cerit scientific cloud

Introduction to Xeon Phi "Knight Landing"

Jiří Filipovič
Institute of Computer Science
Masaryk University

30. března 2017

UNIVERSITAS MASARYKIANA BRUNENSIS

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

cerit scientific cloud

Introduction

Xeon Phi

- Intel Many Integrated Core Architecture (MIC)
- competitor of GPU processors (high floating point per second, high memory bandwidth)
- programmable as CPU (C++, OpenMP, MPI, . . .)

Knights Landing Architecture

- bootable processor (previous generation was PCI-E accelerator only)
- up to 72 cores
- 16 GB high-bandwidth memory (HBM)
- up to 384 GB DDR4 memory

cerit scientific cloud

KNL core

Core is based on Atom Airmont

- scalar performance significantly lower than Xeon CPU
- AVX-512 instruction set: 2× more flops per cycle comparing AVX2+FMA3
- SMT-4: four virtual cores using hyper threading

Binary compatible with x86_64

- the same programming model, OS, tools
- existing software runs out of the box, but re-optimization may help

cerit scientific cloud

KNL memory

Main memory

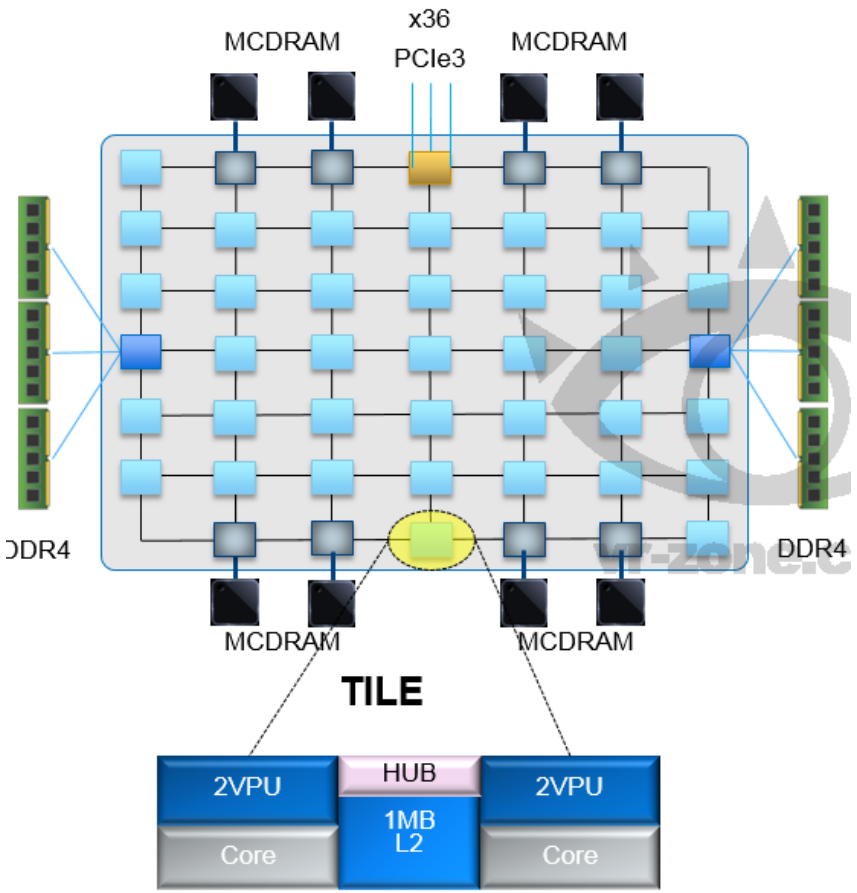
- the same as contemporary Xeon-based systems (DDR4)
- about 100 GB/s
- symmetric access by default

High-bandwidth memory

- limited size 16 GB
- about 450 GB/s
- explicit NUMA node, cache or both

cerit scientific cloud

KNL architecture



KNL in CERIT-SC

6 nodes equipped with Xeon Phi 7210

- phi1-phi6.cerit-sc.cz
- 1.3 GHz, 64 cores
- 384 GB RAM (phi5-phi6), 192 GB RAM (phi1-phi4)
- available in queue phi@wagap-pro.cerit-sc.cz

cerit scientific cloud

HW comparison

Comparison of HW available in MetaCentre

Processor	SP/DP (GFLOPs)	Memory (GB/s)	Cache (L1, L2, L3, L4)
Xeon Phi 7210	5 324/2 662	450/102	64, 512 KB, 32 MB, 16 GB
Xeon E7-4830v4	896/448	85	64, 256 KB, 35 MB
Tesla K40	5 040/1 680	288	64 KB, 1.5 MB

cerit scientific cloud

Phi vs. GPU

Phi advantages

- no PCI-E bottleneck: fits for applications extensivelly using large data structures
- may execute applications not prepared for Phi (GPU has different programming model)
- easier migration of applications

