

**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**

Студентка гр. 8381

Лисок М.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

### **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.**

При начальной загрузке программы формируется PSP, который размещается в начале первого сегмента программы. PSP занимает 256 байт и располагается с адреса, кратного границе сегмента. При загрузке модулей типа .COM все сегментные регистры указывают на адрес PSP. При загрузке модуля типа .EXE сегментные регистры DS и ES указывают на PSP. Именно по этой причине значения этих регистров в модуле .EXE следует переопределять.

Формат PSP:

Смещение	Длина поля(байт)	Содержимое поля
0	2	int 20h
2	2	Сегментный адрес первого байта недоступной памяти. Программа не должна модифицировать содержимое памяти за этим адресом.
4	6	Зарезервировано
0Ah (10)	4	Вектор прерывания 22h (IP,CS)
0Eh (14)	4	Вектор прерывания 23h (IP,CS)
12h (18)	4	Вектор прерывания 24h (IP,CS)
2Ch (44)	2	Сегментный адрес среды, передаваемой программе.
5Ch		Область форматируется как стандартный неоткрытый блок управления файлом (FCB)
6Ch		Область форматируется как стандартный неоткрытый блок управления файлом (FCB). Перекрывается, если FCB с адреса 5Ch открыт.
80h	1	Число символов в хвосте командной строки.
81h		Хвост командной строки - последовательность символов после имени вызываемого модуля.

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, PROMPT, 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 программы. Маршрут также заканчивается байтом 00h.

### Описание функций и структур данных.

Название функции	Назначение
BYTE_TO_HEX	Переводит число AL в коды символов 16 с/с, записывая получившиеся в AL и AH.
TETR_TO_HEX	Вспомогательная функция для работы BYTE_TO_HEX
WRD_TO_HEX	Переводит число AX в строку в 16 с/с, записывая получившиеся в di, начиная с младшего разряда.
PRINT	Печатает строку на экран

Название	Тип	Назначение
SegAddInMem	db	Строка для хранения адреса сегмента недоступной памяти.
SegAddEnv	db	Строка для хранения адреса сегмента среды окружения.
CommTail	db	Строка с информацией о хвосте командной строки.
NoSymb	db	Строка с информацией о том, что символы в хвосте командной строки отсутствуют.

ContEnv	db	Строка с информацией о содержании среды окружения.
DirectLine	db	Строка с информацией о пути загружаемого модуля.
Endline	db	Конец строки.

### Результат выполнения.

```

C:\>LAB2.COM
Segment address of inaccessible memory: 9FFFh
Segment address of environment: 0188h
There are no characters in the tail of the command line!
The contents of the environment in symbolic form:
PATH=Z:\
COMSPEC=Z:\COMMAND.COM
BLASTER=A220 I7 D1 H5 T6
Path of program:
C:\LAB2.COM
C:\>_

```

Рисунок 1 – результат работы программы lab2.com.

### Выводы.

В ходе работы было проведено исследование интерфейса управляющей программы и загрузочных модулей, а также исследование префикса сегмента программы (PSP) и среды, передаваемой программе.

## **Ответы на контрольные вопросы.**

### **Сегментный адрес недоступной памяти.**

1. На какую область памяти указывает адрес недоступной памяти?

Ответ: адрес недоступной памяти указывает на границу области, доступной для загрузки программ, и границу основной оперативной памяти.

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

Ответ: адрес располагается сразу за памятью, отведённой программе.

3. Можно ли в эту область памяти писать?

Ответ: да, можно, потому что в DOS не предусмотрена защита памяти.

### **Среда, передаваемая программе.**

1. Что такое среда?

Ответ: область памяти, содержащая переменные среды, которые могут использоваться приложениями для получения некоторой системной информации и для передачи данных между программами.

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

Ответ: при загрузке DOS; при запуске программы происходит лишь копирование среды в новую область памяти.

3. Откуда берется информация, записываемая в среду?

Ответ: информация записывается в среду из системного файла autoexec.bat.

