

Шаблон отчёта по лабораторной работе

4

Талебу тенке франк устон , НКАбд-05-23

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы :	6
2.1	Транслятор NASM :	7
2.1.1	Расширенный синтаксис командной строки NASM :	8
3	Теоретическое введение	10
4	Задание для самостоятельной работы :	11
4.1	Выводы по результатам выполнения заданий :	13
5	Выводы	14
	Список литературы	15

Список иллюстраций

2.1	Ресукнек 1	6
2.2	Ресукнек 2	6
2.3	Ресукнек 3	7
2.4	Ресукнек 4	7
2.5	Ресукнек 5	8
2.6	Ресукнек 6	8
2.7	Ресукнек 8	9
4.1	Ресукнек 9	11
4.2	Ресукнек 10	12
4.3	Ресукнек 11	12
4.4	Ресукнек 12	13
4.5	Ресукнек 13	13

Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . .	10
-----	---	----

1 Цель работы

- В пятой лабораторной работе мы рассмотрим, как освоить процедуру компиляции и сборки программ, написанных на ассемблере `nasm`.

2 Выполнение лабораторной работы :

- В этом разделе мы хотели создать программу, которая выводит строку “Hello world!” но на языке ассемблера nasm.
- Вот почему мы начали с рекурсивного создания нового каталога “~/work/arch-pc/lab05”.(рис. [2.1])

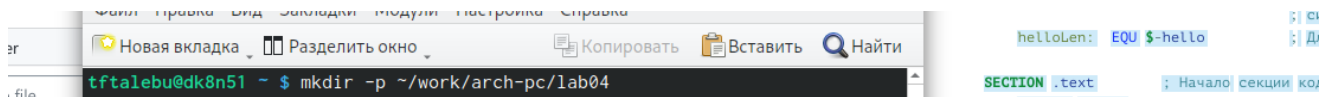


Рис. 2.1: Ресункек 1

- После этого мы создали текстовый файл в формате .asm, затем открываем только что созданный файл с помощью текстового редактора gedit.(рис. [2.2])

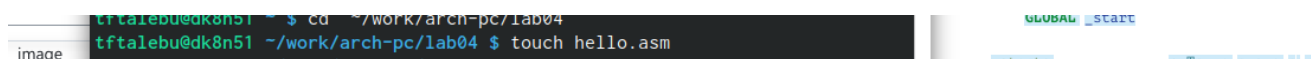
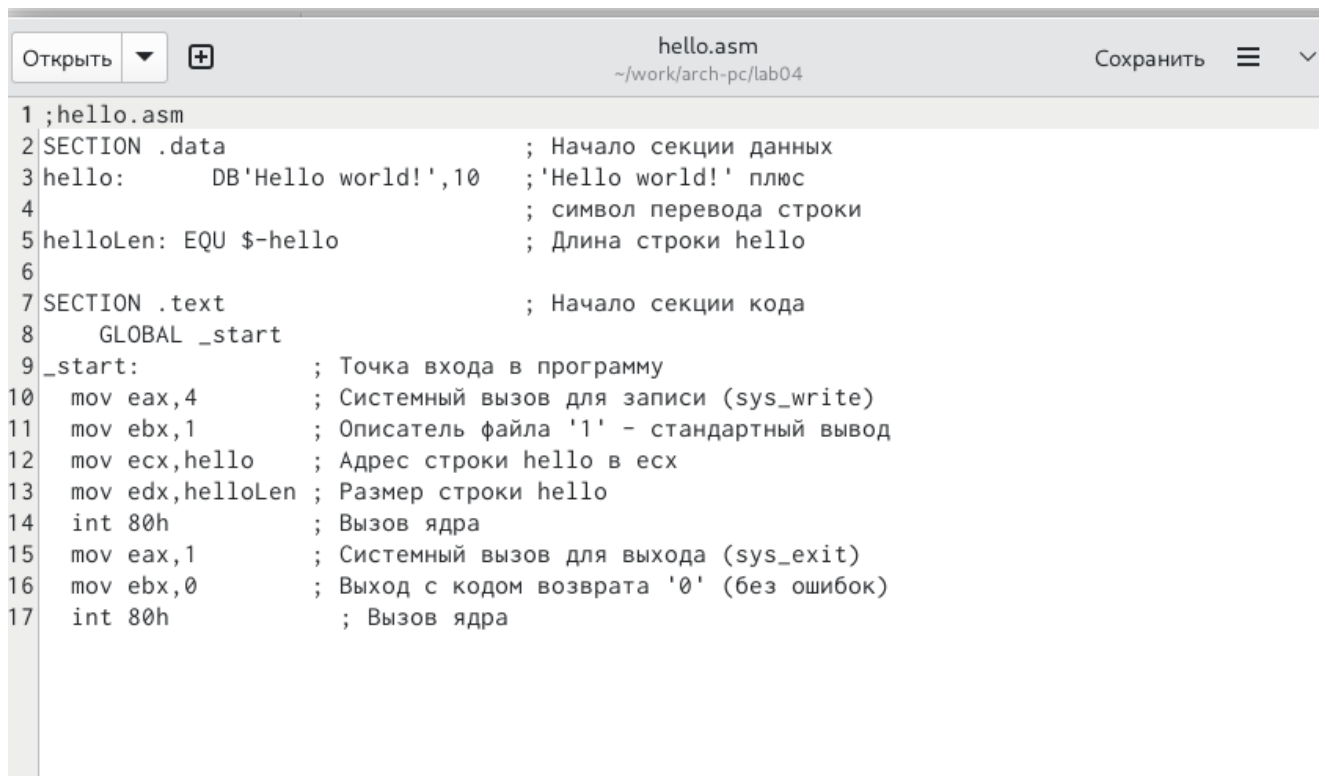


