

# Evaluarea performantelor structurilor pipe-line

## 1. Principiul pipe-line (paralelism temporal). Cresterea vitezei de prelucrare in raport cu o structura non-pipe-line

Cresterea vitezei de prelucrare intr-o structura pipe-line ideala, in raport cu o structura non-pipe-line echivalenta, este ilustrata in figura 1.

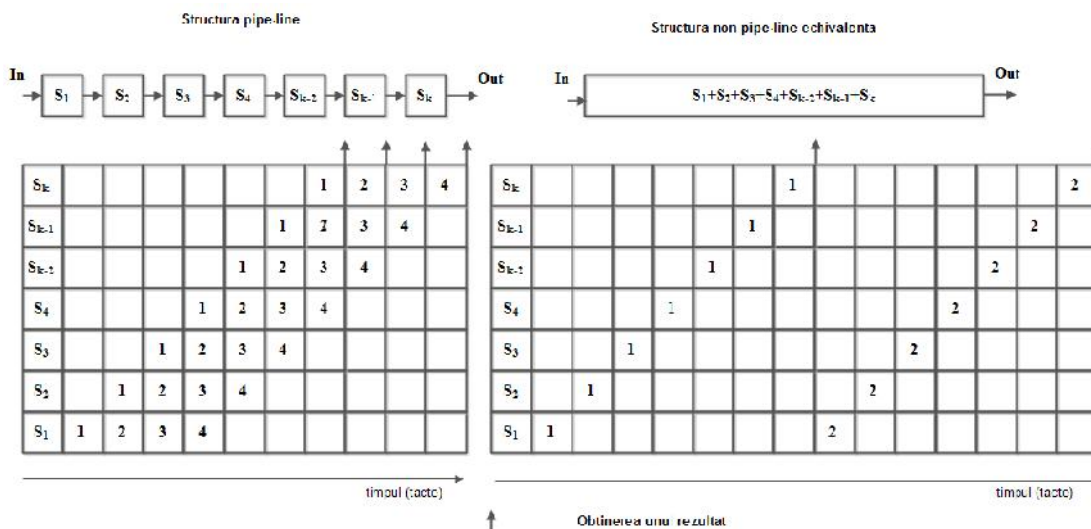


Figura 1. Cresterea vitezei de prelucrare intr-o structura pipe-line ideala

Totusi, intr-o structura pipe-line reala, apar urmatoarele aspecte care influenteaza performanta, in sensul scaderii vitezei de prelucrare. Acesti factori sint:

- Echilibrarea vitezelor de prelucrare a stagiilor pipe-line (nu toate stagiile de prelucrare au, intotdeauna, aceeasi viteză; de exemplu accesul la memorii, pentru incarcarea operanzilor, poate fi uneori mai lent)
- Asigurarea incarcarii continue cu operanzi de intrare (instructiuni) – trebuie asigurate mecanisme rapide pentru incarcarea instructiunilor
- Tipul instructiunii executate in structura pipe-line (comutari de context: instructiuni de salt, salt conditionat, apeluri de subrutina, dependentele dintre instructiuni, instructiunile mai lente: accesul la memoria de date, instructiuni de virgule mobile – conduc la anulara unor operatii deja efectuate si/sau la oprirea temporara (*stall*) a functionarii structurii pipe-line, prin introducerea unor *goluri* (*holes*) – tacte de neutilizare a stagiilor pipe-line.)

## 2. Performanta unei structuri pipe-line non-ideale

### a. Expresia generala a performantei unei structuri pipe-line reale

Performanta unei structuri pipe-line cu  $k$  stagii este data de expresia:

$$P = \lim_{n \rightarrow \infty} \frac{T_k}{T_k + T_s}$$

unde  $T_k$  este timpul (in numar de tacte pipe-line) necesar executiei a  $n$  instructiuni in structura pipe-line cu  $k$  stagii, iar  $T_s$  este timpul suplimentar (in numar de tacte pipe-line) datorat tipului de instructiune executata.

Exista relatia:  $T_k = k + n - 1$ , iar  $T_s$  depinde de tipul instructiunii executate.

### b. Performanta unei structuri pipe-line cu instructiuni de salt

Aparitia unei instructiuni de salt conduce la necesitatea anularii unui numar de instructiuni deja incarcate in structura pipe-line. Numarul acestor instructiuni depinde de momentul in care se determina adresa de salt (ca in figura 2).

Timpul suplimentar este:

$$T_s = n \cdot p_1 \cdot t_1$$

unde  $p_1$  este probabilitatea ca instructiunea executata sa fie de salt si  $t_1$  este numarul de tacte necesar anularii instructiunilor deja incarcate in mod eronat in structura pipe-line.

Rezulta performanta:

$$P = \lim_{n \rightarrow \infty} \frac{T_k}{T_k + T_s} = \lim_{n \rightarrow \infty} \frac{k + n - 1}{k + n - 1 + n \cdot p_1 \cdot t_1} = \frac{1}{1 + p_1 \cdot t_1}$$

O situatie echivalenta este aceea a apelurilor de subrutina si a intreruperilor.

