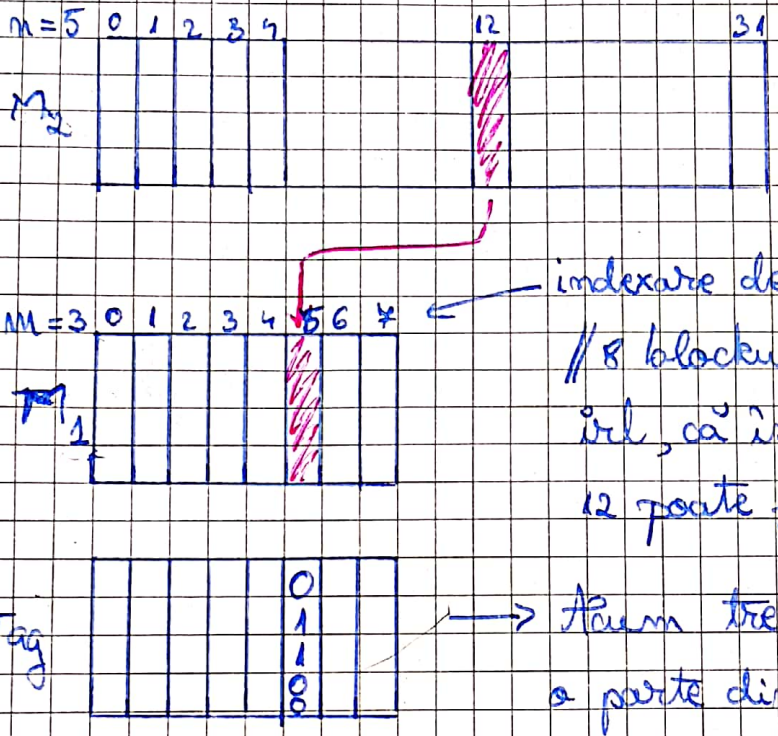
CURS 11 14.05.2019

3.4. Full associative mapping

k - associativity : $k = 2^s$

$s = 0 \Rightarrow$ direct mapping

$s = m \Rightarrow$ full associativity unde 2^m = nr. of blocks in cache



indexare de forma

// 8 blocuri ce nu mai sunt indexate ird, că is toate la același index.

12 poate fi mapat oriunde: ex: pe 5.

⟶ Acum trebuie scris tot nr, nu doar o parte din el.

Exemplu:

2^{24} words

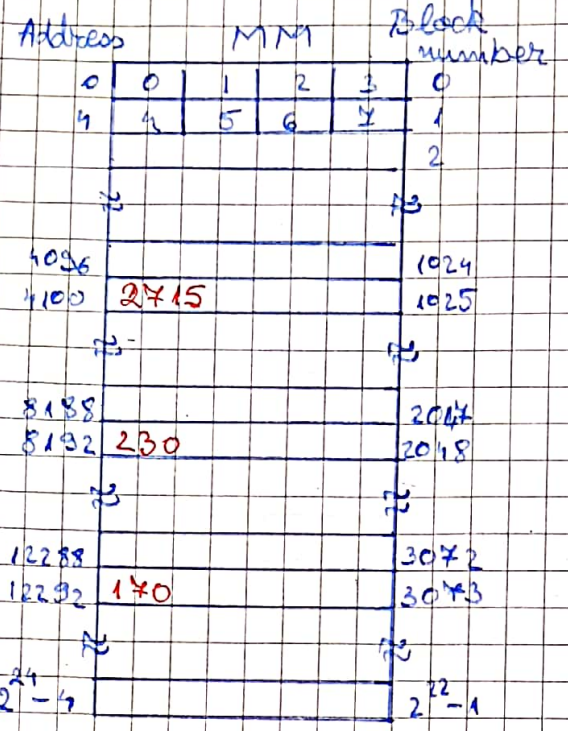
MM size = 16 MiB = 2^{24} B = $\frac{2^{24}\text{B}}{2^2\text{ B/block}}$ = 2^{22} blocks (*)

