EXASCALE COMPUTING PROJECT

ADDRESSING A NATIONAL IMPERATIVE

Software Technology Update
October 2019

exascaleproject.org

U.S. DEPARTMENT OF ENERGY | Office of Science

NNSA National Nuclear Security Administration

Table of Contents

Letter from the Software Technology Leadership Team 3
Acronyms 4
Advanced Scientific Computing Research (ASCR) Program 6
Advanced Simulation and Computing (ASC) Program 7
Introduction 13
Preparing for the Nation's Exascale Systems 17
Software Technology 27
Programming Models and Runtimes 32
Development Tools 50
Mathematical Libraries 62
Data And Visualization 76
Software Ecosystem and Delivery 90
NNSA Software Technology 96
LANL NNSA Software Technology 99
LLNL NNSA Software Technology 103
SNL NNSA Software Technology 107
Bringing It All Together 111
The ECP's Enduring Legacy 114
Credits 116

Letter from the Software Technology Leadership Team

This project overview and update report presents the Exascale Computing Project (ECP) Software Technology focus area. It is the second in a series of three documents to provide an in-depth update on activities related to the ECP's three technical focus areas. The Application Development update report was released in September 2019 and is available on the ECP website at https://www.exascaleproject.org/the-ecp-2019-application-development-report-is-available/. The third report in the series, an update on the ECP's Hardware and Integration activities, will be released in December 2019.

The ECP's software technology effort represents the key bridge between exascale systems and the scientists developing applications that will run on those platforms. The ECP offers a unique opportunity to build a coherent set of software (referred to as the Extreme-scale Scientific Software Stack–E4S) that will enable application developers to maximize their ability to write highly efficient and portable applications targeting multiple exascale architectures. Furthermore, the capabilities of E4S will aid in unlocking the latent scientific and engineering insights available from the unprecedented data produced by the applications running on exascale systems by providing a complete analysis workflow that includes new technology to collect, reduce, organize, curate, and analyze the data into actionable decisions at exascale.

Accomplishing all that the ECP is charged to do requires approaching scientific computing in a holistic manner, encompassing the entire workflow—from identification of a challenge to designing and developing algorithms, tuning an application and performing high-fidelity simulations, applying uncertainty quantification, and analyzing and visualizing the results. The ECP software stack aims to address all these needs by extending current technologies to exascale where possible, by performing the research required to conceive of new approaches necessary to address unique problems where current approaches will not suffice, and by packaging and delivering high-quality and robust software products for exascale.

As of this report, the ECP Software Technology group is managing the development and testing of capabilities that contribute to 70 unique software products spanning programming models and run times, math libraries, data, and visualization. In addition to providing the needed capabilities for ECP applications, the E4S software stack will enable many other applications to realize the performance potential of emerging high performance computing (HPC) architectures. As a collection of open-source, reusable software components, E4S is available to the broader HPC community, including other US agencies and industry.

We hope you find this overview and update report focused on the ECP's software technology efforts informative, and we welcome your inquiries.

For the Exascale Computing Project,

Doug Kothe
ECP Director

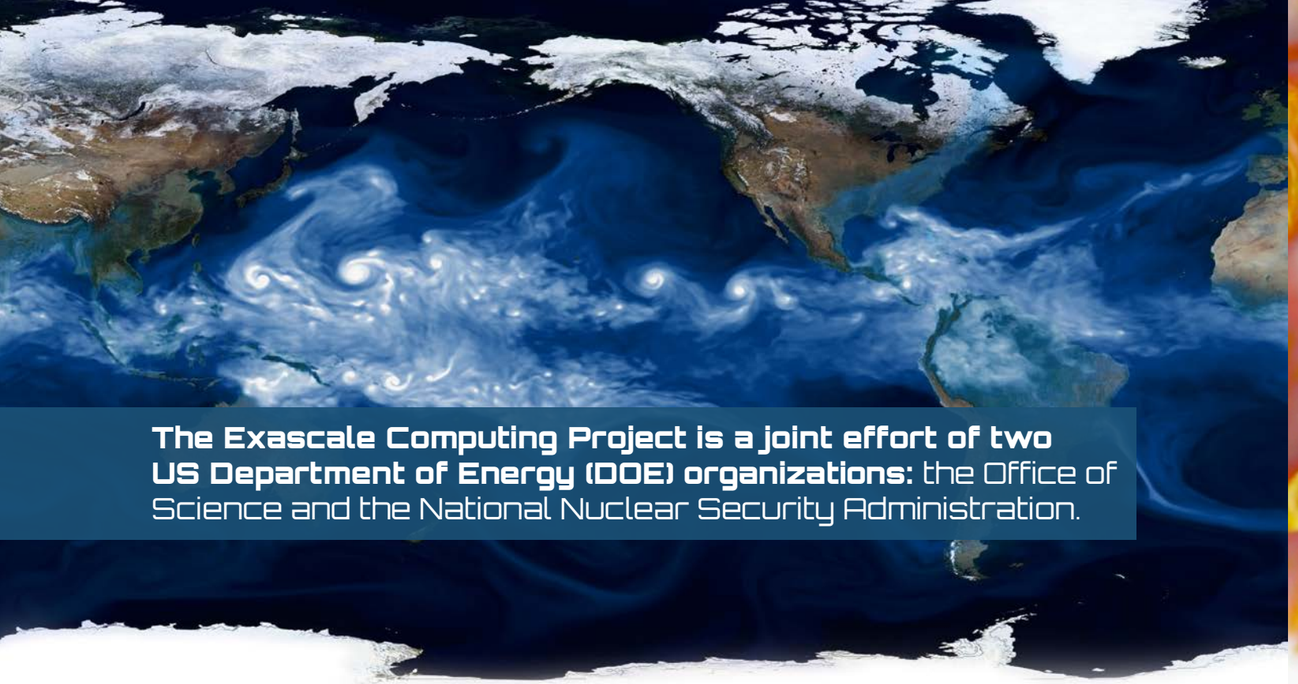
Lori Diachin
ECP Deputy Director

Mike Heroux
Software Technology Director

Jonathan Carter
Software Technology Deputy Director

ACRONYMS

ADIOS	Adaptable I/O Systems
ALCF	Argonne Leadership Computing Facility
ALE	Arbitrary Lagrangian-Eulerian
ALEXa	Accelerated Libraries for Exascale
AMR	Adaptive Mesh Refinement
ANL	Argonne National Laboratory
API	Application Programming Interface
ASC	Advanced Simulation and Computing
ATDM	Advanced Technology Development and Mitigation
ATS	Advanced Technology Systems
BEE	Build and Execution Environment
BOD	Board of Directors
CI	Continuous Integration
CPU	Central Processing Unit
DAOS	Distributed Asynchronous Object Storage
DOE	US Department of Energy
DTK	Data Transfer Kit
E4S	Extreme Scale Scientific Software Stack
ECI	Exascale Computing Initiative
ECP	Exascale Computing Project
FleCSI	Flexible Computational Science Infrastructure
GPU	Graphics Processing Unit
HDF5	Hierarchical Data Format version 5
HPC	High Performance Computing
I/O	Input/Output
LANL	Los Alamos National Laboratory
LBNL	Lawrence Berkeley National Laboratory
LLNL	Lawrence Livermore National Laboratory
LLVM	Low Level Virtual Machine
LOTF	Laboratory Operations Task Force
MPI	Message Passing Interface
NIC	Network Interface Controller
NNSA	National Nuclear Security Administration
ORNL	Oak Ridge National Laboratory
OS&ONR	OS and On-Node Runtime
OS/R	Operating System and Runtime System
PAPI	Performance Application Programming Interface
PCP	Performance Co-Pilot
PGAS	Partitioned Global Address Space
PMIx	Process Management Interface – Exascale
PMR	Programming Models and Runtimes
PROTEAS-TUNE	Programming Toolchain for Emerging Architectures and Systems
SC	Office of Science
SciDAC	Scientific Discovery through Advanced Computing
SDE	Software-Designed Event
SDK	Software Development Kit
SICM	Simplified Interface to Complex Memory
ST	Software Technology
V&V	Verification and Validation
VOL	Virtual Object Layer
XSDK	Extreme-scale Scientific Software Development Kit



The Exascale Computing Project is a joint effort of two US Department of Energy (DOE) organizations: the Office of Science and the National Nuclear Security Administration.

Community Atmospheric Model used in attributing changes in the risk of extreme weather and climate. Source: Department of Energy, Argonne Leadership Computing Facility.

Magnetic reconnection, the continuous breaking and rearrangement of magnetic field lines in a plasma, is a fundamental process in physics. Understanding reconnection phenomena has broad implications and may eventually help us protect astronauts, communications satellites, and electrical power grids. Source: Department of Energy, Los Alamos National Laboratory.

US Department of Energy
Office of Science
Advanced Scientific Computing Research (ASCR) Program

The US Department of Energy's Advanced Scientific Computing Research (ASCR) Program is one of six interdisciplinary scientific program offices within the Office of Science along with Basic Energy Sciences, Biological and Environmental Research, Fusion Energy Sciences, High Energy Physics and Nuclear Physics.

The ASCR program leads the nation and the world in supercomputing, high-end computational science, and advanced networking for science with its mission to discover, develop, and deploy computational and networking capabilities to analyze, model, simulate, and predict complex phenomena important to the US Department of Energy (DOE).

ASCR has already initiated investments to address the challenges of hybrid, multi-core computing up to the exascale (capable of an exaflop, or 10^{18} floating point operations per second). However, there are significant technological challenges that must be addressed at the exascale to reduce the energy demands and increase the memory available so the systems will be useful for science and engineering. Addressing these challenges will result in not only exascale systems but also in affordable, energy-efficient petascale systems and high-end desktops to drive scientific and engineering discovery across the country. With this integrated approach, ASCR will continue to deliver scientific insights to address national problems in energy and the environment while advancing US competitiveness in information technology and the nation's high-tech industry.

US Department of Energy
National Nuclear Security Administration
Advanced Simulation and Computing (ASC) Program

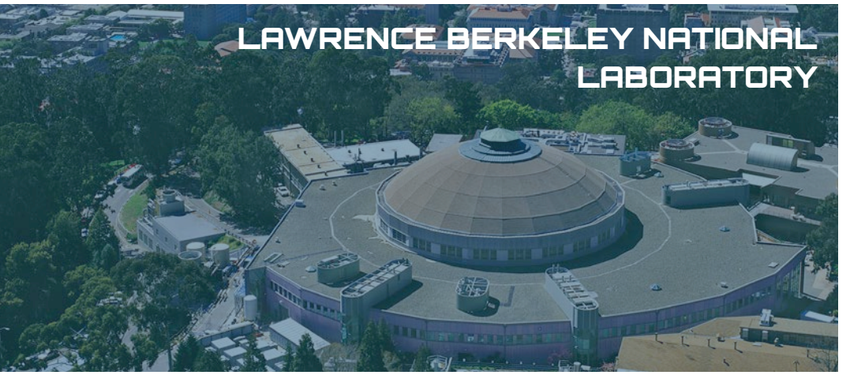
Established in 1995, the Advanced Simulation and Computing (ASC) Program provides the National Nuclear Security Administration (NNSA) Office of Defense Programs the simulation-based predictive science capabilities for the stewardship of the US nuclear weapons stockpile. Under ASC, high-performance simulation capabilities are developed to analyze and predict the performance, safety, and reliability of nuclear weapons and to certify their functionality.

Beyond the stewardship of today's stockpile, the rapidly growing capability of potential adversaries to defeat US weapons using advanced defensive systems represents a sobering threat. ASC applications must also target performance assessment of current and life-extended weapon systems subject to a wide variety of hostile environments and potential threat scenarios. To execute its mission for the NNSA Stockpile Stewardship Program (SSP), ASC oversees the high-performance simulation and computing work of three NNSA laboratories—Los Alamos National Laboratory (LANL), Lawrence Livermore National Laboratory (LLNL), and Sandia National Laboratories (SNL) as a nationally coordinated program.

The ECP Core Partner Laboratories

Six US Department of Energy (DOE) laboratories manage and oversee the Exascale Computing Project (ECP) through the project's Board of Directors, Laboratory Operations Task Force (LOTF), and the ECP senior leadership team

Argonne National Laboratory (ANL) is a multidisciplinary research center with a pioneering history in high performance computing. Home to the Argonne Leadership Computing Facility, the laboratory provides supercomputing resources to the research community to accelerate scientific discovery and innovation. Argonne is preparing to deploy Aurora, one of DOE's three planned exascale systems, in 2021.



The Computing Sciences Area at Lawrence Berkeley National Laboratory (LBNL) provides the computing and networking resources and expertise critical to advancing Department of Energy Office of Science (DOE-SC) research missions: developing new energy sources, improving energy efficiency, developing new materials, and increasing our understanding of ourselves, our world, and our universe.

Oak Ridge National Laboratory (ORNL) is home to Summit, the world's most powerful computer. Since 2005, the Oak Ridge Leadership Computing Facility has deployed Jaguar, Titan, and Summit, each the world's fastest computer in its time, and will launch its first exascale system, Frontier, in 2021.

Founded in 1952, Lawrence Livermore National Laboratory (LLNL) will welcome El Capitan, the first exascale supercomputer to support the National Nuclear Security Administration, in 2022–23. LLNL has a long pedigree of world-class supercomputing in the service of national security and basic science, from the Univac in 1953 to today's Sierra and Sequoia pre-exascale systems.

As the senior laboratory in the DOE system, Los Alamos National Laboratory (LANL) executes work in all of DOE's missions: national security, science, energy, and environmental management. LANL's contributions are part of what makes DOE a science, technology, and engineering powerhouse for the nation. LANL is strongly represented across the breadth of ECP governance, applications, and software technologies which form the exascale ecosystem.

Sandia National Laboratories (SNL) is a multidisciplinary national laboratory that develops advanced technologies to ensure global peace. SNL leads and participates in many projects across the spectrum of the ECP. Sandia's leadership spans the ECP software technology director role, combustion science and climate modeling applications, as well as software technologies ranging from performance portability to large-scale visualization.

1,000,000,000,000,000

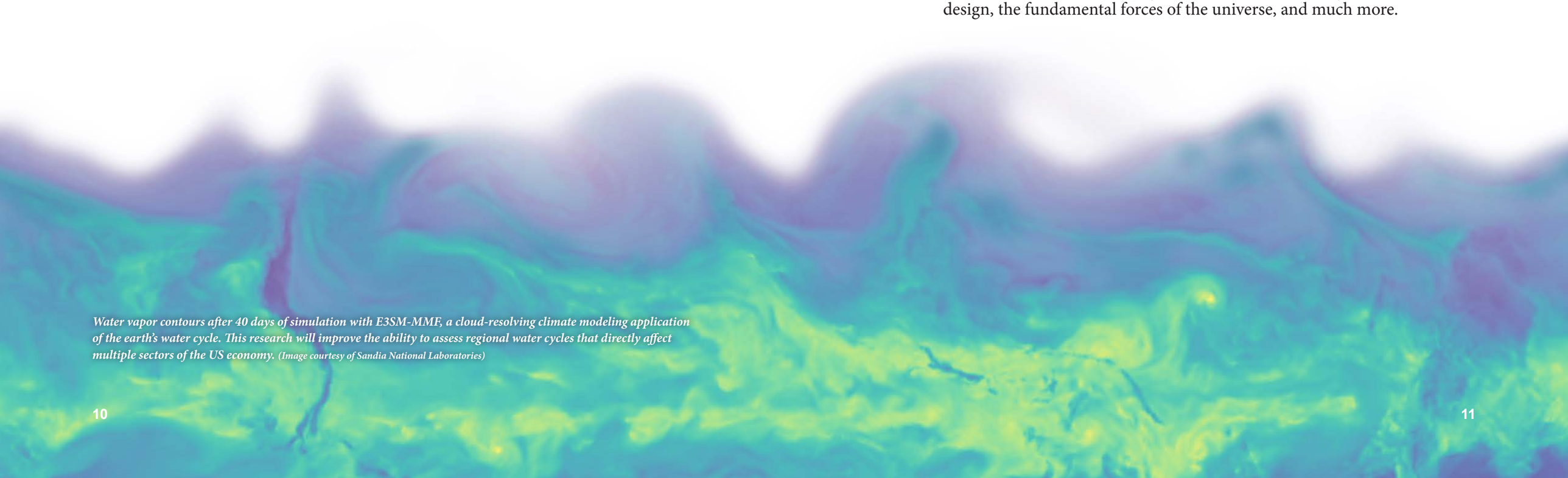
Exascale systems will perform more than a quintillion (a billion billion) operations per second.

Achieving exascale will have profound effects on the American people and the world—improving the nation's economic competitiveness, advancing scientific discovery, and strengthening our national security.

The fastest supercomputers in the world today solve problems at the petascale—that is a quadrillion (10^{15}) calculations each second.

While these petascale systems are quite powerful, the next milestone in computing achievement is the exascale—a higher level of performance in computing that will have profound impacts on everyday life.

At a quintillion (10^{18}) calculations each second, exascale supercomputers will more realistically simulate the processes involved in scientific discovery and national security such as precision medicine, regional climate, additive manufacturing, the conversion of plants to biofuels, the relationship between energy and water use, the unseen physics in materials discovery and design, the fundamental forces of the universe, and much more.



Water vapor contours after 40 days of simulation with E3SM-MMF, a cloud-resolving climate modeling application of the earth's water cycle. This research will improve the ability to assess regional water cycles that directly affect multiple sectors of the US economy. (Image courtesy of Sandia National Laboratories)

Oak Ridge National Laboratory

Oak Ridge National Laboratory is the host site for the project office of the Exascale Computing Project.

ORNL's supercomputing program grew from humble beginnings to deliver the world's most powerful system several times over the past three decades. On the way, it has helped researchers deliver practical breakthroughs and new scientific knowledge in climate, materials, nuclear science, and a wide range of other disciplines.

INTRODUCTION

The quest to develop a capable exascale ecosystem is a monumental effort

The US Department of Energy (DOE) is a long-time global leader in the development and use of high performance computing (HPC). HPC-based modeling and simulation is vital to the execution of DOE missions in science and engineering and to DOE's responsibility for stewardship of the nation's nuclear weapons stockpile.

Over the past several decades, sustained technology investment has supported the development of increasingly powerful HPC systems and produced substantial benefits for the United States. However, as other nations have recognized the benefits of HPC and increased their investments, US leadership in this important area is no longer ensured—yet this leadership is essential to our economic, energy, and national security.

