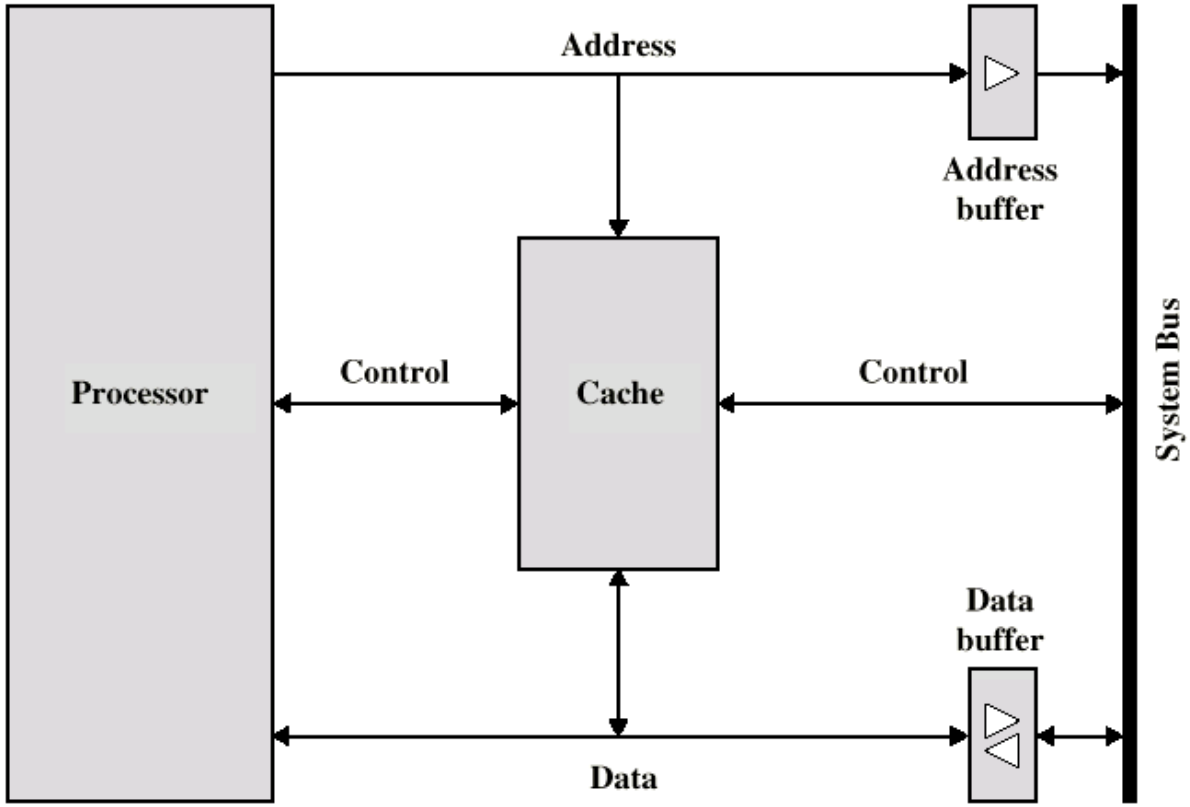
COMPUTER ORGANIZATION (IS F242)

LECT 37: CACHE MEMORY

Cache Design

- Cache access
  - Look through, Look aside
- Block Placement
  - Direct, Fully Associative, Set-Associative
- Block Identification
  - TAG, INDEX, OFFSET
- Block Replacement
  - LRU, PLRU, LFU, OPT, FIFO, RANDOM
- Write Policies
  - Write Through, Write Back, Write Buffer
- Coherency
  - Snooping, MESI

