
Computer Architecture

Ngo Lam Trung
Department of Computer Engineering
School of Information and Communication Technology (SoICT)
Hanoi University of Science and Technology
E-mail: trungnl@soict.hust.edu.vn

Course administration

- ❑ Instructor: Ngo Lam Trung
505 B1, SoICT, HUST
- ❑ Text: [Required] Computer Organization and Design, 4th edition revised printing
Patterson & Hennessy 2012.

[Optional] Computer Organization and Architecture, 8th Edition, William Stalling
- ❑ Slides: pdf
- ❑ Schedule: as in timetable

Course content

- ❑ Chapter 1: Introduction
- ❑ Chapter 2: Instruction Set Architecture
- ❑ Chapter 3: Computer Arithmetic
- ❑ Chapter 4: CPU
- ❑ Chapter 5: Memory
- ❑ Chapter 6: I/O system
- ❑ Chapter 7: Multicores and multiprocessors

Outcomes from this course

- ❑ Understanding of how high level language programs (C, Java...) translate into computer language programs, and how hardware execute the latter programs.
- ❑ Hardware/software interface, and how software instruct hardware to perform functions.
- ❑ The factors that determine computer performance.
- ❑ What techniques can be used to improve computer performance.
- ❑ Why computer moves from sequential to parallel processing.

Prerequisites - What You Should Know

- ❑ Look and feel of computer
 - | How computer parts look like?
- ❑ Basic logic design
 - | Combinational, sequential circuit design
- ❑ Create, compile, and run C/C++ programs

Chapter 1: Introduction

1. Computer Abstraction and Technology
2. Performance Evaluation

[with materials from *Computer Organization and Design, 4th Edition*,
Patterson & Hennessy, © 2008, MK
and M.J. Irwin's presentation, PSU 2008]

1. Computer Abstraction and Technology

- ❑ What is a computer?
- ❑ Computer classification
- ❑ Computer generations
- ❑ The key of computer evolution: IC making technology
- ❑ Computer organization

1. Computer Abstraction and Technology

❑ What is a computer?

❑ A machine that

- | Accepts input data
- | Processes data by executing a stored program
- | Produces output data

❑ Which one is computer?



1. Computer Abstraction and Technology

❑ **Computer architecture:** attributes of a system

- | Visible to a programmer
- | The logical execution of a program.

❑ **Computer organization**

- | The operational units and their interconnections that realize the architectural specifications.
- | include those hardware details transparent to the programmer,
 - control signals;
 - interfaces between the computer and peripherals;
 - the memory technology used...

❑ Which is more hardware/software related?

Classes of Computers

❑ Supercomputers

- | Super fast + expensive for high-end applications

❑ Server

- | Network based
- | High capacity, performance, reliability
- | Range from small servers to building sized

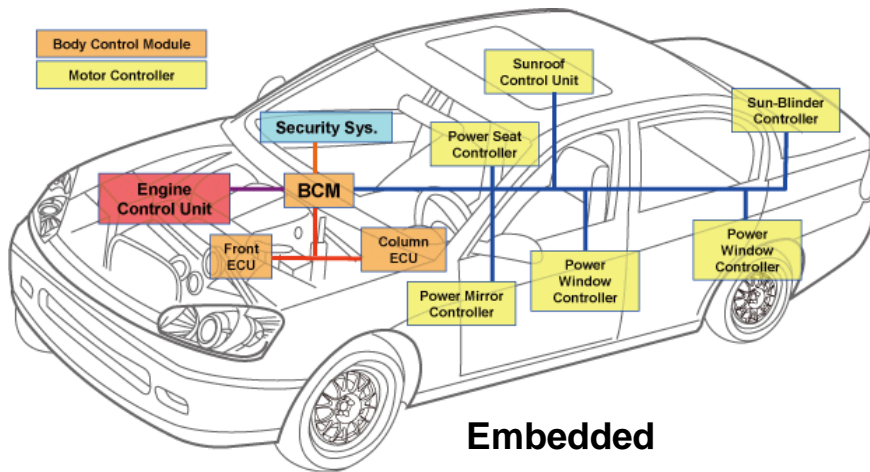
❑ Desktop computers

- | General purpose, variety of software
- | Subject to cost/performance tradeoff

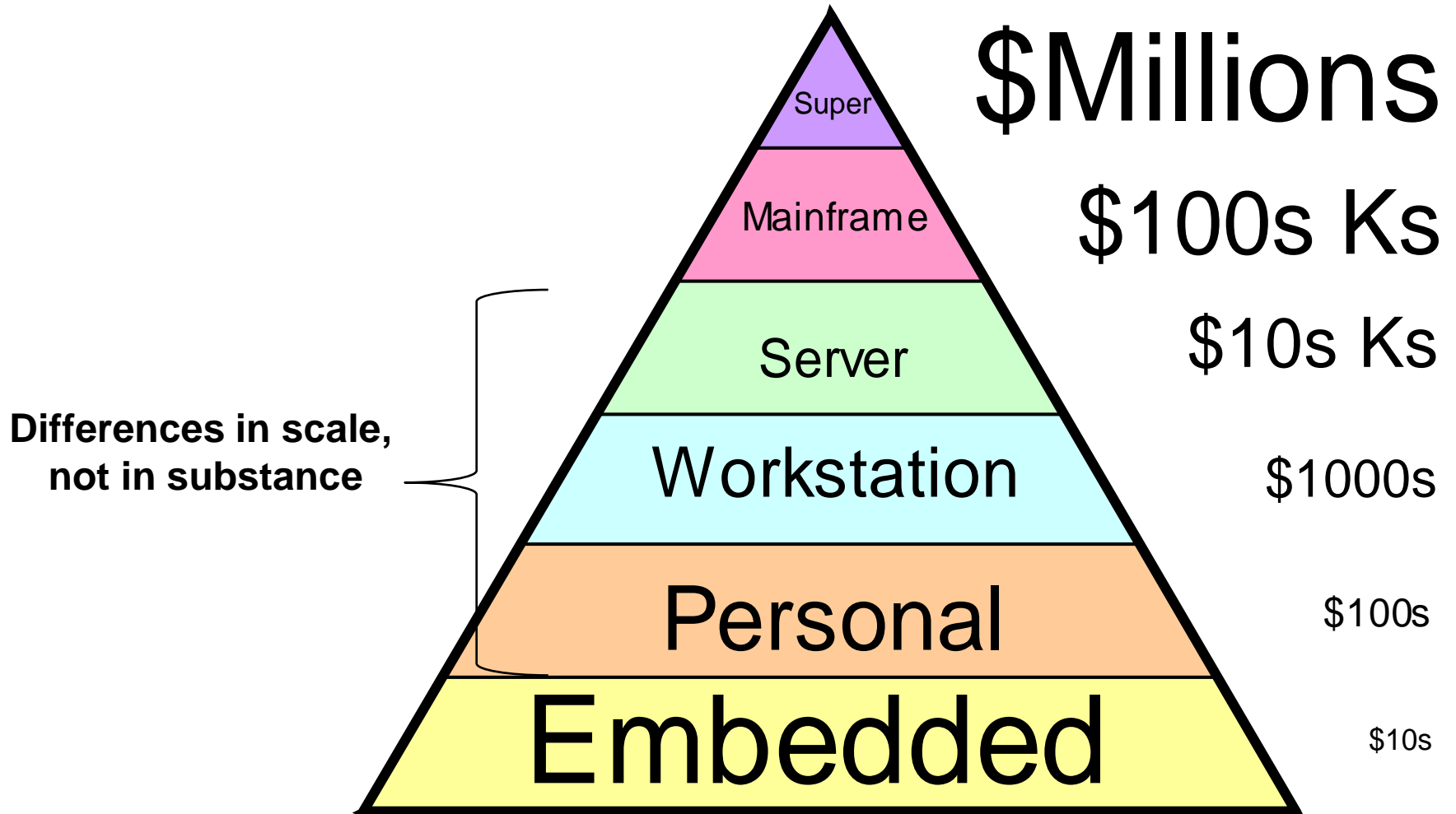
❑ Embedded computers

- | Hidden as components of systems
- | Stringent power/performance/cost constraints

