



New player incoming Intel Ponte Vecchio GPU architecture in server space

Michal Mrozek – Principal Engineer Intel

PPAM 2022

Agenda

Ponte Vecchio Architecture

oneAPI Software Stack

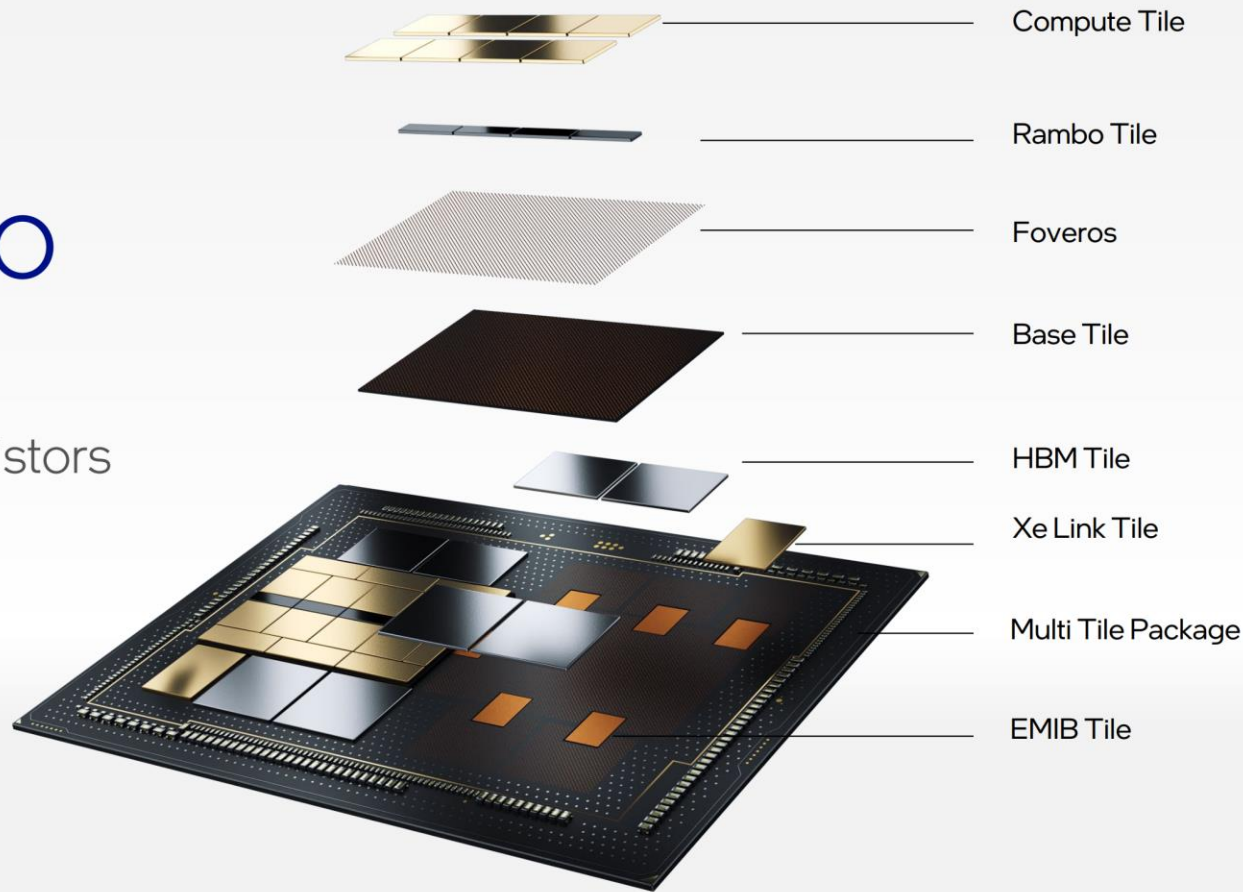
Application Performance

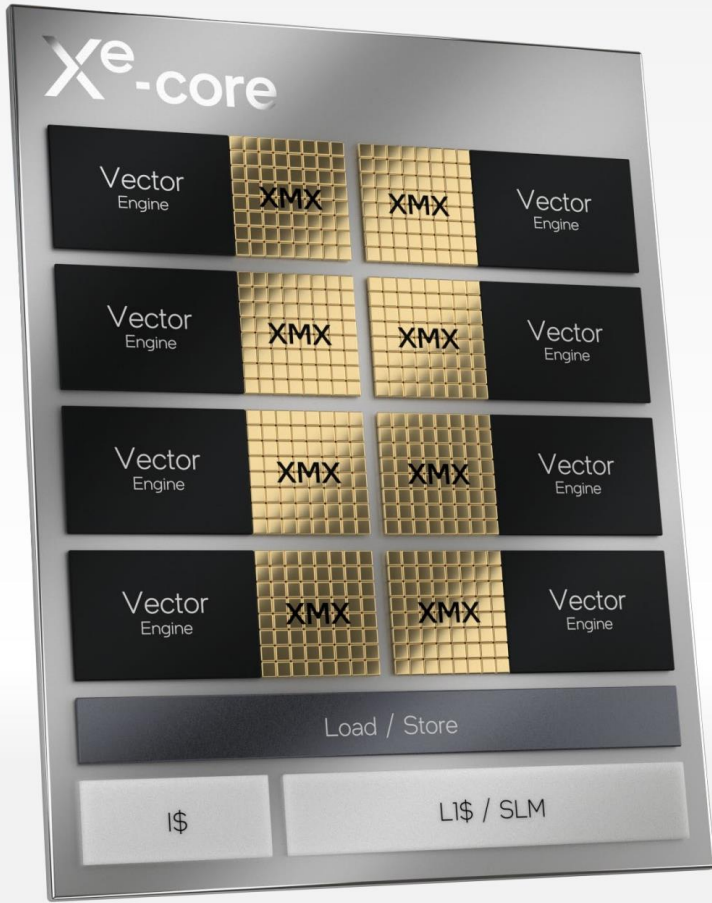
Ponte Vecchio soc

>100 Billion Transistors

47 Active Tiles

5 Process Nodes





Xe-core

Compute Building Block of Xe HPC-based GPUs

8
Vector
Engines

512 bit
per engine

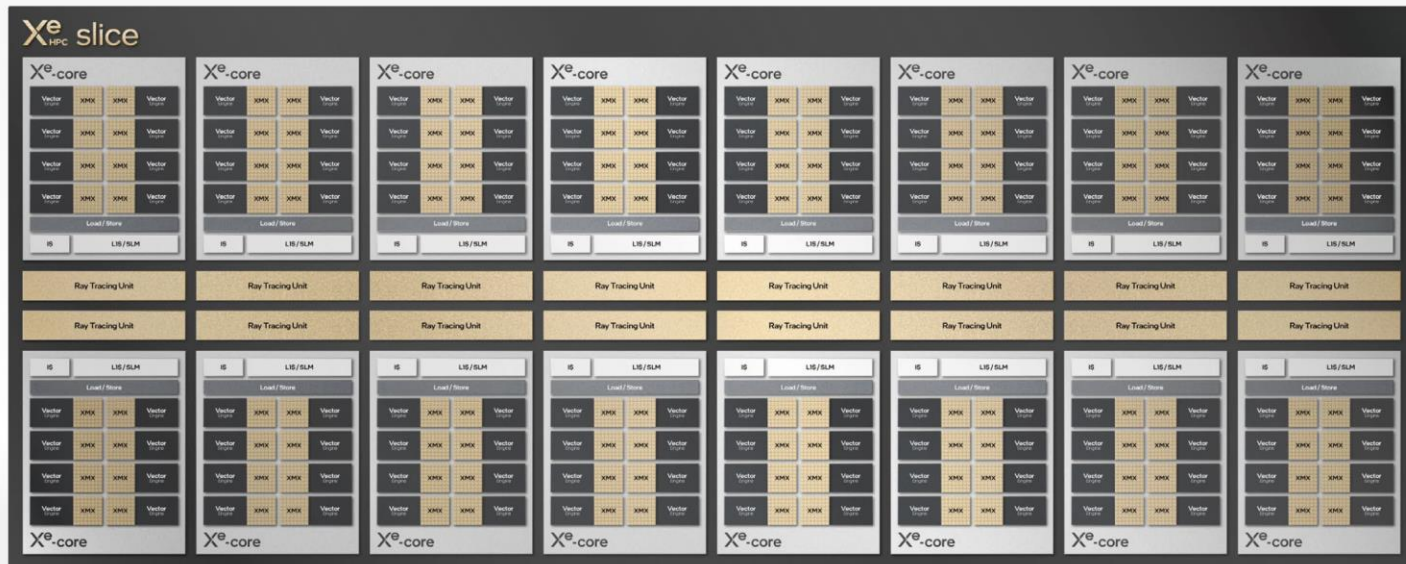
8
Matrix
Engines

4096 bit
per engine

Load / Store
512 B/CLK

Cache
L1\$ / SLM (512KB), I\$

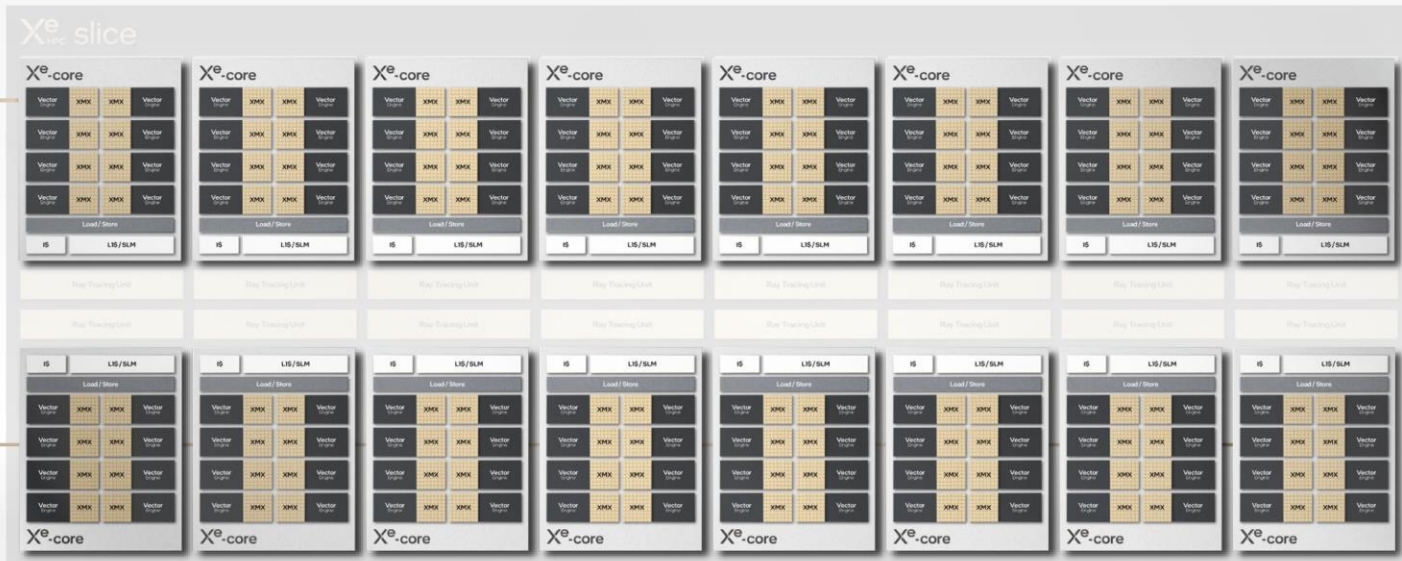
Xe HPC Slice



Xe HPC Slice

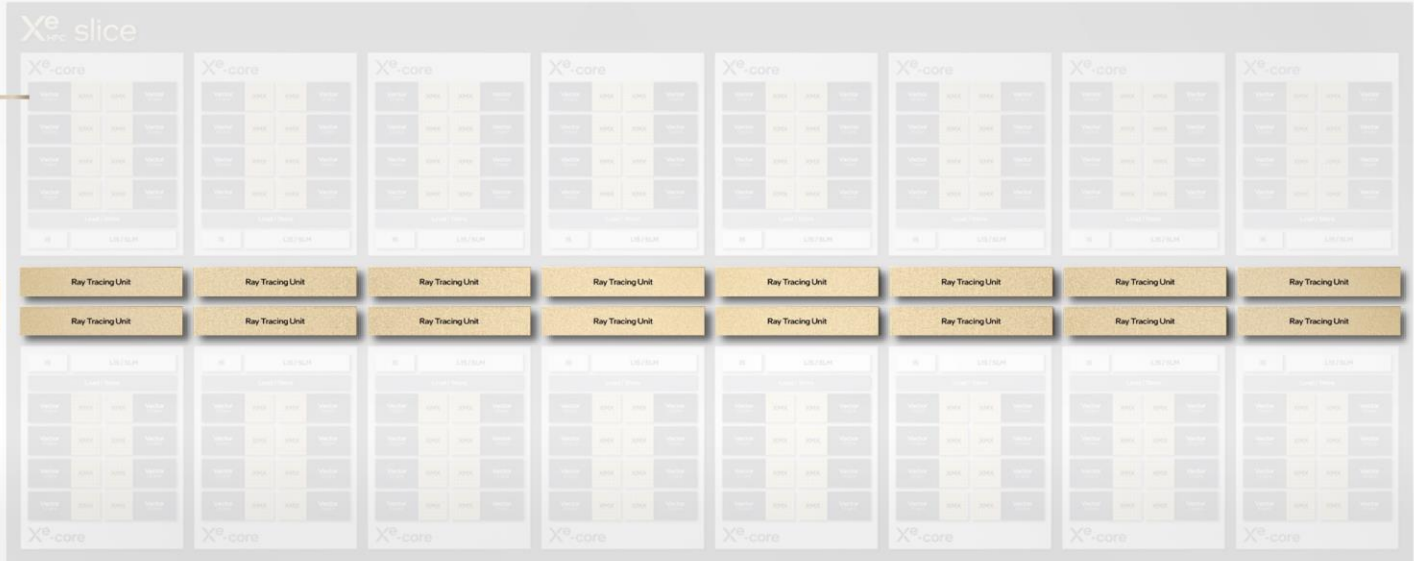
16 Xe – cores

8MB LI Cache



