PROBLEME PROPUSE SPRE REZOLVARE

1. Sa se proiecteze un cache de instructiuni cuplat la un procesor superscalar (VLIW). Lungimea blocului din cache se considera egala cu rata de fetch a procesorului, în acest caz 4 instructiuni / bloc. Cache-ul va fi de tipul:

a. **semiasociativ**, cu 2 blocuri / set (2 – way set associative)
b. **complet asociativ** (full - associative)
c. cu **mapare directa** (direct mapped)

Ce se întâmpla daca în locul adresarii cu adrese fizice se considera adresare cu adresa virtuala?

2. a. De ce este dificila procesarea «Out of Order» a instructiunilor Load respectiv Store într-un program si de ce ar putea fi ea benefica?

b. Care dintre cele doua secvente de program s-ar putea procesa mai rapid pe un procesor superscalar cu executie «Out of Order» a instructiunilor? Justificati.

B1.

```
for i=1 to 100
        a[2i]=x[i];
        y[i]=a[i+1];
```

B2.

```
a[2]=x[1];
y[1]=a[2]+5;
for i=2 to 100
        a[2i]=x[i];
        y[i]=a[i+1]+5;
```

3. Se considera un procesor scalar pipeline, în 3 faze diferite de procesare (**IF**,**EX**,**WR**), fiecare faza necesitând un tact, cu urmatoarea semnificatie:

IF = aducere si decodificare a instructiunii
EX=selectie operanzi din setul de registri si executie
WR=înscriere rezultat în registrul destinatie

Se considera secventa de program:

1: R1<-- (R11)+(R12)
2: R1<-- (R1)+(R13)
3: R2 <-- (R3)+4
4: R2 <-- (R1)+(R2)
5: R1<-- (R14)+(R15)
6: R1<-- (R1)+(R16)

a. În câte impulsuri se executa secventa? (initial, structura «pipe» de procesare este «goala») Reorganizati aceasta secventa de program în vederea minimizarii timpului de executie (procesorul detine o infinitate de registri generali disponibili). În câte impulsuri de tact s-ar procesa în acest caz secventa ?
b. În câte tacte (minimum) s-ar procesa secventa daca procesorul ar putea executa simultan un numar nelimitat de instructiuni independente? Se considera ca procesorul poate aduce în acest caz, simultan, 6 instructiuni din memorie. Justificati.

4. Se considera o structura «pipe» de procesare a instructiunilor având un nivel de citire a operanzilor din setul de registri (RD), situat anterior unui nivel de scriere a rezultatului în setul de registri (WR). Careia dintre cele doua operatii (RD, WR) i se da prioritate în caz de conflict si în ce scop ?

5. Se considera ca 20% dintre instructiunile unui program determina ramificarea acestuia (salt efectiv). Care ar fi în acest caz rata de fetch (FR) posibila pentru un procesor superscalar (VLIW – Very Long Instruction Word) având resurse harware nelimitate si o predictie perfecta a branch-urilor (cunoastere anticipata a

adresei de salt) ? Este posibila o depasire a acestei limitari fundamentale ? Daca da, care ar fi *noua* limitare impusa parametrului FR prin solutia Dvs. ?

6. Un procesor superscalar poate lansa în executie simultan maxim N instructiuni ALU independente. Logica de detectie a posibilelor hazarduri RAW (Read After Write) între instructiunile ALU are costul «C» ($). Cât va costa logica de detectie daca s-ar dori ca sa se poata lansa simultan în executie maxim (N+1) instructiuni ALU independente ? (Se vor considera costurile ca fiind direct proportionale cu «complexitatea» logicii de detectie a hazardurilor RAW).

7. Relativ la o memorie cache cu mecanism de adresare tip «mapare directa», precizati valoarea de adevar a afirmatiilor de mai jos, cu justificarile de rigoare.

a. Rata de hit creste daca capacitatea memoriei creste;
b. O data de la o anumita locatie din memoria principala poate fi actualizata la orice adresa din cache;
c. Scrieri în cache au loc numai în ciclurile de scriere cu miss în cache;
d. Are o rata de hit net mai mare decât cea a unei memorii complet asociative si de aceeasi capacitate.

