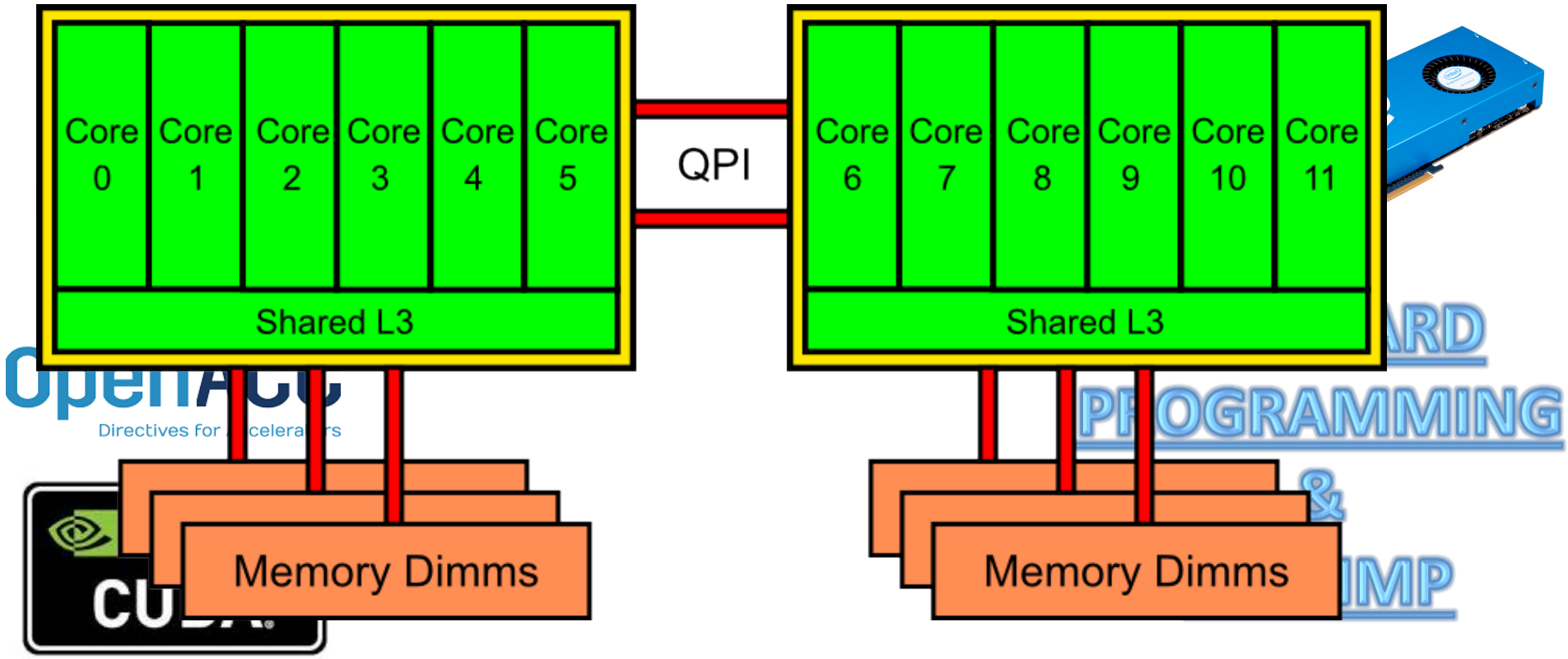
Massively Parallel Processors

Ivan Girotto – igirotto@ictp.it

Information & Communication Technology Section (ICTS)

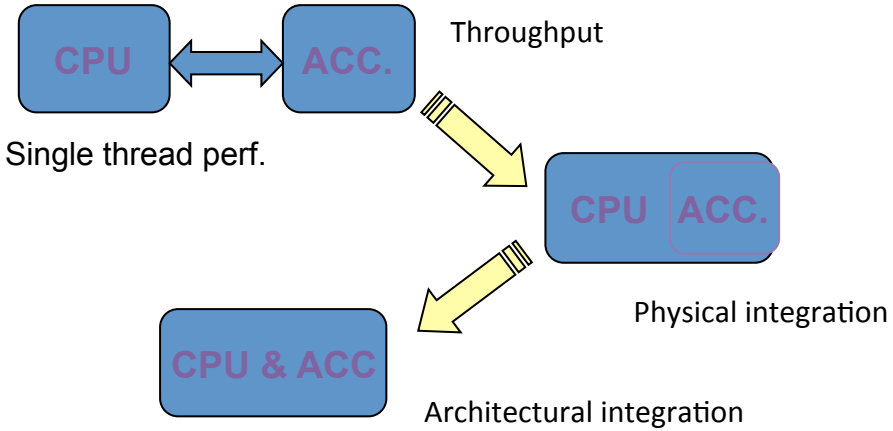
International Centre for Theoretical Physics (ICTP)

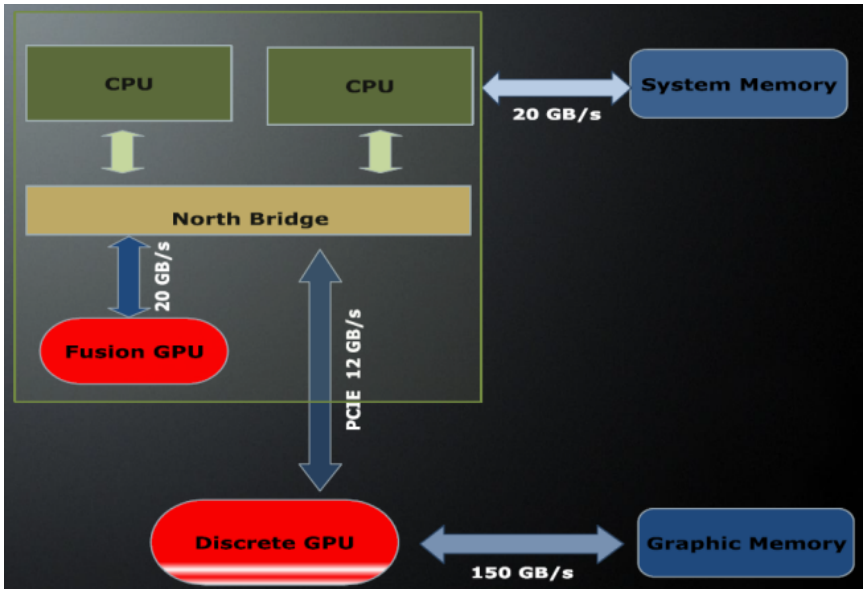
Multiple Socket CPUs + Acceleretors



Accelerated co-Processors

- A set of simplified execution units that can perform few operations (with respect to standard CPU) with very high efficiency. When combined with full featured CPU can accelerate the “nominal” speed of a system.





