

# **Лабораторная работа №4**

**Архитектура вычислительных систем**

Чулкова Валерия

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>5</b>	<b>Выполнение самостоятельной работы</b>	<b>10</b>
<b>6</b>	<b>Выводы</b>	<b>13</b>
	<b>Список литературы</b>	<b>14</b>

## Список иллюстраций

4.1	создание каталога . . . . .	7
4.2	gedit . . . . .	7
4.3	файл hello.asm . . . . .	7
4.4	успешная компиляция . . . . .	8
4.5	транслятор . . . . .	8
4.6	ged it report.md . . . . .	9
4.7	картинки . . . . .	9
4.8	файл . . . . .	9
5.1	самостоятельная работа.png . . . . .	10
5.2	самостоятельная работа.png . . . . .	10
5.3	самостоятельная работа.png . . . . .	11
5.4	самостоятельная работа.png . . . . .	11
5.5	самостоятельная работа.png . . . . .	11
5.6	самостоятельная работа.png . . . . .	12

# 1 Цель работы

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

## 2 Задание

1. В соответствующем каталоге сделайте отчёт по лабораторной работе №4 в формате Markdown. В качестве отчёта необходимо предоставить отчёты в 3 форматах: pdf, docx и md.
2. Загрузите файлы на github.

### 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы. Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую систему
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно об Unix см. в [1–6].

## 4 Выполнение лабораторной работы

- 1) Создаём каталог для работы с программами на языке ассемблера NASM:

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

Рис. 4.1: создание каталога

- 2) Создаём текстовый файл с именем hello.asm и открываем этот файл с помощью любого текстового редактора gedit:

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

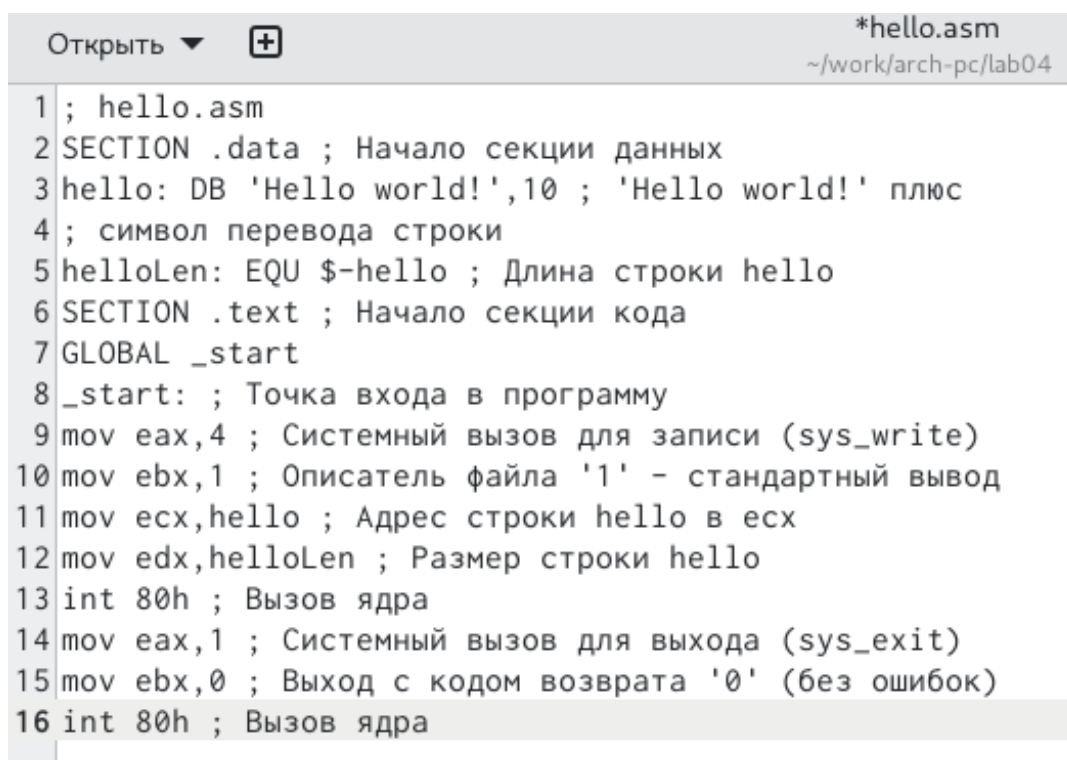
Рис. 4.2: gedit

- 3) Вводим в него следующий текст:

```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 4.3: файл hello.asm

4) NASM превращает текст программы в объектный код. Например, для компиляции приведённого выше текста программы «Hello World» необходимо написать следующее



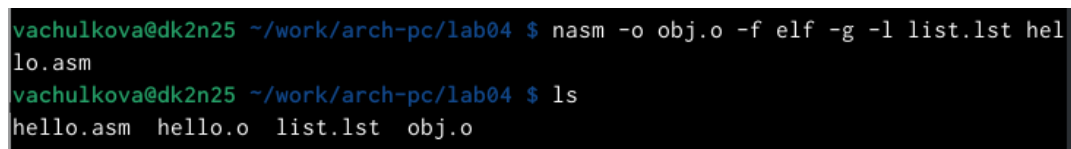
```
Открыть ▾ + *hello.asm
~/work/arch-pc/lab04
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5 helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для записи (sys_write)
10 mov ebx,1 ; Описатель файла '1' - стандартный вывод
11 mov ecx,hello ; Адрес строки hello в ecx
12 mov edx,helloLen ; Размер строки hello
13 int 80h ; Вызов ядра
14 mov eax,1 ; Системный вызов для выхода (sys_exit)
15 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16 int 80h ; Вызов ядра
```

Рис. 4.4: успешная компиляция

Т. к. текст программы набран без ошибок, транслятор преобразует текст программы из файла hello.asm в объектный код, который записан в файл hello.o.

- 5) С помощью команды ls проверим, что объектный файл был создан. У нас есть два файла hello.asm и hello.o.

Следующая команда скомпилирует исходный файл hello.asm в obj.o , при этом формат выходного файла будет elf, и в него будут включены символы для отладки (опция -g), кроме того, создается файл листинга list.lst .Выполним следующую команду:



```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
```

Рис. 4.5: транслятор



6) Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику, а потом с командой `ls` проверим содержимое:

```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис. 4.6: `ged it report.md`

7) Ключ `-o` с последующим значением задаст в данном случае имя создаваемого исполняемого файла. Выполним следующую команду

Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику, а потом с командой `ls` проверим содержимое:

```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ls
```

Рис. 4.7: картинки

11) Запустим на выполнение созданный исполняемый файл, находящийся в текущем каталоге, набрав в командной строке `./hello`:

```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
```

Рис. 4.8: файл

## 5 Выполнение самостоятельной работы

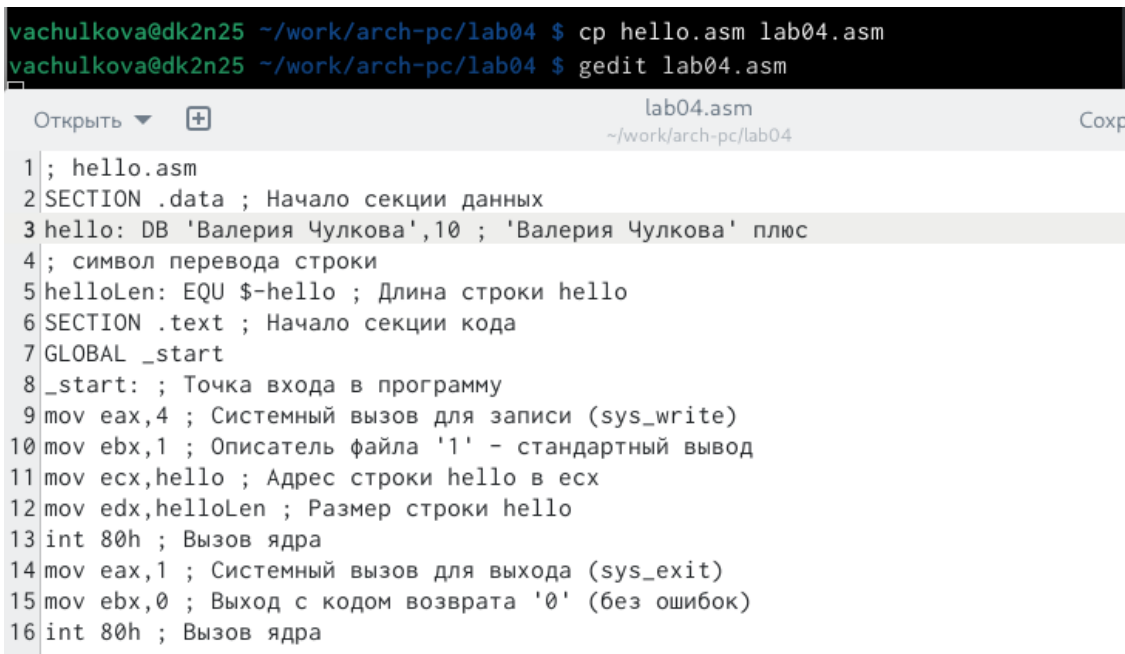
- 1) В каталоге ~/work/arch-pc/lab04 с помощью команды `cp` создали копию файла `hello.asm` с именем `lab04.asm`.



```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

Рис. 5.1: самостоятельная работа.png

С помощью текстового редактора вносим изменения в текст программы в файле `lab04.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с фамилией и именем. Для этого вместо “`Hello world`” пишем своё имя.



