Modern Compute Architecture: From CPU to GPU

NRIS

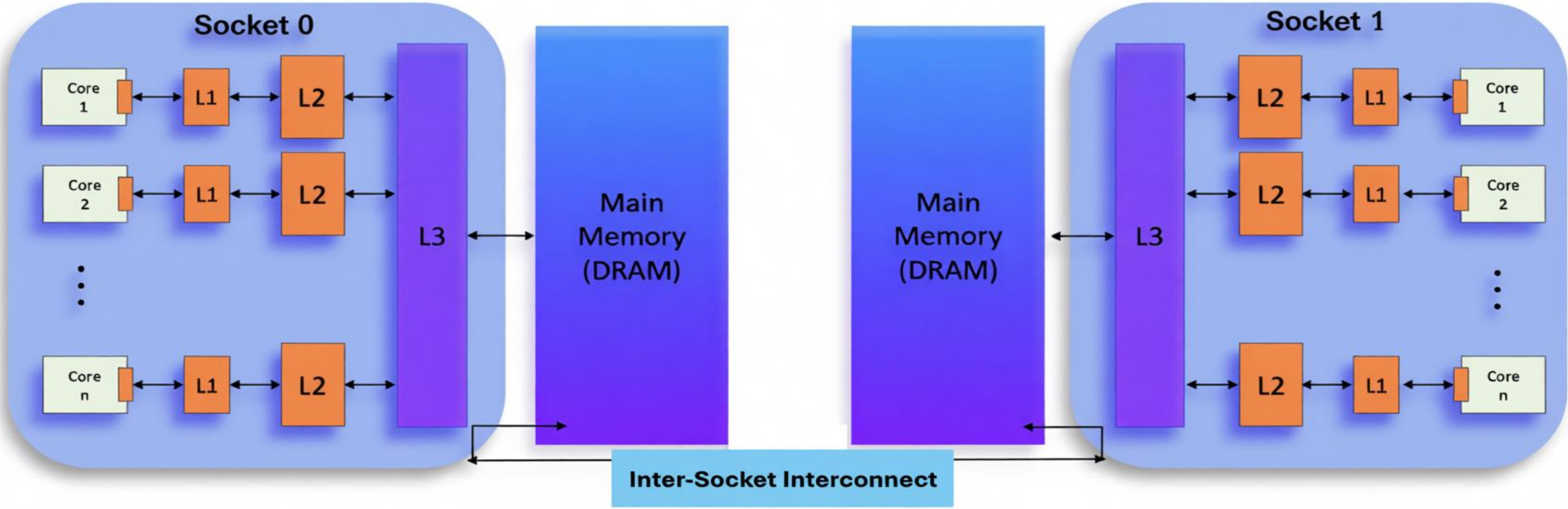
Norwegian research infrastructure services

Hicham Agueny, PhD
Scientific Computing Group
Info. Tech. Department, UiB/NRIS

Outline

- Why GPUs matter ?
- CPU & GPU Architecture Overview
- Evolution of GPUs
- Architecture of Modern GPUs: NVIDIA & AMD GPUs
- Software ecosystem of GPUs: CUDA & ROCm

Traditional CPU architecture



Adapted from *https://indico.cern.ch/event/814979/contributions/3401193/attachments/1831477/3105158/comp_arch_codas_2019.pdf*

AMD EPYC 7742, Zen 2 architecture, 2 sockets, 128 cores total.

	L1 cache	L2 cache	L3 cache
Size	32 KiB per core	256 KiB per core	256 MiB per socket
Latency	~4 cycles	~12–15 cycles	~35–45 cycles
Scope	Private per core	Private per core	Shared by 4 cores (per CCX)

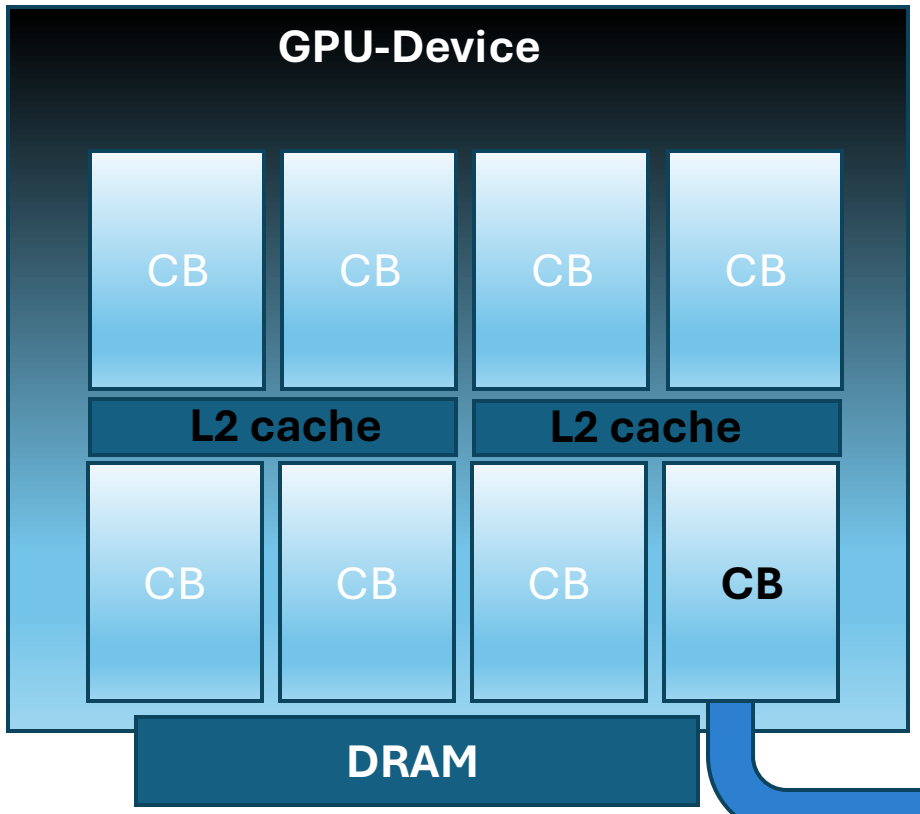
Traditional CPU architecture

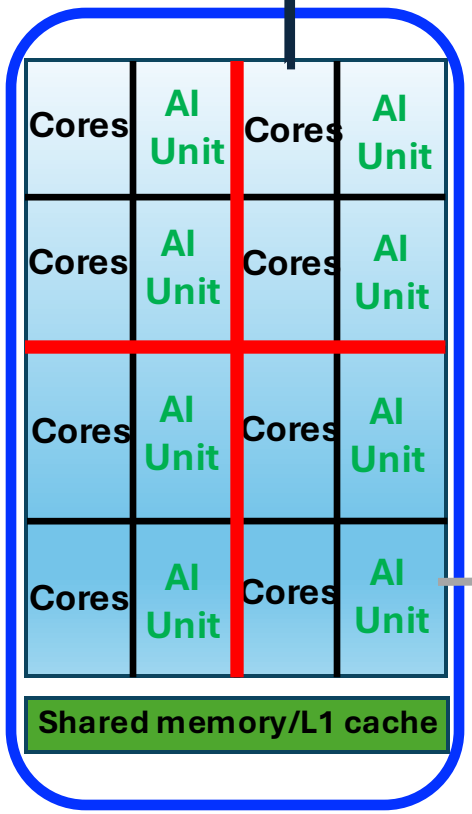
CPUs are designed for

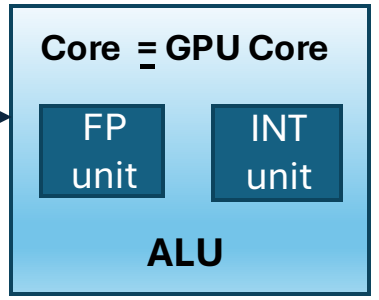
- Low latency (use caches L1/L2/L3)
- Responsive (fast change of instruction e.g. “if” block, loops with variable lengths)
- single-threaded performance (running one main task faster, even with **irregular steps**)

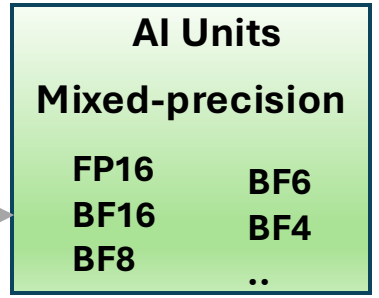
The CPU design becomes less efficient when processing millions of simple operations in parallel.

Simplified GPU architecture









Compute Block (CB)

All threads share L1 cache.

