Exploiting multi-level parallelism in HPC applications

Jeongnim Kim & TCG Micro
March, 2015

Node Configuration / Compilers / Runtime

Endeavor† cluster

- CPU: 2-socket/14 cores/56 threads
  - Processor: Intel® Xeon® processor E5-2697 V3 @ 2.60GHz (14 cores) with Intel® Hyper-Threading Technology[4]
  - Memory: 64GB
- Coprocessor: Intel® Xeon Phi™ coprocessor 7120P
  - 61 cores @ 1.238 GHz, 4-way Intel® Hyper-Threading Technology, Memory: 15872 MB
  - Intel® Many-core Platform Software Stack Version 3.3
- Network: InfiniBand* Architecture Fourteen Data Rate (FDR)
- Operating System: Red Hat Enterprise Linux* 2.6.32-358.el6.x86_64.crt1 #4 SMP Fri May 17 15:33:33 MDT 2013 x86_64 x86_64 x86_64 GNU/ Linux

Compilers

Intel® Parallel Studio XE 2015 Update 1 for Linux*

- C/C++/Fortran Compilers
- MKL library
- MPI library 5.0

Compiler options

Host: -xAVX -restrict -unroll -O3 –qopenmp

Coprocessor: -mmic -fp-model source -restrict -unroll -O3 –qopenmp

Link options

```
-L${MKLROOT}/lib/(intel64,mic) \
 -lmkl_cdft_core \
-lmkl_blacs_intelmpi_lp64 -lmkl_intel_lp64 \
-lmkl_intel_thread -lmkl_core -Wl,--end-group \
-lpthread -lm
```

† http://www.top500.org/system/176908

Outline

- Use case: quantum materials codes in a plane-wave basis
- Goals
- Performance analysis of core computations
  - GEMM
  - FFT
- Conclusions

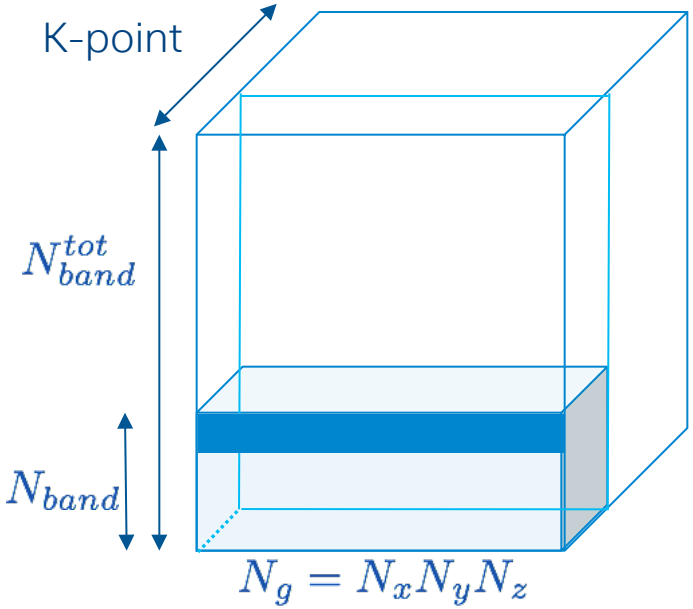
Generic Plane-wave Electronic Structure Code

Application abstractions
(QE,VASP, QBOX,…)

$$E = \sum_k w_k \sum_i \sum_j (\hat{S}^k)^{-1}_{i,j} (\Psi^k_j)^T \hat{H}_k \Psi^k_i$$

$$\hat{S}^k_{i,j} = (\Psi^k_i)^T \Psi^k_j$$

Data parallelization with MPI



Typical problems

- $N_{\mathbf{k}}$ (# of k points) 1-1000
- N^{tot}_{band} 100-10000
- N_g (FFT grid) > 10^3

Characteristics

- $N^{tot}_{band} \propto N_g$
- Less $N_{\mathbf{k}}$ for larger N^{tot}_{band}
- Complex or Complex-to-Real FFT
- Double precision

Solving bigger problems, faster, better

- A large fraction of NERSC resources is consumed by ES applications, e.g., BerkelyGW, NWCHEM, QE, VASP ….
  - Critical in materials, chemistry and physics research and computationally demanding.

How to enable existing ES applications to take advantage of Xeon and Xeon Phi™ platforms of today and tomorrow?

- Exploit multiple parallel opportunities through hybrid programming
- Develop adaptive parallel algorithms and implementations
- Enable code design for portable and performance portable applications

A plane-wave ES code

Each MPI task executes

```
void main_mpi()
{
  int nband_tot=512;
  int nband=nband_tot/mpi_tasks;
  PWBand  psi[nband];


  for(int i=0; i<nband; ++i)
  {
    fft(psi[i]);
    compute_g(psi[i],psi);
    ifft(psi[i]);
    compute_r(psi[i],psi);
  }
  do_mpi(psi);//reduction
}
```

A typical computation

```
void compute_g(PWBand& me, PWBand* all)
{
  const int i= me.id;
  const int mband=me.siblings;
  Matrix overlap(nband,mband);


  for(int j=0; j<mband; ++j)
    overlap(i,j)=dot(me,all[j]);


  for(int x=0; x<nions; ++x)
    do_more(me,x);//gemm,gemv,…
}
```

PWBand: abstraction of a single-particle orbital (SPO)

