KIIT POLYTECHNIC

# LECTURE NOTES

## ON

# Computer System Architecture

## Compiled by

**Abhaya Kumar Panda**
**Lecturer, Department of Computer Science & Engineering,**
**KIIT Polytechnic, Bhubaneswar**

# COURSE OUTCOMES

**CO1:** Explain a comprehensive understanding of the fundamental architecture of computer systems, including basic hardware components, instruction sets, addressing modes, and memory organization.

**CO2:** Explain the complete instruction execution process, including register files, hardware control, and memory management techniques such as paging and virtual memory

**CO3:** Illustrate the principles and techniques of input/output operations, including various I/O transfer modes, interrupt-driven I/O, DMA, and I/O processor concepts.

**CO4**: Analyze the concept of parallel processing, including linear pipelines, multiprocessors, and Flynn's classification.

## MAPPING OF COs with POs/PSOs

| COID POID→ ↓ | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PSO1 | PSO2 |
|---|---|---|---|---|---|---|---|---|---|
| **CO1** | 3 | 2 | | | | | | 3 | |
| **CO2** | 3 | 3 | 2 | | | | | 3 | 1 |
| **CO3** | 2 | 3 | 3 | | | | | 3 | |
| **CO4** | 3 | 2 | 2 | 3 | | 2 | 2 | 3 | 2 |

# CONTENTS

# CHAPTER-1

## Basic structure of computer hardware

### Computer Architecture

*In **computer** engineering, **computer architecture** is a set of rules and methods that describe the functionality, organization, and implementation of **computer** systems*

### Functional unit

- A computer consists of five functionally independent main parts input, memory, arithmetic logic unit (ALU), output and control unit.
- Input device accepts the coded information as source program i.e., high level language.
- This is either stored in the memory or immediately used by the processor to perform the desired operations.
- The program stored in the memory determines the processing steps.
- Basically, the computer converts one source program to an object program. i.e., into machine language.
- Finally, the results are sent to the outside world through output device. All of these actions are coordinated by the control unit.

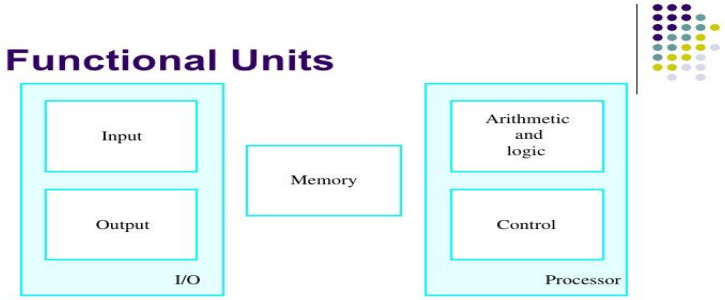



Figure 1.1. Basic functional units of a computer.

**Input unit: -**

- The source program/high level language program/coded information/simply data is fed to a computer through input devices keyboard is a most common type.
- Whenever a key is pressed, one corresponding word or number is translated into its equivalent binary code over a cable & fed either to memory or processor.
- Example –Keyboard, Joysticks, trackballs, mouse, scanners etc are other input devices.

**Memory unit: -**

Its function is to store programs and data.

It is basically to two types 1. Primary memory 2. Secondary memory

**Primary memory: -**

- Is the one exclusively associated with the processor and operates at high speed.
- The memory contains a large number of semiconductors storage cells.
- These are processed in a group of fixed size called word.

- Programs must reside in the memory during execution. Instructions and data can be written into the memory or read out under the control of processor.
- **Secondary memory: -**
- This type of memory is used where large amounts of data & programs have to be stored, particularly information that is accessed infrequently.
- Examples: - Magnetic disks & tapes, optical disks (i.e., CD-ROM's), floppies etc.

**Arithmetic logic unit (ALU): -**
Most of the computer operation are executed in ALU of the processor like addition, subtraction, division, multiplication, etc. The operands are brought into the ALU from memory and stored in high-speed storage elements called register.

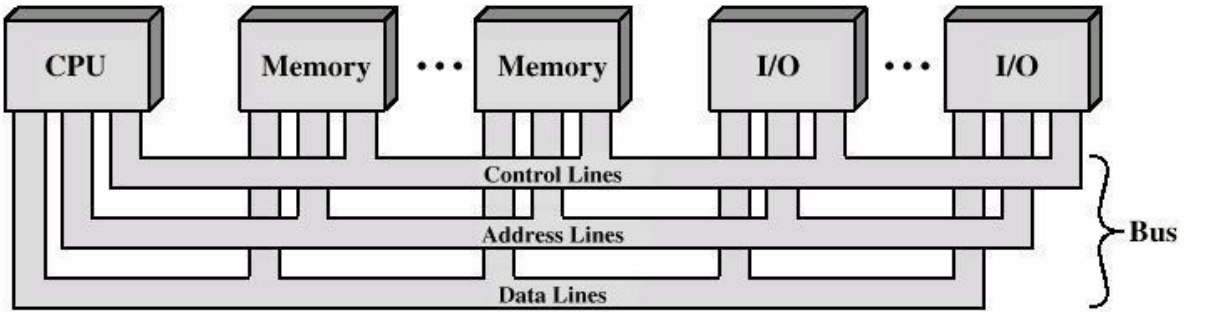
**Control unit: -**
The operations of all the units are coordinated by the control unit i.e., it acts as a nerve centre that sends signals to other units and senses their states. The actual timing signals that govern the transfer of data between input unit, processor, memory and output unit are generated by the control unit.

**Output unit:**
These are actually are the counterparts of input unit. Its basic function is to send the processed results to the outside world after processing. Examples: - Printer, speakers, monitor etc.

## Bus Structure

A bus is a communication system that transfers information (in any form like data, address or control information) between components, inside a computer, or between computers.

- It is the group of wires carrying a group of bits in parallel.
- There are three kinds of bus according to the type of information they carry like
    1. Data Bus
    2. Address Bus
    3. Control Bus
- A bus which carries a word from or to memory ate called Data bus. It carries the data from one system module to other. Data bus may consist of 32,64, 128 or even more numbers of separate lines. This number of lines decides the width of the data bus. Each line can carry one bit at a time. So, a data bus with 32 lines can carry 32bit at a time.
- Address Bus is used to carry the address of source or destination of the data on the data bus.
- Control Bus is used to control the access, processing and information transferring.




- In this method bus architecture, the processor will completely supervise and participate in the transformation.
- The information will be first taken to the processor register and then to the memory such that transfer is known as program-controlled transfer.
- The interconnection between i/o unit, processor and memory accomplished by two independent system bus is known as two-way bus interconnection structure.
- The system bus between i/o unit and processor consist of DAB (Device address bus), DB (Data bus), CB (Control bus). Similarly, the system bus between memory processor consists of MAB (Memory address bus), DB, CB.
- The communication exists between
  - ✓ Memory to processor
  - ✓ Processor to memory
  - ✓ I/o to processor
  - ✓ processor to i/o
  - ✓ I/o to memory

## Performance measures: -

Performance is the ability of the computer to quickly execute a program.

- The speed at which the computer executes a program is decided by the design of its hardware and machine language instruction.
- Computer performance measures is of very big term when used in context of the computer system.
- System that executes program in less time are called to have higher performance.

**Basic performance measures: -**

The speed of operation of a system is generally decided by two fractions.

1. Response time
2. Throughput.

**Response Time: -**

- Response time is the time spend to complete an event or an operation.
- It is also called as execution time or latency.

**Throughput: -**

Throughput is the amount of work done per unit of time. i.e., the amount of processing that can be accomplished during a given interval of time.

- It is also called as bandwidth of the system.
- In general, faster response time leads to better throughput.

**Elapsed time: -**

- Elapsed time is a time spent from the start of execution of the program to its completion is called elapsed time.
- This performance measure is affected by the clock speed of the processor and the concerned input output device.

**MIPS**

- A nearly measure of computer performance has the rate at which a given machine executed instruction.
- This is calculated by dividing the no. of instruction and the time required to run the program.

**CPI/IPC**

- CPI – Clock cycle per Instruction, IPC – Instruction per cycle.
- It is another measuring that which is calculated as the number of clock cycle required to execute one instruction (cycle per instruction) by the instruction executed per cycle.

**Speedup: -**

- Computer architecture use the speed up to describe the performance of architectural changes as different improvement are made to the system.
- It is defined as ratio of execution time before to the execution time after the charge.
- Speed up = execution time before /Execution time after

**Amdahl's law: -**

- This law states that "performance improvement to be gained by using a faster mode of execution is limited by the fraction of time the faster mode can be used".
- Amdahl's law defines the term speed up.
- Speed up = performance of entire task using enhancement/ Performance of entire task without using enhancement
- Performance = 1 / Execution time
- Speed up = execution time without using enhancement/Execution time with using enhancement
- Factors affecting speedup are as follows:

  1) The fraction of computation time in the original machine can be modified to use the advantage of the enhancement. This is called fraction enhanced which is always less than or equal to one.

  $$\text{Fraction enhanced} \leq 1$$

  2) Improvement granted by the enhanced execution made is the speed with which the task could run faster using the enhancement.

  $$\text{Speed up} > 1$$

- Speed up enhanced = time in original mode /Time in enhance mode
- Ex. Let us a program takes 5 second in enhanced mode while it takes 10 second earlier. So, speedup enhanced = 10/5 = 2

- The new execution time can be calculated as follow:
- Execution time new = Execution time original X ((1-fraction enhanced) + fraction enhanced/speedup enhanced)
- The speedup overall = execution time original/execution time new
- Speedup overall = 1/ ((1-fraction enhancement) +fraction enhance/Speedup enhance)

**Performance parameter**

- The basic performance equation is given by T =(NxS)/R
- Where T– Performance parameter of an application program.
- N – No. of instruction required to complete the execution of a program.
- S – Average no of steps to execute an instruction.
- R-Clock rate of the processor in cycles per second.

**Clock rate: -**

- Clock rate is one of the important performance measures by improving in the clock rate. There are two ways in which clock rate may be increased.
- Improving IC technology which makes logic circuits faster thus reading time taken to complete a basic step.
- By reducing the processing amount in one basic step which by reduces the clock period as R = 1/T.

**CPU performance Equation: -**

- Normally the CPUs are constructed by using a clock running at a constant rate. This discrete time event is known as a clock cycle.
- CPU time = CPU clock cycle for a program x clock cycle time
- = CPU clock cycle/ clock rate
- IC X CPI X C

**Pipelining and parallel processing**

The performance of the system can be improved substantially by overlapping the execution of successive instruction. This technique is called as pipelining. Parallel processing can also be implemented to achieve the high performance instead of traditional sequential processing.

## Memory addressing

The maximum size of the memory that can be used in any computer is determined by the addressing scheme.

For ex- a 16-bit computer that generates a 16-bit address is capable of addressing up to $2^{16}$ k memory location.

The no of location represents the size of the address space of computer.

Most modern computers are byte addressable computer.

# CHAPTER- 2

## Instructions & instruction Sequencing

## Introduction

The instruction set defines many of the function performed by the CPU and plays a significant role in the implementation of the CPU.

The factors to be taken into account while designing an instruction set are:

1. Operation Repertoire

   This gives an idea how many & what kind of operation need to be provided and also the complexity of such operation.

2. Data type

   Information needs to be provided on the various types of data & operation to be performed on them.

3. Format of instruction

   This includes the length of the instruction in bits, number of addresses to be used with the instruction and the size of each field in the instruction.

4. Register

   The number of CPU register that can be accessed by instruction for storage of data & operands.

5. Addressing mode

   The instruction set also specifies addressing methods for accessing operands either in the memory or in the processor register.

### Types of Operands

Machine operation depend on the types of data being processed.

The different format of data to be used with assembly and high-level language program are as follows

1.Address

2.Number

3.Character

4.Logical data

Address

- Address is a form of numbers that represent specific location in the memory.
- Address may be considered as unsigned integer.

Number

- Numeric data types are used by all machine language.
- Three types of numerical data are commonly used :
  1. Integer or Fixed point
  2. Floating point
  3. Decimal

Character

Another common form of data is represented in a program are character string. As it cannot be stored or processed so these form of data are coded by any of the conversion method like IRA,ASCII,EBCDIC.

Logical data

- This is the bit-oriented view of data.
- These types of data can be represented as an array of Boolean or binary data items (1 for T and 0 for F).

## Addressing Modes

- The operand field of an instruction specifies the address from where the data has to be fetched.
- This may be a memory address , register or may be a direct value.
- The operand chosen is dependent on the addressing mode of the instruction.

The most common addressing techniques are

- Immediate
- Direct
- Indirect
- Register
- Register Indirect
- Displacement
- Stack

### Immediate Addressing:

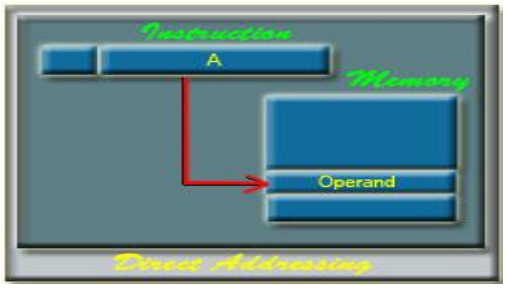
The simplest form of addressing is immediate addressing, in which the operand is actually present in the instruction:

OPERAND = A

The advantage of immediate addressing is that no memory reference other than the instruction fetch is required to obtain the operand.


**Direct Addressing:**

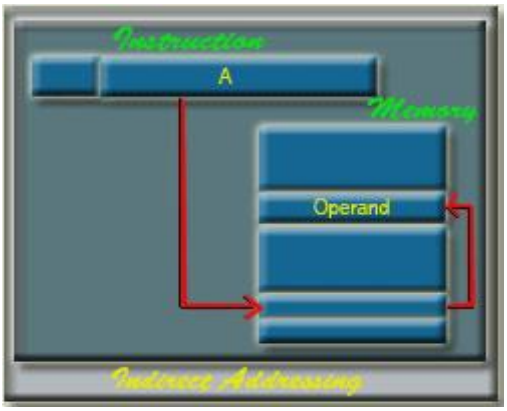
- A very simple form of addressing is direct addressing, in which the address field specifies the address of the memory location.
- It requires only one memory reference and no special calculation.