Typical Cache Organization

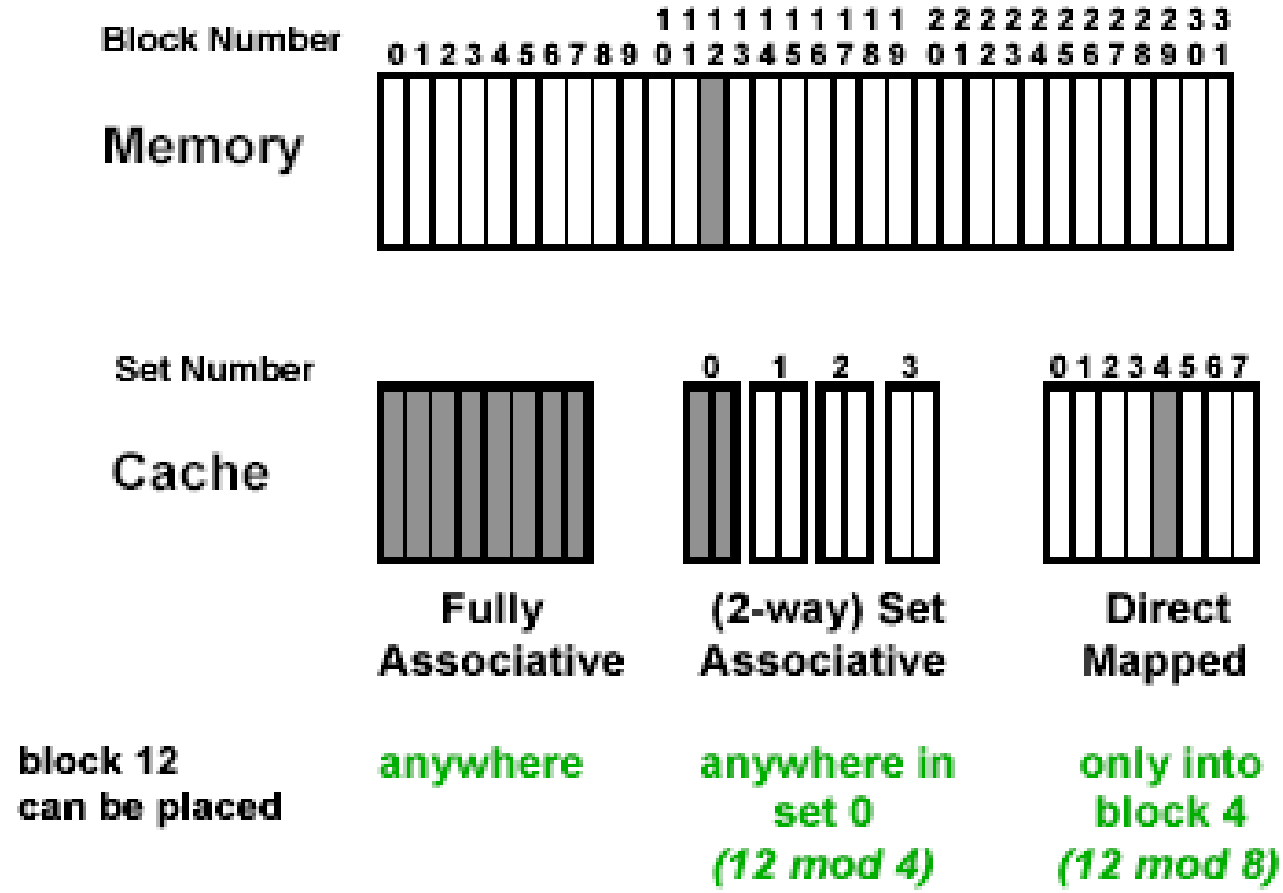


- **Look Through:** **Access Cache, if data not found access the lower level**
- **Look Aside:** **Request to Cache and its lower level at the same time. If cache access is hit then cancel the lower level request.**

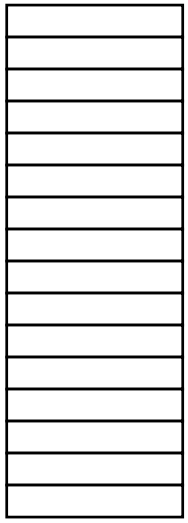
Cache memory: Mapping Function

- Cache memory size is smaller than main memory
- The correspondence between the main memory blocks in the cache lines is specified by a mapping function
- The processor doesn't need to know the existence of the cache!!!
- Mapping Functions
  - Direct
  - Fully Associative
  - Set Associative

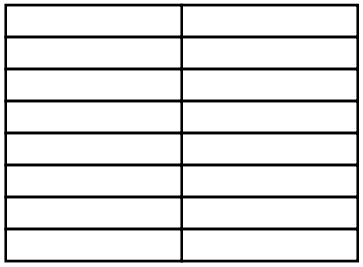
Placement Policy



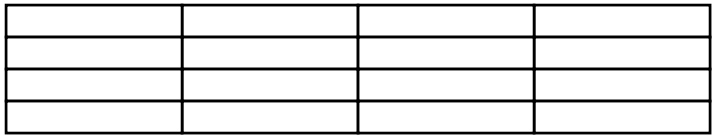
Cache Shapes



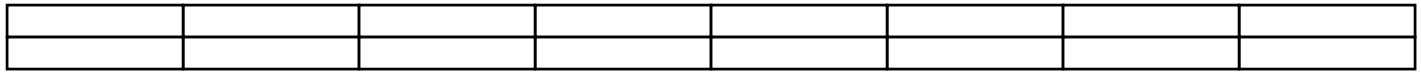
Direct-mapped
(A = 1, S = 16)



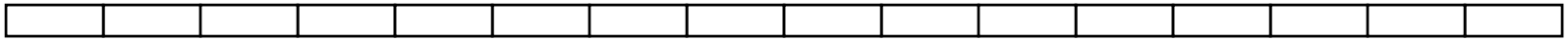
2-way set-associative
(A = 2, S = 8)



4-way set-associative
(A = 4, S = 4)