Рис. 2.2: Ресункек 2

- После этого мы добавили код сборки, который выводит “Hello world!” в файл hello.asm. (рис. [2.3])



```
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 _start:                    ; Точка входа в программу
10  mov eax,4                ; Системный вызов для записи (sys_write)
11  mov ebx,1                ; Описатель файла '1' - стандартный вывод
12  mov ecx,hello            ; Адрес строки hello в ecx
13  mov edx,helloLen         ; Размер строки hello
14  int 80h                  ; Вызов ядра
15  mov eax,1                ; Системный вызов для выхода (sys_exit)
16  mov ebx,0                ; Выход с кодом возврата '0' (без ошибок)
17  int 80h                  ; Вызов ядра
```

Рис. 2.3: Ресункек 3

2.1 Транслятор NASM :

- На этом этапе, используя переводчик NASM, мы смогли скомпилировать или перевести код в объектный код, который создал другой файл с форматом **.o..**(рис. [2.4])



```
+ Остановлен gedit hello.asm
alebu@dk8n51 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
alebu@dk8n51 ~/work/arch-pc/lab04 $ ls
```

```
ld -m elf_i386 obj.o -o main
```

Какое имя будет иметь исполняемый файл?
собирается этот исполняемый файл?

Рис. 2.4: Ресункек 4

- Используя команду **ls**, мы проверили работу, проделанную переводчиком, и обнаружили, что объектный файл был создан с тем же именем, что и текстовый файл.

2.1.1 Расширенный синтаксис командной строки NASM :

- Здесь мы запустили полную команду NASM и проверили выходные файлы, которые дала нам. Разница заключалась в том, что с помощью полной команды нам нужно указать имя объектного файла и список файлов, и это то, что получилось после проверки с помощью запятой **ls**. (рис. [2.5])

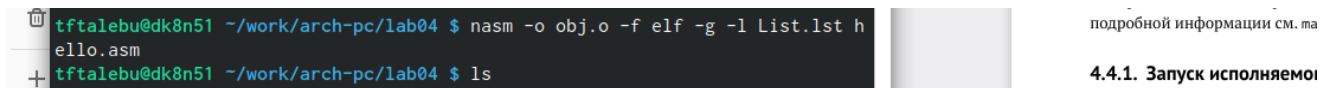


Рис. 2.5: Ресункек 5

2.1.1.1 Компоновщик LD :

- На этом шаге и с помощью компоновщика с командой **ld** мы смогли получить исполняемый файл, обработав объектный файл. Затем, используя команду **ls**, мы проверили, что файл был создан. (рис. [2.6])

