

# **Отчёт по лабораторной работе №7**

**Дисциплина: Архитектура компьютера**

Мишина Анастасия Алексеевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Выполнение заданий самостоятельной работы</b>	<b>17</b>
<b>4</b>	<b>Выводы</b>	<b>20</b>

# Список иллюстраций

2.1	Создание каталога и файла . . . . .	6
2.2	Копирование файла in_out.asm . . . . .	7
2.3	Работа с файлом lab7-1.asm . . . . .	7
2.4	Редактирование файла lab7-1.asm . . . . .	8
2.5	Работа с измененным файлом lab7-1.asm . . . . .	8
2.6	Работа с файлом lab7-2.asm . . . . .	9
2.7	Редактирование файла lab7-2.asm . . . . .	10
2.8	Работа с измененным файлом lab7-2.asm . . . . .	10
2.9	Запуск измененного файла . . . . .	10
2.10	Работа с файлом lab7-3.asm . . . . .	12
2.11	Работа с измененным файлом lab7-3.asm . . . . .	13
2.12	Вычисление варианта . . . . .	15
3.1	Моя программа . . . . .	19

# Список таблиц

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

## 2 Выполнение лабораторной работы

Для начала создадим каталог для программ 7-ой лабораторной работы, перейдем в нее и создадим файл lab7-1.asm (рис. 2.1).

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

Рис. 2.1: Создание каталога и файла

Вводим текст программы из листинга 7.1 в наш файл. Копируем из загрузок файл in\_out.asm в каталог lab07 (рис. 2.2). Создадим и запустим исполняемый файл (рис. 2.3). Видим в результате символ “j” (сумму ASCII кодов символов 4 и 6).

Программа lab7-1.asm:

```
%include 'in_out.asm'  
  
SECTION .bss  
buf1: RESB 80  
  
SECTION .text  
GLOBAL _start  
  
_start:  
mov eax, '6'  
mov ebx, '4'
```

```

add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintf
call quit

```

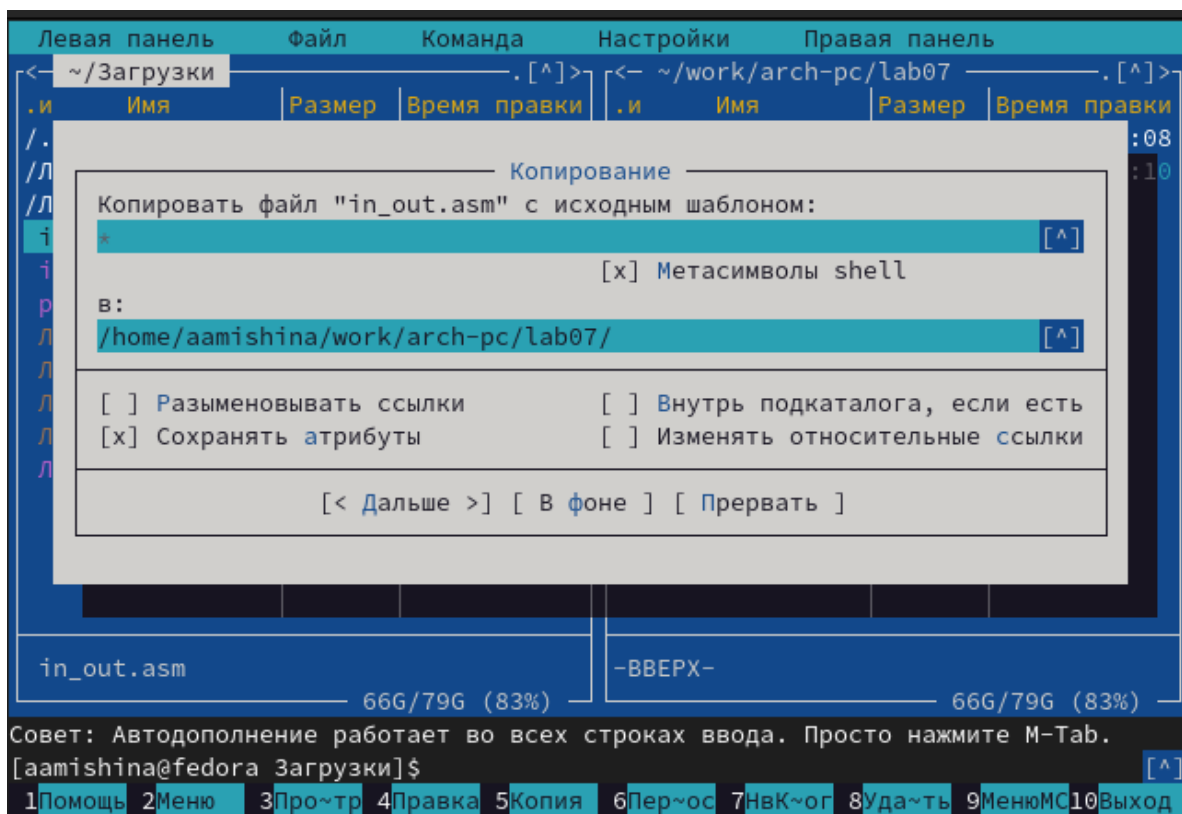


Рис. 2.2: Копирование файла in\_out.asm

```

[aamishina@fedora lab07]$ nasm -f elf lab7-1.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[aamishina@fedora lab07]$ ./lab7-1
j
[aamishina@fedora lab07]$

