

# Migrating from Cori to Perlmutter



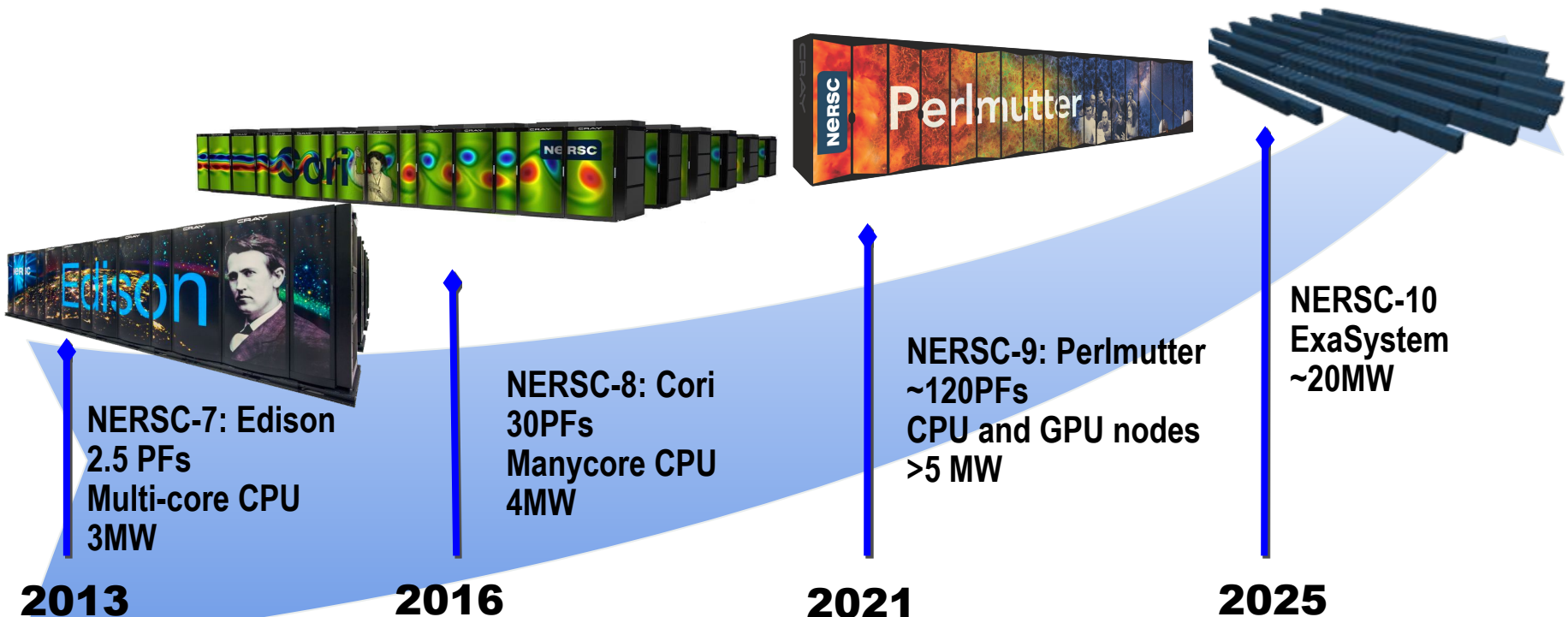
## Intro to Perlmutter and GPUs

### Overview

December 2022

**Jack Deslippe**  
Application Performance Lead

# NERSC Systems Roadmap



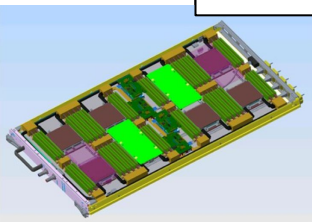
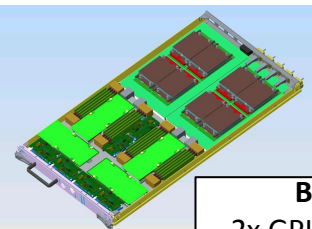
# Perlmutter system configuration

## NVIDIA "Ampere" GPU Nodes

4x GPU + 1x CPU (>75 TF)  
160 GiB HBM + DDR  
4x 200G "Slingshot" NICs

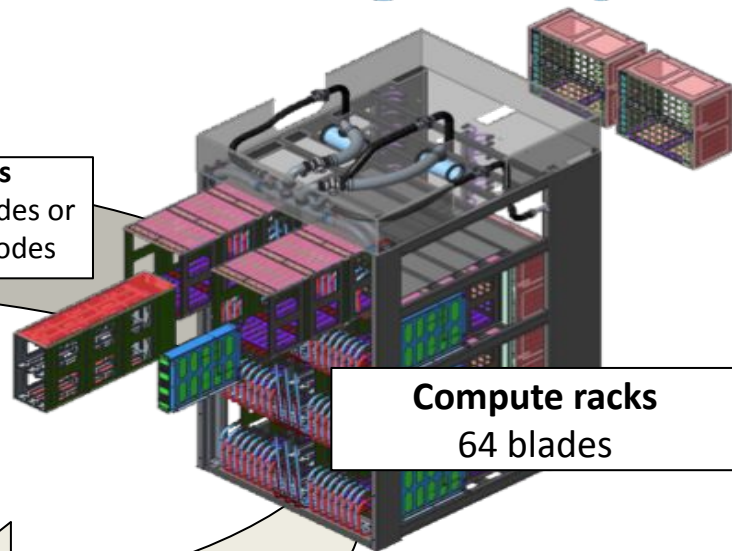
## AMD "Milan" CPU Node

2x CPUs  
> 256 GiB DDR4  
1x 200G "Slingshot" NIC



### Blades

2x GPU nodes or  
4x CPU nodes



### Compute racks

64 blades

### Centers of Excellence

Network  
Storage  
App. Readiness  
System SW

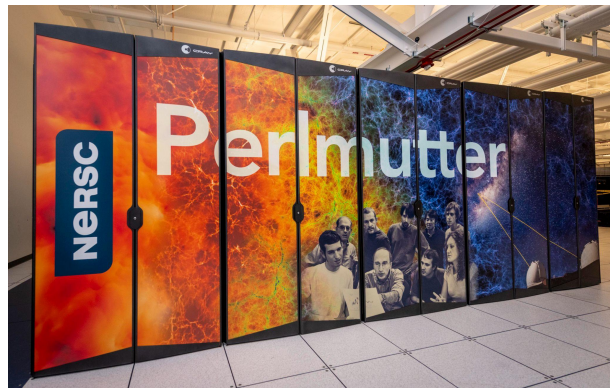
### Perlmutter system

GPU racks  
CPU racks  
~6 MW





# The System



## System Specifications

Partition	# of nodes	CPU	GPU	NIC
GPU	1536	1x <a href="#">AMD EPYC 7763</a>	4x <a href="#">NVIDIA A100</a> (40GB)	4x <a href="#">HPE Slingshot 11</a>
CPU	3072	2x <a href="#">AMD EPYC 7763</a>	-	1x <a href="#">HPE Slingshot 11</a>
Login	40	1x <a href="#">AMD EPYC 7713</a>	1x <a href="#">NVIDIA A100</a> (40GB)	-
Large Memory	4	1x <a href="#">AMD EPYC 7713</a>	1x <a href="#">NVIDIA A100</a> (40GB)	1x <a href="#">HPE Slingshot 11</a>

## System Performance

Partition	Type	Aggregate Peak FP64 (PFLOPS)	Aggregate Memory (TB)
GPU	CPU	3.9	384
GPU	GPU	59.9 tensor: 119.8	240
CPU	CPU	7.7	1536

More Details:

<https://docs.nersc.gov/systems/perlmutter/architecture/>

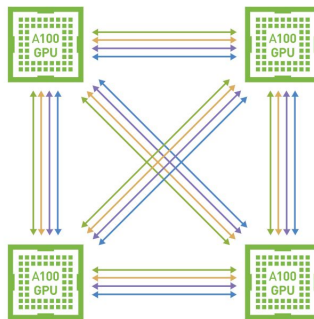
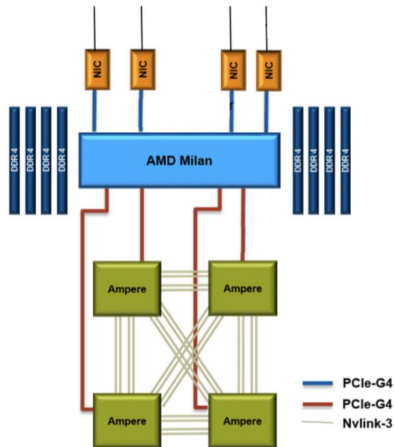


# The System

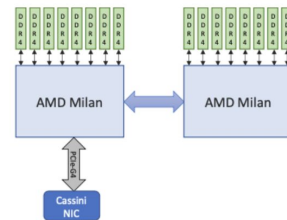
## GPU Nodes:

- Single [AMD EPYC 7763](#) (Milan) CPU
- 64 cores per CPU
- Four [NVIDIA A100](#) (Ampere) GPUs
- PCIe 4.0 GPU-CPU connection
- PCIe 4.0 NIC-CPU connection
- 4 [HPE Slingshot 11](#) NICs
- 256 GB of DDR4 DRAM
- 40 GB of HBM per GPU with
- 1555.2 GB/s GPU memory bandwidth
- 204.8 GB/s CPU memory bandwidth
- 12 third generation NVLink links between each pair of gpus
- 25 GB/s/direction for each link

Data type	GPU TFLOPS
FP32	19.5
FP64	9.7
TF32 (tensor)	155.9
FP16 (tensor)	311.9
FP64 (tensor)	19.5



## CPU Nodes:



- 2x [AMD EPYC 7763](#) (Milan) CPUs
- 64 cores per CPU
- AVX2 instruction set
- 512 GB of DDR4 memory total
- 204.8 GB/s memory bandwidth per CPU
- 1x [HPE Slingshot 11](#) NIC
- PCIe 4.0 NIC-CPU connection
- 39.2 GFlops per core
- 2.51 TFlops per socket
- 4 NUMA domains per socket (NPS=4)

# The System

## All Flash Filesystem:

- 35 PB of disk space
- an aggregate bandwidth of >5 TB/sec
- 4 million IOPS (4 KiB random)
- It has 16 MDS (metadata servers)
- 274 I/O servers called OSSs
- 3,792 dual-ported NVMe SSDs.

