

Лабораторная работа №6

Архитектура компьютера и операционные системы

Бабенко Константин, НКАбд-01-23

Содержание

1	Цель работы	1
2	Выполнение лабораторной работы.....	1
2.1	Домашняя работа.....	Ошибка! Закладка не определена.
3	Выводы	7

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

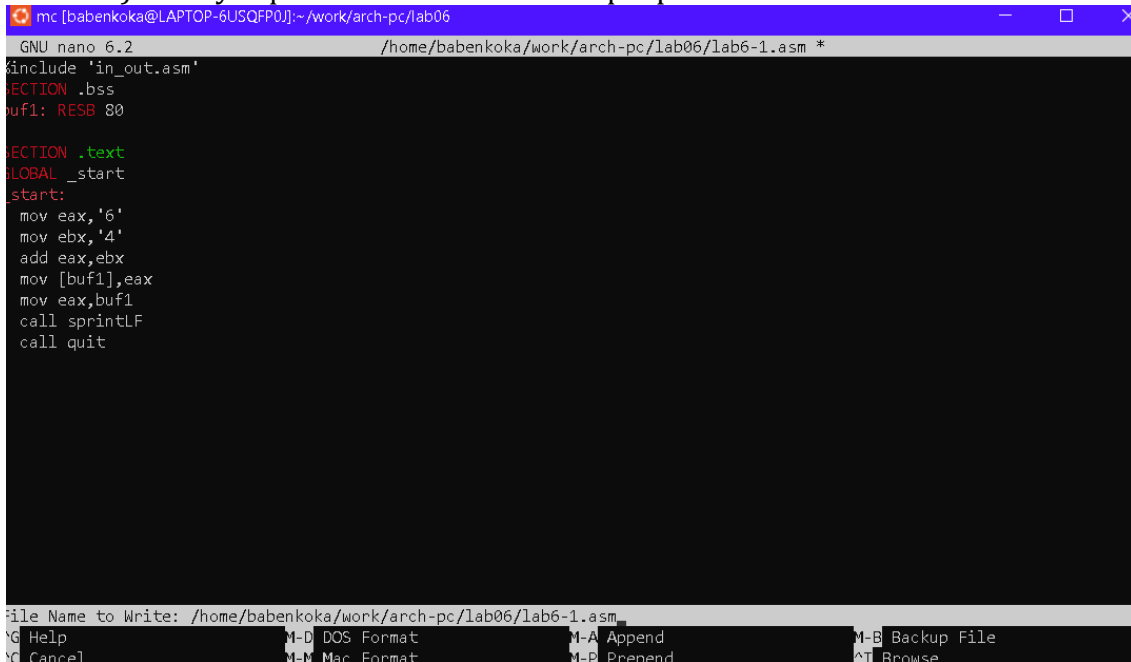
Символьные и численные данные в NASM

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

```
babenkoka@LAPTOP-6USQFP0J:~$ mkdir ~/work/arch-pc/lab06
babenkoka@LAPTOP-6USQFP0J:~$ cd ~/work/arch-pc/lab06
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ touch lab6-1.asm
```

Figure 1: Создаю файл.

2) Ввожу в файл lab6-1.asm текст программы из листинга 6.1:

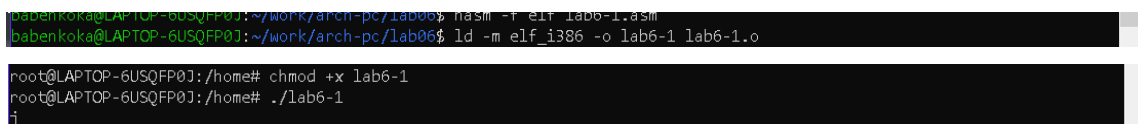


```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/lab6-1.asm *
#include 'in_out.asm'
SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:
    mov eax, '6'
    mov ebx, '4'
    add eax, ebx
    mov [buf1], eax
    mov eax, buf1
    call sprintf
    call quit
```

Figure 2: Ввожу текст программы.

Создаю исполняемый файл и запускаю его:




```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o

root@LAPTOP-6USQFP0J:/home# chmod +x lab6-1
root@LAPTOP-6USQFP0J:/home# ./lab6-1
j
```

Figure 3: Работа файла.

