Лабораторная работа №6

Архитектура компьютера

Казначеева Кристина Никитична

Содержание

# 1 Цель работы

В рамках лабораторной работы мы изучим арифметические инструкции языка ассемблера NASM, чтобы в дальнейшем научиться решать простые арифметические задачи с помощью ассемблера.

# 2 Задание

В этой лабораторной работе мы познакомимся с основными типами данных в ассемблере NASM, освоим выполнение арифметических операций, научимся выводить значения регистров на экран и напишем программу для вычисления арифметических выражений.

# 3 Выполнение лабораторной работы

Создадим каталог и перейдём в него, затем создадим файл lab6-1.asm: (рис. 1).

Рис. 1: Создание каталога и файла

Рис. 1: Создание каталога и файла

Вводим в файл lab6-1.asm текст программы вывода значения регистра eax (рис. 2).

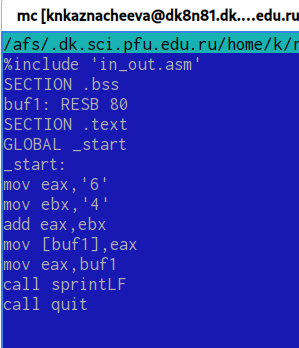


Рис. 2: Ввод текста программы

Создаём исполняемый файл и запускаем его (рис. 3).

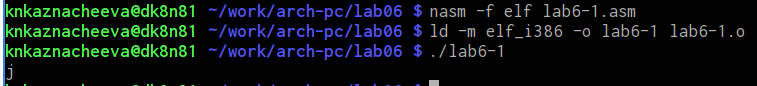


Рис. 3: Запуск файла

Далее изменим текст программы и вместо символов, запишем в регистры числа (рис. 4).

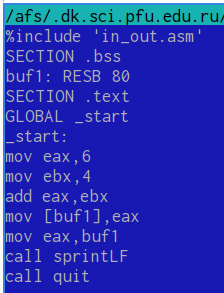


Рис. 4: Замена текста программы

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

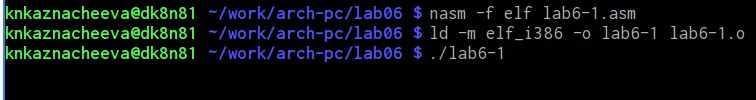


Рис. 5: Запуск файла

Создаём файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 (рис. 6).

Рис. 6: Создание файла

Рис. 6: Создание файла

Вводим в него текст программы вывода значения регистра eax (рис. 7).

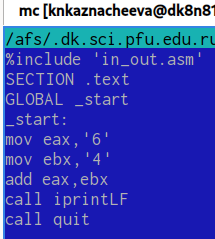


Рис. 7: Ввод текста программы

Создаём исполняемый файл и запускаем его (рис. 8):

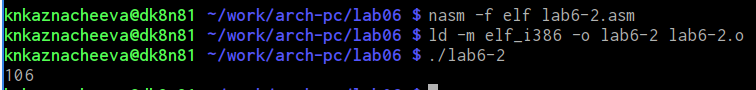


Рис. 8: Запуск файла

Далее заменяем текст программы и вместо символов, записываем в регистры числа (рис. 9).

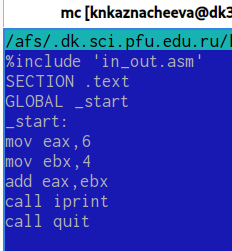


Рис. 9: Замена текста программы

Создаём исполняемый файл и запускаем его (рис. 10).

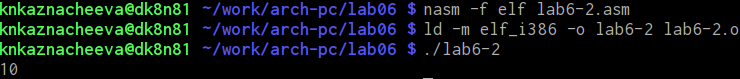


Рис. 10: Запуск файла

Заменяем функцию iprintLF на iprint (рис. 11).

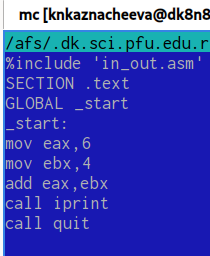


Рис. 11: Замена подпрограммы sprintLF на sprint

При замене функции iprintLF на iprint сообщение выводится в одну строку (рис. 12).

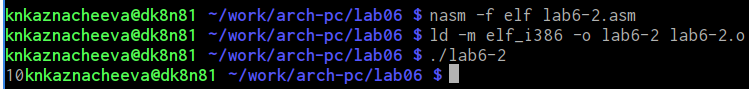


Рис. 12: Замена подпрограммы sprintLF на sprint

Создаём файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. 13).

Рис. 13: Создание файла

Рис. 13: Создание файла

Вводим в файл lab6-3.asm текст программы вычисления выражения f(x) = (5 ∗ 2 + 3)/3 (рис. 14).

