



# Programming and Optimization for Intel<sup>®</sup> Architecture

The Hands-On Workshop (HOW) Series

Andrey Vladimirov, PhD, and Ryo Asai  
Colfax International — [@colfaxintl](#)

March 2016 , Rev. 02a

# About This Document

This document represents the materials of a Web-based training “Programming and Optimization with Intel Architecture” developed and run by Colfax International.

© Colfax International, 2013-2015

## Parallel Programming Boot Camp (1-Day) / Workshop (4-Days)



Instructor-led 1-day or 4-days training, at your office or at Colfax facility in Sunnyvale, CA

[Click here to learn more](#)

### 1-Day Parallel Programming Boot Camp

For software engineers and architects, providing an overview of parallel programming frameworks and optimization guidelines for multi-core CPUs (Intel® Xeon®) and many-core coprocessors (Intel® Xeon Phi™):

- Discussions about three layers of parallelism: SIMD, Threads, Cluster environment
- Tips for quick porting/development of HPC software applications
- Real-life examples of code and optimization techniques
- Hardware solution and corresponding software implementations, APIs, and frameworks

### 4-Days Parallel Programming Workshop

For the developer who wants to hit the ground running with the modern multi-core CPUs (Intel® Xeon®), many-core coprocessors (Intel® Xeon Phi™) and leading software development tools:

- Hardware installation
- MPSS tools and the Linux environment on the Intel® Xeon Phi™ coprocessor
- Exploring differences in serial vs. parallel programming / processing / hardware usage
- Accelerated clusters
- Optimizations of vector arithmetics, memory traffic, thread parallelism and communication
- Using the Intel® Math Kernel Library

[Register Now!](#)

[colfaxresearch.com/how-series](http://colfaxresearch.com/how-series)

# Disclaimer

While best efforts have been used in preparing this training, Colfax International makes no representations or warranties of any kind and assumes no liabilities of any kind with respect to the accuracy or completeness of the contents and specifically disclaims any implied warranties of merchantability or fitness of use for a particular purpose. The publisher shall not be held liable or responsible to any person or entity with respect to any loss or incidental or consequential damages caused, or alleged to have been caused, directly or indirectly, by the information or programs contained herein. No warranty may be created or extended by sales representatives or written sales materials.

# Course Roadmap

- 1 Why Intel Parallel Architectures?
  - ▶ Parallelism and specialization – March 7
  - ▶ Programming model continuity – March 7
- 2 Programming models for Xeon Phi coprocessors
  - ▶ Native programming – March 7
  - ▶ Offload programming – March 8
- 3 Expressing Parallelism
  - ▶ Introduction to vectorization – March 9
  - ▶ Crash-course on OpenMP – March 10
- 4 Optimization – intro on March 11
  - ▶ Vectorization tuning – March 14
  - ▶ Multi-threading – March 15, 16
  - ▶ Memory traffic – March 17
- 5 Tools: MKL and VTune – March 18

March 2016						
S	M	T	W	H	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		
■ — Lecture+remote access						
■ — Self-study/remote access						

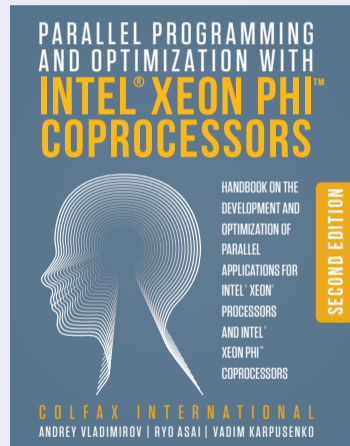
# Textbook

ISBN: 978-0-9885234-0-1 (508 pages, Electronic or Print)

## Parallel Programming and Optimization with Intel® Xeon Phi™ Coproprocessors

Handbook on the Development and  
Optimization of Parallel Applications  
for Intel® Xeon® Processors  
and Intel® Xeon Phi™ Coprocessors

© Colfax International, 2015



<http://xeonphi.com/book>

**COLFAX RESEARCH**
Log In/Out or Register

CONTRIBUTING TO INNOVATIONS IN COMPUTING

READ WATCH LEARN CONNECT JOIN



To search, type and hit enter

**Popular**

**The Hands-On Tutorials (HOT) webinars: details on efficient programming for Intel architecture**

**The Hands-On Workshop (HOW) Series**

**Introduction to Intel DAAL, Part 1: Polynomial Regression with Batch Mode Computation**

**Research and Educational Publications**

**Introduction to Intel DAAL, Part 1: Polynomial Regression with Batch Mode Computation**


**Optimization Techniques for the Intel MIC Architecture, Part 3 of 3: False Sharing and Padding**

**Software Developer's Introduction to the HGST Ultrastar Archive HaaS SMR Drives**

**Optimization Techniques for the Intel MIC Architecture, Part 2 of 3: Strip-Mining for Vectorization**

**Optimization Techniques for the Intel MIC Architecture, Part 1 of 3: Multi-Threading and Parallel Reduction**

**Performance to Power and Performance to Cost Ratios with Intel Xeon Phi Coprocessors (and why xx Acceleration May be Enough)**





**Parallel Programming Book**

Introduction to parallel programming, deep discussion of optimization techniques, exercises.

© 2015, Colfax International. 508 pages.

**Featured Video**



Additional Reading

See Research material on vectorization in a streaming code

<http://colfaxresearch.com/?p=708>

Events
Documentation
Caricature

## Consulting

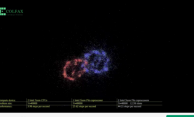



Colfax offers consulting services for enterprises, research help you to:

- Optimize your existing application to take advantage of parallelism, from vectors to cores to clusters and GPUs
- Future-proof your application for upcoming innovations
- Accelerate your application using coprocessor technology
- Investigate the potential system configurations that satisfy your cost, power, performance requirements.
- Take a Hands-On Workshop: a virtual workshop to explore your computing problems, software experience in architecture, software development, and hardware design.