3) Далее изменяю текст программы и вместо символов, записываю в регистры числа. Исправляю текст программы следующим образом: заменить строки `mov eax, '6'` `mov ebx, '4'` на строки `mov eax, 6` `mov ebx, 4`:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/lab6-1.asm
#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
    call quit
```

Figure 4: Измененный текст.

Создаю исполняемый файл и запустите его:

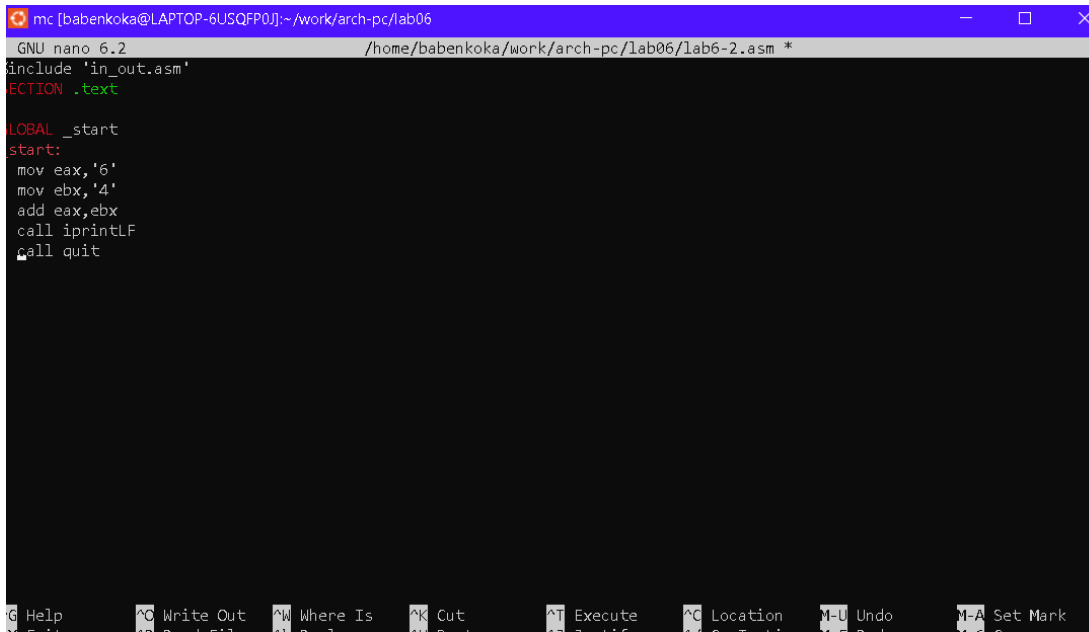
```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o

root@LAPTOP-6USQFP0J:/home# chmod +x lab6-1
root@LAPTOP-6USQFP0J:/home# ./lab6-1
```

Figure 5: Работа измененного файла.

Код 10 соответствует символу BS, он не отображается.

4) Создаю файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и ввожу в него текст программы из листинга 6.2:



```
mc [babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/lab6-2.asm *
include 'in_out.asm'
SECTION .text

GLOBAL _start
start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

Figure 6: Введенный текст.

Создаю исполняемый файл и запускаю его:

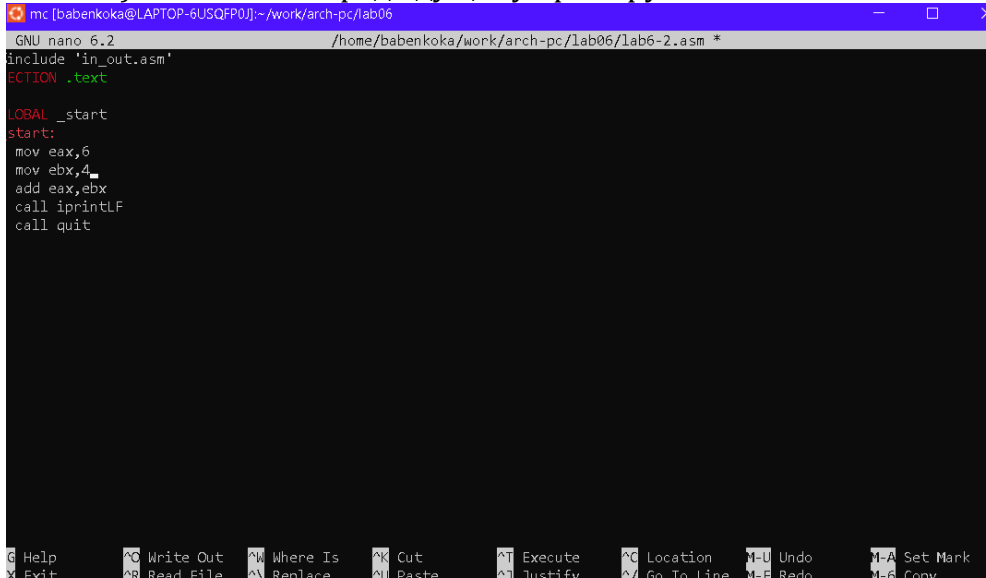
```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o

root@LAPTOP-6USQFP0J:/home# chmod +x lab6-2
root@LAPTOP-6USQFP0J:/home# ./lab6-2
106
```

