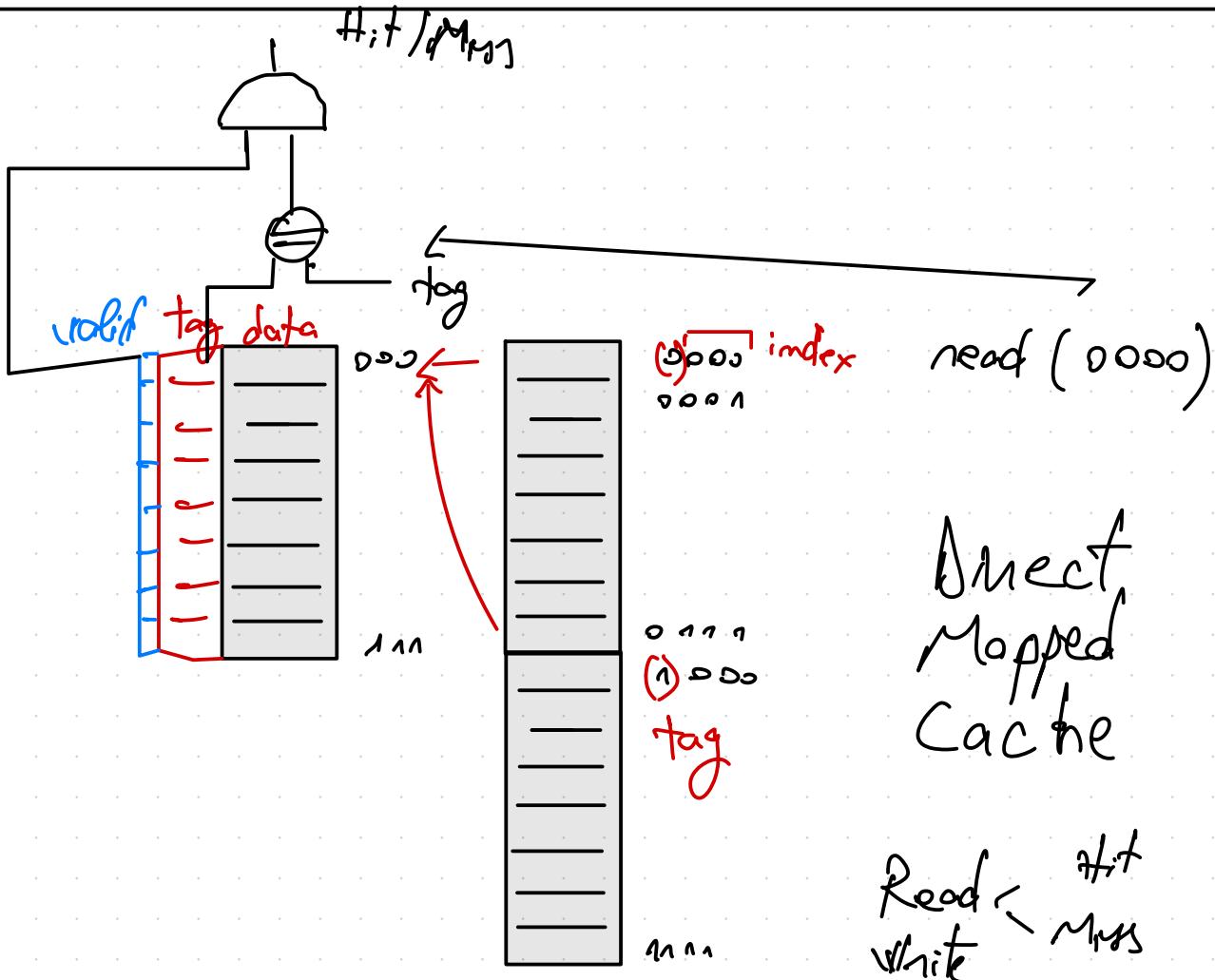


# Cache Structure

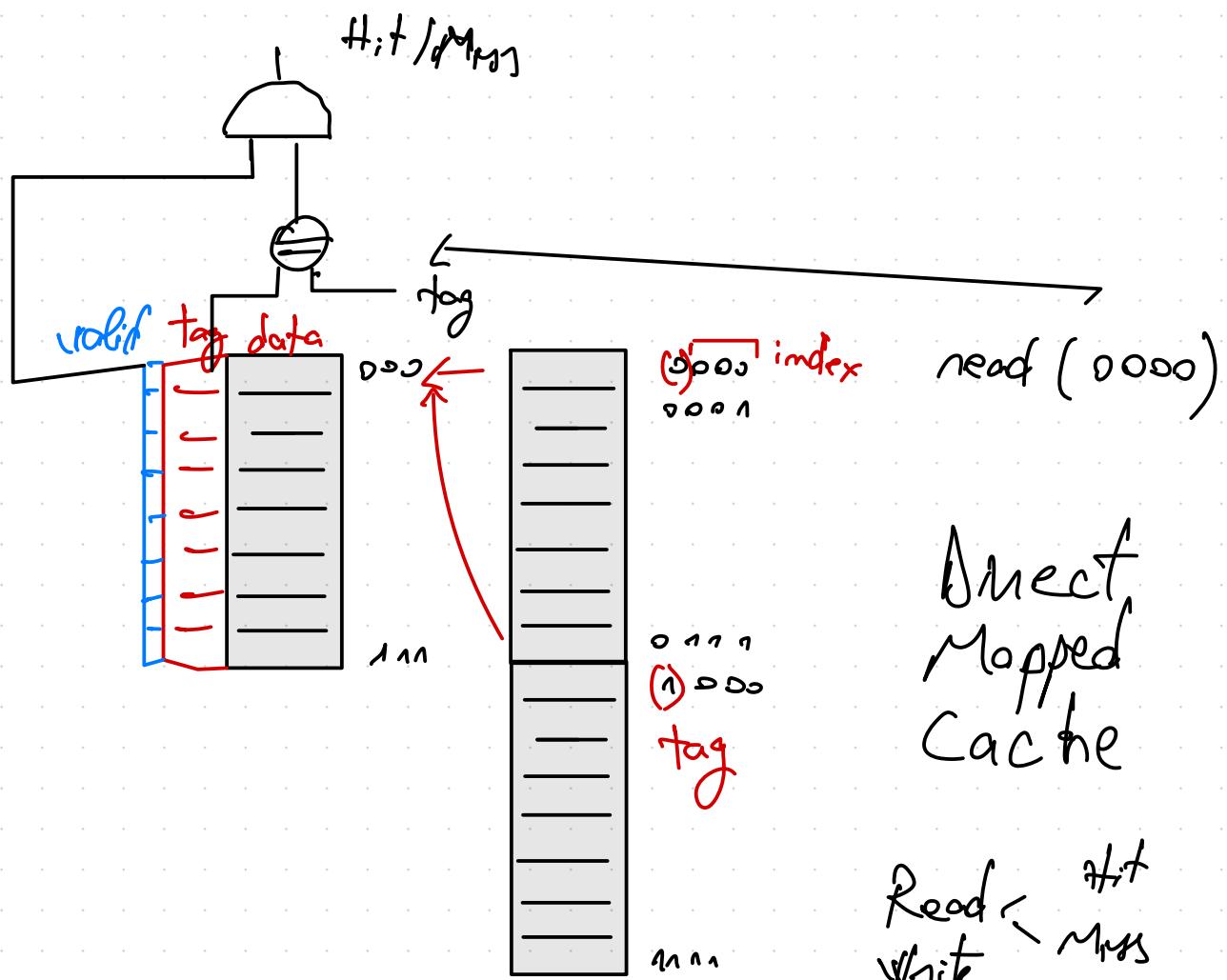
Locality Principle  $\hookrightarrow$  Temporal Locality  $\rightarrow$  appropriate cache  
 Locality Principle  $\hookrightarrow$  Spatial Locality  $\rightarrow$  temporal access  $\rightarrow$  cache spatial access



