**Отчет по лабораторной работе №6**

**Дисциплина: архитектура компьютера**

Чувакина Мария Владимировна

**Содержание**

**1  Цель работы 5**

**2  Задание 6**

**3  Теоретическое введение 7**

**4  Выполнение лабораторной работы 8**

4.1 Символьные и численные данные в NASM . . . . . . . . . . . . ………… 8

4.2 Выполнение арифметических операций в NASM . . . . . . . . . . ……..14

4.2.1 Ответы на вопросы по программе............................................... 18

4.3 Выполнение заданий для самостоятельной работы. . . …………… 19

**5  Выводы 22**

**6  Список литературы 23**

**Список иллюстраций**

4.1  Создание директории ...................................... ……………………………..8

4.2  Создание файла .............................................................................................. 8

4.3  Создание копии файла................................................................................... 8

4.4  Редактирование файла................................................................................... 9

4.5  Запуск исполняемого файла......................................................................... 9

4.6  Редактирование файла................................................................................. 10

4.7  Запуск исполняемого файла........................................................................ 10

4.8  Создание файла ............................................................................................ 10

4.9  Редактирование файла.................................................................................. 11

4.10 Запуск исполняемого файла........................................................................ 11

4.11 Редактирование файла................................................................................. 12

4.12 Запуск исполняемого файла........................................................................ 12

4.13 Редактирование файла................................................................................. 13

4.14 Запуск исполняемого файла........................................................................ 13

4.15 Создание файла ............................................................................................14

4.16 Редактирование файла................................................................................. 14

4.17 Запуск исполняемого файла........................................................................ 15

4.18 Изменение программы ............................................................................... 15

4.19 Запуск исполняемого файла........................................................................16

4.20 Создание файла ........................................................................................... 16

4.21 Редактирование файла................................................................................. 17

4.22 Запуск исполняемого файла........................................................................ 18

4.23 Создание файла ........................................................................................... 19

4.24 Написание программы ................................................................................ 20

4.25 Запуск исполняемого файла........................................................................ 20

**1 Цель работы**

Цель данной лабораторной работы - освоение арифметческих 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**

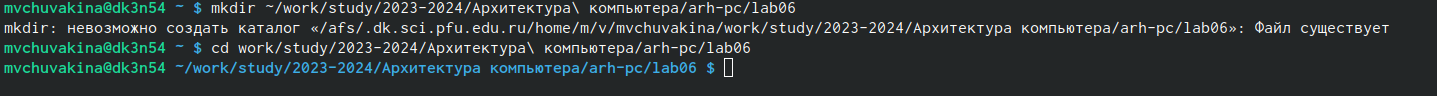
Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация – операнды хранятся в регистрах и в команде используются имена 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

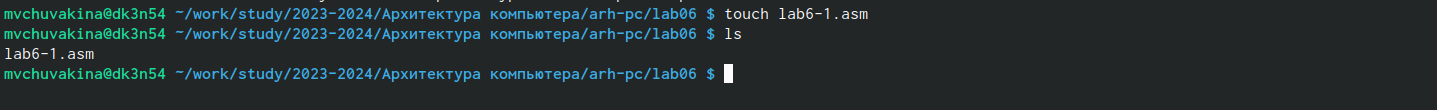
**4 Выполнение лабораторной работы**

**4.1 Символьные и численные данные в NASM**

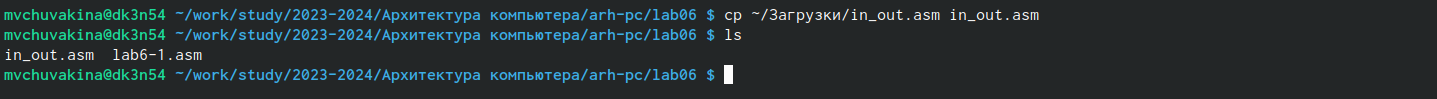
С помощью утилиты mkdir создаю директорию, в которой буду создавать файлы с программами для лабораторной работы №6 (рис. 4.1). Перехожу в созданный каталог с помощью утилиты cd.

 Рис. 4.1: Создание директории

С помощью утилиты touch создаю файл lab6-1.asm (рис. 4.2).

 Рис. 4.2: Создание файла

Копирую в текущий каталог файл in\_out.asm с помощью утилиты cp, т.к. он будет использоваться в других программах (рис. 4.3).

