

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА



Институт радиоэлектроники и информационных технологий

ОТЧЕТ

по лабораторной работе №2

«Разработка программы ввода-вывода и обработки последовательности кодов
на ассемблере»

Вариант 17

по дисциплине

«Принципы и методы организации системных
программных средств»

РУКОВОДИТЕЛЬ:

(подпись)

Викулова Е.Н.
(фамилия, и.,о.)

СТУДЕНТ:

(подпись)

Сухоруков В.А.
(фамилия, и.,о.)

19-В-2
(шифр группы)

Работа защищена «__» _____

С оценкой _____

Нижний Новгород 2021

Оглавление

Цель.....	3
Вариант задания.....	3
Теоретическая часть	3
Форматы исполняемых файлов	3
Модели памяти	3
При проектировании EXE файла программист должен:	3
Структура программы, организация ввода-вывода, используемые функции, особенности работы с видеопамью	5
Структура программы	5
Многосегментная программа	5
Односегментная программа.....	5
Функции, используемые для ввода-вывода	5
Работа с видеопамью	6
Тексты программ с комментариями.	7
lab2.asm.....	7
lab2com.asm	10
video.asm.....	13
Текст файла .lst и разбор содержимого файл.....	15
Lab2.lst	15
Разбор файла	19
Тексты файлов .map.....	20
Lab2.map	20
Lab2com.map	20
Результаты выполнения программ	21
Lab2.exe	21
Lab2com.com	21
video.exe.....	21
Выводы	22

Цель

Приобретение навыков: разработки одно- и многосегментных программ на языке ассемблер, использования функций прерываний для организации ввода-вывода, управление трансляцией и компоновкой.

Вариант задания

Перестановка $a(n), a(n-1), a(n-2), \dots, a(n/2), a(1), a(2), \dots, a(n/2-1)$.

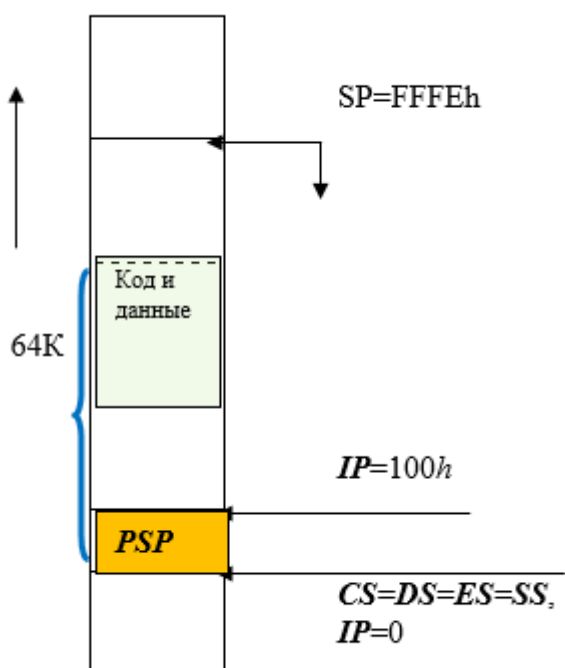
Теоретическая часть

Форматы исполняемых файлов

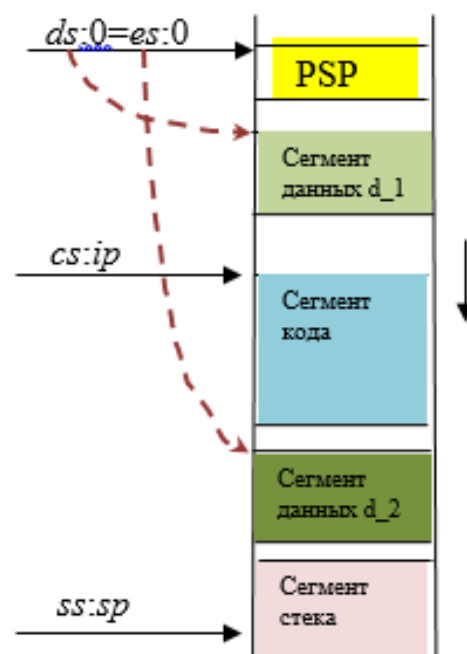
COM – полностью располагается в одном сегменте 64Кб, вся адресация - это смещения относительно одного сегментного адреса. Такой код не зависит от точки загрузки и может без настройки выполняться в любой области памяти.

