MPI Programming on the Intel® Xeon Phi[™] Coprocessor



Dr.-Ing. Michael Klemm Software and Services Group Intel Corporation (michael.klemm@intel.com)



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Legal Disclaimer & Optimization Notice

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.

Copyright © 2013, Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon, Core, Phi, VTune, and Cilk are trademarks of Intel Corporation in the U.S. and other countries. *Other names and brands may be claimed as the property of others.

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 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. 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



Intel® Xeon Phi[™] Architecture

Agenda

- Overview
- Programming Models
- Hybrid Computing
- Load Balancing



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Agenda

Overview

- Programming Models
- Hybrid Computing
- Load Balancing



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Enabling & Advancing Parallelism (intel) Parallel High Performance Parallel Programming Cluster Intel tools, libraries and parallel models extend to multicore, many-core and heterogeneous computing Compiler ALL SINC, ALL(WAIT=1) 01MG=234,84, INAGESD Libraries THE INVESTIGATION SUM = SUM + SUMPAG-TI Parallel Models Code Inte Cluster **Multicore** Software Multicore Multicore Cluster CPU

Use One Software Architecture Today. Scale Forward Tomorrow.

Intel® Xeon Phi[™] Architecture

Software

Software & Services Group, Developer Relations Division

Enabling & Advancing Parallelism (intel) Parallel Cluster High Performance Parallel Programming studio Intel tools, libraries and parallel models extend to multicore, many-core and heterogeneous computing Compiler LE SINC, ALL/WAT+1) DIMG-234,84, EVAGESD Libraries THE INVESTIGATION SUM = SUM + SUMPAG-TI Parallel Models Code Inte **Multicore** Many-core Cluster Software Multicore Multicore Multicore Cluster CPU CPU Intel® Xeon Phi[™] Coprocessor Multicore & Many -core Cluster

Use One Software Architecture Today. Scale Forward Tomorrow.

Intel® Xeon Phi[™] Architecture

inte

Software

Software & Services Group, Developer Relations Division

Preserve Your Development Investment

Common Tools and Programming Models for Parallelism



Develop Using Parallel Models that Support Heterogeneous Computing

Software

Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Intel® MPI Library Overview

- Intel is a leading vendor of MPI implementations and tools
- Optimized MPI application performance
 - Application-specific tuning
 - Automatic tuning
- Lower latency
 - Industry leading latency
- Interconnect Independence & Runtime Selection
 - Multi-vendor interoperability
 - Performance optimized support for the latest OFED capabilities through DAPL 2.0
- More robust MPI applications
 - Seamless interoperability with Intel® Trace Analyzer and Collector





Intel® Xeon Phi™ Architecture

Levels of communication speed

- Current clusters are not homogenous regarding communication speed:
 - Inter node (Infiniband, Ethernet, etc)
 - Intra node
 - Inter sockets (Quick Path Interconnect)
 - $_{\circ}$ Intra socket
- Two additional levels to come with Intel® Xeon Phi[™] coprocessor:
 - Host-coprocessor communication
 - Inter coprocessor communication



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Intel® MPI Library Architecture & Staging



Software

Selecting network fabrics

- Intel[®] MPI selects automatically the best available network fabric it can find.
 - Use I_MPI_FABRICS to select a different communication device explicitly
- The best fabric is usually based on Infiniband (dapl, ofa) for inter node communication and shared memory for intra node
- Available for Intel(R) Xeon Phi(TM):
 - shm, tcp, ofa, dapl
 - Availability checked in the order shm:dapl, shm:ofa, shm:tcp (intra:inter)
- Set I_MPI_SSHM_SCIF=1 to enable shm fabric between host and Intel® Xeon Phi[™]



Intel® Xeon Phi[™] Architecture

Agenda

- Overview
- Programming Models
- Hybrid Computing
- Load Balancing



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Coprocessor only Programming Model

- MPI ranks on Intel® Xeon Phi[™] (only)
- All messages into/out of coprocessors
- Intel® Cilk[™] Plus, OpenMP*, Intel® Threading Building Blocks, Pthreads used directly within MPI processes



Build Intel[®] Xeon Phi[™] binary using Intel[®] compiler.

