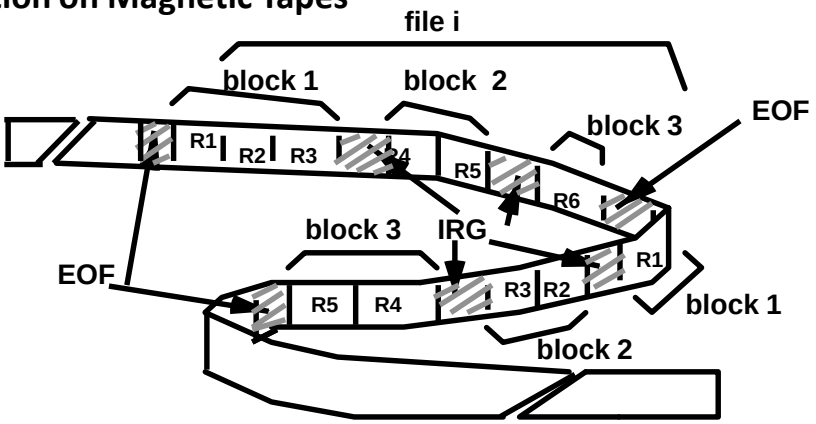
# Overview

- Memory Hierarchy
- Main Memory
- **Auxiliary Memory**
- **Associative Memory**
- Cache Memory
- Virtual Memory

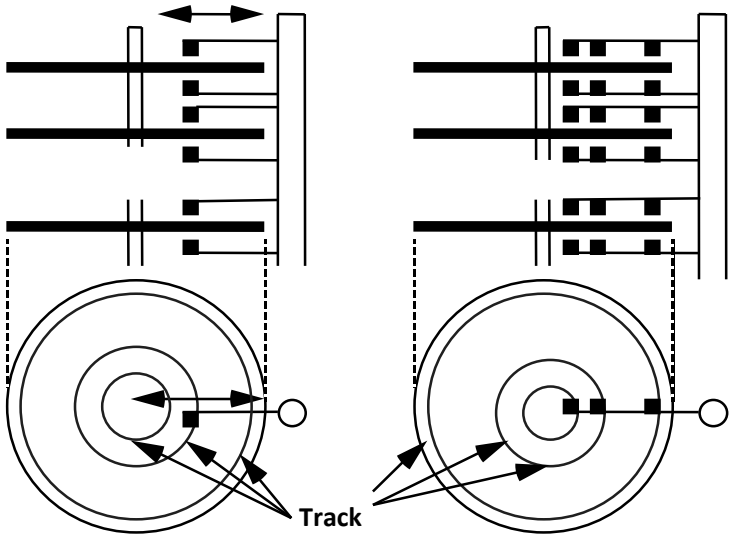
# Auxiliary Memory

**Information Organization on Magnetic Tapes**




**Organization of Disk Hardware**

**Moving Head Disk** **Fixed Head Disk**



# Associative Memory

- ✓ Many Data processing requires Search Operation
- ✓ The search procedure is strategy for choosing a sequence of address ,reading the content of the memory at each address and comparing the information read with item being searched until a Match occurs
- ✓ The number of accesses to memory depends on the location of the item and the efficiency of the search algorithm .
- ✓ Time can reduced if instead of address ,content can be used .
- ✓ When a word is written in associative memory no address is given
- ✓ When a word is to be read from an associative memory ,the content of the word or part is specified .

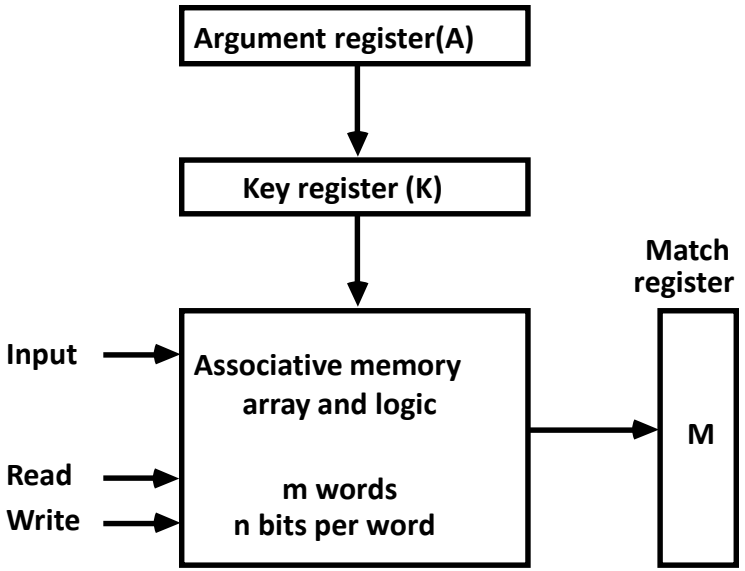
# Associative Memory

- ✓ Uniquely suited for parallel search by data association .
- ✓ AM is expensive than a random access memory because each cell must have storage capability as well as logic circuit for matching its content with and external argument .
- ✓ AM is used where search time is very critical and must be very short
- ✓