EXE - программа состоит из нескольких сегментов. В EXE-файлах присутствует заголовочная часть с некоторым набором служебных таблиц для ОС (размером 512 байт или более).

Модели памяти



Структура COM файла



Структура EXE файла

При проектировании EXE файла программист должен:

1. Определить точку входа в программу, на которую при загрузке инициализируются cs: ip, например

2. Описать сегмент стека для инициализации SS: SP, например, так:

```
st1 segment para stack 'stack'
```

```
...
```

```
st1 ends
```

или так

```
.model small
```

```
.stack 256
```

3. В начале программы явно проинициализировать регистры DS (и если нужно ES) на начало соответствующего сегмента.

Например, если есть два сегмента данных с именами d_1 и d_2, и мы хотим чтобы сегментные адреса этих сегментов были в ds и es, то в начале программы необходим следующий код:

Структура программы, организация ввода-вывода, используемые функции, особенности работы с видеопамью

Структура программы Многосегментная программа

Сегмент *d1*

```
d1 segment para public 'data'
mess1 db 'Input: ',10,13,'$'
in_str db 22 dup (?)
d1 ends
```

Сегмент данных. Используется для инициализации строки сообщения и исходного массива.

Сегмент *e1*

```
e1 segment para public 'data'
mess2 db 10,13, 'Output: ',10,13,'$'
out_str db 20 dup ('$')
e1 ends
```

Сегмент данных. Используется для инициализации строки сообщения и нового массива.

Сегмент *c1*

Сегмент кода. Содержит код программы.

Односегментная программа

Содержит только один сегмент “.code”.

Функции, используемые для ввода-вывода

Для считывания массива байтов использована 10 функция 21h прерывания. Для вывода использована 9 функция 21h прерывания.

```
;СЧИТЫВАНИЕ ИСХОДНОГО МАССИВА
mov dx, offset in_str
mov in_str, 20
mov ah, 10
int 21h
```

```
;Вывод нового массива  
  
mov dx, offset out_str  
  
mov ah, 9  
  
int 21h
```

Работа с видеопамятью

Для вывода информации из биоса были использованы константа, содержащая адреса памяти и цвета текста.

```
.data  
  
bios equ 0FFFFh   ;Адрес биоса  
  
video equ 0B800h  ;Начало видеопамати  
  
color equ 0A3h    ;Цвет текста
```

Инициализация сегментов данных:

```
mov ax, bios  
  
mov es, ax  
  
mov ax, video  
  
mov ds, ax
```

Вывод информации:

```
mov al, es:[si]  
  
mov ah, color  
  
mov ds:[di], ax
```

Тексты программ с комментариями.

lab2.asm

```
; Лабораторная работа №2  
; Сухоруков Валерий  
; Перестановка a(n), a(n-1), a(n-2),  
; ..., a(n/2), a(1), a(2), ..., a(n/2-1).  
; Многосегментная программа. Создаётся exe файл
```

```
.model small  
.stack 200h
```

```
d1 segment para public 'data'  
mess1 db 'Input: ',10,13,'$'  
in_str db 22 dup (?)  
d1 ends
```

```
e1 segment para public 'data'  
mess2 db 10,13, 'Output: ',10,13,'$'  
out_str db 20 dup ('$')  
e1 ends
```

```
c1 segment para public 'code'  
assume cs:c1, ds:d1, es:e1
```

```
start:
```

```
;Инициализация сегментов
```

```
mov ax, d1  
mov ds, ax  
mov ax, e1  
mov es, ax
```

;Вывод строки запроса

`mov dx, offset mess1`

`mov ah, 9`

`int 21h`

;Считывание исходного массива

`mov dx, offset in_str`

`mov in_str, 20`

`mov ah, 10`

`int 21h`

;Инициализация регистра источника данных

; на конец исходного массива

`mov cl, in_str + 1` *;Количество элементов в массиве*

`xor ch, ch` *;Очистка регистра ch*

`mov ax, offset in_str+1` *;Адрес, предшествующий первому
элементу*

`add ax, cx` *;Запись адреса последнего
элемента*

`mov si, ax` *;Инициализация регистра
источника данных*

;Инициализация регистра приёмника данных

;на начало выходного массива

`mov di, offset out_str`

;Определения числа повторений первого цикла

*;- Перестановка второй части исходного массива в начало
нового в обратном порядке.*

;Число повторений - количество элементов делить на 2.