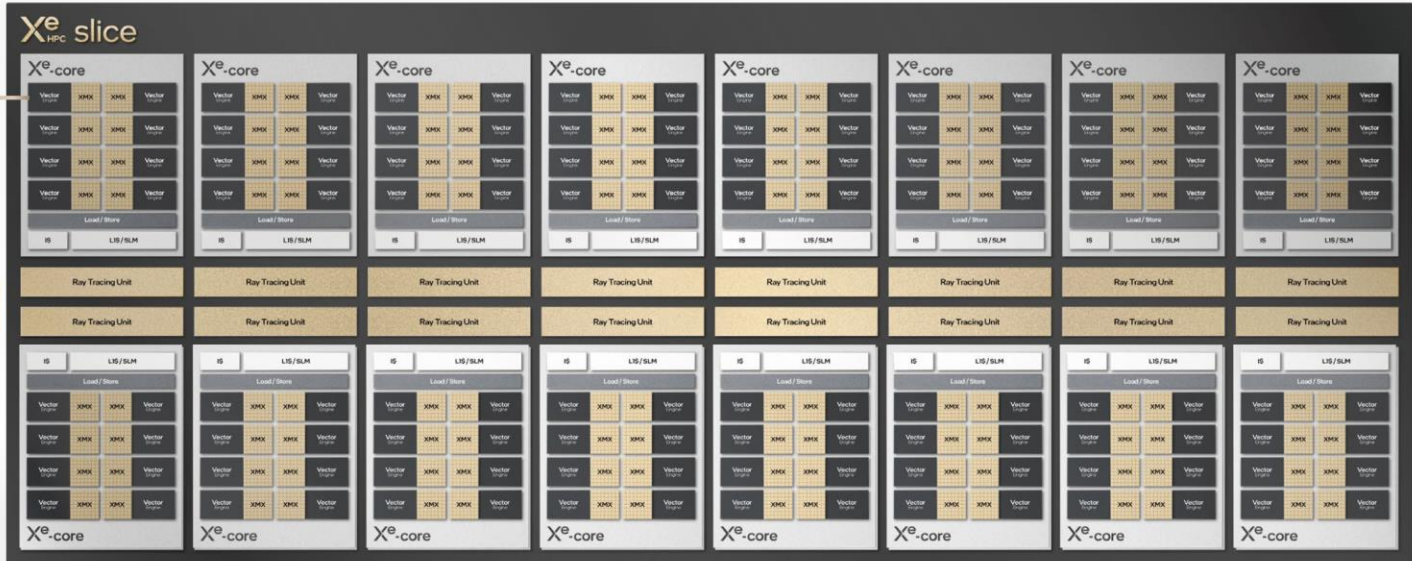
Xe HPC Slice

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



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 Stack

Up to

4 Slices

64 Xe^e - cores

64 Ray Tracing Units

4 Hardware Contexts

L2 Cache

4 HBM2e controllers

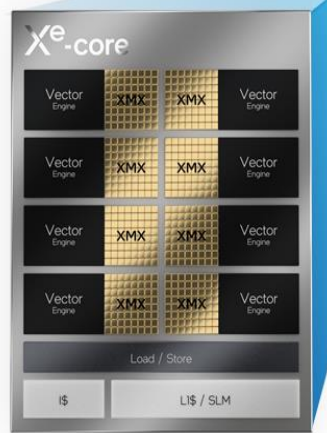
1 Media Engine

8 Xe^e Links

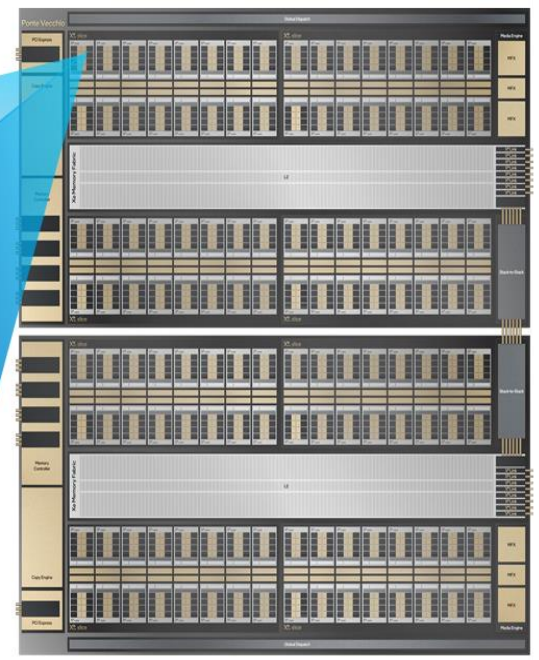


Xe HPC Architecture

2 Stacks	128 Xe-cores 8 Hardware Contexts
8	HBM2e controllers
16	Xe Links



Xe core

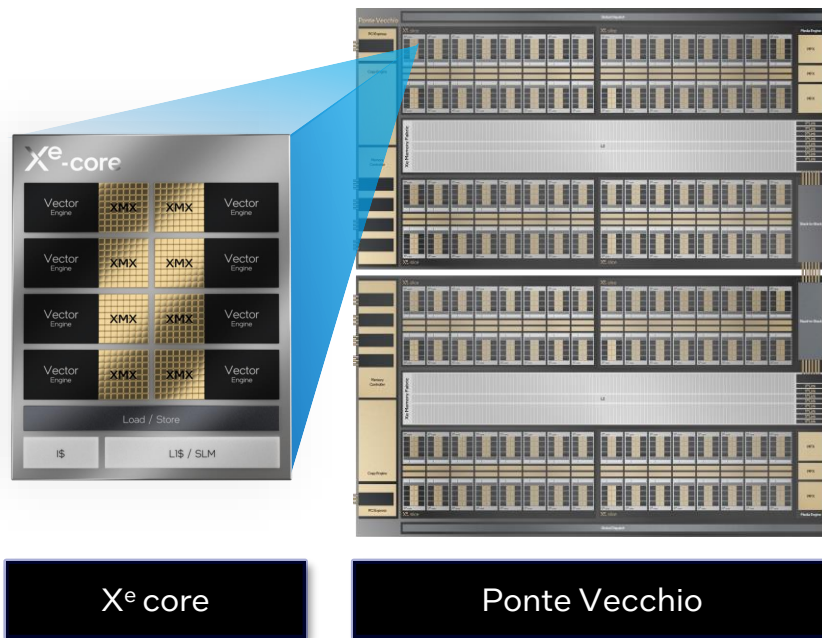


Ponte Vecchio 2-Stack

Ponte Vecchio – Compute Throughput

Peak Throughput	Ponte Vecchio 2-Stack
FP64	52 TFLOPS
FP32	52 TFLOPS
XMV Float 32 (TF32)	419 TFLOPS
XMV BF16	839 TFLOPS
XMV FP16	839 TFLOPS
XMV INT8	1678 TOPS

