

# **Отчет по лабораторной работе No.6**

**Дисциплины: Архитектура компьютера**

Ракутуманандзара Цантамписедрана Сарубиди

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>18</b>
	<b>Список литературы</b>	<b>19</b>

# Список иллюстраций

3.1	рис 1	7
3.2	рис 2	7
3.3	рис 3	8
3.4	рис 4	8
3.5	рис 5	9
3.6	рис 6	9
3.7	рис 7	9
3.8	рис 8	10
3.9	рис 9	10
3.10	рис 10	10
3.11	рис 11	11
3.12	рис 12	11
3.13	рис 13	11
3.14	рис 14	12
3.15	рис 15	12
3.16	рис 16	12
3.17	рис 17	13
3.18	рис 18	13
3.19	рис 19	13
3.20	рис 20	14
3.21	рис 21	15
3.22	рис 22	15
3.23	рис 23	16

## **Список таблиц**

# 1 Цель работы

Целью данной работы является освоение арифметических инструкций на языке 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

### 1. Символьные и численные данные в NASM

Я создам каталог для программ лабораторных работ 6 с помощью команды `mkdir`, зайду в него с помощью команды `cd` и создам файл `lab6-1.asm` с помощью команды `touch`(рис 1)

```
tsanta@tsanta-VirtualBox:~$ mkdir ~/work/os-intro/lab06
tsanta@tsanta-VirtualBox:~$ cd ~/work/os-intro/lab06
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ touch lab6-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.1: рис 1

Я скопирую файл `in_out.asm` в текущий каталог с помощью команды `cp`, потому что буду использовать его в программах(рис 2)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ cp ~/work/os-intro/lab05/in_out.
asm ~/work/os-intro/lab06
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ls
in_out.asm  lab6-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.2: рис 2

Открою созданный файл `lab6-1.asm` и скопирую в него программу вывода значения регистра `eax`(рис 3)



```
ta.lab6.md | in_out.asm | lab6-1.asm x

%include 'in_out.asm'

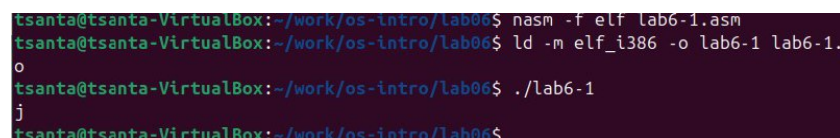
SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax, '6'
mov ebx, '4'
add eax, ebx
mov [buf1], eax
mov eax, buf1
call sprintf
```

Рис. 3.3: рис 3

Я создам исполняемый файл программы и запущу его. Программа выведет символ j, поскольку программа выводит символ, соответствующий сумме ASCII двоичных кодов символов 4 и 6(рис 4)



```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ nasm -f elf lab6-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-1
j
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.4: рис 4

Я заменю символы '6' и '4' в тексте программы на цифры 6 и 4(рис 5)





```
Open ▾  tsanta.lab6.md  in_out.asm  lab6-1.asm x
lab6-1.asm
~/work/os-intro/lab06

#include 'in_out.asm'

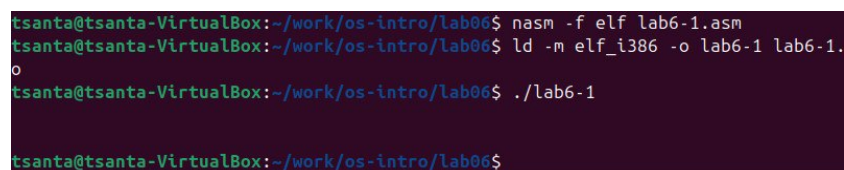
SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintf
```

Рис. 3.5: рис 5

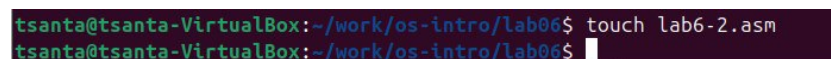
Я создам новый исполняемый файл программы и запущу его. Теперь отображается символ с кодом 10, это символ перевода строки(рис 6)



```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ nasm -f elf lab6-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-1
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.6: рис 6

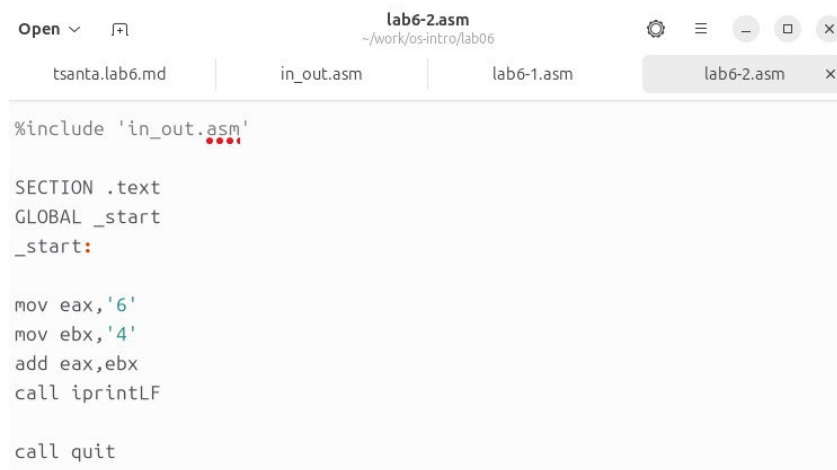
Я создам новый файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 с помощью команды touch(рис 7)