Write Hit  $\rightarrow$  same as Read Hit

Write Policies  $\hookrightarrow$  Write Through  $\rightarrow$  writes both cache and MM  $\rightarrow$  write buffer  
 Write Back  $\rightarrow$  write only cache, set dirty bit, write MM on eviction

dirty bit  $\rightarrow$  different data from cache and MM

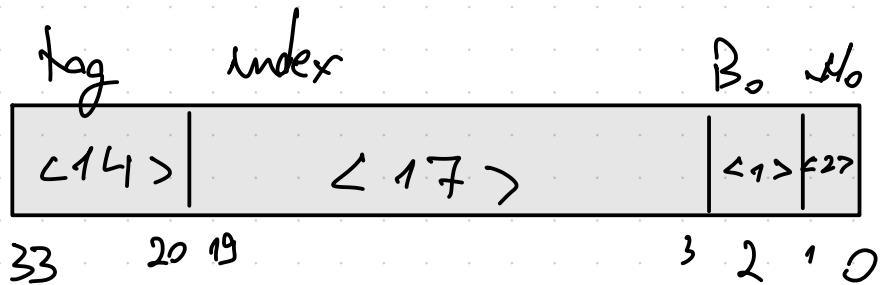


$\text{write } (\overbrace{1001}, 7) \rightarrow \text{miss}$

$\rightarrow$  write allocate  $\rightarrow$  writes MM, alloc. cache  
 $\rightarrow$  no allocate  $\rightarrow$  writes MM, don't alloc cache

↳ write back + allocate  
 ↳ through + no allocate

$$16 \text{ GiB} = 2^4 \times 2^{30} = 2^{34} \text{ B}$$



$$128 \text{ kB} = 2^7 \cdot 2^{10} \text{ B} = 2^{17} \text{ B}$$

$$1 \text{ word} = 4 \text{ B} = 2^2$$

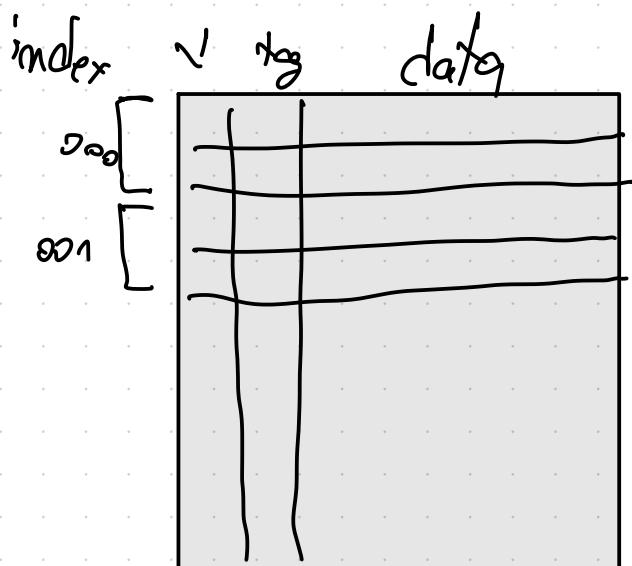
$$1 \text{ block} = 2 \text{ words} = 2^3$$

