Reflections on the Exascale Era: Then and Now

EXASCALE COMPUTING PROJECT

HPC User Forum

September 6, 2023

Tucson, AZ

Douglas B. Kothe, Sandia National Laboratories
Chief Research Officer
Associate Labs Director, Advanced Science & Technology
dbkothe@sandia.gov

NNSA National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

Recent Leadership Changes: ECP is in Good Hands

Charged with taking ECP across the finish line (formal Project Completion)

- ECP Director: Lori Diachin (LLNL)
- ECP Deputy Director: Ashley Barker (ORNL)
- ECP Hardware & Integration Focus Area Director: Richard Gerber (LBNL)
- Me (Doug Kothe): Immersed full time in the Exascale Computing Initiative (ECI) since Apr 1, 2015. Agreed to take on current position at Sandia Jun 5, 2023. Almost made it. Mixed feelings that I did not. ☺
- Kudos to Lori Diachin in particular for her visionary leadership, passion, commitment, and energy. Leading ECP is not an easy gig and invariably a contact sport. She is driving ECP across the finish line.

ECP Reflections: Some Observations and Lessons Learned

And there are many more . . . lots of scare tissue here that could consume a few books. ☺

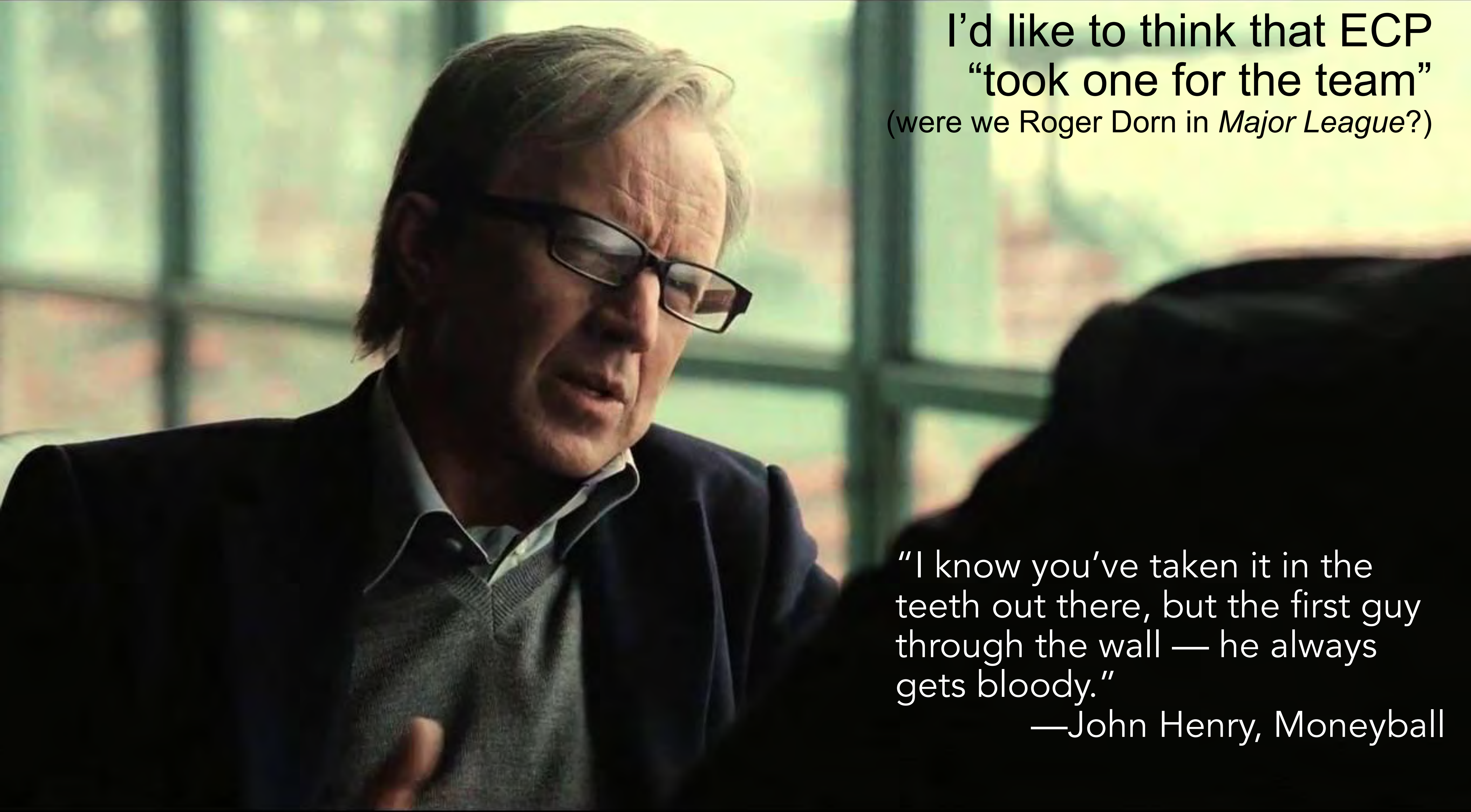
- Projectizing R&D works if agile PM & aggressive change control is in place
- S/W investment must be 1st class citizen along with H/W
- Upstream R&D investment at low H/W tech TRL crucial
- Highly functioning diverse leadership team are a must
- Take calculated risks with appropriate and understood mitigations
- Empower the leadership team then hold them accountable
- Put overachieving, field-leading competitive PIs together and they ”one-up” each other to death

- Open, frequent comms (good/bad/ugly) imperative with sponsors, stakeholders, staff
- Sponsor confidence in leadership team a must
- Build integration into project structure and operations
- You improve what you measure so be careful what is measured
- Good centralized PM tools do not guarantee success but bad ones can sure impede progress
- Understand and manage external dependencies
- Small diverse talented teams can do a *lot* if left undistracted

- Formalize and document institutional commitments
- Discoveries cannot be planned but stable longer-lived support that focus R&D teams on a single challenge virtually guarantees them
- Design and quality reqms are not known - must be iterated on
- Use external advisory and SME bodies to inform leadership
- Avoid top down mandating of technology solutions
- Don’t underestimate the importance of staff training & education and staff diversity

EXASCALE COMPUTING PROJECT

I’d like to think that ECP
“took one for the team”
(were we Roger Dorn in *Major League*?)

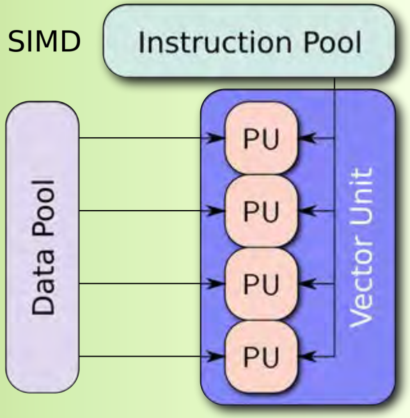


“I know you’ve taken it in the teeth out there, but the first guy through the wall — he always gets bloody.”
—John Henry, Moneyball

HPC: Change is the only constant

Vector Era
MFLOP/s - GFLOP/s

- Parallelism through vector processors.
- Codes often written at very low level to make optimal use of hardware.



Distributed Memory Era
GFLOP/s - TFLOP/s – PFLOP/s

- Parallelism through MPI.
- Using an optimal parallel algorithm was critical to avoid duplication of work or unnecessary communication.
- Once distributed, code could be treated serially.

- For the most part, an MPI code ran anywhere. For best performance, key kernels could be tuned.
- As CPU frequencies stopped increasing, parallelism became more extreme and specialized hardware more common.

10s to 100s of cores — *1000s of cores* — *10^4 to 10^6 cores*

1980s — 1990s — 2000s — 2010s

HPC: Change is the only constant

What's next? 8-bit Zettascale before 2030? LLMs writing all our code ready to verify?

Heterogeneous Era
PFLOP/s - EFLOP/s

- CPUs + accelerators with separate memory spaces to start, unclear what else will join the fray.
- Massive fine-grained parallelism required.
- Programming model has to match the architecture.
- Architectural landscape is changing rapidly, with an unclear future.

2010s 2020s

Heterogeneity is the new reality

- Computational horsepower has significantly outpaced memory capacity and speed.
- Separate memory spaces add complexity, and can cause performance issues (e.g. NUMA) or errors if not handled correctly.
- Performance or portability?
- Refactoring an existing code is a lot of work! You <u>really</u> don't want to have to do it again in ten years.

Frontier enables science today

Frontier is the world's fastest supercomputer and the world’s first supercomputer to break the performance barrier known as exascale, debuting in May 2022 at 1.1 exaflops.

Compute Node
1 AMD EPYC CPU
4 AMD MI250X GPUs

System Size
>9,000 nodes

Memory
4.6 PB DDR4
4.6 PB HBM2e
36 PB on-node storage

On-node Interconnect
AMD Infinity fabric
Node-level coherence

System Interconnect
Four-port Slingshot network
100 GB/s

EXASCALE COMPUTING PROJECT

1.1 EXAFLOPS

FRONTIER CAN DO MORE THAN **1 QUINTILLION** CALCULATIONS PER SECOND.

1 SECOND

IF EACH PERSON ON EARTH COMPLETED **ONE CALCULATION PER SECOND,** IT WOULD TAKE MORE THAN **4 YEARS** TO DO WHAT AN EXASCALE COMPUTER CAN DO IN **1 SECOND.**

6,000 GALLONS

OF **WATER** IS MOVED THROUGH THE SYSTEM **PER MINUTE** BY FOUR **350-HORSEPOWER PUMPS.** THESE POWERFUL PUMPS COULD FILL AN **OLYMPIC-SIZED SWIMMING POOL** IN ABOUT **30 MINUTES.**

700 PETABYTES

FRONTIER’S ORION STORAGE SYSTEM HOLDS **33 TIMES** THE AMOUNT OF DATA HOUSED IN **THE LIBRARY OF CONGRESS.**

8,000 POUNDS

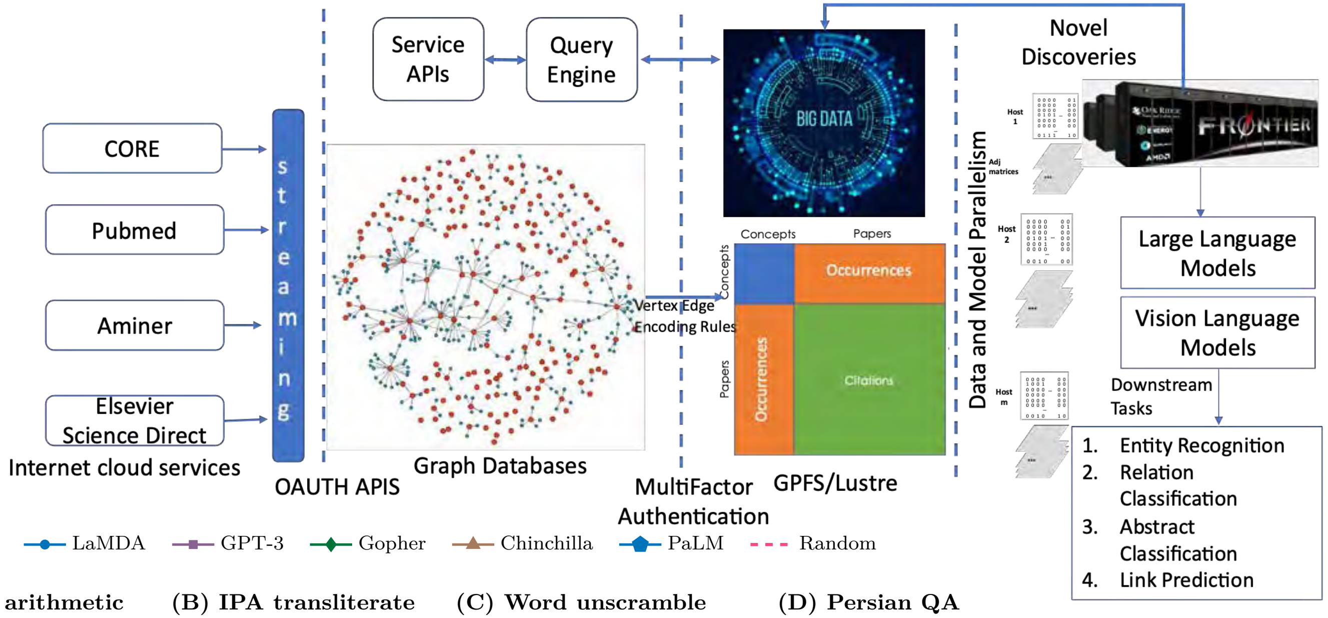
EACH CABINET WEIGHS THE EQUIVALENT OF **2 FULL-SIZE PICKUP TRUCKS.**

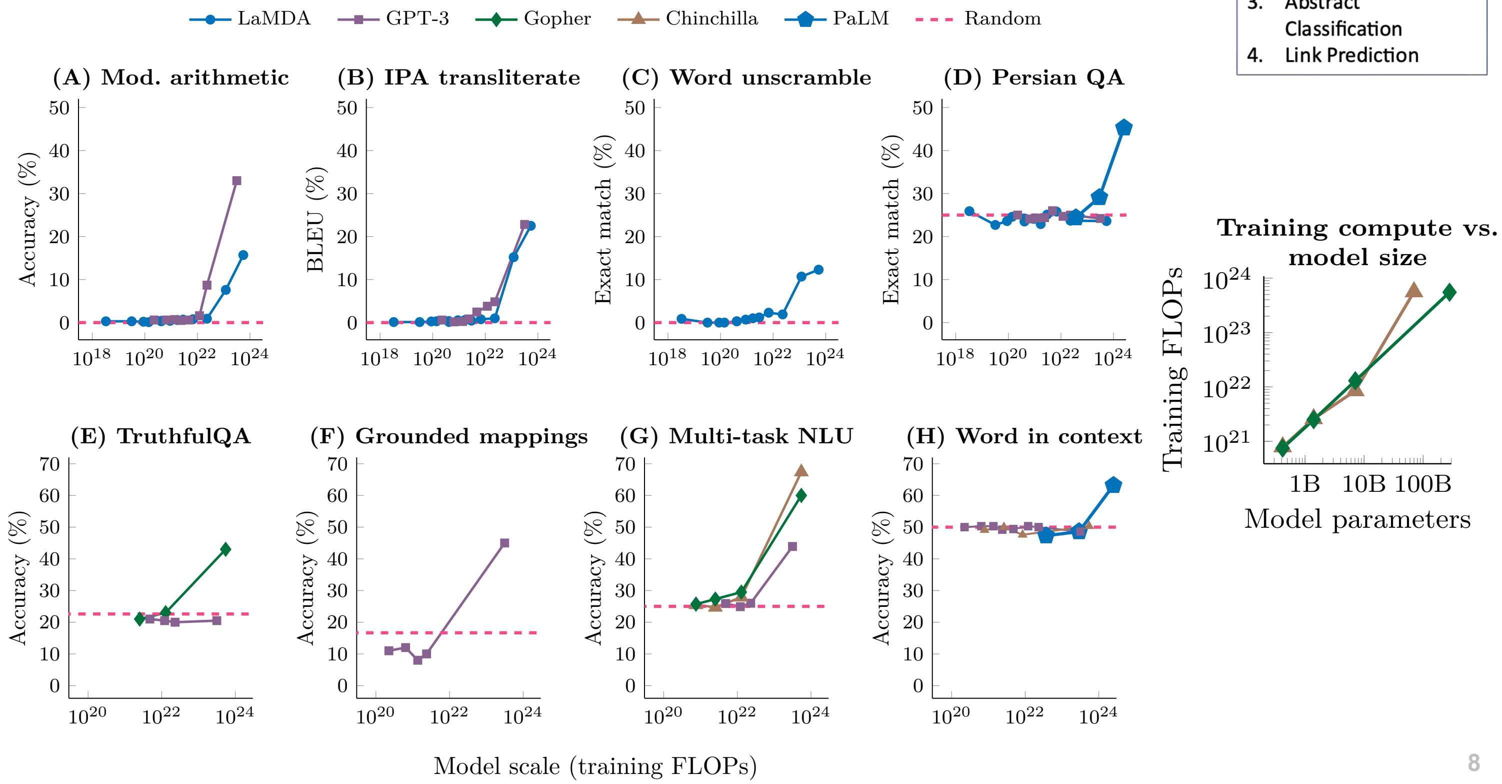
40 MEGAWATTS

FRONTIER’S MECHANICAL PLANT CAN COOL THE EQUIVALENT POWER DEMAND OF ABOUT **30,000 U.S. HOMES.**

Can Frontier Train the Largest AI Models (>10^{14} Parameters)?

