Blocuri multiple de memorie.

Cresterea capacitatii memoriei principale/interne, indiferent de tehnologia de implementare, conduce la cresterea timpului de acces, ceea ce poate avea drept consecinta intrarea procesorului in stari inactive, de asteptare.

Adesea, exista situatii in care se cunosc anticipat adresele de acces la memorie. Pornind de la aceasta constatare, memoria interna poate fi organizata pe blocuri, cu porturi proprii de acces, adresele fiind intretesute intre blocuri.

In figura de mai jos este prezentata o memorie cu capacitatea de 256 kcuvinte x 32 biti, structurata sub forma a 4 blocuri cu adrese intretesute. Organizarea adreselor este data in hexa, la nivelul fiecarui bloc.

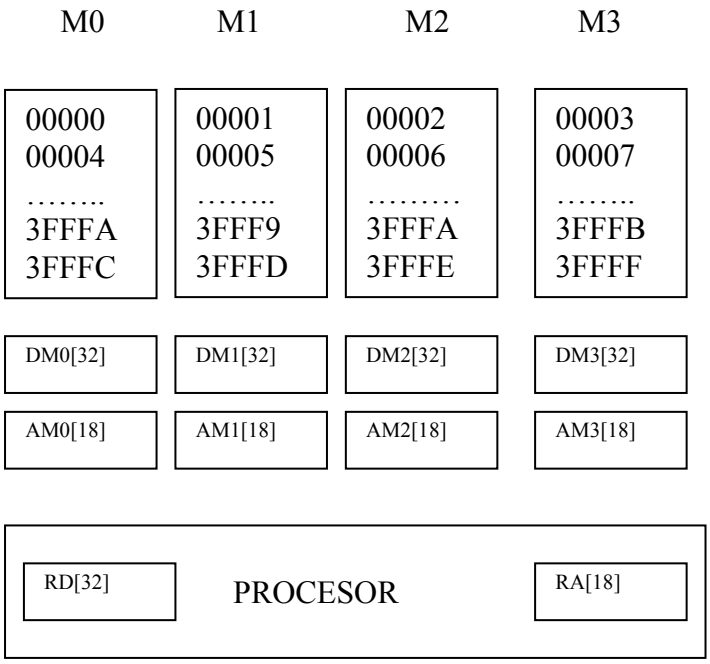

Fig. Memorie organizata in blocuri cu adrese intretesute.

Fiecare bloc va contine 2^{16} cuvunte de cite 32 de biti. In cazul unui procesor sofisticat este avantajos sa se citeasca simultan continutul mai multor adrese din memorie. Aceste cuvinte/informatii vor fi transferate catre o memorie rapida. In cadrul organizarii din figura de mai sus se pot extrage maximum 4 cuvinte simultan.

In continuare se va studia organizarea unei memorii pe blocuri, cu adrese intretesute si puncte multiple de acces.

Acest gen de organizare ofera avantajul ca asigura accesul simultan la mai multe blocuri, prin porturile specifice de acces.

La aceste porturi se pot conecta procesoare, in sistemele multiprocesor, sau echipamente de I/E, care lucreaza in regim de acces direct la memorie, ceea ce permite executia unor programe in paralel cu operatiile de I/E.

In continuare se va proiecta secventa de comanda pentru o memorie organizata pe blocuri cu porturi de acces. Fiecare bloc i de memorie poseda un registru de date DMi si un registru de adrese AMi. De asmenea, fiecare port de acces j este prevazut cu un registru de comunicatii de date CDj si cu un registru de comunicatii de adrese CAj.

Oricare din cele patru blocuri de memorie poate comunica cu oricare din cele patru porturi de acces. Unitatea de comanda a memoriei si procesorul/procesoarele sunt controlate cu acelasi semnal de ceas. Astfel, un procesor poate introduce un cuvant intr-un registru de comunicatie CAj asociat, intr-o perioada de ceas, iar unitatea de control a memoriei il poate prelua in perioada imediat urmatoare. Printr-un registru CDj transferul poate avea loc in ambele sensuri.

