

Отчёт по лабораторной работе №4

дисциплина: Архитектура компьютера

Курбанов Рахман Акмурадович

Содержание

1 Цель работы

2 Задание

3 Теоретическое введение

4 Выполнение лабораторной работы

4.1 Программа Hello world!

4.2 Транслятор NASM

4.3 Расширенный синтаксис командной строки NASM

4.4 Компоновщик LD

4.5 Запуск исполняемого файла

4.6 Задания для самостоятельной работы

5 Выводы

6 Список литературы

Список иллюстраций

- 4.1 Создание рабочей директроии
- 4.2 Создание .asm файла
- 4.3 Редактирование файла
- 4.4 Компиляция программы
- 4.5 Возможности синтаксиса NASM
- 4.6 Отправка файла компоновщику
- 4.7 Создание исполняемого файла
- 4.8 Запуск программы
- 4.9 Создание копии
- 4.10 Редактирование копии
- 4.11 Проверка работоспособности скомпонованной программы
- 4.12 Отправка файлов в локальный репозиторий
- 4.13 Загрузка изменений

1 Цель работы

Цель данной лабораторной работы - освоить процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Задание

1. Создание программы Hello world!
2. Работа с транслятором NASM
3. Работа с расширенным синтаксисом командной строки NASM
4. Работа с компоновщиком LD
5. Запуск исполняемого файла
6. Выполнение заданий для самостоятельной работы.

3 Теоретическое введение

Основными функциональными элементами любой ЭВМ являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской плате. Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора входят следующие устройства: - арифметико-логическое устройство (АЛУ) — выполняет логические и арифметические действия, необходимые для обработки информации, хранящейся в памяти; - устройство управления (УУ) — обеспечивает управление и контроль всех устройств компьютера; - регистры — сверхбыстрая оперативная память небольшого объёма, входящая в состав процессора, для временного хранения промежуточных результатов выполнения инструкций; регистры процессора делятся на два типа: регистры общего назначения и специальные регистры. Для того, чтобы писать программы на ассемблере, необходимо знать, какие регистры процессора существуют и как их можно использовать. Большинство команд в программах написанных на ассемблере используют регистры в качестве операндов. Практически все команды представляют собой преобразование данных хранящихся в регистрах процессора, это например пересылка данных между регистрами или между регистрами и памятью, преобразование (арифметические или логические 7 операции) данных хранящихся в регистрах. Доступ к регистрам осуществляется не по адресам, как к основной памяти, а по именам. Каждый регистр процессора архитектуры x86 имеет свое название, состоящее из 2 или 3 букв латинского алфавита. В качестве примера приведем названия основных регистров общего назначения (именно эти регистры чаще всего используются при написании программ): - RAX, RCX, RDX, RBX, RSI, RDI — 64-битные - EAX, ECX, EDX, EBX, ESI, EDI — 32-битные - AX, CX, DX, BX, SI, DI — 16-битные - AH, AL, CH, CL, DH, DL, BH, BL — 8-битные

Другим важным узлом ЭВМ является оперативное запоминающее устройство (ОЗУ). ОЗУ — это быстродействующее энергозависимое запоминающее устройство, которое напрямую взаимодействует с узлами

процессора, предназначенное для хранения программ и данных, с которыми процессор непосредственно работает в текущий момент. ОЗУ состоит из одинаковых пронумерованных ячеек памяти. Номер ячейки памяти — это адрес хранящихся в ней данных. Периферийные устройства в составе ЭВМ: - устройства внешней памяти, которые предназначены для длительного хранения больших объёмов данных. - устройства ввода-вывода, которые обеспечивают взаимодействие ЦП с внешней средой. В основе вычислительного процесса ЭВМ лежит принцип программного управления. Это означает, что компьютер решает поставленную задачу как последовательность действий, записанных в виде программы.

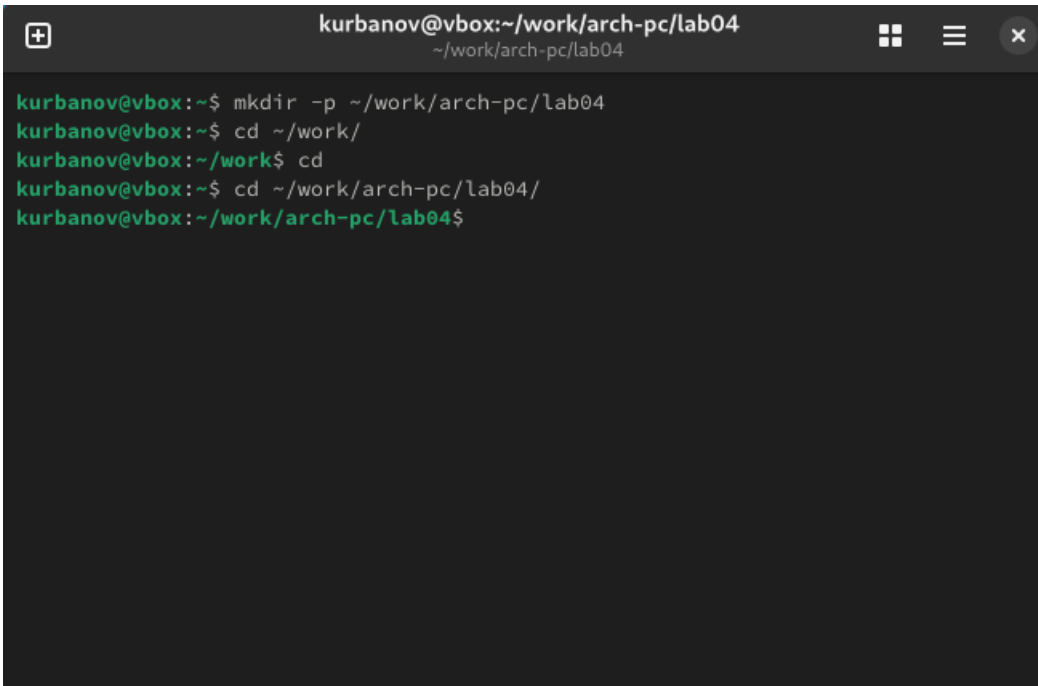
Коды команд представляют собой многоразрядные двоичные комбинации из 0 и 1. В коде машинной команды можно выделить две части: операционную и адресную. В операционной части хранится код команды, которую необходимо выполнить. В адресной части хранятся данные или адреса данных, которые участвуют в выполнении данной операции. При выполнении каждой команды процессор выполняет определённую последовательность стандартных действий, которая называется командным циклом процессора. Он заключается в следующем: 1. формирование адреса в памяти очередной команды; 2. считывание кода команды из памяти и её дешифрация; 3. выполнение команды; 4. переход к 8 следующей команде.

Язык ассемблера (assembly language, сокращённо asm) — машинноориентированный язык низкого уровня. NASM — это открытый проект ассемблера, версии которого доступны под различные операционные системы и который позволяет получать объектные файлы для этих систем. В NASM используется Intel-синтаксис и поддерживаются инструкции x86-64.

4 Выполнение лабораторной работы

4.1 Программа Hello world!

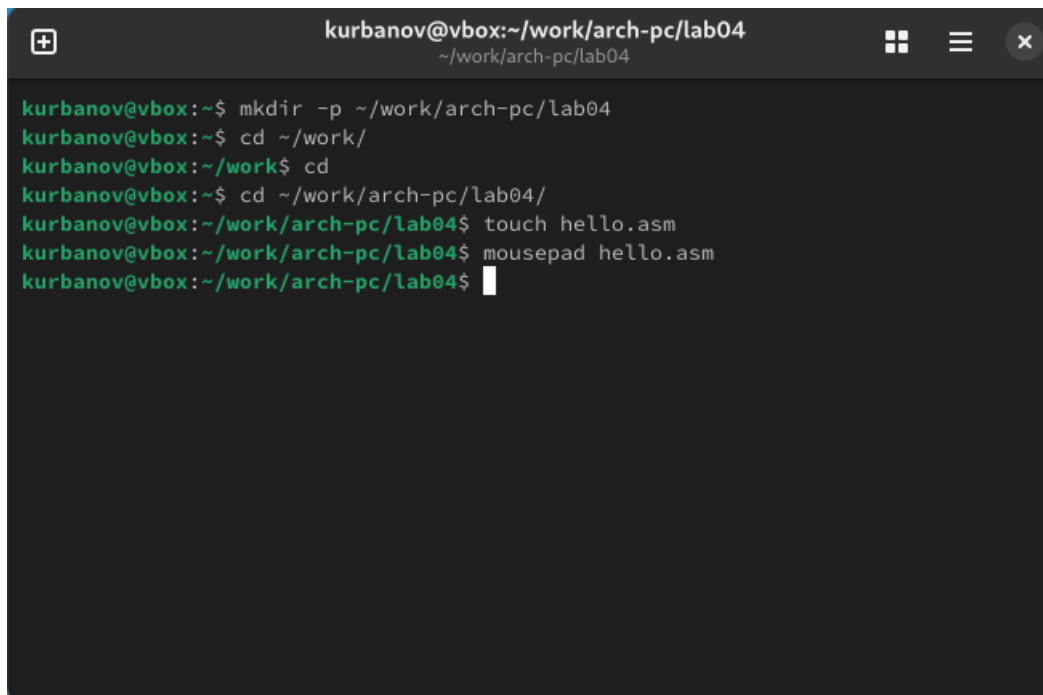
В домашней директории создаю каталог, в котором буду хранить файлы для текущей лабораторной работы. (рис. 4.1)

