Отчёт по лабораторной работе №7

Архитектура компьютера

Андреева Софья Владимировна

Содержание

Список иллюстраций

1 Цель работы

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

2 Выполнение лабораторной работы

Создадим каталог для программ лабораторной работы № 7, перейдем в него и создадим файл lab6-7.asm.Введем в файл lab7-1.asm текст программы из листинга 7.1.Создадим исполняемый файл и запустим его. Использование инструкции jmp _label2 меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки _label2, пропустив вывод первого сообщения.(рис. 2.1).

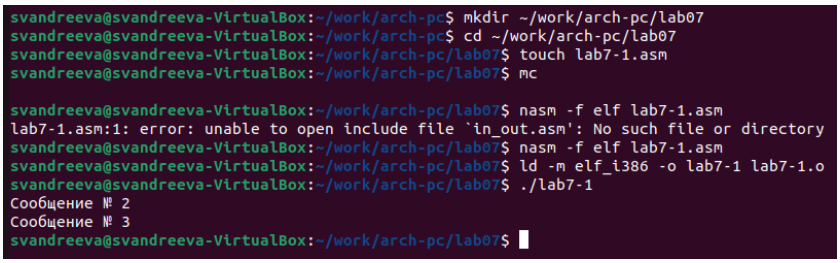

Рис. 2.1: Результат работы файла lab7-1.asm

Изменим программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию jmp с меткой _label1 и после вывода сообщения № 1 добавим инструкцию jmp с меткой _end.Создадим исполняемый файл и проверим его работу(рис. 2.2).

```
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ mc

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

Рис. 2.2: Запуск измененного файла.

Изменим текст программы (рис. 2.3), чтобы вывод программы был следующим: Сообщение № 3 Сообщение № 2 Сообщение № 1

```
%include 'in_out.asm' ; подключение внешнего файла
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 sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.3: Измененный текст файла

Создадим исполняемый файл и запустим его. Всё получилось (рис. 2.4).

```
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ mc

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

Рис. 2.4: Запуск измененного файла.

Создадим файл lab7-2.asm в каталоге ~/work/arch-pc/lab07.Внимательно изучим текст программы из листинга 7.3 и введем в lab7-2.asm.Создим исполняемый файл и проверим его работу для разных значений B, я ввела сначала 5, а затем 70.(рис. 2.5).

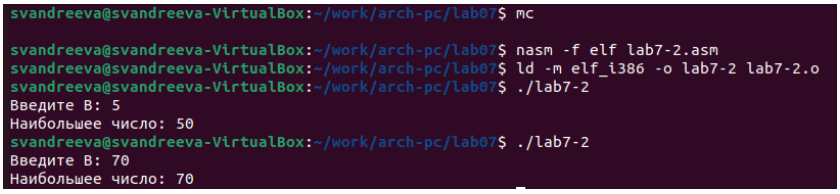

Рис. 2.5: Работа файла lab7-2.asm

Создадим файл листинга для программы из файла lab7-2.asm: nasm -f elf -l lab7-2.lst lab7-2.asm Откроем файл листинга lab7-2.lst с помощью текстового редактора mcedit.Внимательно ознакомимся с его форматом и содержимым.(рис. 2.6).

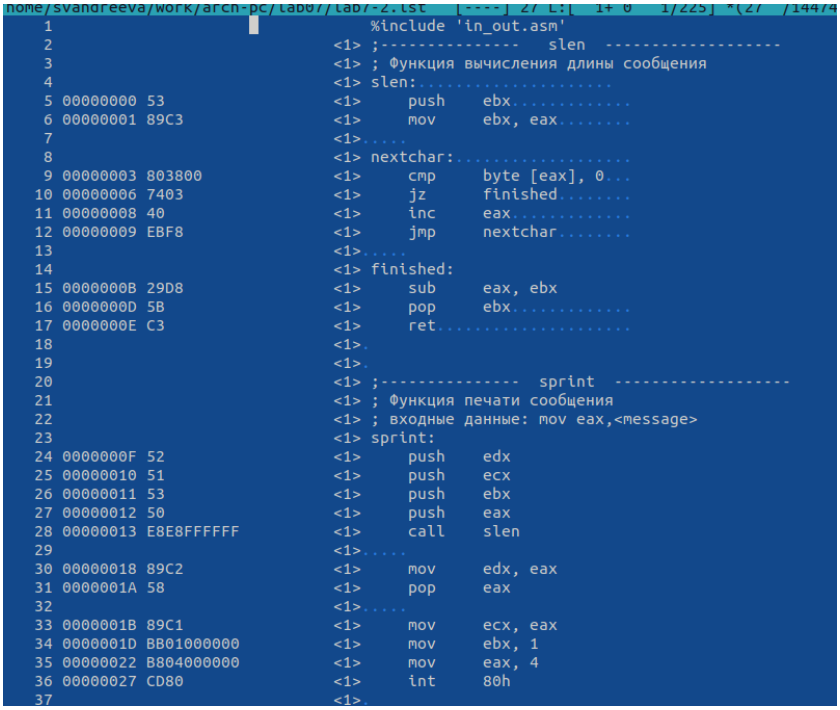

Рис. 2.6: Файл листинга lab7-2.lst.

Рассмотрим 24 строку: “00000101 B8 [0A000000] mov eax,B”. Ее адрес “00000101”, Машинный код - “B8 [0A000000]”, а mov eax,B - исходный текст программы, означающий что в регистр eax мы вносим значения переменной B. Рассмотрим 38 строку: “00000134 E863FFFFFF call atoi”. Ее адрес “00000134”, Машинный код - E863FFFFFF, а call atoi - исходный текст программы, означающий что символ лежащий в строке выше переводится в число. Рассмотрим 50 строку:“00000162 A1[00000000] mov eax,[max]”. Ее адрес “00000162”, Машинный код - A1[00000000], а mov eax,[max] - исходный текст программы, означающий что число хранившееся в переменной max записывается в регистр eax.

Откроем файл с программой lab7-2.asm и в инструкции с двумя операндами удалим один операнд (вместо cmp ecx, [C] оставим cmp ecx)(рис. 2.7).