# Our Common Challenge

---



Enable a diverse community of scientific users and codes to run efficiently on advanced architectures like Cori, Perlmutter and beyond

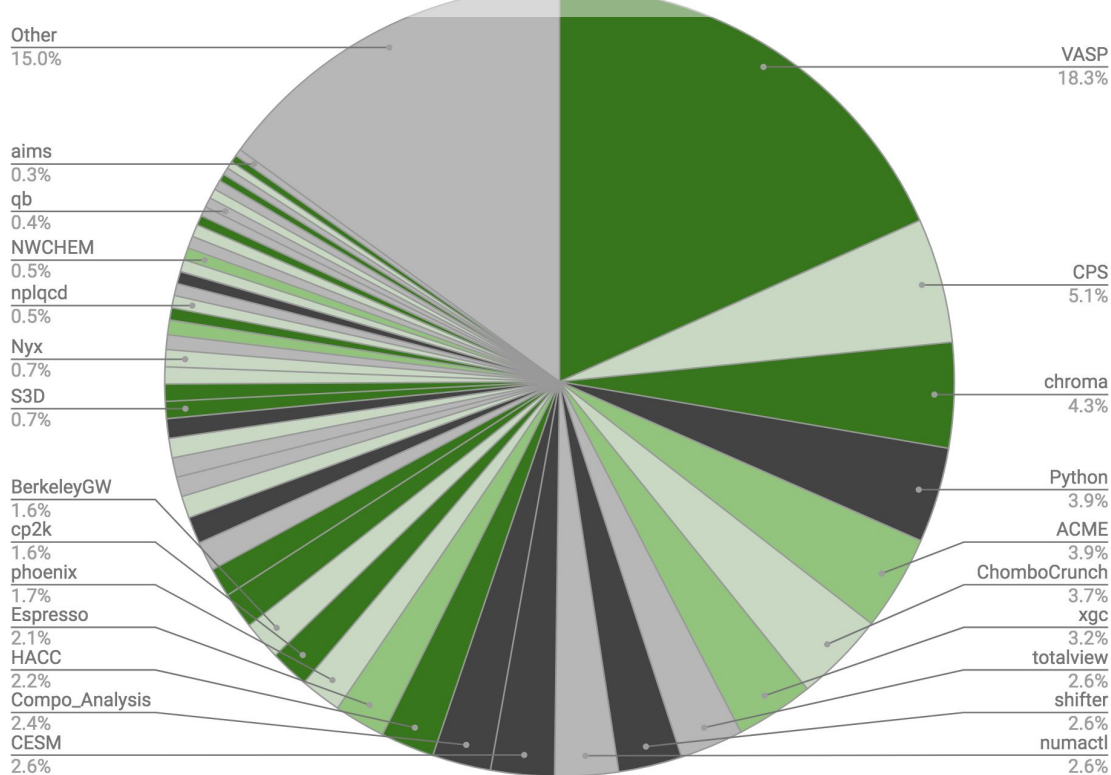


# Comparison of Perlmutter and Cori

Attribute	Cori (2016)	Perlmutter (2021)
Peak Performance	~30 PF	~120 PF
Peak Power	< 4MW	~6 MW
System Memory	~ 1PB (DDR4 + HBM)	> 2PB (DDR4 + HBM)
Node Performance	> 3 TF	> 70 TF
Node Processors	Intel KNL + Intel Haswell	AMD EPYC (Milan) + Nvidia A100 GPUs
# of Nodes	9300 KNL + 1900 Haswell	1536 GPU Accelerated + 3072 CPU only
Intra-Node Interconnect	N/A	NVLink across GPUs; PCIe
Inter-Node Interconnect	Aries	Slingshot
Filesystem	28 PB, 0.75 TB/s	35PB All-Flash; > 4TB/s

# GPU Readiness Among NERSC Codes (Aug'17 - Jul'18)

Breakdown of Hours at NERSC



GPU Status & Description	Fraction
<b>Enabled:</b> Most features are ported and performant	32%
<b>Kernels:</b> Ports of some kernels have been documented.	10%
<b>Proxy:</b> Kernels in related codes have been ported	19%
<b>Unlikely:</b> A GPU port would require major effort.	14%
<b>Unknown:</b> GPU readiness cannot be assessed at this time.	25%

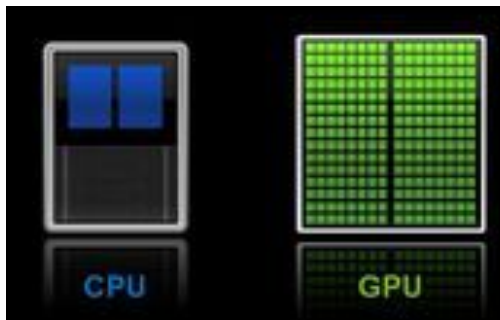
**A number of applications in NERSC workload were GPU enabled already.**

**We leveraged existing GPU codes from CAAR + Community**

# NESAP Motivation: CPUs vs GPUs

## CPU (Haswell)

- 64 cores
- 2 threads each
- 2x256-bit vectors
- double precision
  - ~2000 way parallelism ( $64 \times 4 \times 8$ )



## GPU (A100)

- 108 SM
- Up to 64 warps per SM  
(2 active at a time)
- 32 **SIMT** per warp
- double precision
  - 200,000+ way parallelism ( $108 \times 64 \times 32$ )

CPU - Speed



GPU - Throughput

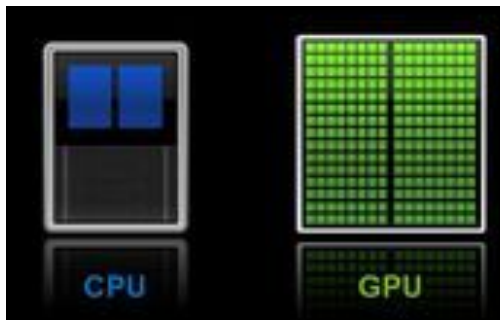




# NESAP Motivation: CPUs vs GPUs

## CPU (Haswell)

- 64 cores
- 2 threads each
- 2x256-bit vectors
- double precision
  - ~2000 way parallelism ( $64 \times 4 \times 8$ )



## GPU (A100)

- 108 SM
- Up to 64 warps per SM  
(2 active at a time)
- 32 **SIMT** per warp
- double precision
  - 200,000+ way parallelism ( $108 \times 64 \times 32$ )

CPU - Speed



GPU - Throughput

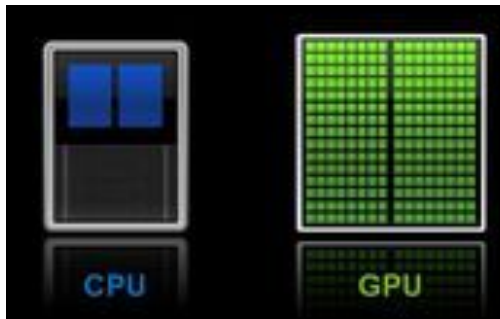


Oversubscribing GPUs  
(w/ Warps and  
Streams) helps hide  
latency, too!

# GPUs vs. CPUs Memory Bandwidth

## CPU (Haswell)

- 128GB DDR
- ~120 GB/Sec Memory Bandwidth



## GPU (A100)

- 40GB HBM
- 1,500 GB/Sec Memory Bandwidth



PCIe ~ 32 GB/Sec

CPU - Speed



GPU - Throughput



Try to avoid moving data back and forth frequently

# Challenge - There are Multiple GPU Optimization Avenues

1. You Need Orders of Magnitude More Parallelism
2. A100 GPU Memory is Very Fast. But, moving data to the GPU is Not.

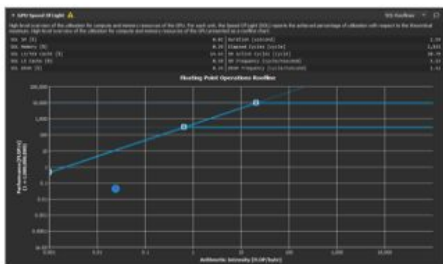
## Other Second Order Considerations:

3. There is some overhead in launching kernels. Fusing short kernels together and defining “CUDA Graphs” can help.
4. HBM is fast, but keeping data in registers, cache and “shared” memory is better! Find optimal balance between maximizing parallelism and minimizing register spills.



# Determining Which Optimizations to Pursue

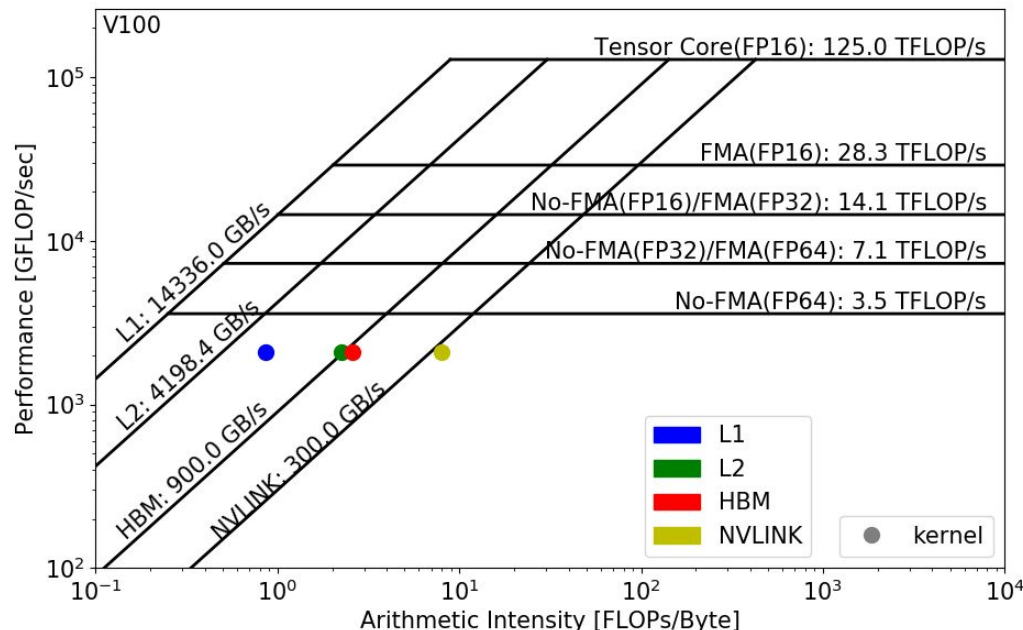
## Co Designing-Vendor Tools



Roofline Analysis



NERSC worked closely with NVIDIA to design speed of light and roofline modeling in NVIDIA's NSIGHT profiler.



