



Akdeniz University

Computer Engineering Department

CSE206 Computer Organization
Week05-06: Cache Memory

Assoc.Prof.Dr. Taner Danişman
tdanisman@akdeniz.edu.tr

Course program (Textbook: Stalling 10th Edt.)

Week 1	28-Feb-2023	Introduction	Ch1
Week 2	7-Mar-2023	Computer Evolution	Ch2
Week 3	14-Mar-2023	Computer Systems	Ch3
Week 4	21-Mar-2023	Cache Memory, Direct Cache Mapping	Ch4
Week 5	28-Mar-2023	Associative and Set Associative Mapping	Ch4
Week 6	4-Apr-2023	Internal Memory, External Memory, I/O	Ch5-Ch6-Ch7
Week 7	11-Apr-2023	Number Systems, Computer Arithmetic	Ch9-Ch10
Week 8	18-Apr-2023	Midterm (Expected date, may change)	Ch1-...-Ch10
Week 9	25-Apr-2023	Digital Logic	Ch11
Week 10	2-May-2023	Instruction Sets	Ch12
Week 11	9-May-2023	Addressing Modes	Ch13
Week 12	16-May-2023	Processor Structure and Function	Ch14
Week 13	23-May-2023	RISC, Instruction Level Parallelism	Ch15-Ch16
Week 14	30-May-2023	Assembly Language (TextBook : Assembly Language for x86 Processors)	Kip Irvine
Week 15	6-Jun-2023	Assembly Language (TextBook : Assembly Language for x86 Processors)	Kip Irvine

Course attendance information

- ➡ Theory = 30% ($3 \cdot 14 \cdot 30 / 100 = 12.6 \rightarrow 13$ lecture hours)
- ➡ Practice = 20% ($1 \cdot 14 \cdot 20 / 100 = 2.8 \rightarrow 3$ lecture hours)

Key Characteristics of Computer Memory Systems

Location Internal (e.g. processor registers, cache, main memory) External (e.g. optical disks, magnetic disks, tapes)	Performance Access time Cycle time Transfer rate
Capacity Number of words Number of bytes	Physical Type Semiconductor Magnetic Optical Magneto-optical
Unit of Transfer Word Block	Physical Characteristics Volatile/nonvolatile Erasable/nonerasable
Access Method Sequential Direct Random Associative	Organization Memory modules

Table 4.1 Key Characteristics of Computer Memory Systems

Characteristics of Memory Systems

➤ Location

- Refers to whether memory is internal and external to the computer
- Internal memory is often equated with main memory
- Processor requires its own local memory, in the form of registers
- Cache is another form of internal memory
- External memory consists of peripheral storage devices that are accessible to the processor via I/O controllers

➤ Capacity

- Memory is typically expressed in terms of bytes

➤ Unit of transfer

- For internal memory the unit of transfer is equal to the number of electrical lines into and out of the memory module

Method of Accessing Units of Data

Sequential access

Memory is organized into units of data called records

Access must be made in a specific linear sequence

Access time is variable

Direct access

Involves a shared read-write mechanism

Individual blocks or records have a unique address based on physical location

Access time is variable

Random access

Each addressable location in memory has a unique, physically wired-in addressing mechanism

The time to access a given location is independent of the sequence of prior accesses and is constant

Any location can be selected at random and directly addressed and accessed

Main memory and some cache systems are random access

Associative

A word is retrieved based on a portion of its contents rather than its address

Each location has its own addressing mechanism and retrieval time is constant independent of location or prior access patterns

Cache memories may employ associative access

Capacity and Performance:

The two most important characteristics of memory

Three performance parameters are used:

Access time (latency)

- For random-access memory it is the time it takes to perform a read or write operation
- For non-random-access memory it is the time it takes to position the read-write mechanism at the desired location

Memory cycle time

- Access time plus any additional time required before second access can commence
- Additional time may be required for transients to die out on signal lines or to regenerate data if they are read destructively
- Concerned with the system bus, not the processor

Transfer rate

- The rate at which data can be transferred into or out of a memory unit
- For random-access memory it is equal to $1/(\text{cycle time})$

Memory

- The most common forms are:
 - Semiconductor memory
 - Magnetic surface memory
 - Optical
 - Magneto-optical
- Several physical characteristics of data storage are important:
 - Volatile memory
 - Information decays naturally or is lost when electrical power is switched off
 - Nonvolatile memory
 - Once recorded, information remains without deterioration until deliberately changed
 - No electrical power is needed to retain information
 - Magnetic-surface memories
 - Are nonvolatile
 - Semiconductor memory
 - May be either volatile or nonvolatile
 - Nonerasable memory
 - Cannot be altered, except by destroying the storage unit
 - Semiconductor memory of this type is known as read-only memory (ROM)
- For random-access memory the organization is a key design issue
 - Organization refers to the physical arrangement of bits to form words

Memory Hierarchy

- Design constraints on a computer's memory can be summed up by three questions:
 - How much, how fast, how expensive
- There is a trade-off among capacity, access time, and cost
 - Faster access time, greater cost per bit
 - Greater capacity, smaller cost per bit
 - Greater capacity, slower access time

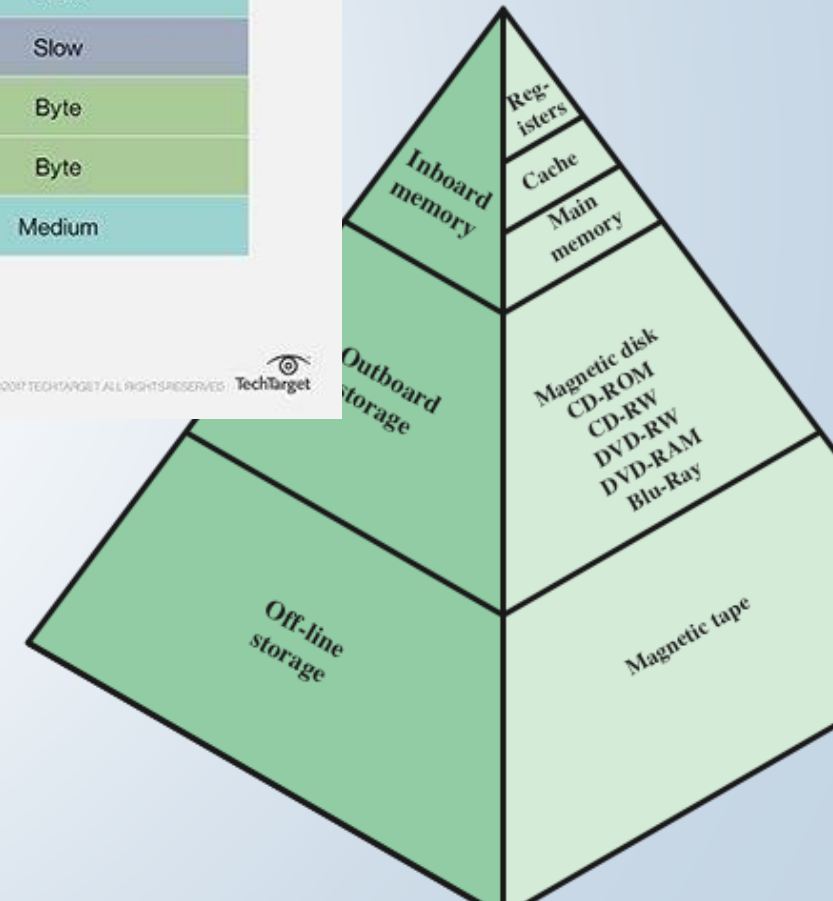
Memory Hierarchy - Diagram

Comparing memory types

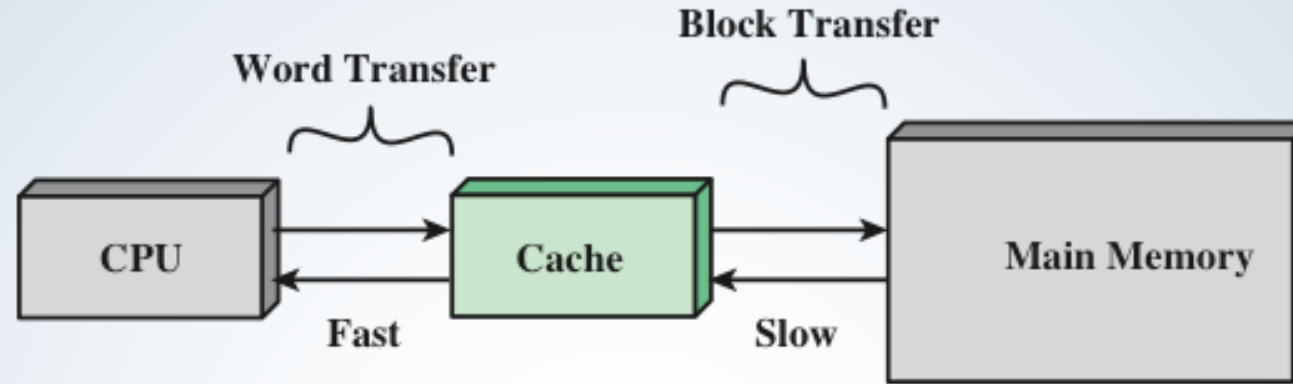
TYPE	SRAM	DRAM	NAND FLASH	NOR FLASH
Non-volatile	No	No	Yes	Yes
Price per GB	High	Low	Very low	Low
Read speed	Very fast	Fast	Slow	Fast
Write speed	Very fast	Fast	Slow	Slow
Smallest write	Byte	Byte	Page	Byte
Smallest read	Byte	Page	Page	Byte
Power	High	High	Medium	Medium

■ UNDESIRABLE/LEAST DESIRABLE ■ MIDDLE ■ MOST DESIRABLE

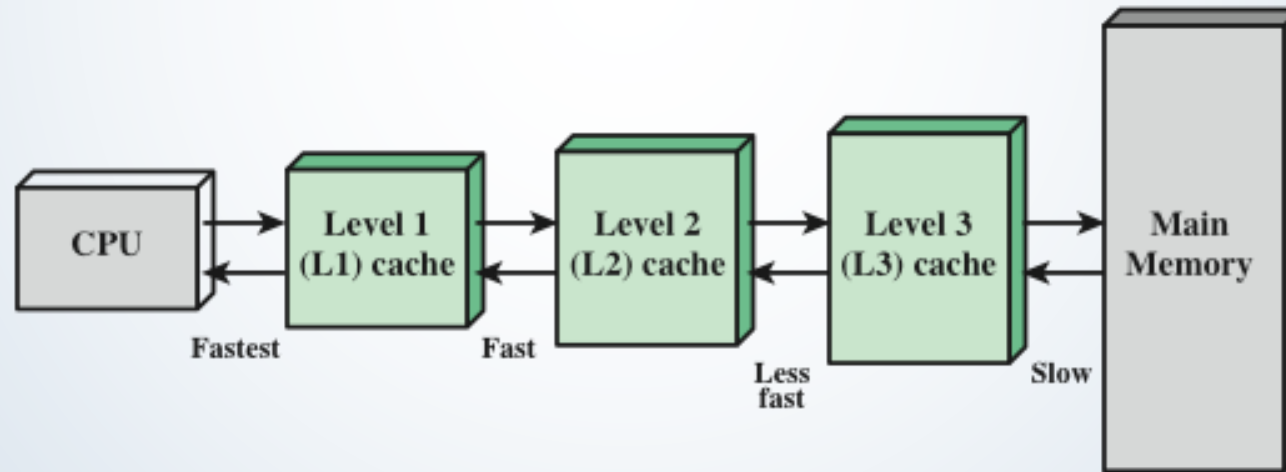
©2007 TECHTARGET ALL RIGHTS RESERVED TechTarget