```

Рис. 2.3: Работа с файлом lab7-1.asm

Меняем символы на числа (рис. 2.5), пересоздаем исполняемый файл и запускаем его (рис. 2.6). Теперь программа выведет символ с кодом 10.

```

/home/aamishina/work/arch-pc/lab07/lab7-1.asm
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit

```

Рис. 2.4: Редактирование файла lab7-1.asm

```

J
[aamishina@fedora lab07]$ nasm -f elf lab7-1.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[aamishina@fedora lab07]$ ./lab7-1

[aamishina@fedora lab07]$

```

Рис. 2.5: Работа с измененным файлом lab7-1.asm

Пользуясь таблицей ASCII определяем, что код 10 соответствует символу перевода строки. Данный символ не отображается на экране, однако мы можем заметить пустую строку.

Создаем файл lab7-2.asm в каталоге ~/work/arch-pc-lab07 и вводим в него текст программы (листинг 7.2) (рис. 2.6). В результате выводится число 106 (сумма кодов символов '6' и '4', преобразованная в число).

Программа lab7-2.asm:



```
%include 'in_out.asm'

SECTION .text

GLOBAL _start

_start:

mov eax, '6'

mov ebx, '4'

add eax, ebx

call iprintLF

call quit
```

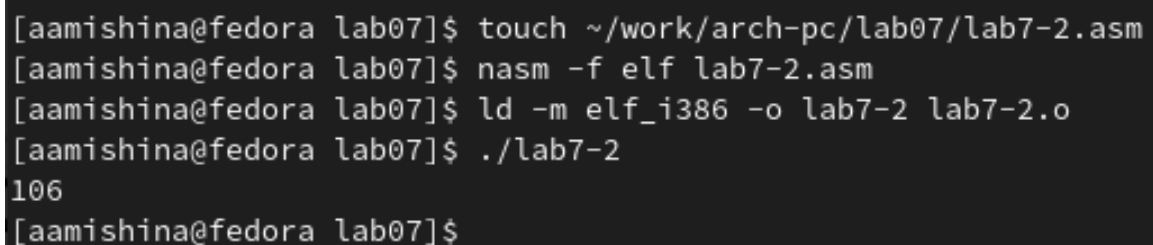
A terminal window with a dark background and light-colored text. It shows a series of commands and their outputs. The commands are: 'touch ~/work/arch-pc/lab07/lab7-2.asm', 'nasm -f elf lab7-2.asm', 'ld -m elf\_i386 -o lab7-2 lab7-2.o', and './lab7-2'. The output of the last command is '106'. The prompt is '[aamishina@fedora lab07]\$'.

Рис. 2.6: Работа с файлом lab7-2.asm

Аналогично действиям выше меняем символы на числа (рис. 2.7). Создаем и запускаем измененный исполняемый файл. Теперь программа складывает сами числа, а не их коды, результатом является число 10 (рис. 2.8).

```
/home/aamishina/work/arch-pc/lab07/lab7-2.asm
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 2.7: Редактирование файла lab7-2.asm

```
[aamishina@fedora lab07]$ nasm -f elf lab7-2.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[aamishina@fedora lab07]$ ./lab7-2
10
[aamishina@fedora lab07]$
```

Рис. 2.8: Работа с измененным файлом lab7-2.asm

Поменяем функцию `iprintLF` на `iprint`. Пересоздаем исполняемый файл, заметим, что теперь нет символа перевода строки (рис. 2.9).

```
[aamishina@fedora lab07]$ nasm -f elf lab7-2.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[aamishina@fedora lab07]$ ./lab7-2
10[aamishina@fedora lab07]$
```

Рис. 2.9: Запуск измененного файла

Создаем файл `lab7-3.asm`, вводим в него текст программы из листинга 7.3

для вычисления следующего выражения:  $(5*2+3)/3$ . Создаем и запускаем исполняемый файл (рис. 2.10).

Программа lab7-3.asm:

```
;-----  
; Программа вычисления выражения  
;-----  
  
%include 'in_out.asm' ; подключение внешнего файла  
  
SECTION .data  
  
div: DB 'Результат: ',0  
rem: DB 'Остаток от деления: ',0  
  
SECTION .text  
  
GLOBAL _start  
  
_start:  
  
; ---- Вычисление выражения  
  
mov eax,5 ; EAX=5  
mov ebx,2 ; EBX=2  
mul ebx ; EAX=EAX*EBX  
add eax,3 ; EAX=EAX+3  
  
xor edx,edx ; обнуляем EDX для корректной работы div  
mov ebx,3 ; EBX=3  
div ebx ; EAX=EAX/3, EDX=остаток от деления  
mov edi,eax ; запись результата вычисления в 'edi'  
  
