

**CF** (*Carry Flag*) este flagul de transport. Are valoarea 1 în cazul în care în cadrul ultimei operații efectuate (UOE) s-a efectuat transport în afara domeniului de reprezentare a rezultatului și valoarea 0 în caz contrar. De exemplu, pt

1001 0011 +	147 +		93h +	-109 +
<u>0111 0011</u>	<u>115</u>	rezulta un transport de cifra semnificativa si	<u>73h</u>	<u>115</u>
<b>1</b> 0000 0110	262	valoarea 1 este depusa automat in CF	106h	06
(fara semn)			(hexa)	(cu semn)

**Flagul CF semnalează depășirea în cazul interpretării FĂRĂ SEMN.**

**OF** (*Overflow Flag*) este flag pentru depășire **CU SEMN**. Dacă rezultatul ultimei instrucțiuni în interpretarea CU SEMN a operanzilor nu a încăput în spațiul rezervat operanzilor (intervalul de reprezentare admisibil), atunci acest flag va avea valoarea 1, altfel va avea valoarea 0. Pentru exemplul de mai sus, OF=0.

**Definiție.** O *depășire* este o condiție/situație matematică ce exprimă faptul că rezultatul unei operații nu a încăput în spațiul rezervat acestuia.

La nivelul procesorului și a limbajului de asamblare o *depășire* este o condiție/situație matematică ce exprimă faptul că rezultatul UOE nu a încăput în spațiul rezervat acestuia SAU acest rezultat nu aparține intervalului de reprezentare admisibil SAU că operația efectuată este un nonsens matematic în respectiva interpretare (cu semn sau fără semn) și nu poate fi astfel acceptată drept o operație matematică corectă.

## CF vs. OF. Conceptul de depășire.

1001 0011 +	147 +		93h +	-109 +
<u>1011 0011</u>	<u>179</u>	rezulta un transport de cifra semnificativa si	<u>B3h</u>	<u>- 77</u>
<b>1</b> 0100 0110	326	valoarea 1 este depusa in CF	146h	- 186 !!!!
(fara semn)			(hexa)	(cu semn)
CF= 1				OF= 1

- 326 și -186 sunt rezultatele corecte în baza 10 a celor două interpretări ale operației binare de mai sus

DAR, in limbaj de asamblare avem că ADD b+b → b, deci ceea ce vom obține ca și rezultate pe 1 byte va fi :

1001 0011 +	147 +		93h +	-109 +
<u>1011 0011</u>	<u>179</u>	rezulta un transport de cifra semnificativa si	<u>B3h</u>	<u>- 77</u>
<b>1</b> 0100 0110	+70	valoarea 1 este depusa in CF	146h	+ 70 !!!!
(fara semn)			(hexa)	(cu semn)
CF= 1				OF= 1

(operatii incorecte matematic in ambele interpretari !!!!)

Prin setarea ambelor flag-uri CF și OF la valoarea 1, procesorul ne « transmite mesajul » că ambele interpretări în baza 10 ale operației binare de adunare de mai sus sunt operații matematice incorecte !

---

0101 0011 +	83 +		53h +	83 +
<u>0111 0011</u>	<u>115</u>	NU rezulta un transport de cifra semnificativa	<u>73h</u>	<u>115</u>
1100 0110	198	deci CF=0	C6h	198 !!!!
(fara semn)			(hexa)	(cu semn)
CF=0				OF= 1

- 198 este rezultatul corect în baza 10 pentru ambele interpretări ale operației binare de adunare de mai sus

DAR, in limbaj de asamblare avem că ADD b+b → b, deci ceea ce vom obține ca și rezultate pe 1 octet va fi :

0101 0011 +	83 +		53h +	83 +
<u>0111 0011</u>	<u>115</u>	NU rezulta un transport de cifra semnificativa	<u>73h</u>	<u>115</u>
1100 0110	198	deci CF=0	C6h	-58 !!!!
(fara semn)			(hexa)	(cu semn)
CF=0				OF= 1

Setarea CF=0 exprimă faptul că interpretarea fără semn în baza 10 a adunării în baza 2 de mai sus este o operație corectă. OF va fi însă setat de către procesor la valoarea 1 acest lucru însemnând că interpretarea cu semn în baza 10 a adunării în baza 2 de mai sus este o operație incorectă !