A screenshot of a terminal window with a dark background. The title bar at the top shows the user 'kurbanov@vbox' and the current directory '~/work/arch-pc/lab04'. The terminal contains the following commands and their outputs:

```
kurbanov@vbox:~$ mkdir -p ~/work/arch-pc/lab04
kurbanov@vbox:~$ cd ~/work/
kurbanov@vbox:~/work$ cd
kurbanov@vbox:~$ cd ~/work/arch-pc/lab04/
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.1: Создание рабочей директории

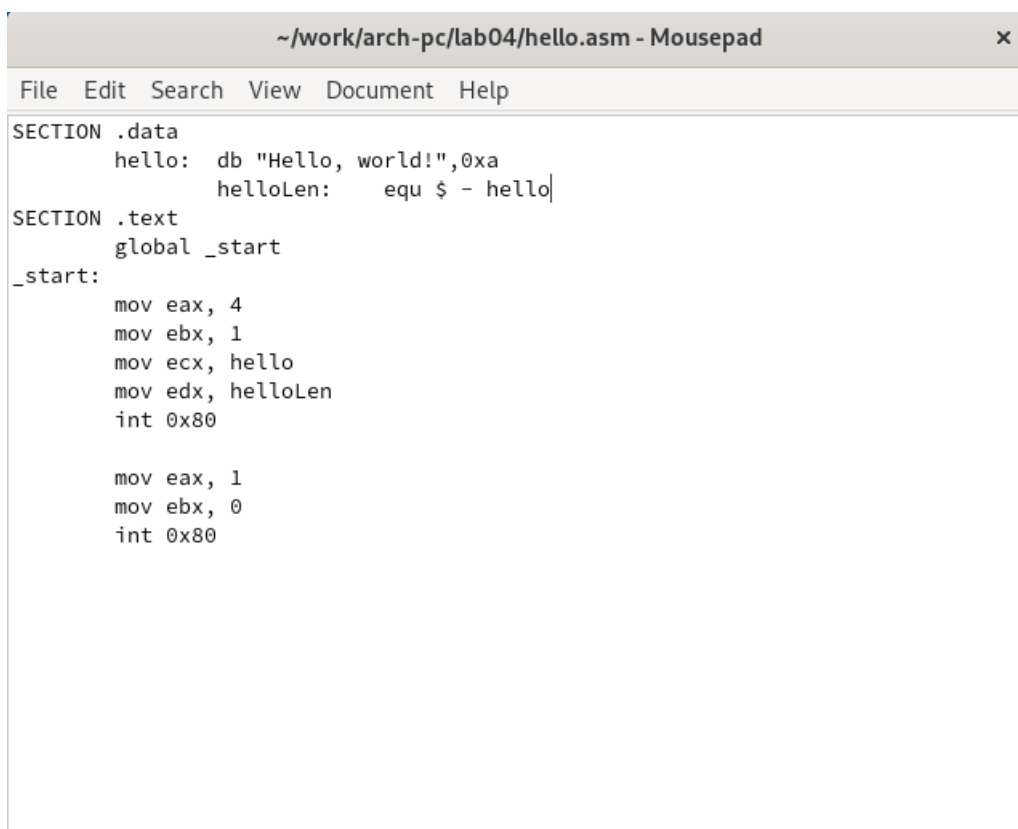
Создаю в нем файл hello.asm, в котором буду писать программу на языке ассемблера. (рис. 4.2)



