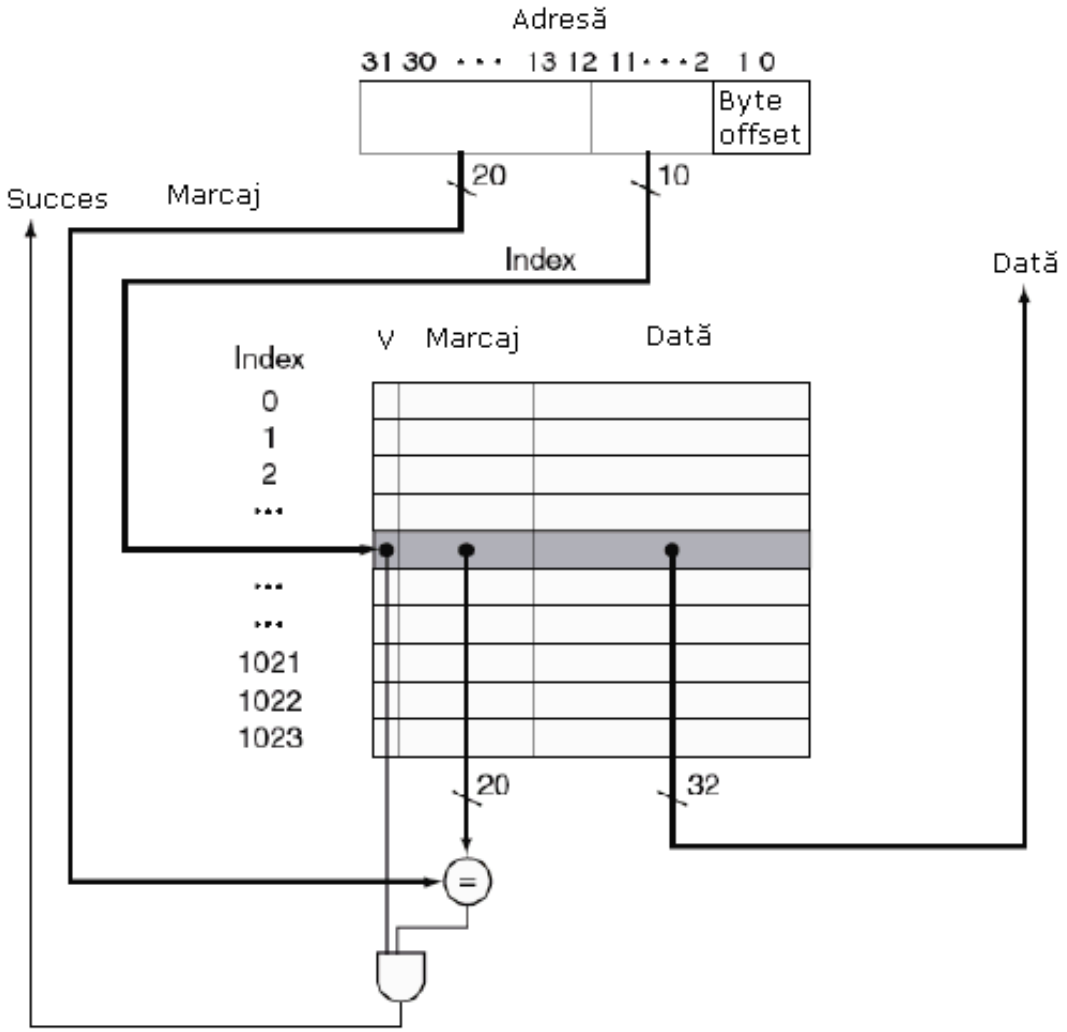
Tag = Marcaj



1. indexul memoriei cache, folosit la selectarea blocului de memorie cache

2. câmpul marcajului, folosit la compararea cu valoarea din câmpul marcaj al memoriei cache

Numărul total de biți dintr-o memorie cache cu corespondență directă este:

2^n * (dimensiunea blocului de memorie + dimensiune marcaj + dimensiunea câmpului de validare).

