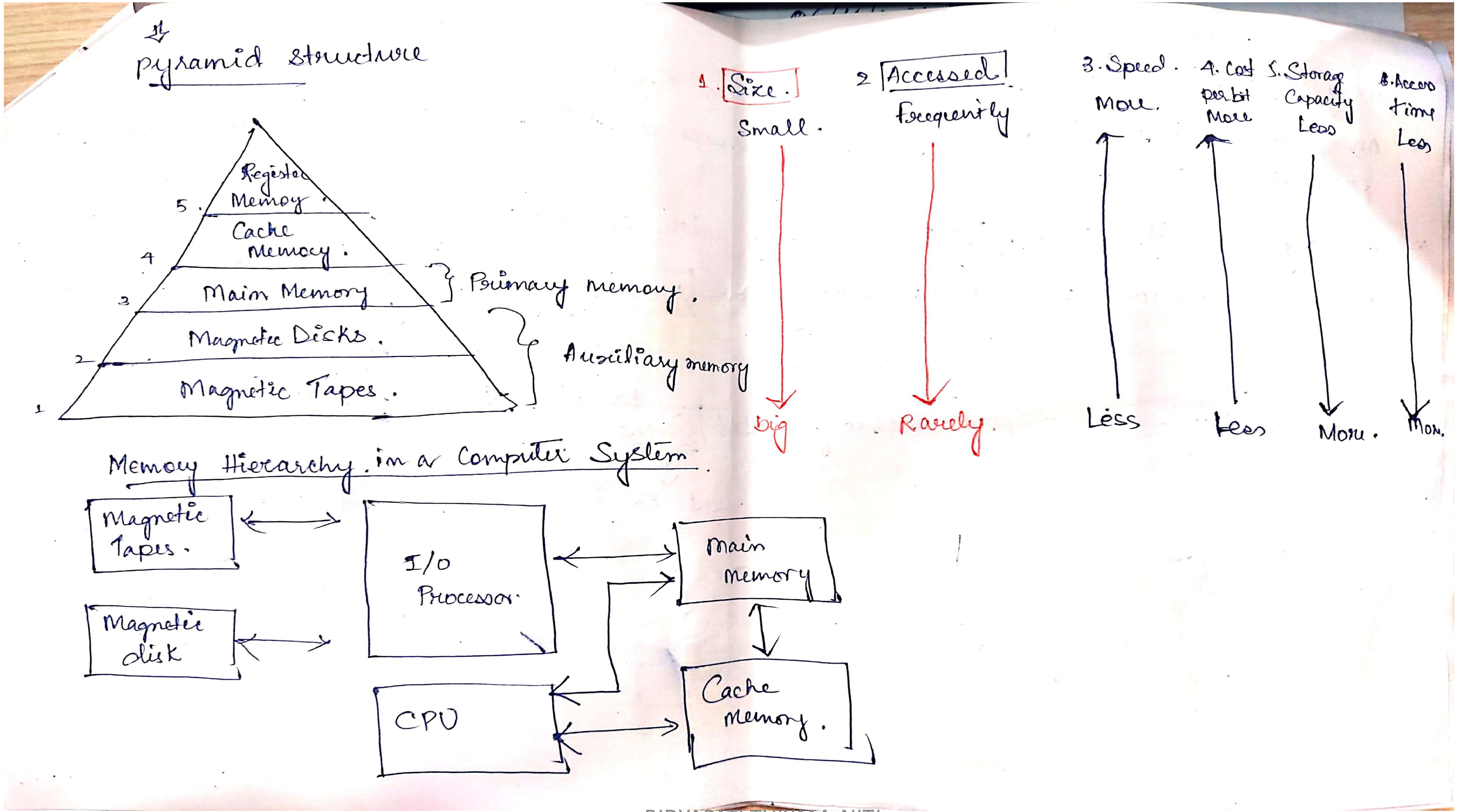
Memory Organization

Unit-4

Memory Organization in Computer Architecture

- A memory unit is the collection of storage units or devices together.
- The memory unit stores the binary information in the form of bits.
- Generally, memory/storage is classified into 2 categories:
  - Volatile Memory: This loses its data, when power is switched off.
  - Non-Volatile Memory: This is a permanent storage and does not lose any data when power is switched off.

MEMORY HIERARCHY



- Memory is used for storing programs and data that are required to perform a specific task.
- For CPU to operate at its maximum speed, it required an uninterrupted and high speed access to these memories that contain programs and data.
- Some of the criteria need to be taken into consideration while deciding which memory is to be used:
    - Cost
    - Speed
    - Memory access time
    - size
    - storage capacity

Memory Access Methods

- Each memory type, is a collection of numerous memory locations. To access data from any memory, first it must be located and then the data is read from the memory location.
- Following are the methods to access information from memory locations:

1. Random Access: Main memories are random access memories, in which each memory location has a unique address. Using this unique address any memory location can be reached in the same amount of time in any order.

2. Sequential Access: This methods allows memory access in a sequence or in order.

3. Direct Access: In this mode, information is stored in tracks, with each track having a separate read/write head.

Types of Memory:

- A computer system contains various types of memories like auxiliary memory, cache memory, register memory and main memory.

1. Auxiliary Memory

- The auxiliary memory is at the bottom and is not connected with the CPU directly. However, being slow, it is present in large volume in the system due to its low pricing. This memory is basically used for storing the programs that are not needed in the main memory. This helps in freeing the main memory which can be utilized by other programs that needs main memory. The main function of this memory is to provide parallel searching that can be used for performing a search on an entire word.

2. Main Memory

- The main memory is at the second level of the hierarchy. Due to its direct connection with the CPU, it is also known as central memory. The main memory holds the data and the programs that are needed by the CPU. The main memory mainly consists of RAM, which is available in staticand dynamic mode.

3. Cache Memory

- Cache memory is at the top level of the memory hierarchy. This is a high speed memory used to increase the speed of processing by making current programs and data available to the CPU at a rapid rate. Cache memory is usually placed between the CPU and the main memory.

4. Register memory

- Register memory is the smallest and fastest memory in a computer. It is located in the CPU in the form of registers. It temporarily holds frequently used data, instructions and memory address that can be quickly accessd by the CPU.

Main Memory:

- Central storage unit in a computer system
- Large memory
- Made up of Integrated chips
- Types of main memory:
    1. RAM (Random access memory)
    2. ROM (Read only memory)

RAM (Random Access Memory)

- Random access memory (RAM) is the best known form of computer memory.
- RAM is considered "random access" because you can access any memory cell directly if we know the row and column that intersect at that cell.
- Types of RAM:-

  1. Static RAM (SRAM)
  2. Dynamic RAM (DRAM)

1. Static RAM (SRAM)
    – a bit of data is stored using the state of a flip-flop.
    – Retains value indefinitely, as long as it is kept powered.
    – Mostly uses to create cache memory of CPU.
    – Faster and more expensive than DRAM.
2. Dynamic RAM (DRAM)
    – Each cell stores bit with a capacitor and transistor.
    – Large storage capacity
    – Needs to be refreshed frequently.
    – Used to create main memory.
    – Slower and cheaper than SRAM

ROM (Read- Only- Memory)

- ROM is used for storing programs that are Permanently resident in the computer and for tables of constants that do not change in value once the production of the computer is completed
- The ROM portion of main memory is needed for storing an initial program called bootstrap loader, witch is to start the computer software operating when power is turned on.
- There are five basic ROM types:
    - ROM - Read Only Memory
    - PROM - Programmable Read Only Memory
    - EPROM - Erasable Programmable Read Only Memory
    - EEPROM - Electrically Erasable Programmable Read Only Memory
    - Flash EEPROM memory

Auxiliary Memory

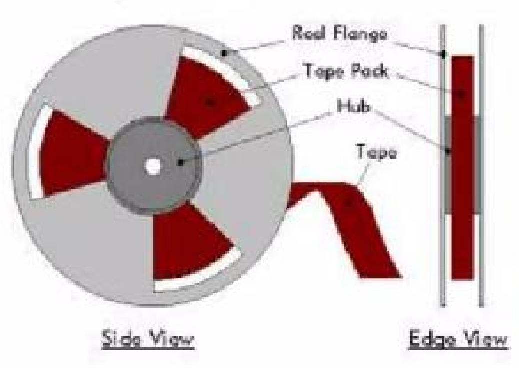
- Devices that provide backup storage are called auxiliary memory.
- For example: Magnetic disks and tapes are commonly used auxiliary devices.
- Other devices used as auxiliary memory are magnetic drums, magnetic bubble memory and optical disks.
- It is not directly accessible to the CPU, and is accessed using the Input / Output channels.

