

# **Отчет по лабораторной работе №6**

**Дисциплина: Архитектура компьютера**

Чипурной Михаил Евгеньевич

# **Содержание**

<b>1 Цель работы</b>	<b>5</b>
<b>2 Выполнение лабораторной работы</b>	<b>6</b>
2.1 Символьные и численные данные в NASM . . . . .	6
2.2 Выполнение арифметических операций в NASM . . . . .	10
2.3 Ответы на вопросы . . . . .	12
<b>3 Задание для самостоятельной работы</b>	<b>14</b>
<b>4 Выводы</b>	<b>16</b>

# Список иллюстраций

2.1	Создаем каталог с помощью команды <code>mkdir</code> и файл с помощью команды <code>touch</code> . . . . .	6
2.2	. . . . .	6
2.3	Копируем файл в <code>Midnight Commander</code> . . . . .	7
2.4	Запускаем файл и смотрим на его работу . . . . .	7
2.5	Изменяем файл . . . . .	7
2.6	Запускаем файл и смотрим на его работу . . . . .	8
2.7	Создаем файл . . . . .	8
2.8	Заполняем файл . . . . .	8
2.9	Смотрим на работу программы . . . . .	8
2.10	Изменяем файл . . . . .	9
2.11	Смотрим на работу программы . . . . .	9
2.12	Изменяем файл . . . . .	9
2.13	Смотрим на работу программы . . . . .	10
2.14	Создаем файл командой <code>touch</code> . . . . .	10
2.15	Заполняем файл . . . . .	10
2.16	Смотрим на результат работы программы . . . . .	10
2.17	Редактируем файл . . . . .	11
2.18	Рис. 3.17: Смотрим на результат работы программы . . . . .	11
2.19	Создаем файл командой <code>touch</code> . . . . .	11
2.20	Заполняем файл . . . . .	12
2.21	Смотрим на результат работы программы . . . . .	12
3.1	Создаем файл . . . . .	14
3.2	Заполняем файл . . . . .	14
3.3	Проверяем работу программы . . . . .	15
3.4	Проверяем работу программы . . . . .	15

# **Список таблиц**

# **1 Цель работы**

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Выполнение лабораторной работы

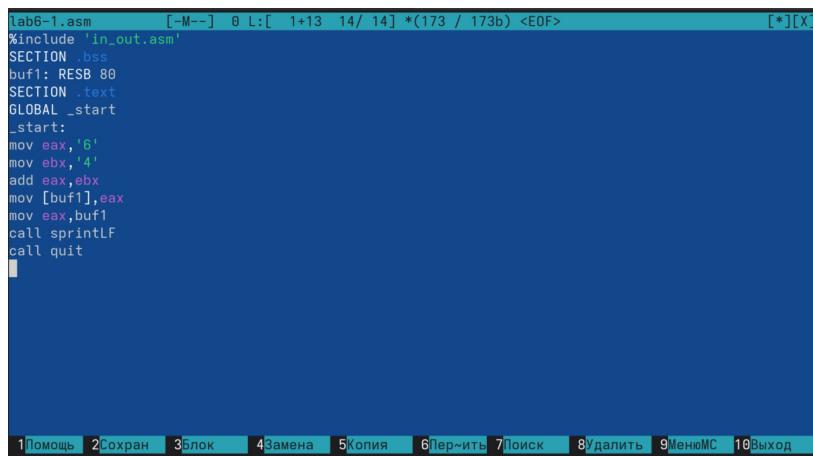
## 2.1 Символьные и численные данные в NASM

Создаем каталог для программ ЛБ6, и в нем создаем файл (Рисунок 2.1).

```
mechipurnoyj@fedora:~$ mkdir ~/work/arch-pc/lab06
mechipurnoyj@fedora:~$ cd ~/work/arch-pc/lab06
mechipurnoyj@fedora:~/work/arch-pc/lab06$ touch lab6-1.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$ █
```

Рисунок 2.1: Создаем каталог с помощью команды `mkdir` и файл с помощью команды `touch`

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 6.1 (Рисунок 2.2).



```
lab6-1.asm      [-M--]  0 L:[ 1+13 14/ 14] *(173 / 173b) <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
█
```

Рисунок 2.2

Копируем файл `in_out.asm` в новый каталог `lab06` (Рисунок 2.3).

