CS 223: Computer Architecture & Organization

Cache Memory

J. K. Deka

Professor

Department of Computer Science & Engineering

Indian Institute of Technology Guwahati, Assam.

Cache Design

- Size
- Mapping Function
- Replacement Algorithm
- Write Policy
- Block Size
- Number of Caches

Mapping Function

- Cache of 64kByte
- Cache block of 16 bytes
  - i.e. cache is 4k (2^{12}) lines of 16 bytes
- 16MBytes main memory
- 24 bit address
  - (2^{24}=16M)
  - No. of block in main memory = 16MB/16B=1M

Direct Mapping

- Each block of main memory maps to only one cache line
  - i.e. if a block is in cache, it must be in one specific place
- Address is in two parts
- Least Significant w bits identify unique word
- Most Significant s bits specify one memory block
- The MSBs are split into a cache line field r and a tag of s-r (most significant)

Direct Mapping Address Structure

Tag s-r	Line or Slot r	Word w
8	12	4

- 24 bit address
- 4 bit word identifier (16 byte block)
- 20 bit block identifier
  - 8 bit tag (=20-12)
  - 12 bit slot or line
- No two blocks in the same line have the same Tag field
- Check contents of cache by finding line and checking Tag

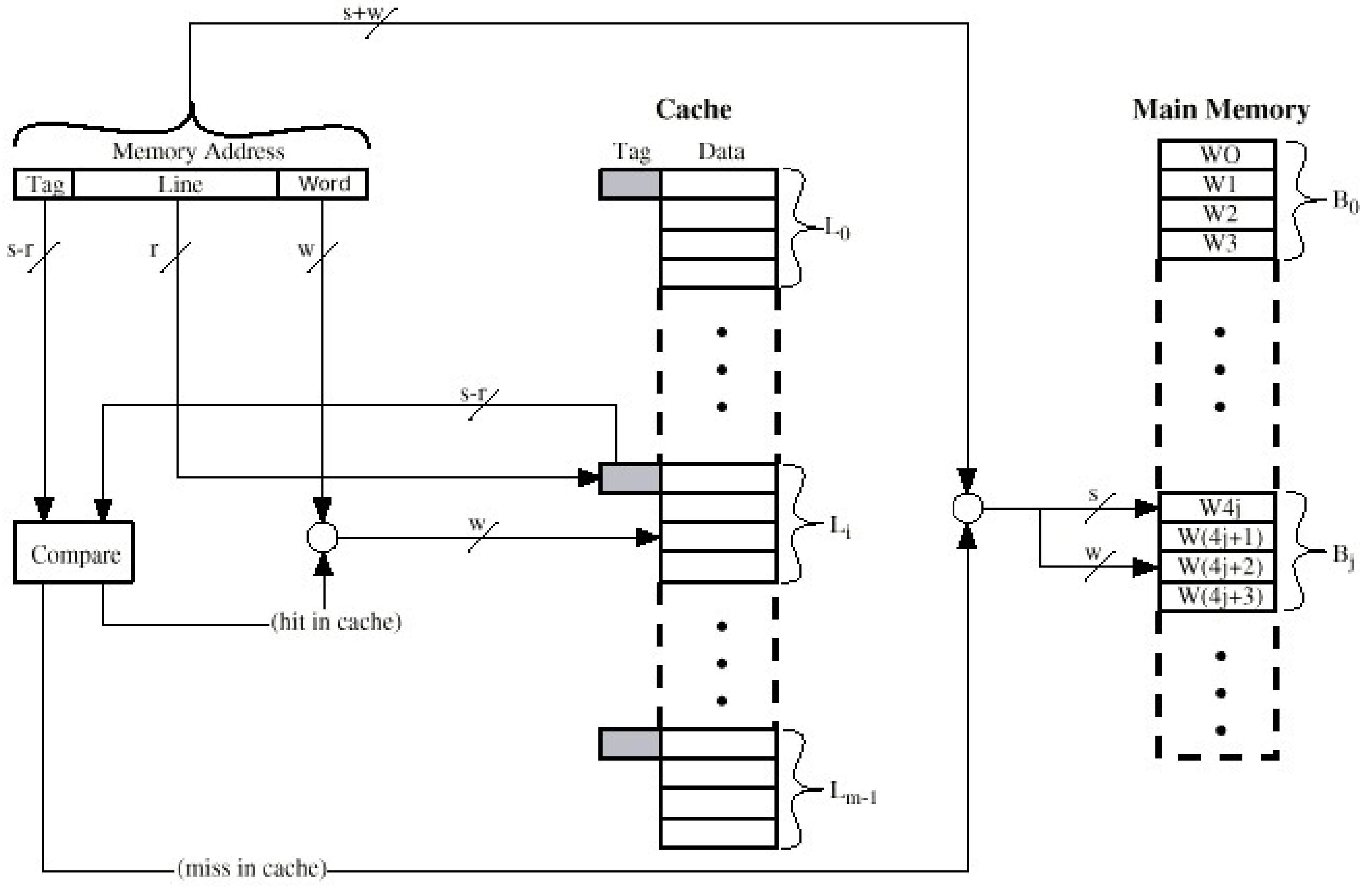
Direct Mapping Function

- Direct mapping function:
  - i = j modulo m
- Where
  - i = cache line number
  - j = main memory block number
  - m = number of lines in the cache

