11 Microprocesorul Intel 8086

11.1 Generalități

Lansat în 1978 de firma *Intel*, se prezintă sub forma unei capsule cu 40 de pini, este realizat în tehnologia HMOS şi are în structură circa 29.000 tranzistoare, pe o suprafață de siliciu de 37 mm^2. Apariția lui a fost urmată la scurt timp de o familie de componente:

Intel 8284 - generator de tact;

Intel 8288 - controler de magistrale;

Intel 8087 - coprocesor aritmetic;

Intel 8089 - coprocesor de intrare-ieşire .

În 1979 apare microprocesorul *Intel* 8088 care păstrează caracteristicile lui 8086 dar cu magistrala de date externă este de 8 biți. A cunoscut o largă utilizare prin includerea sa în numeroase produse ale firmei IBM.

Microprocesorul Intel 8086 este cel mai răspândit microprocesor pe 16 biți; registrele interne şi magistralele de date interne şi cea externă sunt de 16 biți.

Caracteristici tehnice principale :

- multiplexarea în timp a magistralelor de date, adrese şi stări, pentru păstrarea capsulei de 40 de pini;
- magistrala de adrese de 20 de biți, ceea ce permite adresarea unei memorii de capacitate maximă de 1 MB;
- o singură tensiune de alimentare : + 5 Vcc;
- frecvența semnalului de tact: 4 MHz, 5 MHz sau 8 MHz, în funcție de variantă;
- compatibilitate cu limbajul de asamblare al microprocesorului Intel-8080 şi Intel-8085.
- setul de instrucțiuni conține 94 de tipuri de instructiuni, inclusiv operații aritmetice în cod BCD şi operații de înmulțire, împărțire;
- operează cu digiți (cod BCD, 4 biți/digit sau 8 biți/digit), cu octeți (byte), cu cuvinte de 16 biți (word), cu cuvinte duble de 32 de biți (double word), şiruri de caractere de 8 biți (string) şi blocuri de date.
- acoperă o gamă largă de aplicații datorită celor două moduri de lucru: *modul minim* pentru aplicații simple, în care procesorul generează el însuşi semnalele electrice necesare transferului de date cu memoria şi porturile de intrare/ieşire şi *modul maxim*, pentru aplicații complexe, inclusiv sisteme multiprocesor, în care semnalele

de comandă pentru memorie şi porturi sunt generate de un circuit specializat, 8288 (controler de magistrale).

11.2 Structura internă

Microprocesorul Intel-8086 cuprinde două unităţi funcţionale care lucrează asincron şi independent una faţă de cealaltă:

- Unitatea de execuţie EU (*Execution Unit*), care efectuează operaţiile conţinute codificat în instrucţiuni.
- Unitatea de interfaţă cu magistralele (*Bus Interface Unit*), care are rolul de a extrage instrucţiunile din memorie şi de a transfera operanzii între unitatea de execuţie şi memorie sau porturi de intrare/ieşire.

11.2.1. Unitatea de interfaţă cu magistralele BIU (*Bus Interface Unit*)

Realizează conectarea microprocesorului cu exteriorul prin intermediul magistralelor de adrese (20 de linii) şi de date (16 linii). De asemenea, unitatea BIU generează semnalele de comandă pentru realizarea operaţiile de citire şi scriere cu memoria sau cu porturile.

Unitatea BIU realizează extragerea în avans a instrucţiunilor din memorie, pe care le stochează într-un fişier de instrucţiuni de 6 octeţi, care este de fapt o listă de tip FIFO (*First In First Out*).

Dacă în acest fişier sunt cel puţin două locaţii libere şi unitatea de execuţie nu solicită transfer de operanzi cu exteriorul, unitatea BIU va iniţia un ciclu maşină de extragere în avans a unei instrucţiuni din memorie, de la adresa următoare. Se asigură astfel un important câştig de timp, prin suprapunerea execuţiei cu extragerea instrucţiunilor, operaţie care necesită timp de acces la memorie şi timp de transfer.

Instrucţiunile de salt pot modifica succesiunea normală prin comutarea execuţiei la o adresă "de salt"; în această situaţie, unitatea BIU iniţializează fişierul prin ştergerea instrucţiunilor existente şi încărcarea lui cu o nouă secvenţă de instrucţiuni extrase începând cu adresa de salt.

Dacă în timpul execuţiei unei instrucţiuni unitatea EU solicită un operand din memorie, unitatea BIU calculează adresa fizică (20 de biţi) în funcţie de modul de adresare comandat, utilizând registrele proprii şi unitatea aritmetică proprie (UA) şi generează semnalele de comandă pentru transferul operandului către unitatea de execuţie.

11.2.2. Unitatea de execuție EU (*Execution Unit*)

Extrage succesiv instrucțiunile din fişierul de instrucțiuni, le decodifică pe baza unui microprogram rezident şi le execută prin intermediul registrelor de uz general şi unității aritmetice şi logice (UAL).

Dacă pentru execuția unei instrucțiuni este necesar accesul la memorie sau la porturi I/O, unitatea de execuție transmite către BIU o adresă de 16 biți (adresă efectivă sau *offset*) ce va fi utilizată pentru operațiile de transfer.

Unitatea de execuție dispune de un bloc de comandă care coordonează funcționarea unității. În acest bloc există o memorie ROM în care este stocat microcodul de interpretare şi execuție pentru fiecare instrucțiune.

11.2.3. Setul de registre

Registrele sunt specializate pe funcții; ele pot fi grupate în 5 categorii:

- registre de date (AX, BX, CX, DX);
- registre index, pentru accesul în interiorul unui segment (SP, BP, DI, SI);
- registre de segment (CS , DS , SS , ES) ;
- registru indicator de adresă (IP);
- registru de stare (F).

Primele două categorii alcătuiesc registrele de uz general.

Utilizatorul are acces la toate registrele, dimensiunea lor fiind de 16 biți, egală cu dimensiunea magistralei de date.

Registrele de date. Deşi sunt registre de 16 biți, pot fi utilizate ca două registre de 8 biți: registrul superior H (High) şi registrul inferior L (Low). Oricare din registrele de uz general poate fi utilizat în operații aritmetice şi logice dar au şi funcții specifice, care nu pot fi modificate de programator (tab.1).

Tabel 1. Funcțiile specifice ale registrelor de uz general

Registrul	Funcția specifică
AX	Înmulțire, împărțire, intrare / ieşire pe cuvânt
AL	Înmulțire, împărțire, intrare / ieşire pe octet
AH	Înmulțire, împărțire, pe octet
BX	Translatare (adresare indexată)
CX	Contor pentru operații cu şiruri, bucle
CL	Contor pentru deplasări, rotații
DX	Înmulțire, împărțire, intrare / ieşire indirectă

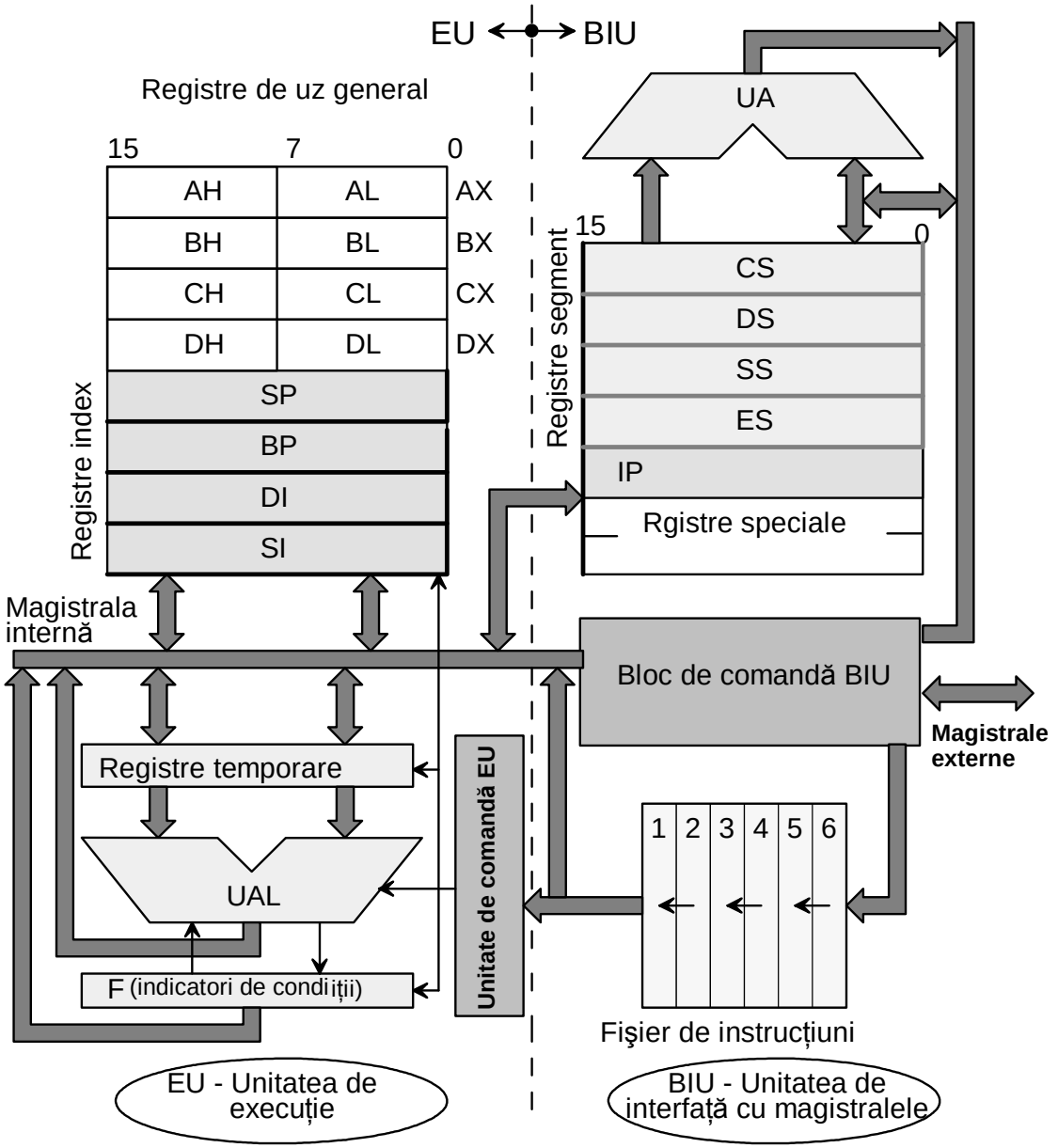

Fig.1 Arhitectura internă a microprocesorului Intel 8086

Registrele de segment

Sunt registre de 16 biți care conțin adresa de bază a unui segment de memorie. Memoria direct adresabilă de 1 MB necesită o magistrală de adrese de 20 biți.

Registrele interne ale procesorului fiind de 16 biți, memoria este divizată din punct de vedere logic în segmente de 64 kB, fiecare segment fiind astfel adresabil cu 16 biți. Procesorul poate să utilizeze simultan atâtea segmente de memorie câte registre de segment posedă.

Adresele de început (de bază) ale celor 4 segmente sunt conținute în cele 4 registre de segment (Cod, Date, Stivă şi de Date suplimentar).