Magnetic Disk (Hard Disk, Floppy Disk)

- It is a circular Plate constructed of metal or plastic coated magnetized material.
- Both sides of disks are used.
- Several disks may be stacked on one spindle, with Read-Write head available on each surface.
- All sides rotate together at high speed.
- Bits are stored in magnetized surface along concentric circles are called Tracks. Tracks are divided into sections called Sectors

Magnetic Tap

- In magnetic tape only one side of the ribbon is used for storing data.
- It is sequential memory which contains thin plastic ribbon to store data and coated by magnetic oxide.
- Data read/write speed is slower because of sequential access.
- It is highly reliable which requires magnetic tape drive writing and reading data.



Locality of Reference

Locality of reference is manifested in two ways :

1. **Temporal**- means that a recently executed instruction is likely to be executed again very soon.

- The information which will be used in near future is likely to be in use already(e.g. reuse of information in loops)

2. **Spatial** - means that instructions in close proximity to a recently executed instruction are also likely to be executed soon.

- If a word is accessed, adjacent (near) words are likely to be accessed soon (e.g. related data items (arrays) are usually stored together; instructions are executed sequentially)

- If active segments of a program can be placed in afast (cache) memory , then total execution time can be reduced significantly
- Temporal Locality of Reference suggests whenever an information (instruction or data) is needed. first , this item should be brought in to cache
- Spatial aspect of Locality of Reference suggests that instead of bringing just one item from the main memory to the cache ,it is wise to bring several items that reside at adjacent addresses as well (ie a block of information)

Cache Memory & Performance:

- **Principles of cache:**

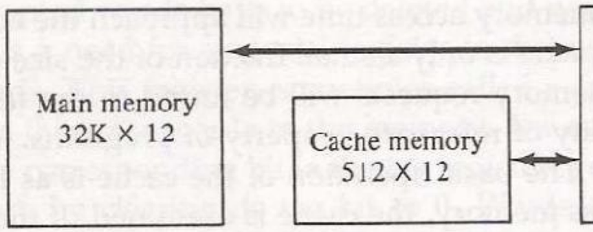


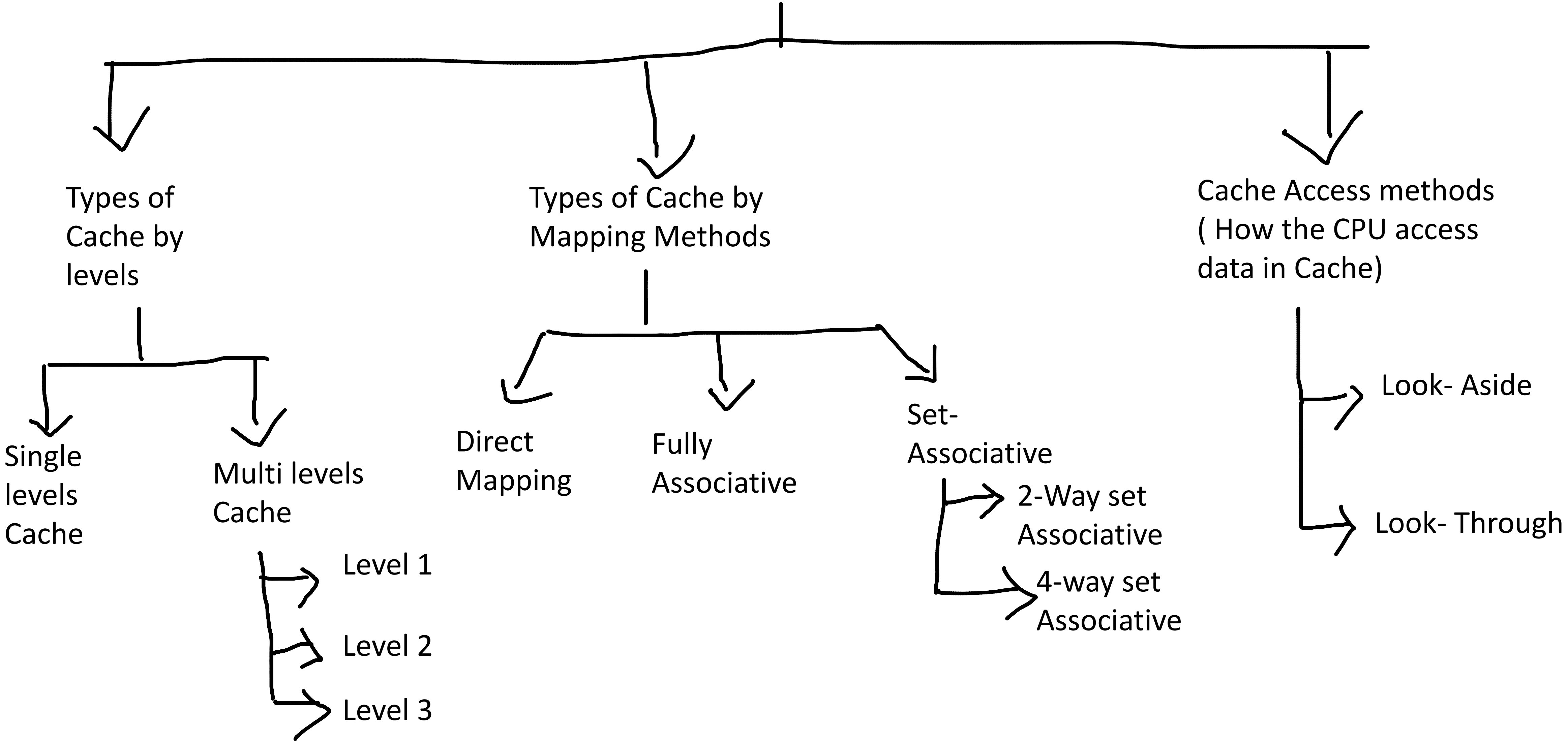
Figure 12-10 Example of cache memory.

- ✓ The main memory can store 32k words of 12 bits each. The cache is capable of storing 512 of these words at any given time. For every word stored , there is a duplicate copy in main memory. The CPU communicates with both memories. It first sends a 15 bit address to cache. If there is a hit, the CPU accepts the 12 bit data from cache. If there is a miss, the CPU reads the word from main memory and the word is then transferred to cache.
- ✓ When a read request is received from CPU,contents of a block of memory words containing the location specified are transferred in to cache. When the program references any of the locations in this block , the contents are read from the cache Number of blocks in cache is smaller than number of blocks in main memory
- ✓ Correspondence between main memory blocks and those in the cache is specified by a mapping function
- ✓ Assume cache is full and memory word not in cache is referenced
- ✓ Control hardware decides which block from cache is to be removed to create space for new block containing referenced word from memory.
- ✓ Collection of rules for making this decision is called “Replacement algorithm ”

Cache Memory:

- If the active portions of the program and data are placed in a fast small memory, the average memory access time can be reduced . Thus reducing the total execution time of the program such a fast small memory is referred to as cache memory
- The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component
- When CPU needs to access memory, the cache is examined
- If the word is found in the cache, it is read from the fast memory
- If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word

Cache Organization



Cache Access methods:

- Look-aside and look-through caches are two different types of cache architectures that differ in how they handle data requests:

1. Look-aside cache

- This cache is parallel to main memory, and both the cache and main memory see the bus cycle at the same time. When the processor requests data, the cache checks to see if the address is in the cache:

 - Cache hit: The cache contains the memory location, so it responds to the read cycle and terminates the bus cycle.

 - Cache miss: The cache does not contain the memory location, so main memory responds to the processor and terminates the bus cycle. The cache then sniffs the data so that it can be used the next time the processor requests it.

