OAK RIDGE National Laboratory

Summit and **Frontier** at the **Oak Ridge Leadership Computing Facility**

Swaroop Pophale
Computer Science Research
Oak Ridge National Laboratory

July 27, 2020
Argonne Training Program on Extreme-Scale Computing

ORNL is managed by UT-Battelle LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Outline

- OLCF Roadmap to Exascale
- Summit Architecture Details
- Summit Programming Environment
- Summit and Scientific Discovery
- Frontier Overview

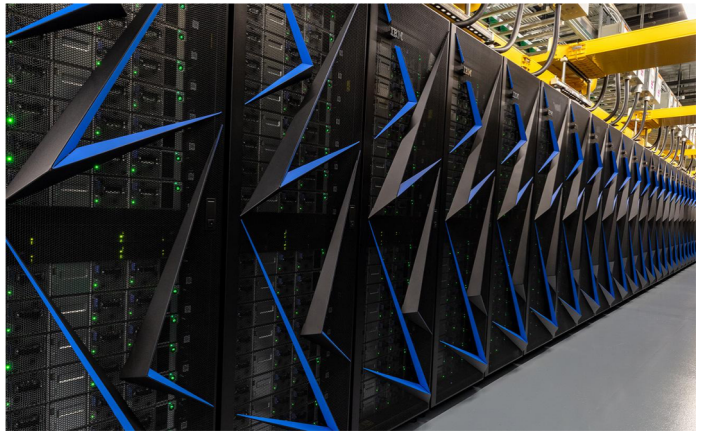
OAK RIDGE National Laboratory

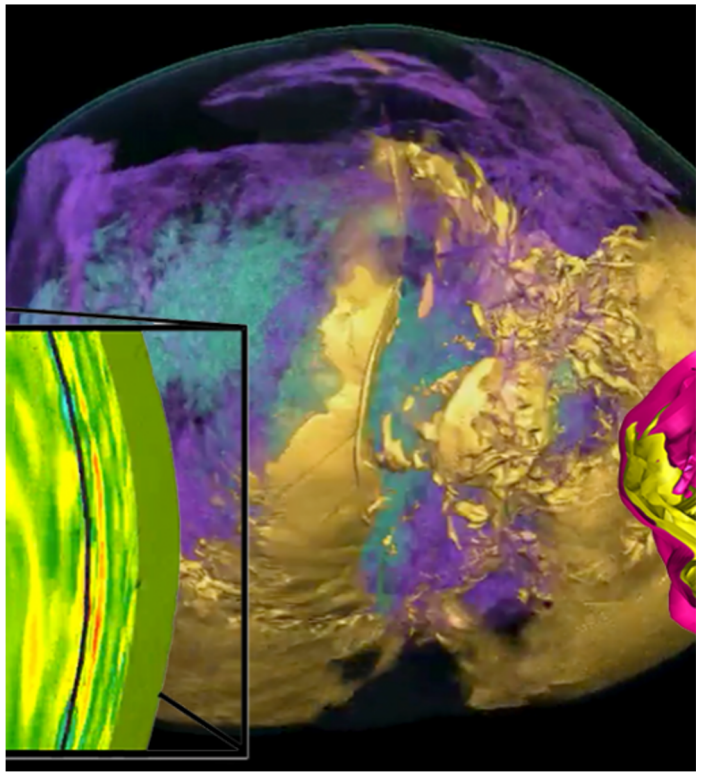
Oak Ridge Leadership Computing Facility (OLCF)

Oak Ridge Leadership Computing Facility (OLCF) Mission

The OLCF is a DOE Office of Science National User Facility whose mission is to enable breakthrough science by:

- Fielding the most powerful capability computers for scientific research,
- Building the required infrastructure to facilitate user access to these computers,
- Selecting a few time-sensitive problems of national importance that can take advantage of these systems,
- Partnering with these teams to deliver breakthrough science (Liaisons)





ORNL Leadership-class Supercomputers

OLCF-1

OLCF-2

OLCF-3

18.5 TF
2004
Cray X1E
Phoenix

25 TF
2005
Cray XT3
Jaguar

54 TF
2006
Cray XT3
Jaguar

62 TF
2007
Cray XT4
Jaguar

263 TF
2008
Cray XT4
Jaguar

1 PF
2008
Cray XT5
Jaguar

2.5 PF
2009
Cray XT5
Jaguar

World's fastest

1000-fold improvement in 8 years

27 PF
2012
Cray XK7
Titan

World's fastest

Titan, after **7 years** of valuable service, was decommissioned on August 2nd, 2019.

OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

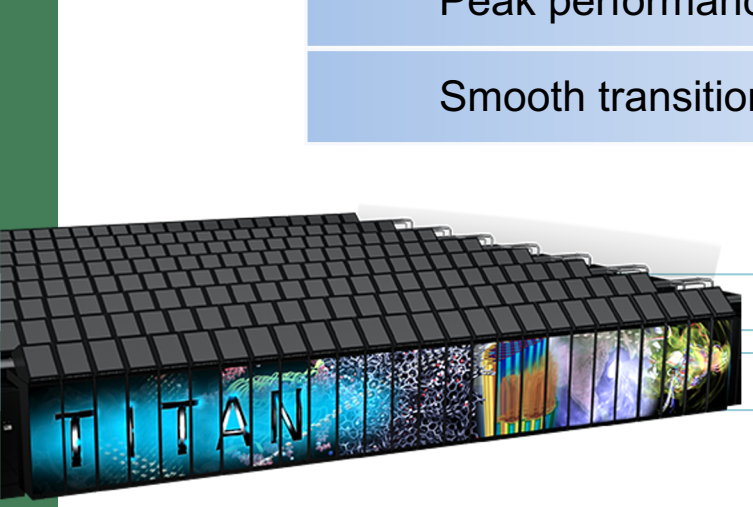
OLCF Path to Exascale

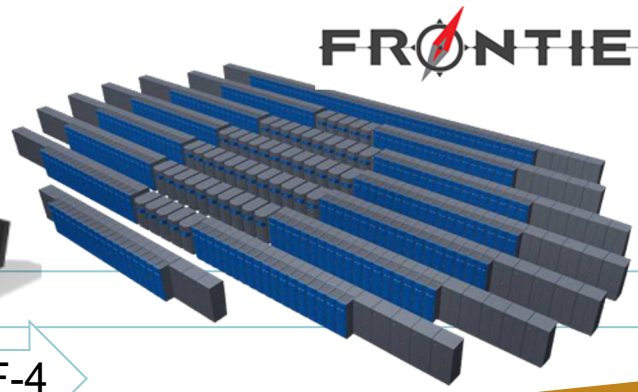
Competitive procurement asking for:

- 50–100× application performance of Titan
- Support for traditional modeling and simulation, high-performance data analysis, and artificial intelligence applications
- Peak performance of at least 1300 PF
- Smooth transition for existing and future applications

The Exascale Computing Project has emphasized that Exascale is a measure of application performance, and this RFP reflects that, asking for nominally 50× improvement over Sequoia and Titan.

-- Design Reviewer





OLCF-4

OLCF-5

27 PF

2012
Cray XK7
Titan

200 PF

2018
IBM
Summit

World's fastest

~1 EF

2021
Frontier

From Jaguar to Frontier, 500-fold improvement

World's fastest

OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

ORNL Summit System Overview

System Performance

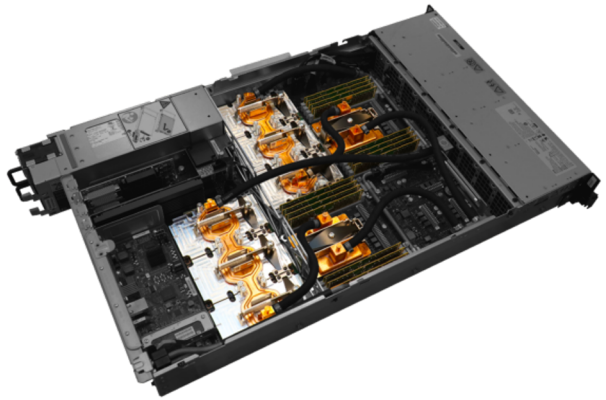
- Peak of 200 Petaflops (FP_{64}) for modeling & simulation
- Peak of 3.3 ExaOps (FP_{16}) for data analytics and artificial intelligence

The system includes

- 4,608 nodes
- Dual-rail Mellanox EDR InfiniBand network
- 250 PB IBM file system transferring data at 2.5 TB/s

Each node has

- 2 IBM POWER9 processors
- 6 NVIDIA Tesla V100 GPUs
- 608 GB of fast memory
  (96 GB HBM2 + 512 GB DDR4)
- 1.6 TB of non-volatile memory



Summit Demonstrated Its Balanced Design (2018)