- We are in the quest of demonstrating the HPC needs for training real world scientific AI problems – specifically scientific text and images.
- Pre-train large language models (LLM) such as GPT-3, BLOOM, PALM, LaMDA, Gopher and Vision Language models on scientific texts like Pubmed, Aminer, MAG and materials related publication texts
- Frontier
  - We believe we train up to 150 Trillion FP32 Parameter model in Frontier. This is approximately ~300X bigger than the largest PaLM model with 540B parameters.
  - Training some of these off the shelf large language models could at least take 12 days on Frontier at HPL parallel performance efficiency





Wei, Jason, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama et al. "Emergent abilities of large language models." *arXiv preprint arXiv:2206.07682* (2022).

EXASCALE COMPUTING PROJECT

AI for Science

What Comes After Exascale

- Over 1,300 scientists participated in 4 town halls during the summer/fall of 2019
- Research opportunities in AI
  - Biology, chemistry, materials,
  - Climate, physics, energy, cosmology
  - Mathematics and foundations
  - Data life cycle
  - Software infrastructure
  - Hardware for AI
  - Integration with scientific facilities
- Modeled after the Exascale Series in 2007
- ASCAC subcommittee report Sept. 2020


ANL-22/91

ADVANCED RESEARCH DIRECTIONS ON
AI FOR SCIENCE, ENERGY, AND SECURITY

Report on Summer 2022 Workshops

Jonathan Carter
Lawrence Berkeley National Laboratory

John Feddema
Sandia National Laboratories

Doug Kothe
Oak Ridge National Laboratory

Rob Neely
Lawrence Livermore National Laboratory

Jason Pruet
Los Alamos National Laboratory

Rick Stevens
Argonne National Laboratory

U.S. DEPARTMENT OF ENERGY | U.S. DEPARTMENT OF ENERGY Office of Science

January 2023

Leadership AI aimed at mission needs

Scientific discovery, user facilities, energy research, environment and national security

Leverages relevant DOE assets

- Exascale class computing
- Exascale class data infrastructure
- Large-scale Experimental Facilities
- Large-scale Scientific Simulation Capabilities
- Interdisciplinary teams

AI for Advanced Properties Inference and Inverse Design
Energy Storage
Proteins, Polymers

AI and Robotics for Autonomous Discovery
Materials, Chemistry, Biology
Light-Sources, Neutrons, ..

AI Based Surrogates for HPC
Climate Ensembles
Effective Zettascale on Exa

AI for Programming and Software Engineering
Code Translation, Optimization
Quantum Compilation, QAlgs

AI for Prediction and Control of Complex Engineered Systems
Accelerators, Buildings, Cities
Reactors, Power Grid, Networks

Foundation AI for Scientific Knowledge
Hypothesis Formation, Math
Theory and Modeling Synthesis

Aurora is building out . . .

Argonne's upcoming exascale supercomputer will leverage several technological innovations to support machine learning and data science workloads alongside traditional modeling and simulation runs.

PEAK PERFORMANCE
≥2 Exaflop DP

Intel® X^e ARCHITECTURE-BASED GPU
Data Center GPU Max Series

INTEL® XEON® SCALABLE PROCESSOR
Intel Xeon CPU Max Series

PLATFORM
HPE Cray EX

Compute Node
2 Intel® Xeon® CPU Max Series processors; 6 Intel® Data Center GPU Max Series GPUs; Unified Memory Architecture; 8 fabric endpoints; RAMBO

GPU Architecture
Intel® Data Center GPU Max Series; Tile-based chiplets, HBM stack, Foveros 3D integration, 7nm

CPU-GPU Interconnect
CPU-GPU: PCIe
GPU-GPU: X^e Link

System Interconnect
HPE Slingshot; Dragonfly topology with adaptive routing

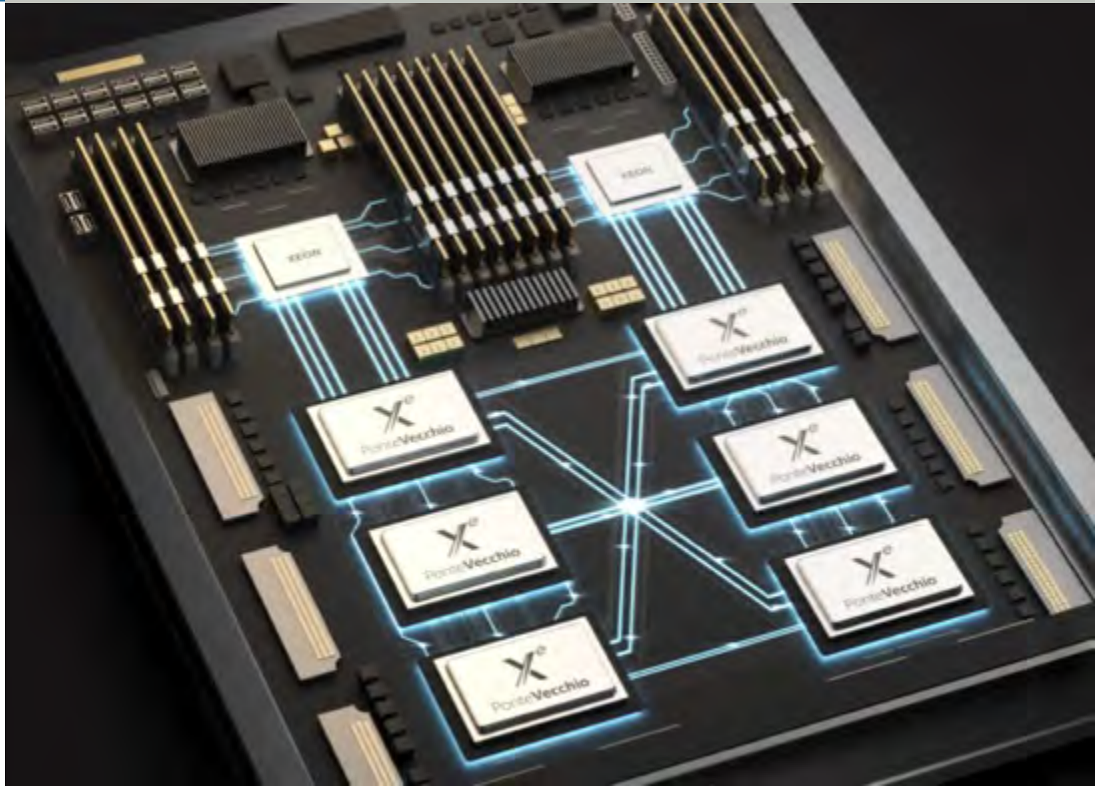
Network Switch
25.6 Tb/s per switch, from 64–200 Gbs ports (25 GB/s per direction)

High-Performance Storage
≥230 PB, ≥25 TB/s (DAOS)

Programming Models
Intel oneAPI, MPI, OpenMP, C/C++, Fortran, SYCL/DPC++

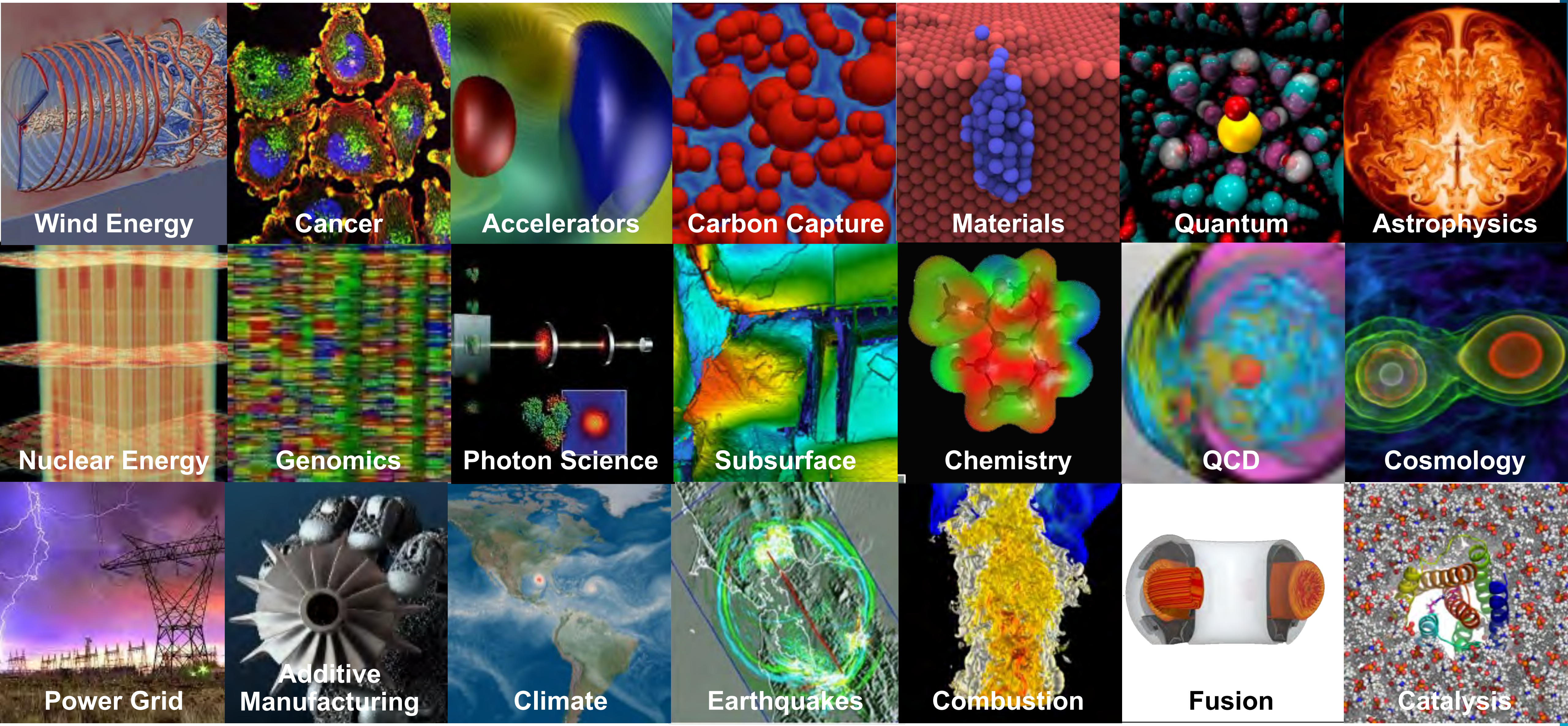
Node Performance
>130 TF

System Size
>10,000 nodes



ECP Took on a Diverse and Somewhat Risky Application Portfolio

Some of these apps were little more than “half baked prototypes” in 2016 . . .



But we required ECP Apps to have a specific Challenge Problem

Focuses development, allows measurable outcomes, facilitates scope / de-scope decisions

Domain*	Base Challenge Problem	Risks and Challenges
Wind Energy	2x2 5 MW turbine array in 3x3x1 km^3 domain	Linear solvers; structured / unstructured overset meshes
Nuclear Energy	Small Modular Reactor with complete in-vessel coolant loop	Coupled CFD + Monte Carlo neutronics; MC on GPUs
Fossil Energy	Burn fossil fuels cleanly with CLRs	AMR + EB + DEM + multiphase incompressible CFD
Combustion	Reactivity controlled compression ignition	AMR + EB + CFD + LES/DNS + reactive chemistry
Accelerator Design	TeV-class 10^{2-3} times cheaper & smaller	AMR on Maxwell's equations + FFT linear solvers + PIC
Magnetic Fusion	Coupled gyrokinetics for ITER in H-mode	Coupled continuum delta-F + stochastic full-F gyrokinetics
Nuclear Physics: QCD	Use correct light quark masses for first principles light nuclei properties	Critical slowing down; strong scaling performance of MG-preconditioned Krylov solvers
Chemistry: GAMESS	Heterogeneous catalysis: MSN reactions	HF + MP2 + coupled cluster (CC) + fragmentation methods
Chemistry: NWChemEx	Catalytic conversion of biomass	CCSD(T) + energy gradients
Extreme Materials	Microstructure evolution in nuclear matls	AMD via replica dynamics; OTF quantum-based potentials
Additive Manufacturing	Born-qualified 3D printed metal alloys	Coupled micro + meso + continuum; linear solvers

*Required to demonstrate a capability *and* performance metric
*Required to demonstrate a capability metric

But we required ECP Apps to have a specific Challenge Problem

Focuses development, allows measurable outcomes, facilitates scope / de-scope decisions

Domain*	Challenge Problem	Computational Hurdles
Quantum Materials	Predict & control matls @ quantum level	Parallel on-node perf of Markov-chain Monte Carlo; OpenMP
Astrophysics	Supernovae explosions, neutron star mergers	AMR + nucleosynthesis + GR + neutrino transport
Cosmology	Extract "dark sector" physics from upcoming cosmological surveys	AMR or particles (PIC & SPH); subgrid model accuracy; in-situ data analytics
Earthquakes	Regional hazard and risk assessment	Seismic wave propagation coupled to structural mechanics
Geoscience	Well-scale fracture propagation in wellbore cement due to attack of CO_2-saturated fluid	Coupled AMR flow + transport + reactions to Lagrangian mechanics and fracture
Earth System	Assess regional impacts of climate change on the water cycle @ 5 SYPD	Viability of Multiscale Modeling Framework (MMF) approach for cloud-resolving model; GPU port of radiation and ocean
Power Grid	Large-scale planning under uncertainty; underfrequency response	Parallel nonlinear optimization based on discrete algebraic equations; multi-period optimization
Cancer Research	Scalable machine learning for predictive preclinical models and targeted therapy	Increasing accelerator utilization for model search; exploiting reduced/mixed precision; resolving data management or communication bottlenecks
Metagenomics	Discover and characterize microbial communities through genomic and proteomic analysis	Graph algorithms, distributed hashing, matrix operations and other discrete algorithms
FEL Light Source	Protein and molecular structure determination using streaming light source data	Parallel structure determination for ray tracing and single-particle imaging

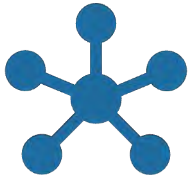
*Required to demonstrate a capability *and* performance metric
*Required to demonstrate a capability metric

Four Key Ingredients of an ECP Application Development Project

Science goal

Algorithmic innovation

Porting



Integration

Four Key Ingredients of an ECP Application Development Project

Science goal	Algorithmic innovation	Porting	Integration

Exascale Apps: Impact Will Be Far-reaching for Decades to Come

- Predictive microstructural evolution of novel **chemicals and materials for energy** applications.
- Robust and selective **design of catalysts** an order of magnitude more efficient at temperatures hundreds of degrees lower.
- Accelerate the widespread adoption of additive manufacturing by enabling the **routine fabrication of qualifiable metal alloy parts**.
- Design **next-generation quantum materials** from first principles with predictive accuracy.
- Predict **properties of light nuclei** with less than 1% uncertainty from first principles.
- **Harden wind plant design** and layout against energy loss susceptibility, allowing higher penetration of wind energy.
- Demonstrate commercial-scale transformational energy technologies that **curb fossil fuel plant CO2 emission** by 2030.
- Accelerate the **design and commercialization of small and micronuclear reactors**.
- Provide the foundational underpinnings for a **'whole device' modelling capability for magnetically confined fusion plasmas** useful in the design and operation of ITER and future fusion reactors.