- The Fig. 8.4.2 shows system of look-aside cache organization. Here,the cache and the main memory are directly connected to the system bus.
- In this system, the CPU initiates a memory access by placing a physical address on the memory address bus at the start of read or write cycle.
- The cache memory M1 immediately compares physical address to the tag addresses currently residing in its tag memory. If a match is found, i.e., in case of cache hit, the access is completed by a read or write operation executed in the cache. The main memory is not involved in the process of read or write.
- If match is not found, i.e., in case of cache miss, the desired access is completed by a read or write operation directed to M2. In response to cache miss, a block of data that includes the target address is transferred from M2 to M1. The system bus is used for this transfer and hence it is unavailable for other uses like I/O operations.

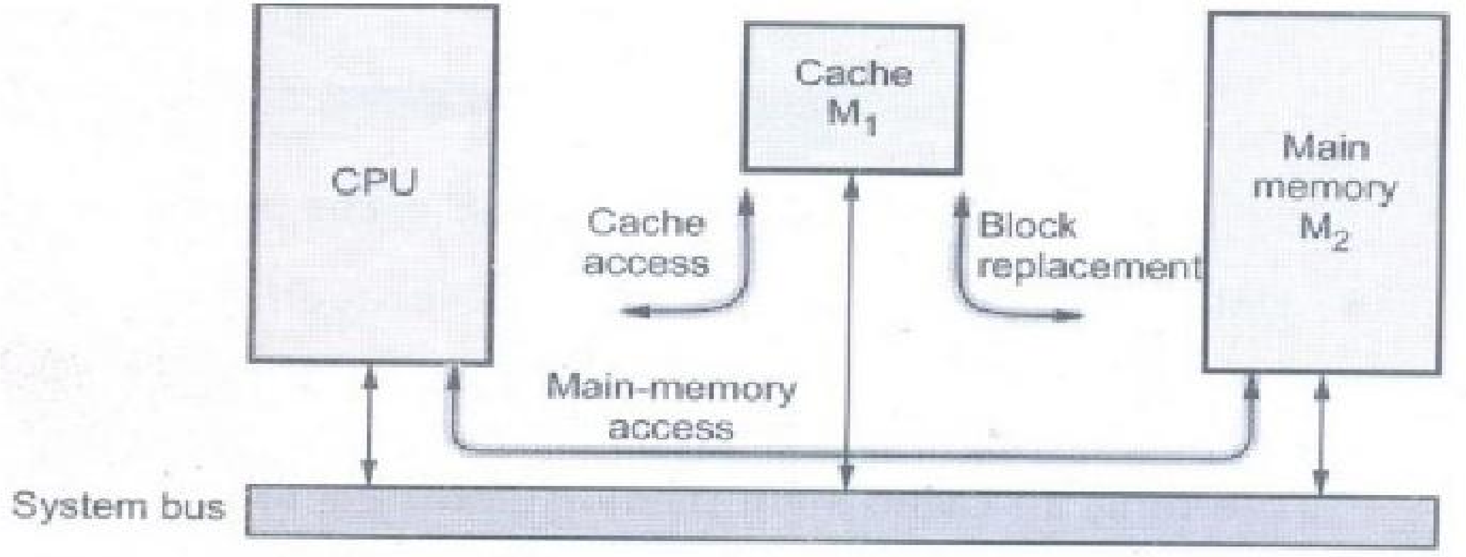


Fig. 8.4.2 Look-aside cache system organization

2. **Look-through cache**

- The Fig. 8.4.3 shows look-through system of cache organization. Here, the CPU communicates with cache via a separate (local) bus which is isolated from the main system bus. Thus during cache accesses, the system bus is available for use by other units, such as I/O controllers, to communicate with main memory.

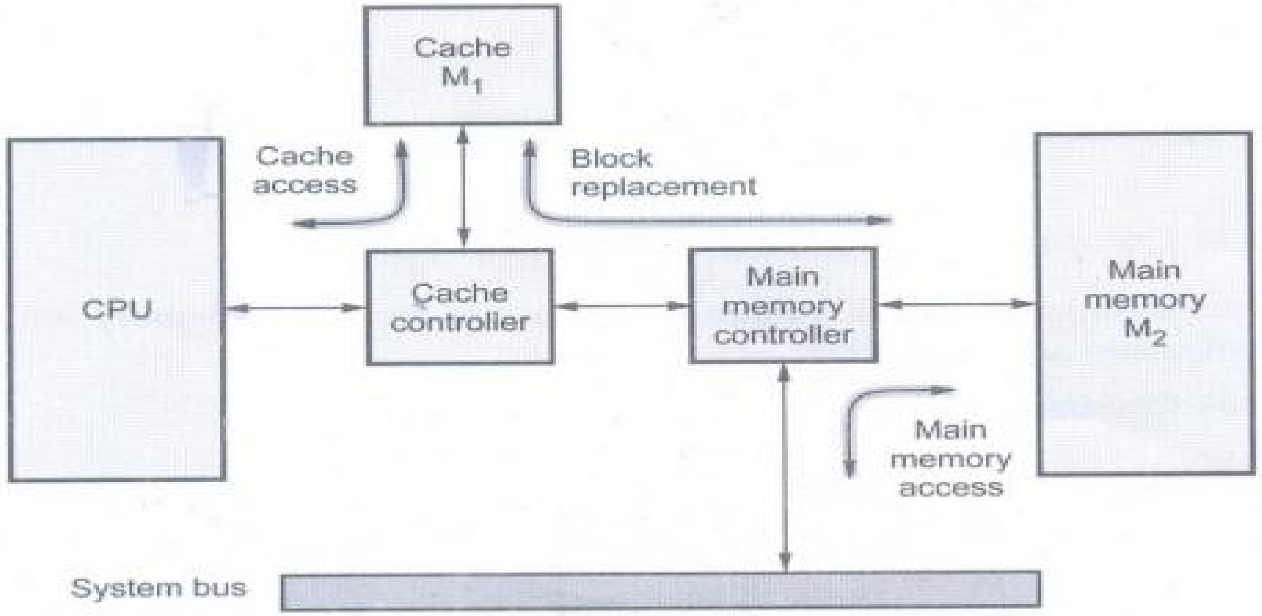


Fig. 8.4.3 Look-through cache system organization

- Unlike the look-aside system, look-through cache system does not automatically send all memory requests to main memory; it does so only after a cache miss.
- A look-through cache systems use wider local bus to link M1 and M2, thus speeding up cache-main-memory transfers (block transfers).
- Look-through cache system is faster.

- Disadvantages:

  - It is complex.
  - It is costly.
  - It takes longer time for M2 to respond to the CPU when a cache miss occurs.

Types of Cache Levels:

Cache memory is categorized in levels based on it's closeness and accessibility to the microprocessor.

There are three levels of a cache.

1. **Level 1(L1) Cache:**

- This cache is inbuilt in the processor and is made of SRAM(Static RAM) .
- Each time the processor requests information from memory, the cache controller on the chip uses special circuitry to first check if the memory data is already in the cache.
- If it is present, then the system is spared from time consuming access to the main memory. In a typical CPU, primary cache ranges in size from 8 to 64 KB, with larger amounts on the newer processors.
- This type of Cache Memory is very fast because it runs at the speed of the processor since it is integrated into it.

