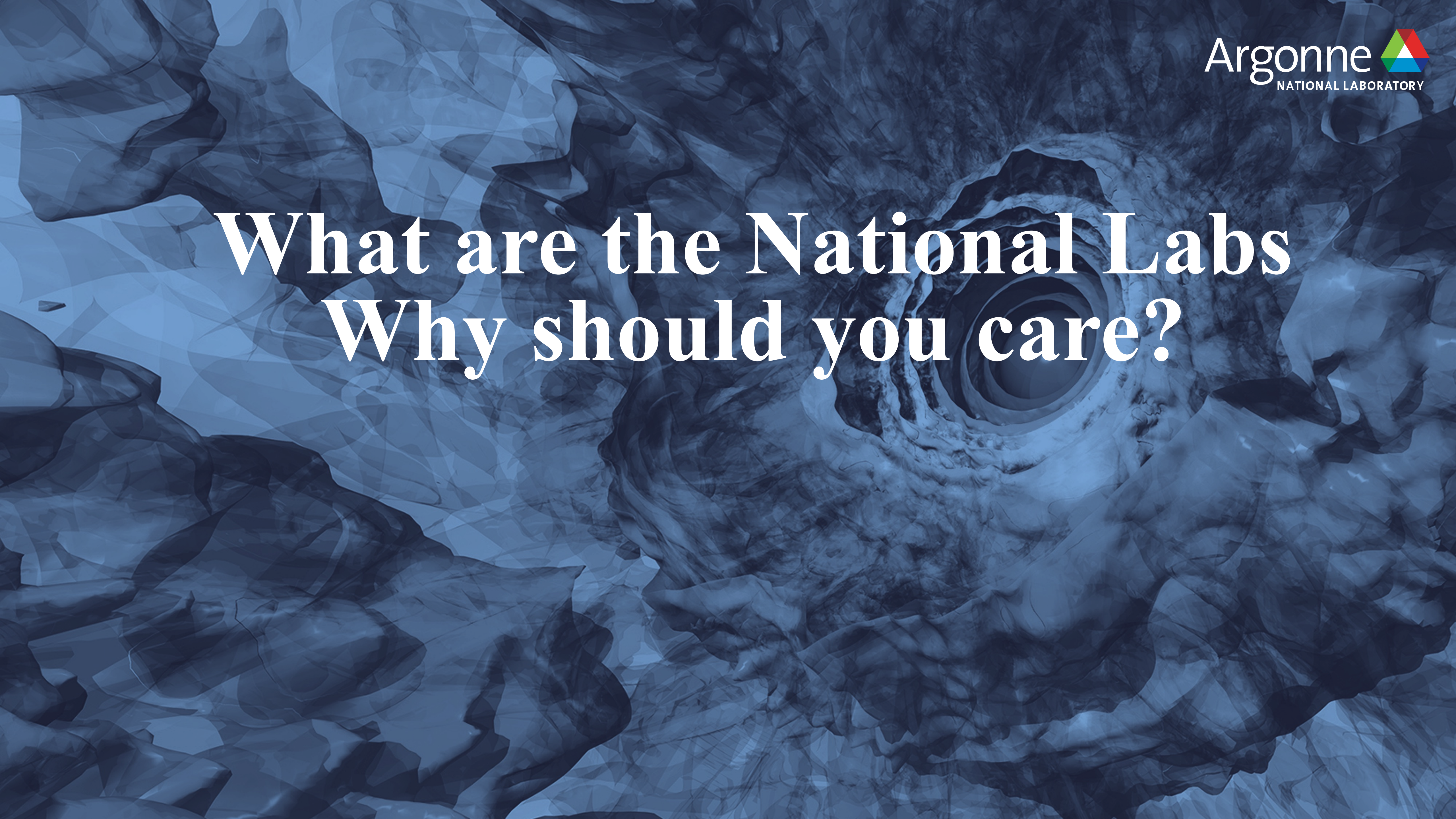
Argonne
NATIONAL LABORATORY

My View on the Future of Leadership Computing

William E Allcock, ALCF Director of Operations 23 May 2023

This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

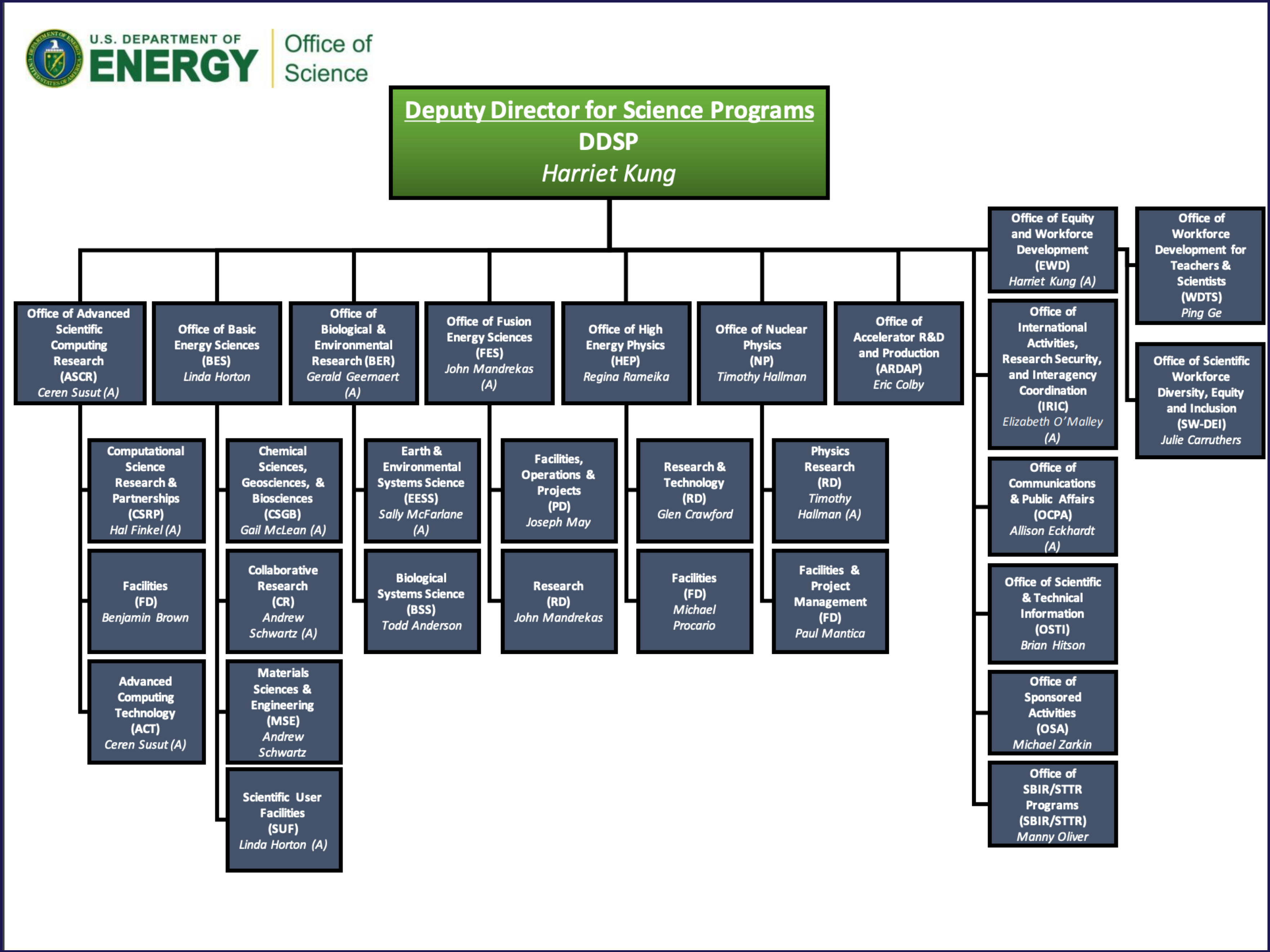


Argonne NATIONAL LABORATORY

What are the National Labs
Why should you care?

Background

- The United States Department of Energy (https://www.energy.gov/)
  - —Funds a significant amount of research in the US including the national laboratory system, which Argonne is a part of.
  - —https://www.energy.gov/maps/doe-national-laboratories
- Argonne National Laboratory – located outside of Chicago (https://www.anl.gov/)
- Argonne Leadership Computing Facility (ALCF)) https://www.alcf.anl.gov/)
  - —Who I work for; We are a “User Facility” and run supercomputers for scientists from around the world to use.
- “Leadership Computing”
  - —We want the biggest, baddest, computational science problems that just can’t be solved anywhere else.
- INCITE, ALCC, DD – These are the programs that allocate time on our systems
  - —https://www.doeleadershipcomputing.org/ (60%)
  - —https://science.osti.gov/ascr/Facilities/Accessing-ASCR-Facilities/ALCC (30%)
  - —https://www.alcf.anl.gov/dd-program (10%)



U.S. DEPARTMENT OF ENERGY | Office of Science

Deputy Director for Science Programs
DDSP
Harriet Kung

Office of Advanced Scientific Computing Research (ASCR) *Ceren Susut (A)*
- Computational Science Research & Partnerships (CSRP) *Hal Finkel (A)*
- Facilities (FD) *Benjamin Brown*
- Advanced Computing Technology (ACT) *Ceren Susut (A)*

Office of Basic Energy Sciences (BES) *Linda Horton*
- Chemical Sciences, Geosciences, & Biosciences (CSGB) *Gail McLean (A)*
- Collaborative Research (CR) *Andrew Schwartz (A)*
- Materials Sciences & Engineering (MSE) *Andrew Schwartz*
- Scientific User Facilities (SUF) *Linda Horton (A)*

Office of Biological & Environmental Research (BER) *Gerald Geernaert (A)*
- Earth & Environmental Systems Science (EESS) *Sally McFarlane (A)*
- Biological Systems Science (BSS) *Todd Anderson*

Office of Fusion Energy Sciences (FES) *John Mandrekas (A)*
- Facilities, Operations & Projects (PD) *Joseph May*
- Research (RD) *John Mandrekas*

Office of High Energy Physics (HEP) *Regina Rameika*
- Research & Technology (RD) *Glen Crawford*
- Facilities (FD) *Michael Procario*