Cache and Main Memory



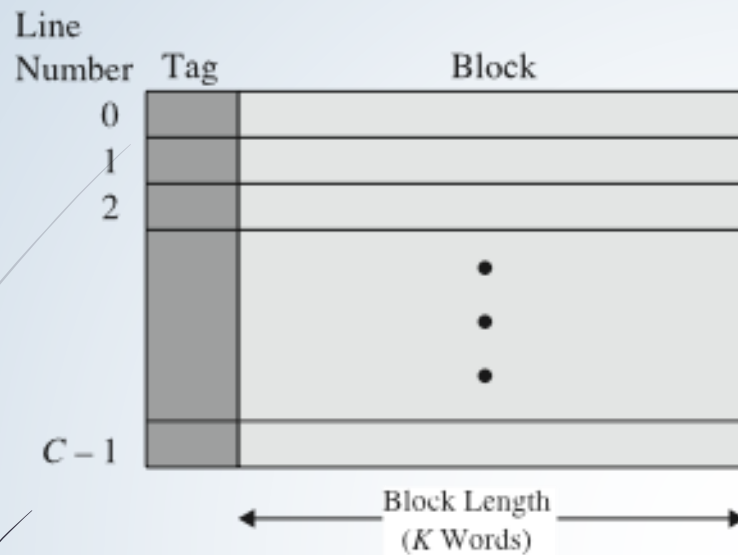
(a) Single cache



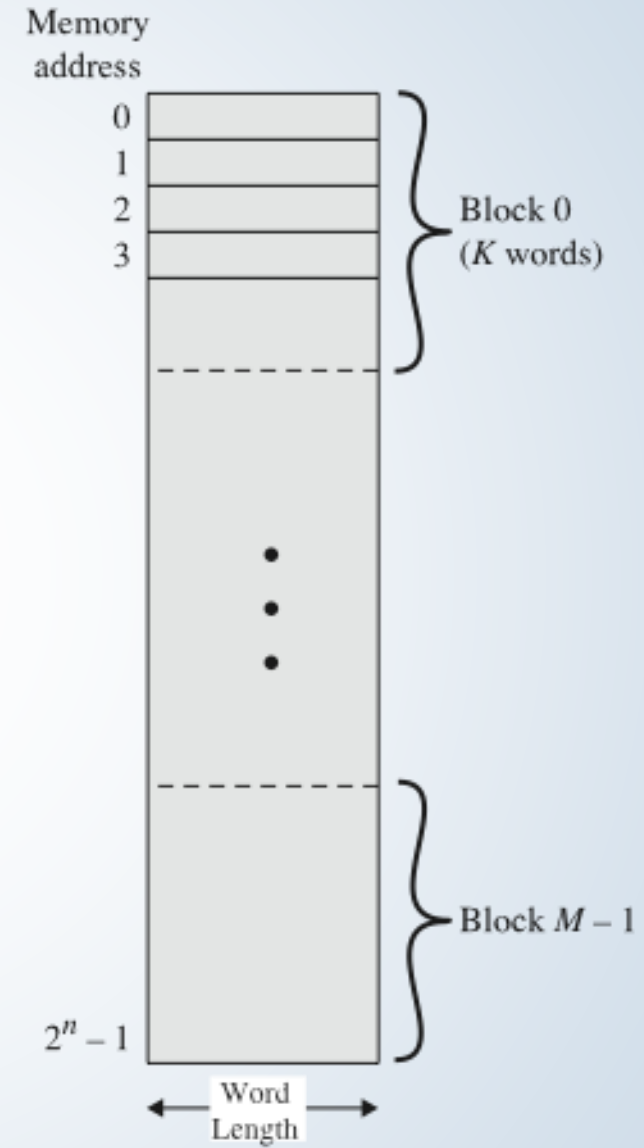
(b) Three-level cache organization

Figure 4.3 Cache and Main Memory

Cache/Main Memory Structure



(a) Cache



(b) Main memory

Figure 4.4 Cache/Main-Memory Structure

Cache Read Operation

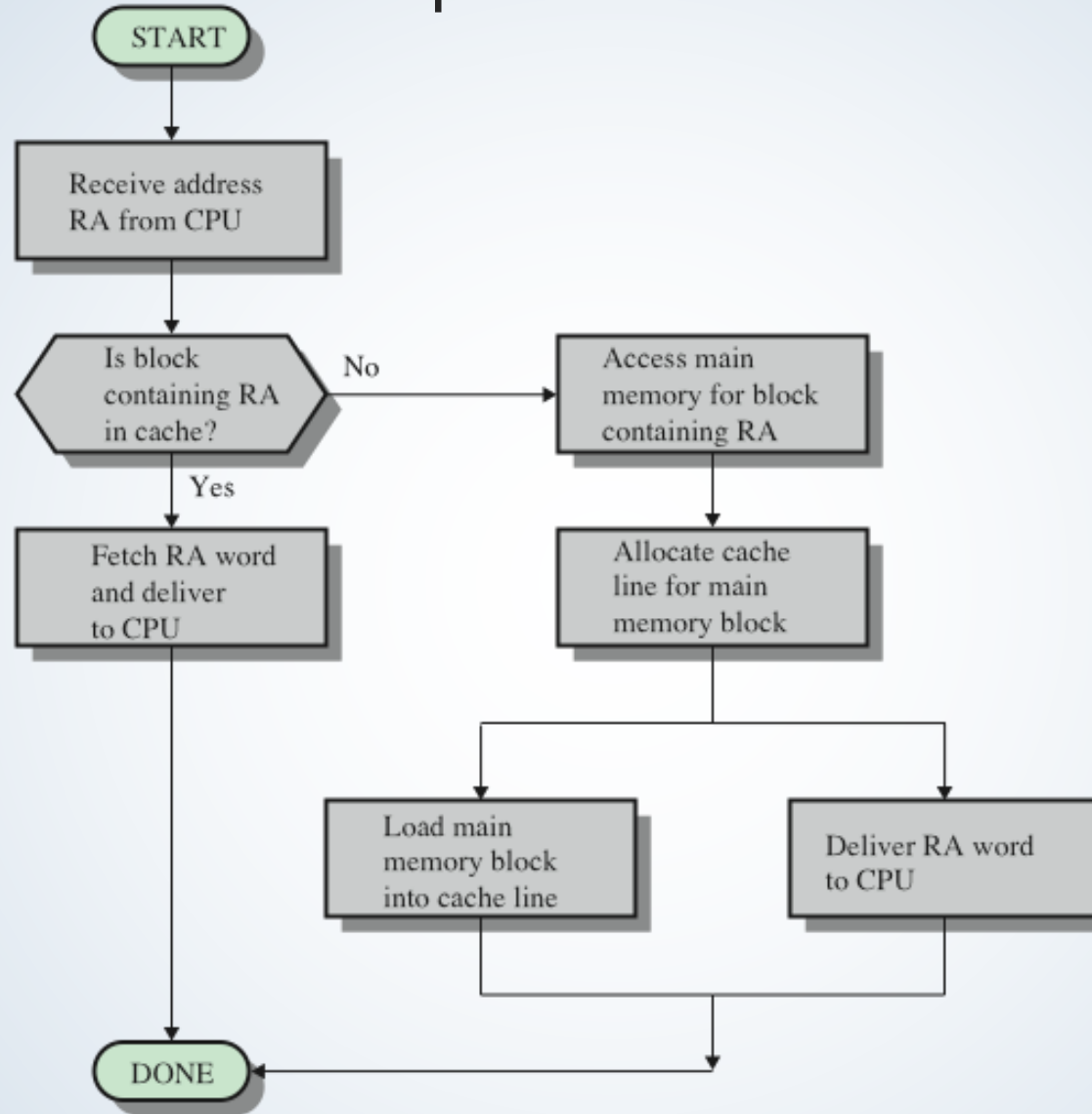
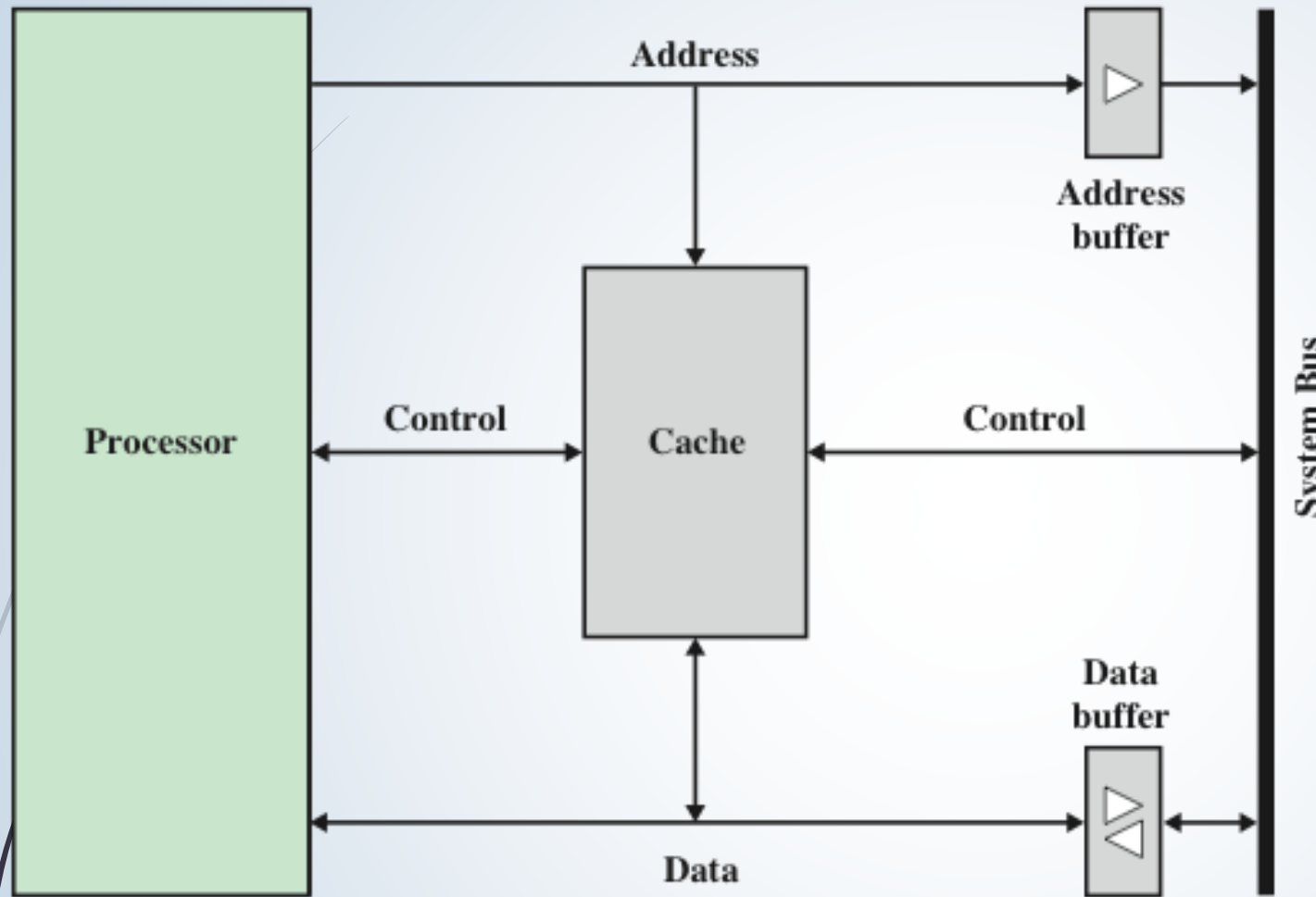


Figure 4.5 Cache Read Operation

Typical Cache Organization



- When a cache **hit** occurs, the data and address buffers are disabled and communication is only between processor and cache, with no system bus traffic.
- When a cache **miss** occurs, the desired address is loaded onto the system bus and the data are returned through the data buffer to both the cache and the processor.

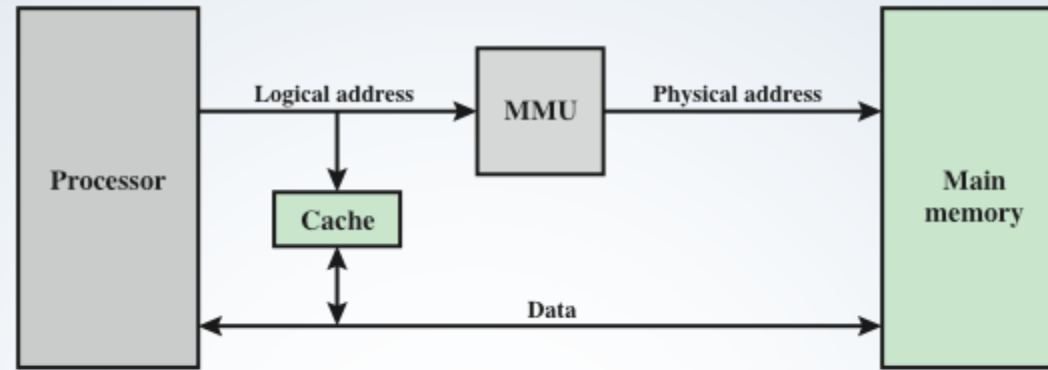
Figure 4.6 Typical Cache Organization

Cache Addresses

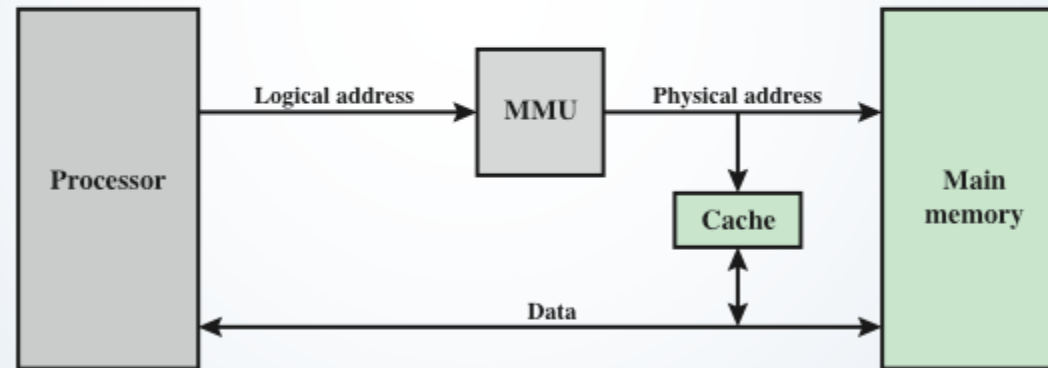
➤ Virtual memory

- Facility that allows programs to address memory from a logical point of view, without regard to the amount of main memory physically available
- When used, the address fields of machine instructions contain virtual addresses
- For reads to and writes from main memory, a hardware memory management unit (MMU) translates each virtual address into a physical address in main memory

