Intel Xeon Phi
From hard to soft

Intel Xeon Phi
16-17 Avril-2013

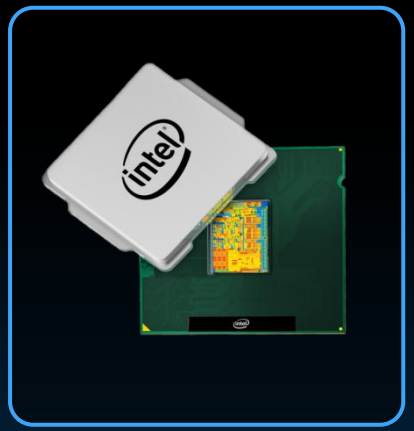
Alain Dominguez
Intel

Agenda

- Architecture and Platform overview
- General environment, management tools and settings
- Intel associated software development tools
- Execution and Programming model choice
- Algorithm and Performance extraction
- Summary and questions

Architecture and Platform overview

Intel® Xeon® Processor

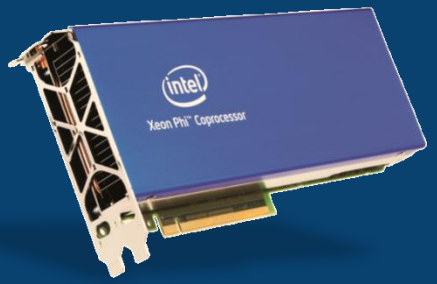


General HPC Workloads

Intel® Xeon Phi™ Coprocessor

Highly-Parallel HPC Workloads

www.intel.com/xeonphi



Intel® Xeon Phi™
Coprocessor 5110P

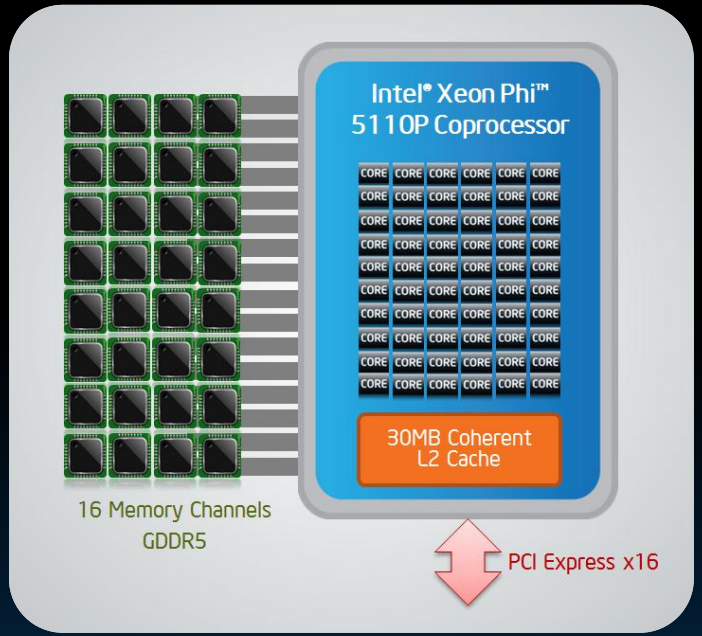
60 Cores, 240 Threads
1.053 GHz
512-bit SIMD instructions
1.01 TFLOPS DP-F.P. peak
8GB GDDR5 Memory, 320 GB/s
PCIe* x16
225W TDP (card)

22nm with the world's first
3-D Tri-Gate transistors
Linux* operating system
IP addressable
Common x86/IA
Programming Models and SW-Tools

Common Architectural Characteristics

Intel® Xeon® Processor E5-2690		Intel® Xeon Phi™ 5110P Coprocessor
2.9GHz	FREQUENCY	1.053GHz
8 (Multi-Core)	CORES	60 (Many-Core)
16	THREADS	240
256-bit	SIMD	512-bit
Cache Coherent	CACHE	Cache Coherent
Shared Memory	MEMORY	Shared Memory

Intel® Xeon Phi™ Coprocessor Overview

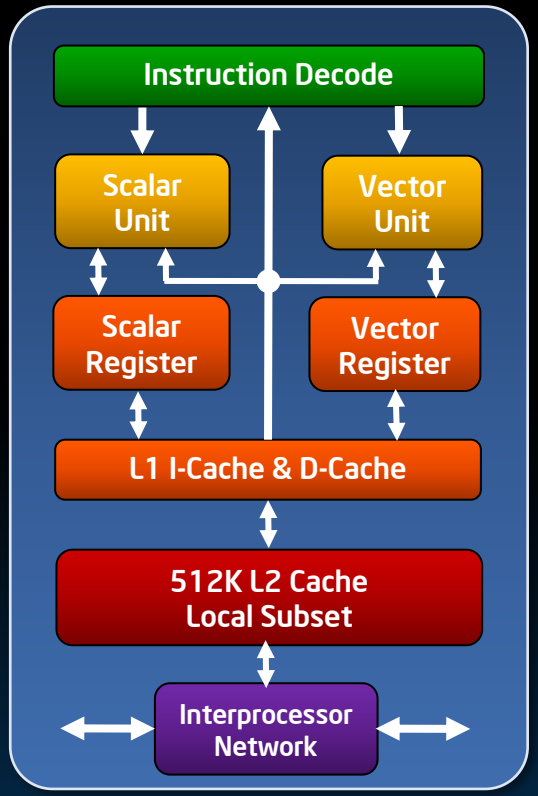


Standard IA Shared Memory Programming

60 cores (240 threads)
1.053GHz
1.01 TFLOPS DP-F.P. peak performance
Advanced VPU per core (512-bit SIMD)
30MB common coherent L2 cache
16 memory channels
320GB/s peak memory bandwidth
8GB GDDR5 memory capacity
PCIe x16 host interface card

Future options subject to change without notice.

Intel® Xeon Phi™ Coprocessor Core



Intel® Xeon Phi™ co-processor core:

- Pipeline derived from the dual-issue Pentium processor
- Short execution pipeline
- Fully coherent cache structure
- Significant modern enhancements
  - such as multi-threading, 64-bit extensions, and sophisticated pre-fetching.
- 4 execution threads per core
- 32KB instruction cache and 32KB data cache for each core.

Enhanced instructions set with:

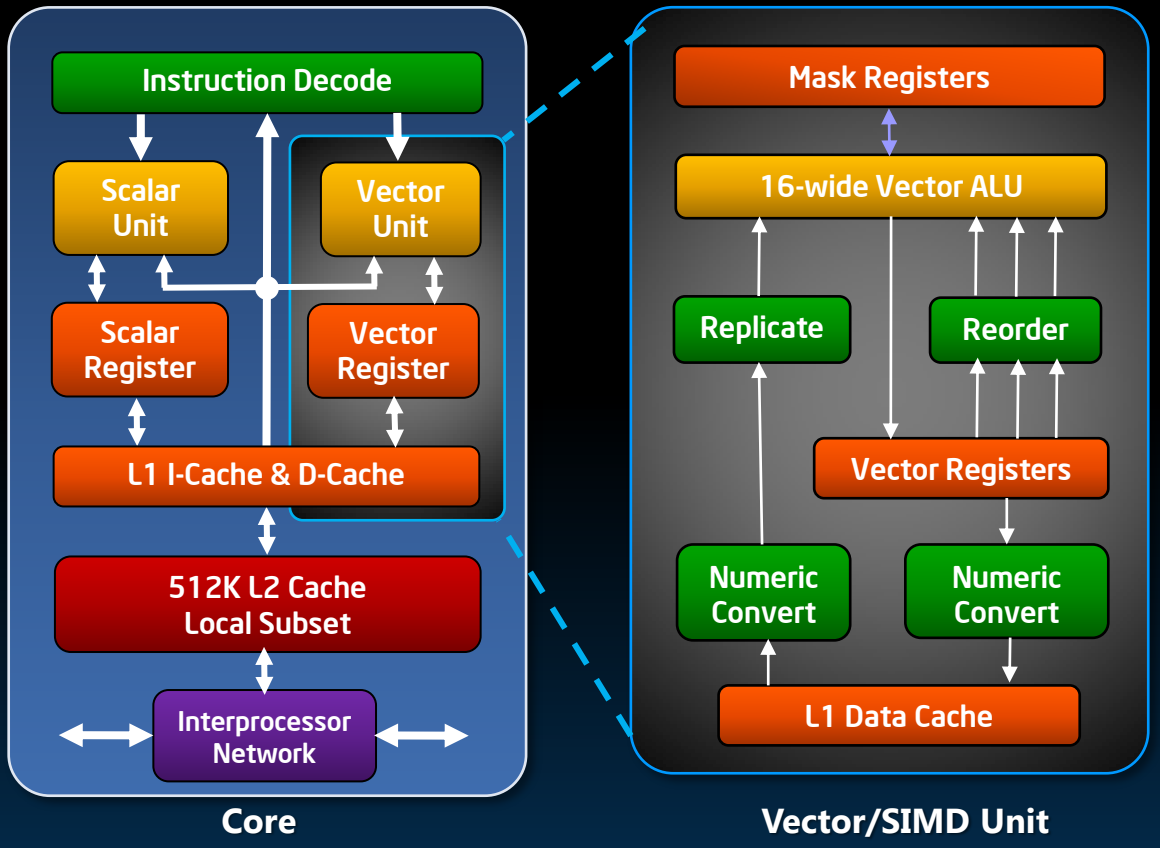
- Over 100 new instructions
- Some specialized scalar instructions
- 3-operand, source non-destructive instruction
- Supports IEEE 754 2008 for floating point arithmetic

Interprocessor Network

1024 bits wide, bi-directional (512 bits in each direction)

Future options subject to change without notice.

