

Архитектура компьютера Отчёт

по лабораторной работе №7

Бембо жозе лумингу

Содержание

1 Цель работы	5
2 Задание	6
3 Теоретическое введение	7
4 Выполнение лабораторной работы	8
5 Выполнение лабораторной работы	16
6 Выводы	22
Список литературы	23

Список иллюстраций

4.1	Создание каталога и файла	8
4.2	Содержимое файла	8
4.3	Работа файла	9
4.4	текст программы	9
4.5	Работа файла	9
4.6	Текст программы.....	11
•	Работа файла.....	11
•	Создание файла.....	11
•	Работа файла.....	12
•	Создание файла листинга.....	12
•	Открытый файл листинга.....	13
•	Копирование файла.....	14
•	Измененный текст программы.....	14
•	Созданные файлы	14
•	Файл листинга	15
1.	Создание файла.....	16
2.	Работа файла.....	16
3.	Создание файла.....	18
4.	Текст файла	18
5.	Работа файла.....	21

Список таблиц

1 Цель работы

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

2 Задание

- 5.1 Создайте каталог для программ лабораторной работы № 7, перейдите в него и создайте файл lab7-1.asm
- 5.2 Инструкция `jmp` в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции `jmp`. Введите в файл lab7-1.asm текст программы из листинга 7.1.
- 5.3 Использование инструкции `jmp` приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход должен происходить если выполнено какое-либо условие. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: A, B и C. Значения для A и C задаются в программе, значение B вводится с клавиатуры.
- 5.4 Обычно `nasm` создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ `-l` и задав имя файла листинга в командной строке. Создайте файл листинга для программы из файла lab7-2.asm

3 Теоретическое введение

Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов: • 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

- 1) Создаю каталог для программ лабораторной работы № 7, перехожу в него и создаю файл lab7-1.asm

```
zlbembo@fedora:~:[0]$ mkdir -p ~/work/arch-pc/lab07
zlbembo@fedora:~:[0]$ cd ~/work/arch-pc/lab07
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ touch lab07-1.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$
```

Рис. 4.1: Создание каталога и файла

- 2) Ввожу в файл lab7-1.asm текст программы из листинга 7.1.

```
GNU nano 7.2                                lab07-1.asm                                Modifié
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.2: Содержимое файла

3) Создаю исполняемый файл и запускаю его

```
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf lab07-1.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ld -m elf_i386 -o lab07-1 lab07-1.o
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-1
Сообщение No 2
Сообщение No 3
zlbembo@fedora:~/work/arch-pc/lab07:[0]$
```

Рис. 4.3: Работа файла

4 Изменяю текст программы в соответствии с листингом 7.2

```
GNU nano 7.2 lab07-1.asm
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_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'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.4: текст программы

5 Создаю исполняемый файл и запускаю его

```
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf lab07-1.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ld -m elf_i386 -o lab07-1 lab07-1.o
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-1
Сообщение No 2
Сообщение No 1
zlbembo@fedora:~/work/arch-pc/lab07:[0]$
```

Рис. 4.5: Работа файла

6 Изменяю текст программы изменив инструкции jmp, чтобы вывод программы был следующим: Сообщение № 3 Сообщение № 2 Сообщение № 1

```
%include 'in_out.asm' ; подключение внешнего файлаSECTION
```

```
.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 sprintf ; 'Сообщение № 1'
```

```
jmp _end
```

```
_label2:
```

```
mov eax, msg2 ; Вывод на экран строки
```

```
call sprintf ; 'Сообщение № 2'
```

```
jmp _label1
```

```
_label3:
```

```
mov eax, msg3 ; Вывод на экран строки
```

```
call sprintf ; 'Сообщение № 3'
```

```
jmp _label2
```

```
_end:
```

```
call quit ; вызов подпрограммы завершения
```

```

GNU nano 7.2                                lab07-1.asm                                Modifié
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
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. 4.6: Текст программы

7 Создаю исполняемый файл и запускаю его

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf lab07-1.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ld -m elf_i386 -o lab07-1 lab07-1.o
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
zlbembo@fedora:~/work/arch-pc/lab07:[0]$