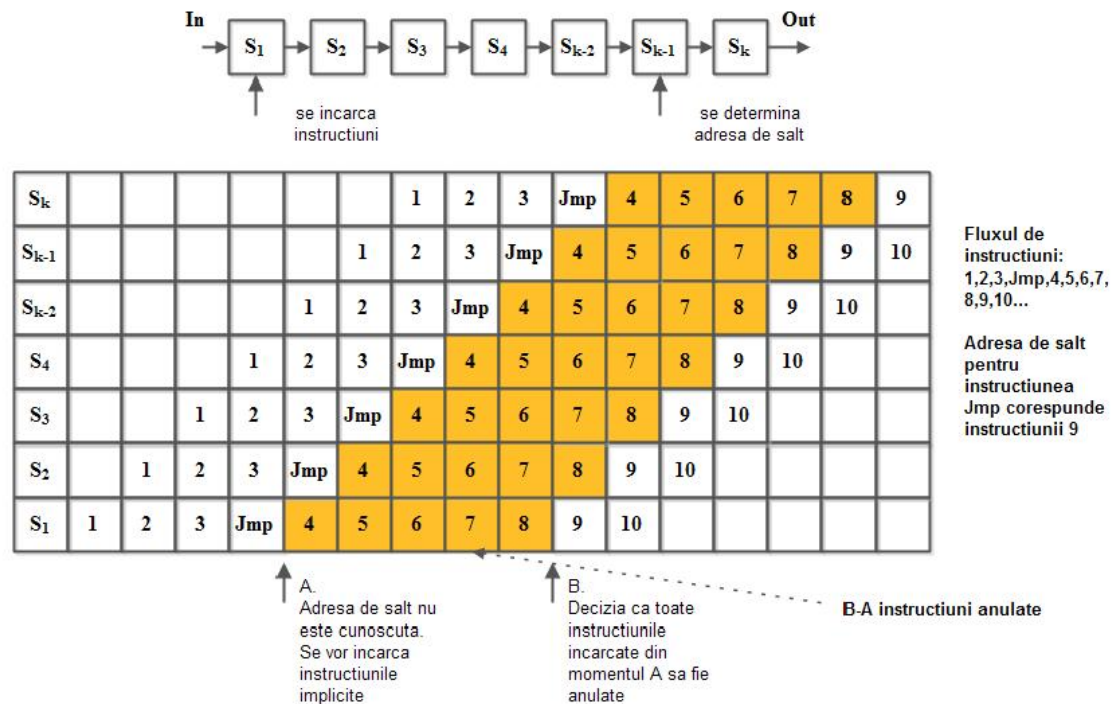


Figura 2. Aparitia instructiunilor de salt intr-o structura pipe-line

### c. Performanta unei structuri pipe-line cu instructiuni de salt conditionat

Aparitia unei instructiuni de salt conditionat este similara aparitiei unei instructiuni de salt (neconditionat). Figura 3 ilustreaza efectul aparitiei unei instructiuni de salt conditionat.

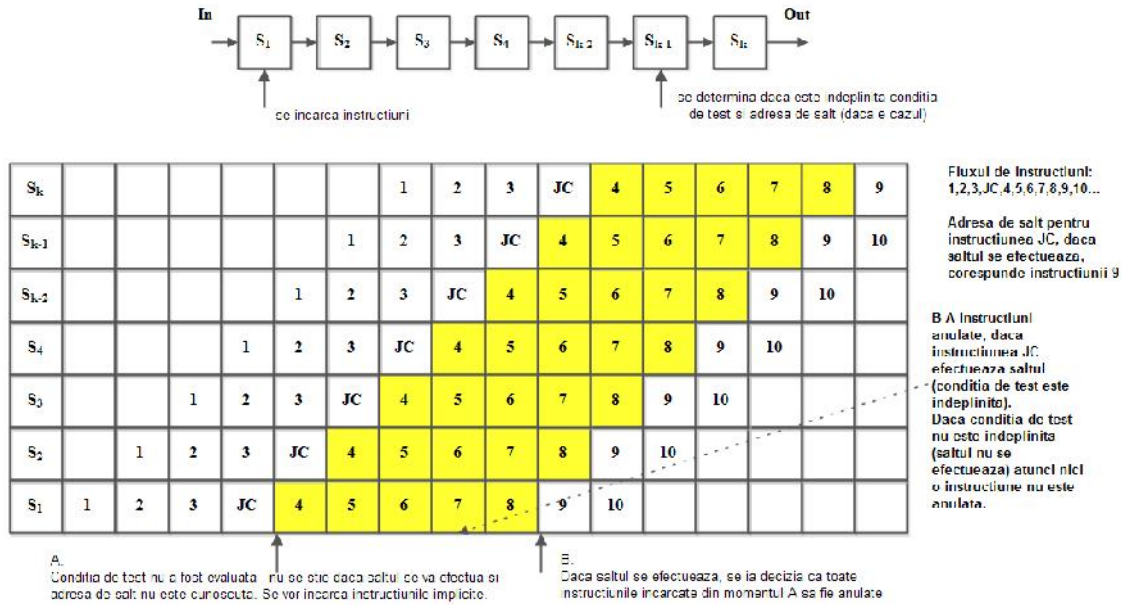


Figura 3. Aparitia instructiunilor de salt conditionat intr-o structura pipe-line

Timpul suplimentar este:

$$T_s = n \cdot q \cdot p_2 \cdot t_2$$

unde  $p_2$  este probabilitatea ca instructiunea executata sa fie de salt conditionat,  $q$  este probabilitatea ca saltul se efectueze (*taken*) si  $t_2$  este numarul de tacte necesar anularii instructiunilor deja incarcate in mod eronat in structura pipe-line.

Rezulta performanta:

$$P = \lim_{n \rightarrow \infty} \frac{T_k}{T_k + T_s} = \lim_{n \rightarrow \infty} \frac{k + n - 1}{k + n - 1 + n \cdot q \cdot p_2 \cdot t_2} = \frac{1}{1 + q \cdot p_2 \cdot t_2}$$

### d. Performanta unei structuri pipe-line cu instructiuni dependente

In cazul aparitiei unei instructiuni dependente de o instructiune incarcata cu  $m$  tacte anterior, este necesar ca instructiunea care produce dependenta sa se termine. Figura 4 ilustreaza aparitia unei instructiuni dependente.

