Отчет по лабораторной работе №7

Арифметические операции в NASM

Акопян Сатеник

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM

# 2 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обра- ботке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. Далее рассмотрены все существующие способы задания адреса хранения операндов – способы адресации. Существует три основных способа адресации: -Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: mov ax,bx. -Непосредственная адресация – значение операнда задается непосред- ственно в команде, Например: mov ax,2. -Адресация памяти – операнд задает адрес в памяти. В команде указывает- ся символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.# Выполнение лабораторной работы

# 3 Выполнение работы

1. Создаем каталог для программ лабораторной работы № 7, переходим в него и создаем файл lab7-1.asm (рис. 1)



Рис. 1: рисунок 1

1. Вводим в файл lab7-1.asm текст программы из листинга, представленного в лабораторной работе (рис. 2)

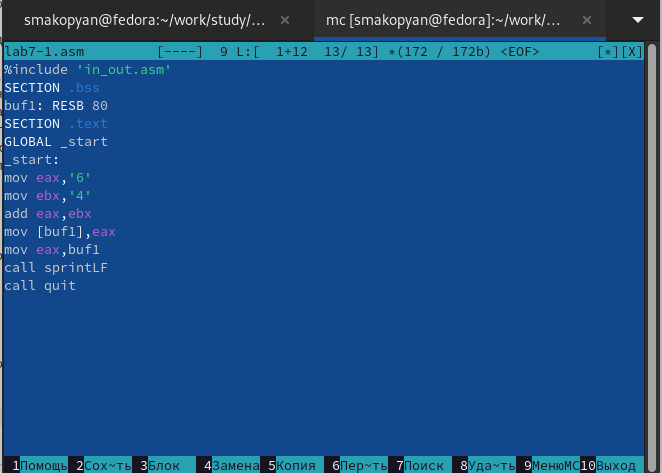


Рис. 2: рисунок 2

1. Создаем исполняемый файл и запускаем его, предварительно скопировав подключаемый файл in\_out.asm в каталог с текстом программы (рис. 3)

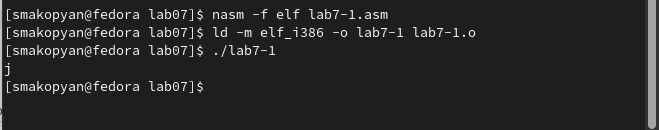


Рис. 3: рисунок 3

1. Изменим текст программы и вместо символов запишем в регистры числа (рис. 4)

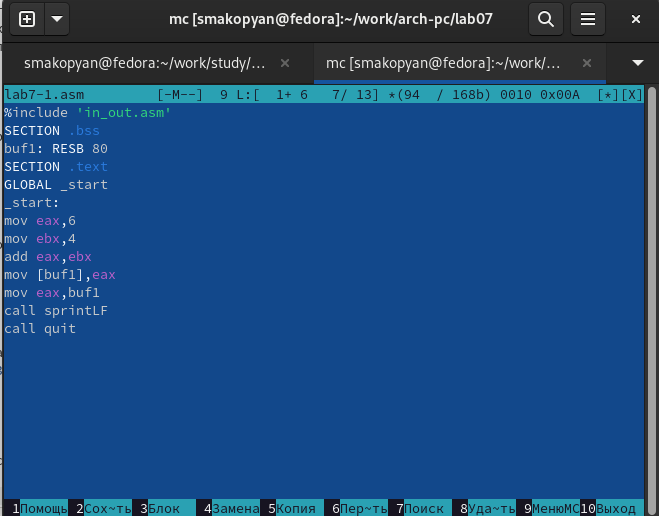


Рис. 4: рисунок 4

1. Создадим исполняемый файл и запустим его (рис. 5).

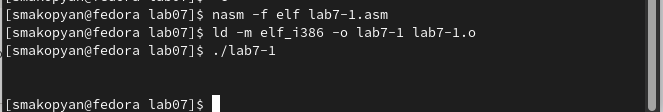


Рис. 5: рисунок 5

Пользуясь таблицей ASCII, определяем что код 10 соответствует символу перевода на новую строку, что является причиной того, почему данный символ вывел на экран только 2 пустые строки.

1. Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 (рис. 6)

Рис. 6: рисунок 6

Рис. 6: рисунок 6

1. Вводим в созданный файл текст программы из листинга 7.2, представленного в лабораторной работе (рис. 7)

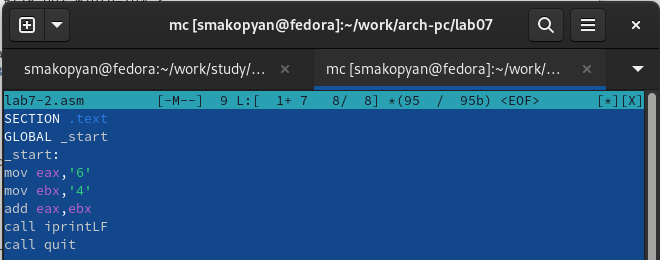


Рис. 7: рисунок 7

8.Создаем исполняемый файл и запускаем его (рис. 8)

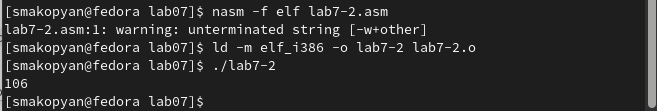


Рис. 8: рисунок 8

1. Аналогично предыдущему примеру изменим символы на числа. (рис. 9)

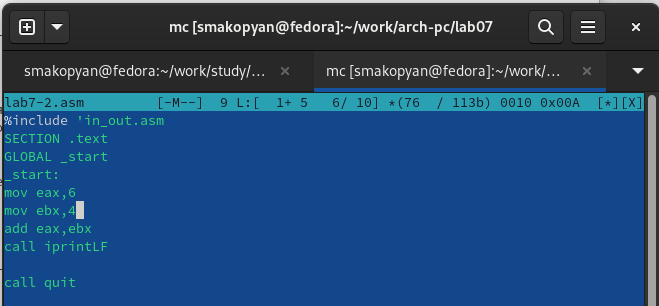


Рис. 9: рисунок 9

1. Создаем исполняемый файл и запускаем его (рис. 10)

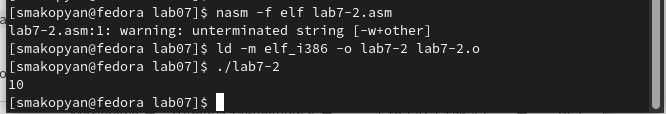


Рис. 10: рисунок 10

1. Меняем функцию iprintLF на iprint. Создаем исполняемый файл и запускаем его. (рис. 11)

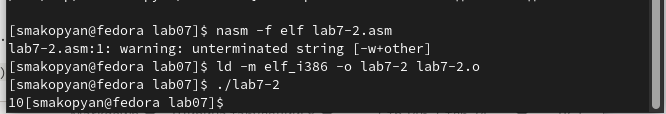


Рис. 11: рисунок 11

Нетрудно заметить, что после проведенных действий изменилось только то, что следующая строка после вывода программы, начинается с текущей.

1. Создаем файл lab7-3.asm в каталоге ~/work/arch-pc/lab07 и вводим в него текст программы из листинга, данного в лабораторной работе (рис. 12)

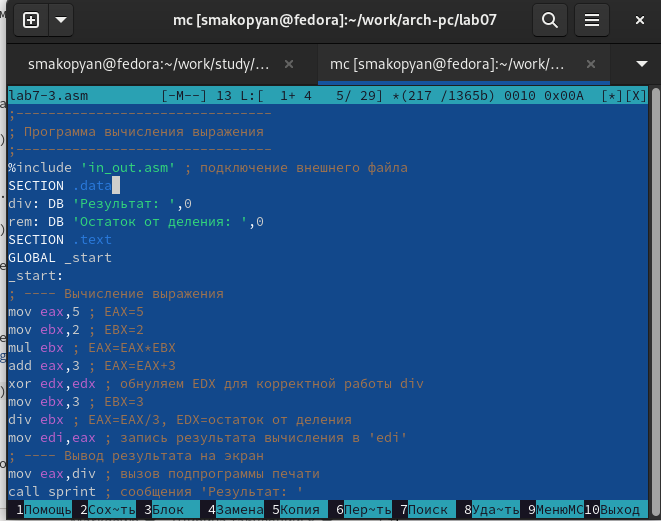


Рис. 12: рисунок 12

1. Создаем исполняемый файл и запускаем его (рис. 13)

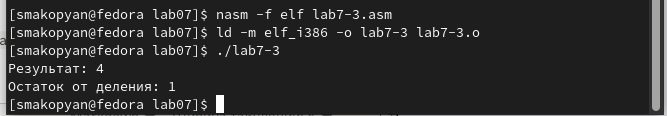


Рис. 13: рисунок 13

1. Изменим текст программы для вычисления выражения (рис. 14), создадим исполняемый файл и проверим его работу (рис. 15)