# NESAP Strategy

**NESAP** is NERSC's Application Readiness Program for preparing our workload for new systems.

**Strategy:** Partner with application development teams and vendors to port & optimize key applications of importance to the Office of Science. Share lessons learned with with NERSC community via documentation and training.

**Resource Available to Teams:** NERSC Staff technical liaisons, performance postdocs, access to vendor application engineers, hackathons, early access to hardware (GPU nodes on Cori and Perlmutter)

**Simulation:** 14 application teams

**Data:** 9 applications

**Learning:** 5 applications

# This was an all hands on deck activity!



Jack Deslippe  
(NESAP Lead)



Brandon Cook  
(Simulation Area Lead)



Johannes Blaschke  
(Data Area Lead)



Steve Farrell  
(Learning Area Lead)



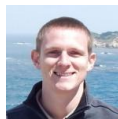
Lisa Gerhardt  
(User Int lead)



Hannah Ross



Rahul Gayatri



Chris Daley



Phillip Thomas



Amanda Dufek



Bill Arndt



Wahid Bhimji



Bjoern Enders



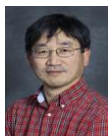
Muaaz Awan



Lisa Claus  
(Math Libs)



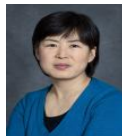
Paul Lin  
(Early Science Lead)



Woo-Sun Yang



Neil Mehta



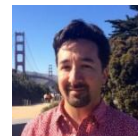
Zhengji Zhao



Helen He  
Training



Kevin Gott  
(Hackathons)



Rollin Thomas  
(Jupyter)



Laurie Stephey  
(Python)



Stephen Leak  
(PE Lead)



# Hackathons Have Been Effective for Reaching Code Teams



“Hackathons” have proven to be a highly effective tool for preparing applications for new architectures.

## NERSC Supports Two Types of Hackathons:

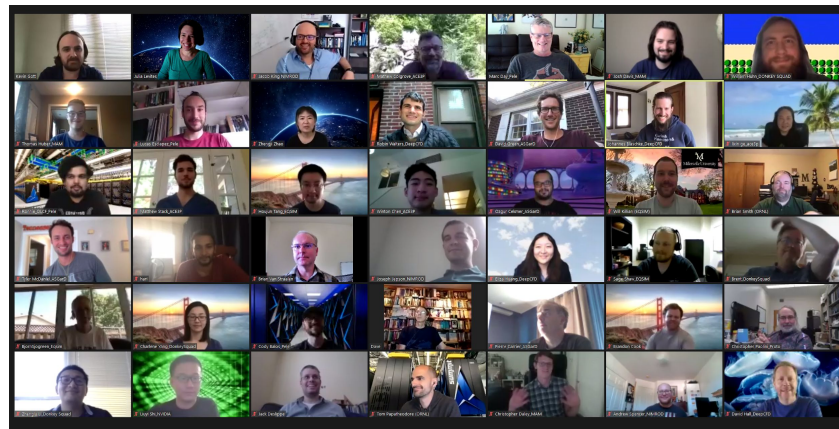
### 1. **Private** N9 Project COE Hackathons

Quarterly with 2-3 NESAP teams + Cray and NVIDIA engineer support. **Project controlled and focused on FOM optimizations.** 12 conducted!

### 2. **Public** GPU Hackathons

(<https://www.gpuhackathons.org>) **NERSC provided more team mentors than any other institution to worldwide events.**

Allows us to reach NERSC teams all around the country and world - **amplifies NESAP impact to the broad NERSC workload.**



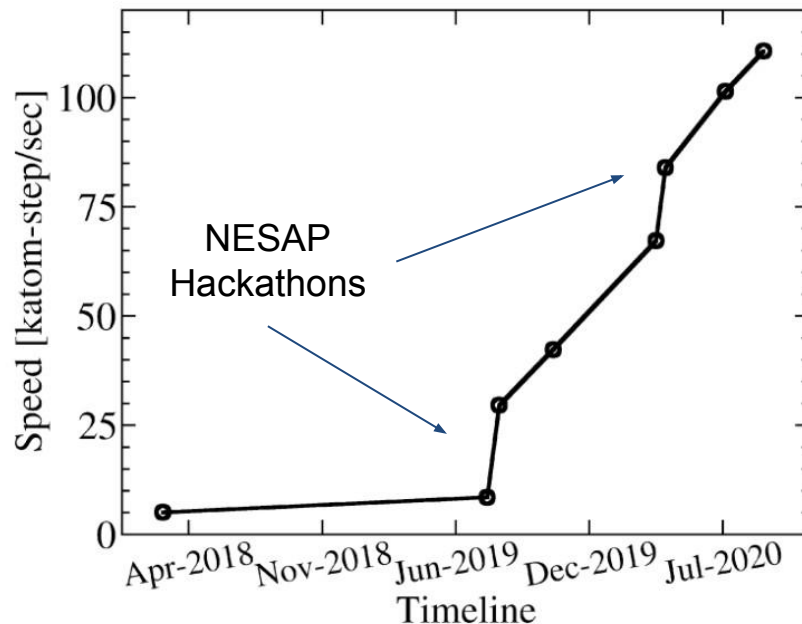
NERSC adapted the hackathon format for the COVID work-from-home environment. Instead of on-site, multiple-day, full-day sessions, we moved to a series of shorter sessions spread out over 6-8 weeks. Some features of this format were popular and effective and we plan to incorporate them into future hackathons.

# NESAP Had a Big Impact on Applications (Example)

## LAMMPS

- LAMMPS is a classical molecular dynamics code with a focus on materials modeling
- Production LAMMPS Kokkos version was highly optimized over a series of hackathons (**joint effort of NERSC/NESAP, ECP, NVIDIA and HPE**)
- Every kernel was rewritten and optimized individually
- SSI (system-wide throughput increase over Edison) in atom-steps/second

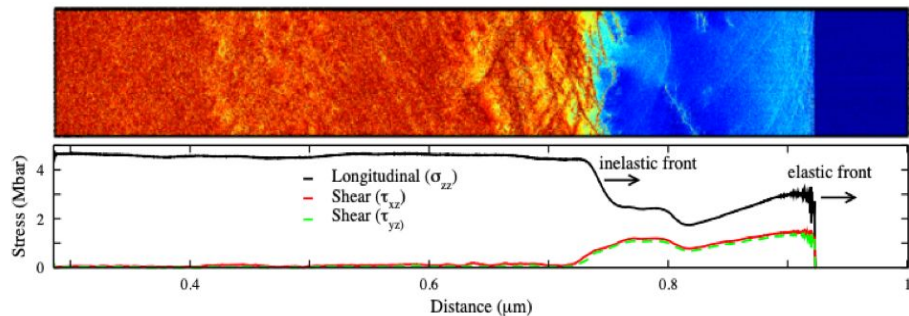
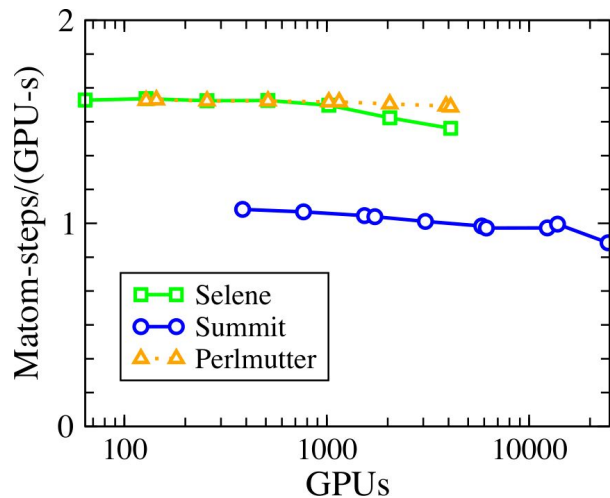
SSI: 35.3  
Node vs Node Speedup: 171



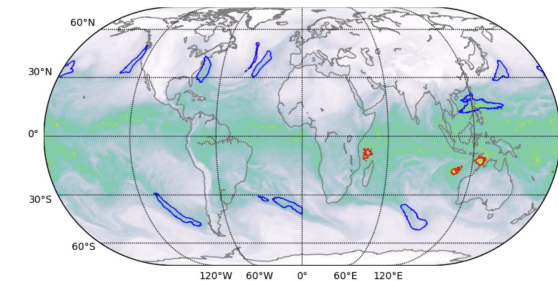
# Record Scale MD With LAMMPS

## Gordon Bell Finalists

- Collaborative effort: University of South Florida, Sandia, NERSC and NVIDIA
- Billion atom molecular dynamics simulation (20B atoms)
  - SNAP quantum-accurate machine learned interatomic potential
  - Kokkos CUDA backend for NVIDIA GPUs
- Simulation model shock compression of carbon at extreme pressures and temperatures.



