PROIECTAREA UNEI STRUCTURI CU MICROPROCESOR

În acest capitol ne propunem să proiectăm o structură simplă cu microprocesor. Deoarece la ora actuală marea majoritate a sistemelor de pe piaţă (în special cea românească) au la bază familia Intel de procesoare, vom porni în analiza noastră de la microprocesorul 8086. Structura pe care o dezvoltăm în jurul acestui procesor, împreună cu resursele necesare funcţionării ca modul independent, conţine o serie de restricţii de proiectare grupate sub forma unor specificaţii de proiectare. Structura proiectată are un pronunţat caracter didactic.

1.1. SPECIFICAŢIILE DE PROIECTARE

Specificaţiile de proiectare ale structurii cu microprocesor sunt următoarele:

1. Unitatea centrală de prelucrare este organizată în jurul microprocesorului 8086/8088.

2. Unitatea centrală de prelucrare acceptă coprocesor matematic (în modul maxim).

3. Memoria principală este structurată din punct de vedere funcţional în:

– Memorie RAM dinamică având capacitatea de 256/512 K, realizată cu capsule 64Kx1. Memoria RAM este prevazută cu bit de paritate la nivel de octet.

– Memorie EPROM având capacitatea de 8÷64K. Se constituie în trei bancuri: două pentru BIOS şi unul pentru eventualele programe de test necesare depanării. Capsulele folosite vor fi 2716(2Kx8), 2732(4Kx8), 2764(8Kx8), 27128(16Kx8).

4. Interfaţa cu tastatura, care asigură o cuplare serială a tastaturii în condiţiile în care logica de pe placa de bază transformă informaţia în paralel, furnizând către microprocesor informaţia paralelă de la tastatură. Interfaţa furnizeză din punct de vedere logic coduri de scanare, unul la apăsarea tastei, iar celalalt la ridicarea tastei. Avantajul utilizării codurilor de scanare este că pe lângă informaţia "tastă apăsată/neapăsată" acestea dau implicit şi informaţia privind durata apăsării tastei.

5. Interfaţa serială pentru comunicaţia asincronă.

6. Interfaţa paralelă.

7. Ceas de timp real. Structura conţine şase contoare programabile realizate cu 8253, folosite astfel:

– Canalul 0 - generator de semnal pentru ceasul de timp al sistemului;

– Canalul 1 - cereri de ciclu DMA care să asigure ritmul pentru procesul de reîmprospătare a memoriei;

– Canalul 2 - generator de tonuri pentru difuzor;

– Canalul 3, 4 - semnale care asigură rata de transfer a interfeţei seriale.

8. Sistemul de întreruperi organizat pe opt niveluri prioritare care foloseşte circuitul 8259A. Întreruperile externe mascabile sunt plasate pe spaţiul 8÷15 al întreruperilor luate în considerare de microprocesor, iar pe spaţiul 0÷7, întreruperile externe nemascabile.

9. Modulul de acces direct la memorie (DMA). Este realizat cu un circuit 8237, care conţine 4 canale DMA, dintre care canalul 0 este folosit pentru reîmprospătarea memoriei dinamice, iar canalul 2 pentru cuplarea discului flexibil.

10. Interfaţa pentru discuri flexibile.

11. Magistrala de extensie pentru modulele suplimentare.

1.2. SCHEMA BLOC A SISTEMULUI

În figura 4.1 se prezintă schema bloc a microsistemului. Pentru a se evidenţia modul de conectare a resurselor microsistemului am utilizat descrierea PMS a structurii.

Una din primele decizii de proiectare este dacă realizăm un microsistem nestandard sau unul pe care vom rula aplicaţii deja existente. Cu alte cuvinte, se pune problema dacă dorim să păstrăm compatibilitatea cu microsistemele bazate pe microprocesorul 8086. Microprocesorul 8086 este un microprocesor cu magistrala de date pe 16 biţi. Pentru a păstra compatibilitatea se impune conectarea tuturor resurselor la o singură magistrală de 8 biţi. În acest fel se face şi o economie de *buffer*-e, deoarece nu mai este necesară utilizarea unui *buffer* pentru fiecare interfaţă ci cu ajutorul a două *buffer*-e se conectează toate interfeţele la magistrală. Această magistrală o numim magistrală locală (vezi figura 4.1). Acesta este motivul pentru care în descrierea structurii se utilizează magistrala locală $LOCD_{0:7}$.

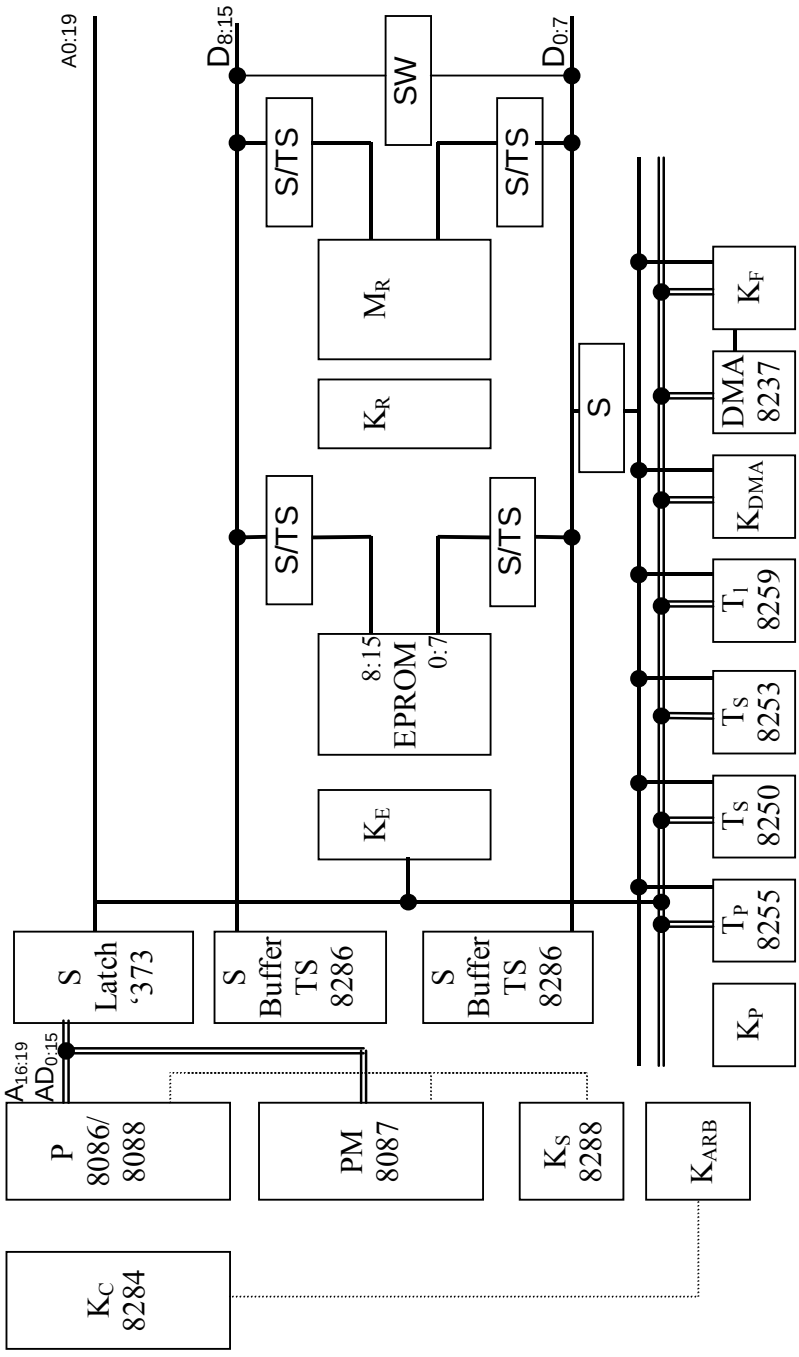


Figura 4.1. Schema bloc a microsistemului

În figura 4.1 se observă unităţile de comandă Kc, Ks, Ke, Kr, Kp, Karb, K_{DMA}, K_{adc}, K_F care asigură controlul primar al resurselor sistemului prin efectuarea următoarelor funcţiuni:

– generarea semnalelor de ceas pentru microprocesor şi logica externă (K_C);

– sincronizarea acţiunilor microprocesorului cu cele ale modulelor exterioare (K_C);

– interpretarea ciclului microprocesorului şi generarea semnalelor de comandă către resursele sistemului (K_S);

– rezolvarea conflictului de acces la magistrală între microprocesor şi modulul DMA (K_{arb});

– controlul resurselor CPU ce permit accesul la magistralele de adrese, date şi comenzi (K_{adc}).

– controlul accesului la memoria EPROM şi memoria RAM (K_E, K_R);

– controlul accesului interfeţelor la magistrala locală (K_P);

– controlul modulului de acces direct la memorie (K_{DMA});

– controlul interfaţei de disc flexibil (K_F);

1.3. PROIECTAREA UNITĂŢII DE COMANDĂ

Unitatea de comandă asigură controlul primar al resurselor sistemului prin efectuarea următoarelor funcţii:

- generează semnalele de ceas pentru microprocesor şi logica externă;
- sincronizează activitatea microprocesorului cu cea a modulelor externe;
- interpretează ciclul curent al microprocesorului şi generează semnalele de comandă către resursele sistemului în vederea efectuării acestuia;
- rezolvă conflictul de acces la magistrala între unitatea centrală de prelucrare şi modulul de acces direct la memorie;
- controlează resursele unităţii centrale de prelucrare ce permit accesul la magistrala de adrese, date şi comenzi.

În vederea îndeplinirii funcţiilor prezentate anterior, unitatea de comandă conţine următoarele unităţi funcţionale:

- Kc – logica pentru generarea ceasului şi sincronizarea microprocesorului cu logica externă;
- Ks – logica pentru generarea comenzilor pe magistrala sistemului;
- logica de control a accesului microprocesorului la magistrala sistemului;

- Karb – logica de arbitrare a accesului la magistrala între microprocesor şi DMA.

1.3.1. LOGICA PENTRU GENERAREA CEASULUI, SINCRONIZAREA MICROPROCESORULUI CU LOGICA EXTERNĂ ŞI RESET – K_C

Această unitate funcţională se proiectează utilizând circuitul 8284, conform recomandărilor Intel (vezi §1.4.4), şi are ca scop:

– generarea semnalelor de tact pentru resursele sistemului;

– sincronizarea microprocesorului cu modulele externe, asincrone, care au un timp de răspuns mai lent decât acesta;

– generarea semnalului de RESET.

Semnalele de intrare / ieşire aferente acestei unităţi funcţionale sunt prezentate în figura 4.2.

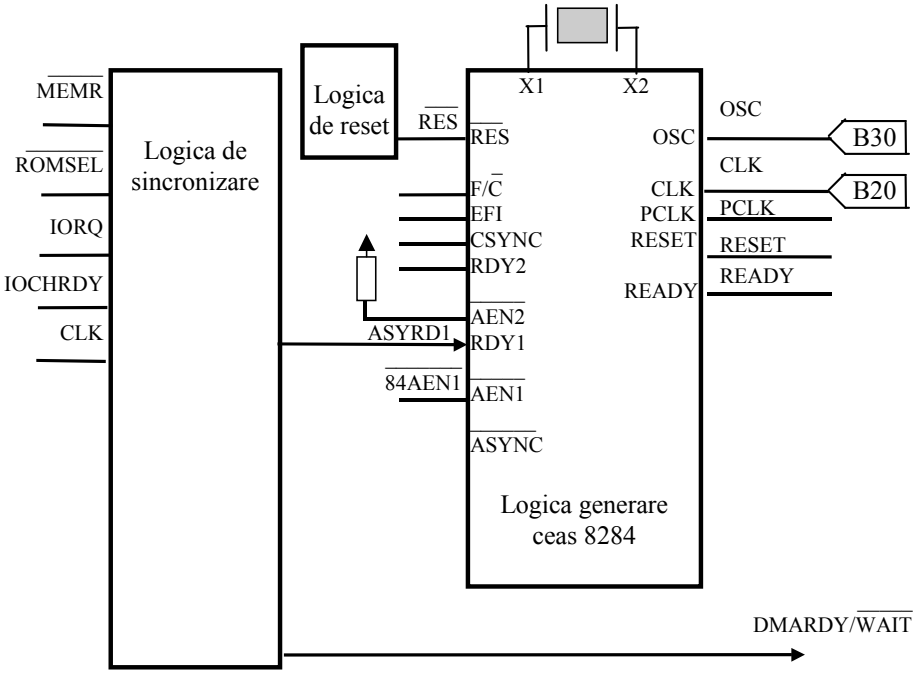


Figura 4.2. Logica pentru generarea ceasului, sincronizarea microprocesorului cu logica externă şi RESET

1.3.1.1. **Generarea semnalelor de tact**

Modulul generează următoarele semnale de ceas:

XOSC: tact de referinţă, cu frecvenţa de 14, 31818 MHz stabilită de cristalul de cuarţ conectat la intrările X1, X2. Acest semnal este amplificat, furnizând pe magistrala sistemului semnalul de ceas de referinţă sub numele OSC (B30). Perioada semnalului este de 70 ns cu un factor de umplere de 50%.

Observaţie: În structura propusă se utilizează ca tact de referinţă frecvenţa 14, 31818 Mhz deoarece s-a dorit păstrarea compatibilităţii cu sistemul IBM - XT. O modificare a acestei frecvenţe conduce la incompatibilitate în ceea ce priveşte ceasul sistemului ce contorizează ora, minutul şi secunda. Sistemul IBM - XT foloseşte frecvenţa de 14, 31818 MHz deoarece aceasta este de patru ori mai mare decât frecvenţa subpurtătoarei de crominanţă a normei de televiziune NTSC care este de 3, 579545 MHz. Deci dacă s-ar utiliza un televizor color NTSC se poate obţine uşor frecvenţa subpurtătoarei de crominanţă. Frecvenţa tactului de referinţă este cea mai apropiată de 3 x 5 MHz, care constituie frecvenţa maximă de lucru a microprocesoarelor 8086 din gama obişnuită, şi este în acelaşi timp multiplul frecvenţei de crominanţă.

