Отчёт по лабораторной работе N6

Арифметические операции в NASM

Соловьев Богдан Михайлович

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM

# 2 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес опе- ранда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. Далее рассмотрены все существующие способы задания адреса хранения операндов – способы адресации. 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. Например, определим переменную intg DD 3 – это означает, что задается область памяти размером 4 байта, адрес которой обозначен меткой intg. В таком случае, команда mov eax,[intg] копирует из памяти по адресу intg данные в регистр eax. В свою очередь команда mov [intg],eax запишет в память по адресу intg данные из регистра eax. Также рассмотрим команду mov eax,intg В этом случае в регистр eax запишется адрес intg. Допустим, для intg выделена память начиная с ячейки с адресом 0x600144, тогда команда mov eax,intg аналогична команде mov eax,0x600144 – т.е. эта команда запишет в регистр eax число 0x600144.

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Расширенная таблица ASCII состоит из двух частей. Первая (символы с кодами 0-127) является универсальной (см. Приложение.), а вторая (коды 128-255) предназначена для специальных символов и букв национальных алфавитов и на компьютерах разных типов может меняться. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). По- этому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные дан- ные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно. Для выполнения лабораторных работ в файле in\_out.asm реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Это: • iprint – вывод на экран чисел в формате ASCII, перед вызовом iprint в регистр eax необходимо записать выводимое число (mov eax,). • iprintLF – работает аналогично iprint, но при выводе на экран после числа добавляет к символ перевода строки. • atoi – функция преобразует ascii-код символа в целое число и записает результат в регистр eax, перед вызовом atoi в регистр eax необходимо записать число (mov eax,).

# 3 Выполнение лабораторной работы

## 3.1 Выполнение заданий лабораторной работы

Создаю новую директорию и новый файл lab6-1.asm (рис. [1](#fig:001)).

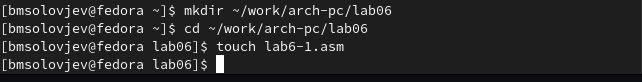


Figure 1: Создание файла и директории

Записываю в файл lab6-1.asm код программы (рис. [2](#fig:002)).

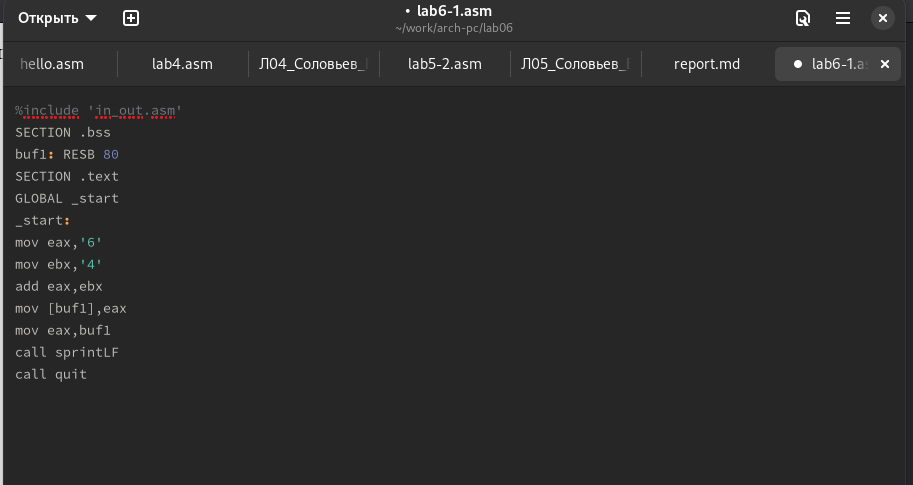


Figure 2: Код в файле lab6-1.asm

Меняю в коде “4” и “6” на 4 и 6. Создаю и запускаю исполяемый файл (рис. [3](#fig:003)).

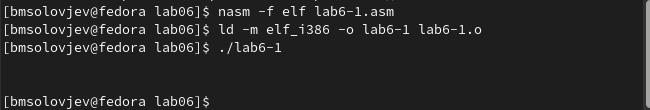


Figure 3: Программа lab6-1.asm

Создаю новый файл (рис. [4](#fig:004)).

Figure 4: Создание нового файла

Figure 4: Создание нового файла

В файл lab6-2.asm копирую другой текст программы и запуская исполняемый файл (рис. [5](#fig:005)).

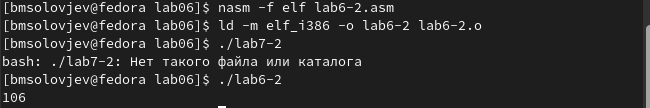


Figure 5: Программа lab6-2.asm с ковычками

Меняю в коде программы lab6-2.asm “6” и “4” на 6 и 4 (рис. [6](#fig:006)).

