Weather, Climate, and Earth System Modeling with Intel® Xeon Phi™ Processors:

Practice, Experience, and Preparing for Future Generations

Richard Tran Mills

September 16-17, 2015

Multi-Core 5 Workshop, Boulder, CO
Notice and 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.

All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

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

Intel, Intel Xeon, Intel Xeon Phi™ are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries.

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

Copyright © 2015 Intel Corporation. All rights reserved.
## 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
This presentation includes contributions from several Intel colleagues:

- Sunny Gogar, Mike Brown, Jeff Hammond, Ruchira Sasanka, Karthik Raman, Mike Greenfield

Thanks is also due to

- The NCAR ASAP team, particularly John Dennis, Ben Jamroz (now at DigitalGlobe), and Chris Kerr
Growing Intel® Parallel Computing Centers Community

Collaborating to accelerate the pace of discovery

Updated: 08/18/15
Software Barrier for Modern HPC

The breakdown in Dennard scaling has led to significant changes in HPC computer architectures that achieve power-efficient performance.

Modifying legacy software for these systems can be a significant barrier to performance

• Example: Still only a subset of HPC codes can efficiently utilize multicore server processors with GPGPU accelerators at scale

The Intel roadmap seeks to address both challenges with x86-based many-core coprocessors and bootable processors that achieve performance with standard programming models and similar optimization approaches to those used on Intel® Xeon® Processors.

2013 Knights Corner
Intel® Xeon Phi™ x100 product family
• 22 nm process
• Coprocessor
• Over 1 TF DP Peak
• Up to 61 Cores
• Up to 16GB GDDR5

2016 Knights Landing ("KNL")
The processor version of the next generation Intel Xeon Phi product family
• 14 nm process
• Processor & Coprocessor
• Over 3 TF DP Peak
• Up to 72 Cores
• On Package High-Bandwidth Memory
• 3x single-thread performance
• Out-of-order core
• Integrated Intel® Omni-Path

FUTURE
Knights Hill ("KNH")
Next generation of Intel® MIC Architecture
Product Line
• 10 nm process
• 2nd Generation Integrated Intel® Omni-Path
• In planning –
Trend: More parallelism, deeper hierarchies

Intel® Xeon Phi™ processors amplify importance of fine-grained parallelism, but this direction holds for machines based on “conventional” CPUs as well:

• More cores/threads in socket and across machines (on per-node basis, core counts becoming roughly equivalent)

• More data parallelism/fine-grained parallelism AVX512 (512 bit vectors w/ FMA) coming in both Intel® Xeon Phi™ and Xeon® processors.

• More NUMA domains/level of storage hierarchy (DRAM NUMA domains, MCDRAM, NVRAM, IO subsystem)
There are no differences!

Because both use the same programming models and optimization approaches, divergent source code is greatly reduced.
Reducing Divergent Source Code

Acceleration with GPGPU can require different algorithms forcing an expanded source code base and further complicating optimization and validation for HPC systems

• Example: GPGPU acceleration for MD uses different algorithms and potentially different MPI* communications to efficiently handle the large number of lightweight threads in flight.

<table>
<thead>
<tr>
<th></th>
<th>Intel® Xeon® Processors</th>
<th>GPGPU</th>
<th>Intel® Xeon Phi™ Coprocessors</th>
<th>Intel® Xeon Phi™ Bootable Processors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Electrical Power</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Efficiency for HPC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Source Code Modification for Hybrid Computers</td>
<td>✓</td>
<td>✓</td>
<td>Not Necessary</td>
<td>✓</td>
</tr>
<tr>
<td>Software Modification for non-x86 Architectures</td>
<td>✓</td>
<td>✓</td>
<td>Not Necessary</td>
<td>✓</td>
</tr>
</tbody>
</table>
The Software Challenge in Modern HPC

Example (courtesy of Mike Brown, Intel): Optimizations for GPGPU acceleration in LAMMPS* did not improve performance on CPUs

- Many production codes supporting GPU acceleration still use the CPU for many routines in order to achieve performance and must still support CPU-only

Optimizations for x86 coprocessors also improved CPU performance, but ...