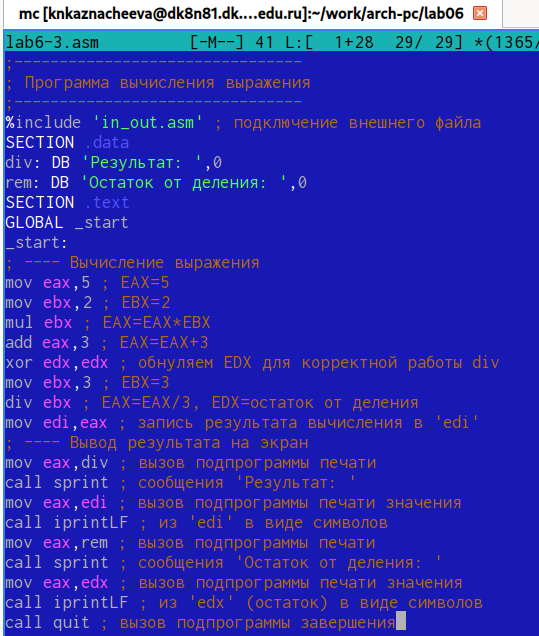


Рис. 14: Ввод текста программы

Создаём исполняемый файл и запускаем его (рис. 15).

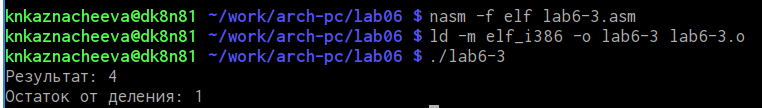


Рис. 15: Запуск получившегося файла

Измените текст программы для вычисления выражения f(x) = (4 ∗ 6 + 2)/5 (рис. 16).

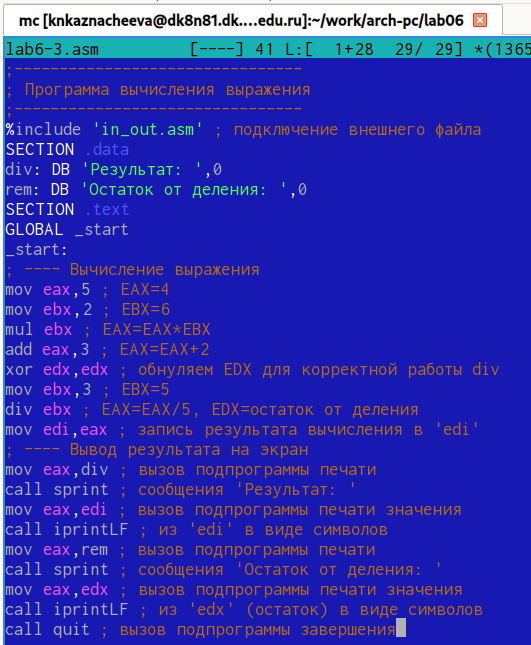


Рис. 16: Замена текста программы

Создаём исполняемый файл и проверяем его работу (рис. 17).

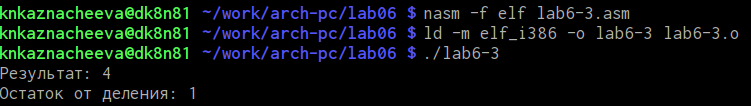


Рис. 17: Запуск получившегося файла

Создаём файл variant.asm в каталоге ~/work/arch-pc/lab06, (рис. 18).

Рис. 18: Создание файла

Рис. 18: Создание файла

Затем вводим в него текст программы вычисления варианта задания по номеру студенческого билета (рис. 19).

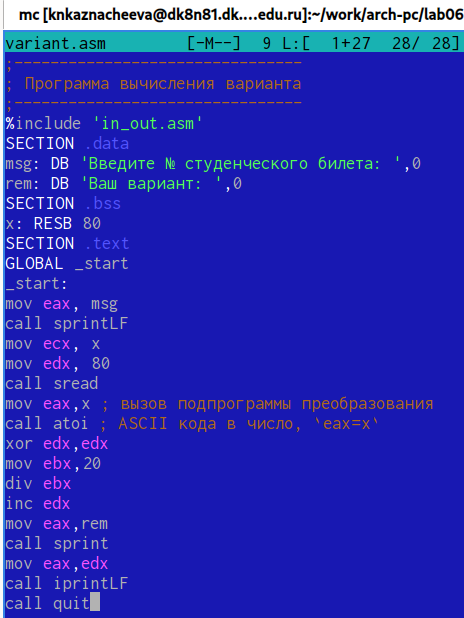


Рис. 19: Воод текста программы

Создаём исполняемый файл и запускаем его, затем проверим результат работы программы, вычислив номер варианта аналитически (рис. 20).

