

Отчет по лабораторной работе №3

Дисциплина: Архитектура компьютера

Ли Евгения Олеговна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	14
	Список литературы	15

Список иллюстраций

4.1	Создание файла	8
4.2	Результат	8
4.3	Результат	8
4.4	Вывод	9
4.5	Редактирование	9
4.6	Создание файла	9
4.7	Проверка	10
4.8	Создание файла	10
4.9	Открытый файл	10
4.10	Строки	11
4.11	Удалила данный операнд	11
4.12	Трансляция	11
4.13	Изменения в листинге	11
4.14	Программа	12
4.15	Программа	13
4.16	Результат	13

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 Задание

Изучить команды условного и безусловного переходов. Приобрести навыки написания программ с использованием переходов. Ознакомиться с назначением и структурой файла листинга.

3 Теоретическое введение

условный переход – выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия.

безусловный переход – выполнение передачи управления в определенную точку программы без каких-либо условий.

4 Выполнение лабораторной работы

8.3.1. Реализация переходов в NASM

1. Я создала каталог для программ лабораторной работы No 8, перешла в него и создала файл lab8-1.asm:(рис. 4.1)

```
eoli@dk3n37 ~ $ mkdir ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/lab08
eoli@dk3n37 ~ $ cd work/study/2022-2023/"Архитектура компьютера"/arch-pc/lab08
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ touch lab8-1.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $
```

Рис. 4.1: Создание файла

2. Ввела в файл lab8-1.asm текст программы из листинга 8.1. Создала исполняемый файл и запустила его (рис. 4.2)

```
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf lab8-1.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 3
```

Рис. 4.2: Результат

Изменила текст программы в соответствии с листингом 8.2. Создала исполняемый файл и запустила его.(рис. 4.3)

```
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf lab8-1.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 1
```

Рис. 4.3: Результат

Изменила текст программы, чтобы вывод программы был следующим:(рис. 4.4)

```
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf lab8-1.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
```

Рис. 4.4: Вывод

Изменения: (рис. 4.5)

```
lab8-1.asm [-----] 11 L: [ 5+10 15/ 23] *(384 [*
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.5: Редактирование

3. Создала файл lab8-2.asm. и ввела в него текст программы из листинга 8.3.(рис. 4.6)

```
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ touch lab8-2.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $
```

Рис. 4.6: Создание файла

Создала исполняемый файл и проверила его работу для разных значений В.(рис. 4.7)

```
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf lab8-2.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-2
Введите В: 6
Наибольшее число: 50
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-2
Введите В: 3
Наибольшее число: 50
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-2
Введите В: 8
Наибольшее число: 50
```

Рис. 4.7: Проверка

8.3.2. Изучение структуры файлы листинга

4. Создала файл листинга для программы из файла lab8-2.asm (рис. 4.8)

```
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $
```

Рис. 4.8: Создание файла

Открыла файл листинга lab8-2.lst с помощью текстового редактора mcedit: (рис. 4.9)

```
lab8-2.lst  [----]  0 L: 1+ 0 1/225] *(0 /14457b) 0032 0x020
1          %include 'in_out.asm'
2          <I> ;----- slen -----
3          <I> ; Функция вычисления длины сообщения
4          <I> slen:
5          00000000 53          <I> push    ebx
6          00000001 89C3       <I> mov     ebx, eax
7          <I>
8          <I> nextchar:
9          00000003 803800     <I> cmp     byte [eax], 0
10         00000006 7403       <I> jz      finished
11         00000008 40         <I> inc     eax
12         00000009 EBF8       <I> jmp     nextchar
13         <I>
14         <I> finished:
15         0000000B 29D8       <I> sub     eax, ebx
16         0000000D 5B         <I> pop     ebx
17         0000000E C3         <I> ret
18         <I>
19         <I>
20         <I> ;----- sprint -----
21         <I> ; Функция печати сообщения
22         <I> ; входные данные: mov eax,<message>
23         <I> sprint:
24         0000000F 52         <I> push    edx
25         00000010 51         <I> push    ecx
26         00000011 53         <I> push    ebx
27         00000012 50         <I> push    eax
28         00000013 E8E8FFFF   <I> call    slen
29         <I>
30         00000018 89C2       <I> mov     edx, eax
31         0000001A 58         <I> pop     eax
32         <I>
33         0000001B 89C1       <I> mov     ecx, eax
```

Рис. 4.9: Открытый файл

Подробно объяснить содержимое трёх строк файла листинга по выбору(рис. 4.10)

```
28 0000011C 3B0D[39000000]      cmp ecx,[C] ;
29 00000122 7F06                      jg check_B ; e
30 00000124 8B0D[39000000]      mov ecx,[C] ;
31
```

Рис. 4.10: Строки

28. сравниваем АиС, 29. переход на метку Б если А>С, 30. иначе есх=С

В инструкции с двумя операндами файла с программой lab8-2.asm удалила один операнд. (рис. 4.11)

```
; преобразование в число символов в члене
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
```

Рис. 4.11: Удалила данный операнд

Выполнила трансляцию с получением файла листинга:(рис. 4.12)

```
eol1@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:25: error: invalid combination of opcode and operands
eol1@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
ld: невозможно найти lab8-2.o: Нет такого файла или каталога
eol1@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./lab8-2
bash: ./lab8-2: Нет такого файла или каталога
eol1@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $
```

Рис. 4.12: Трансляция

В этом случае не получается выходных файлов

Листинг выдает ошибку, которая отражается в терминале (рис. 4.13)

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

Рис. 4.13: Изменения в листинге

8.4. Задание для самостоятельной работы

1. Написала программу нахождения наименьшей из 3 целочисленных переменных a,b,c. Значения переменных выбрала из табл. 8.5 в соответствии с 13 вариантом (рис. 4.14, 4.15)

```
/afs/.dk.sci.pfu.edu.ru/home/e/o/eoli/work/study/2022-2023/Архитектура компьют
#include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наименьшее число: ",0h
A dd '84'
C dd '77'
section .bss
min 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 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jl check_B ; если 'A<C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'min = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,min
call atoi ; Вызов подпрограммы перевода символа в число
mov [min],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'min(A,C)' и 'B' (как числа)
mov ecx,[min]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jl fin ; если 'min(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
```

Рис. 4.14: Программа

```

call atoi ; Вызов подпрограммы перевода символа в число
mov [min],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'min(A,C)' и 'B' (как числа)
mov ecx,[min]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jl fin ; если 'min(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [min],ecx
; ----- Вывод результата
fin:
mov eax, msg2
call sprint ; Вывод сообщения 'Наименьшее число: '
mov eax,[min]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход

```

Рис. 4.15: Программа

Создала исполняемый файл и проверила его работу.(рис. 4.16)

```

eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ nasm -f elf hw1.asm
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ld -m elf_i386 -o hw1
hw1.o
eoli@dk3n37 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/lab08 $ ./hw1
Введите B: 32
Наименьшее число: 32

```

Рис. 4.16: Результат

2. Написала программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции и выводит результат вычислений.

Создала исполняемый файл и проверила его работу для значений (3;9);(6;4)

5 Выводы

Изучила команды условного и безусловного переходов. Приобрела навыки написания программ с использованием переходов. Ознакомилась с назначением и структурой файла листинга.

Список литературы