

Лабораторная работа №4

Архитектура вычислительных систем

Кавказова Диана Алексеевна

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
5	Выполнение самостоятельной работы	12
6	Выводы	15
	Список литературы	16

Список иллюстраций

4.1	создание каталога	7
4.2	файл hello.asm	7
4.3	gedit	7
4.4	файл	8
4.5	успешная компиляция	8
4.6	транслятор	9
4.7	ged it report.md	9
4.8	картинки	9
4.9	ls	10
4.10	-hf	10
4.11	ls	10
4.12	Ключ -o	11
4.13	файл	11
5.1	111.png	12
5.2	222.png	12
5.3	223.png	13
5.4	333.png	13
5.5	0.png	14

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
/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:

```
dakavkazova@dk5n55 ~/work $ mkdir arch-pc
dakavkazova@dk5n55 ~/work $ cd arch-pc
dakavkazova@dk5n55 ~/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

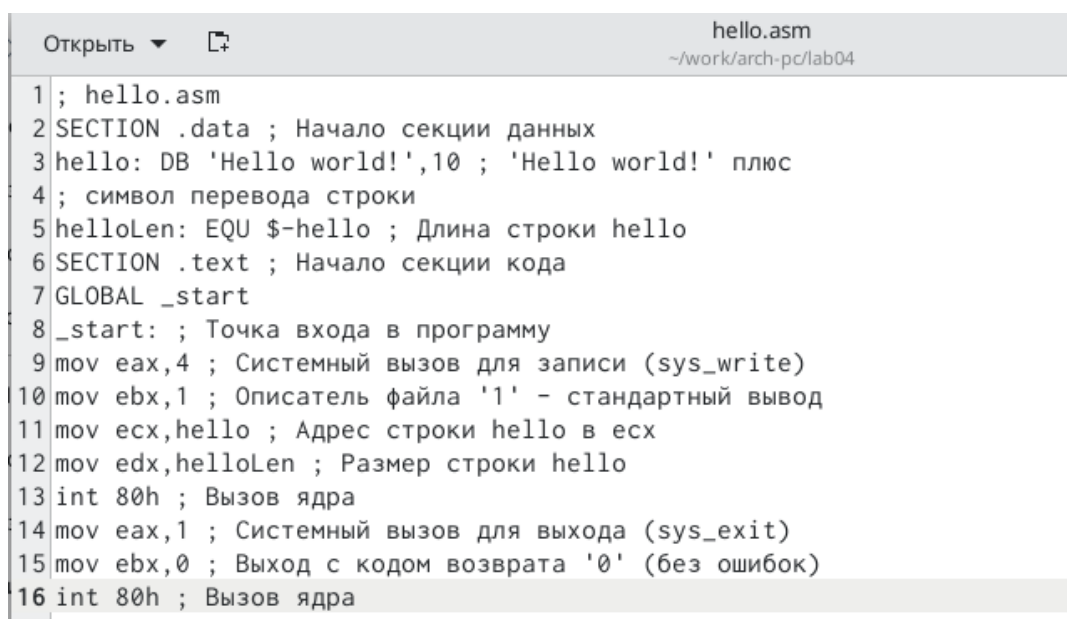
```
dakavkazova@dk5n55 ~/work/arch-pc $ cd lab04
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ touch hello.asm
```

Рис. 4.2: файл hello.asm

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ gedit hello.asm
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
```

Рис. 4.3: gedit

- 3) Вводим в него следующий текст:



```
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: файл

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



