



PROGRAMMING AND OPTIMIZATION FOR INTEL[®] ARCHITECTURE

Hands-On Workshop (HOW) Series "Deep Dive"
Session 1

Colfax International — colfaxresearch.com

April 2017

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.

- ▶ **Module I. Programming**
 - 01. Intel Architecture and Modern Code – Apr 17
 - 02. Xeon Phi, Coprocessors, Omni-Path – Apr 18
- ▶ **Module II. Expressing Parallelism**
 - 03. Automatic vectorization – Apr 19
 - 04. Multi-threading with OpenMP – Apr 20
 - 05. Distributed Computing, MPI – Apr 21
- ▶ **Module III. Optimization**
 - 06. Optimization Overview: N-body – Apr 24
 - 07. Scalar tuning, Vectorization – Apr 25
 - 08. Common Multi-threading Problems – Apr 26
 - 09. Multi-threading, Memory Aspect – Apr 27
 - 10. Access to Caches and Memory – Apr 28

April 2017						
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						

■ — Webinar+remote access

Course page:

colfaxresearch.com/how-17-04

- ▶ Slides
- ▶ Code
- ▶ Video
- ▶ Chat

More workshops:

colfaxresearch.com/training



GET YOUR QUESTIONS ANSWERED: CHAT



colfaxresearch.com/how-17-04

GET YOUR QUESTIONS ANSWERED: FORUMS

[READ](#)[WATCH](#)[LEARN](#)[FORUMS](#)[CONNECT](#)[JOIN](#)

Forum

[Colfax Cluster](#)

Discussion of Colfax Cluster usage policies, troubleshooting.

[Developer Training, HOW Series](#)

Questions about any of the Colfax trainings? Usage of training servers, experience with specific exercises, inquiries on what's inside, suggestions for future trainings - post them here.

[Performance Optimization and Parallelism](#)

Discuss with Colfax Research and colleagues any topics related to computational science, parallel programming, performance optimization and code modernization.

colfaxresearch.com/discussion

- ▶ All registrants receive an invitation from `cluster@colfaxresearch.com`
- ▶ Queue-based access to Intel Xeon E5, Intel Xeon Phi (KNC and KNL)
- ▶ Can access the cluster the entire 2 weeks of the workshop



MC² SERIES LEARN HOW THEY DID IT



Scientists, engineers and developers are achieving breakthrough computational performance through code modernization.

Join us and hear experts discuss performance optimization methods used in real-life applications in the all-new Modern Code Contributed (MC²) Talks - a series of free webinars.



[Learn from the experts - register now >](#)

mc2series.com

COLFAX RESEARCH

CONTRIBUTING TO INNOVATIONS IN COMPUTING

[Log In/Out](#) or [Register](#)

/
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

Parallel Programming Book

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

© 2015, Colfax International, 508 pages.

Featured Video

See Research material on vectorization in a training video

Events

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

Software Developer's Introduction to the HGST Ultrastar Archive H700 SMR Drives

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

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

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

Performance to Power and Performance to Cost Ratios with Intel Xeon Phi Coprocessors (and why ix Acceleration May Be Enough)

Consulting

Share

Share

Colfax offers consulting services for enterprises, research help you:

- Optimize your existing application to take advantage of parallelism, from vectors to cores to clusters and
- Future-proof your application for upcoming innovations
- Accelerate your application using coprocessor tech
- Investigate the potential system configurations that satisfy your cost, power, performance requirements.
- Take a clean slate to develop a novel approach to solve your computing problem

48 Videos (Categories: CPU VSI - Chapter 1 - Episode 1.1)

Episode 2.1 — Purpose of the MIC architecture

[View](#)
[Share](#)

Software Developer's Introduction to the HGST Ultrastar Archive H700 SMR Drives

[View](#)
[Share](#)

Fluid Dynamics with Fortran on Intel Xeon Phi coprocessors

[View](#)
[Share](#)

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

[View](#)
[Share](#)