8. Se considera secventa de program RISC:

```
1:    ADD  R1, R11, R12
2:    ADD  R1, R1, R13
3:    ADD  R2, R3, R9
4:    ADD  R2, R1, R2
5:    ADD  R1, R14, R15
```

a. Reprezentati graful dependentelor de date (numai dependentele de tip RAW)

b. Stiind ca între 2 instructiuni dependente RAW si succesive e nevoie de o întârziere de 2 cicli, în câti cicli s-ar executa secventa ?

c. Reorganizati secventa în vederea unui timp minim de executie (nu se considera alte dependente decât cele de tip RAW).

9. a. Considerând un procesor RISC pe 5 nivele pipe (IF, ID, ALU, MEM, WB), fiecare durând un ciclu de tact, precizati câti cicli de întârziere («*branch delay slot*») impune o instructiune de salt care determina adresa de salt la finele nivelului ALU ?

b. De ce se prefera implementarea unor busuri si memorii cache separate pe instructiuni, respectiv date în cazul majoritatii procesoarelor RISC (pipeline) ?

c. De ce sunt considerate instructiunile CALL / RET mari consumatoare de timp în cazul procesoarelor CISC (ex. I-8086) ? Cum se evita acest consum de timp în cazul microprocesoarelor RISC ?

10. Considerând un microprocesor virtual pe 8 biti, având 16 biti de adrese, un registru A pe 8 biti, un registru PC si un registru index X, ambele pe 16 biti si ca opcode-ul oricarei instructiuni e codificat pe 1 octet, sa se determine numarul impulsurilor de tact necesare aducerii si executiei instructiunii «memoreaza A la adresa data de (X+deplasament)». Se considera ca instructiunea e codificata pe 3 octeti si ca orice procesare (operatie interna) consuma 2 tacte. Un ciclu de fetch opcode dureaza 6 tacte si orice alt ciclu extern dureaza 4 tacte.

11. Relativ la o arhitectura de memorie cache cu mapare directa se considera afirmatiile:

a. Nu permite accesul simultan la câmpul de date si respectiv «tag» al unui cuvânt accesat.

b. La un acces de scriere cu hit, se scrie în cache atât data de înscris cât si «tag-ul» aferent.

c. Rata de hit creste usor daca 2 sau mai multe blocuri din memoria principala - accesate alternativ de catre microprocesor - sunt mapate în acelasi bloc din cache.

Stabiliti valoarea de adevar a acestor afirmatii si justificati pe scurt raspunsul.

12. Ce corectie (doar una!) trebuie facuta în secventa de program asamblare pentru ca translatarea de mai jos sa fie corecta si de ce ? Initial, registrii R_i, R_k, R_l, R_j contin respectiv variabilele i, k, l, j. Primul registru dupa mnemonica este destinatie. (R_j+offset) semnifica operand în memorie la adresa data de (R_j+offset).

k = a[i+2]+5;
l = c[j+9] - k;

i1: ADD R_k, #2, R_i
i2: LOAD R_k, (R_k+0)
i3: ADD R_k, R_k, #5
i4: ADD R_l, #9, R_j
i5: LOAD R_l, (R_j+0)
i6: SUB R_l, R_l, R_k

13. Se considera un microsistem realizat în jurul unui microprocesor care ar accepta o frecventa maxima a tactului de 20 MHZ. Regenerarea memoriei DRAM se face <u>în mod transparent</u> pentru microprocesor. Procesul de regenerare dureaza 250 ns. Orice ciclu extern al procesorului dureaza 3 perioade de tact. Poate functiona în aceste conditii microprocesorul la frecventa maxima admisa? Justificati.

14. Explicati concret rolul fiecareia dintre fazele de procesare (ALU, MEM, WB) în cazul instructiunilor:

a. STORE <u>R5</u>, (R9)06h;
sursa

b. LOAD R7, (R8)F3h;
dest

c. AND R5, R7, R8.
dest

15. Se considera un procesor scalar pipeline, în 3 faze diferite de procesare (IF, EX, WR), fiecare faza necesitând un tact, astfel:

IF = fetch instructiune si decodificare;
EX = selectie operanzi din setul de registri si executie;
WB = înscriere rezultat în registrul destinatie.

a. În câte impulsuri de tact se executa secventa de program de mai jos ?
b. Reorganizati aceasta secventa în vederea minimizarii timpului de executie.

