

# **Отчет по лабораторной работе №6**

**Простейший вариант**

Чигладзе Майя Владиславовна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Порядок выполнения лабораторной работы</b>	<b>6</b>
2.1	Символьные и численные данные в NASM . . . . .	6
2.2	Задание 2 . . . . .	6
2.3	Задание 3 . . . . .	8
2.4	Задание 4 . . . . .	8
2.5	Задание 5 . . . . .	9
2.6	Задание 6 . . . . .	10
2.7	Задание 7 . . . . .	11
<b>3</b>	<b>Ответы на вопросы</b>	<b>12</b>
<b>4</b>	<b>Задание для самостоятельной работы</b>	<b>14</b>
4.1	Задание 1 . . . . .	14
<b>5</b>	<b>Выводы</b>	<b>16</b>
	<b>Список литературы</b>	<b>17</b>

# Список иллюстраций

2.1	Создание каталога . . . . .	6
2.2	Заполняем файл . . . . .	7
2.3	Запускаем файл . . . . .	7
2.4	Переход в созданный каталог . . . . .	8
2.5	Создание файла . . . . .	8
2.6	Создаем файл . . . . .	9
2.7	Создаем исполняемый файл и запускаем . . . . .	9
2.8	Создание файла . . . . .	9
2.9	print . . . . .	9
2.10	Создание файла . . . . .	10
2.11	Создаем исполняемый файл и запускаем . . . . .	10
2.12	Меняем текст программы . . . . .	10
2.13	Запуск файла . . . . .	11
2.14	Создание файла . . . . .	11
2.15	Создаем исполняемый файл и запускаем . . . . .	11
4.1	Код . . . . .	15
4.2	Результат . . . . .	15

## **Список таблиц**

# 1 Цель работы

Освоение навыков работы с арифметическими командами в языке ассемблера  
NASM

## 2 Порядок выполнения лабораторной работы

### 2.1 Символьные и численные данные в NASM

Создадим каталог для программ лабораторной работы No 6, перейдем в него и создадим файл lab6-1.asm (рис. [2.1])



```
mvchigladze@dk3n64 ~ $ mkdir ~/work/arch-pc/lab06
mvchigladze@dk3n64 ~ $ cd ~/work/arch-pc/lab06
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ touch lab6-1.asm
```

Рис. 2.1: Создание каталога

### 2.2 Задание 2

Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения записанные в регистрах. Заполняем файл по листингу (рис. [2.2]), и создаем и запускаем исполняемый файл (рис. [2.3])

```

a e lab6-1.asm [-----] 10
БТА %include 'in_out.asm'
ход
а еа
стр 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: Заполняем файл

```

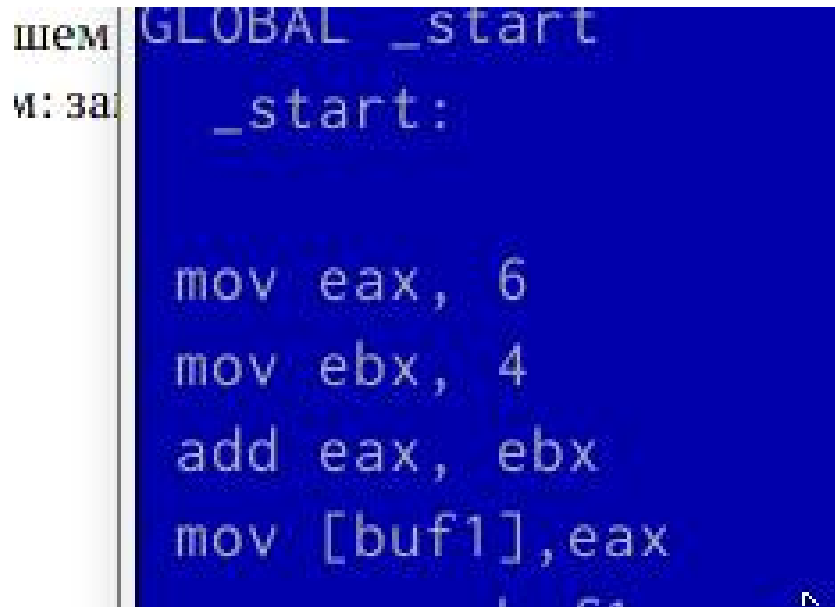
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ./lab6-1
j
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $

```

Рис. 2.3: Запускаем файл

## 2.3 Задание 3

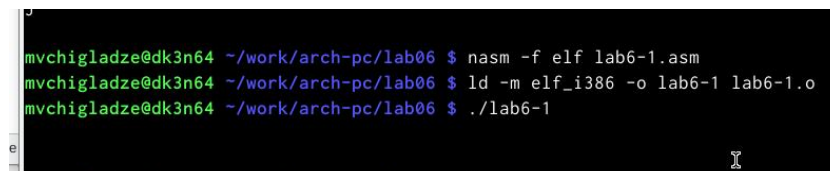
Далее изменим текст программы и вместо символов, запишем в регистры числа, замените строки (рис. [2.4]) и запустим исполняемый файл (рис. [2.5]) Символ не отображается, потому что это отступ строки



```
GLOBAL _start
_start:

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

Рис. 2.4: Переход в созданный каталог



```
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ./lab6-1
```