XMV: X^e Matrix Extensions



Ponte Vecchio - Memory Hierarchy

Large Bandwidth and Cache bring data close to Compute

Ponte Vecchio 2-Stack	Register File	L1 Cache	L2 Cache	HBM
Maximum Size	64 MB	64 MB	408 MB	128 GB
Peak Read Bandwidth	419 TB/s	105 TB/s	13 TB/s	3.2 TB/s

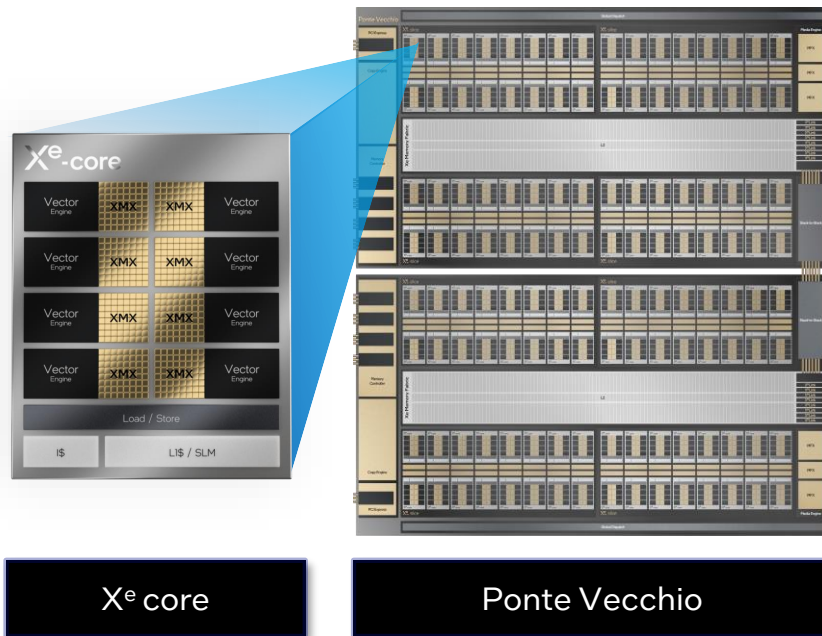
1:1

1~6

4:1

8:1

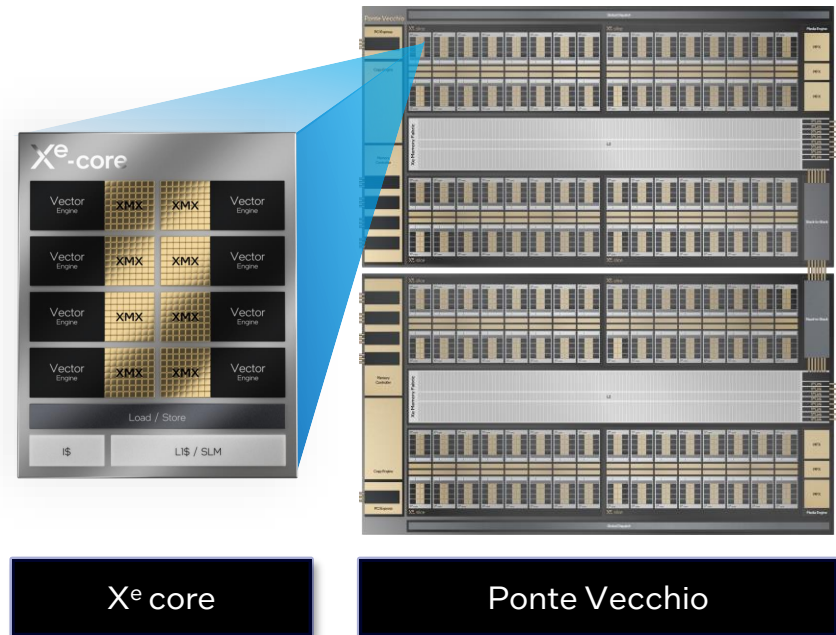
4:1



Ponte Vecchio - Memory Hierarchy

Compute Efficiency Techniques

	Techniques
Register File	<ul style="list-style-type: none">▪ Register caching▪ Accumulators
L1 and L2 Cache	<ul style="list-style-type: none">▪ Uncached▪ Write Bypass▪ Write Back▪ Write Through▪ Write Streaming
Prefetch	<ul style="list-style-type: none">▪ Software (instruction) prefetch to L1 and/or L2▪ Command Streamer prefetch to L2 for instructions and data

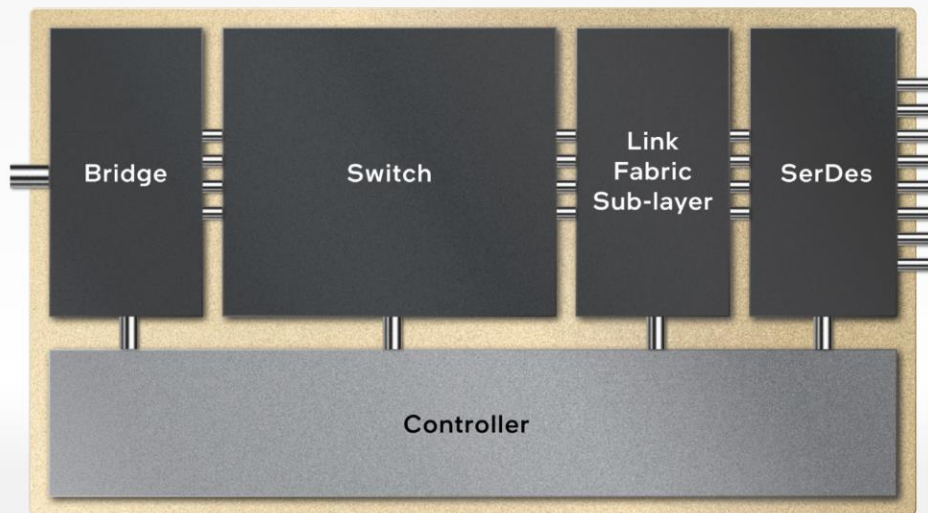


Xe Link

High Speed Coherent
Unified Fabric (GPU to GPU)