Exascale Apps: Impact Will Be Far-reaching for Decades to Come

- Address **fundamental science questions** such as the **origin of elements** in the universe, the **behavior of matter at extreme densities**, the source of **gravity waves**; and demystify key unknowns in the dynamics of the universe (**dark matter, dark energy and inflation**).
- Reduce the current major uncertainties in **earthquake hazard and risk assessments** to ensure the safest and most cost-effective seismic designs.
- Reliably guide safe long-term **consequential decisions about carbon storage and sequestration**.
- Forecast, with confidence, **water resource availability, food supply changes and severe weather probabilities in our complex earth system environment**.
- **Optimize power grid planning and secure operation** with very high reliability within narrow operating voltage and frequency ranges.
- Develop treatment **strategies and pre-clinical cancer drug response models** and mechanisms for RAS/RAF-driven cancers.
- Discover, through **metagenomics analysis**, knowledge useful for environment remediation and the manufacture of novel chemicals and medicines.
- Dramatically **cut the cost and size of advanced particle accelerators** for various applications impacting our lives, from sterilizing food of toxic waste, implanting ions in semiconductors, developing new drugs or treating cancer.

Four Key Ingredients of an ECP Application Development Project

Science goal

Algorithmic innovation

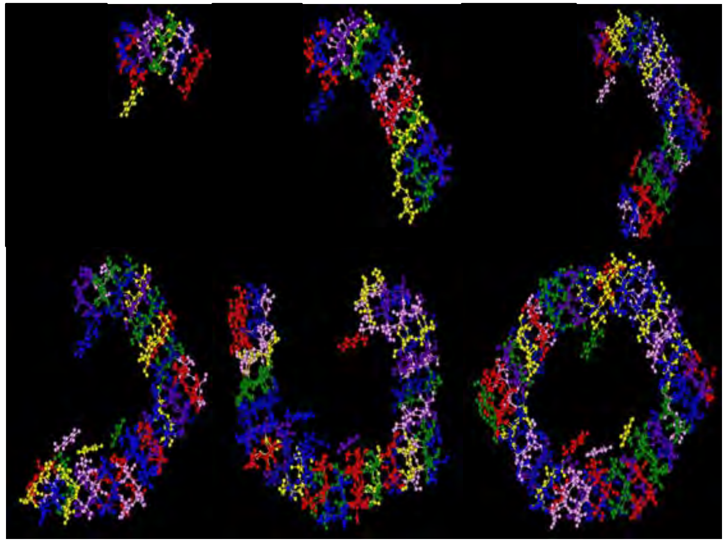
Porting

Integration

GPUs Do Best for Codes Given …

- ✓ massive fine-grained parallelism
- ✓ concentrated performance bottlenecks
- ✓ weak scaling problems
- ✓ high arithmetic intensity and/or low data movement
- ✓ minimal branching
- ✓ high FLOP to byte (of storage) ratio
- ✓ use of specialized instructions

Algorithmic Innovation: Domain-driven Adaptations Critical for Making Efficient Use of Exascale Systems



Inherent strong scaling challenges on GPU-based systems →

- Ensembles vs. time averaging
- Fluid dynamics, seismology, molecular dynamics, time-stepping

Increased dimensions of (fine-grained) parallelism to feed GPUs

- Ray tracing, Markov Chain Monte Carlo, fragmentation methods

Localized physics models to maximize "free flops"

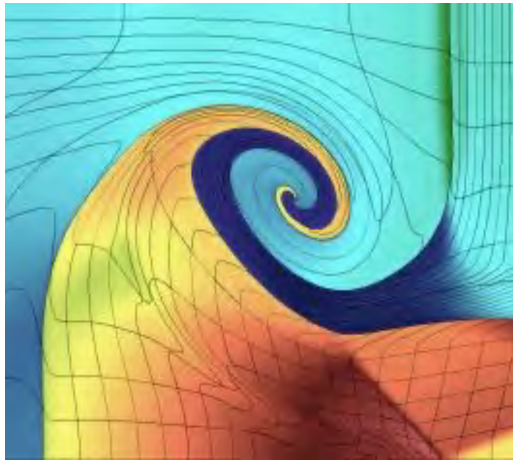
- MMF, electron subcycling, enhanced subgrid models, high-order discretizations

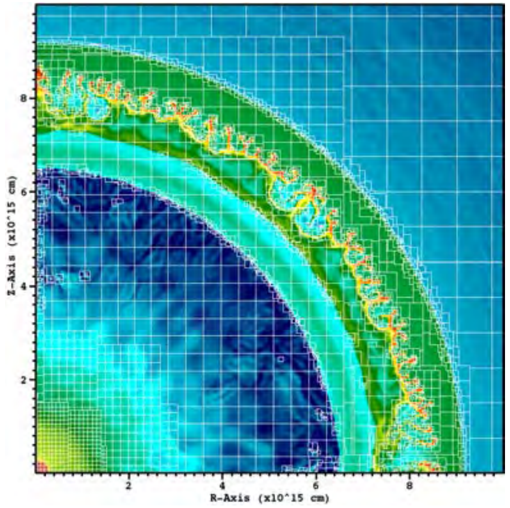
Alternatives to sparse linear systems

- Higher order methods, Monte Carlo

Reduced branching

- Event-based models

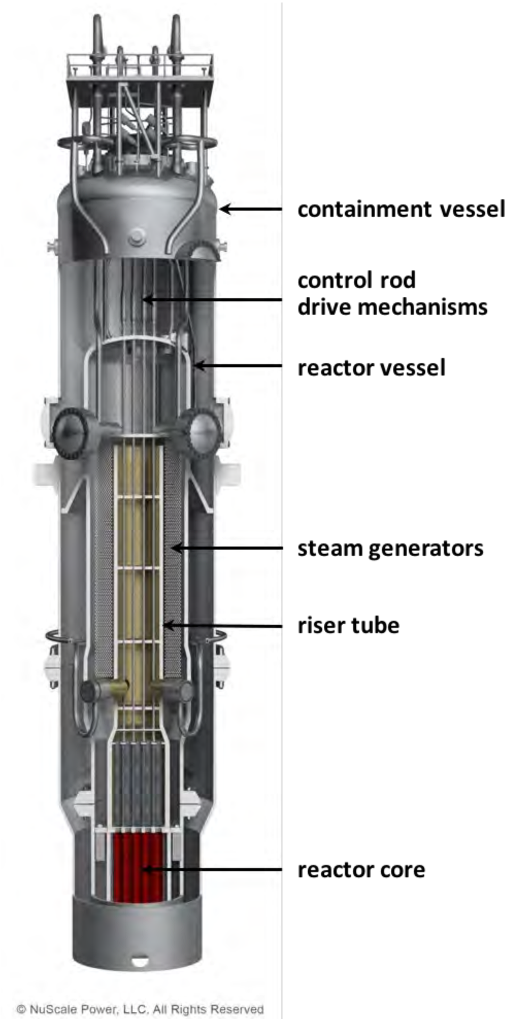


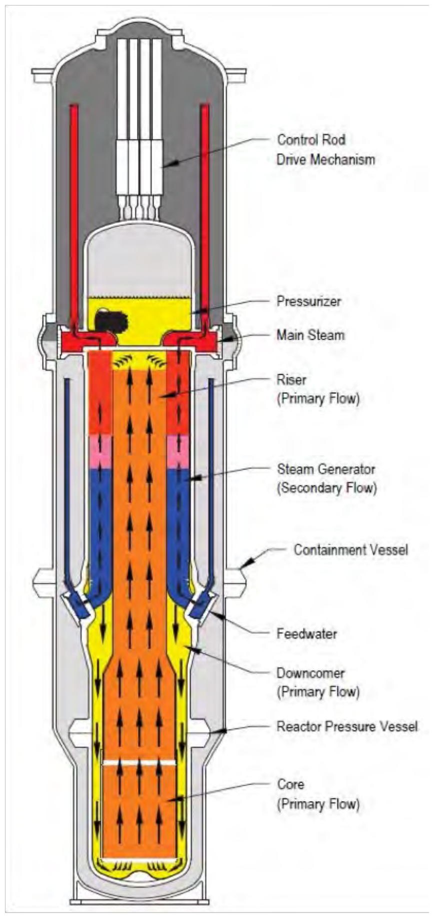


EXASCALE COMPUTING PROJECT

Example: Modeling and Simulation of Small Modular Reactors

- ExaSMR is a coupled multiphysics ECP application to perform “virtual experiment” simulations of small modular nuclear reactor designs.
- Small modular nuclear reactors present significant simulation challenges
  - Small size invalidates existing low-order models
  - Natural circulation flow requires high-fidelity fluid flow simulation
- Two primary methods:
  - Monte Carlo neutronics
  - CFD with turbulence models

