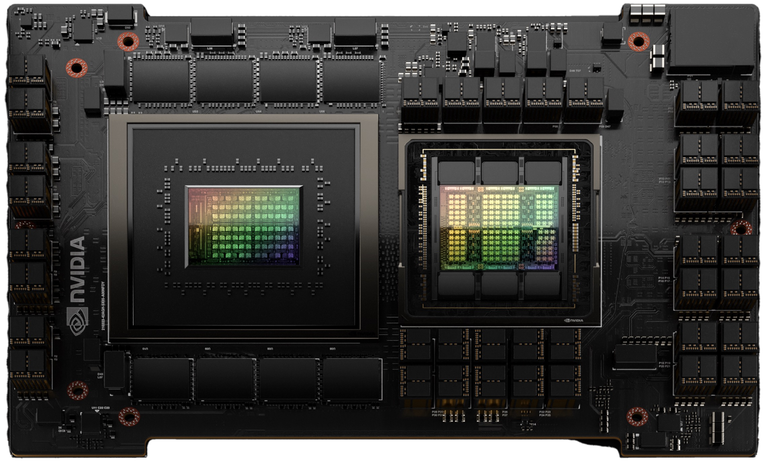

# NVIDIA Grace

Contact: Filippo Spiga <fspiga@nvidia.com>

# NVIDIA Grace for HPC & AI Infrastructure

## Grace Hopper Superchip

### Giant Scale AI & HPC



*Accelerated applications where CPU performance and system memory BW are critical; extreme and highly atomic collaboration between CPU & GPU contexts for flagship AI & HPC*

## Grace CPU Superchip

### CPU Computing

*Applications that run on CPU but where absolute performance, energy efficiency, and datacenter density matter, such as in scientific computing, data analytics, and hyperscale computing applications*