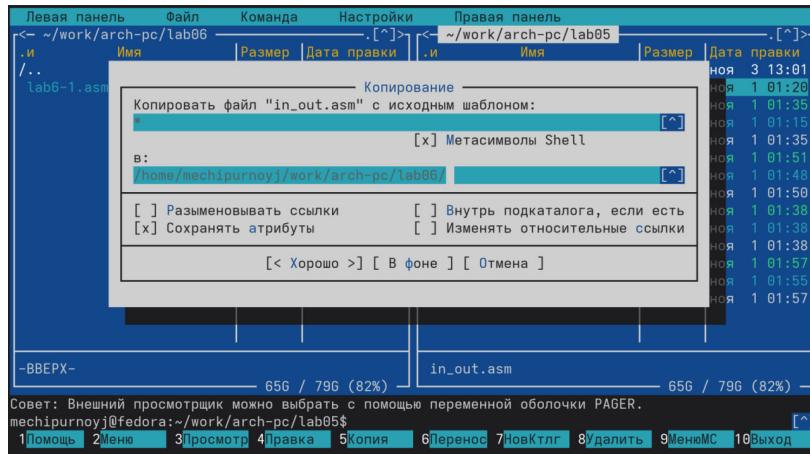


Рисунок 2.3: Копируем файл в Midnight Commander

Создаем исполняемый файл и запускаем его (Рисунок 2.4).

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

Рисунок 2.4: Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и убираем кавычки с числовых значений (Рисунок 2.5).

```
lab6-1.asm [-M--] 9 L:[ 1+2 13/ 14 ] *(168 / 169b) 0010 0x00A [*][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
```

Рисунок 2.5: Изменяем файл

Создаем исполняемый файл и запускаем его (Рисунок 2.6).

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

mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 2.6: Запускаем файл и смотрим на его работу

Создаем новый файл в каталоге (Рисунок 2.7).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ touch lab6-2.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o  lab6-2.asm
```

Рисунок 2.7: Создаем файл

Заполняем файл в соответствии с листингом 6.2 (Рисунок 2.8).

The screenshot shows the assembly code for `lab6-2.asm`. The code includes an include directive for `in_out.asm`, defines a section `.text`, and contains a global symbol `_start`. Inside `_start`, there is a sequence of instructions: `mov eax,'6'`, `mov ebx,'4'`, `add eax,ebx`, `call iprintLF`, and `call quit`. The status bar at the bottom of the window shows various menu options in Russian: Помощь, Сохран, Блок, Замена, Копия, Пер~ить, Поиск, Удалить, МенюМС, Выход.

```
lab6-2.asm      [-M--]  0 L:[ 1+10 11/ 12] *(110 / 120b) 0010 0x00A      [*][X]
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

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

    call quit
```

Рисунок 2.8: Заполняем файл

Создаем исполняемый файл и запускаем его (Рисунок 2.9).

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

Рисунок 2.9: Смотрим на работу программы

Снова открываем файл для редактирования и убираем кавычки с числовых значений (Рисунок 2.10).

```
lab6-2.asm [-M--] 8 L:[ 1+ 7 8/ 12] *(78 / 116b) 0052 0x034 [*][X]
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
    mov eax,6
    mov ebx,4
    add eax,ebx
    call iprintLF

    call quit
```

Рисунок 2.10: Изменяем файл

Создаем исполняемый файл и запускаем его (Рисунок 2.11).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ./lab6-2
10
mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 2.11: Смотрим на работу программы

Снова открываем файл для редактирования и меняем iprintLF на iprint (Рисунок 2.12).

```
lab6-2.asm [-M--] 11 L:[ 1+ 9 10/ 12] *(103 / 114b) 0010 0x00A [*][X]
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
    mov eax,6
    mov ebx,4
    add eax,ebx
    call iprint
```

Рисунок 2.12: Изменяем файл

Создаем исполняемый файл и запускаем его (Рисунок 2.13).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ./lab6-2
10mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 2.13: Смотрим на работу программы

Вывод функций iprintLF и iprint отличаются только тем, что LF переносит на новую строку.

## 2.2 Выполнение арифметических операций в NASM

Создаем новый файл в каталоге (Рисунок 2.14).

```
10mechipurnoyj@fedora:~/work/arch-pc/lab06$ touch lab6-3.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 2.14: Создаем файл командой touch

Открываем файл и редактируем в соответствии с листингом 6.3 (Рисунок 2.15).

```
lab6-3.asm [---] 0 L:[ 6+26 32/ 34 ] *(1103/1245b) 0099 0x0003
rem: DB 'Остаток от деления: ',06
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 print ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call print ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintF ; из 'edx' (остаток) в виде символов
```

Рисунок 2.15: Заполняем файл

Создаем исполняемый файл и запускаем его (Рисунок 2.16).

