Отчет по лабораторной работе №6

Дисциплина: архитектура компьютера

Швецов Михаил Романович

Содержание

# 1 Цель работы

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

# 2 Задание

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

# 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. [1](#tbl:std-dir) приведено краткое описание стандартных каталогов Unix.

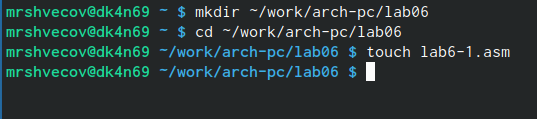
Table 1: Описание некоторых каталогов файловой системы GNU Linux

| Имя каталога | Описание каталога |
| --- | --- |
| / | Корневая директория, содержащая всю файловую |
| /bin | Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям |
| /etc | Общесистемные конфигурационные файлы и файлы конфигурации установленных программ |
| /home | Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя |
| /media | Точки монтирования для сменных носителей |
| /root | Домашняя директория пользователя root |
| /tmp | Временные файлы |
| /usr | Вторичная иерархия для данных пользователя |

Более подробно об Unix см. в [1–6].

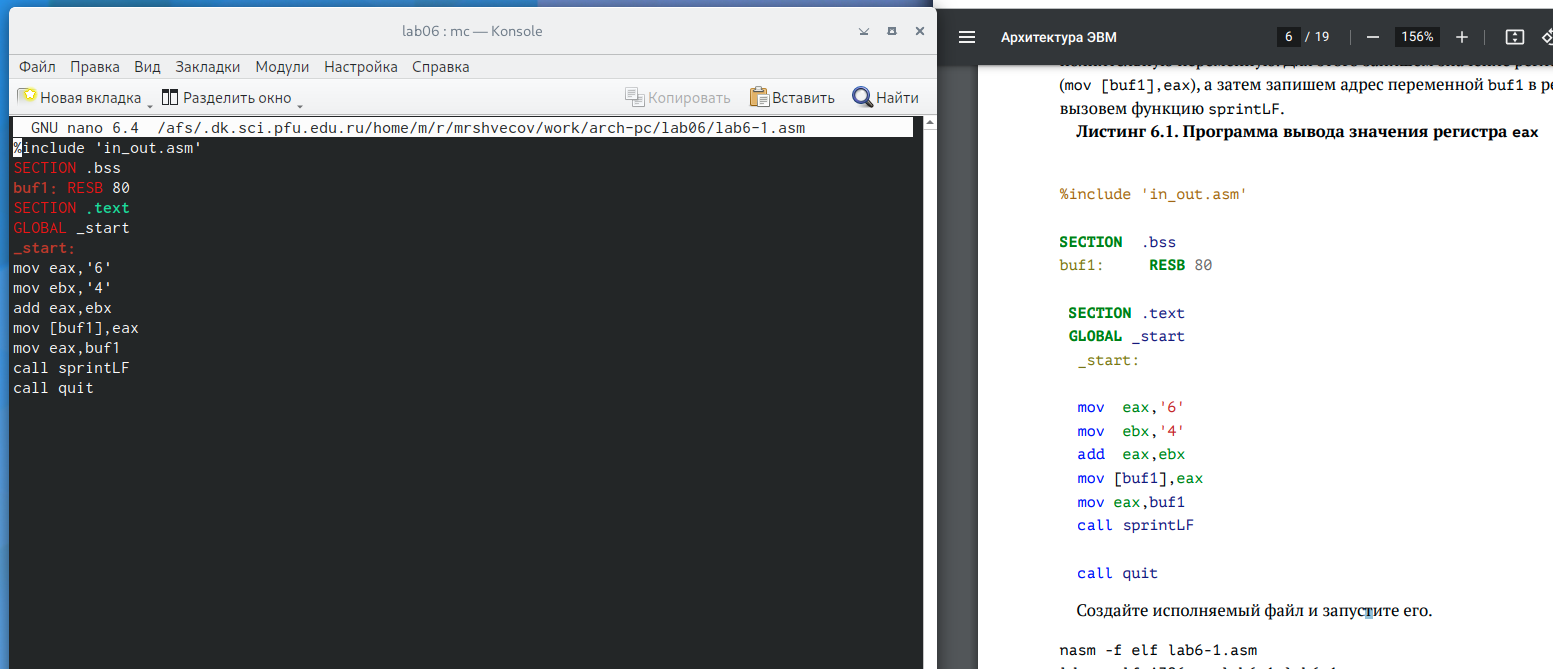
# 4 Выполнение лабораторной работы

Создал директорию, в которой буду создавать файлы с программами для лабораторной работы №6 Перехожу в созданный каталог с помощью команды cd. И создаю файл lab6-1.asm



