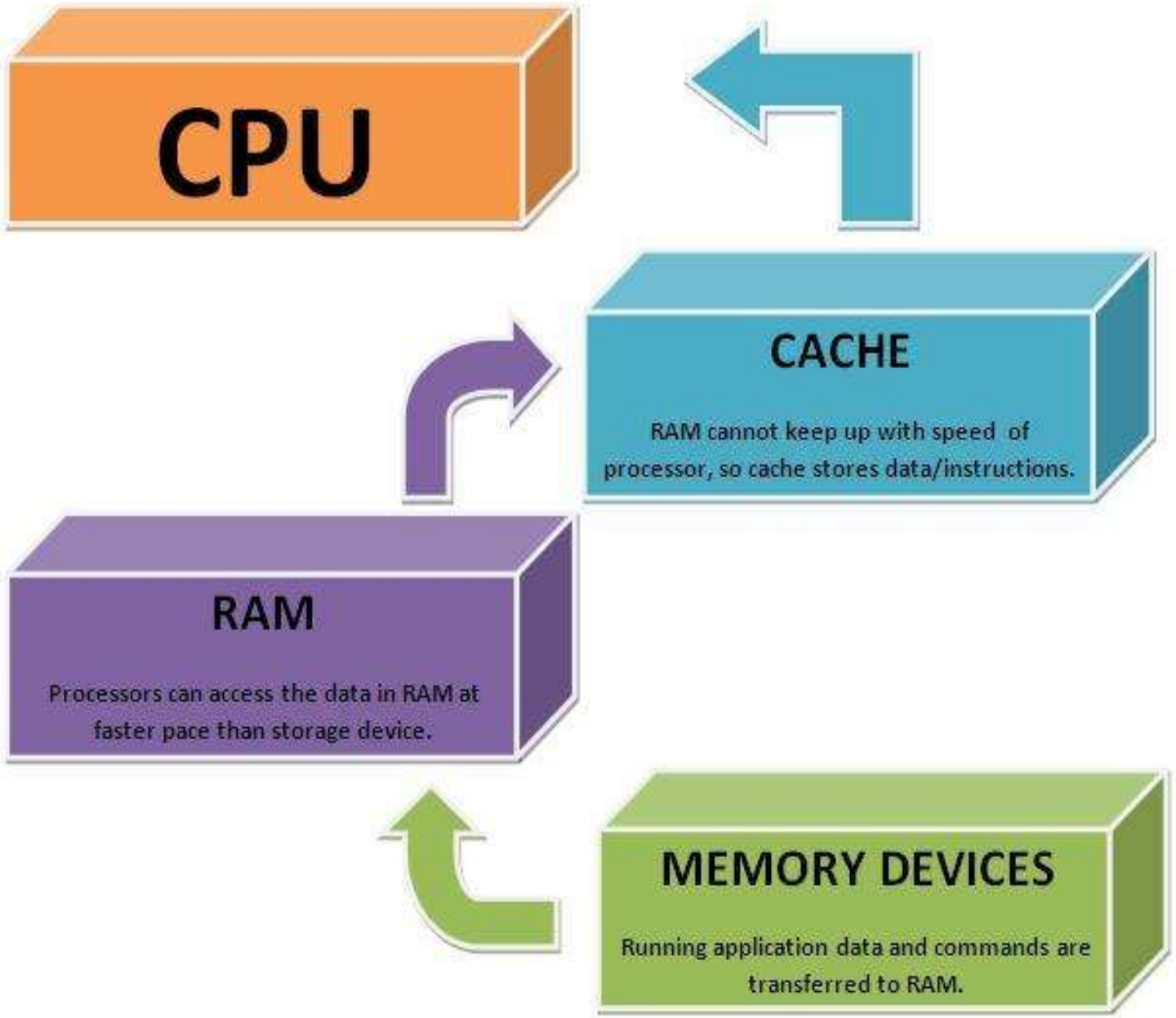
STUDY OF FUNCTIONING OF CACHE MEMORY AND ITS LATEST DEVELOPMENTS IN CACHE MEMORY

