Computer Organization and Architecture

Lecture notes

SHAMBHUNATH
Group of Institutions
... Shaping the future

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Prepared by: Rajan Mani Tripathi

UNIT -4

Memory Organization

DETAILED SYLLABUS

Unit-1

Introduction: Functional units of digital system and their interconnections, buses, bus architecture, types of buses and bus arbitration. Register, bus and memory transfer. Processor organization, general registers organization, stack organization and addressing mode.

Unit-2

Arithmetic and logic unit: Look ahead carries adders. Multiplication: Signed operand multiplication, Booths algorithm and array multiplier. Division and logic operations. Floating points arithmetic operation, Arithmetic & logic unit design. IEEE Standard for Floating Point Numbers

Unit-3

Control Unit: Instruction types, formats, instruction cycles and sub cycles (fetch and execute etc.), micro operations, execution of a complete instruction. Program Control, Reduced Instruction Set Computer, Pipelining. Hardwire and micro programmed control: micro programmed sequencing, concept of horizontal and vertical microprogramming.

Unit-4

Memory: Basic concept and hierarchy, semiconductor RAM memories, 2D & 2 1/2D memory organization. ROM memories. Cache memories: concept and design issues & performance, address mapping and replacement Auxiliary memories: magnetic disk, magnetic tape and optical disks Virtual memory: concept implementation..

Unit-5

Input / Output: Peripheral devices, I/O interface, I/O ports, Interrupt type of interrupt and exceptions. Modes of Data Transfer: Program Direct Memory Access., I/O channels and processors. Serial asynchronous communication, standard communication interfaces.

Memory Organization

Introduction:

Memory unit is an essential component of digital computer science it is needed for storing programs and data. Two or three level of memory such as

- Main Memory
- Secondary memory and
- Catch Memory

Are provided in digital computers. The main memory is the fast memory. It stores programs along with data, which are to be executed. it also stores necessary programs of the system software, which are required to execute the user program . The catch memory is placed in between the CPU and main memory. It is much faster than the main memory. Secondary memory is permanent storage used to store programs and data that is used infrequent.

Memory Device Characteristics:

To identify the behavior of various memories certain characteristics are considered. These are given a

Memory Type: on the basis of the location inside the computer, memory can be placed in four groups as:

CPU Registers:

These high-speed registers in the cpu work as memory for temporary storage of instruction and data. They usually form a general purpose register file for storing data as it is processed. The data can be read from or written into a register within a single clock cycle.

Main Memory or Primary Memory:

Main memory size is large and fast accessing external memory stores programs and data. Storage locations in main memory are addressed directly by the CPU's load and store instructions. Main memory is slower compare to CPU registers because of main memory has large storage capacity is typically 1 and 2^{10} megabytes.

Secondary memory:

This memory has larger is capacity but slower than main memory. Secondary memory stores system programs, large data file and like the data are not continually required by the cpu. It

also acts as an overflow memory when the capacity of the main memory is exceeded. Information in secondary storage is accessed indirectly via input-output processor (IOP) that transfers information between main and secondary memory.

Cache Memory:

Most computer have another level of ic memory called cache memory. It is placed between the CPU register and main memory. A cache memory capacity is less than that of main memory but it is faster than main memory because some or all of it can reside on the same IC as the CPU. Cache memories are essential components of high-performance computers.

Location: - the memory which is inside the processor called the internal memory. The memory which is external to processor is known as external memory.

Access Method: Each memory is collection of various memory locations. Accessing the memory means finding and reaching the desired location and then reading information from the memory location the information from the location can be acceded in the following ways:

1. **Random Access**
   It is the access mode where each memory location has a unique address. Using these unique addresses each memory location can be addressed independently in any order in equal amount of time. That is whether the desired location is in beginning of memory or it is in the end the access time of information from both the location would be same. This is done with the help of access mechanism which is separate for each location. Generally, main memories are random access memories.
2. **Sequential Access**
   If storage locations can be accessed only in a certain predetermined sequence, the access method is known as serial access or sequential access
3. **Direct Access**
   In direct access information is stored on tracks and each track has a separate read/write head. Using that read/write head information of that track would be accessed sequentially. This feature makes it a semi-random mode which is generally used in magnetic disks.