**All Video Courses - COP 801 - Chapter 2 - Episode 2.1**

**Episode 2.1 - Purpose of the MIC architecture**



Learn the purpose of the system heterogeneous computing models, in a 2.1 episode we will discuss the requirements, as reported in the last training. These scenarios make an outline, because they are effective getting CPU based applications to execute with high performance efficiency.

about networking design to architect an accelerating hardware that supporting them


we will also learn the requirements, the 2.1 and 2.2 episodes, in a 2.2 episode you will

## Software Developer's Introduction to the HGST Ultrastar Archive HaaS SMR Drives



For this paper we will discuss the new HGST Ultrastar Archive HaaS SMR drives, different features to allow efficient storage operations of 100 exabyte data center high density storage capacities. These drives are well suited for large "data center" applications, such as data archive applications, the drives benefiting from better features.

## Fluid Dynamics with Fortran on Intel Xeon Phi coprocessors



Fluid dynamics is a Fortran program that simulates a fluid under the influence of aerodynamic forces and heat transfer. The equations of the conservation of mass, momentum, and energy are solved numerically for the flow over an airfoil. The program is written in Fortran and runs on Intel Xeon Phi coprocessors. The code is available on GitHub and the performance results will be published in an upcoming paper.

## Configuration and Benchmarks of Peer-to-Peer Communication over Gigabit Ethernet and InfiniBand in a Cluster with Intel Xeon Phi Coprocessors



Peer-to-peer communication over Gigabit Ethernet and InfiniBand in a cluster with Intel Xeon Phi coprocessors. The diagram shows a cluster of nodes connected via network interfaces. The nodes are labeled with their IP addresses and the network interfaces are labeled with their names.

## Interview with James Reinders: future of Intel MIC architecture, parallel programming, education




James Reinders is a software engineer at Intel. He is the author of the book "Intel MIC Architecture: Parallel Programming and Education". In this interview, he discusses the future of Intel MIC architecture, parallel programming, and education.

## Parallel Computing in the Search for New Physics at LHC



Parallel computing in the search for new physics at LHC. The image shows a particle detector with a central vertex and many tracks radiating outwards. The tracks are labeled with their respective particle types and energies.

## Interview with James Reinders: future of Intel MIC architecture, parallel programming, education



James Reinders is a software engineer at Intel. He is the author of the book "Intel MIC Architecture: Parallel Programming and Education". In this interview, he discusses the future of Intel MIC architecture, parallel programming, and education.

<http://colfaxresearch.com/>  
 (already registered? **get \$10 off the book**)

# HOW Online

Course page: [colfaxresearch.com/how-series](http://colfaxresearch.com/how-series)

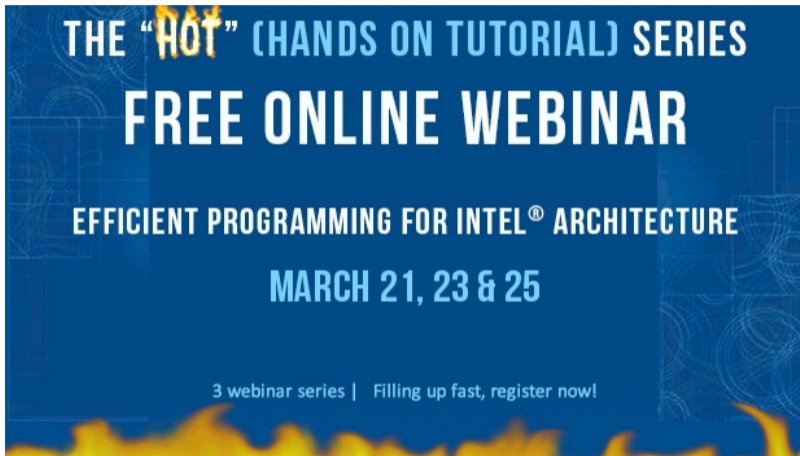
- Slides (including this one), code downloads
- Video of recorded sessions
- Chat (during webinars or offline)



Additional resources:

- More workshops like this one: [colfaxresearch.com/how-series](http://colfaxresearch.com/how-series)
- Video courses: [colfaxresearch.com/video-courses](http://colfaxresearch.com/video-courses)
- [Intel Many Integrated Core Architecture Forum](#)

# HOT Series

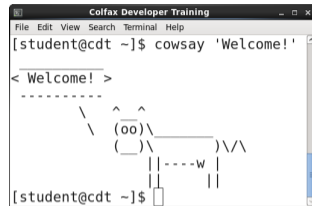
The banner has a dark blue background with faint white technical drawings of gears and circuitry. At the bottom, there is a stylized flame graphic in yellow and orange. The text is centered and uses a mix of white and light blue colors.

THE “HOT” (HANDS ON TUTORIAL) SERIES  
**FREE ONLINE WEBINAR**  
EFFICIENT PROGRAMMING FOR INTEL® ARCHITECTURE  
MARCH 21, 23 & 25  
3 webinar series | Filling up fast, register now!

[colfaxresearch.com/hot-16-03/](http://colfaxresearch.com/hot-16-03/)

# Hands-On Exercises and Remote Access

- 96 people receive a remote access token
- Can access the system the entire 3 weeks of the workshop
- Not among the 96? Stay tuned: follow along with instructor, use own system, or wait for a seat
- Use it or lose it: if you do not log in for a while, remote access token goes to next student on the list



```
Colfax Developer Training
File Edit View Search Terminal Help
[student@cdt ~]$ cowsay 'Welcome!'
< Welcome! >
-----
      \   ^__^
         (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||

[student@cdt ~]$
```

## §2. Intel Architecture

# Computing Platforms

# Computing Platforms

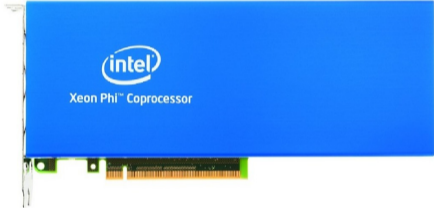
Intel Xeon Processor



Current: Haswell  
Upcoming: Broadwell

Multi-Core Architecture

Intel Xeon Phi Coprocessor, 1st generation



Current: Knights Corner (KNC)

Intel Xeon Phi Processor, 2nd generation\*



\* socket and coprocessor versions

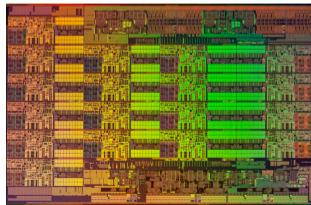
Upcoming: Knights Landing (KNL)

Intel Many Integrated Core (MIC) Architecture

# Intel Xeon CPU: Purpose and Specifications

General-purpose platform for demanding computing applications.

