Upcoming ALCF Systems

2022 ALCF Computational Performance Workshop
May 24, 2022
Scott Parker

Polaris

Polaris will provide a platform utilizing several of the Aurora technologies and similar architectures to provide ALCF staff and users a platform for early scaling and testing purposes.

PEAK PERFORMANCE

44 Petaflop DP

NVIDIA GPU

A100

AMD EPYC PROCESSOR

Rome*

PLATFORM

HPE Apollo Gen10+

Compute Node
1 AMD EPYC 7532* processor; 4 NVIDIA A100 GPUs; Unified Memory Architecture; 2 fabric endpoints; 2 NVMe SSDs

GPU Architecture
NVIDIA A100 GPU; HBM stack

Processor Interconnects
CPU-GPU: PCIe
GPU-GPU: NVLink

System Interconnect
HPE Slingshot 10*; Dragonfly topology with adaptive routing

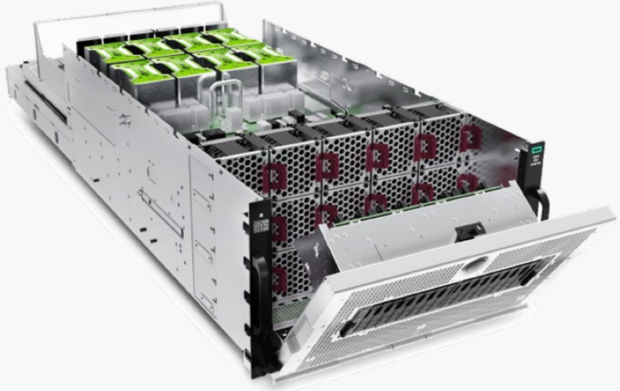
Network Switch
25.6 Tb/s per switch, from 64–200 Gb/s ports (25 GB/s per direction)

Programming Models
CUDA, MPI, OpenMP, C/C++, Fortran, DPC++

Node Performance
78 TF

Aggregate Memory
368 TB (88 BG GPU, 280 CPU)

System Size
40 racks
560 CPUs
2240 GPUs
560 nodes, 1.78 MW



*Initial technology to be upgraded later

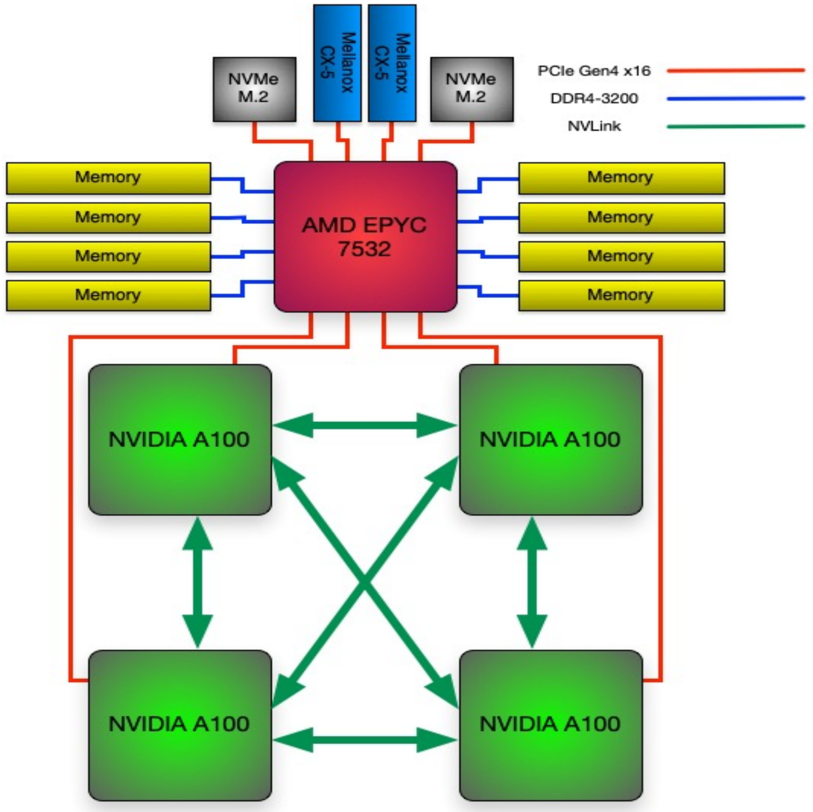
Polaris

- ALCF’s latest computational resource
  - #12 on the Top 500, 24 PF
- https://www.alcf.anl.gov/polaris

- Available in the coming months
  - Currently limited to ALCF staff as part of standup
  - Targeting 2H2022 for wider general access

Polaris Single Node Configuration

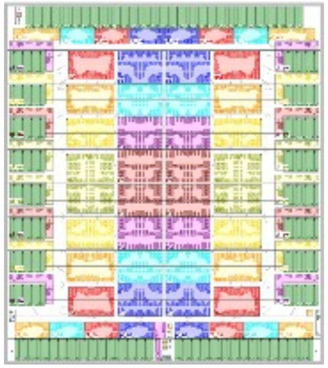
# of AMD EPYC 7532 CPUs	1
# of NVIDIA A100 GPUs	4
Total HBM2 Memory	160 GB
HBM2 Memory BW per GPU	1.6 TB/s
Total DDR4 Memory	512 GB
DDR4 Memory BW	204.8 GB/s
# OF NVMe SSDs	2
Total NVMe SSD Capacity	3.2 TB
# of Cassini NICs	2
Total Injection BW (w/ Cassini)	50 GB/s
PCIe Gen4 BW	64 GB/s
NVLink BW	600 GB/s
Total GPU DP Tensor Core Flops	78 TF



Slingshot Interconnect

Rosetta Switch

- Multiple QoS levels
- Aggressive adaptive routing
- Advanced congestion control
- Very low average and tail latency
- High performance multicast and reduction