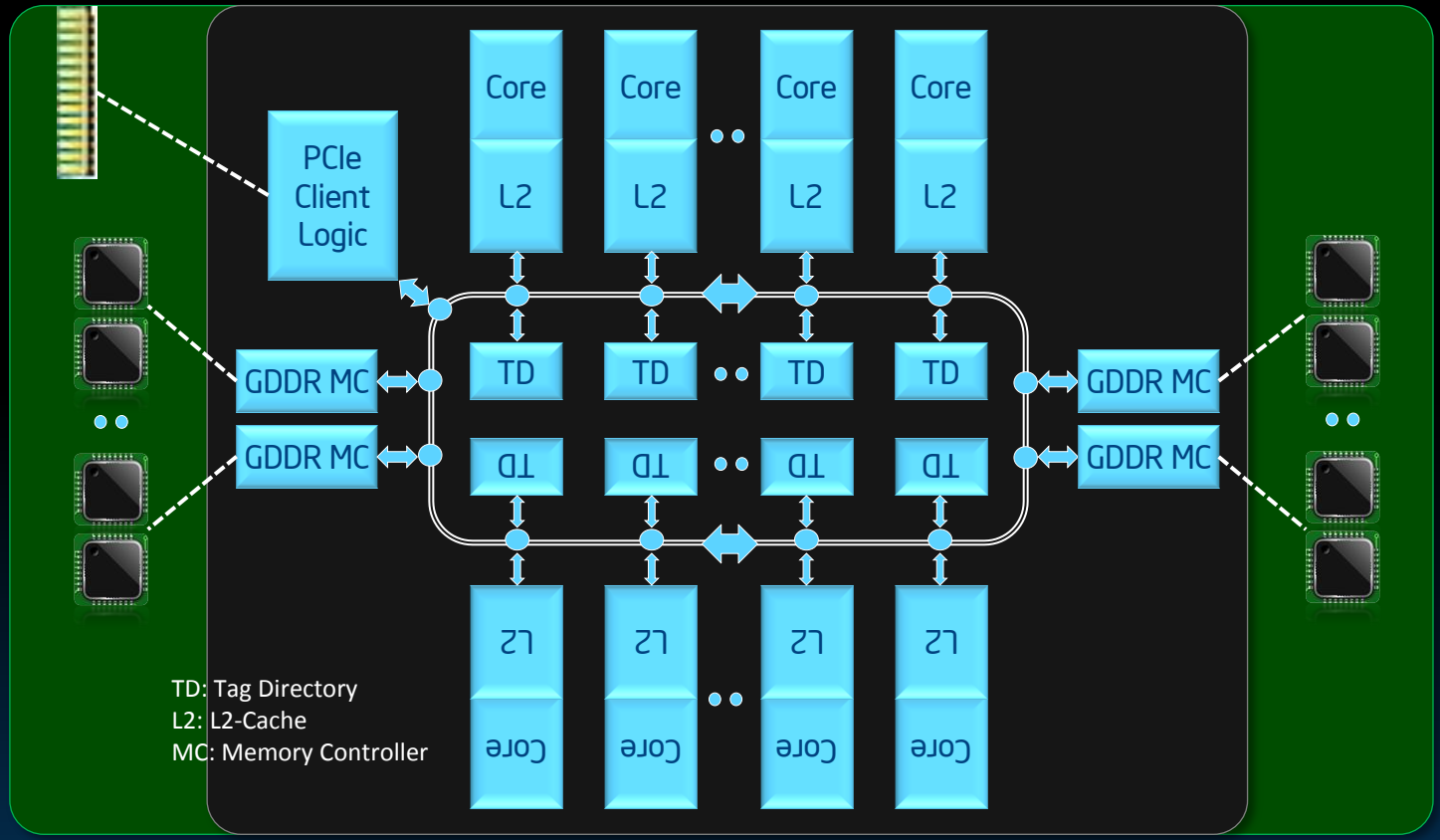
Vector/SIMD High Computational Density



- 32 novel 512-bit SIMD instruction set , officially known as Intel® Initial Many Core Instructions (Intel® IMCI)
- VPU can execute 16 single-precision (SP) or 8 double-precision (DP) operations per cycle
- VPU also supports Fused Multiply-Add (FMA) instructions and hence can execute 32 SP or 16 DP floating point operations per cycle
- 8 mask register
- VPU also supports gather and scatter instructions (non-unit stride vector memory accesses) directly in hardware.
- VPU also features an Extended Math Unit (EMU) : hardware transcendental acceleration such as reciprocal, square root, and log.

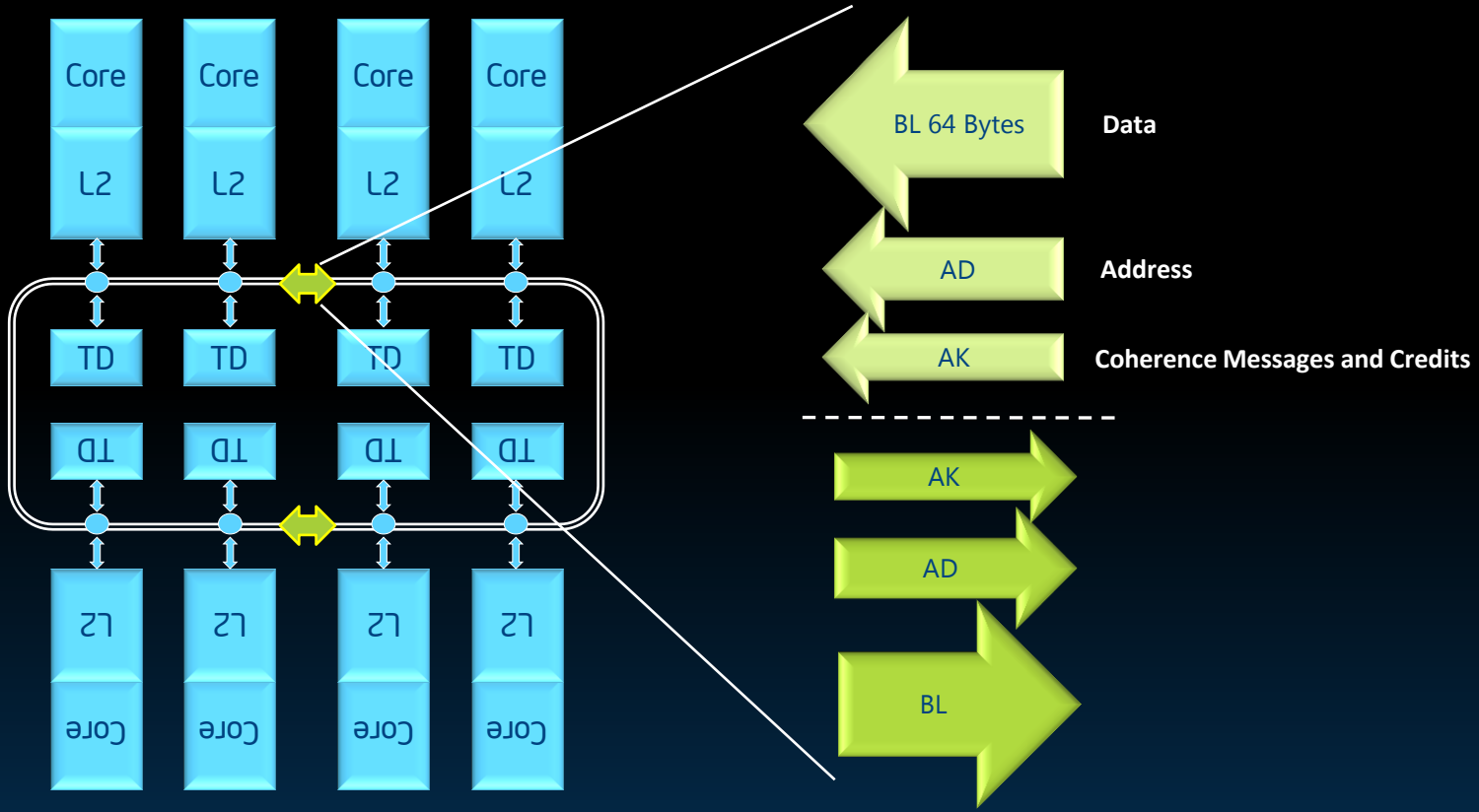
Future options subject to change without notice.

