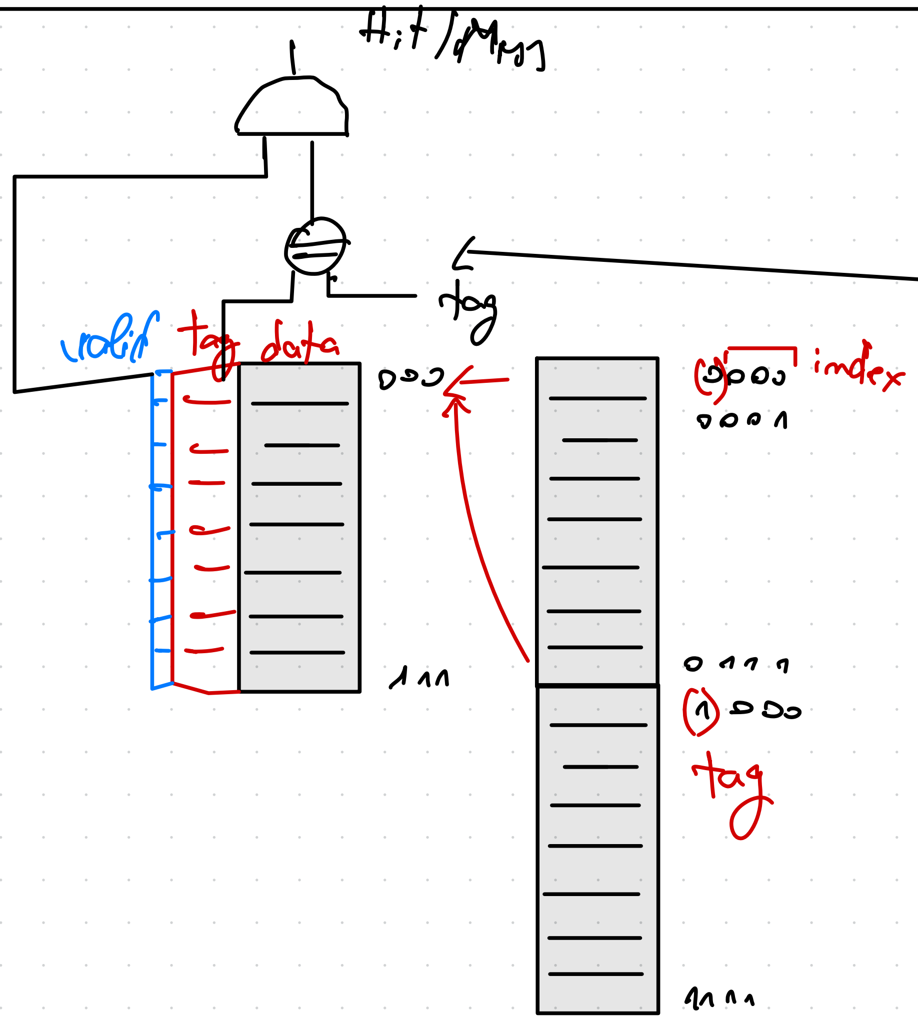
Cache Structure

Locality Principle
- → Temporal Locality → proprietate ca timp
- → Spatial Locality → ca spatiu de acces



read (0000)

