Computer Architecture 1

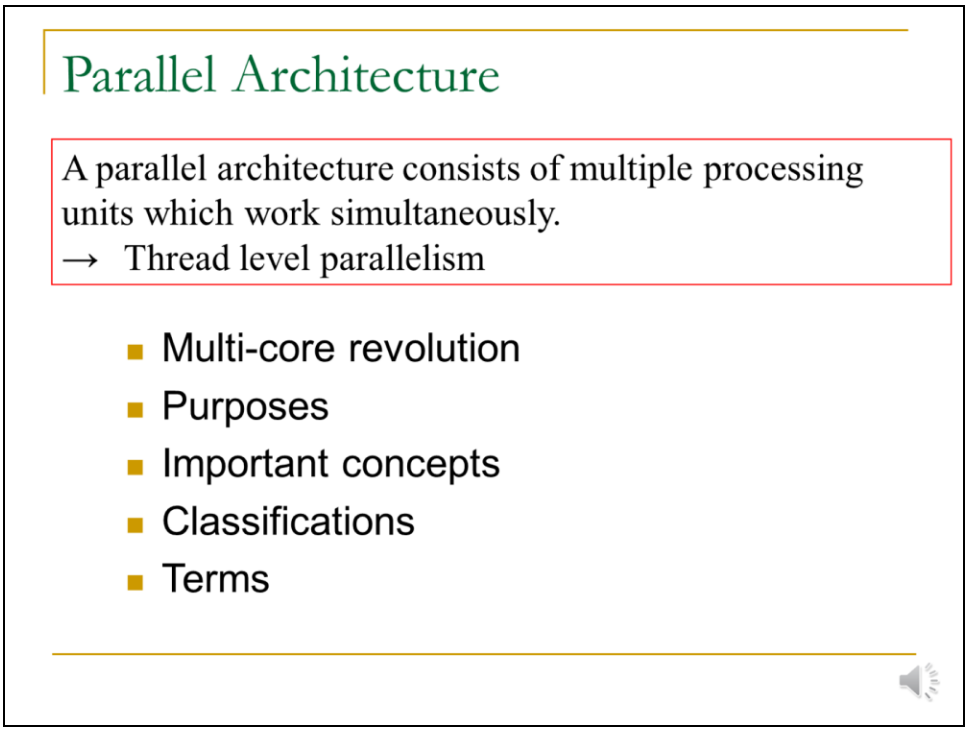
Introduction to Parallel Architectures

Keio University
AMANO, Hideharu
hunga@am. ics. keio. ac. jp

Thank you for taking this class; Special Course of Computer Architecture. This class focuses on a class of computers: parallel computers.

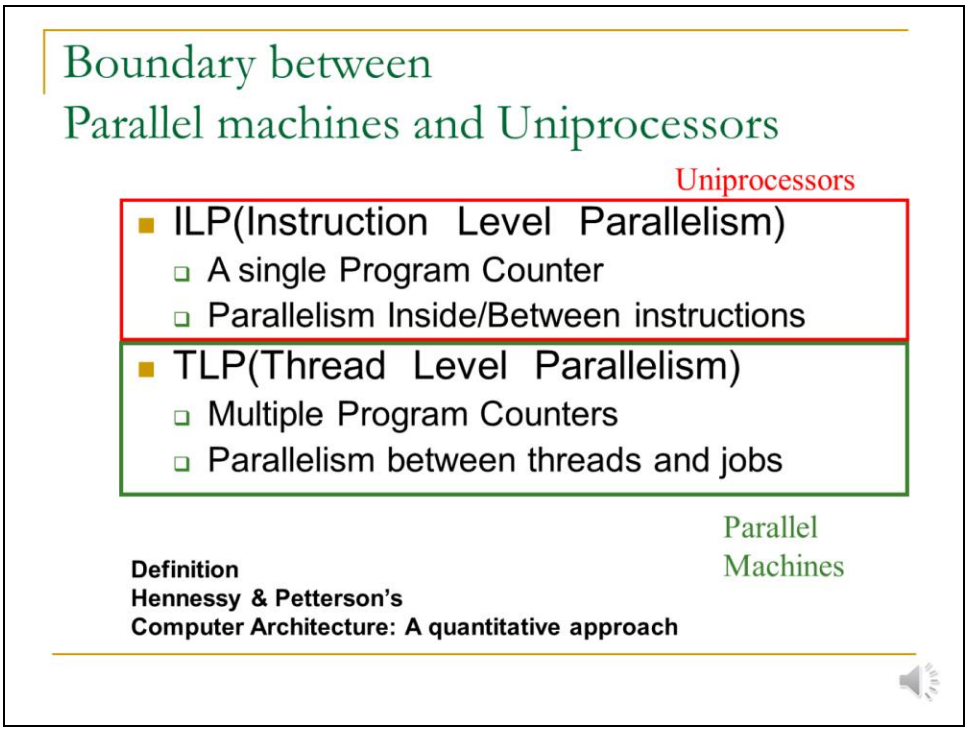
Recently most of the computers have been parallel computers. So, this course treats a wide are of computers. My name is Amano.

Today, we have the first lesson which introduces parallel computer architecture.



The definition of parallel architectures is simple. A computer that consists of multiple processing units working simultaneously. It make the use of thread level parallelism explained later.

Today, I will talk on …



First, this slide shows the boundary between parallel machines and uniprocessors.

Every computer uses parallel execution mechanism, but uniprocessors use ILP or instruction level parallelism.

That is, there is only a single program counter, and the parallelism inside or between instructions is utilized.

On the contrary, parallel machines use mainly the TLP or thread level parallelism. There are multiple program

counters, and parallelism between processes, tasks or threads sometimes jobs are utilized.

Uniprocessor Performance

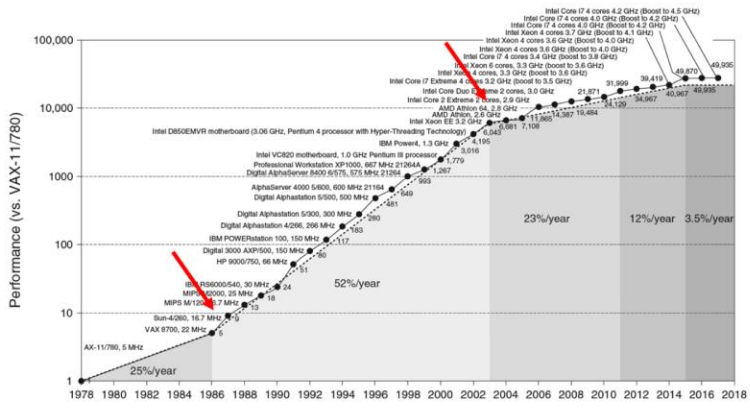


Figure 1.1 Growth in processor performance over 40 years. This chart plots program performance relative to the VAX 11/780 as measured by the SPEC integer benchmarks (see Section 1.8). Prior to the mid-1980s, growth in processor performance was largely technology-driven and averaged about 22% per year, or doubling performance every 3.5 years. The increase in growth to about 52% starting in 1986, or doubling every 2 years, is attributable to more advanced architectural and organizational ideas typified in RISC architectures. By 2003 this growth led to a difference in performance of an approximate factor of 25 versus the performance that would have occurred if it had continued at the 22% rate. In 2003 the limits of power due to the end of Dennard scaling and the available instruction-level parallelism slowed uniprocessor performance to

Hennessy & Patterson "Computer Architecture a quantitative approach 6th Edition"

Parallel architectures have become popularly used from 2003 or 2004. This diagram shows how the relative performance has been improved year by year. Note that the vertical axis uses a log-scale. From 1986, the performance of uniprocessors was improved 57% per year. It means that the performance was doubled for every 18 months. It is sometimes called Moore's Low. But, from 2003, the performance improvement was suddenly decayed and this tendency has been kept now. From 2003, Intel and other computer makers changed their policy to increase the number of processors or cores in a chip rather than improving the performance of each processor. That is, the multi-core revolution happened at that time.

Multicore Revolution

1. The end of increasing clock frequency
   1. Consuming power becomes too much.
   2. A large wiring delay in recent processes.
2. The gap between CPU performance and memory latency
3. The limitation of ILP

Since 2003, almost every computer became multi-core.
Even smartphones use 2-8-core.

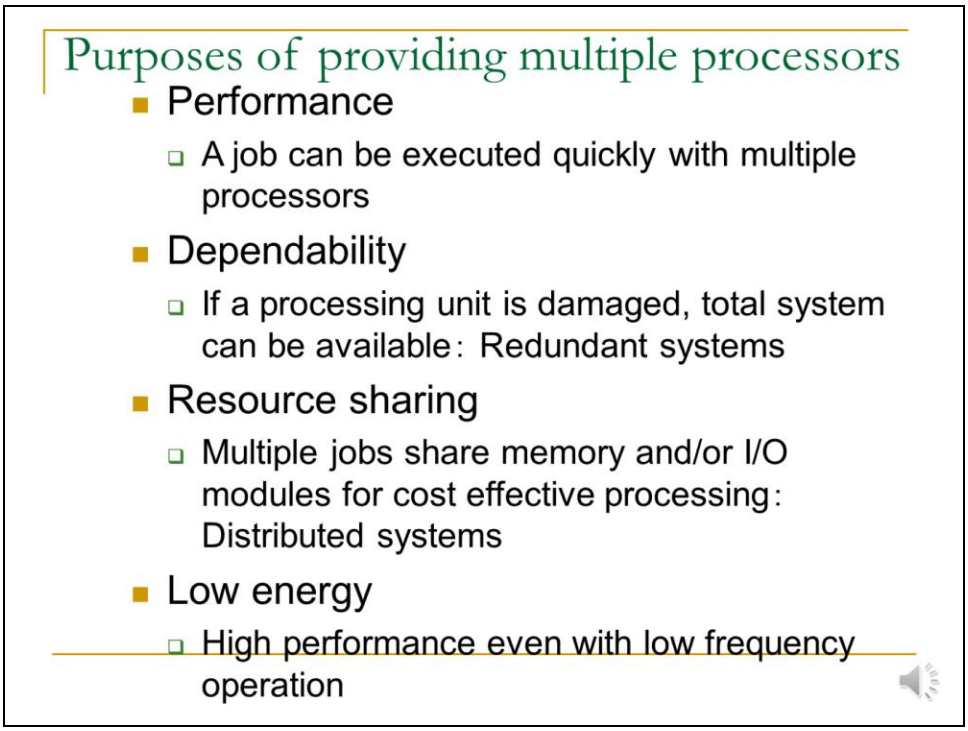
Now, except forsmall micro-controllers, every computer is a parallel computer.

OK. but why the computer makers changed their policy. There were mainly three reasons. First increasing clock frequency became difficult. Since the power consumption is proportional to the clock frequency, at that time, the power consumption of a single processor reached more than 100W. It means that a large heat sink and fun were needed. Also, the operational speed of semiconductor was saturated. The performance of the transistors was still improved, but the total frequency was limited with a wiring delay. If you designed a large processor, it was difficult to keep the performance.

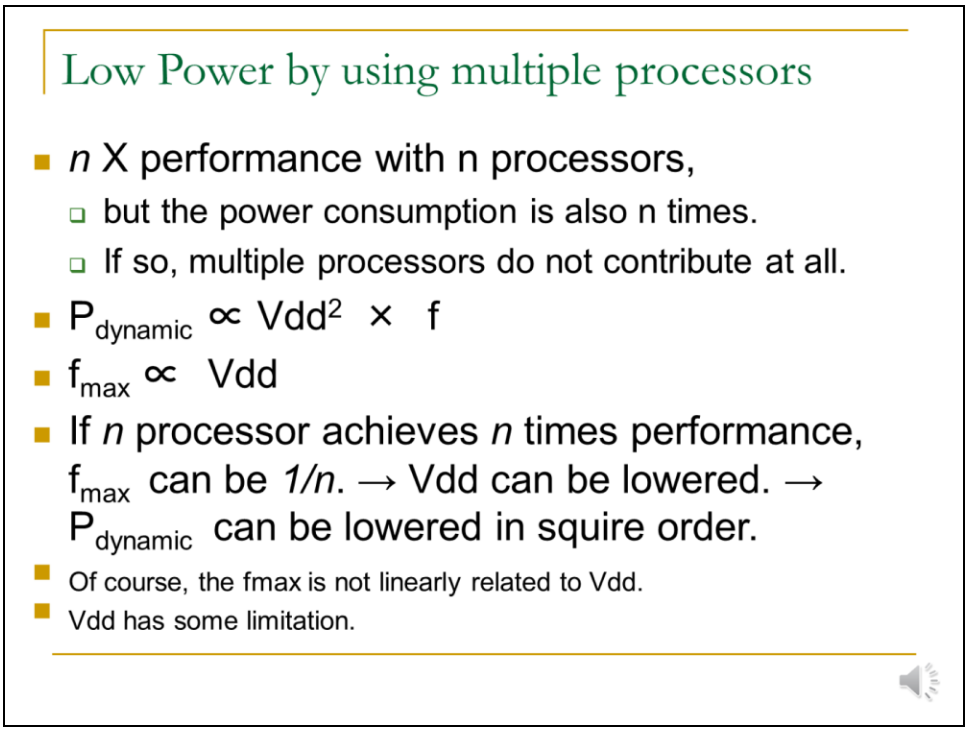
Then, there was also a problem called Memory Wall. That is, the memory performance was also saturated, and if the CPU performance was improved, the total performance might not increases because of memory bottleneck.

