

Parallel Computer Architecture

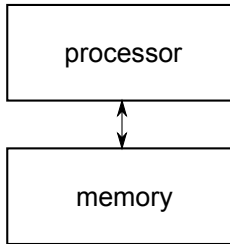
Pangfeng Liu
National Taiwan University

February 20, 2022

Architecture

- Multiprocessor
- Multicomputer
- Flynn's Taxonomy

Uniprocessor



- This is how we think of a computer.

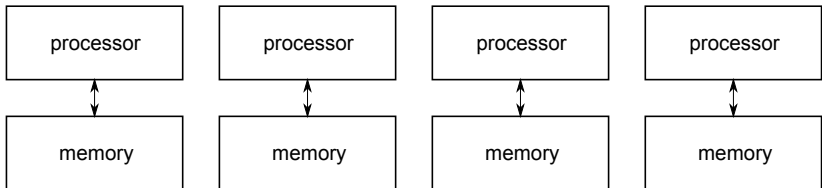
Uniprocessor

- It has a single processor for computation.
- It has a single memory for storing instructions and data.
- The CPU fetches instructions from memory, executes the instructions, updates the contents of registers and possibly data in the memory, and then repeats.
- Intel, AMD, ARM, etc.

Multiple Uniprocessors

- It is intuitive to have multiple uniprocessors working together to have high performance.
- We believe that we can solve the problem faster/better if more processors work together.
- How do they work *together*?

Work Together?

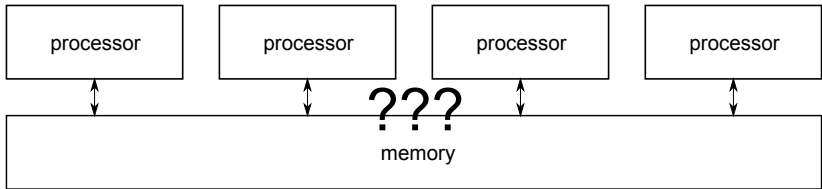


???

Communication

Processors must communicate to coordinate their actions, and exchange data if necessary.

Work Together?



- We cannot simply connect all processors to a memory.
- The memory will be *inconsistent*.

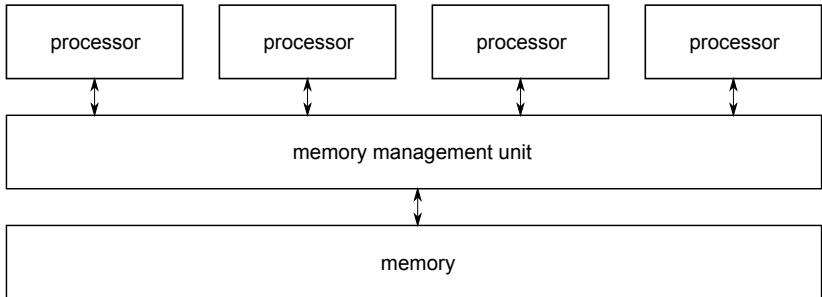
Race Condition

- Suppose two processors want to add 1 to the same counter.
- The first processor fetches the old contents of the counter and adds 1 to it.
- Before the first processor can store the new content, and the second processor fetches the old content.
- The first processor now stores its new content to memory.
- The second processor adds 1 to the old content and stores the new content back to memory.
- The counter only increases by 1, which is *incorrect*.

Multiple Uniprocessors

- We put a *memory management/arbitration unit* between the processors and the memory so that every processor can access memory in a consistent manner.
- This memory management unit must be very efficient in providing *point-to-point* data transfer so as to provide fast memory access for every processor.

Multiprocessor



Discussion

- What could happen if three processors want to add 1, 2, and 4 into a memory with an initialized value 0?