- Optimization still required
- What are the optimizations? →

Intel Results: Intel Measured August 2014

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. See benchmark tests and configurations in the speaker notes. For more information go to http://www.intel.com/performance

Intel Results: Intel Measured August 2014

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. See benchmark tests and configurations in the speaker notes. For more information go to http://www.intel.com/performance
Optimizing for Many-Core Processors

- The general optimization guidelines for Intel® Xeon® and Intel® Xeon Phi™ processors remain the same for codes that have been running on large-scale clusters and supercomputers for some time.

- However, the increase in parallelism required for power efficiency with many-core can amplify bottlenecks due to:
  - Unvectorized code
    - Structure loops to facilitate compiler vectorization; use vectorization directives when compiler can’t prove vectorization is safe
  - Inefficient data layout for vectorization and/or many cores sharing the memory subsystem
    - Modify data layout for efficient (re)use of cache lines, software prefetching for random access, correctly aligned data, efficient loading of vector registers, and use hyper-threads as appropriate
  - Inefficient synchronization
    - Avoid collective synchronization where possible
  - Lack of overlap of internode communications and computation
    - Use non-blocking MPI* calls, task-based parallelism, etc. to minimize idle time
  - Suboptimal MPI task mapping of subdomains to physical cores
    - Map MPI tasks to minimize the surface to volume ratio of domains mapped to nodes or NUMA nodes, not just individual MPI tasks
Intel® Xeon Phi™ (Knights Landing) Processors

Compute
- 36 tiles with 2D mesh interconnect
- Tile: 2 OoO cores + 2 VPU/core + 1MB L2
- Intel® Xeon® Processor Binary-Compatible
  (No need to rebuild NetCDF libraries, etc.)
- 3+ TFLOPS¹, 3X ST² (single-thread) perf. vs KNC

On-Package Memory
- Up to 16 GB at launch
- Over 5x STREAM³ vs. DDR4 at launch

Platform Memory
Up to 384 GB DDR4

Up to 72 Cores

Knights Landing

Integrated Fabric

Processor Package

I/O
Up to 36 PCIe 3.0 lanes

>50 System Providers Expected⁴

¹Over 3 Teraflops of peak theoretical double-precision performance is preliminary and based on current expectations of cores, clock frequency and floating point operations per cycle. FLOPS = cores x clock frequency x floating-point operations per second per cycle.

²Projected peak theoretical single-thread performance relative to 1st Generation Intel® Xeon Phi™ Coprocessor 7120P (formerly codenamed Knights Corner).

³Projected result based on internal Intel analysis of STREAM benchmark using a Knights Landing processor with 16GB of ultra high-bandwidth versus DDR4 memory only with all channels populated.

⁴Intel internal estimate

*Other names and brands may be claimed as the property of others. Products, dates, and figures may be preliminary and are subject to change without any notice. Copyright © 2015, Intel Corporation.
KNL/KNH Leadership Class, Open Science Systems

NERSC Cori

• ~ 1400 dual socket nodes w/ Intel® Xeon® v3 (“Haswell”) Processors, 16 cores per socket
• Over 9,300 single socket nodes w/ 2nd gen Intel Xeon Phi Processors (“Knights Landing”—KNL), w/ up to 16GB on-package, high-bandwidth memory
• Cray Aries dragonfly topology interconnect

ALCF Aurora

• Over 50,000 nodes with 3rd gen Intel® Xeon Phi™ Processors
• Over 8 PB aggregate on-package high-bandwidth memory and persistent memory
• 2nd gen Intel® Omni-Path Architecture with silicon photonics
• Intel® Lustre* filesystem, > 1 TB/s throughput
Exposing Concurrency on Many Levels

On prior leadership-class machines, near-exclusive focus on flat MPI optimizations was often sufficient.

Machines like Cori and Aurora require attention to many levels of concurrency:

1. Within single threads (vectorization)
   - Strategies similar to those on AVX2 CPUs apply.

2. Across shared memory on a node (usually OpenMP threads)
   - Minimizing thread serialization is critical w/ so many cores.