TEMĂ

Câți biți sunt necesari pentru o memorie cache cu corespondență directă având 64KB de date și blocuri de 1 cuvânt, folosind adrese de 32 de biți ?

Tratarea eşecurilor

a). Soluția cea mai simplă presupune staționarea UCP-ul, înghețând conținutul tuturor registrelor.

O unitate de control separată tratează eșecul, aducând data din memoria principală în memoria cache

Execuția este reluată începând cu ciclul care a cauzat eșecul.

Tratarea eșecului se face de către unitatea de control a procesorului și de către o unitate de control separată ce inițiează accesul la memorie și aduce datele în memoria cache.

b). În cazul implementării pipeline tratarea eșecurilor la memoria cache este mai dificilă deoarece execuția unor instrucțiuni trebuie continuată, în timp ce altele staționează.

1). se trimite valoarea originală a PC-ului (PC-4) la memorie;

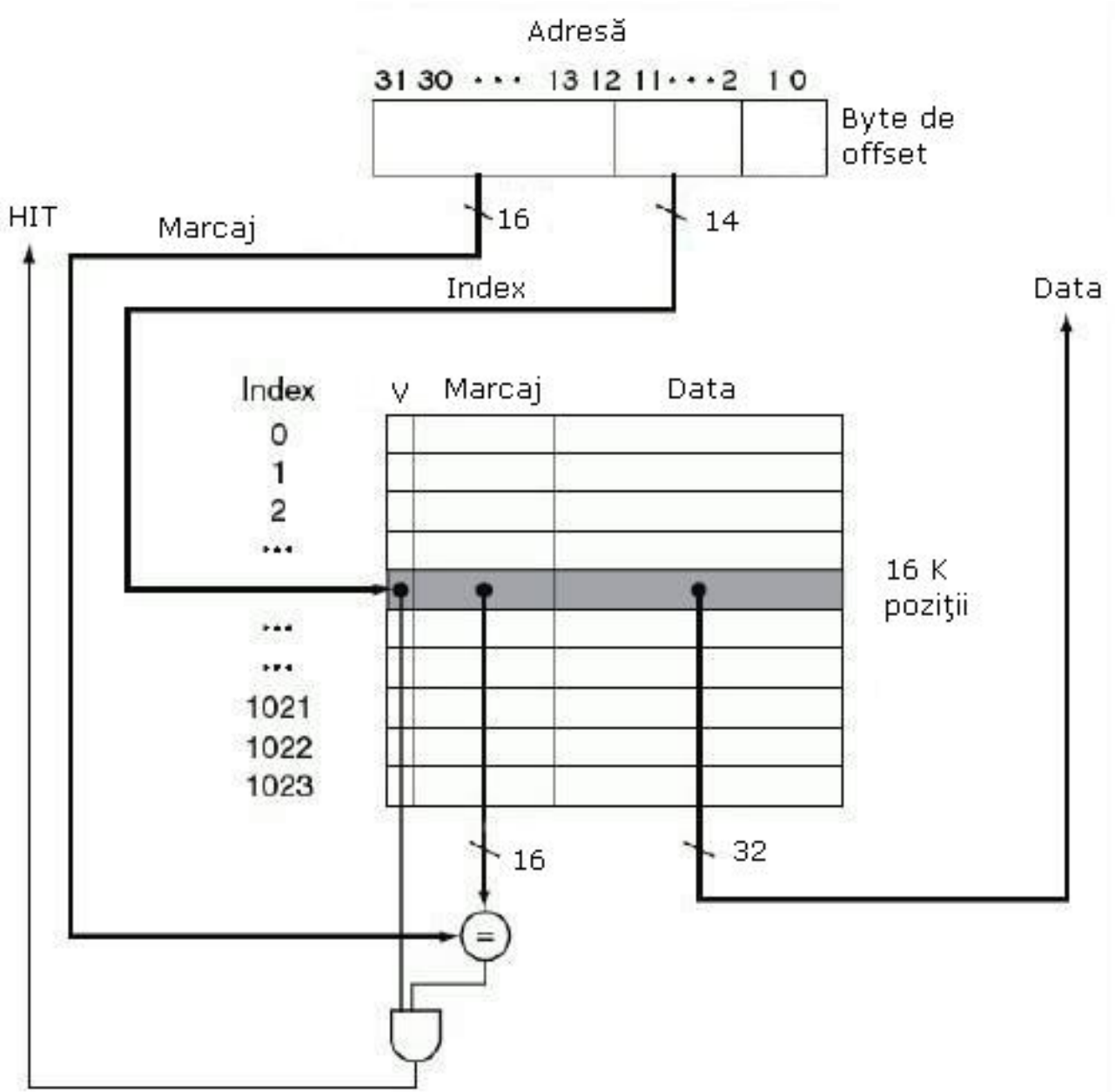
2). se instruiește memoria principală să execute o citire și așteaptă până când acesta termină accesul;

