

# ATPESC

(Argonne Training Program on Extreme-Scale Computing)

## Computer Architecture Essentials

James Reinders

August 1, 2016, Pheasant Run, St Charles, IL

09:30-10:15



© 2016, James Reinders. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.  
\*Other names and brands may be claimed as the property of others.

9:30 am - 10:15 am

## Presentation: Computer Architecture Essentials

Lecturer Room



James Reinders, Recently Semi-retired, Former Intel Director

10:45 am - 12:00 pm

## Presentation: Structured Parallel Programming

Lecturer Room



James Reinders, Recently Semi-retired, Former Intel Director

1:00 pm - 1:45 pm

## Presentation: Performance: SIMD, Vectorization and Performance Tuning

Lecturer Room



James Reinders, Recently Semi-retired, Former Intel Director

Knights Landing Clustering and Memory Modes, use and implications on the future of architecture and memory configurations.

Vectorization, current state of the art thinking, use and implications on the future of data parallelism through threading + SIMD instructions.





I have been fortunate, and I like to share. 😊

The image shows the cover of the book 'Intel Xeon Phi Processor High Performance Programming Knights Landing Edition'. The cover features a dark background with a glowing, abstract pattern of light blue and green, resembling a network or data flow. The title is written in white, bold, sans-serif font. Below the title, the authors' names are listed in a smaller white font. The Intel logo is visible in the bottom left corner of the cover.

**INTEL® XEON PHI™  
PROCESSOR  
HIGH PERFORMANCE  
PROGRAMMING  
KNIGHTS LANDING EDITION**

Jim Jeffers | James Reinders | Avinash Sodani

**MK**

**2016  
KNL die.  
ENZO cosmology simulation of the  
intergalactic “cosmic web” of  
gravitational filaments that link  
galaxies, define the structure of the  
universe, and indicate dark matter.**

© 2016, James Reinders. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.  
\*Other names and brands may be claimed as the property of others.



The book cover features a blue-tinted wafer shot of an Intel Xeon Phi coprocessor die. A white corner is peeling away from the top right, revealing a colorful, abstract pattern. The title is in a bold, black sans-serif font. Below the title, the authors' names are listed in a smaller font. The MK logo is positioned in the lower-left corner of the cover area. The year 2013 and a note about the cover art are at the bottom of the cover area.

# Intel® Xeon Phi™ Coprocessor High Performance Programming

Jim Jeffers, James Reinders

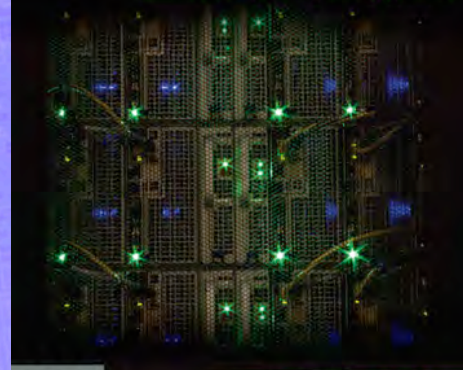
MK  
Morgan Kaufmann

2013  
KNC dies (close in wafer shot).  
Artist added the cosmic scene.



## High Performance Parallelism Pearls

*Multicore and Many-core Programming Approaches*



MK  
MORGAN KAUFMANN

James Reinders, Jim Jeffers

**2014**  
**First KNC machine (TACC).**

## Intel® Xeon Phi™ Coprocessor High Performance Programming

Jim Jeffers, James Reinders



MK  
MORGAN KAUFMANN

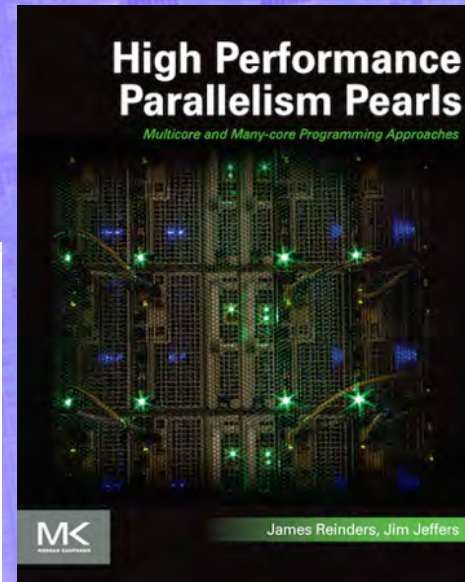
**2013**  
**KNC dies (close in wafer shot).**  
**Artist added the cosmic scene.**



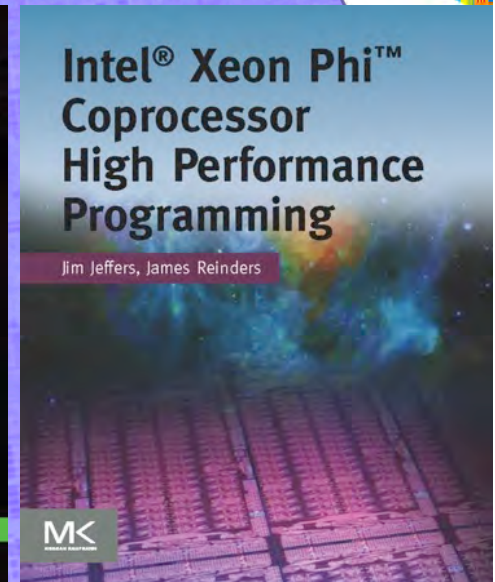
# 'Titanic' night sky adjusted after Neil deGrasse Tyson criticized James Cameron



© Getty Images © Wikimedia Commons  
The sky's the limit: Astrophysicist Neil deGrasse Tyson, right, convinced Titanic director James Cameron, left, to make the night sky historically accurate in the film's 3D release

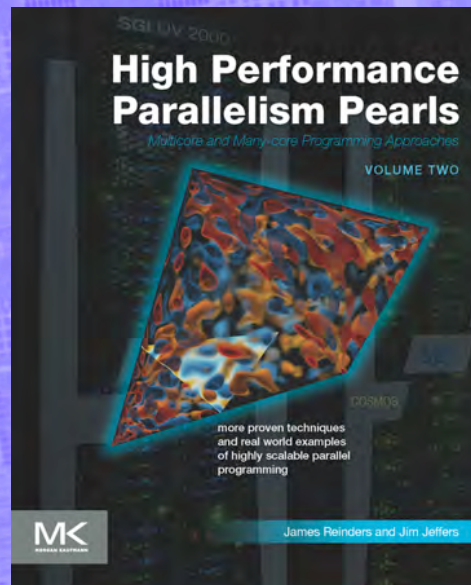


2014  
First KNC machine (TACC).

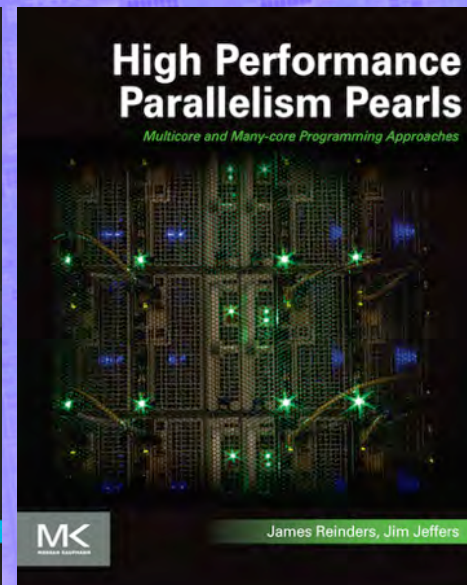


2013  
KNC dies (close in wafer shot).  
Artist added the cosmic scene.

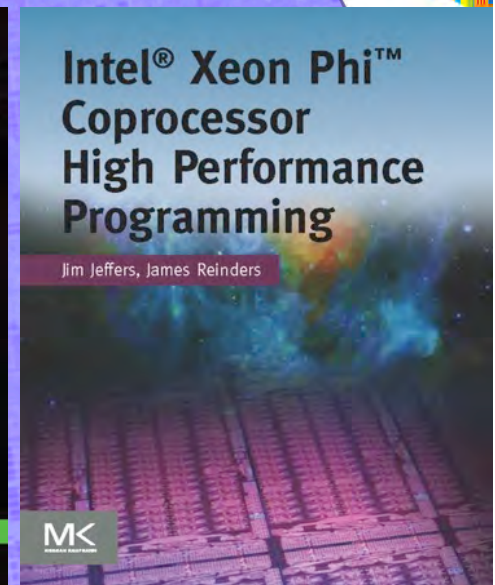




**2015**  
**SGI machine (Cambridge).**  
**3D visualization of statistical fluctuations in the Cosmic Microwave Background, the remnant of the first visible light after the Big Bang. CMB data is from the Planck satellite and is the topic of Chapter 10 providing insights into the new physics and how the universe evolved.**



**2014**  
**First KNC machine (TACC).**

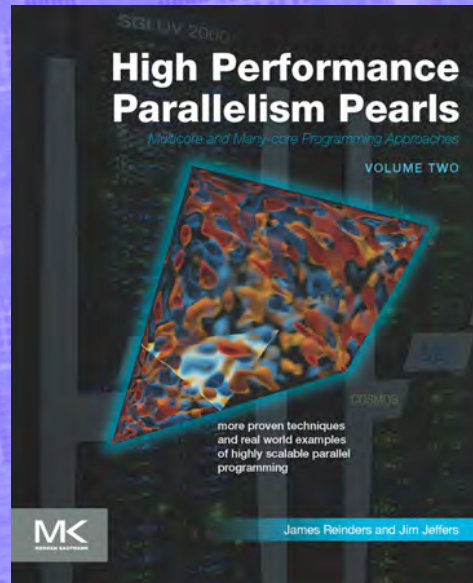


**2013**  
**KNC dies (close in wafer shot). Artist added the cosmic scene.**

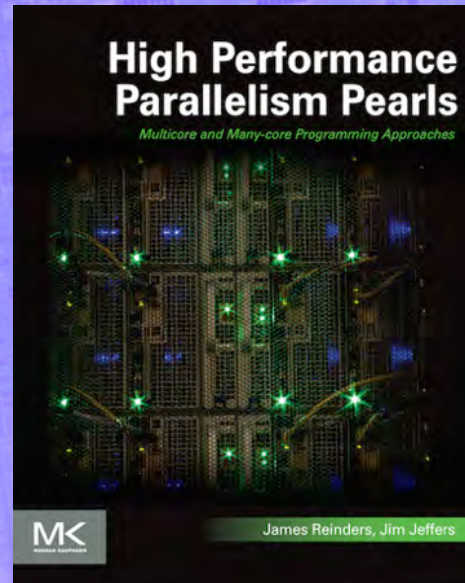




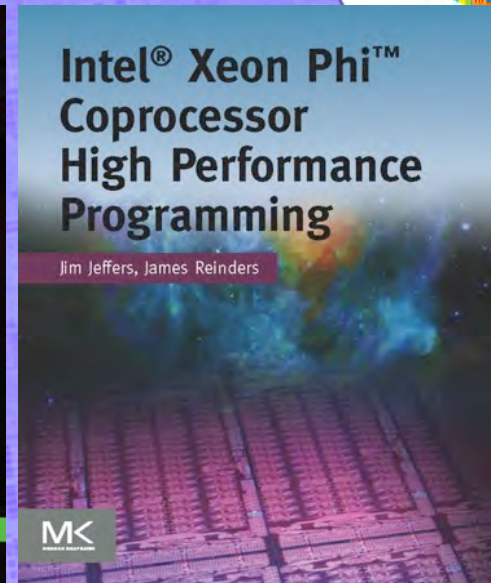
**2016**  
**KNL die.**  
**ENZO cosmology simulation of the intergalactic “cosmic web” of gravitational filaments that link galaxies, define the structure of the universe, and indicate dark matter.**



**2015**  
**SGI machine (Cambridge).**  
**3D visualization of statistical fluctuations in the Cosmic Microwave Background, the remnant of the first visible light after the Big Bang. CMB data is from the Planck satellite and is the topic of Chapter 10 providing insights into the new physics and how the universe evolved.**



**2014**  
**First KNC machine (TACC).**



**2013**  
**KNC dies (close in wafer shot).**  
**Artist added the cosmic scene.**



I have found...  
example after example  
getting performance and  
performance portability with  
“*just* parallel programming.”





# MULTITHREADING FOR VISUAL EFFECTS

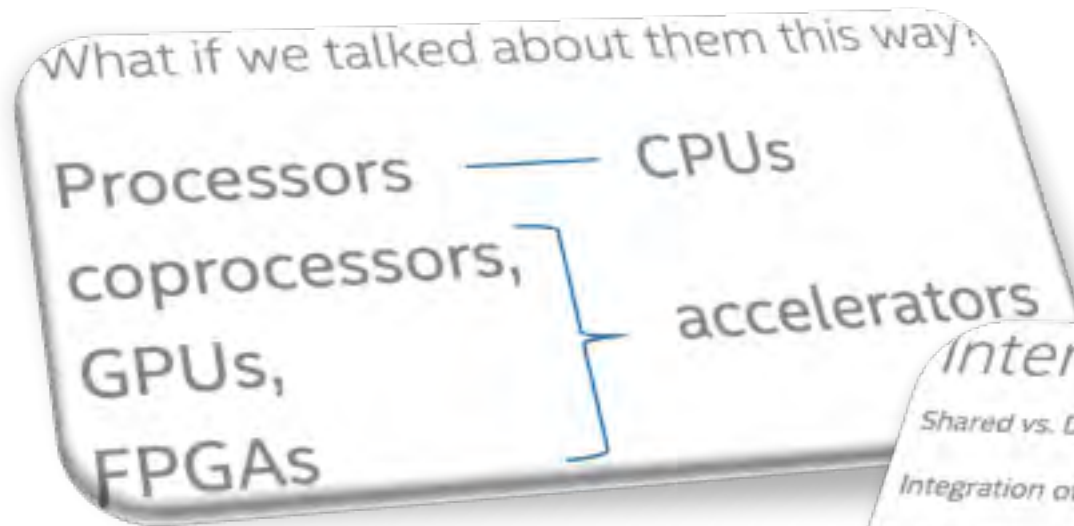
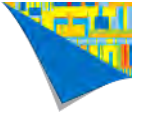
Martin Watt • Erwin Coumans • George Elkoura • Ronald Henderson  
Manuel Kraemer • Jeff Lait • James Reinders

 CRC Press  
Taylor & Francis Group



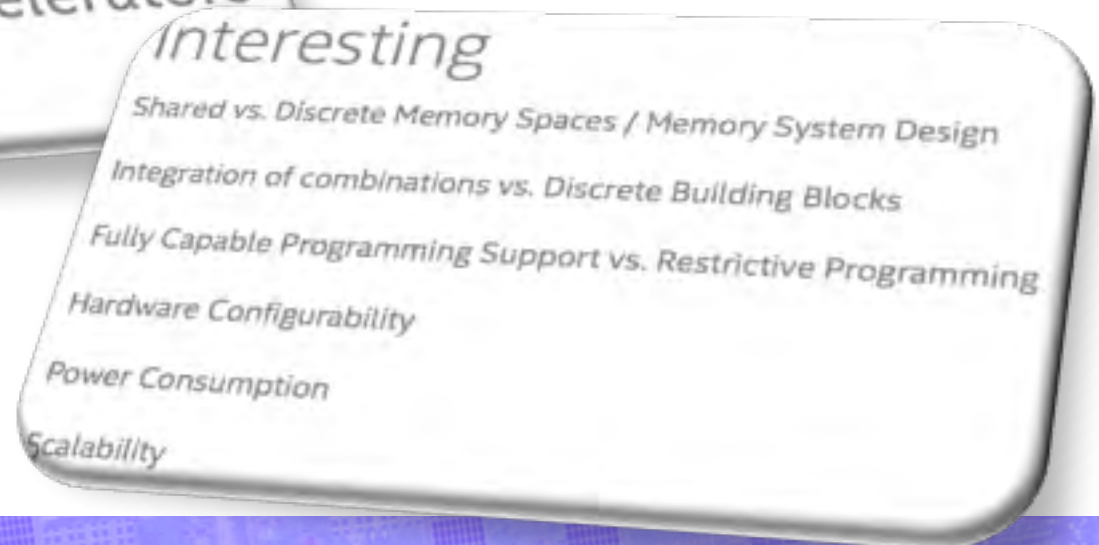
< My  
daughter's  
favorite  
movie.

# Computer Architecture is FUN AGAIN

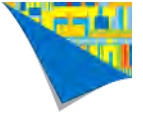


we need to make sure  
software is not  
collateral damage.

Performance and  
Performance Portability  
should be a requirement.







# Interesting

Shared vs. Discrete Memory Spaces / Memory System Design

Integration of combinations vs. Discrete Building Blocks

Fully Capable Programming Support vs. Restrictive Programming

Hardware Configurability

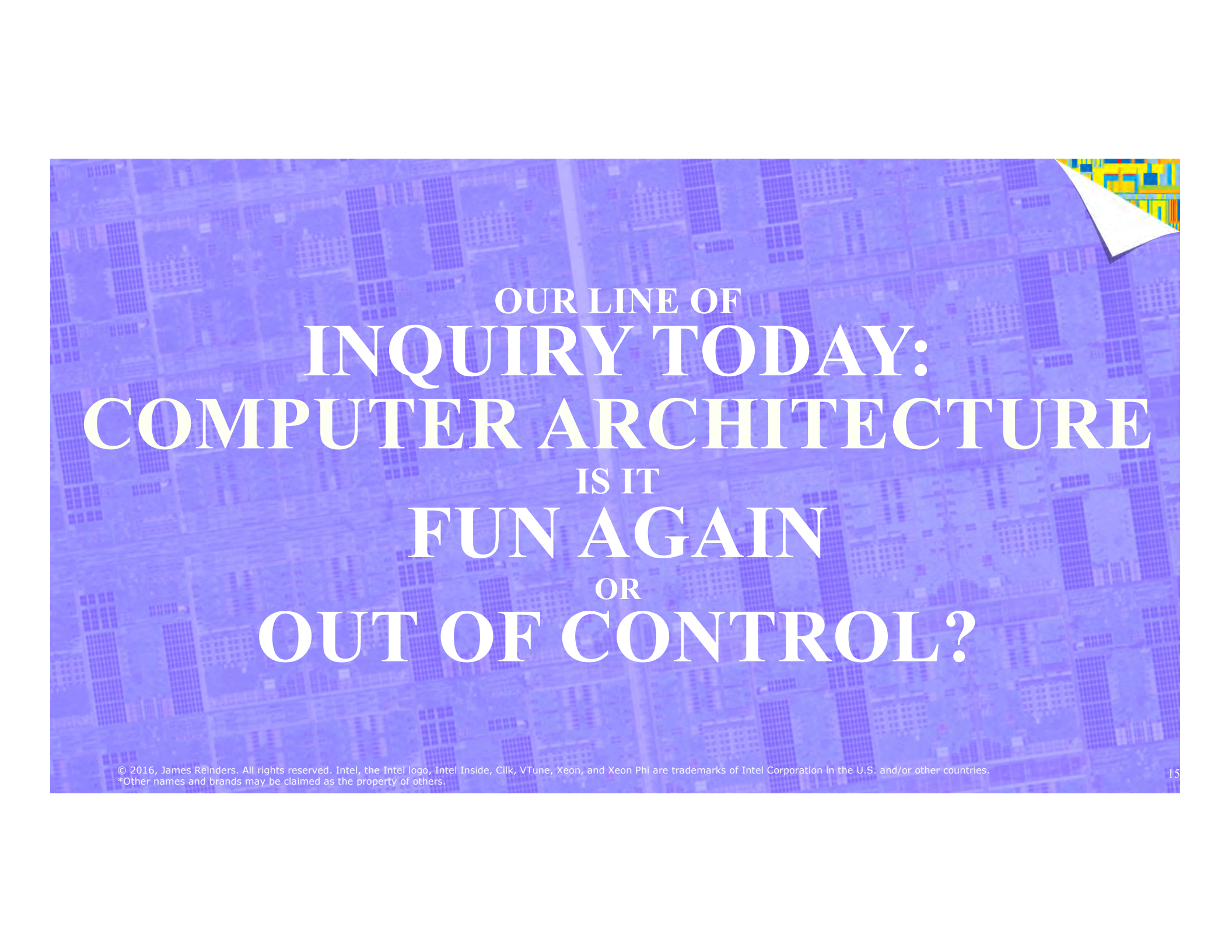
Power Consumption

Scalability

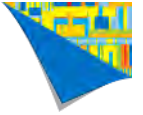


# KEEP CALM AND BE INQUISITIVE ENGINEERS



The background of the slide is a blue-tinted image of a microchip die, showing a grid of circuitry. In the top right corner, there is a white, curled-up corner effect, suggesting the slide is a page from a book or document.

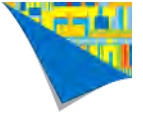
OUR LINE OF  
INQUIRY TODAY:  
COMPUTER ARCHITECTURE  
IS IT  
FUN AGAIN  
OR  
OUT OF CONTROL?



# Three debates changing our lives

Scale:                   “many but lower performance each” vs.  
                              “higher performance each but fewer”

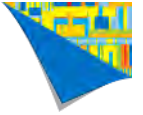




# Three debates changing our lives

Scale: “many but lower performance each” vs.  
“higher performance each but fewer”

Capabilities: “high performance but more niche” vs.  
“more general but less performance”



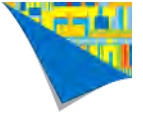
# Three debates changing our lives

Scale: “many but lower performance each” vs.  
“higher performance each but fewer”

Capabilities: “high performance but more niche” vs.  
“more general but less performance”

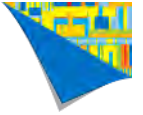
Besides the compute (memory / communication / data movement):  
Oh so many things to consider.  
We will brush the surface.





