

# Отчёт по лабораторной работе №6

## Арифметические операции в NASM.

Данил Евгеньевич Овчиников

### 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

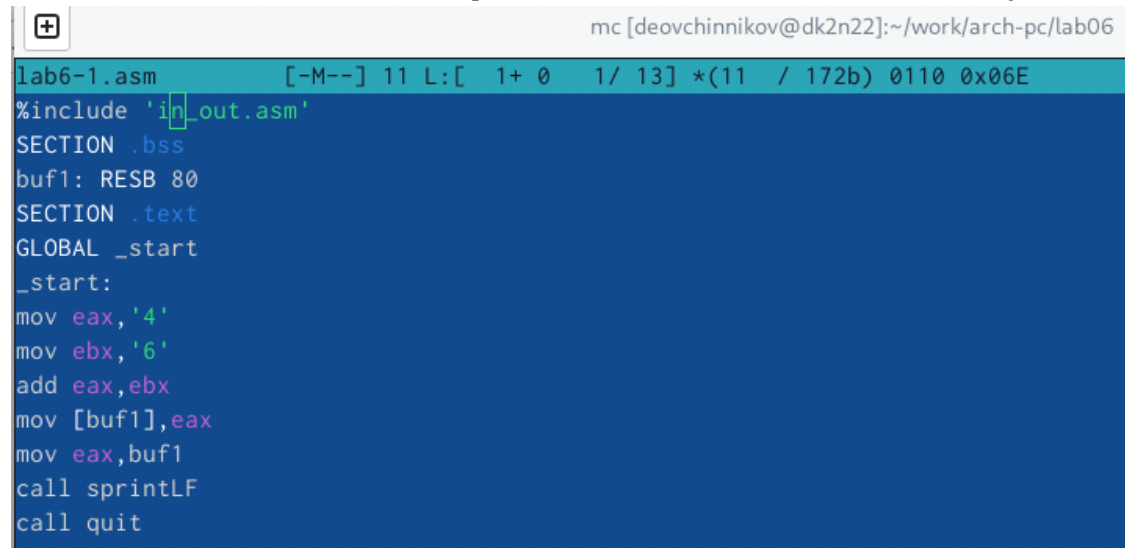
### 2 Выполнение лабораторной работы

- Создадим каталог для программ лабораторной работы № 6, перейдите в него и создайте файл lab7-1.asm:

```
deovchinnikov@dk2n22 ~/work/arch-pc/lab07 $ mkdir ~/work/arch-pc/lab06
deovchinnikov@dk2n22 ~/work/arch-pc/lab07 $ cd ~/work/arch-pc/lab06
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ touch lab6-1.asm
```

Рис. 1: Создание каталога для программ лабораторной работы.

- Создадим исполняемый файл, введем в него листинг 6.1 и запустим его.



```
mc [deovchinnikov@dk2n22]:~/work/arch-pc/lab06
lab6-1.asm [-M--] 11 L:[ 1+ 0 1/ 13] *(11 / 172b) 0110 0x06E
#include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, '4'
mov ebx, '6'
add eax, ebx
mov [buf1], eax
mov eax, buf1
call sprintLF
call quit
```

Рис. 2: Ввод листинга.

```

deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ mc

deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ d -m elf_i386 -o lab6-1 lab6-1.o
bash: d: команда не найдена
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-1
j

```

Рис. 3: Создание и запуск файла.


- Изменим текст программы и вместо символов, запишем в регистры числа. Исправим текст программы (Листинг 1) следующим образом.

```

lab6-1.asm  [-M--]  8 L:[ 1+ 6  7/ 13] *(93 / 168b) 0052 0x034
#include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,4
mov ebx,6
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit

```

Рис. 4: Замена значений кода листинга.