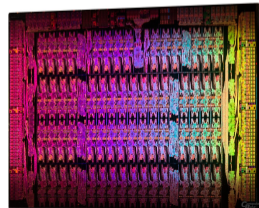
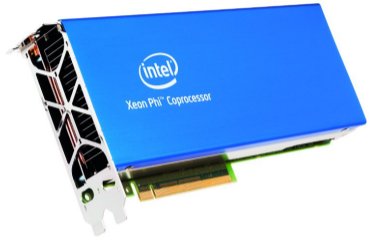
- Up to  $\sim 1$  TFLOP/s in DP
- Up to  $\sim 2$  TFLOP/s in SP
- Up to 768 GiB DDR4 RAM
- $\sim 126$  GB/s bandwidth
- Hardware-rich: forgiving of sub-optimal code



# Intel Xeon Phi Processors (1st Gen)

Specialized platform for demanding computing applications.

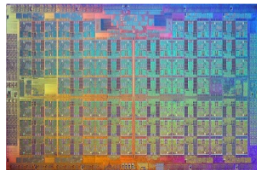
- PCIe end-point device
- ~ 1.2 TFLOP/s in DP
- ~ 2.4 TFLOP/s in SP
- Up to 16 GiB GDDR5 RAM
- ~ 176 GB/s bandwidth
- Heterogeneous clustering
- Runs special Linux distribution



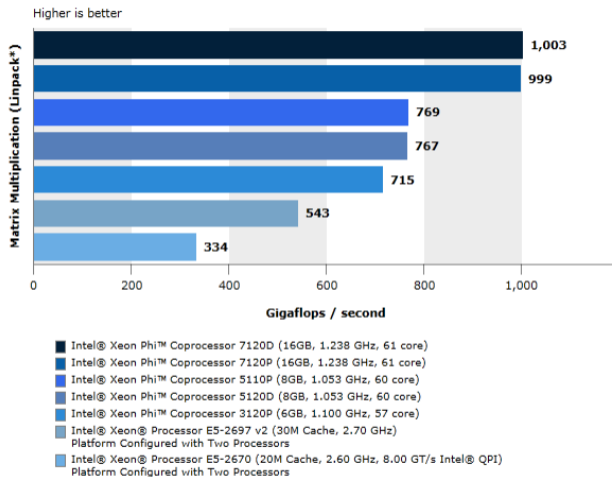
# Intel Xeon Phi Processors (2nd Gen)

Specialized platform for demanding computing applications.

- Socket version or coprocessor
- 3+ TFLOP/s in DP
- 6+ TFLOP/s in SP
- Up to 16 GiB MCDRAM
- ~ 400 GB/s MCDRAM bandwidth
- Up to 384 GiB DDR4 RAM
- ~ 90 GB/s DDR4 bandwidth
- Supports common OS
- **Public disclosures**

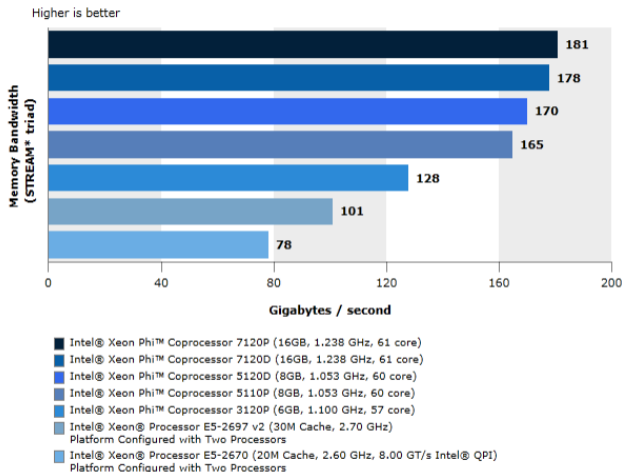


# Intel Xeon Phi Coprocessors: HPC LINPACK Benchmark



Source: “Intel Xeon Phi Coprocessor LINPACK\* and STREAM\* Performance”

# Intel Xeon Phi Coprocessors: STREAM Benchmark

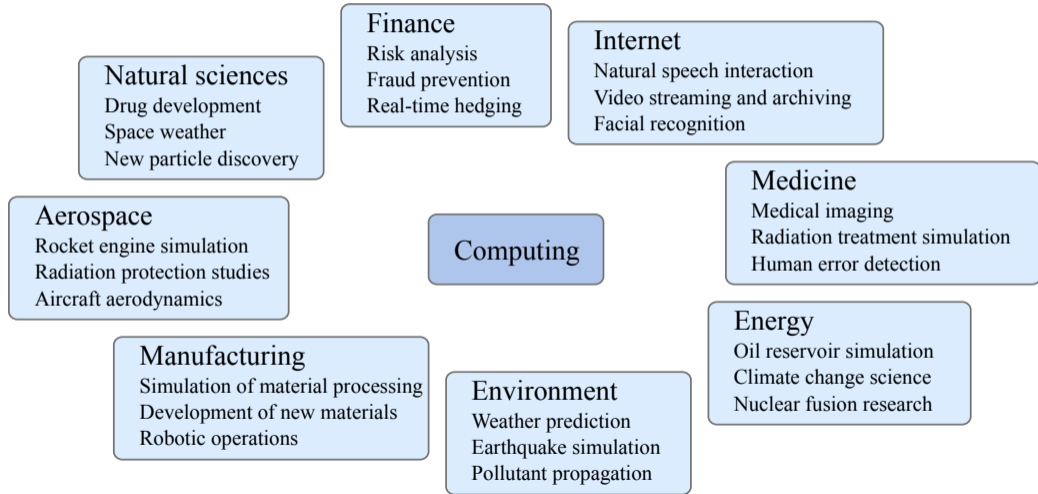


Source: “Intel Xeon Phi Coprocessor LINPACK\* and STREAM\* Performance”