3). se scrie locația memoriei cache, punând datele din memorie în porțiunea de date a acestei locații, scriind biții cei mai semnificativi ai adresei (din UAL) în câmpul marcajului și setând bitul de validare;

4). se repornește execuția instrucțiunii la primul pas, care va reextrage instrucțiunea - de data aceasta se regăsește în memoria cache.

O metodă de reducere a efectului eșecurilor la memoria cache este folosirea tehnicii **staţionare la utilizare**.

EXEMPLU


Adresă
31 30 ... 13 12 11 ... 2 1 0
Byte de offset
16
14
HIT
Marcaj
Index
Data
Index
V
Marcaj
Data
0
1
2
...
...
...
1021
1022
1023
16 K poziţii
16
32
=

CITIREA

1 – se trimite adresa la memoria cache corespunzătoare. Adresa vine fie de la PC (pt instrucțiuni) fie de la UAL (pt date).

2– dacaă HIT cuvântul este disponibil pe liniile de date. Dacă MISS, se trimite adresa la memoria principală. Când memoria transmite datele de la adresa respectivă, acestea sunt scrise în memoria cache.

SCRIEREA

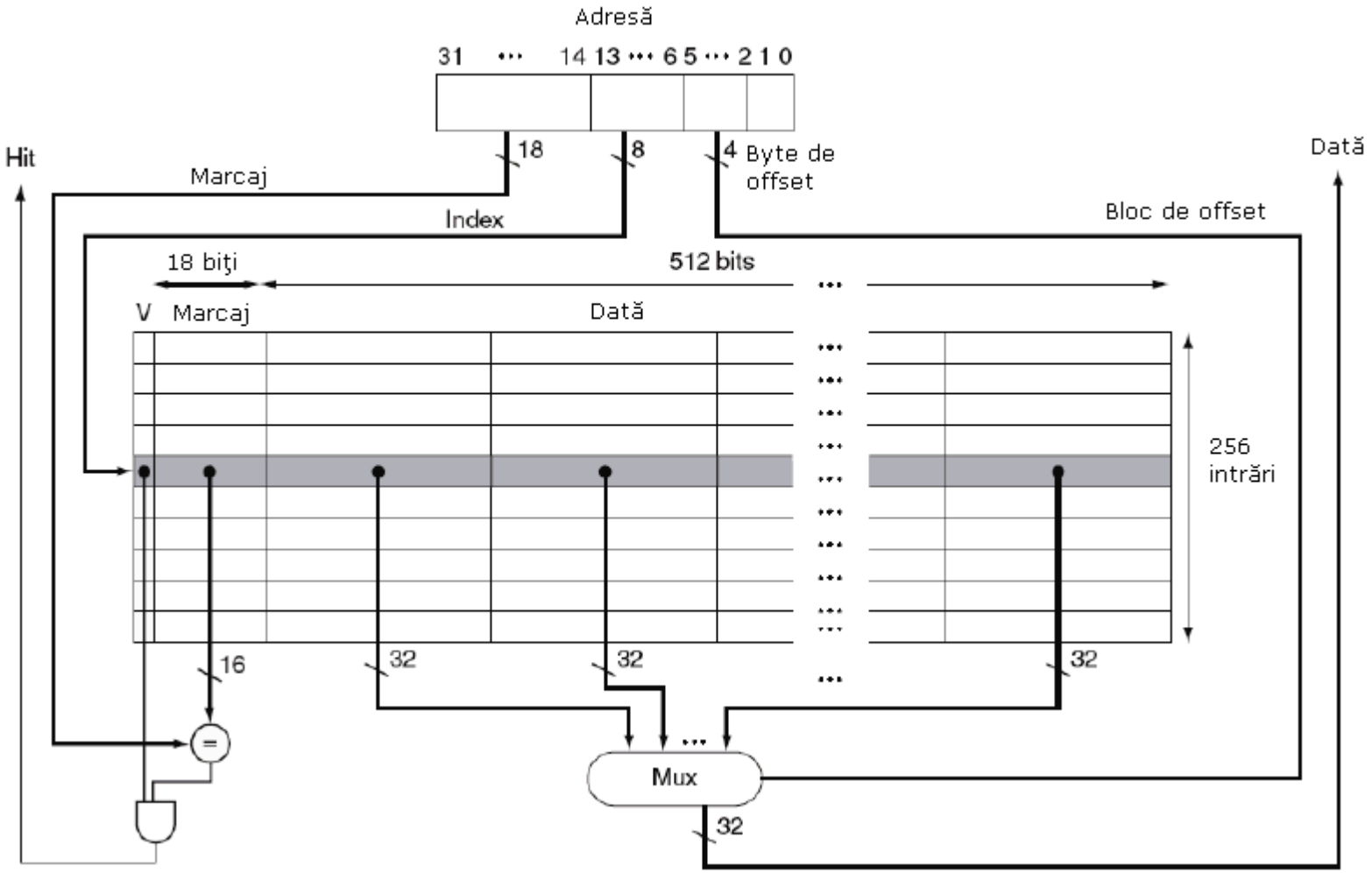
Dacă se scrie doar în memoria cache și nu și în memoria principală => memoria principală și memoria cache sunt inconsistente. Cea mai simplă metodă de evitare este scrierea în ambele memorii => **scriere simultană – write through**.

1. memoria cache este indexată folosind biții 15–2 ai adresei
2. se scriu biții 31–16 ai adresei în marcaj, se scrie cuvântul în zona de date și se setează bitul de validitate
3. se scrie cuvântul în memorie folosind întreaga adresă

