ECP Update

Doug Kothe (ORNL), ECP Director
Lori Diachin (LLNL), ECP Deputy Director
Lois Curfman McInnes (ANL), ECP Math Libraries Lead

Winter ASCAC Meeting
Washington, DC
January 13-14, 2020

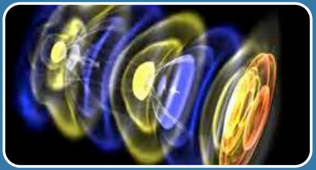
ECP Update
CD-2/3 Review

Doug Kothe (ORNL)
ECP Director

ECP recent activities (since Sep 2019 ASCAC)

- **Application Development (AD)**
  - Reviewed all subprojects (with external SMEs), leading to an updated Application Assessment Report
  - Focus of review: scrutinize progress on pre-exascale hardware, with focus on GPU utilization
- **Software Technology (ST)**
  - Reviewed all subprojects (with external SMEs), leading to an updated Capability Assessment Report (CAR)
  - Release v1.0 of ECP's Extreme Scale Scientific Software Stack: 50 full- & 6 partial-release products (e4s.io)
- **Hardware & Integration (HI)**
  - App performance engineers now in place @ ALCF, OLCF, NERSC; Aurora & Frontier deep-dive workshops
  - Coordination with ALCF & OLCF HPC Vendor Centers of Excellence for hackathons, early hardware access
- **Review for approval of Critical Decision (CD) 2 / 3 (ECP's FY20-24 Performance Baseline)**
  - Oct 15-19, 2019 Red Team Review uncovered a key corrective action, e.g., AD <-> ST scope dependency
- **Outreach activities**
  - With ECP's Industry Council, NOAA, NASA, CERN (LHC), UK (UKRI / STFC), Japan (Riken), Switzerland (CSCS), NSF, DoD (HPCMP)
- Participation in Town Halls on AI for Science and ASCAC subcommittee on ECP Transition
- Planning for our 4th Annual Meeting (Houston, TX; Feb 3-7, 2020)

Key ECP preparations for approval of Critical Decision 2 / 3



Project Management

- Set FY20-23 Performance Baseline, updated BoEs
- Refined AD & ST Performance Measurement methodology
- Continued tracking CPI / SPI and began variance reporting
- Established living database of critical AD - ST dependencies
- Evolved Project Dashboard

Application Development

- Finalized KPP-1/2 applications and their completion criteria
- Finalized KPP-1/2 metrics
- Refined planning approach for application development with longer-duration mileposts
- Established prioritized application focus for Facilities

Software Technology

- Finalized KPP-3 integration goals and metrics
- Refined planning approach to incorporate more agility
- Defined ST product dictionary to aid dependency tracking
- Established SDK -> E4S delivery process

Hardware and Integration

- For KPP-4, authorized all remaining PathForward milestones
- Revised and baselined the Facility Engagement Plans
- Negotiated exascale system and early hardware access
- Hired Facility performance engineers to focus on specific ECP applications

The road ahead to ECP completion is busy

FY19

- **IDR of Final Design**
- Establish performance baseline
- AD KPP completion criteria and ST integration goals set
- *Access to pre-exascale systems*

FY20

- **CD-2/3 Review and Approval**
- Did PathForward deliver? Are AD and ST performance and integration projections on track?
- *Access to Aurora and Frontier early hardware*

FY21

- **Status IPR**
- AD application projections firm for target system
- ST integration goals assessed
- *Access to Aurora and Frontier Test and Development Systems*

FY22

- **Status IPR**
- AD and ST readiness demonstrated
- *Access to El Capitan early hardware*
- *Access to Aurora and Frontier full system*

FY23

- **Project Completion (plan date – Q4)**
- *Access to El Capitan*
- Deliver KPP completion evidence

Project Critical Decision
Project Review
Project Design

ECP is proactively accounting for critical external dependencies

- **Facilities**
  - Scope for ECP / Facility collaboration defined for training, AD and ST performance optimization, application integration, continuous integration, and access to exascale hardware resources (early hardware, test and development [TDS], and full systems).
  - Schedule for ECP access to Aurora and Frontier agreed to.
  - Process for pre-exascale system allocations for ECP development resources in place.
  - Application integration prioritized for specific Facilities (ALCF, OLCF, NERSC).
- **HPC Vendors**
  - Regular ECP-vendor interactions (through established nondisclosure agreements [NDAs]) now occurring including information meetings, training sessions, hackathons. Helps ECP identify "gaps."
- **Open Source Software**
  - ECP relies on and benefits from evolving standards (MPI, OpenMP, C++, etc.). ECP is pushing these standards to benefit HPC by having a strong presence on standards committees.
  - LLVM compiler technology is becoming the nexus for vendor and community compiler development and evolution. ECP scope is helping to drive this development.

CD-2/3 Review Charge Questions for ECP

1. Have the recommendations from the October 2018 IPR and Final Design Review been addressed? **YES**
2. Are the proposed cost and schedule and scope baselines sufficient to meet the KPPs and complete the project? **YES**
3. Have the risks been adequately identified and have appropriate risk responses been developed for this phase of the project? Is there adequate contingency? **YES**
4. Is the management of the ECP appropriately structured and empowered to ensure the project's success? Has ECP accounted for the critical external dependencies required for ECP success? **YES**
5. Has the project met all the requirements for a CD-2/3 and is the project ready for CD-2/3 approval? **YES**

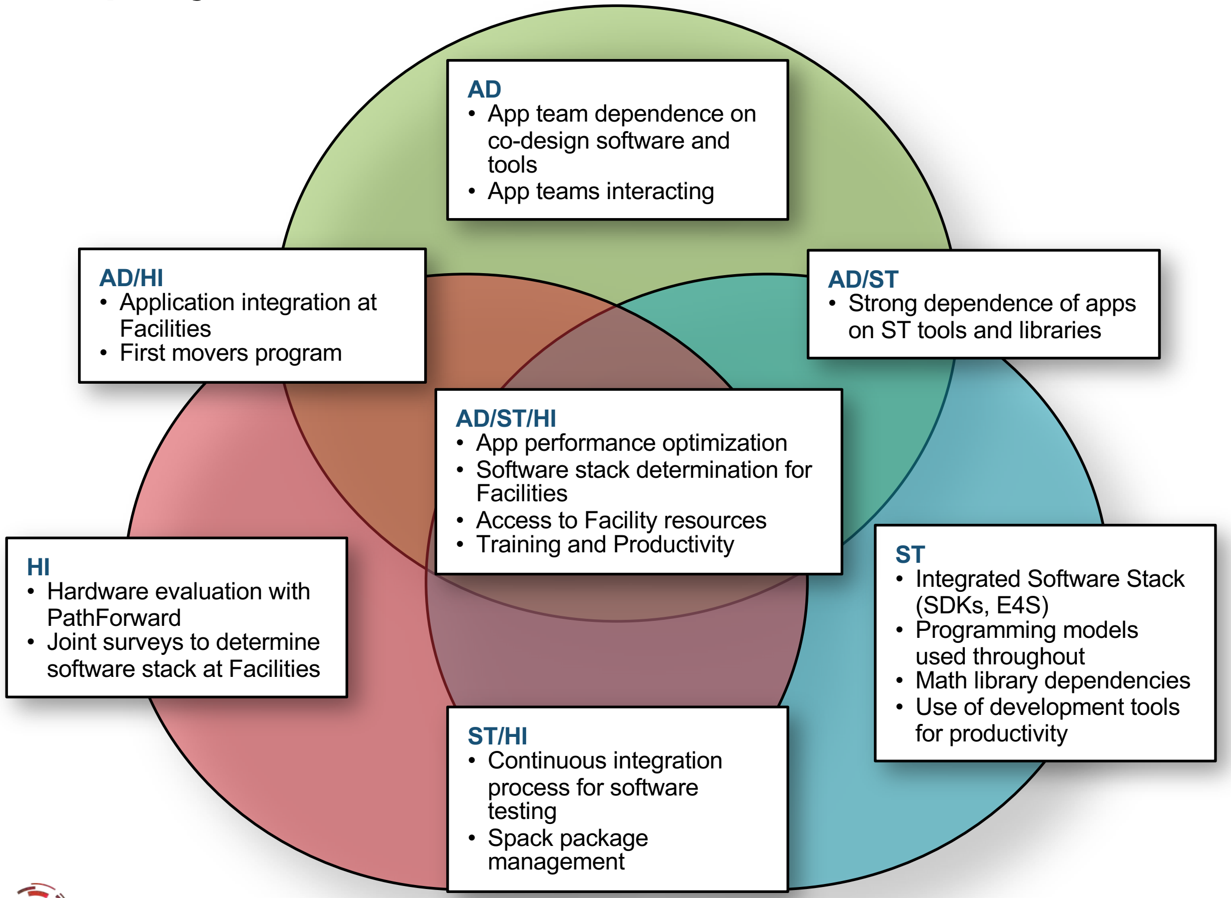
ECP CD-2/3 review recommendations and comments

- **Recommendations** (respond before project completion)
  - Capture, archive, and publish experience with the solution Dashboard, including design philosophies, successes, failures and challenges.
  - Prior to turning over applications to science sponsors, publish a *Community Outreach Document* to disseminate knowledge and experience gained and document maturity for supporting scientific discovery.
- **Key comments**
  - Application Development (AD): document milepost concept, continue strong integration with ST & HI
  - Software Technology (ST): coordinate testing resources, CI, software ecosystems (including AD apps)
  - Hardware & Integration (HI): release hardware evaluation reports; continue to work NDA issues
  - Project Office (PO): monitor critical paths; evaluate schedule impact of PCRs; monthly cost reviews
  - Project Management (PM): strengthen relationship with ALCF; address personnel retention risk; develop contingency spend plan

