Отчёт по лабораторной работе 6

Архитектура компьютеров

Сюй Хайфэн

Содержание

# 1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

# 2 Выполнение лабораторной работы

## 2.1 Символьные и численные данные в NASM

Я создал каталог для программ лабораторной работы № 6, перешел в него и создал файл lab6-1.asm.

В этом разделе рассмотрим примеры программ, которые выводят символьные и численные значения. Программы будут выводить данные, записанные в регистр eax.

В данной программе в регистр eax записывается символ ‘6’ (с помощью команды mov eax, '6'), в регистр ebx записывается символ ‘4’ (с помощью команды mov ebx, '4'). Затем к значению в регистре eax прибавляется значение из регистра ebx (командой add eax, ebx), и результат сохраняется в eax. После этого выводим результат.

Так как для работы функции sprintLF в регистр eax должен быть записан адрес, создаем дополнительную переменную. Сначала записываем значение из регистра eax в переменную buf1 (команда mov [buf1], eax), затем записываем адрес этой переменной в регистр eax (команда mov eax, buf1) и вызываем функцию sprintLF.

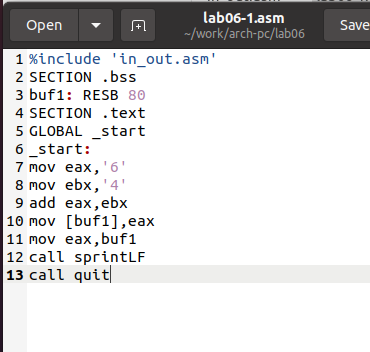


Рис. 1: Программа lab6-1.asm

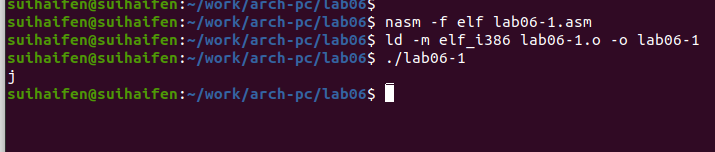


Рис. 2: Запуск программы lab6-1.asm

При выводе значения из регистра eax мы ожидали увидеть число 10, но на самом деле вывелся символ ‘j’. Это связано с тем, что код символа ‘6’ равен 00110110 (54 в десятичной системе), а код символа ‘4’ — 00110100 (52). Когда эти значения сложились (с помощью команды add eax, ebx), результатом стало значение 106, что в свою очередь соответствует символу ‘j’ в таблице ASCII.

Затем я изменил программу, заменив символы на числа.

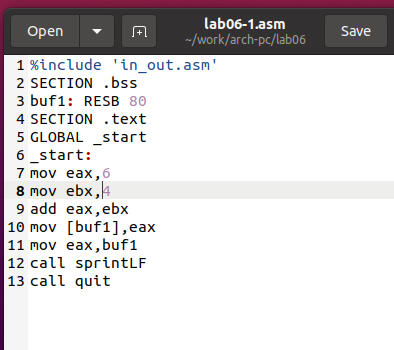


Рис. 3: Программа lab6-1.asm с числами

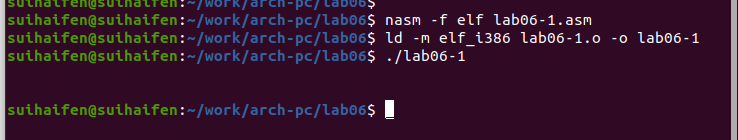


Рис. 4: Запуск программы lab6-1.asm с числами

Как и в предыдущем примере, при исполнении программы мы не получим число 10, а на экране появится символ с кодом 10, который представляет собой символ конца строки (возврат каретки). В консоли он не отображается, но добавляет новую строку.

Для работы с числами в файле in\_out.asm предусмотрены подпрограммы для преобразования символов в числа и наоборот. Я преобразовал программу, используя эти функции.

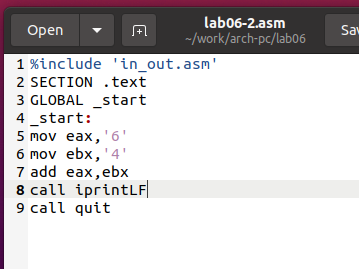


Рис. 5: Программа lab6-2.asm

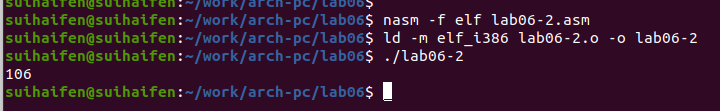


Рис. 6: Запуск программы lab6-2.asm

Результатом выполнения программы стало число 106. Здесь, как и в предыдущем примере, команда add складывает коды символов ‘6’ и ‘4’ (54 + 52 = 106). Однако теперь функция iprintLF позволяет вывести это число, а не символ, код которого равен 106.

Заменил символы на числа, и результат вывода — число 10, так как функции выводят именно числовые значения.