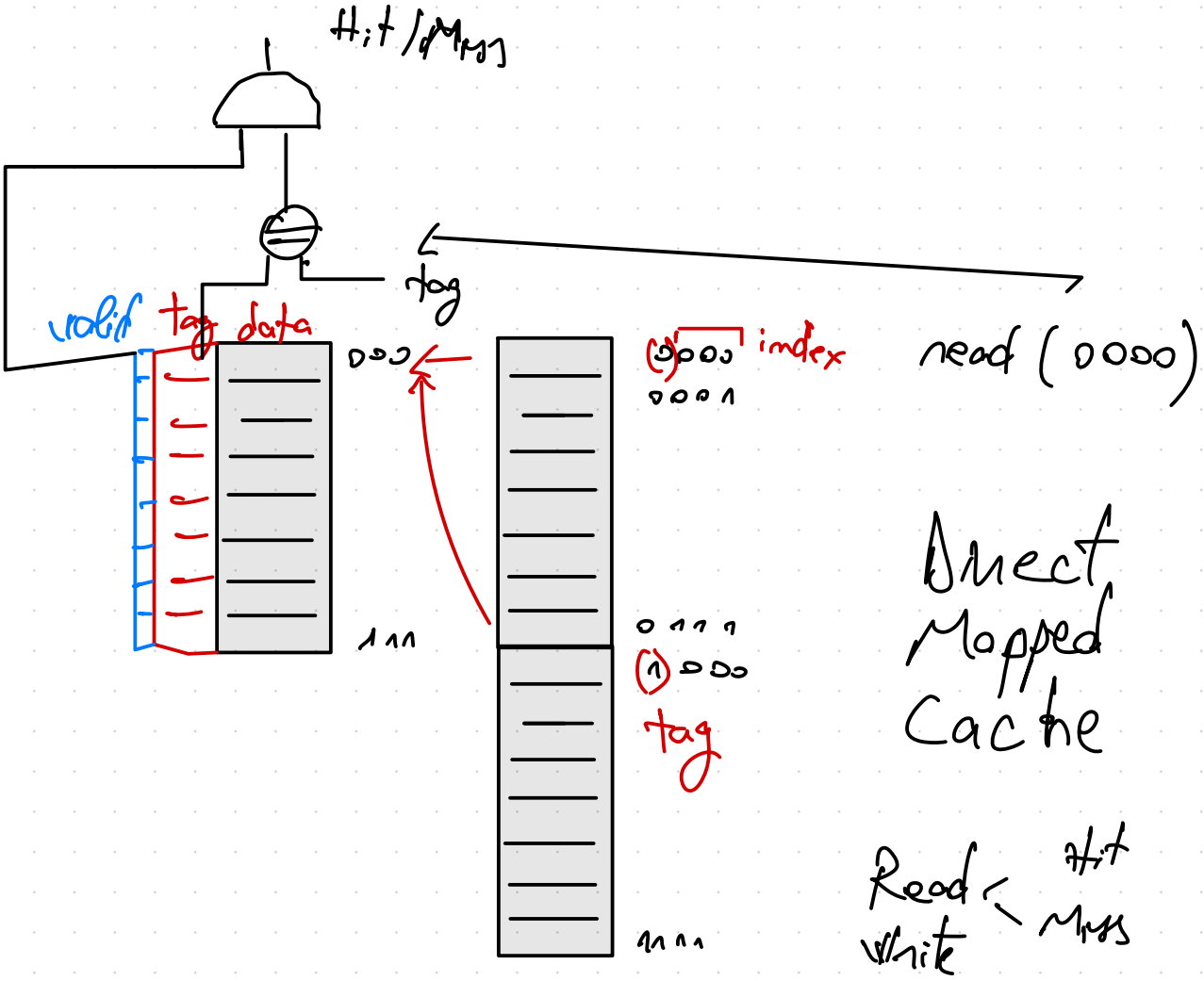
Direct Mapped Cache

Read / Write < Hit / Miss

Write Hit → same as Read Hit

Write Policies
- write through → writes both cache and MM → write buffer
- write back → write only cache, set dirty bit, write MM on eviction

dirty bit → different data from cache and MM

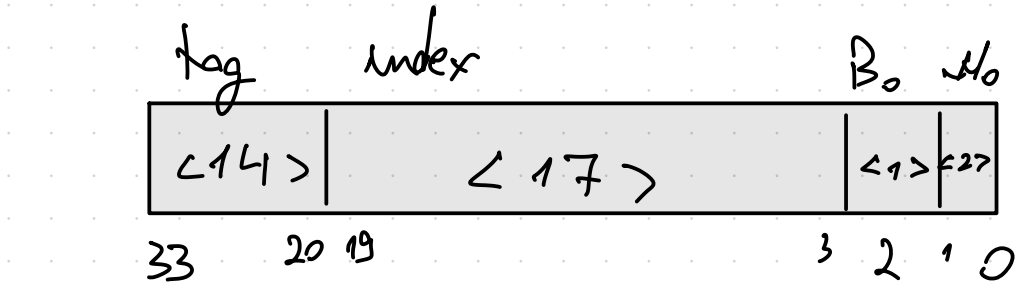


write (1001, 7) -> miss

- write allocate -> writes MM, alloc. cache
- no allocate -> writes MM, don't alloc cache