To maintain leadership and to address future challenges in economic impact areas such as national security, energy assurance, economic competitiveness, healthcare, and scientific discovery, as well as growing security threats, the United States is making a strategic move in HPC—a grand convergence of advances in co-design, modeling and simulation, data analytics, machine learning, and artificial intelligence.

Collaboration, Partnership, and a Strategic Focus

The DOE-led Exascale Computing Initiative (ECI), a partnership between two DOE organizations, the Office of Science (SC) and the National Nuclear Security Administration (NNSA), was formed in 2016 to accelerate research, development, acquisition, and deployment projects to deliver exacale computing capability to the DOE labs by the early to mid 2020s. The ECI includes three main components: (1) SC and NNSA computer facility site preparation investments, (2) computer vendor nonrecurring engineering activities needed for the delivery of exascale systems within this time fame, and (3) the Exascale Computing Project (ECP), which was launched in 2016 and brings together research, development, and deployment activities as part of a capable exascale computing ecosystem to ensure an enduring exascale computing capability for the nation.

The ECP is focused on delivering specific applications, software products, and outcomes on DOE computing facilities. Integration across these elements with specific hardware technologies for exascale system instantiations is fundamental to the success of the ECP. The outcome of the ECP is the accelerated delivery of a capable exascale computing ecosystem to provide breakthrough solutions addressing our most critical challenges in scientific discovery, energy assurance, economic competitiveness, and national security. This outcome is not simply a matter of ensuring more powerful computing systems. The ECP is designed to create more valuable and rapid insights from a wide variety of applications ("capable"), which requires a much higher level of inherent efficacy in all methods, software tools, and ECP-enabled computing technologies to be acquired by the DOE laboratories ("ecosystem").

ECP is built on a strategic collaboration of government, academia, and industry

Advanced leadership computing capabilities are required to

- discover new energy solutions needed for a sustainable future;
- extend our knowledge of the natural world through scientific inquiry;
- maintain a vibrant effort in science and engineering as a cornerstone of the nation's economic prosperity;
- deliver new technologies to advance DOE's mission; and
- sustain a world-leading workforce in advanced technology.

The ECP is led by a team of senior scientists, project management experts, and engineers from six of the largest DOE national laboratories. Working together, this leadership team has established an extensive network to deliver a capable exascale computing ecosystem for the nation.

The ECP enables US revolutions in technology development: scientific discovery; healthcare; energy, economic, and national security

ECP Mission

Develop exascale-ready applications and solutions that address currently intractable problems of strategic importance and national interest.

Create and deploy an expanded and vertically integrated software stack on DOE HPC pre-exascale and exascale systems.

Deliver US HPC vendor technology advances and deploy ECP products to DOE HPC pre-exascale and exascale systems.

ECP Vision

Deliver exascale simulation and data science innovations and solutions to national problems that enhance US economic competitiveness, improve our quality of life, and strengthen our national security.

Argonne National Laboratory, the nation’s first national laboratory, will be home to Aurora, one of the US Department of Energy’s three planned exascale supercomputers.

PREPARING FOR THE NATION’S EXASCALE SYSTEMS

Preparing for the Nation's Exascale Systems

Today, US scientists can take advantage of powerful pre-exascale systems at two of DOE's leading HPC facilities: the 200 petaflop IBM Summit system at Oak Ridge National Laboratory and the 125 petaflop IBM Sierra system at Lawrence Livermore National Laboratory. These systems serve numerous scientific and national security programs today and are critical stepping stones as we prepare for the nation's first exascale supercomputers to be procured between 2021 and 2023.

As of June 2019, the DOE had the top two systems on the world ranking of the TOP500 computing systems, a list that is updated twice yearly. Access to these systems is proving invaluable in helping scientists prepare their codes for the forthcoming exascale platforms. The planned US exascale systems are critically important for addressing the growing computational challenges and sustaining the nation's preeminence in technological advances and economic competitiveness. In addition to solving important computational grand challenge science problems and addressing serious economic, environmental, and national security challenges, these exascale systems and the technology that will ultimately be made available to the broad HPC community will ensure the United States is at the forefront of HPC on a global scale.

For the past few decades, DOE's continued commitment to advances in supercomputing has fueled a robust partnership with the computer industry in software and hardware, which has led to the development of new technologies and markets surrounding them in a variety of areas ranging from HPC architectures, memory technologies, high-speed interconnects, systems software, programming environments, and high-performance storage systems. This commitment to the delivery of new supercomputing technologies has served to advance our energy and science missions as well as the technology industry and sustain a world-leading workforce. Advances in hardware and software not only enable new and more advanced science but also lead to important advances in all areas touched by computing and computing technology, thereby driving innovations in many other industries such as automotive, aerospace, chemical production, enhanced oil recovery, precision medicine, and nuclear energy.

Supported by the efforts of the ECP, the United States is preparing for the arrival of three exascale systems starting in 2021—***Aurora*** at Argonne National Laboratory, ***Frontier*** at Oak Ridge National Laboratory, and ***El Capitan*** at Lawrence Livermore National Laboratory.

The success of these first US exascale systems will depend largely on the ECP to fulfill its mission of building exascale-ready software and applications and influencing the development of vendor hardware specifically needed to deploy capable exascale systems. In support of this mission, ECP is funding several efforts from national laboratories, industry, and academia to fill the gaps in hardware, software services, and applications.

Pre-Exascale Systems

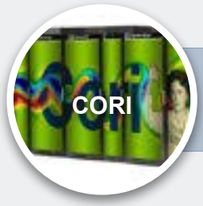
2012

ORNL
Cray/AMD/
NVIDIA

ANL
IBM BG/Q

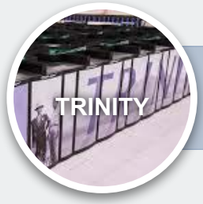
LLNL
IBM BG/Q

2016



LBNL
Cray/Intel

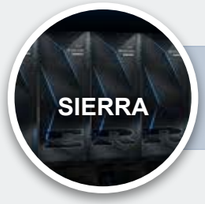
ANL
Intel/Cray



LANL/SNL
Cray/Intel

2018

ORNL
IBM/NVIDIA



LLNL
IBM/NVIDIA

Future Exascale Systems

2020

LBNL
Cray/AMD/
NVIDIA

2021–2023

ORNL
Cray/AMD

ANL
Intel/Cray

LANL/SNL
TBD

LLNL
Cray

AURORA

Argonne National Laboratory

Argonne National Laboratory's next-generation supercomputer, Aurora, will be one of the nation's first exascale systems when it is delivered in 2021. Designed in collaboration with Intel and Cray, Aurora will help ensure continued US leadership in high-end computing for scientific research.

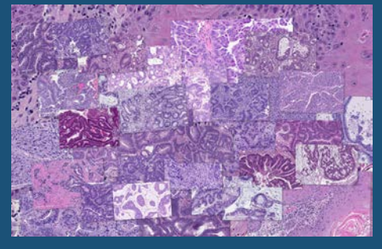
Scientists will use Aurora to pursue some of the farthest-reaching science and engineering breakthroughs ever achieved with supercomputing. From mapping the human brain and designing new functional materials to advancing the development of alternative energy sources, Argonne's forthcoming machine will enable researchers to accelerate discoveries and innovation across scientific disciplines.

Aurora will be based on Intel's Xeon Scalable processors, high-performance Intel Xe GPU compute accelerators, and Optane DC persistent memory. The system will rely on Cray's Shasta exascale-class architecture and Slingshot interconnect technology, which can provide concurrent support for advanced simulation and modeling, artificial intelligence (AI), and analytics workflows. Aurora will leverage historical advances in software investments along with increased application portability via Intel's OneAPI. The supercomputer will also introduce a new I/O system called Distributed Asynchronous Object Storage (DAOS) to meet the needs of exascale workloads.

SYSTEM SPECS	AURORA
Sustained Performance	≥ 1 EF
Cabinets	> 100
Node	Intel Xeon scalable processor, multiple Xe arch based GP-GPUs
Aggregate System Memory	> 10 PB
System Interconnect	Cray Slingshot providing 100 GB/s network bandwidth. Slingshot dragonfly network providing adaptive routing, congestion management, and quality of service.
High-Performance Storage	> 230 PB, > 20 TB/s (DAOS)
Programming Models	Intel OneAPI, OpenMP, DPC++/SYCL

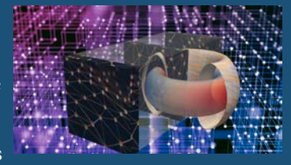
Aurora will usher in a new era of scientific discovery and innovation

"What excites me most about exascale systems like Aurora is the fact that we now have, in one platform and one environment, the ability to mix simulation and artificial intelligence. This idea of mixing simulation and data-intensive science will give us an unprecedented capability and open doors in research which were inaccessible before, like cancer research, materials science, climate science, and cosmology."



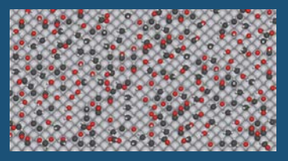
– Rick Stevens, Argonne Associate Laboratory Director for Computing, Environment, and Life Sciences; Aurora Early Science Program Principal Investigator

"Exascale computing's ability to handle much larger volumes of data unlocks our ability to prove what was once unprovable. Plus, the incredible speed of supercomputers shortens our time-to-discovery by a huge margin. Work that used to take months or years now takes hours or days. Therefore, we finally have the means to validate theories statistically and prove their reality."

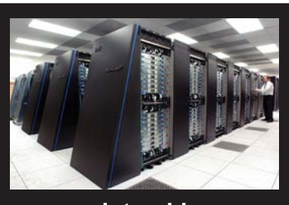


– William Tang, Principal Research Physicist at Princeton Plasma Physics Laboratory; Aurora Early Science Program Principal Investigator

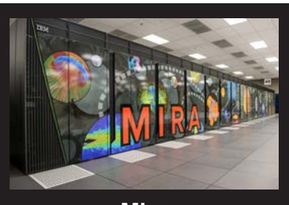
"Exascale will allow us to solve yesterday's insurmountable scientific problems. Until now, we have been forced to navigate trade-offs between the fidelity of a simulation, the time necessary to derive it, and the number of simulations we can perform with a finite number of cycles. With a greatly increased computational resource like Aurora, we can perform vastly more high-fidelity simulations. Therefore, we can get much better quantitative descriptions to fuel our work."



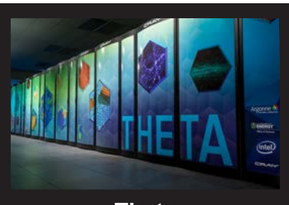
– David Bross, Argonne Computational Chemist; Aurora Early Science Program Principal Investigator



Intrepid
2007–2013



Mira
2012–2019



Theta
2016–TBD

Aurora
Coming in 2021

Leadership Computing

Aurora will be housed at the Argonne Leadership Computing Facility (ALCF), a DOE Office of Science User Facility that deploys and operates world-class supercomputers for open science research. The ALCF was established at Argonne National Laboratory in 2006 as part of a DOE initiative dedicated to providing leading-edge computing resources to the science and engineering community to advance fundamental discovery and understanding in a broad range of disciplines. From Intrepid to Mira to Theta, each new ALCF system brings advanced capabilities that enable researchers to expand their investigations in both scope and scale.

FRONTIER

Oak Ridge National Laboratory

Scheduled for delivery in 2021, Frontier is ORNL's exascale supercomputer. Frontier will accelerate innovation in science and technology and maintain US leadership in high-performance computing and artificial intelligence. Frontier will be able to simulate the detailed life cycle of a nuclear reactor, help to uncover the genetics of complex diseases, and allow scientists to build on recent developments in science and technology by further integrating artificial intelligence with more detailed data analytics coupled to new approaches to modeling and simulation.

The system will be based on Cray's new Shasta architecture and Slingshot interconnect with high-performance AMD EPYC CPU and Radeon Instinct GPU technology. The new accelerator-centric compute blades will support a 4:1 GPU-to-CPU ratio with high-speed links and coherent memory between them within the node. With Frontier, scientists will be able to pack in more calculations, identify new patterns in data, and develop innovative data analysis methods to accelerate the pace of scientific discovery.

SYSTEM SPECS	SUMMIT	FRONTIER
Peak Performance	200 PF	> 1.5 EF
Cabinets	256	> 100
Node	2 IBM POWER9 CPUs 6 NVIDIA Volta GPUs	1 HPC and AI Optimized AMD EPYC CPU 4 Purpose-built AMD Radeon Instinct GPU
CPU-GPU Interconnect	NVLINK Coherent memory across the node	AMD Infinity Fabric Coherent memory across the node
System Interconnect	2x Mellanox EDR 100G InfiniBand Non-Blocking Fat-Tree	Multiple Slingshot Network Interface Controller (NICs) providing 100 GB/s network bandwidth. Slingshot dragonfly network provides adaptive routing, congestion management, and quality of service.
Storage	250 PB, 2.5 TB/s, GPFS	2–4x performance and capacity of Summit's I/O subsystem. Frontier will have near node storage like Summit.

Frontier will help guide researchers to new discoveries at exascale.

"As a third-generation accelerated system—following the world-leading Summit system deployed at ORNL in 2018—Frontier will provide unmatched capability for modeling and simulation studies along with new capabilities for deep learning, machine learning and data analytics for applications ranging from manufacturing to human health."

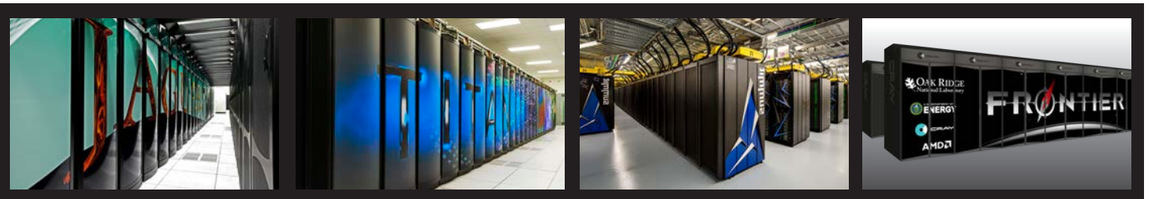
–James J. Hack, Director, National Center for Computational Sciences at ORNL

"The ability to have a large amount of very fast memory like we're going to have on Frontier will be a real boon to our simulations."

– Bronson Messer,
ORNL and the ECP ExaStar Team

"The thing that's really attractive about Frontier is the powerful nodes. Having fewer powerful nodes with a very tightly integrated set of CPUs and GPUs at the node level gives us the ability to distribute hundreds or thousands of microstructure and property calculations on one or a few nodes across the machine."

– John Turner, ORNL and the ECP ExaAM Team



World-Leading Systems

Oak Ridge National Laboratory has decades of experience in delivering, operating, and conducting research on world-leading supercomputers. Since 2005, ORNL has deployed Jaguar, Titan, and Summit, each the world's fastest computer in its time. Frontier will leverage ORNL's extensive experience and expertise in GPU-accelerated computing to become the US DOE's next record-breaking supercomputer when it debuts in 2021.

EL CAPITAN

Lawrence Livermore National Laboratory

El Capitan will be the National Nuclear Security Administration's (NNSA's) first exascale supercomputer. Scheduled for delivery to LLNL in late 2022, El Capitan will feature advanced capabilities for modeling, simulation and artificial intelligence.

Boasting a peak performance of more than 1.5 exaflops, El Capitan will perform essential functions for the NNSA's Stockpile Stewardship Program, which supports US national security missions through leading-edge scientific, engineering, and technical tools and expertise, ensuring the safety, security, and effectiveness of the nation's nuclear stockpile in the absence of underground testing. El Capitan will be used to make critical assessments necessary for addressing evolving threats to national security and other purposes such as nonproliferation and nuclear counterterrorism.

El Capitan will be built on Cray's Shasta supercomputing architecture and will be composed of Shasta compute nodes and a future generation of ClusterStor storage. This unique architecture will be connected with Cray's new Slingshot high-speed interconnect. The Shasta architecture can accommodate a variety of processors and accelerators, making it possible for Cray and LLNL to make a late-binding decision on CPU and GPU components in the coming months.

	SIERRA	EL CAPITAN	EL CAPITAN and SIERRA comparison
Peak (Exaflop/s)	125 PF	> 1.5	10.5× more performance
System Power (MegaWatts)	11.0	< 40	>3.3× more power efficient
Application Performance Improvement		6× to 12× over Sierra	

El Capitan will provide unprecedented capabilities in support of the nation's nuclear deterrent.

"El Capitan will allow our scientists and engineers to get answers to critical questions about the nuclear stockpile faster and more accurately than ever before, improving our efficiency and productivity and enhancing our ability to reach our mission and national security goals."

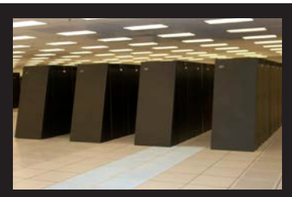
– Bill Goldstein, Director, Lawrence Livermore National Laboratory

"A machine of this magnitude will be key for the rapid, 3D iterative analyses required to modernize our deterrent, as adversaries are making rapid improvements in their defensive and offensive capabilities."

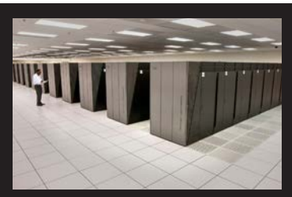
–Michel McCoy, LLNL Advanced Simulation & Computing Program Director

"El Capitan will continue the GPU-accelerated era begun at LLNL with Sierra. This system architecture offers outstanding price/performance that will ensure that ASC contributes critical computing cycles to NNSA's mission in FY24 and beyond. We are excited to resume the LLNL partnership with Cray after its long dormancy."

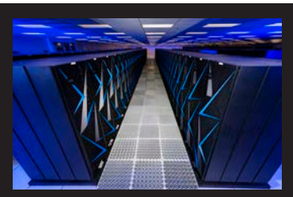
– Bronis R. de Supinski, Chief Technology Officer for Livermore Computing, Lawrence Livermore National Laboratory



BlueGene/L
No. 1 in 2004-2008



Sequoia
No. 1 in 2012



Sierra
No. 2 in 2018,2019

El Capitan
expected 2022

A History of Leading-Edge Supercomputing at LLNL

Since it was founded in 1952, Lawrence Livermore National Laboratory has prided itself on being the tip of the spear for cutting-edge computing. The first computer, the UNIVAC I, was ordered before the Lab even opened, marking the beginning of a decades-long mission to develop the world's fastest and most powerful computers and to use those machines to solve large, complex problems. Over the years, LLNL supercomputers have topped more Top500 lists of the world's fastest and most-powerful systems than any other computing facility on Earth, the most recent being the IBM/Blue Gene Sequoia in 2012. LLNL's current most powerful supercomputer Sierra is second only to Oak Ridge's Summit.