# It Is All About Performance

# Computing Applications

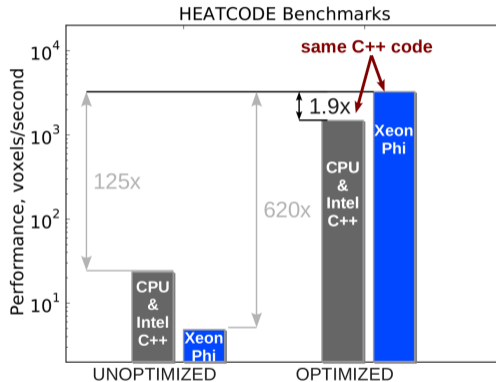
## Just some examples



# Will My Code Run Faster on KNL?

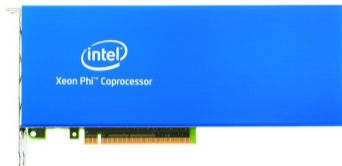
Performance on MIC architecture is a function of optimization Level

- Performance will be disappointing if code is not optimized for multi-core CPUs
- Optimized code runs better on the MIC platform *and* on the multi-core CPU
- Single code for two platforms + Ease of porting = Incremental optimization



See [this case study](#)

# Coprocessor vs Processor Performance



One Intel Xeon Phi 7120P  
coprocessor

*vs.*



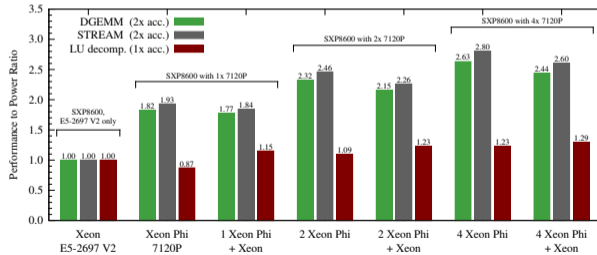
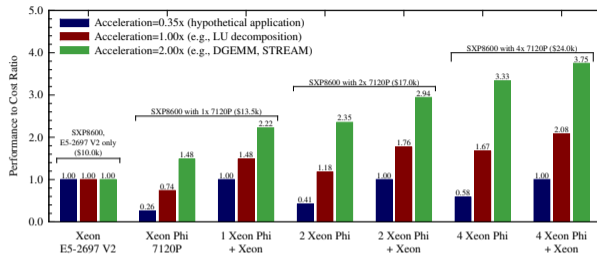
Two Intel Xeon E5-2697 v2  
CPUs

- Why compare 1 coprocessor against 2 processors?  
Same thermal design power (TDP).

See also [“Intel Xeon Product Family: Performance Brief”](#)

# What is Your Performance Metric?

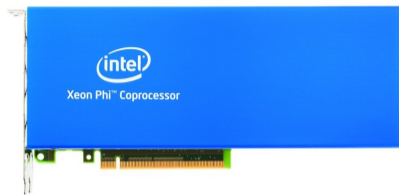
- Performance per System
- Performance per Watt
- Performance/Cost Ratio



See [this paper](#) for details

# Common Architecture, Programming Models

# Bird's Eye View

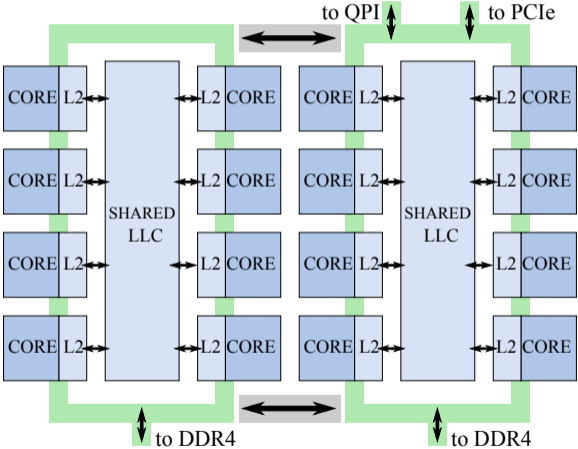


- C/C++/Fortran; OpenMP/MPI
- Standard Linux OS
- Up to 768 GiB of DDR4 RAM
- $\leq 18$  cores/chip  $\approx 3$  GHz
- 2 hyper-threads per core
- 256-bit AVX vectors

- C/C++/Fortran; OpenMP/MPI
- Special Linux distribution
- 3–16 GiB cached GDDR5 RAM
- Up to 61 cores at  $\approx 1.2$  GHz
- 4 hardware threads per core
- 512-bit IMCI vectors

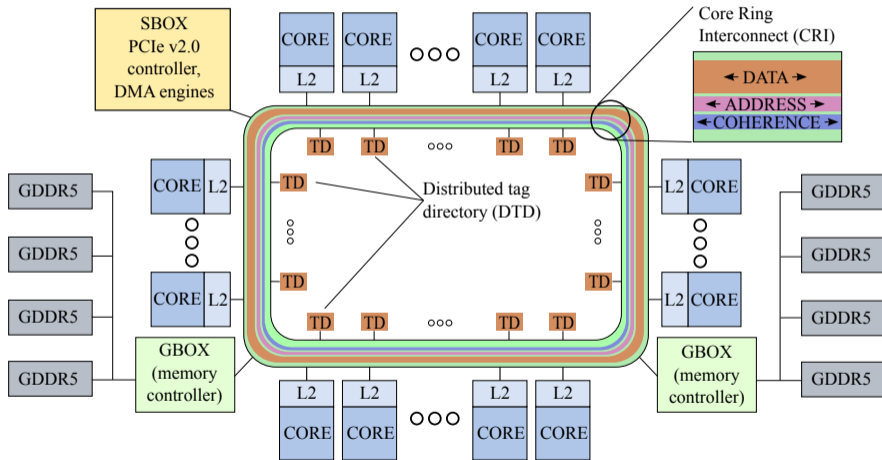
# Intel Xeon CPU: Die Organization

Likes data locality, but large LLC is forgiving.