Intel® Xeon Phi™ Coprocessor Microarchitecture Overview



For illustration only.

Terascale On-Chip Interconnect



For illustration only.

Xeon PHI summary

~60 in-order cores, ~1GHz

Teraflops Platform !

4 hardware threads per core

Two pipelines

Pentium® processor family-based scalar units

Fully-coherent L1 and L2 caches

64-bit addressing

All new vector unit

512-bit SIMD Instructions – not Intel® SSE, MMX™, or Intel® AVX

32 X 512-bit wide vector registers

Hold 16 singles or 8 doubles per register

Pipelined one-per-clock throughput

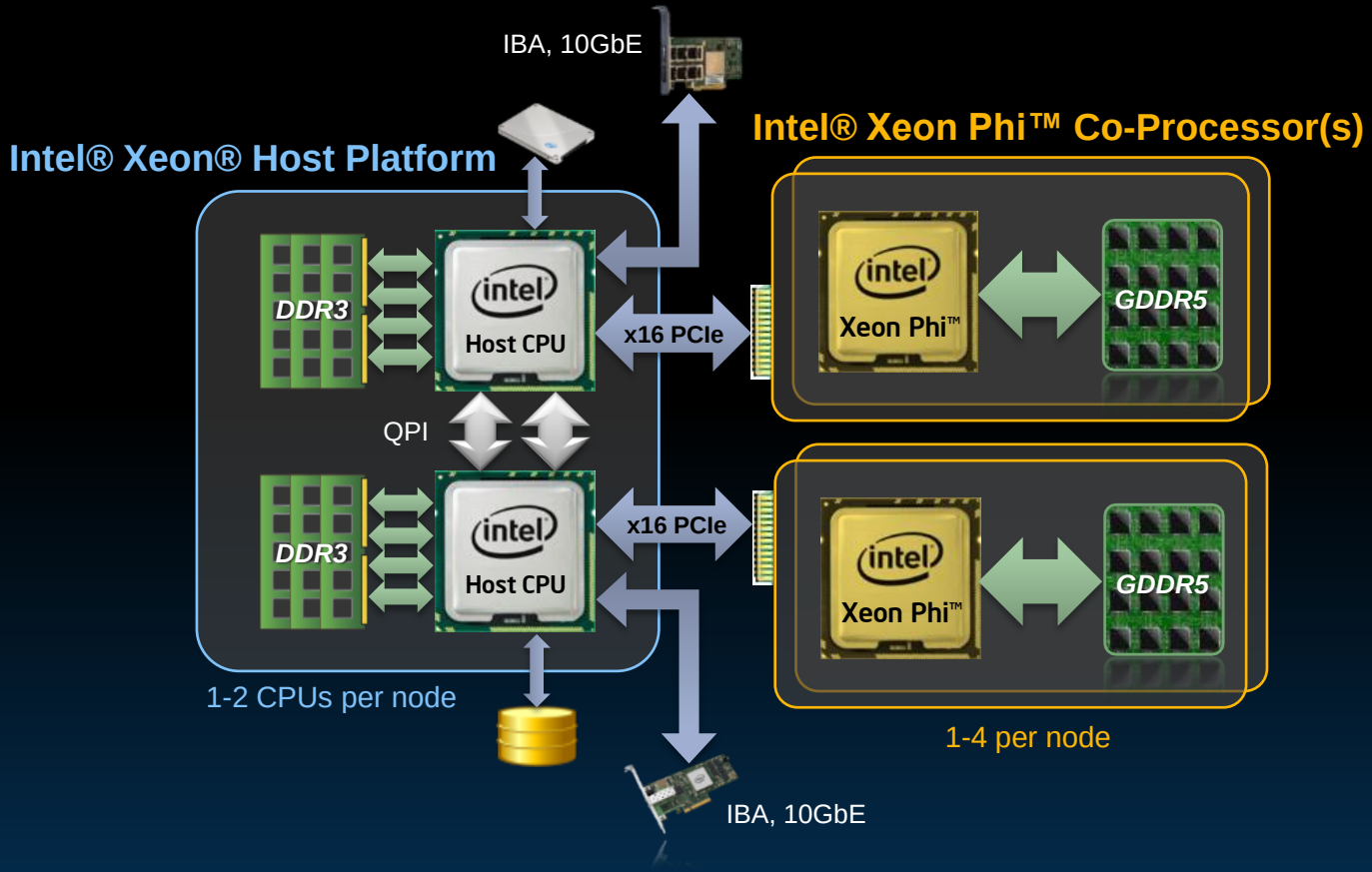
4 clock latency, hidden by round-robin scheduling of threads

Dual issue with scalar instructions

GDDR5 (6/8 GB)

High Bandwitdh : > 320 GB/sec

Typical Platform with Intel® Xeon Phi Coprocessor



For illustration only.

General environment, management tools and settings

- Once Xeon PHI plugged on your PCIe
- Download MicPlatformSoftwareStack
- Install from host a Linux image and boot via « service mpss start »

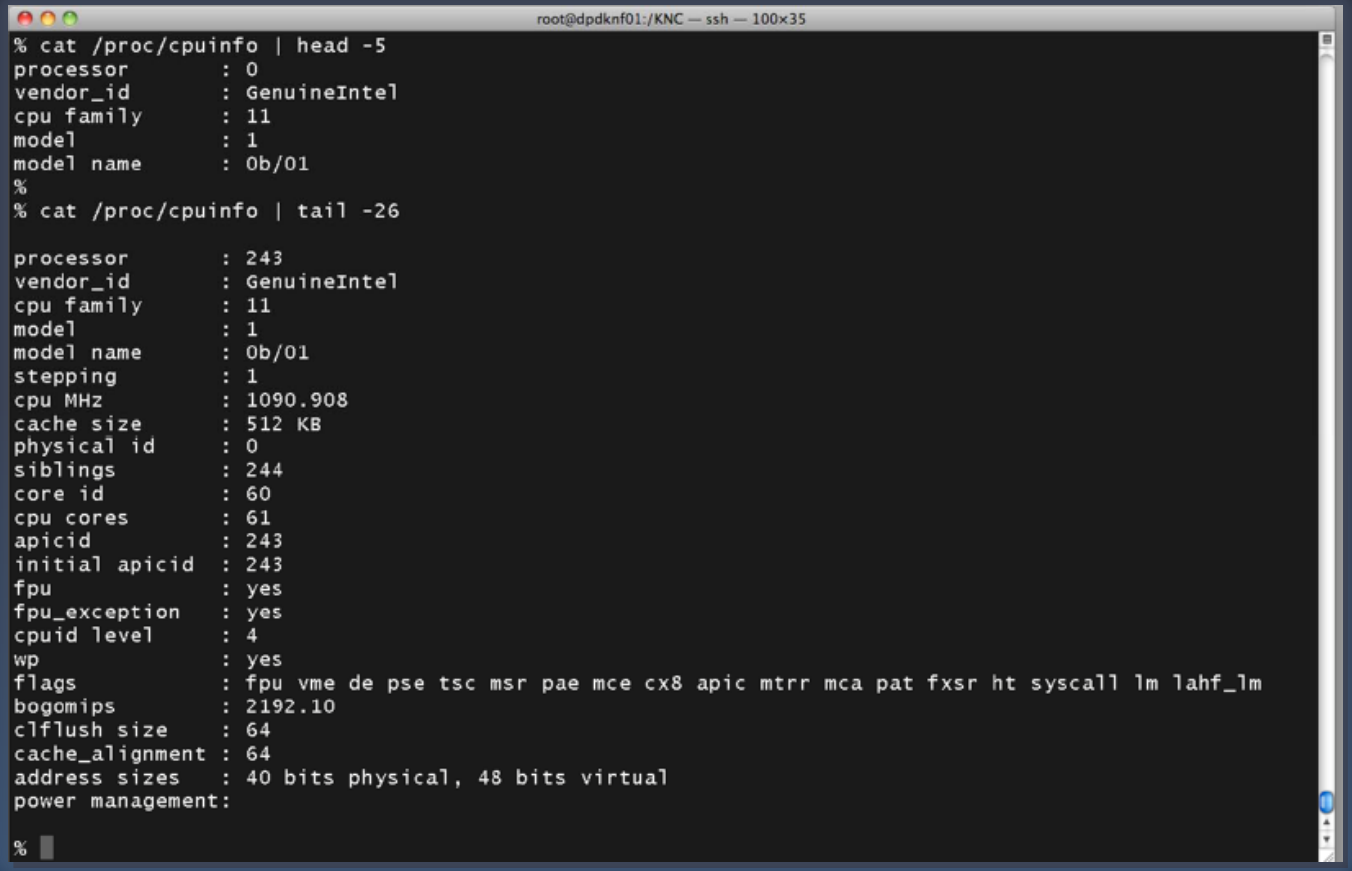
➔ you've now 2 (Linux) connected machines

