Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Programare în limbaj de asamblare

6. Arhitectura de bază, resursele procesoarelor, moduri de lucru (real, protejat).

Arhitectura de bază. Setul de registre

Familia de procesoare 8086, 88, 186, 286 conține același set de registre de bază, instrucțiuni şi moduri de adresare. Pentru procesoarele 386, 486 şi respectiv Pentium se vor face specificările necesare.

Setul de registre

Procesorul 286 are 15 registre, grupate în patru categorii, conform figurii de mai jos.

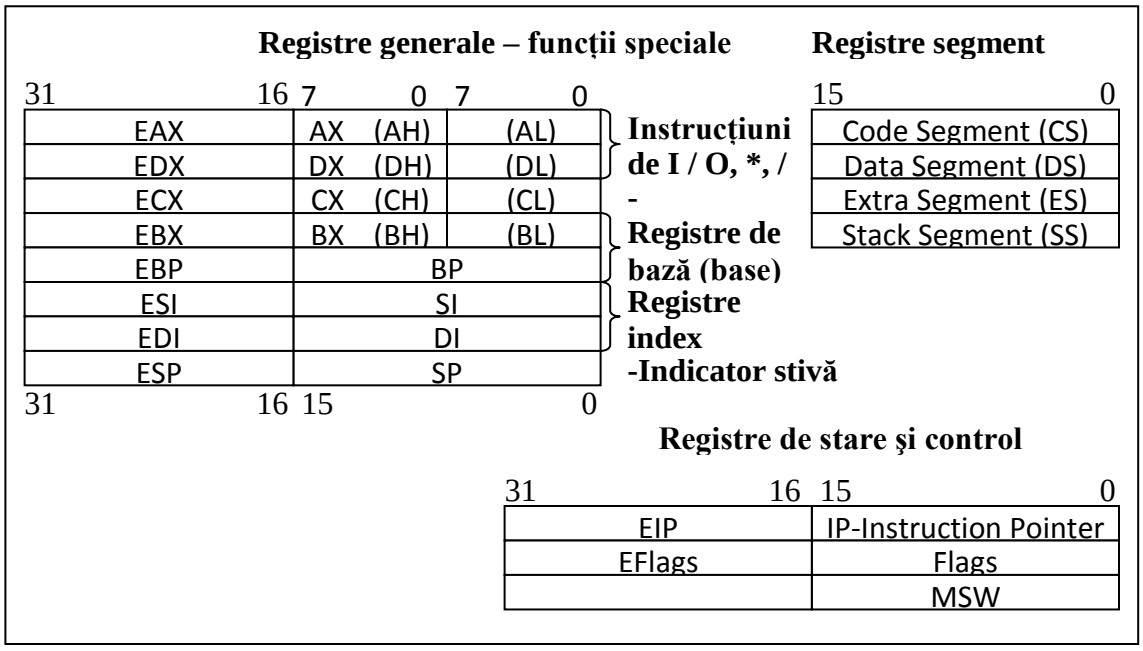
Cele patru categorii de registre sunt: generale, de segment, index şi de bază, de stare şi control.

Registrele generale

Sunt opt registre cu scop general: AX, BX, CX, DX, BP, SP, SI şi DI, utilizate pentru a stoca operanzi logici şi aritmetici. Patru dintre acestea pot fi utilizate fie ca registre de 16 biți, sub numele de AX, BX, CX, DX, fie pot fi împărțite în perechi de registre separate de câte opt biți, denumite astfel: AH, AL, BH, BL, CH, CL, DH, DL (H=high, L=low, respectiv primii 8 biți şi ultimii 8 biți din registrele respective).

Majoritatea instrucțiunilor utilizează în același mod toate registrele; există însă instrucțiuni pentru care anumite registre generale au o anumită semnificație:

- AX (denumit şi acumulator) şi DX sunt utilizate în instrucțiunile de înmulțire, împărțire şi cele de I/O cu o semnificație prestabilită; de asemenea, registrul AL se foloseşte pentru operații de translație de cod sau în operațiile cu numere reprezentate în BCD (binary coded decimal);
- CX este utilizat drept numărător pentru operații de deplasare, rotire, bucle soft, repetări hard a unor instrucțiuni; la 386/486, aceste registre sunt extinse la 32 biți: EAX, EBX, ECX, EDX, ESP, EBP, ESI şi EDI.



