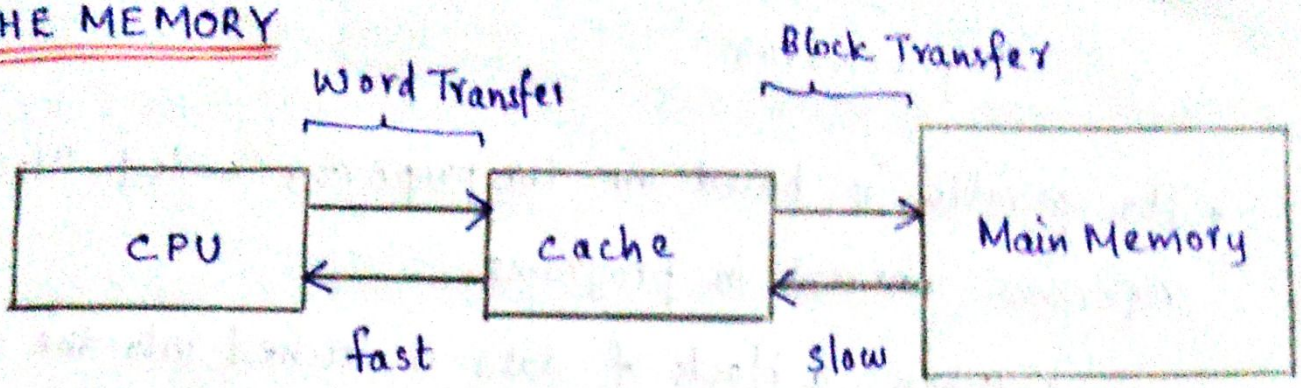
CASHE MEMORY



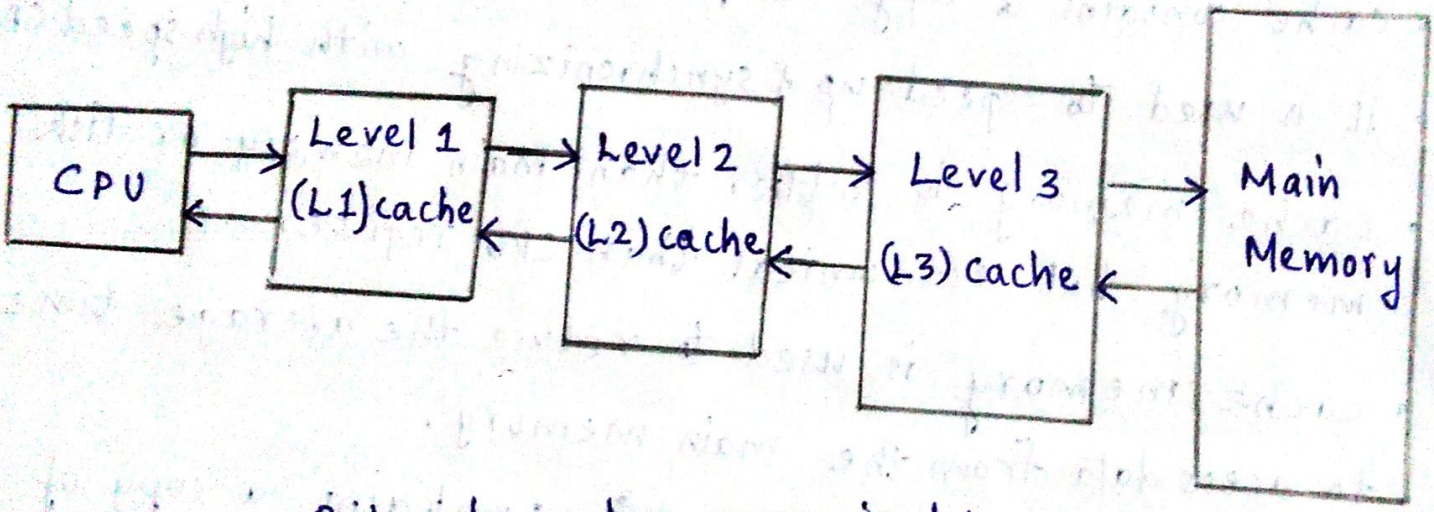
Single cashe.