CLK: semnalul de ceas al sistemului. Are frecvenţa de 1/3 din cea a semnalului XOSC, adică 4,77 MHz. Perioada semnalului este de 210 ns cu un factor de umplere de 33%. Este amplificat şi trimis pe magistrala de extensie a sistemului sub numele de CLOCK (B20). Este utilizat de următoarele resurse:

– microprocesorul 8086 şi coprocesorul matematic 8087;

– logica de arbitrare a accesului la magistrala între microprocesor şi circuitul DMA;

– modulul de acces direct la memorie.

PCLK: semnal de ceas pentru circuite periferice. Are frecvenţa de 1/6 din frecvenţa semnalului XOSC, respectiv 1/2 din frecvenţa semnalului CLK. Perioada semnalului este de 420 ns, cu un factor de umplere de 50%. Este amplificat şi trimis pe magistrala de extensie a sistemului sub numele de BPCLK (B45). Este utilizat de următoarele resurse:

– interfeţele seriale;

– interfaţa de tastatură;

– prin divizare cu doi, se utilizează de către numărătoarele programabile (8253-5) şi modulul de acces direct la memorie;

Relaţia între semnalele de ceas generate este prezentată în figura 4.3.

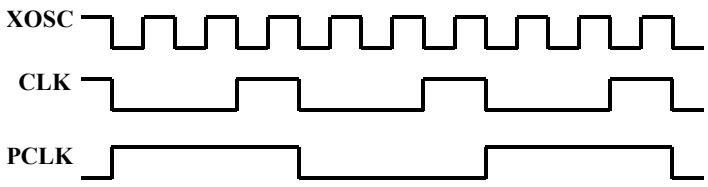


Fig. 4.3. Relaţia dintre semnalele de tact

1.3.1.2. **Logica de RESET**

Modulul generează şi semnalul RESET necesar iniţializării unor resurse ale sistemului. Semnalul RESET este utilizat de:

- microprocesorul 8086 şi coprocesorul matematic 8087;
- modulul de acces direct la memorie şi registrul de pagină DMA;
- logica de arbitrare a accesului la magistrala dintre microprocesor şi DMA;
- logica pentru generarea întreruperii nemascabile;
- logica de divizare a ceasului pentru circuite periferice (PCLK), a cărei ieşire este utilizată de numărătorul programabil 8253 şi modulul DMA;
- interfaţa de tastatură;
- interfaţa paralelă 8255-5;
- interfaţa serială.

Semnalul RESET este amplificat şi furnizat pe magistrala de extensie a sistemului sub numele BRESET (B2), în vederea iniţializării resurselor modulelor cuplate pe conectorii de extensie a sistemului.

Semnalul RESET se generează la punerea sub tensiune a sistemului (power-on reset) sau prin acţionarea unui buton de reset aflat pe panoul frontal al calculatorului. La primirea semnalului RESET microprocesorul 8086 iniţializează contorul program la zero, şi registrul de segment (CS) cu valoarea 0FFFFH, ceea ce este echivalent cu începerea execuţiei programului BIOS din memoria permanentă EPROM începând cu adresa 0FFFF:0000H.

De notat faptul că în urma activării semnalului RESET pentru un interval de timp suficient de mare, se pierde conţinutul memoriei interne RAM, deoarece memoria fiind de tip dinamic se perturbă procesul de reîmprospătare.

În figura 4.4 se prezintă schema de generare a semnalului de RESET.

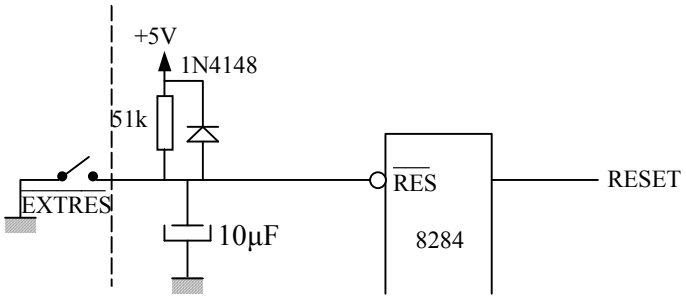


Fig 4.4. Schema de generare a semnalului RESET

1.3.1.3. **Logica de sincronizare**

Pentru sincronizarea microprocesorului 8086 şi a coprocesorului 8087 cu logica externă, în vederea introducerii unor stări de aşteptare T_W, când anumite module externe nu pot furniza sau prelua datele în timp util, se generează de către acestea semnalul READY.

Dacă READY este zero se va introduce un număr întreg de stări de aşteptare corespunzător cu durata acestuia (v. § 1.4.2.5 şi 1.4.2.6).

Semnalul READY este generat pe baza semnalelor 84AEN1 şi ASYRDY furnizate de logica de arbitrare a accesului la magistrala sistemului între microprocesor şi DMA. Semnificaţia semnalelor care participă la generarea lui READY este urmatoarea:

$\overline{\text{84AEN1}}$: este un semnal care indică care din modulele *master* ale sistemului, microprocesorul sau DMA, va controla magistrala.

Când $\overline{\text{84AEN1}}$ este zero logic, microprocesorul 8086 va fi cel care furnizează adresele, datele şi comenzile pe magistrala sistemului.

Când $\overline{\text{84AEN1}}$ este unu logic, modulul de acces direct la memorie (DMA) va fi cel care furnizează adresele, datele şi comenzile pe magistrala sistemului, microprocesorul 8086 trebuind să ramână într-o stare de aşteptare iar *buffer*-ele de acces ale acestuia la magistrală să fie dezactivate.

ASYRDY: este un semnal prin care se face sincronizarea transferului de date între microprocesor şi logica externă (memorie, *port*-uri de intrare/ieşire). În general, o operaţie de citire/scriere cu o memorie mai lentă (cu timp de acces mare) sau cu un *port* de intrare/ieşire necesită introducerea microprocesorului într-o stare de aşteptare (unul sau mai mulţi cicli de WAIT). Ieşirea din această stare se realizează în momentul în care logica respectivă (memorie sau *port* de intrare/ieşire) a efectuat operaţia în care era implicată. Logica externă trebuie să se sincronizeze cu microprocesorul prin generarea unui semnal de confirmare a încheierii operaţiei (ASYRDY).

În cazul modulului de bază al structurii propuse, operaţiile de citire din memoria EPROM (timp de acces maxim de circa 350÷450 ns) vor introduce o stare de aşteptare WAIT.

De asemenea, s-a prevăzut posibilitatea ca interfeţele de intrare/ieşire să introducă microprocesorul în starea de aşteptare, pentru unul sau mai mulţi cicli de WAIT. Interfeţele de intrare/ieşire conectate pe magistrala de extensie a

sistemului furnizează un răspuns de terminare a operaţiei curente. Răspunsul de la aceste interfeţe este furnizat prin intermediul semnalului IOCHRDY care este disponibil şi la conectorul de extensie (A10). Cu ajutorul semnalului de sincronizare IOCHRDY, logica de generare ASYRDY poate controla ieşirea READY prin intermediul circuitului 8284 astfel încât să se introducă un număr adecvat de stări de aşteptare.

Semnalul ASYRDY1 se află în mod normal pe "1". În ciclurile de I/O logica din figura 4.5 introduce o stare de aşteptare (WAIT). Semnalul IOCHRDY poate introduce un număr nedefinit de stări WAIT.

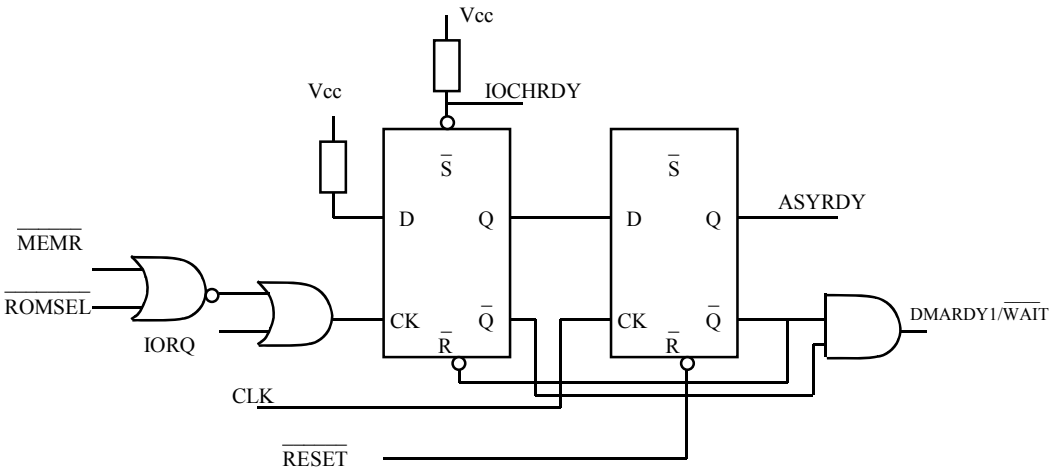


Figura 4.5. Logica de WAIT

1.3.2. LOGICA PENTRU GENERAREA COMENZILOR PE MAGISTRALA SISTEMULUI – K_S

Această unitate funcţională asigură generarea semnalelor de comandă necesare controlului resurselor unităţii centrale şi a modulelor de extensie. Semnalele aferente acestei unităţi funcţionale sunt prezentate în figura 4.6.

Această unitate funcţională are la bază circuitul 8288, care decodifică informaţia de stare $\overline{S0:2}$ furnizată de microprocesorul 8086 şi generează unele din semnalele de comandă ale magistralei sistemului.

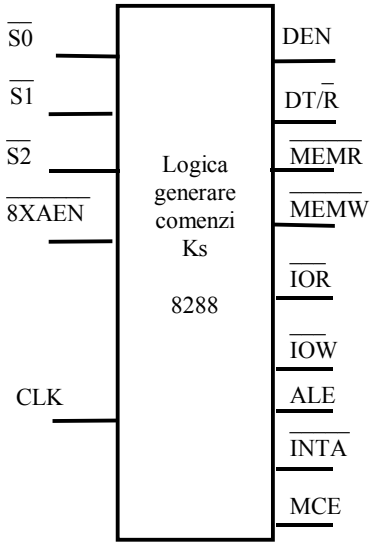


Figura 4.6. Logica de generare comenzi

Semnalele de comandă generate, în corelaţie cu informaţia de stare furnizată de microprocesor, pentru a specifica ciclul maşină curent care se execută, sunt prezentate în tabelul 4.1 (v. şi tabelul de la pag. 28).

Semnalele de comandă generate de către această unitate funcţională pot să fie dezactivate în momentul în care pe magistrala sistemului este activ un alt modul *master*. Dezactivarea comenzilor, corespunzătoare cu trecerea lor în starea de impedanţă mare, se realizează pe baza semnalului de intrare $\overline{8xAEN}$. Acest semnal este generat de logica de arbitrare a accesului la magistrală şi are aceeaşi semnificaţie cu $\overline{84AEN1}$, durata de acţiune a acestora fiind diferită.

Tabelul 4.1

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	
0	0	0	Recunoastere intrerupere $\overline{INTA}$
0	0	1	$\overline{IOR}$
0	1	0	$\overline{IOW}$
0	1	1	HALT
1	0	0	$\overline{MEMR}$ fetch
1	0	1	$\overline{MEMR}$
1	1	0	$\overline{MEMW}$
1	1	1	pasiv

În funcţie de acţiunea pe care o implică asupra resurselor sistemului, semnalele de comandă se împart în două categorii şi anume:

- semnale de comandă caracteristice execuţiei ciclului maşină curent;
- semnale de control al fluxului de date şi adrese.

Semnificaţia semnalelor din prima categorie este următoarea:

$\overline{\text{MEMR}}$: este un semnal care indică o operaţie de citire din memorie. Acest semnal este conectat atât la resursele modulului de bază cât şi la magistrala de extensie a sistemului (B11). Este furnizat de ieşirea $\overline{\text{MRDC}}$ a circuitului 8288 şi conectat împreună cu semnalul $\overline{\text{MEMR}}$ de la DMA, în vederea controlului operaţiei de citire din memorie. Activarea unuia dintre aceste semnale este realizată de logica de arbitrare a accesului la magistrală în funcţie de modulul care are acces la magistrală în acel moment. Asigură un *fan-out* de 17 unităţi logice de încărcare pentru magistrala sistemului.

$\overline{\text{MEMW}}$: este un semnal care indică o operaţie de scriere în memorie (ciclul maşină curent este de scriere în memorie). Acest semnal este conectat atât la resursele modulului de baza cât şi la magistrala de extensie a sistemului (B12). Este generat de ieşirea $\overline{\text{AMWC}}$ a circuitului 8288 şi conectat împreună cu semnalul $\overline{\text{MEMW}}$ de la DMA, în vederea controlului operaţiei de scriere în memorie. S-a folosit comanda anticipată de scriere furnizată de 8288, pentru a preveni introducerea unei stări de aşteptare care nu era necesară. În cazul în care s-ar fi utilizat $\overline{\text{MWTC}}$ s-ar fi introdus în mod implicit o astfel de stare pentru operaţiile de scriere în memorie deşi nu era necesară. Activarea unuia dintre aceste semnale (furnizat de microprocesor sau de DMA), este realizată de logica de arbitrare a accesului la magistrală în funcţie de modulul care are acces la magistrală în acel moment. Asigură un *fan-out* de 17 unităţi logice de încărcare pentru magistrala sistemului.