# NERSC Staff Gordon Bell Finalists/Winners



**2018 (Winners):**  
Climate/AI

Thorsten Kurth, Jack Deslippe, Mr. Prabhat

**2020 (Finalists):**  
Materials

Charlene Yang, Mauro Del Ben, Jack Deslippe

**2021 (Finalists):**  
Molecular Dyn./AI

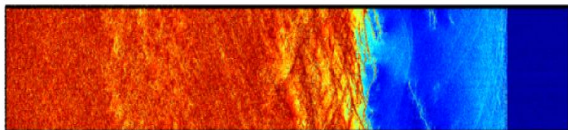
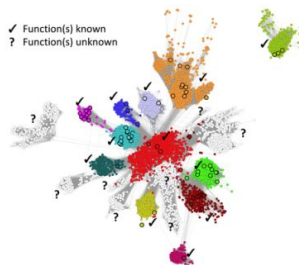
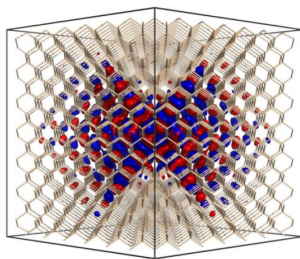
Rahul Gayatri

**2022 (Finalists):**  
Bio-Informatics

Muaaz Awan

**2022 (Winner):**  
Accelerator Physics

Kevin Gott



# Observations

- Many applications have been successful in preparing for Perlmutter
- We'd like to keep engaging with the broad NERSC community to enable it to use Perlmutter productively
  - We are continuing to encourage everyone to join community hackathons at [GPUHackathons.org](https://www.gpuhackathons.org) – events all over the country in the next year
- Multiple GPU optimization angles exist. Profiling and roofline modeling are key to determining optimization paths.
- The scientific community is motivated to optimize their codes for GPUs.

# Perlmutter Supports Every GPU Programming Model

	Fortran/ C/C++	CUDA	OpenACC 2.x	OpenMP 5.x	CUDA Fortran	Kokkos / Raja	MPI	HIP	DPC++ / SYCL
NVIDIA									
CCE									
GNU									
LLVM									

Vendor  
Supported

NERSC  
Supported



# The System Has a Robust Programming Environment



## Debuggers

DDT

GDB (CUDA-GDB)

## Profilers

NSIGHT

CrayPat

Tensorboard

# Getting started with GPUs in Python

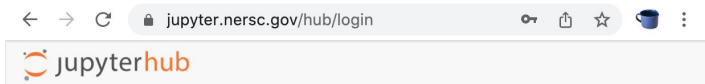
- NumPy and SciPy do not utilize GPUs out of the box
- There are many Python GPU frameworks out there:
  - “drop in” replacements for numpy, scipy, pandas, scikit-learn, etc
    - **CuPy**, **RAPIDS**, **cuNumeric** (coming soon?)
  - “machine learning” libraries that also support general GPU computing
    - **PyTorch**, **TensorFlow**
  - “I want to write my own GPU kernels”
    - **Numba**, **PyOpenCL**, **PyCUDA**
- Many of these GPU libraries have adopted the [CUDA Array Interface](#) which makes it easier to share array-like objects stored in GPU memory between the libraries
- There is also some effort in the community to standardize around a common [Python array API](#)



```
numpy:      mean(a, axis=None, dtype=None, out=None, keepdims=<no value>)
dask.array: mean(a, axis=None, dtype=None, out=None, keepdims=<no value>)
cupy:      mean(a, axis=None, dtype=None, out=None, keepdims=False)
jax.numpy: mean(a, axis=None, dtype=None, out=None, keepdims=False)
mxnet.np:  mean(a, axis=None, dtype=None, out=None, keepdims=False)
sparse:    s.mean(axis=None, keepdims=False, dtype=None, out=None)
torch:     mean(input, dim, keepdim=False, out=None)
tensorflow: reduce_mean(input_tensor, axis=None, keepdims=None, name=None,
                        reduction_indices=None, keep_dims=None)
```

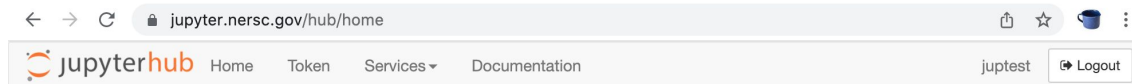
# How do I run Jupyter notebooks on Perlmutter

<https://jupyter.nersc.gov>



Log In

<https://jupyter.nersc.gov/hub/home> (home or “console”)



	Shared CPU Node	Shared GPU Node	Exclusive CPU Node	Exclusive GPU Node	Configurable GPU
Perlmutter	<a href="#">start</a>			<a href="#">start</a>	<a href="#">start</a>
Cori	<a href="#">start</a>	<a href="#">start</a>	<a href="#">start</a>		<a href="#">start</a>
Resources	Use a node shared with other users' notebooks but outside the batch queues.		Use your own node within a job allocation using defaults.		Use multiple compute nodes with specialized settings.
Use Cases	Visualization and analytics that are not memory intensive and can run on just a few cores.		Visualization, analytics, machine learning that is compute or memory intensive but can be done on a single node.		Multi-node analytics jobs, jobs in reservations, custom project charging, and more.

Note: Your console may look a little different if you don't have Cori GPU access for instance

# The Programming Environment Supports All Major GPU Prog. Models

- The NVHPC, GCC, CCE and LLVM (Clang) compilers provide additional support for applications. NERSC Staff are part of ECP Flang (in testing)
- **OpenMP 4.5/5.x** support has been enabled in NVHPC through NERSC NRE
- NERSC is a member of the **OpenACC** board. Supported in multiple compilers.
- **DPC++** supported on Perlmutter enabled via CodePlay NRE (LLVM)
- NERSC participating in ECP **HIP** project - Enabling/Testing on Perlmutter
- **Kokkos** works well on the system. Multiple NERSC Staff members are part of the ECP Kokkos team.

