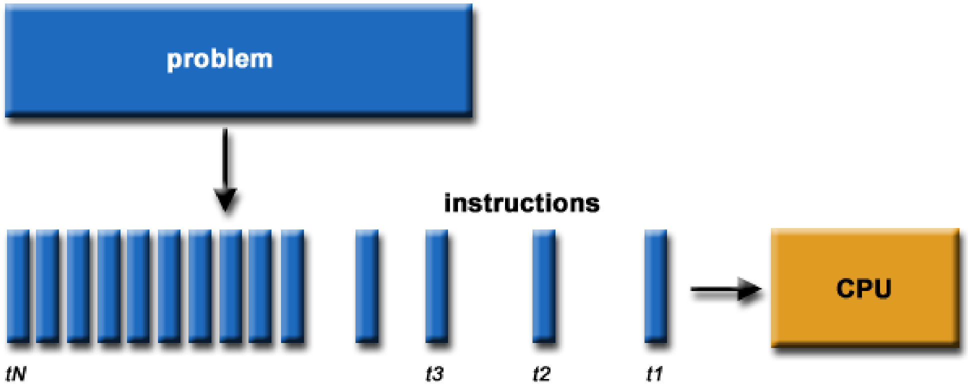
Principles of Computer Architecture Processors

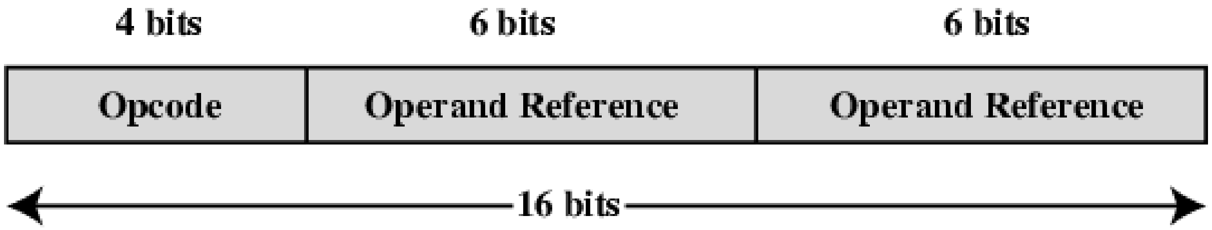
Lecture 2

Instruction-Level Parallelism,

Threads, Cores, SMP

Instruction





Instruction is an operation that is executed by processor.

Processor architecture

- 1975 - 1986 bit level parallelism (word 4, 8, 16, 32, 64, 128 bits).
- **A word is simply a fixed-sized group of bits that are handled together by the machine*.***
- 1980 - 1998 m. instruction level parallelism.
  - Pipeline,
  - Superscalar, Superpipelined processors
  - VLIW, RISC
- 1991 - ... Thread level parallelism
- 2004 - ... Core level parallelism

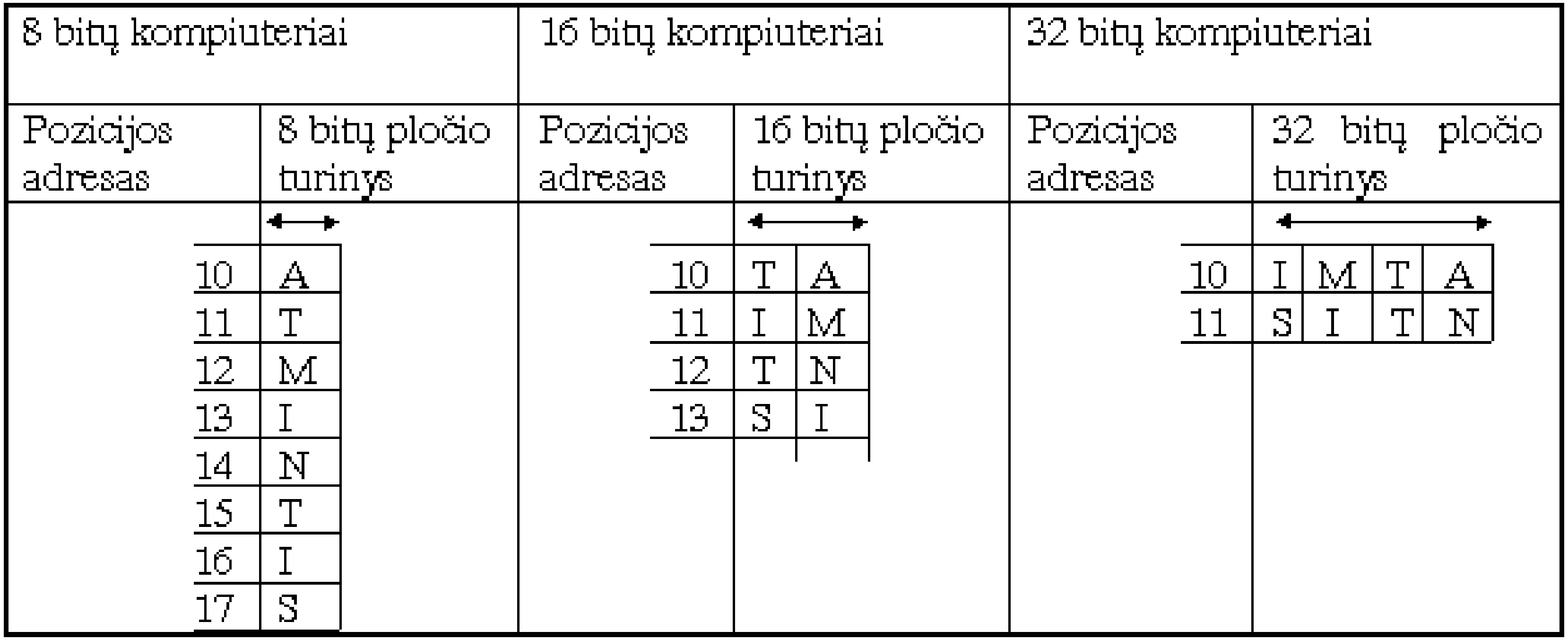
Word

In computing, "word" is a term for the natural unit of data used by a particular computer design.

A word is simply a fixed-sized group of bits that are handled together by the machine. The number of bits in a word (the word size or word length) is an important characteristic of a computer architecture.

The size of a word influences many aspects of a computer's structure and operation. The majority of the registers in the computer are usually word-sized. The typical numeric value manipulated by the computer is probably word sized. The amount of data transferred between the processing part of the computer and the memory system is most often a word. An address used to designate a location in memory often fits in a word.

Word



8 bitų kompiuteriai		16 bitų kompiuteriai		32 bitų kompiuteriai	
Pozicijos adresas	8 bitų pločio turinys	Pozicijos adresas	16 bitų pločio turinys	Pozicijos adresas	32 bitų pločio turinys
10	A	10	T A	10	I M T A
11	T	11	I M	11	S I T N
12	M	12	T N		
13	I	13	S I		
14	N				
15	T				
16	I				
17	S				

Instruction level parallelism

Instruction-level parallelism (ILP) refers on the degree to which the instructions of a program can be executed in parallel.

