

Лабораторная работа №8

Архитектура компьютера

Косолапов Матвей Эдуардович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	15
	Список литературы	16

Список иллюстраций

2.1	Создание каталога lab08 и файла lab8-1.asm	6
2.2	Текст программы из листинга 8.1	6
2.3	Создание исполняемого файла, проверка его работы	7
2.4	Изменяем текст программы в соответствии с листингом 8.2	7
2.5	Создание исполняемого файла, проверка его работы	7
2.6	Текст измененной программы	8
2.7	Создание исполняемого файла. проверка его работы	8
2.8	Текст программы из листинга 8.3	9
2.9	Создание исполняемого файла, проверка его работы	10
2.10	Создание файла листинга	10
2.11	Открытие файла листинга	10
2.12	Строка до удаления операнда	10
2.13	Строка после удаления операнда	11
2.14	Ошибка при создании листинг-файла	11
2.15	Ошибка в самом файле на месте удалённого операнда	11
2.16	Программа №1 для нахождения наименьшего числа	12
2.17	Создание исполняемого файла, проверка его работы	12
2.18	Программа №2	13
2.19	Создание исполняемого файла, проверка его работы	14

Список таблиц

1 Цель работы

Изучить команды условного и безусловного перехода, приобрести навыки написания программ с использованием перехода, познакомиться с назначением и структурой файла листинга.

2 Выполнение лабораторной работы

1. Создаем каталог для программ лабораторной работы №8, переходим в него и создаем файл lab8-1.asm (рис. 2.1):

```
[mekosolapov@fedora report]$ mkdir ~/work/arch-pc/lab08
[mekosolapov@fedora report]$ cd ~/work/arch-pc/lab08
[mekosolapov@fedora lab08]$ touch lab8-1.asm
[mekosolapov@fedora lab08]$
```

Рис. 2.1: Создание каталога lab08 и файла lab8-1.asm

2. Вводим в файл lab8-1.asm текст предложенной программы (рис. 2.2):

```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение No 1',0
4 msg2: DB 'Сообщение No 2',0
5 msg3: DB 'Сообщение No 3',0
6
7 SECTION .text
8 GLOBAL _start
9 _start:
10
11 jmp _label2
12
13 _label1:
14 mov eax, msg1 ; Вывод на экран строки
15 call sprintf ; 'Сообщение No 1'
16 _label2:
17 mov eax, msg2 ; Вывод на экран строки
18 call sprintf ; 'Сообщение No 2'
19 _label3:
20 mov eax, msg3 ; Вывод на экран строки
21 call sprintf ; 'Сообщение No 3'
22 _end:
23 call quit ; вызов подпрограммы завершения
```

Рис. 2.2: Текст программы из листинга 8.1

3. Создаем исполняемый файл и запускаем его(рис. 2.3):

```
[mekosolapov@fedora lab08]$ nasm -f elf lab8-1.asm
[mekosolapov@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[mekosolapov@fedora lab08]$ ./lab8-1
Сообщение No 2
Сообщение No 3
[mekosolapov@fedora lab08]$
```

Рис. 2.3: Создание исполняемого файла, проверка его работы

4. Далее в текст программы после вывода сообщения №2 добавим инструкцию `jmp` с меткой `_label1` и после вывода сообщения №1 добавим инструкцию `jmp` с меткой `_end`. Изменим текст программы в соответствии с листингом 8.2 (рис. 2.4):

```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение No 1',0
4 msg2: DB 'Сообщение No 2',0
5 msg3: DB 'Сообщение No 3',0
6
7 SECTION .text
8 GLOBAL _start
9 _start:
10
11 jmp _label2
12
13 _label1:
14     mov eax, msg1 ; Вывод на экран строки
15     call sprintf ; 'Сообщение No 1'
16     jmp _end
17
18 _label2:
19     mov eax, msg2 ; Вывод на экран строки
20     call sprintf ; 'Сообщение No 2'
21     jmp _label1
22
23 _label3:
24     mov eax, msg3 ; Вывод на экран строки
25     call sprintf ; 'Сообщение No 3'
26 _end:
27     call quit ; вызов подпрограммы завершения
```