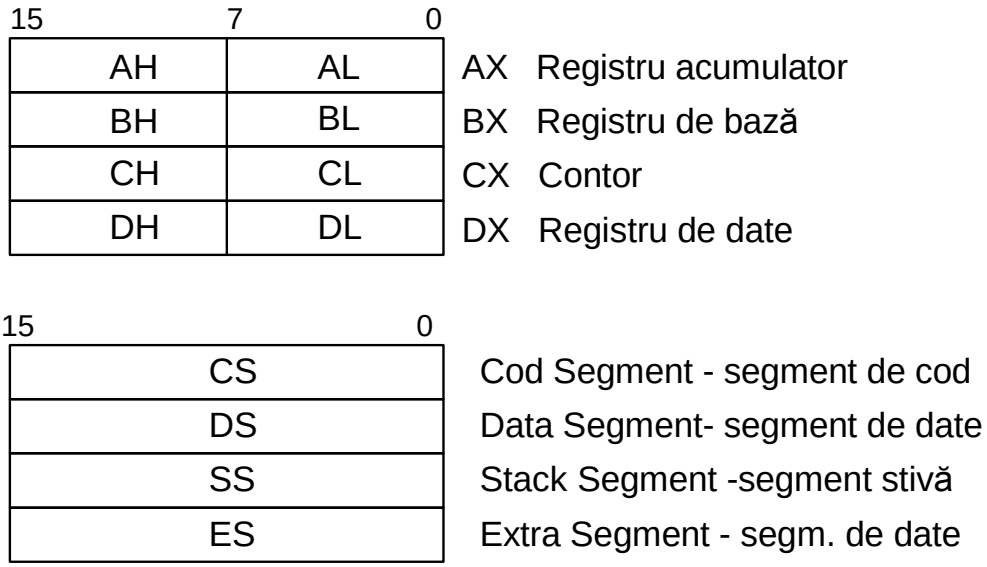

Fig. 2. Registrele de date şi de segment la Intel 8086

Registrul CS conține adresa de început a segmentului de cod unde se află codurile instrucțiunilor.

Pentru a adresa o instrucțiune, microprocesorul combină conținutul registrului segment de cod CS cu al registrului indicator de adresă, IP, obținând o adresă fizică de 20 biți .

Registrul DS conține adresa de început a segmentului de date, SS conține adresa de început a segmentului stivă iar ES conține adresa de început a unui segment de date suplimentar. Segmentul de cod conține instrucțiuni iar ultimele trei segmente de memorie sunt dedicate operanzilor (date).

Registrele pentru accesul în interiorul unui segment.

Împreună cu registrele de date, reprezintă registrele generale; SP şi BP sunt registre indicator iar SI şi DI sunt registre index .

Registrele indicator conțin adresa relativă față de baza segmentului la care se află un operand în cadrul segmentului stivă; SP indică adresa relativă a vârfului stivei în raport cu conținutul registrului SS (baza stivei), iar BP indică adresa relativă a unui operand în cadrul segmentului stivă.

Registrele index conțin adresa relativă față de baza segmentului la care se află un operand în cadrul segmentelor de date sau de date suplimentar, deci în raport DS sau ES; SI şi DI se utilizează în general în cazul operațiilor cu şiruri de date, SI indicând adresa operandului sursă iar DI adresa operandului destinație.

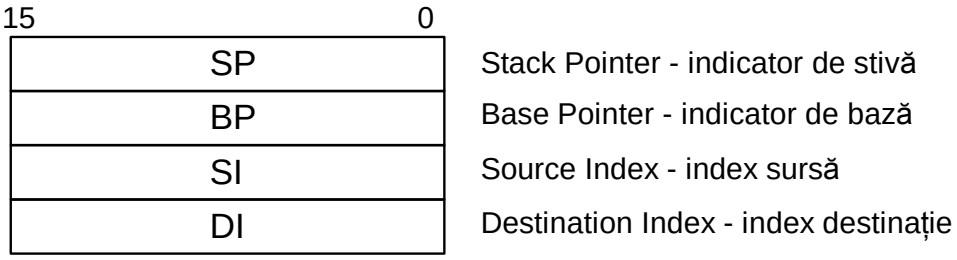

Fig. 3. Registrele indicator şi index la Intel 8086

Registrul indicator de adresă IP (*Instruction Pointer*)

Este similar registrului de adresă PC (*Program Counter*) de la Intel 8080 sau Zilog Z80, indicând adresa instrucțiunii ce se extrage din segmentul de cod. Spre deosebire de PC, IP nu conține adresa fizică a instrucțiunii ci adresa relativă față de baza segmentului de cod. Conținutul lui IP se combină cu cel al lui CS şi astfel se obține adresa fizică a instrucțiunii. După transferul fiecărui octet în fişierul de instrucțiuni, conținutul lui IP creşte cu o unitate: IP = IP + 1, fiind astfel pregătit pentru adresarea octetului următor din segmentul de cod.

Registrul de stare sau al indicatorilor de condiții F (*Flags*)

Acest registru face parte din unitatea aritmetică şi logică UAL (sau ALU - *Arithmetic and Logic Unit*). Deşi este un registru de 16 biți, doar 9 sunt semnificativi, reprezentând indicatorii de condiții ai procesorului 8086.

Indicatorii se poziționează în "0" sau "1" după efectuarea unei operații aritmetice, logice sau după o instrucțiune de control; pot fi testați prin intermediul instrucțiunilor condiționale şi se pot lua decizii în funcție de valoare lor.

CF: (*Carry Flag*) indicatorul de transport; se activează (în "1" logic) la apariția unui bit de transport (depăşirea lungimii normale a rezultatului) la adunare sau de împrumut la scădere; este asociat rezultatului unei operații aritmetice sau logice şi se comportă ca al noulea bit (b8) al acestuia.

PF: (*Parity Flag*) indicator de paritate; se activează (în "1" logic) când rezultatul unei operații aritmetice sau logice este un octet cu număr par de unități.

AF: (*Auxiliary Flag*) indicator de transport la jumătate; se activează când apare transport sau împrumut în operațiile aritmetice şi logice între biții 3 şi 4.

ZF: (*Zero Flag*) indicator de zero; se activează (în "1" logic) dacă rezultatul unei operații aritmetice sau logice este zero.

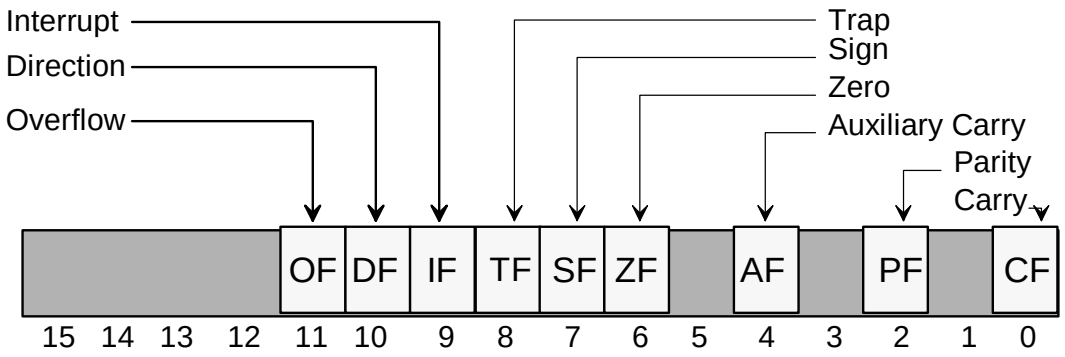

Fig.6. Registrul indicatorilor de condiții la microprocesorul Intel 8086/8088

SF: (*Sign Flag*) indicator de semn; conține bitul cel mai semnificativ al rezultatului, deci copiază bitul de semn; va fi "1" pentru rezultat negativ şi "0" pentru un rezultat pozitiv.

OF: (*Overflow Flag*) indicator de depăşire; se activează (în "1" logic) când apare depaşirea capacității registrului ce conține rezultatul, ca urmare a unei operații aritmetice cu operanzi cu semn; va fi "1" când apare transport sau împrumut în/din rangul de semn.

TF: (*Trap Flag*) indicator pentru modul de lucru "pas cu pas"; dacă este poziționat în "1" logic, microprocesorul nu mai lucrează la viteza dată de semnalul de tact ci în ritmul impus de utilizator. Acest mod de lucru este deosebit de util la depanarea programelor.

IF: (*Interrupt Flag*) indicator pentru controlul întreruperilor; dacă este poziționat în "1" logic (prin instrucțiunea corespunzătoare), sunt validate cererile de întrerupere de tip INTR (mascabile); invalidarea acestora se face prin poziționarea lui IF în "0", caz în care cererile INTR nu vor fi luate în considerație.

DF: (*Direction Flag*) indicator de "direcție", are efect în operațiile cu şiruri. Dacă este poziționat în "1", după fiecare transfer, adresa operandului din şir se incrementează (creşte cu 1) iar dacă este poziționat în "0" adresa se decrementează.

Observație: indicatorii DF, IF şi CF pot fi poziționați în "0" sau "1" prin instrucțiunile de control specifice:

CLC (*Clear Carry*): CF = 0 **STC** (*Set Carry*): CF = 1
CLD (*Clear Direction*): DF = 0 **STD** (*Set Direction*): DF = 1
CLI (*Clear Interrupt*): IF = 0 **STI** (*Set Interrupt*): IF = 1

11.2.4. Terminale şi semnificația lor

Prin multiplexarea în timp a magistralei de date cu cea de adrese la aceleaşi terminale precum şi prin dublarea rolului unor terminale ale

microprocesorului în funcţie de cele două moduri de lucru ale sale, a fost posibilă închiderea sa într-o capsulă cu 40 de terminale.

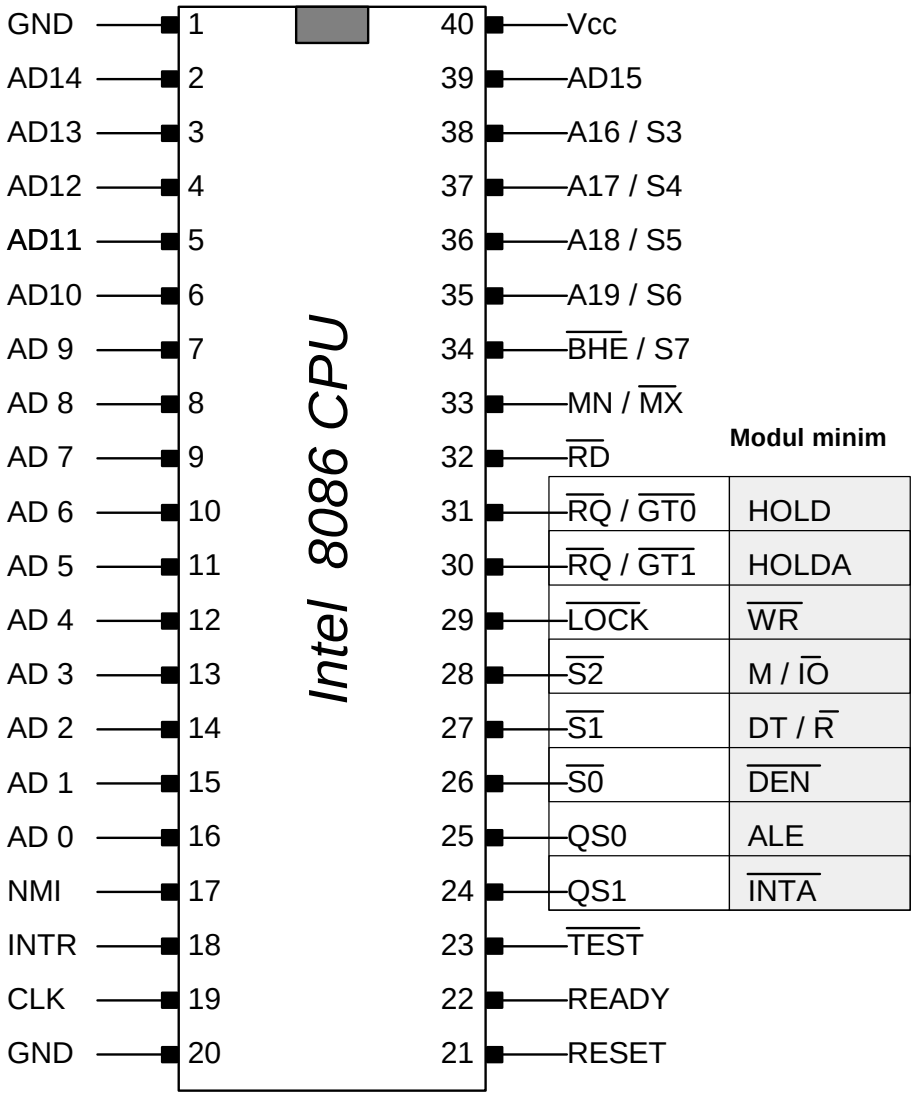

Fig.3 Intel 8086 - Configuraţia semnalelor la pini

Terminalele cu acelaşi rol în ambele moduri de lucru sunt :

AD_0 - AD_{15} : linii bidirecţionale cu trei stări (*three state*), constituie magistrala de adrese şi date multiplexate. Pe durata stării T_1 a ciclului maşină, pe cele 16 linii se încarcă o adresă de memorie sau pentru porturi I/O iar pe durata stărilor T_2, T_3, T_w şi T_4 liniile constituie magistrala de date.

