

# Отчёта по лабораторной работе №5

## Дисциплина: архитектура компьютера

Царев Максим Александрович

### Содержание

1	Цель работы .....	1
2	Задание.....	1
3	Теоретическое введение .....	1
4	Выполнение лабораторной работы .....	3
5	Выводы.....	7

### 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 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 долговременного 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.

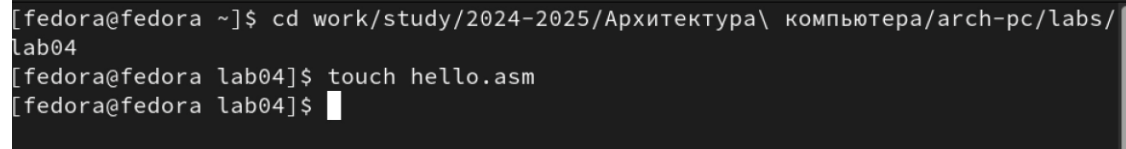
Коды команд представляют собой многоразрядные двоичные комбинации из 0 и 1. В коде машинной команды можно выделить две части: операционную и адресную. В операционной части хранится код команды, которую необходимо выполнить. В адресной части хранятся данные или адреса данных, которые участвуют в выполнении 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 следующей команде.

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный 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

С помощью утилиты `cd` перемещаюсь в каталог и создаю файл

A terminal window with a dark background and light text. It shows three lines of commands and their outputs. The first line shows the user navigating to a specific directory using the 'cd' command. The second line shows the user creating a new file named 'hello.asm' using the 'touch' command. The third line shows the prompt after the command is executed.

```
[fedora@fedora ~]$ cd work/study/2024-2025/Архитектура\ компьютера/arch-pc/labs/  
lab04  
[fedora@fedora lab04]$ touch hello.asm  
[fedora@fedora lab04]$
```

*Рис. 1: Перемещение между директориями*

С помощью текстового редактора `mouespad` открываю файл `hello.asm` и вношу изменения. Заполняю файл, вставляя в него программу для вывода “Hello word!”

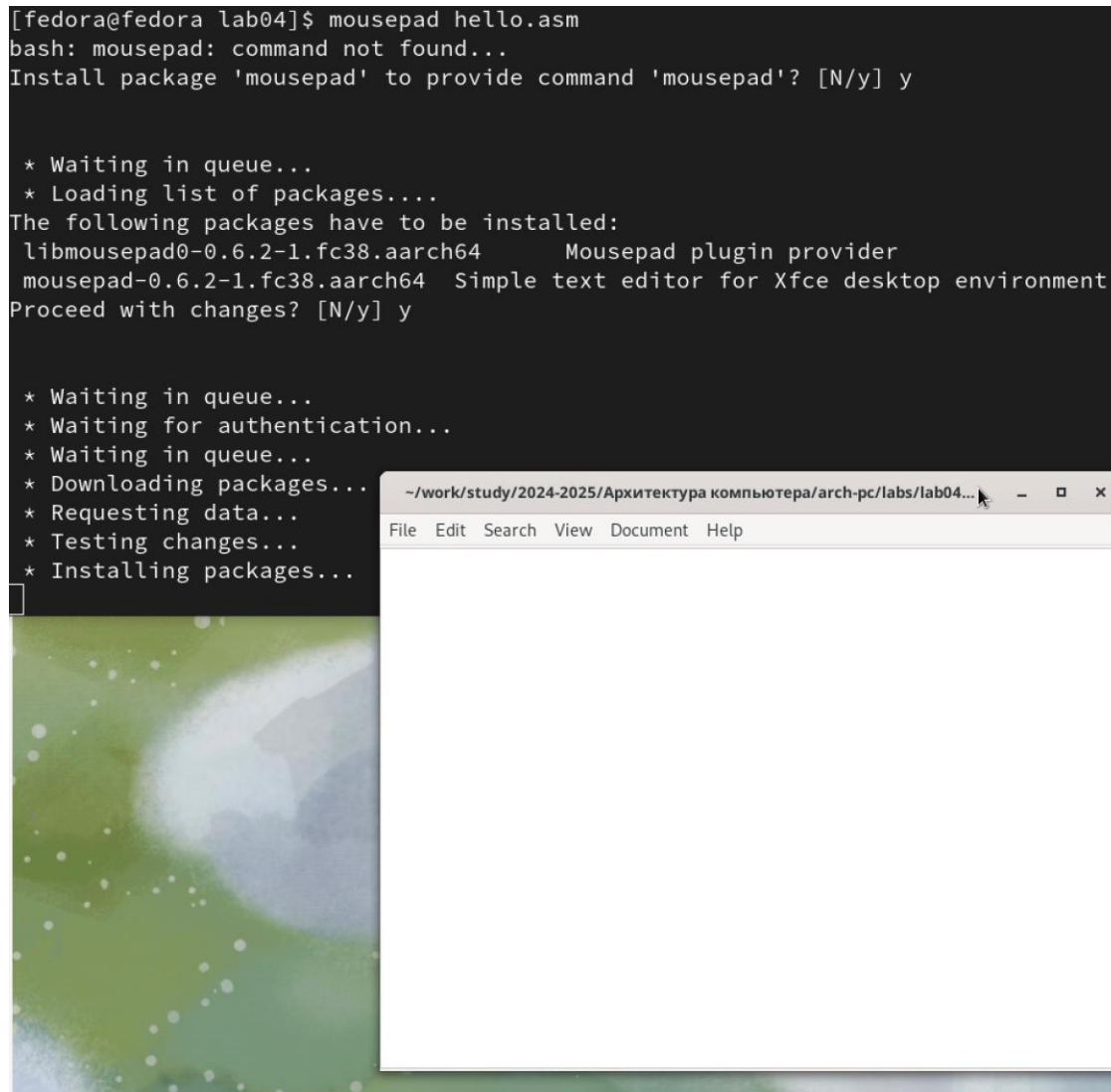
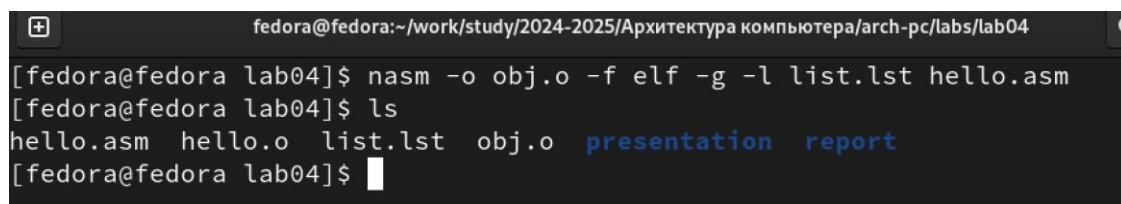
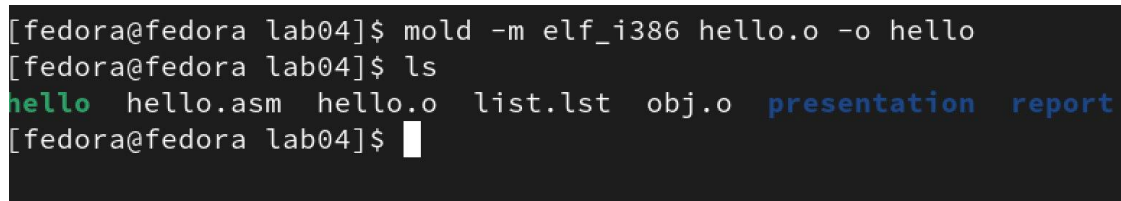