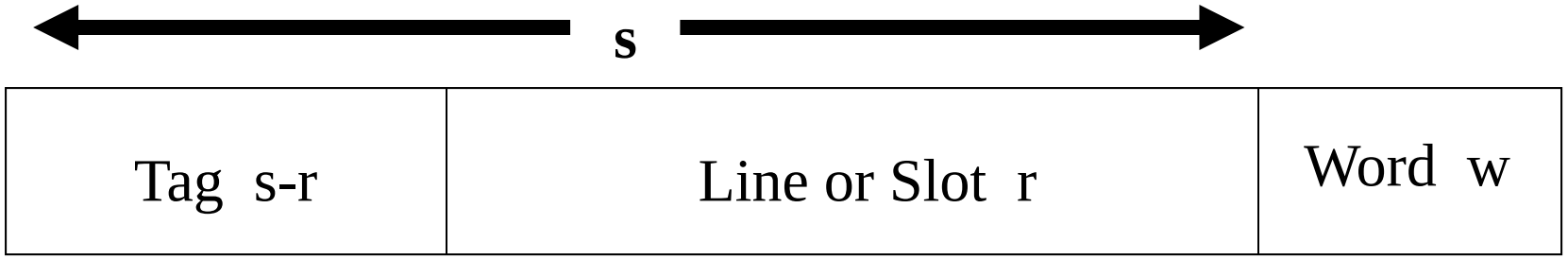


8-way set-associative
(A = 8, S = 2)



Fully associative
(A = 16, S = 1)

Address Mapping



- No two blocks in the same line have the same Tag field
- Check contents of cache by finding line and checking Tag