2. **Level 2(L2) Cache:**

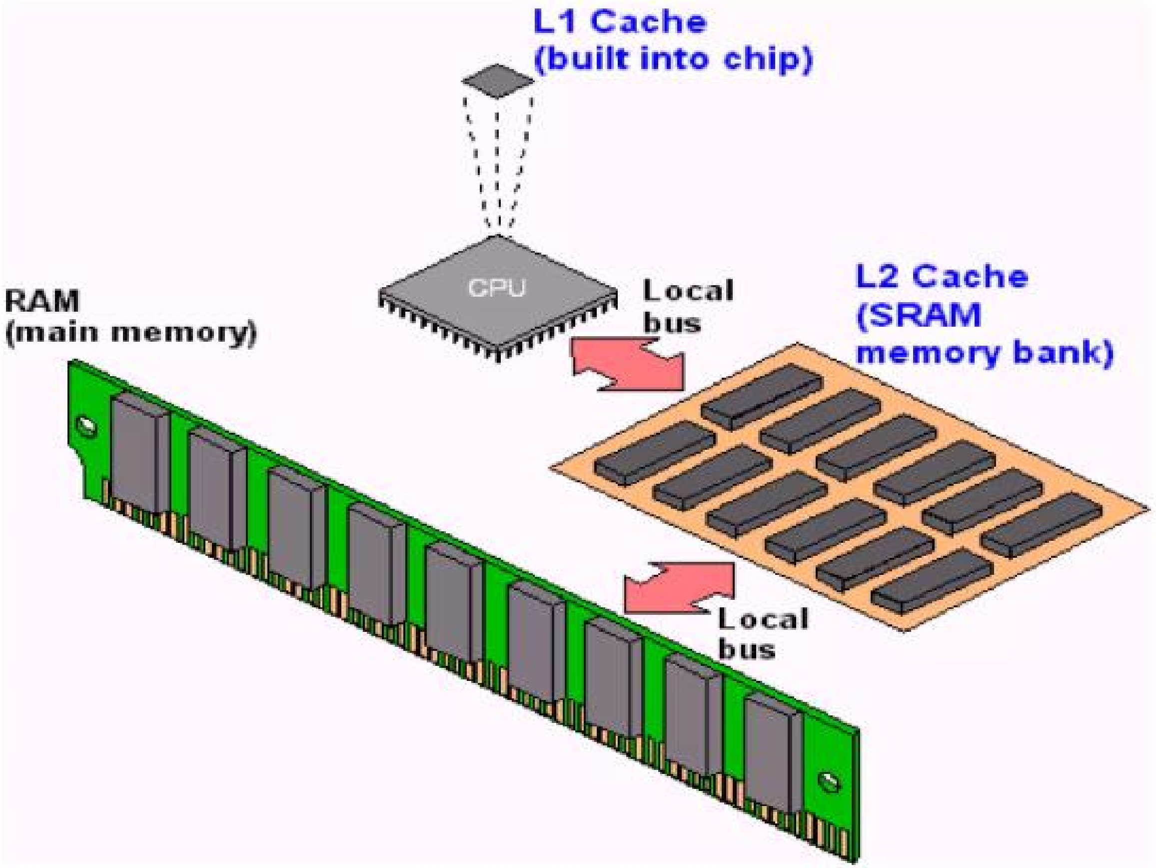
- The L2 cache is larger but slower in speed than L1 cache.
- It is used to see recent accesses that is not picked by L1 cache and is usually 64 to 2 MB in size.

- A L2 cache is also found on the CPU.
- If L1 and L2 cache are used together, then the missing information that is not present in L1 cache can be retrieved quickly from the L2 cache.
- Like L1 caches, L2 caches are composed of SRAM but they are much larger.
- L2 is usually a separate static RAM (SRAM) chip and it is placed between the CPU & DRAM(Main Memory)

3. Level 3(L3) Cache:

- L3 Cache memory is an enhanced form of memory present on the motherboard of the computer.
- It is an extra cache built into the motherboard between the processor and main memory to speed up the processing operations.
- It reduces the time gap between request and retrieving of the data and instructions much more quickly than a main memory.
- L3 cache are being used with processors nowadays, having more than 3 MB of storage in it.

Diagram showing different types of cache and their position in the computer system



Types of Mapping methods:

- DIFFERENT ORGANIZATION TECHNIQUES USED FOR CACHE MEMORY.
- THE THREE TECHNIQUES ARE :

    1. DIRECT MAPPING.
    2. FULLY ASSOCIATIVE MAPPING
    3. SET-ASSOCIATIVE MAPPING

1. **Direct mapping**

- This is the simplest among the three techniques. Its simplicity stems from the fact that it places an incoming main memory block into a specific fixed cache block location.
- The placement is done based on a fixed relation between the incoming block number, i, the cache block number, j, and the number of cache blocks, N: $j = i \bmod N$

Direct mapping : Example 1

__Example 1:__ Consider, for example, the case of a main memory consisting of **4K** blocks, a cache memory consisting of **128** blocks, and a block size of **16** words.
As the figure shows, there are a total of 32 main memory blocks that map to
a given cache block. For example, main memory blocks 0, 128, 256, 384, . . . , 3968 map to cache block 0.

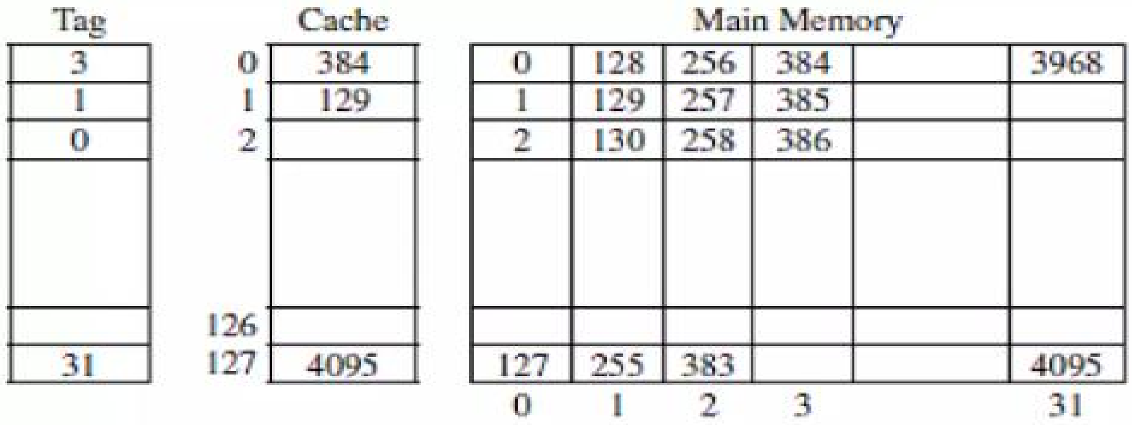
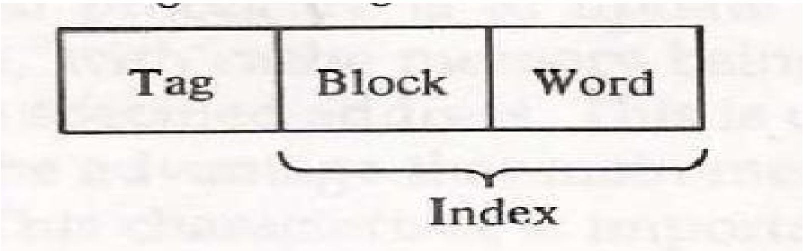


Figure 6.4 Mapping main memory blocks to cache blocks



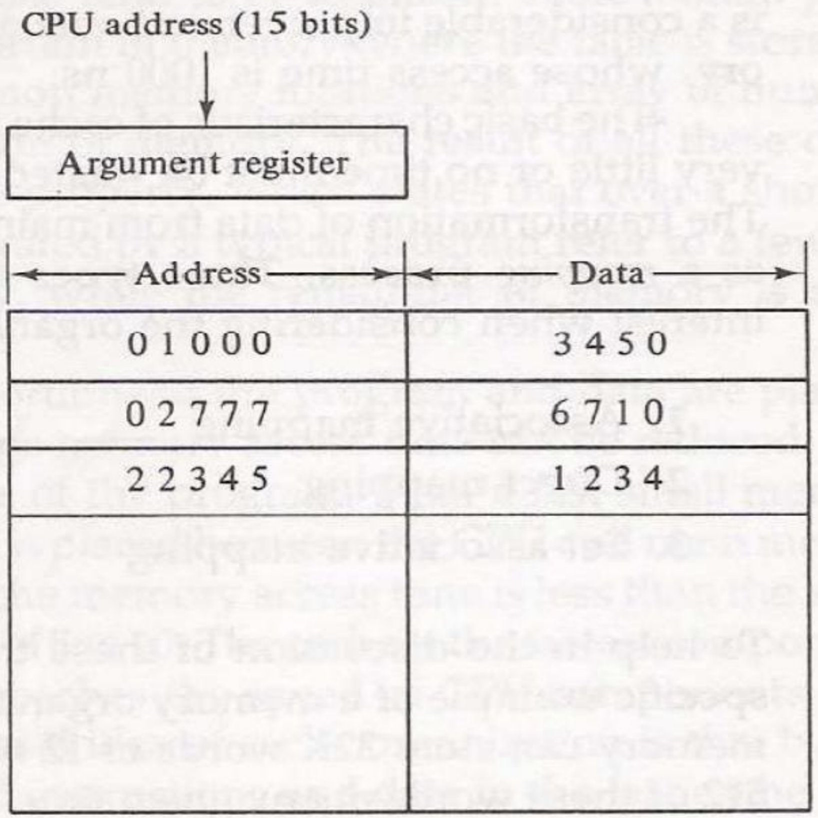
ADVANTAGE & DISADVANTAGE

- The main advantage of the direct-mapping technique is its simplicity in determining where to place an incoming main memory block in the cache.
- Its main disadvantage is the inefficient use of the cache. This is because a number of main memory blocks may compete for a given cache block even if there exist other empty cache blocks. This disadvantage should lead to achieving a low cache hit ratio.

