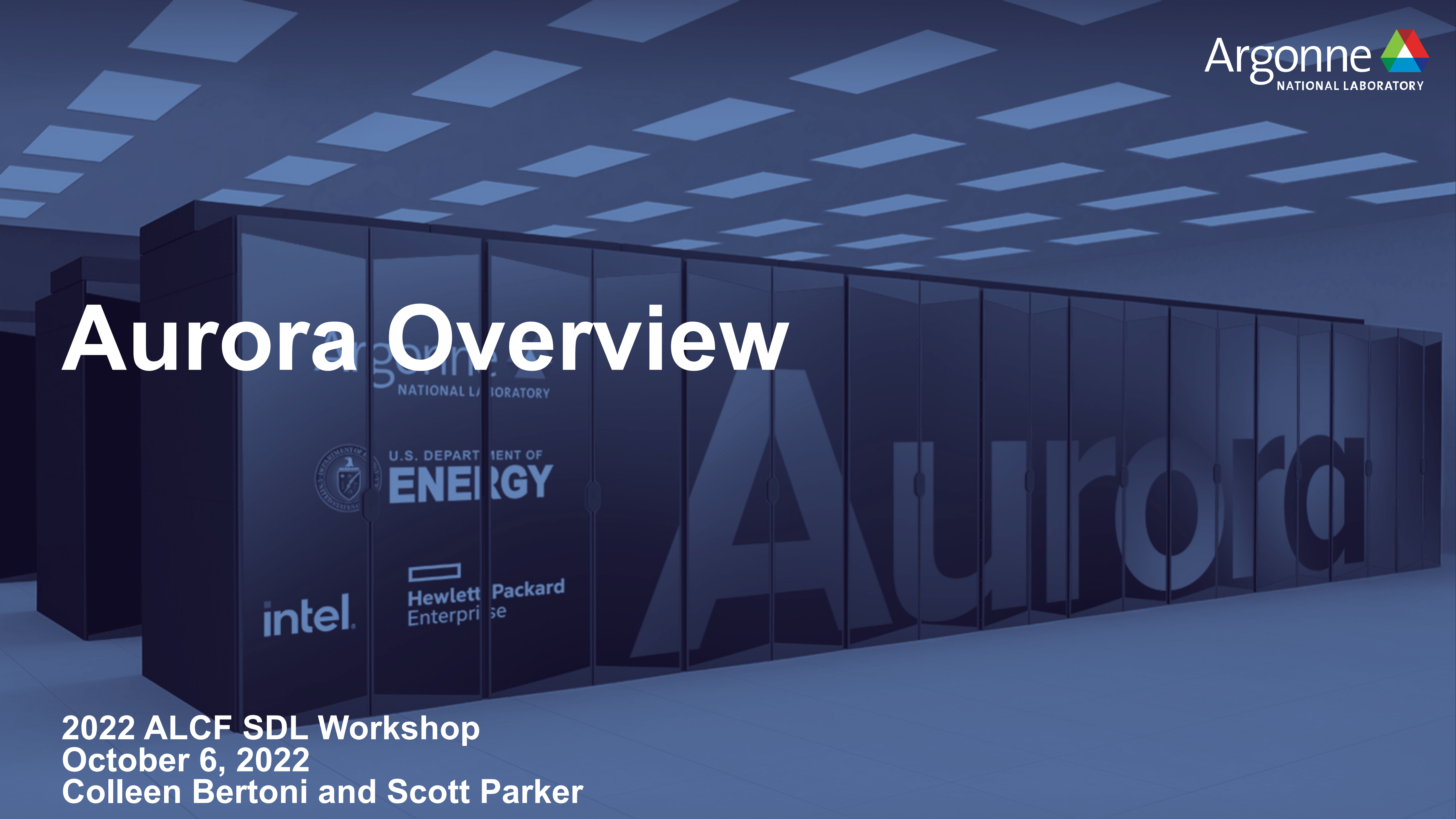
Aurora Overview



2022 ALCF SDL Workshop
October 6, 2022

Colleen Bertoni and Scott Parker

Argonne NATIONAL LABORATORY
U.S. DEPARTMENT OF ENERGY
intel
Hewlett Packard Enterprise
Aurora

Aurora

Leadership Computing Facility
Exascale Supercomputer

Peak Performance
≧ 2 Exaflops DP

Intel GPU
Ponte Vecchio (PVC)

Intel Xeon Processor
Sapphire Rapids with High Bandwidth Memory

Platform
HPE Cray-Ex

Compute Node
2 Xeon SPR+HBM processors
6 Ponte Vecchio GPUs
Node Unified Memory Architecture
8 fabric endpoints

GPU Architecture
Intel XeHPC architecture
High Bandwidth Memory Stacks

Node Performance
>130 TF

System Size
>9,000 nodes

Aggregate System Memory
>10 PB aggregate System Memory

System Interconnect
HPE Slingshot 11
Dragonfly topology with adaptive routing

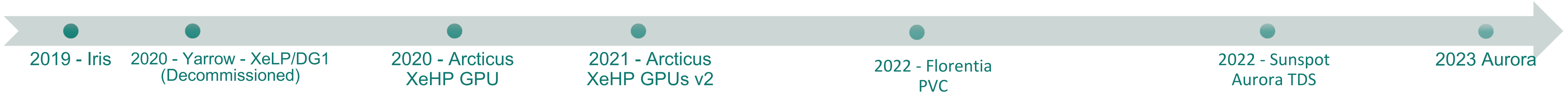
Network Switch
25.6 Tb/s per switch (64 200 Gb/s ports)
Links with 25 GB/s per direction

High-Performance Storage
220 PB
≧25 TB/s DAOS bandwidth

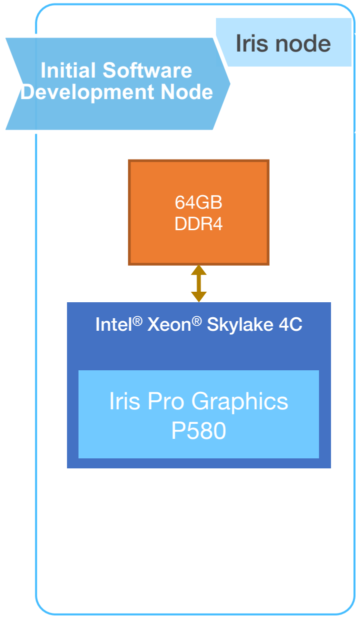
Software Environment
- C/C++
- Fortran
- SYCL/DPC++
- OpenMP offload
- Kokkos
- RAJA
- Intel Performance Tools

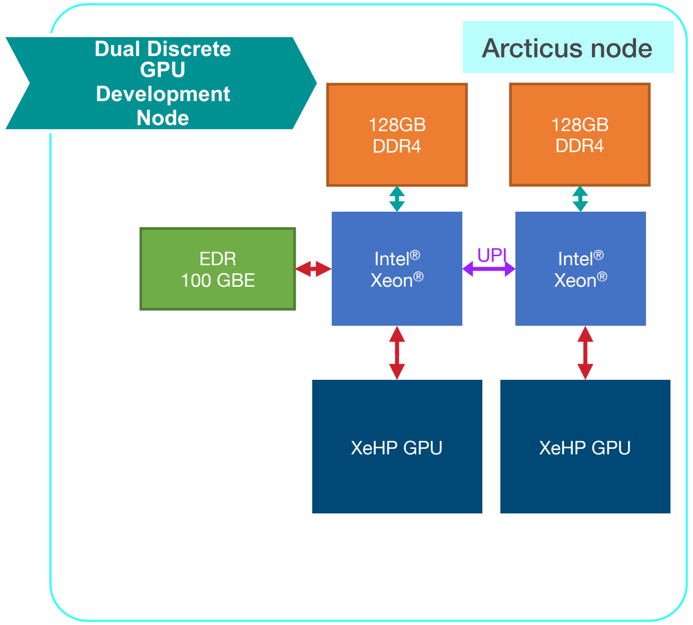
Aurora Cabinets Installed at Argonne

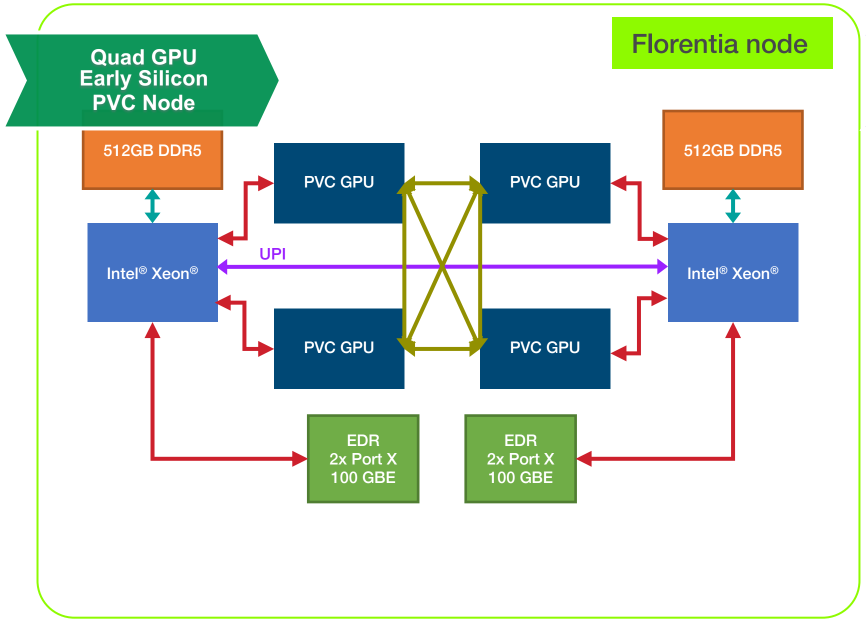
JLSE Testbeds to Aurora Node

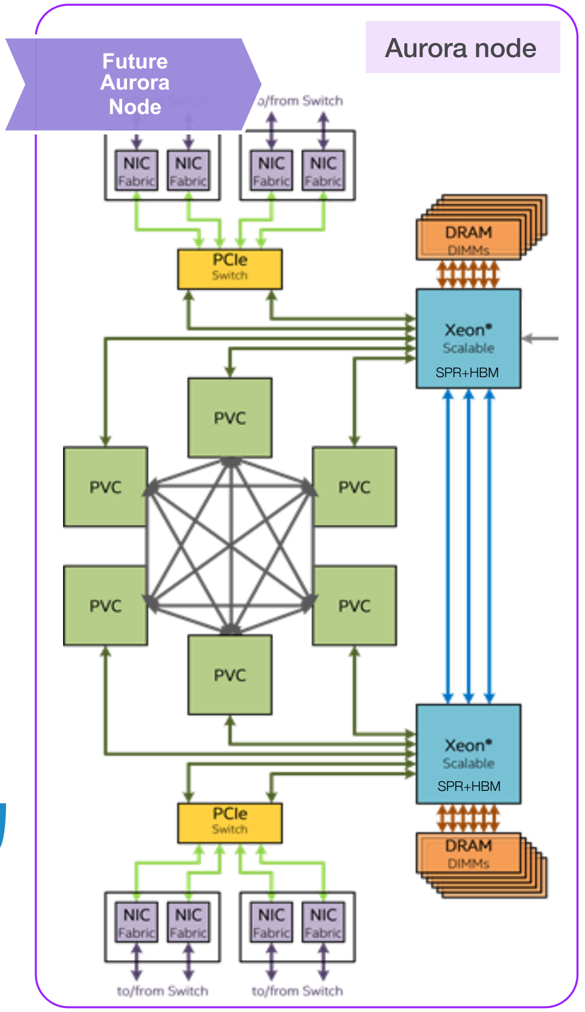


2019 - Iris
2020 - Yarrow - XeLP/DG1 (Decommissioned)
2020 - Arcticus XeHP GPU
2021 - Arcticus XeHP GPUs v2
2022 - Florentia PVC
2022 - Sunspot Aurora TDS
2023 Aurora