http://colfaxresearch.com/

colfaxresearch.com/how-17-04

RESOURCES

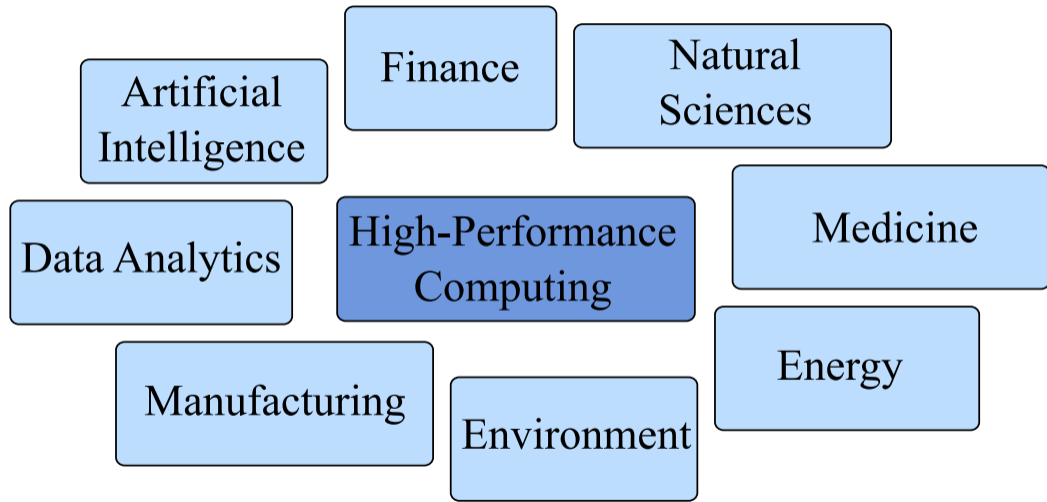
© Colfax International, 2013–2017



§2. MODERN CHALLENGES



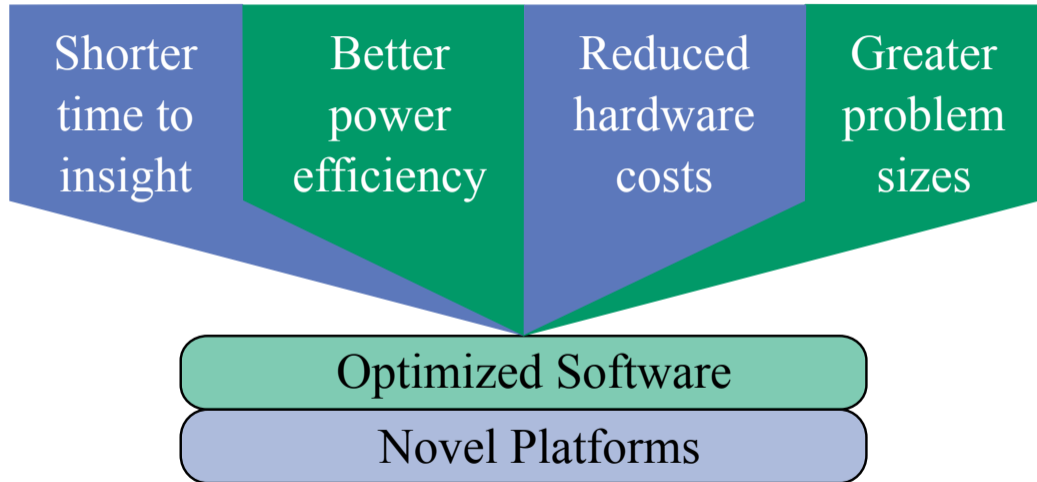
AREAS OF APPLICATION





NEED FOR SPEED

PERFORMANCE OPTIMIZATION OUTCOMES



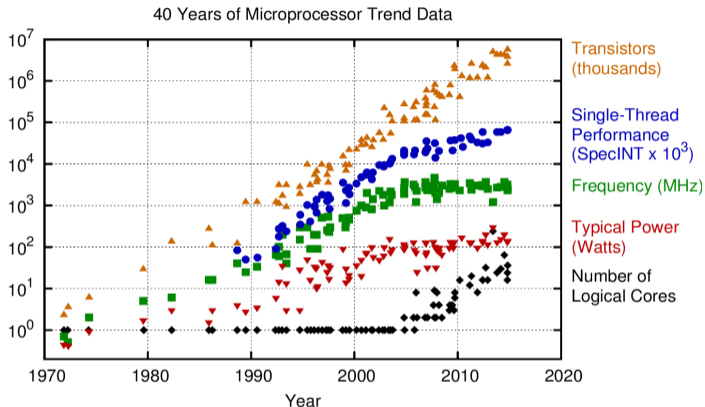


§3. MODERN ARCHITECTURE



HOW PROCESSORS GET FASTER

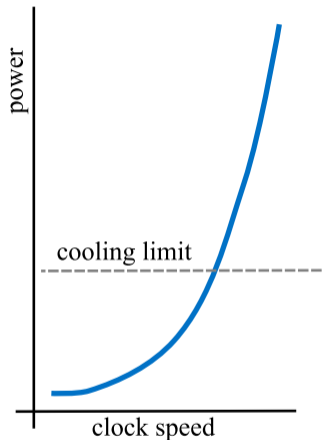
40 YEARS OF MICROPROCESSOR DATA



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
 New plot and data collected for 2010-2015 by K. Rupp

Source: karlrupp.net

POWER WALL



Liquid nitrogen for CPU
overclocking

youtu.be/WZr0W_g0dqk



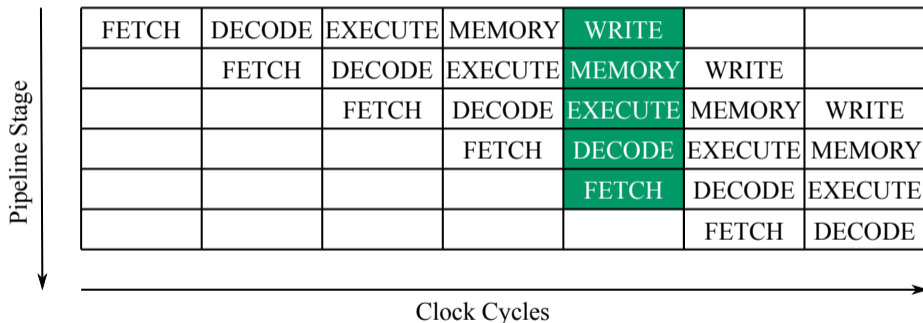
Microsoft's project Natick:
immersed datacenter

news.microsoft.com/natick

Cooling solutions for high clock speeds are not practical or expensive

INSTRUCTION-LEVEL PARALLELISM (ILP) WALL: PIPELINING

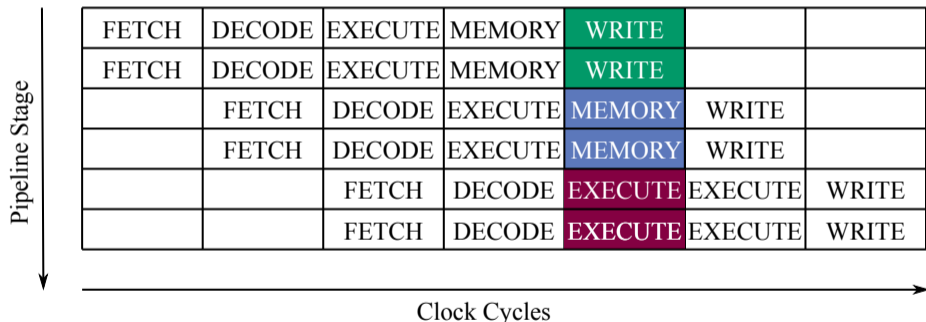
Pipelining – replication of hardware to run different stages of different instruction streams at the same time