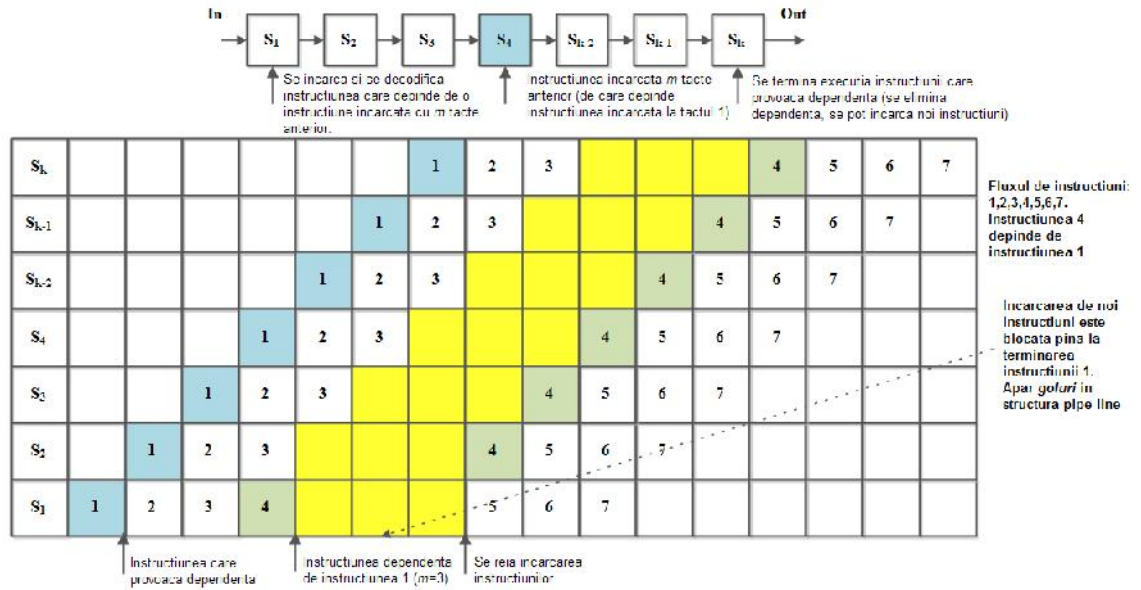


Figura 4. Aparitia instructiunilor dependente intr-o structura pipe-line

Timpul suplimentar necesar este:

$$T_s = n \cdot p_3 \cdot (k - m - 1)$$

unde  $p_3$  este probabilitatea ca instructiunea sa fie dependenta de o instructiune incarcata cu  $m$  tacte anterior.

Trebuie indeplinita, in mod evident, conditia:  $m < k$ .

Rezulta performanta:

$$P = \lim_{n \rightarrow \infty} \frac{T_k}{T_k + T_s} = \lim_{n \rightarrow \infty} \frac{k + n - 1}{k + n - 1 + n \cdot p_3 \cdot (k - m)} = \frac{1}{1 + p_3 \cdot (k - m - 1)}$$

#### e. Performanta unei structuri pipe-line cu instructiuni de acces la memorie

Aparitia unei instructiuni de acces la memorie are ca efect introducerea unui timp suplimentar datorat timpului de acces la memorie. Se considera ca, in general, instructiunile sint incarcate din registre sau memorii cache rapide ce lucreaza la viteza structurii pipe-line.

Figura 5 prezinta situatia in care o instructiune este incarcata dintr-o memorie lenta sau a aparut o situatie de *miss cache*.



### 3. Metode pentru evitarea degradarii performantelor structurilor pipe-line

Exista mai multe metode prin care efectele prezentate anterior sa fie atenuate. Pe langa faptul ca se poate incerca scrierea unor programe cu un numar relativ mic de instructiuni de comutare a contextului (salturi, salturi conditionate, apeluri de subrutine), se pot aplica urmatoarele procedee software sau hardware:

- a. Rearanjarea instructiunilor
- b. Utilizarea unor predictoare de salturi
- c. Utilizarea unor metode rapide de acces la memorie

Rearanjarea instructiunilor (efectuata de catre compilator sau manual) micsoreaza termenul  $(k - m - 1)$  ceea ce reduce timpul suplimentar asociat instructiunilor dependente.

Predictorul de salturi va reduce numarul de instructiuni incarcate in mod eronat, prin aceea ca decizia efectuarii saltului se va lua cit mai devreme.

Metodele rapide de acces la memorie (accesul simultan sau accesul concurrent, utilizarea de memorii cache cu o metoda de mapare ce conduce la o rata *miss cache* redusa) elimina problema acceselor lente la memorie.

### 4. Descrierea simulatorului pentru evaluarea performantelor structurilor pipe-line

Evaluarea performantelor structurii pipe-line se efectueaza cu ajutorul unui simulator, **simpl**, cu structura din figura 6

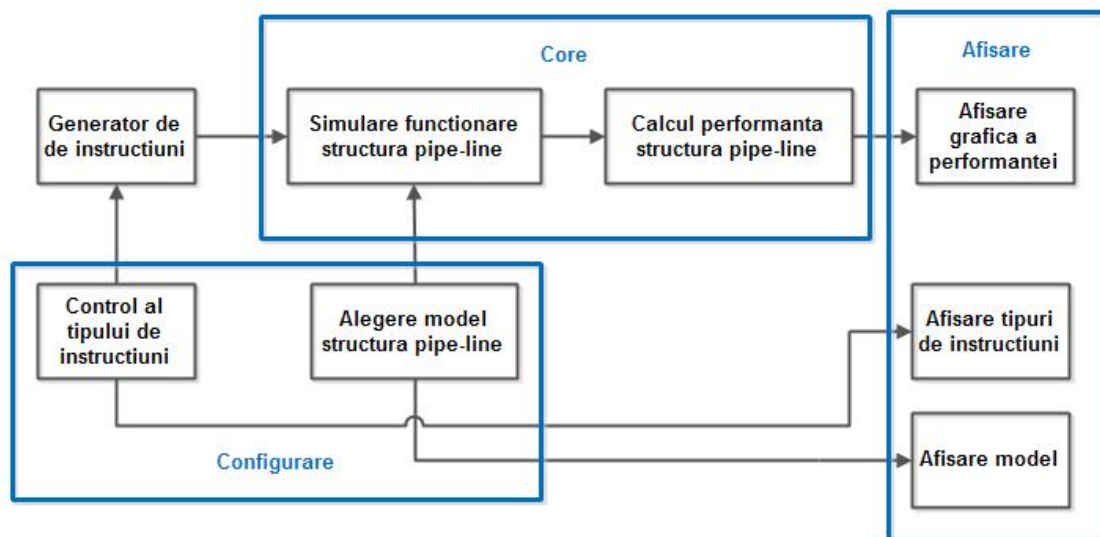


Figura 6. Structura simulatorului **simpl** (SIMulator Pipe-Line)

Simulatorul **simpl** are 4 blocuri funcționale: un bloc principal – **core**, un bloc de configurare, un bloc de generare a tipurilor de instructiuni si un bloc de afisare.