Upload the binary to the Intel[®] Xeon Phi[™].

Run instances of the MPI application on Intel[®] Xeon Phi[™] nodes.



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Coprocessor only Programming Model

- MPI ranks on the Intel[®] Xeon Phi[™] coprocessor(s) only
- MPI messages into/out of the coprocessor(s)
- Threading possible
- Build the application for the Intel® Xeon Phi[™] Architecture
 # mpiicc -mmic -o test_hello.MIC test.c
- Upload the coprocessor executable
 - # sudo micput 172.31.1.1 ./test_hello.MIC
 /tmp/test_hello.MIC
 - Remark: If NFS available no explicit uploads required (just copies)!
- Launch the application on the coprocessor from host
 - # mpiexec -n 2 -wdir /tmp -host 172.31.1.1
 /tmp/test_hello.MIC
- Alternatively: login to the coprocessor and execute the already uploaded mpiexec.hydra there!



Intel® Xeon Phi[™] Architecture

Symmetric Programming Model

- MPI ranks on Intel® Xeon Phi[™] Architecture and host CPUs
- Messages to/from any core
- Intel® Cilk[™] Plus, OpenMP*, Intel® Threading Building Blocks, Pthreads* used directly within MPI processes



Heterogeneous network of homogeneous CPUs



Build Intel® 64 and Intel® Xeon Phi[™] binaries by using the resp. compilers targeting Intel® 64 and Intel® Xeon Phi[™].

Upload the Intel® Xeon Phi[™] binary to the Intel® Xeon Phi[™] Architecture.

Run instances of the MPI application on different mixed nodes.



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Symmetric Programming Model

- MPI ranks on the coprocessor(s) and host CPU(s)
- MPI messages into/out of the coprocessor(s) and host CPU(s)
- Threading possible
- Build the application for Intel®64 and the Intel® Xeon Phi[™] Architecture separately
 - # mpiicc -o test_hello test.c
 - # mpiicc -mmic -o test_hello.MIC test.c
- Upload the Intel® Xeon Phi[™] executable
 - # sudo micput 172.31.1.1 ./test_hello.MIC
 /tmp/test_hello.MIC
- Launch the application on the host and the coprocessor from the host

```
# mpiexec -n 2 -host <hostname> ./test_hello : -wdir
/tmp -n 2 -host 172.31.1.1 /tmp/test_hello.MIC
```



Intel® Xeon Phi[™] Architecture

MPI+Offload Programming Model

- MPI ranks on Intel[®]
 Xeon[®] processors (only)
- All messages into/out of host CPUs
- Offload models used to accelerate MPI ranks
- Intel® Cilk[™] Plus, OpenMP*, Intel® Threading Building Blocks, Pthreads* within Intel® Xeon Phi[™]



Homogenous network of heterogeneous nodes



Build Intel[®] 64 executable with included offload by using the Intel[®] 64 compiler.

Run instances of the MPI application on the host, offloading code onto coprocessor.

Advantages of more cores and wider SIMD for certain applications



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

MPI+Offload Programming Model

- MPI ranks on the host CPUs only
- MPI messages into/out of the host CPUs
- Intel[®] Xeon Phi[™] Architecture as an accelerator
- Compile for MPI and internal offload
 # mpiicc -o test test.c
- Latest compiler compiles by default for offloading if offload construct is detected!
 - Switch off by -no-offload flag
 - Previous compilers needed -offload-build flag
- Execute on host(s) as usual
 # mpiexec -n 2 ./test
- MPI processes will offload code for acceleration



Intel® Xeon Phi[™] Architecture

Agenda

- Overview
- Programming Models
- Hybrid Computing
- Load Balancing



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Traditional Cluster Computing