- Identity and its relations to the set to which this SPO belongs

A plane-wave materials code : threading opportunities

Each MPI task executes

```
void main_mpi()
{
  int nband_tot=512;
  int nband=nband_tot/mpi_tasks;
  PWBand  psi[nband];

  #pragma omp parallel for
  for(int i=0; i<nband; ++i)
  {
    fft(psi[i]);
    compute_g(psi[i],psi);
    ifft(psi[i]);
    compute_r(psi[i],psi);
  }
  do_mpi(psi);//reduction
}
```

A typical computation

```
void compute_g(PWBand& me, PWBand* all)
{
  const int i= me.id;
  const int mband=me.siblings;
  Matrix overlap(nband,mband);

  #pragma omp parallel for
  for(int j=0; j<mband; ++j)
    overlap(i,j)=dot(me,all[j]);

  #pragma omp parallel for
  for(int x=0; x<nions; ++x)
    do_more(me,x);//gemm,gemv,…

}
```

- The loop over Bands: possible to minimize synchronization (reduction/flush) overhead
- Computations contain parallelizable loops
- FFT can utilize threaded implementation

How to increase the performance

Minimize the time-to-solution at the sustained throughputs (# of simulations/day)

- Exploit shared memory on Xeon and Xeon Phi™ to minimize memory footprint and MPI communication
  - > 20% spent on MPI communication on modest scale problems
  - Some methods, e.g., hybrid functionals, GW0, RPA, are limited by the memory available per MPI task due to replications
- Exploit optimized libraries and OpenMP* runtime
  - Threaded FFT
  - Threaded GEMM and per band operation

Going to analyze the performance of MKL libraries to predict the performance when OpenMP* is enabled in PW materials codes.

Predicting performance with nested OpenMP

```
void main(){

  int nband_tot=512;//arbitrary
  int nband=nband_tot/mpi_tasks;
  PWBand  psi[nband];

  double res=0.0;
  #pragma omp parallel reduction(+:res)
  { double res_t=0.0;

    for(int t=0; t<Iter; ++t)
    #pragma omp for reduction(+:res_t)
      for(int i=0; i<nband; ++i)
      res_t+=compute(psi[i]);

    res+=res_t;
  }

  mpi_allreduce(res);
}
```

Concurrent computation of

- DGEMM
- 3D FFT: forward-and-backward C2C FFT

Run-time variables: MPI, OMP, MKL

- N_{pp}=MPI*OMP*MKL=(# of cores)*HT
- On KNC: 60, 120, 180 or 240
- On HSW CPU: 28 or 56

Measured quantities

- DGEMM : total GFLOPs/sec
- 3D FFT: total # of FFTs/sec

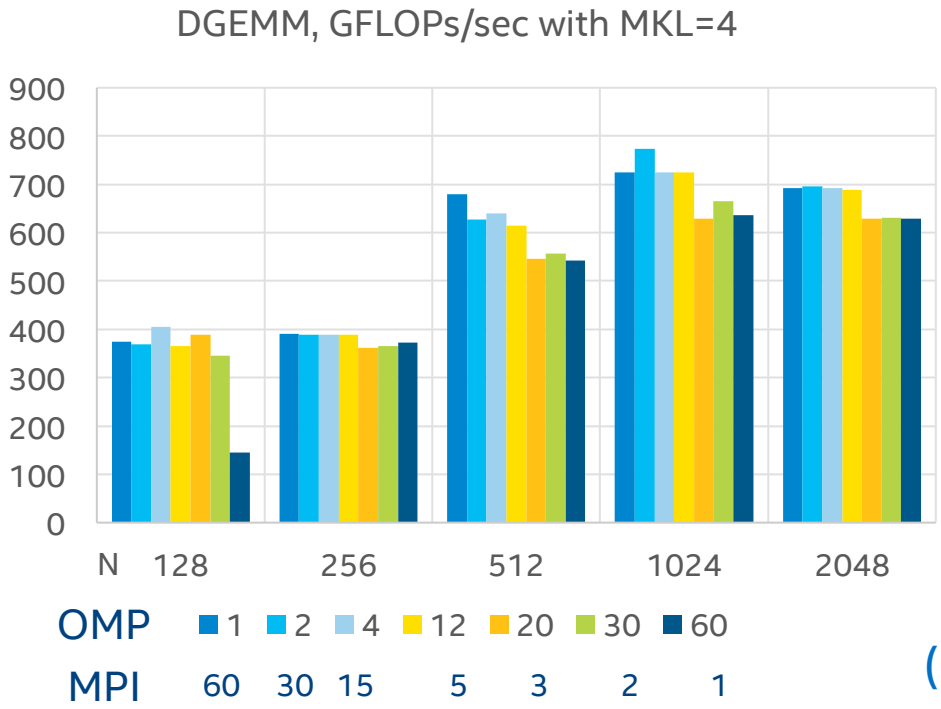
More to consider but not extensively discussed here

- Cluster FFT
  - Some PW codes adopt parallel FFT over MPI tasks
    - Becomes less critical with increasing capacity of SMP node : 1 node in 2000 ~ 1 core in 2014
    - Will compare cluster FFT (CFFT) and threaded FFT (FFT3D)
- Effective use of SIMD throughout applications
- Blas I/II efficiency
  - MKL overhead can be higher than necessary and relying on the compilers' ability to vectorize can be more effective
- Language-specific performance and code modernization issues
- MPI3 shared memory optimization and non-blocking collectives