Another reason is the limitation of Instruction Level Parallelism. Until 2003, various types of techniques have been invented, but the instruction level parallesim to be used reached a limitation.

For these reasons, since 2003, almost every computer became multi-core, and now even smartphones use from 2 to 8-core processors. Now, except for small micro-controllers, every computer is a parallel computer.



There are several benefit to have multiple processors…. The last one, low energy computing will need further explanation.



It seems strange that low power computing is achieved by using multiple processors. If we use n processors, we can achieve n times performance at maximum, but the power consumption becomes also n times. Thus, multiple processors do not contribute for low power at all. However, note that the dynamic power is proportional to Vdd, the supply voltage square and operational frequency f. Also, the operational frequency f is proportional to Vdd in a certain range, it means that if we want to use high frequency the supply voltage must be high. Here, if we can achieve n times performance with n processors, we can lower the Vdd to achieve the same performance. Since dynamic power is proportional to Vdd squire, we can much lower the total power consumption. Of course the maximum frequency is not proportional to very low Vdd, this theory cannot be established ideally. But, parallel processing is still useful for low power computing.

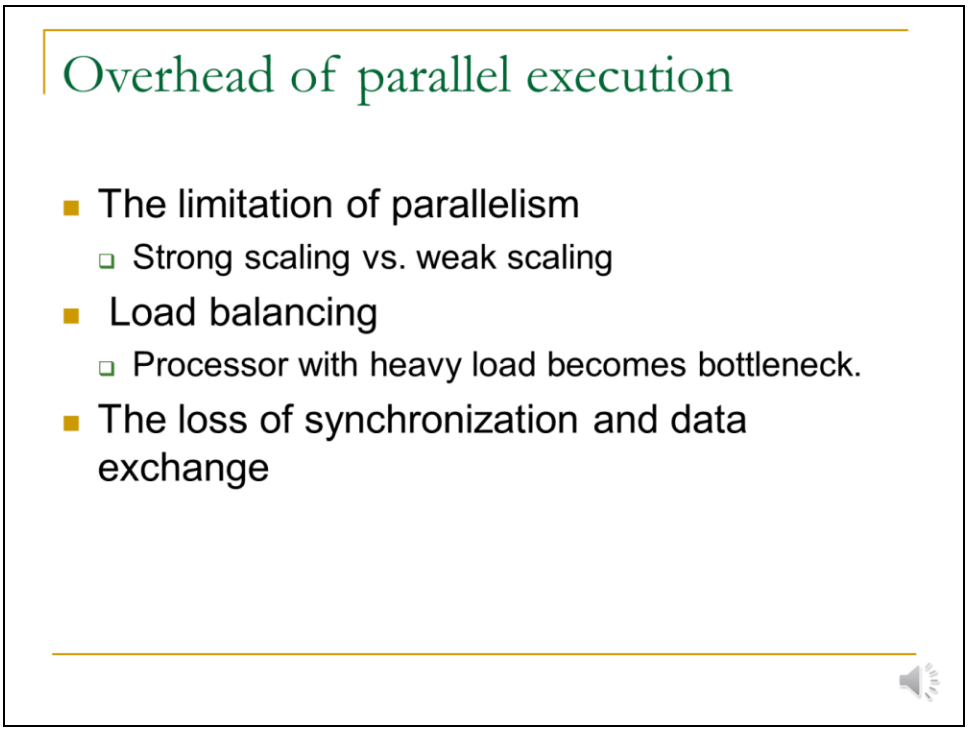
Quiz

- Assume a processor which consumes 10W with 1.8V Vdd and 3GHz clock.
- You can improve performance by 10x with 10 processors, it means that the same performance can be achieved with 300MHz clock.
- In this case, Vdd can be 1.0V.
- How much power does the machine with 10 processors consume?

Parallelism in a target application

- For high speed execution of a single job
  - TLP: Thread(Task)-level parallelism: a job is divided into threads executed in parallel by a compiler/programmer.
  - DLP: Data-level parallelism: an operation is applied to a lot of data in parallel.
- RLP: Request Level Parallelism
  - Execution of multiple-jobs independent from each other.
  - Easy to execute in parallel.
  - Request level parallelism?

→Data centers treat such parallelism



Overhead is required for parallel execution. First there is the limitation of parallelism. A program cannot be executed completely in parallel.

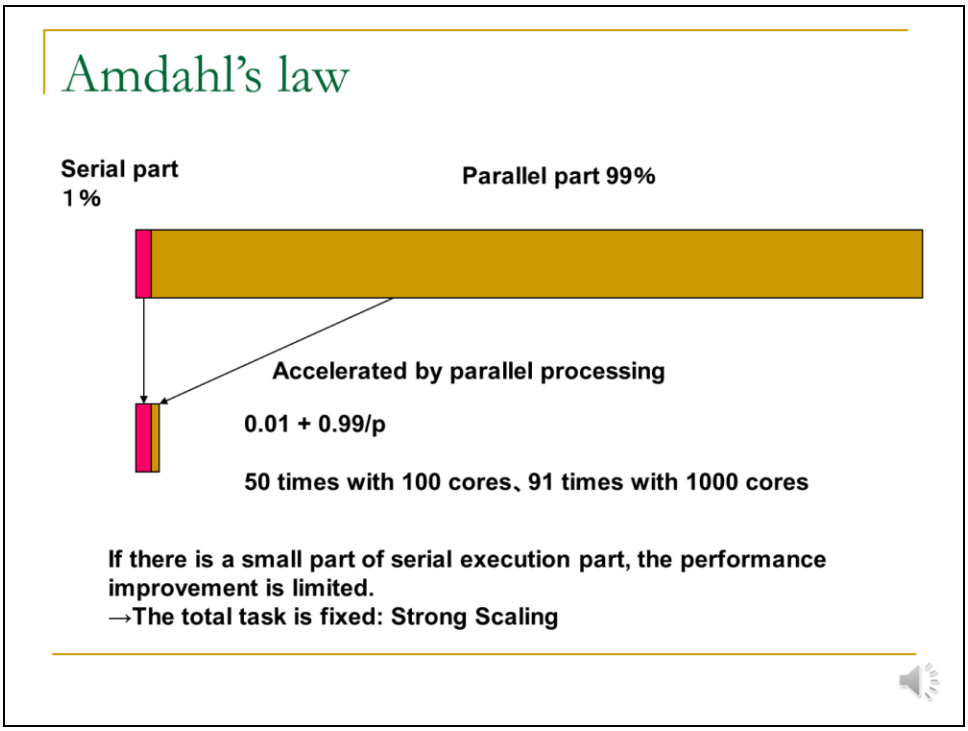
If strong scaling is used, this limitation becomes severe.

Second, the computational load cannot always distributed evenly. If load balancing is not kept, the processor with heavy load becomes bottleneck.

Load balancing is important problem, but in this lesson, I will skip this issue, since it is not in a field of computer architectures.

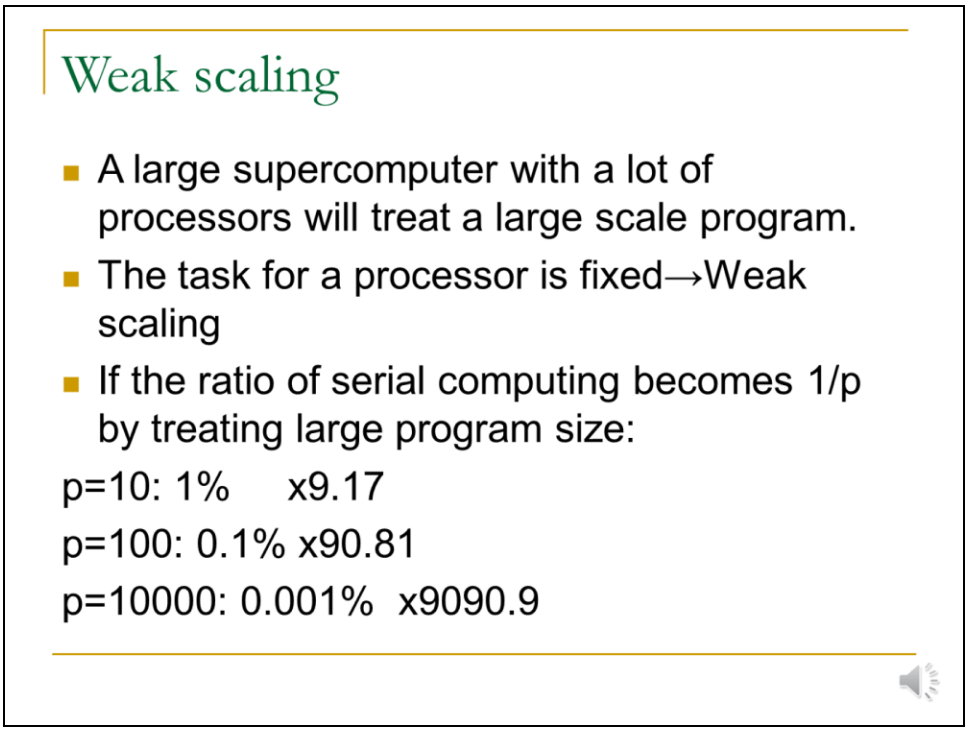
Third, the time for synchronization and data exchange is added to the execution time. In this lesson, I will introduce the efficient synchronization method and high speed interconnection network in this lesson.

Today, I will explain about the first issue.



May be most of you know of the famous Amdahl's law. The portion which can apply the enhancement method limits the total performance enhancement. Assume that there is only one percent of the job which cannot be executed in parallel, and the other 99% is completely parallel executable. In this case, if we execute it with 100 cores, 50 times performance improvement is obtained. But if we use 1000 cores only 91 times performance improvement is achieved. That is, the performance improvement never beyond 100 times even with infinite number of cores.

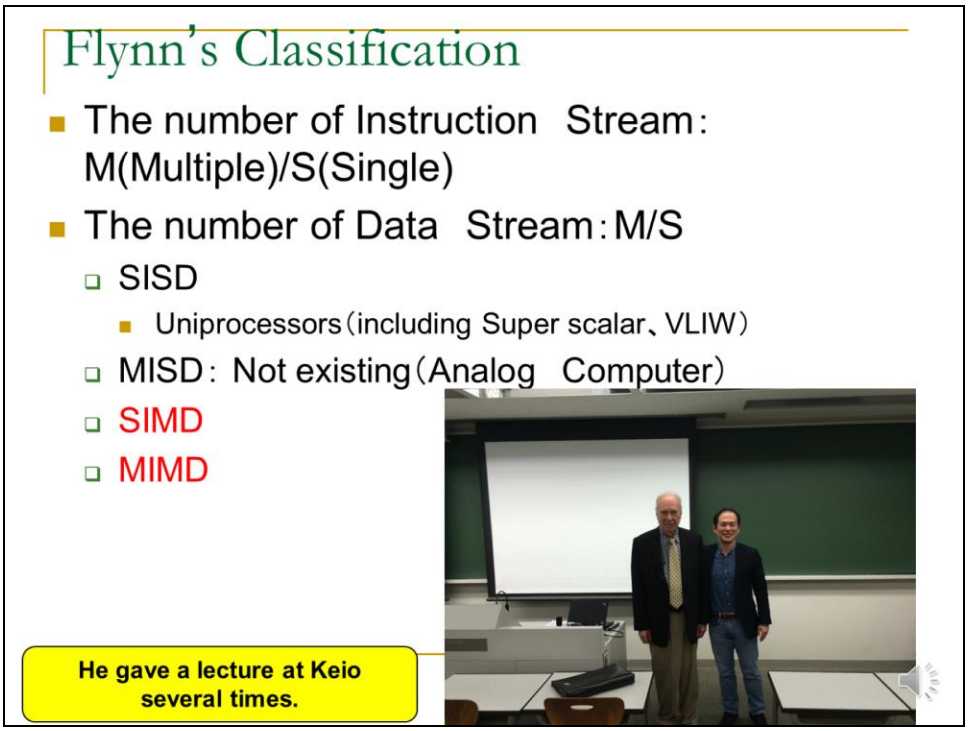
The performance measurement with a fixed task is called strong scaling. When we use strong scaling, a large size parallel machine is hopeless.



