**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Операционные системы»**

Тема: Исследование интерфейсов программных модулей

|  |  |  |
| --- | --- | --- |
| Студент гр. 6383 |  | Медведев Г.О. |
| Преподаватель |  | Губкин А.Ф. |

Санкт-Петербург

2018

**Цель работы.**

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

**Описание функций и структур данных.**

|  |  |
| --- | --- |
| **Название функции** | **Назначение** |
| MEMORY\_ | печатает адрес недоступной памяти |
| SEGMENT\_ | печатает сегментный адрес среды |
| SPACE | перенос содержимого на новую строку |
| PRINT | вызывает функцию печати строки |
| TAIL\_ | печатает хвост командной строки |
| PATH\_ | печатает путь загружаемого модуля |
| BYTE\_TO\_HEX | переводит число AL в коды символов 16-ой с/с, записывая получившееся в al и ah |
| TETR\_TO\_HEX | вспомогательная функция для работы функции BYTE\_TO\_HEX |
| WRD\_TO\_HEX | переводит число AX в строку в 16-ой с/с, записывая получившееся в di, начиная с младшей цифры |
| BYTE\_TO\_DEC | переводит байт из AL в десятичную с/с и записывает получившееся число по адресу si, начиная с младшей цифры |

**Последовательность действий, выполняемых утилитой.**

1. Печатает сегментный адрес первого байта недоступной памяти
2. Печатает сегментный адрес среды, передаваемой программе
3. Печатает хвост командной строки
4. Печатает содержимое области среды в символьном виде
5. Печатает путь загружаемого модуля
6. Выходит в DOS

Результат выполнения программы lab\_2.com представлен на Рисунке 1.

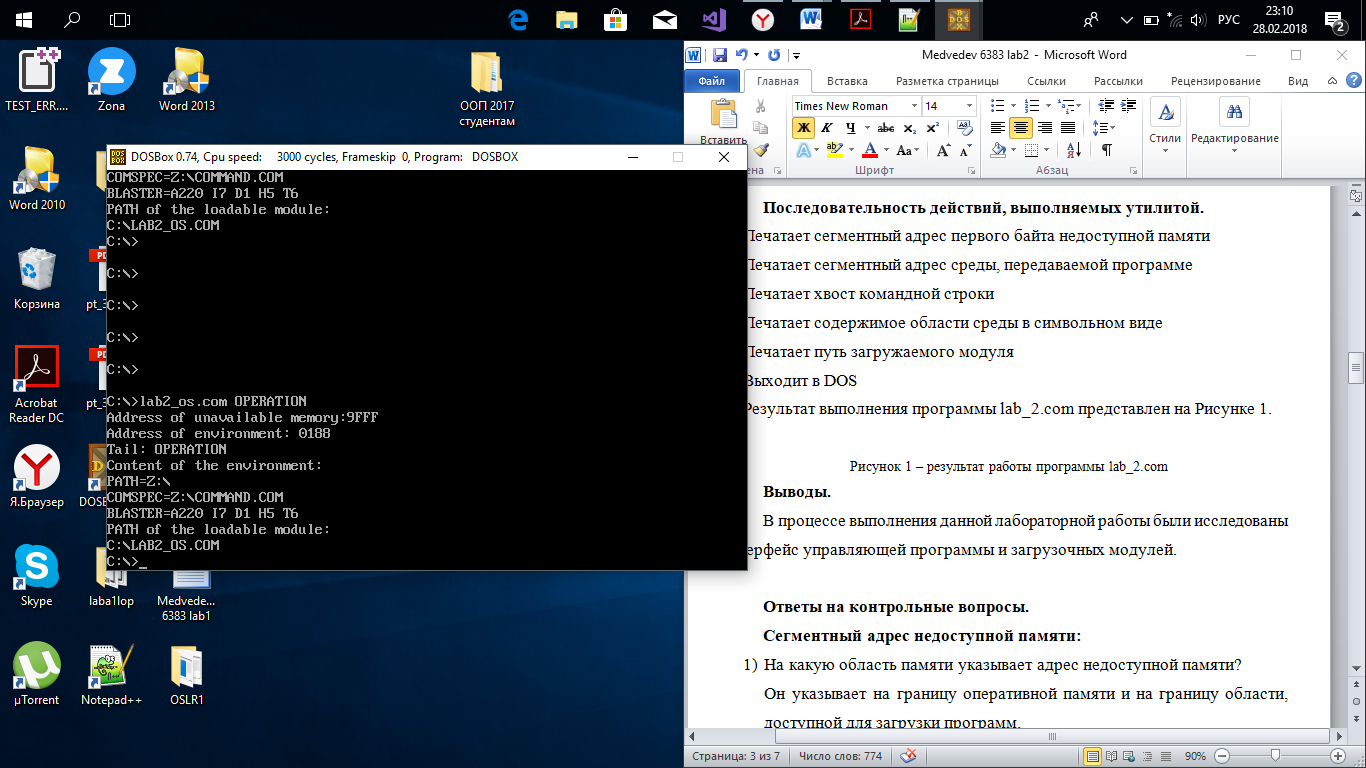


Рисунок 1 – Результат выполнения программы lab2\_os.com

**Выводы.**

В процессе выполнения данной лабораторной работы были исследованы интерфейс управляющей программы и загрузочных модулей.