#1 on Top 500, #1 HPCG, #1 Green500, and #1 I/O 500

144 PF HPL
#1 raw performance

2.9 PF HPCG
#1 fast data movement

14.668 GF/W
#1 energy efficiency

#1 HPC storage performance

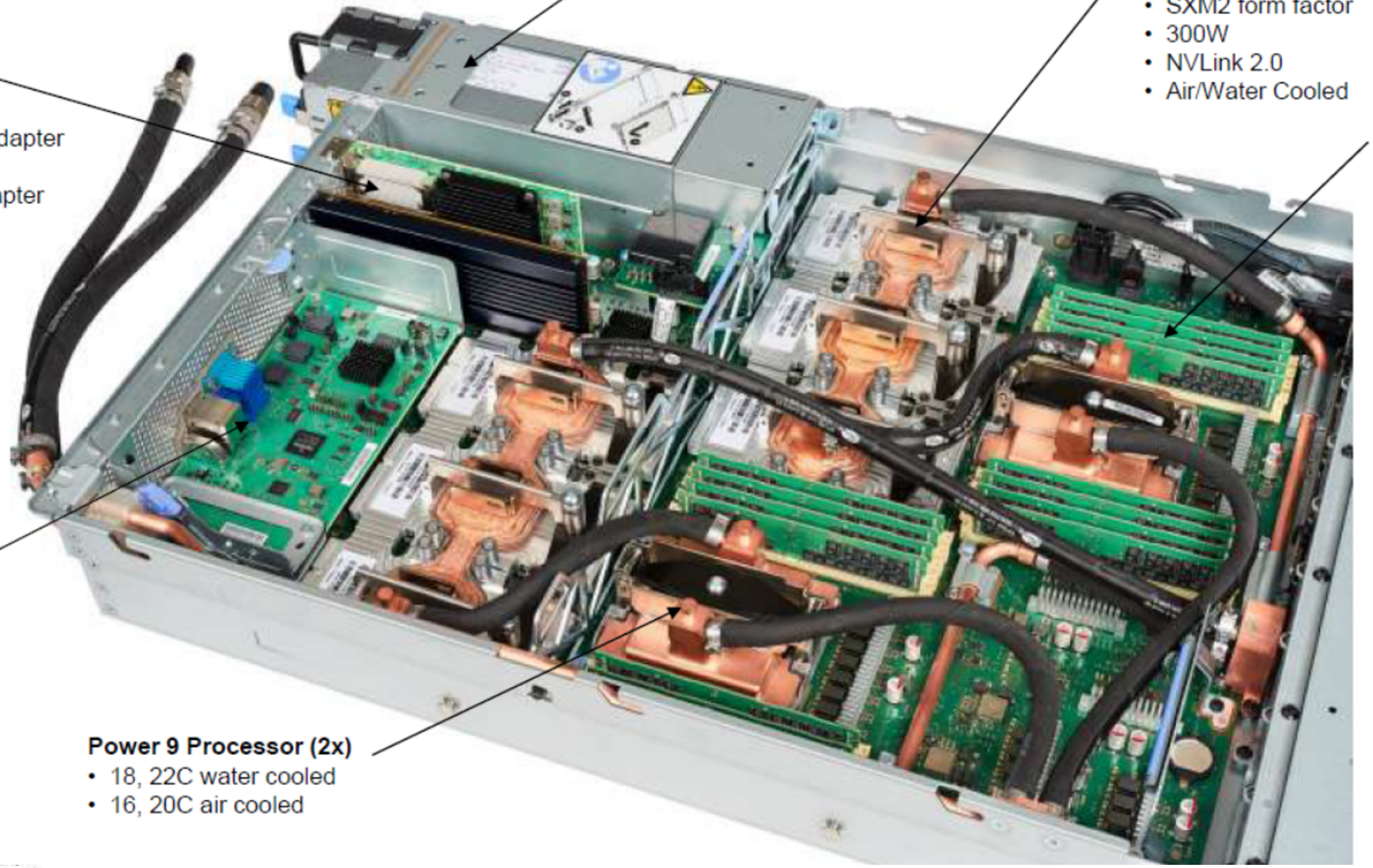
OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

How is Summit different from Titan?

- Many fewer nodes
- Much more powerful nodes
- Much more memory per node and total system memory
- Faster interconnect
- Much higher bandwidth between CPUs and GPUs
- Much larger and faster file system
- ~7X more performance for slightly more power (Summit's 8.8 MW vs Titan's 8.2)

Feature	Titan	Summit
Application Performance	Baseline	5-10x Titan
Number of Nodes	18,688	4,608
Node performance	1.4 TF	42 TF
Memory per Node	32 GB DDR3 + 6 GB GDDR5	512 GB DDR4 + 96 GB HBM2
NV memory per Node	0	1600 GB
Total System Memory	710 TB	>10 PB DDR4 + HBM2 + Non-volatile
System Interconnect	Gemini (6.4 GB/s)	Dual Rail EDR-IB (25 GB/s)
Interconnect Topology	3D Torus	Non-blocking Fat Tree
Bi-Section Bandwidth	112 TB/s	115.2 TB/s
Processors	1 AMD Opteron™ 1 NVIDIA Kepler™	2 IBM POWER9™ 6 NVIDIA Volta™
File System	32 PB, 1 TB/s, Lustre®	250 PB, 2.5 TB/s, GPFS™
Power Consumption	9 MW	13 MW

Summit Board (1 node)



Power Supplies (2x)
- 2200W
- 200VAC, 277VAC, 400VDC input

NVidia Volta GPU
- 3 per socket
- SXM2 form factor
- 300W
- NVLink 2.0
- Air/Water Cooled

Memory DIMM's (16x)
- 8 DDR4 IS DIMMs per socket
- 8, 16, 32,64, 128GB DIMMs

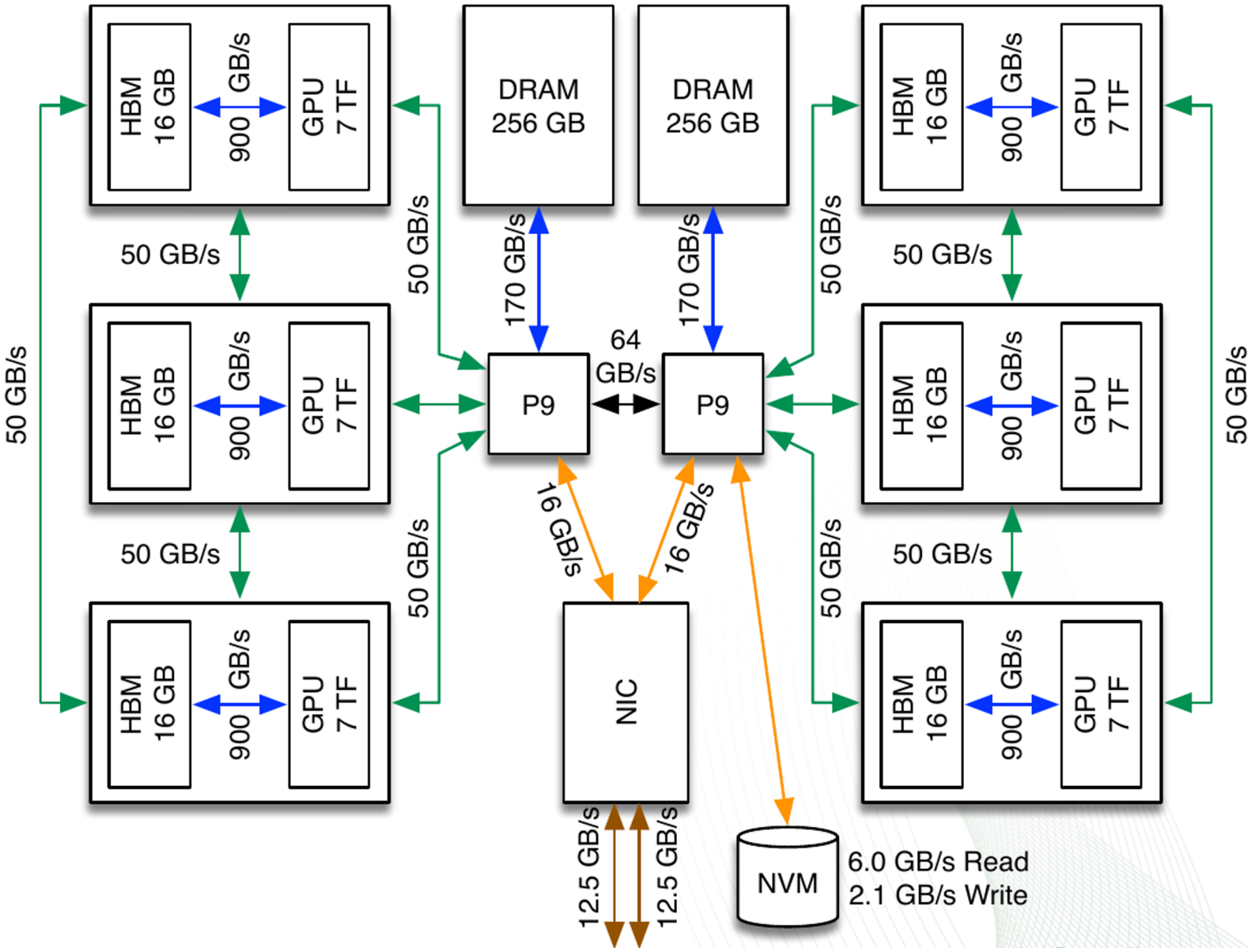
PCIe slot (4x)
- Gen4 PCIe
- 2, x16 HHHL Adapter
- 1, Shared slot
- 1 x8 HHHL Adapter