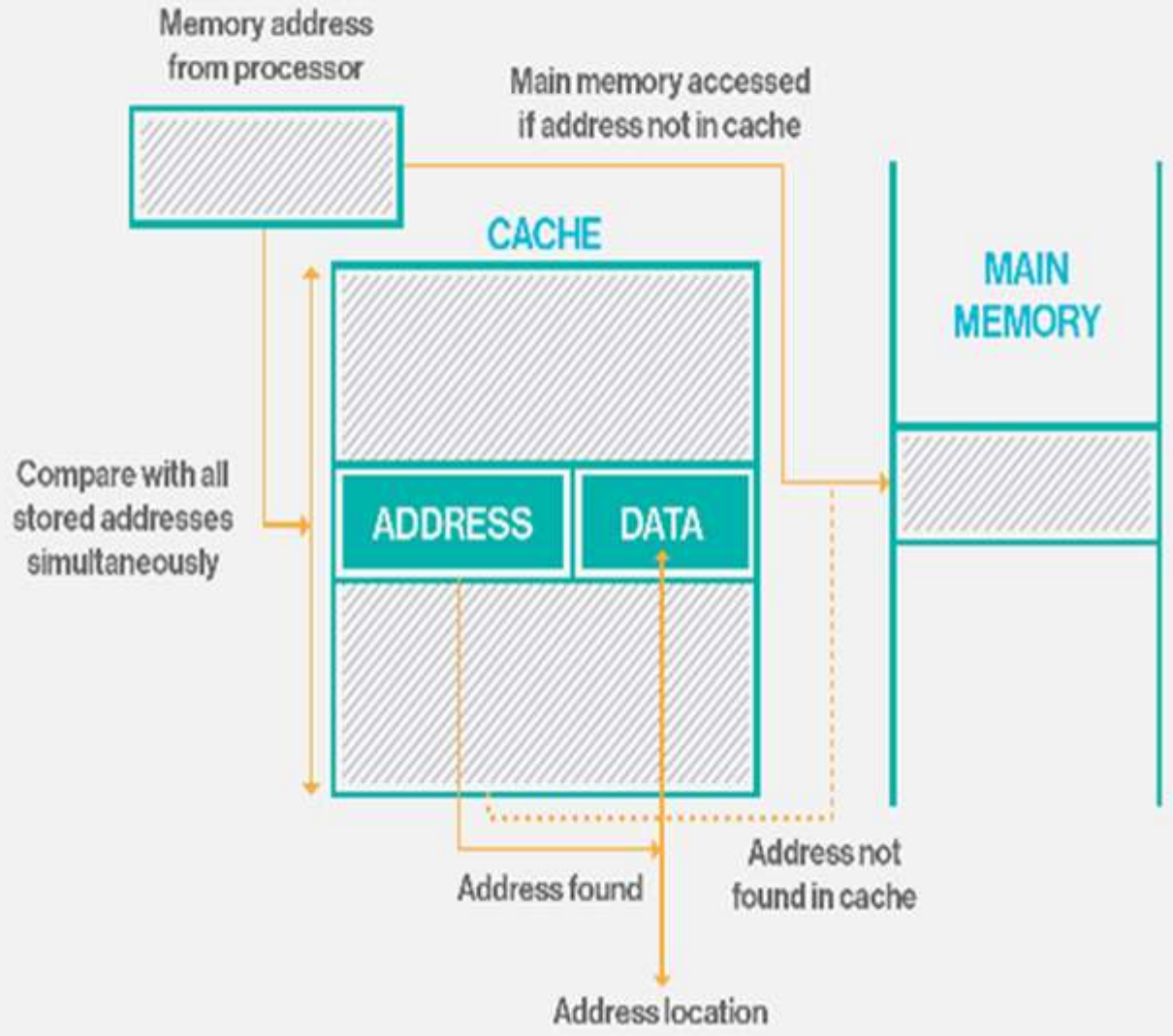
WHAT IS CACHE ?

