COMPUTER SCIENCE

Computer Organization and Architecture

Cache Memory

Lecture_05

Vijay Agarwal sir

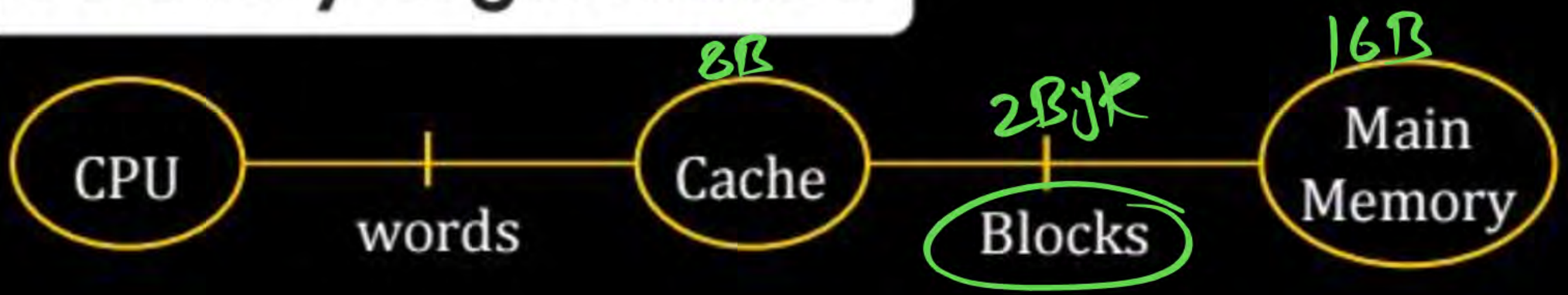
01 Mapping Techniques

Mapping

The Process of transfer the Data from Main memory to Cache memory is called Mapping.

1. Direct Mapping
2. Set Associative Mapping
3. Fully Associative Mapping

Memory Organization



Mapping

The process of transfer the Data from Main Memory to Cache Memory is called mapping. There are 3 Type of Mapping Technique

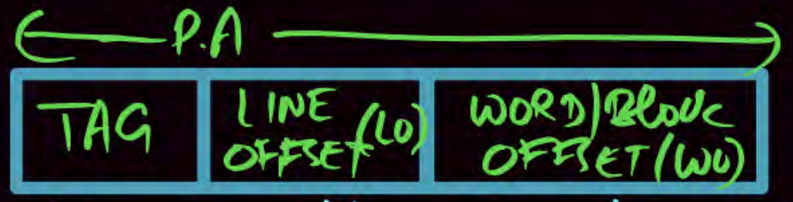
1) Direct Mapping
2) Set Associative Mapping
3) Fully Associative Mapping

Mapping Function

- Because there are fewer cache lines than main memory blocks, an algorithm, is needed for mapping main memory blocks into cache lines.
- Three techniques can be used:

Direct	Associative	Set Associative
• The simplest technique • Maps each block of main memory into only one possible cache lines.	• permits each main memory block to be loaded into any line of the cache. • The cache control logic interprets a memory address simply as a Tag a word field • To determine whether a block is in the cache, the cache control logic must simultaneously examine every line's Tag for a match	• A compromise that exhibits the strength of both the direct and associative approaches while reducing their disadvantage.

Direct Mapping : Cache Controller interpret the Physical address as



TAG = MM Size / CM Size

Tag bit = $\lceil \log_2 \#Tag \rceil$

(OR)

Tag bit = P.A − [LO + WO]

How Many Number of Main Memory Block Possible (Fighting) for 1 Cache Line But Only One Block can present at a time.

Number of CM Blocks / CM Lines

↳ # Address in a Block.

#LINES = CM Size / Block size

LO = $\log_2$ # Lines

Direct Mapping :

Mapping function : $k \bmod N = i$

k : MM Block No

N : # CM Lines

i : CM Line No [0 to N-1]

OR

Cache Address	=	MM Block No (Request)	MOD / %	Number of CM Lines.

Direct Mapping

In this Direct Cache Controller interprets the CPU generated Request as follows:

$$\text{Word Offset} = \log_2(\text{Block Size})$$

$$\#\,\text{LINE} = \frac{\text{CM Size}}{\text{BLOCK Size}}$$

$$\text{LINE Offset} = \log_2(\#\text{LINE})$$

$$\text{TAG} = \text{Physical Address} - (\text{Line offset} + \text{Word offset})$$

$$\text{TAG Memory Size} = \#\text{LINE's} \times \text{Tag bits}\ (\text{Depend on the mapping technique})$$

1) Direct Mapping

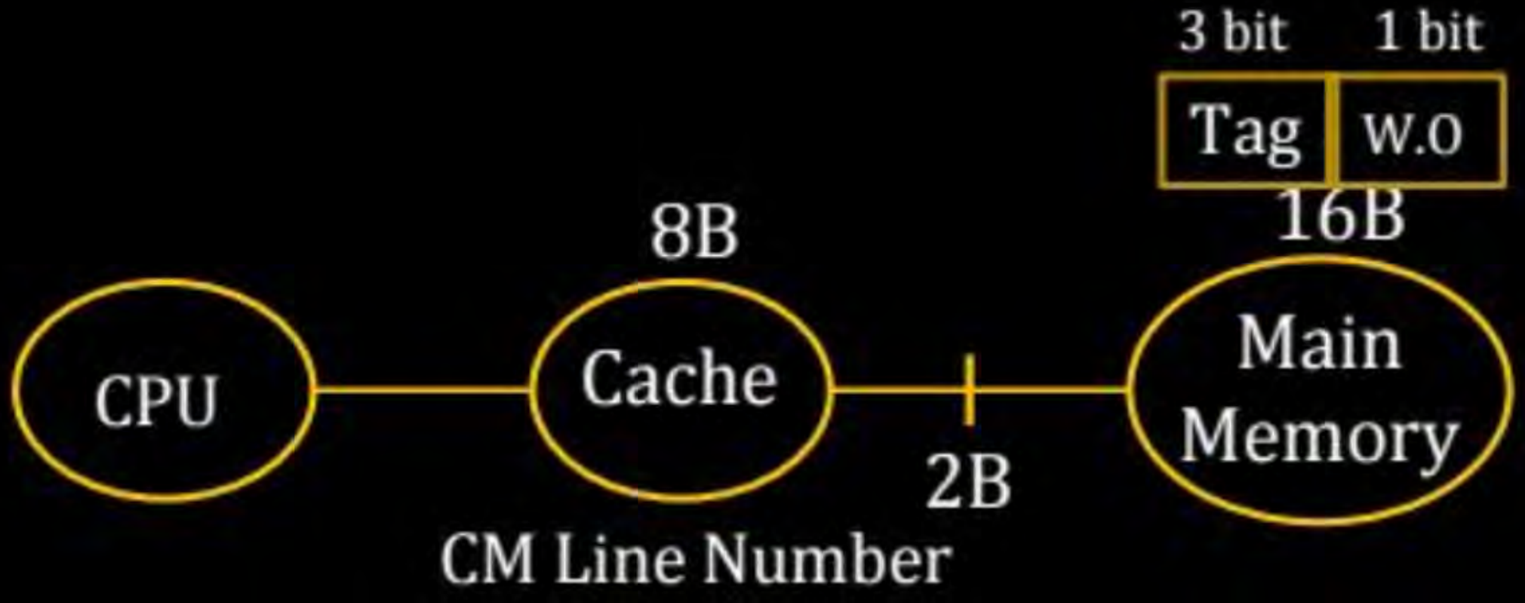
In this Technique mapping function is used to transfer the data from Main Memory to Cache Memory. The Mapping Function is

Cache address = Main Memory request MOD # CM LINES

(Or)

K MOD N = i

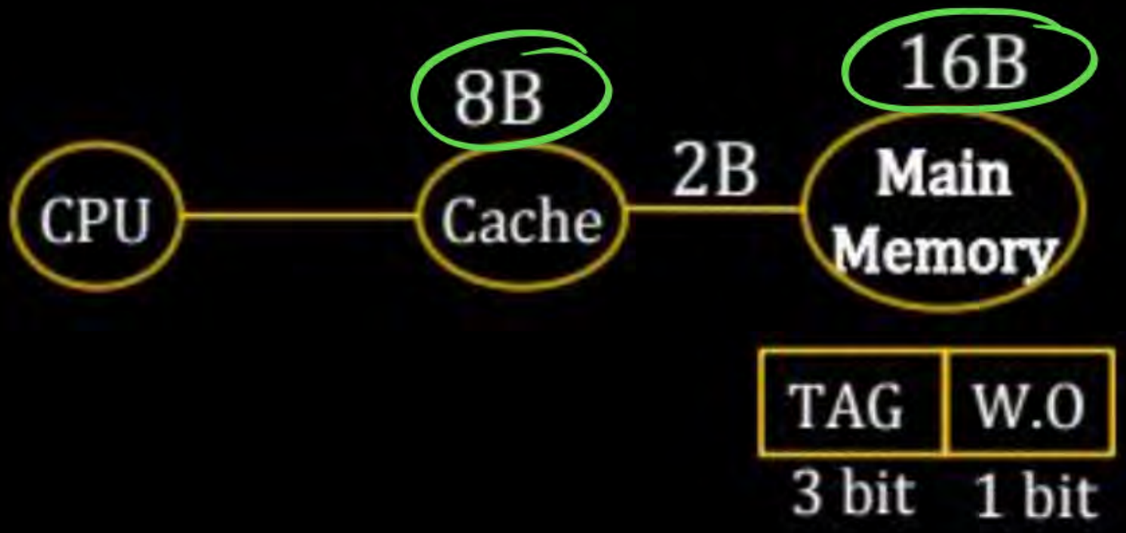
K: MM Block No.
N: # of Cache Line
i: CM Line Number

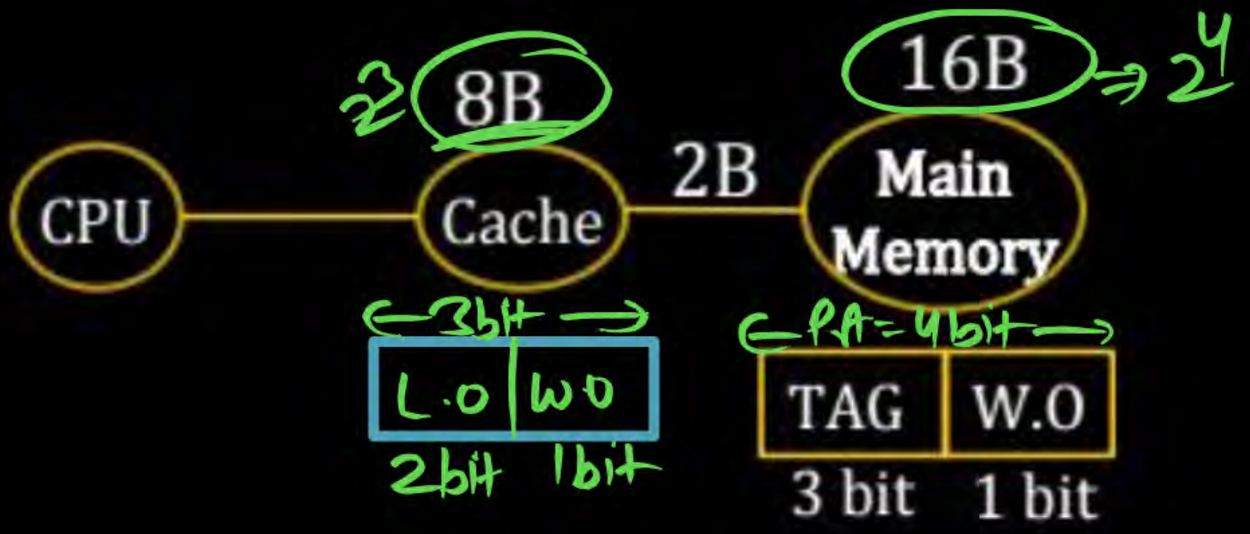


0 MOD 4 = 0
1 MOD 4 = 1
2 MOD 4 = 2
3 MOD 4 = 3
4 MOD 4 = 0
5 MOD 4 = 1
6 MOD 4 = 2
7 MOD 4 = 3

CM

B_0 & B_4: LINE 0
B_4 & B_5: LINE 1
B_2 & B_6: LINE 2
B_3 & B_7: LINE 3


8B
16B
CPU
Cache
2B
Main Memory
TAG
W.O
3 bit
1 bit

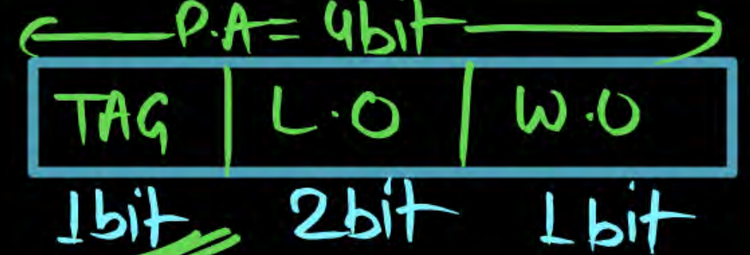


$\#LINE = \frac{CM\,size}{BC} = \frac{8B}{2B} = 4\ Lines$

OR

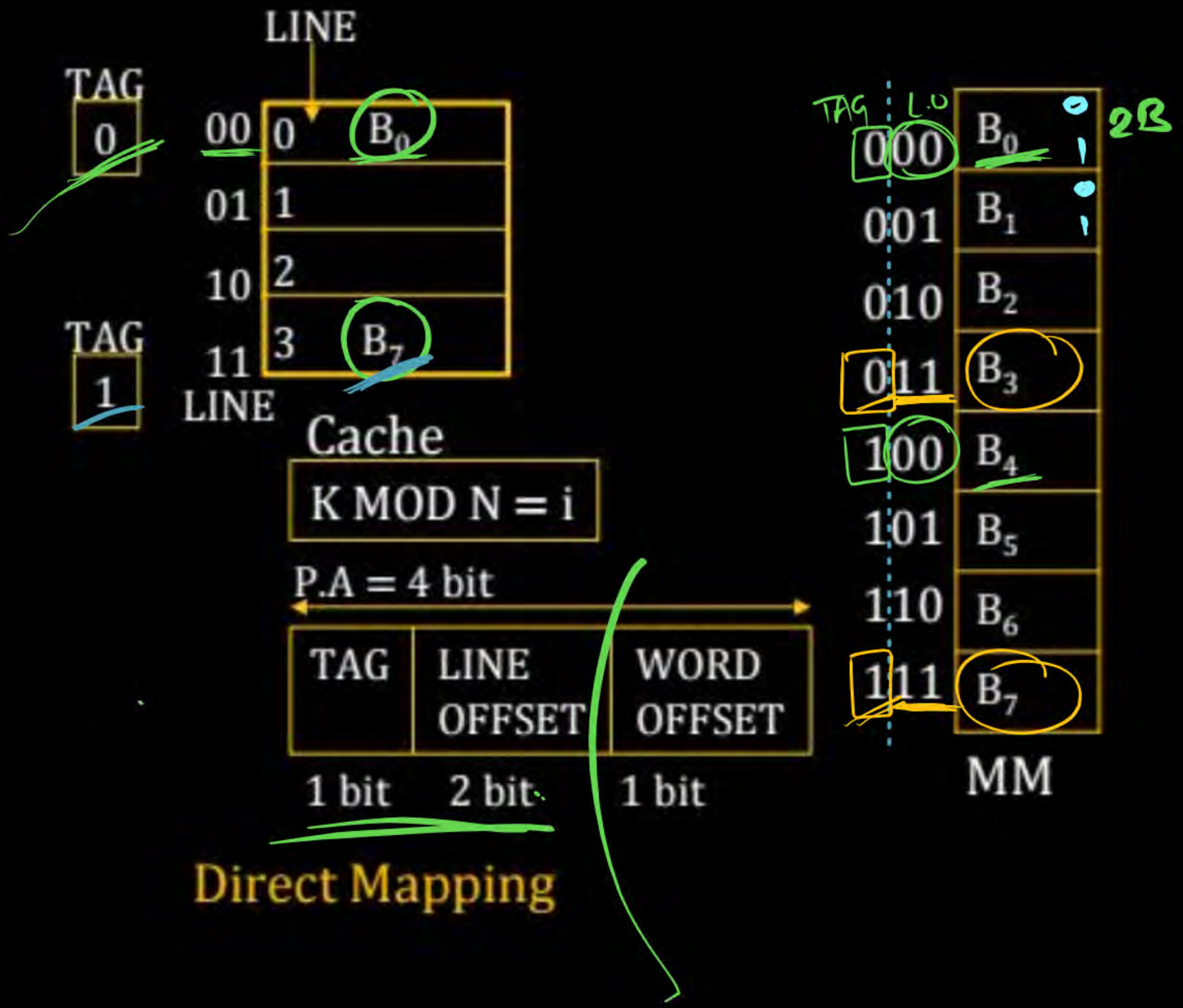
L.O = 2bit

$\#LINE = 2^2 = 4\ Lines$



$K \bmod 4 = i$

TAG = 1 bit $\Rightarrow 2^1 \Rightarrow$ 2 MM Blocks fighting for 1 CM Lines But Only One can Present at a time.


