Ministerul Educației al Republicii Moldova

Universitatea Tehnică a Moldovei

Catedra Automatica și Tehnologii Informaționale

RAPORT

Lucrare de laborator Nr.2
la Arhitectura Calculatoarelor
Tema:Bazele limbajului Assembler

A efectuat:

A verificat:

Chisinau 2016

Scopul Lucrarii
Se prezinta problemele principale legate de conversii de date, reprezentarea datelor întregi, reprezentarea întregilor in format BCD, reprezentarea caracterelor si a șirurilor de caractere, reprezentarea valorilor reale, elemente de memorie, tipuri de date utilizate si modurile de adresare a operanzilor.

Desfasurarea lucrarii de laborator
Se cere obtinerea fisierului executabil pentru urmatoarea portiune de cod si rularea apoi pas cu pas.

Varianta Nr.4
*z=(a*3+b*b*5)/(a*a+2*a*b)-a-b*

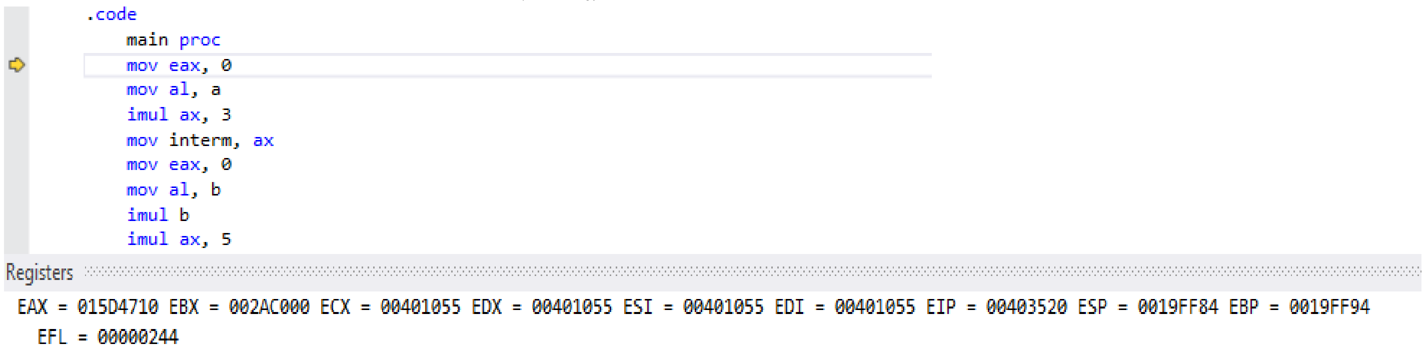
Cod Sursa

```
INCLUDE Irvine32.inc
.data
        a db 1
        b db 2
        interm dw ?
        interm1 dw ?
        rez dw ?
.code
        main proc
        mov eax, 0
        mov al, a
        imul ax, 3
        mov interm, ax
        mov eax, 0
        mov al, b
        imul b
        imul ax, 5
        add interm, ax
        mov eax, 0
        mov al, a
        imul a
        mov interm1, ax
        mov ebx, 0
        mov al, a
        imul b
        imul ax, 2
        add interm1, ax
        xchg interm1, ax
        xchg ax, interm
        cwd
        div interm
        sub al, a
        sub al, b
        call WriteInt
        exit
        main ENDP
        END main
```

Listing Cod

```
INCLUDE Irvine32.inc
 00000000                      .data
 00000000 01                           a db 1
 00000001 02                           b db 2
 00000002 0000                                 interm dw ?
 00000004 0000                                 interm1 dw ?
 00000006 0000                                 rez dw ?
 00000000                      .code
 00000000                              main proc
 00000000  B8 00000000                         mov eax, 0
 00000005  A0 00000000 R               mov al, a
 0000000A  66| 6B C0 03                        imul ax, 3
 0000000E  66| A3                      mov interm, ax
           00000002 R
 00000014  B8 00000000                         mov eax, 0
 00000019  A0 00000001 R               mov al, b
 0000001E  F6 2D 00000001 R                    imul b
 00000024  66| 6B C0 05                        imul ax, 5
 00000028  66| 01 05                   add interm, ax
           00000002 R
 0000002F  B8 00000000                         mov eax, 0
 00000034  A0 00000000 R               mov al, a
 00000039  F6 2D 00000000 R                    imul a
 0000003F  66| A3                      mov interm1, ax
           00000004 R
 00000045  BB 00000000                         mov ebx, 0
 0000004A  A0 00000000 R               mov al, a
 0000004F  F6 2D 00000001 R                    imul b
 00000055  66| 6B C0 02                        imul ax, 2
 00000059  66| 01 05                   add interm1, ax
           00000004 R
 00000060  66| 87 05                   xchg interm1, ax
           00000004 R
 00000067  66| 87 05                   xchg ax, interm
           00000002 R
 0000006E  66| 99                      cwd
 00000070  66| F7 35                   div interm
           00000002 R
```

