

Лабораторная работа №7. Команды безусловного и условного переходов в Nasm. Программирование ветвлений

Аджигалиева Амина Руслановна

Содержание

1	Цель работы	4
2	Порядок выполнения лабораторной работы	5
2.1	Реализация переходов в NASM	5
2.2	Изучение структуры файлы листинга	11
2.3	Задание для самостоятельной работы	13
3	Выводы	18

Список иллюстраций

2.1	Новый каталог	5
2.2	Текст программы	6
2.3	Запуск программы	6
2.4	Меняем программу	7
2.5	Запуск программы	7
2.6	Замена текста	8
2.7	Запуск программы	9
2.8	Новый файл	9
2.9	Текст программы	10
2.10	Запуск программы	11
2.11	Файл листинга	11
2.12	Откроем файл листинга	12
2.13	Трансляция	12
2.14	Новый файл	13
2.15	Программа	14
2.16	Запуск программы	15
2.17	Новый файл	15
2.18	Программа 2	16
2.19	Запуск программы	17
2.20	Запуск программы	17

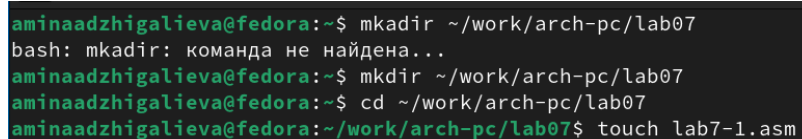
1 Цель работы

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

2 Порядок выполнения лабораторной работы

2.1 Реализация переходов в NASM

Создаем каталог для программ лабораторной работы № 7, перейдем в него и создаем файл lab7-1.asm: (рис. 2.1).



```
aminaadzhigalieva@fedora:~$ mkdir ~/work/arch-pc/lab07
bash: mkdir: команда не найдена...
aminaadzhigalieva@fedora:~$ mkdir ~/work/arch-pc/lab07
aminaadzhigalieva@fedora:~$ cd ~/work/arch-pc/lab07
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ touch lab7-1.asm
```

Рис. 2.1: Новый каталог

Введем в файл lab7-1.asm текст программы из листинга. (рис. 2.2).

```

%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 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'

_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'

_end:
call quit

```

Рис. 2.2: Текст программы

Создадим исполняемый файл и запустим его. (рис. 2.3).

```

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

```

Рис. 2.3: Запуск программы

Изменим программу таким образом, чтобы она выводила сначала 'Сообщение № 2', потом 'Сообщение № 1' и завершала работу (рис. 2.4).

```

#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 ; 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

```

Рис. 2.4: Меняем программу

Создадим исполняемый файл и запустим его. (рис. 2.5).

```

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

```

Рис. 2.5: Запуск программы

Изменим текст программы в соответствии с листингом 7.2. (рис. 2.6).

```
%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 ; 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
```

Рис. 2.6: Замена текста

Создадим исполняемый файл и запустим его. (рис. 2.7).


```

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

```

Рис. 2.7: Запуск программы

Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. (рис. 2.8).

```

aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ touch lab7-2.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$

```

Рис. 2.8: Новый файл

Запишем текст листинга в файл (рис. 2.9).

```

#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:
mov eax,msg1
call sprint
mov ecx,B
mov edx,10
call sread
mov eax,B
call atoi
mov [B],eax
mov ecx,[A]
mov [max],ecx
check_B:
mov eax,max
call atoi
mov [max],eax
mov ecx,[max]
cmp ecx,[B]
jg fin
mov ecx,[B]
mov [max],ecx
fin:
mov eax, msg2
call sprint
mov eax,[max]
call iprintLF
call quit

```

Рис. 2.9: Текст программы

Создадим исполняемый файл и запустим его. (рис. 2.10).

```
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2
.o
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 15
Наибольшее число: 20
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2
.o
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 25
Наибольшее число: 25
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$
```

Рис. 2.10: Запуск программы

2.2 Изучение структуры файлы листинга

Создаем файл листинга для программы из файла lab7-2.asm (рис. 2.11).