Logical and Physical Caches



(a) Logical Cache



(b) Physical Cache

Figure 4.7 Logical and Physical Caches

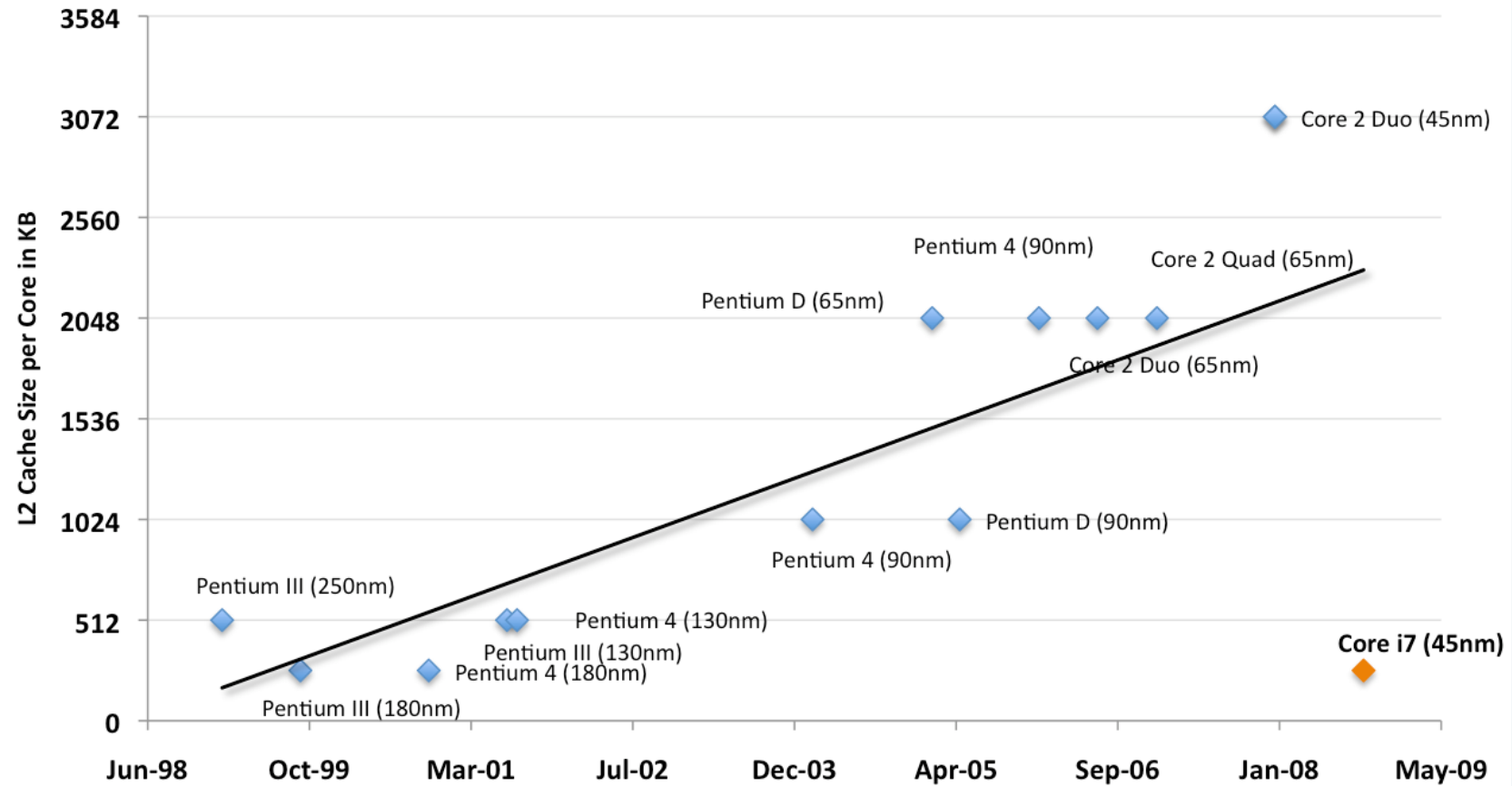
Cache Sizes of Some Processors

Processor	Type	Year of Introduction	L1 Cache _a	L2 cache	L3 Cache
IBM 360/85	Mainframe	1968	16 to 32 kB	—	—
PDP-11/70	Minicomputer	1975	1 kB	—	—
VAX 11/780	Minicomputer	1978	16 kB	—	—
IBM 3033	Mainframe	1978	64 kB	—	—
IBM 3090	Mainframe	1985	128 to 256 kB	—	—
Intel 80486	PC	1989	8 kB	—	—
Pentium	PC	1993	8 kB/8 kB	256 to 512 KB	—
PowerPC 601	PC	1993	32 kB	—	—
PowerPC 620	PC	1996	32 kB/32 kB	—	—
PowerPC G4	PC/server	1999	32 kB/32 kB	256 KB to 1 MB	2 MB
IBM S/390 G6	Mainframe	1999	256 kB	8 MB	—
Pentium 4	PC/server	2000	8 kB/8 kB	256 KB	—
IBM SP	High-end server/ supercomputer	2000	64 kB/32 kB	8 MB	—
CRAY MTA _b	Supercomputer	2000	8 kB	2 MB	—
Itanium	PC/server	2001	16 kB/16 kB	96 KB	4 MB
Itanium 2	PC/server	2002	32 kB	256 KB	6 MB
IBM POWER5	High-end server	2003	64 kB	1.9 MB	36 MB
CRAY XD-1	Supercomputer	2004	64 kB/64 kB	1MB	—
IBM POWER6	PC/server	2007	64 kB/64 kB	4 MB	32 MB
IBM z10	Mainframe	2008	64 kB/128 kB	3 MB	24-48 MB
Intel Core i7 EE 990	Workstation/ server	2011	6 × 32 kB/32 kB	1.5 MB	12 MB
IBM zEnterprise 196	Mainframe/ Server	2011	24 × 64 kB/ 128 kB	24 × 1.5 MB	24 MB L3 192 MB L4

^a Two values separated by a slash refer to instruction and data caches.

^b Both caches are instruction only; no data caches.

L2 Cache Size per Core vs. Time



Mapping Function

- Because there are fewer cache lines than main memory blocks, an algorithm is needed for mapping main memory blocks into cache lines
- Three techniques can be used:

Direct

- The simplest technique
- Maps each block of main memory into only one possible cache line

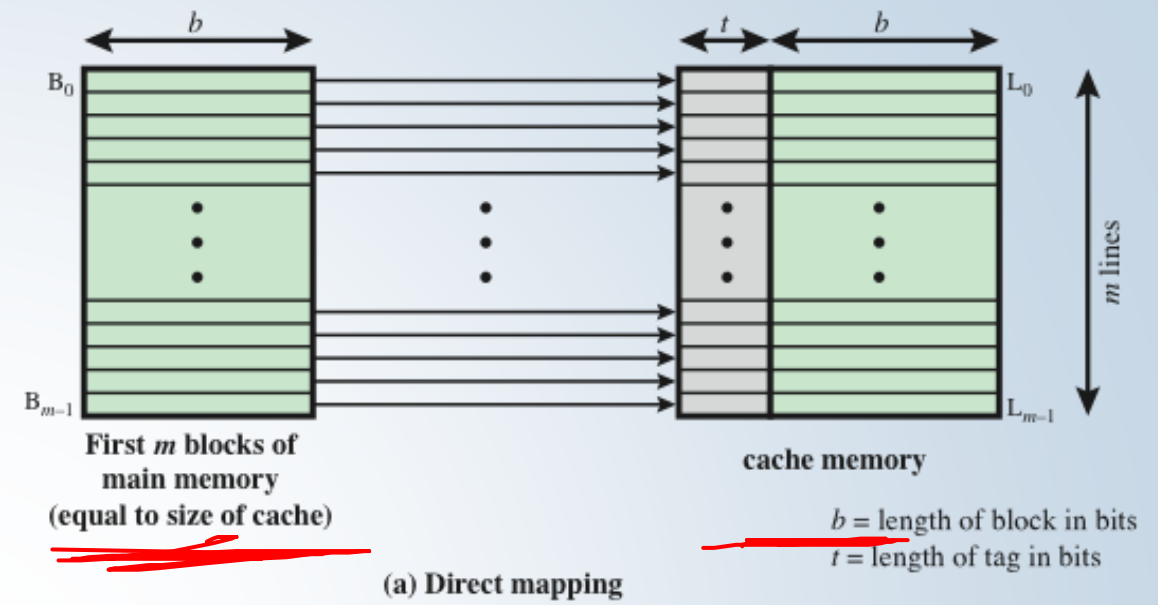
Associative

- Permits each main memory block to be loaded into any line of the cache
- The cache control logic interprets a memory address simply as a Tag and a Word field
- To determine whether a block is in the cache, the cache control logic must simultaneously examine every line's Tag for a match

Set Associative

- A compromise that exhibits the strengths of both the direct and associative approaches while reducing their disadvantages

Direct Mapping



Associative Mapping

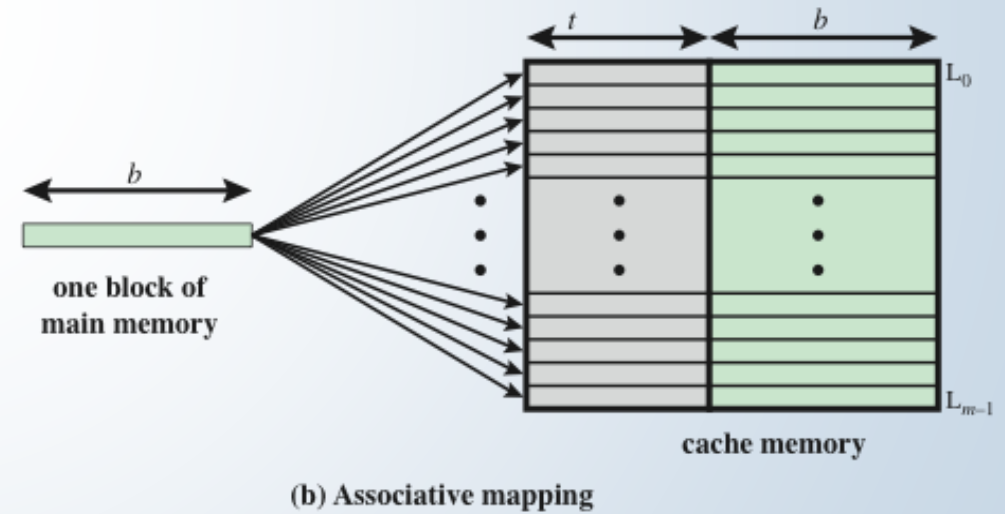


Figure 4.8 Mapping From Main Memory to Cache:
Direct and Associative

Direct Mapping Cache Organization

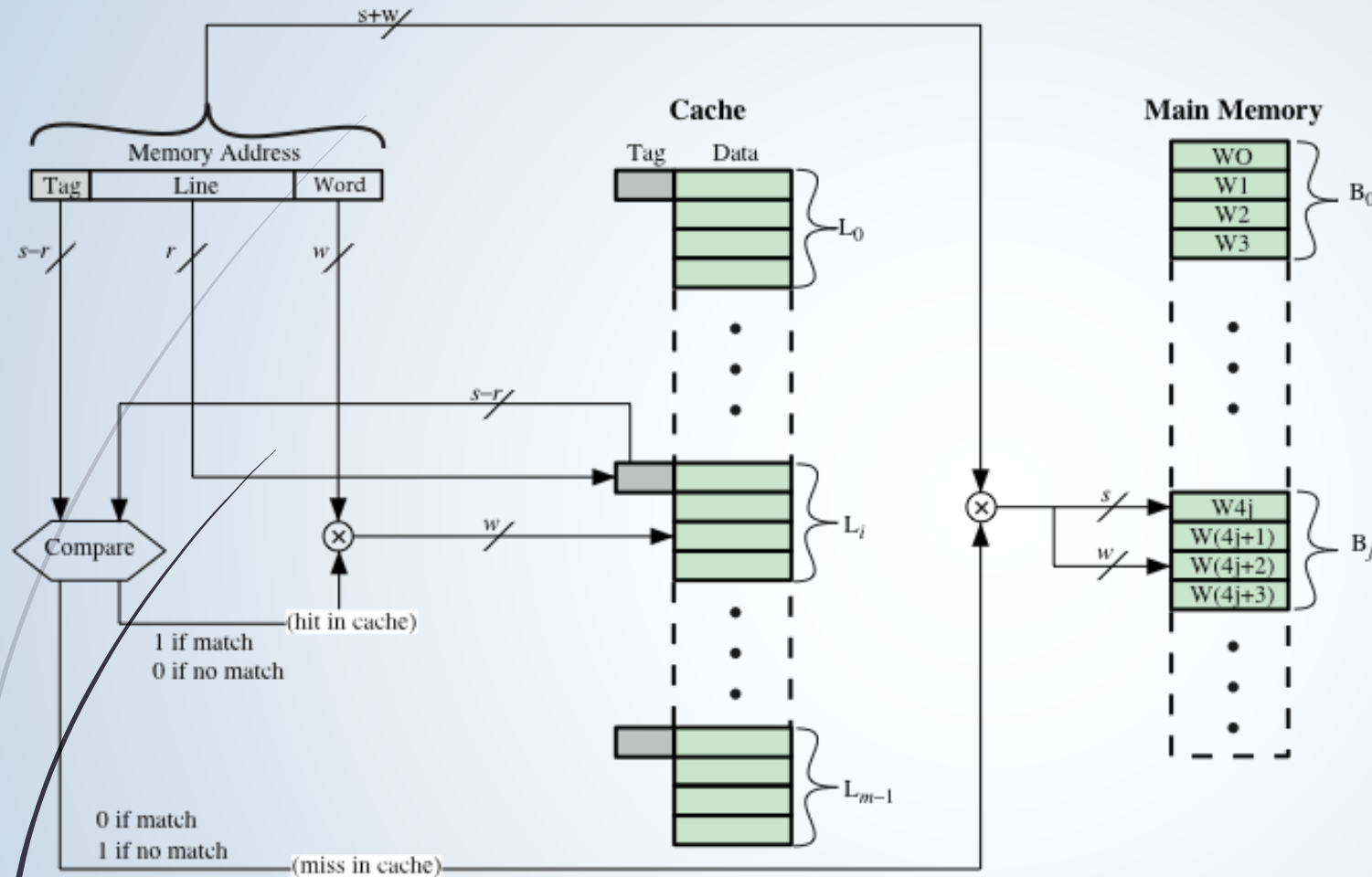
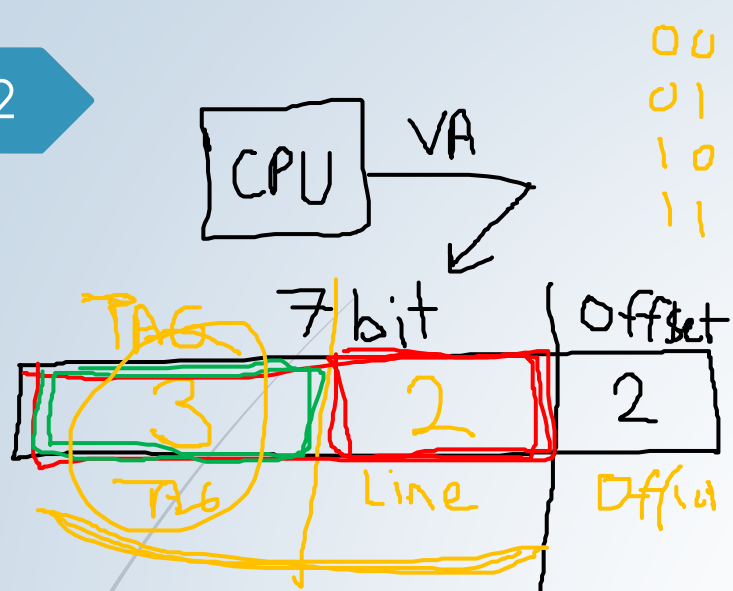
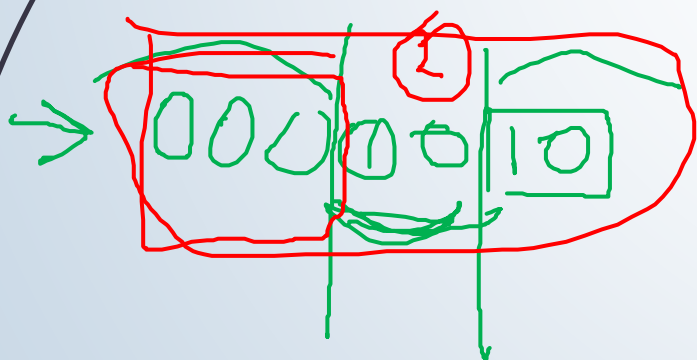