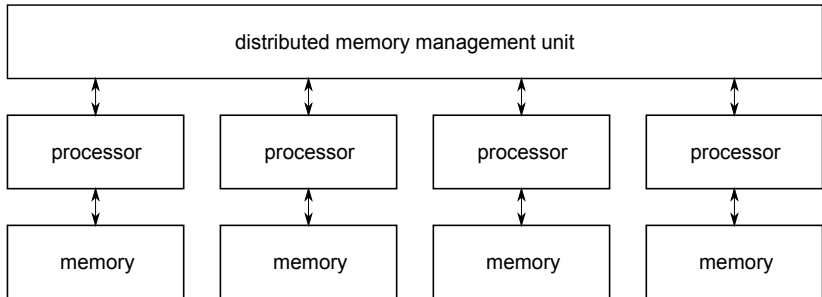
Shared Memory

- Logically we do not care about the memory management unit – we simply believe that every processor can access this *shared memory*.
- This shared memory provides a *shared addressing space* for all processors.
- This *shared* memory is also a *global memory*.
- The cost for every processor to access every part of the memory is the same, and then we have Uniform Memory Access (UMA).

Distributed Shared Memory

- It is a distributed version of shared memory.
- Every processor has local memory, and the collection of the local memory forms a global memory.
- We connect the processors to a memory management unit, which determines the address is local or remote.

Multiprocessor



Distributed Multiprocessor

- If the address is local, then it is retrieved from the local memory.
- If the address is remote, it is retrieved from someone else's local memory.
- The cost for every processor to access every part of the memory is *not* the same, then we have Non-Uniform Memory Access (NUMA).

Discussion

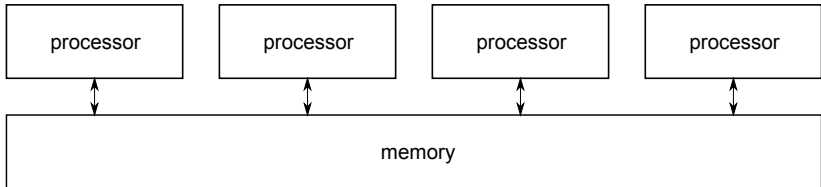
- Describe the difference between NUMA and UMA.
- Recall that each A64FX processor has four NUMA nodes.
What does that mean?

Multiprocessor

- Logically we do not care about the memory management unit – we simply believe that every processor can access this *shared memory*.
- This shared memory provides a *shared addressing space* for all processors.
- This *shared* memory is also a *global memory*.

A Logical View

- In summary, we can think of our multiprocessor is like this.



Multiprocessor

- Conceptually multiple processors are connected by a shared memory – that is all we need to know in terms of programming.
- For some programming models, we still need to consider the possibility of race condition and use the construct provided by the programming model to avoid it, e.g., Pthread.

Functions

- Every processor has its computing resource, like ALU, registers, etc., so you can run multiple processes on them simultaneously.
- Every processor works on its assigned tasks using its computing resource.
- Every processor can read and write the shared memory to communicate or synchronize with other processors.

Synchronization

- Processor can synchronize with each other by shared memory.
- For example, in a barrier synchronization, a processor cannot proceed until all others have reached the same barrier.
- We can set a shared variable count. Every finished processor adds 1 to count. When the value of count reaches the number of processors, then every processor knows that it has synchronized with everyone.

Discussion

- Describe the dependency graph using *functions* and *synchronization*.

Memory Conflict

- Multiple processors can access the memory at the same time, causing conflicts.
- When a computation has different outcome due to different execution order, we have a *race condition*.

Software Solution

- Programming environment must provide mechanism that prevents race condition.
- Critical section, lock, synchronization, etc.
- More specific details will be provided when we discuss parallel programming.

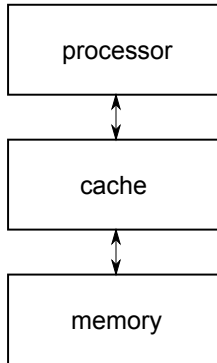
Discussion

- Describe the concept *critical section*.

Cache

- To make the case even worst, we need to deal with the cache.
- Cache is fast memory, usually small and expensive.
- To improve performance, we put often-used data/instructions in a cache. If we need the data/instructions again, we can access them in the fast cache instead of the slow main memory.