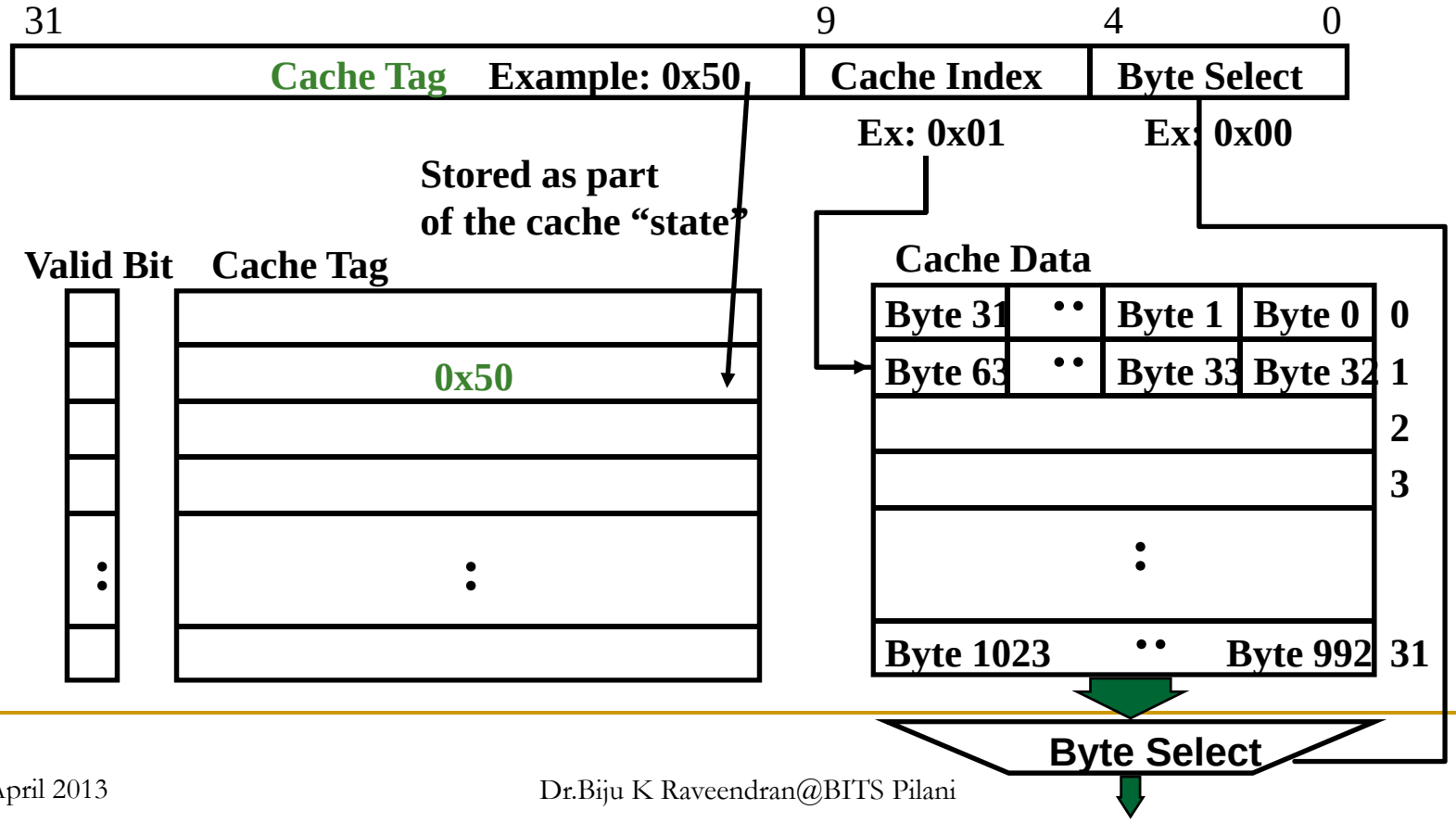
Direct Mapping

Cache Line Table

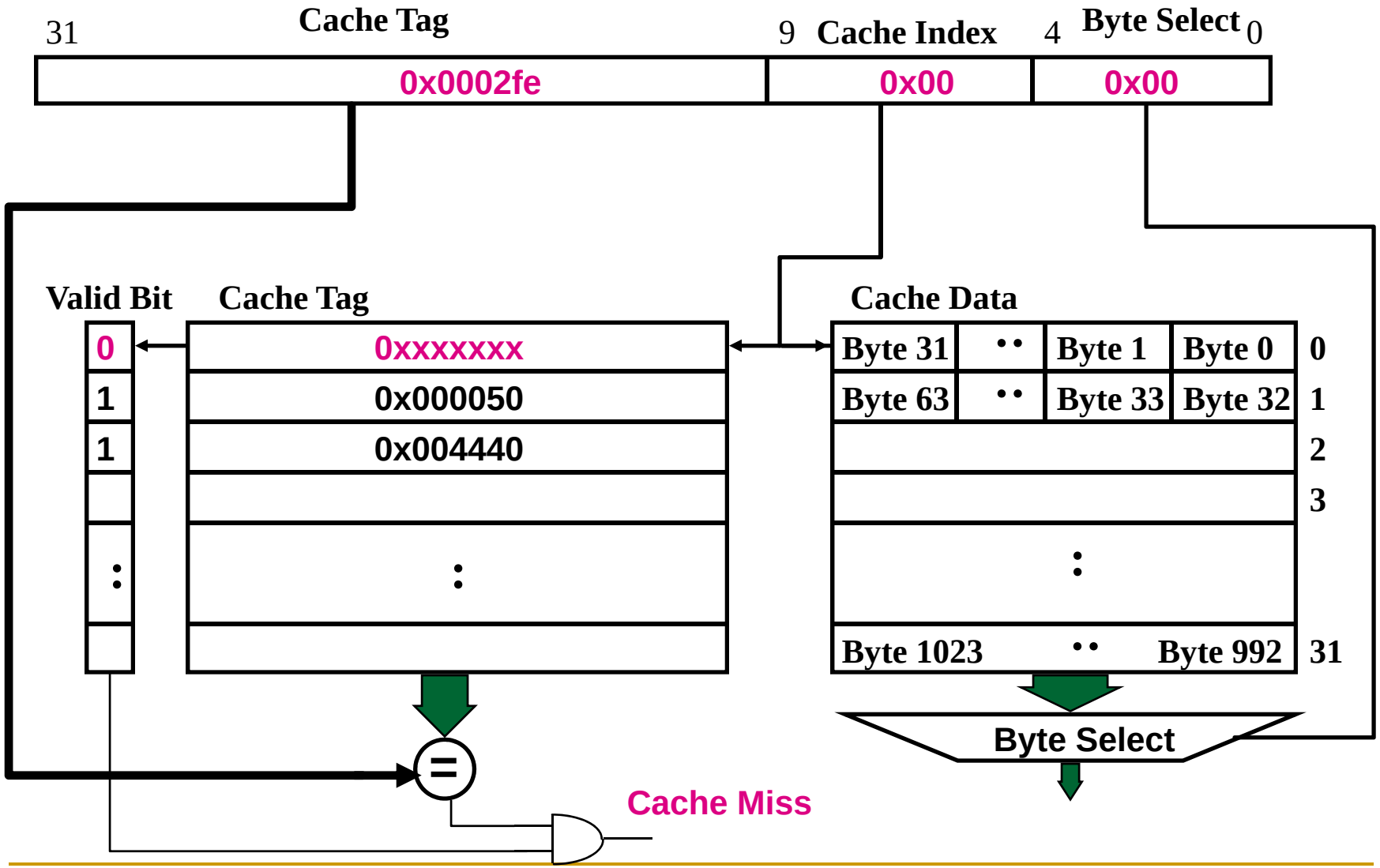
Cache line	Main Memory blocks held
0	0, m, 2m, 3m…2^s-m
1	1,m+1, 2m+1…2^s-m+1
…	
m-1	m-1, 2m-1,3m-1…2^s-1

1KB Direct map cache (32B block size)