Рис. 2.4: Изменяем текст программы в соответствии с листингом 8.2

5. Создаем исполняемый файл и проверяем его работу (рис. 2.5):

```
[mekosolapov@fedora lab08]$ nasm -f elf lab8-1.asm
[mekosolapov@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[mekosolapov@fedora lab08]$ ./lab8-1
Сообщение No 2
Сообщение No 1
[mekosolapov@fedora lab08]$
```

Рис. 2.5: Создание исполняемого файла, проверка его работы

6. Изменим текст программы, добавив и изменив инструкцию `jmp`, чтобы сообщения выводились с 3-го по 1-ый (рис. 2.6):

```

1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение No 1',0
4 msg2: DB 'Сообщение No 2',0
5 msg3: DB 'Сообщение No 3',0
6
7 SECTION .text
8 GLOBAL _start
9 _start:
10
11 jmp _label3
12
13 _label1:
14 mov eax, msg1 ; Вывод на экран строки
15 call sprintLF ; 'Сообщение No 1'
16 jmp _end
17
18 _label2:
19 mov eax, msg2 ; Вывод на экран строки
20 call sprintLF ; 'Сообщение No 2'
21 jmp _label1
22
23 _label3:
24 mov eax, msg3 ; Вывод на экран строки
25 call sprintLF ; 'Сообщение No 3'
26 jmp _label2
27
28 _end:
29 call quit ; вызов подпрограммы завершения

```

Рис. 2.6: Текст измененной программы

7. Создаем исполняемый файл и проверяем его работу (рис. 2.7):

```

[mekosolapov@fedora lab08]$ nasm -f elf lab8-1.asm
[mekosolapov@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[mekosolapov@fedora lab08]$ ./lab8-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
[mekosolapov@fedora lab08]$

```

Рис. 2.7: Создание исполняемого файла. проверка его работы

8. Создаем файл lab8-2.asm, вводим в него текст программы из листинга 8.3 (рис. 2.8):


```

2 section .data
3 msg1 db 'Введите B: ',0h
4 msg2 db "Наибольшее число: ",0h
5 A dd '20'
6 C dd '50'
7 section .bss
8 max resb 10
9 B resb 10
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15 call sprint
16 ; ----- Ввод 'B'
17 mov ecx,B
18 mov edx,10
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A] ; 'ecx = A'
26 mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 jg check_B ; если 'A>C', то переход на метку 'check_B',
30 mov ecx,[C] ; иначе 'ecx = C'
31 mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi ; Вызов подпрограммы перевода символа в число
36 mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 mov ecx,[B] ; иначе 'ecx = B'
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov eax, msg2
46 call sprint ; Вывод сообщения 'Наибольшее число: '
47 mov eax,[max]
48 call iprintLF ; Вывод 'max(A,B,C)'
49 call quit ; Выход

```

Рис. 2.8: Текст программы из листинга 8.3

9. Создаем исполняемый файл и проверяем его (рис. 2.9):

```

[mekosolapov@fedora lab08]$ nasm -f elf lab8-2.asm
[mekosolapov@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[mekosolapov@fedora lab08]$ ./lab8-2
Введите B: 15
Наибольшее число: 50
[mekosolapov@fedora lab08]$ ./lab8-2
Введите B: 39
Наибольшее число: 50
[mekosolapov@fedora lab08]$ ./lab8-2
Введите B: 102
Наибольшее число: 102
[mekosolapov@fedora lab08]$ ./lab8-2
Введите B: 51
Наибольшее число: 51
[mekosolapov@fedora lab08]$

```

Рис. 2.9: Создание исполняемого файла, проверка его работы

10. Далее создаем файл листинга для программы из файла lab8-2.asm.(рис. 2.10):

```

[mekosolapov@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
[mekosolapov@fedora lab08]$ gedit lab8-2.lst
[mekosolapov@fedora lab08]$

