Nvidia Grace-Hopper Architecture

Giri Chukkapalli

outline

- Future HPC and AI Applications
- Post-Moore HW Landscape
- Nvidia Grace-Hopper
- Future SW stack
- Conclusions
- White Papers & References

FUTURE HPC/DL APPLICATIONS

OCT 14, 2022

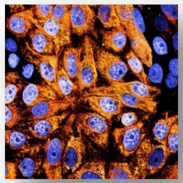
POST-EXASCALE APPLICATIONS

- Full systems modeling
  - Earth systems: Complex Climate Models
  - Biological systems: Human Microbiome Modeling
  - Complex Engineering Systems: Aircraft, Power Plant etc.
- Extreme scale Data driven
  - CERN LHC data
  - Square KM Array Sky Survey data
  - Emerging IoT, High Res Sensor data
- AI/ML/DL Applications
  - LLM
  - Multi-Modal
  - Games, Multi-Agent systems (Reinforcement Learning methods)
- Combination of all three
  - workflow

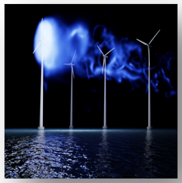
- o Exascale Systems: Homogeneous CPU compute to Homogeneous GPU compute

Workloads of the Modern Supercomputer

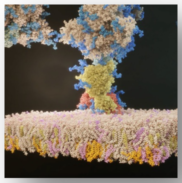
EDGE



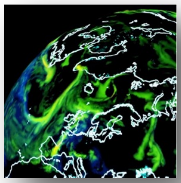
SIM + AI



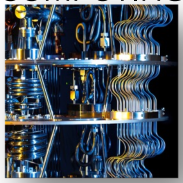
SIMULATION

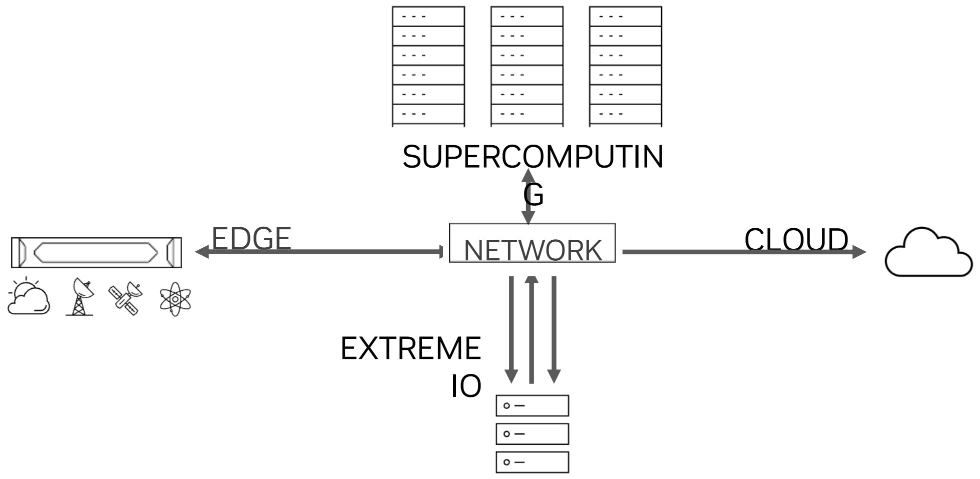


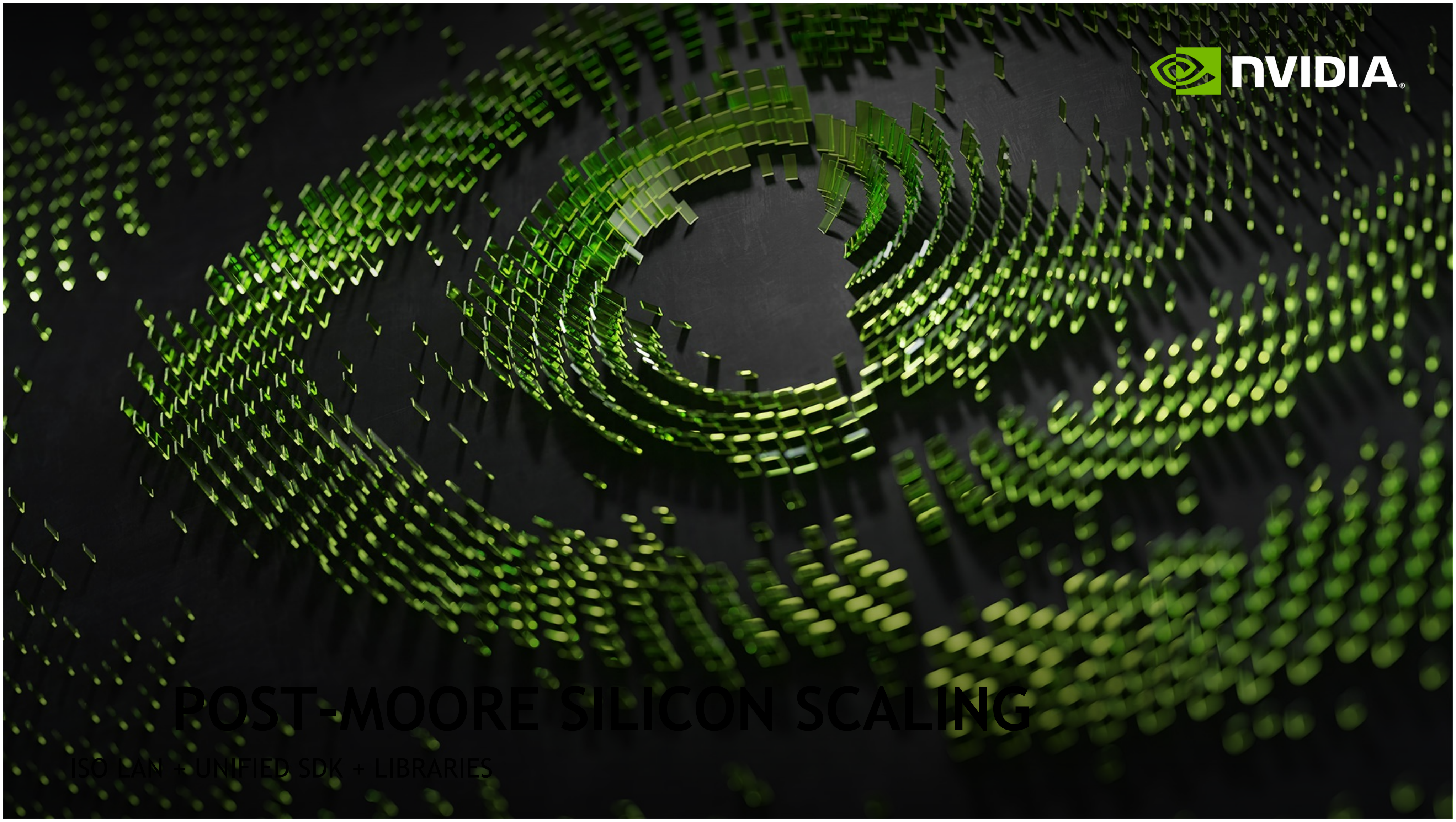
DIGITAL TWIN



QUANTUM COMPUTING





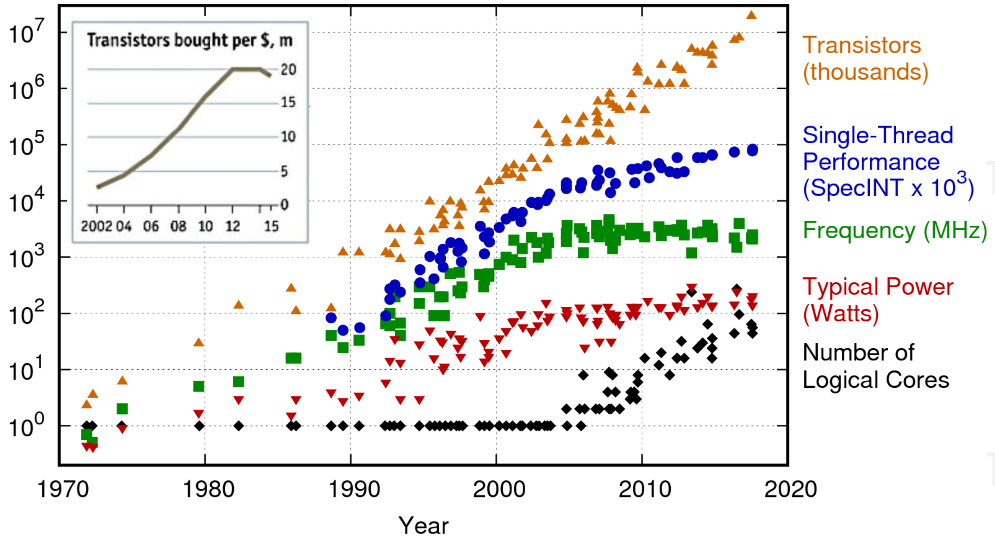


POST-MOORE SILICON SCALING

ISO LAN + UNIFIED SDK + LIBRARIES

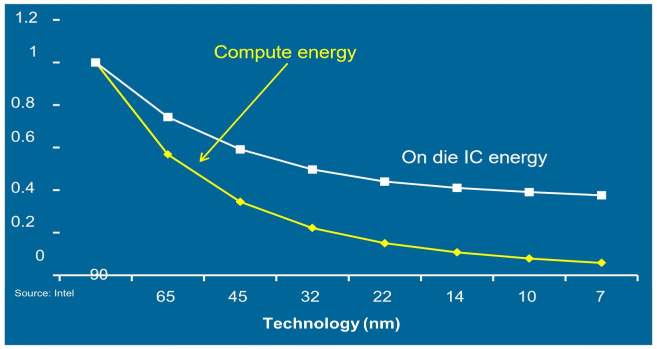
MOORE'S OBSERVATION

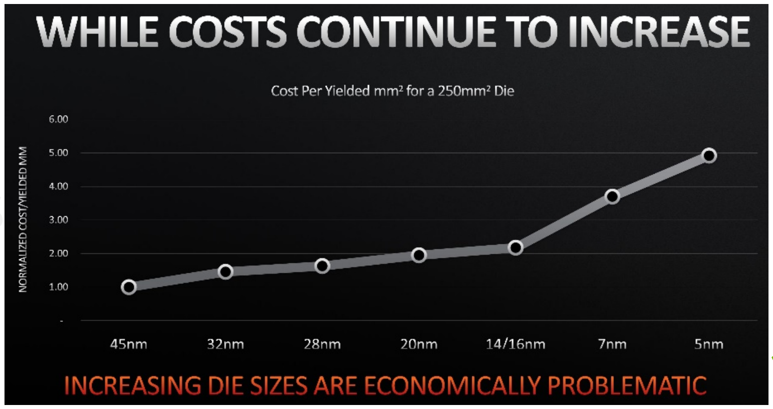
42 Years of Microprocessor Trend Data



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

- Original:
  - Transistor density doubles
  - Power stays same and Cost stays same
- Last two are gone
  - Due to loss of Dennard scaling and Litho and other process complexities
- Different features scale differently with process
  - Logic, SRAM, Wires, and IOs
    - Distorting over time