Setul de registre al procesoarelor Intel 286 (386/ 486/ Pentium)

Registrele segment

Sunt patru registre speciale, ce permit selectarea în orice moment a segmentelor de memorie imediat adresabile pentru cod (instrucțiuni) – CS, stivă – SS şi date (DS, ES).

Pe lângă registrele de segment, vizibile programatorului pentru fiecare registru segment mai există şi registre cache de descriptori de segment, care nu sunt vizibile programatorului, dar este util să cunoaştem conținutul lor. Aceste registre, asociate cu fiecare registru segment, conțin: adresa de bază a segmentului (24 biți), limita segmentului (16 biți) şi alte atribute de segment (8 biți), deci în total 48 biți. Când valoarea unui selector este încărcată într-un registru segment, registrul cache de descriptor asociat este în mod automat actualizat.

În modul real de adresare se actualizează direct numai adresa de bază (prin deplasarea valorii selectorului cu patru biți la stânga), deoarece limita maximă a segmentului şi atributele sunt fixe în acest mod.

În modul protejat, toate componentele (bază, limită şi atribute) sunt actualizate de conținutul descriptorului de segment la care face referire selectorul. Structura unui selector va fi descrisă ulterior în acest capitol.

Orice referire la memorie va implica în mod automat registrul cache de descriptor de segment, asociat cu registrul segment utilizat pentru referire. Adresa de bază a segmentului devine o componentă în calculul adresei fizice (sau a adresei liniare, la 486), limita este utilizată pentru operații de verificare a limitei segmentului, iar atributele sunt folosite pentru verificarea tipului (drepturilor) de referire la memorie.

La 386/486 mai există încă două registre segment pentru date: FS şi GS. Bineînțeles că şi pentru acestea există registre cache de descriptori de segment asociate. Față de 286 însă, dimensiunea lor este mai mare, deoarece adresa de bază a segmentului este extinsă la 32 de biți, limita segmentului la 20 de biți, iar atributele de segment ocupă 12 biți (deci în total 64 biți).

Registre index şi de bază

Patru dintre registrele generale (BX, BP – registre de bază, SI, DI – registre index) pot fi utilizate, de asemenea, pentru a determina adresele de offset (deplasament) ale operanzilor din memorie. Aceste registre pot conține adresele de bază sau indexul pentru anumite locații de memorie dintr-un segment. Modul de adresare determină registrul specific utilizat pentru calcularea adresei efective a operandului. Registrele index sunt utilizate în mod implicit pentru a face referire la şirurile sursă, respectiv destinație pentru operațiile pe şiruri.

Registrul BP se referă implicit (dacă instrucțiunea nu este precedată de un prefix segment) la segmentul de stivă curent (SS). Tot la segmentul de stivă face referire, implicit pentru operațiile cu stiva, şi registrul SP (Stack Pointer), dar operațiile cu stiva (PUSH, POP, CALL, RET, INT, IRET) actualizează în mod automat registrul SP, afectând în mod corespunzător stiva (extragerea sau introducerea informației în stivă). În schimb, registrul BP poate fi folosit pentru acces la date din stivă (parametrii sau variabilele locale procedurii), fără însă a extrage datele din stivă (adică fără a modifica stiva).

Registre de stare şi control

Sunt trei registre cu scop special, care memorează şi controlează anumite stări ale procesorului: IP, MSW şi F.

Registrul IP (Instruction Pointer) conține adresa de offset pentru următoarea instrucțiune secvențială ce va fi executată. La 386/486, registrele au 32 biți: EIP, EFLAGS.

