Отчет по лабораторной работе №6.

Арифметические операции в NASM

Зайцева П. Е.

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Теоретическое введение

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 ко- манде, Например: mov ax,2. • Адресация памяти – операнд задает адрес в памяти. В команде указывается символи- ческое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

# 3 Выполнение лабораторной работы

Создала каталог для программ по лабораторных работ №6, перешла в него и создала файл lab6-1.asm

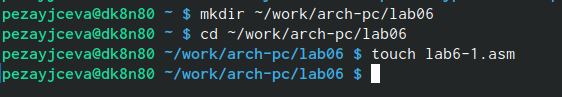


рис.1

Ввела в файл lab6-1.asm текст программы из листинга 6.1. Далее создала исполняемый файл и запустила его.

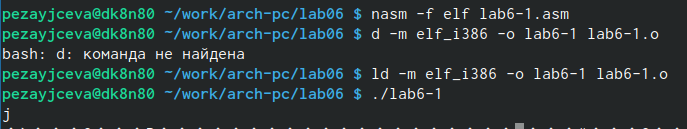


рис.2

В данном случае при выводе значения регистра eax мы ожидаем увидеть число 10. Однако результатом будет символ j.

После изменила текст программы и вместо символов, записала в регистры числа.

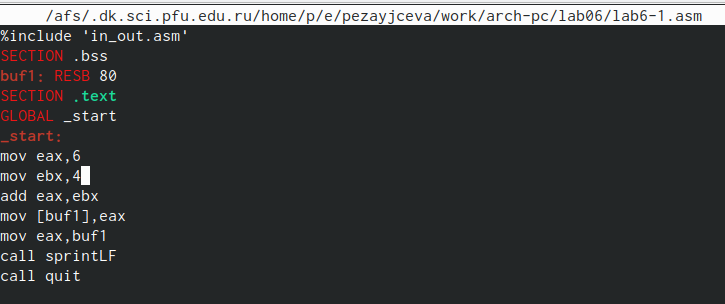


рис.3

Создала исполняемый файл и запустила его.

Создала файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 .

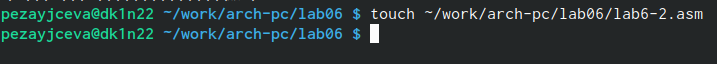


рис.5

Ввела в него текст программы из листинга 6.2.

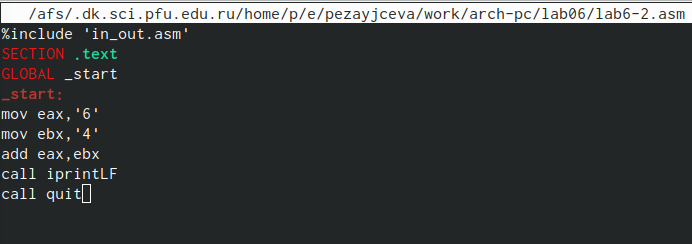


рис.6

Создала исполняемый файл и запустила его.

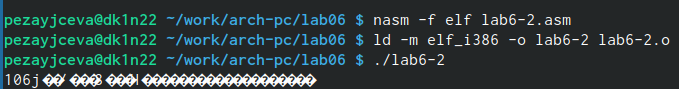


рис.7

В результате работы программы получила число 106.

Аналогично предыдущему примеру изменила символы на числа.

Создала исполняемый файл и запустила его. Результат 10.

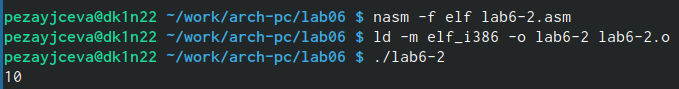


рис.8

Создала файл lab6-3.asm в каталоге ~/work/arch-pc/lab06

рис.9

рис.9

Из листинга 6.3 ввела текст в lab6-3.asm

Создала исполняемый файл и запустила его.

Далее изменила текст программы под выражение 𝑓(𝑥) = (4 ∗ 6 + 2)/5 и заупстила.

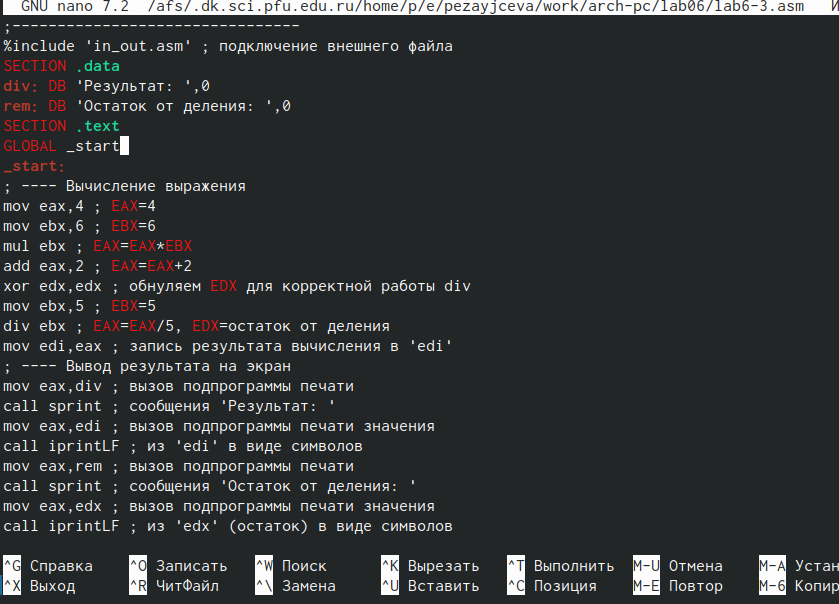


рис.10

Создала файл variant.asm в каталоге ~/work/arch-pc/lab06

Запустила файл и получила свой номер варианта.

рис.15

рис.15

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

mov eax,rem call sprint

1. 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 sread - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры
2. call atoi используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр eax
3. 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

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

mov eax,edx call iprintLF

Написать программу вычисления выражения 18(𝑥 + 1)/6

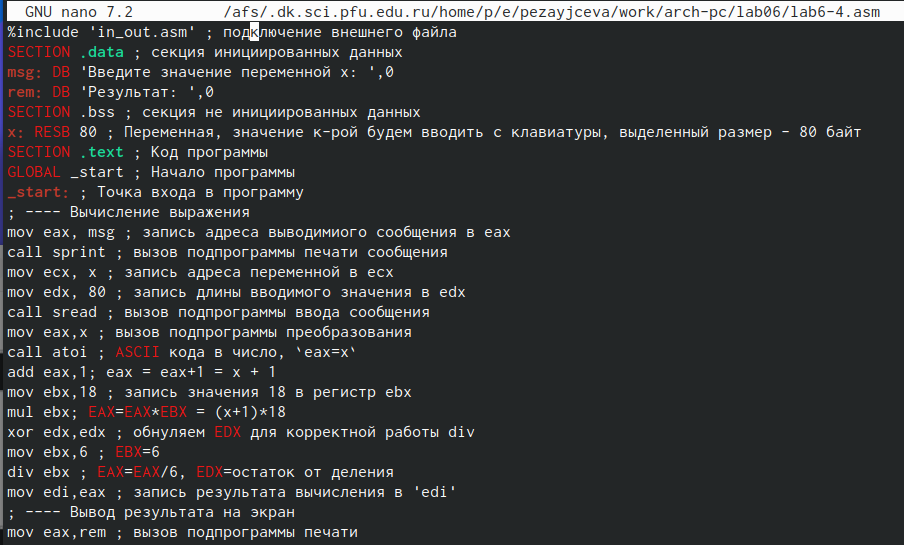


рис.12

Вывод при х=3

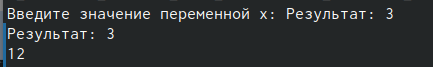


рис.13

Вывод при х=1

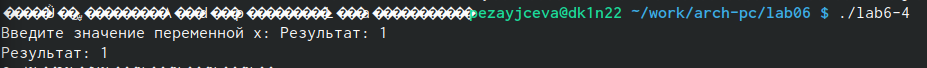


рис.14

# 4 Выводы

Освоила арифметические инструкции языка ассемблера NASM.

# Список литературы