ARHITECTURA SISTEMELOR DE CALCUL
LUCRAREA DE LABORATOR NR. 5

# Memoria cache. Gestiunea memoriei cache

## Scopul lucrării

a) studiul caracteristicilor memoriei cache
b) studierea corespondenței dintre memoria principală şi memoria cache (metode de mapare a memoriei cache).
c)analiza performanțelor diferitelor metode de mapare cu ajutorul programului de simulare *simcache.exe*.

## 1.Memoria cache

Memoria cache reprezintă un *buffer* de mare viteză plasat între procesor şi memoria principală ce are drept scop să stocheze porțiuni din memoria principală care sînt în utilizare curentă. De asemenea memoria cache poate fi plasată între memoria principală şi memoria secundară (virtuală).

Datorită faptului că memoria cache este cu un ordin de mărime mai rapidă decît memoria principală, se poate reduce timpul efectiv de acces la memorie cu condiția ca aceasta să fie corect dimensionată şi să se utilizeze o politică adecvată de gestiune a memoriei cache.

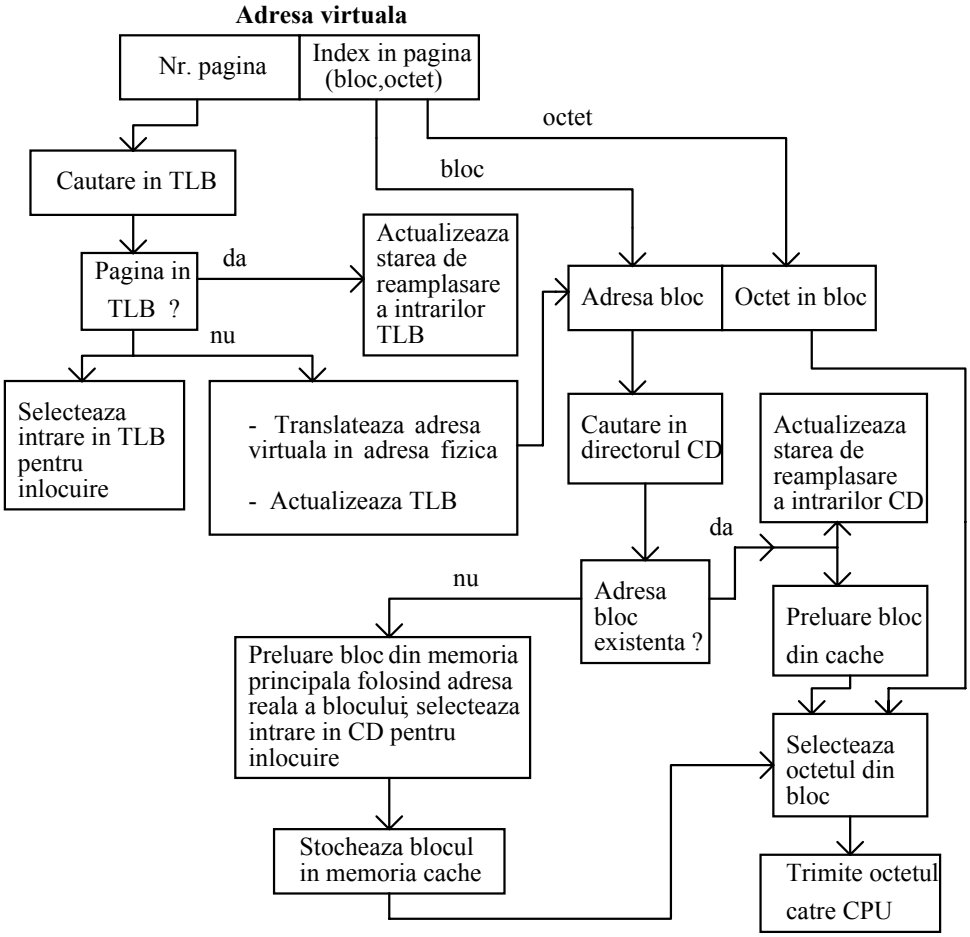



Figura 1. Operațiile cu memoria cache (fetch din cache)

Memoria cache este o memorie adresabilă prin conţinut şi este constituită dintrun index numit *cache directory* (CD) şi o *memorie RAM* (cu acces aleator).

Directorul CD constă în cîmpuri (tags) de adrese de blocuri de memorie şi biţi de control (de exemplu de protecţie). Cîmpurile de adrese "tags" conţin adresele blocurilor ce sînt la momentul de timp curent în memoria cache.

Adresa virtuală este translatată într-o adresă fizică de către sistemul de gestiune a memoriei.

Există o tabelă de corespondenţă între adresa virtuală şi adresa fizică (*Translation Lookahead Buffer* - TLB ) care conţine un număr de adrese virtuale (şi adresele fizice corespondente) cele mai recent utilizate.

Operaţia de access la memoria cache este prezentată în figura 1.