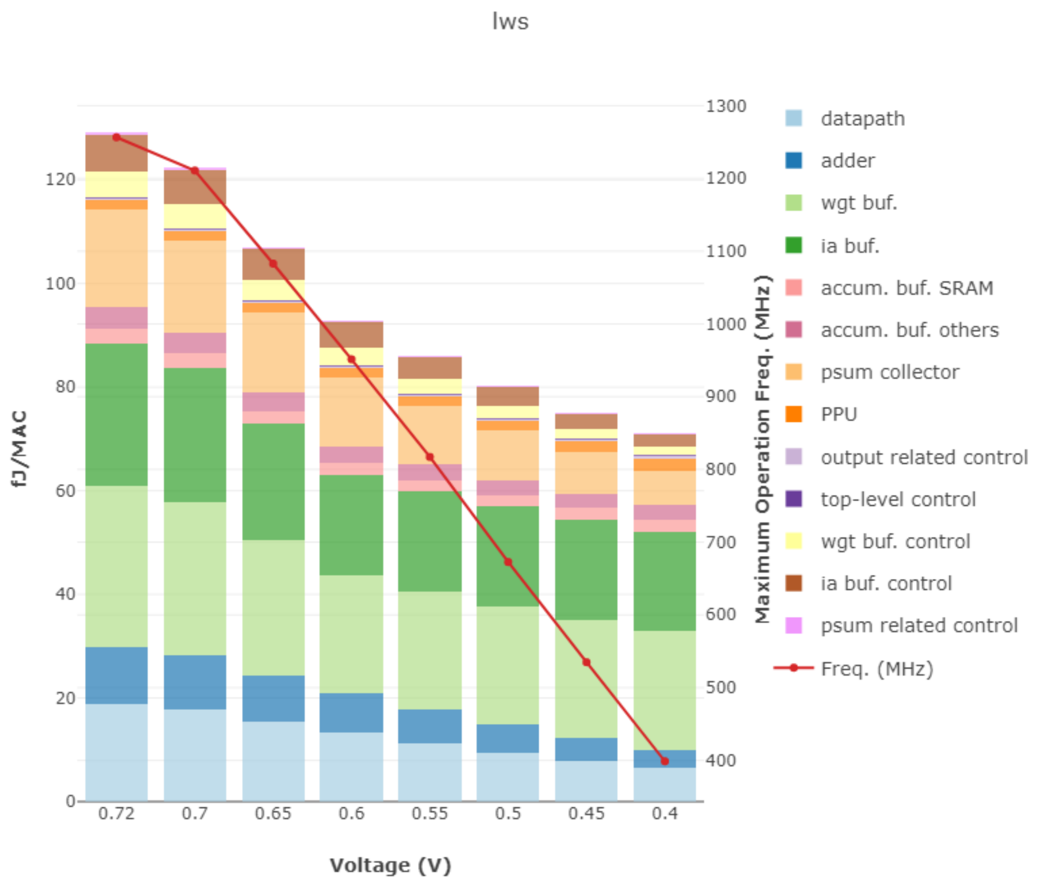
NVIDIA

ENERGY DOMINATED BY MEMORY AND DATA

70 fJ/MAC

35 fJ/OP

29 TOPS/W



MORE THAN MOORE

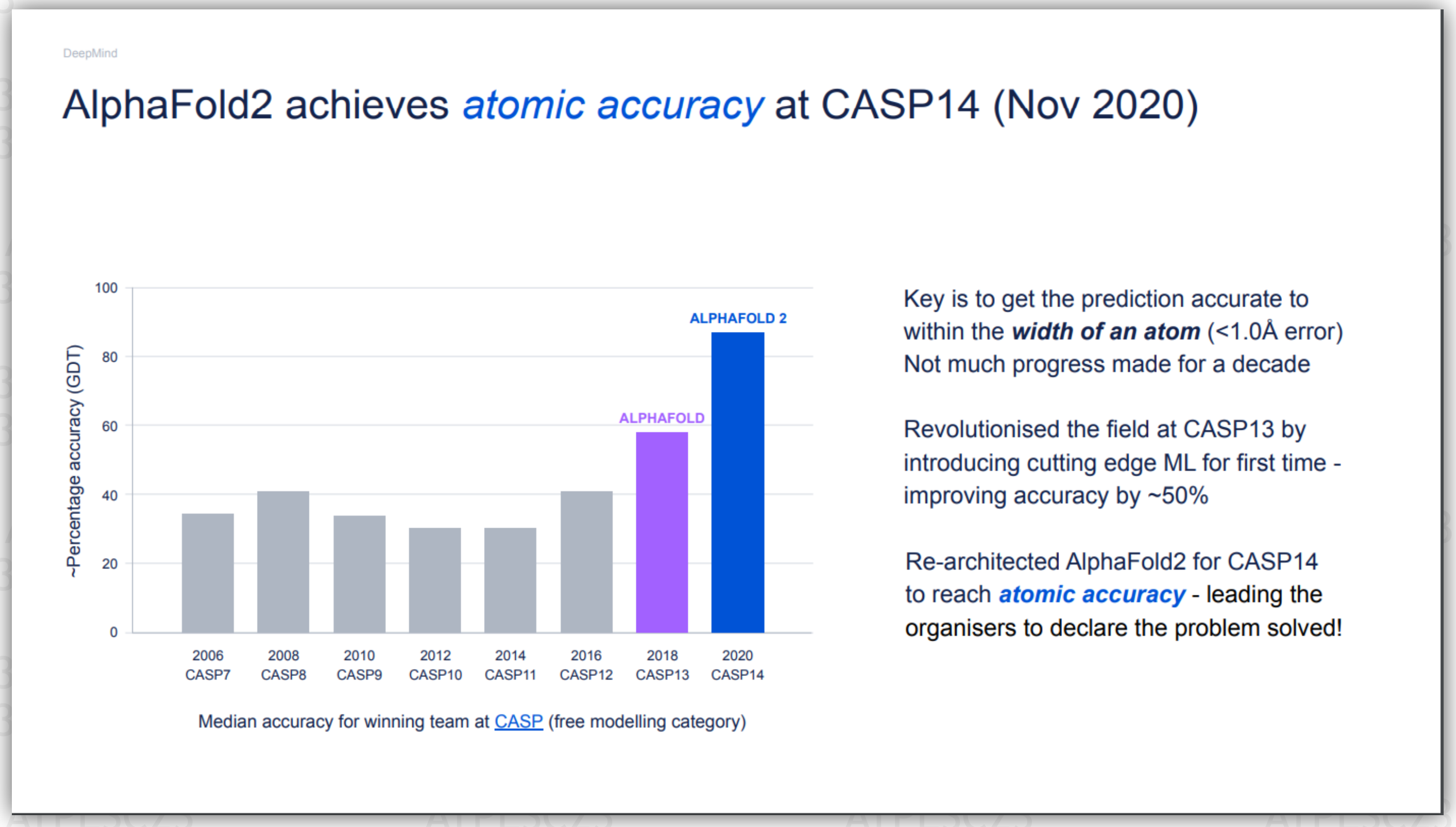
- Nvidia maintains "More than Moore" by optimizing
  - End-to-End mapping of Applications to Supercomputing System
  - Algorithmic, SW, architectural, Packaging, Process Technology
  - Try to maintain 2X to 6X Gen-to-Gen perf improvement
  - Requires close collaboration with HPC/ML/DL Community
  - Transformer Acceleration in Hopper is the best recent example

ML/DL WILL CONTINUE MOORE'S OBSERVATION IN HPC

- ML/DL is rapidly becoming the 4th pillar of Scientific Discovery
- Approximations in Mod-Sim, Inverse problems, Insights from experimental/empirical data etc.
- Protein Structure Prediction became a solved problem
- Sub-Seasonal Forecast Competition is won by MSFT ML/DL Team
- Physics informed NNs in CFD

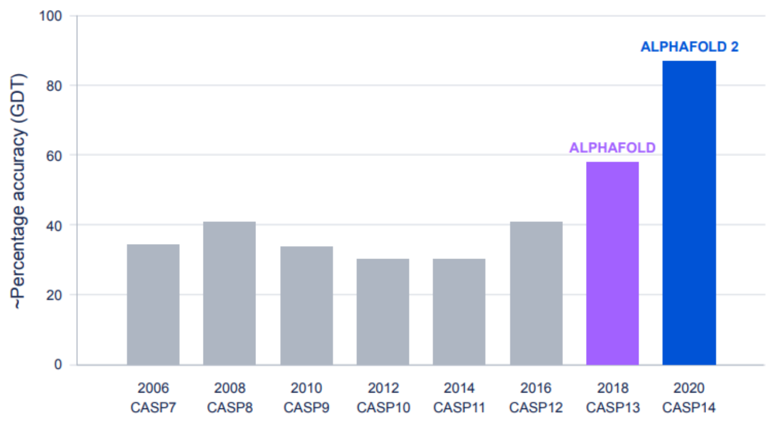
AI TO ACCELERATE SCIENTIFIC DISCOVERY

Demis Hassabis - DeepMind



DeepMind

AlphaFold2 achieves *atomic accuracy* at CASP14 (Nov 2020)