```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ touch lab6-2.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.7: рис 7

Я открою вновь созданный файл и скопирую в него заданный текст программы(рис 8)



```
Open ▾  ↵  lab6-2.asm
~/work/os-intro/lab06

tsanta.lab6.md | in_out.asm | lab6-1.asm | lab6-2.asm x

%include 'in_out.asm'

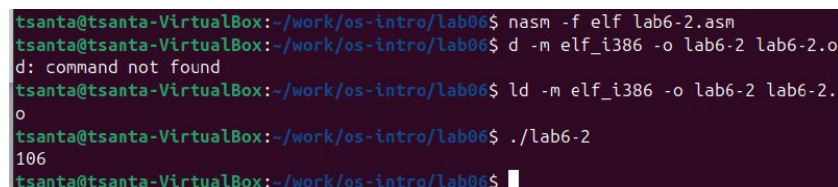
SECTION .text
GLOBAL _start
_start:

mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF

call quit
```

Рис. 3.8: рис 8

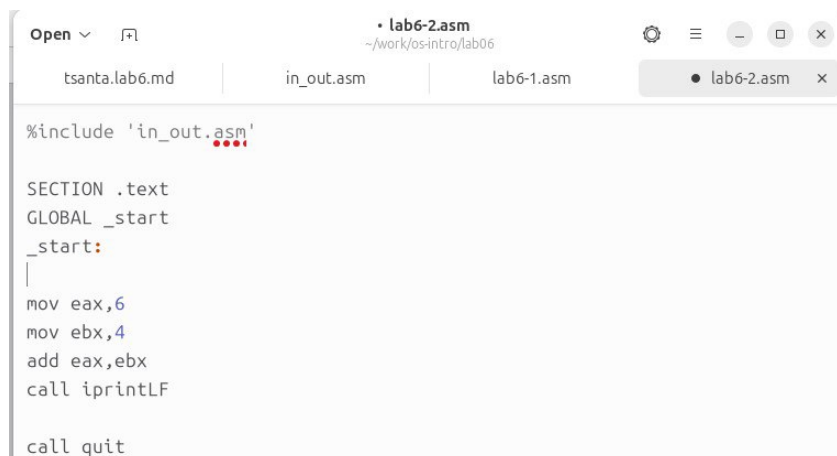
Я создаю и запускаю исполняемый файл lab6-2. Теперь на выходе будет число 106, поскольку функция iprintLF позволяет программе выводить точное число вместо символа ASCII(рис 9)



```
tsanta@tsanta-VirtualBox: ~/work/os-intro/lab06$ nasm -f elf lab6-2.asm
tsanta@tsanta-VirtualBox: ~/work/os-intro/lab06$ d -m elf_i386 -o lab6-2 lab6-2.o
d: command not found
tsanta@tsanta-VirtualBox: ~/work/os-intro/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
tsanta@tsanta-VirtualBox: ~/work/os-intro/lab06$ ./lab6-2
106
tsanta@tsanta-VirtualBox: ~/work/os-intro/lab06$
```