Load/Store, Bulk Data Transfer &
Sync Semantics

Up to 8 Fully Connected GPUs
through Embedded Switch



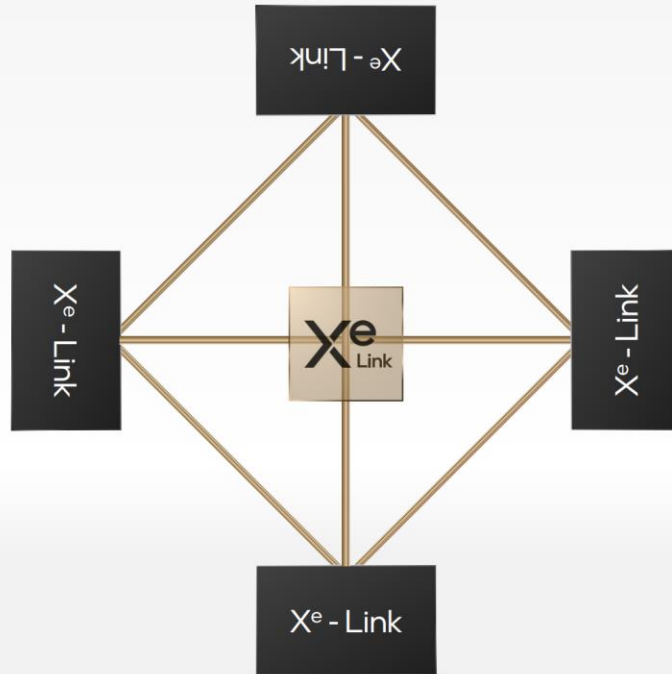


Link for Scalability



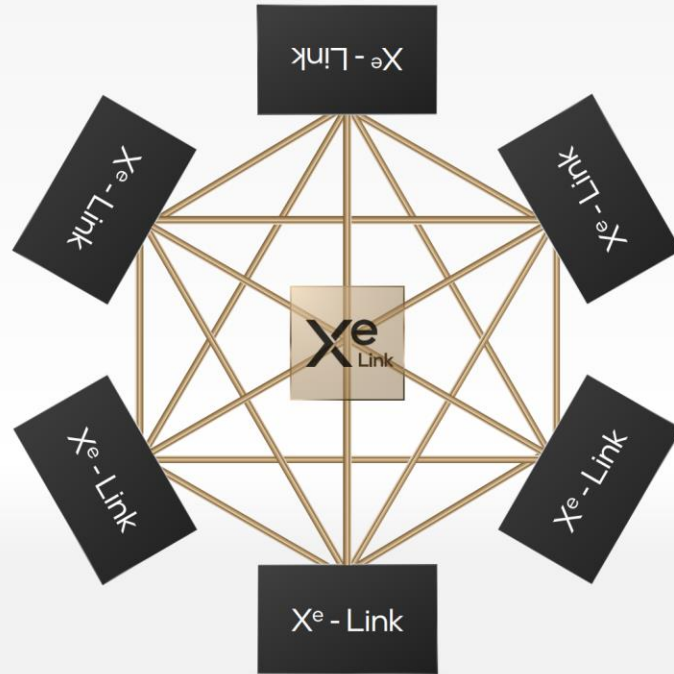


Link for Scalability



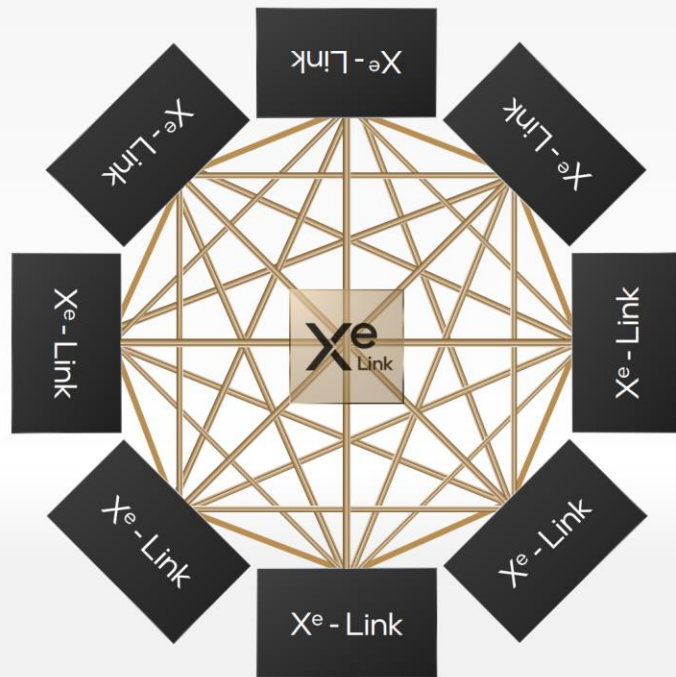


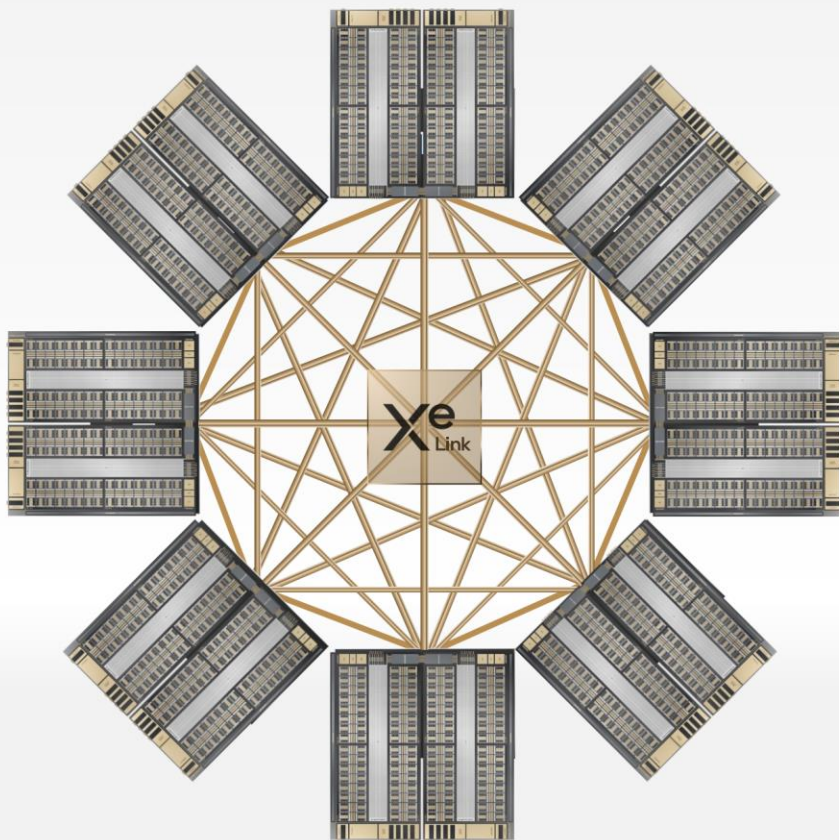
Link for Scalability





Link for Scalability





Xe HPC

8x System Compute Rates

	Vector	Matrix
8x	Up to 32,768 FP64 Ops/CLK	8x Up to 262,144 TF32 Ops/CLK
8x	Up to 32,768 FP32 Ops/CLK	8x Up to 524,288 BF16 Ops/CLK
		8x Up to 1,048,576 INT8 Ops/CLK

Accelerated Compute Systems

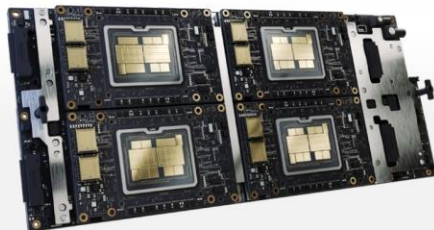
Ponte Vecchio x4 Subsystem
with X^e Links