## 2. Organizarea memoriei cache

Există trei metode de bază pentru plasarea blocurilor din memoria principală în memoria cache (denumite asocieri sau mapări) :

- mapare directă
- mapare asociativă (*fully associative*)
- mapare asociativă pe seturi (*set associative*) .

Pentru simplificarea figurilor ce ilustrează organizarea memoriei cache (în aceste patru cazuri) se consideră dimensiunea memoriei cache de 2k cuvinte şi dimensiunea blocului de 16 cuvinte.

Memoria principală se alege de 256k cuvinte.

Adresa fizică va fi reprezentată pe 18 biţi ($256k = 2^8 . 2^{10} = 2^{18}$) . Exista $256k / 16 = 2^{18}/2^4 = 2^{14} = 16384$ blocuri în memoria principală şi $2k/16 = 2.2^{10} /2^4 = 128$ blocuri în memoria cache.

### *Maparea directă*

Maparea directă a memoriei cache este ilustrată în figura 2.

Adresa din memoria principală (adresa fizică) se compune din cîmpurile : *tag*, *bloc* şi *cuvînt.*

În acest caz blocul i din memoria principală este asociat cu blocul (i mod nr_blocuri_în _cache) = (i mod 128) din memoria cache. Fiecare bloc din cache are asociat un cîmp "tag" (primii 7 biţi mai semnificativi ai adresei fizice în memoria principală ).

Cîmpul tag din adresa memoriei principale se compară cu fiecare cîmp "tag" al memriei cach. După găsirea corespondenţei dintre cîmpurile tag se alege cîmpul "bloc" din adresa memoriei principale şi cîmpul "cuvînt" din aceeaşi adresă pentru selectarea octetului ce va fi transferat din blocul memoriei cache (dat de tag) în blocul memoriei principale (dat de cîmpul bloc şi cîmpul cuvînt al adresei în memoria principală).

Avantajul acestei mapări directe este accesul simultan la datele dorite şi la cîmpul tag (nu se face o căutare asupra cîmpului tag). Dacă cîmpul tag nu există în memoria cache , datele vor fi suprimate.

Un dezavantaj al mapării directe apare atunci cînd două sau mai multe blocuri din memoria principală, utilizate alternativ, sînt asociate aceluiaşi bloc din cache. Această situaţie are

probabilitate mică în sistemele uniprocesor, dar probabilitatea creşte pentru sistemele multiprocesor cu memorie cache comună.

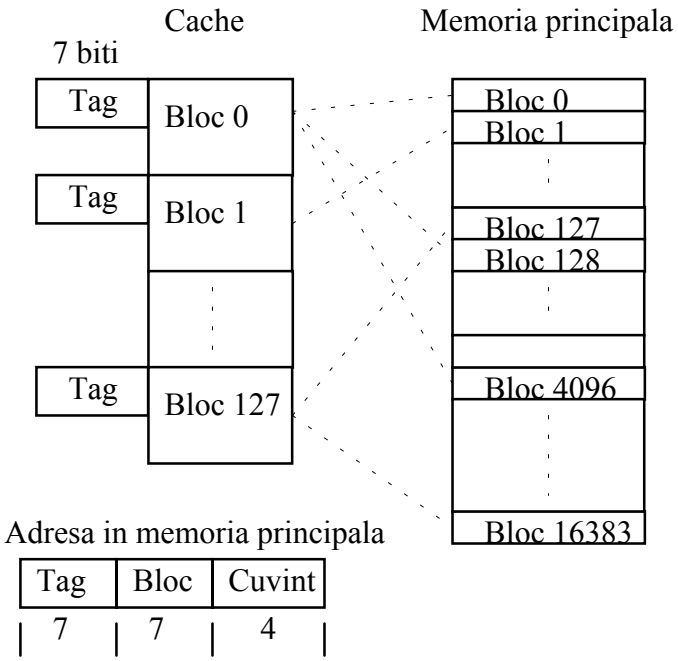



Figura 2. Organizarea cu mapare directă a memoriei cache

## *Maparea asociativă (fully associative)*

Maparea asociativă a memoriei cache este ilustrată în figura 3.
Adresa din memoria principală (adresa fizică) se compune din cîmpurile : *tag* şi *cuvînt.*