# Associative Memory

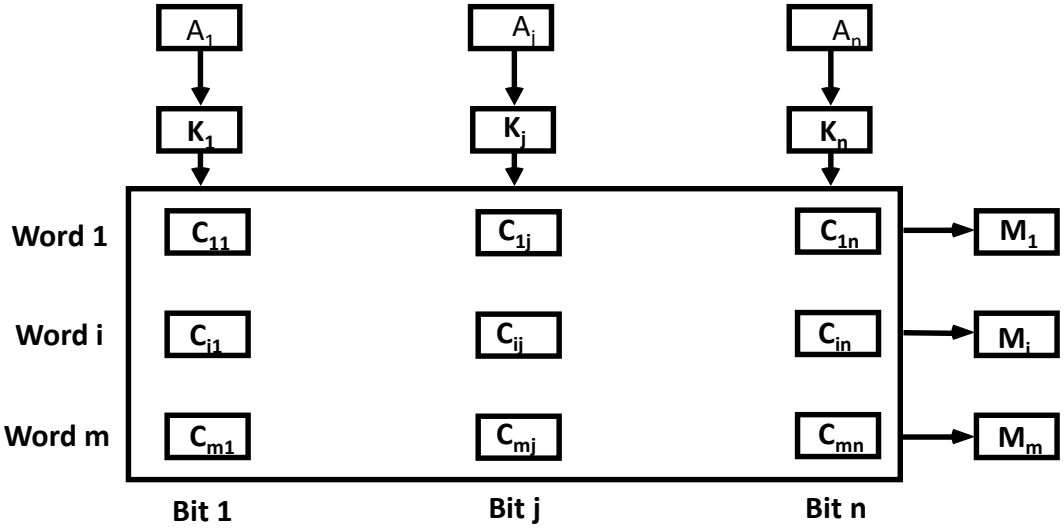
- Accessed by the content of the data rather than by an address
- Also called Content Addressable Memory (CAM)

**Hardware Organization**

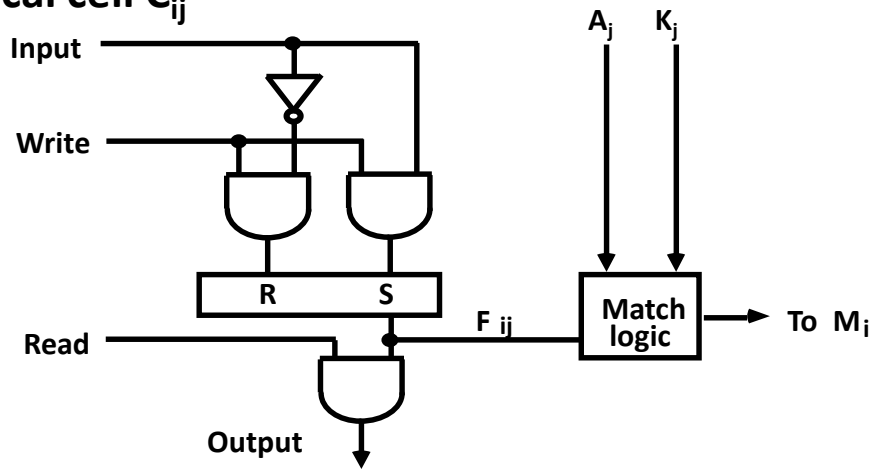



- Compare each word in CAM in parallel with the content of A(Argument Register)
- If CAM Word[i] = A, M(i) = 1
- Read sequentially accessing CAM for CAM Word(i) for M(i) = 1
- K(Key Register) provides a mask for choosing a particular field or key in the argument in A (only those bits in the argument that have 1's in their corresponding position of K are compared)

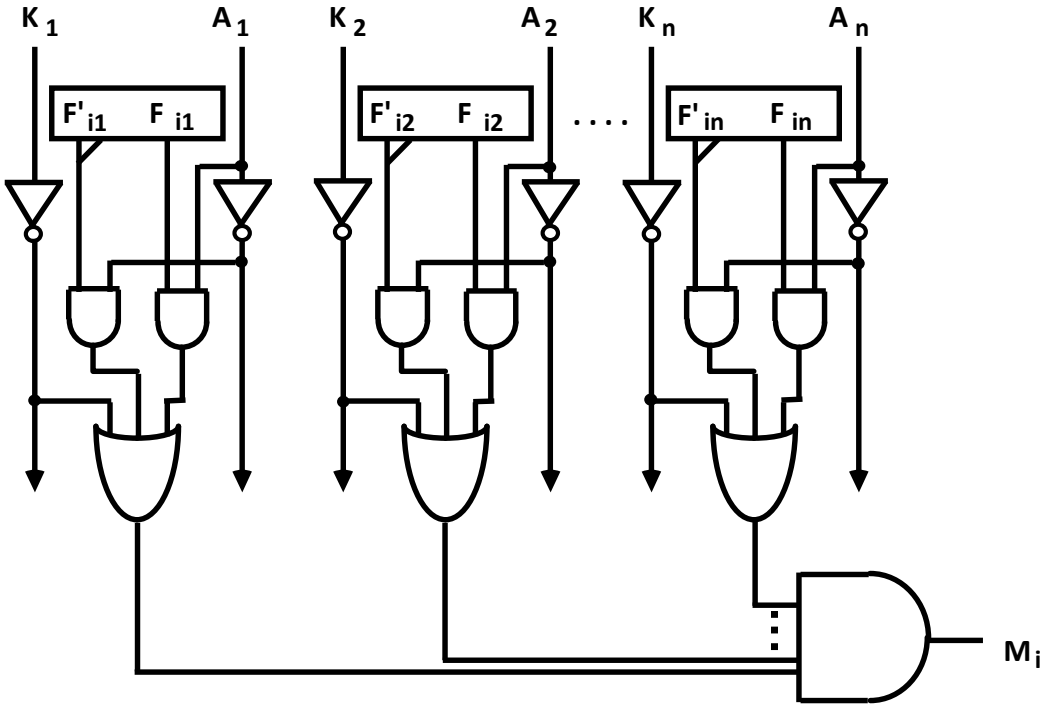
# Organization of CAM




**Internal organization of a typical cell $C_{ij}$**



# Match Logic



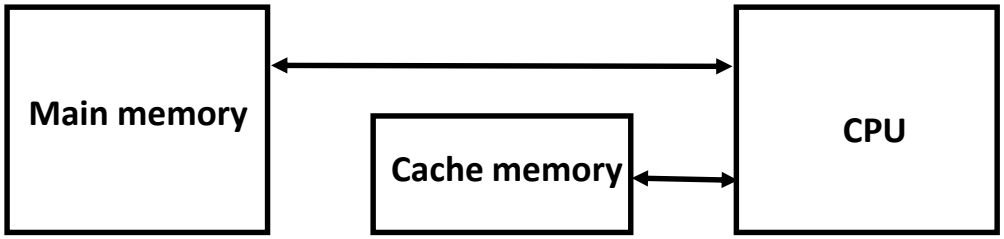
# Cache Memory

Locality of Reference

- The references to memory at any given time interval tend to be confined within a localized areas
- This area contains a set of information and the membership changes gradually as time goes by
- *Temporal Locality*
  The information which will be used in near future is likely to be in use already( e.g. Reuse of information in loops)
- *Spatial Locality*
  If a word is accessed, adjacent(near) words are likely accessed soon (e.g. Related data items (arrays) are usually stored together; instructions are executed sequentially)

Cache

- The property of Locality of Reference makes the cache memory systems work
- Cache is a fast small capacity memory that should hold those information which are most likely to be accessed



# Performance of Cache

**Memory Access**

**All the memory accesses are directed first to Cache**
**If the word is in Cache; Access cache to provide it to CPU**
**If the word is not in Cache; Bring a block (or a line) including that word to replace a block now in Cache**

**- How can we know if the word that is required is there ?**
**- If a new block is to replace one of the old blocks, which one should we choose ?**

**Performance of Cache Memory System**

**Hit Ratio - % of memory accesses satisfied by Cache memory system**
**Te: Effective memory access time in Cache memory system**
**Tc: Cache access time**
**Tm: Main memory access time**

$$Te = Tc + (1 - h)\, Tm$$

**Example: Tc = 0.4 μs, Tm = 1.2μs, h = 0.85%**

$$Te = 0.4 + (1 - 0.85) * 1.2 = 0.58\mu s$$

# Memory and Cache Mapping – (Associative Mapping)

**Mapping Function**

**Specification of correspondence between main memory blocks and cache blocks**

| |
|---|
| **Associative mapping** |
| **Direct mapping** |
| **Set-associative mapping** |

**Associative Mapping**

- **Any block location in Cache can store any block in memory**
  **-> Most flexible**
- **Mapping Table is implemented in an associative memory**
  **-> Fast, very Expensive**
- **Mapping Table**
  **Stores both address and the content of the memory word**