

Отчёт по лабораторной работе №7

дисциплина: Архитектура компьютеров

Ясиновская Дарья Олеговна

Содержание

Цель работы	4
Выполнение лабораторной работы	5
Самостоятельная работа	12
Выводы	15

Список иллюстраций

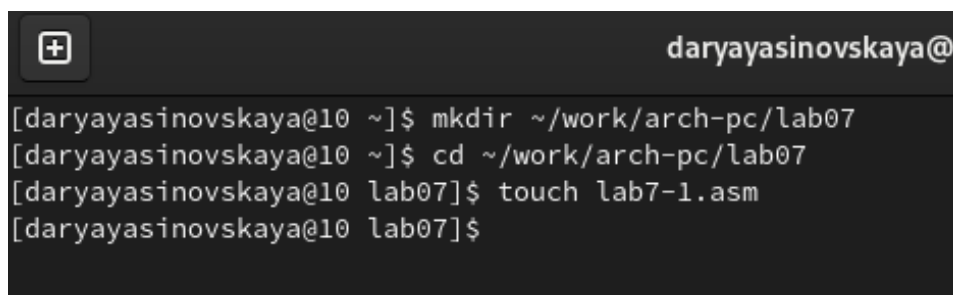
1	Создание каталога и файла	5
2	Файл с текстом	6
3	Запуск файла	6
4	Изменение файла	7
5	Запуск файла	7
6	Изменение текста	8
7	Запуск файла	8
8	Ввод текста	9
9	Вывод программы для разных значение В	10
10	Файл листинга	10
11	Открытие файла	11
12	Файл с ошибкой	12
13	Программа	13
14	Запуск файла	13
15	Программа	14
16	Запуск файла	14

Цель работы

Целью лабораторной работы является освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

Выполнение лабораторной работы

Создаю каталог для лабораторной работы №7, перешла в него и создала файл.

A terminal window with a dark background. The title bar shows a plus icon on the left and the username 'daryayasinovskaya@' on the right. The terminal contains four lines of text: the first line shows the user running 'mkdir ~/work/arch-pc/lab07'; the second line shows the user running 'cd ~/work/arch-pc/lab07'; the third line shows the user running 'touch lab7-1.asm'; and the fourth line shows the prompt after the command has finished.

```
[daryayasinovskaya@10 ~]$ mkdir ~/work/arch-pc/lab07
[daryayasinovskaya@10 ~]$ cd ~/work/arch-pc/lab07
[daryayasinovskaya@10 lab07]$ touch lab7-1.asm
[daryayasinovskaya@10 lab07]$
```

Рис. 1: Создание каталога и файла

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

```

GNU nano 7.2 /home/daryayasinovskaya/wo
%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 sprintLF ; 'Сообщение No 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение No 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение No 3'
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 2: Файл с текстом

Создаю исполняемый файл и запускаю его.

```

[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-1.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[daryayasinovskaya@10 lab07]$ ./lab7-1
Сообщение No 2
Сообщение No 3
[daryayasinovskaya@10 lab07]$ █

```

Рис. 3: Запуск файла

Добавляю инструкции в соответствии с листингом 7.2

```
GNU nano 7.2 /home/daryayasinovskaya/work/arc
#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 sprintLF ; 'Сообщение No 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение No 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение No 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4: Изменение файла

Создаю исполняемый файл и запускаю его.

```
[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-1.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[daryayasinovskaya@10 lab07]$ ./lab7-1
Сообщение No 2
Сообщение No 1
[daryayasinovskaya@10 lab07]$
```

Рис. 5: Запуск файла

Изменяю текст программы, чтобы он выводил сначала сообщение №3, затем сообщение №2, а затем сообщение №1

```

GNU nano 7.2 /home/daryayasinovskaya/work/arc
#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 sprintLF ; 'Сообщение No 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение No 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение No 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 6: Изменение текста

Создаю исполняемый файл и запускаю его.