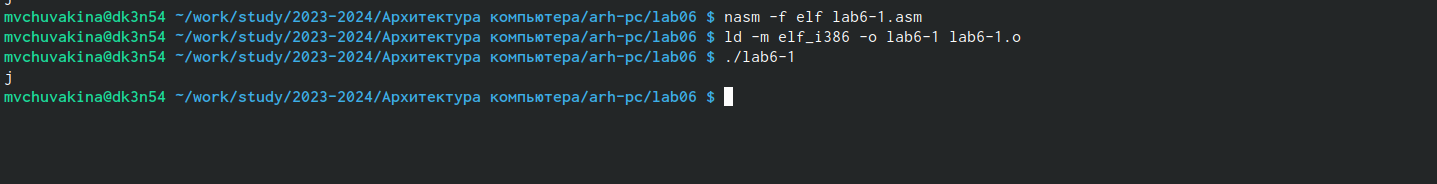
 Рис. 4.3: Создание копии файла

Открываю созданный файл lab6-1.asm, вставляю в него программу вывода значения регистра eax (рис. 4.4).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание Рис. 4.4: Редактирование файла

Создаю исполняемый файл программы и запускаю его (рис. 4.5). Вывод программы: символ j, потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6.

 Рис. 4.5: Запуск исполняемого файла

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4 (рис. 4.6).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рис. 4.6: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его (рис. 4.7). Те- перь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран.

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание Рис. 4.7: Запуск исполняемого файла

Создаю новый файл lab6-2.asm с помощью утилиты touch (рис. 4.8).

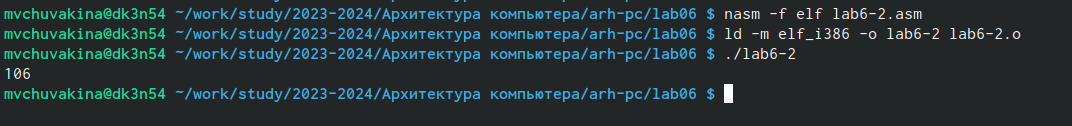
 Рис. 4.8: Создание файла

Ввожу в файл текст другой программы для вывода значения регистра eax (рис. 4.9).

Изображение выглядит как текст, программное обеспечение, снимок экрана, веб-страница

Автоматически созданное описание Рис. 4.9: Редактирование файла

Создаю и запускаю исполняемый файл lab6-2 (рис. 4.10). Теперь вывод число 106, потому что программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4”.

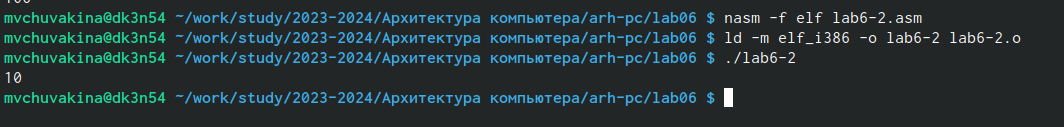
 Рис. 4.10: Запуск исполняемого файла

Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4 (рис. 4.11).

Изображение выглядит как текст, программное обеспечение, веб-страница, Значок на компьютере

Автоматически созданное описание Рис. 4.11: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис. 4.12). Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 10.

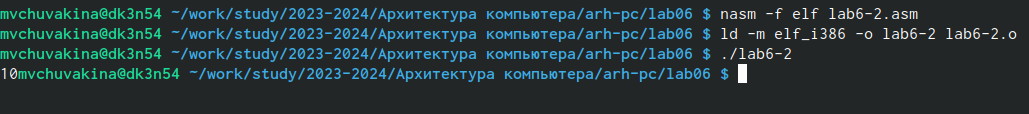
 Рис. 4.12: Запуск исполняемого файла

Заменяю в тексте программы функцию iprintLF на iprint (рис. 4.13).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

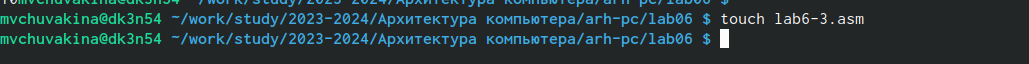
Автоматически созданное описание Рис. 4.13: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис. 4.14). Вывод не изменился, потому что символ переноса строки не отображался, когда программа исполня- лась с функцией iprintLF, а iprint не добавляет к выводу символ переноса строки, в отличие от iprintLF.

 Рис. 4.14: Запуск исполняемого файла

**4.2 Выполнение арифметических операций в NASM**

Создаю файл lab6-3.asm с помощью утилиты touch (рис. 4.15).

 Рис. 4.15: Создание файла

Ввожу в созданный файл текст программы для вычисления значения выражения f(x) = (5 \* 2 + 3)/3 (рис. 4.16).

