Лабораторная работа №7

Арифметические операции в NASM

Медникова Екатерина Михайловна

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Теоретическое введение

Микропроцессор может выполнять целочисленные операциии и операции с плавающей точкой. Для этого в его архитектуре есть два отдельных блока:

устройство для выполнения целочисленных операций;  
  
устройство с плавающей точкой.

Каждое из этих устройств имеет свою систему команд. Целочисленное устройство может взять на себя многие функции устройства с плавающей точкой, но это потребует больших вычислительных затрат. Для большинства задач, использующих язык ассемблера, достаточно целочисленной арифметики.

# 3 Выполнение лабораторной работы

1. Создала каталог и файл.

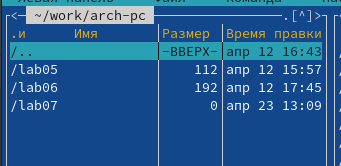


Figure 1: Создание каталога

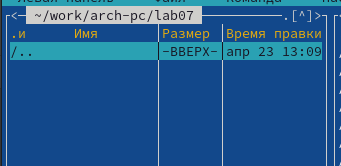


Figure 2: Переход в каталог

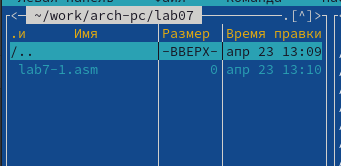


Figure 3: Создание файла

1. Ввела в файл текст программы из листинга 7.1. Создала исполняемый файл и запустила его.

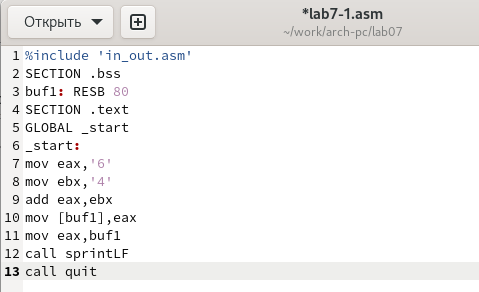


Figure 4: Ввод текста

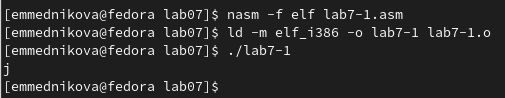


Figure 5: Результат

1. Изменила в файле текст программы, как было указано в лабораторной работе. Создала исполняемый файл и запустила его.

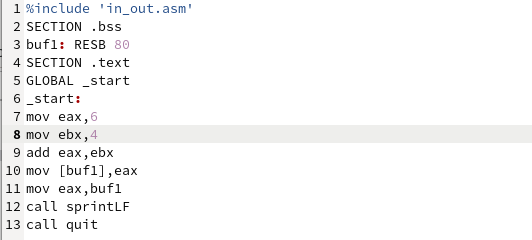


Figure 6: Изменение текста

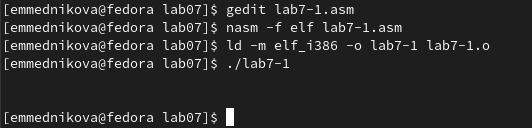


Figure 7: Результат

1. Создала файл lab7-2.asm и ввела в него текст программы из листинга 7.2. Создала исполняемый файл и запустила его.

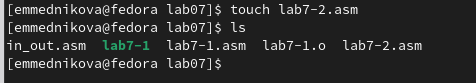


Figure 8: Создание файла

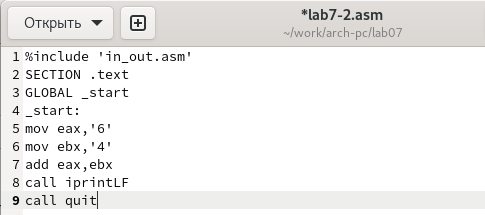


Figure 9: Ввод текста

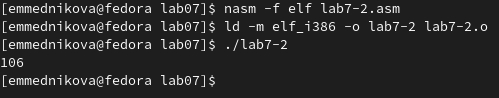


Figure 10: Результат

1. Изменила символы на числа. Создала исполняемый файл и запустила его. При исполнении программы результат получился 10.