- **Main approaches to accelerators:**
  - **Task Parallelism (MIMD) → MIC**
  - **Data Parallelism (SIMD) → GPU**


PCI Express 3.0 Host Interface
GigaThread Engine
SMX
SMX
SMX
SMX
SMX
SMX
SMX
SMX
Memory Controller
Memory Controller
Memory Controller
Memory Controller
Memory Controller
Memory Controller
L2 Cache
SMX
SMX
SMX
SMX
SMX
SMX
SMX

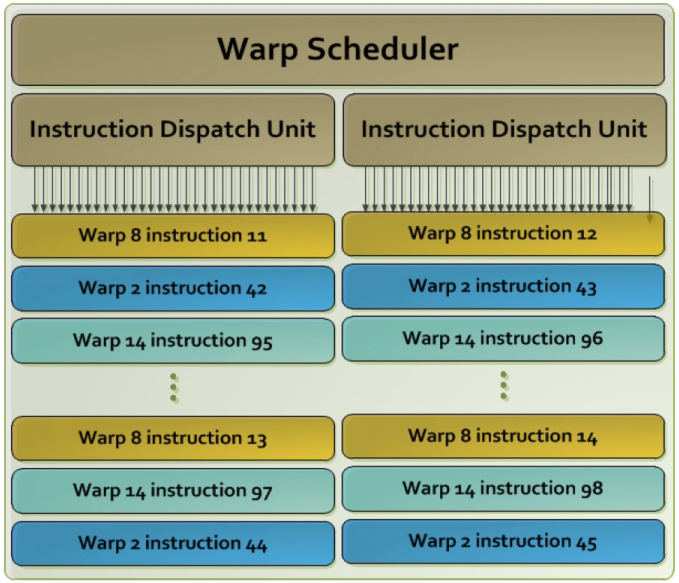
Memory Coherency (Oversimplified)

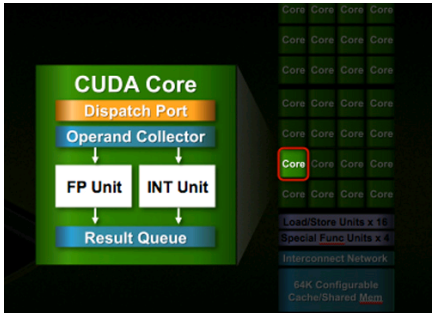
- Memory writes by CPU cores that share the same L1/L2 caches are visible to peers
- Memory writes by one CPU socket are (eventually) visible to all other peer CPUs in a multi-socket system
- Cache hardware permits multiple cores to read without penalties
- Write conflicts are solved by mutual exclusion locks, memory barrier instructions, atomic CAS and other ops

Issues With Memory Coherency

- Hardware that supports coherency is MUCH more complex than hardware that does not
- Coherency is difficult to achieve between hardware devices that are not intimately tied together in the same memory system
- Compromise solutions (e.g. GPUs) gain speed, energy efficiency, lower cost by giving up some coherency
- Future CPU/GPU hardware will likely make new compromises so that performance and efficiency keep increasing, albeit with more complexity for programmers
- Future memory systems will have more complicated and deeper memory hierarchies, and bigger performance costs when not used correctly...

SMX Processor & Warp Scheduler & Core



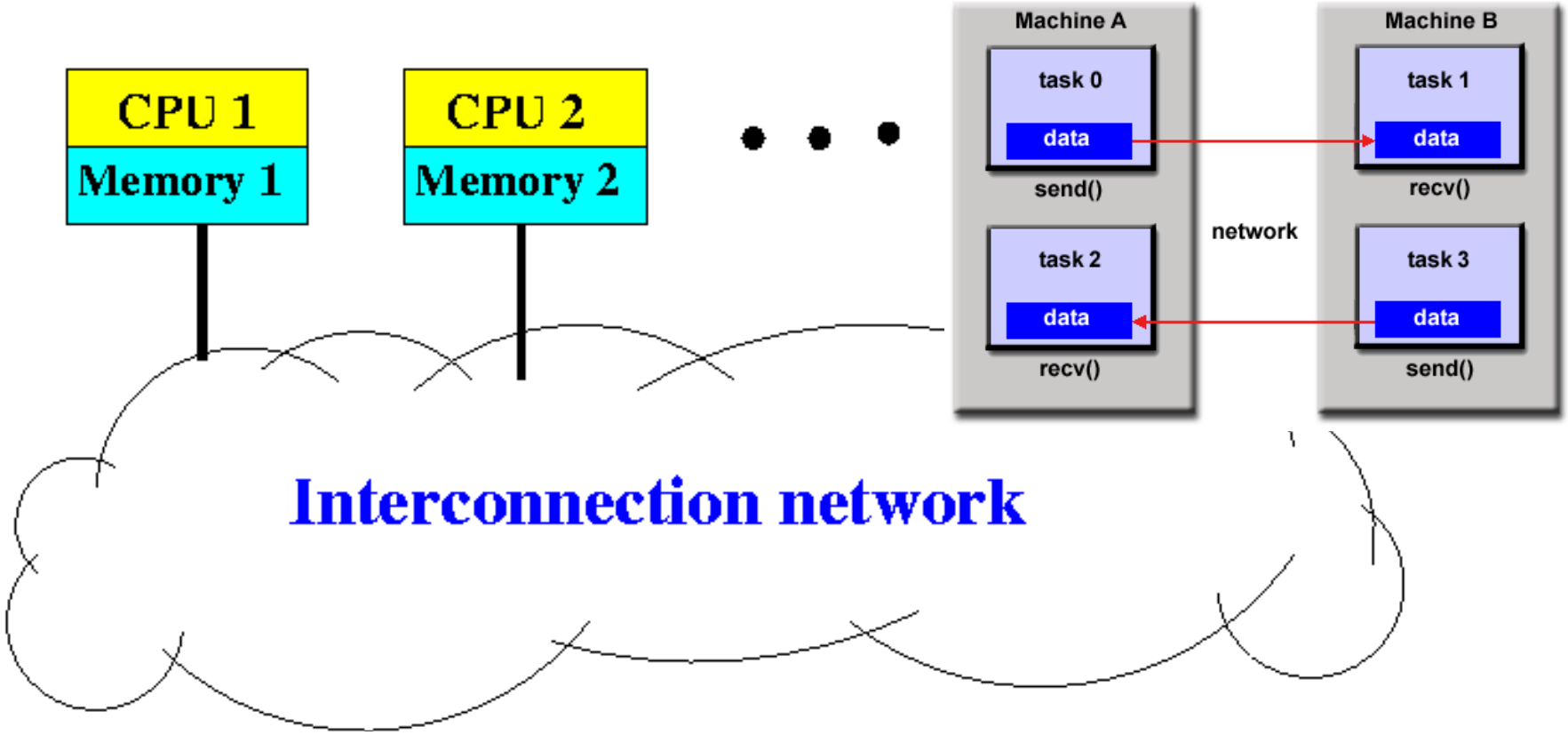


What is CUDA?