# KNC Die Organization

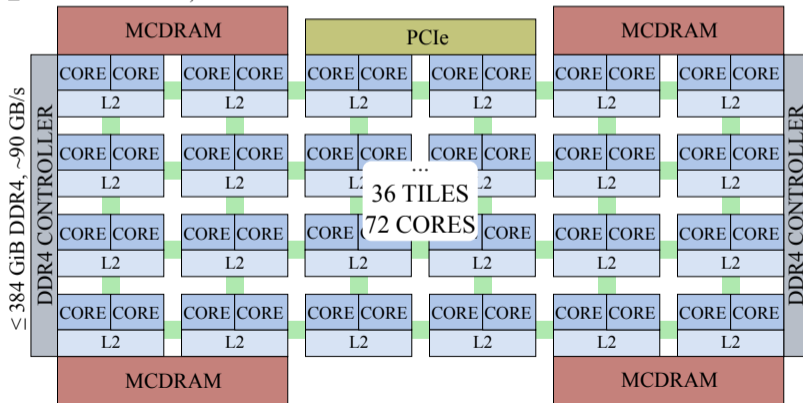
In a ring bus with distributed cache, data access locality is key.



# KNL Die Organization

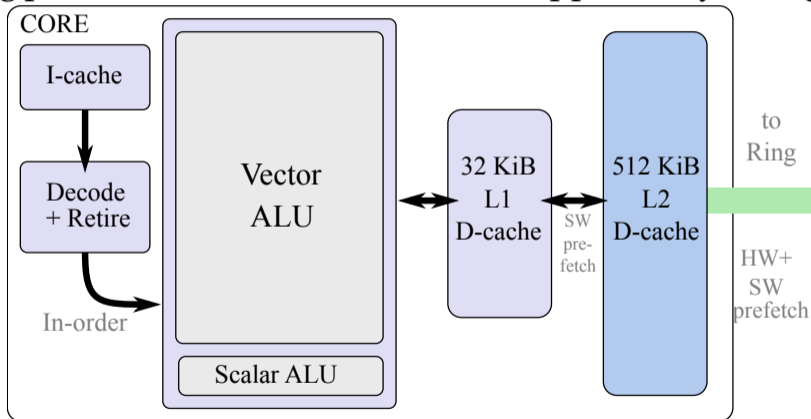
- Mesh interconnect relaxes data locality requirement [somewhat]
- All-to-all, quadrant or sub-numa domain communication in mesh

≤ 16 GiB MCDRAM, ~ 400 GB/s



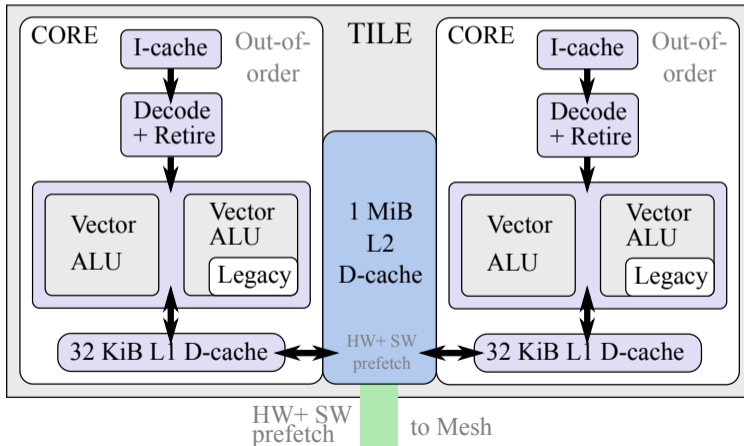
# KNC Cores

Computing power is in vector units. Scalar support only for legacy usage.

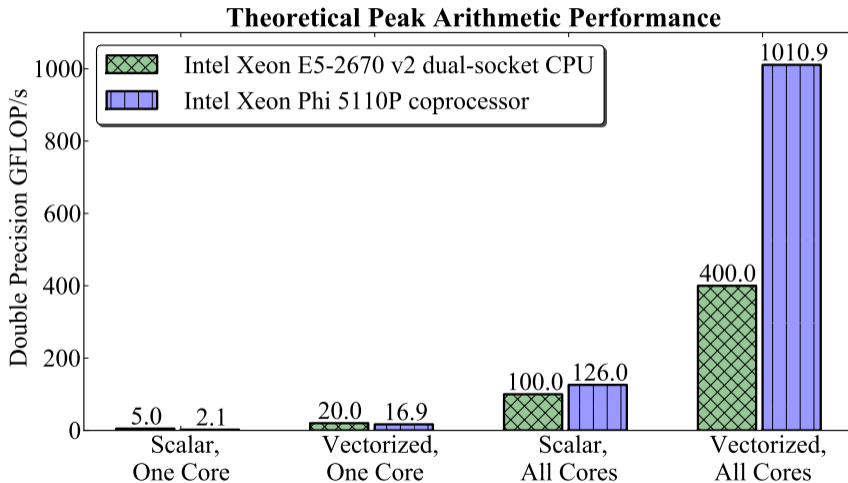


# KNL Cores

- Even more power in vector units
- Binary compatible with Xeon, but in legacy mode

