

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МОЭВМ**

**ОТЧЕТ**  
**по лабораторной работе №2**  
**по дисциплине «Операционные системы»**  
**Тема: “Исследование интерфейсов программных модулей”**

Студент гр. 7381

\_\_\_\_\_

Габов Е. С.

Преподаватель

\_\_\_\_\_

Ефремов М. А.

Санкт-Петербург

2018

### Цель работы.

Исследование интерфейса управляющей программы и загрузочных модулей. Этот интерфейс состоит в передаче запускаемой программе управляющего блока, содержащего адреса и системные данные. Так загрузчик строит префикс сегмента программы (PSP) и помещает его адрес в сегментный регистр. Исследование префикса сегмента программы (PSP) и среды, передаваемой программе.

### Основные теоретические положения.

Смещение	Длина поля	Содержимое поля
<b>0</b>	2	Int 20h
<b>2</b>	2	Сегментный адрес первого байта недоступной памяти.
<b>4</b>	6	Зарезервировано
<b>0Ah (10)</b>	4	Int 22h
<b>0Eh (14)</b>	4	Int 23h
<b>12h (18)</b>	4	Int 24h
<b>2Ch(44)</b>	2	Сегментный адрес среды, передаваемой программе
<b>5Ch</b>		Область форматируется как стандартный неоткрытый блок управления файлом
<b>6Ch</b>		Область форматируется как стандартный неоткрытый блок управления файлом. Перекрывается, если FCB с адреса 5Ch открыт.
<b>80h</b>	1	Число символов в хвосте командной строки
<b>81h</b>		Хвост командной строки – последовательность символов после имени вызываемого файла

Область среды содержит последовательность символьных строк вида:

*имя=параметр*

Каждая строка завершается байтом нулей.

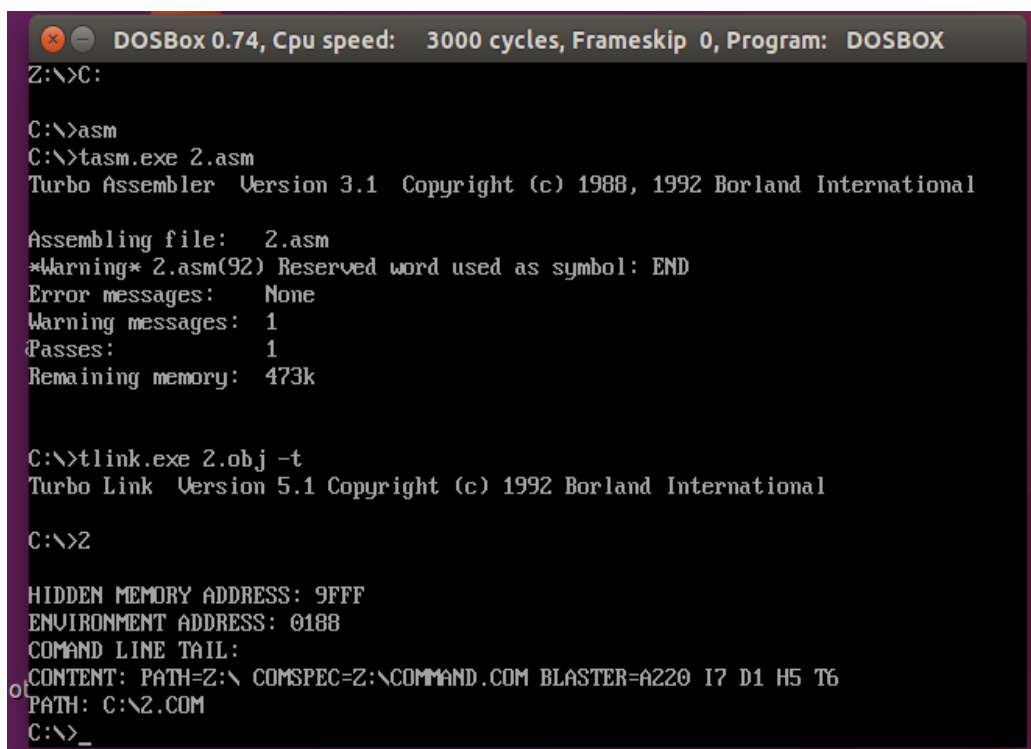
В первой строке указывается имя COMSPEC, которая определяет используемый командный процессор и путь к COMMAND.COM. Следующие строки содержат информацию, задаваемую командами PATH, PROMT, SET.