1: ADD R3, R2, #2
2: ADD R1, R9, R10
3: ADD R1, R1, R3
4: ADD R2, R3, #4
5: ADD R2, R1, R2
6: STORE R3, (R1)2

16. Un procesor pe 32 biti la 50 MHZ, lucreaza cu 3 dispozitive periferice prin interogare. Operatia de interogare a starii unui dispozitiv periferic necesita 100 de tacte. Se mai stie ca:

a. interfata cu mouse-ul trebuie interogata de 30 de ori / s pentru a fi siguri ca nu se pierde nici o «miscare» a utilizatorului.
b. floppy - discul transfera date spre procesor în unitati de 16 biti si are o rata de transfer de 50 ko / s.
c. hard - discul transfera date spre procesor în unitati de 32 biti si are o rata de transfer de 2 Mo / s.

Determinati în [%], fractiunea din timpul total al procesorului, necesara interogarii starii fiecarui periferic. Comentati.

17. Se considera instructiunea (I-8086):

3000h: MOV [BX]0F3h, AX

EA dest sursa

a. La ce adresa fizica se aduce opcode-ul instructiunii ?
b. La ce adrese fizice se scriu registrii AL, respectivAH ?

Înainte de executia instructiunii avem: CS = 1D00h
BX = 1B00h
SS = 2000h
DS = DF00h.

18. Se considera instructiunea (I-8086):

2000h: PUSHAX

EA

a. De la ce adresa se aduce instructiunea ?
b. La ce adrese fizice se scriu registrii AL, respectiv AH ?

Se stie ca înainte de executia instructiunii PUSH avem: CS = AE00h
SS = 1FF0h
SP = 001Eh
DS = 1F20h.

19. Un automat de regenerare al memoriilor DRAM declanseaza efectiv procesul de regenerare daca sunt simultan îndeplinite conditiile:

a. activare semnal cerere refresh (CREF).

b. microprocesorul nu lucreaza momentan cu memoria. Având în vedere dezideratul regenerarii «transparente» (sa nu fie simtita de catre microprocesor), ar functiona corect automatul ? Comentati si sugerati o eventuala corectie.

20. Consideram 3 memorii cache care contin 4 blocuri a câte un cuvânt / bloc. Una este complet asociativa, alta semiasociativa cu 2 seturi a câte 2 cuvinte si ultima cu mapare directa. Stiind ca se foloseste un algoritm de evacuare de tip LRU, determinati numarul de accese cu HIT pentru fiecare dintre cele 3 memorii, considerând ca procesorul citeste succesiv de la adresele 0, 8, 0, 6, 8, 10, 8 (primul acces la o anumita adresa va fi cu MISS).

21. Se considera secventa de program RISC:

```
1:    ADD  R3, R2, #2
2:    ADD  R1, R9, R10
3:    ADD  R1, R1, R3
4:    ADD  R2, R3, #4
5:    ADD  R2, R1, R2
```

Între doua instructiuni dependente RAW si succesive în procesare, e nevoie de o întârziere de 1 ciclu de tact.

a. În câti cicli de tact se executa secventa initiala ?

b. În câti cicli de tact se executa secventa reorganizata aceasta secventa în vederea unui timp minim de procesare ?

22. Se considera o unitate de disc având rata de transfer de 25×10^4 biti/s, cuplata la un microsistem. Considerând ca transferul între dispozitivul periferic si CPU se face prin întrerupere la fiecare octet, în mod sincron, ca timpul scurs între aparitia întreruperii si intrarea în rutina de tratare este de 2μs si ca rutina de tratare dureaza 10μs, sa se calculeze timpul pe care CPU îl are disponibil între 2 transferuri succesive de octeti.

23. a. Daca rata de hit în cache ar fi de 100%, o instructiune s-ar procesa în 8.5 cicli de tact. Sa se exprime în [%] scaderea medie de performanta daca rata de hit devine 89%, orice acces la memoria principala se desfasoara pe 6 tacte si ca orice instructiune face 3 referinte la memorie.

b. De ce e avantajoasa implementarea unei pagini de capacitate «mare» într-un sistem de memorie virtuala ? De ce e dezavantajoasa aceasta implementare ? Pe ce baza ar trebui facuta alegerea capacitatii paginii ?

24. Se considera un procesor scalar pipeline, în 3 faze diferite de procesare (IF, EX, WR), fiecare faza necesitând un tact, astfel:

IF = fetch instructiune si decodificare;
EX = selectie operanzi din setul de registri si executie;
WB = înscriere rezultat în registrul destinatie.

a. În câte impulsuri de tact se executa secventa de program de mai jos ?
b. Reorganizati aceasta secventa în vederea minimizarii timpului de executie (se considera ca procesorul detine o infinitate de registri generali).