```

[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-1.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[daryayasinovskaya@10 lab07]$ ./lab7-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
[daryayasinovskaya@10 lab07]$ █

```

Рис. 7: Запуск файла

Создаю файл lab7-2.asm и ввожу в него текст из листинга 7.3


```
GNU nano 7.2 /home/daryayasnovskaya/work/arch-p
#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:
^G Справка      ^O Записать     ^W Поиск        ^K Вырезать     ^T Вып
^X Выход        ^R ЧитФайл     ^\ Замена       ^U Вставить     ^J Выр
```

Рис. 8: Ввод текста

Создаю исполняемый файл и проверяю его работу для разных значений.

```

[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-2.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[daryayasinovskaya@10 lab07]$ ./lab7-2
Введите В: 10
Наибольшее число: 50
[daryayasinovskaya@10 lab07]$ ./lab7-2
Введите В: 70
Наибольшее число: 70
[daryayasinovskaya@10 lab07]$ █

```

Рис. 9: Вывод программы для разных значение В

Создаю файл листинга для программы из файла lab7-2.asm

```

[daryayasinovskaya@10 lab07]$ nasm -f elf -l lab7-2.lst lab7-2.asm
[daryayasinovskaya@10 lab07]$

```

Рис. 10: Файл листинга

Открываю файл листинга с помощью текстового редактора mcedit.

```

lab7-2.lst      [----] 0 L: 1+ 0 1/225] *(0 /14458b) 0032 0x020 [X]
1
1               <I> ;----- slen -----
2               <I> ; Функция вычисления длины сообщения
3               <I> slen: .....
4 00000000 53      <I> push    ebx.....
5 00000001 89C3    <I> mov     ebx, eax.....
6               <I> .....
7               <I> nextchar: .....
8 00000003 803800  <I> cmp     byte [eax], 0...
9 00000006 7403    <I> jz      finished.....
10 00000008 40     <I> inc     eax.....
11 00000009 EBF8   <I> jmp     nextchar.....
12               <I> .....
13               <I> finished: .....
14 0000000B 29D8   <I> sub     eax, ebx
15 0000000D 58     <I> pop     ebx.....
16 0000000E C3     <I> ret     .....
17               <I> .....
18               <I> .....
19               <I> ;----- sprint -----
20               <I> ; Функция печати сообщения
21               <I> ; входные данные: mov eax,<message>
22               <I> sprint: .....
23 0000000F 52     <I> push    edx
24 00000010 51     <I> push    ecx
25 00000011 53     <I> push    ebx
26 00000012 50     <I> push    eax
27 00000013 E8E8FFFF <I> call   slen
28               <I> .....
29 00000018 89C2   <I> mov     edx, eax
30 0000001A 58     <I> pop     eax
31               <I> .....
32 0000001B 89C1   <I> mov     ecx, eax
33 0000001D B801000000 <I> mov     ebx, 1
34 00000022 B804000000 <I> mov     eax, 4
35 00000027 CD80   <I> int     80h
36               <I> .....
37 00000029 5B     <I> pop     ebx
38 0000002A 59     <I> pop     ecx
39 0000002B 5A     <I> pop     edx
40 0000002C C3     <I> ret     .....
41               <I> .....
42               <I> .....
43               <I> ;----- sprintf -----
44               <I> ; Функция печати сообщения с переводом строки
45               <I> ; входные данные: mov eax,<message>

```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Пере-тить 7Поиск 8Удалить 9МенюМС 10Выход

Рис. 11: Открытие файла

Удаляю один операнд из строчки с двумя операндами. Теперь программа вы-
дает ошибку (error: invalid combination of opcode and operands)