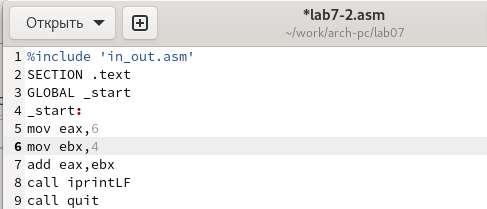


Figure 11: Корректировка текста

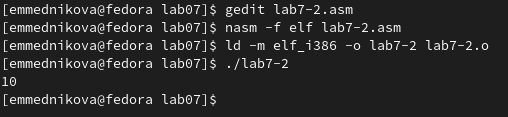


Figure 12: Результат

1. Заменила функцию iprintLF на iprint. Создала исполняемый файл и запустила его. При iprint числа вводятся в формате ASCII. Функция iprintLF работает аналогично, но при выводе на экран после числа добавляет к символу перевод строки.

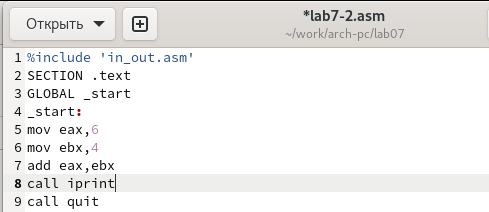


Figure 13: Замена функции

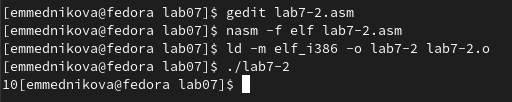


Figure 14: Результат

1. Создала файл lab7-3.asm. Ввела в него текст программы из листинга 7.3.

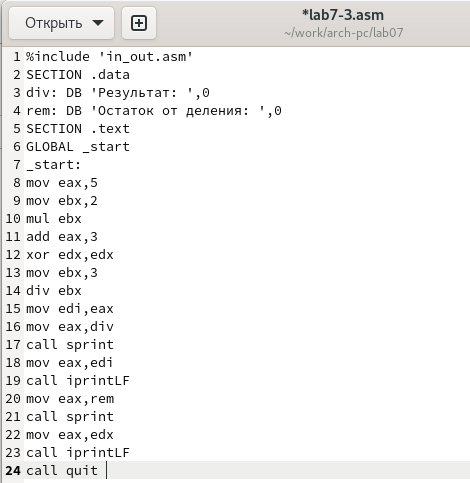


Figure 15: Ввод текста программы

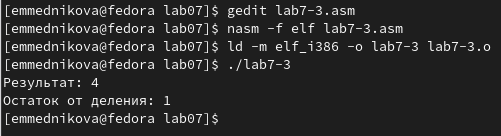


Figure 16: Результат

1. Изменила текст программы для вычисления выражения. Создала исполняемый файл и проверила его работу.

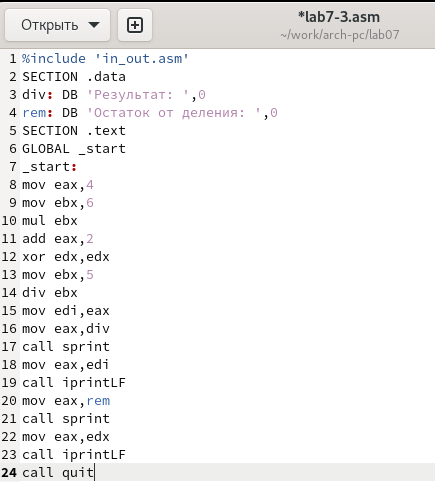


Figure 17: Изменение текста

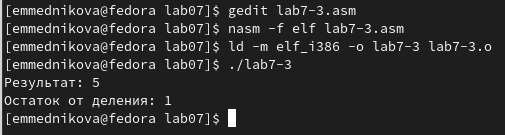


Figure 18: Проверка работы

1. Создала файл variant.asm. Ввела в него текст программы из листинга 7.4. Создала исполняемый файл и запустила его.