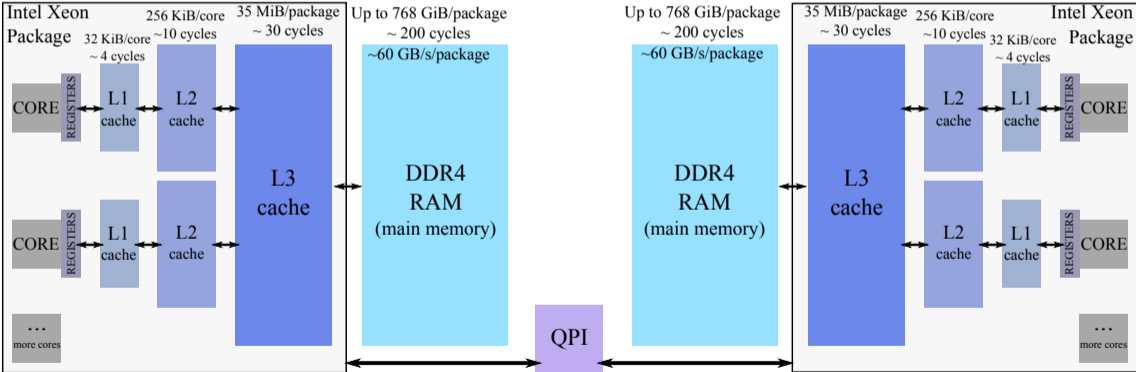


# Task and Data Parallelism



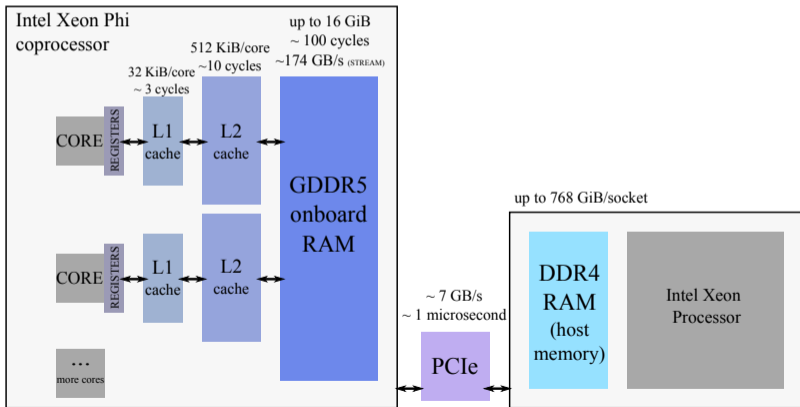
# Intel Xeon CPU: Memory Organization

- Hierarchical cache structure
- Two-way processors have NUMA architecture



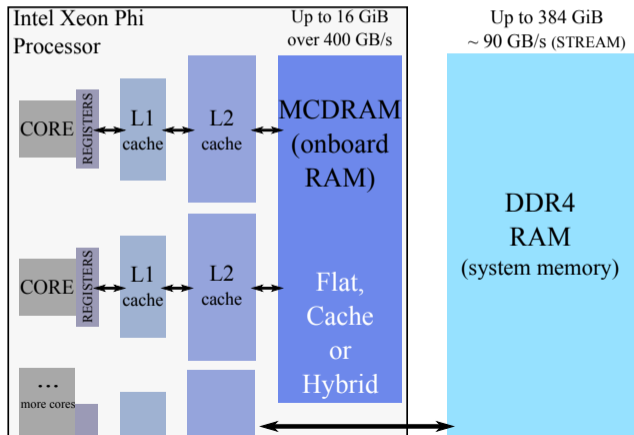
# KNC Memory Organization

- Direct access to  $\leq 16$  GiB of cached GDDR5 memory on board
- No access to system DDR4, connected to host via PCIe

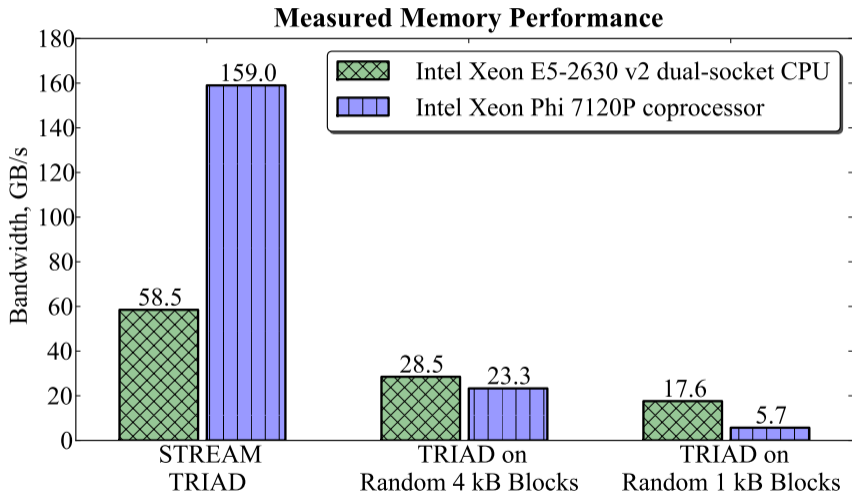


# KNL Memory Organization

- Direct access to onboard MCDRAM *and* system DDR4
- Use MCDRAM as cache, in flat mode, or as hybrid



# Memory Access Pattern



# What's New in Knights Landing

## KNL: What's New

	<b>Haswell</b>	<b>KNC</b>	<b>KNL</b>
Form factor	Socket	PCIe	PCIe, Socket
Core	Out-of-order	In-order	Out of order
Vectors	$\leq$ AVX2	IMCI	$\leq$ AVX2 + AVX-512
Memory	DDR4	GDDR5	MCDRAM+DDR4
RDMA	Over PCIe	Over PCIe	PCIe or Integrated
Programming	Traditional	Native, Offload	Traditional, Offload
Performance, Optimization	High and forgiving	Higher; needs good code	3x KNC 1-threaded and parallel

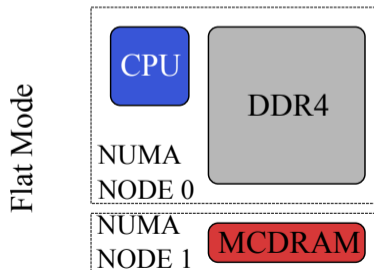
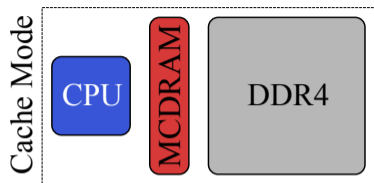
# Using High-Bandwidth Memory (MCDRAM)

## Option 1 : cache/hybrid mode

- Treat it as LLC
- Data locality techniques
- Miss latency 2x the direct DDR4 access

## Option 2 : flat mode

- Application fits in 16 GiB? `numactl`
- More than 16 GiB data? Use special allocators (e.g., `memkind`)



# Using New Vector Features

Efficient gather/scatter, conflict detection, high-precision exponential and reciprocal, two vector units per core...