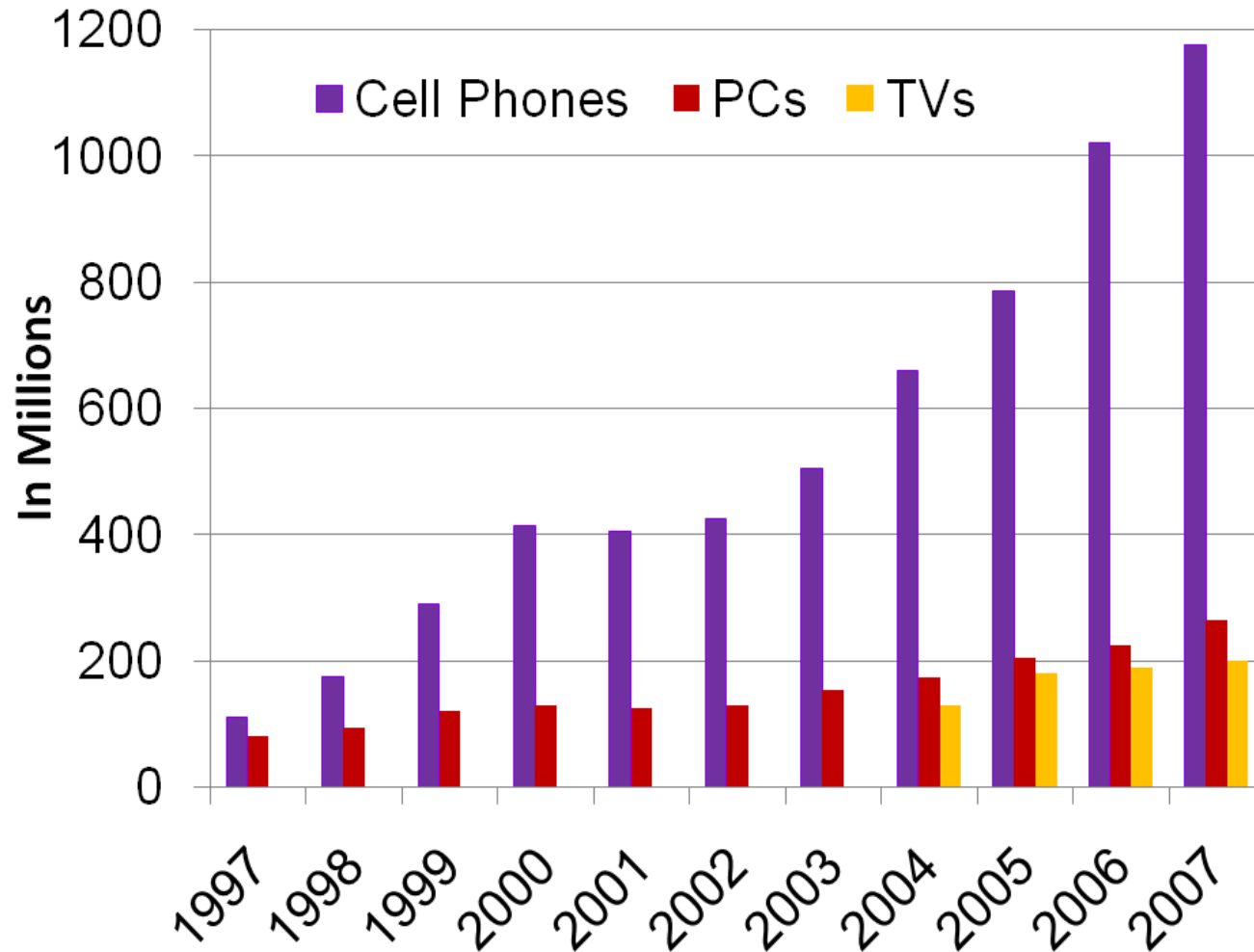
Dominant look and feel of computer classes



Price/performance of computer classes



The Processor Market



embedded growth >> desktop growth

Generations of Progress

Generation (begun)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & fell
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini
4 (1980s)	LSI/VLSI	SRAM/DRAM	Network, CD, mouse, sound	Desktop/ laptop micro
5 (1990s)	ULSI/GSI/ WSI, SOC	SDRAM, flash	Sensor/actuator point/click	Invisible, embedded

[Ref. 2]