➔ IP addressable

- Set Xeon PHI ‘BIOS’ features (Turbo,ECC,PC states,…)
- As Phi is diskless, set on host (/opt/intel/mic …) users, ssh keys, file system mods for boot, etc ..
- Good idea : prepare NFS mount between Host and Phi
- Other interaction command in /opt/intel/mic/bin:
  - micsmc : core/card status and usage, temperature,…
  - micinfo : card hardware info, software driver version,…
  - …

➔ **You’re ready to work with Xeon Phi.**

Well, it is an SMP-on-a-chip running Linux*



Intel® Xeon Phi™ Coprocessor Platform Status Panel

Cards Advanced Cards: 1 Memory: 8 GB Cores: 61

Utilization View (All Devices)

System [%]
User [%]

Average Core Utilization

60

Average Core Temperature °C

3.174 GB
Total Memory Usage

106 Watts
Total Power Usage

mic0: Core Histogram View

Individual Core Usage

System [%]
User [%]

System Monitor

Monitor Edit View Help

System Processes Resources File Systems

CPU History

CPU1 0.0% CPU2 2.1% CPU3 7.0% CPU4 5.8%
CPU5 2.0% CPU6 0.0% CPU7 6.1% CPU8 0.0%

adomingu@knfsdv:~/Documents/Supersonic/synthetic

File Edit View Search Terminal Help

```
adomingu/ micuser/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/filesystem/mic0/home/
adomingu/ micuser/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/filesystem/mic0/home/
adomingu/ micuser/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/filesystem/mic0/home/
adomingu/ micuser/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/filesystem/mic0/home/
adomingu/ micuser/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/filesystem/mic0/home/
adomingu/ micuser/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/
bin/        docs/       flash/      lib/        LICENSE/    scif/       share/      tutorials/
coi/        filesystem/ include/    lib64/      myo/        sep3.8/     sysmgmt/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/sysmgmt/
docs/ sdk/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/sysmgmt/
docs/ sdk/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/sysmgmt/docs/
de_DE/ en_US/ es_ES/ fr_FR/ it_IT/ ja_JP/ ko_KR/ zh_CN/ zh_TW/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/sysmgmt/docs/
de_DE/ en_US/ es_ES/ fr_FR/ it_IT/ ja_JP/ ko_KR/ zh_CN/ zh_TW/
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/sysmgmt/docs/fr_FR/Intel\(R\)_Xeon_Phi\(TM\)_Coprocessor_Platform_Status_Pa
Intel(R)_Xeon_Phi(TM)_Coprocessor_Platform_Status_Panel_User_Guide.docx
Intel(R)_Xeon_Phi(TM)_Coprocessor_Platform_Status_Panel_User_Guide_files/
Intel(R)_Xeon_Phi(TM)_Coprocessor_Platform_Status_Panel_User_Guide.htm
Intel(R)_Xeon_Phi(TM)_Coprocessor_Platform_Status_Panel_User_Guide.pdf
[adomingu@knfsdv synthetic]$ ls /opt/intel/mic/sysmgmt/docs/fr_FR/Intel\(R\)_Xeon_Phi\(TM\)_Coprocessor_Platform_Status_Pa
nel_User_Guidels
ls: cannot access /opt/intel/mic/sysmgmt/docs/fr_FR/Intel(R)_Xeon_Phi(TM)_Coprocessor_Platform_Status_Panel_User_Guidels:
No such file or directory
[adomingu@knfsdv synthetic]$
```

adomingu@knfsdv:~/Desktop

```
| # | ZZZZZZ X=phiComp | 0.129496 | 0.0129496 | 0.0105805 | 0.0180557 | 0.002202 | 10 |
[adomingu@knfsdv-mic0 EDF]$ ./mainMICFastOpt 300 16 10 244
Snoopy Bench
main.cxx [24] : nx=300
main.cxx [25] : sn order=16
main.cxx [26] : macro cell size =10
main.cxx [27] : nthread=244
main.cxx [72] : GFlops=100.0432217
main.cxx [75] : Legolas::norm2(X)=86458.51781
|    | Name | Total Time (s) | Average Time | Min Time | Max Time | RMS | Calls |
|---+----+--------------+-------------+----------+----------+-----+------- |
| / | <> |              <>  |          <  |          |          | >   |  <>   |
| # | ZZZZ SpatialStreamMatrixSolver::solve total | 19.4316 | 19.4316 | 19.4316 | 19.4316 | 19.431
6 | -nan | 1 |
| # | ZZZZZZ Solve para | 19.309 | 1.9309 | 1.9009 | 2.06478 | 0.0459846 | 10 |
| # | ZZZZZZ X=phiComp | 0.121439 | 0.0121439 | 0.0101921 | 0.0174053 | 0.00194723 | 10
|
[adomingu@knfsdv-mic0 EDF]$ pwd
/home/adomingu/SHMIC/EDF
[adomingu@knfsdv-mic0 EDF]$ ls
libtbb.so.2     mainMIC     mainMICFast     mainMICFastOpt  run             run.sh
[adomingu@knfsdv-mic0 EDF]$ ./mainMICFastOpt 300 16 10 244
Snoopy Bench
main.cxx [24] : nx=300
main.cxx [25] : sn order=16
main.cxx [26] : macro cell size =10
main.cxx [27] : nthread=244
main.cxx [72] : GFlops=101.0740183
main.cxx [75] : Legolas::norm2(X)=86458.51781
|    | Name | Total Time (s) | Average Time | Min Time | Max Time | RMS | Calls |
|---+----+--------------+-------------+----------+----------+-----+------- |
| / | <> |              <>  |          <  |          |          | >   |  <>   |
| # | ZZZZ SpatialStreamMatrixSolver::solve total | 19.2334 | 19.2334 | 19.2334 | 19.2334 | 19.2334 | -na
1 |
| # | ZZZZZZ Solve para | 19.1135 | 1.91135 | 1.88826 | 1.9922 | 0.0297455 | 10 |
| # | ZZZZZZ X=phiComp | 0.118922 | 0.0118922 | 0.0107864 | 0.0129631 | 0.000735021 | 10 |
[adomingu@knfsdv-mic0 EDF]$ ./mainMICFastOpt 300 16 10 244
Snoopy Bench
main.cxx [24] : nx=300
main.cxx [25] : sn order=16
main.cxx [26] : macro cell size =10
main.cxx [27] : nthread=244
```