```

Рис. 2.10: Создание файла листинга

11. Открываем файл листинга с помощью текстового редактора gedit. В строках 24, 25, 28 мы видим последовательно идущие номер строки, смещение машинного кода от начала текущего сегмента, машинный код(инструкции, используемые для разных целей), исходный текст программы(рис. 2.11):

20	20	<1> ;----- sprint -----
21	21	<1> ; Функция печати сообщения
22	22	<1> ; входные данные: mov eax,<message>
23	23	<1> sprint:
24	24 0000000F 52	<1> push edx
25	25 00000010 51	<1> push ecx
26	26 00000011 53	<1> push ebx
27	27 00000012 50	<1> push eax
28	28 00000013 E8E8FFFFFF	<1> call slen
29	29	<1>

Рис. 2.11: Открытие файла листинга

12. Открываем файл с программой lab8-2.asm и на 14 строке удаляем операнд(рис. 2.12 - рис. 2.13):

```

14 mov eax,msg1

```

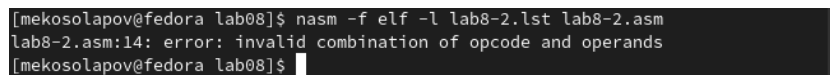
Рис. 2.12: Строка до удаления операнда



```
14 mov eax
```

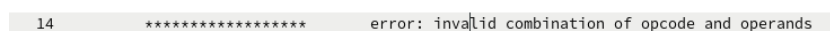
Рис. 2.13: Строка после удаления операнда

13. Создаём файл листинга, получаем сообщение об ошибке, заходим в созданный файл и видим ошибку(рис. 2.14 - рис. 2.15):



```
[mekosolapov@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:14: error: invalid combination of opcode and operands
[mekosolapov@fedora lab08]$
```

Рис. 2.14: Ошибка при создании листинг-файла

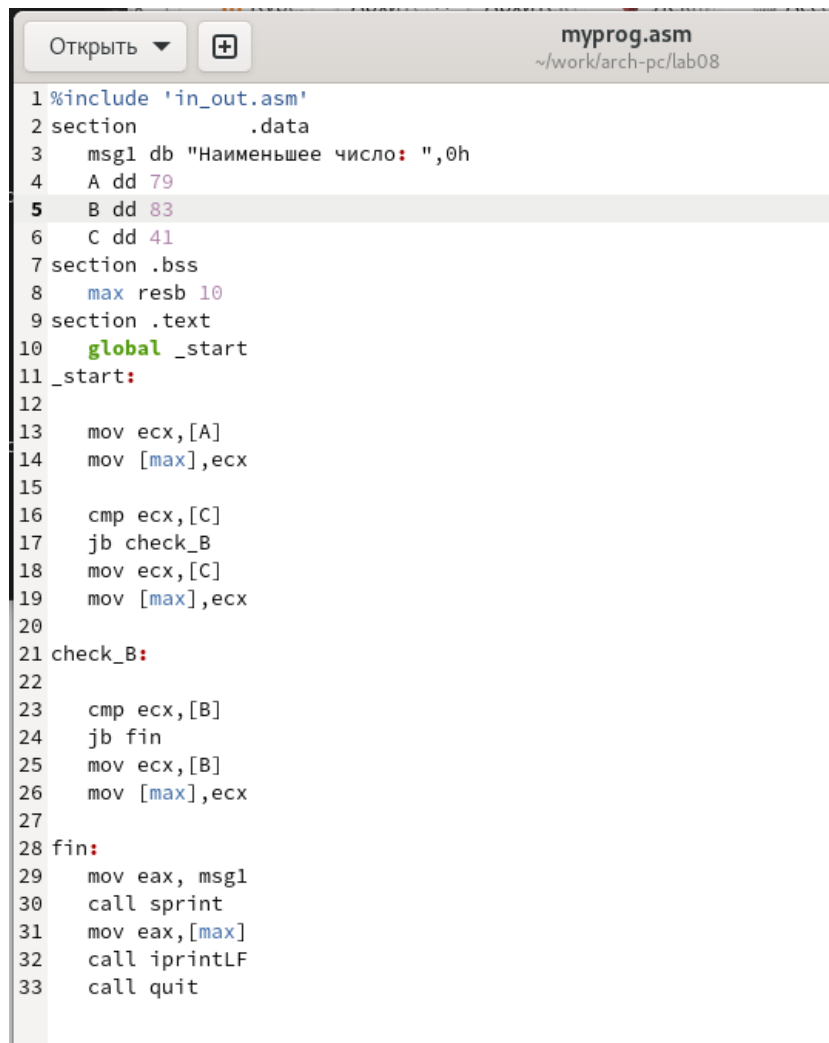


```
14 ***** error: invalid combination of opcode and operands
```