Registrul MSW (Machine Status Word) memorează starea procesorului, adică dacă a avut loc o comutare de task, şi controlează modul de operare a procesorului. Dintre cei 16 biți sunt folosiți doar ultimii patru – unul trece procesorul în mod protejat, iar ceilalți trei controlează interfața cu extensia procesor – şi care au următoarea semnificație:

- PE (Protected mode Enable) este poziționat pe 1 când se trece în modul protejat de lucru şi nu poate fi pus pe 0 decât cu semnalul RESET;
- MP (Monitor Processor extension / Math Present) specifică dacă este (1) sau nu (0) prezentă în sistem extensia procesor; pe durata instrucțiunii WAIT, dacă MP=1, se testează TS şi se generează excepția 7 ;
- EM (EMulate processor extension) specifică dacă funcțiile extensiei procesor sunt emulate prin soft (1) sau nu (0); permite emularea (dacă este 1), la execuția instrucțiunii ESC, prin generarea unei excepții de extensie procesor inexistentă (7);
- TS (Task Switched) memorează dacă a avut loc o comutare de task, pentru a permite unei excepții de extensie procesor (7) să testeze în ce măsură contextul curent al extensiei procesor aparține taskului curent.

Există instrucțiuni pentru încărcarea/memorarea MSW, în modul real de adresare: LMSW, respectiv SMSW.

La 386, în plus, mai sunt utilizați încă doi biți, cu următoarea semnificație:

- ET (Extension Type) specifică tipul coprocesorului (extensiei procesor) 287 sau 387;
- PG (Paging) specifică dacă procesorul utilizează tabele de pagină pentru a transfera adresa liniară în adresă fizică;

Registrul indicatori (F – Flags register) conține trei categorii de indicatori: de stare, de control şi speciali. Structura acestui registru este prezentată în figura următoare.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	NT	IO	PL	OF	DF	IF	TF	SF	ZF	x	AF	x	PF	x	CF

Structura registrului de indicatori

Indicatorii de stare

- OF (Overflow Flag) – este poziționat pe 1 dacă rezultatul unei operații aritmetice depăşeşte domeniul de reprezentare (limita superioară/inferioară a acestuia) pentru numerele utilizate, adică rezultatul nu poate fi memorat în destinația stabilită de instrucțiune (de exemplu, împărțirea prin zero sau adunarea/înmulțirea a două numere mari – la limita superioară a domeniului).
- SF (Sign Flag) – este indicatorul de semn al rezultatului unei operații şi, de fapt, coincide cu primul bit al rezultatului, fiind 1 dacă numărul este negativ, respectiv 0 pentru pozitiv.
- ZF (Zero Flag) – este poziționat pe 1 dacă rezultatul unei operații aritmetice sau logice este zero, altfel bitul este poziționat pe 0.
- AF (Auxiliary carry Flag) – este pus pe 1 dacă în urma execuției unei instrucțiuni a apărut un transport din rangul 3 în rangul 4, sau un împrumut dinspre rangul 4 spre rangul 3. Acest indicator este utilizat la implementarea aritmeticii pentru numere zecimale codificate binar (BCD – Binary Coded Decimal).
- PF (Parity Flag) – este poziționat pe 1 când numărul de unități din rezultat este par (paritate pară); a nu se confunda cu paritatea unui număr (de exemplu numărul 2, care este un număr par, are

reprezentarea binară 00000010, iar acest octet are paritatea impară – o singură unitate).

- CF (Carry Flag) – se poziționează pe 1 dacă a apărut un transport sau un împrumut în/din rangul cel mai semnificativ al rezultatului, în urma execuției unei instrucțiuni aritmetice.