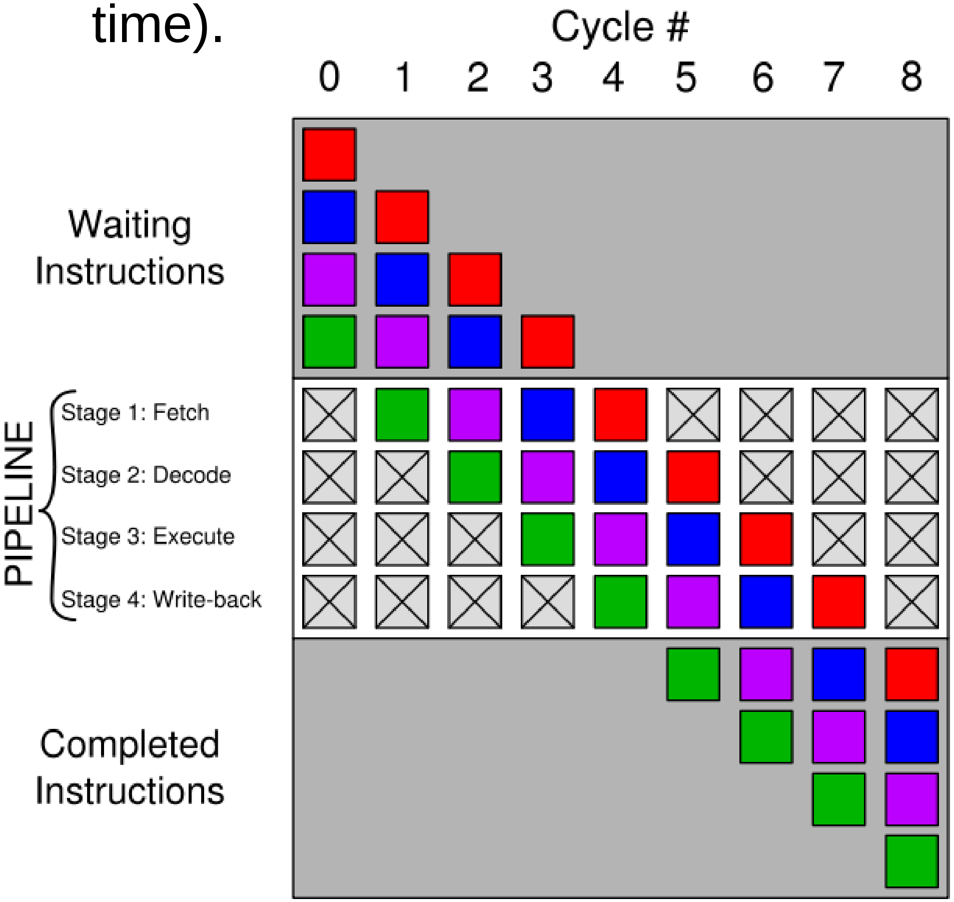
Example:

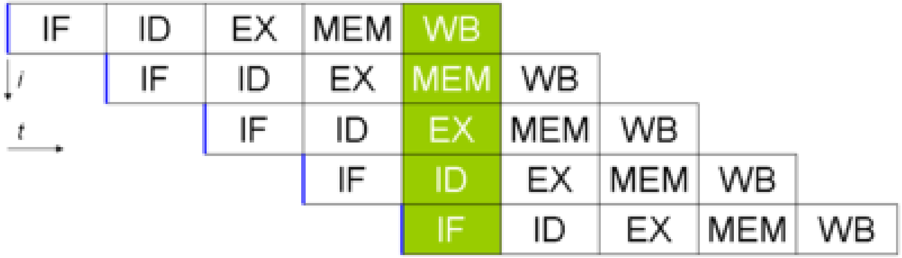
1. e = a + b
2. f = c + d
3. g = e * f

$$\frac{\text{time}}{\text{program}} = \frac{\text{time}}{\text{cycle}} \text{ x } \frac{\text{cycles}}{\text{instruction}} \text{ x } \frac{\text{instructions}}{\text{program}}$$

Instruction pipeline

An **instruction pipeline** is a technique used to increase their instruction throughput (the number of instructions that can be executed in a unit of time).





IF	ID	EX	MEM	WB				
	IF	ID	EX	MEM	WB			
		IF	ID	EX	MEM	WB		
			IF	ID	EX	MEM	WB	
				IF	ID	EX	MEM	WB

Superscalar processor

A **superscalar processor** is one in which multiple independent instruction pipelines are used.

Each pipeline consists of multiple stages, so that each pipeline can handle multiple instructions at a time.

Multiple pipelines introduce a new level of parallelism, enabling multiple streams of instructions to be processed at a time.

Superscalar processor

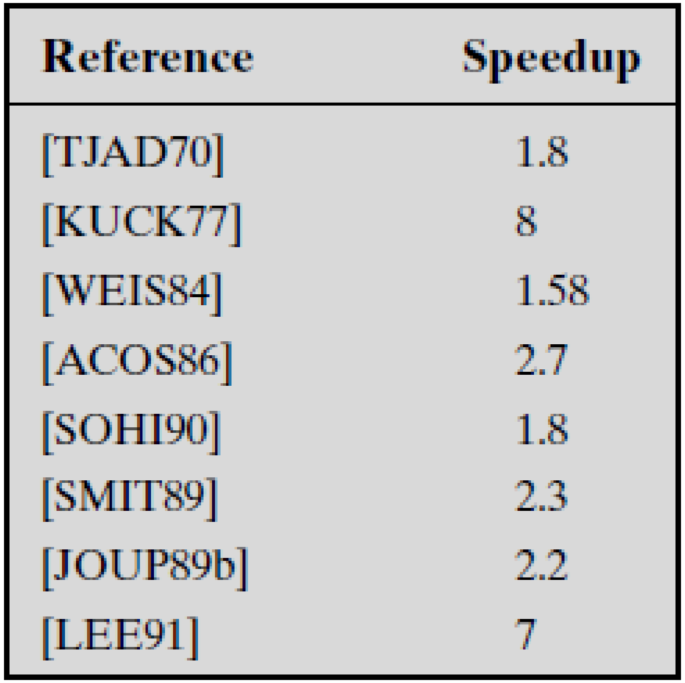
A **superscalar processor** typically fetches multiple instructions at a time and then attempts to find nearby instructions that are independent of one another and can therefore be executed in parallel.

If the input to one instruction depends on the output of a preceding instruction, then the latter instruction cannot complete execution at the same time or before the former instruction.

Once such dependencies have been identified, the processor may issue and complete instructions in an order that differs from that of the original machine code.

Speedup

Speedups of Superscalar-Like Machines



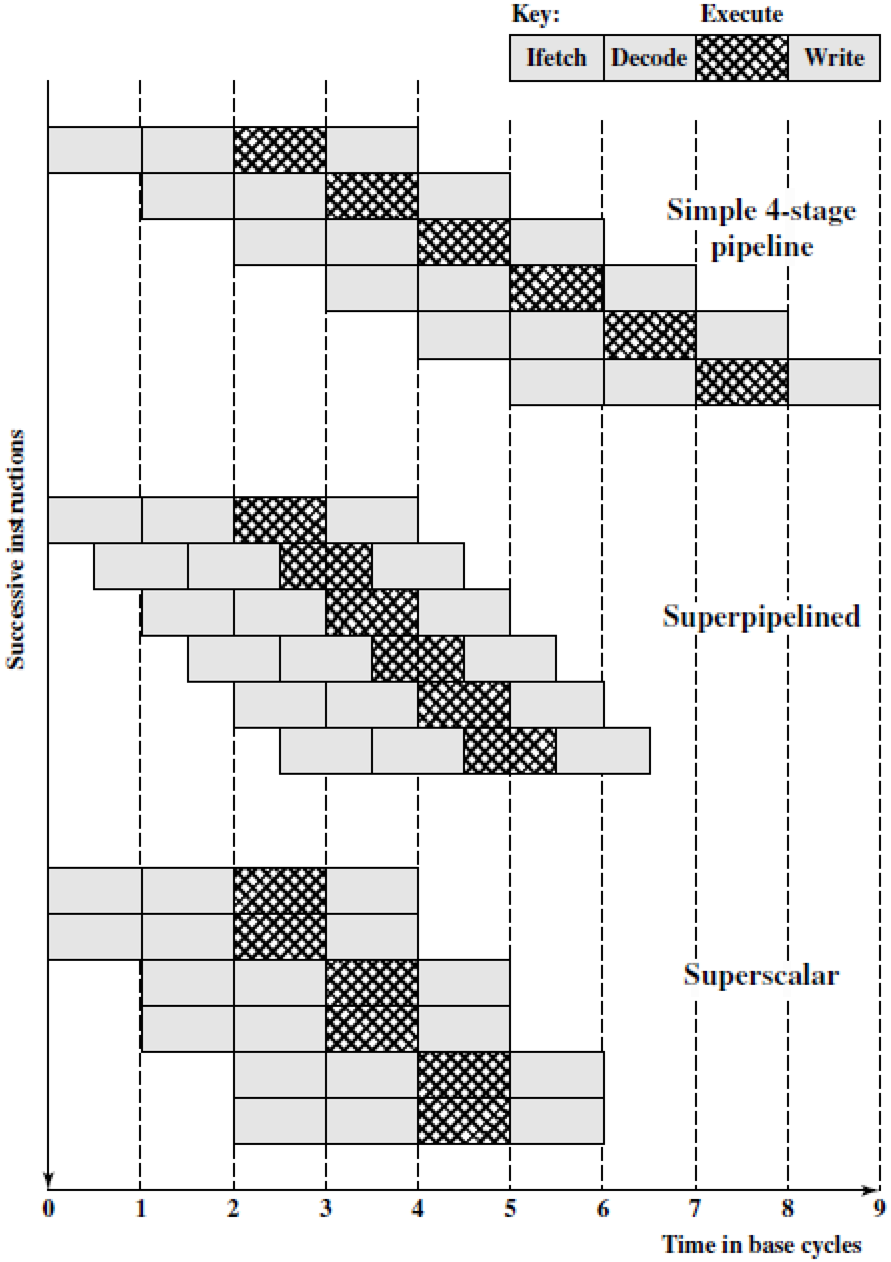
Reference	Speedup
[TJAD70]	1.8
[KUCK77]	8
[WEIS84]	1.58
[ACOS86]	2.7
[SOHI90]	1.8
[SMIT89]	2.3
[JOUP89b]	2.2
[LEE91]	7

