Programare in limbaj de asamblare

Modul de lucru protejat la procesoarele x86

Necesitatea introducerii modului protejat

- Cresterea spatiului de adresare de la 1Mo la 4Go
- Control mai eficient al alocarii memoriei
- Protectie sporita a zonelor de memorie alocate pentru diferite scopuri
- Asigura mai multe nivele de protectie si de acces la resursele calculatorului
- Control mai strict al operatiilor critice, care pot sa afecteze functionarea sitemului:
  - operatii de intrare/iesire (acces la interfete de I/E)
  - intreruperi

Necesitatea introducerii modului protejat

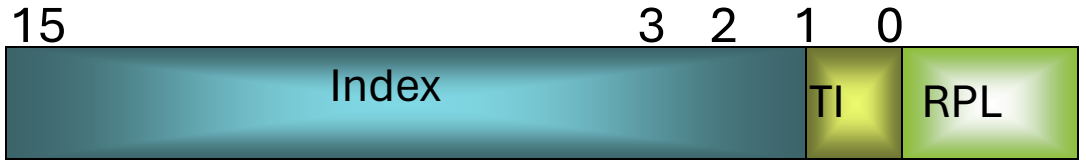
- Ofera suportul necesar pentru implementarea unui sistem de operare multitasking si multiutilizator, prin:
  - managementul memoriei (segmentare si paginare)
  - managementul intreruperilor
  - managementul taskurilor
- Asigura un control mai bun al fiabilitatii, prin mecanisme de detectie a erorilor de acces:
  - tentativa de acces la o zona de memorie nealocata taskului
  - tentativa de scriere intr-o zona nepermise
  - lipsa nivelului de prioritate solicitat

Modurile de lucru ale procesoarelor x86

- Modul real (8086)
  - spatiu de memorie de 1Mo
  - segmente de lungime fixa de 64ko
  - nu exista mecanisme de protectie a memoriei
- Modul protejat (‘286-partial, 386, 486, Pentium-perfectat):
  - spatiu de memorie de 4Go
  - segmente de lungime variabila (1octet-4Go)
  - metode avansate de protectie
- Modul virtual:
  - simularea modului real in regim protejat

Elementele modului protejat

- Selectori de segment
  - se pastreaza in registrele segment (CS, DS, ES, SS, GS, FS)

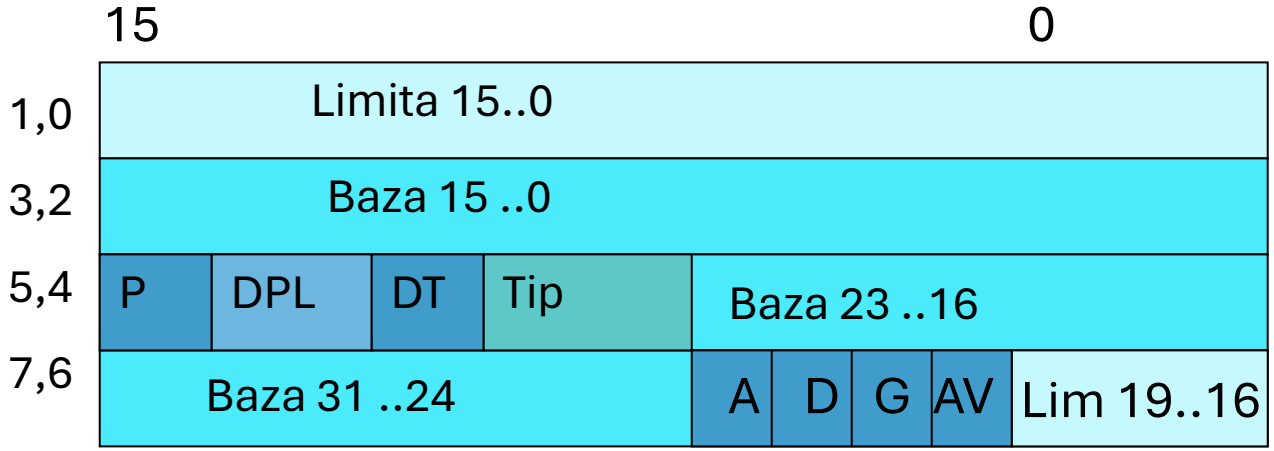


  - continut:
    - Index - arata pozitia Descriprorului de segment in Tabela de descriptori
    - TI - identificatorul tabelei de descriprori
      - 0 - Tabela de descriprori generali
      - 1 - Tabela de descriprori locali
    - RPL - Requested Privilege Level - nivelul de prioritate solicitat