Ex- LDA 2500

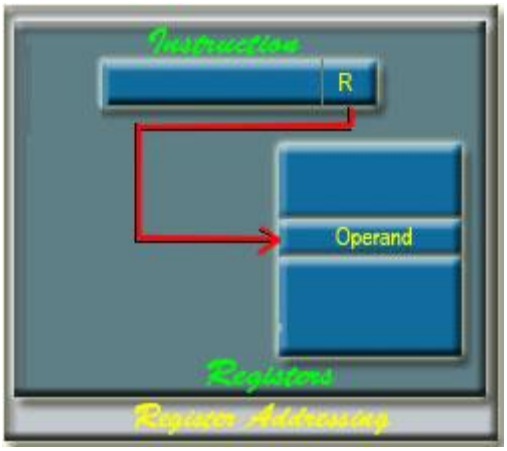
**Indirect Addressing:**

- The address field of the instruction uses the address of the operand stored in the memory.
- The effective address of the operand is given by the address part of the instruction.
  EA = (A)



**Register Addressing:**

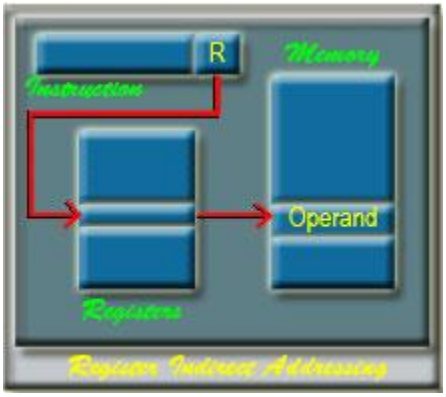
- Register addressing is similar to direct addressing.
- The only difference is that the address field refers to a register rather than a main memory address




Ex- MOV A B

**Register Indirect Addressing:**

Register indirect addressing is similar to indirect addressing, except that the address field refers to a register instead of a memory location.



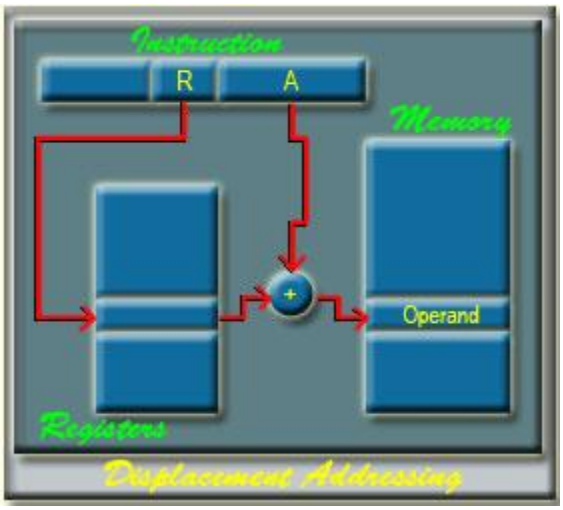

**Displacement Addressing:**

A very powerful mode of addressing combines the capabilities of direct addressing and register indirect addressing, which is broadly categorized as displacement addressing:

EA = A + (R)

Three of the most common use of displacement addressing are:

- Relative addressing
- Base-register addressing
- Indexing




**Relative Addressing:**

- In this mode the instruction specifies the operand address as the relative position of the current instruction address i.e., content of PC.
- The current instruction address is added to the address field to produce EA.

**Base-Register Addressing:**

In this mode the reference register contains a memory address, and the address field contains a displacement from that address.

**Index addressing:**

- In this mode an index register contains the offset value.
- The instruction contains the address that should be added to the offset in the index register to get the effective address.

## Instruction format

An **instruction format** defines the layout of the bits of an **instruction** in an instruction set.

Usually there are several instruction formats in an instruction set.

1. **An operation code:-**This specifies the operation to be performed.
2. **Source operand** :- The operand specified by the opcode may involve one or more sources for operand, these act as the input for the operation.
3. **Resultant operand** :-After processing the result must have to be stored in an

operand.

4. **Next reference instruction** :-This field tells the CPU from where the next instruction is to be fetched after the execution is completed.

**Three address Instruction**

Computer with three addresses instruction format can use each address field to specify either processor register or memory operand.

| | | | |
|---|---|---|---|
| ADD | R1, A, B | R1 ® M [A] + M [B] | |
| ADD | R2, C, D | R2 ® M [C] + M [D] | X = (A + B) * (C + A) |
| MUL | X, R1, R | M [X] R1 * R2 | |

- The advantage of the three address formats is that it results in short program when evaluating arithmetic expression.
- The disadvantage is that the binary-coded instructions require too many bits to specify three addresses.

**Two Address Instruction**

- Most common in commercial computers.
- Each address field can specify either processes register or a memory word.

| | | | |
|---|---|---|---|
| MOV | R1, A | R1 ® M [A] | |
| ADD | R1, B | R1 ® R1 + M [B] | |
| MOV | R2, C | R2 ® M [C] | X = (A + B) * (C + D) |
| ADD | R2, D | R2 ® R2 + M [D] | |
| MUL | R1, R2 | R1 ® R1 * R2 | |
| MOV | X, R1 | M [X] ® R1 | |

**One Address instruction**

It used an implied accumulator (AC) register for all data manipulation. For multiplication/division, there is a need for a second register.

| | | | |
|---|---|---|---|
| LOAD | A | AC ® M [A] | |
| ADD | B | AC ® AC + M [B] | |
| STORE | T | M [T] ® AC | X = (A +B) × (C + A) |

- All operations are done between the AC register and a memory operand.
- It's the address of a temporary memory location required for storing the intermediate result.

| | | |
|---|---|---|
| LOAD | C | AC ® M (C) |
| ADD | D | AC ® AC + M (D) |

| | |
|---|---|
| ML T | AC ® AC * M (T) |
| STORE X | M [X] ® AC |

**Zero Address Instruction**

A stack organized computer does not use an address field for the instruction ADD and MUL. The PUSH & POP instruction, however, need an address field to specify the operand that communicates with the stack (TOS ® top of the stack)

| | |
|---|---|
| PUSH A | TOS ® A |
| PUSH B | TOS ® B |
| ADD | TOS ® (A + B) |
| PUSH C | TOS ® C |
| PUSH D | TOS ® D |
| ADD | TOS ® (C + D) |
| MUL | TOS ® (C + D) * (A + B) |
| POP X | M [X] TOS |

# CHAPTER- 3

## Processor System

INTRODUCTION

- The part of the computer that performs the bulk of data processing operation is called the central processing unit, CPU.
- The CPU is made up of three major parts-
    1. ALU: - Arithmetic Logic Unit performs the required microoperation for executing the instruction.
    2. Register Set: - Stores the intermediate data used during the execution of instruction.
    3. Control Unit: - Supervises the transfer of information among register and ALU by sending suitable control signal.

## Control unit

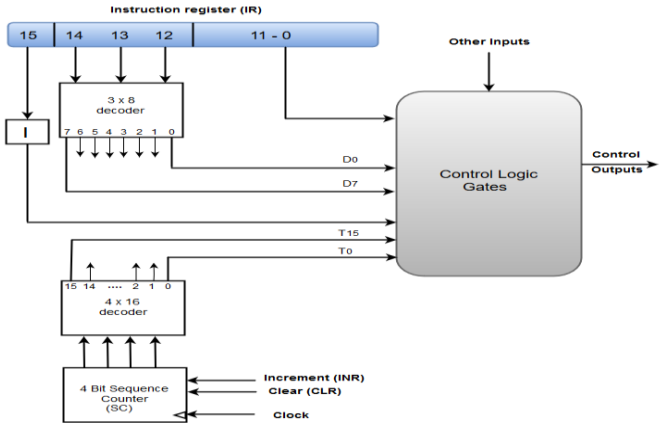
The Control Unit (CU) is the heart of the CPU. Every instruction that the CPU supports has to be decoded by the CU and executed appropriately. Every instruction has a sequence of microinstructions behind it that carries out the operation of that instruction

- Any digital system consists of 2 units
    - ✓ Data processor
    - ✓ Control logic
- Data processor consists of individual register and all functional unit.
- Control logic initiates all micro-operation in the data processor.
- Control unit generates control signal which initiates sequence of micro-operation.
- By issuing control signal which micro-operation are activated are represented as sequence of 1s and which are not activated represented as 0s.
- Traditionally there are two general approaches to implement a control and decode unit for a CPU :
    - ✓ Hardwired control Unit
    - ✓ Micro programmed control Unit

### Hardwired control unit

- In the hardwired organization, the control logic is implemented with gates, flipflop, decoder and other digital circuits.
- It has the advantage that it can be optimized to produce a fast mode of operation.
- A hardwired control for all the basic computer is shown in the above figure.
- The hardwired control unit consists of 2 decoders, a sequence counter and a number of control logic gates.

**Control Unit of a Basic Computer:**




- The instruction fetched from memory is placed in IR(instruction register).The IR is divided into 3 parts. The I bit, the operation code and bits 0-11.
- The opcode 12-14 are decoded with a 3×8 decoder. the eight o\p of the decoder are designated by the symbol $D_0$-$D_7$.
- The subscripted decimal is equivalent to the binary value of the corresponding operation code.
- Bits 0-11 are applied to the control logic gates.
- The 4-bit SC (sequence counter) is decoded into 16 timing signals through $T_0$-$T_{15}$.
- The SC can be incremented/cleared synchronously.
- All these inputs like 0-11 of IR, decoded opcode from $D_0$-$D_7$,the o/p of I flipflop, the timing signal $T_0$-$T_{15}$ and other input are given to control logic gates.
- These control logic gates decide the states of I/p and provide the appropriate control signal and timing signal.
- Depending on this control signal, the ALU takes the appropriate action.

## Micro programmed control unit

- Every instruction in a CPU is implemented by a sequence of one or more sets of concurrent micro-operation.
- Each micro-operation is associated with a specific set of control lines which when activated cause the micro-operation to take place.
- Here to generate the control signal, the CU execute a series of sequential steps of micro-operation.
- The control variable at any given time can be represented by a string of 1s or 0s called control word.
- In micro programmed the binary control words are stored in memory called control memory.
- The computer which employs micro programmed control possess 2 separate memories :1)Main memory 2)Control memory

- Main memory is available for storing program. content of main memory may alter by changing the program.
- Each machine instruction in main memory initiates a series of micro instruction in control memory.
- Microinstruction generates micro-operation such as fetch instruction from main memory, calculate effective address, fetch operand, execute the operation etc.
- Each control word in control memory contains within it a microinstruction.
- A sequence of microinstruction constitutes a micro program.
- For a particular control signal a particular micro program is written, since there is no need to change the micro program stored in control memory, the control memory can be stored in ROM.

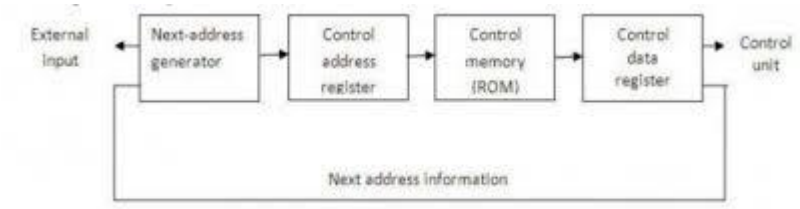



**figure 4.1: Micro-programmed control organization**

- So, to execute a micro program stored in control memory to get the appropriate control word for generation of control signal, the following operation is to be done.
- The control memory address specifies the address of micro instruction & the control data register holds the microinstruction read from memory.
- The microinstruction contains a control word that specifies one or more micro-operation for the data processor, once these operations are executed , the control must determine the next address.
- The next address may also be a function of external input condition.
- The next address generator is a circuit that generates the next address which is then transferred into the control address register to read the micro instruction.
- The next address generator sometimes called as micro program sequencer.

## REGISTER FILES: -

- Most modern CPUs have a set of general purpose (GPRs) register R0 to Rn-1 called register files.
- Each register Rm in RF is individually addressable with address subscript m.
  Example: - R2=f (R1, R2)
- This way the processor is able to retain intermediate results in fast and accessible registers rather than external memory M.
- For accessing RF needs several ports for simultaneous reading and writing purposes, so it is often realized as a 'multiport RAM.'
- A multiport RF is built using a set of registers of proper size and multiplexer-demultiplexer.

- The read operation can take place through several devices reading from the same register using different ports though the writing is normally done through one port only.
- The above RF shows a three port, where simultaneous read can occur from port A and port B and writing takes place using port C.

## Complete instruction Execution

The CPU executes each instruction in a series of small steps:

1. fetch the next instruction from memory the instruction register (IR).
2. change the program counter (PC) to point to the following instruction.
3. Decode the instruction just fetched.
4. If the instruction uses data in memory determine where they are.
5. Fetch the data if any, into internal CPU register.
6. Execute the instruction.
7. Store the result in the appropriate place.
8. Go to step 1 to begin executing the following instruction.

This above sequence of steps (micro-operation) is frequently referred to as the **fetch-decode-execute cycle** or **instruction cycle.**

During an instruction cycle, the action of the CPU is defined by the sequence of microoperation it executes.

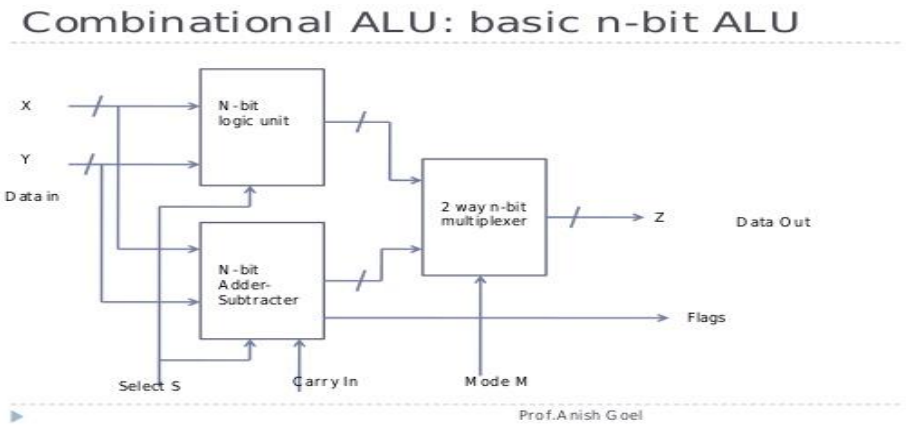
The time required by the CPU to execute a microoperation is the CPU cycle time or clock period.

## DESIGN OF ALU: -

- The circuits which carry out the data processing instructions, is the ALU.
- The complexity of the ALU depend upon how the instruction are realized.
- The ALU using combinational circuits can perform fixed point arithmetic as well as word based logical operation.
- Some extra control logic and some extensive data processing circuits called as co-processor are employed to perform floating point operation.

## COMBINATIONAL ALU: -

- The simple ALU combines the both features of 2's complement added subtractor and word-based logic unit.
- The combinational ALU is nothing but a combination of combinational circuits and multiplexer.




## The above figure shows that: -

- Mode control line attached to the two-way n bit multiplexer determines the type of operation i.e., logical or arithmetic.
- The select line S determines the specific operation to be performed by the desired sub unit.
- As the select line is of 4 bits, we can get 16 different logical and also 16 different arithmetic operation.
- Example: - by taking the values of M, the four combinations are

  $M_3 = x_i y_i$ $M_2 = x_i \overline{y}_i$ $M_1 = \overline{x}_I y_i$ $M_0 = \overline{x}_i \overline{y}_i$

  $F(x, y) = M_3S_3 + M_2S_2 + M_1S_1 + M_0S_0$
  $= x_i y_i S_3 + x_i \overline{y}_i S_2 + \overline{x}_I y_i S_1 + \overline{x}_i \overline{y}_i S_0$
- So, for every combination of S3, S2, S1 and S0 we will get a different operation.

