

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: Лисенков Е.Р.

Группа: НКАбд-03-23

МОСКВА

2023 г.

Оглавление

1 Цель работы	4
2 Задание	5
3 Теоретическое введение	6
4 Выполнение лабораторной работы	8
4.2 Транслятор NASM	9
4.3 Работа с расширенным синтаксисом командной строки NASM	9
4.4 Работа с компоновщиком LD	10
4.5 Запуск программы	10
4.6 Задания для проверки	10
Выводы	13
Список литературы	14

Список иллюстраций

Изображение 1	Создание каталога.....	8
Изображение 2	Переход в каталог.....	8
Изображение 3	Создание файла.....	8
Изображение 4	Переход в файл.....	8
Изображение 5	Редактирование файла.....	9
Изображение 6	Компиляция текста программы.....	9
Изображение 7	Компиляция текста программы.....	10
Изображение 8	Передача объектного файла на обработку компоновщику.....	10
Изображение 9	Передача объектного файла на обработку компоновщику.....	10
Изображение 10	Запуск программы.....	10
Изображение 11	Создание копии файла с другим именем.....	11
Изображение 12	Редактирование файла.....	11
Изображение 13	Компиляция текста в объектный файл.....	11
Изображение 14	Передача компоновщику.....	11
Изображение 15	Успешный запуск программы.....	11
Изображение 16	Отправка файлов на GitHub.....	12

1 Цель работы

Главной целью данной работы является приобретение навыков работы с языком программирования ассемблер 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.

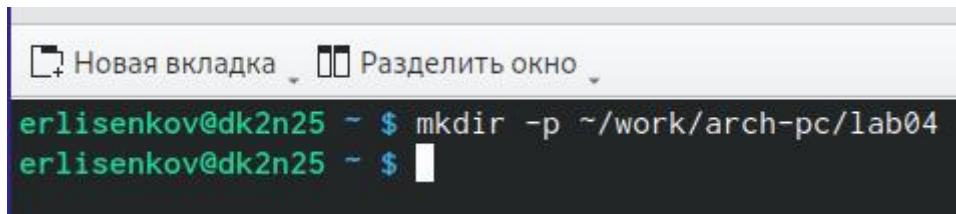
3 Теоретическое введение

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 6 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 7 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

Создаю каталог для работы с программами, с помощью команды `mkdir -p ~/work/arch-pc/lab04`. (Из. 1)



```
erlisenkov@dk2n25 ~ $ mkdir -p ~/work/arch-pc/lab04
erlisenkov@dk2n25 ~ $
```

Изображение 1 Создание каталога.

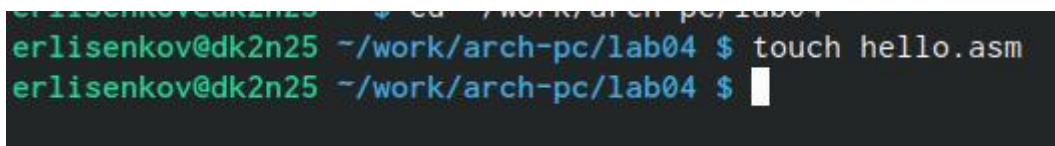
Выполню переход в созданный каталог. (Из. 2)



```
erlisenkov@dk2n25 ~ $ mkdir -p ~/work/arch-pc/lab04
erlisenkov@dk2n25 ~ $ cd ~/work/arch-pc/lab04
erlisenkov@dk2n25 ~/work/arch-pc/lab04 $
```

