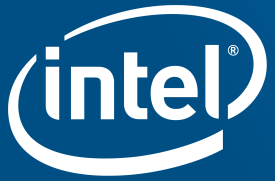


INTEL® XEON PHI™ PROCESSORS AND THEIR POTENTIAL FOR COMPUTATIONAL ATOMISTIC SIMULATION

Jeff Hammond

Intel Corporation

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © 2015, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

My Disclaimer

- You may or may not be able to reproduce any performance numbers I report.
- These are not official benchmarks in any way. They are meant to be illustrative of the performance you might see.
- All of my test codes are available on GitHub* for your evaluation.
- **Hanlon's Razor (blame stupidity, not malice).**
- I assume good faith when referencing third-party work.

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

Computational atomistic simulation

- Very complex software that lasts for decades.
- Require performance across tremendous range of problems.
  - For every 100K-core NWChem* CCSD(T) run with 20+ atoms, there are 100K single-node CFOUR* runs with <10 atoms.
  - …QBOX*…1000+…VASP*…<100…
  - …NAMD*…10M+…Gromacs*…<1M…
- Dense, irregular compute is common.
- Significant use of numerical libraries (BLAS/LAPACK/FFT).

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

KNL ARCHITECTURE

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

KNL architectural summary

- Intel Architecture: *binary compatible* with Intel® Xeon™ processors
- 64/68/72 cores per node, 4 threads/core
- 2-issue per core, 2-issue per thread (KNC was 1-issue per thread)
- BW (FP) max at 64+ (64+) threads
- 16 GB fast (MCDRAM) memory and up to 384 GB regular (DDR4) memory
- NUMA modes: quadrant ("1-socket"), SNC-2 ("2-socket"), SNC-4 ("4-socket")
- MCDRAM as user-managed memory or memory-side cache.
- AVX-512: 8-way 64b, 16-way 32b SIMD with FMA support.

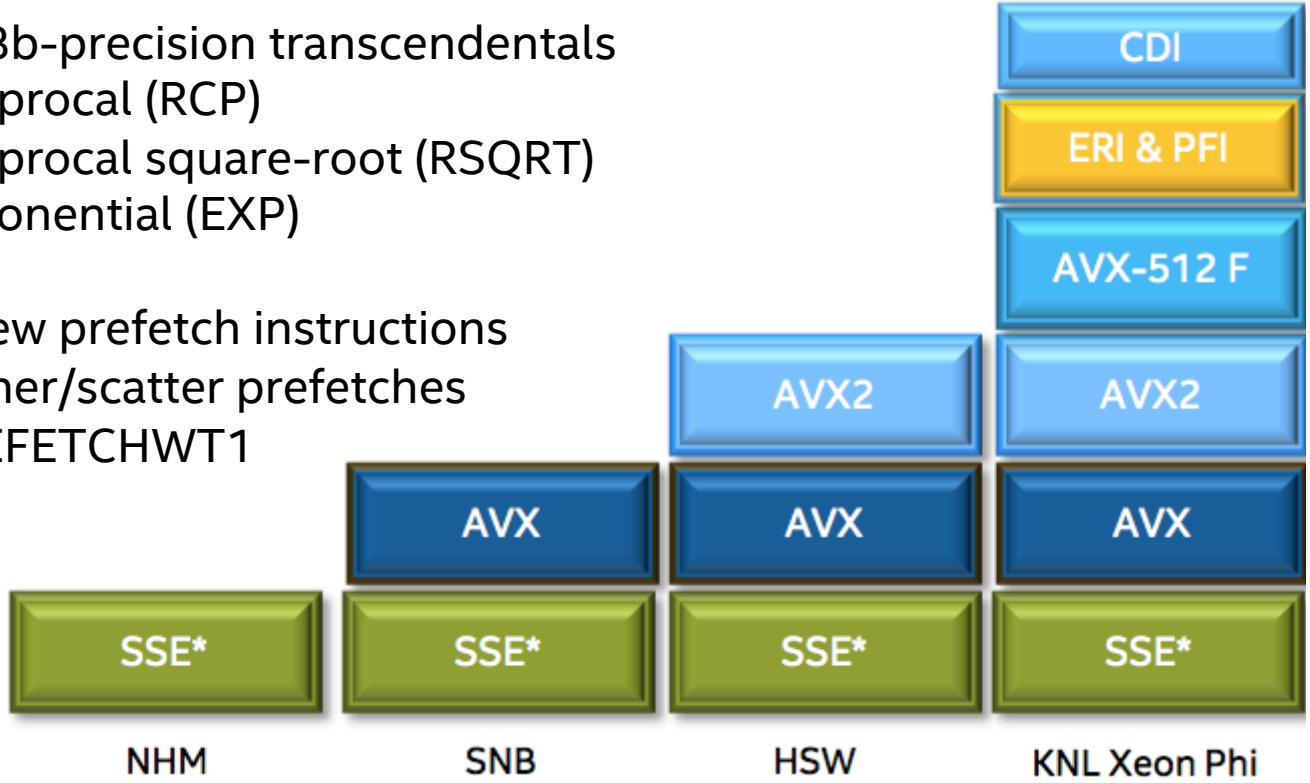
Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

