

Отчёт по лабораторной работе № 6

Дисциплина: Архитектура компьютера

Мутале Чали

Содержание

1	Цель работы	1
2	Задание	1
3	Выполнение лабораторной работы	1
4	Выполнение самостоятельной работы	9
5	Выводы	12
	Список литературы	12

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Ответы на вопросы по программе

3 Выполнение лабораторной работы

1. Символьные и численные данные в NASM

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

```
(cmutale@cmutale)-[~/work/arch-pc]
$ mkdir -p ~/work/arch-pc/lab06

(cmutale@cmutale)-[~/work/arch-pc]
$ cd ~/work/arch-pc/lab06
```

Рис 1

Копирую в текущий каталог файл in_out.asm, потому что он будет использоваться в других программах:

```

(cmutale@cmutale)~[/work/arch-pc/lab06]
$ cp ~/Downloads/in_out.asm in_out.asm

(cmutale@cmutale)~[/work/arch-pc/lab06]
$ ls
in_out.asm  lab6-1.asm

```

Рис 2

Открываю созданный файл lab6-1.asm, вставляю в него программу вывода значения регистра eax:

```

mc [cmutale@cmutale]:~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/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 sprintLF
call quit

```

Рис 3

Создаю исполняемый файл программы и запускаю его. Программа выводит символ 'j', потому что это сумма кодов символов 6 и 4 по системе ASCII:

```

(cmutale@cmutale)~[/work/arch-pc/lab06]
$ mc

(cmutale@cmutale)~[/work/arch-pc/lab06]
$ nasm -f elf lab6-1.asm

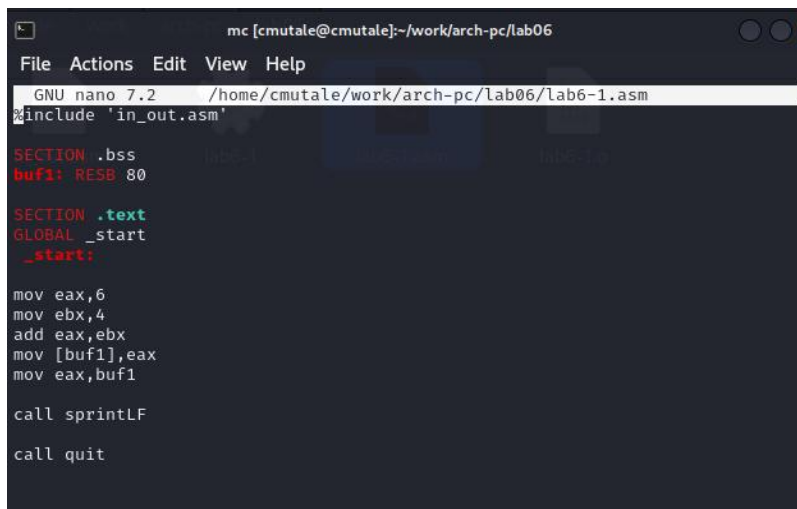
(cmutale@cmutale)~[/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-1 lab6-1.o

(cmutale@cmutale)~[/work/arch-pc/lab06]
$ ./lab6-1
j

```

Рис 4

Изменяю текст программы и вместо символов, пишу в регистры числа:



```
mc [cmutale@cmutale]:~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/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
```

Рис 5

Создаю исполняемый файл программы и запускаю его. Теперь она выводит новую строку (код символа 10):



```
(cmutale@cmutale)-[~]
$ cd ~/work/arch-pc/lab06

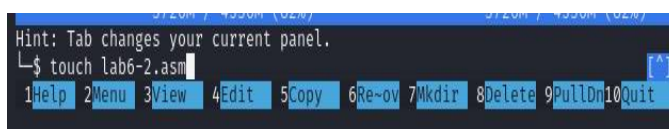
(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-1.asm

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-1 lab6-1.o

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ./lab6-1
```