Superpipelined

Superpipelining (term first coined in 1988) exploits the fact that many pipeline stages perform tasks that require less than half a clock cycle.

Thus, a doubled internal clock speed allows the performance of two tasks in one external clock cycle.

Superscalar vs. Superpipelined



Limitations

Fundamental limitations to parallelism with which the system must cope. List of five limitations:

- True data dependency
- Procedural dependency
- Resource conflicts
- Output dependency
- Antidependency

Vector type processor

Scalar processor

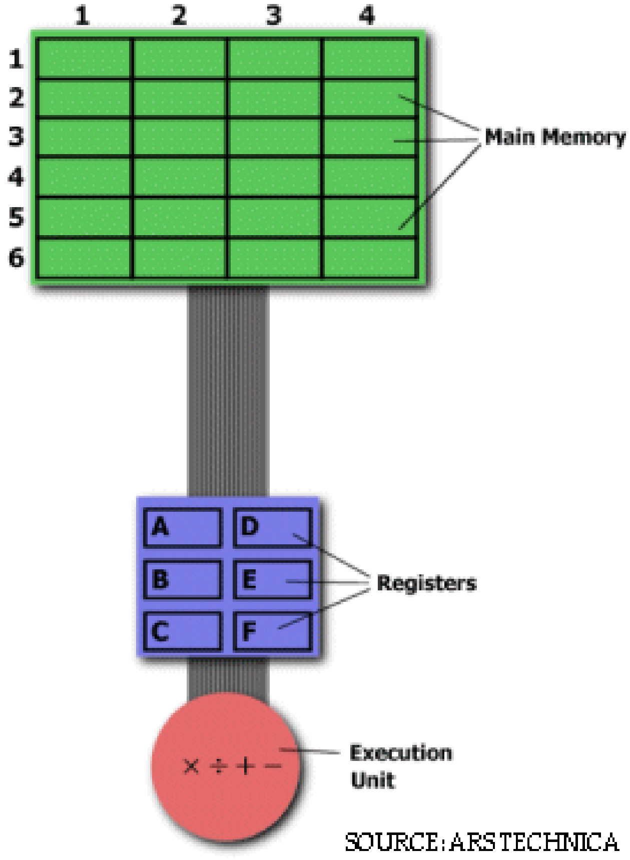
read the next instruction and decode it
get this number
get that number
add them
put the result here
read the next instruction and decode it
get this number
get that number
add them
put the result there

Vector processor

read instruction and decode it
get these 10 numbers
get those 10 numbers add them
put the results here

CISC

CISC - *Complex Instruction Set Computing*
RISC - *Reduced Instruction Set Computing*



SOURCE:ARSTECHNICA

CISC

The primary goal of CISC architecture is to complete a task in as few lines of assembly as possible.

This is achieved by building processor hardware that is capable of understanding and executing a series of operations.

For this particular task, a CISC processor would come prepared with a specific instruction (example "MULT"). When executed, this instruction loads the two values into separate registers, multiplies the operands in the execution unit, and then stores the product in the appropriate register. Thus, the entire task of multiplying two numbers can be completed with one instruction:

- MULT 2:3, 5:2

CISC

MULT is what is known as a "complex instruction." It operates directly on the computer's memory banks and does not require the programmer to explicitly call any loading or storing functions. It closely resembles a command in a higher level language.

For instance, if we let "a" represent the value of 2:3 and "b" represent the value of 5:2, then this command is identical to the C statement "a = a * b."

One of the primary advantages of this system is that the compiler has to do very little work to translate a high-level language statement into assembly. Because the length of the code is relatively short, very little RAM is required to store instructions. The emphasis is put on building complex instructions directly into the hardware.

RISC

RISC processors only use simple instructions that can be executed within one clock cycle.

Thus, the "MULT" command described above could be divided into three separate commands:

```
LOAD A, 2:3
LOAD B, 5:2
 PROD A, B
 STORE 2:3, A
```

Because there are more lines of code, more RAM is needed to store the assembly level instructions. The compiler must also perform more work to convert a high-level language. Processor is more simple.

RISC

RISC processors only use simple instructions that can be executed within one clock cycle.

Thus, the "MULT" command described above could be divided into three separate commands: "LOAD," which moves data from the memory bank to a register, "PROD," which finds the product of two operands located within the registers, and "STORE," which moves data from a register to the memory banks.

In order to perform the exact series of steps described in the CISC approach, a programmer would need to code four lines of assembly.

RISC advantages

- Processors only use simple instructions that can be executed within one clock cycle
- Register-Register type commands (*load-store* ISA)
- Simple structure of the addresses
- Simple formats of the commands
- Complex compilers
- Effective use of pipeline

Type of Paralellism

- Bit level (8, 16, 32, 64 bits words)
- Instruction Level Parallelism
- *Multi-Threading*
- *Multi-core*
- *Multi-processing*

Threads

Instructions are simultaneously issued from multiple threads to the execution units of a superscalar processor. This combines the wide superscalar instruction issue capability with the use of multiple thread contexts.

Threads are controlled by system calls.
open, **read**, **write**, **close**, **wait**, **exec**, **fork**, **exit**, **kill**… etc.