AD-ST Dependency Database

Lori Diachin (LLNL)
ECP Deputy Director

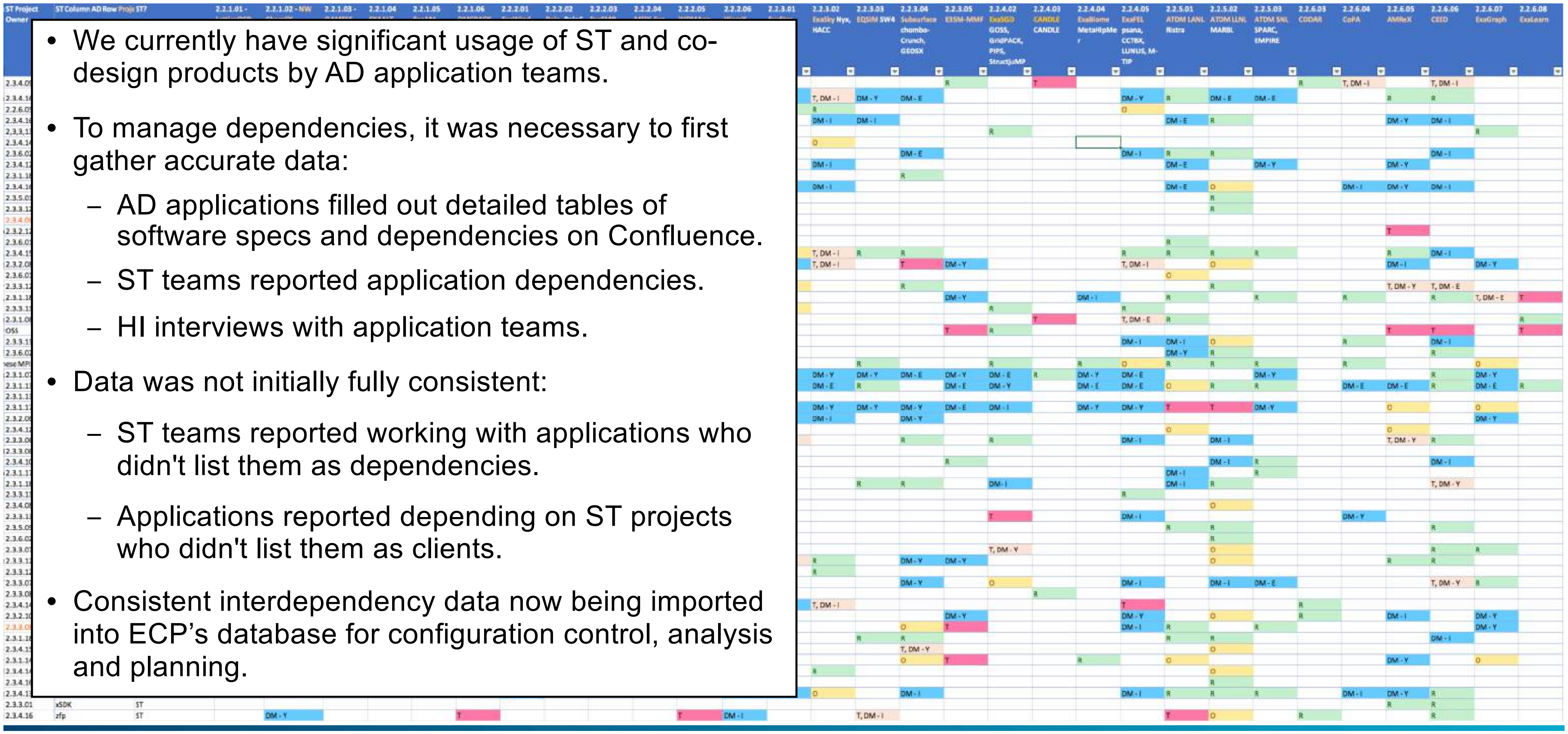
The ECP is proactively managing several dependencies both within the project and with the DOE Facilities


AD
• App team dependence on co-design software and tools
• App teams interacting
AD/HI
• Application integration at Facilities
• First movers program
AD/ST
• Strong dependence of apps on ST tools and libraries
AD/ST/HI
• App performance optimization
• Software stack determination for Facilities
• Access to Facility resources
• Training and Productivity
HI
• Hardware evaluation with PathForward
• Joint surveys to determine software stack at Facilities
ST
• Integrated Software Stack (SDKs, E4S)
• Programming models used throughout
• Math library dependencies
• Use of development tools for productivity
ST/HI
• Continuous integration process for software testing
• Spack package management

DOE Facility Dependencies

- ECP requires access to Facility resources to develop, test, and demonstrate KPPs
- ECP software stack must leverage and complement vendor and Facility software stack
- PathForward program designed to keep US industry healthy and feed into Facility procurements

Managing AD-ST complexity was initially done through extensive surveys

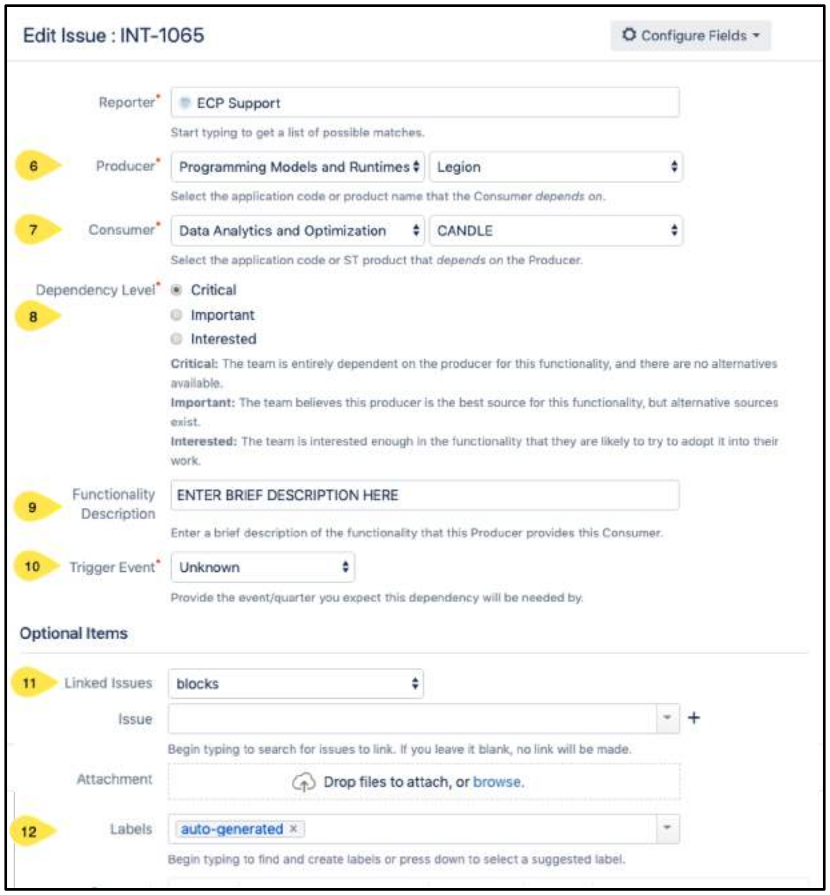


- We currently have significant usage of ST and co-design products by AD application teams.
- To manage dependencies, it was necessary to first gather accurate data:
  - AD applications filled out detailed tables of software specs and dependencies on Confluence.
  - ST teams reported application dependencies.
  - HI interviews with application teams.
- Data was not initially fully consistent:
  - ST teams reported working with applications who didn't list them as dependencies.
  - Applications reported depending on ST projects who didn't list them as clients.
- Consistent interdependency data now being imported into ECP's database for configuration control, analysis and planning.

ECP has developed and evolved a Jira database that allows more rigorous tracking of AD/ST/CD dependencies

- Identify producer and consumer from pre-defined ST and AD code lists
- Dependency Level
  - ***Critical***: entirely dependent, no alternates
  - ***Important***: best path forward, some alternates exist
  - ***Interested***: will try to adapt it for their work
- Functionality Description
- Trigger event if known

- Now working to enhance level of detail captured on dependency functionality, POCs, verification for KPP-3 scores

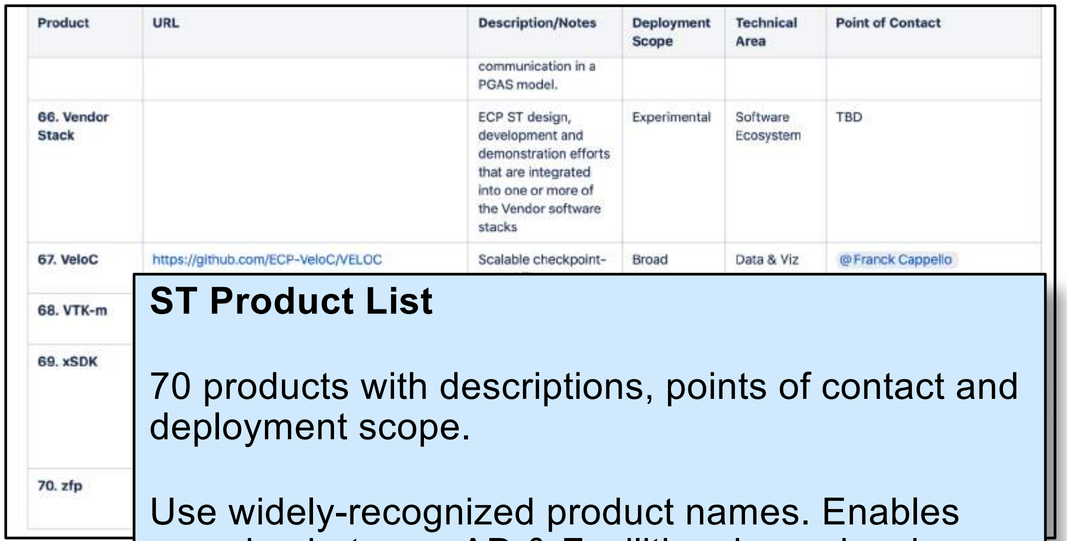

Edit Issue : INT-1065

	Reporter*	ECP Support Start typing to get a list of possible matches.
6	Producer*	Programming Models and Runtimes / Legion Select the application code or product name that the Consumer *depends* on.
7	Consumer*	Data Analytics and Optimization / CANDLE Select the application code or ST product that *depends* on the Producer.
8	Dependency Level*	[✓ Critical] Important Interested **Critical:** The team is entirely dependent on the producer for this functionality, and there are no alternatives available. **Important:** The team believes this producer is the best source for this functionality, but alternative sources exist. **Interested:** The team is interested enough in the functionality that they are likely to try to adopt it into their work.
9	Functionality Description	ENTER BRIEF DESCRIPTION HERE Enter a brief description of the functionality that this Producer provides this Consumer.
10	Trigger Event*	Unknown Provide the event/quarter you expect this dependency will be needed by.

Optional Items

11	Linked Issues	blocks
	Issue	Begin typing to search for issues to link. If you leave it blank, no link will be made.
	Attachment	Drop files to attach, or browse.
12	Labels	auto-generated × Begin typing to find and create labels or press down to select a suggested label.