Рис 6

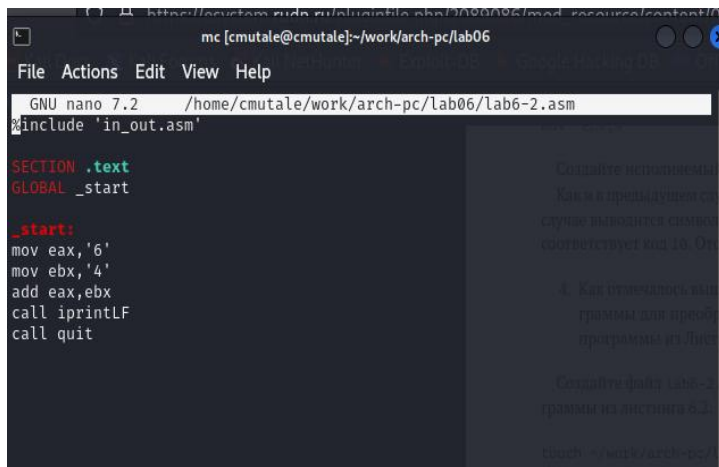
Создаю новый файл lab6-2.asm:



```
Hint: Tab changes your current panel.
└─$ touch lab6-2.asm
1Help 2Menu 3View 4Edit 5Copy 6Re~ov 7Mkdir 8Delete 9PullDn10Quit
```

Рис 7

Открываю созданный файл lab6-2.asm, вставляю в него программу вывода значения регистра eax:



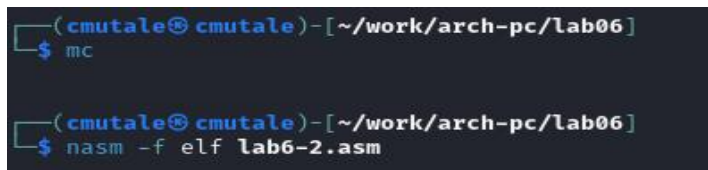
```
mc [cmutale@cmutale]~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/work/arch-pc/lab06/lab6-2.asm
#include 'in_out.asm'

SECTION .text
GLOBAL _start

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

Рис 8

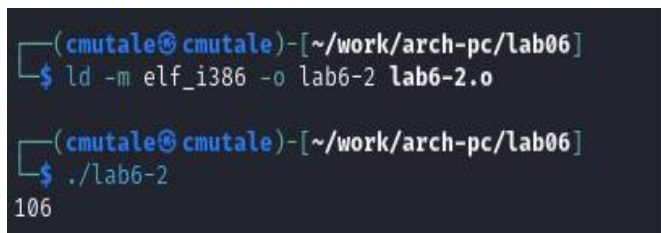
Создаю исполняемый файл программы и запускаю его. Она выводит сумму кодов символ 6(54) и символ 4(52):



```
(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ mc

(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ nasm -f elf lab6-2.asm
```

Рис 9

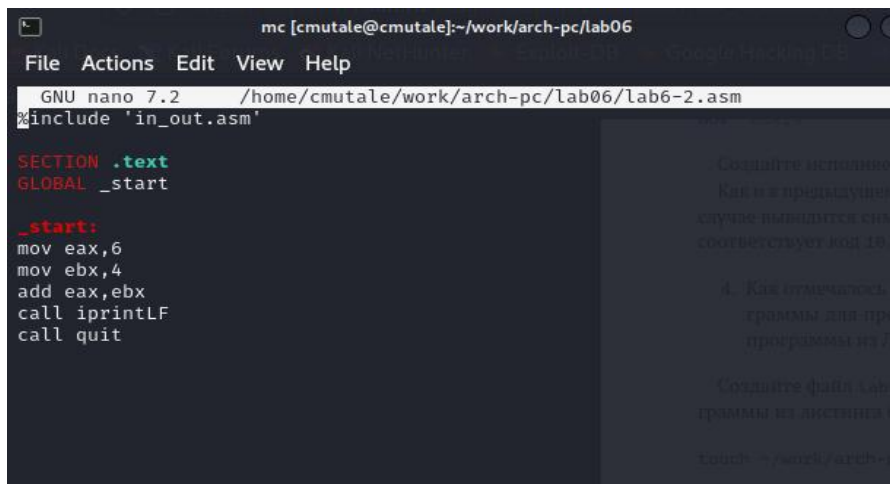