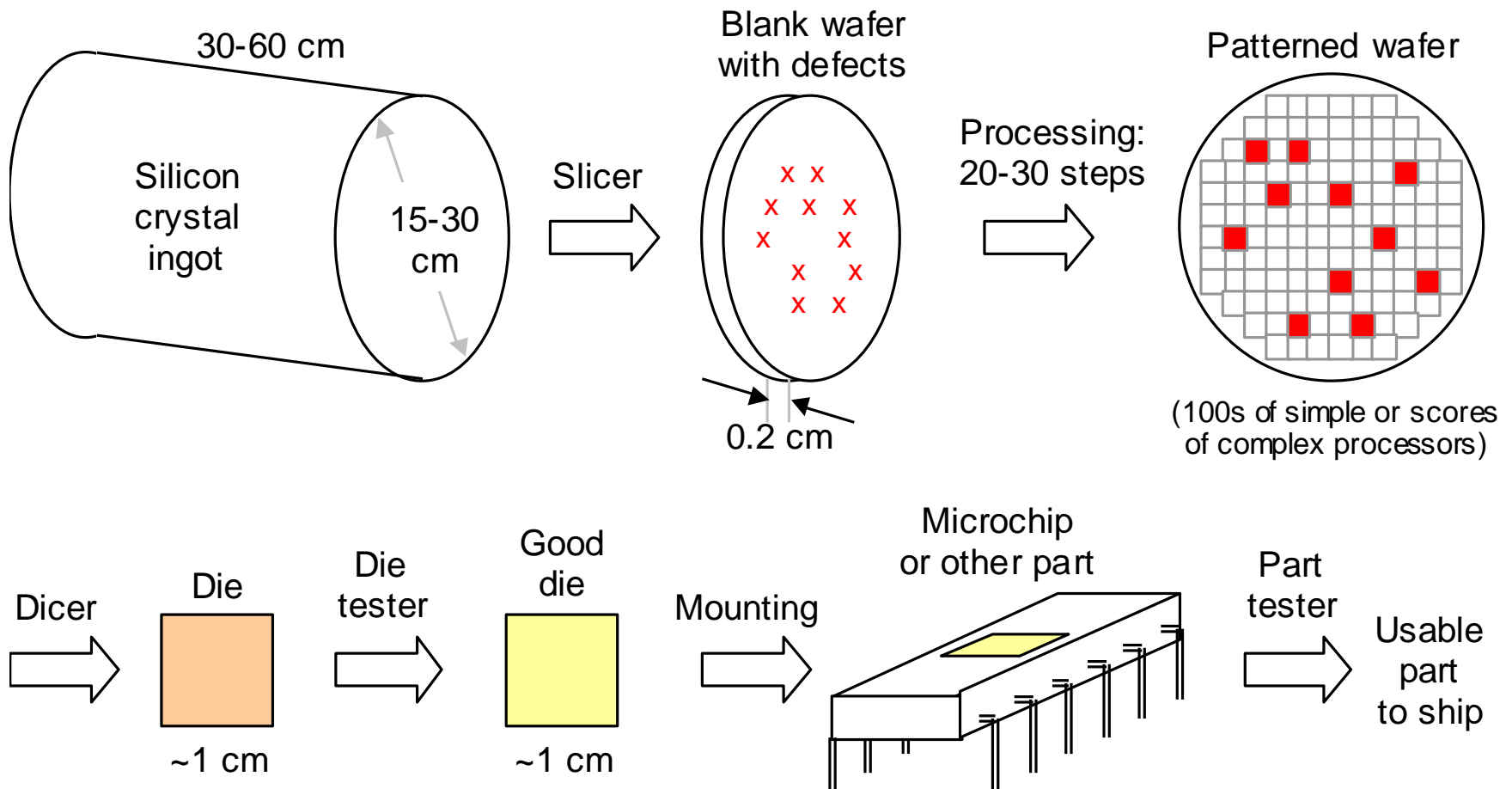
Key to computer evolution: IC making technology

- ❑ Electronics technology continues to evolve
 - | Increased capacity and performance
 - | Reduced cost

Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2005	Ultra large scale IC	6,200,000,000

[Textbook]

IC making



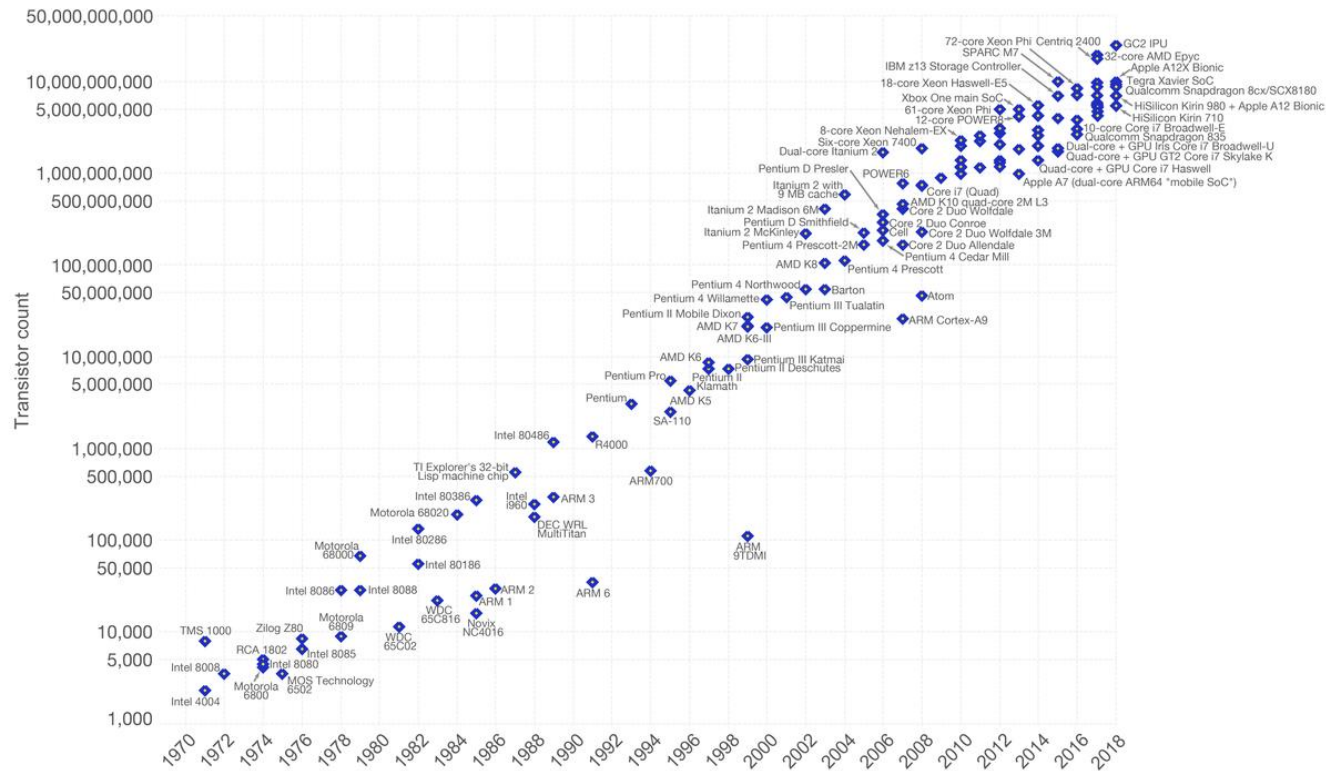
The manufacturing process for an IC part

Video: How an IC is made

Moore's Law

Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.