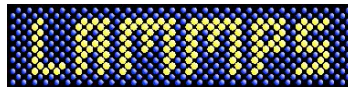
# Broad impact and enablement

Perlmutter Supports all Major Programming models and languages

kokkos



Pre-installed/Optimized Community Codes



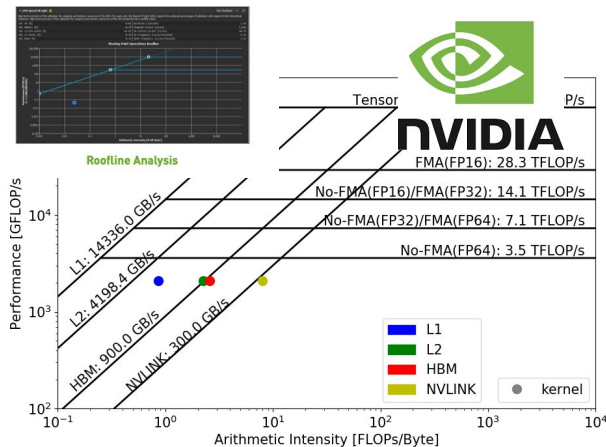
Community Resources

NERSC

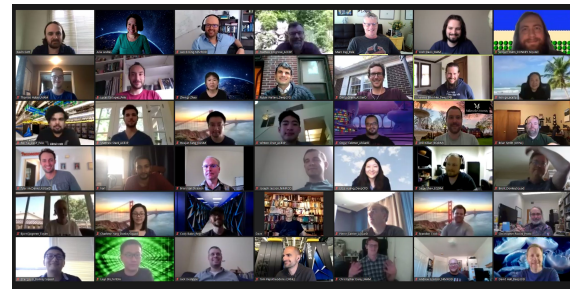
NERSC Documentation

NERSC TRAINING EVENTS

Co Design-Vendor Tools



Community GPU hack-a-thons

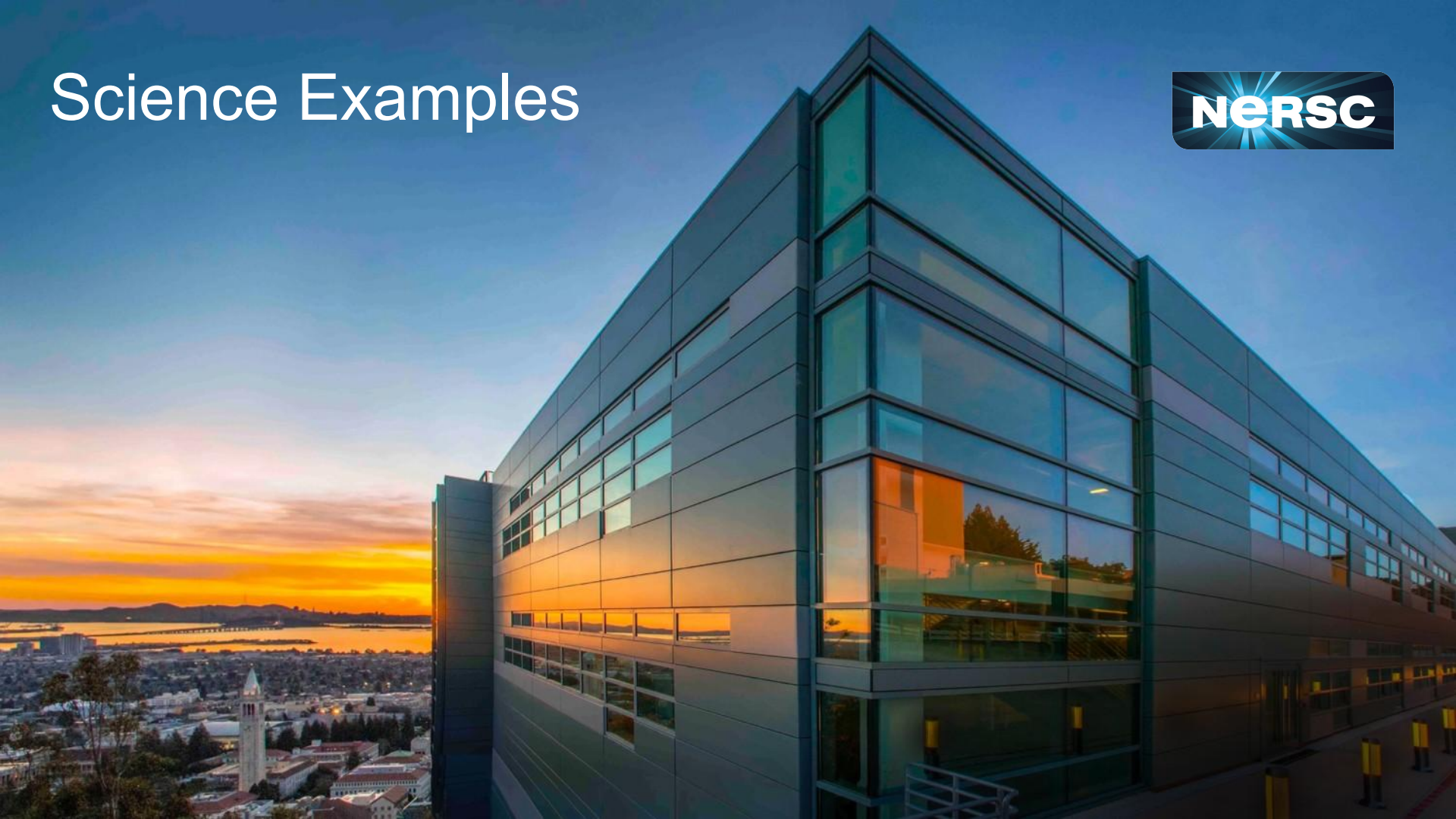


# NERSC Invested in Performance Portable Programming with OpenMP & DPC++ NRE (Off Project)

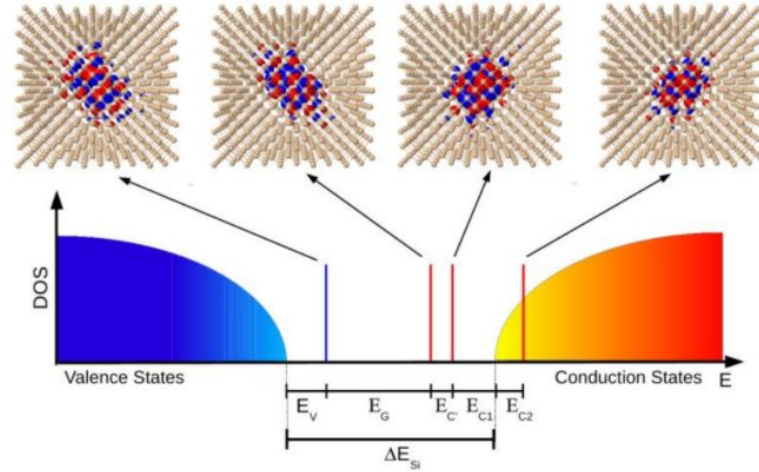
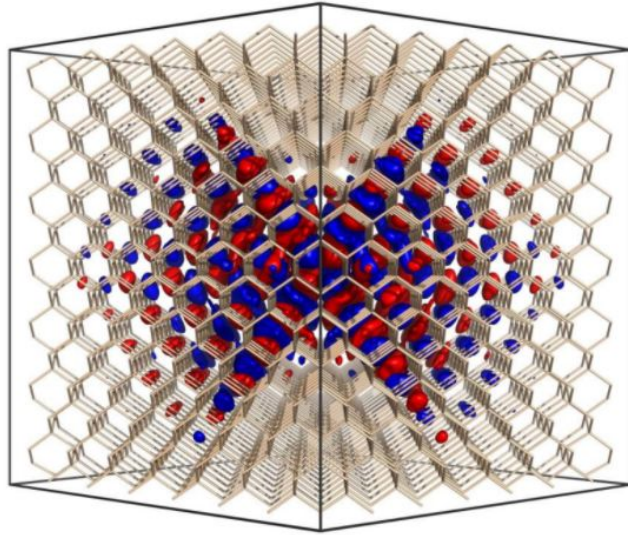
- **OpenMP 4.5/5.x** support has been enabled in NVHPC through NERSC NRE  
Multi-Year NRE that Utilized NESAP and ECP applications for testing  
Settled on a well-defined subset of the OpenMP standard for optimized GPU acceleration  
Released in production NVHPC SDK
- **DPC++** supported on Perlmutter enabled via CodePlay NRE (LLVM)  
Multi-Year NRE in collaboration with ALCF to enable optimized DPC++ execution on A100.  
Based on Open-Source LLVM and available to use on Perlmutter



# Science Examples



# Qubit Design With BerkeleyGW



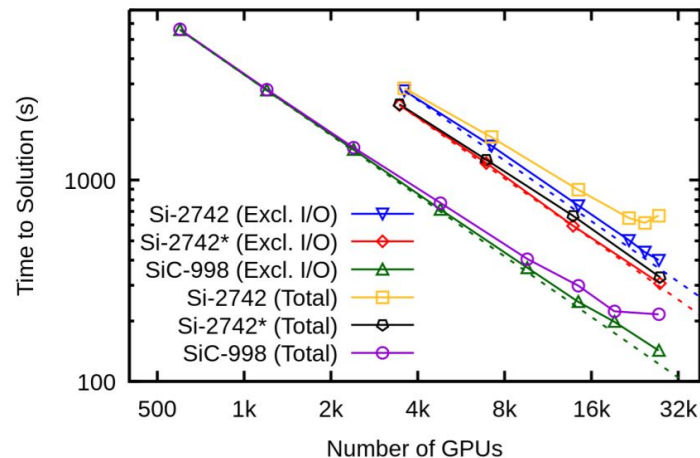
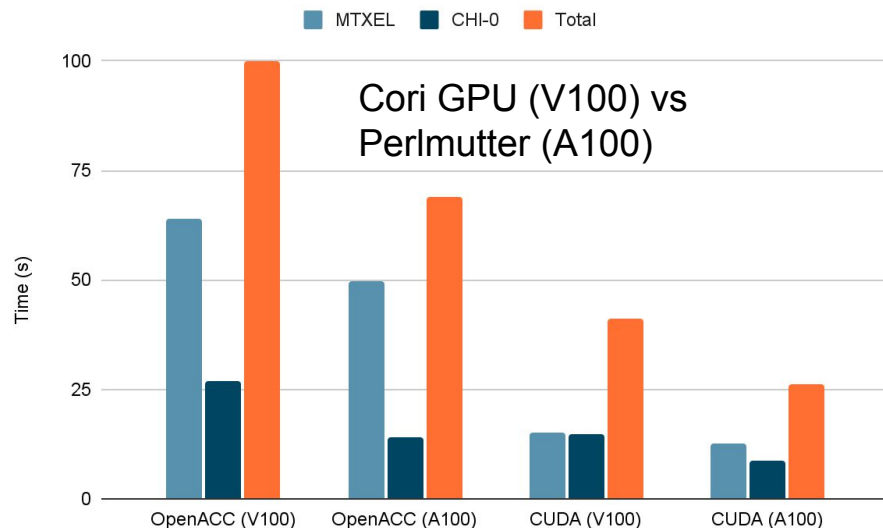
Example: Divacancy point defect in crystalline silicon, prototype of a solid-state Qubit

Accurate prediction requires:

- Accuracy beyond DFT: **GW** and **GW+BSE**
- Unprecedented simulation sizes: **1000's of atoms**

# Qubit Design

The BerkeleyGW NESAP team was recognized as a Gordon Bell finalist in 2020.



	MTXEL	CHI-0	Total
OpenACC (V100)	64	27	100
OpenACC (A100)	49.8	14.2	69
CUDA (V100)	15.2	14.7	41
CUDA (A100)	12.6	8.7	26.2

- Si-214 system (scaled: 4Ry CT ; 3000 bands). 8 GPUs each.



