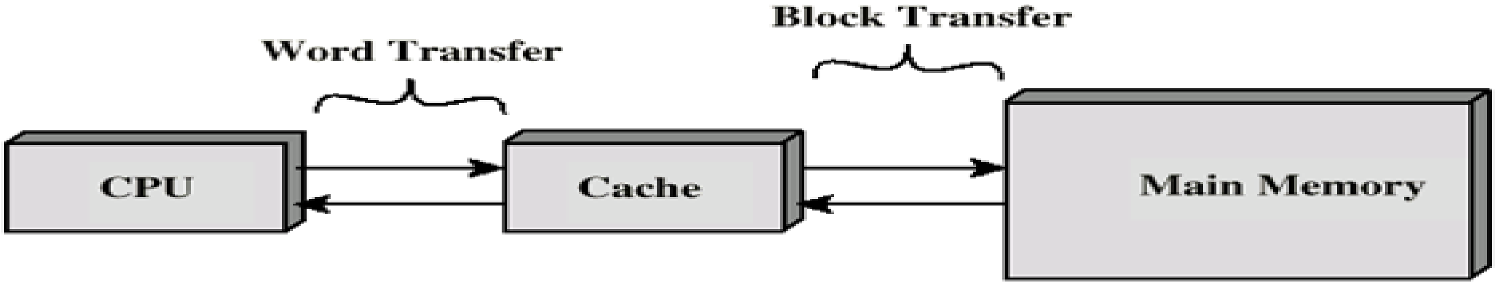
CACHE MEMORY

P. K. Roy

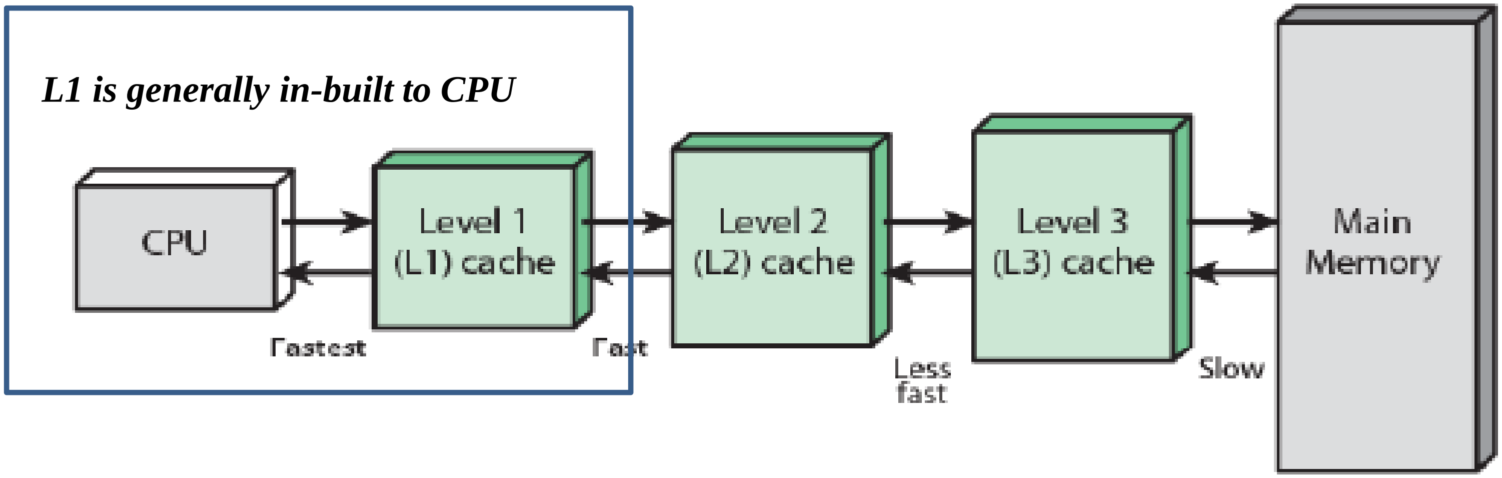
What is Cache ?

- A Small amount of fast SRAM-based memory
  - Sits between normal main memory and CPU
  - May be located on CPU chip or in system
  - Objective is to make slower memory system look like fast memory.
  - There may be more levels of cache (L1, L2,..)
  - Unlike main memory, which is accessed by address, cache is typically accessed by content; hence, it is often called ***content addressable memory***.
  - Capitalizes on ***spatial locality and temporal locality***

Cntd…



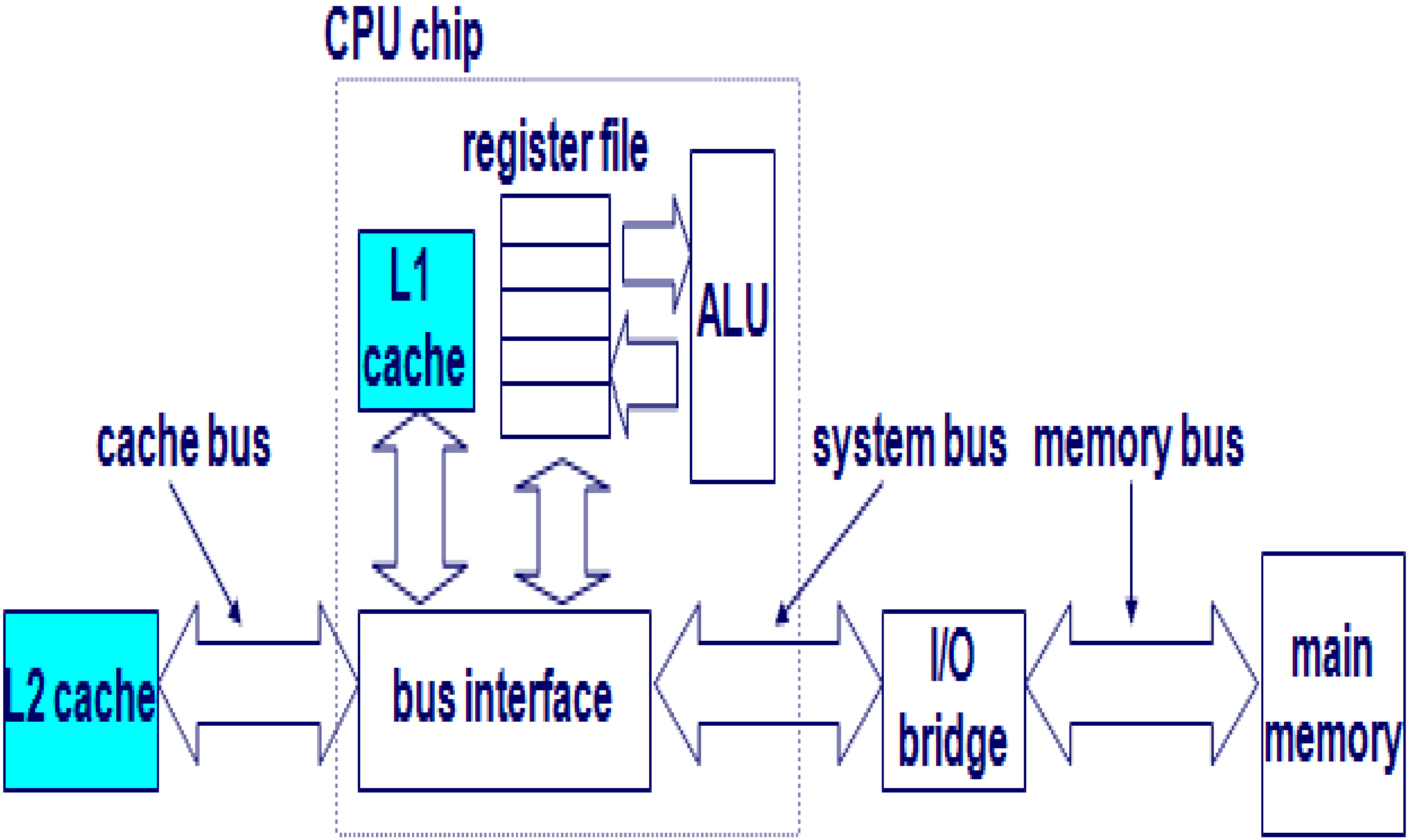
a) Cache between CPU & Main Memory



L1 is generally in-built to CPU

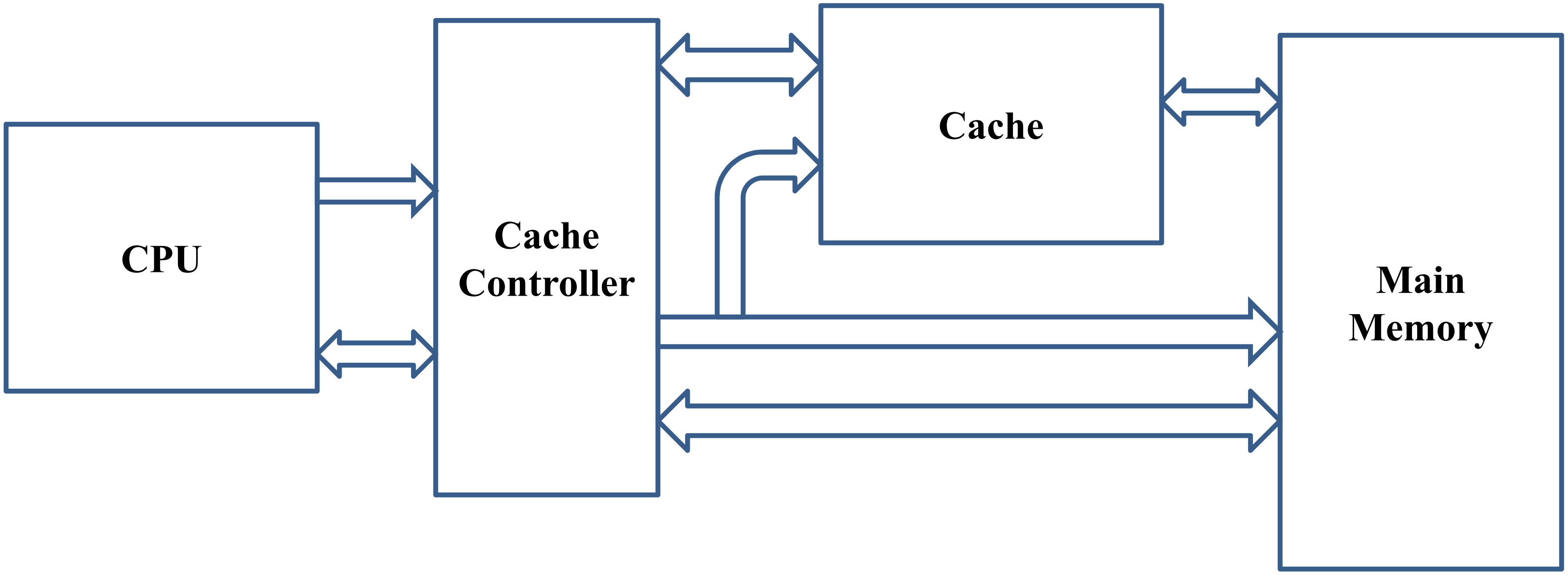
b) Computer system with 3-levels of cache