# Three needs affecting the debate

**Power:** 1MW/year electricity costs ~\$1M/year

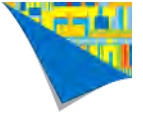


# Three needs affecting the debate

**Power:** 1MW/year electricity costs ~\$1M/year

**Portability:** Coding is expensive, doing it more is moreso.





# Three needs affecting the debate

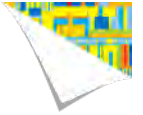
**Power:** 1MW/year electricity costs ~\$1M/year

**Portability:** Coding is expensive, doing it more is moreso.

## **Performance portability:**

You can start a bar fights with just trying to define this. \*

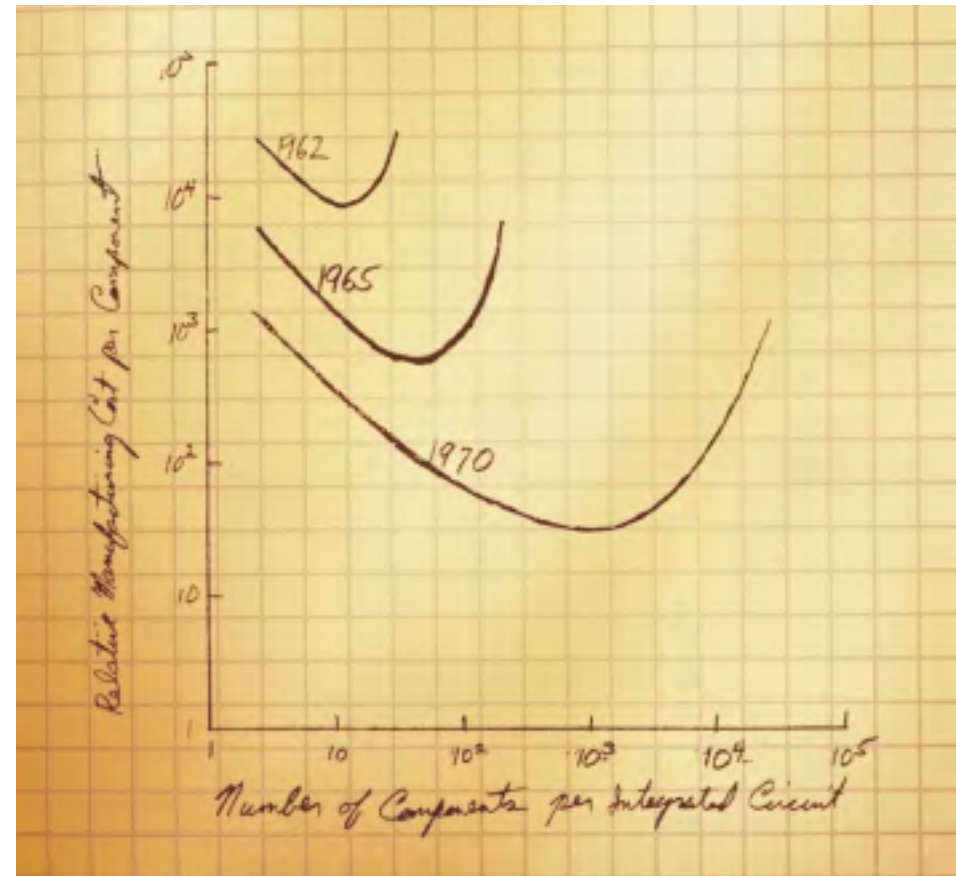
*\* if the bar patrons are HPC programmers*

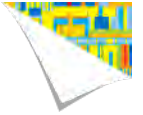


## Why Multicore?

The “Free Lunch” is over, really.

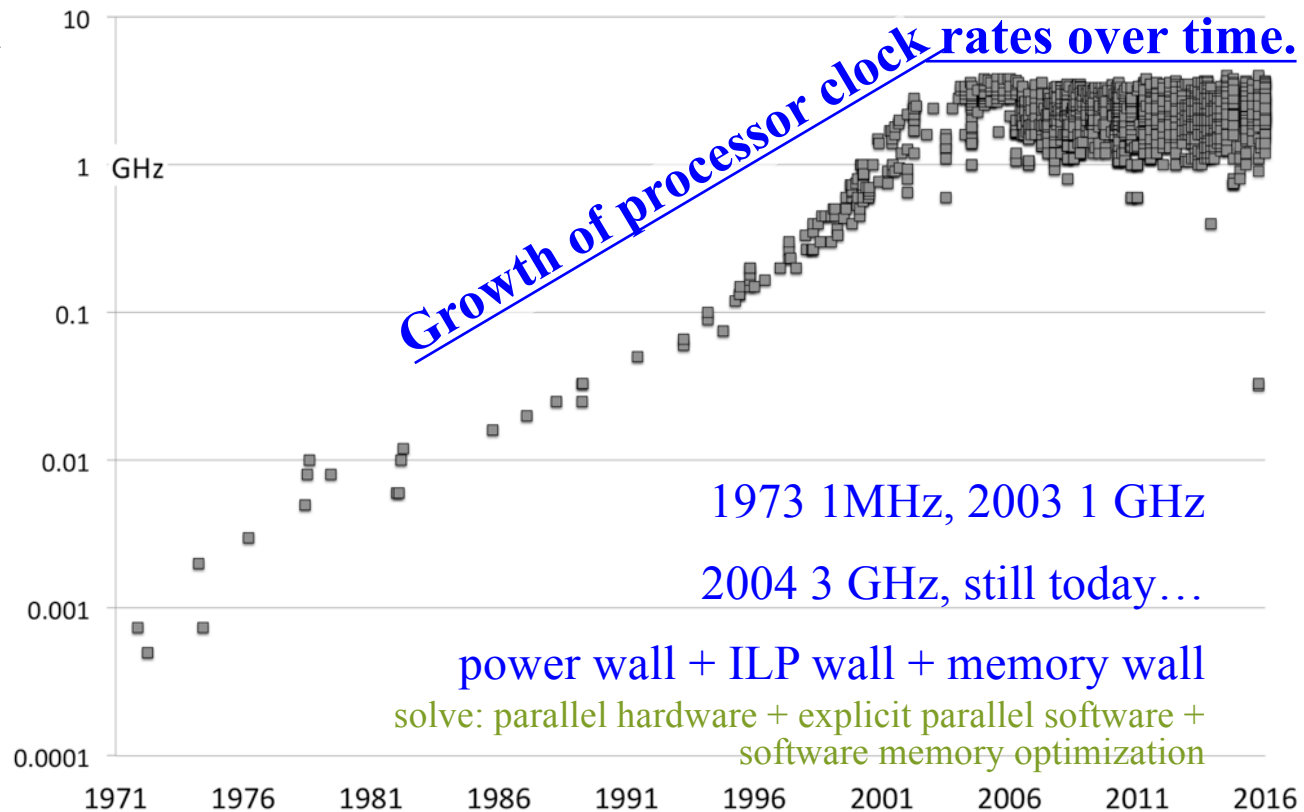
But Moore’s Law continues!





# Processor Clock Rate over Time

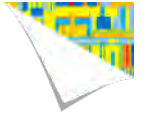
Growth halted  
around 2005



**Figure 1.1**  
**(KNL book)**

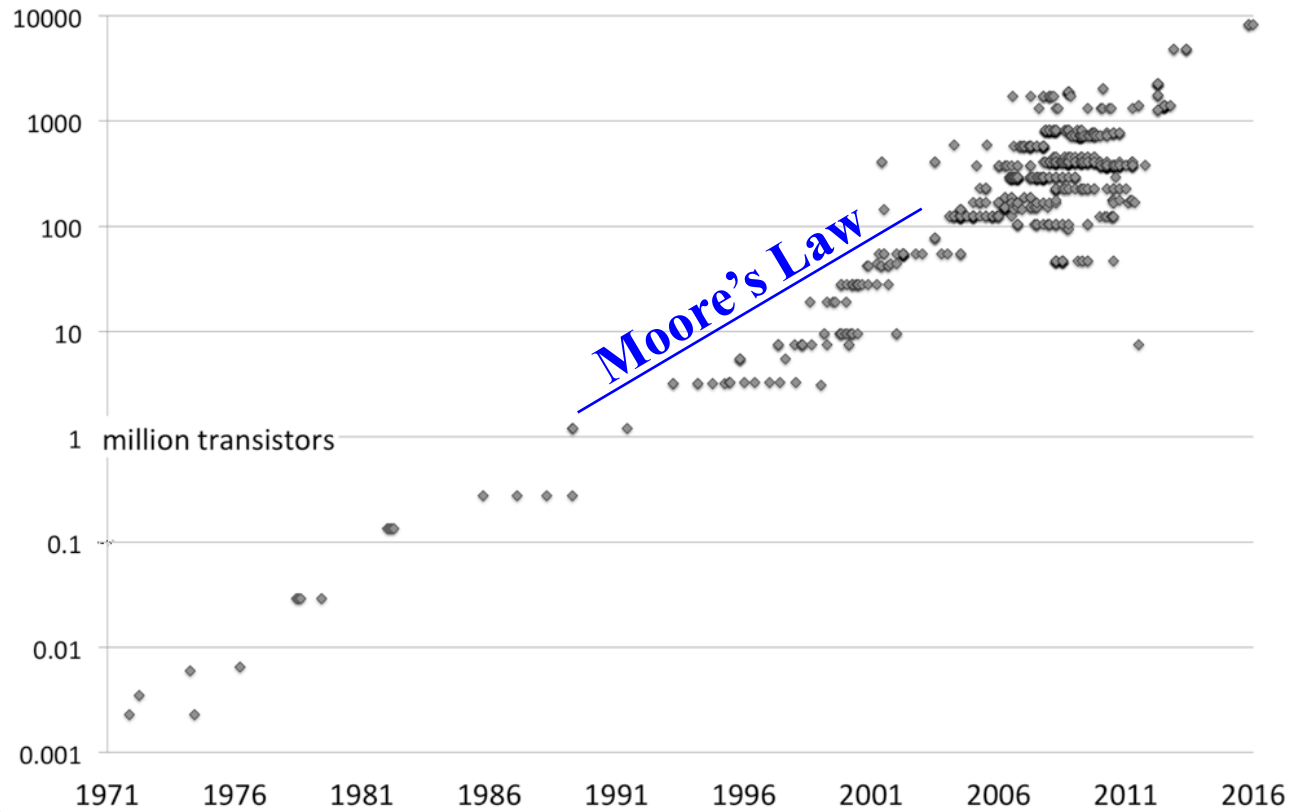
© 2016, James Reinders, used with permission. <http://lotsofcores.com>





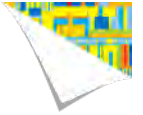
# Transistors per Processor over Time

Continues to grow exponentially (Moore's Law)

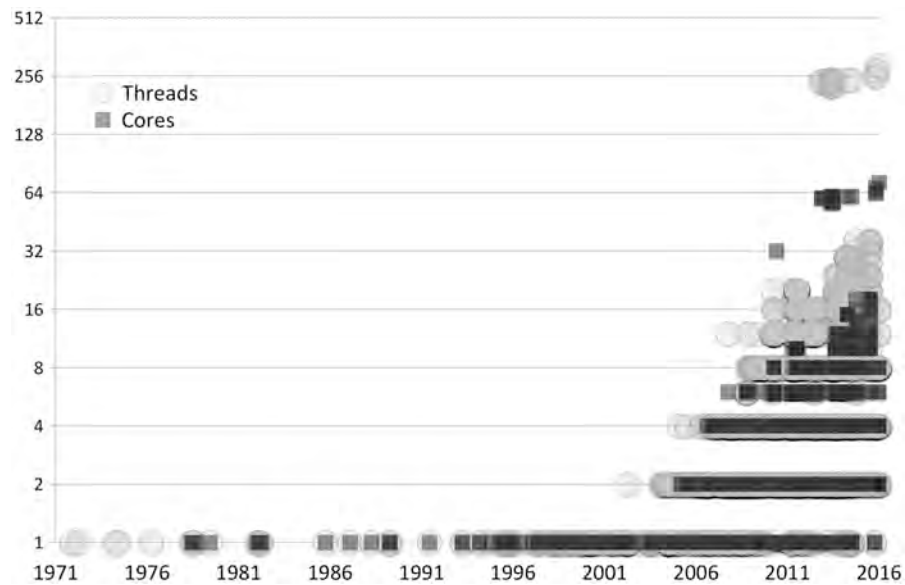


**Figure 1.4**  
**(KNL book)**

© 2016, James Reinders, used with permission. <http://lotsofcores.com>



## Core and Thread Counts



Single core, single thread, ruled for decades.

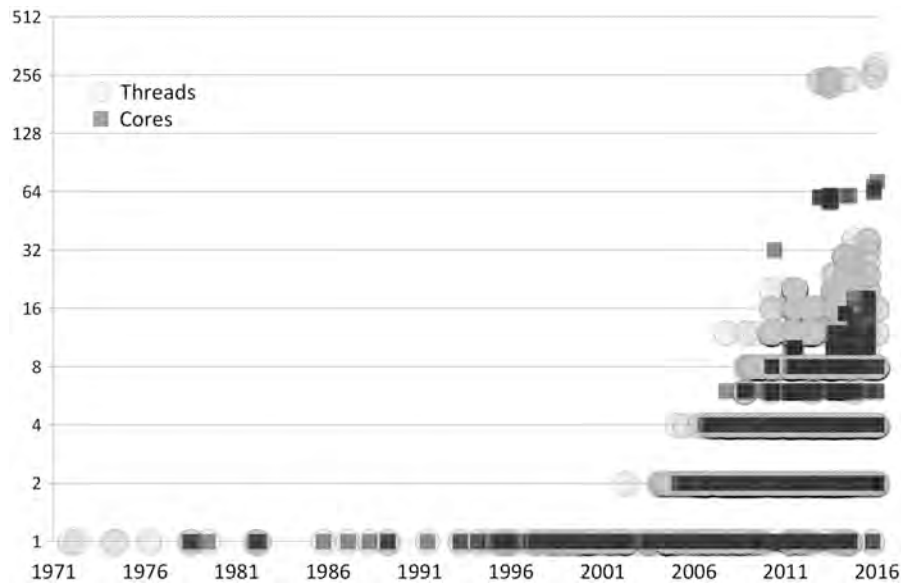
**Multithread:** grow die area small % for addition hardware thread(s) sharing resources.

**Multicore/Many Core:** 100% die area for additional hardware thread without sharing,

**Figure 1.2**  
**(KNL book)**

© 2016, James Reinders, used with permission. <http://lotsofcores.com>

## Core and Thread Counts

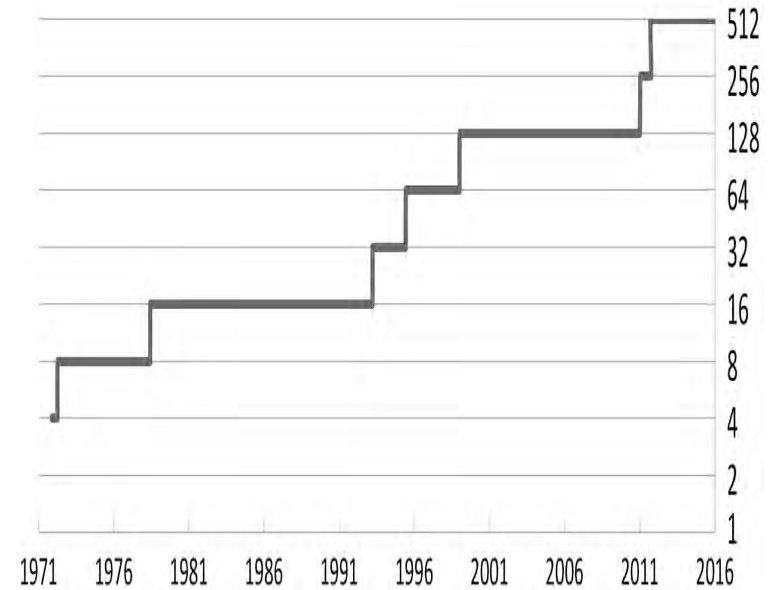


Single core, single thread, ruled for decades.

**Multithread:** grow die area small % for addition hardware thread(s) sharing resources.

**Multicore/Many Core:** 100% die area for additional hardware thread without sharing,

## Width

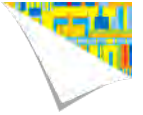


**Data parallelism:** handling more data at once, multibyte, multiword, many words.

**Figures 1.2 and 1.3**  
**(KNL book)**

© 2016, James Reinders, used with permission. <http://lotsofcores.com>

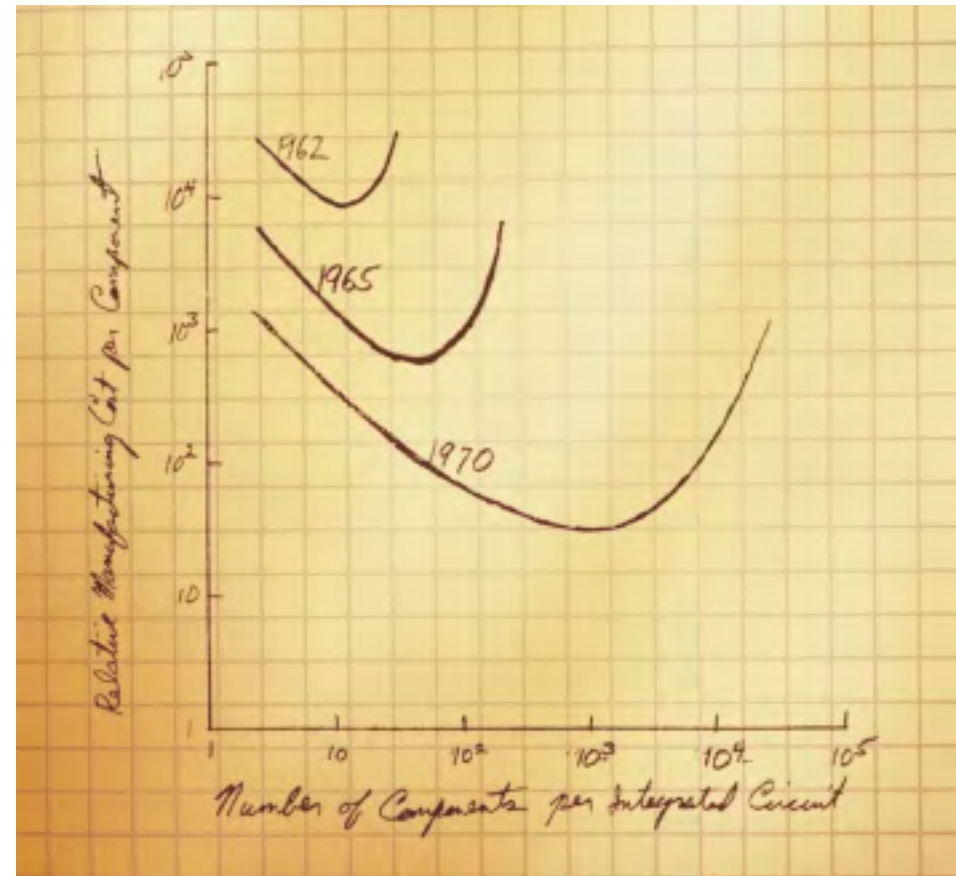


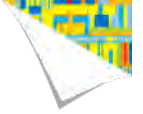


## Why Multicore?

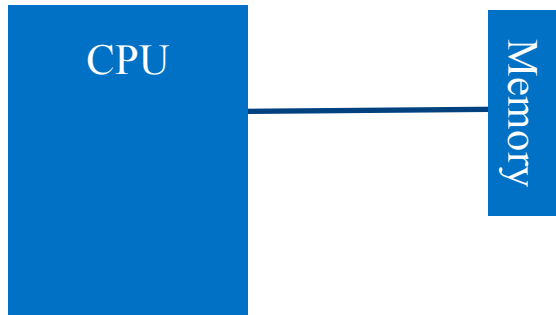
The “Free Lunch” is over, really.

But Moore’s Law continues!

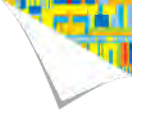




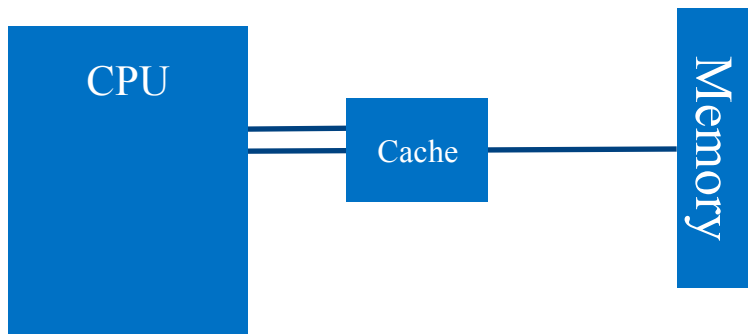
# CPU



These were simpler times.