Рис. 2.5: Создание файла

При замене printLF на print будет всего 1 отступ

## 2.4 Задание 4

Создадим файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 (рис. [2.6]) и введем в него текст программы из листинга 6.2 В результате работы программы мы



получим число 106 (рис. [2.7]). В данном случае, как и в первом, команда `add` складывает коды символов '6' и '4' ( $54+52=106$ ). Однако, в отличие от программы из листинга 6.1, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число

```
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-2.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ls
in_out.asm lab6-1 lab6-1.asm lab6-1.o lab6-2.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $
```

Рис. 2.6: Создаем файл

```
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-2
106
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $
```

Рис. 2.7: Создаем исполняемый файл и запускаем

## 2.5 Задание 5

Далее изменим текст программы запишем в регистры числа, и запустим исполняемый файл (рис. [2.8]). У нас сложилось два числа, а не числа с этим кодом.

```
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-2
10
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $
```

Рис. 2.8: Создание файла

При замене `printLF` на `print` отступа не будет (рис. [2.9]).

```
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-2
10mvchigladze@dk3n40 ~/work/arch-pc/lab06 $
```

Рис. 2.9: `print`

## 2.6 Задание 6

Создадим файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. [2.10]) и введем в него текст программы из листинга 6.3 В результате работы программы мы получим нужные нам фразы 6 (рис. [2.11]).

```
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-3.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $
```

Рис. 2.10: Создание файла

```
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
```

Рис. 2.11: Создаем исполняемый файл и запускаем

Далее изменим текст программы (рис. [2.12]), и запустим исполняемый файл (рис. [2.13]). Арифметические операции выполнены корректно

```
; ----- 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'
```

Рис. 2.12: Меняем текст программы

```

Остаток от деления: 1
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1

```

Рис. 2.13: Запуск файла

## 2.7 Задание 7

Создадим файл `variant.asm` в каталоге `~/work/arch-pc/lab06` (рис. [2.14]) и введем в него текст программы из листинга 6.4 В результате работы программы мы получим нужные ввод с консоли и определение варианта (рис. [2.15]).

```

Остаток от деления: 1
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/variant.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $

```

Рис. 2.14: Создание файла

```

mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1132239399
Ваш вариант: 20
mvchigladze@dk3n40 ~/work/arch-pc/lab06 $

```

Рис. 2.15: Создаем исполняемый файл и запускаем

### 3 Ответы на вопросы

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения 'Ваш вариант:'?

Ответ: `rem: DB 'Ваш вариант:',0 mov eax,rem call sprint`

2. Для чего используются следующие инструкции? `mov ecx, x mov edx, 80 call sread`

Ответ: Запись адреса переменной x в EAX; запись длины вводимого сообщения в EBX; вызов подпрограммы ввода сообщения

3. Для чего используется инструкция "call atoi"?

Ответ: Ввод с клавиатуры осуществляется в символьном виде и для корректной работы арифметических операций в NASM символы необходимо преобразовать в числа. Для этого может быть использована функция atoi.

4. Какие строки листинга 6.4 отвечают за вычисления варианта?

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

5. В какой регистр записывается остаток от деления при выполнении инструкции "div ebx"?

Ответ: Результат будет записан в регистр EAX, а остаток в регистр EDX

6. Для чего используется инструкция "inc edx"?

Ответ: Инструкция INC используется для увеличения операнда на единицу

7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?

Ответ: `mov eax,edx` - вызов подпрограммы печати значения; `call iprintLF` - из 'edx' (остаток) в виде символов

## 4 Задание для самостоятельной работы

### 4.1 Задание 1

Написать программу вычисления выражения  $y=f(x)$ . Программа должна выводить выражение для вычисления, выводить запрос на ввод значения  $x$ , вычислять заданное выражение в зависимости от введенного  $x$ , выводить результат вычислений. Вид функции  $f(x)$  выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы (Мой номер 20). Создайте исполняемый файл и проверьте его работу для значений  $x_1$  и  $x_2$  из 6.3. Моя функция:  $x^3 \cdot 1/3 + 21$  Я обнулила переменную  $ebx$ , записала в нее  $eax$ , умножила 2 раза, для получения 3 степени, обнулила  $edx$ , записала в  $ebx$  - 3, разделила на  $ebx$ , добавила 21 (рис. [4.1] и рис. [4.2]).

```

mov ecx, x
mov edx, 80
call sread

mov eax,x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'

xor ebx,ebx
mov ebx,eax
mul ebx
mul ebx
xor edx,edx
mov ebx,3
div ebx
add eax,21

mov edi, eax

mov eax,rem
call sprint
mov eax,edi
call iprintLF

```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить

Рис. 4.1: Код

```

mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ nasm -f elf lab6-4.asm
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-4 lab6-4.o
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ./lab6-4
Введите x:
1
Результат: 21
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ./lab6-4
Введите x:
2
Результат: 23
mvchigladze@dk3n64 ~/work/arch-pc/lab06 $ ./lab6-4
Введите x:
3
Результат: 30

```

Рис. 4.2: Результат

## **5 Выводы**

В ходе лабораторной работы, я освоила навыки работы с арифметическими командами в языке ассемблера NASM.



## **Список литературы**