CACHE 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 between Hit Ratio, Search-time and Tag size.
Every cache block has a **Tag** indicating which block of Main Memory is mapped into that block.
A collection of such tags is called the **cache director**y, very similar to a page table.
Cache directory is a part of the cache.
Since Cache Memory is **very expensive**, we need the cache directory to be as small as possible.
That means the **Tag must be of minimum size**.
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 number 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) **Associative Mapping** also called Fully Associative Mapping
2) **Direct Mapping** also called One-Way Set Associative Mapping
3) **Set Associative Mapping** also called Two-Way Set Associative Mapping

ASSOCIATIVE MAPPING (FULLY ASSOCIATIVE MAPPING)

During memory operations, Blocks are loaded from Main Memory to Cache Memory.
Cache Mapping decides which block of Main Memory comes into which block of Cache Memory.
Fully Associative Mapping technique states:
Any block of Main Memory can be mapped at Any available block of Cache Memory.
There are no rules restricting the mapping at all.
This means the Full Cache is available for mapping hence the name Fully Associative.

Consider Pentium Processor Cache

Size of Main Mem:	**4GB** = $\mathbf{2^{32}}$
Size of Cache Mem:	**8KB** = $\mathbf{2^{13}}$
Size of Cache Block (Line):	**32** bytes (words) = $\mathbf{2^5}$
No. of Blocks in Main Mem:	Size of Main Memory ($\mathbf{2^{32}}$) ÷ Size of Block ($\mathbf{2^5}$) = $\mathbf{2^{27}}$
No. of Blocks in Cache Mem:	Size of Cache Memory ($\mathbf{2^{13}}$) ÷ Size of Block ($\mathbf{2^5}$) = $\mathbf{2^8}$ = **256**
Main Mem address:	**32 bits** (because main Mem is of **4GB** = $\mathbf{2^{32}}$)

Tag Size:
A block of Cache Memory can **contain any block** of main memory out of a possible $\mathbf{2^{27}}$ **blocks**. Hence, the **Tag** next to every block in Cache Memory must be of **27 bits**.

Searches:
A block of main Memory can be **mapped into any block** of Cache Memory out of **256 Blocks**. Hence, we need to do **256 searches** in Cache Memory.

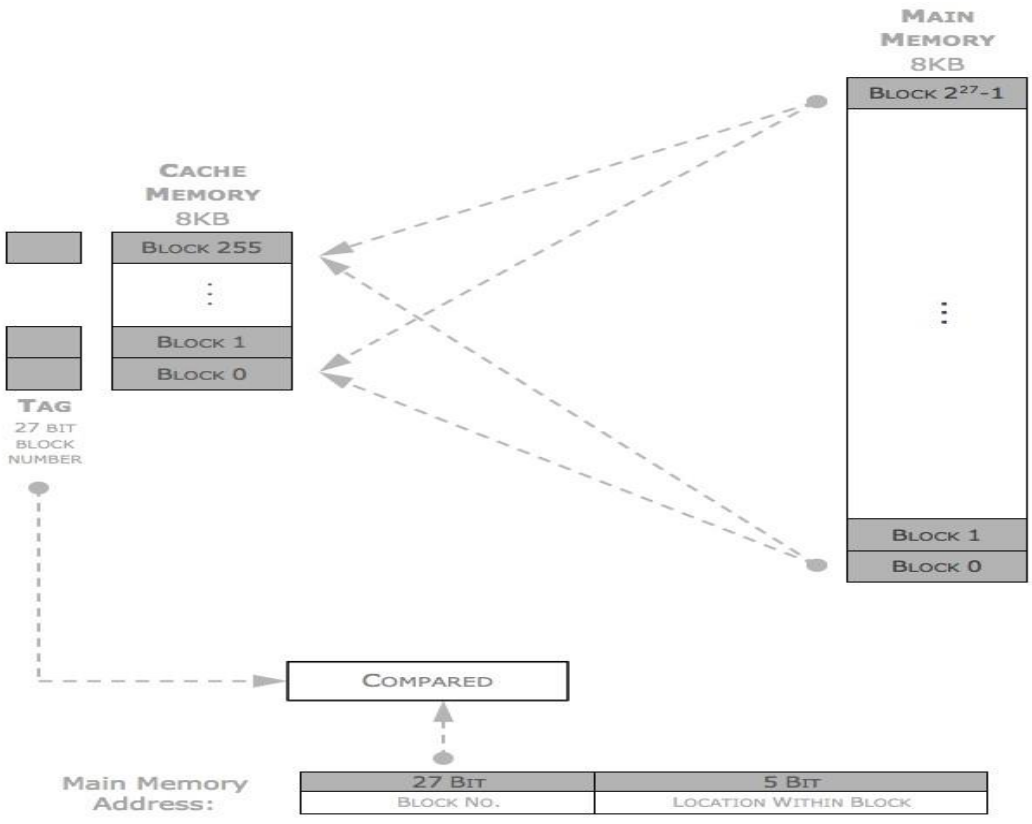
Method of Searching:
The Processor issues a 32 bit Main Memory address. It can be divided as:

Main Memory Address:	27 BIT	5 BIT
	Block No.	Location Within Block

This 27 bit Block number is the block number we need to search.
The Tag of each cache block also contains a 27-bit block number.
This is the block number that's present in that respective cache block.
These two block numbers are compared. **If they are equal, it's a HIT.**
If not equal, the search is repeated with the Tag of the next cache block.
This is done a total of 256 times as there are 256 blocks in Cache Memory.
If none of them match with the block number we are searching, then it's a Miss.

Advantage
Since the full cache is available for mapping, it causes maximum utilization of Cache Memory hence gives the **Best Hit Ratio.**

Drawback
Tag Size too big: **27bits.**
Searches are too many: **256.**



DIRECT MAPPING (ONE WAY SET ASSOCIATIVE MAPPING)

Direct Mapping technique states:
Any block of Main Memory can only be mapped at ONE block of Cache Memory.
Since there is only one way of Mapping, its also called One Way Set Associative Mapping.
We treat the entire Cache as One Set.
The Main Memory is divided into Sets which are then subdivided into Blocks.
A Block of Main Mem. (of any set), can only be mapped into the same Block No. in Cache Mem.
This means, Block 0 of Main Memory (of any set), can only be mapped into Block 0 of Cache Memory. In other words, Block 0 of Cache Memory can only contain Block 0 of Main Memory but of any Set.

Consider Pentium Processor Cache

Size of Main Mem:	**4GB** = 2^{32}
Size of Cache Mem:	**8KB** = 2^{13} … this is treated as One Set
Hence Size of Set:	**8KB** = 2^{13}
Size of Cache Block (Line):	**32** bytes (words) = 2^{5}
No. of Blocks in a set:	Size of Set (2^{13}) ÷ Size of Block (2^{5}) = 2^{8} = **256**
No. of Sets in Main Mem:	Size of Main Mem (2^{32}) ÷ Size of Set (2^{13}) = 2^{19}
No. of Sets in Cache Mem:	1
Main Mem address:	**32 bits** (because main Mem is of **4GB** = 2^{32})

Tag Size:
Since, Block 0 of Cache Memory can only contain Block 0 of Main Memory but of any Set, the Tag has to only indicate the Set No. of Main Memory, from which the block is present. As Main Memory has 2^{19} sets, the **Tag** size is **19 bits**.

Searches:
If we need Block 0 if Main Memory, we only need to search Block 0 of Cache Memory. Hence, we need to do only **1 search** in Cache Memory to know if it is a Hit or a Miss.

Method of Searching:
The Processor issues a 32 bit Main Memory address. It can be divided as:

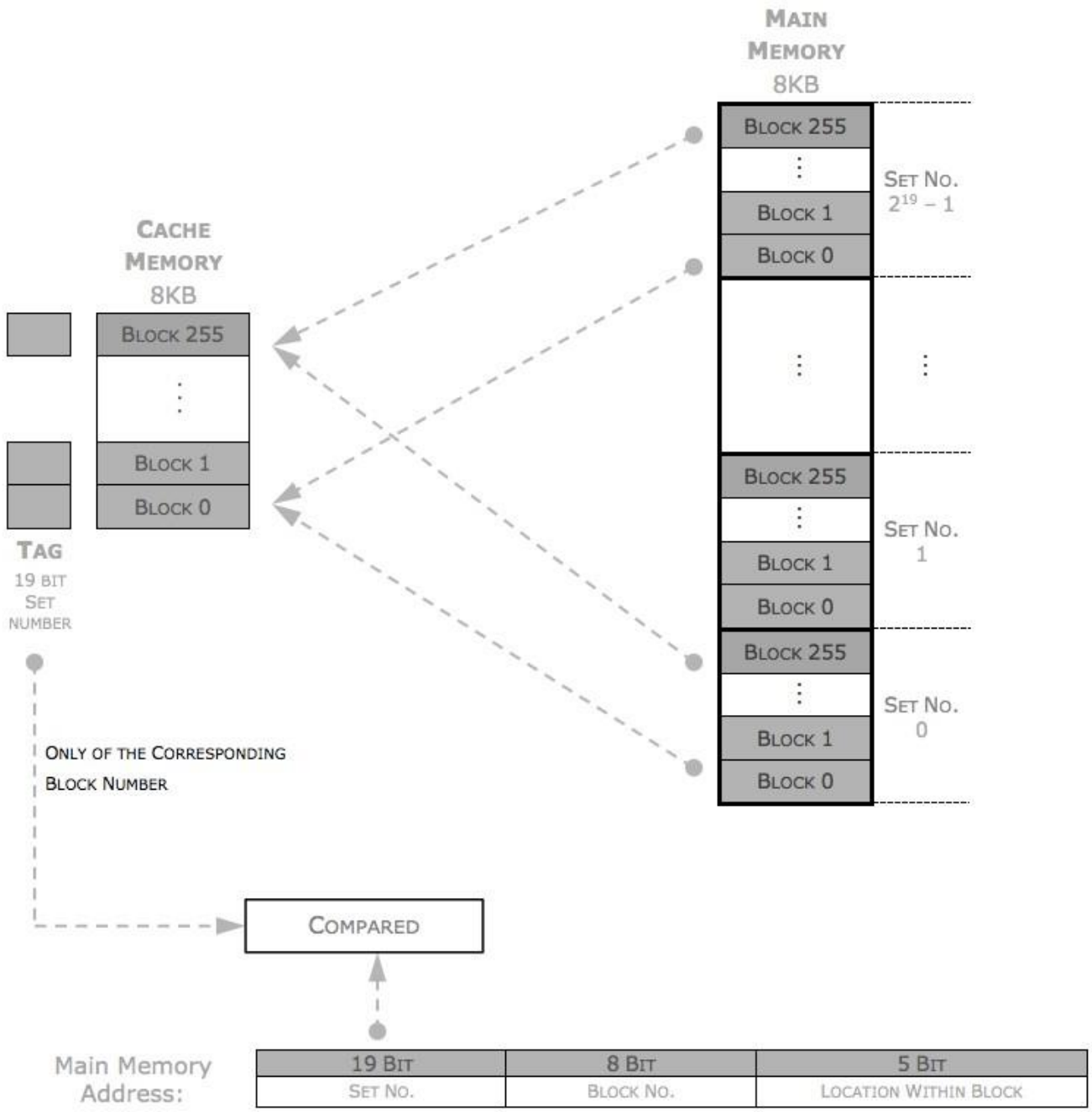
Main Memory Address:

19 BIT	8 BIT	5 BIT
Set No.	Block No.	Location Within Block

First we look at the block no. we need, to know where we need to search.
We then look at the Set No. that we need and compare it with the Tag of the corresponding Block no in the Cache Memory.
Assume the Main Memory address is 5:0:6. This means we need location 6 of Block 0 of set 5.
We go to Block 0 of Cache Memory.
It has a Tag, which gives the Set No of Main Memory whose Block 0 is present in Cache Memory. These two set numbers are compared. **If they are equal, it's a HIT, else it's a Miss.**

Advantage
In **1 Search** we know if it is a Hit or a Miss. **Tag Size** = **19 bits**.

Drawback
Since the method is **very rigid**, the **Hit Ratio drops tremendously.**



Set Associative Mapping (Two Way Set Associative Mapping)

Two way Set Associative Mapping technique states:
A block of Main Memory can only be mapped into the same corresponding Block No. of Cache Memory, in any of the two sets.
Since there are two ways of Mapping, its called Two Way Set Associative Mapping.
We treat the entire Cache as **Two Sets**. The Main Memory is divided into Sets, subdivided into Blocks.
A Block of Main Mem. (of any set), can only be mapped into the same Block No. in Cache Memory again of any set. This means, Block 0 of Main Memory (of any set), can only be mapped into Block 0 of Cache Memory, into one of its two sets.
In other words, Block 0 of Cache Memory can only contain Block 0 of Main Memory but of any Set.

Consider Pentium Processor Cache (This is Actually how Pentium's Cache is implemented)

Size of Main Mem:	**4GB** = 2^{32}
Size of Cache Mem:	**8KB** = 2^{13} … this is treated as Two Sets
Hence Size of Set:	**4KB** = 2^{12}
Size of Cache Block (Line):	**32** bytes (words) = 2^{5}
No. of Blocks in a set:	Size of Set (2^{12}) ÷ Size of Block (2^{5}) = 2^{7} = **128**
No. of Sets in Main Mem:	Size of Main Mem (2^{32}) ÷ Size of Set (2^{12}) = 2^{20}
No. of Sets in Cache Mem:	2
Main Mem address:	**32 bits** (because main Mem is of **4GB** = 2^{32})

Tag Size:
Since, Block 0 of Cache Memory can only contain Block 0 of Main Memory but of any Set, the Tag has to only indicate the Set No. of Main Memory, from which the block is present. As Main Memory has 2^{20} sets, the **Tag** size is **20 bits**.

Searches:
If we need Block 0 if Main Memory, we only need to search Block 0 of Cache Memory, but in any of the two sets. Hence, we need **2 searches** in Cache Memory to know if it is a Hit or a Miss.