```
(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-2 lab6-2.o

(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ ./lab6-2
106
```

Рис 10

Изменяю текст программы и вместо символов, пишу в регистры числа:



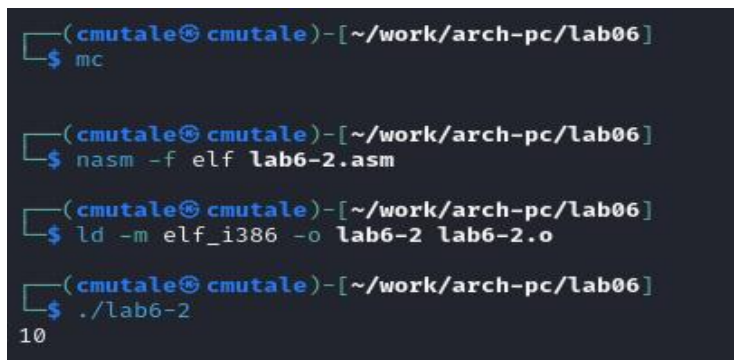
```
mc [cmutale@cmutale]:~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/work/arch-pc/lab06/lab6-2.asm
#include 'in_out.asm'

SECTION .text
GLOBAL _start

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

Рис 11

Создаю исполняемый файл программы и запускаю его. Теперь она выводит сумму чисел 6 и 4:



```
(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ mc

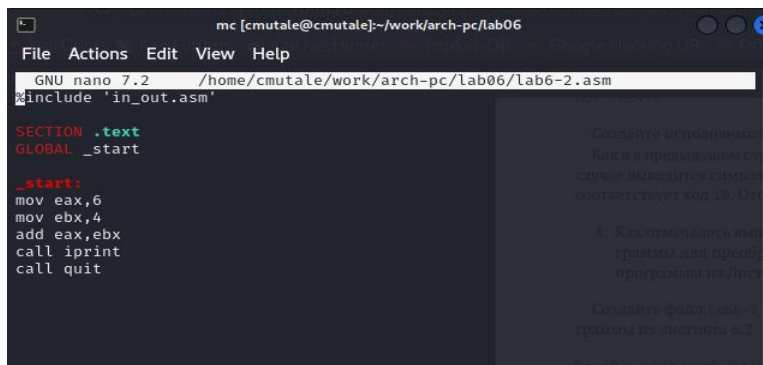
(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ nasm -f elf lab6-2.asm

(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-2 lab6-2.o

(cmutale@cmutale) - [~/work/arch-pc/lab06]
$ ./lab6-2
10
```

Рис 12

Изменяю iprintLF в iprint:



```
mc [cmutale@cmutale]:~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/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 iprint
call quit
```

Рис 13

Создаю исполняемый файл и запускаю его:

```
(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ mc

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-2.asm

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-2 lab6-2.o

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ./lab6-2
10

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$
```