$\overline{\text{IOR}}$: este un semnal care indică o operaţie de citire de la un *port* de intrare (ciclul maşină curent este de citire de la un *port* de intrare). Acest semnal este conectat atât la resursele modulului de bază cât şi la magistrala de extensie a sistemului (B14). Este generat de ieşirea $\overline{\text{IORC}}$ a circuitului 8288 şi conectat împreună cu semnalul $\overline{\text{IOR}}$ de la DMA, în vederea controlului operaţiei de citire *port* de intrare. Activarea unuia dintre aceste semnale este realizată de logica de arbitrare a accesului la magistrală în funcţie de modulul care are acces la magistrală în acel moment. Asigură un *fan-out* de 17 unităţi logice de încărcare pentru magistrala sistemului.

$\overline{\text{IOW}}$: este un semnal care indică o operaţie de scriere la un *port* de ieşire (ciclul maşină curent este de scriere la un *port* de ieşire). Acest semnal este conectat atât la resursele modulului de bază cât şi la magistrala de extensie

a sistemului (B13). Este generat de ieşirea $\overline{\text{AIOWC}}$ a circuitului 8288 şi este conectat împreună cu semnalul $\overline{\text{IOW}}$ de la DMA, în vederea controlului operaţiei de citire la un *port* de ieşire. S-a folosit comanda anticipată de scriere furnizată de 8288, pentru a preveni introducerea unei stări de aşteptare care nu era necesară. În cazul în care s-ar fi utilizat $\overline{\text{IOWC}}$ s-ar fi introdus în mod implicit o astfel de stare pentru operaţiile de scriere la *port*-urile de ieşire deşi nu era necesară. Activarea unuia dintre aceste semnale este realizată de logica de arbitrare a accesului la magistrală în funcţie de modulul care are acces la magistrală în acel moment. Asigură un *fan-out* de 17 unităţi logice de încărcare pentru magistrala sistemului.

$\overline{\text{INTA}}$: este un semnal prin care microprocesorul confirmă luarea în considerare a întreruperii lansate de către sistemul de întreruperi. Este generat în momentul acceptării unei întreruperi externe, pentru a cere sistemului de întreruperi 8259A să activeze pe magistrala de date codul nivelului de întrerupere cu cea mai mare prioritate dintre cele pentru care au apărut cereri de întrerupere. Acest semnal este asociat numai întreruperilor externe mascabile (INTR) controlate de către 8259A. Pentru întreruperile externe nemascabile (NMI) nu se generează semnalul INTA. Semnalul este conectat la magistrala de extensie a sistemului (B42).

Semnificaţia semnalelor din a doua categorie este următoarea:

MCE: este un semnal prin care se permite circuitului 8259A MASTER situat pe modulul de bază al sistemului să transmită codul de selecţie pentru un circuit 8259A SLAVE, dispus pe una dintre plăcile de extensie nestandard. Ca efect circuitul 8259A SLAVE va transmite codul semnalului de întrerupere cu prioritatea cea mai mare dintre cele active şi gestionate de acesta. Din punct de vedere hardware structura sistemului permite conectarea în cascadă a mai multor circuite 8259A, însă BIOS-ul nu tratează decât semnalele de întrerupere provenite de la circuitul 8259A MASTER. În cazul în care se utilizează module 8259A SLAVE trebuie să se încorporeze în nucleul sistemului de operare şi procedurile pentru tratarea întreruperilor gestionate de către aceste circuite.

ALE: este un semnal furnizat de logica de generare comenzi la fiecare ciclu maşină şi este utilizat pentru a încărca adresa furnizată de microprocesor în starea T1, în registrul de adrese. Pe tranziţia din "1" în "0" va stroba adresele care sunt stabile pe magistrala comună de adrese şi date AD0:AD15 a microprocesorului într-un registru de adrese. Pentru a furniza adresa pe magistrala sistemului imediat ce microprocesorul o generează pe magistrala sa deci înainte de momentul de preluare în registrul de adrese al sistemului este necesar să se utilizeze registre de tip transparent.

DEN: este un semnal care activează, prin intermediul logicii de control al accesului microprocesorului la magistrala sistemului, circuitele de interfaţare la magistrala de date. Acest semnal indică momentele de timp când datele sunt

stabile pe magistrala microprocesorului pentru operaţiile de ieşire (scriere) şi când logica externă poate activa datele pe magistrală pentru a fi preluate de microprocesor în operaţiile de intrare (citire).

DT / $\overline{\mathbf{R}}$: este un semnal prin care logica de generare comenzi stabileşte, în funcţie de codul maşină curent, sensul fluxului de date prin circuitele de interfaţare la magistrala de date. Schema detaliată a logicii de generare comenzi pe magistrala sistemului este prezentată în figura 4.X.

1.3.3. LOGICA DE CONTROL AL ACCESULUI MICROPROCESORULUI LA MAGISTRALA SISTEMULUI

Această unitate funcţională are ca rol controlul registrului de adrese şi a circuitului de interfaţare cu magistrala de date astfel încât microprocesorul să fie conectat în mod corespunzător la magistrala sistemului. Semnalele asociate acestei unităţi funcţionale sunt prezentate în figura 4.7.

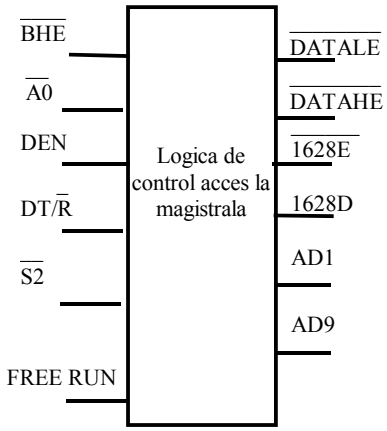


Figura 4.7. Semnalele din logica de control a accesului la magistrală

Controlul accesului la magistrală va trebui să asigure conectarea microprocesorului 8086 la magistrala sistemului, în momentul în care acesta utilizează resursele sistemului (memorie, interfeţe de intrare/ieşire) şi să dezactiveze acest acces în perioada de timp când aceste resurse sunt sub controlul modulului DMA.

Semnalele generate de logica de control a accesului la magistrala împreună cu semnalele furnizate de logica de generare a comenzilor contribuie la realizarea următoarelor funcţii:

- preluarea adreselor microprocesorului în registrul de adrese al sistemului (registru de tip transparent, cu ieşiri de tip trei stări - 74LS373);
- activarea / dezactivarea ieşirilor registrului de adrese în funcţie de modulul care deţine controlul magistralei, microprocesorul sau DMA;
- activarea / dezactivarea şi stabilirea sensului circuitelor de interfaţare la magistrala de date (circuite bidirecţionale cu ieşiri de tip trei stari – 8286 sau T4LS245) în vederea controlului fluxului de date;
- conectarea magistralei de date ce conţine biţii mai puţin semnificativi D0:7 la biţii cei mai semnificativi D8:15 şi invers (operaţia de *swapping*), în vederea asigurării cuplării unei magistrale locale pe 8 biti.

Această funcţie (de *swapping*) este necesară pentru a permite construirea unei magistrale locale de date pe 8 biţi, care să permită cuplarea circuitelor de interfaţă standard: 8259A, numărătoare programabile 8253-5, interfaţa paralelă 8255-5, etc. Aceste circuite de interfaţă sunt organizate pe 8 biţi şi este necesar ca ele să aibă porturi de intrare / ieşire atât la adrese pare cât şi la adrese impare. În acest fel magistrala locală de date organizată pe 8 biţi trebuie să poată fii conectată atât la biţii D0:7 ai magistralei sistemului cât şi la biţii D8:15 în funcţie de adresa portului de intrare / ieşire (adresă pară respectiv impară). Acest mecanism este realizat prin introducerea unui circuit bidirecţional cu ieşiri de tip trei stări care face conectarea între biţii de date D0:7 şi D8:15 sub controlul semnalelor $\overline{1628E}$, 1628D.

În vederea măririi testabilităţii sistemului, în scop didactic, s-a prevăzut o funcţie de execuţie liberă (freerun) a microprocesorului. Stabilirea regimului de execuţie liberă este realizată prin poziţionarea unui comutator DSW1 (2-15) pe poziţia ON. Activarea regimului de execuţie liberă activează permanent semnalul ASYRDY. Totodată, pe durata activă a semnalului DEN liniile AD1 şi AD9 sunt forţate la zero. Ca urmare a acestor acţiuni, microprocesorul are semnalul READY activ şi va citi la fiecare ciclu instrucţiune instructiunea STD - set direcţie (cu codul 0FDH). După fiecare ciclu instrucţiune, microprocesorul incrementează contorul de instrucţiuni IP. În felul acesta liniile de adresă A1:15 evoluează în regim de numărare liniară. Această facilitate de depanare permite detectarea scurtcircuitelor între liniile de adrese A1:15 precum şi întreruperile acestor linii.

Funcţiile logice ale semnalelor generate de către această unitate funcţională au următoarea exprimare analitică:

$$\overline{1628E} = INTA + \overline{A}_0 + \overline{S2}.BHE$$

$$1628D = \overline{MEMW + IOR}$$

$$\overline{DATAHE} = \overline{DEN} + FREERUN$$

$$\overline{DATALE} = \overline{DEN} + FREERUN + 1628E$$

$$AD1 = \overline{DEN} + \overline{FREERUN}$$
$$AD9 = \overline{DEN} + \overline{FREERUN}$$

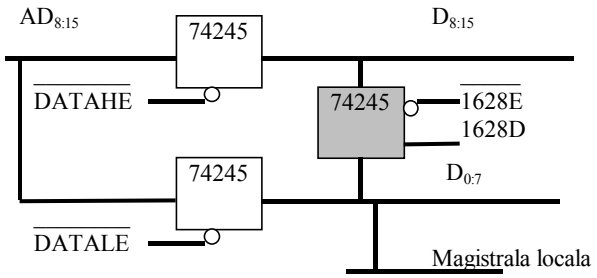


Figura 4.8. Logica pentru controlul accesului la magistrala sistemului

1.3.3.1. **Controlul registrului de adresă**

În stabilirea semnalului de comandă pentru registrul de adrese s-a ţinut seama de funcţiile enumerate anterior. Astfel, ca semnal de încărcare în registrul de adrese s-a folosit ALE, figura 4.9, iar pentru activarea adreselor pe magistrala sistemului s-a folosit $\overline{8xAEN}$, semnal generat de logica de arbitrare a accesului la magistrala dintre microprocesor şi DMA.

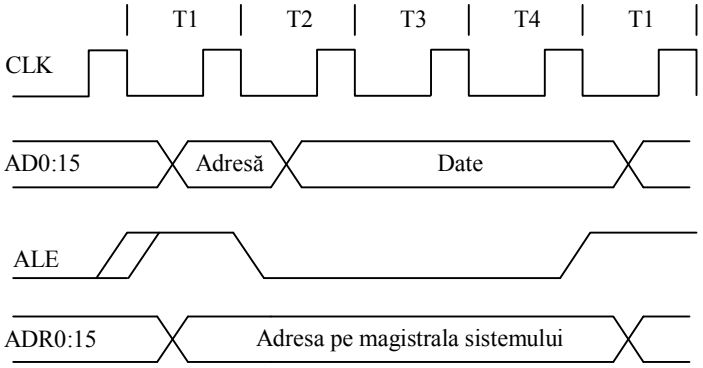


Figura 4.9. Relaţia dintre adresa furnizată de microprocesor AD0:15 şi adresa pe magistrala sistemului

Registrul de adrese al sistemului s-a implementat cu circuite de tip transparent, cu ieşiri trei stări, 74LS373. S-a ales acest tip de circuit deoarece este necesar să se furnizeze adresa (celulei de memorie sau a *port*-ului de I/O)

pe magistrala sistemului imediat ce aceasta este generată de microprocesor. Deci pe durata cât ALE = 1, intrarea registrului de adrese este transferată la ieşirea acestuia (funcţia de transparenţă) cu scopul ca logica externă care o utilizează să beneficieze imediat de adresa furnizată de microprocesor şi nu abia în momentul în care este memorată în registru (când ALE trece din "1" în "0"), pentru a evita introducerea unei eventuale stări de asteptare. Adresa nu poate fi preluată pe frontul "0 - 1" al semnalului ALE deoarece nu este înca stabilizată. Ieşirile registrului de adrese sunt activate de semnalul $\overline{8XAEN} = 0$, microprocesorul fiind cel care furnizează adresa pe magistrala sistemului.

În cazul unui ciclu DMA microprocesorul este într-o stare de aşteptare $\overline{8XAEN} = 1$, ieşirile registrului de adrese sunt dezactivate (stare de impedanţă mare) adresa pe magistrala sistemului fiind furnizată de modulul de acces direct la memorie.

1.3.3.2. **Controlul circuitelor de interfaţare la magistrala de date**

Magistrala AD0:15 a microprocesorului este conectată la magistrala bidirecţională de date a sistemului D0:15 prin intermediul circuitelor de interfaţare la magistrala de date - 8286 sau 74LS245.

Aşa cum s-a menţionat, semnalul FREERUN = 0 activ, stabilit în faza de depanare a sistemului dezactivează ambele circuite de interfaţare la magistrală şi forţează pe magistrala de date AD0:15 configuraţia 0FDFDH, prin intermediul unor circuite cu colectorul în gol, figura 4.8, ceea ce reprezintă codul instrucţiunii STD.

În funcţie de ciclul maşină curent al microprocesorului trebuie activate circuitele de interfaţare. Sensul de transfer al datelor este stabilit de logica de comandă prin intermediul semnalului DT/R.

Semnalele sunt specifice fiecăruia din circuitele de interfaţare în funcţie de modul în care se face citirea / scrierea datelor pe cuvânt sau octet şi în funcţie de ciclul maşină curent.