Office of Nuclear Physics (NP) *Timothy Hallman*
- Physics Research (RD) *Timothy Hallman (A)*
- Facilities & Project Management (FD) *Paul Mantica*

Office of Accelerator R&D and Production (ARDAP) *Eric Colby*

Office of Equity and Workforce Development (EWD) *Harriet Kung (A)*
- Office of Workforce Development for Teachers & Scientists (WDTS) *Ping Ge*
- Office of Scientific Workforce Diversity, Equity and Inclusion (SW-DEI) *Julie Carruthers*

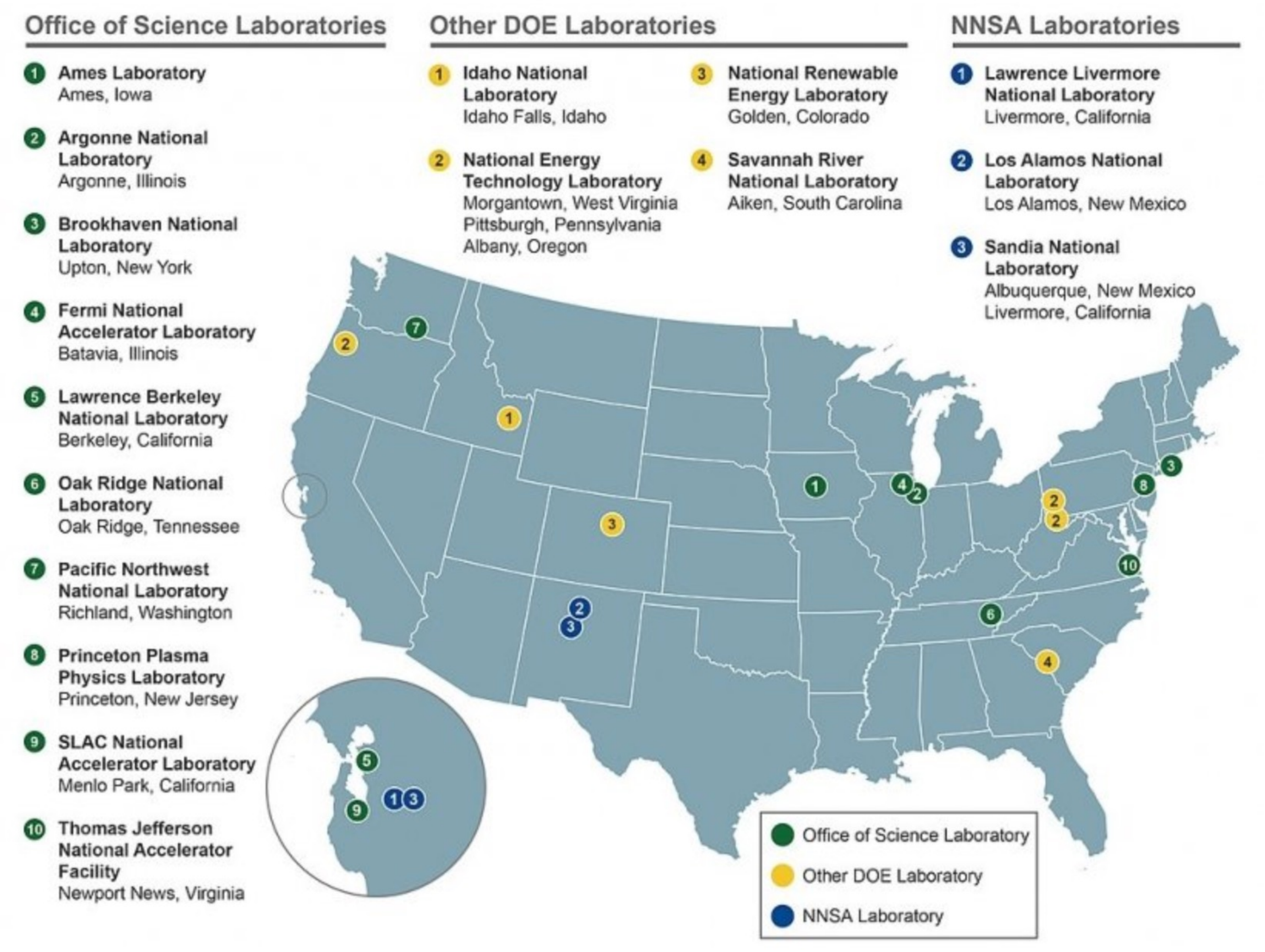
Office of International Activities, Research Security, and Interagency Coordination (IRIC) *Elizabeth O'Malley (A)*

Office of Communications & Public Affairs (OCPA) *Allison Eckhardt (A)*

Office of Scientific & Technical Information (OSTI) *Brian Hitson*

Office of Sponsored Activities (OSA) *Michael Zarkin*

Office of SBIR/STTR Programs (SBIR/STTR) *Manny Oliver*



Office of Science Laboratories

1. **Ames Laboratory**
   Ames, Iowa
2. **Argonne National Laboratory**
   Argonne, Illinois
3. **Brookhaven National Laboratory**
   Upton, New York
4. **Fermi National Accelerator Laboratory**
   Batavia, Illinois
5. **Lawrence Berkeley National Laboratory**
   Berkeley, California
6. **Oak Ridge National Laboratory**
   Oak Ridge, Tennessee
7. **Pacific Northwest National Laboratory**
   Richland, Washington
8. **Princeton Plasma Physics Laboratory**
   Princeton, New Jersey
9. **SLAC National Accelerator Laboratory**
   Menlo Park, California
10. **Thomas Jefferson National Accelerator Facility**
    Newport News, Virginia

Other DOE Laboratories

1. **Idaho National Laboratory**
   Idaho Falls, Idaho
2. **National Energy Technology Laboratory**
   Morgantown, West Virginia
   Pittsburgh, Pennsylvania
   Albany, Oregon
3. **National Renewable Energy Laboratory**
   Golden, Colorado
4. **Savannah River National Laboratory**
   Aiken, South Carolina

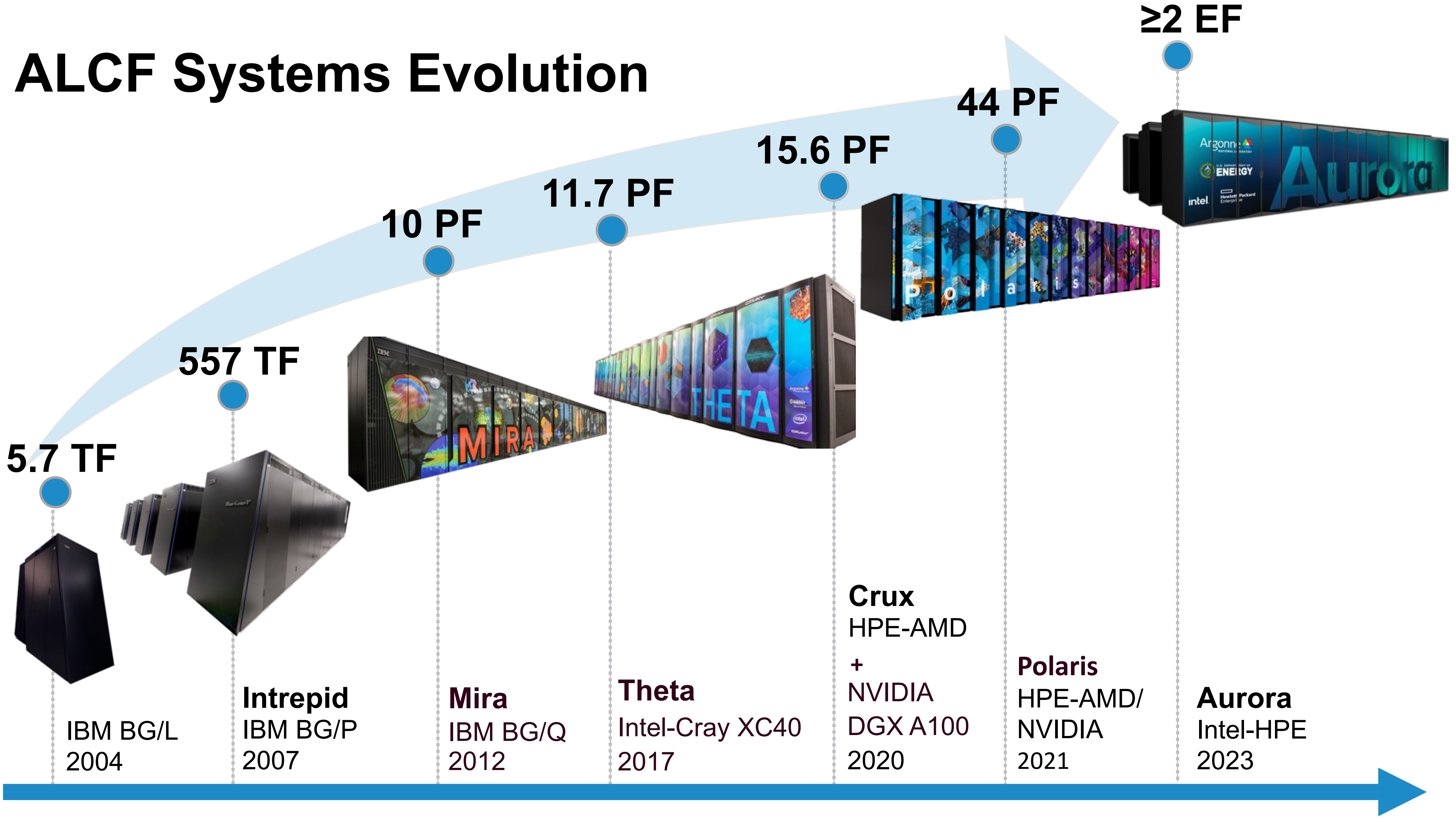
NNSA Laboratories

1. **Lawrence Livermore National Laboratory**
   Livermore, California
2. **Los Alamos National Laboratory**
   Los Alamos, New Mexico
3. **Sandia National Laboratory**
   Albuquerque, New Mexico
   Livermore, California

