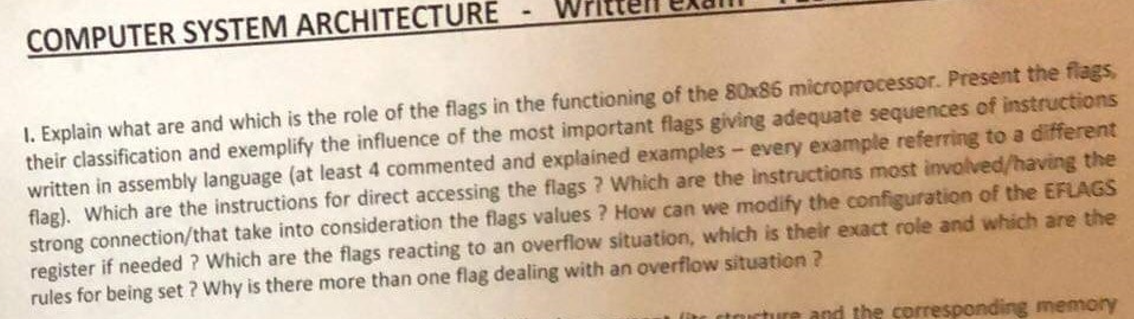
**Flags**



**Un flag** este **un indicator** **reprezentat pe un bit**. **O configurație a registrului de flaguri** indică **un rezumat sintetic a execuției fiecărei instrucțiuni**. Pentru x86, **registrul EFLAGS are 32 de biți din care folosim uzual doar 9**.

**Carry Flag (CF)** este **flagul de transport**. Are valoarea 1 în cazul în care în cadrul **UOE** (**ultimei operații efectuate**) **s-a efectuat transport în afara domeniului de reprezentare a rezultatului** și **valoarea 0 în caz contrar**. Acesta marchează depășirea în cazul interpretării **FĂRĂ SEMN**.  
Exemplu:  
 mov AL, 1000 0000b   
 mov BL, 1111 1111b   
 add AL, BL // rezultatul adunării celor 2 numere este **1** 0111 1111, deci **CF va primi valoarea 1**, rezultatul operației adunării dintre doi octet fiind mereu un octet, iar, deoarece rezultatul **UOE** nu încape pe un octet, în AL se vor păstra doar ultimii 8 biți (0111 1111b) și CF va fi setat la 1 (255 + 128 = 127).

**Parity Flag (PF),** are **valoarea 1** dacă **octetul cel mai puțin semnificativ al rezultatului UOE are un număr par de cifre 1** și **valoarea 0 în caz contrar**.  
Exemplu:  
 mov AX, 1000 0000 1000 0000b  
 mov BX, 0000 0000 0111 1111b  
add AX, BX // AX 🡨 1000 0000 1111 1111b **PF va primi valoarea 1** deoarece octetul cel mai puțin semnificativ (1111 1111b) are un număr par de cifre de 1 (8 mai exact).

**Auxiliary Flag (AF)**, indică valoarea transportului de la bitul 3 la bitul 4 al reprezentării UOE.  
Exemplu:  
 mov AL, 1000 1000b  
 mov BL, 0000 1001b  
 add AL, BL // AL 🡨 1001 0001 **AF = 1** deoarece avem transp de la bitul 3 la 4

**Zero Flag (ZF)**, primește valoarea 1 dacă rezultatul UOE este egal cu 0 și valoarea 0 la rezultatul diferit de 0.  
Exemplu:  
 mov AL, 1111 1111b  
 mov BL, 0000 0001b  
 add AL, B0000 0000 // **ZF = 1** deoarece rezultatul operației este 0, CF va fi și el setat la 1 deoarece avem transport.

**Trap Flag (TF)** este un flag de depanare, dacă are valoarea 1 atunci mașina se oprește după fiecare instrucțiune.

**Interrupt Flag (IF)** este un flag de întrerupere, programatorul nu poate să-l modifice direct la programarea de peste 16 biți.

**Direction Flag (DF)** este un flag pentru operarea șirurilor. Dacă are valoarea 0, atunci deplasarea în șir se face de la început spre sfârșit, iar dacă are valoarea 1, deplasarea în șir se face de la sfârșit spre început.