Reamintim că microprocesorul poate să transfere date cu memoria sau cu porturile de intrare / ieşire pe cuvânt (16 biţi) sau octet (8 biţi). Pentru a specifica modul de transfer al datelor, microprocesorul poziţionează semnalele $\overline{BHE}$ şi A0 în felul următor (tabelul 4.2):

Tabelul 4.2

$\overline{BHE}$	ADR0	Tip date
0	0	Date pe cuvânt D0:15, de la adresă pară
0	1	Date pe octet D8:15, de la adresă impară
1	0	Date pe octet D0:7, de la adresă pară
1	1	Neutilizat

Circuitele de interfaţare la magistrala de date asigură fluxul de date între microprocesor şi:

- sistemul de întreruperi;
- *port*-urile de intrare/ieşire;
- memorie.

1.3.3.3. **Ciclul de recunoaştere a întreruperii**

În cadrul unui ciclu de recunoaştere a întreruperii este necesar să se activeze partea mai puţin semnificativă a circuitului de interfaţă, biţii D0:7, pentru a se permite vectorului de întrerupere furnizat de circuitul 8259A să ajungă pe magistrala microprocesorului în vederea ajungerii la celula capcană care face legătura cu rutina de tratare a întreruperii ce urmează să fie tratată.

În cadrul unui ciclu de recunoaştere a întreruperii $\overline{INTA} = 0$ şi indiferent de celelalte semnale ($\overline{S2} = x$, $\overline{BHE} = x$, $A_0 = x$) sunt activate ambele circuite de interfaţare la magistrala de date, însă vectorul de întreruperi va circula numai prin circuitul D0:7, figura 4.10.

Observaţie: circuitele de interfaţare nehaşurate sunt active, iar cele haşurate sunt inactive având ieşirile în starea de impedanţa mare.

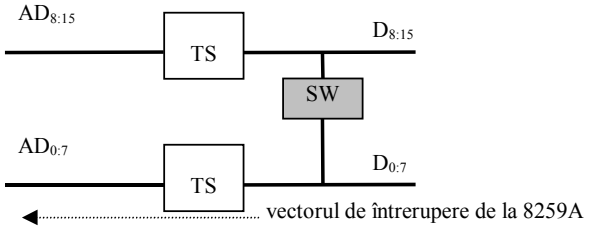
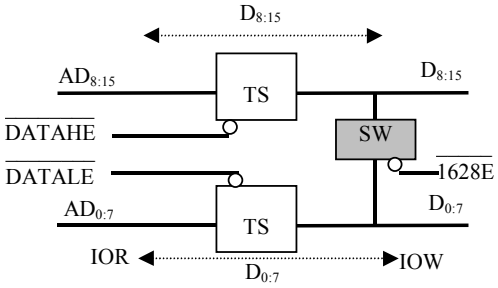


Figura 4.10. Tranferul vectorului de întrerupere

În cazul în care ciclul curent nu este un ciclu de recunoaştere a unei întreruperi, fluxul de date este controlat de semnalele specifice.

1.3.3.4. **Ciclul de transfer de date**

Pentru un ciclu de transfer de date cu un *port* de intrare/ieşire $\overline{S2} = 0$, respectiv cu memoria $\overline{S2} = 1$, controlul circuitelor de interfaţare la magistrala de date se face ca în figura 4.11 respectiv 4.12.



$\overline{DATAHE}$ =0

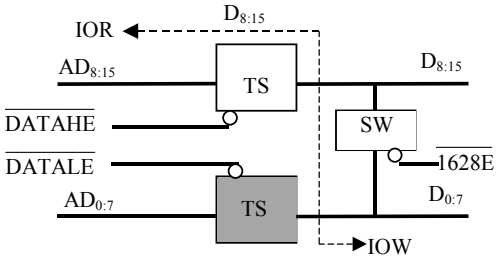
$\overline{DATALE}$ =0

$\overline{1628E}$ =1 (inactiv)

$\overline{BHE} = 0$

ADR0 = 0

a) Transferul datelor de intrare/ieşire pe 16 biţi



$\overline{DATAHE}$ =0

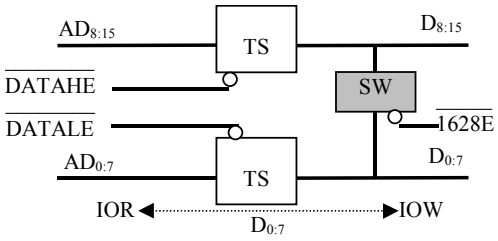
$\overline{DATALE}$ =1 (inactiv)

$\overline{1628E}$ =0

$\overline{BHE} = 0$

ADR0 = 1

b) Transferul datelor de intrare/ieşire pe 8 biţi pe magistrala D8:15

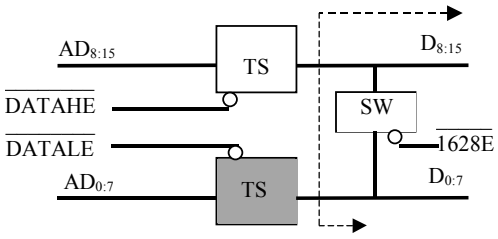


$\overline{DATAHE}$ =0 $\overline{BHE}$ = 1

$\overline{DATALE}$ =0 ADR0 = 0

$\overline{1628E}$ =1 (inactiv)

c) Transferul datelor de intrare/ieşire pe 8 biţi pe magistrala D0:7



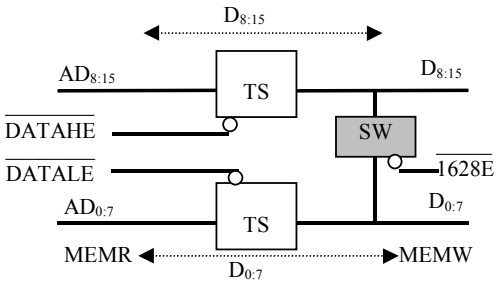
$\overline{DATAHE}$ =0 $\overline{BHE}$ = 1

$\overline{DATALE}$ =1 (inactiv) ADR0 = 1

$\overline{1628E}$ =0

d) Transferul datelor de intrare/ieşire pe 8 biţi (controlat de alt *master*)

Fig. 4.11. Ciclul de transfer de date de intrare/ieşire

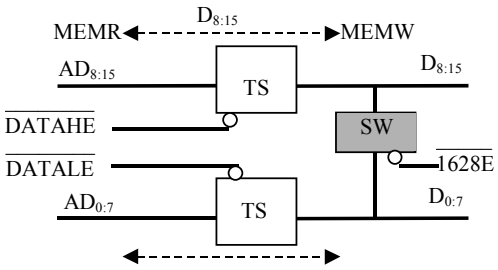


$\overline{DATAHE}$ =0 $\overline{BHE}$ = 0

$\overline{DATALE}$ =0 ADR0 = 0

$\overline{1628E}$ =1 (inactiv)

a) Transferul datelor cu memoria pe 16 biţi

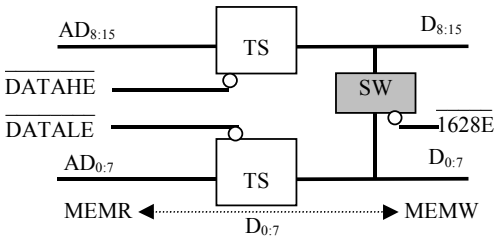


$\overline{DATAHE}$ =0 $\overline{BHE}$ = 0

$\overline{DATALE}$ =0 ADR0 = 1

$\overline{1628E}$ =1 (inactiv)

b) Transferul datelor cu memoria pe 8 biţi pe magistrala D8:15

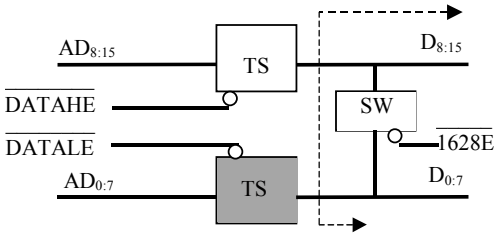


$\overline{DATAHE}$ =0 $\overline{BHE}$ = 1

$\overline{DATALE}$ =0 ADR0 = 0

$\overline{1628E}$ =1 (inactiv)

c) Transferul datelor cu memoria pe 8 biţi pe magistrala D0:7



$\overline{DATAHE}$ =0 $\overline{BHE}$ = 1

$\overline{DATALE}$ =1 (inactiv) ADR0 = 1

$\overline{1628E}$ =0

d) Transferul datelor cu memoria pe 8 biţi (controlat de alt *master*)

Fig. 4.12. Ciclul de transfer de date cu memoria

Structura propusă poate avea o structură reconfigurabilă pe 8 sau 16 biţi în funcţie de microprocesorul care se utilizează, 8088 sau 8086. În configuraţia standard structura propusă se proiectează cu microprocesorul 8086, magistrala sistemului fiind organizată pe 16 biţi.

Schimbarea microprocesorului 8086 cu 8088 este posibilă deoarece cele două microprocesoare sunt compatibile la nivel de conexiuni şi la nivel de set de instrucţiuni numai că microprocesorul 8086 comunică cu exteriorul printr-o magistrală de date de 16 biţi iar 8088 comunică cu exteriorul printr-o magistrala de 8 biţi.

Microprocesorul 8088 execută operaţiile de transfer pe cuvânt, cu memoria sau *port*-urile de intrare/ieşire, întotdeauna în două cicluri maşină, pe când 8086 le execută într-un singur ciclu maşină dacă cuvântul se află la adresă pară, respectiv în două cicluri maşină dacă cuvântul se află la adresă impară.

1.3.4. LOGICA DE ARBITRARE A ACCESULUI LA MAGISTRALA SISTEMULUI ÎNTRE MICROPROCESOR ŞI MODULUL DE ACCES DIRECT LA MEMORIE

Această unitate funcţională are rolul de a realiza excluderea mutuală a modulelor MASTER aflate pe placa de bază care doresc accesul la magistrala sistemului. Magistrala sistemului este o resursă unică, multiplexată în timp, care trebuie atribuită unui singur modul la un moment dat.

Controlul adreselor, datelor şi comenzilor pe magistrala sistemului se poate realiza de către microprocesorul 8086, modulul de acces direct la memorie şi cu anumite rezerve sau precauţii speciale de către un modul *master* nestandard cuplat pe magistrala de extensie.

Pentru a nu aparea conflict, în ceea ce priveşte accesul la această resursă unică, s-a proiectat o logică de arbitrare a accesului la magistrala sistemului. Această unitate funcţională rezolvă conflictul între două cereri simultane de la microprocesor şi DMA şi permite accesul unui eventual modul *master* cuplat pe magistrala de extensie.

În cazul unor cereri simultane de acces la magistrală se dă prioritate modulului DMA. Semnalele de intrare/ieşire aferente acestei unităţi funcţionale sunt prezentate în figura 4.13.

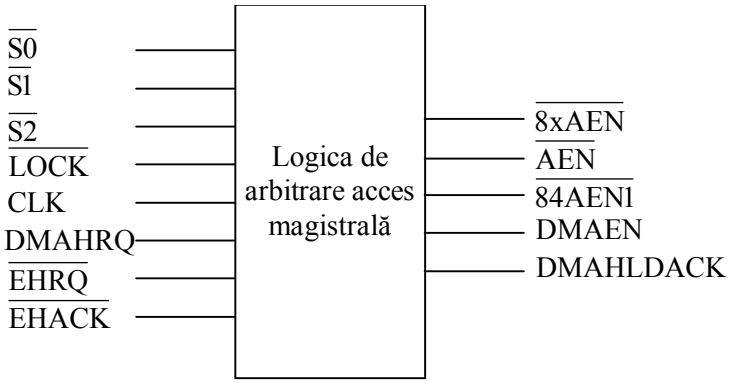


Figura 4.13. Semnalele de intrare/ieşire pentru logica de arbitrare acces magistrală

Pentru a putea explica mai bine logica de arbitrare a accesului la magistrala sistemului este necesar să reamintim evoluţia unui ciclu maşină al microprocesorului şi a unui ciclu DMA, urmând să descriem interacţiunea între aceste cicluri. Reamintim acţiunile efectuate de microprocesor şi DMA în ciclul maşină.

1.3.4.1. **Ciclul maşină procesor**

Un ciclu maşină din cadrul ciclului instrucţiune se desfăşoară în mod normal pe durata a patru stări maşină T1, T2, T3 şi T4, vezi şi §1.4.1, (nu s-a luat în considerare cazul în care logica externă nu răspunde în timp util şi se introduce un număr întreg de stări de aşteptare T_W între starea T3 şi T4).

În starea T1 microprocesorul furnizează adresa, ce va fi păstrată într-un registru de adrese pe toată durata ciclului curent, şi se generează informaţia de stare pe liniile $\overline{S0:2}$ care va specifica natura ciclului maşină curent. Identificarea ciclului maşină curent se realizează de către logica de generare comenzi, în speţă de către circuitul 8288.

Starea T2 este o stare în care se comută direcţia magistralei microprocesorului în vederea efectuării operaţiilor de citire date. În stările T3 şi T4 are loc transferul datelor pe magistrala sistemului. Dacă memoria sau interfeţele de intrare / ieşire nu pot să furnizeze sau să preia datele în timp util, prin intermediul logicii de sincronizare, vor introduce microprocesorul într-o stare de aşteptare T_W care va dura un număr întreg de perioade de tact. Starea de aşteptare este introdusă între T3 şi T4. Terminarea unui ciclu maşină este indicat prin trecerea liniilor de stare $\overline{S0:2}$ în starea pasivă (1, 1, 1). Trecerea în

starea activă a liniilor de stare $\overline{S0:2}$, are loc în timpul stării T4, ceea ce este echivalent cu începutul unui ciclu de acces la magistrală. Trecerea în starea pasivă are loc spre sfârşitul stării T3 sau a ultimului ciclu de aşteptare T_W, ceea ce reprezintă sfârşitul ciclului de acces la magistrală.

1.3.4.2. Ciclul maşină efectuat de modulul de acces direct la memorie

Un ciclu DMA de acces la magistrală se desfăşoară în mod normal pe durata a patru stări S1, S2, S3, S4. Durata unei stări este egală cu durata unei perioade de ceas. Aceste stări sunt precedate de două stări SI şi SO.