Среда заканчивается также байтом нулей. Таким образом, два нулевых байта являются признаком конца переменных среды. Затем идут два байта, содержащих 00h, 01h, после которых располагается маршрут загруженной

### **Выполнение работы.**

#### **Ход работы.**

##### **1. Результат работы программы:**



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
Z:\>C:
C:\>asm
C:\>tasm.exe 2.asm
Turbo Assembler Version 3.1 Copyright (c) 1988, 1992 Borland International

Assembling file: 2.asm
*Warning* 2.asm(92) Reserved word used as symbol: END
Error messages: None
Warning messages: 1
Passes: 1
Remaining memory: 473k

C:\>tlink.exe 2.obj -t
Turbo Link Version 5.1 Copyright (c) 1992 Borland International

C:\>Z

HIDDEN MEMORY ADDRESS: 9FFF
ENVIRONMENT ADDRESS: 0188
COMMAND LINE TAIL:
CONTENT: PATH=Z:\ COMSPEC=Z:\COMMAND.COM BLASTER=A220 I7 D1 H5 T6
PATH: C:\Z.COM
C:\>_
```

Рис. 1. Результат работы программы

#### **Ответы на вопросы.**

##### **Сегментный адрес недоступной памяти:**

###### **1. На какую область памяти указывает адрес недоступной памяти?**

На область памяти ROM BIOS. Адрес недоступной памяти указывает на область оперативной памяти, следующей сразу после памяти, выделенной для использования загружаемыми программами.

###### **2. Где расположен этот адрес по отношению области памяти, отведённой программе?**

Недоступная память располагается *после* адреса 9FFF.

**3. Можно ли в эту область памяти писать?**

Можно, т. к. в DOS отсутствует защита памяти.

**Среда передаваемая программе:**

**1. Что такое среда?**

Среда – это область памяти, в которой в виде символьных строк записаны значения переменных (имя=параметр), называемых переменными среды. Они содержат данные о некоторых директориях операционной системы и конфигурации компьютера, которые передаются программе, когда она запускается.

**2. Когда создается среда? Перед запуском приложения или в другое время?**

Среда создаётся при загрузке DOS. При запуске программы эта среда только копируется в новую область памяти.

**3. Откуда берется информация, записываемая в среду?**

Информация для записи берётся из системного файла AUTOEXEC.BAT.

**Вывод.**

В ходе лабораторной работы были исследованы интерфейсы управляющей программы и загрузочных модулей, а также префикс сегмента программы (PSP) и среды, передаваемой программе. Была написана программа, которая выводит на экран сегментный адрес недоступной памяти, адрес среды, передаваемой программе, хвост командной строки и путь загружаемого модуля.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ ТЕКСТ

```

TESTPC SEGMENT
    ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
    ORG 100H
START:
    jmp BEGIN
;ДАННЫЕ
UNAVAILABLE_MEM          db          'Segment address of unavailable
memory: $'
UNAVAILABLE_MEM_VALUE     db          '      ',10,13,'$'
ADRESS_GET                db          'Segment address of the environment:
$'
ADRESS_GET_VALUE          db          '????',10,13,'$'
TAIL_M                    db 'Tail:      ', '$'
TAILEM                    db 50h DUP(' '), '$'
NOTAIL                    db 'There is no tail', 10, 13, '$'
SREDA_STR                 db 'The contents of the environment area in the
symbolic form: ', 10,13,'$'
PATH_STR                  db 'Load module path: ',10,13,'$'

TETR_TO_HEX              PROC  near
    and     AL,0Fh
    cmp     AL,09
    jbe     NEXT
    add     AL,07
NEXT:
    add     AL,30h
    ret
TETR_TO_HEX              ENDP

