

# **CSC1104 - COMPUTER ORGANISATION & ARCHITECTURE**

## **LECTURE 1 : COMPUTER EVOLUTION AND PERFORMANCE**

**Associate Professor Cao Qi**  
**[Qi.Cao@Glasgow.ac.uk](mailto:Qi.Cao@Glasgow.ac.uk)**

**WORLD  
CHANGERS  
WELCOME**

◆ Module-Lead: Dr Cao Qi:

➤ Email: [Qi.Cao@glasgow.ac.uk](mailto:Qi.Cao@glasgow.ac.uk)

➤ Webpage:

<https://www.gla.ac.uk/schools/computing/staff/qicao/>

◆ Technical Officer: Mr. Vincent Ng Chiew Guan

➤ Email: [vincent.ng@singaporetech.edu.sg](mailto:vincent.ng@singaporetech.edu.sg)

# Acknowledgement

◆ Main contents of CSC1104 - Computer Organisation and Architecture are derived from:

➤ **Computer organization and architecture, Designing for performance.** Author: William Stallings. Publisher: Pearson.

Acknowledgement to: Author and Publisher.

➤ **Computer organization and Design, The hardware/software interface.** Authors: D. Patterson and J. Hennessy. Publisher: Morgan Kaufmann.

Acknowledgement to: Authors and Publisher.

# Course Objectives :

- ◆ To present the nature and characteristics of modern-day computer systems, about their structure and function.
- ◆ To provide a thorough discussion of the fundamentals of computer organization and architecture.
- ◆ To learn programming on IoT devices.

# Course Venues

- ◆ Lectures: Monday 11:00 am – 13:00 pm, by Zoom
- ◆ Tutorials: Tuesday, 9:00 am – 10:00 am, E2-02-14-Lectorial 10
- ◆ Group Labs, E2-04-02-SR232
  - Group P1 : Tuesday, 10:00 – 11:00 am
  - Group P2 : Tuesday, 11:00 am – 12:00 pm
  - Group P3 : Tuesday, 12:00 – 13:00 pm



# Assessments

- ◆ Quiz 1 - worth 30% of your overall marks
- ◆ Project Assignment - worth 30% of your overall marks
- ◆ Weekly group class discussions – worth 5% of your overall marks
- ◆ Final exam - worth 35% of your overall marks

# Course Schedule

Week	Topics
1	Computer Evolution and Performance
2	Computer Function, Cache Memory
3	Internal Memory and External Memory
4	Input/Output, Operating System
5	Number Systems and Computer Arithmetic
6	Assembly Language
7	Break
8	Instruction Set: Characteristics and Data Types Quiz 1 (Week 1-5), 30%
9	Instruction Set: Types of Operations
10	Transfer of Control and Addressing Modes
11	Instruction Pipelining; CISC and RISC
12	Parallelism; Superscalar; Multicore Project assignment due, 30%
13	Revisions Weekly group class discussions, 5% <sup>7</sup>

# Books and References

- ◆ Computer organization and architecture, Designing for performance, William Stallings, Pearson, 10th edition, 2016.
- ◆ Computer organization and Design, The hardware and software interface, David A. Patterson and John L. Hennessy, Morgan Kaufmann; 5th edition 2014.



# Lecture Contents

- ◆ Brief History of Computers:
  - Classes of Computers
  - Four Generations of Computers
- ◆ Evolutions of Processors, CISC and RISC:
  - CISC Processors and RISC Processors
  - Computer Architecture and Organization
- ◆ Processors Performance:
  - Measures of Performance






# Classes of Computers and Their Applications

- ◆ Servers (*supercomputers*):
  - Many processors, large memory, high cost,
- ◆ Personal computers (PCs):
  - For single users at low cost.
- ◆ Embedded computers:
  - Largest class of computers.
- ◆ Smart Mobile Devices:
  - Powerful as PCs.



# History - Generations of Computers

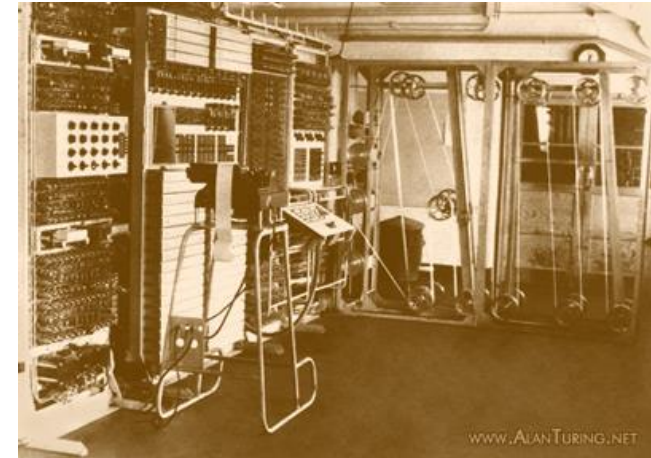
Generation	Approximate Dates	Technology	Typical Speed (operations per second)
1	1946–1957	Vacuum tube	40,000
2	1957–1964	Transistor	200,000
3	1965–1971	Small- and medium-scale integration <b>Integrated Circuits</b>	1,000,000
4	<b>Later Generations</b> {           1972–1977 1978–1991 1991–	Large scale integration	10,000,000
5		Very large scale integration	100,000,000
6		Ultra large scale integration	>1,000,000,000

- ◆ Processing power  Memory capacity  Dimensions 
- ◆ Complexity  Control units 
- ◆ **System software**: load programs, move data from/to peripherals, perform common computations.