; ---- Вывод результата на экран  
  
mov eax,div ; вызов подпрограммы печати  
call sprint ; сообщения 'Результат: '  
  
mov eax,edi ; вызов подпрограммы печати значения  
call iprintLF ; из 'edi' в виде символов  
  
mov eax,rem ; вызов подпрограммы печати
```

```

call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

```

[aamishina@fedora lab07]$ touch ~/work/arch-pc/lab07/lab7-3.asm
[aamishina@fedora lab07]$ nasm -f elf lab7-3.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[aamishina@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
[aamishina@fedora lab07]$

```

Рис. 2.10: Работа с файлом lab7-3.asm

Изменяем код для подсчета выражения:  $(4*6+2)/5$ . Проверяем работу программы (рис. 2.11).

```

;-----
; Программа вычисления выражения
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,4 ; EAX=4
mov ebx,6 ; EBX=6
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+2

```

```

xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=5
div ebx ; EAX=EAX/5, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

```

[aamishina@fedora lab07]$ nasm -f elf lab7-3.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[aamishina@fedora lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1
[aamishina@fedora lab07]$

```

Рис. 2.11: Работа с измененным файлом lab7-3.asm

Теперь создадим файл variant.asm для вычисления варианта по номеру студенческого билета. Вводим в новый файл текст программы из листинга 7.4. Создаем и запускаем исполняемый файл, вводим туда свой номер студ. билета, получаем вариант 13 (рис. 2.12). Аналитически также получаем вариант 13.

Программа variant.asm:

```

;-----
; Программа вычисления варианта

```

```

;-----
#include 'in_out.asm'

SECTION .data
msg: DB 'Введите No студенческого билета: ',0
rem: DB 'Ваш вариант: ',0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
mov ecx, x
mov edx, 80
call sread
mov eax,x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, `eax=x`
xor edx,edx
mov ebx,20
div ebx
inc edx
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit

```

```
[aamishina@fedora lab07]$ touch ~/work/arch-pc/lab07/variant.asm
[aamishina@fedora lab07]$ nasm -f elf variant.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o variant variant.o
[aamishina@fedora lab07]$ ./variant
Введите No студенческого билета:
1132226532
Ваш вариант: 13
[aamishina@fedora lab07]$
```

Рис. 2.12: Вычисление варианта

Ответы на вопросы по программе: 1. Строки, отвечающие за вывод сообщение “Ваш вариант:”

```
mov eax,rem
call sprint
```

2. Конструкция “mov esx, x” используется для того, чтобы положить в регистр esx адрес x. Конструкция “mov edx, 80” нужна, чтобы записать в регистр edx длину вводимого сообщения. “call sread” позволяет вызвать подпрограмму из внешнего файла для ввода сообщения с клавиатуры.
3. Инструкция “call atoi” используется для преобразования ASCII-кода символа в целое число и записи результата в регистр eax.
4. За вычисление варианта отвечают следующие строки кода:

```
xor edx,edx ; обнуление EDX для корректной работы div
mov ebx,20 ; EBX = 20
div ebx ; EAX = EAX/20, EDX - остаток от деления
inc edx ; EDX = EDX + 1
```

5. При выполнении инструкции “div edx” остаток от деления записывается в регистр edx.
6. Инструкция “inc edx” позволяет увеличить значение регистра edx на 1.

7. Строки, отвечающие за вывод на экран результата вычислений:

```
mov eax,edx  
call iprintLF
```



## 3 Выполнение заданий самостоятельной работы

Согласно моему варианту (13) мне следовало запрограммировать подсчет функции  $(8x+6)*10$ . Создаю файл mytask.asm, пишу код, создаю и запускаю исполняемый файл. Проверяю значения при  $x = 1$  и  $x = 4$ . Программа отрабатывает верно (рис. 3.1).

Программа mytask.asm:

```
;-----  
; Программа вычисления выражения  
;-----  
%include 'in_out.asm' ; подключение внешнего файла  
SECTION .data  
msg: DB 'Введите x для подсчета выражения (8x+6)*10: ',0  
res: DB 'Результат: ',0  
  
SECTION .bss  
x: RESB 80  
  
SECTION .text  
GLOBAL _start  
_start:
```

```
mov eax, msg  
call sprint
```

```
mov ecx, x  
mov edx, 80  
call sread
```

```
mov eax, x  
call atoi
```

```
mov ebx, 8  
mul ebx
```

```
add eax, 6
```

```
mov ebx, 10  
mul ebx
```

```
mov edi, eax
```

```
mov eax, res  
call sprint
```

```
mov eax, edi  
call iprintLF
```

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

```
[aamishina@fedora lab07]$ nasm -f elf mytask.asm
[aamishina@fedora lab07]$ ld -m elf_i386 -o mytask mytask.o
[aamishina@fedora lab07]$ ./mytask
Введите x для подсчета выражения  $(8x+6)*10$ : 1
Результат: 140
[aamishina@fedora lab07]$ ./mytask
Введите x для подсчета выражения  $(8x+6)*10$ : 4
Результат: 380
[aamishina@fedora lab07]$
```

Рис. 3.1: Моя программа

## 4 Выводы

В ходе выполнения данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM. Вся моя работа была записана и прокомментирована мной в данной лабораторной.