## Instrucțiuni cu/fără semn. Instrucțiuni aritmetice. Conversii cu/fără semn.

3 clase

* Instructiuni care nu tin cont de reprezentarea numerelor: MOV, ADD, SUB
* Instructiuni care tin cont de reprezentarea numerelor si interpreteaza numerele ca fiind fara semn: MUL, DIV
* Instructiuni care tin cont de reprezentarea numerelor si interpreteaza numerele ca fiind cu semn: IMUL, IDIV, CBW, CWD, CWDE, CDQ

**MUL op / IMUL op**

op \* operand2 = rezultatul

1) op este de tip byte

operand2 = AL

rezultat = AX

2) op este de tip word

operand2 = AX

rezultat = DX:AX (DX = partea high din rezultat si AX = partea low din rezultat)

3) op este de tip doubleword

operand2 = EAX

rezultat = EDX: EAX (EDX = partea high din rezultat si EAX = partea low din rezultat)

Exemple:

MOV AX, -1 ; AX = -1 = 11111111 11111111b = 0FFFFh

; 1 = 00000000 00000001b

; complementul = 11111111 11111111b

MUL AL ; AX = AL \* AL = 255 \* 255

; AL = 0FFh = 11111111b = 255

MOV AX, -1

IMUL AL ; AX = AL \* AL = (-1) \* (-1) = 1

**DIV op / IDIV op**

deimpartit / op = cat r rest

1) op este de tip byte

deimpartit = AX

cat = AL

rest = AH

2) op este de tip word

deimpartit = DX:AX

cat = AX

rest = DX

3) op este de tip doubleword

deimpartit = EDX:EAX

cat = EAX

rest = EDX

Exemple:

MOV AX, 0080h

DIV AL

; AX = 0080h = 128

; AL = 80h = 128

; dupa impartire AL= 128 / 128 = 1 si AH = 0

MOV AX, 0080h

IDIV AL

; AX = 0080h = 128

; AL = 80h = -128

; 10000000b => complement: 10000000b = 80h

; dupa impartire AL = 128 / (-128) = -1 si AH = 0

Exercitiul 1.

X = (a+b)\*c / d, interpretarea cu semn si a,b,c,d – octeti

|  |  |
| --- | --- |
| Data segment | Code segment |
| A db 2  B db 2  C db 2  D db 2  X db 0 | ; al -> a + b  MOV al, [a]  ADD al, [b]  ; ax -> (a+b) \* c  MOV ah, [c]  IMUL ah  ; al – cat, ah - rest  IDIV byte [d]  MOV [x], al |

**Conversii fara semn**

* Punem 0-uri nesemnificative In partea high
* 8 biti => 16 biti

AL => AX

MOV AH,0

BL => BX

MOV BH,0

* 16 biti => 32 biti

AX => DX:AX

MOV DX,0

AX => EAX

MOV BX,AX

MOV EAX,0

MOV AX,BX

PUSH word 0

PUSH AX

POP EAX

* 32 biti => 64 biti

EAX => EDX:EAX

MOV EDX,0

DX:AX => EDX:EAX

PUSH DX

PUSH AX

POP EAX

MOV EDX,0

**Conversii cu semn**

* Punem 0-uri sau 1-uri in partea high in functie de bitul de semn
* 8 biti => 16 biti

**CBW (convert byte to word)**

AL => AX

* 16 biti => 32 biti

**CWD (convert word to doubleword)**

AX => DX:AX

**CWDE (convert word to doubleword extended)**

AX => EAX

* 32 biti => 64 biti

**CDQ (convert doubleword to quadword)**

EAX => EDX:EAX

Exercitiul 2:

x = a\*b/d - c, unde a,c,d – bytes, b – word si interpretarea fara semn

|  |  |
| --- | --- |
| Data segment | Code segment |
| a db 2  b dw 10  d db 5  c db 8  x resw 1 | MOV AL, [a]  ;convertim pe a de la byte la word  Mov ah, 0 ; ax<- a  Mul word[b] ; dx:ax= a\*b  Mov bl , [d]  Mov bh,0 ; bx<-d  Div bx ; ax= a\*b/d ;  Mov cl, [ c]  Mov ch, 0 ; CX<-CX  Sub ax, cx ax=a\*b/d-c  Mov [x], ax |

x = a\*b/d - c, unde a,c,d – bytes, b – word si interpretarea cu semn

|  |  |
| --- | --- |
| Data segment | Code segment |
| a db 10  b dw –7  c db 15  d db 11  x resw 1 | MOV AL, [a]  CBW  MOV DX, AX  MOV AX,[b]  IMUL DX ; DX:AX= AX\*DX= a\*b  push DX  Push AX  MOV AL,[d]  CBW  MOV BX,AX  pop AX  pop DX  IDIV BX ; AX= DX:AX / BX, DX=DX:AX % BX  MOV BX,AX  MOV AL,[c]  CBW  SUB BX,AX ; BX=BX-c  MOV [x], BX |

Exercitiul 3

X = (a-b\*c)/d, a,b,c octeti si d cuvant in interpretarea fara semn

|  |  |
| --- | --- |
| Data segment | Code segment |
| A db 2  B db 2  C db 2  D dw 0  X resw 1 | ; ax <- b\*c  MOV al, [b]  MUL byte [c]  ; convert a to word  MOV dl, [a]  MOV dh, 0  ; dx:ax <- a – b\*c  SUB ax, dx  NEG ax  MOV dx, 0  MOV cx, [d]  DIV cx; cat – ax si dx – rest  MOV [x], ax |

X = (a-b\*c)/d, a,b,c octeti si d cuvant in interpretarea cu semn

|  |  |
| --- | --- |
| Data segment | Code segment |
| A db 2  B db –5  C db –2  D dw 10  X resw 1 | MOV AL,[b];AL=b  IMUL byte[c];AX=b\*c  MOV BX,AX;BX=b\*c  MOV AL,[a]  Cbw  SUB AX,BX;AX=a-b\*c  Cwd  IDIV word [d];AX=(a-b\*c)/d  MOV [x],AX |