Рис. 2: Создание пустого файла



Превращаю текст программы для вывода "Hello world!" в объектный код с помощью транслятора NASM



Ввожу команду, которая скомпилирует файл hello.asm в файл obj.o

```
[fedora@fedora lab04]$ ls
hello hello.asm hello.o list.lst main obj.o presentation report
[fedora@fedora lab04]$
```

Передаю объектный файл hello.o на обработку компоновщику MOLD, чтобы получить исполняемый файл hello

```
fedora@fedora:~/work/
[fedora@fedora lab04]$ hello
Hello, world!
[fedora@fedora lab04]$
```

Файл будет иметь имя main потому что после ключа -o было задано значение main

```
[fedora@fedora lab04]$ cp hello.asm lab4.asm
[fedora@fedora lab04]$ mousepad lab4.asm
[fedora@fedora lab04]$ nasm -f elf lab4.asm
[fedora@fedora lab04]$ ls
hello hello.o lab4.asm list.lst obj.o report
hello.asm lab04.asm lab4.o main presentation
```

Запускаю на выполнение созданный исполняемый файл hello

```
[fedora@fedora lab04]$ mold -m elf_i386 lab4.o -o lab4
[fedora@fedora lab04]$ ls
hello hello.o lab4 lab4.o main presentation
hello.asm lab04.asm lab4.asm list.lst obj.o report
[fedora@fedora lab04]$
```

С помощью утилиты cp создаю в текущем каталоге копию файла hello.asm с именем lab4.asm С помощью редактора mousepad открываю файл lab4.asm и вношу изменения в программу Компилирую текст программы в объектный файл

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04
[fedora@fedora lab04]$ nasm -f elf hello.asm
bash: nasm: command not found...
Install package 'nasm' to provide command 'nasm'? [N/y] y

* Waiting in queue...
* Loading list of packages....
The following packages have to be installed:
nasm-2.16.01-3.fc38.aarch64    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...

[fedora@fedora lab04]$ ls
hello.asm  hello.o  presentation  report
[fedora@fedora lab04]$
```

Передаю объектный файл lab4.o на обработку компоновщику MOLD, чтобы получить исполняемый файл lab4

```
[fedora@fedora lab04]$ ls
hello      hello.o  lab4.asm  list.lst  presentation
hello.asm  lab4     lab4.o    main      report
[fedora@fedora lab04]$ ./lab4
Tsarev Maksim
[fedora@fedora lab04]$
```

Запускаю исполняемый файл lab4, на экран действительно выводятся мои имя и фамилия

```
[fedora@fedora lab04]$ git add .
[fedora@fedora lab04]$ git commit -m "Add fales for lab04"
[master 07dbad5] Add fales for lab04
8 files changed, 55 insertions(+)
create mode 100755 labs/lab04/hello
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/hello.o
create mode 100755 labs/lab04/lab4
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/lab4.o
create mode 100644 labs/lab04/list.lst
create mode 100755 labs/lab04/main
[fedora@fedora lab04]$ git push
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 4 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 3.76 KiB | 3.76 MiB/s, done.
Total 11 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 2 local objects.
To github.com:gravitai06/study_2024-2025_ar-h-pc.git
   8d91c3a..07dbad5  master -> master
[fedora@fedora lab04]$
```

После всех проверок пушу в свой репозиторий

## 5 Выводы

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