LINE
TAG
0
00
0
B_0
01
1
10
2
TAG
1
11
3
B_7
LINE
Cache
K MOD N = i
P.A = 4 bit
TAG
LINE OFFSET
WORD OFFSET
1 bit
2 bit
1 bit
Direct Mapping

TAG
L.O
000
B_0
001
B_1
010
B_2
011
B_3
100
B_4
101
B_5
110
B_6
111
B_7
2B
MM

MM Block | Direct Mapping | CM LINE

TAG	LINE
0	00

$B_{0_{[000]}}$

K MOD N = i

0 MOD 4 = '0'

LINE '0'

TAG	LINE
1	11

$B_{7_{[111]}}$

K MOD N = i

7 MOD 4 = '3'

LINE '3'

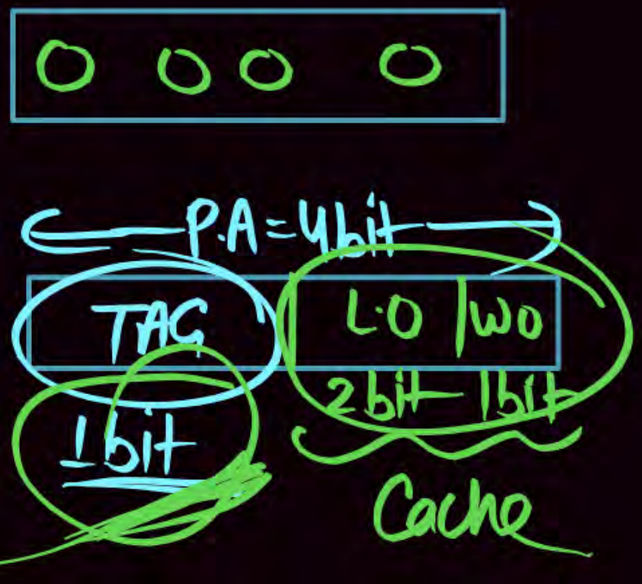
Tag Memory Size = #LINE's × Tag bits [Depends On the Mapping technique]

In the above example: # LINE = 4
Tag bit = 1 bit
(Direct Mapping)

Tag Memory Size = 4 × 1 = 4 bits

(Physical Addel)
→ MM Req.

CPU ⇒ Cache



Tag Memory = #LINES × Tag bit (Tag entry)

⇒ 4 × 1 bit

= 4bits Ans

Q How to check Cache Hit or Cache Miss ?

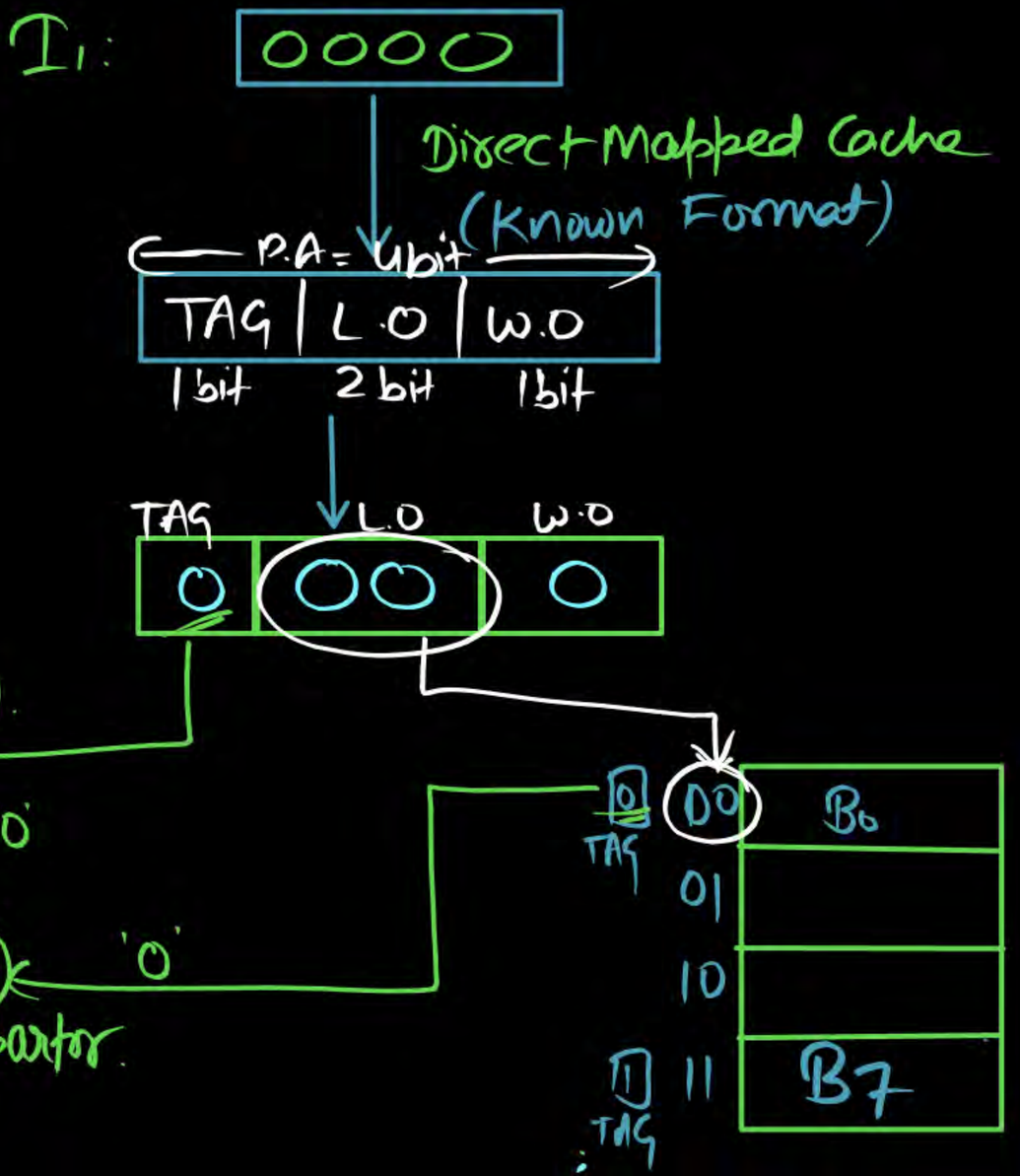
Consider the following program

I_1: MOV r_0 [0000]

I_2: MOV r_1 [1000]

I_3: ADD $r_0 r_1$

CPU generated Request bit by bit refer to the Cache



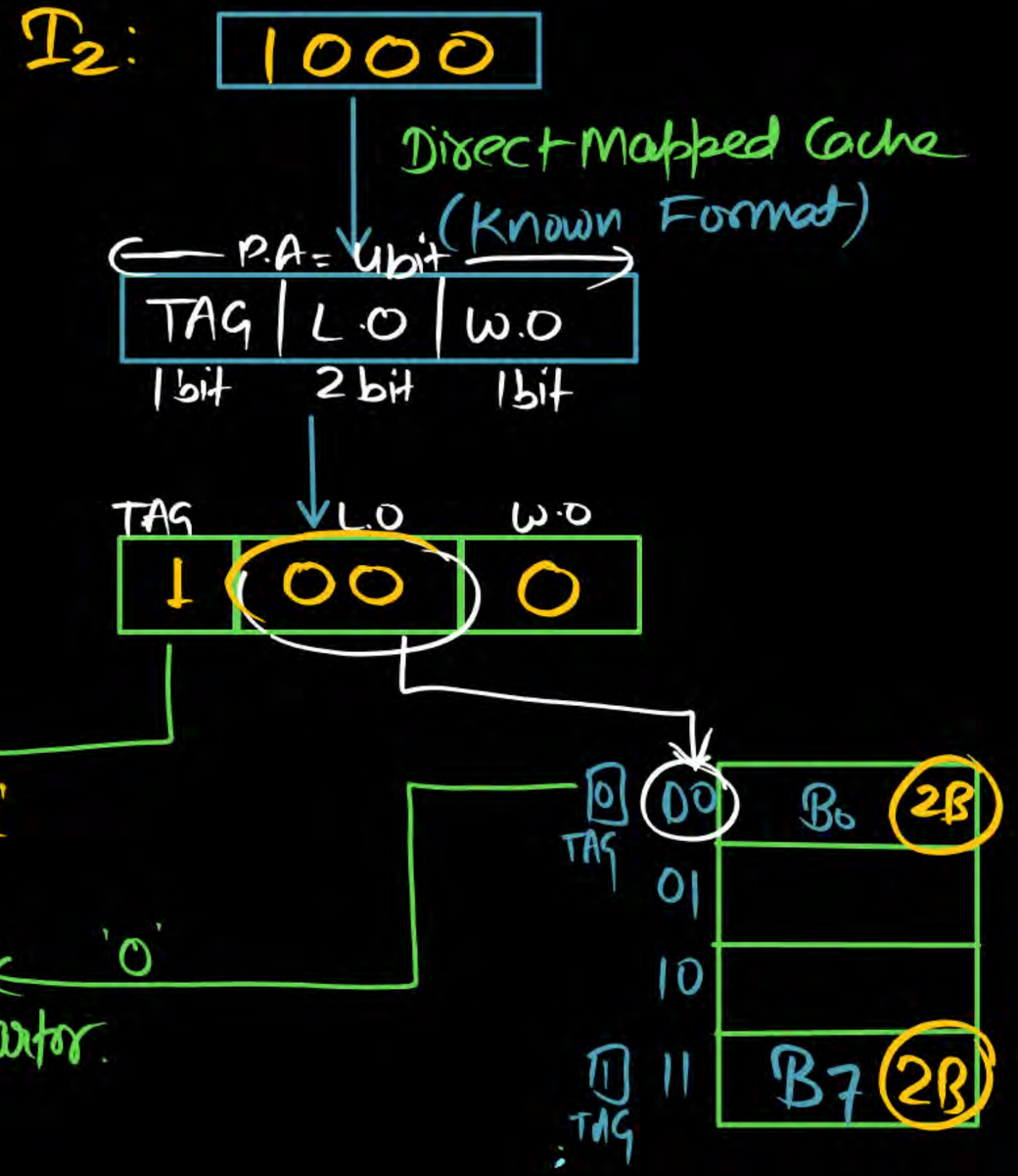
Consider the following program

I_1: MOV r_0 [0000]

I_2: MOV r_1 [1000]

I_3: ADD $r_0 r_1$

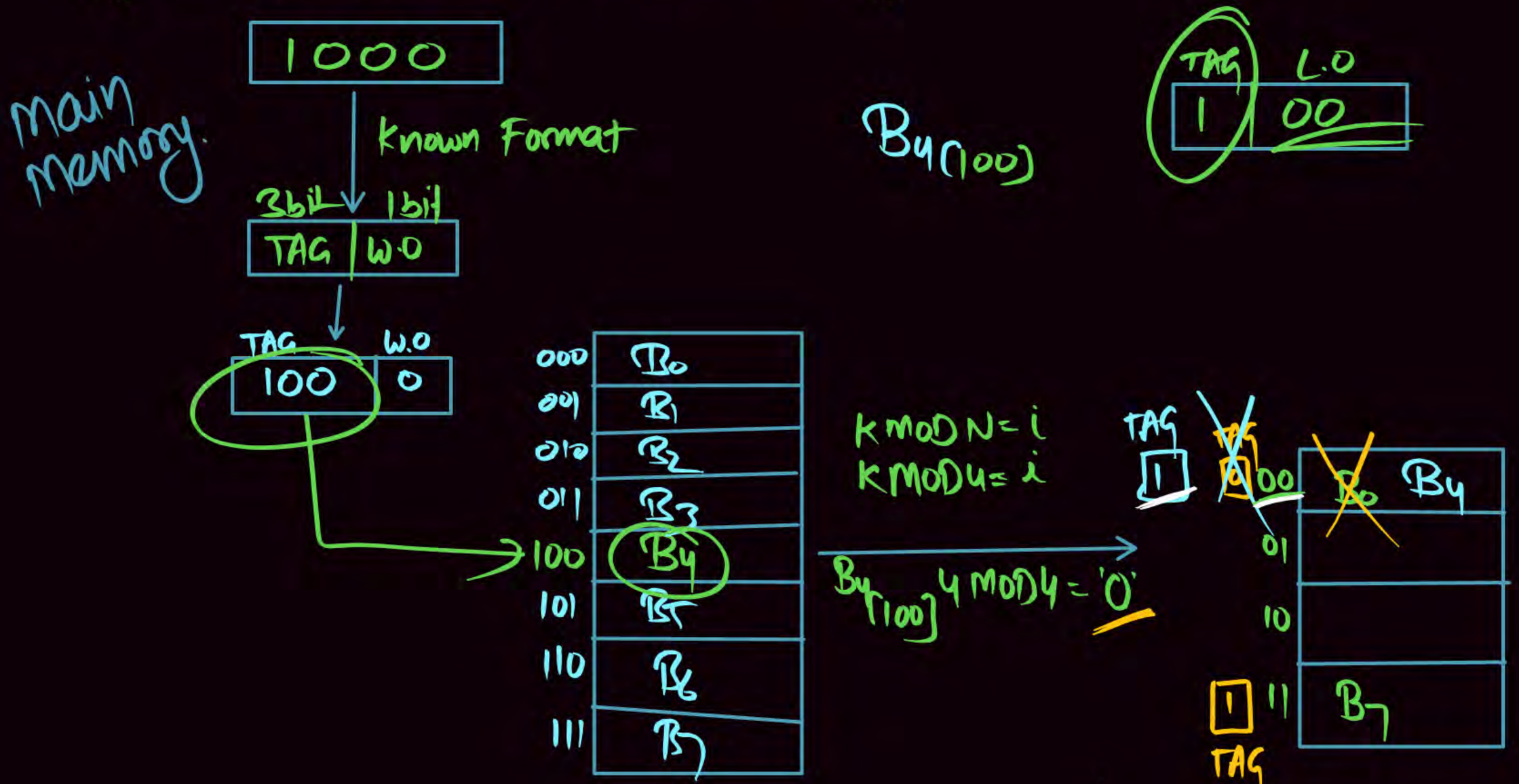
CPU generated Request bits to refer to the Cache



Cache Miss

So Reference forwarded to Main Memory.

Reference forwarded to main memory.



Consider the following program

I_1: MOV r_0 [0000]

I_2: MOV r_1 [1000]

I_3: ADD $r_0 r_1$

Complete working :

- CPU generated Request firstly Refer to the Cache Memory.
- (Direct Mapped Cache) Cache Controller interpreted the Physical Address as :

- Line offset field is Directly Connected to the Address Logic of Cache Memory so Respective Cache Line will be enabled.
- The existing Tag in the enabled Cache Line is Compared with the CPU generated Tag with the Help of Tag Comparator.

Complete working :

- If both Tag [CPU generated Tag & existing Tag on Enabled Cache Line Tag] are Matching then it is Called Cache Hit. [operation Hit (ie Data Present in the Cache]

Note

So Based on Word offset [nth Byte of Block @ 1st Byte of Block] Respective Data Respective Word or Byte is transfered from Cache to CPU.

- If Both Tag are Not Matching then operation Called Cache Miss, then Reference forwarded to the Main Memory.
- According to MM Format. Respective MM Block Enabled & Transfer [copying] from MM to CM With the Help of Direct Mapping function [K MOD N] into Particular Cache Line. the Cache to CPU.

Complete working :

Here Cache Controller Maintain Some extra bits for each Cache Line & that Stored in Tag Directory/memory.

$$\text{Tag Memory} = \text{Number of CM Lines} \times \text{Tag bits}$$

Direct Mapping

Number of Comparator = 1

Size of Comparator = Tag bits.