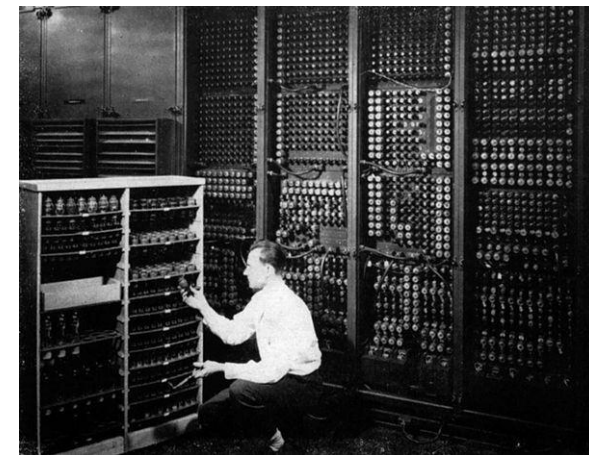
# First Generation: Vacuum Tubes



- ◆ Used **vacuum tubes** for digital logic elements and memory.
- ◆ First computer: **COLOSSUS**, by in 1943-44.
- ◆ First general-purpose computer: **ENIAC** in 1943–46.



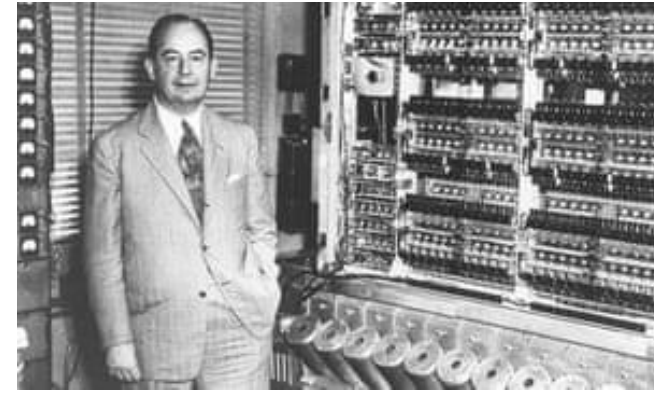
*Computer COLOSSUS*



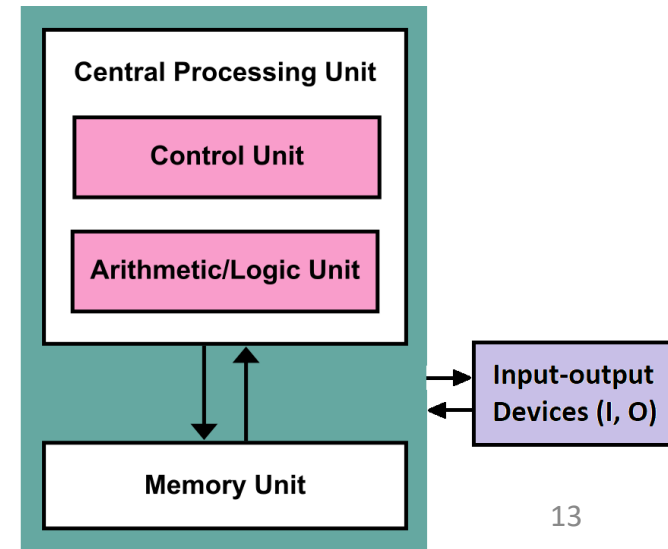
*Computer ENIAC tube change<sup>12</sup>*

# Von-Neumann Architecture

- ◆ **Von-Neumann architecture** by John von Neumann in 1945.
- ◆ **Stored-program computer:** instruction and program stored in same memory.
  - Memory unit.
  - Arithmetic logic unit (ALU).
  - Control unit.
  - Input–output (I/O)

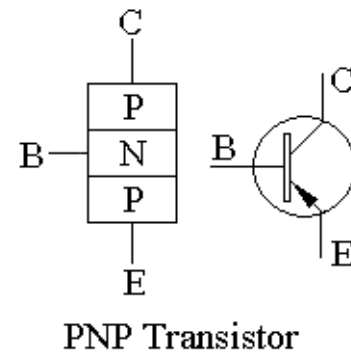
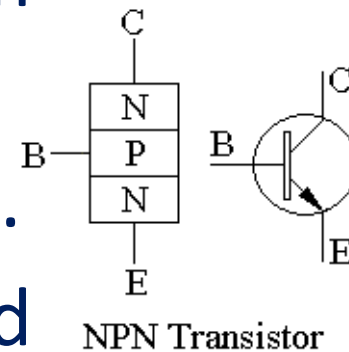