Only so many pipeline stages, possible conflicts

INSTRUCTION-LEVEL PARALLELISM (ILP) WALL: SUPERSCALAR EXECUTION

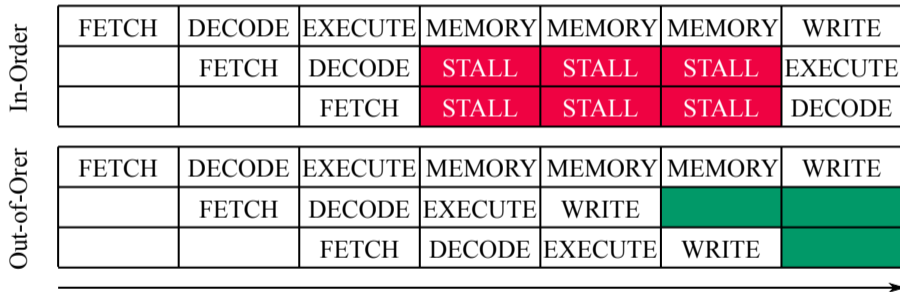
Superscalar Execution – hardware checks for independence of operations, pipelines multiple instructions in a cycle



Automatic search for independent instructions requires extra resources

MEMORY WALL: OUT-OF-ORDER EXECUTION

Out-of-order Execution – hardware re-orders instructions in a stream to minimize latencies

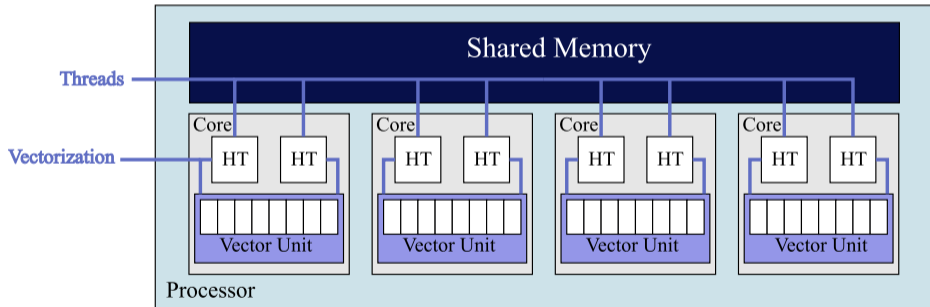


CPU performance grows faster than RAM bandwidth, OOE can't fill gap

PARALLELISM

CORES – multiple instructions on multiple data elements (MIMD)

VECTORS – single instruction on multiple data elements (SIMD)



Unbounded growth opportunity, but **not automatic**

PARALLELISM IS THE PATH FORWARD

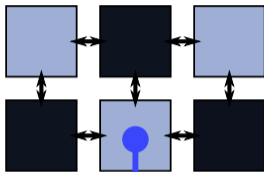
- ▶ Clock speed has hit the power wall
- ▶ Automatic parallelism has hit the ILP wall
- ▶ Out-of-order execution cannot overcome the memory wall

The Show Must Go On

Hardware keeps evolving through parallelism.
Software must catch up!

PARALLEL PROGRAMMING LAYERS

CLUSTER COMPUTING
in distributed memory



```
MPI_Sendrecv(data, k,
              MPI_DOUBLE, data2,
              ... );
```

MULTITHREADING
in shared memory



```
#pragma omp parallel for
for (j = 0; j < m; j++)
  ComputeSubset(j);
```

VECTORIZATION
of floating-point math



```
#pragma omp simd
for (i = 0; i < n; i++)
  A[i] += B[i];
```



INTEL ARCHITECTURE

INTEL COMPUTING PLATFORMS

General-Purpose Processors

Intel® Xeon®
Intel® Core™
Intel® Atom™, ...



Specialized Processors

Intel® Xeon Phi™
processors
and coprocessors



Computing Accelerators

Intel® VCA (x86)
Intel® Nervana™ Platform
Intel® DLIA™ (FPGAs)



Network Interconnects

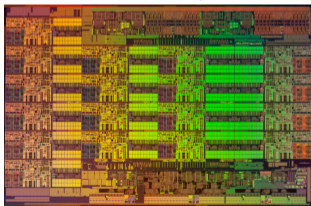
Intel® Omni-Path™
Architecture



INTEL XEON PROCESSORS

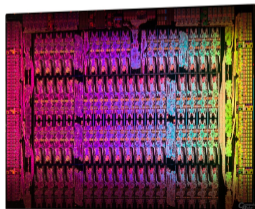
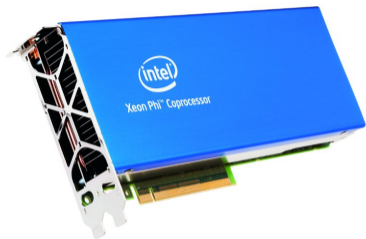
- ▷ 1-, 2-, 4-way
- ▷ General-purpose
- ▷ Highly parallel (44 cores*)
- ▷ Resource-rich
- ▷ Forgiving performance
- ▷ Theor. ~ 1.0 TFLOP/s in DP*
- ▷ Meas. ~ 154 GB/s bandwidth*

* 2-way Intel Xeon processor, Broadwell architecture (2016), top-of-the-line (e.g., E5-2699 V4)



INTEL XEON PHI PROCESSORS (1ST GEN)