Figure 4.9 Direct-Mapping Cache Organization

- 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



6 5 4 3 2 1 0
0000110 → 6



3	2	2
011	00	10
Line		



4 x 3 bit
12 bit
2 x 3

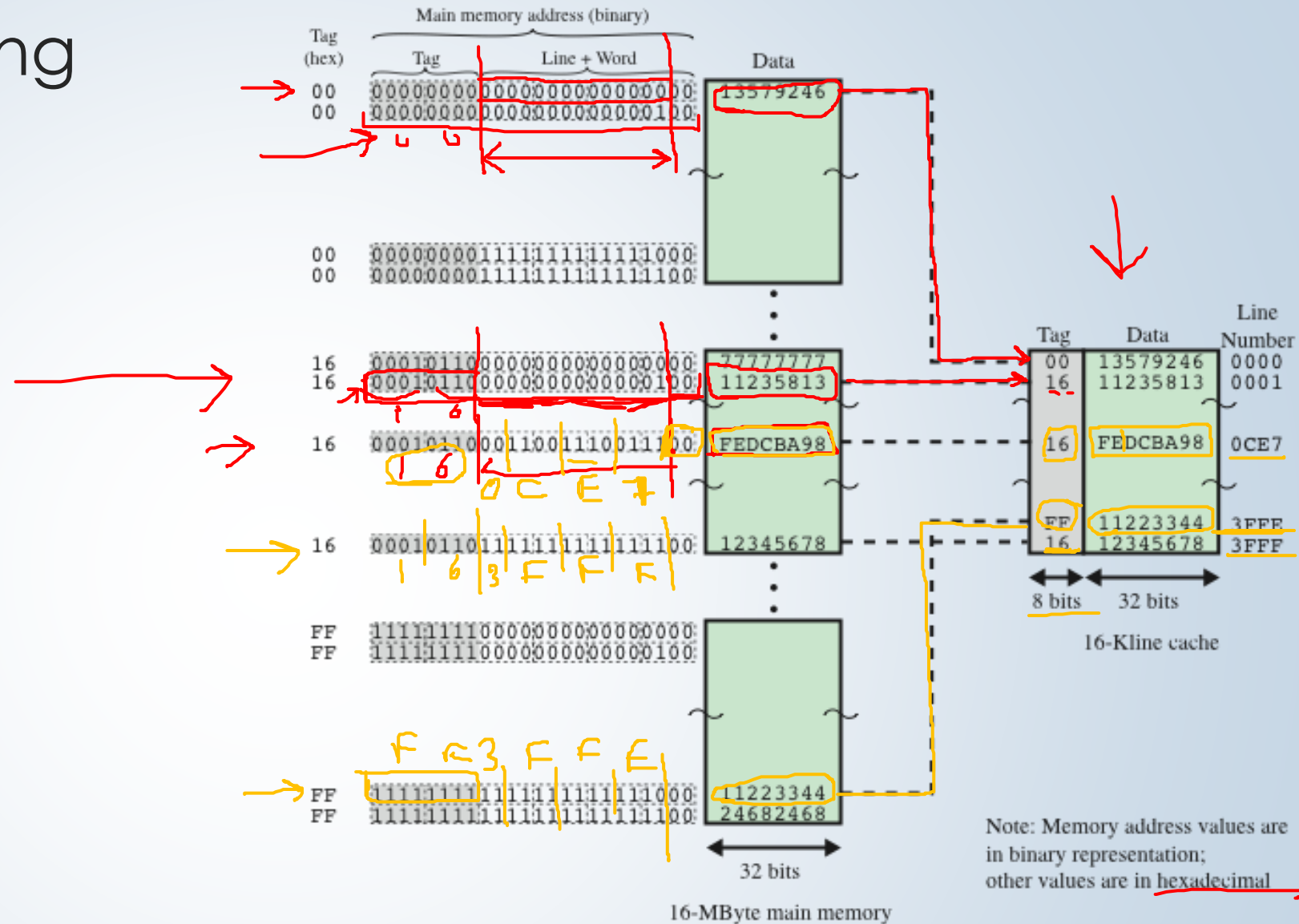
01100000
01
10
11



BS = 4 byte



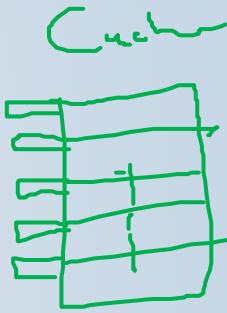
Direct Mapping Example



Direct Mapping Summary

TAG	35 bit	Offset
19	5	11

- 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



$$1GB = 2^{30}$$

$$2^5 \times 2^{20}$$

MM

32GB

$$\downarrow$$

$$2^{35} \text{ bytes}$$

Cache

64KB

$$\downarrow$$

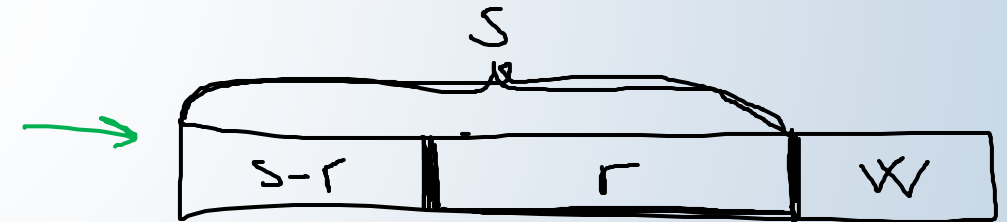
$$2^6 \times 2^{10}$$

$$2^{16} \text{ bytes}$$

Block

2KB

$$2^{11}$$



TAG

TAG Dir
size

$$\frac{2^{16}}{2^{11}} = 2^5$$

$$2^5 \times 19 = 32 \times 19$$

$$= \dots \text{ bits}$$

Associative Cache

with associative mapping, each word maps into multiple cache lines.

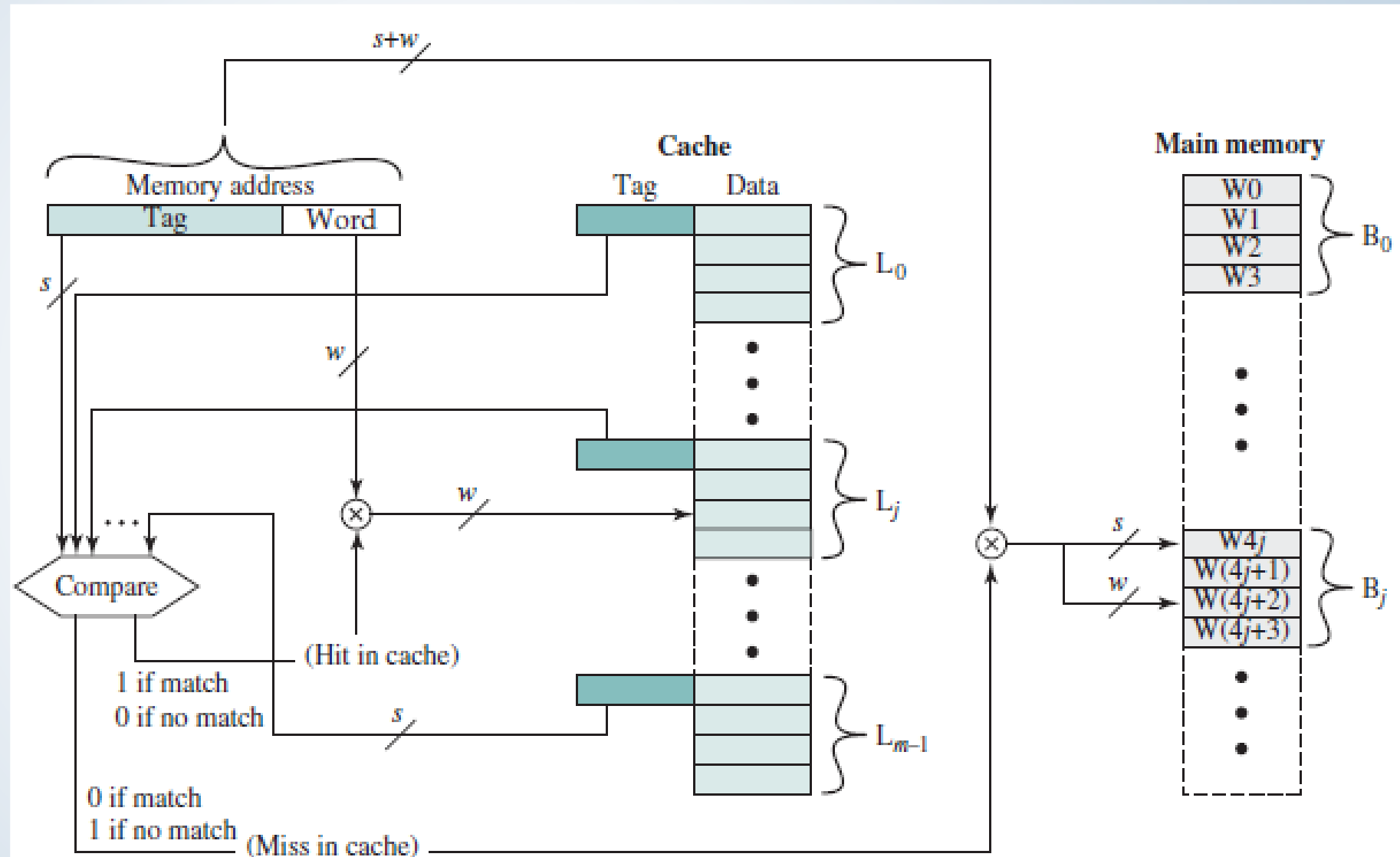
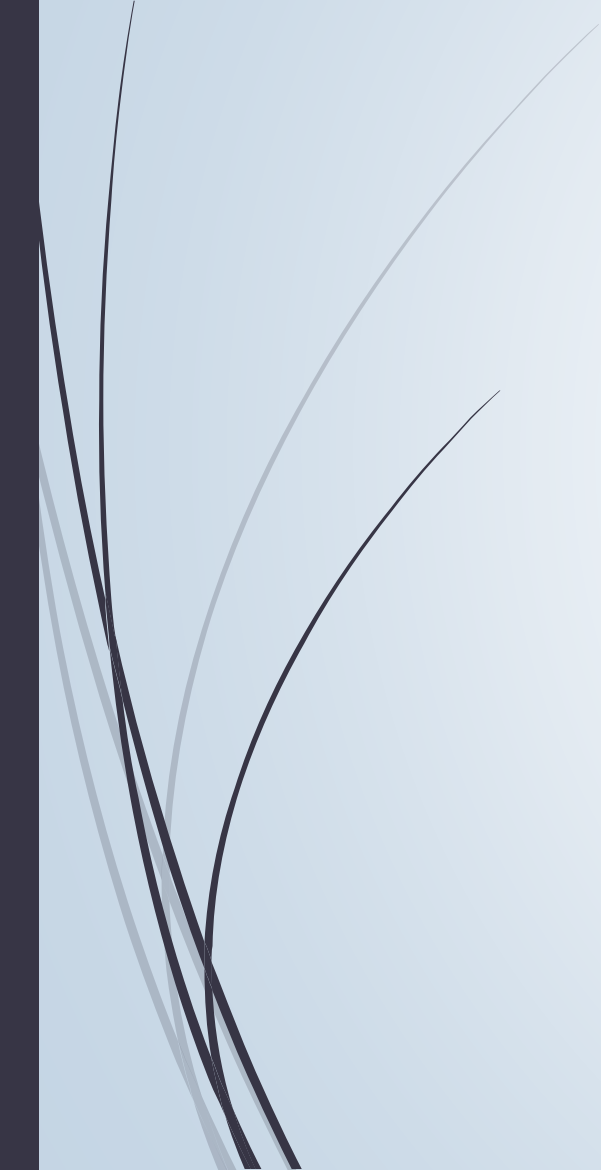


Figure 4.11 Fully Associative Cache Organization



Associative Mapping

- With associative mapping, there is **flexibility** as to which block to replace when a new block is read into the cache.
- The principal **disadvantage** of associative mapping is the **complex circuitry** required to examine the tags of all cache lines in **parallel**.

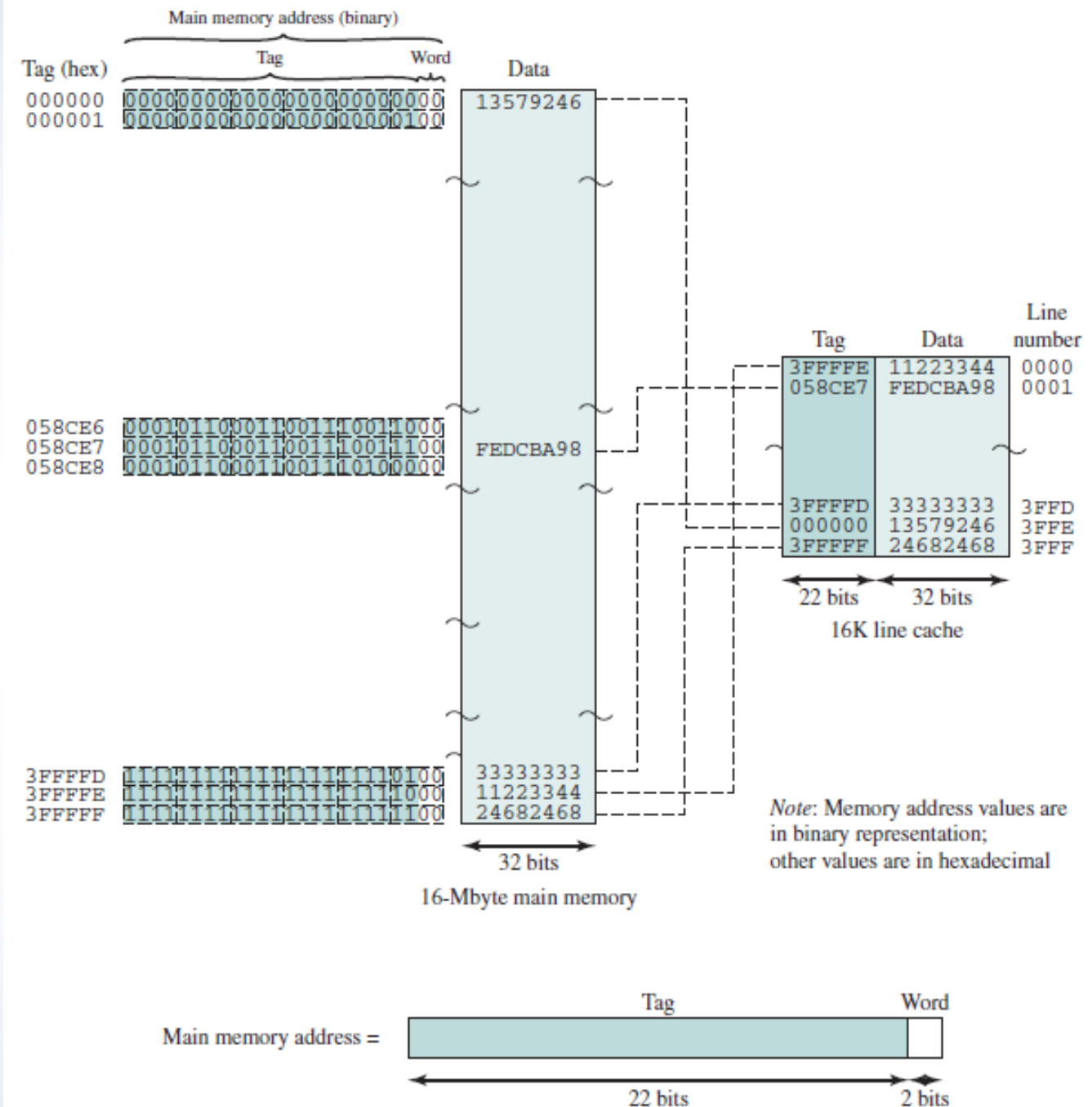


Figure 4.12 Associative Mapping Example

Associative mapping Question

- ▶ We have an 8KB of Associative cache. It has line size of 32 bytes. The main memory size is 1GB.
 - ▶ How many TAG bits required?
 - ▶ How many comparators required?
 - ▶ What is the comparator size in bits?
- ▶ If the Propagation Delay of a comparator is 10 ns for each TAG bit and Propagation delay of the OR gate is 30ns , then what is the hit latency?

k -way set-associative mapping

- Set-associative mapping is a compromise that exhibits the strengths of both the direct and associative approaches while reducing their disadvantages.
- k is the number of cache lines in each set
- v is the number of sets.
- the cache control logic interprets a memory address as three fields: **Tag**, **Set**, and **Word**
- For set-associative mapping, each Word maps into all the cache lines in a specific set, so that main memory block B0 maps into set 0, and so on.
 - Thus, the set-associative cache can be physically implemented as n associative caches.