Argonne
NATIONAL LABORATORY

The ALCF Systems

ALCF Systems Evolution



5.7 TF

IBM BG/L
2004

557 TF

Intrepid
IBM BG/P
2007

10 PF

Mira
IBM BG/Q
2012

11.7 PF

Theta
Intel-Cray XC40
2017

15.6 PF

Crux
HPE-AMD
+
NVIDIA
DGX A100
2020

44 PF

Polaris
HPE-AMD/
NVIDIA
2021

≥2 EF

Aurora
Intel-HPE
2023

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

Milan

PLATFORM

HPE Apollo Gen10+

Compute Node
1 AMD EPYC 7543P 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

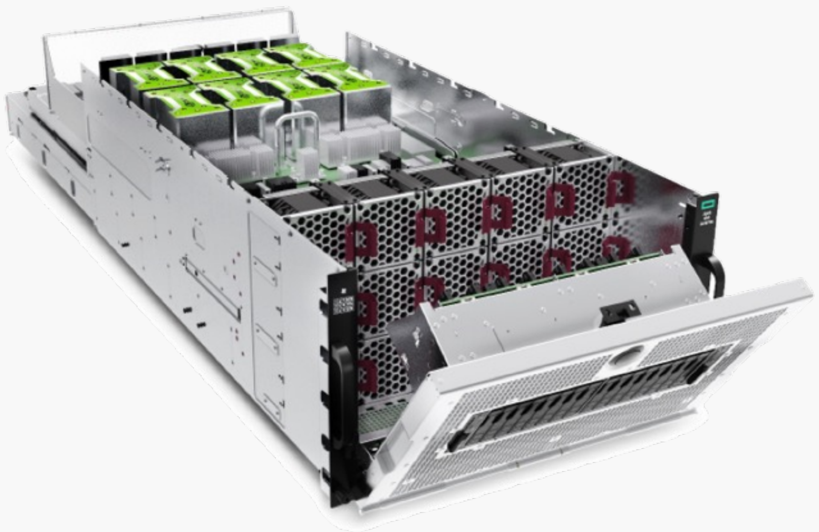
Network Switch
25.6 Tb/s per switch, from 64–200 Gb/s ports (25 GB/s per direction)

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

Node Performance
78 TF

Aggregate Memory
368 TB (CPU + GPU)

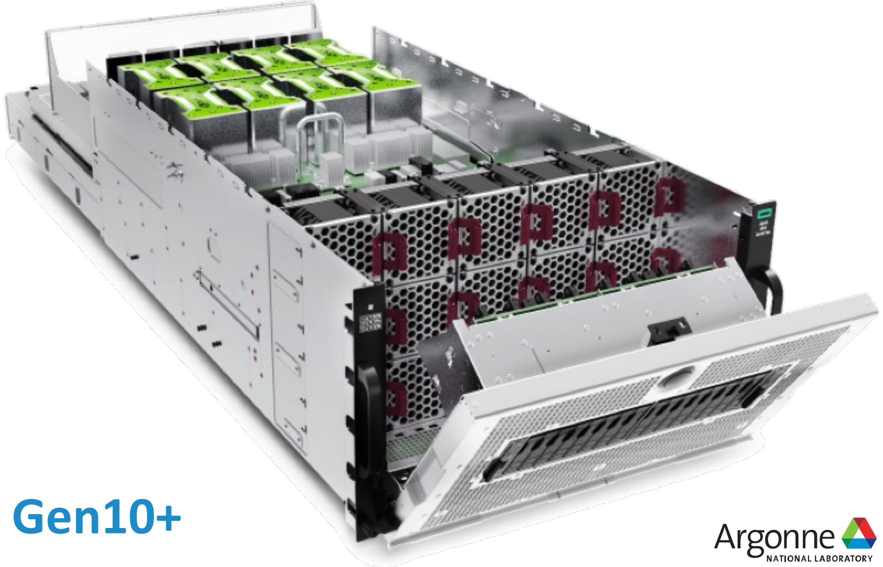
System Size
560 nodes, 1.8 MW



*Initial technology to be upgraded later

Polaris System Configuration

# of River Compute racks	40
# of Apollo Gen10+ Chassis	280
# of Nodes	560
# of AMD EPYC 7543P CPUs	560
# of NVIDIA A100 GPUs	2240
Total GPU HBM2 Memory	87.5TB
Total CPU DDR4 Memory	280 TB
Total NVMe SSD Capacity	1.75 PB
Interconnect	HPE Slingshot
# of Cassini NICs	1120
# of Rosetta Switches	80
Total Injection BW (w/ Cassini)	28 TB/s
Total GPU DP Tensor Core Flops	44 PF
Total Power	1.8 MW



Apollo 6500 Gen10+

Aurora

Leadership Computing Facility
Exascale Supercomputer

Peak Performance
≧ 2 Exaflops DP

Intel GPU
Intel® Data Center GPU Max Series 1550

Intel Xeon Processor
Intel® Xeon® CPU Max Series with HBM

Platform
HPE Cray-Ex

Compute Node
2 CPU, 6 GPU
Node Unified Memory Architecture
8 fabric endpoints

Node Performance
>130 TF

System Size
10,624 nodes
21,248 CPUs
63,744 GPUs

Aggregate System Memory

DDR	HBM	HBM GPU
10.9 PB	1.36 PB	8.16 PB
5.95 PB/s	30.5 PB/s	208.9 PB/s

System Interconnect
HPE Slingshot 11
Dragonfly topology w/ adaptive routing
2.12 PB/s Peak Injection BW
0.69 PB/s Peak Bisection BW

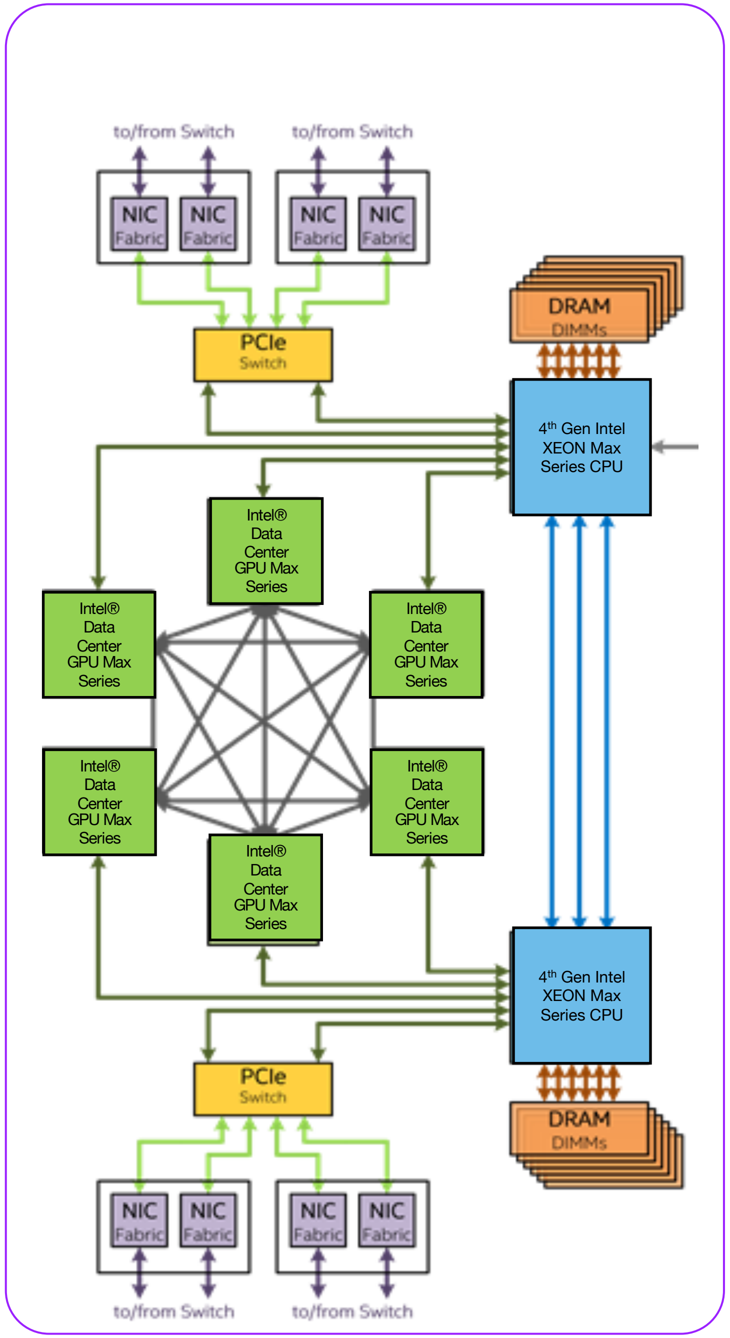
High-Performance Storage
230 PB
31 TB/s DAOS bandwidth
1024 DAOS nodes

Software Environment

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

Aurora Compute Node

- Six Intel® Data Center GPU Max Series
  - All to all connection
- Two 4th Gen Intel XEON Max Series CPUs with:
  - HBM memory
  - DDR memory
- Unified Memory Architecture across CPUs and GPUs
- 8 Slingshot Fabric endpoints



to/from Switch
to/from Switch
NIC Fabric
NIC Fabric
NIC Fabric
NIC Fabric
DRAM DIMMs
PCIe Switch
4th Gen Intel XEON Max Series CPU
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
4th Gen Intel XEON Max Series CPU
PCIe Switch
DRAM DIMMs
NIC Fabric
NIC Fabric
NIC Fabric
NIC Fabric
to/from Switch
to/from Switch

Aurora Cabinets Installed at Argonne

AI PATHFINDING

Goals of ALCF AI Activities at Argonne

Accelerate science by effective coupling of AI-systems, exascale supercomputers and experimental facilities

1. Maturity of software and hardware for science
2. Ability to scale hardware and integrate with facility
3. Application at scale to science

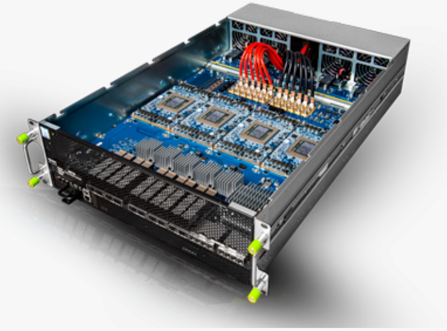
ALCF AI Testbeds

https://www.alcf.anl.gov/alcf-ai-testbed

Cerebras (CS-2)

