**Cache Memory**

- The speed of the main memory is very low in comparison with the speed of modern processors.
- For good performance, the processor cannot spend much of its time waiting to access instructions and data in main memory.
- Hence, it is important to devise a scheme that reduces the time needed to access the necessary information.
- An efficient solution is to use a fast ***cache memory,*** which essentially makes the main memory appear to the processor to be faster than it really is.
- The cache is a smaller, faster memory which stores copies of the data from the most frequently used main memory locations.

- The effectiveness of the cache mechanism is based on a property of computer programs called ***locality of reference.***
- *Locality of Reference :*Many instructions in localized areas of the program are executed repeatedly during some time period and remainder of the program is accessed relatively infrequently.
- It manifests itself in 2 ways.

They are :

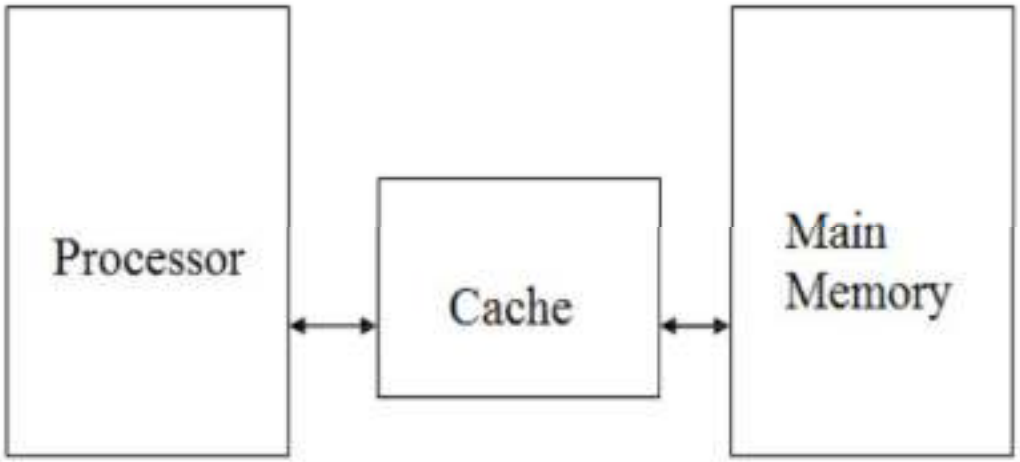
- Temporal: The recently executed instruction are likely to be executed again very soon.

  Examples: Loops, repeatedly called subroutines, setting a variable and then reusing it many times
- Spatial: The instructions in close proximity to recently executed instruction are likely to be executed soon.

  Examples: Arrays and program code

- If the active segment of the program is placed in cache memory, then the total execution time can be reduced significantly.
- The temporal aspects of the locality of reference suggest that whenever an instruction item(instruction or data) is first needed, this item should be brought into the cache where it will hopefully remain until it is needed again.
- The spatial aspects suggests that instead of fetching just one item from the main memory to the cache, it is useful to fetch several items that reside at adjacent addresses as well.
- The term *block* is used to refer to a set of contiguous address locations of some size.
- Another term that is used to refer to a cache block is *cache line*.

# Use of a Cache Memory



- When a Read request is received from the processor ,the contents of a block of memory words containing the location specified are transferred into the cache one word at a time.
- When the program references any of the locations in this block, the desired contents are read directly from the cache.
- 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*

- When the cache is full and a memory word (instruction or data) that is not in the cache is referenced, the cache control hardware must decide which block should be removed to create space for the new block that contains the referenced word.
- The collection of rules for making this decision constitutes 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.
- In a Read operation, the memory will not be involved.

