COA

Chapter 04: | Mapping Function

Prof. Dr. Mostafa Elhosseini

https://youtube.com/drmelhosseini

AGENDA

Elements of Cache Design

- Mapping Function
  - **Direct**
  - Associative
  - Set Associative

Mapping Function

- There **are fewer cache lines** than main memory blocks, an **algorithm** is needed for mapping main memory blocks into cache lines.
  - Need to determine **which main memory block** currently **occupies** a cache line
- The **choice of the mapping function** dictates how the cache is **organized**.
- Mapping Function
  - **Direct**
  - Associative
  - Set Associative

Cache

Block ⇒ memory

Direct

Direct Mapping

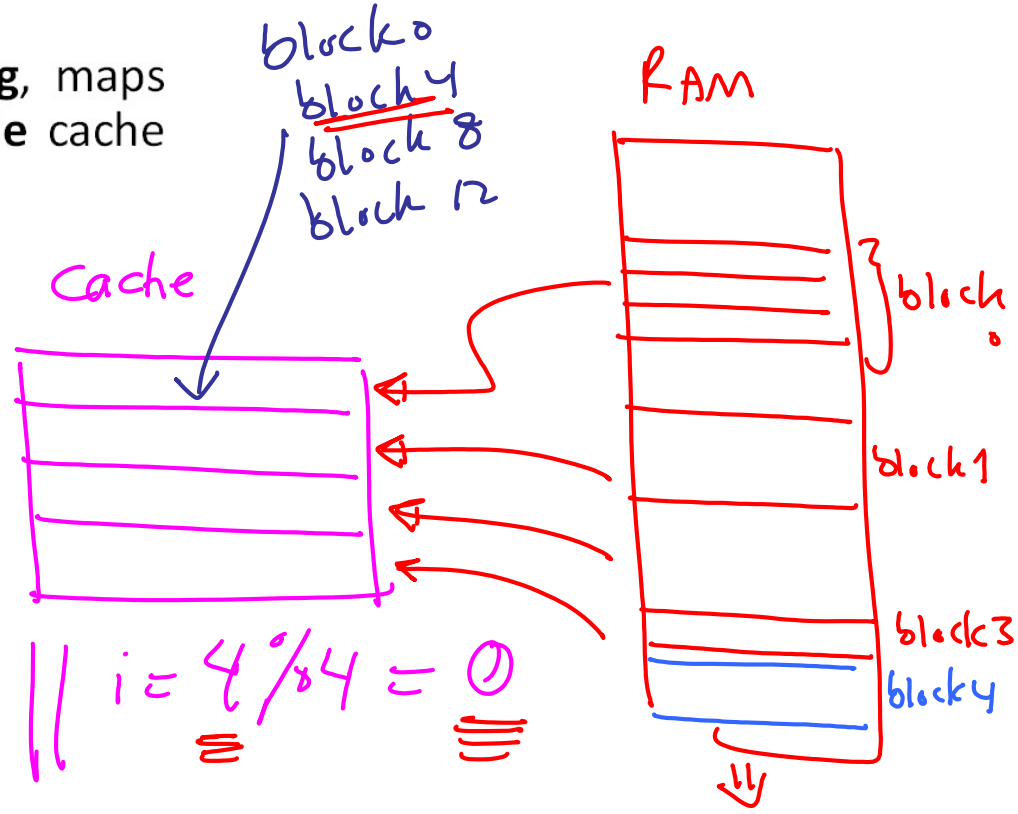
- The **simplest** technique, known as **direct mapping**, maps each **block** of main memory into **only one possible** cache line
- The mapping is expressed as:

$$i = j \ modulo \ m$$

Remainder

Where:

- i cache line number
- j main memory block number
- m number of lines in the cache



