**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ДОКЛАД**

**на тему «НАЗВАНИЕ ТЕМЫ»**

*дисциплина: Архитектура компьютера*

Студент: Ким И.Е.

Группа: НКАбд-03-25

№ ст. билета: 1032253496

**МОСКВА 2025 г.**

**Содержание**

**[1. Цель работы………………………………………………………………...………...4](#k1)**

**[2. Задание…………………………………………………………………...……………5](#k2)**

**[3. Теоретическое введение………………………………………………...…………...6](#k3)**

**[4. Выполнение лабораторной работы……………………………………...………...8](#k4)**

[4.1 Программа Hello world!...........................................................................................8](#k5)

[4.2 Транслятор NASM………………………………………...………………………9](#k6)

[4.3 Расширенный синтаксис……………………………………………………...…..9](#k7)

[4.4 Компоновщик LD……………………………………………………...…………10](#k8)

[4.5 Запуск исполняемого файла………………………………………...…………...11](#k9)

[4.6 Задания для самостоятельной работы……………………………...…………...11](#k10)

**[5. Выводы………………………………………………………………………………14](#k11)**

**[6. Список литературы………………………………………………………………...15](#k12)**

**Список иллюстраций**

[4.1 Создание рабочей директории………………………………………………………8](#k13)

[4.2 Создание .asm файла…………………………………………………………………8](#k14)

[4.3 Редактирование файла……………………………………………………………….8](#k15)

[4.4 Компиляция программы……………………………………………………………..9](#k16)

[4.5 Возможность синтаксиса NASM …………………………………………………...9](#k17)

[4.6 Отправка файла компоновщику……………………………………………………10](#k18)

[4.7 Создание исполняемого файла…………………………………………………….10](#k19)

[4.8 Запуск программы…………………………………………………………………..11](#k20)

[4.9 Создание копии……………………………………………………………………..11](#k21)

[4.10 Редактирование копии…………………………………………………………….12](#k22)

[4.11 Проверка работоспособности скомпованной программы………………………12](#k23)

[4.12 Отправка файлов в локальный репозиторий…………………………………….13](#k24)

**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 про-

водники 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**

**4.1** **Программа Hello world!**

В домашней директории создаю каталог, в котором буду хранить файлы для

текущей лабораторной работы. (рис. 4.1)

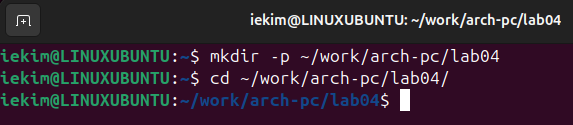


Рис. 4.1: Создание рабочей директории

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

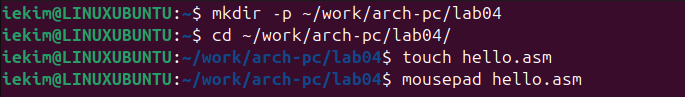


Рис. 4.2: Создание .asm файла

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

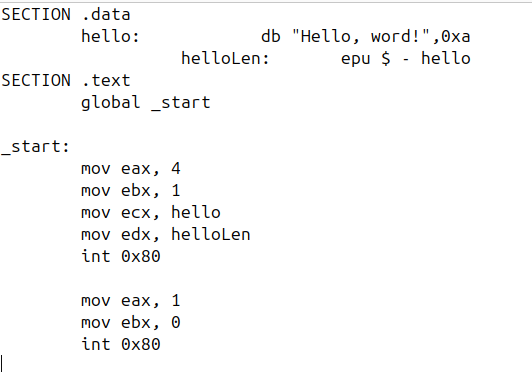


Рис. 4.3: Редактирование файла

**4.2** **Транслятор NASM**

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

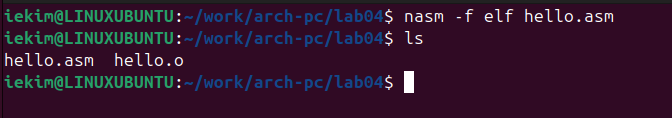


Рис. 4.4: Компиляция программы

**4.3** **Расширенный синтаксис командной строки NASM**

Выполняю команду, указанную на (рис. 4.5), она скомпилировала исходный

файл hello.asm в obj.o, расшиерние .o говорит о том, что файл - объектный, помимо

него флаги -g -l подготвоят файл отладки и листинга соответственно.