```
kurbanov@vbox:~/work/arch-pc/lab04
~/work/arch-pc/lab04
kurbanov@vbox:~$ mkdir -p ~/work/arch-pc/lab04
kurbanov@vbox:~$ cd ~/work/
kurbanov@vbox:~/work$ cd
kurbanov@vbox:~$ cd ~/work/arch-pc/lab04/
kurbanov@vbox:~/work/arch-pc/lab04$ touch hello.asm
kurbanov@vbox:~/work/arch-pc/lab04$ mousepad hello.asm
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.2: Создание .asm файла

С помощью редактора пишу программу в созданном файле. (рис. 4.3)



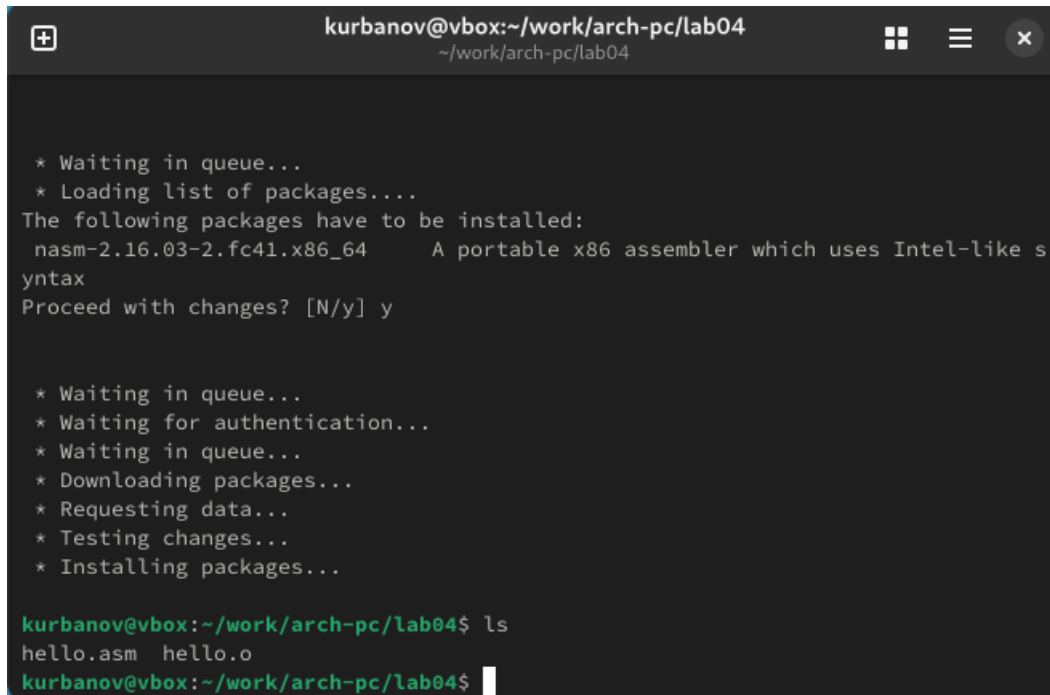
```
~/work/arch-pc/lab04/hello.asm - Mousepad
File Edit Search View Document Help
SECTION .data
    hello: db "Hello, world!",0xa
           helloLen: equ $ - hello
SECTION .text
    global _start
_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, hello
    mov edx, helloLen
    int 0x80

    mov eax, 1
    mov ebx, 0
    int 0x80
```