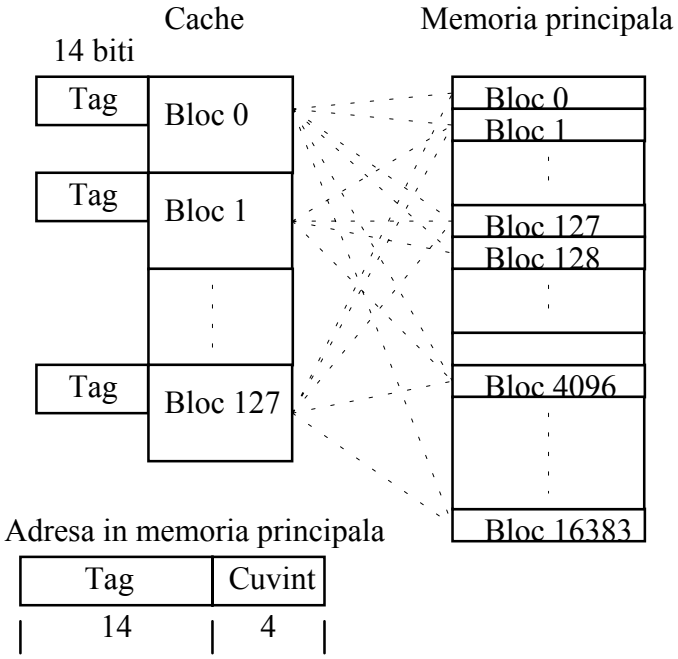



Figura 3. Organizarea cu mapare asociativă a memoriei cache

În acest caz *orice* bloc din memoria principală poate fi asociat cu *orice* bloc din memoria cache.

Metoda este cea mai bună, dar şi cea mai scumpă.

Maparea asociativă elimină conflictele de acces între blocuri, dar timpul de acces creşte datorită faptului că este necesară o căutare asociativă.

## *Maparea asociativă pe seturi*

Maparea asociativă pe seturi a memoriei cache este ilustrată în figura 4.

Adresa din memoria principală (adresa fizică) se compune din cîmpurile : *tag*, *set* şi *cuvînt.*

Această mapare reprezintă un compromis între maparea directă şi maparea asociativă. Memoria cache este împărţită în S seturi . Blocul i , din memoria principală, poate corespunde cu orice bloc din memoria cache aparţinînd setului (i mod S).

Numărul de seturi , S, determină costul căutării (specifice mapării asociative). Setul se determină direct din adresă, apoi se caută cîmpul tag corespunzător adresei în set.

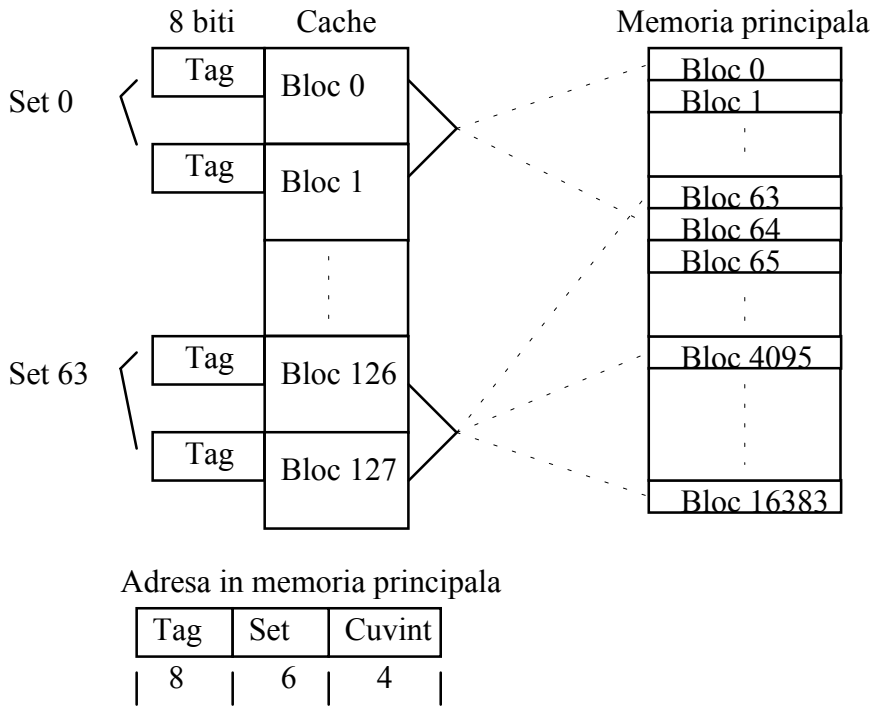



Figura 4. Organizarea cu mapare asociativă pe seturi a memoriei cache

## *Observaţii*

- dimensiunea blocului în memoria cache este aleasă în raport cu proprietatea de *localizare* a programelor .

- memoria cache este adresată cu adrese fizice şi nu cu adrese virtuale (se elimină etapa de translatare a adresei virtuale în adresă fizică)

- algoritmii de reamplasare a blocurilor din memoria cache determină performanţa memoriei cache, exprimată ca *rată de pierderi* (număr de încercări eşuate de a găsi informaţia dorită în cache).

