Master of Computer Applications

CAPOL403R01: Computer Organization & Architecture

Unit III: Lecture 3 – Part 2
Associative & Set associative mappings

Dr. D. MURALIDHARAN
School of Computing
SASTRA Deemed to be University

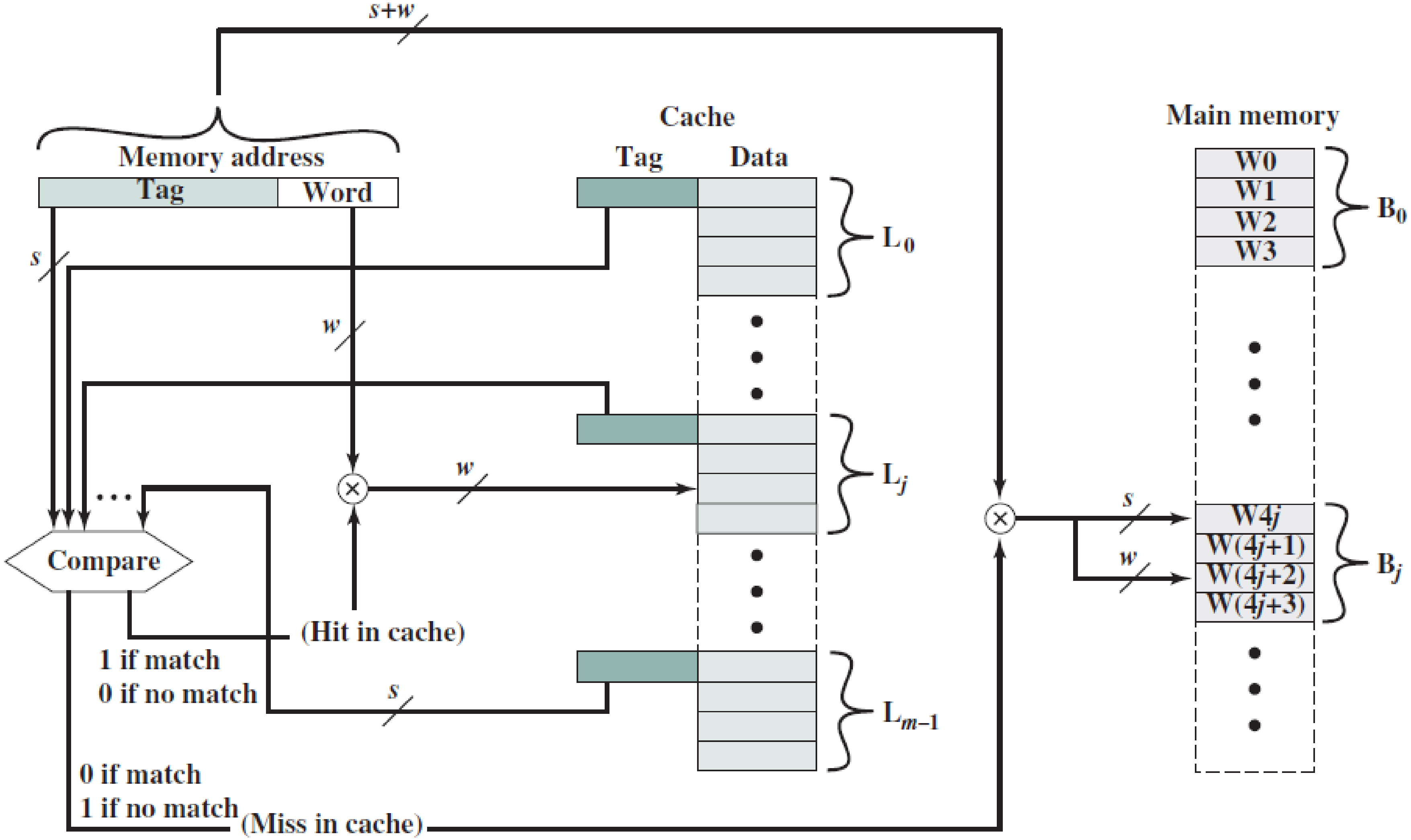
Associative mapping

- In this mapping, any main memory block is mapped on any cache line
- In this way it eliminates the “thrashing” problem
- But, to recognize the cache miss, all tags have to be checked
- To improve the performance, all tags are checked in parallel
  - It makes the design complex compared to direct mapping
- The address format is consider as given below