Cntd…



Internal organization with 2-level cache

Bus Structure Between Cache & CPU

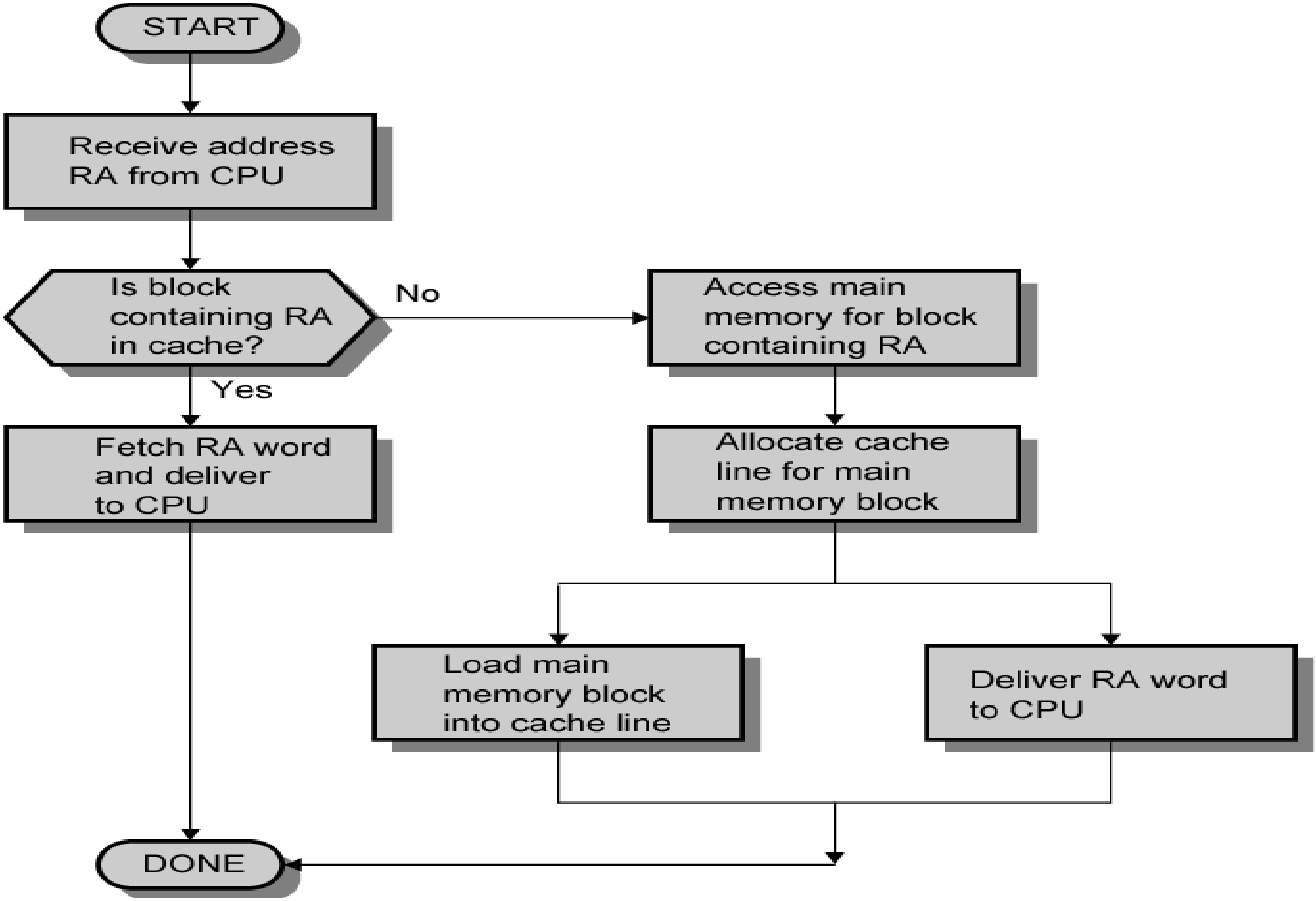


Cache Operation

1. CPU requests contents of memory location
2. Cache is checked for this data
3. If present, get from cache (fast)
4. If not present, read required block from main memory to cache
5. Then deliver from cache to CPU
6. Cache includes tags to identify which block(s) of main memory are in the cache

Note: CPU looks first for data in L1, then in L2, then in main memory (if 2-level cache is used).

Cache Access - Flowchart


START
Receive address RA from CPU
Is block containing RA in cache?
No
Yes
Access main memory for block containing RA
Fetch RA word and deliver to CPU
Allocate cache line for main memory block
Load main memory block into cache line
Deliver RA word to CPU
DONE

The Principle of Locality of Reference

- Program access a relatively small portion of the address space at any instant of time.
- once a byte is accessed, it is likely that a nearby data element will be needed soon.

  Example: 90% of time in 10% of the code

- There are three forms of locality:

  Sequential locality : Instructions tend to be accessed sequentially.

  Temporal Locality (Locality in Time): If an item is referenced, it will tend to be referenced again soon.

  Spatial Locality (Locality in Space): If an item is referenced, items whose addresses are close by tend to be referenced soon.

Write policies

- **Write-through Policy:**
  - Both main memory & cache memory are updated concurrently during a write operation.
  - ***Advantage*** - most recently updated data always in main memory.
  - ***Disadvantage*** - time to update main memory
    - increasing bus traffic, hampering system performance.
- **Write-back Policy:**
  - Only cache memory is updated during a write operation.
  - Main memory is updated only when it is removed from cache.
  - ***Advantage*** – as long as the word remains in cache, all the requests for that word will be fulfilled from the cache without accessing the main memory.
  - ***Disadvantage*** – it is essential to update MM when cache is freed.

Some Definitions

- **Working Set:** Set of memory locations CPU refers to at any instant of time.

- **Cache Hit:** When the requested data found in cache.

- **Cache Hit Time:** Time taken to access data from cache in case of a cache hit.

- **Cache Miss:** When the requested data not found in cache.

  - ***Compulsory Miss (Cold Miss)*** – Occurs at the first time.
  - ***Capacity Miss* –** when the working set is larger than cache's capacity.
  - ***Conflict Miss* –** when more than one requests for a particular location of cache.

Cntd…

- **<u>Miss Penalty</u>:** The additional time required to service a miss.

  For a 2-level memory system (cache & main memory)

  $$\textit{Miss Penalty} = \textit{Cache Access Time} + \textit{Main Memory Access Time}$$

  $$= T_C + T_M$$

- **<u>Hit Ratio</u> :** Probability of getting hits out of some memory references made by the CPU.

  So, hit ratio (H) always within the range of **$0 \leq H \leq 1$**

$$H = \frac{No. of\ Hits}{Total\ Memory\ References}$$

$$H = \frac{No. of\ Hits}{No. of\ Hits + No. of\ Misses}$$

$$\textit{Hit Time} = H \times T_C$$

Cntd…

- **<u>Miss Ratio</u>:** Probability of getting misses out of some memory references made by the CPU.

  So, if ***H*** is the hit ratio the ***(1 – H)*** is the miss ratio.

$$\boldsymbol{Miss\ Ratio = \frac{No.of Misses}{Total\ Memory\ References}}$$

$$\boldsymbol{Miss\ Ratio = \frac{No.of Misses}{No.of\ Hits + No.of\ Misses}}$$

$$\boldsymbol{Miss\ Time = Miss\ Ratio \times Miss\ Penalty}$$

$$\boldsymbol{= (1 - H) \times (T_C + T_M)}$$

Cntd…

- **Average Memory Access Time (Read Cycle):**

For a 2-level memory system (Cache + Main Memory)

$$\begin{aligned} T_{AV} &= \textit{Hit Time} + \textit{Miss Time} \\ &= \textit{Hit Time} + (\textit{Miss Ratio} \times \textit{Miss Penalty}) \\ &= H \times T_C + (1 - H) \times (T_C + T_M) \\ &= T_C + (1 - H) \times T_M \end{aligned}$$

For a 3-level memory system (Cache + Main Memory + Secondary Memory)

$$T_{AV} = \textit{Cache Hit Time} + \textit{Main Memory Hit Time} + \textit{Miss Time}$$
$$= H_C \times T_C + (1 - H_C) \times H_M \times (T_C + T_M) + (1 - H_C) \times (1 - H_M) \times (T_C + T_M + T_S)$$

Where,

H_M = *Main Memory Hit Ratio*

T_M = *Main Memory Access Time*

T_S = *Secondary Memory Access Time*

H_C = *Cache Memory Hit Ratio*

Cntd…

- **<u>Average Memory Access Time (Read + Write Cycle):</u>**

Average Access Time (Write-through Policy)

$$= P_R \times T_{AV} + (1-P_R) \times T_M$$

Where,

P_R = *Probability of Read*

T_{AV} = *Memory Access Time for Read Cycle*

$(1-P_R)$ = *Probability of Write*

Since write-through policy is applied, access time for write cycle will be the main memory access time.