Indicatorii de control

- DF (Direction Flag) – este utilizat de instrucțiunile pe şiruri şi specifică direcția de parcurgere a acestora:
  - 0 – şirurile se parcurg de la adrese mici spre adrese mari;
  - 1 – şirurile sunt parcurse invers.
- IF (Interrupt Flag) – acest indicator controlează acceptarea semnalelor de întrerupere externă. Dacă IF = 1 este activat sistemul de întreruperi, adică sunt acceptate semnale de întrerupere externă (mascabile, pe linia INTR); altfel, acestea sunt ignorate. Indicatorul nu are influență asupra semnalului de întrerupere nemascabilă – NMI.
- TF (Trace Flag) – este utilizat pentru controlul execuției instrucțiunilor în regim pas cu pas (instrucțiune cu instrucțiune), în scopul depanării programelor. Dacă indicatorul este 1, după execuția fiecărei instrucțiuni se va genera un semnal de întrerupere intern (pe nivelul 1). Evident, execuția secvenței de tratare a acestei întreruperi se va face cu indicatorul TF = 0.

Indicatori speciali

- IOPL (Input/Output Privilege Level) – acest indicator ocupă doi biți şi defineşte dreptul de a utiliza instrucțiuni de intrare/ieşire (I/O). Aceste instrucțiuni, precum şi instrucțiunile ce operează asupra lui IF, sunt denumite instrucțiuni „sensibile“ la IOPL, deoarece ele nu pot fi executate decât dacă CPL <=IOPL – procedura care execută aceste instrucțiuni trebuie să se execute la un privilegiu cel puțin egal cu acela specificat de IOPL.
- NT (Nested Task) – este automat poziționat pe 1 sau 0 de operațiile de comutare de task; execuția unei instrucțiuni IRET, cu NT=1, realizează o comutare de task.

La procesorul 386/486, acest registru (EF) are 32 biți, dintre care ultimii 16 sunt identici cu aceştia, dar în plus mai sunt utilizați încă doi biți (16 şi 17) care au următoarea semnificație:

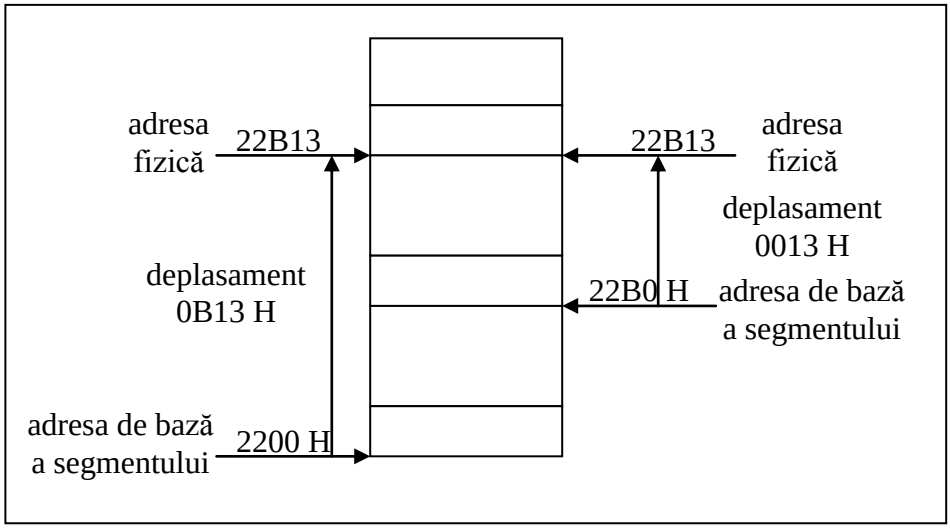
- RF (Resume Flag) – acest indicator dezactivează, temporar, excepțiile de depanare (debug), astfel încât să se poată relua o instrucțiune după o excepție de depanare, fără a se genera imediat o altă excepție de depanare; adică nu se execută o instrucțiune de punct de suspendare dacă este returnat controlul de către excepția de depanare direct la o astfel de instrucțiune.
- VM (Virtual 8086 Mode) – acest bit arată că se execută un program 8086, dacă este poziționat pe 1.