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

Рисунок 2.16: Смотрим на результат работы программы

Открываем файл и редактируем его для вычисления выражения  $f(x) = (4 * 6 + 2)/5$  (Рисунок 2.17).

```
lab6-3.asm      [-M--]  9 L:[ 4+27 31/ 31] *(350 / 350b) <EOF> [•][X]
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
    mov eax,4
    mov ebx,6
    mul ebx
    add eax,2
    xor edx,edx
    mov ebx,5
    div ebx

    mov edi,eax

    mov eax,div
    call sprint
    mov eax,edi
    call iprintLF

    mov eax,rem
    call sprint
    mov eax,edx
    call iprintLF

    call quit
1Помощь 2Сохран 3Блок 4Смена 5Копия 6Перетягните 7Поиск 8Удалить 9МенюMS 10Выход
```

Рисунок 2.17: Редактируем файл

Компилируем файл и запускаем программу (Рисунок 2.18).

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

Рисунок 2.18: Рис. 3.17: Смотрим на результат работы программы

Создаем новый файл в каталоге (Рисунок 2.19).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 2.19: Создаем файл командой touch

Открываем файл и редактируем в соответствии с листингом 6.4 (Рисунок 2.20).

```

variant.asm      0 L:[ 2+27 28/ 35] *(440 / 501b) 9810 0x00A [*][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 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 iprintf
    call quit

```

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

Рисунок 2.20: Заполняем файл

Компилируем файл и запускаем его (Рисунок 2.21).

```

mechipurnoyj@fedora:~/work/arch-pc/lab06$ nasm -f elf variant.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032253636
Ваш вариант: 17
mechipurnoyj@fedora:~/work/arch-pc/lab06$ 

```

Рисунок 2.21: Смотрим на результат работы программы

## 2.3 Ответы на вопросы

- 1) Стока «`mov eax,rem`» и строка «`call sprint`»
- 2) Эти инструкции используются для чтения строки с вводом данных от пользователя. Начальный адрес строки сохраняется в регистре `ecx`, а количество символов в строке (максимальное количество символов, которое может быть считано) сохраняется в регистре `edx`. Затем вызывается процедура `sread`, которая выполняет чтение строки.
- 3) Инструкция «`call atoi`» используется для преобразования строки в целое число. Она принимает адрес строки в регистре `eax` и возвращает полученное число в регистре `eax`.

- 4) Стока «xor edx,edx» обнуляет регистр edx перед выполнением деления.  
Стока «mov ebx,20» загружает значение 20 в регистр ebx. Стока «div ebx» выполняет деление регистра eax на значение регистра ebx с сохранением частного в регистре eax и остатка в регистре edx.
- 5) Остаток от деления записывается в регистр edx.
- 6) Инструкция «inc edx» используется для увеличения значения в регистре edx на 1 В данном случае, она увеличивает остаток от деления на 1
- 7) Стока «mov eax,edx» передает значение остатка от деления в регистр eax.  
Стока «call iprintLF» вызывает процедуру iprintLF для вывода значения на экран вместе с переводом строки.

# 3 Задание для самостоятельной работы

Создаем новый файл в каталоге (Рисунок 3.1).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ touch lab6-4.asm  
mechipurnoyj@fedora:~/work/arch-pc/lab06$ mc
```

Рисунок 3.1: Создаем файл

Открываем его и заполняем, чтобы решалось выражение  $f(x) = 18^x \cdot (x + 1) / 6$  (Рисунок 3.2).

The screenshot shows the assembly code for the program. The code defines a global variable `_start` and contains a main loop that reads a value from `stdin`, converts it to an integer using `atoi`, calculates  $18^x \cdot (x + 1) / 6$  using `mul` and `div` instructions, and prints the result using `sprintf` and `stdout`. The assembly code is as follows:

```
[lab6-4.asm] 1.1:{ 1+ 0 -1/ 36}* (1 - / 381b) @105 0x003  
Kinclude "1c.out.ase"  
SECTION data  
msg: DB "Введите x: ", 0  
div: DB "Результат: ", 0  
SECTION bss  
rez: RESB 80  
x: RESB 80  
SECTION text  
GLOBAL _start  
.start:  
    mov eax,msg  
    call sprintf  
    mov eax,x  
    mov ebx,80  
    call read  
    mov eax,x  
    call atoi  
    add eax,1  
    mov eax,18  
    mul ebx  
    mov ebx,6  
    div ebx  
    mov [ebx],eax  
    mov eax,div  
    call sprint  
    mov eax,[ebx]  
    call sprintf  
    call quit
```

Рисунок 3.2: Заполняем файл

Компилируем программу и проверяем для  $x=1$  (Рисунок 3.3).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
1
Результат: 6
mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 3.3: Проверяем работу программы

проверяем для x=3 (Рисунок 3.4).

```
mechipurnoyj@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
3
Результат: 12
mechipurnoyj@fedora:~/work/arch-pc/lab06$
```

Рисунок 3.4: Проверяем работу программы

## **4 Выводы**

Мы приобрели навыки создания исполнительных файлов для решения выражений и освоили арифметические инструкции в NASM.