

Лабораторная работа №7

Архитектура компьютера

Башиянц Александра Кареновна

Содержание

1	Цель работы	3
2	Задание	4
3	Выполнение лабораторной работы	5
3.1	Реализация переходов в NASM	5
3.2	Изучение структуры файлы листинга	8
3.3	Задание для самостоятельной работы	9
3.3.1	Задание 1	9
3.3.2	Задание 2	10
4	Выводы	12

1 Цель работы

Цель работы — изучить команды условного и безусловного переходов.

2 Задание

В этой лабораторной работе необходимо изучить работу условного и безусловного переходов в NASM.

Необходимо научиться:

- Изучить команды условного и безусловного переходов;
- Приобрести навыки написания программ с использованием переходов;
- Узнать назначение и структуру файла листинга.

Выполняя это задание, мы получим практический опыт работы условного и безусловного переходов в NASM.

3 Выполнение лабораторной работы

3.1 Реализация переходов в NASM

Создадим директорию для 7 лабораторной работы и создадим файл lab7-1.asm (рис. 3.1).

```
akbashiyc@fedora1:~$ mkdir ~/work/arch-pc/lab07
akbashiyc@fedora1:~$ cd ~/work/arch-pc/lab07
akbashiyc@fedora1:~/work/arch-pc/lab07$ touch lab7-1.asm
akbashiyc@fedora1:~/work/arch-pc/lab07$ ls
lab7-1.asm
```

Рис. 3.1: Создание директории

Скопируем файл in_out.asm из lab06 с помощью mc (рис. 3.2).

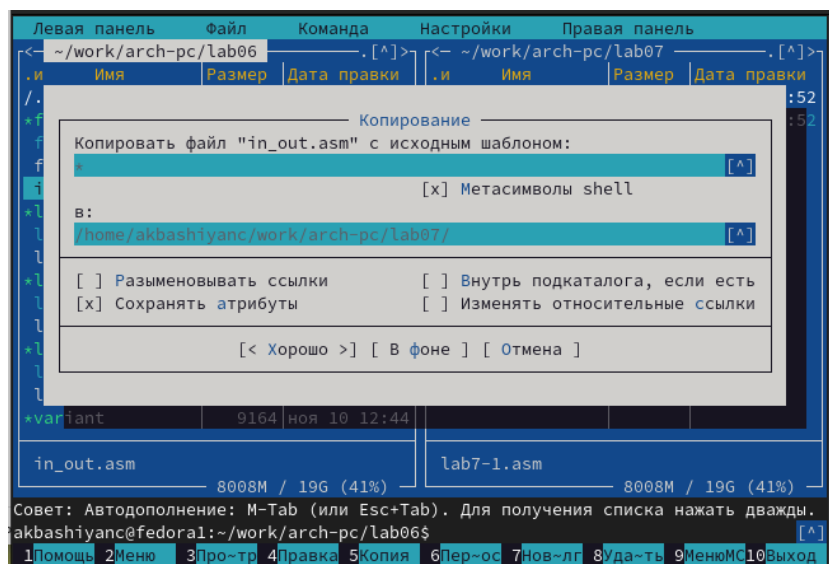


Рис. 3.2: Копирование in_out.asm

Введем код в lab7-1.asm и создадим исполняемый файл и запустим его (рис. 3.3).

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

Рис. 3.3: Запуск файла

Изменим код так, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу (рис. 3.4).

```
_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
```

Рис. 3.4: Изменение файла

Создадим исполняемый файл и запустим его (рис. 3.5).

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

Рис. 3.5: Запуск файла

Изменим код так, чтобы она выводила сначала ‘Сообщение № 3’, потом ‘Сообщение № 2’ и ‘Сообщение № 1’ и завершала работу (рис. 3.6).

```
GLOBAL _start
_start:
<----->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:
<----->call quit
```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход

Рис. 3.6: Изменение файла

Создадим исполняемый файл и запустим его (рис. 3.7).