https://wiki.alcf.anl.gov/parts/index.php/Knights_Landing

KNL instruction set features

ERI: 28b-precision transcendentals
- reciprocal (RCP)
- reciprocal square-root (RSQRT)
- exponential (EXP)

PFI: New prefetch instructions
- gather/scatter prefetches
- PREFETCHWT1



Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.
https://gcc.gnu.org/wiki/cauldron2014?action=AttachFile&do=get&target=Cauldron14_AVX-512_Vector_ISA_Kirill_Yukhin_20140711.pdf

APPLICATION SUPPORT

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

Atomistic codes that are being optimized for KNL

- LAMMPS – highly optimized for a range of *realistic* potentials.
- Gromacs, NAMD – AVX-512, etc. in-progress.
- CP2K – uses “perfectly fast” LIBXSMM library for small DGEMM.
- Quantum Espresso* and BerkeleyGW – Jack Deslippe knows more than I do…
- QBOX – MKL is tuned for KNL; improved ScaLAPACK* eigensolvers.
- NWChem…
  - KNC optimizations for CCSD(T) being re-tuned for KNL (sans offload).
  - KNL: explicit management of fast memory, more extensive threading in TCE.
  - Holistic OpenMP in PW-DFT and non-TCE perturbative triples.

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

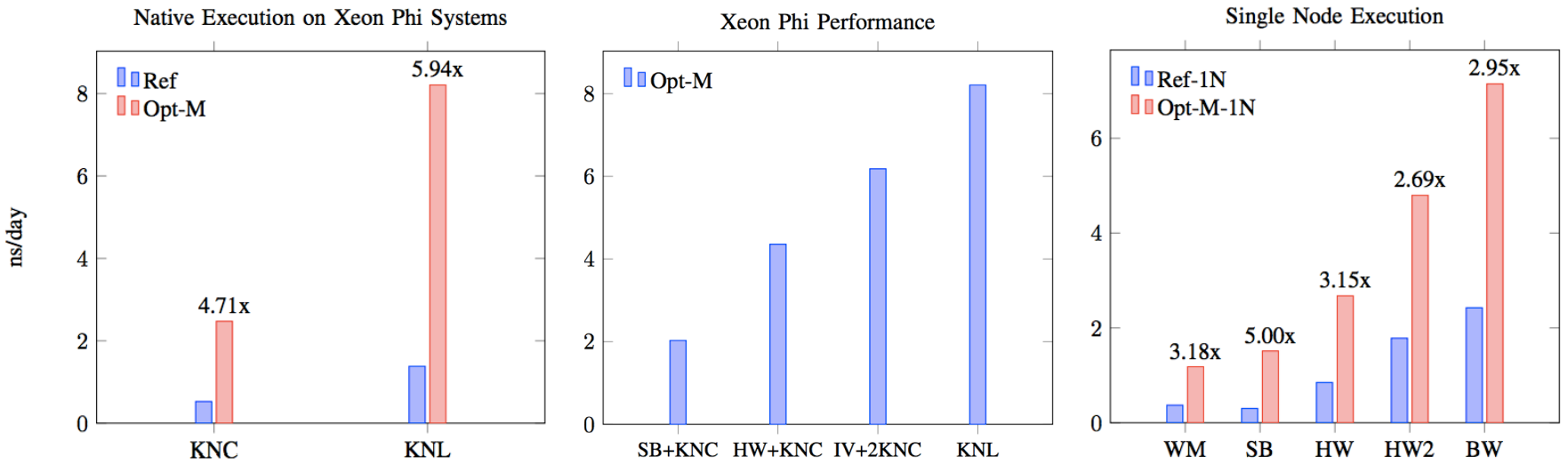
This is not intended to be a complete list…