```
; ---------- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ---------- Преобразование 'max(A,C)' из символа в число
check_B:
```

Рис. 2.7: Удаление одного операнда

Выполним трансляцию с получением файла листинга.Нам выдало ошибку, так как для программы нужно два операнда(рис. 2.8).

```
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ mc

svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:28: error: invalid combination of opcode and operands
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.8: Ошибка

В файле листинга нам показывает где именно ошибка и с чем она связана.(рис. 2.9) .

```
27                              ; ---------- Сравниваем 'A' и 'C' (как символы)
28                              cmp ecx ; Сравниваем 'A' и 'C'
28                      ****************  error: invalid combination of opcode and operands
29 0000011C 7F0C                jg check_B ; если 'A>C', то переход на метку 'check_B
```

Рис. 2.9: Файл листинга

3 Задание для самостоятельной работы.

Напишем программу в файле samr12.1.asm для нахождения наименьшей из 3 целочисленных переменных A,B и C (рис. 3.1).

```
SECTION .data
A1 DB 'Введите число A: ',0h
B1 DB 'Введите число B: ',0h
C1 DB 'Введите число C: ',0h
otv DB 'Наименьшее число: ',0h
SECTION .bss
min RESB 20
A RESB 20
B RESB 20
C RESB 20

SECTION .text
GLOBAL _start
_start:

mov eax,A1
call sprint

mov ecx,A
mov edx,20
call sread

mov eax, A
call atoi
mov [A],eax

xor eax,eax

mov eax,B1
call sprint

mov ecx,B
mov edx,20
call sread

mov eax,B
```

Рис. 3.1: Программа вычисления функции

Создадим исполняемый файл и запустим его.Мой вариант 12, поэтому проверим программу для значений 99,29 и 26.Все исполнилось корректно (рис. 3.2).

```
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ nasm -f elf samr12.1.asm
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 -o samr12.1 samr12.1.o
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$ ./samr12.1
Введите число A: 99
Введите число B: 29
Введите число C: 26
Наименьшее число: 26
svandreeva@svandreeva-VirtualBox:~/work/arch-pc/lab07$
```

Рис. 3.2: Запуск файла

Напишем программу в файле samr12.2.asm, которая для введенных с клавиатуры значений x и a вычисляетзначение заданной функции и выводит результат вычисления.Мой вариант-12.Составим программу для функции f(x)=ax при x < 5 и f(x)= x − 5 при x ⩾ 5 (рис. 3.3).

```
 GNU nano 6.2                          /home/svandreeva/work/arch-pc/la
%include 'in_out.asm'

section .data
    msgf db 'f(x) = a*x, x<5',10, 9, 'x-5, ','x>=5',0h
    msgx db 'Введите x: ',0h
    msga db 'Введите a: ',0h
    msg2 db 'f(x) = ',0h

section .bss
    res resb 10
    x resb 10
    a resb 10

section .text
    global _start

_start:
; ---------- Вывод функции
    mov eax, msgf
    call sprintLF

; ---------- Получение переменной x
    mov eax, msgx
    call sprint

    mov ecx,x
    mov edx,10
    call sread

    mov eax,x
    call atoi ; Вызов подпрограммы перевода символа в число
    mov [x],eax ; запись преобразованного числа в 'x'

; ---------- Получение переменной a
    mov eax, msga
    call sprint
```

Рис. 3.3: Программа вычисления функции.Вариант 12

Создадим исполняемый файл и запустим его.Проверим его для значений (3;7)

и (6;4).Все исполнилось корректно (рис. 3.4).

Рис. 3.4: Запуск файла

4 Выводы

Я изучила команды условного и безусловного переходов и приобрела навыки написания программ с использованием переходов, познакомилась с назначением и структурой файла листинга.