PORTING CP2K TO THE INTEL XEON PHI

ARCHER Technical Forum, Wed 30th July

Iain Bethune (ibethune@epcc.ed.ac.uk)

Outline

- Xeon Phi Overview
- Porting CP2K to Xeon Phi
- Performance Results
- Lessons Learned

- Further Reading

Xeon Phi Overview

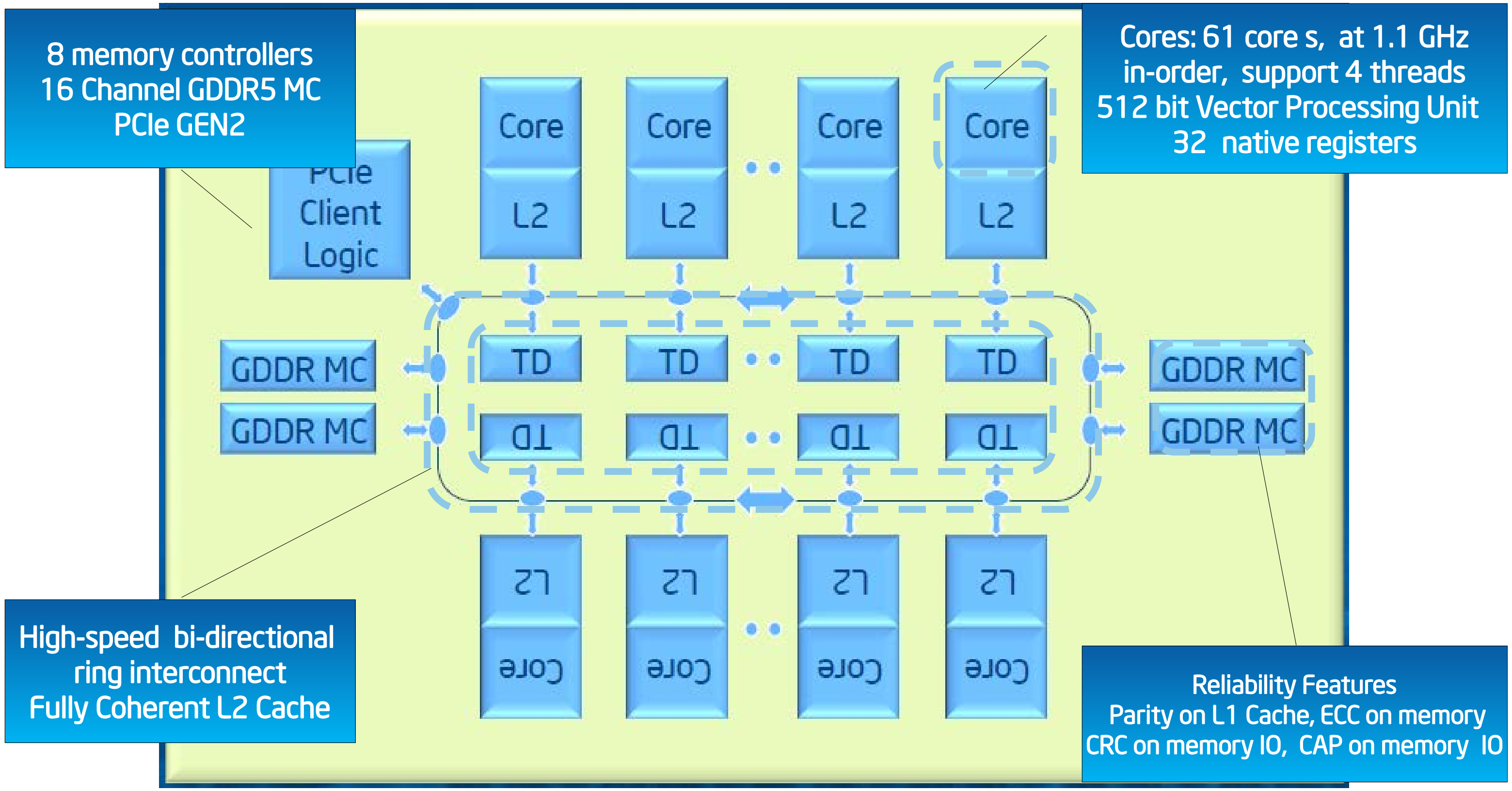
- Terminology:
  - Project Larrabee
    - Intel research project – planned to develop a GPGPU architecture shelved in 2010
  - Many Integrated Core (MIC)
    - Scalable architecture for many-core computing based on x86 cores
  - Intel® Xeon Phi™
    - Product name e.g. Xeon Phi 5110P
  - Code names:
    - Knight's Ferry – MIC prototype (2010), 45 nm
    - Knight's Corner – current generation (2013), 22 nm
    - Knight's Landing – announced (2015), 14 nm

Xeon Phi Overview

- Xeon Phi 5110P
  - Co-processor attached via PCIe bus
  - 60 cores @ 1.053 GHz
  - 4 virtual threads per core
  - 512-bit (8 doubles) vector unit per core
  - Coherent L2 cache
  - 8 GB main memory
  - 1.011 GFLOP/s per co-processor

Xeon Phi Overview

Image © Intel, ARCHER Xeon Phi Training Course



Xeon Phi Overview

- Operation Modes:
  - Native Mode
    - Xeon Phi runs cut-down Linux
    - SSH in and launch (parallel) program
    - Later case study used native mode only
  - Offload Mode
    - Main program thread runs on host CPU
    - Marked-up regions are offloaded for execution on Xeon Phi
  - OpenCL
    - Main program runs on CPU
    - Launch OpenCL kernels on the Xeon Phi

Xeon Phi Overview

- Programming Models:
  - Intel Compiler
    - Standard C, C++, Fortran …
    - Required to generate vector instructions
  - MPI
    - In native mode on single Xeon Phi (or across several)
    - In offload mode between host CPUs
    - Single communicator spanning host and MIC
  - OpenMP
    - Within a kernel in offload mode
    - In native mode, possibly hybrid with MPI
  - Intel TBB, Cilk+, OpenCL, OpenACC…

Porting CP2K to Xeon Phi

"CP2K is a program to perform atomistic and molecular simulations of solid state, liquid, molecular, and biological systems. It provides a general framework for different methods such as e.g., density functional theory (DFT) using a mixed Gaussian and plane waves approach (GPW) and classical pair and many-body potentials."

From www.cp2k.org (2004!)

Porting CP2K to Xeon Phi

- Many force models:
  - Classical
  - DFT (GPW)
  - Hybrid Hartree-Fock
  - LS-DFT
  - post-HF (MP2, RPA)
- Simulation tools
  - MD (various ensembles)
  - Monte Carlo
  - Minimisation (GEO/CELL_OPT)
  - Properties (Spectra, excitations …)
- Open Source
  - GPL, www.cp2k.org
  - 1m loc, ~2 commits per day
  - ~10 core developers
  - 3rd most used code on ARCHER

archer

epcc

THE UNIVERSITY OF EDINBURGH

Porting CP2K to Xeon Phi

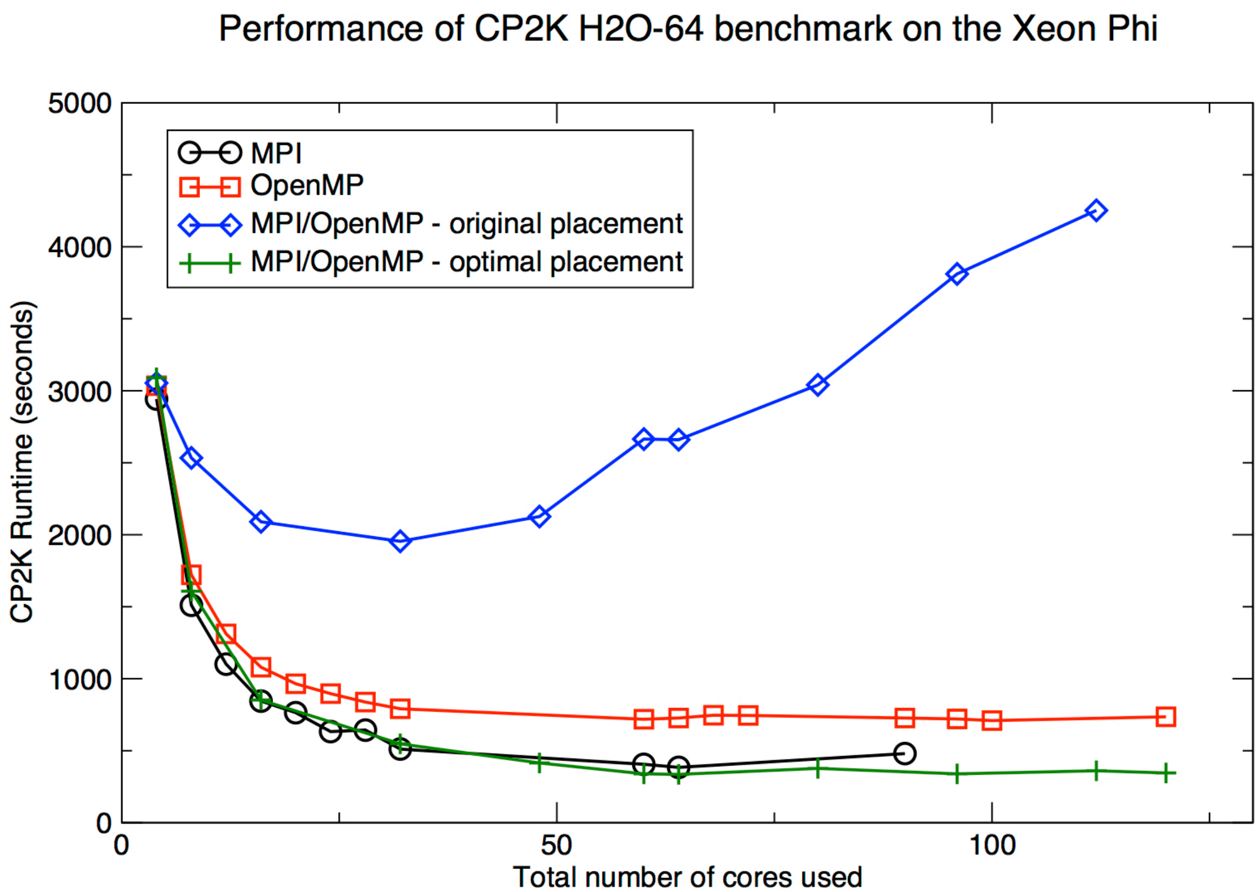
- Work done thanks to funding from PRACE