Рис 14

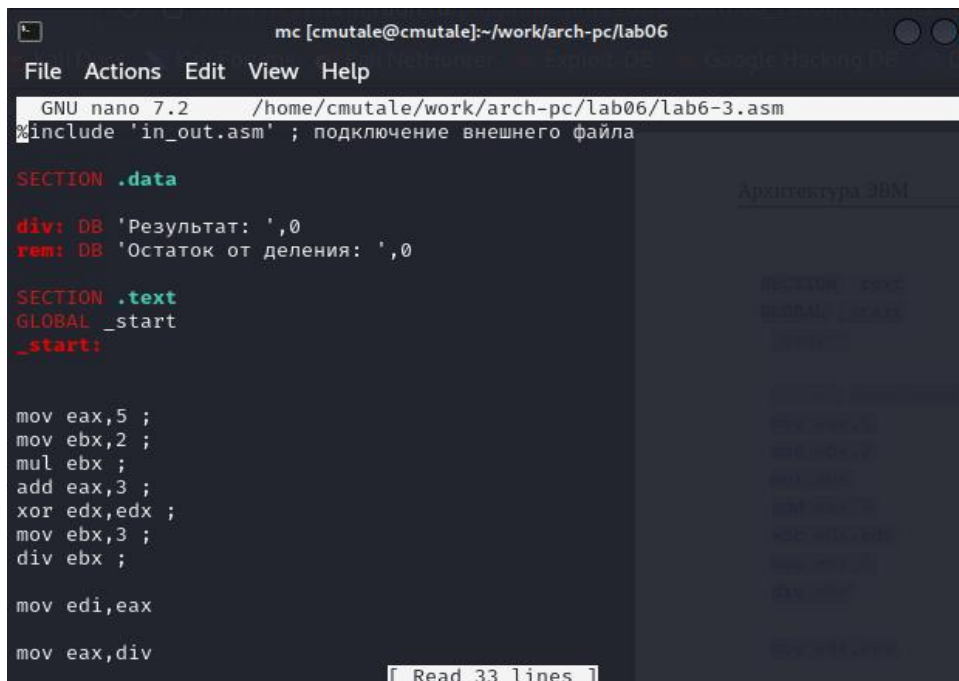
2. Выполнение арифметических операций в NASM

Создаю новый файл lab6-3.asm:

```
(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ touch lab6-3.asm
```

Рис 15

Вставляю в него программу для вычисления выражения $f(x) = (5 * 2 + 3)/3$:



```
mc [cmutale@cmutale]:~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/work/arch-pc/lab06/lab6-3.asm
%include 'in_out.asm' ; подключение внешнего файла

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

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

[ Read 33 lines ]
```

Рис 16

Создаю исполняемый файл и запускаю его:

```

[cmutale@cmutale] - [~/work/arch-pc/lab06]
$ mc

[cmutale@cmutale] - [~/work/arch-pc/lab06]
$ nasm -f elf lab6-3.asm

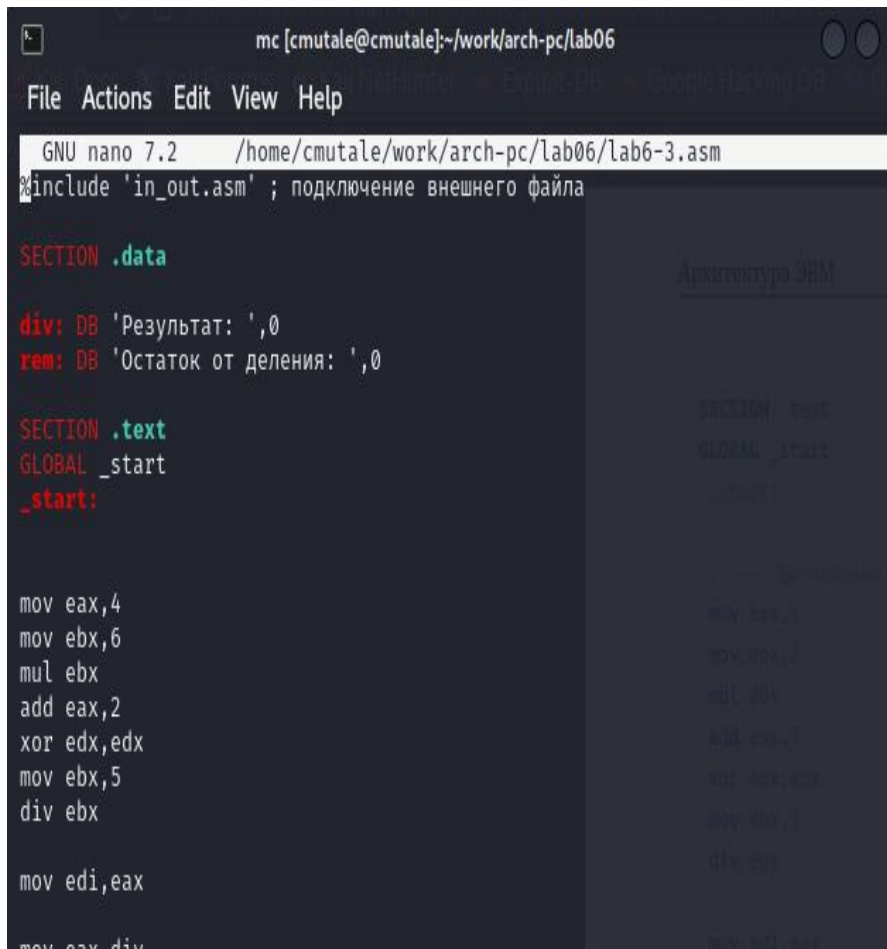
[cmutale@cmutale] - [~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-3 lab6-3.o

[cmutale@cmutale] - [~/work/arch-pc/lab06]
$ ./lab6-3
Результат: 4
Остаток от деления: 1