`shr cx, 1` *;Деление выполняется сдвигом
на 1 вправо*

m1:

```
mov al, [si]
mov es:[di],al
dec si
inc di
loop m1
```

*;Инициализация регистра источника данных на начало
исходного массива*

```
mov si, offset in_str+2
```

*;Определения числа повторений второго цикла - Перестановка
первой части исходного массива в конец нового.*

;Число повторений - количество элементов делить на 2.

```
mov cl, in_str + 1
shr cx,1
```

m2:

```
mov al,[si]
mov es:[di],al
inc si
inc di
loop m2
```

;Вывод строки, предшествующей выходному массиву

```
mov ax, es
mov ds, ax
mov dx, offset mess2
mov ah, 9
int 21h
```

```

    ;Вывод нового массива
    mov dx, offset out_str
    mov ah, 9
    int 21h

    mov ah, 7
    int 21h

    mov ax, 4c00h
    int 21h

c1 ends

end start

```

lab2com.asm

```

; Лабораторная работа №2
; Сухоруков Валерий
; Перестановка  $a(n), a(n-1), a(n-2),$ 
; ...,  $a(n/2), a(1), a(2), \dots, a(n/2-1)$ .
; Односегментная программа. Создаётся com файл

.model tiny

.code
org 100h
_main:
    ;Считывание исходного массива
    mov dx, offset in_str
    mov in_str, 20
    mov ah, 10

```

int 21h

```
    ;Инициализация регистра источника данных
; на конец исходного массива
mov cl, in_str + 1      ;Количество элементов в массиве
xor ch, ch              ;Очистка регистра ch
mov ax, offset in_str+1 ;Адрес, предшествующий первому
                        ;элементу
add ax,cx               ;Запись адреса последнего
                        ;элемента
mov si,ax               ;Инициализация регистра
                        ; источника данных
```

```
    ;Инициализация регистра приёмника данных
```

```
    ;на начало выходного массива
```

```
mov di, offset out_str
```

```
    ;Определения числа повторений первого цикла -
```

```
    ;Переставка второй части исходного массива в начало
```

```
    ;нового в обратном порядке.
```

```
    ;Число повторений - количество элементов делить на 2.
```

```
    shr cx,1            ;Деление выполняется сдвигом
                        ; на 1 в право
```

m1:

```
mov al, [si]
mov es:[di],al
dec si
inc di
loop m1
```

```

; Инициализация регистра источника данных
;на начало исходного массива
mov si, offset in_str+2

;Определения числа повторений второго цикла -
;Перестановка первой части исходного массива в конец нового.
;Число повторений - количество элементов делить на 2.
mov cl, in_str + 1
shr cx,1

m2:
mov al,[si]
mov es:[di],al
inc si
inc di
loop m2

;Вывод нового массива
mov dx, offset out_str
mov ah, 9
int 21h

mov ah, 7
int 21h

mov ax, 4c00h
int 21h

in_str db 22 dup (?)
out_str db 20 dup ('$')
end _main

```

video.asm

```
.model small
.stack 200h
.data
    bios equ 0FFFFh
    video equ 0B800h
    color equ 0A3h

.code
_main:
    ;Инициализация сегментов
    mov ah, 0
    mov al, 3
    int 10h
    mov ax, bios
    mov es, ax
    mov ax, video
    mov ds, ax

    ;Инициализация регистров источника и приёмника данных
    mov si, 05h
    mov di, 00h

    ;Инициализация счётчика
    mov cx, 0008h

m1:
    mov al, es:[si]
    mov ah, color
    mov ds:[di], ax
    inc di
```

```
inc di  
inc si  
loop ml
```

```
mov ah, 4ch  
mov al, 0  
int 21h
```

```
end _main
```