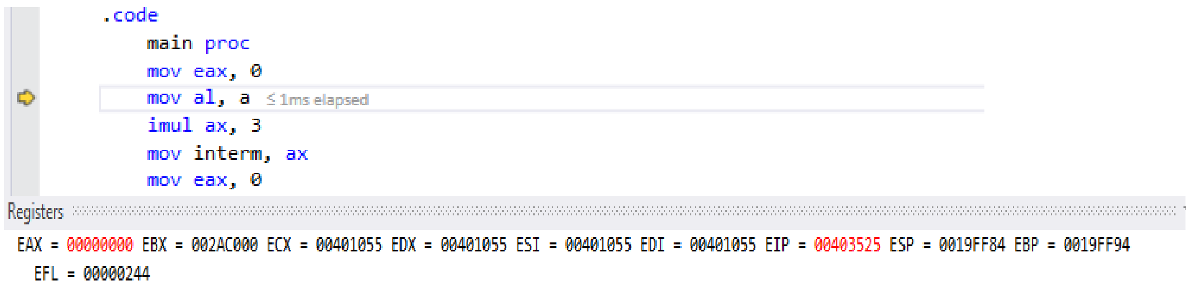
```
00000077  2A 05 00000000 R                  sub al, a
0000007D  2A 05 00000001 R                  sub al, b
00000083  E8 00000000 E             call WriteInt
                                    exit
00000088  6A 00          *      push   +000000000h
0000008A  E8 00000000 E   *          call   ExitProcess
0000008F                            main ENDP
                                    END main
```



```
.code
    main proc
    mov eax, 0
    mov al, a
    imul ax, 3
    mov interm, ax
    mov eax, 0
    mov al, b
    imul b
    imul ax, 5
```

Registers

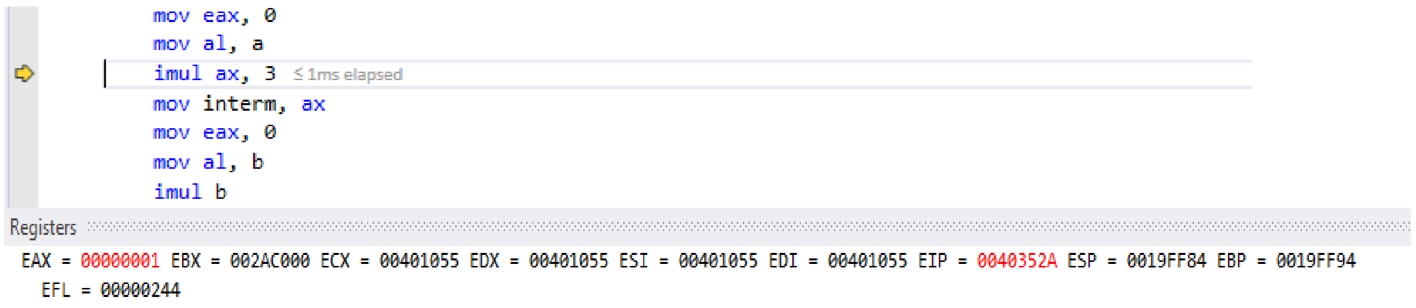
```
EAX = 015D4710 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403520 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000244
```



```
.code
    main proc
    mov eax, 0
    mov al, a  ≤ 1ms elapsed
    imul ax, 3
    mov interm, ax
    mov eax, 0
```

Registers

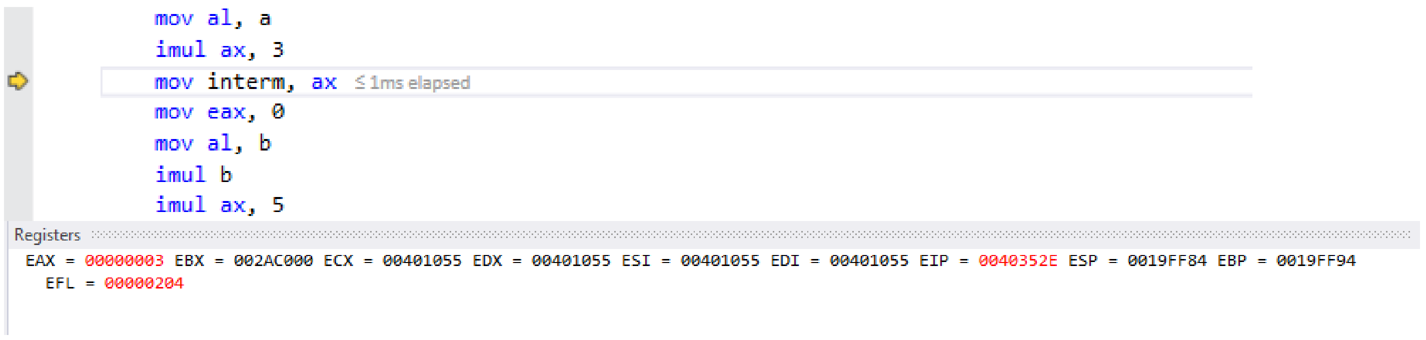
```
EAX = 00000000 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403525 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000244
```



```
    mov eax, 0
    mov al, a
    imul ax, 3  ≤ 1ms elapsed
    mov interm, ax
    mov eax, 0
    mov al, b
    imul b
```