1: R1 ← (R11) + (R12)
2: R1 ← (R1) + (R13)
3: R2 ← (R3) + 4
4: R2 ← (R1) + (R2)

5: R1 ← (R14) + (R15)

6: R1 ← (R1) + (R16)

25. Se considera un microprocesor RISC cu o structura «pipe» de procesare a instructiunii, având vectorul de coliziune atasat 01011. Sa se determine rata teoretica optima de procesare a instructiunii pentru acest procesor [instr/ciclu].

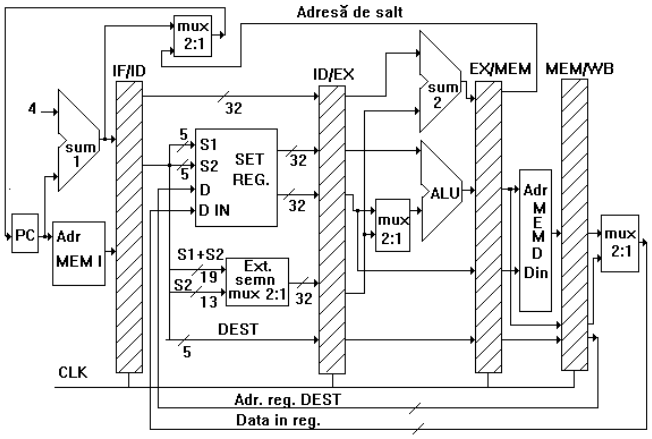
26. De ce implementarea algoritmului lui R. TOMASULO într-o arhitectura superscalara ar putea reduce din «presiunea» la citire asupra seturilor de registri generali ? Gasiti vreo similitudine în acest sens, între un CPU superscalar având implementat acest algoritm si un CPU de tip TTA (Transport Triggered Architecture) ?

27. De ce considerati o instructiune de tip RETURN este mai dificil de predictionat printr-un predictor hardware ? Puteti sugera vreo solutie în vederea eliminarii acestei dificultati ? În ce consta noutatea «principiala» a predictoarelor corelate pe doua nivele ?

28. Cum credeti ca s-ar putea masura printr-un simulator de tip «*trace driven*», câstigul de performanta introdus de tehnicile de paralelizare a buclelor de program (ex. «Loop Unrolling», «Software Pipelining», etc.)

29. Cum explicati posibilitatea interblocarii proceselor în cadrul limbajului OCCAM ? Ce întelegeti prin «sectiune critica de program» în cadrul unui sistem multimicro ? Care este «mesajul» transmis de «legea lui AMDAHL» pentru sistemele paralele de calcul ?

30. Se considera structura hardware a unui microprocesor RISC, precum în figura de mai jos.



Raspundeti la urmatoarele întrebari.

a. Ce tip de instructiuni activeaza sumatorul «sum 2» si în ce scop ?

b. Într-un tact, la setul de registri pot fi necesare 2 operatii simultane: citire (nivelul RD din pipe), respectiv scriere (nivelul WB din pipe). Carei operatii i se da prioritate si în ce scop ?

c. Ce rol are unitatea ALU în cazul unei instructiuni de tip LOAD ?

d. Ce informatie se memoreaza în latch-ul EX/MEM în cazul instructiunii: ST (R7)05, R2 si de unde provine fiecare informatie ?

31. Se considera secventa de program RISC:

```
1:    SUB  R7, R2, R12
2:    ADD  R1, R9, R10
3:    ADD  R1, R1, R7
4:    SUB  R2, R7, R12
5:    ADD  R2, R1, R2
6:    ADD  R1, R6, R8
7:    ADD  R1, R1, R7
8:    SUB  R1, R1, R12
9:    LD   R1, (R1)2
10:   LD   R4, (R4)6
11:   ADD  R1, R4, R1
```

```
12:   ADD  R1, R1, R2
13:   ST   R1, (R4)16
14:   ST   R7, (R1)16
```

a. Sa se construiasca graful dependentelor / precedentelor de date aferent acestei secvente. Cu exceptia LOAD-urilor care au latenta de 2 cicli, restul instructiunilor au latenta de 1 ciclu.
b. În baza algoritmului LIST SCHEDULING, sa se determine modul optim de executie al acestei secvente (nr. cicli), pentru un procesor superscalar având 2 unitati ADD, 1 unitate SUB si 1 unitate LOAD/STORE. Unitatea pentru LOAD este nepipeline-izata.