ADVANTAGE: -

This type of ALU is much simpler.

DISADVANTAGE: -

1. It is more expensive.
2. It is much slower.

# CHAPTER- 4

## Memory System

## Memory characteristics

The memory unit is the essential component in any computer since it is needed for storing the program and data.

The memory system is classified according to their key characteristics like :

1. Location
2. Capacity
3. Unit of Transfer
4. Access Method
5. Performance
6. Physical type
7. Physical characteristics

### 1. Location:

It deals with the location of the memory device in the computer system. There are three possible locations:

- CPU: This is often in the form of CPU registers and small amount of cache
- Internal or main: This is the main memory like RAM or ROM. The CPU can directly access the main memory.
- External or secondary: It comprises of secondary storage devices like hard disks, magnetic tapes. The CPU doesn't access these devices directly. It uses device controllers to access secondary storage devices.

### 2. Capacity:

The capacity of any memory device is expressed in terms of: i)word size ii)Number of words

**Word size:** Words are expressed in bytes (8 bits). A word can however mean any number of bytes. Commonly used word sizes are 1 byte (8 bits), 2bytes (16 bits) and 4 bytes (32 bits).

**Number of words:** This specifies the number of words available in the particular memory device. For example, if a memory device is given as 4K x 16.This means the device has a word size of 16 bits and a total of 4096(4K) words in memory.

### 3. Unit of Transfer:

It is the maximum number of bits that can be read or written into the memory at a time. In case of main memory, it is mostly equal to word size. In case of external memory, unit of transfer is not limited to the word size; it is often larger and is referred to as blocks.

### 4. Access Methods:

It is a fundamental characteristic of memory devices. It is the sequence or order in which memory can be accessed. There are three types of access methods:

- **Random Access:** If storage locations in a particular memory device can be accessed in any order and access time is independent of the memory location being accessed. Such memory devices are said to have a random-access mechanism. RAM (Random Access Memory) IC's use this access method.
- **Serial Access:** If memory locations can be accessed only in a certain predetermined sequence, this access method is called serial access. Magnetic Tapes, CD-ROMs employ serial access methods.
- **Semi random Access:** Memory devices such as Magnetic Hard disks use this access method. Here each track has a read/write head thus each track can be accessed randomly but access within each track is restricted to a serial access.

**5. Performance:**

The performance of the memory system is determined using three parameters

- **Access Time:** In random access memories, it is the time taken by memory to complete the read/write operation from the instant that an address is sent to the memory. For non-random access memories, it is the time taken to position the read write head at the desired location. Access time is widely used to measure performance of memory devices.
- **Memory cycle time:** It is defined only for Random Access Memories and is the sum of the access time and the additional time required before the second access can commence.
- **Transfer rate:** It is defined as the rate at which data can be transferred into or out of a memory unit.
  - ✓ For a random-access memory, it is equals to 1.
  - ✓ For a non-random access memory, it can be calculated as-

    TN=TA+N/R Where TN-Avg. time to read or write

    TA-Average access time

    N-Number of Bits

    R-Transfer rate in bps

**6. Physical type**: Memory devices can be either semiconductor memory (like RAM) or magnetic surface memory (like Hard disks).

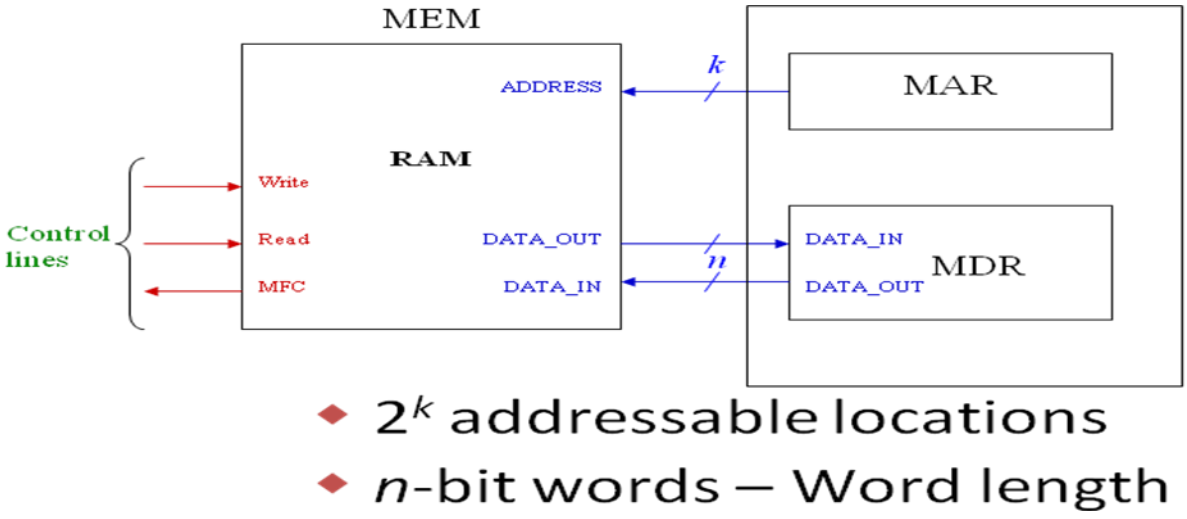
**7.Physical Characteristics:**

- **Volatile/Non- Volatile:** If a memory device continues hold data even if power is turned off. The memory device is non-volatile else it is volatile.

## Memory –Processor Data transfer/ Basic Memory Operation

- The memory unit supports two basic operations: read and write. The read operation reads previously stored data and the write operation stores a new value in memory.

- Both of these operations require a memory address. In addition, the write operation requires specification of the data to be written.
- The address and data of the memory unit are connected to the address and data buses of the system bus, respectively.
- The read and write signals come from the control bus.
- For controlling the movement of these words i.e., in and out two signals are used to
    - Write
    - Read respectively
- The words to be written or moved in are first enter to the register is called memory data register (MDR).
- The location in the memory unit when a word is stored is called as address of the word. To take out or retrieve a word from a memory unit one has to specify its address in another special register which is called as memory address register (MAR).
- MAR of R bit especially the memory size is of 2k word. Similarly, MDR of n bit determines the word size is off n bit (n number of cells present in a word).




## Read operation

- For Read operation the address has to be sent to MAR which is being carried out by the address bus then the READ signal is sent to the memory for read function.
- The memory transfers the corresponding word from the specified location to the MDR through data bus.
- Drop the memory read control signal to terminate the read cycle.

- After the completion of the Read Operation the memory transfer MFC (Memory function completed signal).

## Write operation

- For write operation the CPU specify the location to MAR and data to MBR/MDR.
- Then it transfers the Write control signal after which the data which is present in MDR is transfer to the memory.
- After completion of the write operation memory transfer MFC signal.

## Access time

The duration of time between the initiation of read signal and the availability of required word in the MBR is called as the access time or read time.

## Write time

The duration between the write signal and storing of the word in the specified location is called as write time.

## Memory cycle time: -

It is necessary that the information should be written back from where it was read. The duration of read and write operation is called as memory cycle time.

SEMICONDUCTOR RAM: -

Semiconductor memories are available in a wide range of speed and their cycle time ranges from 100 ns to less than 10ns.

**ORAGANISATION OF MEMORY CHIP:** -

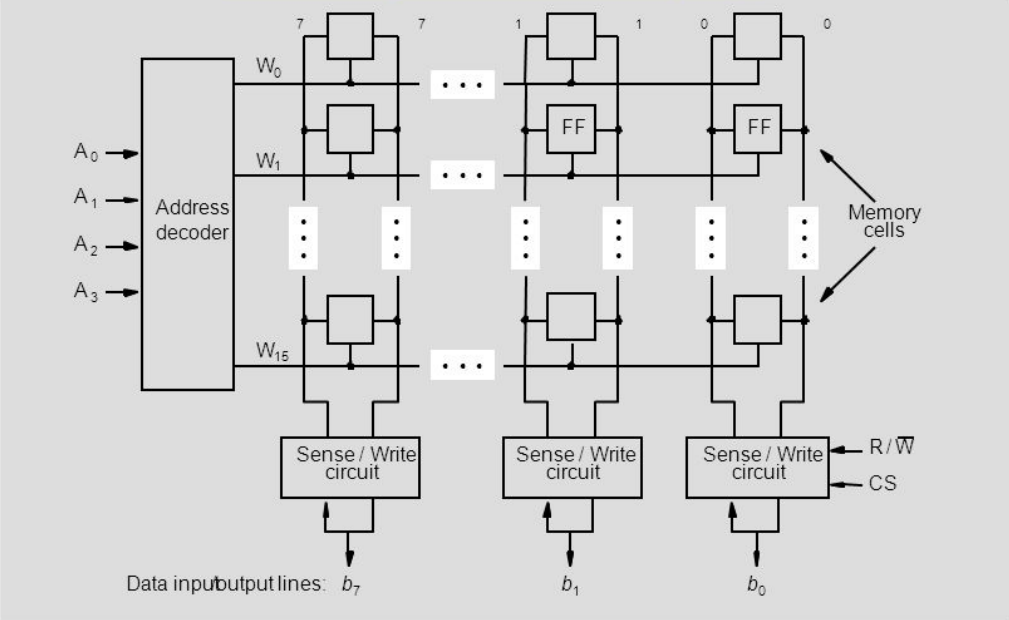
Memory cells are organized in the form of an array of cells, each cell capable of storing one bit of information.

- Each row of cells constitutes a memory word of 8bits b0-b7 and cells of a row are connected to a common signal line called "word line", which is driven by the address decoder.
- Two 'bit lines ' connect the cells in each column to a sense /write circuit.
- The sense / write circuit are connected to the data I/O lines.
- During a 'Read' operation , these circuits read the information stored and transmits this information to the output data line
- During a 'Write' operation , the sense/write circuit receive input information and store it in the selected cell.
- This following organization of a very small memory chip consisting of 8bits of 16 words i.e., 16×8 organization.
- The data I/O of each sense/write circuit are connected to a single bidirectional data line that are connected to the data bus of the computer.

- In addition, there are also two control lines R/W and CS(chip select)
- The R/W signal line specifies ,the required operations and CS selects a given chip in multi - chip memory system.
- This memory circuit stores 128 bit and it requires 14external connection for address data and control line. It also needs two lines for power supply and ground connection.
- For a larger memory circuit let 1k(1024) memory cell can be organized as 128×8 memory requires 19 external connection.

# Semiconductor RAM memories

Internal organization of memory chips




**LATENCY AND BANDWIDTH: -**

- The speed and efficiency of transfer of word or blocks between memory and processor greatly affect the performance of a computer system.
- There are two parameter latency and bandwidth give an indication of performance of computer system.

**LATENCY: -**

- The amount of time it takes to transfer a word of data to or from the memory is referred to a LATENCY of a memory.
- Latency provides a complete indication of memory performance in case of a reading or writing of a single word.

**BANDWIDTH: -**

When transferring a block of data as the block size can be variable, the performance Measures in terms of the number of bits or bytes that can be transferred in one second is known as BANDWIDTH of memory unit.

The bandwidth clearly depends upon the speed of access and no. of bits that can be transferred in parallel ,so the bandwidth is the product of data transferred and the width of data bus.

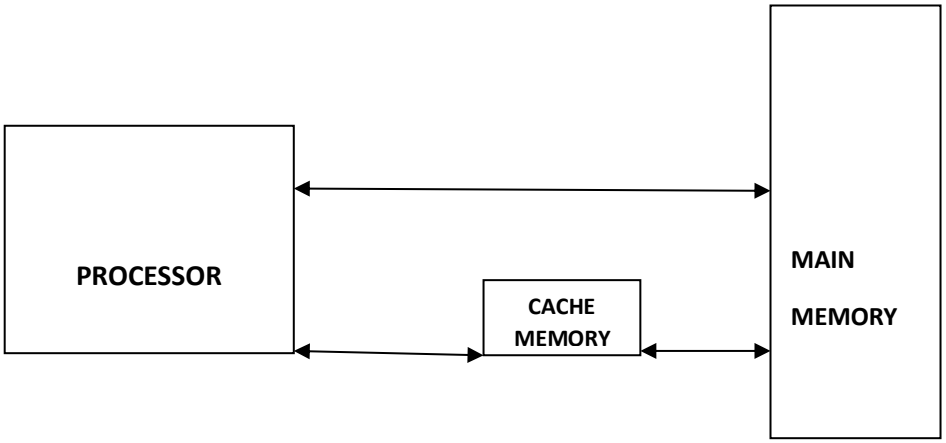
**ROM (READ ONLY MEMORY)**

Both SRAM and DREAM chips are volatile in nature as they lose the stored information if the power is removed.

- Many applications need memory devices to retain the stored information even if power is removed.
- When a computer is turned on the operating system software has to be loaded from the disk into the memory.
- The boot program is quite large and most of it is stored on the disk ,but the processor should execute some instructions that loads the boot program; so, a small amount of non-volatile memory that holds the instruction which helps in loading the boot program from the disk.
- Since, it's normal operation involves only reading of stored data , a memory of this type is called Read only memory.

## Cache memory: -

- The speed of main memory is very slow in compare to the speed of processor. So, for better performance a high-speed memory is used in between main memory and CPU. That is known as cache memory.
- The cache memory comes from the word cache meaning to hide.
- The basic idea behind a cache is simple i.e., the most heavily used memory words are kept in the cache, when the CPU needs a word, it will first look in the cache, only if the word is not there, it goes to main memory.
- Analysis of a program shows the maximum program execution time spent in those portions in which many instructions were executed repeatedly as in loops, hence for the execution of the programs forms a localize area, hence for the execution of the programs forms a localize area, where the programs or instruction executed repeatedly and the remainder of the programs are executed relatively less frequently that is called **locality of reference.**



## Read operation: -

- When the CPU needs to access memory, the cache is first searched, if the word is found, it or read, it is known as **"cache hit"**
- If the word is not found, then the main memory is referred as **"cache miss".**
- When a "cache miss" occurs, it initiates to access main memory to transfer the required byte or word from main memory to cache.
- The performance of the cache memory is known as **hit ratio.**

## Problem: -

Cache Access Time = 100hs

Main memory access time is 1000ns

Hit Ratio=0.9

Find out the average memory access time.

(Ans) Average memory access time = cache access+ main memory time

= h* cache access time + (1-h) *main memory access time

Average Memory Access Time

=0.9*100+ (1-0.9) *100

=90 + 100 =190ns

## Write operation: -

During read operation, when the CPU finds a word in cache memory, then the main memory is not involved in the transfer. But in case of write operation there are two ways of writing.

(1) Write through policy.