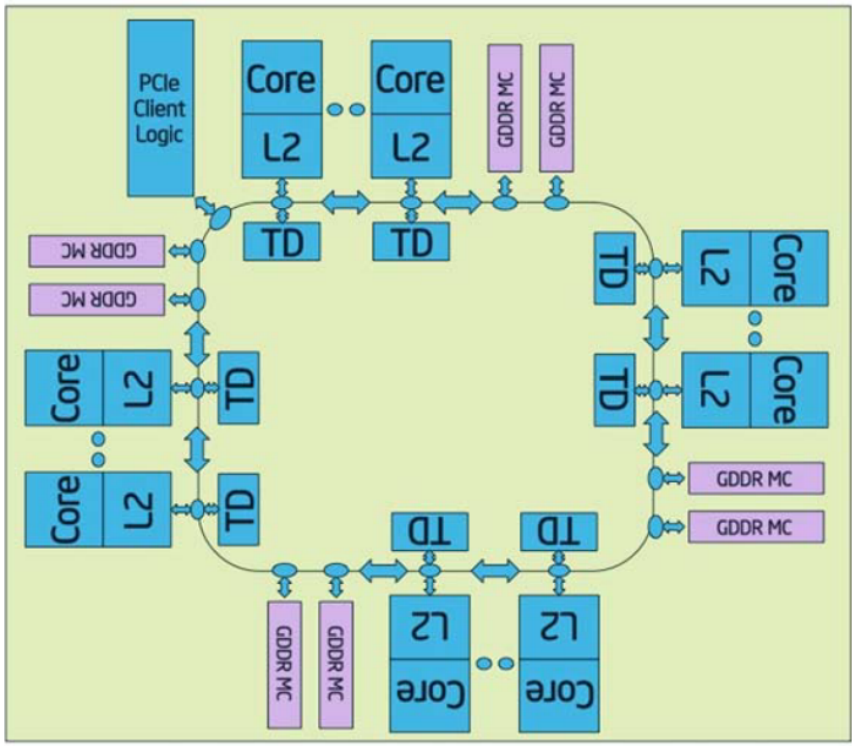
- **NVIDIA** compute architecture
- Quickly maturing software development capability provided free of charge by NVIDIA
- C and C++ programming language extension that simplifies creation of efficient applications for CUDA-enabled GPGPUs
- Available for Linux, Windows and Mac OS X

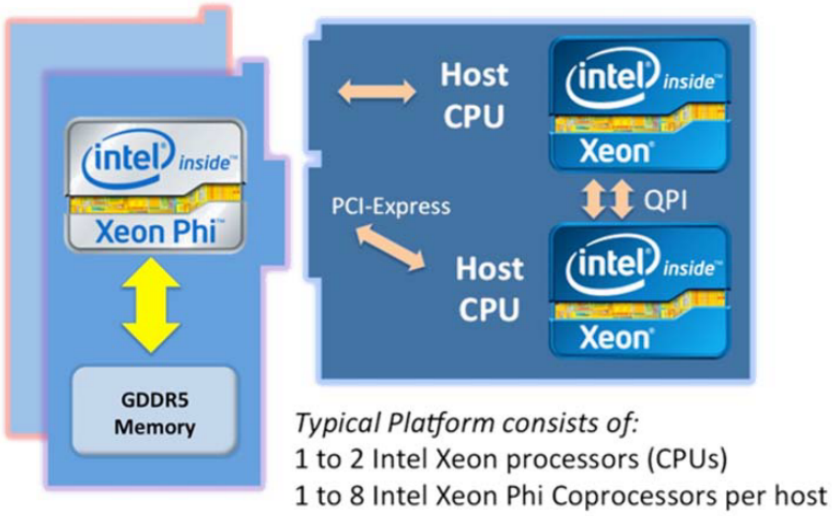

PCI Express 3.0 Host Interface
GigaThread Engine
SMX
SMX
SMX
SMX
SMX
SMX
SMX
SMX
Memory Controller
Memory Controller
Memory Controller
Memory Controller
Memory Controller
Memory Controller
L2 Cache
SMX
SMX
SMX
SMX
SMX
SMX
SMX

TASK Parallelism (MIMD)



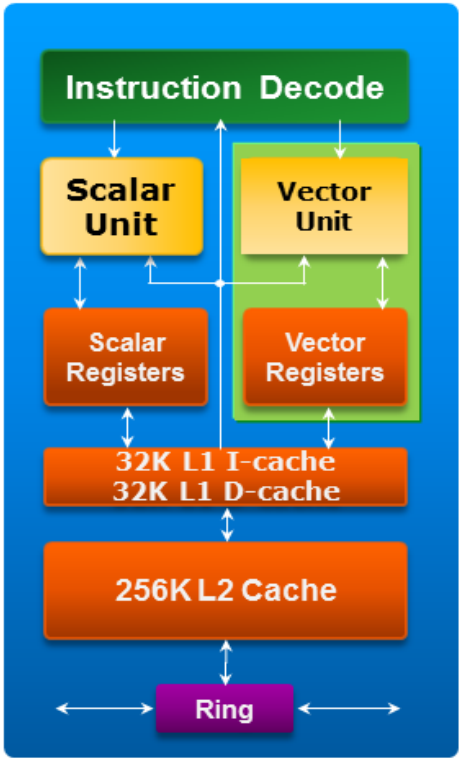
Xeon PHI Architecture





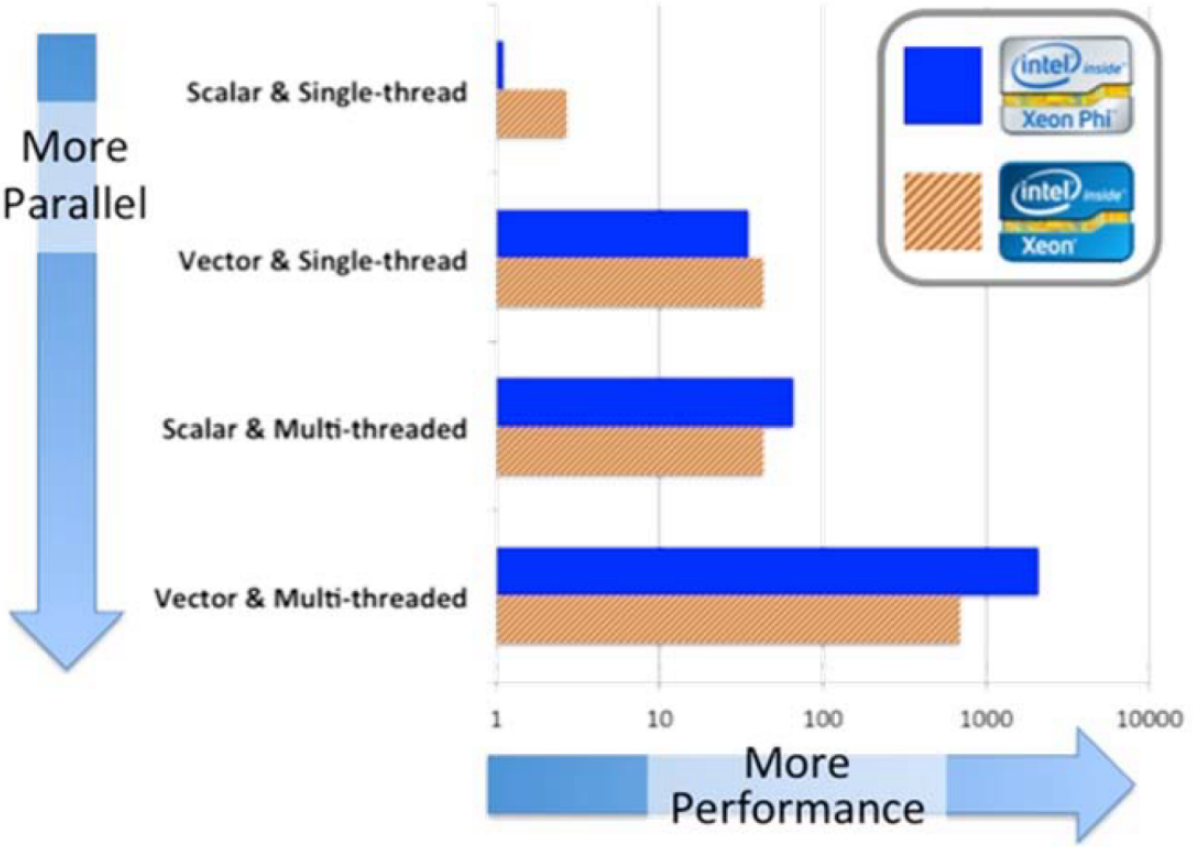
Typical Platform consists of:
1 to 2 Intel Xeon processors (CPUs)
1 to 8 Intel Xeon Phi Coprocessors per host

