

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

Студентка гр. 9382

Сорочина М.В.

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.

Сведения, использованные для составления программы.

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 регистров в модуле .EXE следует переопределить

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	4	Вектор прерывания 22h (IP, CS)
0eh	4	Вектор прерывания 23h (IP, CS)

12h	4	Вектор прерывания 24h (IP, CS)
2ch	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 - послед-ть символов после имени вызываемого модуля

Табл. 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, 00h, 01h, после которых располагается маршрут загруженной программы. Маршрут также заканчивается байтом 00h.

Ход работы.

Был написан и отлажен программный модуль типа .COM, который выводит на экран 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

В итоге были получены данные, представленные на рис. 1:



```
C:\>2.COM
Segment address of memory:9FFF
Segment address of environment:0188
Command line tail:
No tail
Environment content:
PATH=Z:\
COMSPEC=Z:\COMMAND.COM
BLASTER=A220 I7 D1 H5 T6
Loadable module path:
C:\2.COM
```

рис. 1. Результат выполнения программы

Ответы на контрольные вопросы.

Сегментный адрес недоступной памяти.

1. На какую область памяти указывает адрес недоступной памяти?
9FFFh - FFFFh - область недоступной памяти
2. Где расположен этот адрес по отношению области памяти, отведенной программе?
После области памяти, выделенной для программы.
3. Можно ли в эту область памяти писать?
Можно, тк в DOS не предусмотрено защиты памяти.

Среда передаваемая программе.

1. Что такое среда?

Среда - область памяти, в которой в виде символьных строк хранятся значения переменных среды в формате “имя=значение” и байт нулей.

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

При загрузке ОС.

3. Откуда берется информация, записываемая в среду?

Из AUTOEXEC.BAT - файла, который содержит информацию о переменных среды.

Выводы.

В ходе выполнения данной работы был исследован интерфейс управляющей программы и загрузочных модулей.

ПРИЛОЖЕНИЕ А.

Исходный код программы.

TESTPC SEGMENT

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

ORG 100H

START: JMP BEGIN

; Данные

SegAddrMem db 'Segment address of memory: ',0DH,0AH,'\$'

SegAddrEnv db 'Segment address of environment: ',0DH,0AH,'\$'

Tail db 'Command line tail: ',0DH,0AH,'\$'

NoTail db 'No tail',0DH,0AH,'\$'

TailInfo db ' '\$'

EnvContent db 'Environment content: ',0DH,0AH,'\$'

NewLine db 0DH,0AH,'\$'

Path db 'Loadable module 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
;-----
BYTE_TO_DEC PROC near
; перевод в 10с/с, SI - адрес поля младшей цифры
        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_I
        or AL,30h
        mov [SI],AL
end_I:

        pop DX
        pop CX
        ret
BYTE_TO_DEC ENDP
;-----
PRINT proc    near
            mov    ah, 09h
            int 21h
            ret
PRINT endp
;-----
; Код
BEGIN:

;Сегментный адрес недоступной памяти
        mov    ax, ds:[02h]
        mov    di, offset SegAddrMem
        add    di, 29
        call wrd_to_hex
        mov dx, offset SegAddrMem
        call print

;Сегментный адрес среды
        mov    ax, ds:[2ch]
        mov    di, offset SegAddrEnv
        add    di, 34
        call wrd_to_hex
        mov    dx, offset SegAddrEnv
        call    print

```


;Хвост командной строки

```
mov    dx, offset tail
call print
mov    cl, ds:[80h]
cmp    cl, 0
je     empty
;xor    ch, ch
xor    si, si
;xor    ax, ax
```

tailLoop:

```
mov    dl, ds:[81h+si]
mov    ah, 02h
int    21h
inc    si
dec    cl
cmp    cl, 0
jne    tailloop
mov    dx, offset newline
call print
jmp    envcon
```

empty:

```
mov    dx, offset notail
call    print
```

envcon:

;Содержимое области среды

```
mov    dx, offset envcontent
call print
mov    es, ds:[2ch]
xor    si, si
```

printStr:

```
mov    al, es:[si]
cmp    al, 0
```

```

        jne    printSymbol
        inc si
        mov    al, es:[si]
        mov    dx, offset newline
        call print
printSymbol:
        mov    dl, al
        mov    ah, 02h
        int    21h
        inc    si
        mov    ax, es:[si]
        cmp    ax, 0001
        jne    printStr

```

;Путь загружаемого модуля

```

        mov    dx, offset path
        call print
        add    si, 2
printSymb:
        mov    al, es:[si]
        cmp    al, 0
        je     exit
        mov    dl, al
        mov    ah, 02h
        int    21h
        inc    si
        jmp    printsymb

```

exit: ; Выход в DOS

```

        xor    AL,AL
        mov    AH,4Ch
        int    21H

```

TESTPC ENDS

END START