DGEMM

Throughputs of dgemm A(N,N)=B(N,N)*C(N,N)

60 concurrent dgemms on KNC



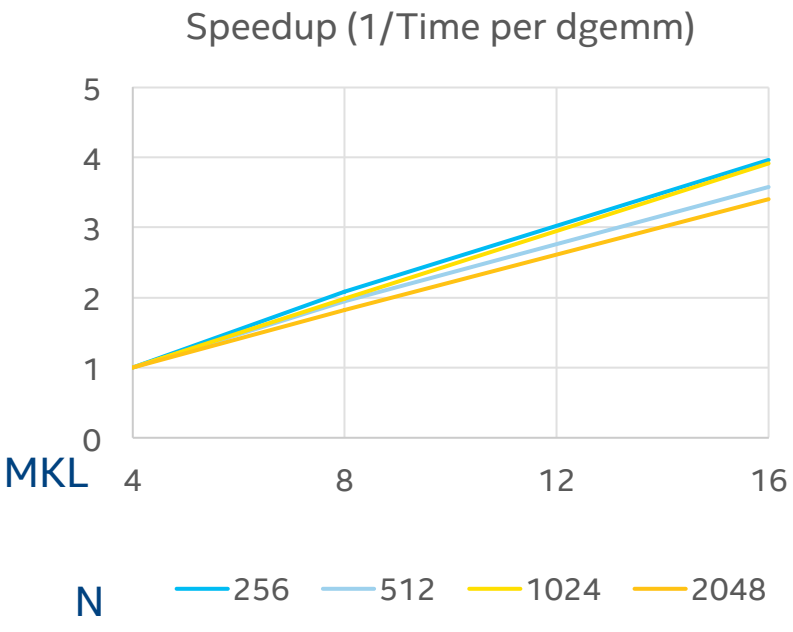
Composer 15 1.133
OpenMP hot teams are enabled

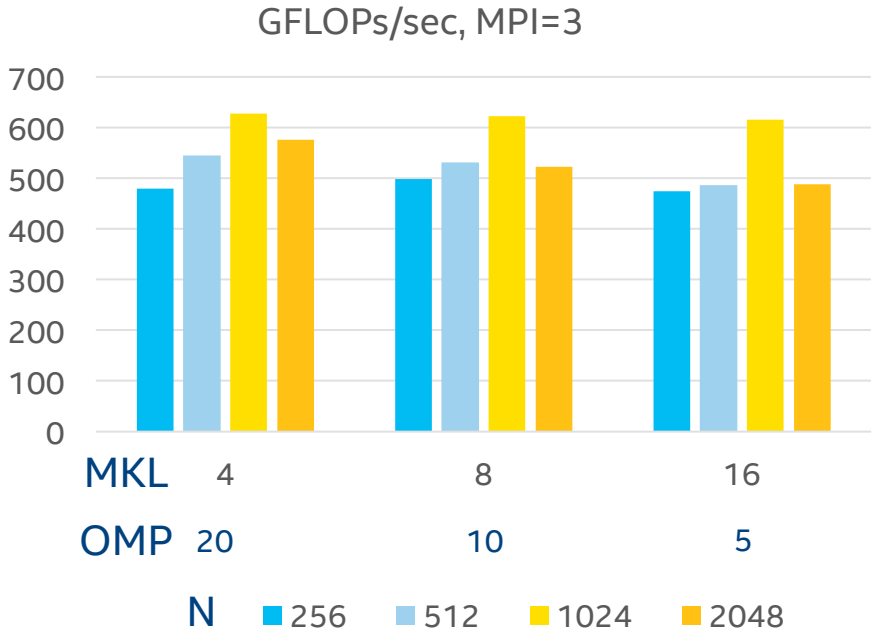
Environments

```
OMP_NESTED=true
MKL_DYNAMIC=false
MKL_NUM_THREADS=4

OMP_NUM_THREADS=$OMP,4
OMP_PROC_BIND=spread,close

KMP_HOT_TEAMS_MAX_LEVEL=2
KMP_HOT_TEAMS_MODE=1
KMP_BLOCKTIME=infinite
```

(MPI,OMP,MKL=4) and MPI*OMP=60

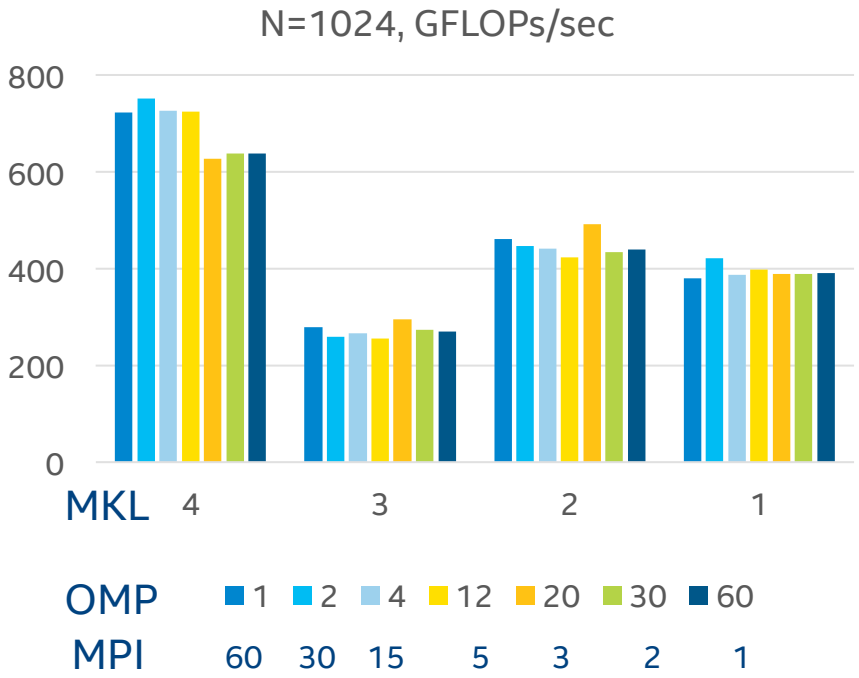
Speedup and throughputs vs MKL threads
MPI_TASKS=3