Elementele modului protejat

- Descriptori de segment
  - controleaza accesul la un segment prin:
    - adresa de inceput a segmentului
    - lungimea segmentului
    - drepturi de acces
    - indicatori
  - tipuri de descriptori:
    - descriptori pt. segment de date si de cod
    - descriptori de sistem
    - descriptori de porti

Descriptori pt. segmente de cod si date



Baza - adresa de inceput (32 b)

Limita - dimensiunea segm. (20b)

Tip: - tipul segmentului

DPL - Descriptor Privilege Level

DT - Dtype (1 pt. cod si date, 0 pt. sist. si poarta)

G - granularitate:

0 - octet; 1- 4ko

P - prezenta

1 -prezent in mem.

D - dimensiuna implicita

0- 16b; 1-32b

AV - disponibil pt. progr.

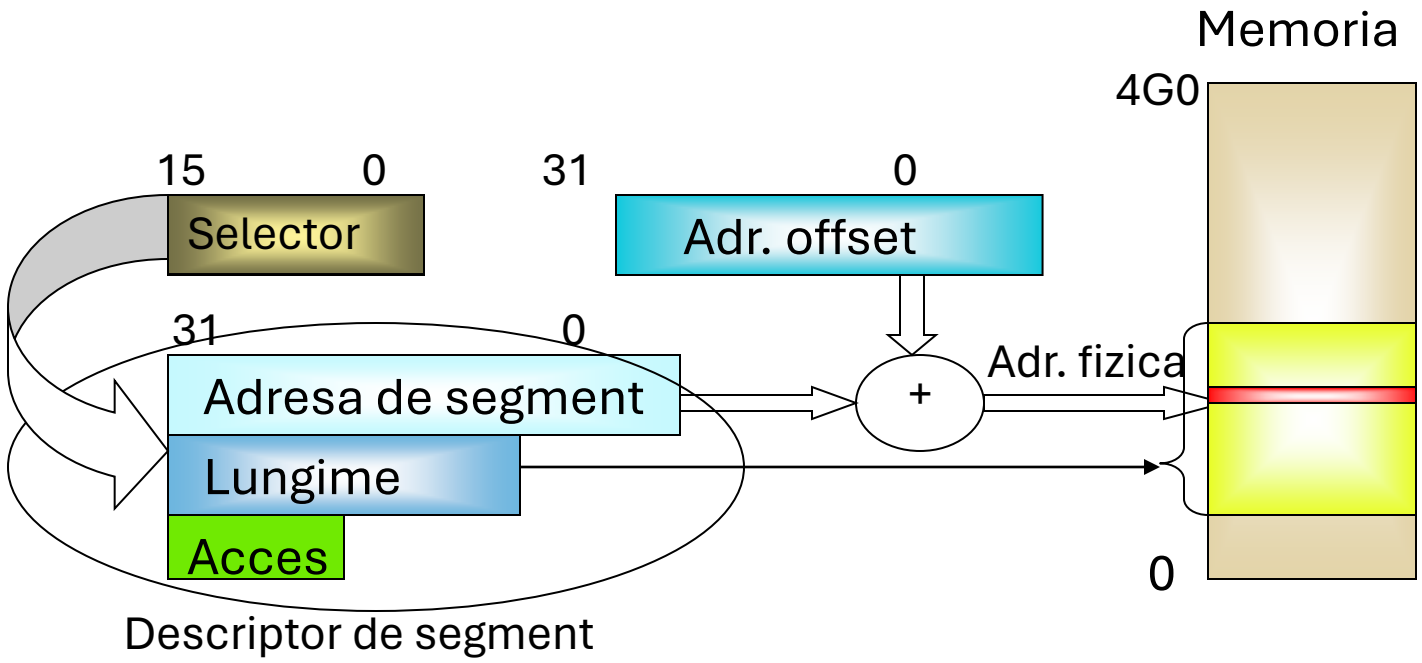
A - accesat

Tipul sectorului

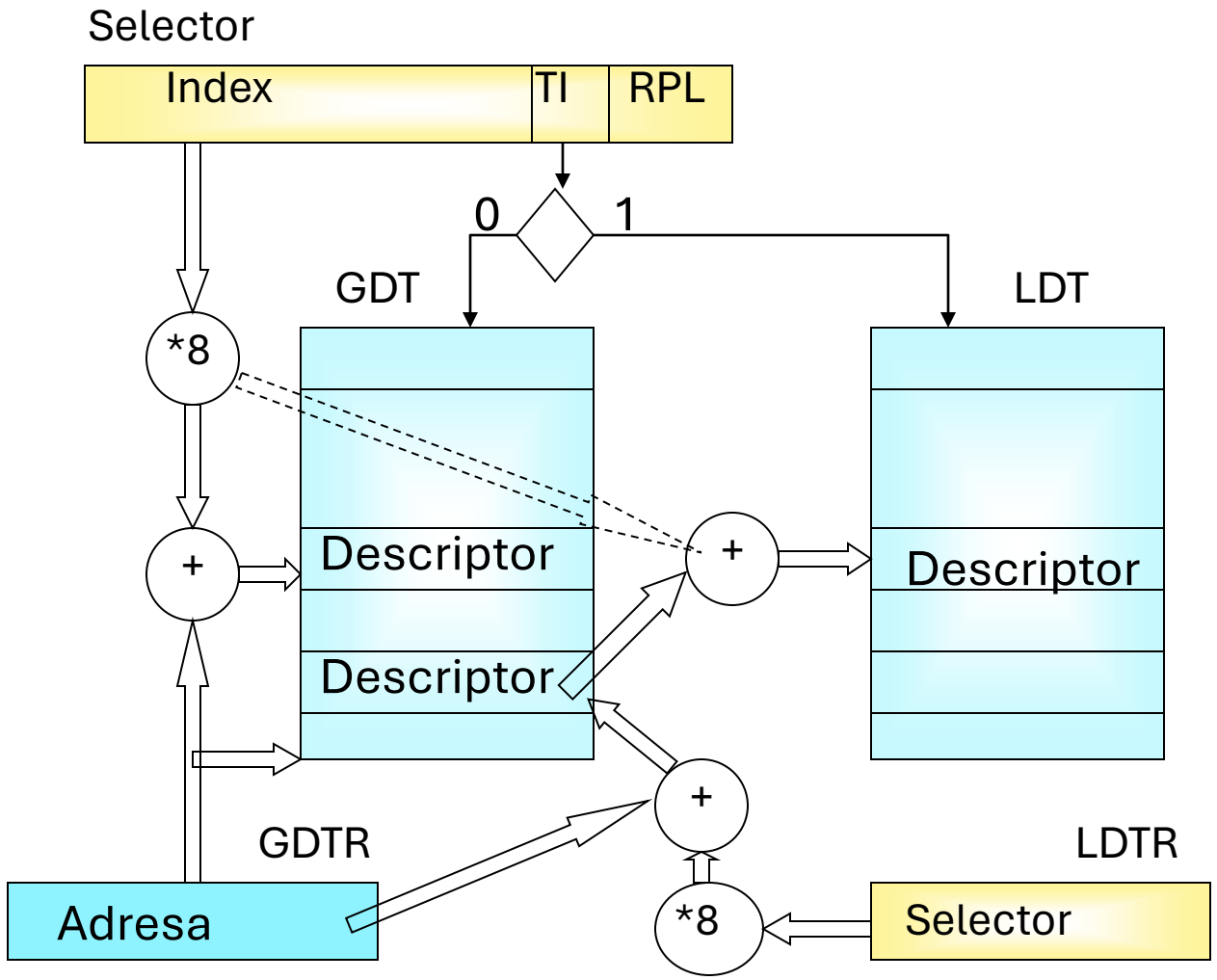
- Pt. segment de cod:
  - Bit 11 - E - executable
    - =1 pt. segment de cod
  - Bit 10 - C - conforming
    - 0- accesibil pt. CPL=DPL
    - 1- accesibil pt. CPL<DPL
  - Bit 9 - R - readable
    - 0- nu se poate citi (numai executa)
    - 1 - se poate citi
  - Bit 8 - A - accessed
    - 0- neaccesat
    - 1- a fost accesat

- Pt. segment de date
  - Bit 11 - E - (executable)
    - =0 pt. segment de date
  - Bit 10 - ED - expansion direction:
    - 0 - extindere spere adrese mari
    - 1 - extindere spre adrese mici
  - Bit 9 - W - writeable
    - 0 - nu se poate scrie
    - 1 - se poate scrie
  - Bit 8 - A - accessed
    - 0- neaccesat
    - 1- a fost accesat

Adresarea memoriei



Adresarea memoriei



Descriptori de sistem

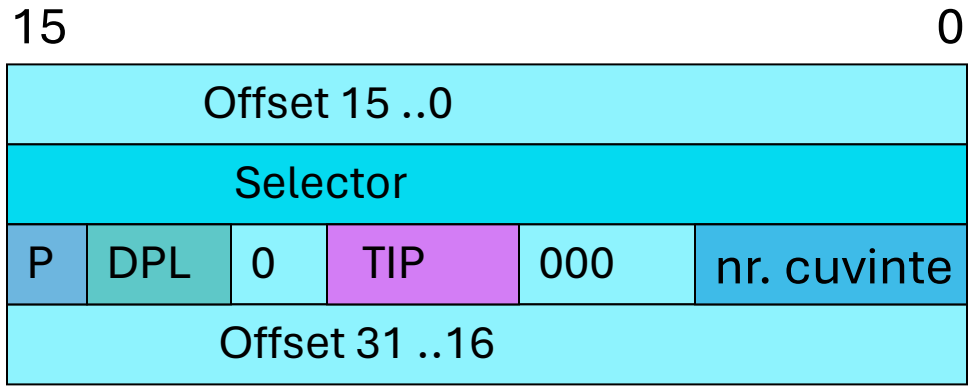
- Descriptor TSS (Task State Segment)
  - determina segmentul care pastreaza starea unui task
  - in TSS se salveaza continutul registrilor procesor la trecerea de la un task la celalalt (comutare de context)
- Descriptor LDT (Local Descriptor Table)
  - descriptorul segmentului care pastreaza tabela descriptorilor locali (LDT)

- similari cu descriprorii de date si cod, difera doar semnificatia campului “TIP”

- Porti (de acces)

Descriptori de porti

- cai de apel ai unor functii sistem, care se afla in segmente mai privilegiate



Selector - selectorul rutinei apelate

Offset - adresa relativa a rutinei apelate

Nr. cuvinte - nr. cuvintelor duble copiate pe stiva(fiecare nivel de prioritate are stiva proprie)

Tip - poarta de apel, de task, de intrerupere sau de exceptie (trap)

Tabele de descriptori

- GDT - General Descriptor Table
  - contine descriptorii pentru segmente comune
  - o singura tabela GDT pe sistem
- LDT - Local Descriptor Table
  - contine descriptorii pentru segmentele alocate unui task
  - fiecare task are propriul tabel LDT
- IDT - Interrupt Descriptor Table
  - contine descriptorii segmentelor care contin rutinele de tratare a intreruperilor
  - un singur tabel pe sistem

Nivele de prioritate

- Asigura protectia impotriva unor accese neautorizate
- 4 nivele de prioritati:
  - 0 - cel mai prioritar, 3- cel mai putin prioritar
  - nivelul 0: nucleul sistemului de operare
    - toate instructiunile protejate sunt permise
  - nivelul 1: rutine de sistem
  - nivelul 2: extensiile sistemului de operare
  - nivelul 3: programe utilizator