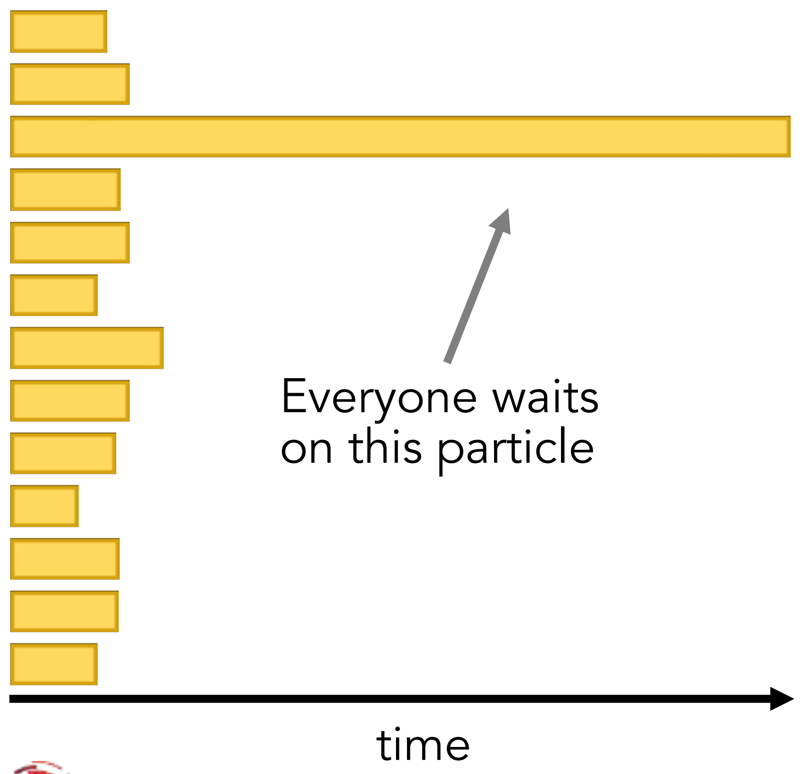


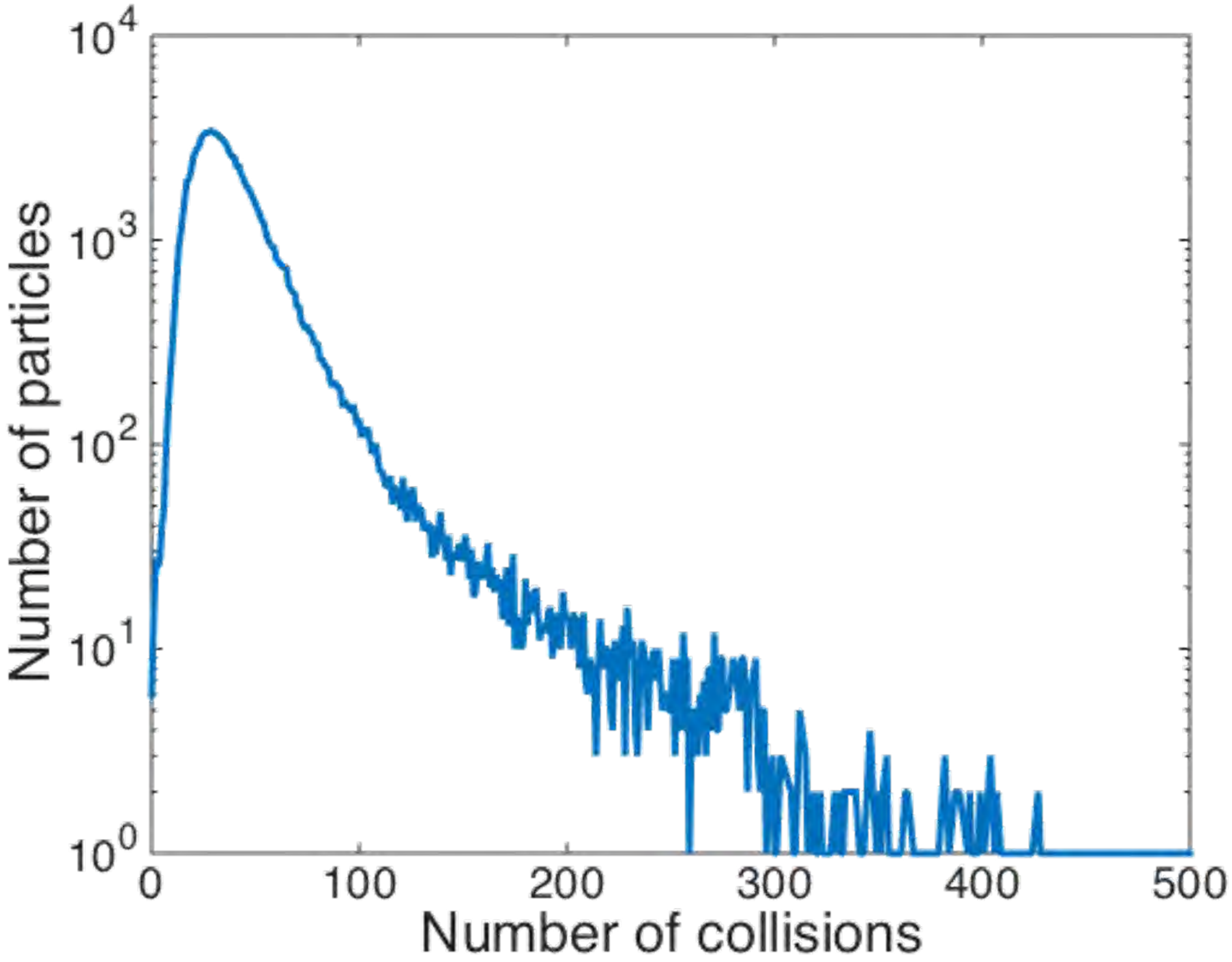


Reproduced with permission

Neutron Transport: Random Particle Statistics Poorly Suited to GPUs

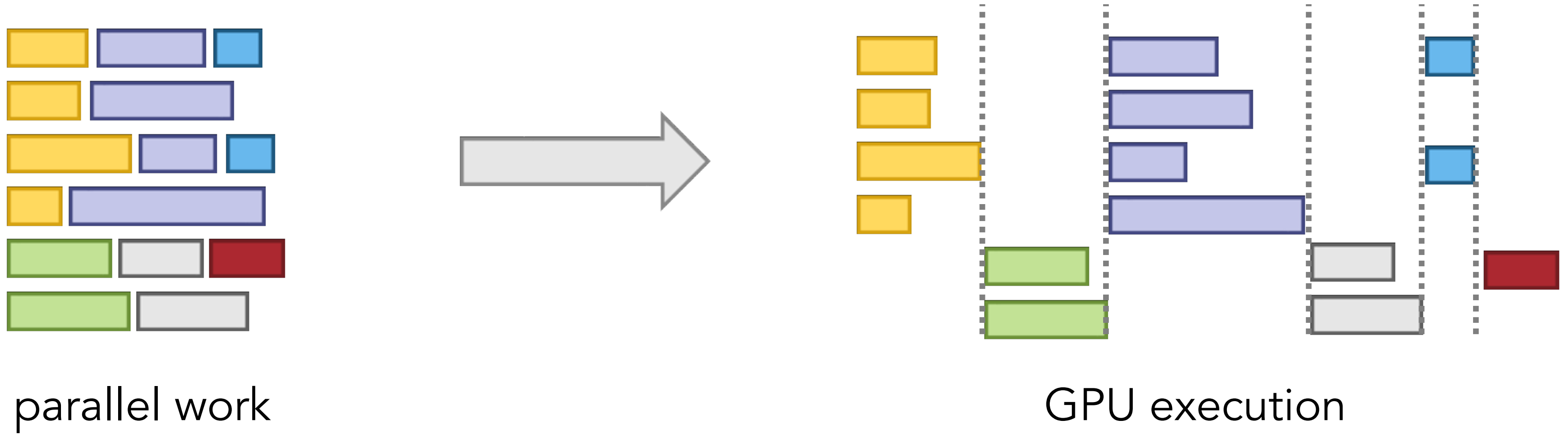
- Stochastic history-based algorithm follows particles from birth to death.
- Most particles are short-lived, a few are not.





Branching Code Is Highly Undesirable on SIMT Architectures (GPUs)

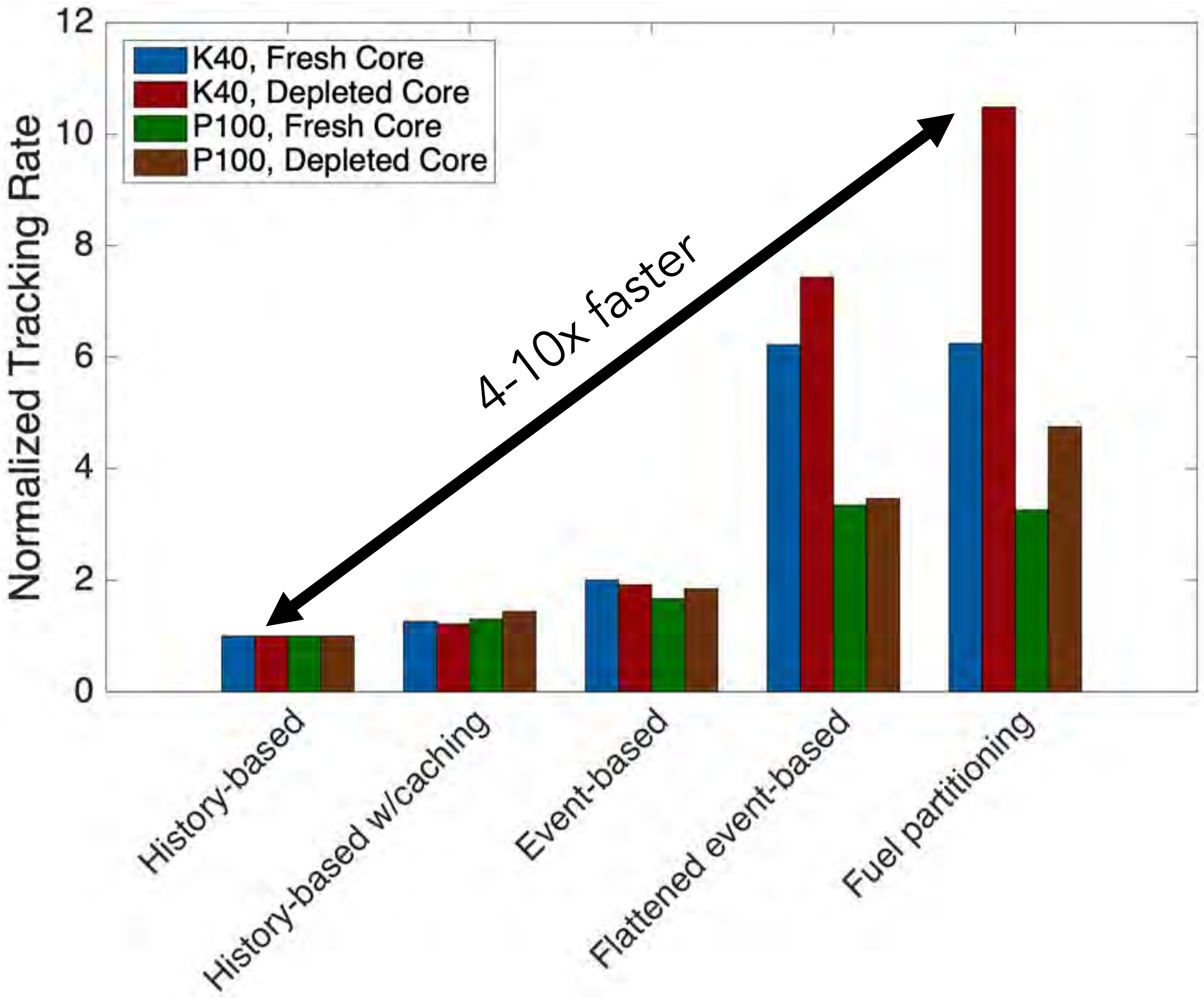
Even when each particle has roughly the same amount of work, thread divergence is a big problem when random sampling sends them down different code paths



Need to rethink code execution based on the target hardware. For example, parallelizing over events (i.e. common code paths) rather than particles.

New Event-based Algorithm Gave ExaSMR Significant Speedup

- Parallelizing over events is a much better match for a SIMT architecture than parallelizing over particles.
- Further improvements gained by identifying parts of the system that have significantly different behavior and separating them out.
- Smaller, focused kernels allow for better occupancy, i.e. more efficient use of the hardware



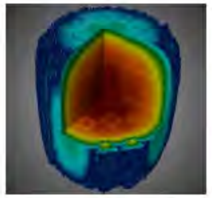
Then (2016) and Now (2023): ExaSMR

Resolved Coupled Neutronics+thermal Hydraulics Phenomena in Nuclear Reactor Cores

MC Neutronics Then

- Minimal GPU support
- Fixed material temperatures
- Single statepoint (limited isotopic depletion)
- Performance: 10^7 particles/second

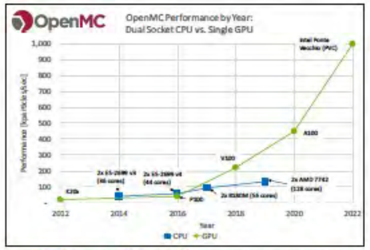
SMR core geometry



Total reaction rate in SMR core

MC Neutronics Now

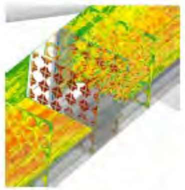
- Support for Nvidia, AMD, and Intel GPUs using HIP and OpenMP target offload
- On-the-fly Doppler broadening
- Integrated isotopic depletion capability
- Performance: >10^9 particles/second



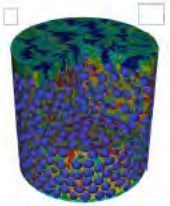
CPU vs. GPU performance over time

CFD Then

- Nek5000: CPU only (experimental OpenACC support)
- Single fuel assembly simulations
- Max problem size: 30 million elements, 10 billion DOF
- Performance: 3×10^9 DOF/second



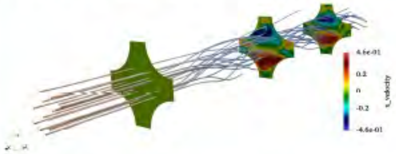
Coolant flow through mixing vane



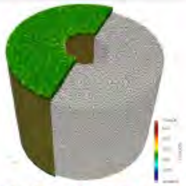
Simulation of flow through 1000 pebbles

CFD Now

- NekRS: Efficient execution on Nvidia, AMD, and Intel GPUs using OCCA
- Full SMR core with effect of heat exchanger
- Improved solver/preconditioning capabilities
- State-of-the-art mixing vane modeling
- Max problem size: 1 billion elements, 350 billion DOF
- Performance: ~5×10^{11} DOF/second



Fluid streamlines downstream of mixing vane



NekRS simulation of FHR pebble bed reactor (350k pebbles)

PI: Steve Hamilton (ORNL)

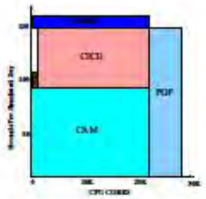
Then (2016) and Now (2023): Energy Exascale Earth System Model

Cloud-resolving Climate Modeling of the Earth's Water Cycle

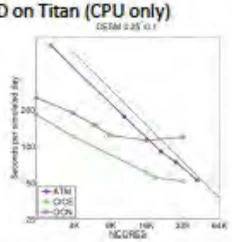
Then

Baseline model (non-MMF)

- E3SM v0 = CESM 1.2 (branch point of the E3SM project)
- High resolution configuration: 25 km atmosphere, 10 km ocean
- GPU acceleration: None
- Hydrostatic (no nonhydrostatic capability)
- 25 km model running at 1.5 SYPD on Titan (CPU only)



Coupled model performance



Strong scaling of atmosphere, ocean and sea ice

Performance at Cloud Resolving resolution

- Performance of E3SM v0 with the atmosphere running at 3 km (cloud resolving) resolution, using all of Titan
- E3SM had never run at 3 km resolution and so performance was estimated based on 25 km atmosphere
- Performance extrapolated to all of Titan, assuming perfect weak scaling, 20% coupler overhead, ocean concurrent with other components:
  - Max(atm_time+ice_time, ocn_time) * 1.2
- Figure of Merit (FOM) = 0.11 Simulated Years Per Day on all of Titan

MMF Cloud Resolving Capability

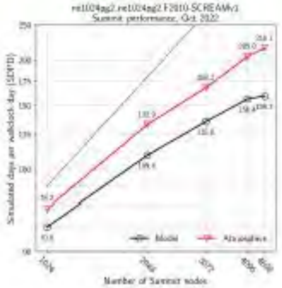
- Promising research results using MMF in CESM
- Not integrated into E3SM

THEN: Figure of Merit (FOM): 0.011 Simulated Years Per Day

Now

Baseline model (non-MMF)

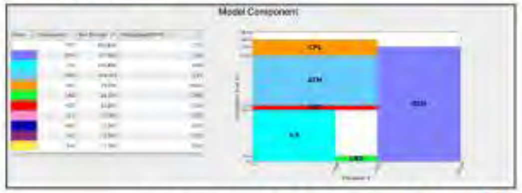
- SCREAM: E3SM's 3 km cloud resolving atmosphere model
- Rewritten from scratch, led by E3SM with many contributions from ECP E3SM-MMF project
- Nonhydrostatic dycore with HEVI-IMEX
- Atmosphere with prescribed SST simulations running on all of Summit (obtaining 0.43 SYPD) on 4600 nodes.

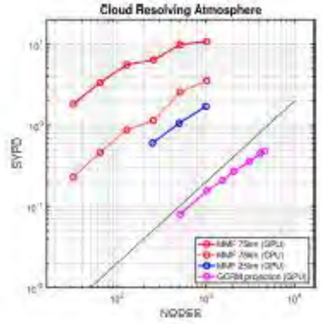


- E3SM-MMF "AMIP" simulations.
- CPU node vs 6 GPUs:
  - w/ 3D CRM: excellent GPU speedup (>20x) and scaling
  - w/ 2D CRM: 9x CRM speedup
- Baseline GCRM projection
  - Based on dycore GPU performance
- E3SM-MMF significantly faster and more efficient than GCRM approach on GPUs

E3SM-MMF Fully coupled model running on Summit

- MMF fully integrated into E3SM with many science and algorithmic improvements, and dramatically improved I/O performance via SCORPIO + ADIOS
- KPP Challenge problem running on Summit
- Weather resolving atmosphere (25 km) coupled with cloud resolving convection and turbulence (1 km)
- Coupled to the MPAS Ocean/Ice components running on the 18to6 km (Eddy Resolving) mesh
- Running at 2.03 SYPD





Node comparison: 2xP9 vs 6xV100

- Strong scaling of E3SM-MMF atmosphere component vs baseline model on Summit
- Red curves: 75 km benchmark problem, GPU vs CPU: Good GPU acceleration out to 15 GCRMS per GPU)
- Bule: 25 km KPP challenge problem running on GPUs – should scale to all of Summit
- Purple: Baseline model running on GPUs
- MMF approach achieves many aspects of a cloud resolving model and is far more efficient than the full cloud resolving baseline approach

Figure of Merit (FOM): 2.0 SYPD on Summit (181x FOM improvement)

KPP Challenge problem: Need to achieve 2.6 SYPD on Frontier

EXASCALE COMPUTING PROJECT

PI: Mark Taylor (SNL)

Four Key Ingredients of an ECP Application Development Project

Science goal	Algorithmic innovation	Porting	Integration

Porting Must Be Done With Hardware in Mind

- Map calculation to GPUs
- Map algorithm to GPUs
- Reduced communication
- Reduced synchronization
- Increased parallelism
- Reduced precision
- Optimized linear algebra

- Rewrite, profile, and optimize
  - Generally preserve the exact answer
- Data Layout for memory coalescing
- Loop ordering
- Kernel flattening
- Increased locality
- Recomputing vs. storing
- Reduced branching
- Eliminating copies

- Identify opportunities for improvement
- Mathematical representation
- "On the fly" recomputing vs. lookup tables
- Prioritization of new physical models
- Alternate discretizations (high AI)
- Localized subgrid models
- Sparse → dense systems
- Defining weak scaling target
- Initial condition from ROM

Hardware has significant impact on all aspects of simulation strategy

EXASCALE COMPUTING PROJECT

Choosing the Right Programming Model is All About Balancing Trade-offs

GPU-specific kernels

- Isolate the computationally-intensive parts of the code into CUDA/HIP/SYCL kernels.
- Refactoring the code to work well with the GPU is the majority of effort.