A_{16} /S_3 , A_{17} /S_4 , A_{18} /S_5 , A_{19} /S_6 : ieşiri cu trei stări; în starea T_1 reprezintă cei mai semnificativi 4 biţi ai magistralei de adrese iar în stările T_2 - T_4 reprezintă informaţii de stare: S_3, S_4 indică registrul segment utilizat în calculul adresei fizice, S_5 copiază indicatorul de întreruperi (IF) iar S_6 = 0.

$\overline{\text{BHE}}$ /S_7 (*Bus High Enable*) : ieşire cu trei stări; în timpul stării T_1 se activează când are loc un transfer pe octetul superior al magistralei de date, validând acest transfer, iar în stările T_2 - T_4 este bit de stare.

$\overline{\text{RD}}$ (*Read*): ieşire cu trei stări, activă (în "0") atunci când microprocesorul execută un ciclu maşină de citire memorie sau porturi.

S4	S3	Registrul segment implicat în adresare
0	0	ES
0	1	SS
1	0	CS
1	1	DS

READY : intrare activă în "1", pentru sincronizarea procesorului cu memoria sau porturile I/O, mai lente.

INTR (*Interrupt Request*) : intrare activă în "1" reprezentând cerere de întrerupere mascabilă.

$\overline{\text{TEST}}$: intrare activă în "0", utilizată în cazul instrucțiunii ESC (Escape) care permite altui procesor să extragă instrucțiuni sau operanzi din memorie, din segmentele curente ale lui 8086/8088. Instrucțiunea ESC poate iniția o subrutină executată de un procesor concurent, ca de exemplu 8087 - coprocesor aritmetic; în timpul execuției subrutinei, procesorul de bază execută programul curent până când devine necesar rezultatul subrutinei. Dacă acesta întârzie, procesorul de bază intră în stare de aşteptare până când TEST devine inactiv.

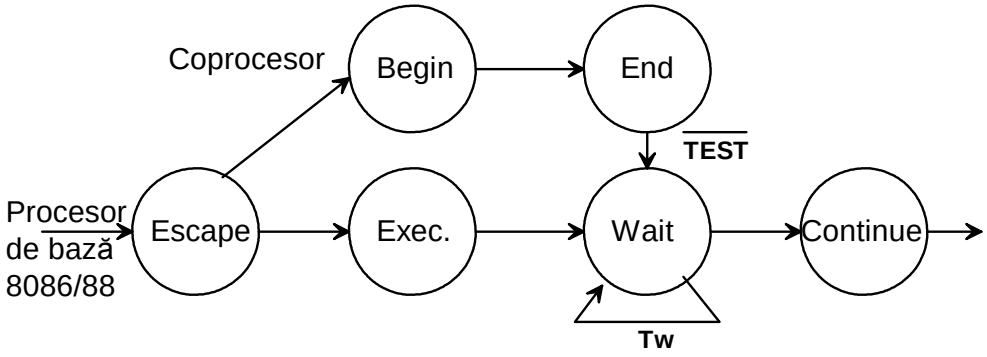

Fig. 4 Rolul semnalului TEST în sisteme multiprocesor

NMI (*Not Mascable Interrupt*) : intrare activă în "1" reprezentând cerere de întrerupere nemascabilă (care nu poate fi invalidată prin soft). Este utilizată pentru intervenția oparatorului sau pentru evenimente a căror luare în considerație nu suportă amânare (avertizare asupra unei iminente fluctuații a tensiunii de alimentare, intervenție *watch dog*).

RESET: intrare activă în "1", pentru inițializarea microprocesorului. Se aplică automat la apariția tensiunii de alimentare sau în timpul

funcționării, de către operator. Procesorul întrerupe toate activitățile până când semnalul devine inactiv. Se efectuează următoarele inițializări interne:

Indicatorii de condiții: toți în "0" IP = 0000 H
CS = FFFF H DS = 0000 H
SS = 0000 H ES = 0000 H
Fişierul de instrucțiuni: vid;
Întreruperile mascabile: invalidate.

Prima instrucțiune ce se va executa după inițializare va fi cea de la adresa FFFF0 H.

De regulă, la adresa FFFF0 se află o instrucțiune de salt către prima instrucțiune a sistemului de operare.

CLK : intrare pentru impulsurile de tact furnizate de un circuit specializat (generator de tact, tipic Intel 8284) cu frecvența uzuală de 8 MHz, nivel TTL şi factor de umplere 1/3.

MODUL MINIM

Se instalează dacă intrarea de mod este la nivel ridicat, **MN/MX**=1.

M/IO ieşire cu trei stări (*Memory / Input-Output*); determină o operație de citire sau scriere asupra memoriei sau porturilor. Dacă este "1" logic, se execută un ciclu maşină de acces la memorie iar dacă este "0" logic se execută un ciclu maşină de transfer cu porturile de intrare / ieşire.

WR: ieşire cu trei stări (*Write*) activă în "0" logic; determină o operație de scriere în memorie sau în porturi.

INTA: ieşire cu trei stări (*Interrupt Acknowledge*), activă în "1" logic, când microprocesorul execută un ciclu maşină de acceptare întrerupere.

ALE: ieşire activă în "1" (*Address Load Enable*) care se activează când pe magistrala multiplexată este încărcată o adresă.

DT/R : ieşire cu trei stări (*Data Transmission Reception*), care indică sensul transferului pe magistrala de date. Dacă este "0" logic, magistrala este orientată către microprocesor (*Reception*) iar în caz contrar, către sistem (*Transmission*).

DEN: ieşire cu trei stări (*Data Enable*) care validează transferul datelor către microprocesor.

HOLD: intrare pentru cereri de cedare magistrale. Un dispozitiv inteligent extern solicită controlul total asupra magistralelor în vederea accesului direct la memorie (tehnică *DMA - Direct Access Memory*).

HLDA: ieşire, răspuns la cererea HOLD, confirmând acceptarea acestei cereri, după trecerea magistralelor de adrese, date şi control în starea SIR (starea de înaltă impedanță sau starea "a treia").

Ultimele opt terminale prezentate au alte funcţii în modul maxim.

MODUL MAXIM

Se instalează dacă intrarea de mo este la nivel coborât, **MN/MX**=0.

RQ/GT0, RQ/GT1 (***Request/Grant Lines***) : linii bidirecţionale utilizate pentru cedarea magistralelor în urma unui dialog cerere - acceptare - renunţare.

În modul maxim, semnalele HOLD şi HOLDA evoluează în două semnale mai complexe, RQ/GT0, RQ/GT1, ce pot servi utilizării în comun a magistralelor de către 8086/8088 şi alte două procesoare.

Cererea şi cedarea magistralelor necesită trei faze distincte: cererea, alocarea, eliberarea. Un procesor cere pe o linie RQ/GT controlul magistralelor şi pe aceeaşi linie CPU 8086 confirmă trecerea lor în starea SIR, deci cedarea lor. În această situaţie, unitatea BIU este deconectată de la magistrale iar EU execută instrucţiuni din fişierul intern până la golirea acestuia sau până când o instrucţiune necesită acces la magistrale. Când procesorul extern termină operaţiile cu magistralele, transmite pe aceeaşi linie, RQ/GT = 1, cu semnificaţia că CPU 8086 poate să preia controlul asupra magistralelor.

Linia RQ/GT0 are prioritate faţă de RQ/GT1, în cazul în care apar cereri simultane pe cele două linii.

S2 , S1 , S0 : ieşiri cu trei stări care exprimă codificat tipul de ciclu maşină ce va fi va executat, conform tabelului; semnalele se aplică circuitului specializat Intel 8288 care generează semnale de comandă pentru memorie şi porturi I/O, corespunzătoare ciclului maşină curent.

S2	S1	S0	Tipul ciclului maşină
0	0	0	Acceptare întrerupere
0	0	1	Citire port I/O (intrare date)
0	1	0	Scriere port I/O (ieşire date)
0	1	1	Halt (oprire)
1	0	0	Aducere instrucţiune din memorie
1	0	1	Citire memorie
1	1	0	Scriere memorie
1	1	1	Combinaţie neutilizată

LOCK : ieşire cu trei stări, activă în "0", cu semnificaţia că magistralele nu pot fi cedate pe durata LOCK = "0", deoarece se află în execuţie o instrucţiune care nu poate fi întreruptă. Semnalul este activat de prefixul LOCK al unei instrucţiuni oarecare şi rămâne activ pe toată

durata execuţiei instrucţiunii. Prefixul constă într-un octet plasat înaintea octeţilor ce definesc instrucţiunea.

QS0 , QS1 : ieşiri (*Queue Status Lines*), care indică dispozitivelor externe tipul de informaţie preluată de unitatea de execuţie din fişierul de aşteptare în starea anterioară. Informaţia este utilă pentru un coprocesor, în cadrul instrucţiunii ESC, în vederea utilizării magistralelor la extragerea unui operand din memorie.

QS1	QS0	Semnificaţie
0	0	Fără citire din fişier
0	1	Octetul preluat a fost primul octet al instrucţiunii
1	0	Fişierul este gol
1	1	Octetul preluat nu a fost primul octet al instrucţiunii

11.3 Implementarea stivei

Stivele sunt formate în memoria RAM şi sunt adresabile cu registrul segment SS şi registrul pointer SP. Un program poate utiliza mai multe stive, fiecare cu lungimea maximă de 64 KB. Registrul SS conţine adresa de bază a stivei iar SP conţine adresa relativă (deplasamentul sau offset-ul) faţă de bază, a vârfului stivei. Stivele au locaţii de 16 biţi şi funcţionează ca liste LIFO (*Last In First Out*), adică ultimul operand introdus în stivă este primul care poate fi extras.

Un cuvânt este salvat în stivă la adresa relativă (SP-2); citirea din stivă se face prin copiere de la adresa (SP), după care se face actualizarea registrului pointer SP: (SP+2). Locaţiile din stivă sunt de 16 biţi şi ca urmare operanzii care se introduc şi se extrag în/din stivă sunt cuvinte de 16 biţi iar adresele a două locaţii consecutive diferă prin 2 (în fiecare locaţie sunt doi octeţi).

Se pot introduce în stivă operanzi de 16 biţi din registre cu excepţia lui CS şi operanzi de 16 biţi din memorie. Extragerea din stivă se face prin copiere într-un registru de 16 biţi, cu excepţia lui CS, sau în memorie.