BMC Card
- IPMI
- 1 Gb Ethernet
- VGA
- 1 USB 3.0

Power 9 Processor (2x)
- 18, 22C water cooled
- 16, 20C air cooled

Summit Node Schematic

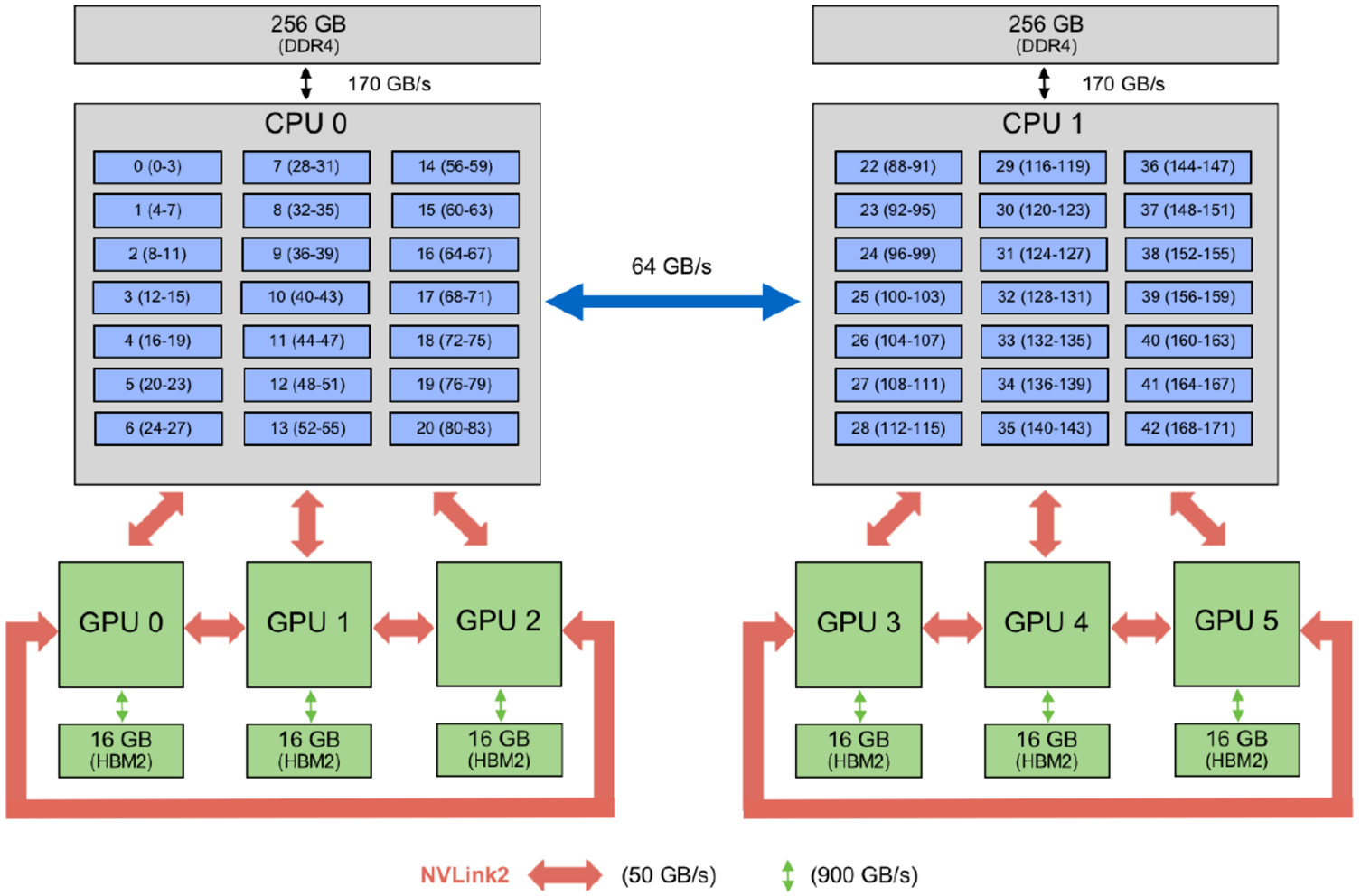
- Coherent memory across entire node
- NVLink v2 fully interconnects three GPUs and one CPU on each side node
- PCIe Gen4 connects NVMe and NIC
- Single shared NIC with dual EDR ports



Summit POWER9 Processors

IBM POWER9 Processor

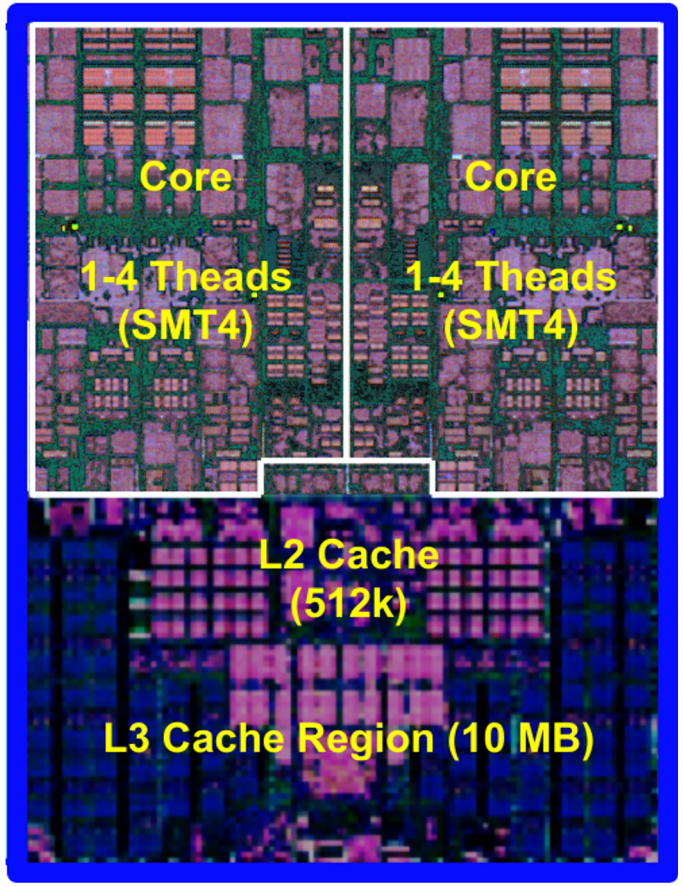
- 22 cores active, 1 core reserved for OS → reduce jitter
- 4 hardware threads (HT) per core
- Three SMT modes: SMT1, SMT2, SMT4. Each thread operates independently.
- 4 HT shares L1 cache, 8 HT (2 cores) shares L2 and L3 cache



Summit POWER9 Processors (2)

IBM POWER9 Processor

- 22 cores active, 1 core reserved for OS → reduce jitter
- 4 hardware threads (HT) per core
- Three SMT modes: SMT1, SMT2, SMT4. Each thread operates independently.
- 4 HT shares L1 cache, 8 HT (2 cores) shares L2 and L3 cache

