Отчет по лабораторной работе №6

Архитектура компьютера

Иванов Александр Олегович

Содержание

# 1 Цель работы

Цель данной лабораторной работы - освоение арифметческих инструкций языка ассемблера NASM.

# 2 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти.

Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: mov ax,bx.  
Непосредственная адресация – значение операнда задается непосредственно в команде, Например: mov ax,2.  
Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

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. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, 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 данные 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.

# 3 Выполнение лабораторной работы

Создание рабочего каталога и рабочего файла для выполнение 6 лабораторной работы (рис. [1](#fig:001)).

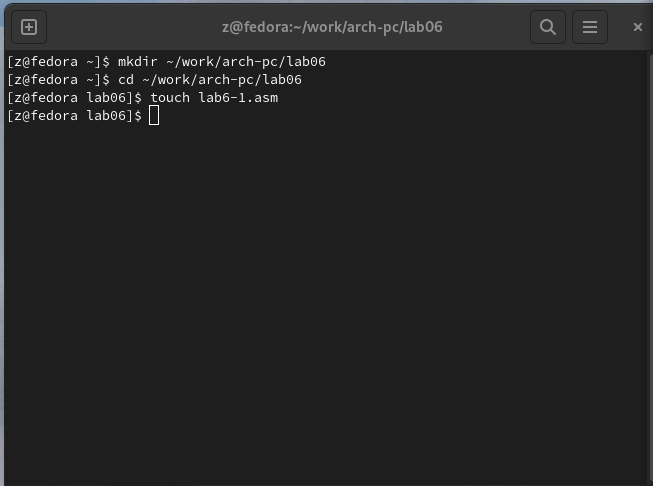


Figure 1: Использование команды mkdir и touch

Создание программы вывода значения регистра eax(рис. [2](#fig:002)).



Figure 2: Использование Midnight Commander для написания кода

Создание исполняемого файла и его запуск(рис. [3](#fig:003)).

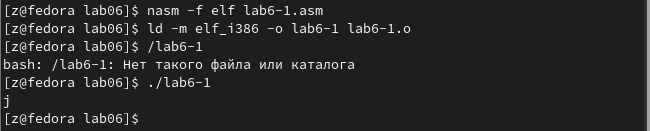


Figure 3: Запуск файла и его результат

Замена символов на регистры чисел(рис. [4](#fig:004)).

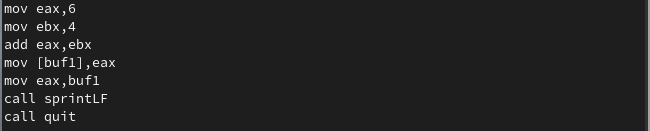


Figure 4: Редактирование изначального файла

Создание файла новой программы и его запуск(рис. [5](#fig:005)).

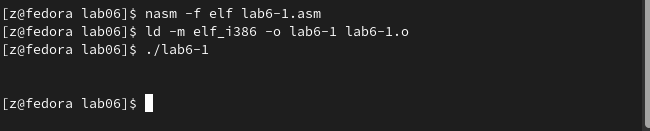


Figure 5: Вывод программы

Создание исполняемого файла lab6-2.asm (рис. [6](#fig:006)).

Figure 6: Использование команды touch

Figure 6: Использование команды touch

Создание программы вывода значения регистра eax с использовнием in out.asm и ее запуск(рис. [7](#fig:007)).

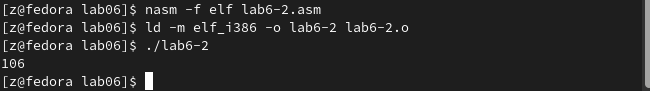


Figure 7: Вывод программы

Редактировние программы вывода значения регистра eax с использовнием in out.asm(рис. [8](#fig:008)).



Figure 8: окно Midnight Commdnder

Создание исполняемого файла и его запуск(рис. [9](#fig:009)).

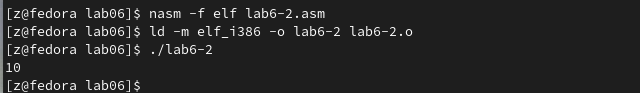


Figure 9: Вывод программы

Создание исполняемого файла lab6-3.asm (рис. [10](#fig:010)).

Figure 10: Использование команды touch

Figure 10: Использование команды touch

Создание программы вычисления выражения (5 \* 2 + 3)/3 (рис. [11](#fig:011)).

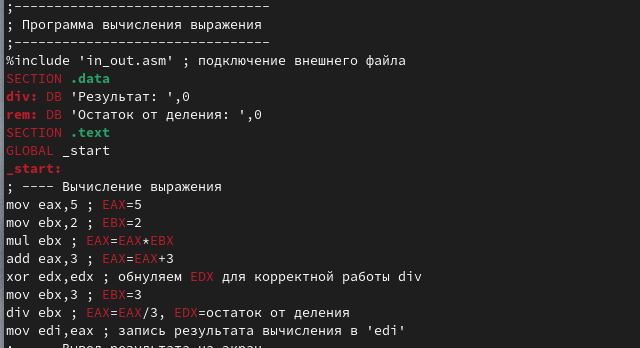


Figure 11: окно Midnight Commdnder

Запуск программы вычисления выражения (5 \* 2 + 3)/3 (рис. [12](#fig:012)).

