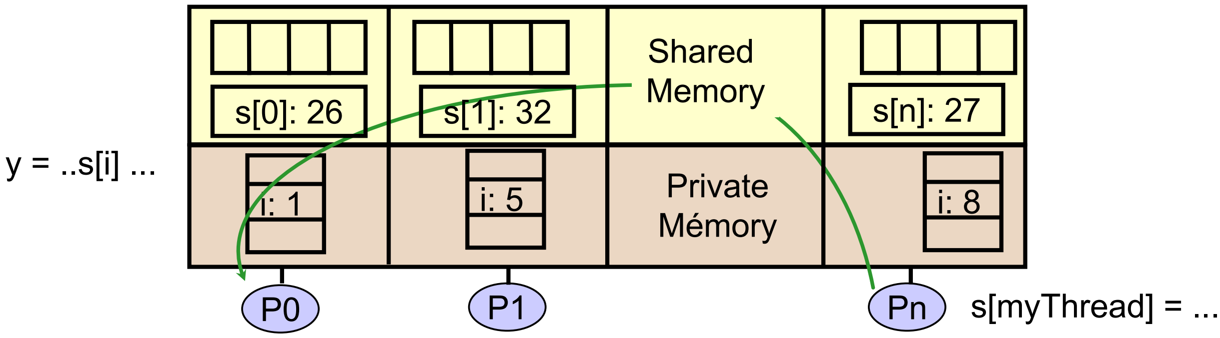
Global address space

The program consists of a collection of named threads

- Generally set at program startup
- Local and shared data as in the shared memory model
- But the shared data is partitioned between local processors (more expensive remote access costs)
- **Examples:** UPC, Titanium, Co-Array Fortran

• Intermediate between shared memory and message passing



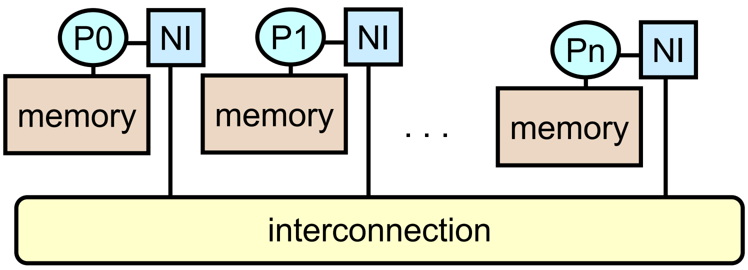
Global address space, contd.

Examples

- Cray T3D, T3E, X1 and HP Alphaserver clusters
- Clusters built with Quadrics, Myrinet, or Infiniband networks

The network interface supports RDMA (Remote Direct Memory Access)

- NIs can directly access the memory without interrupting the CPU
- A processor can read / write to memory with one-sided (put / get) operations,
- Not just a load / store on a shared memory machine
  - Continue computing until memory operation completes
- The "remote" data is usually not cached locally



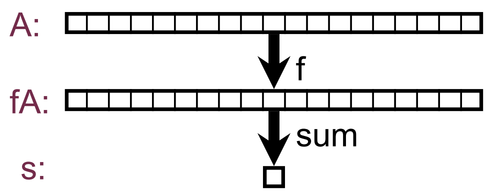
Data-parallel programming models

Data-parallel programming model

- Implicit communications in parallel operators
- Easy to understand and model
- Implicit coordination (instructions executed synchronously)
- Close to Matlab for array operations

- **Drawbacks**
  - Does not work for all models
  - Difficult to port on coarse-grained architectures

A = data array
fA = f(A)
s = sum(fA)



Vector machines

Based on a single processor

- Several functional units
- All performing the same operation

- Exceeded by MPP machines in the 1990s

Come-back since the last ten years

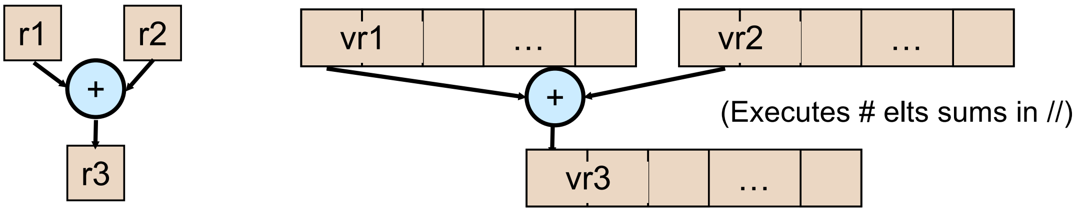
- On a large scale (Earth Simulator (NEC SX6), Cray X1)
- On a smaller scale, processor SIMD extensions
  - SSE, SSE2: Intel Pentium / IA64
  - Altivec (IBM / Motorola / Apple: PowerPC)
  - VIS (Sun: Sparc)
- On a larger scale in GPUs

Key idea: the compiler finds parallelism!

Vector processors

Vector instructions execute on an element vector

- Specified as operations on vector registers

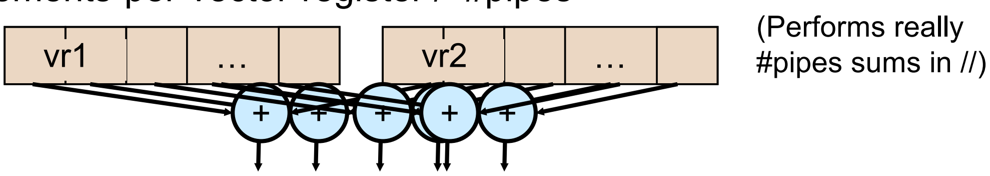


A register contains ~ 32-64 elements

- The number of elements is greater than the number of parallel units (vector pipes/lanes, 2-4)

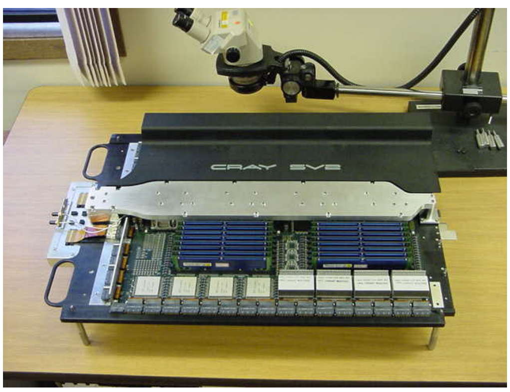
The speed for a vector operation is

#elements-per-vector-register / #pipes



Cray X1: Parallel Vector Architecture

- **Cray combines several technologies in the X1**
  - 12.1 Gflop / s Vector Processors
  - Shared Caches
  - Nodes with 4 processors sharing up to 64 GB of memory
  - Single System Image for 4096 processors
  - Put / get operations between nodes (faster than MPI)



Hybrid machines

Multicore / SMPs nodes used as LEGO elements to build machines with a network

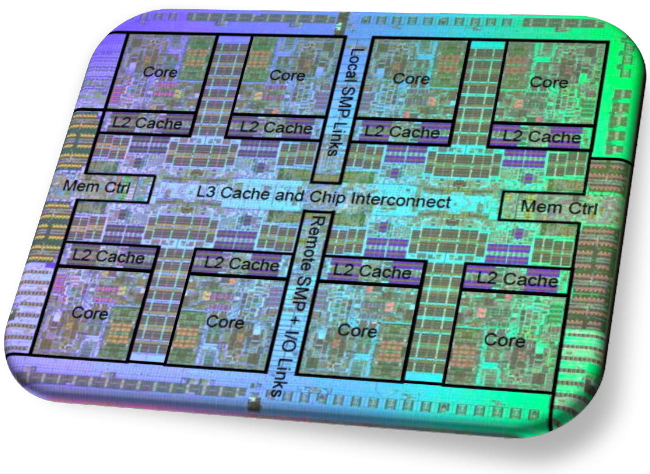
Called CLUMPs (*Cluster of SMPs*)

Examples

- Millennium, IBM SPs, NERSC Franklin, Hopper

- Programming Model
  - Program the machine as if it was on a level with MPI (even if there is SMP)
  - Shared memory within an SMP and passing a message outside of an SMP

- Graphic (co) -processors can also be used



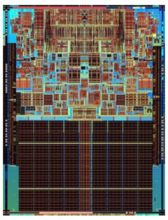
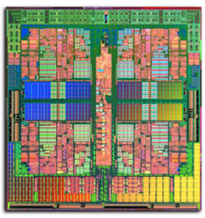
MULTICORES/GPU

Multicore architectures

- A processor composed of at least 2 central processing units on a single chip
- Allows to increase the **computing power** without increasing the **clock speed**
- And therefore **reduce heat dissipation**
- And to **increase the density**: the cores are on the same support, the connectors connecting the processor to the motherboard does not change compared to a single core

Why multicore processors?