Registers

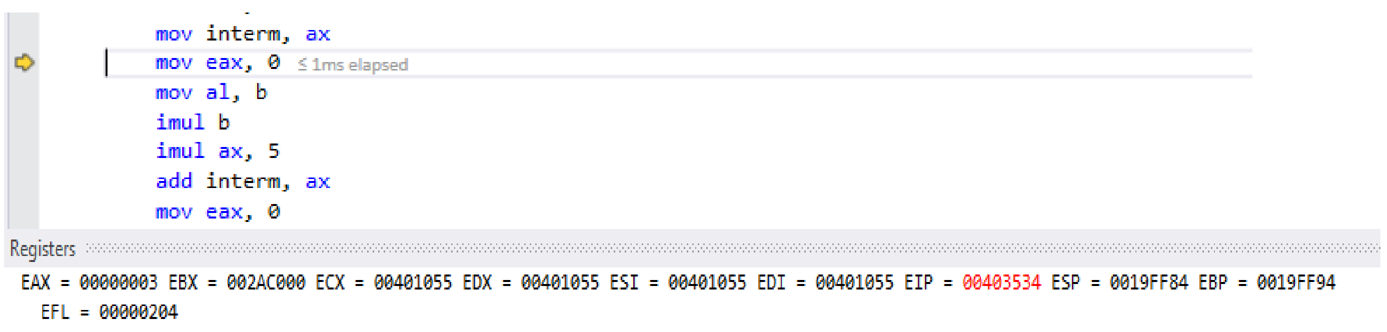
```
EAX = 00000001 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040352A ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000244
```



```
    mov al, a
    imul ax, 3
    mov interm, ax  ≤ 1ms elapsed
    mov eax, 0
    mov al, b
    imul b
    imul ax, 5
```

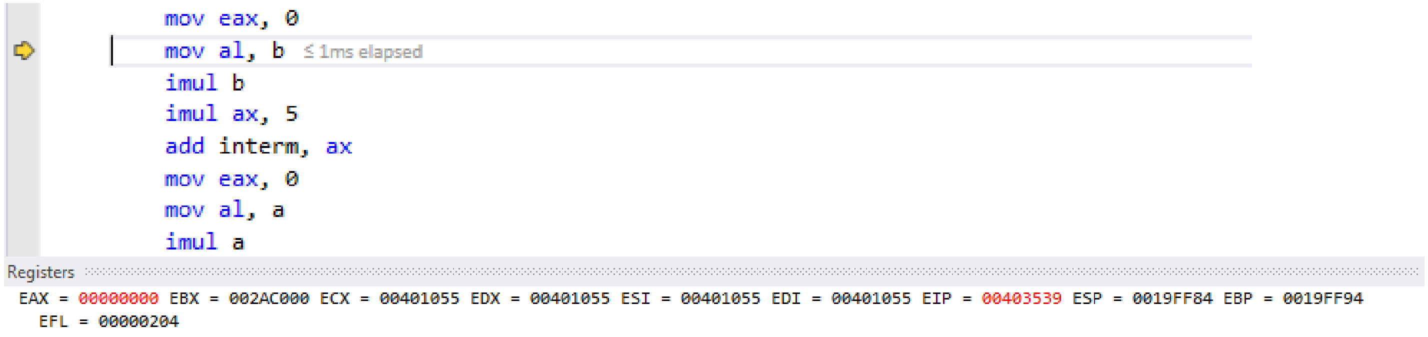
Registers

```
EAX = 00000003 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040352E ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



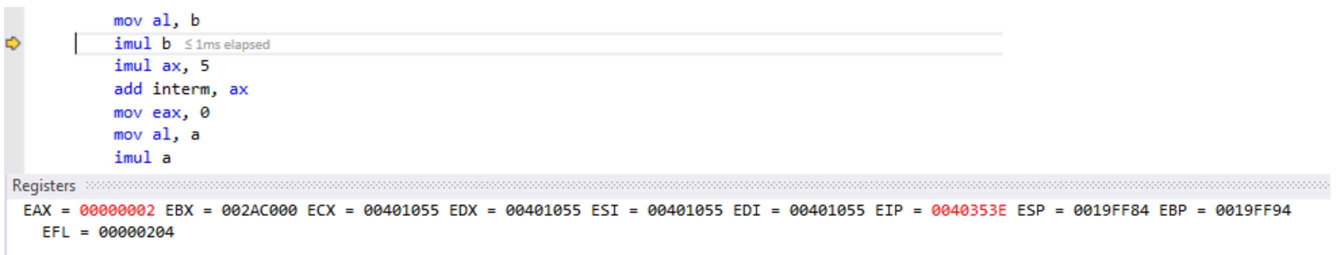
```
mov interm, ax
mov eax, 0  ≤ 1ms elapsed
mov al, b
imul b
imul ax, 5
add interm, ax
mov eax, 0
```

```
Registers
EAX = 00000003 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403534 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