OK that sounds good …

But you've not yet run a line of your application

➔ Let's see Intel associated software development tools

Software Development Environment for Intel® Xeon Phi™ Coprocessors

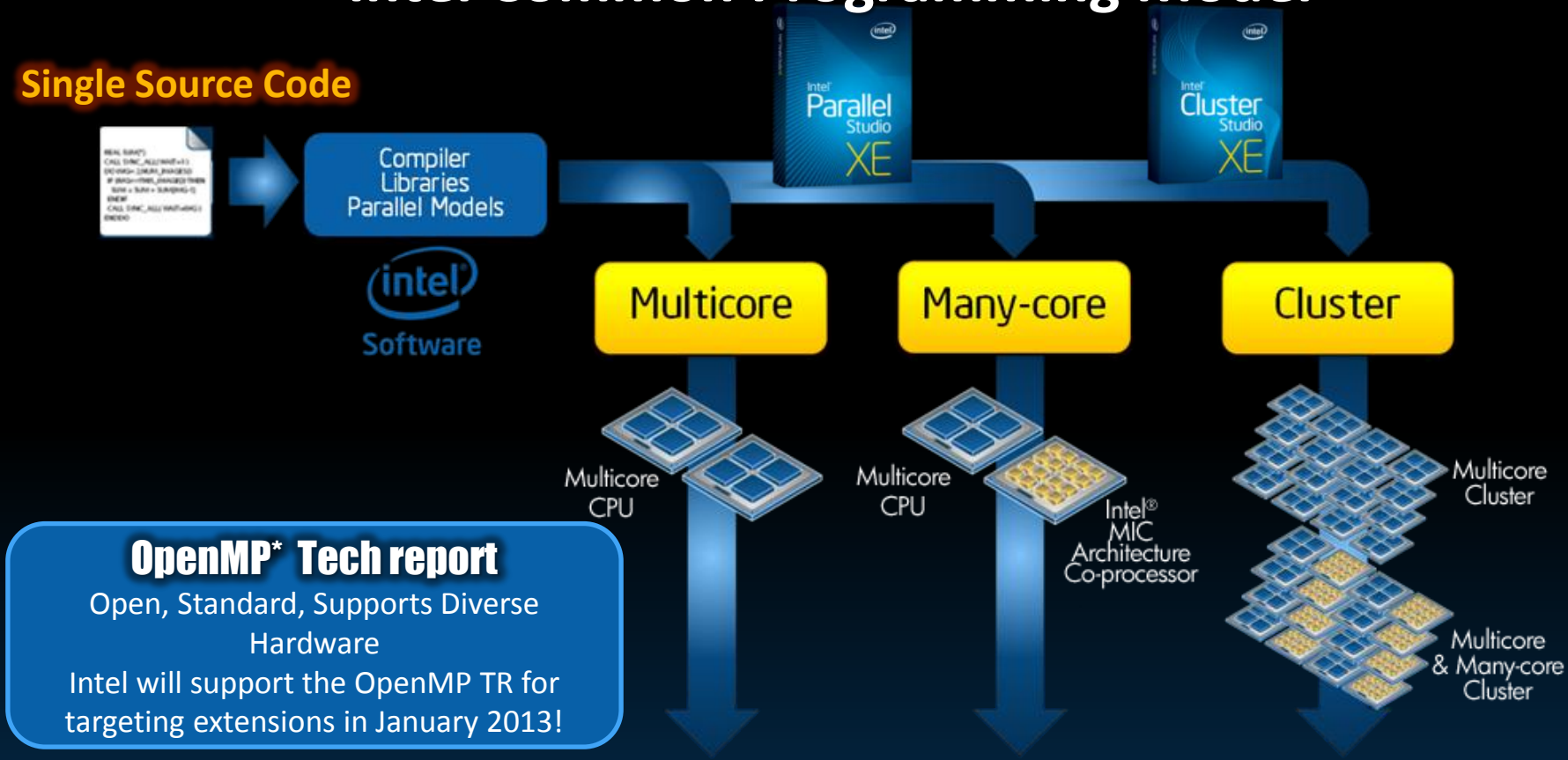
	Open Source	Commercial
Compilers, Run Environments	**gcc** (kernel build only, not for applications), python	**Intel® C++ Compiler, Intel® Fortran Compiler, MYO** CAPS* HMPP* compiler, ScaleMP*
Debugger	**gdb**	**Intel Debugger** RogueWave* TotalView*, Allinea* DDT
Libraries	**TBB[1],** MPICH2, FFTW, NetCDF	NAG*, **Intel® MKL, Intel® MPI, OpenMP*** (in Intel compilers), **Cilk™ Plus** (in Intel compilers), Coarray Fortran (Intel), Rogue Wave* IMSL, Intel® IPP
Profiling & Analysis Tools		**Intel® Vtune™ Amplifier XE,** Intel® Trace Analyzer & Collector, Intel® Inspector XE
Workload Scheduler		Altair* PBS Professional, Adaptive* Computing Moab

[1] – These are all announced. Intel has said there are more actively being developed but are not yet announced. Those in **BOLD** are available as of June 2012.
[2] – Commercial support of TBB available from Intel.

See software.intel.com for software product information

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

Intel Common Programming Model



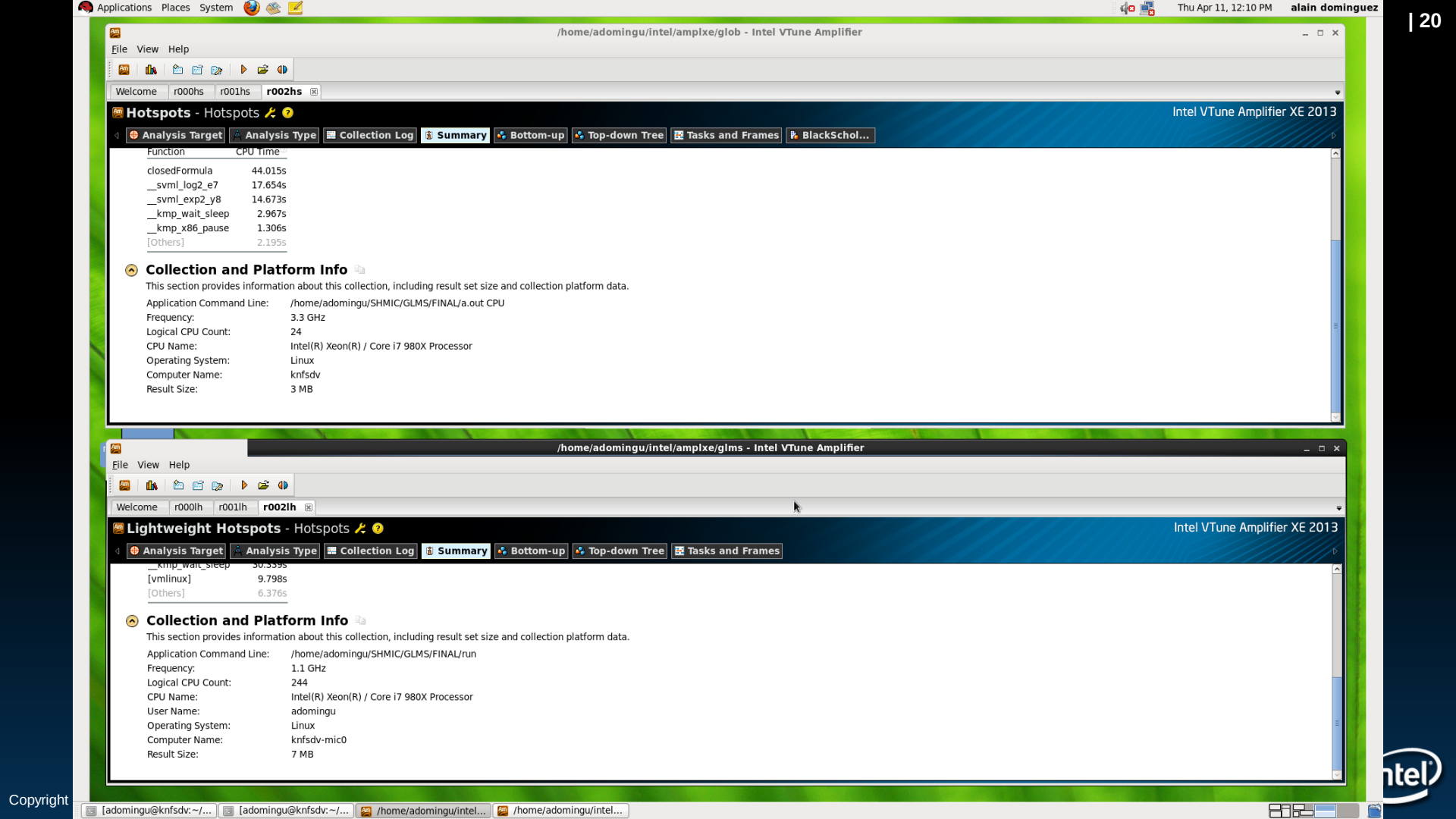
OpenMP* Tech report

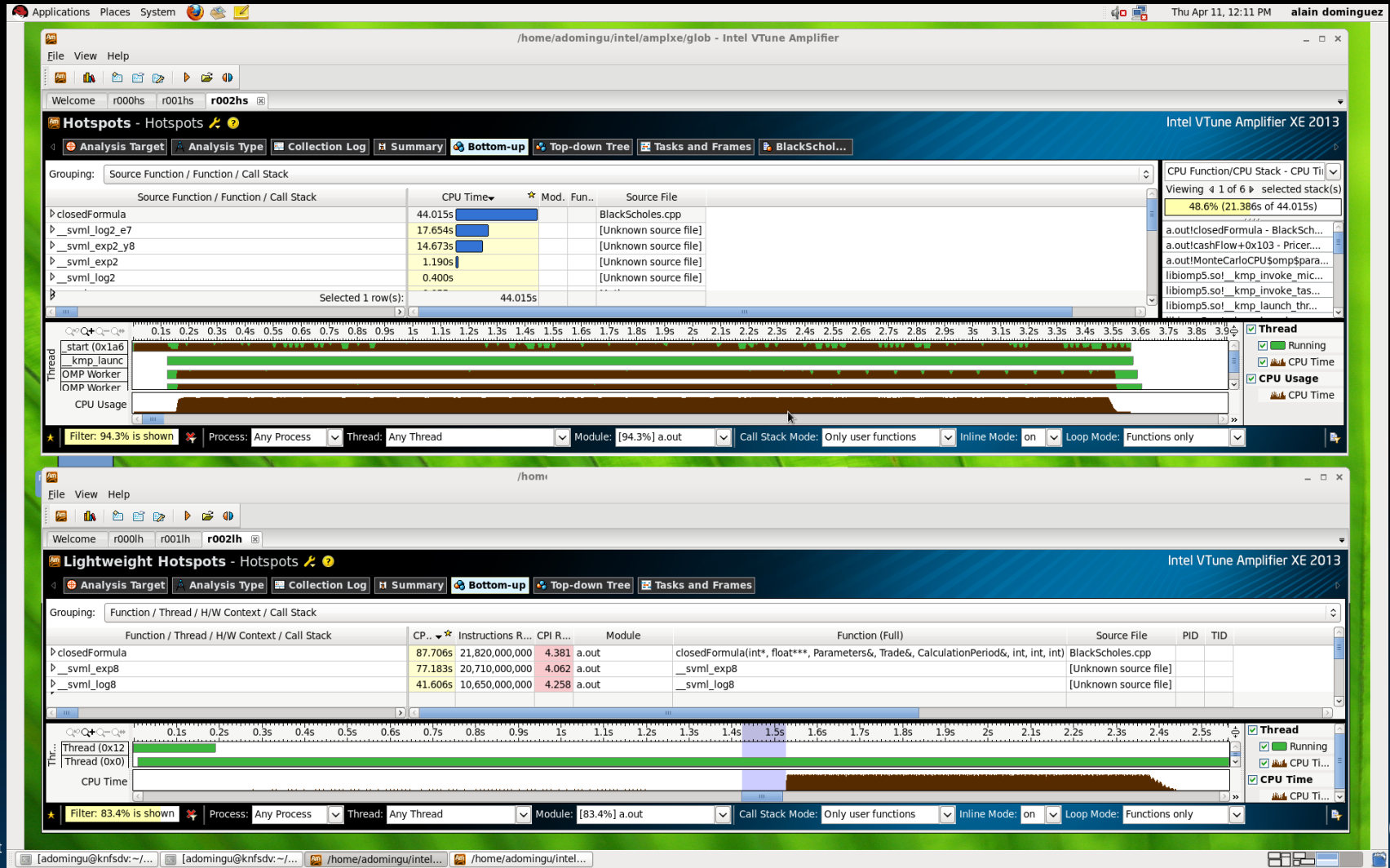
Open, Standard, Supports Diverse Hardware

