

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

дисциплина: Архитектура компьютера

Студент: Скворцова Дарья Дмитриевна

Группа: НКАбд-05-25

МОСКВА

2025 г.

Содержание

1 Цель работы.....	4
2 Задание.....	4
3 Теоретическое введение.....	4
4 Выполнение лабораторной работы.....	5
5 Выводы.....	9
6 Список литературы.....	9

Список иллюстраций

(Рис. 1): Создание рабочей директории 5

(Рис. 2): Создание .asm файла 6

(Рис. 3): Редактирование файла 6

(Рис. 4): Компиляция программы 6

(Рис. 5): Возможности синтаксиса NASM 7

(Рис. 6): Отправка файла компоновщику 7

(Рис. 7): Создание исполняемого файла 7

(Рис. 8): Запуск программы 7

(Рис. 9): Создание копии 7

(Рис. 10): Редактирование копии 8

(Рис. 11): Проверка работоспособности скомпонованной программы 8

(Рис. 12): Отправка файлов в локальный репозиторий 8

(Рис. 13): Загрузка изменений 9

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 operandов. 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 своё название, состоящее из 2 или 3 букв латинского алфавита. В качестве примера приведём 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

4.1 Программа Hello world!

В домашней директории создаю каталог, в котором буду хранить файлы для текущей лабораторной работы. (рис. 1)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~$ mkdir -p ~/work/arch-pc/lab04
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~$ cd ~/work/arch-pc/lab04
```

(Рис. 1): Создание рабочей директории

Создаю в нём файл hello.asm, в котором буду писать программу на языке ассемблера. (рис. 2)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ touch hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ gedit hello.asm
```

(Рис. 2): Создание .asm файла

С помощью редактора пишу программу в созданном файле. (рис. 3)



The screenshot shows the GEdit text editor interface. The title bar says "hello.asm" and the path is "~ / work / arch - pc / lab04". The file contains the following assembly code:

```
1 ; hello.asm
2 SECTION .data
3 hello: DB 'Hello world!',10
4
5 helloLen: EQU $-hello
6 SECTION .text
7 GLOBAL _start
8 _start:
9 mov eax, 4
10 mov ebx, 1
11 mov ecx, hello
12 mov edx, helloLen
13 int 80h
14 mov eax, 1
15 mov ebx, 0
16 int 80h
```

(Рис. 3): Редактирование файла

4.2 Транслятор NASM

Компилирую с помощью NASM свою программу. (рис. 4)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~$ mkdir -p ~/work/arch-pc/lab04
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~$ cd ~/work/arch-pc/lab04
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ touch hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ gedit hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ nasm -f elf hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$
```

~/work/arch-pc/lab04

(Рис. 4): Компиляция программы

4.3 Расширенный синтаксис командной строки NASM

Выполняю команду, указанную на (рис. 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.

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ mkdir -p ~/work/arch-pc/lab04
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ cd ~/work/arch-pc/lab04
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ touch hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ gedit hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ nasm -f elf hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ls
hello.asm hello.o
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ls
hello.asm hello.o list.lst obj.o
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ █
```

(Рис. 5): Возможности синтаксиса NASM

4.4 Компоновщик LD

Затем мне необходимо передать объектный файл компоновщику, делаю это с помощью команды ld. (рис. 6)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
```

(Рис. 6): Отправка файла компоновщику

Выполняю следующую команду, результатом исполнения команды будет созданный файл main, скомпонованный из объектного файла obj.o. (рис. 7)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
```

(Рис. 7): Создание исполняемого файла

4.5 Запуск исполняемого файла

Запускаю исполняемый файл из текущего каталога. (рис. 8)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ./hello
Hello world!
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ █
```

(Рис. 8): Запуск программы

4.6 Задания для самостоятельной работы

Создаю копию файла для последующей работы с ней. (рис. 9)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ cp ~/work/arch-pc/lab04/hello.asm lab4.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4.asm list.lst main obj.o
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$
```

(Рис. 9): Создание копии

Редактирую копию файла, заменив текст на свое имя и фамилию. (рис. 10)



The screenshot shows a text editor window titled "lab4.asm" located at "/work/arch-pc/lab04". The code in the editor is as follows:

```
1 ; hello.asm
2 SECTION .data
3 hello: DB 'Ckvertsova Daria',10
4
5 helloLen: EQU $-hello
6 SECTION .text
7 GLOBAL _start
8 _start:
9 mov eax, 4
10 mov ebx, 1
11 mov ecx, hello
12 mov edx, helloLen
13 int 80h
14 mov eax, 1
15 mov ebx, 0
16 int 80h
```

(Рис. 10): Редактирование копии

Транслирую копию файла в объектный файл, компоную и запускаю. (рис. 11)

```
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ gedit hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ nasm -f elf hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$ ./hello
Ckvertsova Daria
ckvortsova@ckvortsova-HP-Pavilion-Laptop-14-ce3xxx:~/work/arch-pc/lab04$
```

(Рис. 11): Проверка работоспособности скомпонованной программы

Убедившись в корректности работы программы, копирую рабочие файлы в свой локальный репозиторий. (рис. 12)

```
mazurskiy@fedora:~/work/arch-pc/lab04$ cp hello.asm lab4.asm ../../study/2024-2025/arch/arch-pc/labs/lab04/
:~/work/arch-pc/lab04$ cd ../../study/2024-2025/arch/arch-pc/labs/lab04/
```

(Рис. 12): Отправка файлов в локальный репозиторий

Загрузка изменений на свой удаленный репозиторий на GitHub. (рис. 13)

```
mazurskiy@fedora:-/work/study/2024-2025/arch/arch-pc/labs/lab04$ git add .
mazurskiy@fedora:-/work/study/2024-2025/arch/arch-pc/labs/lab04$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:  hello.asm
    new file:  lab4.asm

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:  ../../lab03/report/report.docx
    modified:  ../../lab03/report/report.pdf

:~/work/study/2024-2025/arch/arch-pc/labs/lab04$ git commit -m "feat(main): upload 4 lab work"
[master a3a6999] feat(main): upload 4 lab work
 2 files changed, 32 insertions(+)
 create mode 100644 labs/Lab04/hello.asm
 create mode 100644 labs/Lab04/lab4.asm
:~/work/study/2024-2025/arch/arch-pc/labs/lab04$ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 681 bytes | 681.00 KiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:owherewashesere/study_2024_2025_arch-pc.git
 7fa9af..a3a6999 master -> master
```

(Рис. 13): Загрузка изменений

5 Выводы

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

6 Список литературы