- ▶ PCIe add-in card
- ▶ Specialized for computing
- ▶ Highly-parallel (61 cores*)
- ▶ Balanced for compute
- ▶ Less forgiving
- ▶ Theor. ~ 1.2 TFLOP/s in DP*
- ▶ Meas. ~ 176 GB/s bandwidth*

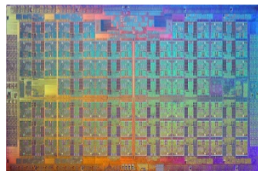
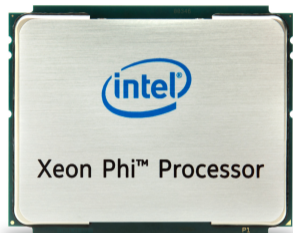


* Intel Xeon Phi coprocessor, Knights Corner architecture (2012), top-of-the-line (e.g., 7120P)

INTEL XEON PHI PROCESSORS (2ND GEN)

- ▶ Bootable or PCIe add-in card
- ▶ Specialized for computing
- ▶ Highly-parallel (72 cores*)
- ▶ Balanced for compute
- ▶ Less forgiving than Xeon
- ▶ Theor. ~ 3.0 TFLOP/s in DP*
- ▶ Meas. ~ 490 GB/s bandwidth*

* Intel Xeon Phi processor, Knights Landing architecture (2016), top-of-the-line (e.g., 7290P)

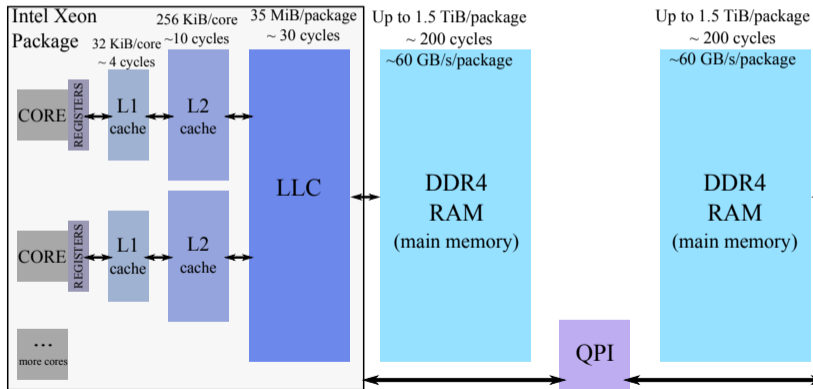




FORM-FACTORS AND MEMORY ORGANIZATION

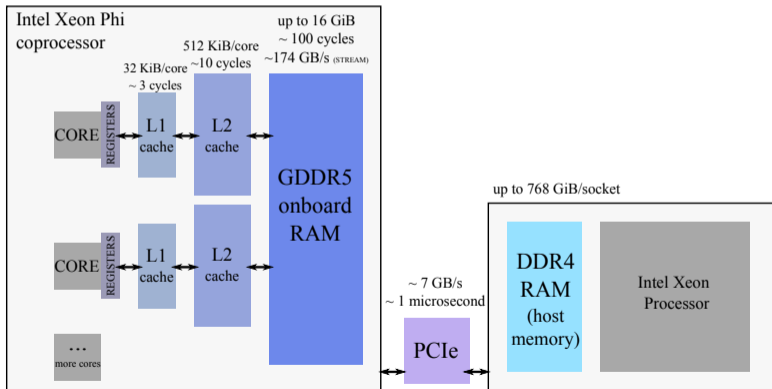
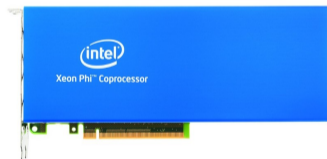
INTEL XEON CPU: MEMORY ORGANIZATION

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



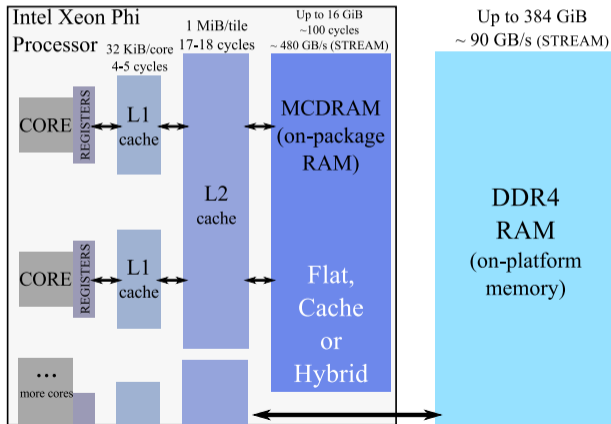
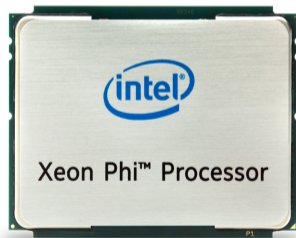
KNC MEMORY ORGANIZATION

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



KNL MEMORY ORGANIZATION (BOOTABLE)