- MPI is »the« portable cluster solution
- Parallel programs use MPI over cores inside the nodes
 - Homogeneous programming model
 - "Easily" portable to different sizes of clusters
 - No threading issues like »False Sharing« (common cache line)
 - Maintenance costs only for one parallelization model



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Traditional Cluster Computing (cont'd)

- Hardware trends
 - Increasing number of cores per node plus cores on coprocessors
 - Increasing number of nodes per cluster
- Consequence: Increasing number of MPI processes per application
- Potential MPI limitations
 - Memory consumption per MPI process, sum exceeds the node memory
 - Limited scalability due to exhausted interconnects (e.g. MPI collectives)
 - Load balancing is often challenging in MPI



Intel® Xeon Phi[™] Architecture

Hybrid Computing

- Combine MPI programming model with threading model
- Overcome MPI limitations by adding threading:
 - Potential memory gains in threaded code
 - Better scalability (e.g. less MPI communication)
 - Threading offers smart load balancing strategies
- Result: Maximize performance by exploitation of hardware (including coprocessors)



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Options for Thread Parallelism



Choice of unified programming to target Intel® Xeon and Intel® Xeon Phi[™] Architecture!



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Intel® MPI Support of Hybrid Codes

- Intel® MPI is strong in mapping control
- Sophisticated default or user controlled
 - I_MPI_PIN_PROCESSOR_LIST for pure MPI
 - For hybrid codes (takes precedence):
 - I_MPI_PIN_DOMAIN =<size>[:<layout>]
- Naturally extends to hybrid codes on Intel[®] Xeon Phi[™]



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Intel® MPI Support of Hybrid Codes

- Define I_MPI_PIN_DOMAIN to split logical processors into nonoverlapping subsets
- Mapping rule: 1 MPI process per 1 domain



Pin OpenMP threads inside the domain with KMP_AFFINITY (or in the code)



Intel® Xeon Phi™ Architecture

Software & Services Group, Developer Relations Division

Intel® MPI Environment Support

- The execution command mpiexec of Intel® MPI reads argument sets from the command line:
 - Sections between ":" define an argument set (also lines in a configfile, but not yet available in Beta)
 - Host, number of nodes, but also environment can be set independently in each argument set
 - # mpiexec -env I_MPI_PIN_DOMAIN 4 -host myXEON ... : -env I_MPI_PIN_DOMAIN 16 -host myMIC
- Adapt the important environment variables to the architecture
 - OMP_NUM_THREADS, KMP_AFFINITY for OpenMP
 - CILK_NWORKERS for Intel® Cilk[™] Plus

Intel® Xeon Phi[™] Architecture

Co-Processor only and Symmetric Support

- Full hybrid support on Intel[®] Xeon from Intel[®] MPI extends to Intel[®] Xeon Phi(TM)
- KMP_AFFINITY=balanced (only on coprocessor) in addition to scatter and compact
- Recommendations:
 - Explicitly control where MPI processes and threads run in a hybrid application (according to threading model and application)
 - Avoid splitting cores among MPI processes, i.e.
 <u>I_MPI_PIN_DOMAIN</u> should be a multiple of 4
 - Try different **KMP_AFFINITY** settings for your application



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

MPI+Offload Support

• Define thread affinity manually per single MPI process (pseudo syntax!):

```
# export OMP NUM THREADS=4
```

. . .

```
# mpiexec -env KMP_AFFINITY=[1-4] -n 1 -host myMIC ... :
```

```
-env KMP_AFFINITY=[5-8] -n 1 -host myMIC ... :
```



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Agenda

- Overview
- Programming Models
- Hybrid Computing
- Load Balancing



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Intel® Xeon Phi[™] Coprocessor Becomes a Network Node



