Arhitectura Sistemelor de Calcul

Lect. Dr. Șotropa Diana
diana.sotropa@ubbcluj.ro

Facultatea de Matematică și Informatică
Universitatea Babeș-Bolyai

Analiza conceptului de depășire (overflow)

Flag-uri

- Următoarele patru instrucţiuni sunt instrucţiuni de transfer al indicatorilor:
  - Instrucţiunea `LAHF` (Load register AH from Flags) copiază indicatorii SF, ZF, AF, PF şi CF din registrul de flag-uri în biţii 7, 6, 4, 2 şi respectiv 0 ai registrului AH. Conţinutul biţilor 5,3 şi 1 este nedefinit. Indicatorii nu sunt afectaţi în urma acestei operaţii de transfer (în sensul că instrucţiunea LAHF nu este ea însăşi generatoare de efecte asupra unor flag-uri – ea doar transferă valorile flag-urilor şi atât).
  - Instrucţiunea `SAHF` (Store register AH into Flags) transferă biţii 7, 6, 4, 2 şi 0 ai registrului AH în indicatorii SF, ZF, AF, PF şi respectiv CF, înlocuind valorile anterioare ale acestor indicatori.
  - Instrucţiunea `PUSHF` transferă toţi indicatorii în vârful stivei (conţiunutul registrului Flags se transferă în vârful stivei). Indicatorii nu sunt afectaţi în urma acestei operaţii.
  - Instrucţiunea `POPF` extrage cuvântul din vârful stivei şi transferă din acesta indicatorii corespunzători în registrul de flag-uri.
- Limbajul de asamblare pune la dispoziţia programatorului nişte instrucţiuni de setare a valorii indicatorilor de condiţie, pentru ca programatorul să poată influenţa după dorinţă modul de acţiune a instrucţiunilor care exploatează flaguri.
- `CLC (CF=0), CMC (CF = ~CF), STC (CF=1), CLD (DF=0), STD (DF=1)`
- `CLI, STI` – actioneaza asupra flagului de intrerupere (IF). Functioneaza efectiv doar in programarea sub 16 biti, aici la programarea sub 32 biti SO interzicand accesul la flag-ul de intreruperi.

Analiza conceptului de depășire (overflow)

- **CF** *(Carry Flag)* – flag pentru depășire FĂRĂ SEMN
  - are valoarea 1 în cazul în care în cadrul ultimei operatii efectuate (UOE) s-a efectuat transport în afara domeniului de reprezentare a rezultatului si valoarea 0 in caz contrar

$1001\ 0011b +$	$147 +$	$93h +$	$-109 +$
$0111\ 0011b$	115	$73h$	115
$\mathbf{1}\ 0000\ 0110b$	262	$106h$	06
	(fără semn)	**(hexa)**	**(cu semn)**
	CF=1		**OF=0**

- **OF** *(Overflow Flag)* - flag pentru depăşire CU SEMN
  - dacă rezultatul ultimei instrucţiuni în interpretarea CU SEMN a operanzilor nu a încăput în spaţiul rezervat operanzilor (intervalul de reprezentare admisibil), atunci acest flag va avea valoarea 1, altfel va avea valoarea 0.

Analiza conceptului de depășire (overflow)

- Definiţie (generală, comprimată şi incompletă).
  - O *depăşire* este o condiţie/situaţie matematică ce exprimă faptul că rezultatul unei operaţii nu a încăput în spaţiul rezervat acestuia.
    (nici -147 nu "încape" în intervalul [-128..+127] şi nici pe un byte însă e mai dificil de intuit că definiţia cuprinde şi se referă şi la acest caz...)
- Definiţie mai exactă şi completă:
  - La nivelul procesorului şi a limbajului de asamblare o *depăşire* este o condiţie/situaţie matematică ce exprimă faptul că:
    - rezultatul UOE nu a încăput în spaţiul rezervat acestuia
      SAU
      că acest rezultat nu aparţine intervalului de reprezentare admisibil pe acea dimensiune de reprezentare
      SAU
      că operaţia efectuată este un nonsens matematic în respectiva interpretare (cu semn sau fără semn) şi nu poate fi astfel acceptată drept o operaţie matematică corectă.

