Отчёт по лабораторной работе №7

Арифметические операции в NASM

Аскеров Александр Эдуардович

Содержание

[1 Цель работы 1](#__RefHeading___Toc373_1717631219)

[2 Выполнение лабораторной работы 1](#__RefHeading___Toc375_1717631219)

[2.1 Символьные и численные данные в NASM 1](#__RefHeading___Toc377_1717631219)

[2.2 Выполнение арифметических операций в NASM 6](#__RefHeading___Toc379_1717631219)

[2.3 Задание для самостоятельной работы 12](#__RefHeading___Toc381_1717631219)

[3 Выводы 14](#__RefHeading___Toc383_1717631219)

# 1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM.

# 2 Выполнение лабораторной работы

## 2.1 Символьные и численные данные в NASM

1. Создадим каталог для программ лабораторной работы № 7, перейдём в него и создадим файл lab7-1.asm.

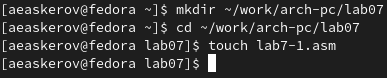


Рис. 1: Создание файла lab7-1.asm в каталоге lab07

1. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистр eax.

Введём в файл lab7-1.asm текст программы из листинга 7.1. В данной программе в регистр eax записывается символ 6 (mov eax,‘6’), в регистр ebx символ 4 (mov ebx,‘4’). Далее к значению в регистре eax прибавляем значение регистра ebx (add eax,ebx, результат сложения запишется в регистр eax). Далее выводим результат. Так как для работы функции sprintLF в регистр eax должен быть записан адрес, необходимо использовать дополнительную переменную. Для этого запишем значение регистра eax в переменную buf1 (mov [buf1],eax), а затем запишем адрес переменной buf1 в регистр eax (mov eax,buf1) и вызовем функцию sprintLF.

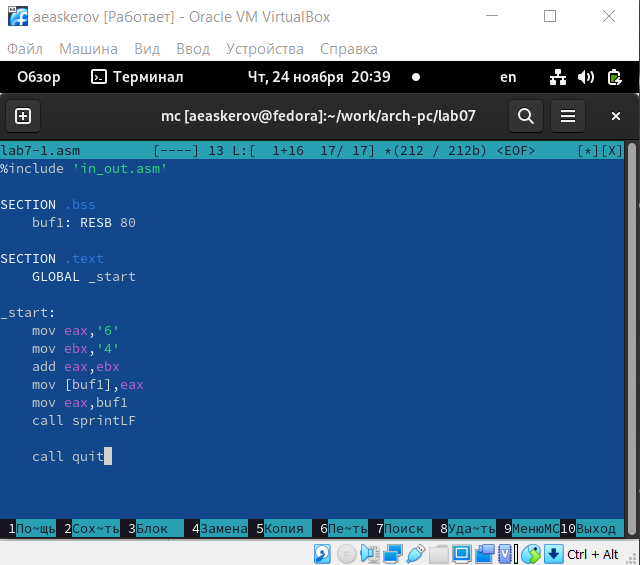


Рис. 2: Программа из листинга 7.1

Создадим исполняемый файл и запустим его.

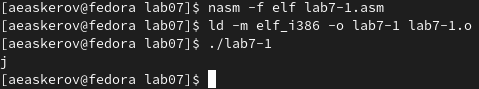


Рис. 3: Результат работы программы lab7-1

В данном случае при выводе значения регистра eax мы ожидаем увидеть число 10. Однако результатом будет символ j. Это происходит потому, что код символа 6 равен 00110110 в двоичном представлении (или 54 в десятичном представлении), а код символа 4 – 00110100 (52). Команда add eax,ebx запишет в регистр eax сумму кодов – 01101010 (106), что в свою очередь является кодом символа j (см. таблицу ASCII в приложении).

1. Далее изменим текст программы и вместо символов, запишем в регистры числа. Исправим текст программы (Листинг 1) следующим образом: заменим строки mov eax,‘6’ mov ebx,‘4’ на строки mov eax,6 mov ebx,4.

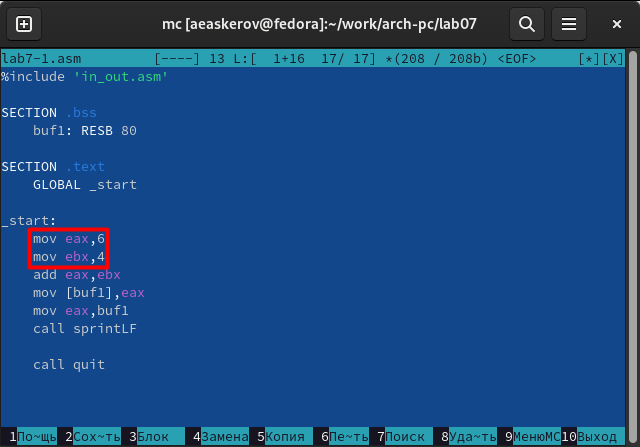


Рис. 4: Изменённый текст программы lab7-1

Создадим исполняемый файл и запустим его.

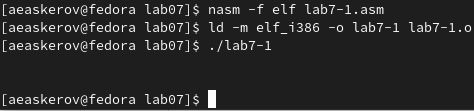


Рис. 5: Результат работы программы lab7-1

Как и в предыдущем случае при исполнении программы мы не получим число 10. В данном случае выводится символ с кодом 10. Из таблицы ASCII видно, что это символ переноса строки. При выводе на экран он не отображается.

1. Как отмечалось выше, для работы с числами в файле in\_out.asm реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы из Листинга 7.1 с использованием этих функций.

Создадим файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 и введём в него текст программы из листинга 7.2.

Рис. 6: Создание файла lab7-2.asm в каталоге lab07

Рис. 6: Создание файла lab7-2.asm в каталоге lab07

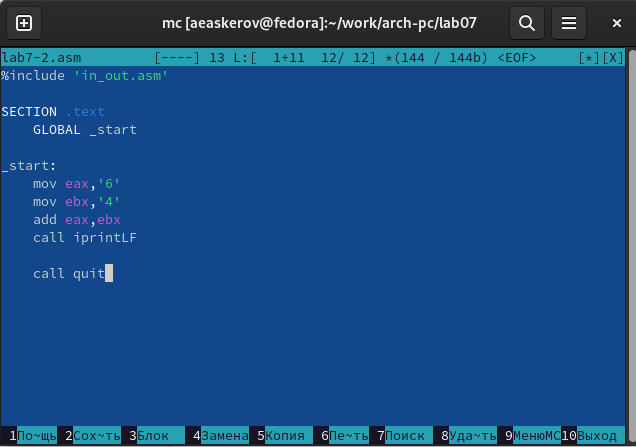


Рис. 7: Программа из листинга 7.2

Создадим исполняемый файл и запустим его.

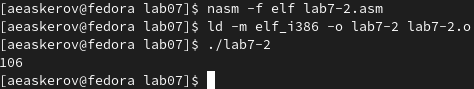


Рис. 8: Результат работы программы lab7-2

В результате работы программы мы получаем число 106. В данном случае, как и в первом, команда add складывает коды символов ‘6’ и ‘4’ (54+52=106). Однако, в отличии от программы из листинга 7.1, функция iprintLF позволяет вывести число, а не символ, кодом которого является это число.

1. Аналогично предыдущему примеру изменим символы на числа. Заменим строки mov eax,‘6’ mov ebx,‘4’ на строки mov eax,6 mov ebx,4.

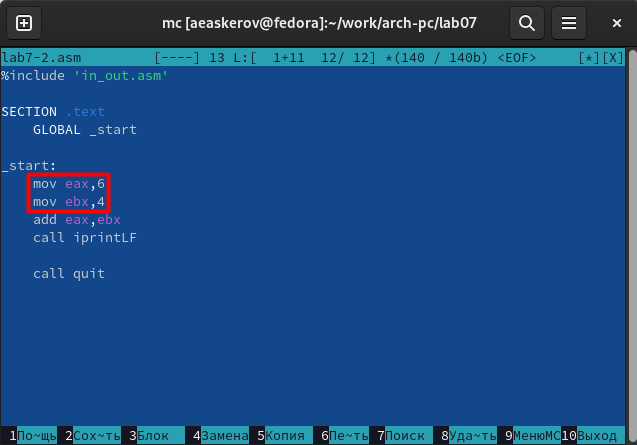


Рис. 9: Изменённый текст программы lab7-2

Создадим исполняемый файл и запустим его. В результате мы получили 10.

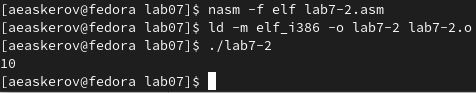


Рис. 10: Результат работы программы lab7-2

Заменим функцию iprintLF на iprint. Создадим исполняемый файл и запустим его. Отличие вывода функции iprintLF от iprint заключается в том, что исчезает перенос строки после вывода информации на экран.

## 2.2 Выполнение арифметических операций в NASM

В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения f(x) = (5 \* 2 + 3)/3.

Создадим файл lab7-3.asm в каталоге ~/work/arch-pc/lab07.

Рис. 11: Создание файла lab7-3.asm в каталоге lab07

Рис. 11: Создание файла lab7-3.asm в каталоге lab07

Внимательно изучим текст программы из листинга 7.3 и введём в lab7- 3.asm.

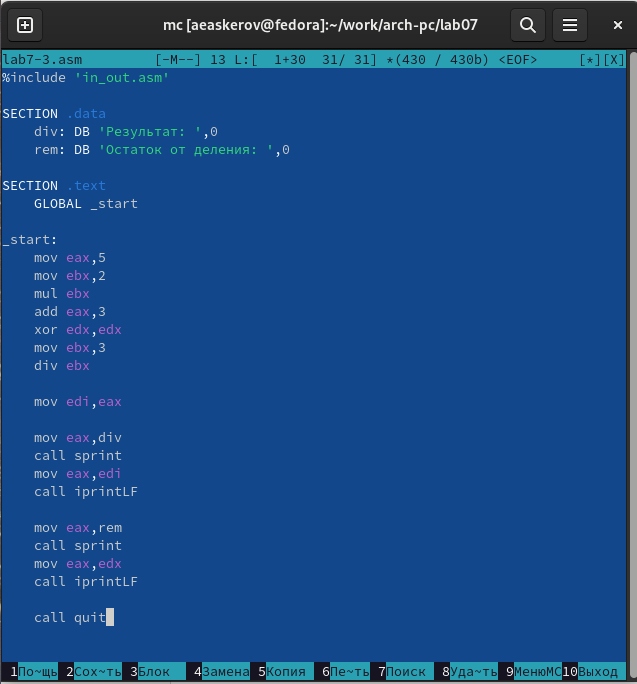


Рис. 12: Программа из листинга 7.3

Создадим исполняемый файл и запустим его.

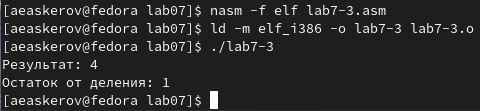


Рис. 13: Результат работы программы lab7-3

Изменим текст программы для вычисления выражения f(x) = (4 \* 6 + 2)/5.

