# Module I: Part B

# Stored Program Concept

- Stored program concept is introduced by John von Neumann in 1940s.
- The idea of a stored program is to store the instructions and data electronically as binary numbers in a storage space associated with a computer.
- The storage space is called as memory.
- Any data such as input or instruction is stored as a binary number in the memory.

# How to implement the Stored-Program Concept in reality?

- To Implement the stored-program concept, there are multiple layers of abstractions required both in the hardware and software.

- The most important among them is the instruction set architecture (ISA) -abstraction between the lowest level software interface and the underlying hardware.

# Von Neumann machine

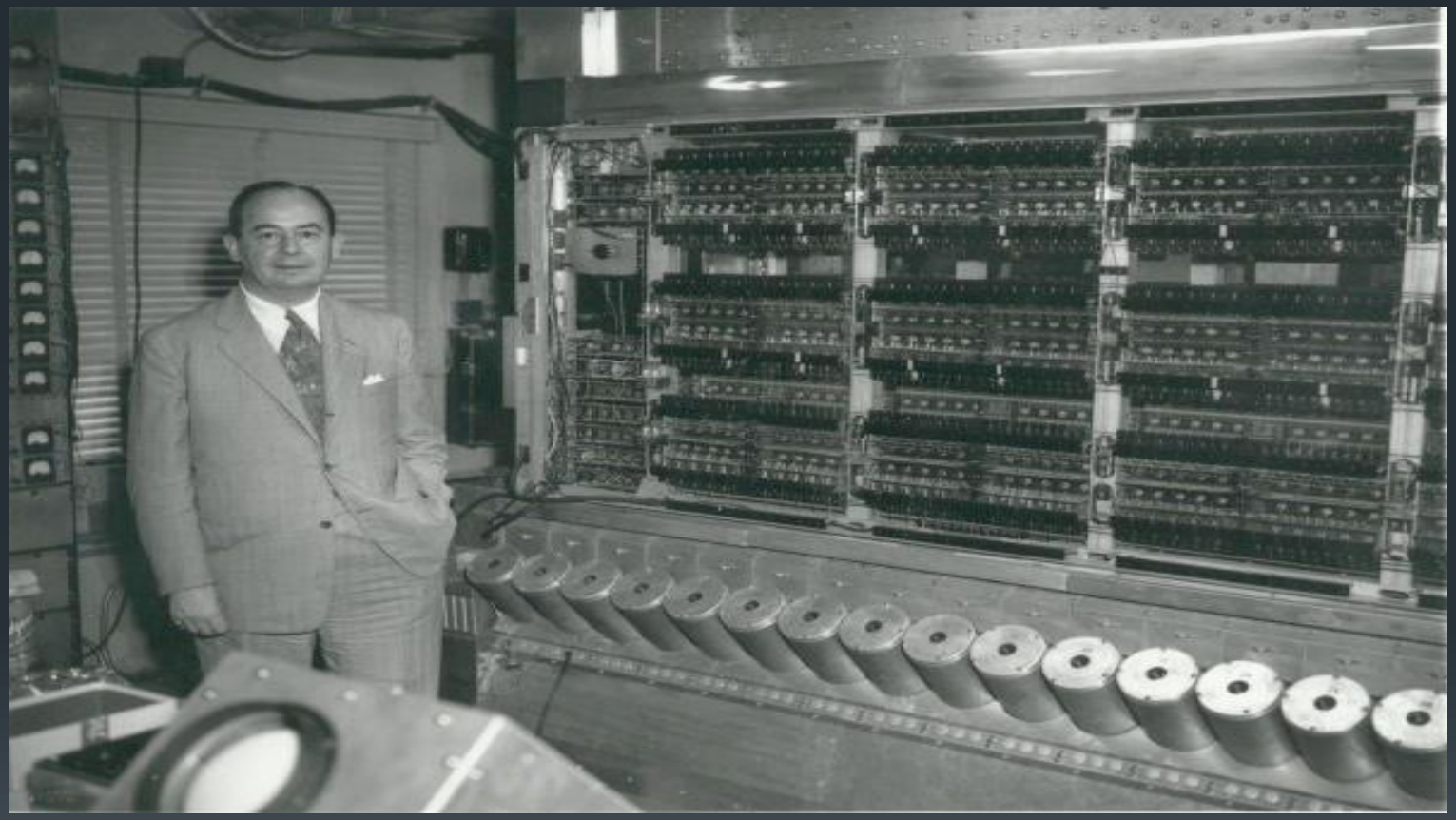
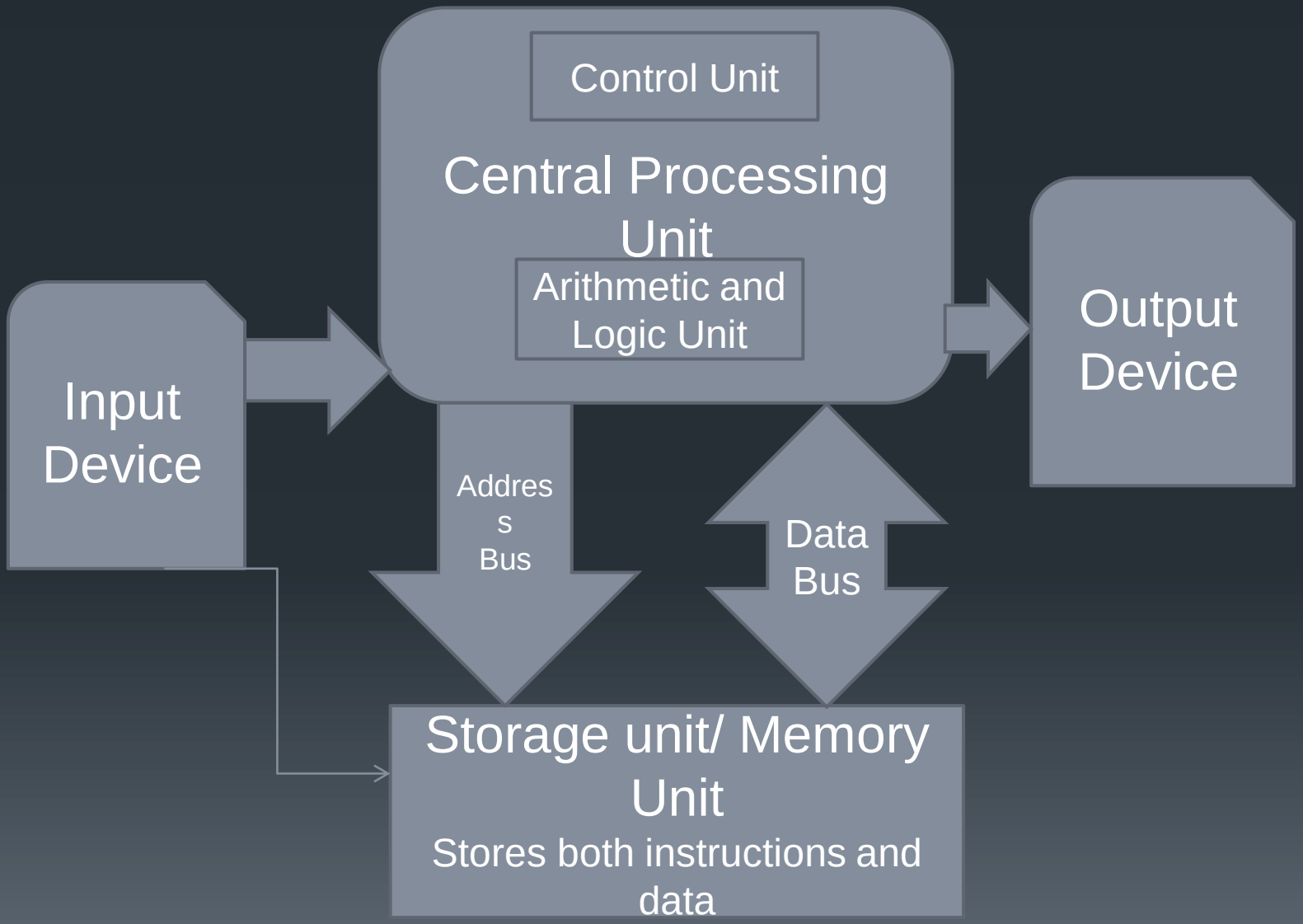


