

# **Отчёт по лабораторной работе № 7**

**Дисциплина: Архитектура компьютера**

Вакутайпа Милдред

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>4</b>	<b>Выводы</b>	<b>21</b>
<b>5</b>	<b>Список литературы</b>	<b>22</b>

# Список иллюстраций

3.1	Рис 1	. . . . .	6
3.2	Рис 2	. . . . .	7
3.3	Рис 3	. . . . .	8
3.4	Рис 4	. . . . .	9
3.5	Рис 5	. . . . .	10
3.6	Рис 6	. . . . .	11
3.7	Рис 7	. . . . .	12
3.8	Рис 8	. . . . .	12
3.9	Рис 9	. . . . .	13
3.10	Рис 10	. . . . .	14
3.11	Рис 11	. . . . .	14
3.12	Рис 12	. . . . .	14
3.13	Рис 13	. . . . .	15
3.14	Рис 14	. . . . .	15
3.15	Рис 15	. . . . .	16
3.16	Рис 16	. . . . .	16
3.17	Рис 17	. . . . .	16
3.18	Рис 18	. . . . .	19
3.19	Рис 19	. . . . .	20

# 1 Цель работы

Цель этой работы - изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

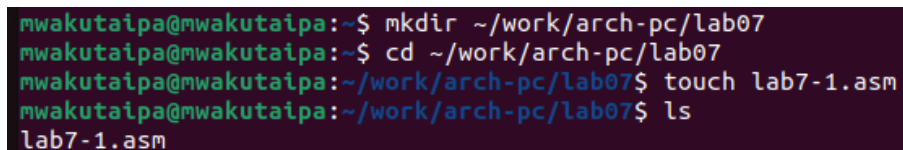
## 2 Задание

1. Реализация переходов в NASM
2. Изучение структуры файлы листинга

## 3 Выполнение лабораторной работы

### 1. Реализация переходов в NASM

Создаю каталог для программ лабораторной работы № 7, перехожу в него и создаю файл lab7-1.asm:

A terminal window with a dark background and light-colored text. The text shows a series of commands being executed in a shell. The user is in a directory named ~/work/arch-pc/lab07. They create a new file named lab7-1.asm using the 'touch' command. Finally, they list the contents of the directory using 'ls', which shows 'lab7-1.asm' as the only file.

```
mwakutaipa@mwakutaipa:~$ mkdir ~/work/arch-pc/lab07
mwakutaipa@mwakutaipa:~$ cd ~/work/arch-pc/lab07
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ touch lab7-1.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ls
lab7-1.asm
```

Рис. 3.1: Рис 1

Открываю файл lab7-1.asm и в него вставляю код программы, которая показывает как работает jmp:

```
GNU nano 6.2
#include 'in_out.asm'

SECTION .data

msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0

SECTION .text
GLOBAL _start
_start:

jmp _label2

_label1:
mov eax, msg1
call sprintfLF

_label2:
mov eax, msg2
call sprintfLF

_label3:
mov eax, msg3
call sprintfLF

_end:
call quit
```

Рис. 3.2: Рис 2

Создаю исполняемый файл и запускаю его. Программа выводит “сооб-

щение № 2” и “сообщение № 3”:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$
```

Рис. 3.3: Рис 3

Изменяю текст программы:



```

GNU nano 6.2
#include 'in_out.asm'

SECTION .data

msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0

SECTION .text
GLOBAL _start
_start:

jmp _label2

_label1:
mov eax, msg1
call sprintf
jmp _end

_label2:
mov eax, msg2
call sprintf
jmp _label1

_label3:
mov eax, msg3
call sprintf

_end:
call quit

```

Рис. 3.4: Рис 4

Создаю исполняемый файл и запускаю его. Программа выводит “сообщение № 2” и “сообщение № 1”:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ mc
```

Рис. 3.5: Рис 5

Изменяю текст программы, чтобы она выводила “сообщение № 3”, “сообщение № 2” и “сообщение № 1”:

```
GNU nano 6.2
#include 'in_out.asm'

SECTION .data

msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0

SECTION .text
GLOBAL _start
_start:

jmp _label3

_label1:
mov eax, msg1
call sprintf
jmp _end
_label2:
mov eax, msg2
call sprintf
jmp _label1

_label3:
mov eax, msg3
call sprintf
jmp _label2

_end:
call quit
```

Рис. 3.6: Рис 6

Создаю исполняемый файл и запускаю его:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

Рис. 3.7: Рис 7

Создаю файл lab7-2.asm:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ touch lab7-2.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ls
in_out.asm lab7-1 lab7-1.asm lab7-1.o lab7-2.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$
```

Рис. 3.8: Рис 8

В него вставляю код программы, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных:

```

#include 'in_out.asm'

SECTION .data
msg1 db 'Введите B: ',0h
msg2 db "Наибольшее число: "
A dd '20'
C dd '50'

SECTION .bss
max resb 10
B resb 10

SECTION .text
GLOBAL _start

_start:
mov eax, msg1
call sprint

mov ecx, B
mov edx, 10
call sread

mov eax, B
call atoi
mov [B], eax

mov ecx, [A]
mov [max], ecx

cmp ecx, [C]
jg check_B
mov ecx, [C]
mov [max], ecx

check_B:
mov eax, max
call atoi
mov [max], eax

mov ecx, [max]
cmp ecx, [B]
jg fin
mov ecx, [B]
mov [max], ecx

fin:

```

Рис. 3.9: Рис 9

```
fin:
mov eax,msg2
call sprint
mov eax,[max]
call iprintLF
call quit
```

Рис. 3.10: Рис 10

Создаю исполняемый файл и запускаю его:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-2
Введите B:
```