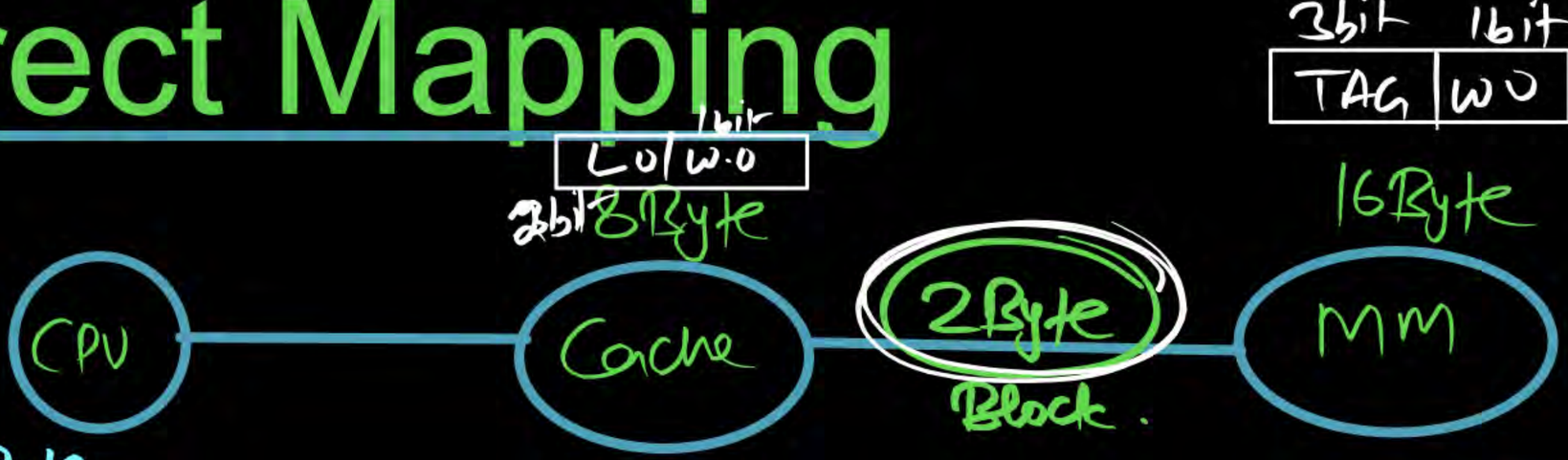
Direct Mapping

Note In a Mapping Process Complete 1 MM Block is transfered from MM to Cache Memory.

Ex 2B, 4B, 64B, 64 Word

Note But Respective Word/Byte of that Particular Block (Based On Word offset) which is demanded (Requested) by the CPU is given from Cache to CPU.

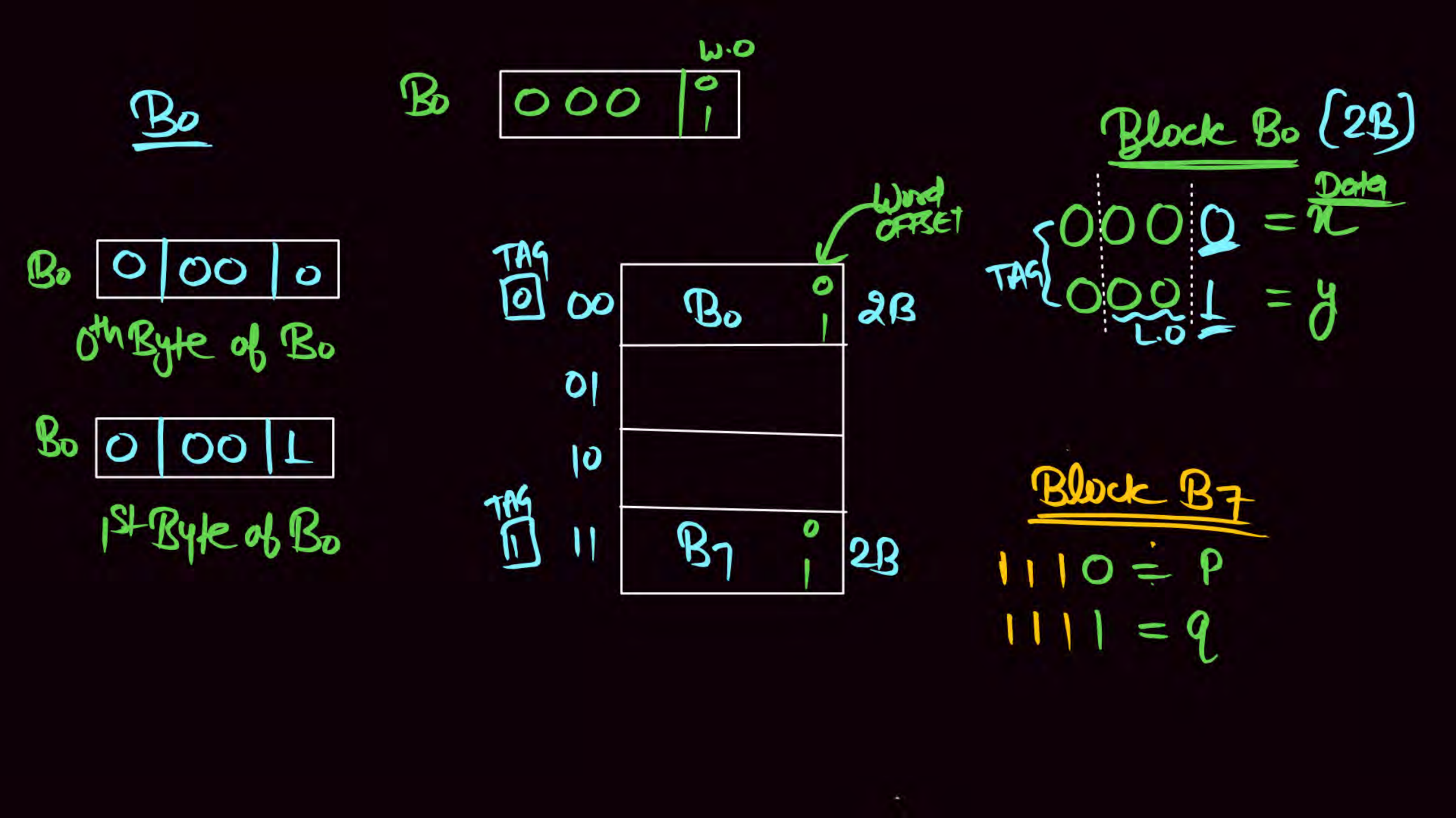
Direct Mapping



Block : 2Byte

000 0 : 0th Byte of B0

000 1 : 1st Byte of B0


B0
B0
000
W.O
0
1
Block B0 (2B)
Data
TAG
0000 = x
0001 = y
L.O
B0
0
00
0
0th Byte of B0
B0
0
00
1
1st Byte of B0
Word OFFSET
TAG
0
00
B0
0
1
2B
01
10
TAG
11
11
B7
0
1
2B
Block B7
1110 = p
1111 = q

B4

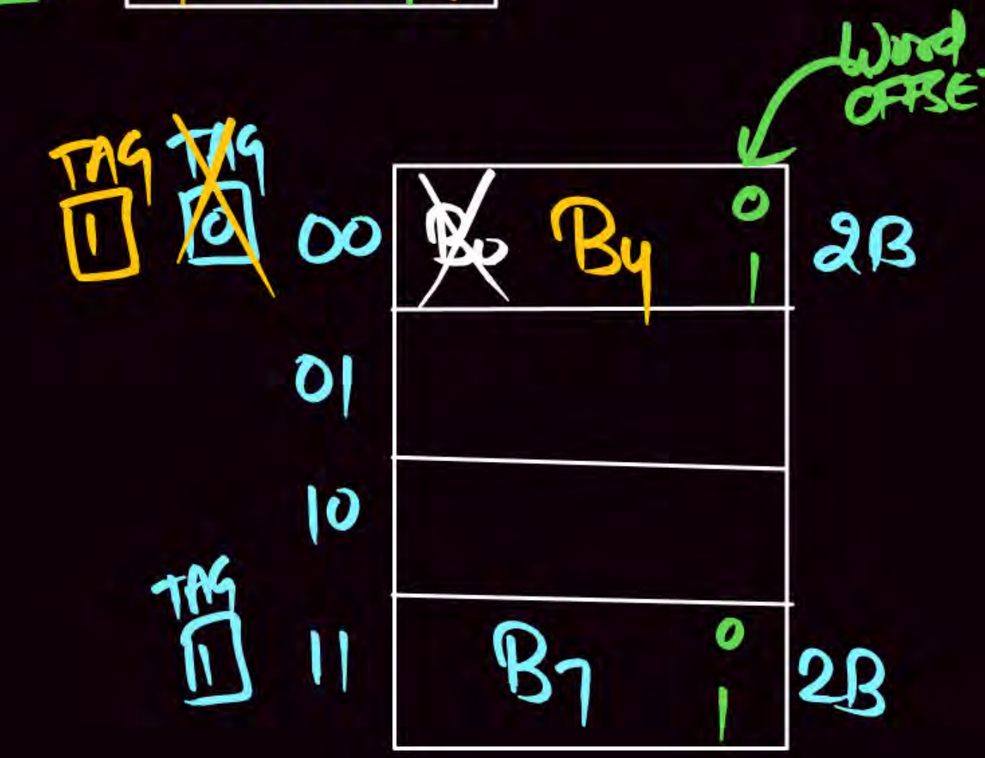
B4: | 1 | 00 | 0 |

0th Byte of B4

B4 | 1 | 00 | 1 |

1st Byte of B4

B4: | 1 | 00 | 0 | W.O.



Block B0 (2B)

TAG { 0000 = Data x

0001 = y

L.O

Block B7

1110 = p

1111 = q

LOR [Locality of Reference]

0000 = x value

0001 = y value

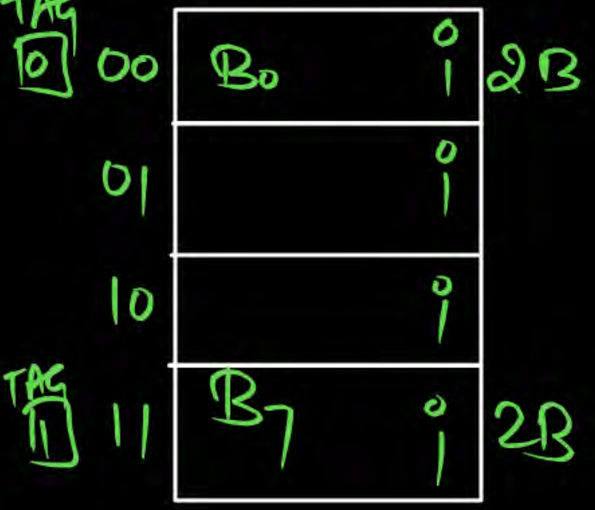
Consider the following program

I_1: MOV r_0 [0000] 0th Byte of B_0

I_2: MOV r_1 [0001] 1st Byte of B_0

I_3: MOV r_2 [1000] 0th Byte of B_4

I_4: MOV r_3 [1001] 1st Byte of B_4



I_1 : [0000] : Block B_0 (0th Byte) : Present in Cache : Cache Hit

I_2 : [0001] : Block B_0 [Cache Hit] ⇒ Present in Cache.

Due to Locality of Reference

LOR [Locality of Reference]

0000 = x value
0001 = y value

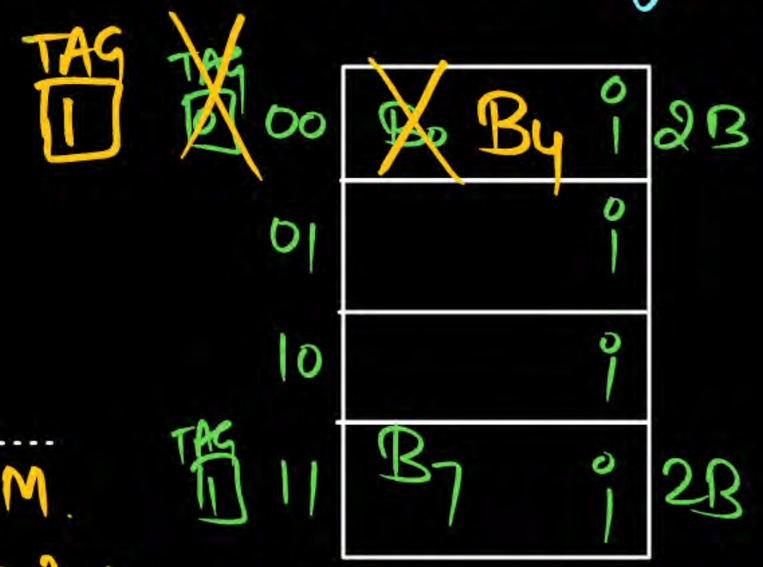
Consider the following program

I_1: MOV r_0 [0000] 0th Byte of B_0

I_2: MOV r_1 [0001] 1st Byte of B_0

I_3: MOV r_2 [1000] 0th Byte of B_4

I_4: MOV r_3 [1001] 1st Byte of B_4



I_3: [1000] ⇒ B_4 ⇒ Cache Miss to Bring B_4 From MM to CM.
↳ then 0th Byte of given to CPU [MM to cache [B_4] then Cache to CPU 0th Byte of B_4].

I_4: [1001]: B_4 (1st Byte of B_4]: Cache Hit
Due to Locality of Reference

Disadvantage of Direct Mapping:

- In Direct Mapping each Cache Line is Able to Hold only (1 Tag) 1 Main Memory Block at a time so Number of Conflict Miss increase.
- If CPU is Refering the Multiple blocks which are Mapped into the Same Cache Line Number [eg $B_0, B_4, B_0, B_4, B_0, B_4 \Rightarrow$ Line^{CM} No '0'] then Number of Conflict Miss [Miss] will be very High Even other Cache Line are Empty. [Thrashing].

∴

Disadvantage of Direct Mapping

I_1: MOV r_0 [0000] 0th Byte of Block B_0

I_2: MOV r_1 [1000] 0th Byte of Block B_4

I_3: MOV r_2 [0001] 1st Byte of Block B_0

I_4: MOV r_3 [1001] 1st Byte of Block B_4

I_5: MOV r_4 [0000] 0th Byte of Block B_0

I_6: MOV r_5 [1000] 0th Byte of Block B_4

Set Associative Mapping

Set Associative Mapping

$$\#LINE = \frac{CM\ Size}{Block\ size}$$

$$\#SET = \frac{\#LINES}{N\text{-}Way}$$

TAG = PA - [S.O + W.O]

$$S.O = \lceil \log_2 \#SET \rceil$$

Mapping function.

$$K \bmod S = i$$

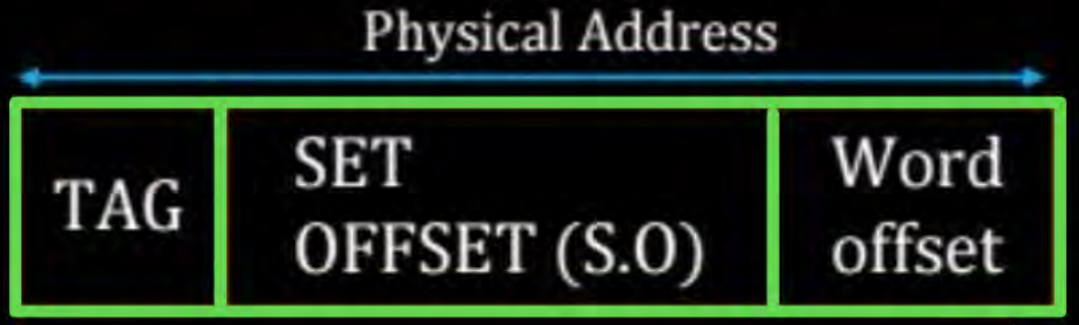
K: MM Block [Request] No.

S: Number of Cache set

i: Cache Set Number [0 to S-1].

