

ОТЧЕТ по лабораторной работе № 6

дисциплина: Архитектура компьютера

Студент: Лисенков Е.Р.

Содержание

1	Цель работы	1
2	Задания	1
3	Теоретическое введение	1
4	Выполнение лабораторной работы	2
4.1	1 Символьные и численные данные в NASM	2
4.2	Выполнение арифметических операций в NASMц	3
4.3	Задание для самостоятельной работы	5
5	Выполнение заданий	6
6	Выводы	7
7	Список литературы	7

1 Цель работы

Цель данной лабораторной работы - освоение арифметических инструкций языка ассемблера NASM.

2 Задания

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: `mov ax,bx`. - Непосредственная адресация – значение операнда

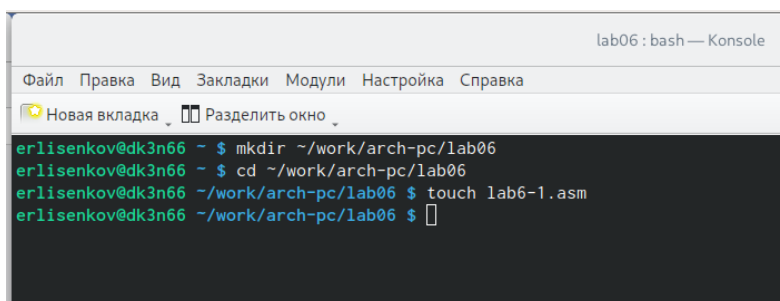
задается непосредственно в команде, Например: `mov ax,2`. - Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные данные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно.

4 Выполнение лабораторной работы

4.1 1 Символьные и численные данные в NASM

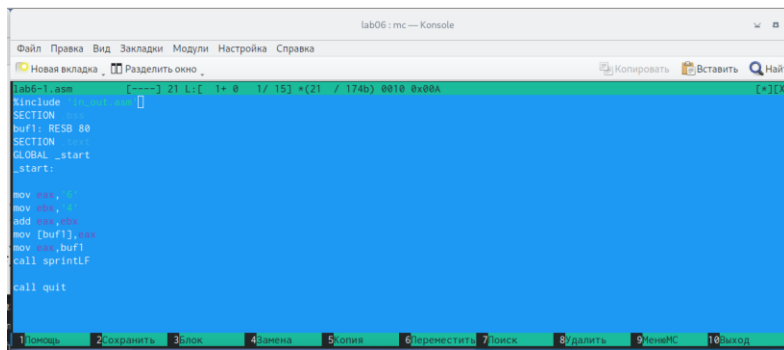
Перейдём в репозиторий и создадим там файл с помощью команды `lab6-1.asm`. (рис. [??]).



```
lab06 : bash — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
+ Новая вкладка  + Разделить окно
erlisenkov@dk3n66 ~ $ mkdir ~/work/arch-pc/lab06
erlisenkov@dk3n66 ~ $ cd ~/work/arch-pc/lab06
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ touch lab6-1.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $
```

Переход в репозиторий и создание

Выполню перенос файла `in_out.asm` в наш репозиторий и начну редактирование файла `lab6-1.asm`. (рис. [??]).



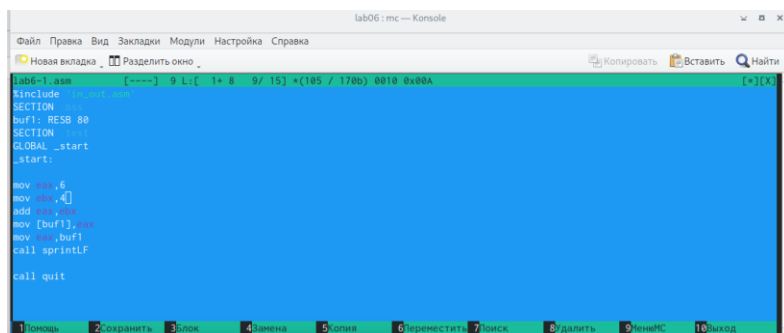
Редактирование файла

Запустим файл. Увидим символ j, потому что программа вывела то, что соответствует по системе ASCII сумме двоичных кодов символов 4 и 6. (рис. [??]).

```
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ./lab6-1
j
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $
```

Запуск файла

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4. (рис. [??]).



