

Отчёт по лабораторной работе № 4

**Создание и процесс обработки программ на языке ассемблера
NASM**

Мальянц Виктория Кареновна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Создание программы Hello world!	7
3.2	Работа с транслятором NASM	9
3.3	Работа с расширенным синтаксисом командной строки NASM . . .	9
3.4	Работа с компоновщиком LD	9
3.5	Запуск исполняемого файла	10
3.6	Выполнение заданий для самостоятельной работы	10
4	Выводы	14

Список иллюстраций

3.1	Создание каталога	7
3.2	Переход в созданный каталог	7
3.3	Создание текстового файла hello.asm	7
3.4	Открытие файла hello.asm в текстовом редакторе gedit через терминал	7
3.5	Окно текстового редактора gedit	8
3.6	Заполнение файла	8
3.7	Компиляция текста программы	9
3.8	Компиляция текста программы	9
3.9	Передача объектного файла на обработку компоновщику	10
3.10	Передача объектного файла на обработку компоновщику	10
3.11	Запуск исполняемого файла	10
3.12	Создание копии файла	10
3.13	Открытие файла lab4.asm в текстовом редакторе gedit через терминал	10
3.14	Изменение программы	11
3.15	Компиляция текста программы	11
3.16	Компиляция текста программы	11
3.17	Запуск исполняемого файла	11
3.18	Создание копии файла в каталоге	12
3.19	Создание копии файла в каталоге	12
3.20	Переход в каталог	12
3.21	Файлы hello.asm и lab4.asm в каталоге	12
3.22	Добавление файлов	12
3.23	Сохранение изменений	12
3.24	Отправка файлов	13

Список таблиц

1 Цель работы

Освоение 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

3 Выполнение лабораторной работы

3.1 Создание программы Hello world!

Создаю каталог для работы с программами на языке ассемблера Nasm с помощью команды `mkdir -p` (рис. 3.1).

```
vkmaljyanc@vbox:~$ mkdir -p ~/work/arch-pc/lab04
```

Рис. 3.1: Создание каталога

Перехожу в созданный каталог с помощью команды `cd` (рис. 3.2).

```
vkmaljyanc@vbox:~$ cd ~/work/arch-pc/lab04
vkmaljyanc@vbox:~/work/arch-pc/lab04$
```

Рис. 3.2: Переход в созданный каталог

Создаю текстовый файл с именем `hello.asm` с помощью команды `touch` (рис. 3.3).

```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ touch hello.asm
```

Рис. 3.3: Создание текстового файла `hello.asm`

Открываю этот файл с помощью текстового редактора `gedit` (рис. 3.4) (рис. 3.5).

```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ gedit hello.asm
```

Рис. 3.4: Открытие файла `hello.asm` в текстовом редакторе `gedit` через терминал

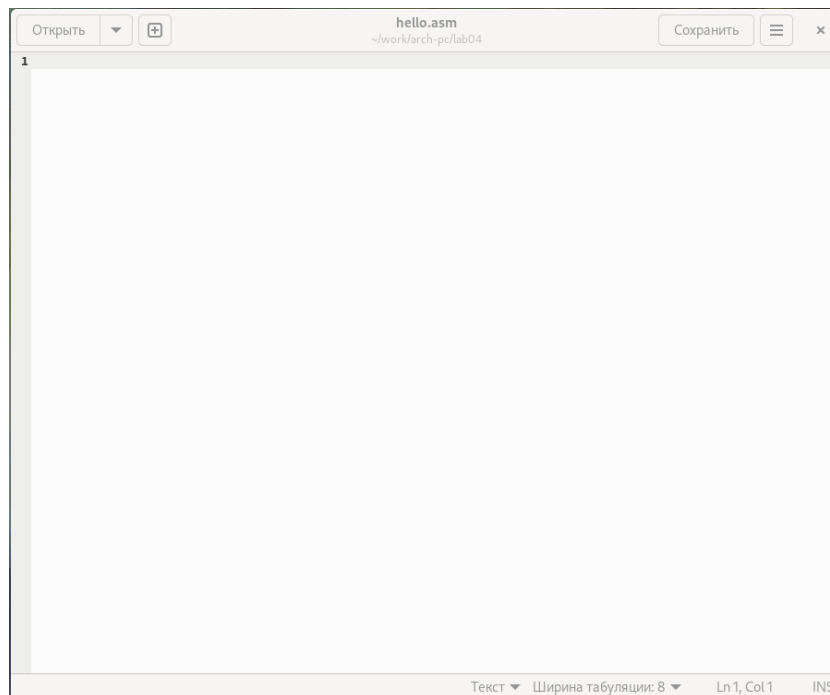


Рис. 3.5: Окно текстового редактора gedit

Заполняю файл, вставляя в него программу для вывода “Hello World!” (рис. 3.6).

