Отчёт по лабораторной работе №7

Арифметические операции в NASM

Кочина Дарья Сергеевна

Содержание

# 1 Цель работы

Целью данной лабораторной работы является освоение арифметических инструкций языка ассемблера NASM.

# 2 Задание

Освоить арифметические инструкции языка ассемблера NASM.

# 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. 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 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.

# 4 Выполнение лабораторной работы

1. Я создала каталог для программам лабораторной работы № 7, перешла в него и создала файл lab7-1.asm. (рис. 1)

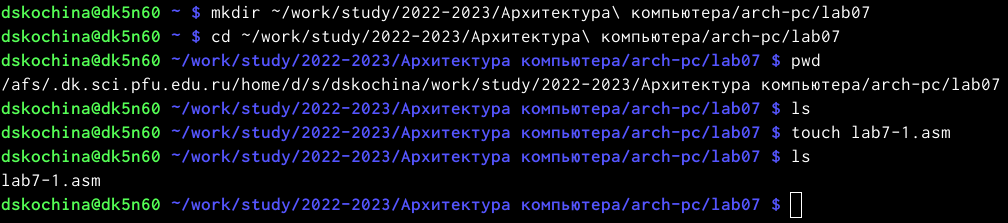


Рис. 1: Создание файла lab7-1.asm

1. Я ввела нужный текст в файле lab7-1.asm, создала файл и проверила его. Программа вывела “j”. (рис. 2, 3)

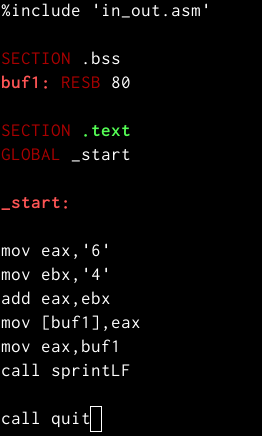


Рис. 2: Текст в файле lab7-1.asm

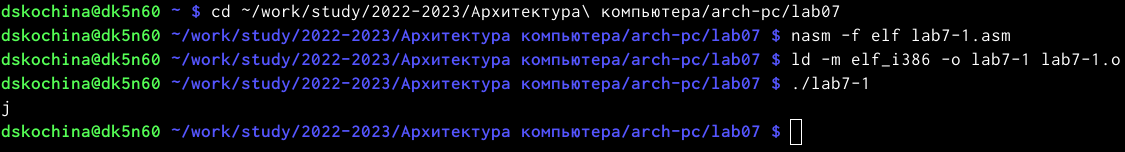


Рис. 3: Создание файла и проверка работы

1. Я изменила текст программы и вместо символов записала в регистры числа. Я создала исполняемый файл и запустила его. Программа вывела невидимый символ. (рис. 4, 5)

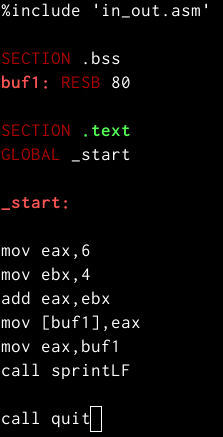


Рис. 4: Изменения в программе lab6-1.asm

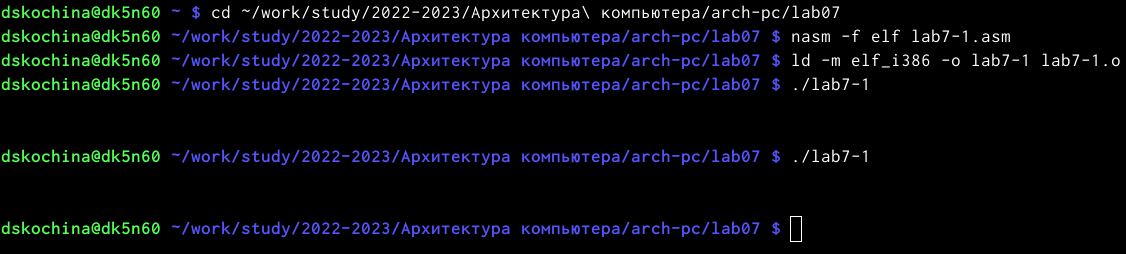


Рис. 5: Проверка программы

1. Я создала файл lab7-2.asm и ввела в него текст программы. (рис. 6, 7)