- Core includes ALU. Various logical operations for computations (FP32, FP64) e.g. addition, multiplication and ...
- ALU executes SIMT/SIMD instructions. (processing multiple data with the same operation concurently (in parallel))

CPU vs GPU

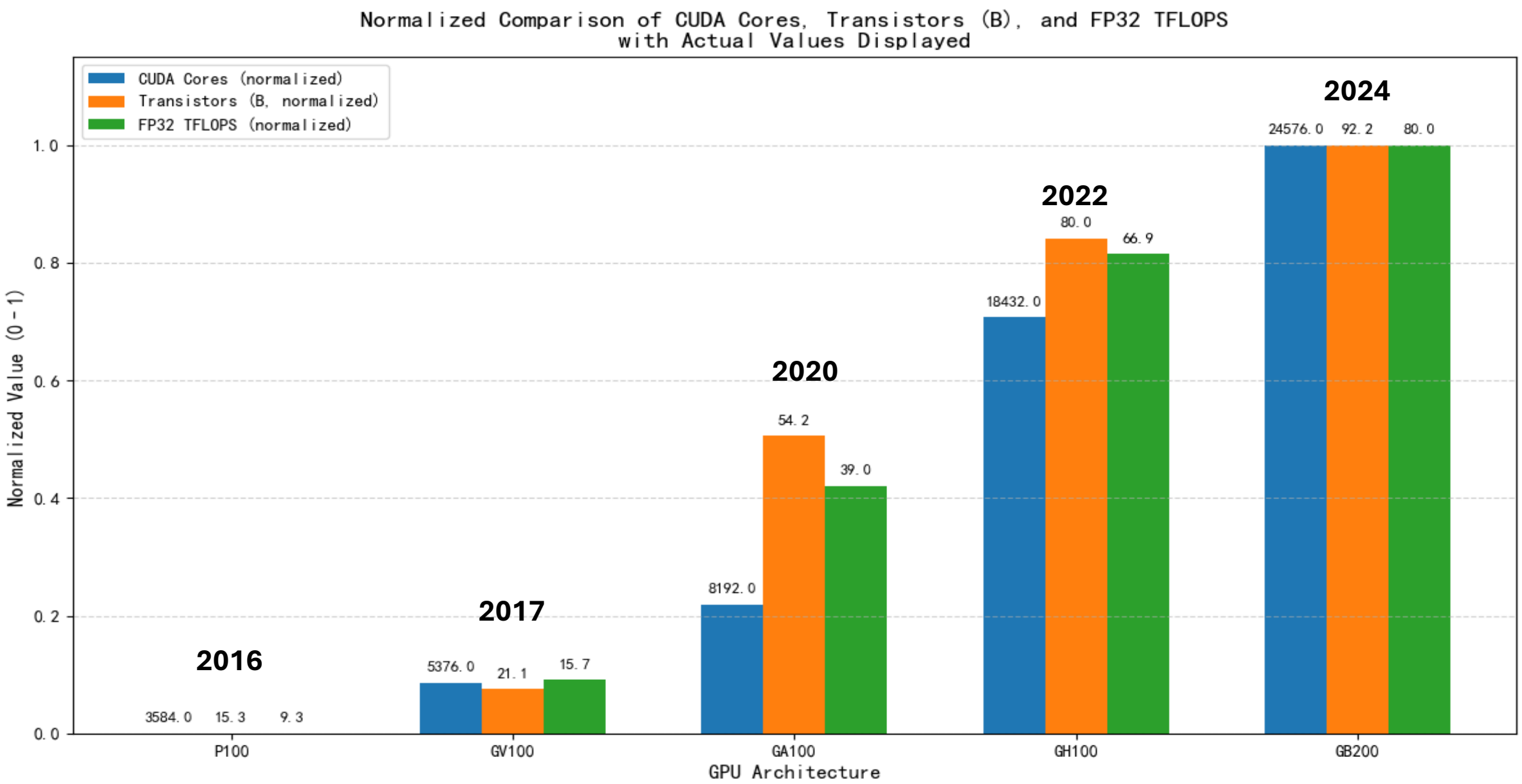
Traditional CPUs built for:

- Low latency
- Complex control flow (if, loops, function calls)
- Single-thread speed
- Not ideal for massive parallelism

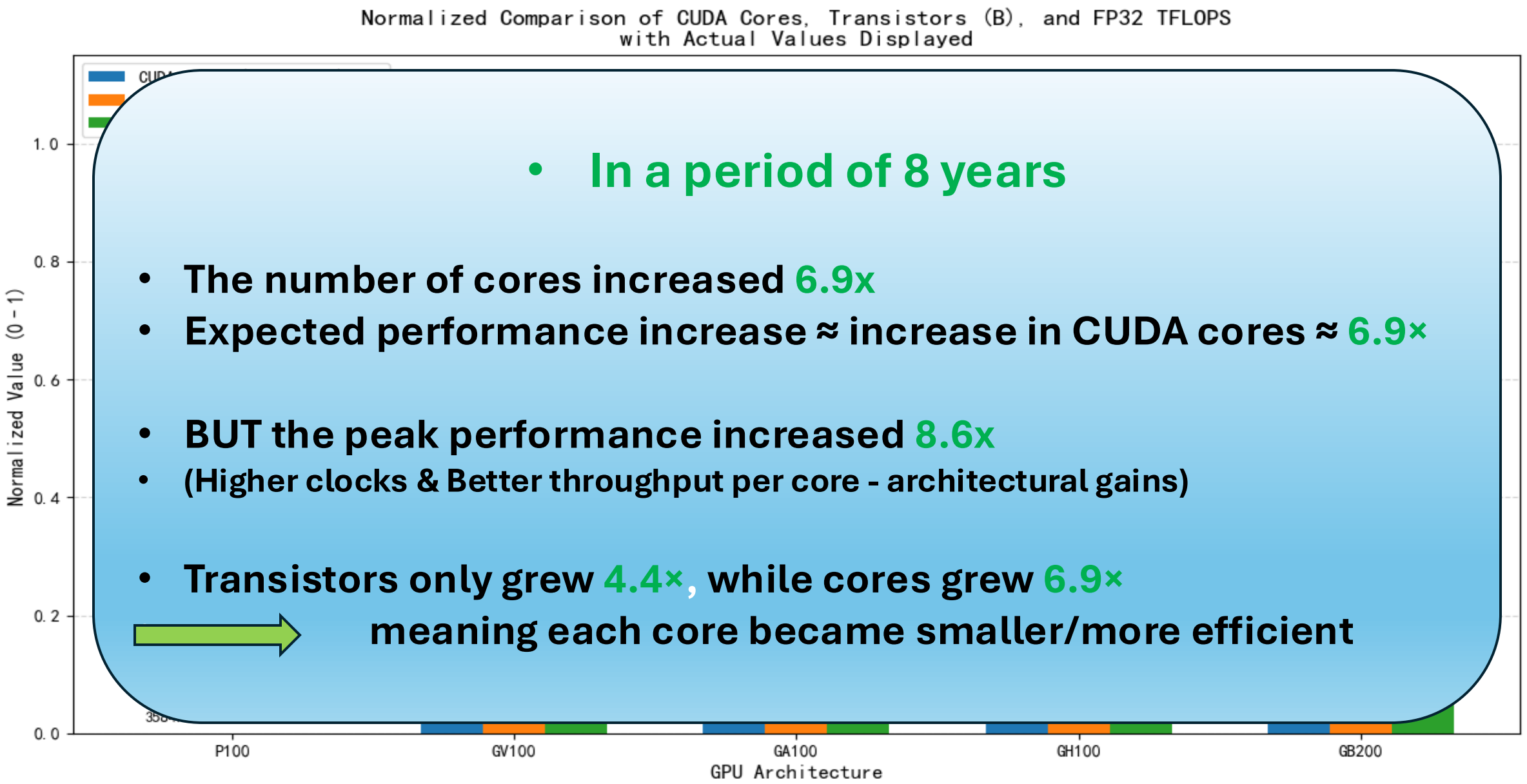
Modern GPUs built for:

- High throughput
- Thousands of simple, synchronized threads
- Regular, data-parallel workloads (SIMT/SIMD)
- Not ideal for branching or irregular code

Evolution of GPUs



Evolution of GPU



- **In a period of 8 years**

- **The number of cores increased 6.9x**
- **Expected performance increase ≈ increase in CUDA cores ≈ 6.9×**

