

# **Отчет по лабораторной работе №7**

**Дисциплина: архитектура компьютера**

Авдеенко Марьяна Дмитриевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Задания самостоятельной работы</b>	<b>11</b>
<b>6</b>	<b>Выводы</b>	<b>13</b>
	<b>Список литературы</b>	<b>14</b>

## Список иллюстраций

4.1	Файл с лимтингом 7.1 . . . . .	8
4.2	Проверка файла . . . . .	8
4.3	Файл с лимтингом 7.2 . . . . .	9
4.4	Проверка файла . . . . .	9
4.5	Файл с лимтингом 7.3 . . . . .	10
4.6	Проверка файла . . . . .	10
5.1	Файл с кодом . . . . .	11
5.2	Проверка файла . . . . .	12

## **Список таблиц**

# 1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

## 2 Задание

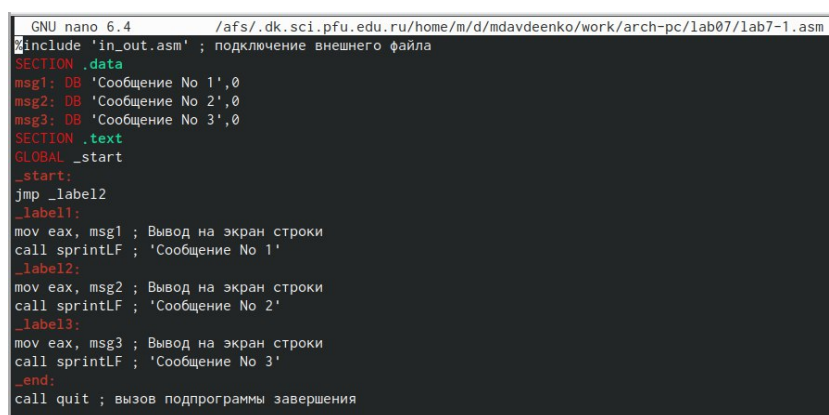
Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

### **3 Теоретическое введение**

Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов: \* условный переход – выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия \* безусловный переход – выполнение передачи управления в определенную точку программы без каких-либо условий.

## 4 Выполнение лабораторной работы

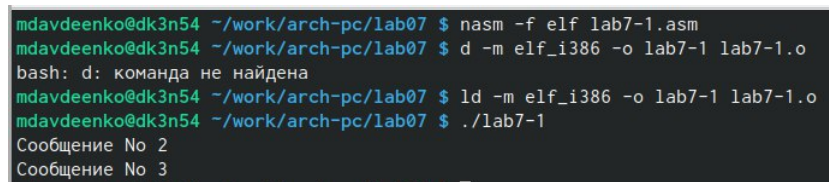
- 1) Открыла терминал.
- 2) Перешла в каталог, созданный для лабораторной работы №7 и создала lab7-1.asm.
- 3) Ввела в файл lab7-1.asm текст программы из листинга 7.1 (рис. 4.1).



```
GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/home/m/d/mdavdeenko/work/arch-pc/lab07/lab7-1.asm
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label12
_label11:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
_label12:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
_label13:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.1: Файл с листингом 7.1

- 4) Создала исполняемый файл и запустила его (рис. 4.2).



```
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
bash: d: команда не найдена
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ./lab7-1
Сообщение No 2
Сообщение No 3
```

Рис. 4.2: Проверка файла



- 5) Изменила программу таким образом, чтобы она выводила сначала ‘Сообщение No 2’, потом ‘Сообщение No 1’ и завершала работу. Изменила текст программы в соответствии с листингом 7.2 (рис. 4.3).

```
GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/home/m/dmdavdeenko/work/arch-pc/lab07/lab7-1.asm
#include "in_out.asm" ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.3: Файл с листингом 7.2

- 6) Создала исполняемый файл и запустила его (рис. 4.4).

```
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ./lab7-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $
```

Рис. 4.4: Проверка файла

- 7) Создала файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. Внимательно изучила текст программы из листинга 7.3 и ввела в lab7-2.asm. (рис. 4.5).

```

GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/home/m/d/mdavdeenko/work/arch-pc/lab07/ab7-2.asm
%include 'in_out.asm'
section .data
msg1 db 'Введите B: ',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 B: '
mov eax,msg1
call sprint
; ----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ----- 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',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:

```

Рис. 4.5: Файл с лимтингом 7.3

- 8) Создайте исполняемый файл и проверьте его работу для разных значений B (рис. 4.6).

```

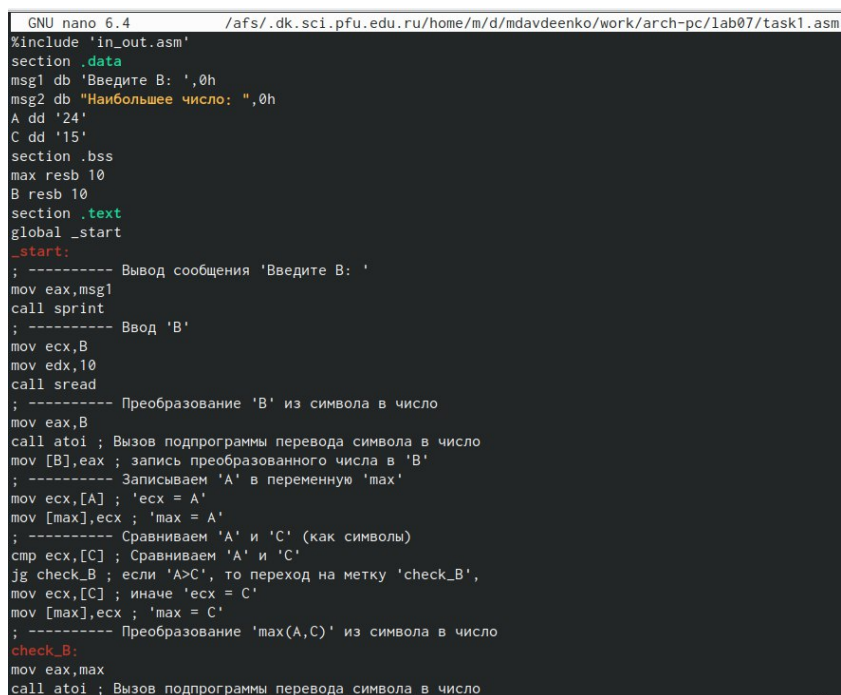
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ nasm -f elf lab7-2.asm
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-2 lab7-2.o
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ./lab7-2
Введите B: 45
Наибольшее число: 50
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $

```

Рис. 4.6: Проверка файла

## 5 Задания самостоятельной работы

- 9) Напишите программу нахождения наименьшей из 3 целочисленных переменных a, b с. Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы №6 (рис. 5.1).



```
GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/home/m/d/mdavdeenko/work/arch-pc/lab07/task1.asm
#include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наибольшее число: ",0h
A dd '24'
C dd '15'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
; ----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
; ----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ----- 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',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
```

Рис. 5.1: Файл с кодом

- 10) Создала исполняемый файл и проверила его работу (рис. 5.2).

```
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ nasm -f elf task1.asm
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o task1 task1.o
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $ ./task1
Введите В: 98
Наибольшее число: 98
mdavdeenko@dk3n54 ~/work/arch-pc/lab07 $
```

Рис. 5.2: Проверка файла

## **6 Выводы**

В ходе данной лабораторной работы были освоены команды условного и безусловного переходов, приобретены навыки написания программ с использованием переходов.

## **Список литературы**