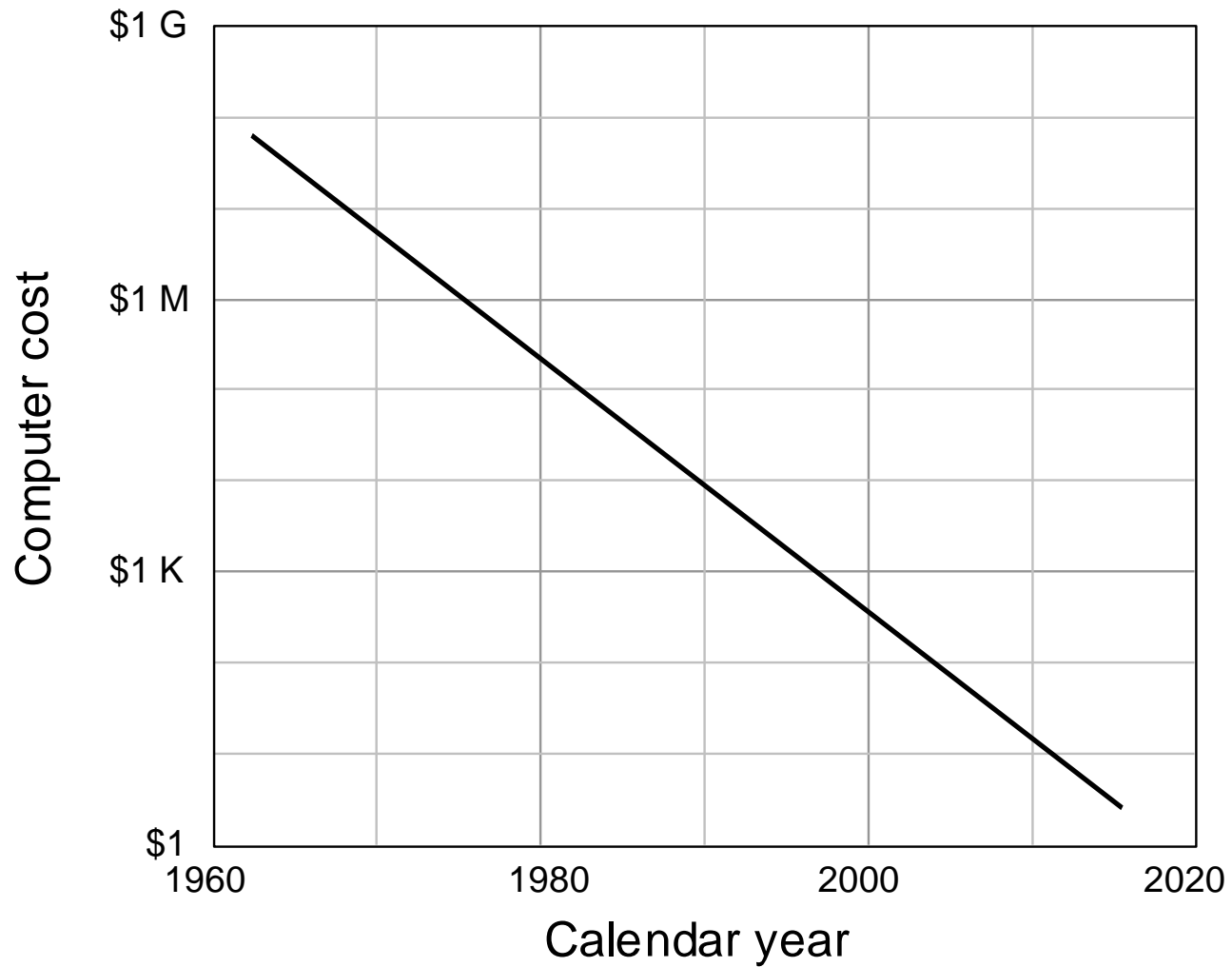
Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)

The data visualization is available at [OurWorldinData.org](https://ourworldindata.org). There you find more visualizations and research on this topic.

Licensed under [CC-BY-SA](#) by the author Max Roser.

How do we benefit from this?

Cost/performance



Homework

- ❑ Why CPU speed does not exceed 3.5-4.5GHz?

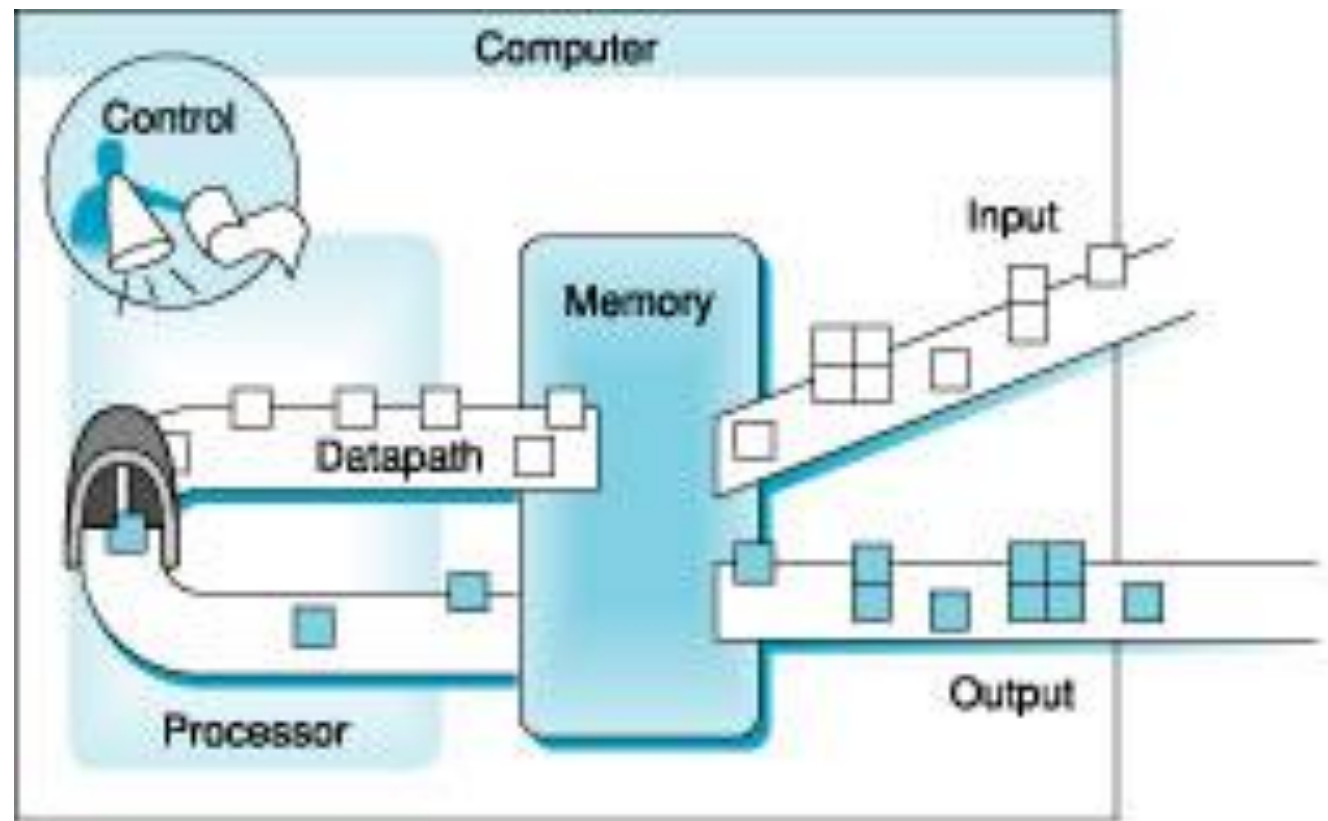
Computer Organization

- ❑ Computer's basic operation
 - | Input data
 - | Process data by executing stored program
 - | Output data
- ❑ What are required components of computer?
 - | For data input:
 - | For storing information:
 - | For program execution and data processing:
 - | For data output:

Computer Organization

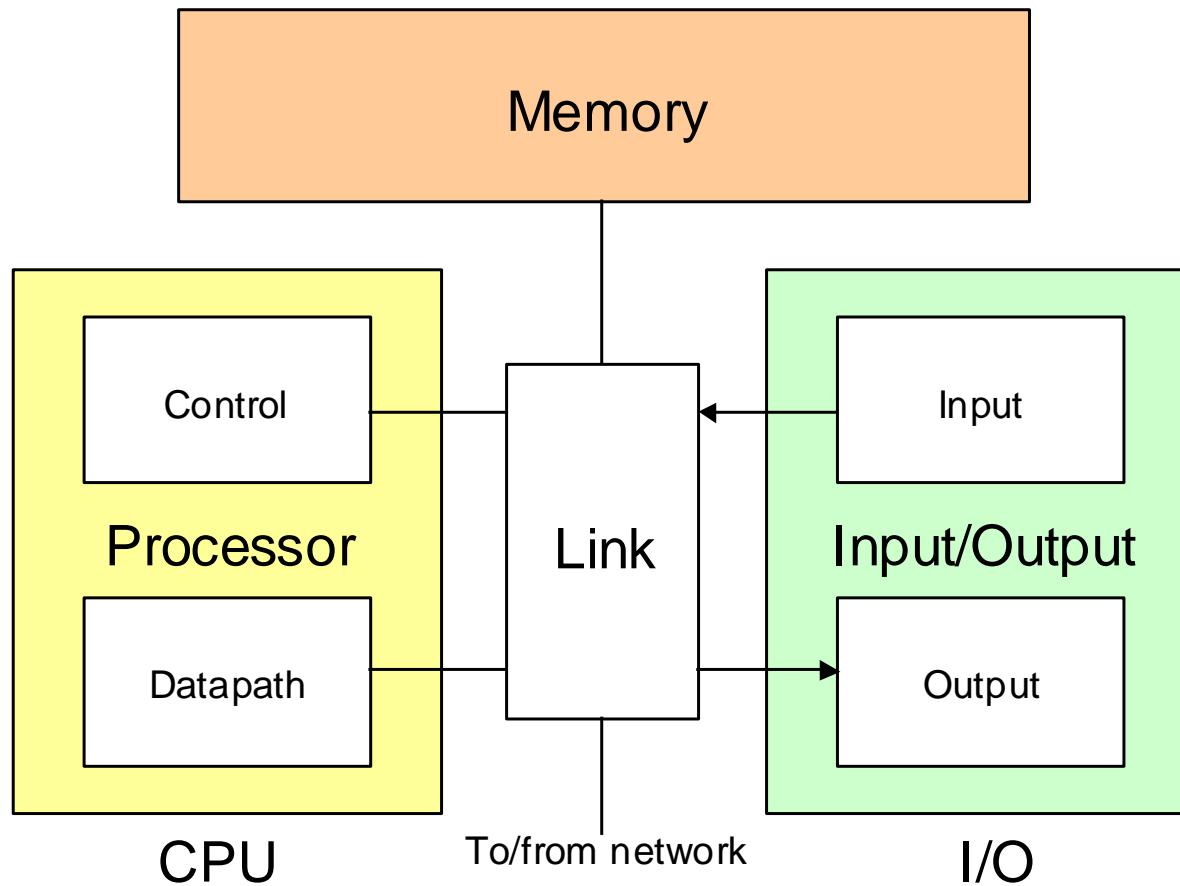
- ❑ Five classic components of a computer – input, output, memory, datapath, and control

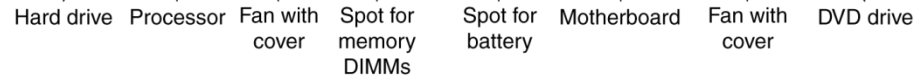
- ❑ datapath + control = processor (CPU)