2) Set Associative Cache

SET associative cache controller, Interpreter the CPU generated request as follows:



$$\text{Word Offset} = \log_2 \text{Block Size}$$

$$\#\text{lines} = \frac{\text{CM Size}}{\text{Block Size}}$$

$$\#\text{SETS} = \frac{\#\text{Lines}}{\text{N-way}}$$

$$\text{SET OFFSET} = \log_2 \#\text{SETS}$$

$$\text{TAG} = \text{Physical address} - (\text{S.O} + \text{W.O})$$

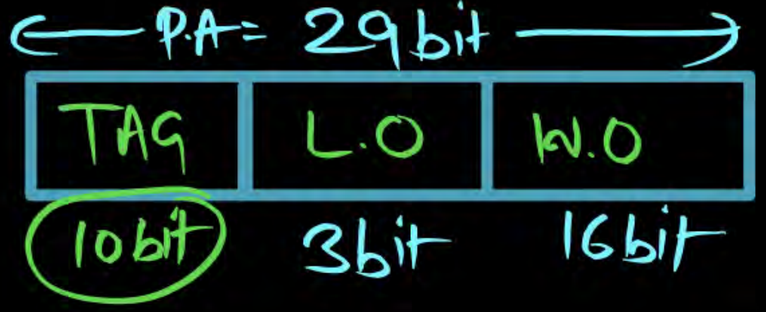
Numerical

5PM

Q.1 Consider a Direct Mapped Cache if the size of cache memory is 512KB & Main Memory 512MB & Cache line size is 64KB then calculate the Number of bit Required for

1. (i) P.A (29) (ii) TAG (10) (iii) L.O (3) (iv) W.O (16)
(2) #lines (8) (3) TAG memory size

Sol



$P.A = 512MB = 2^{29}B \Rightarrow P.A = 29\,bit$

$Line\ Size = 64KB = 2^{16}B \Rightarrow W.O = 16\,bit$

$\#LINES = \frac{CM\ Size}{Block\ Size} = \frac{512KB}{64KB} = \frac{2^{19}B}{2^{16}B}$

$\#LINE = 8$

$L.O = 3\,bit$

$TAG = P.A - (L.O + W.O)$

$TAG = 29 - (3+16)$

$= 10\,bit$

Tag Memory Size = #LINE × Tag bit

$\Rightarrow 8 \times 10$

$= 80\,bit$ @ 10 Byte Ans

Q.1 Consider a 2-way set associative if the size of cache memory is 512KB & Main Memory 512MB & Cache line size is 64KB then calculate the Number of bit Required for

1. (i) P.A (ii) TAG (iii) S.O (iv) W.O

(2) #lines (3) TAG memory size

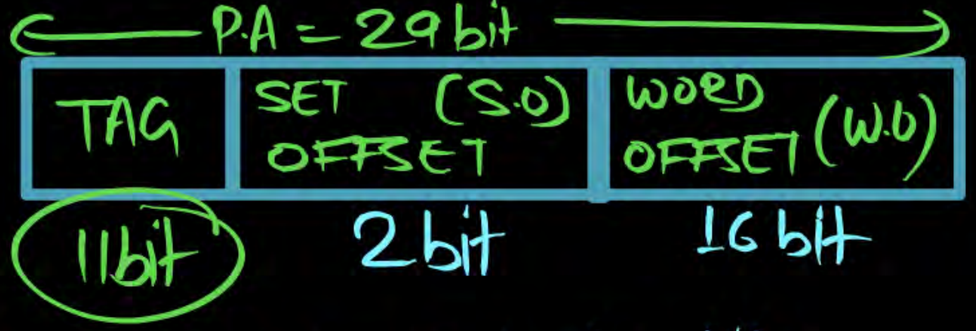
Sol: 2 Way Set Associative.

$$\#LINES = \frac{CM\,Size}{Block\,size} \Rightarrow \frac{512KB}{64KB} = \frac{2^{19}B}{2^{16}B} = 2^3 = 8\,Lines$$

$$\#SET = \frac{\#LINE}{N\text{-}way} = \frac{8}{2} = 4$$

#SET = 4

Set offset = 2 bit



TAG = P.A − [S.O + W.O]

⇒ 29 − (2 + 16)

TAG ⇒ 11 bit

Tag Memory Size = #LINES × Tag bits

⇒ 8 × 11 ⇒ 88 bits (or) 11 Byte

Tag Memory Size = #LINES X Tag bits ⇒ 8 X 11 ⇒ 88 bits

OR

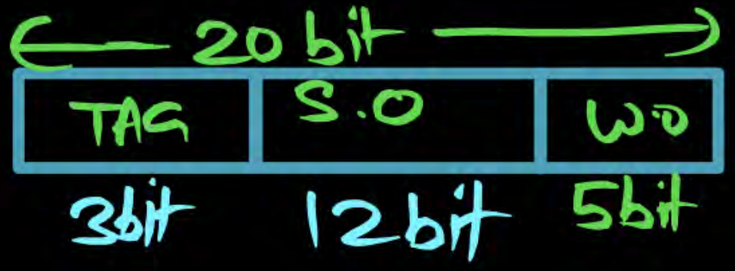
Tag Memory Size = #SET X Block per Set X Tag bits

⇒ 4 X 2 X 11 bit

Tag Memory Size = 88 bits Ans

Q2 Consider a 2-way set associative Cache Size = 256 KB, Line size = 32 Byte, MM = 1MB, then what is the set number of Physical address $(ABCDE)_{16}$?

$$\#LINES = \frac{256KB}{32B} = \frac{2^{18}B}{2^5B} = 2^{13}\ Lines$$

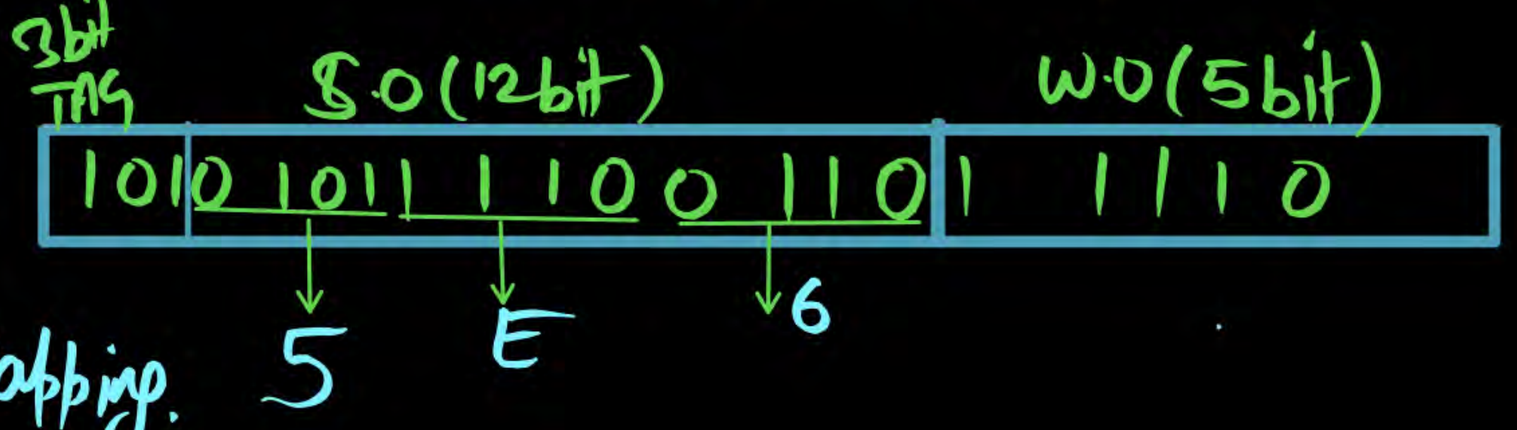


$$\#SET = \frac{\#LINES}{N\text{-}way} = \frac{2^{13}}{2^1} = 2^{12}.$$

S.O = 12 bit

A : 10 : 1010
B : 11 : 1011

Same question $(5E6)_{16}$ Ans
Yesterday already Done With Direct Mapping.



Q. The main memory of a computer has 2 cm blocks while the cache has 2 c blocks. If the cache uses the set associative mapping scheme with 2 blocks per set, then block k of the main memory maps to the set. [GATE - 1999]

(a) (k mod m) of the cache
(b) (k mod c) of the cache
(c) (k mod 2 c) of the cache
(d) (k mod 2 cm) of the cache

Q.

Consider a 4-way set associative cache consisting of 128 lines with a line size of 64 words. The CPU generates a 20-bit address of a word in main memory. The number of bits in the TAG, SET and WORD fields are respectively.

(a) 9, 6, 5 (b) 7, 7, 6 (c) 7, 5, 8 (d) 9, 5, 6

[GATE - 2007]

Q.

[Common Data for this and next question]
Consider a computer with a 4-way set-associative mapped cache of the following characteristics; a total of 1 MB of main memory, a word size of 1 Byte; a block size of 128 words and a cache size of 8 KB.

The number of bits in the TAG, SET and WORD fields, respectively are:

(a) 7, 6, 7 (b) 8, 5, 7 (c) 8, 6, 6 (d) 9, 4, 7

Q.

[Common Data]
Consider a computer with a 4-way set-associative mapped cache of the following characteristics; a total of 1 MB of main memory, a word size of 1 Byte; a block size of 128 words and a cache size of 8 KB.

While accessing the memory location 0C795H by the CPU, the contents of the TAG field of the corresponding cache line is

(a) 000011000 (b) 110001111
(c) 00011000 (d) 110010101 [GATE – 2008]

Q.

A 4-way set-associative cache memory unit with a capacity of 16KB is built using a block size of 8 words. The word length is 32 bits. The size of the physical address space is 4 GB. The number of bits for the TAG filed is

[GATE - 2014]

Q. The size of the physical address space of a processor is 2^P bytes. The word length is 2^w bytes. The capacity of cache memory is 2^N bytes, the size of each cache block is 2^M words. For a k - way set - associative cache memory, the length (in number of bits)of the tag field is

(a) $P - N - \log_2 K$

(b) $P - N + \log_2 K$

(c) $P - N - M - W - \log_2 K$

(d) $P - N - M - W + \log_2 K$

[GATE - 2018]

MCQ

An 8KB direct-mapped write back cache is organized as multiple blocks, each of size 32 bytes. The processor generates 32-bit addresses. The cache controller maintains the tag information for each cache block comprising of the following.

1Valid bit

1Modified bit

Tag Entry = Tag bit + Extra bit

As many bits as the minimum needed to identify the memory block mapped in the cache.

Tag Memory = #LINE x Tag entry.

What is the total size of memory needed at the cache controller to store meta-data (tags) for the cache? [GATE-2011-CS: 2M]

A 4864 bits

B 6144bits

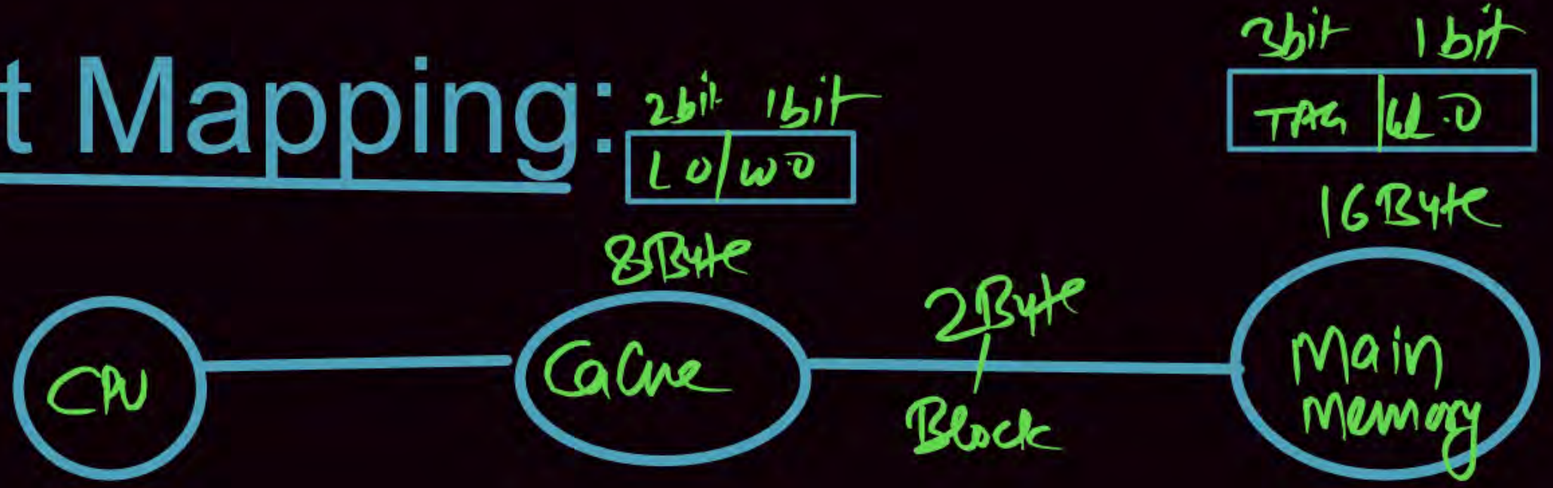
C 6656 bits

D 5376 bits

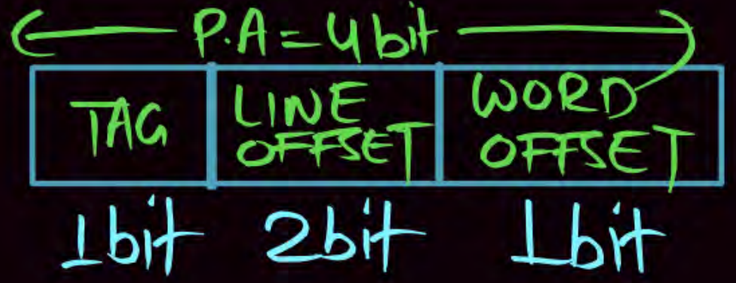
Mapping Process.

1. Direct Mapping
2. SET Associative Mapping

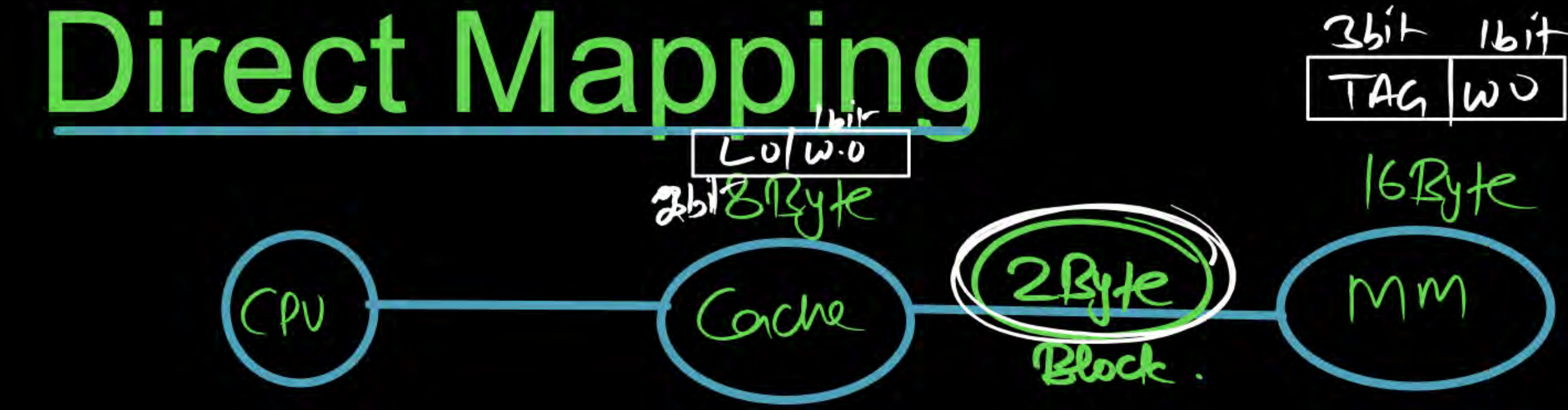
① Direct Mapping:

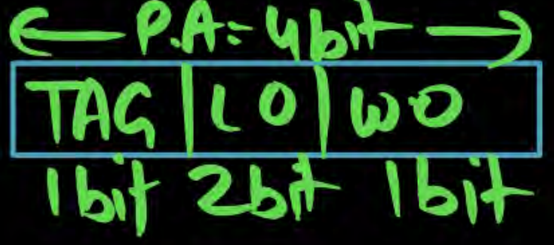


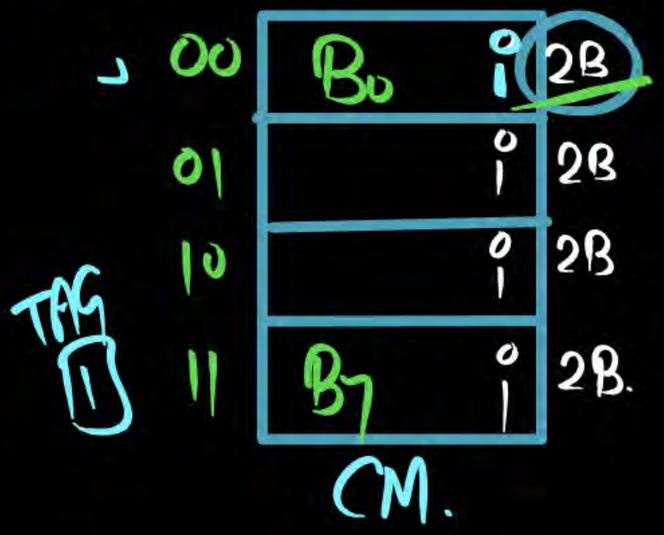
Direct Mapping

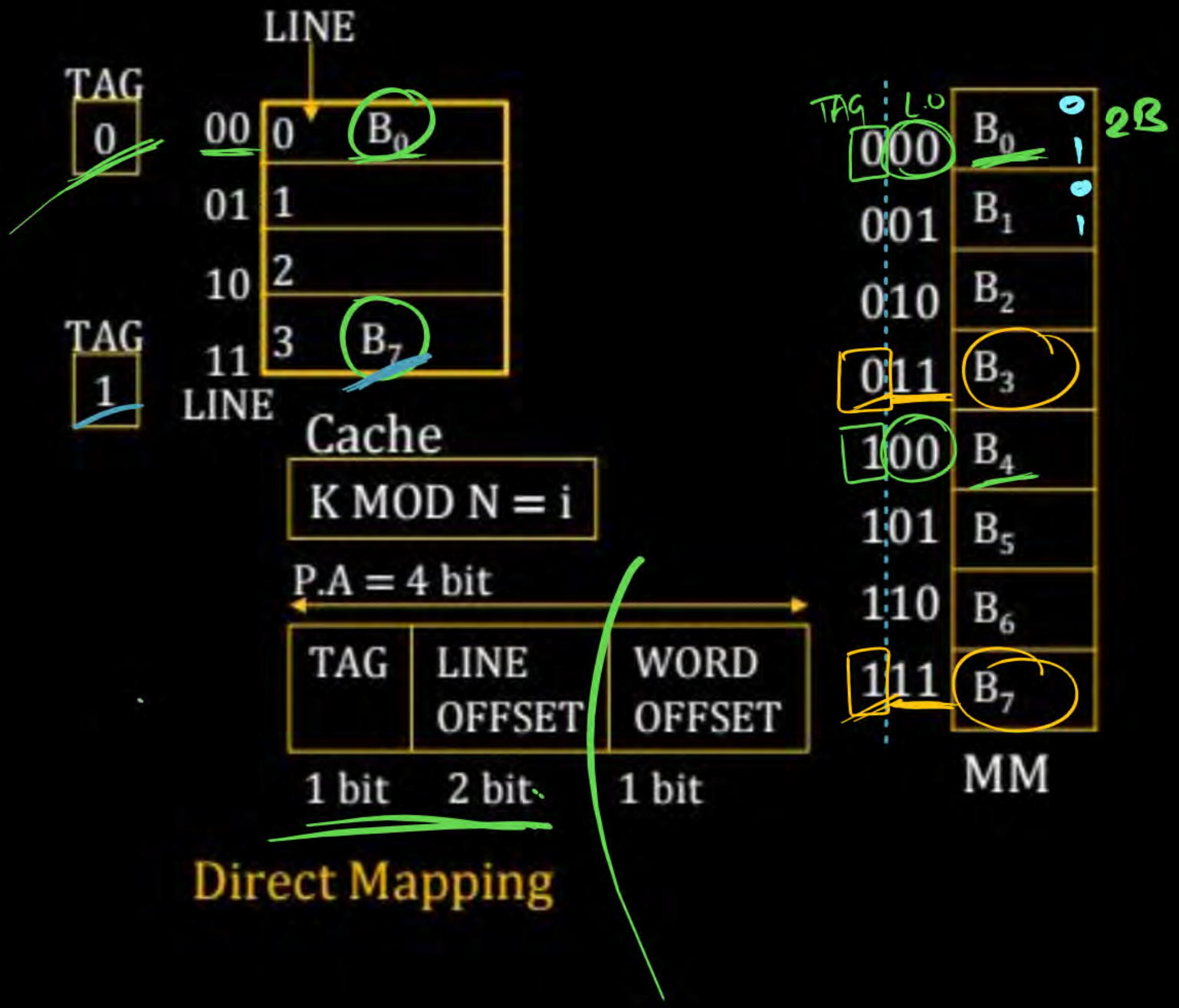


Direct Mapping





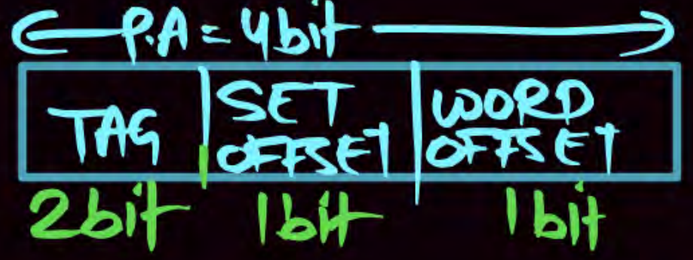



LINE
TAG
0
00
0
B_0
01
1
10
2
TAG
1
11
3
B_7
LINE
Cache
K MOD N = i
P.A = 4 bit
TAG
LINE OFFSET
WORD OFFSET
1 bit
2 bit
1 bit
Direct Mapping
TAG
L.O
000
B_0
001
B_1
010
B_2
011
B_3
100
B_4
101
B_5
110
B_6
111
B_7
2B
MM

K MOD N = i

Q) MM = 16B, CM = 8B, Block Size = 2B, 2 Way Set Associative Mapping then Calculate bits for PA, TAG, S.O, W.O & Tag Memory Size ?

Soln



$$\#LINE = \frac{CM\ Size}{Block\ Size} = \frac{8B}{2B} = 4\ Lines$$

$$\#SET = \frac{\#LINE}{N\text{-}way} = \frac{4}{2} = 2\ Set$$

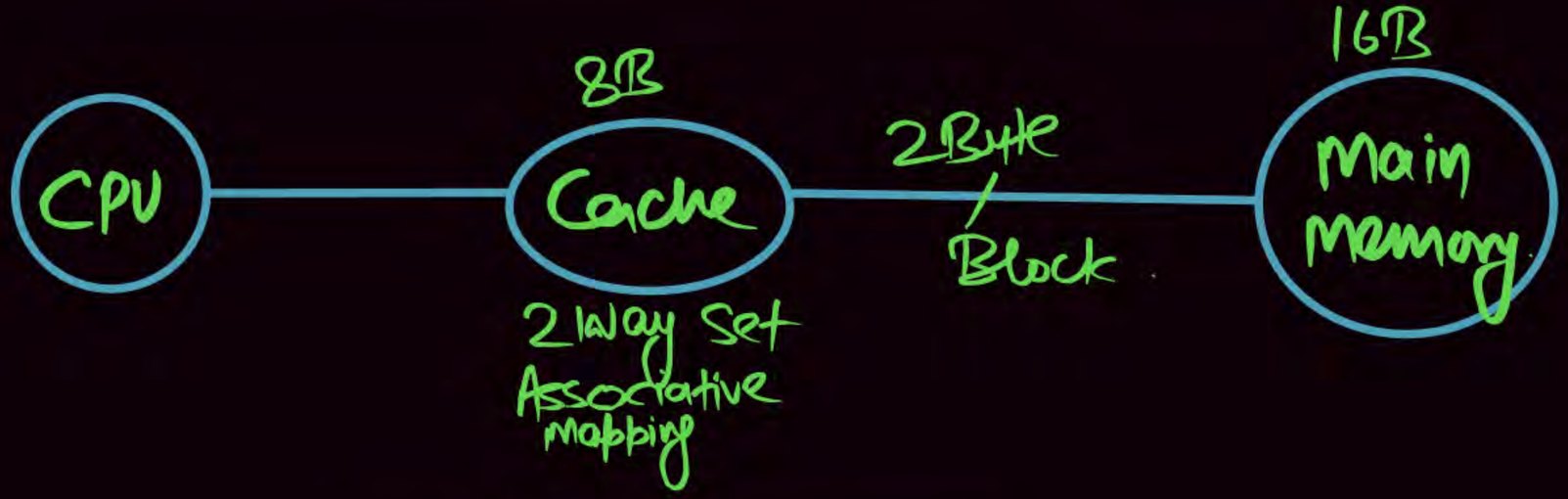
Set offset = 1 bit

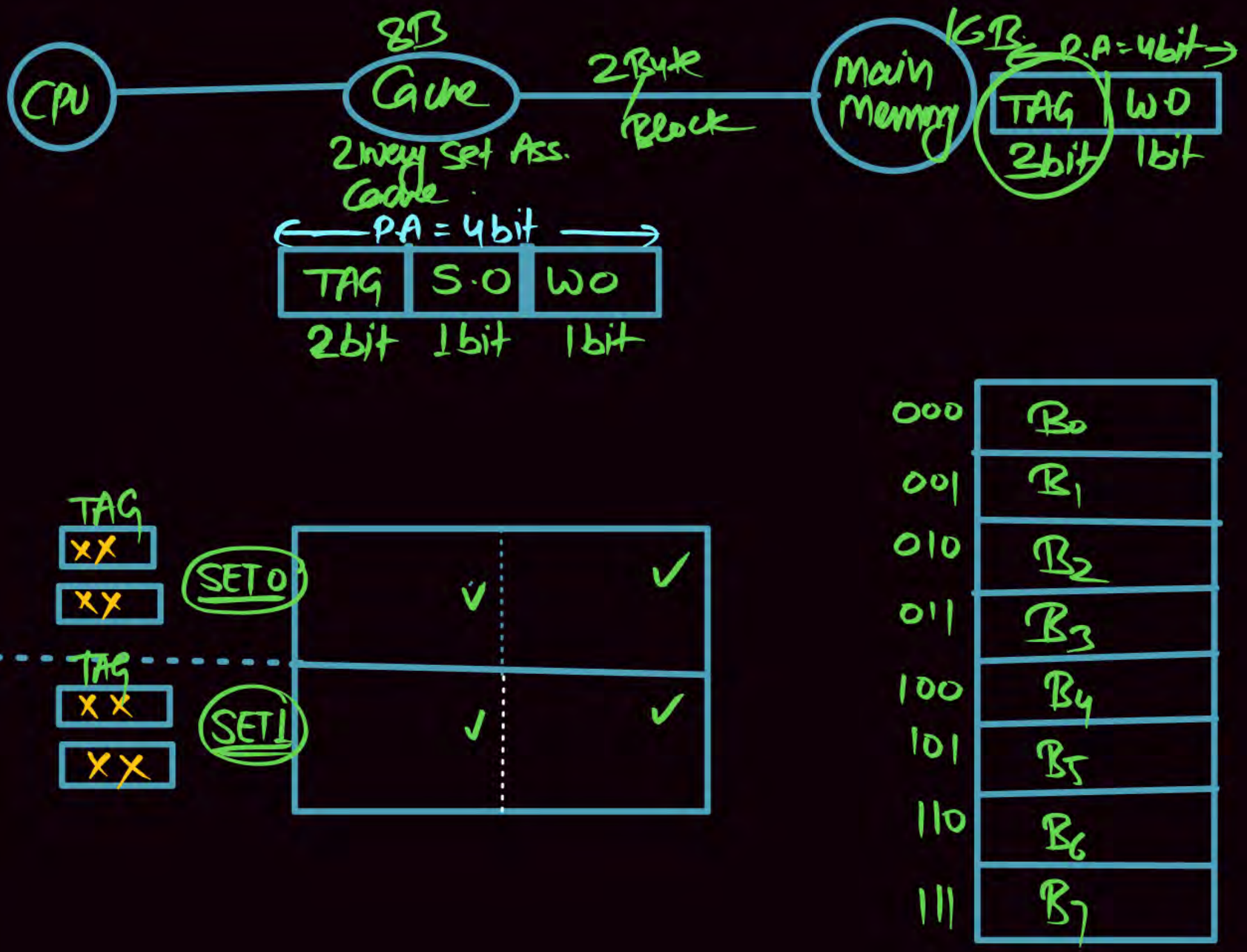
Tag Memory Size = #LINES × Tag bit

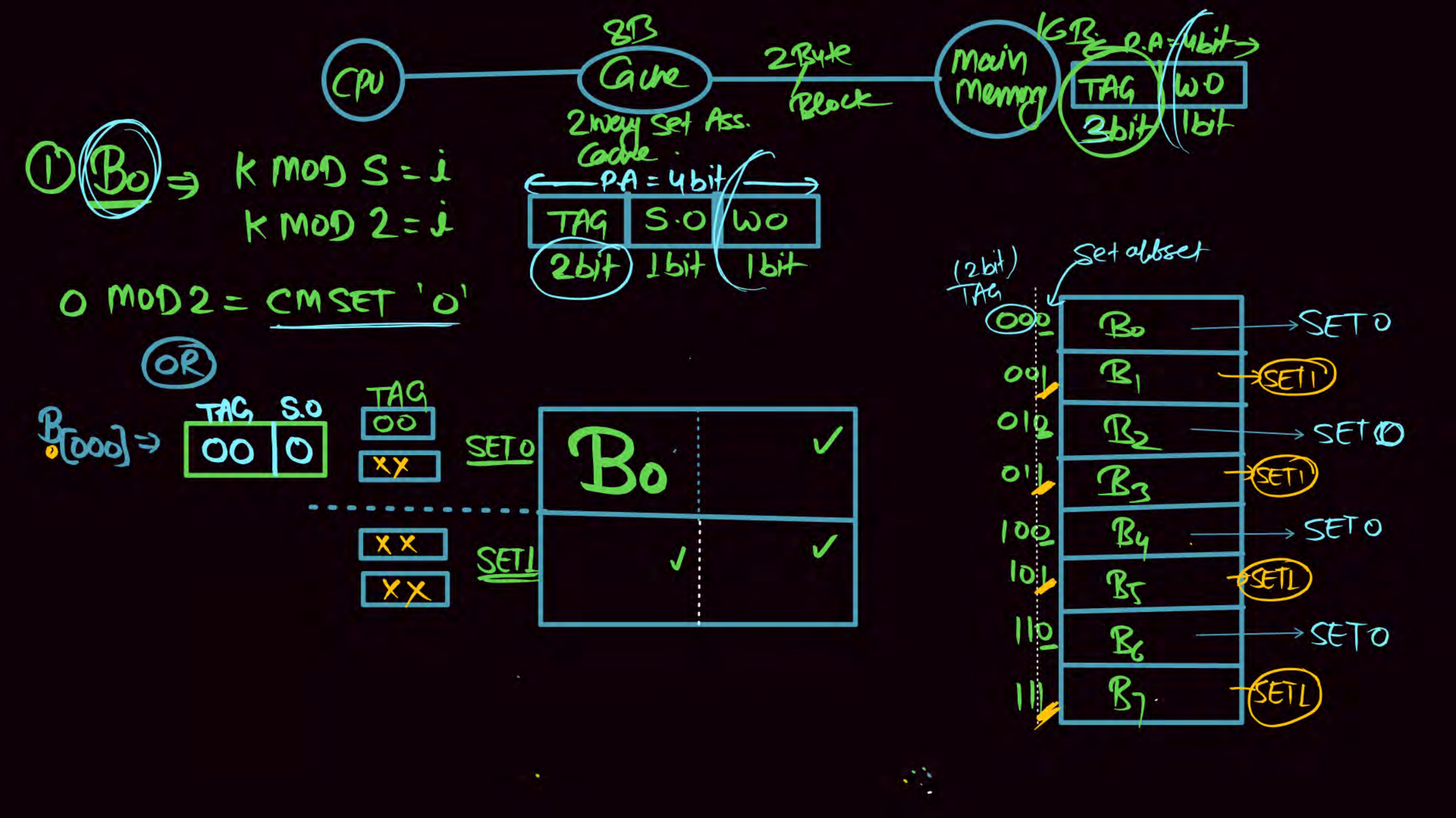
⇒ 4 × 2 = 8bit Ans

Here Tag: 2bit ⇒ ie 2^2 ⇒ 4 MM Blocks are fighting (Mapped) Any One Cache Set.

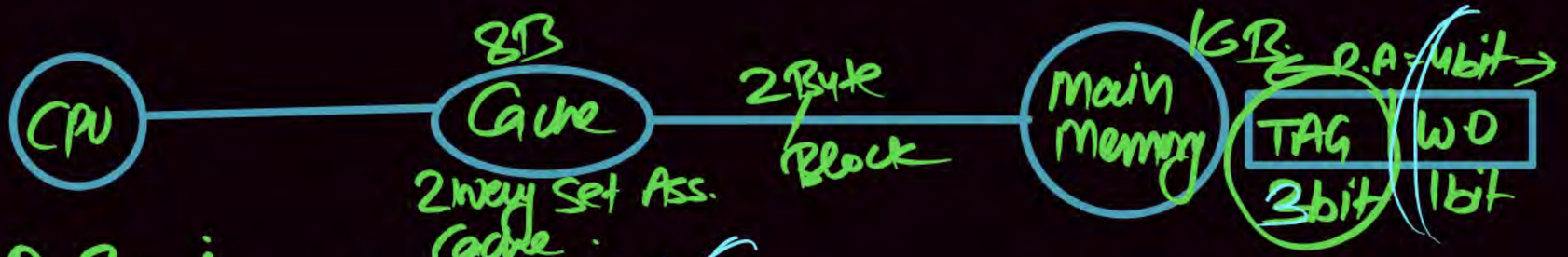
Set Associative Mapping




8B
CPU
Cache
2Byte
Block
main memory
1GB
P.A = 4bit
TAG
W.O
2bit
1bit
2way Set Ass.
Cache
PA = 4bit
TAG
S.O
W.O
2bit
1bit
1bit
TAG
xx
xx
SET 0
TAG
xx
xx
SET 1
000
001
010
011
100
101
110
111
B0
B1
B2
B3
B4
B5
B6
B7


CPU
8B
Cache
2Byte
Block
main memory
1GB
P.A = 4bit
TAG
W.O
3bit
1bit
2way Set Ass.
Cache
P.A = 4bit
TAG
S.O
W.O
2bit
1bit
1bit
(1) Bo ⇒ K MOD S = i
K MOD 2 = i
0 MOD 2 = CM SET '0'
OR
B0[000] ⇒
TAG
S.O
00
0
TAG
00
xx
SET 0
Bo
✓
✓
SET 1
xx
xx
✓
✓
(2bit) TAG
Set offset
000
001
010
011
100
101
110
111
B0
B1
B2
B3
B4
B5
B6
B7
SET 0
SET 1
SET 0
SET 1
SET 0
SET 1
SET 0
SET 1

(2)