Cu fiecare registru CAj sunt asociate doua bistabile de control Cj si Sj.

In cazul in care Cj = 1 se solicita o operatie de citire, iar in cazul in care Sj = 1 - una de scriere (un cuvant aflat in CDj urmeaza sa fie stocat in memorie, la adresa specificata in CAj). In cazul in care Sj = Cj = 0 registrele CAj si CDj sunt accesibile unitatii de comanda a portului de acces, ceea ce specifica unui eventual procesor activ posibilitatea de a trece la operatia urmatoare.

Transferul unidirectional al adreselor din registrele CAj in registrele AMi (i,j = 0,....,3) se realizeaza prin intermediul magistralei MAGA. Transferul datelor intre registrele CDi si DMj, indiferent de sens, se efectueaza cu ajutorul magistralei MAGD.

Procesul de citire/scriere din/in memorie poate necesita mai multe perioade de ceas. Astfel, patru blocuri de memorie pot fi ocupate simultan cu diverse operatii, solicitate de diferite porturi de acces.

Pentru a controla corect fluxul de date este necesara introducerea, la fiecare bloc de memorie, a unor registre de cate doi biti: DSi[2] si STi[2], care specifica destinatia

(portul de acces implicat), in cazul operatiei de citire, si respectiv starea blocului de memorie. Codificarea starii este urmatoarea:

- STi = 00 blocul i este inactiv;
- STi = 01 blocul i executa o operatie de scriere sau de citire;
- STi =10 blocul a terminat o operatie de citire si datele asteapta sa fie preluate din registrul DMi.

In figura de mai jos se prezinta organizarea unei memorii constand in patru blocuri M0,..,M3 si patru porturi de acces PA0,..,PA3, care comunica intre ele prin magistralele de date si adrese MAGD si,respectiv - MAGA.

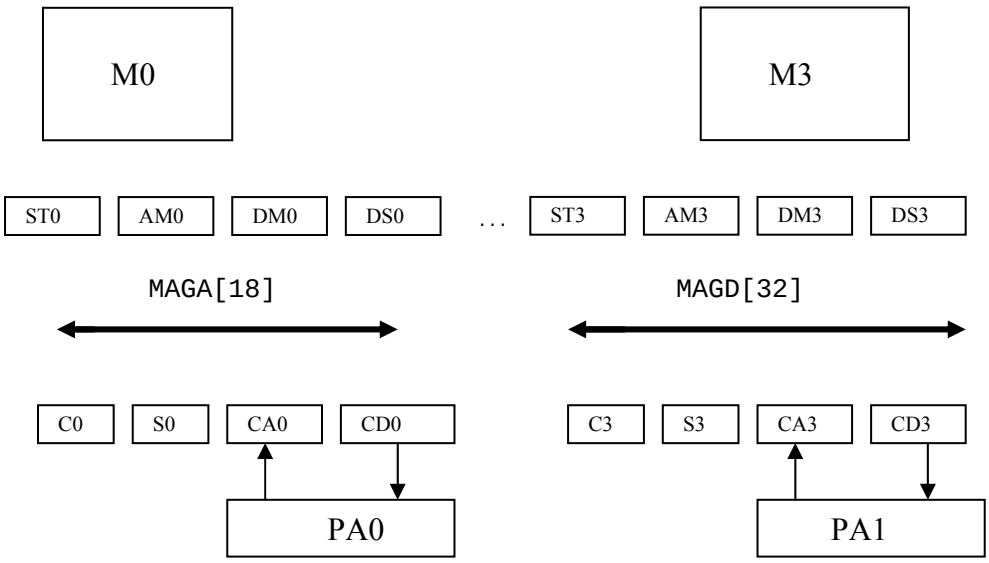

Fig. Organizarea unei memorii cu patru blocuri cu adreseintretesute si patru porturi de acces.