```

Рис 17

Изменяю текст программы для вычисления выражения $f(x) = (4 * 6 + 2) / 5$:



```

mc [cmutale@cmutale]:~/work/arch-pc/lab06
File Actions Edit View Help
GNU nano 7.2 /home/cmutale/work/arch-pc/lab06/lab6-3.asm
%include 'in_out.asm' ; подключение внешнего файла

SECTION .data
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

```

Рис 18

Создаю исполняемый файл и запускаю его:

```

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ mc

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-3.asm

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ld
ld: no input files

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-3 lab6-3.o

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ ./lab6-3
Результат: 5
Остаток от деления: 1

```

Рис 19

Создаю файл variant.asm:

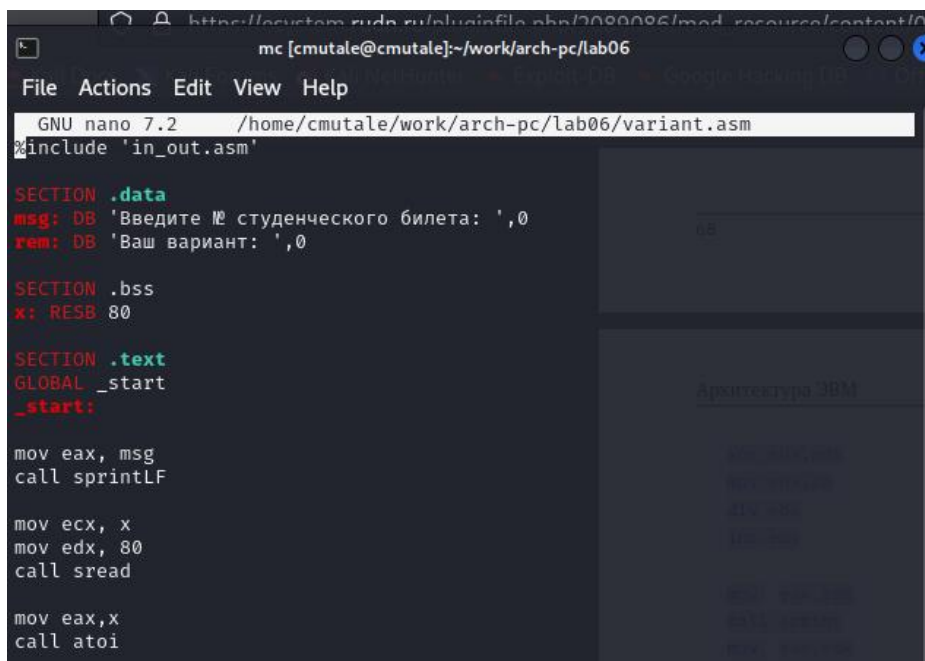
```

(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ touch variant.asm

```

Рис 20

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета:



```

GNU nano 7.2 /home/cmutale/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 sprintLF

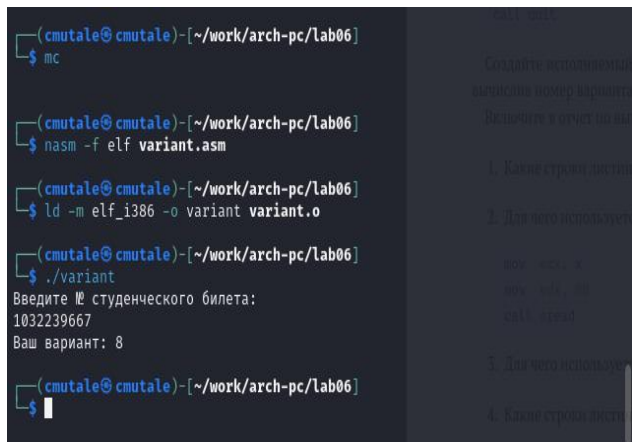
mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi

```

Рис 21

Создаю исполняемый файл и запускаю его. Ввожу номер своего студенческого билета, программа выводит, вариант 10:



```
(cmutale@cmutale) [~/work/arch-pc/lab06]
$ mc

(cmutale@cmutale) [~/work/arch-pc/lab06]
$ nasm -f elf variant.asm

(cmutale@cmutale) [~/work/arch-pc/lab06]
$ ld -m elf_i386 -o variant variant.o

(cmutale@cmutale) [~/work/arch-pc/lab06]
$ ./variant
Введите № студенческого билета:
1032239667
Ваш вариант: 8

(cmutale@cmutale) [~/work/arch-pc/lab06]
$
```

Рис 22

3. Ответы на вопросы по программе

1. За вывод сообщения “Ваш вариант” отвечают строки кода:

```
mov eax,rem
call sprint
```

2. `mov ecx, x` используется, чтобы положить адрес вводимой строки `x` в регистр. `ecx mov edx,80`-запись в регистр `edx` длины вводимой строки `call sread` - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры

3. `call atoi` используется для вызова подпрограммы из внешнего файла, которая преобразует `ascii`-код символа в целое число и записывает результат в регистр `eax`

4. За вычисления варианта отвечают строки:

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

5. Остаток от деления записывается в регистр `edx`.
6. Инструкция `inc edx` используется для увеличения значения регистра `edx` на 1.
7. За вывод на экран результатов вычислений отвечают строки:

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

4 Выполнение самостоятельной работы

Создаю файл `task.asm`:

```
(cmutale@cmutale)-[~/work/arch-pc/lab06]
$ touch task.asm
```

Рис 23

В него пишу программу для вычисления выражения $f(x)=5(x + 18) - 28$. Она берет входное значение "x", прибавляет его к 18, умножает результат сложения на 5 а потом вычитает 28:

```
GNU nano 7.2
#include 'in_out.asm'

SECTION .data
msg: DB 'Введите значение x: ', 0
rem: DB 'Результат: ', 0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprint

mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi

add eax, 18
mov ebx, 5
mul ebx
add eax, -28
mov edi, eax

mov eax, rem
call sprint

mov eax, edi
call iprint

call quit
```

Рис 24

Создаю исполняемый файл и запускаю его. При $x = 2$, она выводит 72. При $x = 3$, она выводит 77:

```
(cmutale@cmutale)-[~/work/arch-pc/lab06]  
$ nasm -f elf task.asm
```

Рис 25

```
(cmutale@cmutale)-[~/work/arch-pc/lab06]  
$ ld -m elf_i386 -o task task.o  
  
(cmutale@cmutale)-[~/work/arch-pc/lab06]  
$ ./task  
Введите значение x: 1  
Результат: 18  
  
(cmutale@cmutale)-[~/work/arch-pc/lab06]  
$ ./task  
Введите значение x: 9  
Результат: 34
```

Рис 26

Код Программы:

```
%include 'in_out.asm'  
  
SECTION .data  
msg: DB 'Введите значение переменной x: ',0  
rem: DB 'Результат: ',0  
  
SECTION .bss  
x: RESB 80  
SECTION .text  
GLOBAL _start  
_start:  
mov eax, msg  
call sprint  
  
mov ecx, x  
mov edx, 80  
call sread  
  
mov eax, x  
call atoi  
  
add eax, 11
```

```
mov ebx,2  
mul ebx  
add eax,-6  
mov edi,eax
```

```
mov eax,rem  
call sprint
```

```
mov eax,edi  
call iprint  
call quit
```

5 Выводы

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

Список литературы

Архитектура ЭВМ :::