РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук Кафедра прикладной информатики и теории вероятностей

Архипов Олег Константинович

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM .

# 2 Выполнение лабораторной работы

## 2.1 Символьные и численные данные в NASM

Создаю каталог для программ ЛР №6, перехожу в него и создаю там файл lab6-1.asm (рис. [1](#fig:001)).

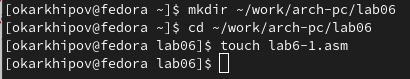


Figure 1: Каталог и файл asm для новой ЛР

Копирую файл in\_out.asm в новую директорию ~/work/arch-pc/lab06 (рис. [2](#fig:002)).

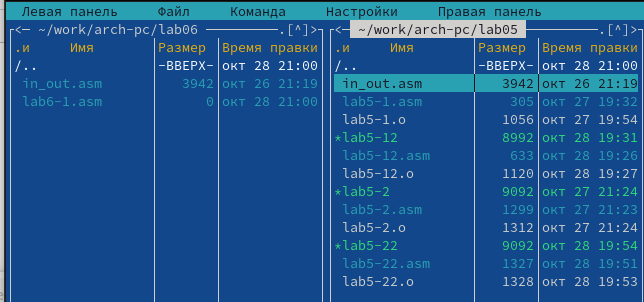


Figure 2: Файл in\_out.asm

Прописываю в файле lab6-1.asm текст программы, которая запишет в регистр eax символ 6, в регистр ebx - символ 4, далее сложит эти два значения и выведет результат (для последнего запишет значение регистра eax в переменную buf1 (в квадратных скобках, т.к. это переменная) и затем в регистр eax - адрес переменной buf1 (адрес - без скобок) (рис. [3](#fig:003)).

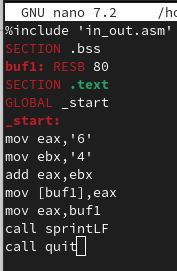


Figure 3: Программа вывода значения регистра eax

Cоздаю и запускаю исполняемый файл и действительно получаю на выходе символ j , т.к. складываются двоичные представления кодов символов 6 и 4 , после чего выводится символ, соответствующий этому суммарному коду (рис. [4](#fig:004)).

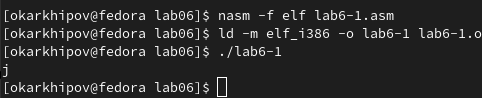


Figure 4: Исполнение lab6-1.asm

Изменяю mov eax,‘6’ и mov ebx,‘4’ на mov eax,6 и mov ebx,4 в исходном коде (рис. [5](#fig:005)).

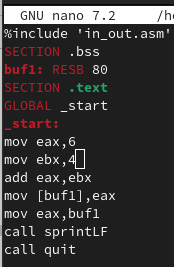


Figure 5: Изменение файла lab6-1.asm

Снова исполняю обновленный файл и поучаю символ с кодом 10 , или LF т.е. символ перевода строки, отображаться он не будет (если не учитывать пропущенную строку) (рис. [6](#fig:006)).

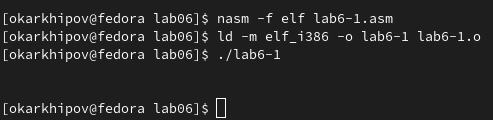


Figure 6: Исполнение и результат обновленного файла lab6-1.asm

Создаю новый файл lab6-2.asm в той же директории (рис. [7](#fig:007)).

Figure 7: Файл lab6-2.asm

Figure 7: Файл lab6-2.asm

Ввожу следующую программу (рис. [8](#fig:008)).

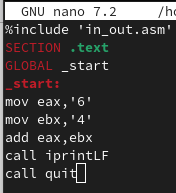


Figure 8: Текст программы в файле lab6-2.asm

Компилирую исполняемый файл для lab6-2.asm и запускаю его, получаю число 106, которое является суммой кодов исходных символов (рис. [9](#fig:009)).

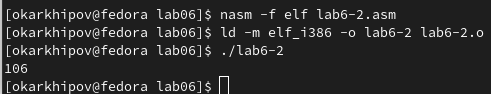


Figure 9: Исполнение файла lab6-2

Заменяю в lab6-2.asm mov eax,‘6’ и mov ebx,‘4’ на mov eax,6 и mov ebx,4 как и в предыдущем файле lab6-1.asm (рис. [10](#fig:010)).

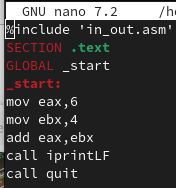


Figure 10: Изменение файла lab6-2.asm

Создаю объектный и исполняемый файлы программы и запускаю последний, получаю число 10 (рис. [11](#fig:011)).

