**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. 8383 |  | Бабенко Н.С. |
| 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.

**Сведения о функциях и структурах данных управляющей программ**ы.

1) PRINT – процедура печати, вызывает функцию 09h. Выводит содержимое сегмента DS со смещением из регистра DX.

2) TETRTOHEX – процедура вывода байта AL в 16-ричной системе счисления.

3) BYTETOHEX – процедура перевода байта в регистре AL в два символа шестнадцатеричного числа в AX.

4) WRDTOHEX – процедура перевода в 16-ричную систему счисления 16-ти разрядного числа в регистре AX. Из регистра DI берется адрес последнего символа.

5) BYTETODEC – процедура перевода в 10-тичную систему счисления байта в регистре AL. Из регистра SI берется адрес поля младшей цифры.

6) PRINTMEMORYADR - Процедура печати сегментного адреса недоступной памяти, взятой из PSP

7) PRINTENVIRONMENTADR - Процедура печати сегментного адреса среды, передаваемой программе

8) PRINTCOMMANDSTRINGTAIL - Процедура печати хвоста командной строки в символьном виде

9) PRINTENVIRONMENTAREA - Процедура печати содержимого области среды в символьном виде.

10) PRINTPATHMODULE - Процедура пути загружаемого модуля

**Последовательность действий, выполняемых утилитой.**

Программа по формату PSP выводит сегментный адрес недоступной памяти в шестнадцатеричном виде, сегментный адрес среды, передаваемой программе в шестнадцатеричном виде, хвост командной троки в символьном виде, содержимое области среды в символьном виде и путь загружаемого модуля

**Ход работы.**

1) Написан текст исходного .COM модуля.

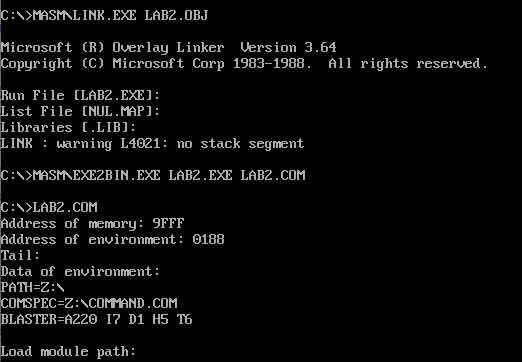


Рисунок 1 – Результат выполнения **.**COM модуля

2) Были получены ответы на контрольные вопросы:

**Сегментный адрес недоступной памяти**

1. Адрес недоступной памяти указывает на сегментный адрес последнего параграфа памяти, который использует DOS, чтобы запустить эту программу.
2. Адрес расположен за областью памяти, которая отводится для программы (первый байт сразу за областью).
3. В эту область памяти можно писать, так как не идет контроль со стороны системы тех адресов, с которыми эта программа обращается, в виду того, что DOS является однозадачной операционной системой

**Среда, передаваемая программе:**

1) Среда представляет собой область памяти, в которой в виде символьных строк вида «параметр = значение» записаны значения переменных, называемых переменными среды. Среда начинается с границы параграфа, после последней строки идет нулевой байт

2) Среда создается при загрузке ОС, но перед запуском приложения она может быть изменена в соответствии с требованиями этого приложения.

3) При загрузке программы содержимое начальной среды копируется в создаваемую среду программы, которая таким образом имеет доступ как к системным переменным, так и к переменным, включённым в среду пользователем с помощью команды SET.

**Выводы.**

Был исследован интерфейс 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. Также исследован префикс сегмента программы (PSP) и среды, передаваемой программе.

**ПРИЛОЖЕНИЕ A**

Исходный код **lab2.asm**

LAB2 SEGMENT

ASSUME CS:LAB2, DS:LAB2, ES:NOTHING, SS:NOTHING

ORG 100H

START: JMP BEGIN

ADRESSMEMORY DB "Address of memory: $"

ADRESSENVIRONMENT DB 13, 10, "Address of environment: $"

TAILSTR DB 13, 10, "Tail: $"

ENVIRONMENTDATA DB 13, 10, "Data of environment: ", 13, 10, "$"

PATHSTR DB "Load module path: ", 13, 10, "$"

NEXTSTR DB 13, 10, "$"

TETRTOHEX PROC NEAR

AND AL, 0FH

CMP AL, 09H

JBE NEXT

ADD AL, 07H

NEXT:

ADD AL, 30H

RET

TETRTOHEX ENDP

BYTETOHEX PROC NEAR

PUSH CX

MOV AH, AL

CALL TETRTOHEX

XCHG AL, AH

MOV CL, 4H

SHR AL, CL

CALL TETRTOHEX

POP CX

RET

BYTETOHEX ENDP

WRDTOHEX PROC NEAR

PUSH BX

MOV BH, AH

CALL BYTETOHEX

MOV [DI], AH

DEC DI

MOV [DI], AL

DEC DI

MOV AL, BH

CALL BYTETOHEX

MOV [DI], AH

DEC DI

MOV [DI], AL

POP BX

RET

WRDTOHEX ENDP

BYTETODEC PROC NEAR

PUSH CX

PUSH DX

XOR AH, AH

XOR DX, DX

MOV CX, 0AH

LOOP\_BD:

DIV CX

OR DL, 30H

MOV [SI], DL

DEC SI

XOR DX, DX

CMP AX, 0AH

JAE LOOP\_BD

CMP AL, 00H

JE END\_L

OR AL, 30H

MOV [SI], AL

END\_L:

POP DX

POP CX

RET

BYTETODEC ENDP

PRINT PROC NEAR

PUSH AX

MOV AH, 09H

INT 21H

POP AX

RET

PRINT ENDP

PRINTMEMORYADR PROC NEAR

MOV AX, DS:[02H]

MOV DI, OFFSET ADRESSMEMORY

ADD DI, 16H

CALL WRDTOHEX

MOV DX, OFFSET ADRESSMEMORY

CALL PRINT

RET

PRINTMEMORYADR ENDP

PRINTENVIRONMENTADR PROC NEAR

MOV AX, DS:[2CH]

MOV DI, OFFSET ADRESSENVIRONMENT

ADD DI, 1DH

CALL WRDTOHEX

MOV DX, OFFSET ADRESSENVIRONMENT

CALL PRINT

RET

PRINTENVIRONMENTADR ENDP

PRINTCOMMANDSTRINGTAIL PROC NEAR

MOV SI, 1

MOV CX, 0

MOV CL, DS:[80H]

CMP CL, 0

JZ EMPTY

MOV DI, OFFSET TAILSTR

ADD DI, 8H

CYCLE:

MOV AL, DS:[80H + SI]

MOV [DI], AL

INC DI

INC SI

LOOP CYCLE

EMPTY:

MOV DX, OFFSET TAILSTR

CALL PRINT

RET

PRINTCOMMANDSTRINGTAIL ENDP

PRINTENVIRONMENTAREA PROC NEAR

MOV DX, OFFSET ENVIRONMENTDATA

CALL PRINT

MOV BX, 2CH

MOV DS, [BX]

MOV DI, 0

STARTofSTR:

CMP BYTE PTR [DI], 00H

JZ PRINTNEXTSTR

MOV DL, [DI]

MOV AH, 02H

INT 21H

JMP ENVIRONMENTEND

PRINTNEXTSTR:

PUSH DS

MOV CX, CS

MOV DS, CX

MOV DX, OFFSET NEXTSTR

CALL PRINT

POP DS

ENVIRONMENTEND:

INC DI

CMP WORD PTR [DI], 0001H

JZ RETURN

JMP STARTofSTR

RET

RETURN:

RET

PRINTENVIRONMENTAREA ENDP

PRINTPATHMODULE PROC NEAR

PUSH DS

MOV AX, CS

MOV DS, AX

MOV DX, OFFSET PATHSTR

CALL PRINT

POP DS

ADD DI, 2

CIRCLE:

CMP BYTE PTR [DI], 00H

JZ PATHFINISH

MOV DL, [DI]

MOV AH, 02H

INT 21H

INC DI

JMP CIRCLE

PATHFINISH:

RET

PRINTPATHMODULE ENDP

BEGIN:

CALL PRINTMEMORYADR

CALL PRINTENVIRONMENTADR

CALL PRINTCOMMANDSTRINGTAIL

CALL PRINTENVIRONMENTAREA

CALL PRINTPATHMODULE

MOV AH, 4CH

INT 21H

LAB2 ENDS

END START