64 ports x 200 Gbps

SS-10 (100Gb)
Injection: ~14 TB/s
Bisection: ~24 TB/s

SS-11 (200Gb)
Injection: ~28 TB/s
Bisection: ~24 TB/s

Mellanox ConnectX NIC

Slingshot 10

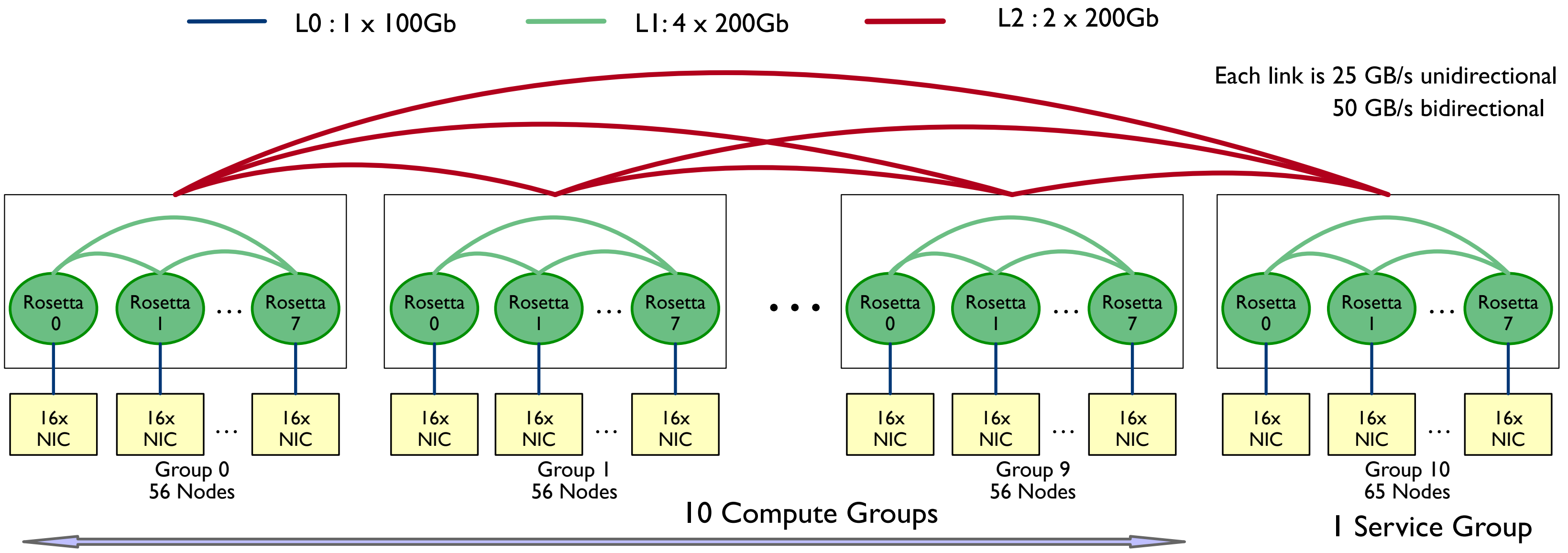
- HPE Cray MPI stack
- Ethernet functionality
- RDMA offload

Cassini NIC

Slingshot 11

- MPI hardware tag matching
- MPI progress engine
- One-sided operations
- Collectives
- 2X injection bandwidth

Slingshot Configuration



L0 : 1 x 100Gb

L1: 4 x 200Gb

L2 : 2 x 200Gb

Each link is 25 GB/s unidirectional
50 GB/s bidirectional

Rosetta 0 · Rosetta 1 … Rosetta 7

16x NIC · 16x NIC … 16x NIC

Group 0
56 Nodes

Group 1
56 Nodes

Group 9
56 Nodes

Group 10
65 Nodes

10 Compute Groups

1 Service Group

- 11 Total dragonfly groups, 10 compute groups and 1 non-compute group
- 2 links/arc between each group
- 4 links/arc within each group (between switches of a group)
- 1 link from each NIC (100Gb with SS10, 200Gb when upgraded to SS11)

Polaris Programming Environment

- HPE Cray PE
- NVIDIA HPC SDK
- Programming models supported:
  - OpenMP
  - SYCL/Data Parallel C++ provided via
    - CodePlay computecpp compiler with Nvidia support
    - LLVM via Intel DPC++ branch which supports offload to Nvidia GPUs as well as Intel GPUs
  - Kokkos
  - RAJA
  - HIP
  - CUDA
  - OpenACC

Polaris as a Bridge to Aurora

Component	Polaris	Aurora
System Software	HPCM	HPCM
Programming Models	OpenMP, DPC++, Kokkos, RAJA, HIP, CUDA, OpenACC	OpenMP, DPC++, Kokkos, RAJA, HIP
Tools	PAT, gdb, ATP, NVIDIA Nsight, cuda-gdb	PAT, gdb, ATP, Intel VTune
MPI	HPE Cray MPI, MPICH	HPE Cray MPI, MPICH, Intel MPI
Multi-GPU	*1 CPU : 4 GPU*	*2 CPU : 6 GPU*
High-Speed Network (HSN)	HPE Slingshot	HPE Slingshot
Data and Learning	DL frameworks, Cray AI stack, Python, Numba, Spark, Containers, RAPIDS	DL frameworks, Cray AI stack, Python, Numba, Spark, Containers, oneDAL
Math Libraries	cu* from CUDA	oneAPI

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

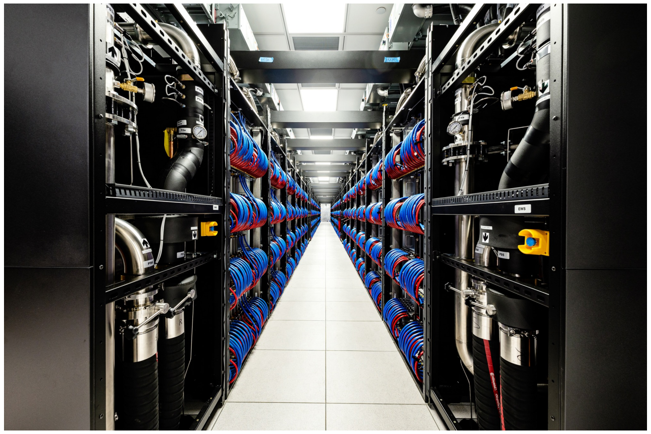
System Interconnect
HPE Slingshot 11
Dragonfly topology with adaptive routing