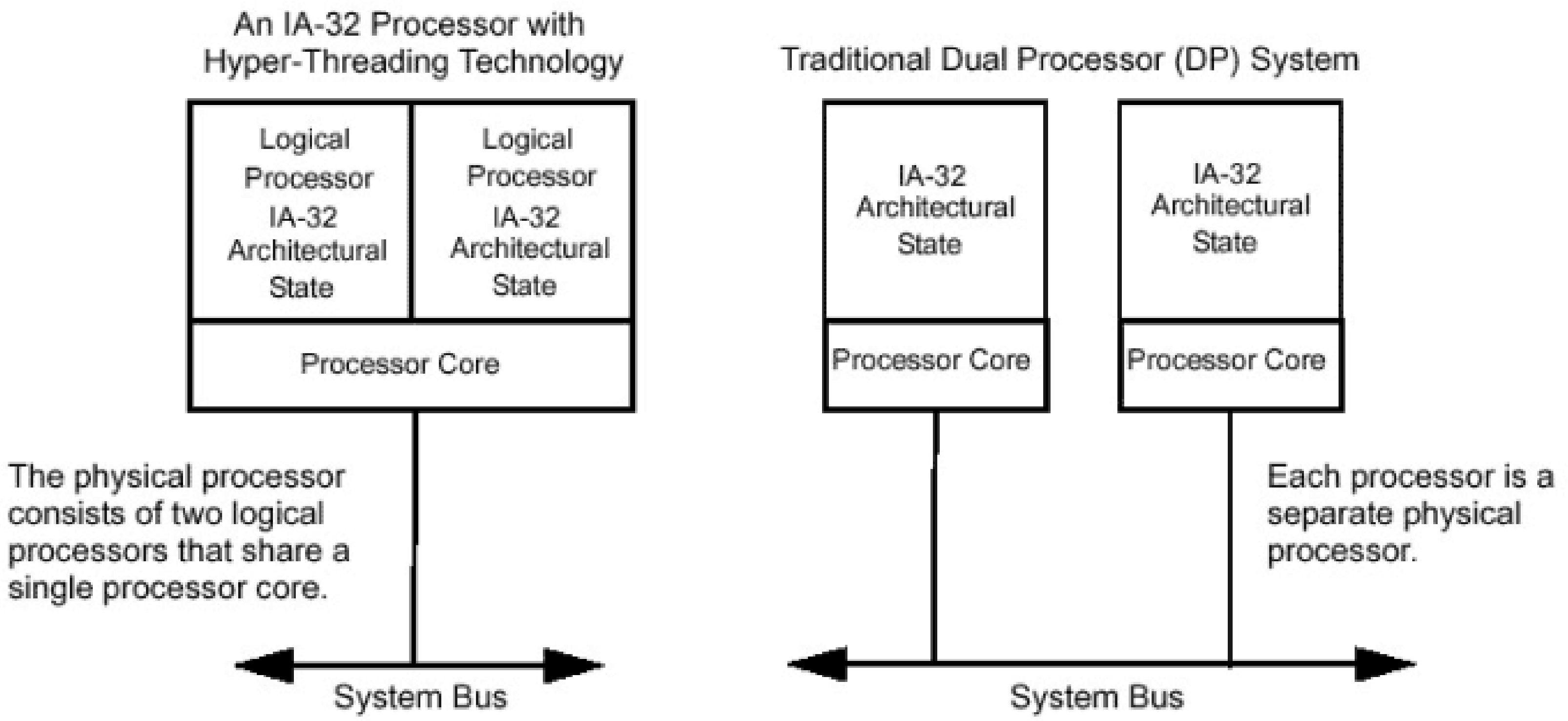
Linux OS has 319 different system calls.

Multi threaded

- When more than one processor are implemented on a single chip, the configuration is referred to as **chip multiprocessing**.

- A related design scheme is to replicate some of the components of a single processor so that the processor can execute multiple threads concurrently; this is known as a **Multi threaded processor.**

Hyper-Threading (Intel)


An IA-32 Processor with Hyper-Threading Technology

Logical Processor IA-32 Architectural State

Logical Processor IA-32 Architectural State

Processor Core

The physical processor consists of two logical processors that share a single processor core.

System Bus

Traditional Dual Processor (DP) System

IA-32 Architectural State

Processor Core

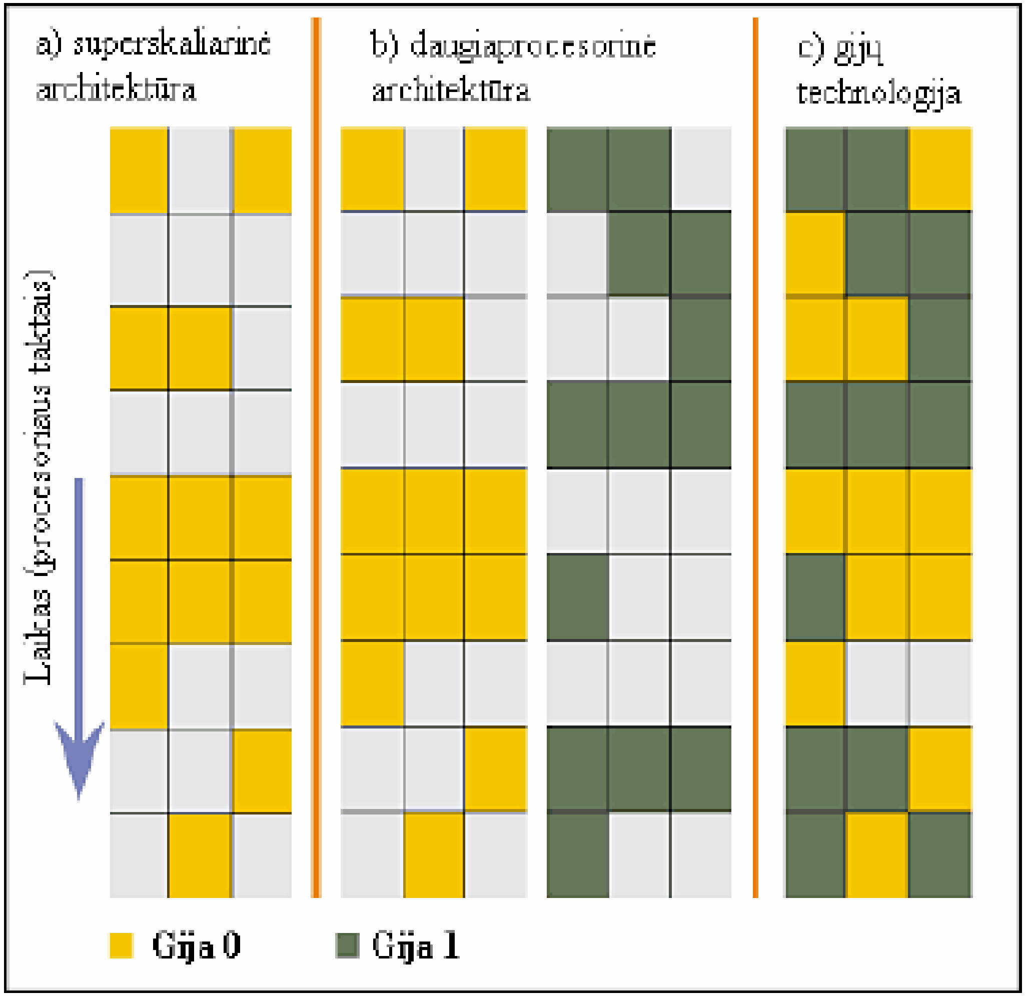
IA-32 Architectural State

Processor Core

Each processor is a separate physical processor.

System Bus

Hyper-Threading


a) superskaliarinė architektūra
b) daugiaprocesorinė architektūra
c) gijų technologija
Laikas (procesoriaus taktais)
Gija 0
Gija 1

SMP

- A traditional way to increase system performance is to use multiple processors that can execute in parallel to support a given workload.

- The two most common multiple-processor organizations are **symmetric multiprocessors** (SMPs) and clusters.