~~cache~~

- cashe memory is small & fast memory used to increase the instruction-processing rate.
- cache contains a copy of portions of main memory
- It is used to speed up & synchronizing with high speed CPU
- cache memory is costlier than main memory or disk memory but economical than CPU registers.
- cache memory is used to reduce the average time to acess data from the main memory.
- cache memory ~~is small & to~~ contains a copy of portions of main memory.
- When the processor attempts to read a word of memory, a check is made to determine if the word is in the cache.
  - ✓ if so, the word is delivered to the processor
  - ✓ if not, a block of main memory, consisting of some fixed number of words, is read into the cache and then the word is delivered to the

processor

- Its operation is based on the property called "locality of reference" inherent in programs.
  - ✓ When a block of data is fetched into the cache to satisfy a single memory reference, it is likely that there will be future references to that same memory location or to other words in the block



3-level cache organization

Above fig depics the use of multiple levels of cache. The L2 cache is slower & typically larger than the L1 cache & the L3 cache is slower & typically larger than the L2 cache.

- Level 1 : It is a type of memory in which data is stored & accepted that are immediately stored in CPU.
- Level 2 : It is the fastest memory which has faster access time where data is temporarily stored for faster access.
- Level 3: It is memory on which computer works currently It is small in size & once power is off data no longer stays in this memory
