CURS 3 - 19 oct. 2022

segment offset → orice segment începe la m_{16}.
(deplasament)
↳ 16 biți

În orice moment al execuției, [a] → conținutul de la adresa.

CURS 4 - 26 oct. 2022

1 octet = 256 de valori

MOD R/M → tipul operandului

nume_instr destinație, sursă

Formula offsetului în memorie (de la 2 noaptea)

adresa_offset = [bază] + [index × scală] + [constantă]

ADRESARE INDIRECTĂ — ADRESARE DIRECTĂ

- bază - registru general (din cei 8)
- index - registru general (oricare din cei 8 - ESP)
- scală - 1, 2, 4, 8
- constantă - orice val. determinabilă la momentul asamblării

[] → operator de dereferențiere, dar în formulă exprimă opționalitatea

Exemplu

```
mov eax, [ebx]  → un offset | bază (usually) sau index
mov eax, ebx    → factor de index
mov eax, [2*edx-4]
mov ebx, [ecx+4*edx+18]
                  ↳ registru de bază
mov eax, 23
mov eax, [23]
mov eax, ebx
mov edx, [ecx+2*esp+4] ⇒ EROARE
                  ↳ nu poate fi index
```

[ecx + 4 * edx + 18]
- ecx → registru de bază
- edx → index
- 18 → constantă

! pe 10 dec. se dă identificarea formulei de la 2 noiembrie

mov [eax], 23 — este corectă
(se ia 23 și se mută în memorie)

mov [eax], [23] — sintaxă incorectă
(procesul nu poate calcula la nivelul memoriei)

Formula de la 2 noiembrie nu se aplică unde nu există [ ].

③

mov eax, [ebx+2]

mov eax, [ebx * 3] ⇔ mov eax, [ebx+ebx*2]
este acceptată (merge doar nr. m. impare x·3=x+x·2)

mov eax, [ebp*7] nu este corectă

mov eax, [esp*5] este corectă

Offsetul oricărei variabile definită în program este întotdeauna o valoare constantă determinabilă la momentul asamblării.

[prefix] + cod + [Mod R/M] + [SIB] + [deplasament] + [imediat]

mov eax, [v] —în debugger→ mov eax, dword ptr [DS:0084]
pointer.

Regiștri - segment: CS, DS, SS, ES, FS, GS

„La programator, pot să le modif. GS și EIP"
→ DA (Întrebare din 10 doc.) → jump for.