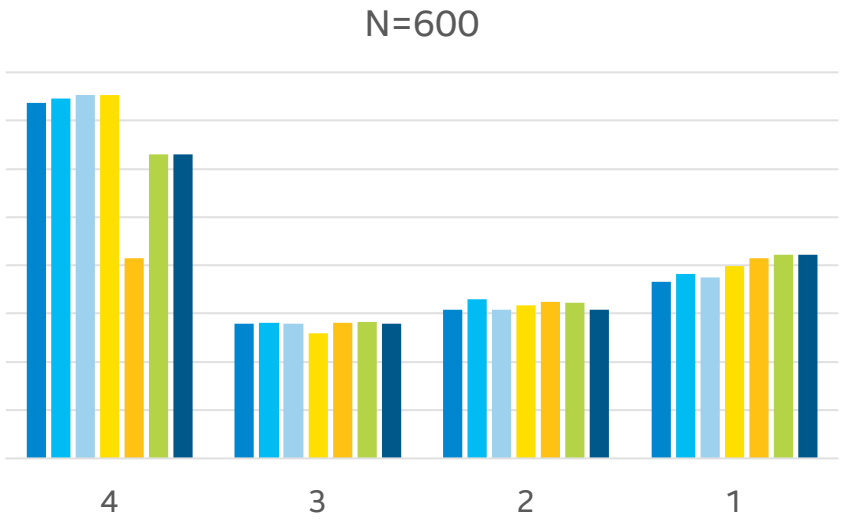




- Free performance gain through threaded DGEMM: Super-scaling for certain problem sizes

Multiple Hardware Threads on Xeon Phi™



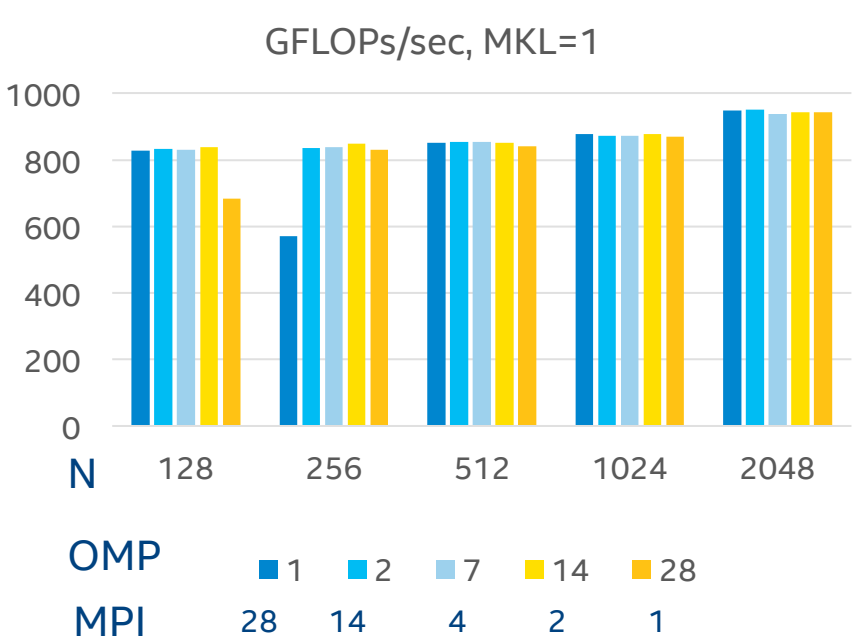


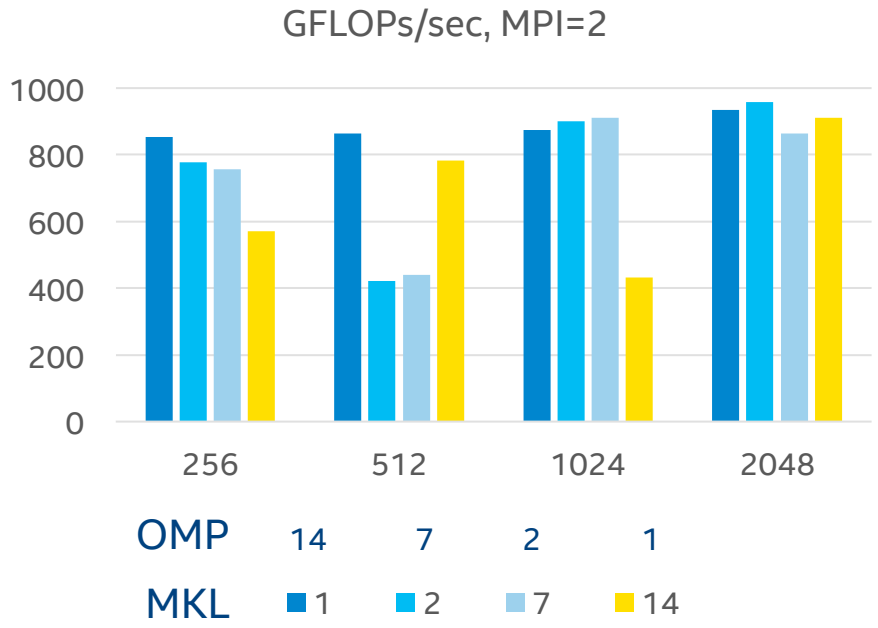
- Using 4 HT per core boost the performance for a wide range of N