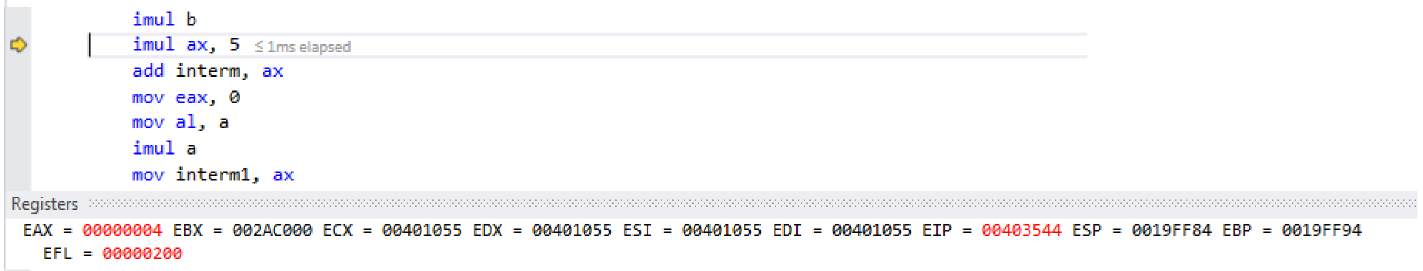
```
mov eax, 0
mov al, b  ≤ 1ms elapsed
imul b
imul ax, 5
add interm, ax
mov eax, 0
mov al, a
imul a
```

```
Registers
EAX = 00000000 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403539 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



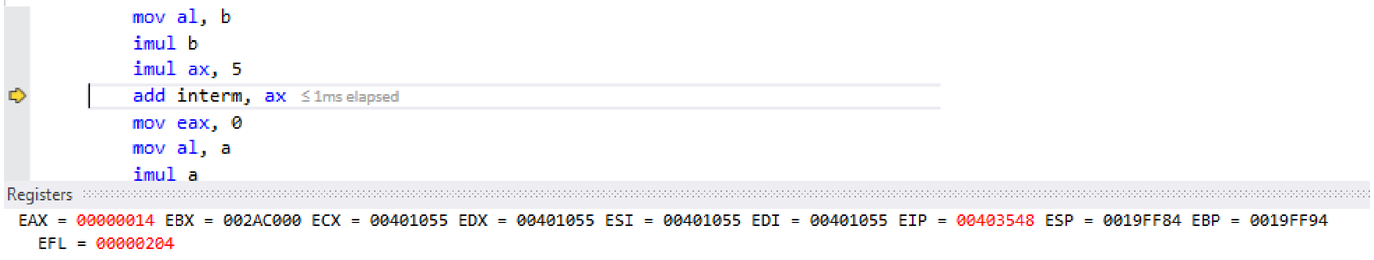
```
mov al, b
imul b  ≤ 1ms elapsed
imul ax, 5
add interm, ax
mov eax, 0
mov al, a
imul a
```

```
Registers
EAX = 00000002 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040353E ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



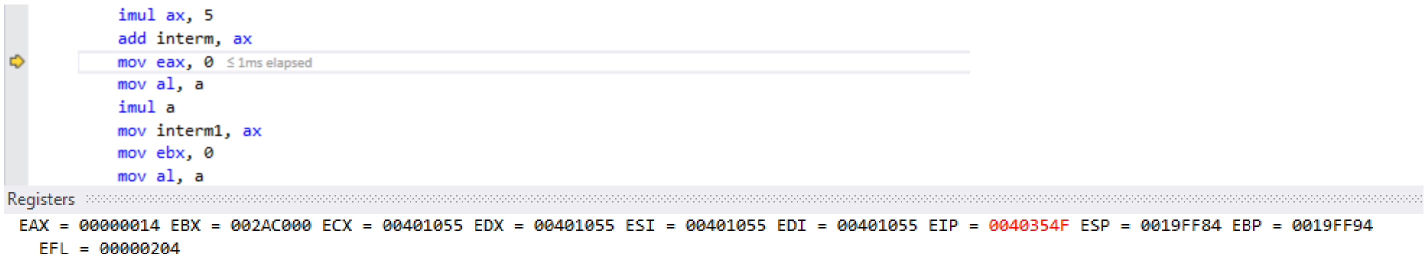
```
imul b
imul ax, 5  ≤ 1ms elapsed
add interm, ax
mov eax, 0
mov al, a
imul a
mov interm1, ax
```

```
Registers
EAX = 00000004 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403544 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000200
```