*John von Neumann with the stored-program IAS computer*



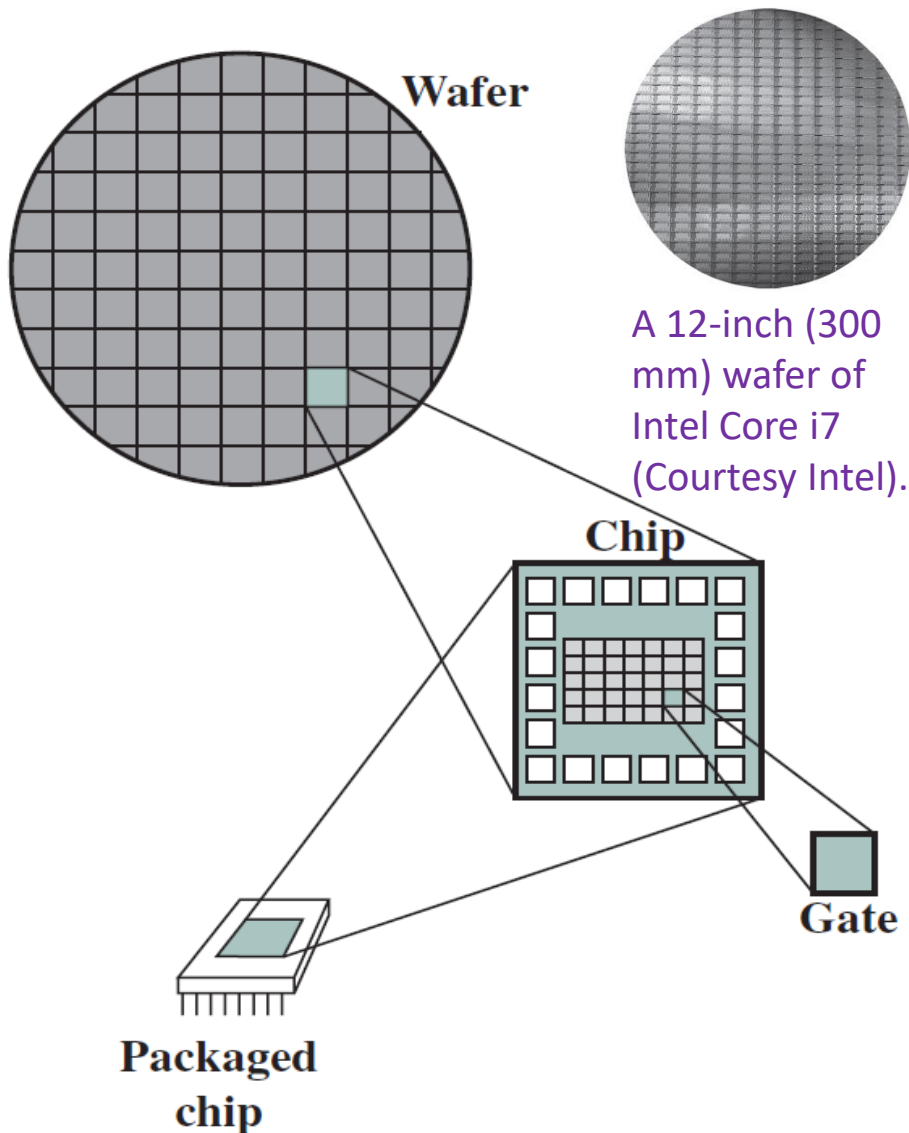
# Second Generation: Transistors

- ◆ **Transistor:** solid-state silicon device.
- ◆ Invented at Bell Labs in 1947.
- ◆ Fully transistorized computers available in late 1950s.





# Third Generation: Integrated Circuits (IC)






- ◆ Microelectronics era: invention of IC in 1958.

# Later Generation: Large-Scale Integration (LSI)

LSI: 1,000 – 10,000  
components per chip

VLSI: 10,000 – 1 million  
components per chip

ULSI: > 1 million  
components per chip

- ◆ Construction of *processors* (control unit, arithmetic and logic unit).
- ◆ Construction of *memory chips* (storage density  cost per bit  access time ).

# Decimal and Binary Notations for Size Terms

Decimal term	Abbreviation	Value	Binary term	Abbreviation	Value	% Larger
<u>kilobyte</u>	KB	<u><math>10^3</math></u>	<u>kibibyte</u>	KiB	<u><math>2^{10}</math></u>	2%
<u>megabyte</u>	MB	<u><math>10^6</math></u>	<u>mebibyte</u>	MiB	<u><math>2^{20}</math></u>	5%
<u>gigabyte</u>	GB	<u><math>10^9</math></u>	<u>gibibyte</u>	GiB	<u><math>2^{30}</math></u>	7%
terabyte	TB	$10^{12}$	tebibyte	TiB	$2^{40}$	10%
petabyte	PB	$10^{15}$	pebibyte	PiB	$2^{50}$	13%
exabyte	EB	$10^{18}$	exbibyte	EiB	$2^{60}$	15%
zettabyte	ZB	$10^{21}$	zebibyte	ZiB	$2^{70}$	18%
yottabyte	YB	$10^{24}$	yobibyte	YiB	$2^{80}$	21%

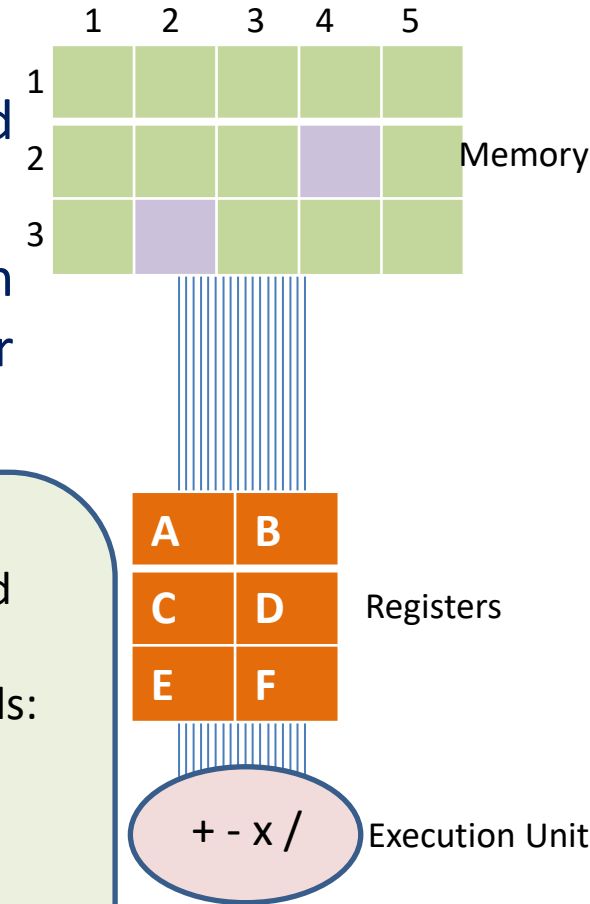
- ◆  $2^x$  vs.  $10^y$  bytes ambiguity was resolved by adding a binary notation.
- ◆ **1Ki** =  $2^{10}$  (1,024), **1Mi** =  $2^{20}$  (1,048,576), **1Gi** =  $2^{30}$  (1,073,741,824).
- ◆ **1K** =  $10^3$  (1,000), **1M** =  $10^6$  (1,000,000), **1G** =  $10^9$  (1,000,000,000).