Image courtesy: www.medium.com

# Von Neumann Architecture



# Von Neumann architecture

- There are four sub-components in von Neumann architecture:
  - Memory
  - input and output devices
  - Arithmetic-Logic Unit
  - Control Unit
- There is a 5th key player in the architecture, i.e. a wire, or bus, that connects the components together
- Bus enables data flow from one sub-component to another.
- Three types of communication lines in the bus : address lines, data lines and control lines.

# Von Neumann Architecture

- Major principles
  - Data and instructions stored together in the storage unit / Memory unit.
  - Memory unit is a collection of locations and each location is designated with an address.
  - Central processing unit fetches the instruction and data from the memory unit.
  - The central processing unit process the data
  - Program and data cannot be accessed simultaneously

# The working of von Neumann model

- The central processing unit, decides the instruction to be executed next and keep its address in the address line i.e. address bus.
- The address bus is a uni-directional bus connected from central processing unit to memory unit.
- From the memory unit, the instruction or data  are placed into the data bus which carries data from the memory to the central processing unit.

# Working continues…

- The instructions gets decoded at the control unit.
- Processor executes the task according to the instruction.
- The arithmetic and logical operations are performed in the arithmetic and logic unit(ALU)
- The execution of program is a sequential process, other than branching instruction execution.
- The output of the execution may be stored in the memory or sent to the output unit.

# Types of instructions

- **Data Transfer Instructions** – transfers data from a source to a destination. The source and destination can be memory locations or registers that are associated with CPU.

  Ex: Mov A,R1

- **Arithmetic Instructions**–performs arithmetic operations such as addition, subtraction, multiplication, division etc. It is performed by ALU.

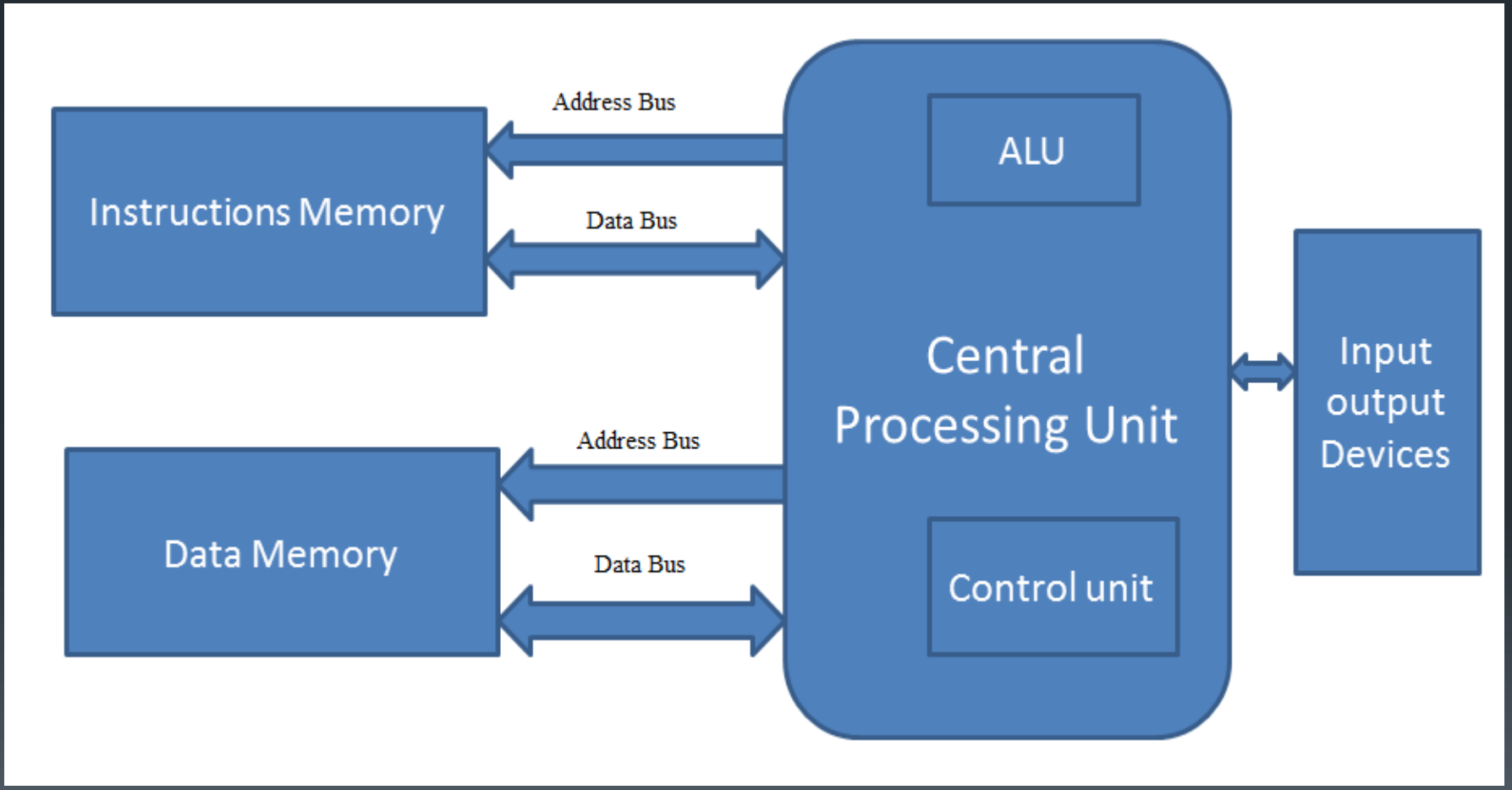
  Add R1,R2

- **Logical Instructions**- Performs logical operations

- **Control transfer instructions** – The flow of sequential execution can be changed with a branch instruction, it can be made according to any condition.