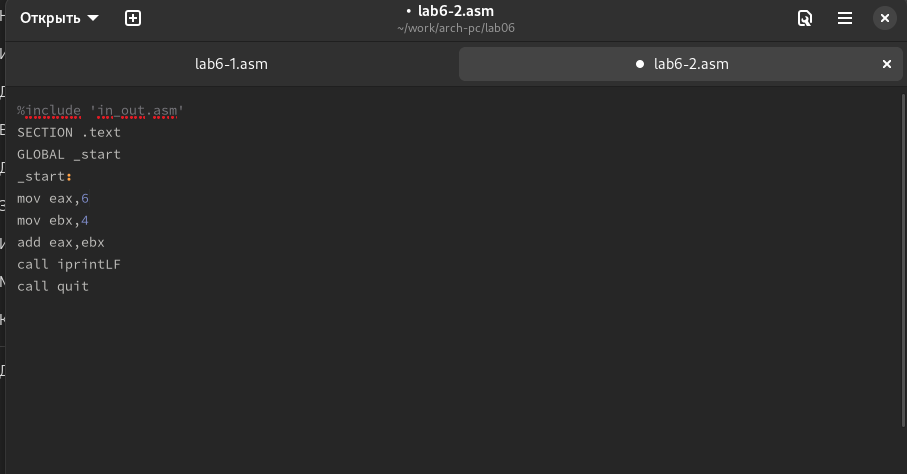


Figure 6:

Запускаю изменённый код программы (рис. [7](#fig:007)).

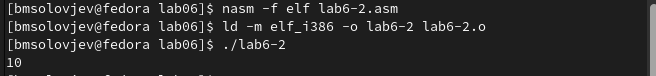


Figure 7: Программа lab6-2.asm без ковычек

Создаю новый файл lab6-3.asm (рис. [8](#fig:010)).

Figure 8: Создание файла lab6-3.asm

Figure 8: Создание файла lab6-3.asm

В этот файл я вставляю текст другой программы и запускаю программу(рис. [9](#fig:011)).

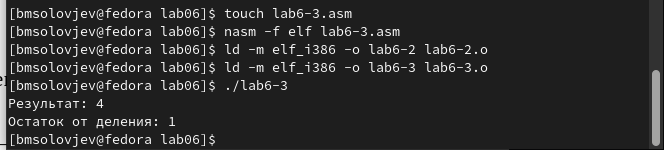


Figure 9: Прогрпмма lab6-3

Меняю код программы так, чтобы она вычисляла значение функции f(x) = (4 \* 6 + 2)/5 (рис. [10](#fig:012)).

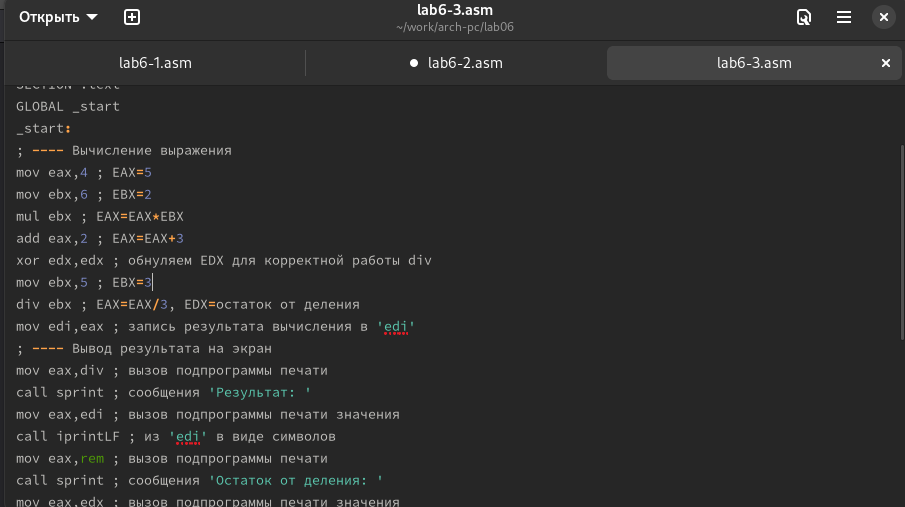


Figure 10: Новый код программы lab6-3

Запускаю иcполняемый файл программы lab6-3 (рис. [11](#fig:013)).

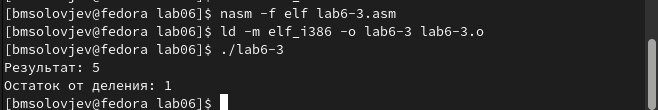


Figure 11: Изменённая программа lab6-3

## 3.2 Отаеты на дополнительные вопросы

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? За вывод этих строк отвечают строки кода:

mov eax,rem  
call sprint

1. Для чего используется следующие инструкции?

mov ecx, x ; перемещаем адрес строки x в регистр ecx  
mov edx, 80 ; запись в регистр edx длины вводимой строки  
call sread ; вызов подпрограммы, которая считывает сооб

1. Для чего используется инструкция “call atoi”? Эта инструкция преобразует ASCII код строки в целое число и записывает результат в регистр eax
2. Какие строки листинга 6.4 отвечают за вычисления варианта? За листинг отвечают следующие 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. 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”? Эта инструкция увеличивает значение регистра edx на 1
3. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений? За вывод на экран результатов вычислений отвечают строки:

mov eax,edx  
call iprintLF

## 3.3 Задачи для самостоятельного решения

Создаю файл nomervar.asm, где ввожу код для программы, которая будет выводить номер варианта (рис. [12](#fig:014)).

