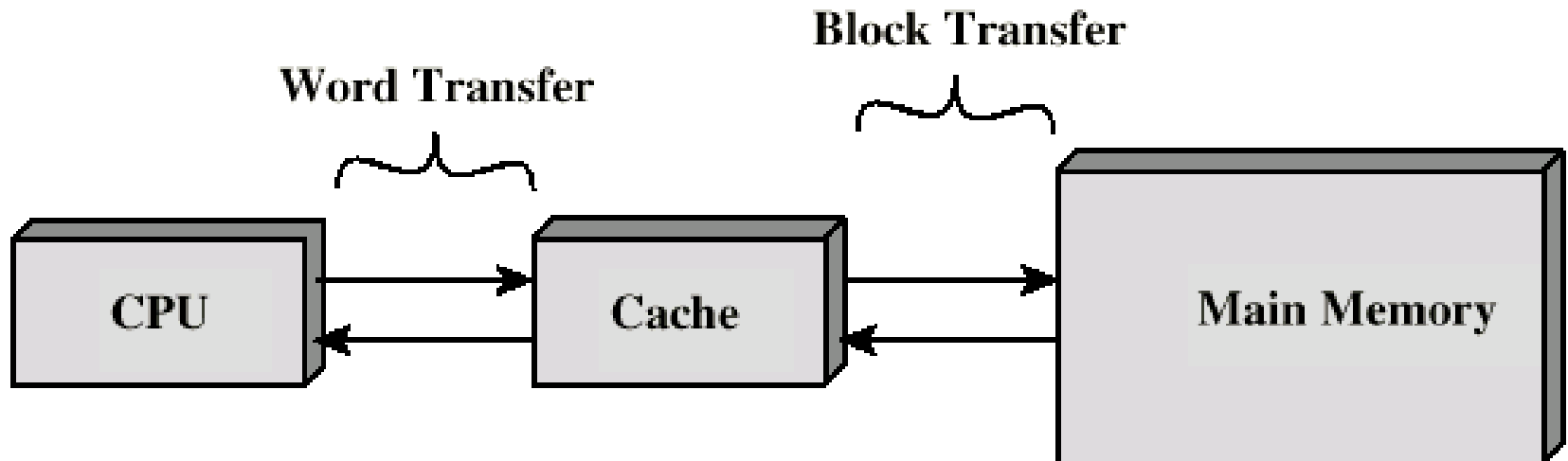


Chapter #4

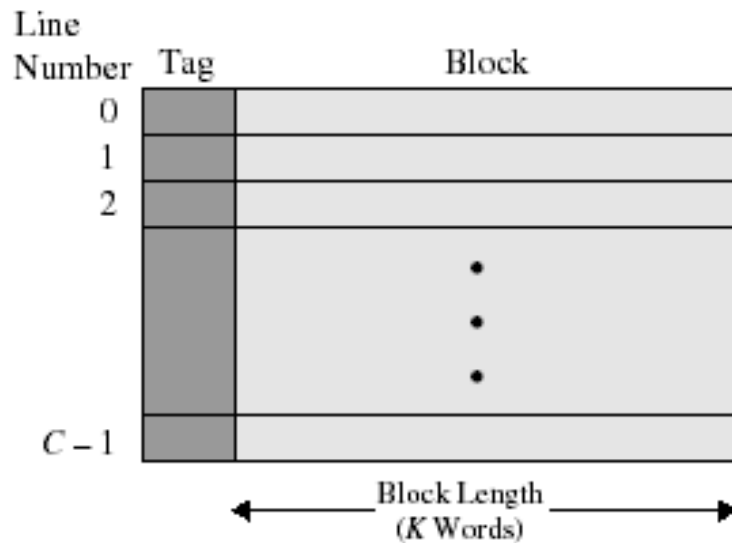
Cache Memory

Cache Memory

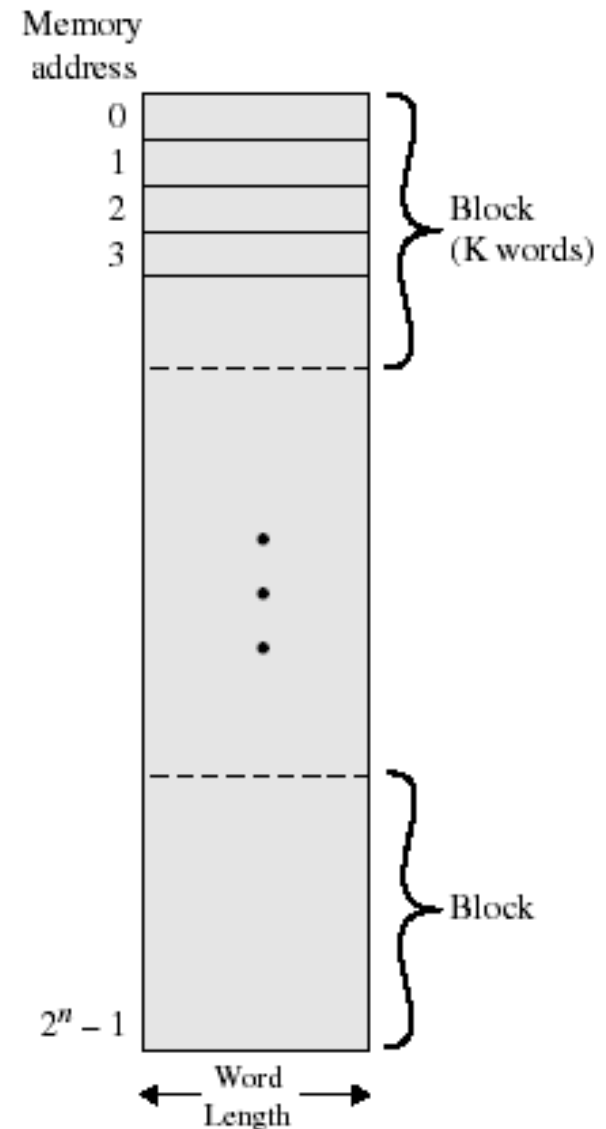
- Relatively small, fast memory
- Checked by CPU before main memory
- May be located on CPU chip, or off as separate module



Cache/Main Memory Structure



(a) Cache



(b) Main memory

Cache Operation

- CPU checks cache if address reference is present
- If present, get from cache (hit)
- If not present, read required block from main memory and update cache (miss)
- Deliver from cache to CPU

Cache Capacity

- Byte
 - 8 bits
- Word
 - fixed number of bytes (2,4,8,16,...)
 - Varies with technology/architecture
- Block/Line
 - fixed number of words (1,2,4,8,16,...)
 - Varies with technology/architecture
- Total storage (bytes):
 - ($\# \text{ bytes/word}$) \times ($\# \text{ words/line}$) \times ($\# \text{ lines}$)

Cache Performance

- Cache access time c
 - Time between presenting the address and getting the valid data
- Hit ratio h
 - Percentage of references found in cache
- Main memory access time m
- Mean access time: $c + (1-h)m$

Direct Mapping

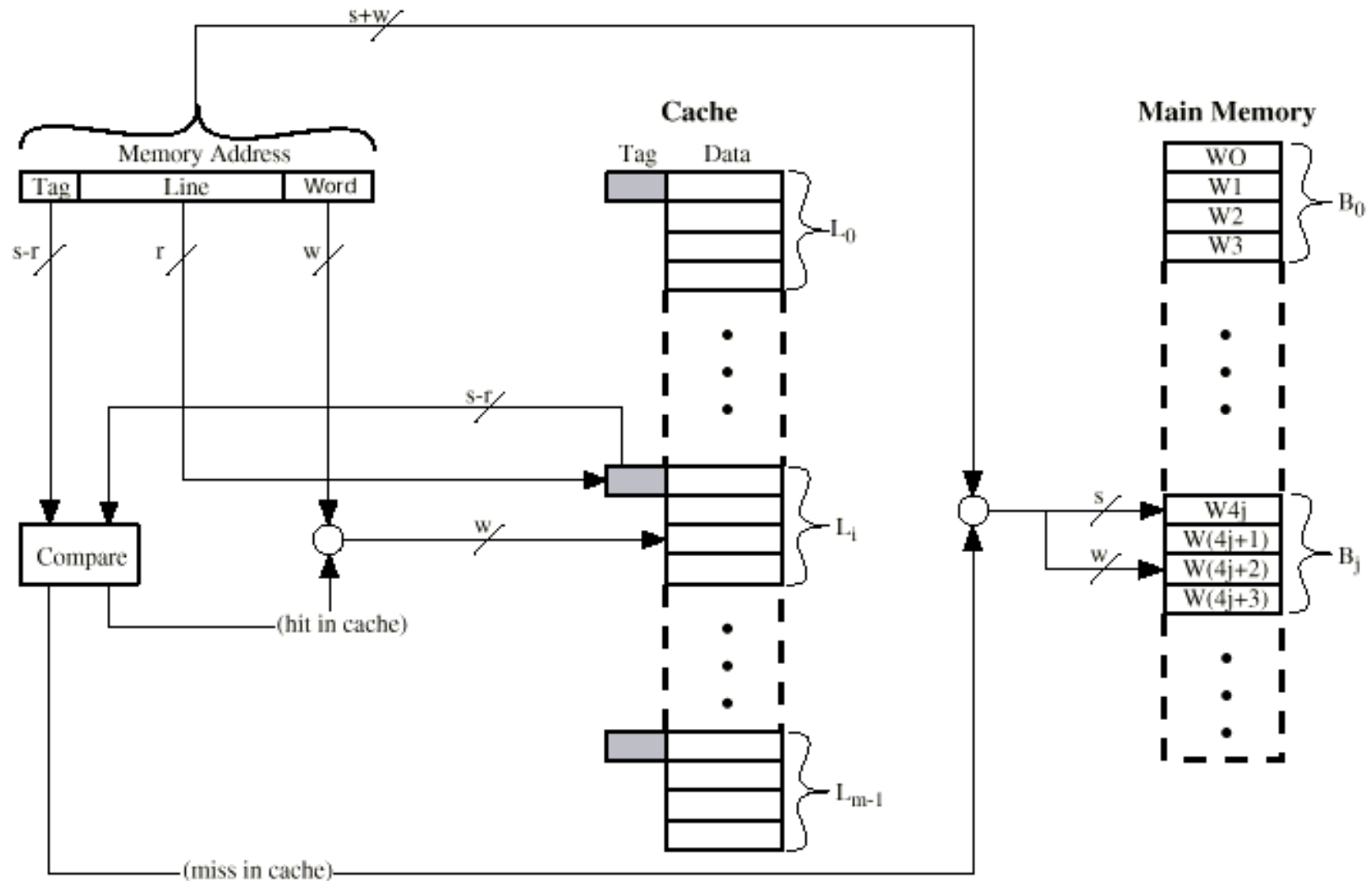
- Each block of main memory maps to one cache line
 - i.e. if a block is in cache, it must be in one specific place
- Least Significant w bits identify unique word
- Most Significant s bits specify one memory block
 - cache line field of r bits and a tag of $s-r$ bits

Direct Mapping Address Structure

Tag s-r	Line r	Word w
8	14	2

- 24 bit address
- 2 bit word identifier (4 words/block)
- 22 bit block identifier
 - 8 bit tag
 - 14 bit line (2^{14} lines)
- Check contents of cache by finding line and comparing tags (matching tag=hit, non-matching tag=miss)

