15 Ianuarie 2025

- rolul principal al unui asamblor= generarea corespunzatoare de octeti

- la un moment dat poate fi activ numai unul din segmentele de fiecare tip(ex. doar singur segment de cod activ din N)

- sub 16 biti registrii segment cs, ds, ss, es contineau ADRESELE DE INCEPUT ale segmentelor active la un moment dat

- sub 32 biti registrii segment cs, ds, ss, es contin valorile SELECTORILOR de segment active

- la orice moment al executiei combinatia de registrii cs:eip exprima /contine adresa instructiunii curente de executat

- aceste valori sunt manipulate exclusiv de catre BIU

- in cadrul unei instructiuni NU putem avea ambii operanzi expliciti din memoria RAM

- BIU poate "aduce" numai un singur operand din memorie odata (ne-ar trebui 2 BIU, 2 seturi de registrii)

$$\textit{adresa\_offset} = [\text{ bază }] + [\text{ index} \times \text{scală }] + [\text{ constanta }]$$
$$\text{(SIB)} \qquad \text{(deplasament + imediat)}$$

FORMATUL INTERN AL UNEI INSTRUCTIUNI ESTE:

[prefixe] + cod + [ModR/M] + [SIB] + [deplasament] + [imediat]

- octetul ModR/M poate exprima operanzi de tip registru şi/sau de tip memorie de adresare indirectă în care apare doar [baza]; dacă apare şi partea de [index*scală] atunci este nevoie SI de octetul SIB !! Deci: dacă Modr/m ne spune ca operandul e in memorie => octetul SIB apare obligatoriu NUMAI DACA AVEM SI PARTEA DE [index × scală], urmat EVENTUAL si de deplasament si/sau imediat

- Deci primele 2 elemente din formula de calcul a offsetului unui operand (baza şi index*scala) sunt exprimate sau precizate prin octeţii ModR/m şi SIB din formatul intern al unei instructiuni

- al treilea element: constanta, daca apare, este exprimata de campurile deplasament sau/şi imediat (displacement and/or immediate)

- daca Modr/m imi spune ca am doar registru pe post de operand urmatoarele 3 campuri din formula nu mai apar (deoarece daca operandul este registru NU poate fi in acelasi timp si operand din memorie si operand imediat)

- campul imediat poate pe de o parte sa participe la calculul offsetului unui operand din memorie (furnizand campul constanta din formula offsetului) sau poate sa apara de sine statator exprimand valoarea imediata a unui operand (Ex: mov ebx, 12345678)

- campul deplasament dacă apare singur în formula de calcul a offset-ului exprima modul de adresare directă la memorie

- campul imediat=constante numerice

- adresarea directă presupune accesul direct la operandul din memorie pe baza deplasamentului sau, fara ca in formula de calcul a offsetului sa apara vreun registru (deci fara baza sau index !)

- daca apar registrii in calculul offsetului => adresare indirecta

- in cadrul instructiunilor in care vom folosi doar exprimari de offseturi, acestea (offseturile) vor fi prefixate in mod IMPLICIT de către asamblor de unul dintre regiştrii de segment CS, DS,SS sau ES. (ex. in debugger push variabila -> DS:[40100...])

CS:EIP – Adresa FAR (completa) a instructiunii curente de executat

EIP – incrementat automat dupa executia instructiunii curente

CS – contine selectorul de segment a segmentului de cod curent (activ) si poate fi modificat numai daca executia “sare” intr-un alt segment (JMP FAR..)

Mov cs, [var] – ok sintactic (illegal instruction in OllyDbg...)

Mov eip, eax – syntax error – symbol ‘eip’ undefined

Jmp FAR undeva_in_memorie; se modifica şi CS şi EIP !
Jmp start1 ; salt NEAR – se modifica doar offset-ul, deci numai EIP !