## 3.Politici de reamplasarea blocurilor din memoria cache

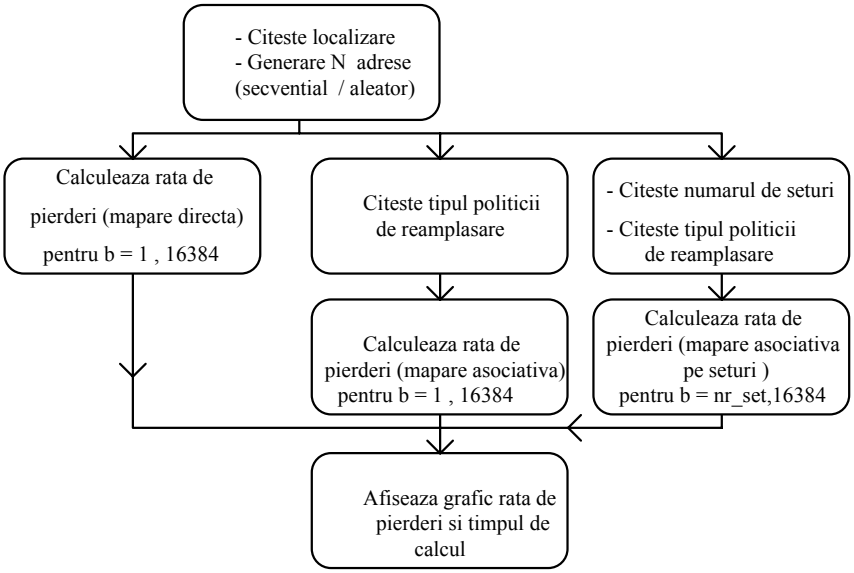
Există următoarele politici de reamplasare a blocurilor din memoria cache:

1. **Least recently used (LRU)** - se înlocuieşte blocul din cache cel mai puţin recent utilizat , relativ la momentul de timp curent (se face o analiza a utilizarii blocurilor existente în memoria cache cu un anumit număr de momente de timp în urmă în raport cu momentul de timp curent, la care trebuie să se producă o reamplasare).
2. **First In First Out (FIFO)** - se înlocuieşte blocul din cache care a stat în memoria cache cel mai mare interval de timp.
3. **Last In First Out (LIFO)** - se înlocuieşte blocul din cache care a stat în memoria cache cel mai mic interval de timp.
4. **Random (RAND)** - se alege, în mod aleator ,un bloc din memoria cache ce va fi înlocuit.

## 4. Programul de simulare a metodelor de mapare a memoriei cache

Pentru evaluarea performanţelor diferitelor metode de mapare a memoriei cache se va utiliza programul de simulare *simcache*, care ilustrează grafic *rata de pierderi* calcualtă ca raport între numărul de încercări eşuate de găsire a informaţiei dorite în cache şi numărul total de accese la memoria cache.

Programul simcache are următoarea organigramă generală (figura 5) :




b = numarul de blocuri din memoria cache N = 500 (numar de realizari)
nr_set = numar de seturi

Figura 5. Organigrama programului SIMCACHE

ARHITECTURA SISTEMELOR DE CALCUL
LUCRAREA DE LABORATOR NR. 5

Pentru o sesiune de lucru cu programul *simcache* trebuie date urmatoarele comenzi :

SIMULATOR MEMORIE CACHE

Generare adrese aleator/secvential [a/s] :s

Gradul de localizare [0 : fara localizare] :100

START SIMULARE MAPARE DIRECTA: _____ END

Numar de seturi (exprimat in biti) :3 <---- **nr. seturi = $2^k$**

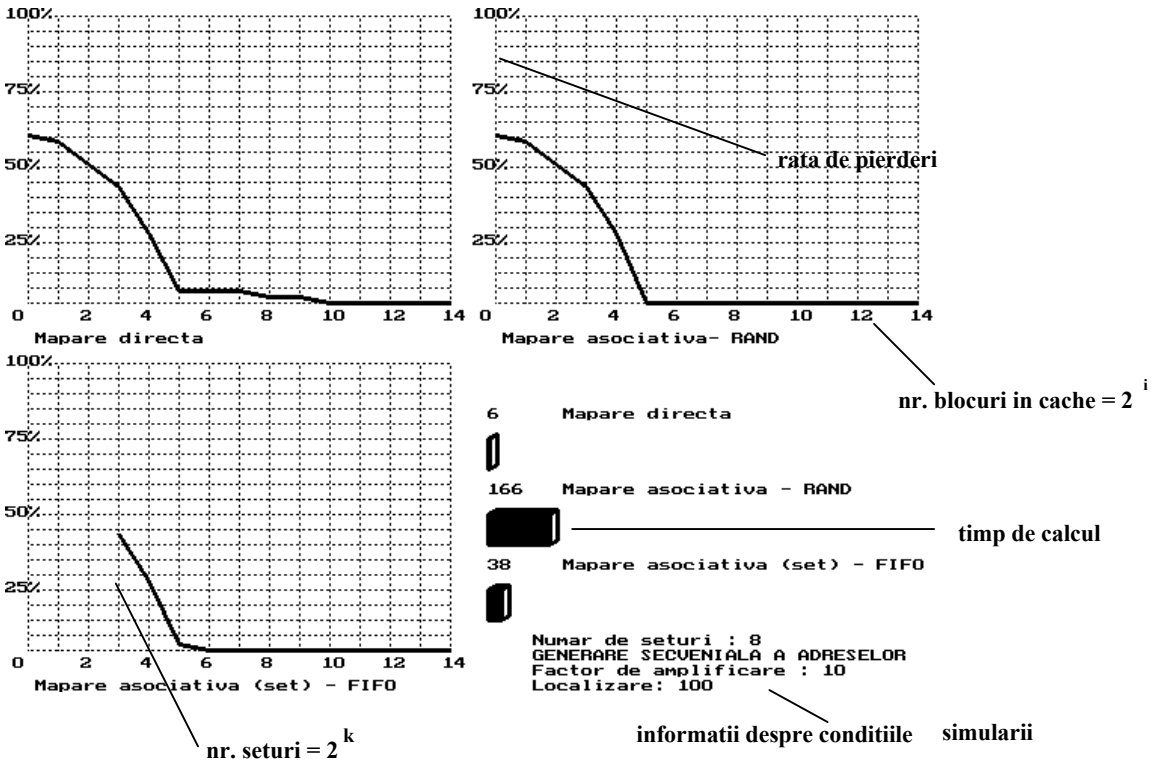
Politici de reamplasare : 0->RAND 1->LRU 2->FIFO 3->LIFO