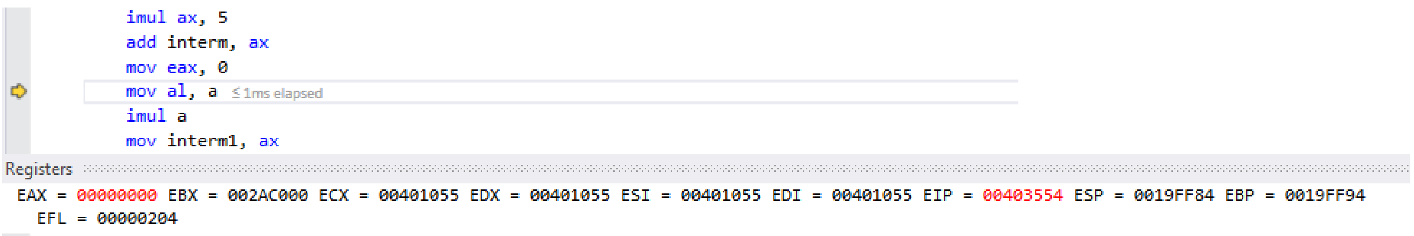
```
mov al, b
imul b
imul ax, 5
add interm, ax  ≤ 1ms elapsed
mov eax, 0
mov al, a
imul a
```

```
Registers
EAX = 00000014 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403548 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



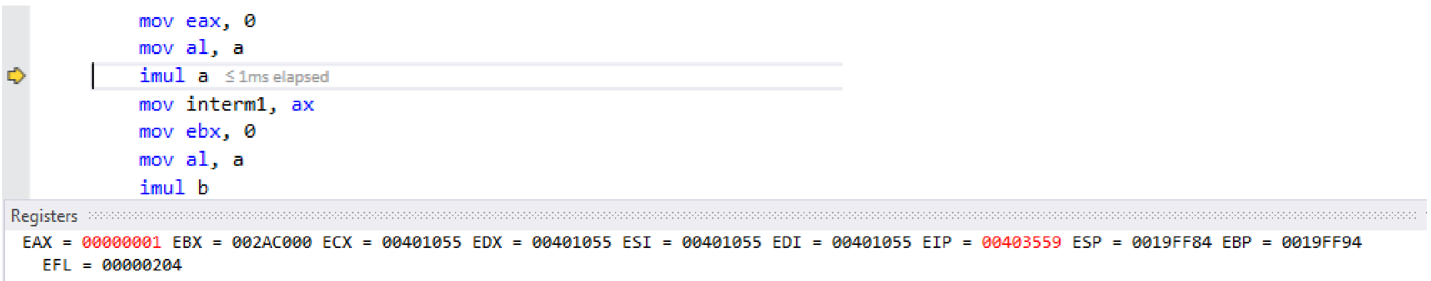
```
imul ax, 5
add interm, ax
mov eax, 0  ≤ 1ms elapsed
mov al, a
imul a
mov interm1, ax
mov ebx, 0
mov al, a
```

```
Registers
EAX = 00000014 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040354F ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



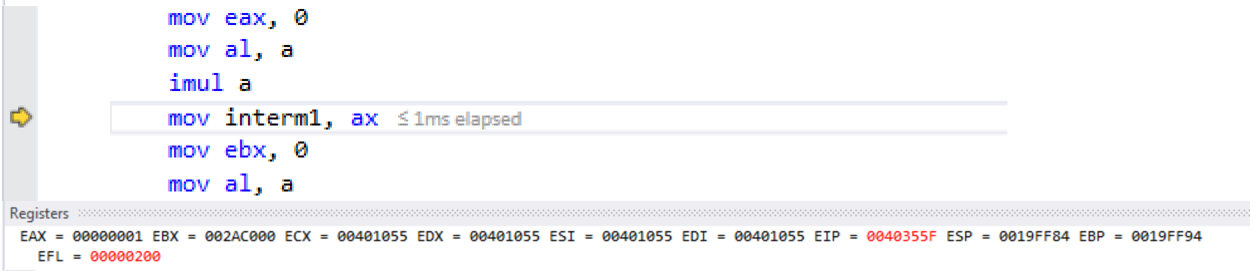
```
imul ax, 5
add interm, ax
mov eax, 0
mov al, a  ≤ 1ms elapsed
imul a
mov interm1, ax
```

```
Registers
EAX = 00000000 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403554 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



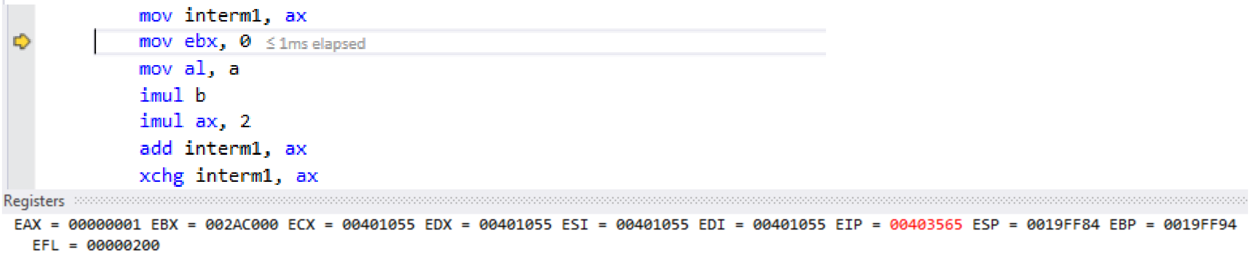
```
mov eax, 0
mov al, a
imul a  ≤ 1ms elapsed
mov interm1, ax
mov ebx, 0
mov al, a
imul b
```

```
Registers
EAX = 00000001 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403559 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