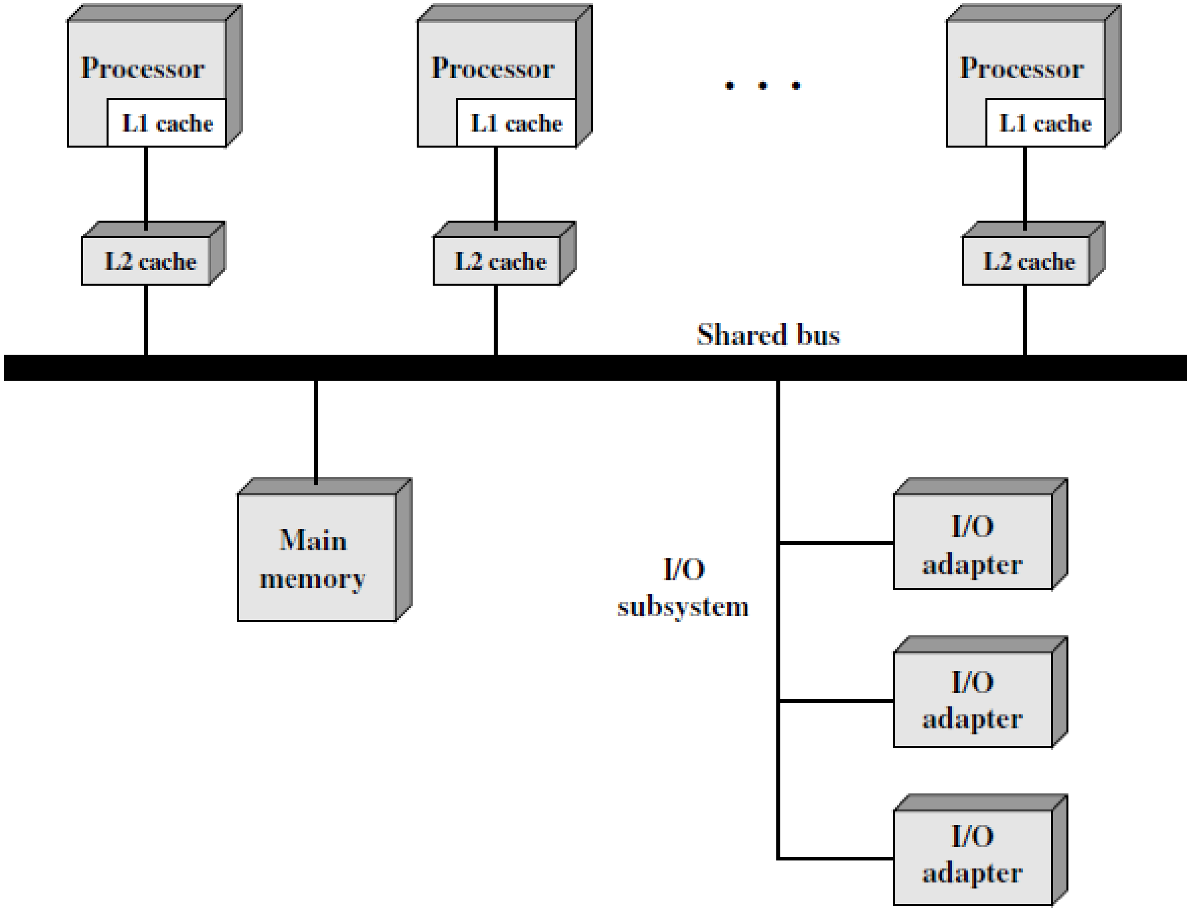
SMP

An SMP consists of multiple similar processors within the same computer, interconnected by a bus or some sort of switching arrangement.

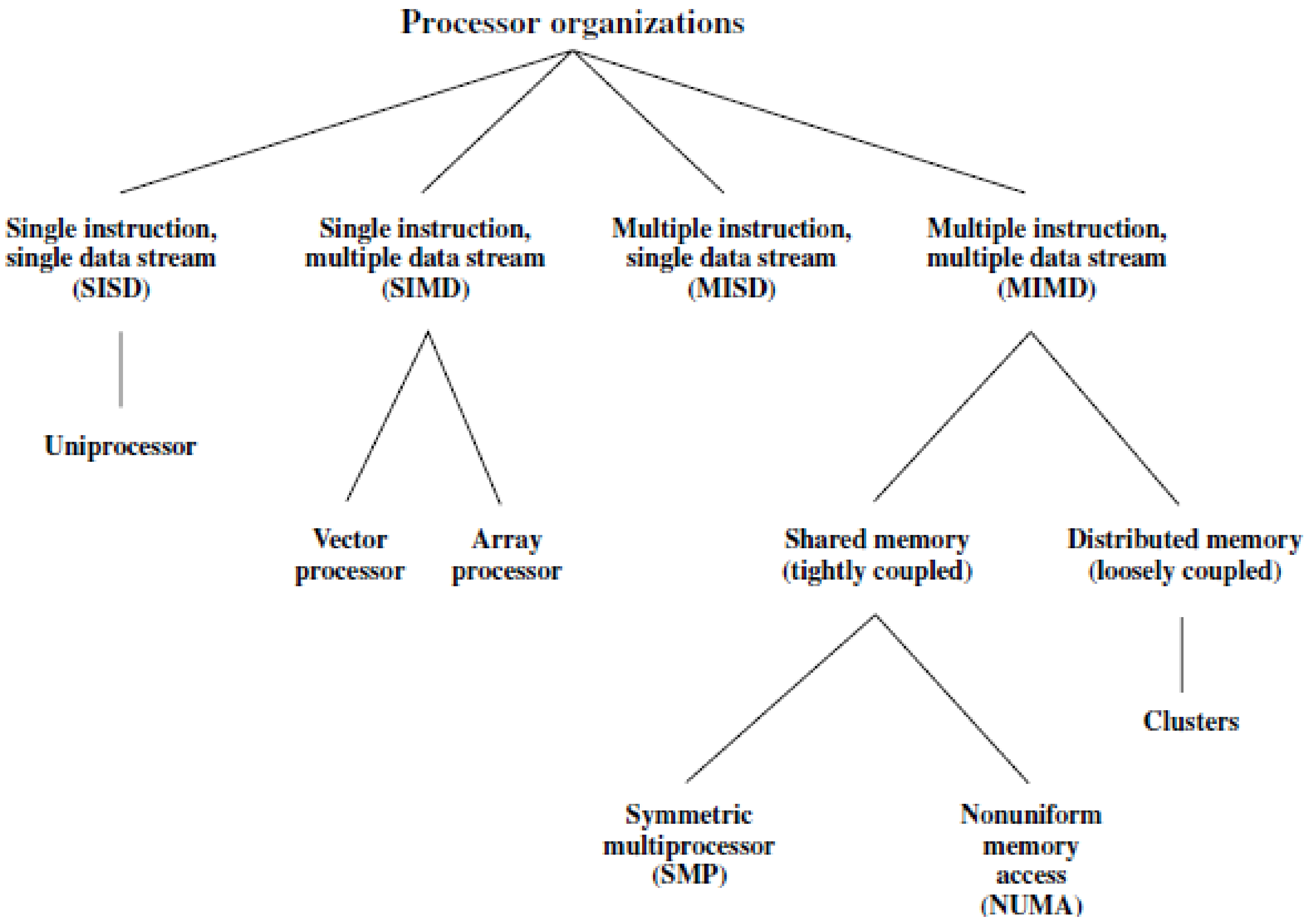
The most critical problem to address in an SMP is that of cache coherence. Each processor has its own cache and so it is possible for a given line of data to be present in more than one cache. If such a line is altered in one cache, then both main memory and the other cache have an invalid version of that line.

Cache coherence protocols are designed to cope with this problem.