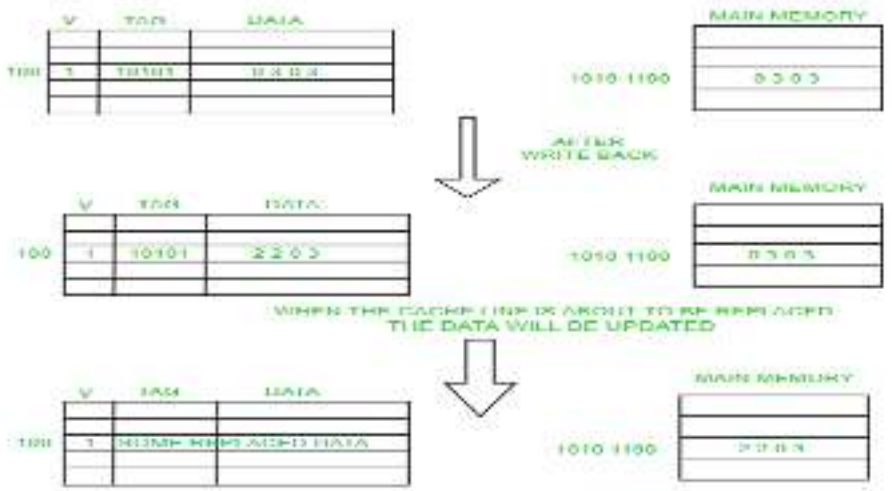
- The write operation proceeds in 2 ways. They are:
  - Write-through protocol
  - Write-back protocol

Write-through protocol:

- Here the cache location and the main memory locations are updated simultaneously.
- The write- through protocol is simpler, but it results in unnecessary write operations in the main memory when a given cache word is updated several times during its cache residency.

Write-back protocol:

- This technique is to update only the cache location and to mark it as with associated flag bit called dirty/modified bit.
- The word in the main memory will be updated later, when the block containing this marked word is to be removed from the cache to make room for a new block.
- If the requested word currently does not exist in the cache during read operation, then read miss will occur.
- To overcome the read miss Load through / early restart protocol is used.



- When the addressed word in a Read operation is not in the cache, a Read Miss occurs.

- The block of words that contains the requested word is copied from the main memory into the cache.

- After the entire block is loaded in to the cache, the particular word request is forwarded to the processor.

- During Write operation ,if the addressed word is not in the cache, a write miss occurs