Lawrence Berkeley National Laboratory

Thirteen Nobel prizes are associated with Lawrence Berkeley National Laboratory, located on a 202-acre site in the hills above the UC Berkeley campus.

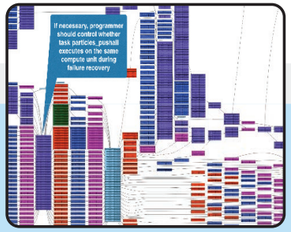
SOFTWARE TECHNOLOGY

SOFTWARE TECHNOLOGY

Each generation and variety of supercomputers offer a range of performance improvements through design innovation and implementation of new technologies. The forthcoming US exascale systems are no exceptions. Adapting scientific applications to realize optimal performance on these diverse, complex systems is extremely challenging. New algorithms, designs, and implementations are essential to taking full advantage of future exascale platforms.

The ECP's Software Technology (ST), group develops new capabilities that enable thousands of scientific application developers to focus on writing their applications on top of a software stack that delivers impressive performance across a broad set of platforms. Application Development teams can rely on the ECP-developed programming environments, libraries, and tools for state-of-the-art algorithms, design, and implementations, greatly reducing how much effort an application team must expend to achieve performance from one machine to the next.
Without the ECP software products, development of ready-for-exascale scientific applications would be extremely costly, likely achieving suboptimal performance and lacking portability. Furthermore, innovative algorithms that enable scalable performance would be difficult to disseminate across all the applications that could possibly benefit. In collaboration with appropriate organizations and software and hardware suppliers throughout the US computer industry, the leading-edge software that the ECP provides makes efficient and effective scientific application development possible.

ECP software technologies are a fundamental underpinning in delivering on DOE's exascale mission



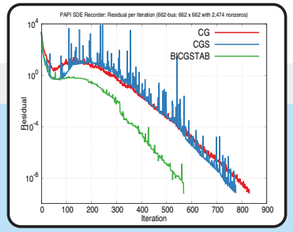
Programming Models & Runtimes

Enhance and prepare OpenMP and MPI programming models (hybrid programming models, deep memory copies) for exascale

Development of performance portability tools (e.g., Kokkos and RAJA)

Support alternate models for potential benefits and risk mitigation: PGAS (UPC++/GASNet), task-based models (Legion, PaRSEC)

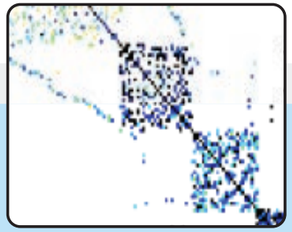
Libraries for deep memory hierarchy and power management



Development Tools

Continued, multifaceted capabilities in portable, open-source LLVM compiler ecosystem to support expected ECP architectures, including support for Flang

Performance analysis tools that accommodate new architectures, programming models, e.g., PAPI, Tau



Math Libraries

Linear algebra, iterative linear solvers, direct linear solvers, integrators and nonlinear solvers, optimization, FFTs, etc.

Performance on new node architectures; extreme strong scalability

Advanced algorithms for multi-physics, multiscale simulation and outer-loop analysis

Increasing quality, interoperability, complementarity of math libraries

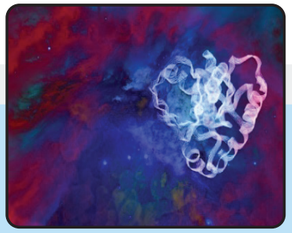
Products within the ECP software portfolio are key components of a productive and sustainable exascale computing ecosystem that will position the US Department of Energy (DOE) and the broader high performance computing (HPC) community with a strong foundation for future extreme-scale computing capabilities. The exascale software portfolio will also have an important near-term impact on the broad HPC community as the software stack components are released for general use on many of today's HPC platforms.

The ECP software products are organized by the following six categories:

- Programming Models and Runtimes
- Development Tools
- Mathematical Libraries
- Data and Visualization
- Software Ecosystem and Delivery
- NNSA Software Technology

Programming Models and Runtimes: The ECP software team is developing key enhancements to message passing interface (MPI) and OpenMP, addressing in particular the important design and implementation challenges of combining massive inter-node and intra-node concurrency into an application. They are also developing a diverse collection of products that further address next-generation node architectures to improve realized performance, ease of expression and performance portability.

Development Tools: The team is enhancing existing widely used performance tools and developing new tools for next-generation platforms. As node architectures become more complicated and concurrency even more necessary, impediments to performance and scalability become even harder to diagnose and fix. Development tools provide essential insight into these performance challenges and code transformation and support capabilities that help software teams generate efficient code, use new memory systems, and more.

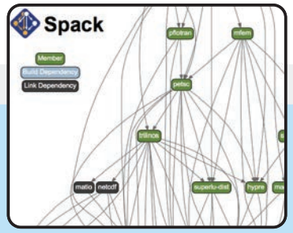


Data and Visualization

I/O via the HDF5 API

Insightful, memory-efficient in situ visualization and analysis — Data reduction via scientific data compression

Checkpoint restart



Software Ecosystem and Delivery

Develop features in Spack necessary to support all ST products in E4S, and the AD projects that adopt it

Development of Spack stacks for reproducible turnkey deployment of large collections of software

Optimization and interoperability of containers on HPC systems

Regular E4S releases of the ST software stack and SDKs with regular integration of new ST products

NNSA Software Technology

Projects that have both a mission role and open science role

Major technical areas: New programming abstractions, math libraries, data and viz libraries

Cover most ST technology areas

Open-Source NNSA software projects

Subject to the same planning, reporting, and review processes

Mathematical Libraries: High-performance scalable math libraries have enabled parallel execution of many applications for decades. Collaborative teams are providing the next generation of these libraries to address needs for latency hiding, improved vectorization, threading, and strong scaling. In addition, they are addressing new demands for system-wide scalability including improved support for coupled systems and ensemble calculations. The math libraries teams are also spearheading the Software Development Kit (SDK) initiative, which is a pillar of the ECP software delivery strategy.

Data and Visualization: The ECP's software portfolio has a large collection of data management and visualization products that provides essential capabilities for compressing, analyzing, moving, and managing data. These tools are becoming even more important as the volume of simulation data that is produced grows faster than the ability to capture and interpret it.

Software Ecosystem and Delivery: This new technical area of the ECP software group provides important enabling technologies such as software build, test and integration tools, in particular Spack, and containers environments that leverage emerging industry standards for portable execution adapted to leadership computing platforms. This area also provides the critical resources and staffing that will enable ECP ST to perform continuous integration testing and product releases. Finally, this area engages with software and system vendors and DOE facilities staff to ensure coordinated planning and support of the the ECP software products.

The ECP Software Stack (https://E4S.io)

The Extreme-scale Scientific Software Stack (E4S) is the capstone effort for ECP software activities. We have launched a community effort to provide open-source software packages for developing, deploying, and running scientific applications on HPC platforms. E4S provides from-source builds and containers of a broad collection of HPC software packages.

E4S exists to accelerate the development, deployment, and use of HPC software, lowering the barriers for HPC users. E4S provides containers and turn-key, from-source builds of more than 80 popular HPC products in programming models, such as MPI; development tools such as HPCToolkit, TAU and PAPI; math libraries such as PETSc and Trilinos; and Data and Viz tools such as HDF5 and Paraview.

E4S is not an ECP-specific software suite: The products in E4S represent a holistic collection of capabilities sponsored by the ECP, and all additional underlying software required to use the full software capabilities.

Software Releases

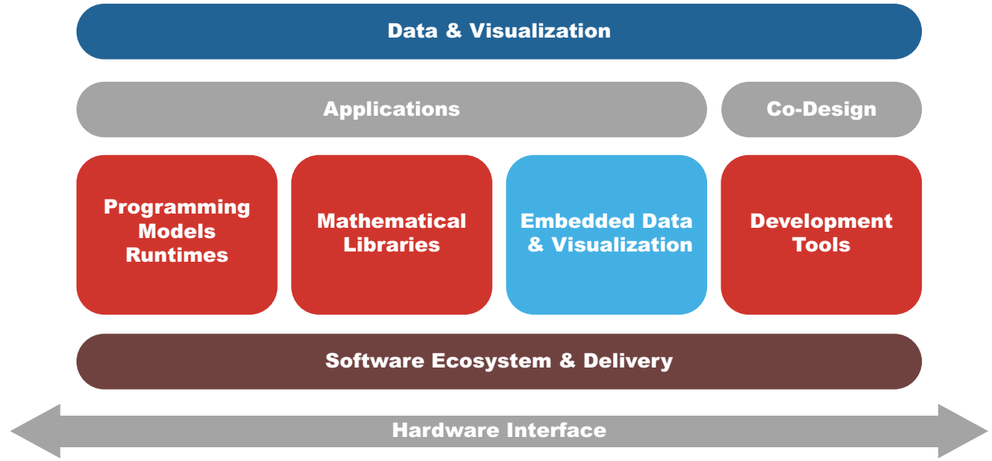
SDKs present an opportunity for a large software ecosystem project, such as the effort within the ECP, to foster increased collaboration, integration, and interoperability among its funded efforts. Part of the ECP software strategy is the creation of SDK. An ECP SDK is a collection of related ECP software products (called packages) in which coordination across package teams will improve usability and practices and foster community growth, among other efforts developing similar and complementary capabilities.

An SDK is more of a project than a product, although it involves several products. It can also be considered an association of products and product teams. The activities that take place inside an SDK promote interoperability (where appropriate and logical) between products. The initial version 0.2 release of E4S contains member packages of one SDK—the Extreme-Scale Scientific Software Development Kit (xSDK). Future releases will incorporate five additional SDKs that are under development.

SDKs provide an important organizational structure for coordinating E4S activities, introducing an intermediate aggregation layer that reduces organizational complexity.

The E4S suite is a large and growing ECP-led effort to build and test a comprehensive scientific software ecosystem. E4S V0.1 contained 25 ECP products. E4S V0.2 contained 37 ECP products and numerous additional products needed for a complete software environment. Eventually E4S will contain all open-source products to which the ECP contributes and all related products needed for a holistic exascale environment. We expect the E4S effort to live beyond the timespan of the ECP, becoming a critical element of the scientific software ecosystem.

ECP SW Stack: Strategic Alignment and Synergies



Software Portfolio

Programming Models and Runtimes

- Exascale MPI / MPICH
- Legion
- PaRSEC
- Pagoda: UPC++/GASNet
- SICM
- OMPI-X
- Kokkos/RAJA
- Argo

Exascale MPI / MPICH

Objective: Enhance the MPI standard and the MPICH implementation of MPI for the exascale task-based programming model

Principal Investigator: Pavan Balaji, Argonne National Laboratory

Legion

Objective: Develop/ enhance this task-based programming model

Principal Investigator: Pat McCormick, Los Alamos National Laboratory

PaRSEC

Objective: Develop/ enhance this task-based programming model

Principal Investigator: George Bosilca, University of Tennessee – Knoxville

Pagoda: UPC++/ GASNet

Objective: Develop/enhance a Partitioned Global Address Space programming model

Principal Investigator: Erich Strohmaier, Lawrence Berkeley National Laboratory

SICM

Objective: Develop an interface and library for accessing a complex memory hierarchy

Principal Investigator: Mike Lang, Los Alamos National Laboratory

OMPI-X

Objective: Enhance the MPI standard and the Open MPI implementation of MPI for exascale

Principal Investigator: David Bernholdt, Oak Ridge National Laboratory

Kokkos/RAJA

Objective: Develop abstractions for node-level performance portability

Principal Investigator: Christian Trott, Sandia National Laboratories

Argo

Objective: Optimize existing low-level system software components to improve performance and scalability and improve functionality of exascale applications and runtime systems

Principal Investigator: Pete Beckman, Argonne National Laboratory

Exascale MPI / MPICH

Efficient communication among the compute elements within high performance computing systems is essential for simulation performance. The Message Passing Interface (MPI) is a community standard developed by the MPI Forum for programming these systems and handling the communication needed. MPI is the de facto programming model for large-scale scientific computing and is available on all the large systems; most of DOE's parallel scientific applications running on pre-exascale systems use MPI. The goal of the Exascale-MPI project is to both evolve the MPI standard to fully support the complexity of the exascale systems and deliver MPICH, a reliable, performant implementation of the MPI standard, for these systems.

While MPI will continue to be a viable programming model on exascale systems, both the MPI standard and the MPI implementations need to address the challenges posed by the increased scale, performance characteristics, evolving architectural features, and complexity expected from the exascale systems as well as provide support for the capabilities and requirements of the applications that will run on these systems.

Therefore, this project addresses five key challenges to deliver a performant MPICH implementation: (1) scalability and performance on complex architectures that include, for example, high core counts, processor heterogeneity, and heterogeneous memory; (2) interoperability with intranode programming models having a high thread count such as OpenMP, OpenACC, and emerging asynchronous task models; (3) software overheads that are exacerbated by lightweight cores and low-latency networks; (4) extensions to the MPI standard based on experience with applications and high-level libraries and frameworks targeted at exascale; and (5) topics that become more significant for exascale architectures—memory and power usage, and resilience.

The MPICH development effort continues to address several key challenges such as performance and scalability, heterogeneity, hybrid programming, topology awareness, and fault tolerance. Several additional features are being developed in order to support the exascale machines that will be deployed, including (1) support for multiple accelerator modes and native hardware models that will facilitate data transfers between GPU accelerators and the communication network in cases where native hardware support is lacking and (2) offline and online performance tuning based on static and dynamic system configurations, respectively.

This team will also produce a significantly larger test suite to stress test various use cases of MPI and develop a test generation toolkit that automatically profiles MPI usage by applications (using the MPI profiling interface) and generates a simple test program that represents the MPI communication pattern of the application, covering basic MPI features, sanitized iterative loops, memory buffer management, and incomplete executions. These activities will help improve both the reliability and performance of the MPICH implementation and other MPI implementations as they evolve.

The team will continue to engage with the MPI Forum to ensure that future MPI standards meet the needs of both the ECP and broader DOE applications. To achieve good performance on exascale machines, the team plans to develop new MPI features for application-specific requirements, such as alternative fault tolerance models and reduction neighborhood collectives, either through the inclusion in the standard or as extensions to the standard.

Progress to date

- The Exascale-MPI team developed a high-performance, production-quality MPI implementation called MPICH. The team continues to improve the performance and capabilities of the MPICH software in order to meet the demands of ECP and other broader DOE applications.

- Some technical risks that have been retired include scalability and performance over complex architectures and interoperability with intranode programming models having high thread count such as OpenMP.

PI: Pavan Balaji, Argonne National Laboratory

Collaborators: Argonne National Laboratory

LEGION

The complexity of the exascale systems that will be delivered, from processors with many cores to accelerators and heterogeneous memory, makes it challenging for scientists to achieve high performance from their simulations. Legion provides a data-centric programming system that allows scientists to describe the properties of their program data and dependencies, along with a runtime that extracts tasks and executes them using knowledge of the exascale systems to improve performance, thus shielding scientists from this complexity.

Increasing hardware specialization, power, and cost constraints will result in exascale systems with billion-way concurrency, a growing gap between memory and network latency and floating-point performance, heterogeneity in both processing and memory capabilities, and more dynamic performance characteristics due to power capping and highly tapered network topologies. Achieving sustained performance on these systems will require significant advances in latency hiding, minimizing data movement, and the ability to extract additional levels of parallelism from applications.

The Legion parallel programming system is a data-centric system for writing portable high-performance programs targeted at distributed, heterogeneous architectures designed to address these challenges. Legion presents abstractions which allow programmers to describe the properties of their program data, such as independence and locality. By making the Legion programming system aware of the structure of program data, it can automate many of the tedious tasks programmers currently face, including correctly extracting task- and data-level parallelism and moving data around complex memory hierarchies. A novel mapping interface provides explicit programmer-controlled placement of data in the memory hierarchy and assignment of tasks to processors in a way that is orthogonal to correctness, thereby enabling easy porting and tuning of Legion applications to new architectures to achieve performance.

The Legion team is focusing on developing new and modified features and integrating them into their programming system to address application requirements unique to the ECP, including better support for complex data structures, scalable data partitioning mechanisms, more versatile decomposition into different forms of parallelism, and more flexible and performant mechanisms to map computations and data to hardware.

Progress to date

- The Legion team provided regular releases of the software that reflect bug fixes, new features, performance improvements, and target system support. The features released are dependent upon testing, evaluation, and input from application teams.
- The team demonstrated significant performance improvements on real-world applications, up to a 7× performance increase over the baseline MPI version of a combustion simulation (S3D) and up to a 2.5× performance increase over the MPI+OpenACC version, and the ability to conduct experiments previously out of reach.
- The team obtained up to a 3× performance improvement in the training time for machine learning models.

PI: Pat McCormick, Los Alamos National Laboratory

Collaborators: Los Alamos National Laboratory, Stanford University, SLAC National Accelerator Laboratory, Argonne National Laboratory, NVIDIA

PaRSEC: Distributed Tasking at Exascale

One difficulty associated with programming exascale systems is expressing the tasks comprising a scientific simulation and then mapping them to the heterogenous computational resources on that system, while achieving high performance. PaRSEC supports the development of domain-specific languages and tools to simplify and improve the productivity of scientists when using a task-based system and provides a low-level runtime that seamlessly leverages the combined computing power of accelerators and manycore processors at any scale when executing the tasks.

PaRSEC helps application developers express dataflow parallelism using domain-specific languages and tools and then maps and executes the resultant program on exascale systems with heterogenous computational and memory resources. The team's interaction with scientists includes building domain-specific languages that both suit their needs and facilitate the expression of algorithmic parallelism with familiar constructs. The runtime maps the resultant tasks to the hardware and supports heterogeneous architectures and accelerators and data transfers between different memory hierarchies.