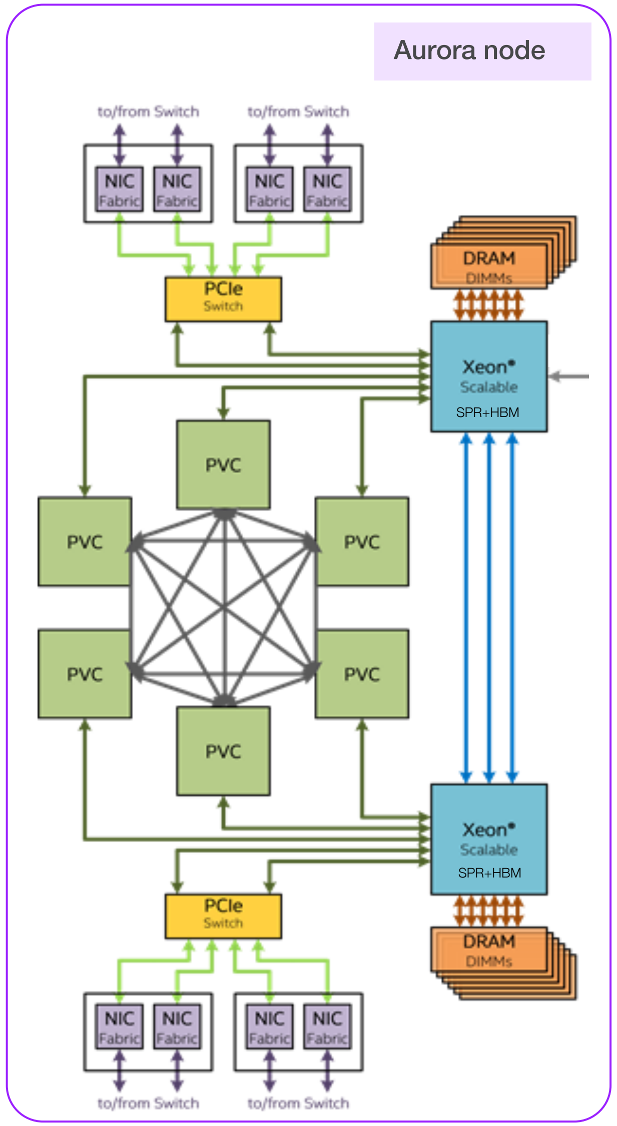






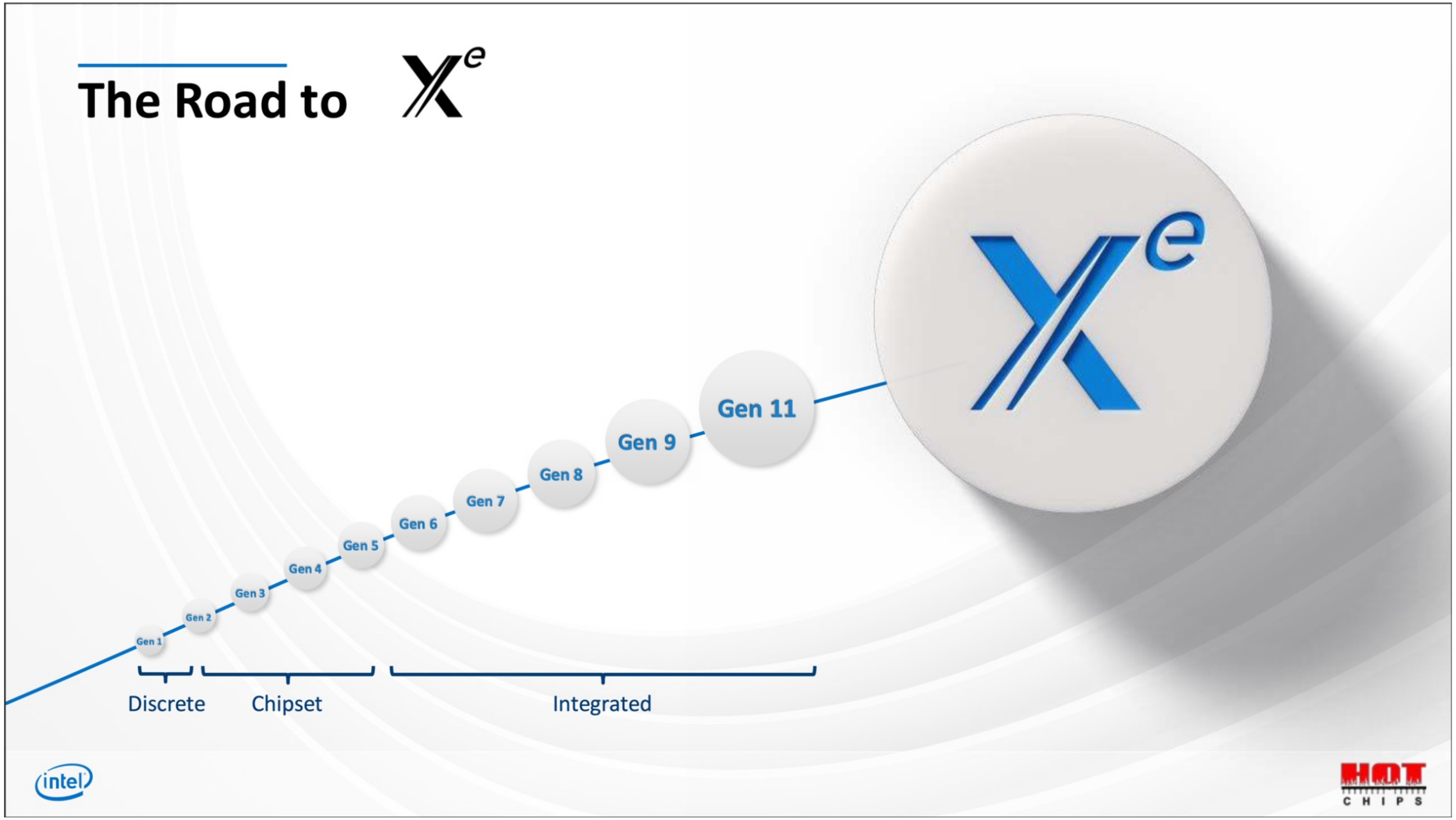
Testbeds for Applications Development and Software Testing

Aurora Compute Node

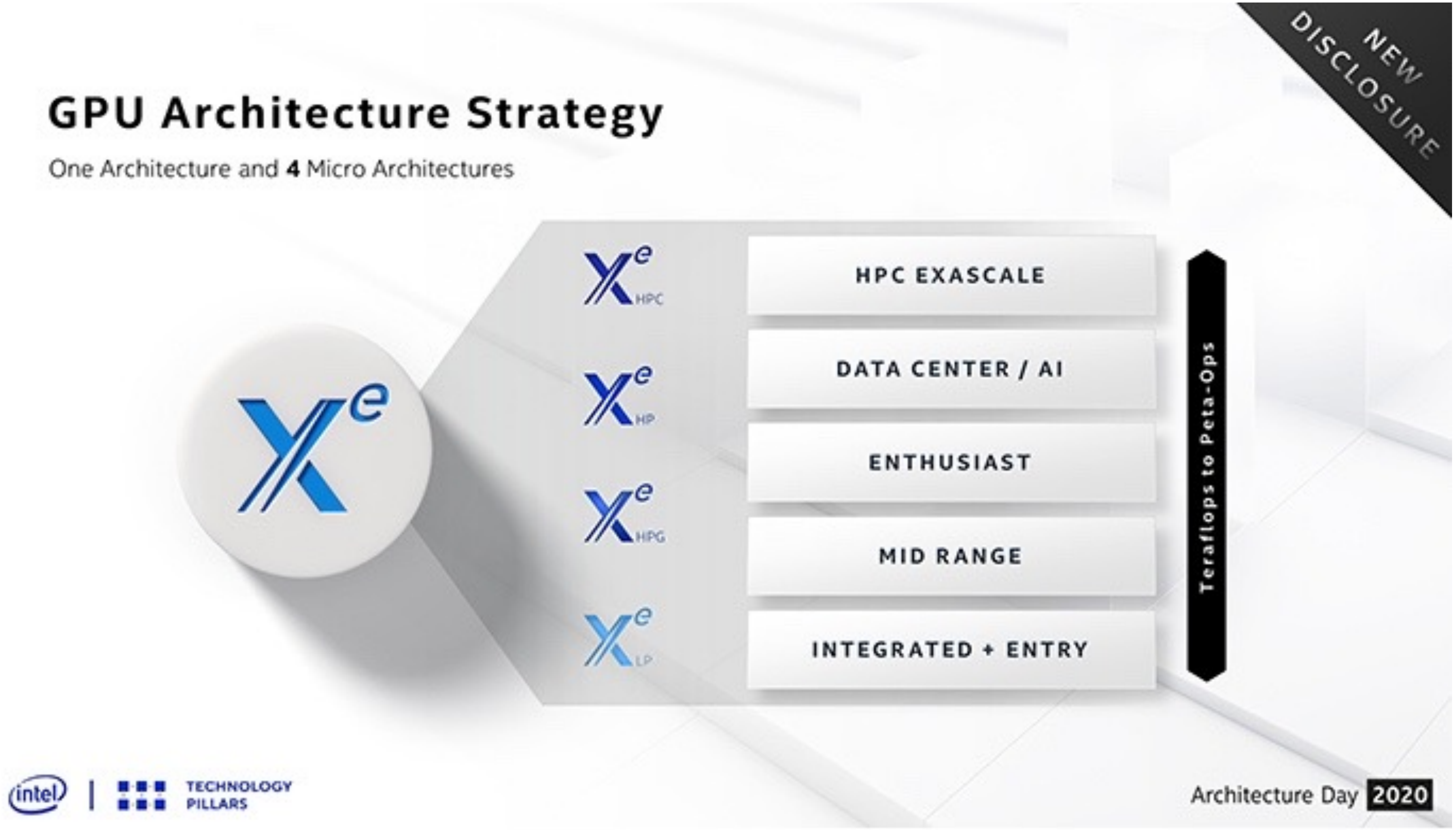


- 6 X^e Architecture based GPUs (Ponte Vecchio)
  - All to all connection
- 2 Intel Xeon (Sapphire Rapids) processors
- Unified Memory Architecture across CPUs and GPUs
- 8 Slingshot Fabric endpoints