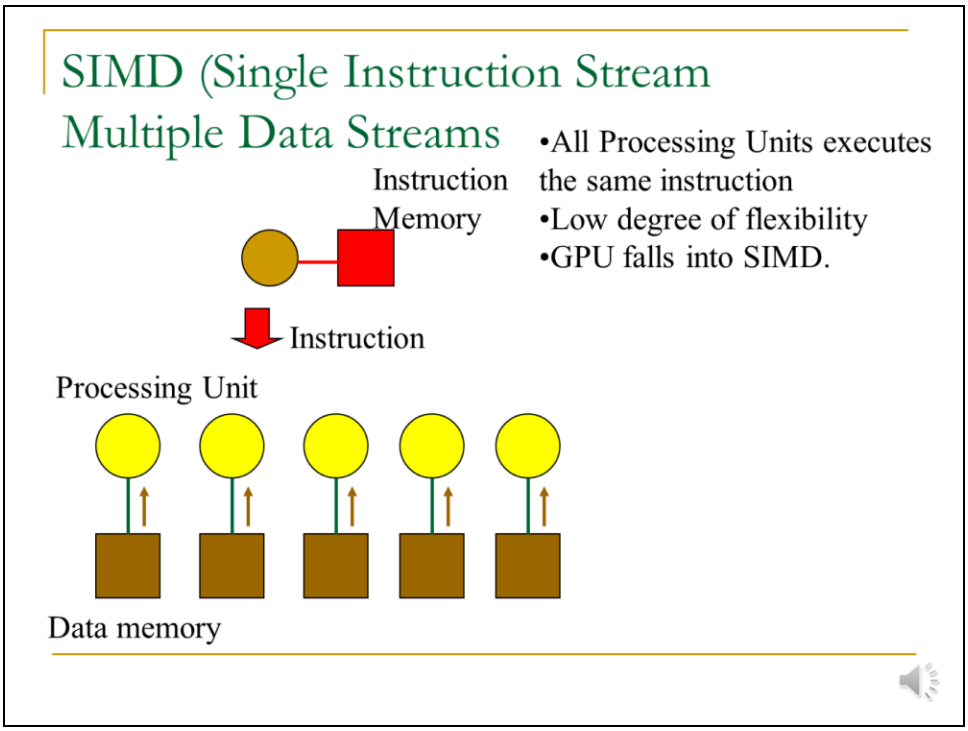
However, in reality, supercomputers with huge number of cores have been developed. Why? Because they are mainly used for a large scale program for its scale. When the task for a each processor is fixed, it is called weak scaling. Generally, the portion of parallel execution part becomes large for larger problems. If the ration of serial computing part becomes 1/p by treating large program size, the performance keeps improving. Some people feels that the weak scaling is unfair. But, it is used for supercomputers which treat only huge scale problems.

glossary 1

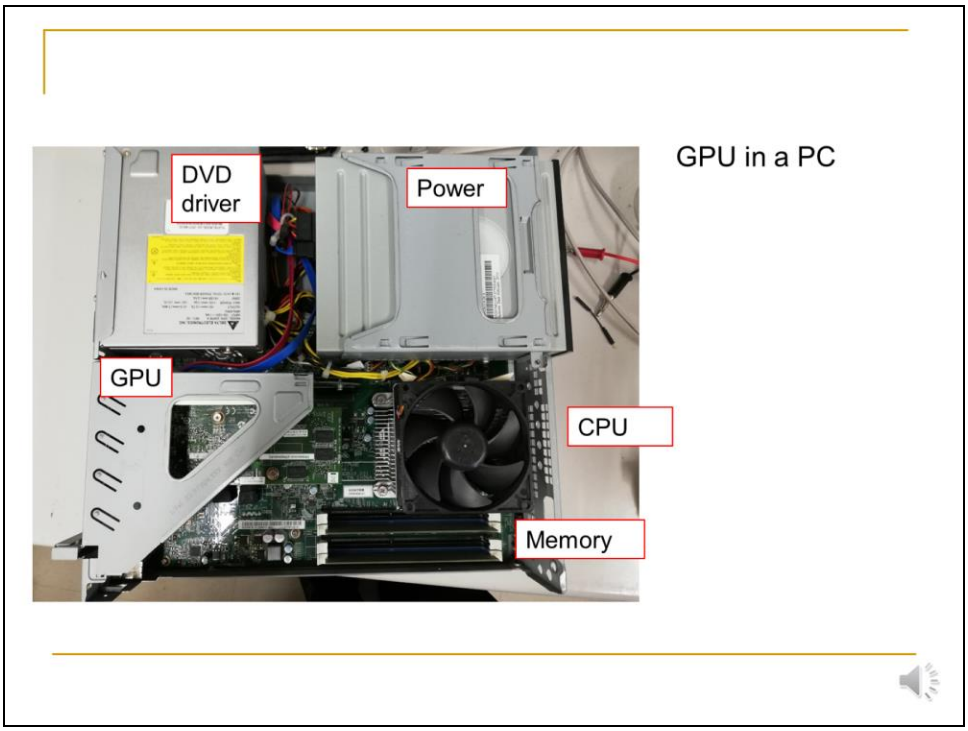
- Simultaneously: 同時に、という意味でin parallelとほとんど同じだが、ちょっとニュアンスが違う。in parallelだと同じようなことを同時にやる感じがするが、simultaneouslyだととにかく同時にやればよい感じがする。
- Thread: プログラムの一連の流れのこと。Thread level parallelism（TLP)は、Thread間の並列性のことで、ここではHennessy and Pattersonのテキストに従ってPCが独立している場合に使うが違った意味に使う人も居る。これに対してPCが単一で命令間にある並列性をILPと呼ぶ
- Dependability: 耐故障性、Reliability(信頼性), Availability(可用性）双方を含み、要するに故障に強いこと。Redundant systemは冗長システムのことで、多めに資源を持つことで耐故障性を上げることができる。
- Distributed system:分散システム、分散して処理することにより効率的に処理をしたり耐故障性を上げたりする



Let me move to the next section. The classification. First of all. I will talk about Flynn's classification. Professor Flynn is still a full professor of Stanford University U.S.A. He is very aged man, but still active duty. In United State, if Professors are successful in business and hire themselves, they can be active duty after the retirement age. He was successful to start a venture company Maxceller and may be he will work as he like. He likes Japan and sometimes comes and makes a lecture here in Keio University. He proposed famous Flynn's classification which classifies computers with the number of instruction stream and data stream in 1979. SISD is just a uniprocessor. MISD is treated as non-existing machines. So, SIMD and MIMD are frequently used words.



Single Instruction Stream Multiple Data Streams or SIMD executes a single instruction with a lot of processing units together. Each processing unit has its own data memory and according to the instruction fetched from the instruction memory, all execute the same operation. The advantage of SIMD is its simple structure and it is suitable to DLP. The problem is low degree of flexibility. The performance sometimes severely degraded if some of processors want to execute different operations from others. The GPU(Graphic Processing Unit) falls into this category.

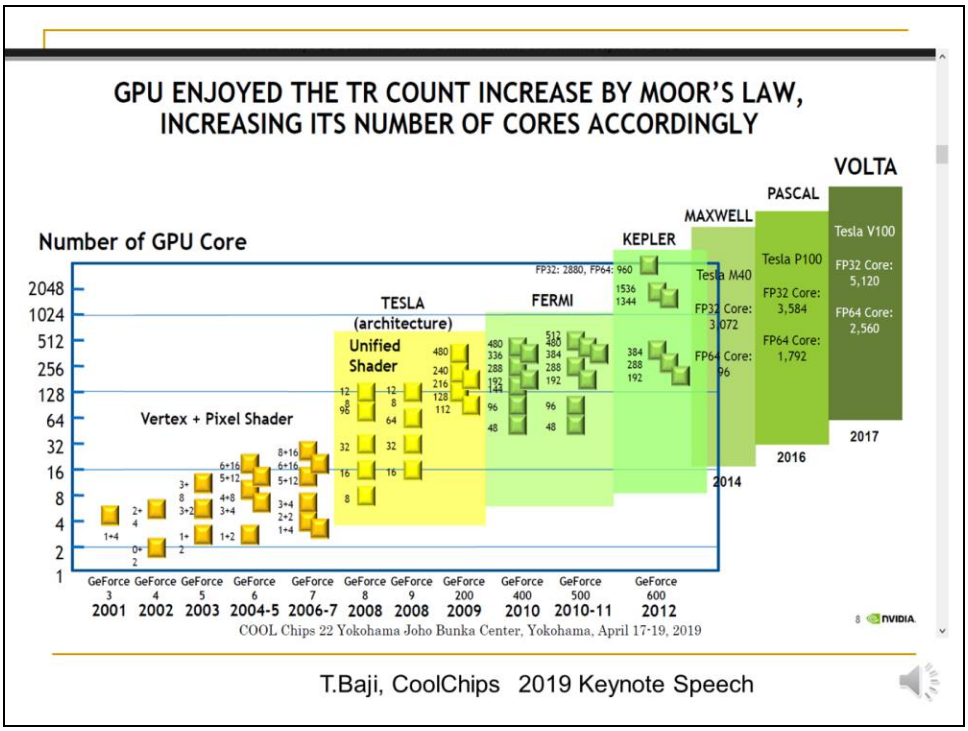


Graphic Processing Unit or GPU is used most of personal computers. The unit is pushed into a slot of mother board as shown this photo.

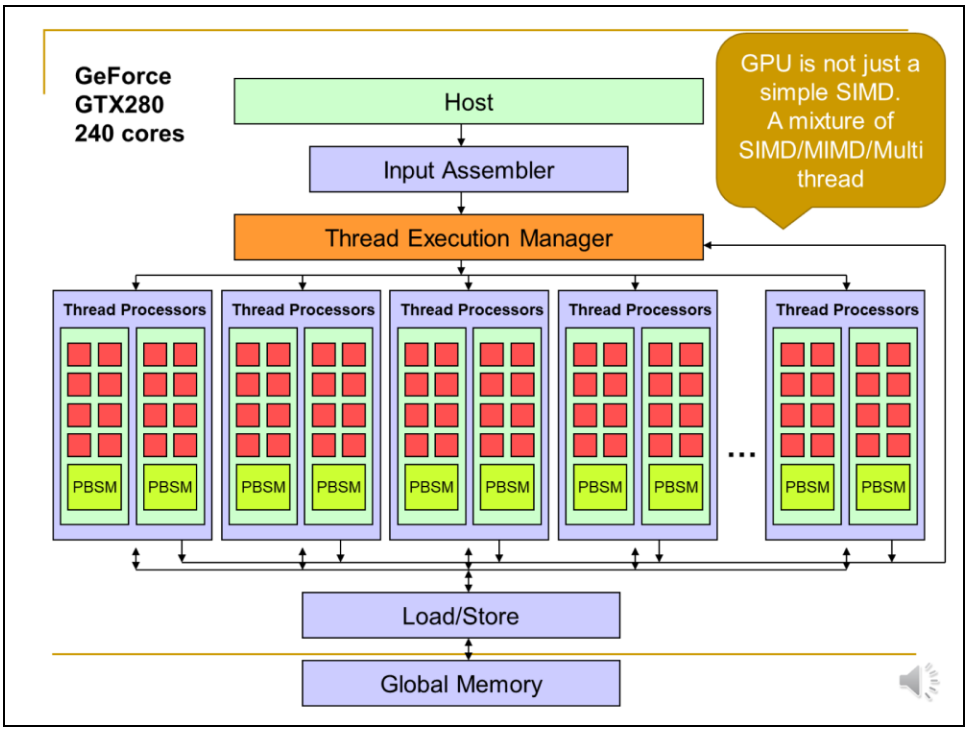
GPU(Graphics Processing Unit)

NVIDIA Quadro

This GPU is NVIDIA's Quadro. It is rather dedicated machine for graphics processing. However, GPU can be used for wide application. The general purpose processing with GPU is called GPGPU.