În starea SI se aşteaptă lansarea unei cereri, din partea interfeţelor de intrare/ieşire, pentru un transfer de date, DRQi.

Ciclul DMA propriu-zis începe în momentul când s-a primit o cerere de transfer. În acest moment modulul DMA, trece în starea SO şi lansează o cerere de acces la magistrală, DMAHRQ. Ramâne în această stare până când primeşte răspuns de acceptare a cererii de la logica de arbitrare, până la activarea de către aceasta a semnalului DMAHLDACK. Se trece în starea S1 în care se începe acccesul la resursele sistemului prin furnizarea adresei după care se trece în starea S2 în care se activează comenzile specifice ciclului curent şi se lansează raspunsul DACKi de confirmare a luării în considerare a cererii DRQi. Comenzile de scriere IOW / MEMW se menţin active şi pe durata stării S3 în care se analizează dacă resursele cu care este cuplat modulul DMA răspund la operaţiile iniţiate. În cazul în care operaţia din ciclul curent nu se poate efectua în timpul stării S3, resursa externă implicată în transfer nefurnizând semnalul DMARDY se intră într-o stare de aşteptare S_W. În cazul în care s-a terminat operaţia în curs de execuţie se trece în starea S4 în care se anulează comenzile şi se indică terminarea ciclului DMA curent prin anularea DACKi.

Organigrama generală de efectuare a unui ciclu DMA este prezentată în figura 4.14.

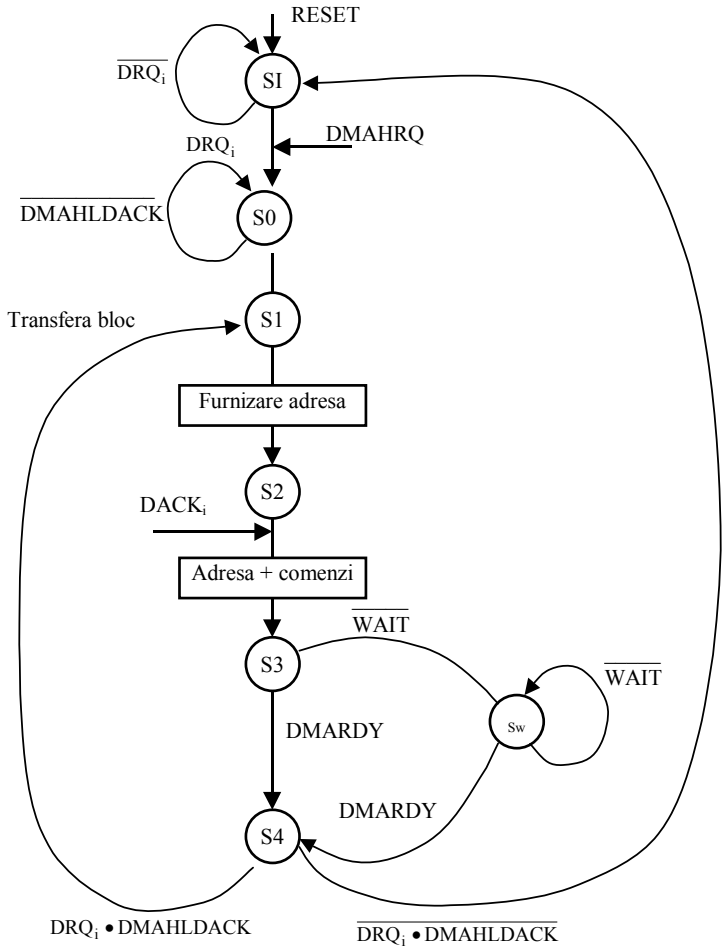


Figura 4.14. Automatul pentru arbitrarea accesului modulului de acces direct la memorie

1.3.4.3. **Logica de arbitrare**

Logica de arbitrare a accesului la magistrala sistemului prezentată în figura 4.15 reprezintă un arbitru simplu care izolează microprocesorul de la magistrală când modulul de acces direct la memorie (DMA) face cerere de acces în vederea preluării controlului magistralei sistemului.

Dacă modulul DMA primeşte o cerere de transfer din partea unei interfeţe de I/E (DRQi) şi canalul DMA respectiv a fost programat în faza de

iniţializare, modulul de acces direct la memorie va face o cerere către arbitrul de magistrală activând semnalul DMAHRQ.

Acest semnal, DMAHRQ, prin care DMA face cerere de acces la magistrală este furnizat şi pe magistrala sistemului (B44) reprezentând o informaţie de stare pentru modulele de extensie, în special pentru eventualele module MASTER.

În urma activării semnalului DMAHRQ, modulul DMA intră în starea SO şi ramâne în această stare până când primeşte DMAHLDACK de la arbitrul de magistrală prin care se confirmă acordarea accesului la magistrala sistemului.

Cererea de acces la magistrală DMAHRQ este luată în considerare de către logica de arbitrare numai când:

– microprocesorul este într-o stare pasivă ($\overline{S0-2}$ = 1, 1, 1);

– semnalul $\overline{LOCK}$, generat de microprocesor, este inactiv;

– semnalul $\overline{EHRQ}$, de pe magistrala sistemului este inactiv.

În cazul în care sunt îndeplinite toate condiţiile ca modulul DMA să primească accesul la magistrală, se dezactivează semnalele: $\overline{84AEN1}, \overline{8xAEN}$ ceea ce implică:

- logica de sincronizare introduce microprocesorul într-o stare de aşteptare pe toată durata efectuării ciclului DMA;
- logica de generare comenzi dezactivează comenzile de pe magistrala sistemului;
- logica de control al accesului la magistrală dezactivează circuitele de interfaţare la magistrala de date şi ieşirile registrului de adrese.

Aceste acţiuni izolează microprocesorul de magistrala sistemului pe toată durata ciclului DMA şi permite modulului de acces la memorie să preia controlul magistralei.

Confirmarea faptului că DMA poate începe ciclul de transfer este specificată prin activarea semnalului DMAHLDACK. Acest semnal poate fi activat numai dacă pe magistrala sistemului se permite acest lucru adică $\overline{EHACK}$ (B43) este inactiv.

După o perioadă de timp se activează DMAEN care permite:

– accesul adreselor furnizate de modulul DMA la magistrala sistemului;

– activarea comenzilor furnizate de DMA pe magistrala sistemului.

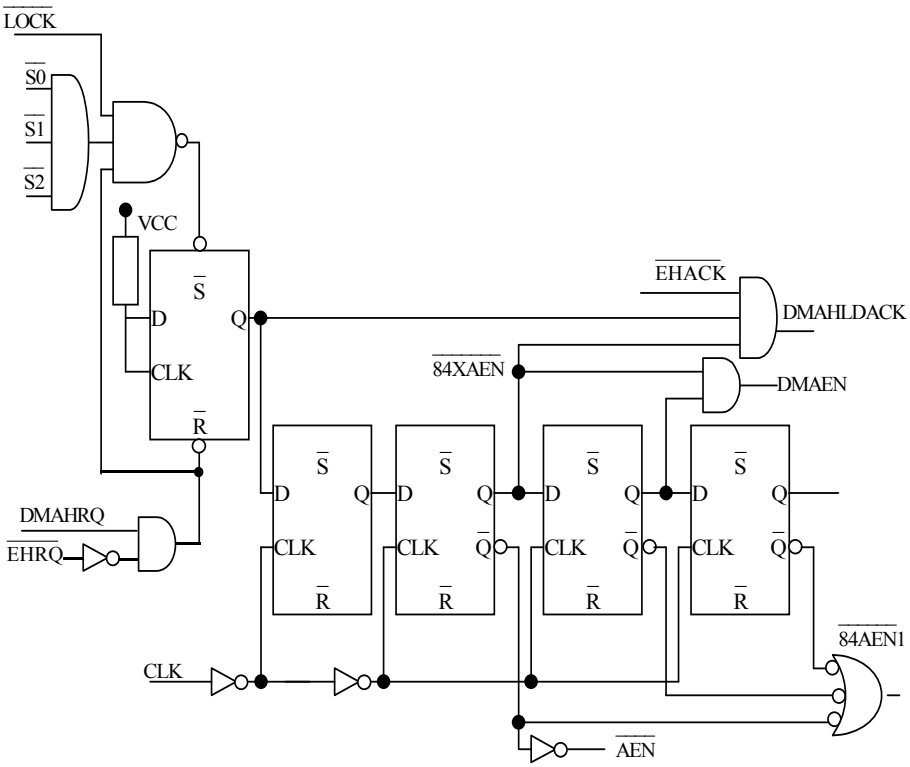


Figura 4.15. Logica de arbitrare a accesului la magistrala sistemului

Logica de arbitrare generează pe magistrala sistemului semnalul AEN (A11) prin care se informează modulele de extensie cine controlează magistrala de adrese, date şi comenzi.

Astfel, dacă $\begin{cases} \overline{AEN} = 0, \text{ magistrala sistemului este controlată de microprocesor} \\ \overline{AEN} = 1, \text{ , magistrala sistemului este controlata de modulul DMA} \end{cases}$

La terminarea unui ciclu de transfer curent, logica de arbitrare va izola modulul DMA de magistrala sistemului prin dezactivarea semnalelor DMAHOLDACK şi DMAEN şi va permite accesul microprocesorului la magistrală prin activarea semnalului AxAEN.

Microprocesorul, după activarea sa pe magistrală, mai stă două perioade în starea de aşteptare datorită semnalului 84AEN1.

De menţionat faptul că după ce AEN devine inactiv, circuitele de interfaţă la magistrala de date permit accesul microprocesorului iar registrul de adrese asociat acestuia este activat însă logica de comandă, în speţă 8288, nu va activa comenzile decât după 105÷275 ns după activarea sa. Acest lucru se

realizează prin întârzierea cu două perioade de tact a activării semnalului 84AEN1 (de la 8484) faţă de activarea semnalului 8xAEN (de la 8288).

Dacă microprocesorul era în stare de aşteptare, în cursul ciclului curent, când a venit o cerere de DMA, 84AEN1 poate permite imediat continuarea ciclului întrerupt de transferul DMA. În cazul în care microprocesorul era în timpul executării unui ciclu curent (fără să fi fost în stare de aşteptare) 84AEN1 va trebui să fie întârziat pentru a furniza timp de acces echivalent cu un ciclu de magistrala normal.

Cererea de acces DMAHRQ devine inactivă după fiecare octet transferat permiţând cel puţin un ciclu maşină microprocesor între două transferuri succesive DMA. De menţionat că modulul MASTER cu care microprocesorul işi dispută accesul la magistrală nu va câştiga controlul magistralei în nici unul din următoarele cazuri:

- 8086 este în timpul unui ciclu de magistrală curent;
- 8086 este în timpul ciclului de instrucţiune prefixat cu LOCK;
- 8086 este în ciclul de continuare al unei întreruperi.

Semnalul LOCK de la 8086 este activ între ciclurile INTA garantând menţinerea controlului magistralei de către microprocesor.

1.4. MEMORIA EPROM

Pe modulul de bază al structurii ce se va proiecta s-a prevăzut posibilitatea cuplării unei memorii permanente EPROM de capacitate între 12K octeţi şi 64K octeţi. De fapt s-au prevăzut 6 socluri de 28 de pini care permit cuplarea unui număr de 6 capsule de memorie EPROM. S-a ales această organizare pentru scop didactic.

În funcţie de tipul de memorie EPROM folosit se realizează capacitatea memoriei astfel:

Tip memorie	Capacitate	Spaţiu de adresare
2716 (2K x 8)	12K	0FD000H - 0FFFFFH
2732 (4K x 8)	24K	0FA000H - 0FFFFFH
2764 (8k x 8)	48K	0F4000H - 0FFFFFH
27128 (16K x 8)	64K	0F0000H - 0FFFFFH

Pentru a introduce diferite tipuri de EPROM - uri în soclurile prevăzute pe modulul de bază este necesar să se prevadă un mecanism hardware de comutare a adresei astfel încât să se asigure liniile de adrese corespunzătoare capsulei de memorie.

Având în vedere spaţiul de adresare în care este plasată memoria EPROM rezultă urmatorul tabel de utilizare a liniilor de adresă:

Tip memorie	A19	A18	A17	A16	A15	A14	A13	A12	A11 - A1
2716	1	1	1	1	x	x	selecţie capsulă		Adresă pt. capsulă
2732	1	1	1	1	x	selecţie capsulă		Adresă pt. capsulă	
2764	1	1	1	1	selecţie capsulă		Adresă pt. capsulă		
27128	1	1	1	1	sel. capsulă	Adresă pt. capsulă			

Se observă că în cazul utilizării memoriei 2716 sau 2732, biţii de adresă ADR15:14 respectiv ADR15 nu sunt utilizaţi, ceea ce conduce la faptul că informaţia din memoria EPROM se regăseşte în mai multe spaţii de adresare.

Astfel memoria EPROM realizată cu capsule 2716 poate să fie adresată în spaţiul:

0FD000H - 0FFFFFH sau

0F1000H - 0F3FFFH

0F5000H - 0F7FFFH

0F9000H - 0FBFFFH

iar când este realizată cu capsule 2732 poate să fie adresată în spaţiile:

0FA000H - 0FFFFFH sau

0F2000H - 0F7FFFH

De asemenea, se observă că în cazul utilizării memoriei 27128, la decodificatorul de adresare participă numai linia ADR15 ceea ce conduce la posibilitatea de adresare numai a 4 circuite din cele 6.

Pentru implementarea în memoria EPROM a unui set de programe de depanare selectabile prin intermediul unor comutatoare, trebuie să se prevadă o posibilitate hardware de interschimbare a unor zone de memorie (din punct de vedere al adresabilităţii). Este necesar ca fiecare program de depanare (de lungime maximă 256 octeţi) aflat fizic în zona de memorie 0FD000H - 0FDFFFH, indiferent de tipul de memorie utilizat, în momentul selectării pentru execuţie să fie văzut de către microprocesor în spaţiul 0FFF00H - 0FFFFFH. Acest lucru este necesar pentru a putea fi lansat direct în execuţie în momentul pornirii sistemului.

