

Отчет по лабораторной работе №6

Дисциплина: архитектура компьютера

Морошан Матвей Корнелиович

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Символьные и численные данные в NASM	6
3.2	Выполнение арифметических операций в NASM	9
3.2.1	Ответы на вопросы	12
3.3	Выполнение заданий для самостоятельной работы	13
4	Выводы	15

Список иллюстраций

3.1	Создание каталога, переход в него и создание файла	6
3.2	Копирование файла in_out.asm	6
3.3	Текст программы lab6-1.asm	7
3.4	Создание и запуск исполняемого файла	7
3.5	Измененный текст программы	7
3.6	Создание и запуск исполняемого файла	7
3.7	Создание файла lab6-2.asm	8
3.8	Текст программы lab6-2.asm	8
3.9	Создание исполняемого файла и запуск его	8
3.10	Измененный текст программы	8
3.11	Создание и запуск файла	9
3.12	Замена iprintLF на iprint	9
3.13	Создание файла и его запуск	9
3.14	Создание файла lab6-3.asm	9
3.15	Текст программы lab6-3.asm	10
3.16	Создание и запуск исполняемого файла	10
3.17	Текст программы для вычисления выражения $f(x)$	11
3.18	Создание и запуск исполняемого файла	11
3.19	Создание файла variant.asm	11
3.20	Текст программы variant.asm	12
3.21	Создание исполняемого файла и запуск	12
3.22	Создание файла для самостоятельной работы	13
3.23	Текст программы для заданной функции	14
3.24	Результат работы	14

1 Цель работы

Целью данной лабораторной работы является освоение арифметических инструкций языка ассемблера NASM

2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

3 Выполнение лабораторной работы

3.1 Символьные и численные данные в NASM

Создаю каталог для программ лабораторной работы №6, перехожу в него и создаю файл lab6-1.asm (рис. 3.1)

```
mkmoroshan@dk2n21 ~ $ mkdir ~/work/arch-pc/lab06
mkmoroshan@dk2n21 ~ $ cd ~/work/arch-pc/lab06
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ touch lab6-1.asm
```

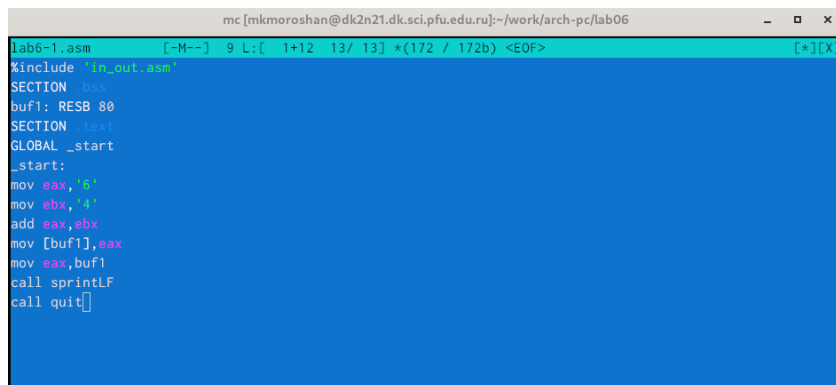
Рис. 3.1: Создание каталога, переход в него и создание файла

Копирую файл in_out.asm в каталог для лабораторной работы №6 для дальнейшей работы (рис. 3.2)

```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ cp ~/Загрузки/in_out.asm in_out.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm
```

Рис. 3.2: Копирование файла in_out.asm

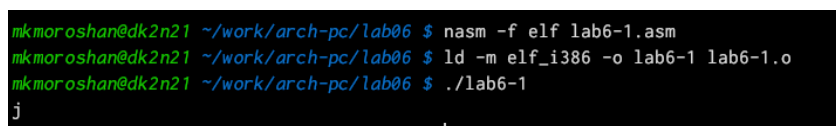
Открываю файл lab6-1.asm и ввожу текст программы из листинга 6.1 (рис. 3.3)



```
lab6-1.asm [-M--] 9 L: [ 1+12 13/ 13] *(172 / 172b) <EOF> [*][X]
#include "in_out.asm"
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, '5'
mov ebx, '4'
add eax, ebx
mov [buf1], eax
mov eax, buf1
call sprintLF
call quit[]
```