Изображение выглядит как текст, программное обеспечение, веб-страница, Веб-сайт

Автоматически созданное описание Рис. 4.16: Редактирование файла

Создаю исполняемый файл и запускаю его (рис. 4.17).

Изображение выглядит как текст, Шрифт, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание Рис. 4.17: Запуск исполняемого файла

Изменяю программу так, чтобы она вычисляла значение выражения f(x) = (4 \* 6 + 2)/5 (рис. 4.18).

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

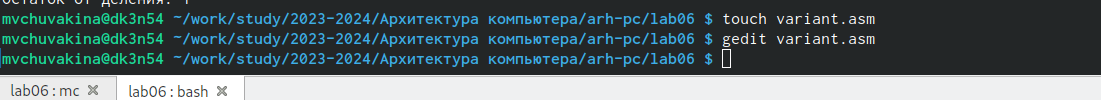
Автоматически созданное описание Рис. 4.18: Изменение программы

Создаю и запускаю новый исполняемый файл (рис. 4.19). Я посчитала для проверки правильности работы программы значение выражения самостоятельно, программа отработала верно.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание Рис. 4.19: Запуск исполняемого файла

Создаю файл variant.asm с помощью утилиты touch (рис. 4.20).

 Рис. 4.20: Создание файла

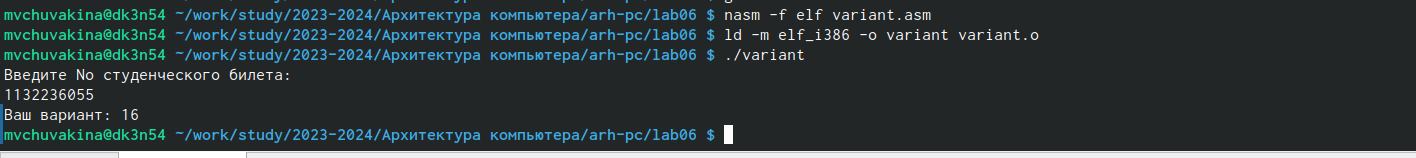
Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис. 4.21).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рис. 4.21: Редактирование файла

Создаю и запускаю исполняемый файл (рис. 4.22). Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 16.

 Рис. 4.22: Запуск исполняемого файла

**4.2.1 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

1. call atoi используется для вызова подпрограммы из внешнего файла, кото- рая преобразует ascii-код символа в целое число и записывает результат в регистр eax
2. За вычисления варианта отвечают строки:

**xor edx**,**edx** ; обнуление edx для корректной работы div с б **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

**4.3 Выполнение заданий для самостоятельной работы**

Создаю файл lab6-4.asm с помощью утилиты touch (рис. 4.23).

 Рис. 4.23: Создание файла

Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения (10\*x-5)^2 (рис. 4.24). Это выражение было под вариантом 16.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание ю Рис. 4.24: Написание программы

Создаю и запускаю исполняемый файл (рис. 4.25). При вводе значения 1, вывод - 20.

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Значок на компьютере

Автоматически созданное описание Рис. 4.25: Запуск исполняемого файла

**Листинг 4.1. Программа для вычисления значения выражения (10\*x-5)^2.**

%include 'in\_out.asm' ; подключение внешнего файла

**SECTION** .data ; секция инициированных данных  
msg: DB 'Введите значение переменной х: ',0  
rem: DB 'Результат: ',0

**SECTION** .bss ; секция не инициированных данных  
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры, выделенный ра

**SECTION** .text ; Код программы  
**GLOBAL** \_start ; Начало программы  
\_start: ; Точка входа в программу  
; ---- Вычисление выражения

**mov eax**, msg ; запись адреса выводимиого сообщения в eax  
**call** sprint ; вызов подпрограммы печати сообщения  
**mov ecx**, x ; запись адреса переменной в ecx  
**mov edx**, 80 ; запись длины вводимого значения в edx  
**call** sread ; вызов подпрограммы ввода сообщения  
**mov eax**,x ; вызов подпрограммы преобразования  
**call** atoi ; ASCII кода в число, `eax=x`  
**add eax**,-5;   
**mov ebx**,(10\*x-5)

**mul ebx**  
**mov edi**,**eax** ; запись результата вычисления в 'edi'  
; ---- Вывод результата на экран

**mov eax**,rem ; вызов подпрограммы печати  
**call** sprint ; сообщения 'Результат: '  
**mov eax**,**edi** ; вызов подпрограммы печати значения  
**call** iprint ; из 'edi' в виде символов  
**call** quit ; вызов подпрограммы завершения

**5 Выводы**

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

**6 Список литературы**

1. Лабораторная работа №6

2. Таблица ASCII