Nivele de prioritate

- RPL - Requested Privilege Level
  - nivelul de prioritate solicitat
- CPL - Current Privilege Level
  - nivelul curent de privilegiu (continut in selectorul din CS)
- DPL - Descriptor Privilege Level
  - nivelul de privilegiu al unui segment - se pastreaza in campul DPL al descriptorului
- Restrictii de acces:
  - conditia de acces: CPL < sau = DPL

Registre speciale folosite in modul protejat

- GDTR - General Descriptor Table Register
  - contine adresa tabelei GDT (32b) si lungimea tabelei (16b)
- LDTR - Local Descriptor Table Register
  - contine selectorul tabelei LDT (descriptorul tabelei se afla in GDT)
- TR - Task Register
  - contine selectorul tabelei TSS

Registre speciale folosite in modul protejat

- CR0 - Control Registre 0
  - contine urmatoarele indicatoare de conditie:
    - PG - page - validare/invalidare paginare
    - ET - extension type - indica tipul coprocesorului matematic (0-80287, 1 80387)
    - TS - task switched - setat la comutarea unui task
    - MP - math present
    - PE -protected mode enabled - validare mod protejat

Instructiuni pt. registrele speciale

- Incarcare registre:
  - LGDT <registru>|<<memorie>
  - LLDT <registru>|<<memorie>
  - LIDT <registru>|<<memorie>
  - MOV CR0, <registru>
- Salvare registre:
  - SGDT <registru>|<<memorie>
  - SLDT <registru>|<<memorie>
  - SIDT <registru>|<<memorie>

Trecerea din modul real in modul protejat

- La punerea sub tensiune procesorul trece implicit in modul real !!!
- Secventa de trecere in modul protejat:
  - se construiesc tabelele de descriptori ce urmeaza sa se foloseasca (GDT, LDT, IDT)
  - se invalideaza intreruperile (daca se fol. intreruperi)
  - se incarca adresele tabelelor in registrele speciale corespunzatoare (GDTR, LDTR, IDTR)
  - se seteaza bitul PE din registrul CR0
  - se incarca registrele segment cu selectoarele dorite
  - se incarca CS cu un selector prin executia unei instructiuni de salt “far”

Revenirea din modul protejat

- La procesorul 286 numai prin resetare
  - la IBM-PC - metoda complicata de revenire, (prin secventa controlata de resetare)
- La procesoarele 386, 486, ... prin stergerea indicatorului PE din CR0
  - trebuie luate masuri pentru revenirea corespunzatoare in modul real:
    - registrele segment trebuie incarcate cu adresele segmentelor corespunzatoare modului real
    - segmentele trebuie sa fie de 64ko

Secventa de revenire din modul protejat (numai la proc. 386, 486, ...)

- invalidare intreruperi
- salt “far” la un segment de cod cu limita 64k (FFFFH)
- incarca SS cu un selector potrivit pt. modul real:
  - limita 64k, granularitatea = 0 (octert)
  - expandare in sus (E=0), validare scriere (W=1)
  - seg. prezent (P=1)
- sterge PE din CR0
- salt “far” la o adresa <16>:<16>
- incarca toate registrele segment de date cu valori corespunzatoare modului real

Secventa de revenire din modul protejat (continuare)

- daca s-au folosit intreruperi in modul protejat se foloseste LIDT pt. a incarca un IDT potrivit modului real:
  - adresa de baza =0, limita 3ffh
- sterge partea superioara a registrilor generali de 32 biti
- validare intreruperi

Tehnici de protectie a acesului in modul protejat

- Ce se protejeaza:
  - executia unor instructiuni protejate (ex: validarea/invalidarea intreruperilor STI/CLI, operatii de intrare/iesire - IN/OUT)
  - accesul unui task la segmentele alocate altui task (cod sau date)
  - apelul necontrolat al unor functii sistem si coruperea unor date privilegiate (ale sistemului de operare)
- Cum actioneaza protectia:
  - se genereaza o exceptie in cazul violarii unei protectii