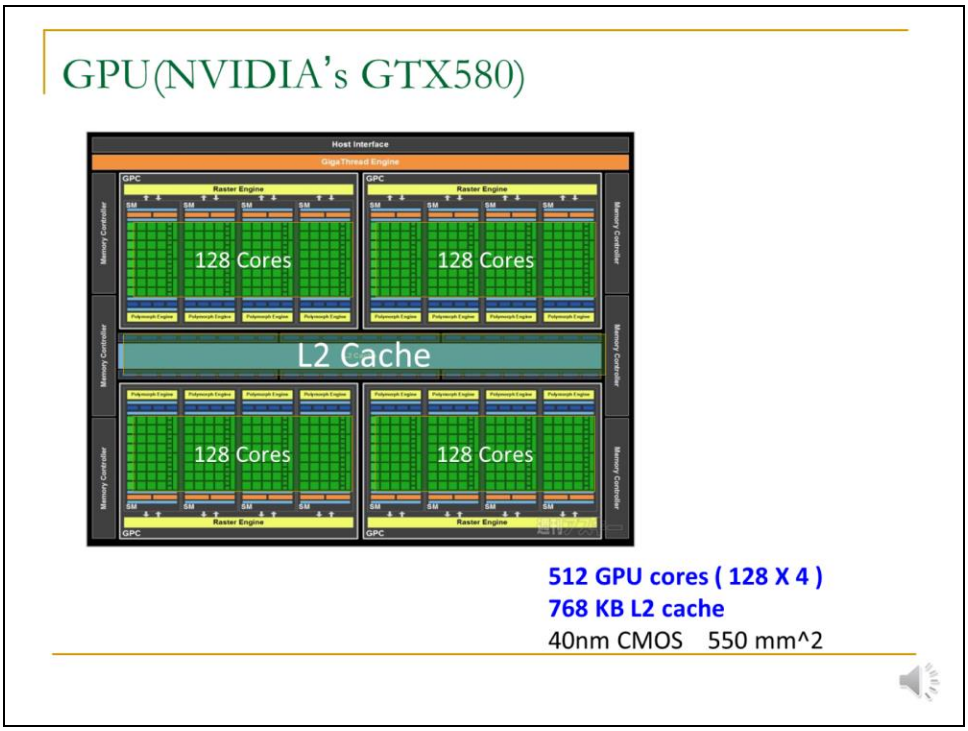


This slide is by Toru Baji-san of NVIDIA in his keynote speech in 2019 CoolChips. GPU enjoyed the advance of semiconductors and increased the number of cores. Now, more than 5000 cores are embedded in a chip.

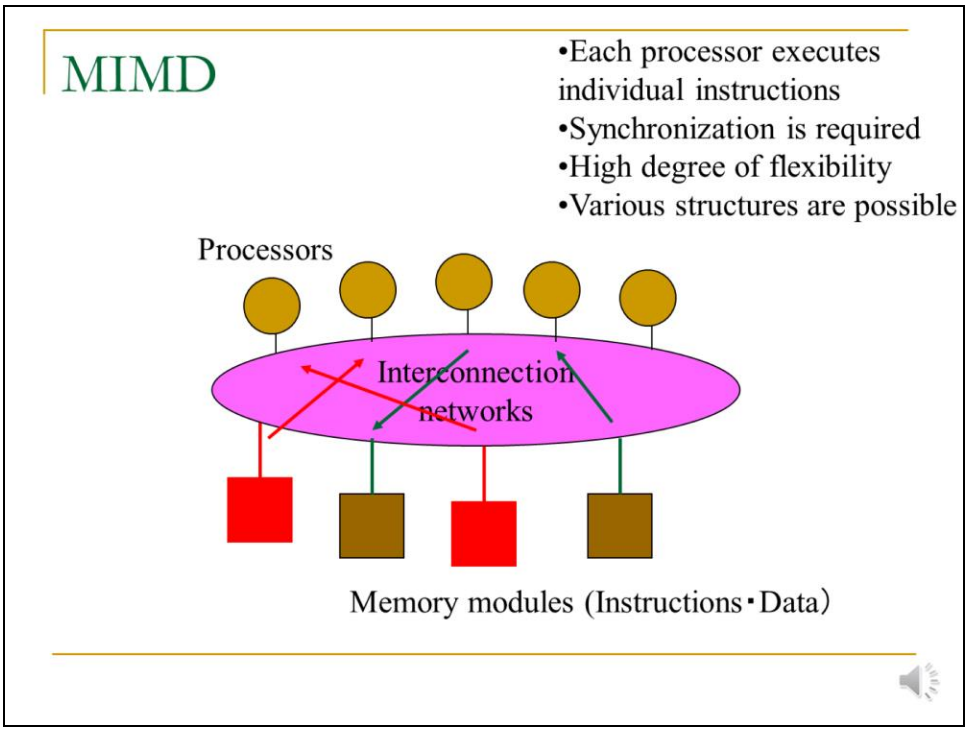


In GPU, a lot of cores called CUDA-core works with a single instruction, but it is not a simple SIMD machine. The techniques of MIMD machines and multithreading are used. Now GPU is a most frequently used accelerator for supercomputers. Also it is a key device for AI application and even for automatic driving control.

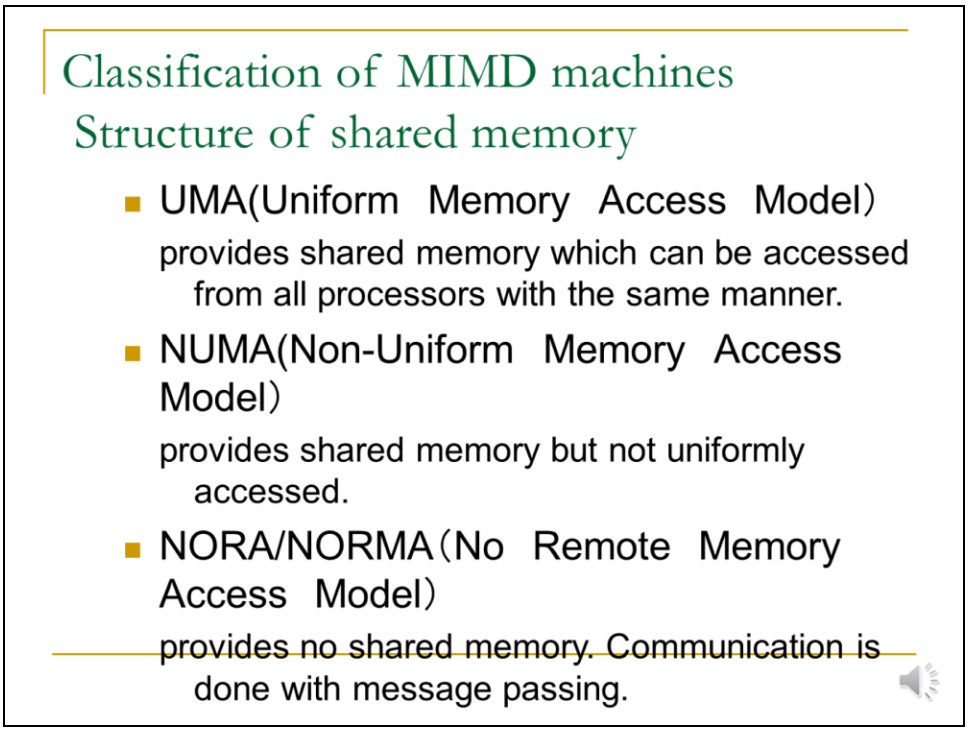
I will introduce this architecture later in this class.



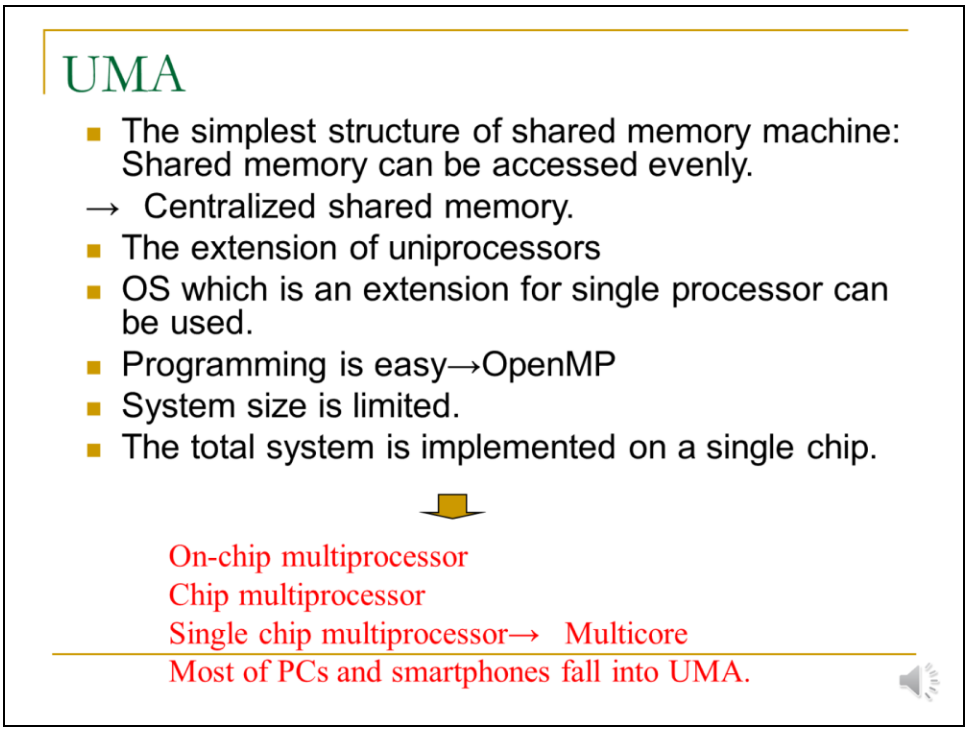
This photo shows the chip layout of NVIDIA's GPU. You can see that a large area of the chip is occupied with cores.



Unlike SIMD, MIMD fetches individual instructions and execute them. For parallel processing, synchronization is required. Since it is a straight forward extension of the uniprocessor, most of PCs, servers, and even smartphones are MIMD machines.

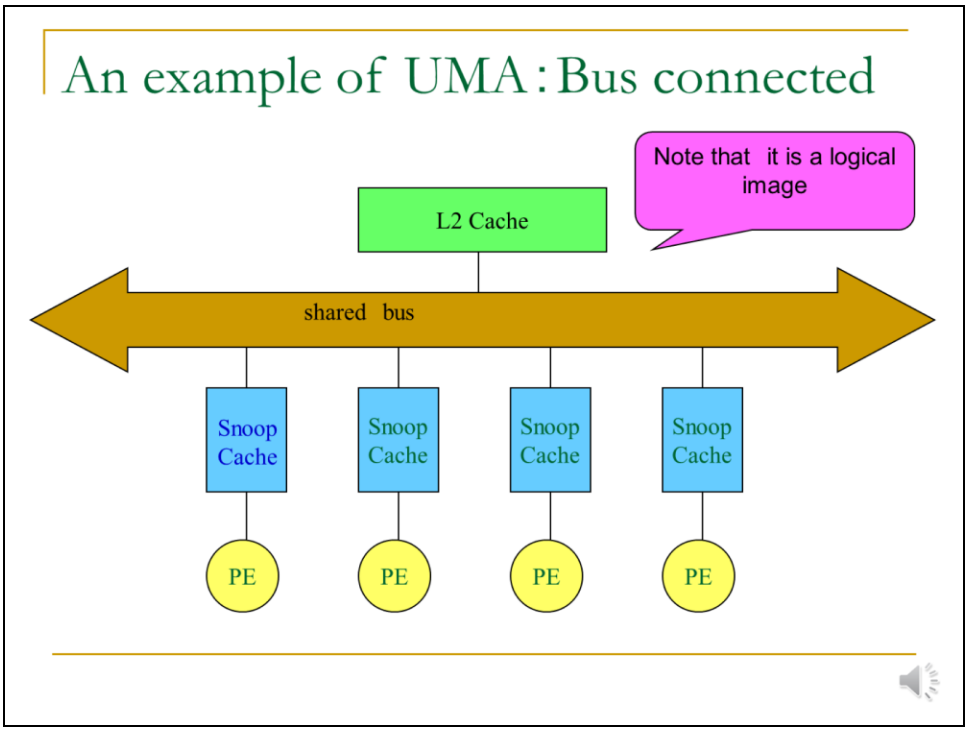


MIMD machines are further classified with their structure of shared memory. UMA, NUMA and NORA or NORMA.