Access time. Access time is a measure of the time required to read from or write the data to a particular address in the memory.

Destructive readout: When data is read from memory, the stored data is extracted (removed) from memory and in the process the data is erased in the source. Because the data is lost, the

process is referred to as destructive readout. If it is desired to restore the same data at the same storage location, the word must be rewritten after reading.

Non-destructive readout: If the data in a memory is not destroyed in the reading process, the system has non-destructive readout. This mean the data can be read over and over again without being rewritten.

Volatile memories: Non-volatile memories are memories that do not lose their contents when power removed.

Non- volatile memories: - Non-volatile memories are memories that do not lose their contents when power is removed

Memory Hierarchy: In the Computer System Design, Memory Hierarchy is an enhancement to organize the memory such that it can minimize the access time. The Memory Hierarchy was developed based on a program behavior known as locality of references. The figure below clearly demonstrates the different levels of memory hierarchy:

1. **External Memory or Secondary Memory –**
   Comprising of Magnetic Disk, Optical Disk, Magnetic Tape i.e. peripheral storage devices which are accessible by the processor via I/O Module.
2. **Internal Memory or Primary Memory –**
   Comprising of Main Memory, Cache Memory & CPU registers. This is directly accessible by the processor.

We can infer the following characteristics of Memory Hierarchy Design from above figure:

1. **Capacity:**
   It is the global volume of information the memory can store. As we move from top to bottom in the Hierarchy, the capacity increases.
2. **Access Time:**
   It is the time interval between the read/write request and the availability of the data. As we move from top to bottom in the Hierarchy, the access time increases.
3. **Performance:**
   Earlier when the computer system was designed without Memory Hierarchy design, the speed gap increases between the CPU registers and Main Memory due to large difference in access time. This results in lower performance of the system and thus, enhancement was required. This enhancement was made in the form of Memory Hierarchy Design because of which the performance of the system increases. One of the most significant ways to increase system performance is minimizing how far down the memory hierarchy one has to go to manipulate data.
4. **Cost per bit:**
   As we move from bottom to top in the Hierarchy, the cost per bit increases i.e. Internal Memory is costlier than External Memory.

Memory Classification

In general, the memory is classified in two types based on their mode of access of a memory system.

1. Random access memory
2. Sequential access memory

Random Access memory: The word of data reading or writing from or to the memory requires same time. We can access the data randomly

Example: Hard Disk.

Sequential access memory: the information stored in some medium is not immediately accessible but is available at certain intervals of time. The time it takes to access a word depends on the position of the word with respect the riding geed position: the fore the access time is variable.

Example: magnetic tape

Page Replacement Algorithms in Operating Systems

In an operating system that uses paging for memory management, a page replacement algorithm is needed to decide which page needs to be replaced when new page comes in.

Page Fault – A page fault happens when a running program accesses a memory page that is mapped into the virtual address space, but not loaded in physical memory.
Since actual physical memory is much smaller than virtual memory, page faults happen.

In case of page fault, Operating System might have to replace one of the existing pages with the newly needed page.

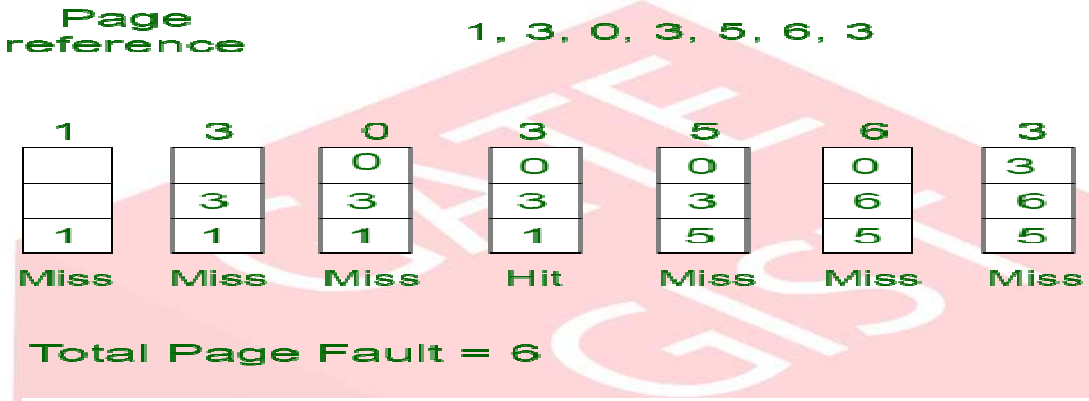
Different page replacement algorithms suggest different ways to decide which page to replace.