- A **cache memory** is a fast and relatively small memory, that stores the most recently used (MRU) main memory(MM) (or working memory) data.
- It is simply a copy of a small data segment residing in the main memory.
- Hold identical copies of main memory.
- The function of this is to speed up the MM data access.

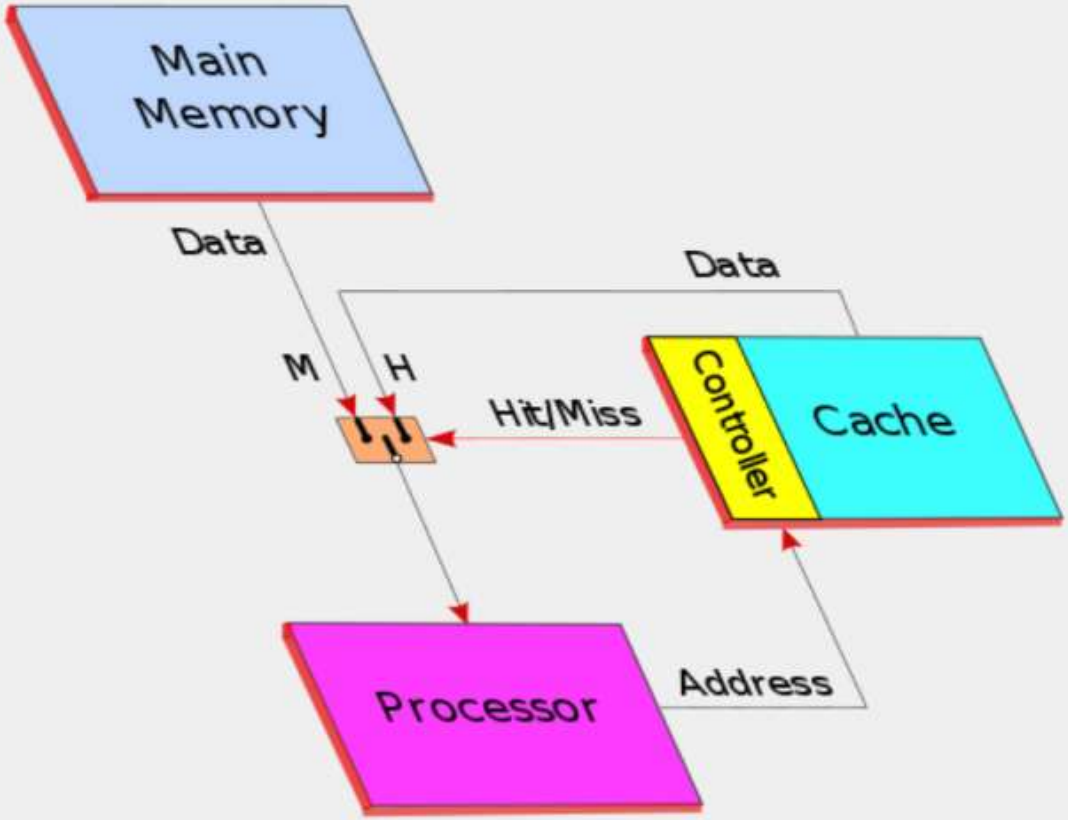

CPU
CACHE
RAM cannot keep up with speed of processor, so cache stores data/instructions.
RAM
Processors can access the data in RAM at faster pace than storage device.
MEMORY DEVICES
Running application data and commands are transferred to RAM.

- Memories that consists of circuits capable of retaining their state as long as power is applied are known as static memory.
- Less expensive RAM‟s can be implemented if simplex calls are used such cells cannot retain their state indefinitely. Hence they are called Dynamic RAM's (DRAM). The information stored in a dynamic memory cell in the form of a charge on a capacitor and this charge can be maintained only for tens of Milliseconds.
- Latency is one of the parameter indicating good performance
- Latency: It refers to the amount of time it takes to transfer a word of data to or from the memory.