write back + allocate
write through + no allocate

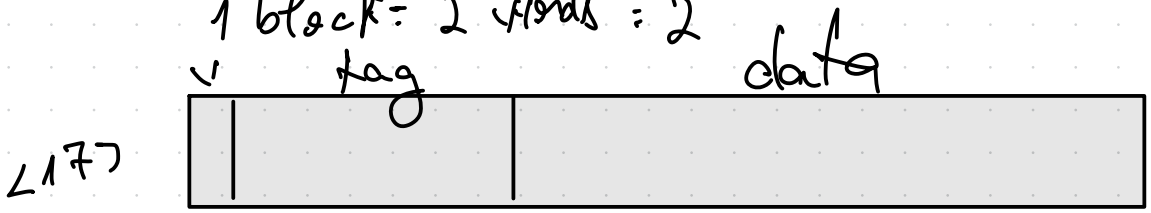
16 GiB $= 2^4 \times 2^{30} = 2^{34}$ B



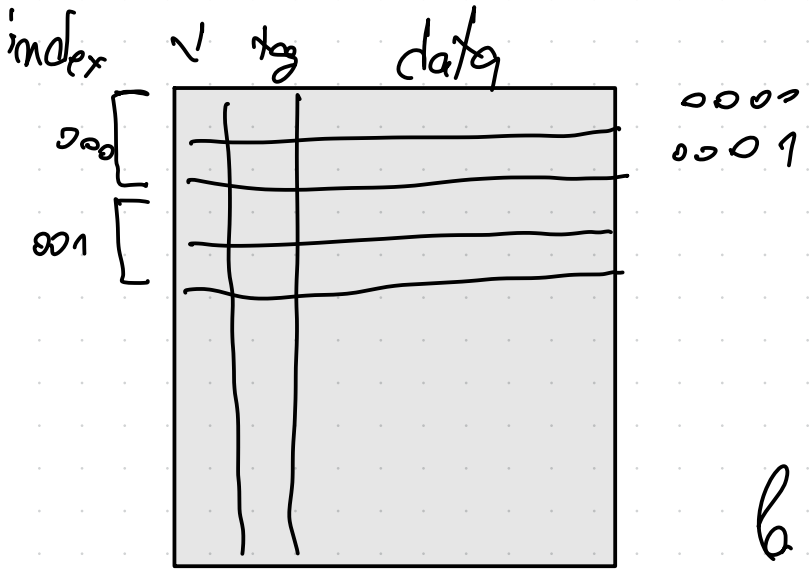
128 KiB $= 2^7 \cdot 2^{10}$ B $= 2^{17}$ B

1 word = 4 B $= 2^2$

1 block = 2 words $= 2^1$



1.2 Set associative caches



trade-off, trebuie verificat la hit ambele tag-uri, mai trebuie comparatoare pt. fiecare set