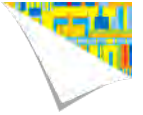
# CPU + cache



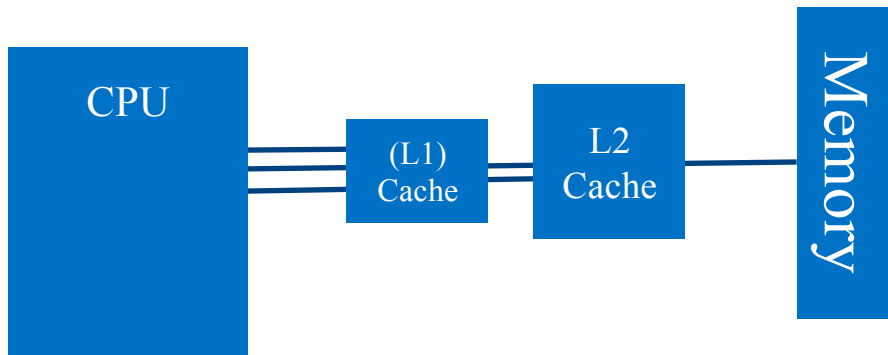
Memories got “further away”  
(meaning: CPU speed increased  
faster than memory speeds)

A closer “cache” for frequently used  
data helps performance when memory  
is no longer a single clock cycle away.



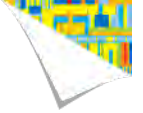


# CPU + caches

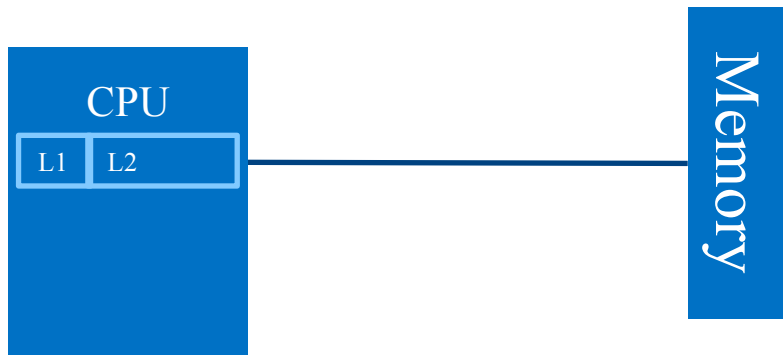


Memories keep getting “further away”  
(this trend continues today).

More “caches” help even more  
(with temporal reuse of data).

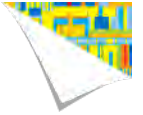


# CPU with caches

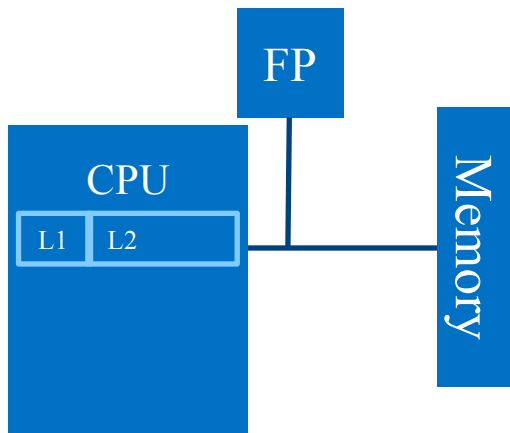


As transistor density increased (Moore's Law), cache capabilities were integrated onto CPUs.

Higher performance external (discrete) caches persisted for some time while integrated cache capabilities increase.

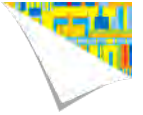


# CPU / Coprocessors

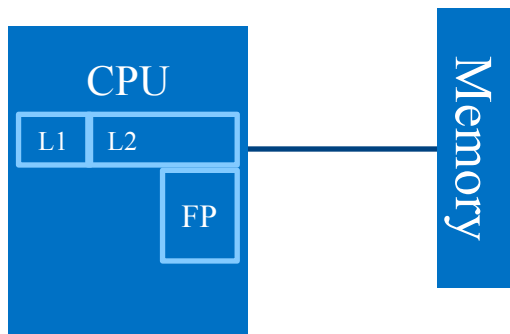


Coprocessors appearing first in 1970s were FP accelerators for CPUs without FP capabilities.



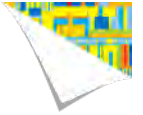


# CPU / Coprocessors



As transistor density increased (Moore's Law), FP capabilities were integrated onto CPUs.

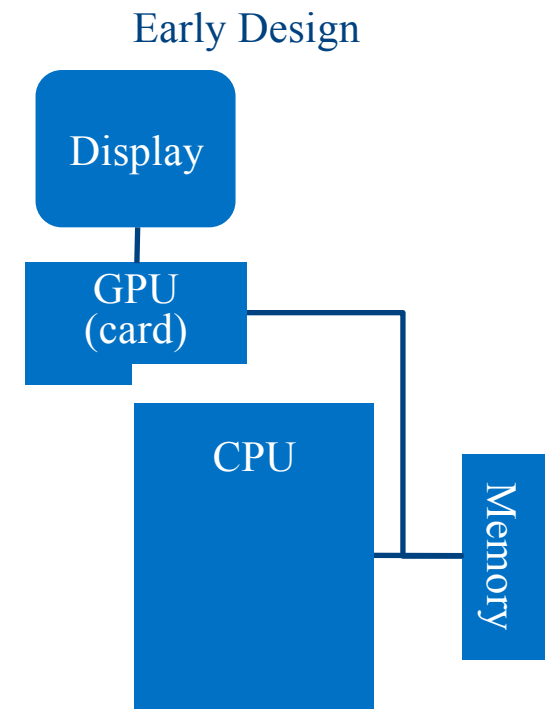
Higher performance discrete FP “accelerators” persisted a little bit while integrated FP capabilities increase.

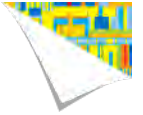


# CPU / Coprocessors

Interest to provide hardware support for displays increased as use of graphics grew (games being a key driver).

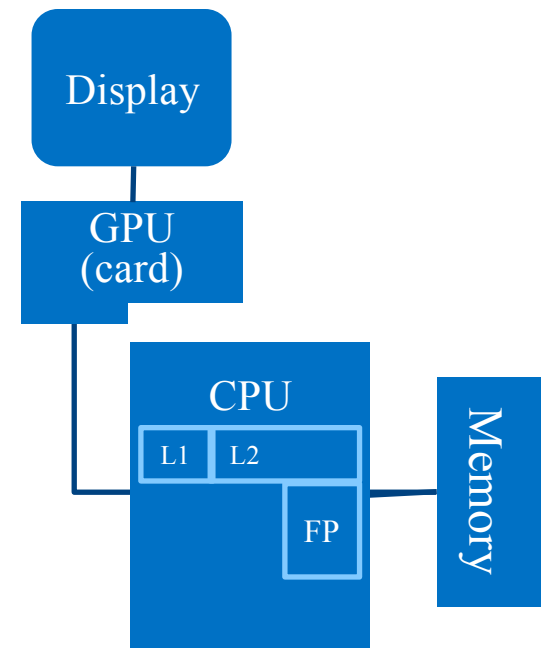
This led to graphics processing units (GPUs) attached to CPUs to create video displays.

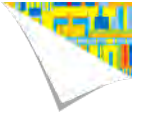




# CPU / Coprocessors

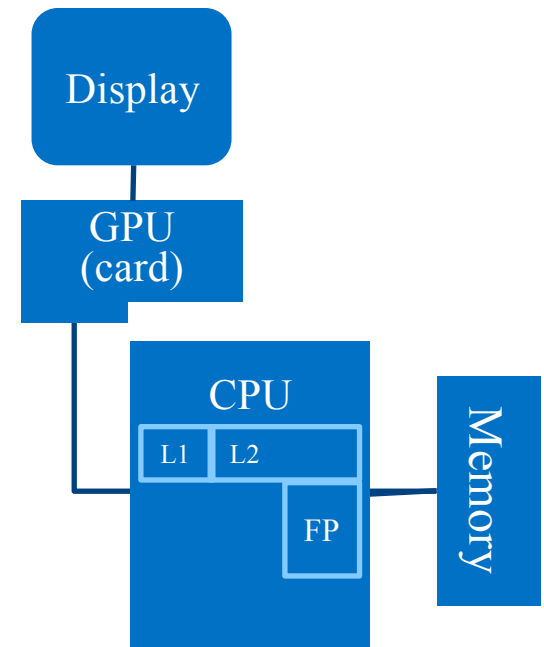
GPU speeds and CPU speeds increase faster than memory speeds. Direct connection to memory best done via caches (on the CPU).



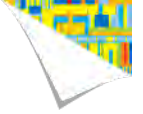


# CPU / Coprocessors

GPU speeds and CPU speeds increase faster than memory speeds. Direct connection to memory best done via caches (on the CPU).



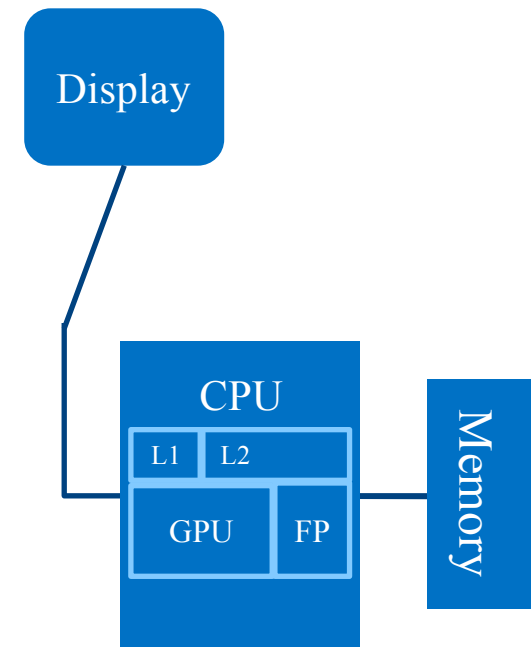


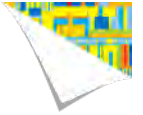


# CPU / Coprocessors

As transistor density increased (Moore's Law), GPU capabilities were integrated onto CPUs.

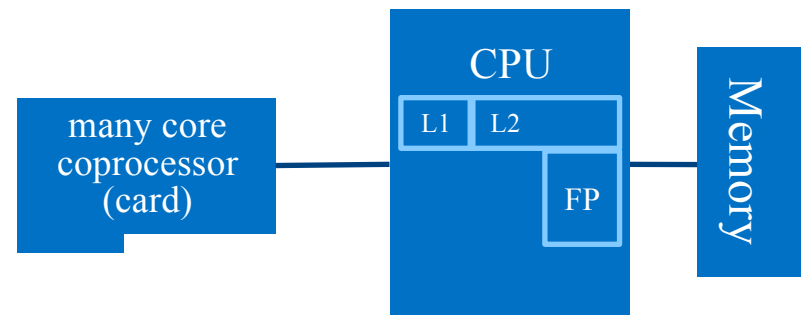
Higher performance external (discrete) GPUs persist while integrated GPU capabilities increase.

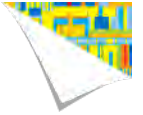




# CPU / Coprocessors

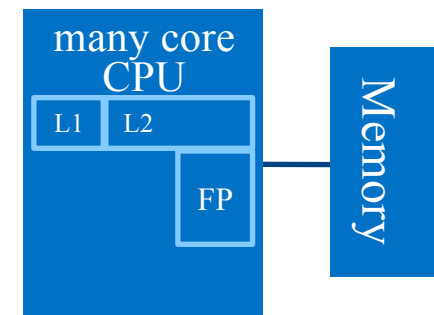
A *many core* coprocessor (Intel® Xeon Phi™) appears, purpose built for accelerating technical computing.

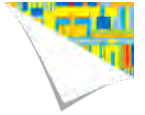




# CPU / Coprocessors

As transistor density increased (Moore's Law), many core capabilities will be integrated to create a many core CPU.  
("Knights Landing")



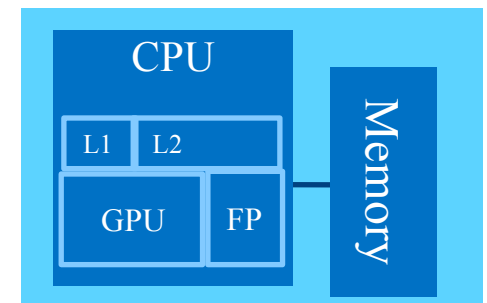
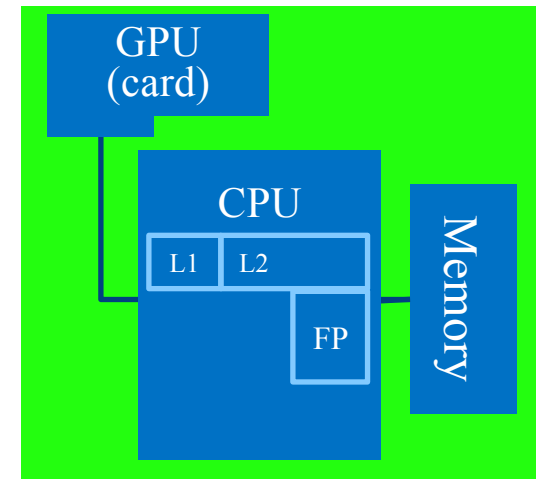
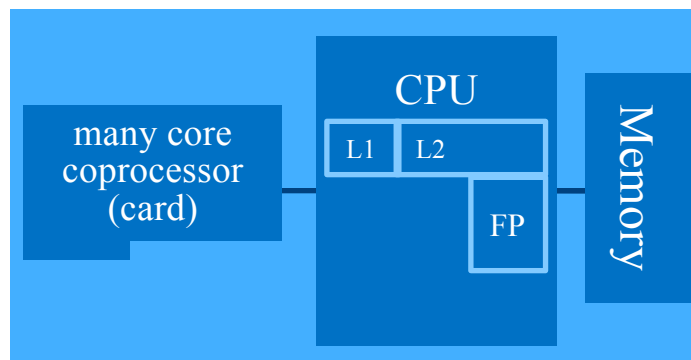
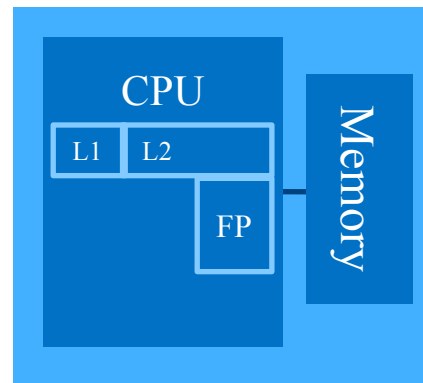


# Nodes

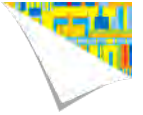
“Nodes” are building blocks for clusters.

With or without GPUs.

Displays not needed.

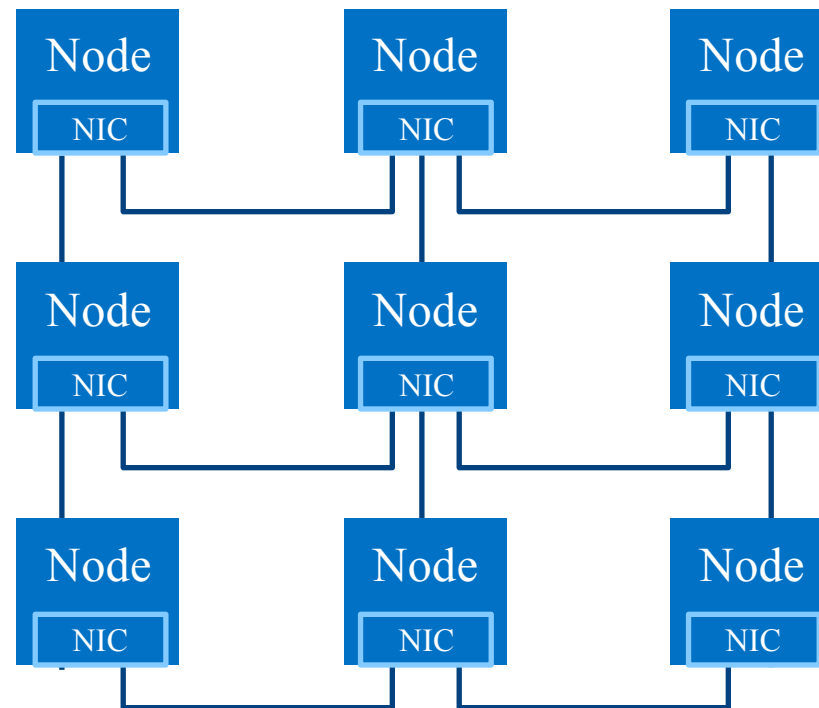


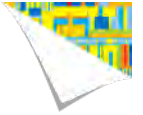




# Clusters

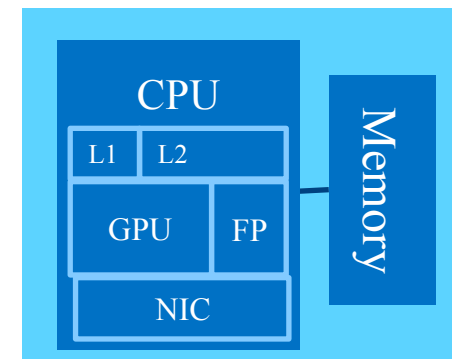
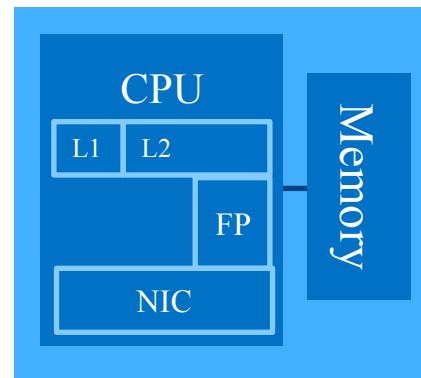
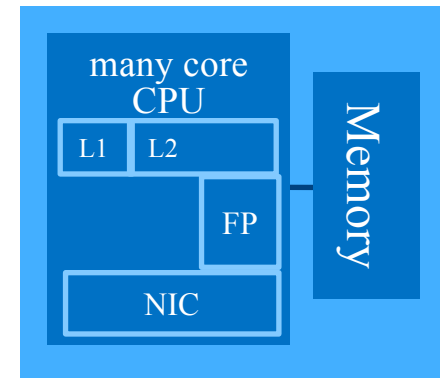
Clusters are made by connecting nodes - regardless of “Nodes” type.





# NIC (Network Interface Controller) integration

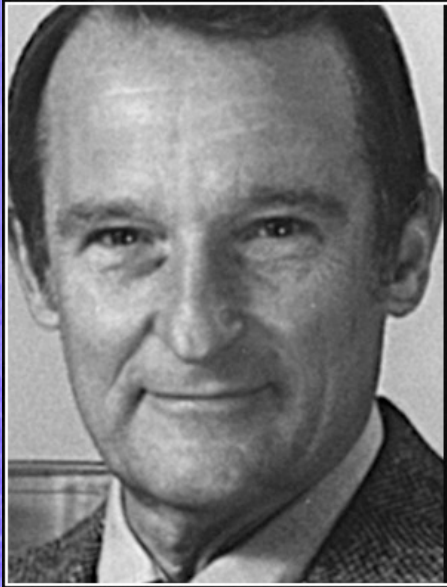
As transistor density increased (Moore's Law), NIC capabilities will be integrated onto CPUs.



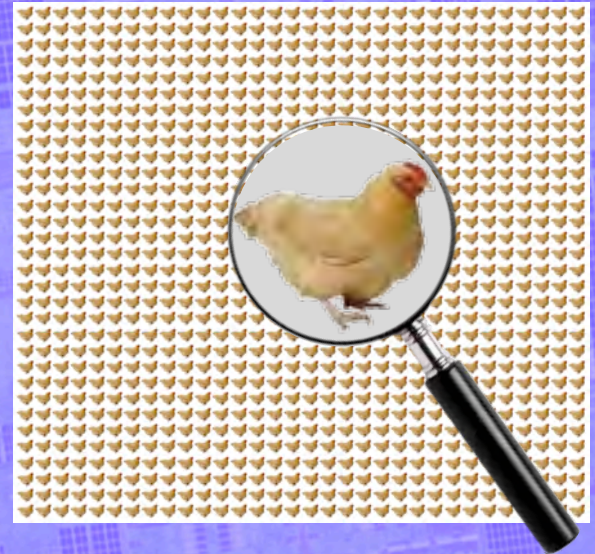
# Why Intel® Xeon Phi™ Processors?





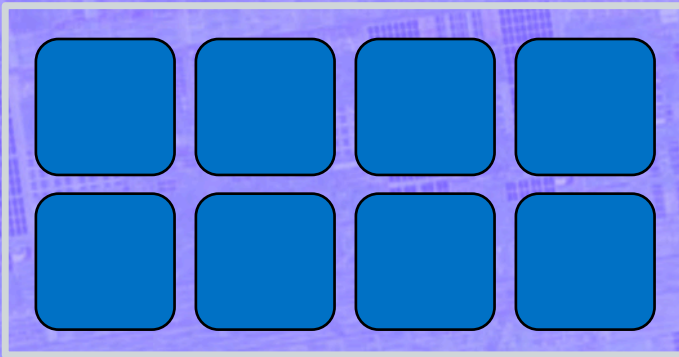


If you were plowing a field,  
which would you rather use...  
two strong oxen, or  
1024 chickens?