Tag	**Word**

- A cache line has “k” words and “w” bits are allocated for word field
- When the address has “n” bits, “n-w” bits are allocated for tag field
  - The cache has any number of cache lines (2^L for some ‘L’ is not mandatory)

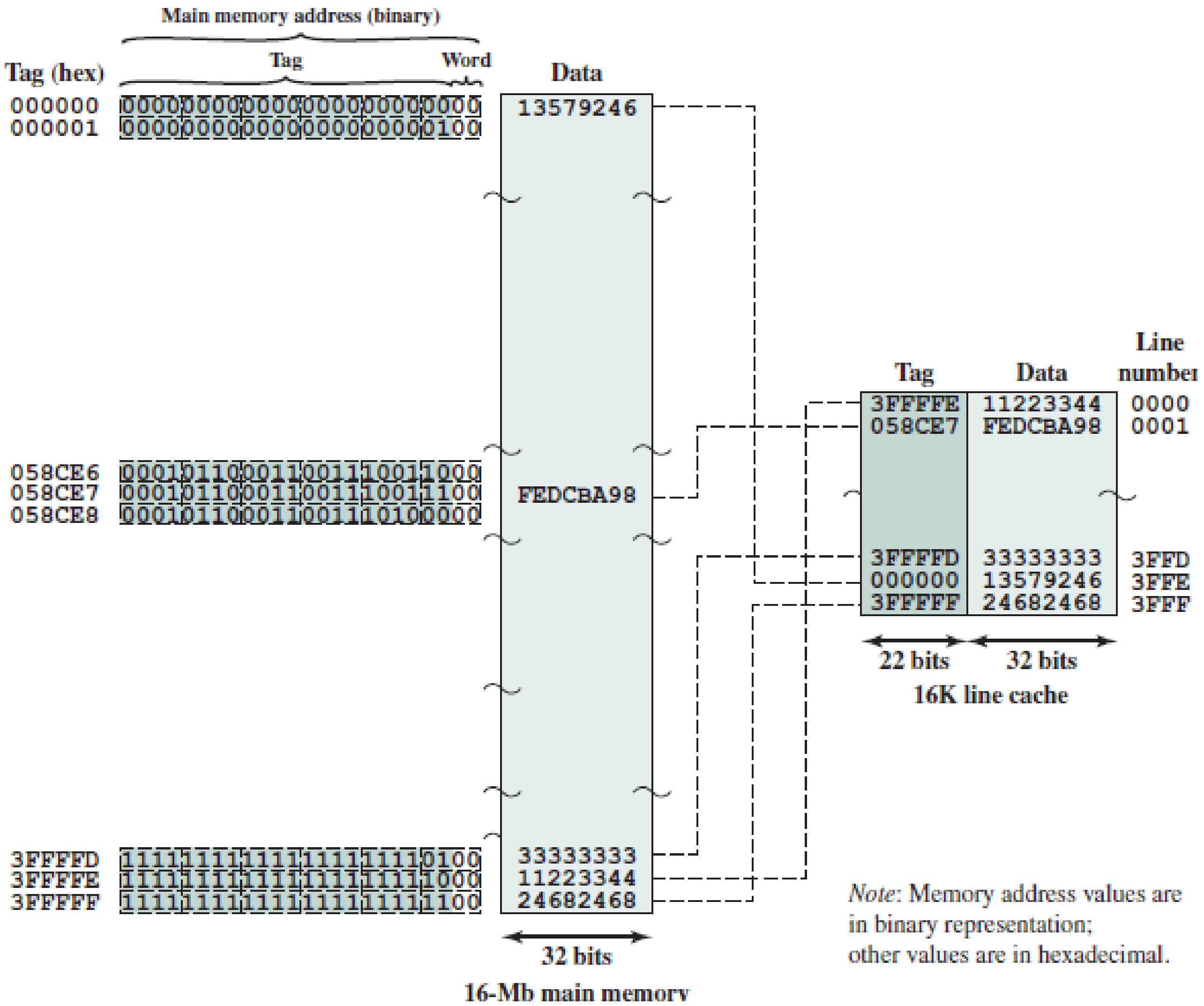
Associative mapping



A numerical example

- An associative cache can hold 64 KB words. Data are transferred between main memory and cache in blocks of 4 bytes each. The main memory size is 16 MB. Explain the address formats and mapping diagram
- Main memory size is 16 MB
  - Number of bits allocated for address is: $\log_2(16M)=\log_2(2^{24}) = 24$
- Block size is 4 bytes
  - Number of blocks in main memory = 16MB/4B = 4M = 2^{22} Blocks = 4 M Blocks
  - 22 bits are needed to identify a block – Tag size has to be 22 bits
  - Line size is 4 bytes – 2 bits are allocated to select a word from a line
  - Number of bits allocated for Tag field = (24 – 2) = 22
- Cache size is 64 KB = 2^{16} Bytes
  - Number of cache lines = $2^{16}/ 2^2 = 2^{14}$ = 16 K Lines
  - Any main memory block (among 2^{22}) is mapped any cache line (among 2^{14})

Fully associative mapping example


Main memory address (binary)
Tag (hex)
Tag
Word
Data
000000 0000 0000 0000 0000 0000 0000 13579246
000001 0000 0000 0000 0000 0000 0100
058CE6 0001 0110 0011 0011 1001 1000
058CE7 0001 0110 0011 0011 1001 1100 FEDCBA98
058CE8 0001 0110 0011 0011 1010 0000