SMP organization

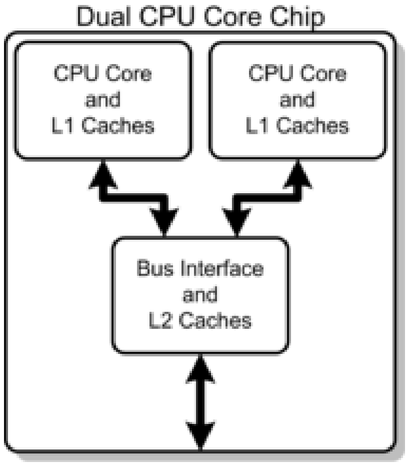


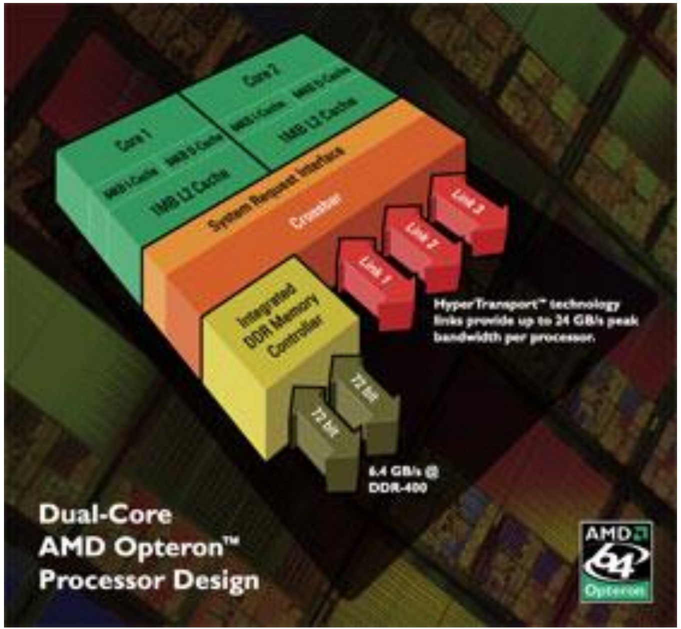
Taxonomy of Parallel Processor Architectures


Processor organizations
Single instruction, single data stream (SISD)
Single instruction, multiple data stream (SIMD)
Multiple instruction, single data stream (MISD)
Multiple instruction, multiple data stream (MIMD)
Uniprocessor
Vector processor
Array processor
Shared memory (tightly coupled)
Distributed memory (loosely coupled)
Clusters
Symmetric multiprocessor (SMP)
Nonuniform memory access (NUMA)

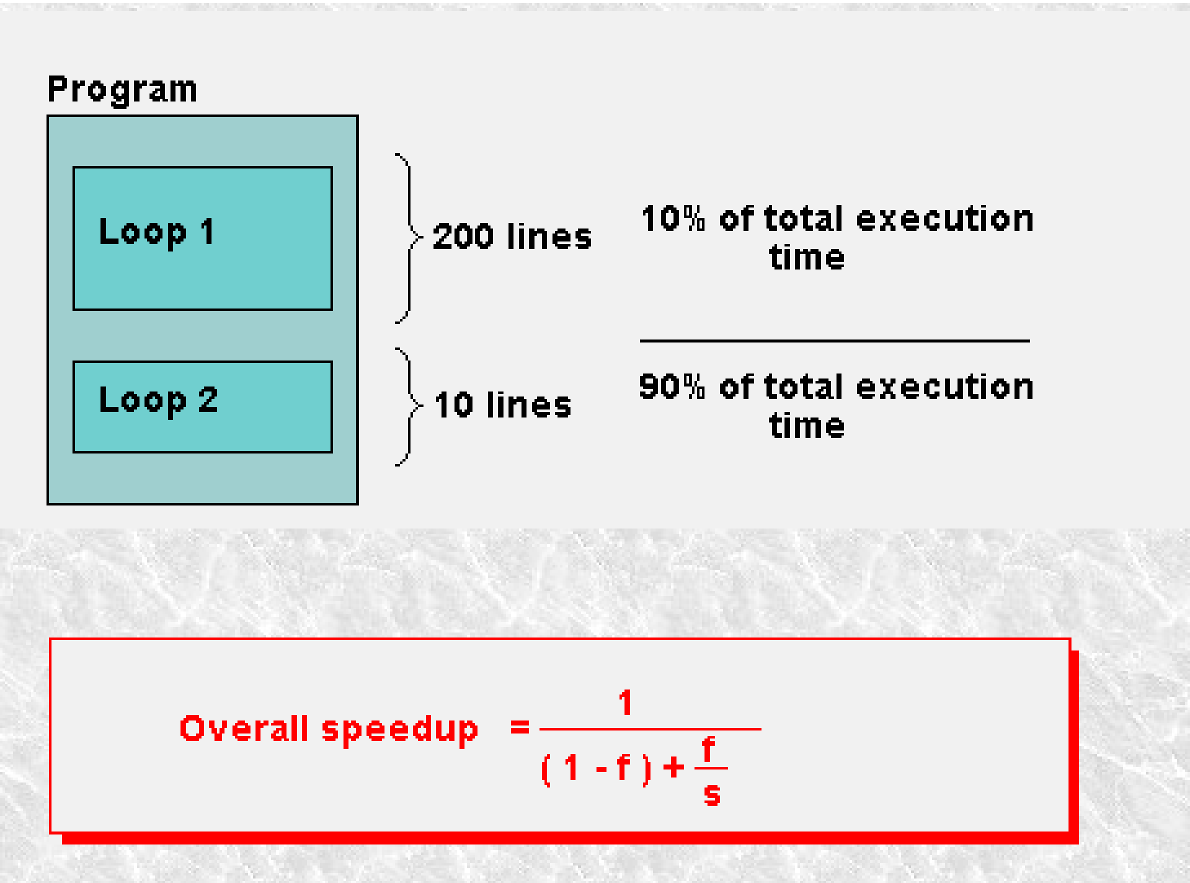
Multi-Core

A multi-core microprocessor is one which combines two or more independent processors into a single package, often a single integrated circuit (IC). A dual-core device contains only two independent microprocessors. In general, multi-core microprocessors allow a computing device to exhibit some form of thread-level parallelism (TLP) without including multiple microprocessors in separate physical packages. This form of TLP is often known as chip-level multiprocessing, or CMP. There is some discrepancy in the semantics by which the terms "multi-core" and "dual-core" are defined.





Amdahl’s law



Program

Loop 1

Loop 2

200 lines

10 lines

10% of total execution time

90% of total execution time

$$\text{Overall speedup} = \frac{1}{(1 - f) + \frac{f}{s}}$$

f – proportion of a program that can be made parallel

S – number of processors

Amdahl's law

What fraction of the program should be executed in parallel in order to obtain speedup equal to 80 when 100 processors are used?

$$\frac{1}{(1-f)+f/100}=80$$

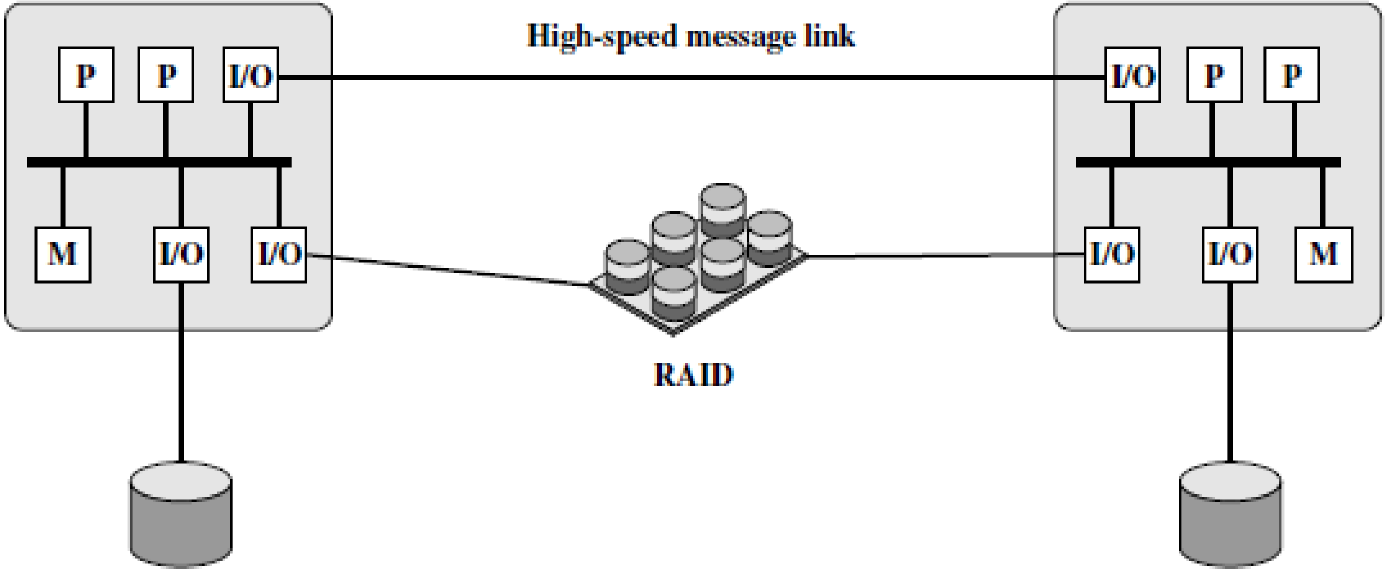
$$f=0.9975$$

Multi processors

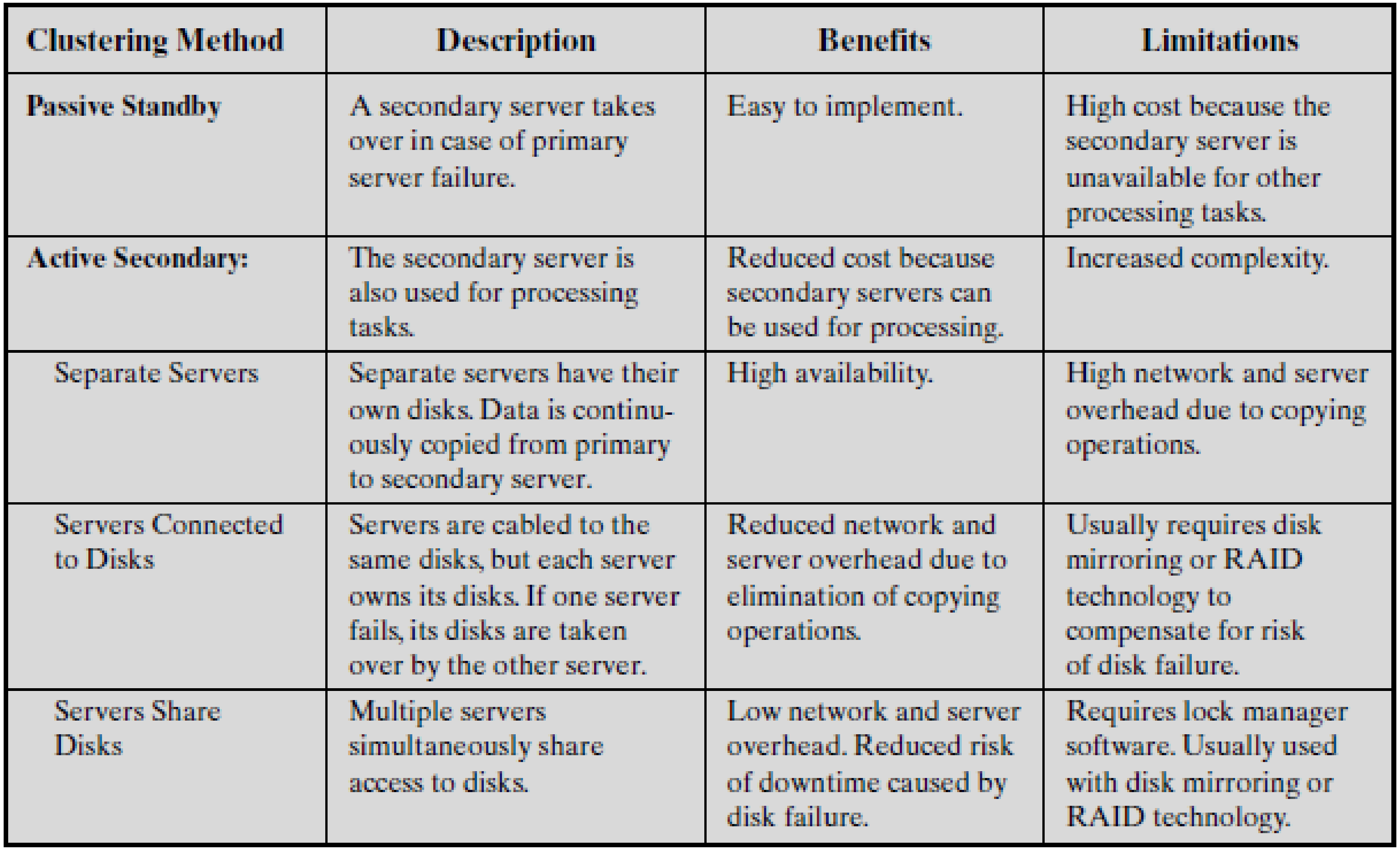
Main concepts:

- **Shared memory systems** (memory used for data interchange).
  - UMA (Uniform Memory Access)
  - NUMA (Non-Uniform Memory Access)
- **Distributed memory systems** (communication based on message passing).
- **Hybrid** “Shared-distributed” memory systems