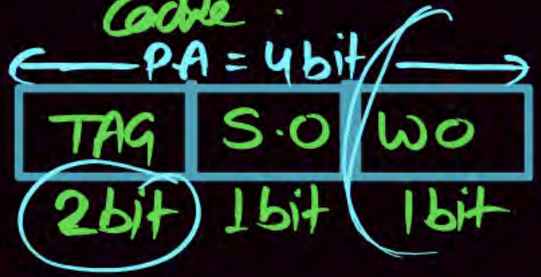


$B_7(111) \Rightarrow$ k MOD S = i

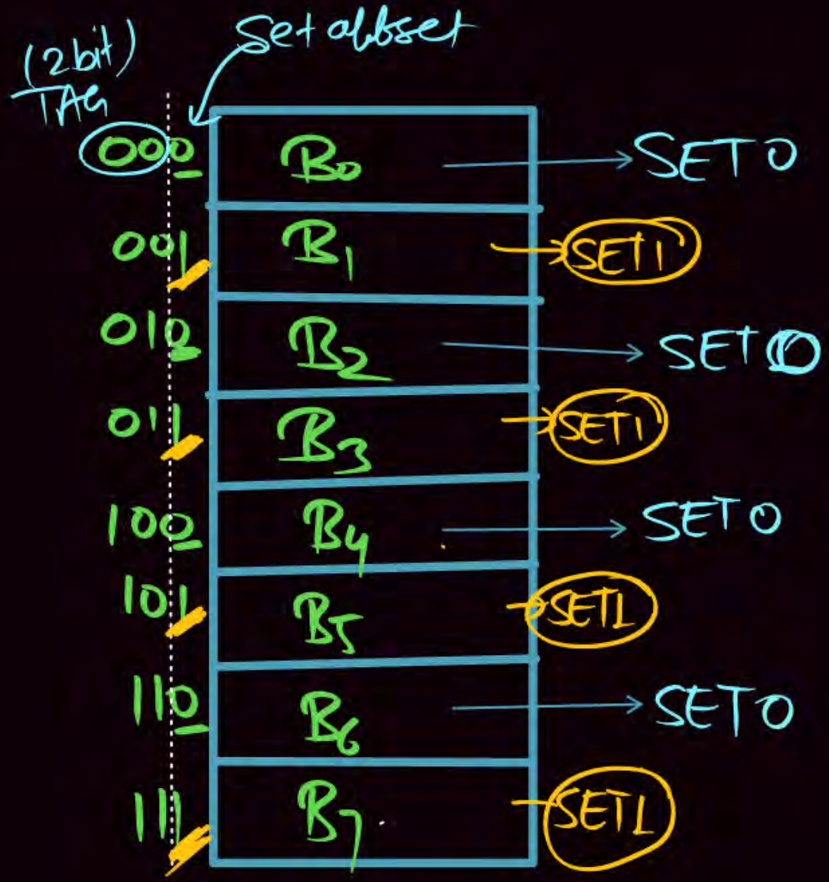
k MOD 2 = i

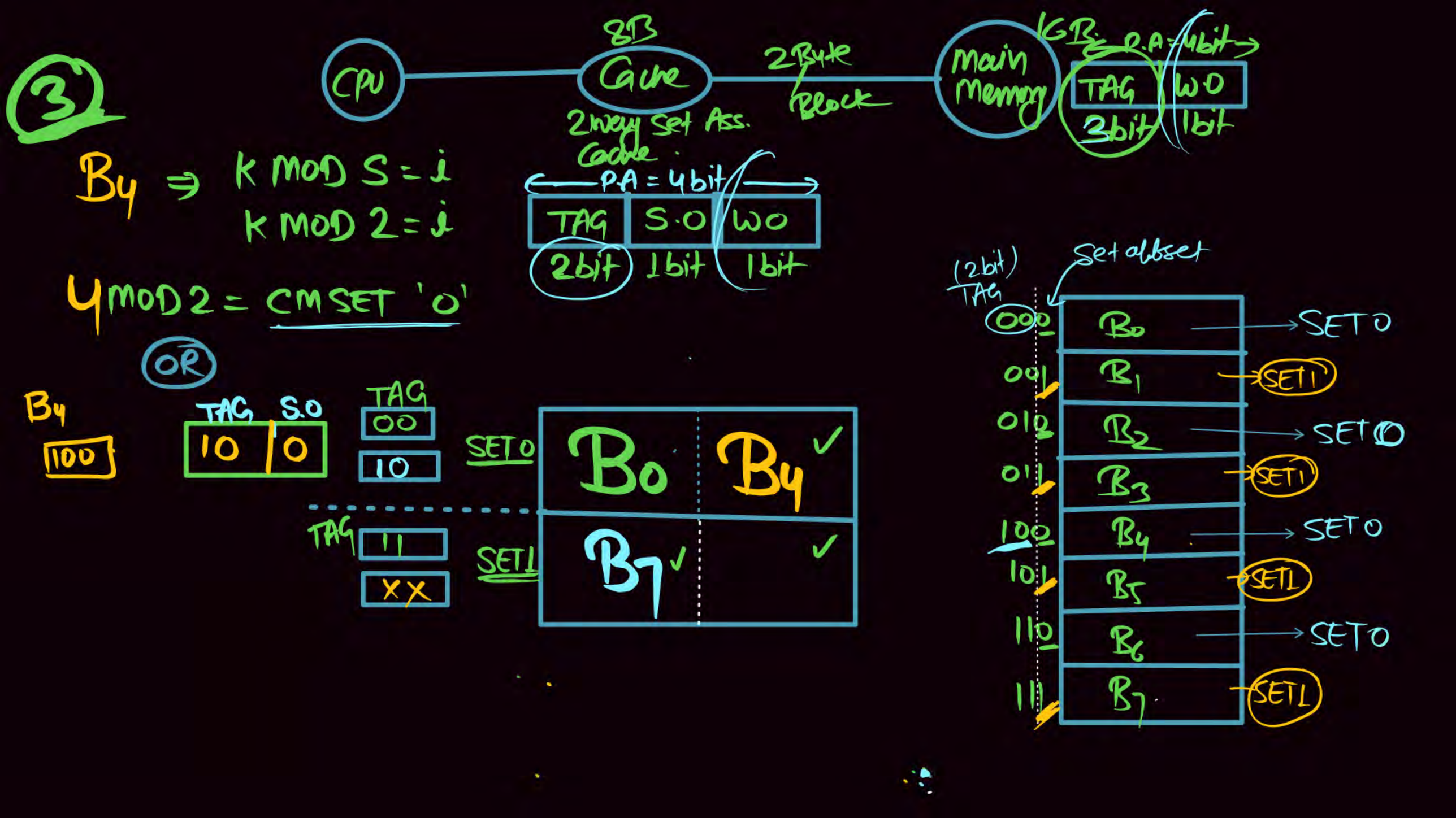
7 MOD 2 = CM SET '1'

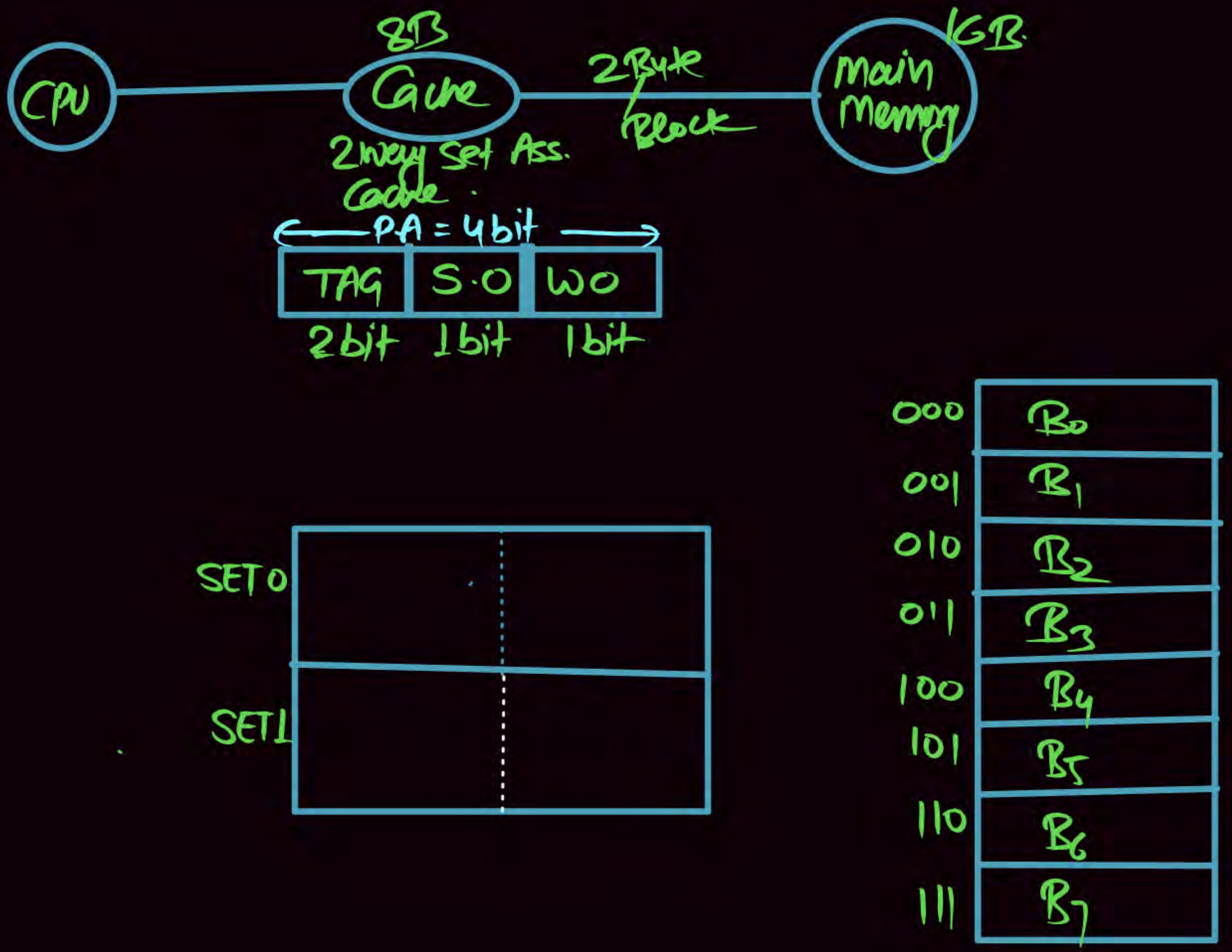
2 way Set Ass. Cache

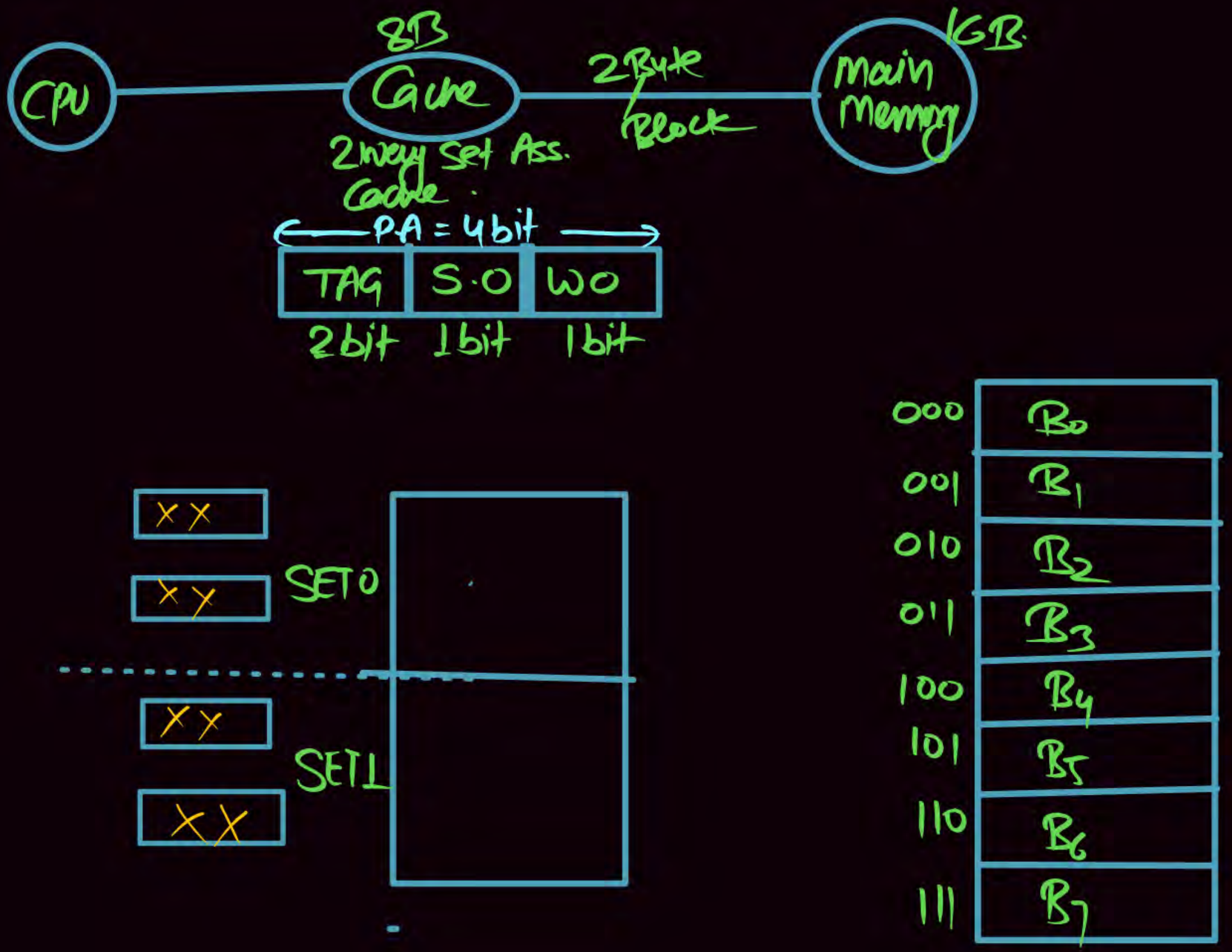


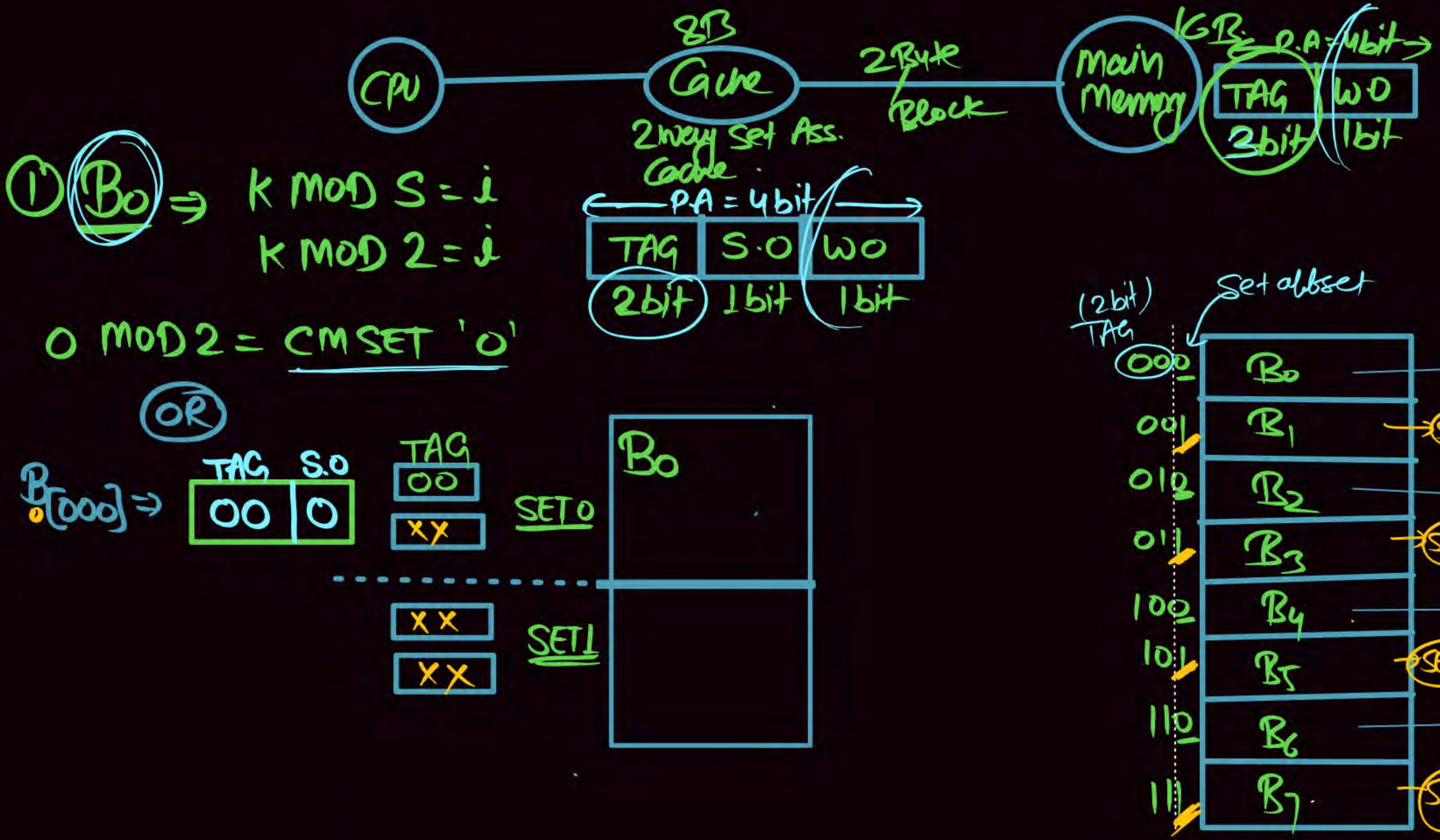
OR




3
CPU
8B
Cache
2Byte
Block
main memory
1GB
P.A = 4bit
TAG
W.O
3bit
1bit
2way Set Ass.
Cache
P.A = 4bit
TAG
S.O
W.O
2bit
1bit
1bit
B4 ⇒ K MOD S = i
K MOD 2 = i
4 MOD 2 = CM SET '0'
OR
B4
100
TAG
S.O
10
0
TAG
00
10
SET 0
TAG
11
XX
SET 1
B0
B4
B7
(2bit) TAG
Set offset
000
001
010
011
100
101
110
111
B0
B1
B2
B3
B4
B5
B6
B7
SET 0
SET 1
SET 0
SET 1
SET 0
SET 1
SET 0
SET 1