Se pot simula 6 modele de structuri pipe-line ilustrate in figura 7. Parametrii acestor modele sint prezentati in tabelul 1.

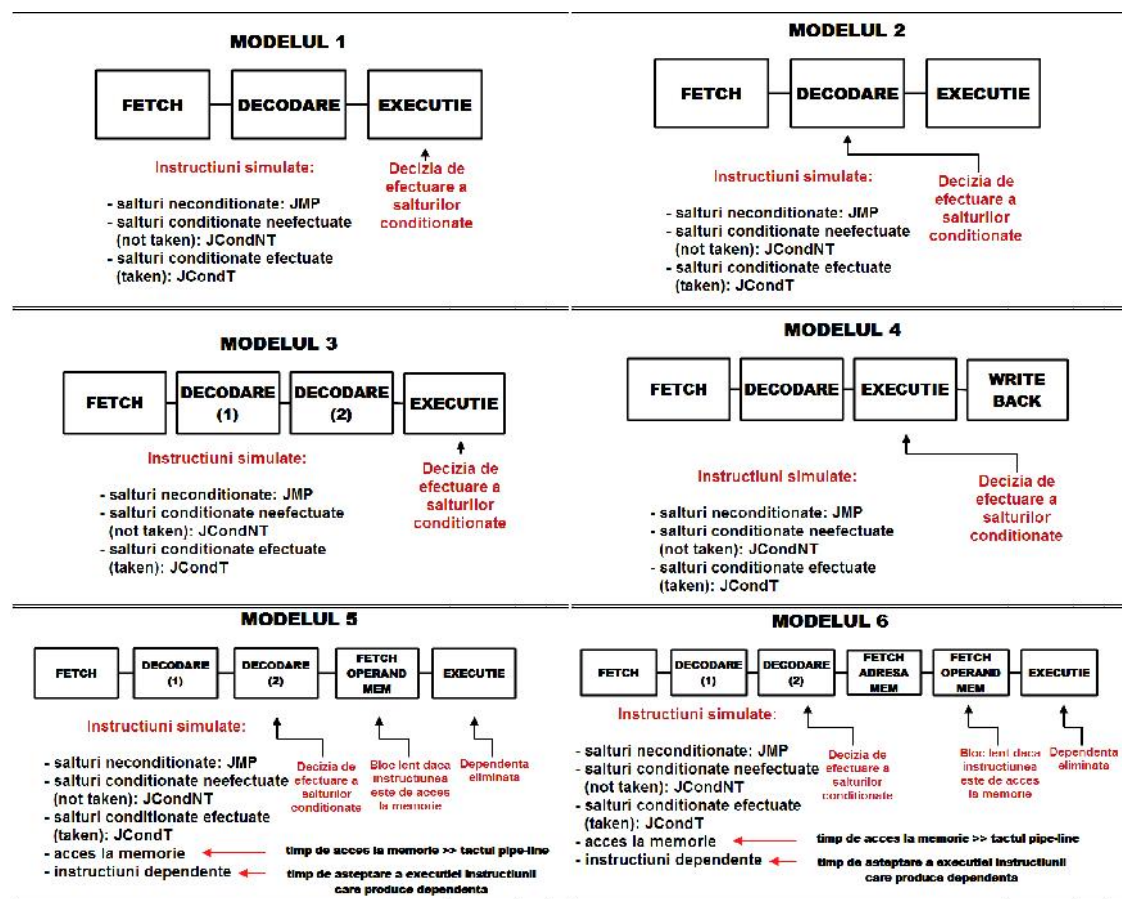


Figura 7. Modelele structurilor pipe-line simulate in simulatorul **simpl**

Tabelul 1. Modele simulate in **simpl**

Model	Numar de stagii	Tipuri de instructiuni	Timp suplimentar pentru o instructiune de salt	Timp suplimentar pentru o instructiune dependenta	Timp suplimentar pentru o instructiune de acces la memorie
1	3	-salt -salt conditionat	2	-	-
2	3	-salt -salt conditionat	1	-	-
3	4	-salt -salt conditionat	3	-	-
4	4	-salt -salt conditionat	2	-	-
5	5	-salt -salt conditionat -dependente -acces la memorie	2	4-m	9
6	6	-salt -salt conditionat -dependente -acces la memorie	2	5-m	9

În aceste modele, testarea condiției pentru salturile condiționate, calculul adresei de salt în situația în care condiția de salt este îndeplinită și calculul adresei de salt pentru salturile necondiționate se efectuează în același stadiu.

Modul de lucru cu simulatorul **simpl** este ilustrat în continuare.

## 1. Lansarea programului de evaluare a performanțelor structurilor pipe-line, **simpl**

După lansarea în execuție a programului **simpl**, se deschid 2 ferestre: o fereastră principală și o fereastră de configurare (figura 8).

Fereastră principală conține meniurile de selectarea a modelului pipe-line simulat (*Models*), resetarea programului (*Clear*), afișarea rezultatelor simulării (*Results*), *Help* și ieșire din program (*Quit*).

Fereastră de configurare setează tipurile de instrucțiuni și probabilitățile de apariție a acestora și controlează modul de afișare a diagramei spatio-temporale ce ilustrează funcționarea structurii pipe-line.