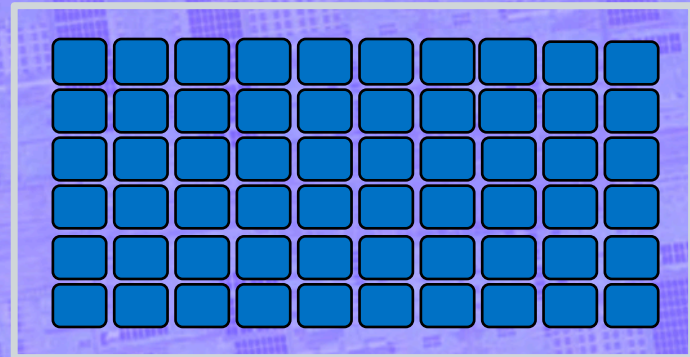


# Design Question - Best for Computing?



A few powerful

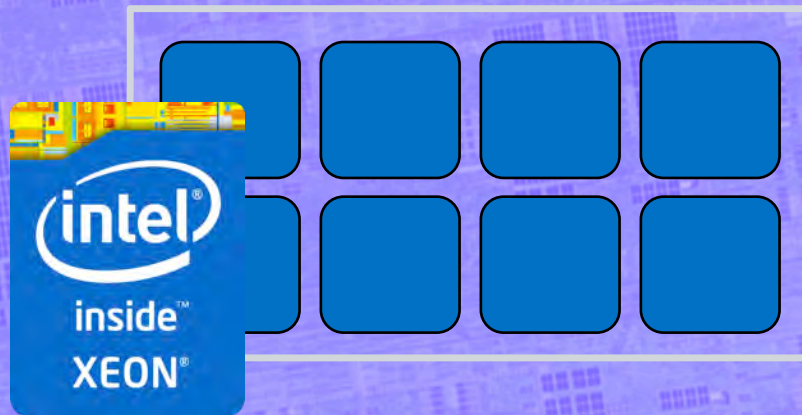
vs.



Many less powerful.

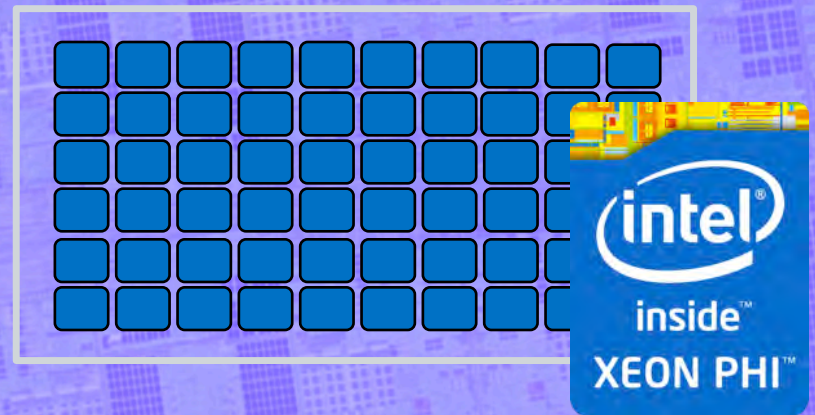
Diagrams for discussion purposes only, not a precise representation of any product of any company.

© 2016, James Reinders. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.  
\*Other names and brands may be claimed as the property of others.



A few powerful

vs.



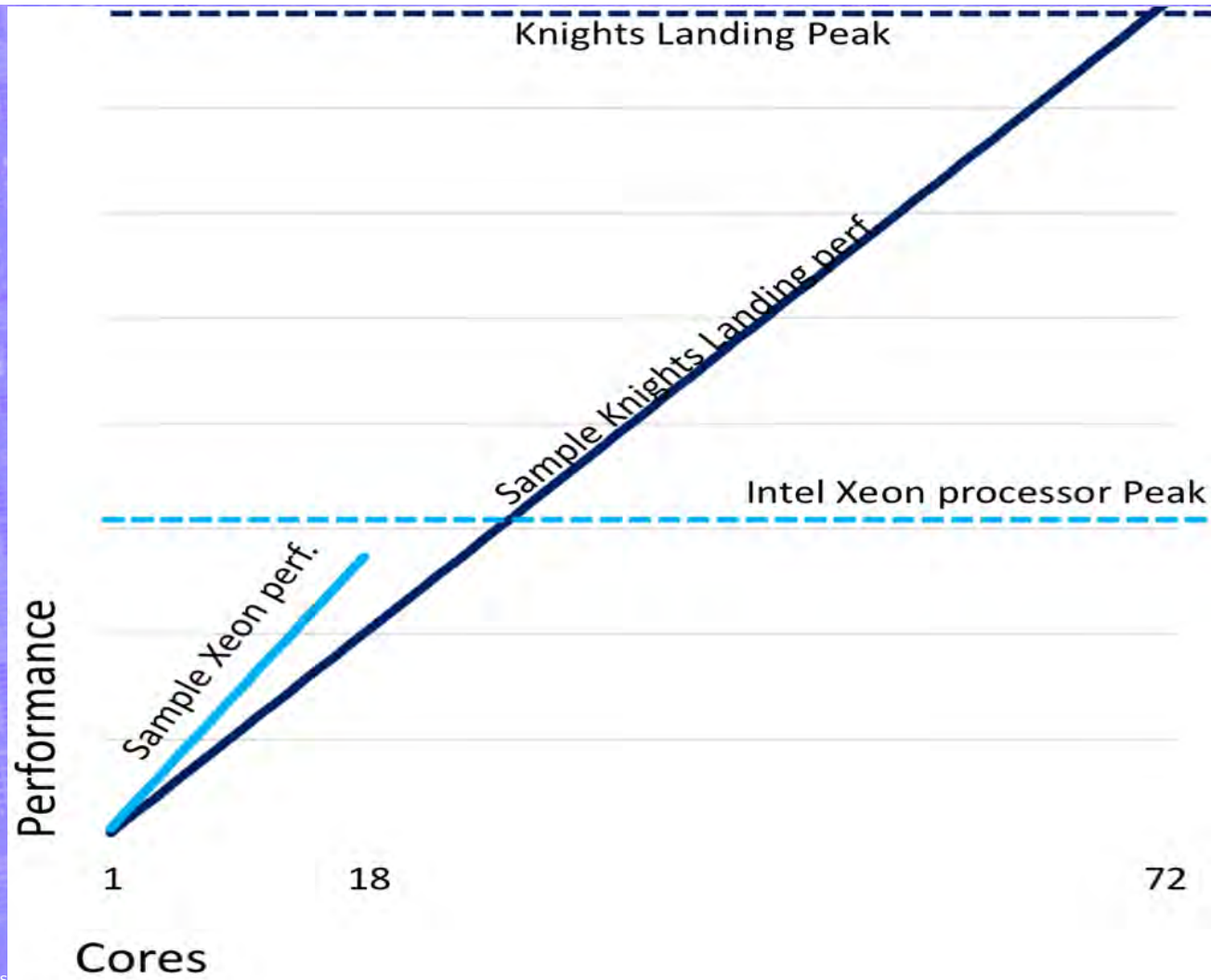
Many less powerful.

Same programming models, languages, optimizations and tools.

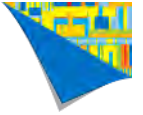
Diagrams for discussion purposes only, not a precise representation of any product of any company.

© 2016, James Reinders. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.  
\*Other names and brands may be claimed as the property of others.





**Figure 1.5**  
**(KNL book)**



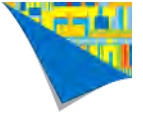
# Three debates changing our lives

**Scale:** “many but lower performance each” vs.  
“higher performance each but fewer”

**Capabilities:** “high performance but more niche” vs.  
“more general but less performance”





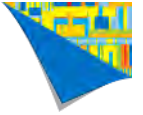


# Three debates changing our lives

**Scale:** “many but lower performance each” vs.  
“higher performance each but fewer”

**Capabilities:** “high performance but more niche” vs.  
“more general but less performance”





# Three needs affecting the debate

**Power:** 1MW/year electricity costs ~\$1M/year

**Portability:** Coding is expensive, doing it more is moreso.

**Performance portability:**

You can start a bar fights with just trying to define this. \*

*\* if the bar patrons are HPC programmers*



The background of the slide is a blue-tinted image of a microchip die, showing a complex grid of circuitry. In the top right corner, there is a white, curled-up corner effect, revealing a small portion of a colorful, abstract pattern underneath.

## ***vision***

span from *few cores* to *many cores*  
with consistent models,  
languages, tools, and techniques





**INTEL® XEON PHI™  
PROCESSOR  
HIGH PERFORMANCE  
PROGRAMMING  
KNIGHTS LANDING EDITION**

Jim Jeffers | James Reinders | Avinash Sodani

**MK**  
MOBBAN KUTTMANN

## Knights Landing 2<sup>nd</sup> Generation Intel® Xeon Phi™

### ALL ABOUT PARALEL PROGRAMMING

Threading, Vectorization, Data Locality  
Fortran, C, C++ (plus a little Python)  
OpenMPI, MPI, TBB

### New with Knights Landing:

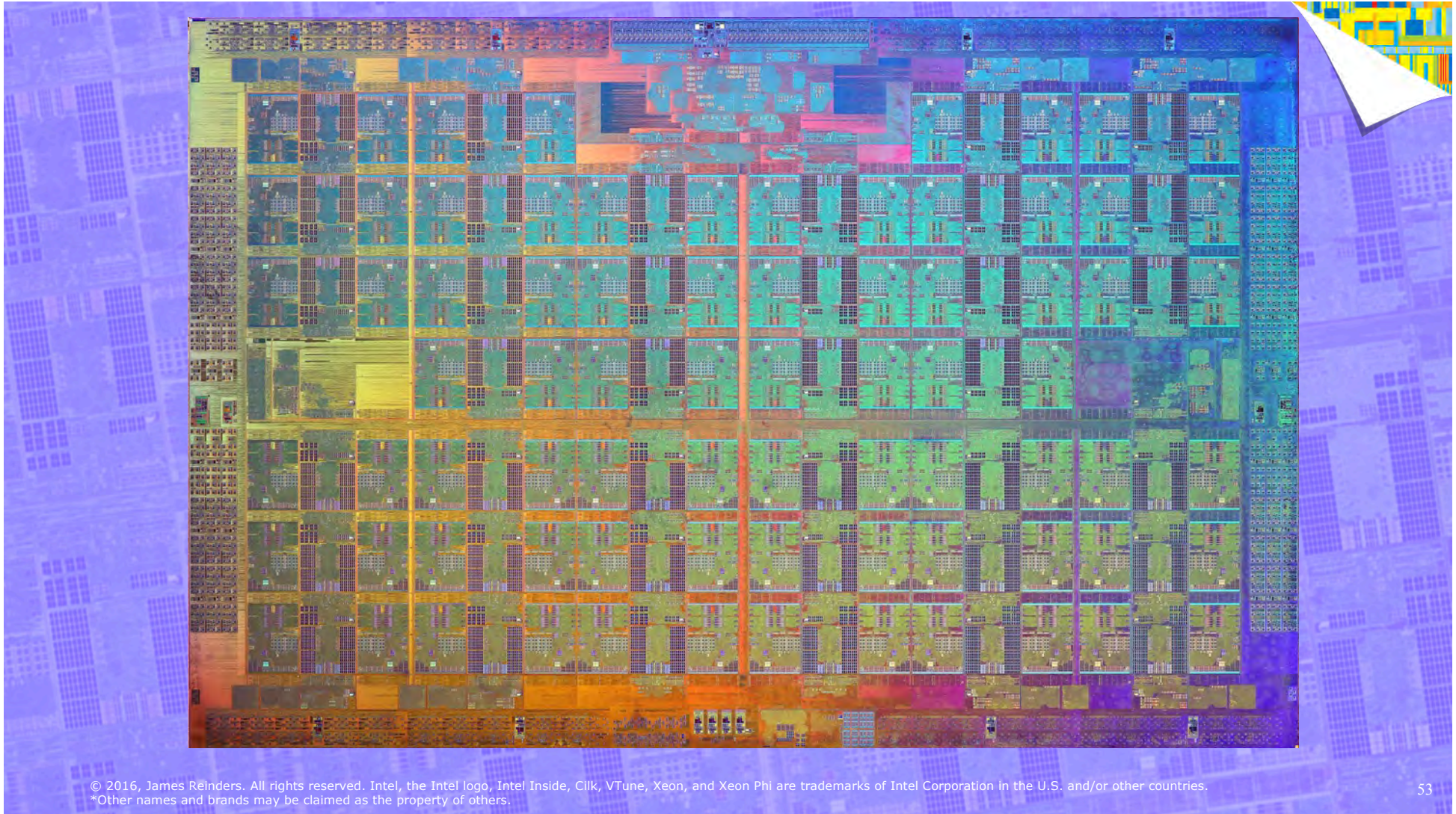
*AVX-512,*

*High Bandwidth Memory (MCDRAM),*

*Cluster Mode,*

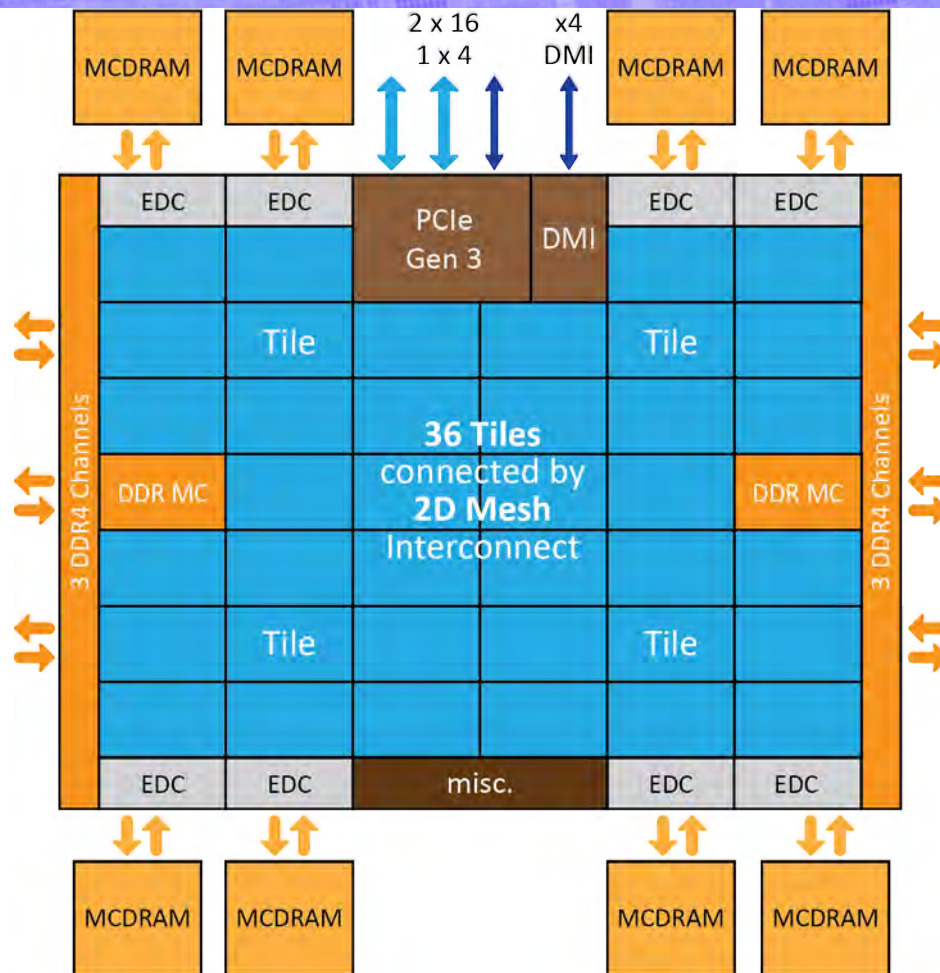
*Omni-Path*





© 2016, James Reinders. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.  
\*Other names and brands may be claimed as the property of others.

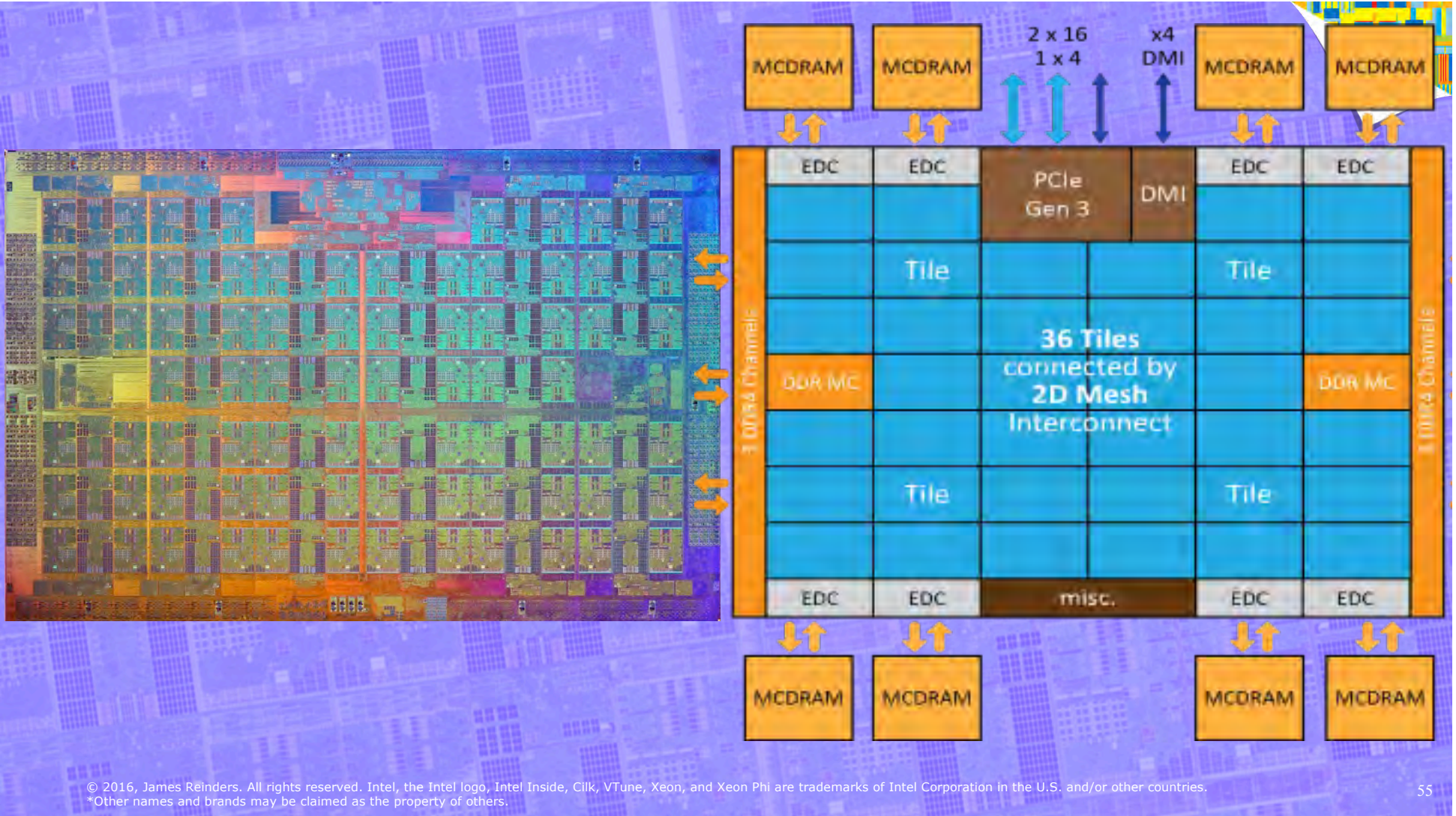




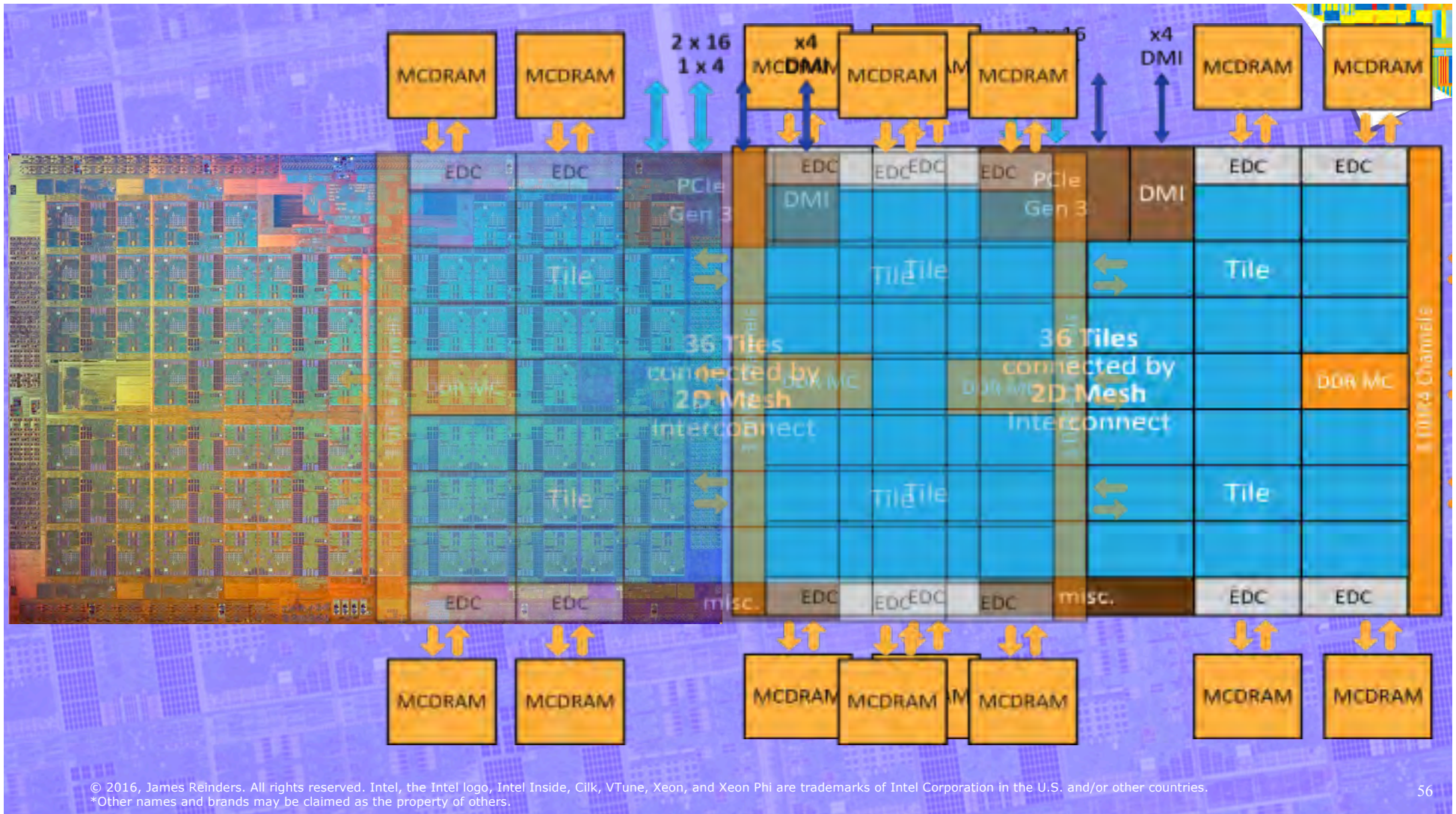
## 2<sup>nd</sup> Generation Intel® Xeon Phi™ Products