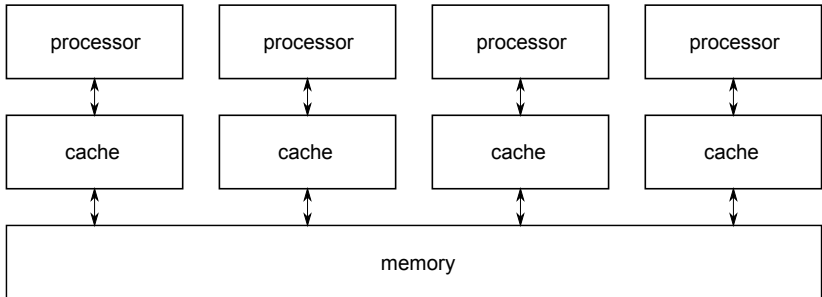
Cache for Uniprocessor



Race Condition

- Now every processor has a cache, and it can take data from its cache, not the main memory.
- Suppose one processor changes the content of the memory, what should happen to the cached data in some processor's cache?
- If one processor changes the data in its cache, would other processor be able to notice this change?

Cache



Hardware Solution

- The hardware must guarantee that the memory and cache are in a consistent state. There are various levels of guarantees.
- If one processor changes the content of the memory, the hardware should invalidate data that other processors have cached.
- If one processor changes the data in its cache, other processors should be able to see it.

Writing Policy

Write-through Write is done synchronously both to the cache and to the backing store.

Write-back Initially, writing is done only to the cache. The write to the backing store is postponed until the cache blocks containing the data are about to be accessed by others.

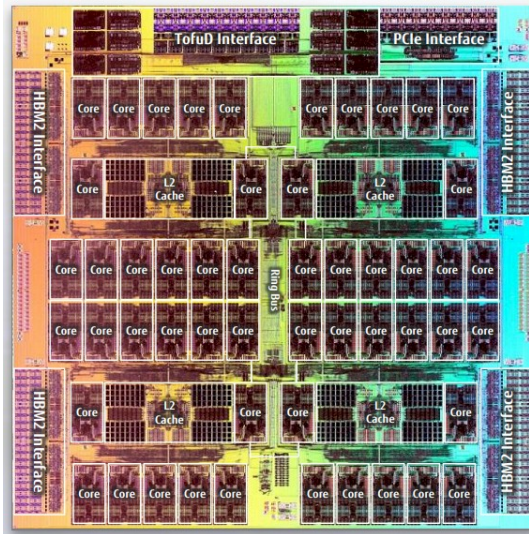
Discussion

- Describe the difference between write-through and write-back caching.
- Describe the advantages of write-through and write-back caching.

ARM/Fujitsu A64FX CPU

- The A64FX is the world's first processor to implement Scalable Vector Extension (SVE), an extension of the Armv8.2-A instruction set architecture for supercomputers.
- It has 48 calculation cores and two or four assistant cores, and has a theoretical peak performance of 3.3792 teraflops in double precision floating-point calculations.
- It is the building block of supercomputer Fugaku.

A64FX



A64FX Specifications

`https://www.fujitsu.com/downloads/SUPER/a64fx/a64fx_
datasheet_en.pdf`

L2 Cache

- The memory controller controls and coordinates the access to shared memory.
- Cores can communicate with the queue.
- Three cores share four L2 caches.
 - Shared L2 cache indicates that if we place wrong processes/threads into the cores that share the L2 cache, the performance will suffer.

Discusson

- Can you tell from the A64FX picture which cores share which L2 cache?

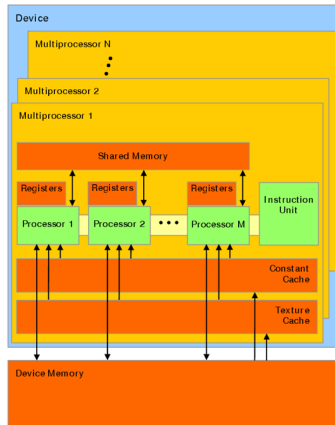
Nvidia RTX

- TITAN RTX is designed for researchers, developers and creators.
- TITAN RTX is based on Turing architecture bringing 130 Tensor TFLOPs of performance, 576 tensor cores, and 24 GB of ultra-fast GDDR6 memory to your PC.
- <https://www.nvidia.com/en-us/deep-learning-ai/products/titan-rtx/>