SambaNova

Graphcore



Habana

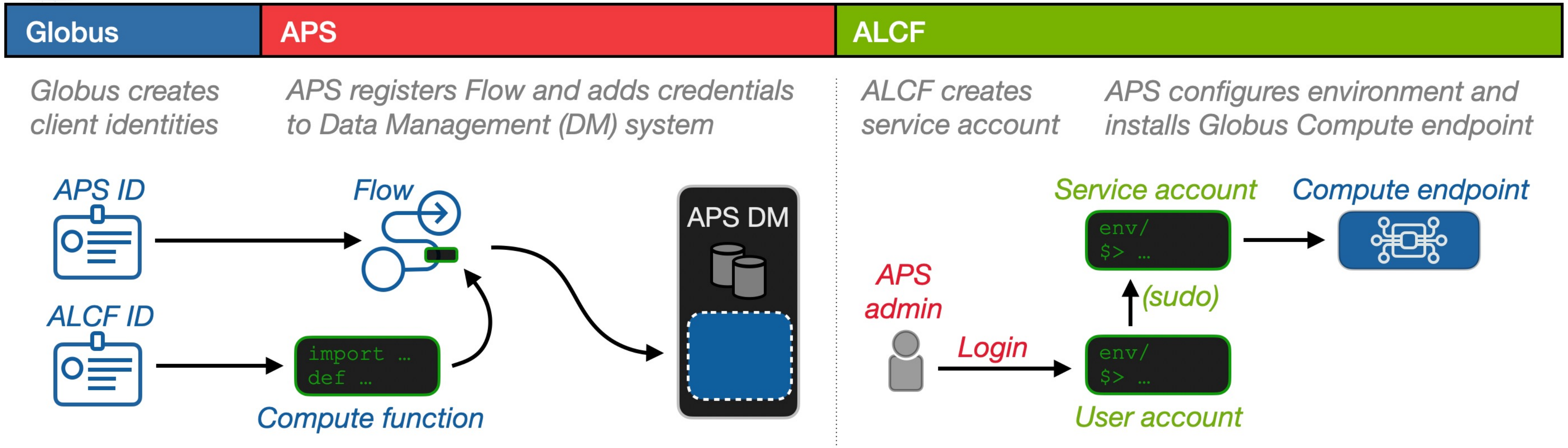
Groq

- Infrastructure of next-generation machines with hardware accelerators customized for artificial intelligence (AI) applications.
- Provide a platform to evaluate usability and performance of machine learning based HPC applications running on these accelerators.
- The goal is to better understand how to integrate AI accelerators with ALCF's existing and upcoming supercomputers to accelerate science insights

	Cerebras CS2	SambaNova Cardinal SN30	Groq GroqCard	GraphCore GC200 IPU	Habana Gaudi1	NVIDIA A100
Compute Units	850,000 Cores	640 PCUs	5120 vector ALUs	1472 IPUs	8 TPC + GEMM engine	6912 Cuda Cores
On-Chip Memory	40 GB L1 >1TB MemoryX	>300MB L1 1TB	230MB L1	900MB L1	24 MB L1 32GB	192KB L1 40MB L2 40-80GB
Process	7nm	7nm	7 nm	7nm	7nm	7nm
System Size	2 Nodes including Memory-X and Swarm-X	8 nodes (8 cards per node)	9 nodes (8 cards per node)	4 nodes (16 cards per node)	2 nodes (8 cards per node)	Several systems
Estimated Performance of a card (TFlops)	>5780 (FP16)	>660 (BF16)	>250 (FP16) >1000 (INT8)	>250 (FP16)	>150 (FP16)	312 (FP16), 156 (FP32)
Software Stack Support	Tensorflow, Pytorch	SambaFlow, Pytorch	GroqAPI, ONNX	Tensorflow, Pytorch, PopArt	Synapse AI, TensorFlow and PyTorch	Tensorflow, Pytorch, etc
Interconnect	Ethernet-based	Ethernet-based	RealScale ™	IPU Link	Ethernet-based	NVLink