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

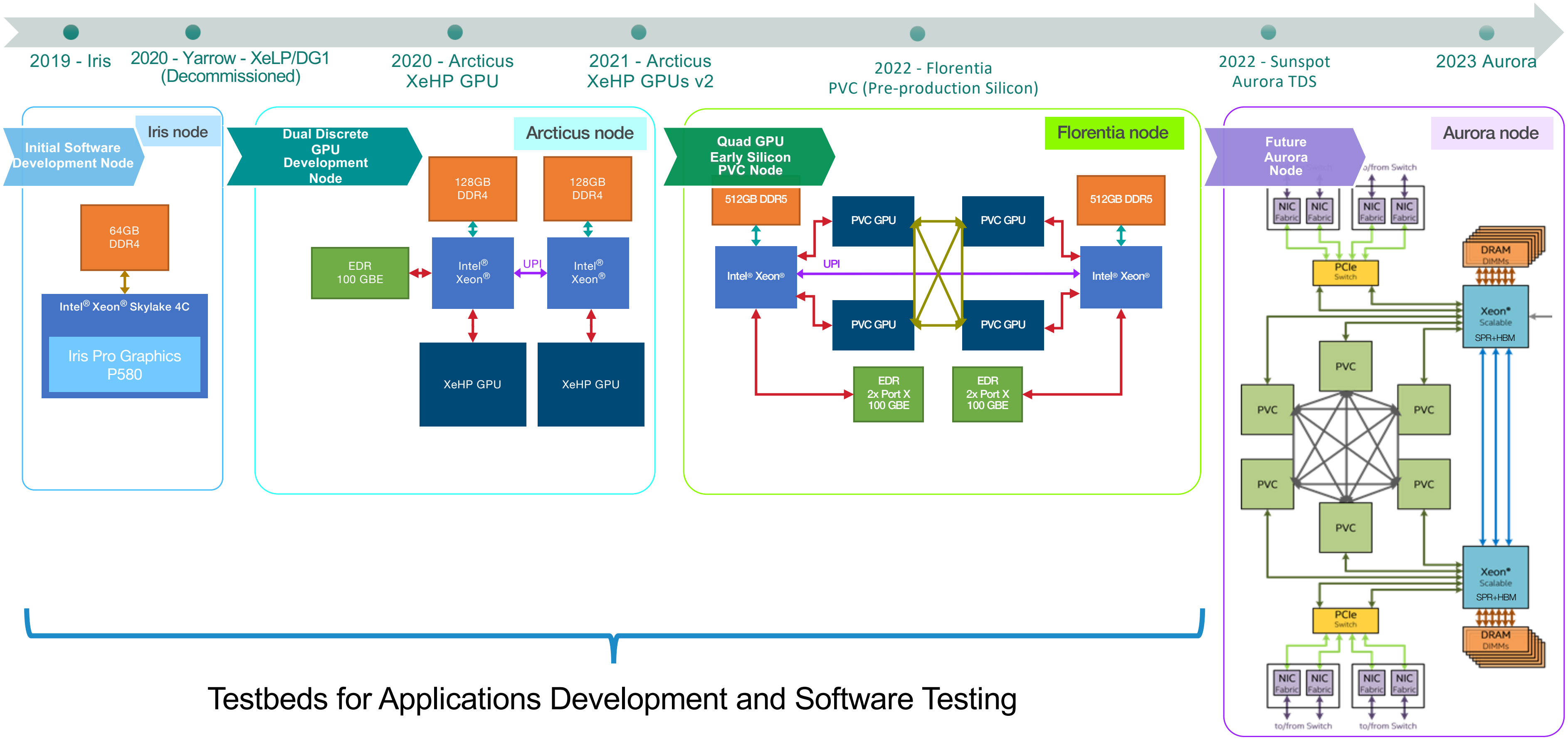
High-Performance Storage
220 PB
≧25 TB/s DAOS bandwidth

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 (Pre-production Silicon)

2022 - Sunspot Aurora TDS

2023 Aurora

Iris node

Initial Software Development Node

Arcticus node

Dual Discrete GPU Development Node

Florentia node

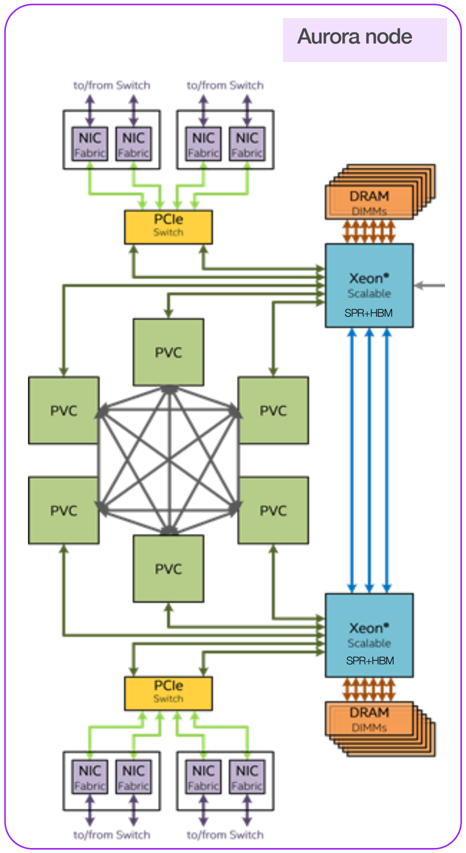
Quad GPU Early Silicon PVC Node

Aurora node

Future Aurora Node

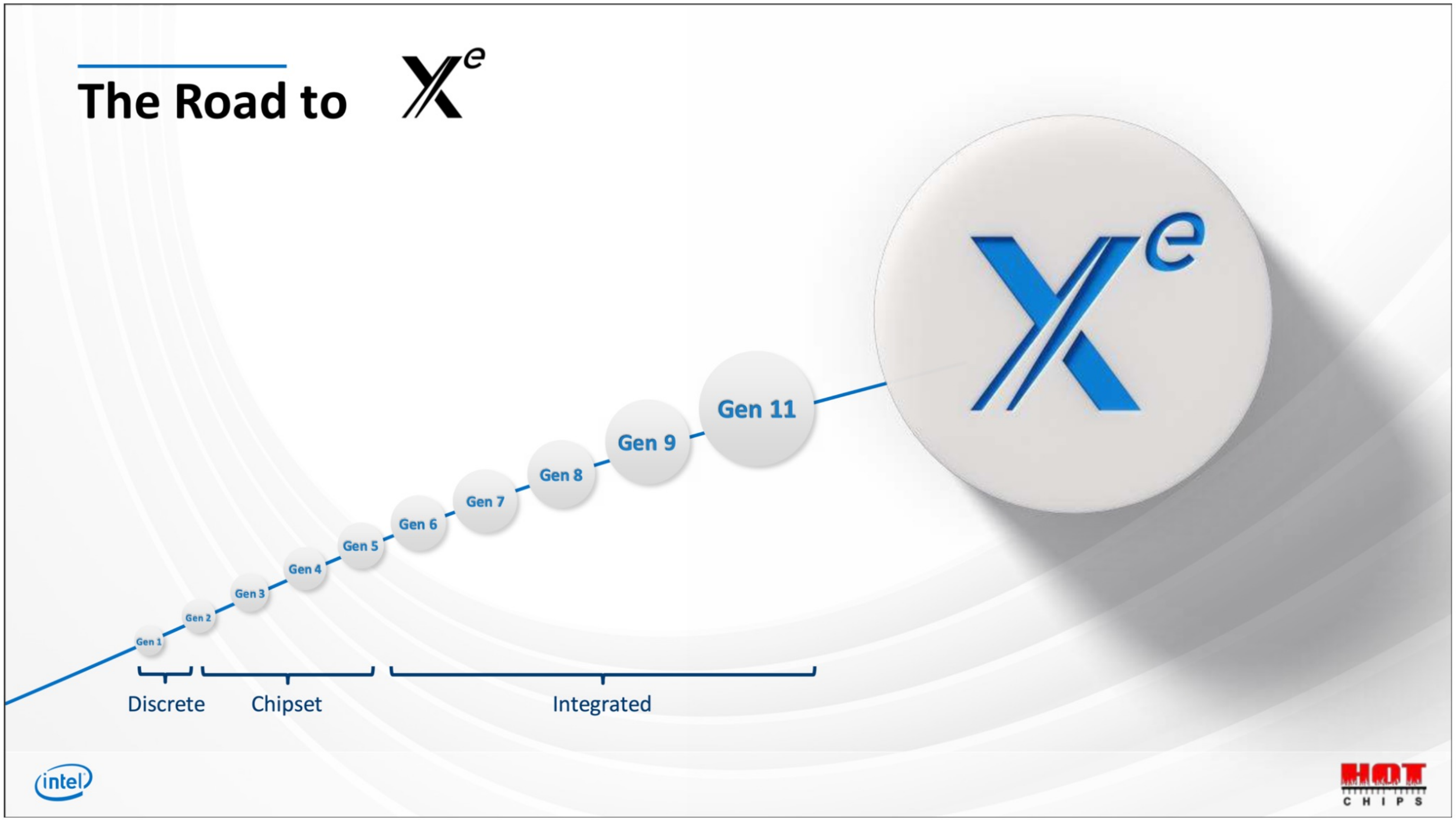
Testbeds for Applications Development and Software Testing

