

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. 8383

Ларин А.

Преподаватель

Ефремов М.А,

Санкт-Петербург

2020

Цель работы.

Исследование интерфейса управляющей программы и загрузочных модулей. Исследование префикса сегмента программы (PSP) и среды, передаваемой программе.

Выполнение

Написан код .COM модуля, который читает из PSP сегментный адрес недоступной памяти, сегментный адрес среды, передаваемой программе и выводит на экран вместе с хвостом командной строки, содержимым области среды и путем загружаемого модуля. Данный код был собран в .COM модуль

Результат исполнения .COM

```
C:\>L3.COM
9FFF
0188

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

C:\>L3.COM
```

Рисунок — Результат исполнения .COM

Код представлен в приложении А.

Среда передаваемая программе

1. Что такое среда?

Набор переменных, несущих в себе информацию о том, в каких условиях выполняется программа

2. Когда создается среда

Среда создается при запуске соответствующей программы обработчиком команд или другой программой.

3. Определяется программой запустившей данную. Если программа была запущена вручную родительской является обработчик команд COMMAND.COM

Выводы.

В результате работы были разобраны некоторые концепции языка ассемблера и работы операционной системы DOS. Был исследование

интерфейса управляющей программы и загрузочных модулей, а так же префикс сегмента программы (PSP) и среды, передаваемой программе.

ПРИЛОЖЕНИЕ А

LR2.ASM

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

```
START:  JMP BEGIN
;data
```

```
NEW_LINE db 0DH,0AH,'$'
```

```
STRING DB 'Some text          ',0DH,0AH,'$'
;procedures
```

```
DIGIT_TO_CHAR PROC near
;AL
```

```
    and al,0Fh
    cmp al,09h
    jle BLW
    add al,'A'
    sub al, 0Ah
    jmp DTC_CONT
```

```
BLW:
    add al,'0'
```

```
DTC_CONT:
    ret
```

```
DIGIT_TO_CHAR ENDP
```

```
;-----
```

```
PRINT_AS_HEX proc near
```

```
;AL - number
```

```
; ;breaks AX,CX,BX
```

```
    push ax
    push bx
    push cx
    push dx
    ;mov bx,dx
```

```
    mov ch,al
```

```
    mov cl,4
```

```
    shr al,cl
```

```
    call DIGIT_TO_CHAR
```

```
    mov dl,al
```

```
    mov ah,02h
```

```
    int 21h
```

```

    mov al,ch
    call DIGIT_TO_CHAR
    mov dl,al
    mov ah,02h
    int 21h
    ;mov dx,bx
    pop dx
    pop cx
    pop bx
    pop ax

    ret
PRINT_AS_HEX ENDP

PRINT_WORD proc near
;AX - word
    xchg AL,AH
    call PRINT_AS_HEX
    xchg AL,AH
    call PRINT_AS_HEX
    ret
PRINT_WORD ENDP

LN PROC
    push AX
    push DX
    mov DX, offset NEW_LINE
    mov AH, 9h
    int 21h
    pop DX
    pop AX
    ret
LN ENDP

;-----
BEGIN:
;Unaccessible memory
    mov AX, DS:[02h]
    call PRINT_WORD

    call LN
;Enviroment
    mov AX, DS:[2Ch]
    call PRINT_WORD

```

```

call LN
;Tail
    xor CX,CX
    mov CL, DS:[80h]
    xor SI, SI
lp: mov DL,[81h + SI]
    mov AH,02h
    int 21h
    inc DX
    loop lp

```

```

    call LN
;Enviroment content
    mov BX, 2Ch
    mov ES, [BX]
    xor SI,SI
    xor AX,AX
lp1:
    mov AX,ES:[SI]
    cmp AX,0001h
    je env_end
    cmp AL,0
    jne cnt
    call LN
cnt:
    mov DL,AL
    xor AX,AX
    mov AH,02h
    int 21h
    inc SI
    loop lp1

```

```

env_end:
    add SI,2
lp2:
    mov AX,ES:[SI]
    cmp AL,0
    je path_end
    mov DL,AL
    xor AX,AX
    mov AH,02h
    int 21h
    inc SI
    loop lp2
path_end:

```

```

EXIT:

```

```
xor AL,AL  
mov AH,4Ch  
int 21h  
TESTPC ENDS  
END START
```