

# **Отчет по лабораторной работе № 6**

**дисциплина: Архитектура компьютера**

Нечаева Кира

# Содержание

<b>1</b>	<b>1 Цель работы</b>	<b>5</b>
<b>2</b>	<b>2 Задание</b>	<b>6</b>
<b>3</b>	<b>3 Выполнение лабораторной работы</b>	<b>7</b>
3.1	1. Реализация переходов в NASM . . . . .	7
3.2	2. Изучение структуры файла листинга . . . . .	10
3.3	3. Задание для самостоятельной работы . . . . .	11
<b>4</b>	<b>4 Вывод</b>	<b>14</b>
<b>5</b>	<b>5 Источники</b>	<b>15</b>

## **Список иллюстраций**

## Список таблиц

# **1 1 Цель работы**

Целью работы является изучение команд условного и безусловного переходов, приобретение навыков написания программ с использованием переходов, а также знакомство с назначением и структурой файла листинга.

## **2 2 Задание**

1. Реализация переходов в NASM
2. Изучение структуры файла листинга
3. Задание для самостоятельной работы

## 3 3 Выполнение лабораторной работы

### 3.1 1. Реализация переходов в NASM

Для начала я создаю каталог для программ лабораторной работы № 7, перехожу

```
[kanechaeva@fedora ~]$ mkdir ~/work/arch-pc/lab07
[kanechaeva@fedora ~]$ cd ~/work/arch-pc/lab07
[kanechaeva@fedora lab07]$ touch lab7-1.asm
[kanechaeva@fedora lab07]$
```

в него и создаю файл lab7-1.asm. (рис. [??])

В качестве примера программы с использованием инструкции `jmp` воспользуюсь текстом программы из листинга 7.1. Для этого я введу его в файл lab7-1.asm, скопировав заранее файл `in_out.asm` в папку (рис. [??])

```
lab7-1.asm      [-M--] 31 L:[ 1+19 20/ 20] *(629
%include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

```
[kanechaeva@fedora lab07]$ nasm -f elf lab7-1.asm
[kanechaeva@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[kanechaeva@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 3
[kanechaeva@fedora lab07]$
```

Создаю исполняемый файл и запускаю его. (рис. [??])

Теперь я изменю программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого изменяю текст программы в соответствии с листингом 7.2. ((рис. [??])

```
lab7-1.asm      [-M--] 11 L:[ 1+21 22/ 22] *(
%include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Создаю исполняемый

```
[kanechaeva@fedora lab07]$ nasm -f elf lab7-1.asm
[kanechaeva@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[kanechaeva@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 1
[kanechaeva@fedora lab07]$
```

файл и запускаю его. (рис. [??])

Теперь мне нужно изменить текст программы таким образом, чтобы сначала выводилось “Сообщение №3”, затем “Сообщение №2” и в конце “Сообщение №1”.



```

jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
jmp _label2
_end:

```

((рис. [??])

1Помощь 2Сох~ть 3Блок 4Замена 5Копия

Создаю ис-

```

[kanechaeva@192 lab07]$ nasm -f elf lab7-1.asm
[kanechaeva@192 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[kanechaeva@192 lab07]$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
[kanechaeva@192 lab07]$

```

полняемый файл и запускаю его. (рис. [??])

Создаю файл lab7-2.asm в каталоге ~/work/arch-рс/lab07 и ввожу в него текст

```

lab7-2.asm      [-M--]  9 L:[ 1+ 0  1/ 49] *(9  /1
%include 'in_out.asm'
section .data
msg1 db 'Введите В: ',0h
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
; ----- Вывод сообщения 'Введите В: '
mov eax,msg1
call sprint
; ----- Ввод 'В'
mov ecx,B
mov edx,10
call sread
; ----- Преобразование 'В' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число

```

программы из листинга 7.3. (рис. [??])

Создаю исполняемый файл и проверяю его работу для разных значений В. (рис.

```

[kanechaeva@fedora lab07]$ nasm -f elf lab7-2.asm
[kanechaeva@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[kanechaeva@fedora lab07]$ ./lab7-2
Введите В: 7
Наибольшее число: 50

```

[??])