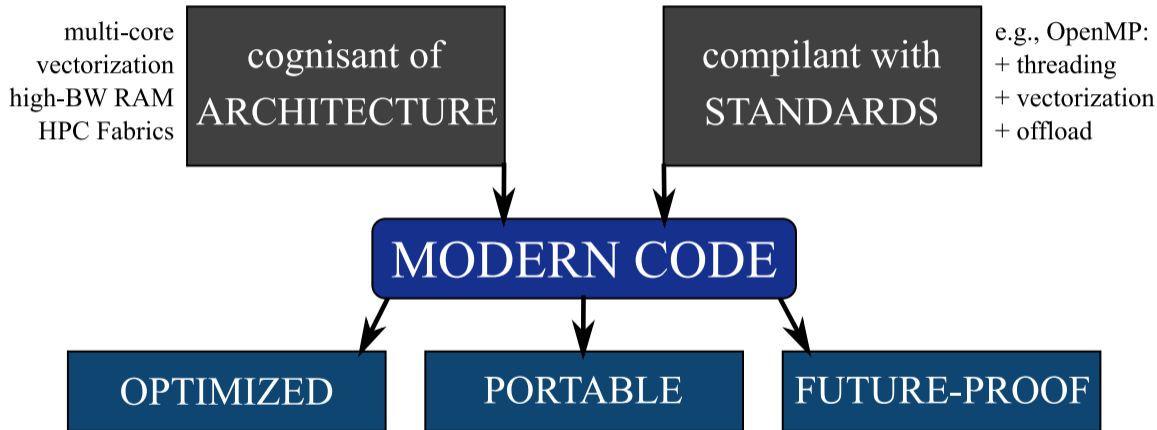
- ▶ On-package high-bandwidth memory (HBM) – MCDRAM
- ▶ Optimized for arithmetic performance and bandwidth (not latency)





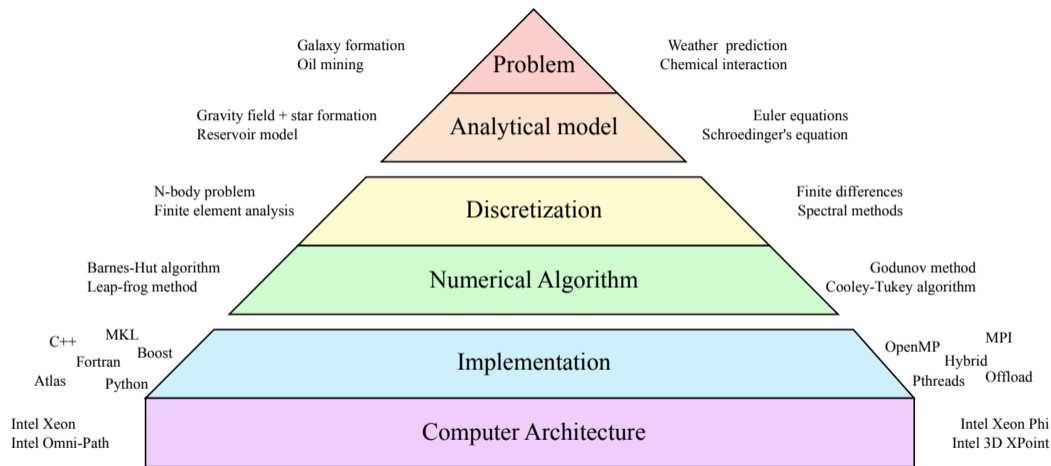
§4. MODERN CODE

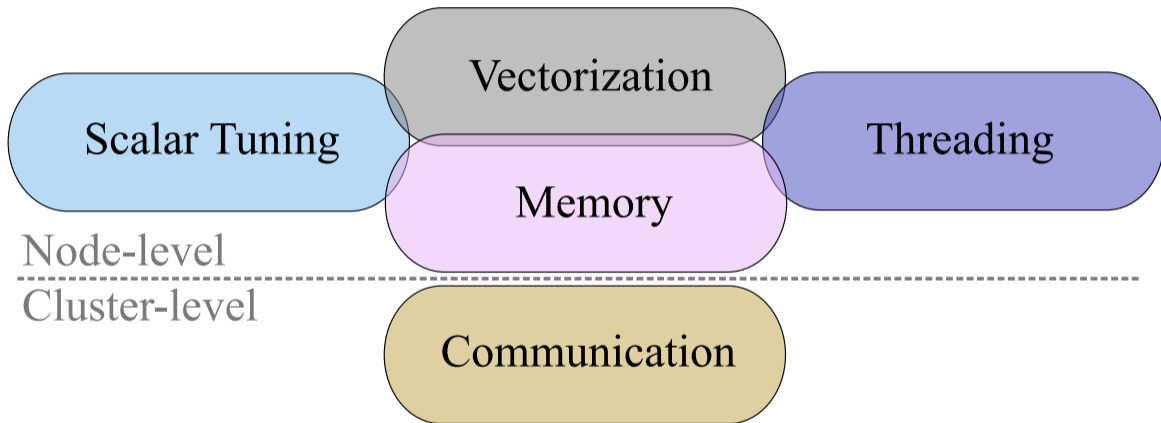
ONE CODE FOR ALL PLATFORMS





OPTIMIZATION AND FUTURE-PROOFING

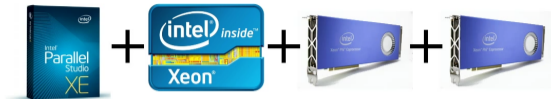






MOTIVATING EXAMPLES

ASTROPHYSICAL CODE HEATCODE: AN OFFLOAD STORY

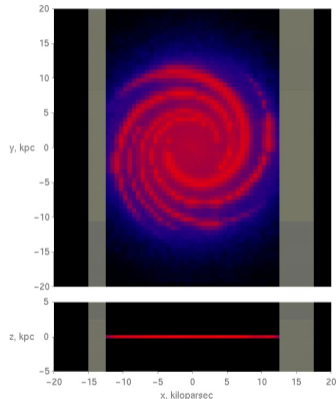


Porting to Intel® Xeon Phi™ coprocessors

- We ported Frankie code using explicit offload model
- Same code & optimization methods for Xeon Phi™
- Simultaneous calculations on CPU and coprocessors with automatic load balancing was easy to implement
- With two Intel® Xeon Phi™ coprocessors, performance for high-res calculations is 3.2x better than with two Intel® Xeon® E5 processors alone.
- **RESULT:** estimated target project calculation time is now 2 weeks (down from 6+ years)