Direct Mapping Cache Organization



Fully Associative Mapping

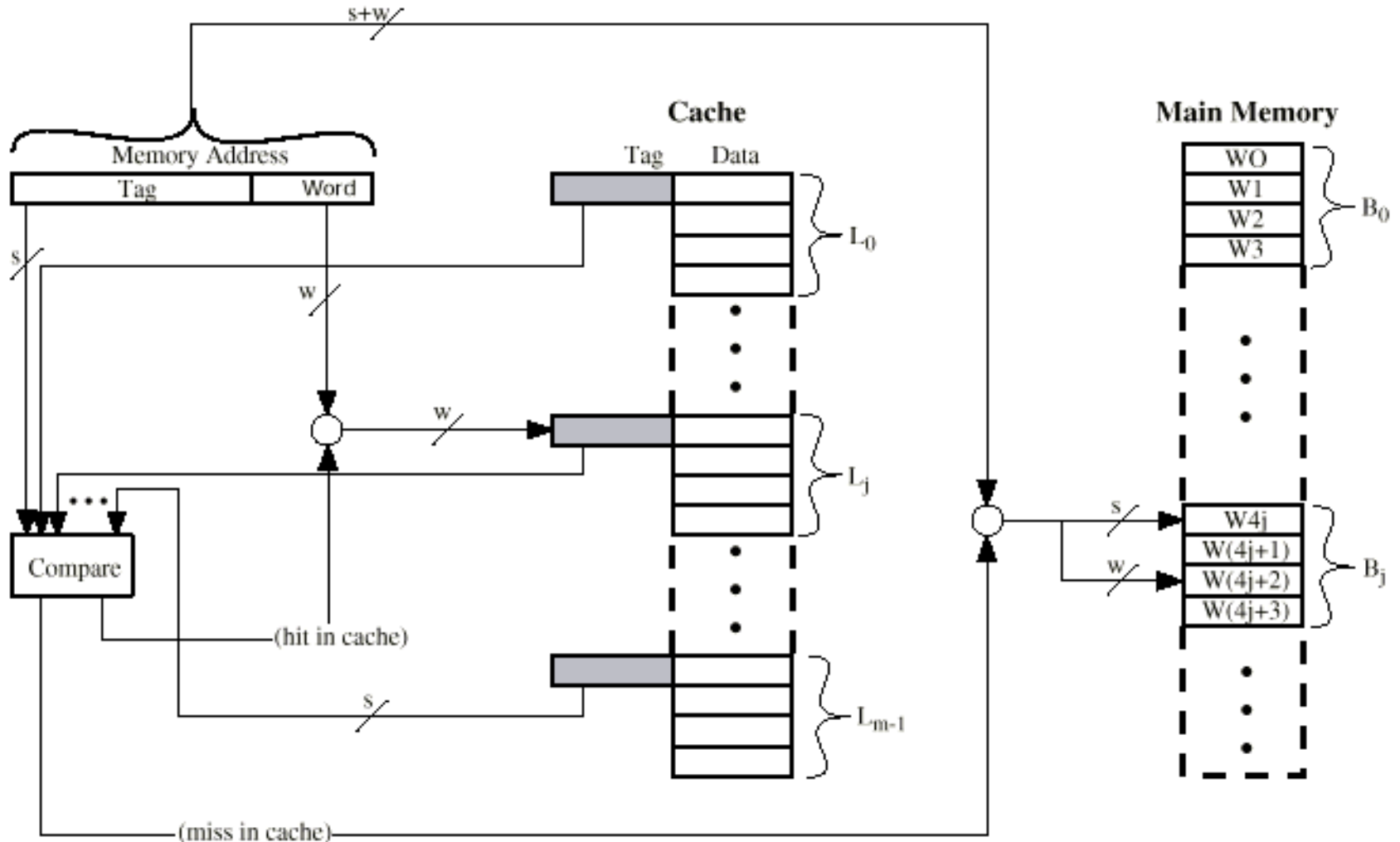
- A main memory block can load into any line of cache
- Memory address is interpreted as tag and word
- Tag uniquely identifies block of memory
- Every line's tag is examined for a match
- Cache searching gets expensive

Fully Associative Mapping Address Structure

Tag 22 bit	Word 2 bit
------------	---------------

- 24-bit address
- 2 bit word identifier (4 words/block)
 - 22 bit block identifier
 - 22 bit tag
- No line identifier
- Compare tag field with each tag entry in cache to check for hit