Core Architecture



- Up to 32 in-order cores
- 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
  - 32 512-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

The Increasing Parallelism



Execution Models: Offload Execution

- Host system offloads part or all of the computation from one or multiple processes or threads running on host
- The application starts execution on the host
- As the computation proceeds it can decide to send data to the coprocessor and let that work on it and the host and the coprocessor may or may not work in parallel.

OpenMP 4.0 TR being proposed and implemented in Intel® Composer XE provides directives to perform offload computations. Composer XE also provides some custom directives to perform offload operations.

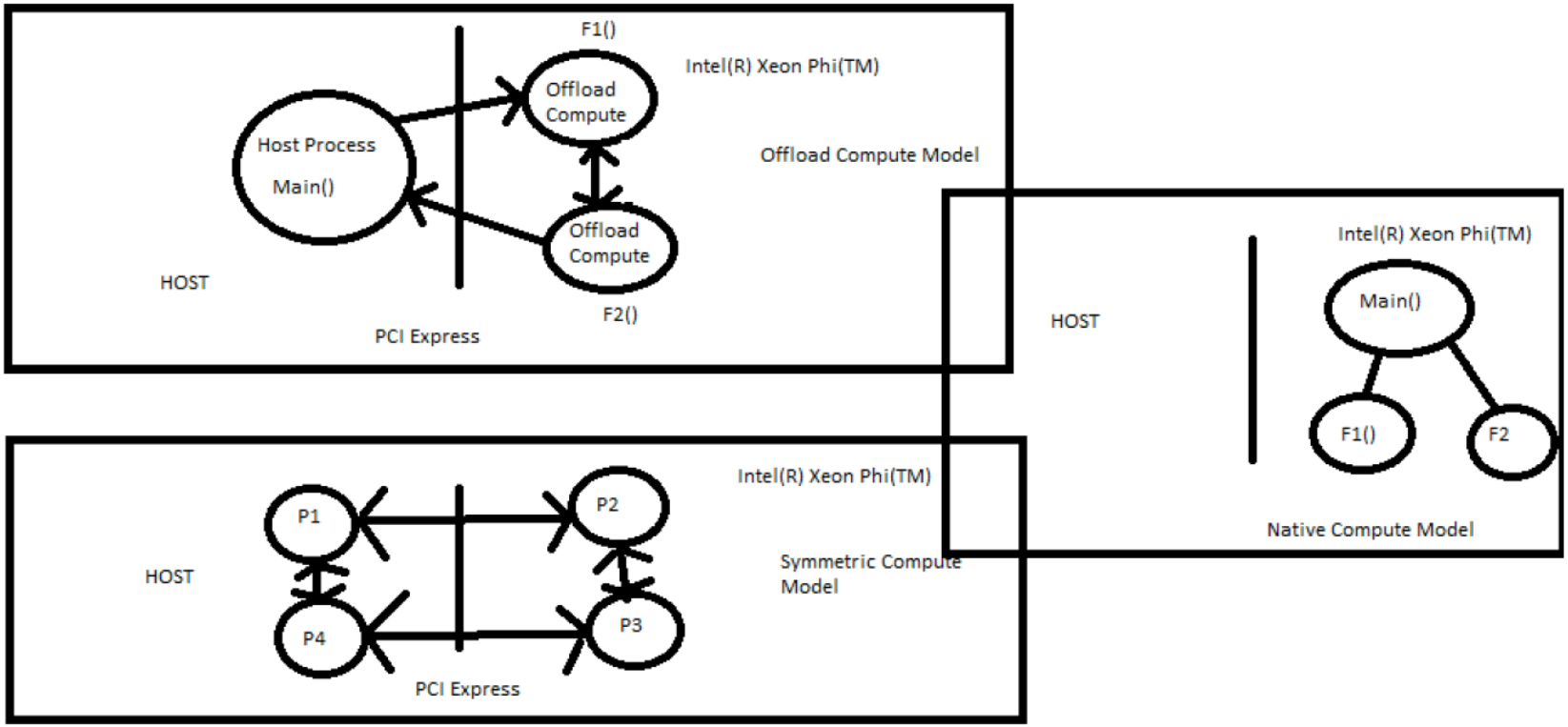
Execution Models: Native Execution

- An Xeon Phi hosts a Linux micro OS in it and can appear as another machine connected to the host like another node in a cluster.
- This execution environment allows the users to view the coprocessor as another compute node.
- In order to run natively, an application has to be cross compiled for Xeon Phi operating environment. Intel® Composer XE provides simple switch to generate cross compiled code.

Execution Models: Symmetric Execution

- The application processes run on both the host and the Phi coprocessor and communicate through some sort of message passing interface like MPI.
- This execution environment treats Xeon Phi card as another node in a cluster in a heterogeneous cluster environment.

Execution Models: Summary



1. Offloading a function call

```
#pragma offload target (mic)
    foo();
```

```
foo() { .... } // Compiled for mic
```