- Level 4 : It is external memory which is not as fast as main memory but data stays permanently in this memory.

Cache Performance:

When the processor needs to read or write a location in main memory, it first checks for a corresponding entry in the cache.

* If the processor finds that the memory location is in the cache, a cache hit has occured & data is read from cache
* If the processor does not find the memory location in the cache, a cache miss has occured. For a cache miss, the cache allocates a new entry & copies in data from main memory, then the request is fulfilled from the contents of the cache.

Cache Performance Improvement

The performance of cache memory is frequently measured in terms of a quantity called Hit ratio.

$$\text{Hit ratio} = \frac{\text{no. of hits}}{(\text{hit} + \text{miss})} = \frac{\text{no. of hits}}{\text{Total accesses}}$$

We can improve cache performance using higher cache block size, higher associativity, reduce miss rate, reduce miss penalty and reduce the time to hit in the cache.

CASHE MAPPING TECHNIQUES

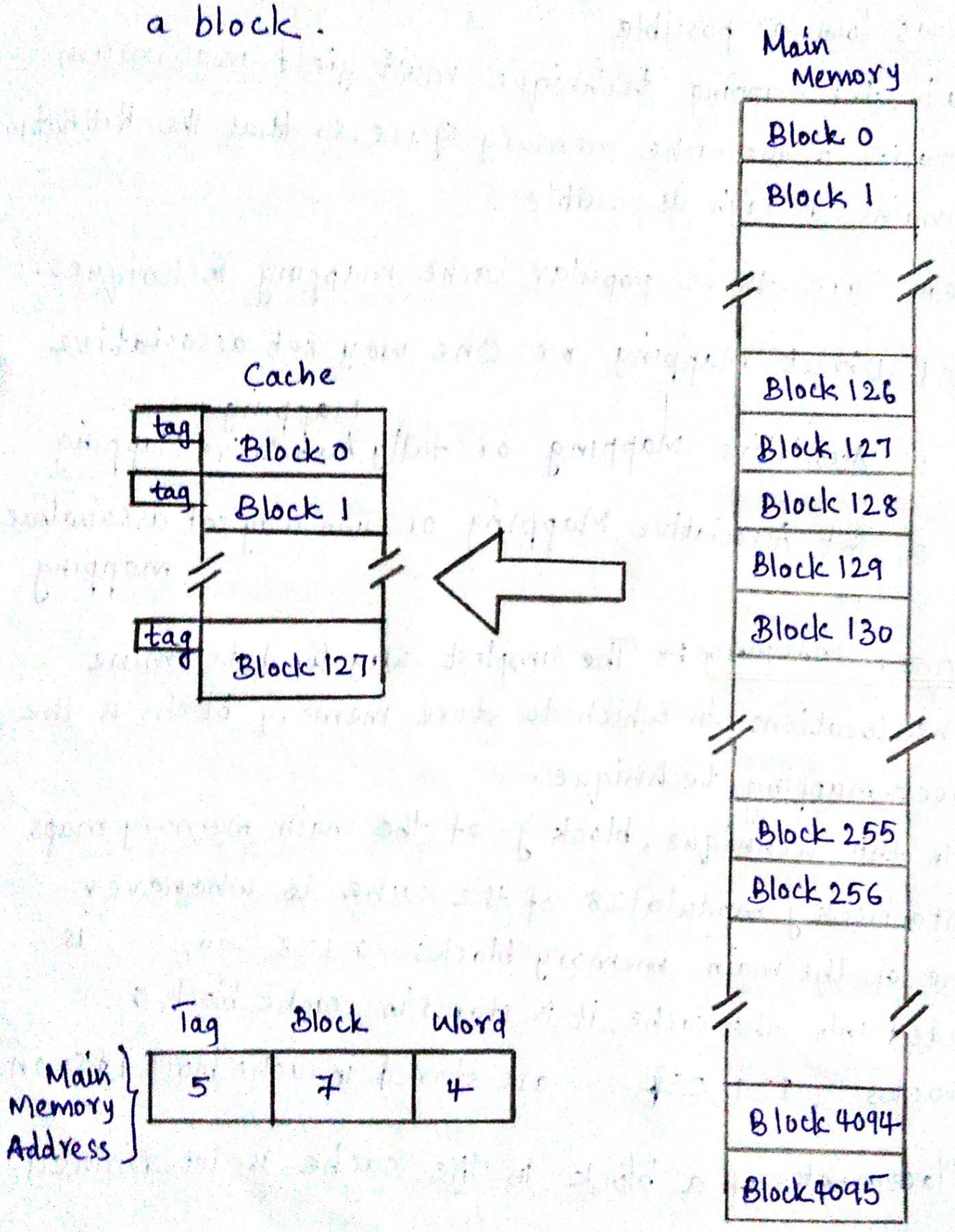
- Blocks are loaded from main memory to cache memory
- Cache mapping decides which block of main memory comes into which block of cache memory
- There are several mapping techniques trying to balance b/w hit ratio, search time & tag size.
- Every cache block has a tag indicating which block of main memory is mapped into that block.
- The main memory address, issued by the processor contains the desired block number
- This is compared to the tag of a cache block, which gives the block number that is present
- If they are equal, it's a Hit. If not, the search