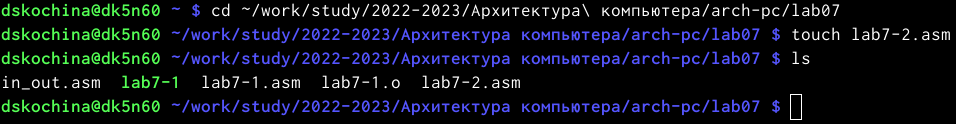


Рис. 6: Создание файла lab7-2.asm

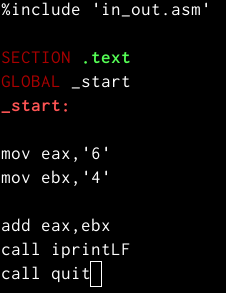


Рис. 7: Текст программы из листинга

1. Я создала исполняемый файл и запустила его. В результате работы программы я получила число 106. (рис. 8)

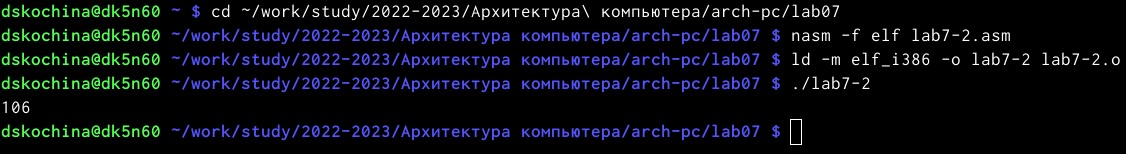


Рис. 8: Запуск программы lab7-2.asm

1. Я изменила текст программы lab7-2.asm и запустила её. Она вывела число 10. (рис. 9, 10)

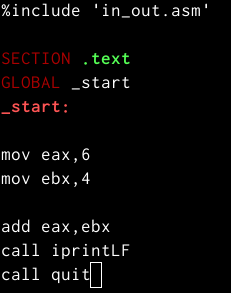


Рис. 9: Изменения в программе lab6-2.asm

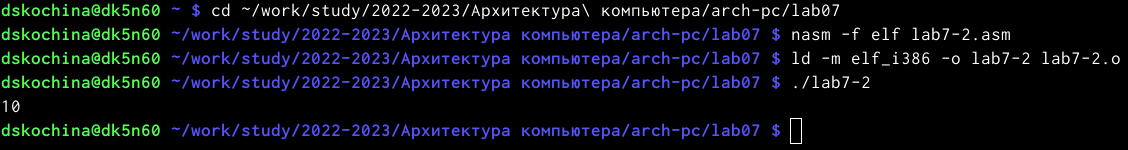


Рис. 10: Проверка программы

1. Я заменила функцию iprintLF на iprint. Создала исполняемый файл и запустила его. Отличие вывода функций iprintLF от iprint заключается в том, что при использовании команды iprintLF мы начинаем вводить команду на следующей строке, а при использовании команды iprint мы вводим следующие данные на той же строке. (рис. 11, 12)

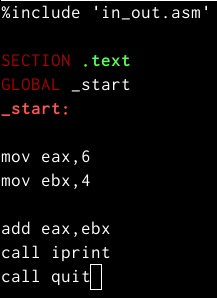


Рис. 11: Замена iprintLF на iprint

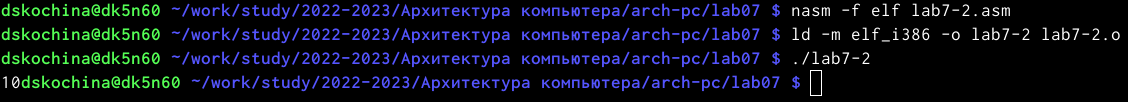


Рис. 12: Запуск программы с изменениями

1. Я создала файл lab7-3.asm и ввела текст программы из листинга. Создала исполняемый файл и запустила его. (рис. 13, 14, 15)

