Cache Mapping

- Direct
- Associative
- Set Associative

Direct Mapping

Suppose MM Size = 64 Byte * Memory is Byte Addressable
Cache Size = 16 Byte

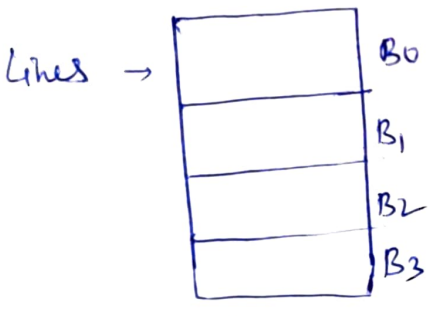
MM is divided into Block (frame). Cache is divided into lines (Block)

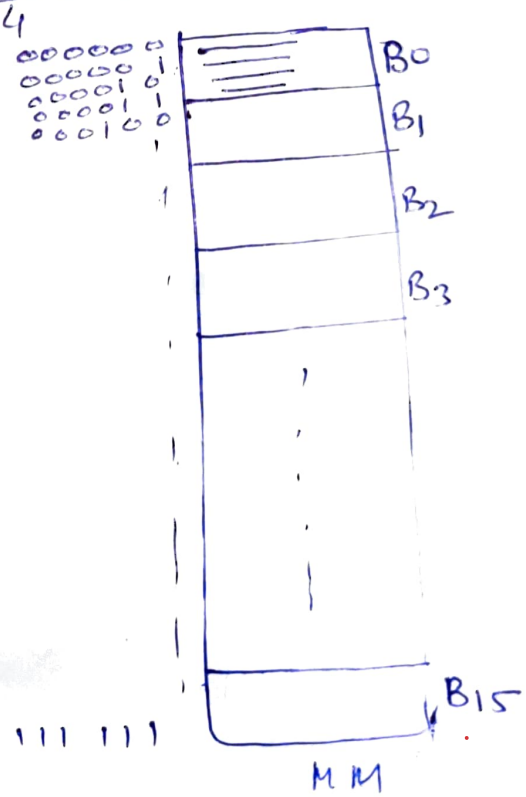
Cache line Size = MM Block Size

Suppose Block Size = 4 byte

No. of Block in MM = $\frac{64}{4}$ = 16 Block

No. of lines in Cache = $\frac{16}{4}$ = 4 lines





Physical Address

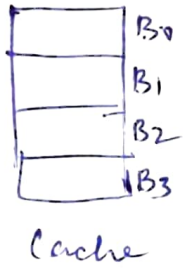
0001	00
Block Number	Block offset

Mapping function

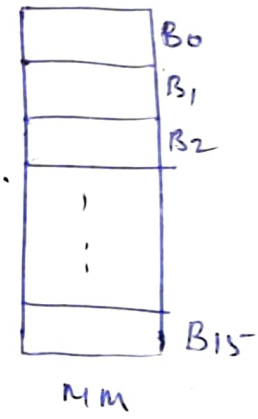
~~Cache Block No. = (Memory Memory Block No) Mod Block~~

Cache Block (lines) No = (MM Block No.) Modulo (~~Cache~~ No. of Block in cache)