Редактирование файла

Выполню запуск программы и замечу, что вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран. [??]).

```
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ./lab6-1

erlisenkov@dk3n66 ~/work/arch-pc/lab06 $
```

Запуск файла

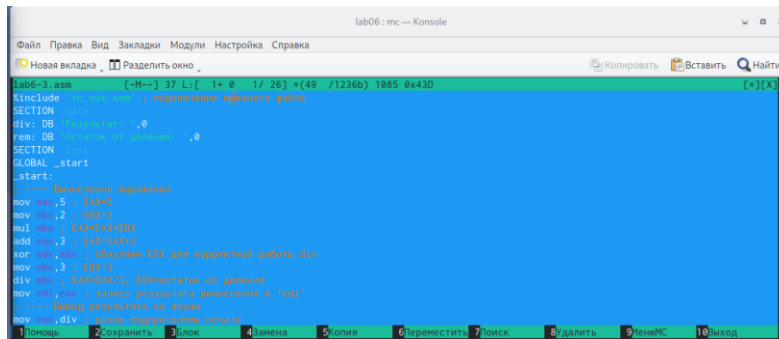
4.2 Выполнение арифметических операций в NASMц

Создам файл lab6-3.asm для дальнейших заданий. (рис. [??]).

```
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ touch lab6-3.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $
```

Создам файл

Добавлю новый код для вычисления математической формулы (рис. [??]).



```
lab6-3.asm [Меню] 37 L: [ 1+ 0 1/ 26] *(49 /12366) 1885 0x43D
#include "asm.h", подключение внешнего файла
SECTION .text
div: DB "Вычисление: ",0
mov: DB "Остаток от деления: ",0
SECTION .start
GLOBAL _start
_start:
---- Вычисление выражения
mov eax,5 ; 10000
mov ebx,2 ; 10000
mul ebx ; 100000000
add eax,3 ; 100000003
xor ebx,ebx ; обнушаем EBX для корректной работы div
mov ebx,3 ; 10000
div ebx ; 100000000, 100000000 от деления
mov ebx,ebx ; выведем результат, посчитав в 'eax'
---- Вывод результата на экран
mov ebx,div ; вывод подстроки пометки
1Помощь 2Сохранить 3Блок 4Имена 5Копия 6Перенести 7Поиск 8Удалить 9Выход 10Выход
```

Редактирование файла

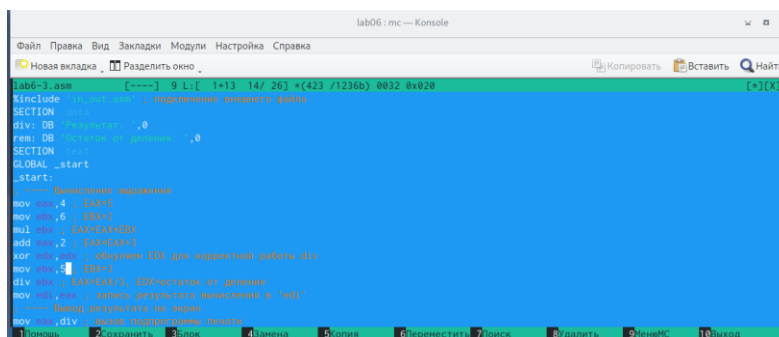
Запустим файл и поймём, что всё работает верно (рис. [??]).

```
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ touch lab6-3.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ mc

erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $
```

Запуск файла

Отредактирую файл и заново запущу программу, чтобы удостовериться, что всё работает верно (рис. [??]) (рис. [??]).



```
lab6-3.asm [Меню] 9 L: [ 1+13 14/ 26] *(423 /12366) 0032 0x820
#include "asm.h", подключение внешнего файла
SECTION .text
div: DB "Вычисление: ",0
mov: DB "Остаток от деления: ",0
SECTION .start
GLOBAL _start
_start:
---- Вычисление выражения
mov eax,4 ; 10000
mov ebx,6 ; 10000
mul ebx ; 100000000
add eax,2 ; 100000002
xor ebx,ebx ; обнушаем EBX для корректной работы div
mov ebx,6 ; 10000
div ebx ; 100000000, 100000000 от деления
mov ebx,ebx ; выведем результат, посчитав в 'eax'
---- Вывод результата на экран
mov ebx,div ; вывод подстроки пометки
1Помощь 2Сохранить 3Блок 4Имена 5Копия 6Перенести 7Поиск 8Удалить 9Выход 10Выход
```