Politica de reamplasare mapare asociativa :0
Politica de reamplasare mapare asociativa pe seturi :2

START SIMULARE MAPARE ASOCIATIVA: _____ END

START SIMULARE MAPARE ASOCIATIVA PE SETURI: _____ END

Factor de amplificare a afisarii :10




Modificare factor de amplificare ? [d/n] : n

Iesire din simulator ? [d/n] : d

Gradul de localizare indică gama de generare a adreselor (diferența între adresa maximă şi minimă generate) în multiplii de 16 blocuri . Pentru grad de localizare nul adresele se generează pe o gamă foarte mare (pe 28 biți).

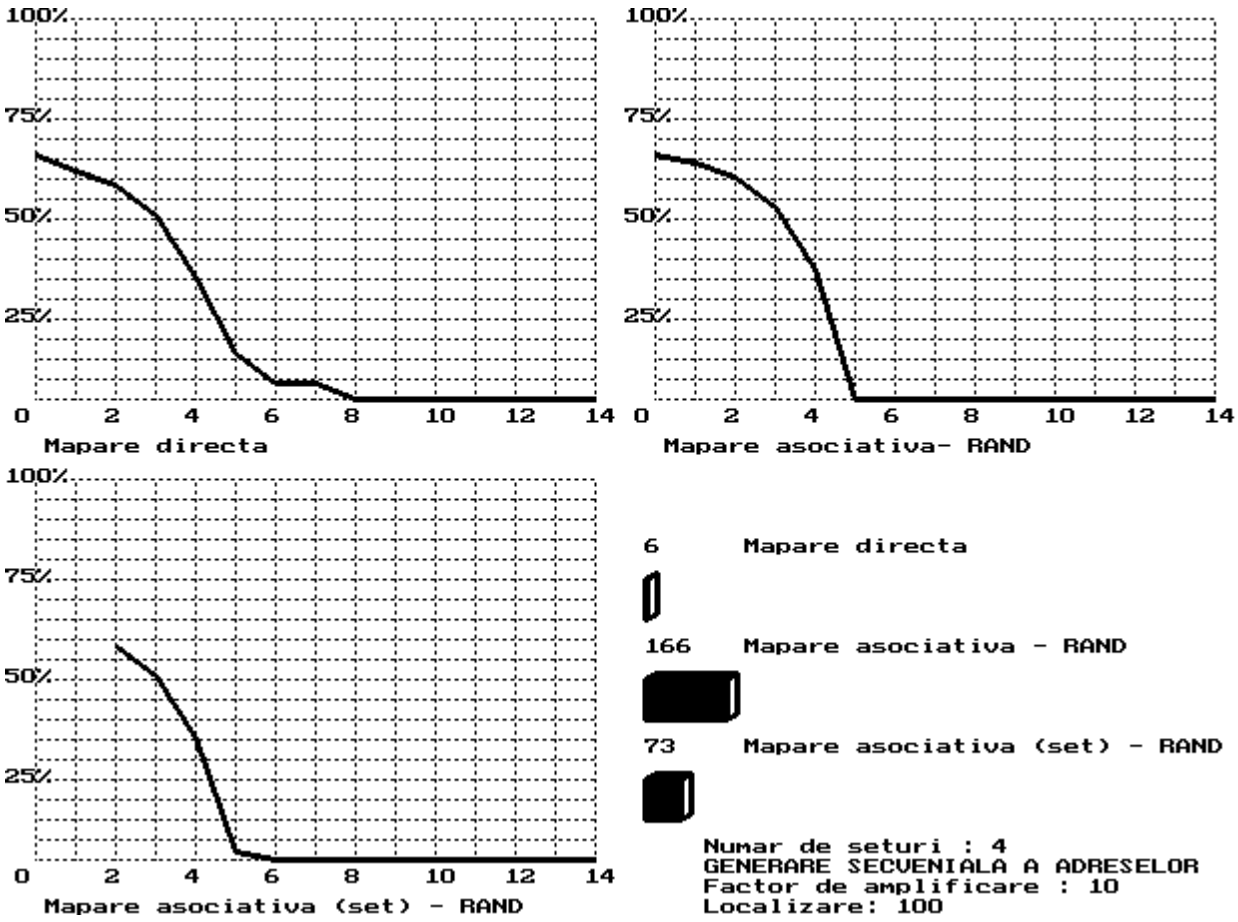
Adresele se pot genera aleator pe întreg intervalul de 28 biti sau pe un interval dat de gradul de localizare.