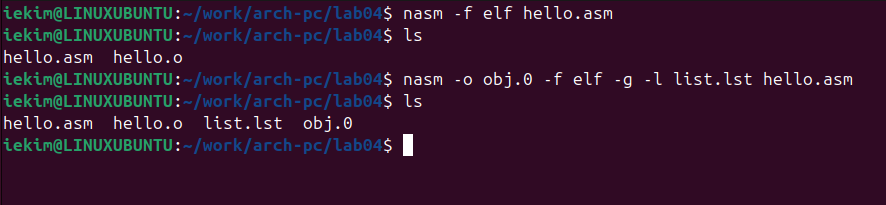


Рис. 4.5: Возможности синтаксиса NASM

**4.4** **Компоновщик LD**

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

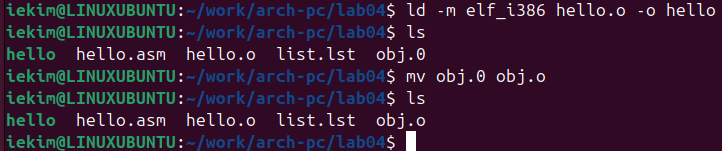


Рис. 4.6: Отправка файла компоновщику

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

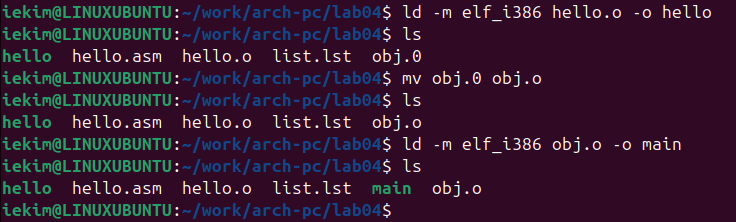


Рис. 4.7: Создание исполняемого файла

**4.5** **Запуск исполняемого файла**

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

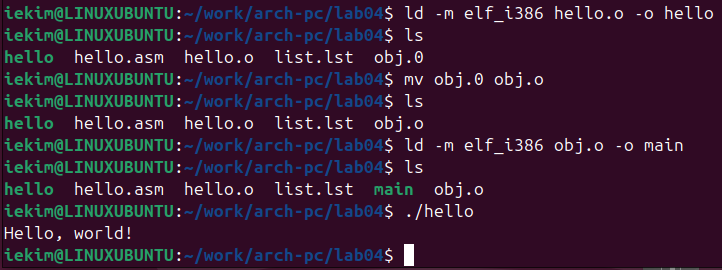


Рис. 4.8: Запуск программы

**4.6** **Задания для самостоятельной работы**

Создаю копию файла, заменив текст на своё имя и фамилию. (рис. 4.10)

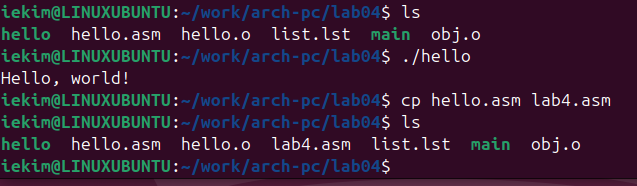


Рис. 4.9: Создание копии

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

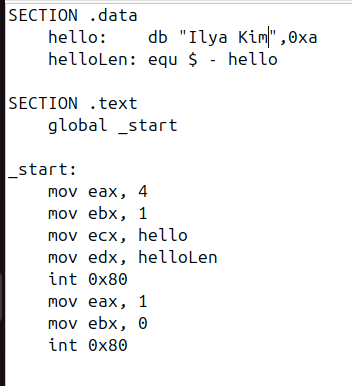


Рис. 4.10: Редактирование копии

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

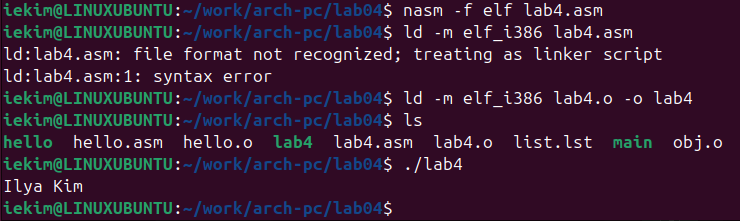
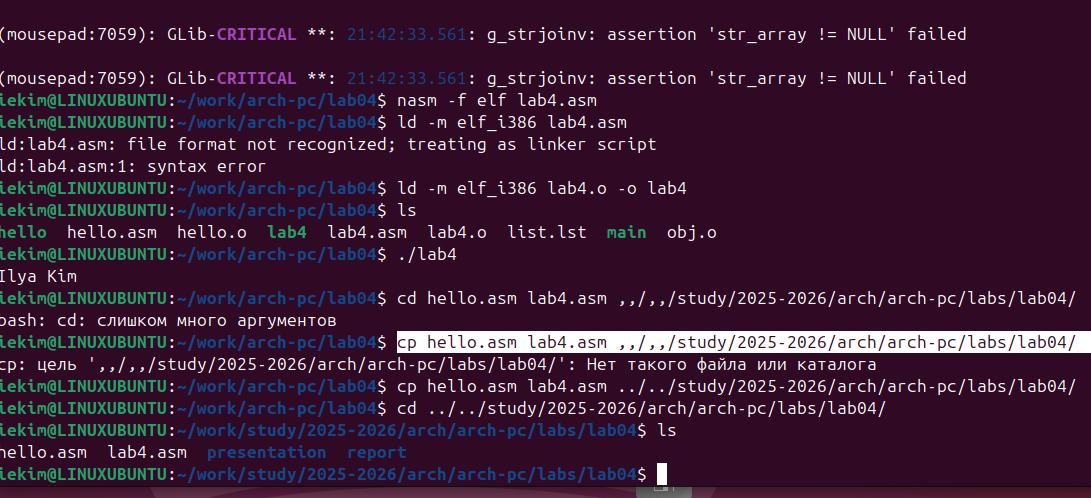


Рис. 4.11: Проверка работоспособности скомпилированной программы

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

Рис. 4.12: Отправка файлов на локальный репозиторий

**5.** **Вывод**

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

**6.** **Список литературы**

1. Пример выполнения лабораторной работы.

2. Курс на ТУИС.

3. Лабораторная работа №4

4. Программирование на языке ассемблера NASM Ким И.Е.