Official AD and ST product lists enable rigorous dependency tracking



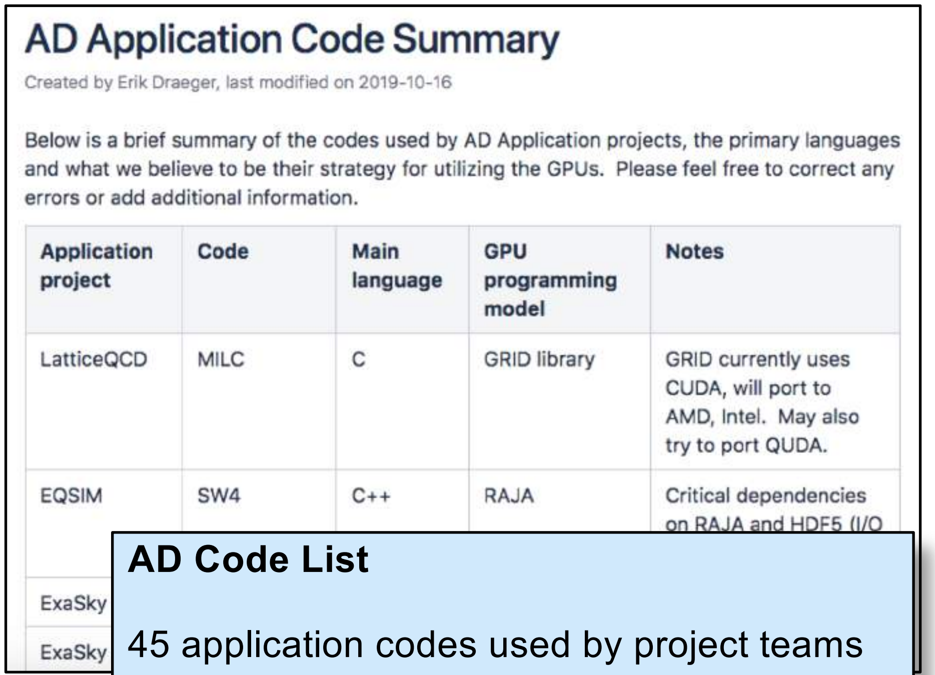
Product	URL	Description/Notes	Deployment Scope	Technical Area	Point of Contact
		communication in a PGAS model.			
66. Vendor Stack		ECP ST design, development and demonstration efforts that are integrated into one or more of the Vendor software stacks	Experimental	Software Ecosystem	TBD
67. VeloC	https://github.com/ECP-VeloC/VELOC	Scalable checkpoint-	Broad	Data & Viz	@Franck Cappello
68. VTK-m					
69. xSDK					
70. zfp					

ST Product List

70 products with descriptions, points of contact and deployment scope.

Use widely-recognized product names. Enables mapping between AD & Facilities dependencies and ST development efforts.

- MPI – MPICH, OpenMP
- C++/C/Fortran - LLVM
- Fortran – Flang
- hypre – hypre



AD Application Code Summary

Created by Erik Draeger, last modified on 2019-10-16

Below is a brief summary of the codes used by AD Application projects, the primary languages and what we believe to be their strategy for utilizing the GPUs. Please feel free to correct any errors or add additional information.

Application project	Code	Main language	GPU programming model	Notes
LatticeQCD	MILC	C	GRID library	GRID currently uses CUDA, will port to AMD, Intel. May also try to port QUDA.
EQSIM	SW4	C++	RAJA	Critical dependencies on RAJA and HDF5 (I/O
ExaSky				
ExaSky				

AD Code List

45 application codes used by project teams

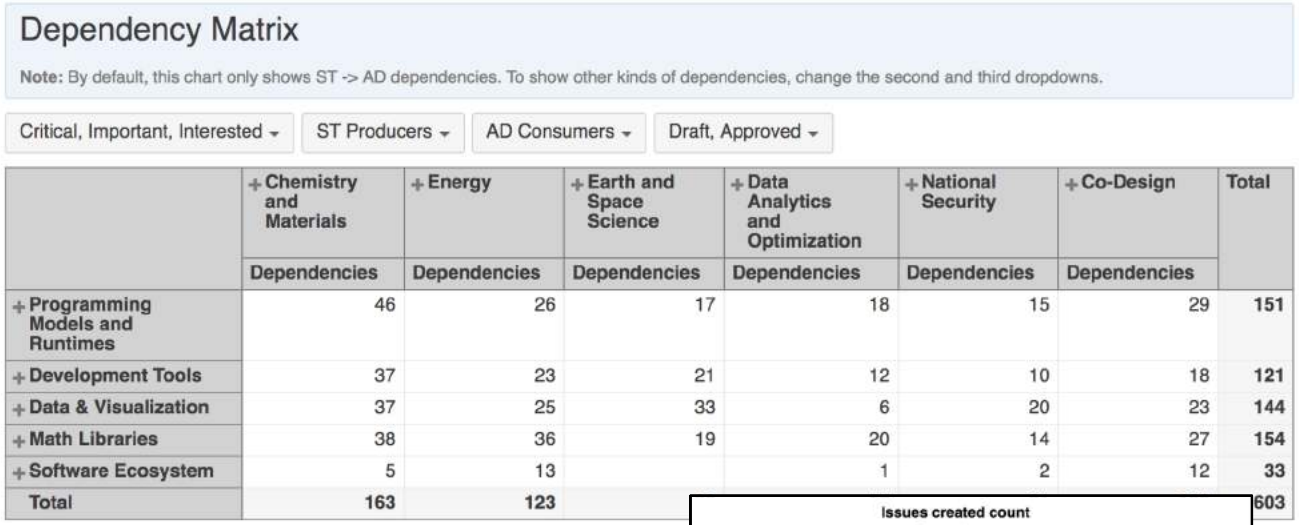
Clear statement of languages used, GPU strategy, some notes on integration

https://confluence.exascaleproject.org/display/1ST/ECP+ST+Product+Dictionary

https://confluence.exascaleproject.org/display/12AD/AD+Application+Code+Summary

ECP’s use of Jira to track these dependencies allows for significant real-time data analytics

Can show data by
- Level of dependency
- Consumer
- Producer
- Cross products thereof

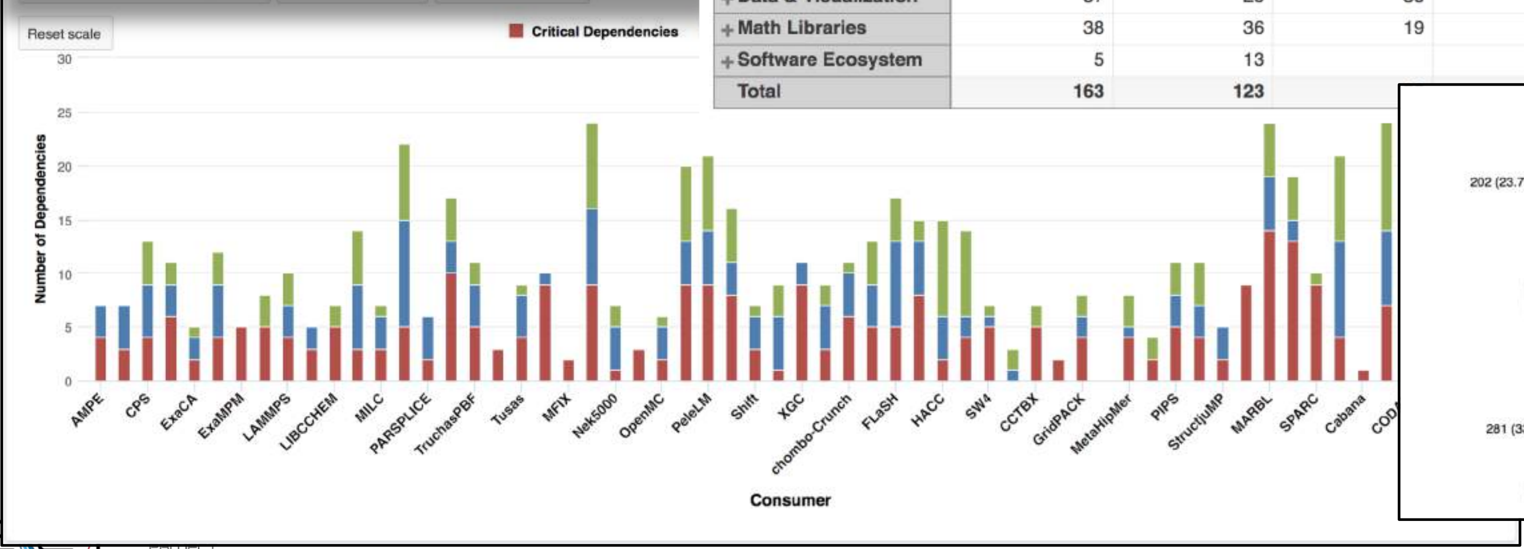


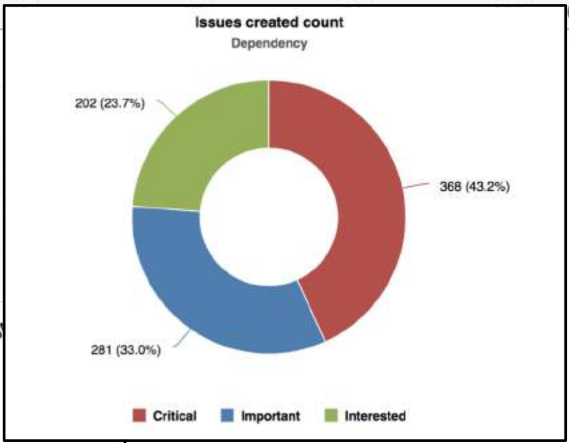
Dependency Matrix

Note: By default, this chart only shows ST -> AD dependencies. To show other kinds of dependencies, change the second and third dropdowns.

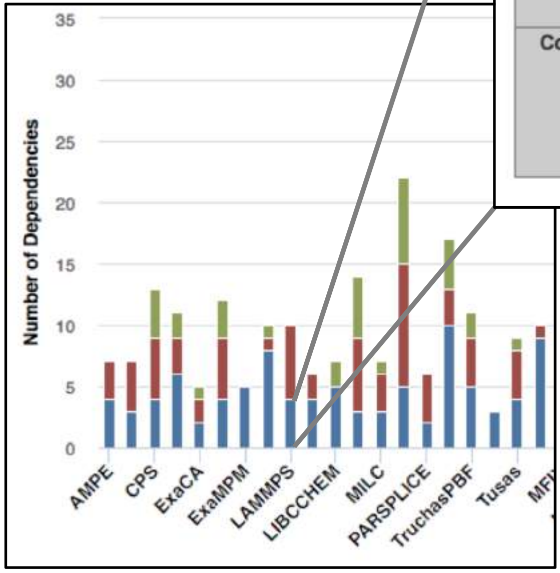
Critical, Important, Interested ▾ | ST Producers ▾ | AD Consumers ▾ | Draft, Approved ▾

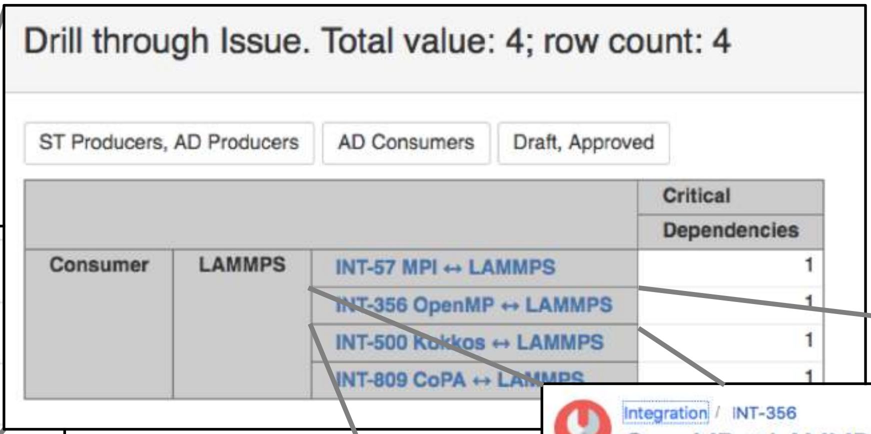
	Chemistry and Materials	Energy	Earth and Space Science	Data Analytics and Optimization	National Security	Co-Design	Total
	Dependencies	Dependencies	Dependencies	Dependencies	Dependencies	Dependencies	
Programming Models and Runtimes	46	26	17	18	15	29	**151**
Development Tools	37	23	21	12	10	18	**121**
Data & Visualization	37	25	33	6	20	23	**144**
Math Libraries	38	36	19	20	14	27	**154**
Software Ecosystem	5	13		1	2	12	**33**
Total	**163**	**123**					**603**