# NVIDIA Grace CPU Superchip

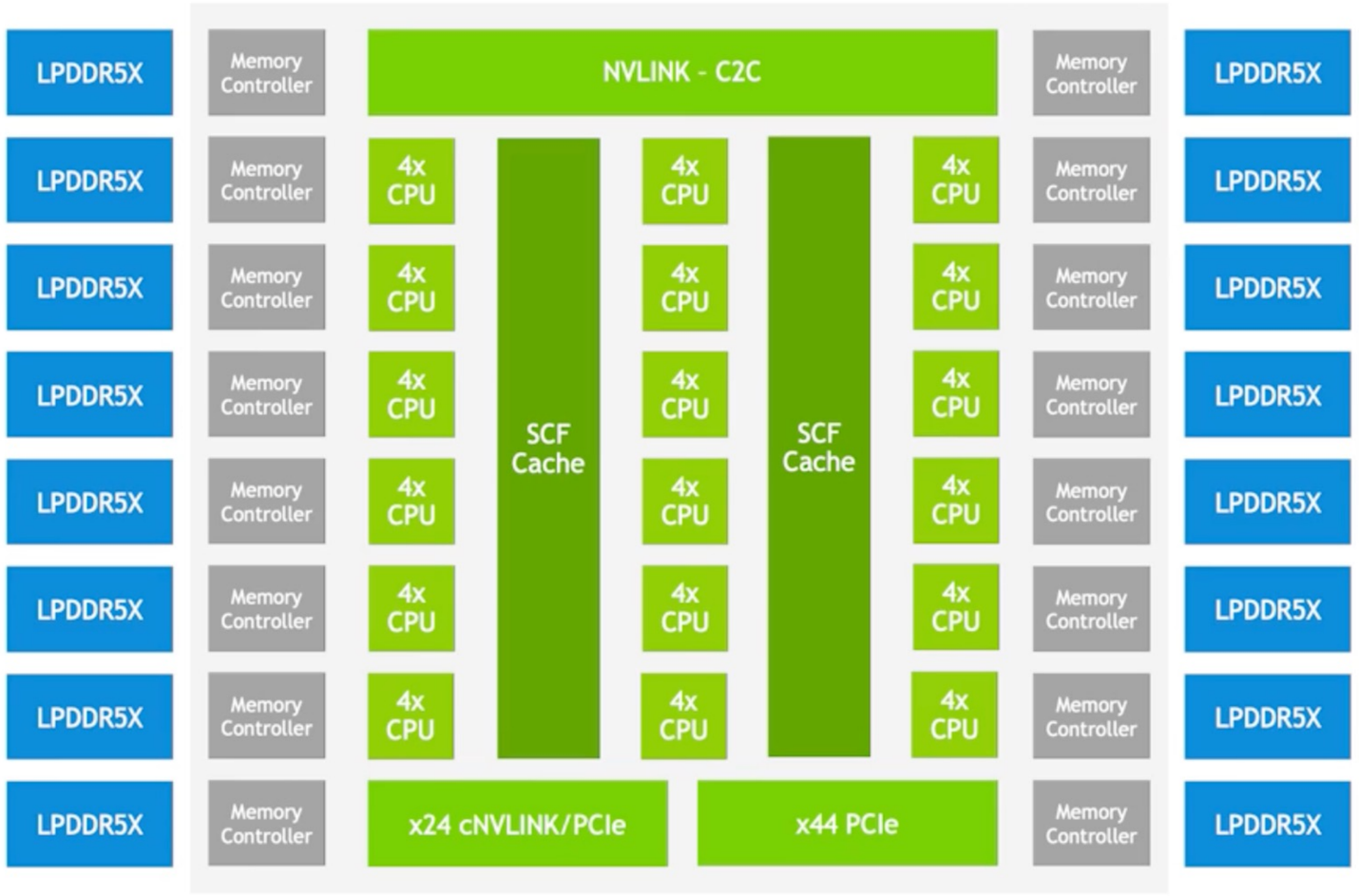
The Full Power of Grace

| Specifications | Grace SuperChip |
|---|---|
| Architecture | Armv9.0-A, SVE2 with 4x128b SIMD pipeline/core |
| Cores / Speed | 144 cores, 3.2GHz |
| Memory | LPDDR5x soldered down, 1TB/s BW<br>Up to 1TB per superchip |
| Cache | L1: 64KB i-cache + 64KB d-cache per core<br>L2: 1MB per core<br>L3: 234MB per superchip |
| Power | 500W including LPDDR5x memory |
| Interfaces | Up to 8x PCIe Gen5 x16 HS interface |
| Process Node | TSMC 4N |
| Availability | H1 2023 |

# NVIDIA GRACE

## Memory

- Up to 512GB of LPDDR5X memory
- 32 channels
- Up to 546 GB/s of memory BW
- But why LPPDR?



# MEMORY CHOICES

HBM, DDR, or LPDDR?

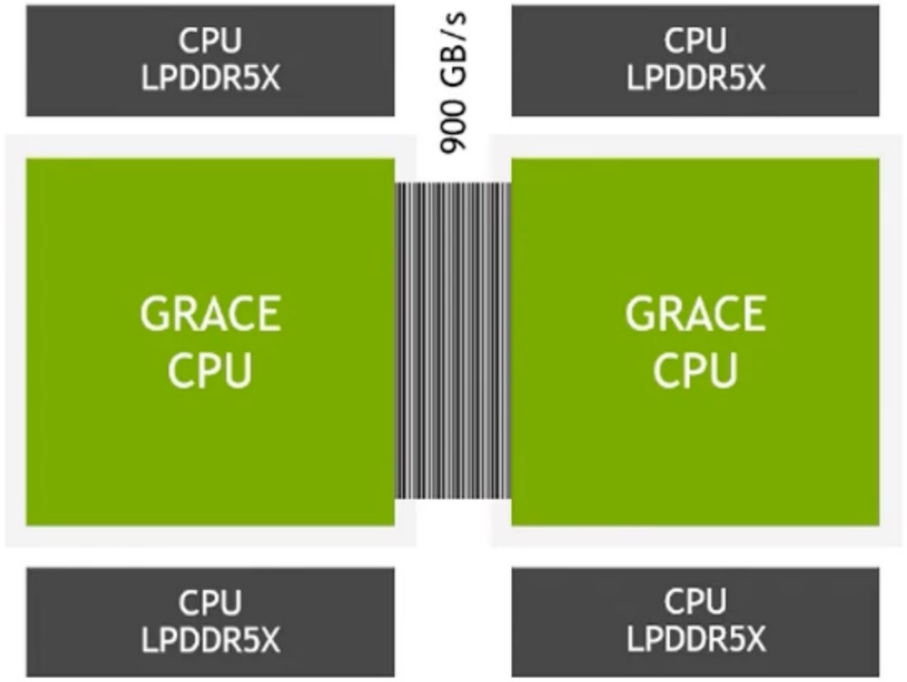
| | HBM2e (4-sites) | DDR5 (8-channel) | LPDDR5X (32-channel) |
|---|---|---|---|
| Capacity | 64GB | Up to 4TB | Up to 512GB |
| BW | Up to 1.8TB/s | Up to 358GB/s | Up to 546GB/s |
| Power/GBps | 1x | 8x | 1x |
| Cost/GB | >3x | 1x | 1x |