+ 2S Sapphire Rapids

Ponte Vecchio
OAM



Ponte Vecchio
x4 Subsystem
with X^e Links



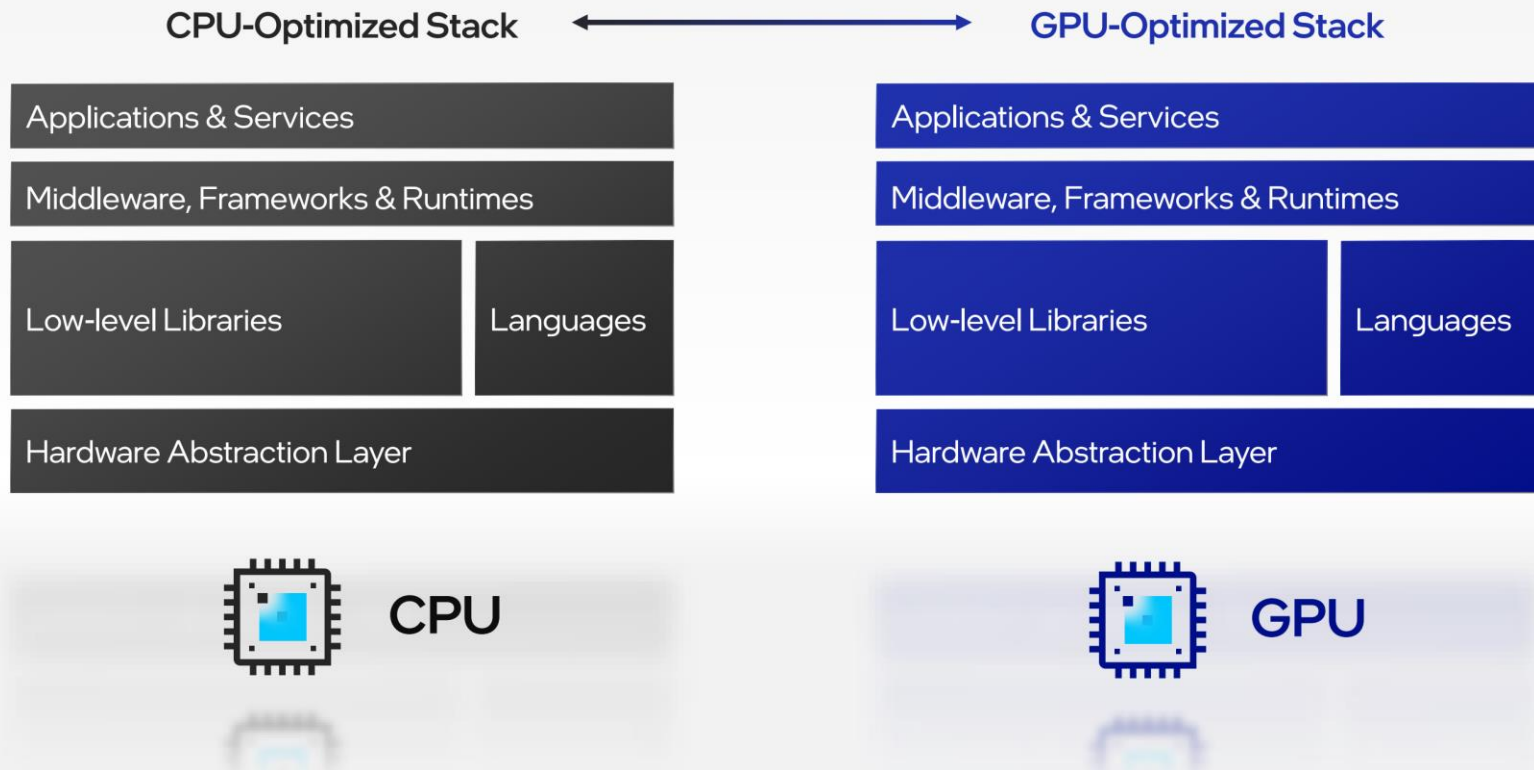
Agenda

Ponte Vecchio Architecture

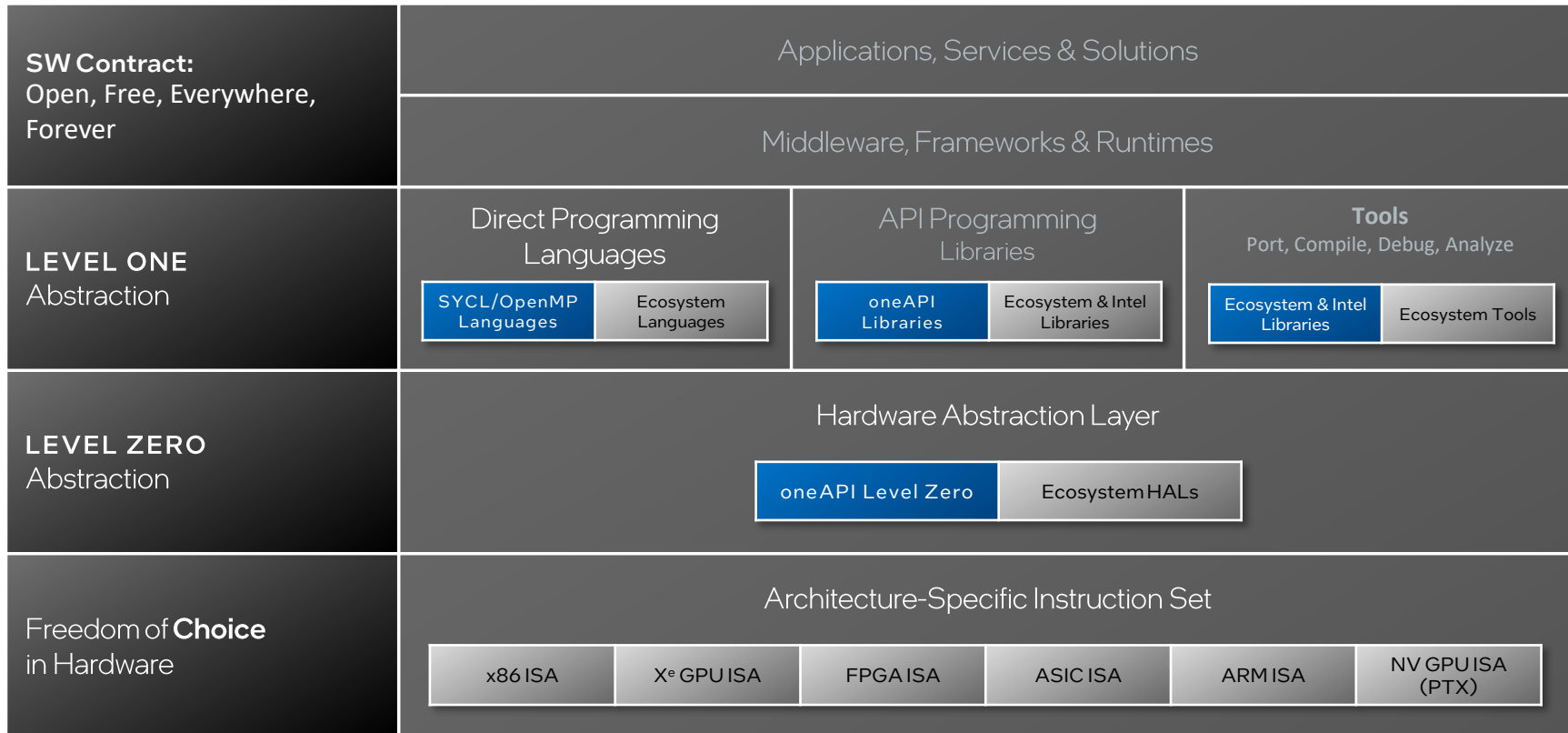
oneAPI Software Stack

Application Performance

Overcoming Separate CPU and GPU Software Stacks



oneAPI: the Big Picture

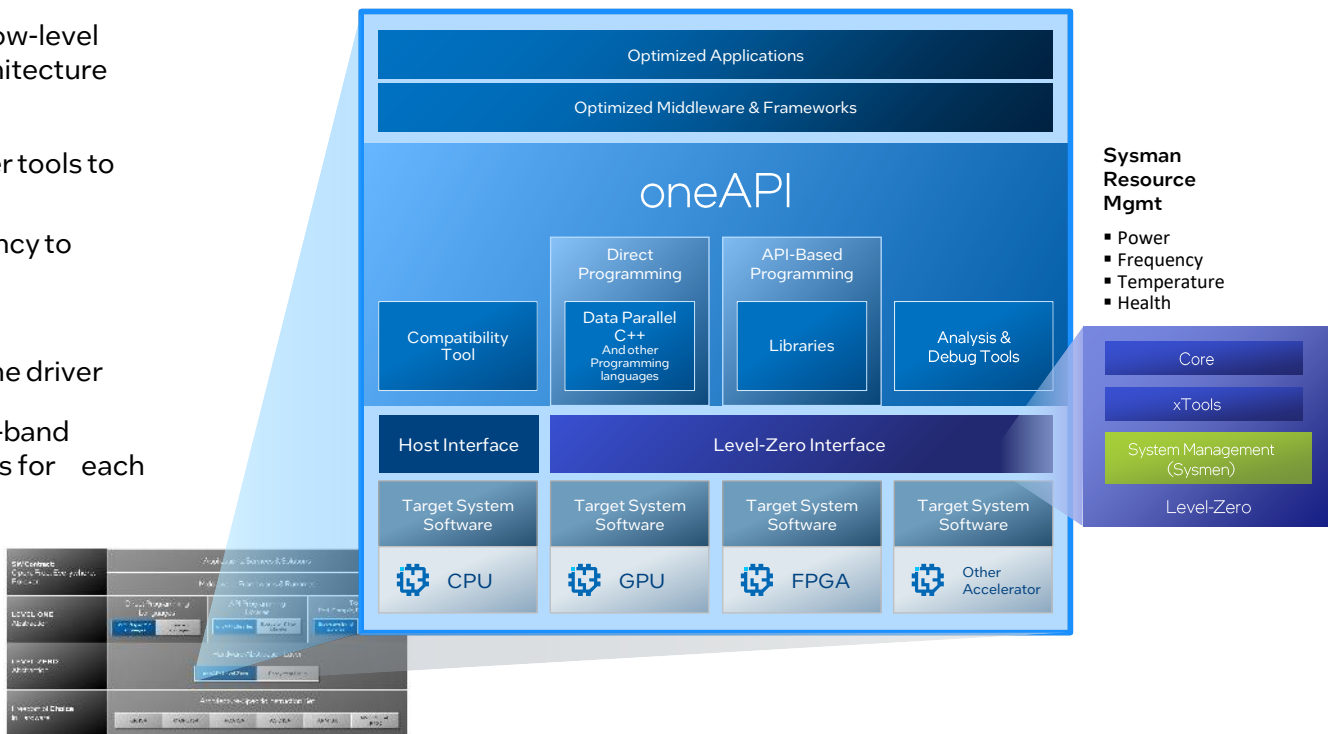


oneAPI and the Level-Zero APIs

The oneAPI **Level-Zero APIs** provide a low-level hardware interface to support cross-architecture programming

- Interface for oneAPI and other tools to accelerator devices
- Fine gain control and low latency to accelerator capabilities
- Multi-threaded design
- For GPUs, ships as a part of the driver

The Level-Zero **Sysman APIs** provide in-band access to resource management features for each accelerator device



Intel® DPC++ Compatibility Tool

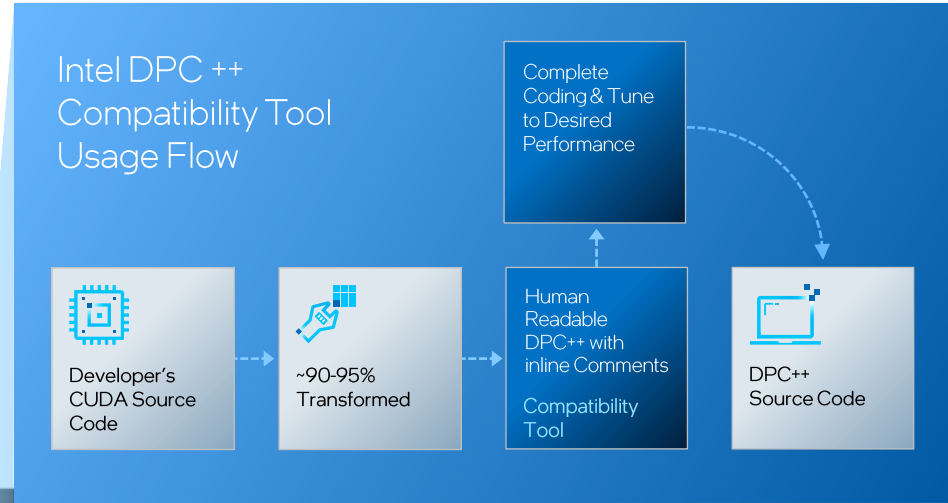
Minimizes Code Migration Time

Assists developers migrating code written in CUDA* to SYCL once, generating human readable code wherever possible

~90-95% of code typically migrates automatically

Inline comments are provided to help developers finish porting the application

Open Source build, SYCLomatic, available today



Intel® DPC++ Compatibility Tool Resources including Training and Examples
<https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/dpc-compatibility-tool.html>

- oneAPI GPU Optimization Guide
<https://www.intel.com/content/www/us/en/develop/documentation/oneapi-gpu-optimization-guide/top.html>
- SYCLomatic: <https://github.com/oneapi-src/SYCLomatic>

Customer Testimonials and Samples

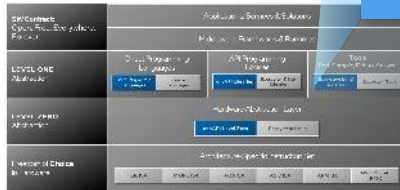
Samsung experience with Compatibility Tool and Vtune

<https://www.youtube.com/watch?v=XB1y5MxRfM>

Multiple examples of others in industry that have used tool

<https://www.intel.com/content/www/us/en/newsroom/news/intel-announces-oneapi-challenge-winners.html>

Argonne National Lab Webinar <https://www.alcf.anl.gov/support-center/aurora/intel-dpc-compatibility-tool>



Ponte Vecchio Supports both SIMT and SIMD Models

SPMD/SIMT:

- SYCL and OpenMP
- Thread Group size of 32 and 16
- Common GPU Programming Model
- DPC++ compatibility tool

SIMD:

- SYCL eSIMD and OpenMP
- Flexible SIMD width in offload regions
- Common CPU Programming Model
- Seamless transition from well-tuned CPU code

```
for each A
  code 1;
  for each B
    code 2;
  for each C
    code 3;
  code 4;
```

```
// SIMT code
for each A // Thread + SIMD32
  code 1;
  for each B
    code 2;
  for each C
    code 3;
  code 4;
```

```
// SIMD code
for each A // Thread
  code 1;
  for each B // SIMD32
    code 2;
  for each C // SIMD8
    code 3;
  code 4;
```

<https://intel.github.io/llvm-docs/>

SPMD Single Program Multiple Data
SIMT Single Instruction Multiple Thread
SIMD Single Instruction Multiple Data

Agenda

Ponte Vecchio Architecture

oneAPI Software Stack

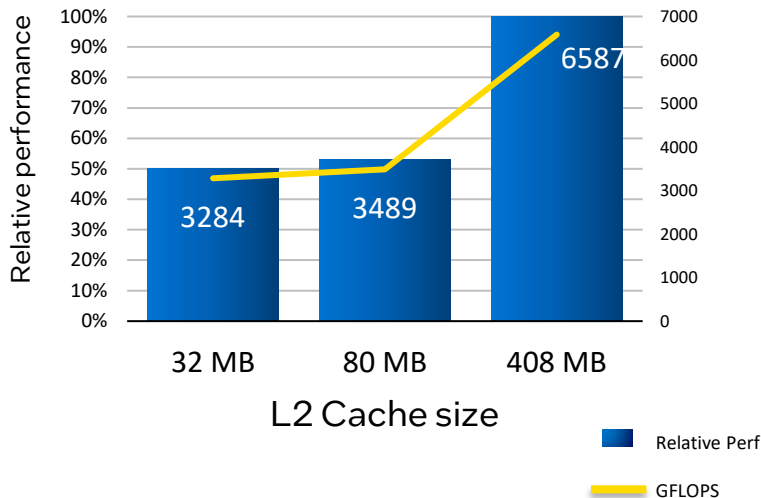
Application Performance

Workload Benefits from Large L2 Cache

2D-FFT Case

Larger L2 holds data between 1D transforms

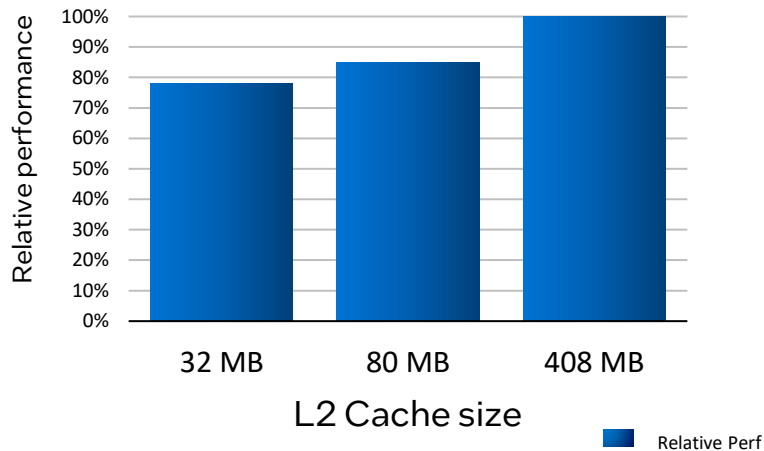
4Kx4K 2D-FFT DP Performance
Sensitivity to L2 Cache Size



DNN Case

Larger L2 captures activations between layers

ResNet-50 Training Performance
Sensitivity to L2 Cache Size

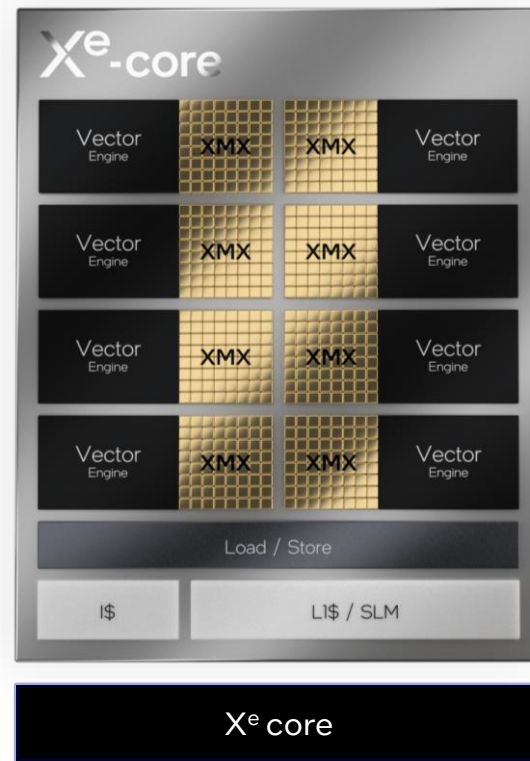


In this sensitivity study, the 408MB L2 cache in Ponte Vecchio is down-configured to 80MB and 32MB.

- 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.

High XMX Matrix Compute Efficiency

- Transpose Dataport:
 - Block data load/store
 - Address calculation offload
 - Hardware data padding
 - Hardware Transpose
- Latency Coverage:
 - Large RF/L1\$
 - Loop unrolling
 - Software prefetch

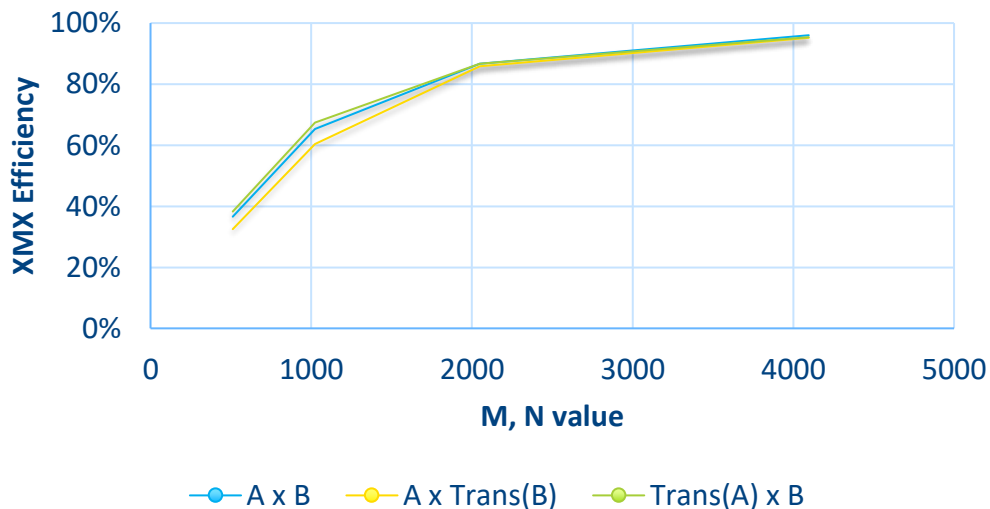


• See backup for workloads and configurations. Results may vary.

High XMX Matrix Compute Efficiency

- Transpose Dataport:
 - Block data load/store
 - Address calculation offload
 - Hardware data padding
 - Hardware Transpose
- Latency Coverage:
 - Large RF/L1\$
 - Loop unrolling
 - Software prefetch
- Roofline performance, including transpose, achieved using C++ code

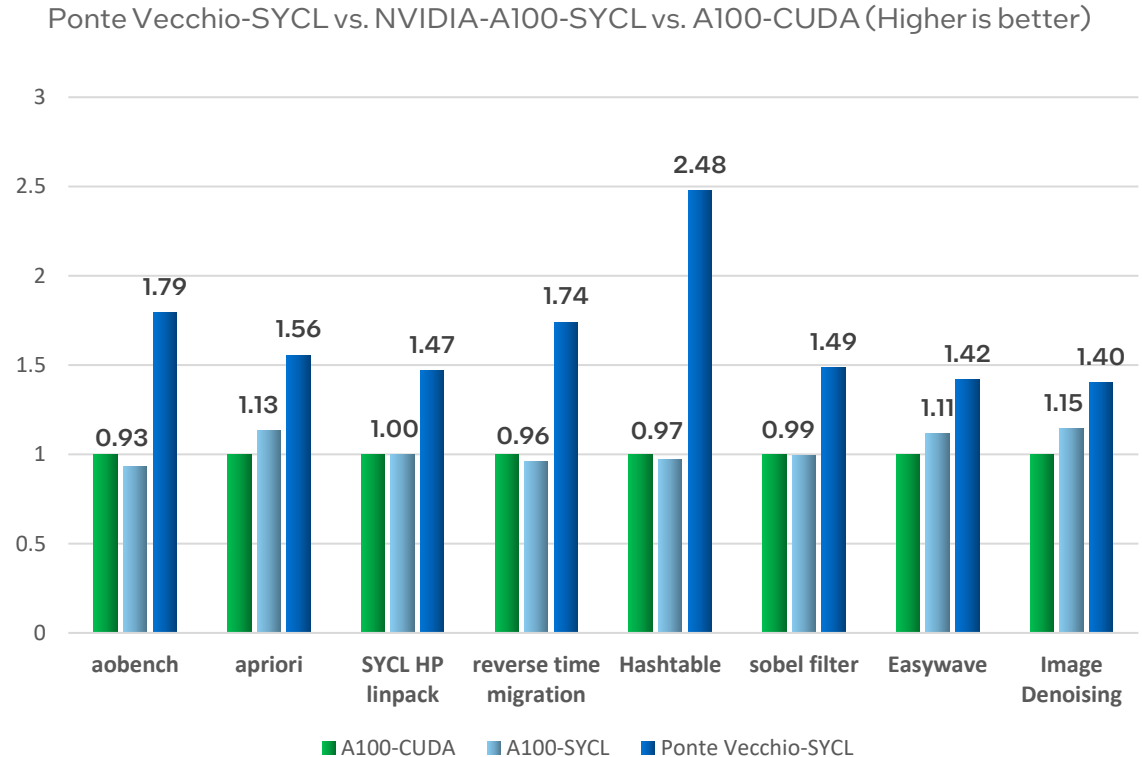
BF16-GEMM EFFICIENCY (K=4096)



> 95% BGEMM Efficiency

Intel® DPC++ Compatibility Tool Results

- Start with some popular legacy CUDA workloads
- Migrated to SYCL using DPC++ Compatibility Tool
 - Comparable performance on competitor's hardware
- Same performance portable SYCL code runs on Ponte Vecchio
 - Ponte Vecchio outperforms NVIDIA A100, up to 2.5x



• See backup for workloads and configurations. Results may vary.