Some numbers			
	Single Core Engraving generation 1	**Dual Core** Engraving generation 2	**Quad Core** Engraving generation 3
Core area	A	~ A/2	~ A/4
Core power	W	~ W/2	~ W/4
Chip power	W + O	W + O’	W + O”
Core performance	P	0.9P	0.8P
Chip performance	P	1.8 P	3.2 P

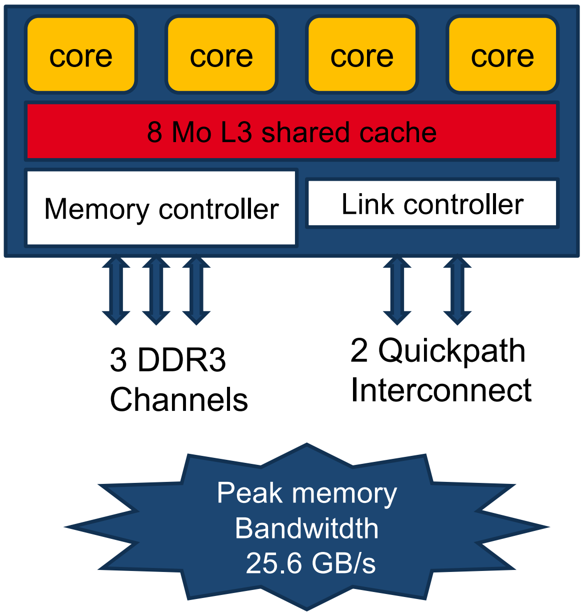



Nehalem-EP architecture (Intel)

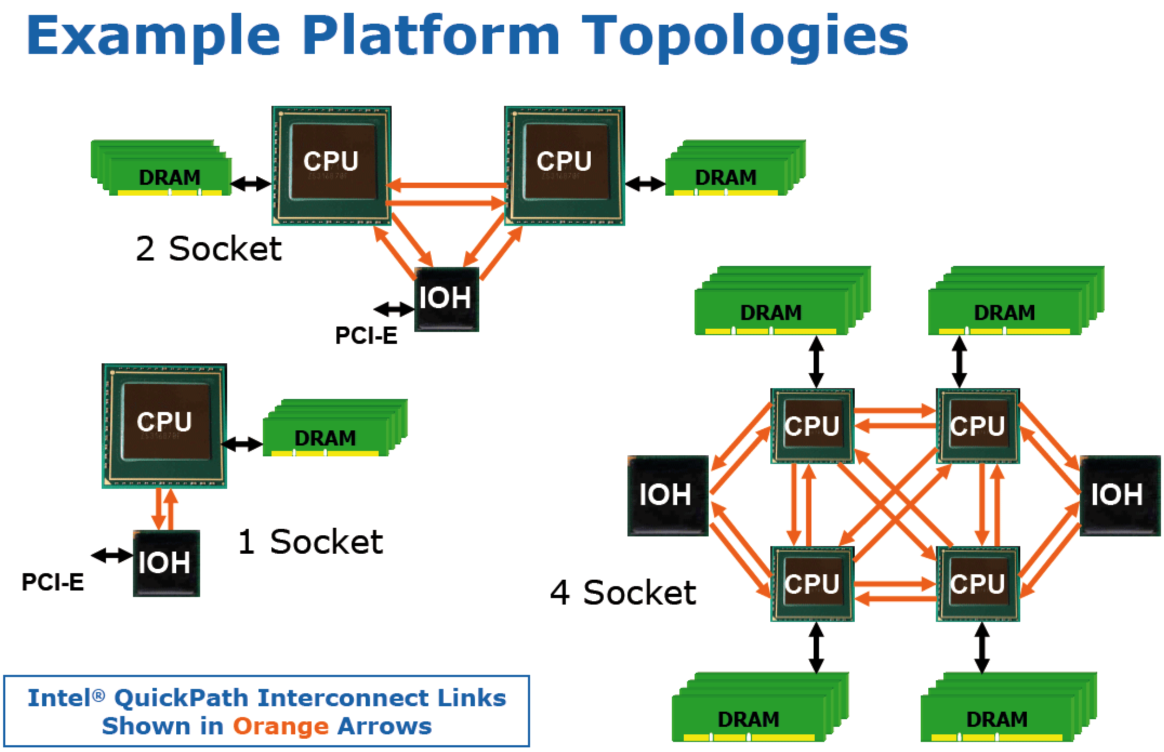
4 cores
On-chip L3 cache shared (8 Mo)
3 cache levels
- Cache L1 : 32k I-cache + 32k D-cache
- Cache L2 : 256 k per core
- Inclusive cache: on-chip cache coherency (SMT)

732 M transistors, 1 single die (263 mm^2)
QuickPathInterconnect
- Point-to-point
- 2 links per CPU socket
- 1 for the connection to the other socket
- 1 for the connection to the chipset

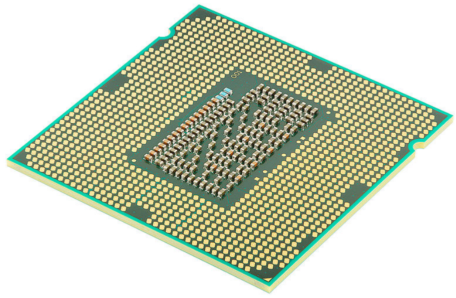
Integrated QuickPath Memory controller (DDR3)



Nehalem



Sandy Bridge-EP architecture



Early 2012 with

- 8 cores per processor
- 3 cache levels

  L1 cache: 32k I-cache + 32k D-cache

  L2 cache: 256 k / core, 8 voies associative

  L3 cache: shared and inclusive (16 Mo on-chip)
- 4 DDR3 memory controller
- AVX instructions → 8 flop DP/cycle (twice of the Nehalem)
- 32 lines PCI-e 3.0
- QuickPathInterconnect

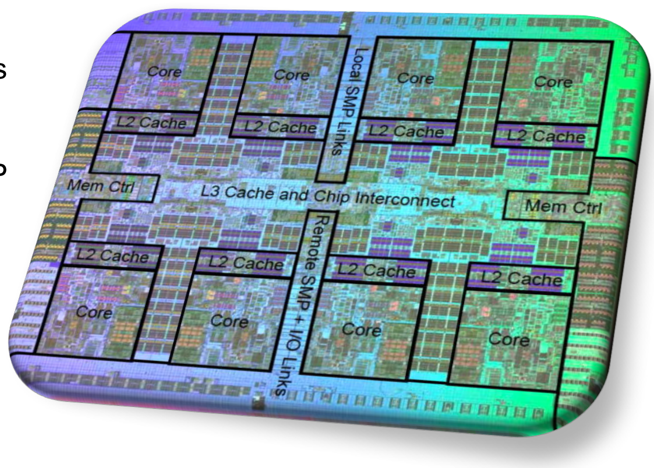
  2 QPI per proc

Power7 Architecture

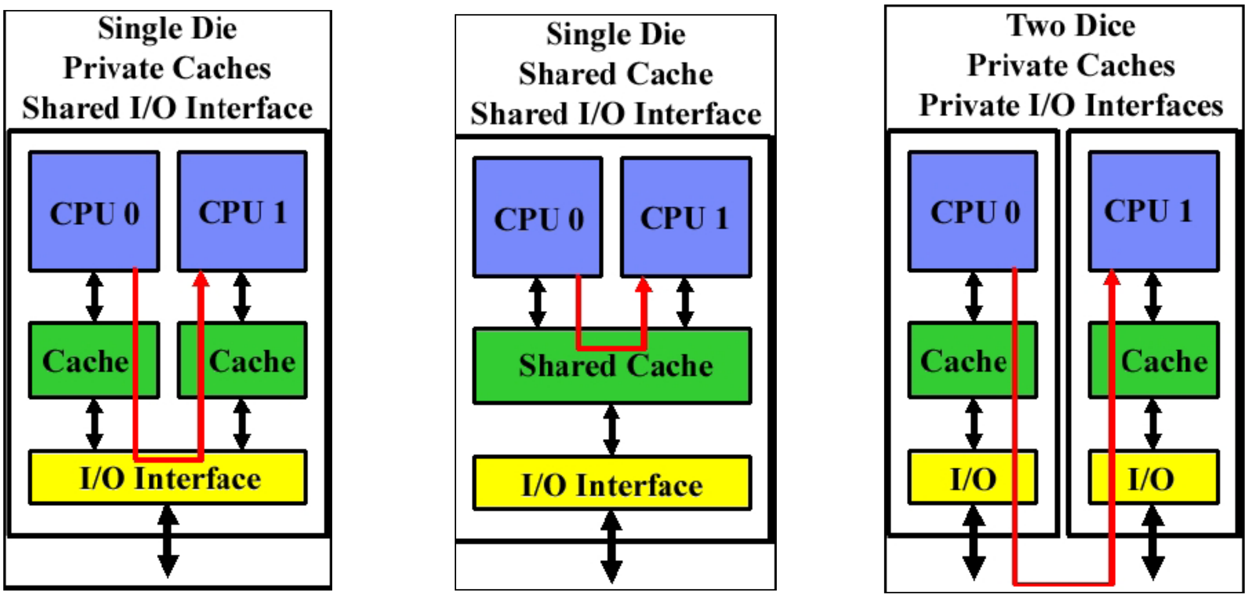
- Cache controller L3 and memory on-chip
- Up to 100 Go/s of memory bandwidth