# CISC v.s. RISC

- ◆ CISC and RISC: different on instruction set.
- ◆ Complex Instruction Set Computer (CISC) processor architecture :
  - Complete task using a smaller number of assembly lines.
  - Example processor: Intel X86 Architecture
- ◆ Reduced Instruction Set Computer (RISC) processor architecture :
  - Complete task utilizing small and highly optimized set of instructions.
  - Example processor: ARM Architecture

# Example - Difference of CISC and RISC

- ◆ Multiplying data (two numbers) stored in memory.
- ◆ Execution unit can only operate data been loaded into registers.
- ◆ Task: 2 numbers - one stored at memory location (2:4), the other stored at (3:2). Calculate their product then store the product back to (2:4).



## CISC:

- A specific instruction is used ("MULT").
- The entire task of multiplying 2 numbers can be completed with one instruction:

**MULT 2:4, 3:2**

## RISC:

- Use simple instructions executed within 1 clock cycle.
- "MULT" divided into 3 commands: "LOAD", "MUL", "STORE".
- Need to code four lines.

**LOAD A, 2:4**

**LOAD B, 3:2**

**MUL A, B**

**STORE 2:4, A**

# Intel x86 Architecture – CISC Processors

- ◆ With instruction set architecture (ISA) for microprocessor-based computing.
- ◆ Program written on older version can execute on newer versions. All changes have involved additions to the instruction set.
- ◆ Over 500 instructions in the instruction set.
- ◆ x86 represents design effort on CISC.



# ARM Architecture – RISC Processors

- ◆ ARM: RISC-based microprocessors by ARM Holdings.
- ◆ For high-performance, low-power-consumption, small-size, and low-cost processor for embedded systems.
- ◆ ***Embedded system***: a dedicated function embedded as a part of a complete device or system.

***Millions of*** computers are sold each year.

***Billions of*** embedded computer systems are produced each year.

# Computer Architecture and Computer Organisation

## Computer Architecture

- Attributes of a system visible to programmers.
- Attributes direct impact on logical execution of program.

### Architectural Example Attributes :

- instruction set,
- number of bits to represent various data types,
- I/O mechanisms,
- memory addressing modes.

## Computer Organisation

- Operational units and their interconnections that realize architectural specifications.

### Organizational Example Attributes :

- hardware details transparent to programmer, such as control signals,
- interfaces between computer and peripherals,
- memory technology used.

# Distinction between Architecture and Organization

- ◆ Many manufacturers offer a family of computer models, with same architecture but different in organization.
- ◆ A particular computer architecture may span many years.
- ◆ Computer organizations change with technology, price, performance characteristics, computer models. etc.



- ◆ For microcomputers, changes in technology influence not only organization but also more architectures.
- ◆ Generally, there is less of a requirement for generation-to-generation compatibility for these smaller machines.

# Function Operations

## ◆ 4 basic functions performed by computers:

### ➤ Data Processing

- ❑ Processing a wide variety of forms of data.

### ➤ Data Storage

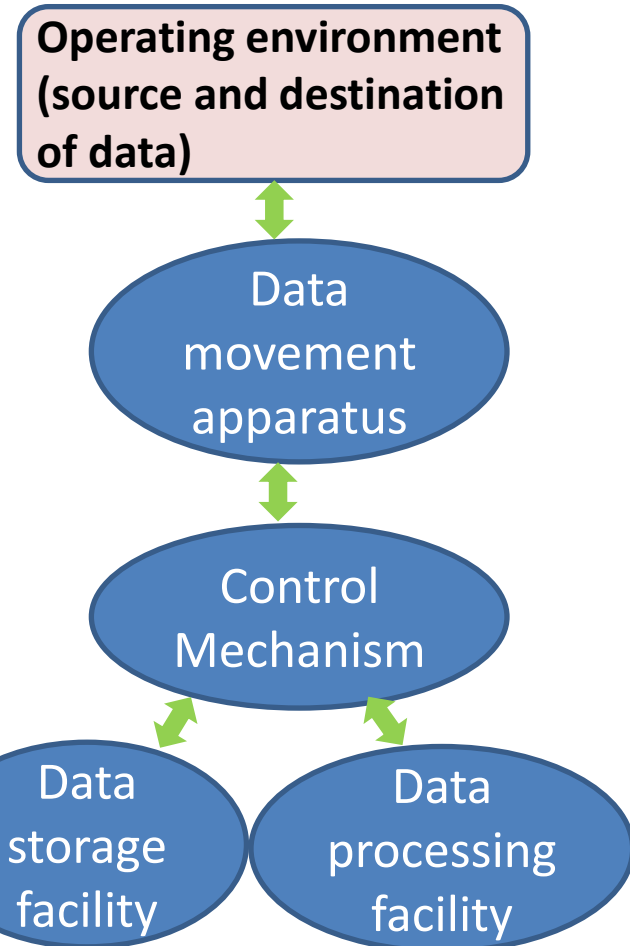
- ❑ Short term storage.
- ❑ Long term storage.