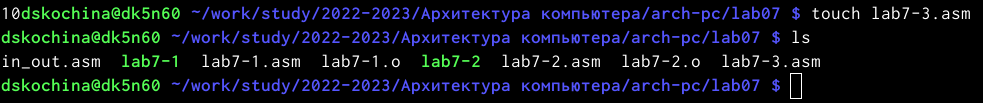


Рис. 13: Создание файла lab7-3.asm

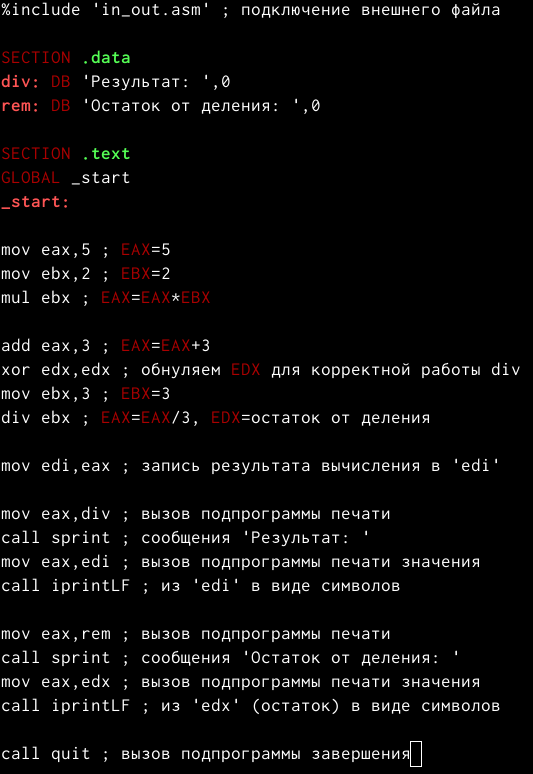


Рис. 14: Текст программы из листинга

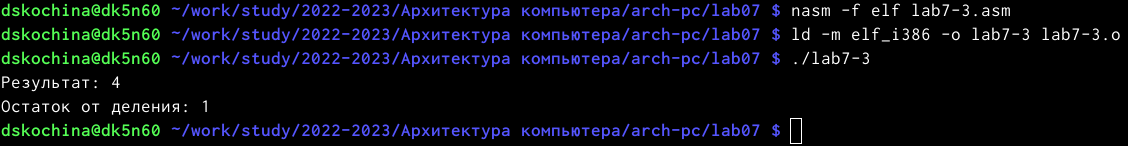


Рис. 15: Запуск программы lab7-3.asm

1. Я изменила текст программы для вычисления выражения f(x)=(4\*6+2)/5. Создала исполняемый файл и проверила его работу. (рис. 16, 17)