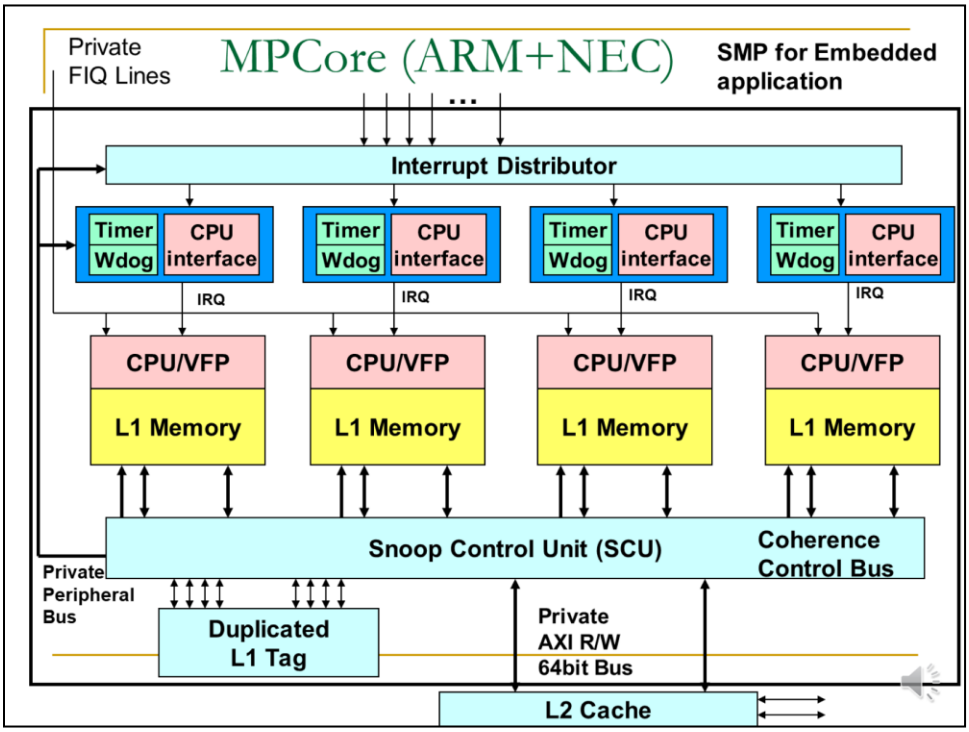


UMA has the simplest structure of shared memory machine. It provides shared memory which can be accessed by all PEs evenly. That is, it provides centralized shared memory.

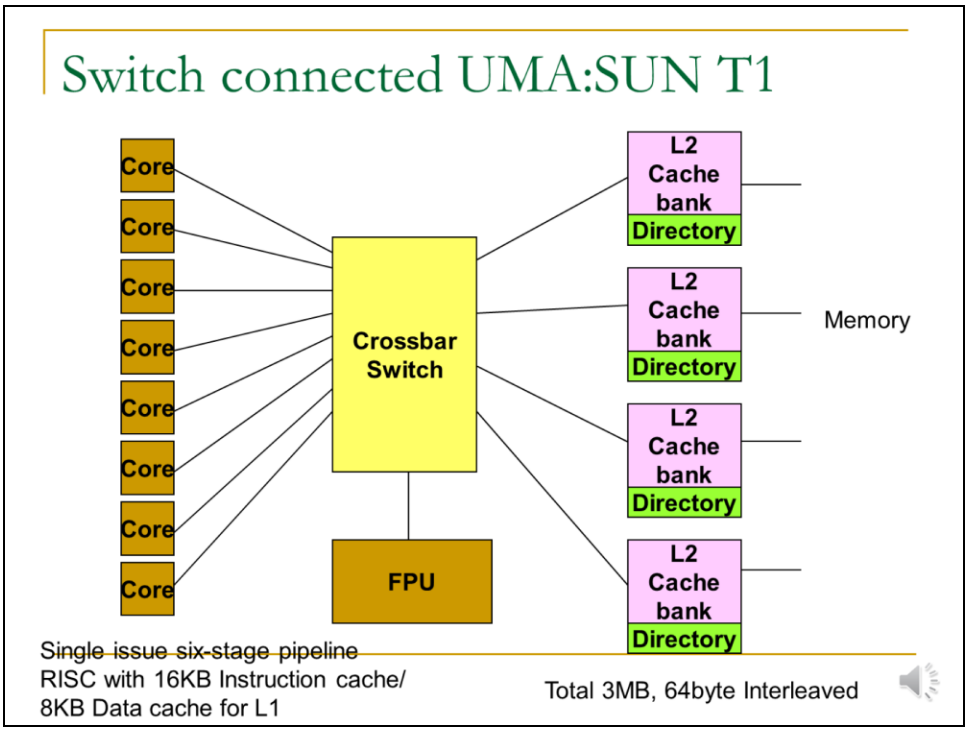
It is a straight forward extension of uniprocessors, and operating systems are also extensions of uniprocessors. Parallel programming is easy to be done for example, OpenMP. This will be introduced in this class. The problem of this style is that the size of the system is limited because of the congestion of the centralized shared memory. So, it is used in small systems. Usually, the total system is implemented on a single ship. Such UMA is called Multicore. Most of PCs, tablets and smartphones fall into UMA.



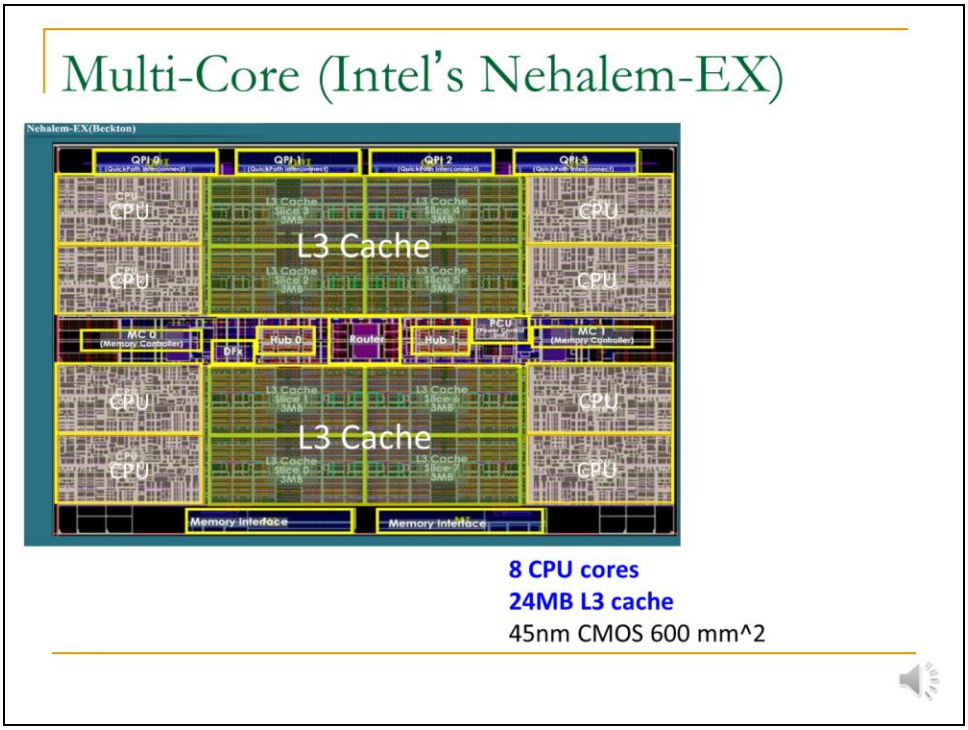
This is a common diagram of UMA. Of course, the shared bus is logical one. Inside the chip, it is built with a type of switch. The snoop cache is used in this type UMA. I will explain this technique later in this class.



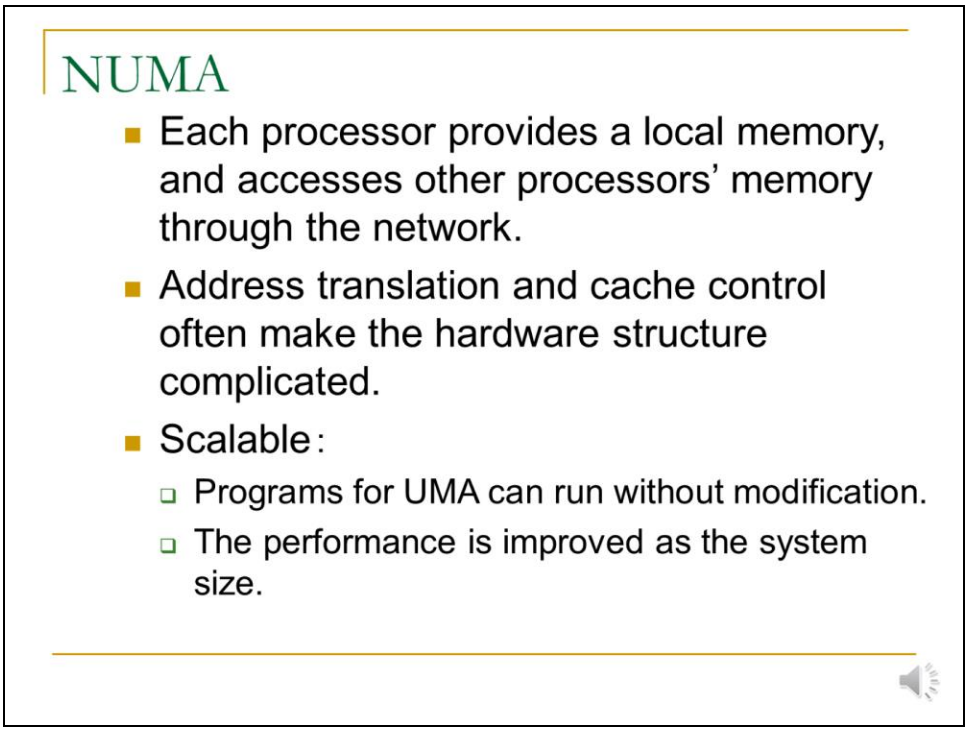
This shows an example of UMA for embedded usage. Four CPU share the L2 cache outside the chip.



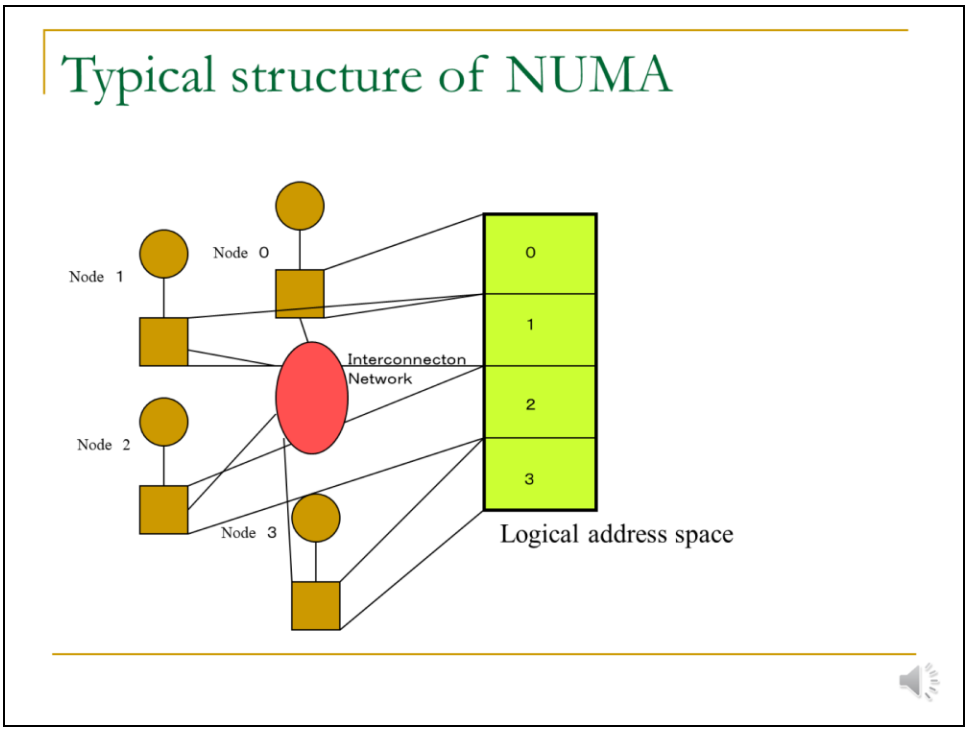
In order to increase the number of cores, a crossbar switch is used instead of the shared bus. However, this structure has the problem for cache coherence, if shared bus is not provided. I will explain on techniques for such machines.



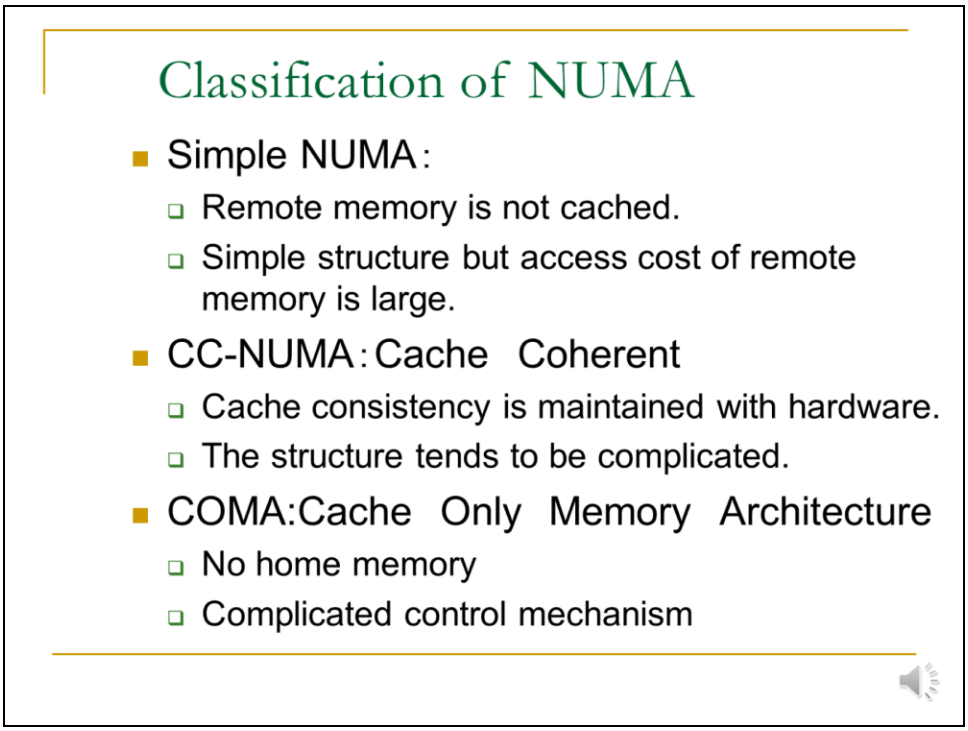
This photo is a UMA system with eight CPUs. Compared with GPU, you can see that the area of cache memory is larger than that for cores.