SOLUȚIE – folosirea unor memorii tampon – write buffer

O soluție la metoda **write through** este schema **write back** – scrie la loc

FOLOSIREA LOCALIZĂRII SPAŢIALE



Se doreşte ca blocul memoriei cache să fie mai mare decât lungimea unui cuvânt

Determinarea blocului din memoria cache pentru o anumită adresă

(Adresa blocului) modulo (Numărul de blocuri din memoria cache)

unde

Adresa blocului = adresa cuvântului / numărul de cuvinte din bloc

EXEMPLU

Se consideră o memorie cache cu 64 de blocuri de date, fiecare cu dimensiunea de 16 octeți. Care este numărul blocului corespunzător adresei de octet 1200 ?

Eşecurile şi succesele de scriere

Un bloc de date conține mai mult de un cuvânt => nu se poate să scriem doar marcajele și datele.

Considerații:
1. două adrese de memorie X și Y au același bloc corespondent C în memloria cache
2. Blocul are 4 cuvinte și conține adresa Y
3. Scriem la adresa X prin simpla suprapunere a datelor și a marcajului din blocul C

Conform considerațiilor de mai sus, ce se întamplă după operația de scriere ?

Spre deosebire de cazul blocurilor de 1 cuvânt, în cazul blocurilor de date cu mai multe cuvinte, eșecurile la scriere necesită o citire din memorie.

Îmbunătățirea performanței în cazul folosirii principiului de localizare temporară

Presupunem că următorii octeți de adrese sunt ceruți de către un program

16, 24, 20

și nici una din aceste adrese nu se găsește în memoria cache.