The Evolution of Intel GPUs



The Evolution of Intel GPUs



X^e Vector Engine (Execution Unit)

- The vector engine executes instructions
  - Register file
  - Multiple issue ports
  - Vector pipelines
    - Float Point
    - Integer
    - Extended Math
    - FP 64 (optional)
  - Matrix Extension (XMX)
  - Thread control
  - Branch
  - Send (memory)

Vector
Engine
XMX

Image: https://www.intel.com/content/www/us/en/develop/documentation/oneapi-gpu-optimization-guide/top/xe-arch.html

X^e-Core (Subslice)

- A X^e Core contains:
  - Vector engines (execution units)
  - Matrix engines (XMX)
  - Thread dispatch
  - Instruction cache
  - L1 and shared local memory
  - Load/Store

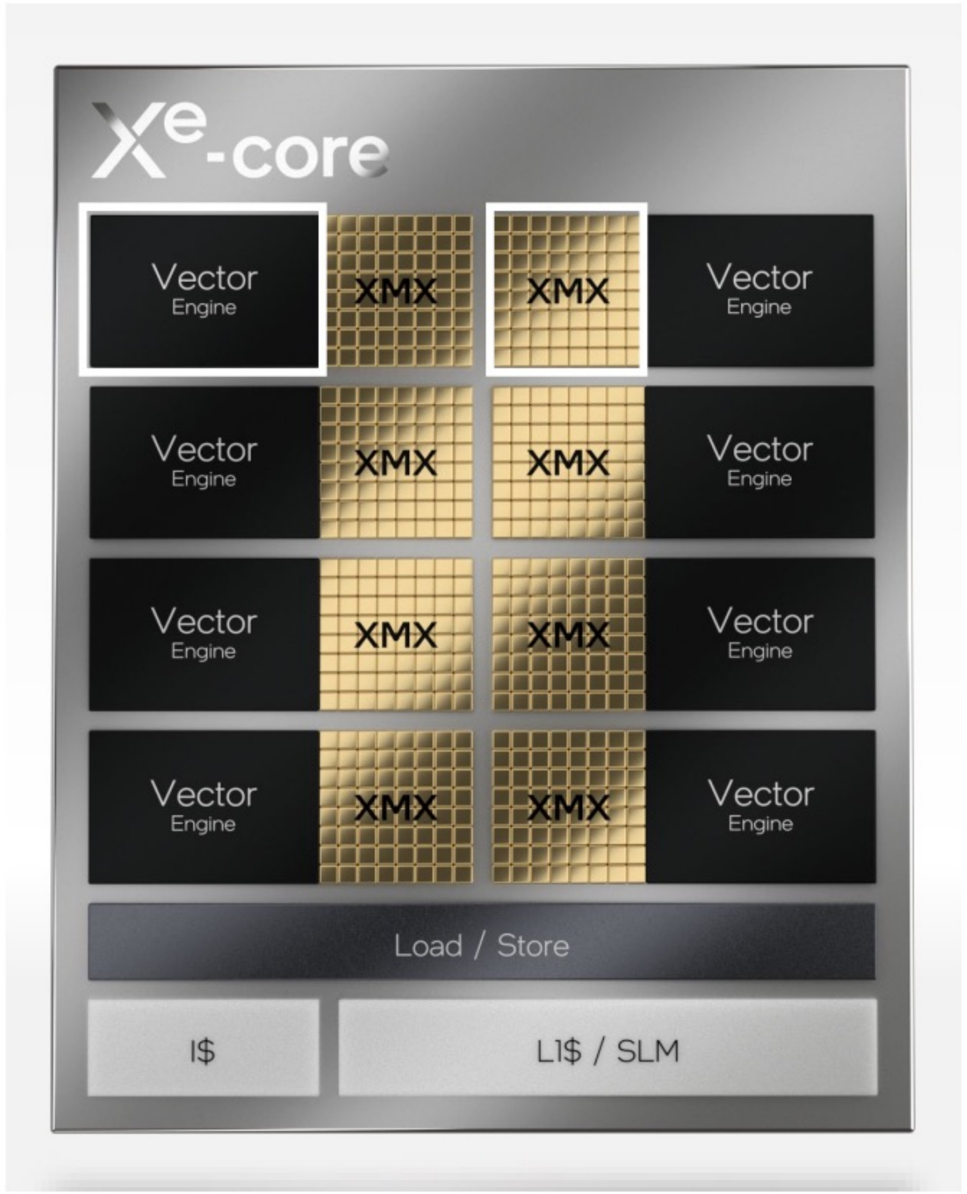


Image: https://www.intel.com/content/www/us/en/develop/documentation/oneapi-gpu-optimization-guide/top/xe-arch.html

X^e-Slice

- ❑ A X^e-Slice contains
  - ❑ Composed of X^e-cores

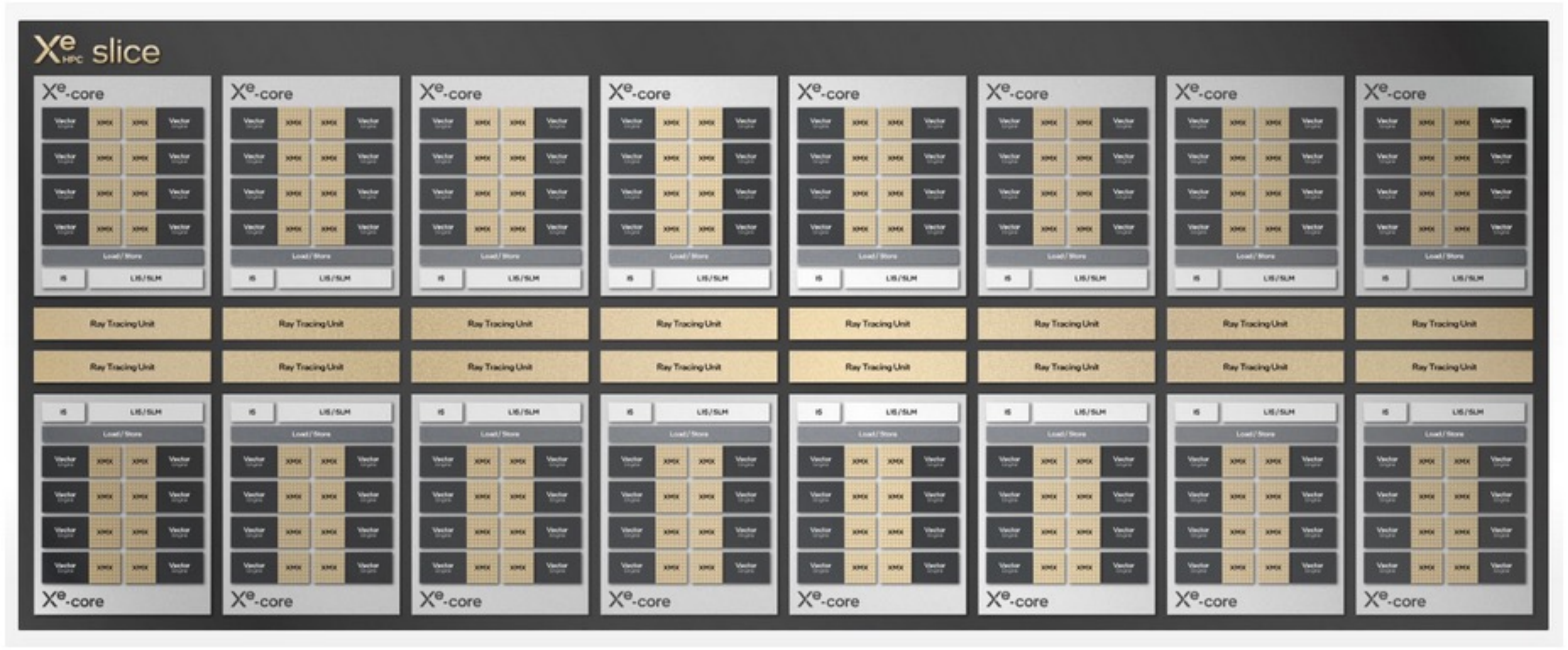


Image: https://www.intel.com/content/www/us/en/develop/documentation/oneapi-gpu-optimization-guide/top/xe-arch.html

X^{e}-Stack (Tile)

- ❑ A X^{e}-Stack contains
  - ❑ Variable number of X^{e}-Slices
  - ❑ Shared L2 cache
  - ❑ Memory controllers
  - ❑ Media engine
  - ❑ X^{e}-links high-speed coherent fabric (GPU to GPU)

