

# **Отчёт по лабораторной работе 7**

**Архитектура компьютеров**

Акмухаммедов Айдын

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	20

## Список иллюстраций

2.1	Программа в файле lab7-1.asm . . . . .	7
2.2	Запуск программы lab7-1.asm . . . . .	8
2.3	Программа в файле lab7-1.asm: . . . . .	9
2.4	Запуск программы lab7-1.asm: . . . . .	9
2.5	Программа в файле lab7-1.asm . . . . .	10
2.6	Запуск программы lab7-1.asm . . . . .	11
2.7	Программа в файле lab7-2.asm . . . . .	12
2.8	Запуск программы lab7-2.asm . . . . .	13
2.9	Файл листинга lab7-2 . . . . .	13
2.10	Ошибка трансляции lab7-2 . . . . .	15
2.11	Файл листинга с ошибкой lab7-2 . . . . .	15
2.12	Программа в файле task.asm . . . . .	16
2.13	Запуск программы task.asm . . . . .	17
2.14	Программа в файле task2.asm . . . . .	18
2.15	Запуск программы task2.asm . . . . .	19

## **Список таблиц**

# 1 Цель работы

Целью работы является изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

## 2 Выполнение лабораторной работы

1. Создал каталог для программ лабораторной работы № 7 и файл lab7-1.asm
2. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp.

Написал в файл lab7-1.asm текст программы из листинга 7.1.

```

%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 sprintLF

_label2:
mov eax, msg2
call sprintLF

_label3:
mov eax, msg3
call sprintLF

_end:
call quit

```

Рис. 2.1: Программа в файле lab7-1.asm

Создал исполняемый файл и запустил его.

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

Рис. 2.2: Запуск программы lab7-1.asm

Инструкция `jmp` позволяет осуществлять переходы не только вперед но и назад. Изменим программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию `jmp` с меткой `_label1` (т.е. переход к инструкциям вывода сообщения № 1) и после вывода сообщения № 1 добавим инструкцию `jmp` с меткой `_end` (т.е. переход к инструкции `call quit`).

Изменил текст программы в соответствии с листингом 7.2.



```

%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 sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF

_end:
call quit

```

Рис. 2.3: Программа в файле lab7-1.asm:

```

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

```

Рис. 2.4: Запуск программы lab7-1.asm:

Изменил текст программы, изменив инструкции `jmp`, чтобы вывод программы был следующим:

Сообщение № 3

Сообщение № 2

Сообщение № 1

```
%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 sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

Рис. 2.5: Программа в файле lab7-1.asm

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

Рис. 2.6: Запуск программы lab7-1.asm

3. Использование инструкции `jmp` приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход должен происходить если выполнено какое-либо условие. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: A, B и C. Значения для A и C задаются в программе, значение B вводится с клавиатуры.

Создал исполняемый файл и проверил его работу для разных значений B.

```

; ----- Записываем 'A' в переменную 'max'
mov ecx,[A]
mov [max],ecx
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C]
jg check_B
mov ecx,[C]
mov [max],ecx
; ----- Преобразование 'max(A,C)' из символа в
число
check_B:
mov eax,max
call atoi
mov [max],eax
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B]
jg fin
mov ecx,[B]
mov [max],ecx
; ----- Вывод результата
fin:
mov eax, msg2
call sprintf
mov eax,[max]
call iprintLF
call quit

```

Рис. 2.7: Программа в файле lab7-2.asm

```

aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-2.o -o lab7-2
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 3
Наибольшее число: 50
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 45
Наибольшее число: 50
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 55
Наибольшее число: 55
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$

```

Рис. 2.8: Запуск программы lab7-2.asm

4. Обычно `nasm` создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ `-l` и задав имя файла листинга в командной строке.

Создал файл листинга для программы из файла lab7-2.asm