- Porting to Intel Compiler
  - Just add `-mmic` !
  - Fixed several bugs (in CP2K and in Intel compiler / MKL)
  - Recommend using ifort 14.1 and MKL 11.1

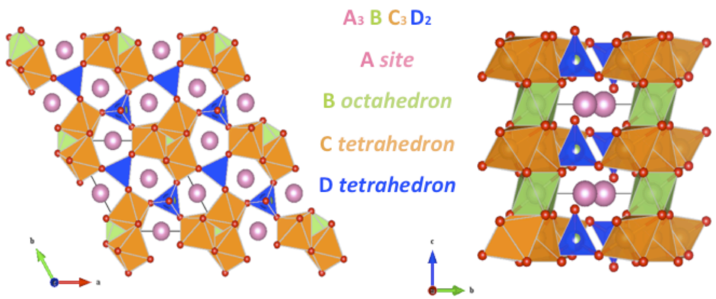
- Using MIC-optimised libraries
  - MKL up to 5x faster than FFTW 3.3.3 for 1D FFT (n=256)
  - MKL up to 3x faster for 3D FFT (n=128)

Porting CP2K to Xeon Phi

- Task placement – how to place 240 virtual threads?



Performance Results



- Initial port (no optimisations):
  - Langasite relaxation (DFT)
  - 8 MPI x 16 OpenMP (128 threads)
  - Around ~8x slower than 8-core Sandy Bridge CPU

- Optimisation:
  - Main bottlenecks are poor scaling with OpenMP
    - Parallelised two expensive routines
  - Total memory usage also a constraint

- Finally, still 5.4x slower than the CPU!

archer

epcc

THE UNIVERSITY OF EDINBURGH

Lessons Learned

- Porting is relatively easy
  - If the source code is portable…
  - … and already parallelised

- Native mode provides an easy way in to using Xeon Phi
  - But finding enough parallelism to fill 240 threads with memory limit of 8 GB is a hard strong-scaling problem
  - Different approach to multi-core, memory rich on-node parallelism
  - Fine-grained parallelism required (e.g. data parallel)
  - Could scale across multiple MICs to harness more memory, but this comes with additional overhead

Lessons Learned

- Offload mode
  - Complex logic, function calls are much slower on KNC core than on Xeon
  - For Knight's Corner, might be better to run use offload mode and only run suitable kernels on the Xeon Phi

- Serial Optimisation
  - Good auto-vectorisation, sometimes necessary to align and/or pad arrays.
  - P54C cores are slow, and memory bandwidth is limited. Expect this to improve on KNL.

Summary

- Xeon Phi offers promising performance gains
  - Comparable performance to current GPUs
  - Works best with data parallel codes with large amount of fine-grained parallelism
- Easy to use
  - Thanks to support for existing common programming models
  - Still need a highly parallel algorithm
- Knight’s Landing expected 2015
  - Self-hosting
  - Higher memory b/w (stacked memory) + 384 GB DDR4
  - AVX-512 vectorisation and Atom cores.

Further Reading

- Programming the Xeon Phi (ARCHER training course materials)
  - https://www.archer.ac.uk/training/course-material/2014/06/XeonPhi_Bristol/
- Evaluating CP2K on Exascale Hardware: Intel Xeon Phi
  - http://www.prace-ri.eu/IMG/pdf/wp152.pdf
- Optimising CP2K for the Intel Xeon Phi
  - http://www.prace-ri.eu/IMG/pdf/wp140.pdf
- Intel Xeon Phi Webinar:
  - https://software.intel.com/en-us/articles/intel-xeon-phi-webinar