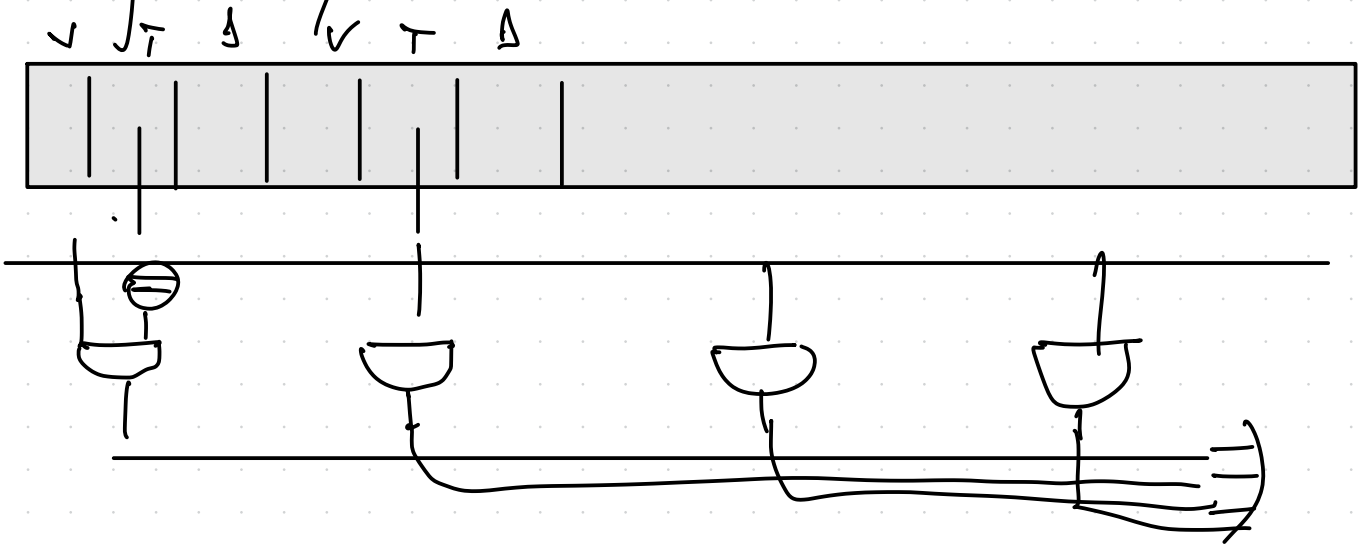
la miss:

Replacement Policies:

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

2-way, 4, 8, 16. - -

fully associative



5.3 cap 5 P&H

DM Cache 64 bit MM address 2^5 blocks in each
offset 4 bits word addressing

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

5.3, 5.5 #HW 502 RiscV FA

5.3 32 kiB cache

SRAM for storage for tags, valid bits

depending on config, total SRAM usage

SMA, 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.