The PaRSEC team focuses on (1) increasing programming flexibility using domain-specific languages benefiting from optimized runtime components, architecture-aware coverage of all target architectures, and reduction of overheads; (2) extending the programming system to new composable paradigms; and (3) providing a production-quality runtime with documentation, testing, packaging, and deployment. This work enables libraries and applications developed by the ECP to efficiently use exascale systems in a pure dataflow programming environment, whereas the domain scientists focus mainly on algorithmic aspects and leave the architectural details and optimizations, such as overlapping of communication/computation and data movement, to the runtime supporting the programming paradigm.

The PaRSEC team has improved their runtime on multiple levels. At the low level, key elements have been modularized and exposed for end-user control. Node-level task schedulers and GPU managers have been designed that support hyperthreading to offload scheduling decisions. The communication subsystem has been extended to take advantage of remote memory access hardware support and to improve the general performance of distributed applications. Critical limitations on the internal representation of the tasks tracking and dependencies tracking have been removed by opting for scalable, efficient, open addressable data structures suitable for shared memory parallelism on many-core architectures. Support for heterogeneous hardware has been improved and includes better memory management strategies, which allow tackling problems many times larger than the available memory on the accelerators without a significant performance penalty. Proof-of-concept integrations with libraries and applications supported by the ECP show promising performance at large scale.

PI: George Bosilca, University of Tennessee – Knoxville

Collaborators: University of Tennessee – Knoxville

Progress to date

- The PaRSEC runtime has been continuously improved to support the exascale architectures and has been integrated with other program models/frameworks and with performance and correctness tools. These new capabilities have been evaluated in a distributed heterogeneous environment.
- The team has also designed programmatic interfaced-to-prefetch data on accelerators that provide memory management advice to the accelerator engine to improve scalability and performance.
- The team has dedicated effort to improve to software quality and usability. Tutorial material has been created to facilitate user adoption, along with developer and user documentation. To ensure that users have reliable access to all capabilities of the runtime system, continuous integration tools have been streamlined in the development process.

Pagoda: UPC++/GASNet

A computation being performed on one part of a large system often needs to access or provide data to another part of the system in order to complete a scientific simulation. The Partitioned Global Address Space (PGAS) model provides the appearance of shared memory accessible to all the compute nodes while implementing this shared memory behind the scenes using physical memory local to the nodes and primitives, such as remote direct memory access. The Pagoda project is developing a performant PGAS programming system to be deployed on exascale systems.

The Pagoda project is developing a programming system to support exascale application development using the PGAS model, with a focus on supporting irregular applications and data structures. There are two components to Pagoda: (1) a portable, high-performance, global-address-space communication library and (2) a template library that provides convenient methods for access and using the global address space. Together, these components enable the agile, lightweight communications that occur in applications, libraries, and frameworks running on exascale systems.

Pagoda enables effective scaling by minimizing the work funneled to heavyweight cores, avoiding the overhead of long, branchy serial code paths and supporting efficient fine-grained communication for both single- and multi-threaded environments. The importance of these properties is exacerbated by application trends; many applications in the ECP require the use of adaptive meshes, sparse matrices, dynamic load balancing, or similar techniques. Pagoda's low-overhead communication mechanisms can maximize the injection rate and network utilization, tolerate latency through overlap, streamline unpredictable communication events, minimize synchronization, and efficiently support small- to medium-sized messages arising in such applications. Pagoda complements other programming models, enabling developers to focus their efforts on optimizing performance-critical communications.

The Pagoda team is focusing on developing new features that will support application and library requirements unique to the ECP and performance improvements that will enable the ECP software stack to exploit the best-available communication mechanisms, including novel features being developed by vendors, such as remote direct memory access mechanisms offered by network hardware and on-chip communication between distinct address spaces.

Progress to date

- The Pagoda team provides regular releases of their communication and template library that typically include new features and performance improvements.
- The team delivered new features in the template library, including support for subset teams and collectives and features for expressing data movement between processors and accelerators.
- The communication library developed by the team supports remote direct memory access, remote procedure calls, future-based contracts, and remote atomics with offload to network hardware.

PI: Erich Strohmaier, Lawrence Berkeley National Laboratory

Collaborators: Lawrence Berkeley National Laboratory

SICM: Simplified Interface to Complex Memory

Exascale systems will have complex, heterogenous memories that need to be effectively managed either directly by the programmer or by the runtime in order to achieve high performance. Natively supporting each memory technology is challenging, as each has its own separate programming interface. The SICM project addresses the emerging complexity of exascale memory hierarchies by providing a portable, simplified interface to complex memory that application programmers and library developers can use to achieve their performance goals.

The SICM project is creating a universal interface for discovering and managing complex memory hierarchies and sharing resources within them. Memory technologies to be supported include, for example, high-bandwidth memory associated with accelerators, nonvolatile memory, 3D stacked memory, and phase-change memory. The result of the SICM project will be a portable, simplified memory interface and software library that will allow application programmers, library developers, and vendors to use these emerging memory technologies without having to program to each technology-specific programming interface.

The SICM team will provide a unified, two-tier node-level complex memory interface. The low-level interface will allow full control of what memory types are being used and is meant for expert developers. This interface will provide support for discovery, allocation/de-allocation, partitioning, and configuration of the memory hierarchy and information on the properties of each specific memory, such as capacity, latency, bandwidth, volatility, and power usage.

The high-level interface will enable developers to define coarser-level constraints on the types of memories needed and leave out the details of the memory management. This interface would potentially be a catalyst for more research as intelligent allocators, migrators, and profiling tools are developed. The high-level interfaces will leverage the low-level interface and library to further decouple applications and libraries from hardware configurations. Specifically, it will emphasize ease of use by developers with a policy/intent–driven syntax enabled through runtime intelligence and system support. Developers will specify which attributes are a priority for each allocation, and the interface will provide the most appropriate configuration.

The impact of the SICM project will be immediate and wide reaching, as developers in all areas are struggling to add support for new memory technologies, and the simplified interface to complex memory can alleviate these challenges.

Progress to date

- The SICM team prototyped and delivered an initial low-level library that supports high-bandwidth memory associated with accelerators and nonvolatile memory.
- The SICM team has provided a high-level interface that supports graph-type allocations on block-based nonvolatile memory devices and produced tools for performance analysis of data structure to inform developers of the potential benefits of migrating those data structures to high-bandwidth memory.
- The SICM team has been working with OpenMP to support the generation of SICM library calls from OpenMP directives to ease adoption of SICM and provide access to heterogenous memory with minimal code changes.

PI: Mike Lang, Los Alamos National Laboratory

Collaborators: Los Alamos National Laboratory, Lawrence Livermore National Laboratory, Oak Ridge National Laboratory, Sandia National Laboratories

OMPI-X

The Message Passing Interface (MPI) is a community standard for inter-process communication and is used by the majority of DOE's parallel scientific applications running on pre-exascale systems. The MPI standard can be implemented on all the large systems. The OMPI-X project ensures that the MPI standard and its specific implementation in Open MPI meet the needs of the ECP community in terms of performance, scalability, and capabilities.

Since its inception, the MPI standard has evolved in response to the changing needs of massively parallel libraries and applications, as well as the systems on which they are run. With the impending exascale era, the pace of change and growing diversity and complexity of architectures pose new challenges that the MPI standard must address. The OMPI-X project team is active in the MPI Forum standards organization and works within it to raise and resolve key issues facing exascale applications and libraries.

The OMPI-X team also developed Open MPI, an open-source, community-based implementation of the MPI standard that is freely available and used by several prominent vendors as the basis for their commercial MPI offerings. The OMPI-X team is focused on prototyping and demonstrating exascale-relevant proposals under consideration by the MPI Forum, as well as improving the fundamental performance and scalability of Open MPI, particularly for exascale-relevant platforms and job sizes. MPI users will be able to take advantage of these enhancements simply by linking against recent builds of the Open MPI library.

In addition to Open MPI, the OMPI-X project will deliver two more products. The Process Management Interface—Exascale (PMIx) is a specification and reference implementation that Open MPI relies upon for the underlying startup and wire-up of the processes involved. It also provides key capabilities that can underpin work on runtime. Qthreads is a library for lightweight user-level threads which, as part of the OMPI-X project, is being integrated into MPI implementations to improve support for and performance of threading within MPI libraries.

Progress to date

- The OMPI-X team has delivered performance and scalability enhancements to the Open MPI implementation. Improvements have been made to the remote memory access implementation to provide both improved performance and scalability. The team has prototyped an improved message matching implementation that can provide up to 2× performance improvement and memory savings. Progress has been made on incorporating topology and congestion awareness.

- The team has demonstrated that Qthreads can achieve equivalent performance to OpenMP.

- The team has made a concerted effort to enhance the quality assurance and testing of the products of this project, including improvements to the Open MPI testing and continuous integration infrastructure, deployment of that testing infrastructure on pre-exascale platforms, and the addition of tests to the test suite that are relevant for exascale libraries and applications.

PI: David Bernholdt, Oak Ridge National Laboratory

Collaborators: Oak Ridge National Laboratory, Los Alamos National Laboratory, Lawrence Livermore National Laboratory, Sandia National Laboratories, University of Tennessee – Knoxville

Kokkos/RAJA

Exascale systems are characterized by computer chips with a large number of cores, a smaller amount of memory, and a range of various architectures, which can result in decreased productivity for library and application developers who need to write specialized software for each system. The Kokkos/RAJA project provides high-level abstractions for expressing the necessary parallel constructs that are then mapped onto a runtime to achieve portable performance across current and future architectures, freeing developers who adopt these technologies of the burden of writing specialized code for each system.

Library and application developers are confronted with the challenges of inventing new parallel algorithms for many-core chips while learning the different programming mechanisms for each architecture and creating and maintaining specialized performant code for each. Adapting libraries and application software as the architectures evolve and become more complex to attain improved performance is a large time investment. The purpose of the Kokkos/RAJA project is to provide portable abstractions that can be adopted by developers to reduce or eliminate this overhead and improve developer productivity.

Kokkos provides a C++ parallel programming model for performance portability that is implemented as a C++ abstraction layer including both parallel execution and data management primitives. RAJA provides various C++ abstractions for parallel loop execution and supports constructs to reorder, aggregate, tile, and partition loop iterations and complex loop-kernel transformations. RAJA's companion projects Umpire and CHAI provide portable memory management and smart data motion capabilities. Application and library developers can implement their code using Kokkos/RAJA, which will map their parallel algorithms onto the underlying execution mechanism using existing parallel programming models, such as OpenMP.

The Kokkos/RAJA team is focused on developing and optimizing backends to support the Aurora and Frontier systems. These backends will ensure that libraries and applications built with the Kokkos/RAJA abstractions will run and achieve high performance on these exascale systems without requiring the library and application developers to change their code, even if these architectures require their own custom programming mechanism.

PI: Christian Trott, Sandia National Laboratories

Collaborators: Sandia National Laboratories, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, Oak Ridge National Laboratory

Progress to date

- The Kokkos team developed a parallel programming model with flexible enough semantics that it can be mapped on a diverse set of exascale architectures including current multi-core CPUs and massively parallel GPUs.
- The Kokkos library implementation consists of a portable Application Programming Interface (API) and architecture-specific backends, including OpenMP, Intel Xeon Phi, and CUDA on NVIDIA GPUs.
- The RAJA team produced a collection of C++ software abstractions that enable architecture portability for exascale applications using standard C++11 features and provided support for multiple backends including OpenMP, CUDA, Intel TBB, and AMD GPUs.
- The Kokkos/RAJA team developed training material and held training events to enable adoption of their abstractions.

Argo

The operating system provides necessary functionality to libraries and applications, such as allocating memory and spawning processes, and manages the resources on the nodes in an exascale system. The Argo project is building portable, open-source system software that improves performance and scalability and provides increased functionality to exascale libraries, applications, and runtime systems, with a focus on resource management, memory management, and power management.

Many exascale applications have a complex runtime structure, ranging from in situ data analysis, through an ensemble of largely independent individual subjobs, to arbitrarily complex workflow structures. To meet the emerging needs of exascale workloads, while providing optimal performance and resilience, the compute, memory, and interconnect resources must be managed in cooperation with applications, libraries, and runtime systems. The goal of Argo is to augment and optimize low-level system software components for use in production exascale systems, providing portable, open-source, integrated software that improves the performance and scalability of and that offers increased functionality to exascale applications, libraries, and runtime systems. The project focuses on resource management, memory management, and power management.

The Argo team is delivering resource management infrastructure to coordinating static allocation and dynamic management of node resources, such as memory and caches. By offloading system-specific aspects such as topology mapping and partitioning of massively parallel resources, this infrastructure will improve the performance and portability of exascale applications and libraries and their runtimes.

They are developing memory management libraries to provide flexible and portable memory management mechanisms that make it easier to obtain high performance and to incorporate nonvolatile memory into complex memory hierarchies using a memory map approach. These libraries will directly support new applications that analyze large, distributed data sets and make it easier to program heterogenous hardware resources.

They are providing fully integrated, end-to-end infrastructure for power and performance management, including power-aware plugins for resource managers, workflow managers, job-level runtimes, and a vendor-neutral power control library. This infrastructure addresses head-on the challenge of managing the performance of exascale applications on highly power-constrained systems.

PI: Pete Beckman, Argonne National Laboratory

Collaborators: Argonne National Laboratory, Lawrence Livermore National Laboratory

Progress to date

- The Argo team developed an initial version of the unified Node Resource Manager, which provides high-level control of node resources, including initial allocation at job launch and dynamic reallocation at the request of the application and other services. The Node Resource Manager integrates dynamic power control and provides support for tracking and reporting of application progress.

- The team released a first version of UMap, a user-space memory map page fault handler for nonvolatile memory that maps virtual address ranges to persistent data sets and transparently pages in active pages and evicts unused pages.

- The team developed AML, a memory library for the explicit management of deep memory architectures that features a flexible and composable interface, allowing applications to implement algorithms similar to out-of-core for deep memory. Multiple optimized versions of these memory migration facilities, using synchronous and asynchronous interfaces and single- and multi-threaded backends, were included.

- The team released an interface between the Node Power and Node Resource Manager services, which in turn allows their Global Resource Manager to control and monitor power and other node-local resources. Additionally, the team studied the effect of power capping on different applications using the Node Power interface and developed the power regression models required for a demand-response policy.

SOFTWARE PORTFOLIO

Development Tools

- EXA-PAPI++
- HPCToolkit
- PROTEAS-TUNE
- SOLLVE
- Flang

EXA-PAPI++

Objective: Develop a standardized interface to hardware performance counters

Principal Investigator: Jack Dongarra, University of Tennessee – Knoxville

HPCToolkit

Objective: Develop an HPC Tool Kit for performance analysis

Principal Investigator: John Mellor-Crummey, Rice University

PROTEAS-TUNE

Objective: Develop a software tool chain for emerging architectures

Principal Investigator: Jeffrey Vetter, Oak Ridge National Laboratory

SOLLVE

Objective: Develop/enhance this OpenMP programming model

Principal Investigator: Barbara Chapman, Brookhaven National Laboratory

Flang

Objective: Develop a Fortran front-end for LLVM

Principal Investigator: Pat McCormick, Los Alamos National Laboratory

EXA-PAPI++

Understanding the performance characteristics of exascale applications is necessary in order to identify and address the barriers to achieving performance goals. This becomes more difficult as the architectures become more complex. The Performance Application Programming Interface (PAPI) provides both library and application developers with generic and portable access to low-level performance counters found across the exascale machine, enabling users to see the relationships between software performance and hardware events. These relationships provide a critical step toward improving performance.

The Exascale Performance Application Programming Interface (Exa-PAPI++) project is developing a new C++ Performance API (PAPI++) software package from the ground up that offers a standard interface and methodology for using low-level performance counters in CPUs, GPUs, on/off-chip memory, interconnects, and the I/O system, including energy/power management. PAPI++ is building upon classic-PAPI functionality and strengthening its path to exascale with a more efficient and flexible software design, one that takes advantage of C++'s object-oriented nature but preserves the low-overhead monitoring of performance counters and adds a vast testing suite.

In addition to providing hardware counter-based information, a standardizing layer for monitoring software-defined events (SDE) is being incorporated that exposes the internal behavior of runtime systems and libraries, such as communication and math libraries, to the applications. As a result, the notion of performance events is broadened from strictly hardware-related events to include software-based information.

Enabling monitoring of both hardware and software events provides more flexibility to developers when capturing performance information.

In summary, the Exa-PAPI++ team is preparing PAPI support to stand up to the challenges posed by exascale systems by (1) widening its applicability and providing robust support for exascale hardware resources; (2) supporting finer-grain measurement and control of power, thus offering software developers a basic building block for dynamic application optimization under power constraints; (3) extending PAPI to support software-defined events; and (4) applying semantic analysis to hardware counters so that the application developer can better make sense of the ever-growing list of raw hardware performance events that can be measured during execution. The team will be channeling the monitoring capabilities of hardware counters, power usage, software-defined events into a robust PAPI++ software package. PAPI++ is meant to be PAPI's replacement—with a more flexible and sustainable software design.

PI: Jack Dongarra, University of Tennessee – Knoxville

Collaborators: University of Tennessee – Knoxville

Progress to date

- On the software event front, the team began with the design and implementation of a new API to expose any kind of software-defined events. It extends PAPI's role so that it becomes the de facto standard for exposing performance-critical events from different software layers.

- Because the concept of software-defined events is new to PAPI, the team worked closely with developers of different libraries and runtimes that serve as natural targets for early adoption of the new SDE API. To date, the team has integrated SDEs into the sparse linear algebra library MAGMA-Sparse, the tensor algebra library TAMM (NWChemEx), the task-scheduling runtime PaRSEC, and the compiler-based performance analysis tool BYFL.

- On the hardware counter front, the team has developed a new PAPI component called "PCP" for IBM POWER9 hardware counters. It adds support for (1) core performance events, which are specific to each core, and (2) shared events, which monitor the performance of node-wide resources that are shared between cores. Access to shared events requires elevated privileges. However, IBM's official route for providing access to shared events is through the Performance Co-Pilot (PCP) for non-root users. The new PAPI-PCP component enables all users to access POWER 9 shared events through PAPI.

HPCToolkit

Exascale machines will be highly complex systems that couple multicore processors with accelerators and share a deep, heterogeneous memory hierarchy. Understanding performance bottlenecks within and across the nodes in extreme-scale computer systems is a first step toward mitigating them to improve library and application performance. The HPCToolkit project is providing a suite of software tools that developers need to measure and analyze the performance of their software as it executes on today's supercomputers and forthcoming exascale systems.