Seamless Transition of CPU SIMD Codes

Start with legacy OpenMP SIMD codes, well-tuned on CPU

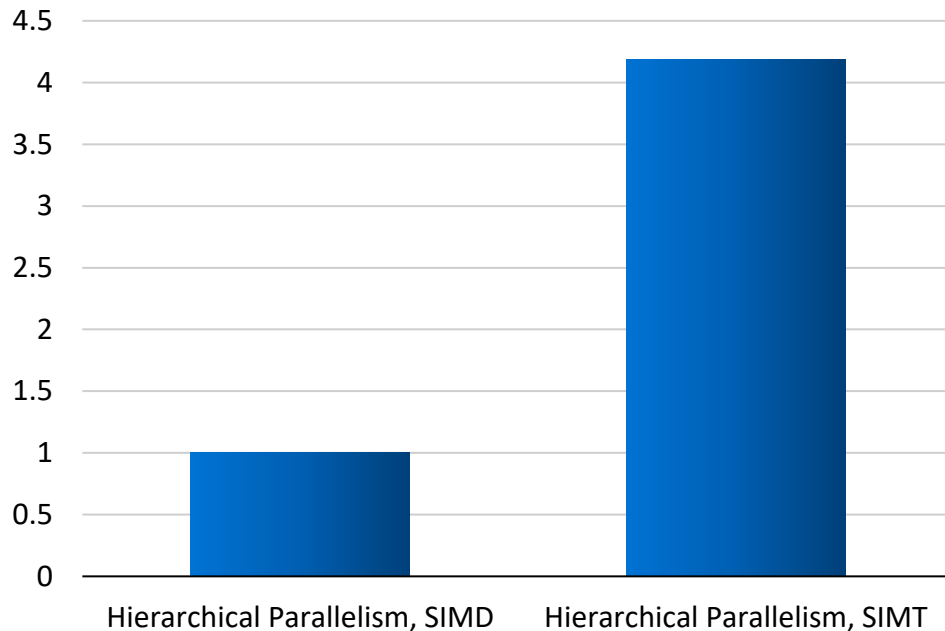
- C, C++, FORTRAN, etc.

Transition to Ponte Vecchio without code refactorization

Quickly achieving performance target on Ponte Vecchio with OpenMP SIMD

- SPMD code needs more tuning

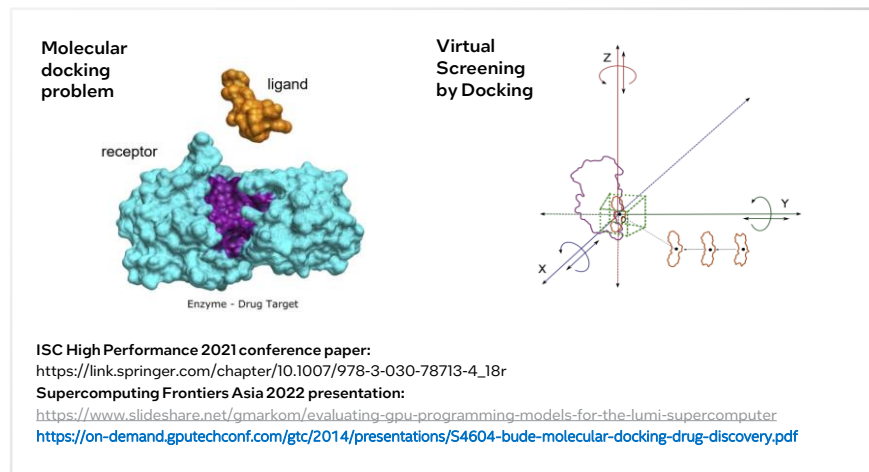
HACC on Ponte Vecchio, 16 MPI Ranks
(Relative time, lower is better)



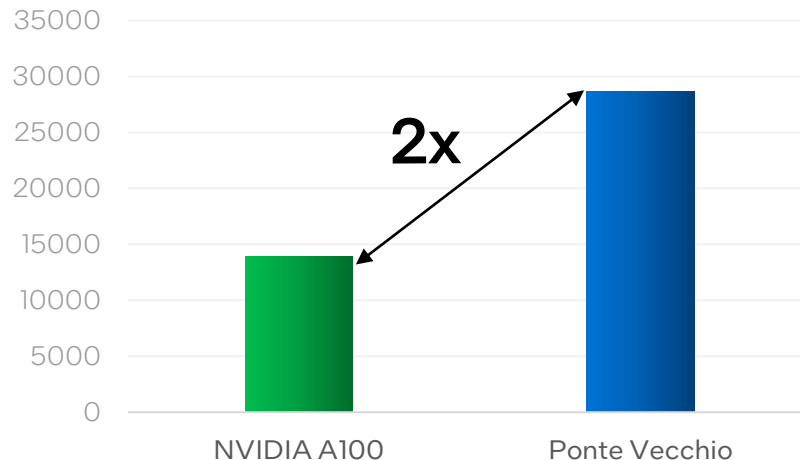
miniBUDE Performance on Ponte Vecchio

miniBUDE, core computation of the Bristol University Engine (BUDE)

Ponte Vecchio with Intel OneAPI DPC++ implementation
2x performance lead



miniBUDE Throughput on Workload BIG5 with 983040 Poses, 2672 Ligands, 2672 Proteins (GFLOPS, higher is better)



Application Summary: MiniBude is an implementation of the core computation of the Bristol University Docking Engine (BUDE) in different HPC programming models, using a tuned empirical free-energy forcefield to predict the binding energy of the ligand with the target. The benchmark is a virtual screening run of the NDM-1 protein and runs the energy evaluation for a single generation of poses repeatedly, for a configurable number of iterations. Increasing the iteration count has similar performance effects to docking multiple ligands back-to-back in a production BUDE docking run.

<https://github.com/UoB-HPC/miniBUDE>

- 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.

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/>

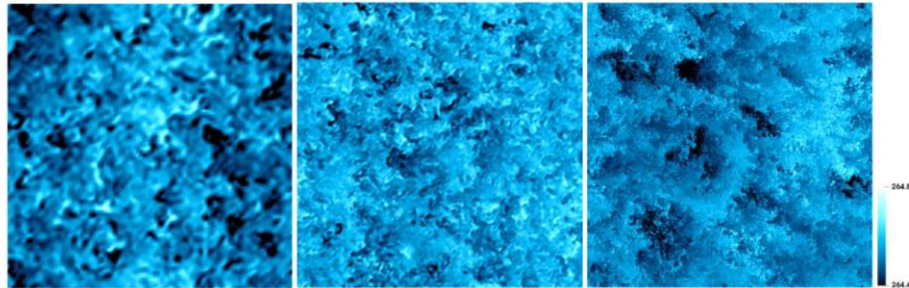
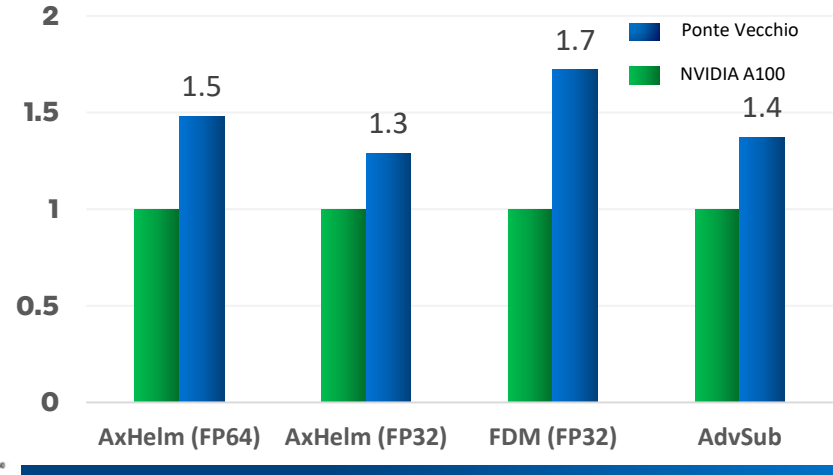


Figure 10: NekRS: potential temperature distributions in [K] at time 6h and $z=100\text{m}$ on different resolutions of $\Delta x=3.12\text{m}$ (left), 1.56m (center), and 0.78m (right) corresponding to the number of grid points, $n=128^3$, 256^3 , and 512^3 , respectively. Δx represents the average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

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

- 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.