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

Рис. 3.7: Запуск файла

Создадим файл lab7-1.asm (рис. 3.8).

```
akbashiyanc@fedoral:~/work/arch-pc/lab07$ touch lab7-2.asm
```

Рис. 3.8: Создание файл

Введем код в lab7-1.asm (рис. 3.9).

```
lab7-2.asm [----] 21 L: [ 1+ 0 1/ 49] *(21 /1879b) 0010 0x00A [*][X]
#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
```

Рис. 3.9: Ввод кода

Создадим исполняемый файл и запустим его (рис. 3.10).

```

akbashiyan@fedora1:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
akbashiyan@fedora1:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
akbashiyan@fedora1:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 1
Наибольшее число: 50
akbashiyan@fedora1:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 30
Наибольшее число: 50
akbashiyan@fedora1:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 100
Наибольшее число: 100

```

Рис. 3.10: Запуск файла

3.2 Изучение структуры файлы листинга

Выполним команду `nasm` с ключом `-l` (рис. 3.11).

```

akbashiyan@fedora1:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
akbashiyan@fedora1:~/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. 3.11: Выполнение команды `nasm` с ключом `-l`

Откроем созданный файл (рис. 3.12).

```

lab7-2.lst  [-----]  0 L: [ 1+ 0 1/225] *(0 /14594b) 0032 0x020 [*][X]
1          %include 'in_out.asm'
1          <1> ;----- slen -----
2          <1> ; Функция вычисления длины сообщения
3          <1> slen:.....
4 00000000 53          <1> push ebx.....
5 00000001 89C3        <1> mov ebx, eax.....
6          <1>.....
7          <1> nextchar:

```

Рис. 3.12: Просмотр файла `.lst`

Удалим в файле `.asm` одну строчку и посмтрим разницу медлу первым и вторым файлом `.lst` (рис. 3.13, 3.14, 3.15)

```

12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15
16 ; ----- Ввод 'B'
17 mov ecx,B

```

Рис. 3.13: Просмотр файла `.lst`


```
akbashiyan@fedora1:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
```

Рис. 3.14: Выполнение команды nasm

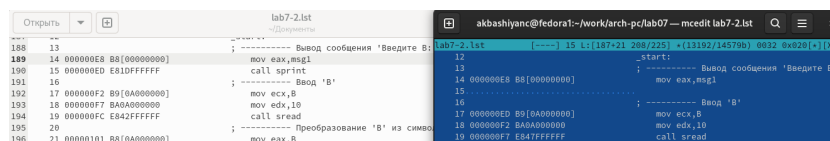


Рис. 3.15: Просмотр файла .lst

Рассмотрим 3 строки: 1. 16 0000000E C3 <1> ret

2. 23 0000000F 52 <1> push edx

3. 118 00000094 E876FFFFFF <1> call sprint

- 16, 23, 118 - номера строк файла листинга
- 0000000E, 0000000F, 00000094 - смещение машинного кода от начала текущего сегмента
- C3, 52, E876FFFFFF - ассемблированная исходная строка в виде шестнадцатеричной последовательности
- ret, push edx, call sprint - исходный текст программы
- ret - команда “return”. Эта команда используется для возврата из подпрограммы или функции обратно в вызывающий код
- push edx - перемещение содержимого регистра edx на вершину стека
- call sprint - вызов функции печати сообщения.

3.3 Задание для самостоятельной работы

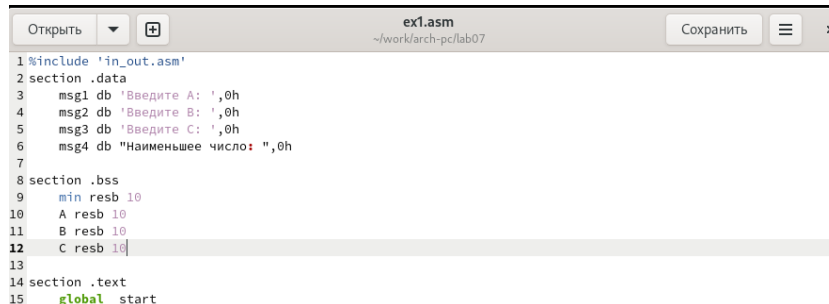
3.3.1 Задание 1

Создадим файл ex1.asm (рис. 3.16).