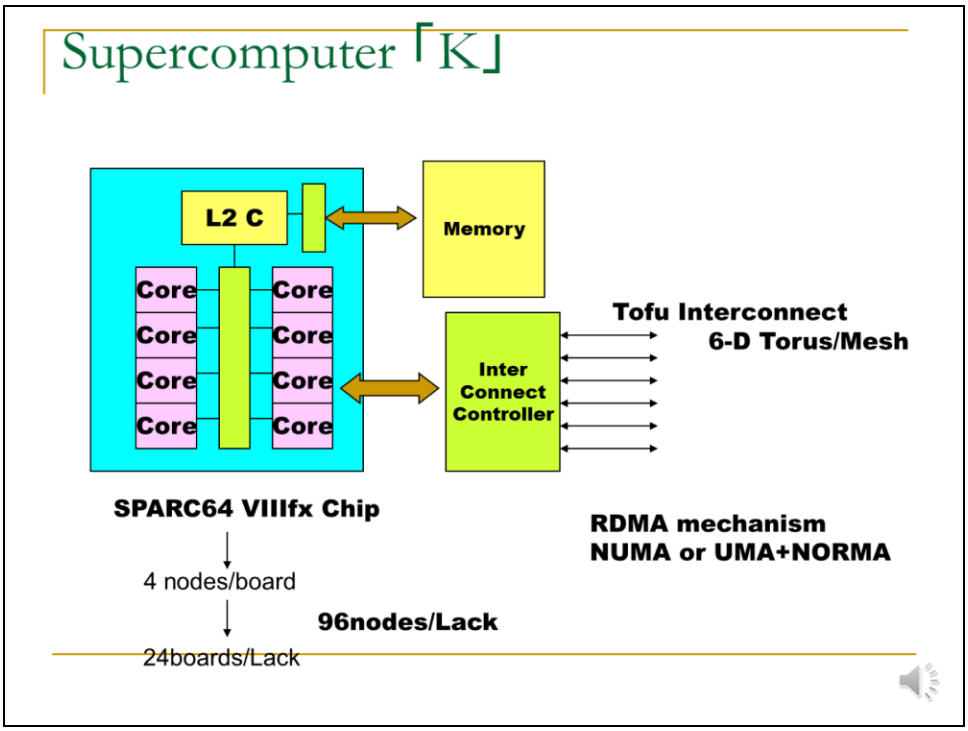
In NUMA, each processor provides a local memory, and accesses other processors' memory through the network. So, it is sometimes called a distributed shared memory machine. Although the address translation and cache control often make the hardware structure complicated, the benefit is that it is scalable.



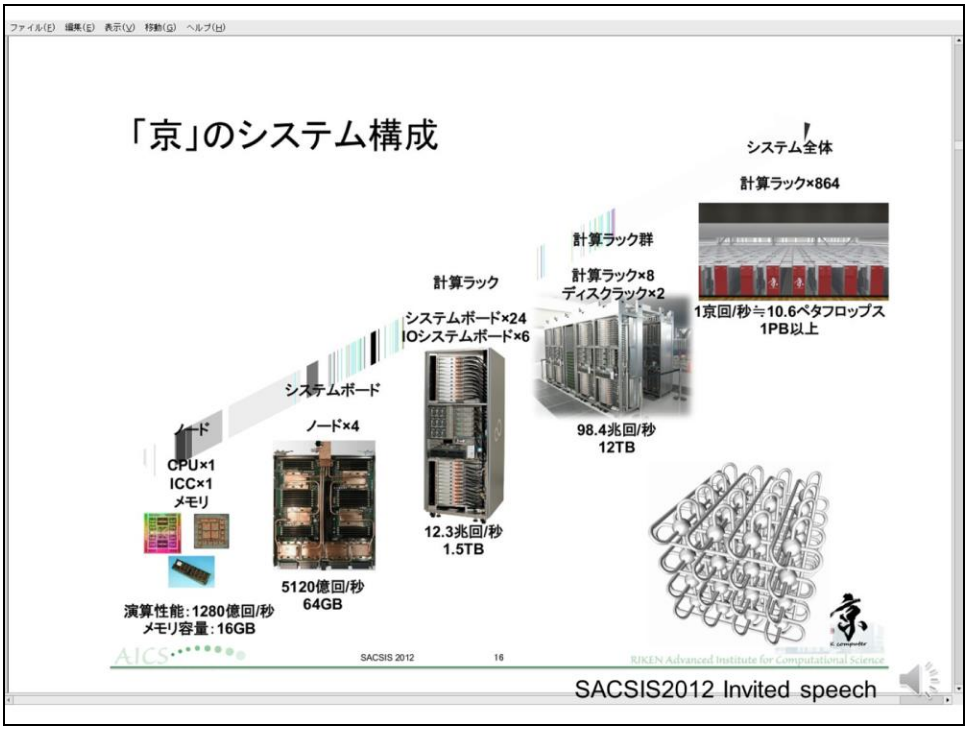
This diagram shows the typical structure of NUMA. All PEs have their own local memory and accessed directly. These memory modules are mapped on to the same address space, so each PE can access the memory attached to the different PEs just accessing the different address. But in this case, request and data are transferred through the network. Of course, the access speed of local and distant memory is different. The same program as UMA can work, but if there are a lot of distant access, the performance is degraded.



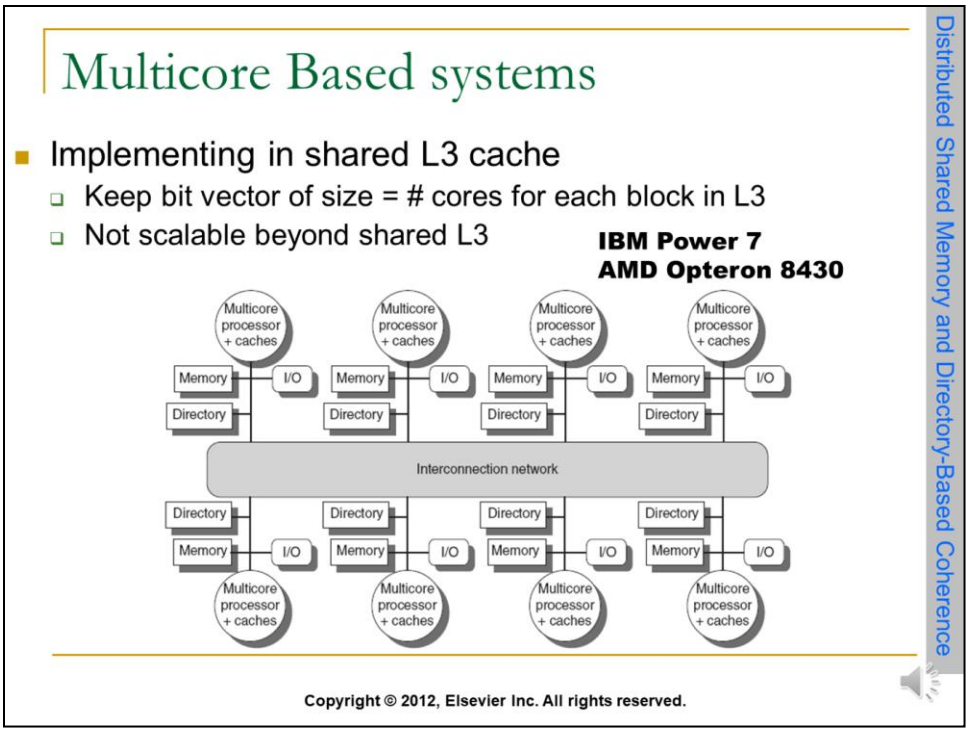
NUMA is classified into Simple NUMA, CC-NUMA and COMA.



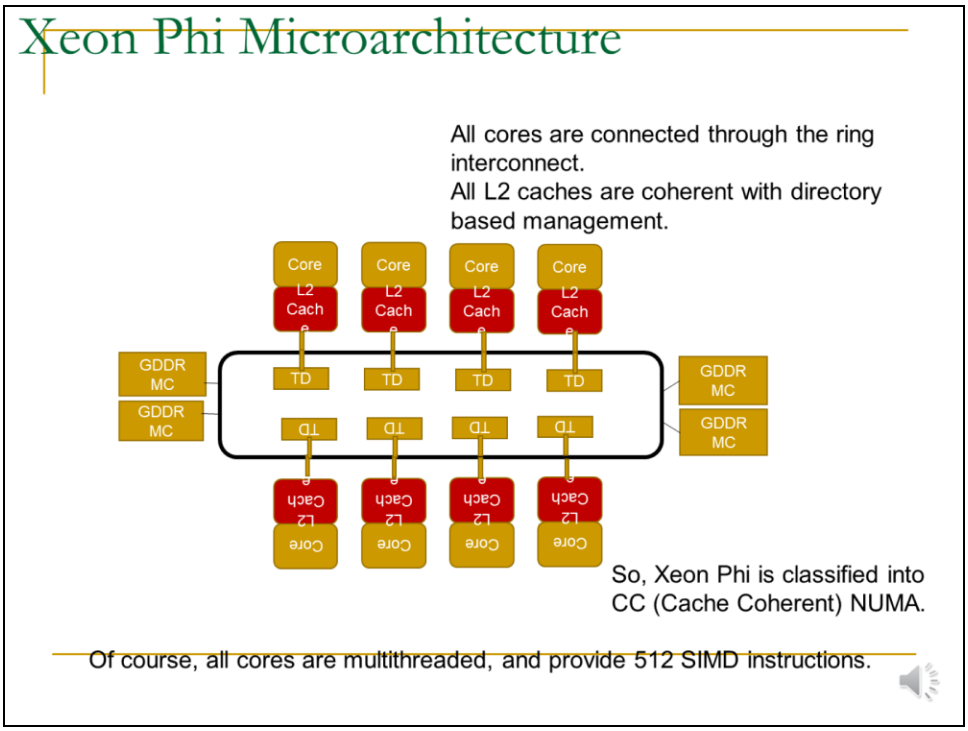
Simple NUMA is useful for supercomputers because of its scalability. Japanese flagship supercomputers use this style to be used by various kind of users. This shows the node structure of Supercomputer K. UMA chip is connected to others through the interconnect controller.



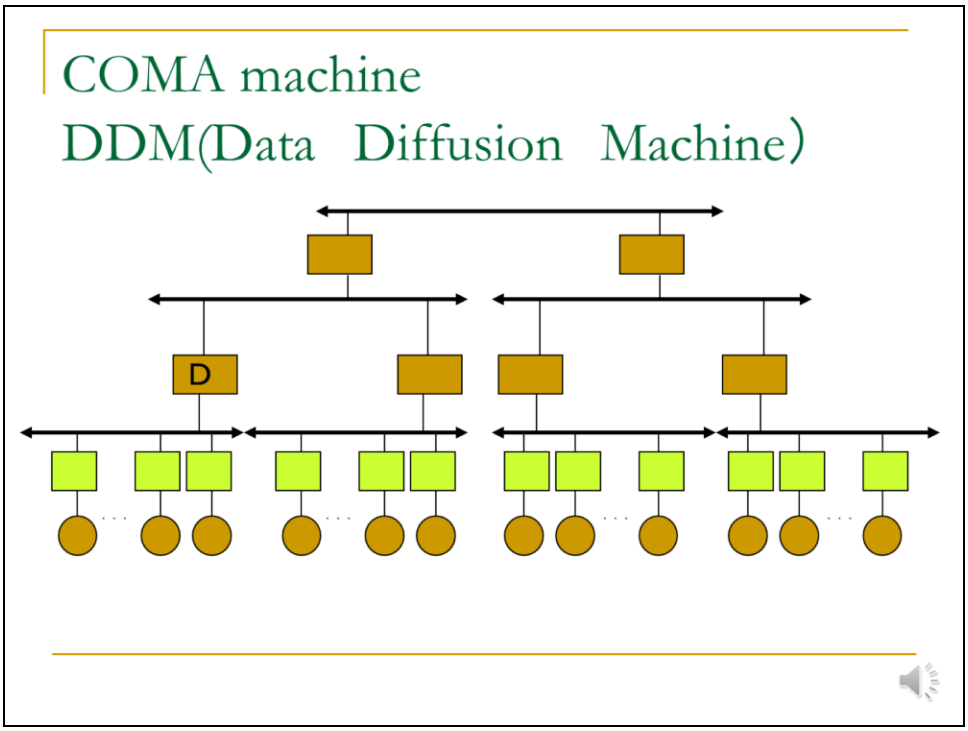
By connecting a lot of nodes, K is constructed. The next generation Fugaku will also take this style.