## 3.2 2. Изучение структуры файла листинга

Создаю файл листинга для программы из файла lab7-2.asm. (рис. [??])

```

[kanechaeva@192 lab07]$ nasm -f elf -l lab7-2.lst lab7-2.asm

```

Открываю файл

листинга lab7-2.lst с помощью текстового редактора mcedit.(рис. [??])

```

lab7-2.lst [----] 0 L: [ 1+ 0 1/225] *(0 /14458b) 0032 0x020 [*][X]
1          %include 'in_out.asm'
2          <1> ;----- slen -----
3          <1> ; Функция вычисления длины сообщения
4          <1> slen:.....
5          00000000 53          <1> push    ebx.....
6          00000001 89C3       <1> mov     ebx, eax.....
7          <1>.....
8          00000003 803800     <1> cmp     byte [eax], 0...
9          00000006 7403       <1> jz      finished.....
10         00000008 40         <1> inc     eax.....
11         00000009 EBF8       <1> jmp     nextchar.....
12         <1>.....
13         <1> finished:
14         0000000B 29D8       <1> sub     eax, ebx
15         0000000D 5B         <1> pop     ebx.....
16         0000000E C3         <1> ret.....
17         <1>.....
18         <1>.....
19         <1> ;----- sprint -----
20         <1> ; Функция печати сообщения
21         <1> ; входные данные: mov eax,<message>

```

Объясняю содержи-

мое трёх строк файла листинга: 35 call atoi - вызов подпрограммы перевода символа в число; 36 mov [max],eax - запись преобразованного числа в max; 39 cmp ecx,[B] - сравнение 'max(A,C)' и 'B'. Открываю файл с программой lab7-2.asm и в инструкции с двумя операндами удаляю один операнд. (рис. [??])

```

; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx, ; Сравниваем 'A' и 'C'

```

Выполняю трансляцию

с получением файла листинга. [??])

```

[kanechaeva@192 lab07]$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:28: error: invalid combination of opcode and operands

```

Выходных файлов я не получаю. Программа выдаёт ошибку, т.к. в данной операции должны присутствовать два операнда, а не один.

### 3.3 3.Задание для самостоятельной работы

№1 Требуется написать программу нахождения наименьшей из 3 целочисленных переменных  $\boxtimes, \boxtimes$  и с. При выполнении лабораторной работы №6 я получила 12 вариантов. Значит, в соответствии с таблицей 7.5, значения моих переменных – 99,29,26. Создаю файл lab7-3.asm и пишу в нём программу. ([??])

```

lab7-3.asm      [----] 13 L: [ 1+ 0 1/ 42] *(13 /1574b) 0111 0x06F [*][X]
#include 'in_out.asm'
section .data
msg1 db "Наибольшее число: ",0h
a dd '99'
b dd '29'
c dd '26'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
; ----- Преобразование 'B' из символа в число
mov eax,b
call atoi ; Вызов подпрограммы перевода символа в число
mov [b],eax ; запись преобразованного числа в 'B'
; ----- Записываем 'A' в переменную 'max'
mov ecx,[a] ; 'ecx = A'
mov [max],ecx ; 'max = A'
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[c] ; Сравниваем 'A' и 'C'
jg check_b ; если 'A>C', то переход на метку 'check_B',
1Помощь 2Сох-ть 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход

```

Создаю исполняемый

```

[kanechaeva@192 lab07]$ nasm -f elf lab7-3.asm
[kanechaeva@192 lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[kanechaeva@192 lab07]$ ./lab7-3
Наибольшее число: 99
[kanechaeva@192 lab07]$

```

файл и проверяю его работу. (рис. [??])

№2 Теперь мне нужно написать программу, которая для введенных с клавиатуры значений  $x$  и  $a$  вычисляет значение заданной функции  $f(x)$  и выводит результат вычислений. Вид моей функции  $f(x)$ :  $f(x) = x^2$ ,  $x < 5$   $f(x) = x - 5$ ,  $x \geq 5$  Создаю файл lab7-4.asm и пишу в нём программу. ([??])

```

lab7-4.asm      [----] 7 L: [ 1+13 14/ 79] *(219 /2117b) 0010 0x00A [*][X]
#include 'in_out.asm'
SECTION .data
msg1 db 'Введите x: ', 0h
msg2 db 'Введите a: ', 0h
msg3 db 'Результат: ', 0h

SECTION .bss
x resb 11
a resb 11
res resb 12

SECTION .text
global _start
_start:
; ----- Вывод сообщения 'Введите x: '
mov eax, msg1
call sprint

; ----- Ввод 'x'
mov ecx, x
mov edx, 10
call sread
1Помощь 2Сох-ть 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход

```

Создаю исполняемый

файл и проверяю его работу сначала для  $x_1=3$ ,  $a_1=7$ , а затем для  $x_2=6$ ,  $a_2=4$ . (рис.

```
[kanechaeva@192 lab07]$ nasm -f elf lab7-4.asm
[kanechaeva@192 lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[kanechaeva@192 lab07]$ ./lab7-4
Введите x: 3
Введите a: 7
Результат: 21
[kanechaeva@192 lab07]$ nasm -f elf lab7-4.asm
[kanechaeva@192 lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[kanechaeva@192 lab07]$ ./lab7-4
Введите x: 6
Введите a: 4
Результат: 1
```

[??])

## **4 4 Вывод**

При выполнении данной лабораторной работы я изучила команды условного и безусловного переходов, приобрела навыки написания программ с использованием переходов, а также ознакомилась с назначением и структурой файла листинга.

