Arhitectura sistemelor de calcul

Curs 8 - elementele limbajului de asamblare

Limbajul mașină este format din instrucțiunile sub forma unor șiruri de biți cu semnificație.

Limbajul de asamblare este unul simbolic în care instrucțiunile în cod binar au reprezentare textuală.

Elementele de bază ale limbajului de asamblare:

etichete: nume scrise de utilizator cu ajutorul cărora se pot referi date (nume de variabile) sau zone de memorie (nume de salturi)

instrucțiuni: sunt scrise sub formă de text și asamblorul generează octeți care codifică instrucțiunea în formatul intern al unei instrucțiuni.

directive: sunt indicații date asamblorului pe care le folosește în scopul generării octeților (db, dw, dd sau segment code și segment data)

contor de locații: este un număr gestionat de asamblor și la fiecare moment valoarea lui coincide cu

numărul de octeți generați corespunzător instrucțiunilor și directivelor deja întâlnite în cadrul segmentului respectiv, deci este offsetul curent. Acest număr poate fi accesat prin simbolul $.

$$ reprezintă offsetul începutului secțiunii.

```
section .data (valoarea lui .data = $$)
    db 'hello'
    db  'h', 'e', 'l','l','o'
    data_segment_size  equ $-$$
```

=> data segment size = 10

Dacă nu se folosește directiva section atunci simbolul $$ va avea implicit valoarea începutului segmentului.

Formatul unei linii sursă:

[etichetă[:]] [prefixe] [mnemonică] [operanzi] [; comentariu]

etichetele pot fi constituite din litere, cifre, caractere, acestea fiind case sensitive (Abc ≠ abc)

la etichetă apar: dacă ne referim la o etichetă de cod

loop:

la etichetă nu apar: dacă ne referim la o etichetă de date

a db 20

mnemonicile sunt directive sau nume de instrucțiuni

Exemple:

```
aici: jmp acolo          ; etichetă + mnemonică + operand + comentariu
      repz cmpsd         ; prefix + mnemonică + comentariu
      start:             ; etichetă + comentariu
                         ; comentariu
      a dw 19872,42h     ; etichetă + mnemonică + operanzi + comentariu
```

Offsetul etichetelor de date și de cod reprezintă valori determinabile la momentul asamblării care rămân constante pe parcursul execuției.

Extragerea de conținut a unei variabile se face folosind operatorul de dereferențiere [], fără acest operator variabila va fi o adresă.

Exemple:

```
mov eax, v      ; încarcă în eax adresa marcată cu eticheta v (4 octeți)
mov eax, [v]    ; încarcă în eax conținutul de la adresa v (4 octeți)
lea eax, [v]    ; încarcă în eax adresa variabilei v (4 octeți)
```

Directivele dirijează asamblorul, iar instrucțiunile dirijează procesorul.

Expresiile sunt evaluate în momentul asamblării, adică valorile lor sunt determinabile la momentul asamblării.

Valoarea operanzilor este calculată / determinată la momentul asamblării pentru operanzii imediați (constante) și pentru operanzii cu adresare directă (adrese). În momentul încărcării programului fiecare adresă FAR se completează de către sistemul de operare.

???: offset (assembly time)
↓
0708: offset (loading time)

Locul alocării unei variabile (adresa sa) rămâne fix. De aceea, offseturile variabilelor reprezintă valori constante la momentul asamblării, adresa de segment va fi determinabilă la momentul încărcării programului.

Operanzii din memorie cu:
- adresare directă sunt constante, offseturi, etichete, funcții
- adresare indirectă se află între []

Orice offset utilizat în cadrul unui program va fi în cele din urmă la o adresă FAR prin prefixarea sa cu o valoare de segment. Această valoare va fi în totdeauna un registru de segment (CS, DS, SS), după următoarele reguli:

- CS pentru etichete de cod destinație ale unor salturi (jmp, call, ret)
- SS pentru adresări SIB care lucrează cu stiva, adică se folosește EBP sau ESP drept bază
- DS pentru restul accesărilor de date