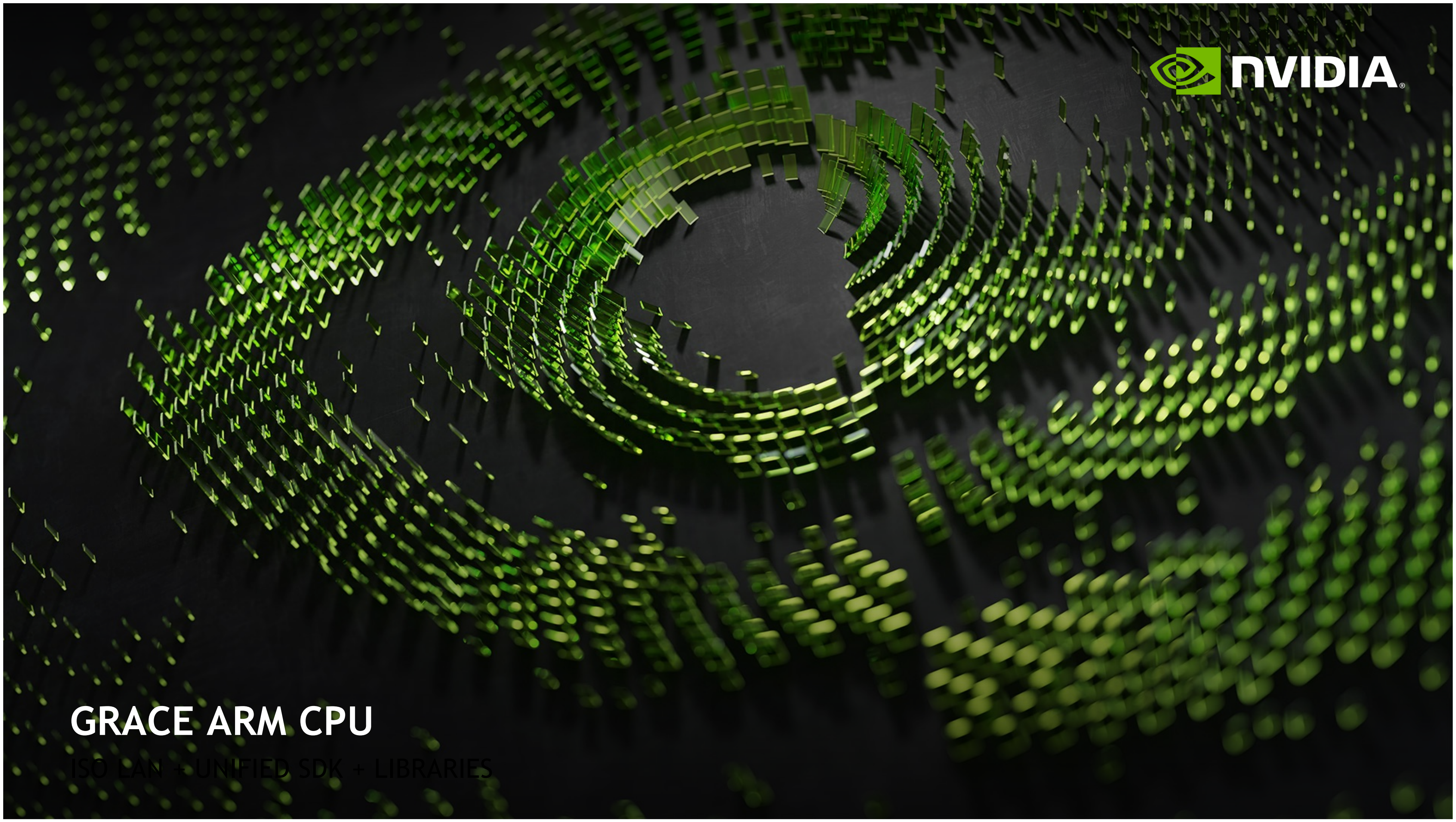


Median accuracy for winning team at CASP (free modelling category)

Key is to get the prediction accurate to within the ***width of an atom*** (<1.0Å error)
Not much progress made for a decade

Revolutionised the field at CASP13 by introducing cutting edge ML for first time - improving accuracy by ~50%

Re-architected AlphaFold2 for CASP14 to reach ***atomic accuracy*** - leading the organisers to declare the problem solved!



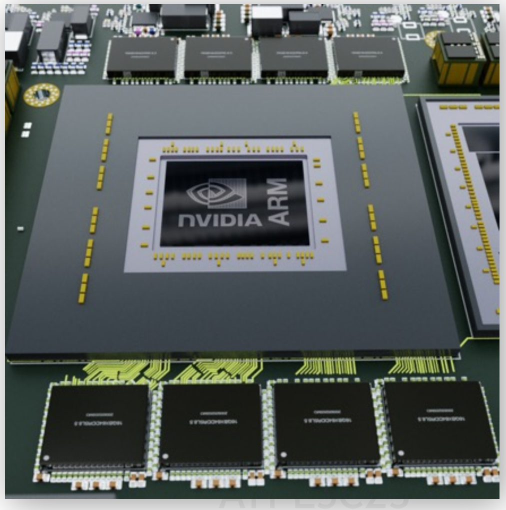
GRACE ARM CPU

ISO LAN + UNIFIED SDK + LIBRARIES

NV ARM OVERVIEW

Server Class ARM CPU

- 64bit Server Class Core and SoC
    - Arm V9.0 ISA Compliant aarch64 core
    - Full SVE-2 Vector Extensions support, inclusive of NEON instructions
    - Supports 48-bit Virtual and 48-bit Physical address space
- Balanced architecture between Single Core Perf, Core count, Memory and IO subsystems
- Supports Arm Server ready (SBSA), Boot compliant (SBBR) and manageability (SBMG) open standards
    - Linux, system management, HPC stacks will run out of the box



NVIDIA ARM CORE

Optimized for Single Thread Performance

Scalar Side

- Wide Super Scalar OoO single-threaded high performance Core Pipeline
  - Supporting Multi-stage branch prediction and advanced prefetching algorithms
  - 1LD + 4LD/ST pipes, 6 ALU pipes
- 64B Cacheline
- 64KB L1ICache, 64KB L1DCache
- 1MB Private L2Cache

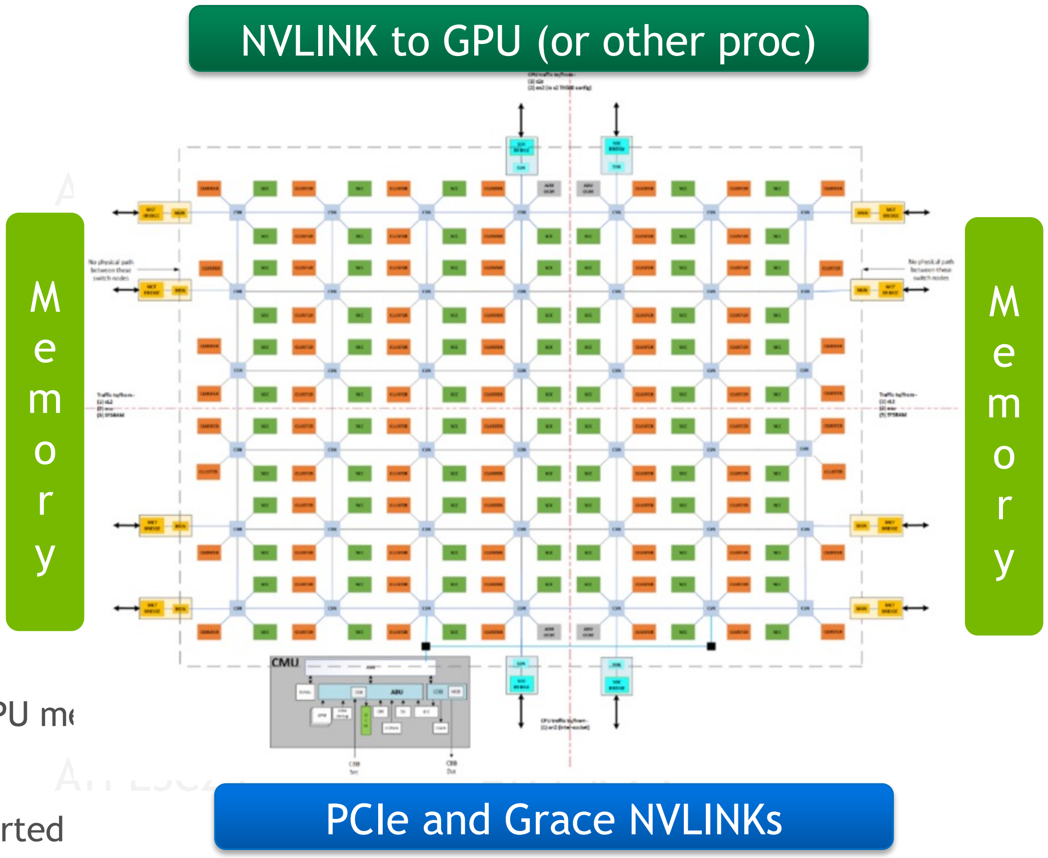
Vector Side

- Four 128-bit SVE2 Execution pipes capable of 16 DP FLOPS per Cycle
- Support 64bit, 32bit, 16bit and bfloat16 FP and int8
- Complex datatypes and math
- Enables easier vectorization through Predicate and mask instructions
- Lane widening and narrowing instructions
- Classic and non-temporal Gather Load and Scatter Store instructions
- Crypto extensions

PROCESSOR SOC

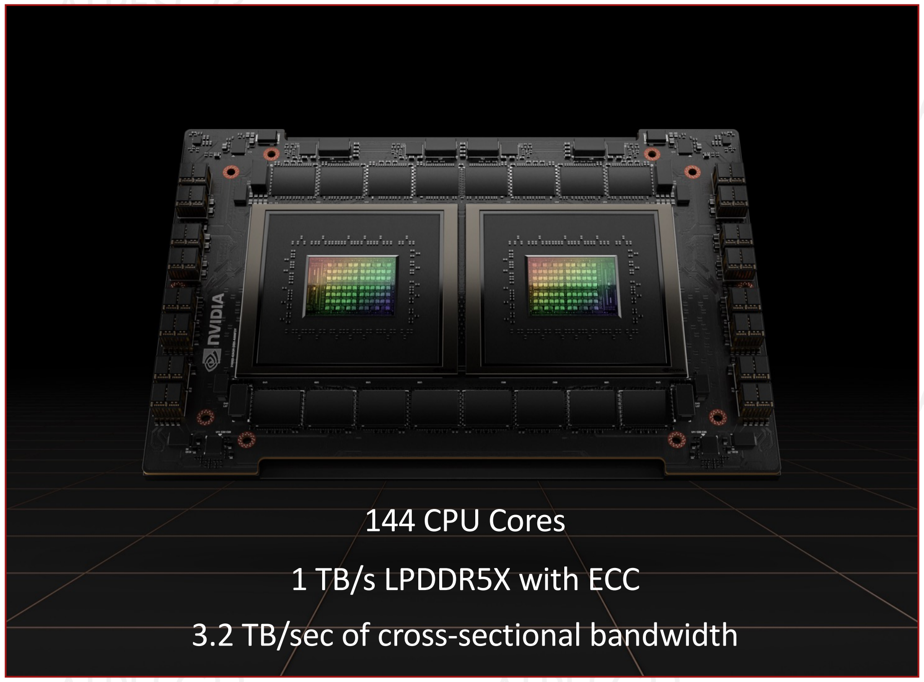
Augmented custom logic to support memory movement

- Monolithic SoC
  - Up to 117MB shared L3 cache
  - >3TB/s on-die mesh bisection BW
- Extensive set of Core and un-Core perf counters
- Thermal monitoring and power management
- DVFS support with multiple voltage domain
- Individual core power and clock gating support
- Tx and Rx paths optimized for 400Gbps fabric
- ARM V9 ISA virtualization and security support
- Custom SoC level logic support for GPUDirect, CPU-GPU m movement and synchronization
- 120GB, 240GB, 480GB LPDDR5 CapaATPESC23es supported