Loop pragma models

- Offload loops to GPU with OpenMP or OpenACC.
- Most common portability strategy for Fortran codes.

C++ abstractions

- Fully abstract loop execution and data management using advanced C++ features.
- Kokkos and RAJA developed by NNSA in response to increasing hardware diversity.

Co-design frameworks

- Design application with a specific motif to use common software components
- Depend on co-design code (e.g. CEED, AMReX) to implement key functions on GPU.

Application Motifs* (What's the App Footprint?)

Algorithmic Methods that Capture a Common Pattern of Computation and Communication

1. **Dense Linear Algebra**
   - Dense matrices or vectors (e.g., BLAS Level 1/2/3)
2. **Sparse Linear Algebra**
   - Many zeros, usually stored in compressed matrices to access nonzero values (e.g., Krylov solvers)
3. **Spectral Methods**
   - Frequency domain, combining multiply-add with specific patterns of data permutation with all-to-all for some stages (e.g., 3D FFT)
4. **N-Body Methods (Particles)**
   - Interaction between many discrete points, with variations being particle-particle or hierarchical particle methods (e.g., PIC, SPH, PME)
5. **Structured Grids**
   - Regular grid with points on a grid conceptually updated together with high spatial locality (e.g., FDM-based PDE solvers)
6. **Unstructured Grids**
   - Irregular grid with data locations determined by app and connectivity to neighboring points provided (e.g., FEM-based PDE solvers)
7. **Monte Carlo**
   - Calculations depend upon statistical results of repeated random trials
8. **Combinational Logic**
   - Simple operations on large amounts of data, often exploiting bit-level parallelism (e.g., Cyclic Redundancy Codes or RSA encryption)
9. **Graph Traversal**
   - Traversing objects and examining their characteristics, e.g., for searches, often with indirect table lookups and little computation
10. **Graphical Models**
    - Graphs representing random variables as nodes and dependencies as edges (e.g., Bayesian networks, Hidden Markov Models)
11. **Finite State Machines**
    - Interconnected set of states (e.g., for parsing); often decomposed into multiple simultaneously active state machines that can act in parallel
12. **Dynamic Programming**
    - Computes solutions by solving simpler overlapping subproblems, e.g., for optimization solutions derived from optimal subproblem results
13. **Backtrack and Branch-and-Bound**
    - Solving search and global optimization problems for intractably large spaces where regions of the search space with no interesting solutions are ruled out. Use the divide and conquer principle: subdivide the search space into smaller subregions ("branching"), and bounds are found on solutions contained in each subregion under consideration

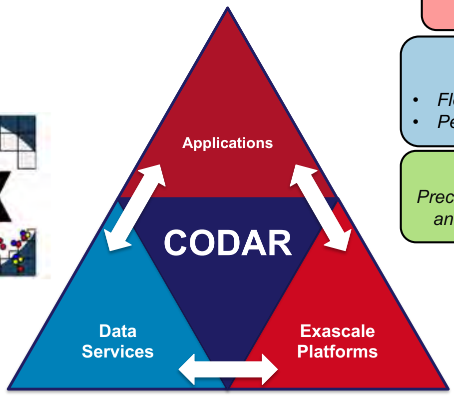
The Landscape of Parallel Computing Research: A View from Berkeley, Technical Report No. UCB/EECS-2006-183 (Dec 2006).

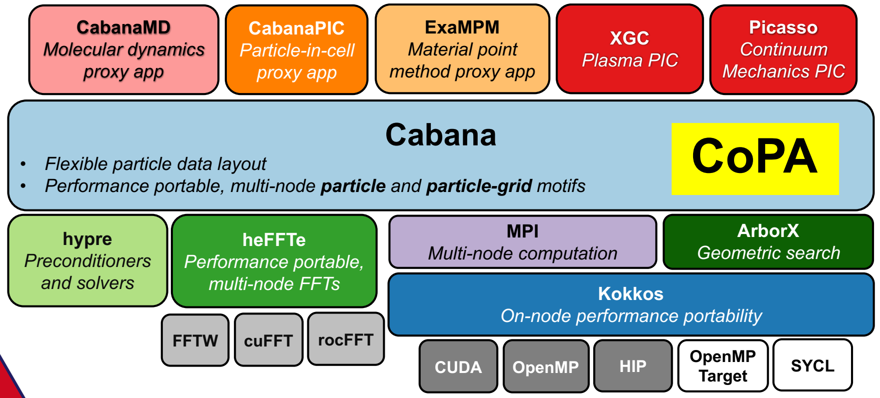
ECP Co-design Centers for Key Computational Motifs

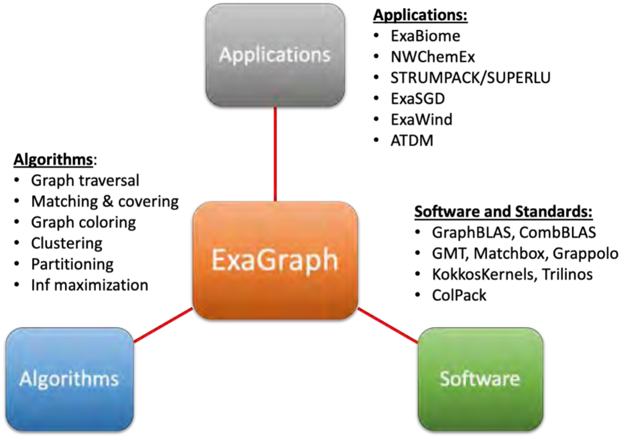
Project	PI Name, Inst	Short Description/Objective
CODAR	Ian Foster, ANL	Understand the constraints, mappings, and configuration choices between applications, data analysis and reduction, and exascale platforms
AMReX	John Bell, LBNL	Build framework to support development of block-structured adaptive mesh refinement algorithms for solving systems of partial differential equations on exascale architectures
CEED	Tzanio Kolev, LLNL	Develop next-generation discretization software and algorithms that will enable finite element applications to run efficiently on future hardware
CoPA	Susan Mniszewski, LANL	Create co-designed numerical recipes and performance-portable libraries for particle-based methods
ExaGraph	Mahantesh Halappanavar, PNNL	Develop methods and techniques for efficient implementation of key combinatorial (graph) algorithms
ExaLearn	Frank Alexander, BNL	Deliver state-of-the-art machine learning and deep learning software at the intersection of applications, learning methods, and exascale platforms

CEED
EXASCALE DISCRETIZATIONS

AMReX





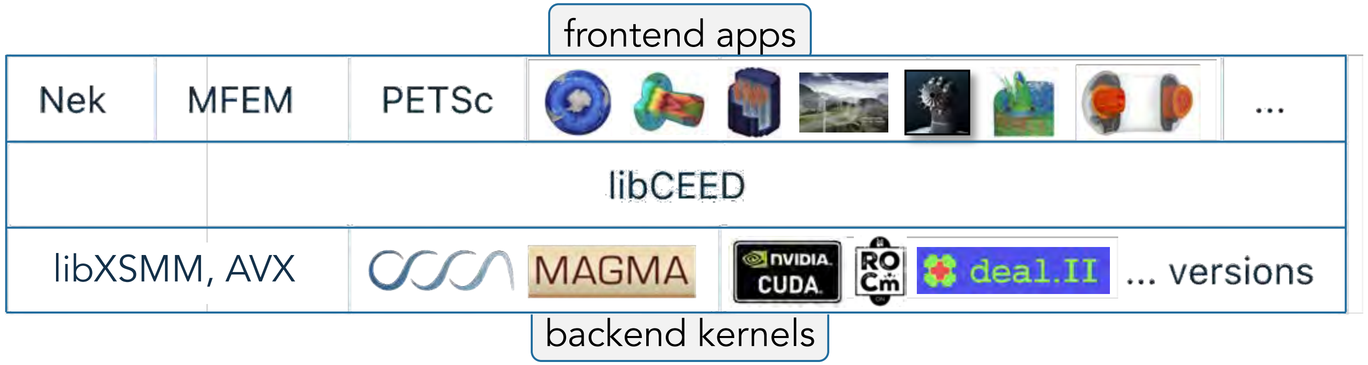


EXASCALE COMPUTING PROJECT

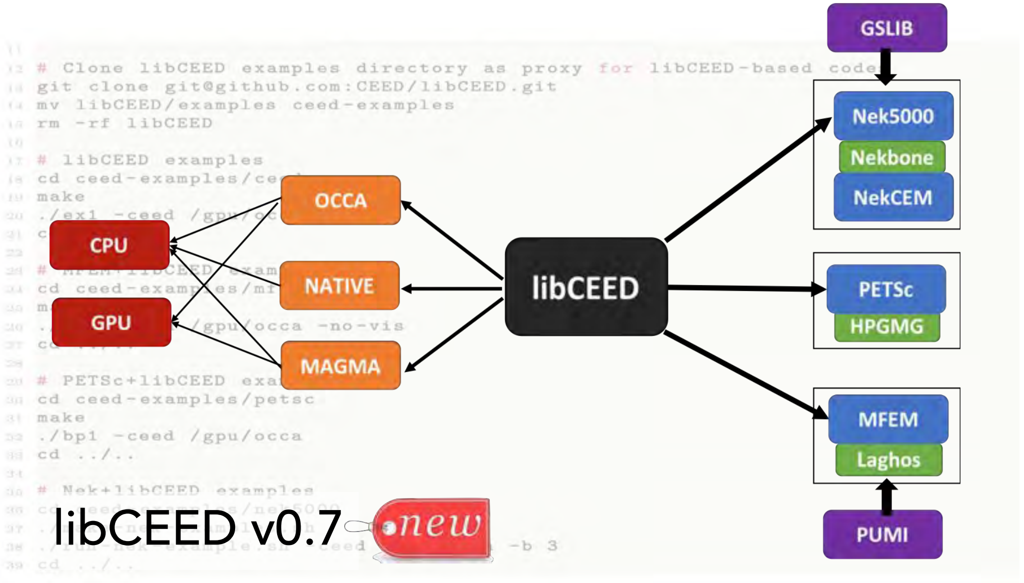
CEED Provides Multiple Back-ends, Including Through Its OCCA Portability Layer

Principal Motif: Unstructured Mesh Finite Element Discretization

CEED EXASCALE DISCRETIZATIONS



- ✓ API between *frontend apps* and *backend kernels*
- ✓ *Efficient operator description* (not global matrix)
- ✓ Clients use any backend as a run-time option
- ✓ Backend can be added as plugins without recompiling
- ✓ Backends compete for best performance, latency vs throughput, optimize for order/device, use JIT, …

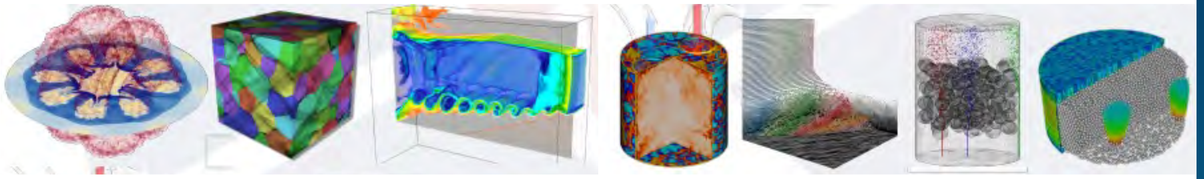


✓ Extensible backends

- **CPU**: reference, vectorized, libXSMM
- **CUDA** using NVRTC cuda-gen
- **OCCA** (JIT): CPU, OpenMP, OpenCL, CUDA
- **MAGMA**

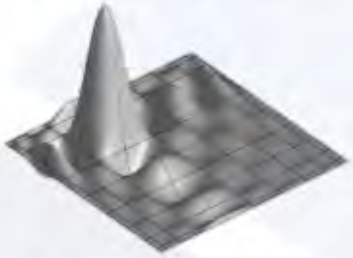
Then (2016) and Now (2023): CEED

Center for Efficient Exascale Discretizations

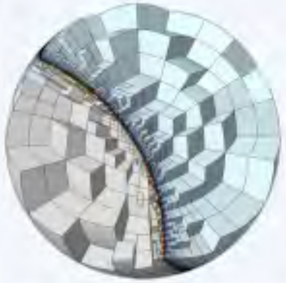


Then

- **PDE-based simulations on unstructured grids**
- **High-order and spectral finite elements**
  - ✓ *any order space on any order mesh*
  - ✓ *curved meshes,*
  - ✓ *unstructured AMR*
  - ✓ *matrix-free methods*
  - ✓ *optimized low-order support*



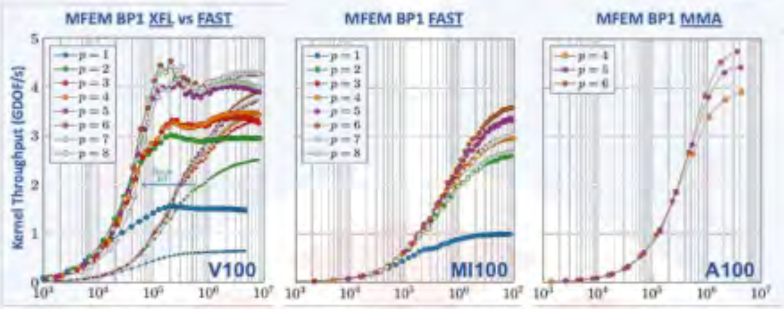
10th order basis function



non-conforming AMR, 2nd order mesh

Now *new*

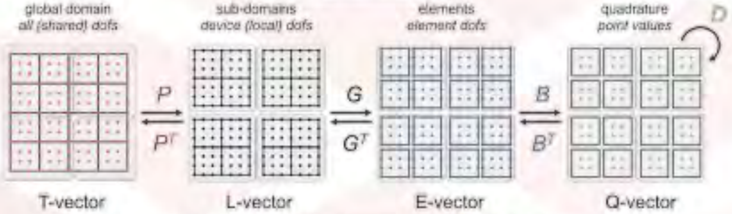
- **CEED discretization libraries**
  - ✓ **High-Level API:** *Nek & MFEM projects*
  - ✓ Nek5000/NekRS: nek5000.mcs.anl.gov
  - ✓ MFEM: mfem.org



New MFEM GPU kernels: (1) have better strong scaling, (2) perform on NVIDIA + AMD GPUs, and (3) can utilize tensor cores

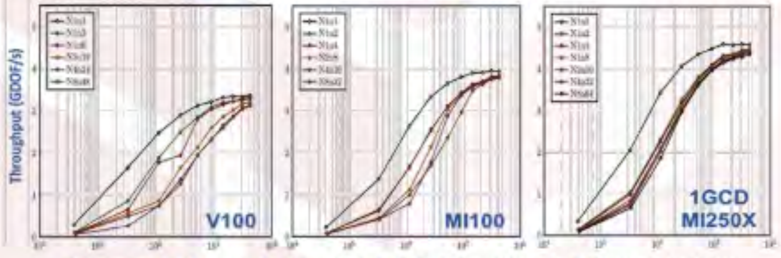
- **libCEED** github.com/CEED/libceed
  - ✓ **Low-Level API:** *new library for efficient operator evaluation*
  - ✓ *state-of-the-art CPU and GPU kernel performance*