```

199      24      ; ----- Записываем 'A' в переменную 'max'
200      25  00000110 8B0D[35000000]    mov ecx,[A]
201      26  00000116 8B0D[00000000]    mov [max],ecx
202      27      ; ----- Сравниваем 'A' и 'C' (как символы)
203      28  0000011C 3B0D[39000000]    cmp ecx,[C]
204      29  00000122 7F0C                jg check_R
205      30  00000124 8B0D[39000000]    mov ecx,[C]
206      31  0000012A 8B0D[00000000]    mov [max],ecx
207      32      ; ----- Преобразование 'max(A,C)' из символа в число
208      33      check_R:
209      34  00000130 B8[00000000]    mov eax,max
210      35  00000135 EB62FFFFFF        call atoi
211      36  0000013A A3[00000000]    mov [max],eax
212      37      ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
213      38  0000013F 8B0D[00000000]    mov ecx,[max]
214      39  00000145 3B0D[0A000000]    cmp ecx,[B]
215      40  0000014B 7F0C                jg fin
216      41  0000014D 8B0D[0A000000]    mov ecx,[B]
217      42  00000153 8B0D[00000000]    mov [max],ecx
218      43      ; ----- Вывод результата
219      44      fin:
220      45  00000159 B8[13000000]    mov eax, msg2
221      46  0000015F EBACFFFFFF        call sprintf
222      47  00000163 A1[00000000]    mov eax,[max]
223      48  00000168 EB19FFFFFF        call printf
224      49  0000016D EB69FFFFFF        call quit

```

Рис. 2.9: Файл листинга lab7-2

Внимательно ознакомился с его форматом и содержимым. Подробно объясню содержимое трёх строк файла листинга по выбору.

строка 211

- 34 - номер строки
- 0000012E - адрес
- B8[00000000] - машинный код
- mov eax, max - код программы

строка 212

- 35 - номер строки
- 00000133 - адрес
- E864FFFFFF - машинный код
- call atoi - код программы

строка 213

- 36 - номер строки
- 00000138 - адрес
- A3[00000000] - машинный код
- mov [max], eax - код программы

Открыл файл с программой lab7-2.asm и в инструкции с двумя операндами удалил один операнд. Выполнил трансляцию с получением файла листинга.

```

aidinakmuhammedov@VirtualBox: ~/work/arch-pc/lab07$
aidinakmuhammedov@VirtualBox: ~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm -l lab
7-2.lst
aidinakmuhammedov@VirtualBox: ~/work/arch-pc/lab07$
aidinakmuhammedov@VirtualBox: ~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm -l lab
7-2.lst
lab7-2.asm:34: error: invalid combination of opcode and operands
aidinakmuhammedov@VirtualBox: ~/work/arch-pc/lab07$
aidinakmuhammedov@VirtualBox: ~/work/arch-pc/lab07$

```

Рис. 2.10: Ошибка трансляции lab7-2

```

197 22 00000100 E877FFFFFF mov eax,
198 23 00000100 A3[0A000000] mov [R],eax
199 24 ; ----- Записываем 'A' в переменную 'max'
200 25 00000110 8B0D[35000000] mov ecx,[A]
201 26 00000116 8B0D[00000000] mov [max],ecx
202 27 ; ----- Сравниваем 'A' и 'C' (как символы)
203 28 0000011C 3B0D[39000000] cmp ecx,[C]
204 29 00000122 7F0C ig check_R
205 30 00000124 8B0D[39000000] mov ecx,[C]
206 31 0000012A 8B0D[00000000] mov [max],ecx
207 32 ; ----- Преобразование 'max(A,C)' из символа в число
208 33 check_R:
209 34 mov eax,
210 34 ***** error: invalid combination of opcode and operands
211 35 00000130 E867FFFFFF call atoi
212 36 00000135 A3[00000000] mov [max],eax
213 37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
214 38 0000013A 8B0D[00000000] mov ecx,[max]
215 39 00000140 3B0D[0A000000] cmp ecx,[B]
216 40 00000146 7F0C ig fin
217 41 00000148 8B0D[0A000000] mov ecx,[B]
218 42 0000014E 8B0D[00000000] mov [max],ecx
219 43 ; ----- Вывод результата
220 44 fin:
221 45 00000154 BA[13000000] mov eax, mae2
222 46 00000159 E8B1FFFFFF call sprintf
223 47 0000015E A3[00000000] mov eax,[max]
224 48 00000163 E81FFFFFFF call printf
225 49 00000168 E86FFFFFFF call quit

```

Рис. 2.11: Файл листинга с ошибкой lab7-2

Объектный файл не смог создаться из-за ошибки. Но получился листинг, где выделено место ошибки.

5. Напишите программу нахождения наименьшей из 3 целочисленных переменных a, b и c. Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 6. Создайте исполняемый файл и проверьте его работу

для варианта 7 - 45,67,15