NVIDIA Grace and Grace Hopper

High Performance for an Energy Constrained World



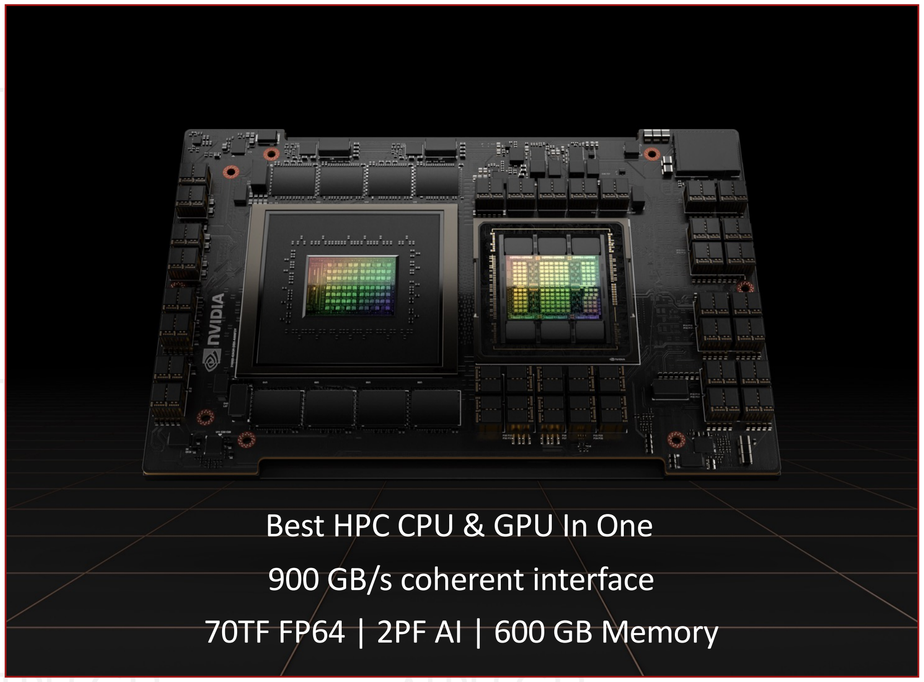
144 CPU Cores

1 TB/s LPDDR5X with ECC

3.2 TB/sec of cross-sectional bandwidth

Grace CPU Superchip

High-performance CPU for HPC and cloud computing



Best HPC CPU & GPU In One

900 GB/s coherent interface

70TF FP64 | 2PF AI | 600 GB Memory

Grace Hopper Superchip

CPU+GPU designed for giant-scale AI and HPC

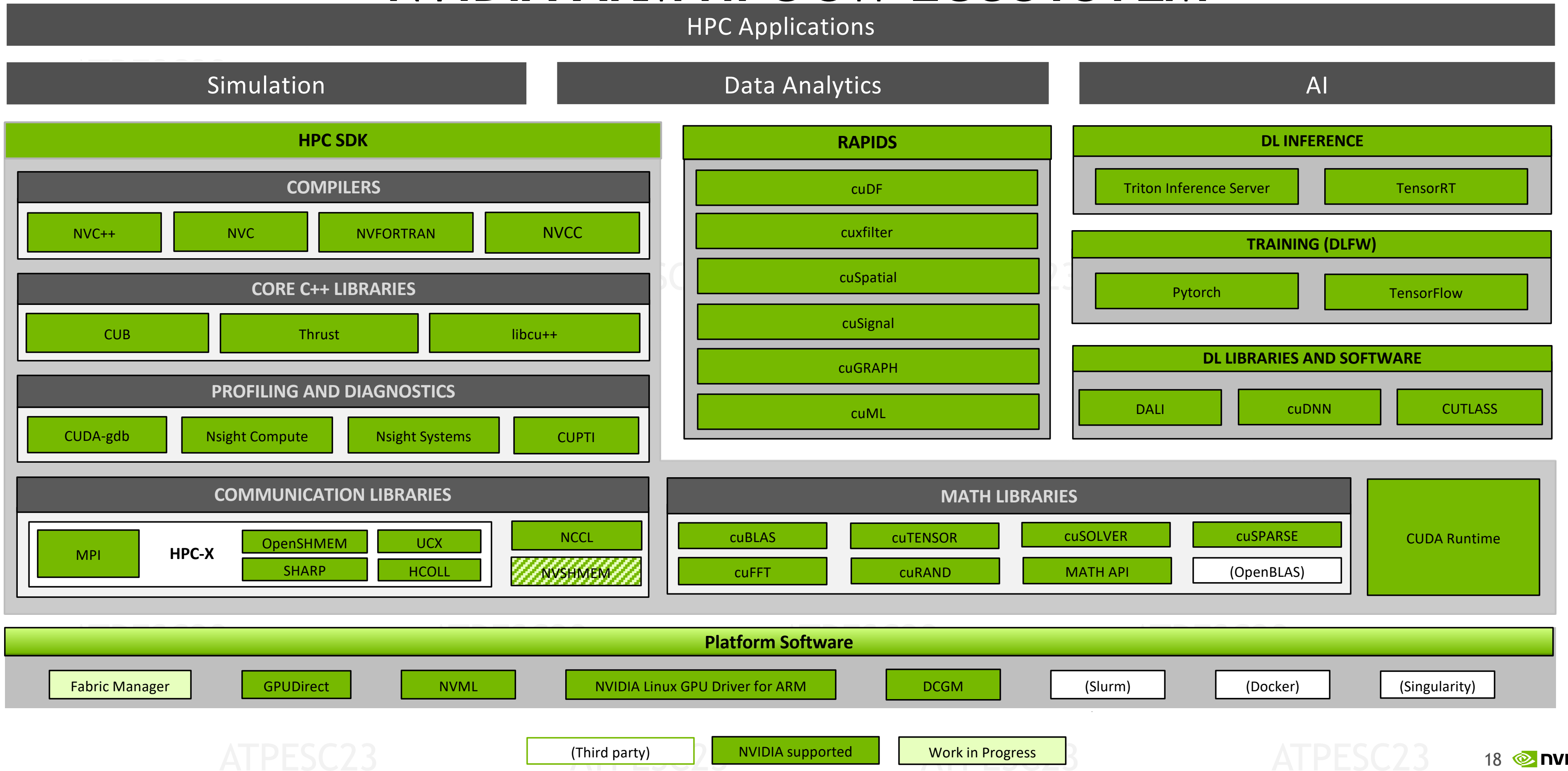
GRACE CPU SUPERCHIP

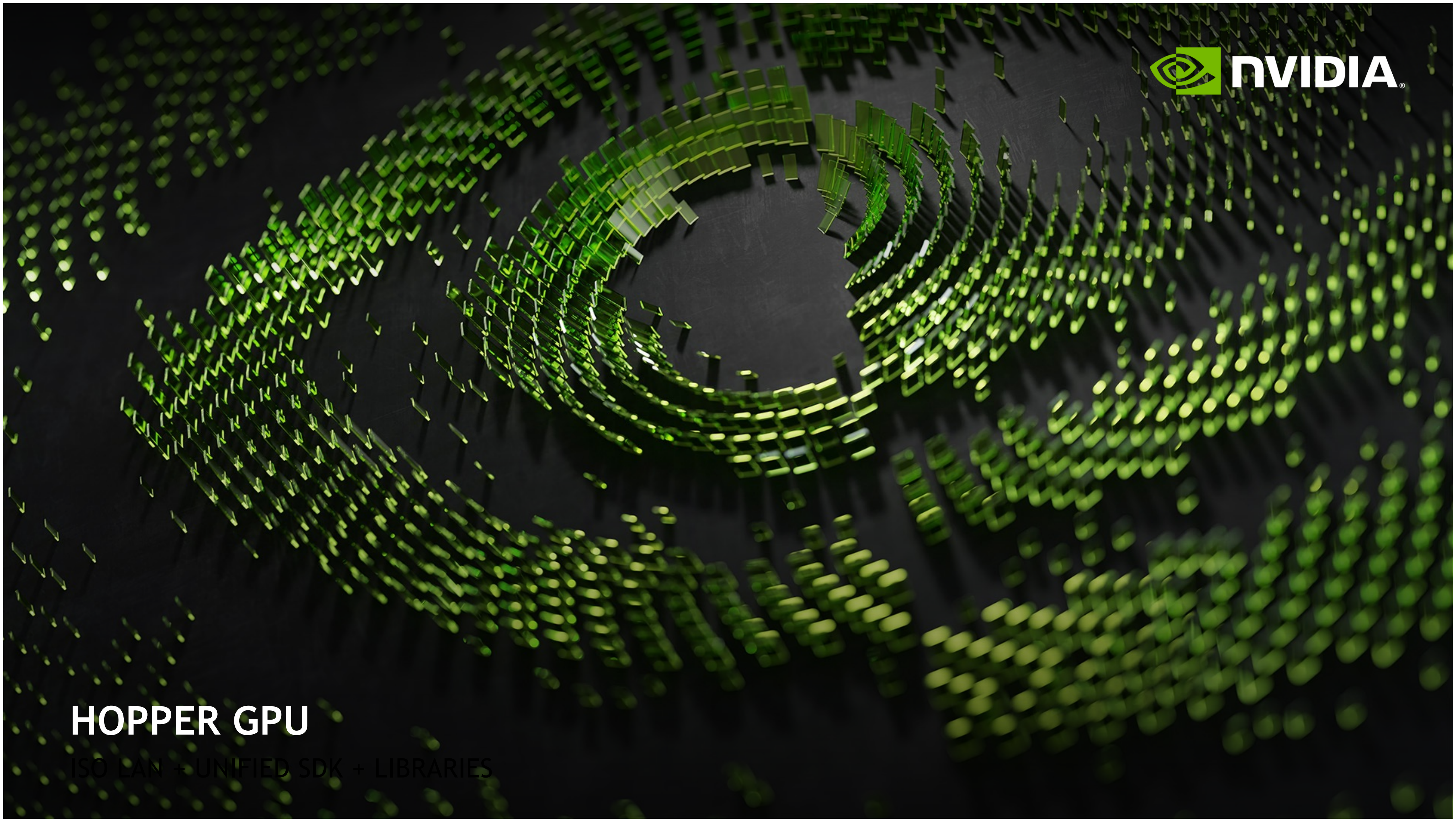
The Full Power of the Grace

Specifications	Grace SuperChip
Architecture	Armv9, SVE2 with 4x 128b pipeline/core
Cores / Speed	144 cores up to 3.2GHz
Memory	LPDDR5x soldered down, 1TB/s BW Up to 1TB per superchip
Cache	L1: 64KB i-cache + 64KB d-cache per core L2: 1MB per core L3: 234MB per superchip
Power	500W including LPDDR5x memory
Interfaces	Up to 8x PCIe Gen5 x16 HS interface
Specrate2017_int_base*	740
Process Node	TSMC 4N
Availability	Q4 2023