Intel will support the OpenMP TR for targeting extensions in January 2013!

Eliminate Need for Dual Programming Software Architecture

For illustration only, potential future options subject to change without notice.

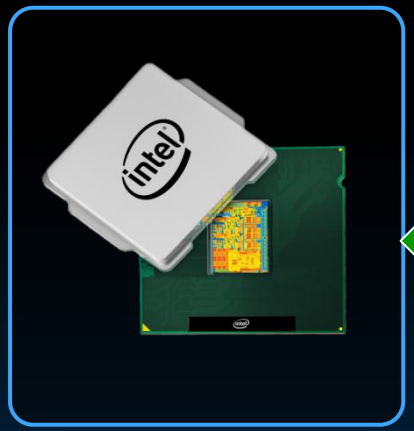

Applications Places System
Thu Apr 11, 12:10 PM alain dominguez
/home/adomingu/intel/amplxe/glob - Intel VTune Amplifier
File View Help
Welcome r000hs r001hs r002hs
Hotspots - Hotspots
Intel VTune Amplifier XE 2013
Analysis Target Analysis Type Collection Log Summary Bottom-up Top-down Tree Tasks and Frames BlackSchol...
Function CPU Time
closedFormula 44.015s
__svml_log2_e7 17.654s
__svml_exp2_y8 14.673s
__kmp_wait_sleep 2.967s
__kmp_x86_pause 1.306s
[Others] 2.195s
Collection and Platform Info
This section provides information about this collection, including result set size and collection platform data.
Application Command Line: /home/adomingu/SHMIC/GLMS/FINAL/a.out CPU
Frequency: 3.3 GHz
Logical CPU Count: 24
CPU Name: Intel(R) Xeon(R) / Core i7 980X Processor
Operating System: Linux
Computer Name: knfsdv
Result Size: 3 MB
/home/adomingu/intel/amplxe/glms - Intel VTune Amplifier
File View Help
Welcome r000lh r001lh r002lh
Lightweight Hotspots - Hotspots
Intel VTune Amplifier XE 2013
Analysis Target Analysis Type Collection Log Summary Bottom-up Top-down Tree Tasks and Frames
[vmlinux] 9.798s
[Others] 6.376s
Collection and Platform Info
This section provides information about this collection, including result set size and collection platform data.
Application Command Line: /home/adomingu/SHMIC/GLMS/FINAL/run
Frequency: 1.1 GHz
Logical CPU Count: 244
CPU Name: Intel(R) Xeon(R) / Core i7 980X Processor
User Name: adomingu
Operating System: Linux
Computer Name: knfsdv-mic0
Result Size: 7 MB
[adomingu@knfsdv:~/...
[adomingu@knfsdv:~/...
/home/adomingu/intel...
/home/adomingu/intel...


Applications Places System
Thu Apr 11, 12:11 PM alain dominguez
/home/adomingu/intel/amplxe/glob - Intel VTune Amplifier
File View Help
Welcome r000hs r001hs r002hs
Hotspots - Hotspots
Intel VTune Amplifier XE 2013
Analysis Target Analysis Type Collection Log Summary Bottom-up Top-down Tree Tasks and Frames BlackSchol...
Grouping: Source Function / Function / Call Stack
Source Function / Function / Call Stack CPU Time Mod. Fun.. Source File
closedFormula 44.015s BlackScholes.cpp
__svml_log2_e7 17.654s [Unknown source file]
__svml_exp2_y8 14.673s [Unknown source file]
__svml_exp2 1.190s [Unknown source file]
__svml_log2 0.400s [Unknown source file]
Selected 1 row(s): 44.015s
CPU Function/CPU Stack - CPU Ti
Viewing 1 of 6 selected stack(s)
48.6% (21.386s of 44.015s)
a.out!closedFormula - BlackSch...
a.out!cashFlow+0x103 - Pricer....
a.out!MonteCarloCPUomppara...
libiomp5.so!__kmp_invoke_mic...
libiomp5.so!__kmp_invoke_tas...
libiomp5.so!__kmp_launch_thr...
Thread
start (0x1a6
__kmp_launc
OMP Worker
OMP Worker
CPU Usage
Running
CPU Time
CPU Usage
CPU Time
Filter: 94.3% is shown
Process: Any Process
Thread: Any Thread
Module: [94.3%] a.out
Call Stack Mode: Only user functions
Inline Mode: on
Loop Mode: Functions only
/hom
File View Help
Welcome r000lh r001lh r002lh
Lightweight Hotspots - Hotspots
Intel VTune Amplifier XE 2013
Analysis Target Analysis Type Collection Log Summary Bottom-up Top-down Tree Tasks and Frames
Grouping: Function / Thread / H/W Context / Call Stack
Function / Thread / H/W Context / Call Stack CP.. Instructions R... CPI R... Module Function (Full) Source File PID TID
closedFormula 87.706s 21,820,000,000 4.381 a.out closedFormula(int*, float***, Parameters&, Trade&, CalculationPeriod&, int, int, int) BlackScholes.cpp
__svml_exp8 77.183s 20,710,000,000 4.062 a.out __svml_exp8 [Unknown source file]
__svml_log8 41.606s 10,650,000,000 4.258 a.out __svml_log8 [Unknown source file]
Thread (0x12
Thread (0x0)
CPU Time
Thread
Running
CPU Ti...
CPU Time
CPU Ti...
Filter: 83.4% is shown
Process: Any Process
Thread: Any Thread
Module: [83.4%] a.out
Call Stack Mode: Only user functions
Inline Mode: on
Loop Mode: Functions only
[adomingu@knfsdv:~/...
[adomingu@knfsdv:~/...
/home/adomingu/intel...
/home/adomingu/intel...

Execution and Programming model choice

You have a mini heterogeneous cluster !!