The Jira dashboard allows both high level and detailed drill down to help manage critical dependences

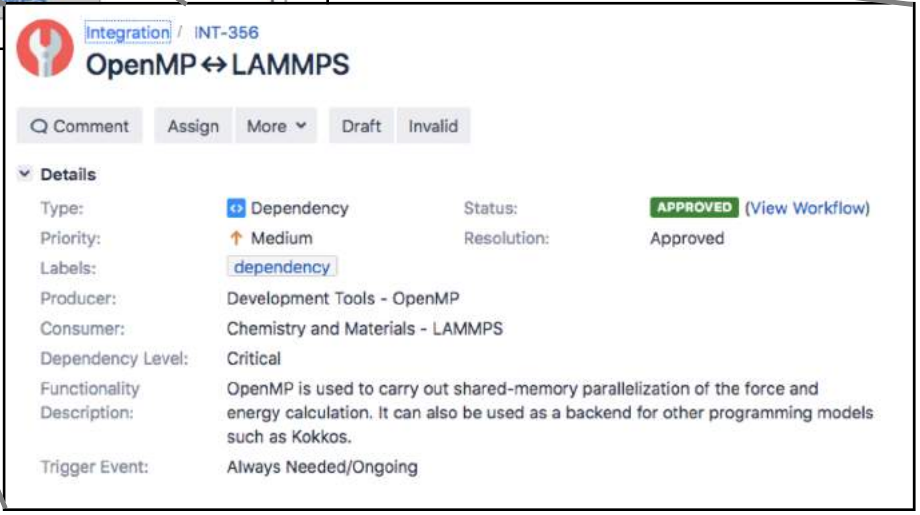




Drill through Issue. Total value: 4; row count: 4

ST Producers, AD Producers | AD Consumers | Draft, Approved

			Critical Dependencies
Consumer	LAMMPS	INT-57 MPI ↔ LAMMPS	1
		INT-356 OpenMP ↔ LAMMPS	1
		INT-500 Kokkos ↔ LAMMPS	1
		INT-809 CoPA ↔ LAMMPS	1

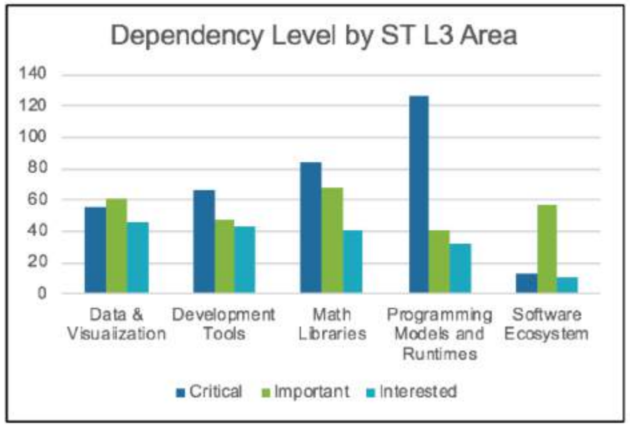


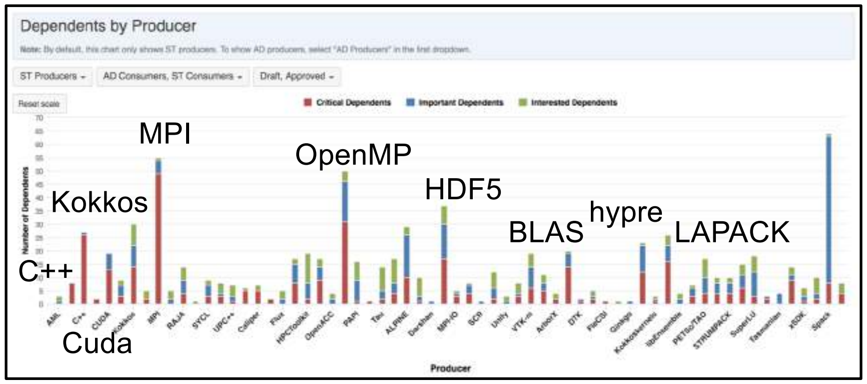
OpenMP ↔ LAMMPS

Type:	Dependency	Status:	APPROVED (View Workflow)
Priority:	Medium	Resolution:	Approved
Labels:	dependency		
Producer:	Development Tools - OpenMP		
Consumer:	Chemistry and Materials - LAMMPS		
Dependency Level:	Critical		
Functionality Description:	OpenMP is used to carry out shared-memory parallelization of the force and energy calculation. It can also be used as a backend for other programming models such as Kokkos.		
Trigger Event:	Always Needed/Ongoing		

We are using the database to proactively manage key dependency information

- Applications have clear critical dependencies on programming models and math libraries in general
- Particular tools/libraries with high dependencies:
  - Greater than 10 Critical: MPI, OpenMP, hypre, Kokkos, HDF5, C++, BLAS, LAPACK, CUDA
  - Greater than 10 Critical/Important: Above plus Fortran, LLVM, ALPINE, vtk-m, AMReX, SUNDIALS, SuperLU, Trilinos, Spack
- Data and visualization/IO libraries and software ecosystem tools are important to applications
  - Often not considered on the critical path for KPP-1/KPP-2 verification
  - May require additional KPP-3 verification runs
- Allows proactive determination of software stack needs for applications at the Facilities
- Identifies opportunities for additional integration activities
  - "Interested" dependency level
  - Applications that may be over or under leveraging ST tools





Some interesting statistics from an analysis of the database (both AD and ST consumers)

Programming Model	Crit	Imp	Int
MPI	52	4	1
OpenMP	32	14	6
OpenACC	1	3	2
CUDA	15	7	0
Kokkos	15	8	7
RAJA	4	5	5
Legion	2	0	2
UPC++	1	2	4
SYCL	3	5	2

Programming Language	Crit	Imp	Int
Fortran	11	7	2
C++	28	1	0
C	8	0	0

Highly Leveraged ST Libraries (>25)	Crit	Imp	Int
C++	28	1	0
Kokkos	15	8	7
MPI	52	4	1
OpenMP	32	14	6
ALPINE	10	16	3
HDF5	17	14	8
LAPACK	17	6	4
Spack	8	55	1

Apps that highly leverage ST (>20)	Crit	Imp	Int
NWChemEx	4	11	6
Nalu-Wind	9	7	8
PeleM	5	7	7
XGC	18	6	3
MARBL	14	5	5
AMReX	4	9	8
CEED	7	7	12
COPA	13	6	7

- MPI and OpenMP dominate the programming models space
- CUDA codes will need to migrate
- Significant uptake of performance portability tools
- Limited update of alternate programming models
- Large dependence on C++
- 1/4 of app codes depend on Fortran
- Provides insight into the tools that AD teams are relying on
- Provides insight into app teams with significant reliance on ST tools

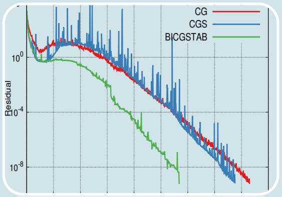
ECP Math Libraries

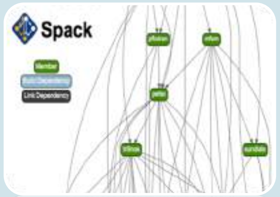
Lois Curfman McInnes (ANL)
ECP Math Libraries Lead

Math Libraries Deep Dive

- Objectives, projects
- Collaborations: ECP applications and math library teams
- What’s new from a math libraries community perspective
- ECP ST software architecture: E4S/SDKs/Products

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





Programming Models & Runtimes

- Enhance & 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 & power management

Development Tools

- Continued, multifaceted capabilities in portable, open-source LLVM compiler ecosystem to support expected ECP architectures, including support for F18
- 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

Data and Visualization

- I/O libraries: HDF5, ADIOS, PnetCDF,
- I/O via the HDF5 API
- Insightful, memory-efficient in-situ visualization and analysis – Data reduction via scientific data compression
- Checkpoint restart
- Filesystem support for emerging solid state technologies.

Software Ecosystem

- 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 ST

- Projects that have both 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

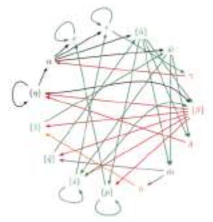
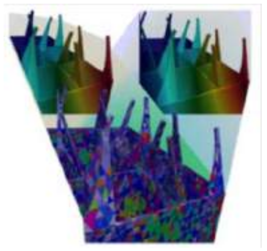
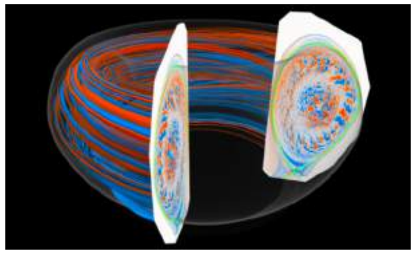
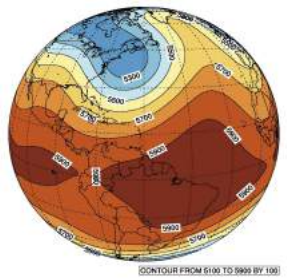
ECP Math Libraries: Context for the portfolio

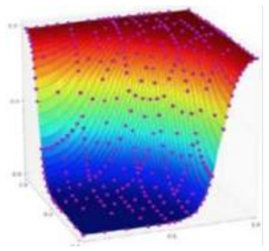
Vision	Provide high-quality, sustainable extreme-scale math libraries that are constantly improved by a robust research and development effort and support exascale needs of the ECP community	
Challenges	Need advances in algorithms and data structures to exploit emerging exascale architectures (high concurrency, limited memory bandwidth, heterogeneity); need new functionality to support predictive simulation and analysis	
Mission	Research, develop, and deliver exascale-ready math libraries to ECP applications	
Objective	Provide scalable, robust, efficient numerical algorithms, encapsulated in libraries that applications can readily use in combination to support next-generation predictive science	
Starting Point	Existing HPC math libraries, used by broad range of ECP applications for the most advanced technologies available in math and computer science R&D	
Portfolio Goals	**Advanced algorithms**	• Advanced, coupled multiphysics and multiscale algorithms (discretizations, preconditioners & Krylov solvers, nonlinear & timestepping solvers, coupling) • Toward predictive simulation & analysis (optimization, sensitivities, UQ, ensembles)
	Performance	• Performance on new node architectures • Extreme strong scalability
	Improving library sustainability & complementarity	• Math library interoperability and complementarity through the xSDK • Improving package usability, quality, sustainability • Community coordination and collaboration while retaining package autonomy