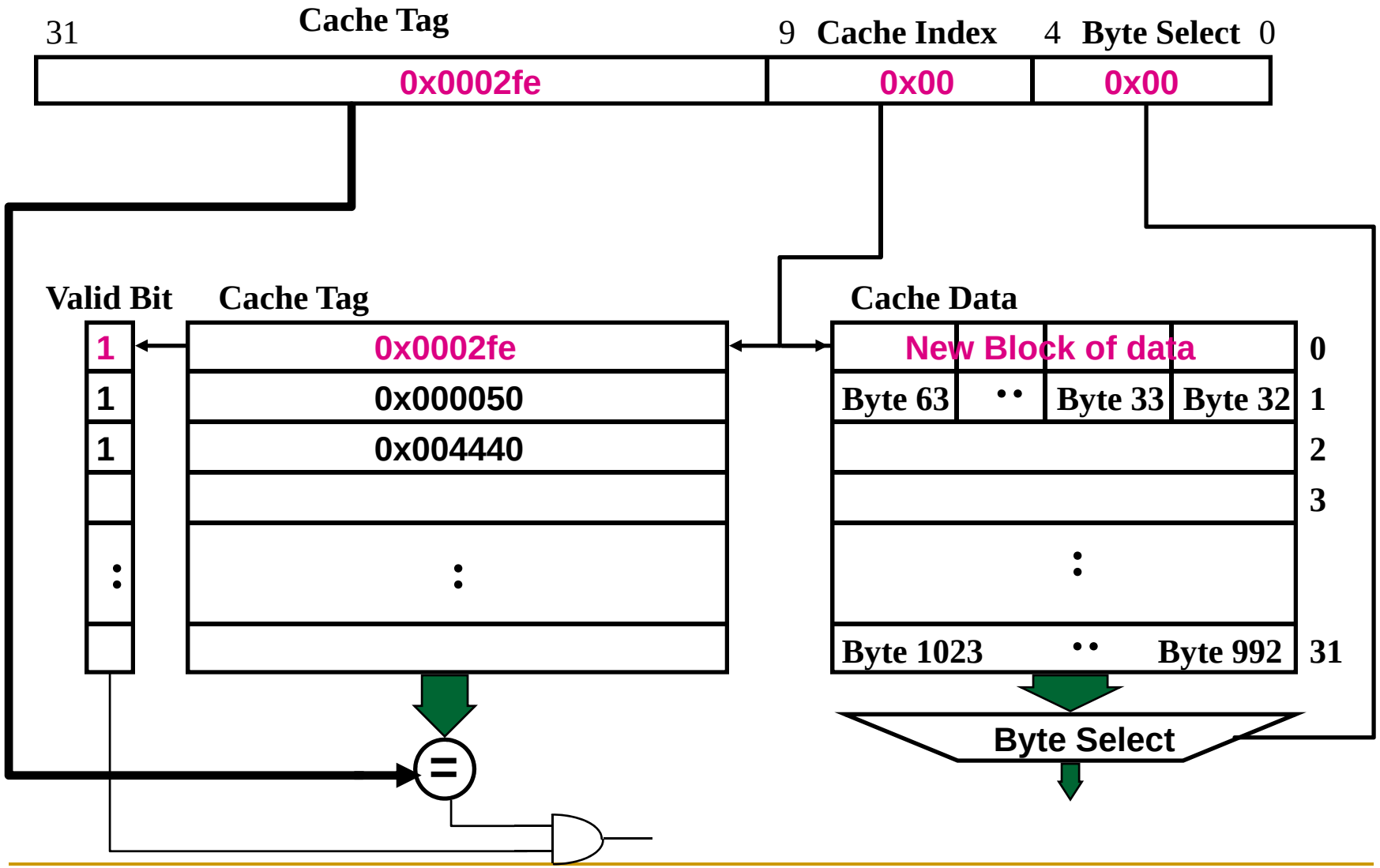
- For a 1024 (2^{10}) byte cache with 32-byte blocks
  - The uppermost 22 = (32 - 10) address bits are the tag
  - The lowest 5 address bits are the Byte Select (Block Size = 2^5)
  - The next 5 address bits (bit5 - bit9) are the Cache Index