3. And across nodes on the interconnect (usually MPI)
   - NEW! MPI-3 introduces several features that can help.
   - At scale, may ultimately need to re-think algorithms to reduce global communications. (E.g., McInnes et al. 2014, Hierarchical Krylov and Nested Krylov Methods for Extreme-Scale Computing, http://www.mcs.anl.gov/papers/P2097-0612.pdf)

Important: Must consider #2 with #3! MPI communications a big source of thread serialization.
Progress on HOMME AVX/AVX2 optimizations

The NCAR ASAP team and Intel have been working to improve performance on AVX2-enabled CPUs, as a proxy for KNL. (See John Dennis's talk.)

Major optimizations by SVN revision:

• r4607, 3/4: Reordering of arrays in element_t data structure; replacement of 2D messaging buffer with 1D one

• r4718, 6/11: Numerous optimizations and restructuring in Eulerian advection routines

• r4718+CK mods, 8/3: Numerous optimizations in flux limiter

Performance of baroCamMoist test (NE=6) on Intel® Xeon® Processor E5-2697v3, 2.6 GHz (dual socket, 18 cores per socket). Performance trends with AVX512 on KNL are similar (cannot yet share these data).
Vectorization Advisor combines static and runtime analysis to facilitate development of vectorizable code.

Summary: Why loop was not vectorized

Summary: How was Loop Vectorized

Vectorization and Compiler optimization details

Pointing to relevant Source Code
Deeper Analysis – Loop wise Dependency and Memory Access Pattern Check

Select Loops for Deeper Analysis. Then click the appropriate workflow button to run Check Dependencies and/or Check Memory Access Patterns Collector.
Thread serialization in MPI+threaded code

What we like to think happens in a multi-threaded MPI code
Thread serialization in MPI+threaded code

Serialization either by user for MPI calls (MPI_THREAD_FUNNELED) or by MPI when touching shared state (MPI_THREAD_MULTIPLE)

What *actually* happens in a multi-threaded MPI code
Removing thread-serial data movement in HOMME

Ensuring that non-communicating threads do useful work during MPI communications is critical to avoid excessive serialization.

Example: Great improvement in HOMME boundary exchange after Ben Jamroz (NCAR ASAP) threaded data copies by the non-communicating threads

Original algorithm

```bash
!$OMP BARRIER
!$OMP MASTER
MPI_Isend
MPI_Irecv
MPI_Waitall x 2
Data Movement
(move received data to send buffer)
!$OMP END MASTER
!$OMP BARRIER
```

New algorithm

```bash
!$OMP BARRIER
!$OMP MASTER
MPI_Isend
MPI_Irecv
MPI_Waitall x 2
!$OMP END MASTER
!$OMP DO SCHEDULE( dynamic , chunksize =4)
Data Movement
(move non-sent data to recv buffer)
!$OMP IMPLICIT BARRIER
```
Leveraging MPI-3 features for manycore MPPs

• MPI-3 introduces several features well-suited to manycore MPPs. Two especially useful features (supported in Intel MPI 5.x):
  • Shared memory windows
  • Neighborhood collectives
  • Can be used in conjunction with OpenMP, or used to develop “MPI+MPI” hybrid shared/distributed memory MPI-only applications.
Graph Topologies and Neighborhood Collectives

MPI-3 introduces distributed graph topologies to allow expression (in a scalable way) of any communication pattern to the runtime.

(Boundary element exchange as $N$ Isend-Irecv + Waitall is perhaps the most common messaging pattern)

Neighborhood collectives perform communications specified on graph topologies. Knowing pattern in advance enables several optimizations, e.g.,

- Persistent allocation of network resources
- Intelligent scheduling (accounting for factors like **transport over shared memory vs. off-node network**)

While simplifying code by expressing communications with a single call.
On-package High-Bandwidth Memory

- Optimizations discussed so far apply to both Intel® Xeon® and Xeon Phi™.
- KNL introduces an additional feature: On-package high bandwidth memory (up to 16 GB), or multi-channel DRAM (MCDRAM)
- Configured in one of three modes at boot time:

  **Cache Mode**
  - SW-Transparent, Mem-side cache
  - Direct mapped, 64B lines
  - Tags part of line
  - Covers whole DDR range

  **Flat Mode**
  - MCDRAM as regular memory
  - SW-Managed
  - Same address space

  **Hybrid Mode**
  - Part cache, Part memory
  - 25% or 50% cache
  - Benefits of both
Using user-addressable high-bandwidth memory

In flat mode, MCDRAM is exposed as a separate NUMA node: (libnuma, mmap() work just as for any NUMA node.)