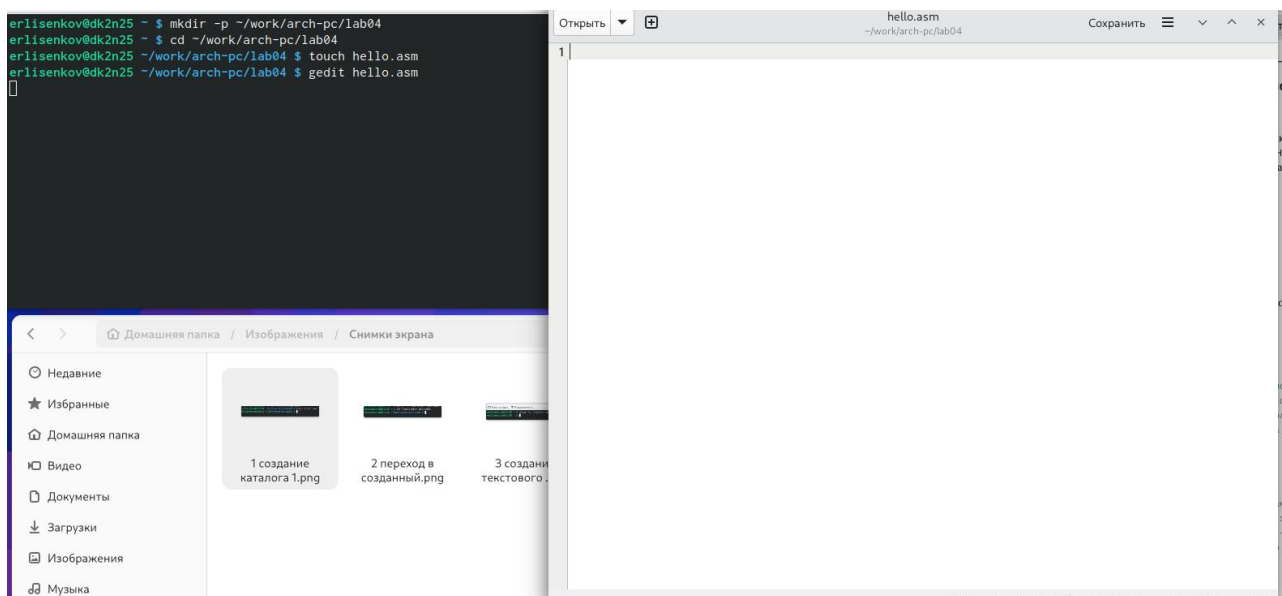
Изображение 2 Переход в каталог.

Создам файл `hello.asm` и затем открою его. (Из. 3) (Из. 4)



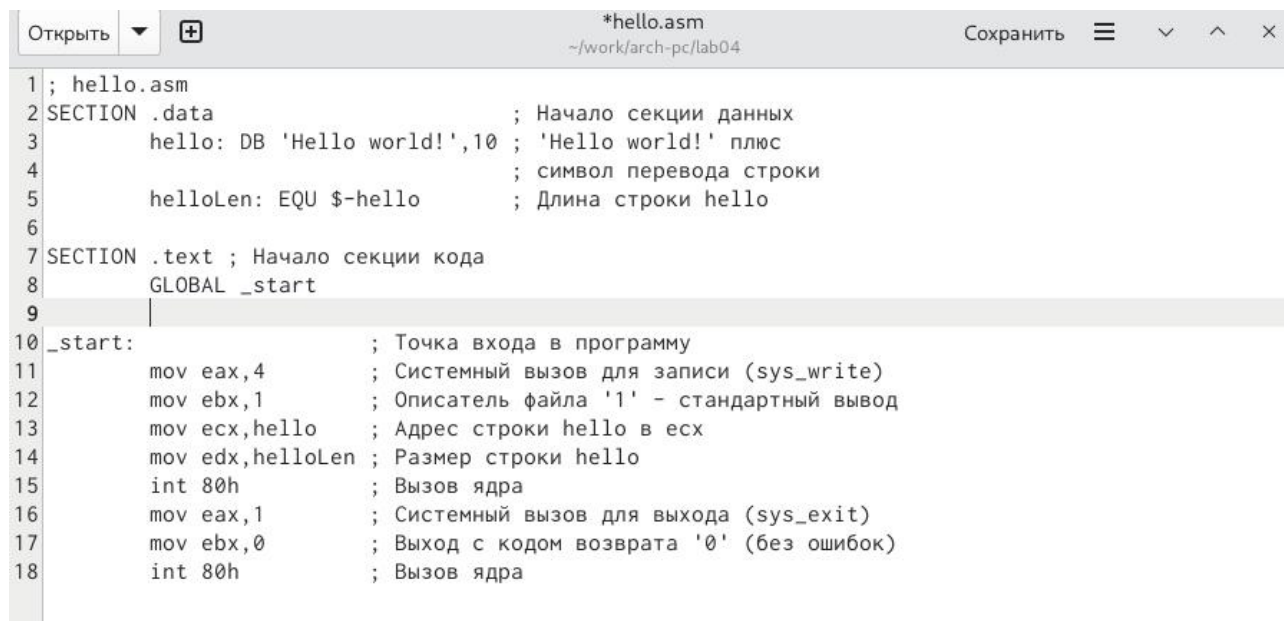
```
erlisenkov@dk2n25 ~ $ cd ~/work/arch-pc/lab04
erlisenkov@dk2n25 ~/work/arch-pc/lab04 $ touch hello.asm
erlisenkov@dk2n25 ~/work/arch-pc/lab04 $
```

Изображение 3 Создание файла



Изображение 4 Переход в файл

Выполню редактирование и напишу код для языка Assembler NASM.
(Из. 5)

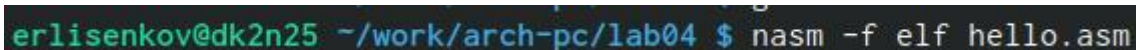


```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4             ; символ перевода строки
5     helloLen: EQU $-hello ; Длина строки hello
6
7 SECTION .text ; Начало секции кода
8     GLOBAL _start
9
10 _start: ; Точка входа в программу
11     mov eax,4 ; Системный вызов для записи (sys_write)
12     mov ebx,1 ; Описатель файла '1' - стандартный вывод
13     mov ecx,hello ; Адрес строки hello в ecx
14     mov edx,helloLen ; Размер строки hello
15     int 80h ; Вызов ядра
16     mov eax,1 ; Системный вызов для выхода (sys_exit)
17     mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
18     int 80h ; Вызов ядра
```

Изображение 5 Редактирование файла

4.2 Транслятор 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 (Из. 6).



```
erlisenkov@dk2n25 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
```

Изображение 6 Компиляция текста программы

4.3 Работа с расширенным синтаксисом командной строки NASM

Ввожу команду для компиляции файла из `hello.asm` в `obj.o`, также там будут содержаться ключи отладки и с помощью ключа `-l` создам файл листинга `list.lst` (Из. 7). Также для собственного удобства работы с программой я перенёс все файлы в `~/work/study/2023-2024/”Архитектура компьютера”/arch-pc/labs/lab04`.

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o  presentation  report
```

Изображение 7 Компиляция текста программы

4.4 Работа с компоновщиком LD

Передаю объектный файл hello.o на обработку компоновщику LD, чтобы получить исполняемый файл hello (Из. 8).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 hello.o -o hello
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o  presentation  report
```

Изображение 8 Передача объектного файла на обработку компоновщику.

Выполняю команду далее, которая создаст исполняемый файл main, т.к. После ключа -o было задано значение main. Теперь же объектный файл имеет имя obj.o (Из. 9).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 obj.o -o main
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  list.lst  main  obj.o  presentation  report
```

Изображение 9 Передача объектного файла на обработку компоновщику.

4.5 Запуск программы

Произвожу запуск файла, который исполнит код (Из. 10).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ./hello
Hello world!
```