3FFFFD 1111 1111 1111 1111 1111 0100 33333333
3FFFFE 1111 1111 1111 1111 1111 1000 11223344
3FFFFF 1111 1111 1111 1111 1111 1100 24682468
32 bits
16-Mb main memory
Tag
Data
Line number
3FFFFE 11223344 0000
058CE7 FEDCBA98 0001
3FFFD 33333333 3FFD
000000 13579246 3FFE
3FFFFF 24682468 3FFF
22 bits
32 bits
16K line cache

Note: Memory address values are in binary representation; other values are in hexadecimal.

Set Associative mapping

- It has the strengths of direct and fully associative mapping techniques
- The cache has a number of sets
  - The number ‘v’ can be represented as 2^d for some positive integer ‘d’
- Each set has ‘k’ lines
  - K may be any positive number
- Size of cache is : $v*k*2^w = k*2^d*2^w = k*2^{d+w}$
  - 2^w is the number of words per line
- A main memory block is mapped to any line of “a particular set” only
  - Set number = block number % k
  - For example when k=8, the block number 32 is mapped only on **set 0**

    Reason: 0= 32%8
  - Any line from the set will be chosen

Set Associative mapping…

- A k-way set associative mapping has ‘k’ lines in a single set
- A k-way set associative mapping has (m/k) sets when the cache has ‘m’ lines
  - K has to be the divisor of m
- When k=1, it is direct mapping
- When k=m, it is fully-associative mapping
- A 2-way set associative mapping has better performance than direct mapping and the complexity is minimized
- Commonly used k values are:2 and 4
- Address format

