Computer System Architecture

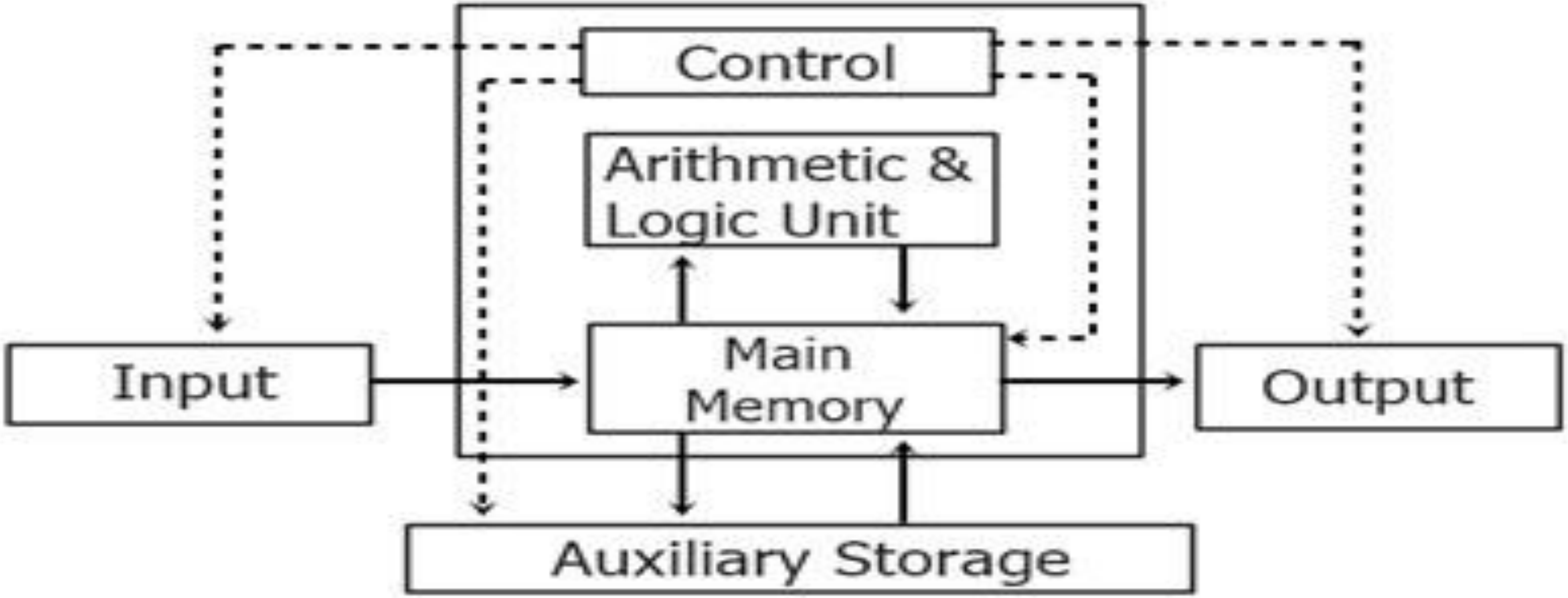
UNIT – 1

Basic Structure Of Computer Hardware

Syllabus

- Basic Structure of Computer Hardware
- Functional Units
- Computer Components
- Performance Measures
- Memory Addressing & Operations