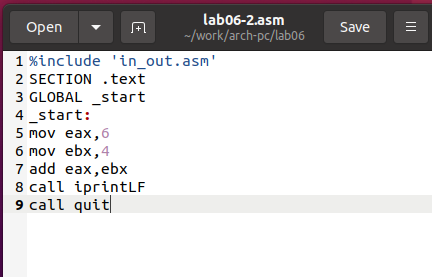


Рис. 7: Программа lab6-2.asm с числами

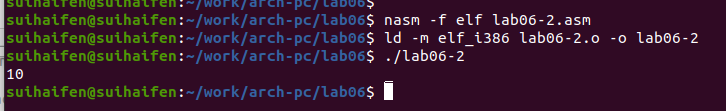


Рис. 8: Запуск программы lab6-2.asm с числами

Позже заменил функцию iprintLF на iprint. Создал исполняемый файл и запустил его. Результат отличается тем, что теперь выводится значение без переноса строки.

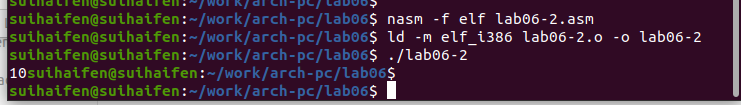


Рис. 9: Запуск программы lab6-2.asm без переноса строки

## 2.2 Выполнение арифметических операций в NASM

Теперь рассмотрим пример программы, которая выполняет арифметические операции, используя выражение $ f(x) = (5 \* 2 + 3)/3 $.

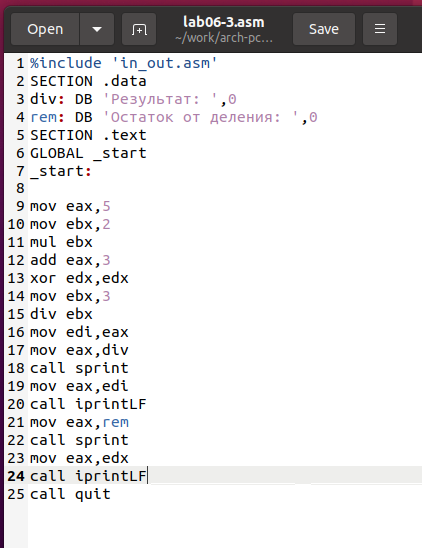


Рис. 10: Программа lab6-3.asm

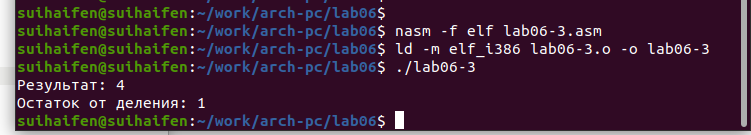


Рис. 11: Запуск программы lab6-3.asm

Я изменил программу для вычисления выражения $ f(x) = (4 \* 6 + 2)/5 $, создал исполняемый файл и проверил его работу.

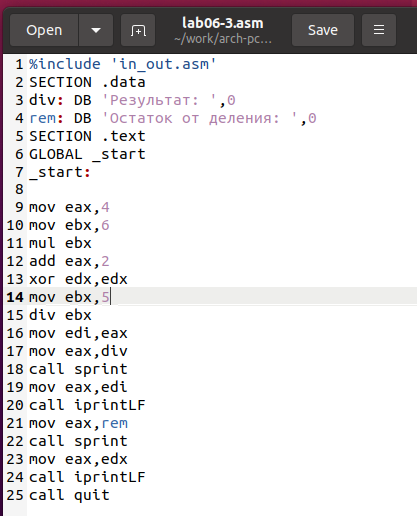


Рис. 12: Программа lab6-3.asm с другим выражением

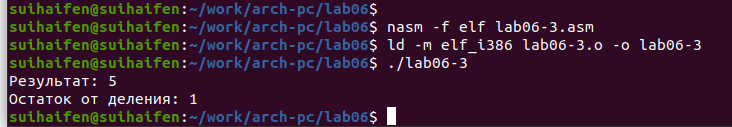


Рис. 13: Запуск программы lab6-3.asm с другим выражением

Еще один пример — программа для вычисления варианта задания, используя номер студенческого билета.

В этом случае значение для вычислений вводится с клавиатуры. Как я уже упоминал, ввод данных осуществляется в символьной форме, и для правильной работы арифметических операций необходимо преобразовать символы в числа. Для этого используется функция atoi из файла in\_out.asm.

