Arhitectura sistemelor de calcul

Laborator 2 - Instrucțiuni aritmetice

ADD

```
add <regd>, <regs> ; <regd> <- <regd> + <regs>
add <reg>, <mem>
add <mem>, <reg>
add <reg>, <con>
add <mem>, <con>
```

Restricții: Cei doi operanzi ai adunării trebuie să aibă același tip (ambii octeți, ambii cuvinte, ambii dublucuvinte). În timp ce ambii operanzi pot fi registrii, cel mult un operand poate fi o locație de memorie.

Exemplu:

```
add EDX, EBX
add AX, [var]
add [var], AX
add EAX, 123456h
add BYTE [var], 10
```

SUB

sub <regd>, <regs> ; <regd> <- <regd> - <regs>

sub <reg>, <mem>

sub <mem>, <reg>

sub <reg>, <con>

sub <mem>, <con>

Restricții: Cei doi operanzi ai adunării trebuie să aibă același tip (ambii octeți, ambii cuvinte, ambii dublucuvinte)
În timp ce ambii operanzi pot fi registrii, cel mult un operand poate fi o locație de memorie.

Exemplu:

```
sub EDX, EBX
sub AX, [var]
sub [var], AX
sub EAX, 12345h
sub byte [var], 10
```

MUL

mul <op8> ; AX ← AL * <op8>

mul <op16> ; DX : AX ← AX * <op16>

mul <op32> ; EDX : EAX ← EAX * <op32>

Restricții: Lungimea operației de înmulțire se păstrează pe o lungime dublă față de lungimea operanzilor.

Instrucțiunea MUL efectuează operația de înmulțire pentru întregi fără semn.

Se impune ca primul operand și rezultatul să se păstreze în registri.

Deși operația este binară, se specifică un singur operand deoarece celălalt este întotdeauna fixat, la fel ca și locația rezultatului.

Operandul explicit poate fi un registru sau o variabilă, dar nu poate fi o valoare imediată (constantă). Exemplu:

mul DH ; AX ← AL * DH

mul DX ; DX : AX ← AX * DX

mul EBX ; EDX : EAX ← EAX * EBX

mul BYTE [mem8] ; AX ← AL * BYTE [mem8]

mul WORD [mem16] , DX : AX ← AX * WORD [mem16]

DIV

div <reg 8> ; AL ← AX / <reg 8> , AH ← AX % <reg 8>

div <reg 16> ; AX ← DX : AX / <reg 16> , DX ← DX : AX % <reg 16>

div <reg 32> ; EAX ← EDX : EAX / <reg 32> , EDX ← EDX : EAX % <reg 32>

div <mem 8> ; AL ← AX / <mem 8> , AH ← AX % <mem 8>

div <mem 16> ; AX ← DX : AX / <mem 16> , DX ← DX : AX % <mem 16>

div <mem 32> ; EAX ← EDX : EAX / <mem 32> , EDX ← EDX : EAX % <mem 32>

Restricții Instrucțiunea DIV efectuează operația de împărțire pentru întregi fără semn.

Se impune ca primul operand și rezultatul să se păstreze în registri.

Primul operand nu se specifică și are o lungime dublă față de al doilea operand

Operandul explicit poate fi un registru sau o variabilă dar nu poate fi o valoare imediată (constantă).

Prin împărțirea unui număr mare la un număr mic, există posibilitatea ca rezultatul să depășească capacitatea de reprezentare. În acest caz, se va declanșa aceeași eroare ca și la împărțirea cu 0.

Exemplu:

div CL ; AL ← AX / CL , AH ← AX % CL

div SI ; AX ← DX : AX / SI , DX ← DX : AX % SI

div EBX ; EAX ← EDX : EAX / EBX , EDX ← EDX : EAX % EBX

INC

inc <reg> ; <reg> ← <reg> + 1

inc <mem>

Exemplu:

inc DWORD [var] ; DWORD[var] ← DWORD[var] + 1

inc EBX

inc DL

DEC

dec <reg> ; <reg> ← <reg> - 1

dec <mem>

Exemplu:

dec EAX

dec BYTE [mem]

NEG

neg <reg> ; <reg> ← 0 - <reg>

neg <mem>

Exemplu

neg EAX ; EAX ← 0 - EAX

Declararea variabilelor / constantelor

- cu valoare inițială

a DB 0A2h ; se declară variabila a de tip BYTE și primește val. 0A2h

b DW ab ; se declară variabila b de tip WORD și primește val. 'ab'

c DD 12345678h ; se declară variabila c de tip DOUBLE WORD și primește valoarea 12345678h

d DQ 1122334455667788h ; se declară variabila c de tip QUAD WORD și primește valoarea 1122334455667788h

- fără valoare inițială

a RESB 1 ; se rezervă 1 octet

b RESB 64 ; se rezervă 64 octeți

c RESW 1 ; se rezervă 1 word

- definirea constantelor

ZECE EQU 10; se definește constanta care are valoarea 10