Organizarea memoriei

Memoria este organizată ca seturi de segmente de lungime variabilă. Fiecare segment este o secvență continuă, liniară, de până la 64 Ko (2^{16}). Adresa fizică, de început, a unui segment este multiplu de 16. Fiecare segment este alcătuit din locații succesive de memorie şi reprezintă o unitate independentă şi adresabilă separat. Fiecărui segment i se asociază în spațiul memoriei o adresă de bază, care este adresa de start. Segmentele, în modul real, pot fi adiacente, disjuncte sau suprapuse total sau parțial. O locație de memorie poate fi conținută în mai multe segmente. Unei locații de memorie i se asociază o adresă fizică, de 20 sau 24 de biți, care identifică în mod unic locația, şi o adresă logică- pe care o utilizează efectiv programatorul – care nu este unică. Adresa logică se compune dintr-un selector (adică adresa de bază a segmentului) şi un deplasament față de începutul segmentului; practic, mai multe adrese logice pot face referire la aceeaşi adresă fizică: modificând în

mod corespunzător deplasamentul, pentru o altă adresă de bază a segmentului se poate obține acces la aceeaşi locație de memorie, ca în figura de mai jos.

Din punct de vedere fizic, memoria este organizată sub forma a două blocuri: blocul superior de adrese impare (high) este conectat la magistrala de date pe liniile D_{15}-D_8, iar blocul inferior de adrese pare (low) este conectat la liniile D_7-D_0. Ambele blocuri sunt selectate în paralel, pe liniile de adresă A_{19}(A_{23})-A_1; pentru accesul selectiv la octetul inferior, la cel superior sau la ambii octeți simultan, deci la un cuvânt, se activează în mod corespunzător liniile A_0 şi BHE\. Când se execută un acces la o adresă impară octetul va fi transferat pe liniile high (D_{15}-D_8), când ar trebui să se găsească pe liniile low (D_7-D_0). Din fericire, CPU recunoaşte această situație şi în mod automat transferă data de pe liniile high pe liniile low. Dacă adresa furnizată este pară, se poate citi un cuvânt de acolo, dar dacă dorim să citim un cuvânt de la o adresă impară, atunci se vor efectua două cicluri de citire. Primul, de la adresa pară imediat inferioară celei impare solicitate, care va furniza octetul low (de la adresa impară), iar al doilea cuprinde o citire de la adresa pară imediat superioară, pentru a citi octetul high al cuvântului solicitat. Accesul la operanzi de 32 de biți, pentru un procesor de 16 biți, va necesita cel puțin două cicluri de acces sau cel mult trei, pentru cei plasați la adrese impare.



Adresa logică

Pentru procesoarele 386/486 pot fi considerate încă două blocuri de memorie, conectate la liniile de date D_{31}-D_{24}, respectiv D_{23}-D_{16}, pentru a se putea citi şi cuvinte pe 32 de biți. Adresa plasată pe magistrală este întotdeauna multiplu de 4. Utilizând diverse linii de „*activare octet*“ CPU poate selecta oricare dintre cei patru octeți de la adresa la care programul doreşte să obțină acces. În mod asemănător procesoarelor pe 16 biți, CPU va aranja automat în mod corespunzător octeții transferați. Procesorul poate accesa un dublu cuvânt într-un singur acces la memorie, dacă adresa sa este divizibilă cu 4, altfel vor fi necesare două cicluri de acces la memorie. CPU va face toate aceste operații în mod automat pentru a încărca corect datele solicitate, dar pentru a beneficia de performanța procesorului datele trebuie aliniate: valorile de tip cuvânt trebuie plasate la adrese pare, iar cele de tip dublu cuvânt trebuie aliniate la adrese divizibile cu 4.

Instrucțiunile şi datele, de 8, 16, 32, 64 sau 80 biți, pot fi memorate la orice adresă, indiferent de aliniament. Prin aceasta se permite obținerea unei densități superioare a programului în memorie. Dacă pentru cod aliniamentul nu este important, întrucât instrucțiunile sunt citite la nivel de cuvânt în

momentul când se eliberează doi octeți din coada de instrucțiuni de către unitatea de citire anticipată din BU, nu acelaşi lucru se întâmplă şi pentru date. Pentru citirea/scrierea datelor de 16 sau 32 biți, memorate la adrese ce nu sunt multiplu de 2 sau 4, vor fi necesare două cicluri magistrală, în loc de unul, dacă datele sunt aliniate în memorie.

