Szimbólum

A szimbólum lehet szimbolikus konstans, változó név vagy címke.

Szimbolikus konstans: Az **=** vagy az **EQU** pszeudo utasítással definiálható. Szimbolikus szöveg konstans csak **EQU**-val definiálható. A szimbolikus konstans a program szövegnek a definíciót követő részében használható, értékét a használat helyét megelőző utolsó definíciója határozza meg.

Ha egy szimbólumot **EQU**-val definiálunk, akkor ezt a szimbólumot a modulban másutt nem definiálhatjuk!


```
S       =       1       ; S értéke 1
N       EQU     14      ; N értéke 14
        MOV     CX,N    ; CX ⇐ 14
ISM:
S       =       S+1 ; S értéke ezután 2, függetlenül
                ; attól, hogy hányadszor fut a ciklus
        MOV     AX,S    ; AX ⇐ 2
        LOOP    ISM
N       =       5       ; hibás
N       EQU     5       ; hibás
S       =       5       ; helyes
S       EQU     5       ; hibás
```


Szimbolikus konstansként használhatjuk a **$** jelet (helyszámláló), melynek az értéke mindenkor a program adott sorának megfelelő **OFFSET** cím. A helyszámláló értékének módosítására az **ORG** utasítás szolgál, pl.:

```
ORG     $+100H ; 100H byte kihagyása
               ; a memóriában
```


Címke: Leggyakoribb definíciója, hogy valamelyik utasítás előtt a sor első pozíciójától **:** -tal lezárt azonosítót írunk. Az így definiált címke **NEAR** típusú. Címke definícióra további lehetőséget nyújt a **LABEL** és a **PROC** pszeudo utasítás:

```
ALFA:   ...         ; NEAR típusú

BETA    LABEL FAR   ; FAR típusú
GAMMA:  ...         ; BETA is ezt az utasítást
                    ; címkézi, de GAMMA NEAR típusú
```


Az eljárás deklarációt a **PROC** pszeudo utasítással nyitjuk meg. A címke rovatba írt azonosító az eljárás neve és egyben a belépési pontjának címkéje. Az eljárás végén az eljárás végét jelző **ENDP** pszeudo utasítás előtt meg kell ismételnünk ezt az azonosítót, de az ismétlés nem minősül címkének. Az eljárás címkéje aszerint **NEAR** vagy **FAR** típusú, hogy maga az eljárás **NEAR** vagy **FAR**. Pl.:

```
A       PROC            ; NEAR típusú
        ...
B       PROC    NEAR    ; NEAR típusú
        ...
C       PROC    FAR     ; FAR típusú
        ...
```


Címkére vezérlés átadó utasítással hivatkozhatunk, **NEAR** típusúra csak az adott szegmensből, **FAR** típusúra más szegmensekből is.

Változó: Definíciója adat definíciós utasításokkal történik. Néha (adat) címkének is nevezik.

Kifejezés

A kifejezés szimbólumokból és konstansokból épül fel az alább ismertetendő műveletek segítségével. Kifejezés az operátorok, pszeudo operátorok operandus részére írható.

Értékét a fordítóprogram határozza meg, és a kiszámított értéket alkalmazza operandusként. Szimbólumok értékén konstansok esetében természetesen a konstans értékét, címkék, változók esetében a hozzájuk tartozó címet – és nem a tartalmat – értjük.

Kifejezés

A kifejezés értéke nemcsak számérték lehet, hanem minden, ami az utasításokban megengedett címzési módok valamelyikének megfelel. Pl. **[BX]** is kifejezés és értéke a **BX** regiszterrel történő indirekt hivatkozás, és ehhez természetesen a fordító programnak nem kell ismernie **BX** értékét.

Természetesen előfordulhat, hogy egy kifejezés egyik szintaktikus helyzetben megengedett, a másikban nem, pl.:

```
mov     ax,[BX]; [BX] megengedett
mul     [BX]   ; [BX] hibás, de
mul     WORD PTR [BX] ; megengedett
```

Egy kifejezés akkor megengedett, ha az értéke **fordítási időben meghatározható** és az adott szintaktikus helyzetben alkalmazható, pl. az adott utasítás lehetséges címzési módja megengedi. A megengedett kifejezés értékeket az egyes utasítások ismertetése során megadtuk.

