Tema laborator 4

Pop Mihai-Daniel, Grupa 215/2

Problema nr. 27:

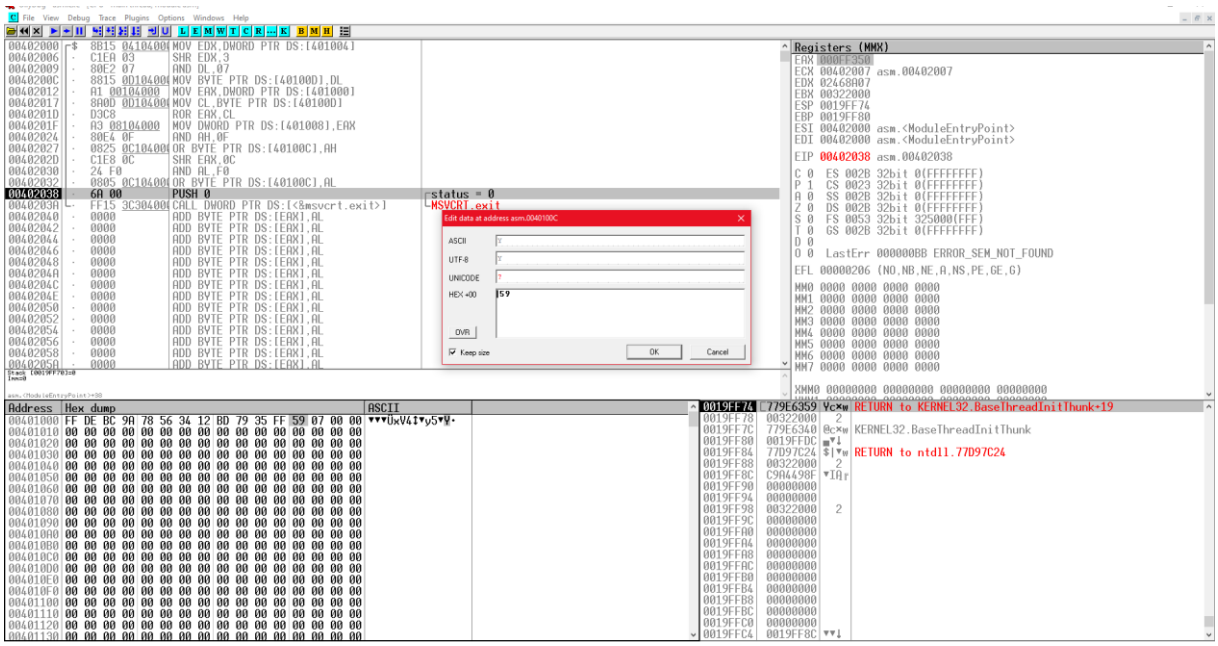
27. Se da quadwordul A. Sa se obtina numarul intreg N reprezentat de bitii 35-37 ai lui A. Sa se obtina apoi in B dublucuvantul rezultat prin rotirea spre dreapta a dublucuvantului inferior al lui A cu N pozitii. Sa se obtina octetul C astfel:

- bitii 0-3 ai lui C sunt bitii 8-11 ai lui B
- bitii 4-7 ai lui C sunt bitii 16-19 ai lui B

```
;Exercitiul 27:
;Se da quadwordul A. Sa se obtina numarul intreg N reprezentat de bitii 35-37 ai lui A. Sa se obtina apoi in B dublucuvantul rezultat prin ;rotirea spre dreapta a dublucuvantului inferior al lui A cu N pozitii. Sa se obtina octetul C astfel:
;bitii 0-3 ai lui C sunt bitii 8-11 ai lui B
;bitii 4-7 ai lui C sunt bitii 16-19 ai lui B

bits 32 ; assembling for the 32 bits architecture
; declare the EntryPoint (a label defining the very first instruction of the program)
global start

; declare external functions needed by our program
extern exit             ; tell nasm that exit exists even if we won't be defining it
import exit msvcrt.dll    ; exit is a function that ends the calling process. It is defined in msvcrt.dll
                        ; msvcrt.dll contains exit, printf and all the other important C-runtime specific functions

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    ; ...
    a dq 123456789ABCDEFFh
    ; = 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 1111
    b dd 0
    c db 0
    n db 0

; our code starts here
segment code use32 class=code
    start:
        ; ...
```

```
mov edx, dword[a+4] ;edx = 12345678 h = 0001 0010 0011 0100 0101 0110 0111 1000b
;salvam in registrul edx dublucuvantul superior al quadword-ului a
shr edx, 3 ;edx = 02468ACFh = 0000 0010 0100 0110 1000 1010 1100 1111b
;shiftam spre dreapta cu 3 pozitii edx, astfel bitii 35-37 ai numarului a ajung pe primii biti
and dl, 00000111b ; dl = 07h = 00000111b
;salvam in dl bitii 35-37 ai lui a
mov [n],dl ;n = dl = 07h
;punem in n numarul intreg format din bitii 35-37 ai numarului a

mov eax, dword[a] ;eax = 9ABCDEFFh = 1001 1010 1011 1100 1101 1110 1111 1111b
;salvam in registrul eax dublucuvantul inferior al quadword-ului a
mov cl, [n] ;cl = n = 7
;mutam in cl valoarea numarului intreg n
ror eax, cl ;eax = FF3579BDh = 1111 1111 0011 0101 0111 1001 1011 1101b
;rotim spre dreapta registrul eax cu n pozitii
mov [b], eax ;b = eax = FF3579BDh
;punem in b numarul format prin rotirea sprea dreapta cu n pozitii a dword-ului inferior a lui a

and ah, 00001111b ;ah = 09h = 0000 1001b
;eax = 1111 1111 0011 0101 0000 1001 1011 1101b
or [c], ah ;c = 09h = 0000 1001b
;salvam in numarul c pe pozitia bitilor 0-3 bitii 8-11 din b
shr eax, 12 ;eax = 000FF350h = 0000 0000 0000 1111 1111 0011 0101 0000b
;shiftam spre dreapta registrul eax, pentru a avea pe pozitia bitilor 4-7 din octetul al bitii 16-19 ai
numarului b
and al, 11110000b ;al = 50h = 0101 0000b
;salvam din al bitii 4-7
or [c], al ;c = 59h = 0101 1001b
;punen in c numarul format din prelucrarea lui b
; exit(0)
push    dword 0      ; push the parameter for exit onto the stack
```

```
call    [exit]      ; call exit to terminate the program
```

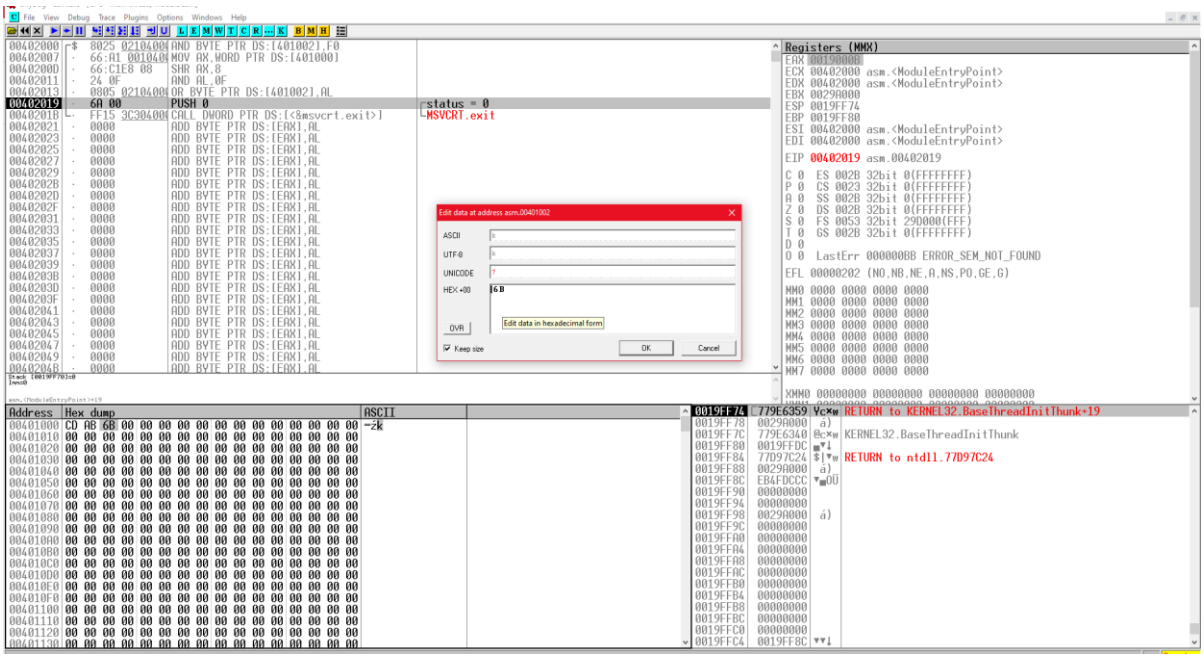


Problema nr. 10:

10. Sa se inlocuiasca bitii 0-3 ai octetului B cu bitii 8-11 ai cuvantului A.

```
;Exercitiul 10:
;Sa se inlocuiasca bitii 0-3 ai octetului B cu bitii 8-11 ai cuvantului A.

bits 32 ; assembling for the 32 bits architecture
global start
extern exit             ; tell nasm that exit exists even if we won't be defining it
import exit msvcrt.dll    ; exit is a function that ends the calling process. It is defined in msvcrt.dll
                        ; msvcrt.dll contains exit, printf and all the other important C-runtime specific
functions

; our data is declared here (the variables needed by our program)
segment data use32 class=data
    ; ...
    a dw 0ABCDh ;= 1010 1011 1100 1101b
    b db 100 ;= 0110 0100b

; our code starts here
segment code use32 class=code
    start:
        ; ...
        and byte[b], 11110000b ;b = 60h = 0110 0000b
        ;punem in bitii 0-3 ai numarului b valoarea 0
        mov ax, [a] ;ax = ABCDh = 1010 1011 1100 1101b
```

;mutam in ax valoarea numarului a
shr ax, 8 ;ax = 00ABh = 0000 0000 1010 1011b
;shiftam numarul a cu 8 pozitii spre dreapta
;/mutam spre dreapta bitii 8-11 ai numarului a pe pozitia bitilor 0-3
and al, 00001111b ;al = 0Bh = 0000 1011b
;salvam in al acesti biti
or [b], al ;b = 6Bh = 0110 1011b
;modificam in b bitii 0-3 cu cei salvati in al ; exit(0)
push dword 0 ; push the parameter for exit onto the stack
call [exit] ; call exit to terminate the program



Problema nr. 13:

13. Dandu-se 4 octeti, sa se obtina in AX suma numerelor intregi reprezentate de bitii 4-6 ai celor 4 octeti.

;Exercitiul 13:
;Dandu-se 4 octeti, sa se obtina in AX suma numerelor intregi reprezentate de bitii 4-6 ai celor 4 octeti.

bits 32 ; assembling for the 32 bits architecture

; declare the EntryPoint (a label defining the very first instruction of the program)
global start

; declare external functions needed by our program
extern exit ; tell nasm that exit exists even if we won't be defining it
import exit msvcrt.dll ; exit is a function that ends the calling process. It is defined in msvcrt.dll
; msvcrt.dll contains exit, printf and all the other important C-runtime specific functions

```
; our data is declared here (the variables needed by our program)
segment data use32 class=data
   ; ...
   a db 0ABh ;= 1010 1011b
   b db 0BCh ;= 1011 1100b
   c db 0CDh ;= 1100 1101b
   d db 0DEh ;= 1101 1110b

; our code starts here
segment code use32 class=code
   start:
      ; ...
      mov dl, [a] ;dl = a = ABh = 1010 1011b
      ;punem in dl valoarea octetului a
      shr dl, 4 ;dl = 0Ah = 0000 1010b
      ;shiftam spre dreapta cu 4 pozitii pentru a salva doar bitii 4-7 ai numarului initial a
      and dl, 00000111b ;dl = 02h = 0000 0010b
      ;salvam doar bitii 4-6 din numarul a, pe care i-am pus pe pozitia bitilor 0-2
      mov dh, 0 ;dh = 0
      mov ax, 0 ;ax = 0
      add ax, dx ;ax = ax+dx = 0+2 = 2

      mov dl, [b] ;dl = b = BCh = 1011 1100b
      ;punem in dl valoarea octetelui b
      shr dl, 4 ;dl = 0Bh = 0000 1011b
      ;shiftam spre dreapta cu 4 pozitii pentru a salva doar bitii 4-7 ai numarului initial b
      and dl, 00000111b ;dl = 03h = 0000 0011b
      ;salvam doar bitii 4-6 din numarul b, pe care i-am pus pe pozitia bitilor 0-2
      mov dh, 0 ;dh = 0
      add ax, dx ;ax = ax+dx = 2+3 = 5

      mov dl, [c] ;dl = c = CDh = 1100 1101b
      ;punem in dl valoarea octetelui c
      shr dl, 4 ;dl = 0Ch = 0000 1100b
      ;shiftam spre dreapta cu 4 pozitii pentru a salva doar bitii 4-7 ai numarului initial c
      and dl, 00000111b ;dl = 04h = 0000 0100b
      ;salvam doar bitii 4-6 din numarul c, pe care i-am pus pe pozitia bitilor 0-2
      mov dh, 0 ;dh = 0
      add ax, dx ;ax = ax+dx = 5+4 = 9

      mov dl, [d] ;dl = d = DEh = 1101 1110b
      ;punem in dl valoarea octetelui d
      shr dl, 4 ;dl = 0Dh = 0000 1101b
      ;shiftam spre dreapta cu 4 pozitii pentru a salva doar bitii 4-7 ai numarului initial d
      and dl, 00000111b ;dl = 05h = 0000 0101b
      ;salvam doar bitii 4-6 din numarul d, pe care i-am pus pe pozitia bitilor 0-2
      mov dh, 0 ;dh = 0
      add ax, dx ;ax = ax+dx = 9+5 = 14
```

```
;rezultatul este salvat in registrul ax
; exit(0)
push    dword 0     ; push the parameter for exit onto the stack
call    [exit]      ; call exit to terminate the program
```