Текст файла .lst и разбор содержимого файл

Lab2.lst

Turbo Assembler

Version 3.1

08/11/21 20:40:05

Page 1

lab2.asm

```
1                                ; Лабораторная работа №2
2                                ; Сухоруков Валерий
3
4                                ; Многосегментная программа. Создаётся exe   файл
5
6
7 0000                          .model small
8 0000                          .stack 200h
9
10 0000                         dl segment para public 'data'
11 0000  49 6E 70 75 74 3A      20+ mess1 db 'Input: ',10,13,'$'
12      0A 0D 24
13 000A  16*(??)               in_str db 22 dup (?)
14 0020                         dl ends
15
16 0000                         el segment para public 'data'
17 0000  0A 0D 4F 75 74 70      75+ mess2 db 10,13, 'Output: ',10,13,'$'
18      74 3A 20 0A 0D 24
19 000D  14*(24)               out_str db      20 dup ('$')
20 0021                         el ends
21
22
23 0000                         cl segment para public 'code'
24                                assume cs:cl, ds:dl, es:el
25
26 0000                         start:
27                                ;Инициализация      сегментов
28 0000  B8 0000s               mov ax, dl
29 0003  8E D8                  mov ds, ax
30 0005  B8 0000s               mov ax, el
31 0008  8E C0                  mov es, ax
32
33                                ;Вывод строки запроса
```

```

34 000A BA 0000r      mov dx, offset      mess1
35 000D B4 09          mov ah, 9
36 000F CD 21          int 21h
37
38                    ;Считывание исходного массива
39 0011 BA 000Ar      mov dx, offset      in_str
40 0014 C6 06 000Ar 14  mov in_str, 20
41 0019 B4 0A          mov ah, 10
42 001B CD 21          int 21h
43
44                    ;Инициализация      регистра источника данных на конец ис-
ходного массива
45 001D 8A 0E 000Br    mov cl, in_str      + 1      ;Количество      элемен-
тов в массиве
46 0021 32 ED          xor ch, ch      ;Очистка регистра ch
47 0023 B8 000Br      mov ax, offset      in_str+1    ;Адрес, предшествующий
первому элементу
48 0026 03 C1          add ax, cx      ;Запись адреса последнего элемента
49 0028 8B F0          mov si, ax      ;Инициализация регистра источника
данных
50
51                    ;Инициализация      регистра приёмника данных на начало
выходного массива
52 002A BF 000Dr      mov di, offset      out_str
53
54                    ;Определения числа повторений первого цикла - Переставка
второй
55                    ;части исходного массива в начало нового в обратном порядке.
56                    ;Число повторений - количество      элементов делить на 2.
57 002D D1 E9          shr cx, 1      ;Деление выполняется сдвигом на 1
в право
Turbo Assembler      Version 3.1      08/11/21 20:40:05      Page 2
lab2.asm

```

```

58
59 002F      m1:
60 002F 8A 04          mov al, [si]
61 0031 26: 88 05          mov es:[di], al
62 0034 4E          dec si
63 0035 47          inc di
64 0036 E2 F7          loop m1
65

```


66		<i>;Инициализация</i>	<i>регистра источника данных на начало</i>
исходного массива			
67 0038	BE 000Cr	mov si, offset	in_str+2
68			
69		<i>;Определения числа повторений второго цикла - Переставка</i>	
первой			
70		<i>;части исходного массива в конец нового.</i>	
71		<i>;Число повторений - количество</i>	<i>элементов делить на 2.</i>
72 003B	8A 0E 000Br	mov cl, in_str	+ 1
73 003F	D1 E9	shr cx, 1	
74			
75			
76 0041		m2:	
77 0041	8A 04	mov al, [si]	
78 0043	26: 88 05	mov es: [di], al	
79 0046	46	inc si	
80 0047	47	inc di	
81 0048	E2 F7	loop m2	
82			
83		<i>;Вывод строки, предшествующей выходному массиву</i>	
84 004A	8C C0	mov ax, es	
85 004C	8E D8	mov ds, ax	
86 004E	BA 0000r	mov dx, offset	mess2
87 0051	B4 09	mov ah, 9	
88 0053	CD 21	int 21h	
89			
90		<i>;Вывод нового массива</i>	
91 0055	BA 000Dr	mov dx, offset	out_str
92 0058	B4 09	mov ah, 9	
93 005A	CD 21	int 21h	
94			
95 005C	B4 07	mov ah, 7	
96 005E	CD 21	int 21h	
97			
98 0060	B8 4C00	mov ax, 4c00h	
99 0063	CD 21	int 21h	
100			
101 0065		cl ends	
102			
103		end start	