Cntd…

- **Efficiency of Memory System:** T_C / T_{AV}

- **Data Streaming:** When a miss occurs, it causes the cache controller to copy the data from main memory to cache and this data is forwarded to the CPU at the same time. This phenomenon is known as ***data streaming or load-o-demand policy.***

- **Dirty Bit:** A status bit is to indicate whether cache content has changed or not.

Cntd…

- **Cache block size or cache line size**– *the* amount of data that gets transferred on a cache miss.

- **Instruction cache** -- *cache that only holds* instructions.

- **Data cache** -- *cache that only caches data.*

- **Unified cache** -- *cache that holds both.*

Exercise

- A hierarchical cache-main memory subsystem has the following specifications –

  1. Cache Access Time = 50ns
  2. Main Memory Access Time = 500ns
  3. 80% of memory request are for read
  4. Hit Ratio for Read Access = 0.9 and the write-through scheme is used

Calculate :

1. Average access time for memory system considering only memory read cycle.
2. Average access time for memory system for both read and write cycle.

Cache Mapping

- The process of transforming data from main memory to cache memory.
- Addressing relationships between Main memory & cache.
- Many blocks of main memory map to a single block of cache. A ***tag*** field in the cache block distinguishes one cached memory block from another.
- ***valid bit*** - A bit of information that indicates whether the data in a block is valid (1) or not (0). It indicates whether data in the cache (hit) or not (miss).

- 3 types of cache mapping –

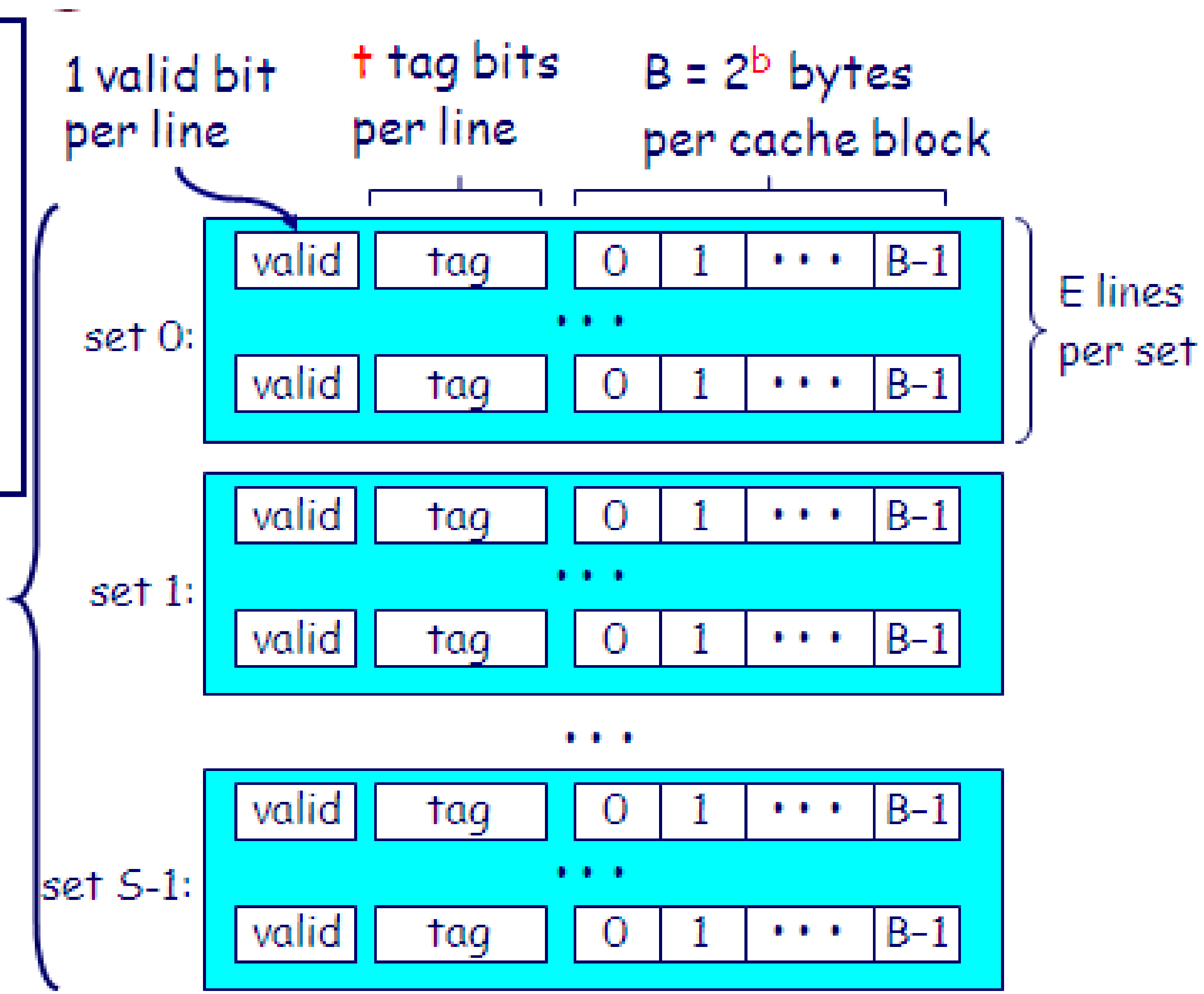
  - Direct Mapping
  - Associative Mapping
  - Set Associative Mapping

General Organization of a Cache

Cache is an array of sets

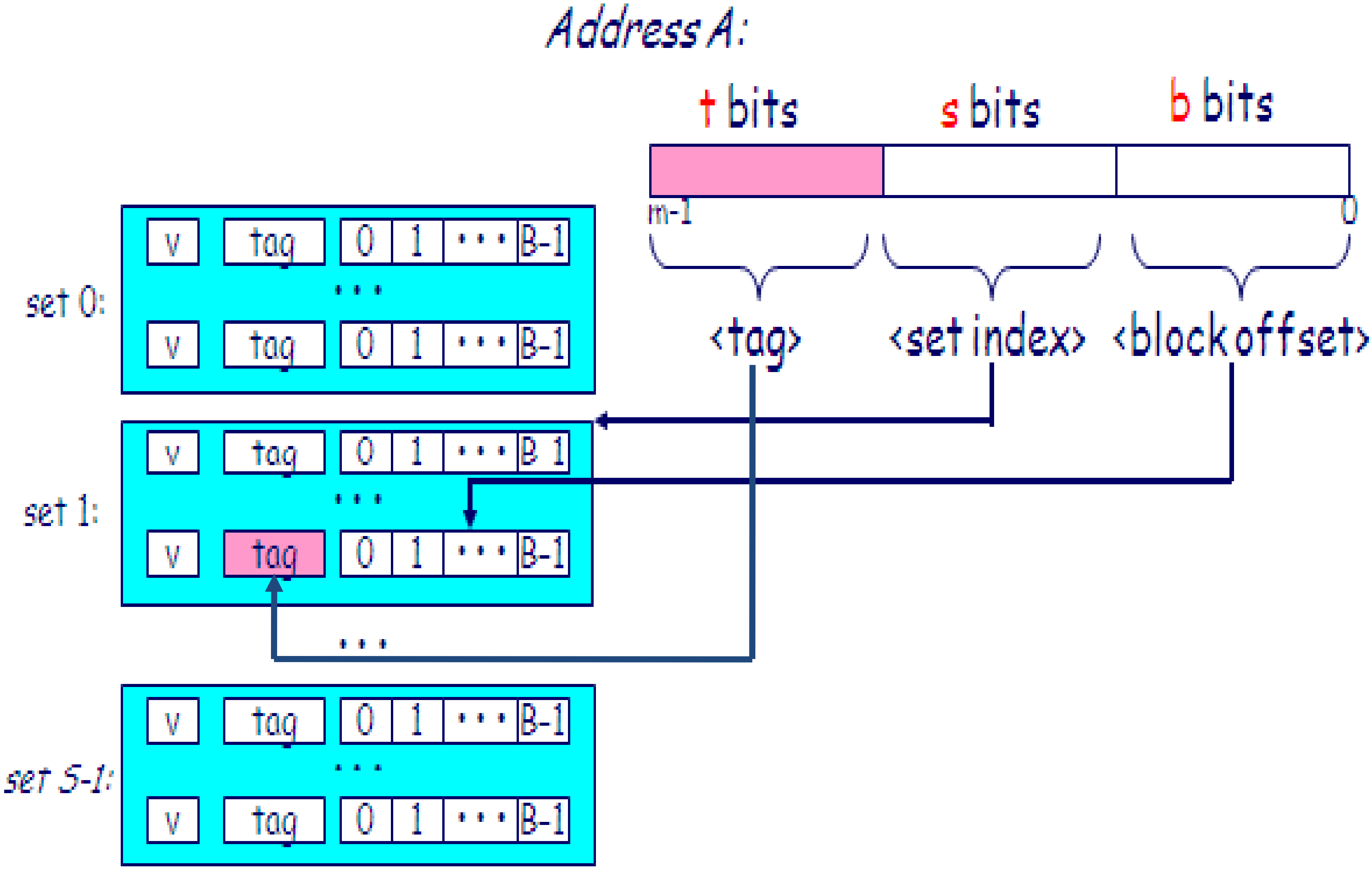
Each set contains one or more lines

Each line holds a block of data



Cache size: C = B x E x S data bytes

Addressing Caches



Direct Mapping

- Each block mapped to exactly 1 cache location.