Remember? C2C BW - 900 GB/s

# NVLINK-C2C

High Speed Chip to Chip Interconnect

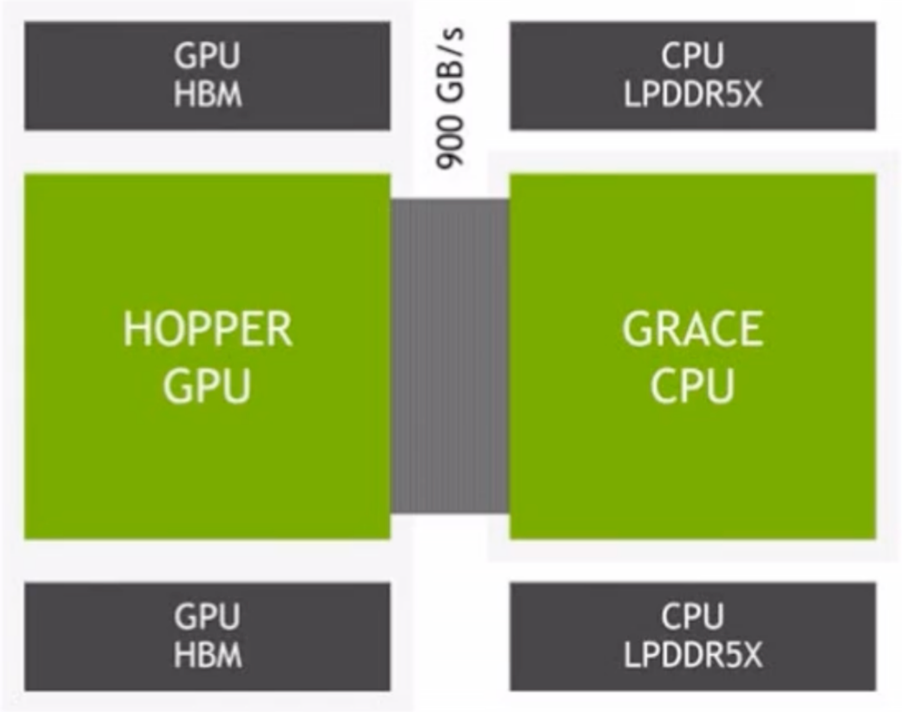
- Used to create the Grace Hopper, and Grace Superchips
- Removes the typical cross-socket bottlenecks
- Up to 900GB/s of raw bidirectional BW
  - Same BW as GPU to GPU NVLINK on Hopper
- Low power interface - 1.3 pJ/bit
  - More than 5x more power efficient than PCIe
- Enables coherency for both Grace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