Рис. 4.3: Редактирование файла

4.2 Транслятор NASM

Компилирую с помощью NASM свою программу. (рис. 4.4)



```
kurbanov@vbox:~/work/arch-pc/lab04
~/work/arch-pc/lab04

* Waiting in queue...
* Loading list of packages....
The following packages have to be installed:
nasm-2.16.03-2.fc41.x86_64      A portable x86 assembler which uses Intel-like s
yntax
Proceed with changes? [N/y] y

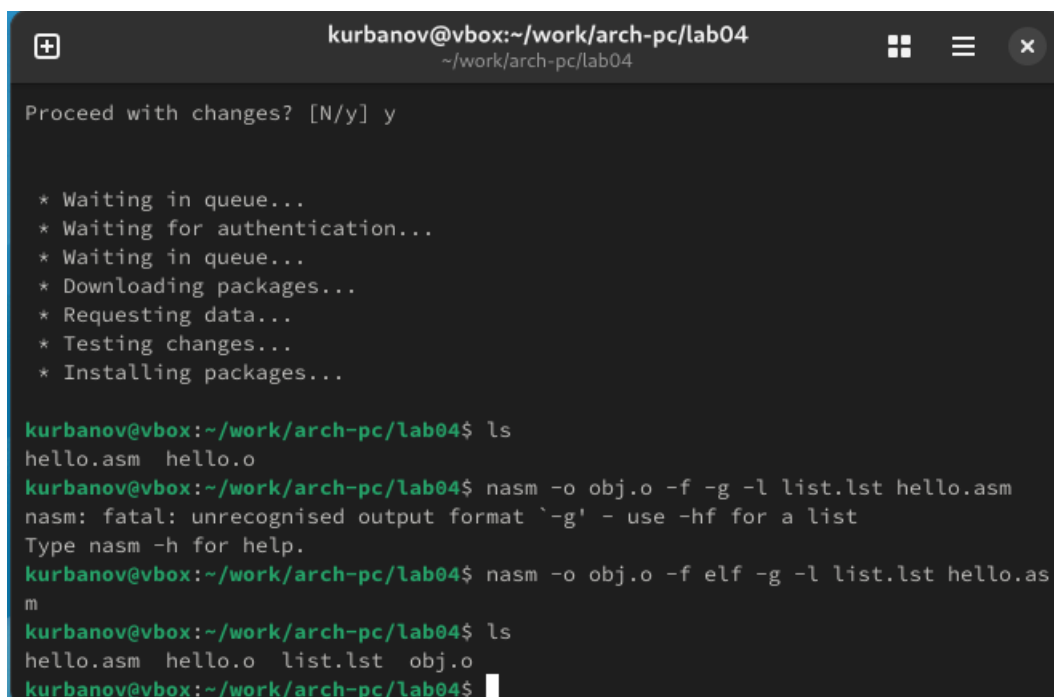
* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.4: Компиляция программы

4.3 Расширенный синтаксис командной строки NASM

Выполняю команду, указанную на (рис. 4.5), она скомпилировала исходный файл hello.asm в obj.o, расширение .o говорит о том, что файл - объектный, помимо него флаги -g -l подготовят файл отладки и листинга соответственно.



```
kurbanov@vbox:~/work/arch-pc/lab04
~/work/arch-pc/lab04

Proceed with changes? [N/y] y

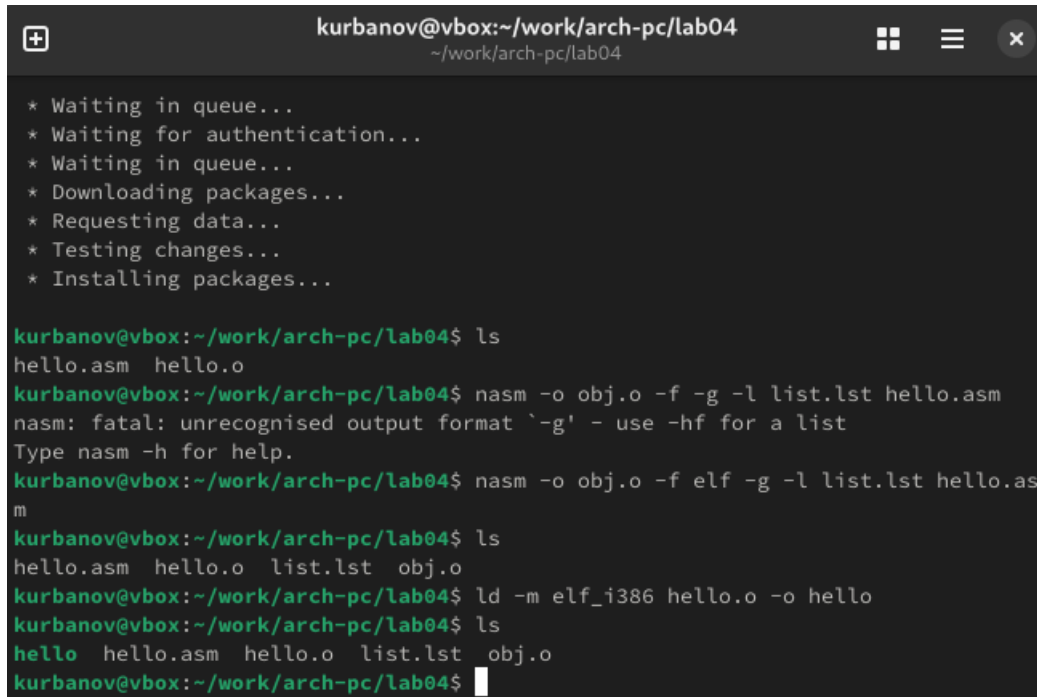
* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f -g -l list.lst hello.asm
nasm: fatal: unrecognised output format '-g' - use -hf for a list
Type nasm -h for help.
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.as
m
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.5: Возможности синтаксиса NASM