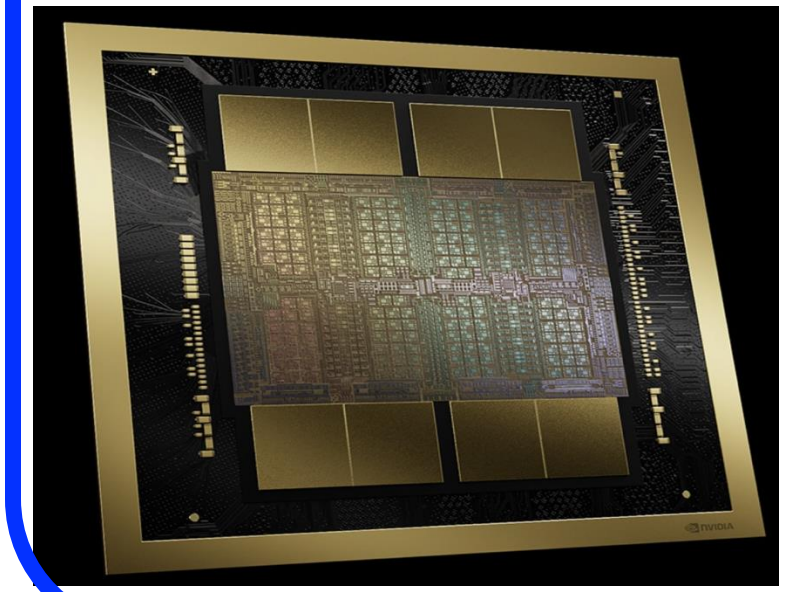
- **BUT the peak performance increased 8.6x**
- **(Higher clocks & Better throughput per core - architectural gains)**

- **Transistors only grew 4.4×, while cores grew 6.9×**

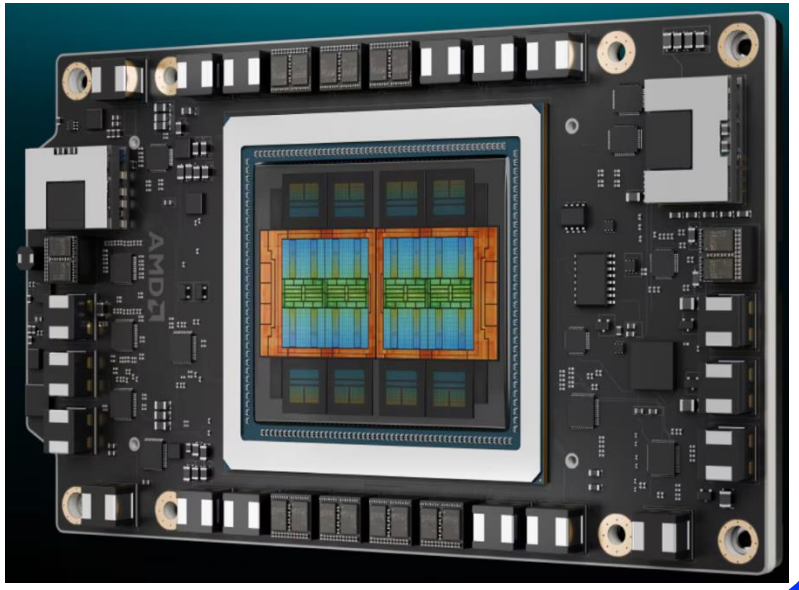
meaning each core became smaller/more efficient

Modern GPU architectures

NVIDIA Blackwell GPU



AMD Instinct MI350 GPUs



Intel® Arc™ Pro B60 Graphics

NVIDIA Blackwell GPU	*https://www.nvidia.com/en-us/data-center/technologies/blackwell-architecture/*
AMD MI350 GPU	*https://www.amd.com/en/products/accelerators/instinct.html*
Intel Arc B60 GPU	*https://www.intel.com/content/www/us/en/products/docs/discrete-gpus/arc/workstations/b-series/overview.html*

Modern GPU architectures

- **Chiplet-Based GPU Design**
  Multi-die GPU with high-bandwidth die-to-die (5.5-10 TB/s)

- **Next-Generation Matrix/AI Units**
  Enhanced AI Units optimized for mixed-precision: FP16/FP8/BF16
  Higher throughput per compute block compared to previous GPUs

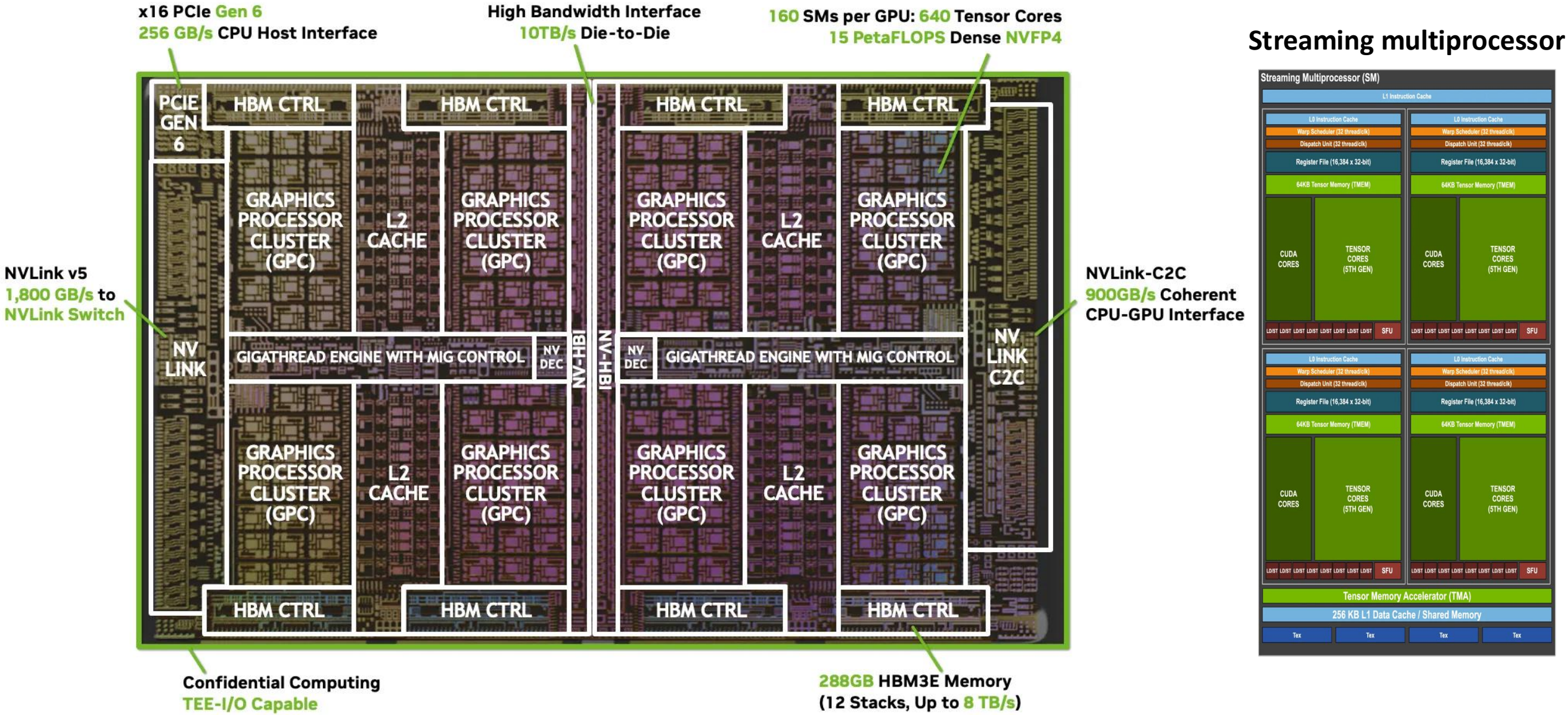
- **Scaled memory Bandwidth** (up to 8 TB/s)

- **Higher memory capacity** per GPU (up to 288 GB)

- **Advanced CPU-GPU Interconnects** with high-bandwidth (128-900 GB/s)

- **Advanced GPU-GPU Interconnects** with high-bandwidth (1.0-1.8 TB/s)