$A = P^T G^T B^T D B G P$



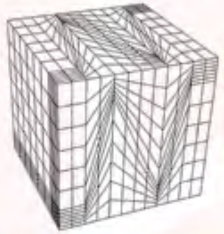
Finite element operator decomposition

- **Miniapps:** Laghos, libParanumal, hipBone



hipBone performance for order 7: 1 MI250X GCD = 1.2 MI100 = 1.3 V100

- **Benchmarks**
  - ✓ *bake-off problems:* **BP1-BP6**
  - ✓ *solver BPs:* **BPS3, BPS5**
  - ✓ *high-order community benchmarks*



Kershaw mesh

- **High-order software ecosystem**
  - ✓ *high-order meshing, optimization*
  - ✓ *high-order visualization*
  - ✓ *performance portability, GPUs*
  - ✓ *scalable "matrix-free" solvers*

RAJA, MAGMA, PETSc, hypre, VTK-m, Ascent, PUMi

- **More information and downloads**
  - ✓ *CEED project website:* ceed.exascaleproject.org
  - ✓ *CEED code repositories:* github.com/CEED

ECP EXASCALE COMPUTING PROJECT

PI: Tzano Kolev (LLNL)

CoPA: Cabana Particle Library is Built on a Kokkos Portability Layer

Principal Motif: Particles

CabanaMD
Molecular dynamics proxy app

CabanaPIC
Particle-in-cell proxy app

ExaMPM
Material point method proxy app

XGC
Plasma PIC

Picasso
Continuum Mechanics PIC

Cabana

- *Flexible particle data layout*
- *Performance portable, multi-node* ***particle*** *and* ***particle-grid*** *motifs*

hypre
Preconditioners and solvers

heFFTe
Performance portable, multi-node FFTs

FFTW | **cuFFT** | **rocFFT**

MPI
Multi-node computation

ArborX
Geometric search

Kokkos
On-node performance portability

CUDA | **OpenMP** | **HIP** | **OpenMP Target** | **SYCL**

Then (2016) and Now (2023): CoPA

Addressing The Challenges For Particle-based Applications To Run On Exascale Architectures

Cabana: A Co-Designed HPC Library for Particle Applications

https://github.com/ECP-CoPA/Cabana

Lead: Sam Reeve (ORNL), Co-lead: Stuart Slattery (ORNL)

*Developers: Christoph Junghans (LANL), Damien Lebrun-Grandie (ORNL), Austin Isner (ORNL), Kwitae Chong (ORNL), Shane Fogerty (LANL), Aaron Scheinberg (PPPL-consultant), Guangye Chen (LANL), **Yuxing Qiu (UCLA), Yu Fang (UCLA), Stephan Schulz (Jülich)**, Jim Glosli (LLNL), Evan Weinberg (NVIDIA)*

Collaborators: Stan Moore (SNL), Lee Ricketson (LLNL), Steve Rangel (ANL), Adrian Pope (ANL), Mark Stock (HPE)

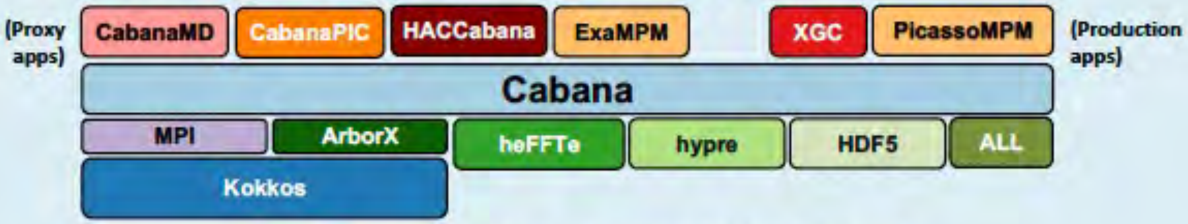
How we started

- Each particle application defined and implemented separate particle data structures, algorithms, and communication, even with some significant overlap between domains: **Cabana did not exist.**
- Each partner application had different strategies for the coming exascale and performance portability (direct vendor backends for **HACC** and Kokkos for **LAMMPS**), but some strategies were unsustainable (multiple sets of conflicting and complex dependencies for **XGC**). Finally, the **PicassoMPM** application did not exist.

Where we are now

Cabana is a full-featured particle library as an extension of Kokkos

- Particle data structures, particle algorithms, and multi-node particle communication
- Structured grids, grid algorithms, multi-node grid communication, and particle-grid interpolation
- Particle algorithms, load balancing, and I/O through optional third-party libraries



Tier-1 application partner integrations

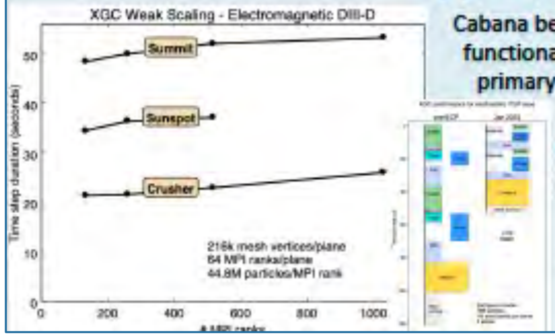
Cabana provides benefits across many use cases, exemplified by our app partners:

- **XGC:** Direct use of Cabana for migration to performance portability with plans for further algorithm adoption
- **PicassoMPM:** Full use of Cabana for development of a brand new particle-grid application
- **HACC:** Proxy app for rapid exploration of new algorithms and designs alongside production codes (**HACCabana**)
- **LAMMPS:** Comparison and sharing of algorithms and Kokkos performance strategies

Additional impact

- PIC algorithm development using Cabana for rapid prototyping (**CabanaPIC**)
- Sharing of algorithm and performance strategies with the **AMReX** adaptive mesh refinement library
- New non-ECP applications: **CabanaPD** (ORNL LDRD peridynamics), **Hyperion** (LANL LDRD multi-physics hybrid PIC), **MRMD** (Max Planck multi-resolution MD), **PUMI-PIC** (RPI PIC), **Beatnik** (UNM PSAAP Z-model)

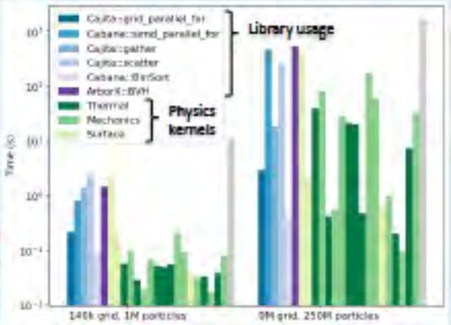
Application performance



Cabana benchmarks across all functionality and runs on all primary Kokkos backends

Left: **XGC** weak scaling & speedup

Right: **PicassoMPM** Crusher single node GPU performance

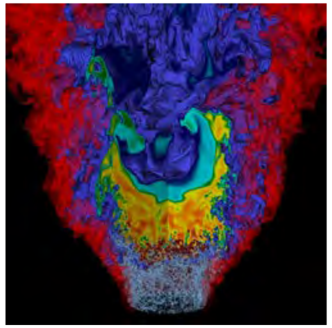


PI: Sue Mniszewski (LANL)

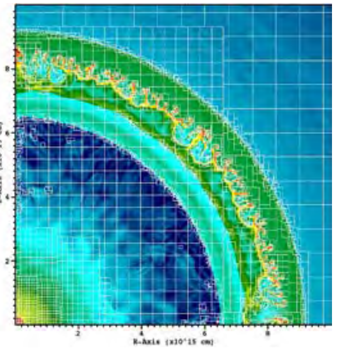
AMReX Provides Portability to ECP Applications Through Multiple Low-level Implementations

Principal Motif: Structured Mesh, Patch-based Adaptive Mesh Refinement

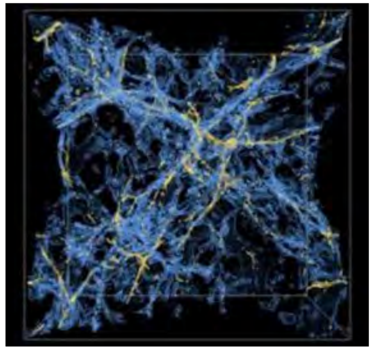
Combustion-PELE (PeleC and PeleLM)



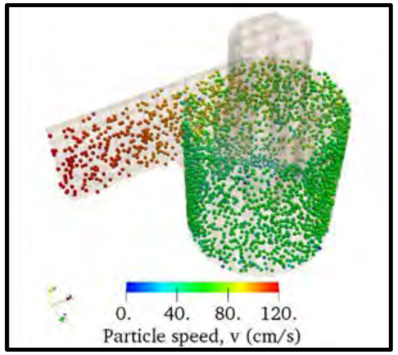
ExaStar (Castro)



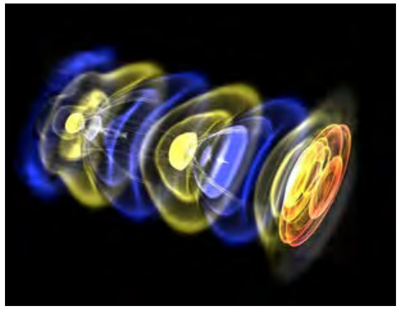
ExaSky (Nyx)



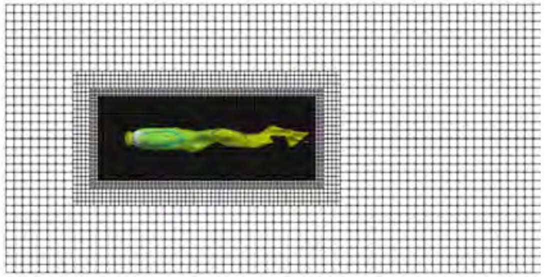
MFIX-Exa



WarpX



ExaWind (AMR-Wind)



AMReX

MPI | **OpenMP** | **OpenACC**

CUDA | **HIP** | **DPC++**

https://amrex-codes.github.io/

Then (2016) and Now (2023): AMReX

Adaptive Refinement of Patch-based Structured Meshes

AMReX Then		AMReX Now
Mix of C++11 (data structures, high-level control flow) and Fortran (low-level numerical operations)	**Source code**	Source code: pure C++17 with extensive use of template metaprogramming
MPI + OpenMP only	**Hybrid Parallelism**	MPI + X , where X is one of OpenMP (CPUs) or CUDA, HIP or SYCL (NVIDIA, AMD, or Intel GPUs)
Support for redistribution and particle-mesh, array-of-structs only	**Particles**	Both array-of-struct and struct-of-array data, halo exchange + neighbor lists for particle-particle collisions
None	**Complex Geometry**	Support for embedded boundaries via cut-cell approach
Native multi-level geometric multigrid	**Linear Solvers**	Same + EB-aware options, interfaces to hypre and PETSc
GNUMake only	**Installation**	CMake + GNUMake for compilation from source One step installation with Spack
Native plotfiles	**IO**	Native plotfiles + HDF5, support for compression with SZ and ZFP, Asynchronous IO
VisIt, yt, Paraview	**Visualization**	Same + support for in-situ analysis and visualization with ALPINE, SENSEI
Manual runs of test suite Limited documentation Informal code reviews for critical changes	**Development policies/practices**	Extensive test coverage with continuous integration Extensive online documentation and tutorials Formal code reviews for all changes
Applications could run at full-scale on Edison, Cori KNL	**Performance**	**AMReX applications can run efficiently at full-scale on Perlmutter, Fugaku, Summit, and Frontier.**

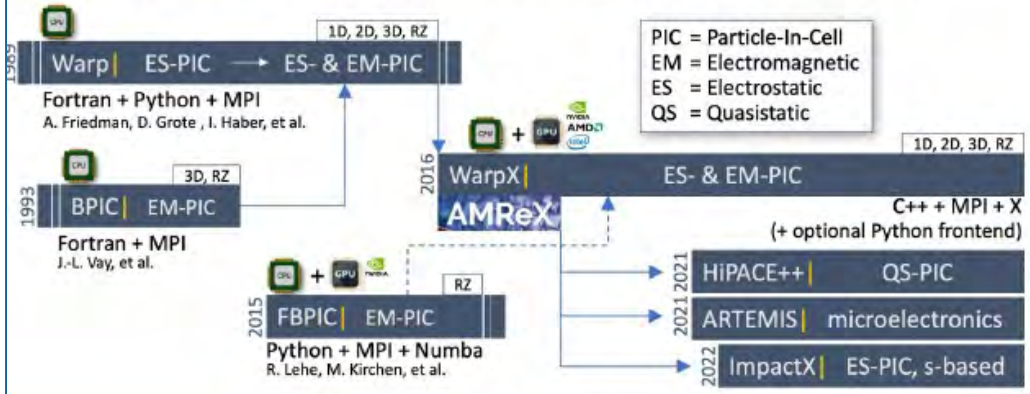
EXASCALE COMPUTING PROJECT

PI: John Bell (LBNL)

Then (2016) and Now (2023): WarpX

Modeling of Charged Particle Beams and Accelerators, Lab & Astro Plasmas, Fusion Devices

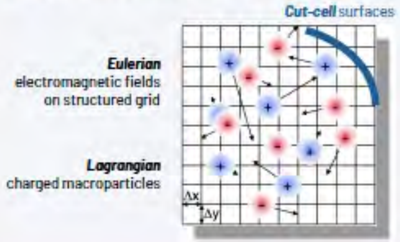
WarpX & Spinoffs History & Roadmap



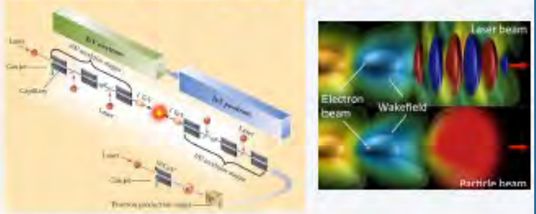
Overview of Warp/WarpX

Warp and WarpX are multiphysics codes/frameworks for the modeling of charged particle beams and accelerators, lab & astro plasmas, fusion devices & more.