(2) Write back policy.

## Write through policy: -

The simplest and most commonly used procedure is to update main memory with every memory write operation with cache memory being update in parallel. This is called as write through policy.

## Advantages: -

This method is the most important characteristics of direct memory access transfer.

### Write back policy

- If the cache follows this policy, then the cache is updated during write operation and the location is marked by a flag.
- When the block of the cache containing the flagged word is required to transfer the main memory at that time it is updated in main memory.

### Advantages

Whenever the word is updated several times, it is better to use write back policy.

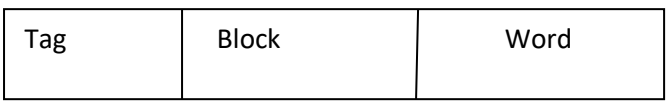
## Mapping

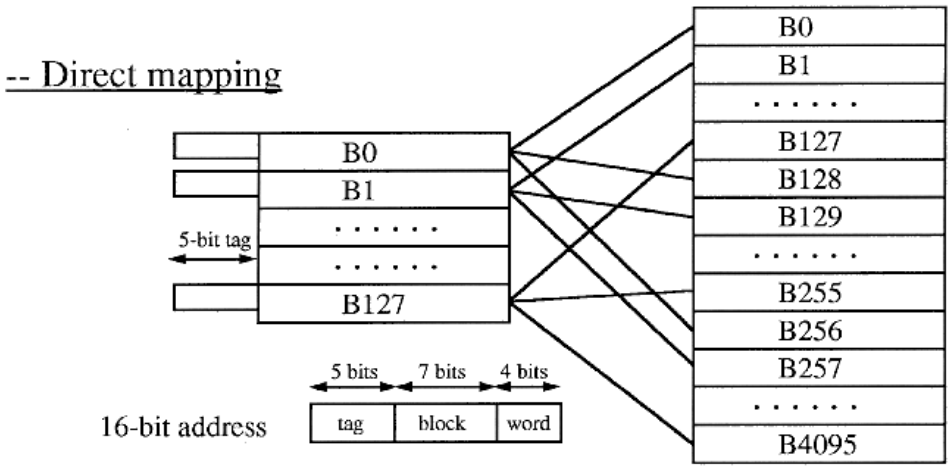
- The transformation of information from main memory to cache memory is known as Mapping.
- There are three types of mapping function
  - ✓ Direct Mapping
  - ✓ Associative Mapping
  - ✓ Set-associative Mapping
- To explain the mapping procedures, we consider 2K cache consisting of 128 blocks of 16 words each, and a 64K main memory addressable by a 16-bit address, 4096 blocks of 16 words each.

### Direct Mapping

- Block m of the main memory maps onto block c of cache memory according to formula c =m mod no of block of cache memory
  c = m mod 128
- By this formula the block o,128,256…. Of main memory will be loaded to 0 block of cache memory. Similarly block 1,129,257 …. of main memory will be loaded to 1 of CM and likewise.
- For this mapping the CPU generates address for cache memory is 16 bits.
- The address is divided into 3 parts.
  - ✓ Word
  - ✓ Block
  - ✓ Tag

The bit pattern will be






- One block contains 16 words , so 4 bits are required to indicate word field, cache contain 128 blocks so 7 bits are required to indicate block field and the rest 16-(7+4)=5 bits are required to indicate tag bit.
- When CPU wants to read or write then the higher order 5 bits of the address are being compared with the tag bit of cache memory,
- If it matches then the desired word is present and a cache hit occurs.
- If not, there will be a cache miss which leads to a write operation.

**Advantage**

Simplest method of implementation.

**Disadvantage**

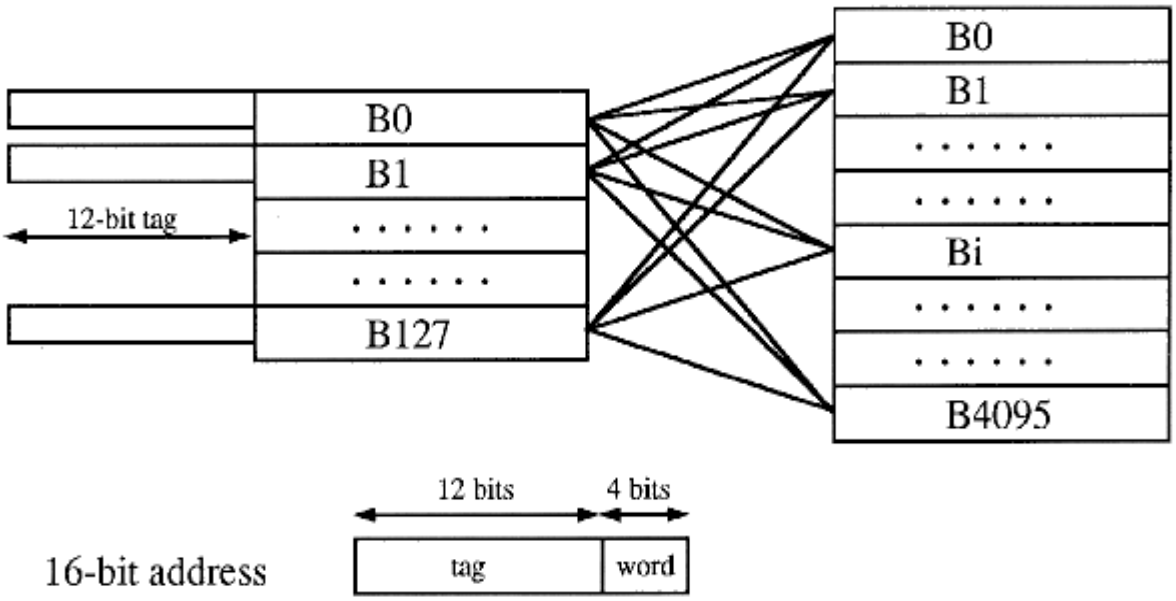
The contention problem may occur even though the cache is not full.

## Associative mapping

- A main memory block can be placed into any cache block position.
- The 12 tag bits identify a memory block residing in the cache.
- The lower-order 4 bits select one of 16 words in a block.
- The cost of an associative cache is relatively high because of the need to search all 128 tags to determine whether a given block is in the cache or not.

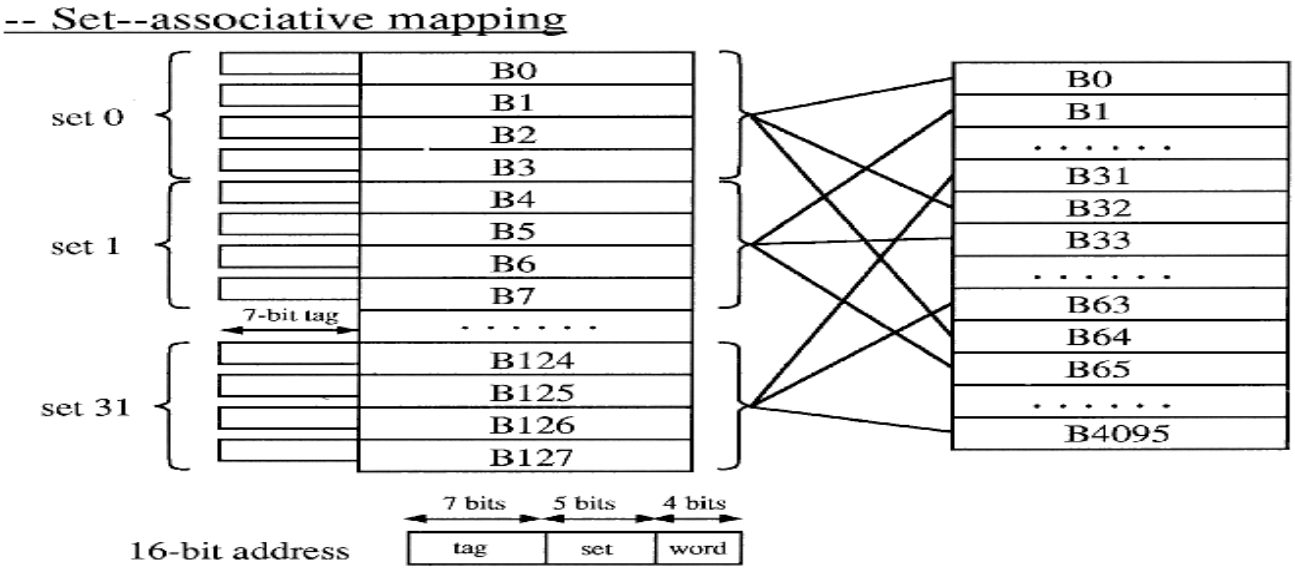
**Advantage:** An MM block can be mapped anywhere in CM.

**Disadvantage:** Slow or expensive. A search through all the 128 CM blocks is needed to check whether the 12 MSBs of the 16-bit address can be matched to any of the tags.




## Set-Associative Mapping:

- Blocks of the cache are grouped into sets, and the mapping allows a block of the main memory to reside in any block of a specific set.
- A cache that has k no of blocks per set is referred to as a k-way set-associative cache.
- The contention problem of the direct method is eased.
- The hardware cost of the associative method is reduced.



## Interleaved Memory

- The two key factors in the success of computer are performance and cost.
- This can be achieved through parallelism.
- In parallel processing or pipeline environment, the main memory is the prime system resources, which is normally shared by all processor or stages of the pipeline.
- In such cases there may be memory interference, which as a result degrades the performance. So, to avoid this problem, a new method is adopted which is known as “Memory interleaving”.
- The Memory interleaving means the main memory of the computer into a no of modules and distributing the address among those modules.
- Each memory module has its own Address Buffer Register (ABR)or Memory Address Register (MAR) and Data Buffer Register (DBR) or Memory Buffer Register (MBR).
- There are two memory address layouts:

    1. High order interleaving (Consecutive words in a module)

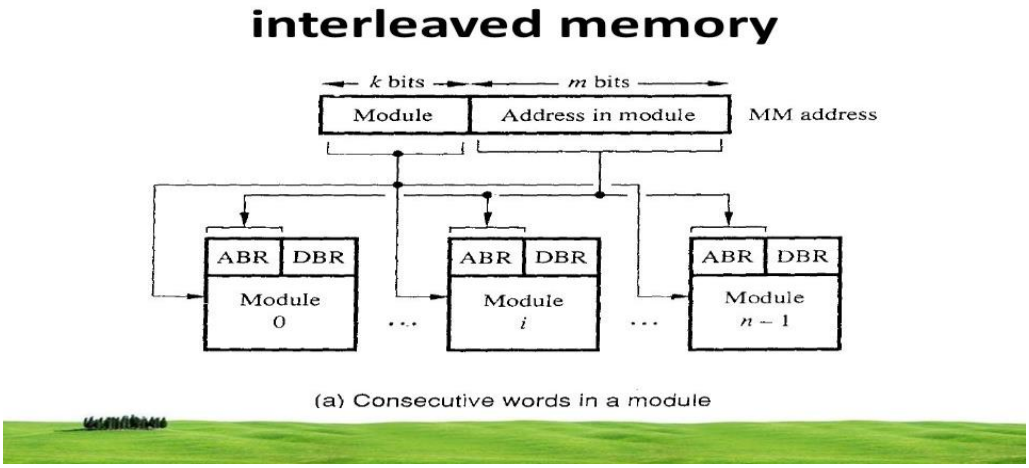
    2. Low order interleaving (consecutive words in consecutive module)

- The address consists of :(1) (2) low-order m-bits point to a particular word in that module

**HIGH ORDER INTERLEAVING: -**

In this type of memory INTERLEAVING the memory is divided into M no. of modules where the consecutive address lies in a single module.

- In this method the higher order bit of the address used for indicating the module no. and the lower order bits are used for indicating the address in module.
- Let for example we have a memory having 16 words .



- In the above case the higher order bits used for indicating the module no. and the lower order bits are used for the words in the module .
- In this case each memory address is of n bit out of which the higher order m bits are used for interleaving and n-m bits are used for the words in particular module.
- The m bits are being decided by the decider which will specify the particular module no. and n-m bit specify the words in the module .
- Every memory module has its own MAR and MBR.

Advantage: -

- It permits easy expansion of memory by addition of one or more memory module as needed to a maximum of m-1.
- Better system reliability in case of a failed module as it affects only a localized area f address space.

Disadvantage: -

- When consecutive location is to be accessed then only one module is involved.

**LOW ORDER INTERLEAVING: -**

In this memory interleaving the consecutive words are distributed in consecutive modules.

- Here the higher order n-m bits are used for address of words in a module while m lower bits are used for module no. .
- This method is efficient way to address the module.
- Here any request for accessing consecutive words can keep several modules busy at the same time ;this is faster than the previous one and so used frequently.
- Ex:-all have 16 words.