In recent years, the complexity and diversity of architectures for extreme-scale parallelism have dramatically increased. At the same time, the complexity of applications is also increasing as developers struggle to exploit billion-way parallelism, map computation onto heterogeneous computing elements, and cope with the growing complexity of memory hierarchies. While library and application developers can employ abstractions to hide some of the complexity of emerging parallel systems, performance tools must assess how software interacts with each hardware component of these systems.

The HPCToolkit project is working to develop performance measurement and analysis tools to enable application, library, runtime, and tool developers to understand where and why their software does not fully exploit hardware resources within and across nodes of current and future parallel systems. To provide a foundation for performance measurement and analysis, the project team is working with community stakeholders, including standards committees, vendors, and open-source developers, to improve hardware and software support for measurement and attribution of application performance on extreme-scale parallel systems.

The HPCToolkit team is focused on influencing the development of hardware and software interfaces for performance measurement and attribution by community stakeholders; developing new capabilities to measure, analyze, and understand the performance of software running on extreme-scale parallel systems; producing a suite of software tools that developers can use to measure and analyze the performance of parallel software as it executes; and working with developers to ensure that HPCToolkit's capabilities meet their needs. Using emerging hardware and software interfaces for monitoring code performance, the team is working to extend capabilities to measure computation, data movement, communication, and I/O as a program executes to pinpoint scalability bottlenecks, quantify resource consumption, and assess inefficiencies, enabling developers to target sections of their code for performance improvement.

Progress to date

- The team developed novel capabilities for measurement, analysis, and attribution of applications that employ GPU accelerators. Today, HPCToolkit can report performance about accelerated applications in source-code centric profiles views and time-centric visualizations of an execution's dynamics.
- To relate performance measurements of accelerated applications back to source code constructs, the team improved HPCToolkit's ability to recover control flow graphs from machine code, which enabled HPCToolkit to relate application performance to inline functions, templates, and loops in highly optimized code on both host processors and accelerators.
- The team added a new measurement substrate to HPCToolkit to measure code performance using the native Linux performance monitoring substrate known as the perf events interface. In addition to measuring application performance, Linux perf events enable HPCToolkit to measure operating system activity and thread blocking.
- The team developed support for handling programming models with short-lived dynamic threads.

PI: John Mellor-Crummey, Rice University

Collaborators: Rice University, University of Wisconsin – Madison

PROTEAS-TUNE

Programmer productivity and performance portability are two of the most important challenges facing users of exascale architectures that include heterogeneous compute nodes, deep memory hierarchies, and persistent memory. Library and application developers targeting these architectures will find it increasingly difficult to meet these two challenges without integrated capabilities that allow for flexibility, composability, and interoperability across a mixture of programming, runtime, and architectural components. The PROTEAS-TUNE project is developing a set of programming technologies that will provide developers with portable programming solutions for exascale architectures.

The PROTEAS-TUNE project focuses on performance portability and productivity across increasingly diverse and complex architectures. Key capabilities include support for heterogeneous computing; performance analysis; autotuning; programming nonvolatile memory; code transformations; and just-in-time compilation. In particular, the PROTEAS-TUNE team is developing and contributing several critical pieces of infrastructure and optimizations to enable application portability and high performance on exascale architectures to the community LLVM compiler project.

The PROTEAS-TUNE team is (1) improving the core-LLVM compiler ecosystem; (2) designing and implementing the OpenACC heterogeneous programming model for LLVM; (3) using performance modeling and optimization to enable code transformation and performance portability; (4) refining autotuning for OpenMP and OpenACC programming models in order to directly target challenges with heterogeneous architectures; (5) improving performance measurement and analysis tools for exascale architectures and applying them to improve application performance; (6) developing and implementing portable software abstractions for managing persistent memory; and (7) aggressively engaging library and application developers to use their technologies.

Progress to date

- The PROTEAS-TUNE team used modeling and performance optimization on proxy-apps to evaluate application kernel speedup on pre-exascale architectures, which has driven improvements in threading, data layout, communication, and specialized hardware capabilities.
- The team built and demonstrated an initial implementation of the OpenACC heterogeneous programming model for LLVM.
- The team added multiple new capabilities to their performance measurement and analysis tools in order to support accelerator profiling, new programming models and languages, and workflows.
- The team developed and implemented a portability abstraction for using nonvolatile memory on pre-exascale architectures.

PI: Jeffrey Vetter, Oak Ridge National Laboratory

Collaborators: Oak Ridge National Laboratory, Los Alamos National Laboratory, Argonne National Laboratory, Lawrence Berkeley National Laboratory, University of Utah, University of Oregon

SOLLVE

Exascale architectures are expected to feature a dramatic increase in the amount of intra-node threading, greater heterogeneity, and more complex hierarchical memory subsystems. OpenMP is a directive-based standard specification and runtime for programming shared-memory and accelerator systems and is used by many exascale applications for in-node programming. The SOLLVE project is advancing the OpenMP specification to address exascale application exascale challenges including programmability gaps for core technologies such as accelerator support, interoperability with MPI, and data migration of complex data structures.

OpenMP is a popular tool for in-node programming and is supported by a strong community including vendors, national labs, and academic groups. Most ECP applications include OpenMP as part of their strategy for reaching exascale levels of performance. Several application teams identified gaps in OpenMP functionality with respect to movement of complex data structures to/from accelerator memories, some require compatibility with the latest C++ standards, and others expect the ability to generate performance portable code. The SOLLVE project is working with application partners and the members of the OpenMP language committee to extend the OpenMP feature set to meet these application needs.

The SOLLVE team is focused on delivering a high-quality, robust implementation of OpenMP and project extensions in LLVM; developing the LLVM BOLT runtime system to exploit light-weight threading for scalability and interoperability with MPI; and creating a validation suite to ensure that quality implementations of OpenMP are being delivered. The team directly interacts with end users to understand and consolidate their application software needs, allowing them to drive and prioritize features in the OpenMP standard, with the goal of delivering the best possible solutions for functionality and performance gaps. They also engage with key vendors and the broader OpenMP community to seek the best-possible solutions to exascale application challenges, aiming to secure their adoption in new versions of the standard and to address scalability requirements in the implementation.

PI: Barbara Chapman, Brookhaven National Laboratory

Collaborators: Brookhaven National Laboratory, Oak Ridge National Laboratory, Lawrence Livermore National Laboratory, Georgia Institute of Technology

Progress to date

- The SOLLVE team has introduced substantial extensions to the OpenMP 4.5 and 5.0 specifications including features to facilitate movement of complex data structures to accelerators and offering more control on the work-sharing directives.
- The team designed and implemented many enhancements to the LLVM compiler and OpenMP runtime implementation including optimizations for parallel regions, improved code generation, and overall data movement optimization. Runtime enhancements concentrated on improving the interoperability of the BOLT runtime with several MPI implementations and providing new data locality and scheduling heuristics.
- The team produced a beta version of their validation suite that supports assessment of compliance to the OpenMP standard and performance of the delivered OpenMP implementation.

Flang

The Fortran programming language is an essential component of many exascale applications and broad scientific missions within the US Department of Energy (DOE). Until recently, Fortran has not had the benefit of using the widely leveraged LLVM Compiler Infrastructure and the vibrant community that supports it. By leveraging a multi-year investment made by the National Nuclear Security Administration's (NNSA's) Advanced Simulation and Computing (ASC) program to establish Fortran as a component of the LLVM infrastructure, the Flang project aims to build upon this foundation and further the capabilities and feature set of Fortran as a first-class language in the LLVM community. This will provide developers with a viable and robust path forward for producing performant Fortran-based applications on DOE's pre-exascale and exascale system architectures.

The Flang project focuses on taking a key role in contributing to the recently accepted open-source LLVM-based Fortran front-end ("Flang") established by a multi-year investment made by the NNSA's ASC program. By contributing to the overall community effort, the infrastructure will provide support for the critical features needed by Fortran applications to obtain performance on pre-exascale and exascale architectures, such as accelerator offload, improved optimizations, and tooling. This effort will help to establish a modernized Fortran environment that will provide a robust and productive infrastructure for mission-critical applications within the DOE and across other US agencies and industry, where Fortran applications are essential for national security, scientific discovery and engineering design.

Progress to date

Project officially starts October 1, 2019

PI: Patrick McCormick, Los Alamos National Laboratory

Collaborators: Los Alamos National Laboratory, Argonne National Laboratory, Lawrence Berkeley National Laboratory, Oak Ridge National Laboratory, NVIDIA

SOFTWARE PORTFOLIO

Mathematical Libraries

- xSDK4ECP
- PETSc/TAO
- STRUMPACK/SuperLU
- SUNDIALS-*hypre*
- CLOVER
- ALExa

xSDK4ECP

Objective: Create a value-added aggregation of DOE math libraries to combine usability, standardization, and interoperability

Principal Investigator: Ulrike Yang, Lawrence Livermore National Laboratory

PETSc/TAO

Objective: Deliver efficient libraries for sparse linear and nonlinear systems of equations and numerical optimization

Principal Investigator: Barry Smith, Argonne National Laboratory

STRUMPACK/ SuperLU

Objective: Provide direct methods for linear systems of equations and Fourier transformations

Principal Investigator: Sherry Li, Lawrence Berkeley National Laboratory

SUNDIALS-*hypre*

Objective: Deliver adaptive time-stepping methods for dynamical systems and solvers

Principal Investigator: Carol Woodward, Lawrence Livermore National Laboratory

CLOVER

Objective: Develop scalable, portable numerical algorithms to facilitate efficient simulations

Principal Investigator: Jack Dongarra, University of Tennessee – Knoxville

ALExa

Objective: Provide technologies for passing data among grids, computing surrogates, and accessing mathematical libraries from Fortran

Principal Investigator: John Turner, Oak Ridge National Laboratory

xSDK4ECP

Extreme-Scale Scientific Software for ECP

The large number of software technologies being delivered to the application developers poses challenges, especially if the application needs to use more than one technology at the same time, such as using a linear solver from the PETSc/TAO mathematics library in conjunction with a time integrator from the SUNDIALS library. The xSDK project is an effort to create a value-added aggregation of mathematics and scientific libraries, to increase the combined usability, standardization, and interoperability of these libraries.

The ability to incorporate multiple libraries in a single executable is necessary as architectures become more complex and applications become reliant on multiple libraries to supply performant capabilities on those architectures to achieve their exascale performance and science goals. The Extreme-scale Scientific Software Development Kit (xSDK) is an effort to provide turnkey installation and use of popular scientific packages needed for next-generation scientific applications. The xSDK project is working to (1) enable the seamless combined use of diverse, independently developed numerical libraries as needed by exascale applications; (2) develop interoperability layers among numerical libraries in order to improve code quality, access, usability, interoperability, and sustainability; and (3) provide an aggregate build and install capability for the numerical libraries that supports hierarchical, modular installation.

The xSDK project focuses on community development and a commitment to combined success via quality improvement policies, better build infrastructure, and the ability the use numerical libraries in combination to solve large-scale multiphysics and multiscale problems. The project represents a different approach to coordinating library development and deployment. Prior to the xSDK, scientific software packages were cohesive with a single team effort but not across these efforts. The xSDK goes a step further by developing community policies followed by each independent library included in the xSDK. This policy-driven, coordinated approach enables independent development that still results in compatible and composable capabilities. Moreover, the xSDK provides a forum for collaborative numerical library development, helping independent teams to accelerate adoption of best practices, enabling interoperability of independently developed libraries, and improving developer productivity and sustainability of the libraries.

The xSDK project will also begin a coordinated effort to investigate and deploy multiprecision functionality in the ECP ST ecosystem to enable the use of low-precision hardware function units, reduce the pressure on memory and communication interfaces, and achieve improved performance. The project will assess current status and functionalities, advance the theoretical knowledge on multiprecision algorithms, design prototype implementations and multiprecision interoperability layers, deploy production-ready multiprecision algorithms in the xSDK math libraries, ensure multiprecision cross-library interoperability, and integrate multiprecision algorithms into ECP application projects.

PI: Ulrike Meier Yang, Lawrence Livermore National Laboratory

Collaborators: Lawrence Livermore National Laboratory, Argonne National Laboratory, Sandia National Laboratories, Lawrence Berkeley National Laboratory, University of California – Berkeley, University of Tennessee – Knoxville, University of Oregon, NexGen Analytics, Oak Ridge National Laboratory, Karlsruhe Institute of Technology, University of Manchester, Charles University at Prague, Tech-X Corporation

Progress to date

- The xSDK team released version 0.4.0, which included 13 new xSDK members (AMRex, deal.II, DTK, MAGMA, MFEM, Omega_h, PHIST, PLASMA, PUMI, SLEPc, STRUMPACK, SUNDIALS, and Tasmanian) in addition to the original xSDK libraries (hypre, PETSc, SuperLU, and Trilinos), and the two domain components Alquimia and PFLOTRAN.

- The team continued development of the community policies. They refreshed the policies and added a new recommended policy including feedback from the ECP community. The policies were moved to github, and the process on changing or proposing policies has been updated.

- The team created reports on node-level resource management, which included survey results of efforts and future plans for the efficient transfer of resources of runtime library developers and approaches and plans of xSDK packages on their use of programming models and transfer of data resources; on the design, approach, and impact of the xSDK, summarizing history, community policies, release processes, library interoperability, xSDK usage, and impact on applications; and on the end-to-end use of the xSDK in three exascale applications.

- The team also interviewed many application teams on their needs of mathematical capabilities, computer usage, library usage, training, and more, to guide further xSDK development.

PETSc/TAO

Many application codes rely on high-performance mathematical libraries to solve the systems of equations that must be solved during their simulation. Because the solvers often dominate the computation time of such simulations, these libraries must be efficient and scalable on the upcoming complex exascale hardware architectures for the application codes to perform well. The PETSc/TAO project delivers efficient mathematical libraries to application developers for sparse linear and nonlinear systems of equations, time-stepping methods, and parallel discretization techniques and provides libEnsemble to manage the running of a large collection of related simulations needed for numerical optimization, sensitivity analysis, and uncertainty quantification (the so-called outer-loop).

Algebraic solvers, generally nonlinear solvers that use sparse linear solvers via Newton's method, and integrators form the core computation of many scientific simulations. The Portable Extensible Toolkit for Scientific Computations/Toolkit for Advanced Optimization (PETSc/TAO) is a scalable mathematical library that runs portably on everything from laptops to the existing pre-exascale machines. The PETSc/TAO project is extending and enhancing the library to ensure that it will be performant on exascale architectures, is delivering the libEnsemble tool to manage a collection of related simulation for outer-loop methods, and is working with exascale application developers to satisfy their solver needs.

There are no scalable "black box" sparse solvers or integrators that work for all applications, nor single implementations that work well for all scales of problem size. Hence, algebraic solver packages provide a wide variety of algorithms and implementations that can be customized for the application and range of problem sizes at hand. The PETSc/TAO team is currently focusing on enhancing the PETSc/TAO library to include scalable solvers that efficiently utilize many-core and GPU-based systems. This work includes implementing reduced synchronization algorithms that scale to larger concurrency than solvers with synchronization points and performance and data structure optimizations for the basic data structures to better utilize many-core and GPU-based computing systems as well as provide scalability to the exascale.

The availability of systems with over 100 times the processing power of today's machines compels the use of these systems, not just for a single simulation but rather within a tight outer-loop of numerical optimization, sensitivity analysis, and uncertainty quantification. This requires the implementation of a scalable library for managing a dynamic hierarchical collection of running, possibly interacting, scalable simulations. The library libEnsemble directs such multiple concurrent simulations. In this area, our team is focused on the development of libEnsemble, the integration of libEnsemble with the PETSc/TAO library, and extension of the PETSc/TAO library to include new algorithms capable of using libEnsemble.

PI: Barry Smith, Argonne National Laboratory

Collaborators: Argonne National Laboratory

Progress to date

- The PETSc/TAO team delivered PETSc/TAO version 3.11, which includes pipeline Krylov implementations; improved GPU support for their algebraic multigrid solver; new data structures and a new communication module backend that uses the star forest paradigm to improve performance; portability support for algebraic multigrid on GPUs; and support for multithreading in third-party libraries that use OpenMP.

- The team also delivered libEnsemble version 0.4.1, which includes an option to run libEnsemble in central or distributed configurations; updated tests, examples, and documentation; and testing using the POUNDERs and APOSSM numerical optimization solvers.

- The team also completed preliminary testing and benchmarking to confirm that the GPU backends in PETSc/TAO are working correctly and delivering performance gains for single-node configuration cases on Summit and continued benchmarking libEnsemble on Summit.

- The PETSc/TAO team shares overlapping membership with the ECP Center for Efficient Exascale Discretizations (CEED) co-design center and is working closely with them on common issues including the use of high-order matrix-free discretizations (libCEED) and scalable mesh management techniques. In addition, the team is collaborating with the AMReX co-design center on using both the TAO optimization algorithms and the PETSc iterative solvers from AMReX applications. PETSc/TAO/libEnsemble is currently used by least nine software components being developed by the ECP application teams.

STRUMPACK/SuperLU/FFTX

Many simulation and data analysis codes need to solve sparse systems of equations. The high-fidelity simulations being solved by exascale application teams involve large-scale multiphysics and multiscale modeling problems that generate highly ill-conditioned and indefinite systems, for which iterative methods struggle. The STRUMPACK/SuperLU/FFTX project is delivering robust and scalable factorization-based algorithms that are indispensable building blocks for solving these numerically challenging problems and a Fourier transform package that is applicable to spectral-based methods used by exascale applications.

Scalable factorization-based methods are important components in solvers for ill-conditioned and indefinite systems of equations that arise in many exascale applications, while performant Fourier transforms are required by applications using certain spectral-based methods. The STRUMPACK/SuperLU/FFTX project is producing robust and scalable factorization-based methods and preconditioners for systems of equations and is providing a Fourier transform software stack suitable for obtaining the highest possible performance on exascale systems.

The team is delivering factorization-based sparse solvers encompassing two widely used algorithm variants, the supernodal SuperLU library and the multifrontal STRUMPACK library. The team is also adding scalable preconditioning functionality using hierarchical matrix algebra to the STRUMPACK library. Both libraries are applicable to a large variety of application domains. These scalable libraries are being enhanced to ensure that they will be performant on the pre-exascale and exascale architectures.

The team will also provide the FFTX library to address the need for Fourier transforms by certain spectral-based methods. This library will use symbolic transformation tools, code generation techniques, and autotuning to create exascale-ready high-level Fourier transform packages for multiple applications that will support multi-GPU and multi-node parallelism.