A similar view

- ❑ Usually, the Link unit is hidden

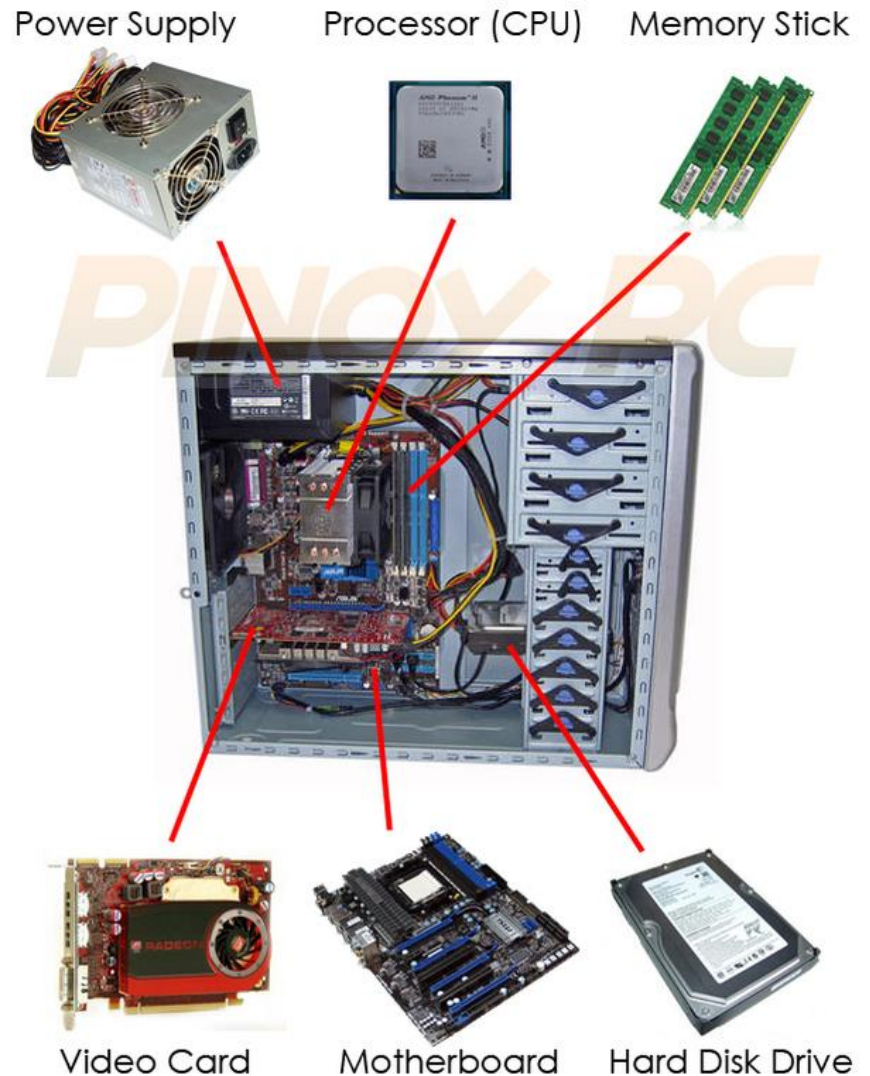




Opening the box: anatomy of computer



**The story of each component
worth a separate course!**

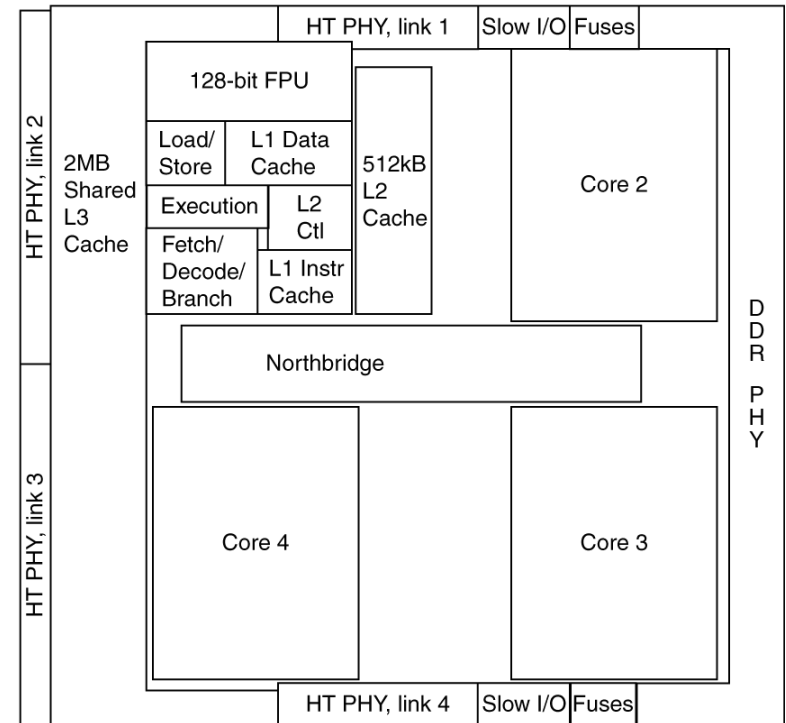
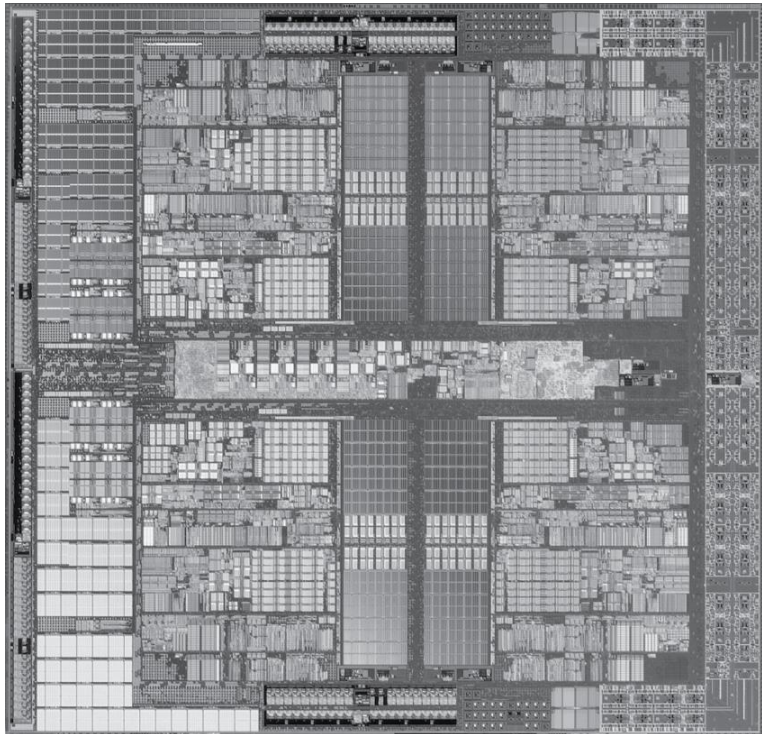


Inside the Processor (CPU)

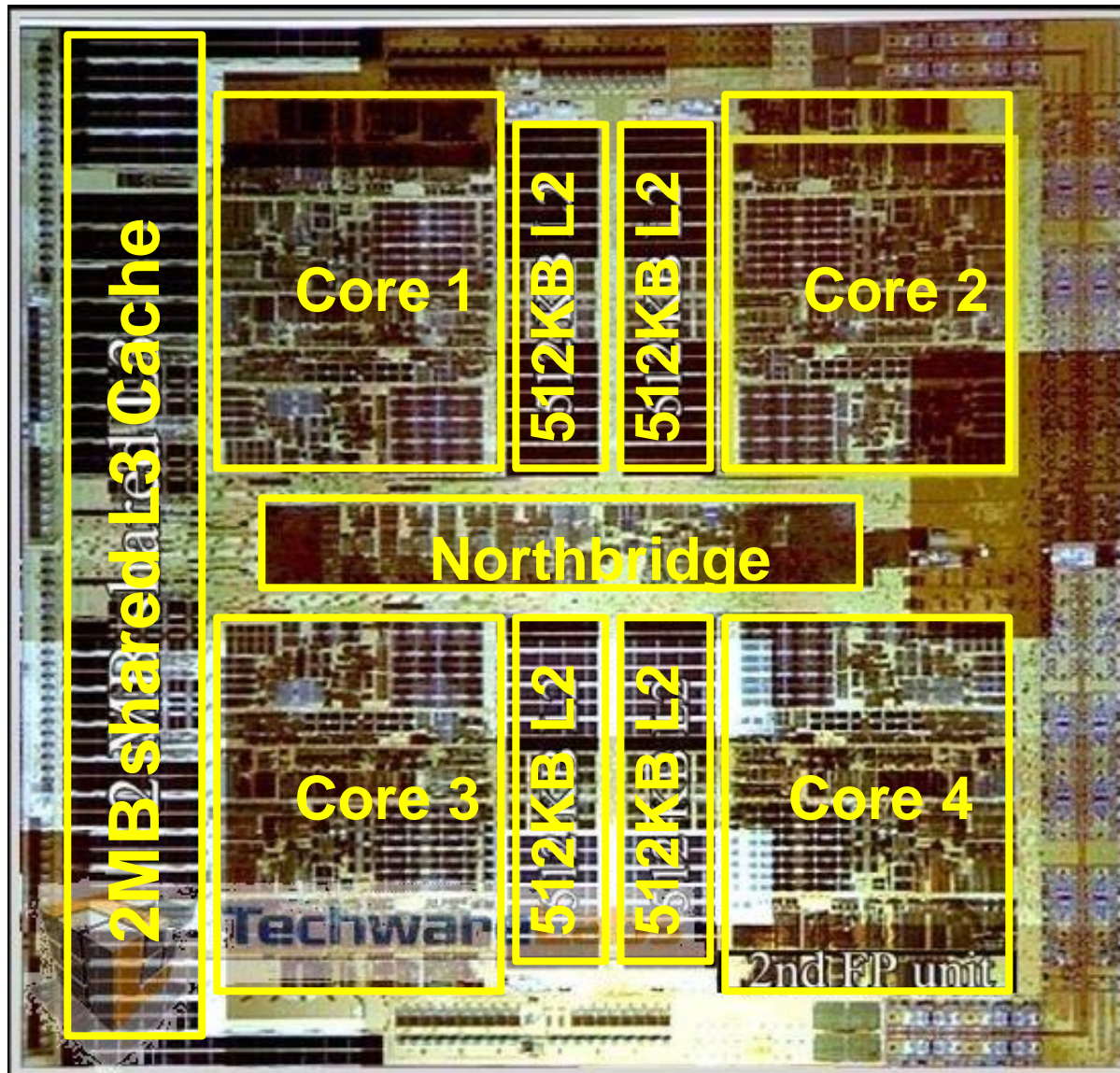
- ❑ Datapath: performs operations on data
- ❑ Control: sequences datapath, memory, ...
- ❑ Cache memory
 - | Small fast SRAM memory for immediate access to data

Inside the Processor

- AMD Barcelona: 4 processor cores



AMD's Barcelona Multicore Chip



- ❑ Four out-of-order cores on one chip
- ❑ 1.9 GHz clock rate
- ❑ 65nm technology
- ❑ Three levels of caches (L1, L2, L3) on chip
- ❑ Integrated Northbridge

What happened when you run your C program?