Method of Searching:
The Processor issues a 32 bit Main Memory address. It can be divided as:

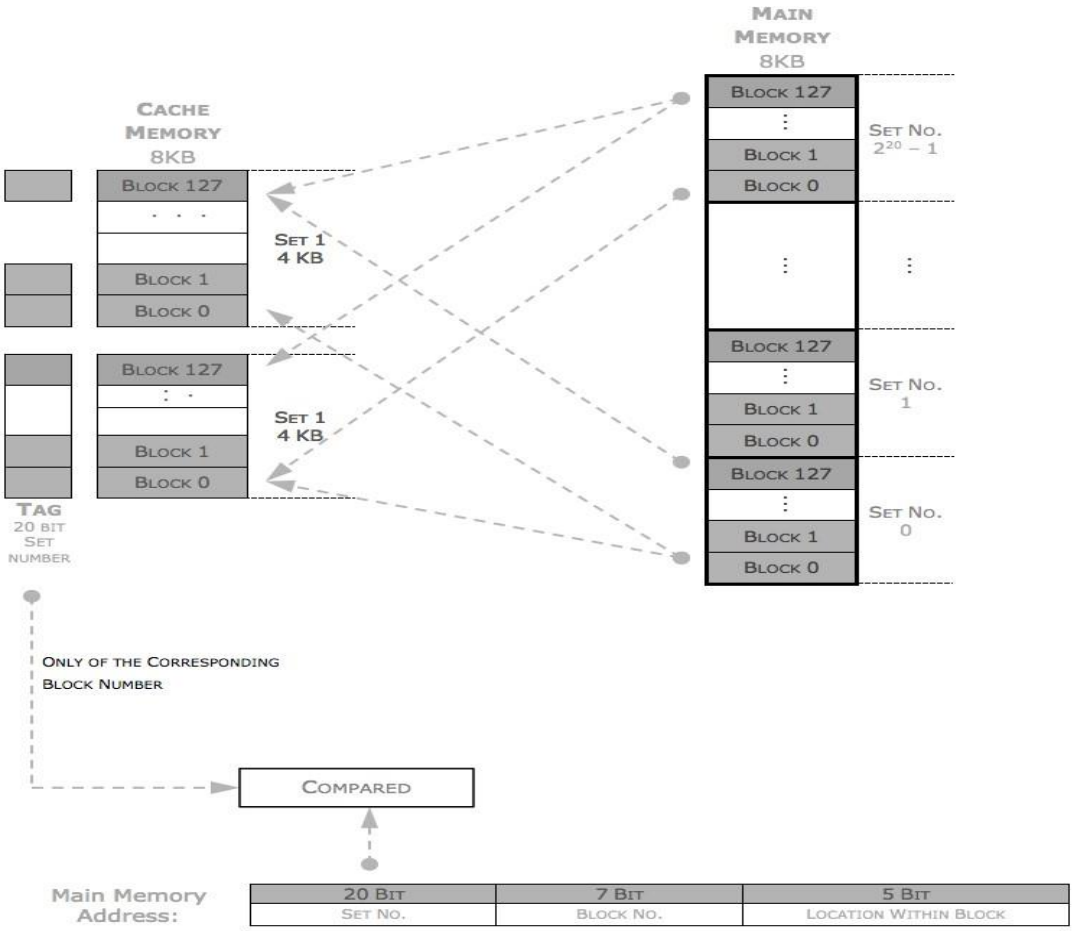
Main Memory Address:

20 BIT	7 BIT	5 BIT
Set No.	Block No.	Location Within Block

First we look at the block no. we need, to know where we need to search.
We then look at the Set No. that we need and compare it with the Tags of the corresponding Block no in the Cache Memory, in any of the two sets.
Assume the Main Memory address is 5:0:6. This means we need location 6 of Block 0 of set 5.
We go to Block 0 of Cache Memory, in both sets.
It has a Tag, which gives the Set No of Main Memory whose Block 0 is present in Cache Memory.
These required Set no. is compared with the two Tags. **If found in any of the 2, it's a HIT, else Miss.**

Advantage
In **2 Searches** we know if it is a Hit or a Miss. **Tag Size** = **20 bits**.

Drawback
Since the method is **flexible**, it significantly **increases** the **Hit Ratio.**



Expanding the logic of set associative cache further, we can derive the following conclusion:

No of Ways	No of Searches	Tag Size	No of Blocks in a Set
2 Way	2	20 bits	128
4 Way	4	21 bits	64
8 Way	8	22 bits	32
16 Way	16	23 bits	16
32 Way	32	24 bits	8
64 Way	64	25 bits	4
128 Way	128	26 bits	2
256 Way	256	27 bits	1
This becomes exactly the same as Fully Associative: 256 Searches, 27-bit Tag			