```

Рис. 4.7: Работа файла

8 Создаю файл lab7-2.asm и проверяю его создание

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ touch lab07-2.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ls
in_out.asm lab07-1 lab07-1.asm lab07-1.o lab07-2.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$

```

Рис. 4.8: Создание файла

9 Ввожу в файл текст листинга 7.3, создаю файл и запускаю его

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf lab07-2.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ld -m elf_i386 -o lab07-2 lab07-2.o
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-2
Введите В: 35
Наибольшее число: 50
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-2
Введите В: 70
Наибольшее число: 70
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-2
Введите В: 30
Наибольшее число: 50
zlbembo@fedora:~/work/arch-pc/lab07:[0]$

```

Рис. 4.9: Работа файла

- 10 Создаю файл листинга для программы из файла lab7-2.asm и открываю его в текстовом редакторе

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf -l lab07-2.list lab07-2.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ mcedit lab07-2.list

```

Рис. 4.10: Создание файла листинга

- 11 Открытый файл листинга

```

lab07-2.list  [----]  0 L:[ 1+ 0  1/226] *(0  /13363b) 0032 0x020  [*] [X]
1      %include 'in_out.asm'
2      <1> ;----- slen -----
3      <1> ; Функция вычисления длины сообщения
4      <1> slen:.....
5      00000000 53      <1>      push     ebx.....
6      00000001 89C3    <1>      mov      ebx, eax.....
7      <1>.....
8      <1> nextchar:.....
9      00000003 803800  <1>      cmp      byte [eax], 0...
10     00000006 7403    <1>      jz       finished.....
11     00000008 40      <1>      inc      eax.....
12     00000009 EBF8    <1>      jmp      nextchar.....
13     <1>.....
14     <1> finished:
15     0000000B 29D8    <1>      sub      eax, ebx
16     0000000D 5B      <1>      pop      ebx.....
17     0000000E C3      <1>      ret.....
18     <1>.....
19     <1> ;----- sprint -----
20     <1> ; Функция печати сообщения
21     <1> ; входные данные: mov eax,<message>
22     <1> sprint:
23     0000000F 52      <1>      push     edx
24     00000010 51      <1>      push     ecx
25     00000011 53      <1>      push     ebx
26     00000012 50      <1>      push     eax
27     00000013 E8E8FFFF <1>      call     slen
28     <1>.....
29     00000018 89C2    <1>      mov      edx, eax
30     0000001A 58      <1>      pop      eax
31     <1>.....
32     0000001B 89C1    <1>      mov      ecx, eax
33     0000001D BB01000000 <1>      mov      ebx, 1
34     00000022 B804000000 <1>      mov      eax, 4
35     00000027 CD80    <1>      int      80h
36     <1>.....
37     00000029 5B      <1>      pop      ebx
38     0000002A 59      <1>      pop      ecx
39     0000002B 5A      <1>      pop      edx
40     0000002C C3      <1>      ret
41     <1>.....
42     <1>.....
43     <1> ;----- sprintf -----

```

Рис. 4.11: Открытый файл листинга

17 000000F2 B9[0A000000] mov ecx,B

(17 - номер строки,

000000F2 - адрес,

B9 - машинный код,

[0A000000] - исходный текст программы

)

```

18 000000F7 BA0A000000 mov edx,10
    ( 18 - номер строки,
      000000F7 - адрес,
      BA - машинный код,
      0A000000 - исходный текст программы
    )

19 000000FC E842FFFFFF call sread
    ( 19 - номер строки,
      000000FC - адрес,
      E8 - машинный код,
      42FFFFFF - исходный текст программы
    )

```

12 Копирую файл lab7-2.asm как lab7-2-2.asm и открываю его

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ cp lab07-2.asm lab07-2-2.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ mcedit lab07-2-2.asm

```

Рис. 4.12: Копирование файла

13 Удаляю один из операндов

```

cmp ecx,[C]
jg check_B
mov ecx,[C]
mov [max],ecx

