

Kokkos: Performance Portability for C++ Codes



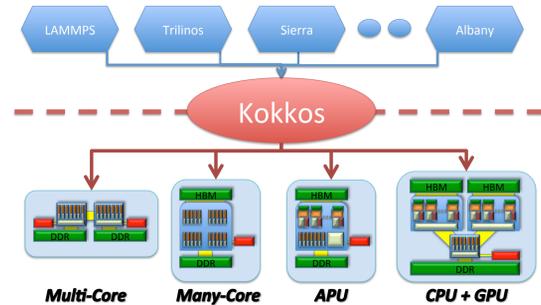
Sandia National Laboratories
H.C. Edwards, C.R. Trott, D. Sunderland, N. Ellingwood,
G.E. Mackey, S.D. Hammond

Los Alamos National Laboratories
G. Shipman (Kokkos-Support)
Oak Ridge National Laboratories
F. Foerster (Kokkos-Support)

ECP 1.3.1.05 Development
ECP 1.3.1.12 Support

Why another Programming Model?

Programming applications for performance and portability across modern computing architectures is extremely challenging for many application developers. Hardware-native programming models may provide much higher performance but may only run on one vendor or even one machine. Standardized cross-platform models provide greater porting potential but often much lower performance. With a C++ abstraction-based model you can have the best of both worlds.



github.com/kokkos

- Code Repository
- Issue Tracking
- Tools Repository
- Tutorials
- Documentation



Capabilities

Concept	Example
Parallel Loops	<code>parallel_for(N, KOKKOS_LAMBDA (int i) { ...BODY... });</code>
Parallel Reduction	<code>parallel_reduce(RangePolicy<ExecSpace>(0,N), KOKKOS_LAMBDA (int i, double&upd) { ...BODY... upd += ... }, result);</code>
Tightly Nested Loops (exp)	<code>parallel_for(MDRangePolicy<Rank<3>>({0,0,0},{N1,N2,N3}},{T1,T2,T3}, KOKKOS_LAMBDA (int i, int j, int k) { ...BODY...});</code>
Non-Tightly Nested Loops	<code>parallel_for(TeamPolicy<Schedule<Dynamic>>(N, TS), KOKKOS_LAMBDA (Team team) { ... COMMON CODE 1 ... parallel_for(TeamThreadRange(team, M(N)), [&] (int j) { ... INNER BODY... }); ... COMMON CODE 2 ... });</code>
Task Dag (exp)	<code>task_spawn(TaskTeam(scheduler , priority), KOKKOS_LAMBDA (Team team) { ... BODY });</code>
Data Allocation	<code>View<double**, Layout, MemSpace> a("A",N,M);</code>
Data Transfer	<code>deep_copy(a,b);</code>

Sibling Projects

Kokkos-Tools	Profiling and Debugging Tools. Can be used on release builds. No overhead if not loaded. Interface to third party tools (VTune, NVProfi etc.).
Kokkos-Kernels	BLAS, Sparse, and Graph Kernels. Takes Kokkos Views. Can internally call vendor libraries if data type / layout matches.

Collaborators

Users



Backend Opt.



Applications/Libraries

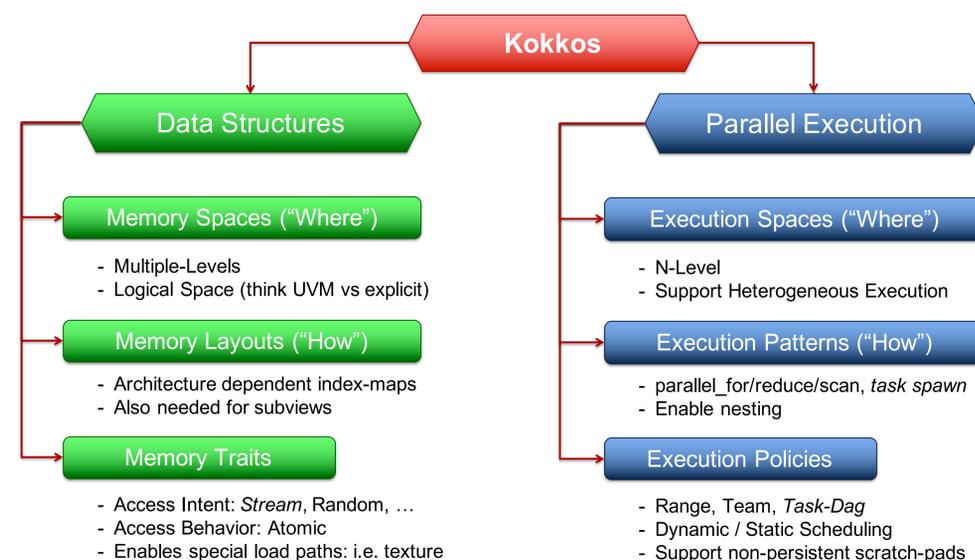


Uintah

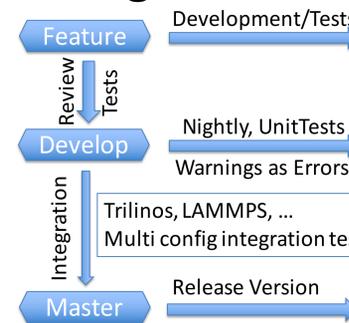


Sierra Mechanics

Abstraction Concepts



Testing and Software Quality



New Features are developed on forks, and branches. Limited number of developers can push to develop branch. Pull requests are reviewed/tested.

Compilers	GCC (4.8-6.0), Clang (3.6-4.0), Intel (15.0-17.1), IBM (13.1.5), PGI (17.1), NVIDIA (7.0-8.0)
Hardware	Intel Haswell, Intel KNL, ARM v8, IBM Power8, NVIDIA K80, NVIDIA P100
Backends	OpenMP, Pthreads, Serial, CUDA

Each merge into master is minor release. Extensive integration test suite ensures backward compatibility, and catching of unit-test coverage gaps.

Kokkos Support – An ECP Project

"ECP Applications effective use of Kokkos to achieve performance portability across exascale architectures"

- Tutorials**: Extensive tutorials available at: github.com/kokkos/kokkos-tutorials. Regularly given at conferences such as SC and GTC. Extra tutorials by request are possible.
- Online Support**: Active Github issues list with typical same day response time. All planning activities are usually publicly available there. Sensitive questions (e.g. with respect to export controlled codes) are handled via email.
- Hackathon**: Hackathons will be organized at Oak Ridge and Sandia National Laboratories. Preference given to ECP applications and libraries.
- Application StartUp**: Explicit application support beyond tutorials, hackathons and email support is organized on a case by case basis.