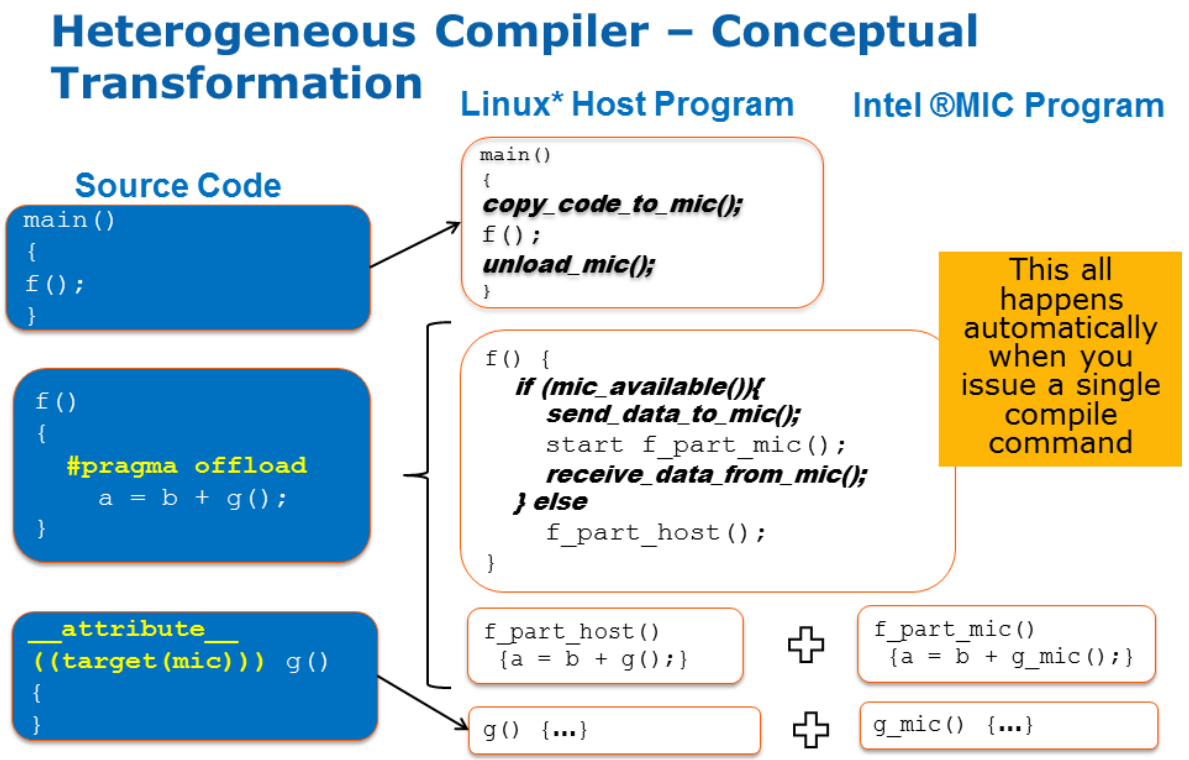
2. Calculating Pi with automatic offload

```
#pragma offload target (mic)
#pragma omp parallel for reduction(+:pi)
 for (i=0; i<count; i++)
 {
       float t = (float)((i+0.5)/count);
       pi += 4.0/(1.0+t*t);
 }
 pi /= count
```

3. Using MKL with offload

```
void your_hook()
{
     float *A, *B, *C; /* Matrices */
     #pragma offload target(mic)
      in(transa, transb, N, alpha, beta) \
      in(A:length(matrix_elements)) \
      in(B:length(matrix_elements)) \
      in(C:length(matrix_elements)) \
     out(C:length(matrix_elements)alloc_if(0))
      sgemm(&transa, &transb, &N, &N,
                &N, &alpha, A, &N, B, &N, &beta, C,
     &N);
}
```

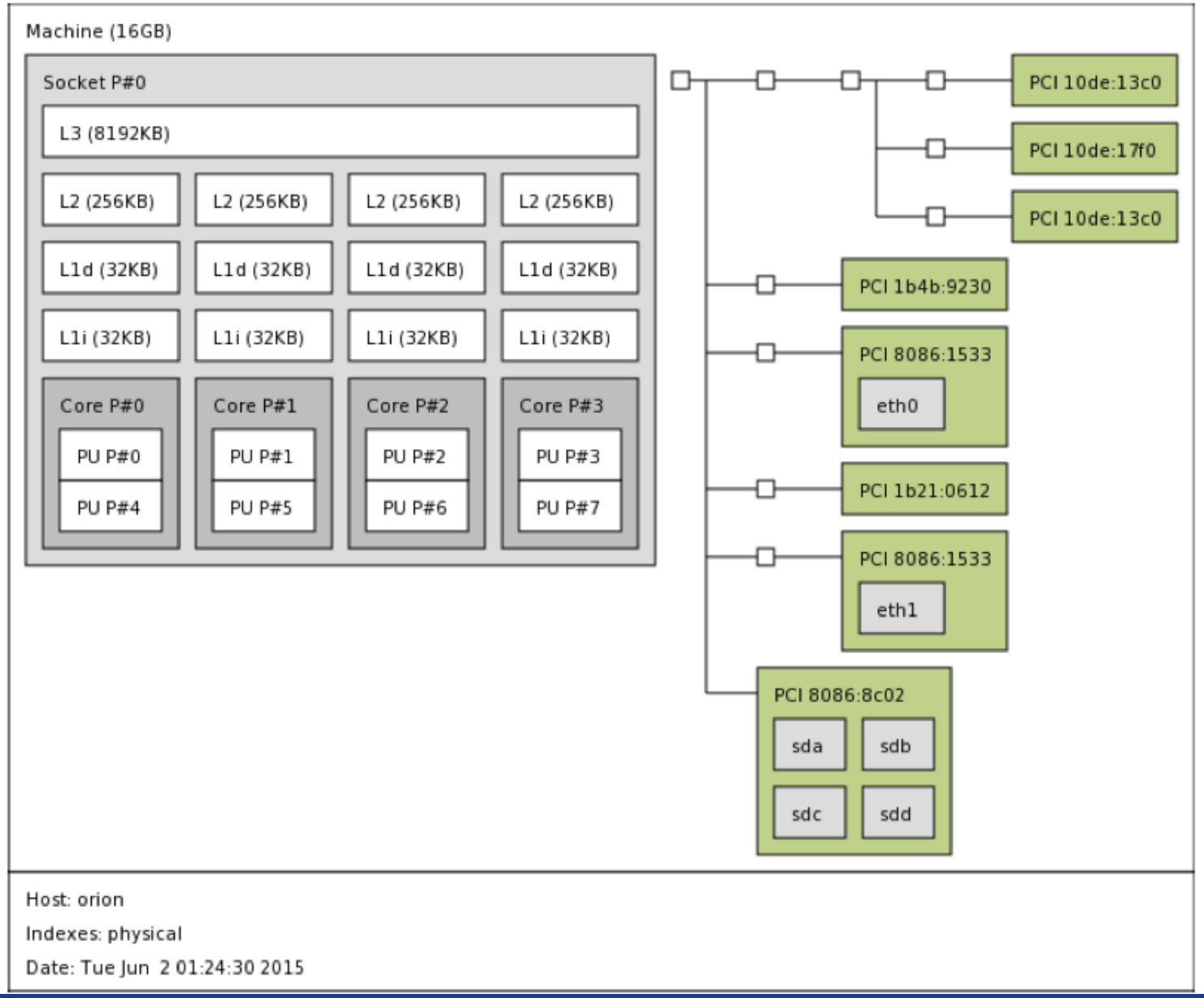
Heterogeneous Compiler



OpenCL

- *O*pen *C*ompute *L*anguage
- Open, royalty-free standard for cross-platform,
- For heterogeneous parallel-computing systems
- Cross-platform. Implementations for
  - ATI GPUs
  - NVIDIA GPUs
  - x86 CPUs