Pentru cod, singurele momente în care pot să apară citiri de la adrese impare sunt cele în care se execută o instrucțiune de salt la o adresă impară; în acest caz, se va citi în primul ciclu de magistrală un octet de la adresa respectivă, după care se continuă citirea de octeți de instrucțiune de la adrese pare.

Convenția de memorare a datelor multiplu de octeți, 16, 32, 64 sau 80 biți, este: se începe cu octetul mai puțin semnificativ la prima adresă (cea mai mică) şi se depun în continuare octeții următori, în ordinea crescândă a semnificației, terminând cu octetul cel mai semnificativ la adresele următoare.

O clasă specială de date este cea pe dublu cuvânt (32 biți), denumite şi pointeri sau referințe, utilizate pentru a adresa date şi cod. Şi acestea se memorează după aceeaşi regulă: la adresa mai mică se află deplasamentul (offsetul), iar la adresele următoare se află adresa de segment a pointerului.

Memoria este adresată utilizând două componente de adresă:

- un selector de segment (16 biți);
- un deplasament (offset), tot de 16 biți.

Selectorul de segment al pointerului indică segmentul dorit din memorie, iar deplasamentul indică adresa octetului dorit în cadrul segmentului. Există două moduri de adresare: real şi protejat, ultimul de adresare virtuală.

Structura segmentată a memoriei permite să se scrie programe independente de poziția în memorie, adică relocabile. Pentru ca un program să fie relocabil dinamic, el nu trebuie să încarce sau să modifice registre segment sau să transfere controlul direct la o locație în afara segmentului de cod curent.

Relocarea dinamică permite un sistem multiprelucrare (multitasking) pentru a utiliza eficient spațiul de memorie disponibil. Programele inactive se salvează pe disc, iar spațiul eliberat este alocat altor programe; dacă programul este necesar ulterior, el poate fi adus înapoi, la orice locație de memorie disponibilă, şi relansat în execuție.