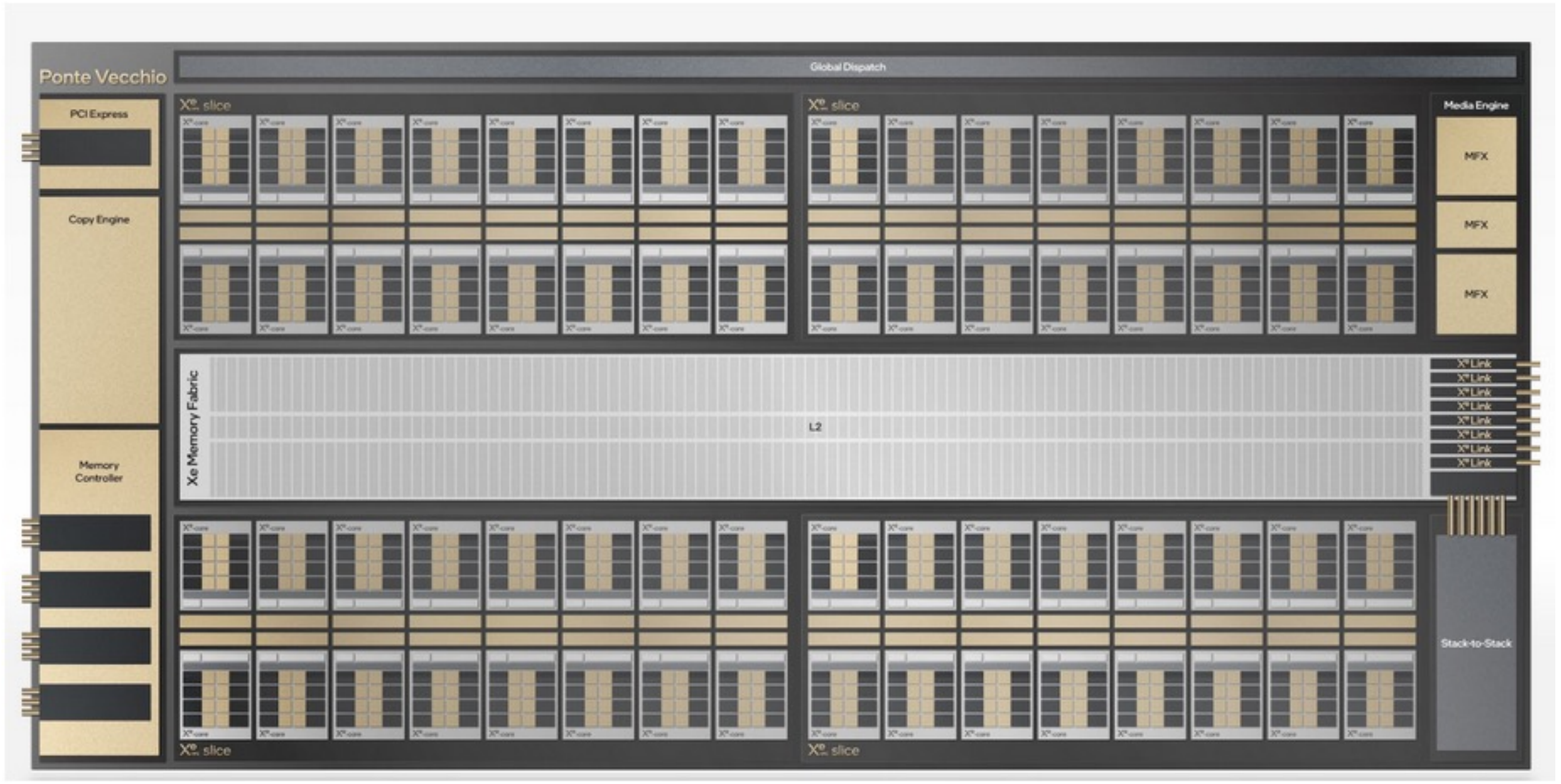


Image: https://www.intel.com/content/www/us/en/develop/documentation/oneapi-gpu-optimization-guide/top/xe-arch.html

High Level X^e Architecture

- ❑ X^e GPU is composed of
  - ❑ X^e Stacks
  - ❑ Memory Fabric

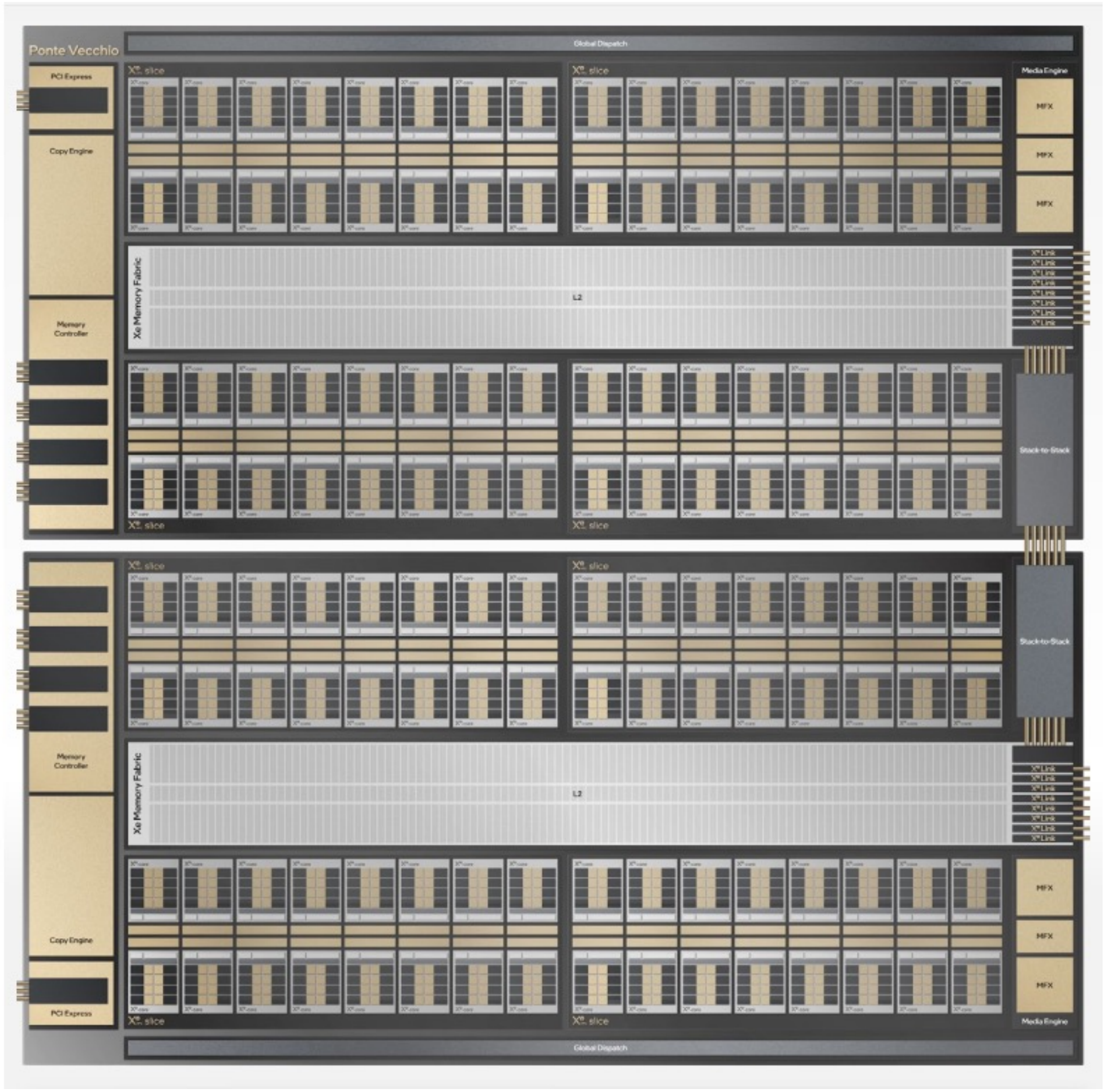
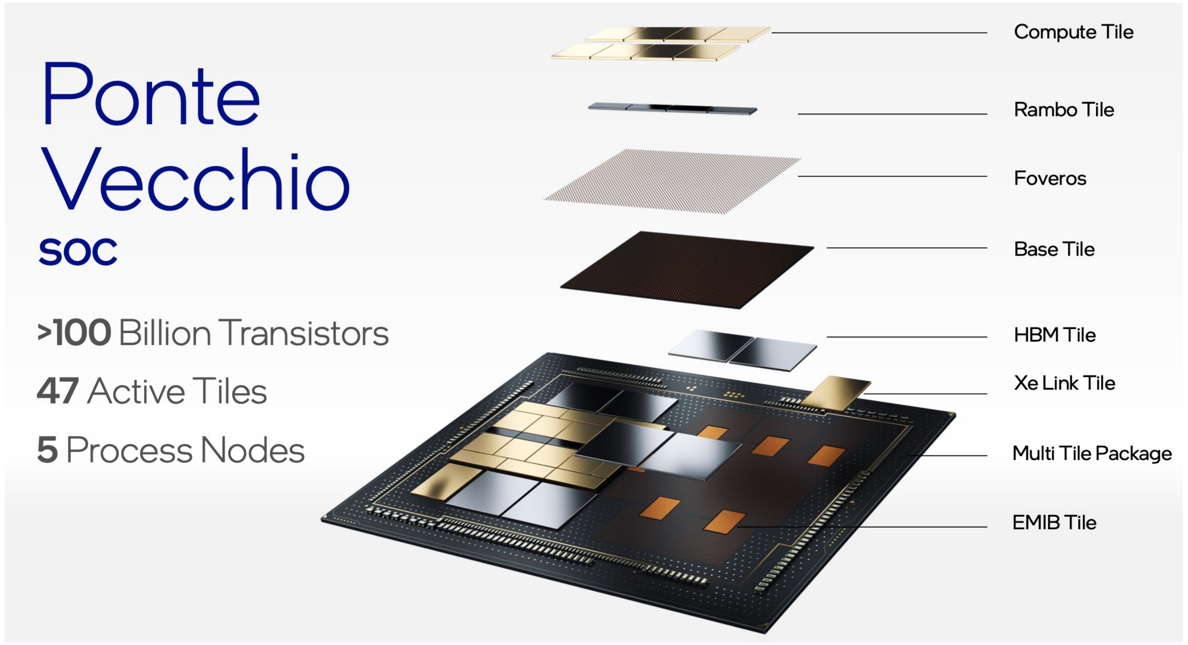