Codename:

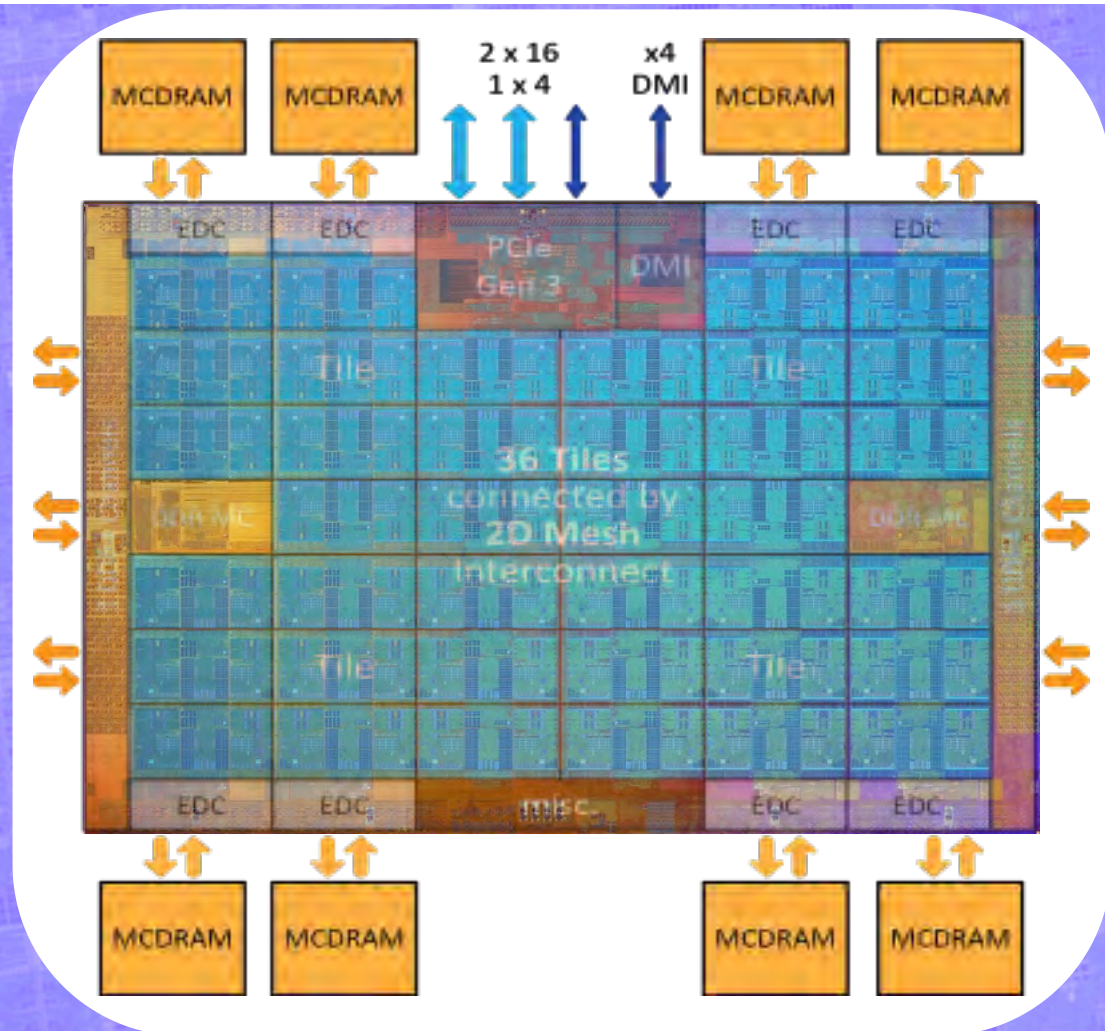
# Knights Landing

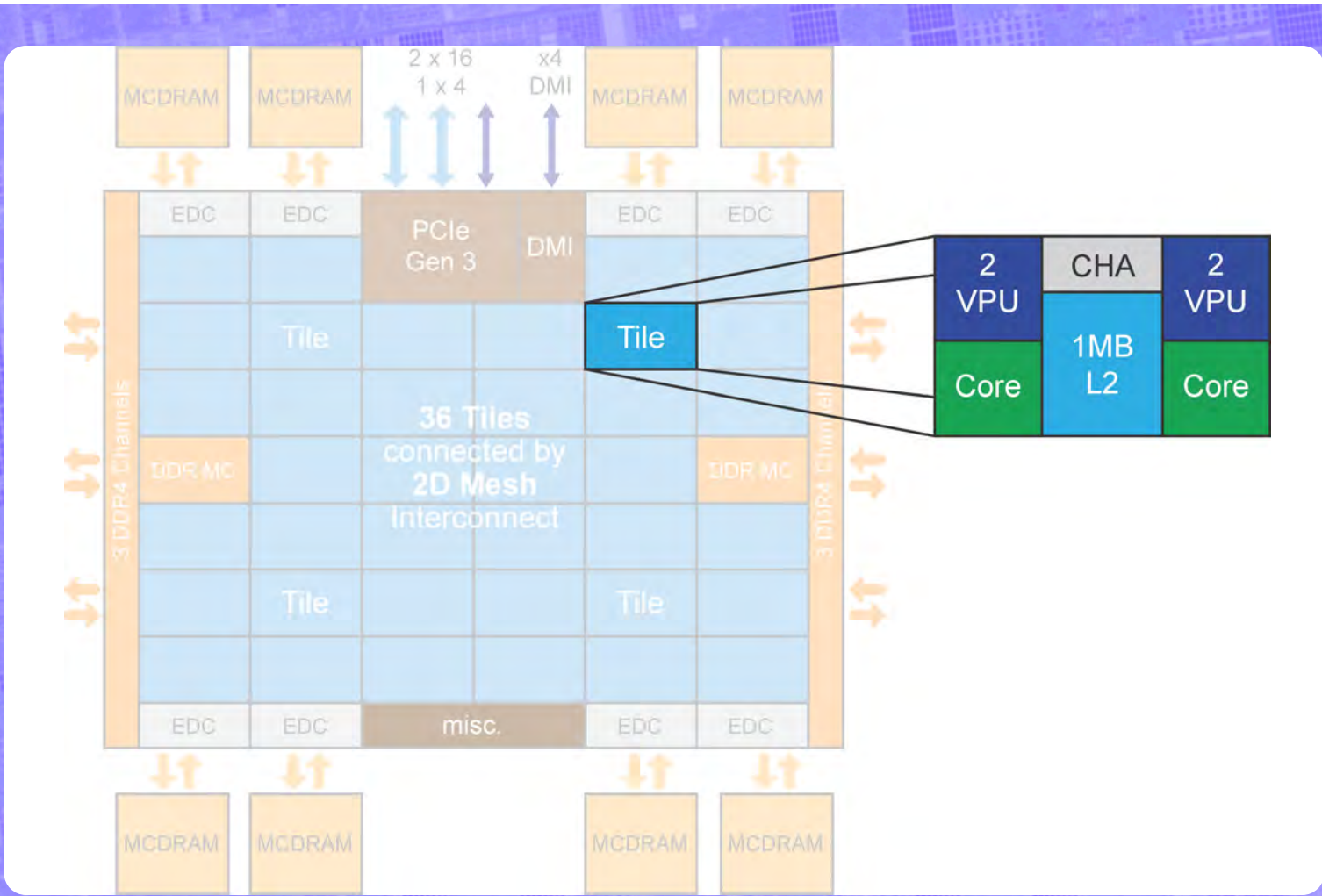














*AVX-512, High Bandwidth Memory, Cluster Mode, Omni-Path*

## Knights Landing: Next Intel® Xeon Phi™ Processor

Intel® Many-Core Processor targeted for HPC and Supercomputing

First self-boot Intel® Xeon Phi™ processor that is **binary compatible** with main line IA. Boots standard OS.

**Significant improvement in scalar and vector performance**

Integration of **Memory on package**: innovative memory architecture for high bandwidth and high capacity.

Integration of **Fabric on package**



### Three products

KNL Self-Boot

KNL Self-Boot w/ Fabric

KNL Card

(Baseline)

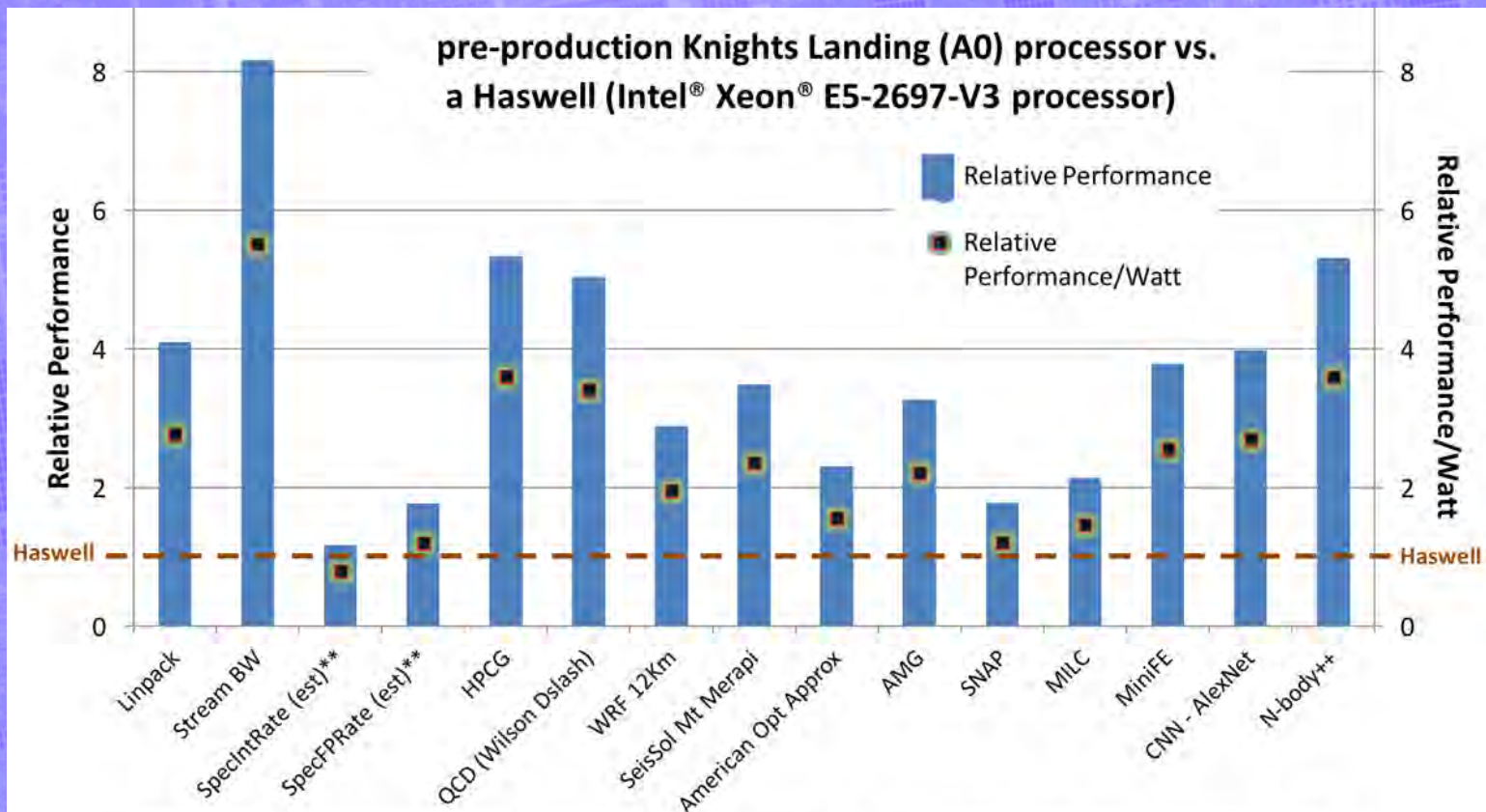
(Fabric Integrated)

(PCIe-Card)

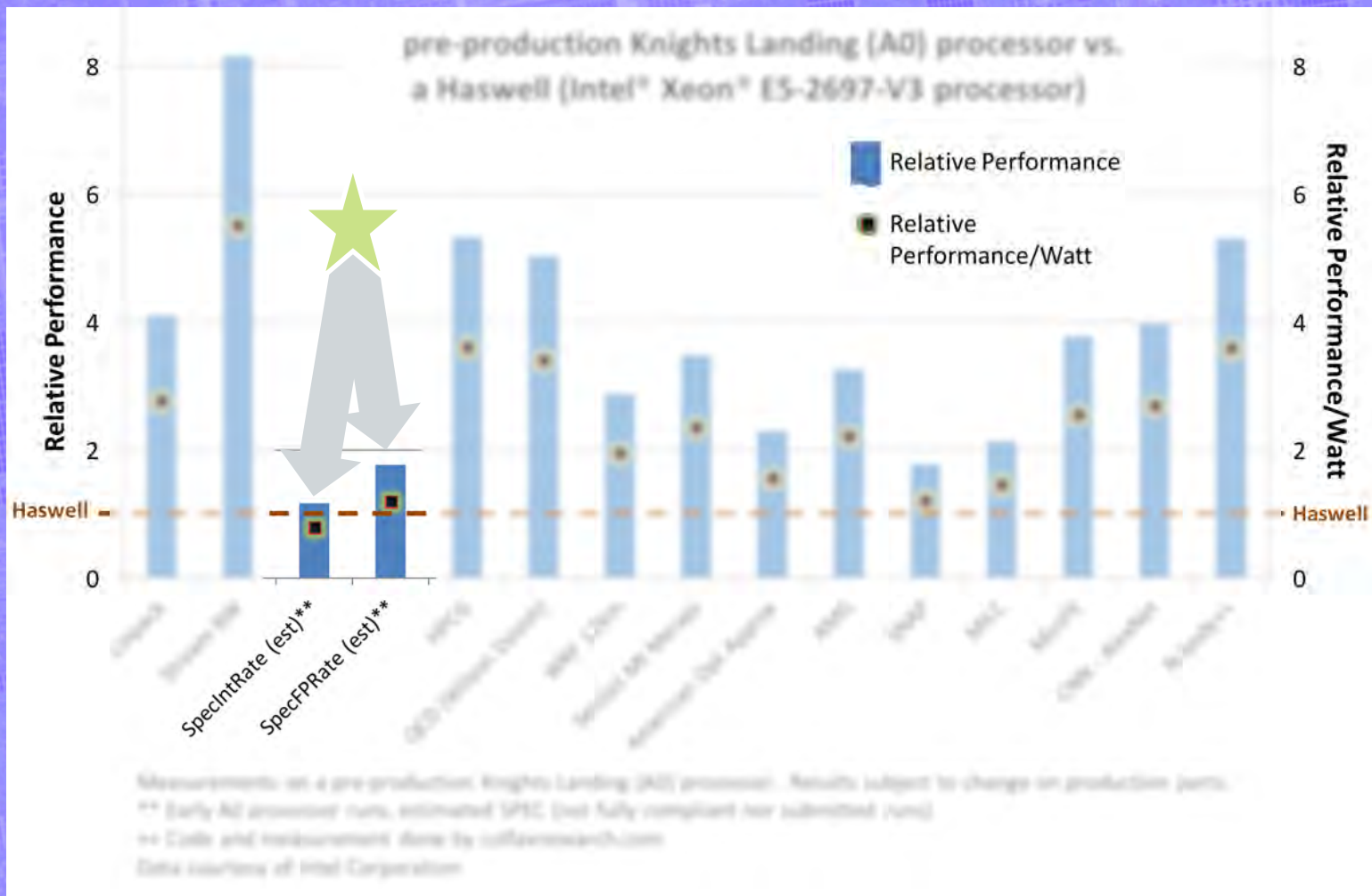
Potential future options subject to change without notice.

All timeframes, features, products and dates are preliminary forecasts and subject to change without further notification.





Measurements on a pre-production Knights Landing (A0) processor. Results subject to change on production parts.  
 \*\* Early A0 processor runs, estimated SPEC (not fully compliant nor submitted runs)  
 ++ Code and measurement done by colfaxresearch.com  
 Data courtesy of Intel Corporation

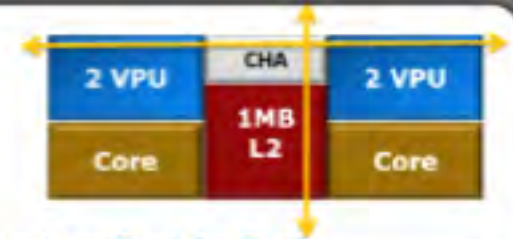




# AVX-512, High Bandwidth Memory, Cluster Mode, Omni-Path

512-bit  
vectors  
via  
AVX-512  
means:  
High  
performance  
and  
Binary  
compatibility

**KNL Tile:** 2 Cores, each with 2 VPU  
1M L2 shared between two Cores



**Core:** Changed from Knights Corner (KNC) to KNL. Based on 2-wide OoO Silvermont™ Microarchitecture, but with many changes for HPC.

4 thread/core. Deeper OoO. Better RAS. Higher bandwidth. Larger TLBs.