**BERKELEY LAB**  
Bringing Science Solutions to the World

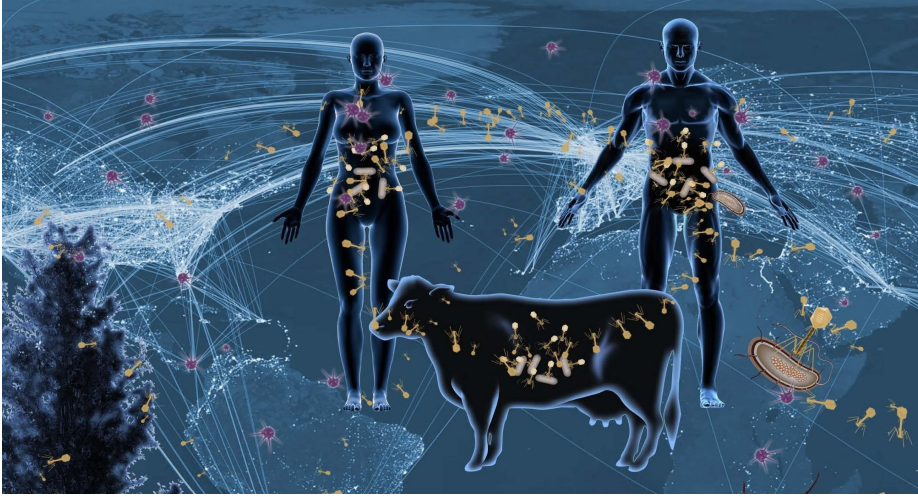


U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science



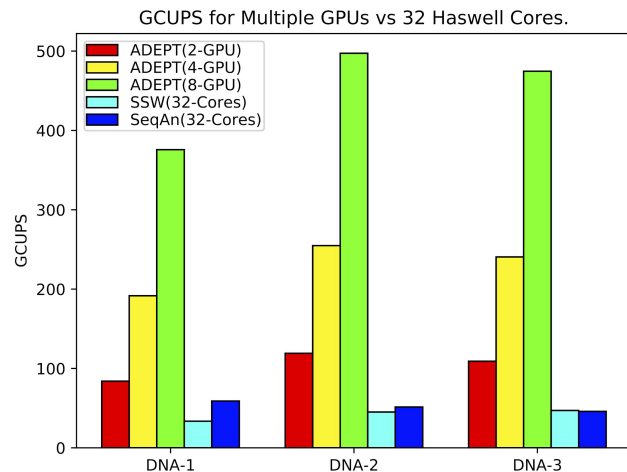
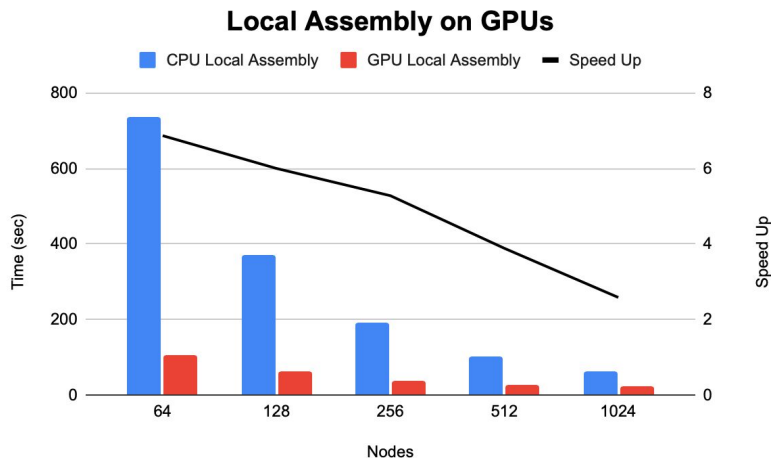
# Exabiome (Meta-Genomics)



- **Microbes:** these are single cell organism, e.g. viruses, bacteria
- **Microbiomes:** communities of microbial species living in our environment.
- **Metagenomics:** genome sequencing of these communities.

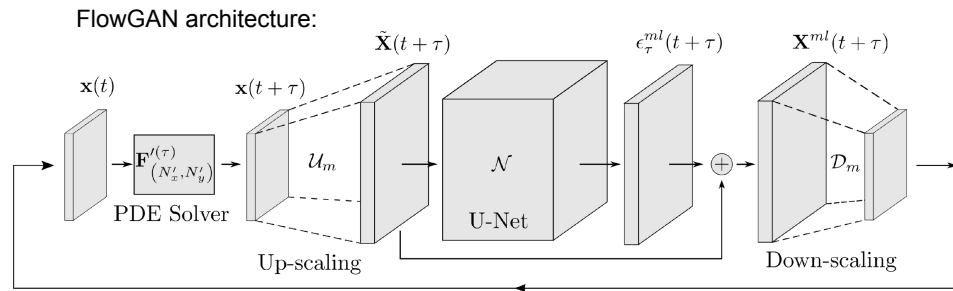
# Exabiome (Meta-Genomics)

- A lot of progress has been made on GPU algorithms for meta-genomics.
- This NESAP team wrote the world's fastest GPU aligners using a lot of clever strategies, newly available GPU intrinsic instructions etc.
- With the help of warp level intrinsics, dynamic data structures were written for GPUs from scratch to re-write the Local Assembly stage.



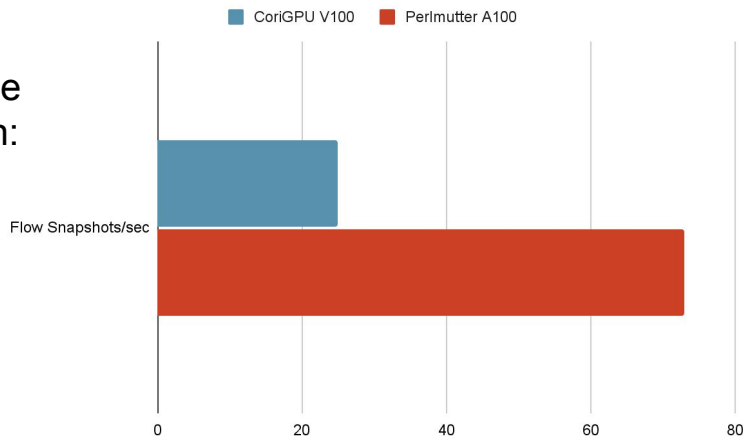
# Accelerating CFD with GANs on Perlmutter

The FlowGAN project introduces a technique based on a deep neural network architecture to augment traditional numerical simulations of fluid flows. The ML model is used to correct the numerical errors induced by a coarse-grid simulation of turbulent flows at high-Reynolds numbers.

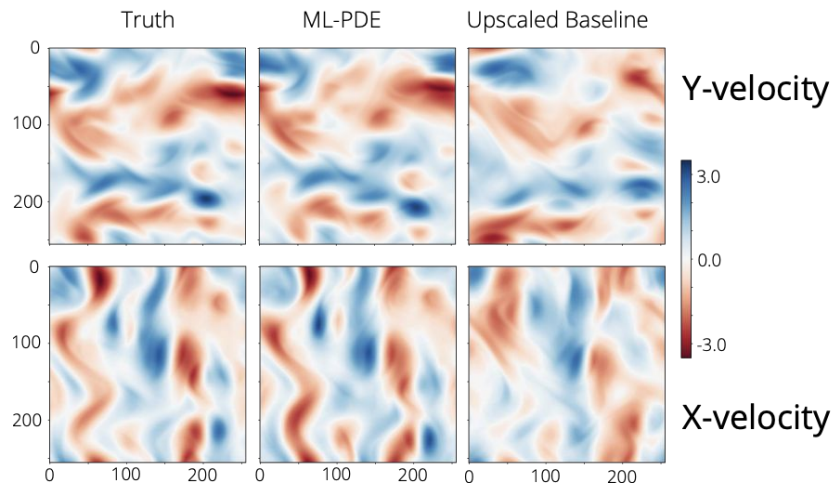


## Performance Comparison:

2.9x  
performance  
improvement  
over CoriGPU  
on ML  
workflow



$t = 10$  model time units

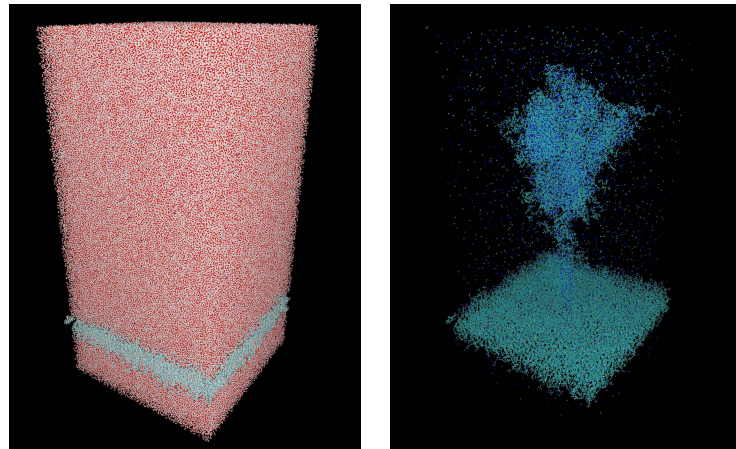




# Exaop Performance for the *Ab-Initio* Molecular Dynamics

Ground-breaking real-world exaop calculation in mixed FP16/32 run on Perlmutter

- The non-orthogonal local submatrix method applied to electronic-structure based molecular dynamics simulations exceeds 1.1 EFLOP/s in FP16/FP32 mixed floating-point arithmetic
- Used 4,400 NVIDIA A100 GPUs on Perlmutter
- The method achieves a sustained fraction of peak performance of about 80%.
- Example calculations are performed for SARS-CoV-2 spike proteins with up to 83 million atoms.



SARS-CoV-2 spike protein in aqueous solution: full cell (left) and without hydrogen and oxygen atoms (right).

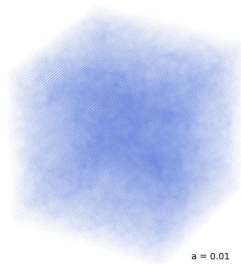
*Robert Schade, Tobias Kenter, Hossam Elgabarty, Michael Lass, Thomas D. Kühne, Christian Plessl, Paderborn University*

*arXiv:2205.12182v1 24 May 2022*

# Early Successes in Data/Learning

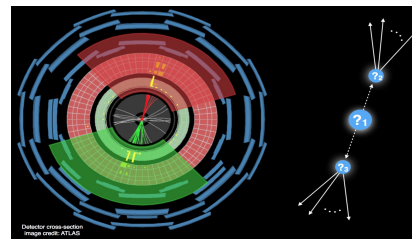
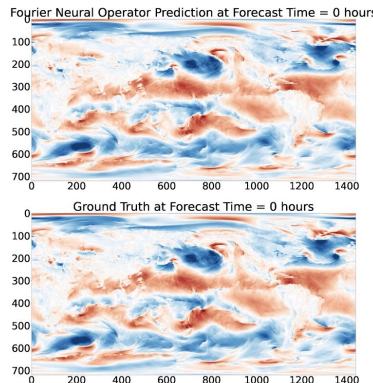
## DESC (Dark Energy Science Collaboration)

- Using GPUs with Tensorflow, via Jupyter, for redshift model-fitting
- Distributed TF at scale on GPU w/ NCCL
- Tested up to  $2048^3$  N-body simulation, distributed on 256 GPUs
- Multiple TB of RAM



## Data-driven Atmospheric Modeling

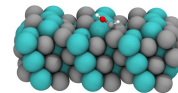
- ML Data-driven prediction of high-resolution atmospheric flow variables
- **2.9x** improvement in throughput using Perlmutter A100 compared to Cori V100 GPUs



## Anomaly Detection, Unfolding & Fast Simulation in Particle Physics

- DL techniques used in searches for fundamental particles at the LHC
- Expanding to more complex models/approaches and higher-fidelity generative networks