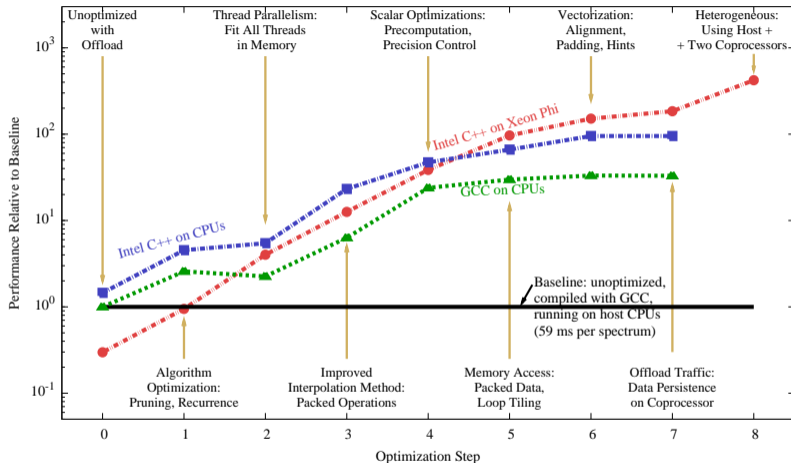
Goal achieved!

Transient Emission of Cosmic Dust Grains
in the Milky Way Galaxy,
Simulation with Frankie Code



<http://xeonphi.com/papers/heatcode>

ASTROPHYSICAL CODE HEATCODE: AN OFFLOAD STORY

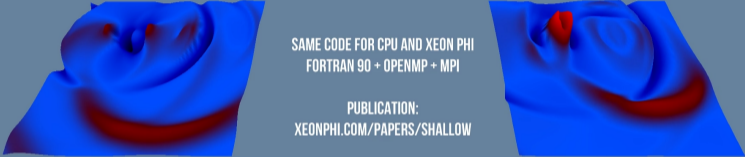


<http://xeonphi.com/papers/heatcode>

COMPUTATIONAL FLUID DYNAMICS: LEGACY CODE

FLUID DYNAMICS WITH FORTRAN ON INTEL® XEON PHI™ COPROCESSORS

SHALLOW WATER EQUATION SOLVER



SAME CODE FOR CPU AND XEON PHI
FORTRAN 90 + OPENMP + MPI


PUBLICATION:
XEONPHI.COM/PAPERS/SHALLOW

PERFORMANCE ON CPU: 19.5 GFLOP/S


PERFORMANCE WITH COPROCESSORS: 52.5 GFLOP/S

SIMULATION SIZE: 9600X9600

ACCELERATION: 2.7X



INTEL XEON E5-2697 V3 PROCESSOR



INTEL XEON E5-2697 V3 PROCESSOR +
TWO INTEL XEON PHI 7120A COPROCESSORS

COLFAX

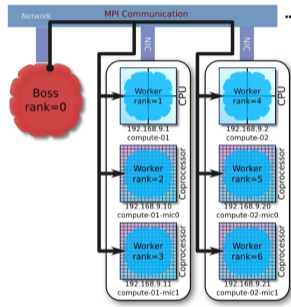
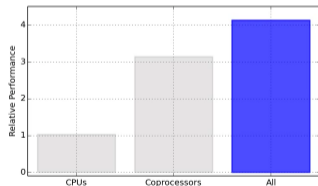
SERVERS WORKSTATIONS TRAINING CONSULTING RESEARCH

WWW.COLFAX-INTL.COM

<http://xeonphi.com/papers/shallow>

ASIAN OPTION PRICING: HETEROGENEOUS CLUSTERING

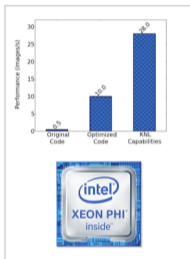
Heterogeneous Clustering with Homogeneous Code:
Asian Option Pricing



<http://xeonphi.com/papers/heterogeneous>

MACHINE LEARNING: OPTIMIZED MIDDLEWARE

INTEL® XEON PHI™ PROCESSORS — MACHINE LEARNING



NEURALTALK2 — OPEN SOURCE IMAGE TAGGING CODE (KARPATY & FEI-FEI, STANFORD)

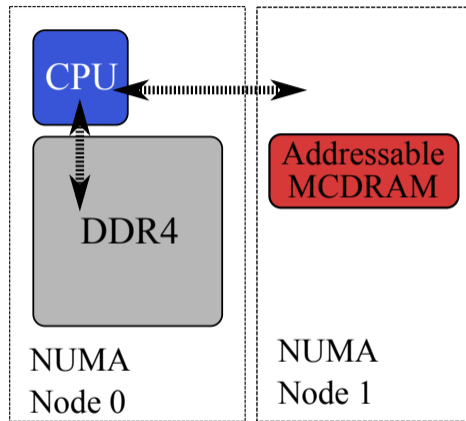
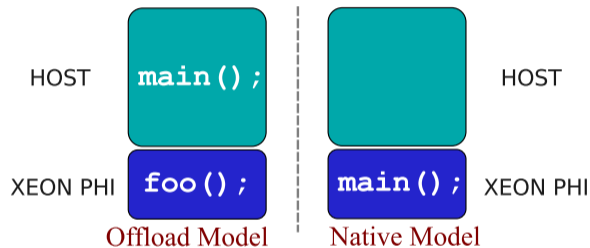


<http://colfaxresearch.com/isc16-neuraltalk>



WHAT YOU ARE GOING TO LEARN

HETEROGENEOUS AND NUMA ARCHITECTURES



Session 2: handling memory organization in Intel Xeon Phi processors

DATA PARALLELISM AND VECTOR INSTRUCTIONS

Vectors – form of SIMD architecture (Single Instruction Multiple Data).

Scalar Instructions

$$\begin{array}{r} 4 + 1 = 5 \\ 0 + 3 = 3 \\ -2 + 8 = 6 \\ 9 + -7 = 2 \end{array}$$

Vector Instructions

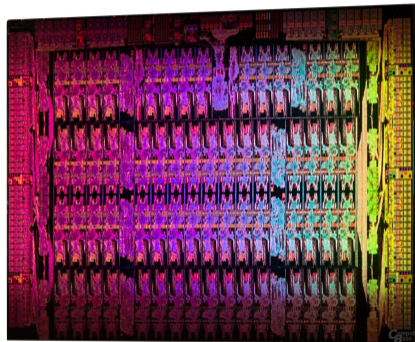
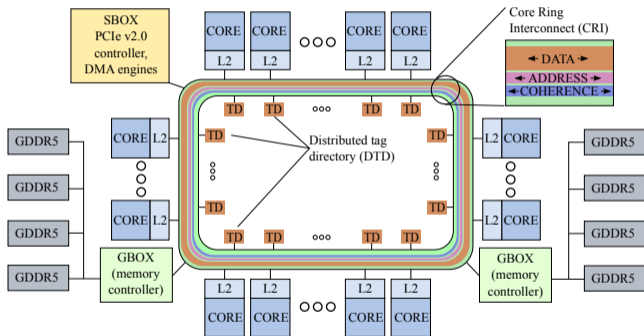
$$\begin{array}{r} 4 \\ 0 \\ -2 \\ 9 \end{array} + \begin{array}{r} 1 \\ 3 \\ 8 \\ -7 \end{array} = \begin{array}{r} 5 \\ 3 \\ 6 \\ 2 \end{array}$$

↑
Vector Length
↓

Session 3: automatic vectorization with Intel compilers

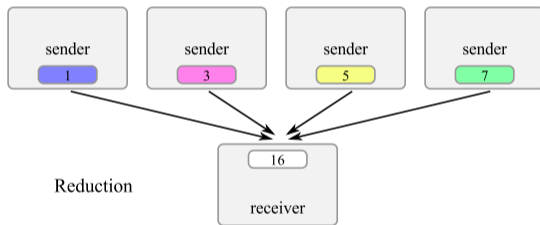
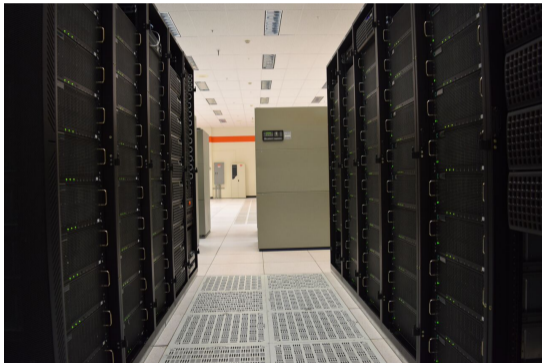
TASK PARALLELISM AND CORES

Cores implement MIMD (Multiple Instruction Multiple Data) arch



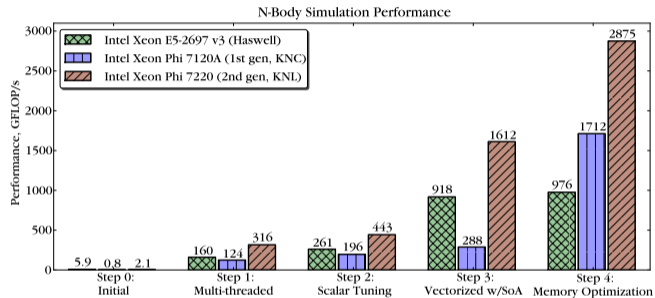
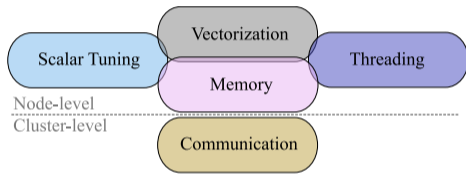
Session 4: multi-threading with OpenMP

Clusters form distributed-memory systems with network interconnects



Session 5: Message Passing Interface (MPI)

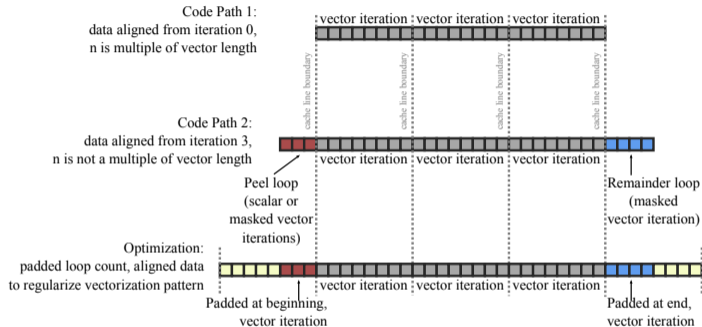
OPTIMIZATION OVERVIEW



Session 6: optimization overview, case study

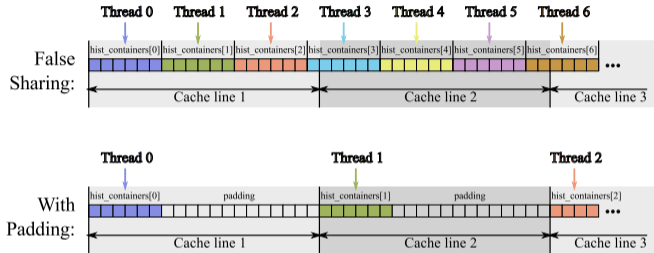
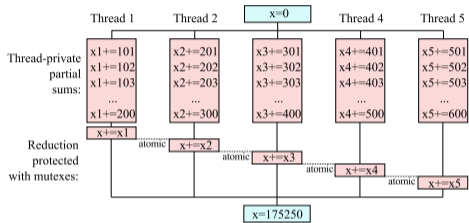
SCALAR TUNING, OPTIMIZATION OF VECTORIZATION

```
for (i = 0; i < n; i++) A[i] = ...
```



