

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. 7381

Кортев Ю.В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2019

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.

Тип IBM PC узнается путем считывания предпоследнего байта с ROM BIOS. Его значение сравнивается с кодами таблицы, представленной ниже.

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
0Eh(14)	4	Вектор прерывания 23h
12h(18)	4	Вектор прерывания 24h
2Ch(44)	2	Сегментный адрес среды, передаваемой программе.
5Ch		Область форматируется как стандартный неоткрытый блок управления файлом
6Ch		Область форматируется как стандартный неоткрытый блок управления файлом
80h	1	Число символов в хвосте командной строки
81h		Хвост командной строки — последовательность символов после имени вызываемого модуля

Таблица 1 – Формат PSP

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

Все функции расположены в табл. 1, структуры данных – в табл.2

Выполнение работы

1. Получаю сегментный адрес недоступной памяти и печатаю.
2. Получую сегментный адрес среды и печатаю.
3. Вывожу хвост командной строки в символьном виде или сообщение о том, что его нет.
4. Вывожу содержимое области среды и путь загружаемого модуля в символьном виде.

Результат работы программы показан на рис. 1.



```
C:\LAB2>lab2.com hello, yura kortev
Unavailable mem adr: 9FFF
Segm Envir Adr: 0188
Tail: hello, yura kortev

Envir content:
PATH=Z:\
COMSPEC=Z:\COMMAND.COM
BLASTER=A220 I7 D1 H5 T6

Programs path:C:\LAB2\LAB2.COM
```

Рисунок 1 – Результат работы программы

Вывод

В ходе выполнения работы было проведено изучение интерфейса управляющей программы и загрузочных модулей, PSP и параметров среды, передаваемой программе при её запуске.

Ответы на контрольные вопросы

Сегментный адрес недоступной памяти программ

1. На какую область памяти указывает адрес недоступной памяти?

Ответ: Адрес недоступной памяти указывает на область памяти, работа с которой запрещена (ведёт к непредсказуемому поведению).

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

Ответ: Этот адрес является первым сразу за концом сегмента памяти, отведённой программе, – в нашем случае этим адресом является 9FFF.

3. Можно ли в эту область памяти писать?

Ответ: При условии отсутствия в управляющей программе операционной системы механизма защиты памяти, запись в эту область возможна.

Среда, передаваемая программе:

1. Что такое среда?

Ответ: Среда – это набор переменных, хранящих информацию о конфигурации и настройках системы, в которой запускается приложение.

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

Ответ: Во время запуска ОС.

3. Откуда берется информация, записываемая в среду?

Ответ: Эта информация хранится в файле системного реестра. В случае операционной системы MS DOS эта информация берётся из файла AUTOEXEC.BAT.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД

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

unavailableMemAdr
  db 'Unavailable
    mem adr: $'
UnavMemAdr db '
  $'
EnvirAdr db 'Segm
  Envir Adr: $'
EnvAdr db '  $'
PrintTail db 'Tail:$'
TAIL db 50h DUP('
  '), '$'
NoTail db 'No Tail$'
EnvCont db 'Envir
  content:', 0DH, 0AH,
  '$'
ModulePath db
  'Programs path:', '$'
ENDL 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
;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
; 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

```

```

;-----

```

```

-----

```

```

BYTE_TO_DEC
PROC near
; перевод в 10с/с,
SI - адрес поля
младшей цифры
push AX
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
pop AX
ret
BYTE_TO_DEC
ENDP
;-----
----
```

```

Print PROC near
mov AH,09h
int 21h
ret
Print ENDP
```

```

Unavailable_memory_adress PROC
near
mov ax,ds:[2]
mov es,ax
mov di,offset
UnavMemAdr+3
```

```

call
WRD_TO_HEX
mov dx,offset
unavailableMemoryAddress
r
call Print
mov dx,offset
UnavMemAdr
call Print
mov dx,offset
ENDL
call Print
ret
Unavailable_memory_adress ENDP
```

Environment_adress

PROC near
mov ax,ds:[2Ch]
mov di,offset
EnvAdr+3

call
WRD_TO_HEX
mov dx,offset
EnvirAdr
call Print
mov dx,offset
EnvAdr
call Print
mov dx,offset
ENDL
call Print
ret

Environment_adress

ENDP

Print_tail PROC

near ; хвост
командной строки
в символьном
виде
xor ch,ch
mov cl,ds:[80h] ;
число символов в
хвосте командной
строки

cmp cl,0
;если 0-нет
хвоста
jne case_tail
mov dx,offset
NoTail
call Print
mov dx,offset
ENDL
call Print
ret
case_tail:

mov dx,offset
PrintTail
call Print

mov bp,offset TAIL
cycle:
mov di,cx
mov bl,ds:[di+80h]


```

mov ds:[bp+di-1],bl
loop cycle

mov dx,offset TAIL
call Print
mov dx, offset
ENDL
call Print
ret
Print_tail ENDP

Print_environment
PROC near
mov dx, offset
EnvCont
call Print

mov ax,ds:[2ch]
mov es,ax

xor bp,bp
cycle1:
cmp word ptr
es:[bp],0001h
je case_exit1
cmp byte ptr
es:[bp],00h
jne noendl
mov dx,offset
ENDL
call Print
inc bp
noendl:
mov
dl,es:[bp]
mov
ah,02h
int 21h
inc bp
jmp
cycle1
case_exit1:
add bp,2

mov dx, offset
ENDL
call Print
mov dx, offset
ModulePath
call Print

cycle2:
cmp byte ptr

```

```

    es:[bp],00h
    je case_exit2
    mov dl,es:[bp]
    mov ah,02h
    int 21h
    inc bp
    jmp cycle2
case_exit2:

    ret
Print_environment
    ENDP

BEGIN:
    call
    Unavailable_mem
    ory_address
    call
    Environment_adre
    ss
    call Print_tail
    call
    Print_environment
    xor AL,AL ;|
    mov AH,4Ch ;| exit
    to dos
    int 21H ;|
TESTPC ENDS
END START

```