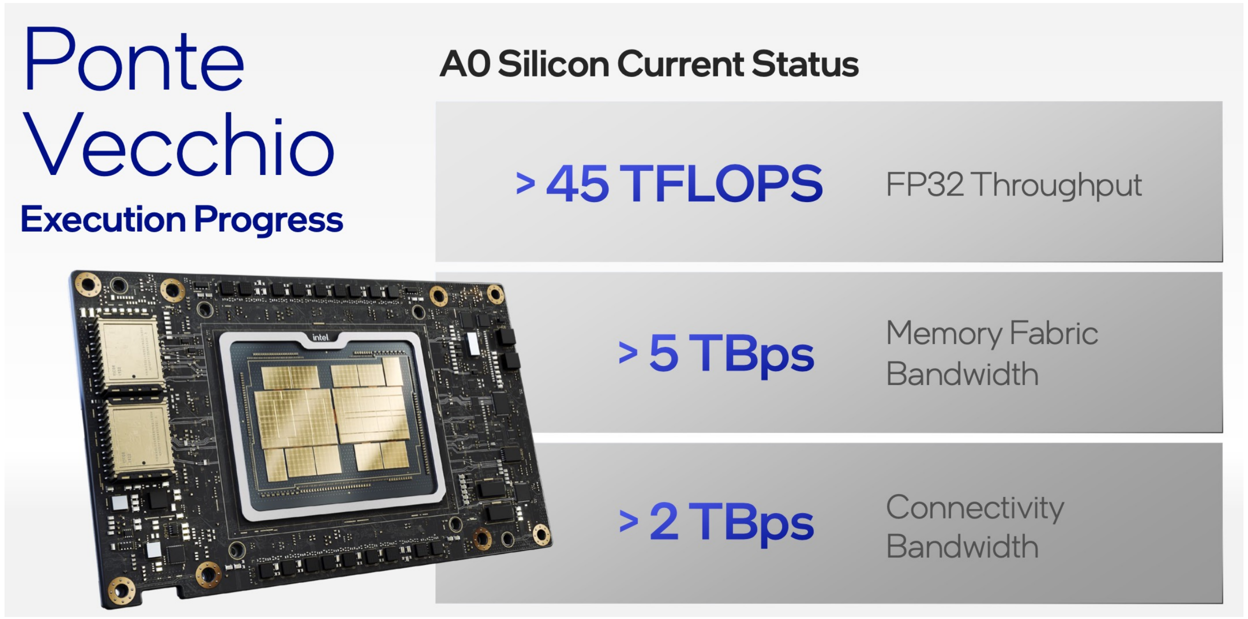
Image: https://www.intel.com/content/www/us/en/develop/documentation/oneapi-gpu-optimization-guide/top/xe-arch.html

Intel Ponte Vecchio (XeHPC) GPU

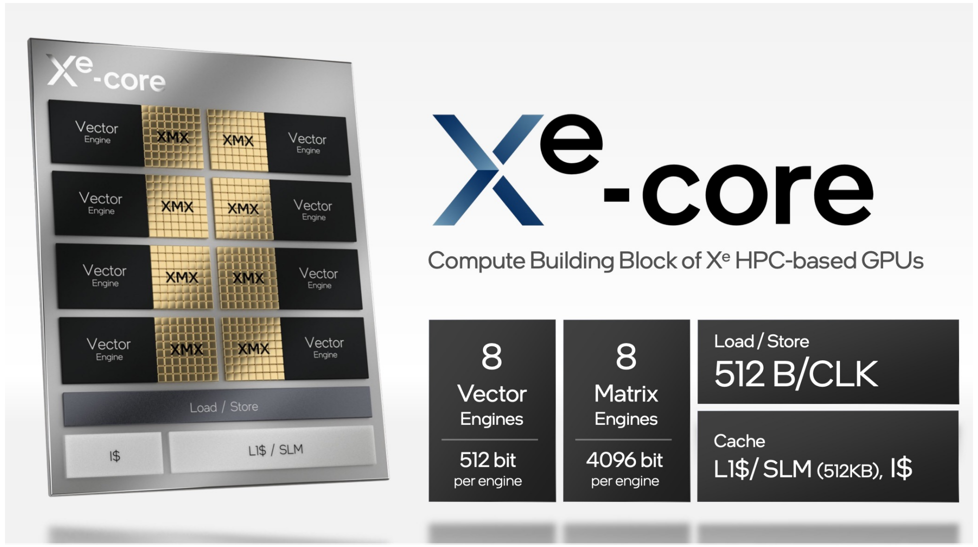
Intel provided an introduction to the Ponte Vecchio GPU at their 2021 Intel Architecture Day event

- https://www.intel.com/content/www/us/en/newsroom/resources/press-kit-architecture-day-2021.html





Intel Ponte Vecchio Architectural Components

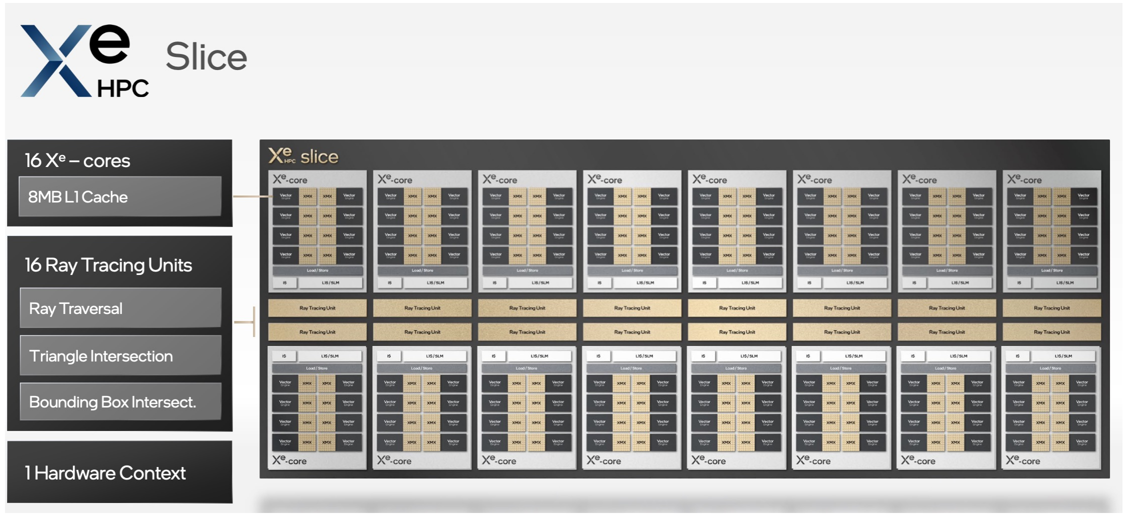


Xe-core

Compute Building Block of Xe HPC-based GPUs

8 Vector Engines	8 Matrix Engines	Load / Store
512 bit per engine	4096 bit per engine	512 B/CLK
		Cache L1$/ SLM (512KB), I$

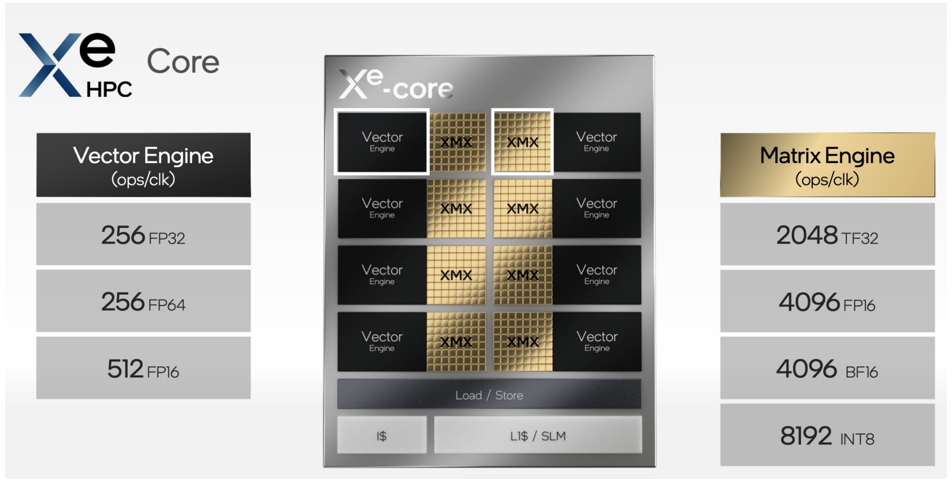
$\rightarrow$



Xe HPC Slice

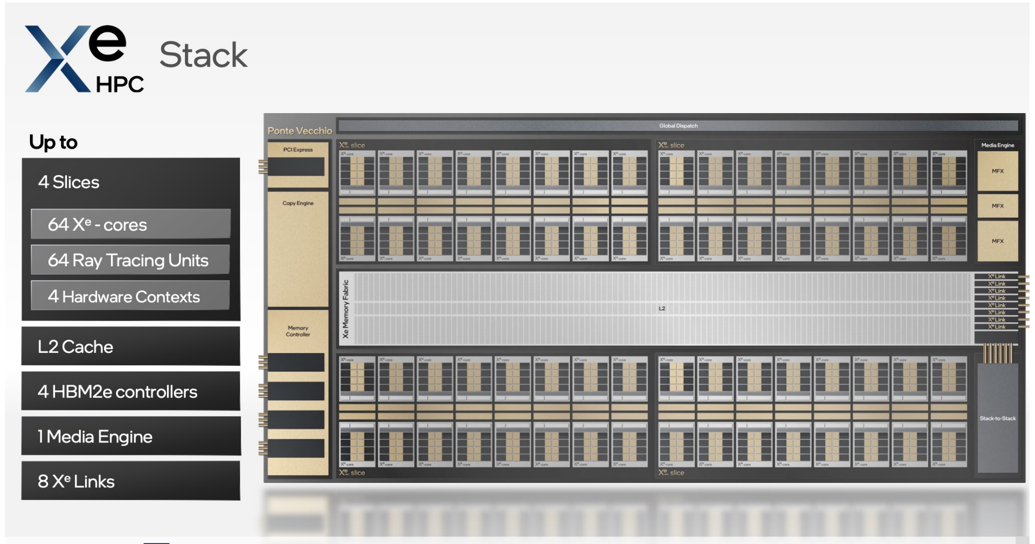
- 16 Xe – cores
  - 8MB L1 Cache
- 16 Ray Tracing Units
  - Ray Traversal
  - Triangle Intersection
  - Bounding Box Intersect.
- 1 Hardware Context

$\downarrow$



Xe HPC Core

Vector Engine (ops/clk)	Matrix Engine (ops/clk)
256 FP32	2048 TF32
256 FP64	4096 FP16
512 FP16	4096 BF16
	8192 INT8