BYTE_TO_HEX              PROC  near
;байт в AL переводится в два символа шестн. числа в AX
    push    CX
    mov     AH,AL
    call    TETR_TO_HEX
    xchg    AL,AH
    mov     CL,4
    shr     AL,CL
    call    TETR_TO_HEX ;в AL старшая цифра
    pop     CX          ;в AH младшая
    ret
BYTE_TO_HEX              ENDP

```

```

WRD_TO_HEX PROC near
    push    BX
    mov     BH,AH
    call    BYTE_TO_HEX
    mov     [DI],AH
    dec     DI
    mov     [DI],AL
    dec     DI
    mov     AL,BH
    call    BYTE_TO_HEX
    mov     [DI],AH
    dec     DI
    mov     [DI],AL
    pop     BX
    ret
WRD_TO_HEX ENDP

BYTE_TO_DEC PROC near
    push    CX
    push    DX
    xor     AH,AH
    xor     DX,DX
    mov     CX,10
Loop_bd:    div     CX
            or      DL,30h
            mov     [SI],DL
            dec     SI
            xor     DX,DX
            cmp     AX,10
            jae     Loop_bd
            cmp     AL,00h
            je      end_L
            or      AL,30h
            mov     [SI],AL
end_L:      pop     DX
            pop     CX
            ret
BYTE_TO_DEC ENDP

PRINT PROC near
    mov     AH,09h
    int     21h

```

```

                                ret
PRINT      ENDP

; Сегментный адрес недопустимой памяти, взятый из PSP
UNAVAILABLE_MEM_F PROC near
                                mov  dx,offset UNAVAILABLE_MEM
                                call  PRINT
                                push  ax
                                mov   ax,es:[2]
                                mov   di, offset UNAVAILABLE_MEM_VALUE
                                add    di,4
                                call   WRD_TO_HEX
                                pop    ax
                                mov    dx,offset UNAVAILABLE_MEM_VALUE
                                call   PRINT
                                ret
UNAVAILABLE_MEM_F ENDP

; Сегментный адрес среды, передаваемой программе
ADRESS_GET_F      PROC near
                                mov    dx,offset ADRESS_GET
                                call    PRINT
                                push    ax
                                mov     ax,es:[2Ch]
                                mov     di,offset ADRESS_GET_VALUE
                                add      di,4
                                call     WRD_TO_HEX
                                pop      ax
                                mov     dx,offset ADRESS_GET_VALUE
                                call     PRINT
                                ret
ADRESS_GET_F      ENDP

; Хвост командной строки в символьном виде
TAIL PROC near
                                xor     ch,ch
                                mov     cl,ss:[80h]

                                cmp     cl,0
                                jne     notnil
                                mov     dx,offset NOTAIL
                                call     PRINT

```

```

        ret
notnil:

mov dx,offset TAIL_M
call PRINT

mov bp,offset TAILEM
T_cycle:
    mov di,cx
    mov bl,ss:[di+80h]
    mov ss:[bp+di-1],bl
loop T_cycle

mov dx,offset TAILEM
call PRINT
ret
TAIL ENDP

SREDA PROC
    mov dx,offset SREDA_STR
    call PRINT
    push es
    mov ax,es:[2Ch]
    mov es,ax
    mov ah,02h
    mov bx,0
SREDA_Loop:
    mov dl,es:[bx]
    int 21h
    inc bx
    cmp byte ptr es:[bx],00h
    jne SREDA_Loop;
    cmp word ptr es:[bx],0000h
    jne SREDA_Loop

    add bx,4
    mov dx,offset PATH_STR
    call PRINT

SREDA_Loop2:
    mov dl,es:[bx]
    int 21h
    inc bx

```



```

        cmp byte ptr es:[bx],00h
        jne SREDA_Loop2
    pop es
    ret
SREDA ENDP

BEGIN:
        call UNAVAILABLE_MEM_F
        call ADRESS_GET_F
        call TAIL
        call SREDA
        mov     AH,4Ch
        int     21H
TESTPC
        ENDS
END START

```