The Vectorization of the Tersoff Multi-Body Potential: An Exercise in Performance Portability

Markus Höhnerbach
RWTH Aachen University

Ahmed E. Ismail
West Virginia University

Paolo Bientinesi
RWTH Aachen University



Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

http://arxiv.org/pdf/1607.02904v1.pdf

STREAM (BANDWIDTH)

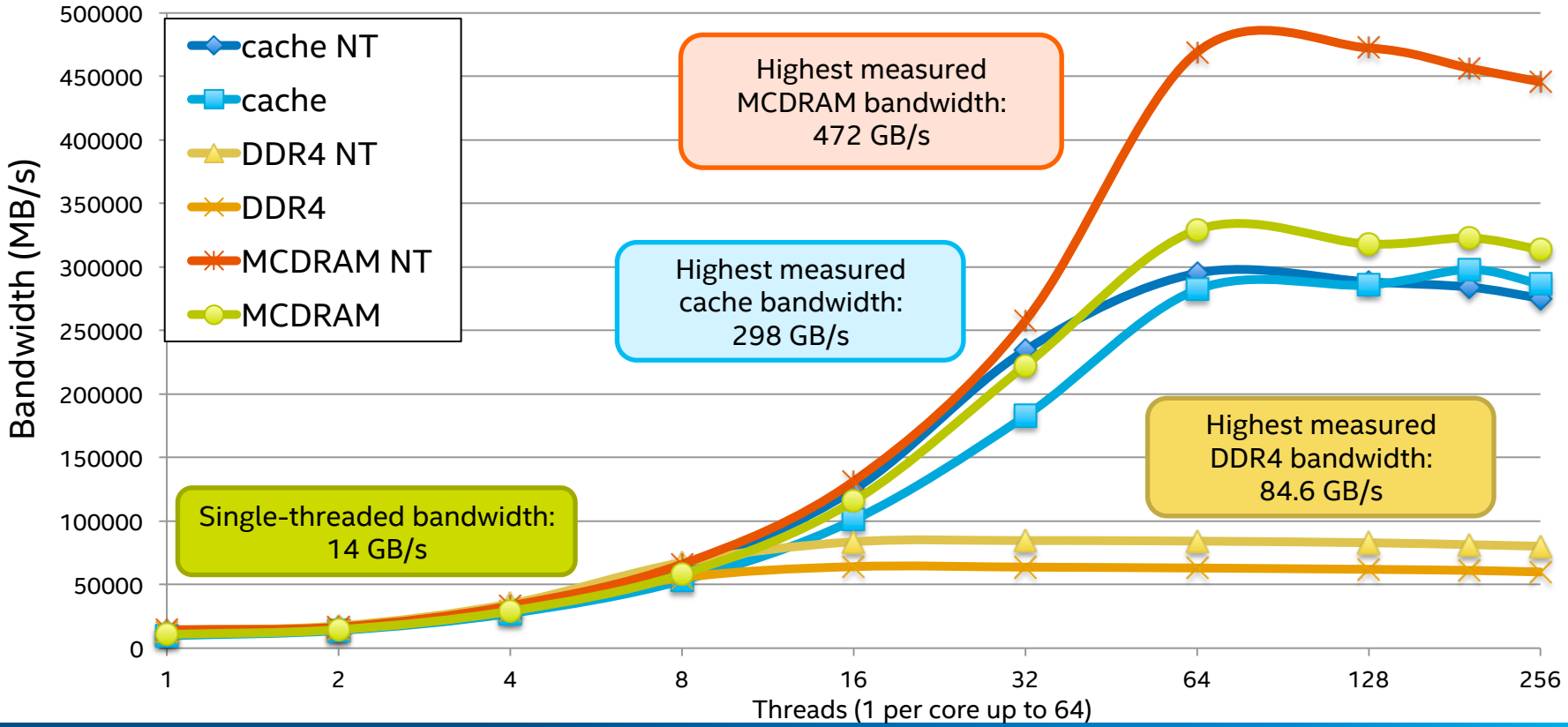
```
#pragma omp parallel for
for (ssize_t j=0; j<STREAM_ARRAY_SIZE; j++)
      a[j] = b[j]+scalar*c[j];
```