Nvidia RTX

- <https://developer.nvidia.com/blog/nvidia-turing-architecture-in-depth/>

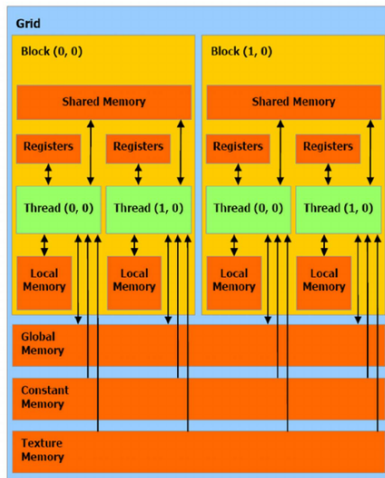
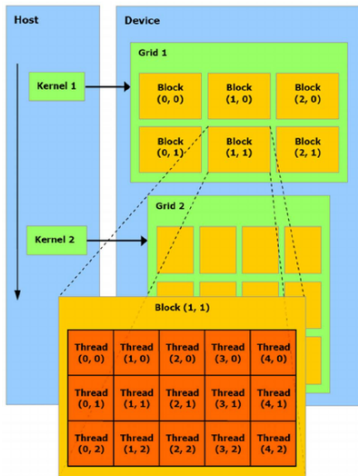
GPU Architecture



A set of SMT multiprocessors with on-chip shared memory.

Figure 3-1. Hardware Model

GPU Architecture



Characteristics

- A GPU is a device that needs a host to provide data and instructions.
- A GPU has many cores – usually much more than a CPU.
- CPU cores receive instructions from the host for execution, which means they must run the same set of instructions.
- GPU cores have their local memory and share a device memory.
- More details in the OpenCL and CUDA lectures.

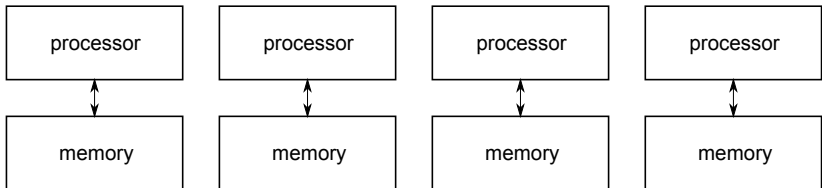
Discussion

- Describe the difference between CPU and GPU.

Multiple Uniprocessor

- It is intuitive to have multiple uniprocessors working together to have high performance.
- How do they work *together*?

Work Together

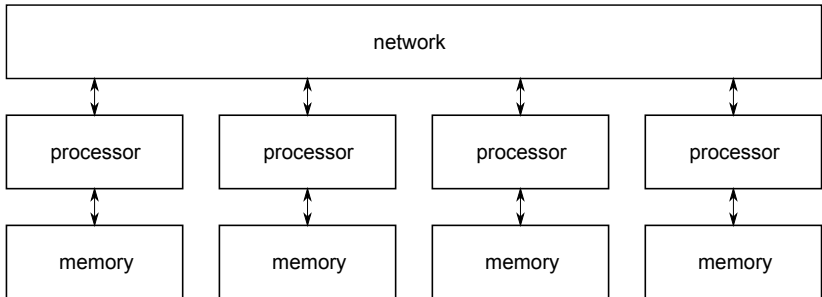


???

Multiple Uniprocessor

- We connect all the computers with a *network*, so they can send *messages* to each other.
- This network could be very slow or fast, depending on the applications.
- The network must provide *point-to-point* data transfer to move data from one processor to another.
- Processors can only synchronize themselves via the network, which is a difficult task.

A Network



Functions

- Every processor works on the tasks assigned to it, using its own computing resource.
- Every processor can send message to each other, so as to communicate or synchronize with other processors.

Multicomputer

- Sometimes we use the term *node*, since these processors are by themselves *computers*.
 - Remember the 16,000 nodes in the Tianhe 2 cluster.
- Each node has its own CPU's, memory, even I/O devices.
- The nodes can communicate with each other by the network, usually through standard TCP/IP protocol.

Memory

- There is no *shared addressing space* for all processors – each processor use its own memory.
- The memory of every processor is called *local memory*.
- Since a memory is accessed by only one processor, we do *not* have memory conflict.

Synchronization

- Processor can synchronize with each other by messages.
- In a barrier synchronization, a processor cannot proceed until all others have reached the same conclusion.
- Every processor sends a message to a master after finishing its work.
- When the master receives a message from every processor, he knows everybody has finished.
- The master sends a message to everyone that it can continue.

Discussion

- Compare the way multicomputer and multiprocessor does a barrier synchronization.