Xe HPC Stack

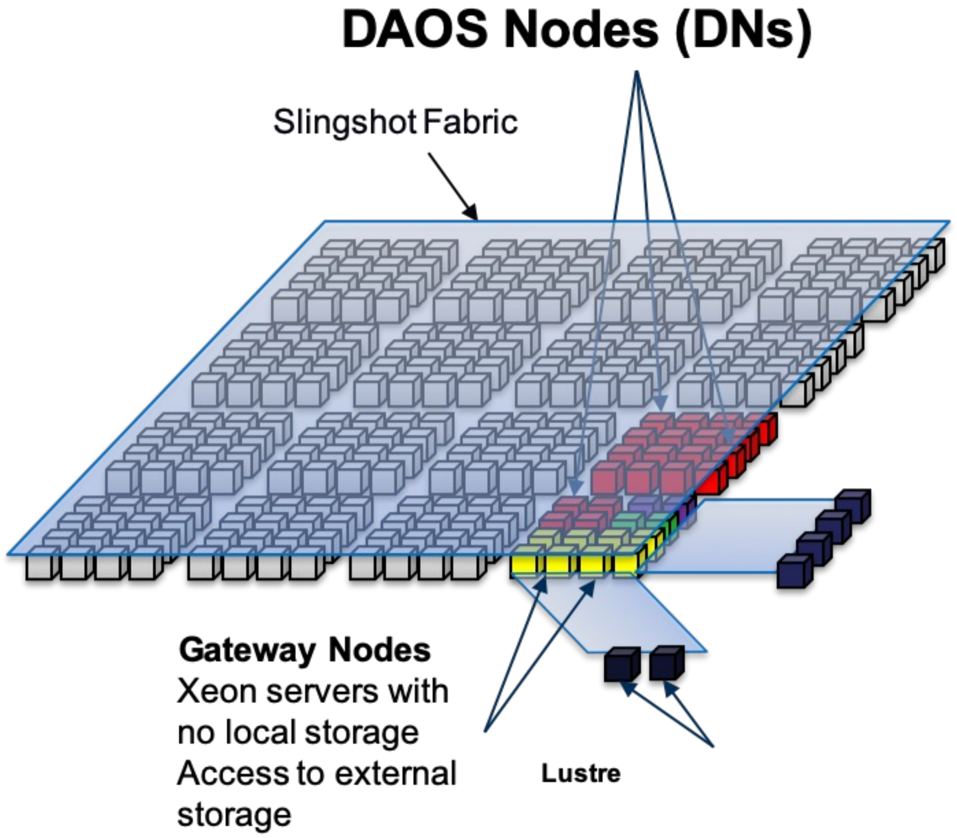
Up to

- 4 Slices
  - 64 Xe - cores
  - 64 Ray Tracing Units
  - 4 Hardware Contexts
- L2 Cache
- 4 HBM2e controllers
- 1 Media Engine
- 8 Xe Links

Argonne NATIONAL LABORATORY

Distributed Asynchronous Object Store (DAOS)

- ❑ Primary storage system for Aurora
- ❑ Offers high performance in bandwidth and IO operations
  - ❑ 230 PB capacity
  - ❑ ≥ 25 TB/s
- ❑ Provides a flexible storage API that enables new I/O paradigms
- ❑ Provides compatibility with existing I/O models such as POSIX, MPI-IO and HDF5
- ❑ Open source storage solution


DAOS Nodes (DNs)
Slingshot Fabric
Gateway Nodes
Xeon servers with no local storage
Access to external storage
Lustre

Pre-exascale and Exascale US Landscape

System	Delivery	CPU + Accelerator Vendor
Summit	2018	IBM + NVIDIA
Sierra	2018	IBM + NVIDIA
Perlmutter	2021	AMD + NVIDIA
Frontier	2021	AMD + AMD
Polaris	2021	AMD + NVIDIA
Aurora	2022	Intel + Intel
El Capitan	2023	AMD + AMD

- Heterogenous Computing (CPU + Accelerator)
- Varying vendors

oneAPI

- Industry specification from Intel (https://www.oneapi.com/spec/)
  - Language and libraries to target programming across diverse architectures (DPC++, APIs, low level interface)
- Intel oneAPI products and toolkits (https://software.intel.com/ONEAPI)
  - Languages
    - Fortran (w/ OpenMP 5+)
    - C/C++ (w/ OpenMP 5+)
    - DPC++
    - Python
  - Libraries
    - oneAPI MKL (oneMKL)
    - oneAPI Deep Neural Network Library (oneDNN)
    - oneAPI Data Analytics Library (oneDAL)
    - MPI
  - Tools
    - Intel Advisor
    - Intel VTune
    - Intel Inspector

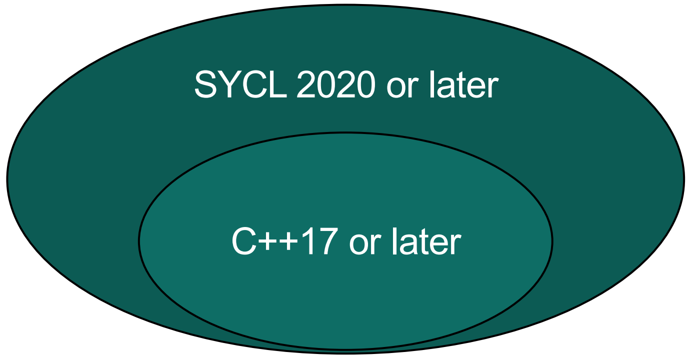
https://software.intel.com/oneapi

Available Aurora Programming Models

- Aurora applications may use:
  - DPC++/SYCL
  - OpenMP
  - Kokkos
  - Raja
  - OpenCL
- Experimental
  - HIP
- Not available on Aurora:
  - CUDA
  - OpenACC

DPC++ (Data Parallel C++) and SYCL

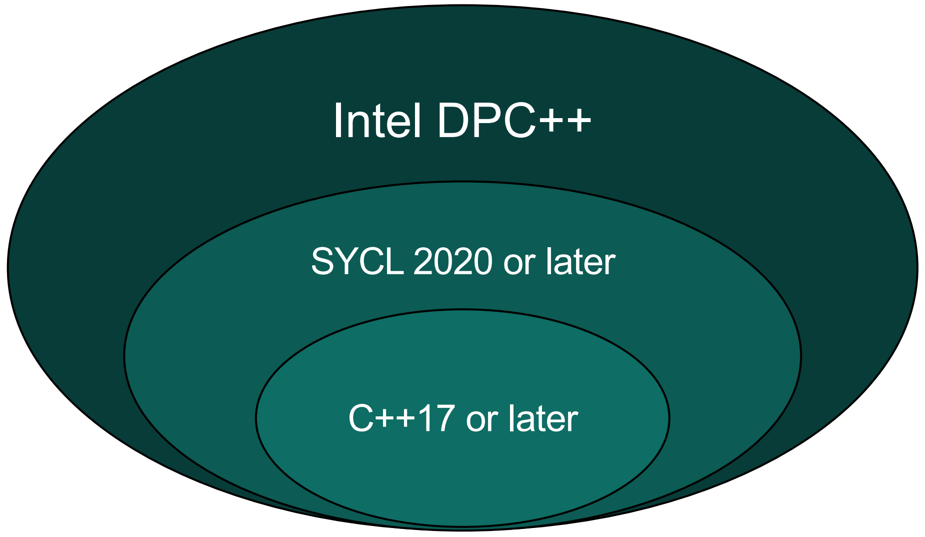
- SYCL
  - Standard developed by Khronos and announced in 2014
  - The latest SYCL specification (SYCL 2020) was release in 2021
  - SYCL is a C++ based abstraction layer (standard C++17)
  - Builds on OpenCL **concepts** (but single-source)
  - *SYCL is designed to be as close to standard C++ as possible*



DPC++ (Data Parallel C++) and SYCL

- SYCL
  - Standard developed by Khronos and announced in 2014
  - The latest SYCL specification (SYCL 2020) was release in 2021
  - SYCL is a C++ based abstraction layer (standard C++17)
  - Builds on OpenCL **concepts** (but single-source)
  - *SYCL is designed to be as close to standard C++ as possible*

- DPC++
  - Part of Intel oneAPI specification and Intel's implementation of SYCL
  - Intel extension of SYCL to support new innovative features
  - Open source and available on github
  - Contains a Plugin Interface (PI) to allow DPC++ to run on multiple devices

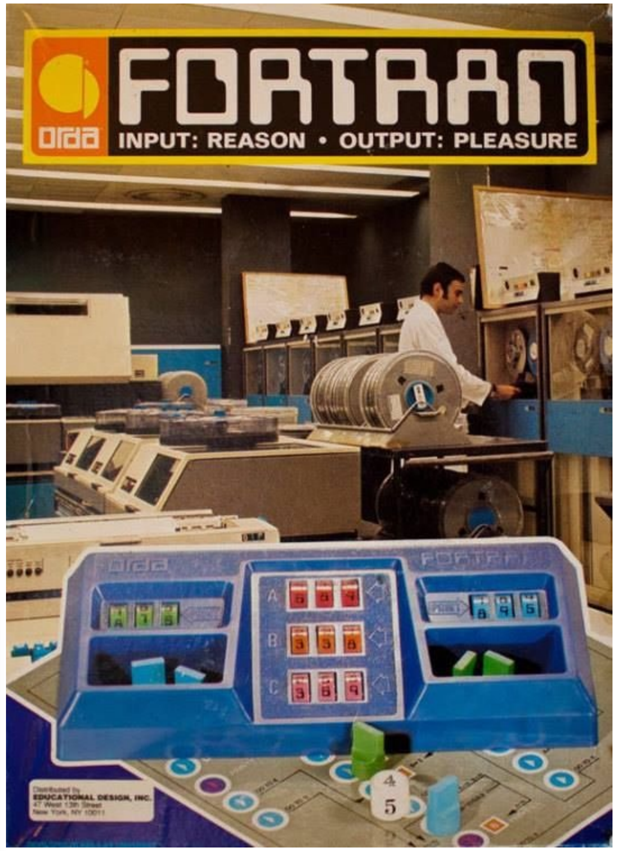


OpenMP

- OpenMP is a widely supported and utilized programming model
- OpenMP 5 constructs will provide directives based programming model for Intel GPUs
- Available for C, C++, and Fortran and optimized for Aurora
- Current OpenMP 5.1 spec supports offloading to an accelerator/GPU
  - Support started with OpenMP 4
- OpenMP with offload support offers a potential path to developing performance portable applications
- Multiple compilers and vendors providing OpenMP implementations
- Community has a consensus what is the “most common” subset of OpenMP features to be supported on devices.
  - OpenMP features inappropriate to GPUs are often not implemented

Intel Fortran for Aurora

- ❑ Fortran 2008
- ❑ OpenMP 5
- ❑ New compiler—LLVM backend
  - ❑ Strong Intel history of optimizing Fortran compilers
- ❑ Beta available today in OneAPI toolkits

