

Лабораторная работа №8

**Команды безусловного и условного переходов в Nasm.
Программирование ветвлений.**

Киньябаева Аиша Иделевна

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Выводы	15

Список иллюстраций

3.1	код	6
3.2	lab8-1.asm_вывод	6
3.3	код	7
3.4	lab8-1.asm_вывод	7
3.5	код	8
3.6	lab8-1.asm_вывод	8
3.7	lab8-2.asm_вывод	8
3.8	фрагмент листинга	9
3.9	код с ошибкой	10
3.10	ошибка в терминале	10
3.11	ошибка в листинге	10
3.12	фрагмент листинга	11
3.13	код	11
3.14	продолжение кода	12
3.15	наименьшее число_вывод	12
3.16	код	13
3.17	продолжение кода	14
3.18	lab7-3.asm_вывод	14

1 Цель работы

Целью данной работы является изучение команд условного и безусловного переходов, приобретение навыков написания программ с использованием переходов, знакомство с назначением и структурой файла листинга

2 Задание

Освоение команд сравнения значений и написание программ, связанных с этим, изучение структуры листинга.

3 Выполнение лабораторной работы

Написание первой программы lab8-1.asm, которая показывает переходы в программе с использованием команды jmp. В результате мы получаем “Сообщение №2”, “Сообщение №3” (рис. 3.1), (рис. 3.2)

```
%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 sprintf

_label2:
mov eax, msg2
call sprintf

_label3:
mov eax, msg3
call sprintf

_end:
call quit
~
```

Рис. 3.1: код

```
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ vim lab8-1.asm
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-1
Сообщение №2
Сообщение №3
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 3.2: lab8-1.asm_вывод

Далее мы преобразовываем программу, чтобы она выводила “Сообщение №2”, “Сообщение №1” (рис. [fig:fig3]), (рис. 3.4)

```
%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
call sprintfLF

jmp _end

_label2:
mov eax, msg2
call sprintfLF

jmp _label1

_label3:
mov eax, msg3
call sprintfLF

_end:
call quit
~
```

Рис. 3.3: код

```
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ vim lab8-1.asm
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-1
Сообщение №2
Сообщение №1
atkinjyabaeva@atkinjyabaeva-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 3.4: lab8-1.asm_вывод

Далее преобразовываю программу так, чтобы она выводила сообщения в обратном порядке. (рис. 3.5), (рис. 3.6)

```

#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
call sprintLF

jmp _end

_label2:
mov eax, msg2
call sprintLF

jmp _label1

_label3:
mov eax, msg3
call sprintLF

jmp _label2

_end:
call quit
~

```

Рис. 3.5: код

```

aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ vim lab8-1.asm
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ nasm -f elf lab8-1.asm
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ ./lab8-1
Сообщение №3
Сообщение №2
Сообщение №1
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$

```

Рис. 3.6: lab8-1.asm_вывод

Создаем новый файл lab8-2.asm, который находит наибольшее значение из двух данных и третьего введенного.(рис. 3.7)

```

aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ touch lab8-2.asm
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ vim lab8-2.asm
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ nasm -f elf lab8-2.asm
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ ./lab8-2
Введите В: 30
Наибольшее число: 50
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$ ./lab8-2
Введите В: 60
Наибольшее число: 60
aikinjjabaeva@aikinjjabaeva-VirtualBox:~/work/arch-pc/Lab08$

```

Рис. 3.7: lab8-2.asm_вывод

Знакомство с листингом. Создаю файл листинга и изучаю его. На картинке представлен фрагмент (рис. 3.8)

```

172 | 2
173 | 3
174 | 4 00000000 D092D0B2D0B5D0B4D0- section .data
175 | 4 00000009 B8D182D0B520D0923A- msg1 db 'Введите B: ',0h
176 | 4 00000012 2000
177 | 5 00000014 D09DD0B0D0B8D0B1D0- msg2 db "Наибольшее число: ",0h
178 | 5 0000001D BED0BBD18CD188D0B5-
179 | 5 00000026 D0B520D187D0B8D181-
180 | 5 0000002F D0BBD0BE3A2000
181 | 6 00000036 32300000 A dd '20'
182 | 7 0000003A 35300000 C dd '50'
183 | 8
184 | 9
185 | 10 00000000 <res Ah> section .bss
186 | 11 0000000A <res Ah> max resb 10
187 | 12 B resb 10
188 | 13
189 | 14 section .text
190 | 15 global _start
191 | 16 _start:
192 | 17 000000E8 B8[00000000] mov eax, msg1
193 | 18 000000ED E81DFFFFFF call sprint
194 | 19

```

Рис. 3.8: фрагмент листинга

Поменяв код lab8-2.asm, вижу как терминал выдает ошибку, а при просмотре листингового файла можно также заметить ошибку (рис. 3.9), (рис. 3.10), (рис. 3.11)

```

ASM lab8-2.asm
1  %include 'in_out.asm'
2
3  section .data
4  msg1 db 'Введите B: ',0h
5  msg2 db "Наибольшее число: ",0h
6  A dd '20'
7  C dd '50'
8
9  section .bss
10 max resb 10
11 B resb 10
12
13 section .text
14 global _start
15 _start:
16
17 mov eax|
18 call sprint
19
20 mov ecx,B
21 mov edx,10
22 call sread
23

```

Рис. 3.9: код с ошибкой

```

aikinjyabaeva@aikinjyabaeva-VirtualBo
x:~/work/arch-pc/lab08$ nasm -f elf -
l lab8-2.lst lab8-2.asm
lab8-2.asm:17: error: invalid combina
tion of opcode and operands
aikinjyabaeva@aikinjyabaeva-VirtualBo
x:~/work/arch-pc/lab08$ 