So



Cache



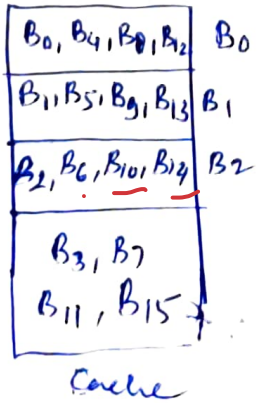
MM

$(MM)_{B_0}$ = : MM Block No. Zero

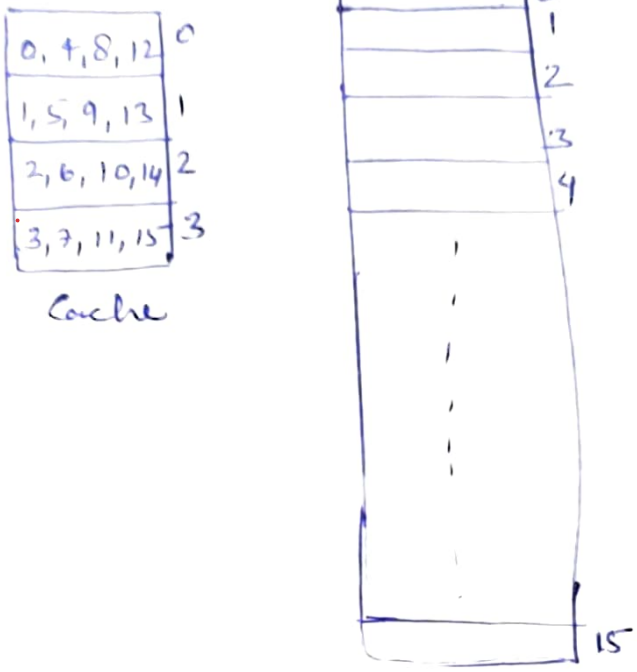
(0) Mod (4) = 0 so it will Map with $(Cache)_{B_0}$

$(MM)_{B_6}$

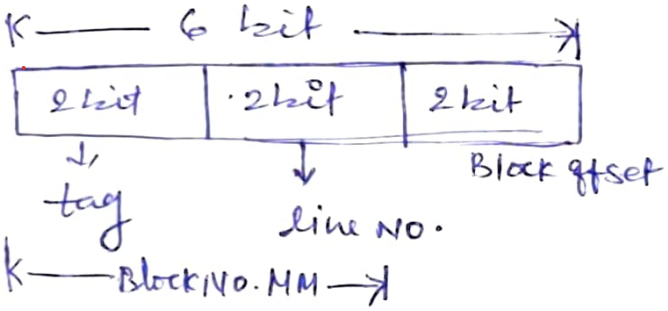
6 Mod 4 = 2 so it will map with $(Cache)_{B_2}$

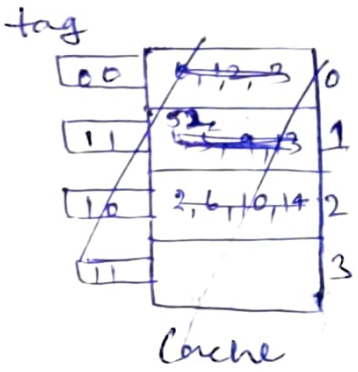


Cache



0	0	0	0 0	→ will go in Cache Block zero
1	0	0	0 1	
2	0	0	1 0	
3	0	0	1 1	
4	0	1	0 0	
5	0	1	0 1	→ will go Cache Block 1
6	0	1	1 0	
7	0	1	1 1	
8	1	0	0 0	
9	1	0	0 1	
10	1	0	1 0	
11	1	0	1 1	
12	1	1	0 0	
13	1	1	0 1	
14	1	1	1 0	
15	1	1	1 1	

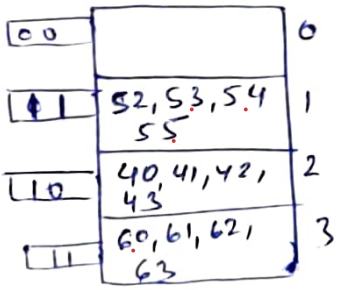




11	01	01	
11	01	00	(52)
11	01	01	(53)
11	01	10	(54)
11	01	11	(55)

11	01	01

Hit (53)



10	10		
10	10	00	40
10	10	01	41
10	10	10	42
10	10	11	43

10	01	11

Miss

11	11		
11	11	01	60
11	11	01	61
11	11	10	62
11	11	11	63

Disadvantage

Cache line	MM block assigned
0	B_0, B_4, B_8, B_{12}
1	B_1, B_5, B_9, B_{13}
2	B_2, B_6, B_{10}, B_{14}
3	B_3, B_7, B_{11}, B_{15}

Suppose B_0 is mapped to line 0
~~the~~ CPU want to Access B_4. we have to Replace B_0 and then only we Can place B_4.
There is free Space in Cache but we Can't use.

So, Cache misses are very high