# GRACE HOPPER

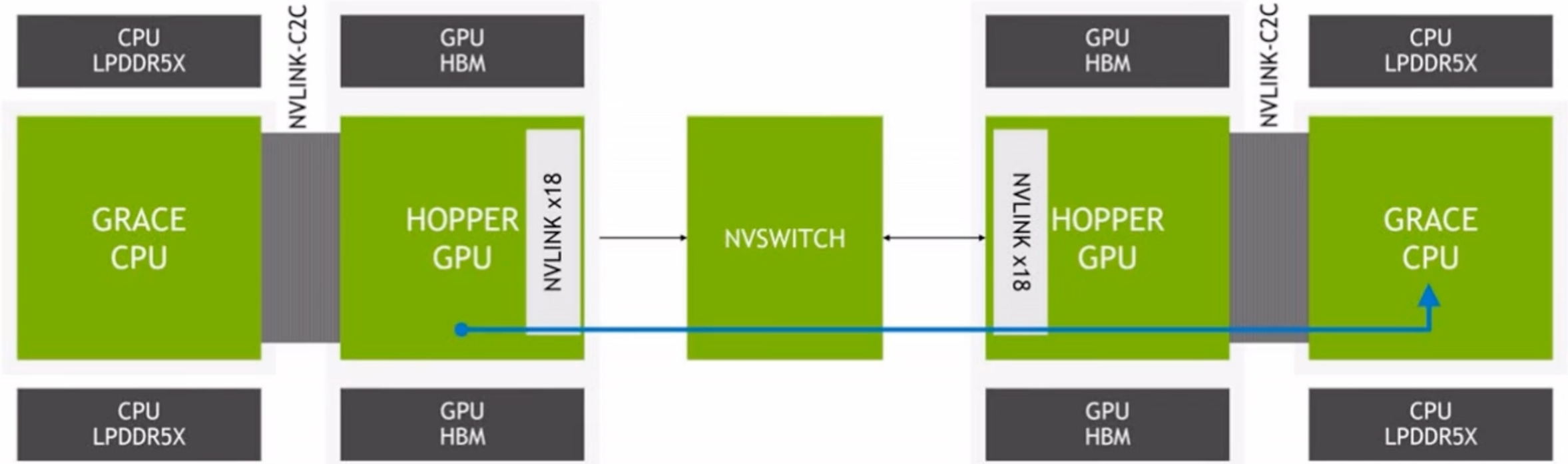
## Heterogenous Coherency

- Unified Memory with shared page tables
  - Shared CPU and GPU virtual address space
  - Transparent GPU access to pageable memory
  - System allocator support for GPU memory
    - Yes, malloced and mmaped pointers!
- Native atomics, including standard C++ atomic support



# NVLINK-SCALING

Superchip Scaling | CPU/GPU | Extended GPU Memory




Enables remote NVLINK connected GPUs, to access Grace's memory at native NVLINK speeds