- Создадим исполняемый файл и запустим его. Символ не отображается при выводе на экран и соответствует символу .

```

deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-1

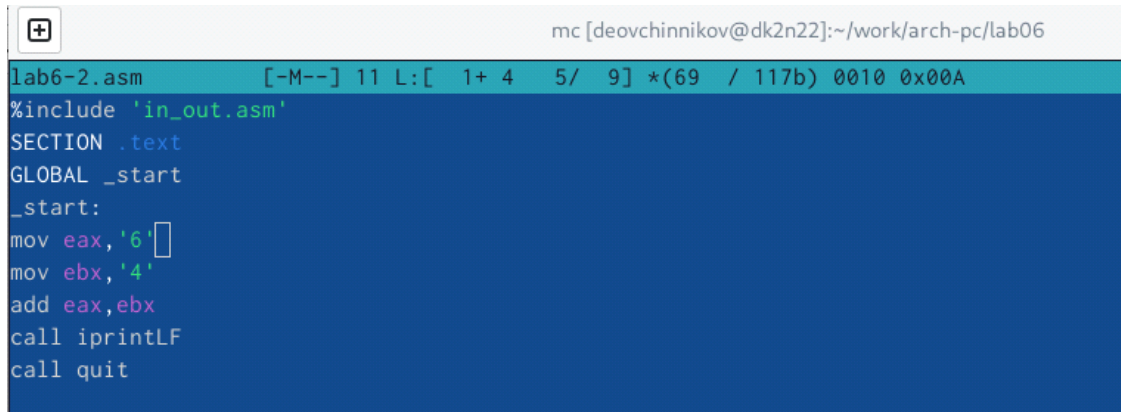
```

Рис. 5: Создание и запуск исполняемого файла.

- Создадим файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и введем в него текст программы из листинга 6.2.

```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-2.asm
```

Рис. 6: Создание файла lab6-2.asm.



```
lab6-2.asm [-M--] 11 L:[ 1+ 4 5/ 9] *(69 / 117b) 0010 0x00A
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF
call quit
```

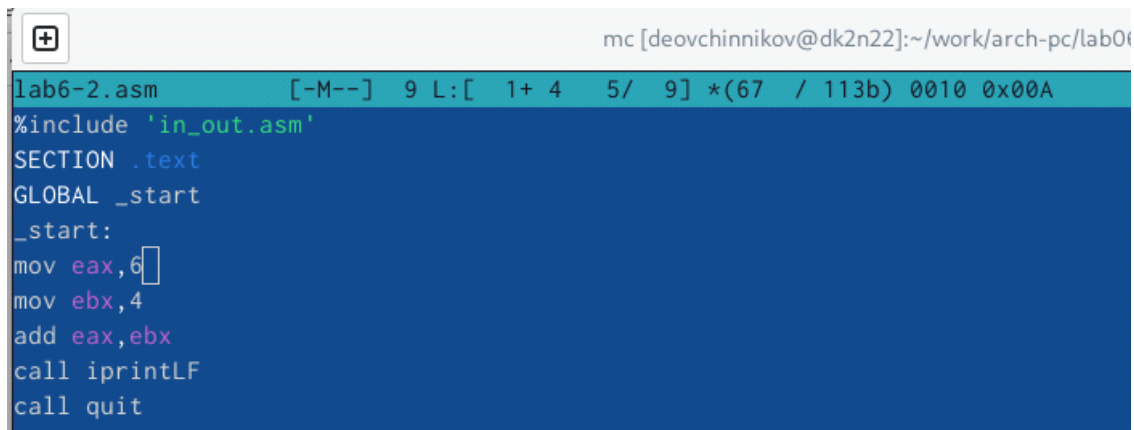
Рис. 7: Ввод текста программы из листинга 6.2.

- Создадим и запустил файл lab6-2.asm.

```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-2
106
```

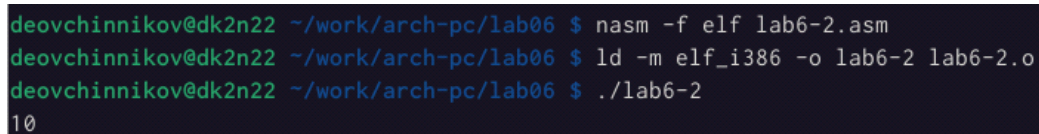
Рис. 8: Создание и запуск файла lab6-2.asm.