may have to be repeated for several other blocks.

- It is obvious to understand, the no. of searches must be as low as possible.
- Finally, the mapping technique must yield maximum utilization of the cache memory space, so that the Hit Ratio remains as high as possible.
- There are three popular cache mapping Techniques:
  1. Direct Mapping or One way set associative Mapping
  2. Associative Mapping or fully Associative Mapping
  3. Set Associative Mapping or Two way set associative mapping

1. DIRECT MAPPING :- The simplest way to determine cache locations in which to store memory blocks is the direct-mapping technique.
   - In this technique, block j of the main memory maps onto block j modulo 128 of the cache ie whenever one of the main memory blocks 0, 128, 256, ... is loaded into the cache, it is stored in cache block 0.
   - Blocks 1, 129, 257 .... are stored in cache block 1 & so on
   - Placement of a block in the cache is determined by its memory address.

- Memory address can be divided into 3 fields - low order 4 bits select one of the 16 words in a block.



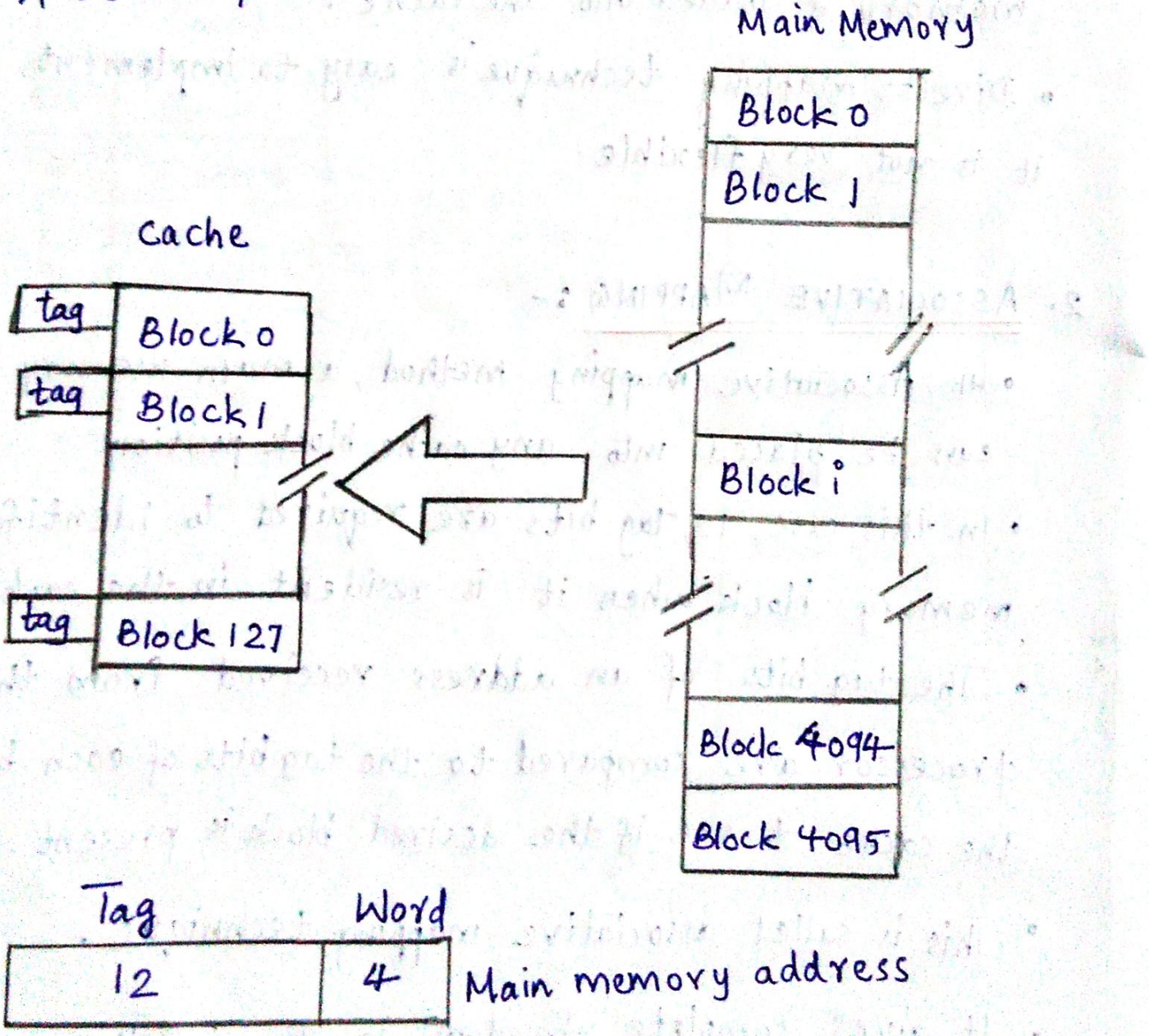
- When a new block enters the cache, the 7-bit cache block field determines the cache position in which this block must be stored.
- If they match, then the desired word is in that block of the cache.
- If ~~there~~ no is no match, then the block containing the required word must first be read from the main memory & loaded into the cache.
- Direct mapping technique is easy to implement, but it is not very flexible.

2. ASSOCIATIVE MAPPING :-

- In Associative mapping method, a main memory block can be placed into any cache block position
- In this case, 12 tag bits are required to identify a memory block when it is resident 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 technique.
- It gives complete freedom in choosing the cache location in which to place the memory block, resulting in a more efficient use of the space in

the cache.

- When a new block is brought into the cache, it replaces an existing block only if the cache is full.
- In this case, we need an algorithm to select the block to be replaced.
- To avoid a long delay, the tags must be searched in parallel.
- A search of this kind is called an associative search.



Main memory address

3. SET-ASSOCIATIVE MAPPING :-

- It uses a combination of direct & associative mapping techniques.
- The blocks of the cache are grouped into sets & the mapping allows a block of the main memory to reside in any block of a specific set.
- Hence contention problem of the direct method is eased by having a few choices for block placement.