```

Рис. 4.13: Измененный текст программы

14 Создаю файл листинга

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf -l lab07-2-2.list lab07-2-2.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ls
in_out.asm  lab07-1.asm  lab07-2      lab07-2-2.list  lab07-2.asm  lab07-2.o
lab07-1     lab07-1.o    lab07-2-2.asm  lab07-2-2.o    lab07-2.list
zlbembo@fedora:~/work/arch-pc/lab07:[0]$

```

Рис. 4.14: Созданные файлы

15 Открытый файл листинга

```

lab07-2-2.list  [-----] 34 L: [ 1+ 0 1/226] *(34 /13363b) 0032 0x020  [*] [X]
1      %include 'in_out.asm'
1      <1> ;----- slen -----
2      <1> ; Функция вычисления длины сообщения
3      <1> slen:.....
4 00000000 53      <1> push    ebx.....
5 00000001 89C3    <1> mov     ebx, eax.....
6      <1>.....
7      <1> nextchar:.....
8 00000003 803800  <1> cmp     byte [eax], 0...
9 00000006 7403    <1> jz      finished.....
10 00000008 40      <1> inc     eax.....
11 00000009 EBF8    <1> jmp     nextchar.....
12      <1>.....
13      <1> finished:
14 0000000B 29D8    <1> sub     eax, ebx
15 0000000D 5B      <1> pop     ebx.....
16 0000000E C3      <1> ret.....
17      <1>.....
18      <1>.....
19      <1> ;----- sprint -----
20      <1> ; Функция печати сообщения
21      <1> ; входные данные: mov eax,<message>
22      <1> sprint:
23 0000000F 52      <1> push    edx
24 00000010 51      <1> push    ecx
25 00000011 53      <1> push    ebx
26 00000012 50      <1> push    eax
27 00000013 E8E8FFFFFF <1> call    slen
28      <1>.....
29 00000018 89C2    <1> mov     edx, eax
30 0000001A 58      <1> pop     eax
31      <1>.....
32 0000001B 89C1    <1> mov     ecx, eax
33 0000001D BB01000000 <1> mov     ebx, 1
34 00000022 B804000000 <1> mov     eax, 4
35 00000027 CD80    <1> int     80h
36      <1>.....
37 00000029 5B      <1> pop     ebx
38 0000002A 59      <1> pop     ecx
39 0000002B 5A      <1> pop     edx
40 0000002C C3      <1> ret
41      <1>.....
42      <1>.....
43      <1> ;----- sprintfLF -----

```

Рис. 4.15: Файл листинга

5 Выполнение лабораторной работы

1 Создаю файл для написания программы

```
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ touch lab07-4.asm  
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ mcedit lab07-4.asm
```

Рис. 5.1: Создание файла

2 Создание и работа файла. У меня вариант 1

```
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ nasm -f elf lab07-4.asm  
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ld -m elf_i386 -o lab07-4 lab07-4.o  
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ./lab07-4  
Наименьшее число: 17  
zlbembo@fedora:~/work/arch-pc/lab07:[0]$
```

Рис. 5.2: Работа файла

Текст файла:

```
%include 'in_out.asm'  
  
section .data  
  
msg2 db "Наименьшее число: ",0h  
  
A dd 17  
  
C dd 23  
  
B dd 45
```



```

section .bss
min resb 10
section .text
global _start
_start:
; ----- 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'
jnl 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:
; ----- Сравниваем 'min(A,C)' и 'B' (как числа)
mov ecx,[min]
cmp ecx,[B] ; Сравниваем 'min(A,C)' и 'B'
jnl 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 'Наибольшее число: '
mov eax,[min]
call iprintLF ; Вывод 'min(A,B,C)'
call quit ; Выход

```

3) Создаю файл для второго задания