Considerind o memorie cu capacitatea de 256 kcuvinte, de cite 32 de biti fiecare, organizata in patru blocuri cu adrese intretesute, se poate stabili o structura a resurselor, la nivel de bloc si de port de acces, ca in figura urmatoare, unde se considera, spre exemplificare, blocul de memorie M0 si portul de acces PA2.

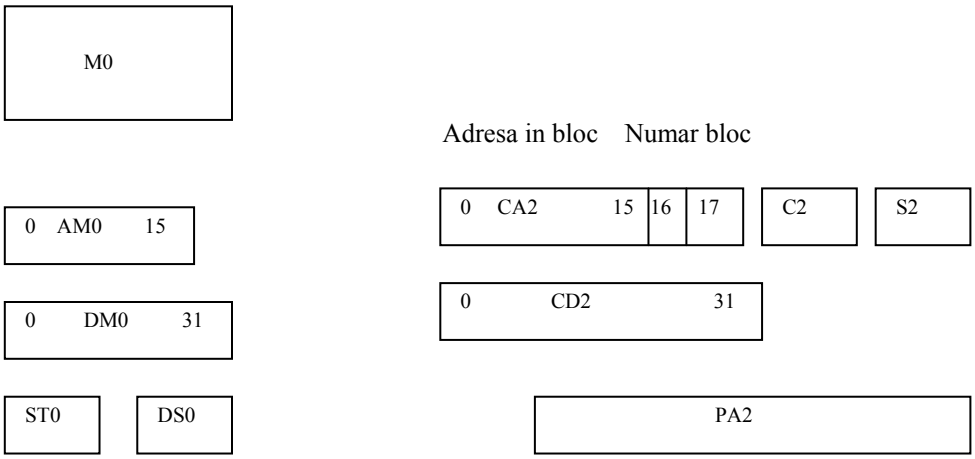

Fig. Resursele unui bloc de memorie si ale unui port de acces.

Se poate observa ca un registru CAi poseda 18 biti dintre care ultimii doi, mai putin semnificativi, reprezinta numarul blocului, iar cei mai semnificativi 16 biti - adresa in bloc. Aceasta corespunde adresarii intretesute. In cazul cand primii doi biti mai semnificativi ai registrului CAi ar fi codificat numarul blocului, adresarea nu ar mai fi fost intretesuta. Distributia adreselor intretesute a fost prezentata mai sus.

Un procesor poate furniza la un moment dat patru adrese consecutive, pentru cele patru blocuri de memorie, numai in doua perioade de tact pentru fiecare daca porturile de acces sunt inactive. Se considera ca daca un port de acces solicita ca patru cuvinte consecutive sa fie citite din memorie portul de acces in cauza va fi pregatit sa receptioneze cuvintele pe masura ce acestea devin disponibile. In caz contrar cuvintele s-ar putea sa fie transferate in registrul CA dat, in alta ordine decat cea in care au fost solicitate. Controlul porturilor de acces trebuie sa tina seama de cererile de date pe care acestea le-au prezentat. In particular, un procesor nu trebuie sa incerce sa scrie intr-un bloc de memorie, daca o cerere de citire nu a fost satisfacuta.

Controlul unitatii va fi abordat sub aspectul traficului de informatii pe magistralele de adrese si de date. Pentru a evita reducerea de viteza a procesului de transfer al informatiei trebuie sa se permita transferuri simultane pe cele doua magistrale.

Secventa de control este constituita din doua parti (vezi organigrama). Prima parte va accepta o noua cerere de citire si simultan va putea livra datele de la o operatie de citire

deja terminata. Partea a doua trateaza cererile de scriere. Ramura din dreapta a primei parti specifica modul de interogare al fiecarui port de acces in scopul detectarii unei eventuale cereri de citire.