The target for all algorithms is to reduce the number of page faults.

Page Replacement Algorithms:

First In first Out (FIFO) –

This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue; the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.
Example-1 Consider page reference string 1, 3, 0, 3, 5, 6 with 3 page frames. Find number of page faults.

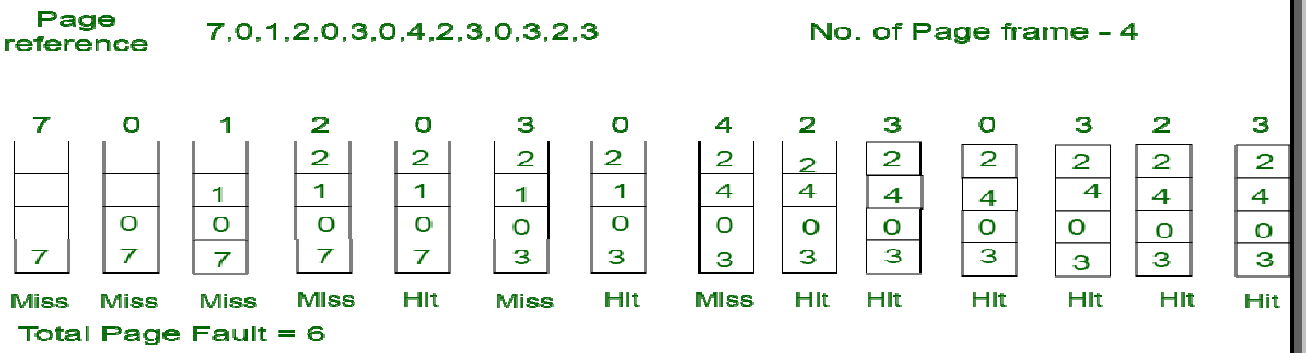


- Initially all slots are empty, so when 1, 3, 0 came they are allocated to the empty slots —> **3 Page Faults.**
- When 3 comes, it is already in memory so —> **0 Page Faults.**
- Then 5 comes, it is not available in memory so it replaces the oldest page slot i.e 1. —>**1 Page Fault.**
- 6 comes, it is also not available in memory so it replaces the oldest page slot i.e 3 —>**1 Page Fault.**
- Finally when 3 come it is not available so it replaces 0 **1 page fault**

Belady's anomaly – Belady's anomaly proves that it is possible to have more page faults when increasing the number of page frames while using the First in First out (FIFO) page replacement algorithm. For example, if we consider reference string 3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4 and 3 slots, we get 9 total page faults, but if we increase slots to 4, we get 10 page faults.

Optimal Page replacement –

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.
Example-2: Consider the page references 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, with 4 page frame. Find number of page fault.



