

EXEMPLU DE UTILIZARE ȘI EXPLOATARE A MEMORIEI INTERNE

Să se determine harta memoriei pentru datele de mai jos și offset-urile (deplasările) șirurilor

```
segm_stiva SEGMENT
    stiva    DB 128 DUP (?)
segm_stiva ENDS
```

```
segm_date SEGMENT
    sir_date  DB    7,10,11, ?
    sir_date2 DW    7,10,11, ?
    sir_car   DB    'ABC 123'
    car       DB    '13'
    nr        DB    13
    nr_mare  DD    1abcdefh
segm_date ENDS
```

```
segm_cod  SEGMENT
    ASSUME CS:segm_cod, DS:segm_date, SS:segm_stiva
```

```
et:      MOV AX, segm_date
         MOV DS, AX
```

```

MOV SI, offset sir_date      ; SI = 00 00
MOV AL, [SI]                   ; AX= ....07
MOV BL, [SI+1]                 ; BX= ....0A
MOV CL, [SI+2]                 ; CX= ....0B
MOV [SI+3], AL                 ; valoarea 07 se mută la adresa 0003
```

```

MOV DI, OFFSET sir_date2    ; DI = 00 04
MOV AX, [DI]                   ; AX= 00 07
MOV BX, [DI+2]                 ; BX= 00 0A
MOV CX, [DI+4]                 ; CX= 00 0B
MOV [DI+6], AX                 ; valoarea 00 07 se mută la adresa
```

000A

```

MOV SI, OFFSET sir_car      ; SI = 00 0C
MOV AL, [SI]                   ; AX= ....41
MOV BL, [SI+1]                 ; BX=.... 42
MOV CL, [SI+2]                 ; CX=.... 43
```

```

MOV DI, OFFSET car         ; DI = 00 13
MOV AL,car                     ; AX= .....31
MOV AH,nr                      ; AX= 0D ...
```

```

MOV SI, OFFSET nr_mare     ; SI = 00 16
MOV AX, WORD PTR [nr_mare]     ; AX= CD EF
MOV BX, WORD PTR [nr_mare+2]   ; BX= 01 AB
```

```
MOV AX, 4C00H
INT 21H
```

```
segm_cod ENDS
```

```
END et
```

Harta memoriei la începutul programului:

<i>sir_date</i>	<i>sir_date2</i>	<i>sir_car</i>	<i>car</i>	<i>nr</i>
-----	-----	-----	-----	---
07 0A 0B 00	07 00 0A 00 0B 00 00 00	41 42 43 20 31 32 33	31 33	0D
EF CD AB 01	-----			
<i>nr_mare</i>				

OBS.: În memorie, numerele la nivel de B (byte) și caracterele (șirurile de caractere) sunt reprezentate în ordinea normală, iar numere la nivel de W (word) și DD (double word) sunt reprezentate invers (se inversează byte-s între ei) – reprezentarea „little endian”

The screenshot shows a DOS debugger window titled "C:\WINDOWS\system32\cmd.exe - td sir.exe". The CPU is at address 80486. The assembly code is as follows:

```

cs:0026 8A5C01  mov  bl,[si+01]
cs:0029 8A4C02  mov  cl,[si+02]
cs:002C BF1300    mov  di,0013
cs:002F A01300    mov  al,[0013]
cs:0032 8A261500  mov  ah,[0015]
cs:0036 BE1600    mov  si,0016
cs:0039 A11600    mov  ax,[0016]
cs:003C 8B1E1800  mov  bx,[0018]
cs:0040 B8004C    mov  ax,4C00
cs:0043 CD21    int  21
cs:0045 0000    add  [bx+si],al
cs:0047 0000    add  [bx+si],al
cs:0049 0000    add  [bx+si],al
cs:004B 0000    add  [bx+si],al
cs:004D 0000    add  [bx+si],al

```

Register values are shown on the right:

```

ax 4C00  c=0
bx 01AB  z=0
cx 0043  s=0
dx 0000  o=0
si 0016  p=0
di 0013  a=0
bp 0000  i=1
sp 0000  d=0
ds 4E90
es 4E98
ss 4E98
cs 4E92
ip 0043

```

A memory dump is visible at the bottom, showing data in the data segment (ds):

```

ds:0000 07 0A 0B 07 07 00 0A 00 00 00 00 00 00 00 00
ds:0008 0B 00 07 00 41 42 43 20 31 32 33 0D EF CD 12313Fn=
ds:0010 31 32 33 31 33 0D EF CD 12313Fn=
ds:0018 AB 01 00 00 00 00 00 00 00 00 00 00 00 00

```

Harta memoriei la sfârșitul programului:

```

sir_date          sir_date2          sir_car          car          nr
-----
07 0A 0B 07  07 00 0A 00 0B 00 07 00  41 42 43 20 31 32 33  31 33  0D
EF CD AB 01
-----
nr_mare

```

Moduri de reprezentare nr

- negative – in CC,
- nr reale – in VMSP => pe 4B folosind DD (Define Double Word)

```

segm_stiva SEGMENT
stiva  DB 128 DUP (?)
segm_stiva ENDS

```

```

segm_date SEGMENT
rez1  DB  ?
rez2  DB  ?
rez3  DB  ?
nr1   DB  -128  ;(80)H
nr2   DW  -128  ;(80 FF)H
nr3   DB  -130  ;(7E)H
nr4   DW  -130  ;(7E FF)H
nr5   DB  +255  ;(FF)H
; nr6  DB  +256  -da eroare la reprez -"value out of range"
; nr6_6 DB  -257  -da eroare la reprez -"value out of range"
nr6_7 DB  -256  ;(00)H
nr7   DW  +256  ;(00 01)H

sir_date DB  7,-13,130, ? ;(07, F3, 82, 00)H
sir_date2 DW  7,-13,+130,? ;(07 00, F3 FF, 82 00, 00 00)H
sir_date3 DD  7.0, -13.0, +130.0, ? ; (00 00 E0 40, 00 00 50 C1, 00 00 02 43)H
segm_date ENDS

```

```
segm_cod SEGMENT
```

```
ASSUME CS:segm_cod, DS:segm_date, SS:segm_stiva
```

```
et: MOV AX, segm_date
```

```
MOV DS, AX
```

```
MOV AL, -1
```

```
ADD AL, -127
```

```
MOV rez1, AL
```

```
MOV AH, -128
```

```
ADD AH, 1
```

```
MOV rez2, AH
```

```
MOV BL, -128
```

```
ADD BL, -2 ; depasire interval reprez pt 8b; ramane rez inainte de adunare, si O=1
```

```
MOV rez3, BL
```

```
MOV SI, offset sir_date
```

```
MOV AL, [SI]
```

```
ADD AL, [SI+1]
```

```
SUB AL, [SI+2]
```

```
MOV [SI+3], AL
```

```
MOV DI, offset sir_date2
```

```
MOV AX, [DI]
```

```
ADD AX, [DI+2]
```

```
SUB AX, [DI+4]
```

```
MOV [DI+6], AX
```

```
MOV AX, 4C00H
```

```
INT 21H
```

```
segm_cod ENDS
```

```
END et
```

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frames...
File View Run Breakpoints Data Options Window Help
[ ]-CPU 80486- [ ]
cs:0005 B0FF mov al,FF ax 8180 c=0
cs:0007 0481 add al,B1 bx 0000 z=0
cs:0009 A20000 mov [0000],al cx 0000 s=1
cs:000C B480 mov ah,80 dx 0000 o=0
cs:000E 80C401 add ah,01 si 0000 p=1
cs:0011 88260100 mov [0001],ah di 0000 a=0
cs:0015 B380 mov bl,80 bp 0000 i=1
cs:0017 80C3FE add bl,FE sp 0000 d=0
cs:001A 881E0200 mov [0002],bl ds 44B5
cs:001E BE0000 mov si,0000 es 449D
cs:0021 8A04 mov al,[si] ss 44AC
cs:0023 024401 add al,[si+01] cs 44B8
cs:0026 2A4402 sub al,[si+02] ip 0015
cs:0029 8B4403 mov [si+03],al
cs:002C BF1100 mov di,0011

ds:0008 FF FF 00 00 01 07 F3 82 0x<é
ds:0010 00 07 00 F3 FF 82 00 00 0x<é
ds:0018 00 00 00 E0 40 00 00 50 0x P
ds:0020 C1 00 00 02 43 00 00 00 0x BC
ds:0028 00 00 00 00 00 00 00 00
```