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 структур загрузочных модулей

Студент гр. 9381 _____ Гурин С.Н.

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.

Порядок выполнения работы.

Для выполнения лабораторной работы необходимо написать и отладить 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.

Сохраните результаты, полученные программой, и включите их в отчет.

Выполнение работы.

Была реализована программа, которая выводит сегментный адрес недоступной памяти, сегментный адрес среды, передаваемой программе, выводится хвост командной строки, записанный в отдельной строке, содержимое области среды посимвольно, а так же путь загружаемого модуля. Примеры выполнения программы:

Рис. 1

Рис. 2

Ответы на контрольные вопросы:

Сегментный адрес недоступной памяти

1) На какую область памяти указывает адрес недоступной памяти?

Адрес недоступной памяти указывает на область основной оперативной памяти.

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

Этот адрес расположен за областью памяти, отведенной программе.

3) Можно ли в эту область памяти писать?

В эту область память можно писать с помощью адресацию для сегментного регистра.

Среда передаваемая программе

1) Что такое среда?

Среда – область памяти, содержащая значения системных переменных, путей и другие данные операционной системы.

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

Среда создается при запуске операционной системы.

3) Откуда берется информация, записанная в среду?

Информация, записываемая в среду берется из файла autoexec.bat, расположенный в корневом каталоге загрузочного модуля.

Вывод

При выполнении данной лабораторной работы был исследован интерфейс управляющей программы и загрузочных модулей

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab2.asm

```
TESTPC    SEGMENT

          ASSUME    CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
          ORG  100H
START:    JMP  BEGIN
; ДАННЫЕ
SEG_AD_UN_MEM   DB        'SEGMENT ADDRESS OF THE UNVAILIBLE MEMORY:
H',0DH,0AH,'$'
SEG_AD_ENV          DB        'SEGMENT ADDRESS OF THE ENVIRONMENT:
H',0DH,0AH,'$'
TAIL_OF_COM_STR DB        'TAIL OF THE COMMAND STRING:',0DH,0AH,'$'
NO_TAIL         DB        'TAIL IS EMPTY',0DH,0AH,'$'
NEW_STR         DB        0DH,0AH,'$'
TAIL            DB        '
',0DH,0AH,'$'
CONT_ENV_AREA   DB        'CONTENT    OF    THE    ENVIRONMENT    AREA:
',0DH,0AH,'$'
PATH            DB        'PATH OF THE LOADED MODULE:',0DH,0AH,'$'
;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 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 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
                    RET
BYTE_TO_DEC     ENDP
;--------------------------------
; КОД

PRINT_INF       PROC  NEAR
          SEG_MEMORY:
                    MOV   AX, DS:[02H]
                    MOV   DI, OFFSET SEG_AD_UN_MEM
                    ADD   DI, 45
                    CALL  WRD_TO_HEX
                    MOV   DX, OFFSET SEG_AD_UN_MEM

                    MOV   AH, 09H
                    INT   21H


          SEG_ENVIRONMENT:
                    MOV   AX, DS:[2CH]
                    MOV   DI, OFFSET SEG_AD_ENV
                    ADD   DI, 39
                    CALL  WRD_TO_HEX
                    MOV   DX, OFFSET SEG_AD_ENV

                    MOV   AH, 09H
                    INT   21H
```

```
TAIL_COM:
          MOV   DX, OFFSET TAIL_OF_COM_STR

          MOV   AH, 09H
          INT   21H


          SUB   CX, CX
          SUB   AX, AX
          SUB   DI, DI
          MOV   CL, DS:[80H]
          MOV   SI, OFFSET TAIL
          CMP   CL, 0
          JE         IF_ZERO
STRING_LOOP:;CX = CX - 1
          MOV   AL, DS:[81H + DI]
          INC   DI
          MOV   [SI], AL
          INC   SI
LOOP STRING_LOOP
          MOV   DX, OFFSET TAIL

          MOV   AH, 09H
          INT   21H

          JMP   CONTENT_OF_ENVIRONMENT

IF_ZERO:
          MOV        DX, OFFSET NO_TAIL

          MOV   AH, 09H
          INT   21H

          JMP   CONTENT_OF_ENVIRONMENT

CONTENT_OF_ENVIRONMENT:
          MOV        DX, OFFSET CONT_ENV_AREA
```

```
            MOV  AH, 09H
            INT  21H

            SUB  DI, DI
            MOV  BX, 2CH
            MOV  DS, [BX]
LOOP_ENV_STRING:
            CMP  BYTE PTR [DI], 00H ;ПРОВЕРКА НА КОНЕЦ СТРОКИ
            JE        NEXT_STRING
            MOV  DL, [DI];ВЫВОД
            MOV  AH, 02H
            INT  21H
            JMP  CHECK_PATH
NEXT_STRING:
            PUSH DS
            MOV  CX, CS
            MOV  DS, CX
            MOV  DX, OFFSET NEW_STR ;ПЕРЕНОС НА НОВУЮ СТРОКУ

            MOV  AH, 09H
            INT  21H

            POP  DS
CHECK_PATH:
            INC  DI
            CMP  WORD PTR [DI], 0001H ;НАЧАЛСЯ ПУТЬ
            JE        PRINT_PATH
            JMP  LOOP_ENV_STRING
PRINT_PATH:
            PUSH DS
            MOV  AX, CS
            MOV  DS, AX
            MOV  DX, OFFSET PATH

            MOV  AH, 09H
            INT  21H

            POP  DS
```

```
                    ADD   DI, 2 ;НА НАЧАЛО ПУТИ
          LOOP_PATH:
                    CMP   BYTE PTR [DI], 00H;ПРОВЕРКА НА КОНЕЦ ПУТИ
                    JE          TO_END
                    MOV   DL, [DI]
                    MOV   AH, 02H
                    INT   21H
                    INC   DI
                    JMP   LOOP_PATH
          TO_END:
                    RET
PRINT_INF       ENDP

BEGIN:
; . . . . . .
                    CALL  PRINT_INF
; . . . . . .
; ВЫХОД В DOS
                    XOR   AL,AL
                    MOV         AH,4CH
                    INT   21H
TESTPC              ENDS
                    END   START
```