Aurora Compute Node



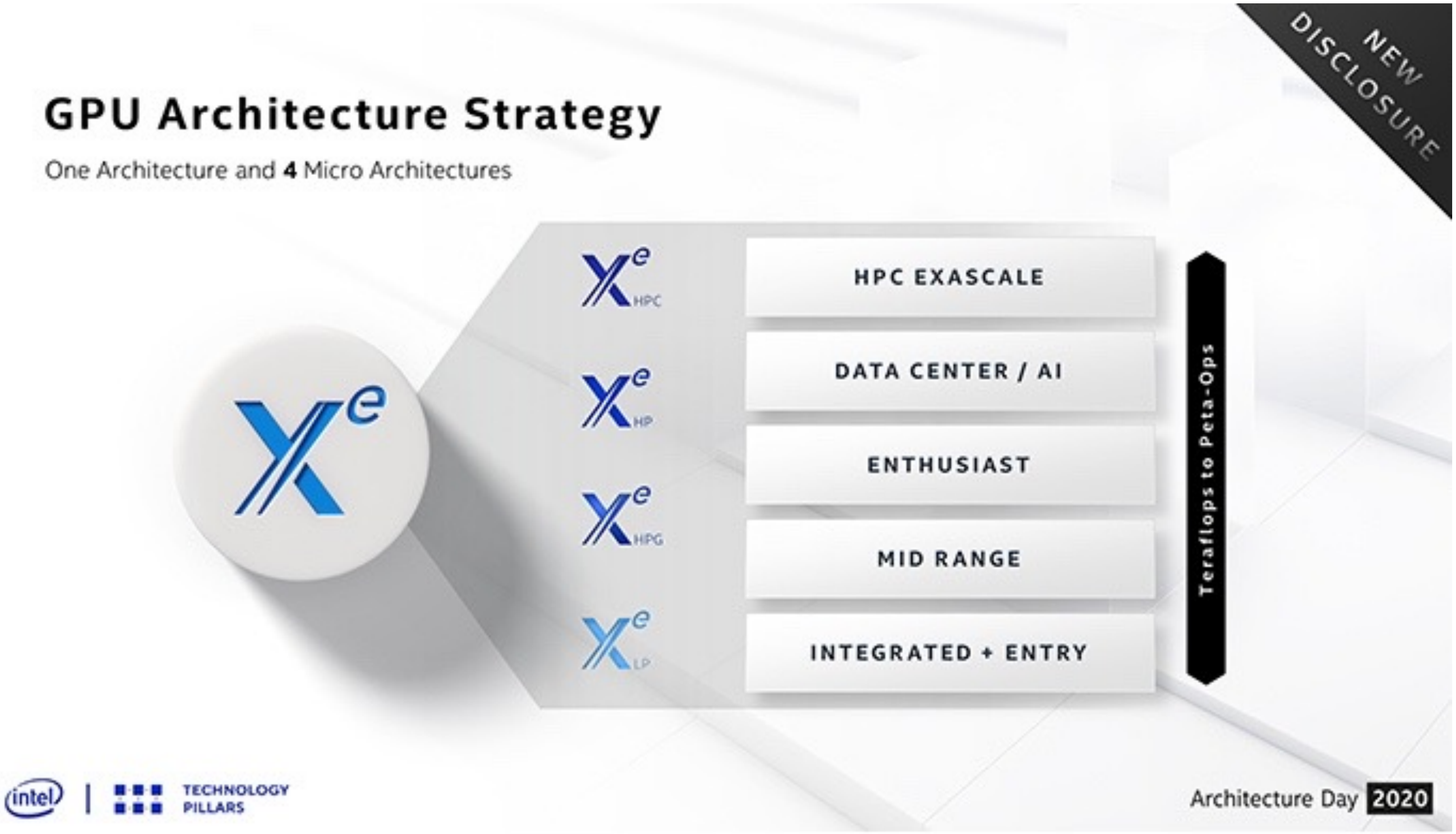
- 6 X^e Architecture based GPUs (Ponte Vecchio)
  - All to all connection
- 2 Intel Xeon (Sapphire Rapids) processors
- Unified Memory Architecture across CPUs and GPUs
- 8 Slingshot Fabric endpoints