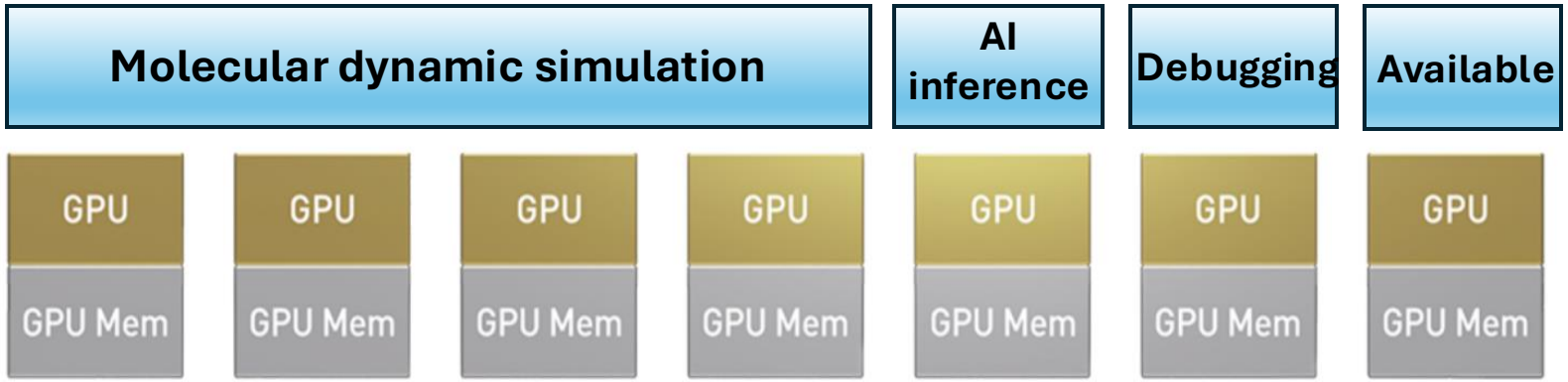
NVIDIA Blackwell Ultra GPU

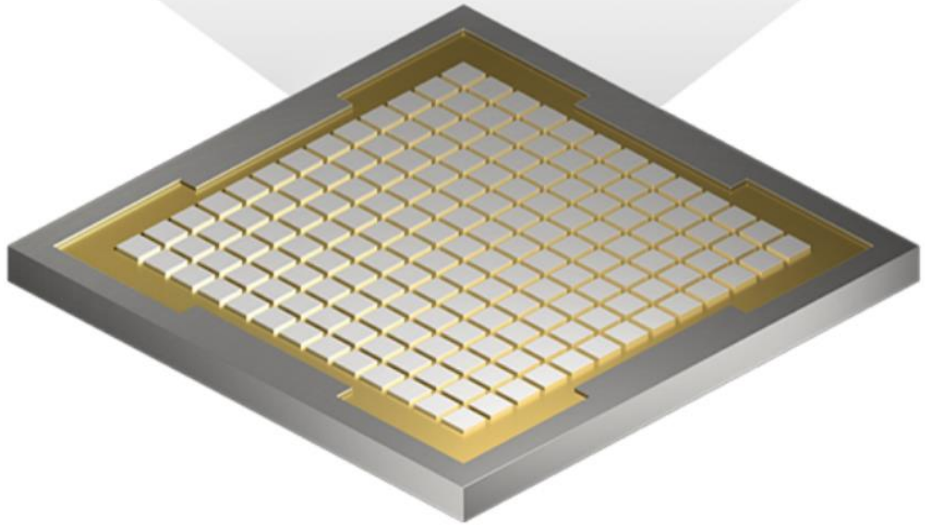


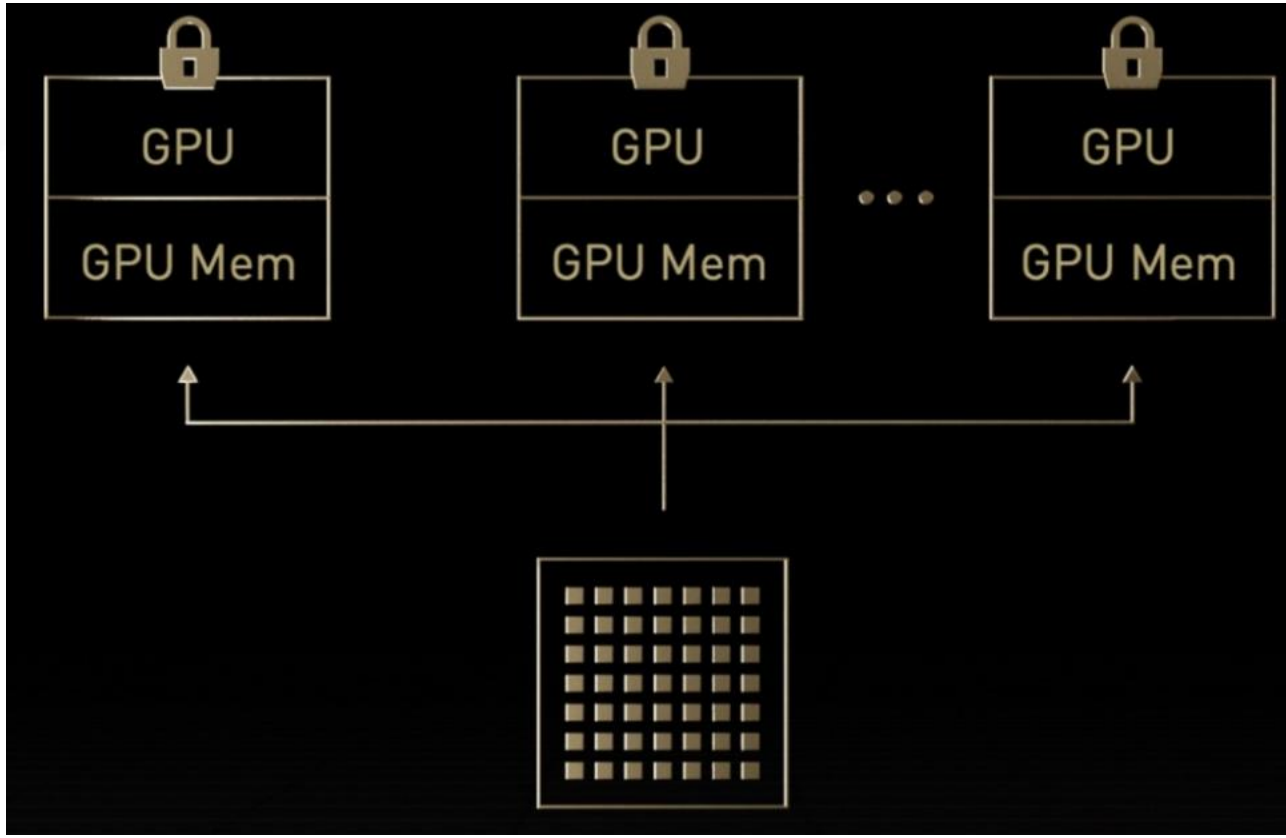
Streaming multiprocessor

https://developer.nvidia.com/blog/inside-nvidia-blackwell-ultra-the-chip-powering-the-ai-factory-era/