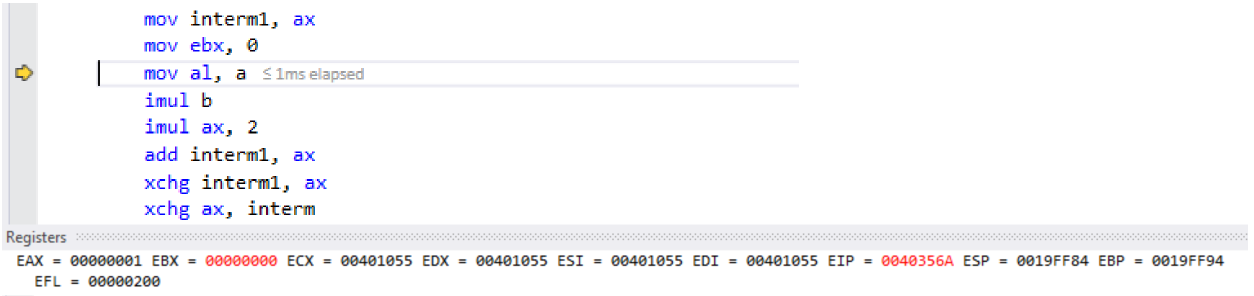
```
mov eax, 0
mov al, a
imul a
mov interm1, ax  ≤ 1ms elapsed
mov ebx, 0
mov al, a
```

```
Registers
EAX = 00000001 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040355F ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000200
```



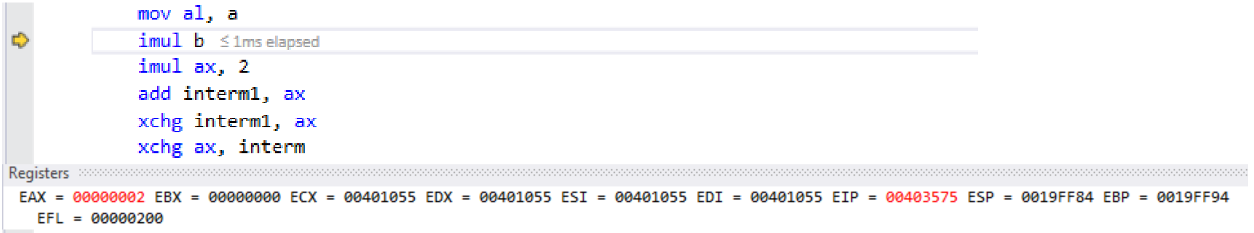
```
mov interm1, ax
mov ebx, 0  ≤ 1ms elapsed
mov al, a
imul b
imul ax, 2
add interm1, ax
xchg interm1, ax
```

```
Registers
EAX = 00000001 EBX = 002AC000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403565 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000200
```



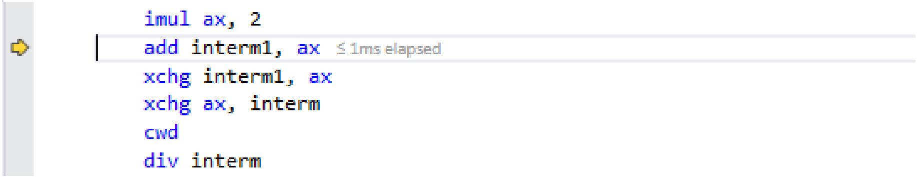
```
mov interm1, ax
mov ebx, 0
mov al, a  ≤ 1ms elapsed
imul b
imul ax, 2
add interm1, ax
xchg interm1, ax
xchg ax, interm
```

```
Registers
EAX = 00000001 EBX = 00000000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040356A ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000200
```

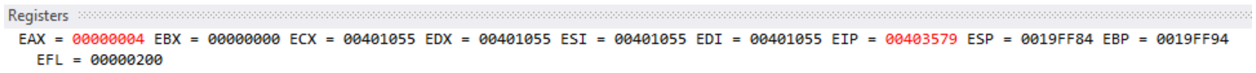


```
mov al, a
imul b  ≤ 1ms elapsed
imul ax, 2
add interm1, ax
xchg interm1, ax
xchg ax, interm
```

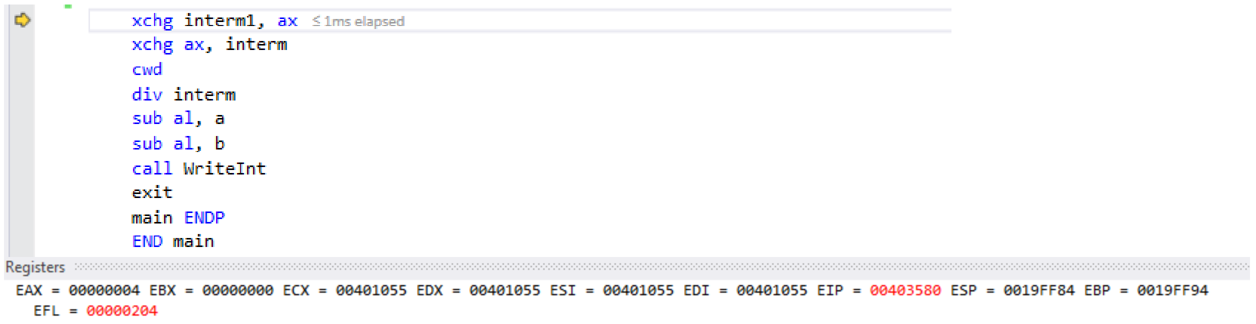
```
Registers
EAX = 00000002 EBX = 00000000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403575 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000200
```



```
imul ax, 2
add interm1, ax  ≤ 1ms elapsed
xchg interm1, ax
xchg ax, interm
cwd
div interm
```