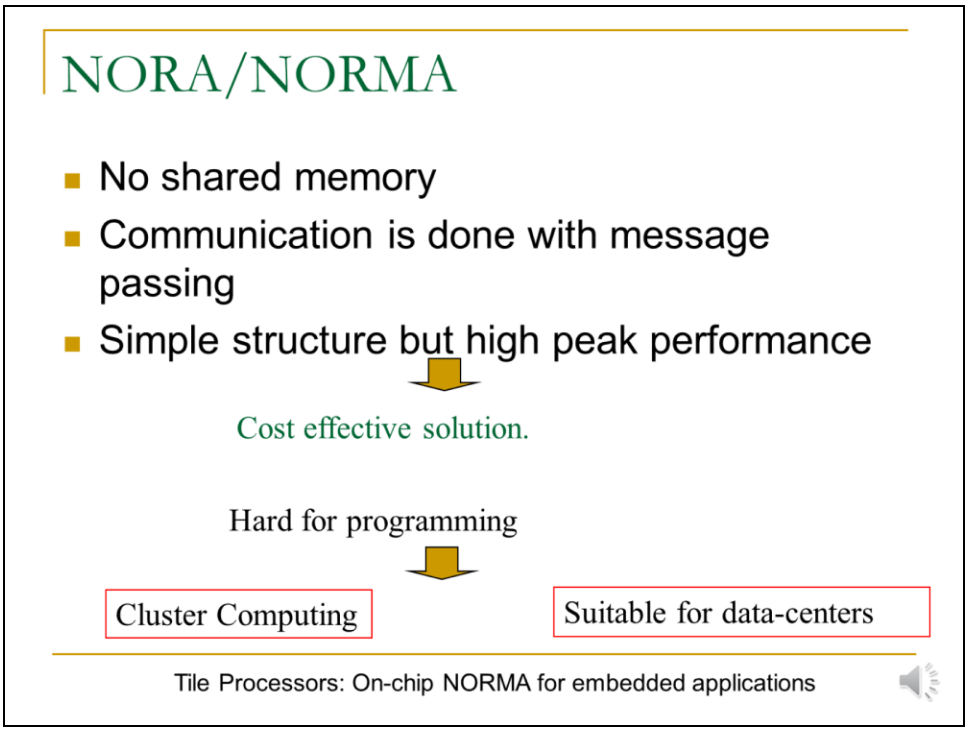
Cache coherent NUMA is a standard style for the server architecture. Also in this case, nodes are UMA processors. Each node has its home memory and directory mechanism which keep the cache coherence. This mechanism is a bit complicated compared with the snoop cache, but I will explain it in this class.



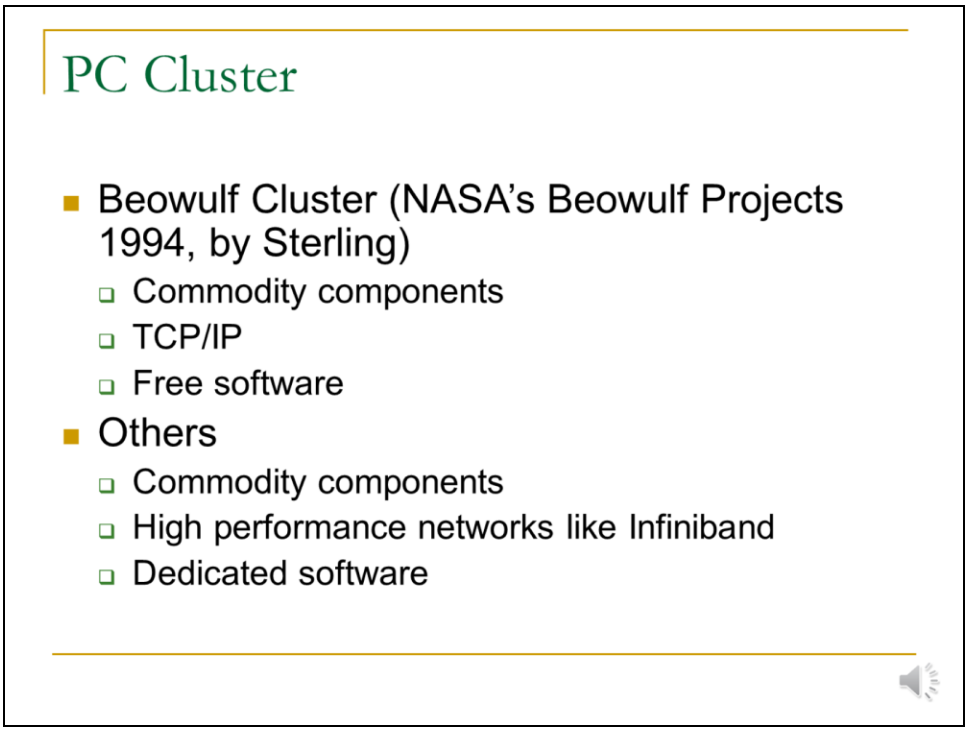
Intel's accelerator Xeon Phi also uses this CC-NUMA architecture. All cores are connected through ring interconnect.



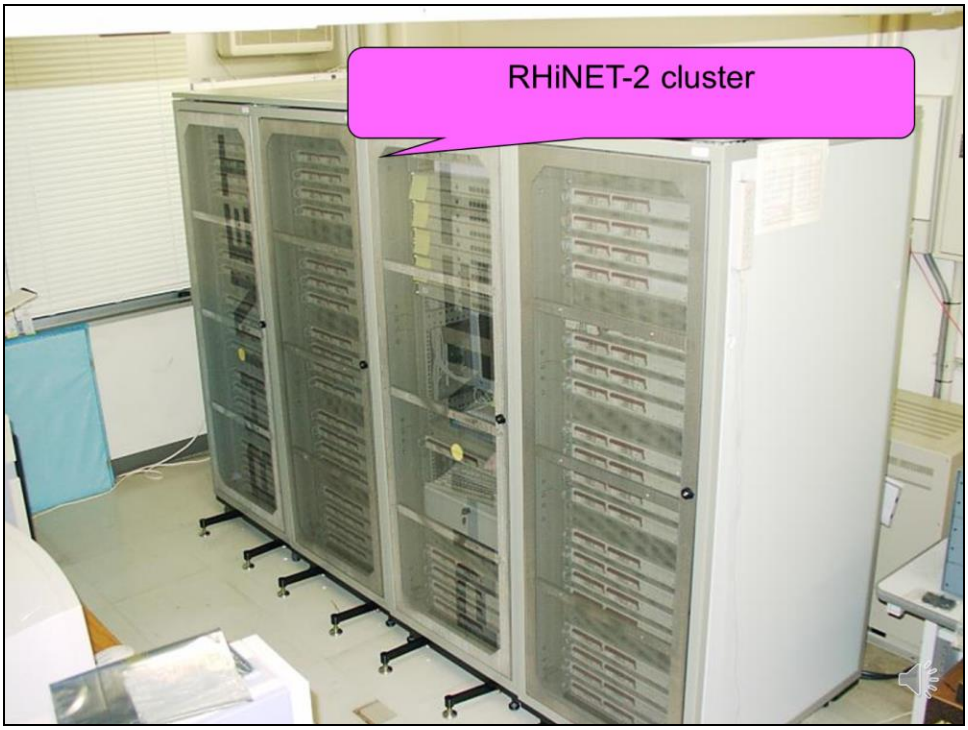
COMA is really interesting architecture. But since it is rarely used recently, I will skip this structure.



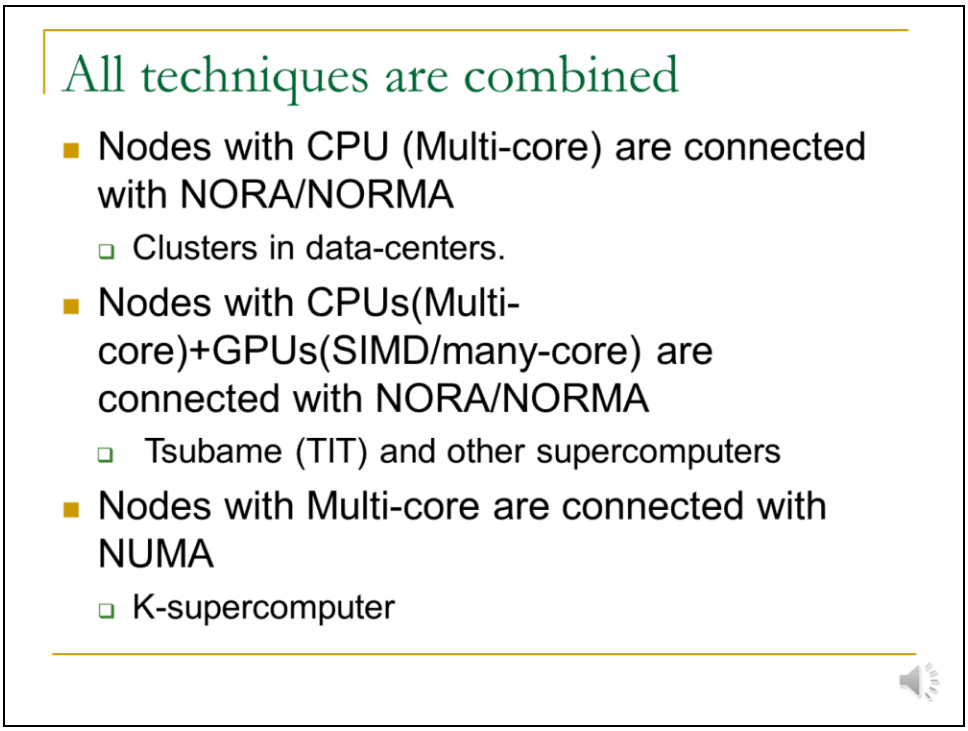
OK. let me explain the last style of MIMD, NORA or NORMA. This machine has no shared memory. Thus, communication must be done with message passing. This machine can be built just by connecting PCs with the network, and by connecting a lot of PCs, the high peak performance can be obtained. That is, it is a cost-effective solution. However, for executing a single job in parallel, we need to use message passing libraly like MPI. I will introduce it in this class. This architecture is sometimes called computer clusters. They are used in data centers, since they are mostly used for request level parallelism.



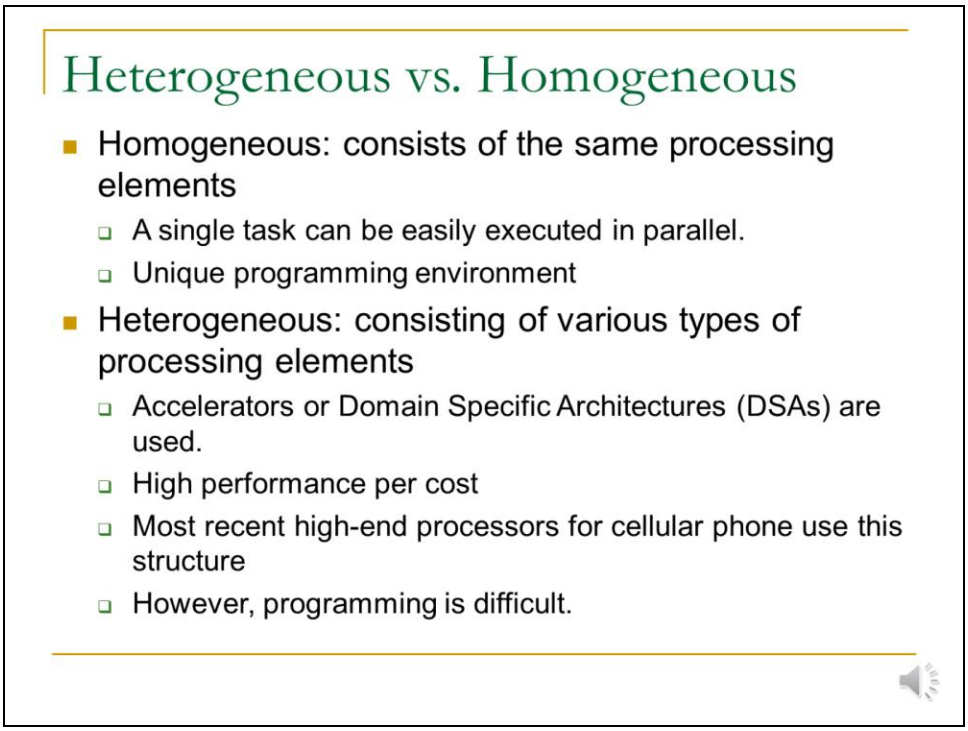
Beowulf Cluster by NASA's Beowulf project is an origin of PC cluster. They tried to build a powerful cluster by PC with commodity components, standard network using TCP/IP, and free software. However, some recent PC clusters use a dedicated interconnection network like Infiniband and dedicated sotware.