GPU advantages

- higher raw performance
- older – thus more applications are prepared

cerit scientific cloud

Efficient applications

Scaling

- efficiently utilize 64/256 cores
- balanced parallel workload is crutial

Vectorized code

- using AVX512 brings much higher performance

cerit scientific cloud

Efficient applications

Memory-bound applications

- FFT, finite difference method, iterative solvers ...
- HBM bandwidth is comparable to 8-socket NUMA nodes
- programs not optimized for NUMA are more efficient on Phi

Compute-bound applications

- dense matrix factorization, n-body problem, data encoding ...
- exploit high arithmetic throughput of Xeon Phi comparable to 4-socket node (equipped by the best Xeons available)

cerit scientific cloud

Inefficient applications

Applications with scaling issues

- application must be able to run in sufficient number of threads (processes)
- workload must be distributed evenly across threads (more difficult with higher number of threads)
- more threads are more sensitive to suboptimal cache usage (false sharing, conflict miss)
- suffers from Amdahl law...

cerit scientific cloud

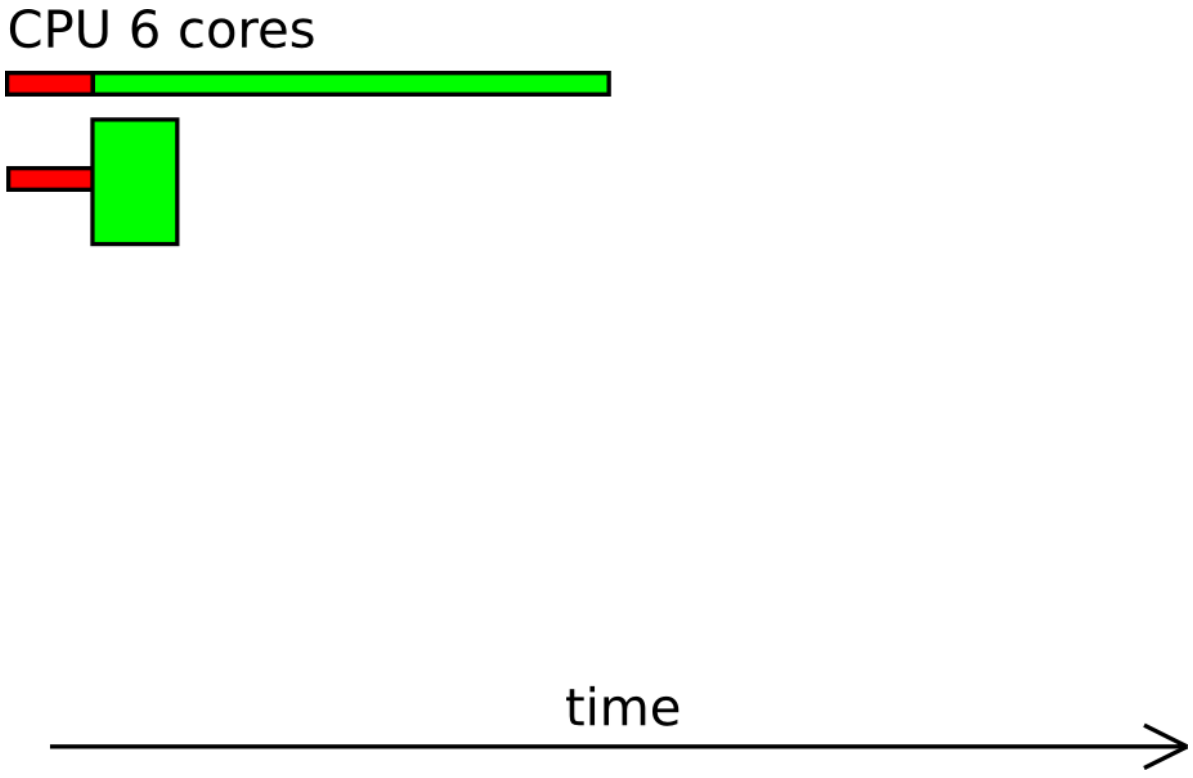
Amdahl Law and Phi

Our hypotetic application from computational chemistry

- 6/7 of workload: simulation steps (forces evaluation, integration): parallelized
- 1/7 of workload: preparing molecules, creation of data structures, results visualization: not parallelized
- suppose that Phi core is $2\times$ slower in this application