PI: Sherry Li, Lawrence Berkeley National Laboratory

Collaborators: Lawrence Berkeley National Laboratory, Carnegie Mellon University, SpiralGen, Inc.

Progress to date

- The team released SuperLU DIST version 6.1.0, with improvements in the strong scaling of the triangular solve—up to 4.4× faster than version 5.x on 4000+ cores—and on-node threading optimization, providing up to a 3× speedup on a Cori-KNL node.
- The team released STRUMPACK version 3.1.0, with improvements in the scalability of the hierarchical matrix algorithms—the dense hierarchical matrix compression is up to 4.7× faster on eight nodes of Cori-Haswell and 2.4× faster on Cori-NKL, while the hierarchical sparse factorization is up to 2.2× faster on eight Cori-KNL nodes—and improvements in the hierarchical solve to reduce communication and more OpenMP support, leading to a 7× faster matrix redistribution and 1.4× faster solve.
- The team specified an initial set of FFTX applications use cases, designed the FFTX API version 1.0, and provided a reference implementation targeting FFTW.

SUNDIALS-*hypre*

Time integrators are at the core of every time-dependent simulation application. In addition, many applications require the solution of linear algebraic systems of equations, whether through use of an implicit approach for integrating the time dependence or for solution of steady state systems. The SUNDIALS-*hypre* project is enhancing the SUNDIALS library of numerical software packages for integrating differential systems in time using state-of-the-art adaptive time step technologies and the *hypre* library for solving large systems of linear equations both for use on exascale systems.

Many exascale applications depend on efficient time integrators and linear solvers yet do not use state-of-the-art algorithms and are not able to easily take advantage of algorithmic advances. Through flexible and efficient libraries, applications can more easily take advantage of new algorithms and more efficient implementations that will allow for easier adaptations to exascale architectures. The SUNDIALS-*hypre* project is enhancing the SUNDIALS and *hypre* libraries, which collectively deliver time integrators, nonlinear solvers, linear solvers, and preconditioners, for use in scientific applications running on exascale systems.

SUNDIALS provides both adaptive multistep and multistage time integrators designed to evolve systems of ordinary differential equations and differential algebraic equations. This suite also includes both Newton and fixed-point nonlinear solvers and scaled Krylov methods with hooks for user-supplied data structures and solvers. The SUNDIALS team is extending SUNDIALS to include an efficient time-dependent mass matrix mechanism, a new GPU-enabled approach for solving multiple ODE systems in parallel, a rewrite of an ordinary differential equation integrator that projects solutions on constraint manifolds, integration into ECP applications, and performance assessments and improvements on pre-exascale and exascale systems.

hypre is a software library of high-performance preconditioners and solvers for the solution of large, sparse linear systems of equations on massively parallel computers. The library includes parallel multigrid solvers for both structured and unstructured grid problems and features conceptual interfaces, which include a structured, a semi-structured interface, and a traditional linear-algebra-based interface. The *hypre* team is adding both CUDA and OpenMP 4.x ports of the *hypre* library and is assessing the performance of these ports, examining performance bottlenecks, and developing new variants of algorithms or new algorithms that are better suited for pre-exascale and exascale architectures.

Progress to date

- The SUNDIALS team released new versions of the SUNDIALS suite that include new linear and nonlinear solver APIs that allow easier interfacing with external packages, a new set of optional fused vector kernels which can result in an over 90% reduction in run time for reduction operations, and a new many-vector capability allowing the underlying data structures to be vectors of vectors. The team also supported ECP applications through the development of CUDA vectors with managed memory, optional streams, and more flexibility in memory management from the application.

- The *hypre* team released new versions of the *hypre* library that include a new GMRES solver with improved communication properties, a new integer type for 64-bit integers allowing for a mixed-integer option that uses less memory and is about 20–25% faster than the 64-bit integer version, and GPU-enabled AMG setup components.

PI: Carol Woodward, Lawrence Livermore National Laboratory

Collaborators: Lawrence Livermore National Laboratory, Southern Methodist University

CLOVER

Scientific applications need to apply efficient and scalable implementations of numerical operations, such as matrix-vector products and Fourier transforms, in order to simulate their phenomena of interest. Software libraries are powerful means of sharing verified, optimized numerical algorithms and their implementations. The CLOVER project is delivering scalable, portable numerical algorithms to facilitate efficient simulations. To the extent possible, the team preserves the existing capabilities in mathematical libraries, while evolving the implementations to run effectively on the pre-exascale and exascale systems and adding new capabilities that may be needed by applications.

Mathematical libraries encapsulate the latest results from the mathematics and computer science communities, and many exascale applications rely on these numerical libraries to incorporate the most advanced technologies available in their simulations. Advances in mathematical libraries are necessary to enable computational science on exascale systems, as the exascale architectures introduce new complexities that algorithms and their implementations need to address in order to be scalable, efficient, and robust. The CLOVER project is ensuring the healthy functionality of the mathematical libraries on which these applications depend. The libraries supported by the CLOVER project, SLATE, heFFTe PEEKS, and Kokkos Kernels, span the range from lightweight collections of subroutines with simple application programming interfaces (APIs) to more "end-to-end" integrated environments and provide access to a wide range of algorithms for complex problems.

SLATE provides dense linear algebra operations for large-scale machines with multiple GPU accelerators per node. The team focuses on adding support to SLATE for the most critical workloads required by exascale applications: linear systems, least squares, matrix inverses, singular value problems, and eigenvalue problems.

heFFTe implements the fast Fourier transform used in many domain applications including molecular dynamics, spectrum estimation, fast convolution and correlation, signal modulation, and wireless multimedia applications. The team is designing and implementing a fast and robust 2D and 3D fast Fourier transform library that targets large-scale heterogeneous systems with multi-core processors and hardware accelerators.

PEEKS is delivering production-quality, next-generation latency-tolerant, and scalable preconditioned iterative solvers. The team is producing the design and infrastructure support required to effectively implement these solvers and delivering them using a standardized API.

Kokkos Kernels provides performance portable sparse and dense linear algebra and graph kernels on current and future heterogeneous architectures. The team is delivering architecture-aware, high-performance algorithms for performance-critical kernels to applications for use on pre-exascale and exascale architectures.

PI: Jack Dongarra, University of Tennessee–Knoxville

Collaborators: University of Tennessee–Knoxville, Sandia National Laboratories

Progress to date

- The CLOVER team produced a version of SLATE that supports BLAS 3, norms, linear solvers, mixed-precision linear solvers, and least-squares solvers and includes compatibility APIs for LAPACK and ScaLAPACK users.
- The team completed a design and implementation for heFFTe targeting distributed accelerated systems that includes various technologies for scheduling computation and communications, highly optimized GPU kernels, and CUDA-aware MPI routines.
- The team deployed and benchmarked the PEEKS implementation of the parallel generation of preconditioners based on incomplete factorization, developed a parallel threshold ILU factorization, and incorporated pipeline Krylov solvers.
- The team is porting Kokkos Kernels to run efficiently on the pre-exascale machines and is adding methods to address the needs identified by exascale applications.

ALExa

Many scientific applications are written in Fortran and need to access scalable algorithms for efficiency, need to pass data between different grids with different parallel distributions, or need reduced representations of high-dimensional data, for example, to optimize storage. The Accelerated Libraries for Exascale (ALExa) project is providing technologies to address these needs for exascale applications.

Complex scientific applications may need to combine results from different computational grids to perform their required simulations, where each computational grid represents only part of the physics. Moreover, the simulations on each grid may be written in Fortran and require access to scalable solvers in C++. The ALExa project is developing three components to address these issues and enable applications to better use exascale systems: the Data Transfer Kit (DTK), Tasmanian, and ForTrilinos.

The DTK provides the ability to transfer computed solutions between grids with different layouts on parallel accelerated architectures, enabling simulations to seamlessly combine results from different computational grids to perform their required simulations. The team is focused on adding new features needed by applications and ensuring that the library is performant on the pre-exascale and exascale architectures.

Tasmanian provides the ability to construct surrogate models with low memory footprint, low cost, and optimal computational throughput, enabling optimization and uncertainty quantification for large-scale engineering problems, as well efficient multi-physics simulations. The team is focused on reducing the GPU memory overhead and accelerating the simulation of the surrogate models produced.

ForTrilinos provides a software capability for easy automatic generation of Fortran interfaces to any C/C++ library, as well as a seamless pathway for large and complex Fortran-based codes to access the Trilinos library through automatically generated interface code.

PI: John Turner, Oak Ridge National Laboratory

Collaborators: Oak Ridge National Laboratory

Progress to date

- The ALExa team demonstrated the DTK performance portable search capability on multi-threaded platforms, which exhibited a 10–15× speedup over standard search libraries.
- The team enabled GPU-accelerated surrogate model simulations in TASMANIAN, developed new algorithms for asynchronous surrogate construction that exploit extreme concurrency, and demonstrated a 100× reduction of memory footprint in sparse representation of neutrino opacities for the ExaStar project.
- The team developed a SWIG/Fortran tool that automatically generates Fortran object-oriented interfaces and necessary wrapper code for any given C/C++ interface, demonstrated advanced inversion-of-control functionality that allows a C++ solver to invoke user-provided Fortran routines, and used this tool to provide Fortran access to a wide variety of linear and nonlinear solvers in the Trilinos library.

SOFTWARE PORTFOLIO

Data and Visualization

- ADIOS
- DataLib
- VTK-m
- VeloC/SZ
- ExaIO
- Alpine/ZFP

ADIOS

Objective: Support efficient I/O and code coupling services

Principal Investigator: Scott Klasky, Oak Ridge National Laboratory

DataLib

Objective: Support efficient I/O, I/O monitoring and data services

Principal Investigator: Rob Ross, Argonne National Laboratory

VTKL-m

Objective: Provide VTK-based scientific visualization software that supports shared memory parallelism

Principal Investigator: Ken Moreland, Sandia National Laboratories

VeloC/SZ

Objective: Develop two software products: VeloC checkpoint restart and SZ lossy compression with strict error bounds

Principal Investigator: Franck Cappello, Argonne National Laboratory

ExaIO

Objective: Develop an efficient system topology and storage hierarchy-aware HDF5 and UnifyFS parallel I/O libraries

Principal Investigator: Suren Byna, Lawrence Berkeley National Laboratory

Alpine/ZFP

Objective: Deliver in situ visualization and analysis algorithms, infrastructure and data reduction of floating-point arrays

Principal Investigator: Jim Ahrens, Los Alamos National Laboratory

ADIOS

Exascale architectures will have complex, heterogeneous memory hierarchies, ranging from node-level caches and main memory all the way to persistent storage via the file system, that applications need to effectively achieve their science goals. At the same time, exascale applications are becoming more complex in their data flows, from multiscale and multiphysics simulations that need to exchange data between separate codes to simulations that invoke data analysis and visualization services to extract information and render it to storing simulation output to the file system for later analysis. The ADIOS project delivers a highly optimized coupling infrastructure that enables efficient synchronous and asynchronous data exchanges to move data between multiple codes running concurrently and to the different layers of the storage system.

The Adaptable I/O Systems (ADIOS) is designed to tackle data management challenges posed by large-scale science applications running on high-performance computers that require, for example, code-to-code coupling for multiphysics and multiscale applications and code-to-service coupling for data analysis and visualization. Notably, ADIOS provides a simple, declarative publish/subscribe input/output interface so that applications can easily describe the data they produce or consume. ADIOS has multiple optimized solutions to transfer data between codes and to the file system.

The ADIOS team is focused on providing high-quality, performant software products to address the data flow requirement of exascale applications on pre-exascale and exascale architectures. Therefore, the team is optimizing and creating new ADIOS microservices to efficiently perform the data transfers needed by applications that can use pre-exascale and exascale architecture features so that the exascale applications can meet their science and performance goals. The framework that the team delivers supports both synchronous and asynchronous data exchanges and provides the capabilities necessary for in situ processing and code coupling and efficient data transfers to the file system.

PI: Scott Klasky, Oak Ridge National Laboratory

Collaborators: Oak Ridge National Laboratory, Lawrence Berkeley National Laboratory, Georgia Institute of Technology, Rutgers University, Kitware, Inc.

Progress to date

- The ADIOS team released ADIOS2, which supports two file-based engines to write and read to/from permanent storage. The BPFile engine provides unmatched I/O performance on pre-exascale machines and supports in situ processing of data by multiple readers, while the HDF5 engine uses the parallel HDF5 library to write and read HDF5 formatted files without any overhead, matching the native library's performance. The BPFile engine is used in several applications for high-performance storage I/O. In particular, the LAMMPS code in the EXAALT project is using ADIOS to dump its atoms at large-scale runs, and ADIOS achieves better performance with a self-describing output that can be processed in situ than its original binary dump.

- The team developed a flexible staging engine (SST) to allow for coupling codes via network communication that can be used for in situ processing where consumers of the data can dynamically connect to and disconnect from the producer's (simulation) output. This engine has been used in code coupling production runs in the WDMApp project.

- The team produced an MPI-based in situ engine for in situ data processing where data are moved from one application to another using asynchronous MPI send/receive operations. This engine is used by applications to perform, for example, in situ data reduction using their ZFP and SZ lossy compression plugins. This engine is optimized for applications where the producer outputs data with a fixed schema at every step and the consumer reads the data with a fixed read pattern.

DataLib

Exascale applications generate massive amounts of data that need to be analyzed and stored to achieve their science goals. The speed at which the data can be written to the storage system is a critical factor in achieving these goals. As exascale architectures become more complex, with multiple compute nodes and accelerators and heterogenous memory systems, the storage technologies must evolve to support these architectural features. The DataLib project is focused on three distinct and critical aspects of successful storage and I/O technologies for exascale applications: enhancing and enabling traditional I/O libraries on pre-exascale and exascale architectures; establishing a nascent paradigm of data services specialized for exascale codes; and working closely with Facilities to ensure the successful deployment of their tools.

The ability to efficiently store data to the file system is a key requirement for all scientific applications. The DataLib project is providing both standards-based and custom storage and I/O solutions for exascale applications on upcoming platforms. The primary goals of this effort are to enable users of the HDF5 standard to achieve the levels of performance seen from custom codes and tools, facilitate the productization and porting of data services and I/O middleware using Mochi technologies, and continue to support application and Facility interactions using DataLib technologies.

HDF5 is the most popular high-level API for interacting with the storage system on high-performance computers. The DataLib team is undertaking a systematic software development activity to deliver an HDF5 API implementation that achieves the highest possible performance on exascale platforms. By adopting the HDF5 API, the team is able to support the I/O needs of all the exascale applications already using this standard.

The Mochi software tool is a building block for user-level distributed data services that addresses performance, programmability, and portability. The Mochi framework components are being used by multiple exascale library and application developers, and the team is engaging with them to customize data services for their needs.

Progress to date

- The DataLib team has improved the ability to understand exascale application I/O performance using their Darshan tool, improved the performance of I/O for codes using their PnetCDF and ROMIO technologies, implemented new capabilities for storing intermediate data on burst buffers and for building custom data services, developed and refined packaging and testing of DataLib software, and supported the use of Mochi technologies in other ECP ST and vendor products.

PI: Rob Ross, Argonne National Laboratory

Collaborators: Argonne National Laboratory, Los Alamos National Laboratory, Northwestern University

VTK-m

As exascale simulations generate data, scientists need to extract information and understand their results. One of the primary mechanisms for understanding these results is to produce visualizations that can be viewed and manipulated. The VTK-m project is developing and deploying scientific visualization software capable of efficiently using exascale architectural features, such as the shared-memory parallelism available on many-core CPUs and GPUs, by redeveloping, implementing, and supporting necessary visualization algorithms.

One of the biggest recent changes in high-performance computing is the increasing use of accelerators. Accelerators contain processing cores that independently are inferior to a core in a typical CPU, but these cores are replicated and grouped such that their aggregate execution provides a very high computation rate at a much lower power. Current and future CPU processors also require much more explicit parallelism. Each successive version of the hardware packs more cores into each processor, and technologies like hyperthreading and vector operations require even more parallel processing to leverage each core's full potential.

The scientific visualization community has been building scalable tools for over 15 years that enable scientists to visualize the results of their simulations. Current tools, however, are based on a message-passing programming model and rely on a coarse decomposition that is known to break down as the level of concurrency increases. The VTK-m project is providing the core capabilities to perform scientific visualization on exascale architectures, thus filling the critical feature gap of performing efficient visualization and analysis on many-core CPU and GPU architectures.

The VTK-m team is providing general-purpose scientific visualization software for exascale architectures that supports shared memory parallelism and fine-grained concurrency. The team is focused on providing abstract models for data and execution that can be applied to a variety of algorithms across many different processor architectures, along with necessary visualization algorithm implementations. The results of this project will be delivered in tools currently used around the world today, like ParaView and VisIt, as well as in a stand-alone form.

PI: Ken Moreland, Sandia National Laboratories

Collaborators: Sandia National Laboratories, Oak Ridge National Laboratory, Los Alamos National Laboratory, University of Oregon, Kitware, Inc.

Progress to date

- The ECP project has been diligently building the features of the VTK-m visualization library to include numerous visualization features including surfacing algorithms like external faces, normal generation, and contouring, multiblock and ghost cell management, geometry transformations, compression, particle advection, and a self-contained rendering library.

- The team added support to VTK-m for multiple threading libraries, including OpenMP, to better match the exascale application codes with which it integrates. The team has also tuned the performance of VTK-m on the Summit supercomputer by introducing custom kernel scheduling parameters for the hardware on that machine, which doubled (or more) the performance for many important algorithms.

- A new functionality for identifying connected components in image and mesh data was recently added. This feature has a wide application area, including image processing, computer vision, and machine learning.

- The Contour filter in VTK-m was extended to handle all 3D cell types. This enables VTK-m to create isosurface contours for unstructured grids of zoo cells in addition to meshes uniformly made of hexahedra (such as structured grids).

- In collaboration with the ECP/ALPINE and the ASC/ATDM teams, pattern recognition for image data based on Moments was added. Based on a serial algorithm, the VTK-m implementation has been extended to take advantage of all the hardware accelerators supported by VTK-m.

VeloC/SZ

Long-running large-scale simulations and high-resolution, high-frequency instrument detectors are generating extremely large volumes of data at a high rate. While reliable scientific computing is routinely achieved at small scale, it becomes remarkably difficult at exascale due to both an increased number of disruptions as the machines become larger and more complex from, for example, component failures and the big data challenge. The VeloC/SZ project addresses these challenges by focusing on ensuring high reliability for long-running exascale simulations and reducing the data while keeping important scientific outcomes intact.