### ➤ Data Movement

- ❑ Input–output (I/O): when data received from or delivered to a peripheral device.
- ❑ Communications: when data moved over longer distances, to/from a remote device.

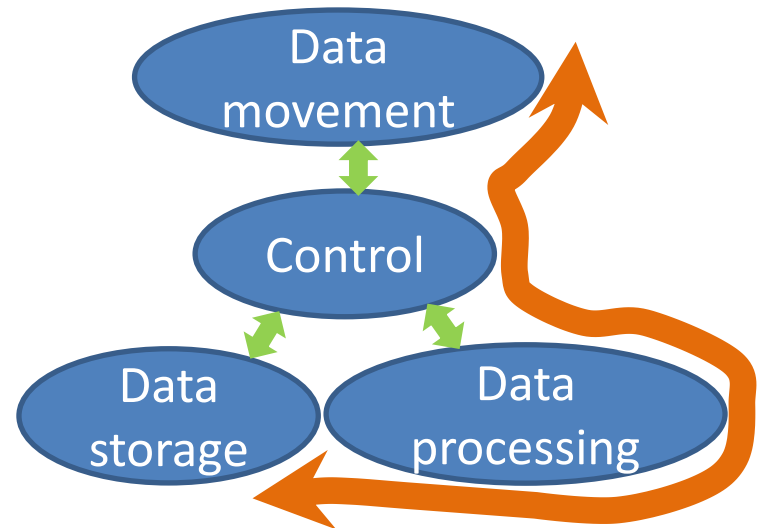
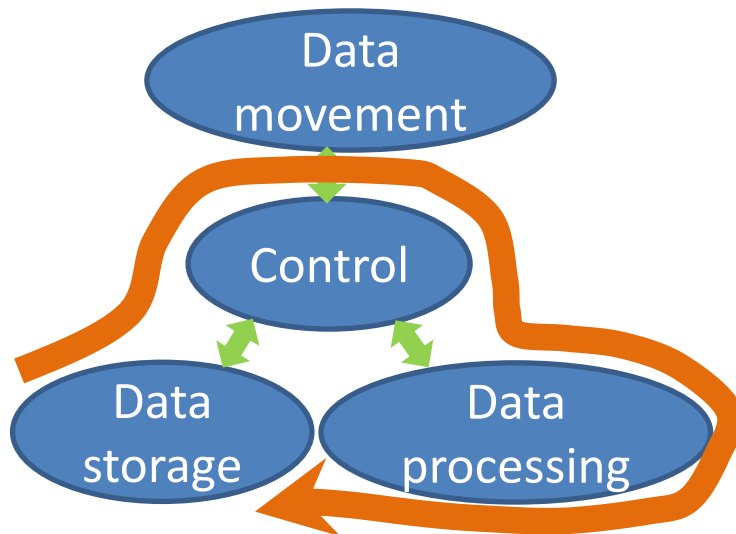
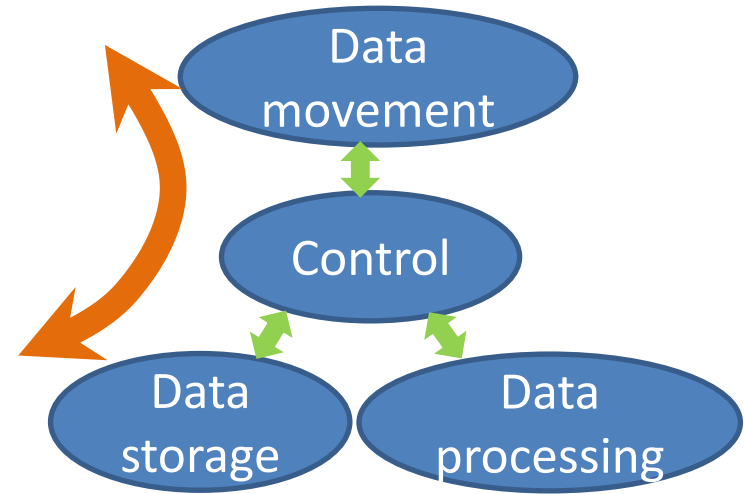
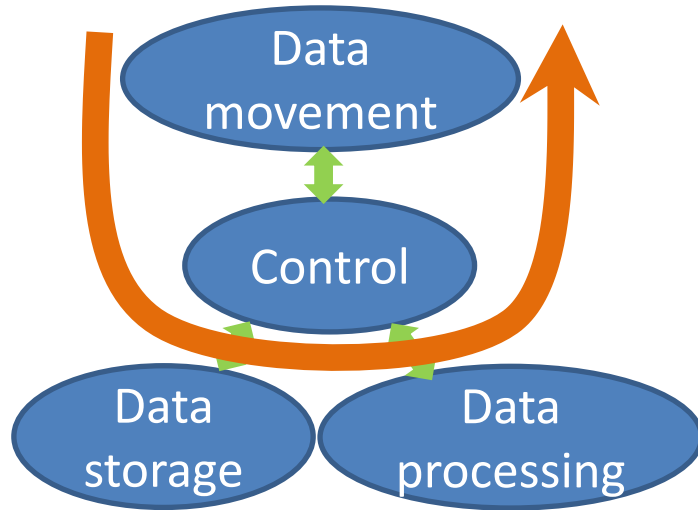
### ➤ Control

- ❑ Managing computer's resources and performance of its functional parts.