The Evolution of Intel GPUs



The Road to Xe

The Evolution of Intel GPUs



NEW DISCLOSURE

GPU Architecture Strategy

One Architecture and **4** Micro Architectures

X^e

X^e_{HPC} HPC EXASCALE

X^e_{HP} DATA CENTER / AI

X^e_{HPG} ENTHUSIAST

MID RANGE

X^e_{LP} INTEGRATED + ENTRY

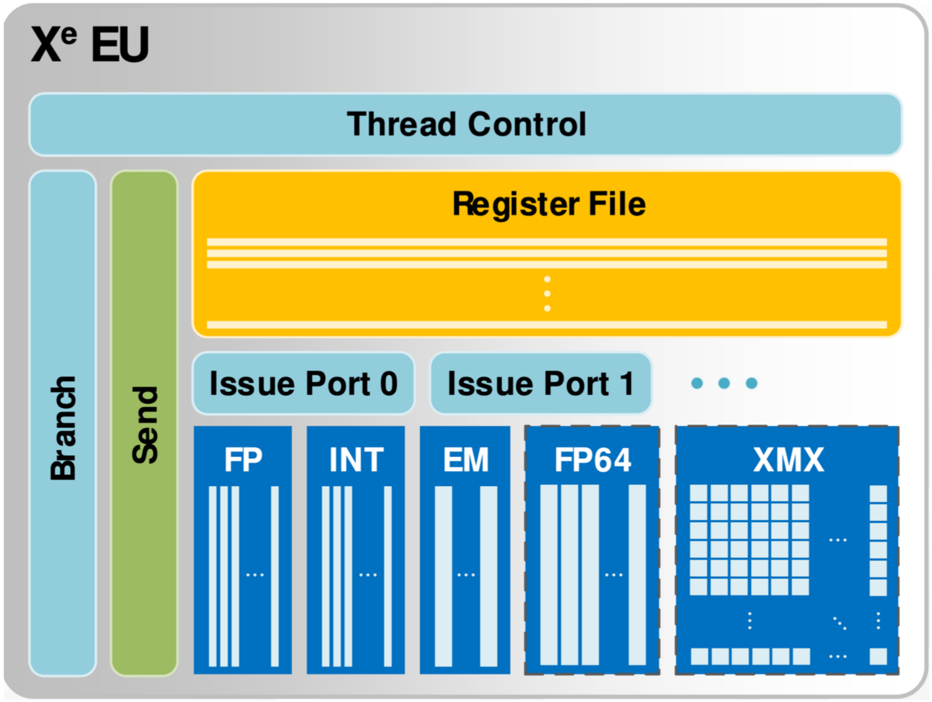
Teraflops to Peta-Ops

intel | TECHNOLOGY PILLARS

Architecture Day 2020

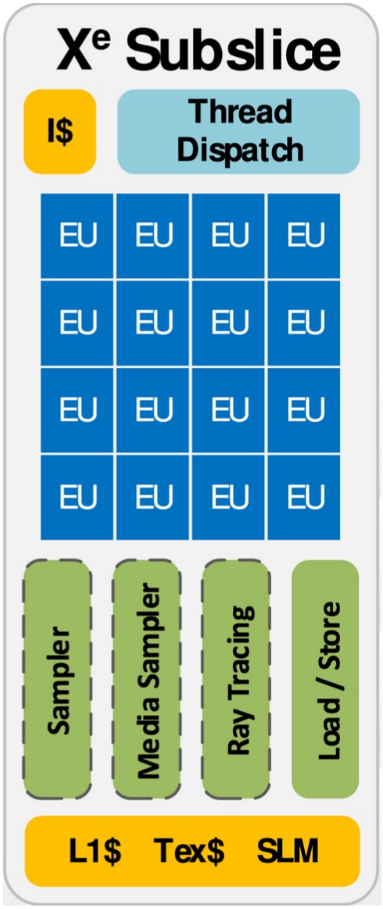
XE Execution Unit

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



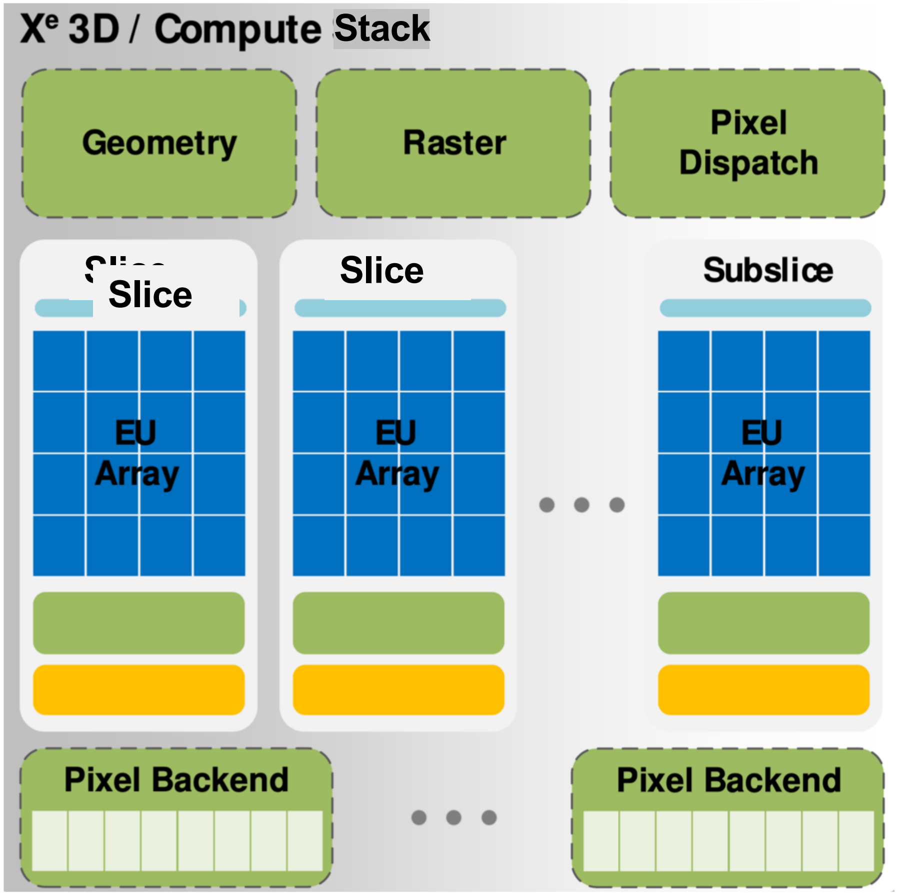
XE Slice

- ❑ A Slice contains:
  - ❑ 16 EUs
  - ❑ Thread dispatch
  - ❑ Instruction cache
  - ❑ L1, texture cache, and shared local memory
  - ❑ Load/Store
  - ❑ Fixed Function (optional)
    - ❑ 3D Sampler
    - ❑ Media Sampler
    - ❑ Ray Tracing



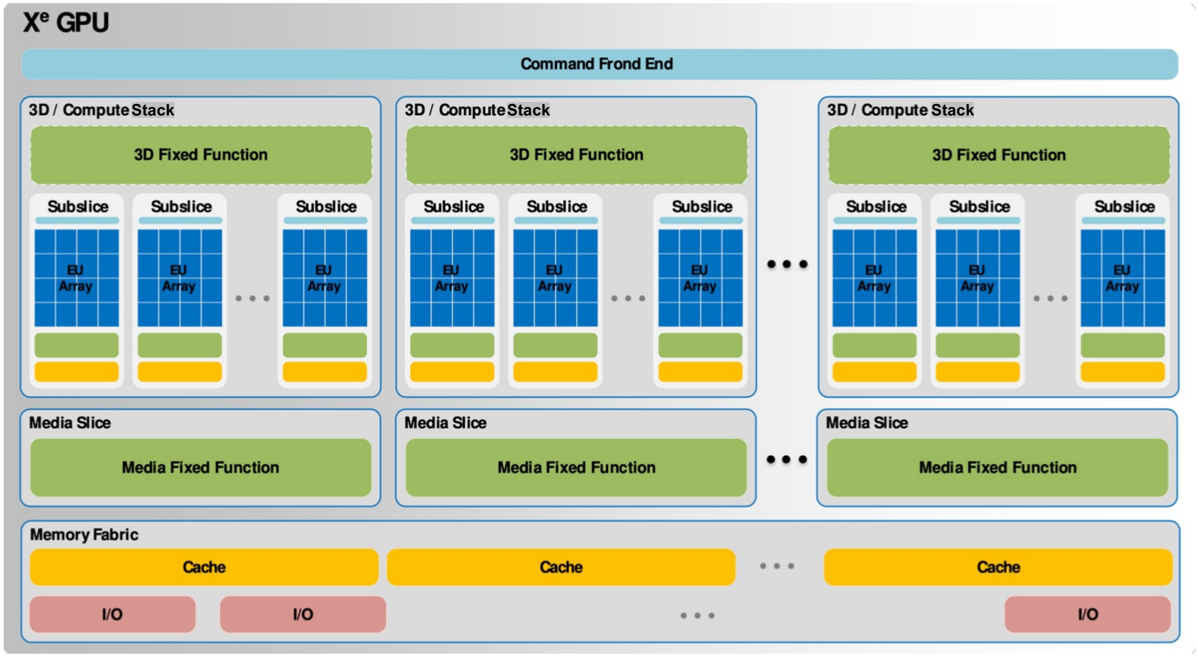
XE Stack

- ❑ A Stack contains
  - ❑ Variable number of slices
  - ❑ 3D Fixed Function (optional)
    - ❑ Geometry
    - ❑ Raster



