Лабораторная работа №8

Команды безусловного и условного переходов в Nasm. Программирование ветвлений.

Киньябаева Аиша Иделевна

Содержание

Список иллюстраций

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 использование 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 sprintLF

_label2:
mov eax, msg2
call sprintLF

_label3:
mov eax, msg3
call sprintLF

_end:
call quit
~
```

Рис. 3.1: код

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

jmp _end

_label2:
mov eax, msg2
call sprintLF

jmp _label1

_label3:
mov eax, msg3
call sprintLF

_end:
call quit
~
```

Рис. 3.3: код

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

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

```
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ touch lab8-2.asm
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ vim lab8-2.asm
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-2
Введите B: 30
Наибольшее число: 50
aikinjyabaeva@aikinjyabaeva-VirtualBox:~/work/arch-pc/lab08$ ./lab8-2
Введите B: 60
Наибольшее число: 60
aikinjyabaeva@aikinjyabaeva-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)

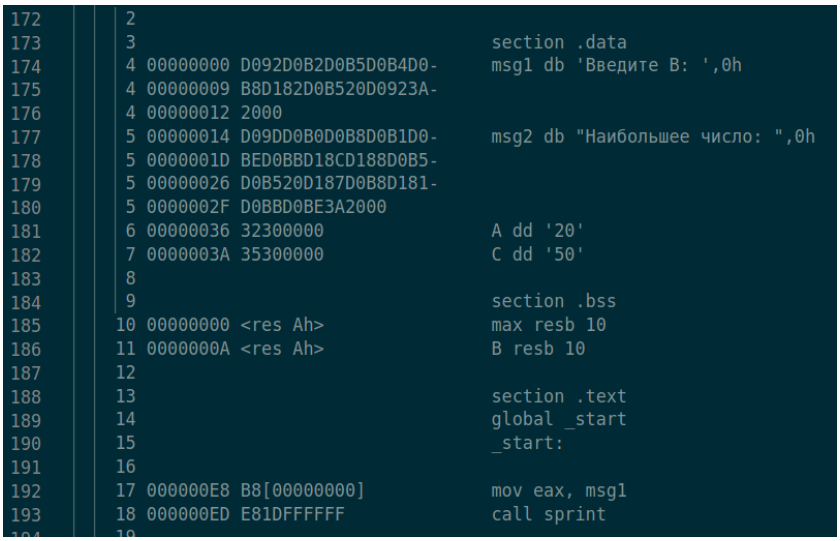

Рис. 3.8: фрагмент листинга

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)

```
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 cmp ecx,[B] - исходный

текст программы (в данном случае значащее сравнение значения ecx с B) 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)

```
%include 'in_out.asm'

section .data

msg1 db 'Введите A: ',0h
msg2 db 'Введите B: ',0h
msg3 db 'Введите C: ',0h

msg4 db "Наименьшее число: ",0h

section .bss
min resb 10
A resb 10
B resb 10
C resb 10

section .text
global _start
_start:

mov eax,msg1
call sprint

mov ecx,A
mov edx,10
call sread

mov eax,msg2
call sprint

mov ecx,B
mov edx,10
call sread

mov eax,msg3
call sprint

mov ecx,C
mov edx,10
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)

lab8-4.asm

```
%include 'in_out.asm'

section .data

msg1 db 'Введите x: ',0h
msg2 db 'Введите a: ',0h

msg3 db "Ответ: ",0h

section .bss
x resb 10
a resb 10

section .text
global _start
_start:

mov eax,msg1
call sprint

mov ecx,x
mov edx,10
call sread

mov eax,msg2
call sprint

mov ecx,a
mov edx,10
call sread

mov eax, a
call atoi
mov [a], eax

```

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.