$i = 0 \% 4 = 0$

$i = 1 \% 4 = 1$

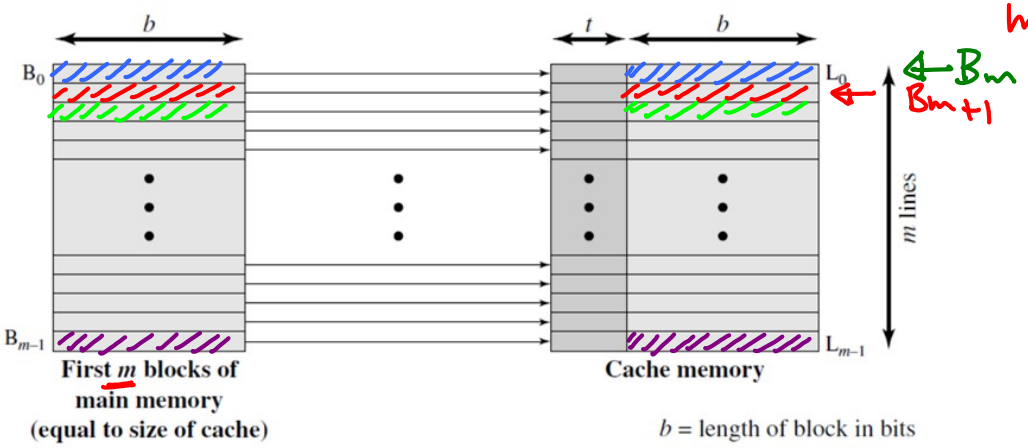
$i = 2 \% 4 = 2$

$i = 3 \% 4 = 3$

$i = 4 \% 4 = 0$

Direct Mapping

- Each block of main memory maps into one unique line of the cache.
- The next m blocks of main memory map into the cache in the same fashion; that is, block B_m of main memory maps into line L_0 of cache, block B_{m+1} maps into line L1, and so on