Intel[®] Xeon Phi[™] Architecture + Linux enables IP addressability



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Load Balancing

- Situation
 - Host and coprocessor computation performance are different
 - Host and coprocessor internal communication speed is different
- MPI in symmetric mode is like running on a heterogenous cluster
- Load balanced codes (on homogeneous cluser) may get imbalanced!
- Solution? No general solution!
 - Approach 1: Adapt MPI mapping of (hybrid) code to performance characteristics: #m processes per host, #n process per coprocessor(s)
 - Approach 2: Change code internal mapping of workload to MPI processes
 - $\circ~$ Example: uneven split of calculation grid for MPI processes on host vs. coprocessor(s)
 - Approach 3: ...
- Analyze load balance of application with ITAC
 - Ideal Interconnect Simulator



Intel® Xeon Phi[™] Architecture

Improving Load Balance: Real World Case





Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Improving Load Balance: Real World Case





Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Improving Load Balance: Real World Case





Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Ideal Interconnect Simulator (IIS)

- What is the Ideal Interconnect Simulator (IIS)?
 - Using a ITAC trace of an MPI application, simulate it under ideal conditions
 - Zero network latency
 - Infinite network bandwidth
 - Zero MPI buffer copy time
 - Infinite MPI buffer size
 - Only limiting factors are concurrency rules, e.g.,
 - A message can not be received before it is sent
 - An All-to-All collective may end only when the last thread starts



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Ideal Interconnect Simulator (Idealizer)

