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. 8382 _____ Чирков С.А.

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 и среды, передаваемой программе.

Выполнение работы.

В ходе работы был написан и отлажен 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.

Результат работы программы представлен на рисунке 1.

```
C:\>lr2.com ready
Address of inaccessible memory is 9FFF
Address of program environment is 0188
Tail of command line is ready
Environment data:  PATH=Z:\ COMSPEC=Z:\COMMAND.COM BLASTER=A220 I7 D1 H5 T6
Path of file: C:\LR2.COM
```

Рисунок 1. Результат работы программы

Контрольные вопросы.

Сегментный адрес недоступной памяти

1. На какую область памяти указывает адрес недоступной памяти?

   От 9FFFh до FFFFh. Адрес 9FFFh, находящийся в PSP, подразумевает начало памяти, в которую нельзя загрузить программу.

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

   Сразу же после памяти, выделенной программе

3. Можно ли в эту область памяти писать?

   Да, так как DOS не контролирует обращение программ к памяти, но это может отразиться на работе программы

Среда передаваемая программе

1. Что такое среда?

   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.

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

   Среда создается при загрузке операционной системы.

3. Откуда берется информация, записываемая в среду?

   Среда копируется из содержимого среды родительского процесса. Также можно изменить ее переменные или просмотреть ее содержимое с помощью команды set.

Выводы.

В ходе работы был исследован интерфейс управляющей программы, который состоит в передаче запускаемой программе управляющего блока, содержащего адреса и системные данные, например, префикс сегмента программы, среда, передаваемая программе.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ LR2.ASM

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

inaccmem db 'Address of inaccessible memory is $'
addenv db 'Address of program environment is $'
linetail db 'Tail of command line is$'
dataenv db 'Environment data:  $'
path db 'Path of file: $'
endl db 10,13,'$'
space db ' $'

PRINT PROC near
      push CX
      push DX

      call BYTE_TO_HEX
      mov CH, AH

      mov DL, AL
      mov AH, 02h
      int 21h

      mov DL, CH
      mov AH, 02h
      int 21h

      pop DX
      pop CX
      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

BEGIN:
      mov dx,offset inaccmem
      mov ah,09h
      int 21h
      mov ax,ds:[03h]
      call print
      mov ax,ds:[02h]
      call print
      mov dx,offset endl
      mov ah,09h
      int 21h

      mov dx,offset addenv
      mov ah,09h
      int 21h
      mov ax,ds:[2Dh]
      call print
      mov ax,ds:[2Ch]
      call print
      mov dx,offset endl
      mov ah,09h
      int 21h

      mov dx, offset linetail
      mov ah,09h
      int 21h
      mov dl, ds:[80h]
      mov bx, 0
looptail:
```

```
      cmp dl, 0
      je endtail
      mov al, ds:[81h+bx]
      push dx
      push ax
      mov dx, ax
      mov ah, 02h
      int 21h
      pop ax
      pop dx
      inc bx
      dec dl
      jmp looptail
endtail:
      mov dx, offset endl
      mov ah,09h
      int 21h

      mov dx, offset dataenv
      mov ah,09h
      int 21h
      mov ss, ds:[2Ch]
      mov bx, 0
loopenv:
      mov al, ss:[bx]
      cmp al, 0
      je islast
      push dx
      push ax
      mov dx, ax
      mov ah, 02h
      int 21h
      pop ax
      pop dx
      inc bx
      jmp loopenv
islast:
      mov al, ss:[bx+2]
      cmp al, 0
      je endenv
      inc bx
      mov dx, offset space
      mov ah,09h
```

```
      int 21h
      jmp loopenv
endenv:
      mov dx, offset endl
      mov ah,09h
      int 21h

      mov dx, offset path
      mov ah,09h
      int 21h
      add bx, 3
looppath:
      mov al, ss:[bx]
      cmp al, 0h
      je endpath
      push dx
      push ax
      mov dx, ax
      mov ah, 02h
      int 21h
      pop ax
      pop dx
      inc bx
      jmp looppath
endpath:
      xor AL,AL
      mov AH,4Ch
      int 21H

TESTPC ENDS
      END START;
```