- 1200 M transistors, 567 mm^2 per die
- up to 8 cores
- 4 way SMT ⇨ up to 32 simultaneous threads
- 12 execution units, including 4 FP
- Scalability: up to 32 8-cores sockets per SMP system , ↗ 360 Go/s of chip bandwidth

  ⇨ Up to 1024 threads /SMP
- 256Ko L2 cache /core
- L3 cache shared using partagé in eDRAM technology (embeddedDRAM)



Caches architectures



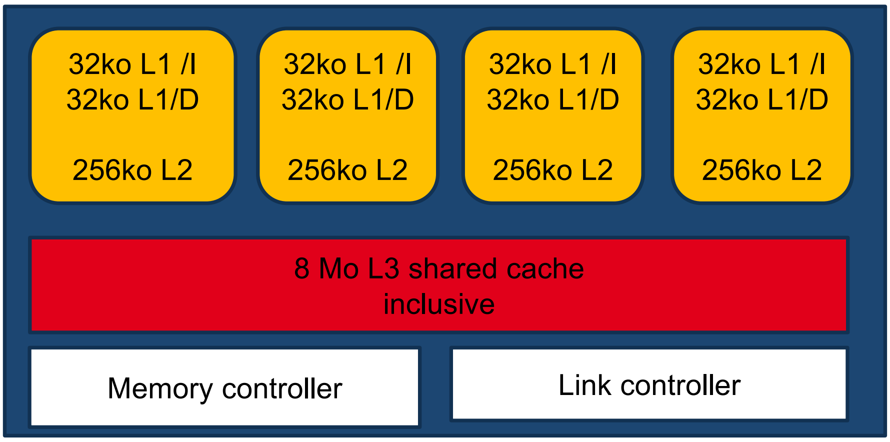
Sharing L2 and L3 caches

- **Sharing the L2 cache (or L3)**
  - ✓ ☺ Faster communication between cores,
  - ✓ ☺ better use of space,
  - ✓ ☺ thread migration easier between cores,
  - ✓ ☹ contention at the bandwidth level and the caches (space sharing),
  - ✓ ☹ coherency problem.

- **No cache sharing**
  - ✓ ☺ no contention,
  - ✓ ☹ communication/migration more costly, going through main memory.

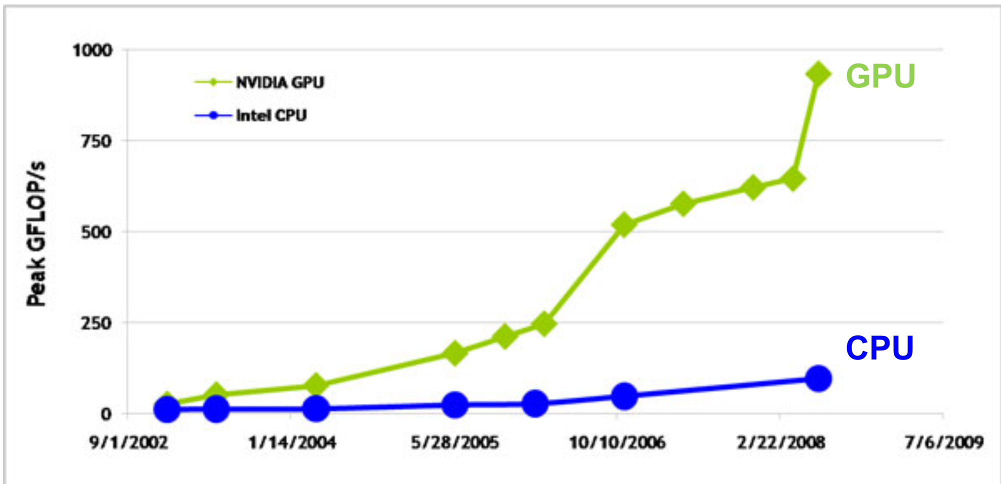
  - Private L2, shared L3 cache: IBM Power5+ / Power6, Intel Nehalem
  - All private: Montecito

Nehalem example: A 3 level cache hierarchy



- L3 cache inclusive of all other levels
  - 4 bits allow to identify in which processor's cache the data is stored
  - ✓ ☺ traffic limitation between cores
  - ✓ ☹ Waste of one part of the cache memory

Performance evolution: CPU vs GPU

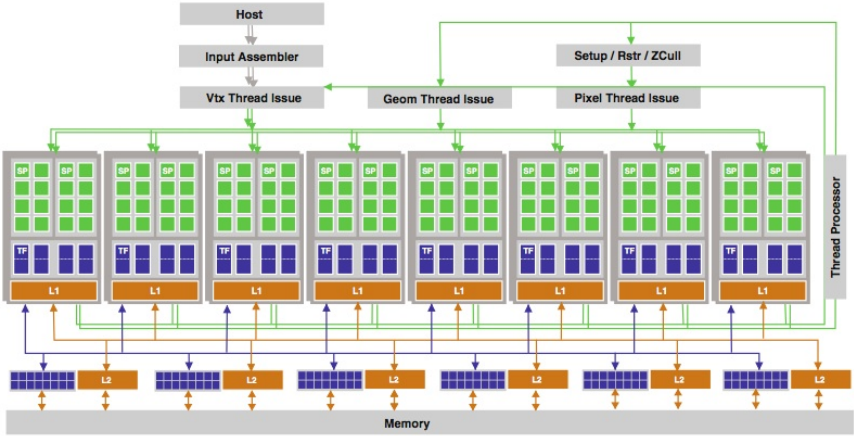


"classical" processors' speed increase *** 2 every 16 months**

GPU processors' speed increase ***2 every 8 months**

GPU

- Theoretical performance GeForce 8800GTX vs Intel Core 2 Duo 3.0 GHz: 367 Gflops / 32 GFlops
- Memory bandwidth: 86.4 GB/s / 8.4 GB/s
- Available in every workstations/laptops: mass market
- Adapted to massive parallelism (thousands of threads per application)
- 10 years ago, only programmed using graphic APIs
- Now many programming models available
    - CUDA , OpenCL, HMPP, OpenACC