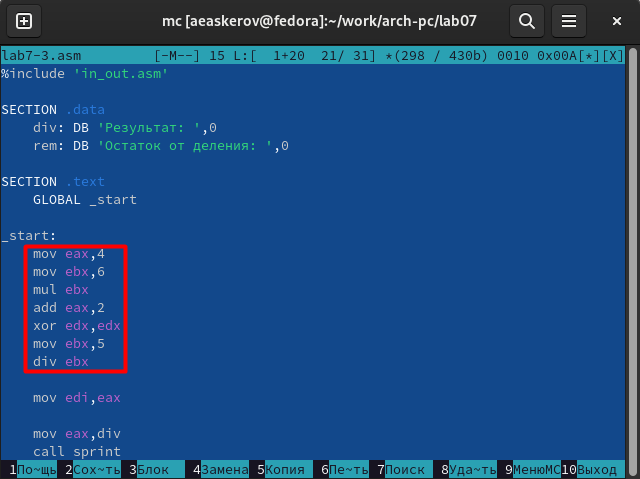


Рис. 14: Изменённый текст программы lab7-3

Создадим исполняемый файл и проверим его работу.

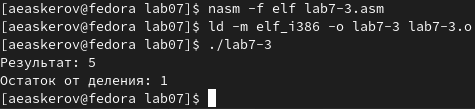


Рис. 15: Результат работы программы lab7-3

1. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму:

* вывести запрос на введение № студенческого билета
* вычислить номер варианта по формуле: (Sn mod 20) + 1, где Sn – номер студенческого билета (в данном случае a mod b – это остаток от деления a на b)
* вывести на экран номер варианта

В данном случае число, над которым необходимо проводить арифметические операции, вводится с клавиатуры. Как отмечалось выше ввод с клавиатуры осуществляется в символьном виде, и для корректной работы арифметических операций в NASM символы необходимо преобразовать в числа. Для этого может быть использована функция atoi из файла in\_out.asm.

Создадим файл variant.asm в каталоге ~/work/arch-pc/lab07.

Рис. 16: Создание файла variant.asm в каталоге lab07

Рис. 16: Создание файла variant.asm в каталоге lab07

Внимательно изучим текст программы из листинга 7.4 и введём в файл variant.asm.

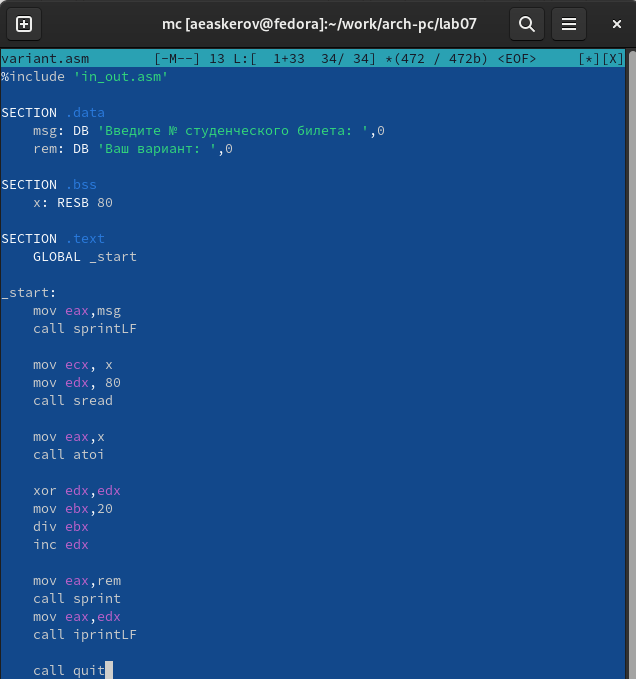


Рис. 17: Программа из листинга 7.4

Создадим исполняемый файл и запустим его.

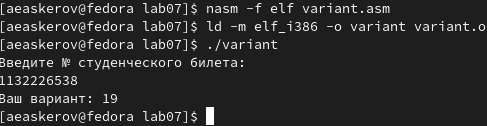


Рис. 18: Результат работы программы variant

Ответы на вопросы:

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

Рис. 19: Вывод на экран сообщения ‘Ваш вариант:’

Рис. 19: Вывод на экран сообщения ‘Ваш вариант:’

1. Для чего используется следующие инструкции? Nasm mov ecx, x mov edx, 80 call sread.