Salvarea în stivă se face cu instrucţiunea PUSH iar extragerea din stivă cu POP. Decrementarea registrului SP la introducere şi incrementarea sa la scoatere se fac automat. Programatorul trebuie să fixeze doar baza stivei în SS şi limita superioară a stivei, în SP. Toate operaţiile cu stiva se realizează apoi foarte simplu, prin instrucţiunile PUSH şi POP. Adresele la care se fac transferurile se calculează automat şi nu prezintă nici un interes. Tot ceea ce contează este ordinea în care se

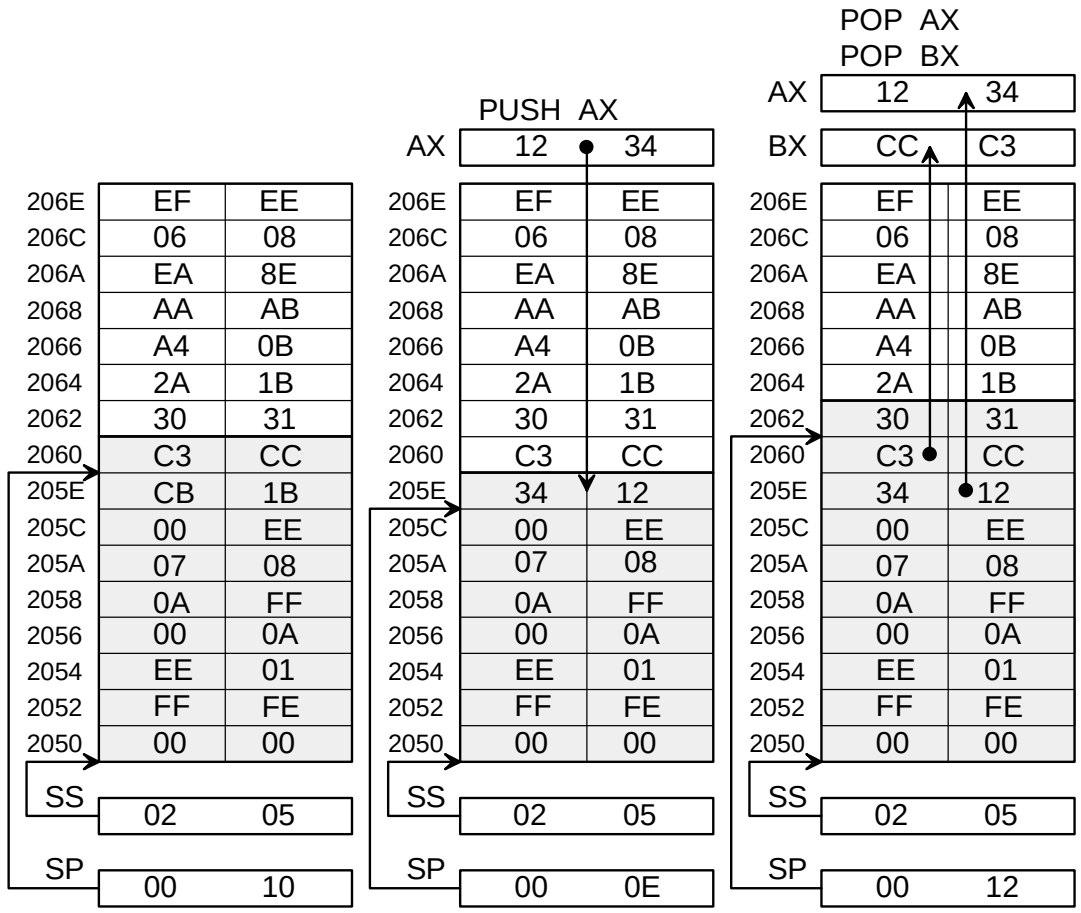

Fig.5 Funcționarea stivei: situația curentă (stânga), introducerea în stivă (mijloc) şi extragerea din stivă (dreapta).

face introducerea în stivă, deoarece la extragere aceasta se inversează: *''cel din urmă va fi cel dintâi''*. Calculul adresei se face astfel:

SS: 0 2 0 5 0 +
SP: 0 0 1 0 = 0 2 0 6 0 H = 2060 H

- *De ce sunt necesare stivele ?*

- Stivele sunt strict necesare în lucrul cu subrutine (proceduri şi funcții), când trebuie eliberate registrele interne în vederea apelării unei subrutine care va încărca registrele cu propriile sale date. Conținutul registrelor este însă necesar la revenirea din subrutină. De aceea eliberarea registrelor se face prin salvarea lor în stivă, într-o anumită ordine şi refacerea lor din stivă la revenirea din subrutină în programul apelant.

- *Nu putem folosi memoria RAM normală pentru salvarea registrelor ?*

- Ba da, dar conținutul registrelor interne este atât de important încât acesta trebuie pus la loc sigur, unde accesul să fie simplu, pentru a nu

complica inutil textul programului, dar controlat de instrucţiuni speciale, pentru evitarea erorilor de program.

11.4 Organizarea şi adresarea memoriei

Pentru microprocesorul Intel 8086, memoria este un şir de 1.048.576 octeţi, (1Megaoctet). Toate registrele sale interne de adresare fiind de 16 biţi, nu se pot transfera intern adrese mai lungi de 16 ranguri binare, adică nu se pot adresa spaţii de memorie mai mari de 64 kB. Pentru acoperirea unui spaţiu de memorie de 16 ori mai mare (64 kB x 16 = 1 MB), acesta este divizat logic (în mod virtual) în segmente de maxim 64 kB.

Microprocesorul poate lucra simultan cu 4 segmente de memorie, fiecare cu dimensiunea între 16 octeţi şi 64 kiloocteţi, adresele lor de început fiind fixate în cele 4 registre segment .

În utilizarea oricărei locaţii de memorie se folosesc două feluri de adrese: *fizică* şi *logică*. Adresa fizică este de 20 de biţi şi identifică în mod unic fiecare byte din spaţiul memoriei între 00000 H şi FFFFF H. Orice schimb de informaţie între CPU şi memorie utilizează adresa fizică, pe care o transmite prin magistrale de adrese.

O adresă logică este formată din doi parametri: *adresa de bază* şi valoarea de *offset*; pentru orice locaţie, adresa de bază indică adresa primului octet din segmentul de memorie iar valoarea de *offset* arată distanţa în octeţi de la adresa de bază la locaţia adresată. Adresa de bază şi *offset*-ul sunt de 16 biţi fiecare. Primul octet dintr-un segment de memorie are offsetul "0000".

Segmentele de memorie sunt blocuri care se pot suprapune; nu există restricţii de delimitare a segmentelor în spaţiul adreselor (fig. 6), singura restricţie fiind ca adresa de început a unui segment sa fie multiplu de 16, adică să aibă ultimii 4 biţi de valoare "0".

Pot exista segmente adiacente (A, B), segmente parţial suprapuse (B, C şi D, E) sau segmente disjuncte (C, D). Definirea unui segment logic se face prin încărcarea adresei de început într-un registru segment; nu se specifică în nici un fel cât de mare este segmentul dar nu pot fi accesaţi decât primii 64 ko, deoarece *offset* - ul este pe 16 biţi.

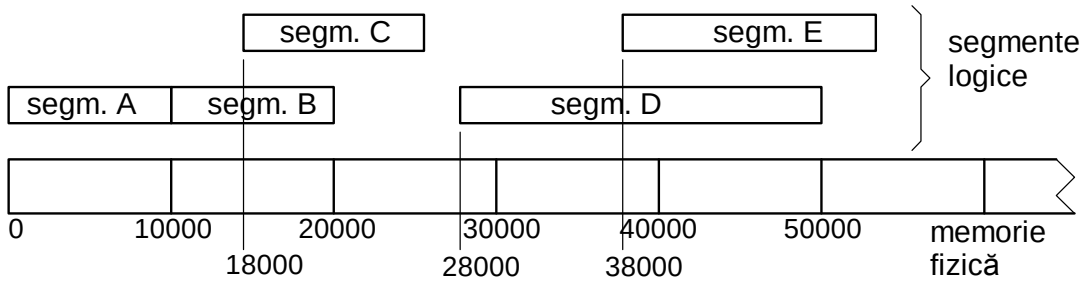

Fig.6 Relația dintre segmentele logice şi memoria fizică

Avantajele segmentarii memoriei sunt :

- Este facilitată programarea modulară; fiecare modul de program poate ocupa unul sau mai multe segmente şi poate fi dezvoltat independent față de celalalte.
- Permite scrierea programelor independent de adresele de memorie în care vor fi stocate, sau poziția lor în memorie poate fi modificată (realocarea dinamică). Pentru aceasta, este necesar ca programele să nu afecteze conținutul registrelor segment şi să nu facă referire la locații din afara segmentului curent; utilizând instrucțiuni de modificare a conținutului registrelor segment, un program poate fi mutat oriunde în memorie.
- Permite deservirea "simultană" a mai multor periferice; de exemplu, un program editor de texte care lucrează cu mai multe imprimante poate transmite câte un bloc de 64 kB fiecărei imprimante care solicită intrarea în "serviciu" prin încărcarea registrului ES cu baza segmentului de date unde se află textul corespunzător imprimantei ce solicită date.
- Se pot utiliza mai multe stive; schimbarea stivei se face prin plasarea adresei de început în registrul segment SS şi a adresei vârfului stivei (limita de sus) în registrul pointer SP.

Dezavantajele segmentării memoriei:

- Limitarea lungimii programelor la dimensiunea maximă a segmentelor.
- Adresa fizică se obține din doua adrese logice (bază şi offset), ceea ce face necesară o operație suplimentară de adunare pe 16 biți şi implicit o unitate aritmetică destinată calculului adesei.

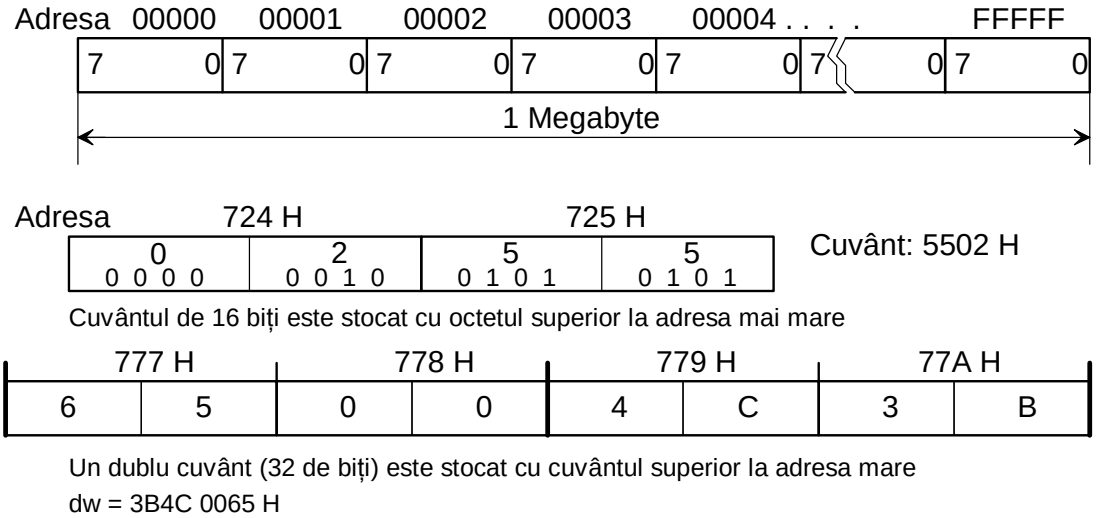

Fig.7 Organizarea informației în memorie

Conform convențiilor firmei Intel cuvântul de 16 biți este memorat în locații consecutive de 8 biți, cu octetul inferior (b_0 - b_7) la adresa de valoare mai mică (*adr*) şi cu octetul superior (b_7 - b_{15}) la *adr+1,* iar dublul cuvânt este memorat cu cuvântul inferior (b_0 - b_{15}) la *adr, adr+1* şi cu cuvântul superior (b_{16} - b_{31}) la *adr+2, adr+3* (fig.7). Adresarea se poate face la nivel de octet dar şi la nivel de cuvânt, adică doi octeți consecutivi, fără restricții cu privire la adresa de început a operanzilor cuvânt.

Microprocesorul generează cei 20 de biți ai adresei fizice prin adunarea adresei de bază cu adresa efectivă (deplasament sau offset), fig.8. Conținutul registrului segment este de 16 biți; pentru a obține adresa de bază a segmentului de memorie, de 20 de biți, se completează conținutul registrului segment cu 4 biți de valoare 0.