  Cache location = (block address of main memory) MOD (no. of blocks in cache)

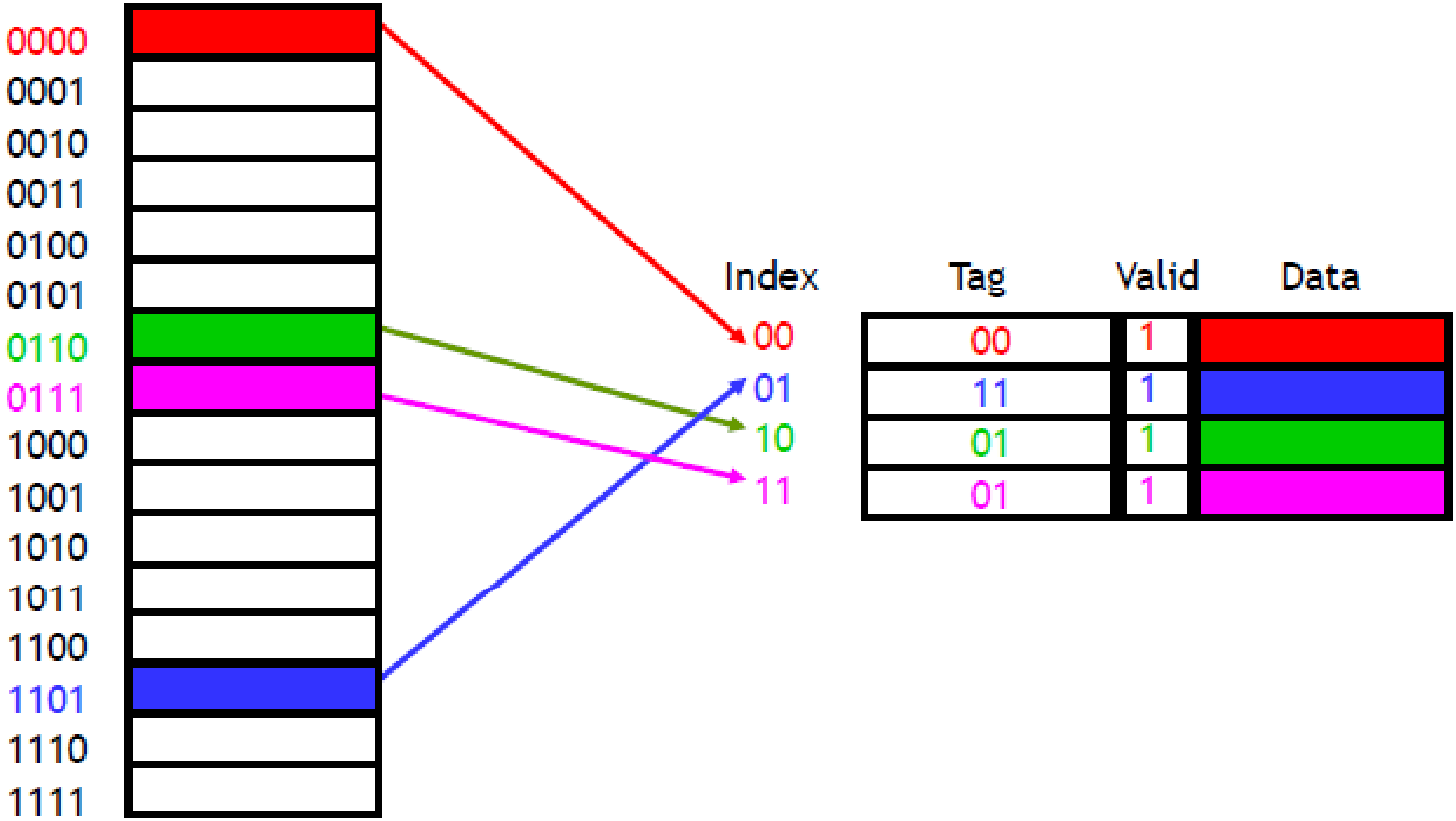
- CPU generated address (main memory address) is divided into 2 or 3 fields –

Tag	Index
High order bits to distinguish several memory locations that map to the same cache location	Low order bits to represent cache locations. Each location can hold one word (byte).

Tag	Block	Word
	Each location or block of the cache can hold a no. of words (multiple bytes)	

- One block per cache line (i.e. a set is equivalent to a block).
- A single valid bit per block to distinguish full and empty blocks.
- Easy to implement & lower cost.
- Suffers from conflict miss.