Some of the Science at the ALCF

One-time configuration per beamline



Globus | APS | ALCF

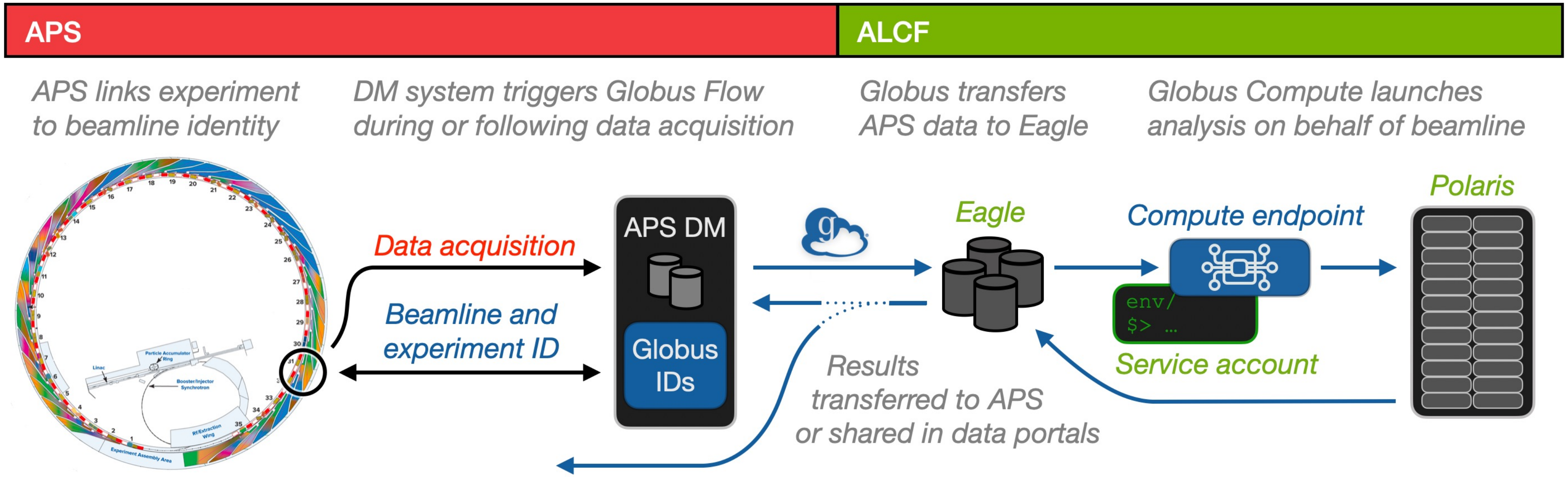
Globus creates client identities

APS registers Flow and adds credentials to Data Management (DM) system

ALCF creates service account

APS configures environment and installs Globus Compute endpoint

Automated workflow during experiments



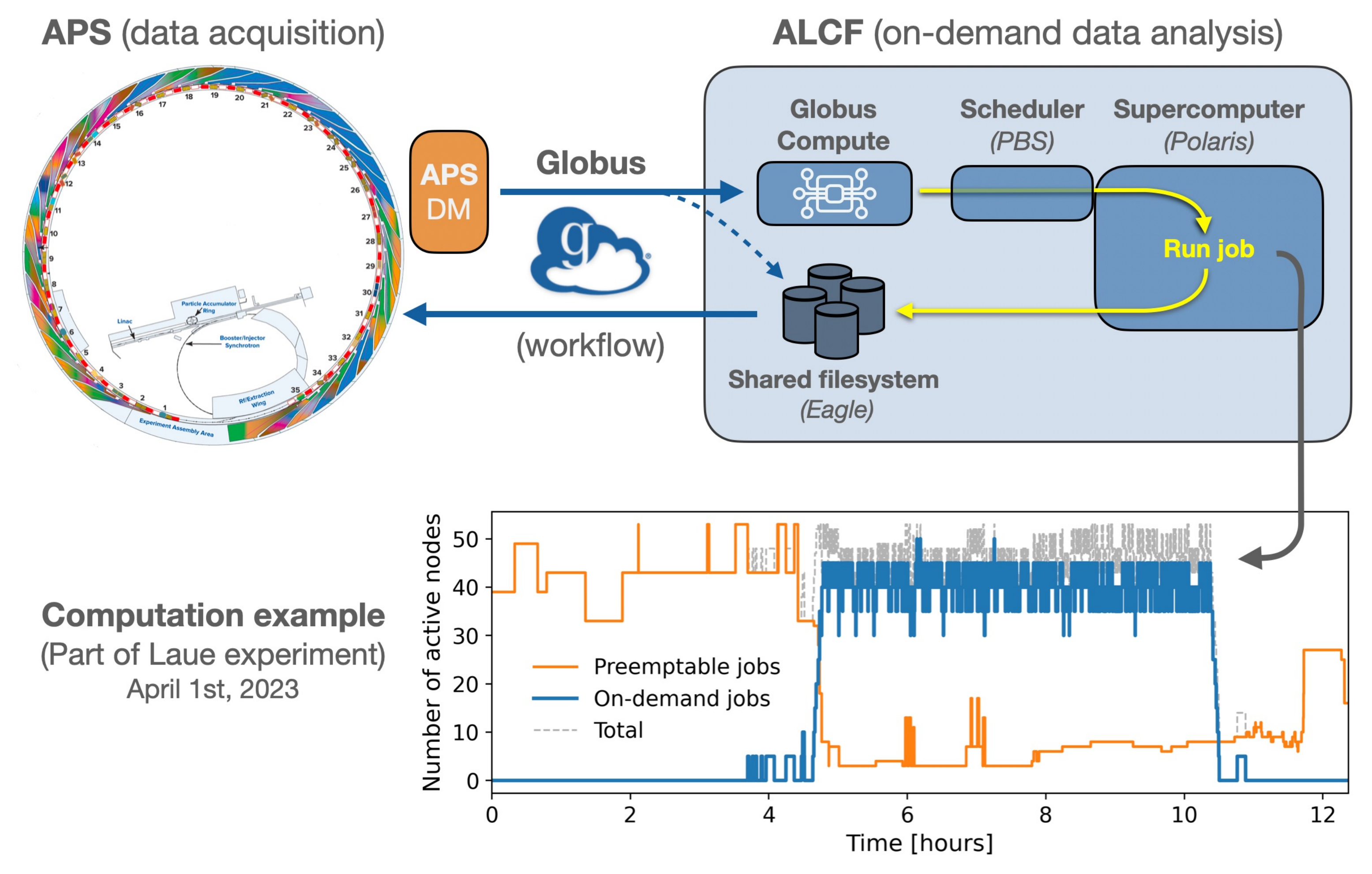
APS | ALCF

APS links experiment to beamline identity

DM system triggers Globus Flow during or following data acquisition

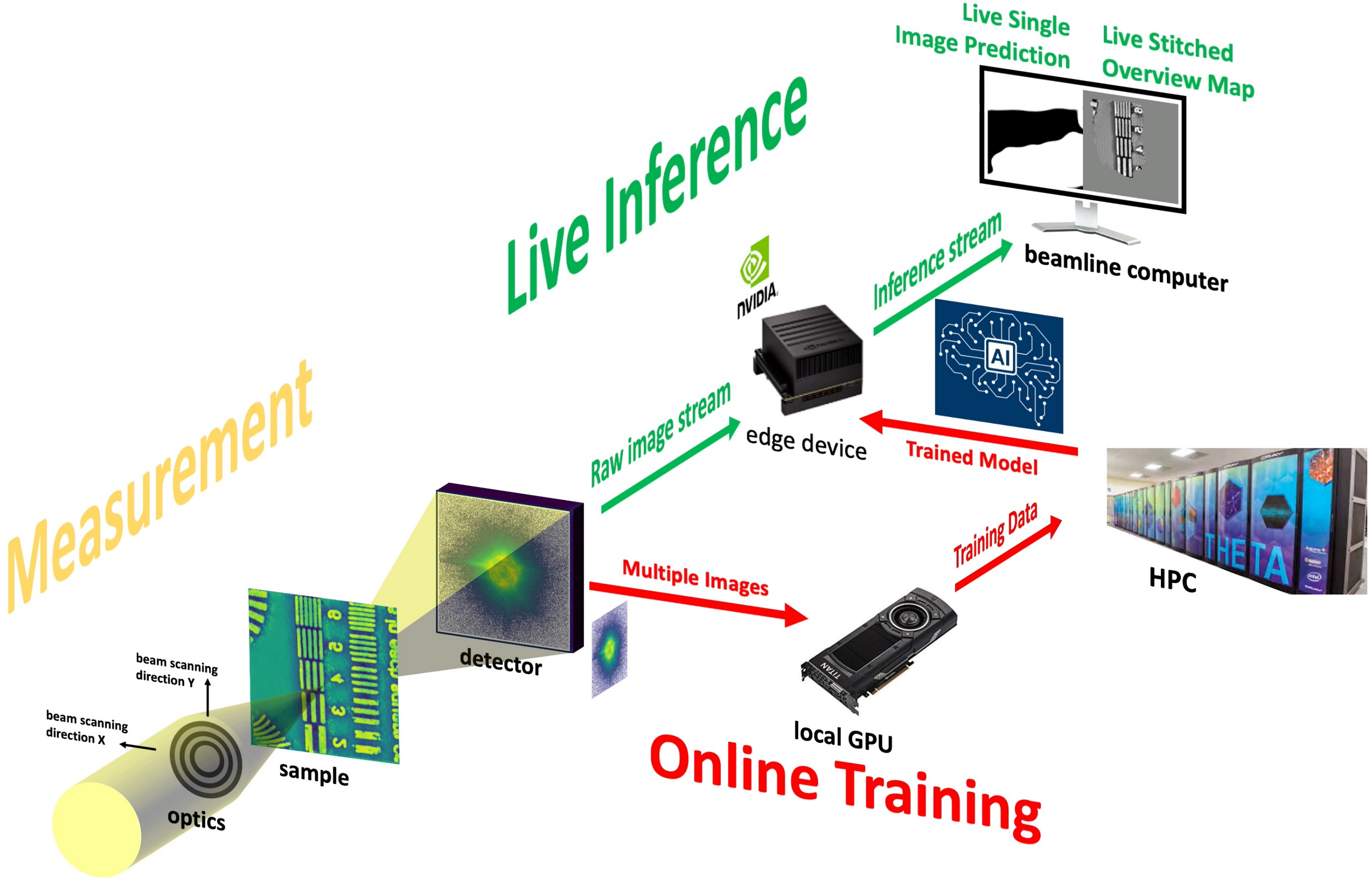
Globus transfers APS data to Eagle

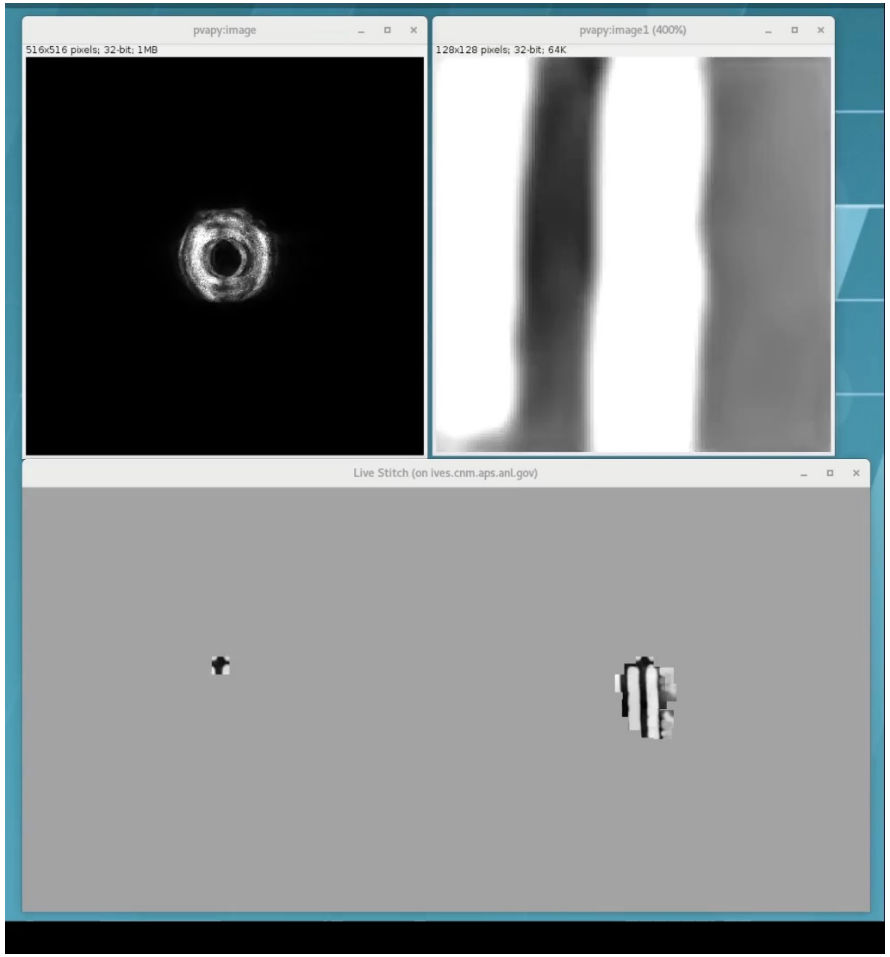
Globus Compute launches analysis on behalf of beamline



AI@EDGE ENABLES REAL-TIME PTYCHOGRAPHY

Train AI @ ALCF, deploy AI @ beamline




pvapy:image
516x516 pixels; 32-bit; 1MB
pvapy:image1 (400%)
128x128 pixels; 32-bit; 64K
Live Stitch (on ives.cnm.aps.anl.gov)

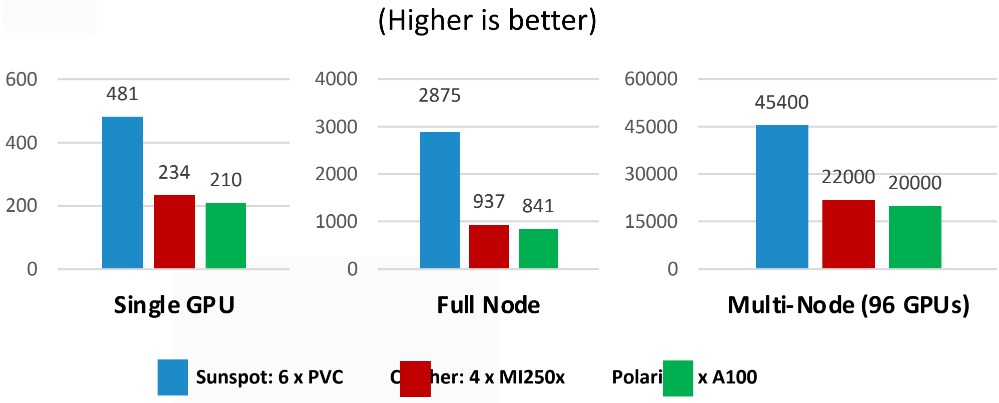
- Real-time imaging: >100X faster than phase retrieval
  - Demonstrated live inference at 100 Hz on 512x512 detector images (1 Gb/s)
- Lower-dose imaging : 25X less data than phase retrieval
- Future work: other techniques, closed-loop experimental steering