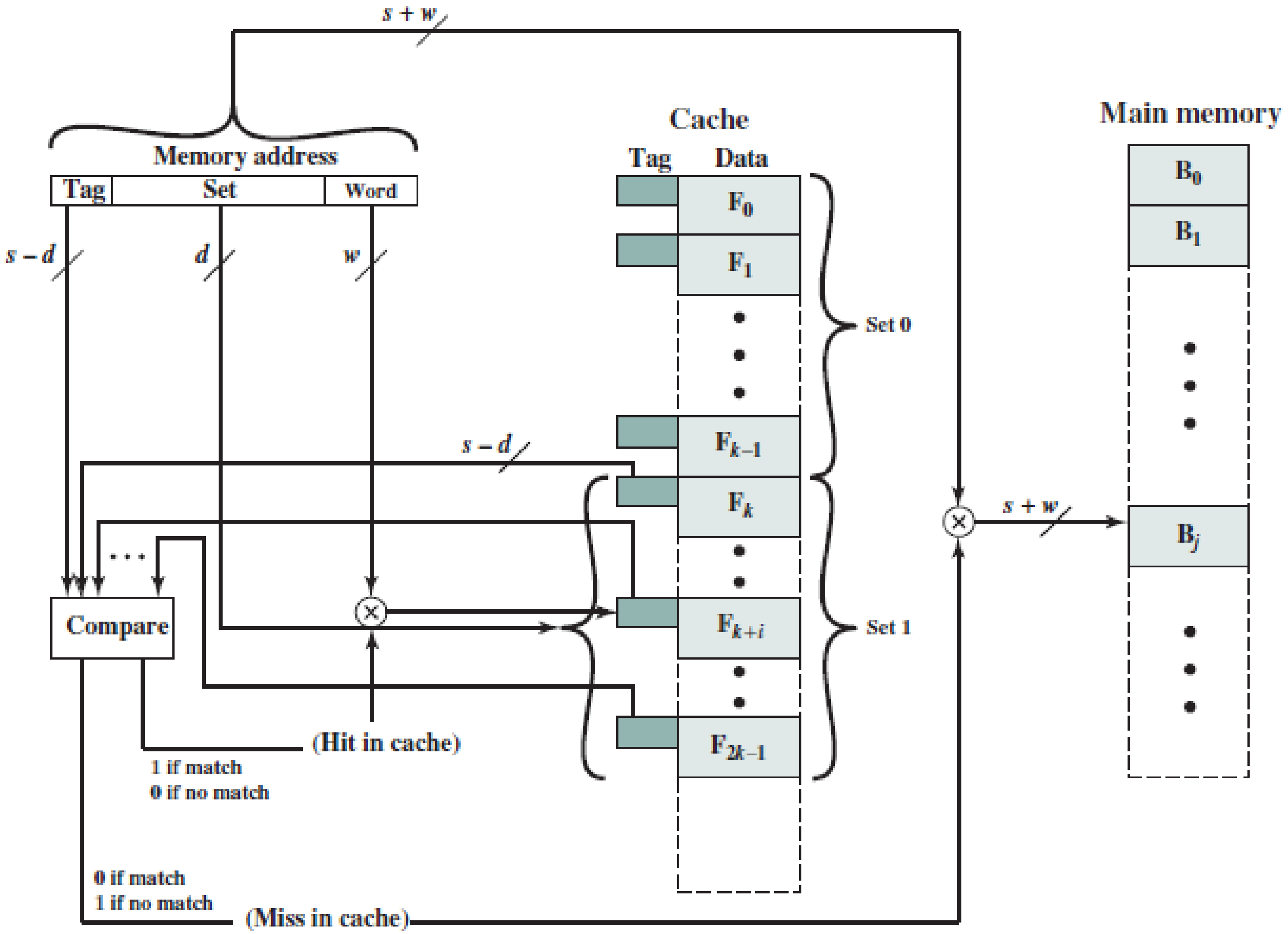
Tag	Set	Word

Set Associative mapping…

Tag	Set	Word

- The address is divided into the above fields
- The appropriate set which is indicated by the bits of set field is chosen
- All tag bits of the lines of that set will be compared in parallel
- If there is a match in tag, the word field is used to access the word
- If not, then the word with the block is moved from main memory to any one lines of that set using replacement algorithms

K-way set associative cache



A numerical example

- A 2-way set associative cache can hold 64 KB words. Data are transferred between main memory and cache in blocks of 4 bytes each. The main memory size is 16 MB. Explain the address formats and mapping diagram
- Main memory size is 16 MB
  - Number of bits allocated for address is: $\log_2(16M)=\log_2(2^{24}) = 24$
- Block size is 4 bytes
  - Line size is 4 bytes – 2 bits are allocated to select a word from a line
- Cache size is 64 KB = 2^{16} Bytes
  - Number of cache lines = $2^{16}/ 2^2 = 2^{14}$
- Two-way set associative cache
  - Number of sets = $2^{14} / 2 = 2^{13}$
  - Number of bits allocated for a set = 13
  - Number of bits allocated for tag = 24 – (13+2) = 24 – 15 = 9

Set Associative mapping…

Tag	Set	Word
(9 bits)	(13 bits)	(2 bits)

- For example, the address “0101_1010_1111_1000_1010_1110” is grouped as

0101_1010_1	1111_0001_0101_1	10

- The word is available in any one of the two cache lines of set number 1_1110_0010_1011 or equivalently on $(1E2B)_h$
- When any one of the line has the tag 0_1011_0101 or $(0B5)_h$, then cache hit
  - The third word (with word number 10 or $(2)_h$) of that line is accessed.
- If no lines has the tag, then the block 01_0110_1011_1110_0010_1011 or $(16BE2B)_h$ is copied from main memory and placed in anyone of the two cache lines of set $(1E2B)_h$

Thank you