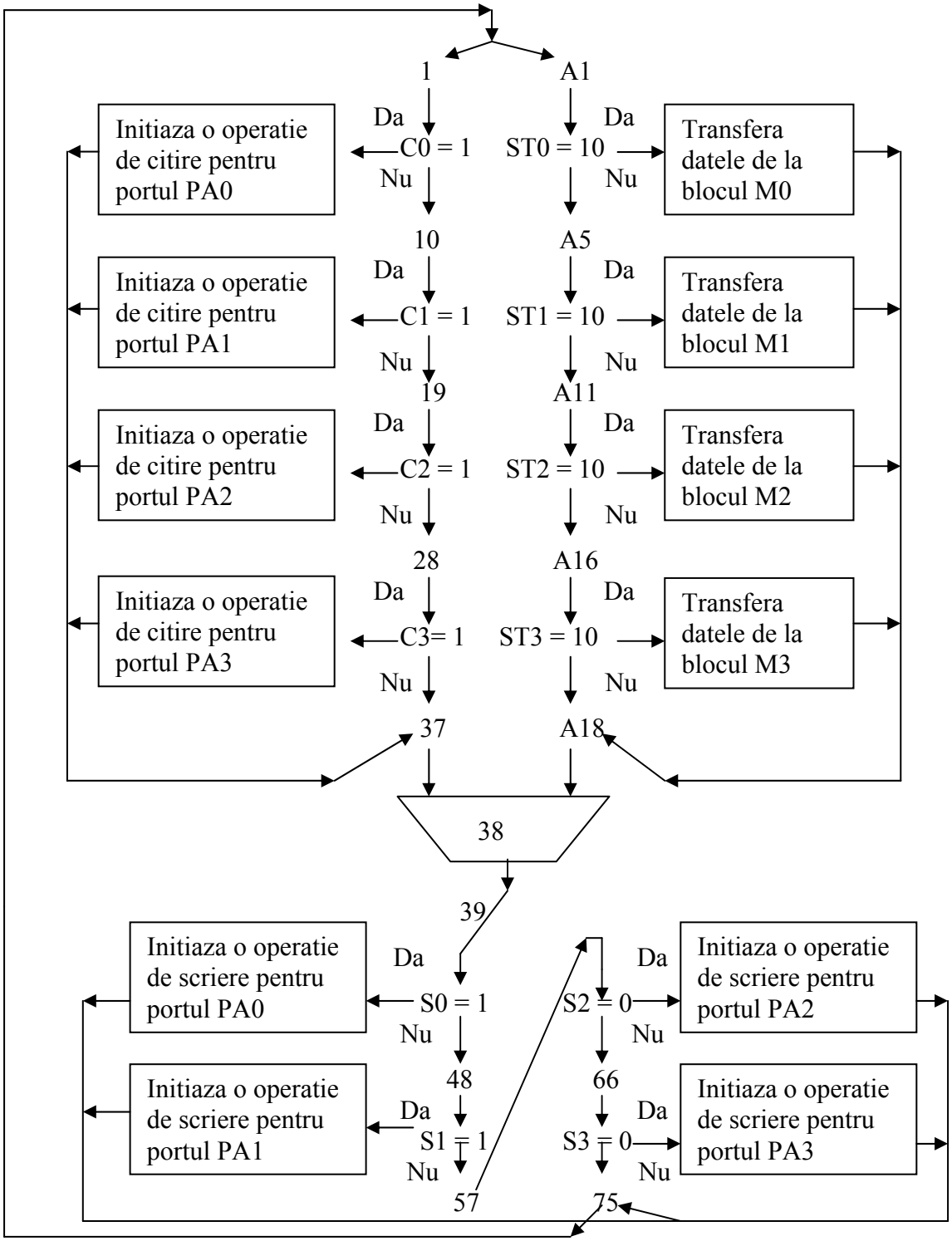

Fig. Secventa de control pentru memorie

Prima cerere intalnita este implementata prin transferul adresei catre blocul de memorie selectat, folosind MAGA. In acelasi timp un cuvant poate fi returnat de la un bloc de memorie catre un port de acces, prin magistrala MAGD - ramura din dreapta.