→ k-way set associative mapping

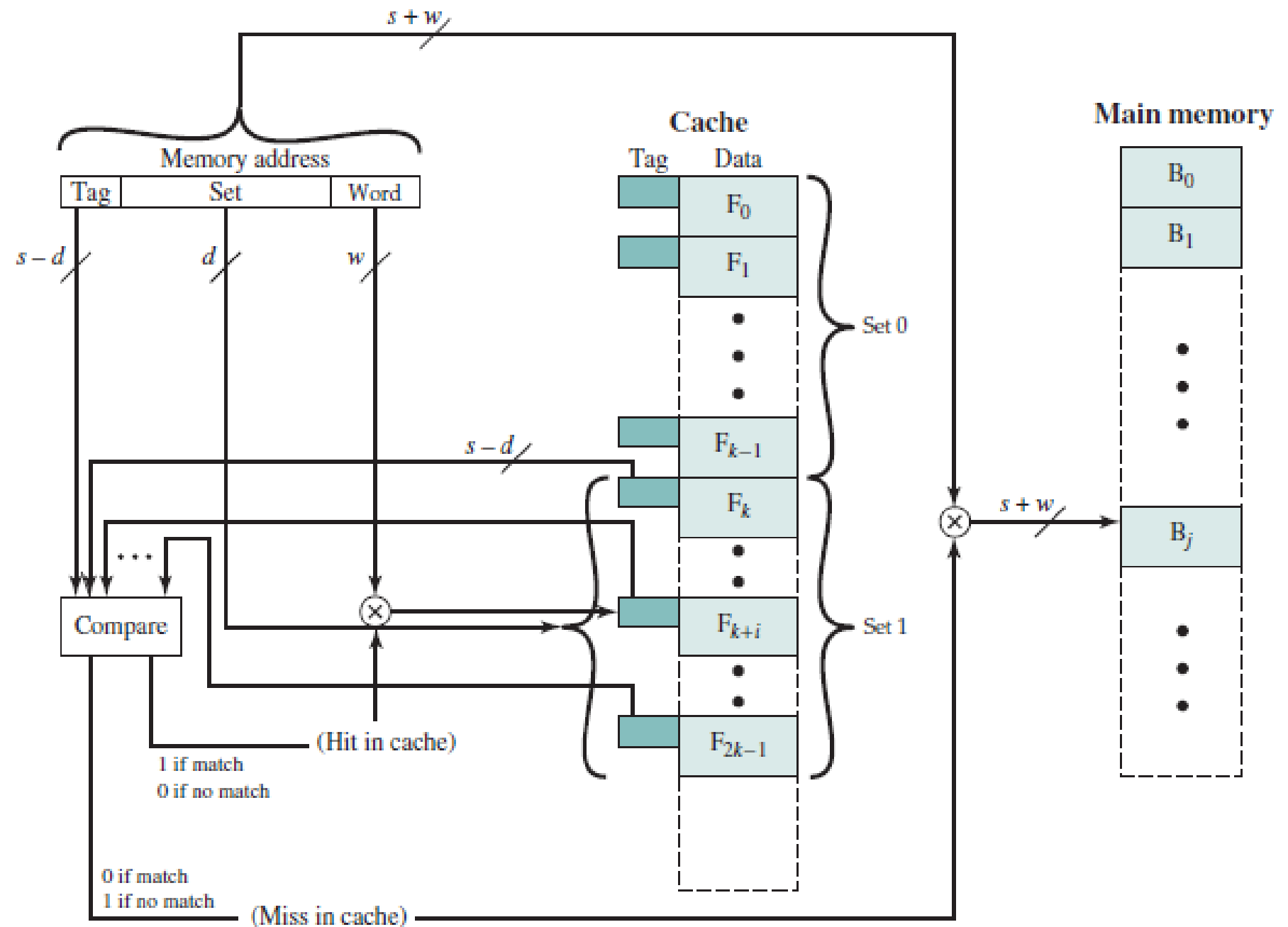
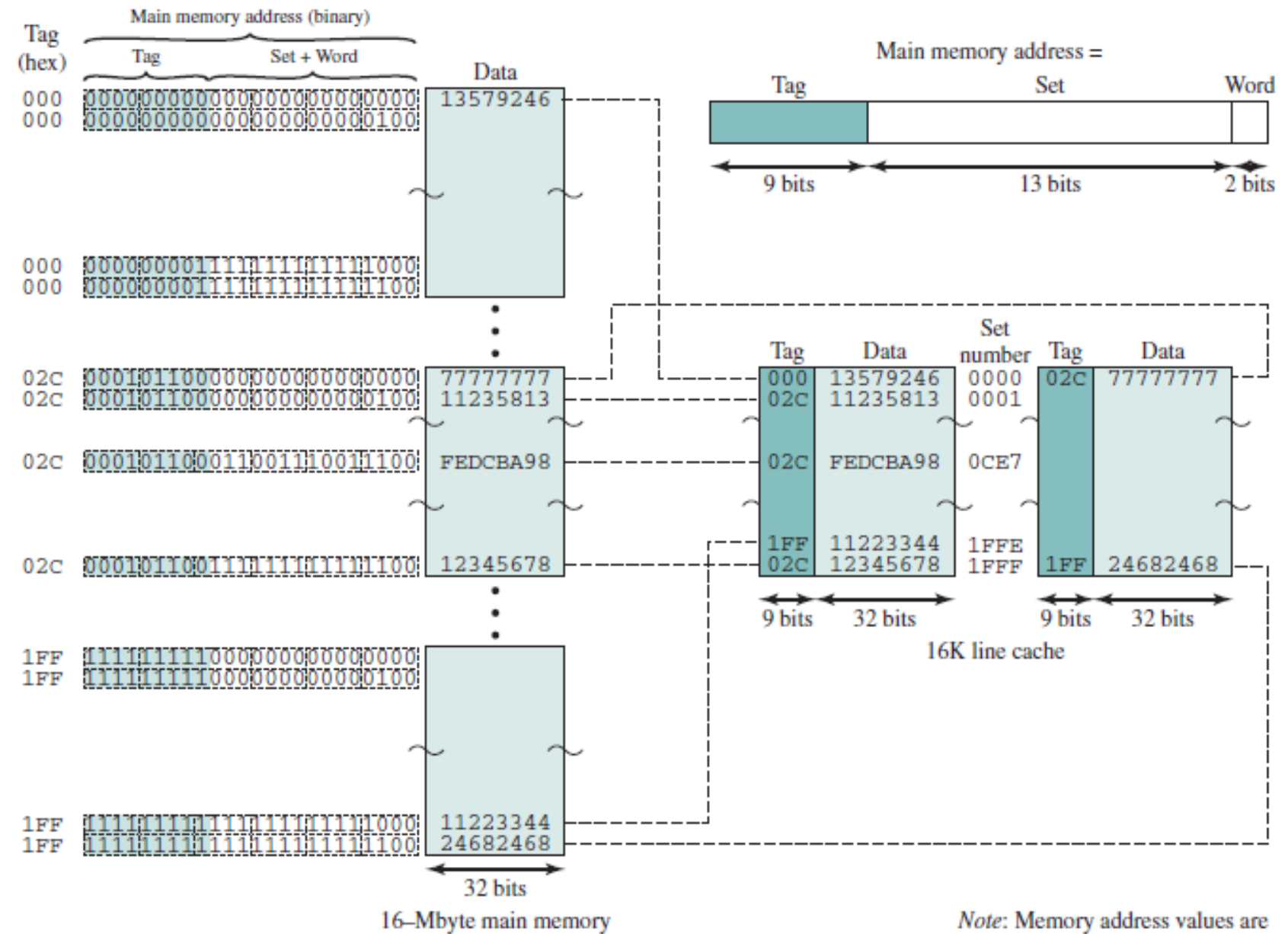


Figure 4.14 K-Way Set Associative Cache Organization



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

Figure 4.15 Two-Way Set-Associative Mapping Example

Sample Question

Block size = 8

Q 7. (15pts) Consider a machine with a byte addressable main memory of 2^{16} bytes and block size of 8 bytes. Assume that a direct mapped cache consisting of 32 lines is used with this machine.

→ a) How is a 16-bit memory address divided into tag, line number, and byte number?



→ b) Into what line would bytes with each of the following addresses be stored?

→

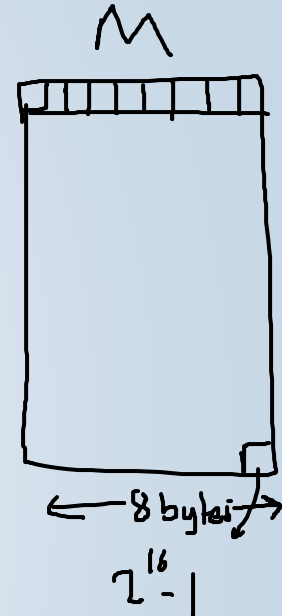
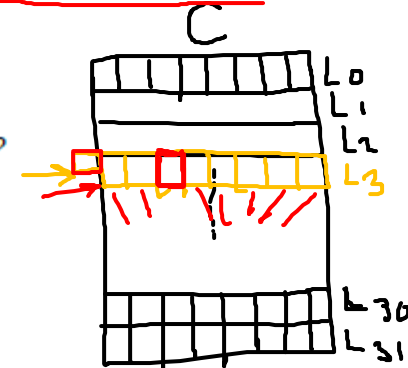
0001 0001 0001 1011	→	00011	→	L3
1100 0011 0011 0100	→	00110	→	L6
1101 0000 0001 1101	→	00011	→	L3
1010 1010 1010 1010	→	10101	→	L21

→ c) Suppose the byte with address 0001 1010 0001 1010 is stored in the cache. What are the addresses of the other bytes stored along with it?

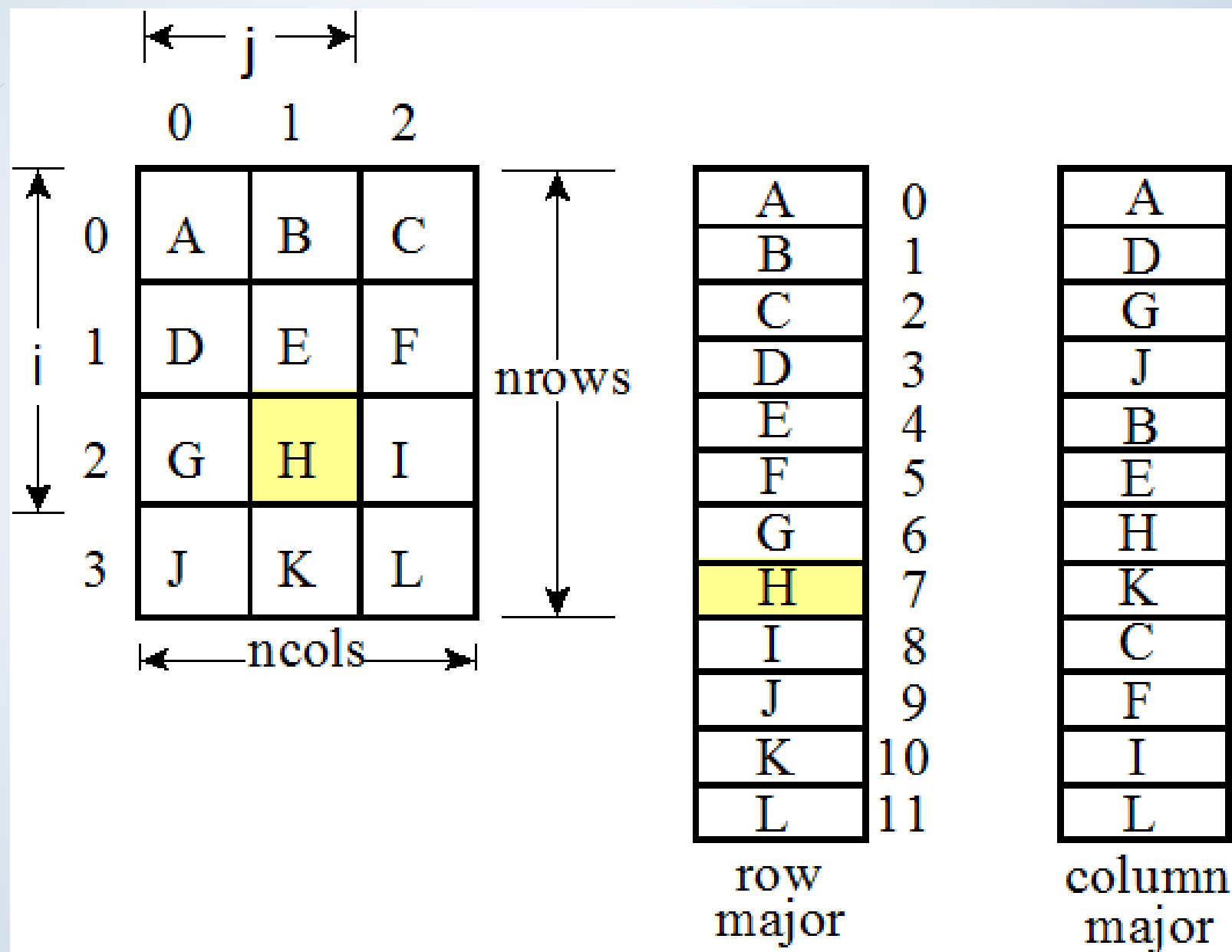
0001 1010 0001 1000 -
 " " " " 001 -
 " " " " 011 -
 " " " " 100 -
 " " " " 101 -
 " " " " 110 -
 " " " " 111 -

→ d) How many total bytes of memory can be stored in the cache?