## Option 1 : automatic vectorization

- No changes to code, possibly adjust to 512-bit vector width
- Recompile with `-xMIC-AVX512`

## Option 2 : explicit vectorization

- Bite the bullet

See also [this post](#)



## Bottom Line

The best way to prepare...



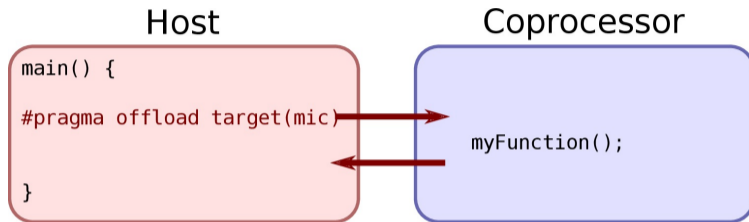
KEEP  
CALM  
AND  
GO  
PARALLEL

# §3. Programming Coprocessors

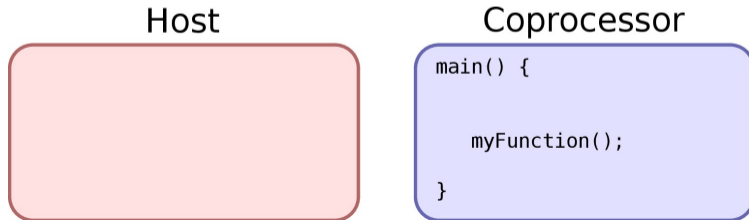
# Offload and Native Models

# Offload and Native Models

- Offload model (explicit/virtual-shared memory/OpenMP 4.0):



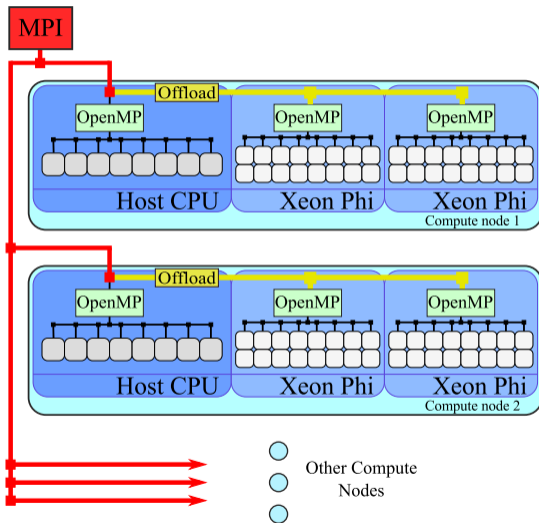
- Native model (standalone application/MPI process):



# Heterogeneous Distributed Computing with Xeon Phi

## Option 1: MPI+OpenMP with Offload.

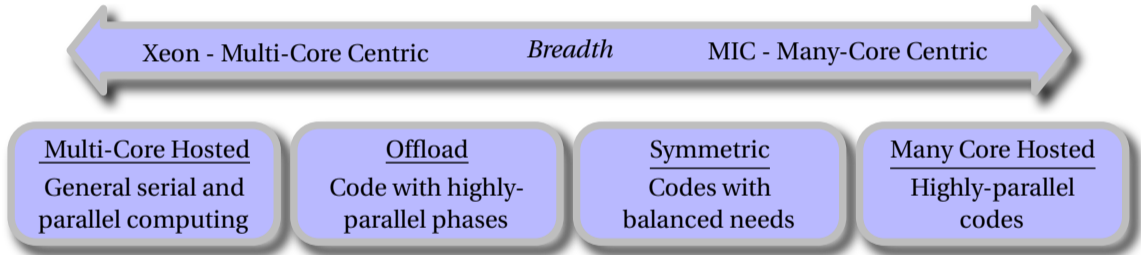
- MPI processes are multi-threaded with OpenMP.
- MPI runs only on CPUs.
- MPI processes offload to coprocessor(s).
- OpenMP in offload regions.





# Teaming Xeon Processors with Xeon Phi Coprocessors

Programming models allow a range of CPU+MIC coupling modes



# Software Configuration

# Intel Manycore Platform Software Stack

- micinfo – system information
- micsmc – monitor and modify the physical parameters: temperature, power modes, core utilization, etc.
- micctrl – configure the Intel Xeon Phi coprocessor operating system
- miccheck – verify the Intel Xeon Phi coprocessor configuration
- micrasd – log of hardware errors reported by Intel Xeon Phi coprocessors
- micflash – flash memory agent



Monitoring MIC activity with  
micsmc (an MPSS tool)

# Linux Environment on Intel Xeon Phi Coprocessors

```
vega@lyra% lspci | grep -i "co-processor"
06:00.0 Co-processor: Intel Corporation Xeon Phi coprocessor 7120 series (rev 20)
82:00.0 Co-processor: Intel Corporation Xeon Phi coprocessor 7120 series (rev 20)
vega@lyra% sudo service mpss status
mpss is running
vega@lyra% cat /etc/hosts | grep mic
172.31.1.1  lyra-mic0 mic0
172.31.2.1  lyra-mic1 mic1
vega@lyra% ssh mic0

vega@mic0% cat /proc/cpuinfo | grep proc | tail -n 3
processor: 241
processor: 242
processor: 243
vega@mic0% ls /
amplxe  dev    home   lib64  oldroot  proc   sbin   sys    usr
bin     etc    lib    linuxrc  opt      root   sep3.10  tmp   var
```