Dupa tratarea a cel mult unei cereri de citire si a unui transfer de date, comanda converge pentru a permite cautarea unei eventuale cereri de scriere. Secventa de scriere va fi implementata numai pentru o singura cerere - prima intalnita. Cererile de scriere sunt tratate separat, deoarece in cazul lor se transmit simultan la blocul de memorie atat adresa, cat si data. In organigrama a fost incorporat un sistem de prioritati pe baza caruia porturile de acces si blocurile de memorie cu indici mai mici au prioritati mai mari. Intervalele de timp pentru citirea si scrierea in blocurile de memorie sunt mult mai mari decat intervalul de timp necesar pentru efectuarea mai multor treceri prin secventa de comanda a magistralelor. Fiecare transfer de adresa sau de data intra-un procesor si un bloc de memorie necesita numai o perioada de tact.

In continuare se prezinta secventa detaliata pentru initializarea unei cereri de citire pentru PA0.

1. $\rightarrow$(C0 $\cap$DCD(CA016:17),$\overline{C0}$)/(2,4,6,8,10)
2. $\rightarrow$(ST00 $\cup$ ST01)/(10); Fara Intarziere
3. DS0,ST0,C0 $\leftarrow$0,0,0,1,0; MAGA = CA0;
   AM0 $\leftarrow$MAGA; cm0 = 1;
   $\rightarrow$(37); Fara Intarziere
4. $\rightarrow$(ST10 $\cup$ST11)/(10); Fara Intarziere
5. DS1,ST1,C0 $\leftarrow$ 0,0,0,1,0; MAGA = CA0;
   AM1 $\leftarrow$ MAGA; cm1 = 1;
   $\rightarrow$(37); Fara Intarziere
6. $\rightarrow$(ST20 $\cup$ST21)/(10); Fara Intarziere
7. DS2,ST2,C0 $\leftarrow$ 0,0,0,1,0; MAGA = CA0;
   AM2 $\leftarrow$ MAGA; cm2 = 1;
   $\rightarrow$(37); Fara Intarziere
8. $\rightarrow$(ST30 $\cup$ ST31)/(10); Fara Intarziere
9. DS3,ST3,C0 $\leftarrow$ 0,0,0,1,0; MAGA = CA0;
   AM3 $\leftarrow$ MAGA; cm3 = 1;

→(37); Fara Intarziere

Secventa detaliata pentru initializarea unei cereri de citire pentru portul PA1.

10. →(C1 ∩ DCD(CA116:17),$\overline{C1}$)/(11,13,15,17,19,)

11. →(ST00 ∪ ST11)/(19); fara Intarziere

12. DS0,ST0,C1 ← 0,10,1,0; MAGA = CA1;

    AM0 ← MAGA; cm0 = 1;

    →(37); Fara Intarziere

13. →(ST10 ∪ ST11)/(19); Fara Intarziere

14. DS1,ST1,C1 ← 0,1,0,1,0; MAGA = CA1;

    AM1 ← MAGA; cm1 = 1;

    →(37); Fara Intarziere

15. →(ST20 ∪ ST21)/(19); Fara Intarziere

16. DS2,ST2,C1 ← 0,1,0,1,0; MAGA = CA1;

    AM2 ← MAGA; cm2 = 1;

    →(37); Fara Intarziere

17. →(ST30 ∪ ST31)/(19); Fara Intarziere

18. DS3,ST3,C1 ← 0,1,0,1,0; MAGA = CA1;

    AM3 ← MAGA; cm3 = 1;

    →(37); fara Intarziere

.....

La pasul 1 se constata efectuarea unui salt cu patru ramificatii in functie de blocul de memorie adresat de CA0. Daca blocul in cauza este inactiv, ceee ce se testeaza la pasul 2, pasul 3 va produce modificarile necesare de stare si va conecta magistrala MAGA la CA0, pentru a se efectua transferul adresei la AM0. La pasul 3 are loc transferul adresei si lansarea unei cereri de citire catre M0. Pentru cazurile in care sunt adresate blocurile de memorie M1, M2, M3 sunt necesare trei secvente separate a cate doi pasi: 4 si 5, 6 si 7, 8 si 9. Pasii 10 - 37 solutioneaza cererile de citire emise de porturile PA1, PA2, PA3 catre blocurile de memorie M0, M1, M2, M3.