**2 VPU** 2x AVX512 units. 32SP/16DP per unit. X87, SSE, AVX1, AVX2 and EMU

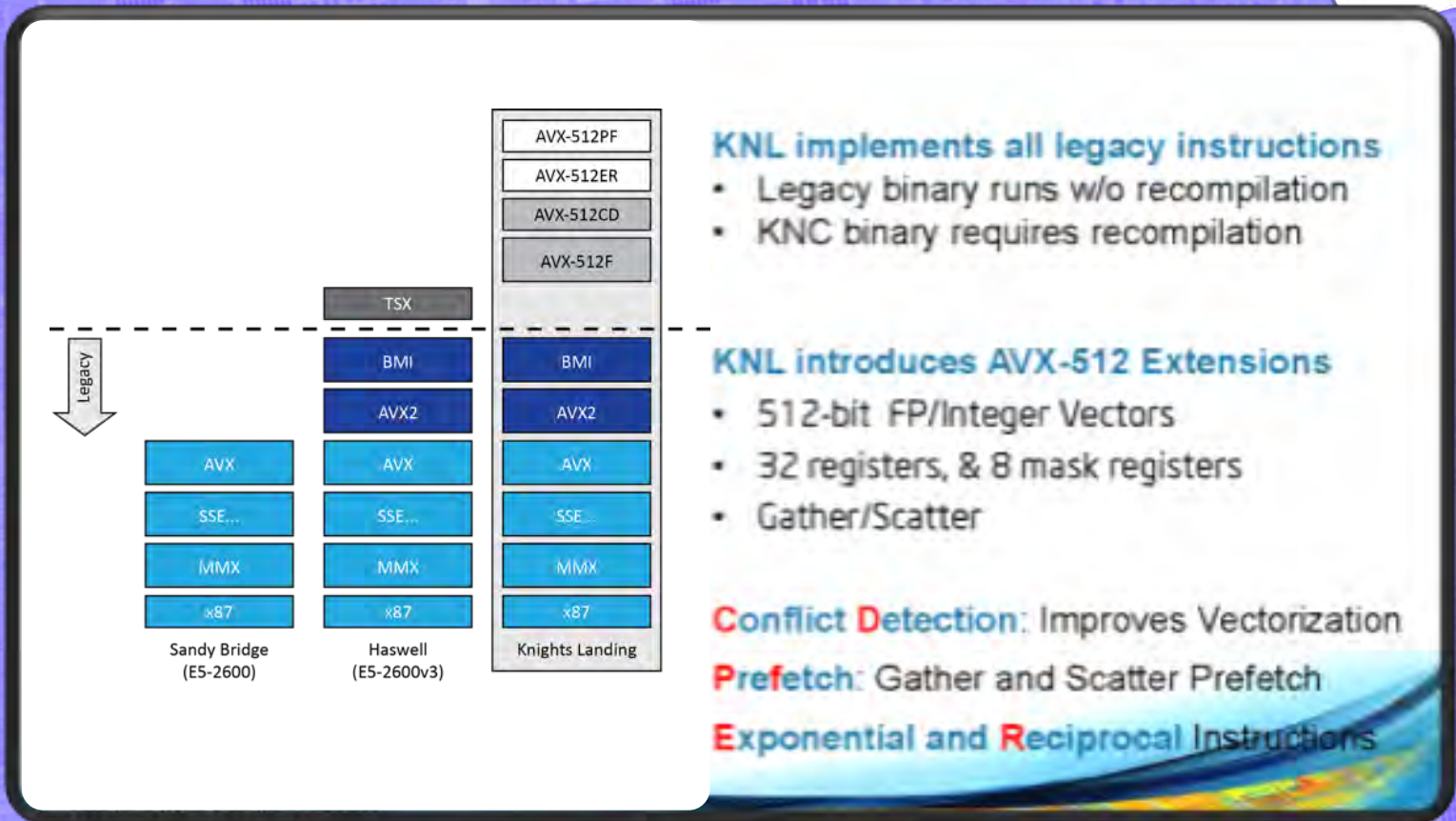
**L2:** 1MB 16-way. 1 Line Read and ½ Line Write per cycle. Coherent across all Tiles

**CHA:** Caching/Home Agent. Distributed Tag Directory to keep L2s coherent. MESIF protocol. 2D-Mesh connections for Tile



# AVX-512, High Bandwidth Memory, Cluster Mode, Omni-Path

Support of vectors under 512-bits in size means: **Binary compatibility**



## KNL implements all legacy instructions

- Legacy binary runs w/o recompilation
- KNC binary requires recompilation

## KNL introduces AVX-512 Extensions

- 512-bit FP/Integer Vectors
- 32 registers, & 8 mask registers
- Gather/Scatter

**Conflict Detection:** Improves Vectorization

**Prefetch:** Gather and Scatter Prefetch

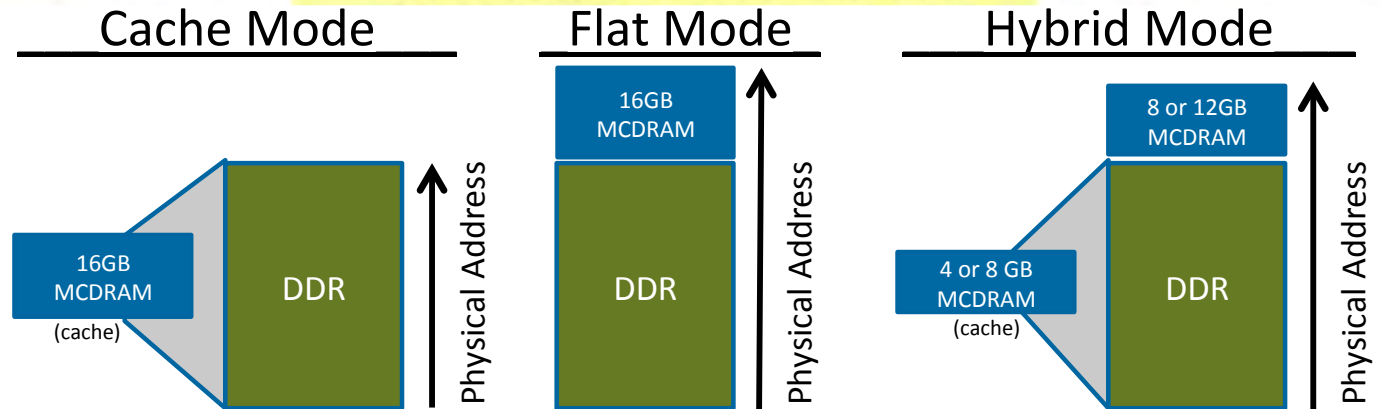
**Exponential and Reciprocal Instructions**

**Figure 2.1**  
**(KNL book)**

16MB  
on package  
DRAM  
means:  
Performance  
with  
options

## Memory Modes

Three Modes. Selected at boot



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

- MCDRAM as regular memory
- SW-Managed
- Same address space

- Part cache, Part memory
- 25% or 50% cache
- Benefits of both

Figure 4.13  
(KNL book)



Cache mode only available when there is DDR

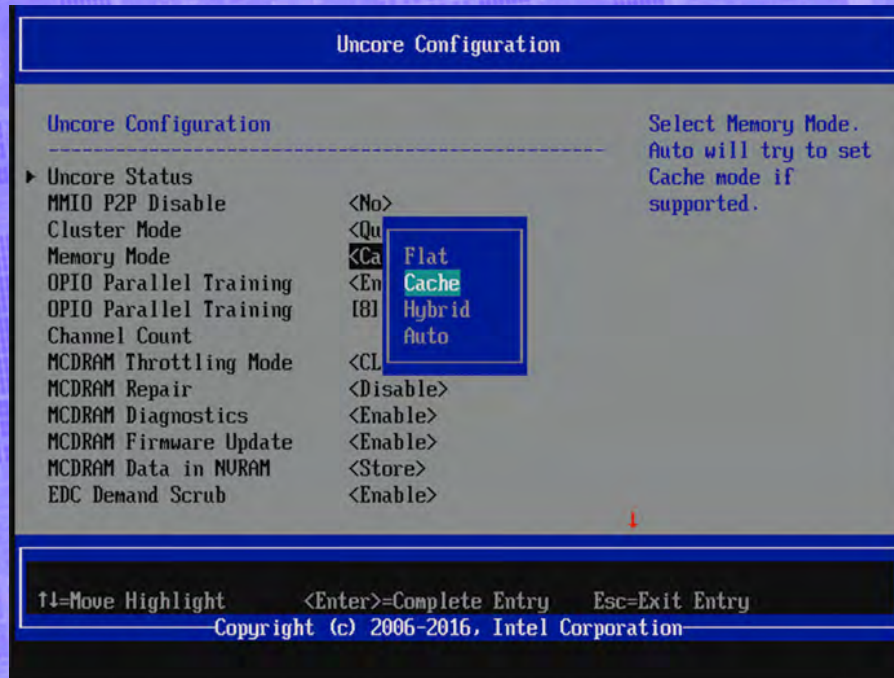


Figure 3.24  
(KNL book)



Cache mode only available when there is DDR

Only available when Memory Mode = Hybrid

**Uncore Configuration**

Uncore Configuration

Select Memory Mode. Auto will try to set Cache mode if supported.

► Uncore Status	
MMIO P2P Disable	<No>
Cluster Mode	<Quadrant>
Memory Mode	<Cache> <b>Flat</b>
OPIO Parallel Training	<Enable> <b>Cache</b>
OPIO Parallel Training	[8]
Channel Count	
MCDRAM Throttling Mode	<CLTT>
MCDRAM Repair	<Disable>
MCDRAM Diagnostics	<Enable>
MCDRAM Firmware Update	<Enable>
MCDRAM Data in NURAM	<Store>
EDC Demand Scrub	<Enable>

↑↓=Move Highlight    <Enter>=Complete Entry    Esc=Exit Entry  
Copyright (c) 2006-2016, Intel Corporation

**Uncore Configuration**

Uncore Configuration

Select the MCDRAM size that will be used as Cache Memory; the remaining MCDRAM memory will be used as Flat Memory. 50% is the only supported option if number of MCDRAMs is less than or equal to 4.

► Uncore Status	
MMIO P2P Disable	<No>
Cluster Mode	<Quadrant>
Memory Mode	
MCDRAM Cache Size	<b>25% of MCDRAM size</b>
Treat MCDRAM as	<b>50% of MCDRAM size</b>
Hot-Pluggable Memory	
OPIO Parallel Training	<Enable>
OPIO Parallel Training	[8]
Channel Count	
MCDRAM Throttling Mode	<CLTT>
MCDRAM Repair	<Disable>
MCDRAM Diagnostics	<Enable>

↑↓=Move Highlight    <Enter>=Complete Entry    Esc=Exit Entry  
Copyright (c) 2006-2016, Intel Corporation

Figures 3.24 and 3.25 (KNL book)

Standard  
“NUMA”  
node  
recognition  
by BIOS, OS,  
and  
applications.

Systems will  
eventually  
have this as  
a “norm.”

## Flat MCDRAM: SW Architecture

MCDRAM exposed as a separate NUMA node



Memory allocated in DDR by default → Keeps non-critical data out of MCDRAM.

Apps explicitly allocate critical data in MCDRAM. Using two methods:

- “Fast Malloc” functions in High BW library (<https://github.com/memkind>)
  - Built on top to existing *libnuma* API
- “FASTMEM” Compiler Annotation for Intel Fortran

**Flat MCDRAM with existing NUMA support in Legacy OS**



C/C++  
“high  
bandwidth”  
malloc  
or new

Fortran  
“high  
bandwidth”  
allocatables

## Flat MCDRAM SW Usage: Code Snippets

C/C++ ([\\*https://github.com/memkind](https://github.com/memkind))

Allocate into DDR

```
float *fv;  
fv = (float *)malloc(sizeof(float)*100);
```



Allocate into MCDRAM

```
float *fv;  
fv = (float *)hbw_malloc(sizeof(float) * 100);
```

Intel Fortran

Allocate into MCDRAM

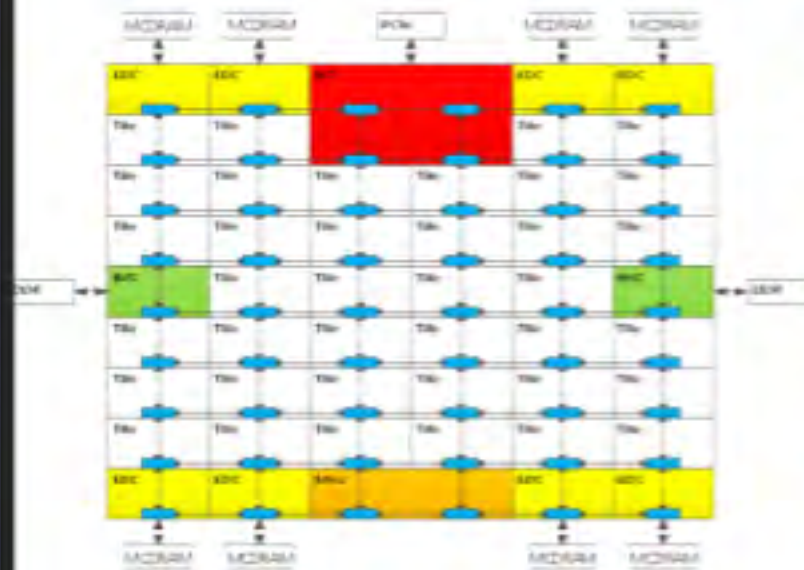
```
c  Declare arrays to be dynamic  
   REAL, ALLOCATABLE :: A(:)  
!DEC$ ATTRIBUTES, FASTMEM :: A  
  
   NSIZE=1024  
c  allocate array 'A' from MCDRAM  
c  
   ALLOCATE (A(1:NSIZE))
```



# AVX-512, High Bandwidth Memory, **Cluster Mode**, Omni-Path

**Mesh interconnect** means:  
**Higher Performance** with options

## KNL Mesh Interconnect



### Mesh of Rings

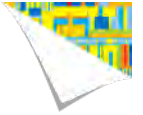
- Every row and column is a (half) ring
- YX routing: Go in Y → Turn → Go in X
- Messages arbitrate at injection and on turn

### Cache Coherent Interconnect

- MESIF protocol (F = Forward)
- Distributed directory to filter snoops

### Three Cluster Modes

- (1) All-to-All
- (2) Quadrant
- (3) Sub-NUMA Clustering



# Tweaks

In the end, selecting a memory mode is a PERFORMANCE TWEAK.

**Code changes may be needed** – unless the working set “fits” or can easily be moved by rules.

Previously, such changes in policies were only decided when the machine was *designed*.

**NEW: Intel made this configurable at BOOT time.** Unprecedented configurability. Does it matter enough to bother?

My opinion: Probably – but time will tell.