8B
CPU
Cache
2Byte
Block
main memory
1GB
2 way Set Ass.
Cache
PA = 4bit
TAG
S.O
WO
2bit
1bit
1bit
SET 0
SET 1
000
001
010
011
100
101
110
111
B0
B1
B2
B3
B4
B5
B6
B7


8B
CPU
Cache
2Byte
Block
main memory
1GB
2 way Set Ass.
Cache
PA = 4bit
TAG
S.O
WO
2bit
1bit
1bit
SET 0
SET 1
000
001
010
011
100
101
110
111
B0
B1
B2
B3
B4
B5
B6
B7


8B
CPU
Cache
2Byte
Block
main memory
1GB
P.A = 4bit
TAG
W.O
3bit
1bit
2way Set Ass.
Cache
P.A = 4bit
TAG
S.O
W.O
2bit
1bit
1bit
(1) Bo ⇒ K MOD S = i
K MOD 2 = i
0 MOD 2 = CM SET '0'
OR
B0[000] ⇒
TAG
S.O
00
0
TAG
00
XX
SET 0
XX
XX
SET 1
Bo
(2bit)
TAG
Set offset
000
001
010
011
100
101
110
111
B0
B1
B2
B3
B4
B5
B6
B7
SET 0
SET 1
SET 0
SET 1
SET 0
SET 1
SET 0
SET 1

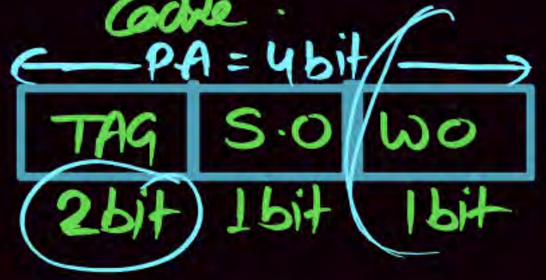
(2)

$B_7(111) \Rightarrow$ k MOD S = i

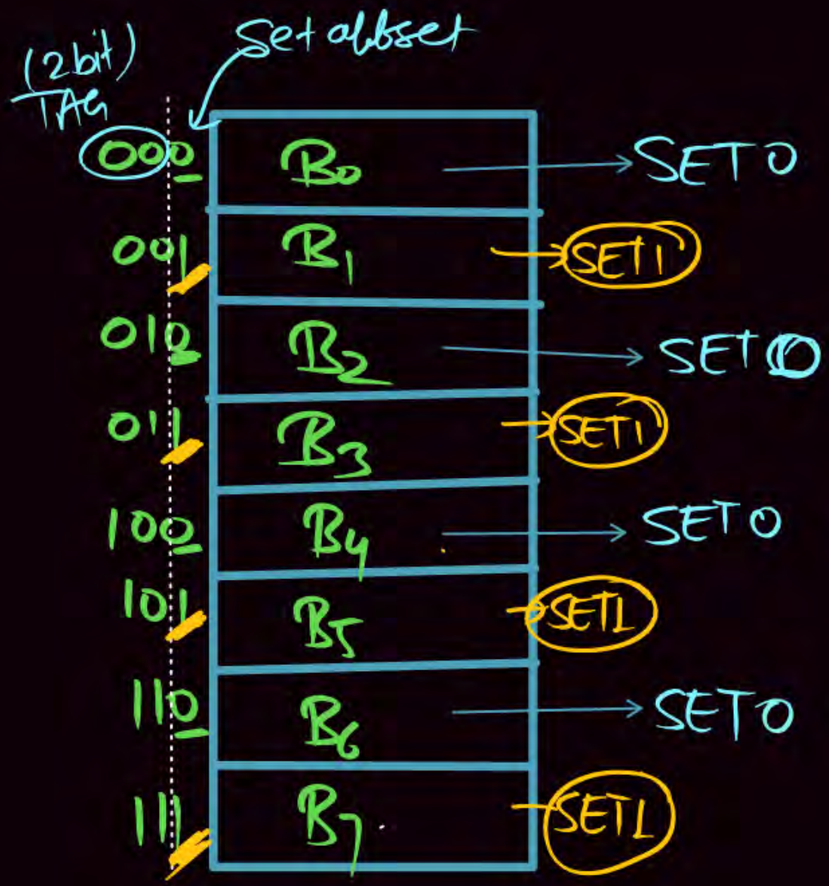
k MOD 2 = i

7 MOD 2 = CM SET '1'

2 way Set Ass. Cache



OR

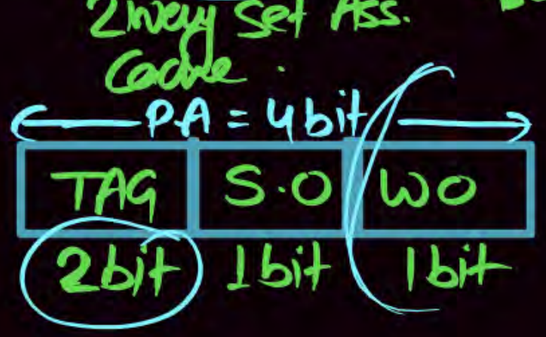


(3)

$B_4 \Rightarrow$ K MOD S = i

K MOD 2 = i

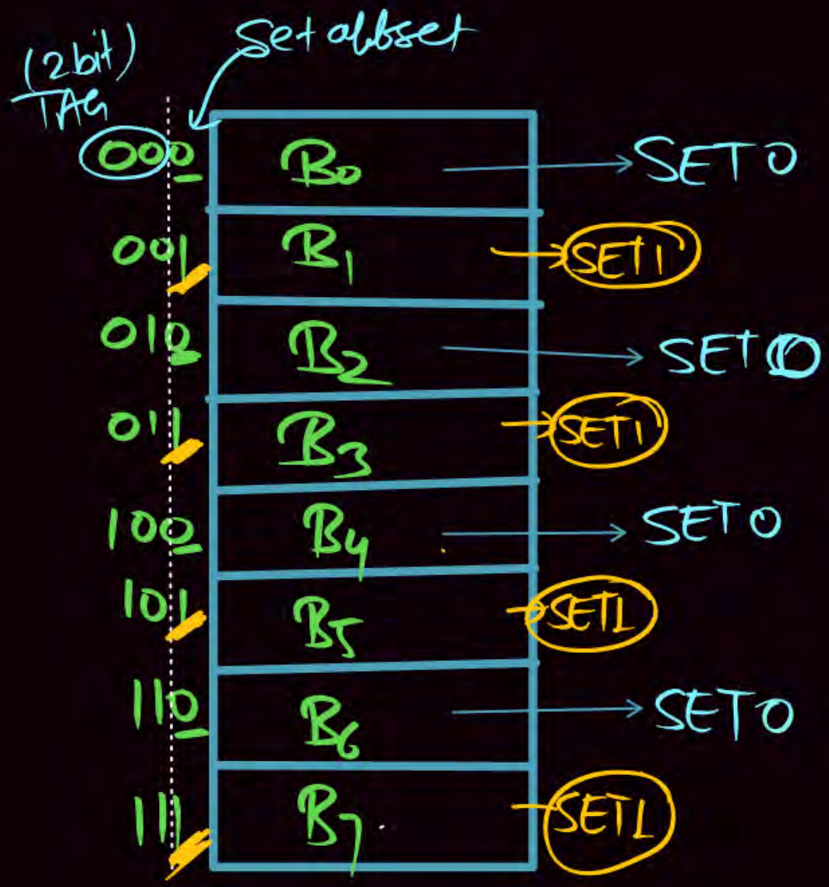
4 MOD 2 = CM SET '0'



OR

B_4

100



Advantage of set Associative Mapping

Disadvantage of Direct Mapping

Advantage of Set Associative Mapping.

I_1: MOV r_0 [0000] 0th Byte of Block B_0

I_2: MOV r_1 [1000] 0th Byte of Block B_4

I_3: MOV r_2 [0001] 1st Byte of Block B_0

I_4: MOV r_3 [1001] 1st Byte of Block B_4

I_5: MOV r_4 [0000] 0th Byte of Block B_0

I_6: MOV r_5 [1000] 0th Byte of Block B_4

Execution

Disadvantage of Direct Mapping

Advantage of Set Associative Mapping.

I_1: MOV r_0 [0000] 0th Byte of Block (B_0) — I_1 : HIT [Cache Hit]

I_2: MOV r_1 [1000] 0th Byte of Block (B_4) — I_2 : Miss [Cache Miss]

I_3: MOV r_2 [0001] 1st Byte of Block (B_0) — I_3 : HIT

I_4: MOV r_3 [1001] 1st Byte of Block (B_4) — I_4 : HIT

I_5: MOV r_4 [0000] 0th Byte of Block (B_0) — I_5 : HIT

I_6: MOV r_5 [1000] 0th Byte of Block (B_4) — I_6 : Hit

6 Accesses Total: 1 Miss

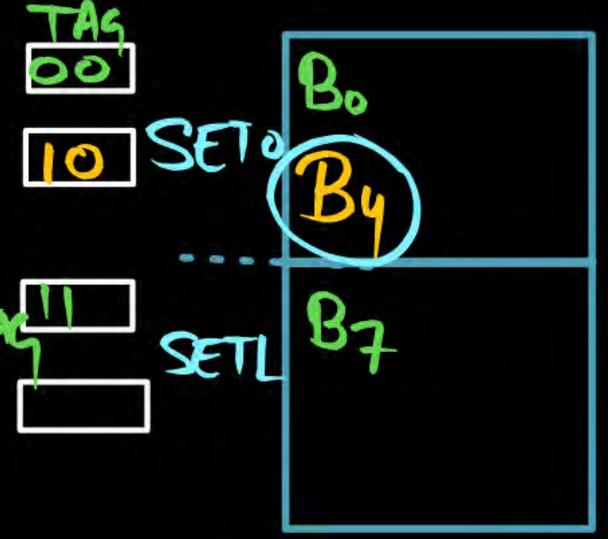
5 Hit

B_4 (100)

SET '0'

1	0	0

B_4 Brought from MM to CM



Disadvantage of Direct Mapping

Advantage of Set Associative Mapping.

I_1: MOV r_0 [0000] 0th Byte of Block (B_0) I_1: HIT

I_2: MOV r_1 [1000] 0th Byte of Block (B_4) I_2: MISS

I_3: MOV r_2 [0001] 1st Byte of Block (B_0) I_3: HIT

I_4: MOV r_3 [1001] 1st Byte of Block (B_4) I_4: HIT

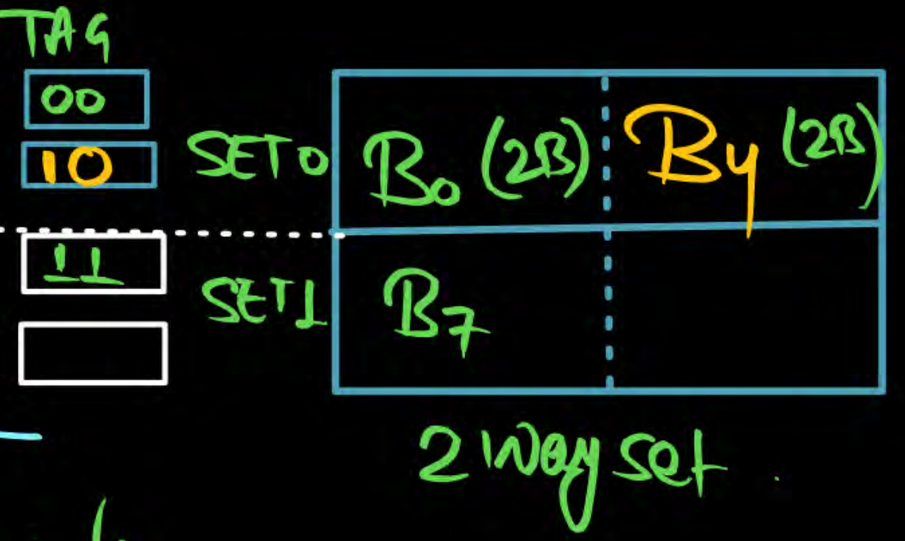
I_5: MOV r_4 [0000] 0th Byte of Block (B_0) I_5: HIT

I_6: MOV r_5 [1000] 0th Byte of Block (B_4) I_6: HIT

Total 6 Accesses : 1 Miss, 5 Cache Hit.

CM SET: 4 MOD 2 = '0' (B4 MM to CM)

TAG	SET
10	0



2 Way set

Disadvantage of Direct Mapping

I_1: MOV r_0 [0000]

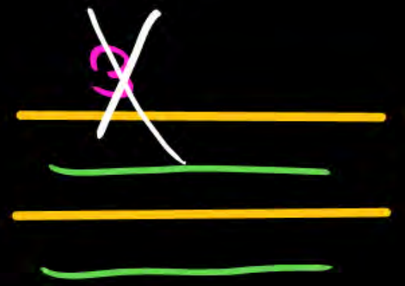
I_2: MOV r_1 [1000]

I_3: MOV r_2 [0001]

I_4: MOV r_3 [1001]

I_5: MOV r_4 [0000]

I_6: MOV r_5 [1000]



Set Associative Mapping function

Cache Set Address = MM Request MOD #SET is in Cache

(OR)

K MOD S = i

k: MM Block Number
s: # Cache Set
i: Cache Set Number

Q What is the Meaning of.

(i) 2 Way Set Associative.