Cntd…

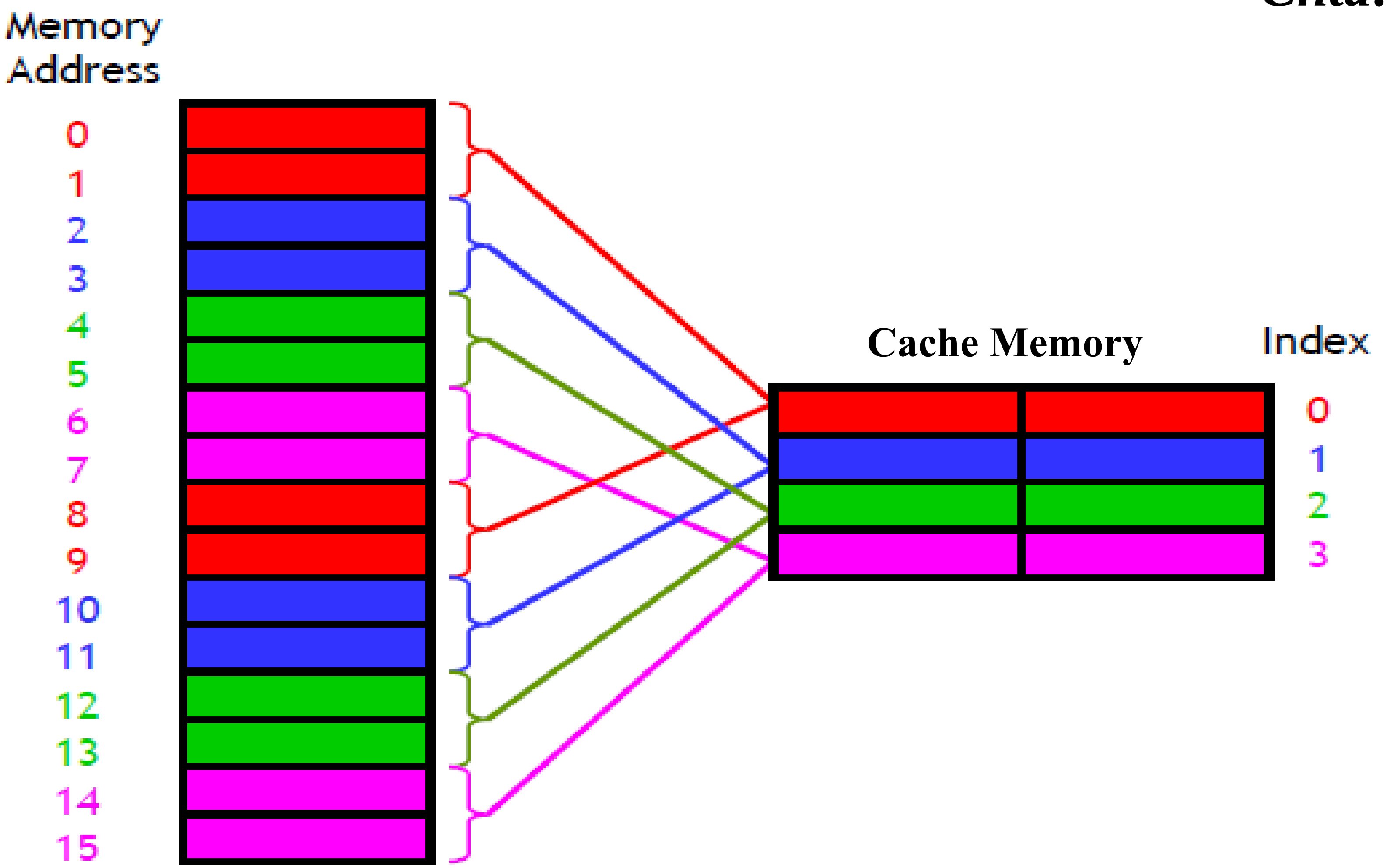


Index	Tag	Valid	Data
00	00	1	🟥
01	11	1	🟦
10	01	1	🟩
11	01	1	🟪

Tag	Index
2	2

CPU generated address

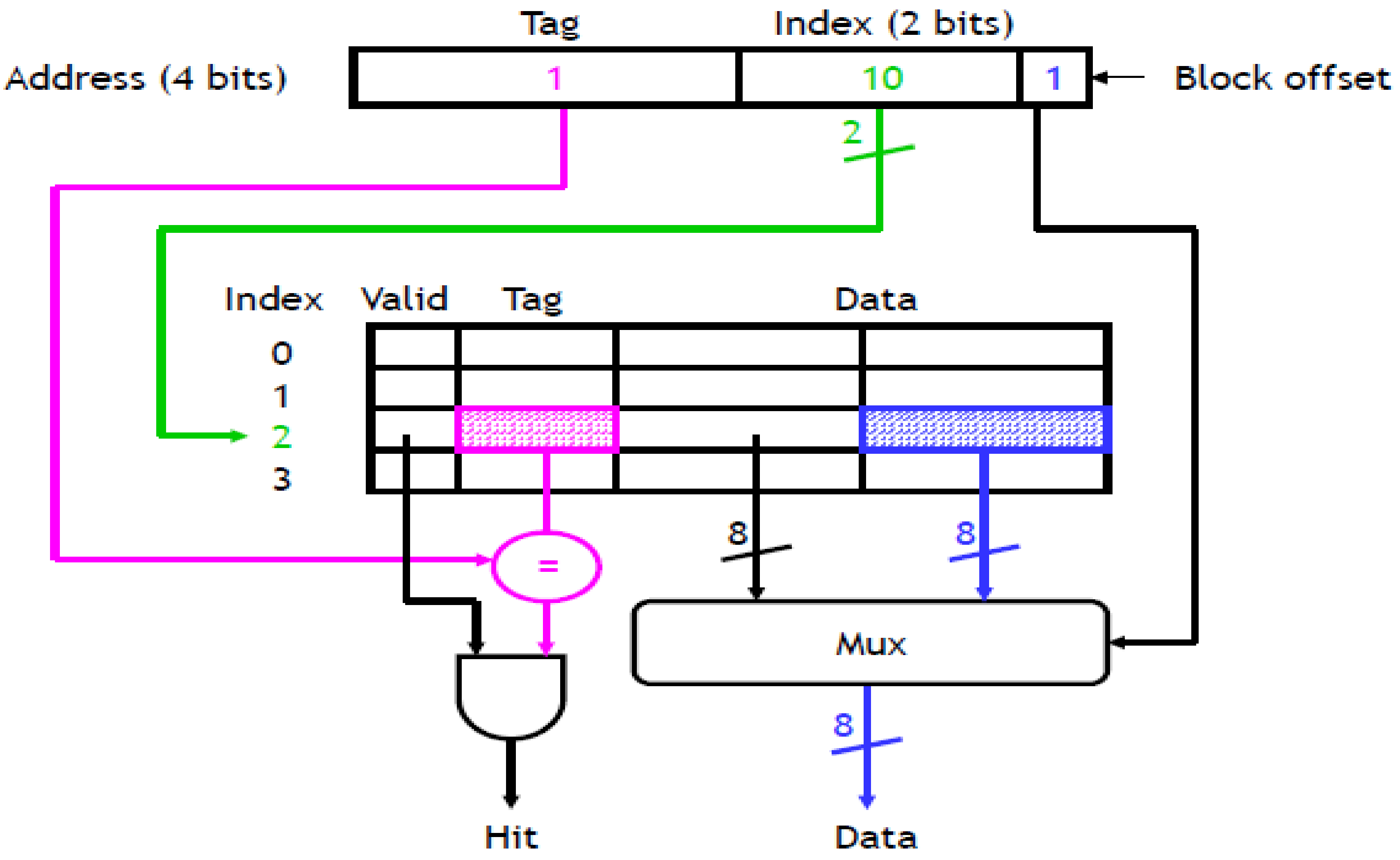
Cntd…



1	2	1
Tag	Block	Word/Byte

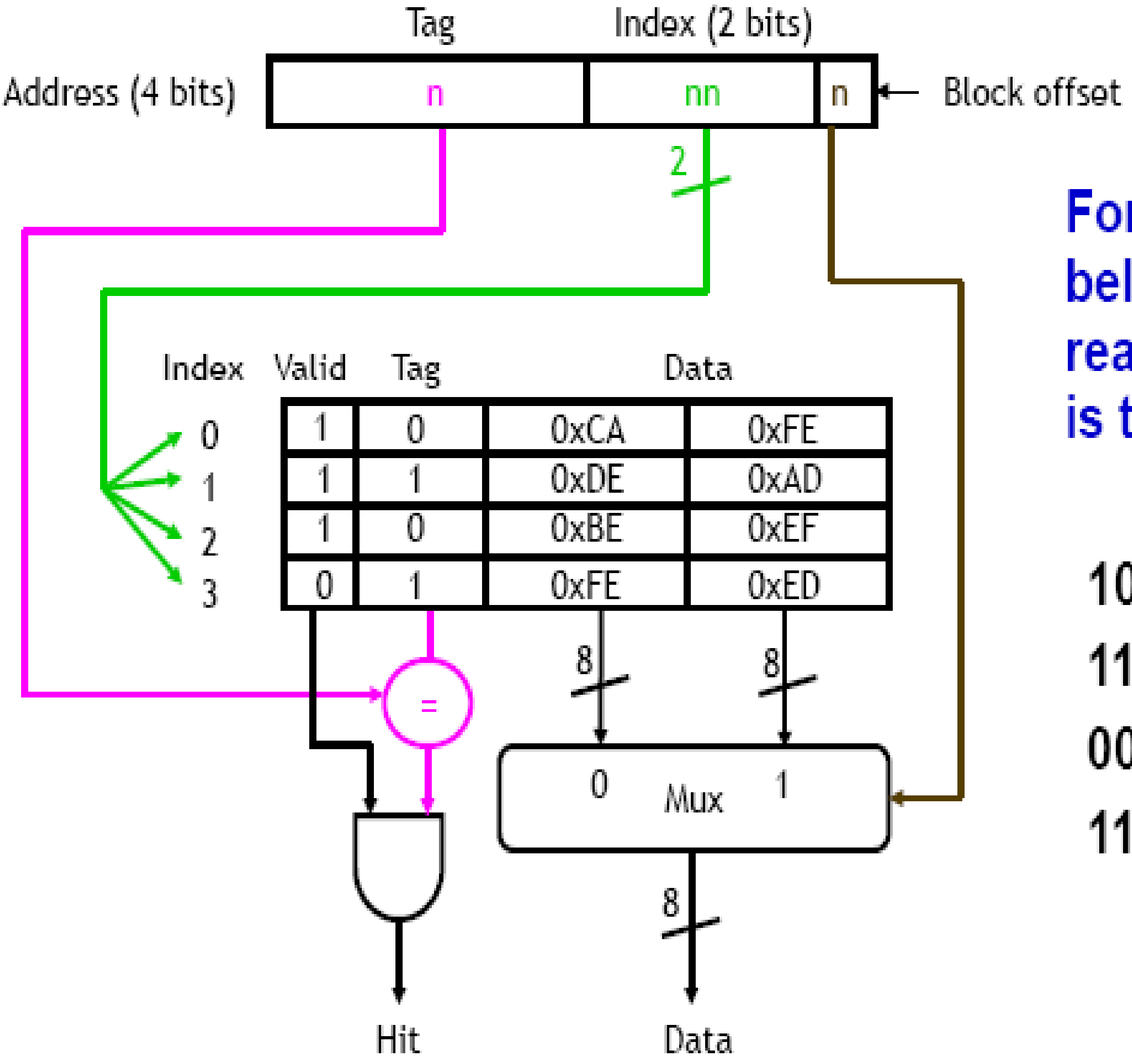
Cntd…

Example



Exercise

Cntd…



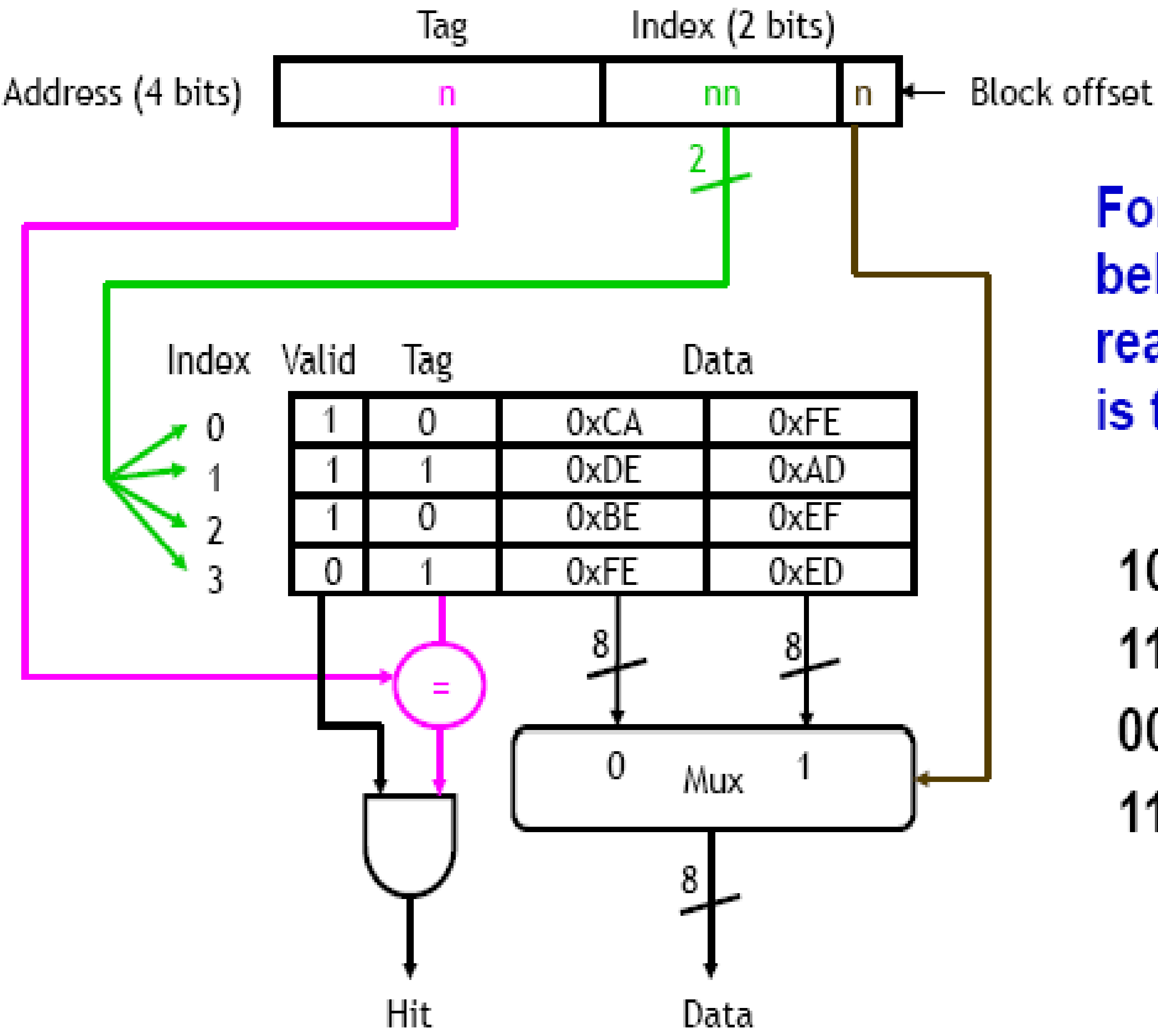
Index	Valid	Tag	Data	
0	1	0	0xCA	0xFE
1	1	1	0xDE	0xAD
2	1	0	0xBE	0xEF
3	0	1	0xFE	0xED

For the addresses below, what byte is read from the cache (or is there a miss)?