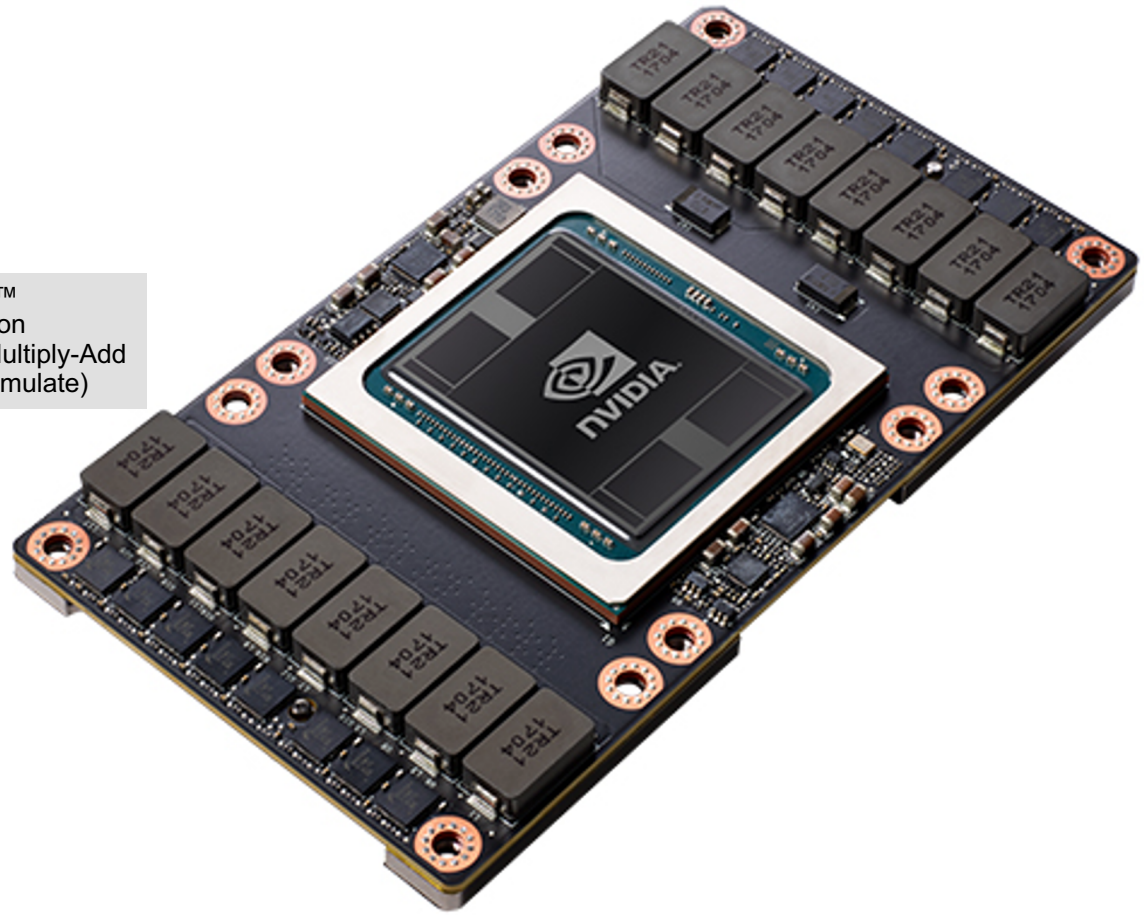


2 POWER9 cores

Summit GPUs: 27,648 NVIDIA Volta V100s

	Tesla V100 for NVLink	Tesla V100 for PCIe
PERFORMANCE with NVIDIA GPU Boost™	DOUBLE-PRECISION 7.8 TeraFLOPS SINGLE-PRECISION 15.7 TeraFLOPS DEEP LEARNING 125 TeraFLOPS	DOUBLE-PRECISION 7 TeraFLOPS SINGLE-PRECISION 14 TeraFLOPS DEEP LEARNING 112 TeraFLOPS
INTERCONNECT BANDWIDTH Bi-Directional	NVLINK 300 GB/s	PCIE 32 GB/s
MEMORY CoWoS Stacked HBM2	CAPACITY 16 GB HBM2 BANDWIDTH 900 GB/s	

TensorCores™
Mixed Precision
(16b Matrix-Multiply-Add and 32b Accumulate)

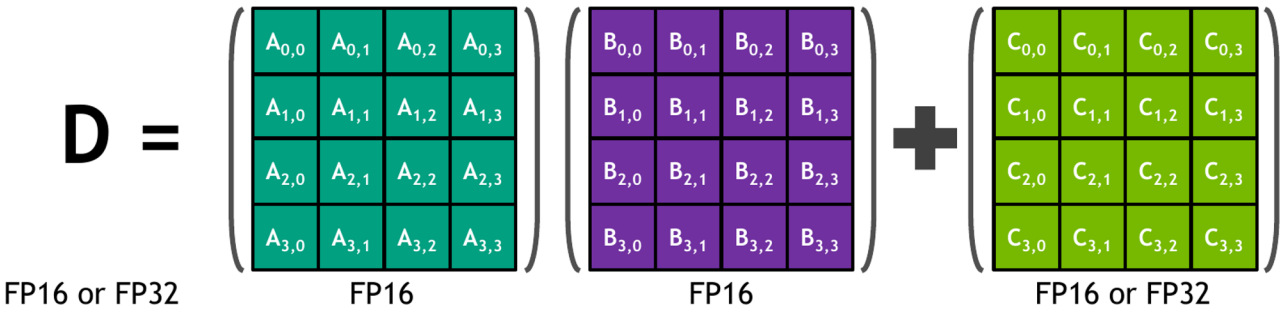


Note: The performance numbers are peak and not representative of Summit's Volta

Summit GPUs: 27,648 NVIDIA Volta V100s (2)

Tensor cores on V100:

- Tensor cores do mixed precision multiply add of 4x4 matrices



- 640 Tensor cores (8 on each 80 SMs)
- Up to 125 Half Precision (FP_{16}) TFlops
- Requires application to figure out if/when utilizing mixed/reduce precision is possible
  - e.g. see Haidar et al (ICL at UTK), SC18 paper
  - CoMet Comparative Genomics application (2018 ACM Gordon Bell Prize winner), achieving 2.36 ExaOps (mixed-precision) on Summit

Stream Benchmark: Summit (vs. Titan)

- A simple synthetic benchmark program that measures achievable memory bandwidth (in GB/s) under OpenMP threading.

System Cores	Peak (Summit) 44	Titan 16
Copy	274.6	34.9
Scale	271.4	35.3
Add	270.6	33.6
Triad	275.3	33.7
Peak (theoretical)	340	51.2
Fraction of Peak	82%	67%

DRAM Bandwidth

System	Peak (Summit)	Titan
Copy	789	181
Scale	788	181
Add	831	180
Triad	831	180
Peak (theoretical)	900	250
Fraction of Peak	92%	72%

GDDR Bandwidth

Slide courtesy of Wayne Joubert, ORNL

NVLink Bandwidth

- Measured from core 0 the achieved CPU-GPU NVLink rates with a modified BandwidthTest from NVIDIA CUDA Samples

GPU	0	1	2	3	4	5	peak
Host to Device	45.93	45.92	45.92	40.63	40.59	40.64	50
Device to Host	45.95	45.95	45.95	36.60	36.52	35.00	50
Bi-Directional	86.27	85.83	77.36	66.14	65.84	64.76	100

Single Node Single GPU NVLink Rates (GB/s)

- Not necessarily a use case that most applications will employ

Slide courtesy of Wayne Joubert, ORNL

NVLink Bandwidth (2)

- Measured the achieved CPU-GPU NVLink rates with a modified BandwidthTest from NVIDIA CUDA Samples using multiple MPI process evenly spread between the sockets.

MPI Process Count	1	2	3	4	5	6	Peak (6)
Host to Device	45.93	91.85	137.69	183.54	229.18	274.82	300
Device to Host	45.95	91.90	137.85	183.80	225.64	268.05	300
Bi-Directional	85.60	172.59	223.54	276.34	277.39	278.07	600

NVLink Rates with MPI Processes (GB/s)

- Ultimately limited by the CPU memory bandwidth
- 6 ranks driving 6 GPUs is an expected use case for many applications

NVLink Bandwidth (2)

- Measured the achieved NVLink transfer rates between GPUs, both within a socket and across them, using p2pBandwidthLatencyTest from NVIDIA CUDA Samples. (Peer-to-Peer communication turned on).

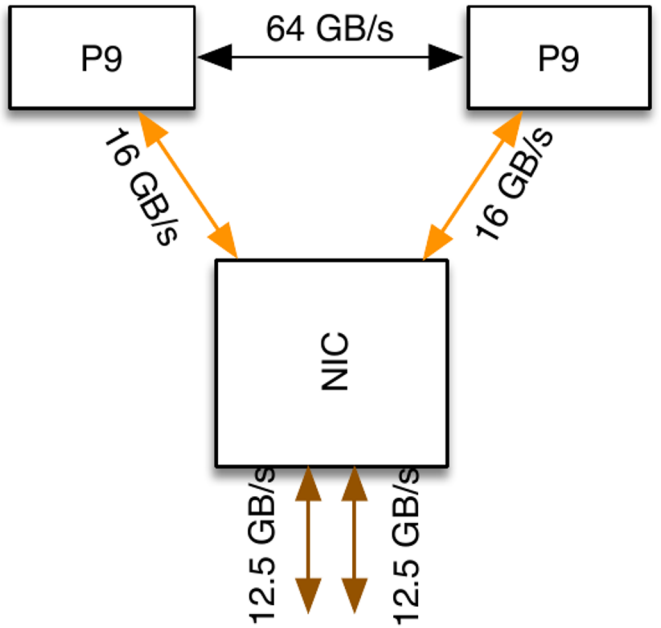
Socket	0	1	Cross	Peak
Uni-Directional	46.33	46.55	25.89	50
Bi-Directional	93.02	93.11	21.63	100

NVLink Rates for GPU-GPU Transfers (GB/s)

- Cross-socket bandwidth is much lower than that between GPUs attached to the same CPU socket

Summit Network

- Mellanox EDR Network with non-blocking fat-tree topology
  - Bisection bandwidth 115 TB/s
  - 2 physical ports per node (4 virtual) at 25 GB/s
    - must use both sockets to get full bandwidth
  - Set to minimize latency by default (tune-able)