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

https://github.com/jeffhammond/STREAM/tree/knl

STREAM TRIAD

Intel® Xeon Phi™ CPU 7250 @ 1.40GHz
Linux* 3.10.0-327.el7.mpsp_1.3.0.66.x86_64

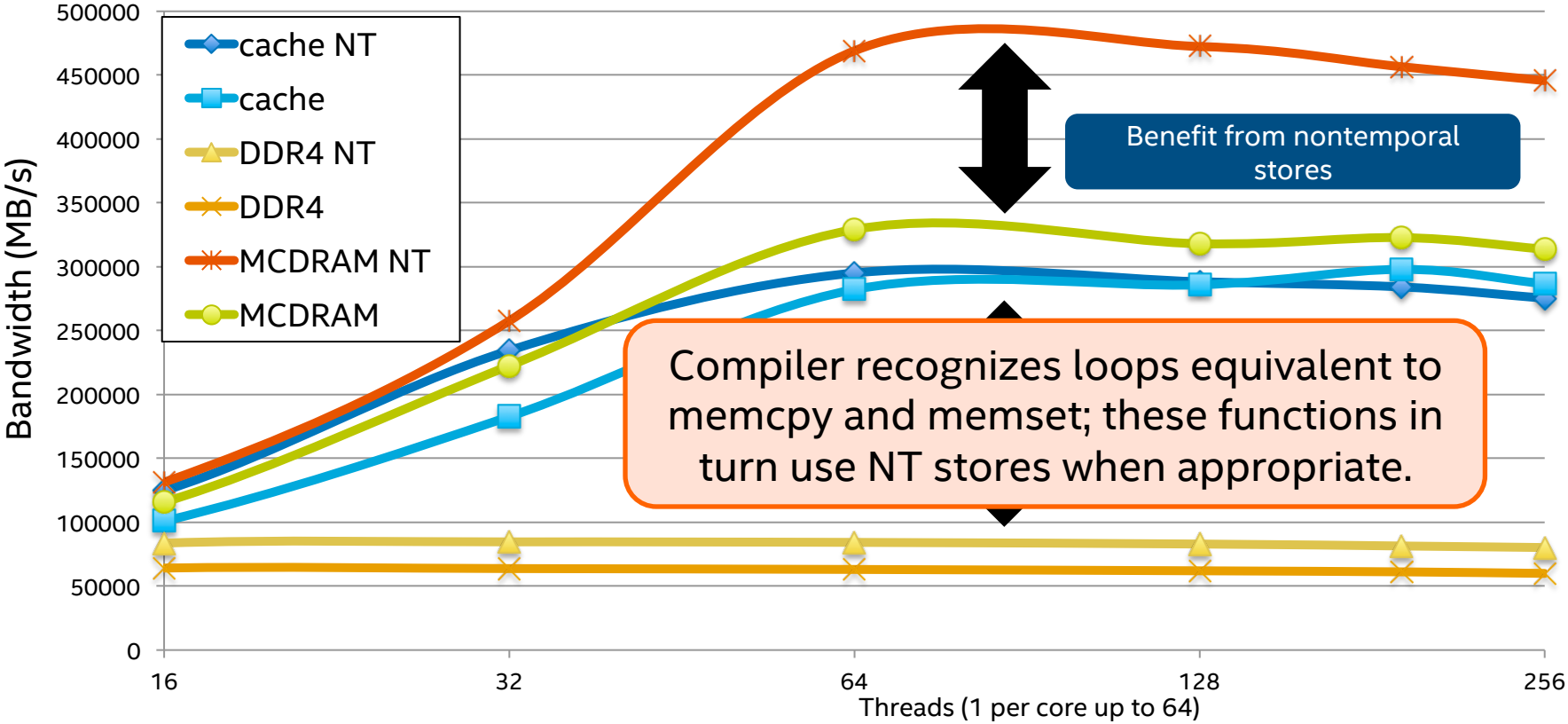


Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

https://github.com/jeffhammond/STREAM/tree/knl

STREAM TRIAD

Intel® Xeon Phi™ CPU 7250 @ 1.40GHz
Linux* 3.10.0-327.el7.mpsp_1.3.0.66.x86_64