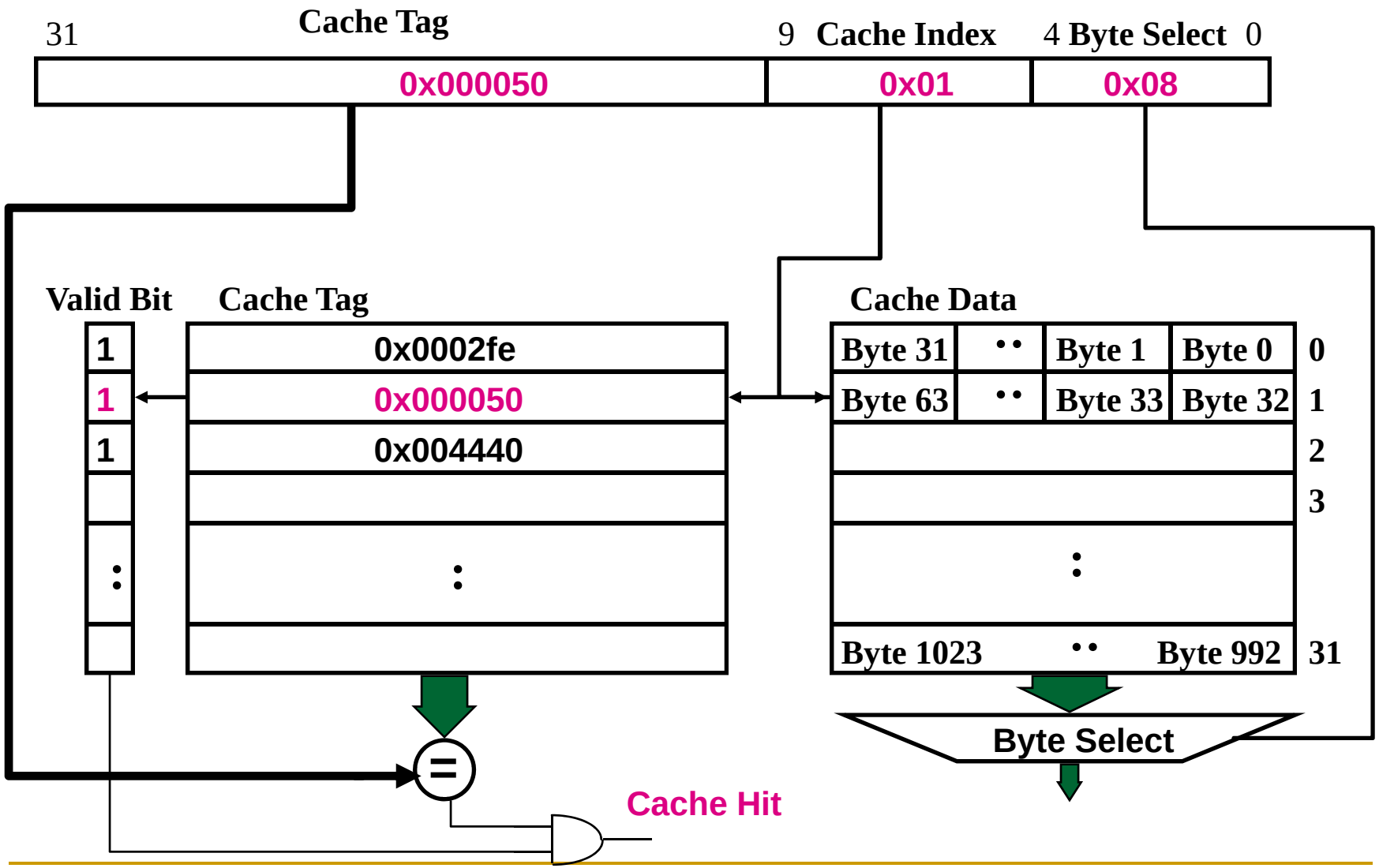
Cache Miss; Empty Block



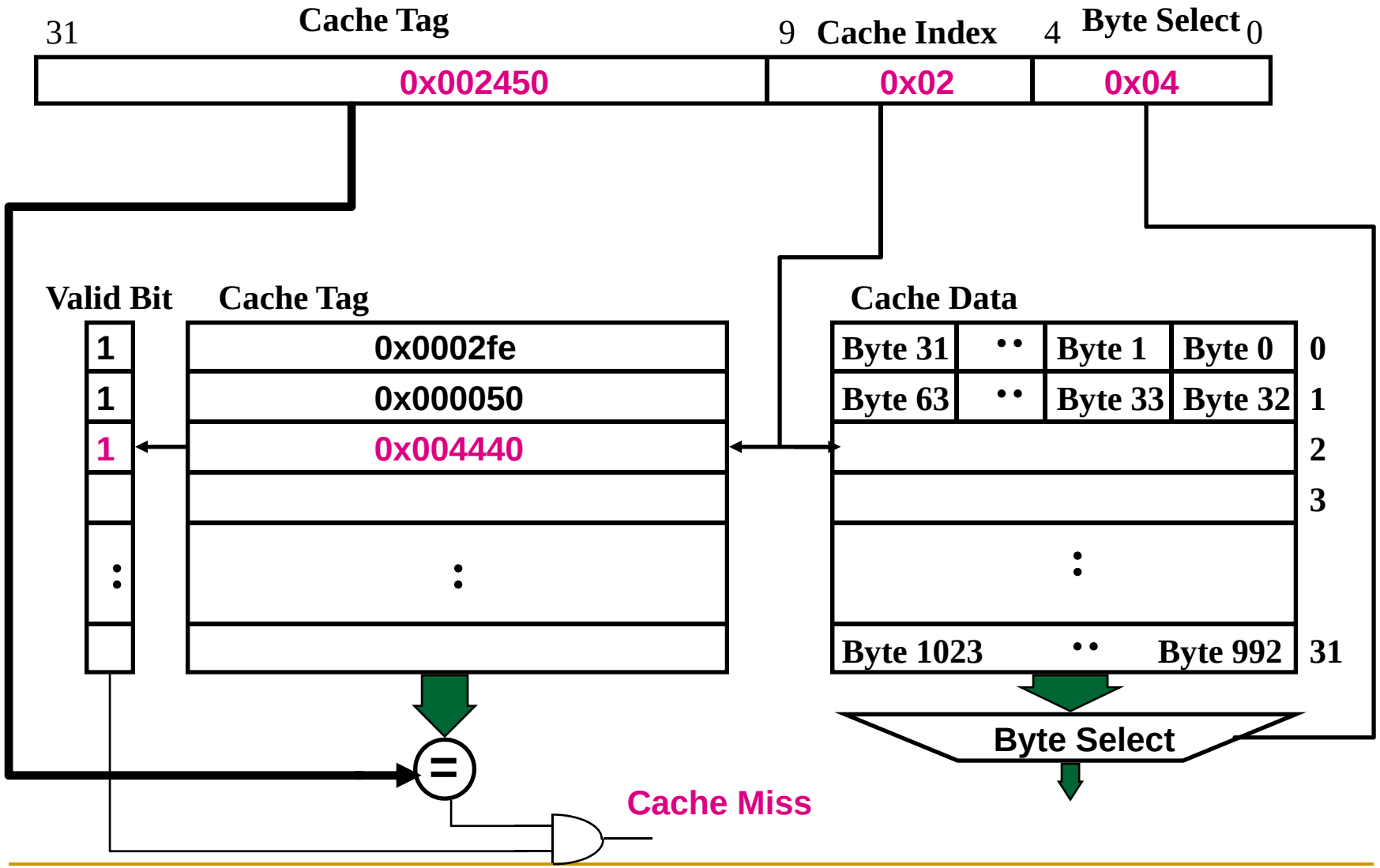
Read in Data


31
Cache Tag
9
Cache Index
4
Byte Select
0
0x0002fe
0x00
0x00
Valid Bit
Cache Tag
Cache Data
1
0x0002fe
New Block of data
0
1
0x000050
Byte 63
Byte 33
Byte 32
1
1
0x004440
2
3
Byte 1023