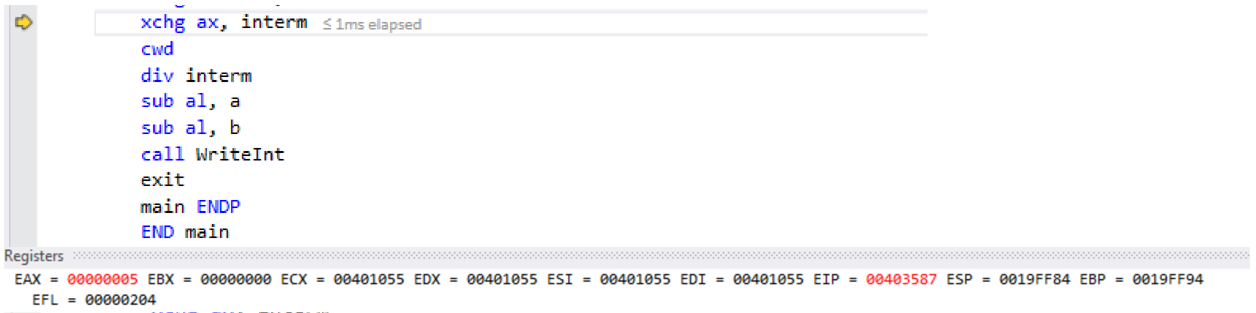


```
Registers
EAX = 00000004 EBX = 00000000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403579 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000200
```



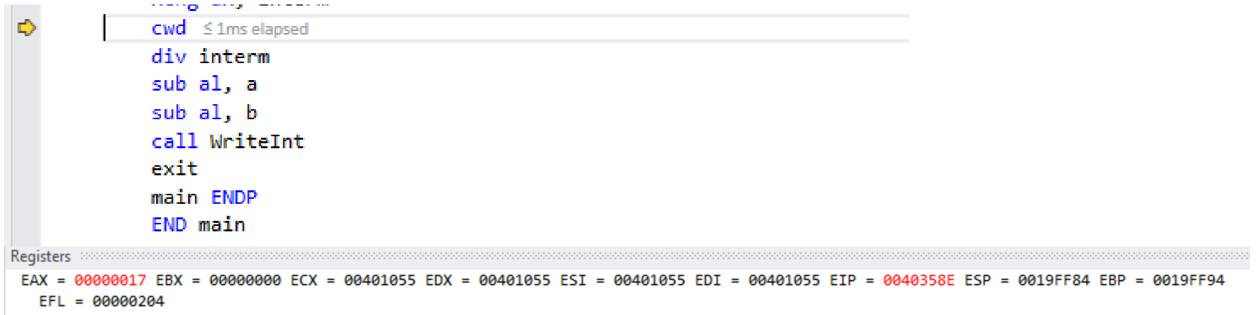
```
xchg interm1, ax  ≤ 1ms elapsed
xchg ax, interm
cwd
div interm
sub al, a
sub al, b
call WriteInt
exit
main ENDP
END main
```

```
Registers
EAX = 00000004 EBX = 00000000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403580 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



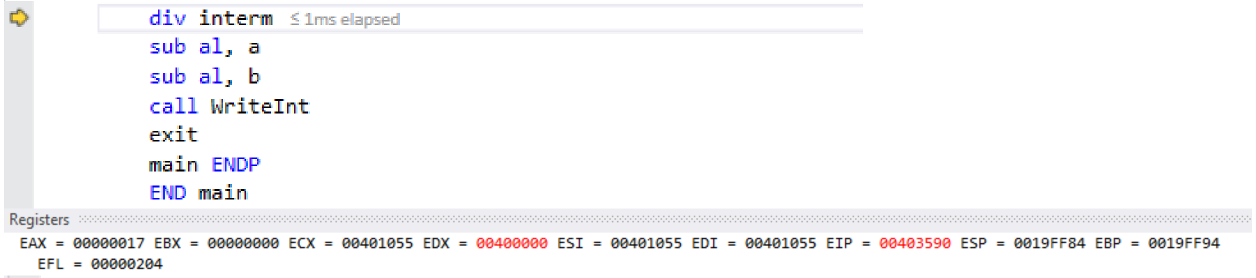
```
xchg ax, interm  ≤ 1ms elapsed
cwd
div interm
sub al, a
sub al, b
call WriteInt
exit
main ENDP
END main
```

```
Registers
EAX = 00000005 EBX = 00000000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 00403587 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