# Software Necessary to Build Xeon Phi Applications:

**Compilers** : Intel C Compiler, Intel C++ Compiler, and Intel Fortran Compiler — mandatory

**Optimization tools** : Intel VTune Amplifier XE and Intel Trace Analyzer and Collector (ITAC) — highly recommended

**Mathematics support** : Intel Math Kernel Library (MKL) — highly recommended

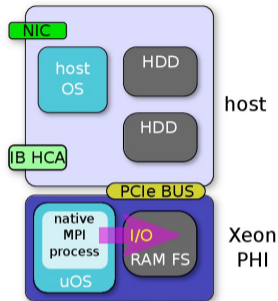
**Cluster Development** : Intel MPI — industry standard parallel framework

**Development** : Intel Inspector XE, Intel Advisor XE — optional

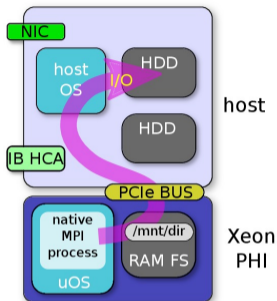


All-in-One bundle,  
**Intel Parallel Studio XE**

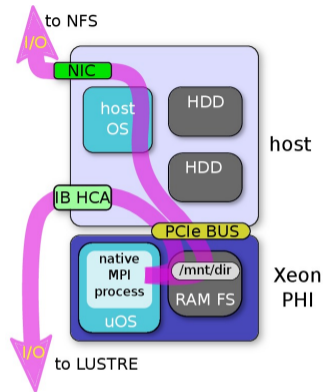
# Working with Files on Coprocessors



RAM Filesystem



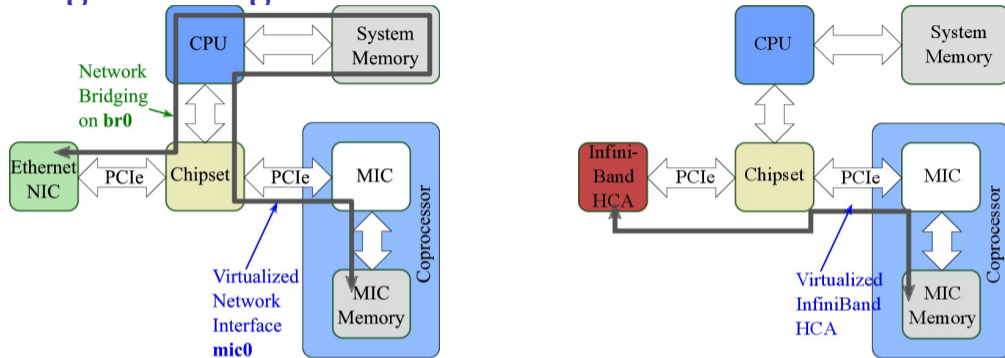
VirtIO Transfer



Network Storage

Details: <http://xeonphi.com/papers/io>

# Bridged Configuration for Peer-to-Peer Communication



- Left: Gigabit Ethernet bridging and TCP/IP virtualization
- Right: InfiniBand on Xeon Phi with the Coprocessor Communication Link (CCL) technology
- Details: <http://xeonphi.com/papers/p2p>

# §4. Hands-On Part: Native Programming

# Native Execution

“Hello World” application:

```
1 #include <stdio>
2 #include <unistd.h>
3 int main(){
4     printf("Hello world! I have %ld logical cores.\n",
5         sysconf(_SC_NPROCESSORS_ONLN ));
6 }
```

Compile and run on host:

```
vega@lyra% icpc hello.cc
vega@lyra% ./a.out
Hello world! I have 48 logical cores.
vega@lyra%
```

# Native Execution

Compile and run the same code on the coprocessor in the native mode:

```
vega@lyra% icpc hello.cc -mmic
vega@lyra% scp a.out mic0:~/
a.out 100% 10KB 10.4KB/s 00:00
vega@lyra% ssh mic0
vega@mic0% pwd
/home/lyra
vega@mic0% ls
a.out
vega@mic0% ./a.out
Hello world! I have 244 logical cores.
vega@mic0%
```

- Use `-mmic` to produce executable for MIC architecture
- Must transfer executable to coprocessor (or NFS-share) and run from shell
- Native MPI applications work the same way (need Intel MPI library)

# Alternative Native Application Launcher: micnativeloadex

The tool `micnativeloadex` automatically transfers code and dependent libraries and runs the application.

```
vega@lyra% export SINK_LD_LIBRARY_PATH=/opt/intel/composerxe/compiler/lib/mic
vega@lyra% icpc hello.cc -mmic
vega@lyra% micnativeloadex a.out
Hello world! I have 244 logical cores.
vega@lyra%
```

- Set `SINK_LD_LIBRARY_PATH` to help the tool find libraries
- Do not have to SSH into the coprocessor
- Runs under `micuser` on coprocessor

# Native Applications with Autotools

- Use the Intel compiler with flag `-mmic`
- Eliminate assembly and unnecessary dependencies
- Use `--host=x86_64` to avoid “program does not run” errors

Example, the GNU Multiple Precision Arithmetic Library (GMP):

```
vega@lyra% wget https://ftp.gnu.org/gnu/gmp/gmp-5.1.3.tar.bz2
vega@lyra% tar -xf gmp-5.1.3.tar.bz2
vega@lyra% cd gmp-5.1.3
vega@lyra% ./configure CC=icc CFLAGS="-mmic" --host=x86_64 --disable-assembly
...
vega@lyra% make
...
```

# Review and What's Next

- Intel Xeon and Intel Xeon Phi – parallel processors
- Xeon Phi (MIC architecture) – specialized for highly parallel workloads without complex memory access
- Coprocessor – either offload device or an additional compute node
- Native+offload programming allow for a range of design options

Next session: details of the offload model.