32 lines x 8 bytes
 = 256 bytes



Row major ordering



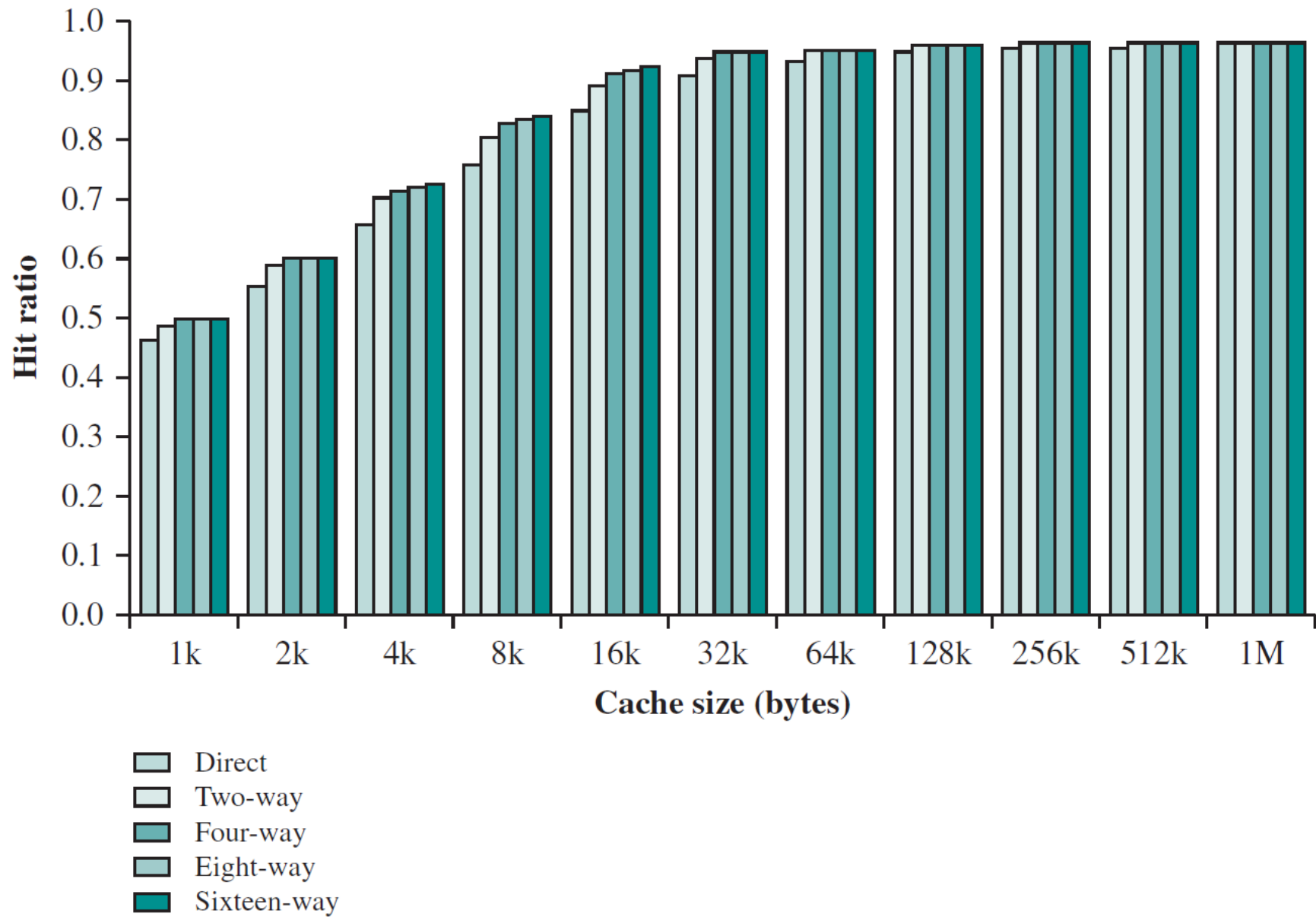


Figure 4.16 Varying Associativity over Cache Size

Replacement Algorithms

- Once the cache has been filled, when a new block is brought into the cache, one of the existing blocks must be replaced
- For direct mapping there is only one possible line for any particular block and no choice is possible
- For the associative and set-associative techniques a replacement algorithm is needed
- To achieve high speed, an algorithm must be implemented in hardware

The four most common replacement algorithms are:

➤ Least recently used (LRU)

- Most effective
- Replace that block in the set that has been in the cache longest with no reference to it
- Each line includes a USE bit
- Because of its simplicity of implementation, LRU is the most popular replacement algorithm

➤ First-in-first-out (FIFO)

- Replace that block in the set that has been in the cache longest
- Easily implemented as a round-robin or circular buffer technique

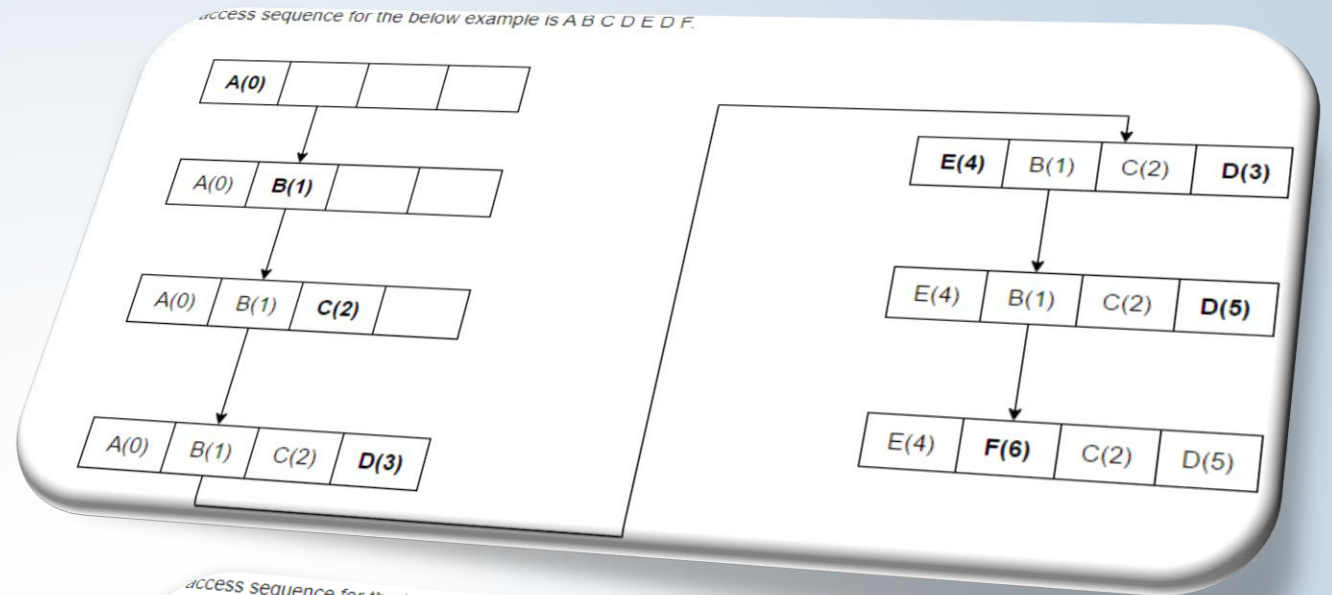
➤ Least frequently used (LFU)

- Replace that block in the set that has experienced the fewest references
- Could be implemented by associating a counter with each line

LRU vs MRU

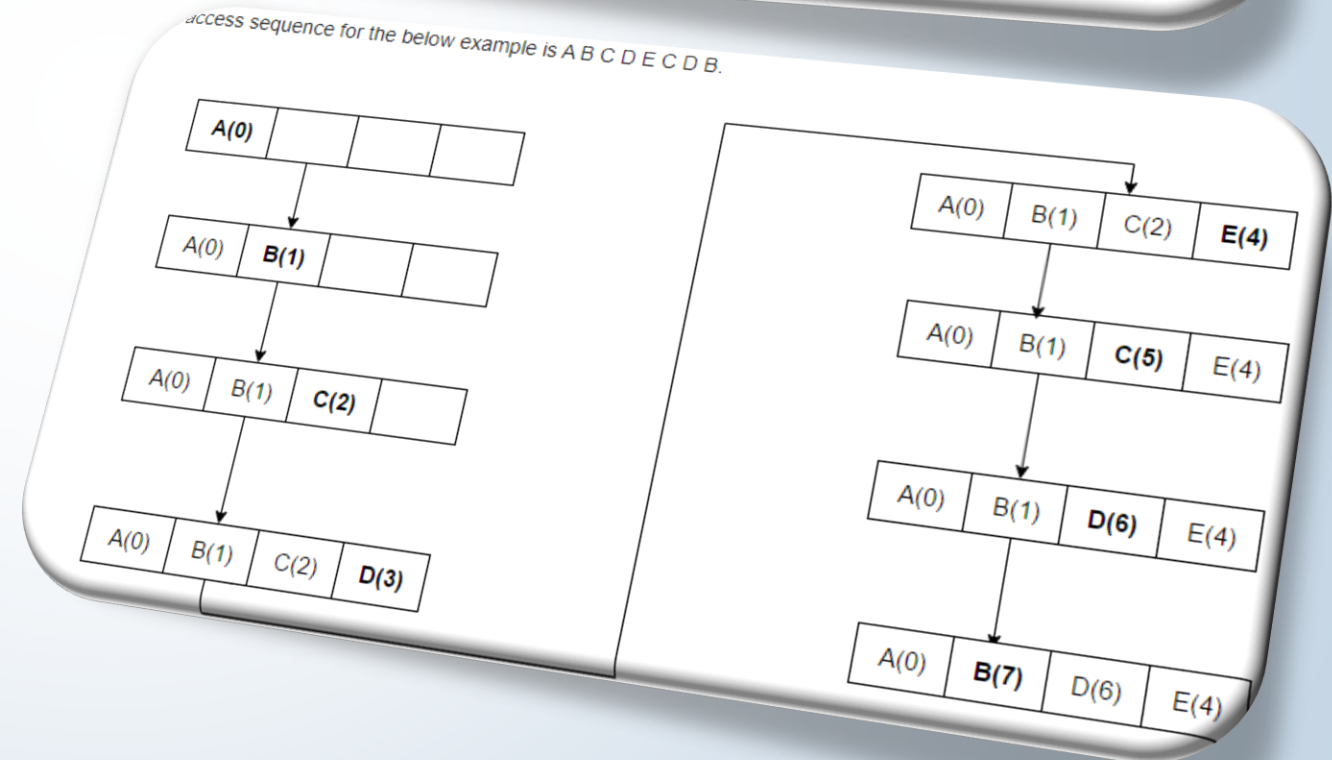
LRU

→ A B C D E D F



MRU

→ A B C D E C D B



Write Policy

When a block that is resident in the cache is to be replaced there are two cases to consider:



If the old block in the cache has not been altered then it may be overwritten with a new block without first writing out the old block



If at least one write operation has been performed on a word in that line of the cache then main memory must be updated by writing the line of cache out to the block of memory before bringing in the new block

There are two problems to contend with:



More than one device may have access to main memory



A more complex problem occurs when multiple processors are attached to the same bus and each processor has its own local cache - if a word is altered in one cache it could conceivably invalidate a word in other caches

Write Through and Write Back

➤ Write through

- Simplest technique
- All write operations are made to main memory as well as to the cache
- The main disadvantage of this technique is that it generates substantial memory traffic and may create a bottleneck

➤ Write back

- Minimizes memory writes by using **dirty bit**
- Updates are made only in the cache
- Portions of main memory are invalid and hence accesses by I/O modules can be allowed only through the cache
- This makes for complex circuitry and a potential bottleneck

Possible approaches to cache coherency:

➤ Bus watching with write through:

- Each cache controller monitors the address lines to detect write operations to memory by other bus masters.
- If another master writes to a location in shared memory that also resides in the cache memory, the cache controller invalidates that cache entry.
- This strategy depends on the use of a write-through policy by all cache controllers.

➤ Hardware transparency:

- Additional hardware is used to ensure that all updates to main memory via cache are reflected in all caches.
- Thus, if one processor modifies a word in its cache, this update is written to main memory. In addition, any matching words in other caches are similarly updated.

➤ Non-cacheable memory:

- Only a portion of main memory is shared by more than one processor, and this is designated as non-cacheable.
- In such a system, all accesses to **shared memory** are **cache misses**, because the shared memory is never copied into the cache. The non-cacheable memory can be identified using chip-select logic or high-address bits.