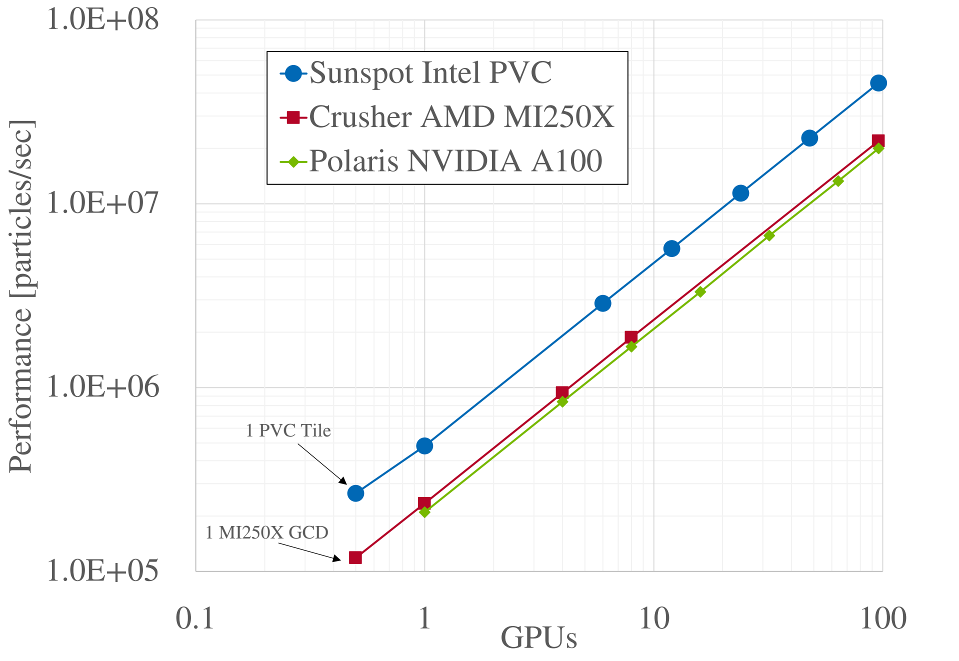
Anakha V. Babu, Tao Zhou, Saugat Kandel, Yi Jiang, Yudong Yao, Sinisa Veselli, Zhengchun Liu, Tekin Bicer, Ekaterina Sirazitdinova, Geetika Gupta, Martin V. Holt, Antonino Miceli and Mathew J. Cherukara, "Real-time nanoscale ptychographic X-ray imaging using deep learning at the edge"

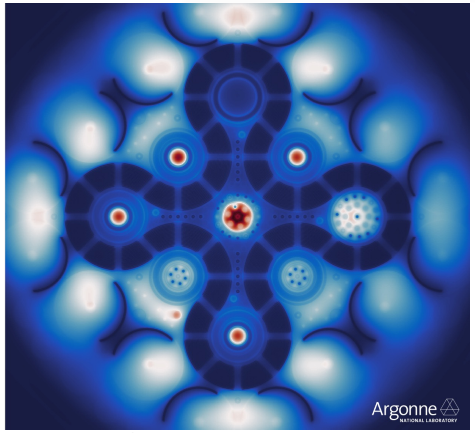
OpenMC *(courtesy of John Tramm)*

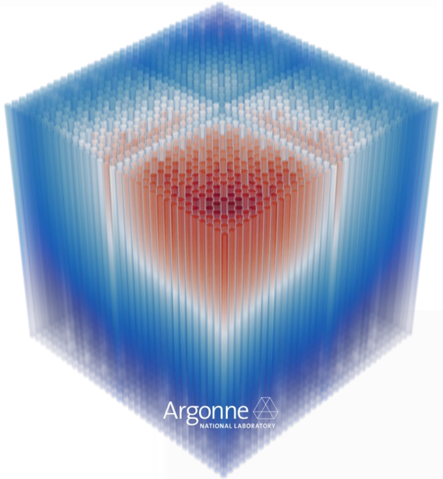
https://docs.openmc.org

- OpenMC is being developed as part of the ECP ExaSMR project (PIs: Steven Hamilton, Paul Romano)
- OpenMC is a Monte Carlo particle transport code written in C++ and the OpenMP target offloading programming model
- The project seeks to accelerate the design of small modular nuclear reactors by generating virtual reactor simulation datasets with high-fidelity, coupled physics models for reactor phenomena that are truly predictive
- The Monte Carlo method employed by OpenMC is considered the "gold standard" for high-fidelity but these methods suffer from a very high computational cost.
- The extreme performance gains OpenMC has achieved on GPUs is finally bringing within reach a much larger class of problems that historically were deemed too expensive to simulate using Monte Carlo methods.





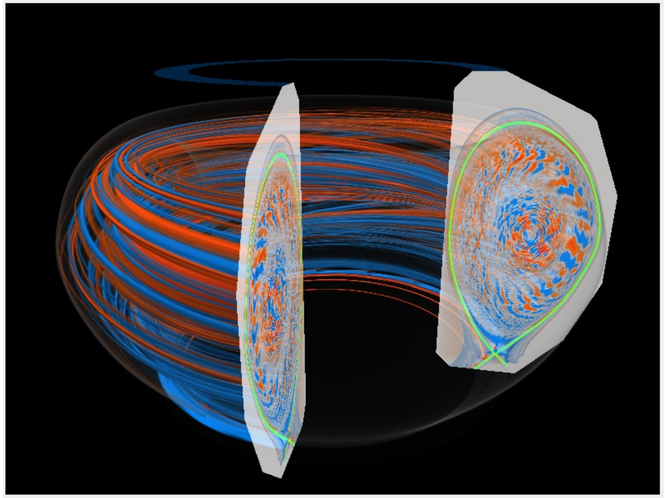


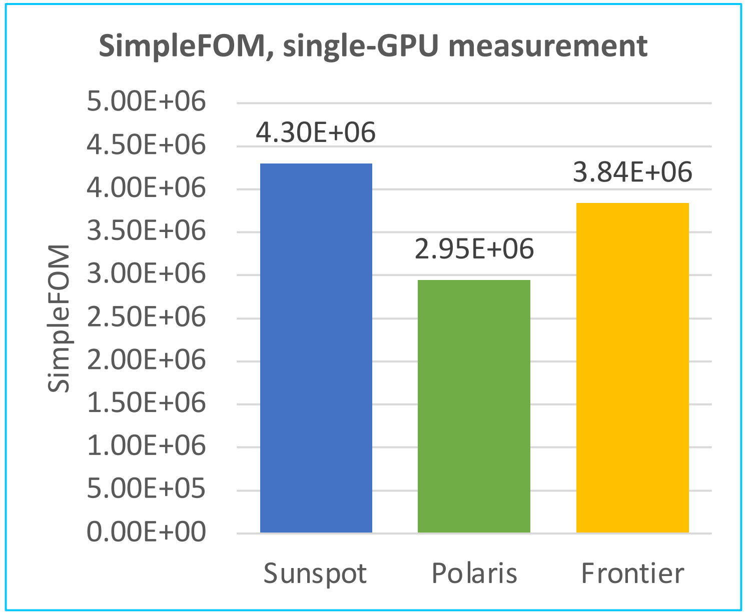


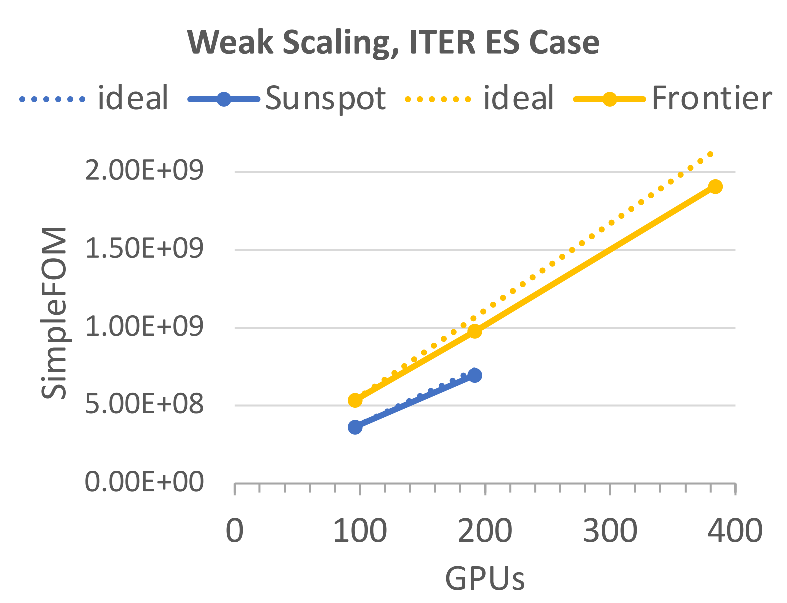
XGC *(courtesy Tim Williams, Aaron Scheinberg)*

ESP Project PI: CS Chang
ECP Project PI: Amitava Bhattacharjee

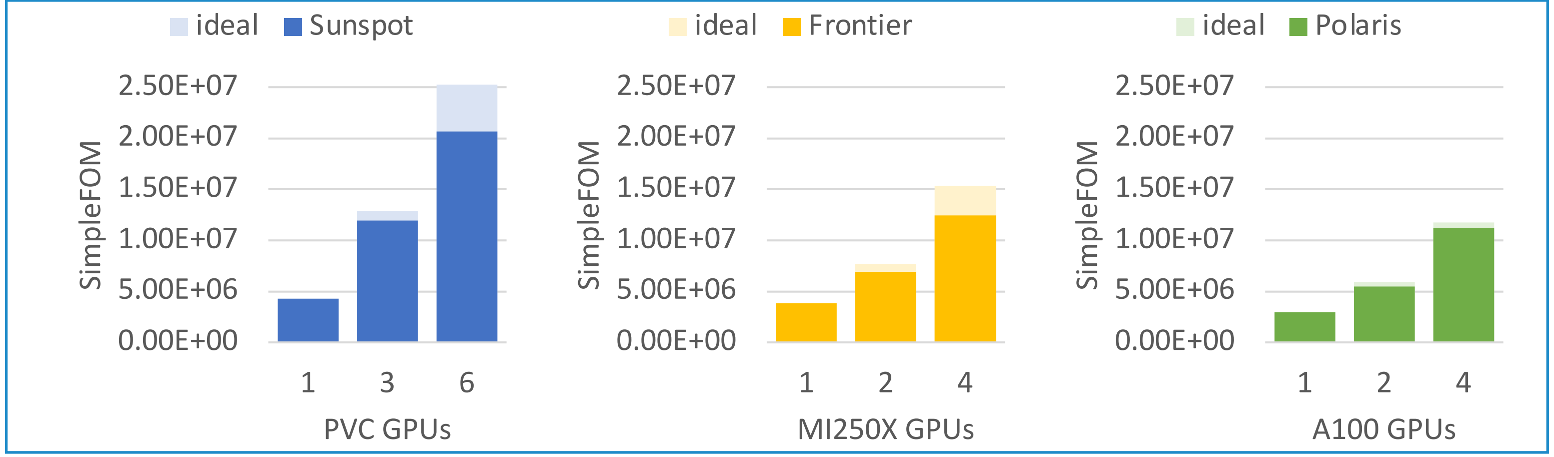
- Science case: Predict ITER fusion reactor plasma behavior with Tungsten impurity ions sputtered from the divertor
- Gyrokinetic particle-in-cell simulation of tokamak plasma using C++ and:
  - —Kokkos/SYCL on Intel GPUs
  - —Kokkos/HIP on AMD GPUs
  - —Kokkos/CUDA on NVIDIA GPUs







