Отчёт по лабораторной работе №8

Дисциплина: Архитектура компьютера

Ищенко Ирина Олеговна

Содержание

Список иллюстраций

Список таблиц

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

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

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

Рис. 2.1: Создание каталога и файла

Введем в файл lab8-1.asm текст программы из листинга 1. Создадим исполняемый файл и запустим его (рис. 2.2). Листинг 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 _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
_label2:
```

```
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

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

Рис. 2.2: Запуск первой программы

Измените текст программы в соответствии с листингом 2. Создадим исполняемый файл и запустим его (рис. 2.3).

```
%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 _label2
_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'
_end:
call quit ; вызов подпрограммы завершения
```

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

Рис. 2.3: Запуск измененной первой программы

В соответствии с листингом 3 изменим текст программы, добавив и изменив инструкции jmp, так чтобы выводилось сначала Сообщение №3, затем Сообщение №2 и в конце Сообщение №1. Проверим работу программы (рис. 2.4). Листинг 3:

```
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
```

```
[ioithenko@fedora lab08]$ mc

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

Рис. 2.4: Запуск измененной первой программы

Создадим файл lab8-2.asm в каталоге ~/work/arch-pc/lab08. Введем текст программы из листинга 4 в lab8-2.asm. Создадим исполняемый файл и запустим программу (рис. 2.5) и (рис. 2.6). Листинг 4:

```
%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:
```

```
; ---------- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
; ---------- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ---------- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
; ---------- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
; ---------- 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:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в `max`
; ---------- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin ; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
```

```
mov [max],ecx
; ---------- Вывод результата
fin:
mov eax, msg2
call sprint ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход
```

```
[ioithenko@fedora lab08]$ touch lab8-2.asm
[ioithenko@fedora lab08]$ mc

[ioithenko@fedora lab08]$ nasm -f elf lab8-2.asm
[ioithenko@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[ioithenko@fedora lab08]$ ./lab8-2
Введите B: 10
Наибольшее число: 50
[ioithenko@fedora lab08]$
```

Рис. 2.5: Запуск второй программы

```
[ioithenko@fedora lab08]$ ./lab8-2
Введите B: 9
Наибольшее число: 50
[ioithenko@fedora lab08]$ ./lab8-2
Введите B: 57
Наибольшее число: 57
```

Рис. 2.6: Проверка второй программы для разных значений

Создадим файл листинга для программы из файла lab8-2.asm и откроем его с помощью тексового редактора (рис. 2.7).

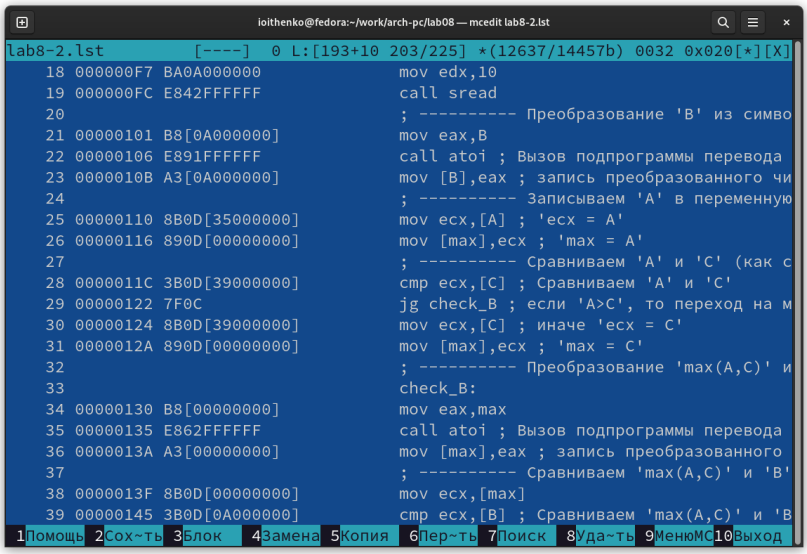

Рис. 2.7: Файл листинга

Рассмотрим содержимое трех строк из файла листинга:

строка 18: 18 - номер строки 000000F7 - адрес BA0A000000 - машинный код mov edx, 10 - исходный текст программы

строка 19: 19 - номер строки 000000FC - адрес E842FFFFFF - машинный код call sread - исходный текст программы

строка 21: 21 - номер строки 00000101 - адрес B8[0A000000] - машинный код mov eax, B - исходный текст программы

Откроем файл с программой lab8-2.asm и в любой инструкции с двумя операндами удалим один операнд. Выполним трансляцию с получением файла листинга (рис. 2.8). Откроем файл листинга (рис. 2.9). При трансляции файла появляется сообщение об ошибке, создается только файл листинга, в котором также появилось сообщение об ошибке.