OF va fi setat la valoarea 1 (*signed overflow*) dacă pentru operația de adunare ne aflăm în una din următoarele două situații (*regulile de depășire la adunare pentru interpretarea cu semn*). Sunt singurele doua situatii care provoaca depășire la adunare in interpretarea cu semn:

0.....+	sau	1.....+	(Semantic, cele doua situatii exprimă imposibilitatea acceptarii matematice a celor
0.....		1.....	2 operatii : nu putem aduna doua numere pozitive si sa obtinem unul negativ si nici
-----		-----	nu putem aduna doua numere negative si sa obtinem unul pozitiv).
1.....		0.....	

In cazul **scaderii**, avem de asemenea doua reguli de depășire în interpretarea cu semn, consecinta celor doua reguli de la depasirea in cazul adunarii :

1.....-	sau	0.....-	(Semantic, cele doua situatii exprimă imposibilitatea acceptarii matematice a celor 2
0.....		1.....	operatii: nu putem scadea un numar pozitiv dintr-un nr.negativ si sa obtinem unul pozitiv
-----		-----	și nici nu putem scadea dintr-un nr pozitiv unul negativ si sa obtinem unul negativ).
0.....		1.....	

<b>1</b> 0110 0010 -	98 -		62h -	98 -
<u>1100 1000</u>	<u>200</u>	Avem nevoie de împrumut de cifra semnificativa	<u>C8h</u>	<u>-56</u>
1001 1010	154	pt efectuarea scaderii si de aceea valoarea 1	9Ah	-102 !!!!
	(fara semn)	este depusa in CF	(hexa)	(cu semn)
	<b>CF=1</b>			<b>OF=1</b>

Nici una dintre cele 2 interpretari nu este consistentă în baza 10 : 98-200 (interpretarea fara semn a scaderii) ar fi trebuit să furnizeze -102 ca rezultat matematic corect (valoarea aceasta fiind disponibilă doar în interpretarea cu semn !!), valoarea 154 nefiind în mod evident un rezultat corect ! Interpretarea CU SEMN furnizeaza 98-(-56) = -102 (rezultat evident incorect !), deoarece  $98+56 = 154$  (acesta ar fi trebuit sa fie rezultatul corect, insa interpretarea 154 pt rezultat este valabila doar in interpretarea fara semn). Ca urmare, se constată că pentru a fi corecte matematic rezultatele finale ar fi trebuit să fie exact invers repartizate celor 2 interpretari, însă nu este așa, cele 2 operații matematice de mai sus (adică cele 2 interpretări asociate scaderii din baza 2) fiind ambele incorecte dpdv matematic. Ca urmare si ca reactie a mP 80x86 la aceasta situatie vom avea **CF=1** și respectiv **OF=1**.

Tehnic vorbind, microprocesorul seteaza OF=1 doar in una din cele 4 situatii prezentate mai sus (2 situatii pt adunare si respectiv 2 situatii pt scadere) plus inca o situatie pt inmultire care va fi explicata in cele ce urmeaza.

Operatia de inmultire NU furnizeaza depășire la nivelul arhitecturii 80x86, spatiul rezervat pt rezultat fiind suficient pentru ambele interpretari. Totusi, pt a nu ramane neutilizate flag-urile CF și OF în cazul înmulțirii s-a luat decizia ca în cazul în care în cadrul operației de înmulțire dimensiunea rezultatului se întâmplă să fie identică cu cea a operanzilor ( $b*b = b$ ,  $w*w = w$  sau  $d*d = d$ ) flag-urile CF și OF să fie setate ambele la valoarea 0 (« no multiplication overflow »,  $CF = OF = 0$ ), iar daca avem în mod real una dintre situațiile  $b*b = w$ ,  $w*w = d$ ,  $d*d = qword$ , atunci  $CF = OF = 1$  (« multiplication overflow »).

Cel mai grav efect al unei situații de depășire se manifestă în cazul împărțirii : în cazul acestei operații, dacă câtul obținut nu încapă în spațiul rezervat (spațiul rezervat de către asamblor fiind byte pentru împărțire word/byte, word pentru împărțire doubleword/word și respectiv doubleword pentru împărțire quadword/doubleword) atunci se va

semnala situație de « depășire la împărțire » cu efectul 'Run-time error' și cu emiterea din partea sistemului de operare a unuia dintre cele 3 mesaje echivalente : 'Divide overflow', 'Division by zero' sau 'Zero divide'.

In cazul unei împărțiri care se efectuează corect, adică fără a se semnala depășire, CF și OF sunt nedefinite. Dacă avem însă depășire, programul « crapă », execuția lui se încheie, deci practic nu mai are nici un sens pentru nimeni să se întrebe ce valoare au la acel moment flag-urile CF și OF...