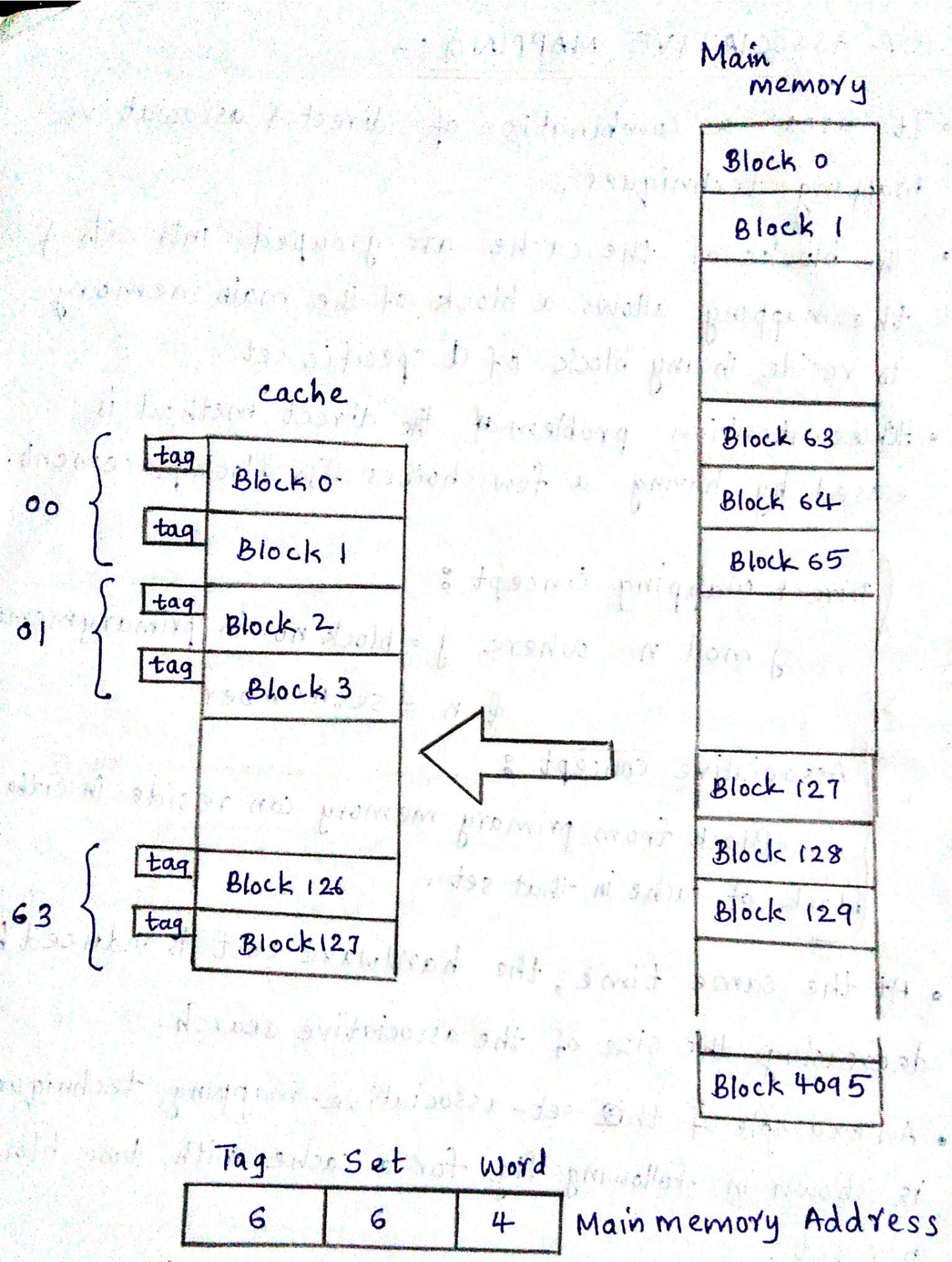
Direct Mapping concept :

$j \bmod n$ where j = block no. in primary memory & n = set number

Associative concept :

Block from primary memory can reside in either block of cache in that set.

- At the same time, the hardware cost is reduced by decreasing the size of the associative search.
- An example of this set-associative-mapping technique is shown in following fig. for a cache with two blocks per set.
- In this case, memory blocks 0, 64, 128, ..., 4032 map into cache set 0 & they can occupy either of the two block positions within this set.



Main memory Address

Number of set = $\frac{\text{Number of blocks}}{2}$

$= \frac{128}{2} = 64$

- Having 64 sets means that the 6 bit set field of the address determines which set of the cache might contain the desired block.
- The tag field of the address must then be associatelly compared to the tags of the two blocks of the set to check if the desired block is present.
- This two-way associative search is simple to implement.
- The number of blocks per set is a parameter that can be selected to suit the requirements of a particular computer.
- For the main memory & cache sizes in figure, 4 blocks per set can be accomodated by a 5 bit set field, eight blocks per set by a 4 bit set field & so on.
- The extreme condition of 128 blocks per set requires no set bits & corresponds to the fully-associative technique, with 12 tag bits.
- The other extreme of one block per set is the direct-mapping

Application of Cache Memory

- Usually, the cache memory can store a reasonable no. of blocks at any given time, but this number is small compared to the total number of blocks in the main memory
- The correspondance between the main memory blocks & those in the cache is specified by a mapping function.

Types of cache

- Primary cache – A primary cache is always located on the processor chip. This cache is small & its access time is comparable to that of processor registers
- Secondary cache – A secondary cache is placed between the primary cache & the rest of the memory. It is referred to as the level 2 (L2) cache. Often, the Level 2 cache is also housed on the processor chip.