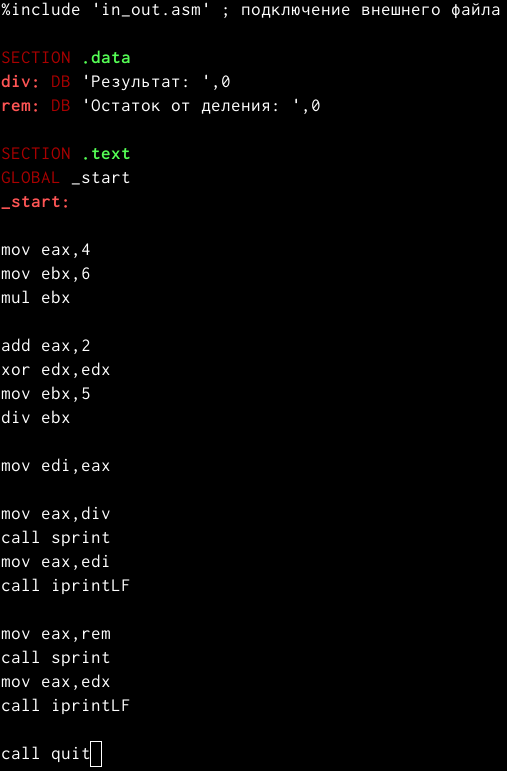


Рис. 16: Изменения в программе lab7-3.asm

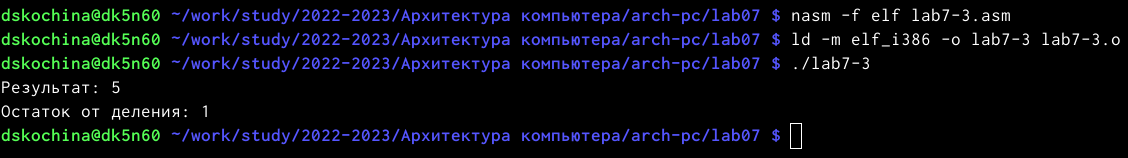


Рис. 17: Запуск программы

1. Я создала файл variant.asm. Ввела текст программы из листинга в файл. (рис. 18, 19)

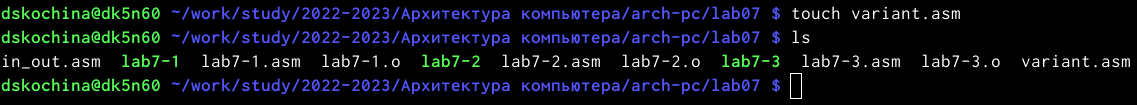


Рис. 18: Создание файла variant.asm

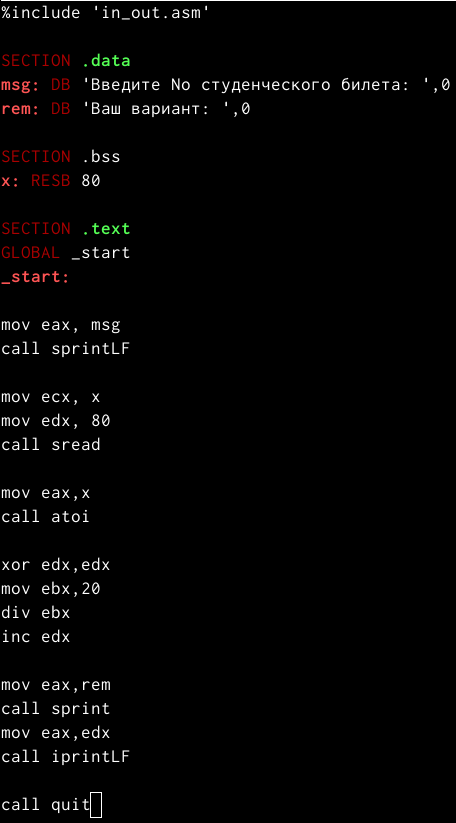


Рис. 19: Текст программы из листинга

1. Я создала исполняемый файл и запустила его. Проверила работу программы, вычислив номер варианта аналитически. Результат совпал. (рис. 20)

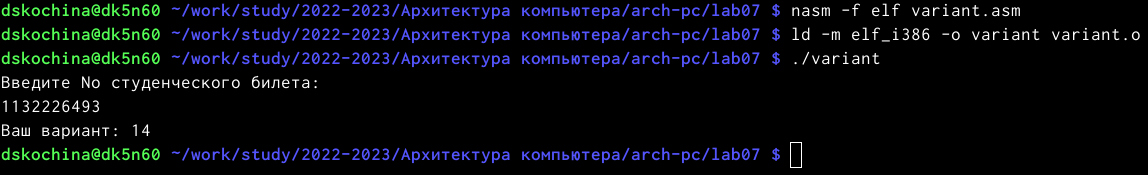


Рис. 20: Запуск программы variant.asm

**Вопросы**

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

Строки mov eax,rem call sprint.

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

Инструкция mov ecx,x записывает адресы выводимого соообщения в EAX. Инструкция mov edx,80 записывает длину вводимого сообщения в EBX. Инструкция call sread выполняет вызов программы ввода сообщения.

1. Для чего используется инструкция “call atoi”?

Эта инструкция используется для преобразования кода переменной ASCII в число.

1. Какие строки листинга 7.4 отвечают за вычисления варианта?

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

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

Остаток от деления при выполнении инструкции “div ebx” записывается в в регистр EBX.

1. Для чего используется инструкция “inc edx”?

