Flaguri

**Scurtă definiție**

**CF** -> carry flag: valoarea transportului final intr-o adunare sau scădere

**OF** -> overflow flag: valoarea 1 daca doua numere cu acelasi semn, dau unul de semn opus

**ZF** -> zero flag: 1 daca rezultatul ultimei operatii effectuate este 0

**SF** -> sign flag: Ia valoarea bitului de semn din rezultatul ultimei operatii effectuate

**DF** -> direction flag: 0 – deplasarea in sir se face de la stanga la dreapta, daca e 1 invers

**PF** -> parity flag: Daca UOE are nr impar de biti atunci PF=1, altfel PF=0

**AF** -> auxialiry flag: valoarea transportului de la bitul 3 la bitul 4

**TF** -> trap flag: val 1 daca masina se opreste dupa fiecare instructiune -> nu poate fi schimbat

**IF** -> interruption flag: pentru sectiuni critice -> nu poate fi schimbat

**Instrucțiuni**

**CLC/STC/CMC**

Syntaxa: **CLC/STC**

Efect: pune in CF valoarea 0/1/!CF

Restricții:

Pe scurt: CF=0/1/!CF

**CLD/STD**

Syntaxa: **CLD/STD**

Efect: pune in DF valoarea 0/1

Restricții:

Pe scurt: CF=0/1

**CLI/STI**

Syntaxa: **CLI/STI**

Efect: pune in IF valoarea 0/1

Restricții:

Pe scurt: IF=0/1

**Speciale**

**PUSHF/POPF**

Syntaxa: **PUSHF/POPF**

Efect: pune/scoate toate flag-urile pe stiva

Restricții:

Pe scurt: EFLAGS=[ESP] / [ESP] = EFLAGS

**MODURI DE SETARE**

**CARRY FLAG**

CF se seteaza la 1 daca avem o adunare/sau o scadere care nu incape pe acelasi interval de reprezentare

Interval de reprezentare pe octet: [0,255]

* Functioneaza aceasta regula doar daca folosim numerele reprezentate fara semn (Daca avem un nr negativ de ex: -1, -1 ca numar pozitiv reprezentat pe octet va fi 256 (2^8) – 1),

Deci vom considera -1 = +255 (se mai numesc si valori complementare)

Regula generala: daca x<0 => x=256+x (Aici e de fapt o adunare cu un nr negativ)

!! Daca rezultatul e mai mic ca 0 sau mai mare decat 255 atunci CF=1, si in rest CF=0

Mare atentie daca avem adunare sau scadere. La adunare vom considera numerele negative ca fiind 256 – acel numar (daca e reprezentat fara semn), dar la scadere daca scadem un numar pozitiv din alt nr pozitiv atunci lasam numarul asa cum este

Exemple:

**ADUNARE**:

1)

mov al,100

mov ah,200

add al, ah

* La sfarsitul secventei CF=1 deoarece 100+200=300 > 255

2)

mov al,100

mov ah,-1

add al, ah

* La sfarsitul secventei CF=1 deoarece 100+(-1) = 100 + (256-1) = 100 + 255 = 355 > 255

**SCADERE**:

1)

mov al,100

mov ah,101

sub al, ah

* La sfarsitul secventei CF=1 deoarece 100-101=-1 < 0

2)

mov al,100

mov ah,-1

sub al, ah

* La sfarsitul secventei CF=1 deoarece 100-(-1)=100-(256-1)=100-255=-155 < 0

**Concluzie**: CF va fi setat daca rezultatul ultimei operatii efectuate nu este in intervalul [0,255]

->Toate numerele negative (din definitie) le transformam in 256-nr respectiv

->NU se aplica regula a+(-b)=a-b !!!!!!!!!!! (-b nr negativ din definitie, si il vom transforma ca si in [0,255]

->Ca regula principala, consideram ca nu exista numere negative, si pe toate le transformam in pozitive cu regula de mai sus

**OVERFLOW FLAG**

Interval de reprezentare pe octet: [-128,127]

Spre deosebire de ce faceam inainte, acum ca sa vedem usor cum se seteaza OF, transformam numerele pozitive din intervalul (127,255] in numere negative astfel: 128 = 128-256 = -128; 200 = 200-256=-56

Regula generala: daca x>127 => x=x-256 !!!!!

