New Trilinos Krylov Solvers
Many ECP application codes rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale computer efficiently, we are incorporating new solvers into the package.

Parallel Preconditioning
Preconditioners help applications solve large, sparse linear systems using iterative methods. Incomplete LU (ILU) factorizations are popular preconditioners for many problems, especially as single-node subdomain solvers for domain decomposition or multigrid smoothing. Typical ILU implementations use a truncated Gaussian Elimination process that has limited parallel scalability. This can quickly become a bottleneck, especially given the increasing parallelism of single nodes.

PARALLEL INCOMPLETE FACTORIZATIONS
We have developed and implemented highly parallel algorithms for computing incomplete factorization preconditioners. This includes both parallel-level based incomplete LU (“ParILU”) and parallel threshold-based incomplete factorizations (“ParILUT”). We have replaced Gaussian Elimination with iterative sweeps that approximate the nonzero entries of the ILU factors. Few sweeps generally suffice to compute a preconditioner of comparable or even superior quality to traditional ILU methods.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

Parallel Preconditioning
Preconditioners help applications solve large, sparse linear systems using iterative methods. Incomplete LU (ILU) factorizations are popular preconditioners for many problems, especially as single-node subdomain solvers for domain decomposition or multigrid smoothing. Typical ILU implementations use a truncated Gaussian Elimination process that has limited parallel scalability. This can quickly become a bottleneck, especially given the increasing parallelism of single nodes.

PARALLEL INCOMPLETE FACTORIZATIONS
We have developed and implemented highly parallel algorithms for computing incomplete factorization preconditioners. This includes both parallel-level based incomplete LU (“ParILU”) and parallel threshold-based incomplete factorizations (“ParILUT”). We have replaced Gaussian Elimination with iterative sweeps that approximate the nonzero entries of the ILU factors. Few sweeps generally suffice to compute a preconditioner of comparable or even superior quality to traditional ILU methods.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Parallel Preconditioning
Preconditioners help applications solve large, sparse linear systems using iterative methods. Incomplete LU (ILU) factorizations are popular preconditioners for many problems, especially as single-node subdomain solvers for domain decomposition or multigrid smoothing. Typical ILU implementations use a truncated Gaussian Elimination process that has limited parallel scalability. This can quickly become a bottleneck, especially given the increasing parallelism of single nodes.

PARALLEL INCOMPLETE FACTORIZATIONS
We have developed and implemented highly parallel algorithms for computing incomplete factorization preconditioners. This includes both parallel-level based incomplete LU (“ParILU”) and parallel threshold-based incomplete factorizations (“ParILUT”). We have replaced Gaussian Elimination with iterative sweeps that approximate the nonzero entries of the ILU factors. Few sweeps generally suffice to compute a preconditioner of comparable or even superior quality to traditional ILU methods.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.

Overview
Many large-scale scientific applications rely heavily on preconditioned iterative solvers for large linear systems. For these solvers to use extreme-scale hardware efficiently, both algorithms and their implementations must be redesigned to address challenges like extreme concurrency, complex memory hierarchies, costly data movement, and heterogeneous node architectures.

The PEEKS effort aims to tackle these challenges and advance the capabilities of the ECP software stack by making the new scalable algorithms accessible within the Trilinos software ecosystem. Targeting exascale–enabled Krylov solvers, incomplete factorization routines, and parallel preconditioning techniques will ensure successful delivery of scalable Krylov solvers in robust, production–quality software that can be relied on by ECP applications.

s-STEP/Pipelined Krylov Solver
To improve Krylov solvers’ parallel scalability, the s-step and pipelined methods aim to reduce and hide the latency costs of global collectives. We have used Trilinos components to develop prototype implementations of such solvers.

PORTABLE IMPLEMENTATION
Trilinos software stack provides portable implementation over different architectures including GPUs. We are studying their performance using linear systems from ECP application codes such as LAMMPS used in EXAALT.