Рис. 2.15: Ошибка в самом файле на месте удалённого операнда

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

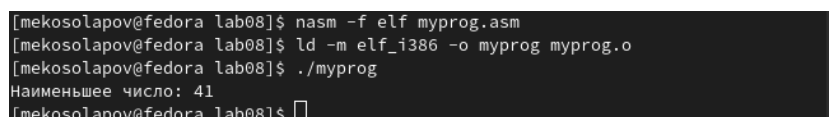
14. Пишем программу №1 для нахождения наименьшего числа среди чисел 79, 83, 41(6 вариант)(рис. 2.16):



```
1 %include 'in_out.asm'
2 section .data
3     msg1 db "Наименьшее число: ",0h
4     A dd 79
5     B dd 83
6     C dd 41
7 section .bss
8     max resb 10
9 section .text
10    global _start
11    _start:
12
13    mov ecx,[A]
14    mov [max],ecx
15
16    cmp ecx,[C]
17    jb check_B
18    mov ecx,[C]
19    mov [max],ecx
20
21 check_B:
22
23    cmp ecx,[B]
24    jb fin
25    mov ecx,[B]
26    mov [max],ecx
27
28 fin:
29    mov eax, msg1
30    call sprint
31    mov eax,[max]
32    call iprintLF
33    call quit
```

Рис. 2.16: Программа №1 для нахождения наименьшего числа

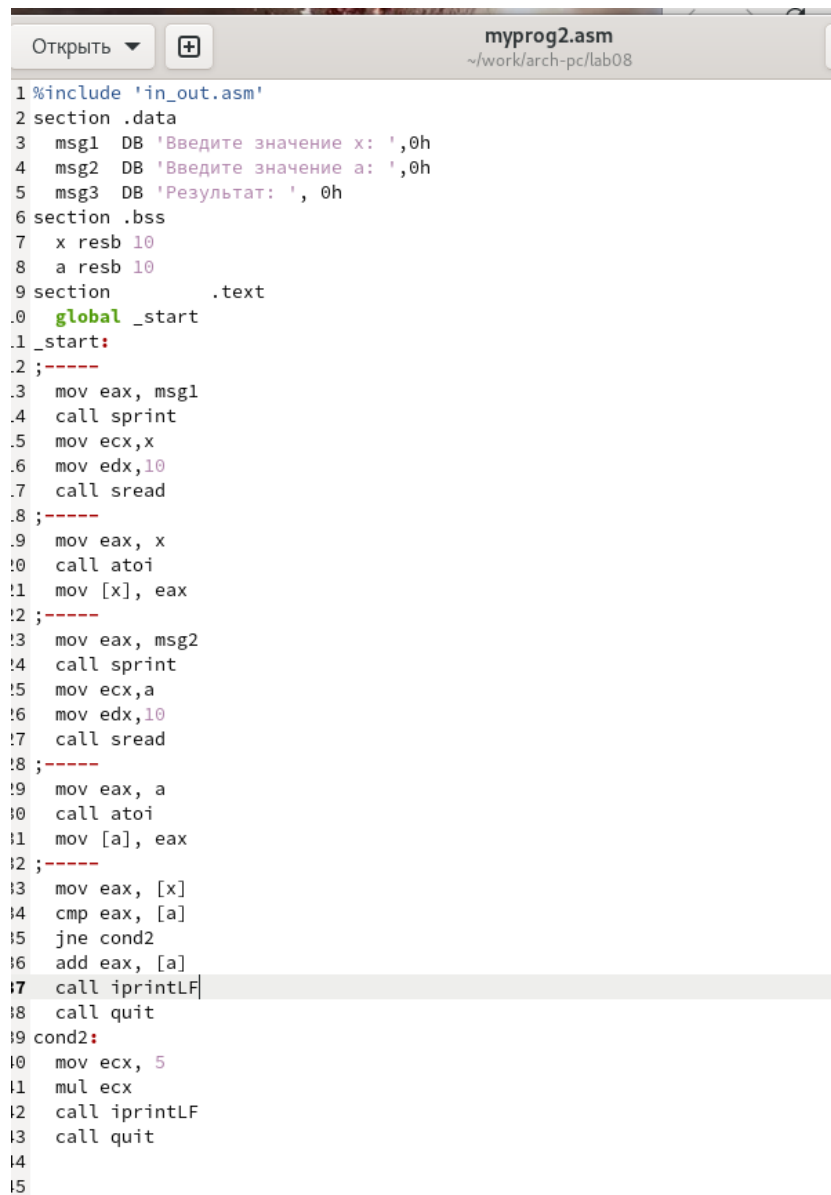
15. Создание исполняемого файла и проверка работы(рис. 2.17):



```
[mekosolapov@fedora lab08]$ nasm -f elf myprog.asm
[mekosolapov@fedora lab08]$ ld -m elf_i386 -o myprog myprog.o
[mekosolapov@fedora lab08]$ ./myprog
Наименьшее число: 41
[mekosolapov@fedora lab08]$
```

