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

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

**MUL op / IMUL op**

**op \* factor = rezultat**

1) op este de tip byte

factor = AL

rezultat = AX

2) op este de tip word

factor = AX

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

3) op este de tip double word

factor = EAX

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

Exemple:

MOV AL, -1 ; AL = 0FFh

MUL AL ; AX = AL \* AL = 0FFh \* 0FFh = 255 \* 255 = ...

MOV AL, -1 ; AL = 0FFh

IMUL AL ; AX = AL \* AL = 0FFh \* 0FFh = (-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 double word

deimpartit = EDX : EAX

cat = EAX

rest = EDX

Exemple:

MOV AX, 0080h ; AX = 0080h = 128; AL = 80h = 128

DIV AL ; AX / AL = 128 / 128 = 1 r 0 => AL = 1 si AH = 0

MOV AX, 0080h ; AX = 0080h = 128; AL = 80h = - 128

; 80h = 1000 0000b

; complementul: 1000 0000b = 80h = 128

IDIV AL ; AX / AL = 128 / (-128) = -1 r 0 => AL = -1 si AH = 0

Exercitiu:

X = (a+b) \*c / d, unde a,b,c,d – bytes + interpretare fara semn

|  |  |
| --- | --- |
| data segment | code segment |
| A DB 3  B DB 4  C DB 3  D DB 2  X DB 0 ; x resb 1 | Mov AL,[A]  Add AL,[B] ;AL = A + B  Mul Byte[C] ; AX = (A+B)\*C  Div Byte [D] ; AL = AX/D // AH = rest  Mov [X],AL |

X = (a+b) \*c / d, unde a,b,c,d – bytes + interpretare cu semn

|  |  |
| --- | --- |
| data segment | code segment |
| A DB –3  B DB 2  C DB –2  D DB 4  X RESB 1 | MOV AL, [A]  ADD AL, [B] ; AL = A+B  IMUL BYTE[C] ;AX = (A+B)\*C  IDIV BYTE[D] ; AL = AX/D , AH = AX%D  MOV [X],AL |

**Conversii fara semn** – se pun 0-uri nesemnificative in partea high

8 biti => 16 biti

AL => AX prin mov AH,0

BL => BX prin mov BH, 0

16 biti => 32 biti

AX => DX:AX prin mov DX,0

AX => EAX prin

MOV BX,AX

MOV EAX,0

MOV AX,BX

PUSH WORD 0

PUSH AX

POP EAX

32 biti => 64 biti

DX:AX => EDX:EAX

PUSH DX

PUSH AX

POP EAX

MOV EDX,0

EAX => EDX:EAX prin mov EDX,0

**Conversii cu semn** – pun 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

BL => AL => CBW => AX => BX

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 quad)

EAX => EDX:EAX

Exercitiu:

X= (a-b\*c)/d, unde a,b,c,d sunt octeti + interpretarea fara semn

|  |  |
| --- | --- |
| data segment | code segment |
| A db 2  B db 3  C db 4  D db 2  X resb 1 | Mov al, [b]  Mul byte [c], ax=b\*c  Mov bl,[a]  Mov bh,0, bx=a  Sub bx,ax , bx=a-b\*c  Mov ax,bx , ax=a-b\*c  Div byte [d] , al=cat ah rest  Mov byte [x] , al |

X= (a-b\*c)/d, unde a,b,c,d sunt octeti + interpretarea cu semn

|  |  |
| --- | --- |
| data segment | code segment |
| A DB –24  B DB –2  C DB 10  D DB – 2  X RESB 1 | MOV AL, [B]  IMUL BYTE [C] ; AX = AL \* C  MOV BX, AX  MOV AL, [A]  CBW ; AL => AX = A  SUB AX, BX  IDIV BYTE [D] ; AL  MOV BYTE [X], AL |

X = a\*b/d - c unde a,c,d sunt octeti, b – cuvant + interpretarea fara semn

|  |  |
| --- | --- |
| data segment | code segment |
| A db 10  C db 5  D db 6  B dw 20  X resw 1 | Mov al, [A]; al=A  Mov ah,0; ax=A  Mul word [B]; dx:ax= A\*B  Mov bl,[D]; bl=d  Mov bh,0;  Div bx; ax=a\*b/d [\\](file:///\\) dx=a\*b%d  Mov bl,[C]; bl=0  Mov bh,0;  Sub ax,bx; ax=a\*b/d-c  Mov [X],ax; X=a\*b/d-c |

X = a\*b/d - c unde a,c,d sunt octeti, b – cuvant + interpretarea cu semn

|  |  |
| --- | --- |
| data segment | code segment |
| A db -15  B dw 20  C db 14  D db –13  X resw 1 | Mov AL, [A] ; AL = A  CBW ; AX = A  Imul word [B] ; dx:ax = ax \* b = a\*b  Push dx  Push ax  Pop ebx  Mov AL, [D]  CBW ; AX = D  Mov CX, AX  Push EBX  Pop AX  Pop DX  Idiv CX ; ax = a\*b/d  Mov bx, ax  Mov AL, [C]  CBW  Sub BX, AX  Mov [X], BX |

X= ((a+b-d) / f + g\*h )/(-i)

a,d,f – word

b,g,h – octeti

i – double word

Interpretarea cu semn

Rezolvare

a+b-d => word => conversie de la word la double word

B => conversie de la byte la word

(a+b-d) / f => word

g \* h => word

(a+b-d) / f + g\*h => word => double word => quad word

((a+b-d) / f + g\*h )/(-i)=> double word

**X= ((a+b-d) / f + g\*h )/(-i)**

|  |  |
| --- | --- |
| data segment | code segment |
| A DW 13  D DW 2  F DW –4  B DB 12  G DB -23  H DB 3  I DD 4  X RESD 1 | MOV AL, [B]  CBW ; AX = B  ADD AX, [A] ; AX = A+B  SUB AX, [D]; AX= A+B-D  CWD  IDIV WORD[F] ; AX=catul DX restul  MOV BX, AX  MOV AL, [G]  IMUL BYTE[H] ; AX = G\*H  ADD AX, BX  CWDE  CDQ  MOV EBX, [I]  NEG EBX  IDIV EBX;  MOV [X], EBX |