Generarea secvențială a adreselor presupune că se generează aleator o adresă de start şi o adresă de subrutină după care se generează un număr de adrese la rînd (secvențial) apoi se simulează saltul la subrutină şi revenirea din subrutină. Toate adresele au valori corespunzătoare gradului de localizare.

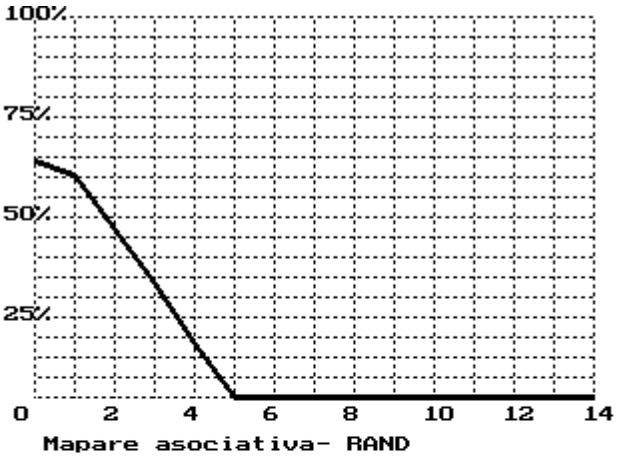
Timpul de calcul este măsurat prin întreruperi periodice generate de ceasul de timp real al PC -ului.

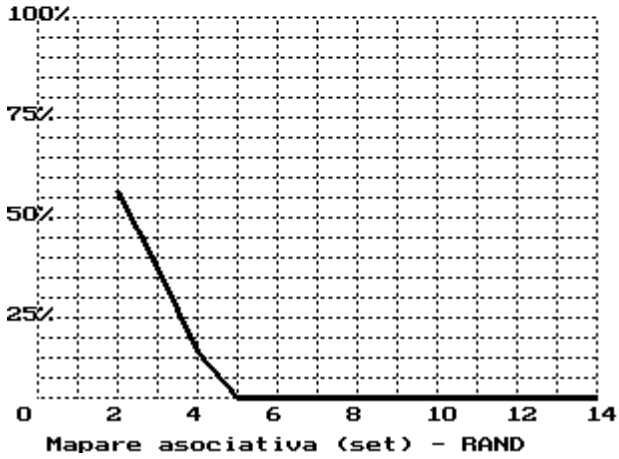
## 5. Exemple de rezultate

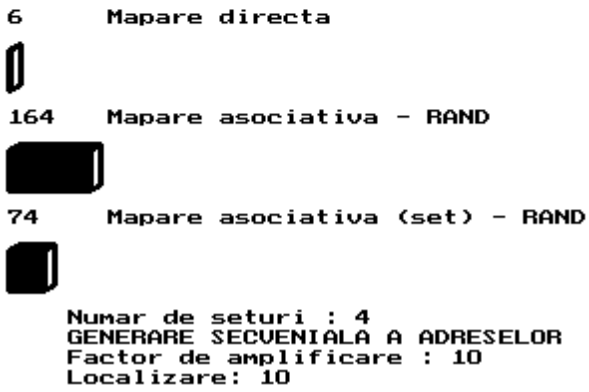
În continuare sînt prezentate cîteva dintre rezultatele simulărilor cu programul *simcache*:

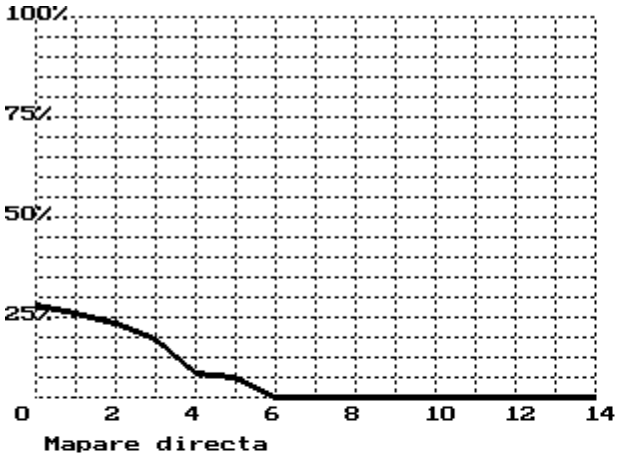

100%
75%
50%
25%
0 2 4 6 8 10 12 14
Mapare directa
Mapare asociativa- RAND
Mapare asociativa (set) - RAND
6 Mapare directa
166 Mapare asociativa - RAND
73 Mapare asociativa (set) - RAND
Numar de seturi : 4
GENERARE SECVENIALA A ADRESELOR
Factor de amplificare : 10
Localizare: 100