```
43     call atoi
44     mov [C],eax
45
46     mov ecx,[A]
47     mov [min],ecx
48
49     cmp ecx, [B]
50     jl check_C
51     mov ecx, [B]
52     mov [min], ecx
53
54 check_C:
55     cmp ecx, [C]
56     jl finish
57     mov ecx,[C]
58     mov [min],ecx
59
60 finish:
61     mov eax,answer
62     call sprint
63
64     mov eax, [min]
65     call iprintLF
66
67     call quit
68
69
```

Рис. 2.12: Программа в файле task.asm



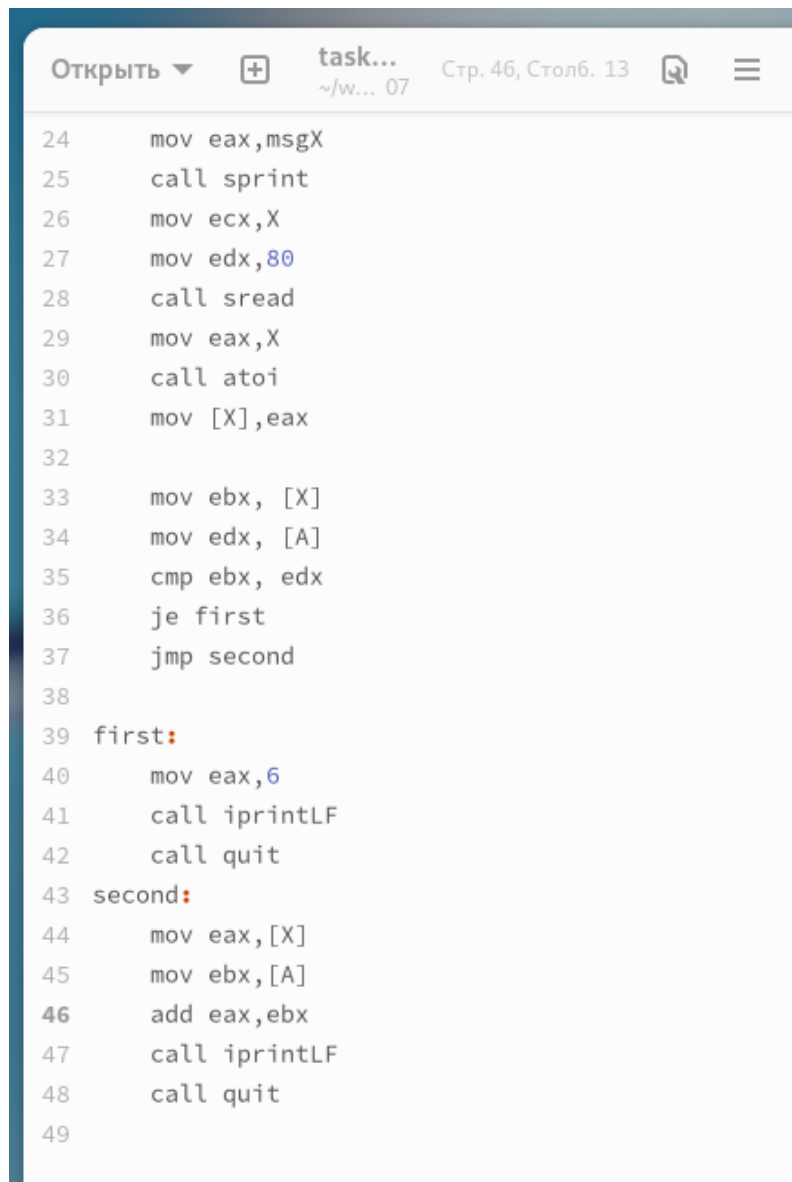
```
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf task.asm  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 task.o -o task  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ./task  
Input A: 45  
Input B: 67  
Input C: 15  
Smallest: 15  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.13: Запуск программы task.asm

6. Напишите программу, которая для введенных с клавиатуры значений  $x$  и  $a$  вычисляет значение заданной функции  $f(x)$  и выводит результат вычислений. Вид функции  $f(x)$  выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений  $x$  и  $a$  из 7.6.

для варианта 7

$$\begin{cases} 6, x = a \\ a + x, x \neq a \end{cases}$$



```
task...
~\w... 07  Стр. 46, Столб. 13

24    mov eax,msgX
25    call sprint
26    mov ecx,X
27    mov edx,80
28    call sread
29    mov eax,X
30    call atoi
31    mov [X],eax
32
33    mov ebx, [X]
34    mov edx, [A]
35    cmp ebx, edx
36    je first
37    jmp second
38
39 first:
40    mov eax,6
41    call iprintLF
42    call quit
43 second:
44    mov eax,[X]
45    mov ebx,[A]
46    add eax,ebx
47    call iprintLF
48    call quit
49
```

Рис. 2.14: Программа в файле task2.asm

```
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf task2.asm  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 task2.o -o task2  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ./task2  
Input A: 1  
Input X: 1  
6  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$ ./task2  
Input A: 1  
Input X: 2  
3  
aidinakmuhammedov@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.15: Запуск программы task2.asm

## 3 Выводы

Изучили команды условного и безусловного переходов, познакомились с фалом листинга.