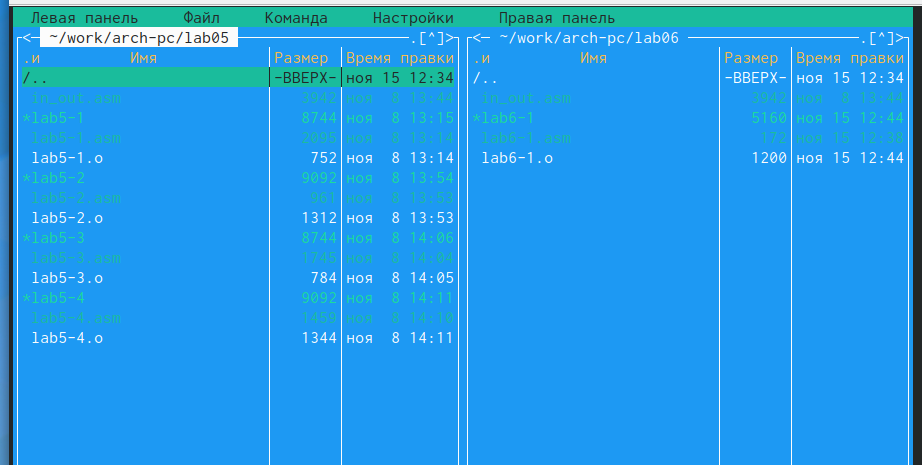
Создание директории и файла

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



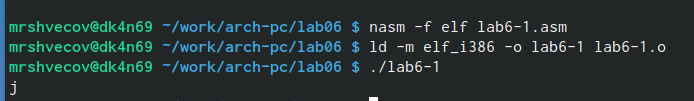
Редактирование файла

Скопировал файл in\_out.asm в директорию lab06



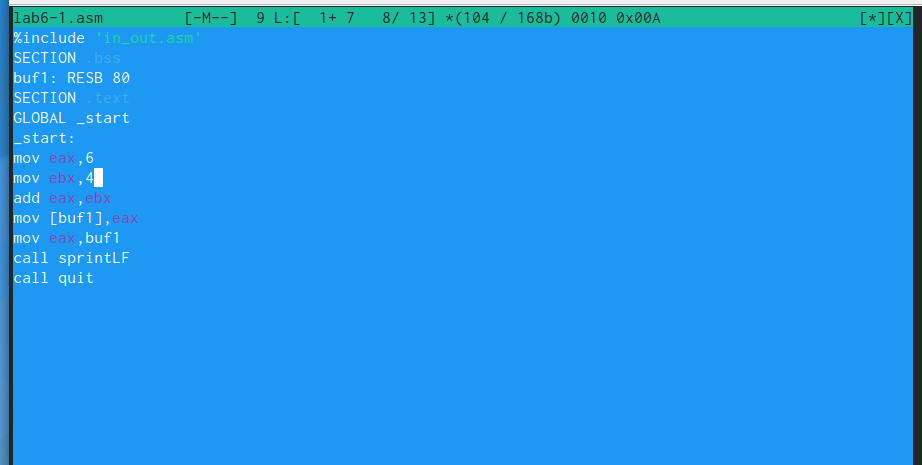
копирование файла in\_out.asm

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.