Intel® Xeon® Processor



General HPC Workloads

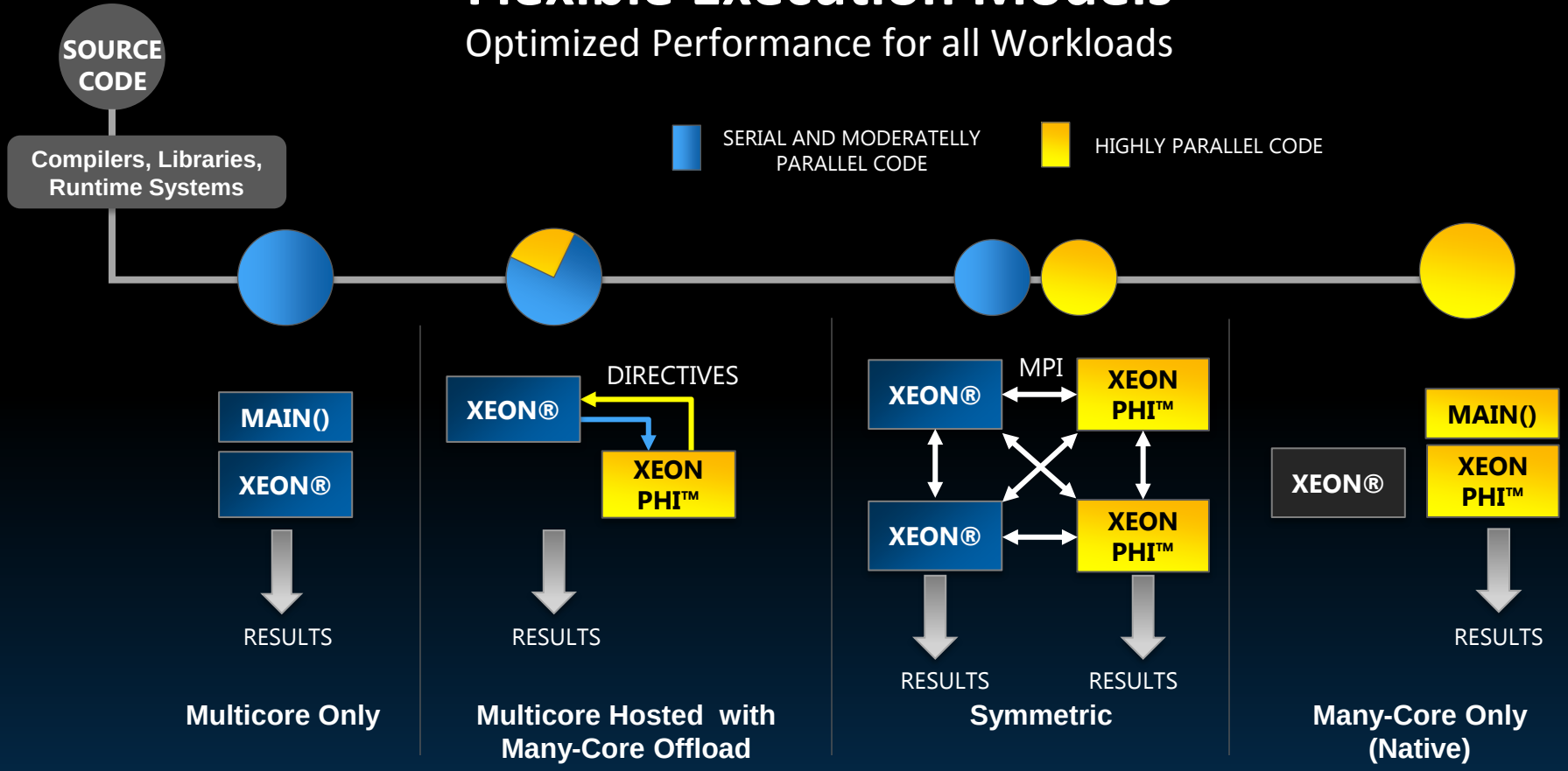
PCIe

Intel® Xeon Phi™ Coprocessor

Highly-Parallel HPC Workloads

Flexible Execution Models

Optimized Performance for all Workloads



Choice of high-performance parallel programming models

Established Standards	Intel® Threading Building Blocks	Domain-Specific Libraries	Intel® Cilk™ Plus	Research and Development
MPI	Widely used C++ template library for parallelism	Intel® Integrated Performance Primitives	C/C++ language extensions to simplify parallelism	Intel® Concurrent Collections
OpenMP*	Open sourced	Intel® Math Kernel Library	Open sourced	Offload Extensions
Coarray Fortran	Also an Intel product		Also an Intel product	Intel® SPMD Parallel Compiler
OpenCL*				
Pthreads				

Native execution model

- Put your original code on host in NFS mounted directory
- Add **-mmic** compiling option in C/C++/Fortran
- Compile it
- Run it in MIC window created with « **ssh mic0** »

➔ You've done your first experiment with Xeon Phi

➔ if it doesn' work :

- ulimit –s unlimited
- copy/create needed library on MIC (LD_LIBRARY_PATH)

Native programming models

- Same as Xeon in general

- MPI, OpenMP, TBB, Cilk, Pthreads or hybrid (MPI/OpenMP)
- Via library usage : MKL , IPP , others

- Ability to use thread pining/affinity as on Xeon :
  - export KMP_AFFINITY (compact, scatter, balanced)
  - export KMP_BLOCKTIME (for thread releasing policy)

Offload execution model

- Co-processor : host drive Xeon Phi
- Compile and run your full application on Xeon
- Programming model and code manage :
  - MIC binary generation, copy, launch and synchronization
  - Data exchanges between devices

- Automatic offload using MKL and MKL_MIC_ENABLE=1

Offload programming models

- OpenCL with new device named ACCELERATOR
- Intel Heterogeneous Compilers : (new/need to learn)

  - Specific directives to manage «offload model » in Intel C/C++/Fortran
  - OpenMP TR 4.0 support
  - Cilk Plus

Offload model basics

	C/C++ Syntax	Semantics
Offload pragma	`#pragma offload <clauses>` `  <statement block>`	Allow next statement block to execute on Intel® MIC Architecture or host CPU
Keyword for variable & function definitions	`__attribute__((target(mic)))`	Compile function for, or allocate variable on, both CPU and Intel® MIC Architecture

Entire blocks of code	`#pragma offload_attribute(push,`
	`  target(mic))`
	`  `
	`#pragma offload_attribute(pop)`
		Mark entire files or large blocks of code for generation on both host CPU and Intel® MIC Architecture

Data transfer	`#pragma offload_transfer target(mic)`	Initiates asynchronous data transfer, or initiates and completes synchronous data transfer

Offload model basics

	Fortran Syntax	Semantics
Offload directive	`!dir$ omp offload <clause> <OpenMP construct>`	Execute next OpenMP* parallel construct on Intel® MIC Architecture
	`!dir$ offload <clauses> <statement>`	Execute next statement (function call) on Intel® MIC Architecture
Keyword for variable/function definitions	`!dir$ attributes offload:<MIC> :: <rtn-name>`	Compile function or variable for CPU and Intel® MIC Architecture
Data transfer	`!dir$ offload_transfer target(mic)`	Initiates asynchronous data transfer, or initiates and completes synchronous data transfer

Offload model basics

Clauses	Syntax	Semantics
Target specification	`target( name[:card_number] )`	Where to run construct
Conditional offload	`if (condition)`	Boolean expression
Inputs	`in(var-list modifiers`$_{opt}$`)`	Copy from host to coprocessor
Outputs	`out(var-list modifiers`$_{opt}$`)`	Copy from coprocessor to host
Inputs & outputs	`inout(var-list modifiers`$_{opt}$`)`	Copy host to coprocessor and back when offload completes
Non-copied data	`nocopy(var-list modifiers`$_{opt}$`)`	Data is local to target
Async. offload	`signal(signal-slot)`	Trigger async offload
Async. offload	`wait(signal-slot)`	Wait for completion

Offload model basics

FORTRAN OPENMP

```
!DIR$ OFFLOAD TARGET(MIC:0)
   CALL ADD_MATRICES(A,B,C)

!DIR$ OFFLOAD TARGET(MIC:0)
   CALL ADD_MATRICES(A,C,D)
```

```
SUBROUTINE ADD_MATRICES(X,Y,Z)

     REAL, DIMENSION(:,:) : X,Y,Z

     Z = X + Y
```

Offload model basics

```
!DIR$ OFFLOAD_TRANSFERT TARGET(MIC:0) IN(A : ALLOC_IF(.TRUE.) FREE_IF(.FALSE.))&
                                      IN(B : ALLOC_IF(.TRUE.) FREE_IF(.FALSE.))&
                                      NOCOPY(C : ALLOC_IF(.TRUE.) FREE_IF(.FALSE.))&
                                      NOCOPY(D : ALLOC_IF(.TRUE.) FREE_IF(.FALSE.))

!DIR$ OFFLOAD TARGET(MIC:0) NOCOPY(A,B,C : ALLOC_IF(.FALSE.) FREE_IF(.FALSE.))
                            CALL ADD_MATRICES(A,B,C)

!DIR$ OFFLOAD TARGET(MIC:0) NOCOPY(A,C,D : ALLOC_IF(.FALSE.) FREE_IF(.FALSE.))
                            CALL ADD_MATRICES(A,C,D)

!DIR$ OFFLOAD_TRANSFERT TARGET(MIC:0)   NOCOPY(A : ALLOC_IF(.FALSE.) FREE_IF(.TRUE.))&
                                       NOCOPY(B : ALLOC_IF(.FALSE.) FREE_IF(.TRUE.))&
                                       NOCOPY(C : ALLOC_IF(.FALSE.) FREE_IF(.TRUE.))&
                                       OUT    (D : ALLOC_IF(.FALSE.) FREE_IF(.TRUE.))
```

Symmetric execution model

- Both Host and MIC work at same time to solve a problem
- « Symmetric » definition is quite restrictive
- I prefer : Host and MIC work at same time to solve a problem.
- Best model for global use of platform performance
- Also best fit for cluster usage