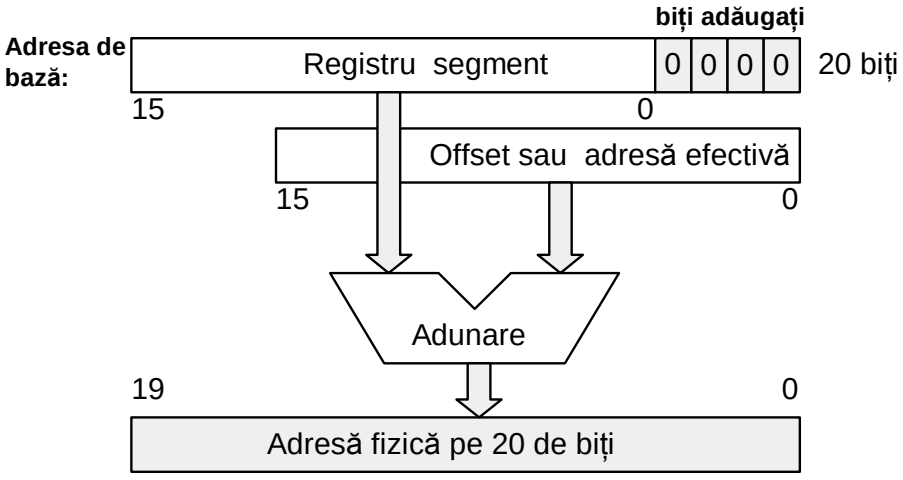

Fig.8 Calculul adresei fizice din adresa de bază şi offset

Două adrese fizice consecutive care se termină cu 0, cuprind între ele 16 locații. Dacă DS=1234 H, ES=1235 H, rezultă că segmentul de date adresat cu DS conține 16 octeți: 12340, . . . ,1234F, iar segmentul de date adresat cu ES începe la adresa de bază 12350 H. De aceea lungimea minimă a unui segment este de 16 locații.

Pentru obținerea adresei fizice se combină un registru segment cu un registru index (SI, DI), cu un registru pointer (SP, BP, IP, BX) sau cu un cuvânt de 16 biți specificat în instrucțiune (adresă efectivă, EA). Nu este posibilă orice combinație între un registru segment şi un altul care furnizează offset - ul.

Registrele segment (care conțin câte o adresă de bază fiecare) permit adresarea imediată a 4 segmente de memorie.

Programele obțin acces la un segment de cod şi 3 segmente de date din orice zonă de memorie, prin încărcarea registrelor segment cu adresele de început ale segmentulor dorite.

În tabelul de mai jos sunt date toate combinațiile posibile.

Tipul referinței la memorie	Registrul segment utilizat	Registrul segment alternativ	Sursa adresei de offset
Extragere instrucțiuni	CS	-	IP
Operații cu stiva	SS	-	SP
Şir de octeți - destinație	ES	-	DI
Şir de octeți - sursă	DS	-	SI
Variabile de program	DS	CS, ES, SS	EA
Adresare cu BP ca bază de offset	SS	CS, DS, ES	EA
Adresare cu BX ca bază de offset	DS	CS, ES, SS	EA

Orice aplicație va defini şi utiliza segmentele proprii. Registrele de adresare curentă (segment) asigură în general ca spațiu de lucru: 64 kbyte pentru coduri, 64 kbyte pentru stivă şi 128 kbytes pentru stocarea datelor. Multe aplicații pot fi srise prin simpla inițializare a registrelor segment (şi apoi le poți uita !). Structura segmentată a memoriei descurajează programele foarte lungi, monolitice.

Instrucțiunile sunt totdeauna extrase cu CS şi IP.

Operanzii din stivă sunt obținuți cu SS şi SP, care conține offset - ul față de adresa de bază din SS.

Majoritatea variabilelor (din memorie) sunt obținute cu DS - curent, dar un program poate informa BIU cu privire la segmentul în care se află

anumite variabile. Diferența față de adresa de bază este calculată de EU; calculul se bazează pe modul de adresare specificat în instrucțiune; rezultatul calculului se numeşte **adresa efectivă a operandului** (EA).

Şirurile de octeți sunt adresate diferențiat, în funcție de variabile.

Operandul sursă din şir este "citit" cu DS dar poate fi specificat şi alt registru segment. Offset-ul este furnizat de SI (registrul index sursă).

Operandul destinație din şir se obține cu ES iar offset-ul se obține din DI (registrul index destinație). Operațiile cu şiruri se execută sub controlul instrucțiunilor specifice, care modifică automt registrele SI şi DI, după cum lucrează cu octeți sau cu cuvinte de 16 biți.

Când este folosit BP ca registru pointer (sursă de offset) într-o instrucțiune, variabila este citită cu SS. Registru BP permite un mod convenabil de adresare pentru datele din stivă; BP poate fi de altfel utilizat la accesarea datelor şi cu CS, DS, ES.

În adresarea memoriei sunt facilitate modurile frecvente de adresare.

Memoria fizică de 1 MB poate fi divizată în două blocuri de câte 512 kB fiecare: blocul locațiilor cu adrese pare şi blocul locațiilor cu adrese impare.

Blocul par este conectat la liniile D0 - D7 ale magistralei de date (octet inferior) iar cel impar la liniile D8 - D15 ale magistralei (octet superior).

Liniile de adresă A1 - A19 se aplică ambelor blocuri de memorie şi ca urmare la încărcarea unei adrese se selectează simultan câte o locație din fiecare bloc.

Transferul de date între locațiile selectate şi magistrala de date se face însă cu încă două linii, A0 şi BHE (fig.9).

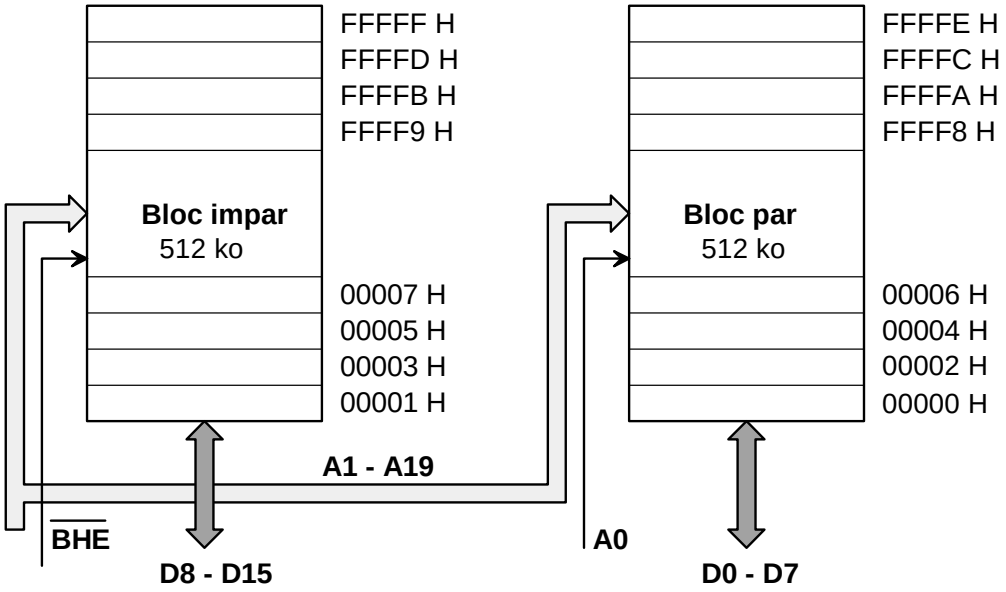

Fig.9 Conectarea blocurilor de memorie la magistrale

BHE	A0	Se transferă
0	0	Ambii octeți
0	1	Octet superior (cu adresă impară)
1	0	Octet inferior (cu adresă pară)
1	1	Fără transfer

Pentru accesul la un operand cu adresă pară, $\mathbf{A_0} = 0$ iar $\overline{\mathbf{BHE}} = 1$ (inactiv) va invalida blocul impar. Ca urmare, are loc un transfer între locația adresată şi octetul inferior al magistralei de date (fig.10).

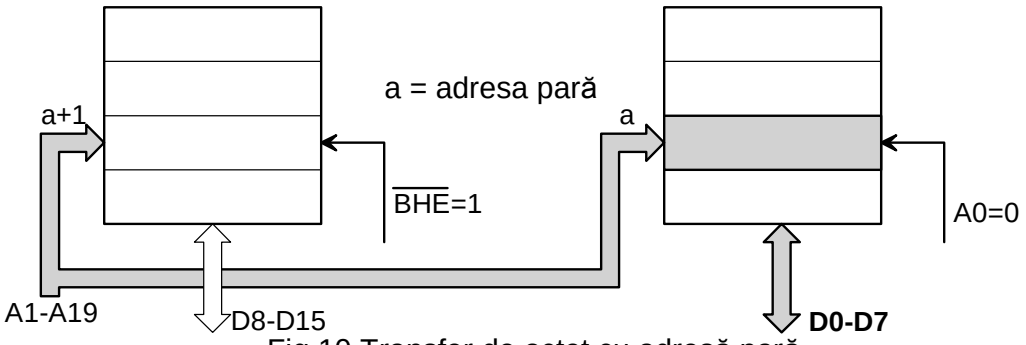

Fig.10 Transfer de octet cu adresă pară

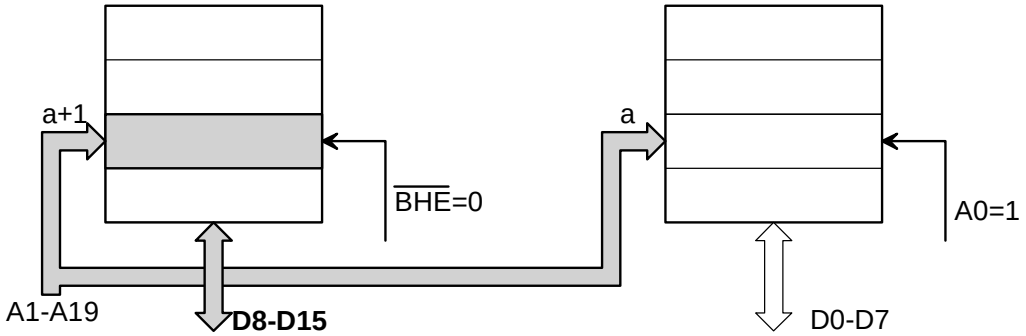

Fig.11 Transfer de octet cu adresă impară

Pentru accesul la un operand cu adresă impară, linia $\mathbf{A_0}$=1va invalida blocul par iar $\overline{\mathbf{BHE}} = 0$ va selecta blocul impar. Ca urmare, are loc un transfer între locația adresată şi octetul superior al magistralei de date (fig.11).

În cazul unui transfer pe 16 biți (cuvânt) începând de la o adresă pară, ambele blocuri vor fi selectate simultan de liniile $\mathbf{A_0}$=0 şi $\overline{\mathbf{BHE}} = 0$. Are loc transferul între cele două locații şi întreaga magistrală de date D0-D15, într-un singur ciclu maşină şi se spune că operandul este *aliniat*, adică începe de la adresă pară.

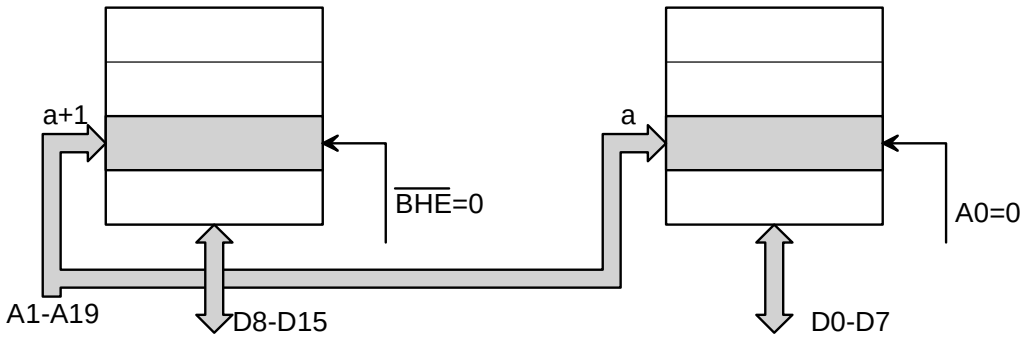

Fig.12 Transfer de cuvânt cu adresă pară

În cazul accesului la un cuvânt (16 biți) care începe la o adresă impară, se spune că operandul este *nealiniat* şi sunt necesare două cicluri maşină pentru transfer. În primul ciclu maşină se transferă octetul de la adresa impară (a+1) pe liniile D8 - D15, fiind considerat octet inferior al operandului de 16 biți. În al doilea ciclu maşină se transferă octetul de la adresa pară (a+2) pe liniile D0 - D7, fiind considerat octet superior al operandului de 16 biți. Toate operațiile, inclusiv direcționarea corectă a octeților, sunt coordonate de microprocesor, procesul de transfer fiind invizibil pentru utilizator. În acest caz, principalul dezavantaj este că procesul de transfer are o durată dublă față de cazul unui operand *aliniat*.