EXASCALE COMPUTING PROJECT

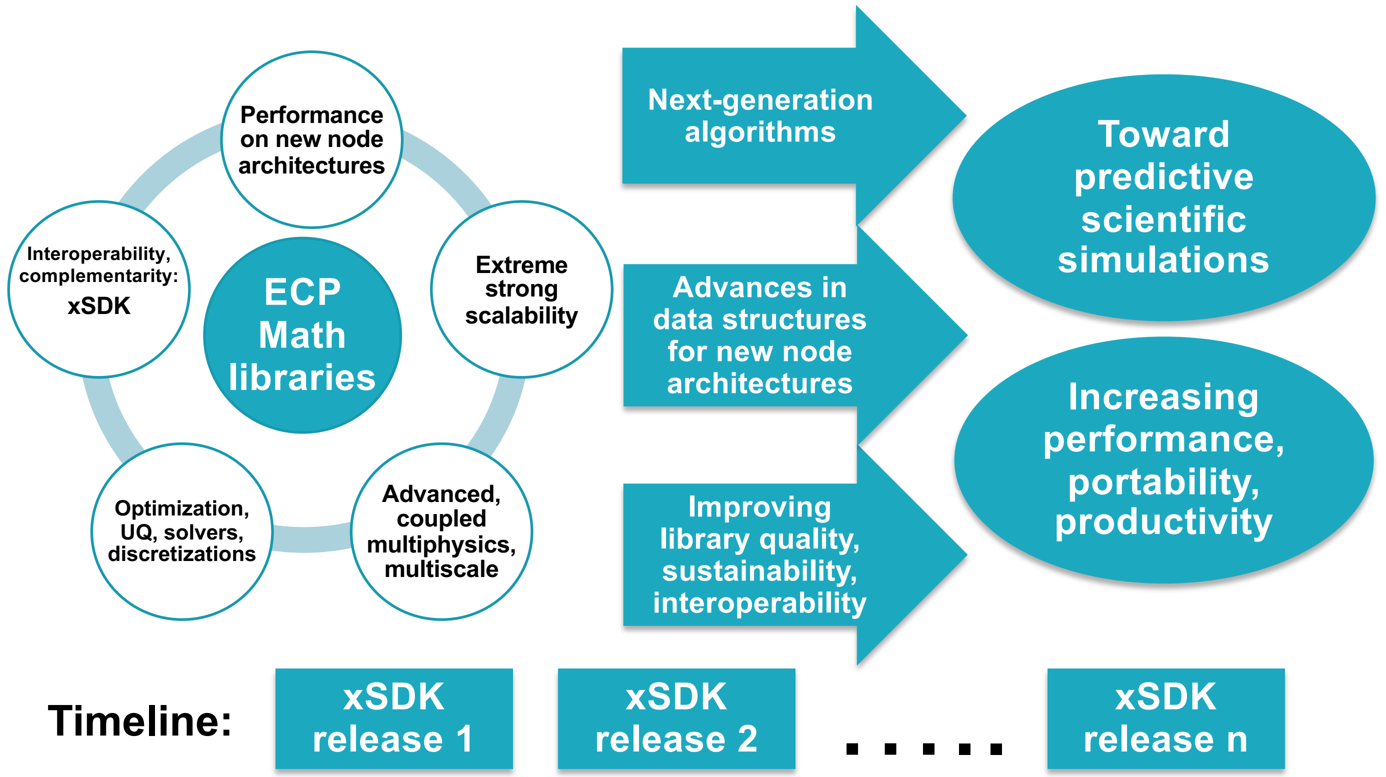
ECP Math Libraries: Projects

Project Short Name	PI Name, Inst	Short Description/Objective
xSDK	Ulrike Meier Yang, LLNL	xSDK (Extreme-scale Scientific Software Development Kit): community policy-based approach to value-added aggregation of independently developed math libraries (increasing quality, combined usability, interoperability)
PETSc / TAO	Barry Smith, ANL	PETSc (scalable linear & nonlinear solvers, integrators), TAO (numerical optimization), libEnsemble (ensemble management for exascale platforms)
STRUMPACK / SuperLU / FFTX	Xiaoye Li, LBNL	STRUMPACK & SuperLU (scalable sparse direct solvers, preconditioners), FFTX (FFT stack, including symbolic analysis and code generation)
SUNDIALS / hypre	Carol Woodward, LLNL	SUNDIALS (adaptive time integrators, nonlinear solvers), hypre (scalable linear solvers, with emphasis on algebraic multigrid)
CLOVER	Jack Dongarra, UTK	SLATE (exascale-capable dense linear algebra), FFT-ECP (scalable FFTs), PEEKS (latency-tolerant preconditioned iterative solvers, via Trilinos, Ginkgo, MAGMA-sparse), KokkosKernels (portable performance kernels for linear algebra and graph algorithms)
ALExa / ForTrilinos	John Turner, ORNL	DTK (parallel data transfer between grids, search tree capability), Tasmanian (uncertainty quantification, surrogate modeling), ForTrilinos (automatic generation of Fortran interfaces for Trilinos)



ECP Math Libraries: Continual advancements toward predictive science


Performance on new node architectures
Extreme strong scalability
Advanced, coupled multiphysics, multiscale
Optimization, UQ, solvers, discretizations
Interoperability, complementarity: xSDK
ECP Math libraries
Next-generation algorithms
Advances in data structures for new node architectures
Improving library quality, sustainability, interoperability
Toward predictive scientific simulations
Increasing performance, portability, productivity
Timeline:
xSDK release 1
xSDK release 2
. . . . .
xSDK release n

Math Libraries Approach:

As motivated & validated by the needs of ECP applications:

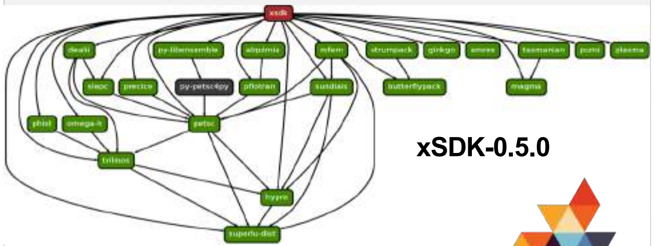
- Establish performance baselines
- Refactor, revise algorithms and data structures for new architectures
- Research into new numerical algorithms for next-generation predictive science

2.3.3. Math Libraries: Example Integration Activities with AD, ST, Facilities

xSDK – ECP Applications & Facilities

xSDK provides sustainable coordination and delivery of math libraries across independent efforts.

- xSDK-0.5.0 released Nov 2019
- **Impact:** Improving quality and sustainability, foundation for broader work on multi-level interoperability and performance



2016: **hypre, PETSc, SuperLU, Trilinos**

2017: **MAGMA, MFEM, SUNDIALS**

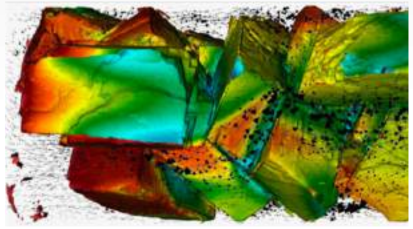
2018: **AMReX, deal.II, DTK, Omega_h, PHIST, PLASMA, PUMI, SLEPc, STRUMPACK, TASMANIAN**

2019: **ButterflyPACK, Ginkgo, libEnsemble, preCICE**

PETSc/TAO – Subsurface flow

PETSc provides GPU-enabled GAMG (geometric-algebraic multigrid) solvers to reduce simulation time for ECP subsurface simulations.

- PETSc-3.12 released Sept 2019 incorporates improved GPU support for GAMG solvers
  - Chombo-Crunch uses PETSc Krylov solvers + GAMG algebraic multigrid for Poisson and Helmholtz solves
- **Impact:** Faster overall runtime for ECP subsurface simulations

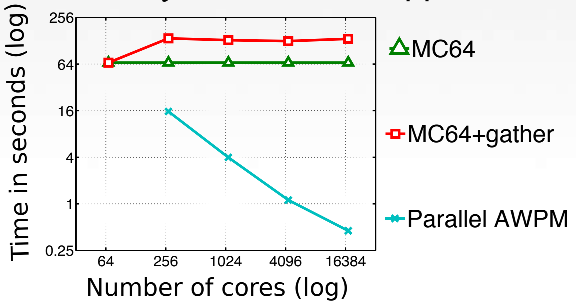


Chombo-Crunch simulated flow and pH inside a crushed calcite capillary experiment.

Optimized sparse GEMM (using sparse MKL primitives, rowmerge algorithm) for coarse operator construction in GAMG improve on-node performance.

STRUMPACK/SuperLU - ExaGraph

- **Factorization-based sparse solvers leverage ExaGraph co-design center partnership.**
- Developed distributed-memory maximum-weight perfect matching algorithm for stable pivot selection
  - https://www.exascaleproject.org/exagraph-with-strumpack-superlu/
- **Impact:** Faster sparse solvers, as needed by various ECP applications



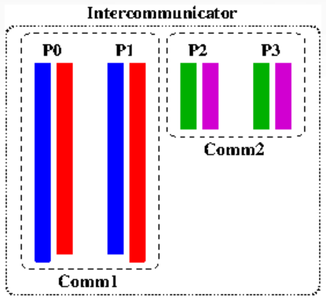
The parallel algorithm runs 300x faster than the sequential algorithm on 16K cores of NERSC/Cori.

2.3.3. Math Libraries: Example Integration Activities with AD, ST, Facilities

SUNDIALS/hypre – Multiphysics

SUNDIALS incorporates support for many-vectors, as needed for multiphysics in CEED co-design center and MARBL app (LLNL ATDM).

- SUNDIALS v5.0.0 (Oct 2019) includes many-vector support, where vector data structure can be a collection of vectors
- **Impact:** Improved multirate integrators for multiphysics, including MARBL and other apps served by MFEM (CEED)

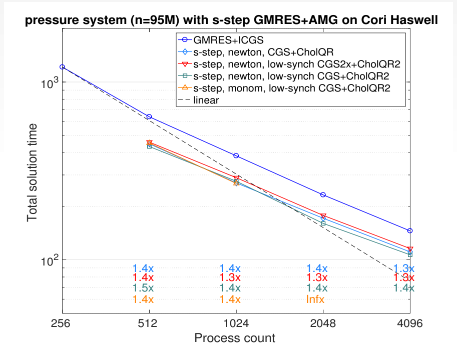


ManyVector in process-based multiphysics decompositions, where Comm1 connects processes 0 and 1, Comm2 connects processes 2 and 3; an MPI intercommunicator allows multiphysics coupling.

CLOVER – Predictive wind plant flow

Communication-avoiding and pipelined Krylov solvers provide advances for ExaWind app (predictive wind plant flow)

- Integration and performance assessment of MAGMA-sparse and CA/pipelined Krylov solvers (Trilinos) in ExaWind, presented at CSE19
- **Impact:** Faster overall runtime for ExaWind simulations

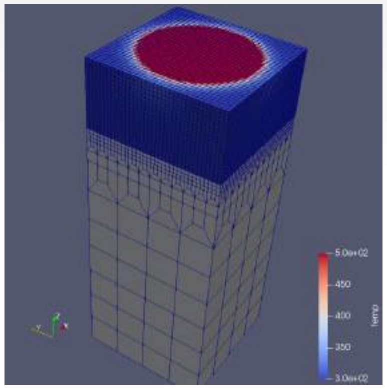


Performance assessment of new Krylov solvers for ExaWind (up to 1.5x on Cori Haswell).