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/>

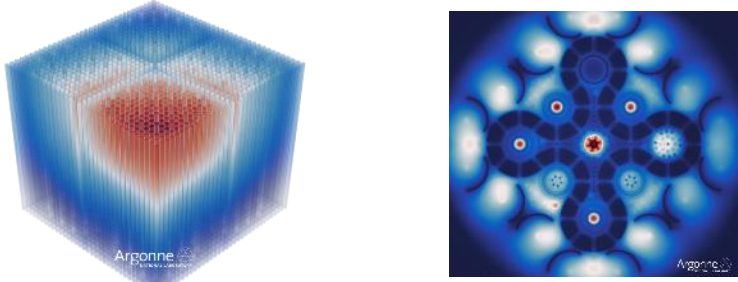
OCCA is an open-source library which aims to make it easy to program different types of devices (e.g. CPU, GPU, FPGA). It provides a unified API for interacting with backend device APIs (e.g. OpenMP, CUDA, OpenCL), uses just-in-time compilation to build backend kernel, and provide a kernel language,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

ExaSMR: OpenMC Performance on Ponte Vecchio

Monte Carlo particle transport code for exascale computations

Ponte Vecchio with OpenMP Target offload

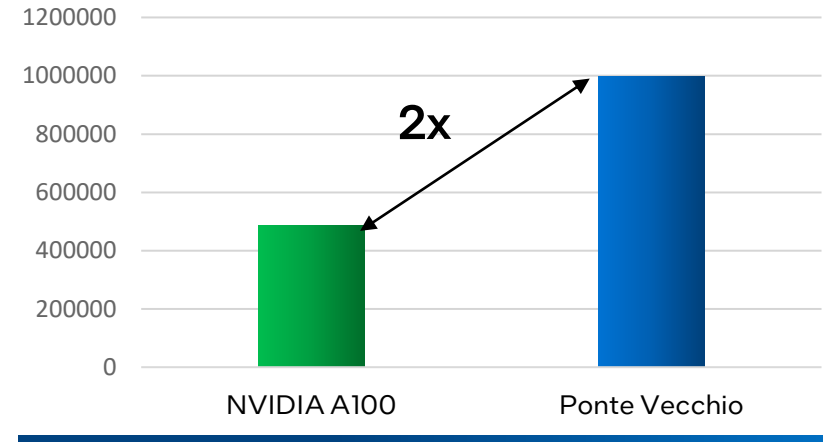
2x performance lead



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.

Summary

Largest high bandwidth cache in a GPU

Open SW model with SPMD/SIMD flexibility

Leadership performance

Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more on the Performance Index site.

Software and workloads used in performance tests may have been optimized for performance only on Intel products.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Results that are based on pre-production systems and components as well as results that have been estimated or simulated using an Intel Reference Platform (an internal example new system), internal Intel analysis or architecture simulation or modeling are provided to you for informational purposes only. Results may vary based on future changes to any systems, components, specifications, or configurations. Intel technologies may require enabled hardware, software or service activation.

Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not "commercial" names and not intended to function as trademarks.

Statements in this document that refer to future plans or expectations are forward-looking statements. These statements are based on current expectations and involve many risks and uncertainties that could cause actual results to differ materially from those expressed or implied in such statements. For more information on the factors that could cause actual results to differ materially, see our most recent earnings release and SEC filings at www.intc.com.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more at www.intel.com/PerformanceIndex. Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See www.intel.com/ArchDay21claims for configuration details. No product or component can be absolutely secure.

All product plans and roadmaps are subject to change without notice. Results that are based on pre-production systems and components as well as results that have been estimated or simulated using an Intel Reference Platform (an internal example new system), internal Intel analysis or architecture simulation or modeling are provided to you for informational purposes only. Results may vary based on future changes to any systems, components, specifications, or configurations. Intel technologies may require enabled hardware, software or service activation.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document. Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not "commercial" names and not intended to function as trademarks.

Intel contributes to the development of benchmarks by participating in, sponsoring, and/or contributing technical support to various benchmarking groups, including the BenchmarkXPRT Development Community administered by Principled Technologies.

Statements in this presentation that refer to future plans and expectations are forward-looking statements that involve a number of risks and uncertainties. Words such as "anticipates," "expects," "intends," "goals," "plans," "believes," "seeks," "estimates," "continues," "may," "will," "would," "should," "could," and variations of such words and similar expressions are intended to identify such forward-looking statements. Statements that refer to or are based on estimates, forecasts, projections, uncertain events or assumptions, including statements relating to future products and technology and the expected availability and benefits of such products and technology, market opportunity, and anticipated trends in our businesses or the markets relevant to them, also identify forward-looking statements. Such statements are based on management's current expectations and involve many risks and uncertainties that could cause actual results to differ materially from those expressed or implied in these forward-looking statements. Important factors that could cause actual results to differ materially from the company's expectations are set forth in Intel's reports filed or furnished with the Securities and Exchange Commission (SEC), including Intel's most recent reports on Form 10-K and Form 10-Q, available at Intel's investor relations website at www.intc.com and the SEC's website at www.sec.gov. Intel does not undertake, and expressly disclaims any duty, to update any statement made in this presentation, whether as a result of new information, new developments or otherwise, except to the extent that disclosure may be required by law.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

Recognition

Big thanks to Hong Jiang for help with materials.

System Configurations and Performance Data Sources

Cache Benefits for 2D-FFT and DNN: Test by Intel as of 8/1/2022, 2s future Intel Xeon CPU codenamed Sapphire Rapids, 1x pre-production two-stack Ponte Vecchio GPU, Ubuntu 20.04, pre-production oneAPI, configuring L2 Cache to 408MB, 80MB, and 32MB. 2D-FFT configuration was with 4K*4K DP 2D FFT. ResNet-50 v1.5 used xemlbench, an internal python software framework; Explicit scaling with Batch-128 on each the two Ponte Vecchio stacks; Dataset: ImageNet2012, Precision: BF16/FP32 mixed, Accuracy: 76.17% top1.

MMX Matrix Compute Efficiency: Test by Intel as of 8/1/2022, 2s future Intel Xeon CPU codenamed Sapphire Rapids, 1x pre-production two-stack Ponte Vecchio GPU, Ubuntu 20.04, oneAPI pre-production software. Internal engineering benchmark software; Explicit scaling with the matrix operations with the given dimensions on each the two Ponte Vecchio stacks.

Intel® DPC++ Compatibility Tool Results: Test by Intel as of 8/1/2022, NVIDIA Config: 2s Intel® Xeon® Platinum 8360Y, PCIe NVIDIA A100 80GB. Software: SYCL open source/CLANG 15.00, CUDA SDK 11.6 with NVIDIA-NVCC 11.6.55, cuMath 11.6, cuDNN 11.6, Ubuntu 21.10. SYCL open source/CLANG compiler switches: -fsycl-targets=nvptx64-nvidia-cuda. NVIDIA NVCC compiler switches: -O3 -gencode arch=compute_80,code=sm_80. Represented workloads with Intel Internal optimizations. Intel Config: Intel pre-production platform with 2s 4th gen Intel® Xeon® Scalable and 1x two-stack Ponte Vecchio GPU running pre-production oneAPI, Ubuntu 20.04.

HACC SIMD and SIMT Results: Test by Intel as of 8/1/2022, 2s future Intel Xeon CPU codenamed Sapphire Rapids HBM, 1x pre-production two-stack Ponte Vecchio GPU, SLES 15 SP3m, Ubuntu 20.04, HACC settings: 16 ranks, Np3 = 5043 particles, Geometry=4x2x2. SIMT HACC version from <https://asc.llnl.gov/coral-2-benchmarks>. SIMD HACC version is Intel Internal, utilizing MPI + OpenMP.

miniBUDE: Tested by Intel as of 8/1/2022. BIG5 dataset (2672 Ligands, 2672 proteins, and 983040 poses) <https://github.com/csahpc/epmhpcgpu/tree/main/miniBUDE/big5>, NVIDIA Config: 2s Intel® Xeon® Platinum 8360Y, PCIe NVIDIA A100 80GB, CUDA 11.4, Intel Config: 2s Intel® Xeon® Platinum 8360Y, 1x two-stack pre-production Ponte Vecchio GPU, Ubuntu 20.04, pre-production oneAPI.

NekRS: Tested by Intel as of 8/1/2022. Benchmark: NekRS AxHelm (BK5) FP64, AxHelm (BK5) FP32, FDM FP32 and advSub with problem size of 8192 ($E = 2 \times 16^3$), NVIDIA Config: GPU: AMD EPYC 7532 32-Core Processor, PCIe NVIDIA A100 80GB, DDR4-3200 256GB (8x32G DIMMs) RAM, Intel Config: 2s Intel® Xeon® Platinum 8360Y @ 2.40GHz; Memory: 256 GB DDR4 3200, 1x two-stack pre-production Ponte Vecchio GPU, Ubuntu 20.04, pre-production oneAPI.

OpenMC: Test by Argonne National Laboratory as of 5/23/2022, 2x AMD EPYC 7532, 256 GB DDR4 3200, HT On, Turbo On, ucode 0x8301038. 1x A100 40GB PCIe. OpenSUSE Leap 15.3, Linux Version 5.3.18, Libraries: CUDA 11.6 with OpenMP clang compiler. Build Knobs: cmake --preset=llvm_a100 -DCMAKE_UNITY_BUILD=ON -DCMAKE_UNITY_BUILD_MODE=BATCH -DCMAKE_UNITY_BUILD_BATCH_SIZE=1000 -DCMAKE_INSTALL_PREFIX=./install -Ddebug=off -Doptimize=on -Dopenmp=on -Dnew_w=on -Ddevice_history=off -Ddisable_xs_cache=on -Ddevice_printf=off. Benchmark: Depleted Fuel Inactive Batch Performance on HM-Large Reactor with 40M particles Test By Intel as of 5/25/2022, 1-node, 2x Intel® Xeon® Scalable Processor 8360Y, 256GB DDR4 3200, HT On, Turbo, On, ucode 0xd0002c1. 1x Pre-production Ponte Vecchio. Ubuntu 20.04, Linux Version 5.10.54, agama 434, Build Knobs: cmake -DCMAKE_CXX_COMPILER="mpicpc" -DCMAKE_C_COMPILER="mpicc" -DCMAKE_CXX_FLAGS="-cxx=icpx -mllvm -indvars-widen-indvars=false -Xclang -fopenmp-declare-target-global-default-no-map -std=c++17 -Dgs_CONFIG_CONTRACT_CHECKING_OFF -fsycl -DSYCL_SORT -D_GLIBCXX_USE_TBB_PAR_BACKEND=0" --preset=spirv -DCMAKE_UNITY_BUILD=ON -DCMAKE_UNITY_BUILD_MODE=BATCH -DCMAKE_UNITY_BUILD_BATCH_SIZE=1000 -DCMAKE_INSTALL_PREFIX=./install -Ddebug=off -Doptimize=on -Dopenmp=on -Dnew_w=on -Ddevice_history=off -Ddisable_xs_cache=on -Ddevice_printf=off Benchmark: Depleted Fuel Inactive Batch Performance on HM-Large Reactor with 40M particles