Mecanisme de protectie

- Accesul la memorie prin descriptori pastrati in GDT si LDT
  - GDT pastreaza descriptorii segmentelor accesibile mai multor taskuri
  - LDT pastreaza descriptorii segmentelor alocate numai unui singur task => segmente protejate
- Nivele de privilegiu:
  - 4 nivele, 0 cel mai prioritat, 3 cel mai putin prioritar
  - nivelele inferioare alocate sistemului de operare
  - un task de prioritate mai mica (nuvel mai mare) nu poate accesa segmente care au prioritate mai mare

Mecanisme de protectie

- Nivele de prioritati (continuare)
  - la trecerea in modul protejat nivelul curent de prioritate (CPL) devine 0 (se considera ca se executa o secventa a sistemului de operare)
  - se pot face numai salturi “far” la segmente care au prioritate mai mare sau egala (CPL_vechi<= DPL_nou)
  - se pot apela segmente de date care au prioritate mai mare sau egala cu prioritatea curenta
  - in cazul unei tentative de acces la un segment mai prioritar se genereaza o exceptie, tratata de sistemul de operare

Mecanisme de protectie

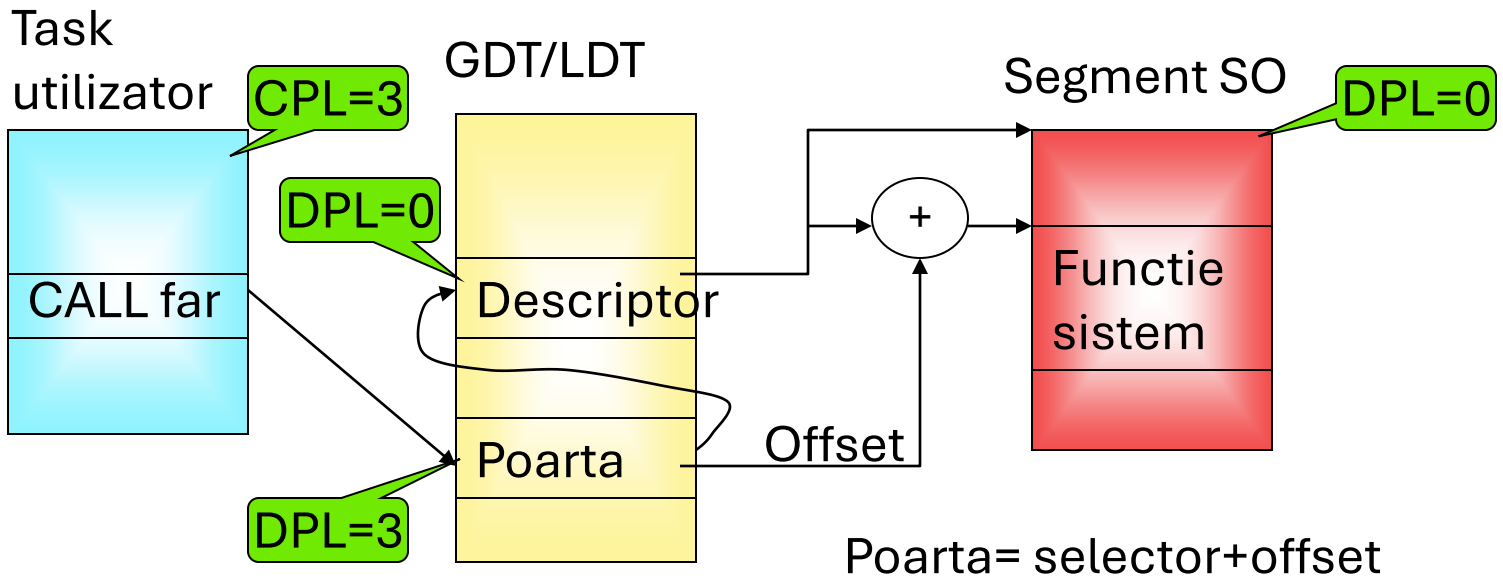
- Accesul la segmente mai privilegiate

(ex: apelul unor functii ale sistemului de operare)