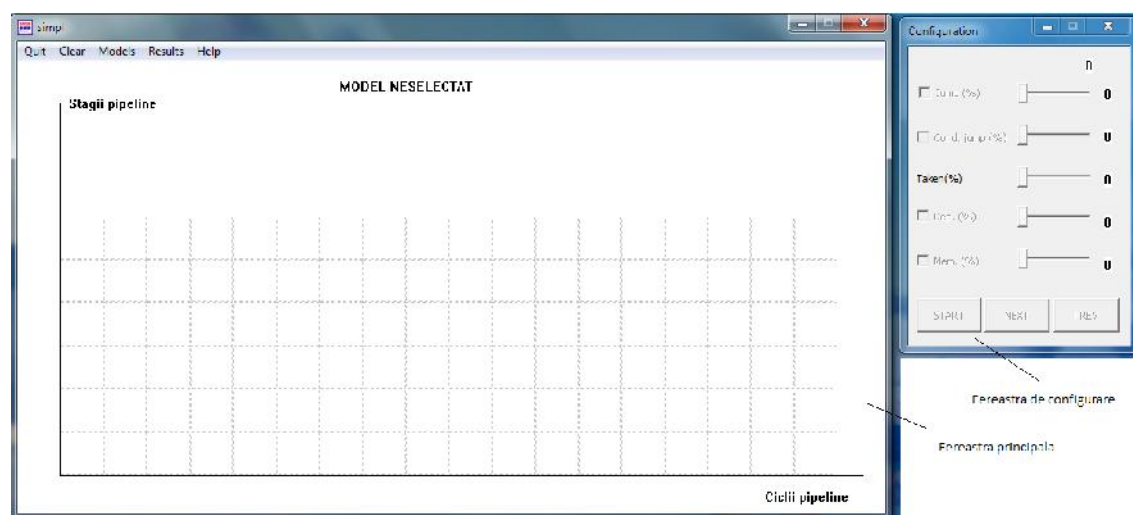



Figura 8. Ferestrele afișate după lansarea programului **simpl** 

## 2. Meniurile ferestrei principale (figura 9).

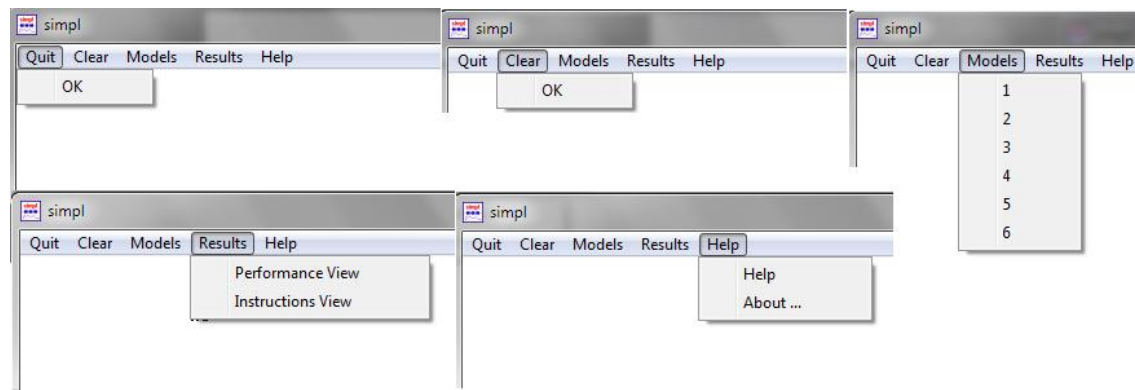



Figura 9. Meniurile ferestrei principale a programului **simpl** 



### 3. Meniurile ferestrei de configurare (figura 10).

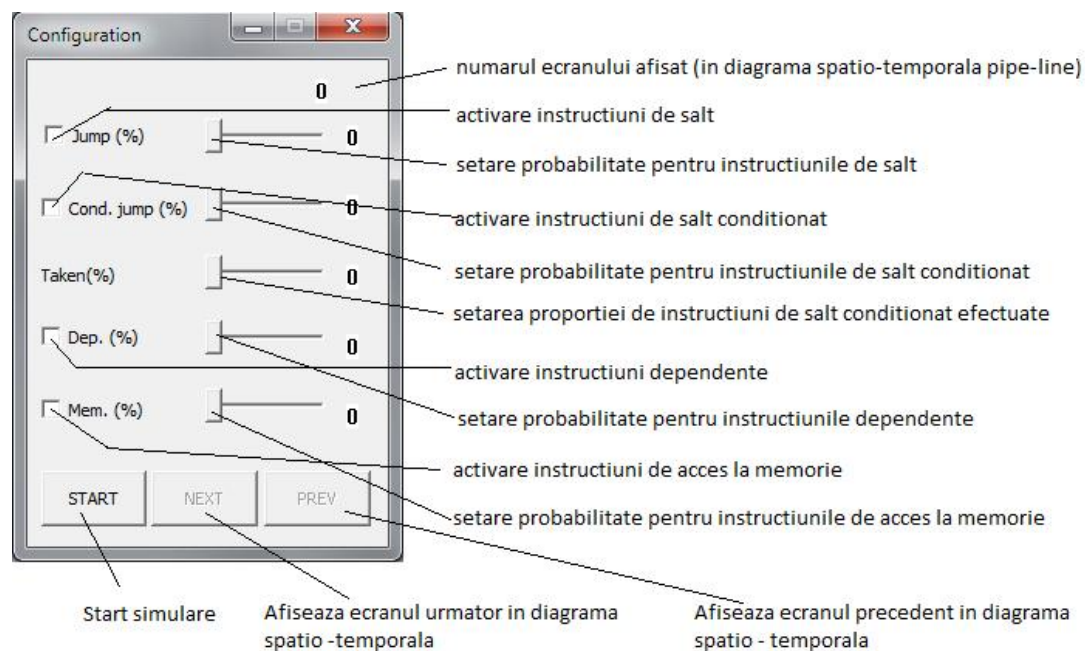



Figura 10. Meniurile ferestrei de configurare a programului **simpl** 

### 4. Meniurile ferestrei de afisare a performantelor (figura 11).

Se afiseaza grafic performanta structurii pipe-line conform modelului selectat sau structura pipe-line selectata pentru simulare (ca in figura 7).