1010
1110
0001
1101

Cntd…

Exercise (Solution)



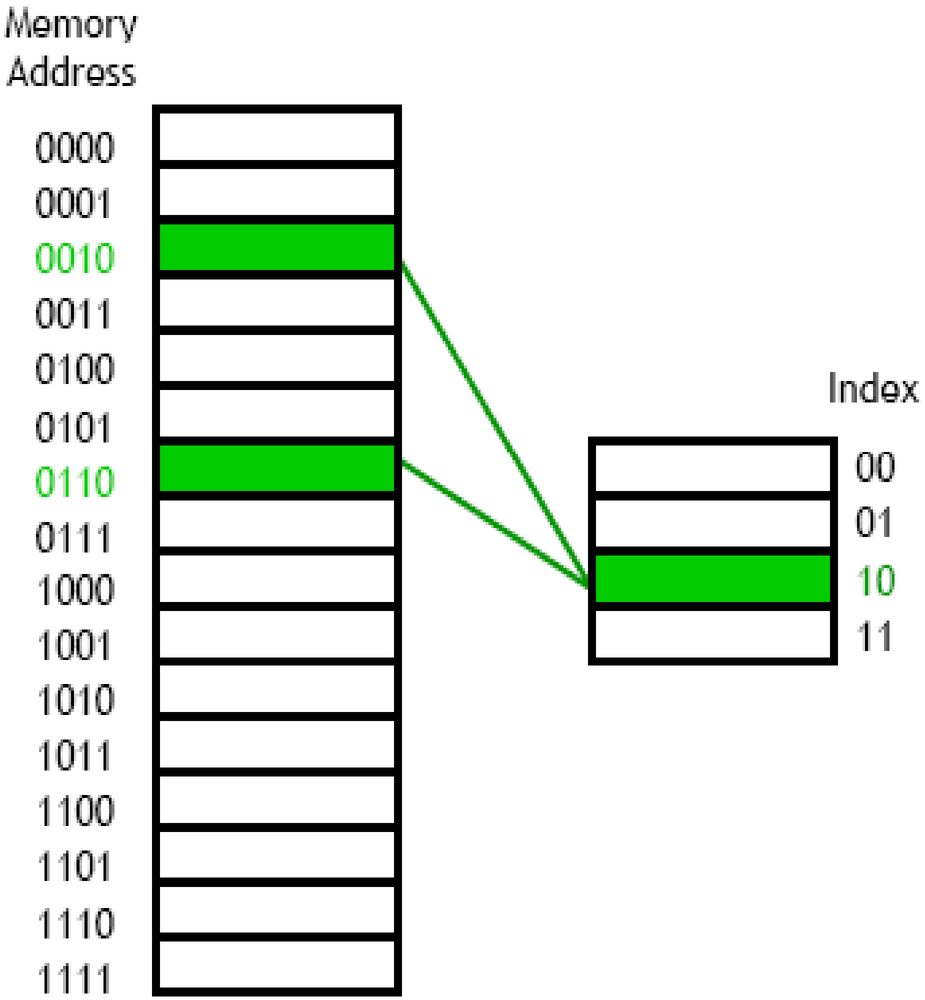
Index	Valid	Tag	Data	
0	1	0	0xCA	0xFE
1	1	1	0xDE	0xAD
2	1	0	0xBE	0xEF
3	0	1	0xFE	0xED

For the addresses below, what byte is read from the cache (or is there a miss)?

1010	(0xDE)
1110	(miss, invalid)
0001	(0xFE)
1101	(miss, bad tag)

Disadvantage of Direct-mapped Cache

- The direct-mapped cache is easy: indices and offsets can be computed with bit operators or simple arithmetic, because each memory address belongs in exactly one block.
- However, this isn't really flexible. If a program uses addresses 2, 6, 2, 6, 2, ..., then each access will result in a cache miss and a load into cache block 2.
- This cache has four blocks, but direct mapping might not let us use all of them.
- This can result in more misses than we might like.



Associative Mapping

- A fully associative cache permits data to be stored in any cache block, instead of forcing each memory address into one particular block. –When data is fetched from memory, it can be placed in any unused block of the cache.

  –This way we'll never have a conflict between two or more memory addresses which map to a single cache block.

- In the previous example, we might put memory address 2 in cache block 2, and address 6 in block 3. Then subsequent repeated accesses to 2 and 6 would all be hits instead of misses.

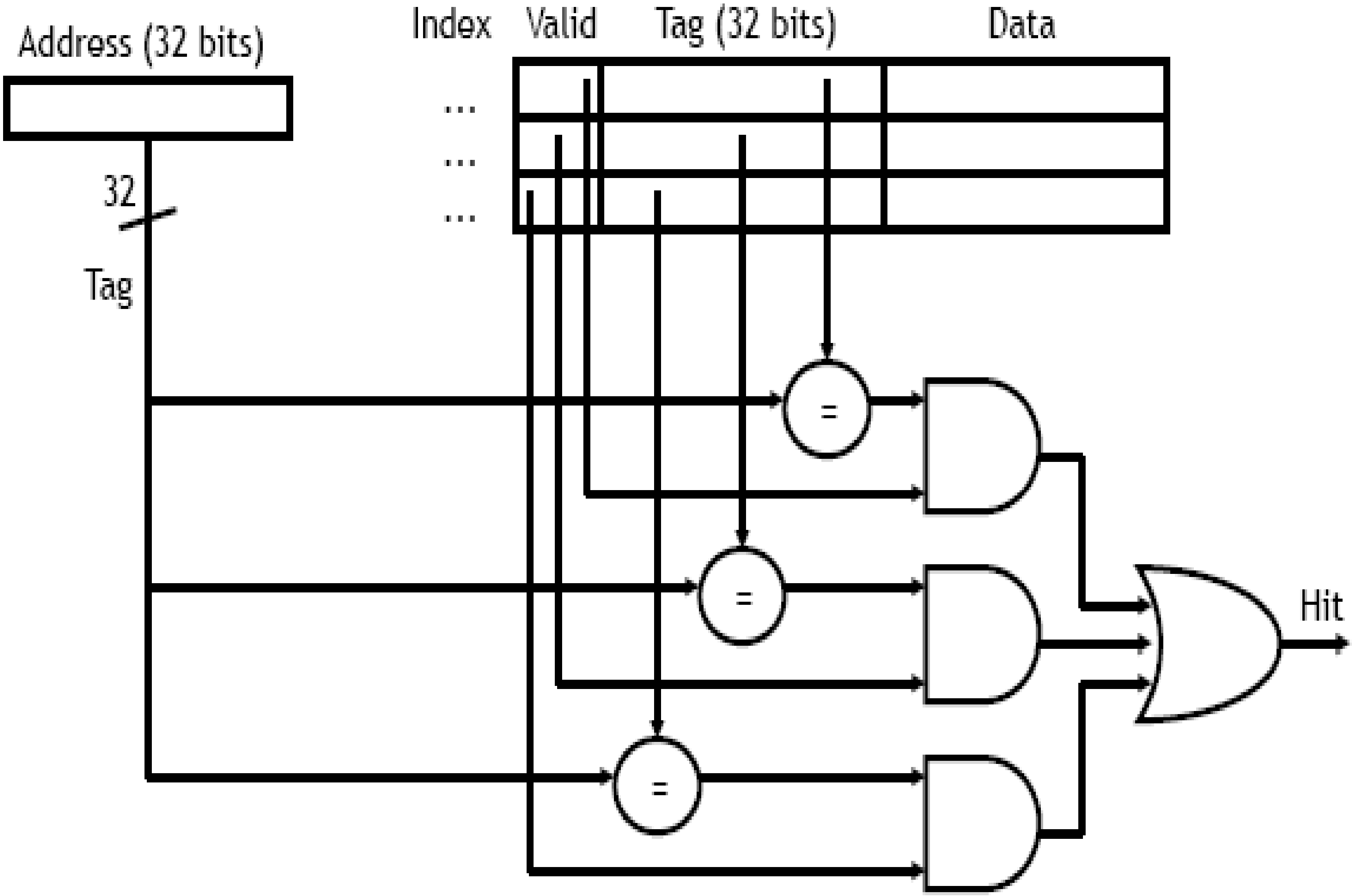
- If all the blocks are already in use, it's usually best to replace the ***least recently used one***, assuming that if it hasn't been used it in a while, it won't be needed again anytime soon.

- CPU generated address is divided into 2 fields –

Tag	Word

CPU generated address

Cntd…



Disadvantage of Fully Associative Mapping

- A fully associative cache is expensive to implement.