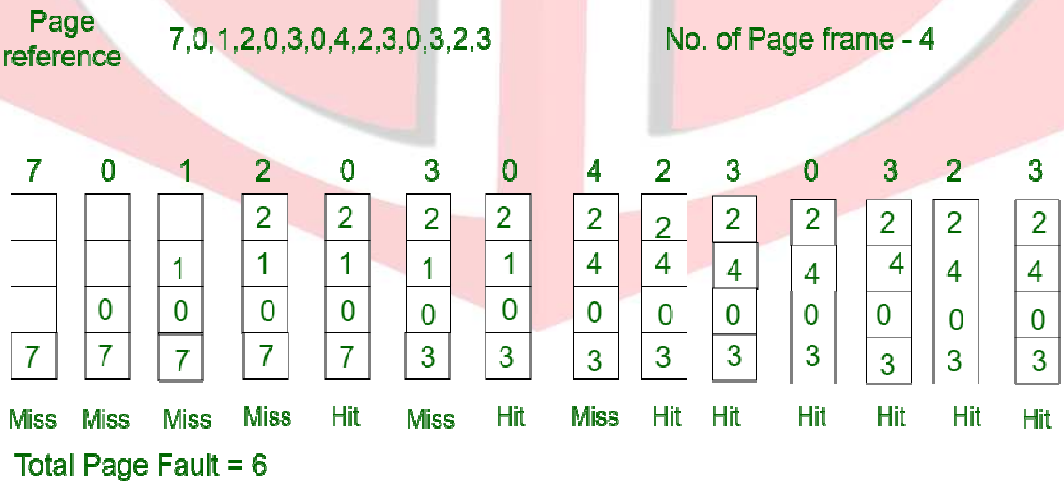
- Initially all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**
- 0 is already there so —> **0 Page fault.**
- when 3 came it will take the place of 7 because it is not used for the
- longest duration of time in the future.—>**1 Page fault.**
- 0 is already there so —> **0 Page fault.**.
- 4 will take place of 1 —> **1 Page Fault.**
- Now for the further page reference string —> **0 Page fault** because they are already available in the memory.
- Optimal page replacement is perfect, but not possible in practice as the operating system cannot know future requests.

The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analyzed against it.

Least Recently Used –

In this algorithm page will be replaced which is least recently used.

Example-3 Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2,



Here LRU has same number of page fault as optimal but it may differ according to question.

3, 0, 3, 2 with 4 page frames.Find number of page faults.

- Initially all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**
- 0 is already their so —> **0 Page fault.**
- when 3 came it will take the place of 7 because it is least recently used —>**1 Page fault**
- 0 is already in memory so —> **0 Page fault**.
- 4 will takes place of 1 —> **1 Page Fault**
- Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

Cache Mapping-

- Cache mapping defines how a block from the main memory is mapped to the cache memory in case of a cache miss.

OR

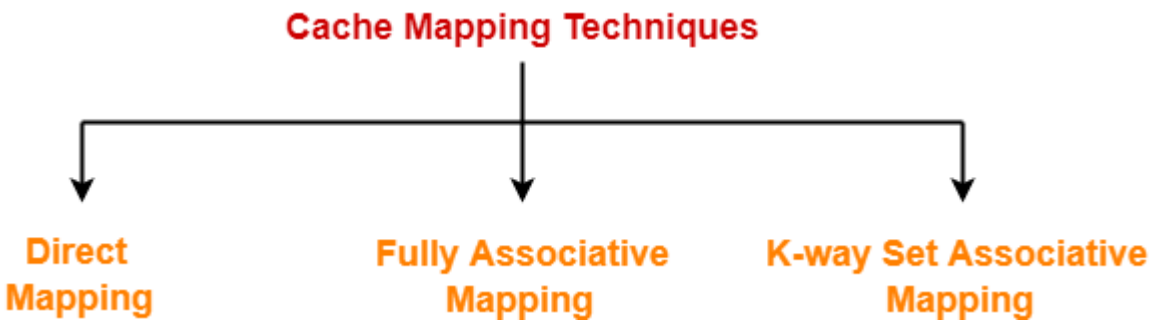
- Cache mapping is a technique by which the contents of main memory are brought into the cache memory.

NOTE:

- Main memory is divided into equal size partitions called as **blocks** or **frames**.
- Cache memory is divided into partitions having same size as that of blocks called as **lines**.
- During cache mapping, block of main memory is simply copied to the cache and the block is not actually brought from the main memory.

Cache Mapping Techniques-

Cache mapping is performed using following three different techniques



- Direct Mapping

- Fully Associative Mapping
- K-way Set Associative Mapping

1. Direct Mapping-

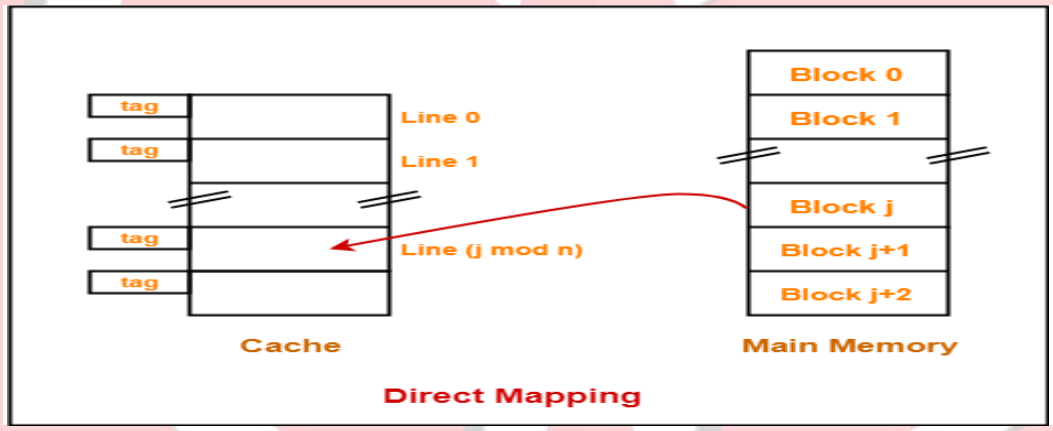
In direct mapping,

- A particular block of main memory can map only to a particular line of the cache.
- The line number of cache to which a particular block can map is given by-

- **Cache line number**
  = (Main Memory Block Address) Modulo (Number of lines in Cache)

Example-

- Consider cache memory is divided into 'n' number of lines.
- Then, block 'j' of main memory can map to line number (j mod n) only of the cache.



Need of Replacement Algorithm-

In direct mapping,

- There is no need of any replacement algorithm.
- This is because a main memory block can map only to a particular line of the cache.
- Thus, the new incoming block will always replace the existing block (if any) in that particular line.

Division of Physical Address-

In direct mapping, the physical address is divided as-

Division of Physical Address in Direct Mapping

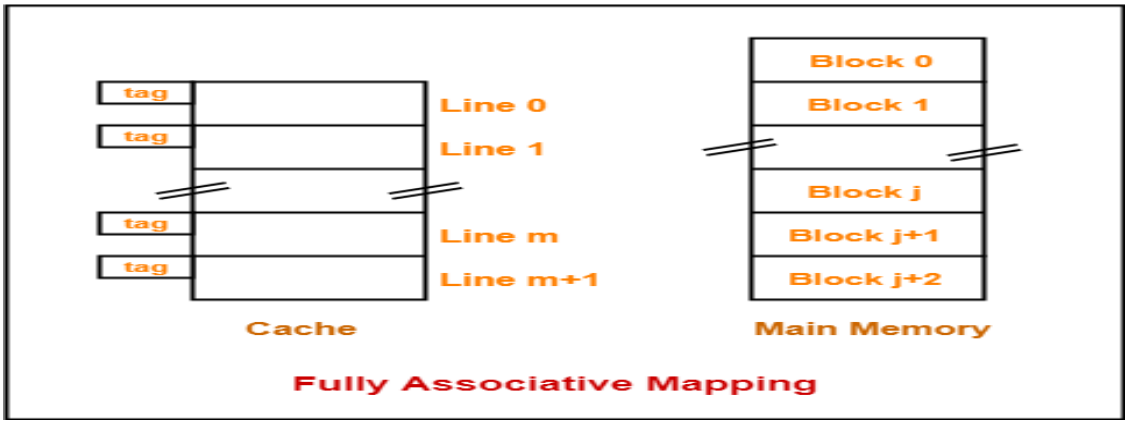
2. Fully Associative Mapping-

In fully associative mapping,

- A block of main memory can map to any line of the cache that is freely available at that moment.
- This makes fully associative mapping more flexible than direct mapping.

Example-

Consider the following scenario-



Fully Associative Mapping

Here,

- All the lines of cache are freely available.
- Thus, any block of main memory can map to any line of the cache.
- Had all the cache lines been occupied, then one of the existing blocks will have to be replaced.

Need of Replacement Algorithm-

In fully associative mapping,

- A replacement algorithm is required.
- Replacement algorithm suggests the block to be replaced if all the cache lines are occupied.
- Thus, replacement algorithm like FCFS Algorithm, LRU Algorithm etc is employed.

Division of Physical Address-

In fully associative mapping, the physical address is divided as-

Division of Physical Address in Fully Associative Mapping

3. K-way Set Associative Mapping-

In k-way set associative mapping,

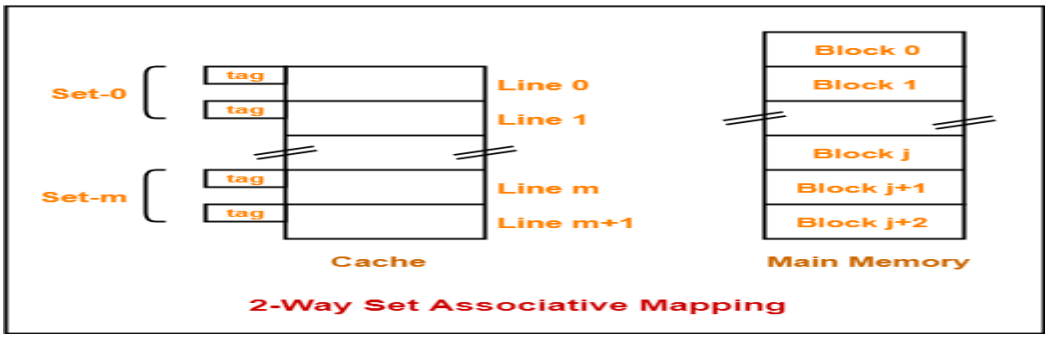
- Cache lines are grouped into sets where each set contains k number of lines.
- A particular block of main memory can map to only one particular set of the cache.
- However, within that set, the memory block can map any cache line that is freely available.
- The set of the cache to which a particular block of the main memory can map is given by-

Cache set number

= (Main Memory Block Address) Modulo (Number of sets in Cache)

Example-

Consider the following example of 2-way set associative mapping-



Here,

- k = 2 suggests that each set contains two cache lines.
- Since cache contains 6 lines, so number of sets in the cache = 6 / 2 = 3 sets.
- Block 'j' of main memory can map to set number (j mod 3) only of the cache.
- Within that set, block 'j' can map to any cache line that is freely available at that moment.
- If all the cache lines are occupied, then one of the existing blocks will have to be replaced.

Need of Replacement Algorithm-

- Set associative mapping is a combination of direct mapping and fully associative mapping.
- It uses fully associative mapping within each set.
- Thus, set associative mapping requires a replacement algorithm.

Division of Physical Address-

In set associative mapping, the physical address is divided as-

Tag	Set Number	Block / Line Offset

Division of Physical Address in K-way Set Associative Mapping

Special Cases-

- If k = 1, then k-way set associative mapping becomes direct mapping i.e.

1-way Set Associative Mapping ≡ Direct Mapping

- If k = Total number of lines in the cache, then k-way set associative mapping becomes fully associative mapping.

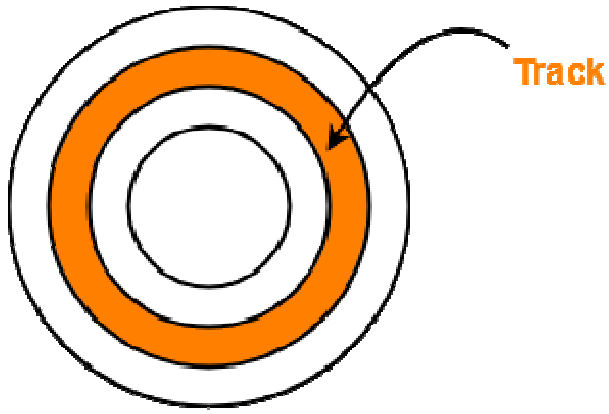
Auxiliary Memory (Magnetic Disk)

Magnetic disk is a storage device that is used to write, rewrite and access data.

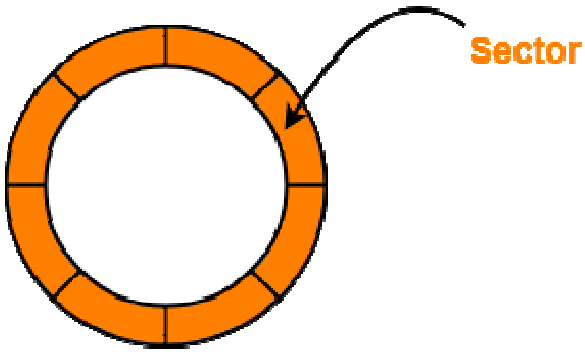
- It uses a magnetization process.

Architecture-

- The entire disk is divided into **platters**.
- Each platter consists of concentric circles called as **tracks**.
- These tracks are further divided into **sectors** which are the smallest divisions in the disk.

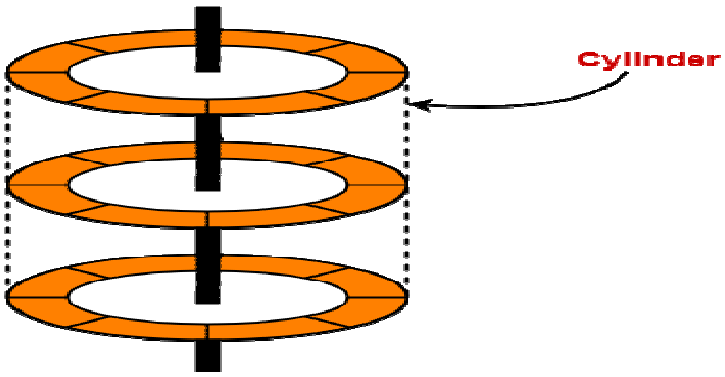


Disk divided into tracks

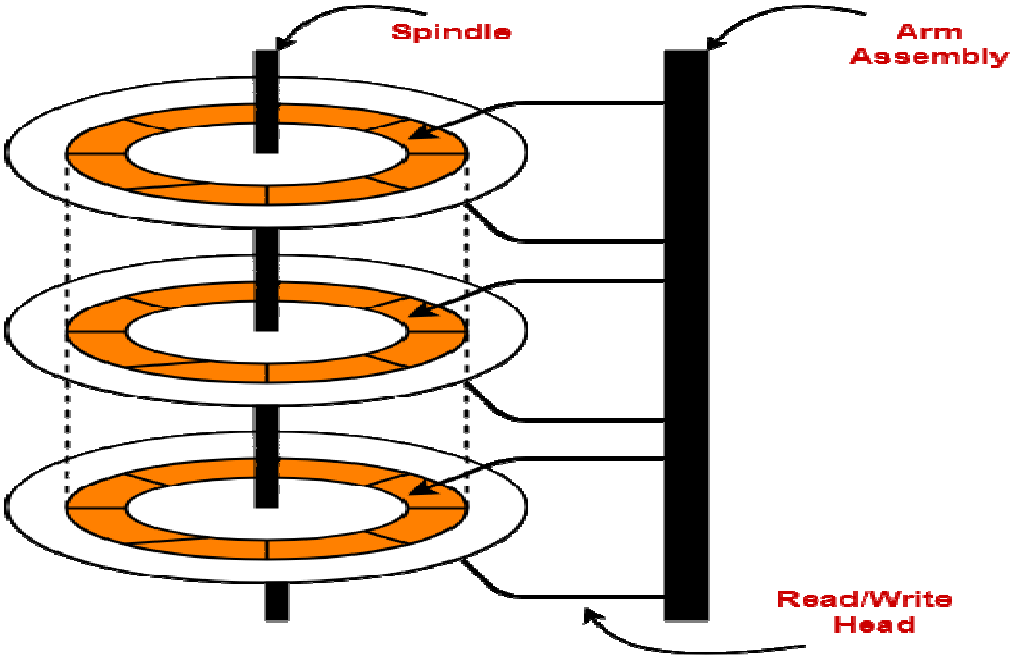


Track divided into sectors

- A **cylinder** is formed by combining the tracks at a given radius of a disk pack.



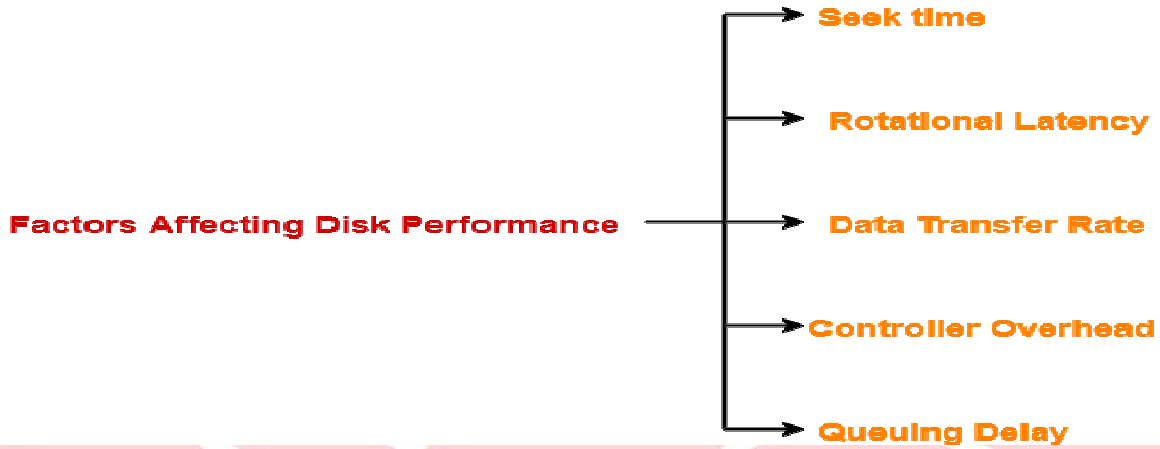
- There exists a mechanical arm called as **Read / Write head**.
- It is used to read from and write to the disk.
- Head has to reach at a particular track and then wait for the rotation of the platter.
- The rotation causes the required sector of the track to come under the head.
- Each platter has 2 surfaces- top and bottom and both the surfaces are used to store the data.
- Each surface has its own read / write head.



Disk Performance Parameters-

The time taken by the disk to complete an I/O request is called as **disk service time** or **disk access time**.

Components that contribute to the service time are-



1. Seek time
2. Rotational latency
3. Data transfer rate
4. Controller overhead
5. Queuing delay

1. Seek Time-

- The time taken by the read / write head to reach the desired track is called as **seek time**.
- It is the component which contributes the largest percentage of the disk service time.
- The lower the seek time, the faster the I/O operation.

Specifications

Seek time specifications include-

1. Full stroke
2. Average
3. Track to Track

1. Full Stroke-

- It is the time taken by the read / write head to move across the entire width of the disk from the innermost track to the outermost track

2. Average-

- It is the average time taken by the read / write head to move from one random track to another.

Average seek time = 1 / 3 x Full stroke

3. Track to Track-

- It is the time taken by the read-write head to move between the adjacent tracks.

2. Rotational Latency-

- The time taken by the desired sector to come under the read / write head is called as **rotational latency**.
- It depends on the rotation speed of the spindle.

Average rotational latency = 1 / 2 x Time taken for full rotation

3. Data Transfer Rate-

- The amount of data that passes under the read / write head in a given amount of time is called as **data transfer rate**.
- The time taken to transfer the data is called as **transfer time**.

It depends on the following factors-

1. Number of bytes to be transferred
2. Rotation speed of the disk
3. Density of the track
4. Speed of the electronics that connects the disk to the computer

4. Controller Overhead-

- The overhead imposed by the disk controller is called as **controller overhead**.
- Disk controller is a device that manages the disk.

5. Queuing Delay-

- The time spent waiting for the disk to become free is called as **queuing delay**.

NOTE-

All the tracks of a disk have the same storage capacity.

Storage Density-

- All the tracks of a disk have the same storage capacity.
- This is because each track has different storage density.
- Storage density decreases as we from one track to another track away from the center.