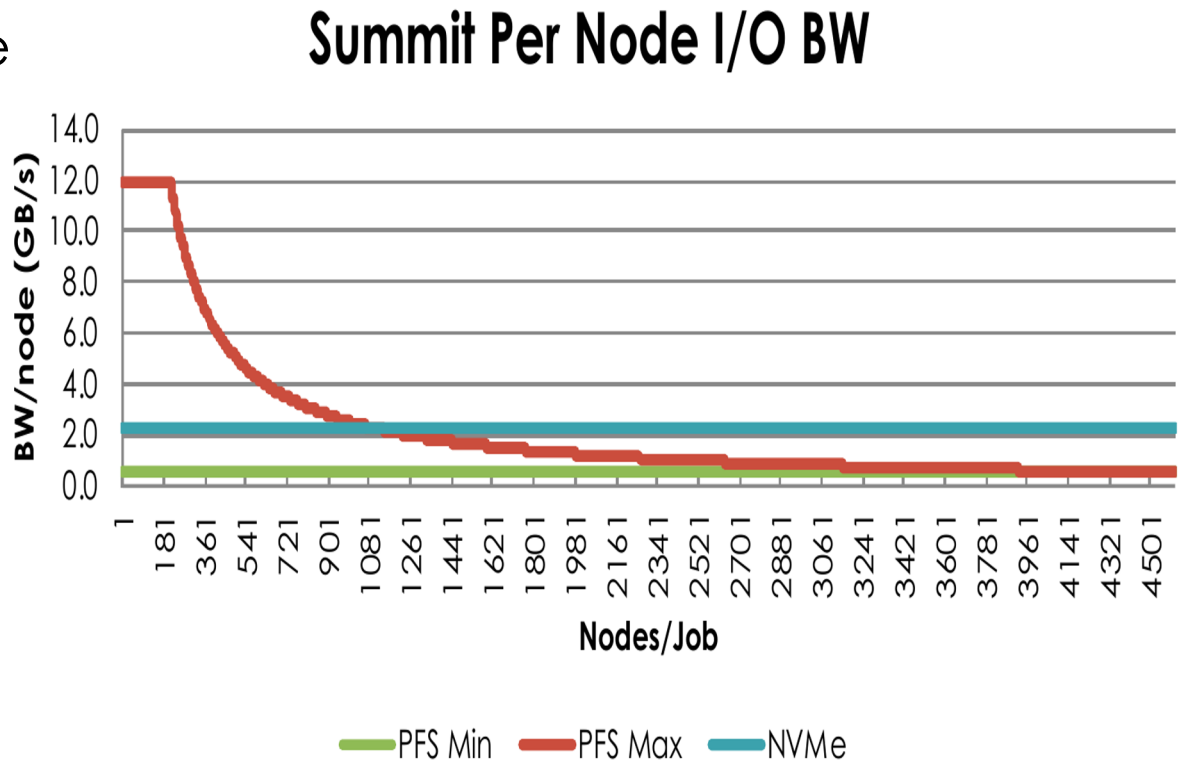
- Adaptive routing
  - Enable bypassing congestions
  - Out of order packets on the network
  - Packets are load balanced at each switch

- Scalable Hierarchical Aggregation (and) Reduction Protocol
  - SHARP: network builds trees in switches to accelerate some collective operations
  - Supported collectives (small <=2048): barrier, broadcast, reduce, allreduce

Summit Parallel File System and Burst Buffers (NVME)

- Alpine "SpectrumScale" File system:
  - 12-14 GB/s per node, 2.5 TB/s aggregate
  - Full system job: ~550 MB/s per node
  - Every node has access to the same space → can support multiple modes: single-shared file, file per rank, etc.

- Node Local NVME:
  - Samsung PMI1725A: Write 2.1 GB/s, Read 5.5 GBs
  - Scales linearly with job size
  - Shared only by ranks on a node,
  - Must drain to PFS at the end of a job (using tools or 'manually')



OAK RIDGE National Laboratory

Summit Programming Environment

Summit Compilers and Programming Model

All compilers (except Clang) support C, C++ and Fortran

Compiler	CUDA (C)	CUDA Fortran	OpenMP 4.5 (offload)	OpenMP (CPU)	OpenACC
PGI					
GCC			(*)		
IBM XL					
LLVM (C & C++)					

*: functional only

Summit Debuggers and Performance Tools

Debugger
DDT
cuda-gdb, -memcheck
Valgrind, memcheck, helgrind
pdb

Performance Tools
Open|SpeedShop
TAU
HPCToolkit (IBM)
HPCToolkit (Rice)
VAMPIR
nvprof
gprof

Summit Numerical Library

Library	OSS or Proprietary	CPU Node	CPU Parallel	GPU
IBM ESSL	Proprietary			
FFTW	OSS			
ScaLAPACK	OSS			
PETSc	OSS			
Trilinos	OSS			*
BLAS-1, -2, -3	Proprietary (thru ESSL)			
NVBLAS	Proprietary			
cuBLAS	Proprietary			
cuFFT	Proprietary			
cuSPARSE	Proprietary			
cuRAND	Proprietary			
Thrust	Proprietary			

Summit Job Launcher: `jsrun`

- jsrun provides abstraction of a node with a concept of 'resource set'
  - motivated by the fact that Summit has powerful nodes
- Resource set:
  - sub group of resources (GPUs, CPUs) within a node
  - using cgroup under the hood
  - executes <N> MPI processes (with threads) and manages placement
- Node-sharing (e.g. multiple executables) is possible within a job (i.e. one user):
  - Multiple Programs Multiple Data (MPMD)
  - concurrently execute compute intensive GPU-only job with CPU-only data analysis / visualization

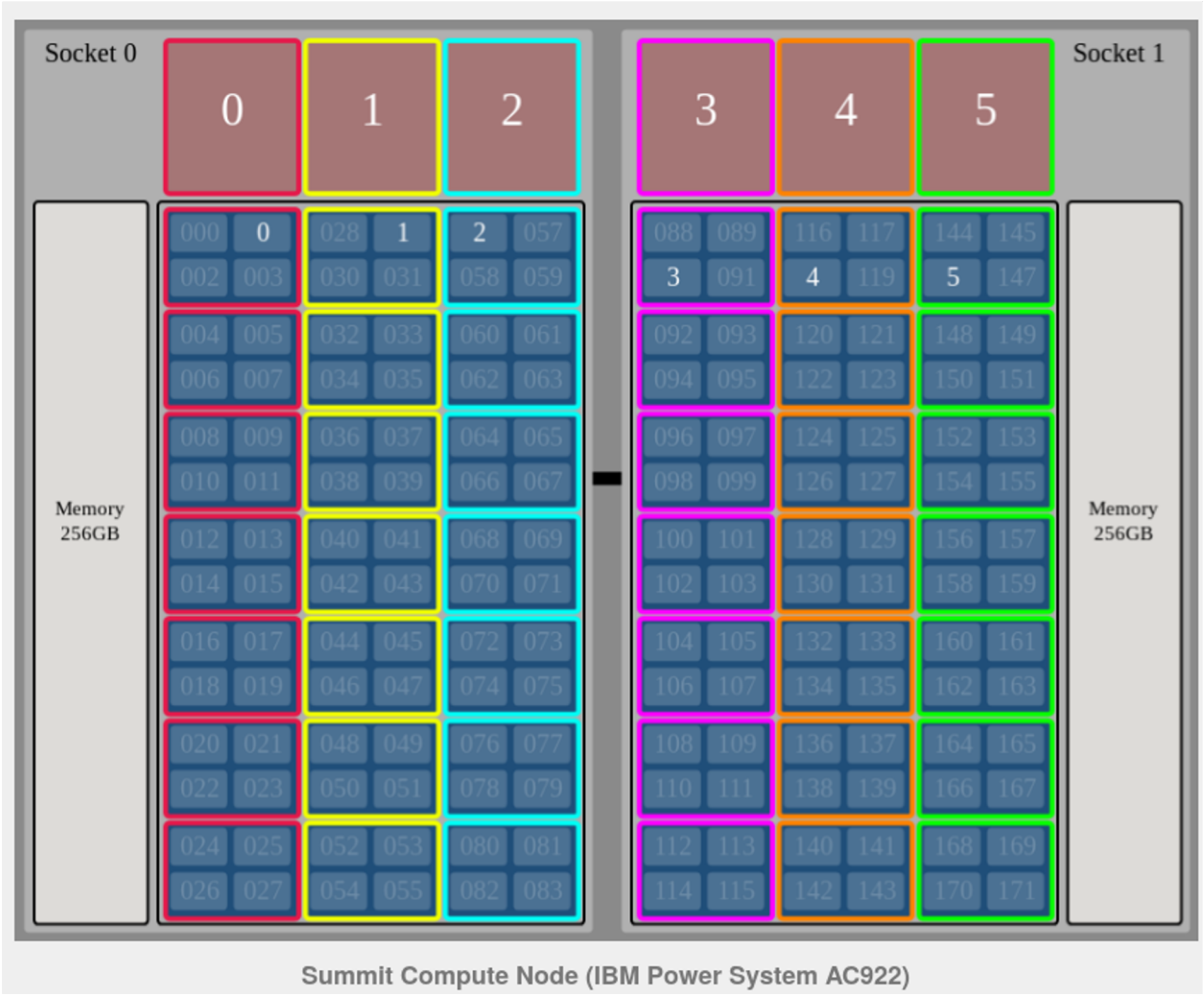
Programming Multiple GPUs

- Multiple paths, with different levels of flexibility and sophistication, e.g.:
  - Simple model: 1 MPI or 1 thread per GPU
  - Sharing GPU: multiple MPIs or threads share a GPU
  - Single MPI using multiple GPUs
  - Expose the node-level parallelism directly: multiple processes multiple GPUs
- Exposing more (node-level) parallelism is key to scalable applications from petascale-up

One GPU per MPI Rank

- 1 MPI rank per GPU → bind each rank to specific GPU
- "Titan-like" model
- MPI rank can use threads (e.g. OpenMP or Pthreads) to utilize more of the CPU cores
  - CPU is only small percentage (~3 %) of the total Flops