Dacă folosim o memorie cache cu blocuri de date de 4 cuvinte, atunci un eșec

la adresa 16 va determina încărcarea în memoria cache a blocului care conține

Adresele 16, 20, 24 și 28.

Din exemplul prezentat se observă că vom avea un singur eșec. Dacă blocul de date ar fi de 1 cuvânt, câte eșecuri am fi avut ?

Măsurarea și îmbunătățirea performanțelor

Timpul de execuție pentru UCP este format din ciclurile de ceas în care UCP execută programul cerut și cele în care UCP-ul așteaptă executarea transferurilor în și din memorie

Timpul UCP = (ciclurile de ceas UCP pt execuție + ciclurile de ceas staționare a UCP

datorate memoriei) x durata unui ciclu de ceas

Ciclurile de ceas staționare datorate memoriei = cicluri staționare pentru citire + cicluri staționare pentru scriere

Ciclurile de ceas de staționare la citire = citiri/program x rata de eșec la citire x penalizarea de eșec la citire

Ciclurile de staționare datorate memoriei = nr de accese la memorie/program x rata de eșec x penalizarea de eșec = nr de accese la memorie/program x nr de eșecuri/instrucțiune x penalizarea de eșec

Până acum am folosit doar schema de amplasare a blocurilor din memoria cache denumită corespondenţă directă.

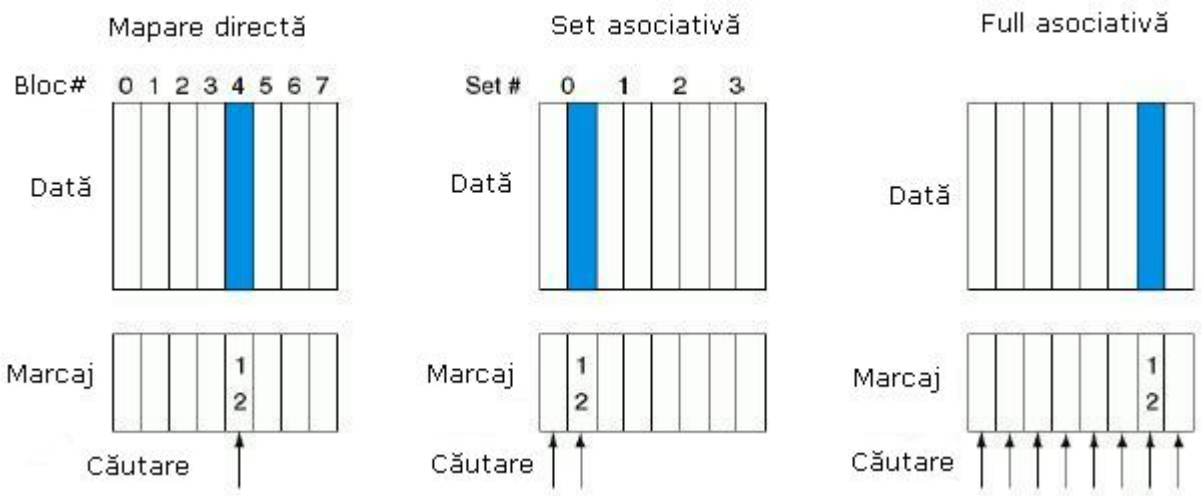
Schema în care un bloc de date poate fi amplasat în orice locaţie din memoria cache se numeşte schemă cu asociativitate totală.

Regăsirea blocului de date presupune examinarea tuturor locaţiilor de memorie cache => paralelizarea căutării

O altă schemă care este între cele două se numeşte schema cu asociativitate parţială.

În aceste tipuri de scheme memoria cache are un număr fix de locaţii în care se poate amplasa fiecare bloc de date. Deci vom avea un număr de seturi fiecare format din n blocuri de date.

Pentru regăsirea blocului este necesar să parcurgem toate blocurile unui set.