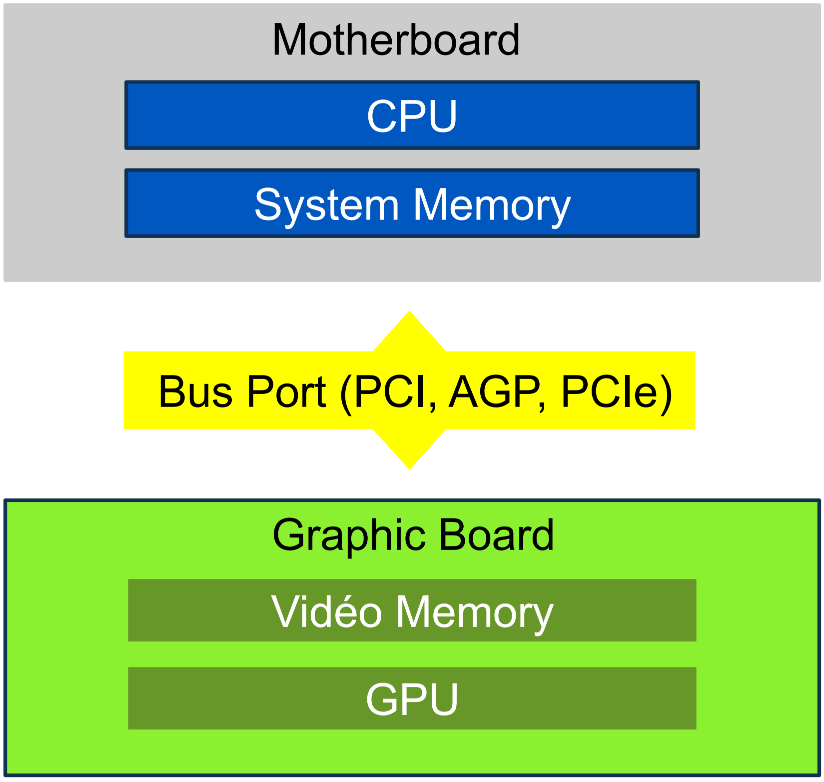
Fermi graphic processor

Major evolutions for HPC

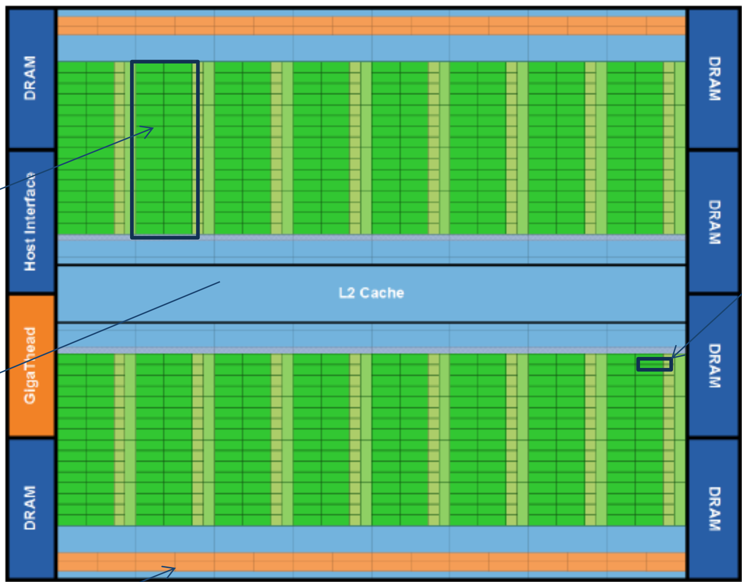
- Floating point operations: IEEE 754-2008 SP & DP
- ECC support (Error Correction Coding) on every memory

- 256 FMAs DP/cycle
- 512 cores
- L1 et L2 cache memory hierarchy
- 64 KB of L1 shared memory (on-chip)
- Up to 1 TB of GPU memory

Classical PC architecture



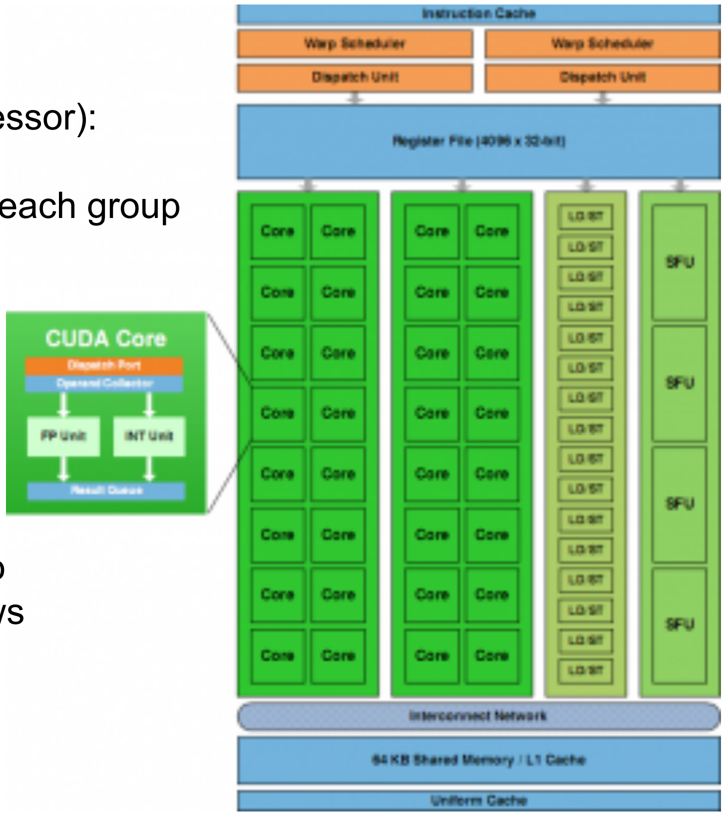
NVIDIA Fermi processor architecture



512 Compute units
16 (32) partitions

NVIDIA Fermi processor architecture

Fermi *SM* (Streaming Multiprocessor):
Each *SM* has 32 cores
A *SM* schedules the threads for each group of 32 threads //



An important evolution
64 Ko of on-chip memory (48 ko shared mem + 16ko L1). It allows threads of a same block to cooperate.
64 bit units

GPU /CPU Comparaison

With equal performance, platforms based on GPUs

- Occupy less space
- Are cheaper
- Consume less energy

But

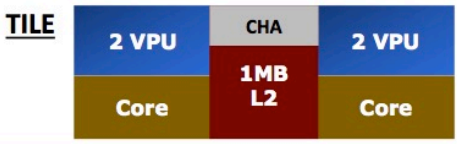
- Are reserved for massively parallel applications
- Require to learn new tools
- What is the guarantee of the durability of the codes and therefore of the investment in terms of application port?

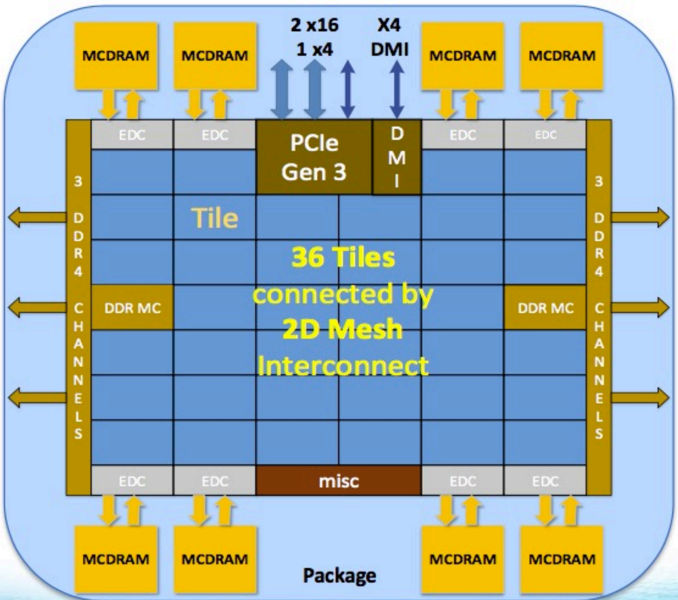
Intel's Many Integrated Core processors: A response to the GPU?

- Manycores processors, ≥ 50 cores on the same chip
- X86 Compatibility
  - Intel software support
- Xeon Phi in June 2012
  - 60 cores/1.053 GHz/240 threads
  - 8 GB memory and 320 GB/s of bandwidth
  - 1 teraflops !

Knights Landing Intel Xeon Phi

Knights Landing Overview





Avinash Sodani CGO PPoPP HPCA Keynote 2016

Chip: up to **36 Tiles** interconnected by **2D Mesh**
Tile: 2 Cores + 2 VPU/core + 1 MB L2

Memory: MCDRAM: up to 16 GB on-package; High BW
DDR4: 6 channels @ 2400 up to 384GB
IO: 36 lanes PCIe Gen3. 4 lanes of DMI for chipset
Node: 1-Socket
Fabric: Intel® Omni-Path Fabric on-package (not illustrated)

Vector Peak Perf: 3+TF DP and 6+TF SP Flops
Scalar Perf: ~3x over Knights Corner
Streams Triad (GB/s): MCDRAM : 450+; DDR: ~90

Note: not all specifications shown apply to all Knights Landing SKUs
Source Intel: All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice. KNL data are preliminary based on current expectations and are subject to change without notice. 1Binary Compatible with Intel Xeon processors using Haswell Instruction Set (except TSX). 2Bandwidth numbers are based on STREAM-like memory access pattern when MCDRAM used as flat memory. Results have been estimated based on internal Intel analysis and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

Kalray MPPA-256 overview

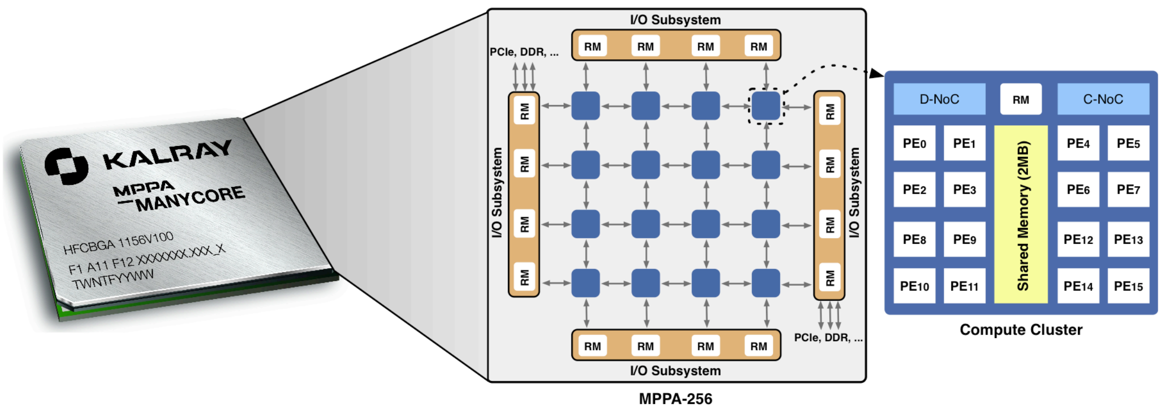
Kalray

- French semiconductor and software company developing and selling a new generation of manycore processors for HPC