Для того чтобы пользователь ввёл с клавиатуры сообщение размером не больше 80 байт, то есть студенческий номер, в переменную x.

1. Для чего используется инструкция “call atoi”?

Для того чтобы можно было оперировать не с символами из ASCII, стоящими под определёнными номерами, а с числами. “atoi– функция преобразует ascii-код символа в целое число и записывает результат в регистр eax, перед вызовом atoi в регистр eax необходимо записать число (mov eax,)”.

1. Какие строки листинга 7.4 отвечают за вычисление варианта?

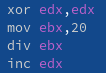


Рис. 20: Строки, отвечающие за вычисление варианта

1. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?

В регистр edx.

1. Для чего используется инструкция “inc edx”?

Для того чтобы увеличить ответ, то есть остаток от деления номера студенческого билета на двадцать, на единицу в соответствии с условием задания.

1. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

Рис. 21: Строки, отвечающие за вывод на экран результата вычислений

Рис. 21: Строки, отвечающие за вывод на экран результата вычислений

## 2.3 Задание для самостоятельной работы

1. Написать программу вычисления выражения y = f(x). Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x, вычислять заданное выражение в зависимости от введенного x, выводить результат вычислений. Вид функции f(x) выбрать из таблицы 7.3 вариантов заданий в соответствии с номером, полученным при выполнении лабораторной работы. Создайте исполняемый файл и проверьте его работу для значений x1 и x2 из 7.3.

Функция 19: f(x)=(x*1/3+5)*7. x1=3;x2=9.

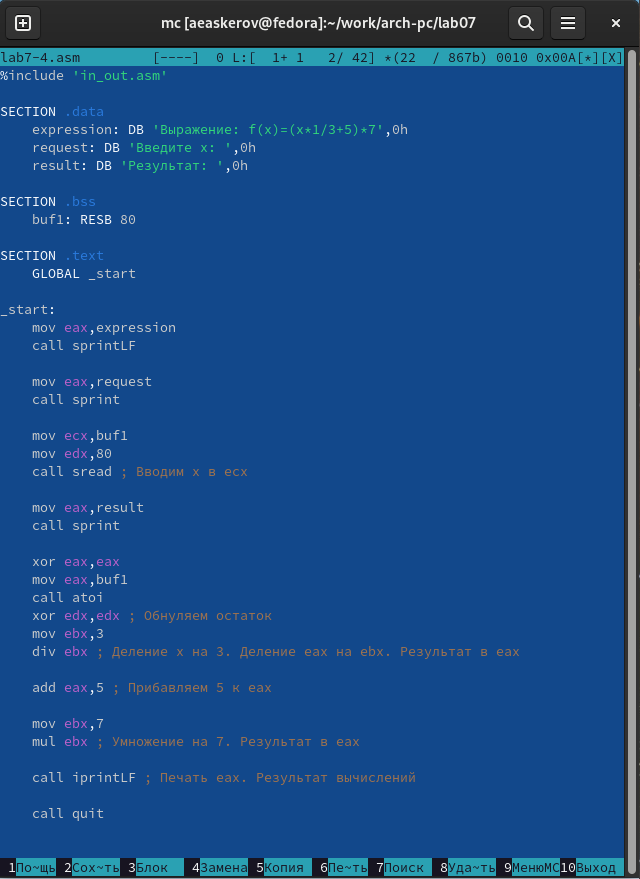


Рис. 22: Программа lab7-4 для самостоятельной работы

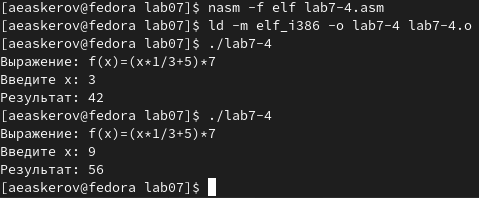


Рис. 23: Результат работы программы lab7-4

# 3 Выводы

Освоены арифметические инструкции языка ассемблера NASM.