High Level Xe Architecture

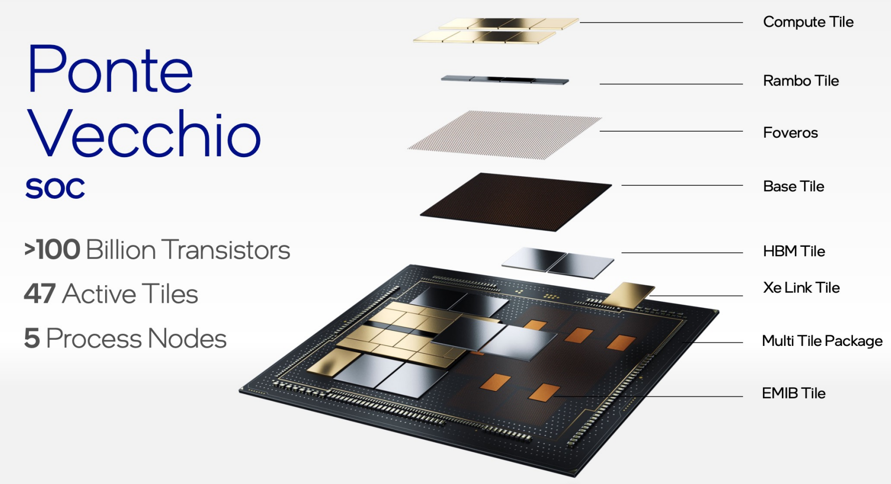
- ❑ X^e GPU is composed of
  - ❑ 3D/Compute Stacks
  - ❑ Media Stack
  - ❑ Memory Fabric / Cache



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



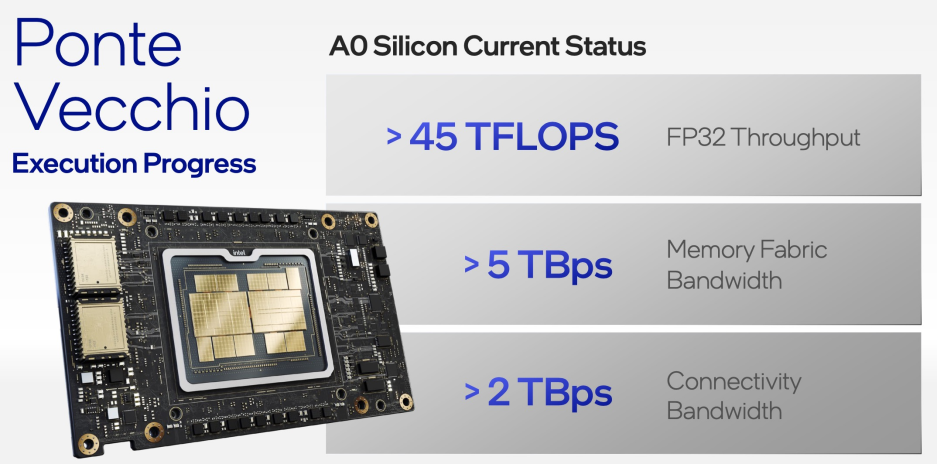
Ponte Vecchio

SOC

>100 Billion Transistors

47 Active Tiles

5 Process Nodes



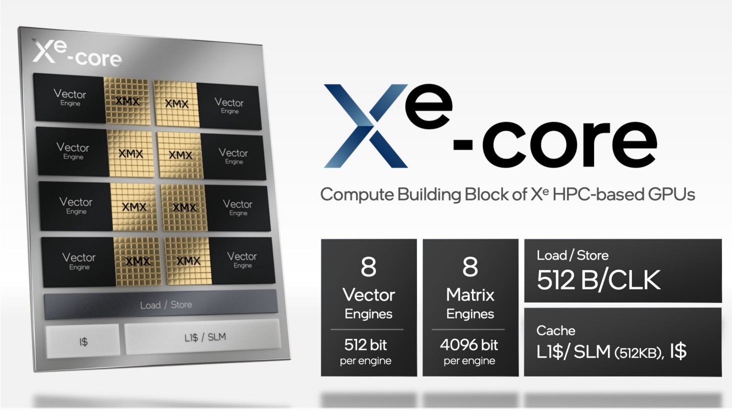
Ponte Vecchio

Execution Progress

A0 Silicon Current Status

> 45 TFLOPS	FP32 Throughput
> 5 TBps	Memory Fabric Bandwidth
> 2 TBps	Connectivity Bandwidth

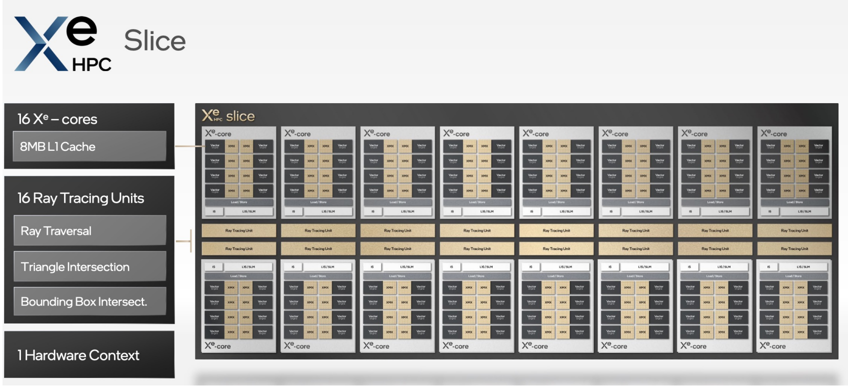
Intel Ponte Vecchio Architectural Components



Xe-core

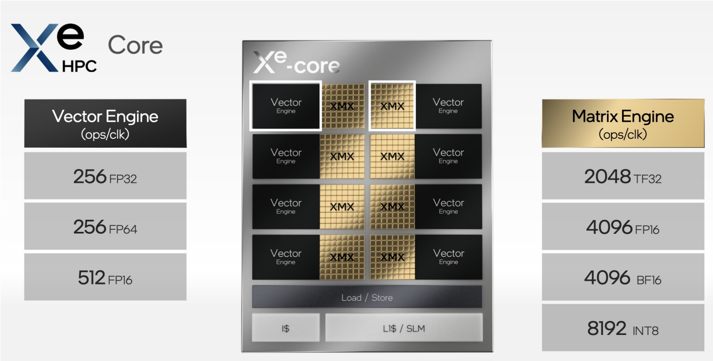
Compute Building Block of Xe HPC-based GPUs