2. Fully- Associative mapping:

- In this mapping function, any block of Main memory can potentially reside in any cache block position. This is much more flexible mapping method.

Figure 12-11 Associative mapping cache (all numbers in octal).



- The associative memory stores both address and content(data) of the memory word.
- This permits any location in cache to store any word from main memory.
- The diagram shows three words presently stored in the cache. The address value of 15 bits is shown as a five-digit ctal number and its corressponding 12-bit word is shown as a four-digit octal number.
- A CPU address of 15-bits is placed in the argument register and the associative memory is searched for a matching address.
- If address is found, the corresponding 12-bit data is read and sent to the CPU. If no match occurs, the main memory is accessed for the word

3. Set-associative mapping:

- In this method, blocks of cache are grouped into sets, and the mapping allows a block of main memory to reside in any block of a specific set.
- From the flexibility point of view, it is in between to the other two methods

Index	Tag	Data	Tag	Data
000	0 1	3 4 5 0	0 2	5 6 7 0
777	0 2	6 7 1 0	0 0	2 3 4 0

Figure 12-15 Two-way set-associative mapping cache.

- The octal numbers listed in Fig.12-15 are with reference to the main memory contents. When the CPU generats a memory request, the index valus of the address is used to access the cache.
- The tag field of the CPU address is then compared with both tags in the cache to determine if a match occurs.
- The comparison logic dine by an associative search of the tags in the set similar to anassociative memory search thus the name “Set Associative”.

Cache Performance:

- The performance of cache memory is frequently measured in terms of a quantity called hit ratio

Hit ratio = hit / (hit+miss)

- Performance Parameter of Cache Memory

1. Hit: When the CPU refers to memory and find the word in cache , it is said to produce a HIT.

2. Miss: The requested data is not found in the cache memory, then it counts as a MISS.

- Hit Rate: Performance of cache memory is measured in terms of quality, is called Hit Rate(Hit Ratio)

Hit Rate = No. of Hits / Total CPU references to Cache memory

Hit Rate= No. of Hits / (Hits + Misses)

- Miss Rate: The percentage of memory accesses not found in a given level of memory.

$$\text{Miss Rate} = 1 - \text{Hit Rate}$$

- Hit Time: The time required to access the requested information in a given level of memory.
- Miss Penalty: The time required to process a miss,which include replacing a block in memory plus the time required to deliver requested data to the processor.

Associative Memory

- The time required to find an item stored in memory can be reduced considerably if stored data can be identified for access by the content of the data itself rather than by an address.
- A memory unit accessed by content is called an **associative memory or content addressable memory (CAM).**

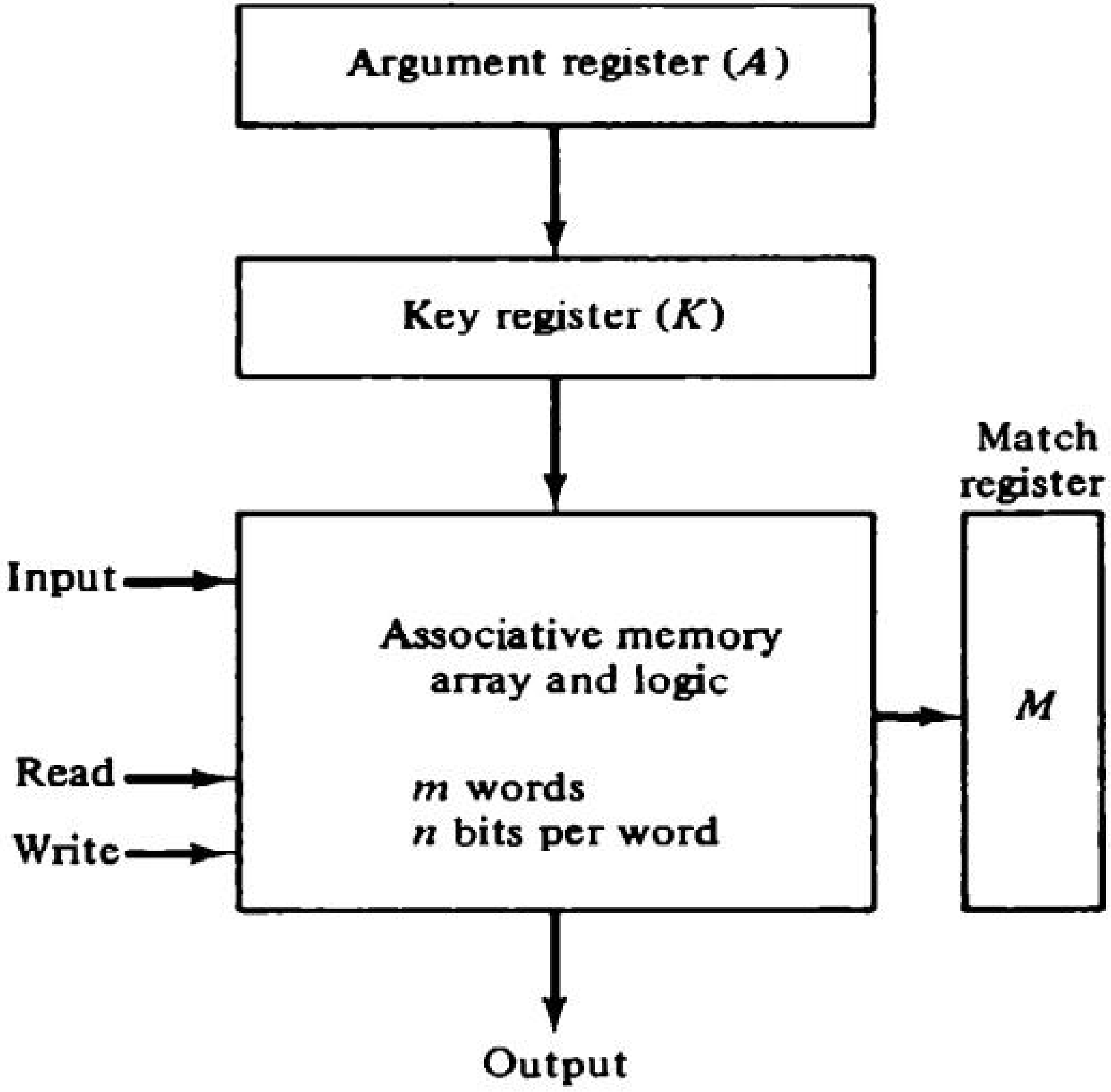
- CAM is accessed simultaneously and in parallel on the basis of data content rather than by specific address or location

- Associative memory is more expensive than a RAM because each cell must have storage capability as well as logic circuits