*GCC

NVIDIA ARM HPC SW ECOSYSTEM

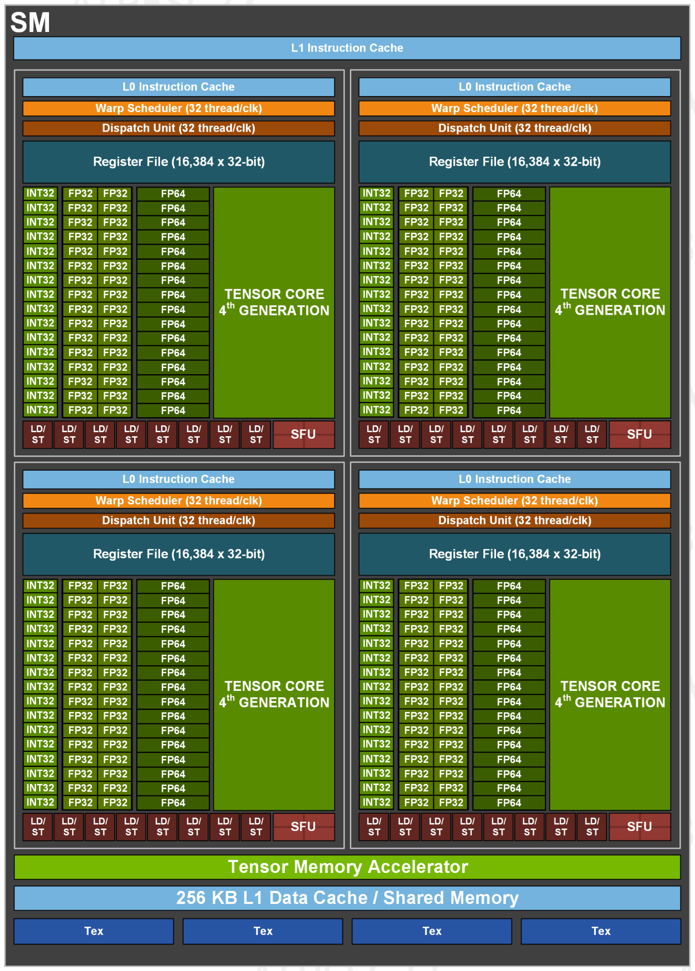




HOPPER GPU

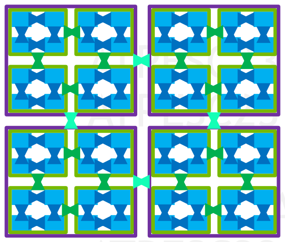
ISO LAN + UNIFIED SDK + LIBRARIES

NEW HOPPER SM DOES MORE THAN IMPROVE RAW SPEEDS AND FEEDS

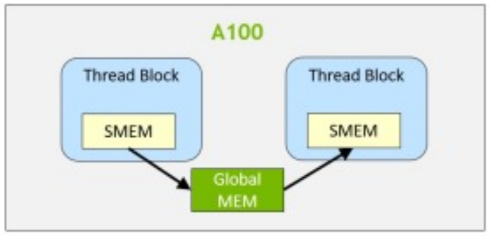

SM
L1 Instruction Cache
L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
INT32
FP32
FP64
TENSOR CORE 4th GENERATION
LD/ST
SFU
Tensor Memory Accelerator
256 KB L1 Data Cache / Shared Memory
Tex

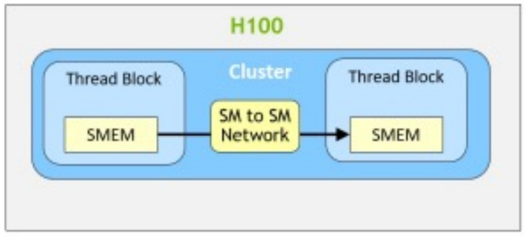
- New Thread Block Clusters
  - Turn locality into efficiency
  - Support Distributed Shared Memory between SMs
- New Asynchronous Transaction Barriers
  - Increased support for asynchronous programming
- New Tensor Memory Accelerator
  - Fully asynchronous data movement
- New DPX instruction set
  - Special Purpose Acceleration
- New Transformer Engine for AI Model Acceleration

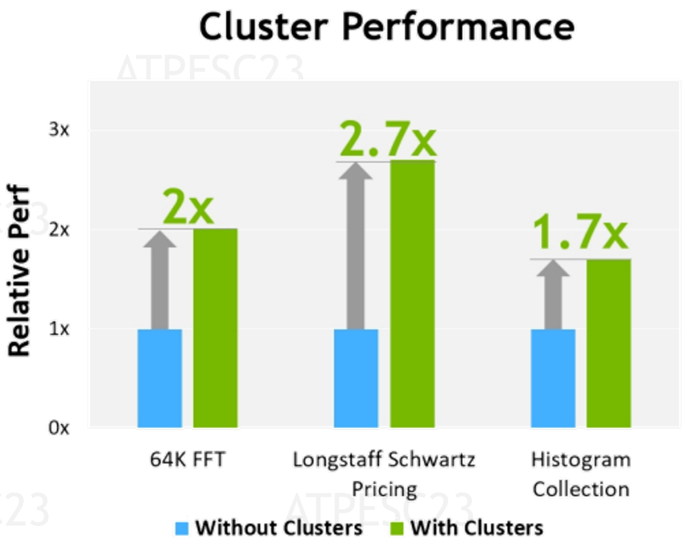
THREAD BLOCK CLUSTERS



- New feature introduces programming locality within clusters of SMs
- About 7X higher throughput vs. using global memory
- Shared memory blocks of SMs within a GPU Processing Cluster (GPC) can communicate directly (w/o going to HBM)
- Leveraged with CUDA cooperative groups API





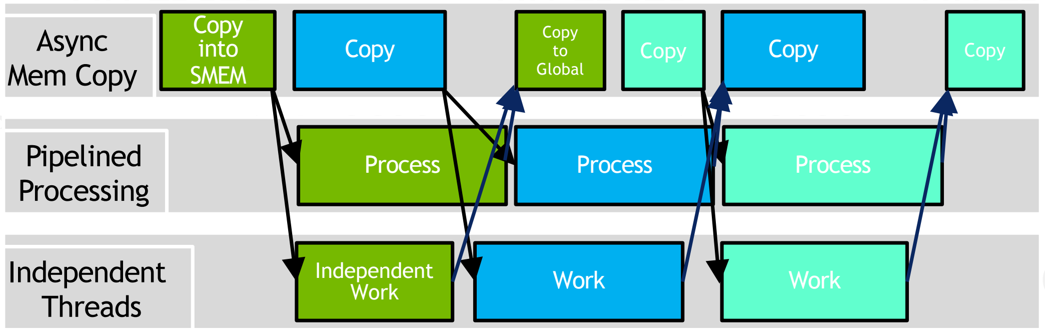


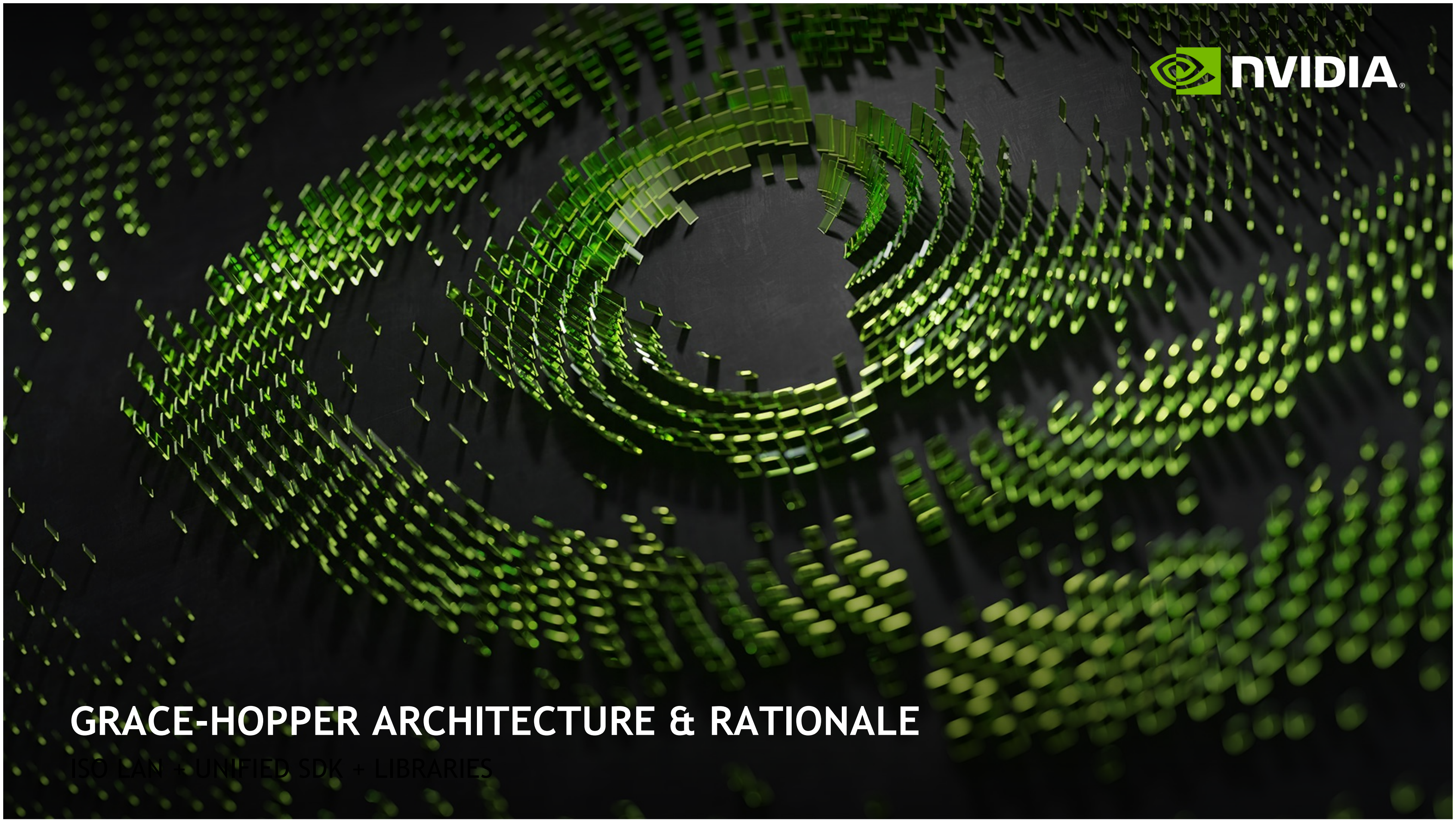
For details, see "NVIDIA H100 Tensor Core GPU Architecture" white paper available for download

ASYNCHRONOUS ENHANCEMENTS

Hopper enables end-to-end fully asynchronous pipelines

- Async Transaction Barriers – Atomic data movement with synchronization
- More efficient Waiting on Barriers
- Async Mem_copy via Tensor Memory Accelerator (TMA)