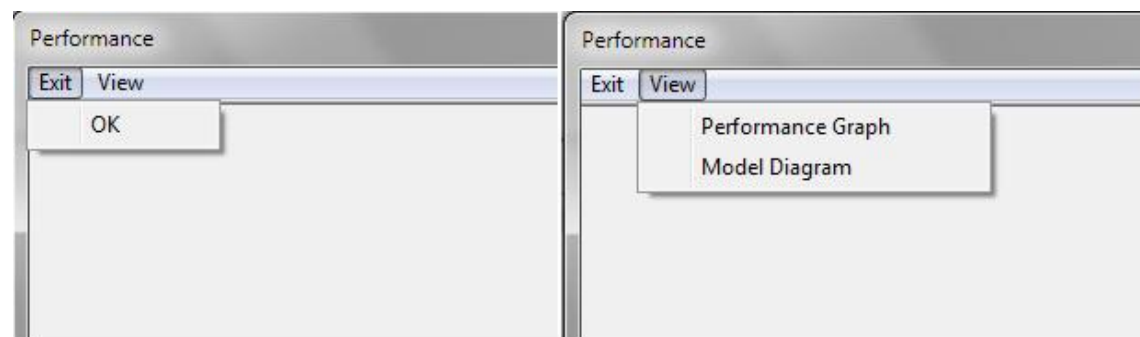



Figura 11. Meniurile ferestrei de afisare a performnatelor a programului **simpl** 

Performanta structurii pipe-line este afisata ca in figura 12.

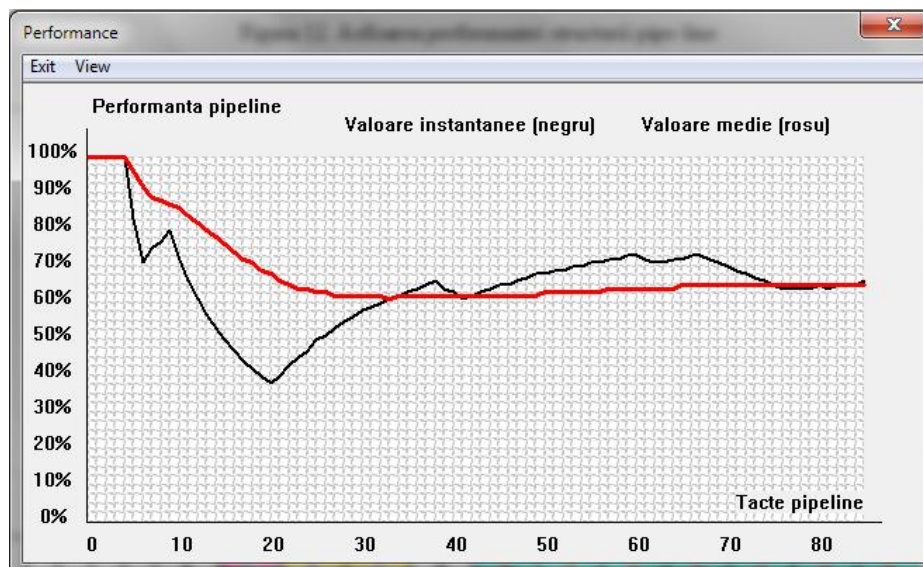


Figura 12. Afisarea performantei structurii pipe-line

#### 4. Meniurile ferestrei de afisare a tipurilor de instructiuni (figura 13).

Se afiseaza (cu comanda Update) tipurile de instructiuni. Pentru instructiunile dependente se trece in paranteza diferenta, in tacte pipe-line, dintre momentul de incarcare a instructiunii dependente si momentul de timp la care s-a incarat instructiunea care produce dependenta.

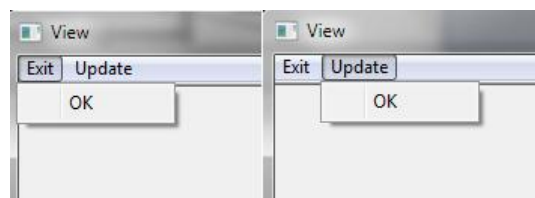


Figura 13. Meniurile ferestrei de afisare a performantelor a programului **simpl**



Un exemplu de afisare a tipurilor de instructiuni este prezentat in figura 14.

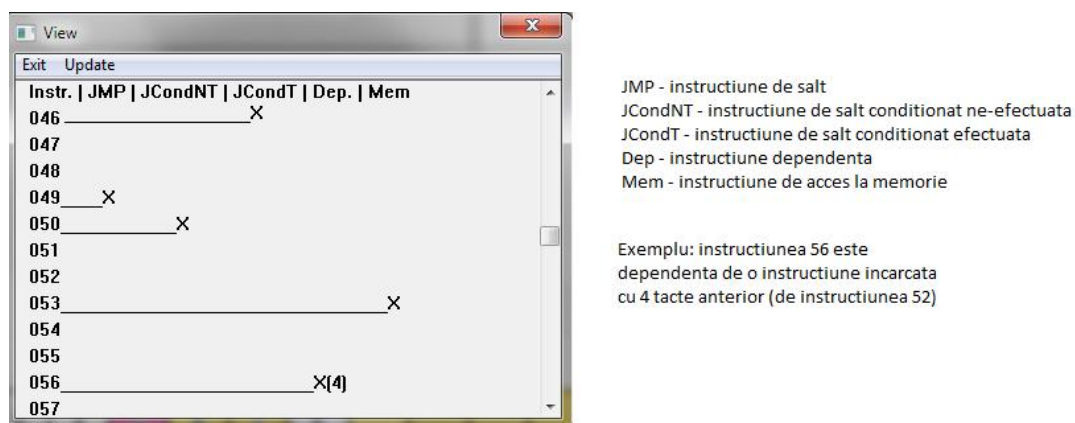


Figura 13. Afisarea tipurilor de instructiuni

Sumarul comenzilor este urmatorul (comanda Help):