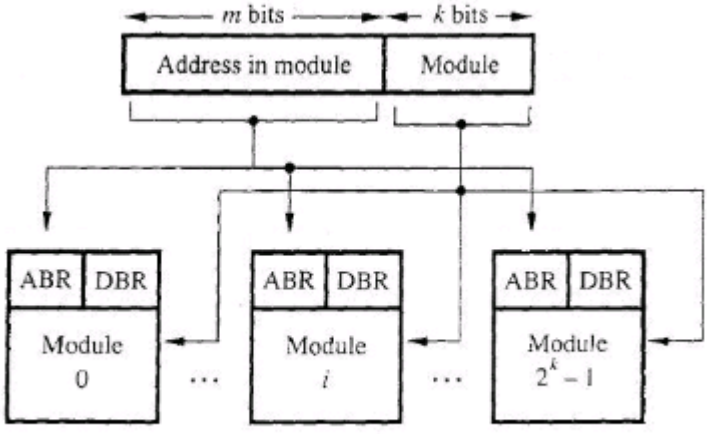



**Figure 9**

High-order arrangement

| | M0 | |
|---|---|---|
| 0 | 00 | 00 |
| 1 | 00 | 01 |
| 2 | 00 | 10 |
| 3 | 00 | 11 |

| | M1 | |
|---|---|---|
| 4 | 01 | 00 |
| 5 | 01 | 01 |
| 6 | 01 | 10 |
| 7 | 01 | 11 |

| | M2 | |
|---|---|---|
| 8 | 10 | 00 |
| 9 | 10 | 01 |
| 10 | 10 | 10 |
| 11 | 10 | 11 |

| | M3 | |
|---|---|---|
| 12 | 11 | 00 |
| 13 | 11 | 01 |
| 14 | 11 | 10 |
| 15 | 11 | 11 |

Low-order arrangement (interleaving)

| | M0 | |
|---|---|---|
| 0 | 00 | 00 |
| 4 | 01 | 00 |
| 8 | 10 | 00 |
| 12 | 11 | 00 |

| | M1 | |
|---|---|---|
| 1 | 00 | 01 |
| 5 | 01 | 01 |
| 9 | 10 | 01 |
| 13 | 11 | 01 |

| | M2 | |
|---|---|---|
| 2 | 00 | 10 |
| 6 | 01 | 10 |
| 10 | 10 | 10 |
| 14 | 11 | 10 |

| | M3 | |
|---|---|---|
| 3 | 00 | 11 |
| 7 | 01 | 11 |
| 11 | 10 | 11 |
| 15 | 11 | 11 |

# CHAPTER- 5

## Input – Output System

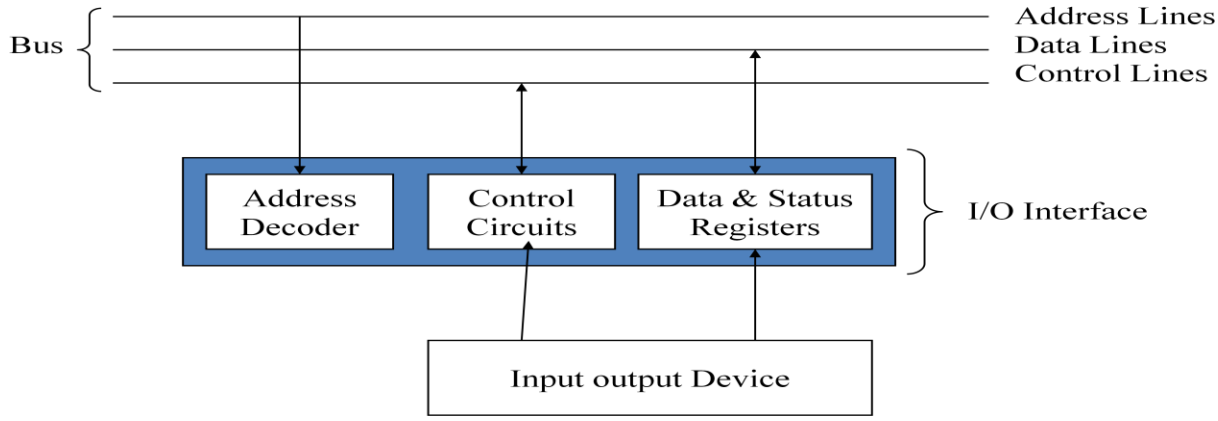
### Introduction

- I/O is an essential & integral part of computer system. Variety of I/O devices on a computer system - keyboard, mouse, display, magnetic disk, network adaptor, etc.
- I/O devices differ widely in terms of function, size, mode ofoperation, transfer speed, power consumption, etc.
- All devices must be connected to the processor and memory using the same basic architecture.
- Over the years, different mechanisms have been developed to connect I/O devices to systems, and to program I/O data transfers over the resulting connections.

### I/O controllers

- An I/O device is connected to the computer system by using a *device controller.*
- I/O devices vary in of some or all of the following characteristics:
    - Representation of data: voltage, current, magnetic field,etc.
    - Speed of operation and data transfer
    - Timing and control requirements
    - Need to detect physical events – e.g., mouse clicks or keypresses
    - Need for error detection and correction.
- To resolve these problems the computer system, include special hardware circuits between CPU & peripherals. These are calledas interface unit. Each device has its own interfacing unit.
- The purpose of communicational link is to solve the following problems:
    - Peripherals are electromechanical devices but CPU & memory are pure electronic devices.
    - Data transfer rate is slower than that of CPU, so synchronization mechanism is required.
    - Data codes and formats of peripheral devices different from the word format of CPU and memory.
    - Operating mode of peripheral are different from each other.Each other must be controlled, so that it will not disturb the operation of others.

**Interfacing between peripherals and processor**



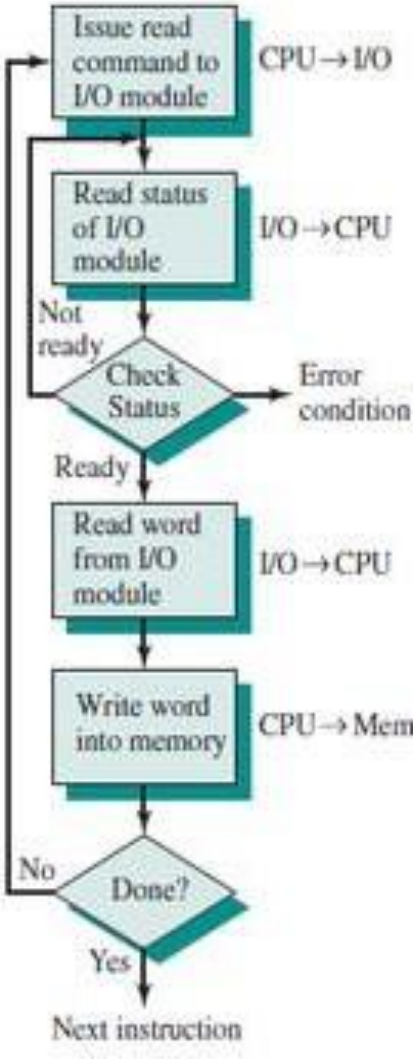

- The above diagram shows the communication link between the processor and peripherals devices.
- They are connected via the I/O bus. i/o bus consist of address bus, control bus, and data bus.
- Each peripheral consists of its own interface. each interface attached to the input output bus which contains an addressdecoder, that monitors the address line.
- When the interface detects its own address then it activates the path between the bus line and device. Rest lines are disabled.
- This is also called as input output command. The following command can be received.
  - ✓ **Control command**-This is issued to activate the peripheraland to inform what to do.
  - ✓ **Test command**-this is used to test various status conditionin the interface and the peripherals.
  - ✓ **Read command**-it causes the interface to respond bytransferring the data from the bus into one of the registers.
  - ✓ **Write command**-this is used to receive an item of datafrom peripheral device and place it in a buffer register.
- All the above process being accompanied by the control circuit of i/o interface.
- After giving the control command in the control line the CPUactivate the data bus. Then the data transfer takes place.

  I\o operation can be done using three techniques.

  1. Programmed I/O
  2. Interrupt driven I/O
  3. Direct memory access

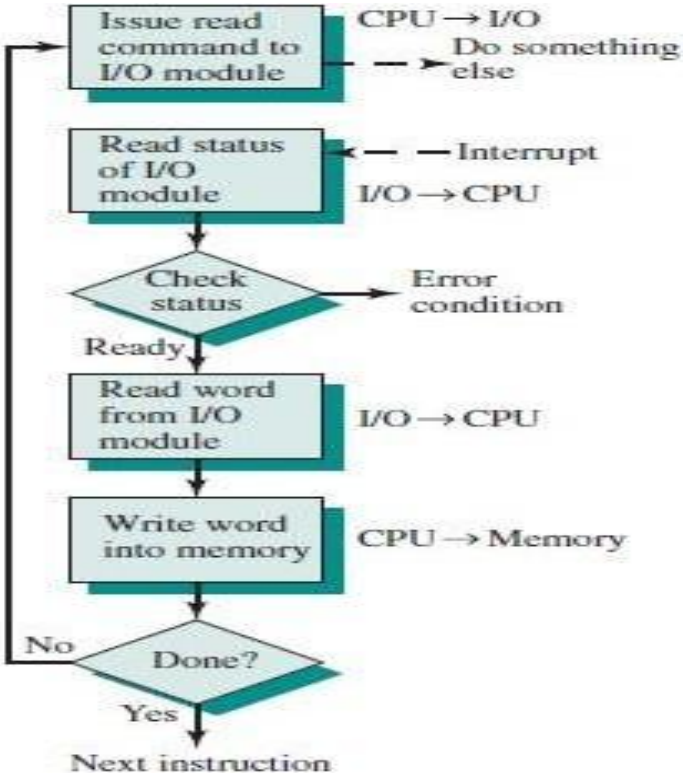
## 1. Programmed I/O

- In programmed I/O, data are exchanged between the processor and the I/O module.
- The processor directly controls of the I/O operation, including sensing device status, sending a read or write command, and transferring the data.
- When the processor issues a command to the I/O module, it must wait until the I/O operation is completed .
- As the processor is faster than the I/O module, a lot of processor time is wasted.




### Interrupt driven I/O

- The problem with programmed I/O is that the processor has to wait a long time for the I/O module of concern to be ready for either reception or transmission of data.
- The solution to this problem is to provide an interruptmechanism. In this approach the processor issues an I/O command to a module and then go on to do some other useful work.
- The I/O module then interrupt the processor to request service when it is ready to exchange data with the processor. Theprocessor then executes the data transfer. Once the data transfer is over, the processor then resumes its former processing.




## Processing of interrupt

**Interrupt**

- It is defined as an event external to the currently executing process that causes a change in the normal flow of instruction execution; usually generated by hardware devices external to the CPU.
- During a processing of interrupt a no of events in both processor hardware and software are triggered by the interrupt.
- It is being categorized into as below.

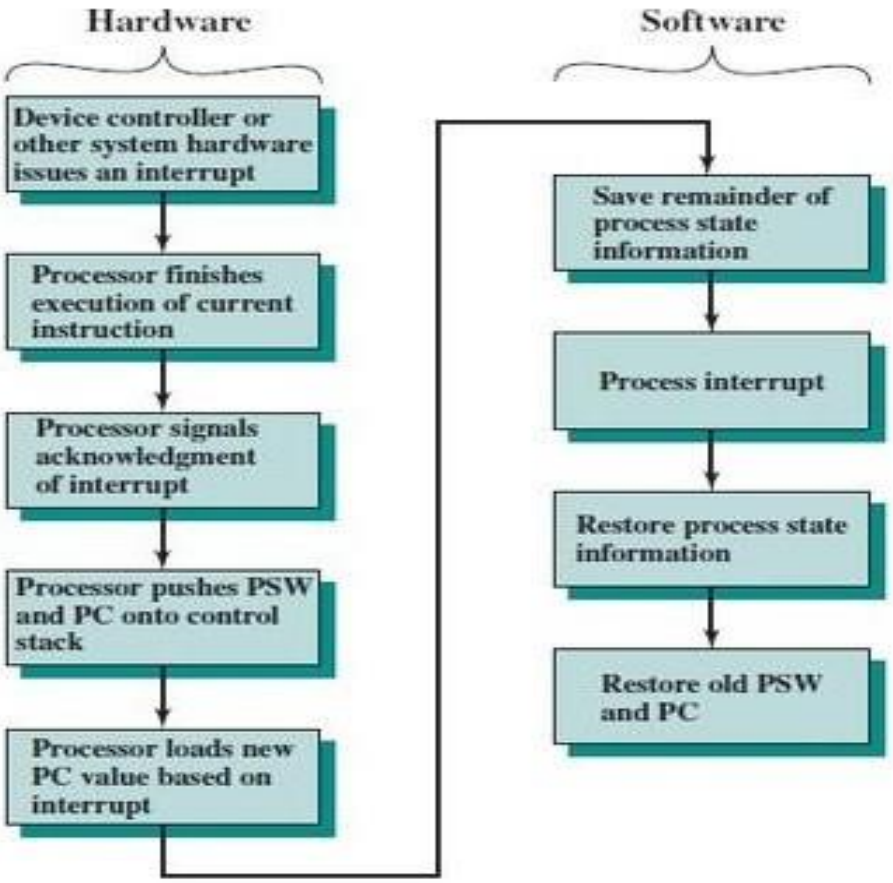
### Hardware processing

The sequence of hardware events that occur when an i\o device completes an operation are:

- ✓ An interrupt signal is issued by the i\o device controller to the processor.
- ✓ The processor responds to the interrupt after finishingexecution of current instruction.
- ✓ The processor sends an acknowledgement signal to the device that issued the interrupt.
- ✓ Before processing the interrupt , processor saves the status in PSW and saves the location of the next instruction to be executed in the program counter.
- ✓ The processor now loads the PC with the address corresponding to the interrupt.

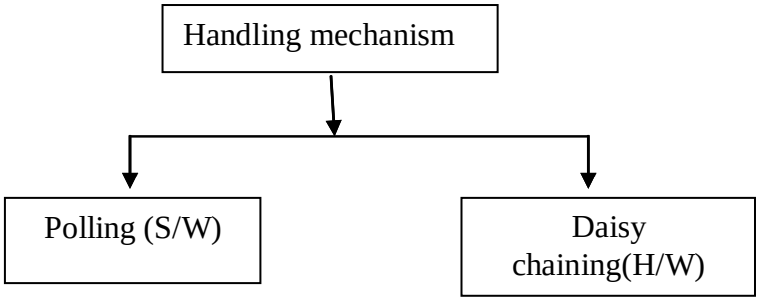
## Software processing

- ✓ The stack pointer is updated to point to the new TOS andthe PC to beginning of interrupt service routine.
- ✓ The interrupt handler will next process the interrupt.
- ✓ When the interrupt processing is completed the savedregister values are restored .
- ✓ The PSW and PC values are finally restored from thestack.
- ✓ All the state of the program information about the ISR hasto be saved for future reference.




## Methods for handling interrupt

- Every interrupt has its service routine to handle it.
- When the CPU gets any interrupt signal, it responds to the interrupt by storing the return address to a stack and the program branches to a service routine that process that interrupt.
- After solving the interrupt, the processor back to the original program.
- As there are a number of i/o devices attached to the computer, at a time more than one interrupt request can arise.In this case the system must also decide which device to serve first.
- This problem can be solved by giving priority & the interrupt ofhigh priority device will be solved first.
- The priority can be established by hardware & softwaremethod.




Polling

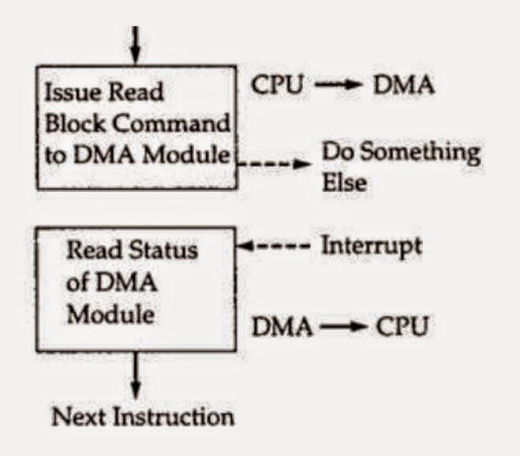
- Polling is a software method which identifies the highestpriority source.
- The highest priority source is tested first & if its interrupt signal is on then controls branches to a service routine for theservice. Otherwise, the next lower priority is tested and so on.
- But this is slower.

Daisy chaining

- It is a hardware method.
- Daisy chaining consists of a several connections of all devicesthat request an interrupt.
- The device with highest priority is placed in the first positionfollowed by lower priority which is placed the last of the chain

## Direct memory access

- The data transfer between the processor and I/O devices namely programmed I/O and Interrupt-driven I/O, both the methods require the active intervention of the processor to transfer data between memory and the I/O module, and any datatransfer must traverse a path through the processor.
- To transfer large block of data at high speed, a special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor. This approach is called ***direct memory access*** **or DMA**.
- DMA module is capable of taking over control of the system fromthe processor to transfer data between I/O and memory over the system bus , this is done by allocating the bus when the processor is not using it.



