**Experiment 7:** Implementation of K- way Set Associative Mapping technique of Cache memory.

Aim : To study the concept of Memory mapping (Set Associative Mapping) with the help of an example.

**Theory**: The transformation of data from main memory to cache memory is referred to as a mapping process. Cache mapping is a technique by which the contents of main memory are brought into the cache memory.

There are three different types of mapping used for the purpose of cache memory which are as follows:

1. Direct Mapped: Each block has only one place that it can appear in the cache.
2. Fully Associative mapping: Each block can be placed anywhere in the cache.
3. Set-Associative mapping: Each block can be placed in a restricted set of places in the cache.
   * If there are n blocks in a set, the cache placement is called n-way set associative

**For example:- Consider Cache size is 8 blocks. Where does word 12 from memory go?**

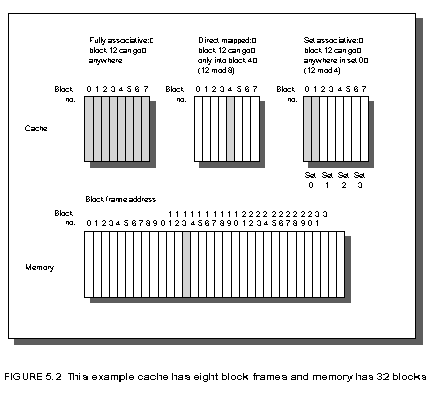
Solution :- 1. Fully associative:

* + Block 12 can go anywhere

1. Direct mapped:
   * Block no. = (Block address) mod (No. of blocks in cache)
   * Block 12 can go only into block 4 (12 mod 8 = 4)
2. 2-way set associative Mapping:

Cache Set no. = (Main Memory Block address) modulo (No. of sets in cache)

Block 12 can go anywhere in set 0 (12 mod 4 = 0)

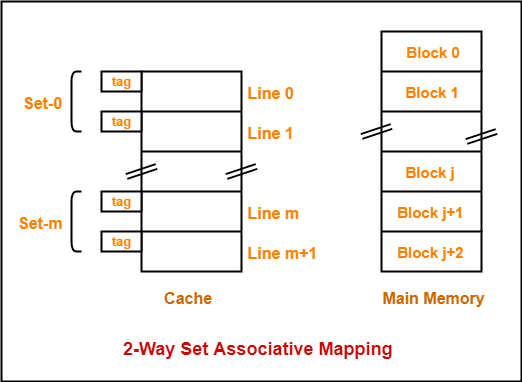


**Set associative mapping :-**

* In Set associative mapping, Cache lines are grouped into sets where each set contains k number of lines.
* A particular block of main memory can map to only one particular set of the cache.
* If there are n blocks in a set, the cache placement is called n-way set associative
* However, within that set, the memory block can map any cache line that is freely available.

**How to select set no.?**

**Cache set number** **= ( Main Memory Block Number ) Modulo (Number of sets in Cache)**



**Numerical on Set Associative addressing**

**(S1 BATCH)- Example-1-** Consider a 4 – way set associative mapped cache with block size 4 KB . The Size of the main memory is 16 GB and there are 10 bits in the tag.   
Find –  
Size of Cache Memory and   
Tag directory size

**(S2 BATCH)- Example-2.** Consider a 2-way set associative mapped cache of size 16 words with block size 4 words. The size of main memory is 128 words. Find-

* Number of bits in tag
* Tag directory size

**(S3 BATCH)- Example-3-** A block-set associative cache memory consists of 128 blocks divided into 4 block sets. The main memory consists of 16,384 blocks and each block contains 256 eight bit words.

1. How many bits are required for addressing the main memory?
2. How many bits are required to represent the TAG, SET and WORD fields?

**(S4 BATCH)- Example-4-** Consider a 2-way set associative mapped cache of size 16 KB with block size 256 bytes. The size of main memory is 128 KB. Find-

* Number of bits in tag
* Tag directory size

**Q1 — Small-word cache**

**Problem:** 2-way set-associative cache

* Cache size = **16 words**
* Block size = **4 words**
* Main memory size = **128 words**

**Step 1 — address size (main memory)**

Main memory has 128 words → number of address bits = log₂(128) = 7 bits.  
(2⁷ = 128)

**Step 2 — block offset bits**

Block size = 4 words → offset bits = log₂(4) = 2 bits.  
(2² = 4)

**Step 3 — number of blocks and sets**

Number of blocks in cache = cache size / block size = 16 / 4 = 4 blocks.  
Associativity = 2 → number of sets = number of blocks / 2 = 4 / 2 = 2 sets.  
Set index bits = log₂(2) = 1 bit.

**Step 4 — tag bits**

Tag bits = (address bits) − (index bits + offset bits)  
= 7 − (1 + 2) = 7 − 3 = **4 bits**

**Step 5 — tag directory size**

Total cache lines (blocks) = 4.  
Tag directory (just tags) = number\_of\_lines × tag\_bits = 4 × 4 = **16 bits**.

If you include one valid bit per line, add 4 valid bits: 16 + 4 = **20 bits**.

**Answer Q1:**

* Number of bits in tag = **4 bits**.
* Tag directory size = **16 bits** (or **20 bits** if you include valid bits).

**Q2 — Byte-addressed cache**

**Problem:** 2-way set-associative cache

* Cache size = **16 KB** = 16 × 1024 = **16384 bytes**
* Block size = **256 bytes**
* Main memory size = **128 KB** = 128 × 1024 = **131072 bytes**

**Step 1 — address size (main memory)**

Main memory size 131072 bytes → address bits = log₂(131072) = 17 bits.  
(2¹⁷ = 131072)

**Step 2 — block offset bits**

Block size = 256 bytes → offset bits = log₂(256) = 8 bits.  
(2⁸ = 256)

**Step 3 — number of blocks and sets**

Number of blocks in cache = cache size / block size = 16384 / 256 = 64 blocks.  
Associativity = 2 → number of sets = 64 / 2 = 32 sets.  
Set index bits = log₂(32) = 5 bits.

**Step 4 — tag bits**

Tag bits = address bits − (index bits + offset bits)  
= 17 − (5 + 8) = 17 − 13 = **4 bits**

**Step 5 — tag directory size**

Total cache lines (blocks) = 64.  
Tag directory (just tags) = 64 × 4 = **256 bits** = **32 bytes**.

If you include one valid bit per line, add 64 bits: 256 + 64 = **320 bits** = **40 bytes**.

**Answer Q2:**

* Number of bits in tag = **4 bits**.
* Tag directory size = **256 bits** (i.e., **32 bytes**) — or **320 bits** (**40 bytes**) if you include valid bits.