Эта инструкция используется для увеличения значения edx на единицу.

1. Какие строки листинга 7.4 отвечают за вывод на экран результата вычис- лений?

mov eax,edx call iprintLF

**Самостоятельная работа**

1. Я создала файл lab7-4.asm для выполнения самостоятельной работы. (рис. 21)

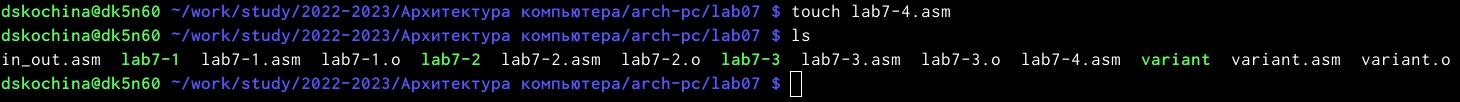


Рис. 21: Создание файла lab7-4.asm

1. Я написала программу вычисления выражения f(x)=(x/2+8)3. Программа вывела выражение для вычисления, вывела запрос на ввод значения x, вычислила заданное выражение в зависимости от введённого x, вывела результат вычислений. Вид функции f(x) я выбрала из таблицы вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Я создала исполняемый файл и проверила его работу. Для этого я вводила значения для x, равные 1 и 4. Результаты работы программы были верными. При вводе 1 программа вывела ответ 24, т.к. учитывалась только целая часть от деления. При вводе 4 ответ оказался 30, что является верным. (рис. 22, 23, 24)

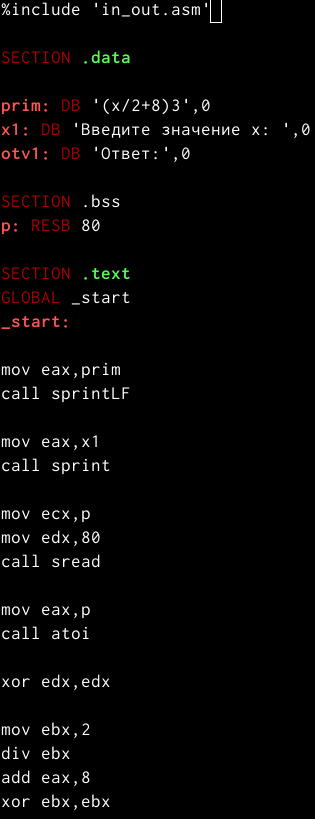


Рис. 22: Программа для вычисления в файле lab7-4.asm

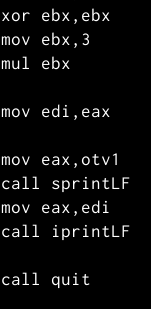


Рис. 23: Программа для вычисления в файле lab7-4.asm

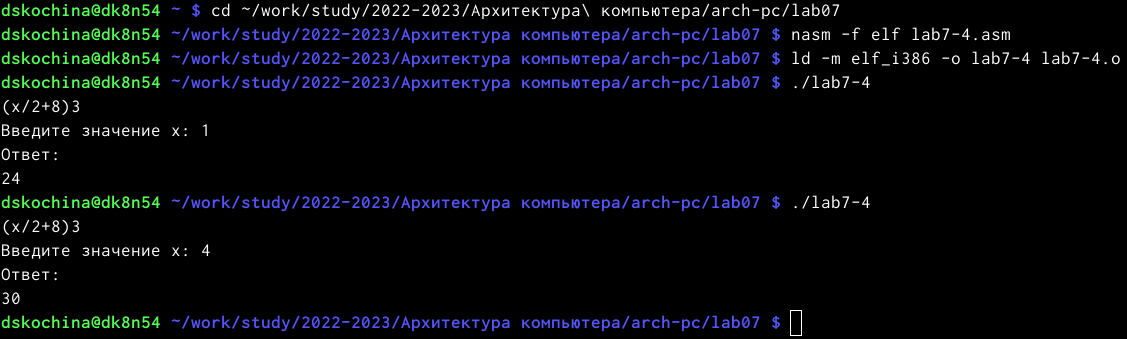


Рис. 24: Результаты работы программы

# 5 Выводы

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