## Operation of DMA module

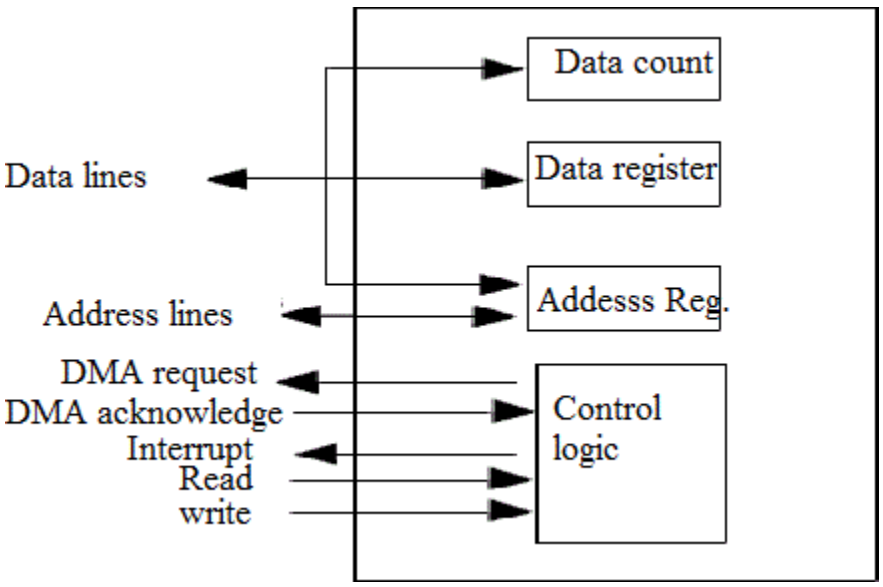
- DMA transfers are performed by a control circuit associated with the I/O device and this circuit is referred as DMA controller.
- With the help of Read or Write control line ,a specific operation isrequested.
- I/o device which wants to send data ,DMA sends a HOLD signal to CPU on receiving HOLD signal it sends a HOLD ACK to the i/odevice indicating that it has received it.
- The address of the I/O device and the starting location in memory need to be Read/Write is communicated on the address line and stored by the DMA module in the addressregister.
- The number of words to be transferred is communicated on the data lines and stored in the data counter register.
- On the completion of the transfer of data DMA module sends an interrupt signal to the processor.
- DMA data transfer scheme are of following two type-
  - ✓ Burst Mode
  - ✓ Cycle stealing Mode.

### Burst mode

This is the most common type of DMA used. In this mode I/o device withdraws the DMA request only after all data bytes aretransferred.

### Cycle stealing

In this mode a block of data is transferred by a sequence of DMA cycle.Here the I/O device withdraws DMA request after transferring one or several bytes.



# CHAPTER- 6

## I/O Interface & Bus architecture

### BUS INTERCONNECTION

- A bus is a communication pathway connecting two or more devices.
- A key characteristic of a bus is that it is a shared transmission medium.
- Multiple devices connect to the bus, and a signal transmitted by any one device isavailable for reception by all other devices attached to the bus (broadcast).
- Typically, a bus consists of multiple communication pathways, or lines. Each line iscapable of transmitting signals representing binary 1 and binary 0.
- Taken together, several lines of a bus can be used to transmit binary digits simultaneously (in parallel). For example, an 8-bil unit of data can be transmitted overeight bus lines.
- Computer systems contain a number of different buses that provide pathways between components at various levels of the computer system hierarchy.
- A bus that connects major computer components (processor, memory, I/O) is called a **system bus**. The most common computer interconnection structures are based onthe use of one or more system buses.

### BUS STRUCTURE

A system bus consists, typically, of from about 50 to hundreds of separate lines. Each line is assigned a particular meaning or function. Although there are many different bus designs, on any bus the lines can be classified into three functional groups data, address, and control lines. In addition, there may he power distribution lines that supply power to the attached modules.

#### Data Bus

- Provide a path for moving, data between system modules. These lines, collectively,are called the data bus.
- The width of the data bus: The data bus may consist of from 32 to hundreds of separate lines, the number of lines being referred to as the width of the data bus. Because each line can carry only 1 bit at a time, the number of lines determines howmany bits can be transferred at a lime. The width of the data bus is a key factor in determining overall system performance. For example, if the data bus is 8 bits wide and each instruction is 16 bits long, then the processor must access the memory module twice during each instruction cycle.
- As data bus carry information from and to the modules, so it is bidirectional in nature.

#### Address Bus

- Address lines are used to designate the source or destination of the data on the databus. For example, if the processor wishes to read a word (8, 16. or 32 bits) of data from memory, it puts the address of the desired word on the address lines.

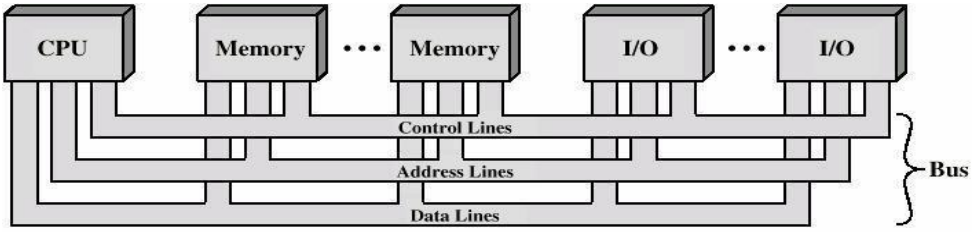
- The width of the address bus: determines the maximum possible memory capacity ofthe system. Furthermore, the address lines are generally also used to address I/O ports.

#### Control Bus

- Control bus are used to control the access to and the use of the data and address lines. Because the data and address lines are shared by all components, there mustbe a means of controlling their use.
- Control signals transmit both command and timing information between system modules.

Typical control lines include the following:

- Memory write: Causes data on the bus to be written into the addressed location.
- Memory read: Causes data from the addressed location to be placed on the bus.
- I/O write: Causes data on the bus to be output to the addressed I/O port.
- I/O read: Causes data from the addressed I/O port to be placed on the bus.
- Transfer ACK: Indicates that data have been accepted from or placed on the bus.
- Bus request: Indicates that a module needs to gain control of the bus.
- Bus grant: Indicates that a requesting module has been granted control of the bus.
- Interrupt request: Indicates that an interrupt is pending.
- Interrupt ACK: Acknowledges that the pending interrupt has been recognized.
- Clock: Used to synchronize operations.
- Reset: Initializes all modules.




## MULTIPLE-BUS ARCHITECTURE

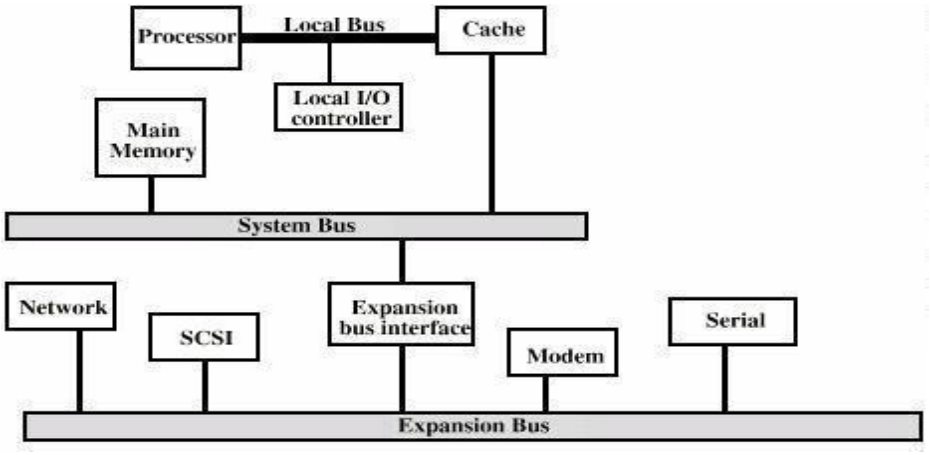
If a great number of devices are connected to the bus, performance will suffer. There are two main causes:

1. Propagation delay - the time it takes for devices to coordinate the use of the bus
2. The bus may become a bottleneck as the aggregate data transfer demandapproaches the capacity of the bus (in available transfer cycles/second).
    - Accordingly, most computer systems use multiple buses, generally laid out in a hierarchy.

There are mainly two typical architecture: 1) Traditional Bus architecture, 2)High performance Bus architecture

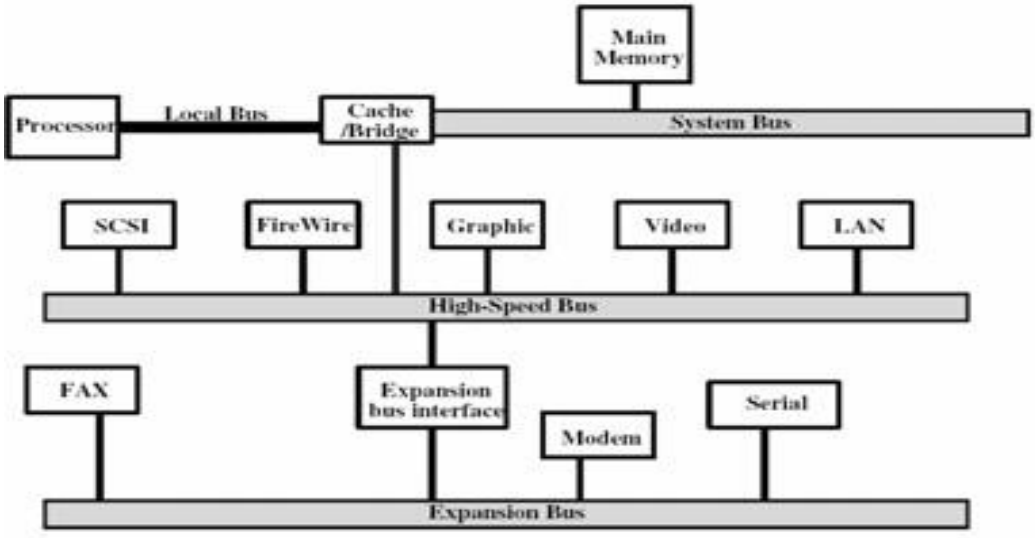
# Traditional Bus architecture

- This consists of three buses, like local bus, system bus and expansion bus
- There is a local bus that connects the processor to a cache memory and that maysupport one or more local devices.
- The cache memory controller connects the cache not only to this local bus, but to asystem bus to which are attached all of the main memory modules.
- It is possible to connected I/O controllers directly onto the system bus. A more efficient solution is to make use of one or more expansion buses for this purpose.
- An expansion bus interface buffers data transfers between the system bus and theI/O controllers on the expansion bus.
- This arrangement allows the system to support a wide variety of I/O devices and atthe same time insulate memory-to-processor traffic from I/O traffic.




## High performance Bus architecture

- Due to the increasing need for adding more i/o devices the traditional architecturecannot support so, it is necessary to build a high-performance bus.
- Similar to the traditional bus architecture, this too contains a local bus that connectsthe processor to a cache controller, which in turn is connected to the main memory through the system bus.
- The cache controller is integrated into a bridge or buffering device that connects tothe high-speed bus, which sometimes referred to as Mezzanine Architecture.
- This bus supports to high-speed LANs, video and graphics work station controllersetc.
- The lower speed devices are still supported by the expansion bus with an interfacebuffering traffic between the expansion bus and high-speed bus.
- This way the high-speed devices are more closely integrated with the processor through the high-speed bus and at the same time leaving processor independent.



## BASIC PARAMETERS OF BUS DESIGN

Before designing a bus some of the following basic parameters are to be considered

1. Bus type
2. Width of the bus
3. Method of arbitration
4. Timing
5. Data transfer

### Bus type

- Bus lines can be separated into two generic types: dedicated and multiplexed.
- A **dedicated bus** line is permanently assigned either to one function or to physicalsubset of computer components.
- Separate data & address lines are used.
- The use of the same lines for multiple purposes is known as **Multiplexing.** Shared lines
- Address valid or data valid control lines are used.

### Width of the bus

- The width of the data bus has an impact on the system performance
- The wider the data bus, the greater the number of bits can be transferred at one time.
- The width of the address bus has an impact on the system capacity.
- The wider the address bus, the greater the range of locations that can be referenced.

### Method of arbitration

It determining who can use the bus at a particular time

- ***Centralized*** - a single hardware device called the bus controller or arbiter allocatestime on the bus
- ***Distributed*** - each module contains access control logic and the modules acttogether to share the bus
- Both methods designate one device (either CPU or an I/O module) as master, whichmay initiate a data transfer with some other device, which acts as a slave.

### Timing

#### *Synchronous Timing*

- Bus includes a clock line upon which a clock transmits a regular sequence ofalternating 1's and 0's of equal duration
- A single 1-0 transmission is referred to as a clock cycle or bus cycle
- All other devices on the bus can read the clock line, and all events start at thebeginning of a clock cycle

#### *Asynchronous Timing*

- The occurrence of one event on a bus follows and depends on the occurrence of aprevious event

### Data transfer

A bus can support various type of data transfer such as-

- Read, Write, Read-modify-write, Read-after-write, Block
- All buses must support write (master to slave) and read (slave to master)transfers.
- ***Read-modify-write***: A read followed immediately by a write to the sameaddress.

- Address is only broadcast once, at the beginning of the operation
- ***Read-after-write***: Indivisible operation consisting of a write followedimmediately by a read from the same address (for error checking purposes) .
- ***Block:*** one address cycle followed by n data cycles
- first data item to or from specified address
- Remaining data items to or from subsequent addresses.

## SCSI

The Small Computer System Interface (SCSI) is a set of parallel interface standards developed by the American National Standards Institute (ANSI) for attaching printers, disk drives, scanners and other peripherals to computers. SCSI (pronounced "skuzzy")is supported by all major operating systems.
It has some versions developed –

**SCSI-1** is the original SCSI standard developed back in 1986 as ANSI X3.131-1986.

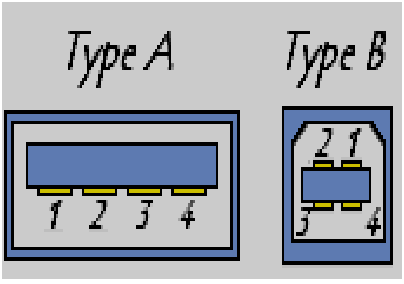
SCSI-1 is capable of transferring up to eight bits a second.