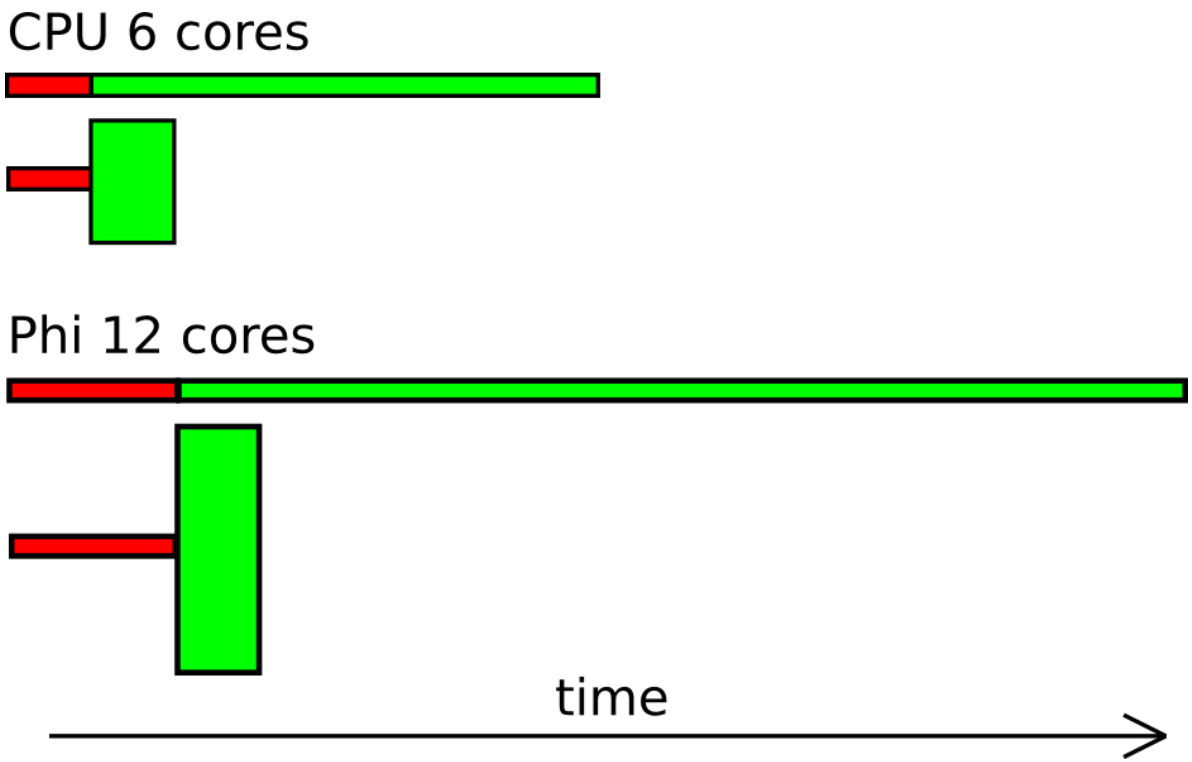
cerit scientific cloud

Amdahl Law and Phi



cerit scientific cloud

Amdahl Law and Phi



cerit scientific cloud

Inefficient applications

Poorly-vectorized applications

- Phi implements AVX512 (two instructions/clock per core): 16 DP or 32 SP operations in one cycle, doubled with FMA
- when not vectorized, code uses only small fraction of theoretical peak
- this is, however, also problem for new CPUs (ops per cycle doubled with Haswell, will double also with Cannonlake)
- non-vectorized code can typically run as fast as about 16 CPU cores

cerit scientific cloud

How can we help?

Short-term view

- we will optimize heavily used applications (conservativelly, by tuning compilation parameters, thread affinity etc.)

Long-term view

- CERIT-SC in-house research: how to program accelerators efficiently, methods for automatic code tuning
- optimization of application's codes (practical result of in-house research)
- possible cooperation on development with CERIT-SC partners

Application performance tuning

Application performance can be often improved by user-level tuning

- tuning compilation, execution of applications
- no source code modification

cerit scientific cloud

Compilation

Generation of AVX-512 instructions

- AVX-512 instructions are generated by automatic vectorization
- cannot help when manual vectorization is used (assembly, intrinsics)

Compiler flags

- Intel C $\geq$ 15.0: add flag -xMIC-AVX512
- gcc $\geq$ 4.9.1 supports AVX-512: add flag -avx512{f,er,cd,pf}

cerit scientific cloud

High-bandwidth Memory

Three possible configurations of HBM

- flat: HBM is extra NUMA node (with no associated cores)
- cache: HBM caches access into main memory
- hybrid: half flat, half cache, seems to be the most reasonable setting in MetaCentre

Execution of the whole application in HBM

- `numactl -m 1 ./binary`

cerit scientific cloud

Parallelism

Experiment with number of threads

- SMT-4 may help when instructions cannot occupy all ALUs
- however, more threads means more issues with load balancing, synchronization, cache locality
- test performance of your applications using 64, 128 or 256 threads

Physical and virtual cores

- physical core = coreID mod 64 (e.g. cores 0, 64, 128 and 192 are the same physical core)
- when subset of cores are used, physical cores should be loaded evenly

cerit scientific cloud

Parallelism

Mapping threads to cores

- OpenMP: `OMP_NUM_THREADS = x` is sufficient (tested on Intel OpenMP), `OMP_THREAD_LIMIT=256` may be needed to set maximal number of threads
- MPI: mpirun -np x -bind-to core:overload-allowed (tested on OpenMPI)
- can be checked by `htop`

cerit scientific cloud

Conclusion

Xeon Phi lies somewhere between GPU and CPU

- CPU code can be executed without modifications
- but Phi architecture is massively parallel
- performance peak comparable to 4-socket or 8-socket CPU node
- performance may be improved by quite simple tuning of application execution (number of threads, used memory)

We will optimize applications for Phi

- specialized modules will appear ...