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



Xe HPC Slice

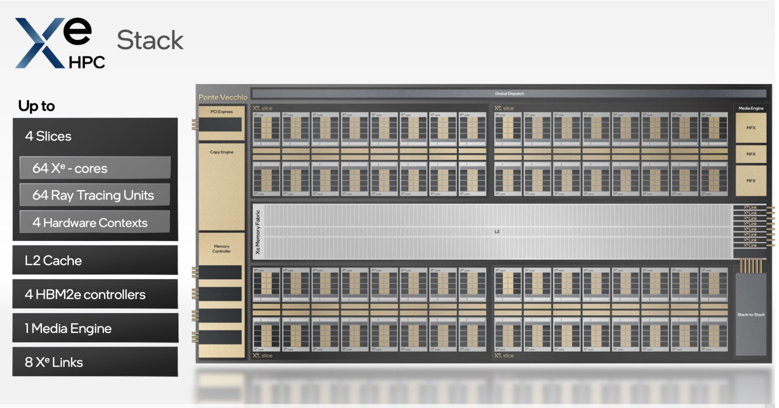
- 16 Xe – cores
  - 8MB L1 Cache
- 16 Ray Tracing Units
  - Ray Traversal
  - Triangle Intersection
  - Bounding Box Intersect.
- 1 Hardware Context



Xe HPC Core

Vector Engine (ops/clk)
256 FP32
256 FP64
512 FP16

Matrix Engine (ops/clk)
2048 TF32
4096 FP16
4096 BF16
8192 INT8



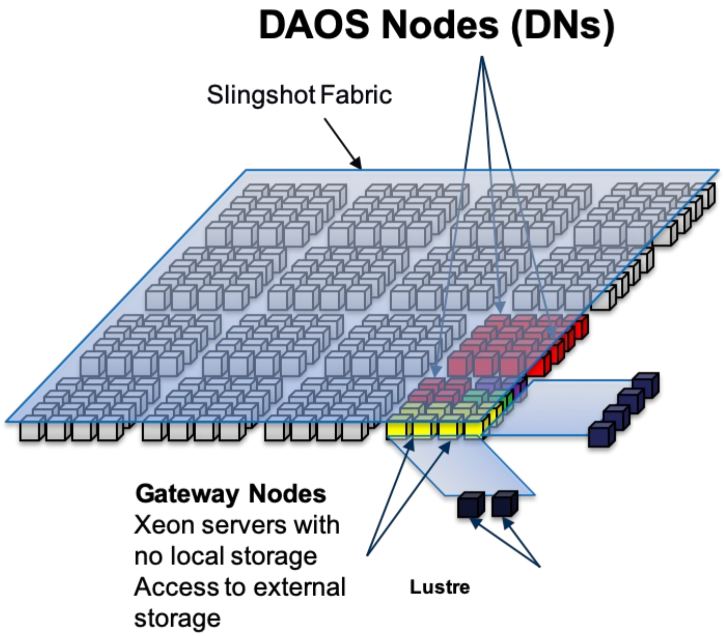
Xe HPC Stack

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

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



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

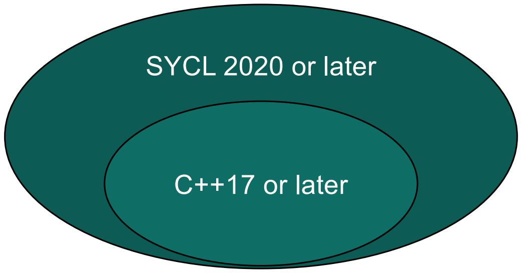
- Aurora applications may use:
  - DPC++/SYCL
  - OpenMP
  - Kokkos
  - Raja
  - OpenCL

- Experimental
  - HIP

- Not available on Aurora:
  - CUDA
  - OpenACC

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

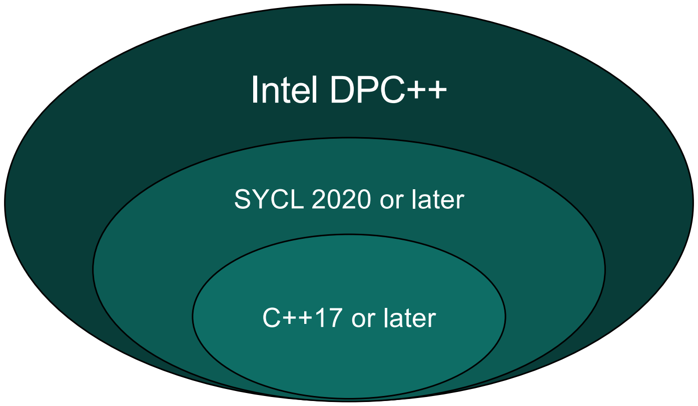
- ❑ SYCL
  - ❑ Standard developed by Khronos and announced in 2014
  - ❑ The latest SYCL specification (SYCL 2020) was release in 2021
  - ❑ SYCL is a C++ based abstraction layer (standard C++17)
  - ❑ Builds on OpenCL **concepts** (but single-source)
  - ❑ *SYCL is designed to be as close to standard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