**SCSI-**2 was approved in 1990, added new features such as Fast and Wide SCSI, and support for additional devices.

**SCSI-3** was approved in 1996 as ANSI X3.270-1996.

**USB** (Universal Serial Bus):

- Universal Serial Bus is a new connector that is introduced 1995 to replace Serial and Parallel ports.
- It is based on serial type architecture. However, it is much quicker than standard serial ports because Serial architecture gives the interface a much higher clock rate than a parallel interface and serial cables are much cheaperthan parallel cables.
- So, from 1995, the USB standard has been developed for connecting a wide range ofdevices like scanners, keyboards, mice, joysticks, printers, modems and some CD- ROMs.
- USB is completely hot-swappable that means we can connect or disconnect anydevice when the computer is running.
- Computer can recognize the device as soon as it plugged in, and the user can use ofthe device immediately.




- There are two types of USB connectors:
  - Type A: This type of connectors is generally used for less bandwidth intensive devices like keyboard, mouse, webcam, etc. and shape is rectangular.
  - Type B: This type of connectors is generally used for high-speed devices like externalhard disks, etc. and shape is square.

**PCI:**

- Stands for "Peripheral Component Interconnect." It is a hardware bus designed by Intel around 1992 and is used in both PCs and Macs.
- It is an intermediate bus located between the processor bus (Northbridge) and the I/O bus (Southbridge).
- Most add-on cards such as SCSI, Firewire, and USB controllers use a PCI connection. Some graphics cards use PCI, but most new graphics cards connect to the AGP slot.
- PCI slots are found in the back of the computer. The PCI interface exists in 32 bits with a 124-pin connector or in 64 bits with a 188-pin connector.
- There are also two signaling voltage levels i.e., 3.3V for laptop computers and 5V for desktop computers. The 64-bit PCI connectors offer additional pins and can accommodate 32-bit PCI cards.
- There are 2 types of 64-bit connectors. They are 64-bit PCI connector, 5V and 64-bitPCI connector, 3.3V.

32-bit PCI connector 3.3V

32-bit PCI connector 5V

64-bit PCI connector, 3.3V

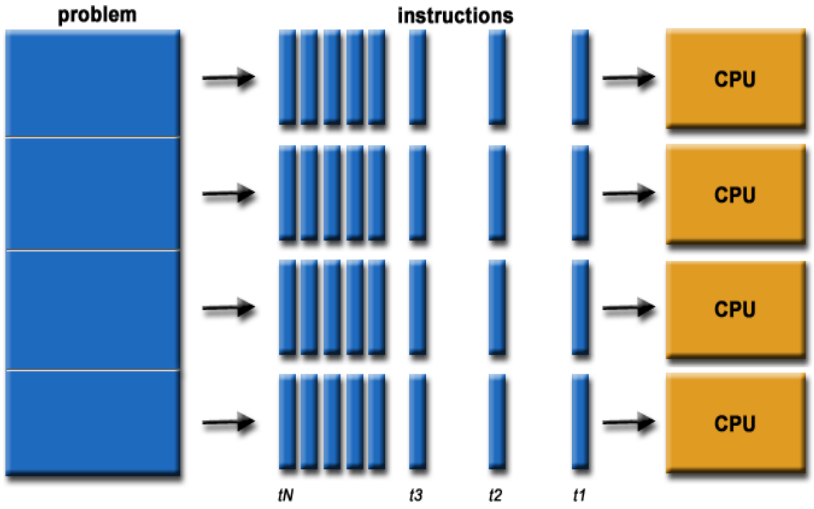
# CHAPTER- 7

## Parallel Processing

With the increased use of computers in every sphere of human activity, computerscientists are faced with two crucial issues today.

1. Processing has to be done faster like never before
2. Larger or complex computation problems need to be solved

- Parallel processing or computing is a form of computation in which many instructions are carried out simultaneously operating on the principle that large problems can often be divided into smaller ones, which are then solvedconcurrently (in parallel).
- Instead of processing each instruction sequentially a parallel processingsystem is able to perform concurrent data processing to achieve faster execution time




- Parallel Computing is used for the following reason
  - Saves time
  - Solve larger problems
  - Cost savings
  - Provide concurrencyIt is broadly categorized into Four types

### Bit level parallelism

When an 8-bit processor needs to add two 16-bit integers, it's to be done in two steps.

- The processor must first add the 8 lower-order bits from each integer using thestandard addition instruction,
- Then add the 8 higher-order bits using an add-with-carry instruction and the carry bitfrom the lower order addition

## Instruction Level Parallelism

The instructions given to a computer for processing can be divided into groups, or re-ordered and then processed without changing the final result. This is known as instruction-level parallelism i.e., ILP.

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

Here, instruction 3 is dependent on instruction 1 and 2 . However, instruction 1 and 2 canbe independently processed.

## Task parallelism

Task Parallelism focuses on distribution of tasks across different processors. It is alsoknown as functional parallelism or control parallelism

## Data Parallelism

Data parallelism focuses on distributing the data across different parallel computing nodes. Itis also called as loop-level parallelism.

Parallel processing can be achieved by using 3 technologies.

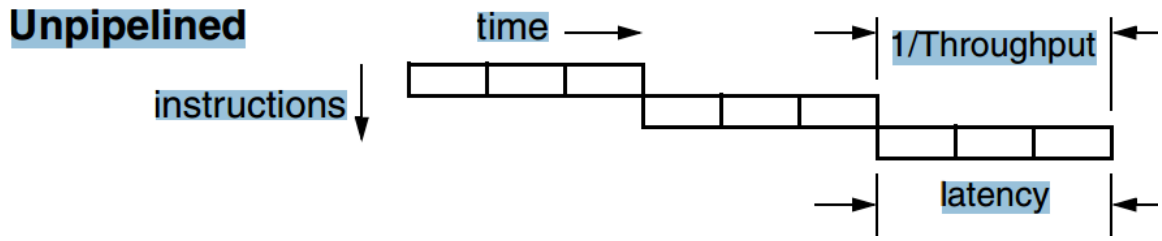
1.pipelining 2.Vector processing3.Array processing

## Linear pipeline

The process of execution of instruction can be divided into 4 major steps.

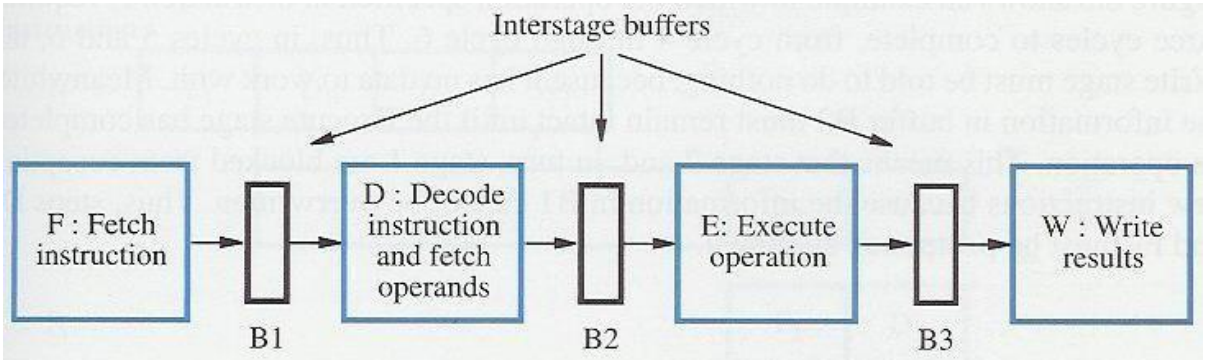
1.Instruction Fetch(IF) 2.Instruction Decode(ID)3.Operand fetch(OF) 4.Execute(EX)

- During IF the instruction is fetched from main memory.
- During ID the operation is identified that is to be performed.
- During OF the operand is fetched from memory (if required).
- During EX the instruction is executed by ALU

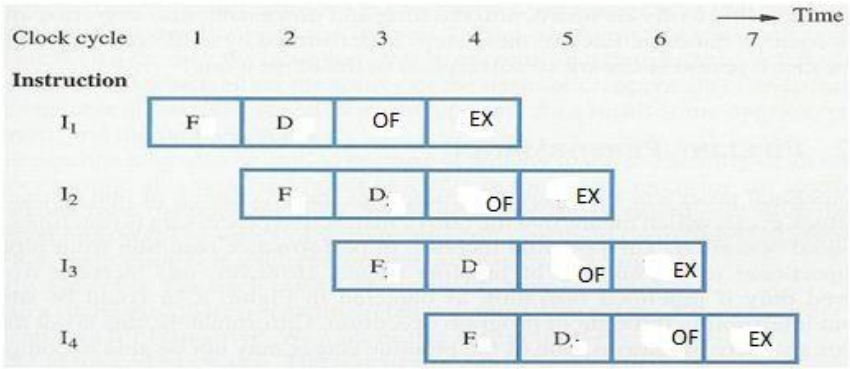



In unpipelined computer all these four steps must be completed before starting ofthe next instruction.
But in case of a pipelined computer successive instruction are executed in anoverlapped fashion.

- It decomposes a sequential process into segments.
- It divides the processor into segment processors each one is dedicated to aparticular segment.
- Each segment is executed in a dedicated segment-processor operatesconcurrently with all other segments.
- Information flows through these multiple hardware segments.




- Instruction execution is divided into k segments or stages
- Instruction exits pipe stage k-1 and proceeds into pipe stage k
- All pipe stages take the same amount of time; called one processor cycle
- Length of the processor cycle is determined by the slowest pipe stage
- There is a inter stage buffer between two successive stages.




## FLYNN'S TAXONOMY

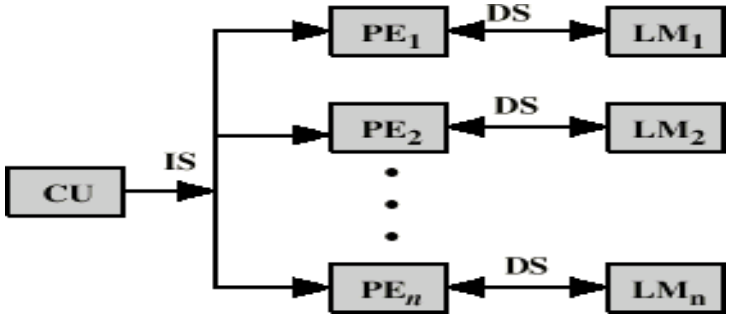
- In general, digital computers may be classified into four categories, according to themultiplicity of instruction and data streams.
- This scheme for classifying computer organizations was introduced by Michael J.Flynn. The essential computing process is the execution of a sequence of instructions on a set of data.
- The term stream is used here to denote a sequence of items (instructions or data) asexecuted or operated upon by a single processor.
- An instruction stream is a sequence of instructions as executed by the machine
- A data stream is a sequence of data including input, partial, or temporary results, called for the instruction stream. Listed below are Flynn's four machine organizations:

## Single instruction streams single data stream (SISD)

SISD computer organization This organization represents most serial computers available today. Instructions are executed sequentially but may be overlapped in their execution stages.

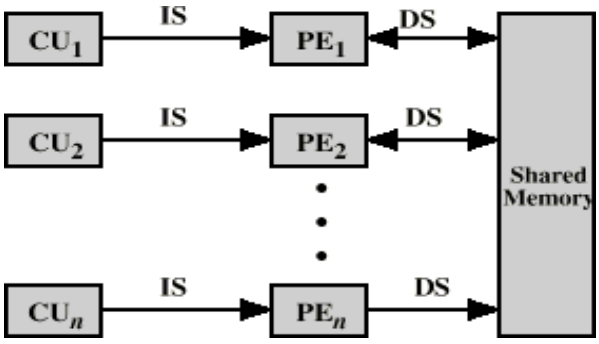

## Single instruction streams multiple data stream (SIMD)

SIMD computer organization: In this organization, there are multiple processing elements supervised by the same control unit. All PE'(processing element) receive the same instruction broadcast from the control unit but operate on different data sets from distinct data streams.




## Multiple instruction streams single data stream (MISD)

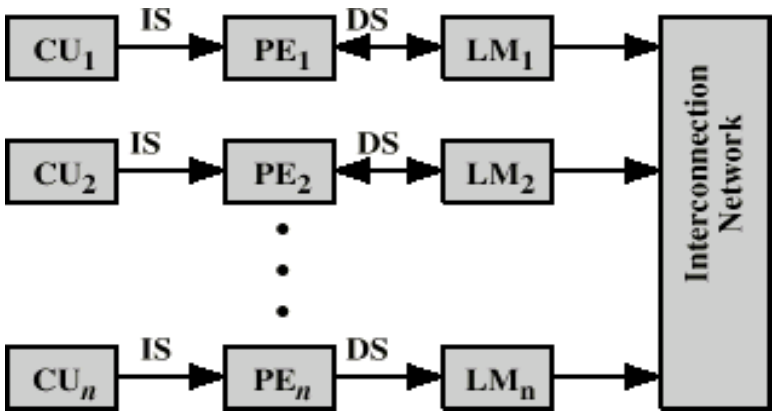
MISD computer organization There are n processor units, each receiving distinct instructions operating over the same data stream. The results (output) of one processor become the input (operands) of the next processor .This approach has no practical implementation.




## Multiple instruction streams multiple data stream (MIMD)

MIMD computer organization Most multiprocessor systems and multiple computer systems

can be classified in this category. MIMD computer implies interactions among the n processors because all memory streams are derived from the same data space shared by allprocessors.




## MULTIPROCESSOR

- A multiprocessor system is an interconnection of two or more CPUs with memory & i\o equipment. The processor in multiprocessor can have either CPU or i\o processor.
- Computers are interconnected with each other by means of communication lines to form a computer network. The network consists of several autonomous computer that may or may not communicate with each other.
- A multiprocessor system is controlled by one OS that provides interaction between processor & all the components of the system cooperate in the solution of a problem.
- Multiprocessor are classified by the way their memory is organized.
    - ✓ Tightly coupled
    - ✓ Loosely coupled.

### Tightly coupled

These systems contain multiple CPUs that are connected at the bus level. These CPUs may have access to a central shared memory (SMP or UMA), or may participate in a memory hierarchy with both local and shared memory (NUMA).

### Loosely coupled

These systems are based on multiple standalone single or dual processor or commodity computers interconnected via a high-speed communication system .

Each PE has its own private local memory. The processor is tied together by a switching scheme designed to route information from one processor to another through a message passing system.