MPPA-256

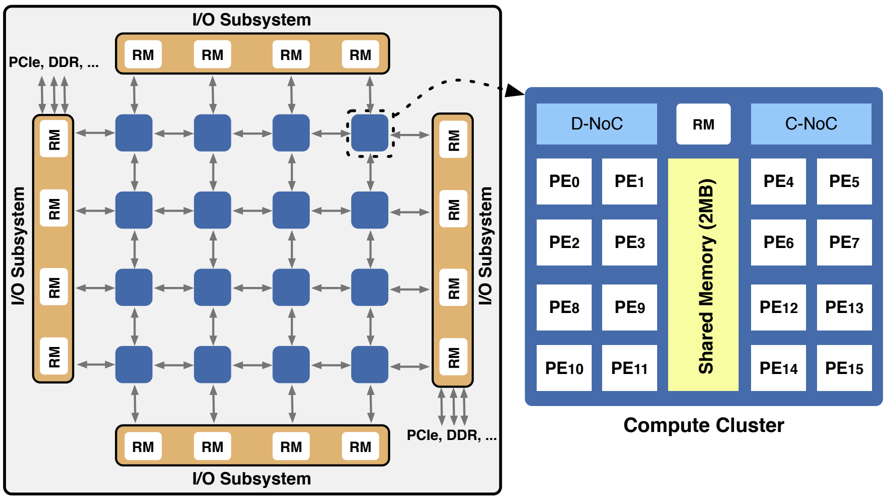
- Multi-Purpose Processor Array (MPPA)
- Manycore processor: 256 cores in a single chip
- Low power consumption (5W - 11W)

Kalray MPPA-256 overview



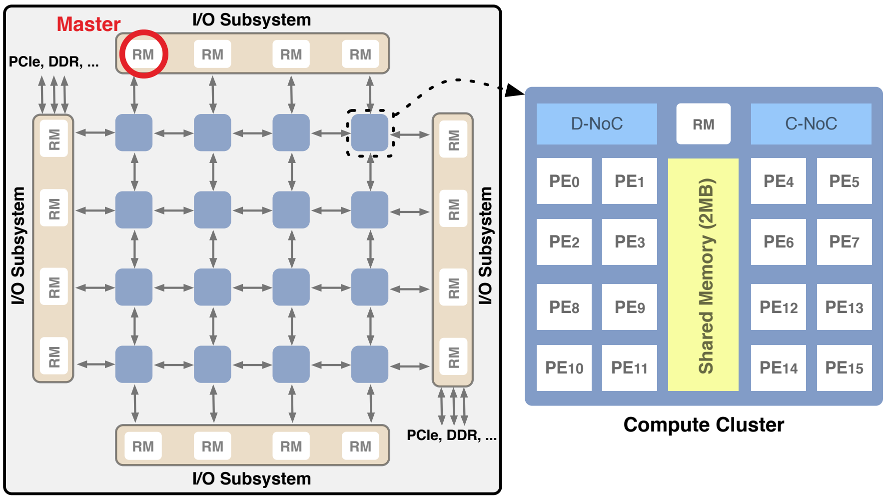
256 cores (PEs) @ 400 MHz: 16 clusters, 16 PEs per cluster
PEs **share 2 MB of memory**
Absence of cache coherence protocol inside the cluster
Network-on-Chip (NoC): communication between clusters
4 I/O subsystems: 2 connected to external memory

Kalray MPPA-256 overview



MPPA-256

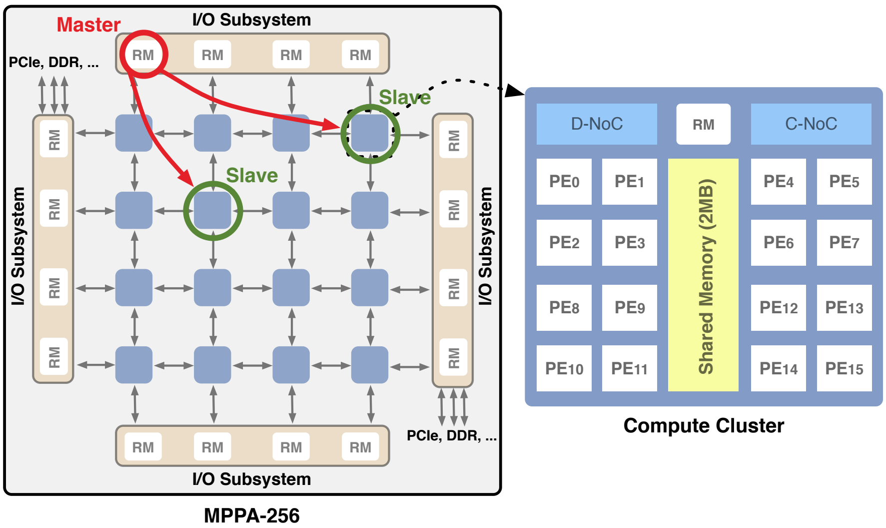
Kalray MPPA-256 overview



MPPA-256

A **master** process runs on an **RM** of one of the **I/O subsystems**

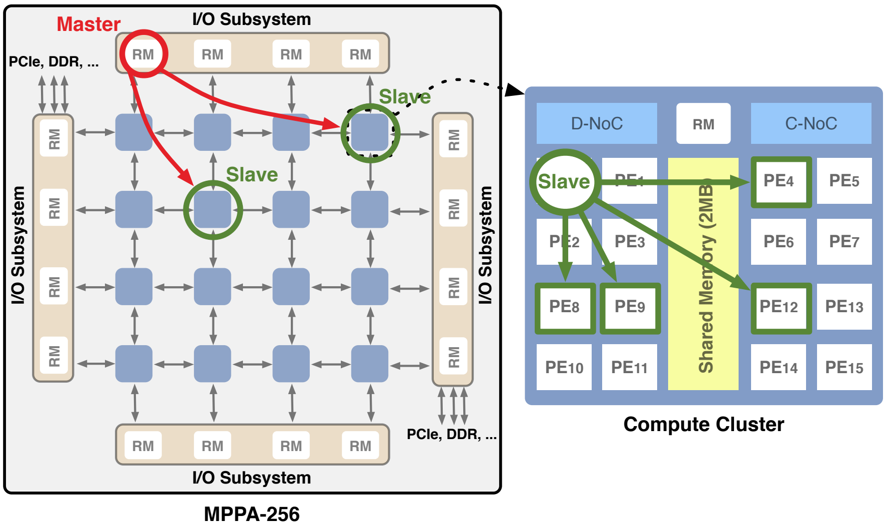
Kalray MPPA-256 overview



The **master** process spawns **worker processes**

One worker process per cluster

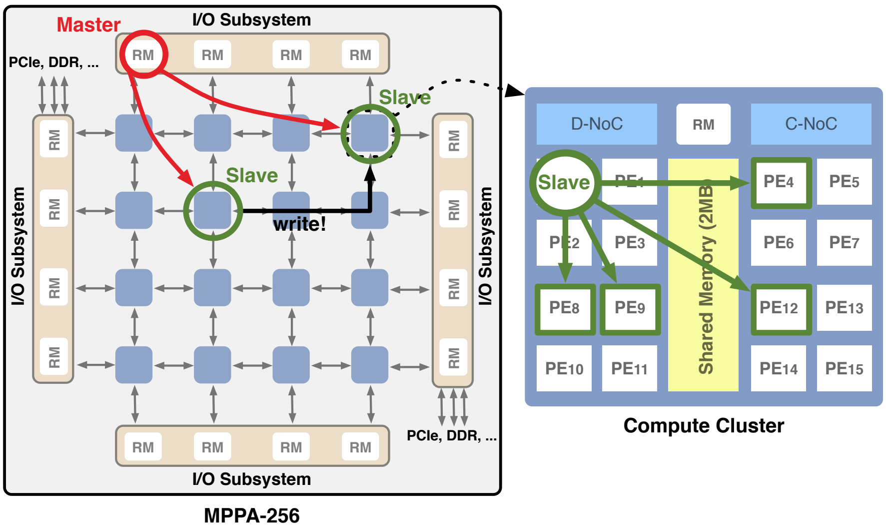
Kalray MPPA-256 overview



The **worker process** runs on the **PE0** and may create up to **15 threads**, one for each PE