```
daryayasinovskaya@10:~/work/arch-pc/lab07 — /usr/bin/mc -P /var/tmp/mc-daryayasinovskaya/mc.pwd.6036
lab7-2.lst [----] 74 L:[181+23 204/226] *(12813/14548b) 0010 0x00A [*] [X]
6 00000039 35300000 C dd '50'
7 section .bss
8 00000000 <res Ah> max resb 10
9 0000000A <res Ah> B resb 10
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 000000E3 B8[00000000] mov eax,msg1
15 000000ED E81DFFFFFF call sprint
16 ; ----- Ввод 'B'
17 000000F2 B9[0A000000] mov ecx,B
18 000000F7 BA0A000000 mov edx,10
19 000000FC E842FFFFFF call sread
20 ; ----- Преобразование 'B' из символа в число
21 00000101 B8[0A000000] mov eax,B
22 00000106 E891FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B], ; запись преобразованного числа в 'B'
24 ***** error: invalid combination of opcode and operands
25 00000108 890D[35000000] mov [A], ecx ; 'ecx = A'
26 00000111 890D[00000000] mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 00000117 3B0D[39000000] cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 0000011D 7F0C jg check_B ; если 'A>C', то переход на метку 'check_B',
30 0000011F 8B0D[39000000] mov ecx,[C] ; иначе 'ecx = C'
31 00000125 890D[00000000] mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 0000012B B8[00000000] mov eax,max
35 00000130 E867FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
36 00000135 A3[00000000] mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 0000013A 8B0D[00000000] mov ecx,[max]
39 00000140 3B0D[0A000000] cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 00000146 7F0C jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 00000148 8B0D[0A000000] mov ecx,[B] ; иначе 'ecx = B'
42 0000014E 890D[00000000] mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 00000154 B8[13000000] mov eax, msg2
46 00000159 E8B1FFFFFF call sprint ; Вывод сообщения 'Наибольшее число: '
47 0000015E A1[00000000] mov eax,[max]
48 00000163 E81EFFFFFF call iprintlnF ; Вывод 'max(A,B,C)'
49 00000168 E86EFFFFFF call quit ; Выход
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Пере-тить 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 12: Файл с ошибкой

Самостоятельная работа

Написала программу для нахождения наибольшей из 3 целочисленных переменных

```

GNU nano 7.2 /home/daryayasinovskaya/work/arch-pc/lab07/lab7-3.asm
#include 'in_out.asm'
section .data
msg2 db "Наибольшее число: ",0h
A dd '79'
B dd '83'
C dd '41'

section .bss
max resb 10

section .text

global _start
_start:
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]

```

[Прочитано 46 строк]

^G Справка	^O Записать	^W Поиск	^K Вырезать	^T Выполнить	^C Позиция	M-U Отмена
^X Выход	^R ЧитФайл	^_ Замена	^U Вставить	^D Выровнять	^/ К строке	M-E Повтор

Рис. 13: Программа

Создаю исполняемый файл и проверяю программу

```

[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-3.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[daryayasinovskaya@10 lab07]$ ./lab7-3
Наибольшее число: 83
[daryayasinovskaya@10 lab07]$

```

Рис. 14: Запуск файла

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

```
GNU nano 7.2 /home/daryayasinovskaya/work/arch-pc/lab07/lab7-4.asm
#include 'in_out.asm'

SECTION .data
input1 db "Введите x: ",0h
input2 db "Введите a: ",0h

SECTION .bss
max resb 10
x resb 10
a resb 10

SECTION .text
GLOBAL _start

_start:
mov eax,input1
call sprint

mov ecx,x
mov edx,10
call sread

mov eax,x
call atoi
mov [x],eax

mov eax,input2
call sprint

mov ecx,a
mov edx,10
call sread

mov eax,a
call atoi
mov [a],eax

mov ebx, [x]
cmp [a], ebx
jl check

mov eax, [x]
mov ebx, [a]
add eax, ebx

[ Прочитано 53 строки ]
^G Справка      ^O Записать    ^W Поиск      ^K Вырезать    ^T Выполнить  ^C Позиция     M-U Отмена
^X Выход        ^R ЧитФайл    ^\ Замена     ^U Вставить    ^D Выровнять  ^_ К строке   M-E Повтор
```

Рис. 15: Программа

Создаю исполняемый файл и проверяю программу

```
[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-4.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[daryayasinovskaya@10 lab07]$ ./lab7-4
Введите x: 2
Введите a: 2
4
[daryayasinovskaya@10 lab07]$ nasm -f elf lab7-4.asm
[daryayasinovskaya@10 lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[daryayasinovskaya@10 lab07]$ ./lab7-4
Введите x: 2
Введите a: 1
10
[daryayasinovskaya@10 lab07]$
```

Рис. 16: Запуск файла

Выводы

Я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.