```
[ioithenko@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:38: error: invalid combination of opcode and operands
[ioithenko@fedora lab08]$ mcedit lab8-2.lst

[ioithenko@fedora lab08]$
```

Рис. 2.8: Создание файла листинга

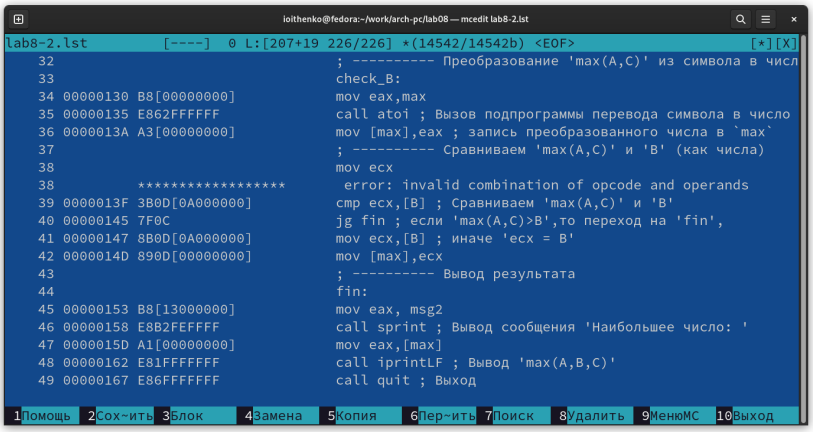

Рис. 2.9: Файл листинга с отображением ошибки

3 Выполнение заданий для самостоятельной работы

Напишем программу нахождения наименьшей из 3 целочисленных переменных a, b и c (листинг 5). Значения переменных выбрали из варианта 10, в соответствие с лабораторной работой №7. Создадим исполняемый файл и проверим его работу (рис. 3.1). Листинг 5:

```
%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
a resb 10
b resb 10
c resb 10
min resb 10

section .text
global _start
_start:
```

```
mov eax,msg1
call sprint
mov ecx,a
mov edx,10
call sread
mov eax,a
call atoi
mov [a],eax

mov eax,msg2
call sprint
mov ecx,b
mov edx,10
call sread
mov eax,b
call atoi
mov [b],eax

mov eax,msg3
call sprint
mov ecx,c
mov edx,10
call sread
mov eax,c
call atoi
mov [c],eax

mov ecx,[a]
mov [min],ecx
```

```
cmp ecx,[b]
jl check_c
mov ecx,[b]
mov [min],ecx

check_c:
cmp ecx,[c]
jl fin
mov ecx,[c]
mov [min],ecx

fin:
mov eax, msg4
call sprint
mov eax,[min]
call iprintLF
call quit
```

```
[ioithenko@fedora lab08]$ nasm -f elf lab8-task.asm
[ioithenko@fedora lab08]$ ld -m elf_i386 -o lab8-task lab8-task.o
[ioithenko@fedora lab08]$ ./lab8-task
Введите a: 41
Введите b: 62
Введите c: 35
Наименьшее число: 35
[ioithenko@fedora lab08]$
```

Рис. 3.1: Запуск программы нахождения наименьшего числа

Напишем программу, которая для введенных с клавиатуры значений x и a вычисляет значение функции f(x) из варианта 10 и выводит результат вычислений (листинг 6). Создадим исполняемый файл и проверим его работу (рис. 3.2).
Листинг 6:

```
%include 'in_out.asm'
SECTION .data
```

```
    msgA:<----->DB 'Введите A: ',0
    msgX:       DB 'Введите X: ',0
    msg: DB 'Результат: ',0

SECTION .bss
    A resb  80
    X resb  80
    result resb 80

SECTION .text
    GLOBAL _start

_start:
    mov eax,msgA
    call sprint
    mov ecx,A
    mov edx,80
    call sread
    mov eax,A
    call atoi
    mov [A],eax

    mov eax,msgX
    call sprint
    mov ecx,X
    mov edx,80
    call sread
    mov eax,X
    call atoi
```

```
    mov [X],eax

    mov ebx, [X]
    cmp ebx, 2
    jg first
    jmp second
    mov eax,msg
    call sprint

first:
    mov eax,[X]
    add eax,-2
    call iprintLF
    call quit
second:
    mov eax, [A]
    mov ebx, 3
    mul eax
    call iprintLF
    call quit
```

```
[ioithenko@fedora lab08]$ nasm -f elf lab8-task2.asm
[ioithenko@fedora lab08]$ ld -m elf_i386 -o lab8-task2 lab8-task2.o
[ioithenko@fedora lab08]$ ./lab8-task2
Введите A: 0
Введите X: 3
1
[ioithenko@fedora lab08]$ ./lab8-task2
Введите A: 2
Введите X: 1
4
[ioithenko@fedora lab08]$
```

Рис. 3.2: Запуск программы вычисления значения программы

4 Выводы

В ходе лабораторной работы я изучила команды условного и безусловного переходов, приобрела навык написания программ с использованием переходов и познакомилась с назначением и структурой файла листинга.