**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: Чувакина Мария Владимировна

Группа: НКАбд-06-23

**МОСКВА**

2023 г.

**Содержание**

1. **Цель работы……………………………………………………….………….4**
2. **Задание……………………………………………………………….………..5**
3. **Теоретическое введение…………………………………………….……….6**
4. **Выполнение лабораторной̆ работы……………………………….……….8**

4.1  Создание программы Hello world!.............................................................8

4.2  Работа с транслятором NASM ...................................................................9

4.3  Работа с расширенным синтаксисом командной строки NASM . . ..…10

4.4  Работа с компоновщиком LD ...................................................................10

4.5  Запуск исполняемого файла......................................................................11

4.6 Выполнение заданий для самостоятельной работы. . . . .. ……………..11

1. **Выводы………………………………………………………………………..14**
2. **Список литературы………………………………………………………….15**

**Список иллюстраций**

4.1  Перемещение между директориями ................................................................ 8

4.2  Создание пустого файла..................................................................................... 8

4.3  Открытие файла в текстовом редакторе . . . ………………….... . . . . . . . . . . 8

4.4  Заполнение файла.................................................................................................9

4.5  Компиляция текста программы........................................................................ 10

4.6  Компиляция текста программы........................................................................ 10

4.7  Передача объектного файла на обработку компоновщику . ………….... . . 10

4.8  Передача объектного файла на обработку компоновщику . ………….. . . . 10

4.9  Запуск исполняемого файла.............................................................................. 11

4.10 Создание копии файла...................................................................................... 11

4.11 Изменение программы ..................................................................................... 11

4.12 Компиляция текста программы....................................................................... 12

4.13 Передача объектного файла на обработку компоновщику . . . ………….. . 12

4.14 Запуск исполняемого файла............................................................................. 12

4.15 Создании копии файлов в новом каталоге . . …………………. . . . . . . . . . . 12

4.16 Удаление лишних файлов в текущем каталоге . . . . ……………….. . . . . . . 13

4.17 Добавление файлов на GitHub.......................................................................... 13

4.18 Отправка файлов................................................................................................ 13

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

1. **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.

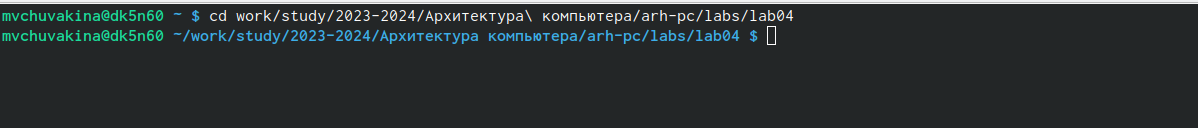
В основе вычислительного процесса ЭВМ лежит принцип программного управления. Это означает, что компьютер решает поставленную задачу как последовательность действий, записанных в виде программы.

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.

1. **Выполнение лабораторной работы**

**4.1 Создание программы Hello world!**

С помощью утилиты cd перемещаюсь в каталог, в котором буду работать (рис. 4.1).

 и Рис. 4.1: Перемещение между директориями

Создаю в текущем каталоге пустой текстовый файл hello.asm с помощью утилиты touch (рис. 4.2).

 и Рис. 4.2: Создание пустого файла  
 Открываю созданный файл в текстовом редакторе gedit (рис. 4.3).

Изображение выглядит как компьютер, текст, снимок экрана, программное обеспечение

Автоматически созданное описание b. b Рис. 4.3: Открытие файла в текстовом редакторе

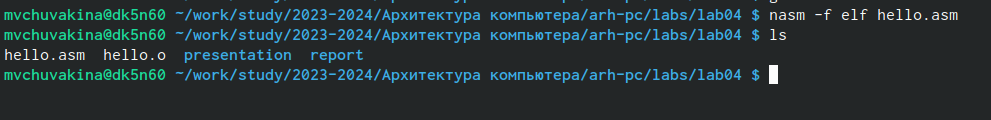
Заполняю файл, вставляя в него программу для вывода “Hello word!” (рис. 4.4).

Изображение выглядит как текст, снимок экрана, компьютер, программное обеспечение

Автоматически созданное описание b b Рис. 4.4: Заполнение файла

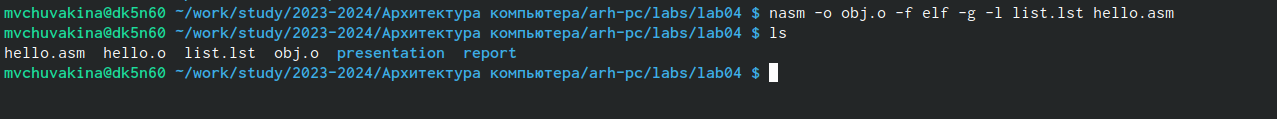
**4.2 Работа с транслятором NASM**

Превращаю текст программы для вывода “Hello world!” в объектный код с помощью транслятора NASM, используя команду nasm -f elf hello.asm, ключ -f указывает транслятору nasm, что требуется создать бинарный файл в формате ELF (рис. 4.5). Далее проверяю правильность выполнения команды с помощью утилиты ls: действительно, создан файл “hello.o”.

 b Рис. 4.5: Компиляция текста программы

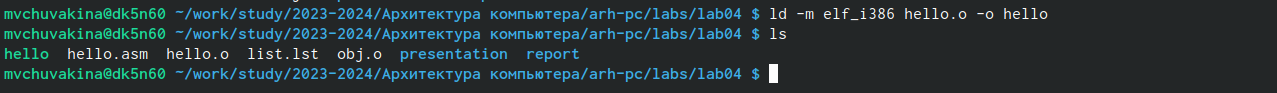
**4.3 Работа с расширенным синтаксисом командной строки NASM**

Ввожу команду, которая скомпилирует файл hello.asm в файл obj.o, при этом в файл будут включены символы для отладки (ключ -g), также с помощью ключа -l будет создан файл листинга list.lst (рис. 4.6). Далее проверяю с помощью утилиты ls правильность выполнения команды.

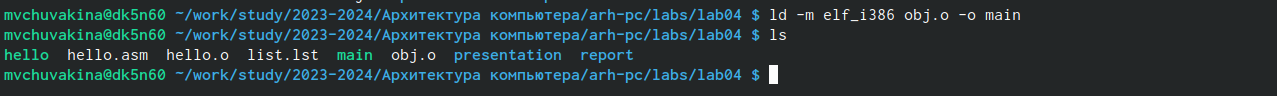
 b Рис. 4.6: Компиляция текста программы

**4.4 Работа с компоновщиком LD**

Передаю объектный файл hello.o на обработку компоновщику LD, чтобы по- лучить исполняемый файл hello (рис. 4.7). Ключ -о задает имя создаваемого исполняемого файла. Далее проверяю с помощью утилиты ls правильность выполнения команды.

 b Рис. 4.7: Передача объектного файла на обработку компоновщику

Выполняю следующую команду (рис. 4.8). Исполняемый файл будет иметь имя main, т.к. после ключа -о было задано значение main. Объектный файл, из которого собран этот исполняемый файл, имеет имя obj.o

 b Рис. 4.8: Передача объектного файла на обработку компоновщику

**4.5 Запуск исполняемого файла**

Запускаю на выполнение созданный исполняемый файл hello (рис. 4.9).

 n Рис. 4.9: Запуск исполняемого файла

**4.6 Выполнение заданий для самостоятельной работы.**

С помощью утилиты cp создаю в текущем каталоге копию файла hello.asm с именем lab4.asm (рис. 4.10).

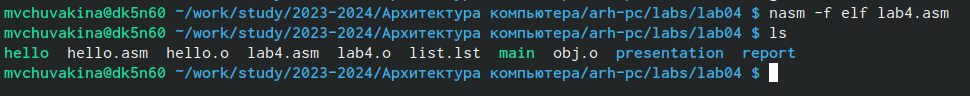
 b Рис. 4.10: Создание копии файла

С помощью текстового редактора gedit открываю файл lab4.asm и вношу изменения в программу так, чтобы она выводила мои имя и фамилию. (рис. 4.11).