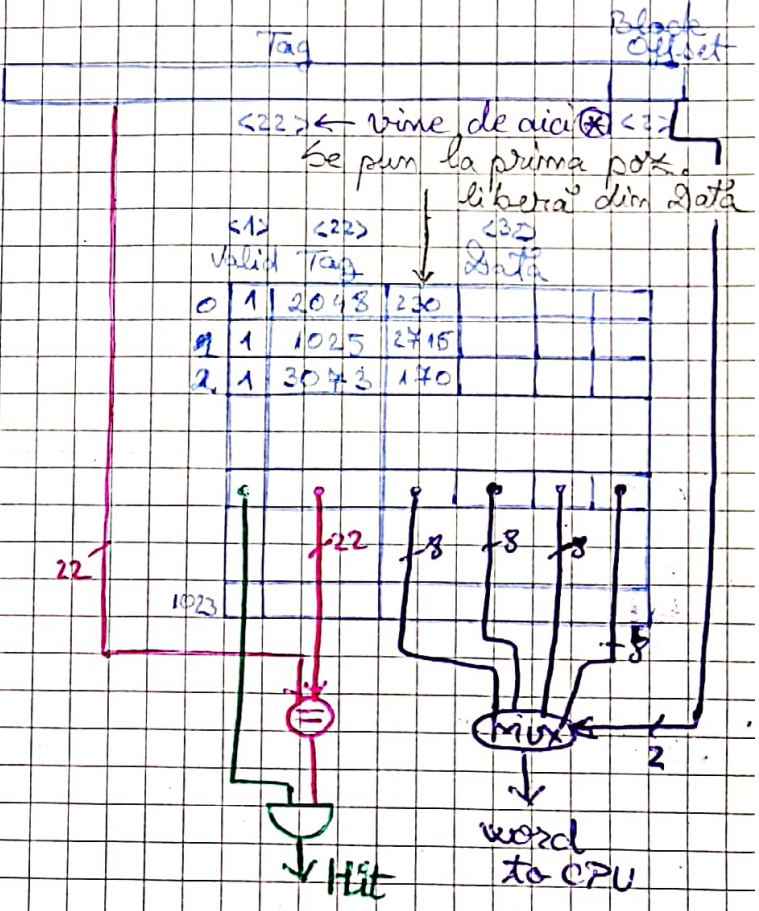
1 word = 1B

1 block = 4 words

Cache data size = 1 k blocks

Address	Code
8192	230
4100	2715
12292	170





3.5. Cache performance metrics

Miss: atunci când CPU nu găsește informația căutată în Cache

Miss Rate = 1 − Hit Rate

Hit: atunci când CPU găsește ce caută în Cache

Miss Causes
- Compulsory (Cold Start misses): missuri apărute când resetezi sistemul și cache-ul e gol
  - → soluție: bigger blocks
- Conflict: 2 blockuri se mapează la același index (ce se întâmplă la set-associative)
  - → soluție: crești set associativity
- Capacity: memorie Cache mică
  - → soluție: bigger cache

Average Memory Access Time (AMAT) =

$= T_{ac} \cdot \text{Hit rate} + (T_{ac} + \text{Miss Penalty}) \cdot \text{Miss rate}$

$= T_{ac} + \text{Miss Penalty} \times \text{Miss rate}$

T_{ac} = timp de acces la cache

Hit rate + Miss rate = 1

Miss Penalty - timpul pe care îl ia ~~o~~ informații să fie încărcată în cache.