**Overflow Flag (OF)** primește **valoarea 1** dacă rezultatul ultimei instrucțiuni în interpretarea CU SEMN a operanzilor nu a încăput în spațiul rezervat operanzilor (intervalul de reprezentare admisibil), altfel primește valoarea 0.  
Exemplu:  
 mov AL, 0100 0001b // 65  
 mov BL, 0100 0000b // 64  
 add AL, BL // AL 🡨 1000 0000b (-128), OF primește valoarea 1 deoarece rezultatul nu a încăput în spațiul rezervat octeților în interpretarea cu semn.

**Flagurile se pot împărți în 2 categorii**:  
a) **cu efect anterior generat de către UOE** (CF, PF, AF, ZF, SF și OF)  
b) **cu efect ulterior setării lor de către programator pentru influențarea modului de operare al instrucțiunilor care urmează** (CF, TF, DF și IF)

Putem seta valoriile flag-urilor cu efect ulterior prin aceste instrucțiuni:  
**CLC** (CF 🡨 0) **STC** (CF 🡨 1) **CMC** (complementează pe CF)  
**CLD** (DF 🡨 0) **STD** (DF 🡨 1)  
**CLI** (IF 🡨 0) **STI** (IF 🡨 1) // inaccesibile peste 16 biți  
Pentru **TF** nu există deoarece acesta are un rol absolut special în depanare.

**Instructiuni cu strânsă legătură cu FLAG-urile**: **ADC** (la fel ca ADD doar că este adunat la rezultat și valoarea din Carry Flag), **SBB** (la fel ca SUB doar că la rezultat se scade din rezultat valoarea din Carry Flag), **RCL**, respectiv **RCR** (funcționează la fel ca ROL, respectiv ROR doar că se ia în considerare și valoarea din Carry Flag) **CMOVcc** (primul operand primește valoarea celui de-al doilea dacă conditia cc este îndeplinită), **SETcc** (setează valoarea operandului primit la 1 dacă condiția cc este adevărată și 0 în caz contrar) și **Jcc** (jump-urile se fac doar dacă conditia cc este îndeplinită). Pentru Jcc, condițiile sunt următoarele:  
**JA, JNA, JAE, JNAE, JB, JNB, JBE, JNBE (legate de CF si ZF, above (mai mare) si below (mai mic) in interpretarea fara semn), JG, JNG, JGE, JNGE, JL, JNL, JLE, JNLE (legate de SF, OF si ZF, greater (mai mare) si lower (mai mic) in interpretarea cu semn), JO, JNO (legate doar de OF), JS, JNS (legate doar de SF), JP, JNP (legate doar de PF), JE, JNE, JZ, JNZ (legate doar de ZF), JC, JNC (legate doar de CF).** La fel și pentru CMOVcc și SETcc.

Dacă dorim **să modificăm configurația registrului EFLAGS**, NASM ne pune la dispoziție 2 comenzi, și anume **PUSHF** și **POPF**. PUSHF depune registrul EFLAGS pe stivă iar POPF extrage vârful stivei și îl depune în EFLAGS. Dacă dorim o anumită configurație în registrul EFLAGS, putem să o punem într-un registru, să îl urcăm în stivă și să folosim POPF.

În cazul în care în cadrul UOE s-a efectuat **o depășire** (rezultatul nu a încăput în spațiul rezervat acestuia), avem **două flag-uri care reacționează la aceasta**, în funcție de cum sunt interpretate numerele de programator, și anume Carry Flag (în interpretarea cu semn) și Overflow Flag (în interpretarea cu semn) CF și OF exprimă eventualele depășiri doar pentru operațiile de adunare și de scădere.

Considerăm că operațiile se efectuează între 2 operanzi cu capacitatea de n biți.  
 Pentru adunare dintre cei 2 operanzi, CF este setat la 1 atunci când rezultatul are n + 1 biți și setat la 0 în caz contrar. OF este setat la 1 doar în una dintre situațiile:  
0…..+ 1.....+  
0….. 1.....  
=== ===  
1….. 0......

Pentru scăderea celor 2 operanzi, CF este setat doar dacă a fost nevoie ca la primul operand să fie ajustat un 1 pentru a se efectua scăderea (cu alte cuvinte dacă primul operand este mai mic decât al doilea). OF este setat la 1 doar în situațiile:  
1…- 0...-  
0… 1...  
== ==  
0… 1...

Avem nevoie simultan de ambele flag-uri deoarece în momentul efectuării unei operații de adunare sau scădere în baza 2 se efectuează de fapt simultan 2 operații în baza 10: una în interpretarea cu semn și cealaltă în interpretarea fără semn.