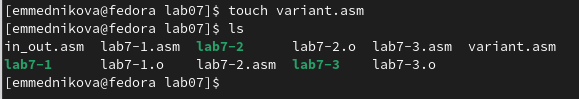


Figure 19: Создание файла

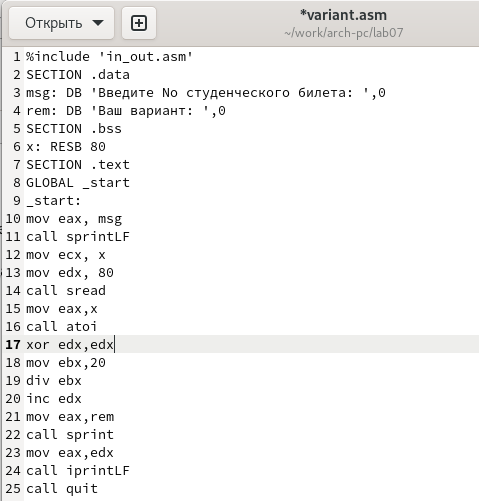


Figure 20: Ввод текста программы

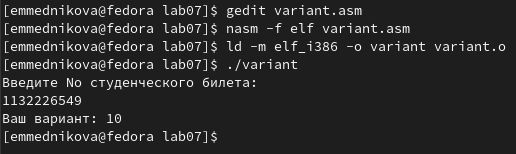


Figure 21: Результат

# 4 Ответы на вопросы

1. За вывод на экран сообщения ‘Ваш вариант:’ отвечают строки mov eax, rem call sprint.
2. nasm mov ecx - записывает адрес выводимого сообщения в eax;

mov edx, 80 - записывает длину вводимого сообщения в ebx;

call sread - выполняет вызов программы ввода сообщения.

1. Инструкция ‘call atoi’ используется для преобразования символа в число.
2. За вычисление варинта отвечают строки: xor edx, edx mov ebx,20 div ebx inc edx.
3. Остаток от деления записывается в регистр ebx.
4. Инструкция ‘inc edx’ используется для увеличения значения edx на единицу.
5. За вывод на экран результата вычислений отвечают строки: mov eax, edx call iprintLF

# 5 Самостоятельная работа

1. Перед началом работы создала файл. Написала программу для вычисления выражения из варианта 10, который был получен ранее при выполнении лабораторной работы.

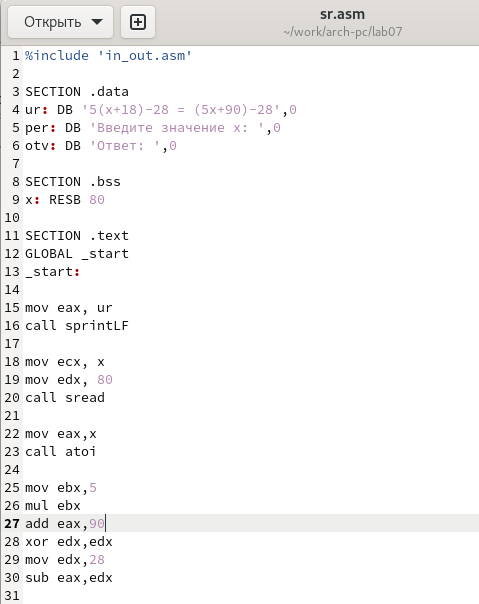


Figure 22: Программа

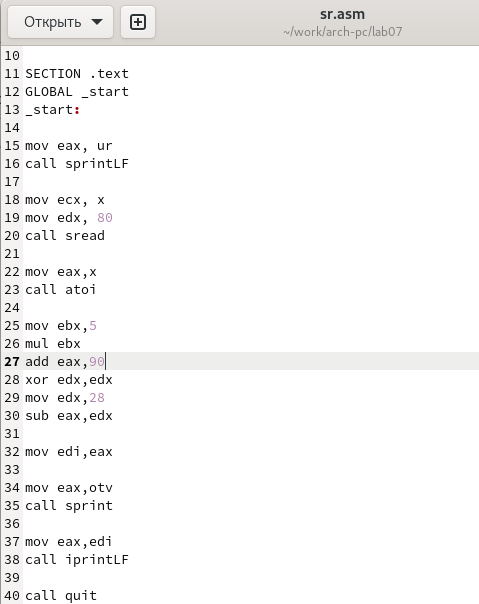


Figure 23: Программа

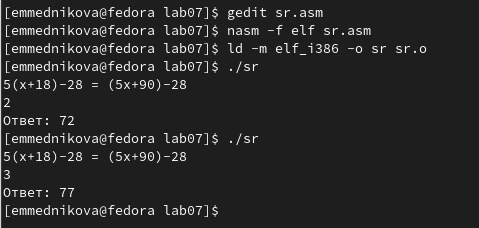


Figure 24: Результат

# 6 Выводы

Освоила арифметические инструкции языка ассемблера NASM.