Analiza conceptului de depășire (overflow)

$1001\ 0011b +$	$147 +$	$93h +$	$-109 +$
$1011\ 0011b$	179	$B3h$	$-\ 77$
$\mathbf{1}\ 0100\ 0110b$	326	$146h$	-186
(reprezentare binară)	**(interpretare fără semn)**	**(reprezentare hexa)**	**(interpretare cu semn)**

Care este nr MINIM de biţi pe care se poate reprezenta 326 și -186?

$$326 \in [0,511] \text{ și } -186 \in [-256, 255]$$

Așadar numărul minim de biți pe care se pot reprezenta numerele este 9.

Ca urmare, TOATE operaţiile de mai sus se desfăşoară CORECT MATEMATIC pe 9 biţi şi operanzii şi rezultatele finale INCAP în spaţiul rezervat, DACA operatiile se desfăşoară pe 9 biţi !!

Analiza conceptului de depășire (overflow)

$1001\ 0011b +$	$147 +$	$93h +$	$-109 +$
$1011\ 0011b$	179	$B3h$	$-\ 77$
$\mathbf{1}\ 0100\ 0110b$	326	$\mathbf{1}46h$	-186
(reprezentare binară)	**(interpretare fără semn)**	**(reprezentare hexa)**	**(interpretare cu semn)**

Insă din păcate, adunarea de mai sus se desfăşoară la nivel de procesor pe 8 biţi (deoarece în limbaj de asamblare avem că ADD b+b = b) şi ca urmare dpdv MATEMATIC, aceasta NU se va desfăşura corect pe 8 biţi, nici 326 şi nici -186 neîncăpând pe 1 octet !!

Acest lucru este semnalat SIMULTAN de către flag-urile CF (pt interpretarea fără semn) şi respectiv OF (pt interpretarea CU semn), ambele flag-uri fiind setate la valoarea 1.

Prin setarea flag-urilor CF şi OF la valoarea 1, procesorul ne transmite mesajul că ambele interpretări în baza 10 ale operaţiei binare de adunare pe 1 byte sunt operaţii matematice incorecte !

Analiza conceptului de depășire (overflow)

$0101\ 0011b +$	$83 +$	$53h +$	$83 +$
$0111\ 0011b$	115	$73h$	115
$1100\ 0110b$	198	$C6h$	198
(reprezentare binară)	**(interpretare fără semn)**	**(reprezentare hexa)**	**(interpretare cu semn)**

198 este rezultatul corect în baza 10 pentru ambele interpretări ale OPERANZILOR binari din adunarea de mai sus, INSA trebuie să vedem acum dacă rezultatul încape pe 8 biţi (DA – încape, de aceea vom avea CF=0) şi respectiv dacă rezultatul operaţiei binare în interpretarea cu semn este unul consistent cu corectitudinea operaţiei matematice efectuate (NU este, deoarece 1100 0110b = -58 NU este un număr pozitiv în interpretarea CU semn !), deci OF=1

Analiza conceptului de depășire (overflow)

- OF va fi setat la valoarea 1 (*signed overflow*) dacă pentru operaţia de adunare ne aflăm în una din următoarele două situaţii (regulile de depăşire la adunare pentru interpretarea cu semn). Sunt singurele două situaţii care provoacă depăşire la **adunare** în interpretarea cu semn:

0 ... ... +	1 ... ... +
0	1 ... ...
1	0 ... ...