# Examples of Computer Function Operations



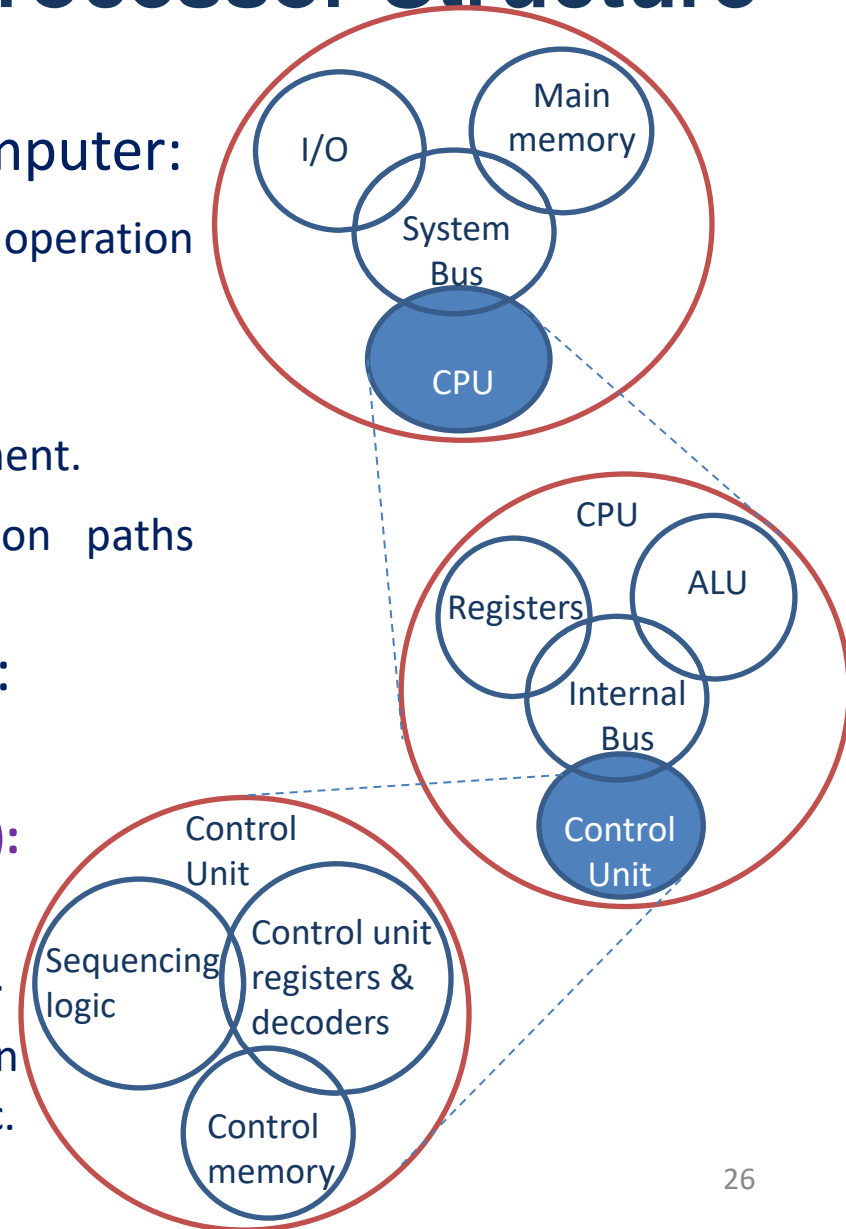
# Single Core Processor Structure

## ◆ Main structural components of a computer:

- **Central processing unit (CPU):** Controls operation and performs data processing functions.
- **Main memory:** Stores data.
- **I/O:** Moves data from/to external environment.
- **System interconnection:** Communication paths among CPU, main memory, and I/O.

## ◆ Major structural components of CPU:

- **Control unit:** Controls operation of CPU.
- **Arithmetic and logic unit (ALU):** Performs data processing functions.
- **Registers:** Provides internal storage of CPU.
- **CPU interconnection:** Communication paths among control unit, ALU, registers, etc.