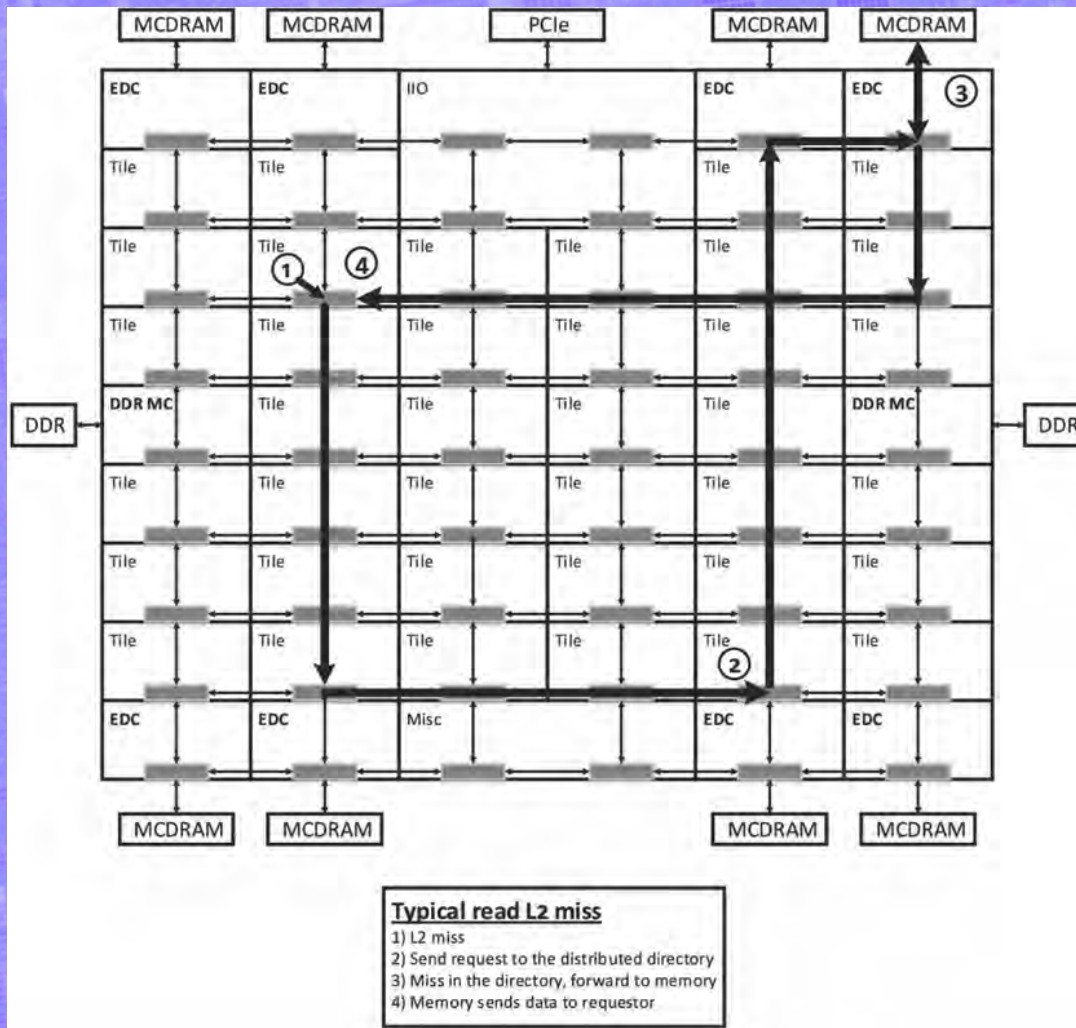


**Figure 3.23**  
**(KNL book)**



## Cluster Mode: ALL TO ALL

*(only used  
is DRAM is  
unbalanced)*

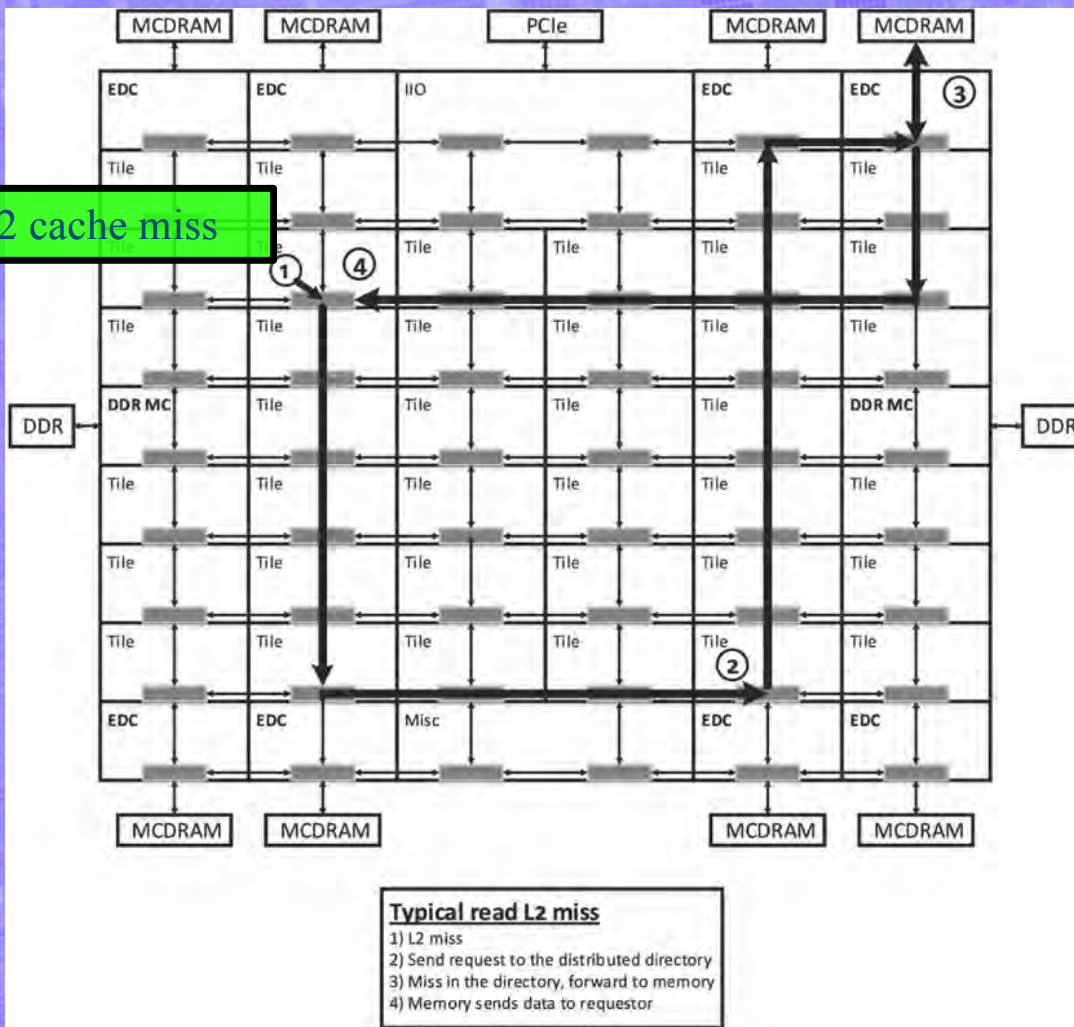


**Figure 4.5**  
**(KNL book)**

**Cluster Mode:  
ALL TO ALL**

*(only used  
is DRAM is  
unbalanced)*

(1) L2 cache miss

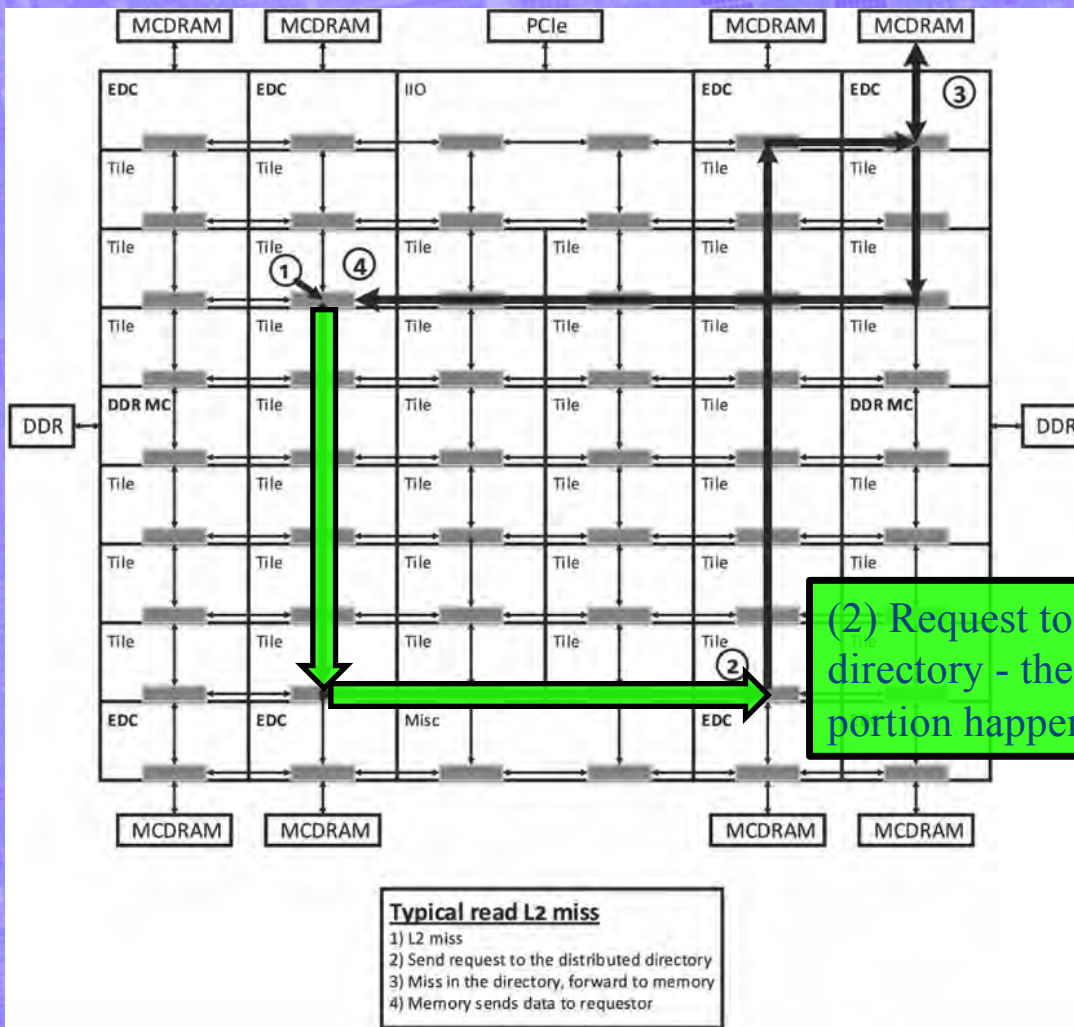


**Figure 4.5  
(KNL book)**



## Cluster Mode: ALL TO ALL

*(only used  
is DRAM is  
unbalanced)*



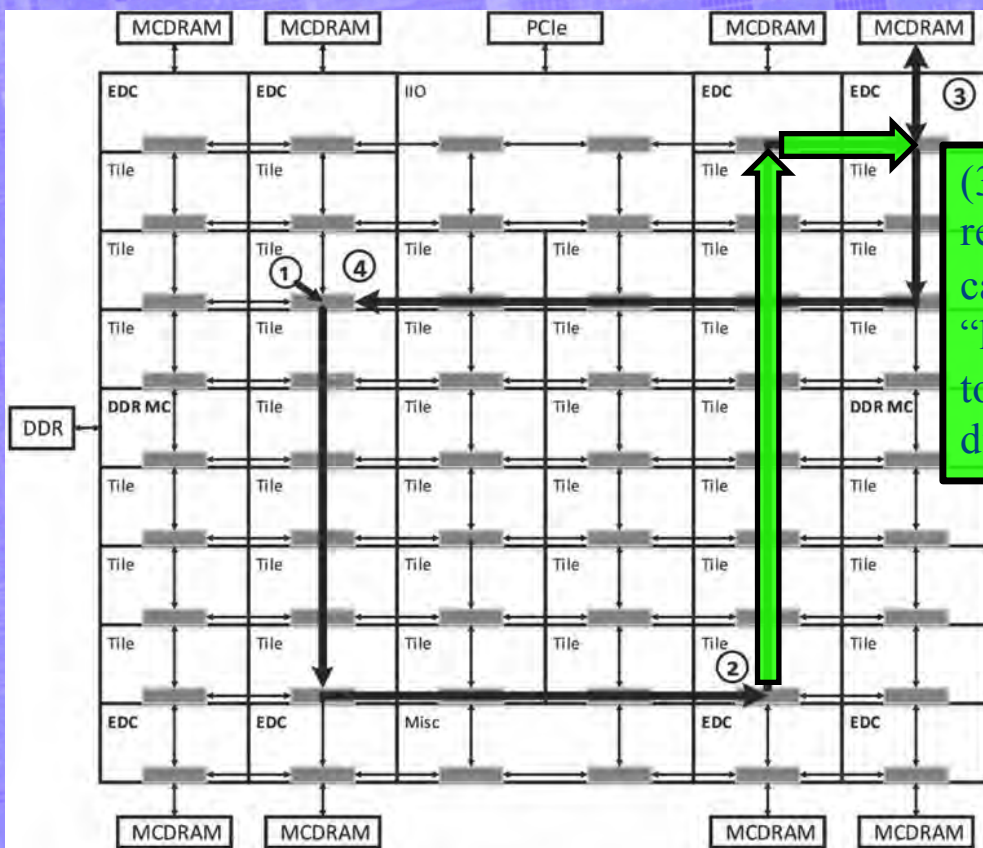
(2) Request to the distributed directory - the “responsible” portion happens to be here

**Figure 4.5  
(KNL book)**



**Cluster Mode:  
ALL TO ALL**

*(only used  
is DRAM is  
unbalanced)*



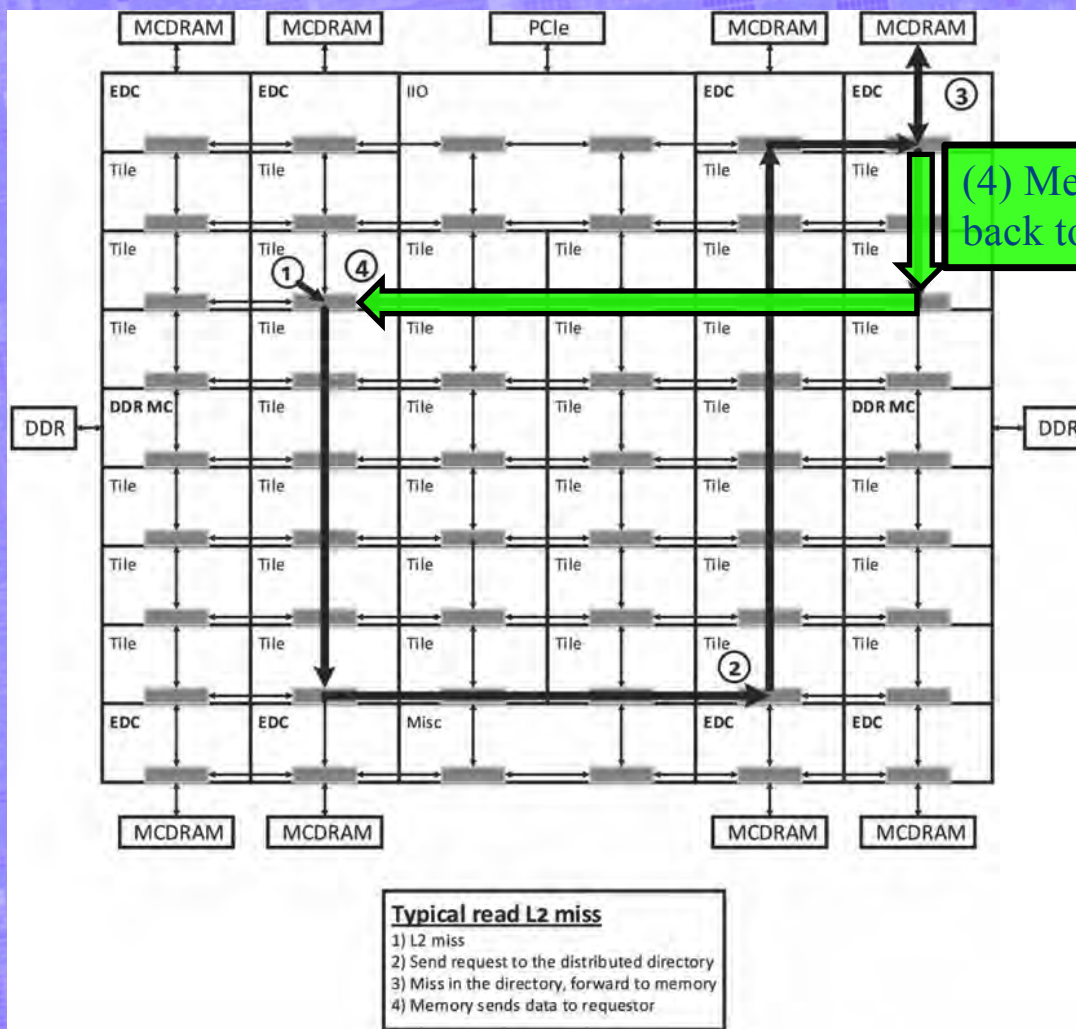
(3) Directory miss sends request to a memory – in this case in the MCDRAM (a “hit” would have sent request to another portion of the distributed L2)

**Typical read L2 miss**  
1) L2 miss  
2) Send request to the distributed directory  
3) Miss in the directory, forward to memory  
4) Memory sends data to requestor

**Figure 4.5  
(KNL book)**

## Cluster Mode: ALL TO ALL

*(only used  
is DRAM is  
unbalanced)*



(4) Memory sends data  
back to the requestor

Figure 4.5  
(KNL book)



## Cluster Mode: QUADRANT

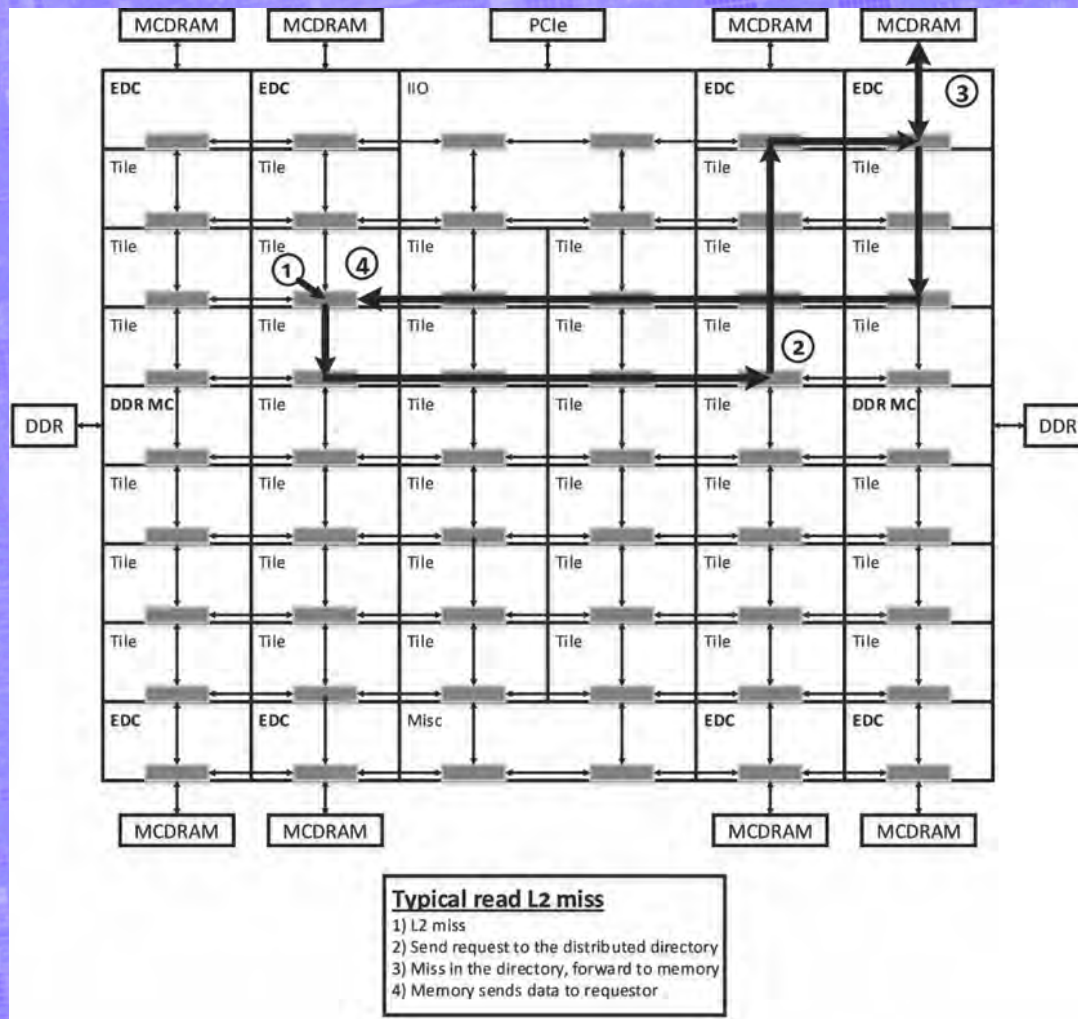
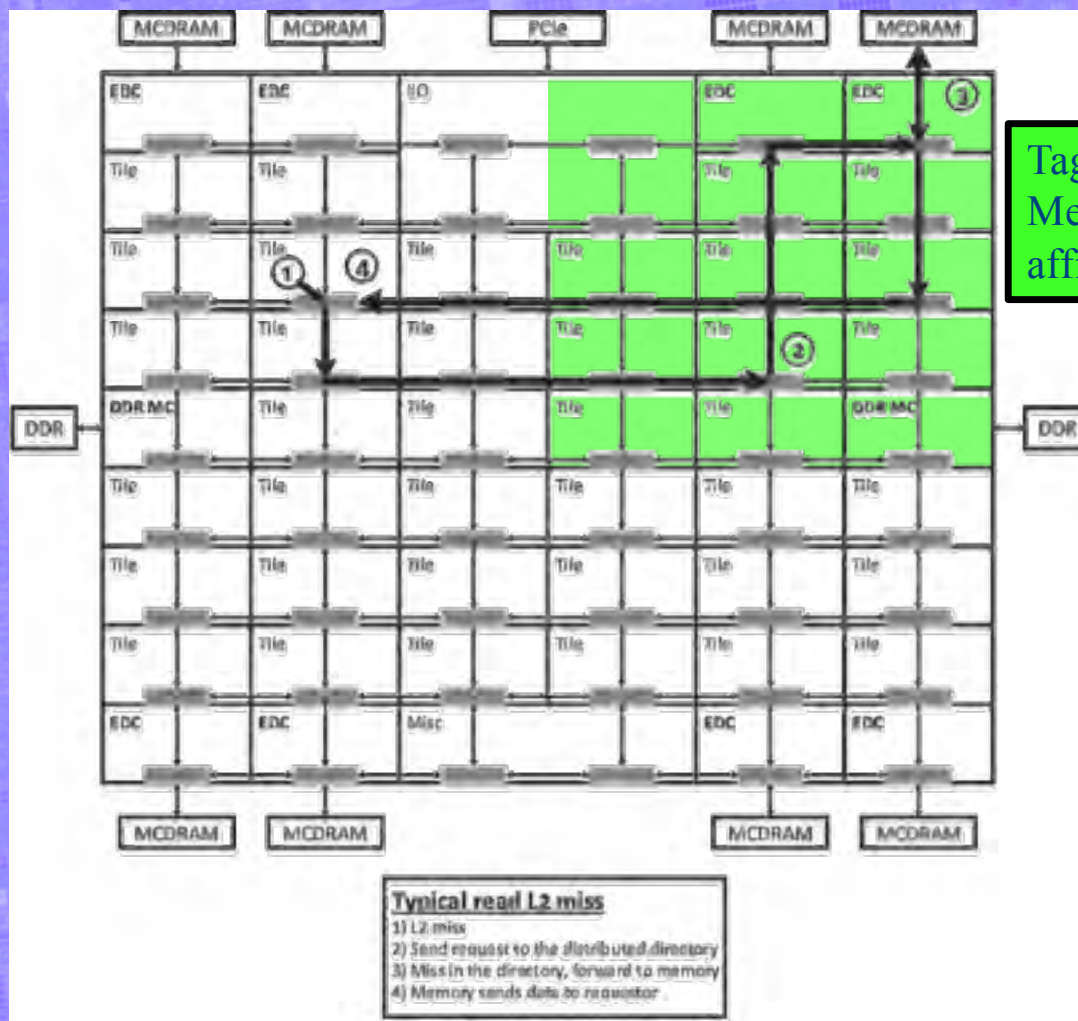


Figure 4.6  
(KNL book)



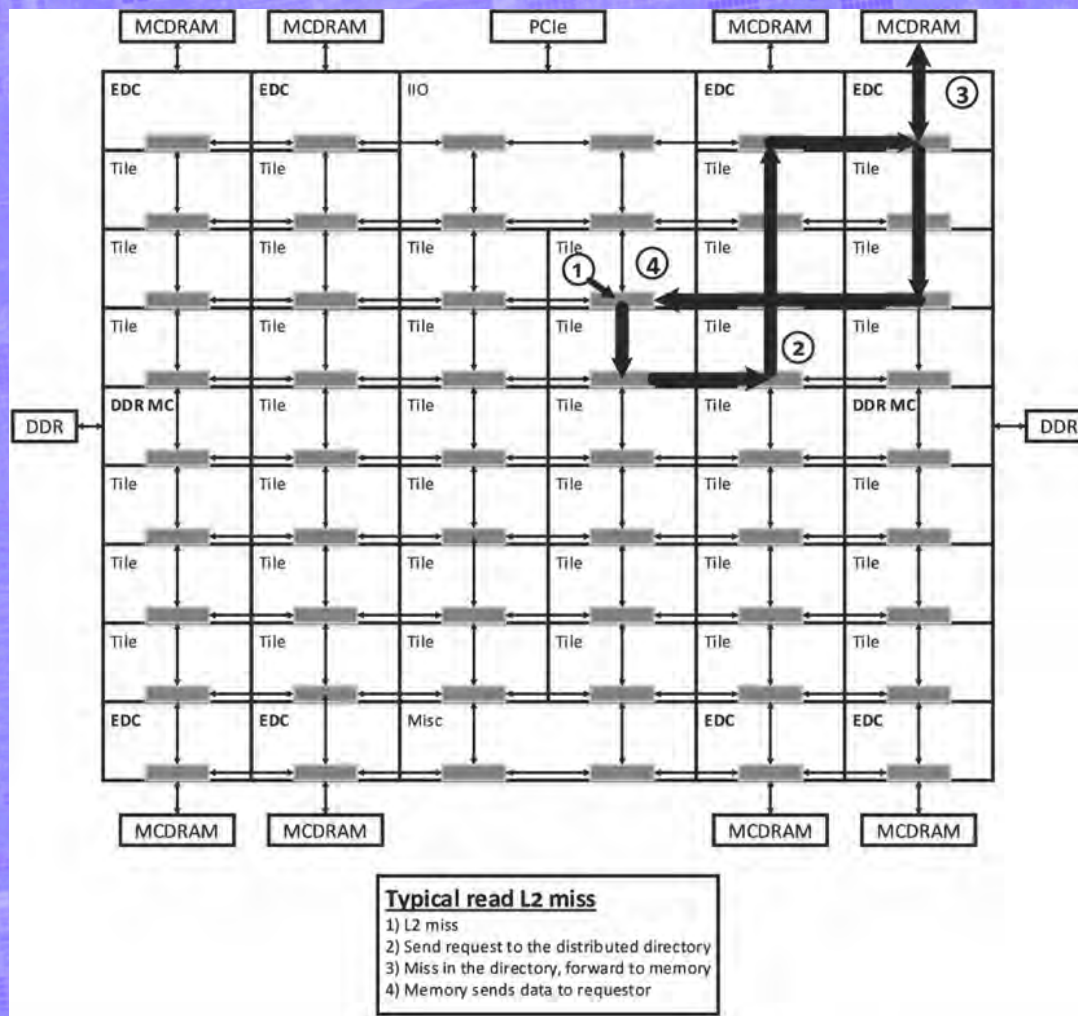
## Cluster Mode: QUADRANT



Tags and  
Memory are  
affinitized

Figure 4.6  
(KNL book)

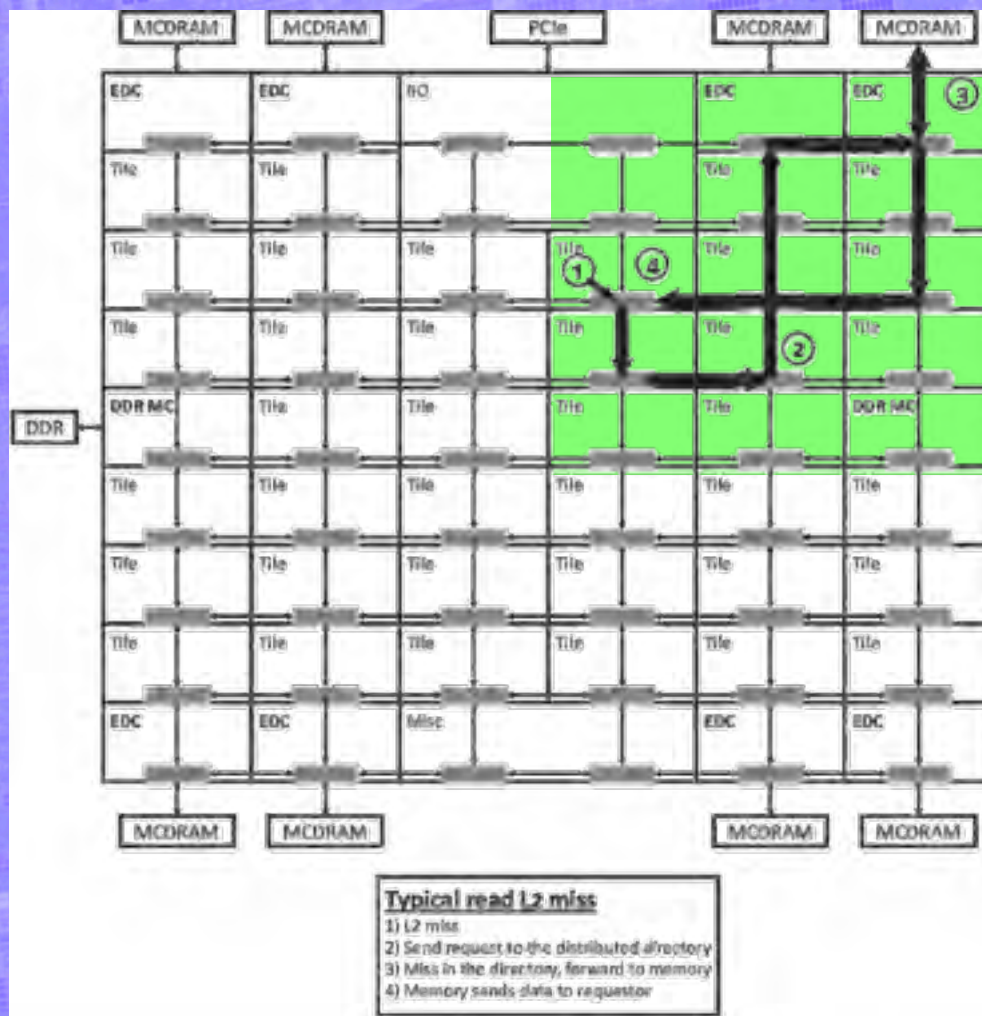
## Cluster Mode: SNC-4



**Figure 4.7**  
**(KNL book)**



**Cluster Mode:  
SNC-4**

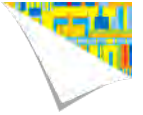


Tags and Memory  
and NUMA nodes  
(close memory) are  
affinitized

ALL memory is  
available  
COHERENTLY to  
ALL cores. Only  
change: some is *close*  
and some is *far*  
(hence, NUMA)

**Figure 4.7  
(KNL book)**





# Tweaks

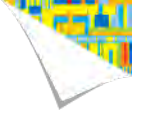
In the end, selecting a cluster mode is a PERFORMANCE TWEAK.