Clusters



Clustering methods

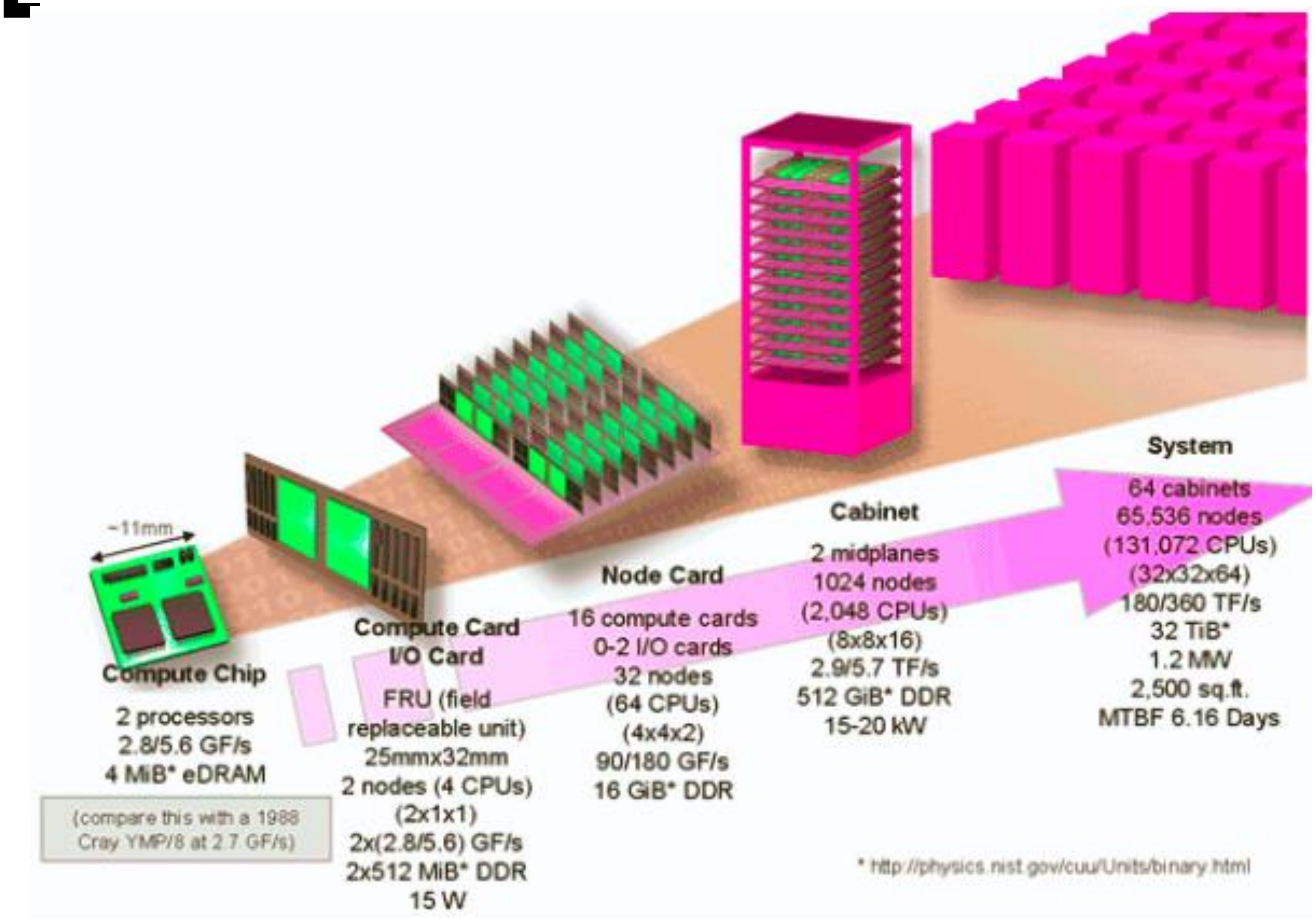


Clustering Method	Description	Benefits	Limitations
Passive Standby	A secondary server takes over in case of primary server failure.	Easy to implement.	High cost because the secondary server is unavailable for other processing tasks.
Active Secondary:	The secondary server is also used for processing tasks.	Reduced cost because secondary servers can be used for processing.	Increased complexity.
Separate Servers	Separate servers have their own disks. Data is continuously copied from primary to secondary server.	High availability.	High network and server overhead due to copying operations.
Servers Connected to Disks	Servers are cabled to the same disks, but each server owns its disks. If one server fails, its disks are taken over by the other server.	Reduced network and server overhead due to elimination of copying operations.	Usually requires disk mirroring or RAID technology to compensate for risk of disk failure.
Servers Share Disks	Multiple servers simultaneously share access to disks.	Low network and server overhead. Reduced risk of downtime caused by disk failure.	Requires lock manager software. Usually used with disk mirroring or RAID technology.

IBM Blue Gene/L

(No. 1 top500.org; 2004-2008)
Peak Performance – 478,2 TFlops.
Theoretical peak – 596,3 TFlops
Number of processors: 212.992

IBM Blue Gene/L



Compute Chip

2 processors
2.8/5.6 GF/s
4 MiB* eDRAM

(compare this with a 1988
Cray YMP/8 at 2.7 GF/s)

Compute Card
I/O Card

FRU (field
replaceable unit)
25mmx32mm
2 nodes (4 CPUs)
(2x1x1)
2x(2.8/5.6) GF/s
2x512 MiB* DDR
15 W

Node Card

16 compute cards
0-2 I/O cards
32 nodes
(64 CPUs)
(4x4x2)
90/180 GF/s
16 GiB* DDR

Cabinet

2 midplanes
1024 nodes
(2,048 CPUs)
(8x8x16)
2.9/5.7 TF/s
512 GiB* DDR
15-20 kW

System

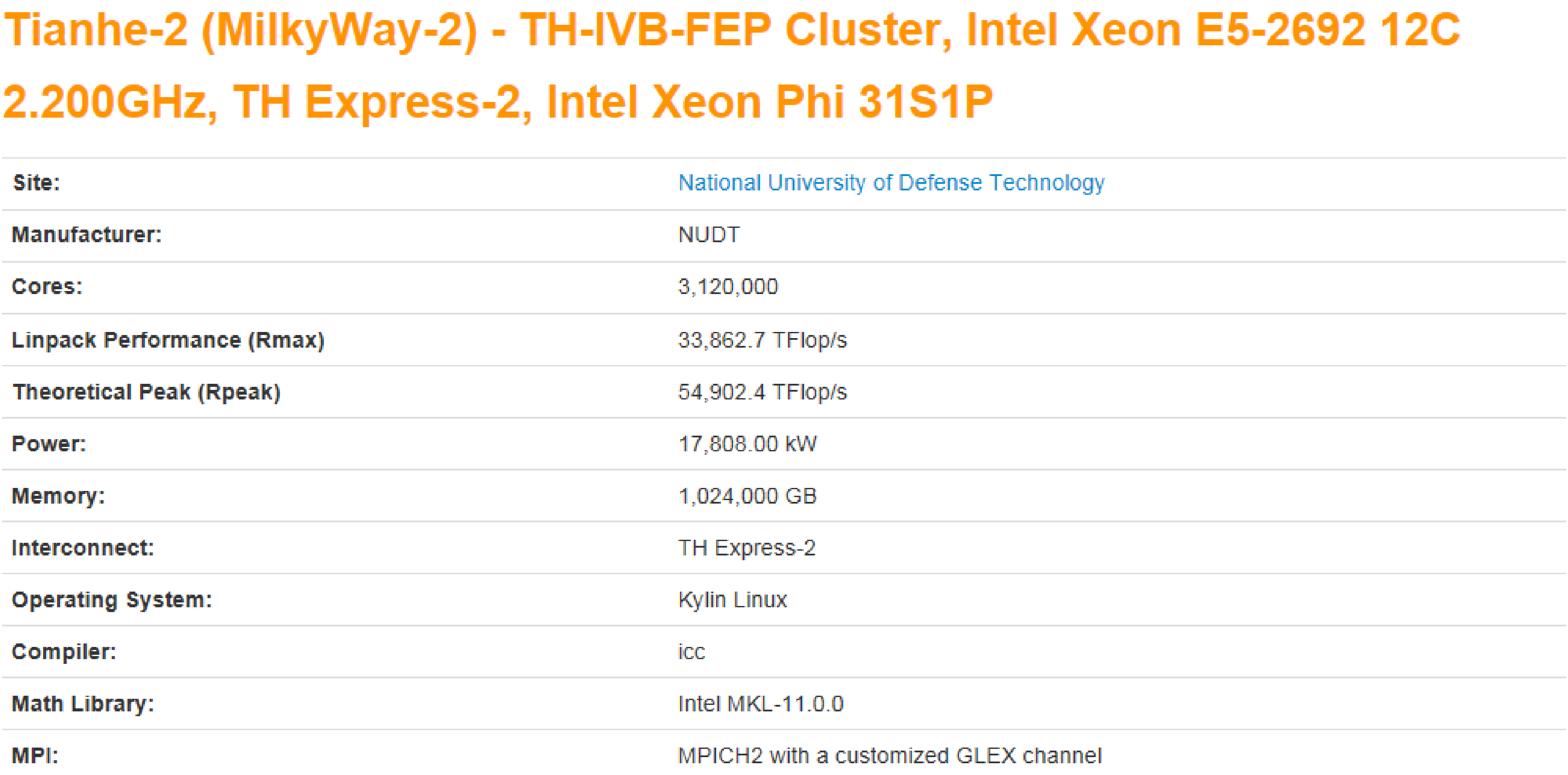
64 cabinets
65,536 nodes
(131,072 CPUs)
(32x32x64)
180/360 TF/s
32 TiB*
1.2 MW
2,500 sq.ft.
MTBF 6.16 Days

* http://physics.nist.gov/cuu/Units/binary.html

IBM Blue Gene/L

- Manufacturer : IBM
- Number processors : 212.992
- Processor type : PowerPC 440 core with FP enhancements,
- 700 MHz, 2.8 GFLOPS (peak)
- Number nodes : 65.536 (each with 2 processors)
- Main memory : 32.768 TB (0.5 GB per node)
- Disk space : 700 TB, 1,6 PB background memory
- Space requirement : 2.500 square feet (232.25 m2)
- Power consumption : 1.5 MW

Tianhe-2 (MilkyWay-2) 2014m



Tianhe-2 (MilkyWay-2) - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P

Site:	National University of Defense Technology
Manufacturer:	NUDT
Cores:	3,120,000
Linpack Performance (Rmax)	33,862.7 TFlop/s
Theoretical Peak (Rpeak)	54,902.4 TFlop/s
Power:	17,808.00 kW
Memory:	1,024,000 GB
Interconnect:	TH Express-2
Operating System:	Kylin Linux
Compiler:	icc
Math Library:	Intel MKL-11.0.0
MPI:	MPICH2 with a customized GLEX channel