Codes are constructed around the Particle-In-Cell (PIC) algorithm:



Challenge ECP problem: the modeling of chains of plasma-based particle accelerators for future high-energy physics colliders



Warp as of 2016		WarpX as of 2023
large set of advanced, novel algorithms	**Algorithms**	Warp advanced algos + new algorithms introduced during ECP
50% Fortran + 50% Python (including programmable frontend) had grown to large >1M lines of codes w/ varying programming styles	**Source code**	Source code: C++17 & optional Python programmable frontend compact thanks to C++ templating
CPUs, MPI-parallel	**Supported hardware**	CPUs, 3 flavors of GPUs, MPI-parallel
limited	**Performance optimization**	extensive
limited support, independently	**Load balancing & AMR**	combined native support
compilation from source some support for binaries	**Installation**	standard (CMake) compilation from source one step with Spack/Conda/PyPI, multi-platform
small team (2+) of computational physicists + individual contributions over several decades	**Development team**	tightly integrated team of computational physicists + applied mathematicians + computer scientists + software engineers
manual runs of test suite partial online documentation, outdated in part informal code reviews for critical changes	**Development policies/practices**	extensive test coverage with continuous integration extensive online documentation formal code reviews for all changes
could perform 3-D modeling of single plasma accelerator stage at moderate resolution	**ECP science case**	**can perform 3-D modeling of chain of tens of plasma accelerator stages at twice the resolution in each direction**

PI: Jean-Luc Vay (LBNL)

WarpX's "Then And Now" is Compelling . . . As It is for Every Team

Each ECP Team's Articulation of This Reality Will Help With Adoption, Sustainability, Evolution

Warp (as of 2016)	WarpX (as of 2022)
Runs on CPUs	Runs on CPUs & 3 vendors of GPUs
~ 50% Fortran + 50% Python	100% C++ + optional Python frontend
Many advanced algorithms & physics	More & better algorithms & physics
Good scaling to ~6000 CPU nodes	Good scaling to ~150000 CPU nodes, 8000 GPU nodes
No dynamic load balancing	Efficient load balancing
"Home-made", brittle Mesh refinement capability	Mesh refinement based on robust AMReX library
Scaling of I/Os was a bottleneck	Good scaling of I/Os with ADIOS/HDF5
Installation required compilation	Easy installation with Spack, Conda, …
Manual tests ensured correctness	~200 physics benchmarks run automatically on every code commit
Modeling of one plasma accelerator stage at moderate resolution	Modeling of 10+ plasma accelerator stages at high resolution

Figure-of-Merit over time

Date	Code	Machine	N_c/Node	Nodes	FOM
3/19	Warp	Cori	0.4e7	6 625	2.2e10
3/19	WarpX	Cori	0.4e7	6 625	1.0e11
6/19	WarpX	Summit	2.8e7	1 000	7.8e11
9/19	WarpX	Summit	2.3e7	2 560	6.8e11
1/20	WarpX	Summit	2.3e7	2 560	1.0e12
2/20	WarpX	Summit	2.5e7	4 263	1.2e12
6/20	WarpX	Summit	2.0e7	4 263	1.4e12
7/20	WarpX	Summit	2.0e8	4 263	2.5e12
3/21	WarpX	Summit	2.0e8	4 263	2.9e12
6/21	WarpX	Summit	2.0e8	4 263	2.7e12
7/21	WarpX	Perlmutter	2.7e8	960	1.1e12
12/21	WarpX	Summit	2.0e8	4 263	3.3e12
4/22	WarpX	Perlmutter	4.0e8	928	1.0e12
4/22	WarpX	Perlmutter†	4.0e8	928	1.4e12
4/22	WarpX	Summit	2.0e8	4 263	3.4e12
4/22	WarpX	Fugaku†	3.1e6	98 304	8.1e12
6/22	WarpX	Perlmutter	4.4e8	1 088	1.0e12
7/22	WarpX	Fugaku	3.1e6	98 304	2.2e12
7/22	WarpX	Fugaku†	3.1e6	152 064	9.3e12
7/22	WarpX	Frontier	8.1e8	8 576	1.1e13

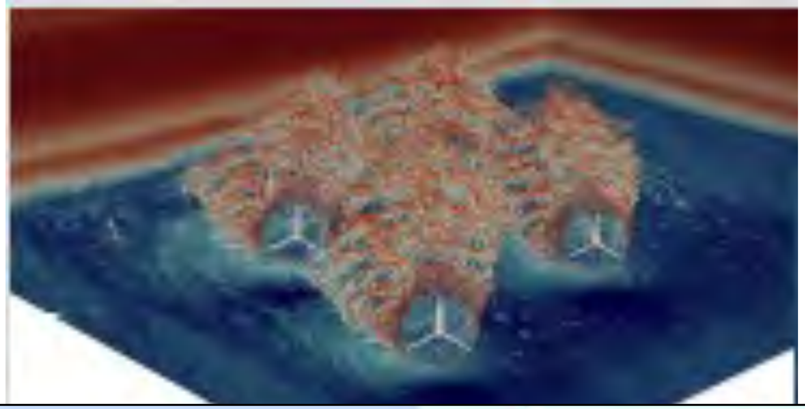
500x

Computational power increase:
- 500x: Warp (2016) ➔ WarpX (2022)

ECP EXASCALE COMPUTING PROJECT

Then (2016) and Now (2023): ExaWind

Predictive Physics-based Simulation Of Wind Plants



Then (2016)

Approach: Create computational fluid/structure dynamics (CFD and CSD) codes for Reynolds-averaged Navier-Stokes (RANS)/large-eddy simulations (LES) where wind turbine geometry and blade boundary layers are resolved and include moving meshes, fluid-structure interaction, and atmospheric turbulence

Starting-Point Codes:

Nalu: https://github.com/nalucfd/

- Unstructured-grid, incompressible-flow CFD
- LES turbulence model
- C/C++
- Built on Trilinos STK, Tpetra/Belos/MueLu solvers, and Kokkos
- Mesh rotation achieved through a sliding-mesh interface

OpenFAST: https://github.com/openfast/

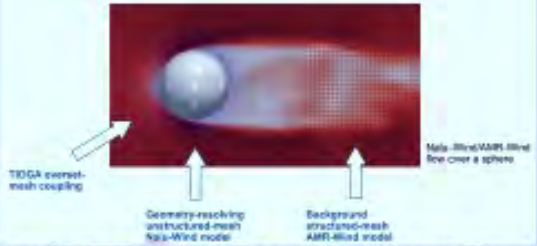
- Whole-turbine simulation code (structural dynamics, control)
- Fortran90

Challenges:

- Target problem requires resolving spatial scales going from blade boundary layers (e.g., 10^{-5} m) to the wind farm domain (e.g., 10^{3} m), i.e., at least eight orders of magnitude
- Finite volumes with extreme aspect ratios (e.g., 10,000), which are necessary for hybrid-RANS/LES, were a serious challenge linear-system solvers
- Time-integration scheme required impractically small time-step sizes (e.g., 10^{-6} s) for production simulations
- Sliding-mesh approach presented mesh-creation challenges and no clear pathway for yaw motions

Now (2023)

Shift in Approach: Added AMR-Wind as a background solver and made Nalu-Wind the near-body solver; coupling via overset meshes



Proof-of-concept simulation of flow over a sphere using the hybrid Nalu-Wind/AMR-Wind solver.

Primary Application Codes:

Nalu-Wind

- https://github.com/exawind/nalu-wind
- Wind-specific offshoot from Nalu; primarily used for near-body flows
- *hypre* is primary linear-system-solver package
- Hybrid-RANS/LES with time integrator that enables practical time step sizes
- Overset meshes (via TIOGA, https://github.com/jsitaraman/tioga) is primary method for moving meshes
- Performant on NVIDIA GPUs; Advanced Micro Devices, Inc. (AMD) GPUs are in progress

AMR-Wind

- https://github.com/exawind/amr-wind
- Structured-grid adaptive mesh refinement (AMR) CFD code; background solver
- C++ and built on the AMReX library
- Performant on NVIDIA and AMD GPUs

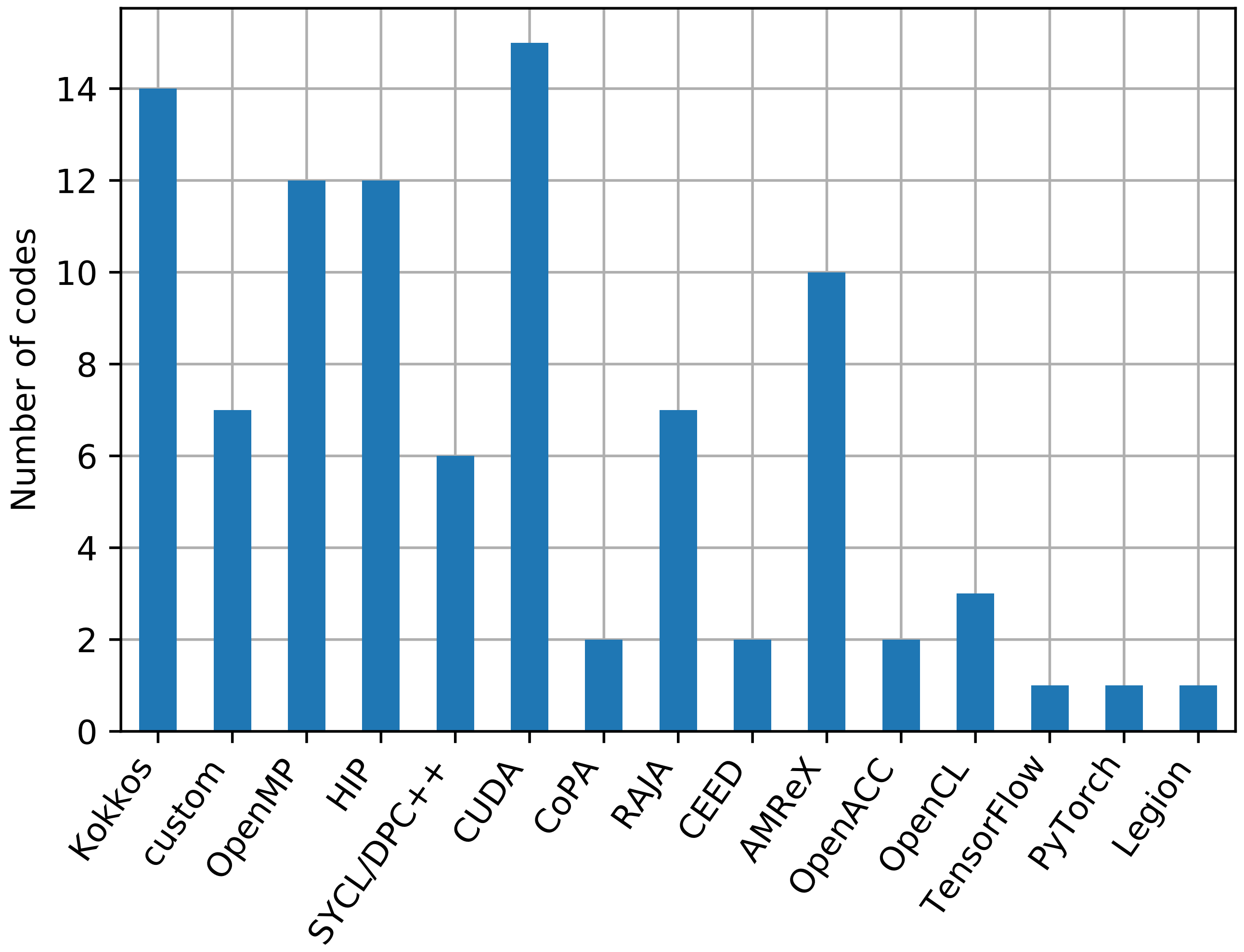
OpenFAST

- No pathway to support parallelization or GPUs
- Starting new FY23 WETO project to create replacement: OpenTurbine https://github.com/exawind/openturbine

EXASCALE COMPUTING PROJECT

PI: Mike Sprague (NREL)

Programming Models Used in ECP Applications



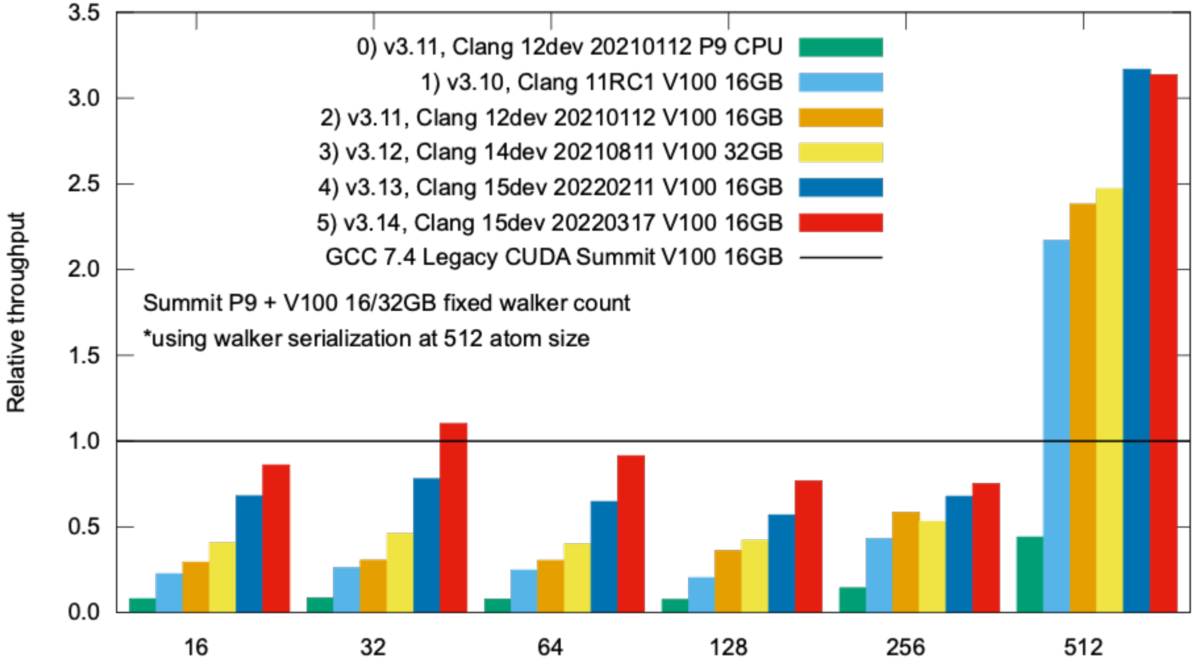
Platform portability provided by co-design projects (CoPA, CEED, AMReX)	33%
Native (CUDA/HIP/SYCL) or custom implementations	33%
ST programming models (Kokkos, RAJA, Legion)	18%
Directive-based programming models: (OpenMP, OpenACC)	16%

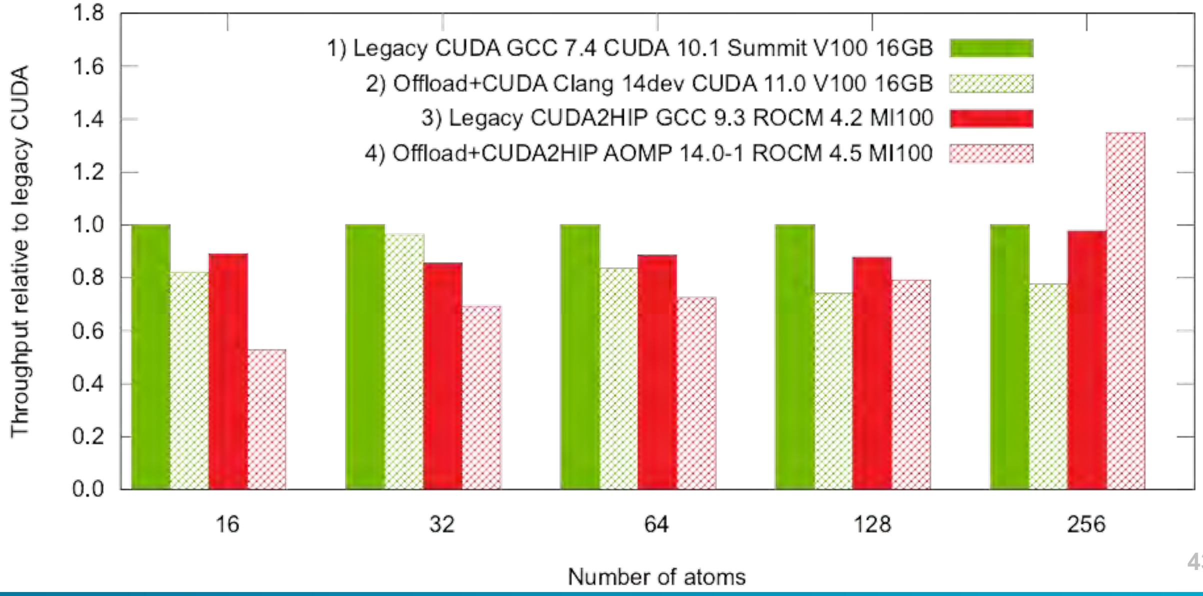
- Use of co-design/ST technologies provides significant benefit. Fine-scale architectural details provided by co-design technologies
- Large percent of custom implementations reflects difficulty of universal platform-portable programming models that span diverse apps