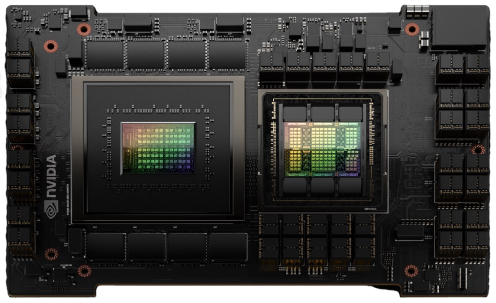
GRACE-HOPPER ARCHITECTURE & RATIONALE

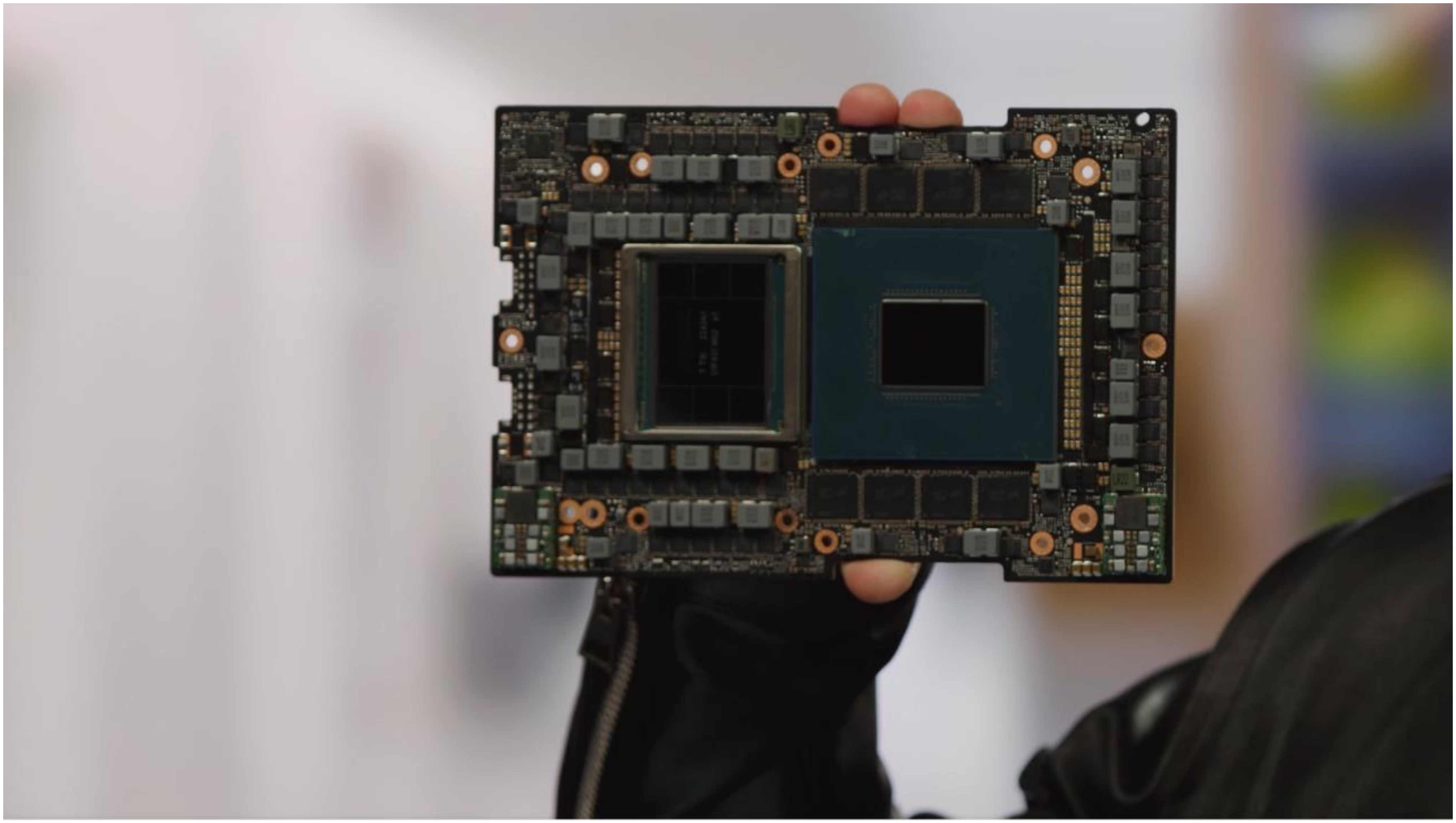
ISO LAN + UNIFIED SDK + LIBRARIES

GRACE-HOPPER

A revolutionary Architecture

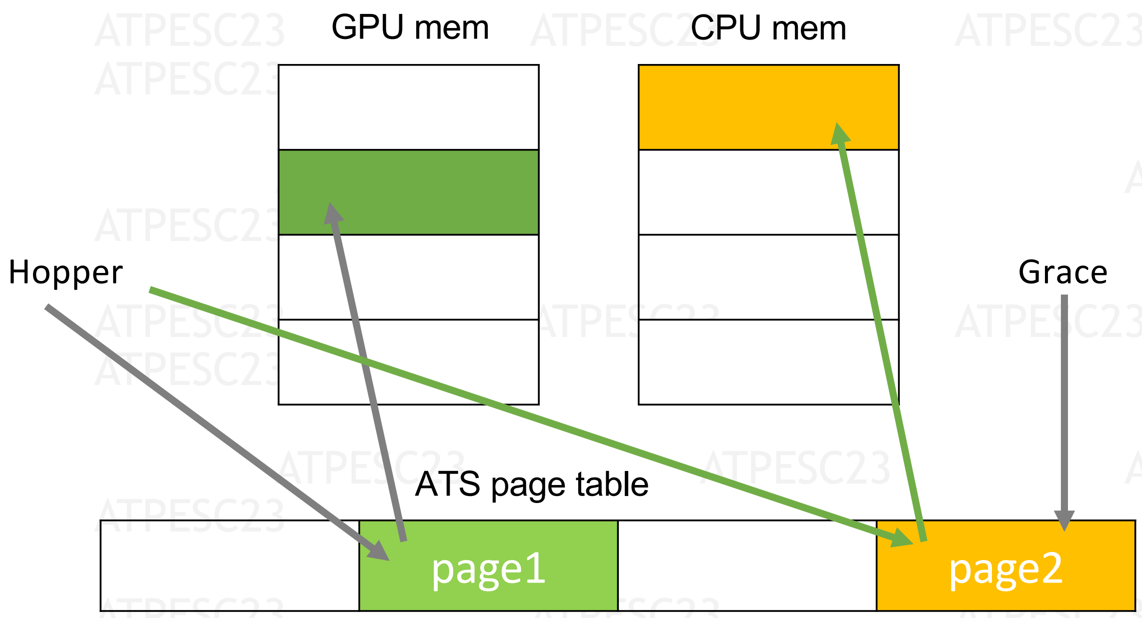
- Nvidia GPUs
  - Latency hiding Throughput Machines => Async Computational Graph solvers
    - Can effectively map ***Dataflow-Complex*** portions of the Algorithm
  - Custom (compute and memory) IP Blocks for energy efficiency
- Nvidia CPUs
  - SuperScalar, OOO Core based tightly coupled SoC with balanced Bytes/s/FLOPS
  - Can effectively map ***ControlFlow-Complex*** portions of the Algorithm
  - Strong Vector and Tensor performance in future
- Unified Compute Substrate





GRACE HOPPER

Unified Programming Model



- Address Translation Service (ATS) enables all CPUs and GPUs in the node to share a single page table
- System-allocated memory is accessible by all CPU and GPU threads
- Runtime system backs system-allocated memory with physical memory on first touch, either on LPDDR5 X or HBM3, depending on whether a CPU or a GPU thread accesses it first .

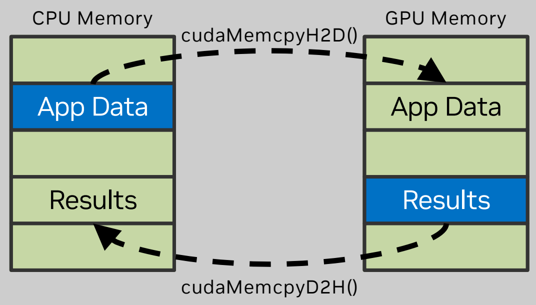
Grace-Hopper Memory Model

Full CUDA support with additional Grace memory extensions

Explicit Copy

Application explicitly moves data between CPU & GPU as needed

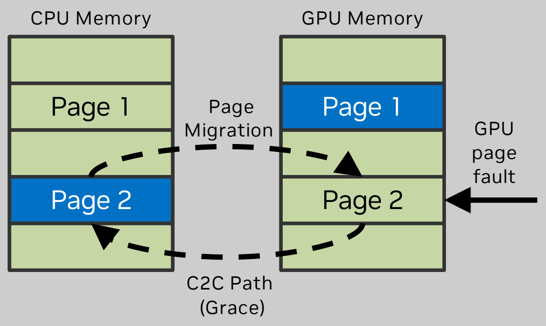
PCIE: ~60 GB/s PCIE transfers (H2D/D2H)
Grace: Faster transfers; up to 450 GB/s C2C transfers



Managed Memory

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

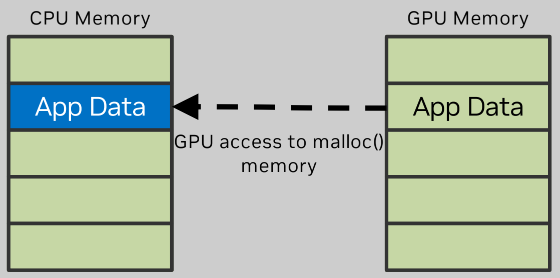
PCIE: Requires migration to GPU
Grace: Migrations not required and faster migrations when they happen



System Allocated

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

PCIE: Access possible with explicit call to `cudaHostRegister()` at PCIe speeds
Grace: `cudaHostRegister()` not needed; access at NVLink C2C speeds



GH NODE ARCH COMPARISON

- Summit vs DGX-A100 vs GH : ratio of CPU perf to GPU perf in a Node
- GH is the first Heterogeneous Compute Substrate
  - Reticle size CPU + Reticle Size GPU tightly coupled in a unified, coherent address space
- CPU-only codes, fully GPU acceleratable codes, Mixed controlflow complex/dataflow complex codes
  - With MPAM on CPU and MIG on GPU all codes can run on GH, GH-Next effficiently
  - Optimization of Applications, Runtime, Scheduler/resource manager are necessary

	Node Arch		
	Summit	DGX	Grace-Hopper
	P9+V100	Rome+4*A100	G+H
CPU Cores	22	64	72
CPU FLOPS (TFLOPS)	1.08	2.30	3.46
GPU FLOPS (TFLOPS)	21.00	78.00	51.00
CPU Mem BW (GB/s)	170.00	190.00	500.00
GPU Mem BW (GB/s)	2700.00	8156.00	3686.40
CPU <=> GPU BW (GB/s)	150.00	256.00	900.00
GPU/CPU FLOPS	19.43	33.85	14.76
GPU/CPU Mem BW	15.88	42.93	7.37
C<=>G link BW to FLOPS ratio	7.14	3.28	17.65