  –Because there is no index field in the address anymore, the *entire block address must be used as the tag, increasing the total cache size.*

  –Data could be anywhere in the cache, so we must check the tag of *every cache block. That's a lot of comparators!*

Set-Associative Mapping

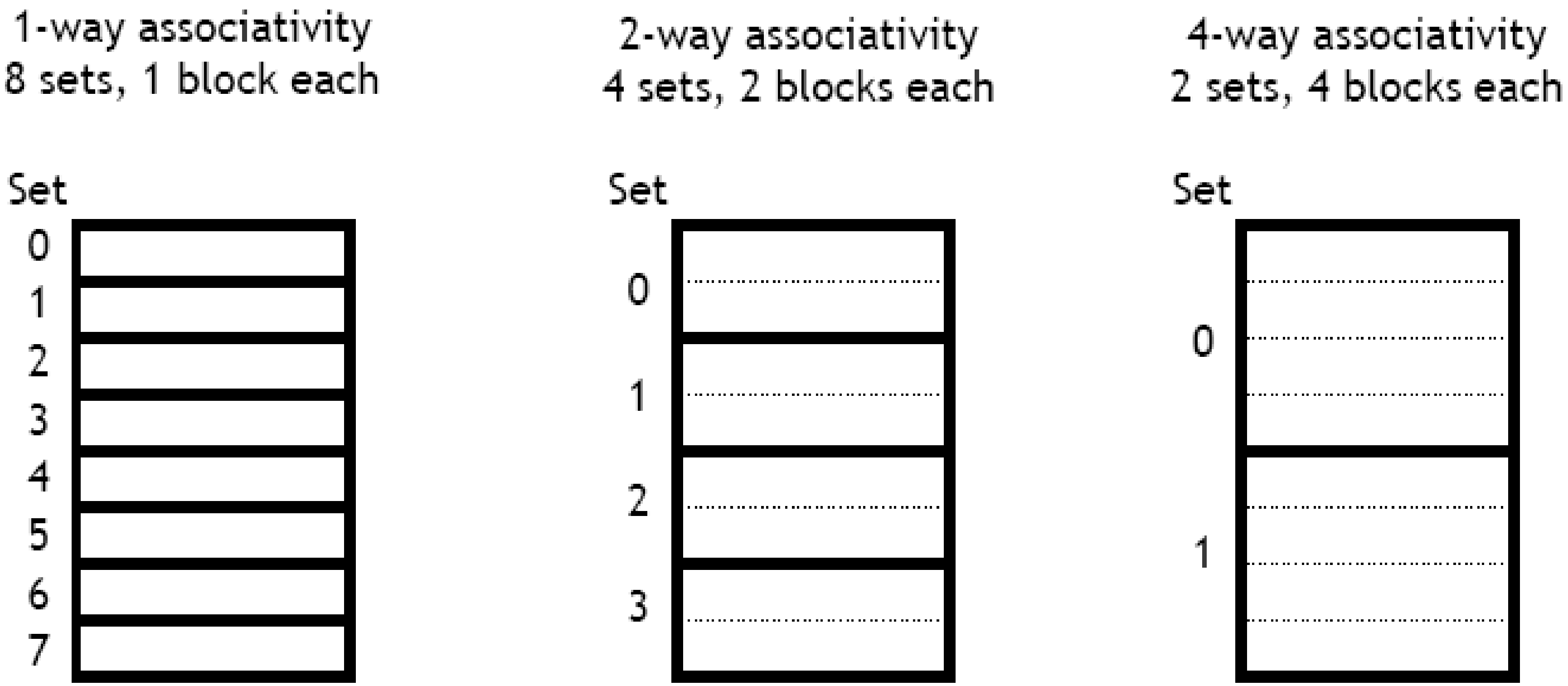
- Combination of direct & associative mapping.
- Cache memory is divided into a no. of sets.
- Each set consists of a no. of blocks.
- Each memory address maps to exactly one set in the cache, but data may be placed in any block within that set.
- If ***k*** no. of blocks within a set, then it is called ***k-way set-associative.***
- CPU generated address is divided into 3 fields –

Tag	**Set**	**Word**

CPU generated address

- Higher cost than direct-mapped cache.
- Improved hit ratio than direct & associative mapping.

Cntd…

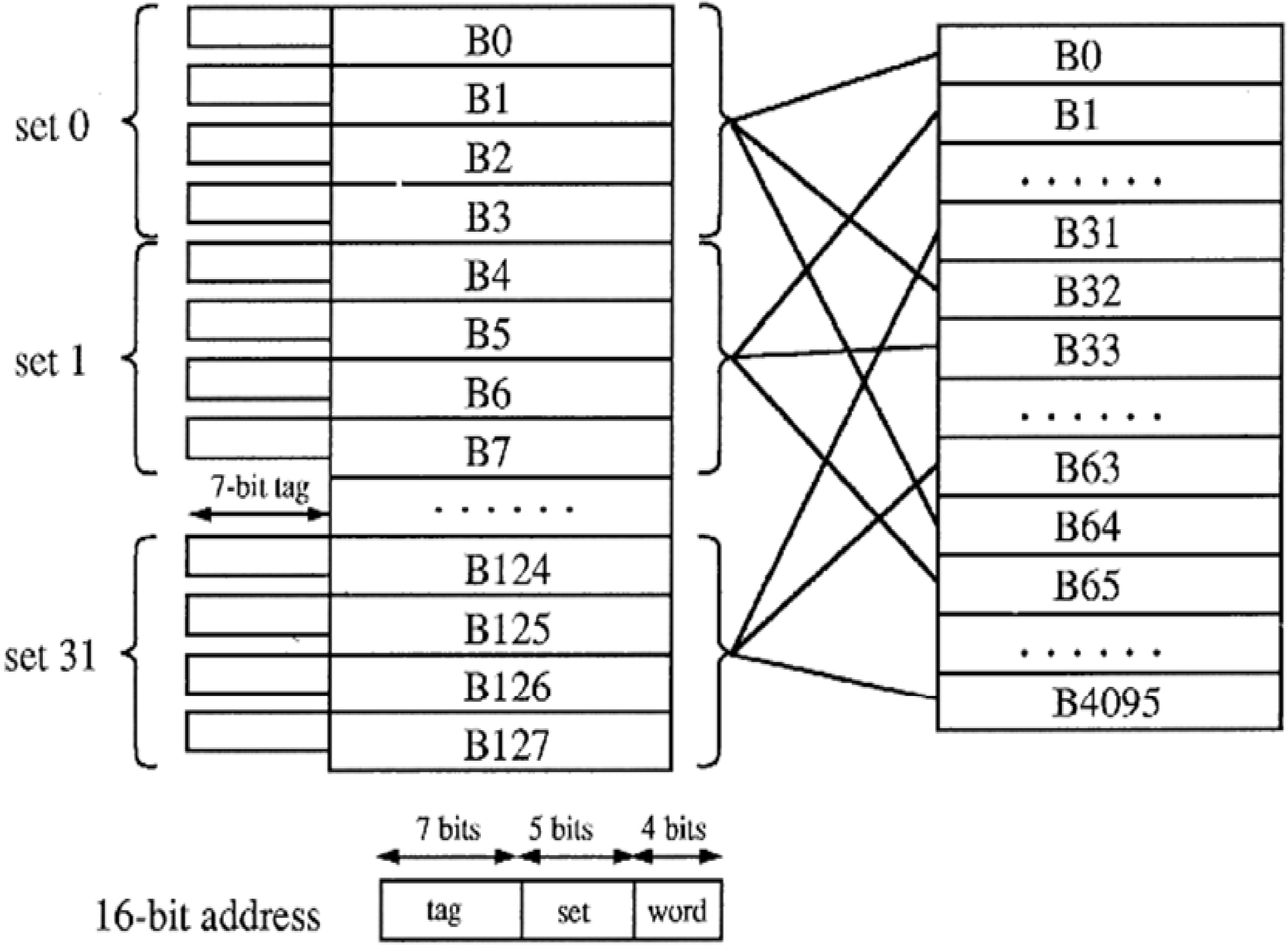


Several possible organizations of an eight-block cache

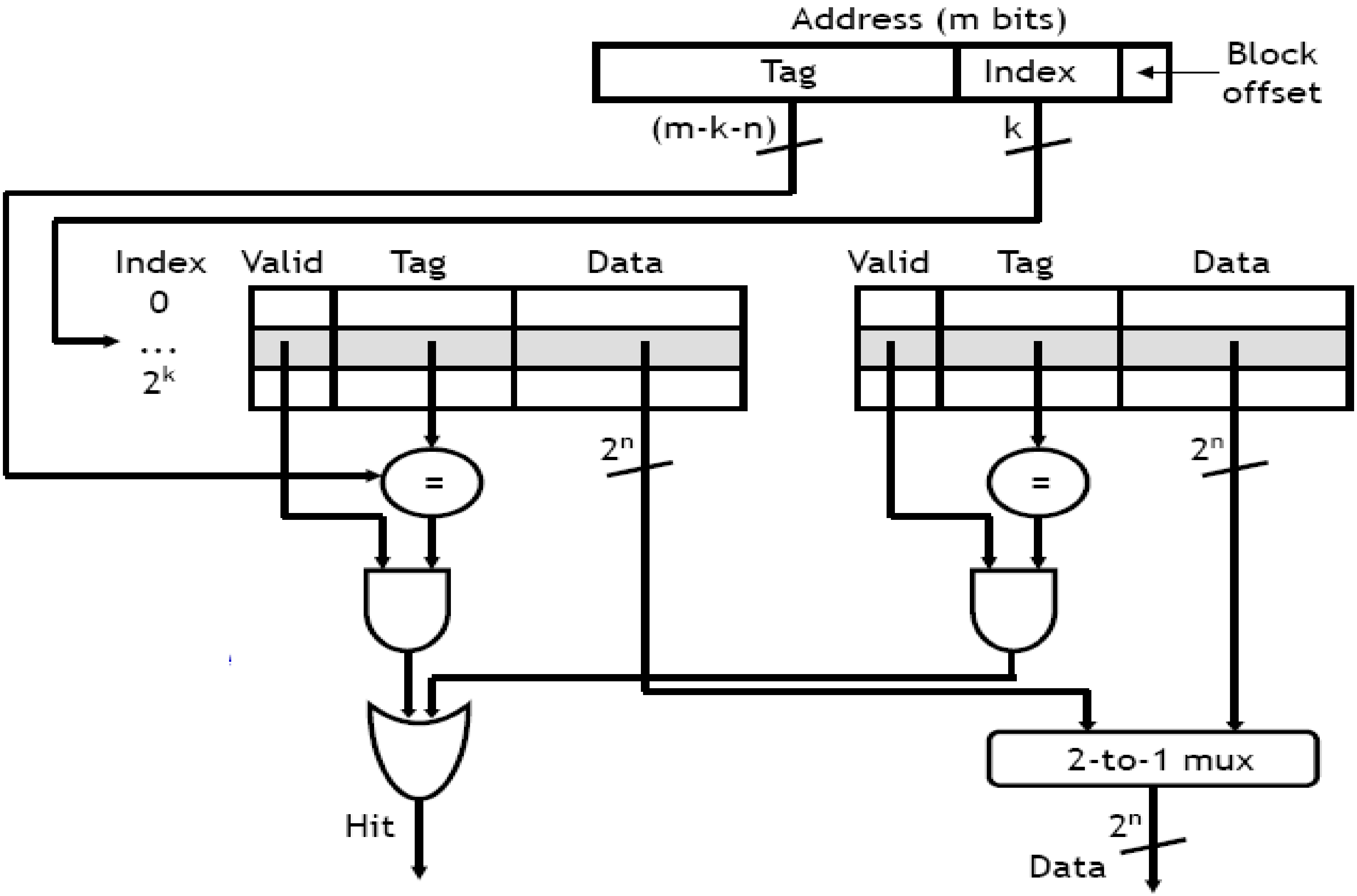
<u>NOTE</u>: ***Block Offset = Memory Address mod 2^n***
Block Address = Memory Address / 2^n
Set Index = Block Address mod 2^s