❖ Argument register –holds an external argument for content matching

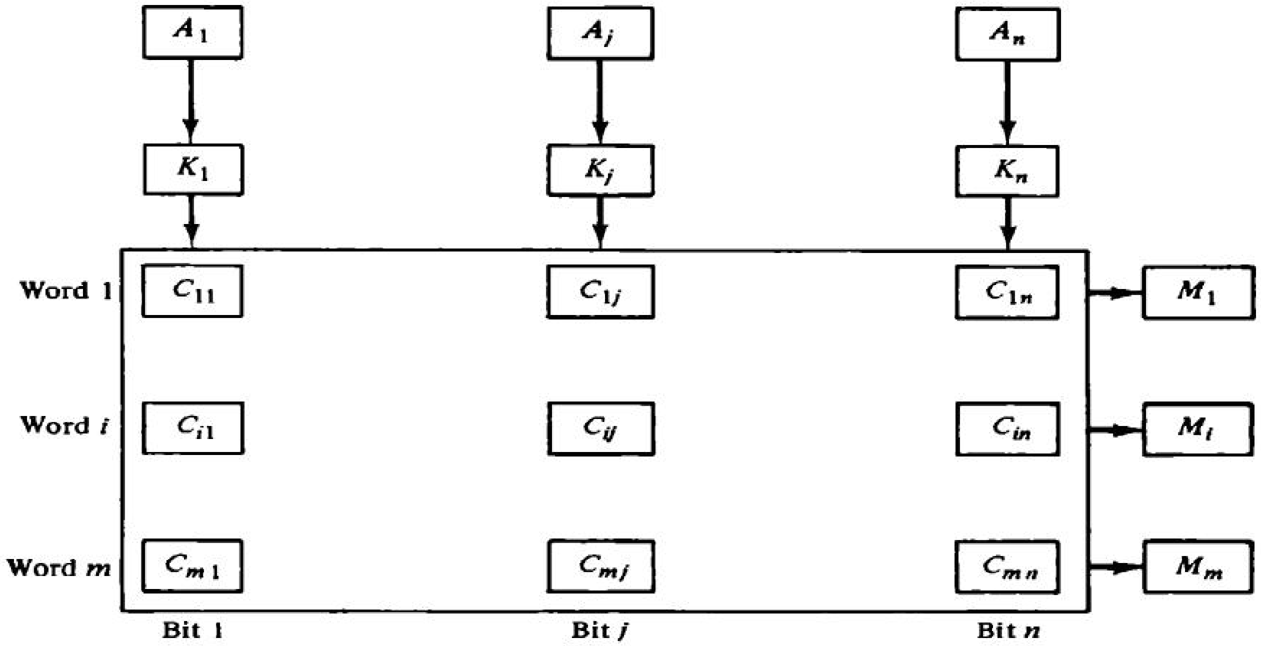
❖ Key register –mask for choosing a particular field or key in the argument word

Figure 12-6 Block diagram of associative memory.



- It consists of a memory array and logic for m words with n bits per word. The argument register A and key register K each have n bits, one for each bit of a word.
- The match register M has m bits, one for each memory word. Each word in memory is compared in parallel with the content of the argument register.
- The words that match the bits of the argument register set a corresponding bit in the match register. After the matching process, those bits in the match register that have been set indicate the fact that their corresponding words have been matched.
- Reading is accomplished by a sequential access to memory for those words whose corresponding bits in the match register have been set

Figure 12-7 Associative memory of *m* word, *n* cells per word.



Virtual Memory

- Early days memory was expensive – hence small .
- Programmers were using secondary storage for overlaying
- Programmers were responsible for breaking programs in to overlays , decide where to keep in secondary memory, arranging for transfer of overlays between main memory and secondary memory
- In 1961 Manchester University proposed a method for performing overlay process automatically which has given rise to the concept of Virtual memory today

Virtual Memory - Background

- Separate concept of address space and memory locations
- Programs reference instructions and data that is independent of available physical memory Addresses issued by processor for Instructions or Data are called Virtual or Logical addresses
- Virtual addresses are translated in to physical addresses by a combination of Hardware and Software components

- Memory on disk
- Allows for effective multiprogramming and relieves the user of tight constraints of main memory

Address Space and Memory Space

- Address used by a programmer is called virtual address and set of such addresses is called address space
- Address in main memory is called a location or physical address and set of such locations is called the memory space
- The Address Space is allowed to be larger than the memory space in computers with virtual memory

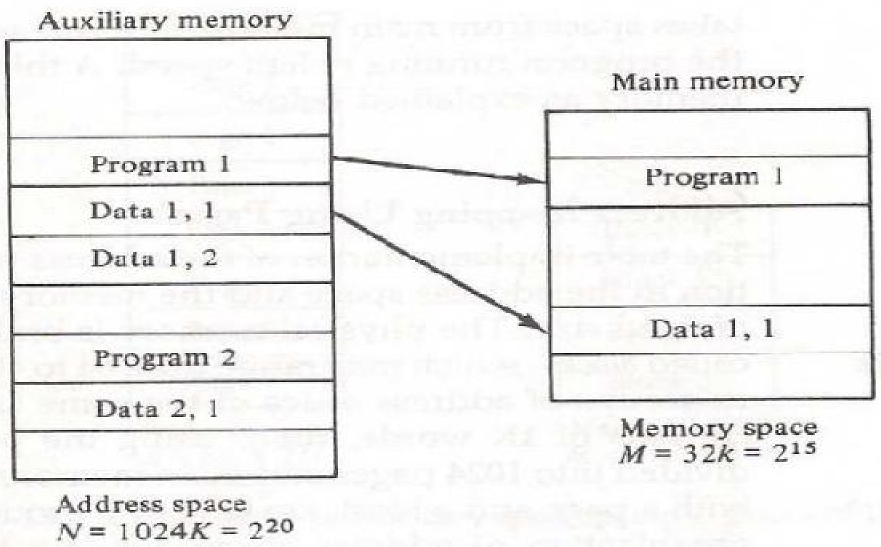


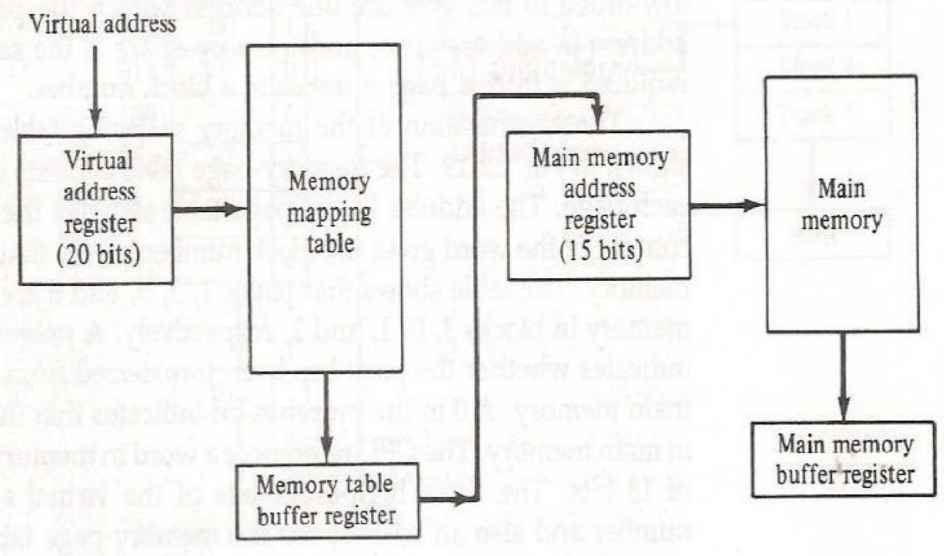
Figure 12-16 Relation between address and memory space in a virtual memory system.

- In a multiprogram computer system, programs and data are transferred to and from auxiliary memory and main memory based on demands imposed by the CPU.
- Suppose that program1 is currently being executed in the CPU.

  - Program1 and a portion of its associated data are moved from auxiliary memory into main memory as shown in fig. 12-16.

  - Portions of programs and data need not be in contiguous locations in memory since information is being moved in out, and empty spaces may be available in scattered locations in memory

Figure 12-17 Memory table for mapping a virtual address.



In fig 12-17, To map a virtual address of 20 bits to a physical address of 15 bits. The mapping is a dynamic operation, which means that every address is translated immediately as a word is referenced by CPU. The mapping table may be stored in a separate memory. In first case, an additional unit is required as well as one extra memory access time. In the second case, the table takes space from main memoy and two accesses to memory are required with program running at half speed. A third alternative is to use an associative memory

Address Mapping Using Pages

The physical memory is broken down into groups of equal size called blocks, which may range from 64 to 4096 word each. The term page refers to groups of address space of the same size. Portions of programs are moved from auxiliary memory to main memory in records equal to the size of a page. The term “page frame” is sometimes used to denote a block.