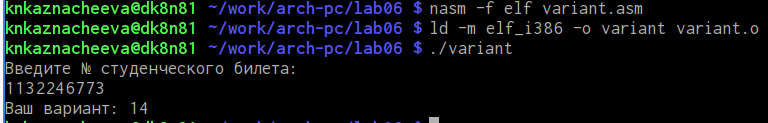


Рис. 20: Запуск получившегося файла

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? • mov eax, rem: Эта строка загружает адрес сообщения “Ваш вариант:” (которое находится в переменной rem) в регистр eax. • call sprint: Эта строка вызывает функцию sprint, которая выводит строку, адрес которой находится в eax. Таким образом, эти две строки совместно выводят на экран сообщение “Ваш вариант:”.
2. Для чего используется следующие инструкции? • mov ecx, x: Эта инструкция загружает адрес буфера x в регистр ecx. Регистр ecx используется в системе вызовов для указания адреса буфера, куда будет записана введенная строка. • mov edx, 80: Эта инструкция загружает значение 80 в регистр edx. Регистр edx указывают максимальную длину вводимой строки (80 байт). • call sread: Эта инструкция вызывает функцию sread. Функция sread - это внешняя функция, которая занимается считыванием данных с клавиатуры. Функция sread использует значения из ecx (адрес буфера) и edx (максимальная длина) для корректного ввода и записи данных в буфер x.
3. Для чего используется инструкция “call atoi”? Инструкция call atoi используется для преобразования строки ASCII-символов в целое число.
4. Какие строки листинга 6.4 отвечают за вычисления варианта? Строки, которые отвечают за вычисление варианта, это: xor edx, edx ; Обнуляем EDX для div mov ebx, 20 div ebx ; eax = x / 20, edx = x % 20 inc edx ; edx = (x % 20) + 1
5. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”? Остаток от деления при выполнении инструкции div ebx записывается в регистр edx.
6. Для чего используется инструкция “inc edx”? Инструкция inc edx увеличивает значение в регистре edx на 1. В данной программе она используется, чтобы вариант студента был в диапазоне от 1 до 20, а не от 0 до 19.
7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений? За вывод результата вычислений отвечают следующие строки: mov eax, rem call sprint mov eax, edx call iprintLF

Создадим файл lab6-4.asm и напишем программу вычисления выражения для 14 варианта: y(x)=( x/2 + 8) ⋅ 3 (рис. 21).

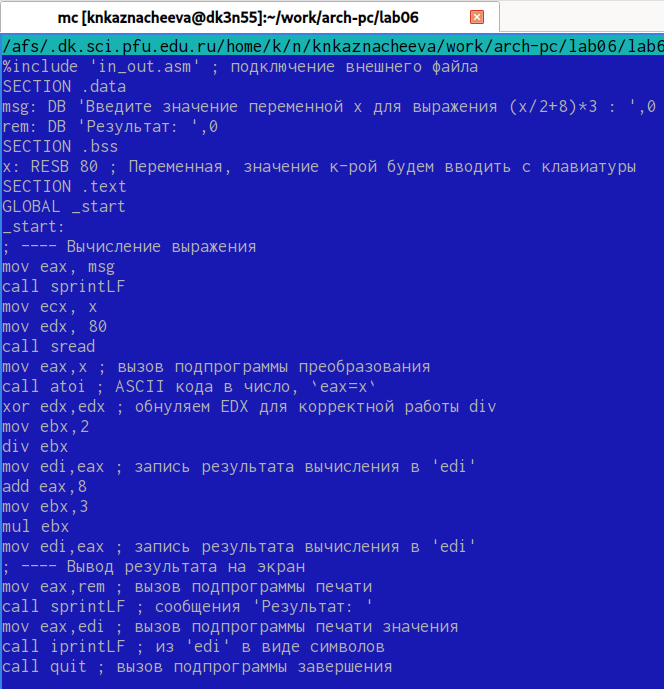


Рис. 21: Программа для 14 варианта

Создаём исполняемый файл и проверяем его работу для значения x1 = 1 (рис. 22).

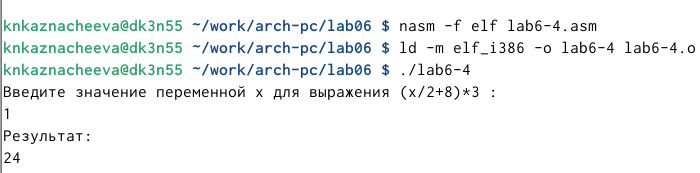


Рис. 22: Проверка файла при x1

Затем проверяем работу файла для значения x2 = 4 (рис. 23)

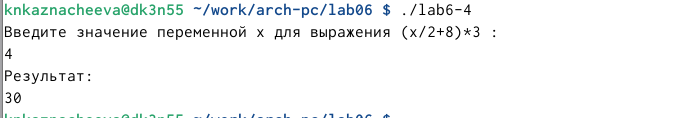


Рис. 23: Проверка файла при x2

# 4 Вывод

В ходе лабораторной работы мы получили практические навыки работы с ассемблером NASM: освоили типы данных, арифметические операции, вывод данных на экран и реализовали программу для вычисления выражений.