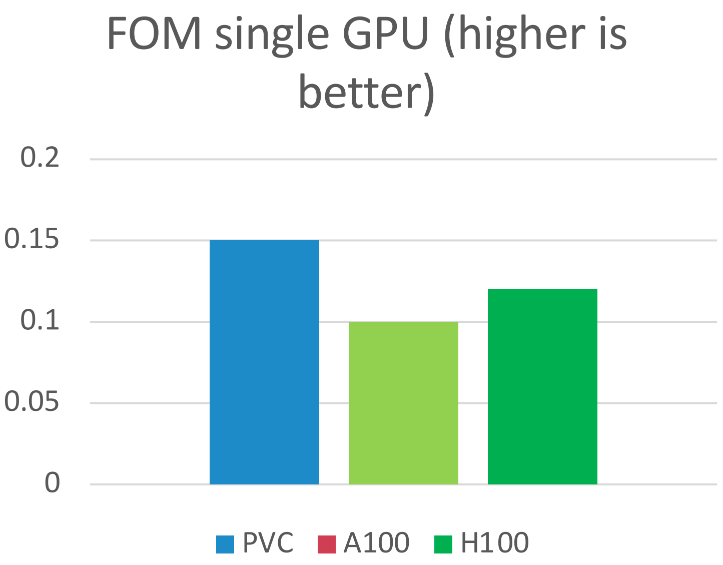
Single-Node Weak Scaling

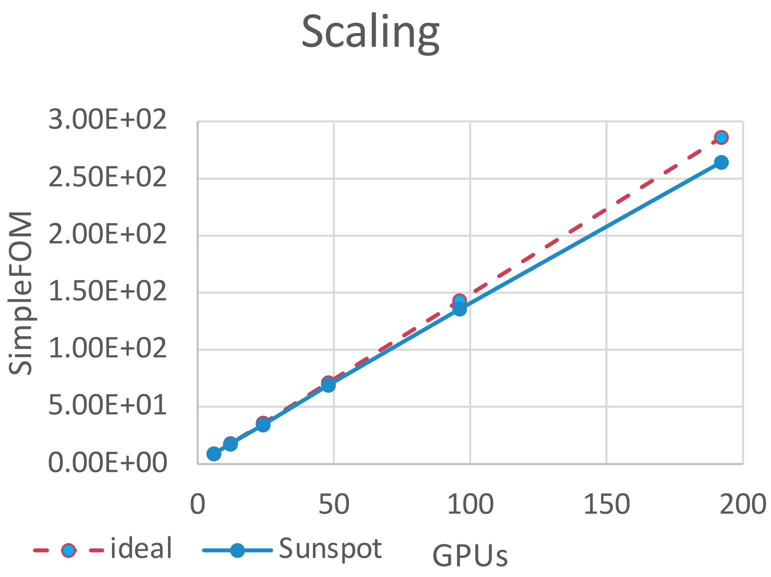


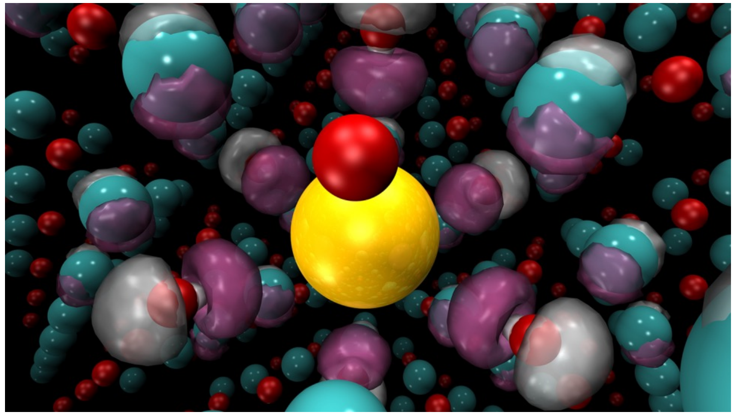
QMCPACK *(courtesy Thomas Applencourt, Ye Luo, Jeongnim Kim)*

ECP Project PI: Paul Kent

- QMCPACK, is a high-performance open-source Quantum Monte Carlo (QMC) simulation code.
- Science case: computing the quantum mechanical properties of materials with benchmark accuracy, including for energy storage and quantum materials.
- QMCPACK uses C++ and OpenMP target offload, plus wrappers (eg SYCL) around vendor optimized linear algebra.







- Running `dmc-a512-e6144-DU64` problem. This simulates a supercell of nickel oxide with 6144 electrons and 512 NiO atoms total.
- Intel® Data Center GPU Max Series: 2 MPI ranks per GPU, 8 Walkers per rank, 64 GB of HBM per stack. Using Intel(R) oneAPI DPC++/C++ Compiler 2022.12.30
- A100 (40GB): 1 MPI Rank, 7 Walkers. LLVM15 compiler.
  H100: llvm/clang 17, cuda 11.8): 1 MPI Rank, 7 Walkers
- The Figure Of Merit (FOM) measure is throughput (walker moves/second). Higher is better.

Aurora genAI

- This is a forward looking project. It is a vision and some proof of concept hackathons. Think ChatGPT but directly targeted at science.
- While Aurora probably has the computational power required, I believe the data aspect of this is a harder nut to crack. Since I will be the one responsible for enabling that, that worries me ☺.
- On the other hand, imagine the possibilities if we could pull this off.
- At the International Supercomputing Conference (ISC) which is going on right now in Germany, this was announced and there were some real examples given. Now this next part if just my imagination so hold on for a wild ride ☺.
    - There is another effort on-going at the lab called ”the self-driving lab”. It involves lab automation, robotics, AI, etc. So how far do we have to jump to go from asking it for what it already knows to asking it to run the computations or lab experiments to do the research and find the answer for us?
    - ”Provide me every protein structure study for <the latest pandemic virus>(TLPV)” – a really good google
    - ”Given the results above, generate code for docking trials to find possible treatments for TLPV” – taking what you can do today and REALLY stretching it. Although the odds of it generating code that complex that works are nil today.
    - Now I am going to start making stuff up:
        - “Execute the code generated that I approved using every available resource we have access to, optimizing for time to first result”
        - ”As results come in from the above, send them to me for review and for the ones I approve, run experiments in the automated lab for targets we have, generate POs to purchase existing targets we don’t own for my approval, and for targets that don’t exists, suggest ways they might be synthesized”
        - “Cure TLPV” – or we could just keep it simple ☺

How have we built these?
How might that need to change?

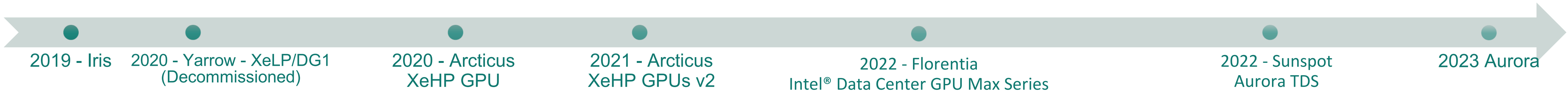
First, we have to show there is a need

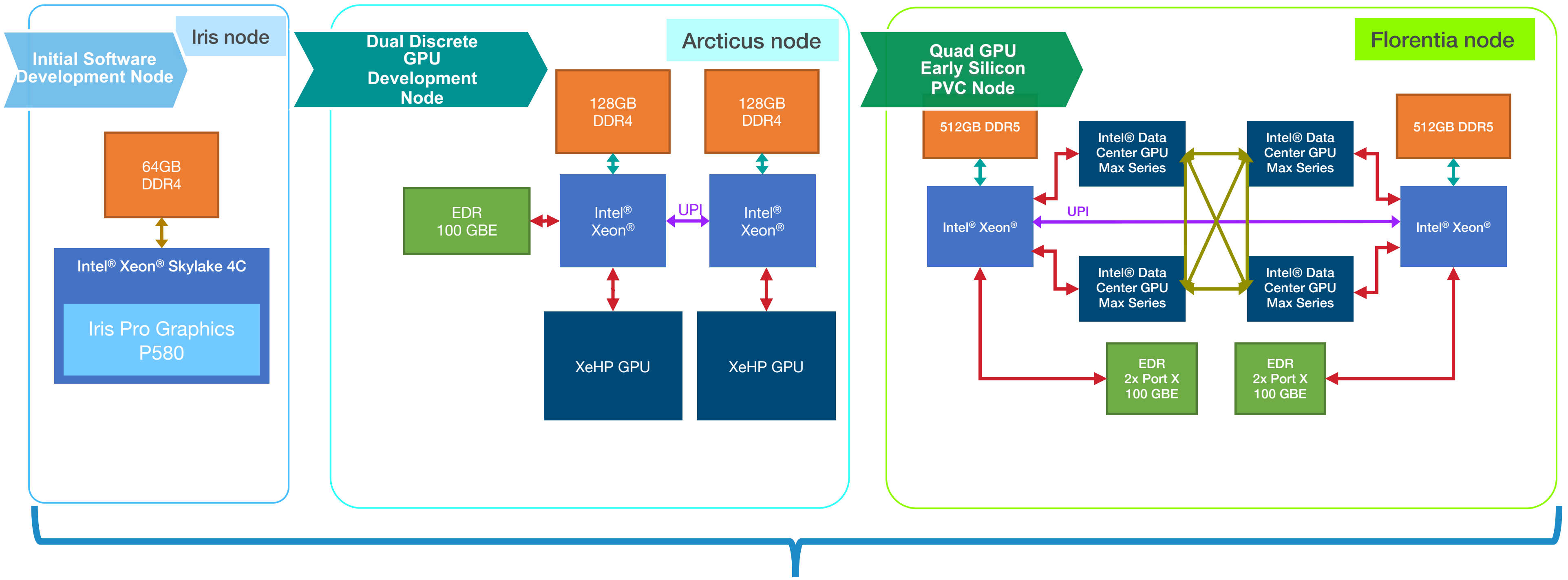
- That is one thing that isn’t going to change.
- We don’t get to spend 10’s or 100’s of millions of dollars because we want bigger toys
- This is also NEVER a problem; There are “holy grail” problems that we are not even close to:
  - —A full multi-physics simulation of an entire nuclear reactor
  - —A complete model of every neuron in the human brain
- This is referred to as “Critical Decision Zero” or CD-0
- The rules for DOE Capital projects are referred to as DOE Order 413.3B
- This link Has a nice overview with links to more details than you could ever want (unless you have insomnia ☺). The REALLY short version:
  - —Is there a need?
  - —Analyze the alternatives, make a selection, determine cost ranges
  - —Get the Performance Baseline approved; This is project performance, which includes the system performance, but is much more.
  - —Approval to start “construction” (spending money)
  - —Project completion, which in our case basically means we are ready to go into production.
- The point being: we are aware it is a lot of money. We take that seriously, and so does an entire PM department at DOE, so a LOT of people look at this before we start to spend that kind of money and review the project annually during its execution.