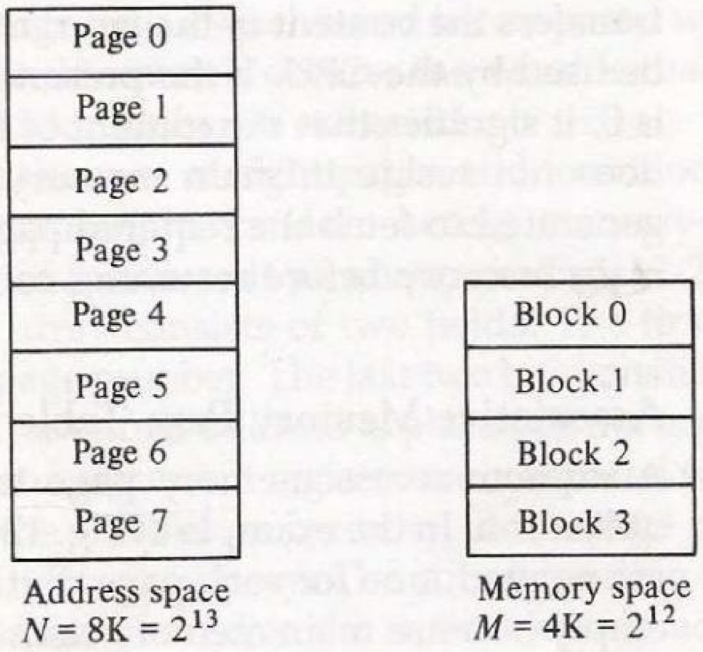
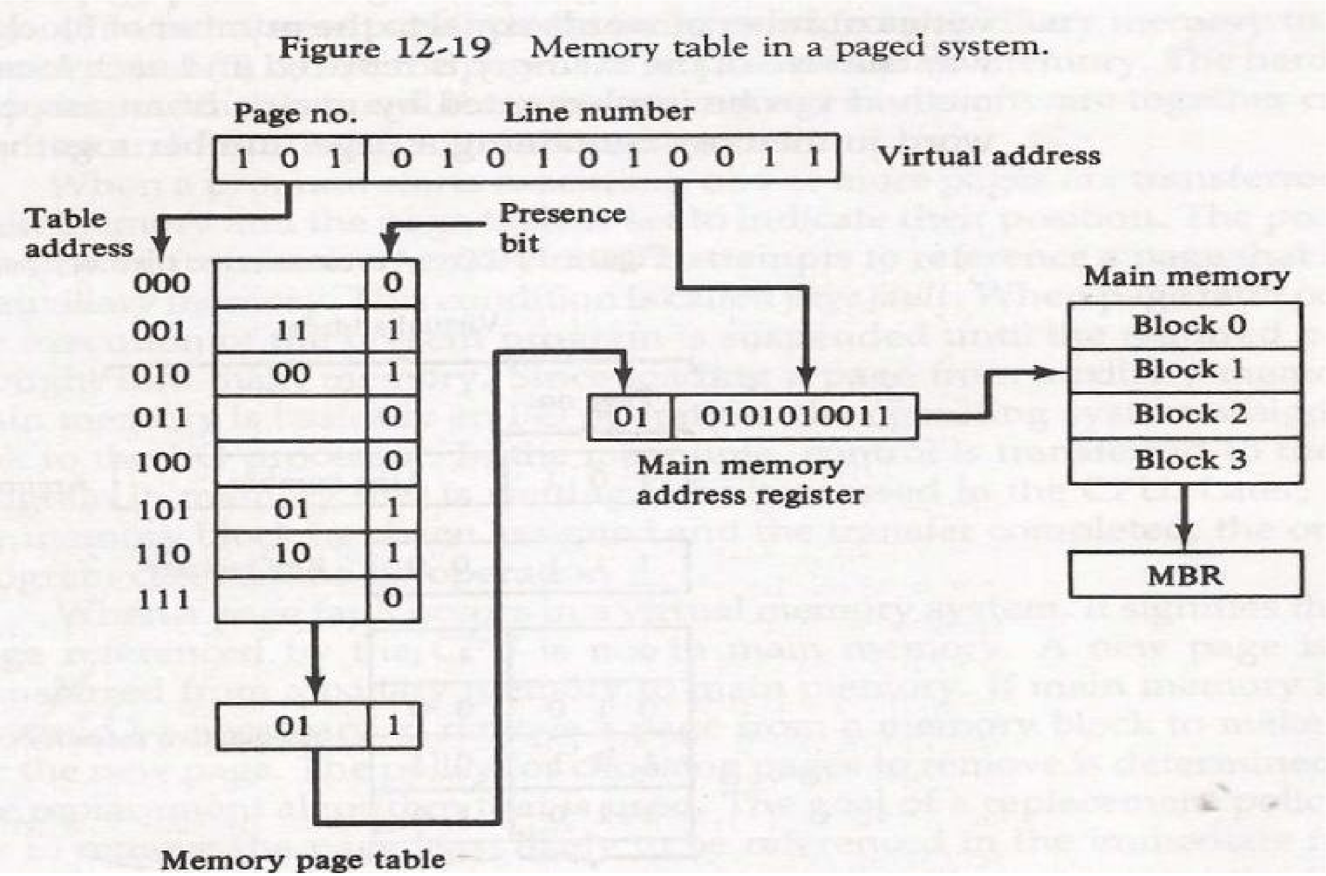


Figure 12-18 Address space and memory space split into groups of 1K words.

In fig 12-18, a virtual address has 13 bits. Since each page consists of 1024 words, the high order three bits of virtual address will specify one of the eight pages and the low order 10 bits give the line address within the page.



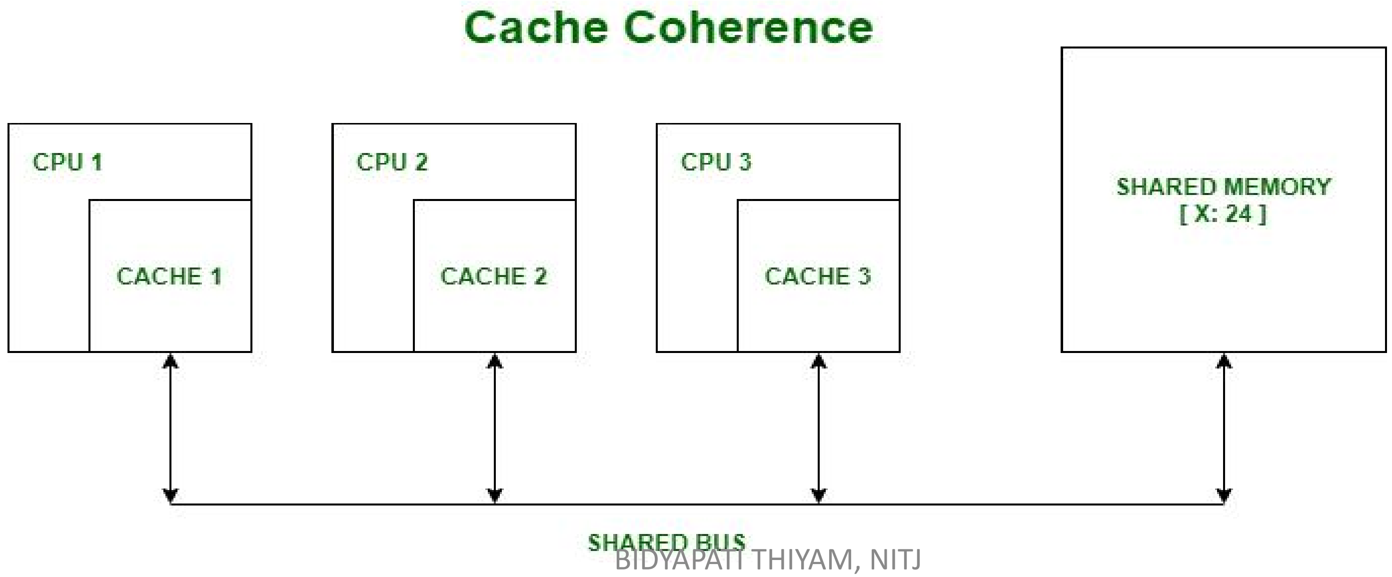
The organization of the memory mapping table in a paged system is shown in Fig.12-19. The memory page table consists of eight word , one for each page. The address in the page tabledenotes the page number and the content of the word gives the block number where that page is stored in main memory. The table showsthat pages 1,2,5 and 6 are now available in main memory in blocks 3,0,1 and 2, respectively.