```
dakavkazova@dk5n55 ~/.work/arch-pc/lab04 $ nasm -f elf hello.asm
dakavkazova@dk5n55 ~/.work/arch-pc/lab04 $ ls
```

Рис. 4.5: успешная компиляция

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

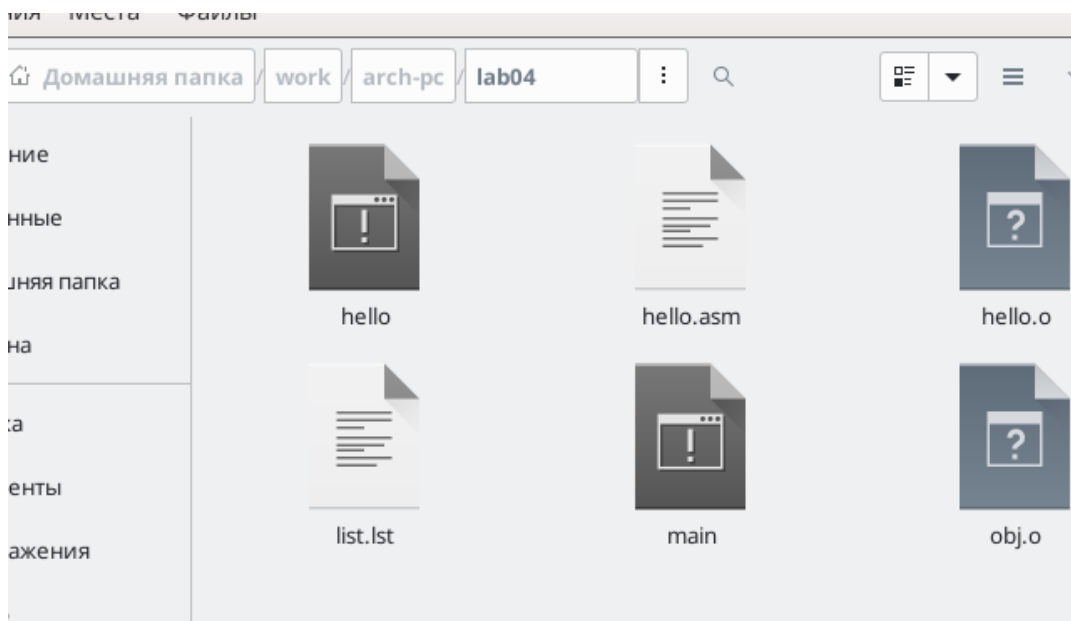


Рис. 4.6: транслятор

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

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o
```

Рис. 4.7: ged it report.md

- 7) 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:

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
```

Рис. 4.8: картинки

- 8) С помощью команды `ls` проверим, что файлы были созданы:

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
```

Рис. 4.9: ls

10) Для получения списка форматов объектного файла смотрим `nasm -hf`.

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ nasm -hf
Usage: nasm [-@ response_file] [options...] [--] filename
        nasm -v (or --v)

Options (values in brackets indicate defaults):

    -h            show this text and exit (also --help)
    -v (or --v)   print the NASM version number and exit
    -@ file       response file; one command line option per line

    -o outfile    write output to outfile
    --keep-all   output files will not be removed even if an error happens

    -Xformat      specify error reporting format (gnu or vc)
    -s           redirect error messages to stdout
    -Zfile        redirect error messages to file

    -M           generate Makefile dependencies on stdout
    -MG          d:o, missing files assumed generated
    -MF file      set Makefile dependency file
    -MD file      assemble and generate dependencies
    -MT file      dependency target name
    -MQ file      dependency target name (quoted)
    -MP          emit phony targets
```

Рис. 4.10: -hf

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

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst obj.o
```

Рис. 4.11: ls

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

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
```

Рис. 4.12: Ключ -o

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

```
dakavkazova@dk5n55 ~/work/arch-pc/lab04 $ ./hello  
Hello world!
```

Рис. 4.13: файл

5 Выполнение самостоятельной работы

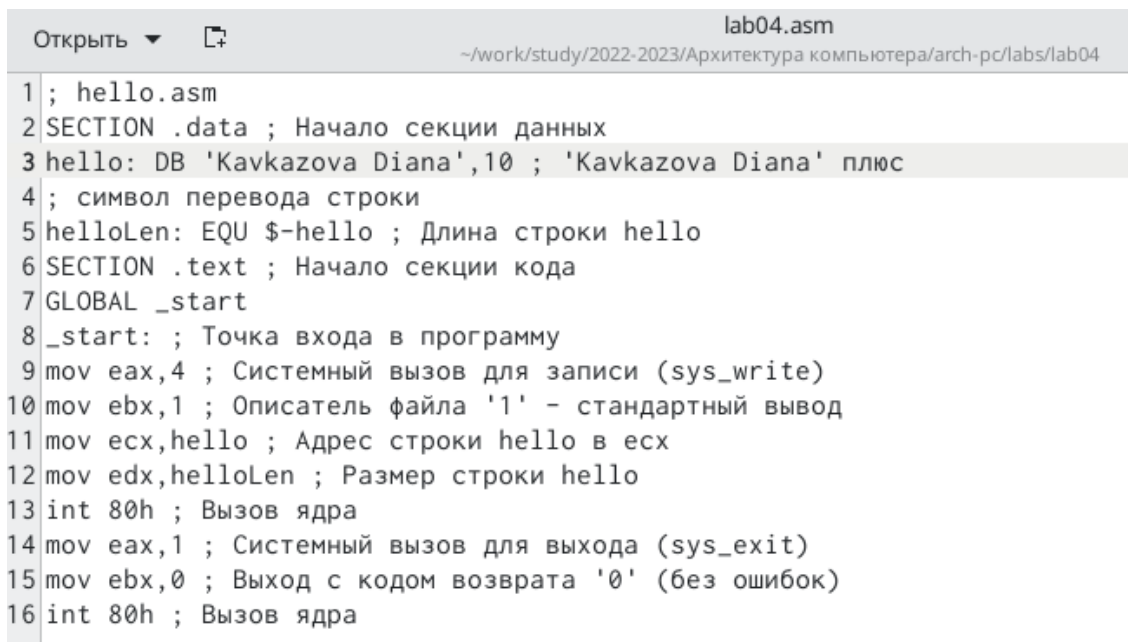
- 1) В каталоге ~/work/arch-pc/lab04 с помощью команды `cp` создали копию файла `hello.asm` с именем `lab04.asm`.