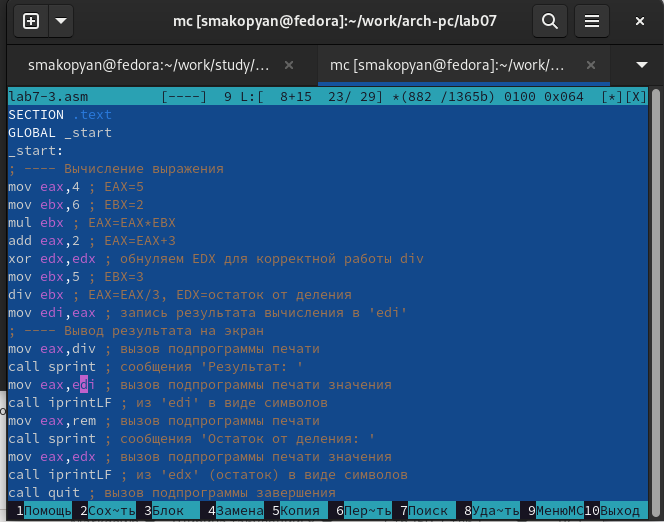


Рис. 14: рисунок 14

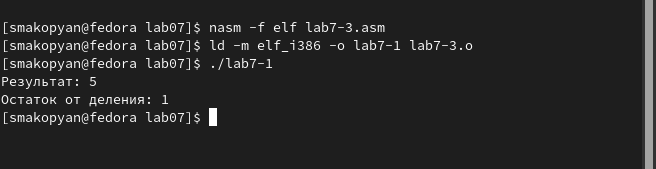


Рис. 15: рисунок 15

1. Создаем файл variant.asm в каталоге ~/work/arch-pc/lab07 (рис. 16), и вводим в него текст, представленный в лабораторной работе (рис. 17)

Рис. 16: рисунок 16

Рис. 16: рисунок 16

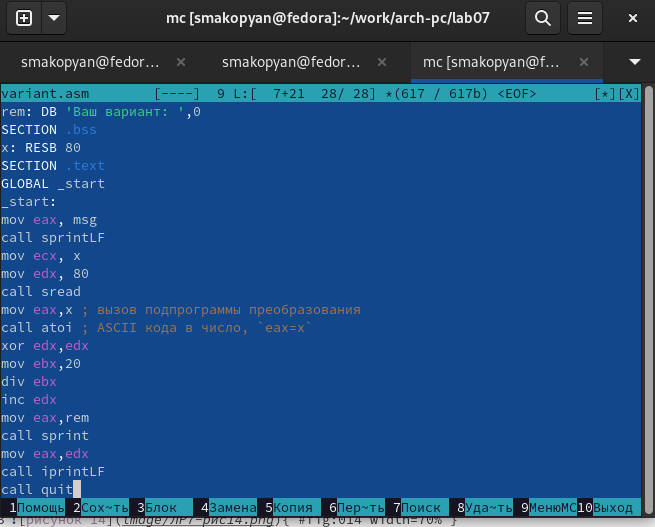


Рис. 17: рисунок 17

1. Создаем исполняемый файл и запускаем его (рис. 18)

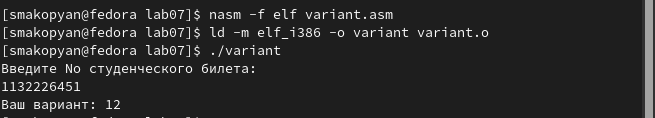


Рис. 18: рисунок 18

# 4 Задание для самостоятельной работы

Вариант 12.

1. Создаем файл, в котором запишем текст программы (рис. 19)

Рис. 19: рисунок 19

Рис. 19: рисунок 19

1. Вводим текст программы в созданный файл (рис. 20)

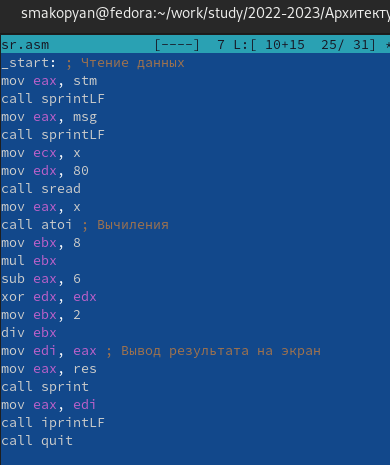


Рис. 20: рисунок 20

1. Создаемый исполняемый файл и проверяем его работу для значений 1 и 5 (рис. 21)

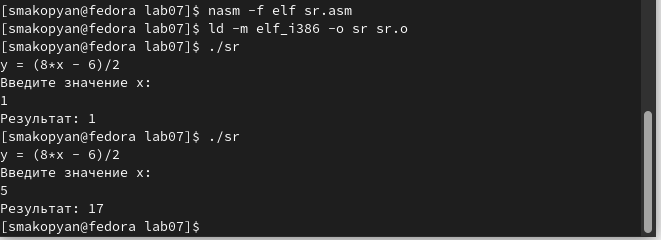


Рис. 21: рисунок 21

# 5 Выводы

В результате данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.

# Список литературы