4.4 Компоновщик LD

Затем мне необходимо передать объектный файл компоновщику, делаю это с помощью команды `ld`. (рис. 4.6)



```
kurbanov@vbox:~/work/arch-pc/lab04
~ /work/arch-pc/lab04

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm hello.o
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f -g -l list.lst hello.asm
nasm: fatal: unrecognised output format '-g' - use -hf for a list
Type nasm -h for help.
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm hello.o list.lst obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o list.lst obj.o
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.6: Отправка файла компоновщику

Выполняю следующую команду ..., результатом исполнения команды будет созданный файл `main`, скомпонованный из объектного файла `obj.o`. (рис. 4.7)

```
kurbanov@vbox:~/work/arch-pc/lab04
~ /work/arch-pc/lab04

* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f -g -l list.lst hello.asm
nasm: fatal: unrecognised output format '-g' - use -hf for a list
Type nasm -h for help.
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o main
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.7: Создание исполняемого файла

4.5 Запуск исполняемого файла

Запускаю исполняемый файл из текущего каталога. (рис. 4.8)

```
kurbanov@vbox:~/work/arch-pc/lab04
~ /work/arch-pc/lab04

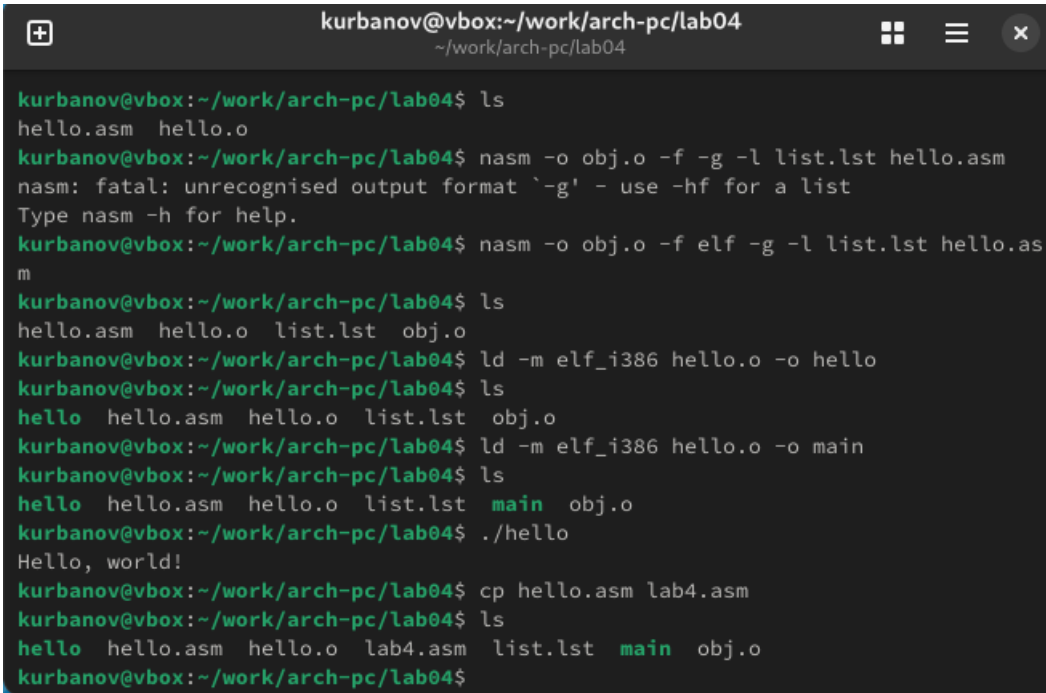
* Testing changes...
* Installing packages...

kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f -g -l list.lst hello.asm
nasm: fatal: unrecognised output format '-g' - use -hf for a list
Type nasm -h for help.
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o main
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ./hello
Hello, world!
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.8: Запуск программы

4.6 Задания для самостоятельной работы

Создаю копию файла для последующей работы с ней. (рис. 4.9)

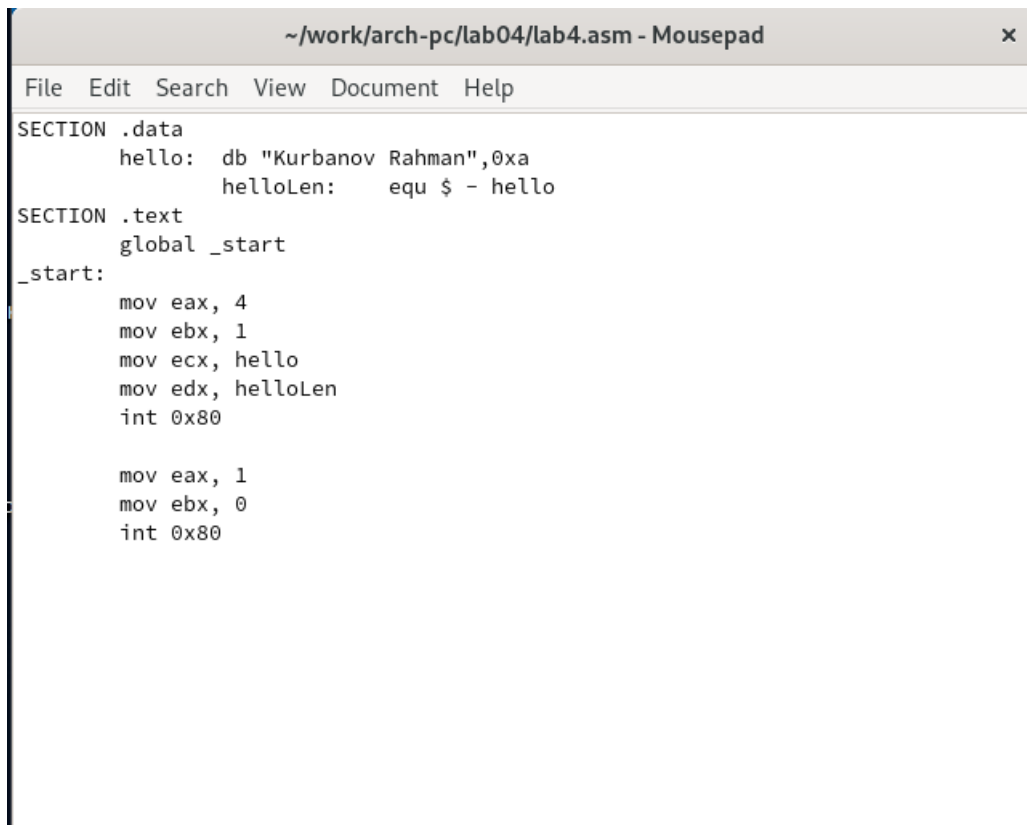