```
akbashiyanc@fedoral1:~/work/arch-pc/lab07$ touch ex1.asm
```

Рис. 3.16: Создание файл

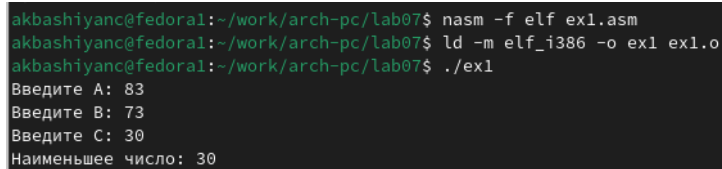
Введем код для нахождения минимального значения в ex1.asm (рис. 3.17).



```
1 %include 'in_out.asm'
2 section .data
3     msg1 db 'Введите A: ',0h
4     msg2 db 'Введите B: ',0h
5     msg3 db 'Введите C: ',0h
6     msg4 db "Наименьшее число: ",0h
7
8 section .bss
9     min resb 10
10    A resb 10
11    B resb 10
12    C resb 10
13
14 section .text
15     global _start
```

Рис. 3.17: Ввод кода

Создадим исполняемый файл и запустим его, подставив для проверки значения 83,73,30 (вариант 18) (рис. 3.18).



```
akbashiyanc@fedoral1:~/work/arch-pc/lab07$ nasm -f elf ex1.asm
akbashiyanc@fedoral1:~/work/arch-pc/lab07$ ld -m elf_i386 -o ex1 ex1.o
akbashiyanc@fedoral1:~/work/arch-pc/lab07$ ./ex1
Введите A: 83
Введите B: 73
Введите C: 30
Наименьшее число: 30
```

Рис. 3.18: Запуск файла

3.3.2 Задание 2

Создадим файл ex2.asm (рис. 3.19).

```
akbashiyanc@fedoral1:~/work/arch-pc/lab07$ touch ex2.asm
```

Рис. 3.19: Создание файл

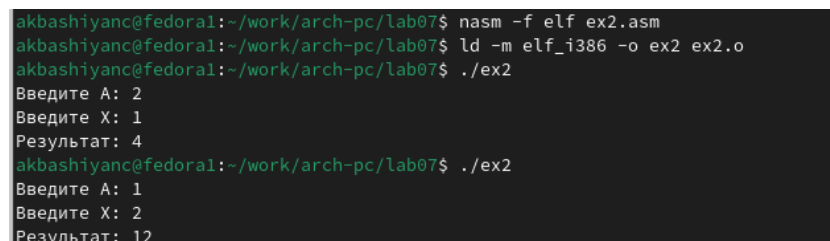
Введем код для нахождения функции $f(x)$ (вариант 18) в ex2.asm (рис. 3.20).



```
1 %include 'in_out.asm' ; подключение внешнего файла
2 section .data
3     msg1 db "Введите A: ",0h
4     msg2 db "Введите X: ",0h
5     msg3 db "Результат: ",0h
6
7 section .bss
8     fin resb 10
9     A resb 10
10    X resb 10
11
```

Рис. 3.20: Ввод кода

Создадим исполняемый файл и запустим его, подставив для проверки значения (1,2) и (2,1) (вариант 18) (рис. 3.21).



```
akbashiyanc@fedora1:~/work/arch-pc/lab07$ nasm -f elf ex2.asm
akbashiyanc@fedora1:~/work/arch-pc/lab07$ ld -m elf_i386 -o ex2 ex2.o
akbashiyanc@fedora1:~/work/arch-pc/lab07$ ./ex2
Введите A: 2
Введите X: 1
Результат: 4
akbashiyanc@fedora1:~/work/arch-pc/lab07$ ./ex2
Введите A: 1
Введите X: 2
Результат: 12
```

Рис. 3.21: Запуск файла

4 Выводы

В ходе выполнения работы были получены навыки практической работы с работой условного и безусловного переходов в NASM.