Threads share 2 MB of memory

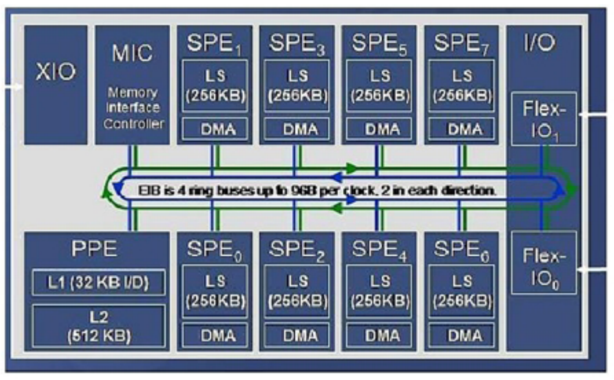
Kalray MPPA-256 overview



Communications take the form of **remote writes**

Data travel through the **NoC**

Specialized processor: CELL



- Developed by Sony, Toshiba and IBM: PlayStation 3 processor
- A processor is composed of a **main core** (PPE) and 8 **specific cores** (SPE)
- The PPE: classic PowerPC processor, without optimization, "in order", it affects the tasks to the SPEs
- SPEs: consisting of a local memory (LS) and a vector computation unit (SPU). Very fast access to their LS but to access the main memory they must perform an asynchronous transfer request to an interconnect bus. The SPEs perform the computational tasks.
- The optimization work is the **responsibility of the programmer**

CELL parallelism

- SPUs allow to process 4 32 bits operations / cycle (128 b register)
- Explicit programming of independent threads for each core
- Explicit memory sharing: the user must manage the data copy between cores

⇨ Harder to program than GPUs (because for GPUs, threads do not communicate between different multiprocessors, except at the beginning and at the end)

> **CELL processor: peak performance (128b registers, SP)**
> 4 (SP SIMD) x 2 (FMA) x 8 SPUs x 3.2 GHz = 204.8 GFlops/socket (in SP)

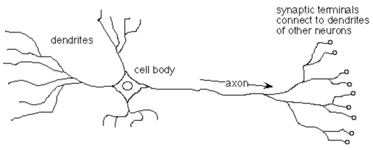
Specialized processors – hybrid programming

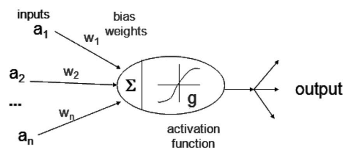
- **FPGA (Field Programmable Gate Array)**
  - ✓ adapted to specific problems
- **CELL**
  - ✓ interesting architecture but difficult to program
- **GPU**
  - ✓ More and more efficient
  - ✓ Better suited to HPC
  - ✓ Tools to program them being developed
  - ✓ Available anywhere, cheap

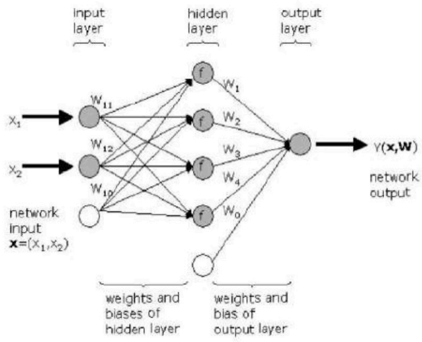
  - But adapted to a massive parallelism
  - PCI-e transfers greatly limit performance
  - The GPU as a co-processor (hybrid architecture) offers new perspectives, introduces new programming models

Tensor Processing Unit (TPU)

- Large number of applications now using neural networks and deep learning
  - Beat human champion at Go
  - Decreasing error in image recognition (from 26 to 3,5%) and speech recognition (by 30%)
- Widely used in Google, Facebook, and Twitter datacenters
- Artificial neural networks made of several layers
  - Parallelism between layers
  - Multiply and add patterns



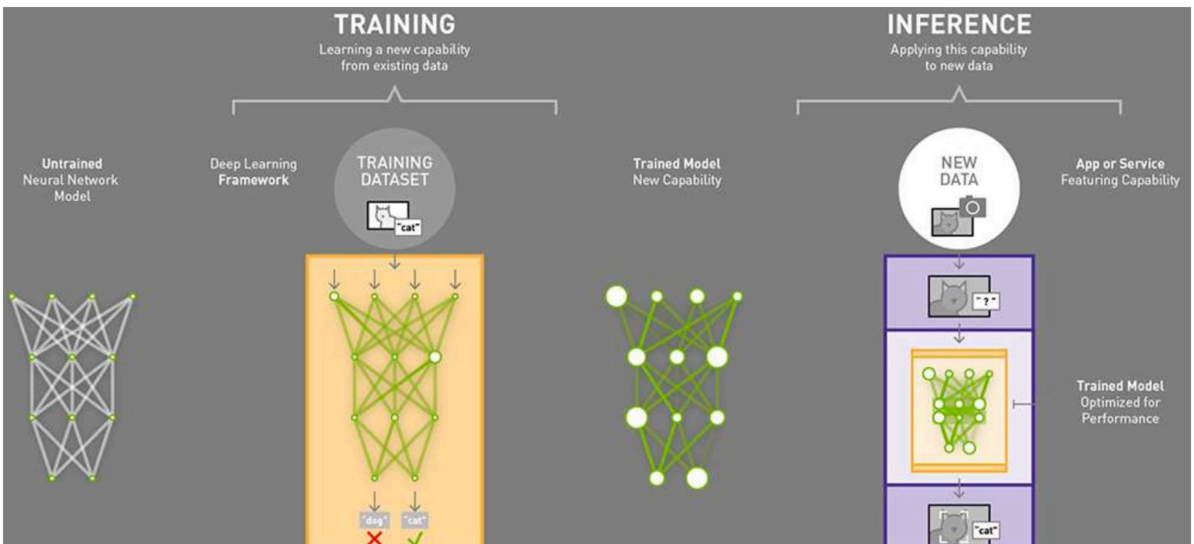




In-Datacenter Performance Analysis of a Tensor Processing Unit, N.P. Jouppi et al., 44th Symposium on Computer Architecture (ISCA), Toronto, Canada, June 26, 2017.

Tensor Processing Unit (TPU), Contd.

- **Two phases**
  - Training (calculation of weights): floating points operations
  - Inference (prediction): addition-multiplications



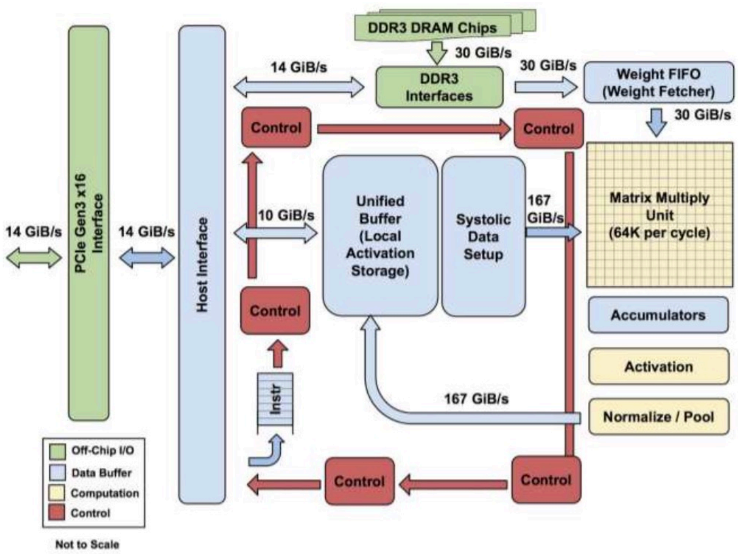
Adapted from
https://blogs.nvidia.com/blog/2016/08/22/difference-deep-learning-training-inference-ai/

Tensor Processing Unit (TPU), Contd.

- Custom ASIC for the inference phase (training done in GPUs)
- **Goals**
  - Improve cost-performance by 10x compared to GPUs
  - Simple design and better response rime guarantees
- **Characteristics**
  - More like a co-processor to reduce time-to-market delays
    - Host sends instructions to TPU
  - Connected through PCIe I/O bus

Tensor Processing Unit (TPU), Contd.

- The **Matrix Multiply Unit (MMU)** is the TPU's heart
  - contains 256 x 256 MACs
- Weight FIFO (4 x 64KB tiles deep) uses 8GB off-chip DRAM to provide weights to the MMU
- Unified Buffer (24 MB) keeps activation input/output of the MMU & host
- Accumulators:
  (4MB = 4096 x 256 x 32bit)
  collect the16 bit MMU products
  - 4096 (1350 ops/per byte to reach peak performance ~= 2048 x2 for double buffering)



Tensor Processing Unit (TPU), Contd.