ECP EXASCALE COMPUTING PROJECT

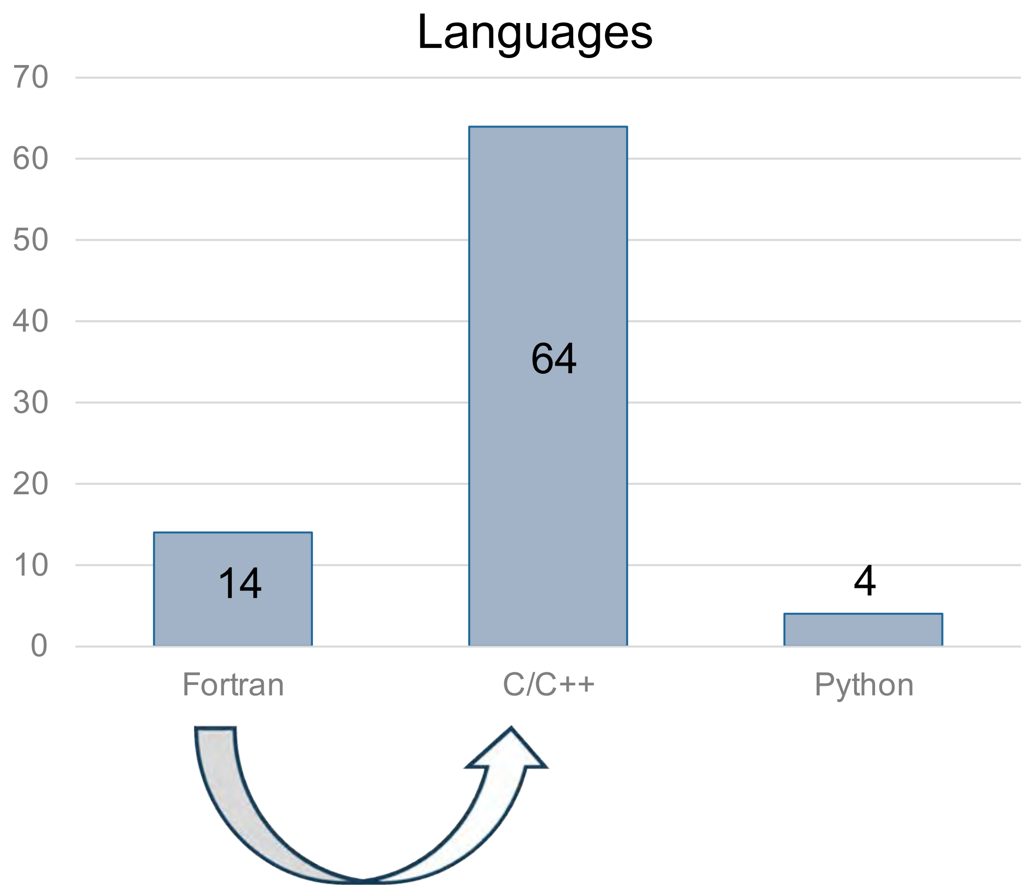
QMCPACK was First Through the Wall

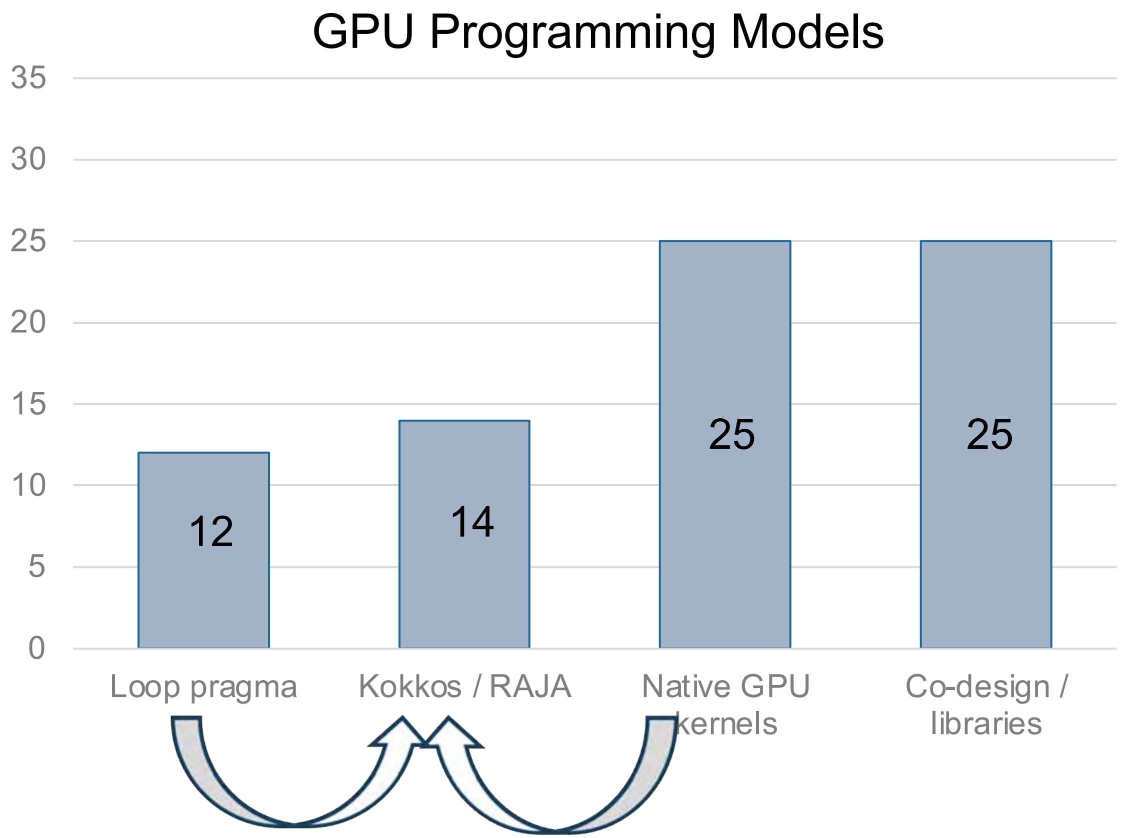
- QMCPACK had a working CUDA implementation of the code that proved invaluable in understanding where OpenMP performance was falling short.
- OpenMP offload runtimes are not yet consistently performant across vendors. Initial OpenMP results were significantly slower than CUDA.
- With careful performance analysis and by working closely with the vendors, the QMCPACK team was able to steadily improve performance of their OpenMP version until it is now on par with CUDA.





Distribution of ECP Programming Models Has Changed Over Time





My have programming language/model choices have evolved over course of ECP!!

- *Of Note: Recent LLMs (CoPilot, etc.) appear to have adequately "learned" abstraction layers (Kokkos / RAJA) well enough to effectively port and translate code. Does that mean ECP was a waste? NO!!!*

EXASCALE COMPUTING PROJECT

Four Key Ingredients of an ECP Application Development Project

Science goal

Algorithmic innovation

Porting

Integration

Integration: ECP Applications Rely Heavily on High Quality Software Tools and Libraries

ECP Applications: C **Cosmology** (ExaSky) · F **Fusion Energy** (WDMApp) · N **National Security** (MAPP) · S **Subsurface Flow** · W **Wind Farm** (ExaWind) · **… and more**

24 apps, 6 co-design centers

Shown are 36 ST products (used or being considered by the 5 apps above)

ST overall has 70 unique software products used by 24 apps and 6 co-design centers

Selected ECP Software Technologies

Programming Models and Runtimes

- MPI: C F N S W
- Kokkos: F W
- RAJA: N S
- CHAI: N S
- Umpire: N S
- … and more

Tools and Technology

- TAU: F W
- HPCToolkit: C F N S
- Flux: N
- Caliper: N S
- PAPI: C F S

Compilers and Support

- LLVM: C F N S W
- OpenMP: C F N W
- … and more

Math Libraries (xSDK)

- hypre: N S W
- PETSc/TAO: F N S
- STRUMPACK: F
- SuperLU: F S W
- Trilinos: F S W
- SUNDIALS: N S
- ArborX: C W
- FFT: C W
- BLAS, LAPACK: F W
- MFEM: N
- … and more

Visualization Analysis and Reduction

- ALPINE: C N S W
- Cinema: C N
- VTK-m: C F N S W
- SZ: C
- zfp: F N
- SPOT: N
- … and more

Data Mgmt, I/O, Checkpoint Restart

- SCR: N
- MPI-IO: F
- HDF5: C F N S W
- PnetCDF: W
- ADIOS: F W
- UnifyFS: N
- VeloC: C
- … and more

Ecosystem: E4S at large

- Spack: F N W
- … and more

Legend: Prog Models & Runtimes · Tools · Math Libraries · Data and Viz · Ecosystems and Delivery

See E4S.io for more ST products

AID
AML
BEE
Darshan
DTK
Dyninst
FleCSI
ForTrilinos
GASNet
Ginkgo
Kokkoskernels
Legion
libEnsemble
MarFS
NRM
OpenACC
Papyrus
PaRSEC
PDT
PowerStack
ScaLAPACK
SCR
SICM
SLATE
SWIG
Tasmanian
Umap
UPC++

ECP apps rely on multiple software technologies; some software products contribute to multiple distinctly developed components of a multiphysics app (such as fusion energy modeling) that must run within a single executable.

EXASCALE COMPUTING PROJECT

ST's Extreme-Scale Scientific Software Stack (E4S) is a Key ECP Product to Sustain and Evolve

- E4S: HPC software ecosystem – a curated software portfolio
- A **Spack-based** distribution of software tested for interoperability and portability to multiple architectures
- Available from **source**, **containers**, **cloud, binary caches**
- Leverages and enhances SDK interoperability thrust
- Not a commercial product – an open resource for all
- Growing functionality: August 2023: E4S 23.08 – 115 full release products

Community Policies Commitment to SW quality	DocPortal Single portal to all E4S product info	Portfolio testing Especially leadership platforms
Curated collection The end of dependency hell	Quarterly releases Release 22.2 – February	Build caches 10X build time improvement
Turnkey stack A new user experience	https://e4s.io	Post-ECP Strategy LSSw, ASCR Task Force

https://spack.io

Spack lead: Todd Gamblin (LLNL)

https://e4s.io

E4S lead: Sameer Shende (U Oregon)

Also includes other products, e.g.,
AI: PyTorch, TensorFlow, Horovod
Co-Design: AMReX, Cabana, MFEM

EXASCALE COMPUTING PROJECT

APPLICATION UPDATE

KPP-1 Definition

- KPP-1 is based on a Figure of Merit (FOM) defined individually for each project to capture the relevant **scientific work rate** for an application.
- Goal of KPP-1 is to measure the overall impact of ECP project, including both hardware-driven and algorithmic improvement.
- Each application measured a **baseline FOM value** at the inception of ECP.

KPP-1 Threshold
50% of KPP-1 applications have a Figure of Merit improvement ≥50

KPP-1 Objective
100% of KPP-1 applications have a Figure of Merit improvement ≥50

- KPP-1 is calculated as the ratio of the FOM **on the exascale challenge problem** to the baseline

$$KPP\text{-}1 = \frac{FOM_{exascale}}{FOM_{baseline}}$$

- The FOM ratio is measured throughout the project to track progress.
- KPP-1 success is determined by an external SME review at end of project.

EXASCALE COMPUTING PROJECT

KPP-2 Definition

- KPP-2 is based on developing new mission-critical capabilities at exascale. Unlike KPP-1 applications, a well-defined baseline was not available at the inception of ECP.
- To meet KPP-2 an application must successfully execute a capability demonstration of the challenge problem on an exascale platform.
- All KPP-2 challenge problems were externally reviewed and determined to require exascale-level compute resources to execute.

KPP-2 Threshold
50% of KPP-2 applications can execute their exascale challenge problem

KPP-2 Objective
100% of KPP-2 applications can execute their exascale challenge problem

- KPP-2 success is determined by an external SME review at end of project. KPP-2 projects must
  - Demonstrate all new capability in place to meet challenge problem specification and utilize full exascale machine
  - Demonstrate reasonably efficient port to exascale machine (uses all accelerator nodes, etc.)
  - Execute demonstration calculation on target exascale platform.

EXASCALE COMPUTING PROJECT

Status of KPP-1 Applications on Frontier

Threshold: 6/11

KPP in progress	KPP completed	KPP submitted	KPP signed (SME)	KPP signed (FPD)
NWChemEx	LatticeQCD	EXAALT	WarpX	ExaSMR
QMCPACK	E3SM-MMF			WDMApp
	CANDLE			ExaSky
				EQSIM

Status of KPP-2 Applications on Frontier

Threshold: 5/10

KPP in progress	KPP completed	KPP submitted	KPP signed (SME)	KPP signed (FPD)
ExaWind	GAMESS	ExaAM		Combustion-PELE
ExaStar	ExaSGD	ExaBiome		MFIX-Exa
Subsurface				
ExaFEL				

ECP Positioned Applications for Long-term Technical Viability

Several factors contributed to this:

- **Exascale challenge problem targets**: by setting ambitious performance and capability targets using real science calculations, teams were always thinking in terms of realistic use cases rather than toy problems or benchmarks.
- **Access to Software Technology**: building upon modular, well-designed software components significantly simplifies the maintenance burden going forward.
- **Access to Hardware Integration**: close coordination with the vendors and Facilities ensured that teams gained an in-depth understanding of how their codes perform in practice and helped with the adoption of portable programming models.
- **Access to Exascale machines**: by getting substantial resources on the most advanced supercomputers in the world, teams are uniquely ready to take advantage of future resources.

By the end of the project, most ECP application codes will be significantly more robust, portable and maintainable than they would have without ECP.

ECP: The whole was indeed greater than the sum of the parts

Jordan Spieth, The Open Championship (Royal Birkdale, Jul 23 2017)

Questions?

https://www.exascaleproject.org/contact-us/

For more info

- Alexander F. et al. *Exascale Applications: Skin in the Game,* Phil. Trans. R. Soc. A 378: 20190056 (2020) (http://dx.doi.org/10.1098/rsta.2019.0056).
- Douglas Kothe, Stephen Lee, and Irene Qualters, *Exascale Computing in the United States,* Computing in Science and Engineering 21(1), 17-29 (2019).