Intel Measured Results

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. For more information go to http://www.intel.com/performance.

Performance on Haswell 2.6 GHz





Close to 90% of the peak over these Ns

Parallel efficiency subject to N

Intel Measured Results

FFT3D

FFT3D as used in the Plane-wave materials codes

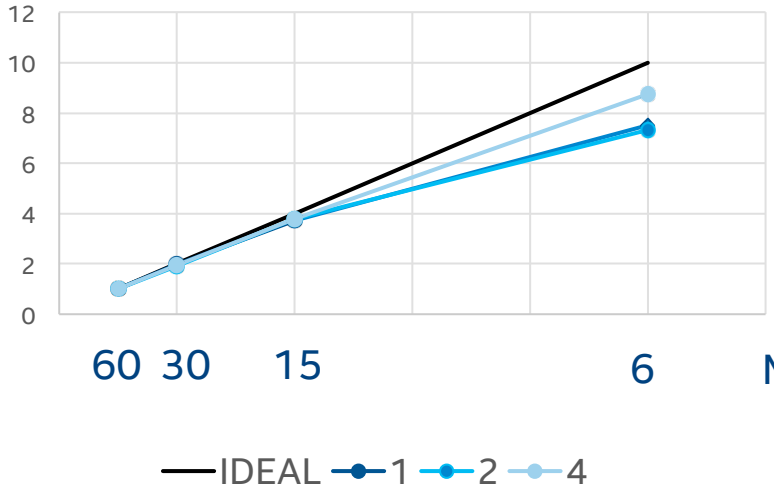
- Each FFT3D handled by the threaded FFT library in MKL using MKL_NUM_THREADS= N_{MKL}

```
void compute_fft(PWBand& me)
{
    fft_forward(me); //threaded FFT
    fft_backward(me);//threaded FFT
}
```

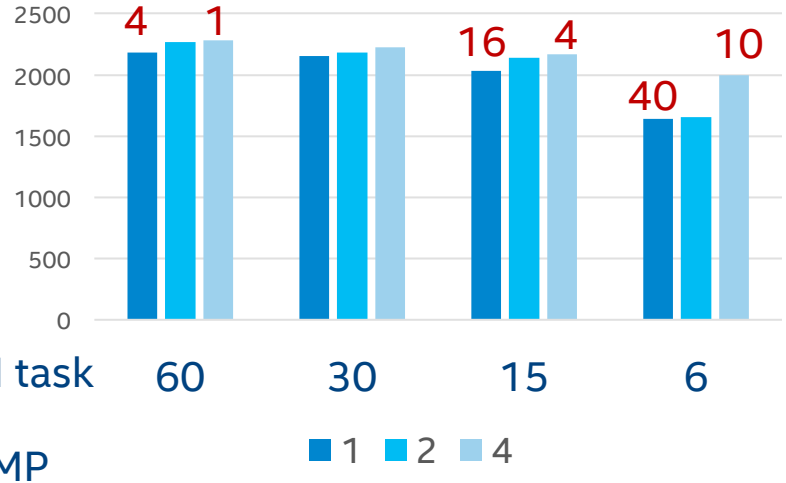
- A FFT plan (DFTI_DESCRIPTOR_HANDLE) per OMP thread
  - FFT plans are created/destroyed outside the main compute loop and excluded from the measurements
  - Unlike FFTW, MKL FFT
    - Overhead of the plan creation is very low
    - Memory per plan is low
  - Possible to create the plan to perform multiple FFT simultaneously
- Quantities of interest
  - Time per FFT through increasing parallelism in FFT:
  - Throughputs (# of FFT pairs per sec) by choosing MPI & OMP for the memory requirement

FFT3D on KNC, Ng=64^3

Parallel Efficiency (1/Time per FFT)



Throughputs (# of FFTs/sec)