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

Direct Mapping Cache Organization



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

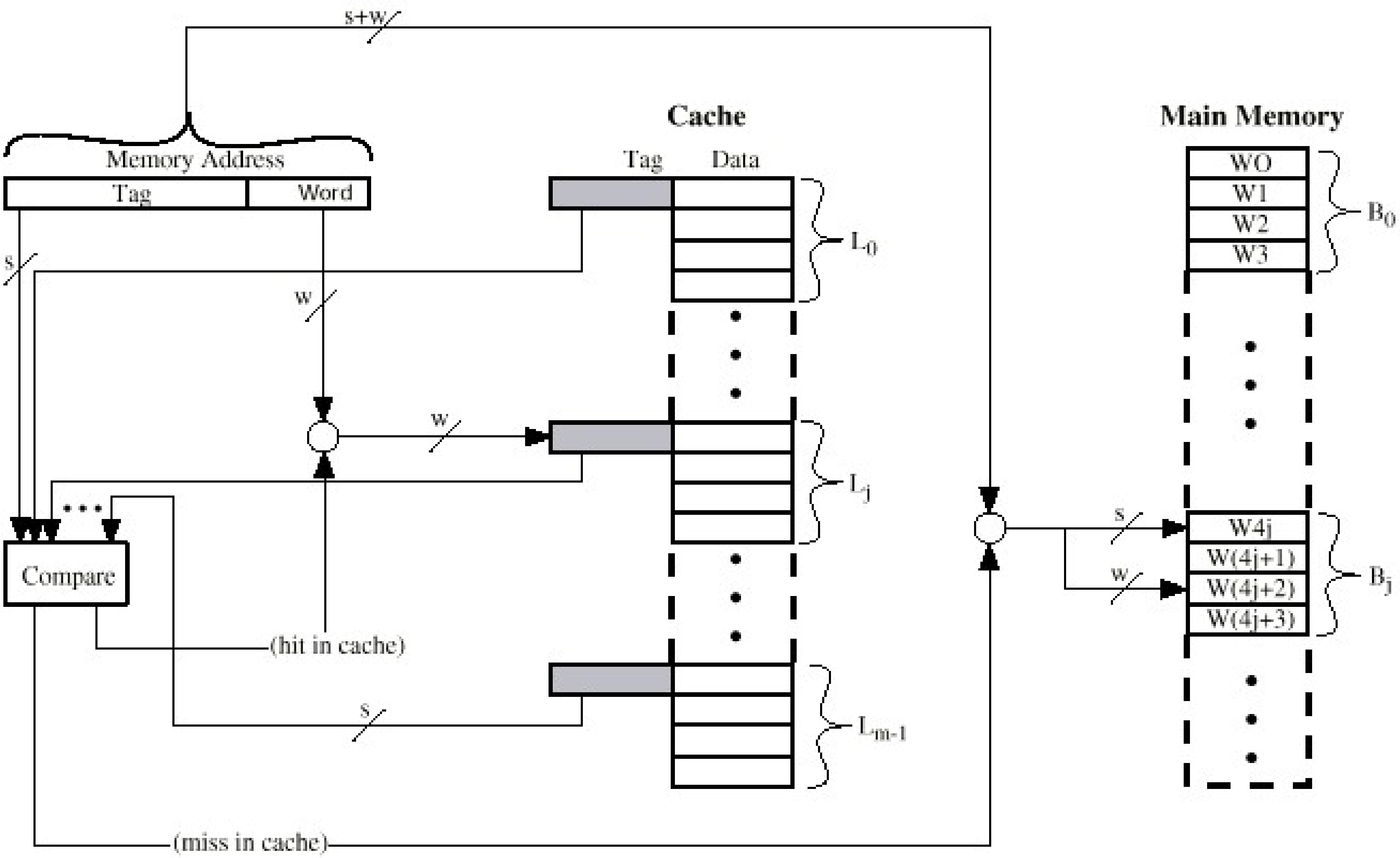
Direct Mapping pros & cons

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

Associative Mapping

- A main memory block can load into any line of cache
- Memory address is interpreted as tag and word
- Tag uniquely identifies block of memory
- Every line’s tag is examined for a match
- Cache searching gets expensive

Fully Associative Cache Organization



Associative Mapping Address Structure

Tag 20 bit	Word 4 bit

- 20 bit tag stored with each 16 byte block of data
- Compare tag field with tag entry in cache to check for hit
- Least significant 4 bits of address identify which byte is required from 16 byte data

Associative 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 = cache size/2^{w}
- Size of tag = s bits

Direct and Associative Mapping

Set Associative Mapping

- Cache is divided into a number of sets
- Each set contains a number of lines
- A given block maps to any line in a given set
  - e.g. Block B can be in any line of set i
- e.g. 2 lines per set
  - 2 way associative mapping
  - A given block can be in one of 2 lines in only one set