Network

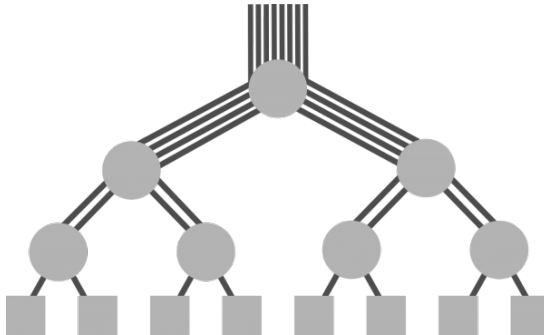
- We do care about the network – a fast network provide better connectivity.²
 - Ethernet 10Mbit/s
 - Fast Ethernet 100Mbit/s
 - Gigabit Ethernet 1 Gbit/s
 - 10 Gigabit Ethernet 10 Gbit/s
 - Myrinet 10 Gbit/s
 - 100 Gigabit Ethernet 100 Gbit/s
 - InfiniBand (12X EDR) 300 Gbit/s

²http://en.wikipedia.org/wiki/Network_bandwidth

Topology

- We cannot connect a large number of nodes to a single switch, so the topology of the network becomes important.
 - Ring
 - Tree and fat tree
 - Two or higher dimensional mesh and torus
 - Hypercube
 - FFT (butterfly)

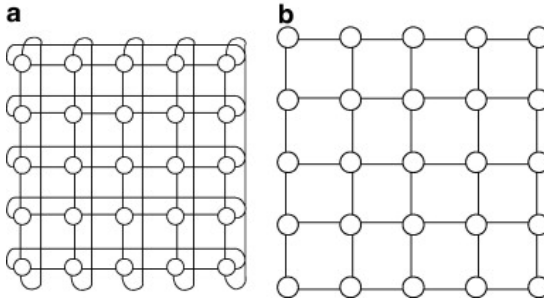
Fat Tree



3

³http://clusterdesign.org/wp-content/uploads/2012/02/fat_tree_varying_ports-600x365.png

Mesh and Torus

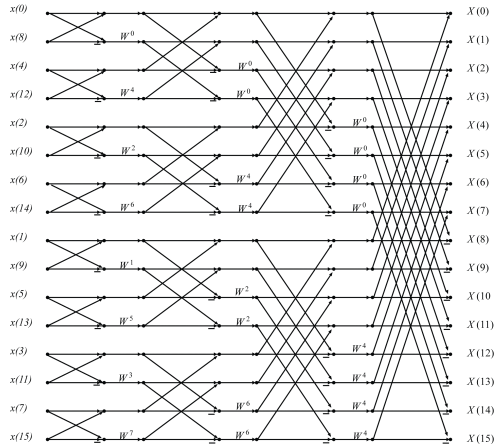


4

⁴[http:](http://ars.els-cdn.com/content/image/1-s2.0-S1383762107000495-gr2.jpg)

[//ars.els-cdn.com/content/image/1-s2.0-S1383762107000495-gr2.jpg](http://ars.els-cdn.com/content/image/1-s2.0-S1383762107000495-gr2.jpg)

FFT



Trend

- The most popular topology appears to be fat tree.
 - For example, the Tianhe 2 cluster connects all its processors with 13 switches as fat tree.
- Two or higher dimension torus are also popular.
 - The Tofu network of K-computer is a six dimensional torus.

Discussion

- What is the difference between torus and a mesh?

Massively Parallel Computer

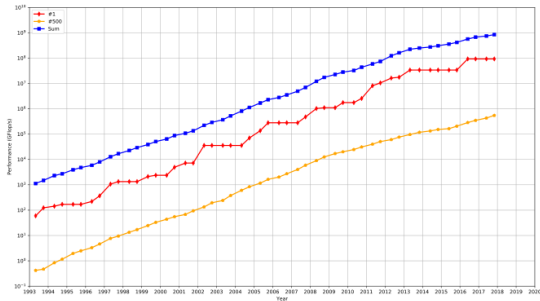
- A computer cluster consists of a set of *tightly* connected *high performance* computers that work together so that in many respects they can be viewed as a single system.
- Tightly connected means they connected by *fast* network.
- The performance is paramount and usually achieved by aggregation of a large number of processors.