Редактирование файла

```
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1
erlisenkov@dk3n66 ~/work/arch-pc/lab06 $
```

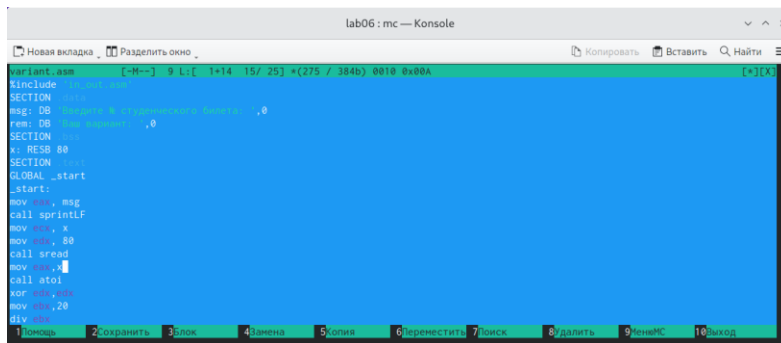
Запуск файла

Создам файл variant (рис. [??]).

```
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ touch variant.asm
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $
```

Создаю файл

Выполню редактирование файла (рис. [??]).



Редактирование файла

Запускаю программу (рис. [??]).

```
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1132232881
Ваш вариант: 2
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $
```

Запуск файла

4.3 Задание для самостоятельной работы

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

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

2. Инструкция `mov ecx, x` используется, чтобы положить адрес вводимой строки `x` в регистр `ecx` `mov edx, 80` - запись в регистр `edx` длины вводимой строки `call read` - 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 ; обнуление edx для корректной работы div
mov ebx,20 ; ebx = 20
div ebx ; eax = eax/20, edx - остаток от деления
inc edx ; edx = edx + 1
```

5. При выполнении инструкции `div ebx` остаток от деления записывается в регистр `edx`
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1
7. За вывод на экран результатов вычислений отвечают строки:

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

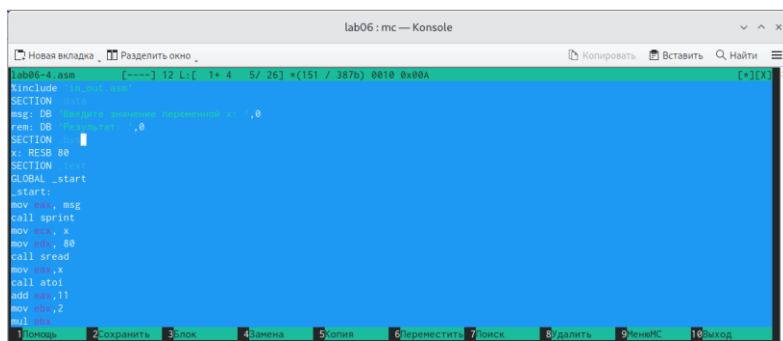
5 Выполнение заданий

Создам программу для выполнения задания (рис. [??]).

```
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ touch lab06-4.asm
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $
```

Создам файл

Изменяю файл для корректной работы и вычисления моего задания (рис. [??]).



Редактирование файла

Выполню запуск программы (рис. [??]).

```
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf lab06-4.asm
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab06-4.asm lab06-4.o
erlisenkov@dk2n26 ~/work/arch-pc/lab06 $ ./lab06-4.asm
Введите значение переменной x: 4
Результат: 24erlisenkov@dk2n26 ~/work/arch-pc/lab06 $
```

Запуск программы

6 Выводы

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

7 Список литературы

Лабораторная работа №6. Арифметические операции в NASM.