Set Associative Mapping

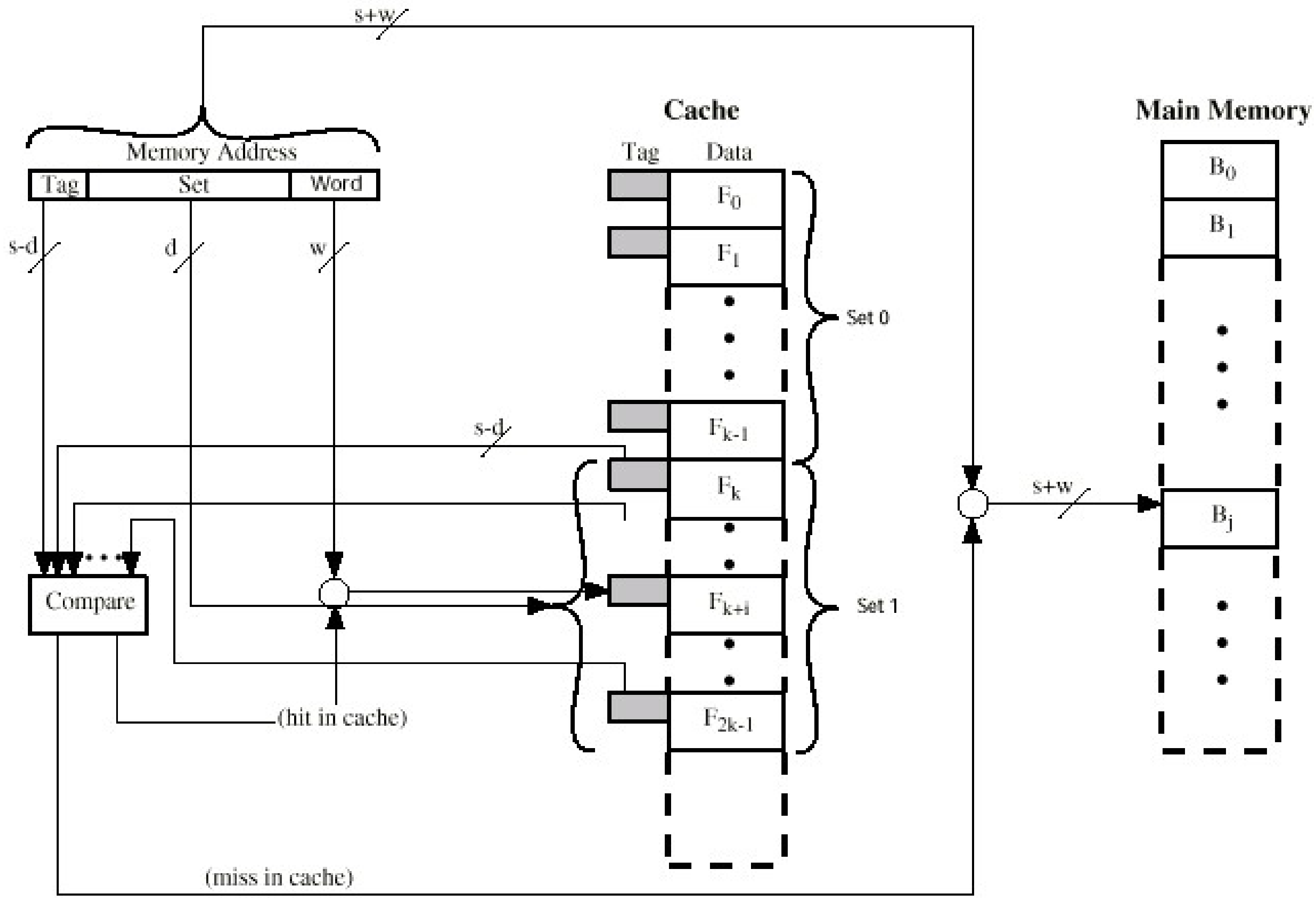
- The cache is divided in v sets
- Each set consists of k lines
- Number of lines in the cache
  - $m = v \times k$
- The mapping function:
  - $i = j$ modulo v
- Where
  - i = cache set number
  - j = main memory block number

K way set associative Mapping

Cache Line Table

Set no	Main Memory blocks held
0	0, v, 2v, 3v,…,2^s-v
1	1,v+1, 2v+1,…,2^s-v+1
v-1	v-1, 2v-1,3v-1,…,2^s-1

K Way Set Associative Cache Organization



Set Associative Mapping Address Structure

Tag 8 bit	Set 12 bit	Word 4 bit

- Use set field to determine cache set to look in
- Compare tag field to see if we have a hit
- e.g

Address	Tag	Data	Set number
1F 17F B	1F	12	17E
20 17E C	20	11	17E

Set Associative 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
- Number of lines in set = k
- Number of sets = v = 2^d
- Number of lines in cache = kv = k * 2^d
- Size of tag = (s – d) bits

Reference

Computer Organization and Architecture –
Designing for Performance
William Stallings, Seventh Edition

Chapter 04: Cache Memory

Computer Organization
Hamacher, Vranesic and Zaky, Fifth Edition

Chapter05: Page No.: 314 - 329