Realocarea dinamică

Structura segmentată a memoriei la 8086, 8088, 80186, 80188 face posibilă scrierea programelor în mod independent de zona de memorie din care vor fi executate. Sistemul de operare poate aloca segmentele de memorie care sunt libere în momentul intrării în execuție - *realocarea dinamică*, ceea ce permite modul de lucru *''multitasking''* (rularea simultană a mai multor programe).

Programele temporar inactive vor fi scrise pe discul magnetic iar spațiul ocupat de ele, alocat altor programe. Dacă programul rezident pe disc este necesar mai târziu, el poate fi reîncărcat în orice zonă disponibilă de memorie. Similiar, dacă este necesar un spațiu mare de memorie pentru blocuri de date, iar memoria disponibilă este formată din fragmente neadiacente, un program segmentat poate fi "compactat" cu toate segmentele dispuse consecutiv, unificând astfel zonele libere pentru blocuri masive de date (fig. 9).

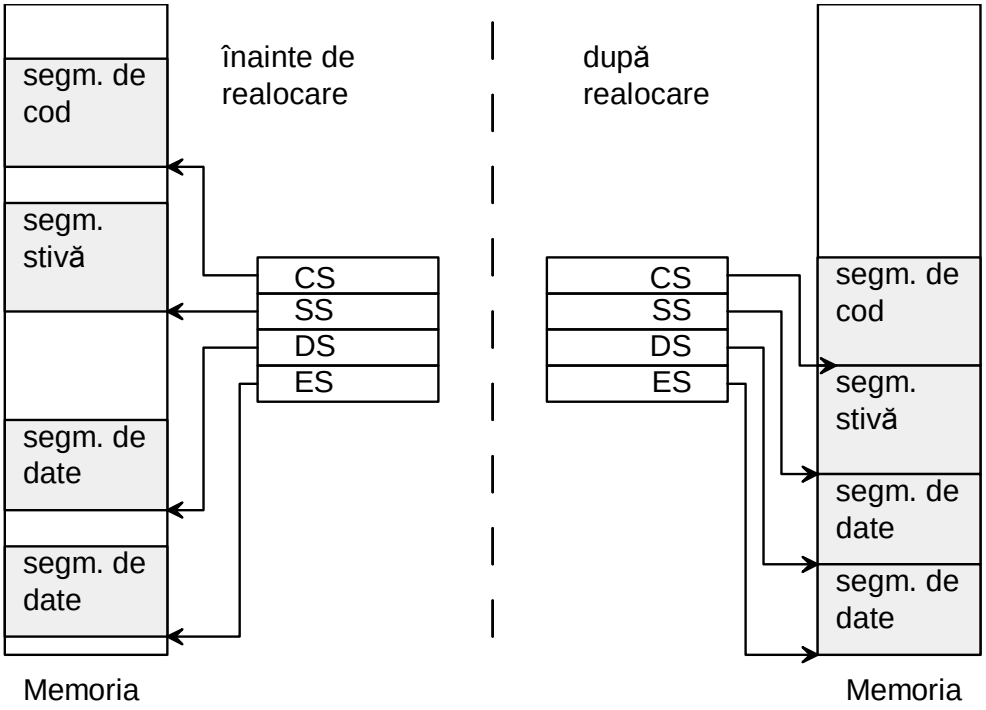

Fig. 9 Realocarea dinamică a memoriei

11.4.1. Locații de memorie dedicate şi locații rezervate

Două zone de memorie (fig. 10) de la adrese joase şi două zone de la adresele înalte sunt dedicate unor funcții specifice ale procesorului sau rezervate de Intel pentru utilizarea produselor sale soft sau hard. Acestea sunt:

- 00000 - 00007F (128 octeți), utilizată pentru memorarea tabelei de vectorilor de întrerupere;
- FFFF0 - FFFFF (16 octeți), zonă adresată la inițializarea sistemului prin activarea semnalului RESET.

Pentru asigurarea compatibilității sistemului cu produsele Intel, aplicațiile nu trebuie să utilizeze zonele de mai sus în alte scopuri.

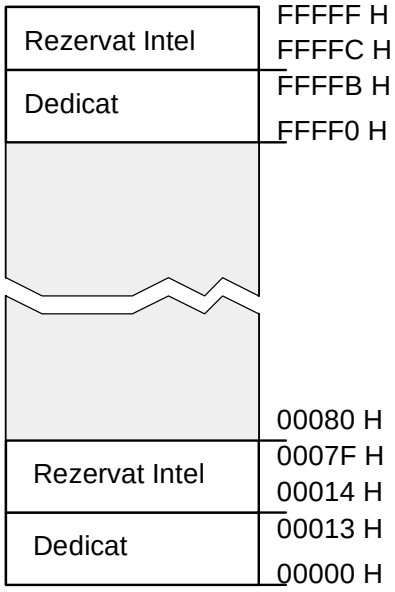

Fig. 10 Organizarea memoriei. Zone dedicate, zone rezervate.

11.5 Organizarea porturilor de intrare / ieşire

Microprocesoarele Intel 8086 / 88 permit utlizarea unui larg spațiu de adrese pentru dispozitive I/O (*Input/Output*), separat față de spațiul de memorie. Pentru transferuri rapide şi complexe se poate folosi accesul direct la memorie sau coprocesorul Intel 8089 specializat în operații I/O.

Porturile I/O pot fi adresate ca locații de memorie, ceea ce permite utilizarea instrucțiunilor de lucru cu memoria, eficiente în privința adresării şi vitezei de lucru.

Spațiul I/O este suprapus ca adrese peste primul segment de 64 ko, în zona 00000 H - 0FFFF H şi este accesibil prin instrucțiunile dedicate, IN şi OUT. Toate porturile sunt considerate în acelaşi segment; ele sunt adresate ca şi locațiile de memorie dar fără registru segment.

Un port poate fi de 8 biți (echivalent cu o locație de memorie de 8 biți), caz în care şi adresa lui este de 8 biți, sau poate fi de 16 biți (echivalent cu o locație de memorie de 16 biți), caz în care adresa lui este de 16 biți. Dacă portul este de 8 biți el se poate conecta fie la octetul

inferior al magistralei de date (D0 - D7), fie la octetul superior (D8 - D15). Adresele asociate porturilor trebuie să corespundă conectării la magistrala de date: dacă portul este conectat la D0 - D7, adresa trebuie să fie pară iar dacă este conectat la D8 - D15, adresa trebuie să fie impară.

Pentru un port de 16 biți, conectarea se face la D0 - D15 iar adresa lui trebuie să fie pară, pentru ca transferul datelor să se facă într-un singur ciclu maşină.

Pentru accesul la un port, unitatea BIU plasează adresa portului (0 - FFFF) pe liniile A0 - A15 ale magistralei de adrese. Adresa portului este specificată în instrucțiune (dacă este pe 8 biți) sau în registrul DX (dacă este pe 16 biți). Instrucțiunile IN (*Input* - intrare) şi OUT (*Output* - ieşire) transferă datele între acumulator şi port. Dacă portul este de 8 biți, transferul se face între port şi AL iar dacă portul este de 16 biți transferul se face între port şi registrul AX.

```
IN  AL, port      ; citire port de 8 biți - adresa = port, data = AL
IN  AX, DX        ; citire port de 16 biți - adresa = DX, data = AX
OUT port, AL      ; scriere port de 8 biți - adresa = port, data = AL
OUT DX, AX        ; scriere port de 16 biți - adresa = DX, data = AX
```

În adresarea indirectă, se utilizează registrul DX pentru adresa de 16 biți (registrul DX este încărcat anterior cu adresa). La execuția instrucțiunii, conținutul lui DX este încărcat pe magistrala de adrese (A0 - A15). Se poate adresa orice port în spațiul I/O din domeniul 0000 H - FFFF H (65 536 porturi).

În adresarea directă sau imediată, adresa portului apare în instrucțiune şi este în domeniul 00 H - FF H (256 porturi).

Dacă porturile sunt adresate ca locații de memorie, orice mod de adresare a operanzilor din memorie poate fi utilizat pentru acces la porturi (exemplu: un grup de terminale de date poate fi accesat ca un tablou).

11.6 Întreruperi

O *întrerupere* opreşte temporar execuția unui program şi transferă controlul unei rutine (subprogram) specifice de tratare ce corespunde cauzei ce a generat întreruperea. Mecanismul prin care se face acest transfer este în esență de tip apel de procedură, ceea ce implică revenirea în programul întrerupt după execuția rutinei de tratare.

Întreruperile hardware externe sunt activate de cereri de întrerupere generate de dispozitive periferice inteligente, sub forma unor semnale electrice, aplicate pe intrările INTR şi NMI ale procesorului; cele interne apar ca urmare a unor condiții speciale de funcționare a procesorului (de exemplu, modul de lucru pas cu pas).

Întreruperile mascabile pot fi dezactivate prin comenzi de program şi sunt cele produse de semnale aplicate pe intrarea INTR a procesorului; cele nemascabile nu pot fi dezactivate prin comenzi de program şi sunt cele produse de semnale aplicate pe intrarea NMI.

Într-un sistem cu procesor 8086 pot exista maxim 256 de întreruperi distincte. Fiecare din aceste nivele poate avea asociată o procedură de tip far, numită rutină de tratare. Adresele acestor rutine sunt înregistrate într-o tabelă de întreruperi aflată în zona de memorie 00000 - 003FFH, ocupând deci 1024 octeți. Fiecare nivel ocupă 4 octeți, primii 2 reprezentând offset-ul iar următorii 2 adresa de segment a procedurii.

Practic, un vector de întrerupere conține adresa completă a subrutinei de tratare întrerupere corespunzătoare.

Tabela vectorilor de întrerupere conține trei zone:

- o zonă dedicată, care este utilizată în mod automat de microprocesor pentru cererile de întrerupere predefinite (00000H-00013 H);
- o zonă rezervată, pentru păstrarea compatibilității cu produsele Intel;
- o zonă disponibilă, la dispoziția utilizatorului (00080H - 003FFH).

Adresa rutinei de tratare (în Hexa)	Tipul întreruperii
003FC	INT 255 - Disponibil
003F8	INT 254 - Disponibil
. .	. .
00080	INT 32 - Disponibil
0007C	INT 31 - Rezervat
. .	. .
00014	INT 5 - Rezervat
00010	INT 4 - Depăşire
0000C	INT 3 - Breakpoint
00008	INT 2 - Nemascabil
00004	INT 1 - Mod pas cu pas
00000	INT 0 - Eroare de divizare

La apariția unei întreruperi au loc următoarele acțiuni:

- se salvează în stivă registrele F, CS, IP;
- se pun în zero indicatorii IF şi TF;
- se furnizează procesorului un octet (0 - 255) numit *vector de întrerupere* care identifică nivelul asociat întreruperii curente;

- prin intermediul tabelei de întreruperi se execută salt intersegment la adresa rutinei de tratare;

Vectorul de întrerupere poate fi furnizat procesorului în unul din următoarele moduri:

- în cazul întreruperilor hard interne nivelul este implicit;
- în cazul întreruperilor hard externe, nivelul este transmis prin magistrala de date în cadrul ciclului maşină de tratare, de către dispozitivul care a generat întreruperea.
- în cazul întreruperilor soft, nivelul este conținut în instrucțiune.