# NVIDIA Grace vs. Fujitsu A64FX

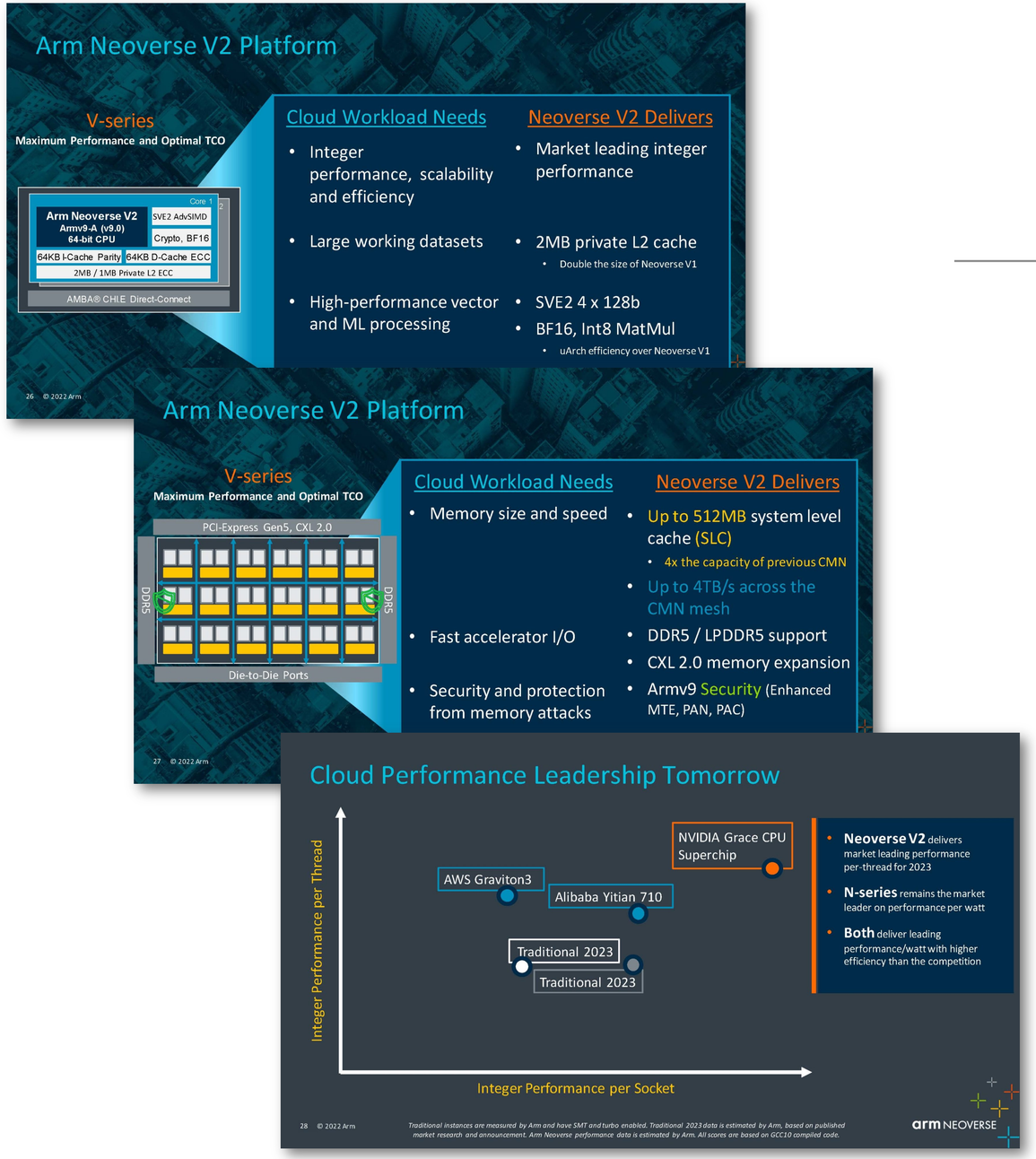
A64FX is an outlier in every way – Grace is mainstream

## MAINSTREAM LEADERSHIP HPC

| | |
|---|---|
| Familiar design | • High single-thread performance<br>• Simple memory hierarchy |
| Large user community | • Runs key HPC applications out-of-the-box<br>• Standard best practices hold true |
| Significant fraction of peak w/o tuning | • OSS toolchains (i.e. GNU) are tuned for u-arch<br>• Performance curves generally follow expectation |


## EXTREME HPC CODESIGN

| | |
|---|---|
| Codesigned for specific application | • Custom hardware or software<br>• Trades generality for performance |
| Small userbase of extreme experts | • Nonstandard software environments<br>• Common assumptions may hurt performance |
| Significant tuning effort required | • OSS toolchains unlikely to be performant<br>• Plan for man-months of optimization effort |