În cele ce urmează se prezintă modul de implementare a memoriei EPROM pentru structura proiectată.

1.4.1. SCHEMA BLOC A MEMORIEI EPROM

Având în vedere cele precizate anterior rezultă urmatoarea schemă bloc pentru memoria EPROM a structurii propuse.

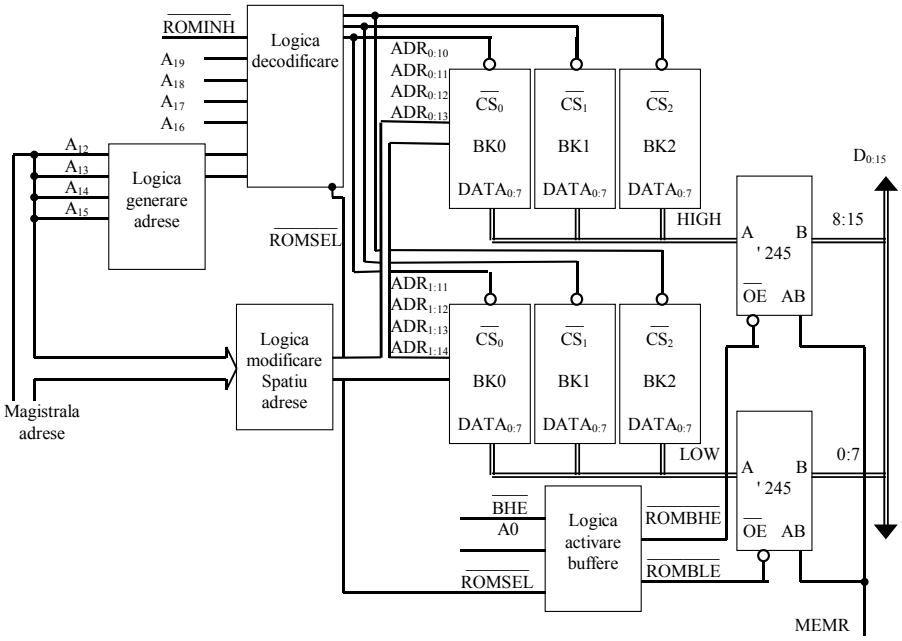


Figura 4.16. Schema bloc a memoriei EPROM

1.4.2. LOGICA DE GENERARE ADRESE

Are scopul de a asigura, pentru decodificatorul spaţiului de adrese, liniile de adrese corespunzătoare tipului de memorie utilizat. Constă dintr-o schemă de multiplexare ca în figura 4.17.

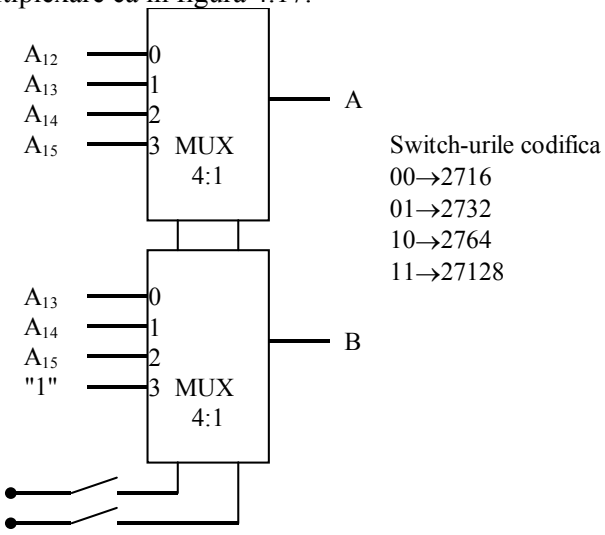


Figura 4.17. Logica de generare adrese

În funcţie de *strap*-urile S36, S37 se asigură următoarele adrese pentru decodificarea spaţiului de adrese.

S36	S37	Tip memorie	Intrare B	Intrare A
da	da	2716	A_{13}	A_{12}
da	nu	2732	A_{14}	A_{13}
nu	da	2764	A_{15}	A_{14}
nu	nu	27128	1	A_{15}

1.4.3. LOGICA DE DECODIFICARE

Asigură selecţia celor trei subspaţii de memorie prin generarea semnalelor de selecţie:

$$\overline{CS_j} = DCD_{j+1}(A_{i+1}, A_i) + \overline{ROMSEL}$$

$$\overline{ROMSEL} = \overline{ROMINH} \cdot A_{19} \cdot A_{18} \cdot A_{17} \cdot A_{16}$$

unde j=0, 1, 2 pentru memoriile de tipul: 2716, 2732, 2764
j=1, 2 pentru memoriile de tipul 27128
respectiv
i = 12 pentru memoriile de tipul 2716
i = 13 pentru memoriile de tipul 2732
i = 14 pentru memoriile de tipul 2764
i = 15 pentru memoriile de tipul 27128

$\overline{CS_i}$ - reprezintă semnalele de selectare ale circuitelor de memorie corespunzatoare

$\overline{ROMINH}$ - este un semnal de pe magistrala de extensie a sistemului care poate inhiba memoria EPROM.

Dacă $\overline{ROMINH} = 0$ memoria EPROM de pe modulul de bază este inhibată, nu poate fi adresată de către microprocesor, $\overline{ROMINH} = 1$ memoria EPROM de pe modulul de bază este accesibilă microprocesorului.

$\overline{ROMSEL}$ - este un semnal care este activ în momentul în care memoria EPROM nu este inhibată şi liniile de adresă specifică o locaţie din spaţiul alocat memoriei EPROM.

Pentru memoria de tip 27128 se consideră A16 = 1 ca intrare în decodificator, deoarece se utilizează numai 64K. În acest caz numai CS1 şi CS2 pot să fie active. Deci programul din memoria permanentă trebuie să fie plasat numai în 4 din cele 6 circuite de memorie.

1.4.4. LOGICA DE GENERARE *BUFFER*-E ACCES PE MAGISTRALĂ

Rolul acestei scheme logice este de a asigura activarea pe magistrala de date a informaţiilor citite din memoria EPROM.

Citirea se poate face pe 16 biţi sau 8 biţi în funcţie de ciclul maşină curent, pe baza valorilor semnalelor $\overline{BHE}$ şi A0.

$\overline{ROMHBE} = \overline{ROMSEL} + \overline{HIGHBE}$ - activare *buffer* date D8:15

$\overline{ROMLBE} = \overline{ROMSEL} + A0$ - activare *buffer* date D0:7

$\overline{HIGHBE} = \overline{BHE} \cdot A0$

Buffer-ele de date sunt orientate dinspre memorie spre magistrala de date de către semnalul MEMR.

Schema detaliata de implementare a memoriei EPROM este prezentată în figura 4.18.

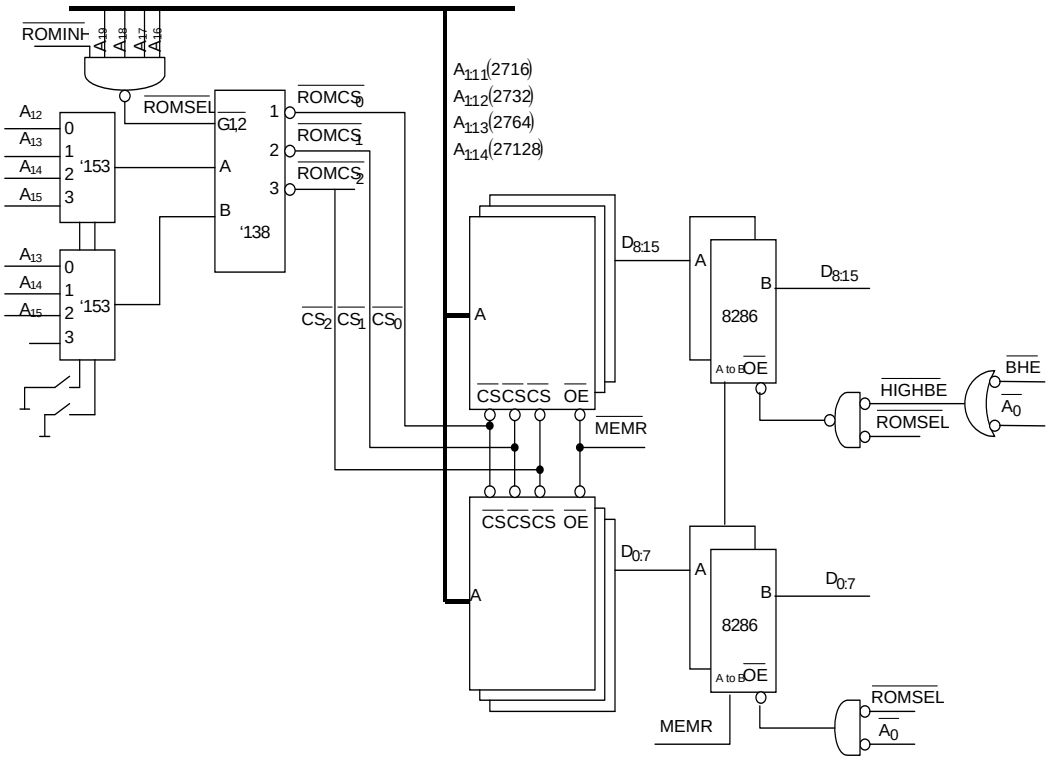


Figura 4.18. Schema detaliată a memoriei EPROM

1.4.5. LOGICA DE MODIFICARE A SPAŢIULUI DE ADRESARE.

Aşa cum s-a prezentat fiecare dintre programele de depanare trebuie să fie lansate în execuţie la punerea sub tensiune a sistemului după ce în prealabil a fost selectat cu ajutorul unor comutatoare.

Pentru aceasta fiecare program de depanare, în momentul selecţiei sale trebuie să fie văzut de către microprocesor în ultimii 256 de octeţi ai spaţiului de adresare de 1Mo, adică în zona 0FFF00H - 0FFFFFH.

Modificarea spaţiului de adresare trebuie facută ca în figura 4.19.

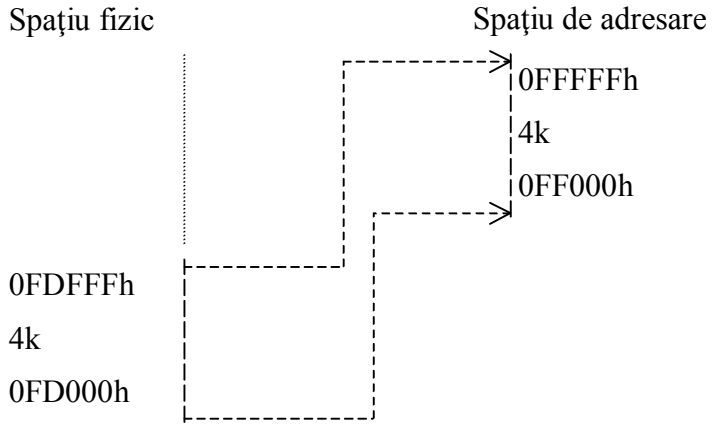


Figura 4.19. Modificarea spaţiului de adresare

Comutarea acestor două zone de memorie se realizează prin modificarea bitului de adresă A_{13} utilizat de logica de decodificare astfel:

$$ADR_{13} = A_{13} \oplus DSW1_7$$

După ce s-au comutat aceste două zone de memorie, de 4K octeţi este necesar ca subzone de 256 octeţi să fie translatate în spaţiul 0FFF00H – 0FFFFFH.

Această translatare se face cu ajutorul comutatoarelor de selecţie a programelor de test: DSW1 modificând liniile de adrese care ajung la circuitele de memorie in modul urmator:

$$ADR_i = A_{i+1} \text{ pentru } 0 \le i \le 6 \text{ , i=13;}$$

$$ADR_i = A_{i+1} \oplus DSW1_{i-4} \text{ pentru } 7 \le i \le 10$$

$$ADR_{12} = A_{13} \oplus DSW1_7$$

unde A_i - sunt liniile de adrese de pe magistrala sistemului,

ADR_i - sunt liniile de adrese ale circuitelor de memorie,

DSW_i – comutator de pe placa de bază (“on” reprezintă 1 logic, “off” reprezintă 0 logic).

Modul în care se translatează programele de depanare ca să ajungă în ultima zonă de 256 de octeţi este prezentat în figura 4.20.

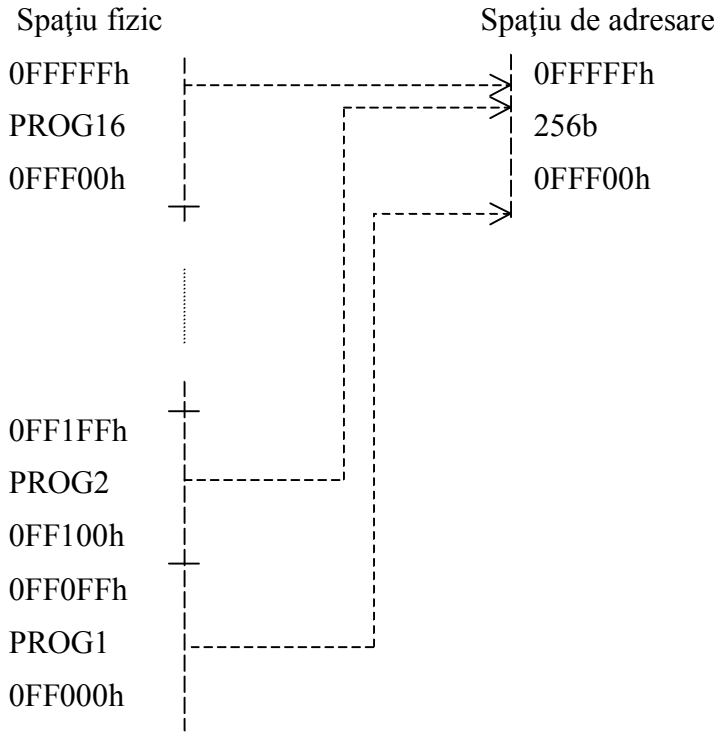


Figura 4.20. Modul de translatare a programelor de depanare în memoria sistemului

1.5. MEMORIA RAM

Restricţiile de proiectare in cazul memoriei RAM sunt:

– capacitatea memoriei RAM este de 256K, realizată cu memoria dinamică de 64Kx1(4164);

– dispune de bit de paritate la nivel de octet 2x(8+1);

– se proiectează o logică ce verifică paritatea la citirea din memorie şi înscrie bitul de paritate BP la o operaţiune de scriere în memorie;

– memoria RAM se mapează la adresa de început a spaţiului de adresare al microprocesorului.

