**Laborator ASC 24.10.2023**

A B A and B

0 0 0

1 0 0

0 1 0

1 1 1

and EAX, 0Fh ; are loc zeroizarea tuturor bitilor din eax cu exceptia ultimilor 4 corespunzatori valorii F

...1100101

...0001111

...0000101

A B A or B

0 0 0

0 1 1

1 0 1

1 1 1

functioneaza la fel ca si and

in consecinta pentru eax din exemplul anterior o sa avem 1 pe ultimii 4 biti

A B A xor B

0 0 0

1 0 1

0 1 1

1 1 0

xor intre un registru si el insusi zeroizeaza intreg registrul respectiv

!!! Aceasta este o modalitate de a pune 0 intr-un registru

TEST AL,01h ; se poate testa in acest mod daca numarul este par sau impar

O sa revenim in viitor la aceasta instructiune

not byte ptr [var] ; inverseaza fiecare bit din [var]

mov al, 00110011b

mov cl, 2

shl al,cl ; -> al=11001100, cf=0

;bitii stocati in destinatie se deplaseaza numar pozitii(modulo 32) spre stanga. Bitul (sau bitii) cel mai din dreapta se completeaza cu 0. Ultimul bit care iese in stanga se pastreaza in CF.

shr – face aceeasi chestie ca si shl doar ca la dreapta

ultimul bit care iese in dreapta se pastreaza in CF

efectul sar sal este identic cu cel al shr shl, doar ca completeaza cu bitul de semn si nu cu 0 implicit

rol - roteste numarul la stanga

ror – roteste numarul la dreapta

rol al,1 ; ultimul bit de la stanga vine la coada

• rcl - Biții stocați în destinație se rotesc număr poziții spre stânga. Odată un bit ieșit în stânga el se păstrează în CF. Valoarea anterioră din CF se adaugă automat în partea dreaptă a destinației.

rcr - Biții stocați în destinație se rotesc numar poziții (modulo 32) spre dreapta. Odată un bit ieșit în dreapta el se păstrează în CF. Valoarea anterioară din CF se adaugă automat în partea stângă a destinației.