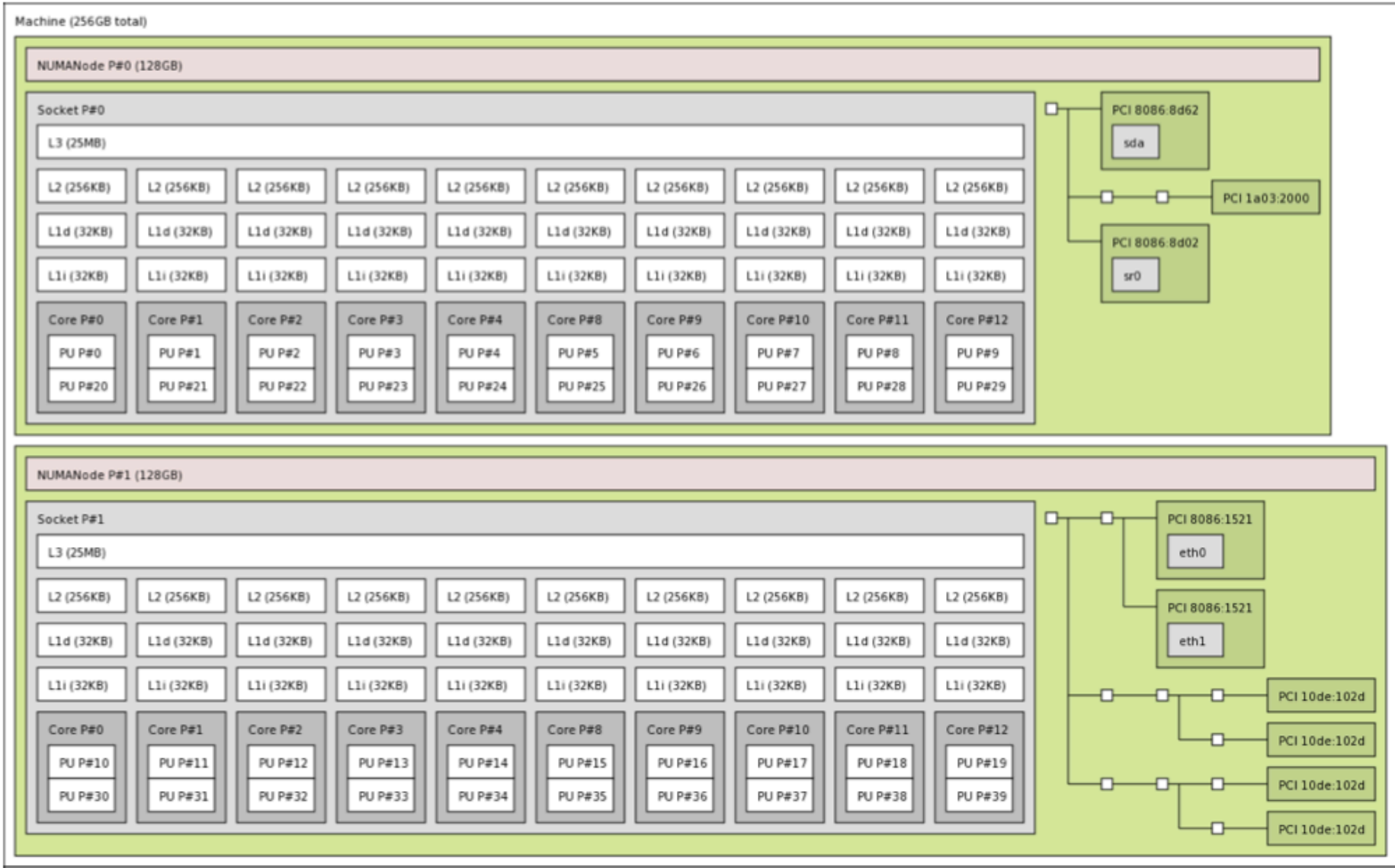
Hwloc ‘lstopo’: Intel Haswell 4771, 3 GPUs, No NUMA



NUMA == Non-Uniform Memory Access

- CPUs directly incorporate memory controllers
- Memory system is not flat, it is instead implemented as a (small) network
- Each CPU socket is associated with its own memory
- Access to memory on a peer CPU socket has much higher latency and lower bandwidth than socket-local memory, must traverse CPU bus topology (QPI or HT)