Top 500

- The TOP500 project ranks and details the 500 most powerful (non-distributed) computer systems in the world.
- The project aims to provide a reliable basis for tracking and detecting trends in high-performance computing and bases rankings on HPL, a portable implementation of the High-Performance LINPACK benchmark written in Fortran for distributed-memory computers.⁶

⁶<http://top500.org/>

Top 500



https://commons.wikimedia.org/wiki/File:Top500_supercomputers_performance_development.svg

K computer

- The K computer is named for the Japanese word *kei*, meaning ten quadrillions (10^{16}).
- A supercomputer manufactured by Fujitsu, currently installed at the RIKEN Advanced Institute for Computational Science campus in Kobe, Japan.
- In June 2011, TOP500 ranked K the world's fastest supercomputer, with a rating of over eight petaflops, and in November 2011, K became the first computer to top 10 petaflops.
- http://en.wikipedia.org/wiki/K_computer

K Computer



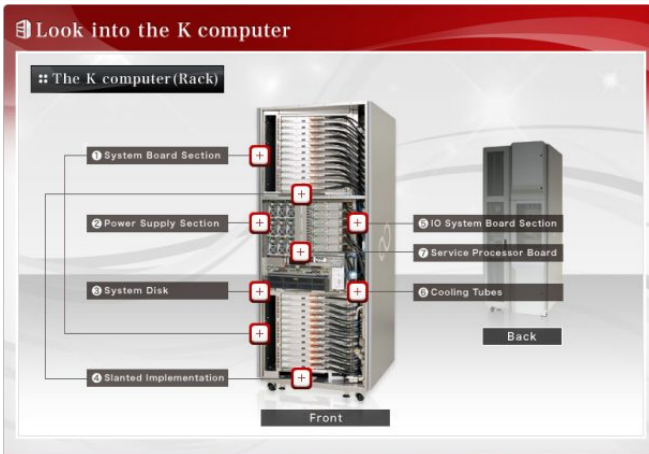
7

⁷http://cdn0.sbnation.com/entry_photo_images/2197300/k-computer_large_verge_medium_landscape.jpg

Configuration

- 864 cabinets, 88,128 SPARC64 VIIIfx processors, over 640,000 cores.
- A proprietary six-dimensional torus interconnect called Tofu.
- A two-level local/global file system with parallel/distributed functions, which provides users with an automatic staging function for moving files between global and local file systems.
- Linux operating system.
- 9.89 MW – the equivalent of almost 10,000 suburban homes.

K Computer



8

⁸<http://cdn-static.zdnet.com/i/story/30/40/093162/k-computer-riken-4.jpg>

Discussion

- What is the difference between supercomputer K-computer and Fugaku?

Cluster

- A computer cluster consists of a set of *loosely* connected computers that work together so that in many respects they can be viewed as a single system.
- Loosely connected means they are *not* connected by fast network.
- An economical alternative to those who cannot afford expensive parallel computers.

TrendMicro Cluster

- A cluster donated by TrendMicro for the Cloud Computing Program.
- Loosely connected means they are not connected by fast network.
- A economical alternative to those who cannot afford expensive parallel computers.

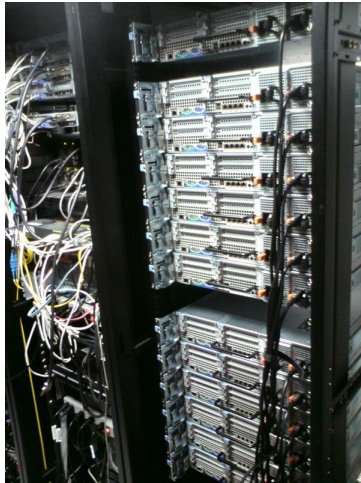
Configuration

- 1 cabinet, 15 Intel X5570 quad-core processors, 120 cores.
- It has a standard Ethernet switch.
- Gluster file system.
- Roystonea operating system, developed by the Parallel and Distributed Processing Laboratory, Department of Computer Science and Information Engineering, National Taiwan University.
- Never made it to top 500.

