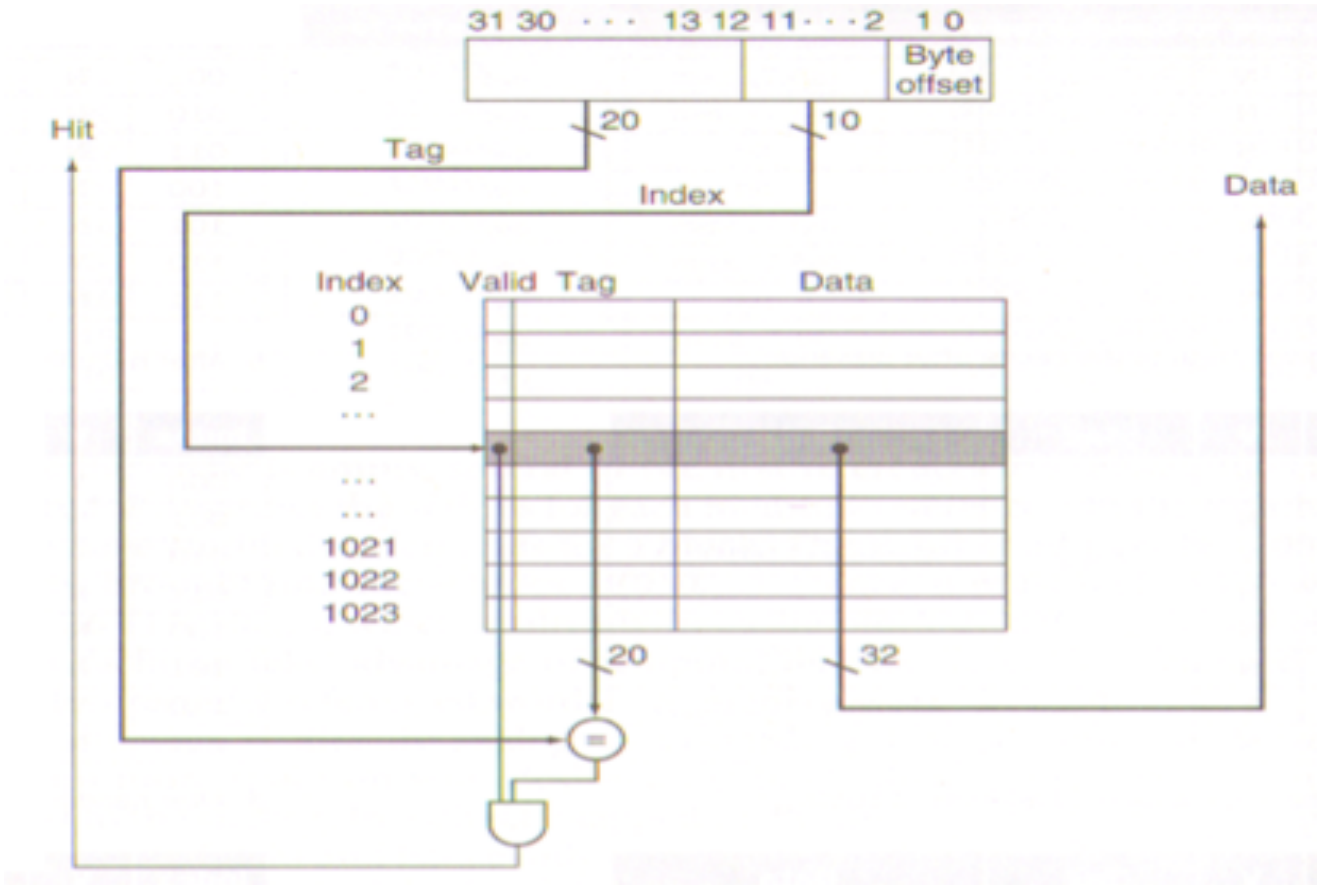
Partea din instrucțiune folosită pentru a selecta o intrare din memoria cache care conține o dată și un tag.



INDEX-ul unui bloc din memoria cache împreună cu conținutul TAG-ului blocului specifică în mod unic adresa de memorie

Memoria cache prezentată memorează 1024 cuvinte sau 4KB.