↓ tag                                  data

$\langle 17 \rangle$



## 1.2 Set associative caches



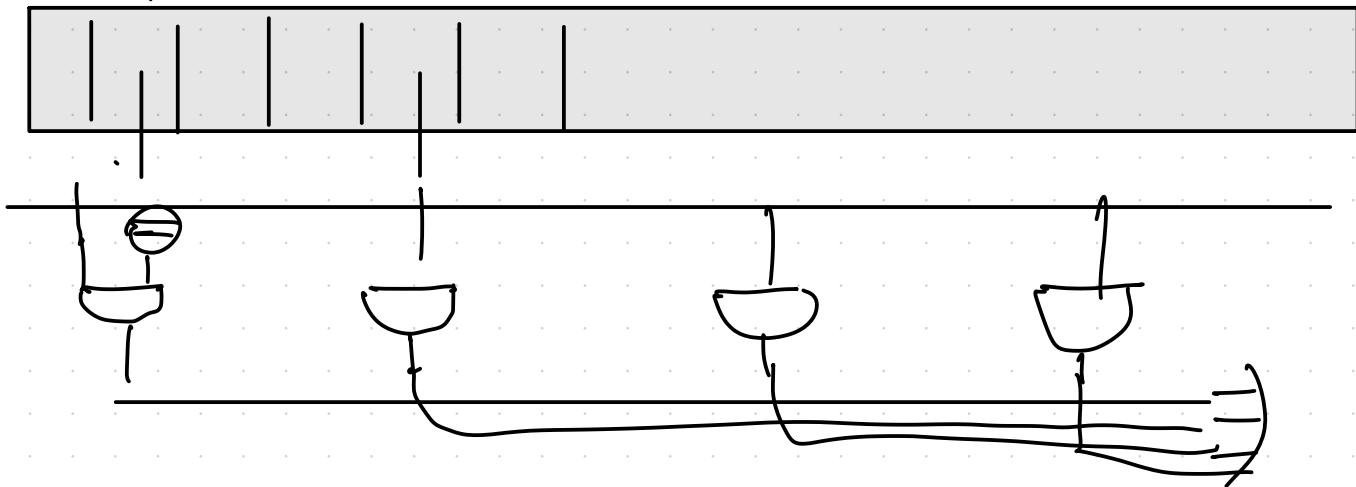
0000 trade off, trebuie  
 0001 verificat la hit ambele  
 taguri, mai trebuie  
 considerat ce pt. fiecare  
 set

6 miss:

## Replacement Policies:

- Random
- FIF
- LRU (Least Recently Used) age bits

2-way, 4, 8, 16. - ~  
fully associative



5.3 cap 5 P & H

LM Cache 64 bit MM address 2<sup>5</sup> blocks in each  
offset 4 bits word addressing