Multi-Instance GPU: NVIDIA GPU

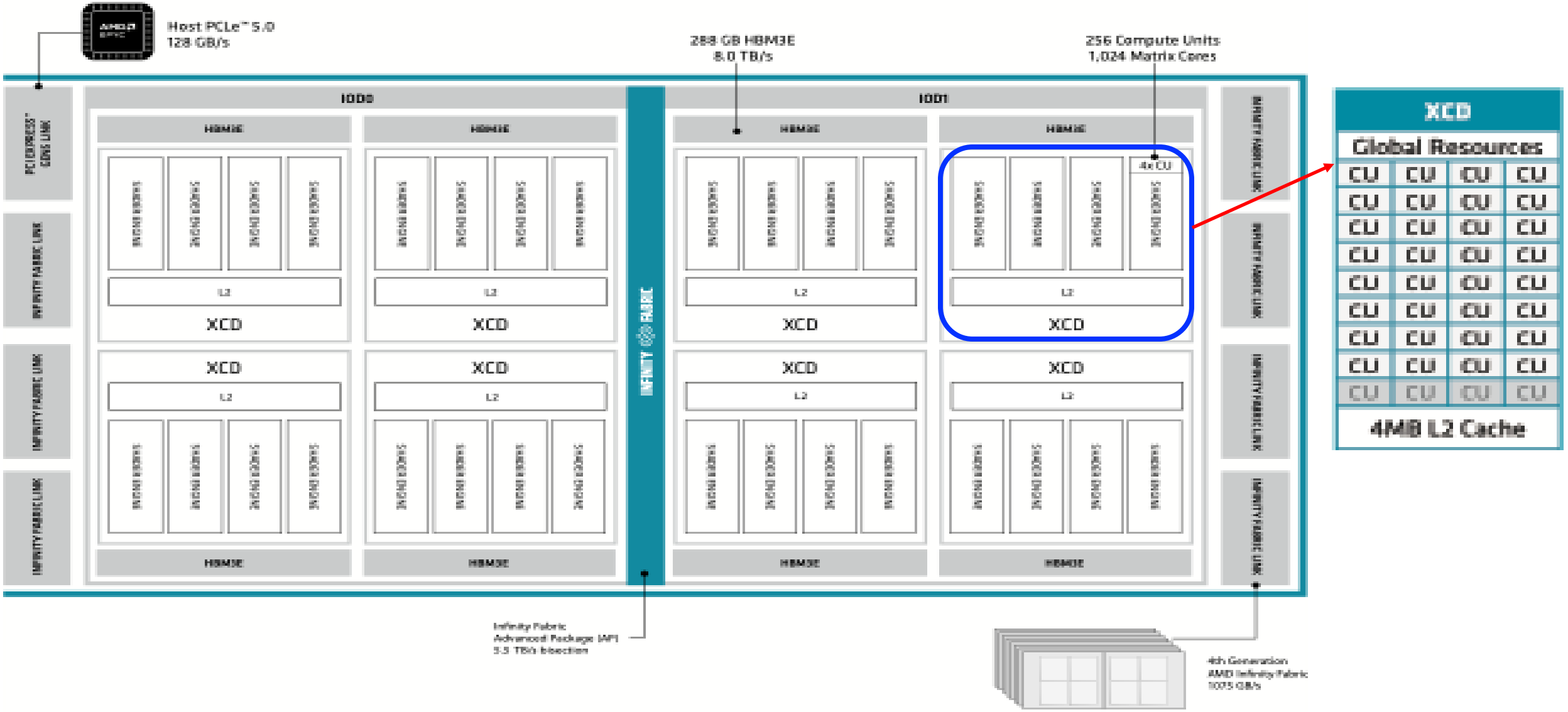






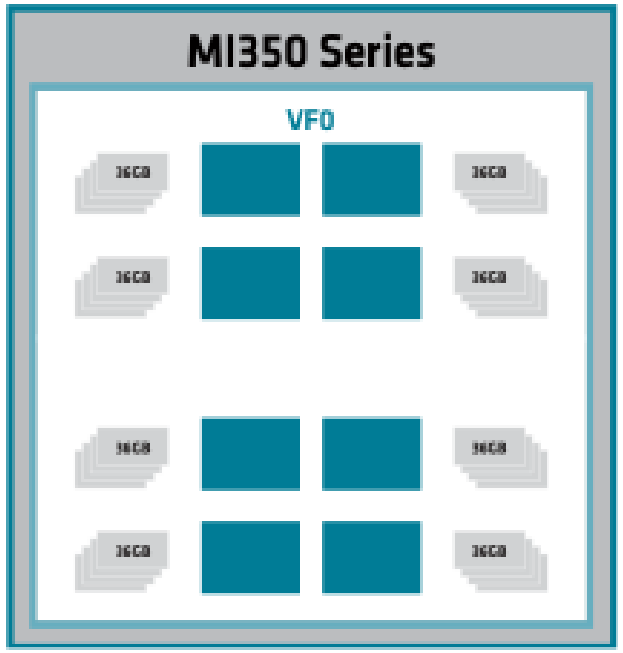
https://www.nvidia.com/en-us/technologies/multi-instance-gpu/

AMD Instinct MI350 Series GPU Multi-Die Chiplet (XCD)



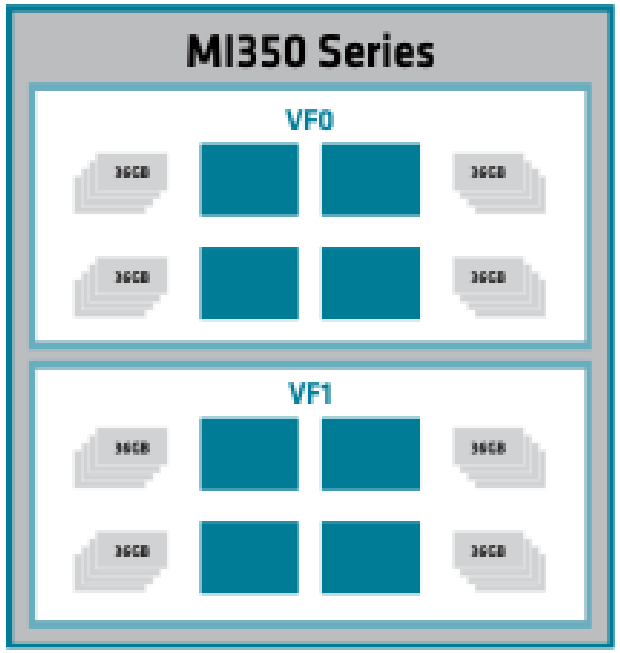
https://www.amd.com/content/dam/amd/en/documents/instinct-tech-docs/white-papers/amd-cdna-4-architecture-whitepaper.pdf

Compute Partitioning (or MIG): AMD GPU



(a) Single Partition: SPX, NPS 1

8 XCDs per Partition / 8 XCDs per VM,
288 GB per Partition / 288 GB per VM

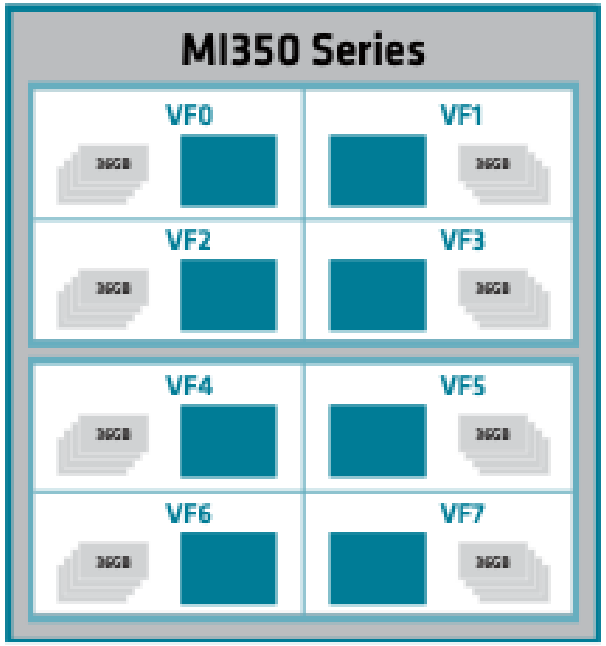


(b) Two Partitions: DPX, NPS 2

4 XCDs per Partition / 4 XCDs per VM,
144 GB per Partition / 144 GB per VM

(c) Four Partitions: QPX, NPS 2

2 XCDs per Partition / 2 XCDs per VM,
72 GB per Partition / 72 GB per VM



(a) Eight Partitions: CPX, NPS 2

1 XCDs per Partition / 1 XCDs per VM,
36 GB per Partition / 36 GB per VM

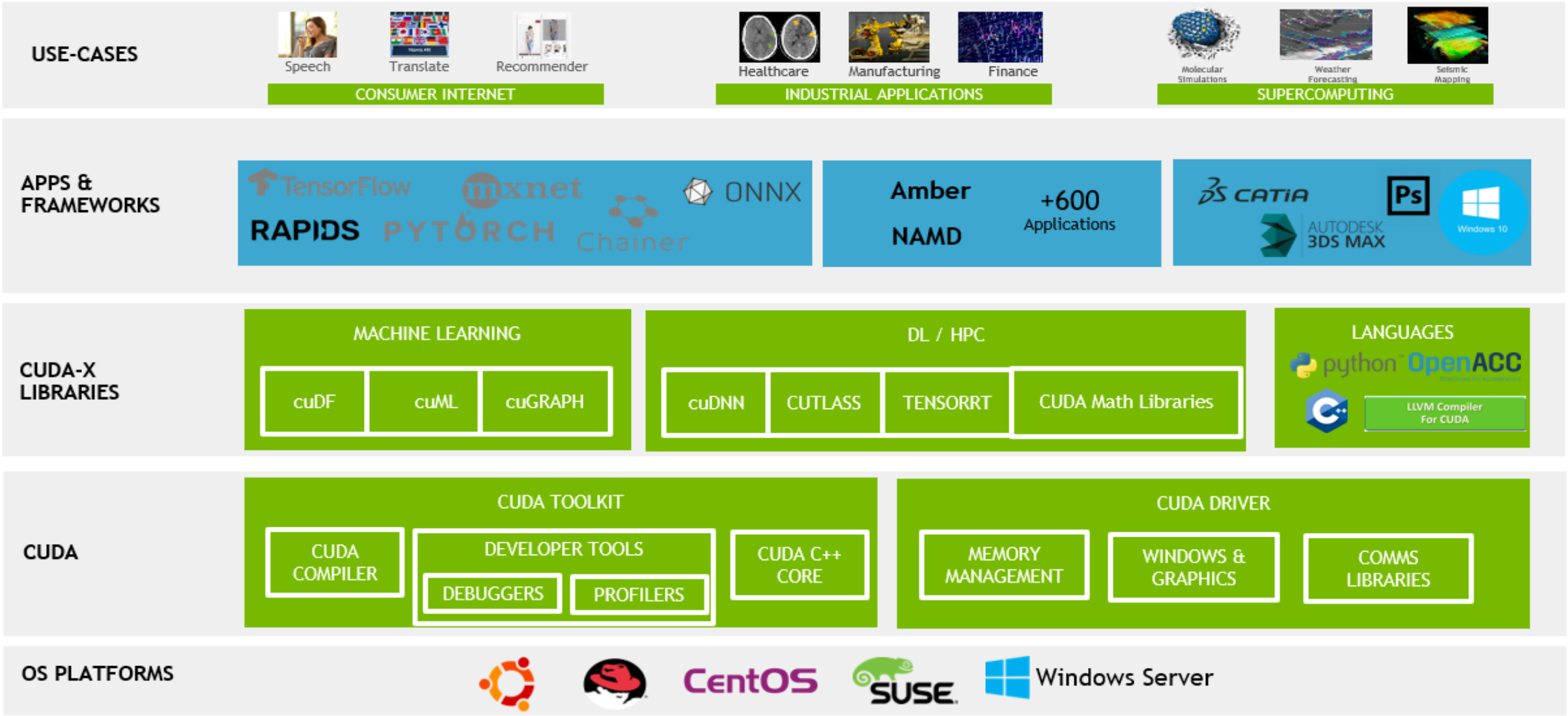
Key GPU Hardware Unit Terminology Comparison