## Open Catalyst Project

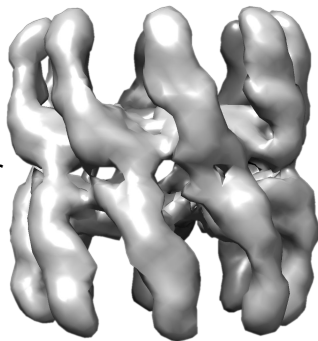


- Deep learning to accelerate catalyst discovery for reactions that are critical for energy storage and climate change mitigation
- Scaling current models from  $O(10-100)$  GPUs to  $O(1000)$  GPUs

# Early Successes in Superfacility

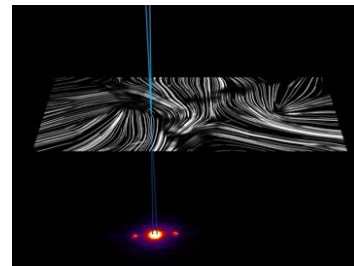
## LCLS

- Reconstruct molecular structure from X-Ray scattering data
- Used Perlmutter for live data processing (ie., determining molecular structures during data collection), enabling real-time steering of the experiment



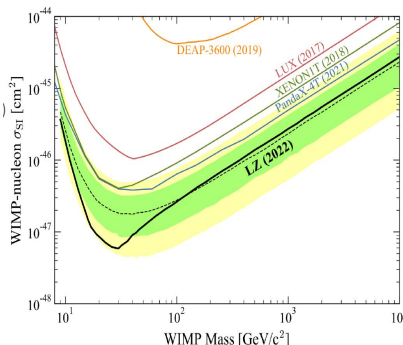
## NCEM

- Multi-TB scale electron microscopy image simulation to train NN for materials research
- VASP to calculate XAS spectra to train a ML model for automated assignment of bond valence
- Real-time processing and reduction of 4D-STEM data with `distiller.lbl.gov`



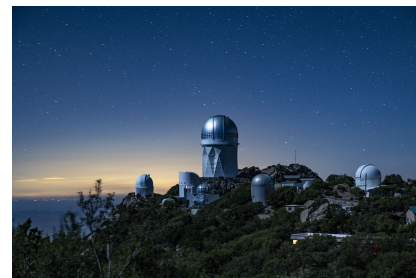
## Lux-Zeplin (LZ)

- Dark matter detection experiment used GPUs for ray tracing in detector simulation
- Used Perlmutter to extract limits on dark matter-nucleon interaction for first science results



## Dark Energy Spectroscopic Instrument

- DESI Spectral Extraction is an image processing code implemented in Python.
- **2.5x** improvement in per-node throughput using Perlmutter A100 compared to Cori V100 GPU (x25 compared to Edison).





# Questions?



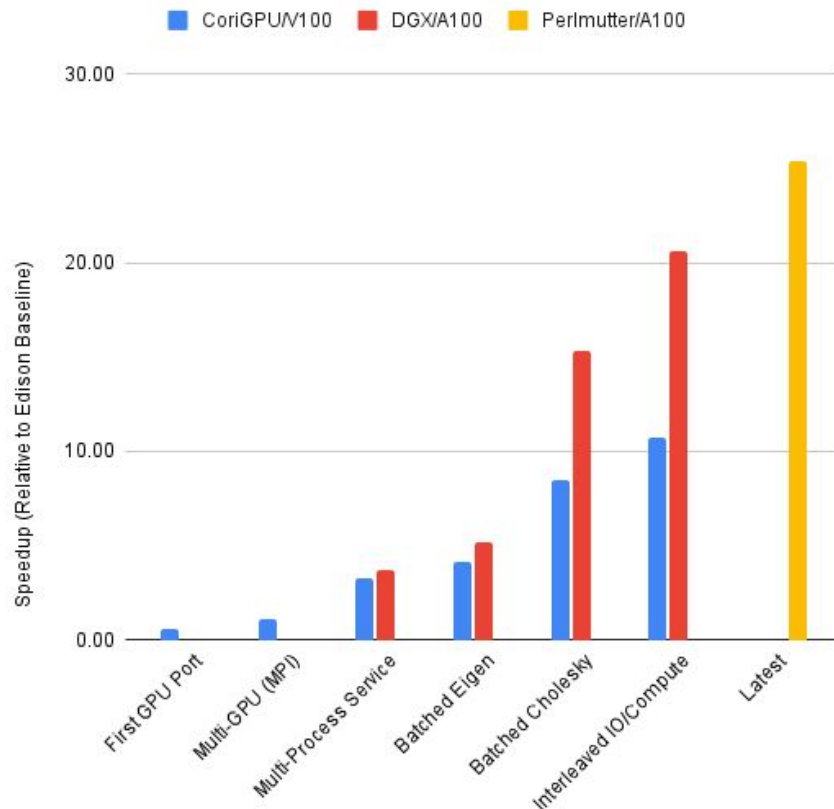


# DESI

## Dark Energy Spectroscopic Instrument

- DESI Spectral Extraction is an image processing code implemented in Python.
- Completed major refactor of optimized CPU code and initial GPU port in early 2020.
- Major optimization milestones include: saturating GPU utilization using MPI and CUDA Multi-Process Service, refactoring code to leverage batched linear algebra operations on GPU, and interleaving IO with computation.
- **25x** improvement in per-node throughput using Perlmutter compared to Edison baseline.

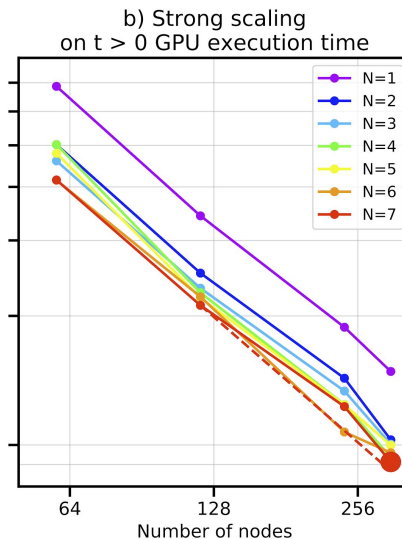
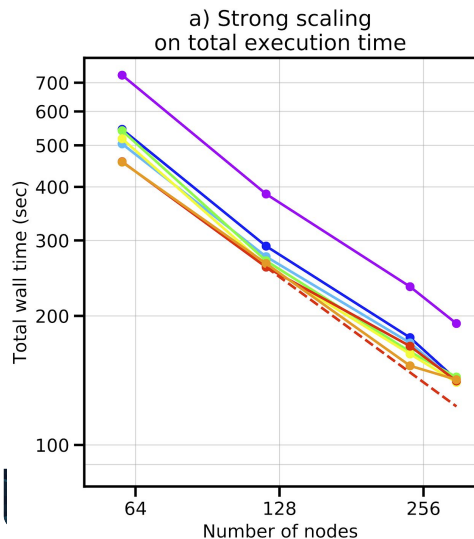
Cumulative Speedup Relative to Edison Baseline



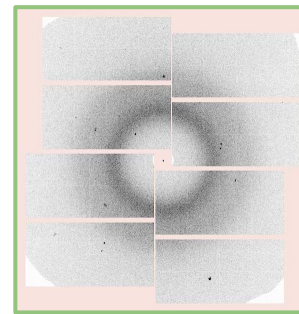
# ExaFEL

XFEL requires **real-time data analysis** to make decisions **during ongoing experiments**. Data collection rates outpacing computational resources at the experimental sites, **requiring a Superfacility approach**.

In two years, NESAP has developed a highly scalable CUDA/GPU application.  
**CCTBX/nanoBragg w/ runtime improved from 12.3 hours on Edison, to 2 minutes**



**CCTBX/nanoBragg** strong scaling on Summit.  
Colored lines show number of concurrent streams per GPU



**BERKELEY LAB**  
Bringing Science Solutions to the World



U.S. DEPARTMENT OF  
**ENERGY**

Office of  
Science



# ExaFEL

NESAP has been essential in **developing a scalable version of the MTIP algorithm** (figure, right). By offloading kernels to CUDA, **MTIP/Spinifel runtime was decreased by 2.4x over CPU-only code**.

Time (s) spent in different modules

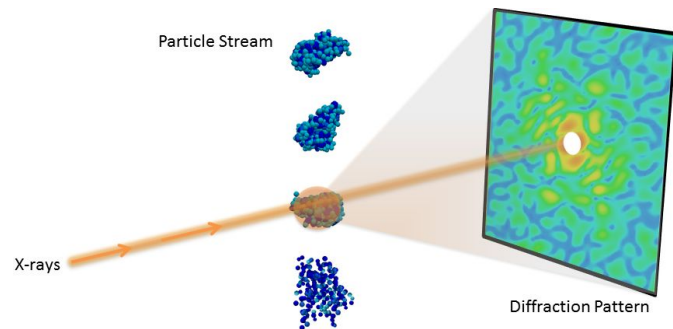
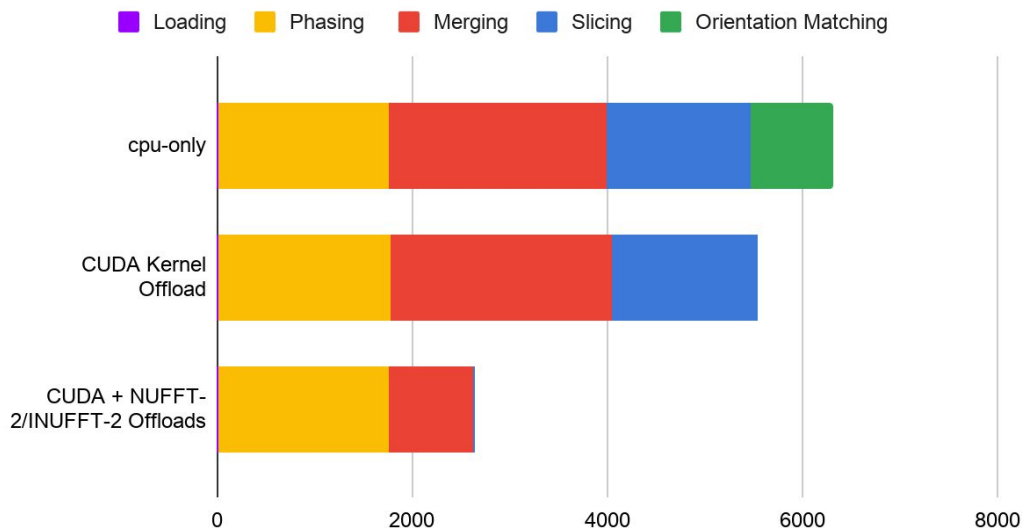
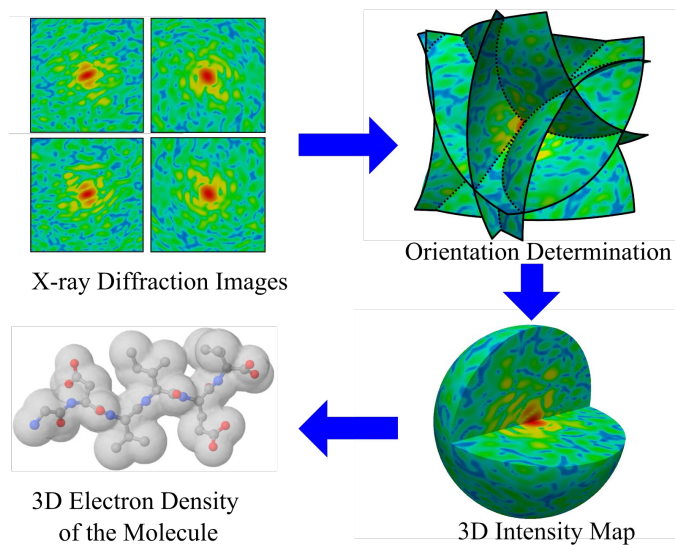