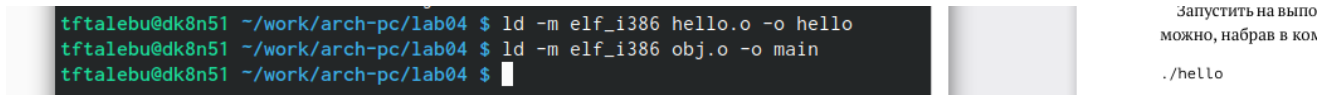


Рис. 2.6: Ресункек 6

- Затем мы проверили, что можем присвоить исполняемому файлу любое имя, а не только то же имя, что и объектному файлу, как показано с помощью команды **ls**. (рис. [2.6])
- Исполняемый файл имеет имя "main", а для объектного файла - "obj".

2.1.1.1.1 Запуск исполняемого файла :

- На этом шаге все, что мы сделали, это запустили исполняемый файл. (рис. [2.7])


```
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ ./hello
Hello world!
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ █
```

1. Создание программы Hello

Рис. 2.7: Ресункек 8

2.1.1.1.1 Выводы по результатам выполнения заданий :

- В этой лабораторной работе мы освоили, как скомпилировать текстовый файл, написанный на языке ассемблера NASM, в объектный файл, затем получить оправдание, и все это ради создания программы, которая печатает знаменитое предложение “**Hello world!**”

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. [3.1] приведено краткое описание стандартных каталогов Unix.

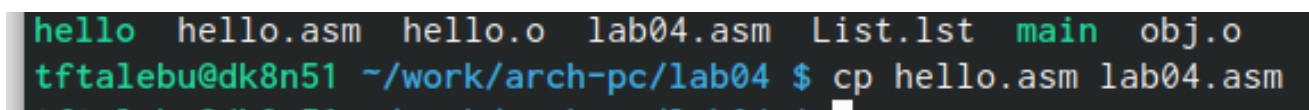
Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

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 Задание для самостоятельной работы :

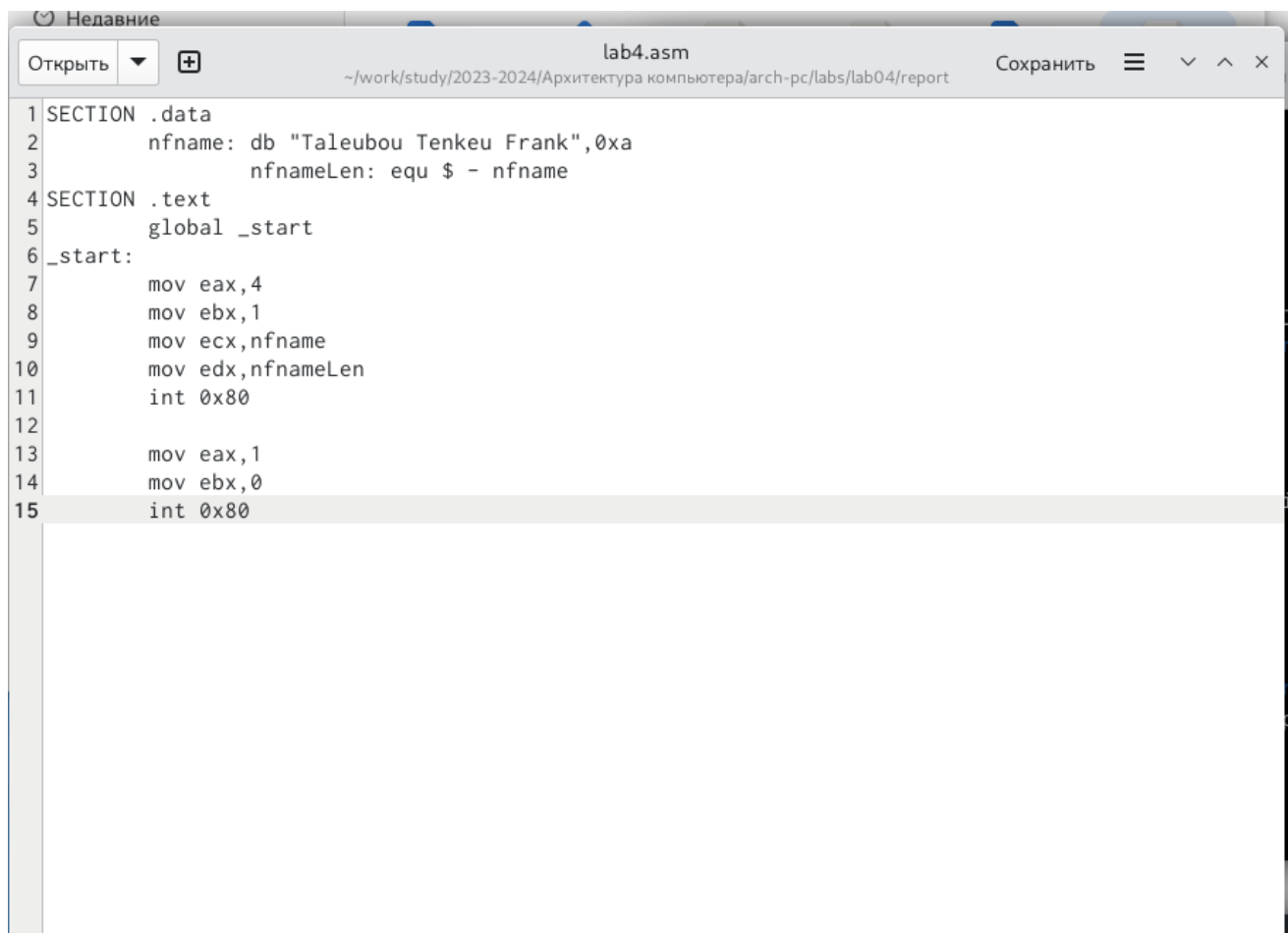
- В каталоге ~/work/arch-pc/lab05 мы создали копию для файла hello.asm и присвоили ему имя lab05. (рис. [4.1])



```
hello  hello.asm  hello.o  lab04.asm  List.lst  main  obj.o
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ cp hello.asm lab04.asm
```