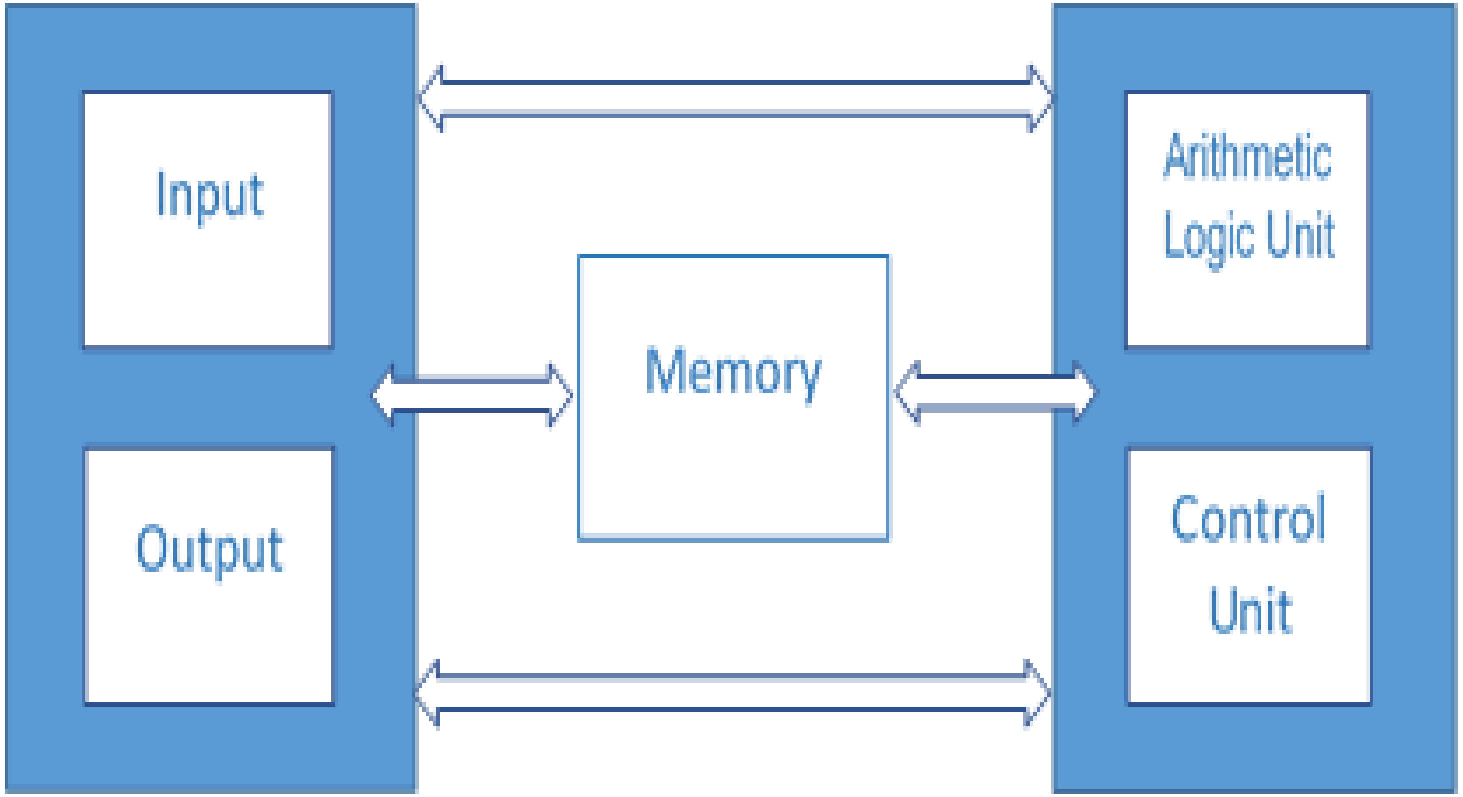
Basic Structure of Computer Hardware



Block Diagram of Computer

Functional Units

- Input Unit
- Output Unit
- Memory Unit
- Arithmetic & Logic Unit (ALU)
- Control Unit (CU)



Functional Units

Input Unit

- Computer accepts coded info through input unit
- Input can be from human operators, electromechanical devices or from other computers over network
- Example: keyboard, mouse, joystick, trackball

Output Unit

- Computer delivers processed output through output unit
- Output unit is the counterpart of the input unit. Its function is to produce processed results in human understandable format
- Example: plotter, graphic display, printer

Functional Units

Memory Unit

- Memory unit is used to store program as well as data
- It can be classified into 2 categories:
  - **Primary Storage**
    - Made up of semiconductor storage cells
    - Operates at high speed
    - Cells are grouped together and called as *word*
    - Normally, word size is 16/32/64 bits
    - Read/write of one word of bits into the memory as a single basic operation
    - Example: RAM, ROM, Cache memory, PROM, EPROM, Registers, etc
  - **Secondary Storage**
    - As primary storage is very expensive, comparatively cheaper secondary memory is used
    - Stores large amount of data/programs that are not in direct use
    - Example: hard disk, floppy disk, magnetic tapes, optical disks, etc

Functional Units

Central Processing Unit (CPU)