- MMU uses a Systolic execution
- Using 256x256 MACs that perform 8-bit integer multiply & add (enough for results)
- Holds 64KB tile of weights + 1 more tile (hide 256 cycles that need to shift one tile in)
  - less SRAM accesses
  - lower power consumption
  - higher performance
- MatrixMultiply(B) A matrix instruction takes a variable-sized B*256 input, multiplies it by a 256x256 constant weight input, and produces a B*256 output, taking B pipelined cycles to complete

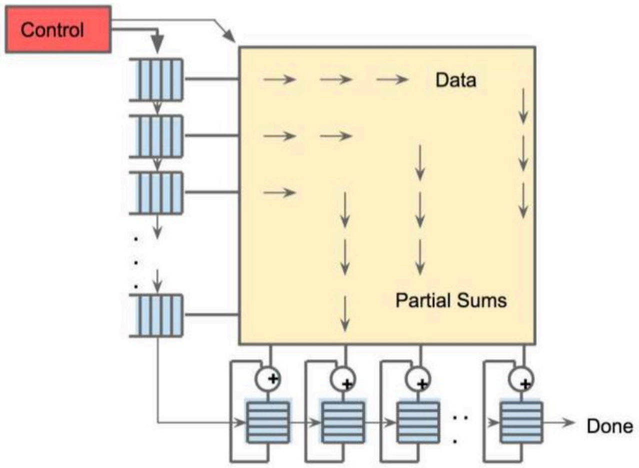
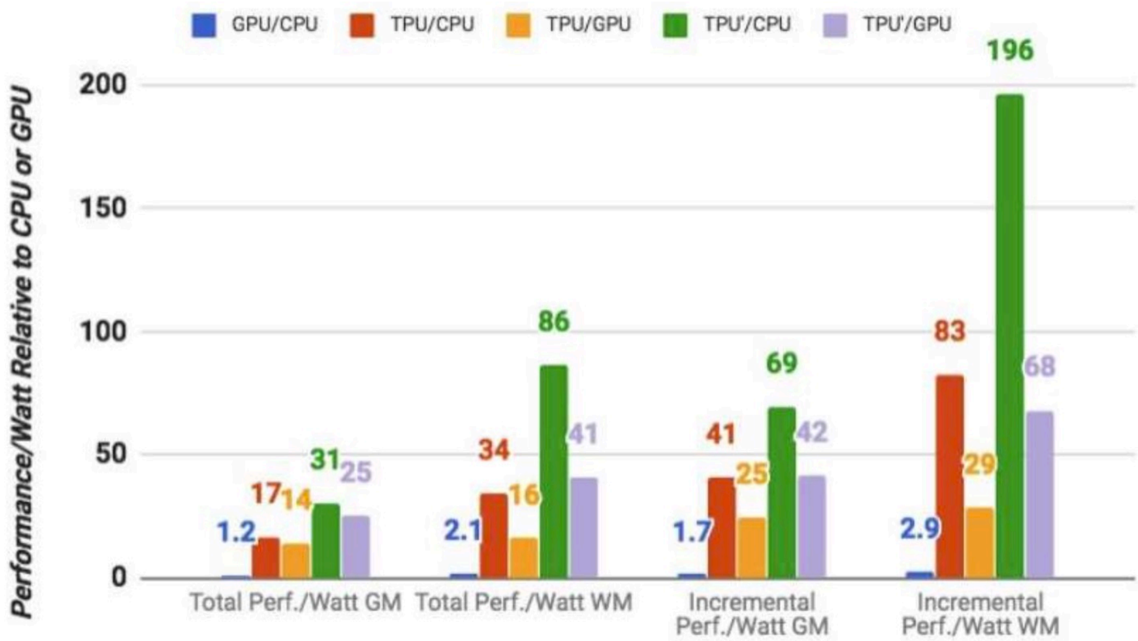


Figure 4. Systolic data flow of the Matrix Multiply Unit. Software has the illusion that each 256B input is read at once, and they instantly update one location of each of 256 accumulator RAMs.

Tensor Processing Unit (TPU), Contd.

- Cost performance results



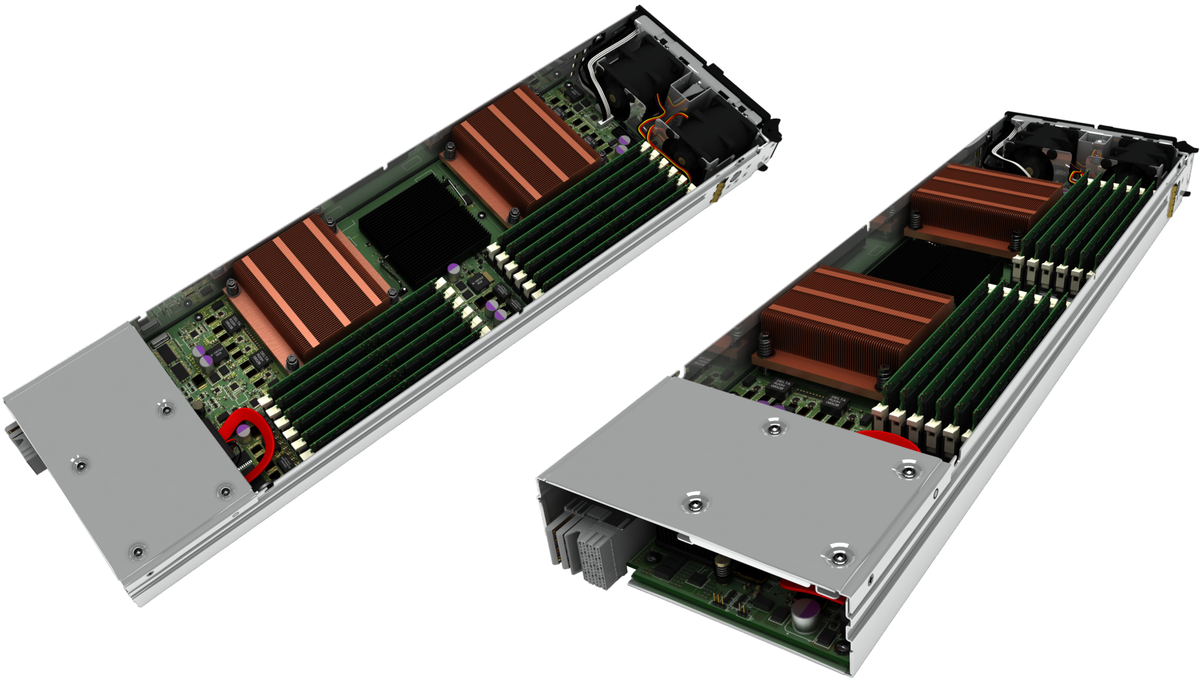
Tensor Processing Unit (TPU), Contd.

- Since Inference apps are user-facing, they emphasize response-time over throughput
- Due to latency limits, the K80 GPU is just a little faster than the CPU, for inference.
- The TPU is about 15X - 30X faster at inference than the K80 GPU and the Haswell CPU
- Four of the six NN apps are memory-bandwidth limited on the TPU; if the TPU were revised to have the same memory system as the K80 GPU, it would be about 30X - 50X faster than the GPU and CPU.
- The performance/Watt of the TPU is 30X - 80X that of contemporary products; the revised TPU with K80 memory would be 70X - 200X better

In-Datacenter Performance Analysis of a Tensor Processing Unit, N.P. Jouppi et al., 44th Symposium on Computer Architecture (ISCA), Toronto, Canada, June 26, 2017.

HOW TO BUILD A PETAFLOP MACHINE?

How to build a petaflop machine?



1 node, 2 sockets, 16 cores

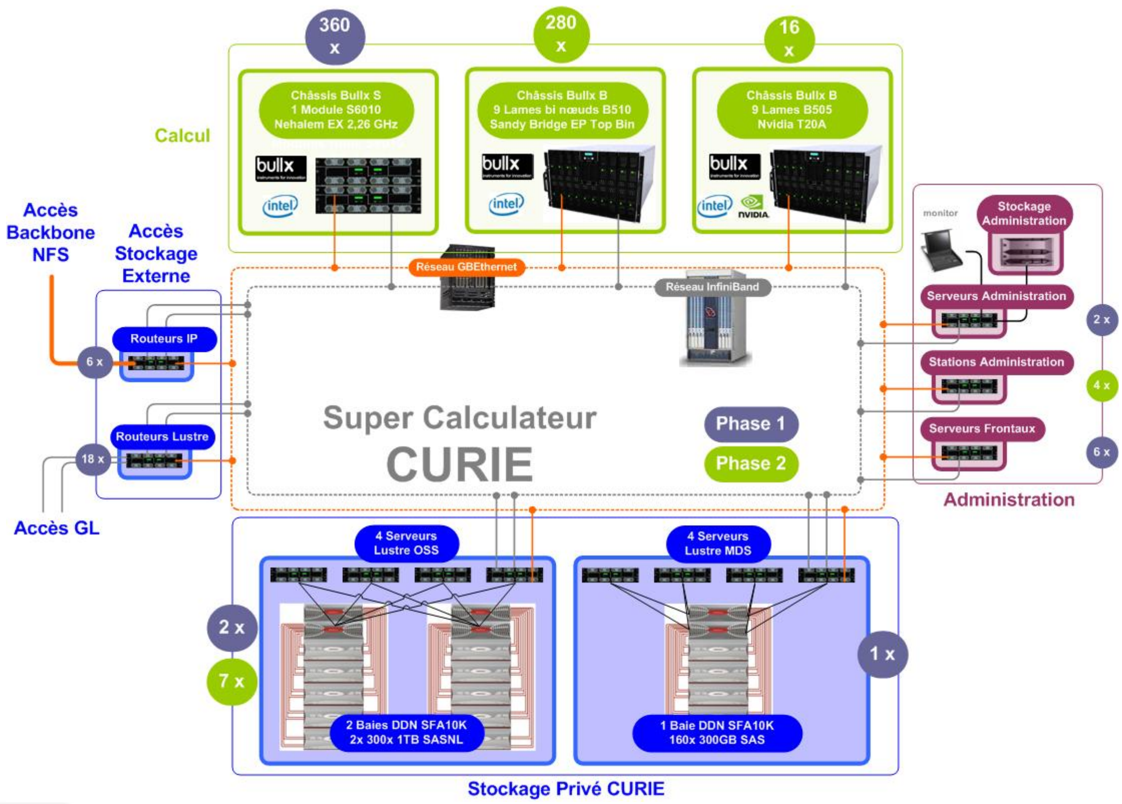
How to build a petaflop machine? Contd.