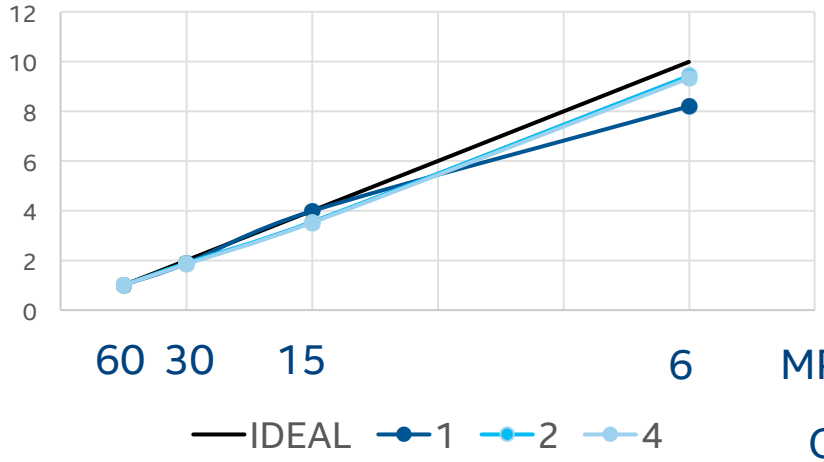
OMP : number of threads of the OMP level

$$N_{MKL} = 240/(N_{MPI} * OMP)$$

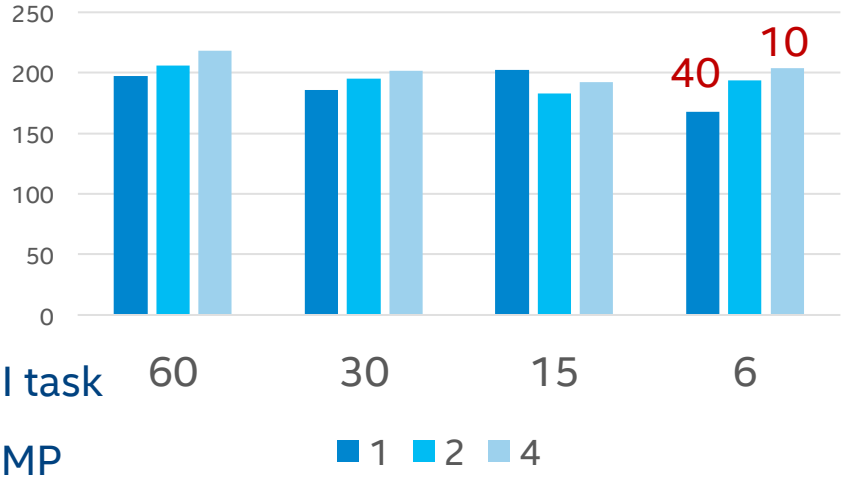
Intel Measured Results

FFT3D on KNC, Ng=128^3

Parallel Efficiency (1/Time per FFT)



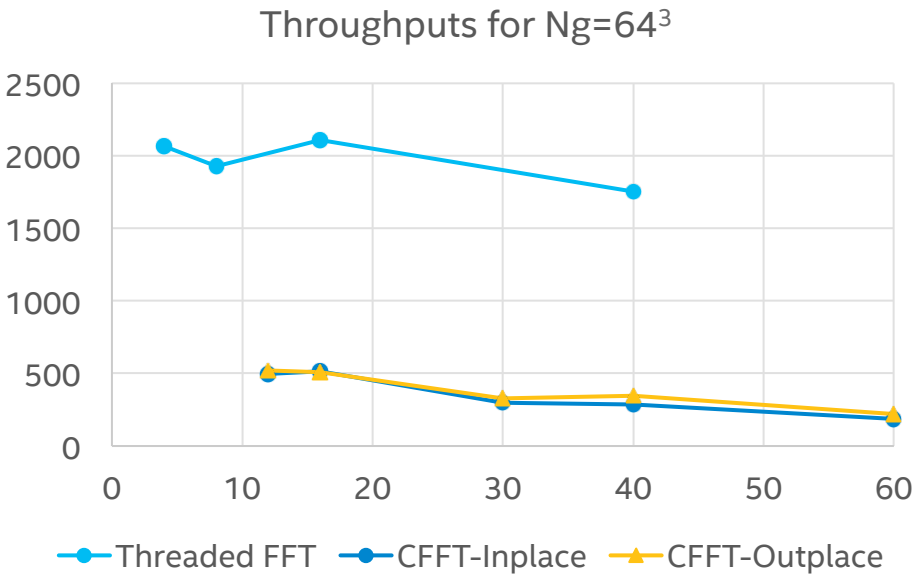
Throughputs (# of FFTs/sec)



OMP : number of threads of the OMP level

$$N_{MKL} = 240/(N_{MPI} * OMP)$$

Comparison of FFT3D and CFFT on KNC



MPI_TASKS or
MKL_NUM_THREADS

- FFT3D (Threaded FFT) 4x better than CFFT (similar story on Xeon)
- Best with 16 threads per FFT with FFT3D

3D FFT on KNC

- Threaded FFT is much more efficient than cluster FFT.
- Sustained performance (throughputs) for a wide range of (MPI, OMP), e.g.,

```
export OMP_NUM_THREADS=5,16
export MKL_NUM_THREADS=16
mpirun –np 3 fft3d …
```

≈

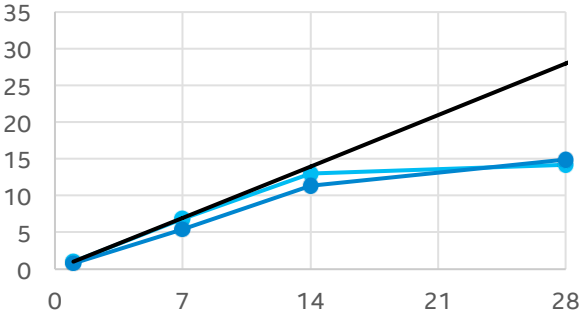
```
export OMP_NUM_THREADS=3,16
export MKL_NUM_THREADS=16
mpirun –np 5 fft3d …
```

- Time per FFT decreases with increasing N_{MKL}: 85% parallel efficiency from 4 to 40 threads for Ng=64^3; better with larger Ng
- 4 HT is most efficient for Ng=64^3 or larger.
- DFTI_NUMBER_OF_TRANSFORMS=1 is best.