ALExa – Additive manufacturing

DataTransferKit (DTK) provides scalable solution transfer for ExaAM app (additive manufacturing)

- Synthesized DTK testing in ExaAM and developed a set of problems and requirements (algs, performance)
- **Impact:** Efficient, scalable solution transfer via DTK in ExaAM coupled simulations



Test problems reflecting strong mismatch in the size of computational grids used in ExaAM coupling problems allow the assessment of accuracy and stability.

Deep Dive: 2.3.3.07: STRUMPACK / SuperLU:
Sparse direct solvers & preconditioners

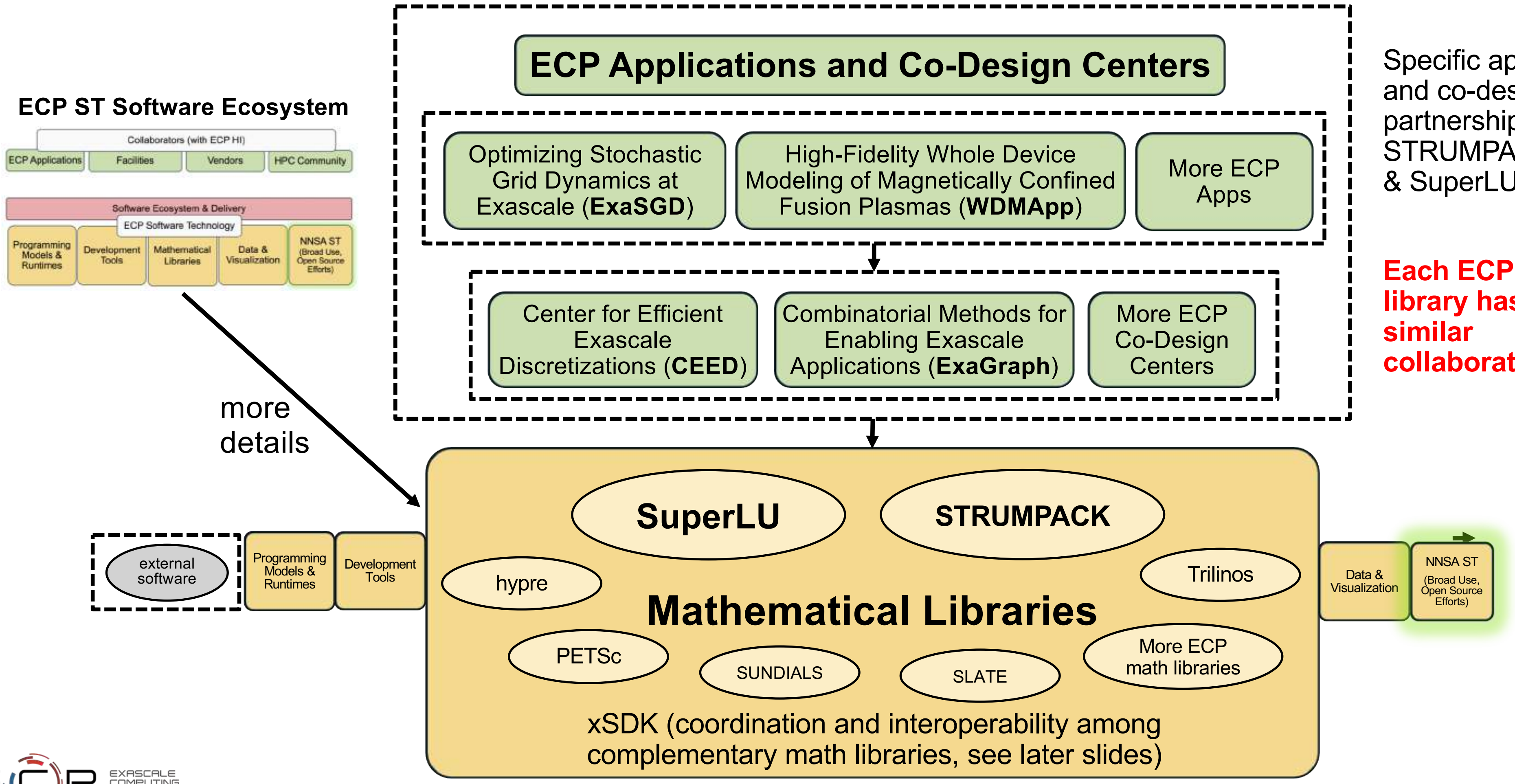
- Team:
  - PI: X. Sherry Li (LBNL)
  - Co-PI: Pieter Ghysels
  - Postdocs: Gustavo Chavez, Yang Liu

better scientific software

BSSw blog article:
X. Li, April 2018,
https://bssw.io/blog_posts/superlu-how-advances-in-software-practices-are-increasing-sustainability-and-collaboration

Scope & Intent	R&D Themes	Delivery Process	Target ECP Users	Support Model
Fill the gap of robust and scalable direct solvers and preconditioners for algebraic systems.	• Lower complexity approximation algorithms • Communication-reduction algorithms • Synchronization-reduction algorithms	• Regular release of software and documentation • Open access to production code at GitHub • In xSDK	• Many apps, including CEED, ExaSGD, WDMApp • Many math libraries, including hypre, PETSc, SUNDIALS, and Trilinos	• Email, issue tracking portals on GitHub • Comprehensive web-based documentation • Tutorials at various venues • Dedicated POC for each ECP app.

STRUMPACK / SuperLU: App users and roles in software ecosystem


ECP ST Software Ecosystem
Collaborators (with ECP HI)
ECP Applications
Facilities
Vendors
HPC Community
Software Ecosystem & Delivery
ECP Software Technology
Programming Models & Runtimes
Development Tools
Mathematical Libraries
Data & Visualization
NNSA ST (Broad Use, Open Source Efforts)
more details
ECP Applications and Co-Design Centers
Optimizing Stochastic Grid Dynamics at Exascale (ExaSGD)
High-Fidelity Whole Device Modeling of Magnetically Confined Fusion Plasmas (WDMApp)
More ECP Apps
Center for Efficient Exascale Discretizations (CEED)
Combinatorial Methods for Enabling Exascale Applications (ExaGraph)
More ECP Co-Design Centers
external software
Programming Models & Runtimes
Development Tools
SuperLU
STRUMPACK
hypre
Trilinos
Mathematical Libraries
PETSc
SUNDIALS
SLATE
More ECP math libraries
xSDK (coordination and interoperability among complementary math libraries, see later slides)
Data & Visualization
NNSA ST (Broad Use, Open Source Efforts)

Specific app and co-design partnerships for STRUMPACK & SuperLU

Each ECP math library has similar collaborations

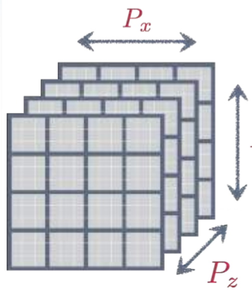
STRUMPACK / SuperLU: App users and roles in software ecosystem

- **Applications**
  - **ExaSGD:** Optimizing Stochastic Grid Dynamics at Exascale
    - Use STRUMPACK as inexact direct solver for saddle point systems in PIPS (parallel inter-point optimization solver)
    - Initial result with saddle-point system of order 1.5M from PIPS NLP: 6x improvement on 1536 cores
    - Implementing new algorithm for matrix inertia in STRUMPACK
  - **WDMApp:** High-Fidelity Whole Device Modeling of Magnetically Confined Fusion Plasmas
    - SuperLU/STRUMPACK used in multiphysics, multiscale magneto-hydrodynamics (MHD) codes
- **Co-Design Centers**
  - **CEED:** Center for Efficient Exascale Discretizations
    - Enhances MFEM with direct solvers and preconditioners for a range of challenging PDE problems
    - Integrated in MFEM; evaluating solvers for the electromagnetic problems in frequency domain
  - **ExaGraph:** Scalable parallel pivoting (ECP highlight)
- **Math Libraries**
  - **xSDK:** coordinated release and interoperability among complementary packages
  - **hypre:** provide coarse-grid solvers; **SUNDIALS:** provide internal linear solvers
  - **PETSc, Trilinos:** provide direct solver / preconditioner options
  - **SLATE**: Replace existing calls to LAPACK and ScaLAPACK with calls to SLATE counterparts
    - Identified and resolved interoperability issues, initial performance experiments
    - Led to significant extension and improvement of the LAPACK and ScaLAPACK compatibility APIs in SLATE

SuperLU: Advances during ECP

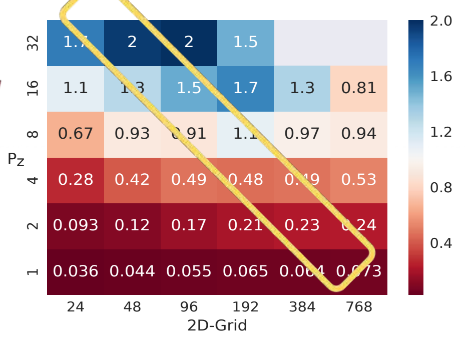
Novel communication-avoiding 3D sparse LU achieved 24x speedup on 4096 Titan nodes (32768 CPUs + 4096 GPUs)

- Need to reduce communication and increase parallelism for sparse LU factorization
- **Algorithm** innovation: 3D grid of MPI processes, Z-dimension has some data replication, but reduced communication and increased parallelism
- **Software** available in Version-7 release on GitHub
- **Theory**: communication volume reduced by sqrt(log(n)) factor, latency reduced by log(n) factor
- **Result**: compared to 2D pre-ECP code, 3D code achieved 27x speedup on 24,000 cores Edison, 24x speedup on 4096 Titan nodes with GPUs



IPDPS'18, JPDC'19

Matrix K2D5pt4096 on 24,576 cores Edison
Teraflop/s



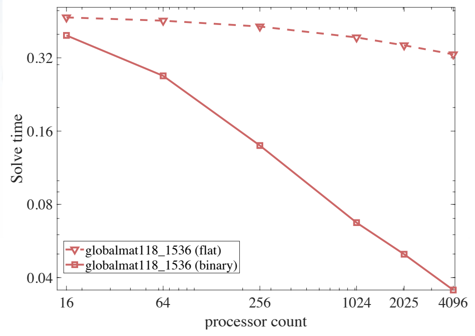
2D to 3D:
→ 23x speedup

32x more cores
→ 2x speedup

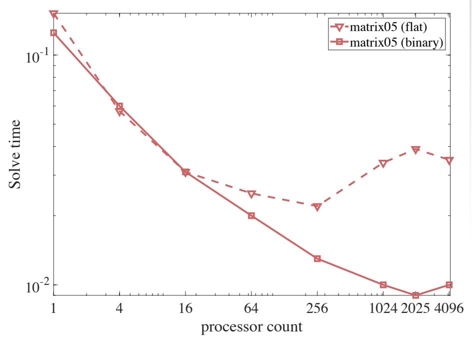
New synchronization-reducing sparse triangular solver achieved 7x speedup on Cori-Haswell cores