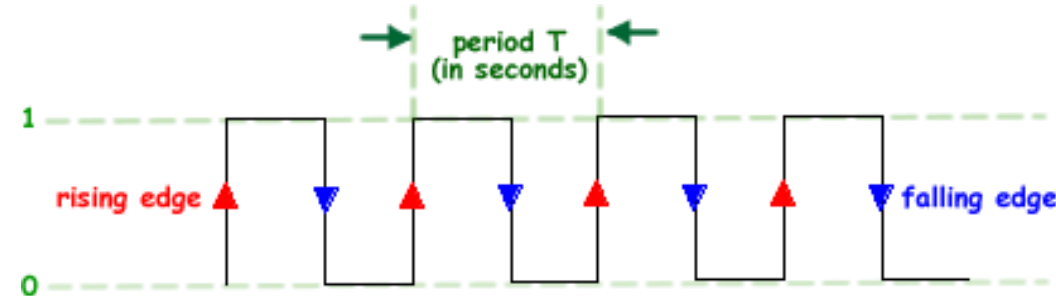




# Clock Cycle Time or Clock Rate

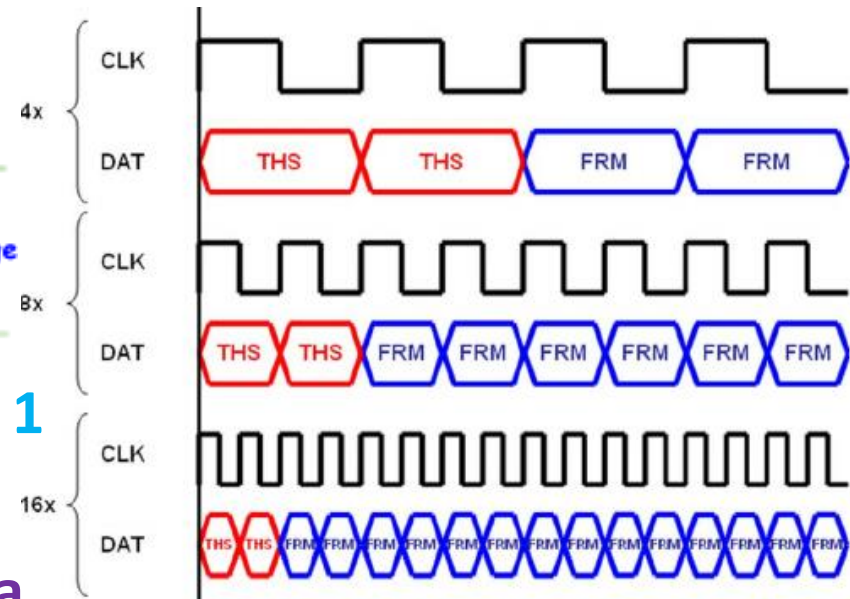
- ◆ Clock speed of processor (**clock cycle time (or period of clock signal)**): period  $T$ . **Clock rate or clock speed**: frequency  $f$ ).

➤ Clock rate  $f = 1/T$



e.g., 1-MHz processor receives 1 million clock pulses per second.

- The higher clock rate, the more data can be processed within a fixed time under same conditions.



- Clock rate  , time to process each operation 

# Processors Performance

◆ Performance formula relates to number of clock cycles and ***clock cycle time*** (Period) to CPU time:

➤ Execution time for a program ↓ = No. of CPU clock cycles needed ↓ × Clock cycle time ↓

◆ ***Clock rate*** (Frequency) inverse to ***clock cycle time***:

➤ Execution time ↓ =  $\frac{\text{No. of CPU clock cycles needed} \downarrow}{\text{Clock Rate} \uparrow}$

◆ Hence, ways to improve CPU performance:

- Reduce number of clock cycles needed for a program
- Reduce the clock cycle time, (or increase clock rate).

# Example 1.1 – CPU Performance

- ❖ A program runs in 10 seconds on computer A which has 2 GHz clock rate. The designer plans to build a computer B, run the same program in 6 seconds. A substantial increase in clock rate is possible, but will affect the rest of CPU design, causing computer B to require 1.2 times as many clock cycles as computer A. What clock rate is for computer B?

**Solution:**

□ **Number of clock cycles needed for the program on computer A:**

- **No. of CPU clock cycles on A = Execution time on A \* Clock Rate on A = 10 seconds \*  $2 \times 10^9$  cycles per second =  $20 \times 10^9$  clock cycles.**
- **No. of CPU clock cycles on B =  $1.2 \times$  No. of CPU clock cycles on A =  $1.2 \times 20 \times 10^9$  cycles =  $24 \times 10^9$  cycles.**
- **Clock Rate on B =  $\frac{\text{No. of CPU clock cycles on B}}{\text{Execution time on B}} = \frac{24 \times 10^9 \text{ cycles}}{6 \text{ seconds}} = 4 \times 10^9 \text{ cycles/second} = 4 \text{ GHz}$**

# Instruction Performance

- ◆ Execution time also depends on No. of CPU instructions in a program, and clock cycles per instruction.
- ◆ ***Clock cycles per instruction*** (CPI): Average number of clock cycles per instruction for a program.

- Different instructions may need different number of clock cycles;
- CPI is average clock cycles of all instructions executed in a program.

$$CPI = \frac{\sum_{i=1}^n (CPI_i \times \text{Instruction Count}_i)}{\text{Total Instruction Count}}$$

where  $CPI_i$  and  $\text{Instruction Count}_i$  are for each instruction class  $i$ .

- ◆ Performance formula relates CPI to CPU time as:

- No. of CPU clock cycles needed =  
Total No. of instructions  $\times$  clock cycles per instruction (CPI).

# Example 1.2 - Instruction Performance

- ❖ Two computers with same instruction set architecture. Computer A has a clock cycle time of 250 ps and a CPI of 2.0 for a program. Computer B has a clock cycle time of 500 ps and a CPI of 1.2 for same program. Which computer is faster for this program, by how much? (*Hint: 1 ps = 1 / 10<sup>12</sup> Hz*).