1 GPU per MPI rank, 6 MPI ranks per node, 1 thread per MPI rank

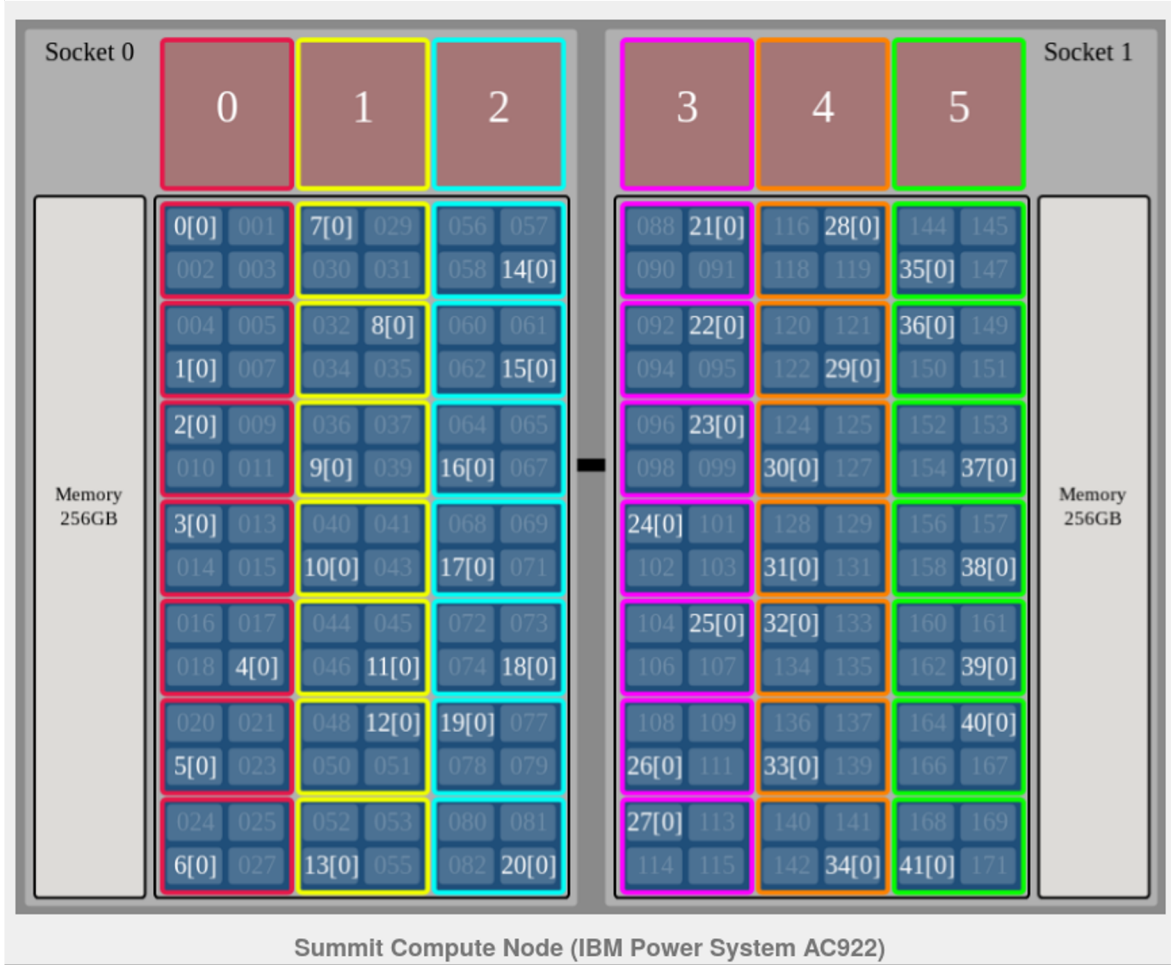


export OMP_NUM_THREADS=1
jsrun --nrs X --tasks_per_rs 1 --cpu_per_rs 7 --gpu_per_rs 1 --rs_per_host 6 --bind packed:7 <exec>

One GPU Shared by Multiple MPI ranks

- Multiple MPI ranks shared a single GPU
  - Using CUDA MPS (Multi-Process Service)
- Useful to increase GPU utilization, i.e. if a single MPI rank cannot fully occupy a GPU
- Can be more prone to comm. congestion
  - using threads is an alternative

7 MPI ranks share a GPU, 21 MPI ranks per node

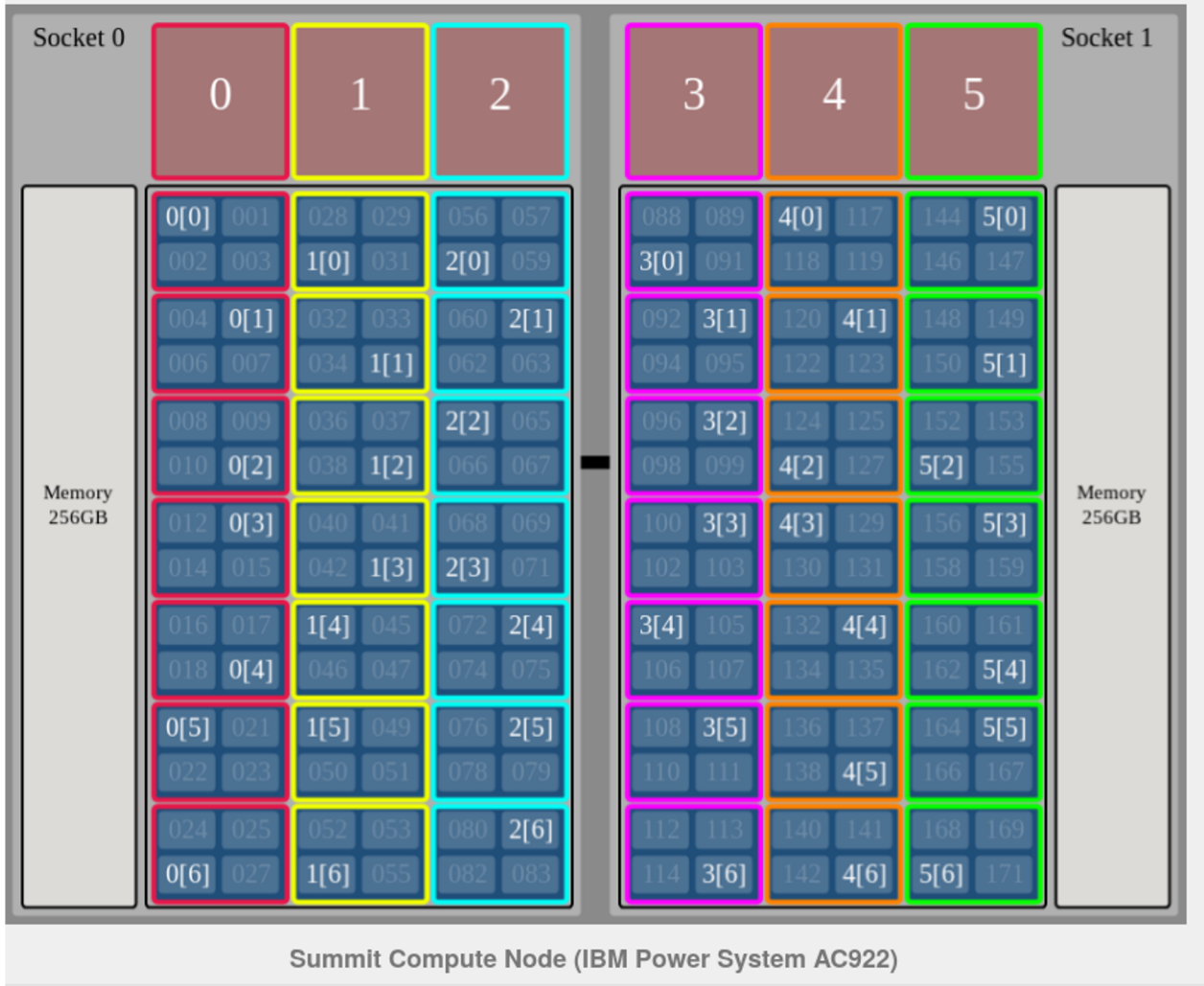


export OMP_NUM_THREADS=1
jsrun --nrs X --tasks_per_rs 7 --cpu_per_rs 7 --gpu_per_rs 1 --rs_per_host 6 --bind packed:1 <exec>

One GPU per MPI Rank (2)

- Expect this to be the most commonly used approach
- Pros:
  - straightforward migration from Titan
  - No extra coding for code that does not handle multiple GPU
- Cons:
  - Assumes similar amount of work among all ranks
  - May leaves some cores or GPUs unused