Рис. 2.17: Создание исполняемого файла, проверка его работы

16. Пишем программу №2 по 6 варианту с предложенным условием(рис. 2.18):

A screenshot of a text editor window titled 'myprog2.asm' with a subtitle '~/.work/arch-pc/lab08'. The editor contains assembly code for a program that prompts for two numbers, x and a, and calculates their sum. The code is as follows:

```
1 %include 'in_out.asm'
2 section .data
3 msg1 DB 'Введите значение x: ',0h
4 msg2 DB 'Введите значение a: ',0h
5 msg3 DB 'Результат: ', 0h
6 section .bss
7 x resb 10
8 a resb 10
9 section .text
10 global _start
11 _start:
12 ;-----
13 mov eax, msg1
14 call sprint
15 mov ecx,x
16 mov edx,10
17 call sread
18 ;-----
19 mov eax, x
20 call atoi
21 mov [x], eax
22 ;-----
23 mov eax, msg2
24 call sprint
25 mov ecx,a
26 mov edx,10
27 call sread
28 ;-----
29 mov eax, a
30 call atoi
31 mov [a], eax
32 ;-----
33 mov eax, [x]
34 cmp eax, [a]
35 jne cond2
36 add eax, [a]
37 call iprintLF
38 call quit
39 cond2:
40 mov ecx, 5
41 mul ecx
42 call iprintLF
43 call quit
44
45
```

Рис. 2.18: Программа №2

17. Создание исполняемого файла и проверка работы(рис. 2.19):

```
[mekosolapov@fedora lab08]$ nasm -f elf myprog2.asm
[mekosolapov@fedora lab08]$ ld -m elf_i386 -o myprog2 myprog2.o
[mekosolapov@fedora lab08]$ ./myprog2
Введите значение x: 2
Введите значение a: 2
4
[mekosolapov@fedora lab08]$ ./myprog2
Введите значение x: 5
Введите значение a: 15
25
[mekosolapov@fedora lab08]$ ./myprog2
Введите значение x: 2
Введите значение a: 1
10
```

Рис. 2.19: Создание исполняемого файла, проверка его работы

3 Выводы

В ходе данной лабораторной работы я изучил команды условного и безусловного перехода, приобрел навыки написания программ с использованием перехода, познакомился с назначением и структурой файла листинга.

Список литературы