```
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ cp hello.asm lab04.asm
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 5.1: 111.png

- 2) С помощью текстового редактора внесли 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” пишу “Kavkazova Diana”.



```
Открыть ▾  lab04.asm
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04

1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Kavkazova Diana',10 ; 'Kavkazova Diana' 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: 222.png

```
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf -g -l list.lst lab04.asm
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 lab04.o -o lab05
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ gedit lab04.asm
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf lab04.asm
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello hello.asm hello.o lab04.asm lab04.o list.lst main obj.o presentation report
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o obj1.o -f elf -g -l list1.lst lab04.asm
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello hello.asm hello.o lab04.asm lab04.o list1.lst list.lst main obj1.o obj.o presentation report
```

Рис. 5.3: 223.png

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

```
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ./hello
Kavkazova Diana
dakavkazova@dk5n55 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ █
```

Рис. 5.4: 333.png

- 4) Загружаю файлы на GitHub при помощи команд “git add .”, “git commit -am” “,”git push”.

```

dakavkazova@dk8n76 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $ git add .
dakavkazova@dk8n76 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $ git commit -am "4"
[master 3a67ade] 4
21 files changed, 129 insertions(+), 17 deletions(-)
delete mode 100644 labs/lab02/report/report.pdf
create mode 100644 labs/lab04/report/hello.asm
create mode 100644 labs/lab04/report/image/1.png
create mode 100644 labs/lab04/report/image/10.png
create mode 100644 labs/lab04/report/image/11.png
create mode 100644 labs/lab04/report/image/111.png
create mode 100644 labs/lab04/report/image/12.png
create mode 100644 labs/lab04/report/image/13.png
create mode 100644 labs/lab04/report/image/2.png
create mode 100644 labs/lab04/report/image/222.png
create mode 100644 labs/lab04/report/image/223.png
create mode 100644 labs/lab04/report/image/3.png
create mode 100644 labs/lab04/report/image/333.png
create mode 100644 labs/lab04/report/image/4.png
create mode 100644 labs/lab04/report/image/5.png
create mode 100644 labs/lab04/report/image/6.png
create mode 100644 labs/lab04/report/image/7.png
create mode 100644 labs/lab04/report/image/8.png
create mode 100644 labs/lab04/report/image/9.png
create mode 100644 labs/lab04/report/lab04.asm
dakavkazova@dk8n76 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04/report $ git push
Перечисление объектов: 36, готово.
Подсчет объектов: 100% (36/36), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (28/28), готово.
Запись объектов: 100% (28/28), 443.94 КиБ | 10.57 МиБ/с, готово.
Всего 28 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To github.com:Roman11tz/study_2022-2023_arh-pc.git
f513922..3a67ade master -> master

```

Рис. 5.5: 0.png

6 Выводы

Я освоила процедуру компиляции и сборки программ, написанных на ассемблере NASM (рассмотрела пример простой программы, научилась изменять содержимое файла, приобрела навык по созданию объектных файлов).

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.