```

zlbembo@fedora:~/work/arch-pc/lab07:[0]$ touch lab07-3.asm
zlbembo@fedora:~/work/arch-pc/lab07:[0]$ ls
in_out.asm  lab07-1.o      lab07-2-2.list  lab07-2.list  lab07-4
lab07-1     lab07-2       lab07-2-2.o    lab07-2.o    lab07-4.asm
lab07-1.asm lab07-2-2.asm lab07-2.asm     lab07-3.asm  lab07-4.o
zlbembo@fedora:~/work/arch-pc/lab07:[0]$

```

Рис. 5.3: Создание файла

4) Текст файла

```

zlbembo@fedora:~/work/arch-pc/lab07 — nano lab07-3.asm
GNU nano 7.2                                lab07-3.asm                                Modi
mov ecx,x
mov edx,10
call sread
; ----- Преобразование 'x' из символа в число
mov eax,x
call atoi ; Вызов подпрограммы перевода символа в число
mov [x],eax ; запись преобразованного числа в 'x'
; ----- Вывод сообщения 'Введите a: '
mov eax,msg2
call sprint
; ----- Ввод 'a'
mov ecx,a
mov edx,10
call sread
; ----- Преобразование 'a' из символа в число
mov eax,a
call atoi ; Вызов подпрограммы перевода символа в число
mov [a],eax ; запись преобразованного числа в 'a'
mov ecx, [a] ; ecx = a

cmp ecx,[x] ; Сравниваем 'a' и 'x'
jg ysl1 ; если 'a>x', то переход на метку 'ysl1',

ysl2:
mov ecx,8
mov [rez],ecx
jmp fin

ysl1:
mov eax,[a]; eax = a
mov ebx,2; ebx = 2
mul ebx; eax = 2*a
sub eax,[x]; eax = 2*a - x
mov [rez],eax; rez = eax
; ----- Вывод результата
fin:
mov eax, msg3
call sprint ; Вывод сообщения 'Результат: '
mov eax,[rez]
call iprintLF ; Вывод
call quit ; Выход

```

Рис. 5.4: Текст файла

Текст файла:

```
%include 'in_out.asm'

section .data

msg1 db 'Введите x: ',0h
msg2 db 'Введите a: ',0h msg3
db "Результат: ",0h section
.bss

a resb 10
x resb 10
rez resb 10

section .text
global _start
_start:

; ----- Вывод сообщения 'Введите x: '
mov eax,msg1
call sprint

; ----- Ввод 'x'
mov ecx,x
mov edx,10
call sread

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

; ----- Вывод сообщения 'Введите a: '
mov eax,msg2
call sprint

; ----- Ввод 'a'
mov ecx,a
mov edx,10
```

```

call sread

; ----- Преобразование 'a' из символа в число
mov eax,a

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

mov ecx, [a] ; ecx = a

cmp ecx,[x] ; Сравниваем 'a' и 'x'
jg ysl1 ; если 'a>x', то переход на метку 'ysl1',

ysl2:
mov ecx,8
mov [rez],ecx
jmp fin

ysl1:
mov eax,[a]; eax = a
mov ebx,2; ebx = 2
mul ebx; eax = 2*a
sub eax,[x]; eax = 2*a - x
mov [rez],eax; rez = eax

; ----- Вывод результата
fin:
mov eax, msg3
call sprintf ; Вывод сообщения 'Результат: '
mov eax,[rez]
call iprintLF ; Вывод
call quit ; Выход

```

5) Работа файла

```
zlbembo@fedora:~/work/arch-pc/lab07: [0]$ nasm -f elf lab07-3.asm
zlbembo@fedora:~/work/arch-pc/lab07: [0]$ ld -m elf_i386 -o lab07-3 lab07-3.o
zlbembo@fedora:~/work/arch-pc/lab07: [0]$ ./lab07-3
Введите x: 1
Введите a: 2
Результат: 3
zlbembo@fedora:~/work/arch-pc/lab07: [0]$ ./lab07-3
Введите x: 2
Введите a: 1
Результат: 8
zlbembo@fedora:~/work/arch-pc/lab07: [0]$
```

Рис. 5.5: Работа файла

6 Выводы

Мною изучены команды условного и безусловного переходов, приобретены навыки написания программ с использованием переходов, я ознакомилась с назначением и структурой файла листинга.

Список литературы