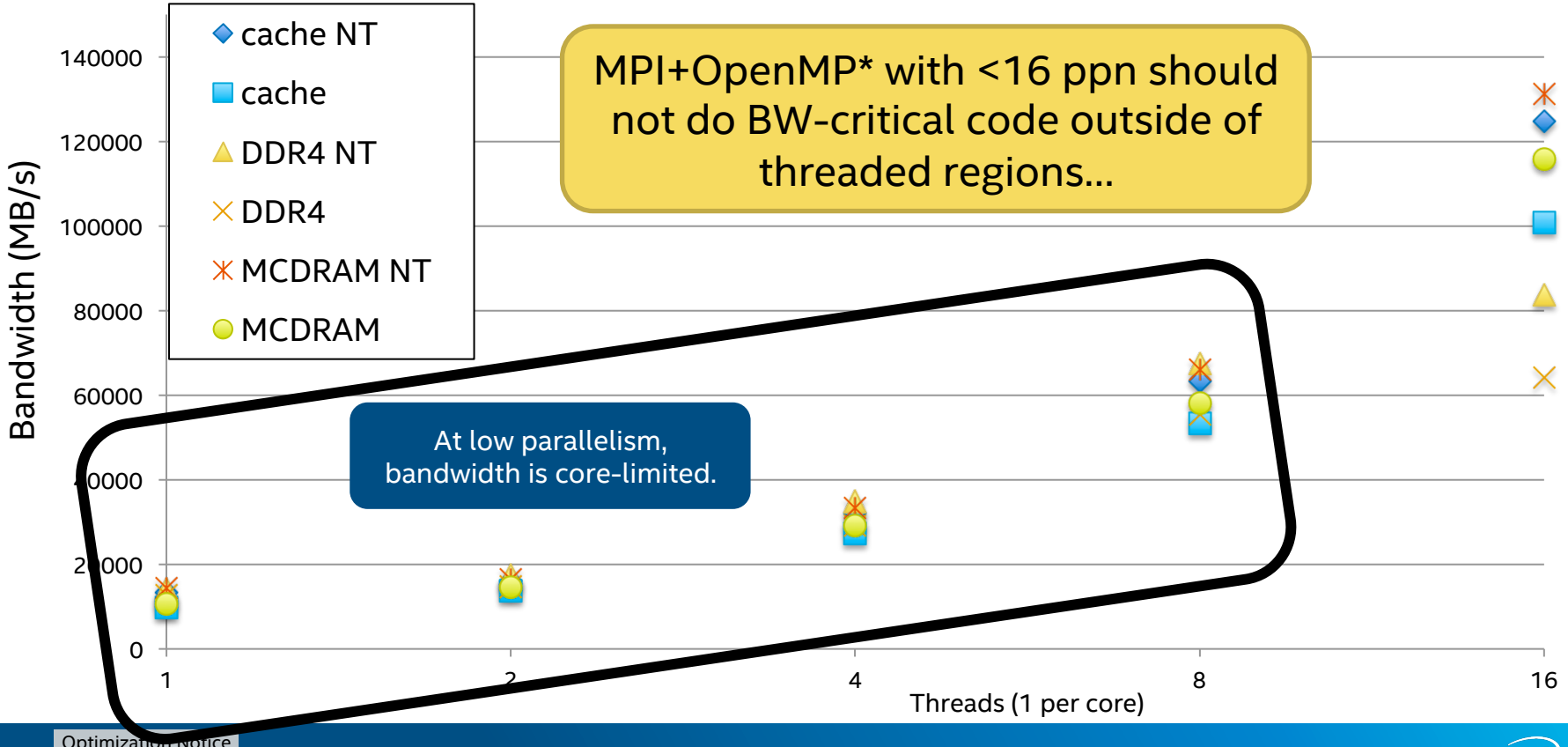


https://github.com/jeffhammond/STREAM/tree/knl

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

STREAM TRIAD

Intel® Xeon Phi™ CPU 7250 @ 1.40GHz
Linux* 3.10.0-327.el7.mpsp_1.3.0.66.x86_64



https://github.com/jeffhammond/STREAM/tree/knl

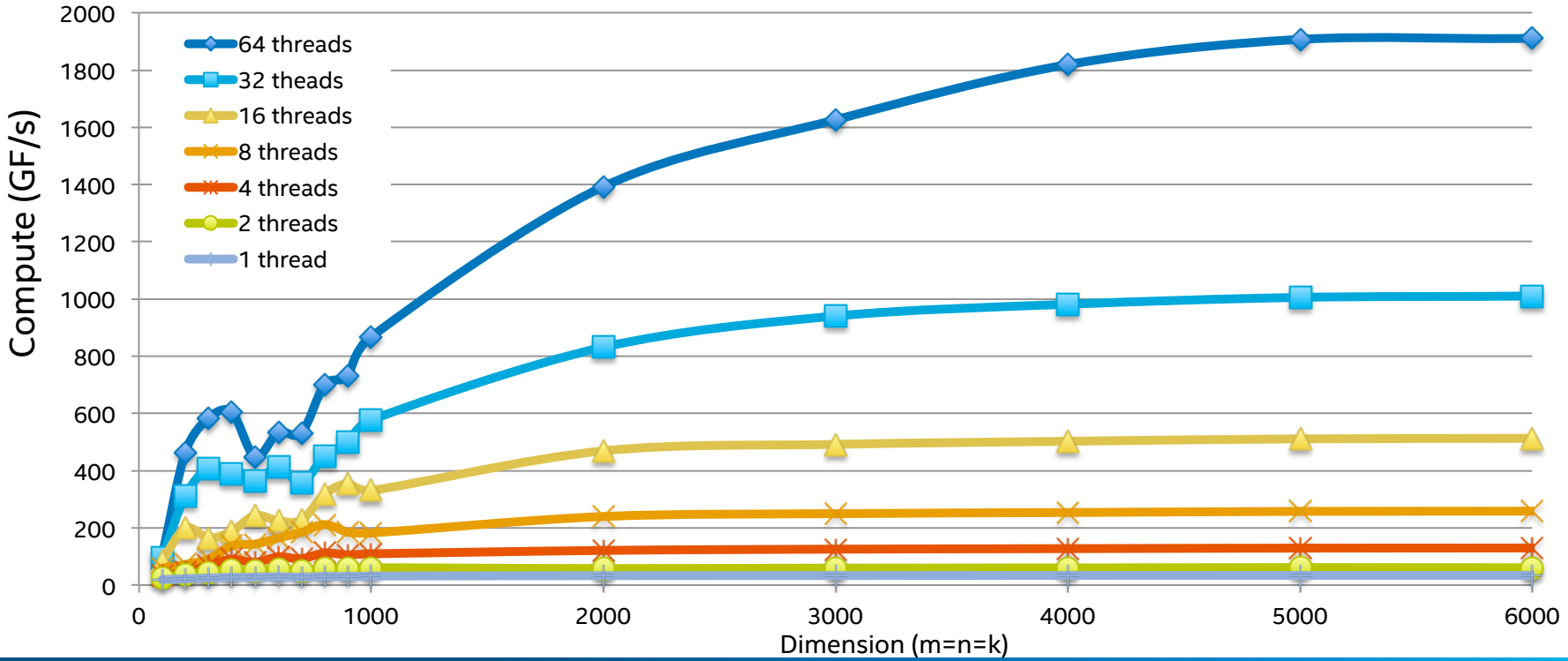
Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

DGEMM (COMPUTE)

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

MKL DGEMM

Intel® Xeon Phi™ CPU 7250 @ 1.40GHz
Linux* 3.10.0-327.el7.mpsp_1.3.0.66.x86_64



Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

https://github.com/jeffhammond/hpcinchemistrytutorial/tree/master/blas_performance

Optimizing small matrix multiplications

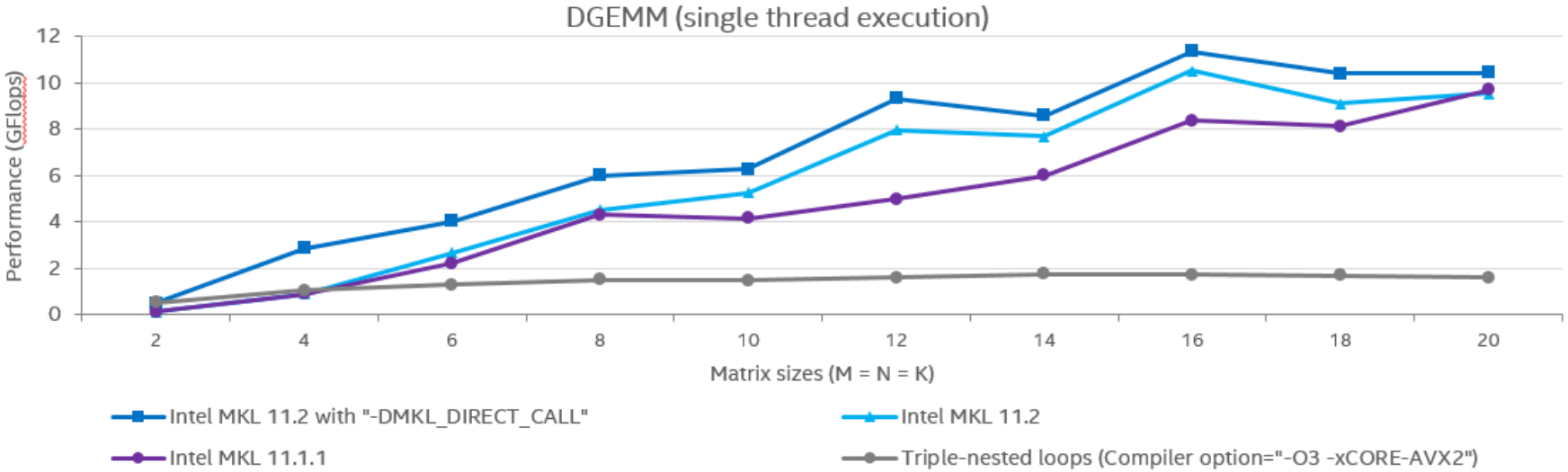
MKL_DIRECT_CALL addresses long-standing issues with software overhead in small GEMM…