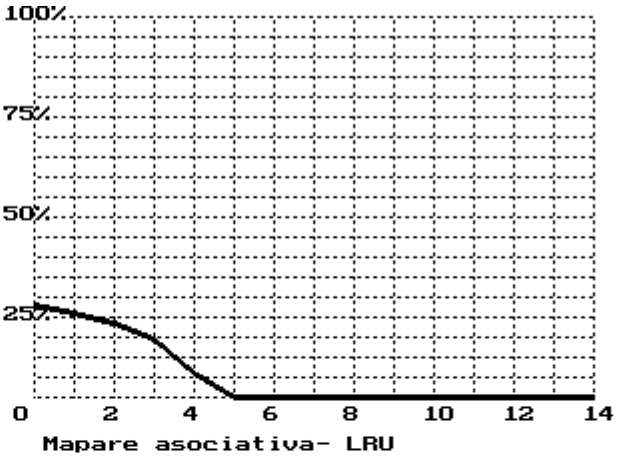
100%
75%
50%
25%
0
2
4
6
8
10
12
14
Mapare directa

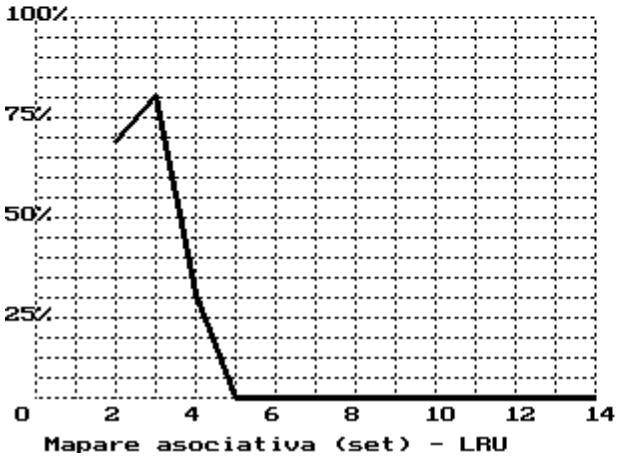

100%
75%
50%
25%
0
2
4
6
8
10
12
14
Mapare asociativa- RAND

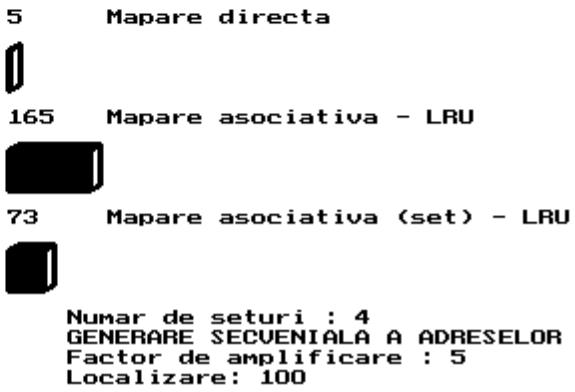

100%
75%
50%
25%
0
2
4
6
8
10
12
14
Mapare asociativa (set) - RAND

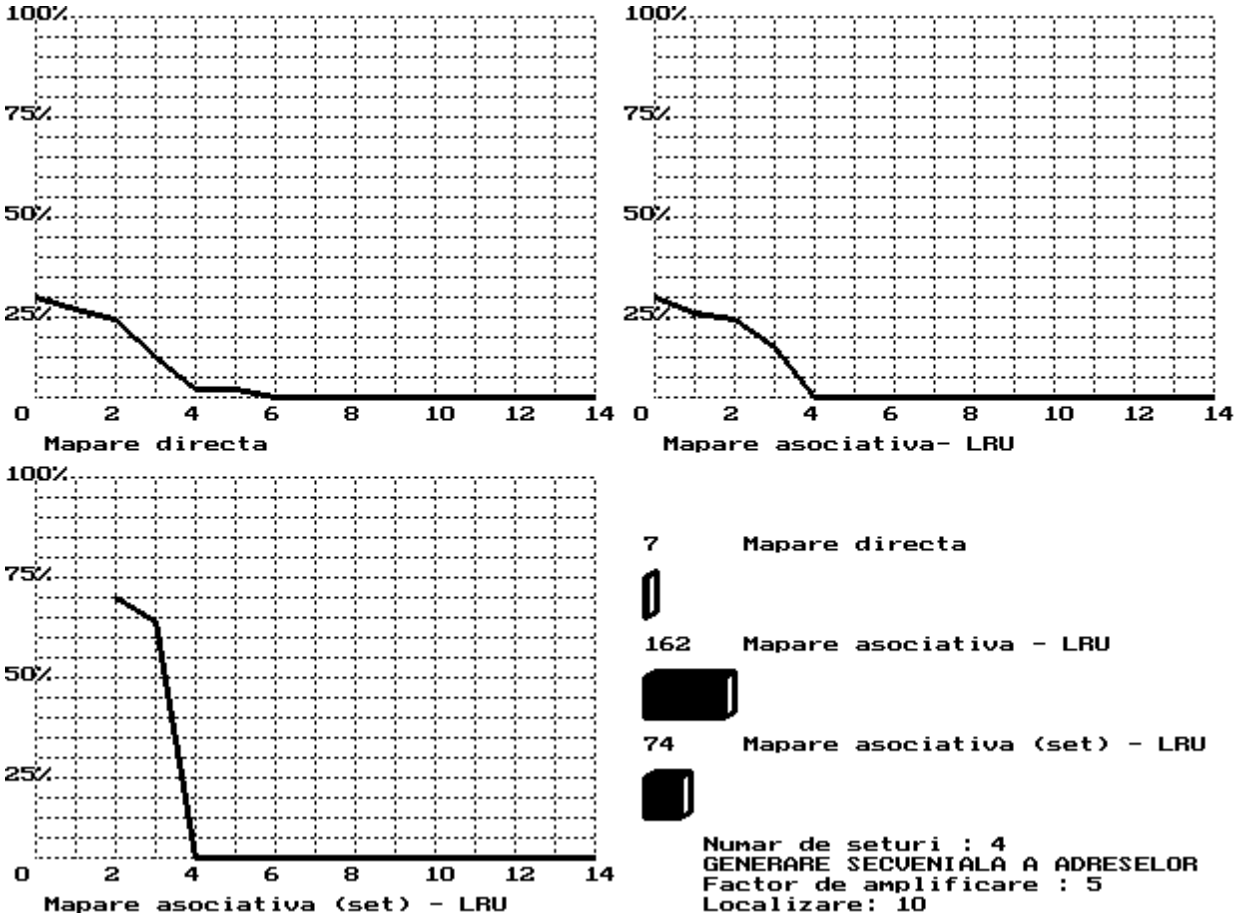

6 Mapare directa
164 Mapare asociativa - RAND
74 Mapare asociativa (set) - RAND
Numar de seturi : 4
GENERARE SECVENIALA A ADRESELOR
Factor de amplificare : 10
Localizare: 10


100%
75%
50%
25%
0 2 4 6 8 10 12 14
Mapare directa


100%
75%
50%
25%
0 2 4 6 8 10 12 14
Mapare asociativa- LRU


100%
75%
50%
25%
0 2 4 6 8 10 12 14
Mapare asociativa (set) - LRU


5 Mapare directa
165 Mapare asociativa - LRU
73 Mapare asociativa (set) - LRU
Numar de seturi : 4
GENERARE SECVENIALA A ADRESELOR
Factor de amplificare : 5
Localizare: 100




## 6. Desfăşurarea lucrării

a) Să se studieze metodele de mapare descrise.
b) Să se ruleze programul de simulare *simcache* după următorul tabel :

| Metoda mapare | Generare adresa | Localizare | Nr. seturi | Politica reamplasare | Timp de calcul | Observatii referitoare rata de pierderi |
|---|---|---|---|---|---|---|
| directa | aleator | 0 | - | - | | |
| directa | secvential | 10 | - | - | | |
