[(a\*b\*3 – d)/a]/e-c, a byte, b byte, d double word, -c quadword (cu semn), e word , REZ dq 0, REZ resq 1

;a\*b

Mov al, [a]

IMul byte[b]; ax=a\*b

Mov bx, 3

Imul bx; dx:ax= ax\*bx

; 16:16 - 32

Push dx

Push ax

Pop ebx ; eax= a\*b\*3

Sub ebx, [d] ; ebx=ebx-d = a\*b\*3 – d

;32 / 8

Mov al, [a]

Cbw ; ax=a

Mov cx, ax

; ebx->dx:ax

Push ebx

Pop ax

Pop dx ; dx:ax = (a\*b\*3 – d)

Idiv cx ; dx:Ax/cx = ax – cat si in DX rest, ax = [(a\*b\*3 – d)/a]

;word/word - > double/word, double=dx:ax

Cwd ; dx:ax = [(a\*b\*3 – d)/a]

Idiv word[e] ; ax – cat si in dx rest, ax = [(a\*b\*3 – d)/a]/e

; word – quadword

Cwde ; eax = [(a\*b\*3 – d)/a]/e

Cdq ; edx:eax= [(a\*b\*3 – d)/a]/e

;c – quad in ecx:ebx

Mov ebx, dword[c+0]

Mov ecx, dword[c+4]

Clc ; clear CF, Cf=0

Sub eax, ebx

Sbb edx, ecx ; edx:eax – rez expresie

; sub eax, dword [c+0]

;Sbb edx, dword [c+4]

; save rez in variab REZ

Mov dword[rez+0], eax

Mov dword[rez+4], edx

; dx:ax -> quadword

Push dx

Push ax

Pop eax

Cdq

; cu variab

Mov word[aux+2], dx

Mov word[aux+0], ax

; aux <- dx:ax

Mov eax, [aux]

Cdq

a dq 1122334455667788h

B dq 1122222222222222h

Sub [a], [b]

1. idiv si div

2. adc si add

3. -21 / 4

4. 300 / 1

300/1

Mov ax, 300

Mov dx, 0

Mov bx, 1

Div bx

5. 5\*d , d word

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| A4 | A3 | A2 | 1 | 1 | Not a2 | Not a1 | Not a0 |
| B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |

Mov al, [a]

Not al ; notA7 .... not A2 not A1 not A0

AND AL, 00000111b ; 00000 not a2 not a1 not a0

MOV BL, AL ; SAVE REZ INTERMEDIAR

OR BL, 00011000b ; BL = 00011 not a2 not a1 not a0

MOV AL, [A]

And al, 00011100b; al 000 a4 a3 a2 00

SHL al, 3 ; al = a4 a3 a2 00000

Or bl, al ; a4 a3 a2 11 not a2 not a1 not a0

Mov [b], bl