Analiza conceptului de depășire (overflow)

- In cazul scăderii, avem de asemenea două reguli de depăşire în interpretarea cu semn, consecinţe a celor două reguli de la depăşirea în cazul adunării:

1 … … –	0 … … –
0 … …	1 … …
0 … …	1 … …

Analiza conceptului de depășire (overflow)

$\mathbf{1}\ 0100\ 0110b\ -$	$98\ -$	$62h\ -$	$98\ -$
$1100\ 1000b$	200	$C8h$	$-\ 56$
$1001\ 1010b$	-102	$9Ah$	154
(reprezentare binară)	**(interpretare fără semn a operanzilor)**	**(reprezentare hexa)**	**(interpretare cu semn a operanzilor)**

- REZULTATELE MATEMATICE CORECTE sunt precizate mai sus- ignorând INTERPRETAREA în vreun fel a configuraţiei binare 1001 1010b
- Insă, dacă luăm în considerare şi interpretările REZULTATULUI obţinut în program, avem din păcate :

$\mathbf{1}\ 0100\ 0110b\ -$	$98\ -$	$62h\ -$	$98\ -$
$1100\ 1000b$	200	$C8h$	$-\ 56$
$1001\ 1010b$	154	$9Ah$	-102
(reprezentare binară)	**(interpretare fără semn) CF=1**	**(reprezentare hexa)**	**(interpretare cu semn) OF=1**

- Ambele interpretări ale rezultatului obţinut în baza 2 SUNT INCORECTE MATEMATIC, deci CF şi OF vor fi ambele setate la valoarea 1.

Analiza conceptului de depășire (overflow)

- **Operatia de înmulţire NU furnizează depăşire la nivelul arhitecturii 80x86**, spaţiul rezervat pt rezultat fiind suficient pentru ambele interpretări
- Pentru a nu rămâne neutilizate flag-urile CF şi OF în cazul înmulţirii s-a luat decizia ca:
  - în cazul în care în cadrul operaţiei de înmulţire dimensiunea rezultatului se întâmplă să fie identică cu cea a operanzilor
    (b*b = b, w*w = w sau d*d = d)
    flag-urile CF şi OF să fie setate ambele la valoarea 0
    (« no multiplication overflow », CF = OF = 0),
  - iar dacă avem în mod real una dintre situaţiile
    b*b = w, w*w = d, d*d = qword,
    atunci
    CF = OF = 1 (« multiplication overflow »).

Analiza conceptului de depășire (overflow)

- Cel mai grav efect al unei situaţii de depăşire se manifestă în cazul împărţirii: în cazul acestei operaţii, dacă câtul obţinut nu încape în spaţiul rezervat
  (spaţiul rezervat de către asamblor fiind byte pentru împărţire word/byte, word pentru împărţire doubleword/word şi respectiv doubleword pentru împărţire quadword/doubleword)

➡ se va semnala situaţie de **« depăşire la împărţire »** cu efectul **'Run-time error'** şi 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 împărţiri care se efectuează corect, adică fără a se semnala depăşire, CF şi OF sunt nedefinite. Dacă avem însă depăşire, programul « crapă », execuţia 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 şi OF…

Analiza conceptului de depășire (overflow)

w / b -> b

1002 / 3 = 334 = situație de depășire (overflow) în cazul împărțirii => **Division by zero**

Oare DE CE se emite un astfel de mesaj care sugerează împărţirea la zero cu toate că aici am împărţit la 3 ?

Analiza conceptului de depășire (overflow)

De ce am nevoie SIMULTAN de CF şi OF in EFLAGS ?
Nu ajunge un singur flag pt a îmi arăta PE RAND daca am sau nu depăşire fie în interpretarea cu semn fie în cea fără semn ?

Analiza conceptului de depășire (overflow)

DE CE AM NEVOIE DE IMUL si IDIV ?

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
UNIVERSITATEA BABEȘ-BOLYAI

Str. Mihail Kogălniceanu nr. 1
Cluj-Napoca, Cluj, România

www.cs.ubbcluj.ro