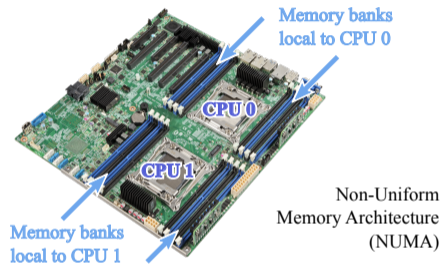
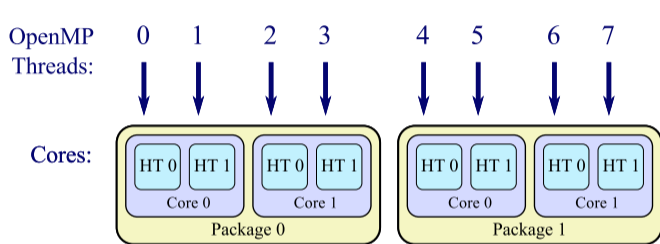
Session 7: precision control, regularizing vectorization patterns

COMMON ISSUES IN MULTI-THREADING



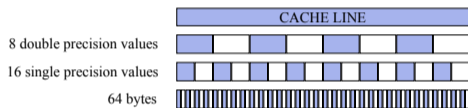
Session 8: minimizing synchronization, avoiding false sharing, strip-mining for parallelism

MULTI-THREADING, MEMORY ASPECT



Session 9: thread affinity, NUMA locality, scheduling

CACHE AND MEMORY ACCESS



Tiling

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Cache-Oblivious Recursion

1	3	9	11
2	4	10	12
5	7	13	15
6	8	14	16

Session 10: loop transformations for locality, bandwidth secrets



§5. HANDS-ON DEMONSTRATION

ACCESS THE COLFAX CLUSTER


[Home](#)
[Learn](#)
[Connect](#)
[Program](#)
[Compute](#)
[Log Out](#)

Welcome to Colfax Cluster!

Learn



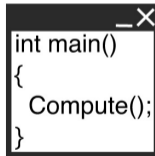
what to expect on
the Colfax Cluster

Connect



from your home
computer to the
cluster

Program



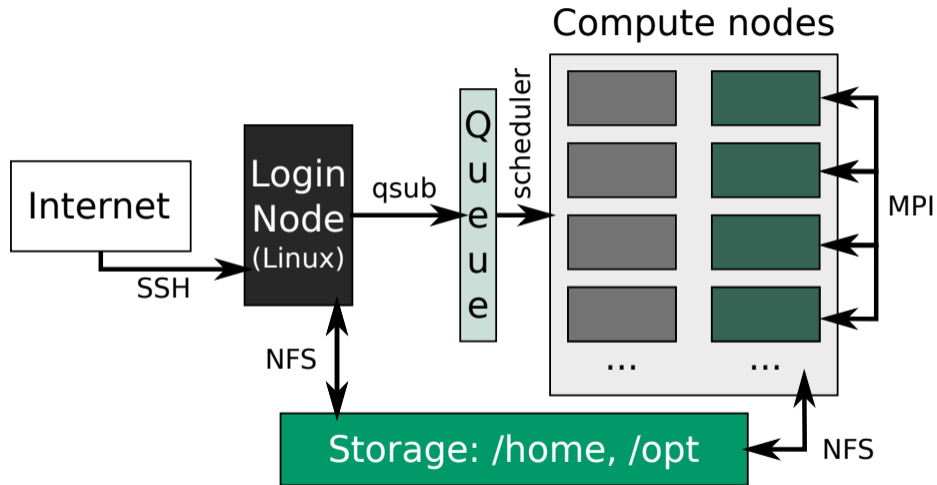
using modern code
practices

Compute



with cluster job
management tools

JOB MANAGEMENT IN THE CLUSTER



DOWNLOAD LABS

```
[u111@c005 ~]% git clone https://github.com/ColfaxResearch/HOW-Series-Labs.git
Cloning into 'HOW-Series-Labs'...
[u111@c005 ~]% ls HOW-Series-Labs/*/
HOW-Series-Labs/2/:
2.01-native-basic      2.03-offload-basic      2.05-shared-virtual-memory-basic
2.02-native-MPI       2.04-offload-asynchronous  2.06-shared-virtual-memory-complex-

HOW-Series-Labs/3/:
3.01-vectorization    3.03-OpenMP-reduction    3.05-Cilk-Plus-basics    3.07-Cilk-Plu
3.02-OpenMP-basics    3.04-OpenMP-tasks        3.06-Cilk-Plus-reducers  3.08-MPI-basi

HOW-Series-Labs/4/:
4.01-overview-nbody          4.07-threading-affinity
4.02-vectorization-data-structures-coulomb  4.08-memory-tiling-matrix_x_vector
4.03-vectorization-tuning-lu-decomposition  4.09-memory-loop-fusion-statistics
4.04-threading-misc-histogram                4.10-offload-double-buffering-dgem
...
```

REVIEW AND WHAT'S NEXT

- ▶ Computers are getting faster through parallelism and specialization
- ▶ Intel Xeon E5 product family – general-purpose parallel processors
- ▶ Intel Xeon Phi product family – specialized parallel processors
- ▶ Coprocessor – either offload device or an additional compute node

Next session: details of Intel Xeon Phi processor and coprocessor programming.