Big data challenges need to be addressed at exascale for applications to achieve their performance and science goals. The VeloC/SZ project addresses two of these big data challenges. First, the ability to run scientific simulations until completion despite disruptions along the way is critical. VeloC provides a highly reliable environment for exascale applications at a minimal cost, enabling them to fully benefit from the extreme data volume and velocity they produce with a low overhead. Second, data reduction is necessary to reduce the size of the data output to the storage system due to bandwidth and storage space limitations. SZ addresses this challenge by enabling application scientists to reduce their scientific data while keeping scientific outcomes intact.

Most large-scale scientific applications use execution state recording techniques to make sure the execution finishes, despite disruptions. If a disruption occurs, the execution state can be restored and the application can be restarted from this state. This technique is known as checkpoint/restart. At exascale, this technique is difficult to implement at low cost for the applications due to an extremely large volume/velocity of data, complex disruption modes, and limited bandwidth to the storage system. Moreover, the diversity and complexity of the storage hierarchy in exascale systems make it very difficult for application developers to implement checkpoint/restart at low cost. VeloC leverages application developer knowledge about state preservation to provide a solution optimizing the performance of checkpoint/restart while masking the complexity and diversity of the storage hierarchies. An existing application can be adapted for VeloC in minimal time. Once adapted, the application can run in a highly reliable way on pre-exascale and exascale machines.

As data sizes increase with exascale systems and updated scientific instruments, lossy compression of scientific data becomes a necessity. Lossy compression reduces the data by removing non-useful information. Lossy compression for scientific data needs to satisfy three main requirements: it should remove only information that does not impact scientific discovery; compression and decompression need to be very fast to avoid raising a performance issue; and it needs to be effective at providing data reduction much higher than lossless compression. The SZ software provides lossy compression for scientific datasets satisfying these three requirements. To keep information relevant for scientific discovery, SZ users set constraints in terms of compression quality. To control the information loss for each data point, SZ provides point-wise error bound controls that the user supplies. To reach extremely high performance, the SZ software has a parallel implementation that benefits from GPU acceleration. The advanced compression pipelines used in SZ provide very high compression ratios compared with lossless compression, enabling SZ to overcome the big data challenges at the exascale.

PI: Franck Cappello, Argonne National Laboratory

Collaborators: Argonne National Laboratory

Progress to date

- The VeloC/SZ team released version 1.0 of the VeloC software. The team closely collaborated with several exascale application teams to refine the VeloC API and make sure it addresses their needs. The client library and backend were designed and implemented. The erasure-coding module and the data transfer module were integrated with the backend into a flexible engine that allows VeloC the capability of running in synchronous mode directly in the application processes or in asynchronous mode in a separate process. Results show that the impact of checkpointing (measured as increase in runtime vs. the case when no checkpointing is used) was reduced by up to 10× when using VeloC.

- The team drastically improved the performance of SZ using innovative algorithms and node-level parallelization and GPU accelerators to reduce compression and decompression time. SZ can be integrated directly in the application, or it can be used transparently through the ADIOS, HDF5, and PnetCDF I/O libraries. Compression results are outstanding in terms of performance and compression ratios, and SZ is currently being used by six ECP applications. **Typically, conventional compression will reach compression ratios between 1 and 2 on scientific data sets. ECP application users of SZ typically reached compression factors of 10.**

ExaIO

In pursuit of more accurate modeling of real-world systems, scientific applications at exascale will generate and analyze massive amounts of data. A critical requirement of these applications to complete their science mission is the capability to access and manage these data efficiently on exascale systems. Parallel I/O, the key technology behind moving data between compute nodes and storage, faces monumental challenges from new application workflows as well as the memory, interconnect, and storage architectures considered in the designs of exascale systems. The ExaIO project is delivering the HDF5 library and the UnifyFS tool to efficiently address these storage challenges.

Parallel I/O libraries of the future must be able to handle file sizes of many terabytes and I/O performance much greater than currently achievable to satisfy the storage requirement of exascale applications and enable them to achieve their science goals. As the storage hierarchy expands to include node-local persistent memory and solid-state storage as well as traditional disk and tape-based storage, data movement among these layers must become much more efficient and capable. The ExaIO project is addressing these parallel I/O performance and data management challenges by enhancing the HDF5 library and developing UnifyFS for using exascale storage devices.

The Hierarchical Data Format version 5 (HDF5) is the most popular high-level I/O library for scientific applications to write and read data files at supercomputing facilities and has been used by numerous applications. The ExaIO team is developing various HDF5 features to address efficiency and other challenges posed by data management and parallel I/O on exascale architectures. The ExaIO team is productizing HDF5 features and techniques that have been previously prototyped, exploring optimizations on upcoming architectures, and maintaining and optimizing existing HDF5 features tailored for the exascale applications. They are also adding new features including transparent data caching in the multi-level storage hierarchy, topology-aware I/O-related data movement, full single-writer and multi-reader for workflows, and asynchronous I/O.

Scientific applications need to periodically checkpoint the progress that has been made by saving the current state of the simulation so that the simulation can be restarted at a later time. This checkpoint/restart workflow has been reported to cause 75–80% of the I/O traffic on some high-performance computing systems. UnifyFS is a user-level file system highly specialized for shared file access on high-performance systems with distributed node-local storage that the ExaIO team is developing to specifically target checkpoint/restart workloads. UnifyFS transparently intercepts I/O calls, allowing integration of UnifyFS cleanly with other software including I/O and checkpoint/restart libraries. Thus, UnifyFS addresses a major usability factor of the pre-exascale and exascale systems.

Progress to date

- The ExaIO team has improved the HDF5 library in terms of performance and productivity. The team developed the Virtual Object Layer (VOL) feature to open up the HDF5 API and developed several optimizations to the improve performance of HDF5, including a topology-aware interface for the implementation of scalable algorithms and optimizations; the ability to stage the data in a temporary fast storage location, such as burst buffe, and move the data to the desired final destination asynchronously; and a capability that enables a single writing process to update an HDF5 file while multiple reading processes access the file in a concurrent, lock-free manner.

- The team completed a full system design for the UnifyFS tool and released version 2.0, which includes near complete removal of MPI dependence by integration of DataLib software for communication; support for Spack to improve the build experience and hide the complexity of UnifyFS' dependencies; and support for the Summit pre-exascale platform.

PI: Suren Byna, Lawrence Berkeley National Laboratory

Collaborators: Lawrence Berkeley National Laboratory, Lawrence Livermore National Laboratory, Oak Ridge National Laboratory, Argonne National Laboratory, The HDF Group

ALPINE/ZFP

Computational science applications generate massive amounts of data from which scientists need to extract information and visualize the results. Performing the visualization and analysis tasks in situ, while the simulation is running, can lead to improved use of computational resources and reduce the time the scientists must wait for their results. The ALPINE/ZFP project is delivering in situ visualization and analysis infrastructure and algorithms including a data compression capability for floating-point arrays to reduce memory, communication, I/O, and offline storage costs.

Many high-performance simulation codes write data to disk to visualize and analyze it after the simulation is completed. Given the exascale I/O bandwidth constraints, this process will need to be performed in situ to fully utilize the exascale resources. In situ data analysis and visualization selects, analyzes, reduces, and generates extracts from scientific simulation while the simulation is running to overcome bandwidth and storage bottlenecks associated with writing the full simulation results to the file system. The ALPINE/ZFP project produces in situ visualization and analysis infrastructure that will be used by the exascale applications along with a lossy compression capability for floating point arrays.

The ALPINE development effort focuses on delivering exascale visualization and analysis algorithms that will be critical for exascale applications; developing an exascale-capable infrastructure for in situ algorithms and deploying it into existing applications, libraries, and tools; and engaging with exascale application teams to integrate ALPINE with their software. This capability will leverage existing, successful software, ParaView and VisIt, including their in situ libraries Catalyst and Libsim, by integrating and augmenting them with ALPINE capabilities to address the challenges of exascale.

Overcoming the performance cost of data movement is also critical. With deepening memory hierarchies and dwindling per-core memory bandwidth due to increasing parallelism, even on-node data motion makes for a significant performance bottleneck and primary source of power consumption. The ZFP software is a floating-point array primitive that mitigates this problem using very high-speed, lossy (but optionally error-bounded) compression to significantly reduce data volumes and I/O times. The ZFP development effort focuses on extending ZFP to make it more readily usable in an exascale computing setting by (1) parallelizing it on both CPU and GPU, while ensuring thread safety; (2) providing bindings for multiple programming languages; (3) adding new functionality; (4) hardening the software and adopting best practices for software development; and (5) integrating ZFP with a variety of exascale applications, I/O libraries, and software tools.

Progress to date

- The ALPINE/ZFP team completed a layer on top of VTK-m for ALPINE algorithms, fully integrated the ALPINE infrastructure into ParaView and VisIt to support ALPINE algorithms to run both on CPUs and GPU accelerators, added ParaView visualization support, and made a production release of the code.
- The team demonstrated parallel implementations of core algorithms and automatic data selection methods in ALPINE, including feature-centric analysis, topolocation analysis, adaptive sampling, and Lagrangian analysis.
- The team developed OpenMP and CUDA parallel compression and decompression in ZFP that support up to 150 GB/s throughput, thus accelerating data transfer between CPU and GPU, extended compressed-array classes to be thread safe, and extended ZFP to support lossless compression and data preconditioning to improve compression of unstructured data.

PI: Jim Ahrens, Los Alamos National Laboratory

Collaborators: Los Alamos National Laboratory, Lawrence Livermore National Laboratory, University of Oregon, Kitware, Inc.

SOFTWARE PORTFOLIO

Software Ecosystem and Delivery

- E4S and SDK Efforts
- Software Packaging Technologies

E4S and SDK Efforts

Objective: Increase the interoperability, availability, quality, and sustainability of the software technologies being developed in the exascale computing project

Principal Investigator: Sameer Shende, University of Oregon

Software Packaging Technologies

Objective: Spack development effort to support software deployment at the DOE HPC facilities and supercontainers development for container-based deployment of applications and software technologies on exascale platforms

Principal Investigator: Todd Gamblin, Lawrence Livermore National Laboratory

E4S and SDK Efforts

The large number of software technologies being delivered to the application developers poses challenges, especially if the application needs to use more than one technology at the same time. The Software Development Kit (SDK) efforts identify meaningful aggregations of products within the programming models and runtimes, development tools, and data and visualization technical areas, with the goal of increasing the interoperability, availability, quality, and sustainability of the software technologies being developed in the ECP while improving developer productivity for both the software and application development teams. The resulting SDKs are packaged and delivered through the Extreme-Scale Scientific Software Stack (E4S) (https://e4s.io).

The forthcoming exascale systems require a sustainable, high-quality software ecosystem, and the ECP is chartered with delivering such an ecosystem that will continuously be improved by a robust research and development effort, deployed on advanced computing platforms, and broadly adopted by application teams and software developers to accelerate their science. The E4S and SDK efforts support a set of activities focused on establishing community policies aimed at increasing the interoperability between and sustainability of software technologies developed by the ECP and coordinating the delivery of those products through the E4S.

The Programming Models and Runtimes SDK effort identifies meaningful aggregations of products in this technical area. It provides the software infrastructure necessary to enable and accelerate the development of exascale applications that perform well and are correct and robust while reducing the cost of both initial development and ongoing porting and maintenance.

The Development Tools SDK is a collection of independent projects specifically targeted to address performance analysis at scale. The team actively works to leverage techniques for common and identified problem patterns and create new techniques for software quality assurance related to performance analysis tools while also supporting advanced techniques such as autotuning and compiler integration for upcoming heterogeneous architectures.

The Data and Visualization SDK focuses on the delivery of efficient data management and storage libraries, services such as checkpoint/restart, monitoring, code coupling and compression, and an efficient in situ visualization and analysis pipeline. The goal is to improve deployment and usage of I/O and analysis capabilities.

The Software Ecosystem SDK effort manages the release and testing of the E4S and ensures that the software technologies within E4S can be either built from source via the Spack package manager or used via pre-built container images. Application developers can build only the subset of the software technologies needed for their specific application. This effort also fosters collaboration between software technologies and interacts heavily with the Hardware and Integration (HI) focus area to facilitate software product installation at the Facilities.

Progress to date

- Version 0.2 of E4S was released, which contains 37 full products from across the software technologies and can be either built from source via the Spack package manager or used via pre-built container images.
- The Dyninst package from the Development Tools SDK was used as a pilot project for the continuous integration workflow using GitLab proposed by the HI focus area. Integration with the Dyninst GitHub source repository was successful, which was a key step in ensuring interoperability with the most popular source control platform.
- The Data and Visualization SDK addressed numerous interoperability issues among major I/O, data, and visualization products.
- The Software Ecosystem SDK team is building lines of communication and working relationships with other SDK efforts and HI staff to jointly define approaches for software deployment and testing and are effectively communicating the definition and purpose of as well as approaches used for the SDKs and E4S.

PI: Sameer Shende, University of Oregon; Bart Miller, University of Wisconsin – Madison; Chuck Atkins, Kitware, Inc.; Jim Willenbring, Sandia National Laboratories

Collaborators: Sandia National Laboratories, University of Oregon, University of Wisconsin – Madison, Kitware,Inc., The HDF Group

Software Packaging Technologies

Exascale application teams will need access to the production-ready software technologies being developed as soon as the exascale systems are delivered to achieve their performance and science goals. The Software Packaging Technologies project is developing foundational infrastructure such as Spack, a flexible package manager popular in most supercomputing environments, and containers, a standard unit of software packaging that enables applications to run quickly and reliably from one computing environment to another, to address these delivery challenges.

Deploying and maintaining an expansive software stack for facilities, developers, and end users across multiple advanced hardware platforms is critical to the success of the ECP. Building and integrating software for supercomputers, however, is notoriously difficult, and an integration effort for high-performance software at this scale is unprecedented. The software deployment landscape is changing as containers and supercomputing-capable software package managers like Spack emerge. The Software Packaging Technologies project will ensure that the Spack and container-based packaging technologies can meet the demands of the exascale ecosystem.

Spack holds the promise to automate the builds of the high-performance software technologies being developed in the exascale computing program, from facility installations to containers, and to allow it to be distributed in new ways, including as binary packages. New capabilities being developed by the team for Spack will enable completely automated deployments of software technologies at exascale Facilities.

Containers will enable entire application deployments to be packaged into reproducible images, and they hold the potential to accelerate development and apply continuous integration workflows. However, there are unique challenges to using containers at extreme scale: portability and performance fundamentally oppose each other at the binary level. The team will develop new techniques and best practices that enable containers to be used without performance loss on advanced architectures and will provide training and outreach to accelerate container adoption.

Progress to date

- Packaging Technologies is a new project that supports Spack and container technologies.
- Spack currently supports of 3,239 products that can be built from source. The team is enhancing Spack to provide turnkey deployment of these product on exascale computing resources with the goal of supporting all the software technologies developed by the ECP.
- The team is also working on performance and interoperability of container runtimes and recently demonstrated an improvement in performance on one of the ExaWind simulation codes when running within a container versus running the same code directly on the machine.

PI: Todd Gamblin, Lawrence Livermore National Laboratory

Collaborators: Lawrence Livermore National Laboratory, Sandia National Laboratories, Lawrence Berkeley National Laboratory, University of Oregon

SOFTWARE PORTFOLIO

NNSA Software Technology

- LANL NNSA Software Technology
- LLNL NNSA Software Technology
- SNL NNSA Software Technology

LANL NNSA Software Technology

Objective: LANL's NNSA/ATDM software technology efforts include Legion (PMR), Kitsune (Tools), Cinema (Data/Viz), and BEE (Ecosystem)

Principal Investigator: Mike Lang, Los Alamos National Laboratory

LLNL NNSA Software Technology

Objective: LLNL's NNSA/ATDM software technology efforts include RAJA, Umpire, and CHAI (PMR), Debugging @ Scale (Tools), MFEM (Math Libs), and Spack and Flux/Power (Ecosystem)

Principal Investigator: Becky Springmeyer, Lawrence Livermore National Laboratory

SNL NNSA Software Technology

Objective: SNL's NNSA/ATDM software technology efforts include Kokkos (PMR), Kokkos Kernels (Math Libs), and VTK-m (Data/Viz)

Principal Investigator: Jim Stewart, Sandia National Laboratories

Los Alamos National Laboratory's mission is to solve national security challenges through scientific excellence. The laboratory's strategic plan reflects US priorities spanning nuclear security, intelligence, defense, emergency response, nonproliferation, counterterrorism, energy security, emerging threats, and environmental management.

LANL NNSA SOFTWARE TECHNOLOGY

LANL NNSA Software Technology

The NNSA supports the development of open-source software technologies that are both important to the success of national security applications and externally impactful for the rest of the ECP and the broader community. These software technologies are managed as part of a larger Advanced Simulation and Computing (ASC) portfolio, which provides resources to develop and apply these technologies to issues of importance to national security. The software technologies at LANL span programming models and runtimes (Legion), development tools (Kitsune), data visualization and analysis (Cinema), and workflow orchestration (BEE).

The Legion effort is focused on delivering programming model technologies to support ASC mission applications. This work includes adding features and the integration of the Legion programming model into higher level libraries/frameworks such as the Flexible Computational Science Infrastructure. The Legion-centric efforts are focused on developing and integrating new capabilities such as dynamic control replication, which enables applications to be written with apparently sequential semantics and parallelize and scale to exascale systems. These capabilities are necessary for complex applications that often require multiple mesh representations, different discretization strategies, and support for multiple materials in a single application.

The Kitsune effort works with the open-source LLVM Compiler Infrastructure to provide tools and capabilities that address exascale needs and challenges faced by applications, libraries, and other components of the software stack. The team is focused on providing a more productive development environment that enables improved compilation times and code generation for parallelism; additional features/capabilities within the design and implementations of LLVM components for improved platform/performance portability; and improved aspects related to composition of the underlying implementation details of the programming environment.

The Cinema tool is an innovative way of capturing, storing, and exploring extreme-scale scientific data. Cinema embodies approaches to maximize insight from extreme-scale simulation results while minimizing data footprint. The team is creating capabilities that allow scientists more options in analyzing and exploring the results of large simulations by providing a workflow that detects features in situ; captures data artifacts from those features in Cinema databases; promotes post hoc analysis of the data; and provides data viewers that enable interactive, structured exploration of the resulting artifacts.