- When SuperLU is used in preconditioning, need many solves per factorization
- Developed **new algorithm** to reduce synchronization, remove communication hot-spot, improve overlap
  - Customized binary broadcast & reduction trees
  - Selected inversion of diagonal blocks
- **Software** available since Version-6 release

7x speedup for **ExaSGD** problem globalmat118_1536



Can use 10x more cores in strong scaling
WDMApp: M3D-C1 matrix05



SIAM CSC'18

EXASCALE COMPUTING PROJECT

Many ECP app teams rely on math libraries, often in combination

ECP AD Teams

Combustion-Pele, EXAALT, ExaAM, ExaFEL, ExaSGD, ExaSky, ExaStar, ExaWind, GAMESS, MFIX-Exa, NWChemEx, Subsurface, WarpX, WDMApp, WarpX, ExaAM, ATDM (LANL, LLNL, SNL) apps, AMReX, CEED, CODAR, CoPA, ExaLearn

Examples:

- **ExaAM:** DTK, hypre, PETSc, Sundials, Tasmanian, Trilinos, FFT, etc.
- **ExaWind:** hypre, KokkosKernels, SuperLU, Trilinos, FFT, etc.
- **WDMApp:** PETSc, hypre, SuperLU, STRUMPACK, FFT, etc.
- **CEED:** MFEM, MAGMA, hypre, PETSc, SuperLU, Sundials, etc.
- And many more …

ECP Math Libraries

hypre PETSc TAO

TRILINOS STRUMPACK MFEM

SuperLU sundials MAGMA PLASMA

ARBORX DTK ECP SLATE ICL

FFTX heFFTe

libEnsemble Ginkgo TASMANIAN

What’s new/different from a perspective of math libraries?

ECP apps need <u>sustainable coordination among math libraries,</u> along with other ST products – all with <u>continually advancing functionality</u>, exploiting new extreme-scale architectures

- **Needs of science applications**
  - Expanded needs for new math library functionality … new and broader/deeper collaborations
- **Software advances for exascale architectures**
  - Partnerships drive advances
- **Sustainable community software ecosystem**
  - xSDK approach and advances

Logos for Laghos, CEED, NALU, ExaWind, Truchas, ExaAM

xSDK release 0.5.0 for ECP

ECP WBS	WBS 2.3.3.01
PI	U.M. Yang (LLNL)
Members	ANL, LBNL, LLNL, ORNL, SNL, UC Berkeley, UTK

Milestone lead: Jim Willenbring, SNL

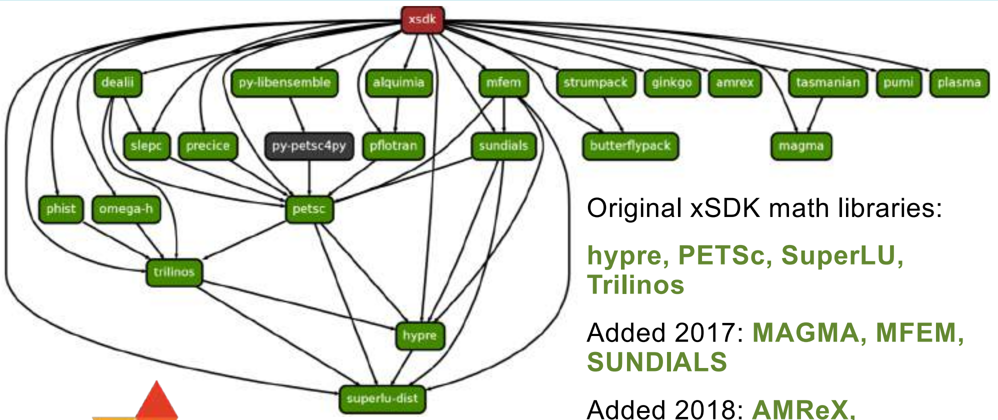
Scope and objectives

- Demonstrate the impact of community policies to simplify the combined use and portability of independently developed software packages.
- Increase formality of xSDK release process.
- Expand xSDK package members to include additional key ECP numerical libraries as well as packages in the broader community.

Project accomplishment and impact

- xSDK-0.5.0 released November 2019
- Improve access to numerical libraries for ECP apps.
- Lay the groundwork for addressing broader issues in software interoperability and performance portability.
- Build roles and experience for future exascale platform porting.

Spack graph for xSDK-0.5.0



xSDK Spack

tested on key platforms at ALCF, NERSC, and OLCF, also Linux and Mac OS X.

Original xSDK math libraries:

hypre, PETSc, SuperLU, Trilinos

Added 2017: **MAGMA, MFEM, SUNDIALS**

Added 2018: **AMReX, deal.II, DTK, Omega_h, PHIST, PLASMA, PUMI, SLEPc, STRUMPACK, TASMANIAN**

Added 2019: **ButterflyPACK, Ginkgo, libEnsemble, preCICE**

Deliverables https://xsdk.info/download https://xsdk.info/release-0-5-0 https://xsdk.info/installing-the-software https://github.com/xsdk-project/installxSDK https://xsdk.info/packages https://xsdk.info/policies

EXASCALE COMPUTING PROJECT

xSDK community policies

https://xsdk.info/policies

xSDK compatible package: Must satisfy mandatory xSDK policies:

M1. Support xSDK community GNU Autoconf or CMake options.
M2. Provide a comprehensive test suite.
M3. Employ user-provided MPI communicator.
M4. Give best effort at portability to key architectures.
M5. Provide a documented, reliable way to contact the development team.
M6. Respect system resources and settings made by other previously called packages.
M7. Come with an open source license.
M8. Provide a runtime API to return the current version number of the software.
M9. Use a limited and well-defined symbol, macro, library, and include file name space.
M10. Provide an accessible repository (not necessarily publicly available).
M11. Have no hardwired print or IO statements that cannot be turned off.
M12. For external dependencies, allow installing, building, and linking against an outside copy of external software.
M13. Install headers and libraries under <prefix>/include/ and <prefix>/lib/.
M14. Be buildable using 64 bit pointers. 32 bit is optional.
M15. All xSDK compatibility changes should be sustainable.
M16. The package must support production-quality installation compatible with the xSDK install tool and xSDK metapackage.

We welcome feedback. What policies make sense for your software?

Also **recommended policies**, which currently are encouraged but not required:

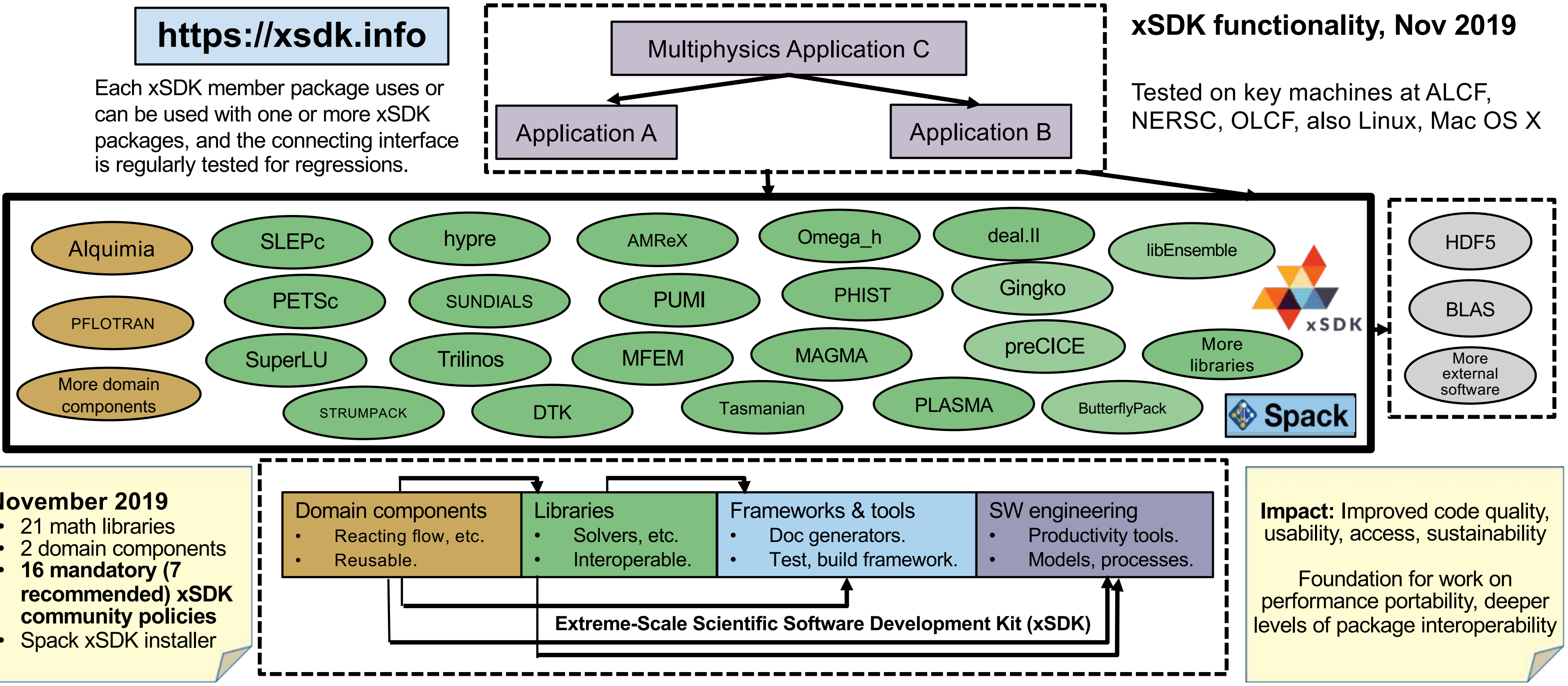
R1. Have a public repository.
R2. Possible to run test suite under valgrind in order to test for memory corruption issues.
R3. Adopt and document consistent system for error conditions/exceptions.
R4. Free all system resources it has acquired as soon as they are no longer needed.
R5. Provide a mechanism to export ordered list of library dependencies.
R6. Document versions of packages that it works with or depends on, preferably in machine-readable form
R7. Have README, SUPPORT, LICENSE, and CHANGELOG files in top directory.

xSDK member package: Must be an xSDK-compatible package, *and* it uses or can be used by another package in the xSDK, and the connecting interface is regularly tested for regressions.