Рис. 4.1: Ресукнек 9

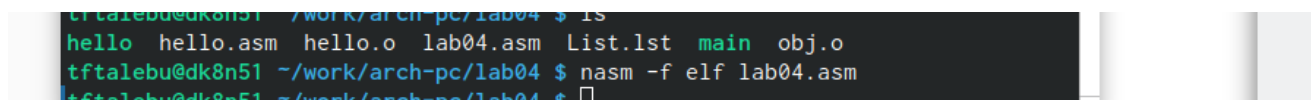
- Используя текстовый редактор gedit, мы изменили текстовый файл, содержащий ассемблерный код, чтобы программа выводила мое имя и фамилию “Max Sylvain”.



```
1 SECTION .data
2     nfname: db "Taleubou Tenkeu Frank",0xa
3             nfnameLen: equ $ - nfname
4 SECTION .text
5     global _start
6 _start:
7     mov eax,4
8     mov ebx,1
9     mov ecx,nfname
10    mov edx,nfnameLen
11    int 0x80
12
13    mov eax,1
14    mov ebx,0
15    int 0x80
```

Рис. 4.2: Ресункек 10

- После написания кода е скомпилировал код в объектный файл после чего получил исполняемый файл с помощью компоновщика. (рис. [4.3])



```
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o lab04.asm List.lst main obj.o
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ nasm -f elf lab04.asm
tftalebu@dk8n51 ~/work/arch-pc/lab04 $
```

Рис. 4.3: Ресункек 11

- Затем мы запустили исполняемый файл.(рис. [4.4])

```
/report $ ./lab4
Talebou Tenkeu Frank
tftalebu@dk4n69 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04
/report $
```

Рис. 4.4: Ресункек 12

- Здесь мы скопировали оба hello.Asm и lab5.asm в ваш локальный репозиторий.(рис. [4.5])

```
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab04.o -o lab04
tftalebu@dk8n51 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab04  lab04.asm  lab04.o  List.lst  main  obj.o
tftalebu@dk8n51 ~/work/arch-pc/lab04 $
```

Рис. 4.5: Ресункек 13

4.1 Выводы по результатам выполнения заданий :

- В этих упражнениях мы применили навыки, полученные в ходе лабораторной работы, в ходе которой получили более глубокое представление об именах регистров и о том, как выделить для них память.

5 Выводы

- В шестой лабораторной работе мы можем получить практические навыки по созданию компиляции и обработке программы с использованием языка ассемблера 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.