Безусловни скок

Основна наредба која пребацује контролу на друго место у програму је JMP. Основна синтакса JMP инструкције : JMP label. Да би се дефинисао наслов у програму откуца се има и дода :. Наслов може бити било која комбинација карактера али не може почети бројем:

x1: x1: MOV AX, 1
MOV AX, 1

Пример01 : Употреба наредбе безусловног скока.

```
name 'skok1'
org   100h
mov    ax, 5            ; postavi ax na 5
mov    bx, 2            ; postavi bx na 2
jmp    racun            ; idi na racun
nazad:  jmp stop        ; idi na stop
racun:
add    ax, bx
jmp    nazad            ; idi na nazad
stop:
ret                     ; povratak u OS
```

Условни скок

Друга врста наредби гранања (скока) су наредбе условног скока које ће се извршити само ако су одређени услови испуњени. Оне се деле у три групе: наредбе које само тестирају један флег, наредбе које упоређују означене бројеве и наредбе које упоређују неозначене бројеве.

Наредбе условног скока које тестирају само један флег:

Инструкција	Опис	Услов	Дуална инст
JZ , JE	Jump if Zero (Equal).	ZF = 1	JNZ, JNE
JC	Jump if Carry	CF = 1	JNC
JS	Jump if Sign.	SF = 1	JNS
JO	Jump if Overflow.	OF = 1	JNO

Наредбе условног скока које упоређују означене бројеве:

Инструкција	Опис	Услов	Дуална инст
JE , JZ	Jump if Equal (=). Jump if Zero.	ZF = 1	JNE, JNZ
JNE , JNZ	Jump if Not Equal (<>). Jump if Not Zero.	ZF = 0	JE, JZ
JG , JNLE	Jump if Greater (>). Jump if Not Less or Equal (not <=).	ZF = 0 and SF = OF	JNG, JLE

Ако се упоређују 5 и 2: 5 -2 = 3, резултат није 0, ZF се ресетује (=0). Ако се упоређује 7 и 7: 7-7=0, резултат је 0, па је ZF сетован и помоћу наредбе JZ ће доћи до скока.

Пример02: Приказ условног скока

```
name 'skok2'
include "emu8086.inc"
org 100h
mov al, 25              ;postavlja AL na 25
mov bl, 10              ;postavlja BL na 10
cmp al, bl              ;poredi AL-BL
je jednak               ;ako je AL=BL (ZF=1) skok
putc 'n'                ;ako je dosao dovde, onda je AL<>BL
jmp stop                ;prikazi n, skok na stop
jednak:
   putc 'y'             ;ako je dosao dovde, onda je AL=BL
stop:
ret
```

Петље

```
mov cx, 8               ; postavljanje brojaca
print: mov ah, 2        ; print funkcija
     ...
loop print
```

Петље раде исто што и скокови. Користи се регистар CX за бројање корака (16-то битни регистар) и при сваком доласку на loop наредбу се вредност у CX аутоматски смањује за 1. Скок ће се извршити ако је вредност у CX различита од 0.