(ii) 4 way Set Associative

(iii) 8 way Set Associative

(iv) N-Way Set Associative

$K \bmod S = i$

S : # SETS (Number of Sets)

eg Number of Lines = 16

(i) 2 Way Set Associative

(ii) 4 way Set Associative

(iii) 8 way Set Associative

(iv) 16 way Set Associative.

$$\#SET = \frac{\#LINE}{N\text{-way}}$$

Example1: # Line's = 16 & 2 way set Associative

$$\#\,SET = \frac{\#LINES}{N\text{-way}} = \frac{16}{2} = 8$$

#SET = 8 | S = 8

K MOD S = i

K MOD 8 = i

SET 0 | SET 1 | 2 | 3 | 4 | 5 | SET 6 | SET 7

K MOD 8 = i 0 - 7

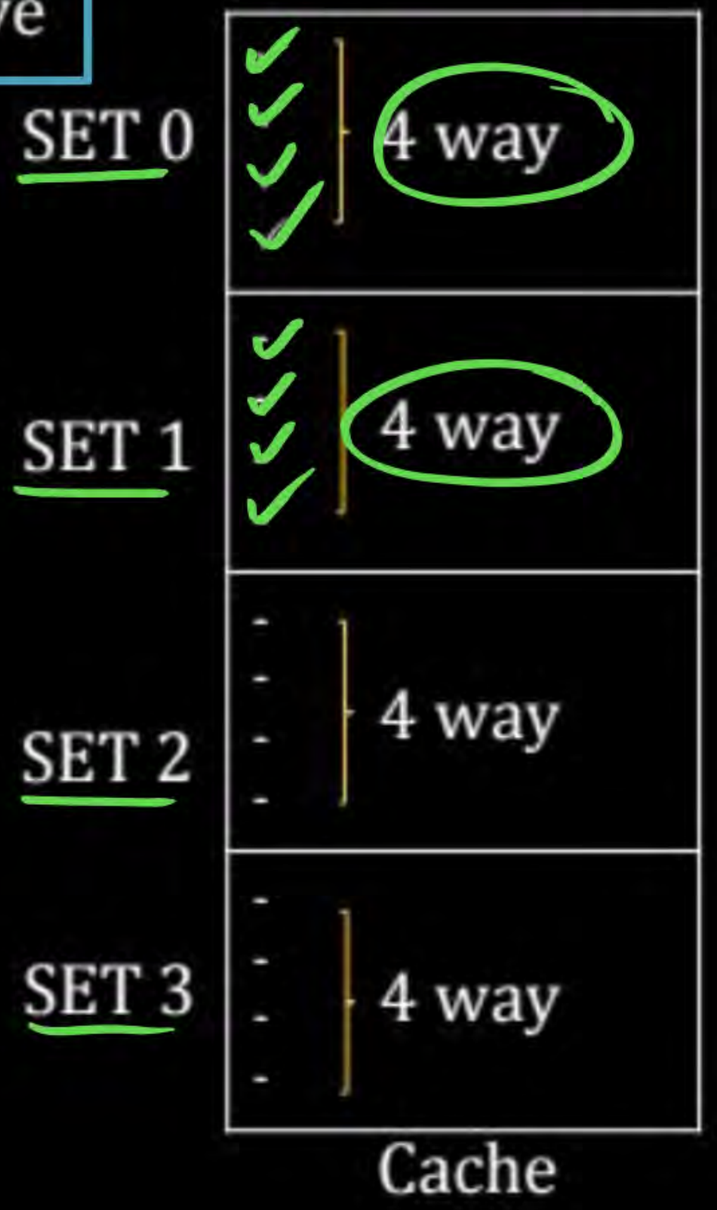
Example2:

#LINE = 16 & 4 way set Associative

$$\#\,SET = \frac{16}{4} \Rightarrow 4$$

S = 4

K MOD 4 = i



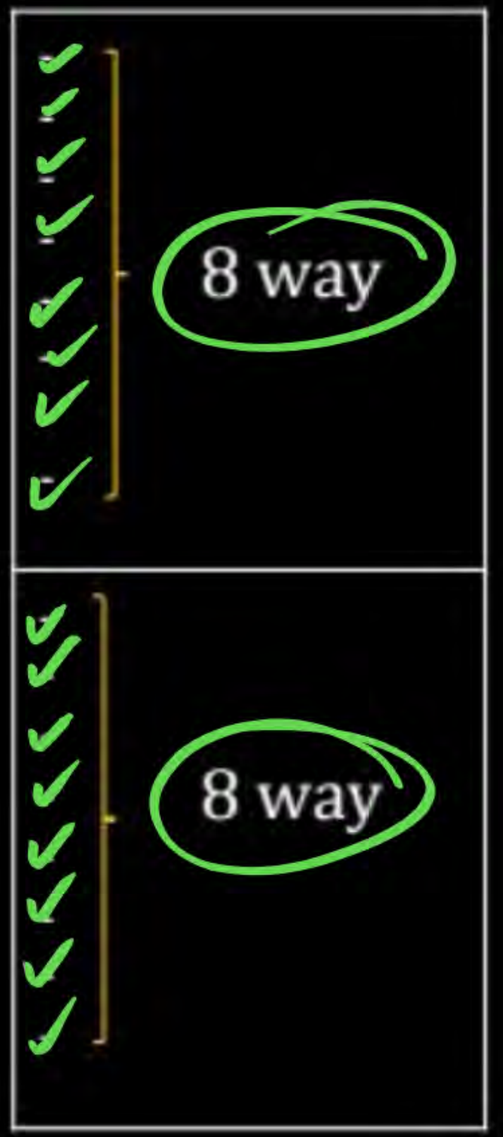
Example3: #LINE = 16 & 8 way set Associative

$\#\,SET = \frac{16}{8} \Rightarrow 2$

S = 2

K MOD 2 = i

k MOD S = i



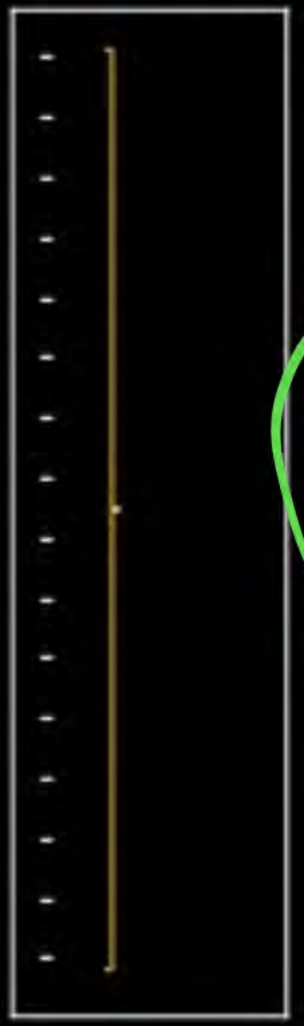
Example4:

#LINE = 16 & 16 way set Associative

$$\#\ SET = \frac{16}{16} \Rightarrow 1$$

S = 1

K MOD 1 = i



Fully
Associative
(Whole cache as a set)

Q What is the Meaning of.

(i) 2 Way Set Associative : In a One Cache Set (2) Main Memory Block can store.

(ii) 4 way set Associative : In a One Cache Set (4) MM Block can Store.

(iii) 8 Way Set Associative : In a One cache set (8) MM Block can Store.

(iv) N-Way Set Associative : In a One Cache set (N) MM Block can Store.

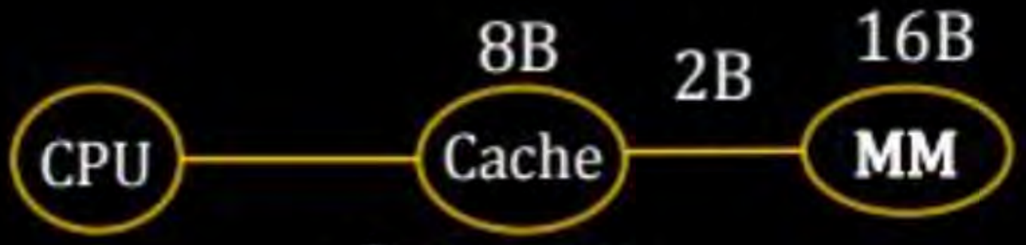
Q What is the Meaning of.

(i) 2 Way Set Associative : In a One Cache Memory Set we can Store a MM Block by 2 ways (2 mm Blocks)

(ii) 4 way Set Associative : In a One CM Set, we can Store MM Block by 4 Way

(iii) 8 Way Set Associative : In a One CM Set, we Can Store MM Block by 8 ways.

(iv) N-Way Set Associative : In a One CM Set, we Can Store MM Block by N Way.



2 way set associative

K MOD S = i

P.A

TAG	S.O	W.O

2 bit 1 bit 1 bit

TAG

00	0	B_0	
XX			
11	1	B_7	
XX	SET		

Cache

000	B_0
001	B_1
010	B_2
011	B_3
100	B_4
101	B_5
110	B_6
111	B_7

$B_{0[000]}$ →

TAG	S.O
00	0

→ SET '0'

2 bit 1 bit

$B_{7[111]}$ →

TAG	S.O
11	1

→ SET '1'

MM BLOCK Mapping Tech. CM SET

$B_{0[000]}$ $\xrightarrow[0 \text{ MOD } 2 = \text{'0'}]{K \text{ MOD } S = i}$ SET '0'

$B_{7[111]}$ $\xrightarrow[7 \text{ MOD } 2 = \text{'1'}]{K \text{ MOD } S = i}$ SET '1'

Tag Memory Size = #SETS × #LINES In a each set × Tag bits

Example: # SET = 2 # LINE in Each set =2 TAG = 2 bit

Tag Memory size = 2 × 2 × 2 = 8 bits

Q.

[Common Data for this and next question]
A computer has a 256 Kbyte, 4-way set associative. Write back data cache with block size of 32 Bytes. The processor sends 32 bit address to the cache controller. Each cache tag directory entry contains, in addition to address tag, 2 valid bits, 1 modified bit and 2 replacement bit.

The number of bits in the tag field of am address is
(a) 11 (b) 14 (c) 16 (d) 27

[GATE - 2012: 2 Marks]

Q.

[Common Data from previous question]
A computer has a 256 Kbyte, 4-way set associative. Write back data cache with block size of 32 Bytes. The processor sends 32 bit address to the cache controller. Each cache tag directory entry contains, in addition to address tag, 2 valid bits, 1 modified bit and 2 replacement bit. [GATE - 2012: 2 Marks]

The size of the cache tag directory is

(a) 160 Kbits (b) 136 Kbits

(c) 40 Kbits (d) 32 Kbits

NAT

Consider a machine with a byte addressable main memory of 2^{32} bytes divided into blocks of size 32 bytes. Assume that a direct mapped cache having 512 cache lines is used with this machine. The size of the tag field in bits is _____.

[GATE-2017(Set-1)-CS: 2M]

MCQ

A computer system with a word length of 32 bits has a 16 MB byte-addressable main memory and a 64 KB, 4-way set associative cache memory with a block size of 256 bytes. Consider the following four physical addresses represented in hexadecimal notation.

A1=0×42C8A4,A2=0×546888,A3=0×6A289C,A4=0×5E4880

Which one of the following is TRUE? [GATE-2020-CS: 2M]

A. A1 and A3 are mapped to the same cache set.

B. A2 and A3 are mapped to the same cache set.

C. A3 and A4 are mapped to the same cache set.

D. A1 and A4 are mapped to different cache sets.

NAT

The width of the physical address on a machine is 40 bits. The width of the tag field In a 512 KB 8-way set associative cache is _____bits.

[GATE-2016(Set-2)-CS: 2M]

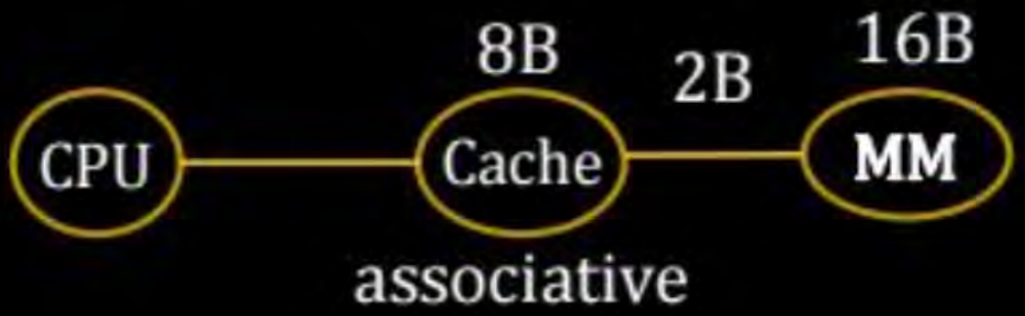
NAT

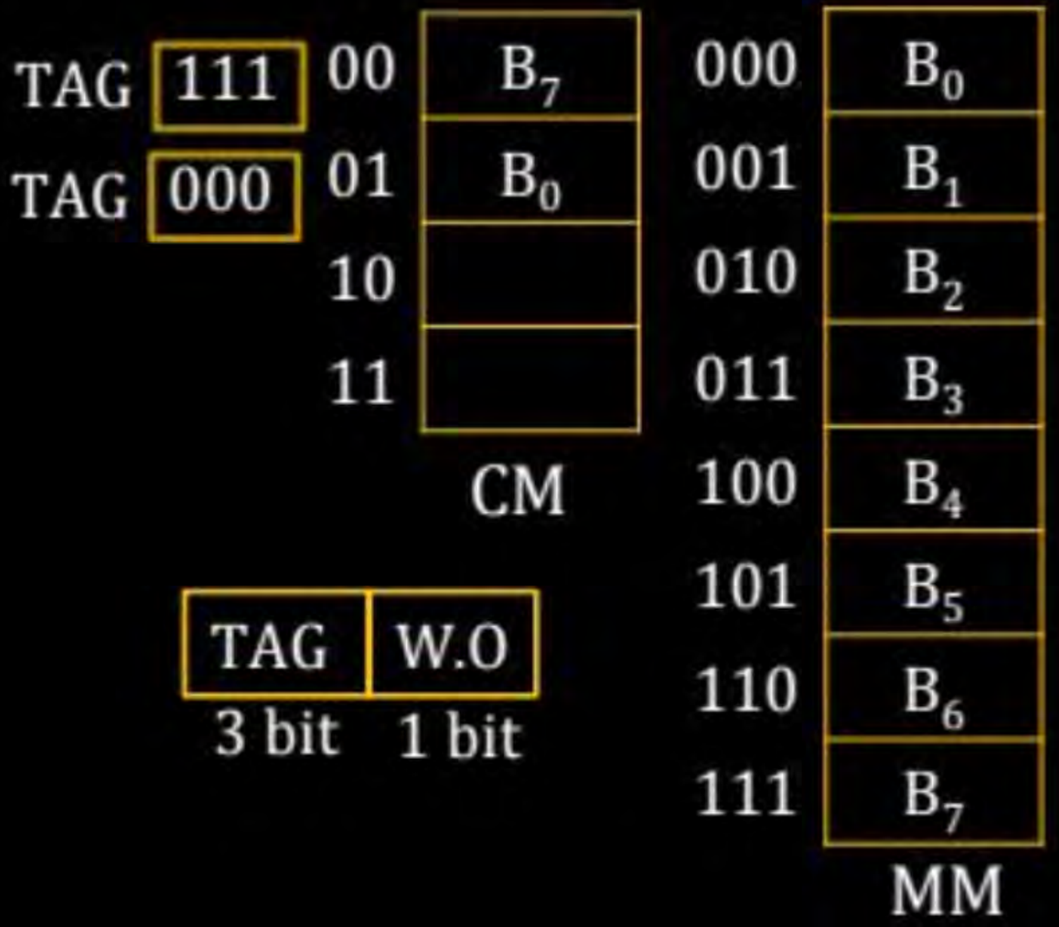
A 4-way set-associative cache memory unit with a capacity of 16 KB is built using a block size of 8 words. The word length is 32 bits. The size of the physical address space is 4 GB. The number of bits for the TAG field is _____.

[GATE-2014(Set-2)-CS: 1M]

3) Associative Mapping

- In this no mapping function is used to transfer the data from MM to CM.
- No Conflict Miss





MM Block	Associative Mapping	CM Line
$B_{7[111]}$	No mapping → Function	Any line
$B_{0[000]}$	No mapping → Function	Any line

Tag Memory Size = # LINES × Tag bits

Example: #LINE =4 & Tag bits =3

Tag Memory Size = $4 \times 3 = 12$ bits

Q.

Consider fully associative cache consists 8 Block & MM contain 128 Block & Request made by the CPU:
119, 84, 37, 0, 16, 0, 84, 120, 121, 93, 37, 0, 43, 39, 47, 48.
Calculate # of compulsory & Capacity miss?

0	119 43
1	84 39
2	37 47
3	0 48
4	16
5	120
6	121
7	93

Hit: 0, 84, 37, 0

#Hits: 4

Compulsory Miss = 8

Capacity Miss = 4

Total Miss = 12

Replacement Algorithm

When Cache is Full, then replacement algorithm are required to replace the exist cache block with new block.

In the CM design 3 type of replacement algorithm is used.

1) Random Algorithm

2) FIFO Replacement

3) LRU Replacement

In the random algorithm, any cache block can be replaced based on the random selection.

Q.

Consider 4 block cache memory (initially empty) with the following MM block references.

7, 8, 10, 15, 7, 8, 16, 7, 8, 10

Identify the Hit Ratio using

(i) FIFO

(ii) LRU

(iii) Direct Mapped cache

(iv) 2 - way Set Associative with LRU

Q.

Consider a small two-way set-associative cache memory, consisting of 4 blocks. For choosing the block to be replaced, use the least recently used (LRU) scheme. The number of cache misses for the following sequence of block addresses is 8, 12, 0, 12, 8

(a) 2 (b) 3 (c) 4 (d) 5

[GATE - 2004]

Q.

Consider a 2- way set associative cache with 256 blocks and uses LRU replacement. Initially the cache in empty conflict misses are those misses which occur due to contention of multiple blocks for the same cache set. Compulsory misses occur due to first time access to the block. The following sequence of accesses to memory blocks (0, 128, 256, 128, 0, 128, 256, 128, 1, 129, 257, 129, 1, 129, 257, 129) is repeated 10 times. The number of conflict misses experienced by the cache is _____.

[GATE - 2017]

Q.

A computer system has a level - 1 instruction cache (1-cache), a level-1 data cache(D-cache) and a level-2 cache(L2-cache) with the following specifications.

	Capacity	Mapping method	Block size
1-cache	4K words	Direct mapping	4 words
D-cache	4K words	2-way set associative mapping	4 words
L2-cache	64K words	4-way set associative mapping	16 words

Capacity mapping method block size 1-cache 4K words direct mapping 4 words D-cache 4 k words 2 way set associative mapping 4 words L2-cache 64K words 4-way set associative mapping 16 words. The length of the physical address of a word in the main memory is 30 bits. The capacity of the tag memory in the 1-cache, D-cache & L2-cache is. Respectively,

(a) 1K×18-bit, 1K×19-bit, 4K×16-bit (b) 1K×16-bit, 1K×19-bit, 4K×18-bit

(c) 1K×16-bit, 512×18-bit, 1K×16-bit (d) 1K×16-bit, 512×18-bit, 1K×18-bit

[GATE - 2006: 2 Marks]

THANK YOU!