```
kurbanov@vbox:~/work/arch-pc/lab04
~/work/arch-pc/lab04

kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f -g -l list.lst hello.asm
nasm: fatal: unrecognised output format '-g' - use -hf for a list
Type nasm -h for help.
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.as
m
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o main
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ./hello
Hello, world!
kurbanov@vbox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.9: Создание копии

Редактирую копию файла, заменив текст на свое имя и фамилию. (рис. 4.10)

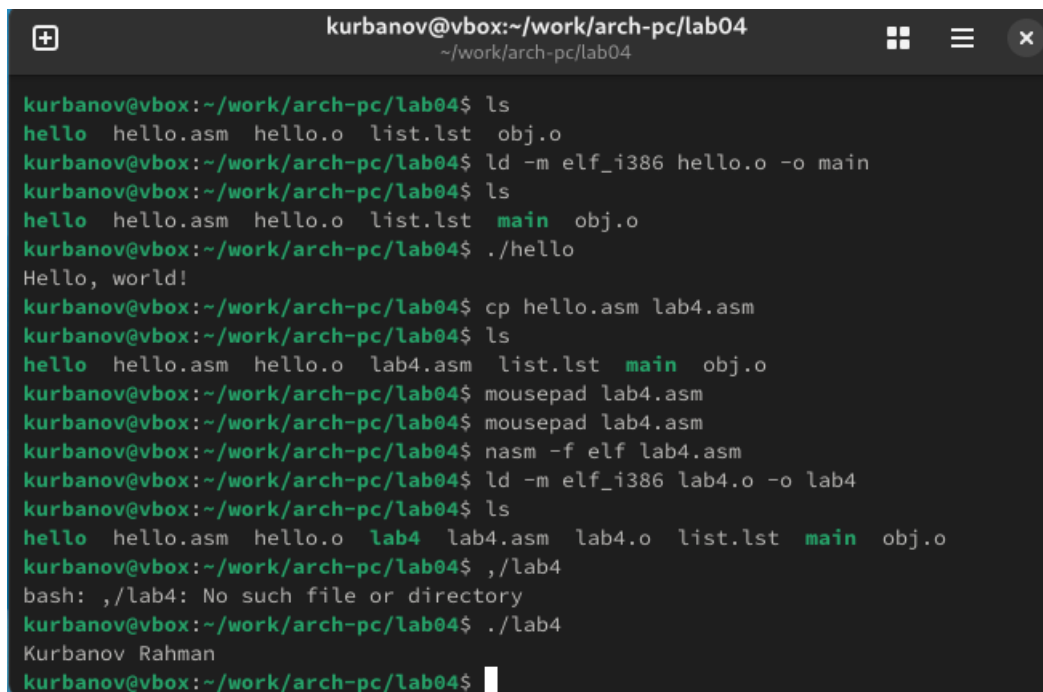


```
~/work/arch-pc/lab04/lab4.asm - Mousepad
File Edit Search View Document Help
SECTION .data
    hello: db "Kurbanov Rahman",0xa
           helloLen: equ $ - hello
SECTION .text
    global _start
_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, hello
    mov edx, helloLen
    int 0x80

    mov eax, 1
    mov ebx, 0
    int 0x80
```

Рис. 4.10: Редактирование копии

Транслирую копию файла в объектный файл, компоную и запускаю. (рис. 4.11)



```
kurbanov@vbox:~/work/arch-pc/lab04
~/work/arch-pc/lab04
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o list.lst obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o main
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o list.lst main obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ./hello
Hello, world!
kurbanov@vbox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4.asm list.lst main obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ mousepad lab4.asm
kurbanov@vbox:~/work/arch-pc/lab04$ mousepad lab4.asm
kurbanov@vbox:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
kurbanov@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
kurbanov@vbox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o
kurbanov@vbox:~/work/arch-pc/lab04$ ./lab4
bash: ./lab4: No such file or directory
kurbanov@vbox:~/work/arch-pc/lab04$ ./lab4
Kurbanov Rahman
kurbanov@vbox:~/work/arch-pc/lab04$
```

Рис. 4.11: Проверка работоспособности скомпонованной программы

Убедившись в корректности работы программы, копирую рабочие файлы в свой локальный репозиторий. (рис. 4.12)

```
kurbanov@vbox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm ../../study/2024-2025/
'архитектура компьютера'/arch-pc/labs/lab04
kurbanov@vbox:~/work/arch-pc/lab04$ cd ../../study/2024-2025/'архитектура компь
тера'/arch-pc/labs/lab04/
kurbanov@vbox:~/work/study/2024-2025/архитектура компьютера/arch-pc/labs/lab04$
ls
hello.asm lab4.asm presentation report
kurbanov@vbox:~/work/study/2024-2025/архитектура компьютера/arch-pc/labs/lab04$
```

Рис. 4.12: Отправка файлов в локальный репозиторий

Загрузка изменений на свой удаленный репозиторий на GitHub. (рис. 4.13)

```
kurbanov@vbox:~/work/study/2024-2025/архитектура компьют...
~/work/study/2024-2025/архитектура компьютера/arch-pc/labs/lab04
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   hello.asm
    new file:   lab4.asm

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ../../hello.asm

kurbanov@vbox:~/work/study/2024-2025/архитектура компьютера/arch-pc/labs/lab04$
git comit -m "feat(main): upload 4 lab work"
git: 'comit' is not a git command. See 'git --help'.

The most similar command is
  commit
kurbanov@vbox:~/work/study/2024-2025/архитектура компьютера/arch-pc/labs/lab04$
git commit -m "feat(main): upload 4 lab work"
[master 101d8f4] feat(main): upload 4 lab work
 2 files changed, 30 insertions(+)
 create mode 100644 labs/lab04/hello.asm
 create mode 100644 labs/lab04/lab4.asm
kurbanov@vbox:~/work/study/2024-2025/архитектура компьютера/arch-pc/labs/lab04$
```

Рис. 4.13: Загрузка изменений

5 Выводы

При выполнении данной лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.