- If footprint fits, can place entire application in MCDRAM using numacli(8).
- Can use AutoHBW interposer library to do automatic size threshold-based placement in MCDRAM.
- Can do explicit placement:
  - **Fortran:** `!DEC$ ATTRIBUTES, FASTMEM :: A`
  - **C:** `fv = (float *)hbw_malloc(sizeof(float) * 100),`
    or use underlying memkind ([https://github.com/memkind](https://github.com/memkind)) library:
    `a = (float *)memkind_malloc(MEMKIND_HBW_PREFERRED, size);`
Memory Bandwidth Analysis with VTune

• User-addressable MCDRAM offers great performance potential, but introduces a (possibly nontrivial) packing problem.

• Intel® VTune™ Amplifier XE 2016 provides analysis capabilities that can help:
  • Memory bandwidth analysis to determine phases of your application that are memory bandwidth-limited
  • Instrumentation of memory allocation/de-allocation and mapping of hardware events to memory objects
Collecting the memory utilization data

Configuration options:

- **Analyze memory objects**: enables the instrumentation of memory allocation/de-allocation and mapping of hardware events to memory objects
  - May cause additional runtime overhead due to the instrumentation of all system memory allocation/de-allocation API

- **Minimal memory object size to track, in bytes**: Specify a minimal size of memory allocations to analyze. This option helps reduce runtime overhead of the instrumentation

Command line usage:

```
amplxe-cl -c memory-access -knob analyze-mem-objects=true -knob mem-object-size-min-thres=1024 -- <app>
```

Example: Profiling 1 Rank out of 60 MPI Ranks for HOMME

```
mpirun -n 59 ./baroCam < baroTemp.nl : -n 1 amplxe-cl –collect memory-access –knob analyze-mem-objects=true –knob mem-object-size-min-thres=1024 -data-limit=0 -r camMoist_HSW_50R_5day_BW_2 -- ./baroCam < baroTemp.nl
```
Analyze Bandwidth Utilization over Time
Identify Code Sections and Memory Objects Consuming Significant Bandwidth.

Use **Bandwidth Utilization Histogram** to see how much time the system bandwidth was utilized by a certain value (per Bandwidth Domain)

- Use sliders at the bottom to categorize bandwidth utilization as High, Medium, and Low
Identify Code Sections and Memory Objects Consuming Significant Bandwidth.

Group by bandwidth categories in the Grid view:

- Allows to easily see, e.g., all functions executing when the system DRAM bandwidth utilization was high.
  - Sort the grid by **LLC Miss Count** to see what functions contributed most to the high DRAM bandwidth utilization.
The “parallel_mod.F90:142 (701 MB)” means memory objects allocated on line 142 of source file parallel_mod.F90. The allocation size is 701 MB.
Summary and Conclusions

• There is a growing body of weather/climate/Earth system modeling work using Intel® Xeon Phi™ processors.

• Applications exposing high intranode and SIMD concurrency can realize high performance on Intel® Xeon Phi™ processors. Furthermore,
  • Optimizations for Intel® Xeon Phi™ processors will generally improve performance on any modern Intel® processors, and vice versa.
  • Trends towards more cores, more SIMD parallelism mean that future CPUs will likely show even more benefit—Intel® Xeon Phi™ processors amplify all of the things that you must do today to be ready for compute platforms of tomorrow.

• On-package high bandwidth memory in 2nd generation Intel® Xeon Phi™ processors will provide an extra boost to memory bandwidth-intensive applications (typical in weather/climate/ESM space).