```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ cp hello.asm lab04.asm
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ gedit lab04.asm
```

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

Рис. 5.2: самостоятельная работа.png

Проводим схожие действия с лабораторной работой, но изменяем название файлов.

```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ nasm -f elf lab04.asm
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ nasm -o obj1.o -f elf -g -l list1.lst lab04.asm
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab04.o -o lab04
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ gedit lab04.asm
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ls
hello      hello.o  lab04.asm  list1.lst  main      obj.o
hello.asm  lab04    lab04.o    list1.lst  obj1.o
```

Рис. 5.3: самостоятельная работа.png

- 3) Оттранслируем полученный текст программы lab04.asm в объектный файл и запустим, получим вывод фамилии и имени.

```
vachulkova@dk2n25 ~/work/arch-pc/lab04 $ ./lab04
Валерия Чулкова
```

Рис. 5.4: самостоятельная работа.png

Переносим файлы в основную папку lab04:

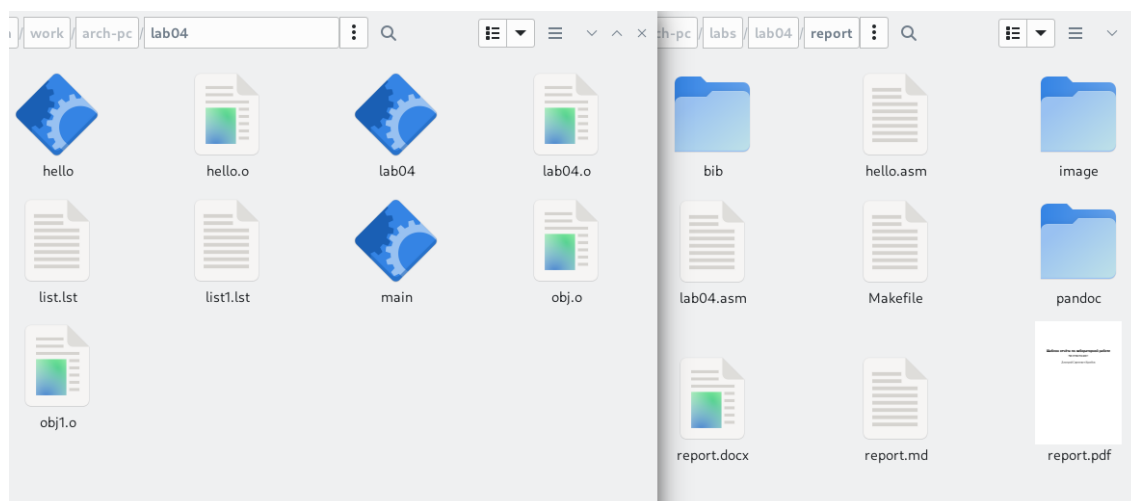
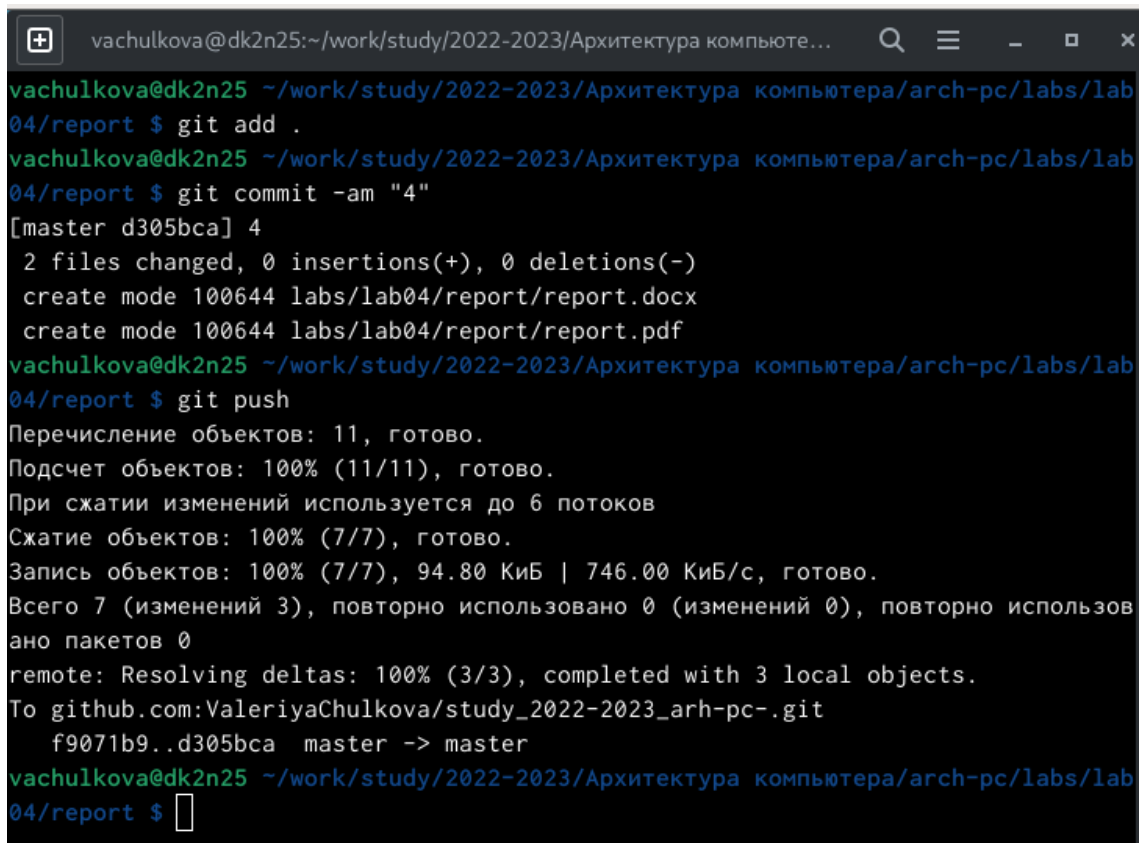


Рис. 5.5: самостоятельная работа.png

- 4) Загружаем файлы на GitHub при помощи команд.

A terminal window with a dark background and light-colored text. The window title bar shows the user 'vachulkova@dk2n25' and the current directory. The terminal displays a sequence of git commands: 'git add .', 'git commit -am "4"', and 'git push'. The output shows the commit of two files (report.docx and report.pdf) and the successful push to the remote repository. The terminal ends with a prompt character.

```
vachulkova@dk2n25 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $ git add .
vachulkova@dk2n25 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $ git commit -am "4"
[master d305bca] 4
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 labs/lab04/report/report.docx
 create mode 100644 labs/lab04/report/report.pdf
vachulkova@dk2n25 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $ git push
Перечисление объектов: 11, готово.
Подсчет объектов: 100% (11/11), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (7/7), готово.
Запись объектов: 100% (7/7), 94.80 КиБ | 746.00 КиБ/с, готово.
Всего 7 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:ValeriyaChulkova/study_2022-2023_arh-pc-.git
   f9071b9..d305bca  master -> master
vachulkova@dk2n25 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $
```

Рис. 5.6: самостоятельная работа.png

## 6 Выводы

Я освоил процедуру компиляции и сборки программ, написанных на ассемблере NASM.

## Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016.  
URL: <https://www.gnu.org/software/bash/manual/>.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.