1. Inlining code for very small problem (if parameters known at compile-time).
2. Call low-level library implementation directly for small problem sizes.
   - Skipping error checks and calls to intermediate library layers – skips optimizations for larger matrices.
   - Works for Intel and non-Intel compilers; C/C++ and Fortran*.

This is in addition to Intel MKL optimizations for smaller matrices when you call the normal symbols.

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

MKL_DIRECT_CALL on Haswell Xeon™



Configuration Info - Versions: Intel® Math Kernel Library (Intel® MKL) 11.1.1 and 11.2, Intel® C++ Compiler 15.0; Hardware: Intel® Xeon® Processor E5-2697v3, 2 Fourteen-core CPUs (35MB LLC, 2.6GHz), 64GB of RAM; Operating System: RHEL 6.1 GA x86_64; Benchmark Source: Intel Corporation.

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

https://software.intel.com/en-us/articles/improve-intel-mkl-performance-for-small-problems-the-use-of-mkl-direct-call

LIBXSMM

Authors
Hans Pabst (Intel)
Alex Heinecke (Intel)
Greg Henry (Intel)

- General-purpose code cannot be optimal:
  - If all cases are supported, the library is too large; too much branching into case.
  - Lack of specialization hurts when matrices are 1-10 SIMD units on a side.
- Runtime specialization captures best of both worlds:
  - “Perfect” code for precisely the cases you need.
  - Unused code is never generated.
  - Just-in-Time (JiT) compilation for general code is hard.
  - Matrix multiplication allows JiT without a compiler.

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

https://github.com/hfp/libxsmm

LIBXSMM

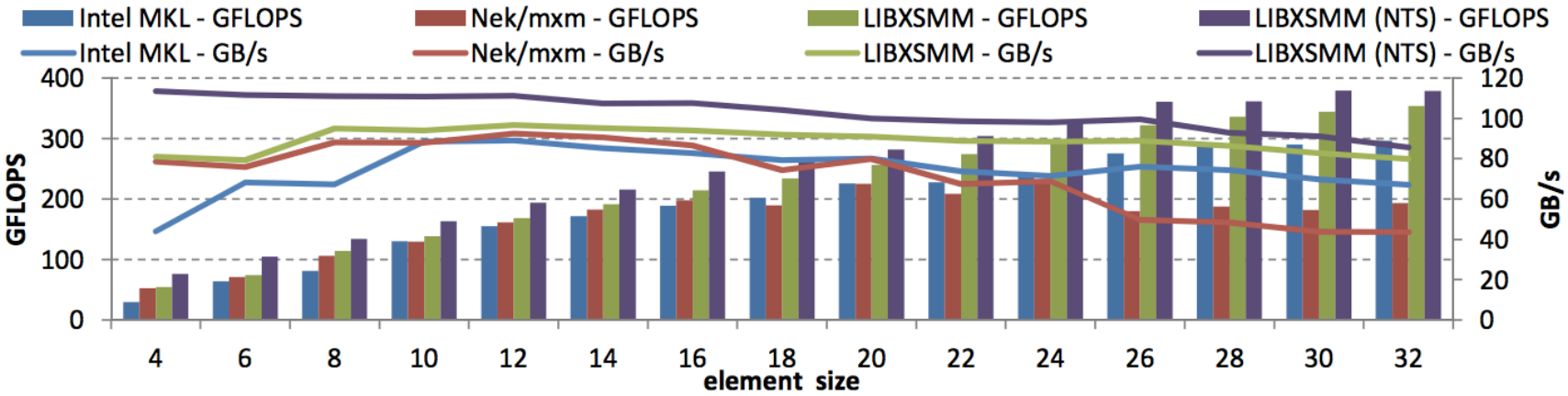


Fig. 1: Performance of the Helmholtz reproducer running on a single node of Shaheen for different implementation of small matrix multiplications. NTS denotes the usage of the non-temporal store optimized module.

Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

"Efficiency of High Order Spectral Element Methods on Petascale Architectures", M. Hutchinson, A. Heinecke, H. Pabst, G. Henry, M. Parsani,and D. Keyes; ISC High Performance 2016,

LIBXSMM references

- “LIBXSMM: Accelerating Small Matrix Multiplications by Runtime Code Generation.” SC'16: The International Conference for High Performance Computing, Networking, Storage and Analysis, Salt Lake City (Utah).
- “LIBXSMM: A High Performance Library for Small Matrix Multiplications.” (poster and two-page extended abstract). SC'15: The International Conference for High Performance Computing, Networking, Storage and Analysis, Austin (Texas).
- “Efficiency of High Order Spectral Element Methods on Petascale Architectures”, Maxwell Hutchinson, Alexander Heinecke, Hans Pabst, Greg Henry,Matteo Parsani,and David Keyes; ISC High Performance (ISC 2016).
- “High Order Seismic Simulations on the Intel Xeon Phi Processor (Knights Landing)”, Alexander Heinecke, Alexander Breuer, Michael Bader, and Pradeep Dubey; ISC High Performance (ISC 2016).

Optimization Notice

Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

PROGRAMMING MODELS

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

KNL optimization strategies

Strategy 1

- Use 16 GB memory.
  - mpirun … numactl -m 1 …
- Focus on scaling threading to at least 64/68/72 per node
- Move from fine-grain to course-grain OpenMP to amortize overheads.
- Consider using MPI-3 shared-memory for intranode.

Strategy 2

- Use same mix of MPI:OpenMP but increase MPI per node.
  - Cache mode for “free” MCDRAM.
  - Manage 2 memories with MEMKIND.
- Less critical to amortize OpenMP.
- Consider MPI-3 NBC or SHM to reduce MPI overheads as PPN grows.
- Intranode MPI bandwidth higher…

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

Threading support

- OpenMP* :
  - Intel 16 supports 4.0
  - Cray* 8.4 supports 4.0
  - GCC* 4.9.1+ supports 4.0, GCC 6.1 supports 4.5 for C/C++
  - Clang* 3.8 supports 3.1 with partial 4.5
- Intel TBB for C++ (now with Apache* 2.0 license)
- C++11 threads (mostly UNVERIFIED)
- User-level threads e.g. Sandia* Qthreads*, etc. (UNVERIFIED)

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

http://openmp.org/wp/openmp-compilers/
https://goparallel.sourceforge.net/intels-threading-building-blocks-turn-10-new-advances-follow/

Memory allocators

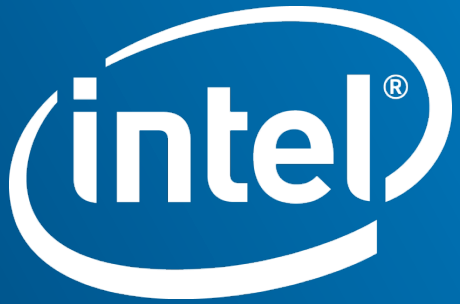
- MEMKIND is based on JEMALLOC*, which is very efficient multi-threaded allocator.
- TBBMALLOC is a very efficient single- or multi-threaded allocator.
- Other memory allocators (e.g. TCMALLOC*) should work fine but are not yet known to support MCDRAM.
- See https://github.com/jeffhammond/myhbwmalloc for an example of DLMALLOC* mspaces on top of libnuma for managing MCDRAM.
  - myhbwmalloc is only for educational purposes. **It is not supported.**
- KNL book has lots of info on MCDRAM programming models…

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.

Easy buttons

- KNL is binary-compatible with peer Xeon (Haswell and Broadwell).
  - Can build objects and binaries (fat or thin) on any Intel Linux* system.
  - Runs AVX2 code (well). Runs SSE, x87 code (YMMV).
  - You can reuse binary libraries from elsewhere.
- Python*, Java*, etc. just work. I do most of my configure+make on KNL.
- Thanks to ABI compatibility, can switch between MPICH and vendor MPI seamlessly.

Optimization Notice
Copyright © 2015, Intel Corporation. All rights reserved.
*Other names and brands may be claimed as the property of others.


intel®