Рис. 3.3: Текст программы lab6-1.asm

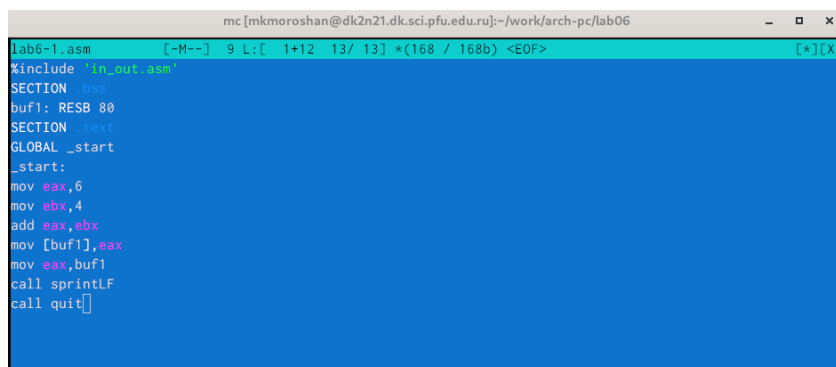
Создаю исполняемый файл и запускаю его (рис. 3.4)



```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-1
j
```

Рис. 3.4: Создание и запуск исполняемого файла

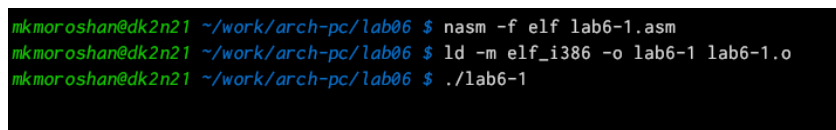
Изменяю текст программы (рис. 3.5)



```
lab6-1.asm [-M--] 9 L: [ 1+12 13/ 13] *(168 / 168b) <EOF> [*][X]
#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. 3.5: Измененный текст программы

Снова создаю исполняемый файл и запускаю его. Символ не отображается на экране (рис. 3.6)



```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-1
```

Рис. 3.6: Создание и запуск исполняемого файла

Создаю файл lab6-2.asm (рис. 3.7)

```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ touch lab6-2.asm
```

Рис. 3.7: Создание файла lab6-2.asm

Ввожу в него текст программы из листинга 6.2 (рис. 3.8)

```
lab6-2.asm  [-M--]  9 L: 1+ 8  9/  9] *(117 / 117b) <EOF>  [*][X]
#include "in_out.asm"
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF
call quit[]
```

Рис. 3.8: Текст программы lab6-2.asm

Создаю исполняемый файл и запускаю его (рис. 3.9)

```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-2
106
```

Рис. 3.9: Создание исполняемого файла и запуск его

Изменяю текст программы (рис. 3.10)

```
lab6-2.asm  [-M--]  9 L: 1+ 8  9/  9] *(113 / 113b) <EOF>  [*][X]
#include "in_out.asm"
SECTION .text
GLOBAL _start
_start:
mov eax, 6
mov ebx, 4
add eax, ebx
call iprintLF
call quit[]
```

Рис. 3.10: Измененный текст программы

Создаю исполняемый файл и запускаю его (рис. 3.11)

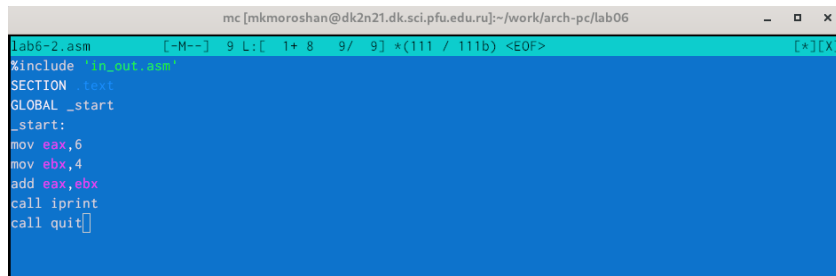

```

mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-2
10

```

Рис. 3.11: Создание и запуск файла

Изменяю `iprintLF` на `iprint` (рис. 3.12)



```

lab6-2.asm  [-M--]  9 L: 1+ 8  9/  9] *(111 / 111b) <EOF>  [*][X]
#include "in_out.asm"
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit

```

Рис. 3.12: Замена `iprintLF` на `iprint`

Вновь создаю исполняемый файл и запускаю его. `iprint` не добавляет переноса строки в отличие от `iprintLF` (рис. 3.13)

