Arhitectura sistemelor de calcul

Curs 6 - registrii de adresă

Arhitectura x86 are 4 tipuri de segmente:

CS: segmentul de cod activ curent care conține instrucțiuni

DS: segmentul de date activ curent care conține date

SS: segmentul de stivă activ curent (generat de asamblor)

ES: segmentul suplimentar de date activ curent

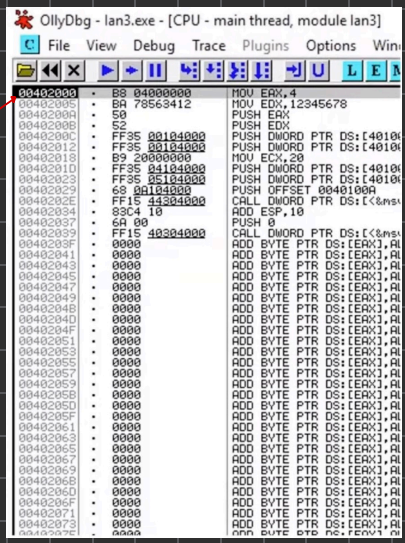
Orice program este obligatoriu să conțină SS deoarece execuția unui program are loc la nivelul stivei de execuție și CS pentru a face posibilă asamblarea. Datele se pot defini în segmentul de cod cu condiția ca procesorul să nu ajungă să execute acele linii deoarece generarea acelor este sarcina asamblorului.

```
Segment code
    Start:
        Jmp Real  start
        V db 17
        V1 dw 54321
    Real  Start:
        Mov ax, [v]
        Add ebx, eax
        Mul [v1]
        .....
```

} procesorul nu ajunge să execute acest cod

Cel mult un singur segment de cod, un singur segment de date, un singur segment de stivă și un singur segment suplimentar este activ la un moment.

Oricare dintre registrii CS, DS, SS și ES conțin valorile selectorilor segmentelor active. Registrul EIP conține offset-ul instrucțiunii curente în cadrul segmentului de cod curent el fiind gestionat de BIU.



EIP adică offsetul instrucțiunii curente din segmentul de cod

La orice moment al execuției, combinația de registrii CS:EIP exprimă adresa instrucțiunii curente de executat. Aceste valori sunt manipulate de BIU.

Nu se poate modifica EIP-ul și nici CS astfel:

```
MOV CS, [var]   X
MOV EIP, EAX    X
```

EIP-ul și CS pot fi modificate astfel:

JMP segment:offset ✓ - se modifică CS și EIP (salt FAR)

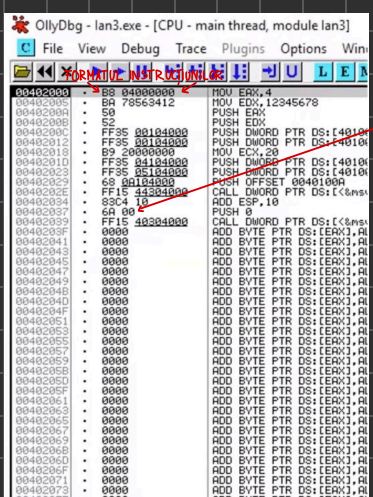
JMP offset ✓ - se modifică doar EIP (salt NEAR)

Orice instrucțiune are maximum un operand din memoria RAM, deoarece BIU poate calcula adresa unui singur operand din memorie.

Formatul intern a unei instrucțiuni poate ocupa între 1 și 15 octeți și este de forma:

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

Nr. de biți: 1 | 1/2/3 | 1 | 1 | 1/2/4 | 1/2/4



PUSH 0 are formatul instrucțiunii cod + [imediat], adică 6A 00

Mod R/M - octet care apare în formulă dacă instrucțiunea are un operand care este registru sau este din memorie.

SIB - octet care apare dacă un operand este din memorie și participă la calculul offsetului operandului după formula:

offset = [bază] + [index · scală] + [constantă]

SIB calculează partea de [bază] și/sau [index · scală], și este un octet de forma $B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0$ (scale, index, base)

deplasament - apare dacă un operand este din memorie și intervine în calculul adresei operandului din memorie și exprimă modul de adresare directă la memorie

imediat - poate să participe la calculul offsetului unui operand din memorie furnizând câmpul constantă din formula offsetului sau poate să apară de sine stătător exprimând valoarea imediată a unui operand. (ex: MOV EAX, 4)

Primele două elemente din formula de calcul a offsetului operand sunt precizate prin octetul SIB din formatul unei instrucțiuni, cel de al 3-lea element din formulă, dacă apare este exprimat de octeții deplasament și imediat din formatul unei instrucțiuni.

Ultimele 3 elemente din formatul unei instrucțiuni sunt elementele care participă la calculul offset-ului operandului identificat din memorie.