

# **Архитектура компьютера**

**Отчёт по лабораторной работе №7**

Арбатова Варвара Петровна

# 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>Выполнение лабораторной работы</b>	<b>14</b>
<b>6</b>	<b>Выводы</b>	<b>17</b>
	<b>Список литературы</b>	<b>18</b>

## **Список таблиц**

## Список иллюстраций

4.1	Создание каталога и файла . . . . .	8
4.2	Содержимое файла . . . . .	8
4.3	Работа файла . . . . .	9
4.4	текст программы . . . . .	9
4.5	Работа файла . . . . .	9
4.6	Текст программы . . . . .	10
4.7	Работа файла . . . . .	11
4.8	Создание файла . . . . .	11
4.9	Работа файла . . . . .	11
4.10	Создание файла листинга . . . . .	11
4.11	Открытый файл листинга . . . . .	12
4.12	Копирование файла . . . . .	12
4.13	Измененный текст программы . . . . .	12
4.14	Созданные файлы . . . . .	13
4.15	Файл листинга . . . . .	13
5.1	Создание файла . . . . .	14
5.2	Работа файла . . . . .	14
5.3	Создание файла . . . . .	15
5.4	Текст файла . . . . .	15

# 1 Цель работы

Изучить команды условного и безусловного переходов. Приобрести навыки написания программ с использованием переходов. Познакомиться с назначением и структурой файла листинга.

## 2 Задание

1. Создайте каталог для программ лабораторной работы № 7, перейдите в него и создайте файл lab7-1.asm
2. Инструкция `jmp` в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции `jmp`. Введите в файл lab7-1.asm текст программы из листинга 7.1.
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.
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. Создайте файл листинга для программы из файла lab7-2.asm

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

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

Рис. 4.1: Создание каталога и файла

- 2) Ввожу в файл lab7-1.asm текст программы из листинга 7.1.

```
/home/vparbatova/work/arch-pc/lab07/lab7-1.asm [----] 41
#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'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение "% 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение "% 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.2: Содержимое файла



3) Создаю исполняемый файл и запускаю его

```
vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-1
Сообщение "% 2
Сообщение "% 3
```

Рис. 4.3: Работа файла

4) Изменяю текст программы в соответствии с листингом 7.2

```
/home/vparbatova/work/arch-pc/lab07/lab7-1.asm [-
#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 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'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.4: текст программы

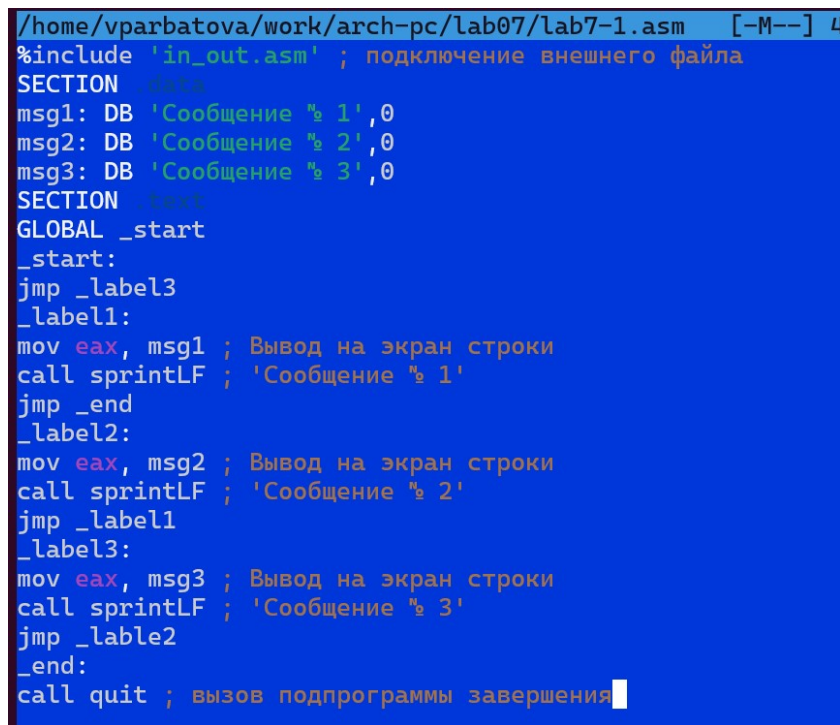
5) Создаю исполняемый файл и запускаю его

```
vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-1
Сообщение "% 2
Сообщение "% 1
```

Рис. 4.5: Работа файла

- 6) Изменяю текст программы изменив инструкции `jmp`, чтобы вывод программы был следующим: Сообщение № 3 Сообщение № 2 Сообщение № 1

`%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 _label3`  
`_label1: mov eax, msg1 ; Вывод на экран строки`  
`call sprintfLF ; 'Сообщение № 1'`  
`jmp _end`  
`_label2: mov eax, msg2 ; Вывод на экран строки`  
`call sprintfLF ; 'Сообщение № 2'`  
`jmp _label1`  
`_label3: mov eax, msg3 ; Вывод на экран строки`  
`call sprintfLF ; 'Сообщение № 3'`  
`jmp _label2`  
`_end: call quit ; вызов подпрограммы завершения`



```
/home/vparbatova/work/arch-pc/lab07/lab7-1.asm [-M--] 4
#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 _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.6: Текст программы

- 7) Создаю исполняемый файл и запускаю его

```

vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-1
Сообщение %s 3
Сообщение %s 2
Сообщение %s 1

```

Рис. 4.7: Работа файла

8) Создаю файл lab7-2.asm и проверяю его создание

```

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

```

Рис. 4.8: Создание файла

9) Ввожу в файл текст листинга 7.3, создаю файл и запускаю его

```

vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 35
Наибольшее число: 50
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 70
Наибольшее число: 70
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 30
Наибольшее число: 50
vparbatova@Varvarishe:~/work/arch-pc/lab07$

```

Рис. 4.9: Работа файла

10) Создаю файл листинга для программы из файла lab7-2.asm и открываю его в текстовом редакторе

```

vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ mcedit lab7-2.lst

```

Рис. 4.10: Создание файла листинга

11) Открытый файл листинга

```

1      ;----- slen -----
2      <1> ; функция вычисления длины сообщения
3      <1> slen:
4      <1> push ebx
5      00000000 53      <1> mov ebx, eax
6      00000001 89C3    <1>
7      <1>
8      <1> nextchar:
9      00000003 803000    <1> cmp byte [eax], 0
10     00000006 7403     <1> jz finished
11     00000008 40      <1> inc eax
12     00000009 EBF8     <1> jmp nextchar
13     <1>
14     <1> finished:
15     0000000B 29D0     <1> sub eax, ebx
16     0000000D 5B      <1> pop ebx
17     0000000E C3      <1> ret
18     <1>
19     <1>
20     <1> ;----- sprint -----
21     <1> ; функция печати сообщения
22     <1> ; входные данные: mov eax, <message>
23     <1> sprint:
24     0000000F 52      <1> push edx
25     00000010 51      <1> push ecx
26     00000011 53      <1> push ebx
27     00000012 50      <1> push eax
28     00000013 E8E8FFFFFF <1> call slen
29     <1>
30     00000018 89C2     <1> mov edx, eax
31     0000001A 50      <1> pop eax
32     <1>
33     0000001B 89C1     <1> mov ecx, eax

```

Рис. 4.11: Открытый файл листинга

17 000000F2 B9[0A000000] mov ecx,B (17 - номер строки, 000000F2 - адрес, B9 - машинный код, [0A000000] - исходный текст программы) 18 000000F7 BA0A000000 mov edx,10 (18 - номер строки, 000000F7 - адрес, BA - машинный код, 0A000000 - исходный текст программы) 19 000000FC E842FFFFFF call sread (19 - номер строки, 000000FC - адрес, E8 - машинный код, 42FFFFFF - исходный текст программы)

12) Копирую файл lab7-2.asm как lab7-2-2.asm и открываю его

```

vparbatova@Varvarishe:~/work/arch-pc/lab07$ cp lab7-2.asm lab7-2-2.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ mcedit lab7-2-2.asm

```

Рис. 4.12: Копирование файла

13) Удаляю один из операндов

```

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

```

Рис. 4.13: Измененный текст программы

14) Создаю файл листинга

```

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

```

Рис. 4.14: Созданные файлы

## 15) Файл листинга пустой

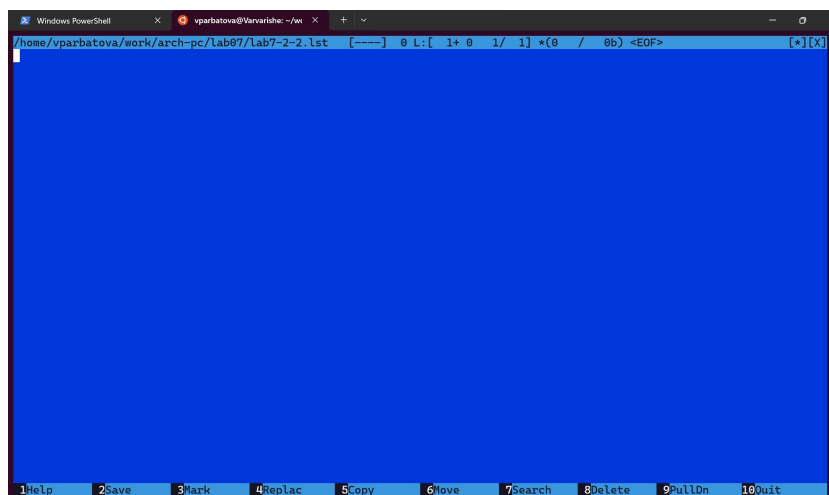


Рис. 4.15: Файл листинга

## 5 Выполнение лабораторной работы

1) Создаю файл для написания программы

```
vparbatova@Varvarishe:~/work/arch-pc/lab07$ touch lab7-4.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ mcedit lab7-4.asm
```

Рис. 5.1: Создание файла

2) Создание и работа файла. У меня вариант 1