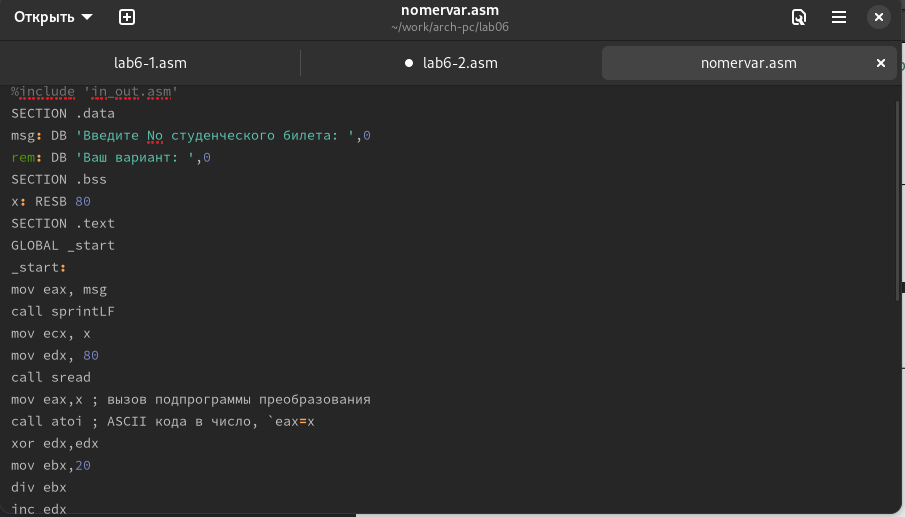


Figure 12: Программа, считающая номер варианта

Запускаю программу (рис. [13](#fig:015)).

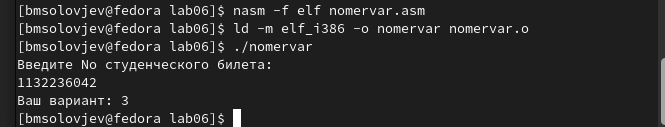


Figure 13: Мой вариант

Код моей программы (рис. [14](#fig:016)).

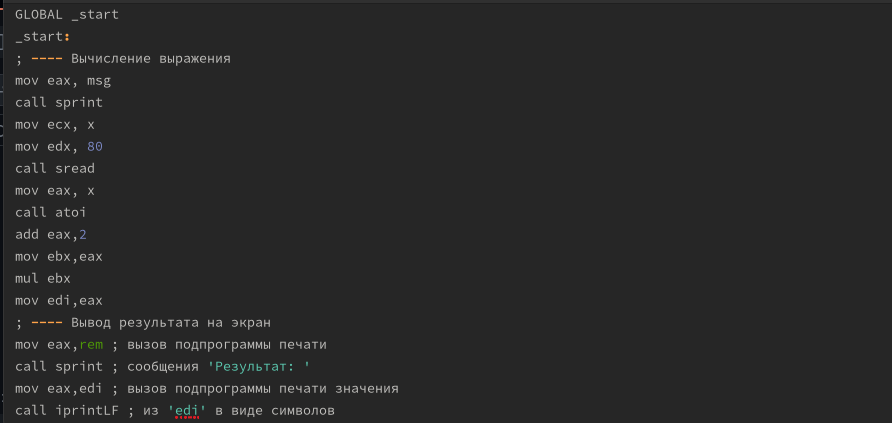


Figure 14: Код моей программы

Проверка моей программы (рис. [15](#fig:017)).

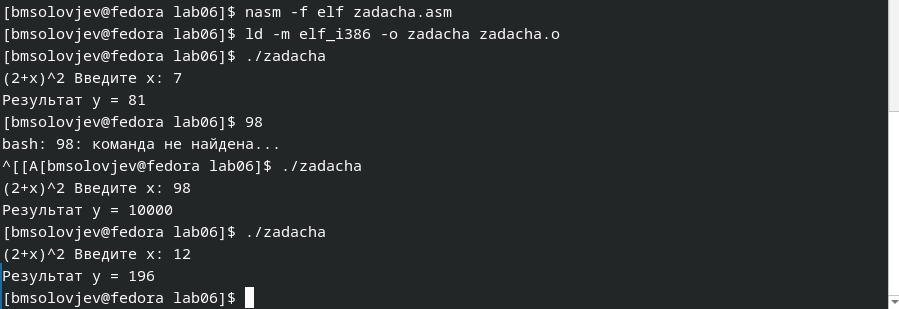


Figure 15: Проверка моей программы

# 4 Выводы

При выполнении данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM.

# Список литературы