Concept	NVIDIA	AMD		Explanation
Large GPU Compute Block	GPC (Graphics Processing Cluster)	XCD (Accelerator Compute Dies)		Handle parallel compute and graphics tasks via SMs/CUs.
Compute Block	SM (Streaming Multiprocessor)	CU (Compute Unit)		Contains ALUs, registers, schedulers.
GPU Core	CUDA Core	Stream Processor		Executes FP/INT operations.
AI / Matrix Acceleration Unit	Tensor Core	Matrix Engine		Accelerates matrix / AI computations. (mixed precision)
Thread	Thread	Work item/thread		Smallest independently scheduled unit of execution.
Thread Group Size	Warp = 32 threads	Wavefront = 32 RDNA: 32 or 64		Implement SIMT/SIMD, allowing one instruction to be executed across many threads/work-items at the same time.

Software ecosystem for GPU accelerators

- **CUDA** (created by NVIDIA)
- **ROCm** (created by AMD)

CUDA Software ecosystem



https://blogs.nvidia.com/blog/what-is-cuda-2/

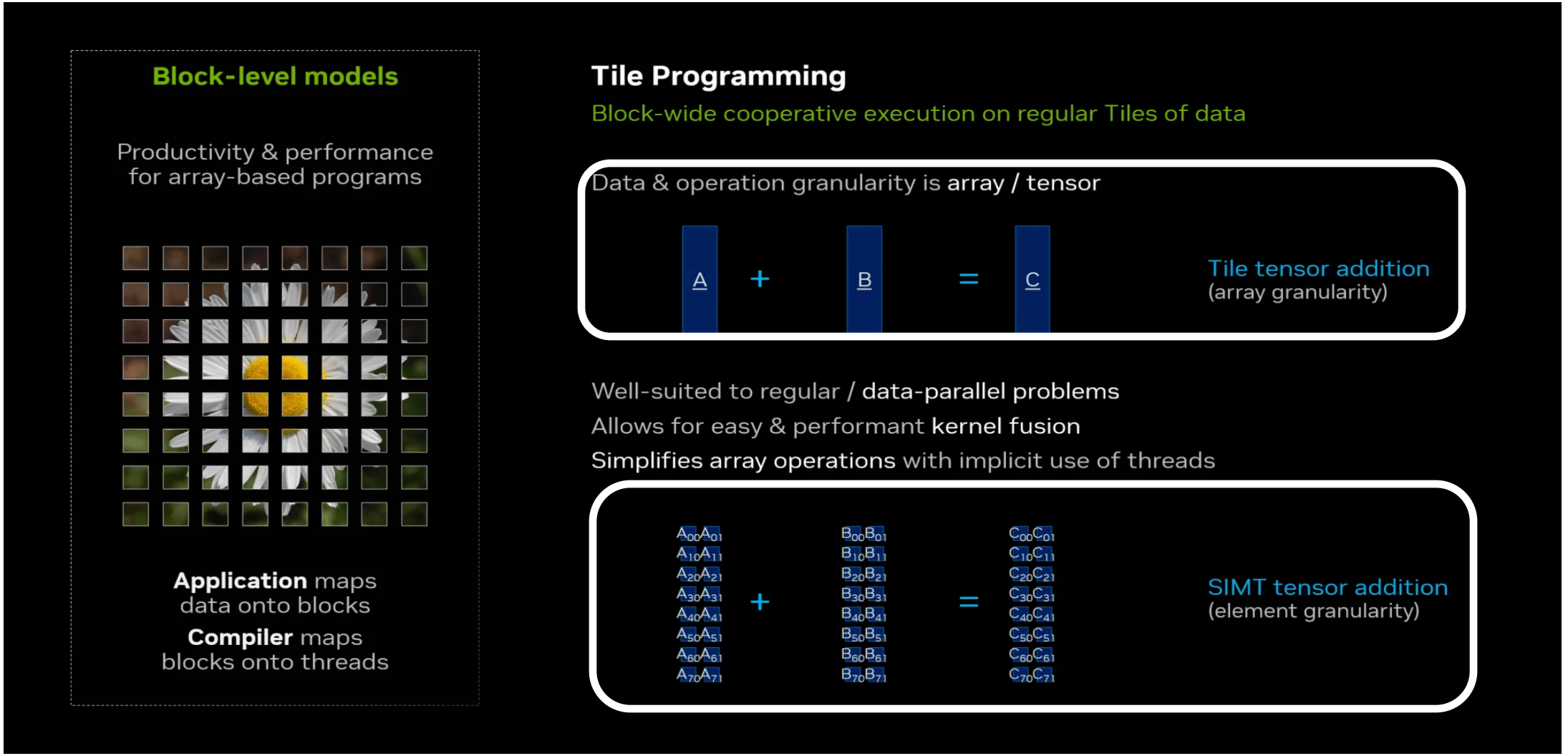
What is new in CUDA Toolkit: 13.0

- **Unified support for Arm CPUs**, enabling CUDA on broader platforms (e.g., Grace CPU systems).
- **Traditional CUDA**: Based on thread-parallel **SIMT**, developers manage low-level thread blocks and warps.
- **CUDA 13+:** Introduces **tile-based programming**, a higher-level abstraction for GPU acceleration.
- **Compiler & runtime** now handle thread scheduling and hardware optimization automatically.
- **Developers focus on logic**, not threads, write less repetitive code, achieve full performance.
- **Tiles supported in Python, C++,** and more, accessible across languages.

https://developer.nvidia.com/blog/whats-new-and-important-in-cuda-toolkit-13-0/

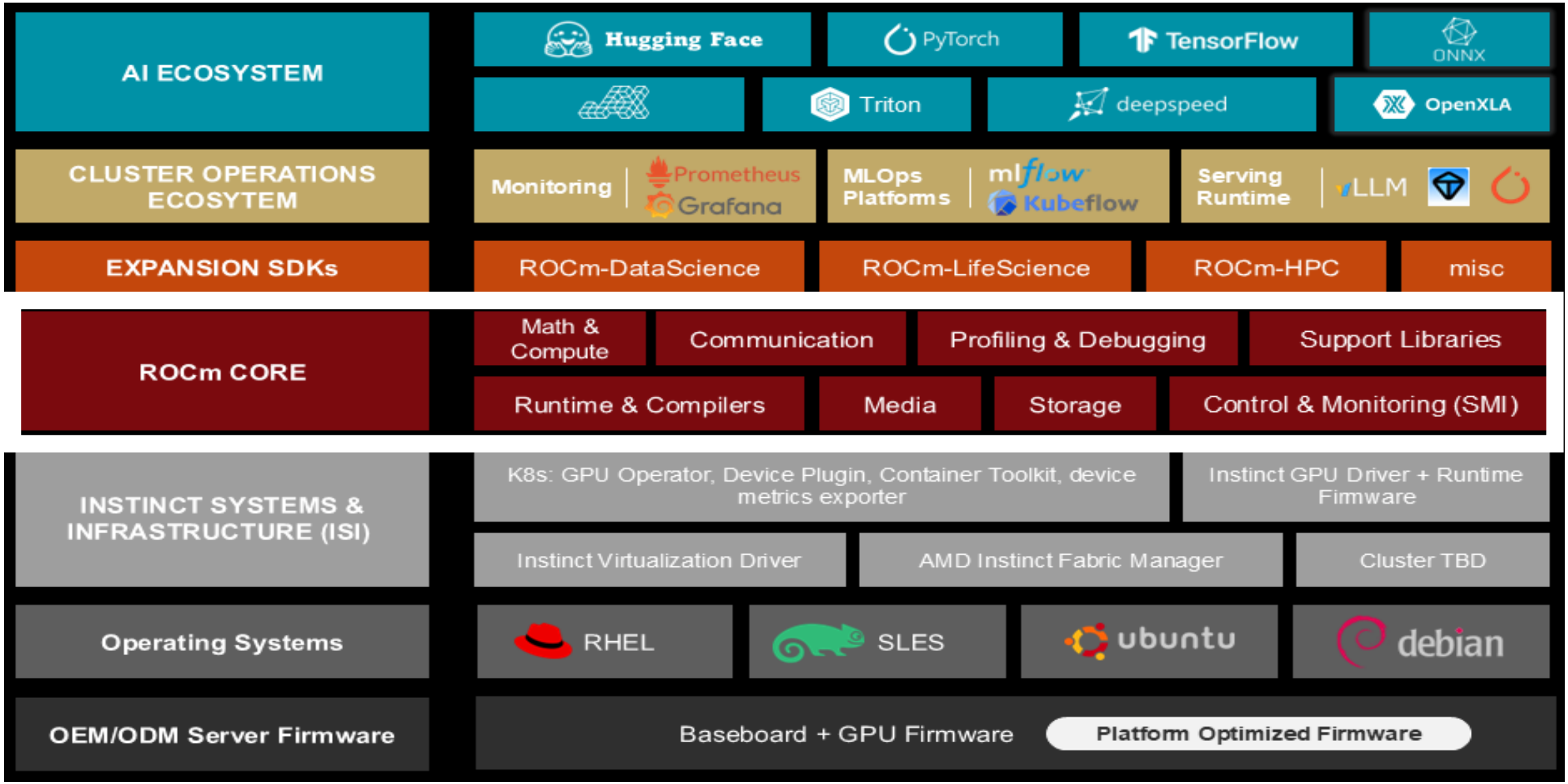
What is new in CUDA Toolkit: 13.0

Tile-based programming: define tiles (array) of data and specify operations over those tiles. Complier handles the low-level execution.