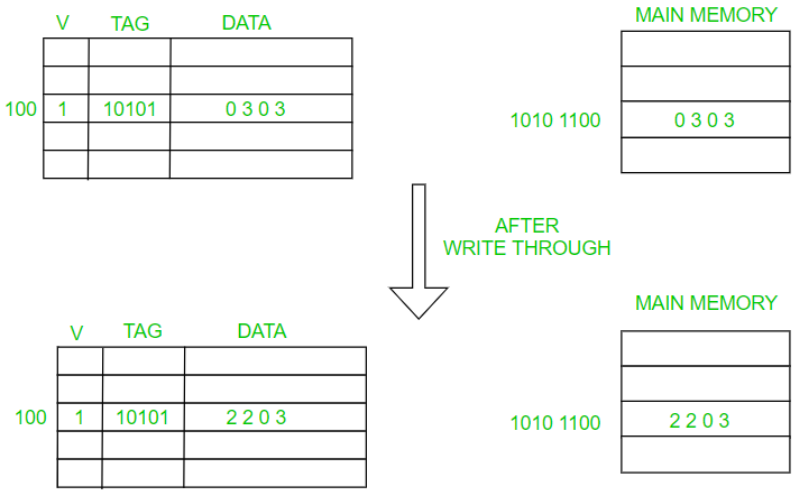
# Advanced Concepts in Cache Memory (Content Beyond Syllabus)

## 1.Write-Through and Write-Back in Cache

**Cache Overview:**
Cache is a temporary storage that holds frequently accessed data to speed up processing. It acts as a buffer between RAM and CPU, enhancing data retrieval speed. When the CPU writes data, it first checks if the corresponding memory address is in the cache (Write Hit). If the data is only updated in the cache, inconsistencies may arise, especially in multiprocessor systems. To handle this, two strategies are used: **Write-Through** and **Write-Back**.

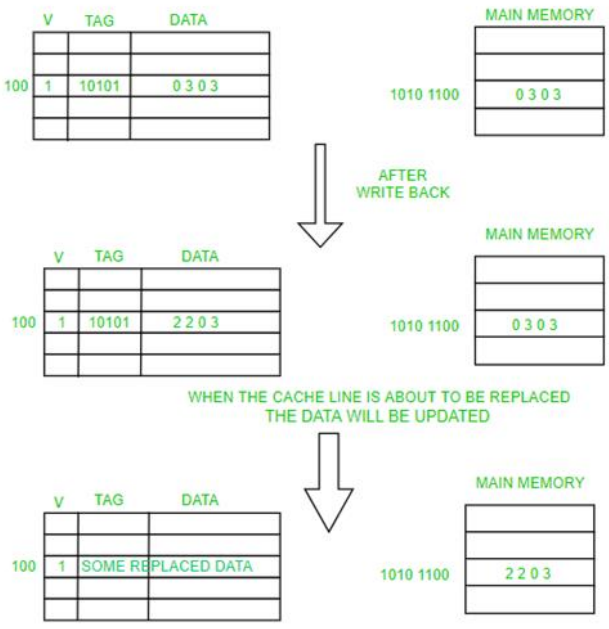
## Write-Through:

- Data is written to both cache and main memory simultaneously.
- Ensures data consistency but adds latency since both cache and memory must be updated.
- Suitable for systems with fewer write operations.
- Helps in data recovery during power failures but reduces cache efficiency during writes.



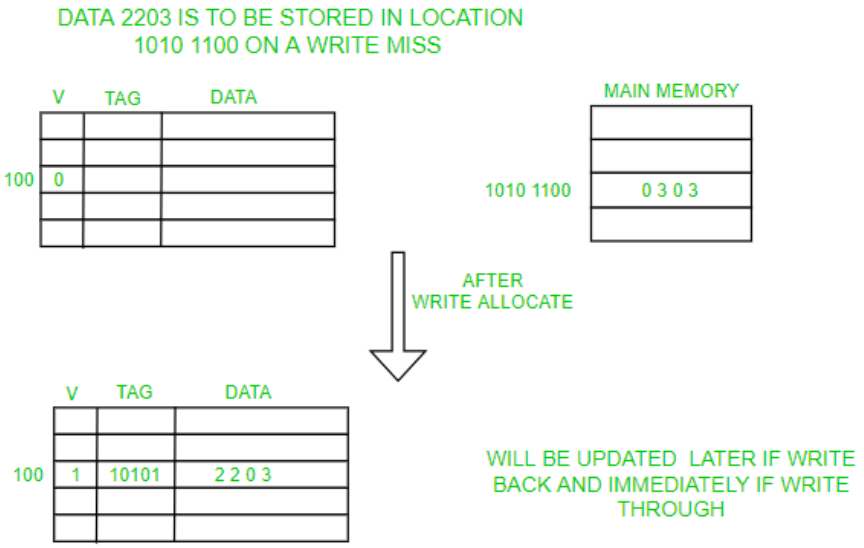

## Write-Back:

- Data is written only to the cache initially and updated in memory later, during cache line replacement.
- Reduces write operations to main memory, improving performance.
- A **Dirty Bit** is used to track modified cache data. Memory is updated only if the dirty bit is set during cache replacement.
- Faster but riskier, as data loss can occur during system failures.

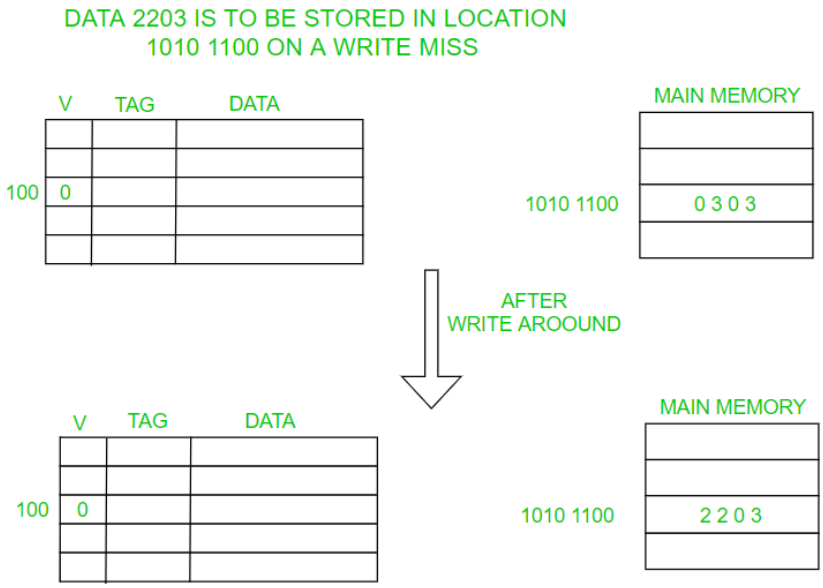


## Handling Write Miss:

1. **Write Allocation:**
   - Data is first loaded into the cache, then updated.
   - Typically used with Write-Back for efficiency.




2. **Write Around:**
   - Data is written directly to memory without affecting the cache.
   - Preferred when the written data is unlikely to be reused soon.



This approach ensures both performance and consistency, depending on the system requirements and workload patterns.

**2. Cache Replacement Algorithms**

**Introduction to Cache Replacement Algorithms**

When a cache becomes full and a new block needs to be loaded, the system must decide which existing block to replace. Different replacement algorithms have been developed to handle this efficiently. Below are the key algorithms:

**1. Least Recently Used (LRU)**

**Description**:
LRU replaces the block that hasn't been used for the longest time. It assumes that the least recently used block is the least likely to be used again in the near future.

**Steps to Apply**:

1. Track the order in which blocks are accessed.
2. When the cache is full, evict the block that was used least recently.

**Example**:
Cache Size = 3, Reference String = **{A, B, C, A, D, B, E}**

| Step | Cache Content | Action |
|---|---|---|
| A | A | Miss |
| B | A, B | Miss |
| C | A, B, C | Miss |
| A | A, B, C | Hit |
| D | B, C, D | Miss (A evicted) |
| B | B, C, D | Hit |
| E | C, D, E | Miss (B evicted) |

**2. Least Frequently Used (LFU)**

**Description**:
LFU replaces the block that has been accessed the least number of times. It keeps a counter for each block to track how frequently it is used.

**Steps to Apply**:

1. Maintain a frequency counter for each block.

2. Evict the block with the smallest counter when the cache is full.
3. If two blocks have the same frequency, use FIFO as a tie-breaker.

**Example**:
Cache Size = 3, Reference String = **{A, B, A, C, B, D, E}**

| Step | Cache Content | Frequency Counters | Action |
|---|---|---|---|
| A | A | A: 1 | Miss |
| B | A, B | A: 1, B: 1 | Miss |
| A | A, B | A: 2, B: 1 | Hit |
| C | A, B, C | A: 2, B: 1, C: 1 | Miss |
| B | A, B, C | A: 2, B: 2, C: 1 | Hit |
| D | B, C, D | B: 2, C: 1, D: 1 | Miss (A evicted, least frequent) |
| E | C, D, E | C: 1, D: 1, E: 1 | Miss (B evicted, least frequent) |

### 3. First In First Out (FIFO)

**Description**:
FIFO replaces the oldest block in the cache, i.e., the block that was loaded first. It operates like a queue.

**Steps to Apply**:

1. Maintain the order in which blocks are loaded into the cache.
2. Evict the oldest block when the cache is full.

**Example**:
Cache Size = 3, Reference String = **{A, B, C, D, E}**

| Step | Cache Content | Action |
|---|---|---|
| A | A | Miss |
| B | A, B | Miss |
| C | A, B, C | Miss |
| D | B, C, D | Miss (A evicted) |
| E | C, D, E | Miss (B evicted) |

### 4. Random Replacement

**Description**:
Random Replacement evicts a randomly selected block from the cache when it is full. This method is simple but may not always result in optimal performance.

**Steps to Apply**:

1. When the cache is full, choose a block randomly for replacement.
2. Evict the randomly chosen block and load the new block.

**Example**:
Cache Size = 3, Reference String = **{A, B, C, D, E}**
Assuming the randomly chosen blocks for replacement:

| Step | Cache Content | Action |
|---|---|---|
| A | A | Miss |
| B | A, B | Miss |
| C | A, B, C | Miss |
| D | A, C, D | Miss (B evicted randomly) |
| E | C, D, E | Miss (A evicted randomly) |

**Comparison of Algorithms**

| Algorithm | Advantages | Disadvantages |
|---|---|---|
| **LRU** | Approximates optimal replacement | Requires tracking recent usage |
| **LFU** | Suitable for repeated access patterns | Can suffer from *cache pollution* (blocks with high initial access remain for too long) |
| **FIFO** | Simple to implement | Can evict frequently used blocks |
| **Random** | Simple and fast | Performance varies depending on randomness |

## Introduction to Pipeline Hazards

- **Definition**: Pipeline hazards are conditions that prevent the next instruction in the pipeline from executing in the next clock cycle.
- **Importance**: Understanding pipeline hazards is crucial for optimizing the performance of pipelined processors.

**Types of Pipeline Hazards**

**1. Data Hazards**

A data hazard occurs when instructions depend on the result of previous instructions that have not yet completed.

**Types of Data Hazards**:

- **Read After Write (RAW)**: Occurs when an instruction needs to read a value that a previous instruction is still writing.
- **Write After Read (WAR)**: Occurs when an instruction writes to a destination before a previous instruction has read it.

- **Write After Write (WAW)**: Occurs when two instructions write to the same destination in an overlapping manner.

**Example**:

assembly

Copy code

Instruction 1: ADD R1, R2, R3 ; R1 = R2 + R3

Instruction 2: SUB R4, R1, R5 ; R4 = R1 - R5

In this case, Instruction 2 depends on the result of Instruction 1. If Instruction 2 tries to read R1 before Instruction 1 completes, a RAW hazard occurs.

**Solutions**:

- **Forwarding/Bypassing**:
  Forwarding sends the result from an intermediate stage (like Execute) directly to a later instruction without waiting for the instruction to complete.
- **Stalling**:
  The pipeline is stalled until the previous instruction completes and produces the required data. This is a simple but inefficient solution.

**Example of Forwarding**: If Instruction 1 completes its execution stage, the result can be forwarded directly to Instruction 2 without waiting for it to be written back to the register.

**2. Control Hazards**

A control hazard occurs when the pipeline makes the wrong decision on branch prediction or when it does not know where to fetch the next instruction due to a branch.

**Example**:

assembly

Copy code

Instruction 1: BEQ R1, R2, LABEL ; Branch if R1 == R2

Instruction 2: ADD R3, R4, R5 ; Next instruction

If the branch is taken, the pipeline has already fetched Instruction 2, which must be discarded, resulting in a control hazard.

**Solutions**:

- **Branch Prediction**:
  The processor guesses whether a branch will be taken or not and continues fetching instructions accordingly.
  - *Static Prediction*: Always assumes a branch is taken or not taken.
  - *Dynamic Prediction*: Uses past behavior to predict future branches.
- **Branch Delay Slot**:
  The instruction following a branch is always executed, regardless of whether the branch is taken or not. The compiler fills this delay slot with a useful instruction.

**Example of Branch Prediction**: If the branch is correctly predicted, the pipeline continues without stalling. If the prediction is wrong, the fetched instructions are discarded, and the correct path is fetched.

**3. Structural Hazards**

A structural hazard occurs when two or more instructions require the same hardware resource at the same time.

**Example**:
If a processor has a single memory unit and two instructions require access to memory simultaneously (e.g., one for fetching an instruction and the other for loading/storing data), a structural hazard occurs.

**Solutions**:

- **Duplicating Resources**:
  Add multiple hardware units to allow concurrent access. For example, separate instruction and data caches eliminate memory conflicts.
- **Pipeline Stalling**:
  Temporarily stall the pipeline until the required resource becomes available.

**Summary of Hazards and Solutions**

| Type of Hazard | Cause | Solution |
|---|---|---|
| **Data Hazard** | Instruction depends on the result of a previous one | Forwarding, Stalling |
| **Control Hazard** | Branch instruction changes the flow of execution | Branch Prediction, Branch Delay Slot |
| **Structural Hazard** | Hardware resource conflict (e.g., memory access) | Resource Duplication, Stalling |

**Solved Examples**

**Example 1: Data Hazard with Forwarding**
Consider the following sequence of instructions:

assembly

Copy code

1. ADD R1, R2, R3 ; R1 = R2 + R3

2. SUB R4, R1, R5 ; R4 = R1 - R5

- Without forwarding, the pipeline will stall Instruction 2 until Instruction 1 writes its result back to R1.
- With forwarding, the result of ADD is directly sent from the Execute stage of Instruction 1 to the Execute stage of Instruction 2, avoiding a stall.

**Example 2: Control Hazard with Branch Prediction**
Consider the following instructions:

assembly

Copy code

1. BEQ R1, R2, LABEL ; Branch if R1 == R2

2. ADD R3, R4, R5 ; Fetched before branch outcome is known

- Assume the processor predicts that the branch is not taken and fetches ADD.
- If the prediction is correct, the pipeline continues without issue.
- If the prediction is incorrect, the ADD instruction is discarded, and the correct instruction from LABEL is fetched.

**References:**


1. Computer System Architecture" by M. Morris Mano
2. Computer Architecture and Organisation" by Er. Rajeev
3. Fundamentals of Computer Architecture" by Parthasarthy & Senthil Kumar
4. Computer Organization and Design: The Hardware/Software Interface" by David A. Patterson and John L. Hennessy
5. Structured Computer Organization" by Andrew S. Tanenbaum and Todd Austin
6. Computer Architecture and Organization" by William Stallings
7. https://www.geeksforgeeks.org
8. https://nptel.ac.in
9. https://en.wikipedia.org
10. https://www.javatpoint.com

- .