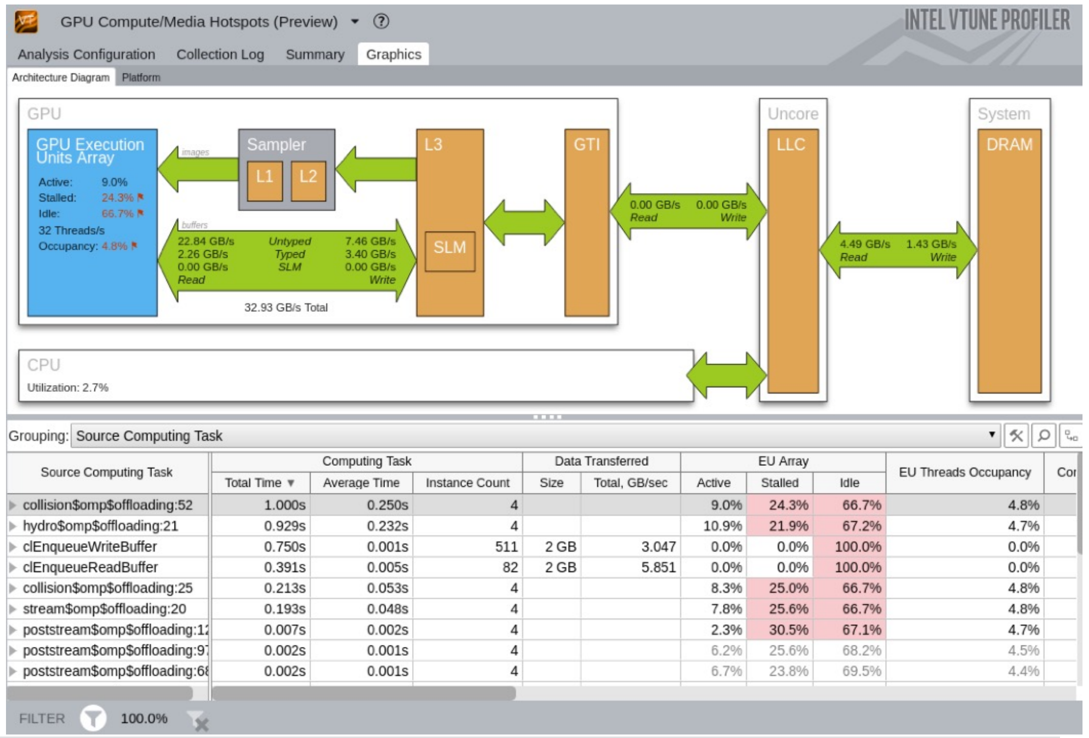


https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/fortran-compiler.html

Intel VTune and Advisor

- ❑ Vtune Profiler
  - ❑ Widely used performance analysis tool
  - ❑ Supports analysis on Intel GPUs

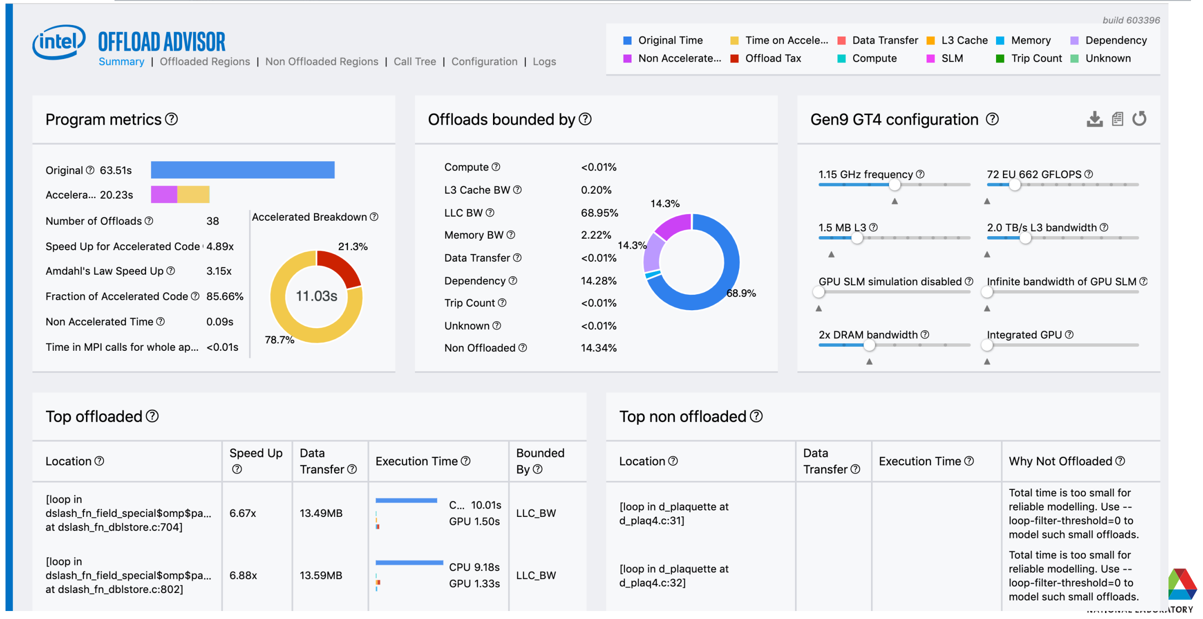
- ❑ Advisor
  - ❑ Provides roofline analysis
  - ❑ Offload analysis will identify components for profitable offload
    - ❑ Measure performance and behavior of original code
    - ❑ Model specific accelerator performance to determine offload opportunities
    - ❑ Considers overhead from data transfer and kernel launch



Grouping: Source Computing Task

Source Computing Task	Computing Task			Data Transferred		EU Array			EU Threads Occupancy	Cor
	Total Time ▼	Average Time	Instance Count	Size	Total, GB/sec	Active	Stalled	Idle		
collision$omp$offloading:52	1.000s	0.250s	4			9.0%	24.3%	66.7%	4.8%	
hydro$omp$offloading:21	0.929s	0.232s	4			10.9%	21.9%	67.2%	4.7%	
clEnqueueWriteBuffer	0.750s	0.001s	511	2 GB	3.047	0.0%	0.0%	100.0%	0.0%	
clEnqueueReadBuffer	0.391s	0.005s	82	2 GB	5.851	0.0%	0.0%	100.0%	0.0%	
collision$omp$offloading:25	0.213s	0.053s	4			8.3%	25.0%	66.7%	4.8%	
stream$omp$offloading:20	0.193s	0.048s	4			7.8%	25.6%	66.7%	4.8%	
poststream$omp$offloading:1	0.007s	0.002s	4			2.3%	30.5%	67.1%	4.7%	
poststream$omp$offloading:9	0.002s	0.001s	4			6.2%	25.6%	68.2%	4.5%	
poststream$omp$offloading:6	0.002s	0.001s	4			6.7%	23.8%	69.5%	4.4%	

FILTER 100.0%



Program metrics

Original	63.51s
Accelera...	20.23s
Number of Offloads	38
Speed Up for Accelerated Code	4.89x
Amdahl's Law Speed Up	3.15x
Fraction of Accelerated Code	85.66%
Non Accelerated Time	0.09s
Time in MPI calls for whole ap...	<0.01s

Offloads bounded by

Compute	<0.01%
L3 Cache BW	0.20%
LLC BW	68.95%
Memory BW	2.22%
Data Transfer	<0.01%
Dependency	14.28%
Trip Count	<0.01%
Unknown	<0.01%
Non Offloaded	14.34%

Gen9 GT4 configuration

- 1.15 GHz frequency
- 72 EU 662 GFLOPS
- 1.5 MB L3
- 2.0 TB/s L3 bandwidth
- GPU SLM simulation disabled
- Infinite bandwidth of GPU SLM
- 2x DRAM bandwidth
- Integrated GPU

Top offloaded

Location	Speed Up	Data Transfer	Execution Time	Bounded By
[loop in dslash_fn_field_special$omp$pa... at dslash_fn_dblstore.c:704]	6.67x	13.49MB	C... 10.01s GPU 1.50s	LLC_BW
[loop in dslash_fn_field_special$omp$pa... at dslash_fn_dblstore.c:802]	6.88x	13.59MB	CPU 9.18s GPU 1.33s	LLC_BW

Top non offloaded

Location	Data Transfer	Execution Time	Why Not Offloaded
[loop in d_plaquette at d_plaq4.c:31]			Total time is too small for reliable modelling. Use --loop-filter-threshold=0 to model such small offloads.
[loop in d_plaquette at d_plaq4.c:32]			Total time is too small for reliable modelling. Use --loop-filter-threshold=0 to model such small offloads.

Intel MKL – Math Kernel Library

- Highly tuned algorithms
  - FFT
  - Linear algebra (BLAS, LAPACK)
  - Sparse linear algebra
  - Statistical functions
  - Vector math
  - Random number generators
- Optimized for every Intel platform
- oneAPI MKL (oneMKL)
  - https://software.intel.com/en-us/oneapi/mkl