1 GPU per MPI rank, 6 MPI ranks per node, 7 threads per MPI rank

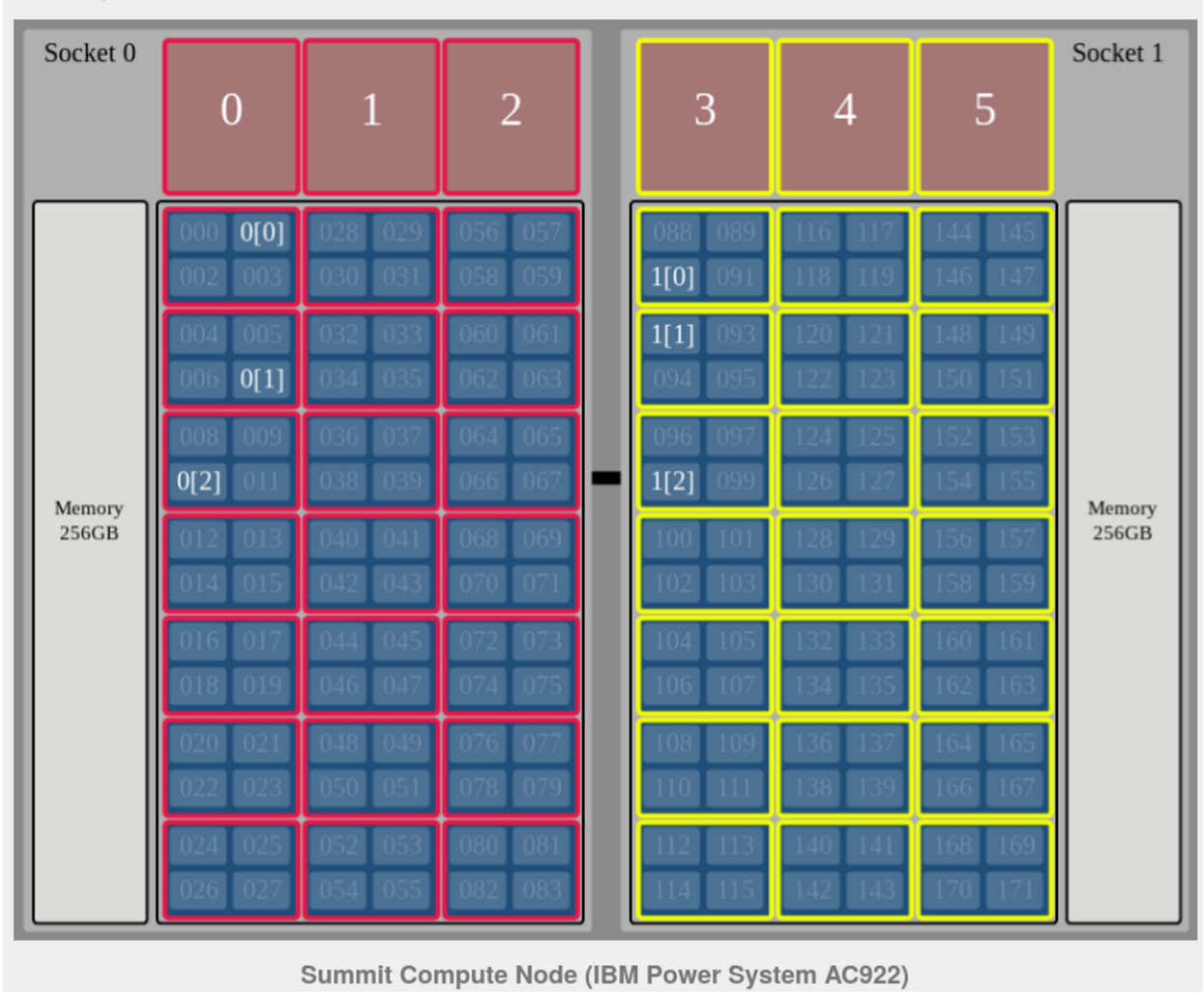


export OMP_NUM_THREADS=7
jsrun --nrs X --tasks_per_rs 1 --cpu_per_rs 7 --gpu_per_rs 1 --rs_per_host 6 --bind packed:7 <exec>

Multiple GPUs per MPI Rank

- Bind 3 - 6 GPUs per MPI rank, e.g.:
  - 2 ranks per node
  - 1 rank per node
- Using programming model constructs to offload to a specific GPU
- Multiple approaches possible

3 GPU per MPI rank, 2 MPI ranks per node, 3 threads per MPI

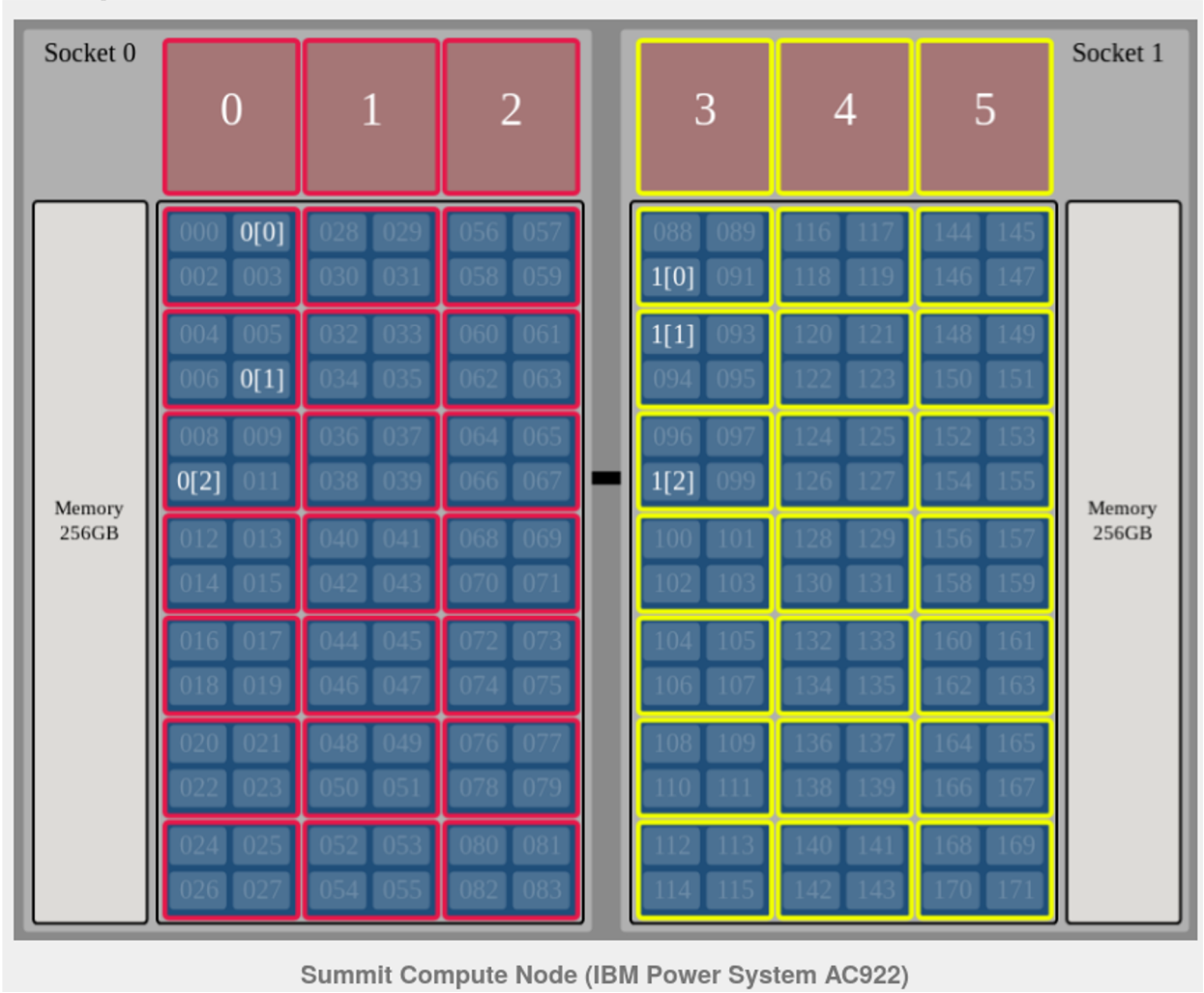


export OMP_NUM_THREADS=3
jsrun --nrs X --tasks_per_rs 1 --cpu_per_rs 21 --gpu_per_rs 3 --rs_per_host 2 --bind packed:7 <exec>

Multiple GPUs per MPI Rank, Explicit Control

- OpenMP + OpenACC:
  - launch 1 OpenMP threads per GPU
  - Within each thread set `acc_set_device_num()`
- OpenMP 4.5:
  - use `device_num()` clause
- CUDA:
  - use `cudaSetDevice()` routine