Изображение 10 Запуск программы

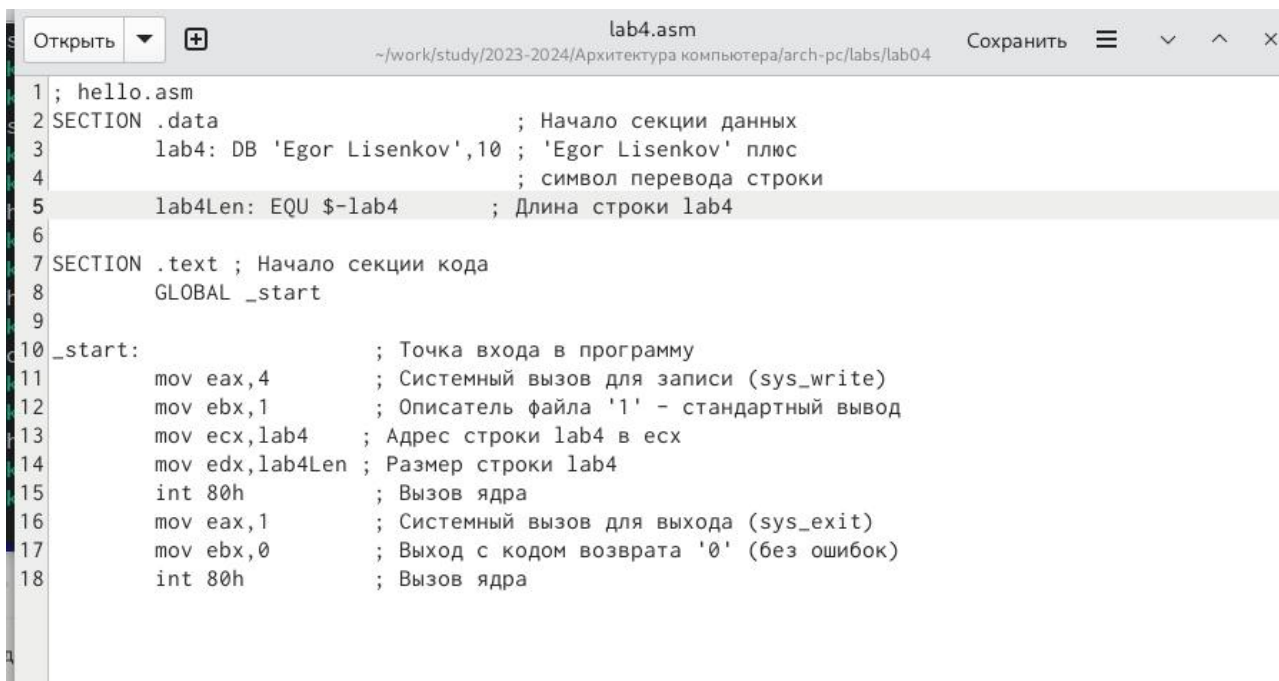
4.6 Задания для проверки

Создам копию в текущем файл из файла с именем hello.asm в lab4.asm (Из. 11).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ cp hello.asm lab4.asm
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o  presentation  report
```

Изображение 11 Создание копии файла с другим именем

Далее с помощью текстового редактора меняю данные в файле lab4.asm.



```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     lab4: DB 'Egor Lisenkov',10 ; 'Egor Lisenkov' плюс
4           ; символ перевода строки
5     lab4Len: EQU $-lab4 ; Длина строки lab4
6
7 SECTION .text ; Начало секции кода
8     GLOBAL _start
9
10 _start: ; Точка входа в программу
11     mov eax,4 ; Системный вызов для записи (sys_write)
12     mov ebx,1 ; Описатель файла '1' - стандартный вывод
13     mov ecx,lab4 ; Адрес строки lab4 в ecx
14     mov edx,lab4Len ; Размер строки lab4
15     int 80h ; Вызов ядра
16     mov eax,1 ; Системный вызов для выхода (sys_exit)
17     mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
18     int 80h ; Вызов ядра
```

Изображение 12 Редактирование файла

Далее компилирую созданный файл в объектный файл (Из. 13).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf lab4.asm
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  lab4.o  list.lst  main  obj.o  presentation  report
```

Изображение 13 Компиляция текста в объектный файл

Выполню передачу файла объектного файла lab4.o на обработку компоновщика LD (Из. 14).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 lab4.o -o lab4
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.o  list.lst  main  obj.o  presentation  report
```

Изображение 14 Передача компоновщику

Выполню запуск программы (Из. 15).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ ./lab4
Egor Lisenkov
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Изображение 15 Успешный запуск программы

После всей проделанной работы выполню отправку файлов в собственный репозиторий GitHub (Из. 16).

```
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git add .
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git commit -m "LAB 4"
[master da96d89] LAB 4
 2 files changed, 36 insertions(+)
 create mode 100644 labs/lab04/hello.asm
 create mode 100644 labs/lab04/lab4.asm
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 1.04 КиБ | 1.04 МиБ/с, готово.
Всего 6 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:erlisenkov/study_2023-2024_arh-pc.git
 65a38cc..da96d89  master -> master
erlisenkov@dk2n25 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $
```

Изображение 16 Отправка файлов на GitHub

Выводы

Данное занятие помогло мне понять принцип работы с языком программирования Assembler NASM.

Список литературы

https://esystem.rudn.ru/pluginfile.php/2089084/mod_resource/content/0/Лабораторная%20работа%20№4.%20Создание%20и%20процесс%20обработки%20программ%20на%20языке%20ассемблера%20NASM.pdf