MAPPING FUNCTIONS

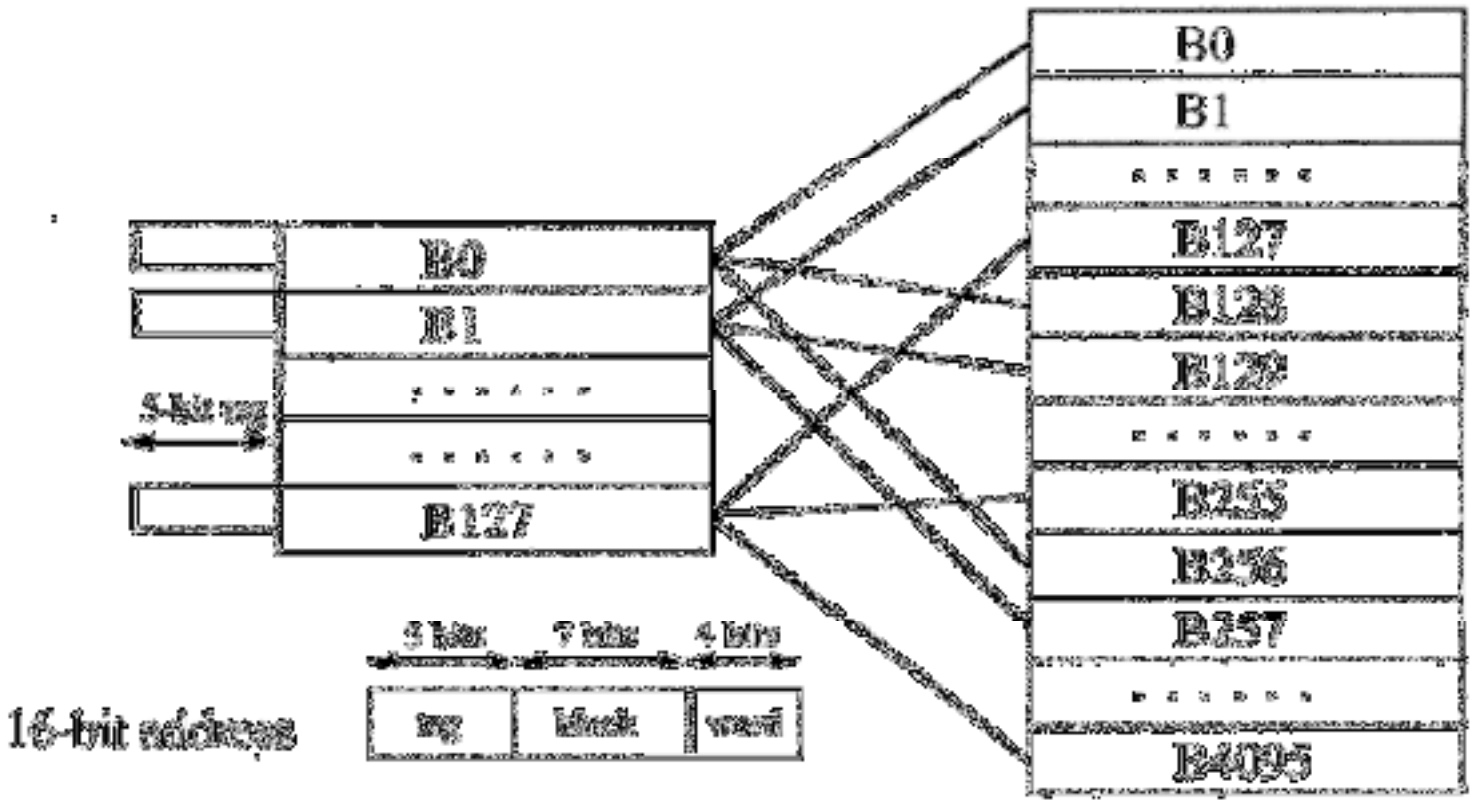
- To specify where memory blocks are placed in the cache

Example:

- Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048(2K) words.
- Main memory is addressable by a 16-bit address, it has 64K words. I.e. 4K blocks of 16 words each.
- Consecutive addresses refer to consecutive words.

Direct Mapping:

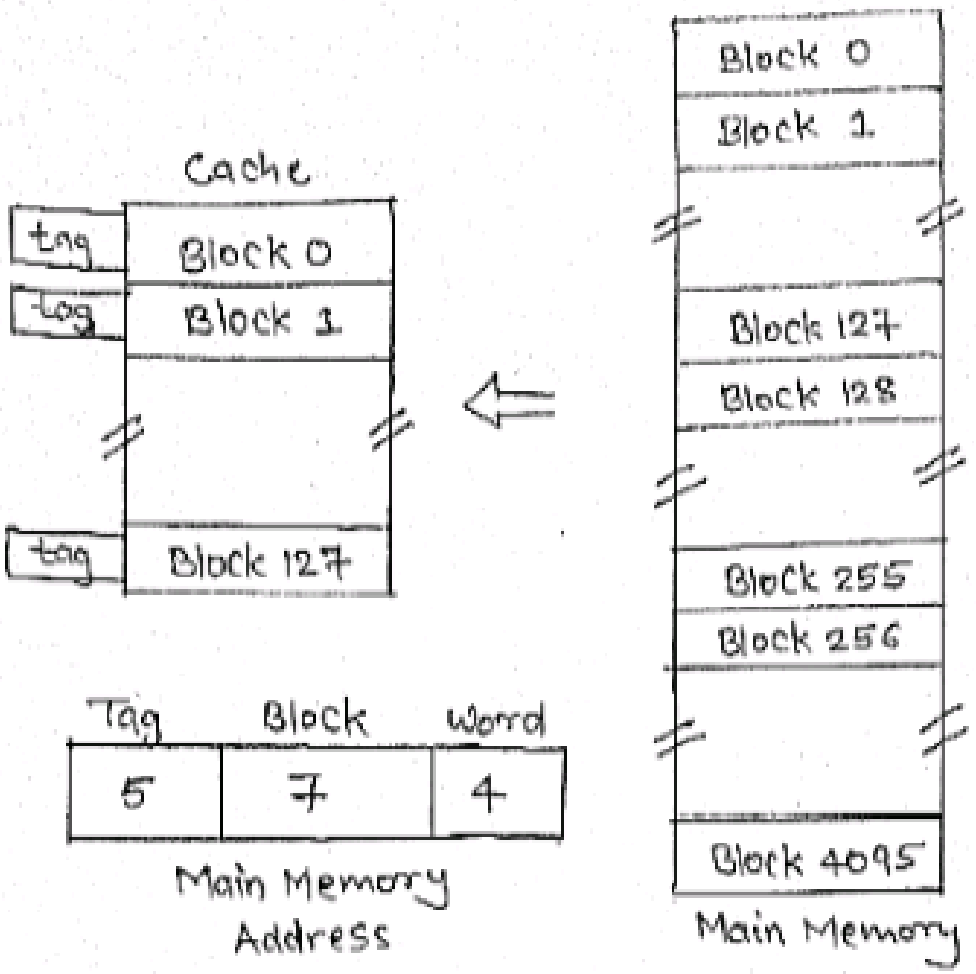
- It is the simplest technique in which block j of the main memory maps onto block j modulo 128 of the cache.
- Thus whenever one of the main memory blocks 0, 128, 256 is loaded in the cache, it is stored in block 0.
- Block 1, 129, 257 are stored in cache block 1 and so on.
- The contention may arise when the cache is full, when more than one memory block is mapped onto a given cache block position.
- The contention is resolved by allowing the new blocks to overwrite the currently resident block.
- Placement of block in the cache is determined from memory address.


B0
B1
B127
B128
B255
B256
B257
B4095
B0
B1
B127
16-bit address

- The memory address is divided into 3 fields, namely,
- Low Order 4 bit field (word) → Select one of 16 words in a block.
- 7 bit cache block field → When new block enters cache, 7 bit determines the cache position in which this block must be stored.
- 5 bit Tag field → The high order 5 bits of the memory address of the block is stored in 5 tag bits associated with its location in the cache.
- They identify which of the 32 blocks that are mapped into this cache position are currently resident in the cache.

---

- Block size =2^4=16 words
- Cache memory=2^11=2048 words=2^7=128 blocks
- Main memory=2^16=64K words=2^12=4096 blocks
- No.of cache lines=2^11/2^4=2^7,thus 7 bits are required to locate 2^7 lines.
- Tag bits=16-7-4=5 bits


Cache
tag
Block 0
tag
Block 1
tag
Block 127
Block 0
Block 1
Block 127
Block 128
Block 255
Block 256
Block 4095
Tag
Block
Word
5
7
4
Main Memory Address
Main Memory