Figure 7: Работа второго файла.

В результате работы программы получено число 106. В данном случае, как и в первом, команда add складывает коды символов '6' и '4' ($54+52=106$). Однако, в отличие от программы lab6-1, функция iprintLF позволяет вывести число, а не символ, кодом которого является это число.

5) Аналогично предыдущему примеру изменяю символы на числа:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/lab6-2.asm *
include "in_out.asm"
SECTION .text

GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Figure 8: Изменения.

Создаю исполняемый файл и запускаю его:

```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o

root@LAPTOP-6USQFP0J:/home# ./lab6-2
0
```

Figure 9: Работа измененной программы.

Заменяю функцию iprintLF на iprint. Создаю исполняемый файл и запускаю его:

```
root@LAPTOP-6USQFP0J:/home# ./lab6-2
10root@LAPTOP-6USQFP0J:/home#
```

Figure 10: Работа программы без LF.

Функция iprintLF в отличие от iprint переводит курсор на другую строку.

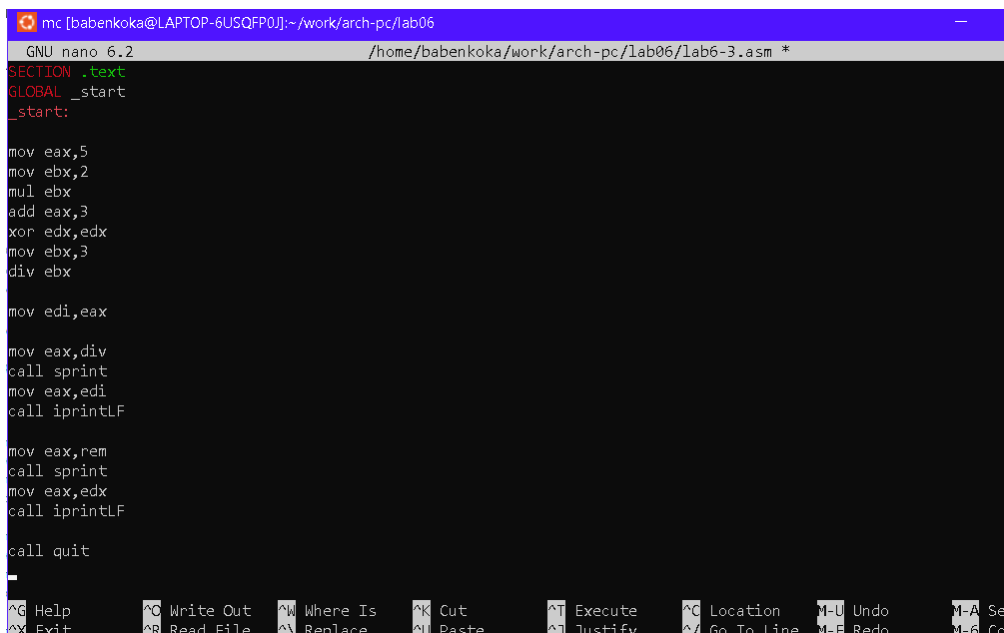
Выполнение арифметических операций в NASM

6) Создаю файл lab6-3.asm в каталоге ~/work/arch-pc/lab06:

```
enkokka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
```

Figure 11: Программа.