Într-o memorie cache cu asociativitate parţială, setul conţinând un anumit bloc de memorie este dat de relaţia:

(numărul blocului) modulo (numărul de seturi din memoria cache)

OBSERVAŢIE : Creşterea gradului de asociativitate reduce rata de eşec, dar creşte timpul de HIT.

EXEMPLU

Avem 3 memorii cache fiecare având 4 blocuri de câte 1 cuvânt. Cele trei memorii cache sunt cu asociativitate totală, asociativitate cu 2 căi şi corespondenţă directă.

Să se găsească numărul de eşecuri pentru fiecare dintre cele 3 scheme de amplasare având următoarea secvenţă de adrese de bloc: 0,8, 0, 6, 8.

SOLUŢIE

Cazul 1 – memoria cache cu corespondenţă directă

Detectăm blocul din memoria cache corespunzător adreselor date:

Adresa blocului	Blocul memoriei cache
0	0 modulo 4 = 0
6	6 modulo 4 = 2
8	8 modulo 4 = 0

Conţinutul memoriei cache după fiecare referinţă

Adresa blocului de memorie accesat	HIT sau MISS	Blocul 0	Blocul 1	Blocul 2	Blocul 3
0	Miss	Mem(0)			
8	Miss	Mem(8)			
0	Miss	Mem(0)			
6	Miss	Mem(0)		Mem(6)	
8	Miss	Mem(8)		Mem(6)	

Cazul 2. Memoria cache cu asociativitate parţială cu 2 căi conţine 2 seturi (indicii fiind 0 şi 1), fiecare având 4 elemente. Vom determina setul corespunzător fiecărei adrese a blocurilor

Adresa blocului	Blocul memoriei cache
0	0 modulo 2 = 0
6	6 modulo 2 = 0
8	8 modulo 2 = 0

Pentru înlocuire vom folosi LRU

Adresa blocului de memorie accesat	HIT sau MISS	Set 0	Set 1	Set 2	Set 3
0	Miss	Mem(0)			
8	Miss	Mem(0)	Mem(8)		
0	HIT	Mem(0)	Mem(8)		
6	Miss	Mem(0)	Mem(6)		
8	Miss	Mem(8)	Mem(6)		