**Code changes should not be needed** – assuming your MPI code is written so you can parameterize `#ranks` and placement of ranks.

Previously, such changes in policies were only decided when the machine was *designed*.

**NEW: Intel made this configurable at BOOT time.** Unprecedented configurability. Does it matter enough to bother?

My opinion: Probably – but time will tell.



# Advice

Default to CACHE (memory mode) + QUADRANT (cluster mode)

Use CACHE mode, unless you have cache-unfriendly code and you will program to explicitly use FLAT memory.

Try SNC if you will divide your tiles neatly into 4x (or 2x) ranks (MPI).

# AVX-512, High Bandwidth Memory, Cluster Mode, **Omni-Path**

**Integrated Fabric** means Lower Cost, power; Higher Density; Plus the Advancements of Omni-Path in latency, bandwidth & scaling

## KNL with Omni-Path™

Omni-Path™ Fabric integrated on package

First product with integrated fabric

Connected to KNL die via 2 x16 PCIe\* ports

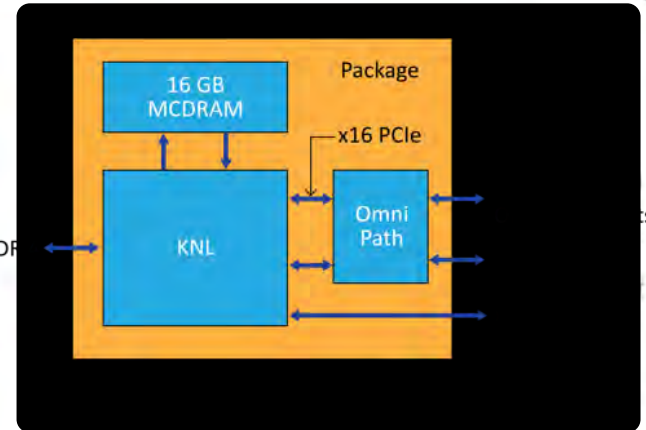
Output: 2 Omni-Path ports

- 25 GB/s/port (bi-dir)

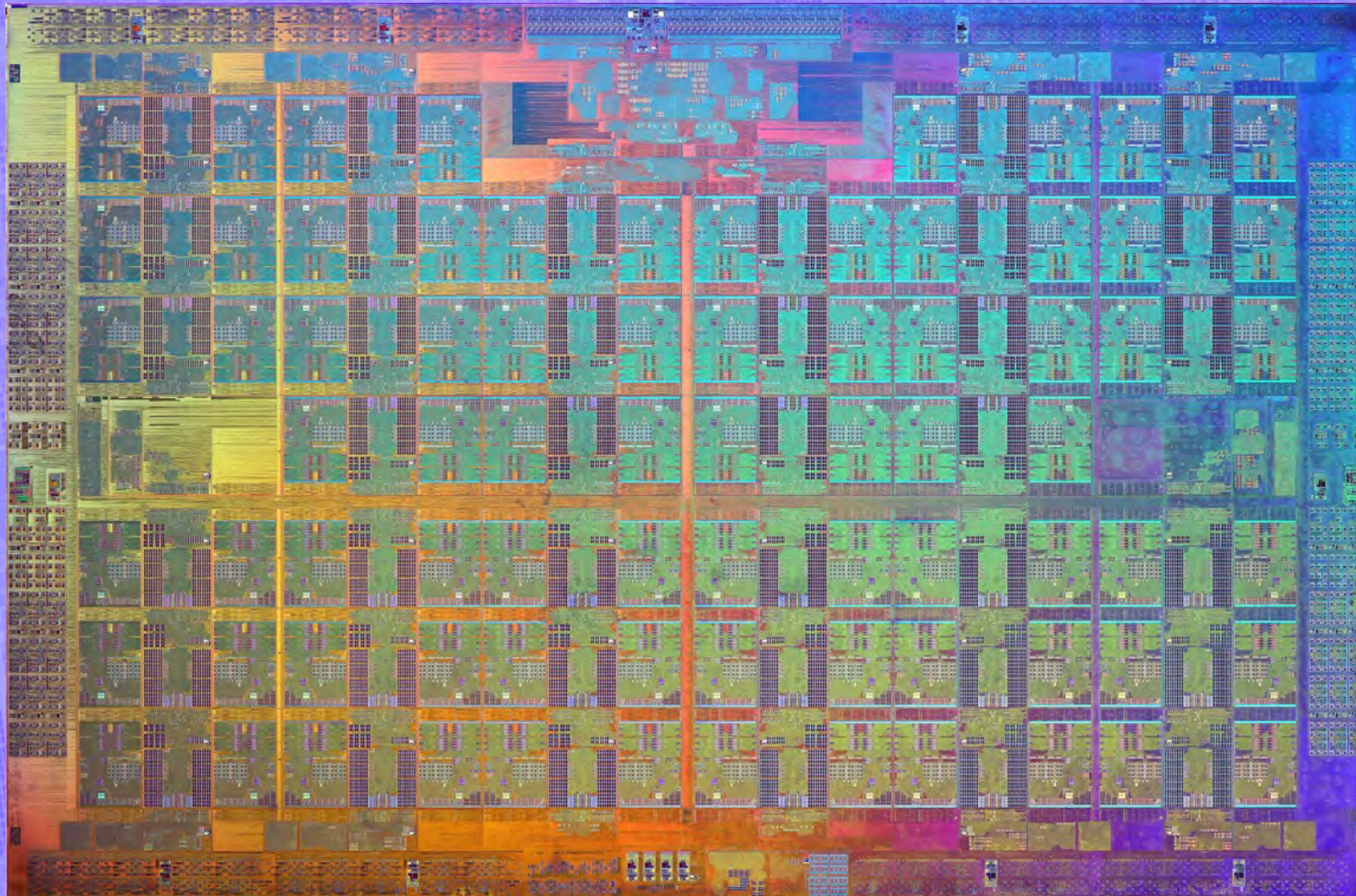
### Benefits

- Lower cost, latency and power
- Higher density and bandwidth
- Higher scalability

\*On package connect with PCIe semantics, with MDP optimizations for physical layer

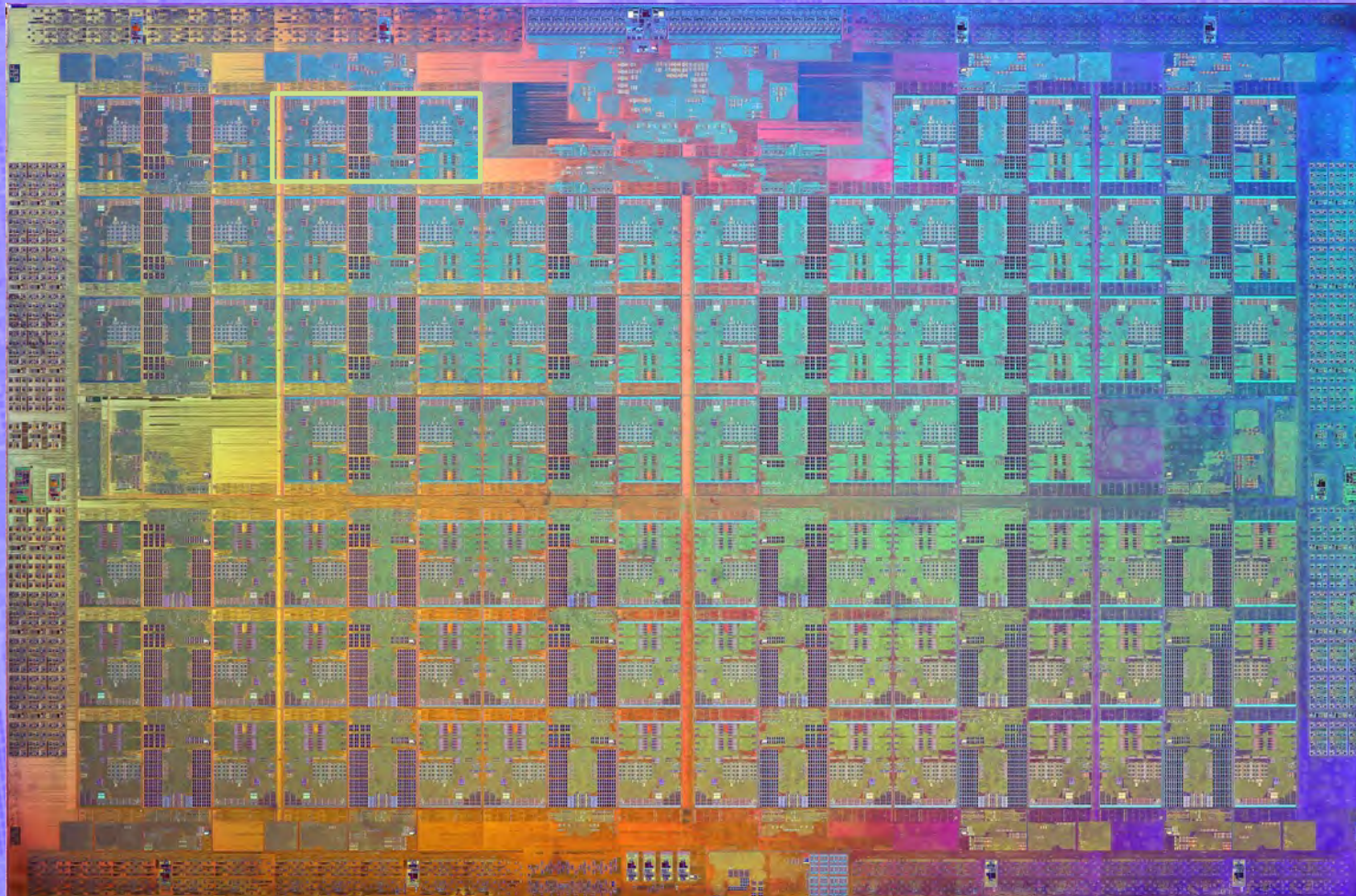




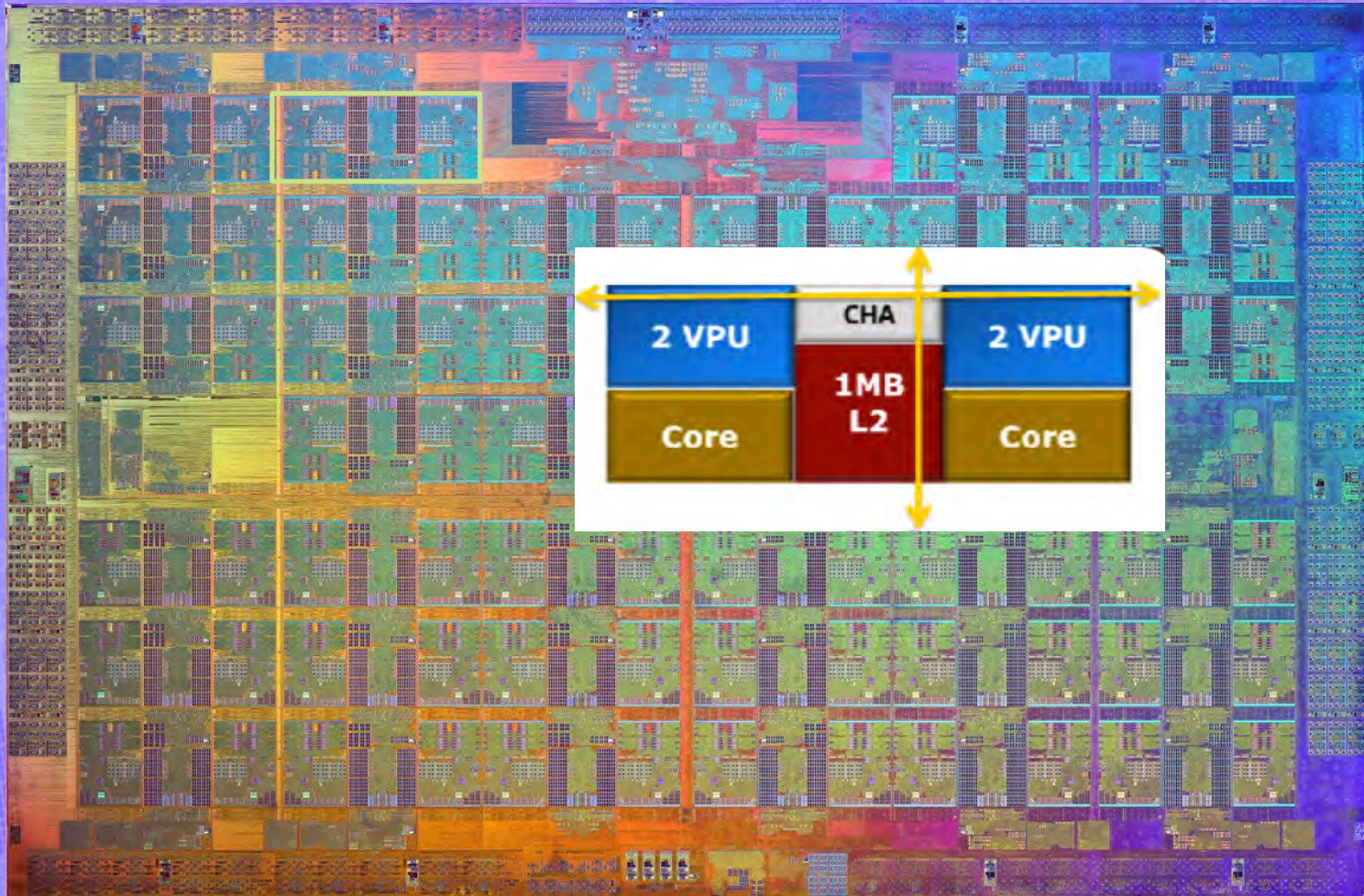


© 2016, James Reinders. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.  
\*Other names and brands may be claimed as the property of others.

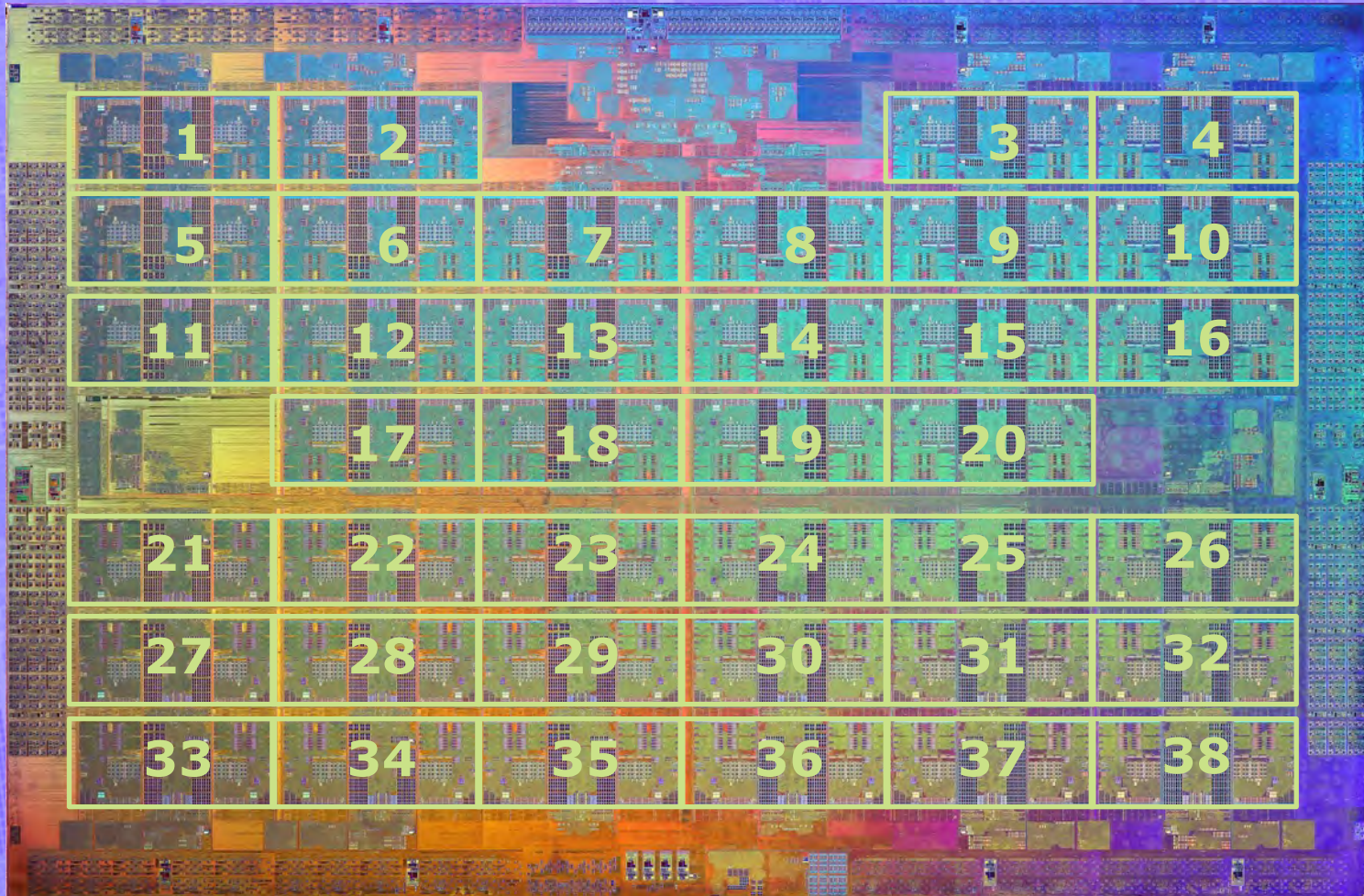
















KEEP  
CALM  
AND  
ASK SOME  
QUESTIONS



# KEEP CALM AND ASK SOME QUESTIONS



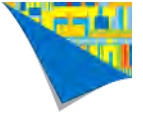


# KEEP CALM AND ASK SOME QUESTIONS



# KEEP CALM AND ASK SOME QUESTIONS





## James Reinders. Parallel Programming Enthusiast

James has been involved in multiple engineering, research and educational efforts to increase use of parallel programming throughout the industry. James worked 10,001 days as an Intel employee 1989-2016, and contributed to numerous projects including the world's first TeraFLOP/s supercomputer (ASCI Red), first 3 TeraFLOP/s supercomputer (ASCI Red upgrade), the world's first TeraFLOP/s microprocessor (Intel® Xeon Phi™ coprocessor) and the world's first 3 TeraFLOP/s microprocessor (Intel® Xeon Phi™ Processor). James been an author on numerous technical books, including VTune™ Performance Analyzer Essentials (Intel Press, 2005), Intel® Threading Building Blocks (O'Reilly Media, 2007), Structured Parallel Programming (Morgan Kaufmann, 2012), Intel® Xeon Phi™ Coprocessor High Performance Programming (Morgan Kaufmann, 2013), Multithreading for Visual Effects (A K Peters/CRC Press, 2014), High Performance Parallelism Pearls Volume 1 (Morgan Kaufmann, Nov. 2014), High Performance Parallelism Pearls Volume 2 (Morgan Kaufmann, Aug. 2015), and Intel® Xeon Phi™ Processor High Performance Programming - Knights Landing Edition (Morgan Kaufmann, 2016).