- Refer CSA_UNIT_3 ppt

Computer Components

- Computer Components: CPU, Main Memory, I/O devices, etc.
- Components can be used in any combination to suit specific purpose
- Components are interconnected in order to achieve the basic functionalities
- Describing function of computer = Describing function of each component = Data & Control signals exchanged between them
- Also important to describe interconnection structure and the control required to manage the interconnection
- Understanding of the top level structure and the function of its components gives insight into system bottlenecks, the magnitude and importance of system failures in case of a component failure, alternate pathways for smooth operation and ease of adding performance enhancements
- Requirements of greater system power and failsafe capabilities can be more easily achieved by design changes rather than by merely increasing speed and reliability of components.
- Virtually all recent computer designs are based on the three key concepts of the Von Neumann architecture:
    1. The main memory stores both data and instructions in binary format
    2. Every memory location can be individually addressed.
    3. The instructions so stored are executed one after another in a predefined sequence.

Performance Measures

- Speed of operation of a system is generally decided by:
  - Response time : Time spent to complete an event or operation (aka exec time or latency)
  - Throughput : Amount of work done per unit time (aka bandwidth)
- Faster response time leads to better throughput
- Elapsed time : Time spent from start of execution of a program till its completion
- Performance of a processor is measured only by considering the periods during which the processor is active
- Basic performance equation:
  - T = (N x S) / R, where
    - T= performance parameter;
    - N = no. of machine language instruction required to complete the execution of the program;
    - S = avg no. of basic steps required to execute one machine instruction;
    - R = clock rate of processor in cycles per second

Performance Measures

- Performance is inversely proportional to execution time
    - Performance = 1 / Execution time
- If a system X is N times faster than Y, then
    - N = $\text{Performance}_X / \text{Performance}_Y$
    - N = $\text{Execution Time}_Y / \text{Execution Time}_X$

Amdahl's Law

- Speedup =

$$\frac{\text{Performance of entire task using the enhancement when possible}}{\text{Performance of entire task without using the enhancement}}$$

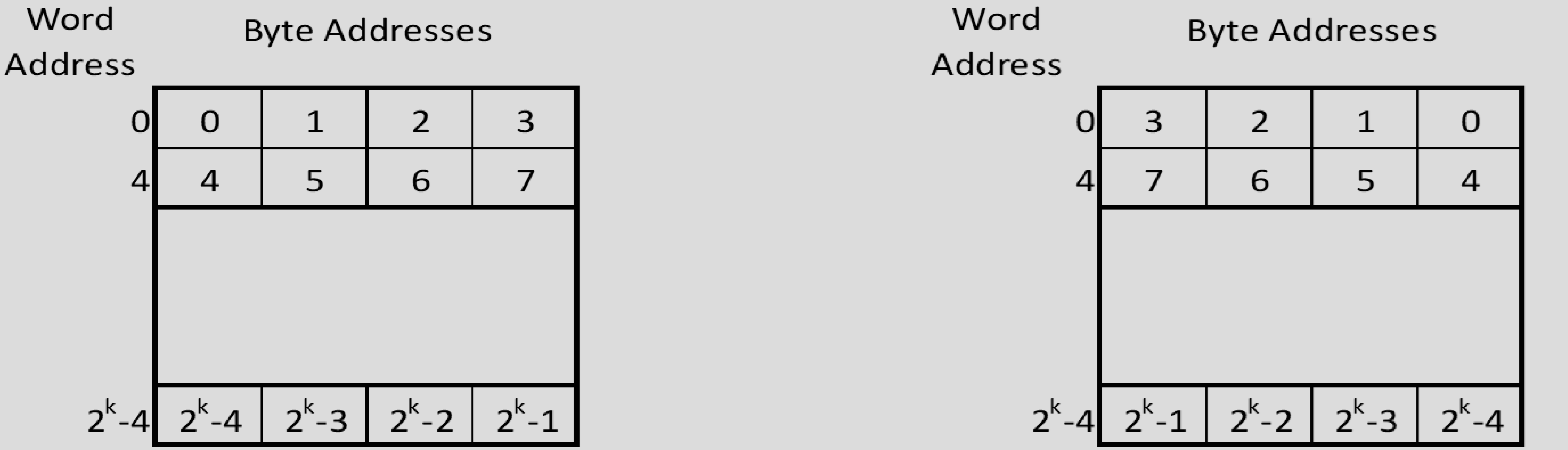
$$= \frac{\text{Exec time for entire task without using enhancement}}{\text{Exec time for entire task using the enhancement when possible}}$$