# Grace Uses Arm Neoverse Cores

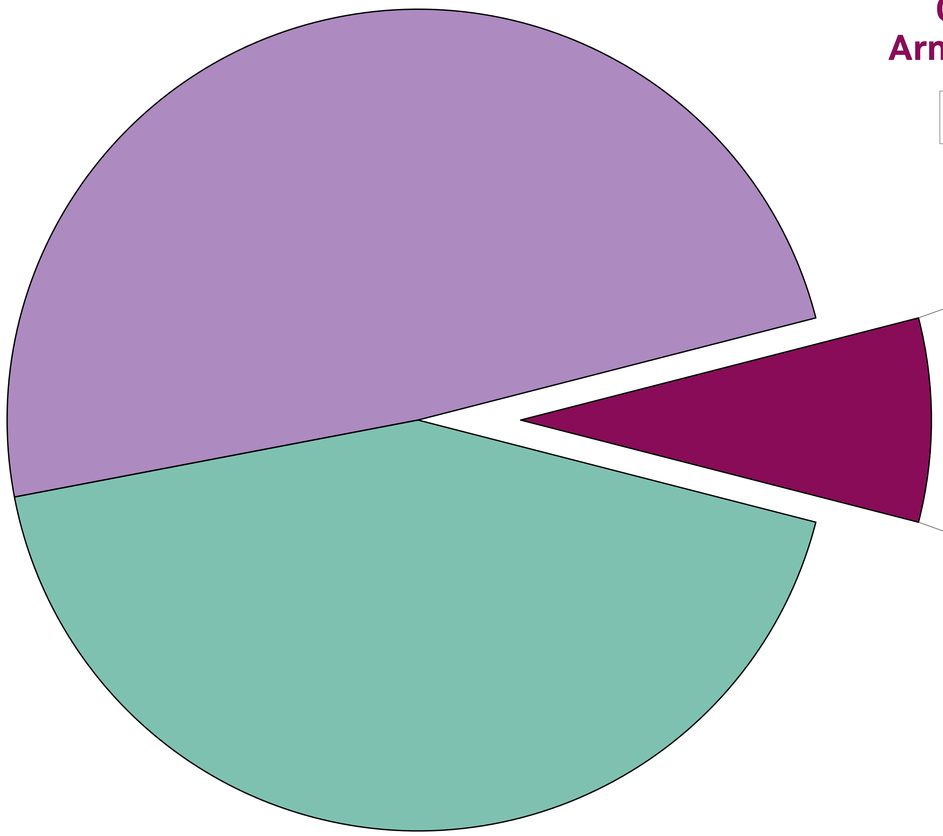
- Arm Neoverse V2 – like AWS Graviton 3 next-gen
- “NVIDIA Grace and CXL 2.0 PCIe Gen5 CPUs”
- https://www.servethehome.com/arm-neoverse-v2-cores-launched-for-nvidia-grace-and-cxl-2-0-pcie-gen5-cpus/
- https://www.theregister.com/2022/09/14/nvidias_grace_arm_neoverse_v2/
- https://www.hpcwire.com/off-the-wire/arm-announces-enhancements-for-its-neoverse-platform/

# Application Porting: Many Non-Trivial Cases Really Are Trivial

Vector intrinsics, dependencies, and nonstandard features are easily ported

**Straightforward, easy work < 1 day**

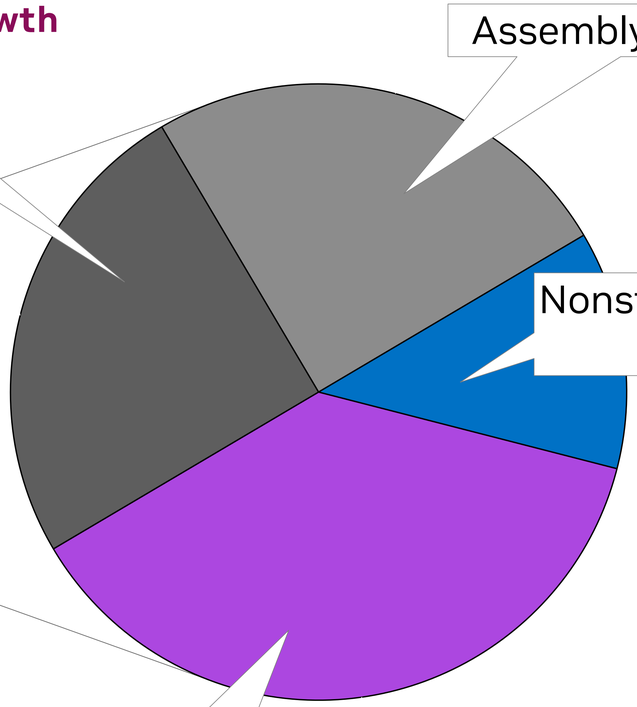
Recompile and reconfigure runtime parameters