🗱 Intel® Trace Analyzer - [7	7: Compare C:/Docume	nts and Settings/sar	mcmill/My Document	s/Software/Intel Trace	Collector & Analyzer/Trace Simulator/	bmd4_np8_dist.stf (A) wit	th C:/Documents and Settings	/samcmill/My Documen	ts/Software/Int 🔳 🗗 🚺	X
File Style Windows Help F1										×
View Charts Navigate Advanced	Layout Comparison									
A: C:/Documents and Settings/samor B: C:/Documents and Settings/samor	mill/My Documents/Software/ mill/My Documents/Software/	'Intel Trace Collector & An 'Intel Trace Collector & An	nalyzer/Trace Simulator/br nalyzer/Trace Simulator/br	1d4_np8_dist.stf 1d4_np8_dist.iis.single.stf						
0 s 40 s 20 s	80 s	120 s 100 s	160 s 140 s	200 s 180 s	240 s 280 220 s 260 s	s 320 s 300 s	360 s 340 s 380 s	400 s 420 s	440 s 480 460 s	
P0 Application	Appl	ication Applica	ation_//#/??/#///A	*/ MINN/IIIINMApplication	MENTERINATIONINATION	ыныныныныныныны	Application	//MPI_Alireduce MMAcM	PI_Alm/NAMATPUUTA /ArMPIMPI/	
P1 MPI_Recv	MPI	Recv	AV. S. I. S. S. MININ	•/•/MIIIIII/IIIII/MMPI_Allredu	uce MF111 TIMMINIMMINIM	N/IMPI_Barrier	ШЫШНААрНА(АНШш.Ш.а.	//MPI_Alireduce NINApM	PI_AlinMTAT/NT	
P2 MPI_Recv	MPI	Recv	NALL ILS. P. MILLION	1. (NIIIIII /IIIII /IFMPI_Allred	uce METHIN MINIMUM INT	NNININININININININI	ПААЛАА <i>ЛГАДЛИРА</i> ШАрА <mark>МРЕ</mark> ЛИИ	MPI_Alireduce NIMppN	PL_AIMTAL MT//////MPL_SWPL	
P3 MPI Recy	MPI	Recy	AND ALL MADE	4 MINUTATION MIMPL Allerda	Ince MELLER PROVIDENT PROVIDENT		CIII (III IIII III AII AA IIII AAD II A (ADD		ation MIAC II UUU ApplitUNPI	Actual trace
							φ			Actual trace
P4/ MPI_Recv	MPI	_Hecv	A 1 A A A 1 A 7	V. NITTE TITLE MIMPL_Alired		III INTIINTNITANTININININININININI	, JIIAIPAIPIPIPIJIJIAIJAJJAÇQIPIPQAÇ		RPI_AIMIN_IT_//// TApMPI/API	
P5 MPI_Recv	MPI	_Recv	N VN M VI 11	47 111111 (1111 MIMPI_Alired)	uce METTI / TINITTININITTINITTI		n <mark>IIIIIIAAIIAAANAINIIAIIIIIAAIIIIAAA</mark> NAA	WWW.Allreduce NMAppl	cMPI_M13/M1/////Naplication	
P61 MPI_Recv	MPI	_Recv	NAN MEMAL (19 <mark>00)00</mark> 1	VNNIIMI MINM MPI_Allred	uce METTE AND THE METTER AND THE MET	N/IMPI_Samer	<mark>HANHAJHAANNAHJIINAA</mark> MPLAA	reduce NINApp!	IPI_AIMTA ⁺ IIT###TM·MPI_Senci	
P7 <mark>II.MPI_Recv</mark>	MPI	_Recv	HANNIN AL	AAMININININI MININA	uce METLU VII. INNINIIINI II IIINIIIINI	IIIMIMIMIMIMIMIMIMIMIMIMIMIMIMIMIMIMIM	//////////////////////////////////////	M N Appl	MPL/MLAVTITIIII MAppl MPL	-
0 s	80 s		160 s		240 s	320 s				×
40 s	Applic	12Us ation Applicatio	on AlitAlitiAlititatiti	20Us	28U	s An MPLANA MUULAAn M	PIMP			
							ΛA			
PIMPI_Recv	MPI_F	(ecv		IPI_Alireduce	TETHERENINA (ACE ACE ACE ACE ACE ACE ACE ACE ACE ACE	e spMPLAInN/ //// App				
P2MPI_Recv	MPI_F	Recv	ANC 112 JUNDARD	IPI_Alireduce//	IN /A//AA ///A/////A///ArA/MPI_Alreduc	e <mark>Apr</mark> MPI_Alin <mark>ATUUAU</mark> (MPI	Pared			Idealized
P3MPI_Recv	MPI_F	Recv	VM V 4	IPI_Allreduce ///	HA HIAHHHAIAAA QIHIAAQIAA Aqot <mark>MPLA</mark> HAA	INITED optication Net all In April	MAPL.			Iueanzeu
P4MPI_Recv	MPI_F	lecv	VNICKI, MMINISCO	IPI_Allreduce	IF INTERPRETATION FOR THE PROPERTY AND A	II.III.I. IppIMPL_AINA IIIIII II AMI	PI_Allr			Trace
P5MPI Recv	MPL F	lecv	MAN & DOMINATOR	IPI Allreduce	a Jeleaalitaa jileleatitelelitelelitelelimet j	Allreduc Applic MPL NATHIN Appl	lication			
PAMPI Remi	MPL C	2001		IPI Allendung	TO ALL ON LAND DOWNER DOWN MIDE Street use					
FONTI_NCV		1000								
P7MPI_Recv	MPI_F	Recv	WANNAL (COMPANY)	IPI_Alreduce <mark>/</mark> I	NA VAIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	ApplicMPI_/NAU.////MAppl	MPI_			<u>v</u>
<										
Flat Profile Load Balance Ca	all Tree									
Children of Group All_Processes	~									-
B/A A	Self TSelf	TTotal	#Calls TSelf /	Call						S
MPI_probe MPI_send										
MPI_Bamer										
- MPI_Alreduce										
- Process 0	0.610	0.610	1.000	0.610						
Process 1	0.780	0.780	1.000	0.780						
Process 2	0.838	0.838	1.000	0.838						
- Process 4	0.758	0.758	1.000	0.758						
Process 5	0.659	0.659	1.000	0.659						
Process 6	0.831	0.831	1.000	0.831						
Process 7	0.805	0.805	1.000	0.805						
vvat									<u>N</u>	<u>2</u>