11.6.1. Întreruperile externe

Sunt inițiate la intrările INTR şi NMI ale procesorului. O cerere pe linia NMI generează o întrerupere nemascabilă şi este predefinită de tipul INT 2, adică procesorul execută accesul la vectorul 2 fără alte informații suplimentare.

O cerere pe linia INTR este generată, de regulă, de un circuit extern specializat, care evaluează prioritatea în cazul apariției cererilor simultane din mai multe surse şi permite trecerea cererii cu prioritate maximă la momentul respectiv.

Intrarea INTR este testată de microprocesor la terminarea execuției fiecărei instrucțiuni. Singurele excepții de la regulă apar în cazul instrucțiunilor pentru şiruri de date, la care linia INTR este testată după prelucrarea fiecărui element din şir şi în cazul instrucțiunii WAIT, la care linia INTR este testată după fiecare verificare a liniei TEST.

Există câteva situații în care cererea de întrerupere este servită numai după execuția instrucțiunii următoare. Este cazul instrucțiunilor precedate de prefixe; nu este luată în considerație cererea de întrerupere între execuția prefixului şi a instrucțiunii. De asemenea este cazul instrucțiunilor MOV şi POP care încarcă registrele de segment; nu se recunoaşte nici o cerere de întrerupere decât după execuția instrucțiunii următoare din motive de protecție. Pentru schimbarea segmentului de memorie este necesară modificarea a două registre. Dacă cererea de întrerupere intervine între modificarea primului registru şi cea a celui de-al doilea, există riscul ca ultimul transfer să nu se mai execute corect ceea ce duce la erori grave în continuarea programului.

Pentru servirea cererilor INTR (mascabile) este necesar ca sistemul de întreruperi să fie activat prin setarea indicatorului IF (IF = 1); setarea se face cu instrucțiunea STI (*Set Interrupt*) iar resetarea (IF = 0) se face cu CLI (*Clear Interrupt*).

Dacă IF = 1, microprocesorul va lua în considerație cererea de întrerupere de pe linia INTR. După citirea codului întreruperii pe D0-D7, registrul F (al indicatorilor de condiții) este depus în stivă; se salvează în stivă, de asemenea, registrele CS şi IP şi se anulează IF şi TP pentru a invalida eventuale cereri INTR şi modul de lucru pas cu pas.

Se execută un ciclu maşină special, de tratare întrerupere, în care CS şi IP sunt încărcați din tabela vectorilor de întrerupere, în funcție de codul pe 8 biți al întreruperii.

Subrutina de tratare a întreruperii trebuie să se încheie cu instrucțiunea IRET care determină revenirea în programul principal şi refacerea din stivă a registrelor CS, IP şi F cu vechile valori.

Cererea de întrerupere de tip NMI are codul 2 şi este luată în considerație dacă are o durată de cel puțin două stări. Operațiile efectuate de microprocesor sunt aceleaşi cu cele de la cererea de tip INTR.

11.6.2. Întreruperile interne

Sursa de întrerupere nu este un eveniment extern. Întreruperea apare ca urmare a execuției unei instrucțiuni *INT nn* sau ca urmare a unui eveniment intern (întreruperi predefinite).

Instrucțiunea *INT nn* este pe doi octeți şi realizează legătura cu vectorul *nn* din tabela de întreruperi. Are acelaşi efect cu o întrerupere externă cu codul *nn*, dar nu este mascabilă şi nu se execută o secvență de acceptare, deci nu se vor genera semnale INTA.

Întreruperile predefinite sunt generate automat de procesor la detectarea unor evenimente interne; acestea sunt:

- împărțire la zero - INT 0;
- funcționare pas cu pas - INT 1;
- breakpoint (punct de oprire) - INT 3;
- depăşire - INT 4;

Întreruperea INT 0 este generată ca urmare a execuției unei împărțiri cu câtul mai mare decât valoarea maximă admisă. În cazul instrucțiunii DIV valoarea maximă a câtului este FF sau FFFF, după cum împărțitorul este octet, respectiv cuvânt. În cazul instrucțiunii IDIV, cele două valori sunt 7F sau 7FFF.

Întreruperea INT 1 este luată în considerație dacă indicatorul TF=1. Efectul ei este modul de lucru pas cu pas, în care după fiecare instrucțiune se poate afişa conținutul registrelor, al locațiilor de memorie, informații necesare verificării şi depanării programelor. Procesorul nu dispune de instrucțiuni pentru modificarea directă a indicatorului TF. Operația se poate realiza indirect, cu ajutorul stivei. Instrucțiunea PUSHF depune

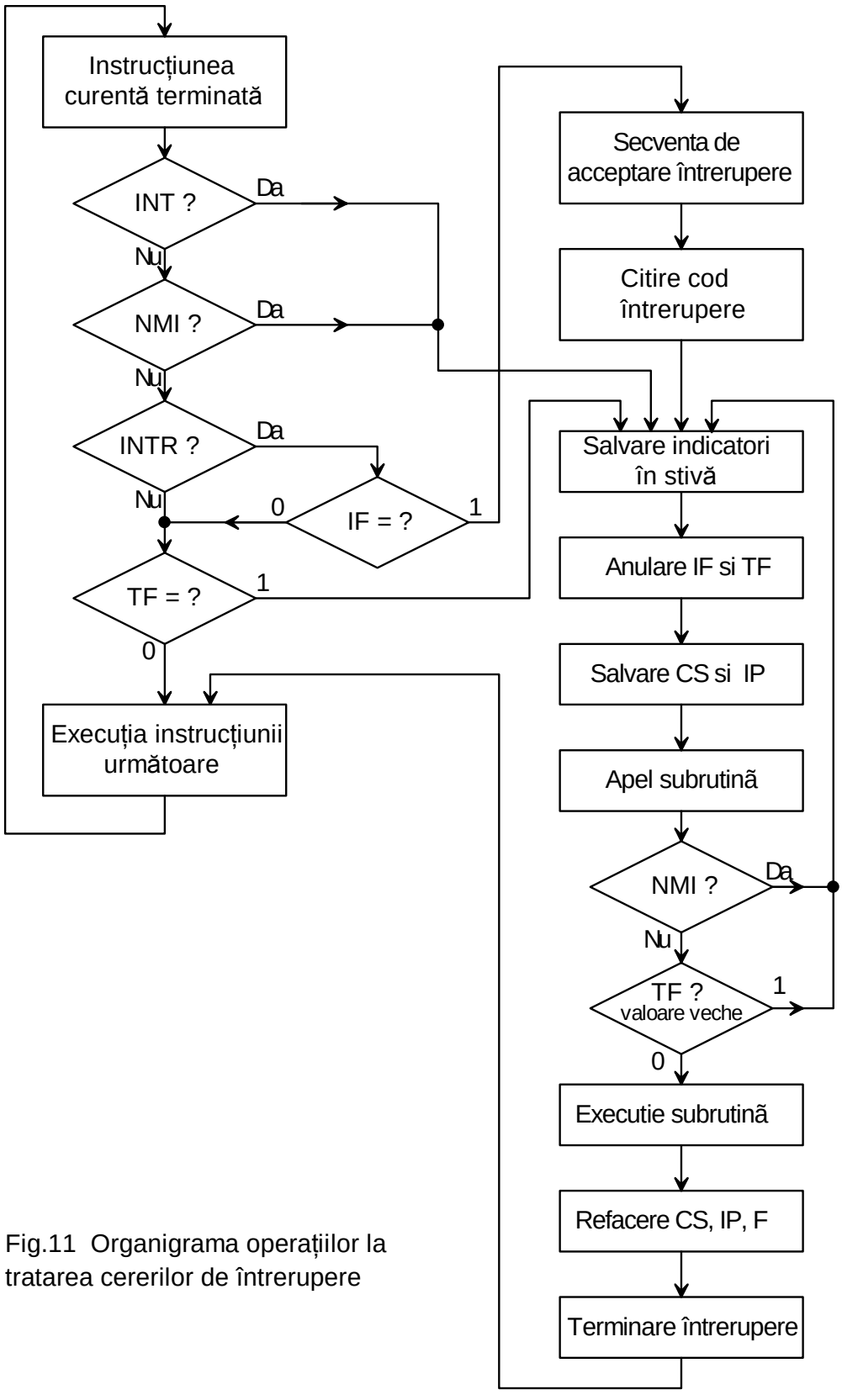

Fig.11 Organigrama operațiilor la tratarea cererilor de întrerupere

registrul F în stivă. Indicatorii pot fi modificați cu instrucțiunile logice şi

reîncărcați cu instrucțiunea POPF.

Întreruperea INT 3, de tip *breakpoint* - punct de oprire, are codul pe un singur octet şi are ca efect oprirea execuției în punctul din program în care apare INT 3. Serveşte testării şi depanării programelor prin examinarea stării procesorului (registre, indicatori etc.) în punctul de oprire.

Întreruperile INT 4 este generată la apariția depăşirii capacității registrelor la execuția unei operații aritmetice, depăşire ce determină setarea indicatorului OF (*Overflow Flag = 1*); întreruperea nu se generează automat ci numai dacă microprocesorul execută instrucțiunea INTO, care, în general, trebuie să urmeze după instrucțiunile aritmetice cu operanzi cu semn. Subrutina de tratare conține, de regulă, un mesaj către utilizator, care îl informează cu privire la apariția depăşirii.

În concluzie, instrucțiunile care generează întreruperi interne sunt:

- INT *nn* - determină generarea unei intreruperi interne de tip *nn*; microprocesorul extrage vectorul de întrerupere *nn* din tabela de întreruperi şi execută subrutina de tratare fără generarea unui ciclu extern de tip INTA;
- INTO - determină generarea unei întreruperi predefinite de tip 4, când indicatorul de depăşire OF=1, ca urmare a execuției unei instrucțiuni aritmetice cu operanzi cu semn;
- IRET - (*Interrupt Return*) determină revenirea în programul principal (din subrutină) prin restaurarea din stivă a registrelor IP, CS şi a indicatorilor de condiții F.

Timpul de procesare definit din momentul recunoaşterii unei cereri de întrerupere şi până când microprocesorul realizează accesul la începutul subrutinei de tratare, depinde de tipul întreruperii şi este dat în tabelul de mai jos.

Tip de întrerupere	Timp (în perioade de tact)
Întrerupere externă INTR	61
Întrerupere externă nemascabilă NMI	50
INT nn	51
INT 3	52
INTO	53
INT 1	50

În tabelul următor sunt prezentate nivelurile de prioritate ale cererilor de întrerupere interne şi externe, care determină ordinea de tratare când cererile apar simultan.

Tip de întrerupere	Prioritatea
Tip 0, INT nn	0 - maximă
INTO	1
NMI	2
INTR	3
Tip 1 (pas cu pas)	4 - minimă

În fig.11 sunt prezentate operațiile executate de microprocesor la acceptarea şi achitarea unei cereri de întrerupere. O cerere de întrerupere este luată în considerare numai la terminarea execuției instrucțiunii curente. Dacă indicatorul IF=1, procesorul execută secvența de acceptare cerere de întrerupere externă, preia codul întreruperii de pe magistrala de date şi execută operațiile comune tuturor tipurilor de întrerupere.

Se salvează în stivă IP, CS şi registrul indicatorilor de condiții, F, după care indicatorii IF şi TF sunt anulați. Dacă în timpul procesării unei cereri de întrerupere, apare o cerere pe linia NMI înaite de realizarea accesului la subrutina de tratare a primei întreruperi, cererea de pe linia NMI va fi servită cu prioritate. După execuția subrutinei de tratare întrerupere se refac din stivă registrele CS, IP, F şi se revine în programul principal. La încheierea instrucțiunii următoare se testează existența unei cereri de întrerupere şi dacă da, procesul se reia.

11.7 Unitate centrală cu microprocesorul 8086

După apariția lui Intel 8086, a fost lansată familia de circuite Intel necesară realizării unităților centrale cu 8086. Ea cuprinde:

- coprocesoarele 8087, 8089;
- 8284 - generator de tact;
- 8228 - controler de magistrală;
- 8282, 8283 - registre tampon de opt biți;
- 8286, 8287 - amplificatoare bidirecționale (8 biți) de magistrală.