**Job done!**

Found on Arm at another HPC center

**Cloud momentum**
**Arm ecosystem growth**




**Intrinsic conversion tools – SIMDe, SSE2NEON, etc.**

**Indicative workload mix inspired by an US DoE lab usage*

# Grace-Hopper Memory Model

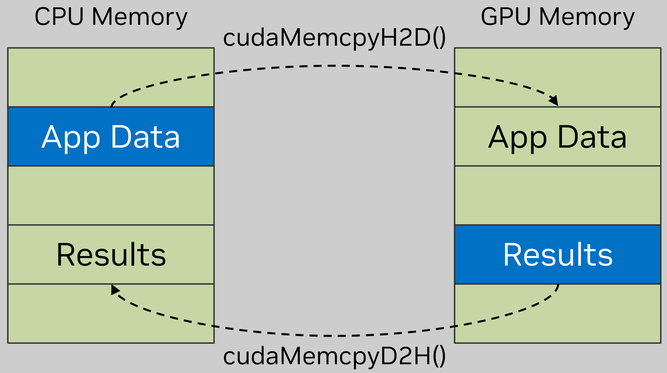
## Full CUDA support with additional Grace memory extensions

### 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 (per direction)

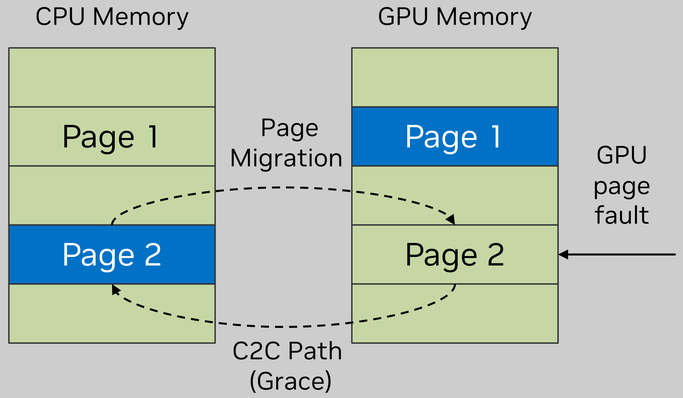



### Managed Memory

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

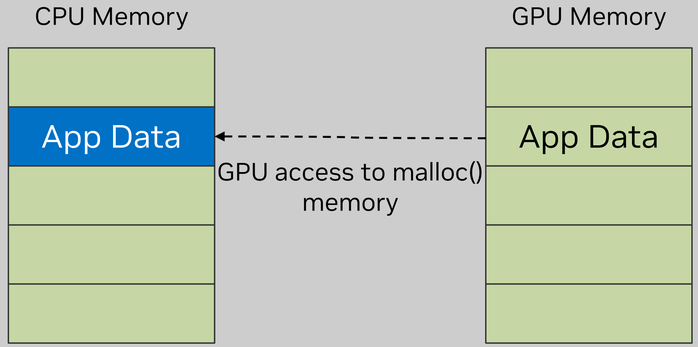



### System Allocated

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



# Simplifying the Developer Experience

## Universal accessibility of memory

**Allocation, Placement and Migration**

- `malloc()` will “just work” across CPU and GPU including all features, e.g. the atomics, the standard memory consistency model, etc.
- All memory, including `mmap()`, stack variables, global variables, Linux kernel syscall returning a pointer, etc.
- Placement/migration gives application flexibility. E.g, with `mmap()` use GPU memory instead of CPU memory for file cache

| Call | Action |
|---|---|
| **`malloc`**, **`mmap`**, `cudaMallocManaged` | • Allocation of either GPU or CPU memory, or combined GPU memory and CPU memory, across sockets<br>• Application control placement at page granularity via first touch<br>• Accessible to all computing devices<br>• Application can migrate existing pages |
| `cudaMalloc` | • allocate one socket’s GPU memory, accessible to all GPUs (not accessible to CPU), does not migrate |
| `cudaMallocHost` | • allocate CPU Memory, universally accessible, does not migrate |


NVIDIA