18 nodes, 36 sockets, 288 cores

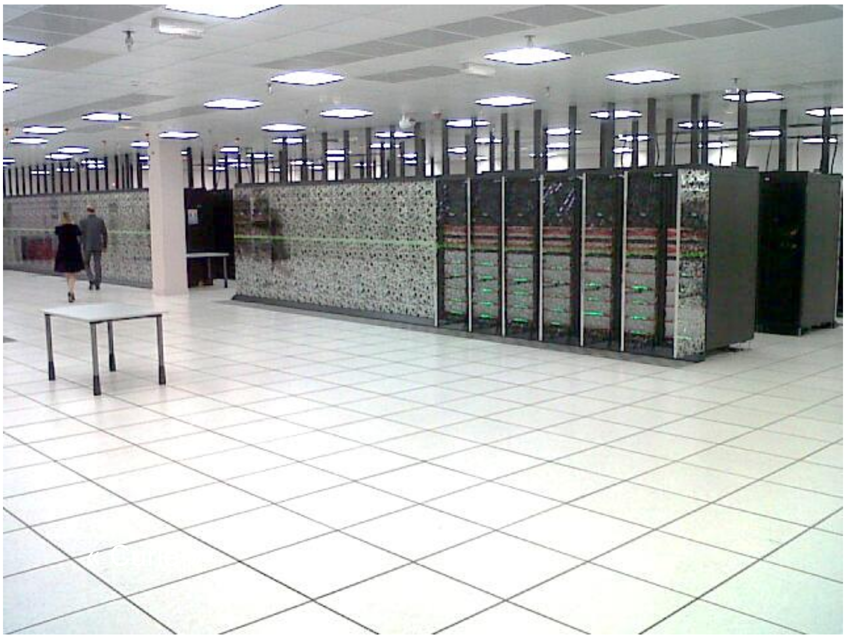
How to build a petaflop machine? Contd.

108 nodes, 216 sockets, 1728 cores

Connecting everything

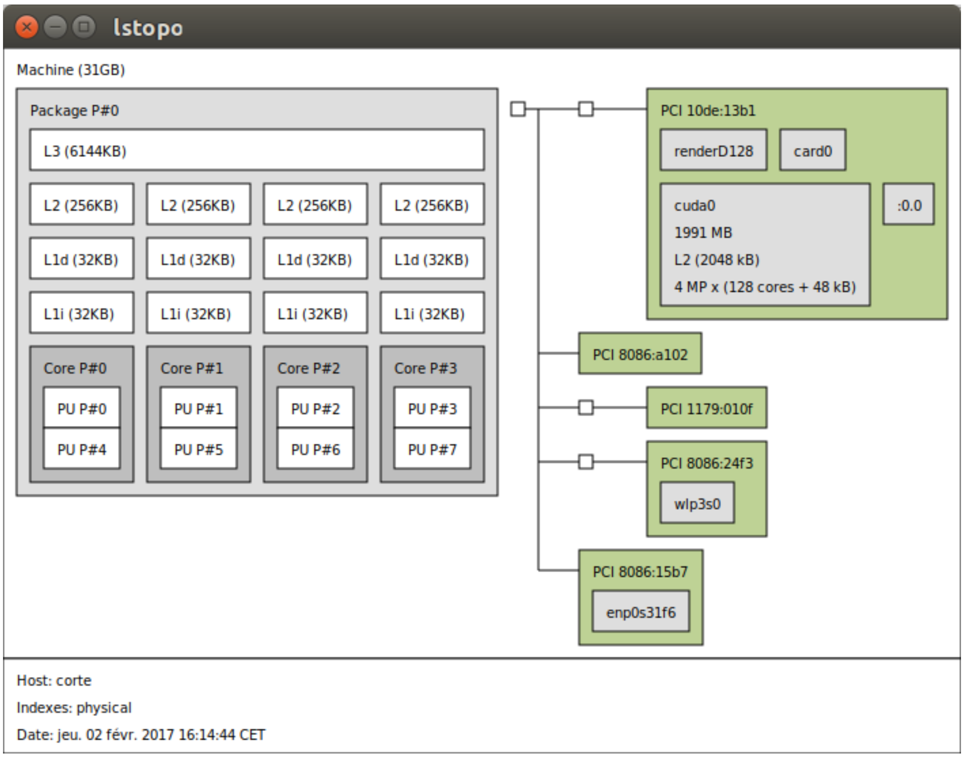


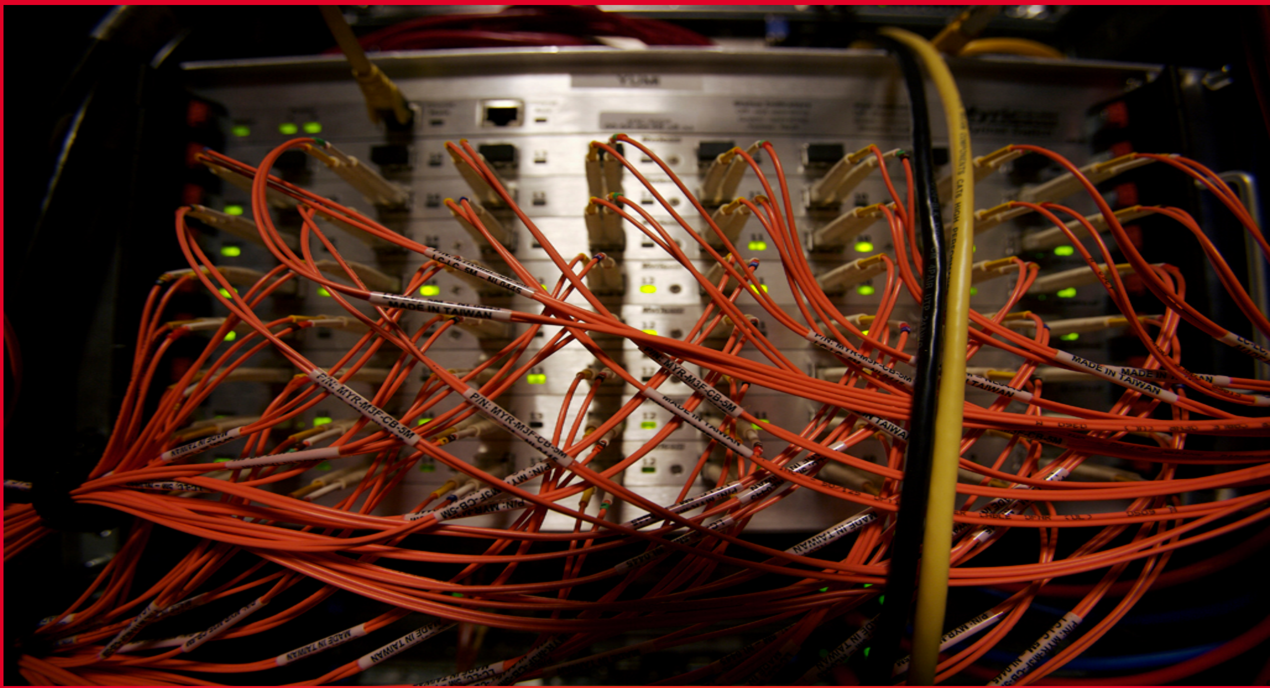
How to build a petaflop machine? Contd.



90 000 cores
360 To memory
10 Po storage
250 Go/s IO
200 m²

Getting information: lstopo




MADE IN TAIWAN