- Аналогично предыдущему примеру изменим символы на числа.



```
lab6-2.asm      [-M--]  9 L:[  1+ 4   5/  9] *(67 / 113b) 0010 0x00A
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

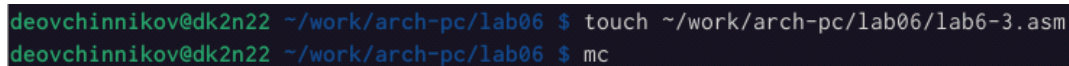
Рис. 9: Замена чисел.



```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-2
10
```

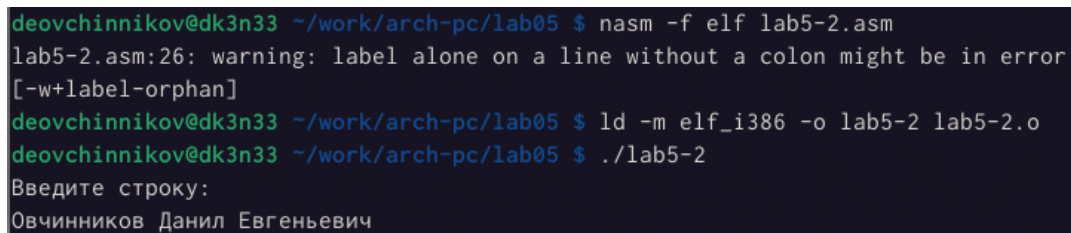
Рис. 10: Создадим исполняемый файл и запустим его.

- Создадим файл lab6-3.asm в каталоге ~/work/arch-pc/lab06:



```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-3.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ mc
```

Рис. 11: Создание файла lab6-3.asm.



```
deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
lab5-2.asm:26: warning: label alone on a line without a colon might be in error
[-w+label-orphan]
deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
Овчинников Данил Евгеньевич
```

Рис. 12: Запуск файла.

- Внимательно изучим текст программы из листинга 6.3 и введем в lab6-3.asm.

```
mc [deovchinnikov@dk2n22]:~/work/arch-pc/lab06
lab6-3.asm      [-M--] 21 L:[ 1+ 1  2/ 29] *(73 /1365b) 1103 0x44F
;-----
; 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
```

Рис. 13: Ввод программы из листинга 6.3.

- Создадим исполняемый файл и запустим его.

```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
```

Рис. 14: Создание копии и внос изменений

- Получим исполняемый файл и проверим его работу

```

deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1copy.asm
lab5-1copy.asm:26: warning: label alone on a line without a colon might be in error [-w+lab
han]
deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1copy lab5-1copy.o
deovchinnikov@dk3n33 ~/work/arch-pc/lab05 $ ./lab5-1copy
Введите строку:
Овчинников

```

Рис. 16: Проверка работы файла

- Изменим текст программы для вычисления выражения  $f(x) = (4 * 6 + 2)/5$ . Создадим и запустим его.