tag

m ⇒ # block

m ⇒ # cache line

B_m

B_{m+1}

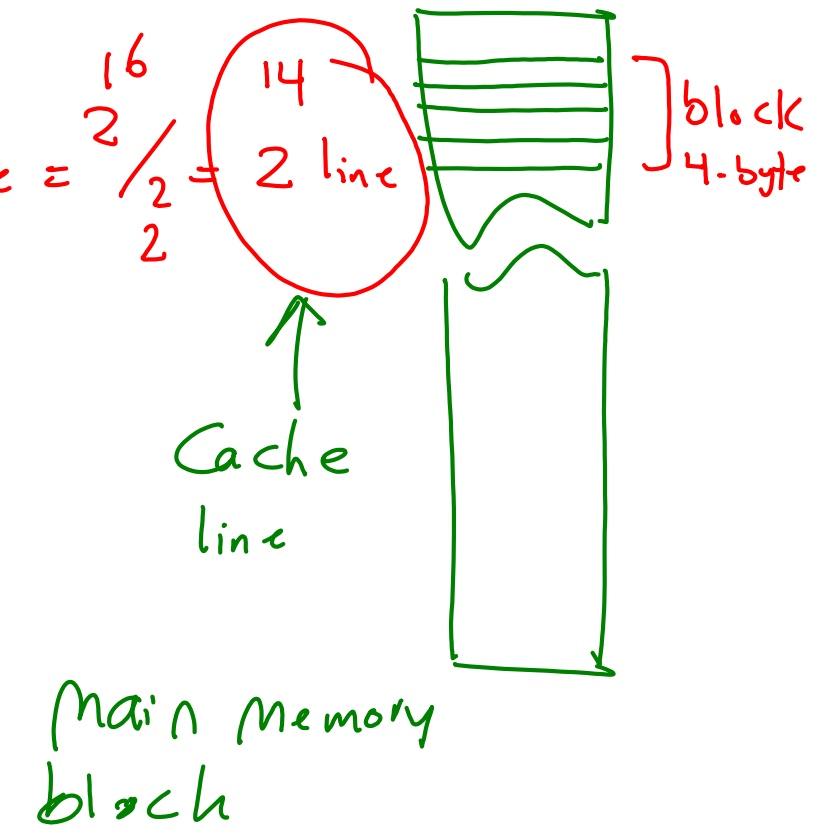
Direct Mapping

line size = Block size = 4 byte

- Cache size : 64 Kbytes. $= 2^{16}$ byte
- Block size: 4 bytes.
- Cache is organized as $16\,K = 2^{14}$ lines of 4 bytes each.
- Memory Size: 16 Mbytes, with each byte directly addressable by a 24-bit address ($2^{14} = 16\,M$).
  - Main memory: 4M blocks of 4 bytes each

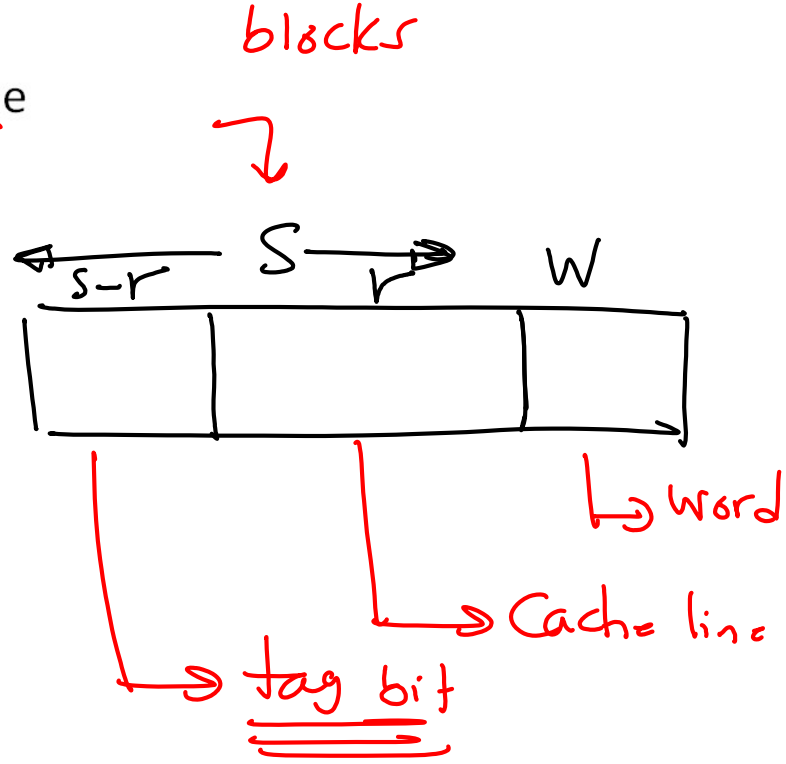
$\#\text{ cache line} = 2^{16}/2^{2} = 2^{14}$ line

$\#\text{ blocks} = 2^{24}/4 = 2^{22}$ block



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
  - w: Least Significant bits identify **unique word**
  - s: Most Significant bits specify **one memory block**
  - $s - bits$ are split into
    - $s - r$: Tag bits
    - r: Cache line



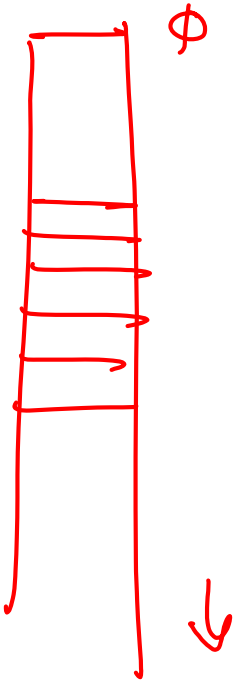
Direct Mapping

To Summarize:

- 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: $\frac{2^{s+w}}{2^{w}} = 2^{s}$
- Number of lines in cache = $m = 2^{r}$
- Size of cache: 2^{r+w} words or bytes
- Size of tag: $s - r$ bits

memory

s → block



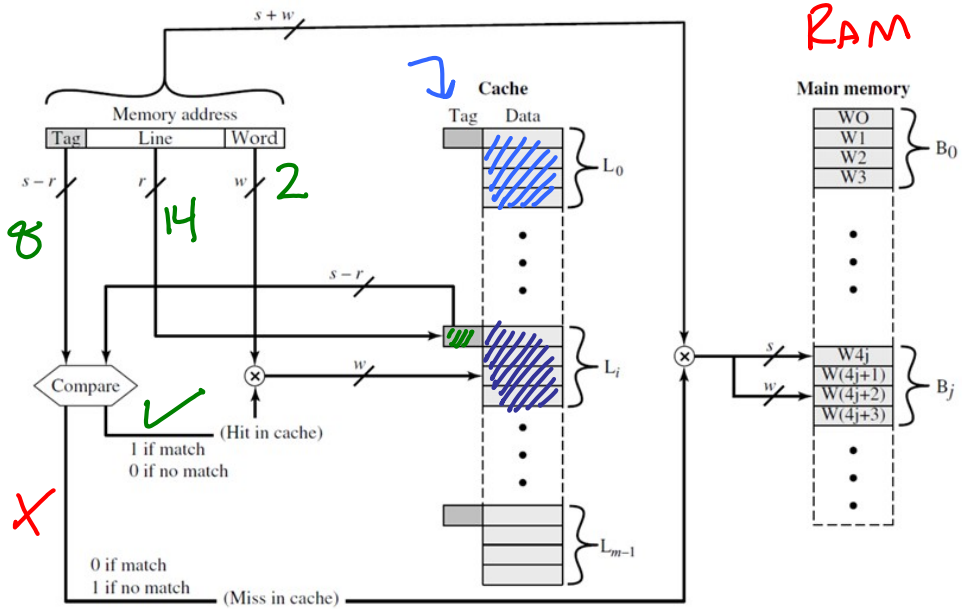
Direct Mapping

Cache Line	Main memory blocks assigned
0	0, m, 2m, 3m, …, $2^s - m$
1	1, m+1, 2m+1, 3m+1, …, $2^s - m+1$
2	2, m+2, 2m+2, 3m+2, …, $2^s - m+2$
. . .	
m - 1	m -1 , 2m-1, 3m-1, 4m-1, …, $2^s - 1$