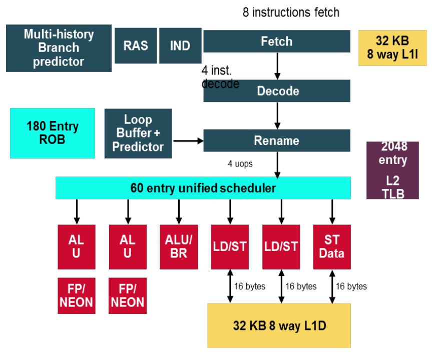
Application Complexity & Architectural Flexibility

Xeon

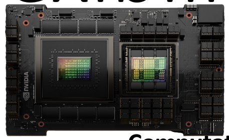
Grace

Hyperscale

Web Servers, Search and Data Analytics



Hyperscale & HPC



Grace-Hopper

Hybrid Data and Control Flow Architecture

Computational Complexity

Architectural Flexibility

pJ/OP

Highly irregular unstructured algorithms

Graph Analytics

Real time Data Analytics and Emerging Deep Learning

Hopper GPU

Complexity

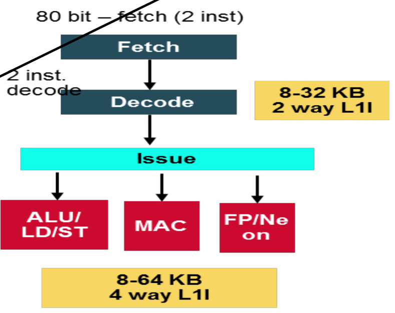
Data Flow

Complexity

HPC

Dense Matrix & Machine Learning

Xeon Phi and Pascal GPU



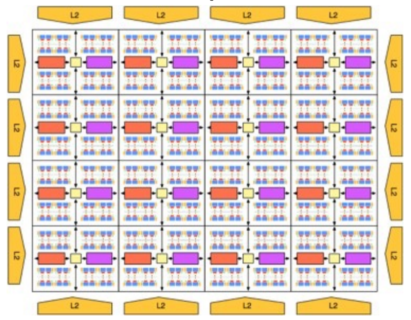
Control Flow

DataFlow Engine

TPU, Graphcore

Embedded Computation

Networking Algorithms

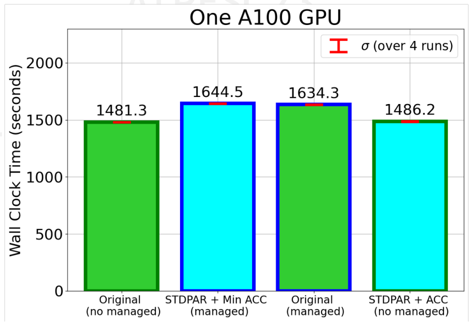


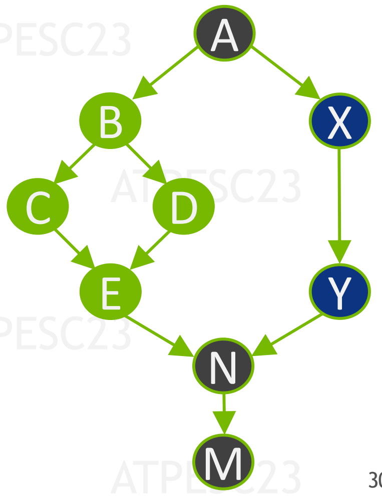
COMPUTATIONAL GRAPH + GRACE HOPPER

Same programming model for CPU and GPU, plus existing applications ready-to-run Day 1.

- True Unified Memory + High Bandwidth Link make data movement less of a bottleneck
- Existing CPU programming models continue to work on Grace CPU with high performance

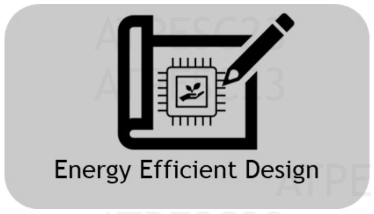
Senders/Receivers enable defining hybrid execution graphs to take advantage of the strengths of each processor.





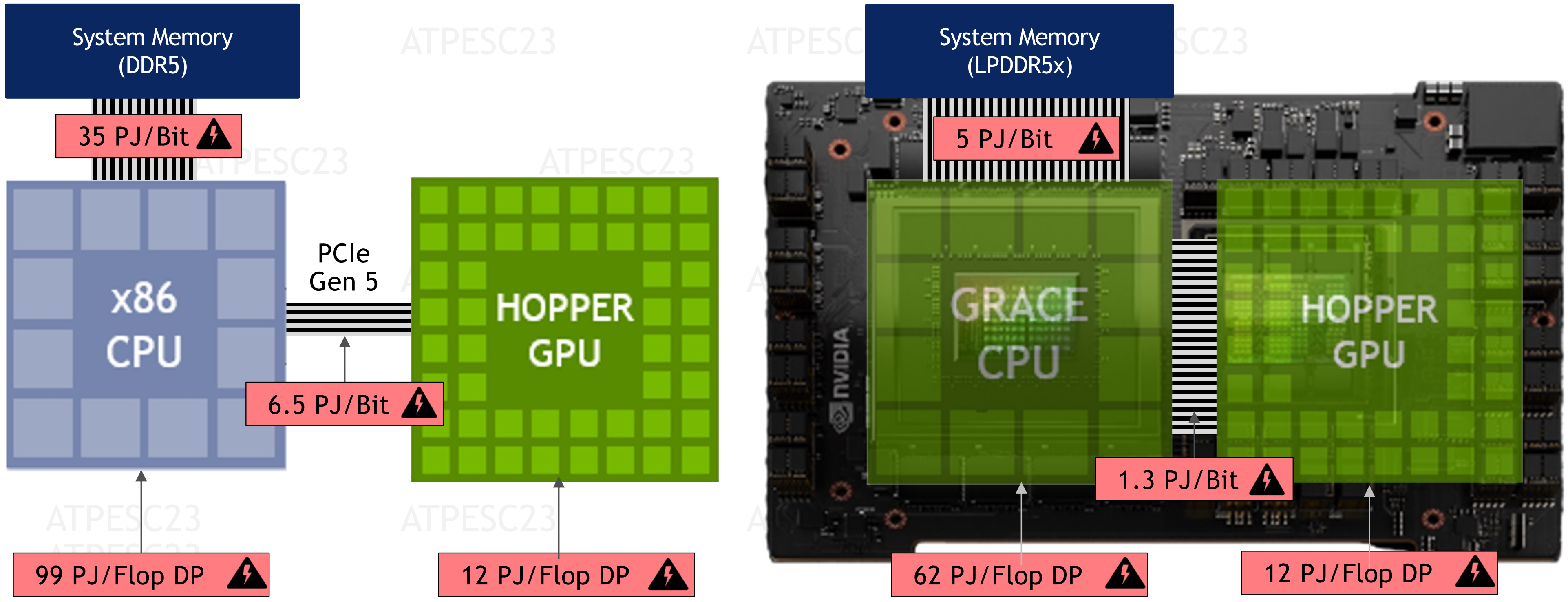
UNIFIED COMPUTE SUBSTRATE

- Grace + Hopper Enables broader set (all!) of codes to be accelerated
- Grace + Hopper enables architectural mapping of both control and data flow complex portions of the algorithm efficiently
    - Enables mapping of Multi-Scale, Multi-Physics Apps, post-Foundation AI and Complex workflows
- Standards-based parallelism enables productive portability
- Non-accelerated, fully-accelerated and mixed controlflow-dataflow complex applications can be run on Grace-Hopper
- Mapping these complex workflows to Distributed Heterogeneous Compute platforms require Omniverse like Digital Twin Frameworks