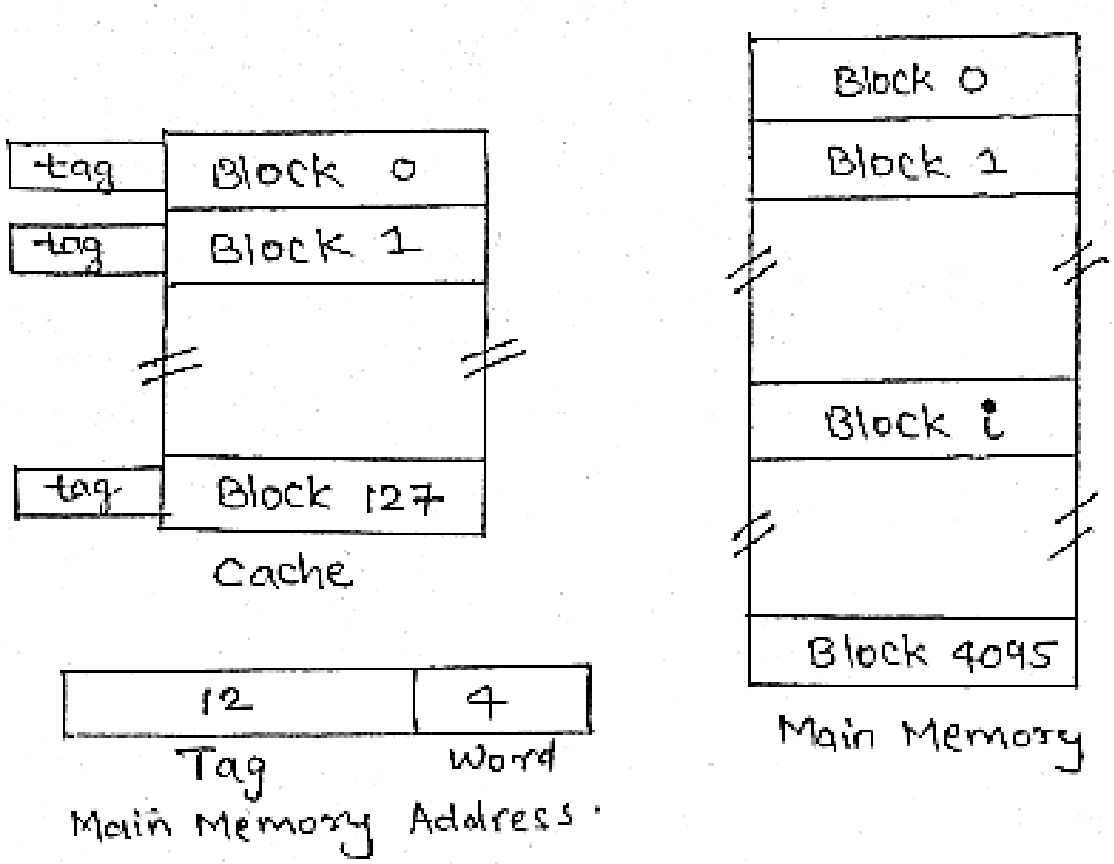
- As execution proceeds, the high order 5 bits of the address is compared with tag bits associated with that cache location.
- If they match, then the desired word is in that block of the cache.
- If there is no match, then the block containing the required word must be first read from the main memory and loaded into the cache
- Merit:

It is easy to implement.

Demerit:

- It is not very flexible.

## ASSOCIATIVE-MAPPING



- Here, 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 present. This is called *associative mapping*. It gives complete freedom in choosing the cache location.
- A new block that has to be brought into the cache has to replace (eject) an existing block if the cache is full. In this case, we need an algorithm to select the block to be replaced.

- In this method, the memory has to determine whether a given block is in the cache. A search of this kind is called an associative search.