```

Рис. 3.10: ошибка в терминале

```

17                                     mov eax
17      *****                      error: invalid combination of opcode and operands
18 000000E8 E822FFFFFF                call sprint
19

```

Рис. 3.11: ошибка в листинге

Ниже представлен еще один фрагмент листинга. В нем видим:(рис. 3.12) 1. 41 - номер строки 0000013A - адрес 8B0D[00000000] - машинный код `mov ecx,[max]` - исходный текст программы (в данном случае значащее `ecx = max`) 2. 42 - номер строки 00000140 - адрес 3B0D[0A000000] - машинный код `str ecx,[B]` - исходный

текст программы (в данном случае значащее сравнение значения есх с В) 3. 43
- номер строки 00000146 - адрес 7F0C - машинный код jg fin - исходный текст
программы (в данном случае значащее если max>B, то переход к fin)

```
41 0000013A 8B0D[00000000]      mov ecx,[max]
42 00000140 3B0D[0A000000]      cmp ecx,[B]
43 00000146 7F0C              jg fin
```

Рис. 3.12: фрагмент листинга

САМОСТОЯТЕЛЬНАЯ РАБОТА

Из прошлой лабораторной работы мне попался 4й вариант.

Для первого задания пишу программу, находящую наименьшее число из 3х введенных (рис. 3.13), (рис. 3.14), (рис. 3.15)

```
1  %include 'in_out.asm'
2
3  section .data
4
5  msg1 db 'Введите A: ',0h
6  msg2 db 'Введите B: ',0h
7  msg3 db 'Введите C: ',0h
8
9  msg4 db "Наименьшее число: ",0h
10
11 section .bss
12 min resb 10
13 A resb 10
14 B resb 10
15 C resb 10
16
17 section .text
18 global _start
19 _start:
20
21 mov eax,msg1
22 call sprint
23
24 mov ecx,A
25 mov edx,10
26 call sread
27
28 mov eax,msg2
29 call sprint
30
31 mov ecx,B
32 mov edx,10
33 call sread
34
35 mov eax,msg3
36 call sprint
37
38 mov ecx,C
39 mov edx,10
40 call sread
```

Рис. 3.13: код

```

42  mov eax,A
43  call atoi
44  mov [A],eax
45
46  mov eax,B
47  call atoi
48  mov [B],eax
49
50  mov eax,C
51  call atoi
52  mov [C],eax
53
54  mov eax,min
55  call atoi
56  mov [min],eax
57
58  mov ecx,[A]
59  mov [min],ecx
60
61  cmp ecx,[B]
62  jl check_B
63  mov ecx,[B]
64  mov [min],ecx
65
66  check_B:
67  mov ecx,[min]
68  cmp ecx,[C]
69  jl fin
70  mov ecx,[C]
71  mov [min],ecx
72
73  fin:
74  mov eax, msg4
75  call sprint
76  mov eax,[min]
77  call iprintLF
78  call quit

```

Рис. 3.14: продолжение кода

```

aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-p
● c/lab08$ ./lab8-3
Введите A: 8
Введите B: 88
Введите C: 68
Наименьшее число: 8
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-p
○ c/lab08$

```

Рис. 3.15: наименьшее число_вывод

Для второго задания пишу программу, вычисляющую функции $2x+a$ или $2x+1$ в зависимости от переменной a (рис. 3.16), (рис. 3.17)

```

ASM lab8-4.asm
1  %include 'in_out.asm'
2
3  section .data
4
5  msg1 db 'Введите x: ',0h
6  msg2 db 'Введите a: ',0h
7
8  msg3 db "Ответ: ",0h
9
10 section .bss
11 x resb 10
12 a resb 10
13
14 section .text
15 global _start
16 _start:
17
18 mov eax,msg1
19 call sprint
20
21 mov ecx,x
22 mov edx,10
23 call sread
24
25 mov eax,msg2
26 call sprint
27
28 mov ecx,a
29 mov edx,10
30 call sread
31
32 mov eax, a
33 call atoi
34 mov [a], eax
35

```

Рис. 3.16: код

```

36  mov eax,x
37  call atoi
38  mov [x],eax
39
40  mov ecx, [a]
41
42  cmp ecx,0
43  jne check_a
44  mov ecx,0
45
46  mov eax, [x]
47  mov ebx, 2
48  mul ebx
49  inc eax
50
51  mov edi,eax
52  jmp fin
53
54  check_a:
55
56  mov eax, [x]
57  mov ebx, 2
58  mul ebx
59
60  mov ecx, [a]
61  add eax, ecx
62
63  mov edi,eax
64
65  jmp fin
66
67  fin:
68  mov eax, msg3
69  call sprint
70  mov eax, edi
71  call iprintLF
72  call quit

```

Рис. 3.17: продолжение кода

```

• c/lab08$ ./lab8-4
Введите x: 3
Введите a: 2
Ответ: 8
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-p
• c/lab08$ ./lab8-4
Введите x: 3
Введите a: 0
Ответ: 7

```

Рис. 3.18: lab7-3.asm_вывод

Загрузка всех файлов на Git.

Далее создается отчет по 8й лабораторной работе с помощью Markdown.

4 Выводы

В ходе данной лабораторной работы были изучены команды условного и безусловного переходов, приобретены навыки написания программ с данными командами. Также была изучена структура листинга.