Multi-scale Application Software Development Ecosystem on ARM

Dr. Xiaohu Guo
STFC Hartree Centre, UK
Overview

• Multiscale simulation framework
• Our early porting experience on Isambard ARM thunderX2 system
• Discussion and the future work
Multiple Scales of Materials Modelling

- FF mapping via DL FIELD
- Coarse graining via DL_CGMAP
- MS&MD via DL_POLY
- DPD & LB via DL_MESO
- KMC via DL_AKMC
- QM/MM bridging via #ChemShell

Accuracy:
- geometrical
- topological
- qualitative
- 1 eV / 40 kT
- 0.1 eV / 4 kT
- 0.0001 eV / 0.004 kT

Atoms: 1, 10, 100, 1000, 1,000,000
Time: 0, 0 ps, 100 ns, 1 µs, 10 ms, ∞

GAP in accuracy and speed
**Multi-scale Simulation Software Eco-system**

*ChemShell* - interface to bridge QM & MM and provide the glue for workflow ability.

### Model Set-up
**DL_FIELD**

### Engines
**DL_POLY/MESO/MONTE/AKMC/etc.**
- MD, MS, MC - thermodynamics,
- DPD, PIMD, shaped particles (GB),
- multipolar electrostatics, reactive dynamics,
- advanced sampling

Hybrid:
- MD/MC
- MD/DFTB
- on-the-fly analysis

### Simulation Analysis
**DL_ANALYSER**

- Interoperability for Marketplace R&D Services-On-Demand
- Translation
- Model Set-up
- RSE Development
- Simulation
- Analysis
- Training

---

**Inter-Scale Best Practices, Digital Twining, Machine Learning**
User Community

Annual Downloads & Valid eMail List Size

2010 :: DL_POLY (2+3+MULTI) - 1,000 (list end)
2017 :: DL_POLY_4 - 4,200 (list start 2011)

2016 Downloads
- UK - 19.2%
- EU-UK - 18.7%
- USA - 11.4%
- India - 10.3%
- China - 9.4%
- France - 5.9%
- London - 5.5%
- Sofia - 2.0%
- Beijing - 1.8%

2017 :: DL_POLY_4 - 4,200 (list start 2011)

Annual Downloads & Valid eMail List Size

User Community
Proteins solvation & binding

DNA strands dynamics

Membranes' processes

Drug polymorphs & discovery

Crystalline & Amorphous Solids – damage and recovery

Dynamic processes in Metal–Organic & Organic Frameworks

Dynamics at Interfaces & of Phase Transformations

Thanks to Dr. Ilian Todorov
DL_MESO: Meso scale simulation Toolkit

- General-purpose, highly-scalable mesoscopic simulation software (developed for CCP5/UKCOMES)
  - Lattice Boltzmann Equation (LBE)
  - Dissipative Particle Dynamics (DPD)

- >800 academic registrations (science and engineering)
- Extensively used for Computer Aided Formulation (CAF) project with TSB-funded industrial consortium

Thanks to Dr. Michael Seaton
CFD software in macro-scale region

IMPORTANCE: Hartree Centre key technologies, align with SCD missions and STFC global challenge schemes.

- Fluidity
- FEM
- SPH/ISPH

Applications:
- Nuclear
- Schlumberger oil reservoir
- Manchester Bob
- Wave impact on BP oil rig
- NERC ocean roadmap
- EPSRC MAGIC
- Tsunami
Concurrent Coupling Toolkit: MUI

Data Exchange Interface

Data Points

DPD and SPH Coupling

Yu-Hang Tang, etc.
Porting the software framework
On ARM Platform
Isambard system specification

Isambard PI:
Prof Simon McIntosh-Smith
University of Bristol / GW4 Alliance

- 10,752 Armv8 cores (168 x 2 x 32)
  - Cavium ThunderX2 32 core 2.1GHz
- Cray XC50 ‘Scout’ form factor
- High-speed Aries interconnect
- Cray HPC optimised software stack
  - CCE, Cray MPI, math libraries, CrayPAT, ...
- Phase 2 (the Arm part):
  - Delivered Oct 22nd
  - Handed over Oct 29th
  - Accepted Nov 9th!
Performance on mini-apps (node level comparisons)

Thanks to Prof. Simon McIntosh-Smith
Single node performance results

https://github.com/UoB-HPC/benchmarks

Thanks to Prof. Simon McIntosh-Smith
Earlier DLPOLY Performance Results

![Graph showing speedup vs. number of threads for different force calculation methods.