Where , n => no. of bits to represent offset field
s => no. of bits to represent set field

Cntd…


set 0
set 1
set 31
B0
B1
B2
B3
B4
B5
B6
B7
. . . .
B124
B125
B126
B127
7-bit tag
B0
B1
. . . .
B31
B32
B33
. . . .
B63
B64
B65
. . . .
B4095
7 bits
5 bits
4 bits
16-bit address
tag
set
word

2-Way Set-Associative



Exercise

- Consider the following specifications -

  1. The capacity of Main Memory = 256MB
  2. The capacity of Cache Memory = 1MB
  3. Block Size = 128 Bytes
  4. A set contains 8 blocks (8-way set associative)

Determine the size of the sub-fields (in bits) in the address for –

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

Page Replacement Algorithms

- When cache miss occurs, a new data from main memory needs to be placed over old data in the selected location of cache.
- When cache is full and we need to insert a new data replacing old data of a location in cache.

  - Page replacement algorithms are applicable for both the cases.

***1. FIFO* :** The word that has been in the cache for a long time should be replaced first (the word which is entered in the cache first).

***2. LRU* :** The word that has been used by the CPU for a minimum no. of times in the recent past should be replaced first.

3. LFU: Least frequently used

4. Random Replacement: Random selection (no restriction).

NOTE: Direct mapping has no replacement strategies since the locations are fixed.

Q+A

1. Which of the following is used to store the initial program needed to start the computer?

   a) SRAM b) DRAM c) RAM d) ROM

2. Cache is placed between ………. & ……….

   a) RAM , ROM b) Primary memory, Secondary memory

   c) CPU, Main Memory d) Registers, CPU

3. A large no. of memory requests are found in cache because of ….

   a) High Hit Ratio b) Locality of reference

   c) less no. of miss d) none of these

4. How many 128×8 RAM chips are required to build 2048 Bytes of memory system.

   a) 16 b) 32 c) 64 d) 128

5. The method of updating the main memory as soon as a word is removed from the cache is called ……….

   a) Write-Back b) Write-Through

   c) Cache-Write d) Protected-Write

6. For each update of cache there will be an update for main memory referred to ...

   a) Write-Back b) Write-Through

   c) Cache-Write d) Protected-Write

7. Two or more requests for same location in the cache results in ……..

   a) compulsory miss b) capacity miss

   c) conflict miss d) none of these

8. Vertical expansion of a memory system means …….

   a) increasing the no. of locations b) increasing the word length

   c) Both a & b d) none of these

9. The average time required to reach a storage location in memory and obtain its contents is called ……..

   a) Latency time. b) Access time.

   c) Turnaround time. d) Response time.

10. Memory unit accessed by content is called ……….

    a) Read only memory b) Programmable Memory

    c) Virtual Memory d) Associative Memory

11. The minimum time delay between two successive memory read operations is …

a) Cycle time b) Latency

c) Delay d) None of these

12. The cells in a row are connected to a common line called ………

a) Work line b) Word line

c) Length line d) Principle diagonal

13. The cells in each column are connected to ………

a) Word line b) Data line

c) Read line d) Sense/ Write line

14. The word line is driven by the …………

a) Chip select b) Address decoder

c) Data line d) Control line

15. A 16 X 8 organisation of memory cells, can store upto _____.

a) 256 bits b) 1024 bits

c) 512 bits d) 128 bits

16. A memory organization that can hold up to 1024 bits and has a minimum of 10 address lines can be organized into …………

a) 128 X 8 b) 256 X 4

c) 512 X 2 d) 1024 X 1

17. The number of external connections required in 16 X 8 memory organization is -

a) 14 b) 15

c) 16 d) 17

18. The SRAM's are basically used as …….

a) Registers b) Caches

c) Primary Memory d) Secondary Memory

19. The contents of the EPROM are erased by

a) Overcharging the chip. b) Exposing the chip to UV rays.

c) Exposing the chip to IR rays. d) Discharging the Chip.

20. The fastest data access is provided using ……….

a) Caches b) DRAM's

c) SRAM's d) Registers

21. The memory which is used to store the copy of data or instructions stored in larger memories, inside the CPU is called ………….

a) Level 1 cache b) Level 2 cache

c) Main Memory d) Auxiliary Memory

22. The larger memory placed between the primary cache and the memory is called-

a) Level 1 cache b) Level 2 cache

c) Main Memory d) Auxiliary Memory

23. The next level of memory hierarchy after the L2 cache is ………

a) Secondary storage b) L1 cache
c) Main memory d) Register

24. The last on the hierarchy scale of memory devices is ……

a) Secondary storage b) Cache
c) Main memory d) Register

25. The reason for the implementation of the cache memory is ……..

a) To increase the internal memory of the system
b) The difference in speeds of operation of the processor and memory
c) To reduce the memory access and cycle time
d) All of the above

26. The effectiveness of the cache memory is based on the property of ………

a) Locality of reference b) Memory localization
c) Memory size d) None of the above

27. The temporal aspect of the locality of reference means

a) That the recently executed instruction wont be executed soon
b) That the recently executed instruction is temporarily not referenced
c) That the recently executed instruction will be executed soon again
d) None of the above

28. The spatial aspect of the locality of reference means

a) That the recently executed instruction is executed again next

b) That the recently executed wont be executed again

c) That the instruction executed will be executed at a later time

d) That the instruction in close proximity of the instruction executed will be executed in future

29. The correspondence between the main memory blocks and those in the cache is given by ……….

a) Hash function

b) Mapping function

c) Local function

d) Assign function

30. The algorithm to remove and place new contents into the cache is called ….

a) Replacement algorithm

b) Renewal algorithm

c) Updation

d) None of the above

31. The write-through procedure is used

a) To write onto the memory directly

b) To write and read from memory simultaneously

c) To write directly on the memory and the cache simultaneously

d) None of the above

32. The bit used to signify that the cache location is updated is ……..

a) Dirty bit
b) Update bit
c) Reference bit
d) Flag bit

33. During a write operation if the required block is not present in the cache then ______ occurs.

a) Write latency
b) Write hit
c) Write delay
d) Write miss

34. In ________ protocol the information is directly written into main memory.

a) Write through
b) Write back
c) Write first
d) None of the above

35. The method of mapping the consecutive memory blocks to consecutive cache blocks is called ……….

a) Set associative
b) Associative
c) Direct
d) Indirect

36. While using the direct mapping technique, the higher order bits is used for -

a) Tag
b) Block
c) Word
d) Id

37. The technique of searching for a block by going through all the tags is ….

a) Linear search b) Binary search
c) Associative search d) None of the above

38. The number successful accesses to memory stated as a fraction is called as -

a) Hit rate b) Miss rate
c) Success rate d) Access rate

39. he number failed attempts to access memory, stated in the form of fraction is called as ………….

a) Hit rate b) Miss rate
c) Failure rate d) Delay rate

40. In associative mapping during LRU, the counter of the new block is set to ’0′ and all the others are incremented by one, when _____ occurs.

a) Delay b) Miss
c) Hit d) Delayed hit

41. In LRU, the refrenced blocks counter is set to’0′ and that of the previous blocks are incremented by one and others remain same, in case of …..

a) Hit b) Miss
c) Delay d) None of the above

42. The extra time needed to bring the data into memory in case of a miss is called as _____.

a) Delay
b) Propagation time
c) Miss penalty
d) None of the above

43. The main purpose of having memory hierarchy is to

a) Reduce access time.
b) Provide large capacity.
c) Reduce propagation time.
d) Both a and b.

44. The directly mapped cache requires no replacement algorithm.

a) True
b) False

45. The surroundings of the recently accessed block is called as ……….

a) Neighbourhood
b) Neighbour
c) Locality of reference
d) None of the above

46. The algorithm which replaces the block which has not been referenced for awhile is called ………

a) LRU
b) FIFO
c) Random replacement
d) All of these

References:

1. Computer System Architecture – Morris Mano
2. Computer Organization & Architecture – T. K. Ghosh