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

Рис. 2.11: Файл листинга

Откроем файл листинга lab7-2.lst с помощью любого текстового редактора, например mcedit: (рис. 2.12).

```

lab7-2.lst      [-M--]  0 L: 1+ 0 1/217] *(0 /13626b) 0032 0x0[*][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:.....
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>
22     <1> sprint:
23     0000000F 52      <1> push edx
24     00000010 51      <1> push ecx
25     00000011 53      <1> push ebx
26     00000012 50      <1> push eax
27     00000013 E8E8FFFF <1> call slen
28     <1>.....
29     00000018 89C2    <1> mov edx, eax
30     0000001A 58      <1> pop eax

```

1По-щ 2Со-ан 3Блок 4Замена 5Копия 6Пе-ть 7Поиск 8Уда-ть 9МенюМС10Выход

Рис. 2.12: Откроем файл листинга

Строка 29: 00000018 - адрес в сегменте кода, 89C2 - машинный код, mov edx, eax - перемещает содержимое регистра eax в регистр edx

Строк 30: 0000001A - адрес в сегменте кода, 58 - машинный код, pop eax - восстанавливает значение из стека в указанный регистр eax

Строка 32: 0000001B - адрес в сегменте кода, 89C1 - машинный код, mov ecx, eax - перемещает содержимое регистра eax в регистр ecx

Откроем файл с программой lab7-2.asm и в любой инструкции с двумя операндами удалим один операнд. Выполним трансляцию с получением файла листинга: (рис. 2.13).

```

aminaadzhigalieva@fedora:~/work/arch-pc/Lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:16: error: invalid combination of opcode and operands
aminaadzhigalieva@fedora:~/work/arch-pc/Lab07$ ls
in_out.asm lab7-1 lab7-1.asm lab7-1.o lab7-2 lab7-2.asm lab7-2.lst
aminaadzhigalieva@fedora:~/work/arch-pc/Lab07$

```

Рис. 2.13: Трансляция

2.3 Задание для самостоятельной работы

Вариант 4.

Создаем файл: (рис. 2.14).



```
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ touch lab7-3.asm
```

Рис. 2.14: Новый файл

Открываем его и пишем программу, которая выберет наименьшее число из трех (рис. 2.15).

```

#include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наименьшее число: ",0h
A dd '8'
C dd '88'
section .bss
min resb 10
B resb 10
section .text
global _start
_start:
mov eax, msg1
call sprint
mov ecx, B
mov edx, 10
call sread
mov eax, B
call atoi
mov [B], eax
mov ecx, [A]
mov [min], ecx
cmp ecx, [C]
jl check_B
mov ecx, [C]
mov [min], ecx
check_B:
mov eax, min
call atoi
mov [min], eax
mov ecx, [min]
cmp ecx, [B]
jl fin
mov ecx, [B]
mov [min], ecx
fin:
mov eax, msg2
call sprint
mov eax, [min]
call iprintLF
call quit

```

Рис. 2.15: Программа

Создадим исполняемый файл и запустим его. (рис. 2.16).

```
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3
.o
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ./lab7-3
Введите В: 68
Наименьшее число: 8
```

Рис. 2.16: Запуск программы

Создаем файл: (рис. 2.17).

```
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ touch lab7-4.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$
```

Рис. 2.17: Новый файл

Открываем его и пишем программу, которая решит систему уравнений: (рис. 2.18).

```

#include 'in_out.asm'
SECTION .data
msg1: DB 'Введите x: ', 0h
msg2: DB 'Введите a: ', 0h
otv: DB 'F(x) = ', 0h
SECTION .bss
x: RESB 80
a: RESB 80
res: RESB 80
SECTION .text
GLOBAL _start
_start:
    mov eax, msg1
    call sprint
    mov ecx, x
    mov edx, 80
    call sread
    mov eax, x
    call atoi
    mov [x], eax
    mov eax, msg2
    call sprint
    mov ecx, a
    mov edx, 80
    call sread
    mov eax, a
    call atoi
    mov [a], eax
    mov eax, [a]
    cmp eax, 0
    je nol_a
calc_a:
    mov eax, [x]
    shl eax, 1
    add eax, [a]
    mov [res], eax
    jmp fin
nol_a:
    mov eax, [x]
    shl eax, 1
    add eax, 1
    mov [res], eax
fin:
    mov eax, otv
    call sprint
    mov eax, [res]
    call iprintLF
    call quit

```

Рис. 2.18: Программа 2

Создадим исполняемый файл и запустим его для (3;0) (рис. 2.19).

```
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4
.o
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 0
F(x) = 7
```

Рис. 2.19: Запуск программы

Создадим исполняемый файл и запустим его для (3;2) (рис. 2.20).

```
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4
.o
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 2
F(x) = 8
aminaadzhigalieva@fedora:~/work/arch-pc/lab07$
```

Рис. 2.20: Запуск программы

3 Выводы

Мы познакомились с структурой файла листинга, изучили команды условного и безусловного перехода.