Secventa concurenta pentru returnarea datelor consta din pasiiA1 - A5, care trateaza transferurile de la M0, la PA0,...,PA3. Secvente similare (pasii A6,...,A20) vor fi elaborate si pentru blocurile de memorie M1,M2, M3.

A1. →(ST00 ∩ DCD(DS0),$\overline{ST00}$)/(A2,A3,A4,A5,A6)

A2. ST0 ← 0,0; MAGD = DM0;

CD0 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A3. ST0 ← 0,0; MAGD = DM0;

CD1 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A4. ST0 ← 0,0; MAGD = DM0;

CD2 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A5. ST0 ← 0,0; MAGD = DM0;

CD3 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A6. →(ST10 ∩ DCD(DS1),$\overline{ST10}$)/(A7,A8,A9,A10,A11)

A7. ST1 ← 0,0; MAGD = DM1;

CD0 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A8. ST1 ← 0,0; MAGD = DM1;

CD1 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A9. ST1 ← 0,0; MAGD = DM1;

CD2 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A10. ST1 ← 0,0; MAGD = DM1;

CD3 ← MAGD; gata = 1;

→(A21); Fara Intarziere

A11. .....

Pasul A1 genereaza un salt, in functie de destinatia datelor, specificata in DS0.

Pasul A2 stabileste conectarea magistralei; inactiveaza starea blocului de memorie; transmite cuvantul solicitat catre CD0 si semnaleaza acest fapt prin semnalul gata = 1. De la pasii 37 si A21 controlul este directionat catre pasul de convergenta 38, de la care incepe secventa de servire a cererilor de scriere.

A18.→(38); Fara Intarziere.

37. →(38); fara Intarziere

38. CONVERGE(A18,37)

39. →(S0 ∩ DCD(CA016:17),$\overline{S0}$)/(40,42,44,46,48)

40. →(ST00 ∪ ST01)/(48); Fara Intarziere

41. ST0,S0 ← 0,1,0; MAGA = CA0;
    MAGD = CD0; AM0 ← MAGA; DM0 ← MAGD; sm0 = 1;
    →(75)

42. →(ST10 ∪ST11)/(48); Fara Intarziere

43. ST1,S0 ← 0,1,0; MAGA = CA0;
    MAGD = CD0; AM1 ← MAGA; DM1 ← MAGD; sm1 = 1;
    →(75)

44. →(ST20 ∪ ST21)/(48); Fara Intarziere

45. ST2,S0 ← 0,1,0; MAGA = CA
    MAGD = CD0; AM2 ← MAGA; DM2 ← MAGD; sm2 = 1;
    →(75)

46. →(ST30 ∪ ST31)/(48); Fara Intarziere

47. ST3,S0 ← 0,1,0; MAGA = CA0;
    MAGD = CD0; AM3 ← MAGA; DM3 ← MAGD; sm3 = 1;
    →(75)

48. .....

75. DIVERGE(1,A1)

Pasii 39 - 47 trateaza cazurile in care sunt adresate, pentru scriere, blocurile de memorie M0, M1, M2, M3 de catre portul de acces PA0. Pasii 48 - 74 trateaza cazurile de scriere generate de porturile de acces PA1, PA2, PA3.

Secventa de control prezentata nu este in mod necesar cea mai rapida sau eficienta. O trecere prin secventa va necesita cel mult patru perioade de ceas. Daca ciclul memoriei este de 50 - 100 de ori mai lung decat perioada ceasului, intervalul de parcurgere de patru perioade nu este semnificativ. Daca ciclul memorie se reduce la 10 perioade de ceas, atunci parcurgerea va introduce o Intarziere importanta.

In principiu se poate recurge la inlocuirea magistralelor prin circuite care sa asigure transferuri directe. O solutie mai eficienta ar consta in eliminarea completa a transferurilor si directionarea combinationala a adreselor. Aceasta va duce la o retea combinationala extrem de extinsa.