Early Experience with P100 on Power8

Christian Trott - Center for Computing Research
Sandia National Laboratories/NM
Sandia’s Path Forward

- Mandate to run across all DOE Leadership class Systems
  - Cori -- NERSC – KNL (2017)
- We need performance portable code:
  - Kokkos for C++ Applications (majority of our apps)
  - OpenMP for Fortran
Kokkos: Performance, Portability and Productivity

https://github.com/kokkos
Kokkos: Performance Portability through Abstraction

**Kokkos**

- **Data Structures**
  - Memory Spaces ("Where")
    - Multiple-Levels
    - Logical Space (think UVM vs explicit)
  - Memory Layouts ("How")
    - Architecture dependent index-maps
    - Also needed for subviews
  - Memory Traits
    - Access Intent: Stream, Random, ...
    - Access Behavior: Atomic
    - Enables special load paths: i.e. texture

- **Parallel Execution**
  - Execution Spaces ("Where")
    - N-Level
    - Support Heterogeneous Execution
  - Execution Patterns ("What")
    - parallel_for/reduce/scan, task spawn
    - Enable nesting
  - Execution Policies ("How")
    - Range, Team, Task-Dag
    - Dynamic / Static Scheduling
    - Support non-persistent scratch-pads
OpenMP Support/Interoperability

- OpenMP 4.5+ backend for Kokkos

  - **Goal:** interoperability of C++ apps/libraries in native OpenMP4.5 with Kokkos libraries/apps.

  - On branch: basic features are now working, including data management and simple data parallelism
  - Expected to go into main-line Kokkos early next year.

- Working closely with IBM CORAL OpenMP research compiler group and IBM XL

https://www.ibm.com/developerworks/community/groups/service/html/communitystart?communityUuid=8e0d7b52-b996-424b-bb33-345205594e0d
P100 + POWER8: a test bed for Sierra

- P100 + POWER8 have common features with Sierra/Summit
  - NVLink: First generation now, second generation later
  - IBM Software stack (XL compiler, Cuda for Power)
  - HBM memory
  - Infiniband Network
  - Significant new P100 feature: Hardware double precision atomic add
- Get software to work
  - Port applications to POWER (not always trivial due to different memory model from X86)
- Benchmarks Focus on Kokkos: using GCC 5.3.0 / CUDA 8.0.44

Possible through dedicated TestBed team at Sandia: Special thanks to Si Hammond
Synthetic Benchmarks HBM

[kokkos/benchmarks/bytes_and_flops]
Synthetic Benchmarks NVLINK

*kokkos/benchmarks/bytes_and_flops*
(Set MemorySpace of Arrays to HostPinned or UVM)
LAMMPS – A Science Application

Lennard Jones

EAM

Tersoff

Million Atomsteps per Second

Million Atomsteps per Second

Million Atomsteps per Second

K40 Full  K80 Full  P100 Full  K40 Half  K80 Half  P100 Half  P100 Half-HWAtomic
Finite Element Assembly

- Our Finite Element Codes have 3 Phases:
  - Graph Construction (reused over multiple timesteps)
  - Matrix Assembly (20%-50%)
  - Solve (30%-80%)
- Matrix Assembly
  - Perform Physics Calculations
  - Add contributions to multiple matrix entries
  - Major limiter is number of load/store operations in flight
  - P100 add hardware atomic add for doubles (used CAS loops before)
Conclusion

- P100 delivers increased performance for our algorithms
  - Main improvement seems to come from memory-system
  - More loads/stores in flight
  - 2x-5x improvement over K40/K80
- Hardware Atomic Add for double helps significantly
  - About 20%-50% improvement vs using CAS Loops for algorithms
  - In many cases makes non-atomic algorithms obsolete
- What about NVLink?
  - Codes ported to GPUs are not designed to benefit from it ...
  - Codes which need it are not yet ported ...
Kokkos Demo Station at DOE booth: today 2-4pm