- a) cache block size (words)
- b) Ratio between total bits

5.3, 5.5 HW 502 Risc √ FA

### 5.3 32 kB cache

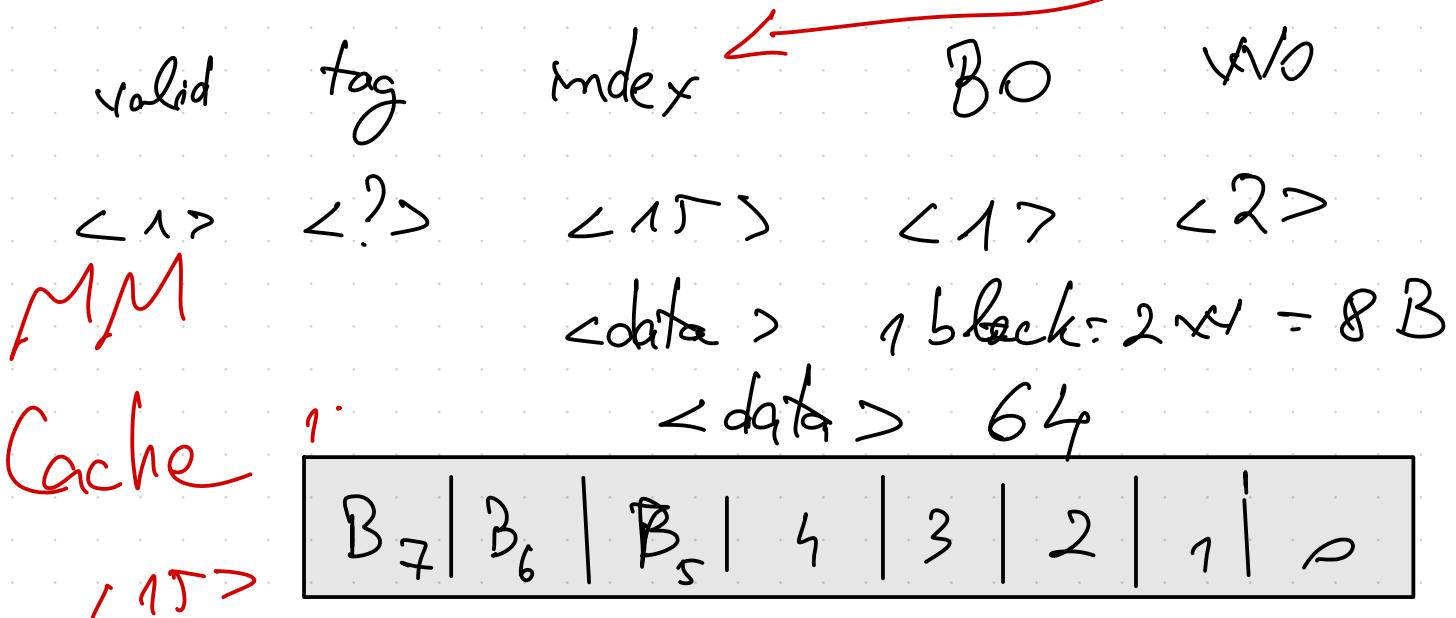
SRAM for storage for tags, valid bits

depending on config, total SRAM usage  
DMA, byte address      32 bit word  
                                4 B  $2^2$

**5.3.1** [10] <§5.3> Calculate the total number of bits required to implement a 32 KiB cache with two-word blocks.

$$\text{block} = 2 \text{ words} = 2 \cdot 2^2 B = 2^3 B$$

$$32 \text{ kB cache} = 2^5 \cdot 2^{10} B = 2^{15}$$



1 b ✓      ? Tag      ? Data  
32 kB

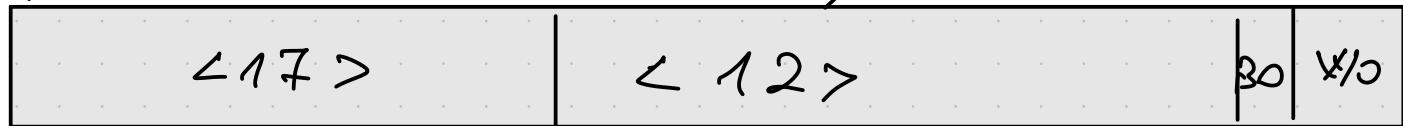
x1 <sub>1</sub>		x1 <sub>0</sub>	64
-----------------	--	-----------------	----

$$32 \text{ kB} \rightarrow 2^5 \cdot 2^{10} / (2 \text{ words/block} \cdot 2^2 \text{ B/word}) = 2^R \text{ blocks}$$

4096

### 3.1 Address Field 15/14 Index

3 2 1 0



$$\frac{32 \text{ kB}}{2 \cdot 2^2 \text{ B/block}} = 2^{12} \text{ Blocks} \Rightarrow \text{index} = 12$$

$$\text{Data Store Size} = 32 \text{ kB} \cdot 8 \frac{\text{B}}{2^5 \cdot 2^{10}} = 2^{15} \cdot 2^3 = 2^{18} \text{ B}$$

$$\begin{aligned} \text{Tag + Valid} &= 4096 \text{ lines} \times 17 \\ &\quad + 4096 \times 1 = \underline{4096 \times 18} \\ &= 2^{12} \times 18 \end{aligned}$$



$$\begin{aligned} \text{Total Bits} &= 2^{18} + 2^{12} \times 18 \\ &= 2^{12}(2^6 + 16) = 82 \cdot 2^{12} \\ &= \underline{3358726} \\ &\quad \text{SRAM} \end{aligned}$$

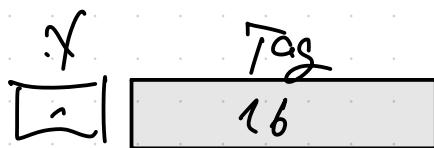
**5.3.2** [10] <§5.3> Calculate the total number of bits required to implement a 64 KiB cache with 16-word blocks. How much bigger is this cache than the 32 KiB cache described in Exercise 5.3.1? (Notice that, by changing the block size, we doubled the amount of data without doubling the total size of the cache.)