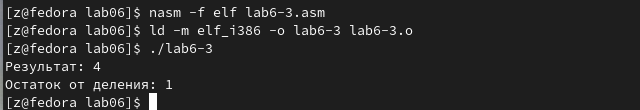


Figure 12: выввод результата программы

Создание программы вычисления выражения (4 \* 6 + 2)/5 (рис. [13](#fig:013)).

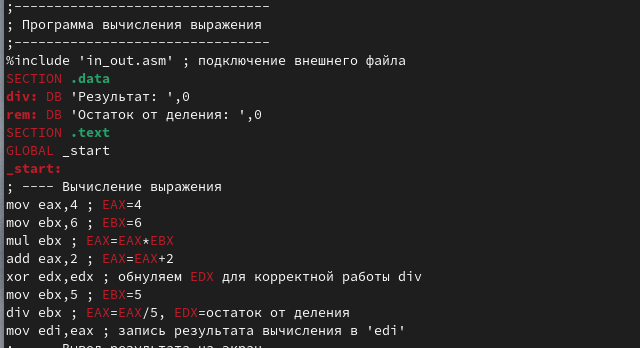


Figure 13: окно Midnight Commdnder

Запуск программы вычисления выражения (4 \* 6 + 2)/5 (рис. [14](#fig:014)).

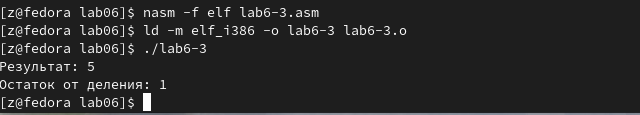


Figure 14: выввод результата программы

Создание исполняемого файла variant.asm (рис. [15](#fig:015)).

Figure 15: Использование команды touch

Figure 15: Использование команды touch

Создание программы вычисления варианта задания по номеру студенческого билета и ее запуск(рис. [16](#fig:016)).

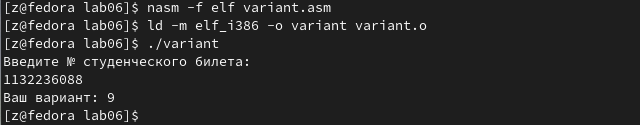


Figure 16: Использование Midnight Commander для написания кода и вывод результата команды

# 4 Ответы на вопросы

1)За вывод сообщения “Ваш вариант” отвечают строки кода: mov eax,rem call sprint

2)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

3)call atoi используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр eax

4)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

5)При выполнении инструкции div ebx остаток от деления записывается в регистр edx

6)Инструкция inc edx увеличивает значение регистра edx на 1

7)За вывод на экран результатов вычислений отвечают строки: mov eax,edx call iprintLF

# 5 Самостоятельная работа

Создание исполняемого файла ans.asm (рис. [17](#fig:017)).

Figure 17: Использование команды touch

Figure 17: Использование команды touch

Создание программы вычисления выражения (x \* 31 - 5) + 10 (рис. [18](#fig:018)).

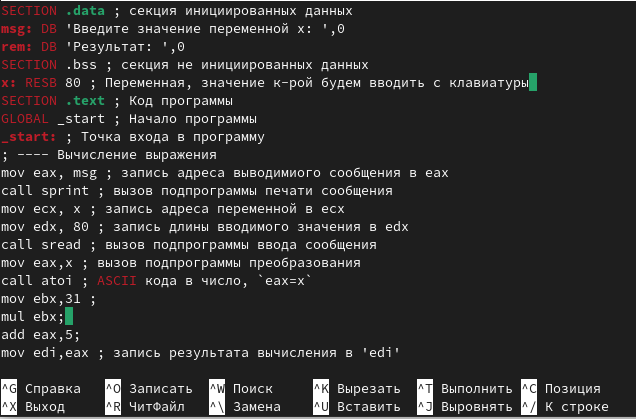


Figure 18: окно Midnight Commdnder

Запуск программы вычисления выражения (x \* 31 - 5) + 10 (рис. [19](#fig:019)).

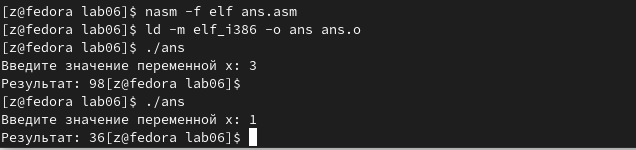


Figure 19: выввод результата программы

# 6 Выводы

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

# Список литературы