Byte 992
31
Byte Select
=

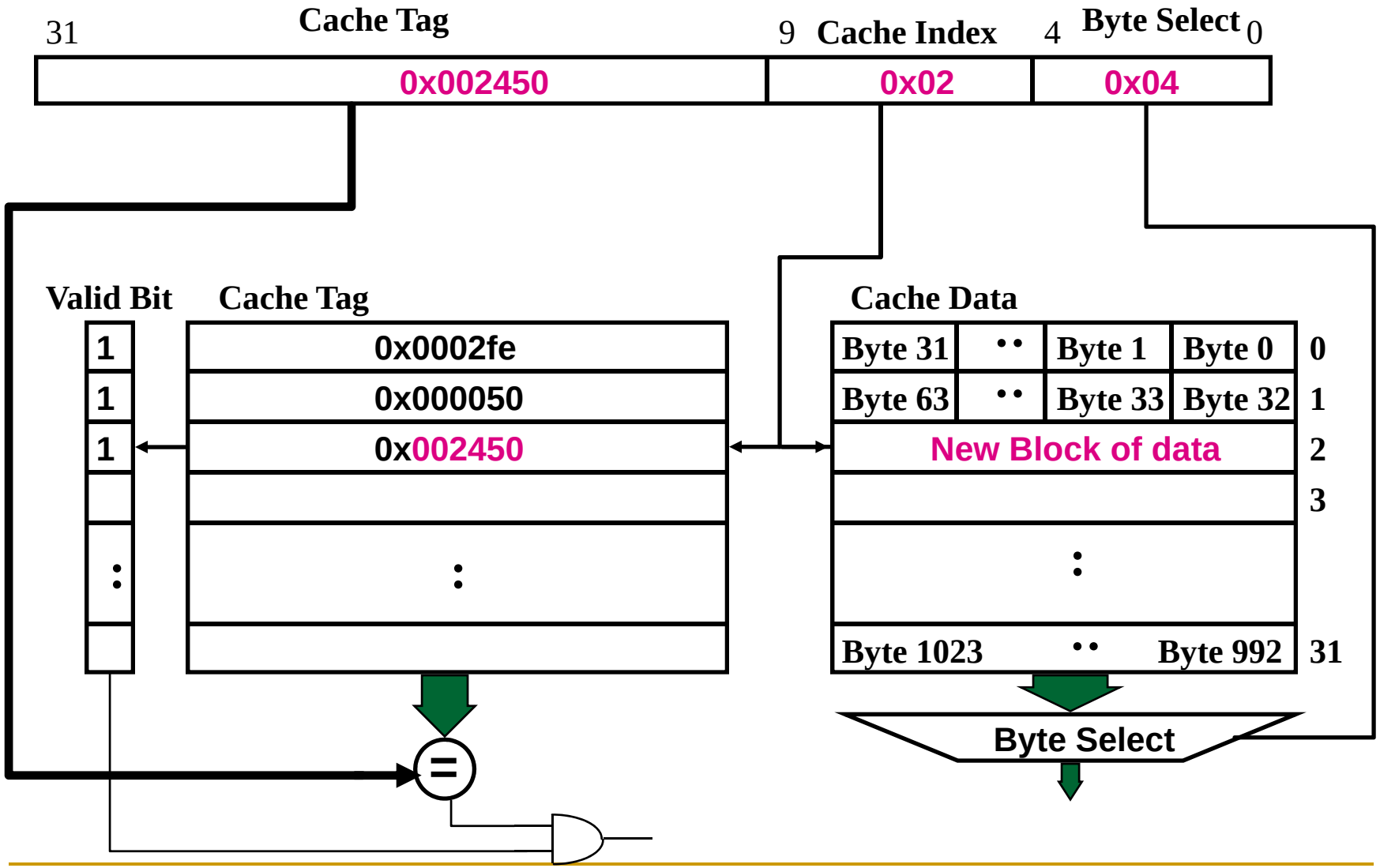
Cache Hit


31
Cache Tag
9
Cache Index
4
Byte Select
0
0x000050
0x01
0x08
Valid Bit
Cache Tag
Cache Data
1
0x0002fe
Byte 31
Byte 1
Byte 0
0
1
0x000050
Byte 63
Byte 33
Byte 32
1
1
0x004440
2
3
Byte 1023
Byte 992
31
Byte Select
=
Cache Hit