```

mc [deovchinnikov@dk2n22]:~/work/arch-pc/lab06
lab6-3.asm  [-M--]  9 L: [ 1+12 13/ 29] *(410 /1365b) 0032 0x020
;-----
; 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/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

```

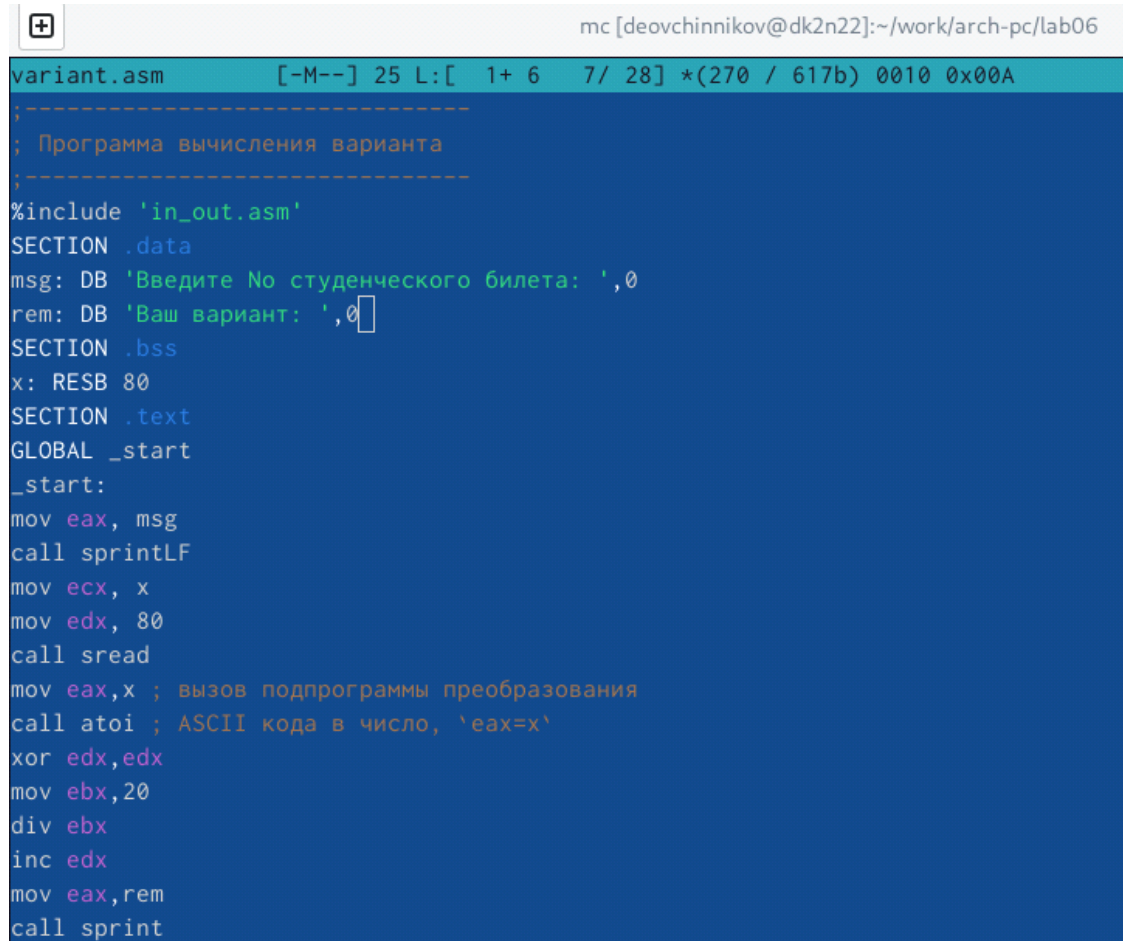
Рис. 17: Изменение текста программы для вычисления.

- Создадим файл variant.asm в каталоге ~/work/arch-pc/lab06:

```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/variant.asm
```

Рис. 18: Создание файла.

- Внимательно изучим текст программы из листинга 6.4 и введем в файл variant.asm.



```
variant.asm      [-M--] 25 L:[ 1+ 6  7/ 28] *(270 / 617b) 0010 0x00A
;-----
; 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 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
```

- Создадим исполняемый файл и запустим его. Проверим результат работы



программы вычислив номер варианта аналитически.

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения 'Ваш вариант?': `mov eax` и `rem call sprint`.

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

`mov ecx, x` - запись входной переменной в регистр `ecx`;

`mov edx, 80` - запись размера переменной в регистр `edx`; `call`

`sread` - вызов процедуры чтения данных;

3. Для чего используется инструкция "`call atoi`"?: Вызов `atoi` – функция преобразующая `ascii`-код символа в целое число и записывающий результат в регистр `eax`.

4. Какие строки листинга 7.4 отвечают за вычисления варианта?: `xor edx, edx` `mov ebx, 20` `div ebx` `inc edx`.

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

6. Для чего используется инструкция "`inc edx`"?: `INC` используется для увеличения операнда на единицу.

7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?: `mov eax, rem` `call sprint` `mov eax, edx` `call iprintLF`.

```
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
deovchinnikov@dk2n22 ~/work/arch-pc/lab06 $ ./variant
Введите Но студенческого билета:
3
Ваш вариант: 4
```

*Рис. 19: Создание файла и его запуск.*

### 3 Выводы

Я освоил арифметические инструкции языка ассемблера NASM.