Fully Associative Cache Organization



Set Associative Mapping

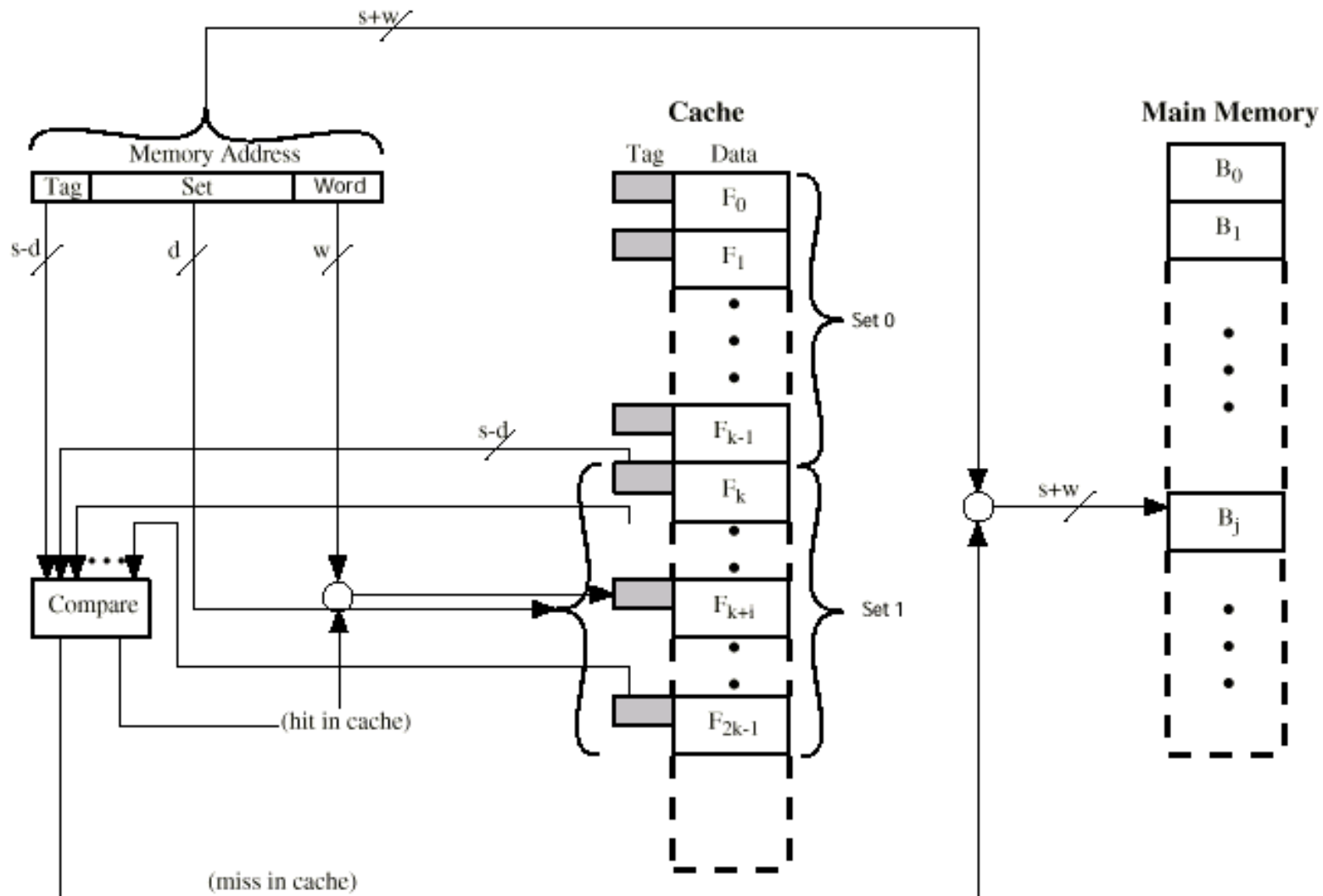
- Cache is divided into a number of sets
- Each set contains a number of lines
- A given block maps to any line in a given set
 - e.g. Block B can be in any line of set i
- e.g. 2 blocks per set
 - 2 way associative mapping
 - A given block can be in one of 2 blocks in only one set

Set Associative Mapping Address Structure

Tag 9 bit	Set 13 bit	Word 2 bit
-----------	------------	------------

- 24 bit address
- 2 bit word identifier
 - 22 bit block identifier
 - 9 bit tag
 - 13 bit set identifier (2^{13} sets in cache)
- Use set field to determine cache set to look in
- Compare tag field to see if there is hit

k Way Set Associative Cache Organization



Replacement Algorithm--Direct mapping

- No choice
- Each block only maps to one line
- Replace that line

Replacement Algorithms-- Fully Associative & Set Associative

- Least Recently used (LRU)
- First in first out (FIFO)
 - replace block that has been in cache longest
- Least frequently used (LFU)
 - replace block which has had fewest hits
- Random

Main Memory to Cache Address Mapping

- Parameters:

MM address: m

Set associativity: k

Cache size: c

Block size: b

- Fields (DIV=integer division, MOD=modulo):

tag = $m \text{ DIV } (c/k)$

set = $(m \text{ MOD } (c/k)) \text{ DIV } b$

word = $m \text{ MOD } b$