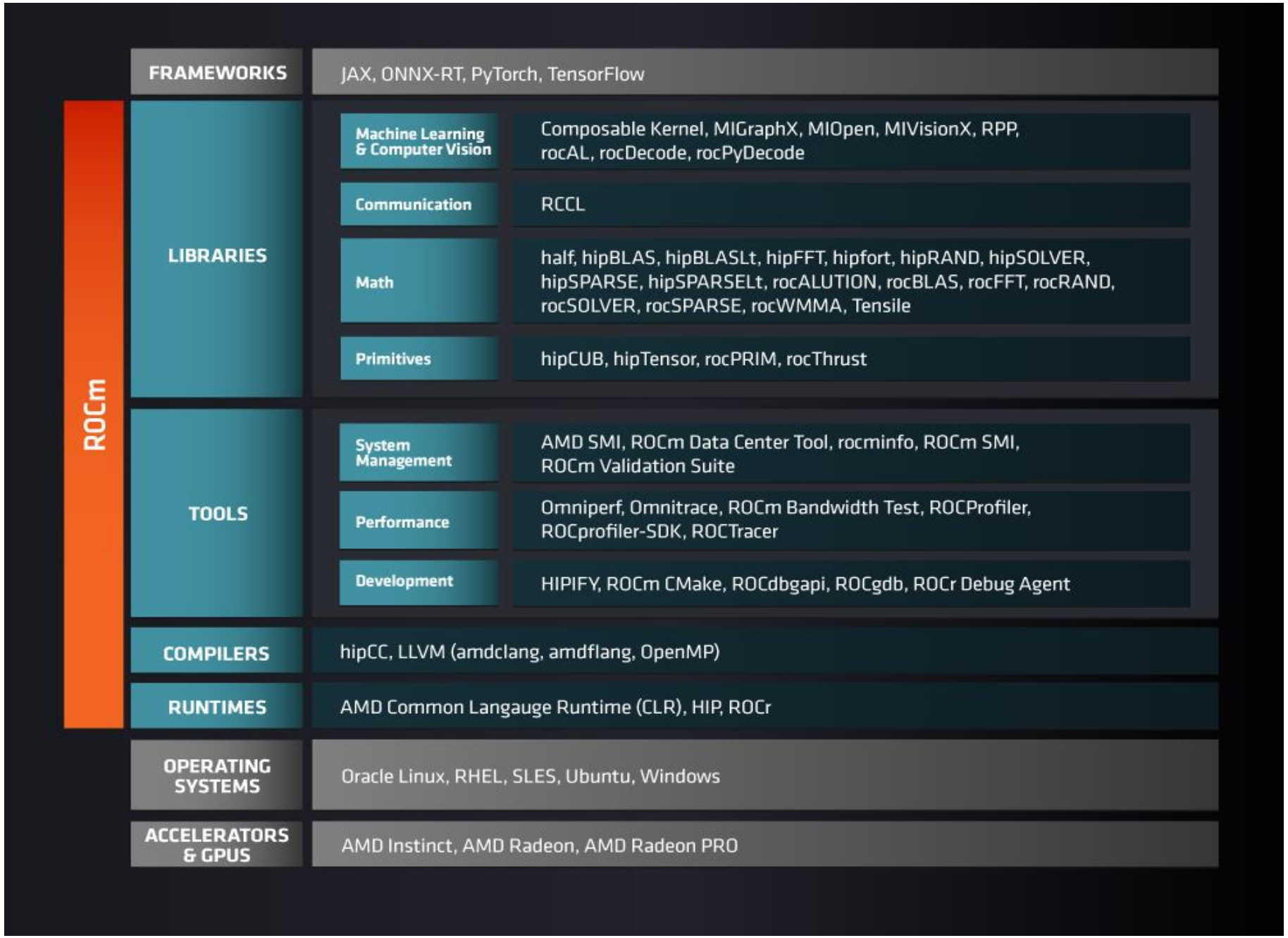
https://developer.nvidia.com/blog/whats-new-and-important-in-cuda-toolkit-13-0/

ROCM Software ecosystem



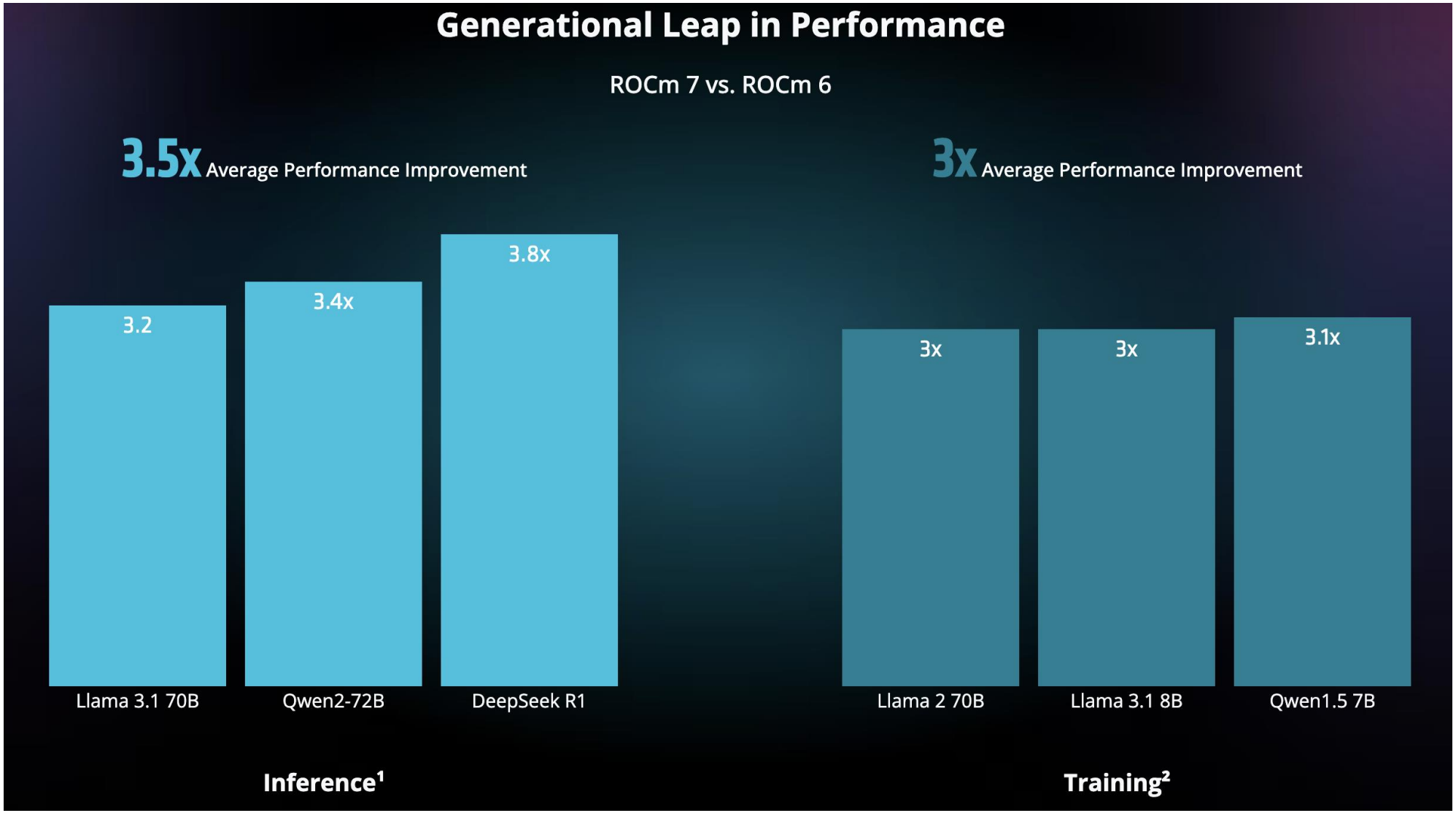
https://rocm.blogs.amd.com/software-tools-optimization/therock/README.html