### Fereastra principala (simpl)

Quit	- OK (iesire din program)
Clear	- OK (reinitializare program)
Models	- 1, 2, 3, 4, 5, 6 (alegere model structura pipe-line)
Results	- Performance View (Vizualizarea performantelor structurii pipe-line) - Instructions View (vizualizarea tipurilor de instructiune)
Help	- Help (afisare Help) - About (afisare versiune program)

### Fereastra de configurare (Configuration)

**Check box** (indica daca instructiunile de tipul corespunzator vor fi generate)

**Sliders** (indica probabilitatea de aparitie a tipului corespunzator de instructiuni)

Jump	- instructiuni de salt
Cond Jump	- instructiuni de salt conditionat
Taken %	- indica procentual cite instructiuni de salt conditionat se vor efectua
Dep	- instructiuni dependente
Mem	- instructiuni de lucru cu memoria

#### Butoane:

START	- porneste simularea executiei
NEXT	- afiseaza urmatorul ecran din diagrama spatiu-timp
PREV	- afiseaza ecranul anterior din diagrama spatiu-timp

### Fereastra de afisarea a tipului de instructiuni (View)

Exit	- OK (inchide fereastra)
Update	- OK (actualizeaza informatia)

### Fereastra de afisare a performantelor (performance)

Exit	- OK (inchide fereastra)
View	- Performance Graph (afisarea grafica a performantei structurii pipe-line) - Model Diagram (afisarea modelului structurii pipe-line)

O sesiune de lucru cu programul **simpl** presupune urmatoarele etape, dupa lansarea in executie a acestuia:

- alegere modelului de structura pipe-line (comanda **Models**, din fereastra principala).
- configurarea tipurilor de instructiuni si a probabilitatilor de aparitie ale acestora, conform cu modelul ales ( din feresatra de configurare).
- simularea propriu-zisa (butonul **START** din fereastra de configurare)

- d) vizualizarea diagramei spatio-temporale asociata executiei instructiunilor in structura pipe-line si identificarea efectelor fiecarei instructiuni (butoanele **NEXT** si **PREV** din fereastra de configurare)
- e) vizualizarea si analiza tipurilor de instructiuni (comanda **Results**→ **Instruction View**, din fereastra principala)
- f) vizualizarea grafica si analiza performantei structurii pipe-line (comanda **Results**→ **Performance View**, din fereastra principala)

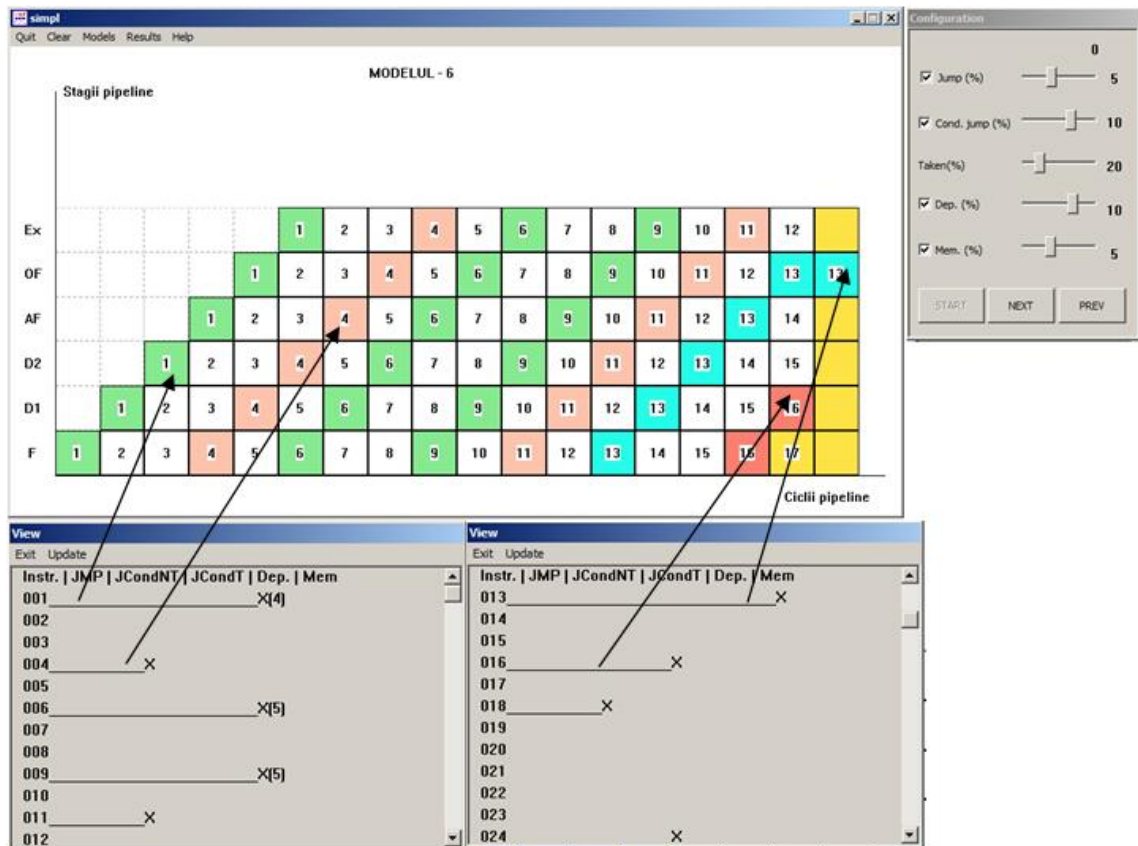
O noua sesiune de simulare incepe dupa comanda **Clear**, din fereastra principala.