Bigger block <-> Miss Rate scade, dar Miss Penalty crește

CPU_{time} (real ori ideal) = Clock cycles for program × CCT =

= (Execution clock cycles + Memory stall clock cycles) × CCT

= IC × (Exec. clock cycles per instruction + Memory stall clock cycles per instruction) × CCT

(Exec. clock cycles per instruction = CPI)

Memory stall clock cycles per instruction =

= Memory accesses per instruction × Miss Rate × Miss Penalty

(Memory accesses per instruction = Misses per instruction)

Reader:

$CPU_{time} = IC \times (CPI_{exec.} + \text{Memory accesses per instr.} \times \text{Miss Rate} \times \text{Miss Penalty}) \times CCT$

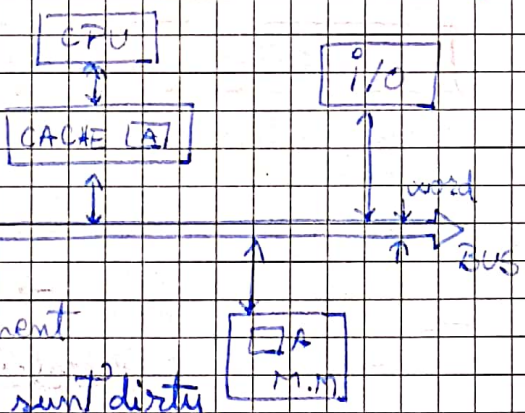
3.6. Write policies : 10% writes

Cache / Memory coherence → Update

a) Write through : implicit update

b) Write back : update on replacement

- dirty bit: dacă blocurile sunt dirty, bitul e 1 și doar acele blocuri sunt înlocuite



What happens in case of miss?

=> Allocate

α) Write Allocate

β) Write No Allocate

Pairings { Write through + Write No-Allocate
Write back + Write Allocate

3.4. Replacement policies (au sens doar la set și full asociativitate)

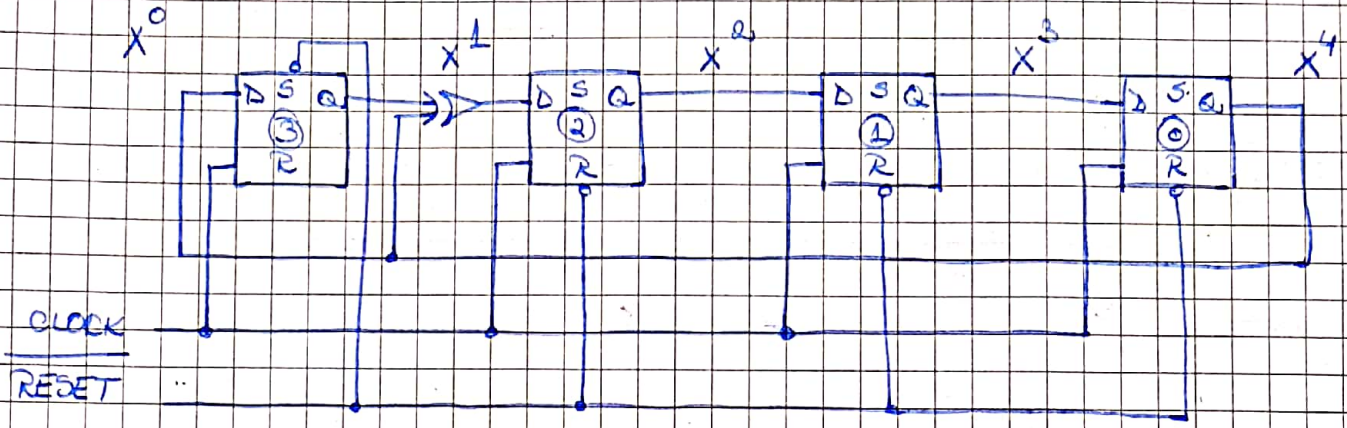
a) random : când SA e mică

b) FIFO

c) Least recently used – LRU

Modul de generare random a numerelor prin polinom indivizibil

$G(X) = X^4 + X + 1$



D_3	D_2	D_1	D_0	EZ
1	0	0	0	1
0	1	0	0	2
0	0	1	0	4
0	0	0	1	8
1	1	0	0	3
0	1	1	0	6
0	0	1	1	12
1	1	0	1	11
1	0	1	0	5
0	1	0	1	10
1	1	1	0	7
0	1	1	1	14
1	1	1	1	15
1	0	1	1	13
1	0	0	1	9
1	0	0	0	1

Perioada de repetiție : $2^4 - 1 = 15$ clock-uri