Memory Addressing

- Maximum size of the memory that can be used in any computer is determined by the addressing scheme
- Example
  - A 16-bit computer that generates a 16-bit address is capable of addressing up to 2^16 memory locations
  - A 32-bit computer that generates a 32-bit address is capable of addressing up to 2^32 memory locations
  - A 64-bit computer that generates a 64-bit address is capable of addressing up to 2^64 memory locations
  - A 40-bit computer that generates a 40-bit address is capable of addressing up to 2^40 memory locations
- The number of locations represents the size of the address space of the computer
- Most modern computers are byte-addressable

Memory Addressing

32-bit addressing



Word Address	Byte Addresses			
0	0	1	2	3
4	4	5	6	7
2^k-4	2^k-4	2^k-3	2^k-2	2^k-1

Big-endian assignment

Word Address	Byte Addresses			
0	3	2	1	0
4	7	6	5	4
2^k-4	2^k-1	2^k-2	2^k-3	2^k-4

Little-endian assignment

Memory Addressing

- The name 'big-endian' is used when lower byte addresses are used for more significant bytes (i.e. the leftmost bytes) of the word.
- The name 'little-endian' is used for the opposite ordering, where the lower byte addresses are used for less significant bytes (i.e. the rightmost bytes) of the word
- The words ‘more significant' and 'less significant' are used in relation to the weights (power of 2) assigned to bits when the word represents a number
- Both big-endian and little-endian assignments are used in commercial machines. In both cases, byte addresses 0, 4, are taken as the addresses of successive words in the memory and are the addresses used when specifying memory read and write operations for words.
- The big-endian arrangement is used in the 68000 processor. The little-endian arrangement is used in Intel processors. The ARM architecture can be configured to use either arrangement
- As far as the memory structure is concerned, there is no substantial difference between the two schemes.

Memory Operations

R/W signal

- Sets high signal for read and low signal for write

MFC(Memory Function Completed) signal

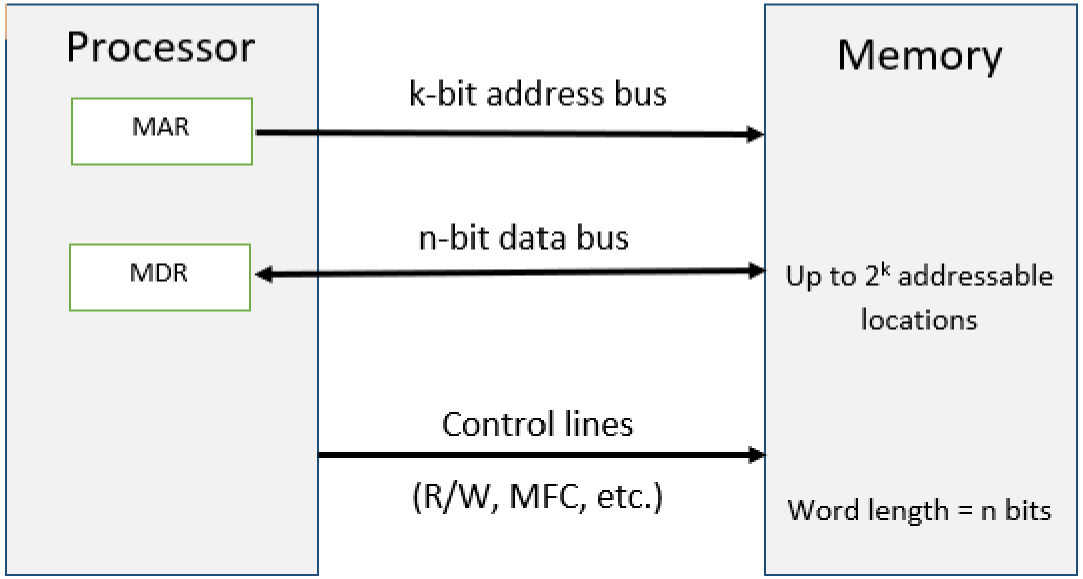
- Sets MFC to high once the memory operation is complete

Memory *write* operation

- The *word* to be stored into the memory location is first loaded by the CPU into MDR
- *Address* of the location into which the word is to be stored is loaded by the CPU into MAR
- A write signal is issued by the CPU.

Memory *read* operation

- *Address* of the location from which the word is to be read is loaded into the MAR
- A read signal is issued by the CPU.
- The required word will be loaded by the memory into the MDR ready for use by the CPU.



Processor and Memory Connection