Utilizarea memoriei RAM dinamice prezintă nişte particularităţi derivate din tehnologia de realizare. Se consideră că memoria este organizată intern sub formă matriceală. Pentru a scrie un bit, trebuie să dăm adresa de linie apoi adresa de coloană ca în figura 4.21.

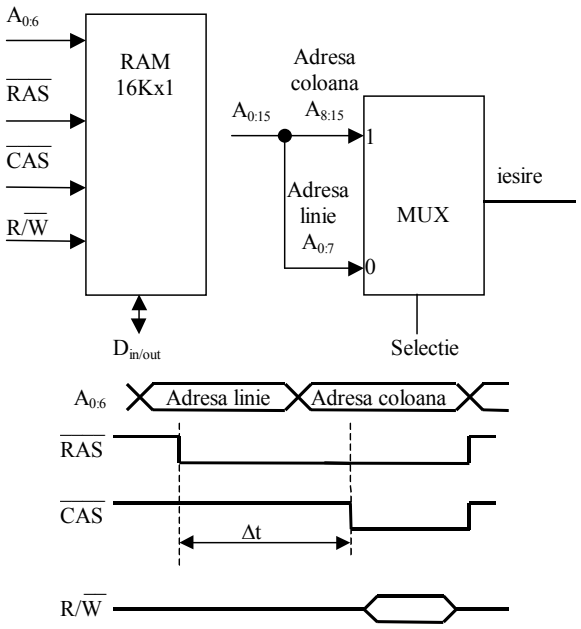


Figura 4.21. Funcţionarea memorie RAM dinamice

Se observă că atunci când semnalul RAS este activ liniile de adresă se interpretează ca adrese de linie, iar atunci când CAS este activ liniile de adresă se interpretează ca adrese de coloană.

O altă particularitate în utilizarea memoriei RAM dinamice este asigurarea operaţiei de reîmprospătare (REFRESH). Aceasta presupune citirea cel puţin a unui bit de pe o linie cu o anumită frecvenţă. La citirea unui bit se face reîmprospătarea întregii linii. În caz contrar conţinutul memoriei se pierde.

Reîmprospătarea trebuie făcută la cel mult 2ms. Pentru realizarea operaţiei de reîmprospătare este suficientă activarea semnalului RAS. Pentru o memorie realizată cu cipuri de 16K(4Kx1) reîmprospătarea se realizează la un interval de 2ms/128=15microsecunde. Adresele de reîmprospătare se realizează cu ajutorul unui registru numărator de adrese, incrementat modulo 128, din 15 în 15 microsecunde (perioadă dată de un timer).

Reîmprospătarea se poate realiza în mai multe moduri:

– cu ajutorul unei logici proprii;

– folosind un circuit specializat INTEL;

– folosirea unui canal DMA;

– folosind un canal DMA-*timer* asociat care generează o cerere către DMA din 15 în 15 microsecunde.

Pentru realizarea unei capacităţi de memorare de 64Kocteţi, folosim 8 capsule, cuplate ca în figura 4.22.

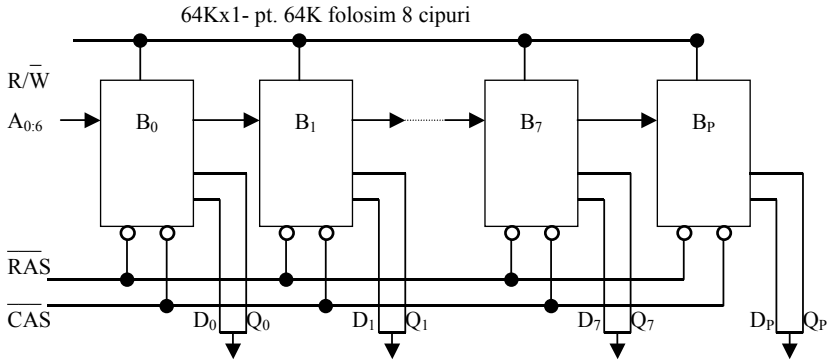


Figura 4.22. Cuplarea capsulelor de memorie RAM dinamica pentru a realiza cuvinte de 8 biţi.

În componenţa lui RAS, CAS, vor intra biţi de adrese pentru selecţia bancului de memorie (LO sau HI) selectaţi cu A17. Citirea se poate face pe 16 biţi, scrierea nu se poate face decât pe 8 biţi (pe magistrala HI sau pe magistrala LO). Semnalul de scriere WRITE se activează numai la blocul selectat efectiv, cu RAS, CAS,

Un caz particular în care acestă funcţionalitate este necesară ar fi atunci când accesul la memoria RAM este realizat de un dispozitiv periferic care funcţionează pe 8 biţi, cum ar fi modulul de acces direct la memorie.

Pentru aceste cazuri se realizează un bit de adresă A^*_8 -A_8 sau A_0.

Daca A_8- microprocesorul controlează memoria.

Daca A_0- DMA-ul controlează memoria.

Dacă $\overline{DACK_0} = 0$ atunci accesul se face de către modulul de acces la memorie şi se utilizează A_0.

Dacă $\overline{DACK0} = 1$ atunci accesul se face de către microprocesor şi se utilizează A_8, figura 4.23.

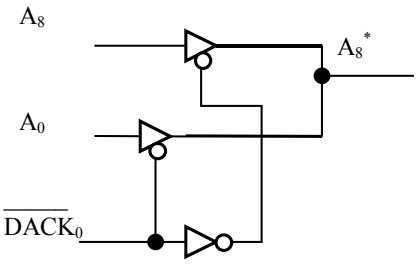


Figura 4.23. Generarea semnalului ADR_8^*

În figura 4.24 se prezintă schema bloc pentru cuplarea memoriei RAM dinamice. Se observă că selecţia multiplexorului pentru comutarea de la adresa de linie la adresa de coloană se obţine din semnalul MEMR/MEMW întârziat printr-un şir de 5 inversoare.

A_{19} şi A_{18} se folosesc pentru selectarea blocurilor de memorie de 256 K.

A_{17} este utilizat pentru generarea semnalului RAS (figura 4.25) şi anume dacă:

A_{17} = 0 generăm $\overline{RAS_0}$

A_{17} = 1 generăm $\overline{RAS_1}$

numai când $\overline{RASEN}$ este activ.

În figura 4.26 se observă modul în care se generează similar semnalele $\overline{CAS_0}$ şi $\overline{CAS_1}$.

Prin scheme logice combinaţionale se obţin semnalele $\overline{RAMLW}$ şi $\overline{RAMHW}$, necesare activării semnalelor de scriere în blocurile de memorie. Cele două inversoare înseriate au rolul de a produce întârzierile rezultate din datele de catalog pentru capsulele de memorie (figura 4.27). Similar se obţin şi semnalele pentru activarea *buffer*-elor pentru citirea datelor din capsulele de memorie (figura 4.28).

Pentru generarea şi verificarea bitului de paritate, se folosesc capsule specializate. Odată generat bitul de paritate se memorează impreună cu cuvântul corespunzător. La citirea cuvântului se face verificarea bitului de paritate stocat cu cel rezultat în urma recalculării lui la citire. Dacă nu corespund se generează un semnal de întrerupere (de obicei nemascabilă).

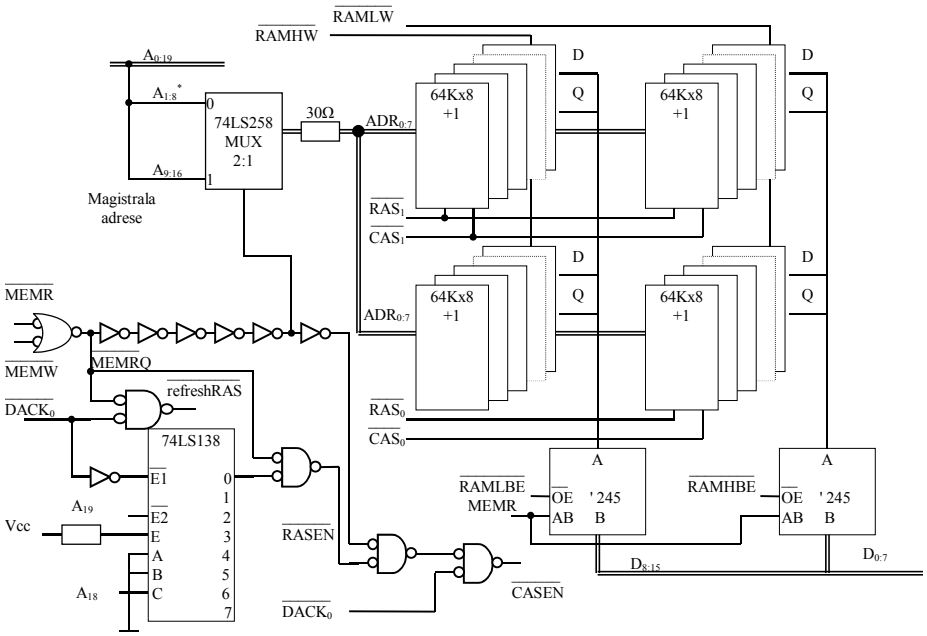


Figura 4.24. Schema dataliată de cuplare a memoriei RAM dinamice

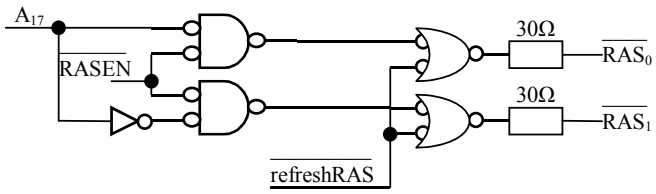


Figura 4.25. Schema de generare RAS

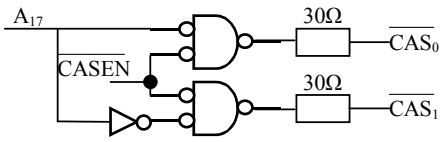
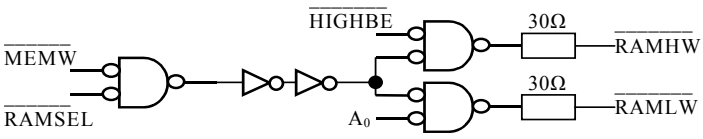


Figura 4.26. Schema de generare CAS



$\overline{HIGHBE}=A_0 . \overline{BHE}$

Figura 4.27. Generare semnale RAMHW şi RAMLW

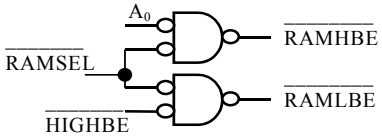


Figura 4.28. Generare semnale RAMHBE şi RAMLW

1.6. INTERFEŢE DE INTRARE/IEŞIRE.

Port-urile de I/O se mapează în funcţie de compatibilitatea pe care dorim să o asigurăm sistemului pe care îl proiectăm. Atunci când se doreşte realizarea unui sistem compatibil cu un IBM-PC XT/AT trebuie ţinut cont de maparea *port*-urilor de intrare / iesire.

1.6.1. *PORT*-URILE DE I/O CONFORM STANDARDULUI IBM XT/AT

În tabelul 4.3 se prezintă maparea *port*-urilor de intrare / ieşire conform standardului IBM XT/AT.

Tabelul 4.3. Maparea *port*-urilor de I/O conform standardului IBM XT/AT

Mapare spaţiu de adresare	Destinaţie	Comentarii
0÷0FFh	Rezervate pentru placa de bază,	Spaţiu de adresare folosit pentru producătorii de plăci de bază
100÷3FFh	Sunt diponibile pe magistrala sistemului fiind repartizate pentru modulele de interfaţare (extensie)	Spaţiul de adresare utilizat în proiectare pentru adăugarea de module noi.
400h÷0FFFh	Nu sunt disponibile pe magistrala sistemului	Spaţiu de adresare folosit pentru producătorii de plăci de bază

Pornind de la specificaţiile de proiectare prezentate la începutul capitolului rezultă că în cele ce urmează ne situăm în spaţiul de adresă 100÷3FFh. Acest lucru necesită o detaliere a spaţiului de adresare pe porţiunea 100÷3FFh pentru a nu utiliza adrese de *port* deja folosite de către IBM.

Acesta se detaliază după cum urmează:

0÷Fh – rezervat modul de acces direct la memorie;

20h÷2Fh – rezervat sistem de întreruperi (8259A). Spaţiul de adresare este incomplet decodificat adresele folosite efectiv fiind 20h şi 21h. Pentru AT limita superioară a spatiului de adresare se extinde până la 3Fh.

40h÷4Fh – rezervat pentru *timer* (8253-5). Pentru AT limita superioară a spatiului de adresare se extinde până la 5Fh.

60h÷63h – rezervat penrtu cuplarea interfeţei paralele. Pentru AT limita superioară a spatiului de adresare se extinde până la 6Fh.

80h÷83h – rezervat pentru registrele de pagină ale modulului de acces direct la memorie.

3F0h÷3F7h – rezervat pentru interfaţa de disc flexibil

378h÷37Fh – rezervat pentru interfaţa paralelă a imprimantei.