Ferestrele asociate vizualizarii instructiunilor si performantei pot ramine deschise, de la o sesiune de lucru la alta, dar trebuie actualizate dupa fiecare simulare (comanda **Update**).

Etapele a) - f) sint ilustrate in continuare:

- a) s-a ales modelul 6
- b) s-au ales urmatoarele tipuri de instructiuni: salt – 5%, salt conditionat – 10%, salturi conditionate efectuate – 20%, instructiuni dependente – 10% si instructiuni de acces la memorie – 5%

Parcursul etapelor c) – f) este ilustrata in figura 14.



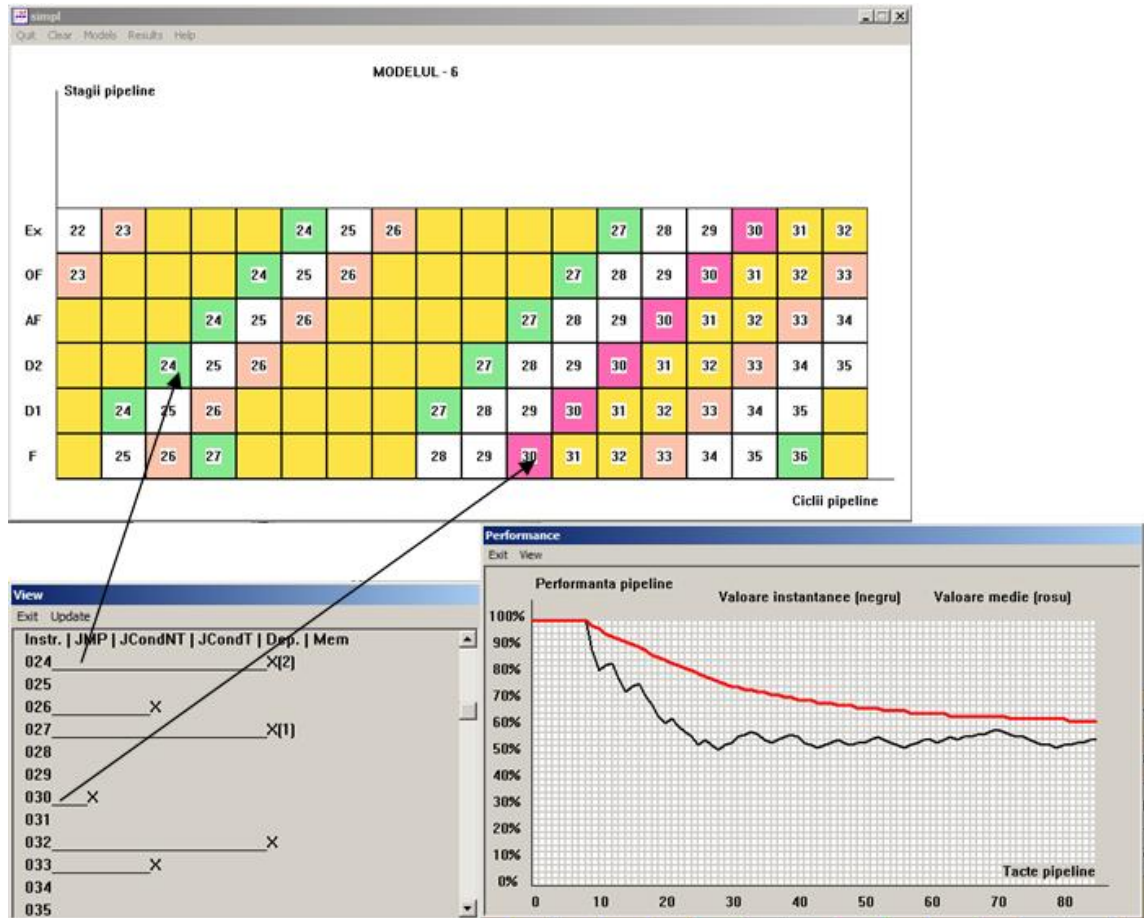



Figura 14. Exemplificarea utilizarii programului **simpl** .

Performanta este calculata astfel:  $P' = \frac{T_{ex}}{T_{ex} + T_{supl}}$ , unde  $T_{ex}$  este timpul necesar executiei instructiunilor si  $T_{supl}$  este timpul suplimentar (instructiuni anulate si *goluri*), exprimate in tacte pipe-line.

## 5. Desfasurarea lucrarii. Rezultate ale simularilor

1. Studiati efectul instructiunilor de salt, salt conditionat, dependente si de acces la memorie asupra functionarii unei structuri pipe-line (comparatie intre figurile 1, 2,3,4,5).
2. Studiati modul de lucru al simulatorului **simpl**.
3. Studiati, separat, efectul instructiunilor de salt, salt conditionat, dependente si de acces la memorie.
4. Comparati modelele 1 si 2, 3 si 4, respectiv 5 si 6. Pentru fiecare model calculati frecventa de aparitie a fiecarei instructiuni si performanta structurii pipe-line. Comparati performanta calculata cu performanta teoretica. Explicati diferentele.
5. Efectuati simularile din tabelul 2:

Nr. Crt.	Modelul	JMP	JC	Taken	Dep.	Mem.	Performanta
1	1	5%	10%	60%	-	-	
2	2	5%	10%	60%	-	-	
3	3	10%	10%	30%	-	-	
4	4	5%	10%	50%	-	-	
5	5	5%	5%	20%	5%	5%	
6	6	5%	5%	30%	5%	5%	

Discutati rezultatele.

6. Precizati metode pentru imbunatatirea performantelor in cazul aparitiei instructiunilor dependente.
7. Precizati metode pentru imbunatatirea performantelor in cazul aparitiei instructiunilor de salt conditionat.
8. Cum este influentata performanta in functie de numarul de stagii pipe-line ? Discutie dupa tipul instructiunii.