```
vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-4
Наименьшее число: 17
```

Рис. 5.2: Работа файла

Текст файла:

```
%include 'in_out.asm' section .data msg2 db "Наименьшее число:",0h A dd 17
C dd 23 B dd 45 section .bss min resb 10 section .text global _start _start: ; ----
Записываем 'A' в переменную 'min' mov ecx,[A] ; 'ecx = A' mov [min],ecx ; 'min =
A' ; ---- Сравниваем 'A' и 'C' (как символы) cmp ecx,[C] ; Сравниваем 'A' и 'C'
jl check_B ; если 'A<C', то переход на метку 'check_B', mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'min = C' ; ---- Преобразование 'max(A,C)' из символа в число
check_B: ; ---- Сравниваем 'min(A,C)' и 'B' (как числа) mov ecx,[min] cmp ecx,[B] ;
Сравниваем 'min(A,C)' и 'B' jl fin ; если 'min(A,C)<B', то переход на 'fin', mov ecx,[B]
; иначе 'ecx = B' mov [min],ecx ; ---- Вывод результата fin: mov eax, msg2 call
```

sprint ; Вывод сообщения 'Наибольшее число:' mov eax,[min] call iprintLF ; Вывод 'min(A,B,C)' call quit ; Выход

3) Создаю файл для второго задания

```
vparbatova@Varvarishe:~/work/arch-pc/lab07$ touch lab7-3.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ls
in_out.asm lab7-1.asm lab7-2 lab7-2.asm lab7-2.o lab7-4 lab7-4.o
lab7-1 lab7-1.o lab7-2-2.asm lab7-2.lst lab7-3.asm lab7-4.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$
```

Рис. 5.3: Создание файла

4) Текст файла

```
/home/vparbatova/work/arch-pc/lab07/lab7-3.asm [----] 22 L: [ 25+31 56/ 57] *(1515/1538b) 001
mov eax,msg2
call sprint
; ----- Ввод 'a'
mov ecx,a
mov edx,10
call sread
; ----- Преобразование 'a' из символа в число
mov eax,a
call atoi ; Вызов подпрограммы перевода символа в число
mov [a],eax ; запись преобразованного числа в 'a'
mov ecx, [a] ; ecx = a

cmp ecx,[x] ; Сравниваем 'a' и 'x'
jg ysl1 ; если 'a>x', то переход на метку 'ysl1',

ysl2:
mov ecx,8
mov [rez],ecx
jmp fin

ysl1:
mov eax,[a]; eax = a
mov ebx,2; ebx = 2
mul ebx; eax = 2*a
sub eax,[x]; eax = 2*a - x
mov [rez],eax; rez = eax
; ----- Вывод результата
fin:
mov eax, msg3
call sprint ; Вывод сообщения 'Наибольшее число: '
mov eax,[rez]
call iprintLF ; Вывод
call quit ; Выход
```

Рис. 5.4: Текст файла

Текст файла:

```
%include 'in_out.asm' section .data msg1 db 'Введите x:',0h msg2 db 'Введите a:',0h
msg3 db "Результат:",0h section .bss a resb 10 x resb 10 rez resb 10 section .text global
_start _start: ; ---- Вывод сообщения 'Введите x:' mov eax,msg1 call sprint ; ----
Ввод 'x' mov ecx,x mov edx,10 call sread ; ---- Преобразование 'x' из символа в
```

число `mov eax,x call atoi` ; Вызов подпрограммы перевода символа в число `mov [x],eax` ; запись преобразованного числа в 'x' ; ——— Вывод сообщения 'Введите a:' `mov eax,msg2 call sprint` ; ——— Ввод 'a' `mov ecx,a mov edx,10 call sread` ; ——— Преобразование 'a' из символа в число `mov eax,a call atoi` ; Вызов подпрограммы перевода символа в число `mov [a],eax` ; запись преобразованного числа в 'a' `mov ecx, [a]` ; `ecx = a`

`cmp ecx,[x]` ; Сравниваем 'a' и 'x' `jg ysl1` ; если 'a>x', то переход на метку 'ysl1',  
`ysl2: mov ecx,8 mov [rez],ecx jmp fin`

`ysl1: mov eax,[a]`; `eax = a` `mov ebx,2`; `ebx = 2` `mul ebx`; `eax = 2a` `sub eax,[x]`; `eax = 2a - x` `mov [rez],eax`; `rez = eax` ; ——— Вывод результата `fin: mov eax, msg3 call sprint` ;  
Вывод сообщения 'Результат:' `mov eax,[rez] call iprintLF` ; Вывод `call quit` ; Выход

```
vparbatova@Varvarishe:~/work/arch-pc/lab07$ mcedit lab7-3.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-3
Введите x: 1
Введите a: 2
Результат: 3
vparbatova@Varvarishe:~/work/arch-pc/lab07$ ./lab7-3
Введите x: 2
Введите a: 1
Результат: 8
```

## 5) Работа файла



## 6 Выводы

Мною изучены команды условного и безусловного переходов, приобретены навыки написания программ с использованием переходов, я ознакомилась с назначением и структурой файла листинга.

## **Список литературы**