Comparison between Mapping Techniques

Sr. No.	Direct-mapping	Associative-mapping	Set-associative-mapping
1.	Each block from the main memory has only one possible location in the cache.	A block of data from main memory can be placed into any cache block position.	A block of data from main memory can go into a particular block location of any direct-mapped cache.
2.	Needs only one comparison.	Needs comparison with all tag bits.	Needs number of comparisons equal to number of blocks per set.
3.	Cache hit ratio decreases if processor needs to access same memory location from two different pages of the main memory frequently.	Cache hit ratio has no effect if processor needs to access same memory location from two different pages of the main memory frequently.	The effect of reduction in cache hit ratio in case of frequent access to the two different pages of the main memory is reduced.
4.	Main memory address is divided into three fields : TAG, BLOCK and WORD.	Main memory address is divided into two fields : TAG and WORD.	Main memory address is divided into three fields : TAG, SET and WORD.
5.	Searching time is less.	Searching time is more.	Searching time increases with number of blocks per set.

Cache Coherence

- In a multiprocessor system, data inconsistency may occur among adjacent levels or within the same level of the memory hierarchy.
- In a shared memory multiprocessor with a separate cache memory for each processor, it is possible to have many copies of any one instruction operand: one copy in the main memory and one in each cache memory.
- When one copy of an operand is changed, the other copies of the operand must be changed also



- Suppose there are three processors, each having cache. Suppose the following scenario:-
  - Processor 1 read X : obtains 24 from the memory and caches it.
  - Processor 2 read X : obtains 24 from memory and caches it.
  - Again, processor 1 writes as X : 64, Its locally cached copy is updated. Now, processor 3 reads X, what value should it get?
  - Memory and processor 2 thinks it is 24 and processor 1 thinks it is 64.

- As multiple processors operate in parallel, and independently multiple caches may possess different copies of the same memory block, this creates a cache coherence problem.
- Cache coherence is the discipline that ensures that changes in the values of shared operands are propagated throughout the system in a timely fashion.
- There are three distinct level of cache coherence :-
  - Every write operation appears to occur instantaneously.
  - All processors see exactly the same sequence of changes of values for each separate operand.
  - Different processors may see an operation and assume different sequences of values; this is known as non-coherent behavior.

Coherency mechanisms :
There are three types of coherence :

1**. Directory-based**

– In a directory-based system, the data being shared is placed in a common directory that maintains the coherence between caches.

-The directory acts as a filter through which the processor must ask permission to load an entry from the primary memory to its cache.

-When an entry is changed, the directory either updates or invalidates the other caches with that entry.

2. Snooping

– First introduced in 1983, snooping is a process where the individual caches monitor address lines for accesses to memory locations that they have cached. It is called a write invalidate protocol.

-When a write operation is observed to a location that a cache has a copy of and the cache controller invalidates its own copy of the snooped memory location.

3. Snarfing

– It is a mechanism where a cache controller watches both address and data in an attempt to update its own copy of a memory location when a second master modifies a location in main memory.

When a write operation is observed to a location that a cache has a copy of the cache controller updates its own copy of the snarfed memory location with the new data.

Memory Interleaving

- Memory interleaving is based on the principle of parallelism, where multiple memory operations can be executed simultaneously.
- It involves dividing the memory into multiple banks or modules and organizing them in such a way that consecutive memory locations are stored in different banks.
- This allows the system to access multiple memory locations in parallel, reducing the time required to fetch data.
- In a system without interleaving, memory is typically organized as a single block, and access to consecutive memory locations requires sequential access.
- This sequential access can lead to increased latency, especially in systems where memory access times are relatively slow compared to processor speeds.

- There are 2-way and 4-way interleaving for specific implementations of memory interleaving that define how memory addresses are distributed across multiple memory modules (or banks).
- The "way" indicates the number of memory banks involved in the interleaving.

1. 2-way Interleaving:

- The memory is divided into 2 banks, and addresses are distributed alternately between the two banks.

Example:

- Memory Banks: Bank 0 and Bank 1.

Address distribution:

- Bank 0: Addresses 0, 2, 4, 6, ...
- Bank 1: Addresses 1, 3, 5, 7, ...

Advantages:

- Reduces memory access conflicts.
- Improves performance in systems with dual processors or pipelined memory access.

Disadvantages:

- Limited concurrency compared to higher-way interleaving.

2. 4-Way Interleaving

- The memory is divided into 4 banks, and addresses are distributed cyclically across these banks.

Example:

Memory Banks: Bank 0, Bank 1, Bank 2, Bank 3.

Address distribution:

Bank 0: Addresses 0, 4, 8, 12, ...

Bank 1: Addresses 1, 5, 9, 13, ...

Bank 2: Addresses 2, 6, 10, 14, ...

Bank 3: Addresses 3, 7, 11, 15, ...

Advantages:

- Greatly increases parallelism.
- Suitable for systems with higher memory access demands, such as multi-core processors.

Disadvantages:

- Requires more complex control logic for address mapping.

Benefits of Memory Interleaving

The benefits of Memory Interleaving are given below:

1. Improved Memory Access Time:

- Memory interleaving improves memory access time by allowing the system to access multiple memory locations simultaneously. This reduces the time required to fetch data, especially for applications that access memory in a sequential or predictable pattern.

2. Increased Bandwidth:

- By accessing multiple memory modules in parallel, memory interleaving increases the overall memory bandwidth of the system. This is particularly beneficial for applications that require high-speed data transfer, such as multimedia processing or scientific computing.

3. Reduced Memory Contention:

- Memory interleaving reduces memory contention by spreading memory access requests across multiple memory modules. This helps avoid bottlenecks and improves overall system performance, especially in multi-threaded or multi-core systems.

4. Enhanced System Performance:

- Overall, memory interleaving enhances system performance by reducing memory access latency, increasing memory bandwidth, and improving system responsiveness. This can lead to a better user experience and improved efficiency in various computing tasks.

RAID (Redundant Arrays of Independent Disks)

- RAID (Redundant Arrays of Independent Disks) is a technique that makes use of a combination of multiple disks for storing the data instead of using a single disk for increased performance, data redundancy, or to protect data in the case of a drive failure.
- RAID (Redundant Array of Independent Disks) is like having backup copies of your important files stored in different places on several hard drives or solid-state drives (SSDs).
- If one drive stops working, your data is still safe because you have other copies stored on the other drives.
- It's like having a safety net to protect your files from being lost if one of your drives breaks down.

RAID Techniques:

- striping, mirroring, and parity, the three fundamental techniques used in RAID systems:

1. Striping

Striping involves dividing data into smaller chunks (stripes) and writing them across multiple drives in sequence.

2. Mirroring

Mirroring involves creating identical copies of data on two or more drives.

3. Parity

Parity is a mathematical method used to provide fault tolerance by storing a checksum that allows for the reconstruction of lost data.

Types of RAID:

1. RAID-0 (Stripping)

- Blocks are "stripped" across disks.
- In the figure, blocks "0,1,2,3" form a stripe.
- Instead of placing just one block into a disk at a time, we can work with two (or more) blocks placed into a disk before moving on to the next one.

RAID 0

Disk 0	Disk 1	Disk 2	Disk 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Disk 0	Disk 1	Disk 2	Disk 3
0	3	4	6
1	3	5	7
8	10	12	14
9	11	13	15

Advantages

- It is easy to implement.
- It utilizes the storage capacity in a better way.

Disadvantages

- A single drive loss can result in the complete failure of the system.

2. RAID-1 (Mirroring)

- More than one copy of each block is stored in a separate disk. Thus, every block has two (or more) copies, lying on different disks.

RAID 1

Disk 0	Disk 1	Disk 2	Disk 3
0	0	1	1
2	2	3	3
4	4	5	5
6	6	7	7

Advantages

- It covers complete redundancy.
- It can increase data security and speed.

Disadvantages

- It is highly expensive.
- Storage capacity is less.

- The above figure shows a RAID-1 system with mirroring level 2.
- RAID 0 was unable to tolerate any disk failure. But RAID 1 is capable of reliability.

3. RAID 5 (Striping with Parity)

- Stripes data across three or more drives and includes parity (a checksum) for fault tolerance.

Advantages:

- Balanced performance and redundancy.
- Can tolerate one drive failure.

Disadvantages:

- Slower write speeds due to parity calculations.
- Requires at least 3 drives.