Ввожу текст программы из листинга 6.3:



```
mc [babenkoka@LAPTOP-6USQFP0J]:~/work/arch-pc/lab06
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/lab6-3.asm *
SECTION .text
GLOBAL _start
_start:

mov eax,5
mov ebx,2
mul ebx
add eax,3
xor edx,edx
mov ebx,3
div ebx

mov edi,eax

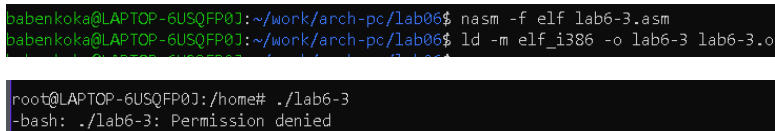
mov eax,div
call sprint
mov eax,edi
call iprintLF

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

call quit
```

Figure 12: Введенный текст.

Создаю исполняемый файл и запускаю его:

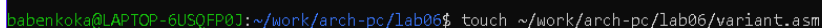


```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o

root@LAPTOP-6USQFP0J:/home# ./lab6-3
-bash: ./lab6-3: Permission denied
```

Figure 13: Работа программы.

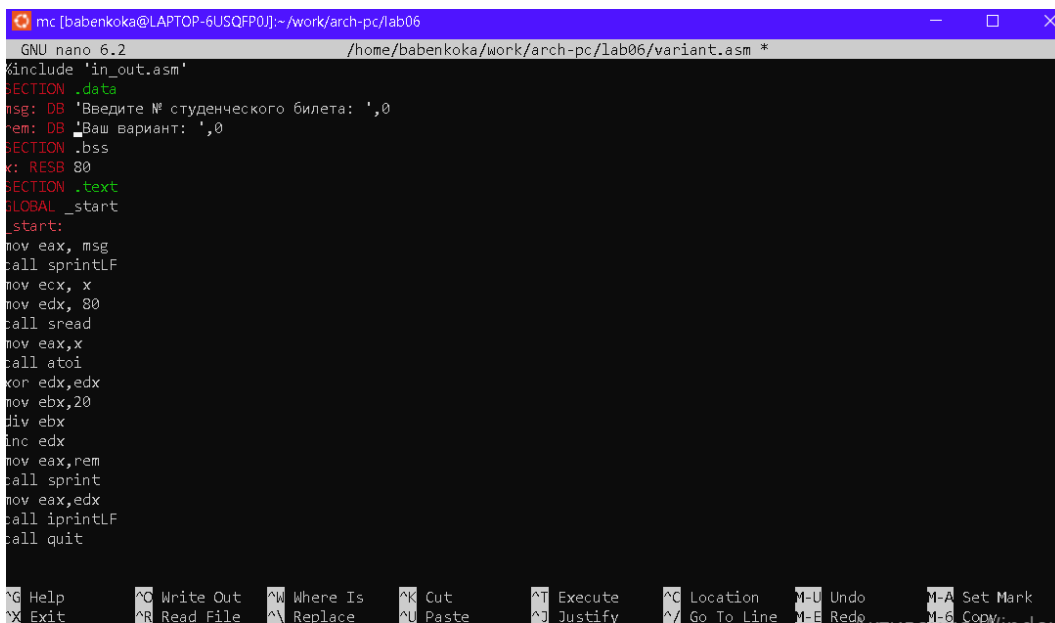
7) Создаю файл variant.asm в каталоге ~/work/arch-pc/lab06:



```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
```

Figure 14: Создаю новый файл.

Ввожу в файл variant.asm текст программы из листинга 6.4:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/variant.asm *
#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
call atoi
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintfLF
call quit
```

Figure 15: Текст программы.

1. Строки, отвечающие за вывод на экран сообщения «Ваш вариант:»:

```
mov eax, rem
call sprintf
mov eax, edx
call iprintfLF
```

2. Следующие инструкции отвечают за считывание данных:

```
mov ecx, x
mov edx, 80
call sread
```

3. Инструкция “call atoi” используется для превращения ASCII кода в число.

4. За вычисления варианта отвечают:

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

5. В edx.

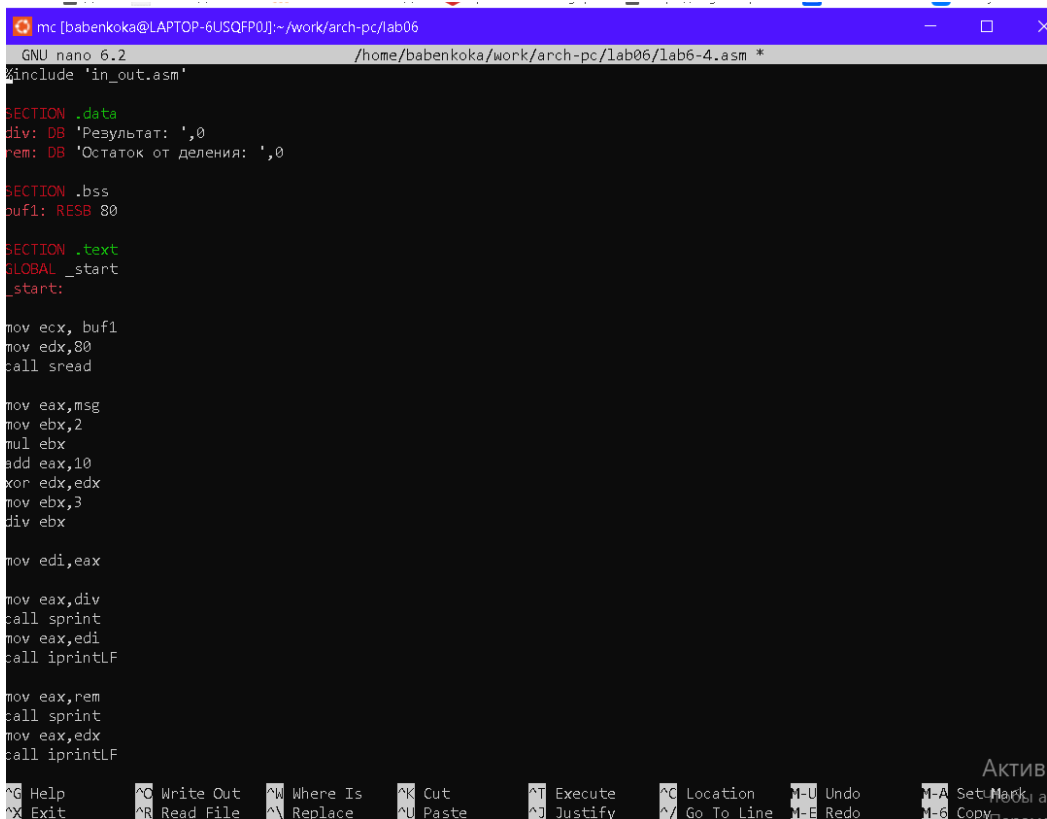
6. Инструкция “inc edx” используется для прибавления единицы.

7. За вывод на экран результата вычислений отвечают:

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

2.1 Домашняя работа

Ввожу программу вычисления выражения $y = (10 + 2x)/3$:



```
mc [babenkoka@LAPTOP-6USQFP0J]:~/work/arch-pc/lab06
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab06/lab6-4.asm *
#include 'in_out.asm'

SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

mov ecx, buf1
mov edx, 80
call sread

mov eax, msg
mov ebx, 2
mul ebx
add eax, 10
xor edx, edx
mov ebx, 3
div ebx

mov edi, eax

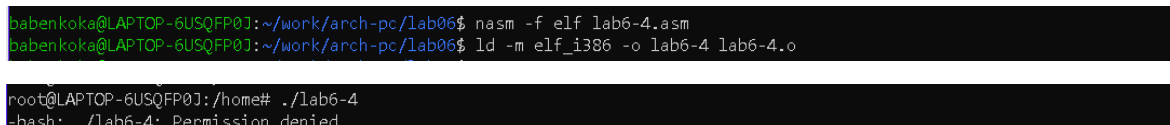
mov eax, div
call sprint
mov eax, edi
call iprintLF

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

Актив
```

Figure 16: Изменяю первый файл.

Создаю исполняемый файл и проверяю его работу для значений 1 и 10:



```
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o

root@LAPTOP-6USQFP0J:/home# ./lab6-4
-bash: ./lab6-4: Permission denied
```

Figure 17: Работа файла для 1.

3 Выводы

Я освоил арифметические инструкции языка ассемблера NASM.