BSSw blog article:
P. Luszczek and U. Yang, Aug 2019,
https://bssw.io/blog_posts/building-community-through-software-policies

xSDK version 0.5.0: November 2019

https://xsdk.info



xSDK functionality, Nov 2019

Tested on key machines at ALCF, NERSC, OLCF, also Linux, Mac OS X

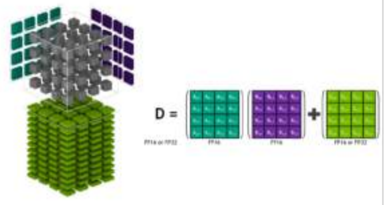
November 2019
- 21 math libraries
- 2 domain components
- **16 mandatory (7 recommended) xSDK community policies**
- Spack xSDK installer

Impact: Improved code quality, usability, access, sustainability

Foundation for work on performance portability, deeper levels of package interoperability

Math Libraries: Preparing for exascale platforms

- **Project reviews**, Sept 2019: Focus on GPU porting efforts, capability integration
  - Strategy, progress, challenges
- **Status and plans for on-node (and internode) parallel computing capabilities**
  - Sharing information among math libraries (via xSDK survey/report)
    - Also PMR (programming models and runtime) libraries, applications
- **New scope to address emerging needs: mixed precision for math libraries**
  - Centered in joint xSDK project for leverage across all math libraries
  - Goals:
    - Efficiently leverage compute power available in low precision tensor cores
    - Reduce arithmetic precision format complexity without impacting algorithmic stability
    - Reduce communication volume on all levels: main memory, I/O, inter-node communication
    - Enable higher performance for solvers, preconditioners, FFT, machine learning kernels, etc.



Volta Tensor Core matrix multiply and accumulate. [Source: NVIDIA Corp.]

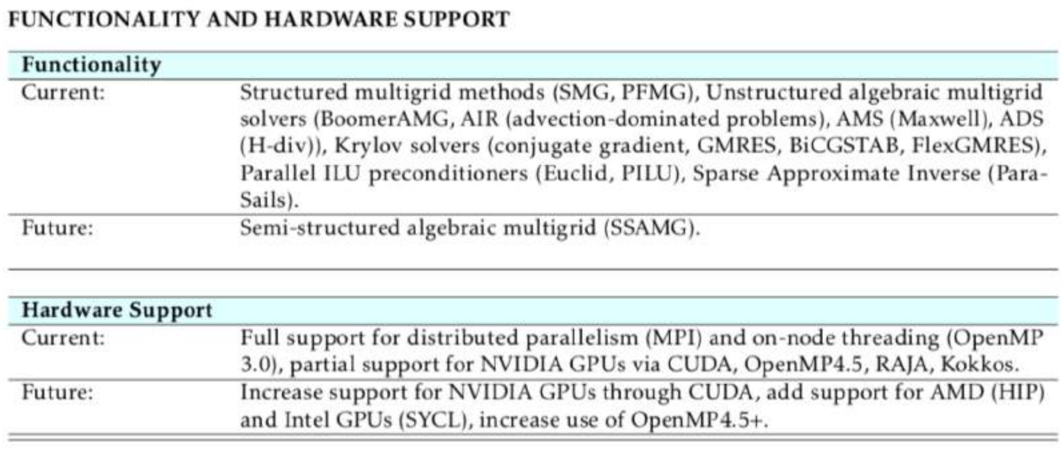
Breakout Session at 2020 ECP Annual Meeting

Speed Dating for ECP: 1-on-1 Conversations between Applications Teams and Math Library Developers (abstract)

Re-examine (1) what math functionalities ECP applications need, (2) what capabilities ECP math libraries provide and plan to develop in FY20-23, and (3) how to expand and deepen collaborations, so that together we can more effectively work to achieve next-generation science goals.

- Pre-meeting homework: Sharing 1-page summaries of math library capabilities and plans, using ECP Dependency Database (working with AD L3s, teams)

Contacts: Rob Falgout and Ulrike Yang (LLNL)



FUNCTIONALITY AND HARDWARE SUPPORT

Functionality	
Current:	Structured multigrid methods (SMG, PFMG), Unstructured algebraic multigrid solvers (BoomerAMG, AIR (advection-dominated problems), AMS (Maxwell), ADS (H-div)), Krylov solvers (conjugate gradient, GMRES, BiCGSTAB, FlexGMRES), Parallel ILU preconditioners (Euclid, PILU), Sparse Approximate Inverse (ParaSails).
Future:	Semi-structured algebraic multigrid (SSAMG).

Hardware Support	
Current:	Full support for distributed parallelism (MPI) and on-node threading (OpenMP 3.0), partial support for NVIDIA GPUs via CUDA, OpenMP4.5, RAJA, Kokkos.
Future:	Increase support for NVIDIA GPUs through CUDA, add support for AMD (HIP) and Intel GPUs (SYCL), increase use of OpenMP4.5+.

2020 ECP Annual Meeting

Software Development Kits (SDKs): Key delivery vehicle for ECP

A collection of related software products (packages) where coordination across package teams improves usability and practices, and foster community growth among teams that develop similar and complementary capabilities

- **Domain scope**
  Collection makes functional sense
- **Interaction model**
  How packages interact; compatible, complementary, interoperable
- **Community policies**
  Value statements; serve as criteria for membership
- **Meta-infrastructure**
  Invokes build of all packages (Spack), shared test suites
- **Coordinated plans**
  Inter-package planning. Augments autonomous package planning
- **Community outreach**
  Coordinated, combined tutorials, documentation, best practices

ECP ST SDKs: Grouping similar products for collaboration & usability

Programming Models & Runtimes Core

Tools & Technologies

Compilers & Support

Math Libraries (xSDK)

Viz Analysis and Reduction

Data mgmt., I/O Services & Checkpoint/ Restart

"Unity in essentials, otherwise diversity"

Extreme-scale Scientific Software Stack (E4S)

A Spack-based distribution of ECP ST products and related and dependent software tested for interoperability and portability to multiple architectures
Lead: Sameer Shende, University of Oregon

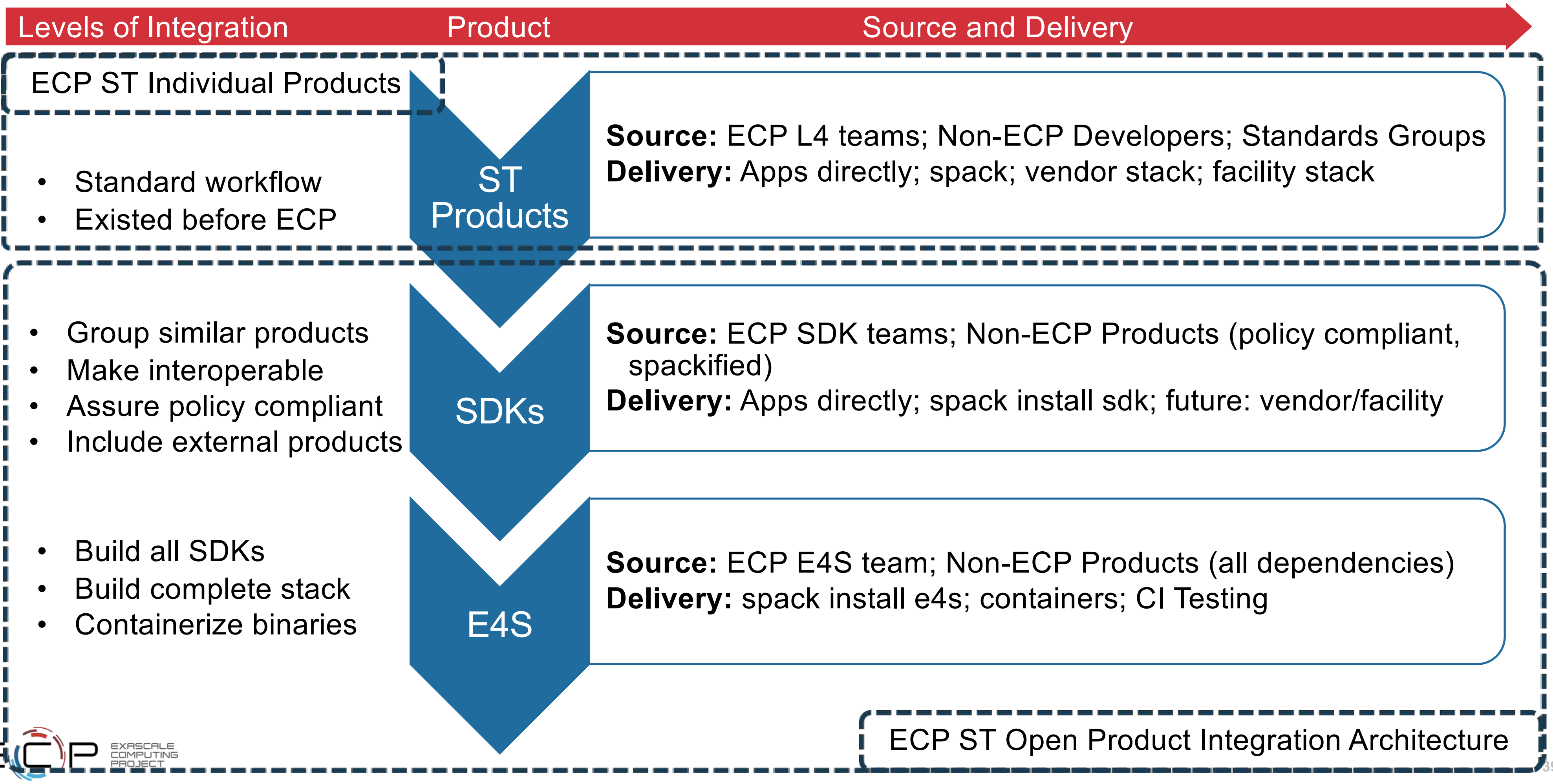
- Provides distinction between SDK usability / general quality / community and deployment / testing goals
- Will leverage and enhance SDK interoperability thrust
- Releases:
  - Oct 2018: E4S 0.1: 24 full, 24 partial release products
  - Jan 2019: E4S 0.2: 34 full, 10 partial release products
  - Nov 2019: E4S 1.0: 50 full, 6 partial release products
- Current primary focus: Facilities deployment
- Ideal mechanism for collaborations with other institutions, agencies, countries

http://e4s.io

E4S

EXASCALE COMPUTING PROJECT

Software Technology Ecosystem



Levels of Integration | **Product** | **Source and Delivery**

ECP ST Individual Products

- Standard workflow
- Existed before ECP

ST Products

Source: ECP L4 teams; Non-ECP Developers; Standards Groups
Delivery: Apps directly; spack; vendor stack; facility stack

- Group similar products
- Make interoperable
- Assure policy compliant
- Include external products

SDKs

Source: ECP SDK teams; Non-ECP Products (policy compliant, spackified)
Delivery: Apps directly; spack install sdk; future: vendor/facility

- Build all SDKs
- Build complete stack
- Containerize binaries

E4S

Source: ECP E4S team; Non-ECP Products (all dependencies)
Delivery: spack install e4s; containers; CI Testing

ECP ST Open Product Integration Architecture

Conclusions

- ECP had a very successful CD-2/3 review; ESAAB approval expected in February
- The AD/ST dependency data base has evolved significantly in the past 4 months
  - Verified by AD and ST teams
  - Used to manage critical and important dependencies
  - Gives key insights into what is needed at the Facilities and KPP-3 integration status with application projects
- ECP Math Libraries teams are:
  - Delivering to the supercomputing community (via DOE facilities and the xSDK / E4S.io effort) high-quality mathematical libraries that
    - provide scalable, robust algorithms that facilitate efficient exascale simulations
    - interoperate with the ECP software stack
    - can be readily used in combination by ECP applications
  - Integrating these math libraries capabilities with ECP applications … collaborating to achieve next-generation science goals

Questions?