Cache Memory



Cache Operation Diagram



By Ferruccio Zulian - Milan. Italy

Functional principles of the cache memory

- In MM read operation, the cache controller first of all checks if the data is stored in cache.
- In case of match (Hit - cache hit) the data is fastly and directly supplied from the cache to the processor without involving the MM.
- Else (Miss - cache miss) the data is read from MM.
- A cache hit is a state in which data requested for processing by a component or application is found in the cache memory. It is a faster means of delivering data to the processor, as the cache already contains the requested data.

- A cache hit occurs when an application or software requests data. First, the CPU looks for the data in its closest memory location, which is usually the primary cache. If the requested data is found in the cache, it is considered a cache hit.
- A cache hit serves data more quickly, as the data can be retrieved by reading the cache memory.
- Cache miss is a state where the data requested for processing by a component or application is not found in the cache memory. It causes execution delays by requiring the program or application to fetch the data from other cache levels or the main memory.

- Each cache miss slows down the overall process because after a cache miss, the CPU will look for a higher level cache, such as L1, L2, L3 and RAM for that data. Further, a new entry is created and copied in cache before it can be accessed by the processor.
- The more cache hits the better.

Cache Memory operation is based on two major "principles of locality"

- **Temporal locality**
- **Spatial locality'**

Temporal locality

- Data which have been used recently have high likelihood of being used again.
- A cache stores only a subset of MM data – the most recent-used MRU. Data read from MM are temporary stored in cache. If the processor requires the same data, this is supplied by the cache.

Spatial locality

- If a data is referenced, it is very likely that nearby data will be accessed soon.
- Instructions and data are transferred from MM to the cache in fixed blocks (cache block), known as cache lines. Cache line size is in the range of 4 to 512 bytes.
- Most programs are highly sequential. Next instruction usually comes from the next memory location. Data is usually structured and data in these structures normally are stored in contiguous memory locations (data strings, arrays, etc.).
- Large lines size increase the spatial locality but increase also the number of invalidated data in case of line replacement .

MAPPING FUNCTIONS

1. **Direct mapping**

- The simplest way to determine cache locations in which to store memoty blocks is the direct mapping technique.