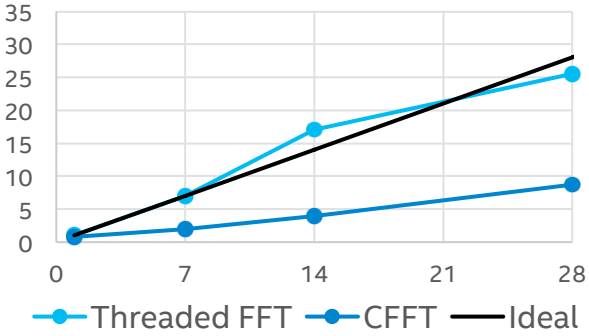
Comparison between FFT3D & CFFT on HW

Speedup wrt a serial FFT

64^3

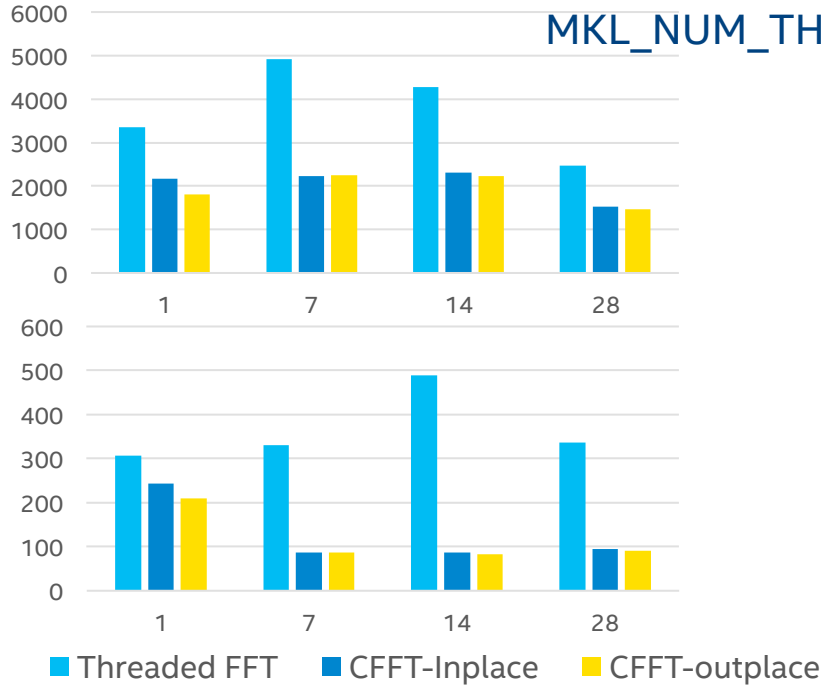


128^3

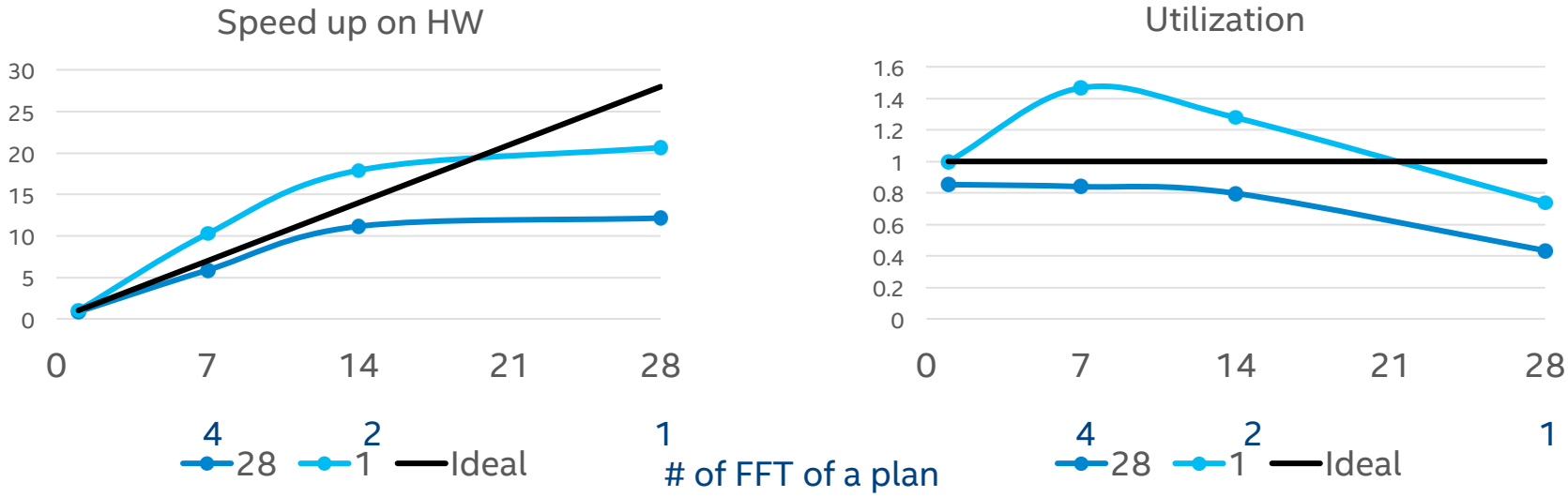


Throughputs

X-axis: MPI Tasks or MKL_NUM_THREADS



FFT3D Ng=64^3 on Haswell



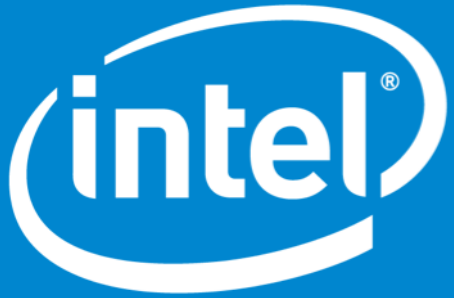
- Similar to KNC with more performance penalty with OMP!=1
- Big difference with the number of FFT (DFTI_NUMBER_OF_TRANSFORMS)
  - DFT_NUMBER_OF_TRANSFORMS=1 best
- MKL=7 for the throughput; MKL=14 for the time-to-solution

Intel Measured Results

Implications to real applications

- Flexibility in choosing data decomposition with MPI and OpenMP parallelizations within a MPI task
- Nested OpenMP can further improve the time-to-solution at the same throughput (# of DGEMMs and FFTs per sec); higher **total** performance & resource utilization
- Big gain in memory with less MPI tasks: very critical to high-level methods such as GW
- Memory management using more specialized allocators and page sizes for the problem size can improve the performance
  - These data are obtained with using `__mm_malloc` with the alignment hint
- Many ways to overlap computations & communications and different computations with minimal code changes
- Work for Xeon and Xeon Phi™

But, it is important to start with good data partitions and parallel algorithms!


intel

Legal Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.
Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to:
http://www.intel.com/design/literature.htm

Knights Landing and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user

Intel, Look Inside, Xeon, Intel Xeon Phi, Pentium, Cilk, VTune and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Legal Disclaimers

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 SSE3 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

Legal Disclaimers

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. For more information go to http://www.intel.com/performance.

Estimated Results Benchmark Disclaimer:
Results have been estimated based on internal Intel analysis and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

Software Source Code Disclaimer:
Any software source code reprinted in this document is furnished under a software license and may only be used or copied in accordance with the terms of that license.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Legal Disclaimers

The above statements and any others in this document that refer to plans and expectations for the third quarter, the year and the future are forward-looking statements that involve a number of risks and uncertainties. Words such as “anticipates,” “expects,” “intends,” “plans,” “believes,” “seeks,” “estimates,” “may,” “will,” “should” and their variations identify forward-looking statements. Statements that refer to or are based on projections, uncertain events or assumptions also identify forward-looking statements. Many factors could affect Intel’s actual results, and variances from Intel’s current expectations regarding such factors could cause actual results to differ materially from those expressed in these forward-looking statements. Intel presently considers the following to be the important factors that could cause actual results to differ materially from the company’s expectations. Demand could be different from Intel's