Latest oneAPI toolkits include DPC++ support and C/Fortran OpenMP offload

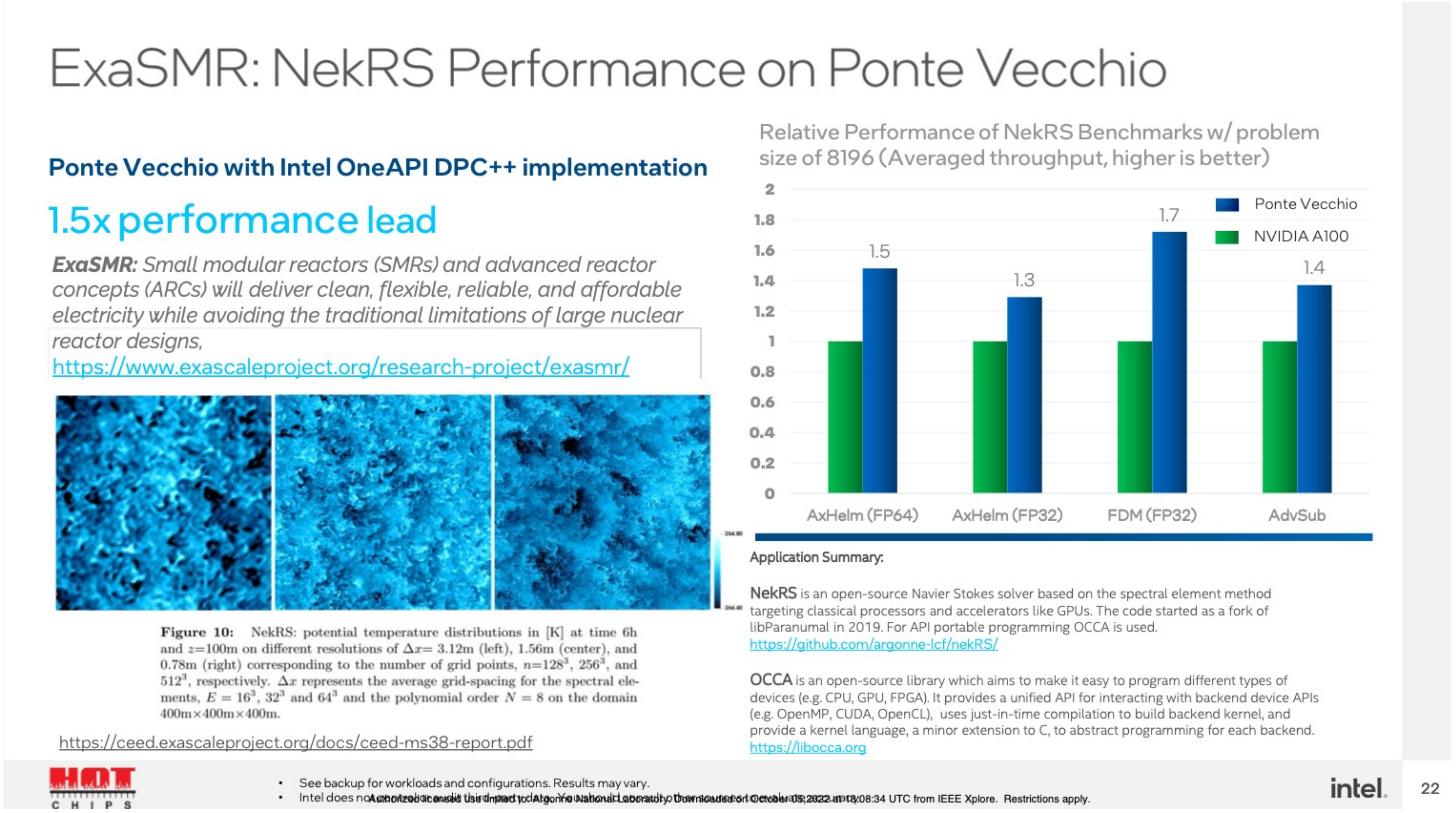
AI and Analytics

- Libraries to support AI and Analytics
  - OneAPI Deep Neural Network Library (oneDNN)
    - High Performance Primitives to accelerate deep learning frameworks
    - Powers Tensorflow, PyTorch, MXNet, Intel Caffe, and more
  - oneAPI Data Analytics Library (oneDAL)
    - Classical Machine Learning Algorithms
    - Easy to use one-line daal4py Python interfaces
    - Powers Scikit-Learn
  - Apache Spark MLlib

Aurora Applications Overview

- ALCF and Intel are working with over 40 projects to ready codes for Aurora:
  - —Argonne Early Science Program (ESP) projects contains a mix of simulations, learning and data projects
  - —DOE Exascale Computing Project (ECP) contains applications (AD) and software (ST) projects
- Over 50 applications and software packages are being prepared for Aurora:
- Involves effort from over 60 Argonne and Intel people and numerous outside teams
- Significant progress on readying applications for Aurora has occurred
  - —ECP and ESP teams have been actively porting and testing code and reporting issues
  - —Argonne and Intel have held quarterly application status reviews to identify top issues
  - —Monthly priority bug meeting between ANL and Intel to follow-up and track issue resolution
  - —Receiving regular SDK updates from Intel
  - —Test framework on JLSE allows issue reproducers and applications tests to be run before software updates and nightly to identify changes

Showcase



ExaSMR: NekRS Performance on Ponte Vecchio

Ponte Vecchio with Intel OneAPI DPC++ implementation

1.5x performance lead

ExaSMR: *Small modular reactors (SMRs) and advanced reactor concepts (ARCs) will deliver clean, flexible, reliable, and affordable electricity while avoiding the traditional limitations of large nuclear reactor designs,*
https://www.exascaleproject.org/research-project/exasmr/

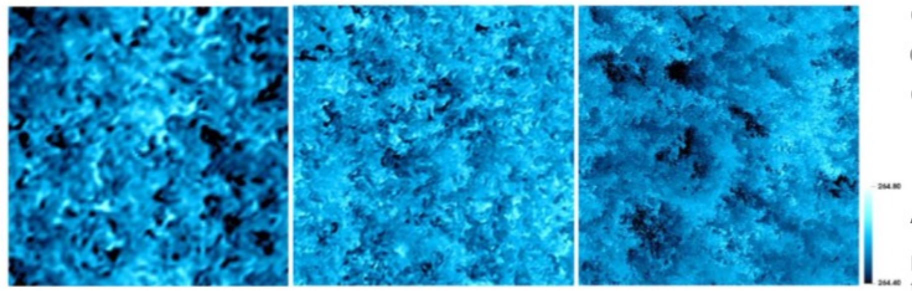
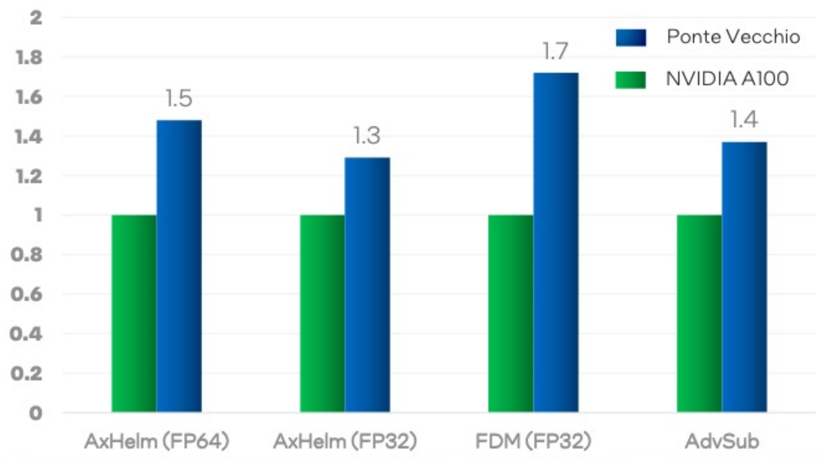


Figure 10: NekRS: potential temperature distributions in [K] at time 6h and z=100m on different resolutions of Δx= 3.12m (left), 1.56m (center), and 0.78m (right) corresponding to the number of grid points, n=128^3, 256^3, and 512^3, respectively. Δx represents the average grid-spacing for the spectral elements, $E = 16^3$, 32^3 and 64^3 and the polynomial order $N = 8$ on the domain 400m×400m×400m.

https://ceed.exascaleproject.org/docs/ceed-ms38-report.pdf

Relative Performance of NekRS Benchmarks w/ problem size of 8196 (Averaged throughput, higher is better)



Application Summary:

NekRS is an open-source Navier Stokes solver based on the spectral element method targeting classical processors and accelerators like GPUs. The code started as a fork of libParanumal in 2019. For API portable programming OCCA is used.
https://github.com/argonne-lcf/nekRS/

OCCA is an open-source library which aims to make it easy to program different types of devices (e.g. CPU, GPU, FPGA). It provides a unified API for interacting with backend device APIs (e.g. OpenMP, CUDA, OpenCL), uses just-in-time compilation to build backend kernel, and provide a kernel language, a minor extension to C, to abstract programming for each backend.
https://libocca.org

HOT CHIPS

- See backup for workloads and configurations. Results may vary.
- Intel does no[illegible]

intel. 22

H. Jiang, "Intel's Ponte Vecchio GPU : Architecture, Systems & Software," *2022 IEEE Hot Chips 34 Symposium (HCS)*, 2022, pp. 1-29, doi: 10.1109/HCS55958.2022.9895631.

Argonne NATIONAL LABORATORY

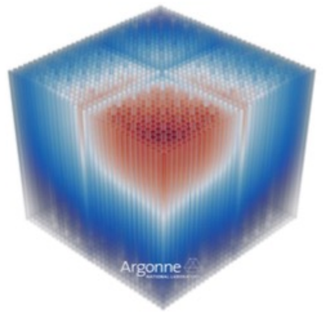
Showcase

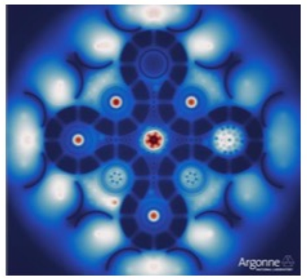
ExaSMR: OpenMC Performance on Ponte Vecchio

Monte Carlo particle transport code for exascale computations

Ponte Vecchio with OpenMP Target offload

2x performance lead

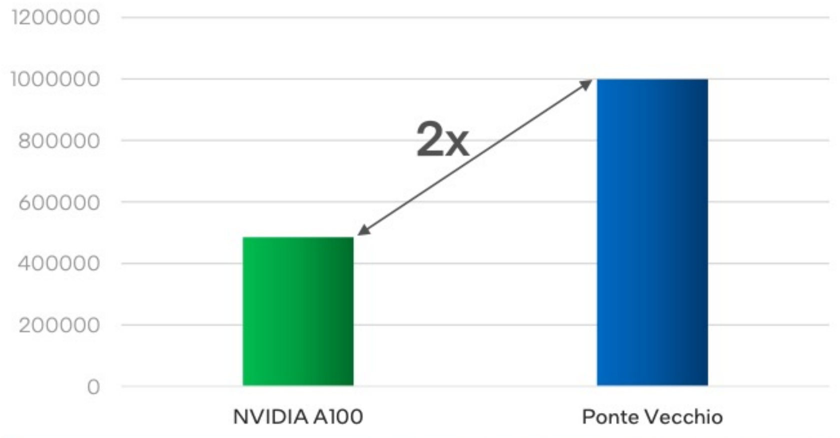




Exascale Compute Project Annual Meeting 2022 presentation:
https://www.alcf.anl.gov/events/2022-ecp-annual-meeting
International Conference on Physics of Reactors 2022 presentation:
https://www.ans.org/meetings/physor2022/session/view-976/

https://docs.openmc.org

OpenMC Depleted Fuel Inactive Batch Performance on HM-Large Reactor with 40M particles (particles/second, Higher is better)



Application Summary: OpenMC is a Monte Carlo particle transport application that has recently been ported to the OpenMP target offloading programming model for use on GPU-based systems. The Monte Carlo method employed by OpenMC is considered the "gold standard" for high-fidelity simulation while also having the advantage of being a general-purpose method able to simulate nearly any geometry or material without the need for domain-specific assumptions. However, despite the extreme advantages in ease of use and accuracy, Monte Carlo methods like those in OpenMC often suffer from a very high computational cost. The extreme performance gains OpenMC has achieved on GPUs, as compared to traditional CPU architectures, is finally bringing within reach a much larger class of problems that historically were deemed too expensive to simulate using Monte Carlo methods. The leap in performance that GPUs are now offering carries with it the potential to disrupt a number of engineering technology stacks that have traditionally been dominated by non-general deterministic methods. For instance, faster MC applications may greatly expand the design space and simplify the regulation process for new nuclear reactor designs -- potentially improving the economics of nuclear energy and therefore helping to solve the world's climate crisis.

- See backup for workloads and configurations. Results may vary.
- Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

H. Jiang, "Intel's Ponte Vecchio GPU : Architecture, Systems & Software," *2022 IEEE Hot Chips 34 Symposium (HCS)*, 2022, pp. 1-29, doi: 10.1109/HCS55958.2022.9895631.

Thank You

Argonne NATIONAL LABORATORY