Рис. 3.9: рис 9

Теперь я заменяю символы '6' и '4' в тексте программы на цифры 6 и 4(рис 10)



```
Open ▾  ↵  lab6-2.asm
~/work/os-intro/lab06

tsanta.lab6.md | in_out.asm | lab6-1.asm | lab6-2.asm 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: рис 10

Как и ранее, я создаю и запускаю исполняемый файл lab6-2. Вывод равен 10, поскольку функция `iprintLF` позволяет программе выводить точное число вместо символа ASCII(рис 11)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ nasm -f elf lab6-2.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-2
10
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.11: рис 11

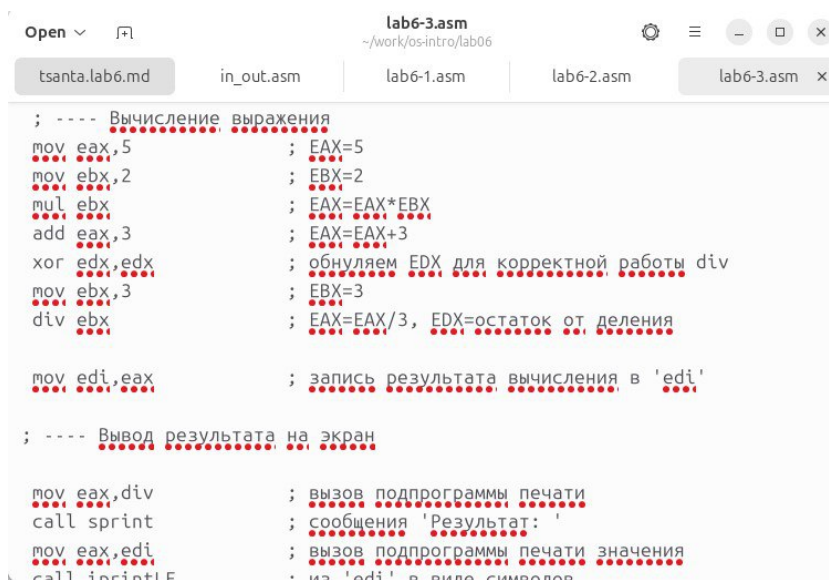
## 2. Выполнение арифметических операций в NASM

Я создам еще один файл lab6-3.asm в каталоге `~/work/arch-pc/lab06` с помощью команды `touch`(рис 12)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ touch lab6-3.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.12: рис 12

В созданный файл ввожу текст программы для вычисления значения выражения  $f(x) = (5 * 2 + 3)/3$ (рис 13)



```

; ---- 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

```

Рис. 3.13: рис 13

Создаю исполняемый файл и запускаю его(рис 14)

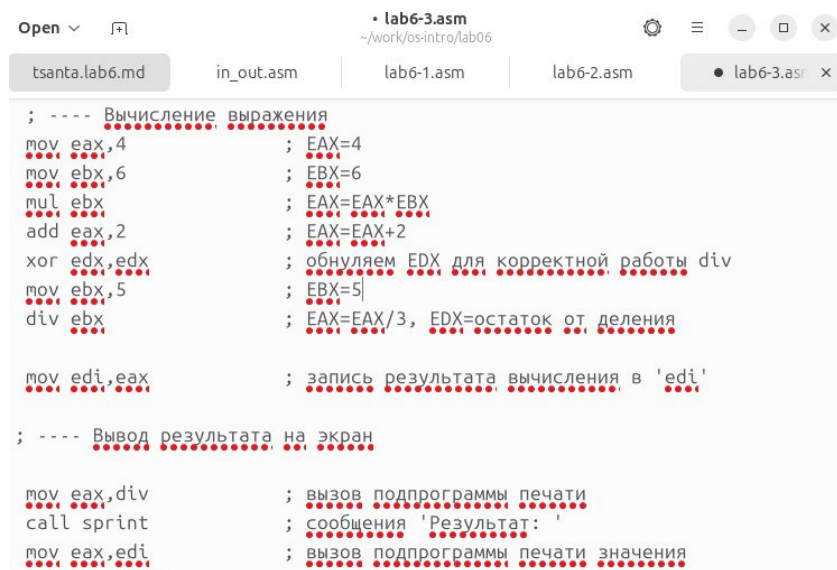
```

tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ touch lab6-3.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ nasm -f elf lab6-3.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$

```

Рис. 3.14: рис 14

Я изменю программу так, чтобы она вычисляла значение выражения  $f(x) = (4 * 6 + 2) / 5$  (рис 15)



```

; ---- 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/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

```

Рис. 3.15: рис 15

Теперь я создаю исполняемый файл и запускаю его (рис 16)

```

tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ nasm -f elf lab6-3.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$

```

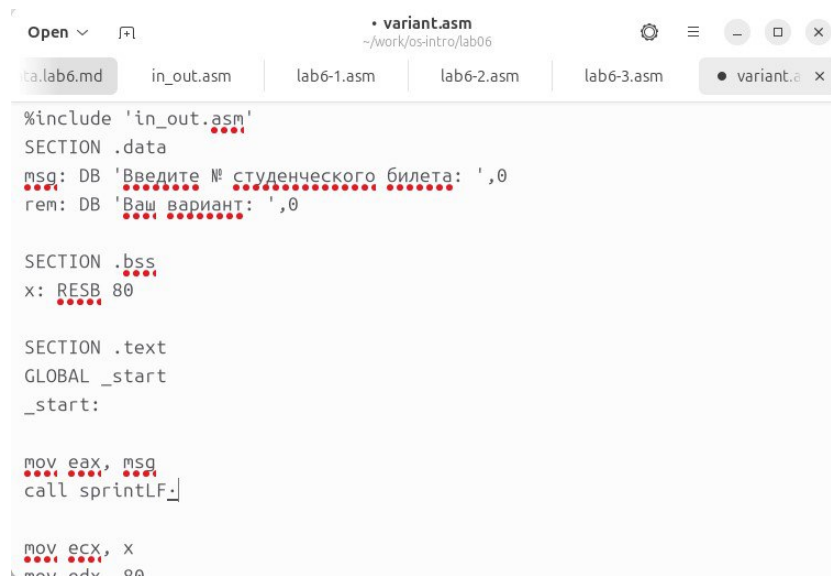
Рис. 3.16: рис 16

Я создам файл variant.asm в каталоге ~/work/arch-pc/lab06 с помощью команду touch (рис 17)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ touch variant.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.17: рис 17

Скопирую текст программы в файл для расчета варианта задания по студенческому билету(рис 18)