Рис. 3.11: Рис 11

Проверяю работу для разных значений B:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 4
Наибольшее число: 50

mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 56
Наибольшее число: 56

mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 30
Наибольшее число: 50
```

Рис. 3.12: Рис 12

## 2. Изучение структуры файлы листинга

Создаю файл листинга для программы из файла lab7-2.asm:

```

mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ls
in_out.asm  lab7-1.asm  lab7-2      lab7-2.lst
lab7-1      lab7-1.o    lab7-2.asm  lab7-2.o

```

Рис. 3.13: Рис 13

Открываю файл листинга lab7-2.lst с помощью mcedit:

```

/home/mw-7-2.lst [----] 0 L:[183+10 193/228] *(11656/13376b) 0032 0x020[*][X]
8.....
9.....          SECTION .bss
10 00000000 <res Ah>      max resb 10
11 0000000A <res Ah>      B resb 10
12.....
13.....          SECTION .text
14.....          GLOBAL _start
15.....
16.....          _start:
17 000000E8 B8[00000000]   mov eax, msg1
18 000000ED E81DFFFFFF    call sprint
19.....
20 000000F2 B9[0A000000]   mov ecx, B
21 000000F7 BA0A000000    mov edx, 10
22 000000FC E842FFFFFF    call sread
23.....
24 00000101 B8[0A000000]   mov eax, B
25 00000106 E891FFFFFF    call atoi
26 0000010B A3[0A000000]   mov [B], eax
27.....
28 00000110 8B0D[33000000]   mov ecx, [A]
29 00000116 890D[00000000]   mov [max], ecx

```

Рис. 3.14: Рис 14

Это пример машинного кода сохранен в lab7-2.lst:

```

20 000000F2 B9[0A000000]   mov ecx, B
21 000000F7 BA0A000000    mov edx, 10
22 000000FC E842FFFFFF    call sread

```

В lab7-2.asm, эти строки предназначены для ввода значения В. 20- номер строки, 000000F2- это смещение машинного кода от начала текущего сегмента (адрес), B9[0A000000]- машинный код и mov ecx,B- исходный текст программы.

Когда я удаляю строку для сравнения А и С, выполняю трансляцию с получением файла листинга, строка удаляется из файла .lst, и ничего не добавляется:

```

26 mov [B],eax
27
28 mov ecx,[A]
29 mov [max],ecx
30
31
32 jg check_B
33 mov ecx,[C]
34

```

Рис. 3.15: Рис 15

```

30 .....
31 .....
32 0000011C 7F0C                jg check_B
33 0000011E 8B0D[37000000]          mov ecx,[C]
34 00000124 890D[00000000]          mov [max],ecx
35 .....
36 .....                check_B:
37 0000012A B8[00000000]          mov eax,max

```

Рис. 3.16: Рис 16

#Выполнение задания для самостоятельной работы

Создаю файл task1.asm:

```

mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ touch task1.asm

```

Рис. 3.17: Рис 17

В него вставляю код программы, которая определяет наименьшей из 3 целочисленных переменных a,b и c:

```
%include 'in_out.asm'
```

```
SECTION .data
```

```
msg1 db "Наименьшее число: ",0h
```

```
A dd '41'
```

```
B dd '35'
```



```
C dd '62'
```

```
SECTION .bss
```

```
min resb 10
```

```
SECTION .text
```

```
GLOBAL _start
```

```
_start:
```

```
mov eax,B
```

```
call atoi
```

```
mov [B],eax
```

```
mov ecx,[A]
```

```
mov [min],ecx
```

```
cmp ecx,[C]
```

```
jnl check_B
```

```
mov ecx,[C]
```

```
mov [min],ecx
```

```
check_B:
```

```
mov eax,min
```

```
call atoi
```

```
mov [min],eax
```

```
mov ecx,[min]
```

```
cmp ecx,[B]
jl fin
mov ecx,[B]
mov [min],ecx
```

```
fin:
mov eax,msg1
call sprint
mov eax,[min]
call iprintLF
call quit
```

```

GNU nano 6.2
#include 'in_out.asm'

SECTION .data
msg1 db "Наименьшее число: ",0h
A dd '41'
B dd '35'
C dd '62'

SECTION .bss
min resb 10

SECTION .text
GLOBAL _start

_start:

mov eax,B
call atoi
mov [B],eax

mov ecx,[A]
mov [min],ecx

cmp ecx,[C]
jl check_B
mov ecx,[C]
mov [min],ecx

check_B:
mov eax,min
call atoi
mov [min],eax

mov ecx,[min]
cmp ecx,[B]
jl fin
mov ecx,[B]
mov [min],ecx

fin:
mov eax,msg1
call sprint
mov eax,[min]
call iprintLF
call quit

```

Рис. 3.18: Рис 18

Создаю исполняемый файл и проверяю его работу для значения переменных из варианта 10:

```
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ nasm -f elf task1.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ gedit task1.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ nasm -f elf task1.asm
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ld -m elf_i386 -o task1 task1.o
mwakutaipa@mwakutaipa:~/work/arch-pc/lab07$ ./task1
Наименьшее число: 35
```

Рис. 3.19: Рис 19

## 4 Выводы

При выполнении лабораторной работы, я изучила команд условного и безусловного переходов в NASM.

## 5 Список литературы

Архитектура ЭВМ