expectations due to factors including changes in business and economic conditions; customer acceptance of Intel’s and competitors’ products; supply constraints and other disruptions affecting customers; changes in customer order patterns including order cancellations; and changes in the level of inventory at customers. Uncertainty in global economic and financial conditions poses a risk that consumers and businesses may defer purchases in response to negative financial events, which could negatively affect product demand and other related matters. Intel operates in intensely competitive industries that are characterized by a high percentage of costs that are fixed or difficult to reduce in the short term and product demand that is highly variable and difficult to forecast. Revenue and the gross margin percentage are affected by the timing of Intel product introductions and the demand for and market acceptance of Intel's products; actions taken by Intel's competitors, including product offerings and introductions, marketing programs and pricing pressures and Intel’s response to such actions; and Intel’s ability to respond quickly to technological developments and to incorporate new features into its products. The gross margin percentage could vary significantly from expectations based on capacity utilization; variations in inventory valuation, including variations related to the timing of qualifying products for sale; changes in revenue levels; segment product mix; the timing and execution of the manufacturing ramp and associated costs; start-up costs; excess or obsolete inventory; changes in unit costs; defects or disruptions in the supply of materials or resources; product manufacturing quality/yields; and impairments of long-lived assets, including manufacturing, assembly/test and intangible assets. Intel's results could be affected by adverse economic, social, political and physical/infrastructure conditions in countries where Intel, its customers or its suppliers operate, including military conflict and other security risks, natural disasters, infrastructure disruptions, health concerns and fluctuations in currency exchange rates. Expenses, particularly certain marketing and compensation expenses, as well as restructuring and asset impairment charges, vary depending on the level of demand for Intel's products and the level of revenue and profits. Intel’s results could be affected by the timing of closing of acquisitions and divestitures. Intel's results could be affected by adverse effects associated with product defects and errata (deviations from published specifications), and by litigation or regulatory matters involving intellectual property, stockholder, consumer, antitrust, disclosure and other issues, such as the litigation and regulatory matters described in Intel's SEC reports. An unfavorable ruling could include monetary damages or an injunction prohibiting Intel from manufacturing or selling one or more products, precluding particular business practices, impacting Intel’s ability to design its products, or requiring other remedies such as compulsory licensing of intellectual property. A detailed discussion of these and other factors that could affect Intel’s results is included in Intel’s SEC filings, including the company’s most recent reports on Form 10-Q, Form 10-K and earnings release.

Rev. 7/17/13