TrendMicro Cluster



TrendMicro Cluster



- We (the Parallel and Distributed Processing Laboratory) learn many things in building Roystonea for the Trend cluster.
- We learned network virtual machine management, virtual machines deployment, network management for the cluster, distributed file system for the cluster, distributed database, and NoSQL database for a cloud system.
- We built a cloud OS called *Roystonea* to manage the Trend cluster.
- These experiences later enable us to take on more ambitious projects, like optimizing the billing system of Chunghwa Telecommunication.

Discussion

- What does the name Roystonea come from?

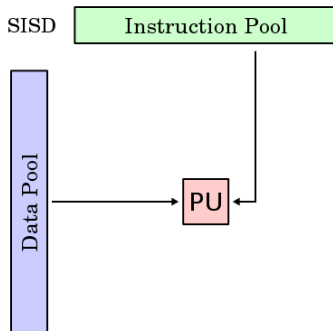
Flynn's Taxonomy

- Single instruction stream and multiple instruction stream.
- Single data stream and multiple data stream.
- We have four combinations – SISD, SIMD, MISD, and MIMD.

SISD

- SISD (single instruction, single data)
- A computer architecture in which a single uniprocessor executes a single instruction stream to operate on a data stream.
- Standard von Neumann architecture.

SISD



Old School

- This is how we have been doing – sequential programming.
- The compiler for sequential programming is quite mature and can transform source program into efficient binaries.
- System support for sequential programming, e.g., system call, user library, debugging are also very useful.

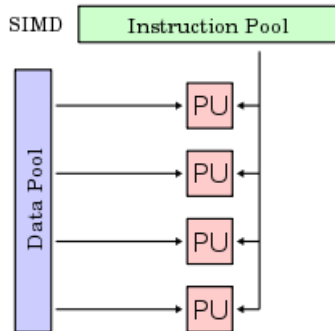
Discussion

- Give an example of SISD machine.

SIMD

- Single instruction, multiple data (SIMD)
- A computer architecture in which multiple processors execute a single instruction stream to operate on multiple data streams.

SIMD



10

¹⁰<http://en.wikipedia.org/wiki/SIMD>

SIMD

- SIMD is strongly related to data-parallel programming since it applies the same instruction to different data and achieves high performance by data parallelism.
- The architecture of GPUs follows the SIMD model – the host issues the same command to all processors to process a large amount of data simultaneously.

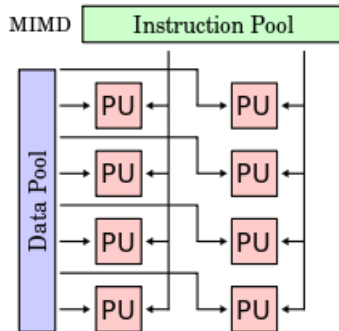
Discussion

- Give an example of SIMD machine.

MIMD

- Multiple instruction, multiple data (MIMD)
- A computer architecture in which multiple processors execute multiple instruction streams to operate on multiple data streams.

MIMD



11

¹¹<http://en.wikipedia.org/wiki/MIMD>

MIMD

- Strongly related to functional parallelism since different processors execute different instructions on different data.
- The instructions are different because they are from different tasks in a wavefront, i.e., tasks that can run in parallel.
- The data are different because they are for different tasks.
- Most multicomputers support the MIMD computation model – computers work independently and synchronize themselves when necessary.

Discussion

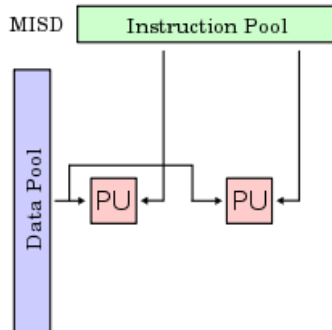
- Give an example of MIMD machine.

Wait! You forgot MISD!

MISD

- Multiple instruction, single data (MISD)
- A computer architecture in which multiple processors execute multiple instruction streams to operate on a single data stream.
- Does it make sense to you?

MISD



12

¹²<http://en.wikipedia.org/wiki/MISD>

Fault Tolerance

- If we consider MISD as multiple copies of SISD, it can tolerate faults.
- The same computation is repeated multiple times by different processors so that at least they can deliver the results even if some of them fail.
- For example, in Google MapReduce computation, certain tasks are duplicated exactly for fault tolerance and performance improvement.

Discussion

- Give an example of MIMD machine.