**Ответы на контрольные вопросы.**

Сегментный адрес недоступной памяти:

1. На какую область памяти указывает адрес недоступной памяти?

Адрес недоступной памяти указывает на конец памяти, которая выделяется под программу, затем следует «старшая память», включающая в себя графический видеобуфер, свободные адреса, текстовый буфер и ПЗУ-расширения BIOS.

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

По отношению области памяти данный адрес расположен сразу за концом памяти, которая отведена программе, начиная с 9FFFh.

1. Можно ли в эту область памяти писать?

В эту область памяти можно писать, так как в MS-DOS отсутствует защита памяти.

Среда, передаваемая программе:

1. Что такое среда?

Среда представляет собой область памяти, в которой в виде символьных строк записаны значения переменных, называемых переменными среды.

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

Среда создается при загрузке DOS, при запуске приложения копируется в новую область памяти.

1. Откуда берется информация, записываемая в среду?

Информация берется из системного файла autoexec.bat.

ПРИЛОЖЕНИЕ А

LAB2\_OS.ASM

TESTPC SEGMENT

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

ORG 100H

START: JMP BEGIN

ADDofMEM db 'Address of unavailable memory: ',0dh,0ah,'$'

ADDofENV db 'Address of environment: ',0dh,0ah,'$'

TAIL db 'Tail:','$'

CONTofENV db 'Content of the environment: ' , '$'

PATHofMOD db 'PATH of the loadable module: ' , '$'

ENDL db 0dh,0ah,'$'

SPACE PROC near

lea dx,ENDL

call PRINT

ret

SPACE ENDP

PRINT PROC near

mov ah,09h

int 21h

ret

PRINT ENDP

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

call BYTE\_TO\_HEX

mov [di],ah

dec di

mov [di],al

pop bx

ret

WRD\_TO\_HEX ENDP

; segment address of unavailable memory

MEMORY\_ PROC near

push ax

mov ax,es:[2]

lea di,ADDofMEM

add di,33

call WRD\_TO\_HEX

pop ax

ret

MEMORY\_ ENDP

; segment address of environment

SEGMENT\_ PROC near

push ax

mov ax,es:[2Ch]

lea di,ADDofENV

add di,27

call WRD\_TO\_HEX

pop ax

ret

SEGMENT\_ ENDP

; finding tail

TAIL\_ PROC near

push ax

push cx

xor ax, ax

mov al, es:[80h]

add al, 81h

mov si, ax

push es:[si]

mov byte ptr es:[si+1], '$'

push ds

mov cx, es

mov ds, cx

mov dx, 81h

call PRINT

pop ds

pop es:[si]

pop cx

pop ax

ret

TAIL\_ ENDP

; path of module

PATH\_ PROC near

push es

push ax

push bx

push cx

mov bx,1

mov es,es:[2ch]

mov si,0

POINT:

call SPACE

mov ax,si

POINT\_:

cmp byte ptr es:[si], 0

je POINT1

inc si

jmp POINT\_

POINT1:

push es:[si]

mov byte ptr es:[si], '$'

push ds

mov cx,es

mov ds,cx

mov dx,ax

call PRINT

pop ds

pop es:[si]

cmp bx,0

jz POINT1\_

inc si

cmp byte ptr es:[si], 01h

jne POINT

lea dx,PATHofMOD

call PRINT

mov bx,0

add si,2

jmp POINT

POINT1\_:

pop cx

pop bx

pop ax

pop es

ret

PATH\_ ENDP

;----------------------------

Write PROC near

mov ah,09h

int 21h

ret

Write ENDP

BEGIN:

call MEMORY\_

call SEGMENT\_

lea dx,ADDofMEM

call PRINT

lea dx,ADDofENV

call PRINT

lea dx, TAIL

call PRINT

call TAIL\_

call SPACE

lea dx,CONTofENV

call PRINT

call PATH\_

xor al,al

mov ah, 04Ch

int 21h

ret

TESTPC ENDS

END START