Daca rezultatul este mai mare ca 127 sau mai mic ca -128 atunci OF=1, si OF=0 altfel

Exemple:

**ADUNARE**:

1)

mov al,100

mov ah,100

add al, ah

* OF-1 deparece 100+100 = 200 > 127

2)

mov al,-100

mov ah,156

add al, ah

* OF-1 deparece -100+156 = -100 + (156-256) = -100 + -100 = -200 < -128

**SCADERE**:

1)

mov al,100

mov ah,-100

sub al, ah

* OF=1 deoarece 100-(-100) =100+100=200>127

2)

mov al,100

mov ah,156

sub al,ah

* OF=1 deoarece 100-(156) =100-(156-256)=100-(-100)=200>127

**Concluzii**:

Daca numarul este mai mare decat 127, el trebuie convertit cu regula precizata ca sa aflam practic daca OF=1

Daca avem o scadere cu un numar negativ din interval (-128<=x<0) atunci functioneaza regula a-(-b)=a+b !!

**OBSERVATII CF SI OF**

Pentru imul si mul CF=OF=1 daca rezultatul obtinut nu este in intervalul de reprezentare stabilit (numere pozitive pentru mul si numre negative si pozitive pentru imul)

Regulile de transformare raman aceleasi pentru numere din intervalul [0,255] (x<0 => x=256+x) sau [-128,127] (x>127 => x=x-256)

Exemple:

1)

mov al,-1

mov ah,2

mul al

* CF=OF=1 deoarece -1 \* 2 = (256-1)\*2 = 255\*2>255 (aici -1 nu e in intervalul de reprezentare, deci l-am transformat)

2)

mov al,-1

mov ah,2

imul al

* CF=OF=0 deoarece -1\*2=-2 (Aici -1 si -2 sunt in intervalul de reprezentare)

3)

mov al,156

mov ah,3

imul al

* CF=OF=1 deoarece 156\*3=(156-256)\*3=-100\*3=-300<-128 (aici 156 nu e in intervalul de reprezentare)

Aici am considerat toate exemplele reprezentate pe octet, dar tot aceleasi reguli se aplica si pe word si dword

La impartire daca rezultatul final nu incape pe intervalul de reprezentare NASM produce „run-time-error” (programul „crapa”) deci nu se face analiza lui CF si OF in acele cazuri

**ZERO FLAG**

ZF=1 daca rezultatul ultimei operatii efectuate

Exemple nasoale:

1)

mov al,15

mov ah,241

add al, ah

* ZF=1 deoarece 241+15=256 = 256-256 = 0 ( 256 nu incape pe un byte, deci se salveaza doar ultimii 8 biti din rezultat si CF=1)

2)

mov al,-1

mov ah,255

sub al, ah

* ZF=1 deoarece -1-255 = (256-1)-255 = 255 – 255 = 0 (aici am putea converti si invers)
* ZF=1 deoarece -1-255 = -1-(255-256)=-1-(-1)=-1+1=0

Observatii:

Cand facem adunarea sau scaderea pentru a analiza ZF, trebuie neapart sa avem numere din acelasi interval de reprezentare, fie cu numere pozitive fie cu numere negative (si pozitive) (rezultatul e tot timpul corect daca convertim la acelasi interval de reprezentare fie pe primul fie pe al doilea)

ZF nu se seteaza la inmultire indiferent de rezultat!

**SIGN FLAG**

Bitul de semn al ultimei operatii efectuate

**Pentru adunare si scadere:**

Convertim amandoua numerele la numere fara semn cu regula de mai sus (daca x<0 =>x=256+x)

Adunam normal numerele.

Daca rezultatul este in depasire (CF=1) atunci rez=rez-256

Daca noul rezultat este mai mic decat 128 atunci SF=0, altfel SF=1

Exemple:

1)

mov al,10

mov ah,-1

add al, ah

* SF=0 deoarece 10 + (-1) = 10+(256-1)=10+255=265 (depasire)=265-256 = 9 <128

2)

mov al,-1

mov ah,-1

add al, ah

* SF=1 deoarece (-1)+(-1)=255+255 (depasire)=255+255-256 = 254 >=128

**Pentru inmultire si impartire:**

La imul daca inmultim un nr pozitiv cu un numar negativ atunci SF=1 si SF=0 altfel

La mul -> val bitului de semn din al/ax/dx (din observatii ca nu am gasit in curs)

La div si idiv -> Nu am vazut schimbari IDK