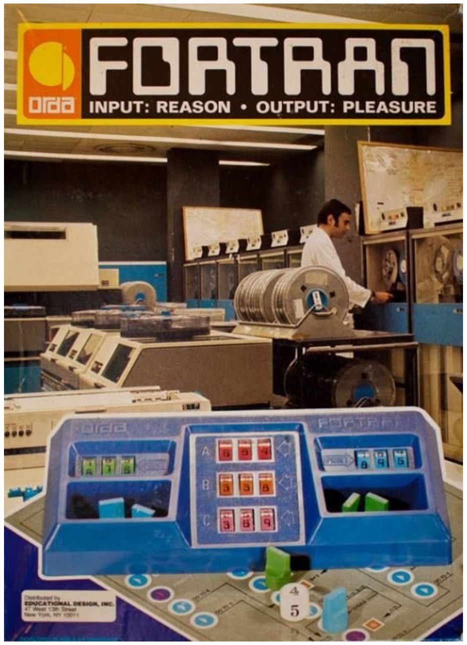


OpenMP

- 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

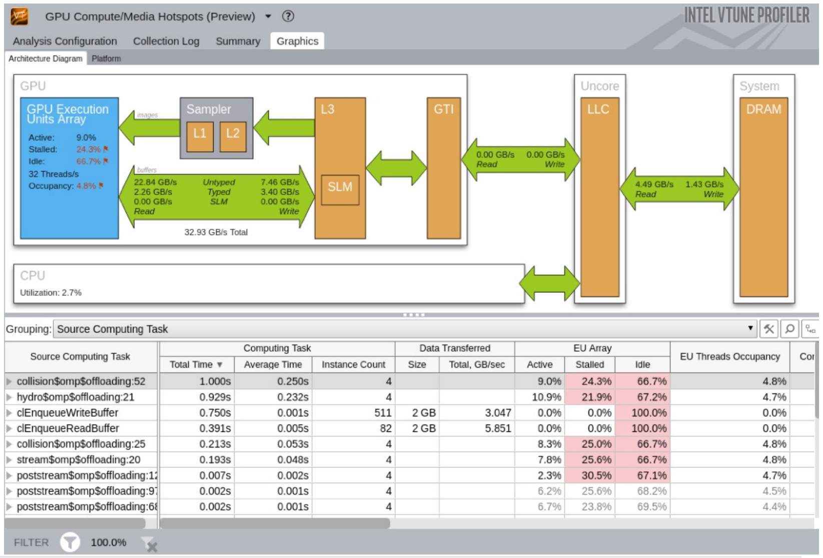


https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/fortran-compiler.html

Intel VTune and Advisor

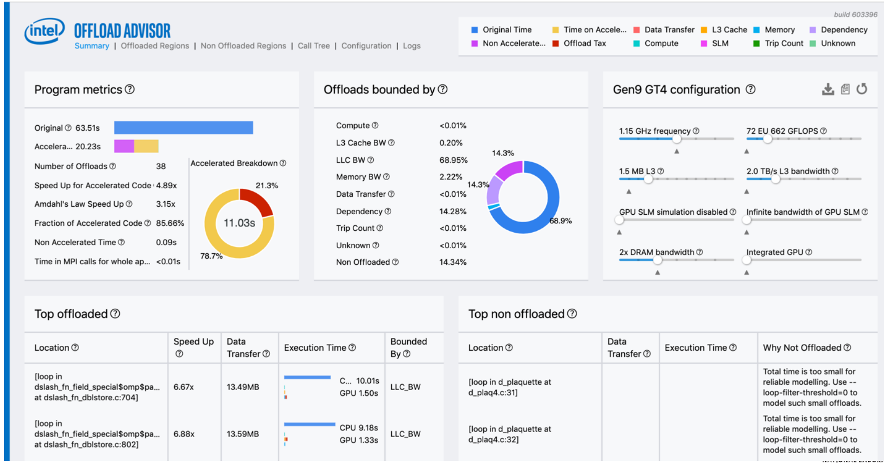
- ❑ Vtune Profiler
  - ❑ Widely used performance analysis tool
  - ❑ Supports analysis on Intel GPUs

- ❑ 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:12	0.007s	0.002s	4			2.3%	30.5%	67.1%	4.7%	
poststream$omp$offloading:97	0.002s	0.001s	4			6.2%	25.6%	68.2%	4.5%	
poststream$omp$offloading:68	0.002s	0.001s	4			6.7%	23.8%	69.5%	4.4%	



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

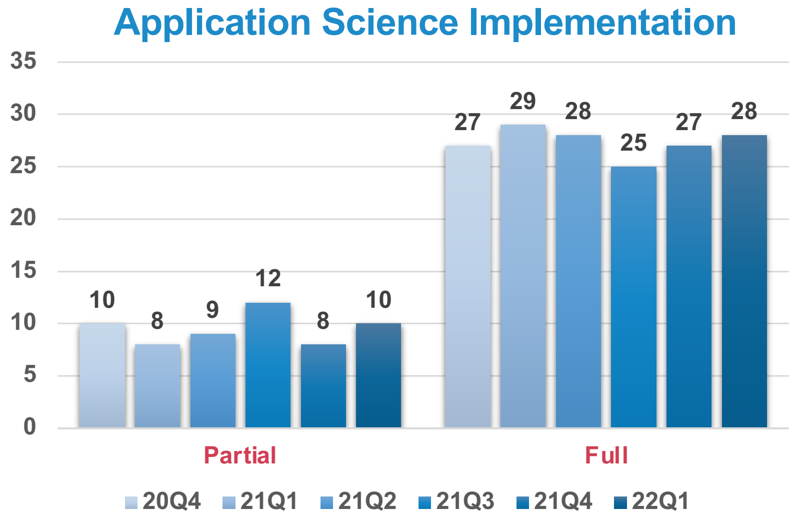
- ❑ Libraries to support AI and Analytics
  - ❑ OneAPI Deep Neural Network Library (oneDNN)
    - ❑ High Performance Primitives to accelerate deep learning frameworks
    - ❑ Powers Tensorflow, PyTorch, MXNet, Intel Caffe, and more
  - ❑ oneAPI Data Analytics Library (oneDAL)
    - ❑ Classical Machine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

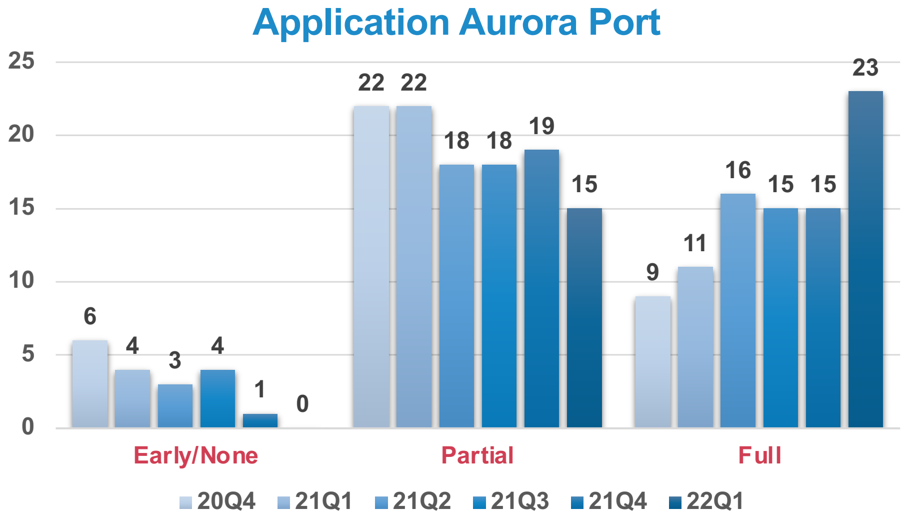
Aurora Applications Overview

- 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

Aurora Applications Development

- **Steps in application preparation**
  - Implementation of science and algorithms
  - Porting to Aurora programming models
  - Testing with Aurora SDK on Aurora testbeds
  - Tuning for performance on Aurora testbeds
  - Scaling across the Aurora system





Arcticus Applications Testing and Tuning Status

Application	Status
XGC*+	Ready
NWChemEx*+	Ready
SW4+	Ready
HACC*+	Ready
NAMD*	Improving Performance
PHASTA*	Improving Performance
GAMESS+	Improving Performance
Grid*+	Improving Performance
FusionDL*	Improving Performance
AMRWind+	Improving Performance
NekRS+	Improving Performance
Madgraph*	Improving Performance
CANDLE/UNO*+	Improving Performance
QMCPack*+	Improving Performance
QUDA*+	Improving Performance
FastCaloSim*	Running
NYX+	Running
DarkSkyMining*	Running
DCMesh*	Running

Applications	Status
FloodFillNetwork*	Running
Chroma*+	Running
BerkelyGW*	Components Running
E3SM-MMF+	Components Running
MFIX-Exa+	Components Running
spiniFEL+	Components Running
OpenMC+	Components Running
LAMMPS+	Components Running
GENE+	Components Running
Uintah*	Components Running
Thornado+	Components Running
Data Driven CFD*	Components Running
E3SM (YAKL) +	Components Running
cctbx+	Components Running
Flow Based Generative Model*	Gated
Nalu-Wind+	Gated
GEM+	Not Tested
MatML Workflow*	Not Tested
Multi-grid Parameter Optimization*	Not Tested

* ESP Code
+ ECP Code

Description
Ready for next testbed
Working to improve performance
Full application running
Components running
Waiting on needed functionality
Not tested yet on system

Thank You