A: 216.389 s, Function Application



Intel® Xeon Phi[™] Architecture

Building Blocks: Elementary Messages





Building Blocks: Collective Operations





Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Application Imbalance Diagram: Total

Faster network	Intel® Trace Analyzer - [Application Imbalance Diagram (Total Mode)]		
Prist Prist </th <th>Total Choose the display mode Display the application time</th> <th>Colors</th> <th></th>	Total Choose the display mode Display the application time	Colors	
P003.37 7827.5 7221.43 6939.28 6232.21 Faster network 939.26 Change parallel 405.71 939.86 Change algorithm 192.87 293.77 293.77 293.78 293.78 193.86 Change algorithm 194.29 196.01	8419.64		
7875 721.13 655.36 689.37 622.21 Faster network 980.07 9125 Change parallel 492.86 492.87 980.71 324.64 328.64	8053.57		
72143 695.38 623.21 Faster network 591.07 591.07 591.07 591.07 591.07 591.07 591.07 591.07 592.26 Change parallel decomposition 300.71 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.64 328.67	7687.5		
6953.30 Faster network "interconnect" 587.14 "load imbalance" "Ioad imbalance" 5125 Change parallel decomposition "Ioad imbalance" Image: State of the s	7321.43		
Geometry Faster network "interconnect" \$357.14 State "load imbalance" \$499.86 "load imbalance" \$499.86 "load imbalance" \$499.86 "load imbalance" \$400.71 State \$283.71 State \$283.72 State \$262.52 Change algorithm \$199.26 "calculation"	6955.36		
Sector Interconnect Sector Change parallel decomposition Sector "load imbalance"			٦
Signal Change parallel decomposition "load imbalance" 4758.93 "load imbalance" 4922.65 Second 560.71 Second 3294.64 Second 228.57 Second 256.43 Change algorithm 193.26 "calculation" 193.36 Second 164.29 Second 193.21 Second 193.22 Second 193.21 Second 193.21 Second 193.22 Second 193.23 Second 193.24 Second 193.25 Second </td <td>S857.14</td> <td>Interconnect</td> <td></td>	S857.14	Interconnect	
5125 Change parallel decomposition "load imbalance" Image: State of the	5491.07		
Change parallel decomposition 4392.86 "load imbalance" 4056.71 3660.71 3294.64 3660.71 2156.43 Change algorithm 164.29 164.29 199.21 164.29 199.21 164.29 199.21 164.29 199.21 164.29 199.21 164.29 199.21 164.29			⊢⊢MPI
decomposition 4026.78 3660.71 3294.64 2928.57 2196.43 Change algorithm 1803.66 1464.29 1998.21 1998.2	4758.93		
4026.78	4392.86- decomposition —		
3660.71 3294.64 2928.57 2562.5 2196.43 Is30.36 1464.29 1098.21 732.143 366.071	4026.78		J
3294.64 2928.57 2562.5 2196.43 1830.36 "calculation" 1464.29	3660.71		
2928.57 2562.5 2196.43 1830.36 1464.29 1098.21 732.143 366.071	3294.64		
2562.5 Change algorithm "calculation" 1830.36	2928.57		
Change algorithm "calculation" 1830.36 1464.29 198.21	2562.5		
1830.36 1464.29 1098.21 732.143 366.071	^{2196,43} Change algorithm	Calculation	
1464.29	1830.36		
1098.21 732.143 366.071	1464.29		
732.143	1098.21		
366.071	732.143		
	366.071		



Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division

Application Imbalance Diagram: Breakdown





Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division



Software

Intel® Xeon Phi™ Architecture

Software

Software & Services Group, Developer Relations Division

Questions?



(intel) Software Intel® Xeon Phi[™] Architecture

Software & Services Group, Developer Relations Division