block = 2 words = $2 \cdot 2^2$ B = 2^3 B → BO

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

valid tag index BO WO

<1> <?> <15> <1> <2>

<data> 1 block: 2W = 8 B

<data> 64

MM

Cache

<15>

B_7	B_6	B_5	4	3	2	1	0

1 b V ? Tag ? Data

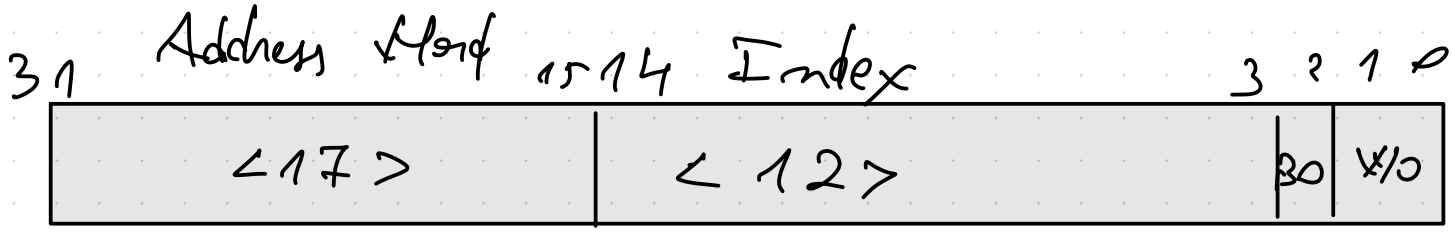
32 kiB

W_1	W_0

<64>

32 kiB → $2^5 \cdot 2^{10}$ / (2 words/block · 2^2 B/word) = 2^K Blocks

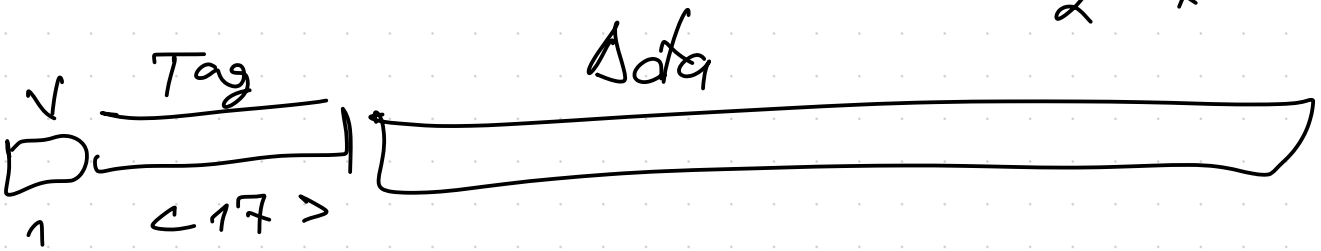
4096



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

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

$$Tag + Valid = 4096\ lines \times 17 + 4096 \times 1 = 4096 \times 18$$
$$= 2^{12} \times 18$$



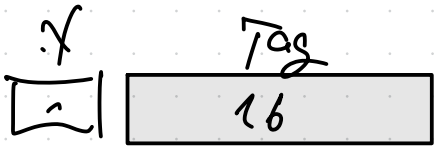
$$Total\ Bits = 2^{18} + 2^{12} \times 18$$
$$= 2^{12}(2^6 + 18) = 82 \cdot 2^{12}$$
$$= 335872\ SRAM$$

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

1 blk = 16 word = 2^4 word = $2^4 \cdot 2^2$ B

64 KiB = 2^{10} blocks => index = 10



16 · 4 · 8 = 512 b

1024 times

1 · 1024 + 16 · 1024 + 512 · 1024

= 1024 (1 + 16 + 512) = 541696

$\frac{541696}{335872} = 1,612$ times the bits needed for 2x 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.

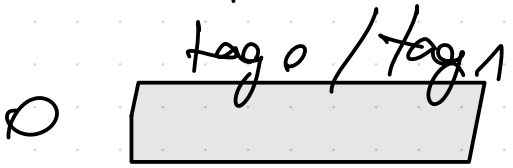
-> lower miss rate for two-way

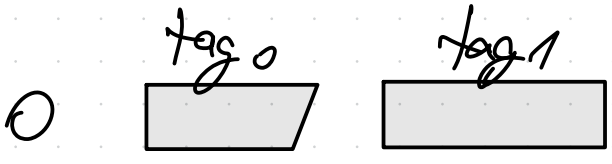
DM

tag		
tag 0	0	M
tag 1	0	M
tag 0	0	M
tag 1	0	M
0	0	
1	0	
0	0	
1	0	

2-way SA

0	M
0	M
1	H
1	H
1	
1	
1	
1	





alternating tag 0 ; tag 1

DM always miss

SA 2 Miss, rest Hit

5.5 For a direct-mapped cache design with a ~~32~~ 64-bit address, the following bits of the address are used to access the cache.

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?

data storage = 32 blocks x 32 words/block x 8

$= 8192 = 2^{13}$ b

total bits = data storage + 32 (valid) + 32 · 22 (tag)

$32 \cdot 23 = 736$

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

$= 1 + \ldots = 1,089$

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

V	Tag	data
<1>	<22>	<256>

M M

0 → index

0 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 rgl.
0	0	0	0	M	0-31
04	0	0	4	H	
16	0	0	16	H	
132	0	4	4	M	128-159
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	3072-3103
180	0	5	20	H	
2180	2	4	4	M	2176-2207

0 0-31
1 32-63
2 64-95
3 96-127
4 128-159
5

132

232

7 8

Tag	Index	Offset
<22>	<5>	<5>
0000	00100	00100
	4	4

160 =>

4 H / 12 total

= 33% HitRate