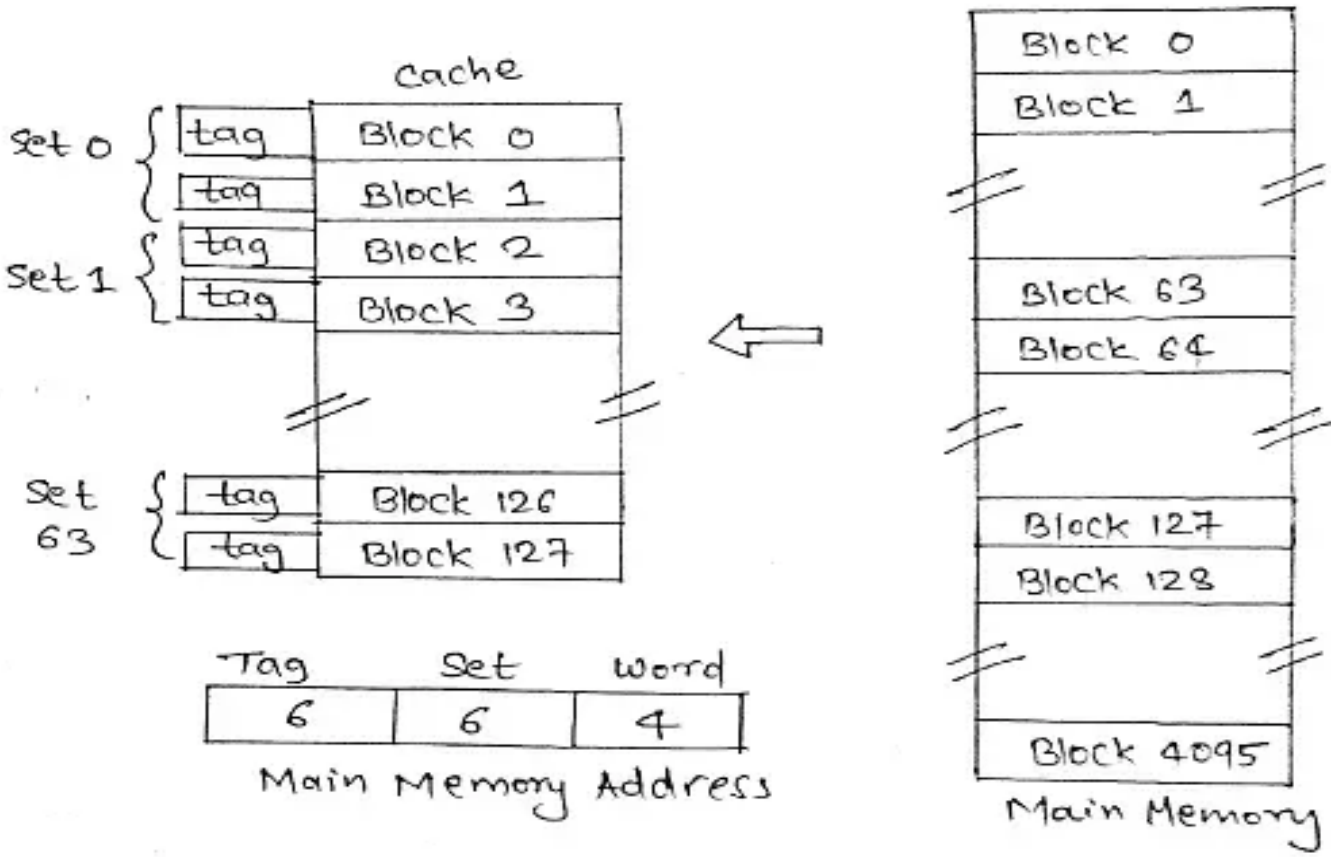
Merit:

- It is more flexible than direct mapping technique.

Demerit:

- Its cost is high.

## 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. In this case, the cache has two blocks per set, so the memory blocks 0, 64, 128…… map into cache set to 0 and they can occupy either of the two block position within the set.
- 6 bit set field → Determines which set of cache contains the desired block.
- 6 bit tag field → the tag field of the address is compared to the tags of the two blocks of the set to clock if the desired block is present.

- The cache which contains 1 block per set is called direct -mapping
- A cache that has k blocks per set is called as k-way set associative cache.
- Each block contains a control bit called a *valid bit*.
- The Valid bit indicates that whether the block contains valid data. The dirty bit indicates that whether the block has been modified during its cache residency
- Valid bit=0→When power is initially applied to system
- Valid bit =1→When the block is loaded from main memory at first time.

- If the main memory block is updated by a source and if the block in the source is already exists in the cache, then the valid bit will be cleared to 0. If Processor and DMA use the same copies of data then it is called as the Cache Coherence Problem.

Merit:

- The Contention problem of direct mapping is solved by having few choices for block placement.
- The hardware cost is decreased by reducing the size of associative search