În limbaj de asamblare MIPS prezentati solutiile urmatoarelor probleme:

32. Scrieti un program folosind recursivitatea, care citeste caractere de la tastatura si le afiseaza în ordine inversa.
Obs. Nu se lucreaza cu siruri, nu se cunoaste numarul de caractere citite, sfârsitul sirului va fi dat de citirea caracterului ‘0’.
Modificati programul astfel încât ‘0’ – care marcheaza sfarsitul sirului, sa nu fie tiparit.

33. Scrieti un program folosind recursivitatea, care citeste de la tastatura doua numere întregi pozitive si afiseaza cel mai mare divizor comun si cel mai mic multiplu comun al celor doua numere.

34. Scrieti un program recursiv care rezolva problema *Turnurilor din Hanoi* pentru *n* discuri (n – parametru citit de la tastatura). Enuntul problemei este urmatorul:
Se dau trei tije simbolizate prin A, B si C. Pe tija A se gasesc n discuri de diametre diferite, asezate în ordine descrescatoare a diametrelor privite de jos

în sus. Se cere sa se mute discurile de pe tija A pe tija B, folosind tija C ca tija de manevra, respectându-se urmatoarele reguli:

- *La fiecare pas se muta un singur disc.*
- *Nu este permis sa se aseze un disc cu diametrul mai mare peste un disc cu diametrul mai mic.*

35. Realizati un program care citeste de la tastatura doua numere naturale *n* si *k* (n>k) si calculeaza si afiseaza pe consola valorile urmatoare:

$$C_n^k \text{ si } A_n^k$$

36. Sa se citeasca un sir de numere întregi de la tastatura, a carui dimensiune este citita tot de la tastatura. Sortati sirul prin metoda "*bubblesort*", memorati succesiv datele la adresa 0x10012000 si afisati sirul sortat pe consola.

37. Scrieti un program care afiseaza primele *n* perechi de numere prime impare consecutive (n - numar impar citit de la tastatura). Exemplu: (3,5), (5,7), etc.

38. Scrieti un program, în limbaj de asamblare DLX, care citeste *n* numere întregi de la tastatura prin intermediul modulului **Input.s** (vezi lucrarea *Investigatii Arhitecturale Utilizând Simulatorul DLX*) si calculeaza maximul, minimul si suma numerelor si le depune succesiv în memoria DLX la adresa 0x1500.

39. Proiectati automatul de control cache, într-un sistem multimicroprocesor simetric pe bus comun, având în vedere ca un bloc din cache se poate afla într-una din starile: PARTAJAT, EXCLUSIV sau INVALID.

40. Se considera un sistem multimicroprocesor (SMM) cu N microprocesoare legate printr-o retea de interconectare (RIC) la N module fizice de memorie. În

ce ar consta si ce ar permite o RIC cu largime de banda maxima ? Dar una cu largime de banda minima ?

41. Într-un SMM cu memorie centrala partajata si în care fiecare procesor detine un cache propriu, un procesor initiaza o scriere cu MISS într-un anumit bloc aflat în starea "partajat" ("shared"). Precizati procesele succesive care au loc în urma acestei operatii.

42. Într-un SMM un procesor initiaza o citire cu MISS la un bloc invalid în cache-ul propriu, blocul aflându-se în copia exclusiva într-alt procesor. Precizati concret procesele succesive care au loc în urma acestei operatii.

Bibliografie

[1] **Vintan L.** – *Metode de evaluare si optimizare în arhitecturile paralele de tip I.L.P.*, Editura Universitatii «Lucian Blaga» Sibiu, Sibiu, 1997.

[2] **Yeh T. Y., Patt Y. N.** – *Alternative Implementation of Two-Level Branch Prediction*, Department of EECS, The University of Michigan, 1992.

[3] **Hennessy J., Patterson D.** – *Computer Architecture*: *A Quantitative Approach*, Morgan Kaufmann Publishers, Second Edition, 1996.

[4] **Knuth D. E.** – *Tratat de programarea calculatoarelor*, vol. I, IV, Editura Tehnica, Bucuresti, 1974.