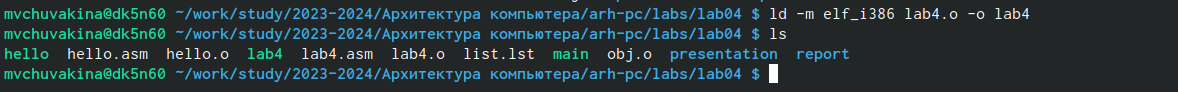
Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание n n Рис. 4.11: Изменение программы

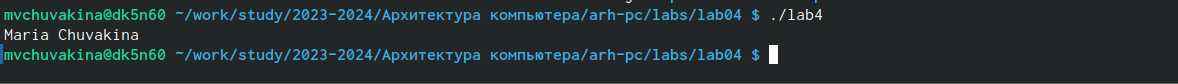
Компилирую текст программы в объектный файл (рис. 4.12). Проверяю с помощью утилиты ls, что файл lab4.o создан.

 b Рис. 4.12: Компиляция текста программы

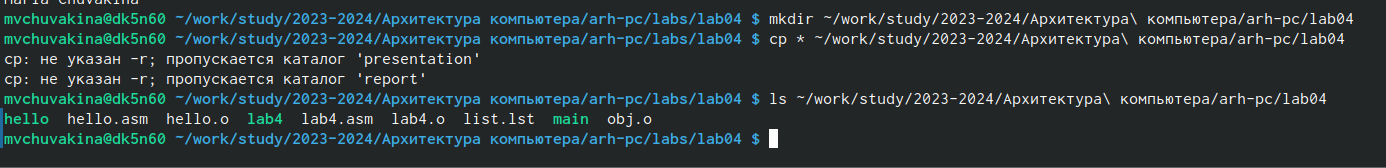
Передаю объектный файл lab4.o на обработку компоновщику LD, чтобы получить исполняемый файл lab4 (рис. 4.13).

 b Рис. 4.13: Передача объектного файла на обработку компоновщику

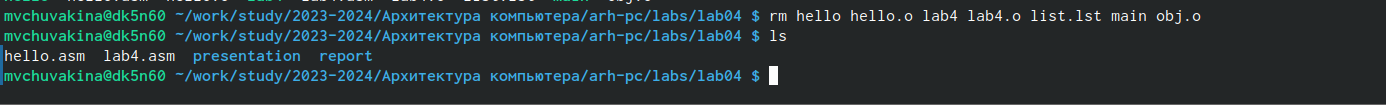
Запускаю исполняемый файл lab4, на экран действительно выводятся мои имя и фамилия (рис. 4.14).

 b Рис. 4.14: Запуск исполняемого файла

К сожалению, я начала работу не в том каталоге, поэтому создаю другую директорию lab04 с помощью mkdir, прописывая полный путь к каталогу, в котором хочу создать эту директорию. Далее копирую из текущего каталога файлы, созданные в процессе выполнения лабораторной работы, с помощью утилиты cp, указывая вместо имени файла символ \*, чтобы скопировать все файлы. Команда проигнорирует директории в этом каталоге, т. к. не указан ключ -r, это мне и нужно (рис. 4.15). Проверяю с помощью утилиты ls правильность выполнения команды.

 b Рис. 4.15: Создании копии файлов в новом каталоге

Удаляю лишние файлы в текущем каталоге с помощью утилиты rm, ведь копии файлов остались в другой директории (рис. 4.16).

 b Рис. 4.16: Удаление лишних файлов в текущем каталоге

С помощью команд git add . и git commit добавляю файлы на GitHub, ком- ментируя действие как добавление файлов для лабораторной работы №4 (рис. 4.17). Изображение выглядит как текст, снимок экрана, Шрифт, Мультимедийное программное обеспечение

Автоматически созданное описание и Рис. 4.17: Добавление файлов на GitHub

Отправляю файлы на сервер с помощью команды git push (рис. 4.18).

Изображение выглядит как текст, программное обеспечение, Шрифт, Мультимедийное программное обеспечение

Автоматически созданное описание т Рис. 4.18: Отправка файлов

1. **Выводы**

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

**6. Список литературы**

1.https://esystem.rudn.ru/pluginfile.php/1584628/mod\_resource/content/1/%D0%9B%D0%B0%D