Hwloc ‘lstopo’: 2x Xeon E5, 2x Tesla K80

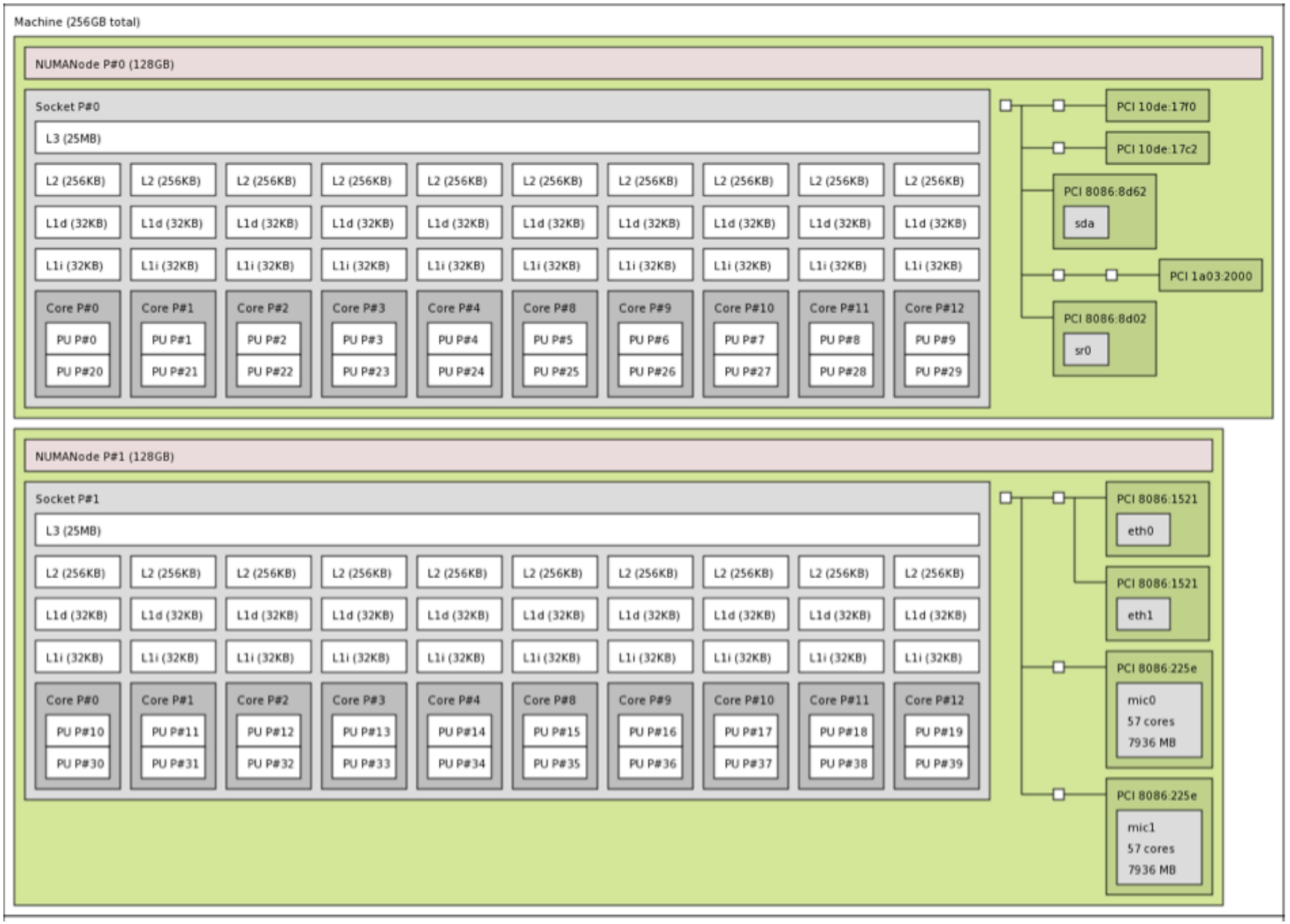


Host: albuquerque.ks.uiuc.edu

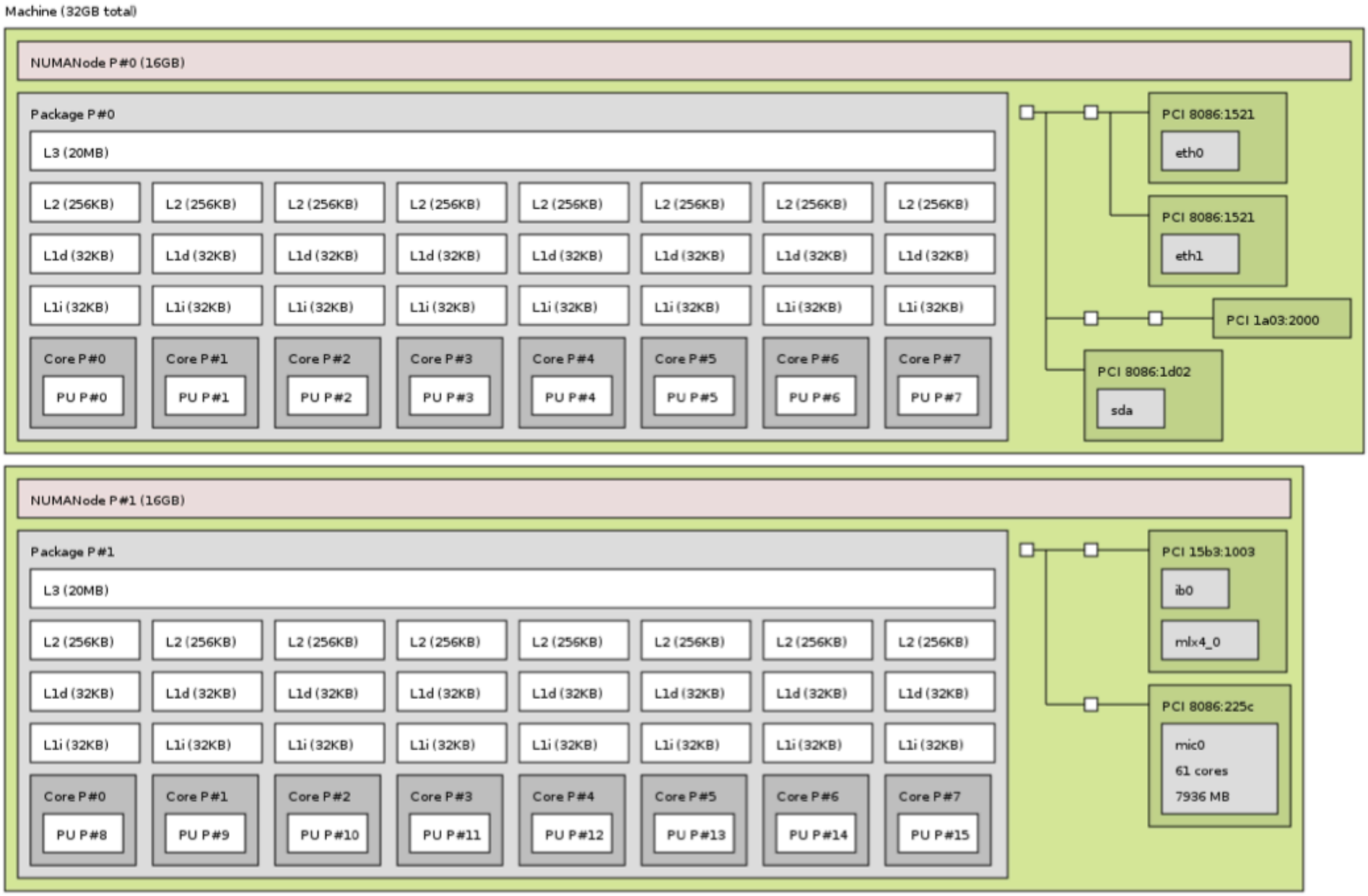
Indexes: physical

Date: Tue 02 Jun 2015 01:07:13 AM CDT

Hwloc ‘lstopo’: 2x Xeon E5, 2x Xeon Phi, 2 GPUs

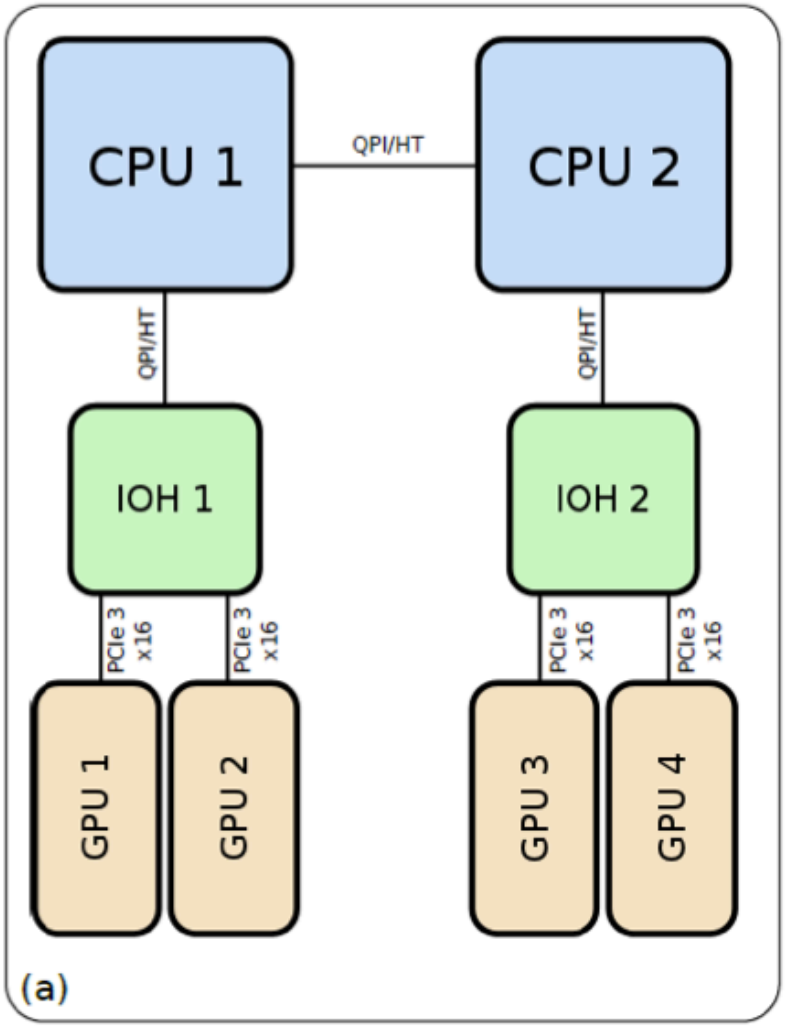


Hwloc ‘lstopo’: 2x Xeon E5 + IB + Xeon Phi



Multi-GPU NUMA Architectures:

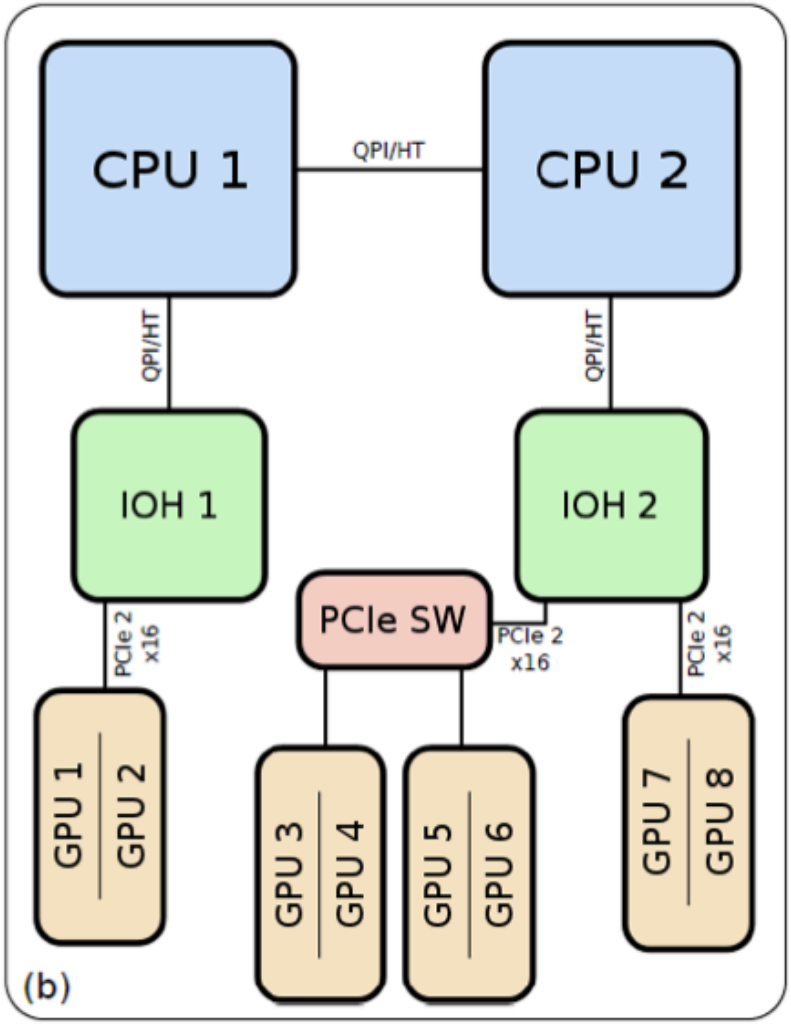
- Example of a "balanced" PCIe topology
- NUMA: Host threads should be pinned to the CPU that is "closest" to their target GPU
- GPUs on the same PCIe I/O Hub (IOH) can use CUDA peer-to-peer transfer APIs
- Intel: GPUs on different IOHs can't use peer-to-peer



Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations
Michael J. Hallock, John E. Stone, Elijah Roberts, Corey Fry, and Zaida Luthey-Schulten.
Journal of Parallel Computing, 40:86-99, 2014.
http://dx.doi.org/10.1016/j.parco.2014.03.009

Multi-GPU NUMA Architectures:

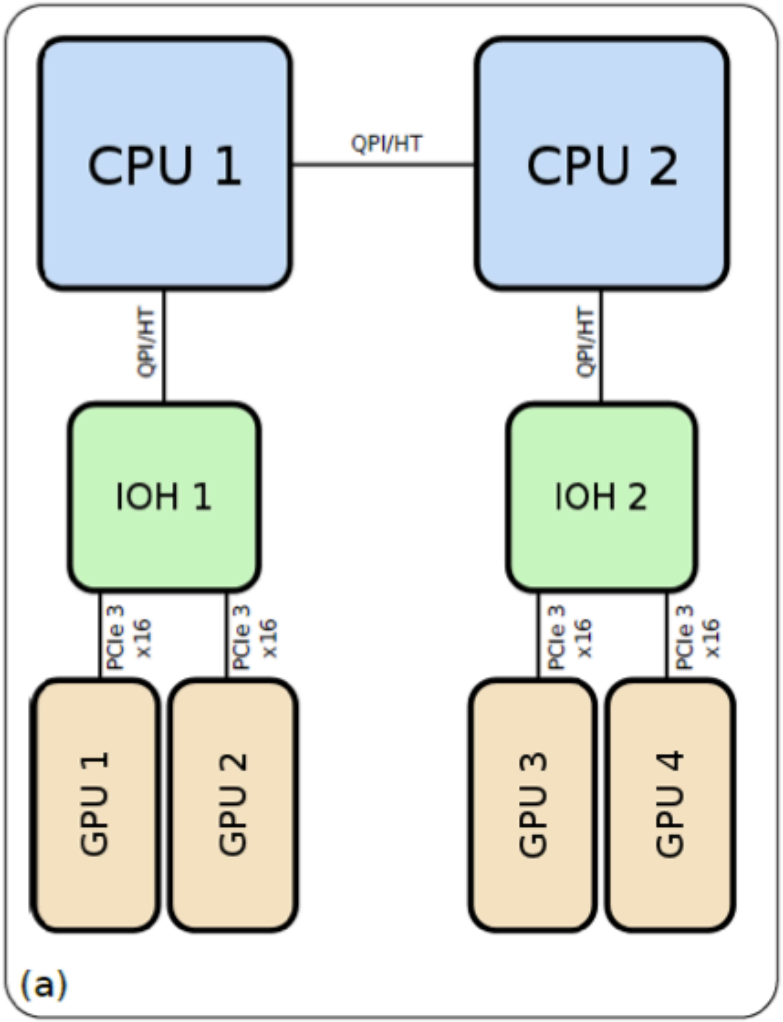
- Example of a very "unbalanced" PCIe topology
- CPU 2 will overhelm its QP/HT link with host-GPU DMAs
- Poor scalability as compared to a balanced PCIe topology



Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations
Michael J. Hallock, John E. Stone, Elijah Roberts, Corey Fry, and Zaida Luthey-Schulten.
Journal of Parallel Computing, 40:86-99, 2014.
http://dx.doi.org/10.1016/j.parco.2014.03.009

Multi-GPU NUMA Architectures:

- GPU-to-GPU peer DMA operations are much more performant than other approaches, particularly for moderate sized transfers
- Likely to perform even better in future multi-GPU cards with direct GPU links, e.g. announced "NVLink"



Simulation of reaction diffusion processes over biologically relevant size and time scales using multi-GPU workstations
Michael J. Hallock, John E. Stone, Elijah Roberts, Corey Fry, and Zaida Luthey-Schulten.
Journal of Parallel Computing, 40:86-99, 2014.
http://dx.doi.org/10.1016/j.parco.2014.03.009