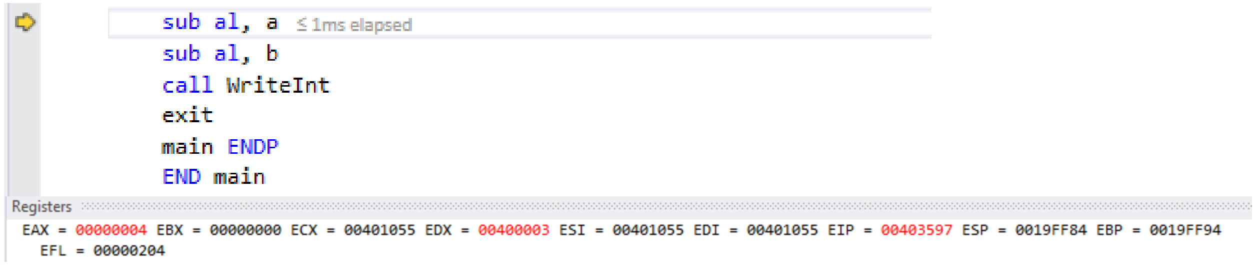
```
cwd  ≤ 1ms elapsed
div interm
sub al, a
sub al, b
call WriteInt
exit
main ENDP
END main
```

```
Registers
EAX = 00000017 EBX = 00000000 ECX = 00401055 EDX = 00401055 ESI = 00401055 EDI = 00401055 EIP = 0040358E ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



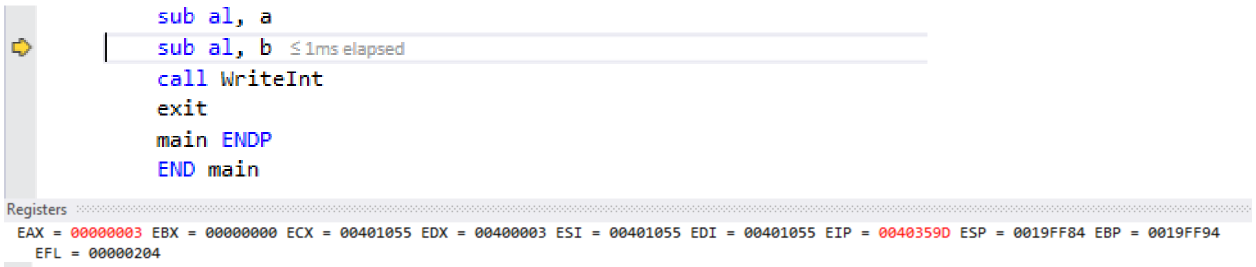
```
div interm  ≤ 1ms elapsed
sub al, a
sub al, b
call WriteInt
exit
main ENDP
END main
```

```
Registers
EAX = 00000017 EBX = 00000000 ECX = 00401055 EDX = 00400000 ESI = 00401055 EDI = 00401055 EIP = 00403590 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



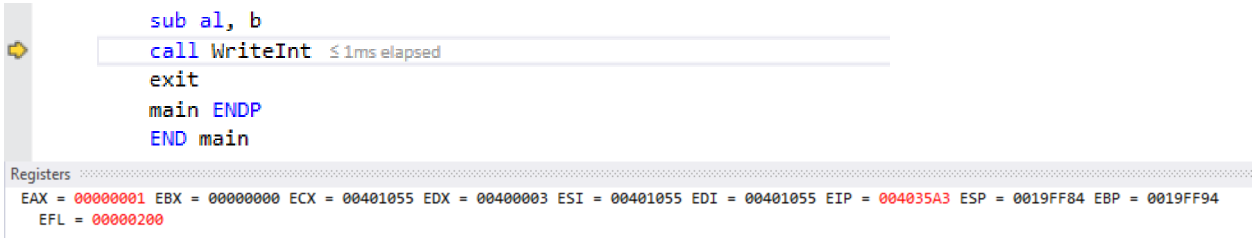
```
sub al, a  ≤ 1ms elapsed
sub al, b
call WriteInt
exit
main ENDP
END main
```

```
Registers
EAX = 00000004 EBX = 00000000 ECX = 00401055 EDX = 00400003 ESI = 00401055 EDI = 00401055 EIP = 00403597 ESP = 0019FF84 EBP = 0019FF94
  EFL = 00000204
```



```
        sub al, a
        sub al, b  ≤ 1ms elapsed
        call WriteInt
        exit
        main ENDP
        END main
```

```
Registers
 EAX = 00000003 EBX = 00000000 ECX = 00401055 EDX = 00400003 ESI = 00401055 EDI = 00401055 EIP = 0040359D ESP = 0019FF84 EBP = 0019FF94
   EFL = 00000204
```



```
        sub al, b
        call WriteInt  ≤ 1ms elapsed
        exit
        main ENDP
        END main
```

```
Registers
 EAX = 00000001 EBX = 00000000 ECX = 00401055 EDX = 00400003 ESI = 00401055 EDI = 00401055 EIP = 004035A3 ESP = 0019FF84 EBP = 0019FF94
   EFL = 00000200
```

Concluzie:
In lucrarea de laborator nr.2 am realizat un program ce efectueaza operatia de calculare a unei valori.Am utilizat diferite tipuri de date(**db-define byte,dw-define word**),instructiuni cum ar fi (**add**,**div,imul,xchg,cwd)** si modurile de adresare a operanzilor.In general am obtinut abilitati in bazele limbajului Assembler.