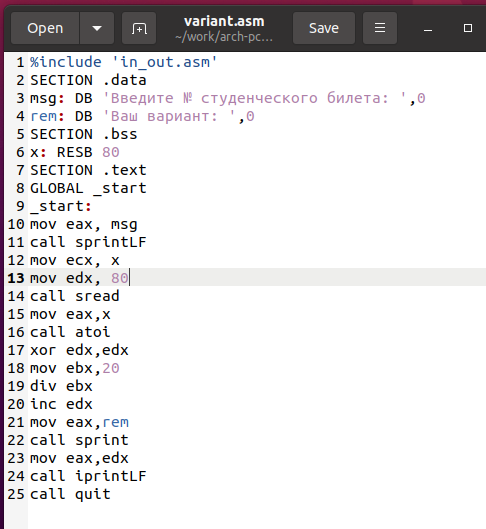


Рис. 14: Программа variant.asm

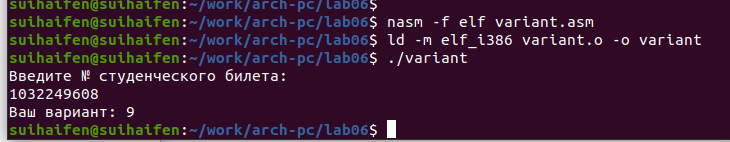


Рис. 15: Запуск программы variant.asm

## 2.3 Ответы на вопросы

1. Какие строки листинга отвечают за вывод на экран сообщения ‘Ваш вариант:’?
   * Инструкция mov eax, rem переносит значение переменной с фразой ‘Ваш вариант:’ в регистр eax.
   * Инструкция call sprint вызывает подпрограмму для вывода строки.
2. Для чего используются следующие инструкции?
   * Инструкция mov ecx, x — для помещения значения переменной x в регистр ecx.
   * Инструкция mov edx, 80 — для помещения значения 80 в регистр edx.
   * Инструкция call sread — для вызова подпрограммы для считывания значения студенческого билета.
3. Для чего используется инструкция “call atoi”?
   * Инструкция “call atoi” используется для преобразования введенных символов в числовой формат.
4. Какие строки листинга отвечают за вычисления варианта?
   * Инструкция xor edx, edx обнуляет регистр edx.
   * Инструкция mov ebx, 20 записывает значение 20 в регистр ebx.
   * Инструкция div ebx выполняет деление номера студенческого билета на 20.
   * Инструкция inc edx увеличивает значение регистра edx на 1.

* В данном случае происходит деление номера студенческого билета на 20. Остаток от деления сохраняется в регистре edx, и к нему прибавляется 1.

1. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?
   * Остаток от деления записывается в регистр edx.
2. Для чего используется инструкция “inc edx”?
   * Инструкция “inc edx” увеличивает значение в регистре edx на 1, что необходимо для вычисления варианта.
3. Какие строки листинга отвечают за вывод на экран результата вычислений?
   * Инструкция mov eax, edx записывает результат вычислений в регистр eax.
   * Инструкция call iprintLF вызывает подпрограмму для вывода результата на экран.

## 2.4 Задание для самостоятельной работы

Написал программу для вычисления выражения $ y = f(x) $. Программа должна выводить выражение для вычисления, запросить ввод значения x, вычислить выражение в зависимости от введенного x и вывести результат. В функцию $ f(x) $ выбрал вариант из таблицы 6.3 в соответствии с номером, полученным в лабораторной работе. Создал исполняемый файл и проверил его работу для значений x1 и x2.

Получил вариант 9: $ 10 + (31x − 5) $ для $ x=3 $ и $ x=1 $.

При $ x=3 $ результат — 98.

При $ x=1 $ результат — 36.

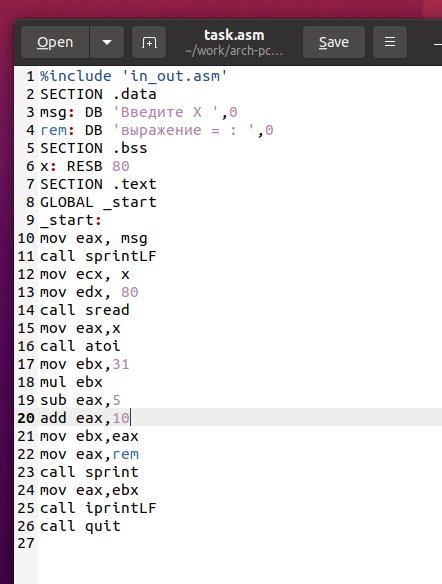


Рис. 16: Программа task.asm

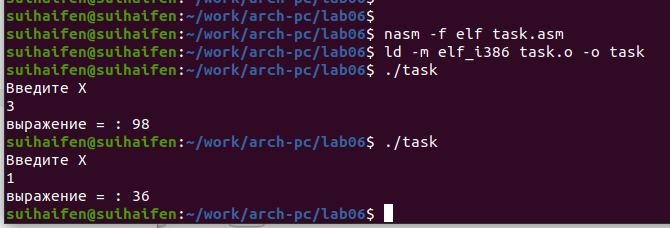


Рис. 17: Запуск программы task.asm

Программа работает корректно.

# 3 Выводы

Изучили работу с арифметическими операциями.