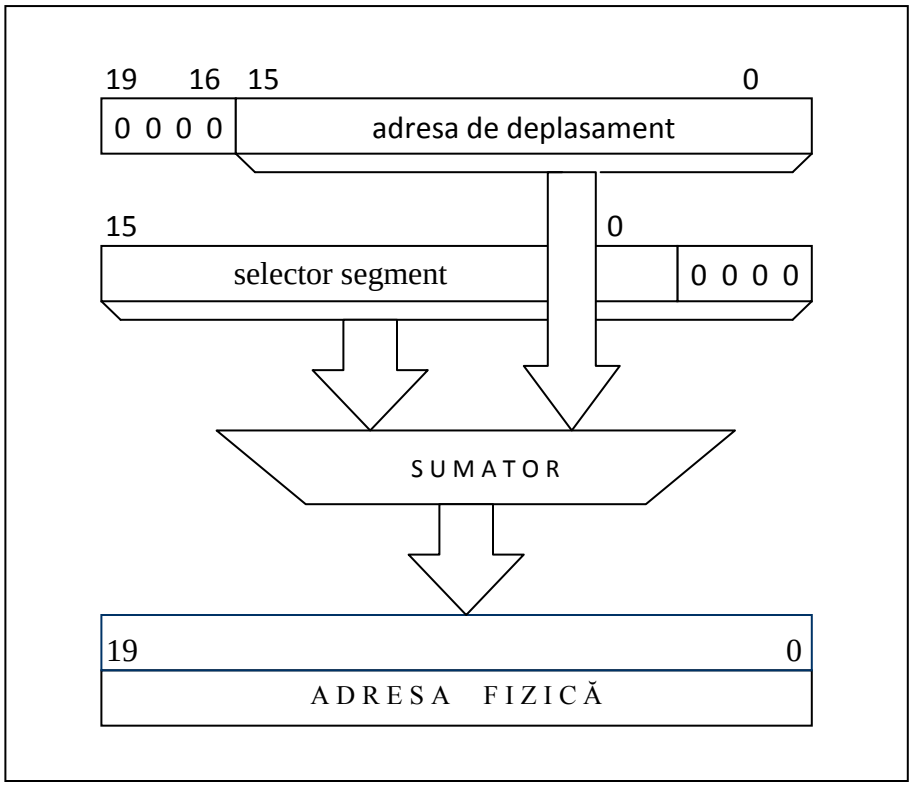
Modul real de adresare 8086

În acest mod, 80286 execută setul de instrucțiuni 8086; în acest mod, codul obiect este compatibil cu 8086. Memoria fizică este o rețea continuă de 1 Mo, adresabilă prin pinii $A_0 \div A_{19}$ şi BHE ($A_{20} \div A_{23}$ pot fi ignorați).

În acest mod, procesorul generează o adresă de 20 biți direct dintr-o adresă de segment de bază de 20 biți şi un offset de 16 biți, ca în figura următoare.

Porțiunea selectorului unui pointer este interpretată ca primii 16 biți dintr-o adresă de segment de 20 biți (ultimii patru sunt întotdeauna zero). Adresele de segment sunt întotdeauna multipli de 16. Toate segmentele, în modul real de adresare, au dimensiunea de 64 Ko. În modul real sunt rezervate două zone de memorie, şi anume:

- zona de inițializare a sistemului (FFFF0÷FFFFFH);
- zona tabelei de întreruperi (00000H÷003FFH).



Generarea adresei fizice în mod real

Modul protejat de adresare virtuală

În acest mod, procesorul execută şi setul de instrucţiuni 8086, dar furnizează mecanisme de protecţie şi administrare a memoriei virtuale şi instrucţiunile asociate.

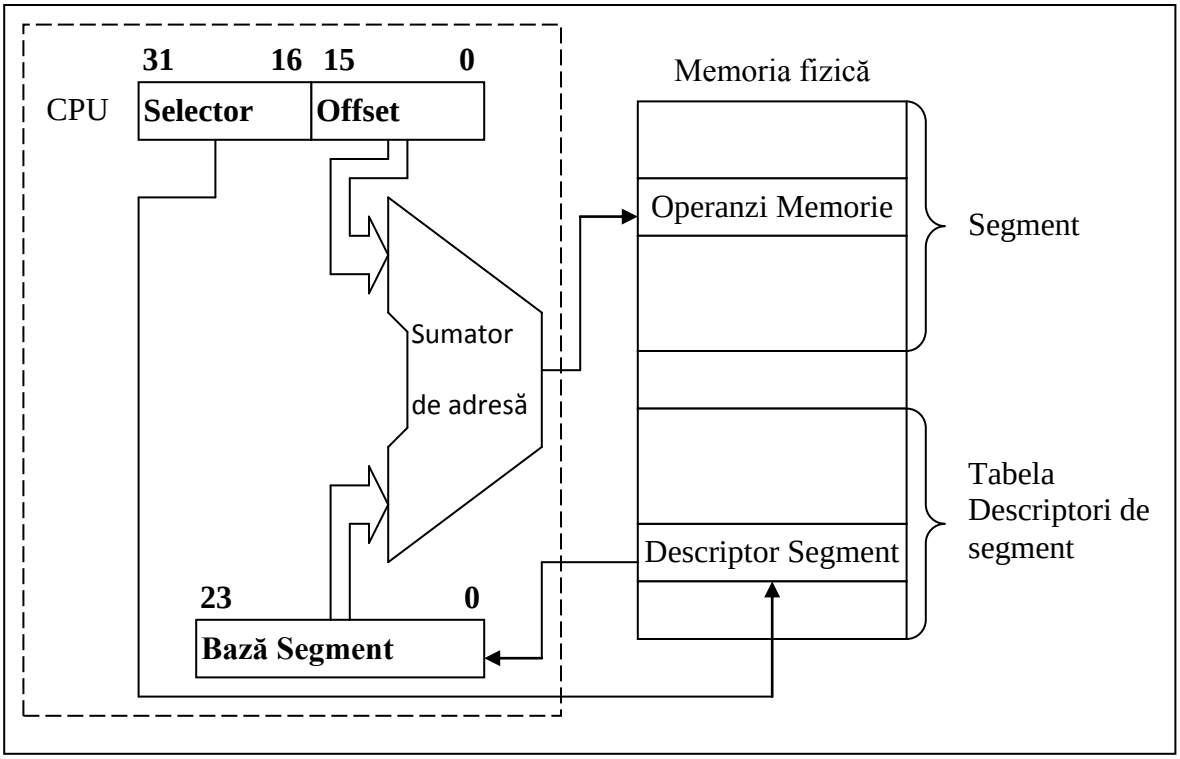
Din punct de vedere al aplicaţiilor în limbaj de asamblare, nu există mari diferenţe între programarea în modul real şi modul protejat. Programatorul nu lucrează direct cu task-urile, având acces la ele doar prin intermediul sistemului. Sistemul de operare este cel care gestionează task-urile. El utilizează instrucţiuni specifice din setul de instrucţiuni al procesorului, care permit iniţializarea modului protejat şi controlul task-urilor simultane. Aceste instrucţiuni nu sunt folosite de programele de aplicaţii.