❑ High-level language program (in C)

```
swap (int v[], int k)
{
    int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

one-to-many

C compiler

❑ Assembly language program (for MIPS CPU)

```
swap:  sll    $2, $5, 2
        add    $2, $4, $2
        lw     $15, 0($2)
        lw     $16, 4($2)
        sw     $16, 0($2)
        sw     $15, 4($2)
        jr     $31
```

one-to-one

assembler

❑ Machine (object, binary) code (for MIPS CPU)

```
000000 00000 00101 0001000010000000
000000 00100 00010 0001000000100000
. . .
```

Levels of Program Code

❑ High-level language

- ❑ Level of abstraction closer to problem domain
- ❑ Provides for productivity and portability

❑ Assembly language

- ❑ Textual representation of instructions

❑ Hardware representation

- ❑ Binary digits (bits)
- ❑ Encoded instructions and data

High-level
language
program
(in C)

```
swap(int v[], int k)
{int temp;
  temp = v[k];
  v[k] = v[k+1];
  v[k+1] = temp;
}
```

Compiler

Assembly
language
program
(for MIPS)

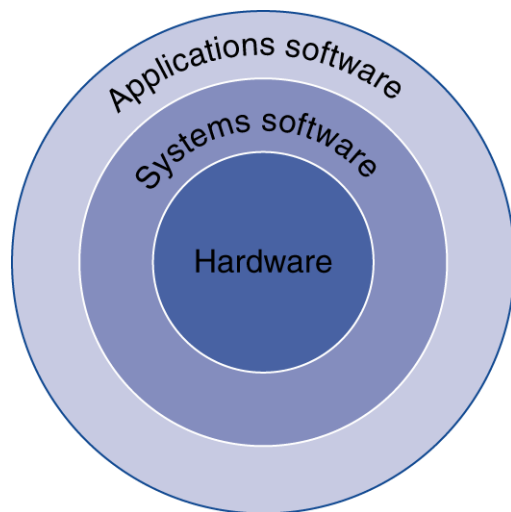
```
swap:
    muli $2, $5, 4
    add  $2, $4, $2
    lw   $15, 0($2)
    lw   $16, 4($2)
    sw   $16, 0($2)
    sw   $15, 4($2)
    jr   $31
```

Assembler

Binary machine
language
program
(for MIPS)

