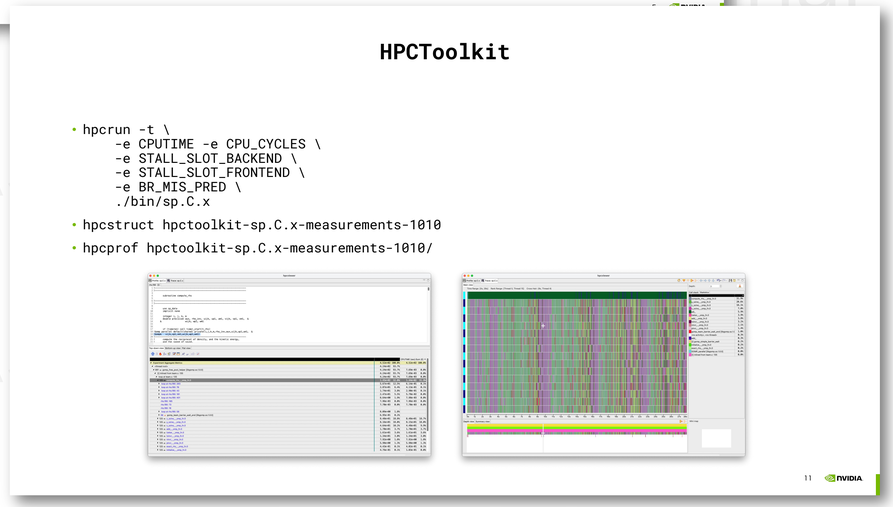
NASA NAS AMS Seminar

OSS Performance Engineering Tools

A mix of NVIDIA tools and community tools – developers want to use the same tools on Grace

- **Objective**: Enable Grace developers to the same extent that they are enabled on other CPUs
- Working with the NSIGHT team to define CPU profiling workflows that enable Grace developers
  - Top-down, Bottom-up, Roofline
  - Metrics required
  - Hardware performance counters
- Working with community tools to better support Grace
  - PAPI, libpfm4 now support Grace in-core counters
  - Enabling open source tool development for Grace





PAPI & libpfm4 with Arm Neoverse Support

- Download PAPI
- cd src
- ./configure --prefix=$PWD/..
- make -j && make install

```
[jlinford@c7g-4xlarge-dy-c7g4xlarge-2 papi-jlinford-arm-neoverse]$ ./bin/papi_native_avail | grep SVE
|           Data memory read accesses (includes SVE)                          |
|           Data memory write accesses (includes SVE)                         |
|           Unaligned accesses (includes speculatively executed SVE load and s|
| SVE_INST_SPEC                                                               |
|           SVE operations sepculatively executed                             |
| SVE_PRED_SPEC                                                               |
|           SVE predicated operations speculatively executed                  |
| SVE_PRED_EMPTY_SPEC                                                         |
|           SVE predicated operations with no active predicates speculatively |
| SVE_PRED_FULL_SPEC                                                          |
|           SVE predicated operations with all active predicates speculatively|
| SVE_PRED_PARTIAL_SPEC                                                       |
|           SVE predicated operations with partially active predicates specula|
| SVE_LDFF_SPEC                                                               |
|           SVE first-fault load operations speculatively executed            |
| SVE_LDFF_FAULT_SPEC                                                         |
|           SVE first-fault load operations speculatively executed which set F|
```

```
[jlinford@c7g-4xlarge-dy-c7g4xlarge-2 papi-jlinford-arm-neoverse]$ ./bin/papi_avail | grep Yes
PAPI_L1_DCM  0x80000000  Yes   No   Level 1 data cache misses
PAPI_L1_ICM  0x80000001  Yes   No   Level 1 instruction cache misses
PAPI_L2_DCM  0x80000002  Yes   No   Level 2 data cache misses
PAPI_TLB_DM  0x80000014  Yes   No   Data translation lookaside buffer misses
PAPI_L2_LDM  0x80000019  Yes   No   Level 2 load misses
PAPI_STL_ICY 0x80000025  Yes   Yes  Cycles with no instruction issue
PAPI_HW_INT  0x80000029  Yes   Yes  Hardware interrupts
PAPI_BR_MSP  0x8000002e  Yes   No   Conditional branch instructions mispredicted
PAPI_BR_PRC  0x8000002f  Yes   Yes  Conditional branch instructions correctly predicted
PAPI_TOT_INS 0x80000032  Yes   No   Instructions completed
PAPI_FP_INS  0x80000034  Yes   No   Floating point instructions
PAPI_LD_INS  0x80000035  Yes   No   Load instructions
PAPI_SR_INS  0x80000036  Yes   No   Store instructions
PAPI_BR_INS  0x80000037  Yes   No   Branch instructions
PAPI_VEC_INS 0x80000038  Yes   Yes  Vector/SIMD instructions (could include integer)
PAPI_RES_STL 0x80000039  Yes   No   Cycles stalled on any resource
PAPI_TOT_CYC 0x8000003b  Yes   No   Total cycles
PAPI_LST_INS 0x8000003c  Yes   Yes  Load/store instructions completed
PAPI_SYC_INS 0x8000003d  Yes   Yes  Synchronization instructions completed
PAPI_L1_DCA  0x80000040  Yes   No   Level 1 data cache accesses
PAPI_L2_DCA  0x80000041  Yes   Yes  Level 2 data cache accesses
PAPI_L1_DCR  0x80000043  Yes   No   Level 1 data cache reads
PAPI_L2_DCR  0x80000044  Yes   No   Level 2 data cache reads
PAPI_L1_DCW  0x80000046  Yes   No   Level 1 data cache writes
PAPI_L2_DCW  0x80000047  Yes   No   Level 2 data cache writes
PAPI_L1_ICH  0x80000049  Yes   Yes  Level 1 instruction cache hits
PAPI_L1_ICA  0x8000004c  Yes   No   Level 1 instruction cache accesses
PAPI_L2_TCA  0x80000059  Yes   No   Level 2 total cache accesses
```

NASA NAS AMS Seminar

Conclusions

NASA NAS AMS Seminar

Conclusions

- Datacenters built on NVIDIA Grace Superchips are approximately **twice as effective**
  - **Versatile**: More than just Grace+Hopper or 2xGrace
  - **Performant**: NVIDIA's Arm-based Grace CPU matches or beats contemporary x86 CPUs
  - **Efficient**: Grace is approximately twice as power efficient as contemporary x86 CPUs
- NVIDIA Grace builds on the rich and mature Arm CPU software ecosystem for an excellent "**it just works**" out-of-the-box experience
- The NVIDIA platform expands the Arm software ecosystem with applications, compilers, libraries, developer tools, etc.
- If you're already using the NVIDIA platform, expect a frictionless move to NVIDIA Grace Superchips