Avem 2^{10} cuvinte, 10 biți utilizați pentru indexarea memoriei cache => 32-10 = 20 biți pentru TAG

Cazul general:

1. **Adresele sunt pe 32 de biți**
2. **Memorie cache cu mapare directă**
3. **Memoria cache are 2^n blocuri => n biți folosiți pentru index**
4. **Dimensiunea blocului este de 2^m cuvinte (2^{m+2} bytes) => m biți folosiți pentru cuvânt în bloc și 2 biți utilizați pentru partea byte a adresei**

Câmpul TAG va avea dimensiunea: 32 – (n+m+2)

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

$$2^n \times (2^m \times 32 + (32 - n - m - 2) + 1)$$

Exemplu:

Câți biți sunt necesari pentru o memorie cache cu mapare directă în care se știu:

1. **16KB de date**
2. **Blocurile au dimensiunea de 4 cuvinte**
3. **Adresele sunt pe 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;

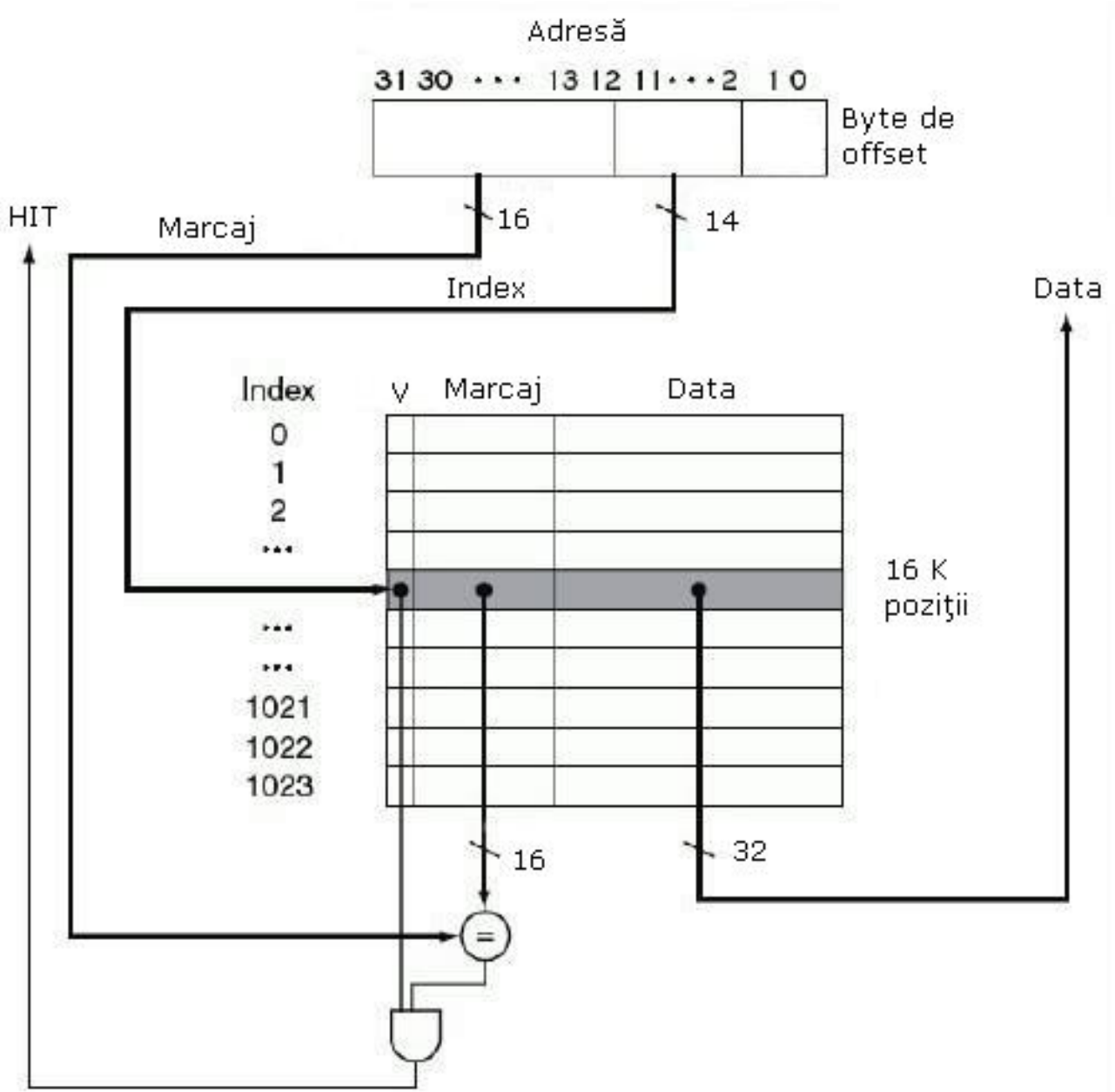
2). se instruiește memoria principală să execute o citire și așteaptă până când acesta termină accesul;

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
HIT
Marcaj
16
14
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

