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. 9383 _____ Орлов Д.С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

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.

Порядок выполнения работы

Для выполнения лабораторной работы необходимо написать и отладить программный модуль типа .COM, который выбирает и распечатывает следующую информацию:

1. Сегментный адрес недоступной памяти, взятый из PSP, в шестнадцатеричном виде.
2. Сегментный адрес среды, передаваемой программе, в шестнадцатеричном виде.
3. Хвост командной строки в символьном виде.
4. Содержимое области среды в символьном виде.
5. Путь загружаемого модуля.

Выполнение работы

В процессе выполнения лабораторной работы была разработана .COM-программа, в которой:

- MEMORY_S db 'Memory segment: ',0DH,0AH,'$' – сегментный адрес недоступной памяти.
- ENV_ADRESS db 'Environment address: ',0DH,0AH,'$' – сегментный адрес среды.
- TAIL db 'Tail of command line: ','$' – хвост командной строки.
- EMPTY_T db 'Tail of command line: EMPTY','$' – пустой хвост.
- ENV_CONTENT db 0DH,0AH,'Environment scope content:',0DH,0AH, '$' – содержимое области среды.
- PATH db 'Path: ',0DH,0AH, '$' – путь загружаемого модуля.

Рис. 1. Пример работы программы без аргументов

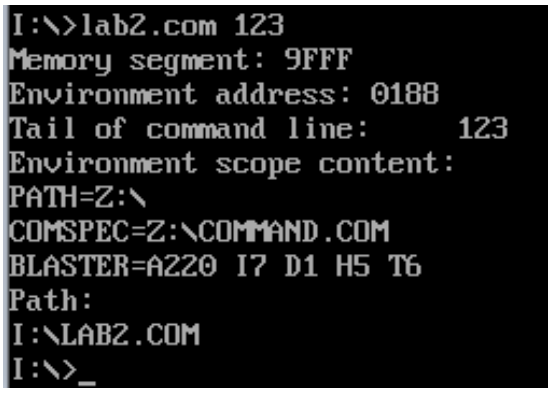

Рис. 2. Пример работы программы с аргументами

Ответы на вопросы

Сегментный адрес недоступной памяти

1. На какую область памяти указывает адрес недоступной памяти?

Адрес недоступной памяти указывает на основную оперативную память, то есть на первый байт после памяти, выделенной программе.

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

Адрес расположен в сегменте PSP со смещением 2Ch.

3. Можно ли в эту область памяти писать?

Можно, т. к. DOS не имеет механизмов защиты от перезаписи памяти различными программами, которая им не отведена.

Среда, передаваемая программе

1. Что такое среда?

Среда — область памяти, хранящая информацию о состоянии системы. Например, настройки, путь к домашней директории компьютера и т.д.

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

Изначальная среда создается при запуске ОС. Эта среда копируется (и после чего может измениться) в адресное пространство запущенной программы.

3. Откуда берется информация, записываемая в среду?

Информация берется из системного файла AUTOEXEC.BAT, который расположен в корневом каталоге загрузочного устройства.

Вывод.

В ходе лабораторной работы были получены навыки работы с PSP, а также изучены интерфейсы программных модулей DOS.

Приложение А

```
TESTPC SEGMENT
  ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
  ORG 100H


START: JMP BEGIN

; Данные
MEMORY_S db  'Memory segment:     ',0DH,0AH,'$'
ENV_ADRESS db 'Environment address:    ',0DH,0AH,'$'
TAIL db 'Tail of command line:    ','$'
EMPTY_T db 'Tail of command line: EMPTY','$'
ENV_CONTENT db 0DH,0AH,'Environment scope content:',0DH,0AH, '$'
END_STRING db 0DH,0AH, '$'
PATH db 'Path:  ',0DH,0AH, '$'


; Процедуры
;-----------------------------------------------------
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 шест. 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
;перевод в 16 с/с 16-ти разрядного числа
; в AX - число, DI - адрес последнего символа
    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
;--------------------------------

MEMORY_DEFINITION PROC near
    mov ax, ds:[02h]

    mov di, offset MEMORY_S
    add di, 19
```

```
    call WRD_TO_HEX
    mov dx, offset MEMORY_S
    mov AH,09h
    int 21h
    ret
MEMORY_DEFINITION ENDP


ENV_ADRESS_DEFINITION PROC near
    mov ax, ds:[2Ch]

    mov di, offset ENV_ADRESS
    add di, 24
    call WRD_TO_HEX
    mov dx, offset ENV_ADRESS
    mov AH,09h
    int 21h
    ret

ENV_ADRESS_DEFINITION ENDP


TAIL_DEFINITION PROC near
    mov cx, 0
    mov cl, ds:[80h]
    cmp cl, 0          ;если пусто
    je empty_tail

          mov dx, offset TAIL
    mov AH,09h
    int 21h


    mov di, 0
    mov ax, 0
```

```
read_tail:
    mov dl, ds:[81h+di]
    mov ah, 02h
    int 21h
    inc di
    loop read_tail



ret

empty_tail:
    mov dx, offset EMPTY_T
          mov AH,09h
    int 21h

ret

TAIL_DEFINITION ENDP


CONTENT_DEFINITION PROC near

    mov dx, offset ENV_CONTENT
    mov AH,09h
    int 21h
    mov di, 0
    mov ds, ds:[2Ch]

read_str:
    cmp byte ptr [di], 0
    je end_str
    mov dl, [di]
    mov ah, 02h
```

```
    int 21h
    jmp find_end

end_str:
    cmp byte ptr [di+1],00h
    je find_end
    push ds
    mov cx, cs
    mov ds, cx
    mov dx, offset END_STRING
    mov AH,09h
    int 21h
    pop ds

find_end:
    inc di
    cmp word ptr [di], 0001h
    je read_path
    jmp read_str

read_path:
    push ds
    mov ax, cs
    mov ds, ax

    mov dx, offset PATH
    mov AH,09h
    int 21h

    pop ds
    add di, 2

loop2:
    cmp byte ptr [di], 0
    je break
```

```
    mov dl, [di]
    mov ah, 02h
    int 21h
    inc di
    jmp loop2

break:
    ret

CONTENT_DEFINITION ENDP

; Код
BEGIN:
    call MEMORY_DEFINITION
    call ENV_ADRESS_DEFINITION
    call TAIL_DEFINITION
    call CONTENT_DEFINITION

    xor AL,AL
    mov AH,4Ch
    int 21H
TESTPC ENDS
END START; конец модуля, START - точка выхода
```