| directa | secvential | 100 | - | - | | |
| directa | secvential | 1000 | - | - | | |
| asociativa | aleator | 0 | - | RAND | | |
| asociativa | aleator | 0 | - | FIFO | | |
| asociativa | aleator | 0 | - | LIFO | | |
| asociativa | aleator | 0 | - | LRU | | |
| asociativa | secvential | 100 | - | RAND | | |
| asociativa | secvential | 100 | - | FIFO | | |
| asociativa | secvential | 100 | - | LIFO | | |
| asociativa | secvential | 100 | - | LRU | | |
| asociativa set | aleator | 0 | 1,2,4,8,16 | RAND | | |
| asociativa set | aleator | 0 | 1,2,4,8,16 | FIFO | | |
| asociativa set | aleator | 0 | 1,2,4,8,16 | LIFO | | |
| asociativa set | aleator | 0 | 1,2,4,8,16 | LRU | | |
| asociativa set | secvential | 100 | 1,2,4,8,16 | RAND | | |
| asociativa set | secvential | 100 | 1,2,4,8,16 | FIFO | | |
| asociativa set | secvential | 100 | 1,2,4,8,16 | LIFO | | |
| asociativa set | secvential | 100 | 1,2,4,8,16 | LRU | | |

## Temă

a) Să se discute avantajele şi dezavantajele diferitelor metode de mapare din punctul de vedere al ratei de pierderi şi a timpului de calcul.

b) Care este influenţa gradului de localizare asupra performanţelor memoriei cache ?

c) Care este influenţa numărului de seturi asupra performanţelor memoriei cache ?