Line Size

When a block of data is retrieved and placed in the cache not only the desired word but also some number of adjacent words are retrieved

As the block size increases more useful data are brought into the cache

Two specific effects come into play:

- Larger blocks reduce the number of blocks that fit into a cache
- As a block becomes larger each additional word is farther from the requested word

As the block size increases the hit ratio will at first increase because of the principle of locality

The hit ratio will begin to decrease as the block becomes bigger and the probability of using the newly fetched information becomes less than the probability of reusing the information that has to be replaced

Hit Ratio (L1 & L2) For 8 Kbyte and 16 Kbyte L1

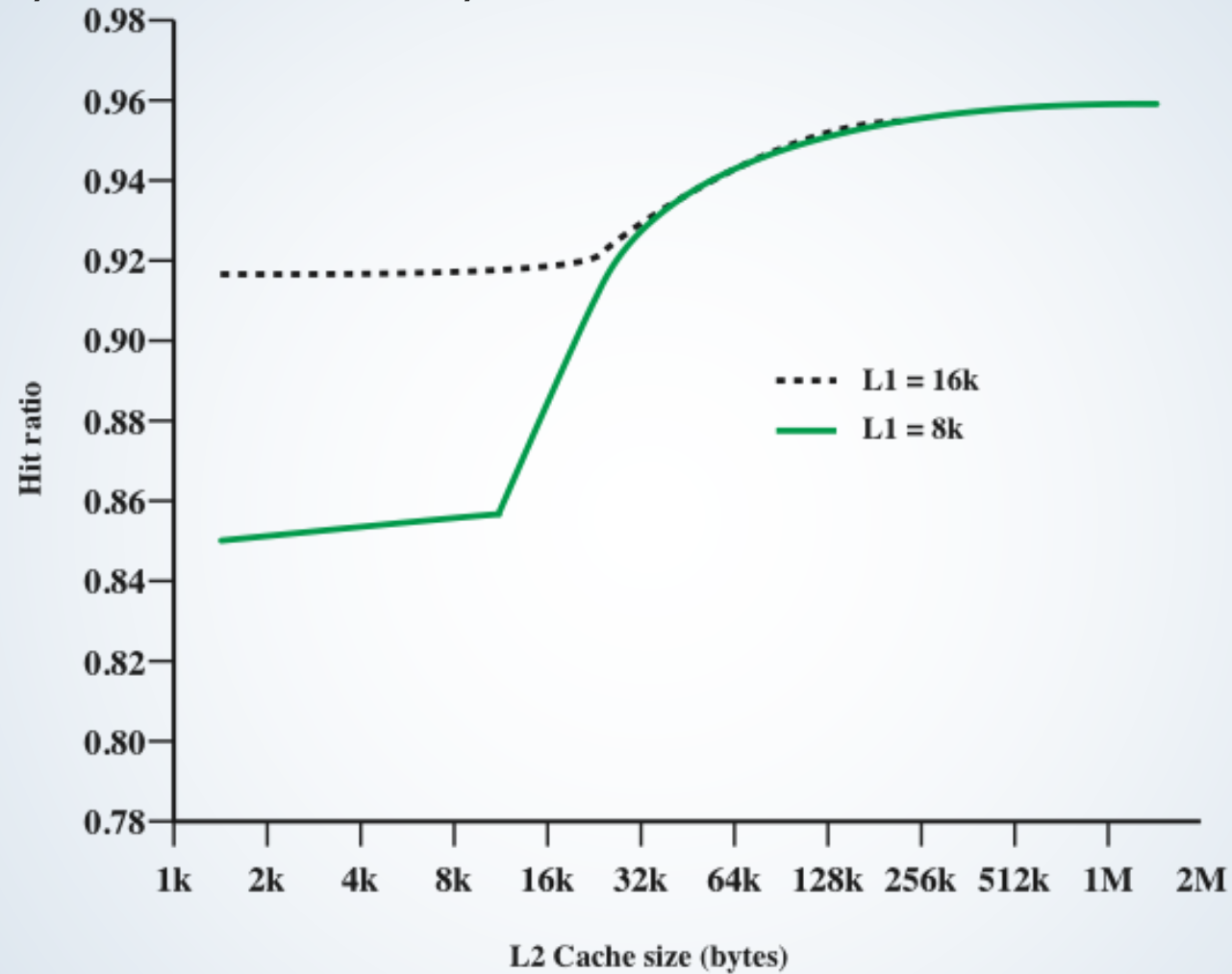


Figure 4.17 Total Hit Ratio (L1 and L2) for 8 Kbyte and 16 Kbyte L1

Summary

- Characteristics of Memory Systems

- Location
- Capacity
- Unit of transfer

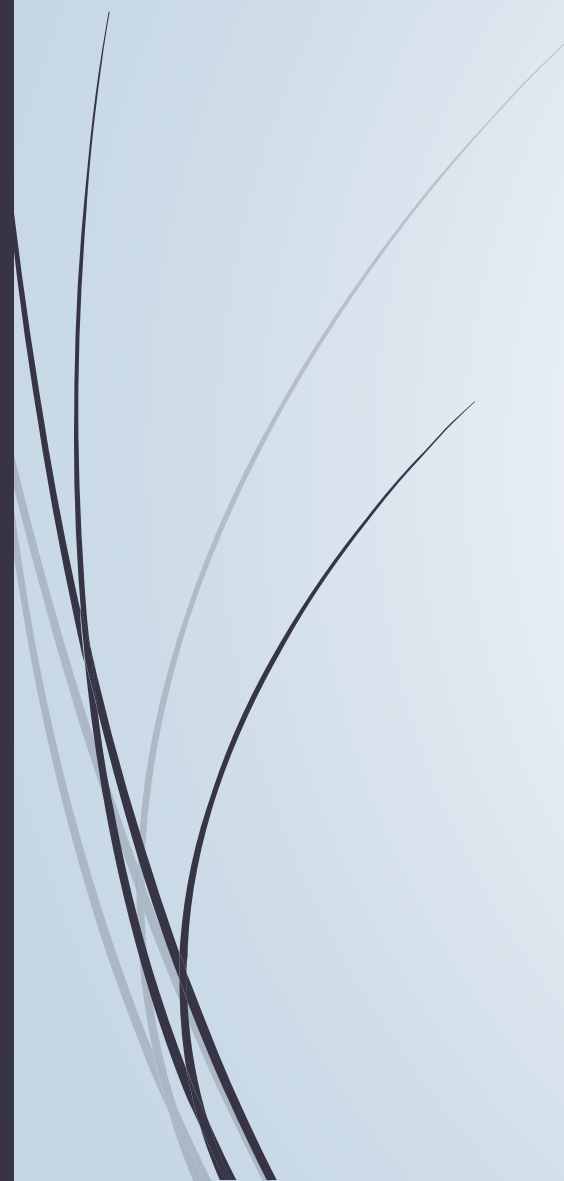
- Memory Hierarchy

- How much?
- How fast?
- How expensive?

- Cache memory principles

- Elements of cache design

- Cache addresses
- Cache size
- Mapping function
- Replacement algorithms
- Write policy
- Line size
- Number of caches



Dynamic RAM (DRAM)

- RAM technology is divided into two technologies:
 - Dynamic RAM (DRAM)
 - Static RAM (SRAM)
- DRAM
 - Made with cells that store data as charge on capacitors
 - Presence or absence of charge in a capacitor is interpreted as a binary 1 or 0
 - **Requires periodic charge refreshing** to maintain data storage
 - The term *dynamic* refers to tendency of the stored charge to leak away, even with power continuously applied

SRAM *versus* DRAM

- Both volatile

- Power must be continuously supplied to the memory to preserve the bit values

- Dynamic cell

- Simpler to build, smaller
 - More dense (smaller cells = more cells per unit area)
 - Less expensive
 - Requires the supporting refresh circuitry
 - Tend to be favored for large memory requirements
 - **Used for main memory**

- Static

- Faster
 - **Used for cache memory** (both on and off chip)

Read Only Memory (ROM)

- Contains a permanent pattern of data that cannot be changed or added to
- No power source is required to maintain the bit values in memory
- Data or program is permanently in main memory and never needs to be loaded from a secondary storage device
- Data is actually wired into the chip as part of the fabrication process
 - **Disadvantages** of this:
 - **No room for error**, if one bit is wrong the whole batch of ROMs must be thrown out
 - Data insertion step includes a relatively large fixed **cost**

Programmable ROM (PROM)

- Less expensive alternative
- Nonvolatile and may be written into only once
- Writing process is performed electrically and may be performed by supplier or customer at a time later than the original chip fabrication
- Special equipment is required for the writing process
- Provides flexibility and convenience
- Attractive for high volume production runs

Typical 16 Mbit DRAM (4M x 4)

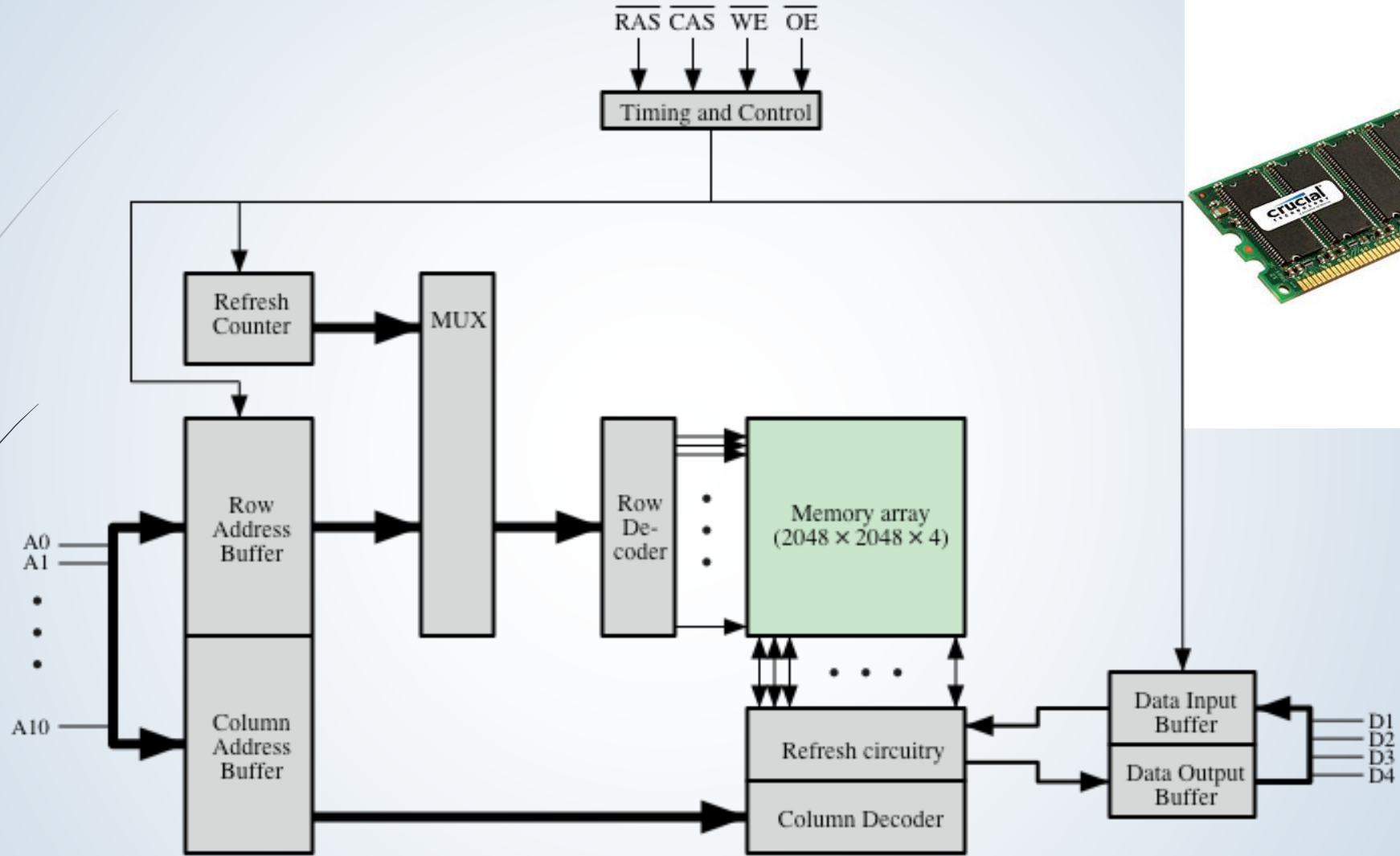
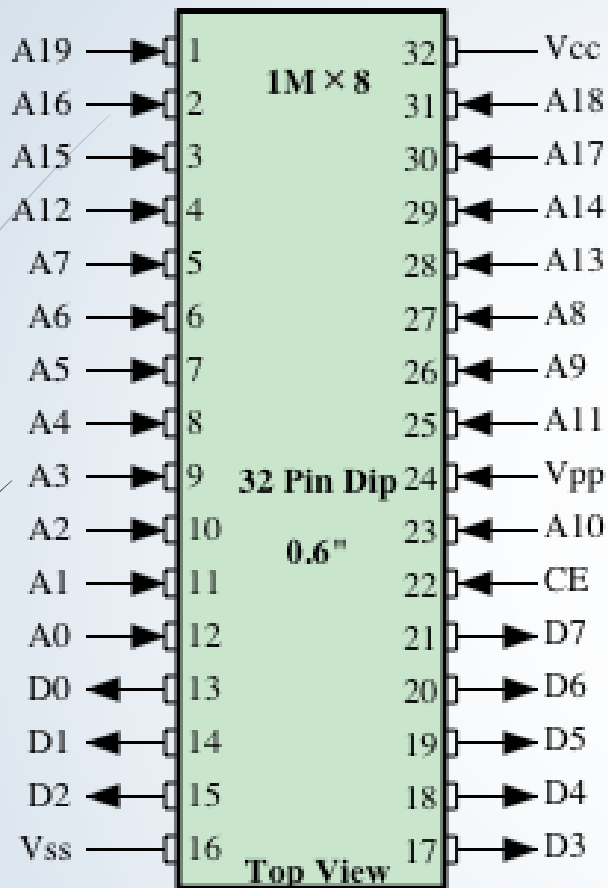
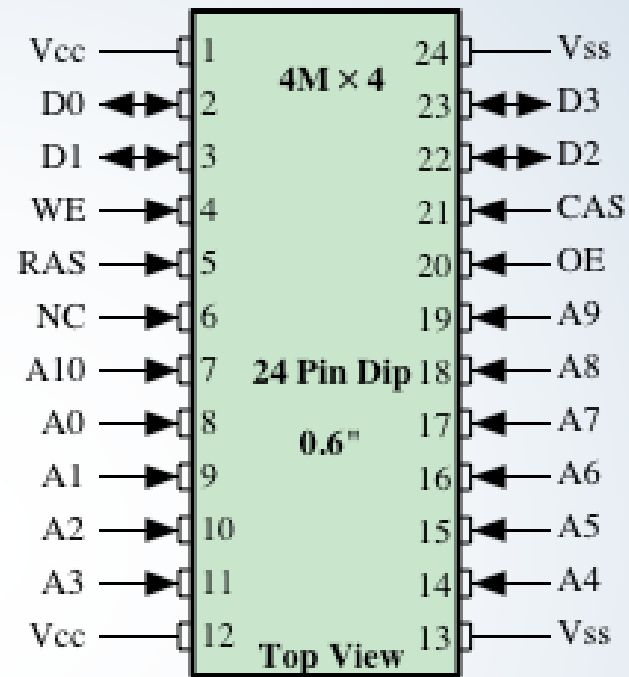


Figure 5.3 Typical 16 Megabit DRAM (4M x 4)

Chip Packaging



(a) 8 Mbit EPROM



(b) 16 Mbit DRAM

Figure 5.4 Typical Memory Package Pins and Signals

Error Correction

➤ Hard Failure

- Permanent physical defect
- Memory cell or cells affected cannot reliably store data but become stuck at 0 or 1 or switch erratically between 0 and 1
- Can be caused by:
 - Harsh environmental abuse
 - Manufacturing defects
 - Wear