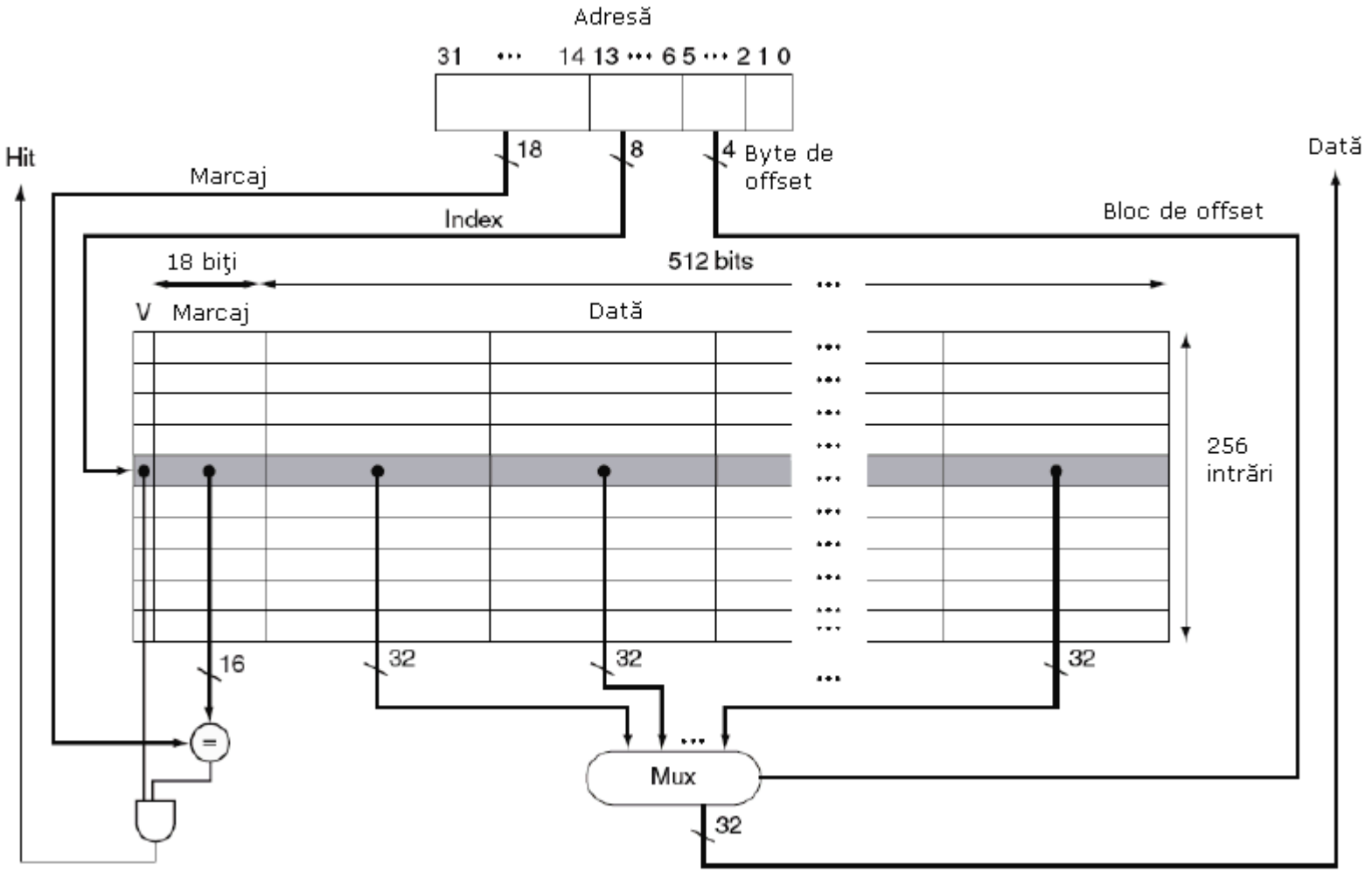
Dacă se scrie doar în memoria cache și nu și în memoria principală => memoria principală și memoria cache sunt inconsistente. Cea mai simplă metodă de evitare este scrierea în ambele memorii => **scriere simultană – write through**.

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

Exemplul 1 – se va efectua în clasă

Se consideră programul gcc având o rată de eșec de instrucțiuni de 2% și una de date de 4%. Dacă mașina de calcul are un CPI de 2 fără staționări din cauza memoriei și o penalizare de eșec de 40 cicluri de ceas pentru toate eșecurile, să se determine cât de rapid ar funcționa aceeași mașină dacă memoria cache ar fi Considerată ca fiind perfectă., fără eșecuri.

Exemplul 2 – se va efectua în clasă

Se presupune că se mărește performanța mașinii din exemplul anterior prin dublarea frecvenței ceasului. Deoarece viteza memorieiprincipale este puțin probabil să crească, se presupune că timpul absolut necesar tratării unui eșec va rămâne același.

Cât de rapidă va fi această mașină, dacă se va mări frecvența de ceas ?

Se presupune că frecvența de eșec este aceeași cu cea din exemplul anterior.