# ПРИЛОЖЕНИЕ А

## ИСХОДНЫЙ ТЕКСТ lab2.asm

```

TESTPC SEGMENT
        ASSUME     CS:TESTPC,      DS:TESTPC,      ES:NOTHING,
SS:NOTHING

        ORG 100H

START:   JMP  BEGIN


        SEGADDINMEM      DB          'SEGMENT          ADDRESS          OF
INACCESSIBLE MEMORY:    H', 0DH, 0AH, '$'

        SEGADDENV        DB          'SEGMENT ADDRESS OF ENVIRONMENT:
H', 0DH, 0AH, '$'

        COMMTAIL         DB          'COMMAND LINE TAIL IN SYMBOLIC FORM:
', '$'

        NOSYMB           DB          'THERE ARE NO CHARACTERS IN THE
TAIL OF THE COMMAND LINE!', 0DH, 0AH, '$'

        CONTENV          DB          'THE          CONTENTS          OF          THE
ENVIRONMENT IN SYMBOLIC FORM:', 0DH, 0AH, '$'

        DIRECTLINE       DB          'PATH OF PROGRAM:', 0DH, 0AH,
'$'


        ENDLINE          DB          0DH, 0AH, '$'


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
```

```
    PUSH CX
    MOV     AH,AL
    CALL TETR_TO_HEX
    XCHG AL,AH
    MOV     CL,4
    SHR     AL,CL
    CALL TETR_TO_HEX
    POP     CX
    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
    XOR     AH,AH
    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
                PUSH AX
                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     AX,00H
                JBE     END_L
                OR      AL,30H
                MOV     [SI],AL
END_L:          POP     AX
                POP     DX
                POP     CX
                RET
BYTE_TO_DEC    ENDP

```

```

PRINT    PROC      NEAR
         PUSH      AX
         MOV       AH,09H
         INT       21H
         POP       AX
         RET
PRINT    ENDP

```

```

BEGIN:

```



```

;SEGMENT ADDRESS OF INACCESSIBLE MEMORY
MOV  AX, ES:[02H]
MOV      DI, OFFSET SEGADDINMEM + 43
CALL     WRD_TO_HEX
MOV      DX, OFFSET SEGADDINMEM
CALL     PRINT

;SEGMENT ADDRESS OF ENVIRONMENT
MOV  AX, ES:[2CH]
MOV      DI, OFFSET SEGADDENV + 35
CALL     WRD_TO_HEX
MOV      DX, OFFSET SEGADDENV
CALL     PRINT

;COMMAND LINE TAIL IN SYMBOLIC FORM
SUB  CX, CX
MOV      CL, ES:[80H]
CMP      CL, 0
JE        FIN
LEA      DX, COMMTAIL
CALL PRINT
MOV      AH, 02H
MOV      BX, 0
CYCLE:
      MOV  DL ,ES:[BX+81H]
      INT  21H
      INC  BX
      LOOP CYCLE
      LEA  DX, ENDLINE
      CALL PRINT
      JMP  ENVIR
FIN:
      LEA  DX, NOSYMB
      CALL PRINT

```

```

        ;THE CONTENTS OF THE ENVIRONMENT
ENVIR:
        LEA      DX, CONTENV
        CALL     PRINT
        MOV  AX, ES:[2CH]
        MOV  ES, AX
        MOV      BX, 0
        MOV  AH, 02H

COPY:
        CMP      WORD PTR ES:[BX], 0000H
        JE       END_CE
        CMP      BYTE PTR ES:[BX], 00H
        JNE      PRINT_SYMB
        LEA      DX, ENDLINE
        CALL PRINT
        INC      BX

PRINT_SYMB:
        MOV      DL, ES:[BX]
        INT      21H
        INC      BX
        JMP      COPY

END_CE:
        LEA      DX, ENDLINE
        CALL PRINT

        ;PATH OF PROGRAM
        ADD  BX, 4;
        LEA      DX, DIRECTLINE
        CALL     PRINT
        MOV  AH, 02H

OUT_PATH:
        CMP  BYTE PTR ES:[BX], 00H
        JE       END_PATH

```

```

        MOV  DL, ES:[BX]
        INT  21H
        INC  BX
        JMP  OUT_PATH
END_PATH:
        LEA   DX, ENDLINE
        CALL  PRINT

        ;EXIT IN DOS
        MOV  AX, 4C00H
        INT  21H

TESTPC  ENDS

        END          START

```