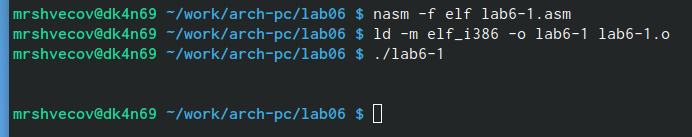
Создание исполняемого файла

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



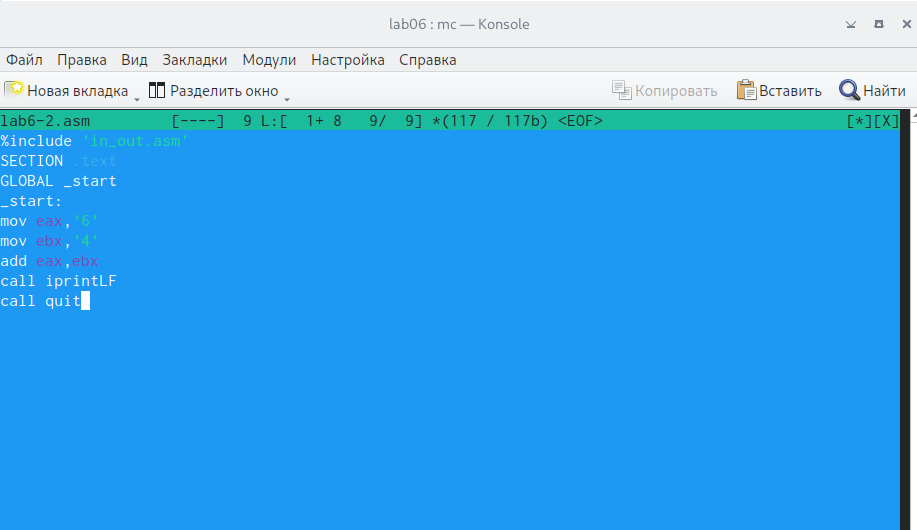
Редактирование файла

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.



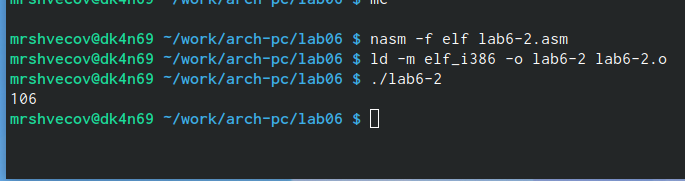
Создание директории и файла

Создал новый файл lab7-2.asm с помощью утилиты touch и ввожу в файл текст другойпрограммы для вывода значения регистра eax



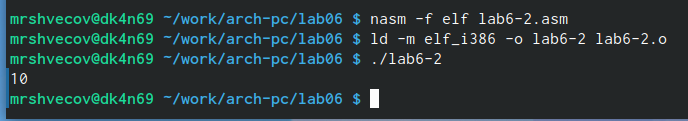
Редактирование файла

Создаю и запускаю исполняемый файл lab6-2. 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”.



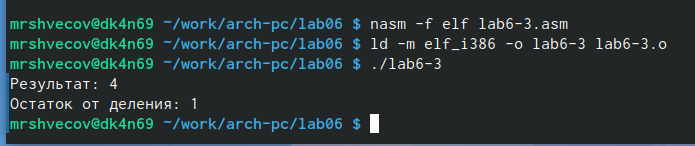
Запуск программы

Заменил в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4. Создаю и запускаю новый исполняемый файл. Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 10.



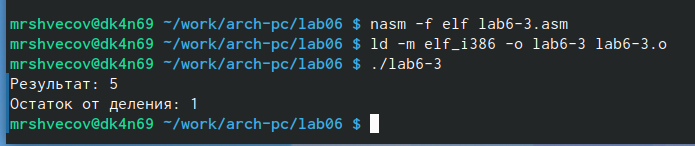
Запуск программы

Создал файл lab6-3.asm с помощью утилиты touch. Ввожу в созданный файл текст программы для вычисления значения выражения f(x) = (5 \* 2 + 3)/3. Создаю исполняемый файл и запускаю его



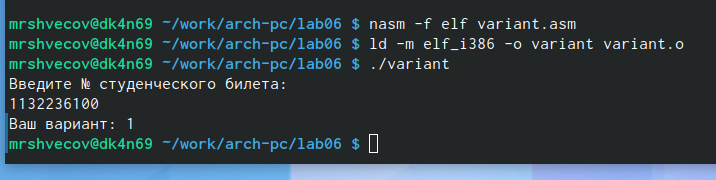
Запуск программы

Изменяю программу так, чтобы она вычисляла значение выражения f(x) = (4 \* 6 + 2)/5, создаю и запускаю новый исполняемый файл (рис. 4.19). Я посчитал для проверки правильности работы программы значение выражения самостоятельно, программа отработала верно.



Запуск программы

Создал файл variant.asm с помощью утилиты touch, ввел в файл текст программы для вычисления варианта задания по номеру студенческого билета. Создаю и запускаю исполняемый файл (рис. 4.22). Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 1.