3F8h÷3FFh – rezervat pentru interfaţa serială asincronă.

În cele ce urmează se utilizează în mod extensiv decodificarea incompletă a spaţiului de adresare deoarece simplifică mult hardware-ul.

În plus faţă de restrictiile anterioare mai alegem încă o adresă de *port* 0Ah necesară pentru comanda activării/dezactivării întreruperilor nemascabile. De asemenea 0E0h se foloseşte pentru *timer*-ul 2.

În figura 4.29 şi figura 4.30 se prezintă logica de comandă care este implementată în esenţă ca un decodificator de adrese.

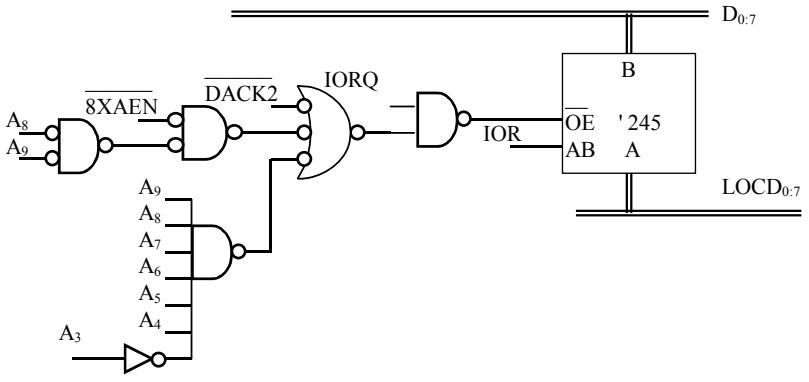


Figura 4.29. Logica de comandă pentru activarea *buffer*-ului magistralei locale

Din figură se observă că se face o distincţie între *port*-urile care sunt proprii magistralei locale şi cele care se află pe modulele de extensie.

De asemenea, în figura 4.30 se face decodificarea pentru *port*-urile folosite pe placa de bază conform mapării prezentate anterior.

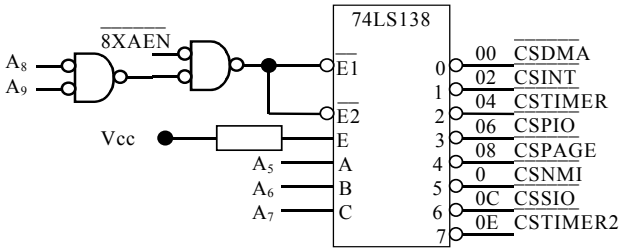


Figura 4.30. Decodificare adrese *port*-uri placa de baza

1.6.2. CUPLAREA INTERFEŢEI PARALELE.

Într-un mod similar se cuplează şi o interfaţă paralelă realizată pe baza circuitul 8255.

Cele trei *port*-uri ale circuitului se folosesc după cum urmează:

– pe *port*-ul A se multiplexează:

  – *switch*-urile de configurare sistem,

  – citirea informaţiei de la tastatură.

– pe *port*-ul B se cuplează:

  – un semnal de activare *timer* TIMER GATE,

  – semnalul de date pentru difuzor,

  – semnalul de comandă pentru activarea / dezactivarea motorului unităţii de disc flexibil,

  – semnalul de activare / dezactivare pentru logica de verificare a parităţii meoriei RAM dinamice ENRAM PCK.

– pe *port*-ul C se cuplează:

  – semnalul prin care se semnalizează eroare de paritate la RAM,

  – citirea semnalului de comandă difuzor.

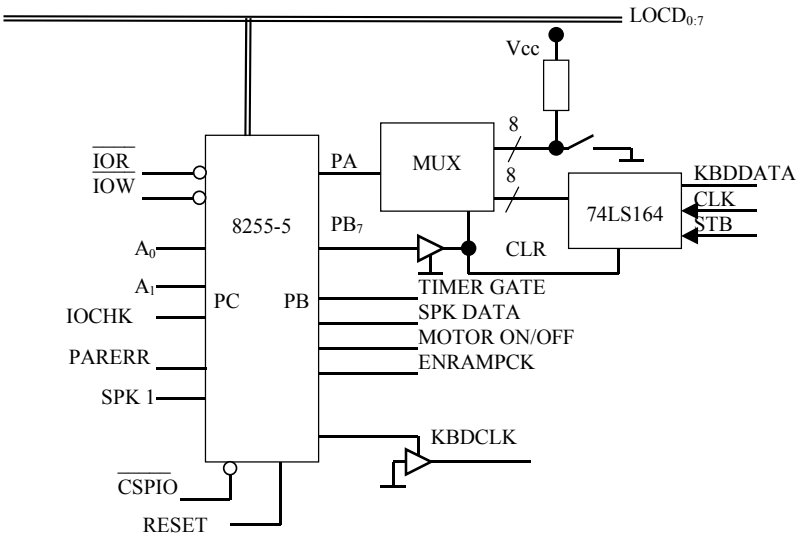


Figura 4.31. Cuplare interfață paralelă

1.6.3. CUPLAREA *TIMER*-ULUI

În figura 4.32 se prezintă cuplarea *timer*-ului la magistrala locală. Se pleacă de la un circuit specializat 8253.

Pornind de la specificaţiile de catalog ale circuitului 8253 se cuplează semnalele de activare GATE0,1 la Vcc pentru a fi activate permanent. Cel de-al treilea semnal de activare a ceasului se cuplează la 8255 asa cum s-a văzut anterior pentru a se putea comanda difuzorul sistemului.

Frecvenţa de 18,432 Mhz s-a ales astfel încât să fie multiplă cu ratele de transfer pentru interfaţa serială.

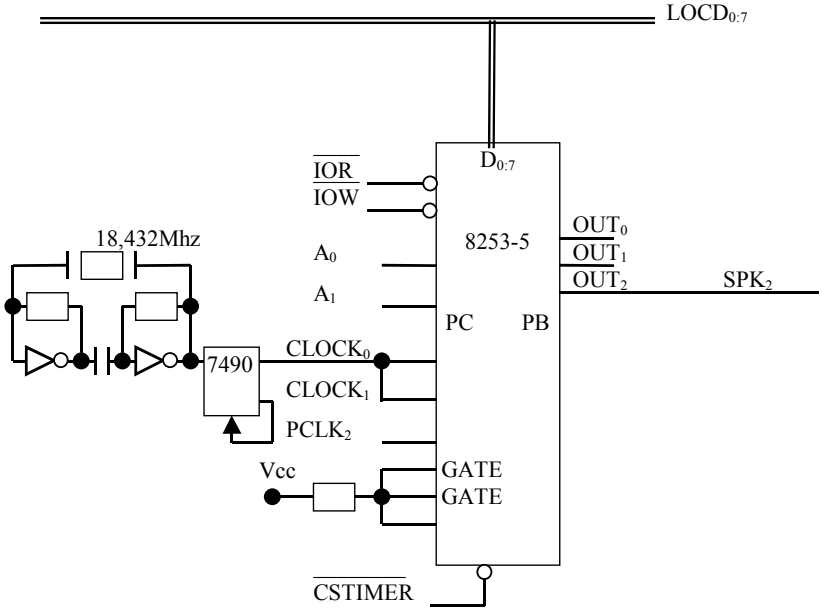
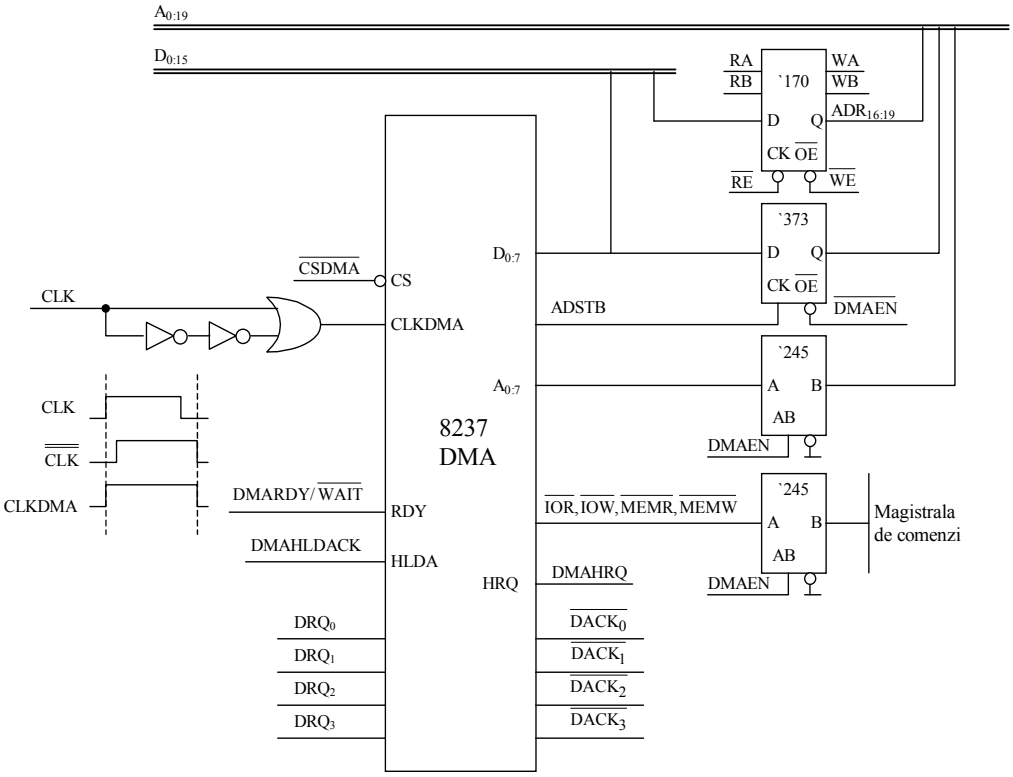
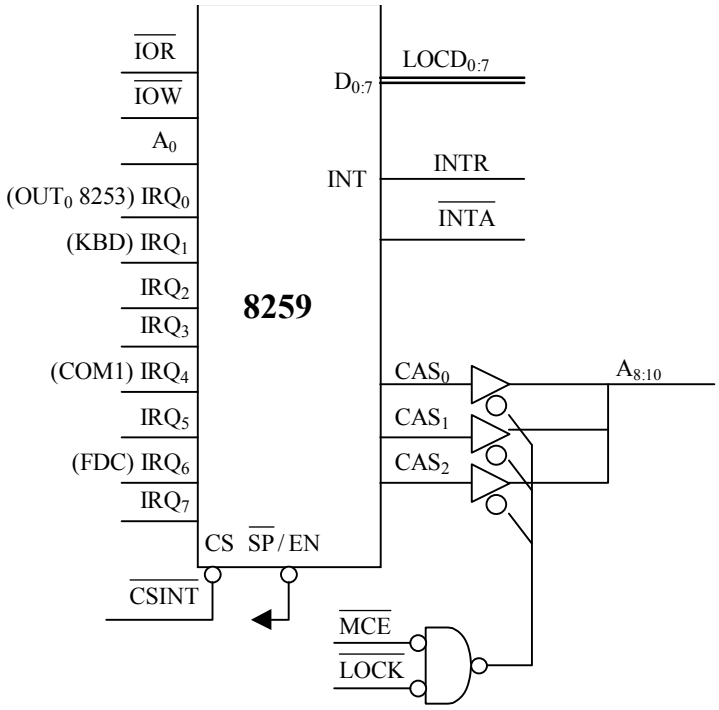


Figura 4.32. Cuplarea circuitelor de ceas de timp real

1.6.4. CUPLAREA CIRCUITULUI DE ACCES DIRECT LA MEMORIE



1.6.5. SISTEMUL DE ÎNTRERUPERI



1.6.6. INTERFAŢA DE DISCURI FLEXIBILE

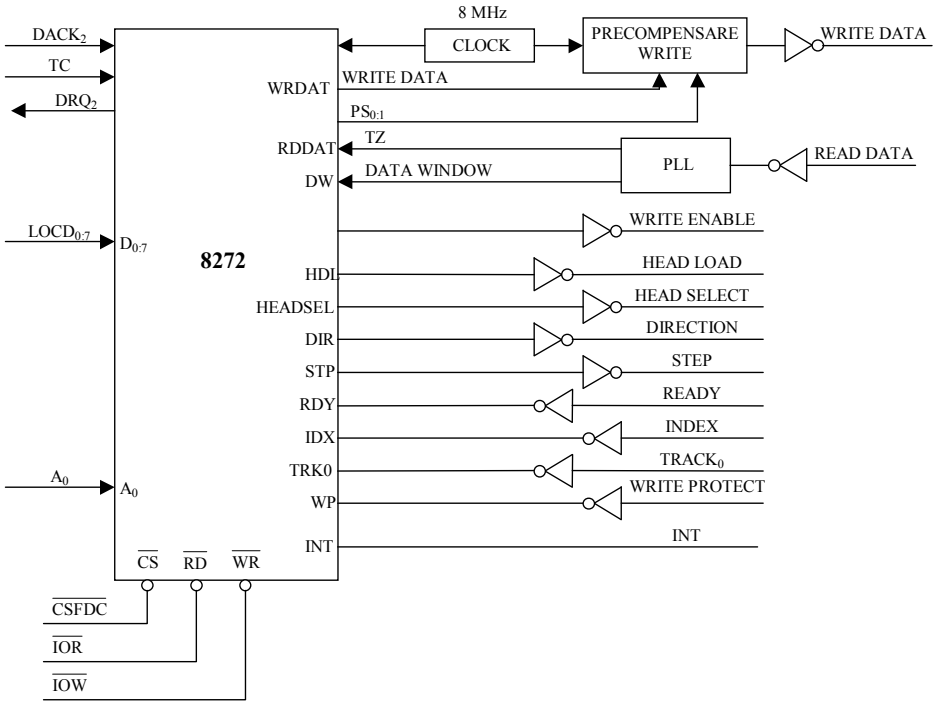


Fig. 4.X. Schema interfeţei de disc flexibil