```

mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-2
10mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ 

```

Рис. 3.13: Создание файла и его запуск

3.2 Выполнение арифметических операций в NASM

Создаю файл `lab6-3.asm` (рис. 3.14)

```

mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ touch lab6-3.asm

```

Рис. 3.14: Создание файла `lab6-3.asm`

Ввожу текст программы из листинга 6.3 и ввожу в `lab6-3.asm` (рис. 3.15)

```

lab6-3.asm      [~M~~] 41 L: [ 1+25 26/ 26] *(1236/1236b) <EOF>      [*][X]
%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

```

Рис. 3.15: Текст программы lab6-3.asm

Создаю исполняемый файл и запускаю его (рис. 3.16)

```

mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1

```

Рис. 3.16: Создание и запуск исполняемого файла

Изменяю текст программы для вычисления выражения $f(x)=(4*6+2)/5$ (рис. 3.17)

```

mc [mkmoroshan@dk2n21.dk.sci.pfu.edu.ru:~/work/arch-pc/lab06
lab6-3.asm [-M--] 41 L: [ 1+25 26/ 26] *(1236/1236b) <EOF> [*][X]
%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

```

Рис. 3.17: Текст программы для вычисления выражения $f(x)$

Создаю исполняемый файл и запускаю его (рис. 3.18)

```

mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1

```

Рис. 3.18: Создание и запуск исполняемого файла

Создаю файл variant.asm (рис. 3.19)

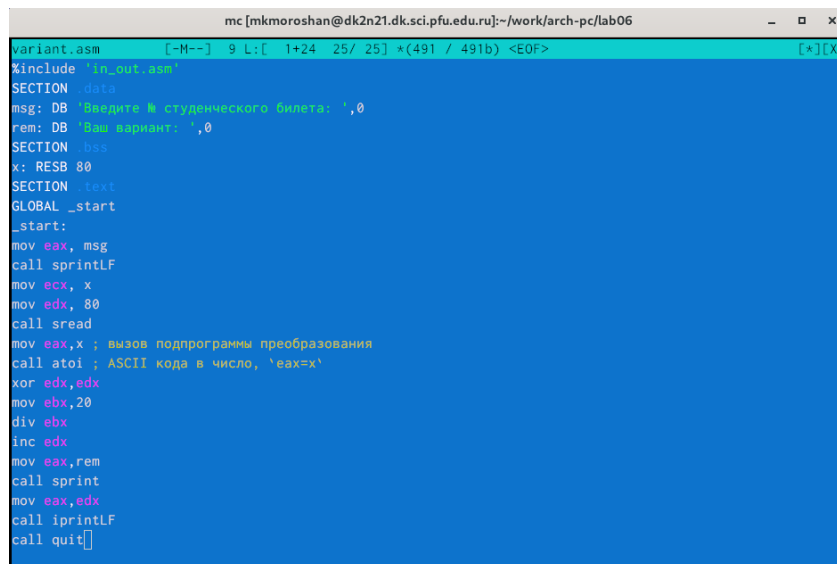
```

mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ touch variant.asm

```

Рис. 3.19: Создание файла variant.asm

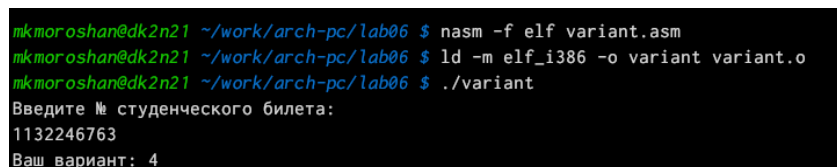
Ввожу текст программы из листинга 6.4 в файл variant.asm (рис. 3.20)



```
variant.asm [-M--] 9 L: [ 1+24 25/ 25] *(491 / 491b) <EOF> [*][X]
%include "in_out.asm"
SECTION .data
msg: DB "Введите № студенческого билета: ",0
rem: DB "Ваш вариант: ",0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
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[]
```

Рис. 3.20: Текст программы variant.asm

Создаю исполняемый файл и запускаю его (рис. 3.21)



