Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Programare în limbaj de asamblare

54. Definirea şi utilizarea de macroinstrucțiuni.

Definirea şi utilizarea de macroinstrucţiuni

O macroinstrucțiune (MI) reprezintă o secvență de instrucțiuni căreia i se asociază un nume. Apariția macroinstrucțiunii în textul programului este înlocuită, automat, de către asamblor, cu secvența de instrucțiuni asociată. Ea poate să fie parametrizată, adică poate fi definită în funcție de o serie de parametrii formali, care sunt actualizați la utilizarea macroinstrucțiunii, fiind, deci, înlocuiți cu parametrii actuali.

Diferențe între MI şi procedură: codul obiect al unei proceduri apare o singură dată în program, în schimb codul obiect generat pentru o MI este repetat în program, ori de câte ori apare numele macroinstrucțiunii. Procedura reduce atât dimensiunea programul sursă, cât şi pe cea a celui obiect, în timp ce MI nu reduce codul obiect, ci doar dimensiunea programului sursă. MI nu modifică timpul de execuție al programului, pe când utilizarea procedurilor creşte timpul de execuție.

Utilizare macroinstrucțiunilor:
- simplificarea şi reducerea secvențelor de cod ce se repetă;
- reducerea erorilor cauzate de codul repetitiv;
- program în limbaj de asamblare mai uşor de urmărit.

```
< nume_macro >    MACRO      [ lista_parametri_formali ]
        < declarații/ corpul macroinstrucțiunii > [;comentarii]
                   ENDM
        aduna macro t1,t2,suma
                   mov   ax, t1
                   add   ax, t2
                   mov   suma, ax
              endm
```

Utilizarea se poate face astfel:

```
aduna a, b, c   →          mov   ax, a
                           add   ax, b
                           mov   c, ax
```

sau

```
aduna bx,cx,dx      →      mov   ax, bx
                           add   ax, cx
                           mov   dx, ax
```

Dacă în definirea unei macro, se utilizează etichete, acestea trebuie declarate ***LOCAL***. De exemplu dacă dorim să realizăm un ciclu, o secvență de forma:

```
              mov   cx, durata_ciclu
        iar:  loop  iar
```

va genera erori de multidefinire a etichetei la utilizările ulterioare.

```
        delay  macro durata
               LOCAL      iar
               push  cx   ;; se salvează registrul CX
               mov   cx, durata
        iar:   loop  iar
               pop   cx
               endm
        LOCAL      < lista_etichete >
```

Rolul său este de a declara simbolurile din listă ca locale pentru fiecare apel. Directiva se poate utiliza numai în MI, şi precede instrucțiunile din corpul MI. Pentru aceste etichete asamblorul va genera, la expandare, nume de etichete succesive:*??0000, ??0001*, etc.

1) Calculul puterii întregi a unui număr întreg.

```
putere macro numar, exponent
        local iar, gata
        push cx         ; salvare (CX) şi (BX)
        push bx
        xor dx, dx      ; rezultatul este returnat în (DX, AX)
        mov ax, 1       ; pregătire rezultat, dacă exponentul e 0
        mov cx, exponent
        jcxz gata       ; dacă CX=0, puterea este 1
        mov bx, numar       ; înmulțitorul în BX
iar:    mul bx
        jc gata         ; dacă apare eroare la *, se poziționează CF
        loop iar
gata:   pop     bx      ; refacerea registrelor salvate
        pop cx
        endm  ; programul ce utilizează această macro, va testa
              ; valoarea lui CF, dacă este 0 valoare corectă în
              ; (DX,AX), dacă însă CF=1, a apărut depăşire
```

2) Înmulțirea unei valori cu 10.

```
ori10   macro x
local gata
        push ax
        push bx
        mov ax, x
        shl ax, 1               ; * 2
        jc gata
        mov bx, ax
        shl ax, 1               ; * 4
        jc gata
        shl ax, 1               ; * 8
        jc gata
        add ax, bx
        jc gata         ; după utilizare macro ori10 se va
        mov     x, ax   ; testa indicatorul CF pentru a testa
gata:   pop     bx      ; o eventuală depăşire
        pop     ax
        endm
```