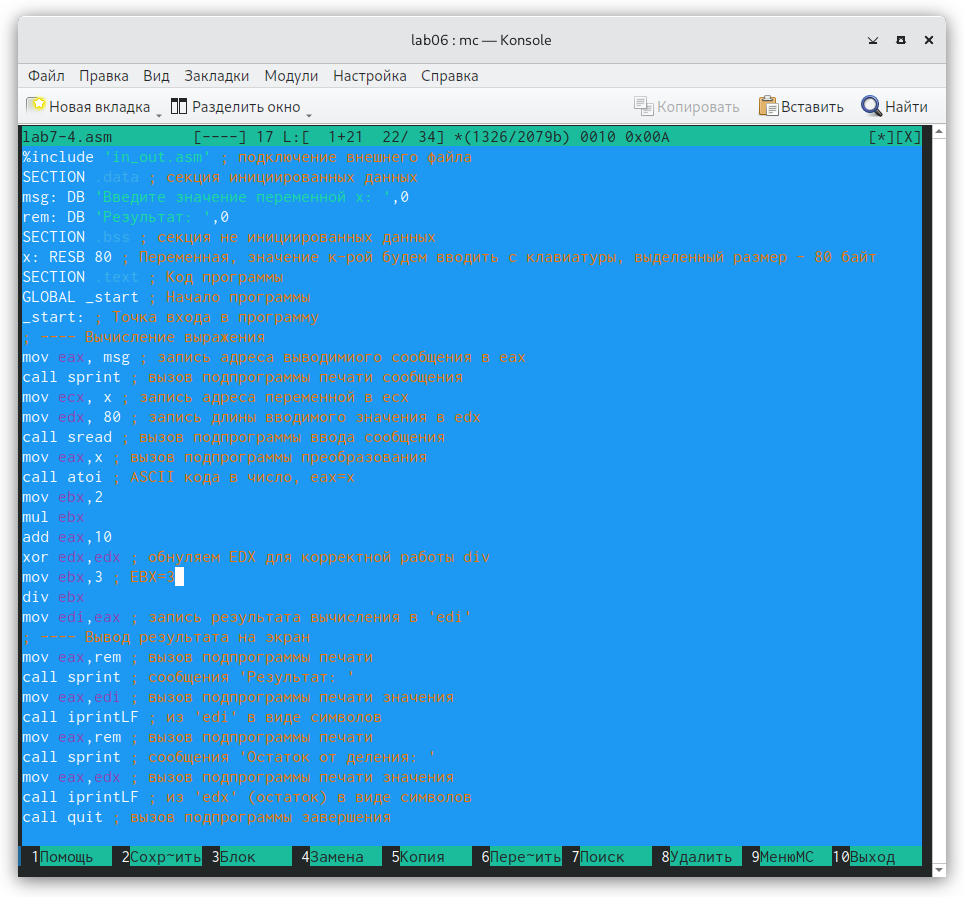
Запуск программы

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

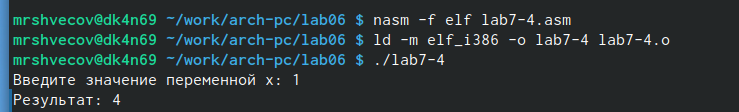
Задания для самостоятельной работы:

Создаю файл lab6-4.asm с помощью утилиты touch. Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения (10 + 2𝑥)/3 . Это выражение было под вариантом 1.



Написание программы

Создаю и запускаю исполняемый файл. При вводе значения 1, вывод - 4.



Запуск кода

Провожу еще один запуск исполняемого файла для проверки работы программы с другим значением на входе. Программа отработала верно.

Запуск кода с другими данными

Запуск кода с другими данными

Листинг 4.1. Программа для вычисления выражения (10 + 2х)/3.

%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 размер - 80 байт

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

mov ebx,2

mul ebx

add eax,10

xor edx,edx ; обнуляем EDX для корректной работы div

mov ebx,3 ; EBX=3

div 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 iprintLF ; из ‘edi’ в виде символов

mov eax,rem ; вызов подпрограммы печати

call sprint ; сообщения ‘Остаток от деления:’

mov eax,edx ; вызов подпрограммы печати значения

call iprintLF ; из ‘edx’ (остаток) в виде символов

call quit ; вызов подпрограммы завершения

# 5 Выводы

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

# Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016. URL: <https://www.gnu.org/software/bash/manual/>.

2. Newham C. [Learning the bash Shell: Unix Shell Programming](http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658). O’Reilly Media, 2005. 354 с.

3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.

4. Robbins A. [Bash Pocket Reference](https://www.ncbi.nlm.nih.gov/pubmed/25246403). O’Reilly Media, 2016. 156 с.

5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.

6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.