Отчет по лабораторной работе №6

Дисциплина архитектура компьютера

Ахатов Эмиль Эрнстович

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

# 3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обра- ботке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация – операнды хранятся в регистрах и в команде использу- ются 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 символь- ном виде. Кодирование этой информации производится согласно кодовой табли- це символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Соглас- но стандарту ASCII каждый символ кодируется одним байтом. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, на- пример, 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 ситу- ация 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

# 4 Выполнение лабораторной работы

## 4.1 Символьные и численные данные в NASM

С помощью утилиты mkdir создаю директорию, в которой буду создавать файлы с программами,перехожу в директорию.

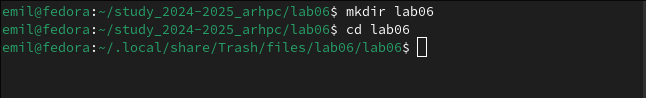


Рис. 1: Создание директории

С помощью утилиты touch создаю файл lab6-1.asm

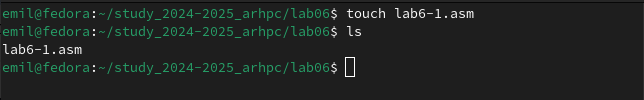


Рис. 2: Создание файла

Открываю созданный файл lab6-1.asm, вставляю в него программу вывода значения регистра eax



Рис. 3: Ввод программы

Создаю исполняемый файл программы и запускаю его. Вывод программы: символ j, потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6.

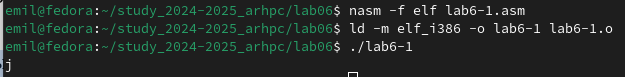


Рис. 4: Запуск исполняемого файла

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4



Рис. 5: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его. Теперь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран.

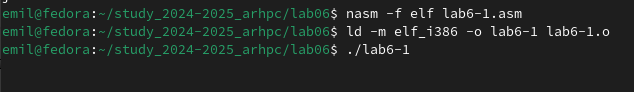


Рис. 6: Запуск исполняемого файла

Создаю новый файл lab6-2.asm с помощью утилиты touch.

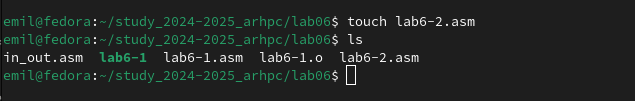


Рис. 7: Создание файла

Ввожу в файл текст другой программы для вывода значения регистра eax

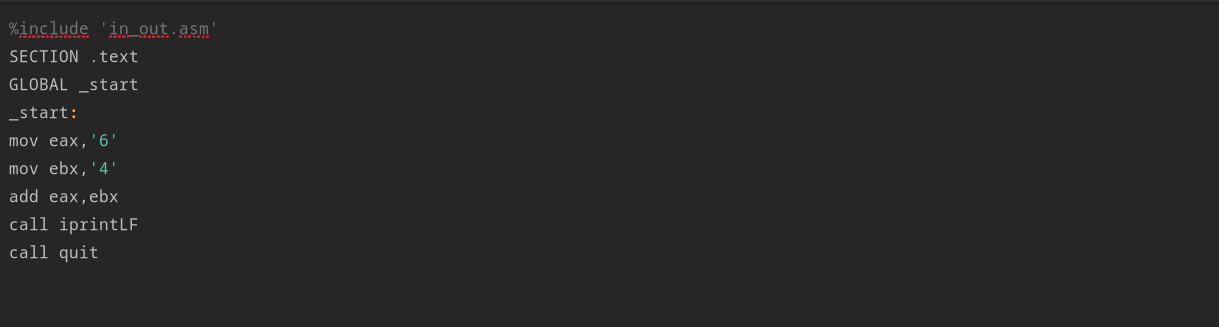


Рис. 8: Редактирование файла

Создаю и запускаю исполняемый файл lab6-2. Теперь вывод числа 106,программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4”.

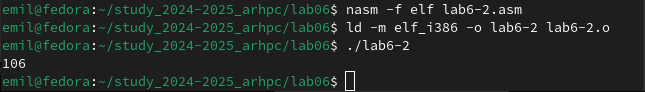


Рис. 9: Запуск исполняемого файла

Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4

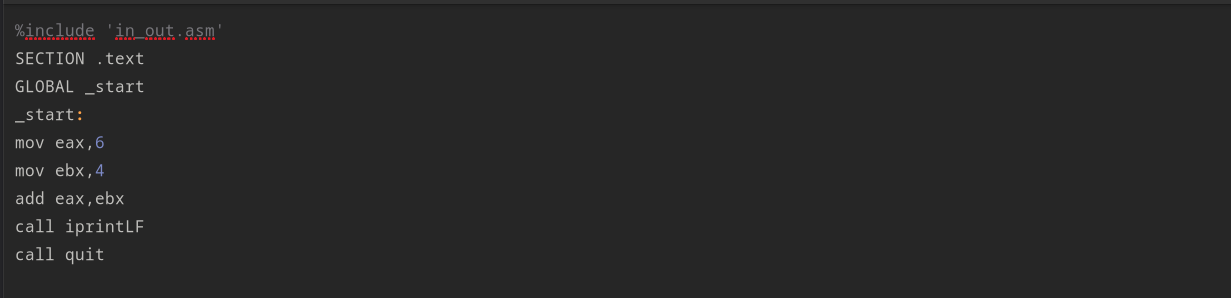


Рис. 10: Редактирование файла

Создаю и запускаю новый исполняемый файл. Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 10.

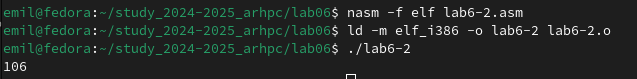


Рис. 11: Запуск исполняемого файла

Заменяю в тексте программы функцию iprintLF на iprint

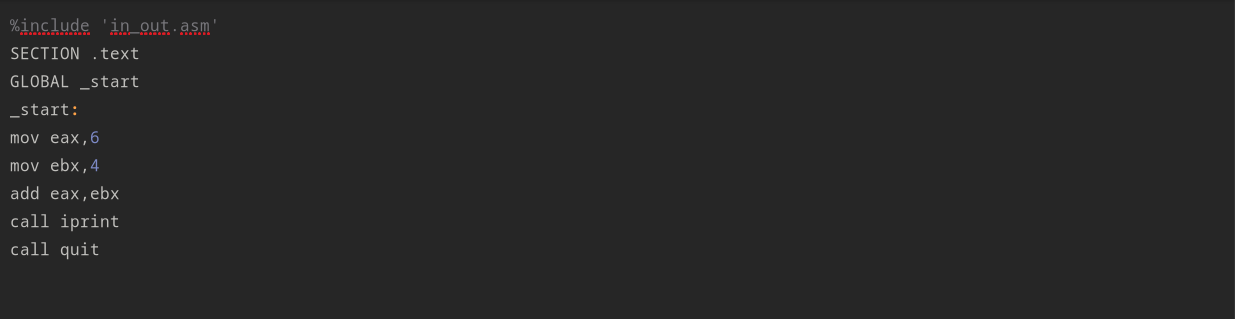


Рис. 12: Редактирование файла

Создаю и запускаю новый исполняемый файл. Вывод не изменился,потому что символ переноса строки не отображался, когда программа исполнялась с функцией iprintLF, а iprint не добавляет к выводу символ переноса строки, в отличие от iprintLF.

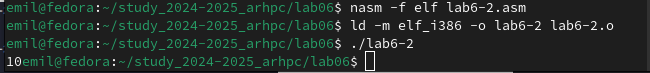


Рис. 13: Запуск исполняемого файла

## 4.2 Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm с помощью утилиты touch

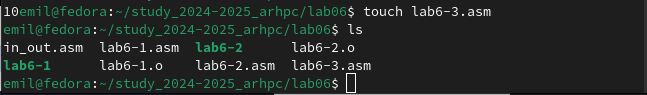


Рис. 14: Создание файла

Ввожу в созданный файл текст программы для вычисления значения выражения f(x) = (5 \* 2 + 3)/3

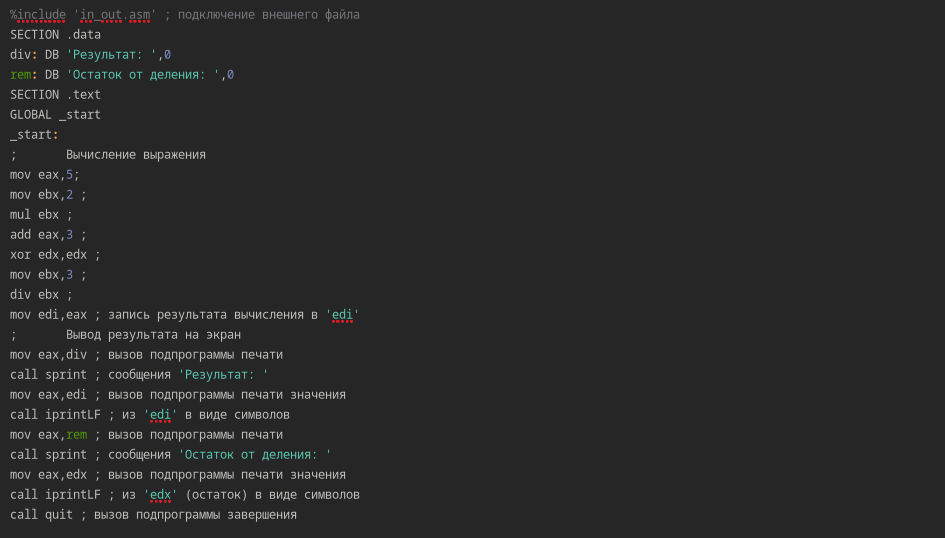


Рис. 15: Редактирование файла

Создаю исполняемый файл и запускаю его

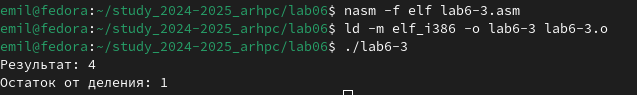


Рис. 16: Запуск исполняемого файла

Изменяю программу так, чтобы она вычисляла значение выражения f(x) = (4 \* 6 + 2)/5

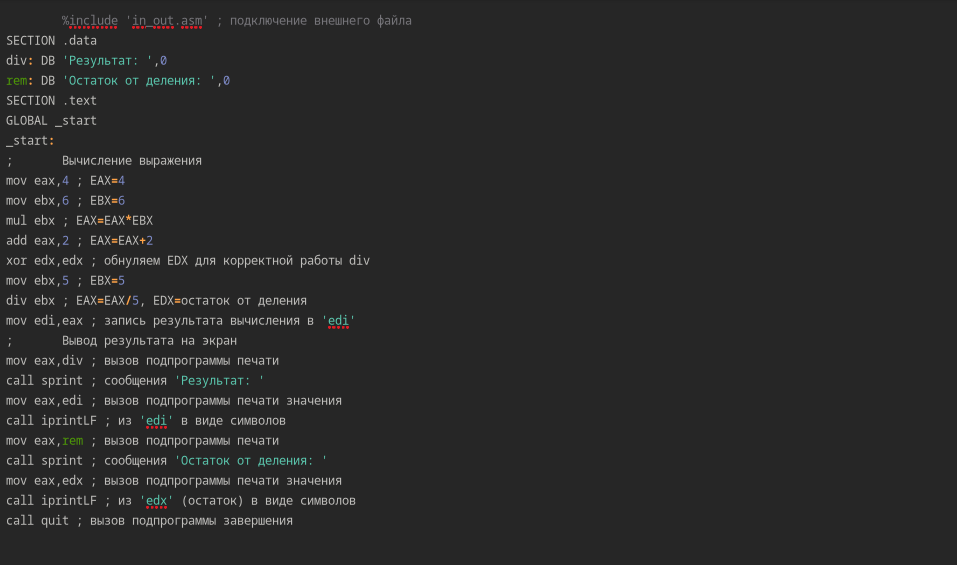


Рис. 17: Редактирование файла

Создаю и запускаю новый исполняемый файл,программа выполняется верно

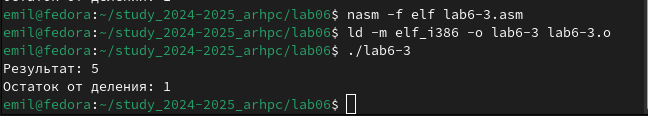


Рис. 18: Запуск исполняемого файла

Создаю файл variant.asm с помощью утилиты touch.

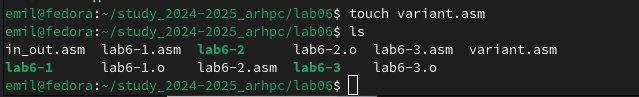


Рис. 19: Создание файла

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета

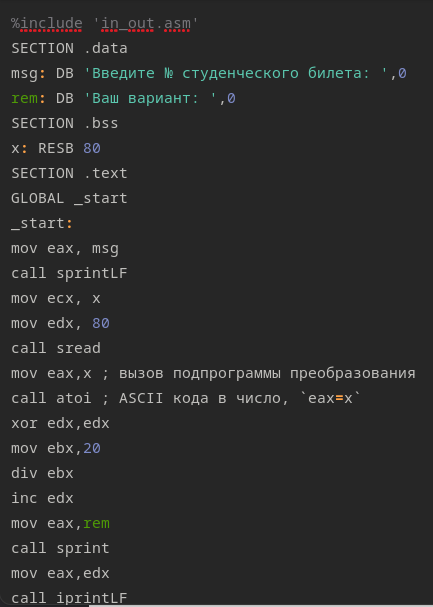


Рис. 20: Редактирование файла

Создаю и запускаю исполняемый файл. Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 12

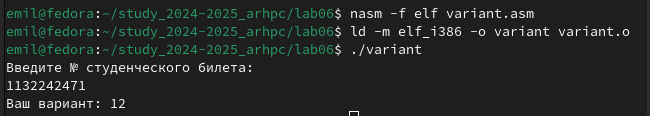


Рис. 21: Запуск исполняемого файла

## 4.3 Ответы на вопросы по программе

1. За вывод сообщения “Ваш вариант” отвечают строки кода:

mov eax,rem  
call sprint

1. 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
2. call atoi используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр eax
3. 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

1. При выполнении инструкции div ebx остаток от деления записывается в регистр edx
2. Инструкция inc edx увеличивает значение регистра edx на 1
3. За вывод на экран результатов вычислений отвечают строки:

mov eax,edx  
call iprintLF

#Выполнение заданий для самостоятельной работы

Создаю файл lab6-4.asm с помощью утилиты touch

Рис. 22: Создание файла

Рис. 22: Создание файла

Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения (8𝑥 − 6)/2

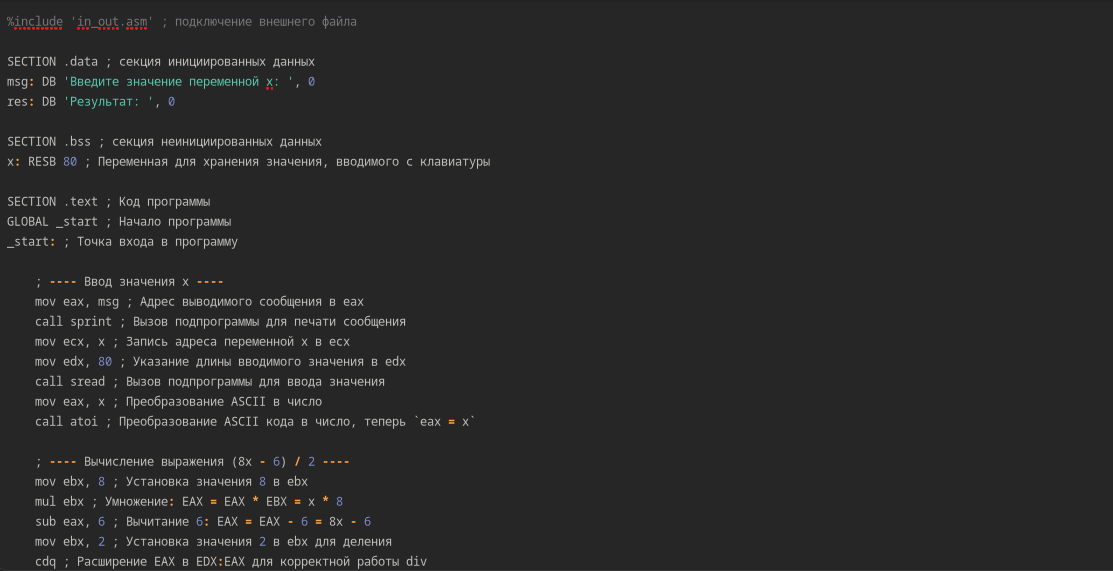


Рис. 23: Редактирование файла

Создаю и запускаю исполняемый файл, при вводе значения 1, вывод 1.При вводе значения 5, вывод 17.Программа отработала верно.

# 5 Выводы

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