- two_body_forces dynamic
- two_body_forces static
- two_body_forces guided
- link_cell_pairs dynamic
- link_cell_pairs static
- link_cell_pairs guided
- Total time dynamic
- Total time static
- Total time guided]
Earlier DLMESO Performance Results

DL_MESO Performance on ARM

- Speedup
- Number of cores

Graph showing the performance of DL_MESO on ARM processors with different numbers of cores.
Earlier ISPH Performance Results

ISPH Performance on ARM
Total Number of Particles = 125500

- Total Wall Time
- Pressure Poisson Solver
- Kernel Calculation

Number of MPI Processes vs. Time [Secs]
Performance comparing with our Scafellpike

Node Performance Comparison with Skylake and ARM

<table>
<thead>
<tr>
<th></th>
<th>Skylake</th>
<th>ThunderX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLPOLY</td>
<td>1.0</td>
<td>0.64</td>
</tr>
<tr>
<td>DLMESO</td>
<td>1.0</td>
<td>0.55</td>
</tr>
<tr>
<td>ISPH</td>
<td>1.0</td>
<td>1.34</td>
</tr>
</tbody>
</table>

Performance Normalised to Skylake
Current Arm software ecosystem

Three mature compiler suites:
- GNU (gcc, g++, gfortran)
- Arm HPC Compilers based on LLVM (armclang, armclang++, armflang)
- Cray Compiling Environment (CCE)

Three mature sets of math libraries:
- OpenBLAS + FFTW
- Arm Performance Libraries (BLAS, LAPACK, FFT)
- Cray LibSci + Cray FFTW

Multiple performance analysis and debugging tools:
- Arm Forge (MAP + DDT, formerly Allinea)
- CrayPAT / perftools, CCDB, gdb4hpc, etc
- TAU, Scalasca, Score-P, PAPI, MPE
More ARM productivity features needed!

- ARM processor does not trap integer divide by Zero
  - Architectural decision – no signal thrown
  - Will return zero \(1/0 == 0\)
  - Do trap float divide by zero SIG-FPE

- Need latest autoconf and automake, update your config.guess and config.sub

- Weak memory model:
  - your threading lock-free implementation may not work here!

- How can we use Nvidia GPUs?
- More math libraries?
  - DD/DLB libraries?
  - Sparse linear solvers? Particular theaded libraries?
Software Ecosystem on Isambard.

Build recipes online at https://gitlab.com/arm-hpc/packages/wikis/home
Motivation: Performance Optimization Space

Application Performance

Serial Efficiency
- IPC Scaling Efficiency
- Frequency Efficiency
- Instruction Scaling Efficiency

Parallel Efficiency
- Load Balance
- Repartition Efficiency
- Synchronization Efficiency
- Communication Efficiency

Data Transfer Efficiency

Partition Quality
- Instruction Mix
- Code Replication

Cache
- Memory BW
- Dependencies
- Sharing Effects
- NUMA Effects

SM Synchronization
- OS noise
These are **early results**, generated quickly in the first few days with no time to tune scaling etc. We expect the results to improve even further as we continue to work on them.

The software stack has been robust, reliable and high-quality (both the commercial and open source parts)
<table>
<thead>
<tr>
<th>Functional Areas</th>
<th>Components include</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base OS</td>
<td>CentOS 7.4, SLES 12 SP3</td>
</tr>
<tr>
<td>Administrative Tools</td>
<td>Conman, Ganglia, Lmod, LosF, Nagios, pdsh, pdsh-mod-slurm, prun, EasyBuild, ClusterShell, mrsh, Genders, Shine, test-suite</td>
</tr>
<tr>
<td>Provisioning</td>
<td>Warewulf</td>
</tr>
<tr>
<td>Resource Mgmt.</td>
<td>SLURM, Munge</td>
</tr>
<tr>
<td>I/O Services</td>
<td>Lustre client (community version)</td>
</tr>
<tr>
<td>Numerical/Scientific Libraries</td>
<td>Boost, GSL, FFTW, Metis, PETSc, Trilinos, Hypre, SuperLU, SuperLU_Dist,Mumps, OpenBLAS, Scalapack, SLEPc, PLASMA, ptScotch</td>
</tr>
<tr>
<td>I/O Libraries</td>
<td>HDF5 (pHDF5), NetCDF (including C++ and Fortran interfaces), Adios</td>
</tr>
<tr>
<td>Compiler Families</td>
<td>GNU (gcc, g++, gfortran), LLVM</td>
</tr>
<tr>
<td>MPI Families</td>
<td>OpenMPI, MPICH</td>
</tr>
<tr>
<td>Development Tools</td>
<td>Autotools (autoconf, automake, libtool), Cmake, Valgrind,R, SciPy/NumPy, hwloc</td>
</tr>
<tr>
<td>Performance Tools</td>
<td>PAPI, IMB, pdtoolkit, TAU, Scalasca, Score-P, SIONLib</td>
</tr>
</tbody>
</table>
Thanks, Questions?
GROMACS scalability, up to 8,192 cores

Thanks to Prof. Simon McIntosh-Smith