```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1132246763
Ваш вариант: 4
```

Рис. 3.21: Создание исполняемого файла и запуск

3.2.1 Ответы на вопросы

1. За вывод на экран сообщения 'Ваш вариант:' отвечают строки листинга:

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

2. Инструкция `mov ecx, x` используется, чтобы вложить адрес вводимой строки `x` в регистр `ecx`

Инструкция `mov edx, 80` - это запись в регистр `edx` длины вводимой строки

Инструкция `call sread` - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры

3. Инструкция `call atoi` используется для вызова подпрограммы из внешнего файла, которая преобразует `ascii`-код символа в целое число и записывает результат в регистр `eax`

4. За вычисление варианта отвечают строки:

```
xor edx,edx
mov ebx,20
div ebx
inc edx
```

5. Остаток от деления при выполнении инструкции `div ebx` записывается в регистр `edx`

6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1

7. За вывод на экран результата вычислений отвечают следующие строки:

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

3.3 Выполнение заданий для самостоятельной работы

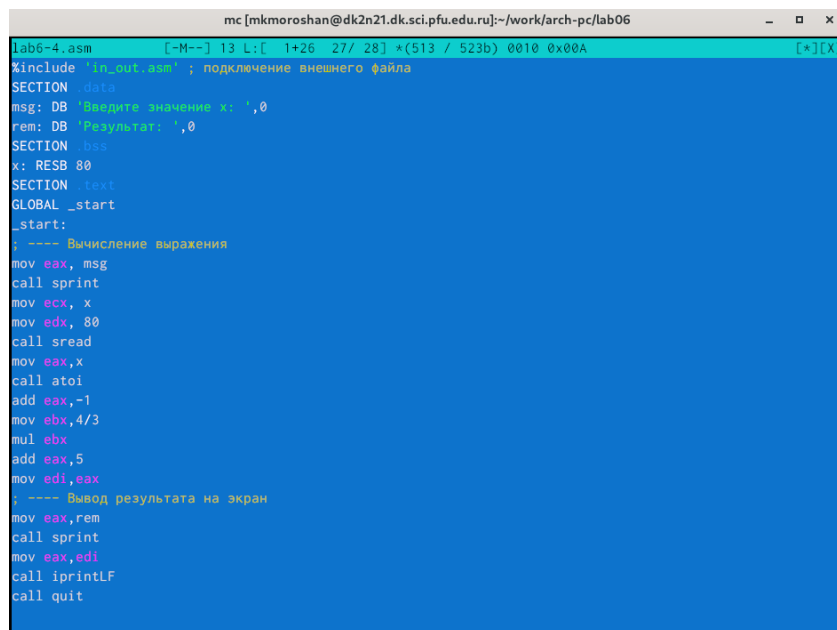
Создаю файл для выполнения в нем заданий для самостоятельной работы (рис. 3.22)



```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ touch lab6-4.asm
```

Рис. 3.22: Создание файла для самостоятельной работы

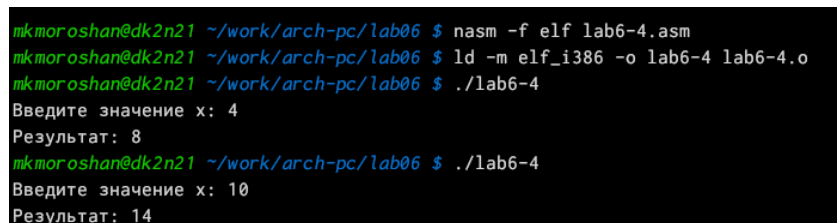
Вписываю текст программы в файл для $f(x)=4/3(x-1)+5$ (рис. 3.23)



```
lab6-4.asm [-M--] 13 L:[ 1+26 27/ 28] *(513 / 523b) 0010 0x00A [*][X]
#include "in_out.asm" ; подключение внешнего файла
SECTION .data
msg: DB 'Введите значение x: ',0
rem: DB 'Результат: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax, msg
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
add eax, -1
mov ebx, 4/3
mul ebx
add eax, 5
mov edi, eax
; ---- Вывод результата на экран
mov eax, rem
call sprint
mov eax, edi
call iprintLF
call quit
```

Рис. 3.23: Текст программы для заданной функции

Создаю файл и запускаю его. Ввожу значения $x=4$ и $x=10$ и получаю результат (рис. 3.24)



```
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ nasm -f elf lab6-4.asm
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-4 lab6-4.o
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-4
Введите значение x: 4
Результат: 8
mkmoroshan@dk2n21 ~/work/arch-pc/lab06 $ ./lab6-4
Введите значение x: 10
Результат: 14
```

Рис. 3.24: Результат работы

4 Выводы

При выполнении данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM