Отчет по лабараторной работе №6

Архитектура компьютера

Исаханян Армен Артурович

Содержание

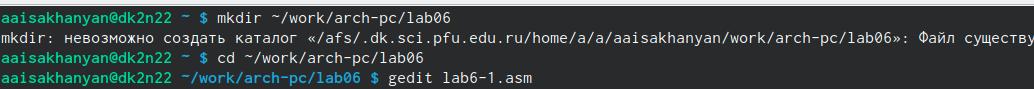
# 1 Цель работы

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 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. # Выполнение лабораторной работы

Создал каталог lab06 перешел в него и создал файл lab6-1.asm и открыл его (рис. ??).



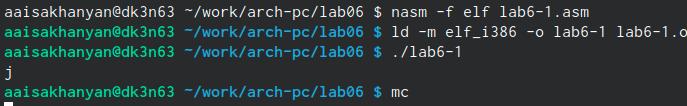
Создание каталога, переход в него, создание файла и его открытие

Ввел программу в файл (рис. ??).



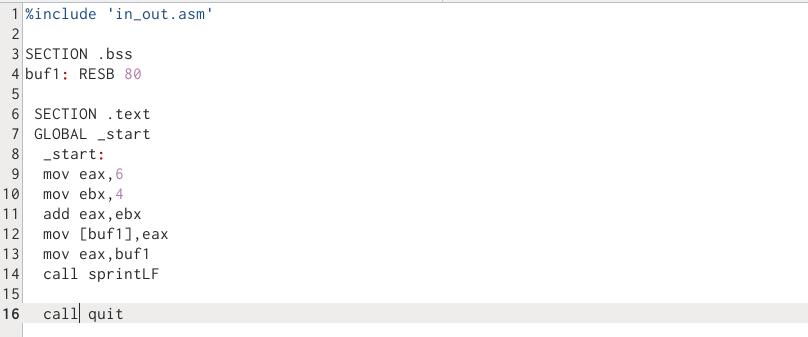
Ввод программы

Скомпилировал исходный файл передал файл компоновщику(рис. ??).



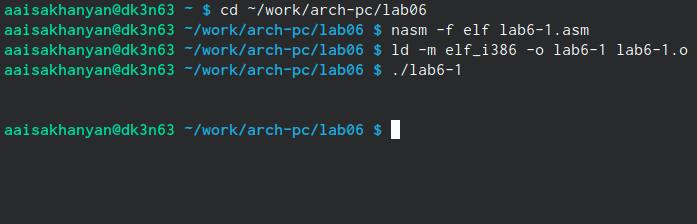
Компиляция исходного файла и текста, передача файла компоновщику

Редактировал программу в файле lab6-1.asm (рис. ??).



Редактрирование программы

Скомпилировал исходный файл передал файл компоновщику (рис. ??).



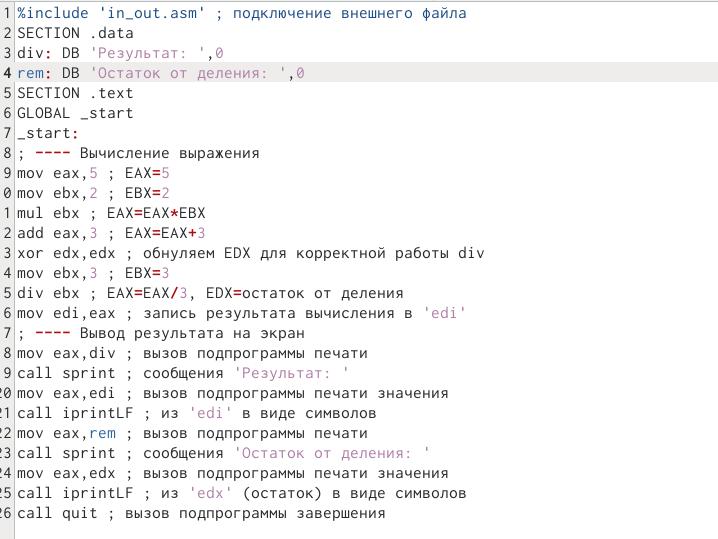
Компиляция файла и передача файла компоновщику

Создание файла lab6-2 в том же каталоге (рис. ??).

Создание файла

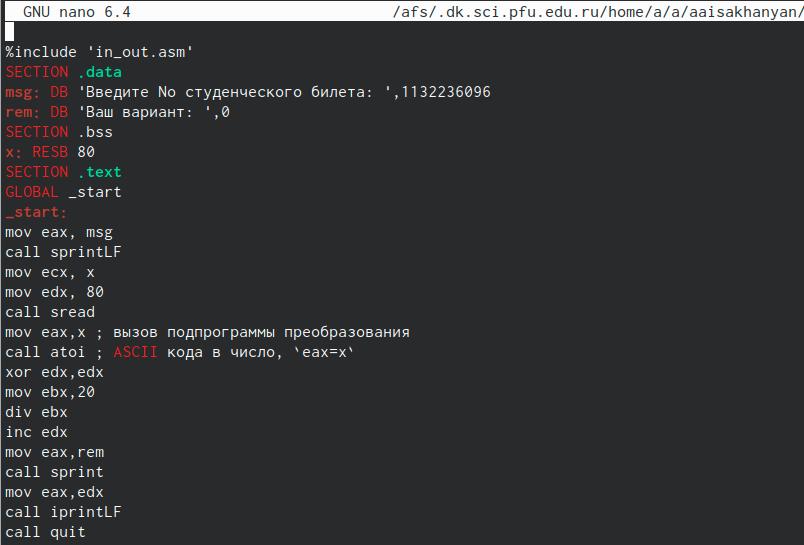
Создание файла

Ввел программу в файл lab6-2.asm (рис. ??).



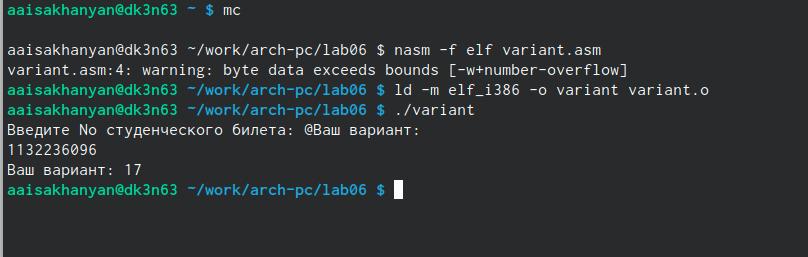
Ввод программы

Создал файл variant.asm в том же каталоге ввел программу, затем ввел номер смоего студенческого билета и узнал свой вариант-17. (рис. ??).



Создание файла, ввод программы, ввод студенческого

Узнал номер своего варианта (рис. ??).



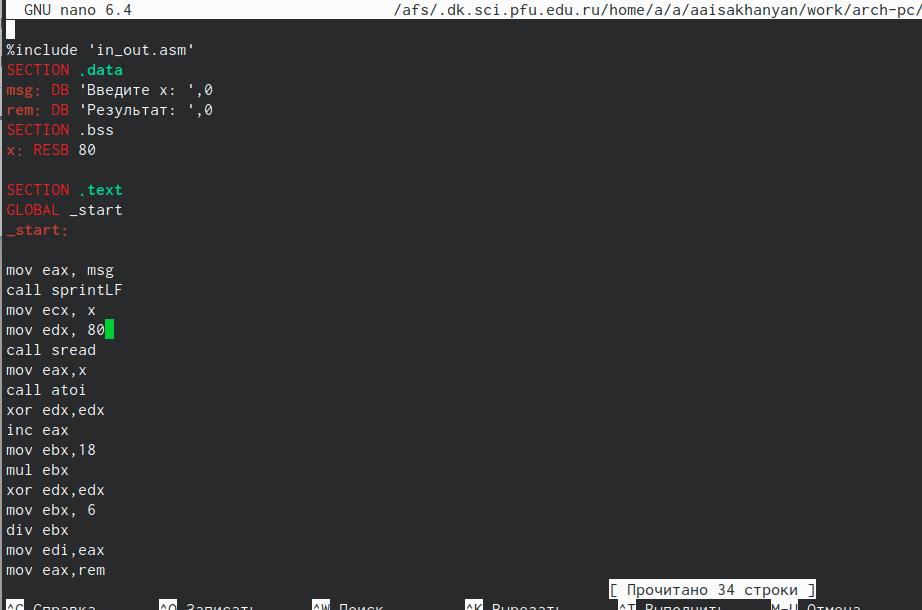
Вариант 17

# 3 Ответы на вопросы по программе

1. За вывод сообщения “Ваш вариант” отвечают строки кода: mov eax,rem call sprint
2. Инструкция mov ecx, x используется, чтобы положить адрес вводимой стро- ки 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, кото- рая 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

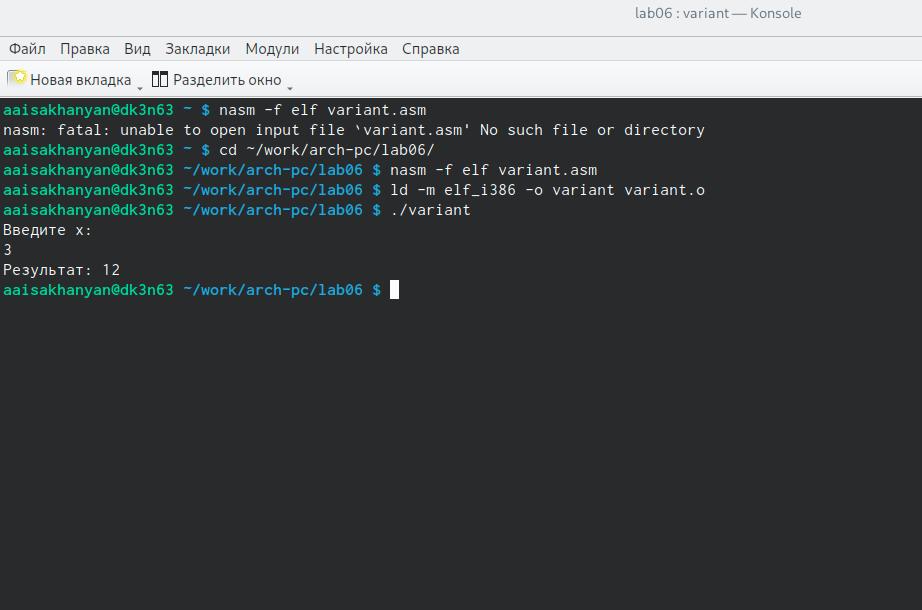
# 4 Выполнение самостоятельной работы

В файле variant.asm очистил предыдущую программу и написал новую программу для выполнения самостоятельной работы(рис. ??).



C/p

Проверил правильность программы(рис. ??).



Проверка

# 5 Выводы

Освоил арифметические инструкции языка ассемблера NASM

# Список литературы