Cache Miss; Incorrect Block



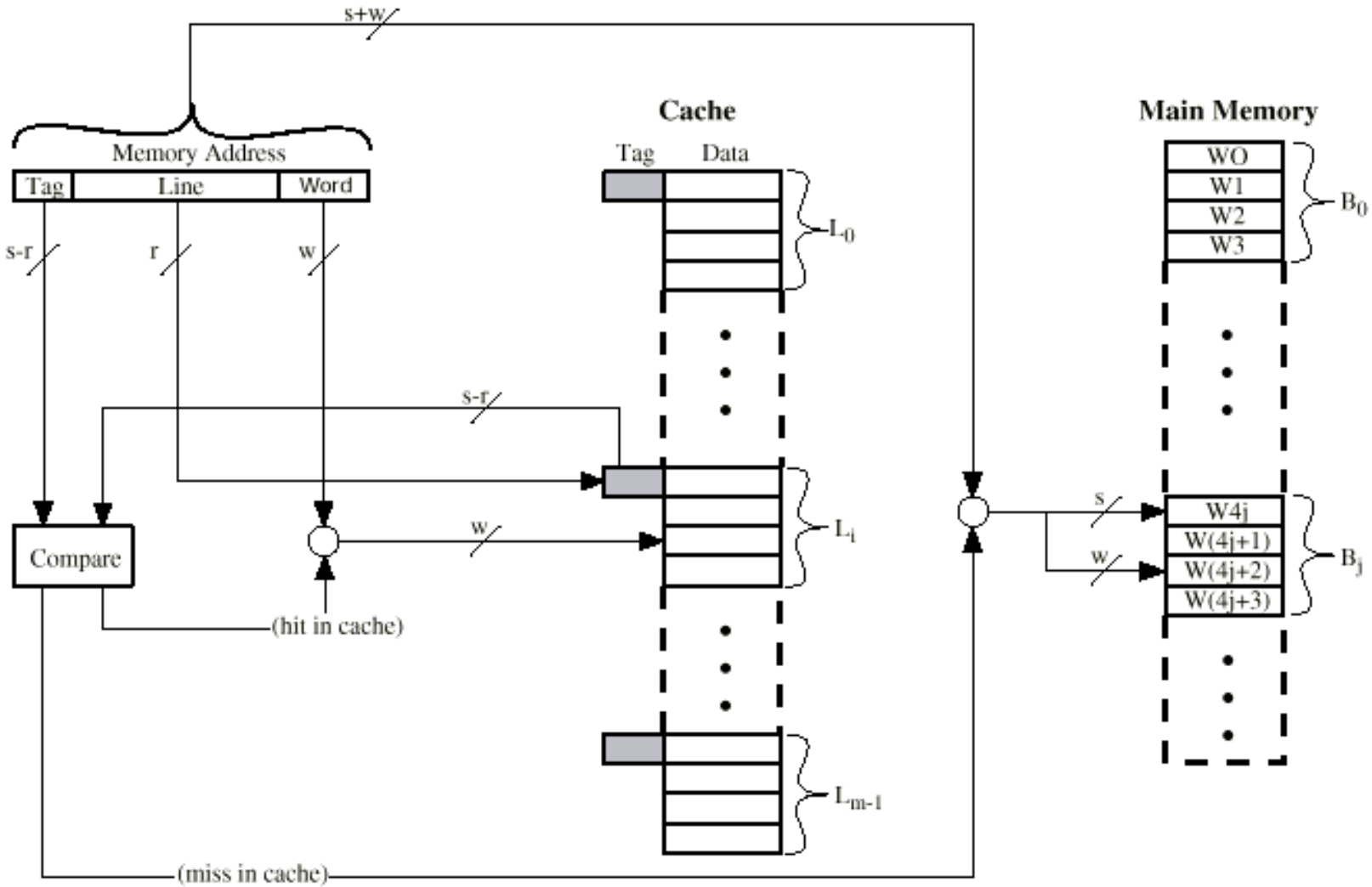
Replace Block



Direct Mapping Summary

- Address length = (s + w) bits
- Number of addressable units = 2^{s+w} words or bytes
- Block size = line size = 2^{w} words or bytes
- Number of blocks in main memory = $2^{s+ w}/2^{w} = 2^{s}$
- Number of lines in cache = m = 2^{r}
- Size of tag = (s – r) bits
- Mapping Function
  - J^{th} Block of the main memory maps to i^{th} cache line
    - I = J modulo M (M = number of cache lines)

Direct Mapping Cache Organization



Direct Mapping pros & cons

- Simple
- Inexpensive
- Fixed location for given block
  - If a program accesses 2 blocks that map to the same line repeatedly, cache misses (conflict misses) are very high