Symbol Table

Symbol Name	Type	Value
??DATE	Text	"08/11/21"
??FILENAME	Text	"lab2 "
??TIME	Text	"20:40:05"
??VERSION	Number	030A
@32BIT	Text	0
@CODE	Text	_TEXT
@CODESIZE	Text	0
@CPU	Text	0101H
@CURSEG	Text	C1
@DATA	Text	DGROUP
@DATASIZE	Text	0
@FILENAME	Text	LAB2
@INTERFACE	Text	00H
@MODEL	Text	2
@STACK	Text	DGROUP
@WORDSIZE	Text	2
IN_STR	Byte	D1:000A
M1	Near	C1:002F
M2	Near	C1:0041
MESS1	Byte	D1:0000
MESS2	Byte	E1:0000
OUT_STR	Byte	E1:000D
START	Near	C1:0000

Groups & Segments	Bit	Size	Align	Combine	Class
C1	16	0065	Para	Public	CODE
D1	16	0020	Para	Public	DATA
DGROUP			Group		
STACK	16	0200	Para	Stack	STACK
_DATA	16	0000	Word	Public	DATA
E1	16	0021	Para	Public	DATA
_TEXT	16	0000	Word	Public	CODE

Разбор файла

1. В начале файла указывается версия Turbo Assembler'a, время создания. `.lst` и `.obj` файлов, название исходного файла с расширением `.asm`
2. После следует код `.asm` файла с указанием номеров строк и кодами команд.
3. Далее записаны:
 - 1) Дата создания
 - 2) Название файла
 - 3) Время создания
 - 4) Версия Turbo Assembler'a
 - 5) Разрядность (32 или 16)
 - 6) Переменные, используемые для исходного и выходного массива и их адреса
 - 7) Метки, по которым осуществлялся переход и их адрес
 - 8) Переменные, используемые для вывода строк
 - 9) Адрес начала сегмента кода

Тексты файлов .map

Lab2.map

Start	Stop	Length	Name	Class
00000H	00000H	00000H	__TEXT	CODE
00000H	00064H	00065H	C1	CODE
00070H	00070H	00000H	__DATA	DATA
00070H	0008FH	00020H	D1	DATA
00090H	000B0H	00021H	E1	DATA
000C0H	002BFH	00200H	STACK	STACK

Program entry point at 0000:0000

Lab2com.map

Start	Stop	Length	Name	Class
00000H	0018EH	0018FH	__TEXT	CODE
00190H	00190H	00000H	__DATA	DATA

Результаты выполнения программ

Lab2.exe

```
Администратор: Командная строка - tlink lab2.obj
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Users\1>cd C:\Users\1\Desktop\assembler\LW2\lab2

C:\Users\1\Desktop\assembler\LW2\lab2>tasm lab2.asm /la
Turbo Assembler Version 3.1 Copyright (c) 1988, 1992 Borland International

Assembling file:   lab2.asm
Error messages:   None
Warning messages: None
Passes:           1
Remaining memory: 451k

C:\Users\1\Desktop\ASSEMB~1\LW2\lab2>tlink lab2.obj
Turbo Link Version 5.1 Copyright (c) 1992 Borland International

C:\USERS\1\DESKTOP\ASSEMB~1\LW2\LAB2>lab2.exe
Input:
Sukhorukov
Output:
vokurSukho
C:\USERS\1\DESKTOP\ASSEMB~1\LW2\LAB2>
```

Lab2com.com

```
Администратор: Командная строка - tlink lab2com.obj /t - lab2com.com
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Users\1>cd C:\Users\1\Desktop\assembler\LW2\lab2com

C:\Users\1\Desktop\assembler\LW2\lab2com>tasm lab2com.asm /la
Turbo Assembler Version 3.1 Copyright (c) 1988, 1992 Borland International

Assembling file:   lab2com.asm
Error messages:   None
Warning messages: None
Passes:           1
Remaining memory: 452k

C:\Users\1\Desktop\ASSEMB~1\LW2\lab2com>tlink lab2com.obj /t
Turbo Link Version 5.1 Copyright (c) 1992 Borland International

C:\USERS\1\DESKTOP\ASSEMB~1\LW2\LAB2COM>lab2com.com
vokurSukho_
```

video.exe

```
Администратор: Командная строка - tlink video.obj
C:\USERS\1\DESKTOP\ASSEMB~1\LW2\VIDEO>
```

Выводы

В ходе выполнения лабораторной работы были приобретены навыки: разработки одно- и многосегментных программ на языке ассемблер, использования функций прерываний для организации ввода-вывода, управление трансляцией и компоновкой.