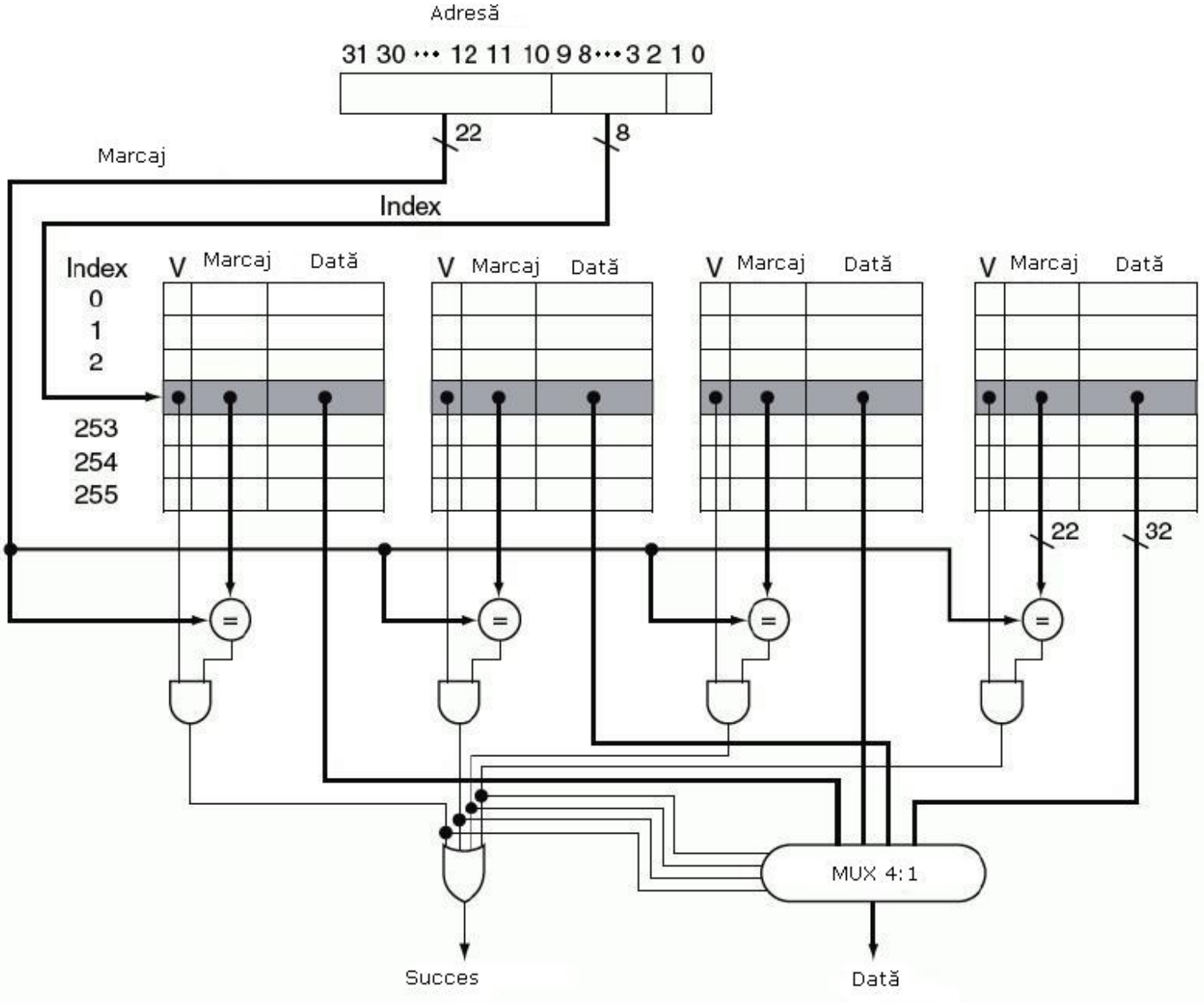
Avem doar 4 eşecuri, deci soluţia aceasta este mai bună decât precedenta

Memoria cache cu asociativitate totală

Adresa blocului de memorie accesat	HIT sau MISS	Set 0	Set 1	Set 2	Set 3
0	Miss	Mem(0)			
8	Miss	Mem(0)	Mem(8)		
0	HIT	Mem(0)	Mem(8)		
6	Miss	Mem(0)	Mem(8)	Mem(6)	
8	HIT	Mem(0)	Mem(8)	Mem(6)	

Aceasta este varianta optimă – avem doar 3 eşecuri

Localizarea unui bloc în memoria cache cu asociativitate parţială



Păstrăm dimensiunea memoriei cache constantă şi încercăm să mărim asociativitatea => numărul de blocuri/set va creşte => va creşte numărul de comparaţii efectuate în paralel.

Creşterea asociativităţii cu un factor de doi va dubla numărul blocurilor din set şi va înjumătăţii numărul de seturi => descreşterea dimensiunii indexului cu 1 bit şi o creştere a dimensiunii marcajului cu 1 bit.

Exemplu: Presupunem o memorie cache cu blocuri de 4Kb şi adrese de 32 de biţi. Să se găsească numărul total de seturi şi de biţi de marcaj pentru memoria cache cu corespondenţă directă, cu asociativitate parţială cu 2 şi 4 căi şi cu asociativitate totală.

a). Nr. de seturi = nr. de blocuri => log2(4Kb) = 12 biţi de index => (32-12)4K=80Kb numărul total al biţilor din marcaj

b). Cu 2 căi: 2K seturi şi numărul total al biţilor de marcaj este (32-11)*2*2K = 84Kb
Cu 4 căi: 1K seturi şi numărul total al biţilor de marcaj este (32-10)*4*1K = 88Kb

c). 1 set cu 4K blocuri iar marcajul are 32 de biţi => 32*4K*1 = 128 biţi pentru marcaj