BEE (Build and Execution Environment) is a toolkit that provides users with the ability to execute application workflows across a diverse set of hardware and runtime environments. Using BEE's tools, users can build and launch applications on high-performance clusters and public and private clouds. The team is providing technology that eases the deployment of new application and software technology via containerization; has a flexible runtime that enables containers to run across a wide variety of high-performance platforms; and supports for deploying containers that support producer-consumer workflows and job coupling.

PI: Mike Lang, Los Alamos National Laboratory

Collaborators: Los Alamos National Laboratory

Progress to date

- The Legion team developed an initial implementation of control replication, which allows the programmer to write tasks with sequential semantics that can be transparently replicated many times, as directed by the Legion mapper interface, and run in a scalable manner across many nodes.
- The Kitsune team focused on supporting an infrastructure that maps multiple language constructs from Kokkos, FleCSI, and OpenMP into a common intermediate representation that explicitly captures the parallel operations for analysis and optimization. This parallel representation is then targeted to different runtime systems.
- The Cinema team demonstrated their capabilities using the Nyx exascale application code by capturing isosurfaces, saving the Cinema database with the isosurfaces data to disk, analyzing that database to determine intersection points in the complexity of those isosurfaces, and presenting the results in a newsfeed viewer linked to other views of the data.
- The BEE team released the BEE-Charliecloud, BEE-OpenStack, BEEFlow, and BEESwarm framework components and demonstrated how the tools can support a multi-physics application in a production environment.

For more than 60 years, the Lawrence Livermore National Laboratory (LLNL) has applied science and technology to make the world a safer place. LLNL will be home to El Capitan, one of the US Department of Energy's three planned exascale supercomputers.

LLNL NNSA SOFTWARE TECHNOLOGY

LLNL NNSA Software Technology

The NNSA supports the development of open-source software technologies that are both important to the success of national security applications and externally impactful for the rest of the ECP and the broader community. These software technologies are managed as part of a larger Advanced Simulation and Computing (ASC) portfolio, which provides resources to develop and apply these technologies to issues of importance to national security. The software technologies at LLNL span programming models and runtimes (RAJA/Umpire/CHAI), development tools (Debugging @ Scale), mathematical libraries (MFEM), productivity technologies (DevRAMP), and workflow scheduling (Flux/Power).

The RAJA/Umpire/CHAI team is providing software libraries that enable application and library developers to meet advanced architecture portability challenges. The project goals are to enable writing performance portable computational kernels and coordinate complex heterogeneous memory resources among components in a large integrated application.
The software products provided by this project are three complementary and interoperable libraries: RAJA provides software abstractions that enable C++ developers to write performance portable numerical kernels; Umpire is a portable memory resource management library that provides a unified high-level Application Programming Interface (API) in C++, C, and Fortran for resource discovery, memory provisioning, allocation, transformation, and introspection; and CHAI contains C++ "managed array" abstractions that enable transparent and automatic copying of application data to memory spaces at run time as needed based on RAJA execution contexts.

Debugging @ Scale provides an advanced debugging, code-correctness, and testing toolset to facilitate reproducing, diagnosing, and fixing bugs within HPC applications. The current capabilities include STAT, a highly scalable lightweight debugging tool; Archer, a low-overhead OpenMP data race detector; ReMPI/NINJA, a scalable record-and-replay and smart noise injector for message passing interface (MPI); and FLiT/FPUChecker, a tool suite for checking floating-point correctness.

The MFEM library is focused on providing high-performance mathematical algorithms and finite element discretizations to next-generation, high-order applications. This effort includes the development of physics enhancements in the finite element algorithms in MFEM and the MFEM-based BLAST Arbitrary Lagrangian-Eulerian code to support ASC mission applications and the development of unique unstructured adaptive mesh refinement algorithms that focus on generality, parallel scalability, and ease of integration in unstructured mesh applications.

DevRAMP is creating tools and services that multiply the productivity of developers through automation. The capabilities include Spack, a package manager for high-performance systems that automates the process of downloading, building, and installing different versions of software packages and their dependencies, and Sonar, a software stack for performance monitoring and analysis that enables developers to understand how high-performance computers and applications interact.

Flux /Power is a next-generation resource management and scheduling software framework. The team is providing a portable, user-level scheduling solution for complex exascale workflows and a system resource manager and scheduler for exascale systems.

PI: Becky Springmeyer, Lawrence Livermore National Laboratory

Collaborators: Lawrence Livermore National Laboratory

Progress to date

- The RAJA/Umpire/CHAI team developed support for multidimensional kernel dispatch; atomic operations on GPU accelerators; memory allocation on CPU, GPU, unified, and "pinned" memory resource and copying data between resources; high-performance memory pools; and an Umpire-based backend for CHAI that adds additional flexibility and capability.

- The Debugging @ Scale team completed the port of STAT, Archer, ReMPI/NINJA, and FLiT/FPUChecker to Sierra, deployed these tools on Sierra, and assisted those using these tools for debugging and testing.

- The MFEM team released MFEM version 3.4 with many new features including a significantly improved nonconforming unstructured adaptive mesh refinement capability, block nonlinear operators; general high-order-to-low-order refined field transfer; and specialized time integrators.

- The DevRAMP team implemented the ability for Spack to output build reports to CDash, developed syntax and a workflow for reproducible multipackage deployments, and provided a cloud-based build farm.

- The Flux/Power team enabled two major scientific workflows to complete their calculations on the Sierra pre-exascale system and released a version of their software containing all the functionalities used by these workflows.

Keeping the US nuclear stockpile safe, secure, and effective is a major part of Sandia's work as a multidisciplinary national security engineering laboratory. The laboratory's highly specialized research staff is at the forefront of innovation, collaborating with universities and companies and performing multidisciplinary science and engineering research programs with significant impact on US security.

SNL NNSA SOFTWARE TECHNOLOGY

SNL NNSA Software Technology

The NNSA supports the development of open-source software technologies that are both important to the success of national security applications and externally impactful to the rest of the Exascale Computing Project and the broader community. These software technologies are managed as part of a larger Advanced Simulation and Computing (ASC) program portfolio, which provides resources to develop these technologies for national security applications. The software technologies at Sandia National Laboratories (SNL) span programming models and runtimes (Kokkos), mathematical libraries (Kokkos Kernels), data analysis and visualization (VTK-m), and system software (OS&ONR).

The Kokkos programming model and C++ library enable performance portable on-compute-node parallelism for exascale C++ applications. The Kokkos library implementation consists of a portable application programmer interface and architecture specific back-ends. These back-ends are developed and optimized as new capabilities are added to Kokkos, back-end programming mechanisms evolve, and architectures change.

Kokkos Kernels implements on-node shared memory computational kernels for linear algebra and graph operations, using the Kokkos shared-memory parallel programming model. The algorithms and the implementations of the performance-critical kernels in Kokkos Kernels are chosen carefully to match the features of the architectures, allowing exascale applications to use high-performance kernels and transfer the burden to Kokkos Kernels developers to maintain them in future architectures.

VTK-m is a toolkit of scientific visualization algorithms for emerging processor architectures that supports fine-grained concurrency within data analysis and visualization algorithms. This fine-grained concurrency is required to achieve performance on exascale architectures. The team is building up the VTK-m codebase with the necessary visualization algorithm implementations that run across the varied hardware platforms to be leveraged at the exascale.

The OS and On-Node Runtime (OS&ONR) project focuses on the design, implementation, and evaluation of operating system and runtime system interfaces, mechanisms, and policies supporting the efficient execution of application codes on next-generation platforms. Priorities in this area include the development of lightweight tasking techniques that integrate network communication, interfaces between the runtime and operating system for management of critical resources, portable interfaces for managing power and energy, and resource isolation strategies at the operating system level that maintain scalability and performance while providing a full-featured set of system services.

PI: Jim Stewart, Sandia National Laboratories

Collaborators: Sandia National Laboratories

Progress to date

- The Kokkos team implemented new features based on customer needs, improving the applicability of Kokkos to a wide range of applications. These features include abstractions to seamlessly switch between data replication and atomic operation for scatter-add algorithms, tiled layouts, and multidimensional loop abstractions.
- The Kokkos Kernels team delivered performance portable kernels to ASC mission critical applications, including a symmetric Gauss-Seidel preconditioner and coloring algorithms.
- The VTK-m team prototyped functional tensor approximation/compression methods using subsets/slices of data. These methods include interpolation onto a structured mesh followed by JPEG-like compression, Tucker compression, canonical low-rank functional approximation, and functional tensor-train.
- The OS&ONR team enabled the first demonstration of a virtual cluster on a Cray system, provided support for coordination of on-node resources between multiple OS/R environments to evaluate and improve performance isolation capabilities, and performed a scaling study comparing a containerized version of the Nalu exascale application code with a native version, which demonstrated that the container can actually reduce runtime while consuming more memory.

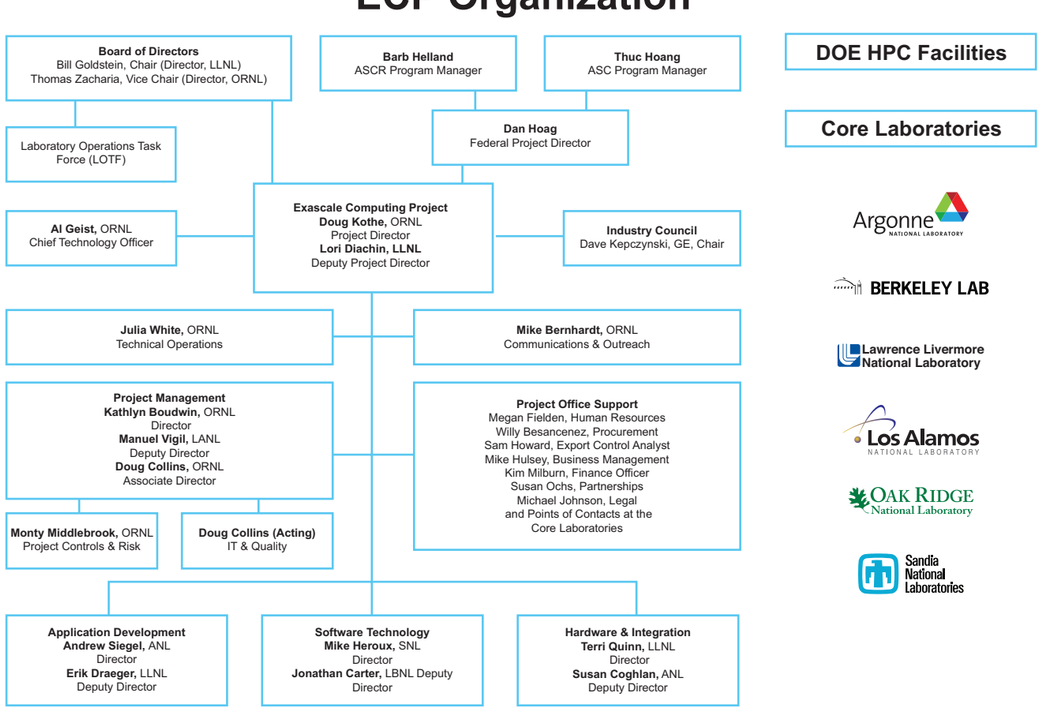
BRINGING IT ALL TOGETHER

BRINGING IT ALL TOGETHER

The ECP is led by a team of senior computer and computational scientists, engineers, and project specialists from six US Department of Energy (DOE) national laboratories that have historically maintained core competency and leadership in high performance computing (HPC), mathematics, and computer and computational science. Working together, the ECP leadership team has established an extensive network to deliver a capable exascale computing ecosystem for the nation, partnering with experts at other DOE national laboratories and HPC facilities, US HPC companies, and leading academic institutions.

The ECP Board of Directors (BOD) consists of the laboratory directors from the six core partner DOE laboratories, who have signed a memorandum of agreement. The board has an active advisory, oversight, and line-management role. Within the board is an executive committee that selects a chair and vice-chair from among its membership. One of the two will be from a DOE Office of Science laboratory and the other from a National Nuclear Security Administration laboratory. The board's primary purpose is to provide strategic direction to the ECP project director and leadership team. The ECP board also appoints a Laboratory Operations Task Force (LOTF), which is composed of associate laboratory directors with line management responsibility for HPC at their respective core partner DOE laboratories. The LOTF assists the board in overseeing the operations of the ECP and supports and advises the ECP project director.

ECP Organization



Ensuring Strong, Competitive HPC Capabilities for US Industry in the Age of Exascale

The ECP Industry Council is an external advisory group that provides essential two-way communication and information exchange between ECP and the HPC industrial user community as well as the commercial HPC software community.

ECP's Industry Council is composed of senior executives from some of the nation's most prominent companies who share a collaborative interest in bringing the potential of exascale computing to a wide range of industry segments.

ECP Industry Council Member Organizations

Altair Engineering, Inc.
ANSYS, Inc.
Cascade Technologies, Inc.
Chevron Corporation
Cummins, Inc.
Animation Eli Lilly and Company
ExxonMobil Corporation
FedEx Corporation
General Electric
General Motors Company
KatRisk, LLC
Procter & Gamble
The Boeing Company
The Goodyear Tire & Rubber Company
Tri Alpha Energy, Inc.
United Technologies Corporation
Westinghouse Electric Company
Whirlpool Corporation

Maintaining the highest level of computational capability is critical to the nation's industrial competitiveness, and the ECP Industry Council is critically important to keeping the project in sync with the real world needs of the US industrial sector.

ECP by the Numbers

7 YEARS $1.8B

A 7 year, $1.8B R&D effort that launched in 2016

Six CORE DOE LABS

Six core DOE national laboratories: Argonne, Lawrence Berkeley, Lawrence Livermore, Los Alamos, Oak Ridge, and Sandia

Staff from most of the 17 DOE national laboratories take part in the project

Three FOCUS AREAS

Three technical focus area: Hardware and Integration, Software Technology, Application Development supported by a Project Management Office

80 R&D TEAMS

More than 80 top-notch R&D teams

1000 RESEARCHERS

Hundreds of consequential milestones delivered on schedule and within budget since project inception

The ECP's Enduring Legacy

The products and solutions generated by the Exascale Computing Project (ECP) will have a tremendous impact across the entire high performance computing (HPC) ecosystem, benefitting the US economy, scientific discovery, and national security for many years to come. Well into the next decade, scientists and researchers will take advantage of the ECP-driven advancements in numerous applications, software tools, and hardware innovations, accelerating a wide range of research efforts that address the toughest problems facing the DOE and the nation.

All Boats Will Rise

The emerging exascale ecosystem and exciting new capabilities made possible by the ECP's efforts will not be limited to applications running only on exascale platforms. The enduring legacy of the ECP will trickle down and benefit computing systems of all sizes (clusters to desktops), impacting R&D in scientific applications, as well as industrial and commercial high-end computing. The ECP's legacy for software will impact a new generation of HPC systems, well beyond Aurora, Frontier, and El Capitan.

While the legacy of the ECP will be most visible in applications, significant advances in software development and delivery, along with innovative hardware architecture enhancements, will be foundational to enabling those applications to meet their anticipated science capability and execute performance goals as the next generation of computational and data science tools. The software technology and hardware and integration efforts represent a behind-the-scenes herculean accomplishment that rounds out the capable exascale ecosystem. Ultimately, the ECP will transfer to the computational science and engineering community a portfolio of exascale-ready applications, a sophisticated, modern software stack, and a blueprint for best practices in co-design, continuous integration, and collaboration among government, academia, and industry enabling unprecedented HPC portability.

It is also worth noting that, with the added impetus from the ECP, the scientific and technical computing communities will experience widespread adoption of accelerators for enhancing the performance of exascale environments supported by documented best practices and extensive workforce training programs.

Our collective project overview and update documents covering the ECP's application development, software technology, and hardware and integration efforts provide an in-depth perspective on the nation's capable exascale ecosystem—the enduring legacy of the ECP.

Credits:

Project Management

Mike Bernhardt, Oak Ridge National Laboratory

Graphics and Design

Adam Malin, Oak Ridge National Laboratory

Production

Kase Clapp, Oak Ridge National Laboratory
Adam Malin, Oak Ridge National Laboratory

Content Consultant

Todd Munson, Argonne National Laboratory

Software Technology Principal Investigators:

(In order of software products listed within this document)

Pavan Balaji, Argonne National Laboratory
Pat McCormick, Los Alamos National Laboratory
George Bosilca, University of Tennessee – Knoxville
Scott Baden, Lawrence Berkeley National Laboratory
Mike Lang, Los Alamos National Laboratory
David Bernholdt, Oak Ridge National Laboratory
Christian Trott, Sandia National Laboratories
Pete Beckman, Argonne National Laboratory
Jack Dongarra, University of Tennessee – Knoxville
John Mellor-Crummey, Rice University
Jeffrey Vetter, Oak Ridge National Laboratory
Barbara Chapman, Brookhaven National Laboratory
Ulrike Yang, Lawrence Livermore National Laboratory
Barry Smith, Argonne National Laboratory
Sherry Li, Lawrence Berkeley National Laboratory
Carol Woodward, Lawrence Livermore National Laboratory
John Turner, Oak Ridge National Laboratory
Scott Klasky, Oak Ridge National Laboratory
Rob Ross, Argonne National Laboratory
Ken Moreland, Sandia National Laboratories
Franck Cappello, Argonne National Laboratory
Suren Byna, Lawrence Berkeley National Laboratory
Jim Ahrens, Los Alamos National Laboratory
Sameer Shende, University of Oregon
Todd Gamblin, Lawrence Livermore National Laboratory
Becky Springmeyer, Lawrence Livermore National Laboratory
Jim Stewart, Sandia National Laboratories

The Exascale Computing Project (ECP) is a joint effort of two US Department of Energy (DOE) organizations: the Office of Science and the National Nuclear Security Administration.

The ECP is led by a team of senior scientists, project management experts and engineers from six of the largest DOE national laboratories.

Argonne NATIONAL LABORATORY

BERKELEY LAB

OAK RIDGE National Laboratory

Los Alamos NATIONAL LABORATORY

Lawrence Livermore National Laboratory

Sandia National Laboratories

U.S. DEPARTMENT OF ENERGY | Office of Science

NNSA National Nuclear Security Administration

exascaleproject.org

Published: October 2019