  Branch > 0

# Harvard Architecture



# Harvard Architecture

- Principles
  - In **Harvard architecture** concept, Memory for data was separated from the memory for instruction.
  - This concept supports the parallel access of data and instructions
  - Modern processors uses Harvard architecture

# Advantages of Harvard Architecture

- Separate data path and instruction path is available.
- Fetching of data and instructions can be done simultaneously
- Different  sized cells can be allowed in both the memories.

# Comparison of von Neumann and Harvard Architecture

| Von Neumann Architecture | Harvard Architecture |
|---|---|
| It is a conceptual design based on stored program architecture | It is a modern computer architecture |
| Memory is common for both instruction and data | Separate memory is there for instruction and data |
| Common bus between memory and CPU for both instruction and data | Separate buses for instruction memory and data memory |
| Two clock cycles required for the processor to execute an instruction | Only one clock cycle is enough for the processor to execute an instruction |
| Data transfer and instruction fetches cannot be done simultaneously | Data transfer and instruction fetches can be done simultaneously |
| Control unit design is simple | Control unit for two buses is more complicated which increses the development cost |

# Performance of computer

- Clock is used to synchronize the working of a unit
- Clock Cycle: Discrete time intervals at which the events happen in a computer system
  - The length of each clock cycle is considered as clock period.
  - A clock period is also called tick, clock tick etc.

# Performance matrix of a computer system

- Execution time: It is the time taken to finish a task
- - CPU execution time is a combination of user CPU time and system CPU time
- Throughput: It is defined as the total quantity of completed work in a specific period of time.

# Relation between the time of execution and performance

- For a given task, in order to maximize the performance of a computer, we need to minimize the time required for execution or response time.
- For a computer system A, the performance is calculated by the following formula

  *Performance of A= 1/Execution time of A*

  *i.e.* $\text{Performance}_A = 1/\text{Execution time}_A$

# Continued…

- i.e. we assume  two computers X and Y, the relation between the performances can be represented by the below formula, if the performance of X is more than the performance of Y.

$$\text{Performance}_X > \text{Performance}_Y$$

$$\frac{1}{\text{Execution time}_X} > \frac{1}{\text{Execution time}_Y}$$

$$\text{Execution time}_Y > \text{Execution time}_X$$

# Continued…

- Computer X is '*n*' times faster than computer Y

can be represented by the following formula,

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = n$$

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = n$$

# CPU performance and its factors

CPU execution time for a program =

CPU clock cycles for a program * Clock cycle time

$$\text{CPU execution time for a program} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

- The above formula is called performance equation
- It indicates that the CPU performance can be improved by reducing either the number of clock cycles required for a program or by reducing the length of the clock cycle.

# Performance Evaluation

- The gain in performance can be achieved by improving some resources of a computer. It can be calculated using Amdhal's law.

- *Amdhal's law states that, the performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used.*

# Amdhal's law

- Amdhal's law defines speed up

$$\text{Speedup} = \frac{\text{Performance for entire task using the enhancement when possible}}{\text{Performance for entire task without using the enhancement}}$$

Alternatively,

$$\text{Speedup} = \frac{\text{Execution time for entire task without using the enhancement}}{\text{Execution time for entire task using the enhancement when possible}}$$

$$\text{Execution time}_{new} = \text{Execution time}_{old} \times \left( (1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}} \right)$$

The overall speedup is the ratio of the execution times:

$$\text{Speedup}_{overall} = \frac{\text{Execution time}_{old}}{\text{Execution time}_{new}} = \frac{1}{(1 - \text{Fraction}_{enhanced}) + \frac{\text{Fraction}_{enhanced}}{\text{Speedup}_{enhanced}}}$$

# Sample Problem

- Consider a CPU used in Web servicing. We need to enhance the processor by increasing the computation speed 10 times faster on computation process in web service applications. We assume that, 30% of the time the original processor is spending for computation process and 70% of the time is waiting for the i/o devices. By incorporating the enhancement, what will be the overall speed up gain?

# Solution

- $Fraction_{enhanced}$=30% =0.3
- $Speed_{enhanced}$ =10
- $Speedup_{overall}$ = 1/(1-0.3)+(0.3/10)
  = 1/0.7+0.03
  =1/0.73
  ≈ 1.369

# References


1. Computer Architecture- A Quantitative Approach by John L. Hennessy, David A.Patterson
2. Computer Organization by Car-Hamacher,5th edition
3. Computer Organization and Architecture by william Stallings