A műveletek csökkenő precedencia szerinti sorrendben:

1. **( )** és **[ ]** (zárójelek) továbbá **< >**: míg a **( )** zárójel pár a kifejezés kiértékelésében csupán a műveletek sorrendjét befolyásolja, addig a **[ ]** az indirekció előírására is szolgál. Ha a **[ ]** -en belüli kifejezésre nem alkalmazható indirekció, akkor a **( )** -lel egyenértékű;
   - **LENGTH változó**: a **változó**-hoz tartozó adat terület elemeinek száma;
   - **SIZE változó**: a **változó**-hoz tartozó adat terület hossza byte-okban;
   - **WIDTH R/F**: az **R** rekord vagy az **F** (rekord) mező szélessége bitekben;
   - **MASK F**: az **F** (rekord) mező bitjein **1**, másutt **0**;

Pl.:

```
v       dw      20 dup (?)
rec     record  x:3,y:4
table   dw      10 dup (1,3 dup (?))
str     db      "12345"
```

esetén:

```
        mov     ax,LENGTH v       ; ax ⇐ 20
        mov     ax,LENGTH rec     ; ax ⇐ 1
        mov     ax,LENGTH table   ; ax ⇐ 10
                          ; a belső DUP ignorálva!
        mov     ax,LENGTH str     ; ax ⇐ 1
                          ; str egy elem
```



```
v       dw      20 dup (?)
rec     record  x:3,y:4
table   dw      10 dup (1,3 dup (?))
str     db      "12345"
```

esetén:

```
        mov     ax,SIZE v         ; ax ⇐ 40
        mov     ax,SIZE rec       ; ax ⇐ 1
        mov     ax,SIZE table     ; ax ⇐ 20
                          ; a belső DUP ignorálva
        mov     ax,SIZE str       ; ax ⇐ 1
                          ; str bájtos
```

```
v       dw      20 dup (?)
rec     record  x:3,y:4
table   dw      10 dup (1,3 dup (?))
str     db      "12345"
```

esetén:

```
        mov     ax,WIDTH rec   ; ax ⇐ 7
        mov     ax,WIDTH x     ; ax ⇐ 3
        mov     ax,MASK x      ; ax ⇐ 70H
```


2. **.** (pont): struktúra mezőre hivatkozásnál használatos;

3. **:** mező szélesség (rekord definícióban) és explicit szegmens megadás (segment override prefix). Az explicit szegmens megadás az automatikus szegmens regiszter helyett más szegmens regiszter használatát írja elő, pl.:

```
mov     ax, ES:[BX]
            ; ax ⇐ (ES:BX) címen lévő szó
```

Nem írható felül az automatikus szegmens regiszter az alábbi esetekben:

- **CS** program memória címzésnél,
- **SS** stack referens utasításokban (**PUSH**, **POP**, **...**),
- **ES** string kezelő utasításban **DI** mellett, de az **SI**-hez tartozó **DS** átírható.

4.

- **típus PTR cím:** (típus átdefiniálás) ahol **típus** lehet **BYTE**, **WORD**, **DWORD**, **QWORD**, **TBYTE**, illetve **NEAR**, **FAR** (előre hivatkozás esetén fontos) és **PROC**. Pl.:

```
        MUL     BYTE PTR [BX] ; a [BX] címet
                              ; byte-osan kell kezelni
```

- **OFFSET kifejezés:** a **kifejezés OFFSET** címe (a szegmens kezdetétől számított távolsága byte-okban);
- **SEG kifejezés:** a **kifejezés** szegmens címe (abban az értelemben, ahogy a szegmens regiszterben szokásos tárolni, tehát valós üzemmódban a szegmens tényleges kezdőcímének **16**-oda);

- **TYPE változó:** az elemek hossza byte-okban, ha változó, de

```
TYPE string = 1,
TYPE konstans = 0,
TYPE NEAR címke = -1,
TYPE FAR címke = -2

JMP   (TYPE cím) PTR [BX]
   ; NEAR vagy FAR ugrás attól függően, hogy cím
   ; közeli vagy távoli címke
```

- **... THIS típus:** a program szöveg adott pontján adott típusú szimbólum létrehozása;

Pl.:

```
ADATB       EQU     THIS BYTE
            ; BYTE típusú változó, helyfoglalás nélkül
ADATW       dw      1234H
            ; ez az adat ADATB-vel byte-osan érhető el
. . .
  mov   al,BYTE PTR ADATW ; al ⇐ 34H, helyes
  mov   al,ADATB          ; al ⇐ 34H, helyes
  mov   ah,ADATB+1        ; ah ⇐ 12H, helyes
```

Emlékeztetünk arra, hogy szavak tárolásakor az alacsonyabb helyértékű byte kerül az alacsonyabb címre!