**Solution:**

- ❑ Each computer executes the same number of instructions for the program, first calculate No. of CPU clock cycles needed :
  - CPU clock cycles on A = No. of instruction  $\times$  CPI on A = No. of instruction  $\times$  2.0.
  - CPU clock cycles on B = No. of instruction  $\times$  CPI on B = No. of instruction  $\times$  1.2.
- Then compute execution time for each computer:
  - Execution time on A = CPU clock cycles on A  $\times$  Clock cycle time A = No. of instruction  $\times$  2.0  $\times$  250 ps.
  - Execution time on B = CPU clock cycles on B  $\times$  Clock cycle time B = No. of instruction  $\times$  1.2  $\times$  500 ps.
- Hence, computer A is faster for it, with less CPU execution time at:
 
$$\frac{\text{No. of instruction} \times 500 \text{ ps}}{\text{No. of instruction} \times 600 \text{ ps}} \times 100\% = \frac{5}{6} \times 100\% = 83.33\%.$$

# CPU Performance Equations

- ◆ CPU performance determined by 3 key factors:
  - ◆ **Instruction count** (No. of instructions executed by a program),
  - ◆ **CPI** (Clock cycles per instruction),
  - ◆ **Clock cycle time** (or *Clock Rate*).

- ◆ CPU performance equation:

➤ Execution time = Instruction count  $\times$  CPI  $\times$  Clock cycle time.

- ◆ Clock rate is inverse to clock cycle time:

➤ Execution time =  $\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock Rate}}$ .

- ◆ Millions of instructions per second (MIPS):

*as a common measure of performance.*

➤ MIPS rate =  $\frac{\text{Instruction count}}{\text{Execution time} \times 10^6} = \frac{1}{\text{CPI} \times \text{Clock Cycle time} \times 10^6} = \frac{\text{Clock Rate}}{\text{CPI} \times 10^6}$

Important equations!



# Units of Measurement for CPU Performance

Components of performance	Units of measure
CPU execution time for a program	Seconds for the program
Instruction count	Instructions executed for the program
Clock cycles per instruction (CPI)	Average number of clock cycles per instruction
Clock cycle time	Seconds per clock cycle

◆ Reliable measure of computer performance is time.

➤ Execution time (unit: seconds/program) =

$$\frac{\text{Instruction count}}{\text{program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycles}}$$

Execution time = Instruction count × CPI × Clock cycle time

# Example 1.3 - CPU Performance

- ◆ A program needs execution of 2 million instructions on a **400 MHz** CPU. The program consists of 4 major types of instructions. Instruction mix and CPI for each type are below, based on a program trace experiment. Calculate the MIPS rate.

Instruction Type	CPI	Instruction Mix (%)
Arithmetic and logic	1	60
Load/store with cache hit	2	18
Branch	4	12
Memory reference with cache miss	8	10

## Solution:

- According to the table above, average **CPI** of all instructions is:

- $\text{CPI} = 1 \times 60\% + 2 \times 18\% + 4 \times 12\% + 8 \times 10\% = 2.24.$

- Clock rate is 400 MHz. Based on the MIPS rate equation:

- $$\text{MIPS Rate} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6} = \frac{1}{\text{CPI} \times \text{Clock Cycle time} \times 10^6} =$$

$$\frac{\text{Clock Rate}}{\text{CPI} \times 10^6} = \frac{400 \times 10^6}{2.24 \times 10^6} = 179.$$

# Example 1.4 – Compare Code Performance

- ◆ A program consists of 3 major classes of instructions. Hardware designers supplied following facts:

	Instruction Class A	Instruction Class B	Instruction Class C
CPI	1	2	3

For a high-level language program, compiler designer plans two code sequences requiring the following instruction counts. Which code sequence executes more instructions? Which is faster? What is the CPI for each sequence?

Code Sequence	Instruction counts for each instruction class		
	Class A	Class B	Class C
X	2	1	2
Y	4	1	1

**Solution:**

- ❑ Total No. of instructions executed by Code Sequence X:  $2 + 1 + 2 = 5$ .
- ❑ Total No. of instructions executed by Code Sequence Y:  $4 + 1 + 1 = 6$ .
- ❑ CPU clock cycles =  $\sum (\text{instruction count}_i \times \text{CPI}_i \text{ of each instruction})$
- ❑ Total No. of CPU clock cycles by Code Sequence X:  $2 \times 1 + 1 \times 2 + 2 \times 3 = 10$ .
- ❑ Total No. of CPU clock cycles by Code Sequence Y:  $4 \times 1 + 1 \times 2 + 1 \times 3 = 9$ .
- ❑ CPI of Sequence X: 
$$CPI_x = \frac{\sum_{i=1}^n (CPI_i \times \text{Instruction Count}_i)}{\text{Total Instruction Count}} = \frac{10}{5} = 2$$
- ❑ CPI of Sequence y: 
$$CPI_y = \frac{\sum_{i=1}^n (CPI_i \times \text{Instruction Count}_i)}{\text{Total Instruction Count}} = \frac{9}{6} = 1.5$$

# Understanding Program Performance

Component	Affect	How?
Algorithm	Instruction count, CPI	Determines No. of instructions executed. May affect CPI, by favoring slower or faster instructions. E.g., if algorithm uses more divisions, tends to a higher CPI.
Programming language	Instruction count, CPI	Affects instruction count, as statements in language are translated to machine instructions. May affect CPI due to its features; e.g., a language with heavy data abstraction (e.g., Java) requires indirect calls, uses higher CPI.
Compiler	Instruction count, CPI	Compiler efficiency affects both instruction count and CPI, as compiler translates source language instructions into machine instructions.
Instruction set architecture	Instruction count, clock rate, CPI	Affects all 3 aspects of CPU performance: instructions needed for a program, clock cycles of each instruction, and overall clock rate of CPU.

## Next Lecture

# Lecture 2: Computer Function and Cache Memory