tag

Count = m → # cache line

Direct Mapping



* Memory size = 16 Mbyte
$= 2^{24}$

Address = 24-bit = S+W

* Block size = 4 word

W = 2

* S = 22-bit

* Cache = 2^{16}

line = $2^{16}/2^{2} = 2^{14}$

r

Direct Mapping

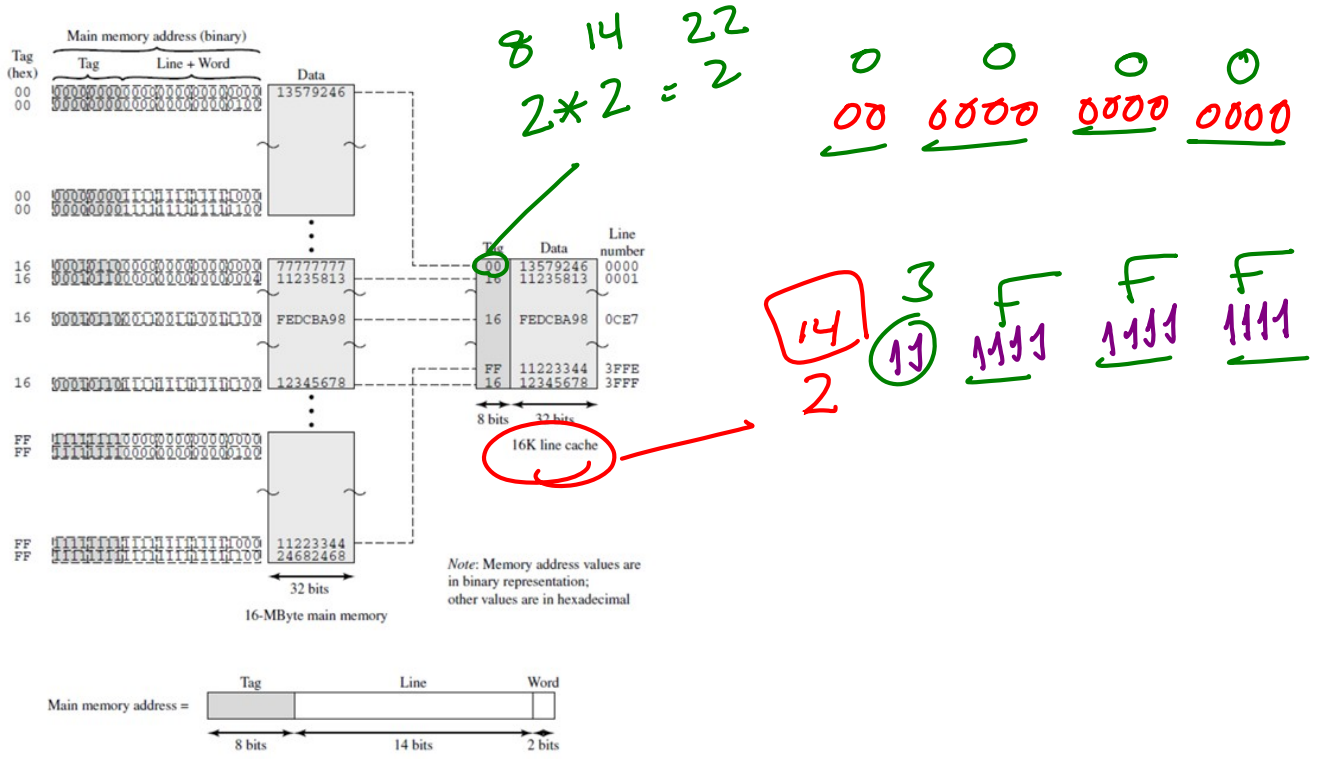
- Address length: $s + w$ bits
- Number of addressable units: 2^{s+w} words or bytes
- Block size = line size = 2^{w} words or bytes
- Number of blocks in main memory: $\frac{2^{s+w}}{2^{w}} = 2^{s}$
- Number of lines in cache = $m = 2^{r}$
- Size of cache: 2^{r+w} words or bytes
- Size of tag: $s - r$ bits

Example

- Cache size : 64 Kbytes.
- Block size: 4 bytes.
- Cache is organized as $16\ K = \ 2^{14}$ lines of 4 bytes each.
- Memory Size: 16 Mbytes, with each byte directly addressable by a 24-bit address ($2^{14} = 16\ M$).
  - Main memory: 4M blocks of 4 bytes each

Example

Example


Main memory address (binary)
Tag
Line + Word
Tag (hex)
Data
13579246
77777777
11235813
FEDCBA98
12345678
11223344
24682468
32 bits
16-MByte main memory
Tag
Data
Line number
00 13579246 0000
16 11235813 0001
16 FEDCBA98 0CE7
FF 11223344 3FFE
16 12345678 3FFF
8 bits
32 bits
16K line cache
Note: Memory address values are in binary representation; other values are in hexadecimal
Main memory address =
Tag
Line
Word
8 bits
14 bits
2 bits

Direct Mapping Pros & Cons

- Simple and inexpensive to implement.
- Its main disadvantage is that there is a **fixed cache location** for any given block. Thus, if a program happens to reference words **repeatedly** from **two different blocks** that **map into the same line**, then the blocks will be continually swapped in the cache, and the **hit ratio will be low** (a phenomenon known as **thrashing**).
  - Remember what was discarded in case it is needed again
  - Victim Cache (Victim cache is a fully associative cache)

Assigned

Miss