This photo is a cluster built by us in a national project.

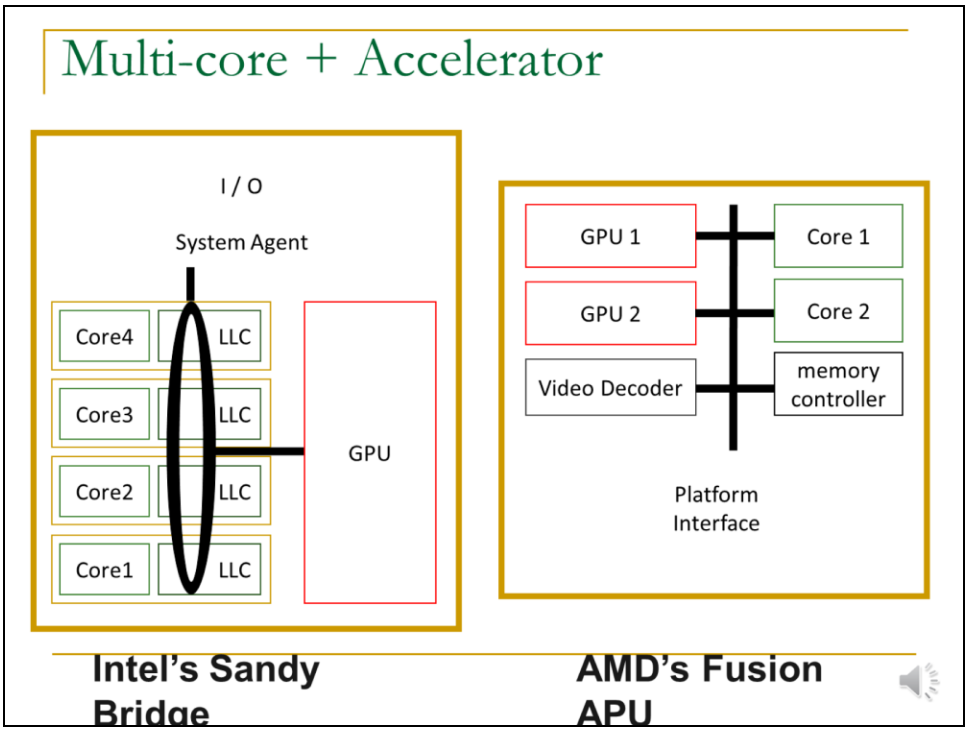


I explained various type of classification, but note that all techniques are combined recently. They are examples.



When a system is built with the same processing elements, it is called a heterogeneous system. It has a benefits like them.

Heterogenous system uses accelerators or domain specific architectures for a specific job. Recently, heterogenous systems are increasing in supercomputing or data centers.



Recently, some multi-core systems embed GPU and multi-core in the same chip.

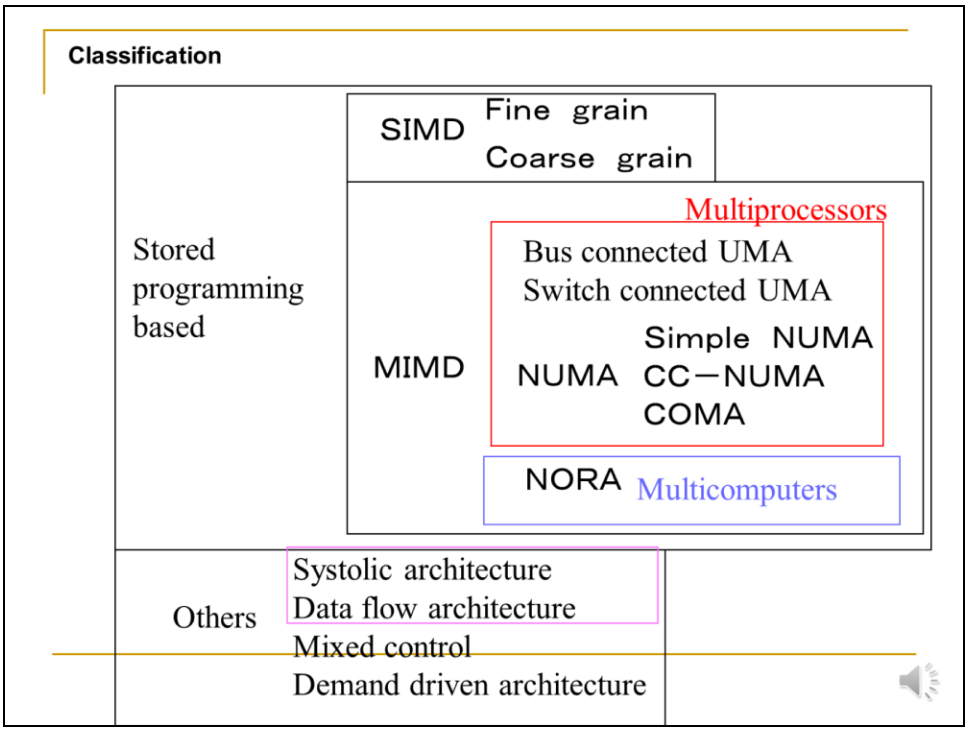
Domain Specific Architectures (DSAs)

Intel's Neural Compute Stick 2

Google's edge TPU
(Tensor Processing Unit)

Parallel Architectures for Specific Application (AI/DNN) are used.

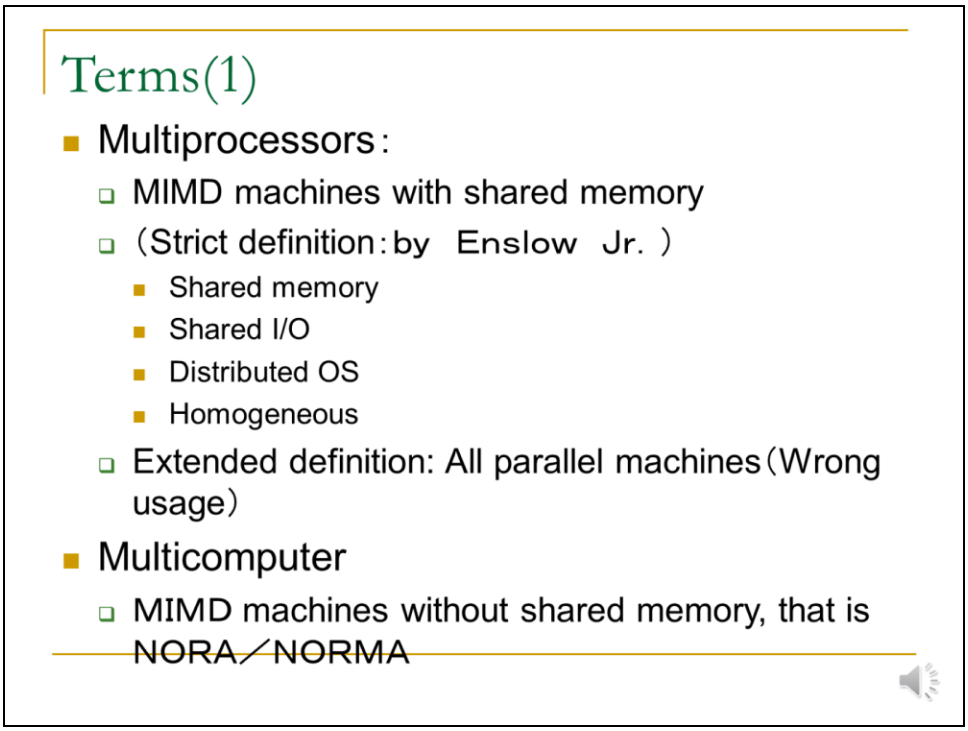
The domain specific architectures are active especially in the field of deep learning of the AI application. For example, in the edge TPU by Google uses a systolic algorithm, a type of parallel hardware algorithm.



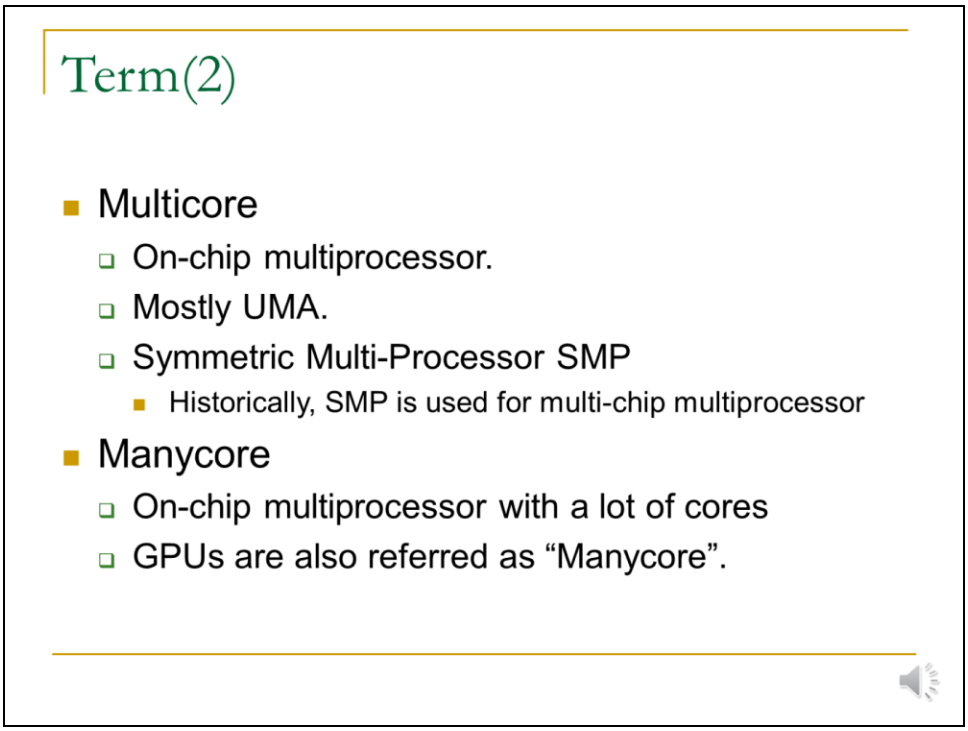
This map is the classification of parallel architectures introduced in this class.

glossary 3

- Flynn's Classification: Flynn(Stanford大の教授)が論文中に用いた分類、内容は本文を参照のこと
- Coarse grain:粗粒度、この場合はプロセッシングエレメントが浮動小数演算が可能な程度大きいこと。反対がFine grain(細粒度)で、数ビットの演算しかできないもの
- Illiac-IV, BSP, GF-11, Connection Machine CM-2, MP-2などはマシン名。SIMDの往年の名機
- Synchronization:同期、Shared Memory:共有メモリ、この辺は後の授業で詳細を解説する
- Message passing:メッセージ交換。共有メモリを使わずにデータを直接交換する方法
- Embedded System:組み込みシステム
- Homogeneous:等質な Heterogeneous:性質の異なったものから成る
- Coherent Cache:内容の一貫性が保障されたキャッシュ、Cache Consistencyは内容の一貫性、これも後の授業で解説する
- Commodity Component: 標準部品、価格が安く入手が容易
- Power 5, Origin2000, Cray XD-1,AP1000,NCUBE などもマシン名。The earth simulatorは地球シミュレータ,IBM BlueGene/Lは現在のところ最速



Finally, I will introduce some terms in this area. Some are old fashioned and not used recently.



Multicore and manycore are not technical words, but recently, they are popularly used.

Exercise 1

- AIST(The National Institute of Advanced Industrial Science and Technology) developed a supercomputer for AI application called ABCI.
- It won the 8th place of the TOP-500 supercomputer ranking (2019.11).
- How do you classify ABCI ?
  - Check the website and describe your opinion.
- If you take this class, send the answer with your name and student number to hunga4125@gmail.com
- You can use either Japanese or English.
- The deadline is 2 weeks later.