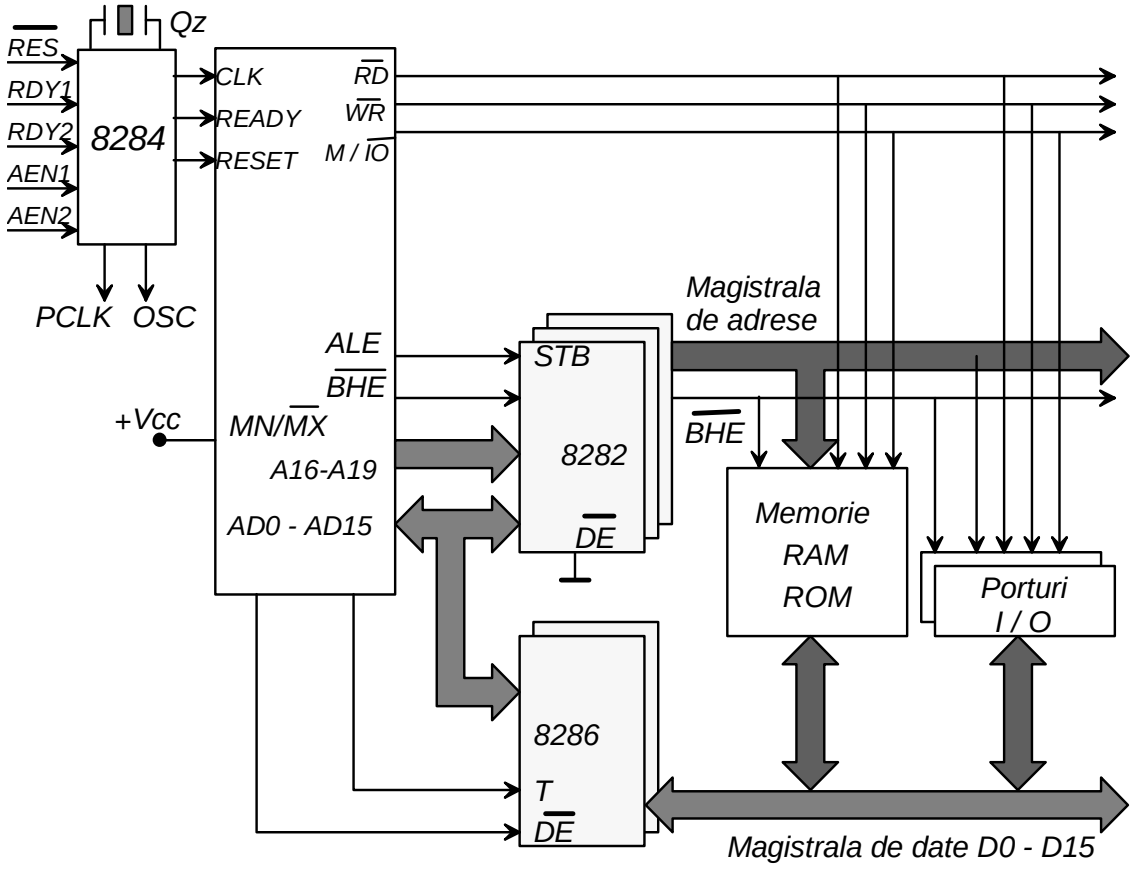

Fig. 12 Unitate centrală cu 8086 în modul minim

Generatorul de tact 8284 generează semnalul CLK pentru microprocesor şi PCLK pentru circuitele specializate cu diferite funcții în sistem. De asemenea generează semnalele RESET şi READY către microprocesor sincronizate cu semnalul de tact.

Pentru generarea semnalului CLK, se divizează semnalul generat de un oscilator cu cuarț sau un semnal exterior; semnalul PCLK are o frecvență de două ori mai mică şi un factor de umplere 1/2.

Registrul 8282 este utilizat ca tampon pentru demultiplexarea magistralei comune de date şi adrese. Are şi rol de amplificator de magistrală, asigurând un *fan - out* de 20 intrări TTL. Informația de la

intrările DI apare al ieşirile DO pe nivelul 1 logic al semnalului STB şi este memorată în cele 8 circuite basculante bistabile de tip D. Pentru ca informația să fie disponibilă la ieşiri, este necesar ca semnalul de validare $\overline{DE} = 0$.

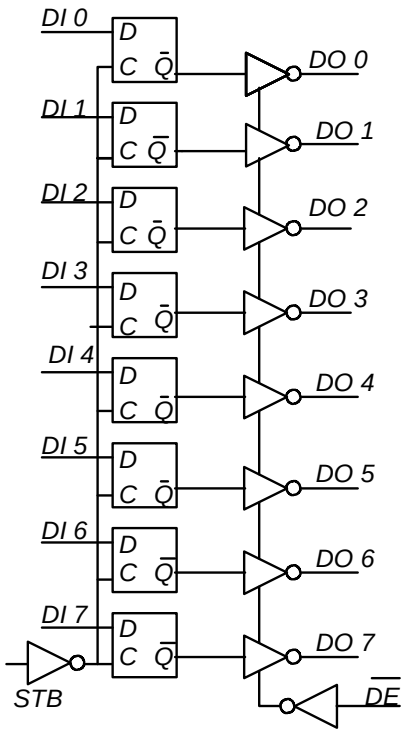

Fig. 13. Schema circuitului 8282

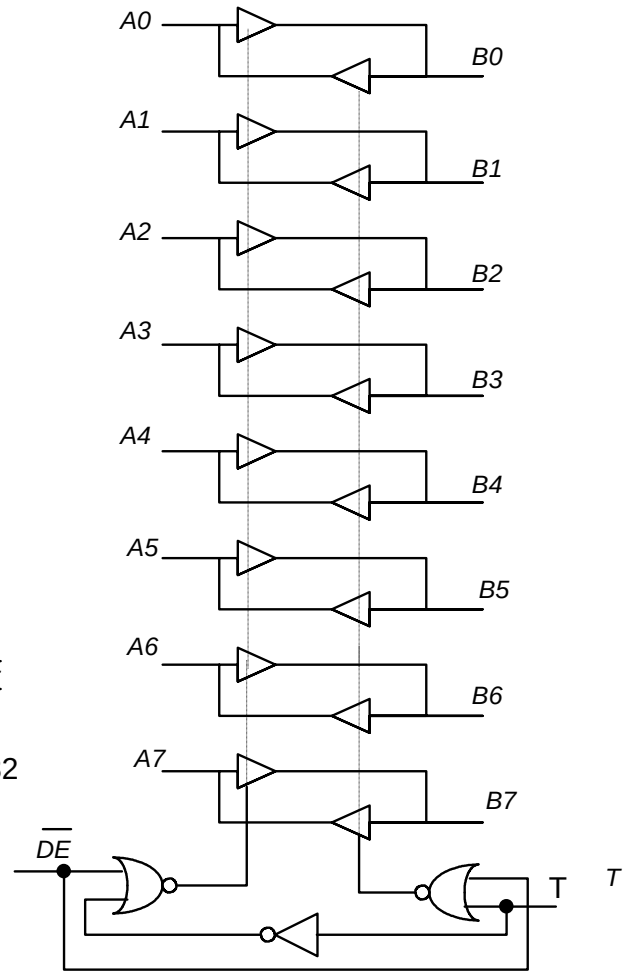

Fig. 14. Schema circuitului 8286

Amplificatorul de magistrală 8286 este bidirecțional, pe 8 biți, cu ieşiri cu trei stări şi *fan-out* 20 pentru ieşirile B, 6 pentru ieşirile A.

Circuitul realizează funcțiile de amplificare şi transfer atâta timp cât $\overline{DE} = 0$. Dacă $\overline{DE} = 1$, ieşirile trec în starea de înaltă impedanță. Sensul transferului este stabilit de intrarea T: dacă *T=1*, transferul se face de la A la B iar dacă *T=0*, de la B la A.

În modul maxim, 8086 poate fi utilizat în sisteme de complexitate mare, de tip multiprocesor. Generarea semnalelor de comandă către memorie şi porturi se face prin intermediul unui circuit specializat din familie, 8288.

Controlerul de magistrale, 8288, amplifică ieşirile de comandă şi date, realizând un *fan-out* 20 pentru ieşirile de comandă şi 10 pentru cele de date.

În funcție de combinația binară dată de liniile de stare, circuitul generează semnalele de comandă pentru citire/scriere memorie şi porturi precum şi pentru acceptarea unei cereri de întrerupere.

Circuitul generează câteva semnale de control: DEN (validarea transferului pe magistrala de date, DT/R (indică sensul transferului), ALE (Adress Latch Enable - pentru demultiplexarea magistralei de adrese şi date)

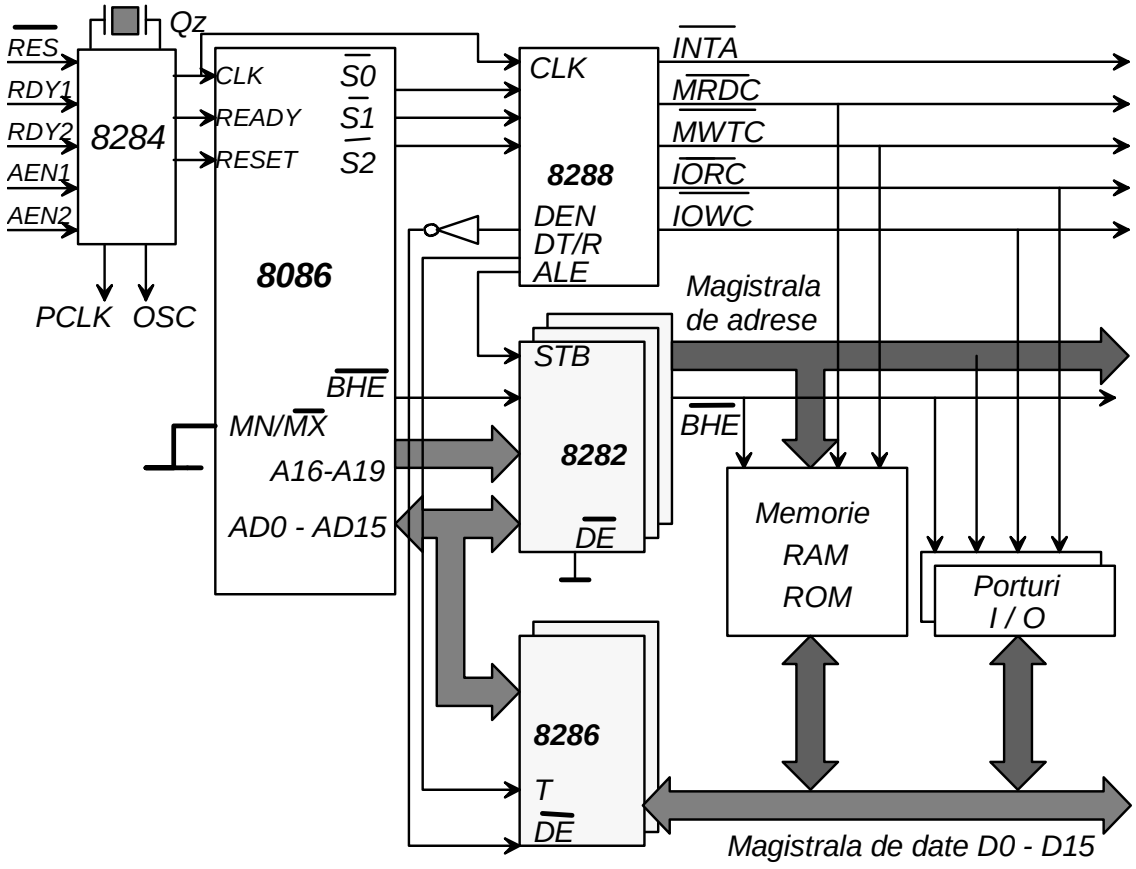

Fig. 15 Unitate centrală cu 8086 în modul maxim