NVIDIA SUPERCHIPS SAVE ENERGY

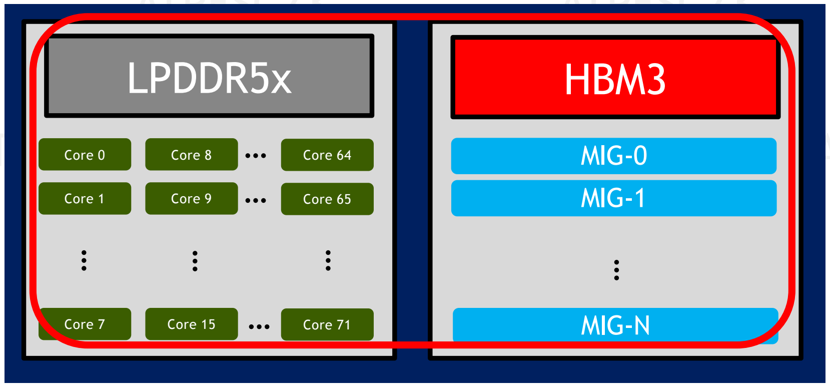
Low Power Data Motion and Computation

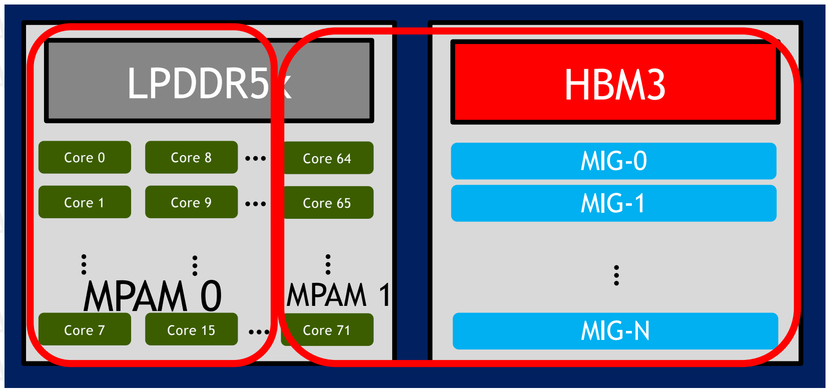


CO-SCHEDULING

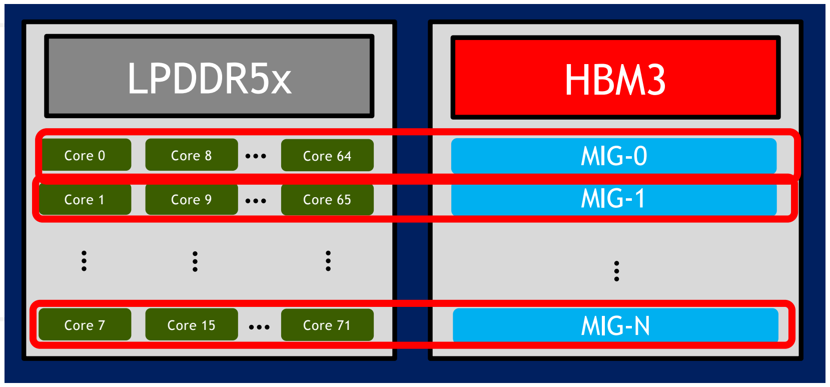
Alternative to CPU-only partition

One Job exclusive to the node





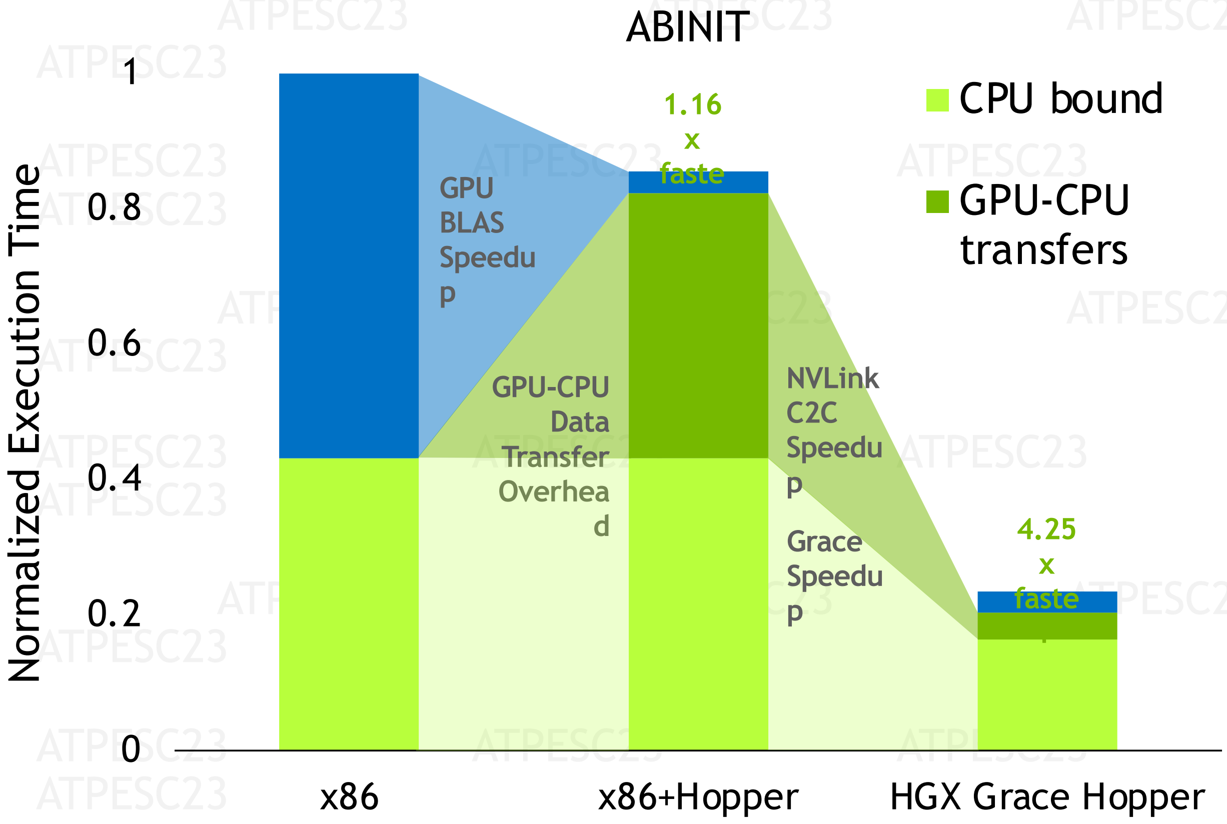
Job A - 64 Grace CPU MPAM
Job B - 8 Grace Cores MPAM + Hopper GPU



Job A - 8 Grace Cores MPAM + Hopper MIG
Job B - 8 Grace Cores MPAM + Hopper MIG
Etc.

GRACE+HOPPER MAKES ACCELERATION MORE ACCESSIBLE

Delivers Superior Performance and Efficiency for HPC



White Paper - Grace Hopper Superchip Architecture

A performance simulation for ABINIT with NVBLAS featuring Titanium 255 Atoms using the LOBPCG algorithm

Same Efficiency Definition for both Phys Sim and DL Sim:

“Given the problem size and required accuracy, time to solution, TCO to solution”

PARALLEL EXPRESSION

- Moving from Prescriptive to descriptive parallel expression
  - ISO-Language Parallelism
- Parallel Expression as a Computational Graph
- Decouple Scheduling from the expression of algorithm (aka Halide/XLA)
- OS/Runtime provide HW capabilities as hints to the scheduler
- Lowering through MLIR
- Scheduler/Runtime tools like XLA

CONCLUSIONS

HPC applications are becoming complex, Heterogeneous, data driven and ML/DL aided

Silicon Process Technology alone loosing steam

Architectural innovations, co-Design, ML/DL aiding the continuation of “Moore’s Observation”
Grace-Hopper architecture is a step in that direction

Accurate definition of performance is critical

Efficiently mapping Heterogeneous applications to heterogeneous HW is becoming complex
STD PAR, MLIR, XLA

Power and Energy are becoming the hard barriers to performance

GRACE, HOPPER WHITE PAPERS

- Hopper:
- https://resources.nvidia.com/en-us-tensor-core

- Grace:
- https://resources.nvidia.com/en-us-grace-cpu/nvidia-grace-cpu-superchip#page=1

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

- DGH-GH200 White paper:
- https://resources.nvidia.com/en-us-dgx-gh200/technical-white-paper

GTC 2023 Sessions to Watch

For more information on these topics

GTC23 Spring

- Asynchronous Acceleration in Standard C++ [S51755]
- A Deep Dive into the Latest HPC Software [S51074]
- Accelerating HPC applications with ISO C++ on Grace Hopper [S51054]
- How to Write a CUDA Program [S51210]
- cuNumeric and Legate: How to Create a Distributed GPU Accelerated Library [S51789]
- Connect with the Experts: C++ Standard Parallelism and C++ Core Compute Libraries [CWES52064]

NVIDIA Developer Blogs

- Developing Accelerated Code with Standard Language Parallelism
- Accelerating Standard C++ with GPUs
- Accelerating Fortran DO CONCURRENT
- Bringing Tensor Cores to Standard Fortran
- Accelerating Python on GPUs with NVC++ and Cython

Open-source codes

- LULESH - https://github.com/LLNL/LULESH
- STLBM - https://gitlab.com/unigehpfs/stlbm
- MiniWeather - https://github.com/mrnorman/miniWeather/
- POT3D - https://github.com/predsci/POT3D
- StdExec - https://github.com/nvidia/stdexec

C++ algorithms and execution policy reference

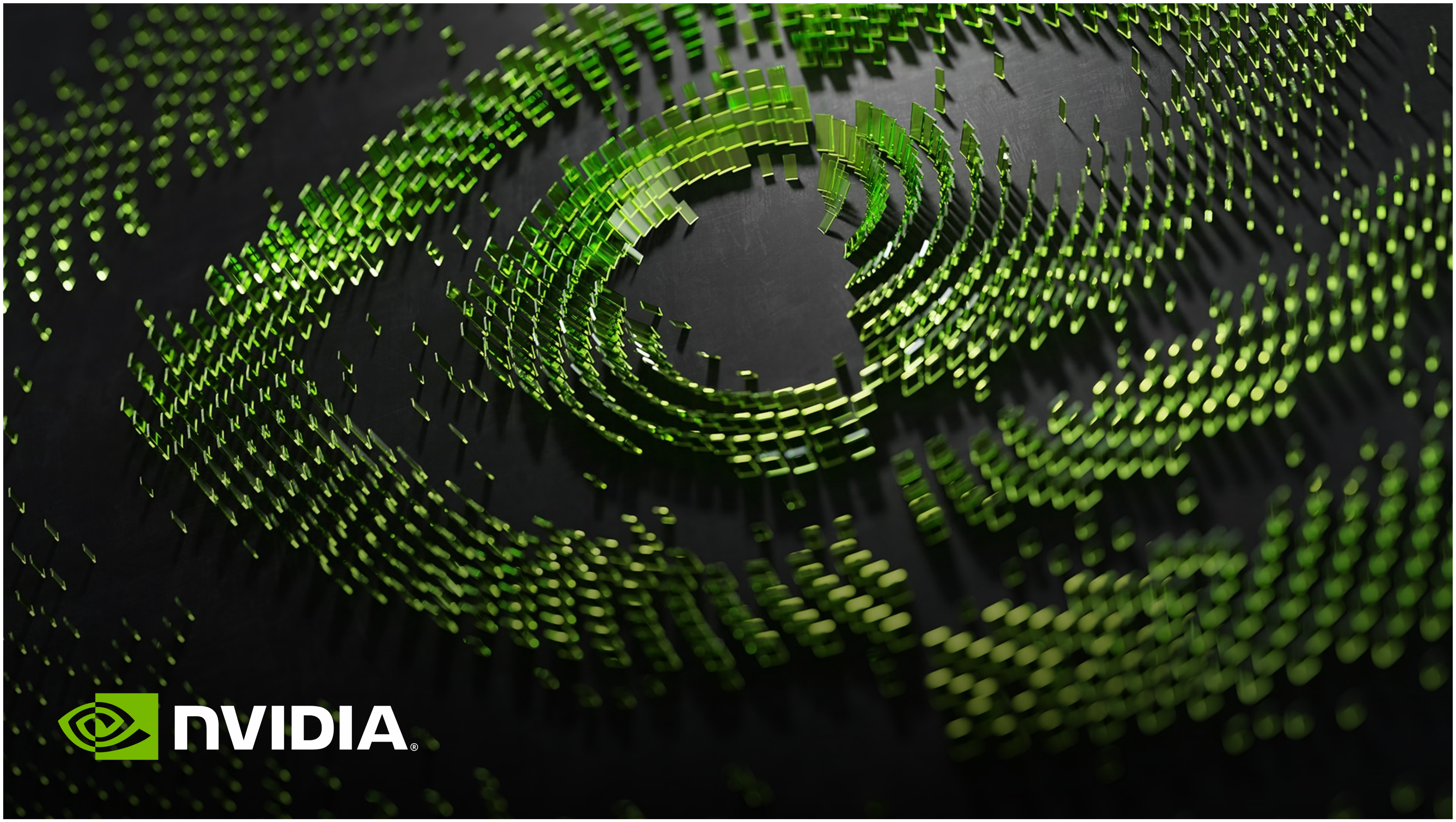
- https://en.cppreference.com/w/cpp/algorithm

NVIDIA HPC Compilers Forum

- https://forums.developer.nvidia.com/c/accelerated-computing/hpc-compilers

Legate and cuNumeric Resources

- https://github.com/nv-legate
- Accelerating Python Applications with cuNumeric and Legate


NVIDIA®