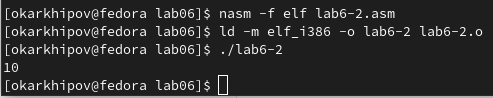


Figure 11: Исполнение измененного файла lab6-2

Заменяю функцию iprintLF на iprint и снова создаю исполняемый файл, после чего запускаю его, получаю то же число 10, без символа перевода строки, что и характеризует отличие функций iprintLF на iprint (рис. [12](#fig:012)-[13](#fig:013)).

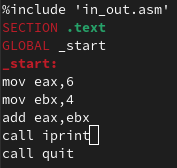


Figure 12: Замена iprintLF на iprint в файле lab6-2.asm

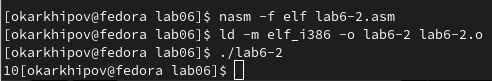


Figure 13: Запуск файла lab6-2 без символа перевода строки

## 2.2 Арифметические операции в NASM

Создаю файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. [14](#fig:014)).

Figure 14: Файл lab6-3.asm

Figure 14: Файл lab6-3.asm

Ввожу текст программы в файл lab6-3.asm (рис. [15](#fig:015)).

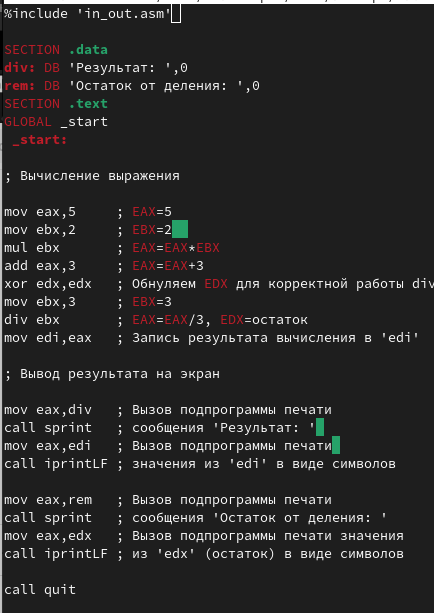


Figure 15: Текст файла lab6-3.asm

Создаю файлы: объектный lab6-3.o и исполняемый lab6-3 и запускаю программу, получаю 4 и 1 в остатке, как и должно быть (рис. [16](#fig:016)).

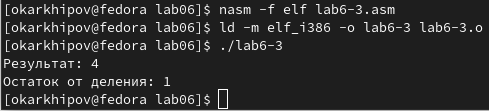


Figure 16: Результат работы файла lab6-3

Изменяю текст программы для вычисления выражения

(можно сравнить рис. [17](#fig:017) и рис. [15](#fig:015)).

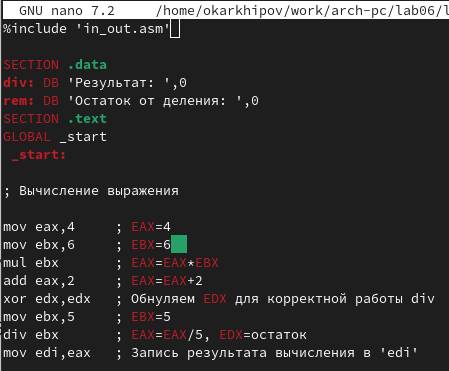


Figure 17: Измененная часть файла lab6-3.asm

Проверяю результат работы программы после внесенных изменений (рис. [18](#fig:018)).

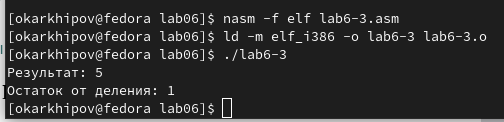


Figure 18: Результат работы измененного файла lab6-3

Для следующего задания по вычислению варианта задания по номеру студенческого билета создаю файл variant.asm в каталоге ~/work/arch-pc/lab06 (рис. [19](#fig:019)).

Figure 19: Файл variant.asm

Figure 19: Файл variant.asm

Ввожу текст программы, которая запросит номер студ. билета, затем вычислит вариант по формуле , т.е. остаток от деления № билета на 20 и ещё плюс 1, после чего выведет ответ на экран (рис. [20](#fig:020)).

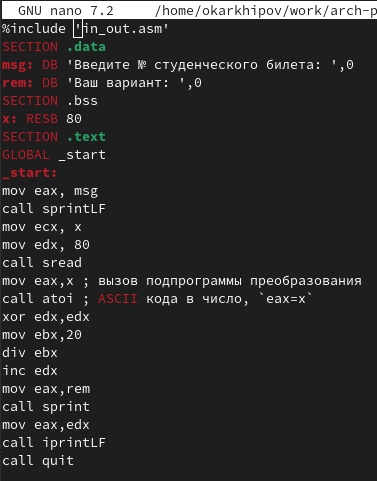


Figure 20: Текст в variant.asm

Создаю исполняемый файл и запускаю его (рис. [21](#fig:021)), ввожу номер билета (рис. [22](#fig:022)) и получаю в ответе 4, что верно (рис. [23](#fig:023)).

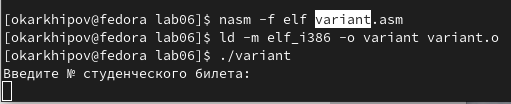


Figure 21: Запуск variant

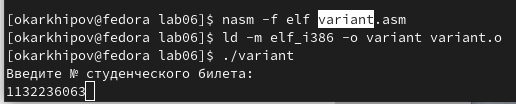


Figure 22: Ввод номера студ. билета

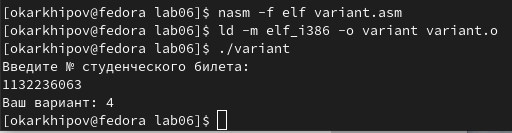


Figure 23: Мой вариант

1. В этой программе за вывод на экран сообщения ‘Ваш вариант:’ отвечают строки:

rem: DB 'Ваш вариант: ',0  
.  
.  
.  
mov eax,rem  
call sprint

1. Строки:

mov ecx, x  
mov edx, 80  
call sread

отвечают за помещение адреса x , где x - вводимая строка (в данном случае номер билета) в регистр ecx , а в регистр edx - максимальную длину этой строки, call sread - вызов подпрограммы внешнего файла in\_out.asm , отвечающей за ввод сообщения с клавиатуры.

1. Инструкция “call atoi” используется для вызова функции преобразования ASCII - кода символа в целое число, после чего записывает результат в регистр eax .
2. За вычисление варианта отвечают команды: “xor edx,edx” - обнуление регистра edx , “mov ebx,20” - присвоение регистру ebx значения 20, “div ebx” - деление на 20 (значение регистра ebx) значения соответствующего регистра eax , в edx - остаток, “inc edx” - увеличение остатка, записанного в регистр edx на 1.
3. Как уже упоминалось выше остаток от деления при выполнении инструкции “div ebx” записывается в регистр edx .
4. Инструкция “inc edx” нужна для увеличения значения в регистре edx на 1.
5. За вывод на экран результата вычислений отвечают команды “mov eax,edx” и “call iprintLF”.

# 3 Самостоятельная работа

Мой номер варианта 4, значит беру функцию вида

где .

Создаю в директории ~/work/arch-pc/lab06 файл solution.asm (рис. [24](#fig:024)).

Figure 24: Файл для вычисления самостоятельного задания

Figure 24: Файл для вычисления самостоятельного задания

Далее прописываю код. Он достаточно подробно описан на рис. [25](#fig:025).

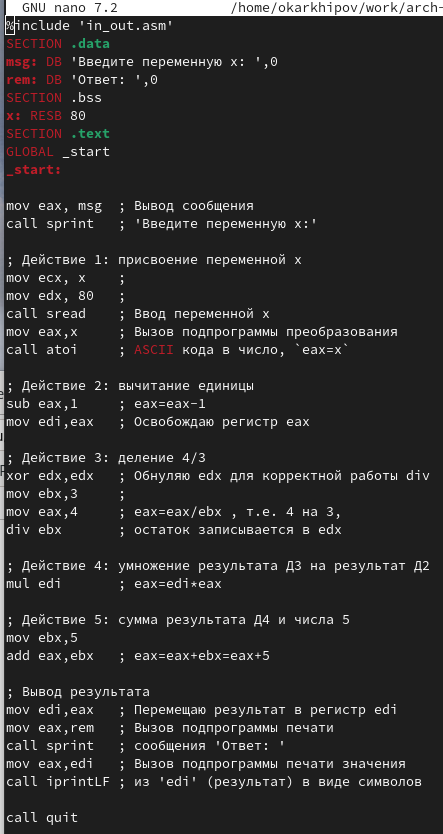


Figure 25: Код программы

Создаю исполняемый файл и запускаю его, в качестве переменной ввожу 4, получаю ответ 8. Это результат без учета остатка в операции деления (рис. [26](#fig:026)).

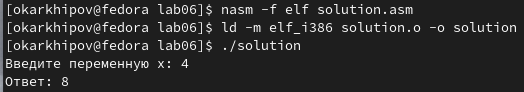


Figure 26: Результат работы программы

Запускаю программу еще раз с переменной 10, чтобы удостовериться в правильности работы программы, результат верный (рис. [27](#fig:027)).

Figure 27: Результат работы программы с альтернативным значением

Figure 27: Результат работы программы с альтернативным значением

# 4 Выводы

Были освоены особенности арифметических операций в ассемблере NASM.

# Список литературы