ROCm CORE



https://rocm.docs.amd.com/en/docs-6.2.1/what-is-rocm.html

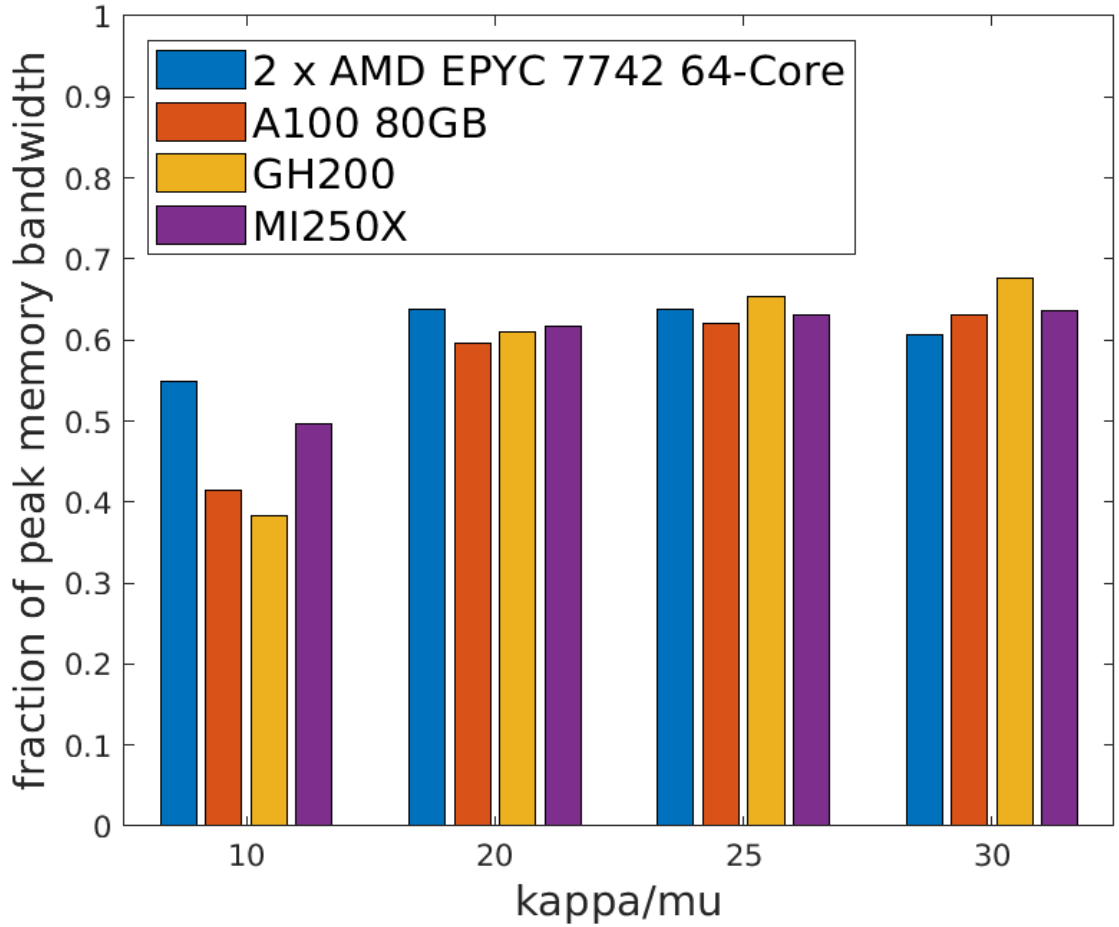
Latest ROCm Software: 7.



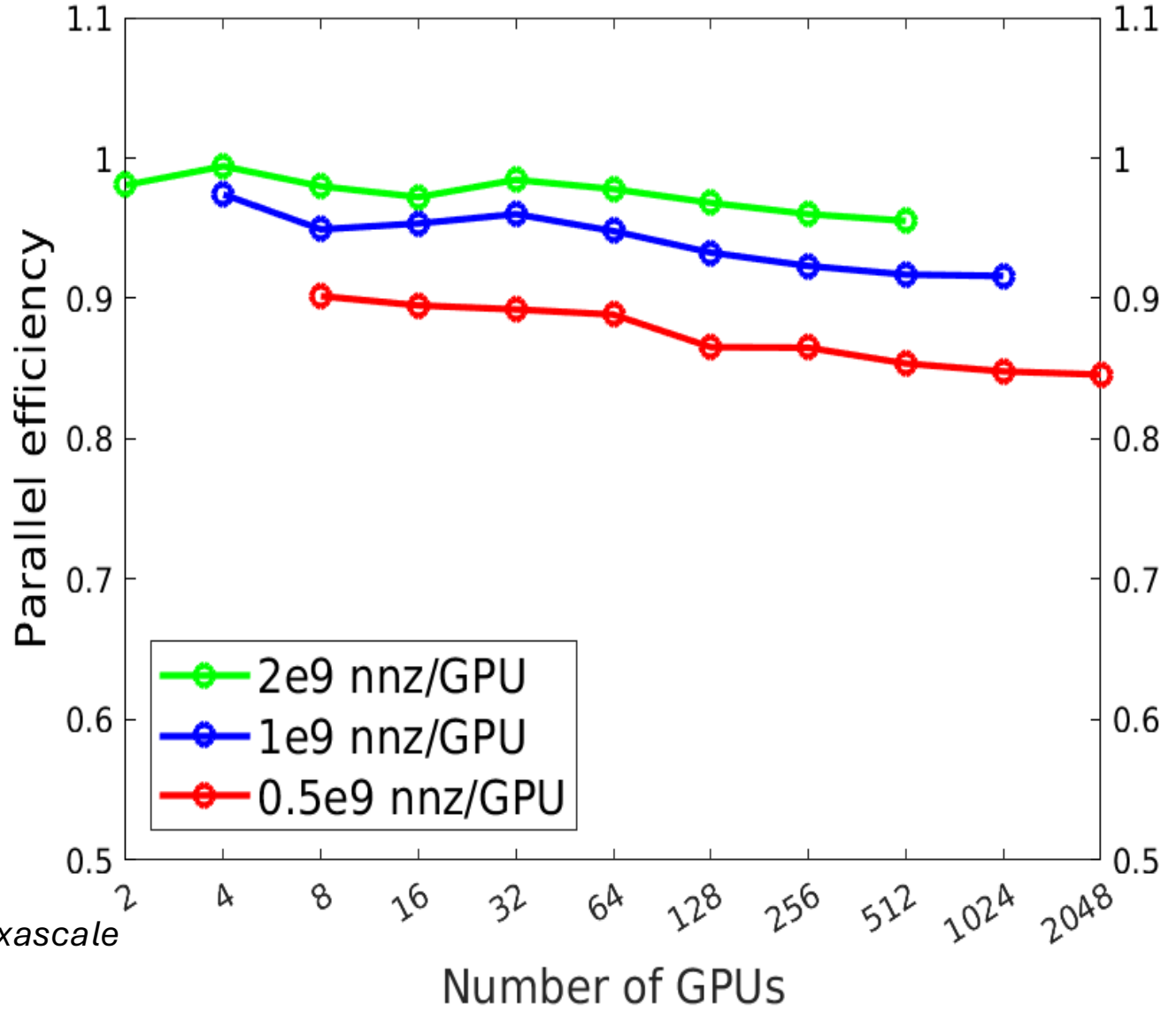
https://www.amd.com/en/products/software/rocm/whats-new.html

Benchmark: GPU-accelerated solver

Time-dependent Dirac Eq.



LUMI-G



GADE - GPU-Acceleration of Time-Dependent Dirac Equation for Exascale
Under review: Computer Physics Communications
https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5174937

Final Takeaway

Hardware & Architecture

- Chiplets, high-bandwidth interconnects, and AI engines power modern GPUs (GB300, MI350).
- Scaling driven by multi-die designs & massive memory bandwidth.

Software Ecosystems

- CUDA and ROCm deliver full-stack acceleration for AI & HPC.
- New tooling makes GPU programming faster and simpler.

Big Picture

- GPUs are the engines of modern AI and scientific computing.

In short: Modern GPUs deliver scalable, efficient performance, from hardware to software.

I stop here ☺

Get in touch:
My LinkedIn profile

https://www.linkedin.com/in/hicham-agueny-956a1368/

Access Slides

https://github.com/HichamAgueny/multigpu_mpi_course