Illustration of **SPI technique**: the X-ray beam interacts with only a few molecules a time



# Large Scale Combustion Modeling w/ Pele

## Combustion Fuel ☐ Methane

- DRM19 chemistry with 21 species
- ERK chemistry solver
- 2 AMR levels

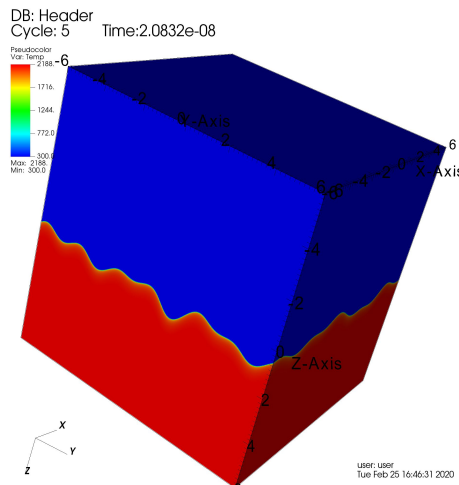
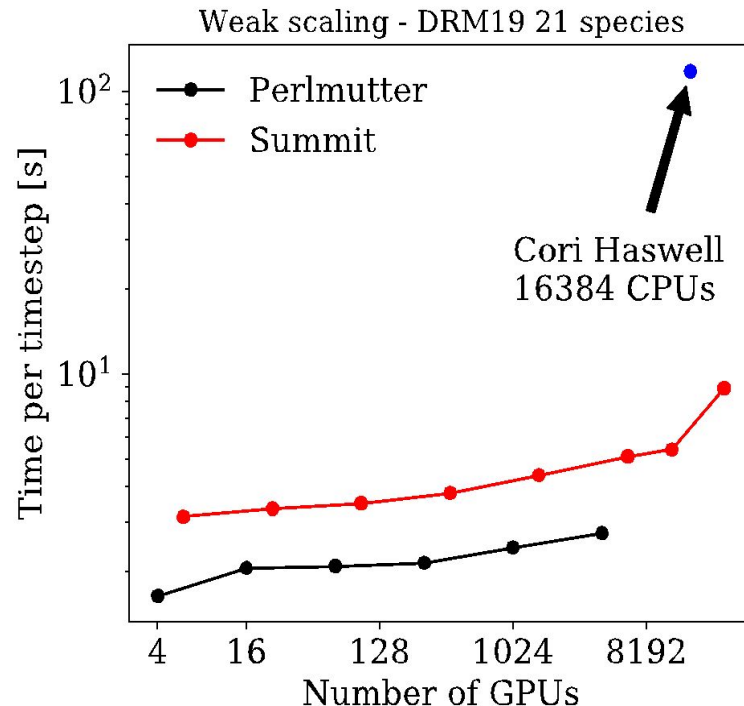


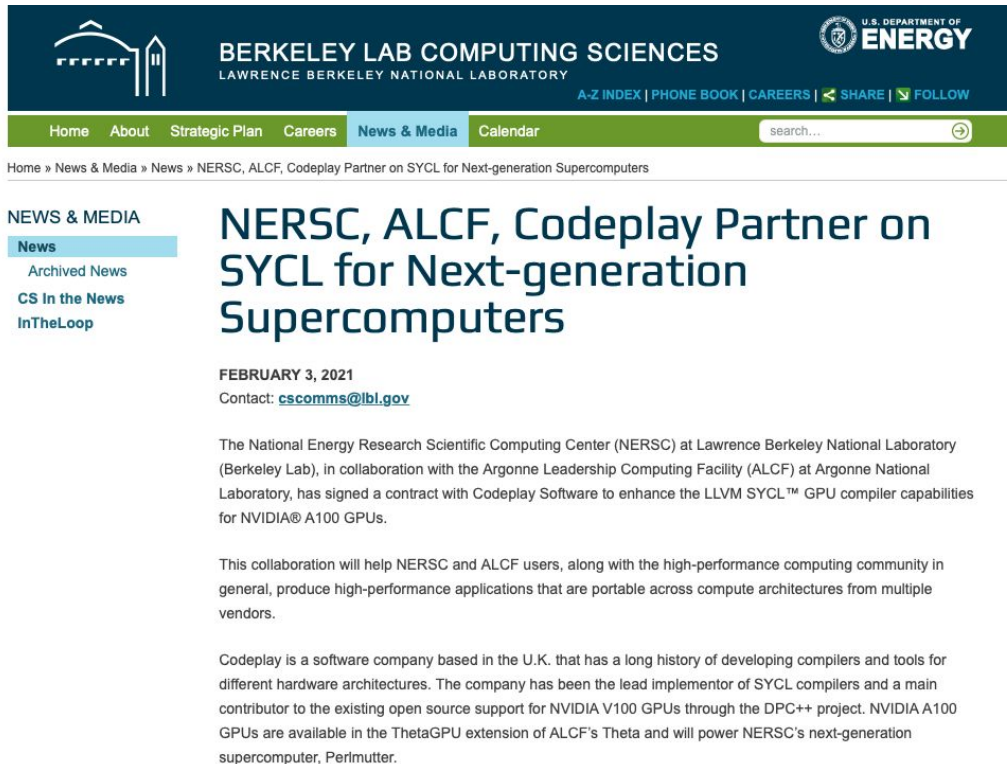
Figure above shows a statistically stationary flame. This flame configuration has been extensively used in DNS calculations and in this case it is used for scaling tests.

The configuration can be easily reproduced with different chemical mechanisms.



# NERSC, ALCF and Codeplay partnership on SYCL

- Target SYCL 2020 (latest specification) support on Ampere A100 GPUs
- Open LLVM based compiler
- Provides Portability for Apps Developed for Aurora
- Extensions for A100
  - Asynchronous Copy
  - Asynchronous Barrier
  - Tensor core types/ APIs



The screenshot shows the Berkeley Lab Computing Sciences website. The header includes the Berkeley Lab logo, the text "BERKELEY LAB COMPUTING SCIENCES" and "LAWRENCE BERKELEY NATIONAL LABORATORY", and the U.S. Department of Energy logo. A navigation bar contains links for Home, About, Strategic Plan, Careers, News & Media, and Calendar. A search bar is also present. The main content area features a "NEWS & MEDIA" section with a "News" tab selected. Below this, there are links for "Archived News", "CS In the News", and "InTheLoop". The main headline reads "NERSC, ALCF, Codeplay Partner on SYCL for Next-generation Supercomputers". The date "FEBRUARY 3, 2021" and contact information "Contact: [cscoms@lbl.gov](mailto:cscoms@lbl.gov)" are provided. The article text states: "The National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (Berkeley Lab), in collaboration with the Argonne Leadership Computing Facility (ALCF) at Argonne National Laboratory, has signed a contract with Codeplay Software to enhance the LLVM SYCL™ GPU compiler capabilities for NVIDIA® A100 GPUs." It further explains that this collaboration will help NERSC and ALCF users produce high-performance applications portable across compute architectures. A concluding paragraph mentions that Codeplay is a U.K. software company with a long history of developing compilers and tools, and that this partnership will power NERSC's next-generation supercomputer, Perlmutter.

NEWS & MEDIA

News

Archived News

CS In the News

InTheLoop

## NERSC, ALCF, Codeplay Partner on SYCL for Next-generation Supercomputers

FEBRUARY 3, 2021

Contact: [cscoms@lbl.gov](mailto:cscoms@lbl.gov)

The National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (Berkeley Lab), in collaboration with the Argonne Leadership Computing Facility (ALCF) at Argonne National Laboratory, has signed a contract with Codeplay Software to enhance the LLVM SYCL™ GPU compiler capabilities for NVIDIA® A100 GPUs.

This collaboration will help NERSC and ALCF users, along with the high-performance computing community in general, produce high-performance applications that are portable across compute architectures from multiple vendors.

Codeplay is a software company based in the U.K. that has a long history of developing compilers and tools for different hardware architectures. The company has been the lead implementor of SYCL compilers and a main contributor to the existing open source support for NVIDIA V100 GPUs through the DPC++ project. NVIDIA A100 GPUs are available in the ThetaGPU extension of ALCF's Theta and will power NERSC's next-generation supercomputer, Perlmutter.