Symmetric programming models

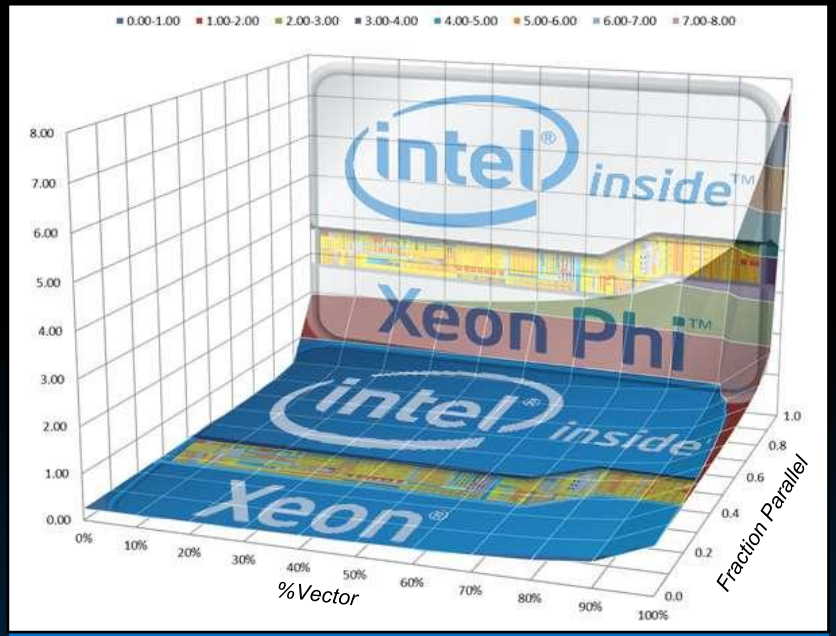
- On 1 SMP node, ability to create « symmetric » execution with previous models (see example)
- For more nodes, MPI and MPI hybrids as MPI/OpenMP, MPI/TBB ..
- MPI hybrids are near mandatory for heterogeneous cluster usage.

Example

```
double __attribue__((target(mic))) myworkload(double input){
   // do something useful here
  return result;}

int main(void){
 //.... Initialize variables
#pragma omp parallel sections
{
#pragma omp section
  {#pragma offload target(mic)
    result1= myworkload(input1); }
#pragma omp section
     result2= myworkload(input2);
 }
}
```

Algorithm and Performance extraction



Theoretical acceleration of a highly parallel processor over a Intel® Xeon® parallel processor (<1Intel® Xeon® faster) – For illustration only

Xeon PHI/Xeon sensitive differences :

Much more cores
➔ High level of parallelism >90%
Much slower cores
➔ Higher level of parallelism >95%
Wider vectors for serial performance
➔ High level of vectorisation >90%
In order execution
➔ Higher level of vectorisation >95%
Largely less cache/core
➔ Data layout and memory access pattern
Memory coherency overhead
➔ Data layout and memory access pattern
More alignment sensitivity
➔ Data layout and memory access pattern

More Cores. Wider Vectors. Performance Delivered.

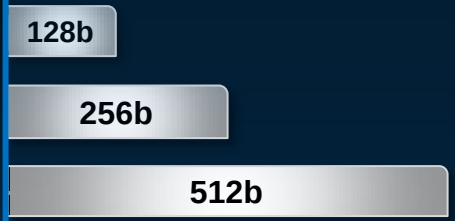
Intel® Parallel Studio XE 2013 and Intel® Cluster Studio XE 2013

More Cores

Multicore (8+)

Many-Core (60)

Wider SIMD/Vector



Scaling Performance Efficiently

Serial

Task & Data Parallel

Distributed

Industry Leading Software Tools

High-Performance from advanced compilers

Comprehensive libraries

Parallel programming models

Insightful analysis tools

Parallel Programming for Intel® Architecture (IA)

CORES	Use threads directly or e.g. via OpenMP* Use tasking, Intel® TBB / Cilk™ Plus
VECTORS	Intrinsics, auto-vectorization, vector-libraries Language extensions for vector programming
BLOCKING	Use caches to hide memory latency Organize memory access for data reuse
DATA LAYOUT	Structure of arrays facilitates vector loads / stores, unit stride Align data for vector accesses

Parallel programming to utilize the hardware resources, in an abstracted way

How to proceed ?

- Use only one node and its standard Xeon to test and calibrate your work
- Use Intel tools to study performance properties on current Xeon
  - Measure parallelization scaling
  - Measure impact of vectorization:
    - Use "–no-vec –no-simd" to disable all compiler vectorization
- Verify memory needed and port to Xeon PHI
- Use node and its standard Xeon transfer with Host versus granularity

➔If OK, you can begin to port on Xeon PHI

How to proceed ? (2)

- Try to work symmetrically with dual experiments on Xeon and MIC, as optimization process is the same, you often improve both

- Intel tools are really helpfull and mandatory to be efficient
- Use Vtune to understand hotspot and verify generated code
- Use Vtune to evaluate MIC loop performance ratio vs Xeon
- Use vec-report3 (or 6) to verify vectorization

Some (standard) hints for performance

- Maximum of unit stride vectorization
- Avoid branchy code inside loops
- Align data and tell the compiler
- Hundreds of threads to manage with 1 Ghz clock increase paralelism overhead :
  - find optimal number of threads
  - reduce barrier, synchronization, locks, critical sections
  - use reduction and minimize parallel regions
  - pin/place and control thread behaviour (OMP_ ,KMP_ , others)

Get Started Now

Download the programming guide to find out whether your workload can benefit from Intel® Xeon Phi™ coprocessors:

Highly recommended reading:

software.intel.com/mic-developer

An Overview of Programming for Intel® Xeon® processors and Intel® Xeon Phi™ coprocessors

Submitted by James Reinders ... on Mon, 11/12/2012 - 12:59

Summary and questions

- Xeon PHI offers highly parallel architecture
  with high aggregate performance and memory bandwidth
- It's a Linux machine in a node to ease management
- Linux OS and X86 architecture make Xeon PHI easily accessible, understandable and programmable.
- Full Intel software tool chain allows Xeon PHI to support vast number of programming models to answer your needs and insure development efficiency and perennity

Legal Disclaimers: Performance

- *Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, Go to: http://www.intel.com/performance/resources/benchmark_limitations.htm.*
- *Intel does not control or audit the design or implementation of third party benchmarks or Web sites referenced in this document. Intel encourages all of its customers to visit the referenced Web sites or others where similar performance benchmarks are reported and confirm whether the referenced benchmarks are accurate and reflect performance of systems available for purchase.*
- *Relative performance is calculated by assigning a baseline value of 1.0 to one benchmark result, and then dividing the actual benchmark result for the baseline platform into each of the specific benchmark results of each of the other platforms, and assigning them a relative performance number that correlates with the performance improvements reported.*
- *SPEC, SPECint, SPECfp, SPECrate. SPECpower, SPECjAppServer, SPECjEnterprise, SPECjbb, SPECompM, SPECompL, and SPEC MPI are trademarks of the Standard Performance Evaluation Corporation. See http://www.spec.org for more information.*
- *TPC Benchmark is a trademark of the Transaction Processing Council. See http://www.tpc.org for more information.*
- *SAP and SAP NetWeaver are the registered trademarks of SAP AG in Germany and in several other countries. See http://www.sap.com/benchmark for more information.*
- *INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.*
- *Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, reference www.intel.com/software/products.*

Legal Disclaimer

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.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

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.

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

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

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

Optimization Notice

Intel® compilers, associated libraries and associated development tools may include or utilize options that optimize for instruction sets that are available in both Intel® and non-Intel microprocessors (for example SIMD instruction sets), but do not optimize equally for non-Intel microprocessors. In addition, certain compiler options for Intel compilers, including some that are not specific to Intel micro-architecture, are reserved for Intel microprocessors. For a detailed description of Intel compiler options, including the instruction sets and specific microprocessors they implicate, please refer to the "Intel® Compiler User and Reference Guides" under "Compiler Options." Many library routines that are part of Intel® compiler products are more highly optimized for Intel microprocessors than for other microprocessors. While the compilers and libraries in Intel® compiler products offer optimizations for both Intel and Intel-compatible microprocessors, depending on the options you select, your code and other factors, you likely will get extra performance on Intel microprocessors.

Intel® compilers, associated libraries and associated development tools 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 Intel® Streaming SIMD Extensions 2 (Intel® SSE2), Intel® Streaming SIMD Extensions 3 (Intel® SSE3), and Supplemental Streaming SIMD Extensions 3 (Intel® SSSE3) instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors.

While Intel believes our compilers and libraries are excellent choices to assist in obtaining the best performance on Intel® and non-Intel microprocessors, Intel recommends that you evaluate other compilers and libraries to determine which best meet your requirements. We hope to win your business by striving to offer the best performance of any compiler or library; please let us know if you find we do not.

Notice revision #20101101

Thank You.