$$64 \text{ KiB} \rightarrow 2^6 \cdot 2^{10} \text{ B} = 2^{16} \text{ B}$$

$$1 \text{ block} = 16 \times 1 \text{ word} = 2^4 \times 1 \text{ word} = 2^4 \cdot 2^2 \text{ B}$$

$$64 \text{ KiB} = 2^{10} \text{ blocks} \Leftrightarrow \text{index} = 10$$

31	Tag	16, 15	Index	6 5 4 3 2 1 0
	<16>	1	<10>	B0   wo



$$16 \cdot 4 \cdot 8 = 512 \text{ b}$$

1024 lines

$$1 \cdot 1024 + 16 \cdot 1024 + 512 \cdot 1024$$

$$= 1024 (1 + 16 + 512) = 541696$$

$$\frac{541696}{335872} = 1,612 \text{ times the bytes needed for } 2 \times \text{storage}$$

**5.3.3** [5] <§5.3> Explain why this 64 KiB cache, despite its larger data size, might provide slower performance than the first cache.

- read / write to larger blocks
  - longer multiplexing operation

**5.3.4** [10] <§§5.3, 5.4> Generate a series of read requests that have a lower miss rate on a 32 KiB two-way set associative cache than on the cache described in Exercise 5.3.1.

- lower miss rate for two-way

△ M

## 2-way SA

tag <sub>0</sub>	0	M	0	M
tag <sub>1</sub>	0	M	0	M
tag <sub>0</sub>	0	M	1	H
tag <sub>1</sub>	0	M	1	H
0	0		1	
1	0		1	
0	0		1	
1	0		1	

Tag 0 / Tag 1

tag o

Tag 1

alternating  $\log 0$ ;  $\log 1$   $\Delta M$  always  
SA miss 2 Mass, rest  $\chi_{eff}$

**5.5** For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache.

64

Tag	Index	Offset
63-10	9-5	4-0

**5.5.1** [5] <§5.3> What is the cache block size (in words)?

32 bits address

1 block = 32 words

Tag	Index	Offset
<22>	<5>	<5>

index : 5 →  $2^5$  blocks

32 blocks

**5.5.2** [5] <§5.3> How many blocks does the cache have?

**5.5.3** [5] <§5.3> What is the ratio between total bits required for such a cache implementation over the data storage bits?

$$\begin{aligned} \text{data storage} &= 32 \text{ blocks} \times 32 \text{ words} / \text{block} \times 8 \\ &= 8192 = 2^{13} \text{ bytes} \end{aligned}$$

$$\text{total bits} = \text{data storage} + 32 (\text{valid}) + 32 (\text{tag})$$

$$32 \cdot 23 = 736$$

$$\text{ratio} = \frac{\text{data storage} + 736}{\text{data storage}}$$

$$= 1 + \dots = \cancel{1,089}$$

Beginning from power on, the following byte-addressed cache references are recorded.

Address												
Hex	00	04	10	84	E8	A0	400	1E	8C	C1C	B4	884
Dec	0	4	16	132	232	160	1024	30	140	3100	180	2180

**5.5.4** [20] <§5.3> For each reference, list (1) its tag, index, and offset, (2) whether it is a hit or a miss, and (3) which bytes were replaced (if any).

32 block      32 words / block  
(32 limit)      index = 5      tag = 22

✓	Tag	data
<1>	<22>	<256>

M M  
0 → index  
0 - 31 → 4

1 32 - 63

2 64 -

3 96

4 128

- - -

0 → tag = 0, index = 0, offset = 0

	tag	index	offset	H/m	bytes req.
0	0	0	0	M	0-31
04	0	0	4	H	
16	0	0	16	H	
132	0	4	4	M	128-139
232	0	7	8	M	224-255
160	0	5	0	M	160-191
1024	1	0	0	M	0-31
30	0	0	30	M	0-31
140	0	5	12	H	
3100	3	0	28	M	2072-3103
180	0	5	20	H	
2180	2	4	4	M	2176-2207

	Tag	Index	Offset
0	0 - 31		
1	32 - 63	<22>	<5>
2	64 - 95	0000	00100
3	96 - 127	-	4
4	128 - 159		
T	↑		
132			
232			
C			
F			

160  $\Rightarrow$  4 H / 12 total  
 $= 33\%$  HitRate