386. mikroprocesora aritmētika

Direktīva .386

```
.code
.startup
.386
```

Operanda paplašināšana *kopā ar zīmi* (**MovSx**)

MOVe with **S**ign e**X**tension

```
NegTwoB     DB   -2
...
Mov    Al,   NegTwoB   ;Ax   = ??FE

MovSx  Cx,   Al        ;Cx   = FFFE (Ch = FF)
MovSx  Bx,   NegTwoB   ;Bx   = FFFE (Bh = FF)
MovSx  EDx,  NegTwoB   ;EDx  = FFFF FFFE
```

Operanda paplašināšana *bez zīmes* (**MovZx**)

MOVe with **Z**ero e**X**tension

```
NegTwoB     DB   -2
...
Mov    Al,  NegTwoB    ;Ax  = ??FE
MovZx  Cx,  Al         ;Cx  = 00FE (Ch=00)
MovZx  Bx,  NegTwoB    ;Bx  = 00FE (Ch=00)
MovZx  EDx, NegTwoB    ;EDx = 000000FE
```

Kļūdas

```
MovSx  Cl,  Bl     ;operandu izmēri sakrīt
MovSx  Cx,  Bx     ;operandu izmēri sakrīt
MovSx  ECx, EBx    ;operandu izmēri sakrīt

MovSx  MemW, MemB  ;formāts "atmiņa" - "atmiņa"

MovSx  ECx, 2      ;jāizmanto komanda Mov
```

Informācijas pārbaude reģistros EAx, EBx, …

RoL: **RO**tate **L**eft (8086/8088)

```
Mov EAx, 0
Rol EAx, 16
Rol EAx, 16
```

```
mov     eax,00000000
rol     eax,10
rol     eax,10
```

Rezultāti: ax 0000 ⇨ ax 0000 ⇨ ax 0000

```
Mov EAx, -2
Rol EAx, 16
Rol EAx, 16
```

```
mov     eax,FFFFFFFE
rol     eax,10
rol     eax,10
```

Rezultāti: ax FFFE ⇨ ax FFFF ⇨ ax FFFE

Piezīme: var izmantot arī komandu **RoR** (**RO**tate **R**ight)

Divu operandu *reizināšana* (**IMul**)

Reizināšanai var izmantot **ne** tikai akumulatoru.

```
Mov   Bx, 3     ;Bx = 0003
Mov   Cx, -2    ;Cx = FFFE
IMul  Bx, Cx    ;Bx = FFFA (-6)
```

ax	0000
bx	FFFA
cx	FFFE
dx	0000

Piezīme: reģistrs Dx **ne**bija izmainīts (nav vērtības FFFF)

```
X       DW 3
...
Mov   Bx, -2    ;Bx = FFFE
IMul  Bx, X     ;Bx = FFFA
```

```
Mov   EBx, -2   ;EBx = FFFF FFFE
IMul  EBx, EBx  ;EBx = 0000 0004
```

```
X       DD -2       ;Define Double word (4 baiti)
...
Mov     EBx, 3      ;EBx = 0000 0003
IMul    EBx, X      ;EBx = FFFF FFFA (-6)
Ror     EBx, 16     ;Bx  = FFFF
Ror     EBx, 16     ;Bx  = FFFA
```

bx 0003 ⇨ bx FFFA ⇨ bx FFFF ⇨ bx FFFA

Piezīme: reizināšanas rezultātu var saglabāt *atmiņas šūnā*:

```
IMul    X, EBx
```

Kļūdas

```
IMul Bl, Cl     ;operandi nav vārdi (dubultvārdi)
Mul  Bx, Cx     ;bez zīmju reizināšana
```

Trīs operandu *reizināšana* (**IMul**).
Trešais operands vienmēr ir *tiešais* operands.
Rezultāts atrodas *pirmajā* operandā-reģistrā.

```
Mov   Bx, 2        ;Bx = 2
IMul  Bx, Bx, 3    ;Bx = 6
```

```
Mov   Bx, 2        ;Bx = 2
IMul  Cx, Bx, 3    ;Cx = 6
```

```
X     DW  2
...
IMul  Cx, X, 3      ;Cx = 6
```

```
Mov   EBx, 2        ;EBx = 2
IMul  ECx, EBx, 3 ;ECx = 6
```

Operandu paplašināšana līdz *četrkāršotām* vārdam (**Cwde**, **Cdq**).
Convert **W**ord to **D**ouble word **E**xtended
Convert **D**ouble word to **Q**uad word

```
Mov Ax, -3 ;Ax = FFFD (-3)
Cwde       ;EAx = FFFF FFFD (-3)
Cdq        ;EDx:EAx = FFFF FFFF:FFFF FFFD
```

Alternatīvais risinājums:

```
Mov EAx, -3
Cdq
```

Dalīšana:

```
Mov EBx, 3 ;EBx = 3
IDiv EBx   ;EAx = FFFF (-1), EDx = 0
```