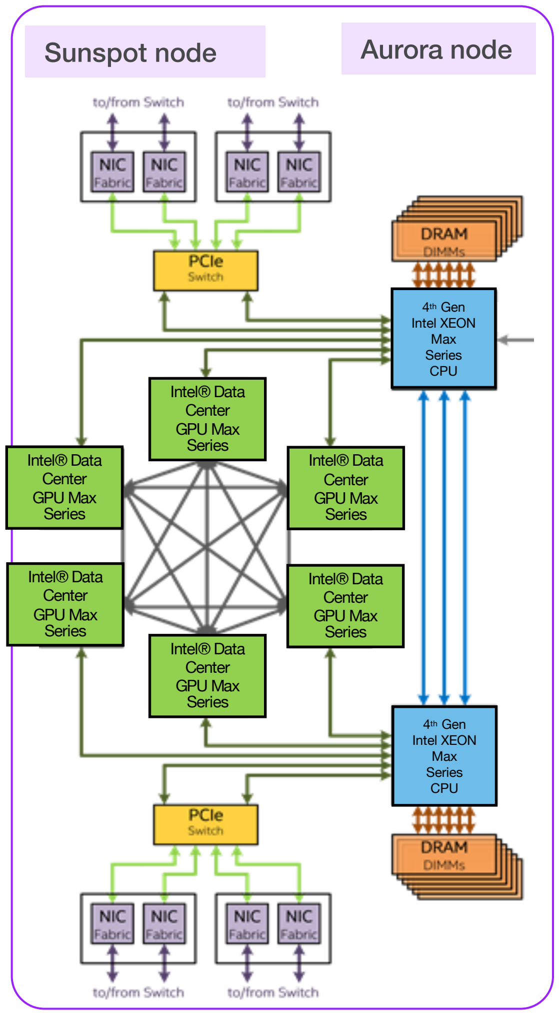
Hey, I have this really cool idea… Trust me

- We sign contracts for processors that are nothing more that designs on paper. That is pretty darn scary. And it carries some risk.
- Trust, but verify
  - —Paper calculations
  - —Cycle accurate simulators
  - —Various incarnations of the HW and software that come closer and closer to the final product
- And you also need software to run on it, day one:
  - —We put significant effort into what we call our early science program. We select science projects that sign NDAs and get “read in” early and follow along with the HW and SW development porting their codes, testing performance, providing feedback on the usability and performance. This is a huge undertaking, and vitally important to the success of the project.
  - —We pick the projects based on a number of factors, but two of the really important ones are:
    - the team: do they have the computing expertise, the interest, the will to invest years of time into this porting to a new architecture.
    - The characteristics of the code: We want a selection that exercises all aspects of the system design. We want computationally constrained codes, memory bandwidth constrained codes, communication bandwidth or latency constrained, etc.

Aurora Testbeds: The Path to the Aurora Hardware


2019 - Iris
2020 - Yarrow - XeLP/DG1 (Decommissioned)
2020 - Arcticus XeHP GPU
2021 - Arcticus XeHP GPUs v2
2022 - Florentia Intel® Data Center GPU Max Series
2022 - Sunspot Aurora TDS
2023 Aurora




Sunspot node
Aurora node
to/from Switch
to/from Switch
NIC Fabric
NIC Fabric
NIC Fabric
NIC Fabric
PCIe Switch
DRAM DIMMs
4th Gen Intel XEON Max Series CPU
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
Intel® Data Center GPU Max Series
4th Gen Intel XEON Max Series CPU
PCIe Switch
DRAM DIMMs
NIC Fabric
NIC Fabric
NIC Fabric
NIC Fabric
to/from Switch
to/from Switch

Aurora Testbeds for Applications Development and Software Testing

- **Sunspot is a two rack test and development system**
- **Sunspot nodes and cabinets have the same hardware as Aurora**
- **Sunspot is fully built and available for application developer use**

The good, the bad, and the ugly

- A CS researcher named Phil Colella looked at HPC codes and came to the conclusion that there were basically 7 classes of computation that made up all HPC. A few years later, 6 more classes or “kernels” were added. The point of this is, if you spend your time writing really high quality libraries that implement those kernels, and focus your attention making them perform on new architectures, you have gone a long way towards getting performance on the different architectures.
  - https://www2.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf
- However, there is also the issue of how you get at these things on different systems. CUDA on Nvidia, different flavors of MPI, etc.. Performance portability has become a real focal point over the last few years.
  - Intel’s OneAPI is an attempt to have one universal code that will run at maximum efficiency regardless of the architecture.
  - Earlier I talked about one of the science codes using Kokkos and its ability to plug in different backends to hide the architectural differences from the application level.
  - We still have a long way to go, but we are making progress.
- AI accelerators are only going to make this worse. There we encourage them to enable the more standard tools on their platforms like pytorch or tensorflow as opposed to developing their own or at least only having their own.
- We had 50ish years of nirvana riding Dennard scaling (Moore’s Law). Software had to change very little to get the performance benefits. Those days are over, at least in our community.

So what are some of things we want to change?

- Reduce the level of effort to respond to the RFP
  - In the past, only the largest companies (IBM, HPE, Cray wasn't big, but HPC was their only business) could afford to bid because our requirements were so onerous.
  - Besides being expensive, time consuming, and annoying for everyone involved, it potentially blocked smaller innovative companies from bidding because they couldn't afford a compliant bid.
  - We want to reduce the number of mandatory requirements to the bare minimum and make most things targets or options of weighted importance so that a smaller company that has good IP in a heavily weighted area can afford to bid.
- Be more agile in our procurements
  - In the past, we turned the crank every 5 years, give or take and installed a completely new system from scratch. Because of the contractual issues noted above, we would be locked into optimizing source code that had long ago ceased being interesting, but it was what was going to be used to decide success or failure by contract.
  - We want to look at how we can reuse our infrastructure. We want to be able to mix and match hardware and software; do "rolling upgrades" to more quickly integrate product improvements from small AI vendors or try out new HW vendors.
  - The downside to this is, you become the victim of the least common denominator. The systems we design have custom power systems to maximize density. What make a supercomputer super is the interconnect, but all the AI vendors are running Ethernet or at best Infiniband, so we will likely lose parallel efficiency. Various system incompatibilities become a concern when you have a highly heterogenous system.

A couple of other points

- We have never been happy with the system software stack in HPC
  - Clearly the hyperscalers can handle way more nodes than we will ever have and all the recent college graduates are trained in their tech, however there are certain base assumptions that don’t hold in HPC and that makes some of their paradigms problematic.
    - “Let it crash” is great when you have 100s of thousands of identical commodity nodes running billions of single node unrelated jobs in data centers all over the world. It is less clear how well that works for us.
    - It brings in a lot of complexity that is absolutely necessary when managing 100s of thousands of nodes in data centers worldwide, but may be unnecessarily complex for our needs.
    - I lean towards taking what they have and seeing if we can’t modularize it or add a plugin architecture or similar that we can use to replace and simplify some functionality.
    - However if all the DOE labs can’t come an agreement on an approach, if not exactly the same stack, we won’t make progress because none of us have enough resources to do it on our own.
- We are always buying untested systems.
  - No one can afford to build a full scale machine in their lab
  - We are solving problems, finishing/debugging software, doing research as we bring the machines up because there is very little other choice.
  - Perhaps some subsystems could be tested via simulation, but that only goes so far.
  - We would love to change this, but it is not clear how…

Software is a major concern

- The software folks (I am one of them) didn't realize what a free ride they were getting from Moore's Law. The speed of the machine doubled, but how you needed to interact with it basically didn't change.
- Now with accelerators and power being such an issue, we need to be rethinking algorithms from the ground up. We will get it done, but it will take a lot of time, effort, and money to get there.
  - When processor manufacturers first put caches on the processors, people disabled them because they *hurt* the performance. Algorithms were not written to be cache friendly. No one would do that today.
  - Moving data is expensive from a power perspective.
    - There is HW work, like trying to put ALUs on the DIMMs, but you still need SW that can use it.
    - Algorithms focused on being able to do all processing on a piece of data once while it is in memory
  - As mentioned previously, how do you hide all the differences between the different accelerators?
  - How do manage data movement between the accelerator and RAM
    - Although coherency across the devices is becoming common
- FORTRAN is a huge issue in our community
  - There is a lot of it and for parallel computing the language has distinct advantages. However, new college graduates don't know it, vendors don't want to support it, the compilers are lagging way behind, etc..
  - Do we port millions of lines of code, or do we basically take on the support of FORTRAN for ourselves?
- software sustainability is a big concern
  - You can write grants to develop new software / features, but you will never get funded for a grant that says "ongoing software maintenance for that code you funded me to write and now everyone is using"
  - We all know it is a problem and there is a significant effort on how to address it. This is policy as much as technical

I hope you found this interesting and useful

Questions?