```
Open ▾  variant.asm
~/work/os-intro/lab06
ta.lab6.md  in_out.asm  lab6-1.asm  lab6-2.asm  lab6-3.asm  variant.a 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 edi, 80
```

Рис. 3.18: рис 18

Я создам и запущу исполняемый файл. Ввожу с клавиатуры номер студенческого билета, программа показывает, что мой вариант-12 (рис 19)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./variant
Введите № студенческого билета:
1032239571
Ваш вариант: 12
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.19: рис 19

## 2.1 Ответы на вопросы по программе

1.mov eax,rem

call sprint

2. `mov ecx,x` – используется для помещения адреса входной строки `x` в регистр `ecx`.

`mov edx,80` – записывает длину входной строки в регистр `edx`

`call sread` – вызов подпрограммы из внешнего файла, позволяющей ввести сообщение с клавиатуры

3. `call atoi` используется для преобразования `ascii`-кода символа в целое число и записи

4. `xor edx,edx`

`mov ebx,20`

`div ebx`

`inc edx`

5. Остаток от деления записывается в регистр `edx`

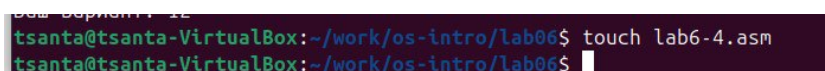
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1

7. `mov eax,edx`

`call iprintLF`

### 3. Выполнение заданий для самостоятельной работы

Я создам файл `lab6-4.asm` с помощью команды `touch` (рис 20)



```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ touch lab6-4.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.20: рис 20

Созданный файл открою для редактирования, введу в него текст программы для вычисления значения выражения  $(8 \cdot x - 6) / 2$  (рис. 4.24). Это выражение было под вариантом 12 (рис 21)

```
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 ; обнуляем EDX для корректной работы div
mov ebx, 8 ; EBX=8
mul ebx ; eax=eax*8
sub eax, 6 ; EAX=EAX-6
mov ebx, 2 ; EBX=2
div ebx ; EAX=EAX/EBX
mov edi, eax ; запись результата вычисления в 'edi'
mov eax, div ; вызов подпрограммы печати
call sprintf ; сообщения 'Результат: '
mov eax, edi ; вызов подпрограммы печати значения
call sprintf ; из 'edi' в виде символов
mov eax, rem ; вызов подпрограммы печати
call sprintf ; сообщения 'Остаток от деления: '
mov eax, edx ; вызов подпрограммы печати значения
call sprintf ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 3.21: рис 21

Я создам и запущу исполняемый файл. Когда вы вводите значение 1, выход 1 (рис 22)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ nasm -f elf lab6-4.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 lab6-4.o -o lab6-4
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-4
Введите значение x:
1
Результат: 1
Остаток от деления: 0
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$
```

Рис. 3.22: рис 22

Я запускаю исполняемый файл еще раз, чтобы проверить работу программы с другим входным значением, на этот раз я буду использовать 5, а на выходе должно быть 17. Программа работала корректно(рис 23)

```

tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 lab6-4.o -o lab6-4
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ld -m elf_i386 lab6-4.o -o lab6-4
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$ ./lab6-4
Введите значение x:
5
Результат: 17
Остаток от деления: 0
tsanta@tsanta-VirtualBox:~/work/os-intro/lab06$

```

Рис. 3.23: рис 23

Листинг 3.1. Программа для вычисления значения выражения  $(8x - 6)/2$

```

#include 'in_out.asm'

SECTION .data
msg: DB 'Введите значение x: ',0
div:DB 'Результат: ',0
rem:DB 'Остаток от деления: ',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 ; обнуляем EDX для корректной работы div
mov ebx,8 ; EBX=8
mul ebx ; eax=eax*8
sub eax,6 ; EAX=EAX-6
mov ebx,2 ; EBX=2

```



```
div ebx      ; EAX=EAX/EBX
mov edi,eax ; запись результата вычисления в 'edi'
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
```

## **4 Выводы**

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.

# Список литературы

1. Архитектура ЭВМ
2. Таблица ASCII