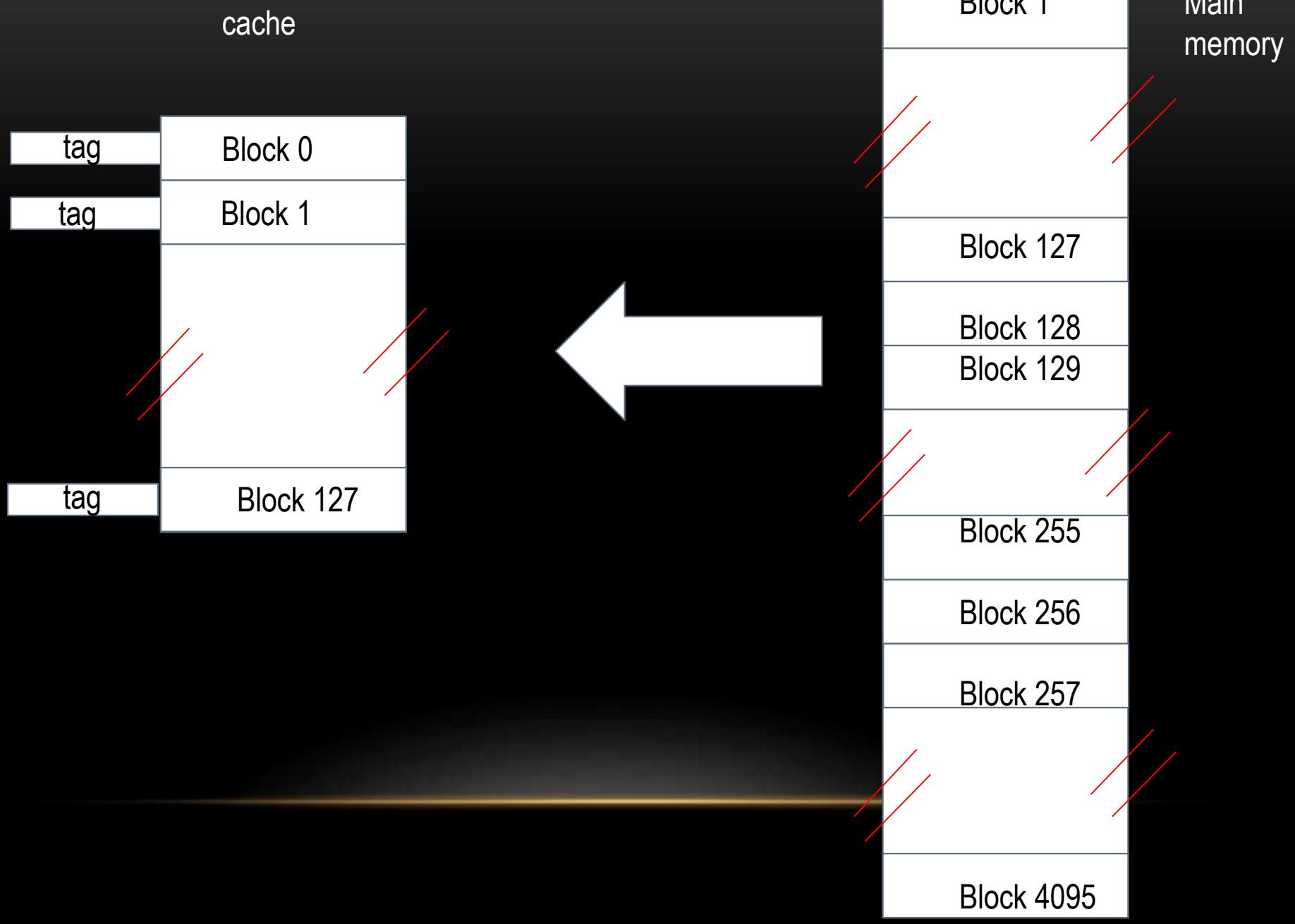
2. Associative mapping

- In this method, the main memory block can be placed into any cache block position.
- 12 tag bits will identify a memory block when it is resolved in the cache.
- The tag bits of an address received from the processor are compared to the tag bits of each block of the cache to see if the desired block is persent.This is called associative mapping.

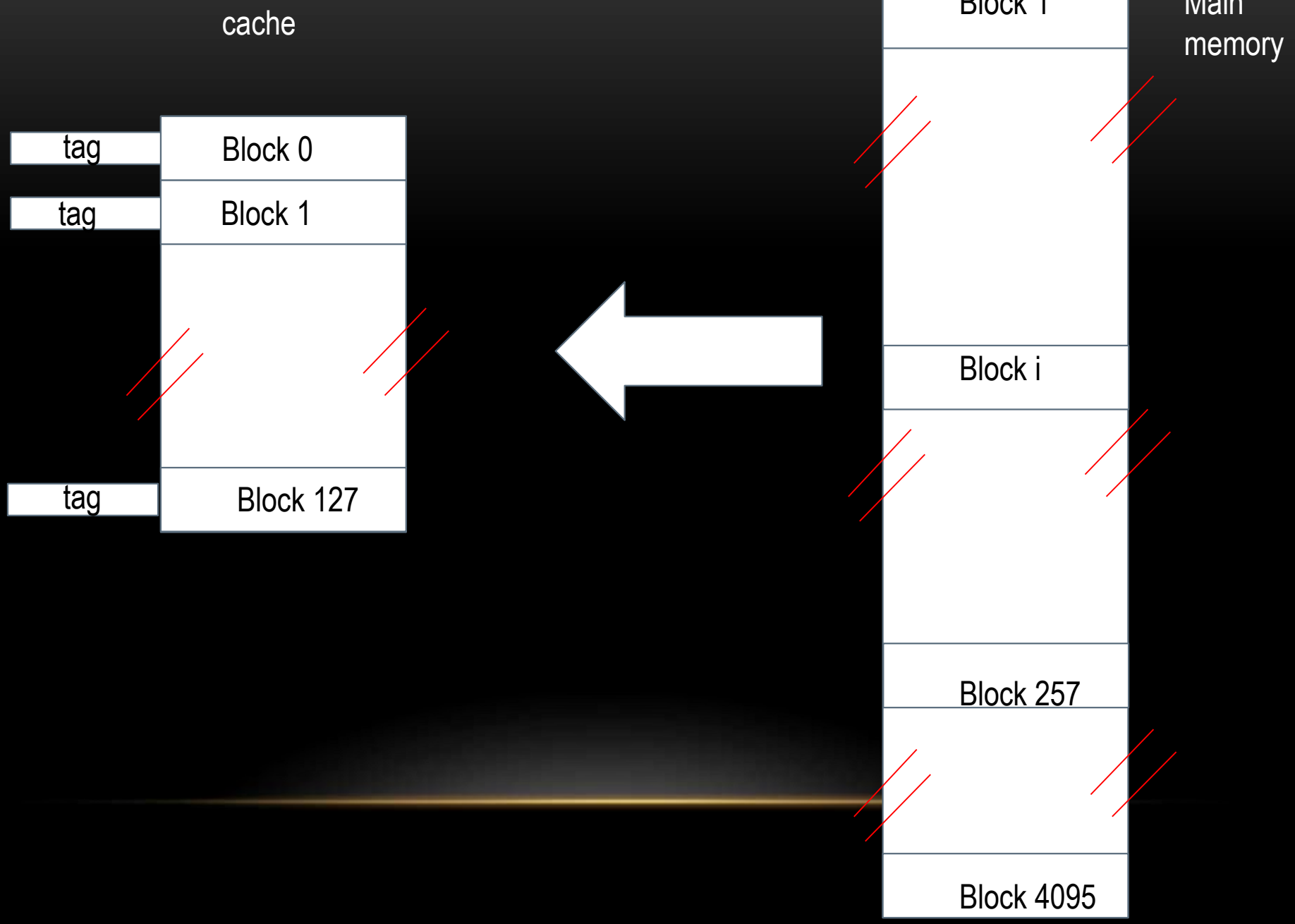
3. Set-associative mapping

- It is the combination of direct and associative mapping. The blocks of the cache are grouped into sets and the mapping allows a block of the main memory to reside in any block of the specified set.

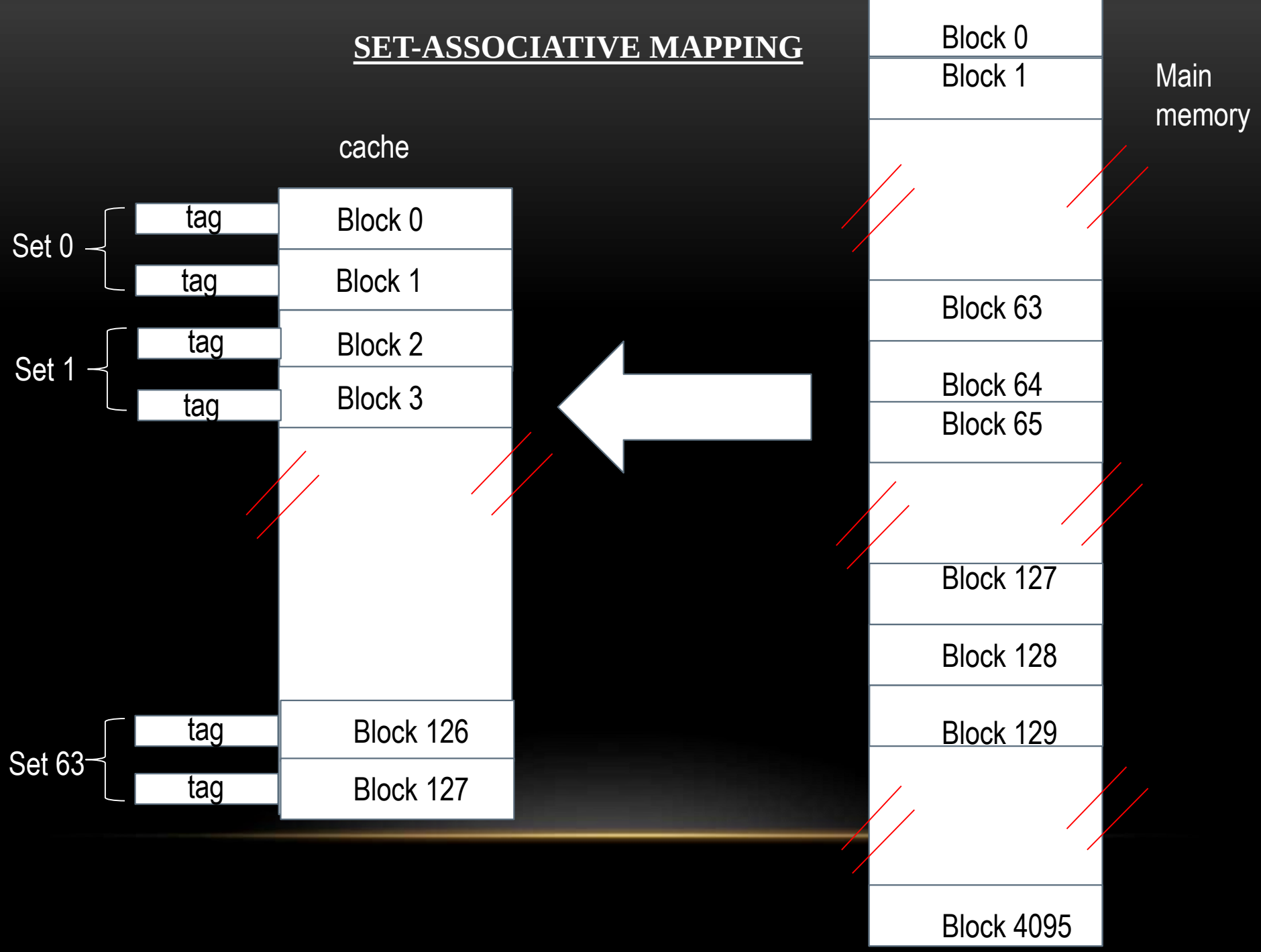
DIRECTION MAPPING



ASSOCIATIVE MAPPING



SET-ASSOCIATIVE MAPPING



- The Cache memory stores a reasonable number of blocks at a given time but this number is small compared to the total number of blocks available in Main Memory.
- The correspondence between main memory block and the block in cache memory is specified by a mapping function.
- The Cache control hardware decide that which block should be removed to create space for the new block that contains the referenced word. The collection of rule for making this decision is called the **replacement algorithm.**
- The cache control circuit determines whether the requested word currently exists in the cache. If it exists, then Read/Write operation will take place on appropriate cache location.
- In this case Read/Write hit will occur.

REPLACEMENT POLICIES

1. First In First Out (FIFO)

Using this algorithm the cache behaves in the same way as a FIFO queue. The cache evicts the first block accessed first without any regard to how often or how many times it was accessed before.

2. Last In First Out (LIFO)

Using this algorithm the cache behaves in the exact opposite way as a FIFO queue. The cache evicts the block accessed most recently first without any regard to how often or how many times it was accessed before.

3. Least Recently Used (LRU)

Discards the least recently used items first. This algorithm requires keeping track of what was used when, which is expensive if one wants to make sure the algorithm always discards *the* least recently used item. General implementations of this technique require keeping "age bits" for cache-lines and track the "Least Recently Used" cache-line based on age-bits.

THANK YOU