În continuare vom prezenta câteva caracteristici ale modului protejat, descrierea memoriei virtuale, controlul task-urilor multiple, mecanismul de protecţie şi instrucţiunile specifice.

Se poate intra în acest mod din modul real prin setarea bitului PE din MSW, cu instrucţiunea LMSW (Load Machine Status Word). Acest mod oferă spaţiu de adresă fizică extinsă şi spaţiu de adresă virtual. Spaţiul de adresă virtual este de 1 Go mapat (suprapus) peste spaţiul de adresă fizică de 16 Mo, definit de pinii de adresă $A_{23} \div A_0$ şi BHE\. Spaţiul de adresă virtual poate fi mai mare decât spaţiul fizic, deoarece orice utilizare a unei adrese care nu se suprapune (mapează) cu o locaţie fizică de memorie va determina o excepţie de reiniţializare.

Ca şi în modul real, modul protejat utilizează pointeri de 32 biţi, constând dintr-un selector de 16 biţi şi un offset de 16 biţi. Selectorul specifică un index într-o tabelă rezidentă în memorie, în loc de primii 16 biţi ai adresei reale de memorie.

Adresa de bază de 24 biți a segmentului dorit se obține de la tabelele din memorie. Offsetul se adună la adresa de bază a segmentului pentru a forma adresa fizică, ca în figura următoare.



Generarea adreselor fizice în modul protejat

CPU face în mod automat referire la tabele ori de câte ori un registru segment este încărcat cu un selector. Toate instrucțiunile care încarcă un registru segment vor face referire la tabelele din memorie, fără soft suplimentar. Aceste tabele conțin valori de 8 octeți, denumiți descriptori. Există descriptori de segment pentru segmentele de cod, stivă, date, descriptori de control ai sistemului pentru segmente de date speciale şi operații de transfer al controlului (comutare de task).

Tabela descriptorilor segment face asocierea între un selector şi un segment fizic din memorie, convertind astfel adresa virtuală într-o adresă fizică. Pe baza tabelei descriptorilor TD, procesorul determină poziția în memoria fizică a segmentului corespunzător unui anumit selector. Unele intrări din tabela DT sunt marcate ca *„segment neîncărcat“*. Pentru aceste intrări din DT, procesorul generează o întrerupere. Rutina de tratare a întreruperii încarcă segmentul de pe disc în memorie, permițând apoi accesul la informația din acel segment. Un segment care nu mai este folosit este salvat pe disc de către sistemul de operare. Sistemul realizează operația de interschimb de segmente între memorie şi disc. Pentru a gestiona eficient memoria, sistemul de operare păstrează în memorie segmentele cel mai recent utilizate. Pentru celelalte segmente, marchează intrările în tabela DT ca *„segment neîncărcat“*, utilizând spațiul disponibil pentru alte segmente ce urmează a fi încărcate.