3 GPU per MPI rank, 2 MPI ranks per node, 3 threads per MPI

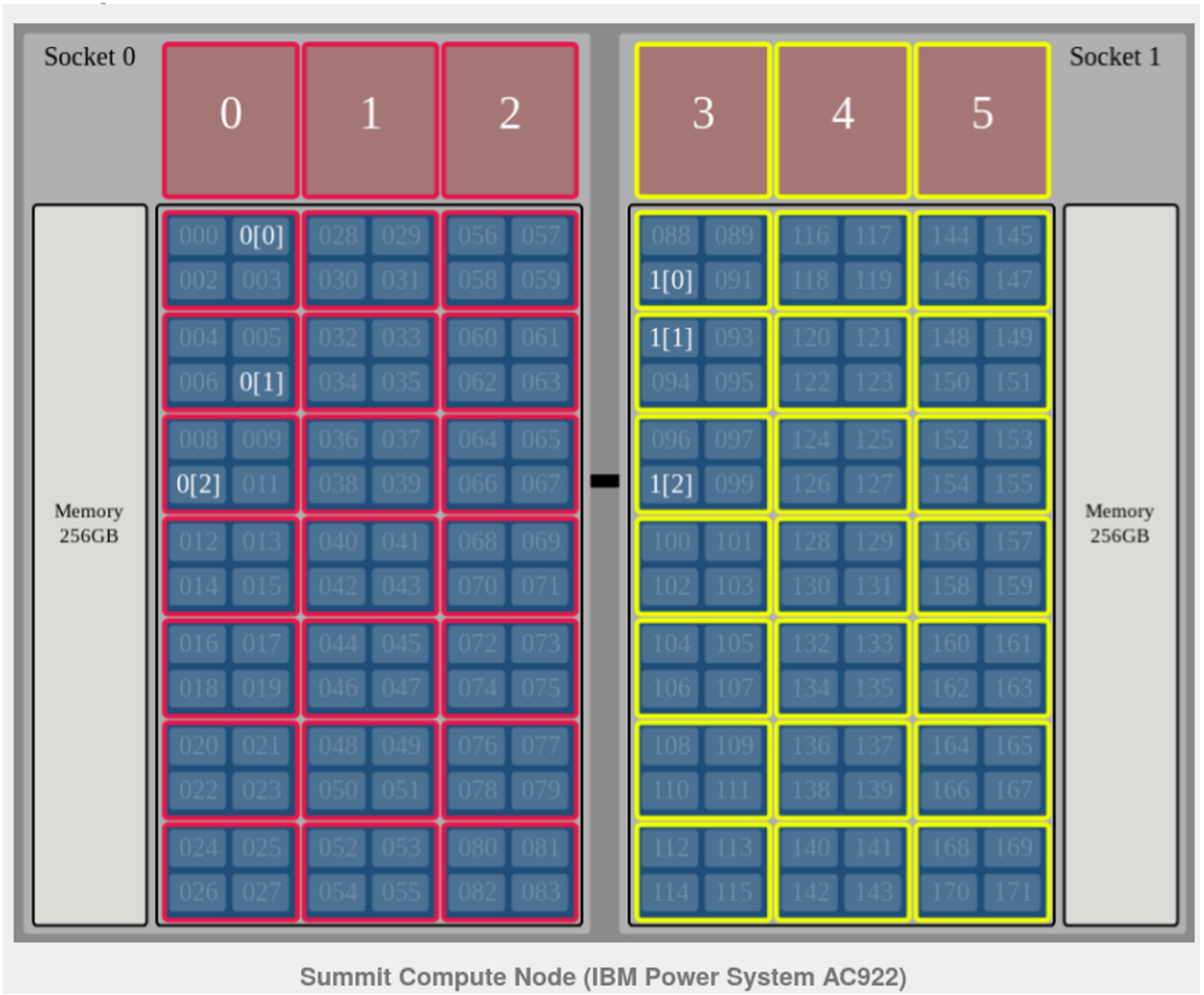


export OMP_NUM_THREADS=3
jsrun --nrs X --tasks_per_rs 1 --cpu_per_rs 21 --gpu_per_rs 3 --rs_per_host 2 --bind packed:7 <exec>

Multiple GPUs per MPI Rank, Implicit Control

- OpenMP and OpenACC:
  - Eventually, compiler + runtime could break up large tasks across multiple GPU automatically
- Task-based execution models are available / under development
- Use Multi-GPU-aware libraries:
  - cuBLASS, cuFFT
- Still need to be careful with process placement

3 GPU per MPI rank, 2 MPI ranks per node, 3 threads per MPI



export OMP_NUM_THREADS=3
jsrun --nrs X --tasks_per_rs 1 --cpu_per_rs 21 --gpu_per_rs 3 --rs_per_host 2 --bind packed:7 <exec>

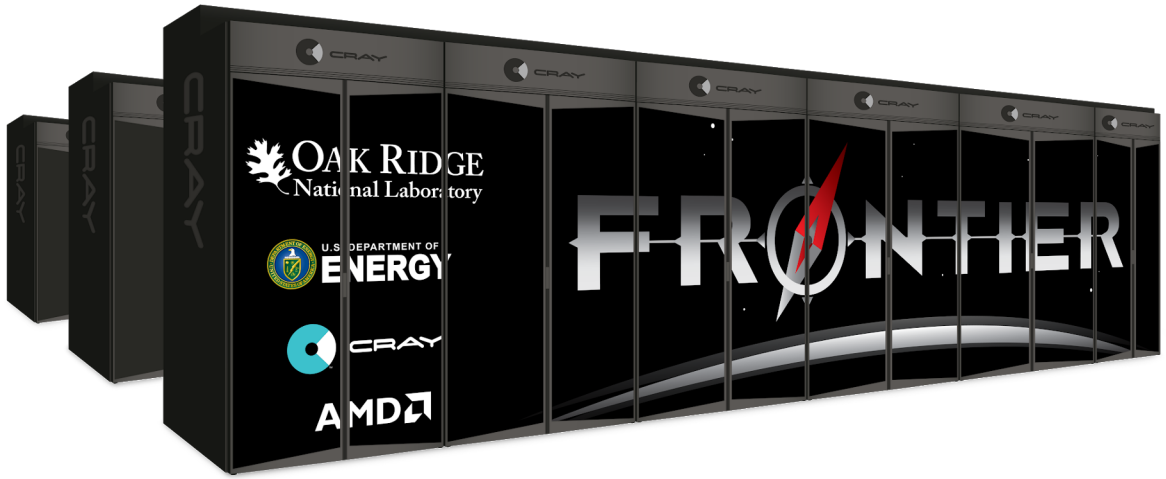
Summit and Scientific Discovery

- Deep Learning for
  - Human System Biology
  - Cancer Research
- Plasma Fusion (XCG)
- Combustion (Raptor)
- Astrophysics (Flash)
- Materials (QMCPACK)


FRONTIER
DIRECTION OF DISCOVERY
ORNL's exascale supercomputer designed to deliver world-leading performance in 2021.

Frontier Overview

- Partnership between ORNL, Cray, and AMD
- Frontier will be delivered in 2021
- Peak performance greater than 1.5 EF
- More than 100 Cray Shasta cabinets
  - connected by Slingshot™ network with adaptive routing, QOS, and congestion control



Frontier Node Architecture

- An AMD EPYC(™) processor with four Radeon Instinct(™) GPU accelerators purpose-built for exascale computing
- Fully connected with high-speed AMD Infinity Fabric links
- Coherent memory across the node
- 100 GB/s injection bandwidth
- Near-node NVM storage

System Comparisons: Titan, Summit, and Frontier

System Specs	Titan	Summit	Frontier
Peak	27 PF	200 PF	~1.5 EF
# cabinets	200	256	Similar foot print
Node	1 AMD Opteron CPU 1 NVIDIA K20X Kepler GPU	2 IBM POWER9™ CPUs 6 NVIDIA Volta GPUs	1 AMD EPYC CPU (HPC and AI Optimized) 4 AMD Radeon Instinct GPUs
On-node interconnect	PCI Gen2 No coherence across the node	NVIDIA NVLINK Coherent memory across the node	AMD Infinity Fabric Coherent memory across the node
System Interconnect	Cray Gemini network 6.4 GB/s	Mellanox Dual-port EDR IB network 25 GB/s	Cray four-port Slingshot network 100 GB/s
Topology	3D Torus	Non-blocking Fat Tree	Dragonfly
Storage	32 PB, 1 TB/s, Lustre Filesystem	250 PB, 2.5 TB/s, IBM Spectrum Scale™ with GPFS™	2-4x performance and capacity of Summit's I/O subsystem.
Near-node NVM (storage)	No	Yes	Yes

Programming Environment and Migration Path

	Summit	Frontier
Compilers	GCC, IBM XL, PGI	GCC, Cray (CCE), AMD ROCm,
Programming Model	CUDA C / C++	HIP C/C++
	OpenACC	OpenMP 5.x
	OpenMP	OpenMP 5.x
	Fortran with CUDA C/C++	Fortran with HIP C/C++
	CUDA Fortran	Fortran with HIP C/C++, OpenMP 5.x

Summit is a premier development platform for Frontier

Programming Environment and Migration Path (2)

- HIP (heterogenous-compute Interface for Portability) is an API developed by AMD for portable code on AMD and NVIDIA GPU
  - uses CUDA or ROCm under the hood
- The API is very similar to CUDA
- AMD has developed a “hipify” tool to convert from CUDA to HIP
- HIP is available on Summit and is updated regularly

Acknowledgments

- Entire OLCF team, particularly
  - Judy Hill, Wayne Joubert, Bronson Messer, Matt Norman, Chris Fuson, Reuben Budiardja, Tom Papatheodore, Chris Zimmer, Jack Morrison, Veronica Melesse Vergara.

This work was performed under the auspices of the U.S. DOE by Oak Ridge Leadership Computing Facility at ORNL under contracts DEAC05-00OR22725

Resources

- More info on Summit:
  - Summit user guide: https://www.olcf.ornl.gov/for-users/system-user-guides/summit/
  - OLCF training archive: https://www.olcf.ornl.gov/for-users/training/training-archive/
  - Vazhkudai, *et. al.* The Design, Deployment, and Evaluation of the CORAL Pre-Exascale Systems. SC18 Proceedings.

- For latest on Frontier:
  - https://www.olcf.ornl.gov/frontier/#4

Thanks

E-mail: pophaless@ornl.gov