➤ Soft Error

- Random, non-destructive event that alters the contents of one or more memory cells
- No permanent damage to memory
- Can be caused by:
 - Power supply problems
 - Alpha particles

Error Correcting Code Function

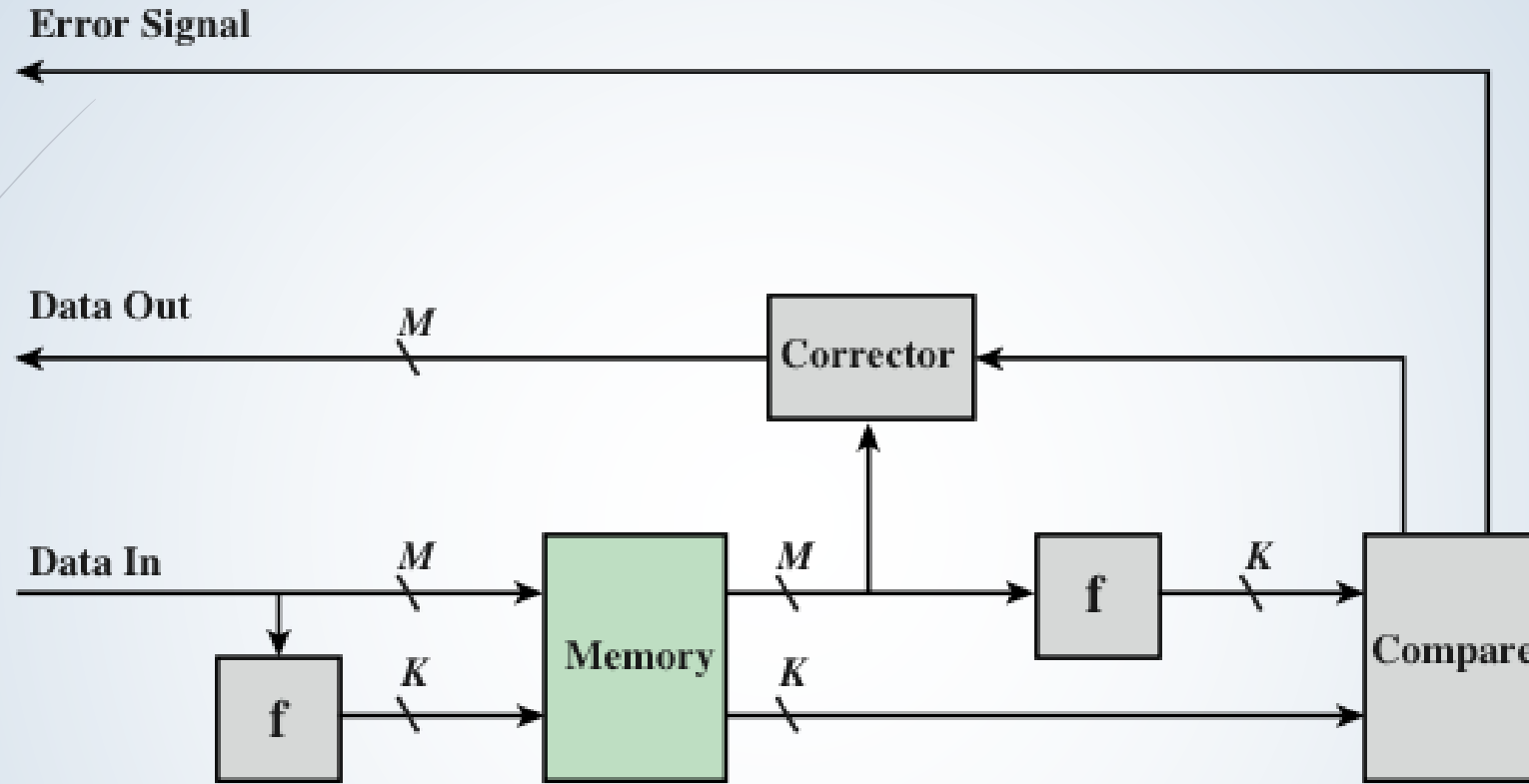


Figure 5.7 Error-Correcting Code Function

Hamming Error Correcting Code

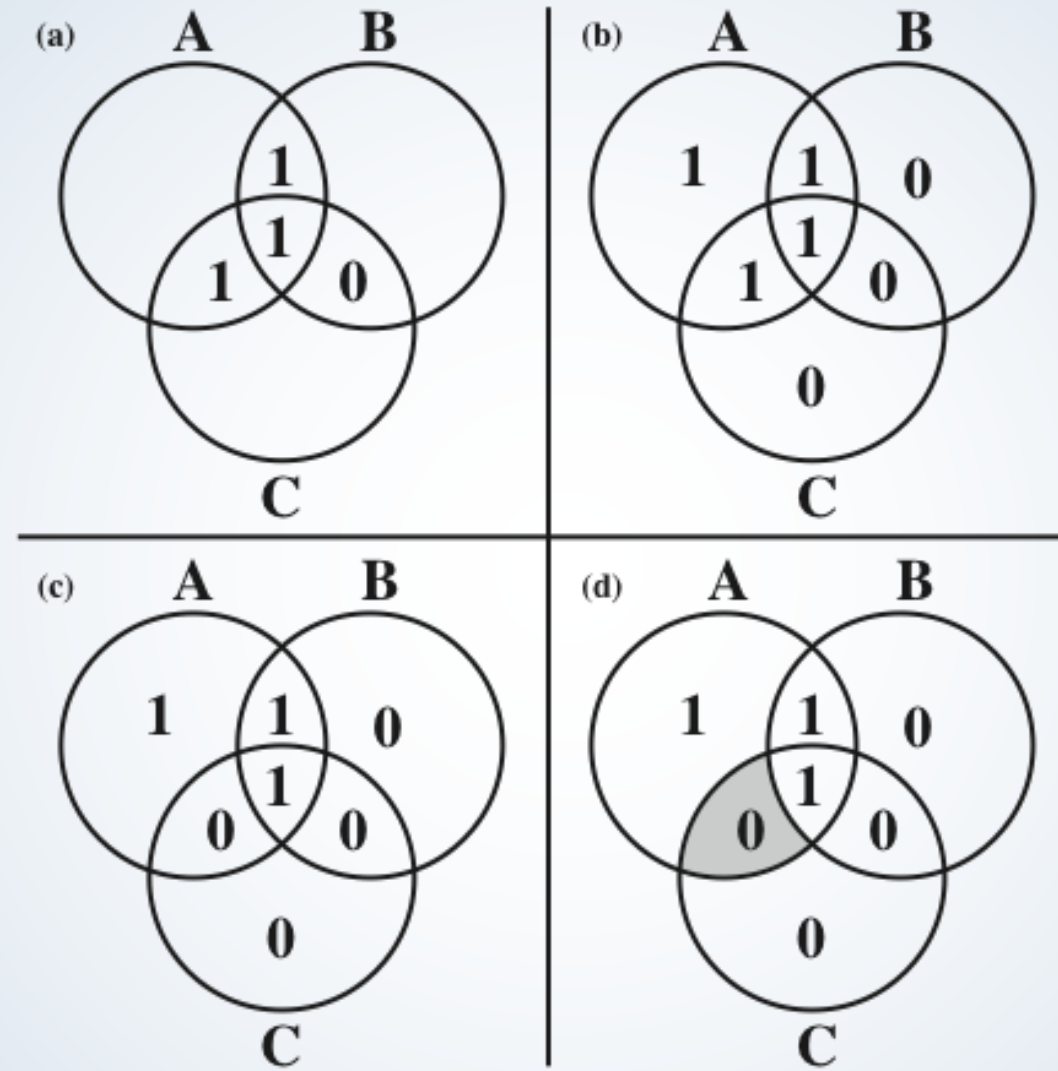


Figure 5.8 Hamming Error-Correcting Code

Hamming Error Correcting Code

➔ $p_1 p_2 d_3 p_4 d_5 d_6 d_7 p_8 d_9 d_{10} d_{11} d_{12}$

➔ Rules

➔ p_1 (one check one skip)

➔ p_2 (two check two skip)

➔ p_4 (four check four skip)

➔ p_8 (eight check eight skip)

➔ Error is at the binary address $p'_8 p'_4 p'_2 p'_1$ where p'_x is 1 when there is an error (computed vs received parity) otherwise 0.

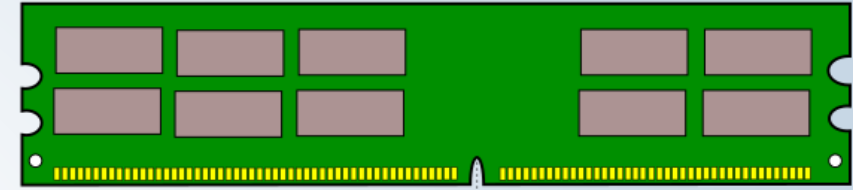
➔ Example Data = 11101001

➔ $p_1=1 p_2=0 p_4=1 p_8=0$

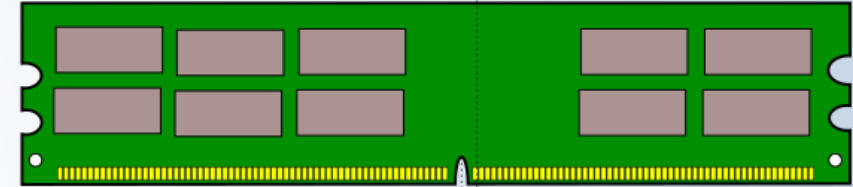
Double Data Rate SDRAM (DDR SDRAM)

- ➔ SDRAM can only send data once per bus clock cycle
- ➔ Double-data-rate SDRAM can send data twice per clock cycle, once on the rising edge of the clock pulse and once on the falling edge
- ➔ Developed by the JEDEC Solid State Technology Association (Electronic Industries Alliance's semiconductor-engineering-standardization body)

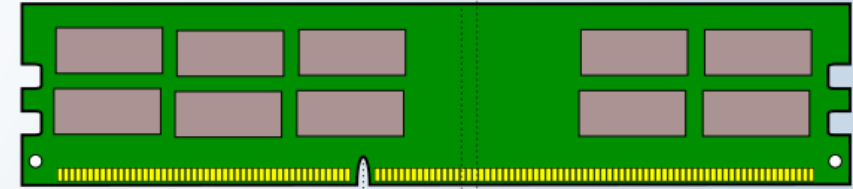
DDR



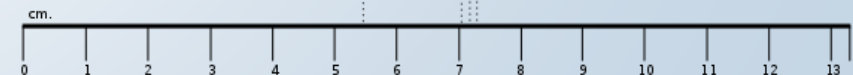
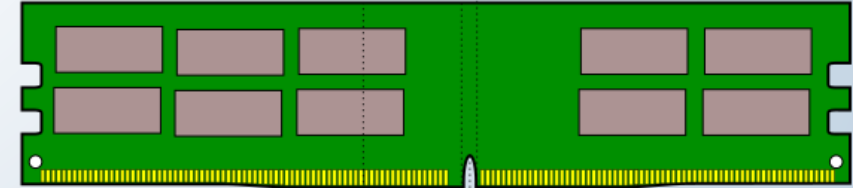
DDR 2



DDR 3

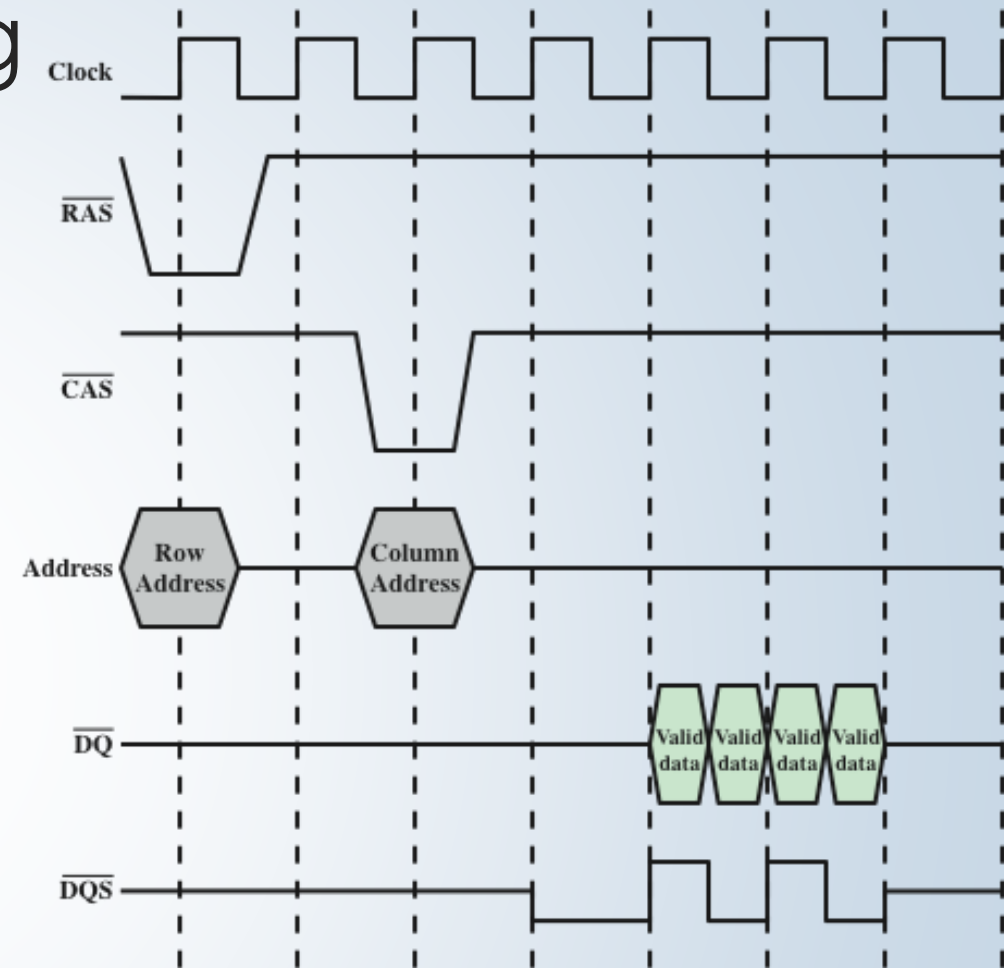


DDR 4



DDR SDRAM Read Timing

- ➔ DDR2 increases the data transfer rate by increasing the operational frequency of the RAM chip and by increasing the prefetch buffer from 2 bits to 4 bits per chip.
- ➔ DDR3, introduced in 2007, increases the prefetch buffer size to 8 bits.



RAS = row address select
CAS = column address select
DQ = data (in or out)
DQS = DQ select

Figure 5.15 DDR SDRAM Read Timing