```
000000001010000100000000000011000
000000000000110000001100000100001
100011000110001000000000000000000
100011001111001000000000000000100
101011001111001000000000000000000
101011000110001000000000000000100
00000011111000000000000000001000
```

Hardware/software interface: below your program



- ❑ Application software
 - ❑ Written in high-level language (HLL)
- ❑ System software
 - ❑ Compiler: translates HLL code to machine code
 - ❑ Operating System: service code
 - Handling input/output
 - Managing memory and storage
 - Scheduling tasks & sharing resources
- ❑ Hardware
 - ❑ Processor, memory, I/O controllers

Problem

- ❑ Computer organization evolves fast
 - ❑ New processor
 - ❑ New hardware
 - ...every year/day/month
- ❑ How a program/software can run on different computers?
- ❑ Instruction Set Architecture
- ❑ Abstraction

2. Computer performance evaluation

- ❑ To maximize performance, need to **minimize** execution time

$$\text{performance}_x = 1 / \text{execution_time}_x$$

If computer X is n times faster than Y, then

$$\frac{\text{performance}_x}{\text{performance}_y} = \frac{\text{execution_time}_y}{\text{execution_time}_x} = n$$

Relative Performance Example

- ❑ If computer A runs a program in 10 seconds and computer B runs the same program in 15 seconds, how much faster is A than B?

We know that A is n times faster than B if

$$\frac{\text{performance}_A}{\text{performance}_B} = \frac{\text{execution_time}_B}{\text{execution_time}_A} = n$$

The performance ratio is $\frac{15}{10} = 1.5$

So A is 1.5 times faster than B

Performance Factors

- ❑ CPU execution time (CPU time) – time the CPU spends working on a task
 - ❑ Does not include time waiting for I/O or running other programs

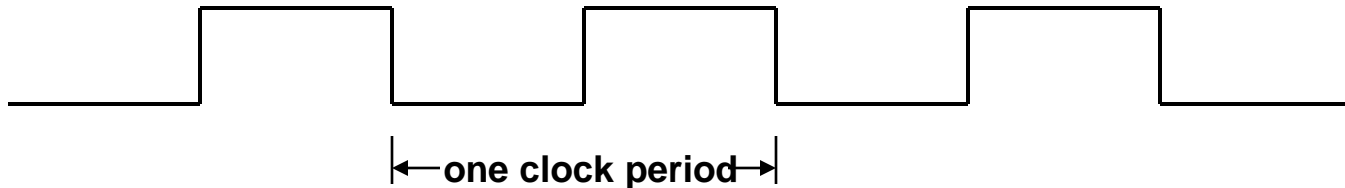
$$\begin{aligned}\text{CPU execution time for a program} &= \text{\# CPU clock cycles for a program} \times \text{clock cycle time} \\ &= \frac{\text{\# CPU clock cycles for a program}}{\text{clock rate}}\end{aligned}$$

- ❑ Can improve performance by reducing either the length of the clock cycle or the number of clock cycles required for a program

Review: Machine Clock Rate

- ❑ Clock rate (clock cycles per second in MHz or GHz) is inverse of clock cycle time (clock period)

$$CC = 1 / CR$$



10 nsec clock cycle => 100 MHz clock rate

5 nsec clock cycle => 200 MHz clock rate

2 nsec clock cycle => 500 MHz clock rate

1 nsec (10^{-9}) clock cycle => 1 GHz (10^9) clock rate

500 psec clock cycle => 2 GHz clock rate

250 psec clock cycle => 4 GHz clock rate

200 psec clock cycle => 5 GHz clock rate

Improving Performance Example

- ❑ A program runs on computer A with a 2 GHz clock in 10 seconds. What clock rate must computer B run at to run this program in 6 seconds? Assume that, computer B will require 1.2 times as many clock cycles as computer A to run the program.

$$\text{CPU time}_A = \frac{\text{CPU clock cycles}_A}{\text{clock rate}_A}$$

$$\begin{aligned}\text{CPU clock cycles}_A &= 10 \text{ sec} \times 2 \times 10^9 \text{ cycles/sec} \\ &= 20 \times 10^9 \text{ cycles}\end{aligned}$$

$$\text{CPU time}_B = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{\text{clock rate}_B}$$

$$\text{clock rate}_B = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{6 \text{ seconds}} = 4 \text{ GHz}$$

Clock Cycles per Instruction

- ❑ Not all instructions take the same amount of time to execute

┆ Average execution time ~ average clock cycles per instruction

$$\begin{array}{l} \text{\# CPU clock cycles} \\ \text{for a program} \end{array} = \begin{array}{l} \text{\# Instructions} \\ \text{for a program} \end{array} \times \begin{array}{l} \text{Average clock cycles} \\ \text{per instruction} \end{array}$$

- ❑ Clock cycles per instruction (CPI) – the average number of clock cycles each instruction takes to execute
 - ┆ A way to compare two different implementations of the same ISA

	CPI for this instruction class		
	A	B	C
CPI	1	2	3

Using the Performance Equation

- ❑ Computers A and B implement the same ISA. Computer A has a clock cycle time of 250 ps and an effective CPI of 2.0 for some program and computer B has a clock cycle time of 500 ps and an effective CPI of 1.2 for the same program. Which computer is faster and by how much?

Each computer executes the same number of instructions, I , so

$$\text{CPU time}_A = I \times 2.0 \times 250 \text{ ps} = 500 \times I \text{ ps}$$

$$\text{CPU time}_B = I \times 1.2 \times 500 \text{ ps} = 600 \times I \text{ ps}$$

Clearly, A is faster ... by the ratio of execution times

$$\frac{\text{performance}_A}{\text{performance}_B} = \frac{\text{execution_time}_B}{\text{execution_time}_A} = \frac{600 \times I \text{ ps}}{500 \times I \text{ ps}} = 1.2$$

The Performance Equation

- ❑ Our basic performance equation is then calculated

$$\begin{aligned}\text{CPU time} &= \text{Instruction_count} \times \text{CPI} \times \text{clock_cycle} \\ &= \frac{\text{Instruction_count} \times \text{CPI}}{\text{clock_rate}}\end{aligned}$$

- ❑ Key factors that affect performance (CPU execution time)
 - ❑ The clock rate: CPU specification
 - ❑ CPI: varies by instruction type and ISA implementation
 - ❑ Instruction count: measure by using profilers/ simulators

Dynamic Instruction Count

How many instructions are executed in this program fragment?

250 instructions

for i = 1, 100 do

20 instructions

for j = 1, 100 do

40 instructions

for k = 1, 100 do

10 instructions

endfor

endfor

endfor

Static count = 326

Each “for” consists of two instructions: increment index, check exit condition

12,422,450 Instructions

2 + 20 + 124,200 instructions

100 iterations

12,422,200 instructions in all

2 + 40 + 1200 instructions

100 iterations

124,200 instructions in all

2 + 10 instructions

100 iterations

1200 instructions in all

for i = 1, n
while x > 0

Improving performance by CPI

Op	Freq	CPI _i	Freq x CPI _i			
ALU	50%	1	.5	.5	.5	.25
Load	20%	5	1.0	.4	1.0	1.0
Store	10%	3	.3	.3	.3	.3
Branch	20%	2	.4	.4	.2	.4
$Avg\ CPI = \sum freq_i * CPI_i$			= 2.2	1.6	2.0	1.95

- ❑ How much faster would the machine be if a better data cache reduced the average load time to 2 cycles?

CPU time new = 1.6 x IC x CC so 2.2/1.6 means 37.5% faster

- ❑ What if branch instruction is only one cycle?

CPU time new = 2.0 x IC x CC so 2.2/2.0 means 10% faster

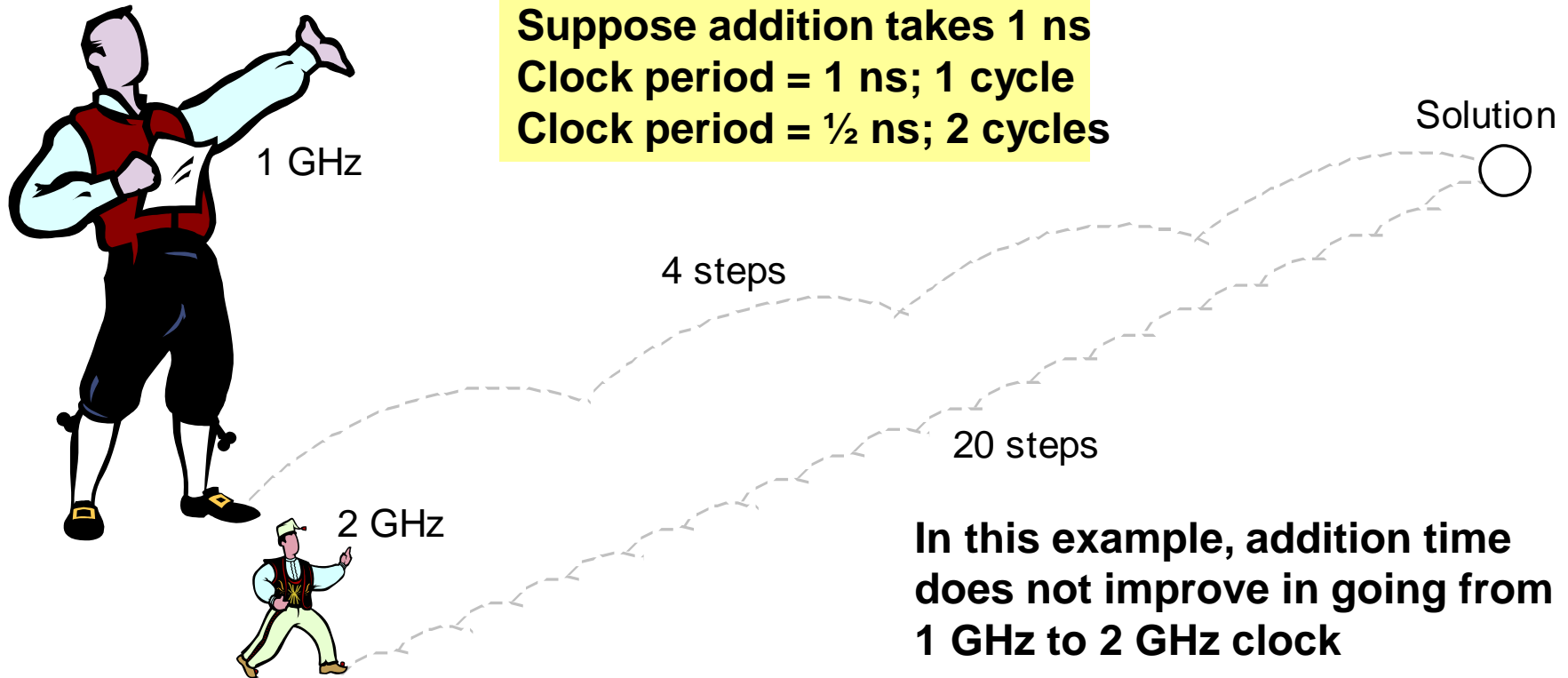
- ❑ What if two ALU instructions could be executed at once?

CPU time new = 1.95 x IC x CC so 2.2/1.95 means 12.8% faster

How to improve performance?

- ❑ Shorter clock cycle = faster clock rate
 - latest CPU technology
- ❑ Smaller CPI
 - optimizing Instruction Set Architecture
- ❑ Smaller instruction count
 - optimizing algorithm and compiler
- ❑ To get best performance, multiple criteria are combined and considered at design time
 - specific CPU for specific class computation problem

Faster Clock \neq Shorter Running Time



Faster steps do not necessarily mean shorter travel time.