  - solutia: porti de acces (Call gates):
    - descriptori speciali care contin un selector si un offset al functiei apelate
    - descriptorul de poarta are de obicei nivel mai mare de privilegiu (privilegiu mai mic) pentru a permite accesul tuturor taskurilor la poarta respectiva de acces
    - descriptorul de poarta poate fi inclus in GDT sau in LDT

Mecanisme de protectie

- Porti de acces (continuare)
  - selectorul continut in descriptorul portii are un nivel de privilegiu mai mic (privilegiu mai mare) pt. a permite accesul la segmente protejate



Mecanisme de protectie

- Controlul operatiilor efectuate asupra unui segment:
  - numai executie (segment de cod)
  - numai executie & citire date (segment de cod)
  - numai citire date (segment de date)
  - citire&scriere date (segment de date)
- Controlul dimensiunii segmentului:
  - se verifica daca adresa de offset (adresa relativa in cadrul segmentului) este mai mica decat limita segmentului

Exemplu de trecere in modul protejat

```
    ; se fol. instructiuni privilegiate
    .386
;macrouri pt. instr. pe 32 biti
LGDT32 MACRO Adr
    DB      66h  ; prefix 32 biti
    DB      8Dh       ;lea eax, Adr
    DB  1Eh
    DD   Adr
    DB      0Fh       ; LGDT  [BX]
    DB   01h
    DB      17h
ENDM
```

```
; macro pt. salt far 32 biti in seg de
; 16 biti
FJMP      MACRO  Selector, Offset
    DB      66H       ; prefix 32 biti
    DB      0EAH  ; jump far
    DD      Offset   ; offset pe 32 biti
    DW     Selector
ENDM
```

Exemplu de trecere in modul protejat

```
_TEXT   SEGMENT PARA USE32
    ASSUME CS: _TEXT
_ENTRY:
;incarca descriptor GDT
    LGDT32  fword ptr GdtDesc
    MOV  EAX, CR0
    OR      AX,1
    MOV  CR0, EAX  ; PE=1
    JMP     $+2  ; descarca coada de
    ;instructiuni
;acum se executa in mod protejat in
; segment de 16 biti
; se face sal la segment de 32 biti
    FJMP32   08,Start32
```

```
;mod protejat,segment de 32 biti
Start32:   ; init. reg. segmente cu
;intrarea nr. 2 di GDT - adr=10h
    MOV  AX, 10h
    MOV  DS, AX
    MOV  ES, AX
.......
    MOV  SS, AX
    MOV   ESP, 8000h
.............
```

Exemplu de trecere in modul protejat

```
GdtDesc:
    DW   dim_GDT-1  ; limita GDT
    DD      GDT

    ALIGN 4
; tabela GDT
GDT:
; GDT[0] intrarea 0 - nu se
    foloseste
    DD   0
    DD      0
; GDT[1] descriptor pt. seg. de cod
;limita FFFFF, granularitate 1 (4ko)
;dim. segment=4Go DW  0FFFFh  ;
    limita 15..0
    DW  0   ; Baza 15..0
    DB   0    ; Baza 23..16
    DB   10011010B
;P=1, DPL=0, S=1, 1, C=0, R=1,A=0
    DB      11001111B
;G=1, D=1, 0 0 Lim 19..16
    DB      0  ; Baza 31..24
```

Exemplu de trecere in modul protejat

```
; GDT[2] descriptor pt. seg. de date limita FFFFF
   DW    0FFFFh  ; limita 15..0
   DW  0   ; Baza 15..0
   DB   0    ; Baza 23..16
   DB   10010010B
;P=1, DPL=0, S=1, 0, C=0, R=1,A=0
   DB     11001111B
;G=1, D=1, 0 0 Lim 19..16
   DB     0  ; Baza 31..24
dim_GDT   EQU $- offset GDT
_TEXT   ENDS
            END
```