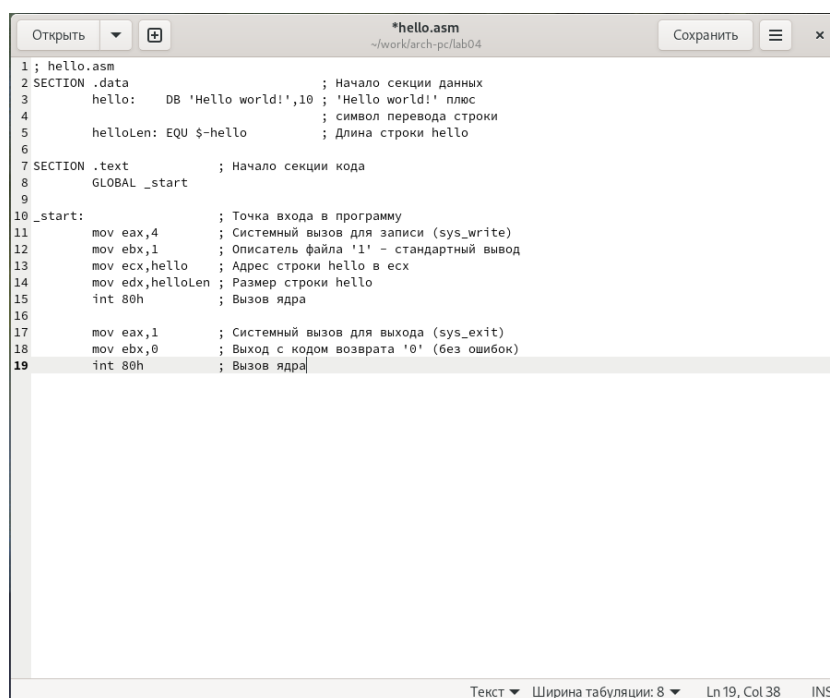


Рис. 3.6: Заполнение файла

3.2 Работа с транслятором NASM

Преобразовываю текст программы “Hello World!” в объектный код с помощью транслятора NASM. Для этого использую команду `nasm -f elf hello.asm`, ключ `-f` указывает транслятору, что требуется создать бинарные файлы в формате ELF. Проверяю, что объектный файл был создан с помощью команды `ls`. Объектный файл имеет имя `hello.o` (рис. 3.7).

```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ nasm -f elf hello.asm
vkmaljyanc@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
```

Рис. 3.7: Компиляция текста программы

3.3 Работа с расширенным синтаксисом командной строки NASM

Ввожу команду, которая 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`) и будет создан файл листинга `list.lst` (опция `-l`). С помощью команды `ls` проверяю, что файлы были созданы (рис. 3.8).

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

Рис. 3.8: Компиляция текста программы

3.4 Работа с компоновщиком LD

Передаю объектный файл на обработку компоновщику LD, чтобы получить программу `hello`. Ввожу команду `ld -m elf_i386 obj.o -o main`. Ключ `-o` задает имя создаваемого исполняемого файла. С помощью команды `ls` проверяю, что файл `hello` был создан (рис. 3.9).

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

Рис. 3.9: Передача объектного файла на обработку компоновщику

Выполняю следующую программу (рис. 3.10). Исполняемый файл будет иметь имя `main`. Объектный файл, из которого собран исполняемый файл, имеет имя `obj.o`.

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

Рис. 3.10: Передача объектного файла на обработку компоновщику

3.5 Запуск исполняемого файла

Запускаю на выполнение созданный исполняемый файл `hello` (рис. 3.11).

```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ ./hello
Hello world!
```

Рис. 3.11: Запуск исполняемого файла

3.6 Выполнение заданий для самостоятельной работы

С помощью команды `cp` создаю копию файла `hello.asm` с именем `lab4.asm` в каталоге `~/work/arch-pc/lab04` (рис. 3.12).

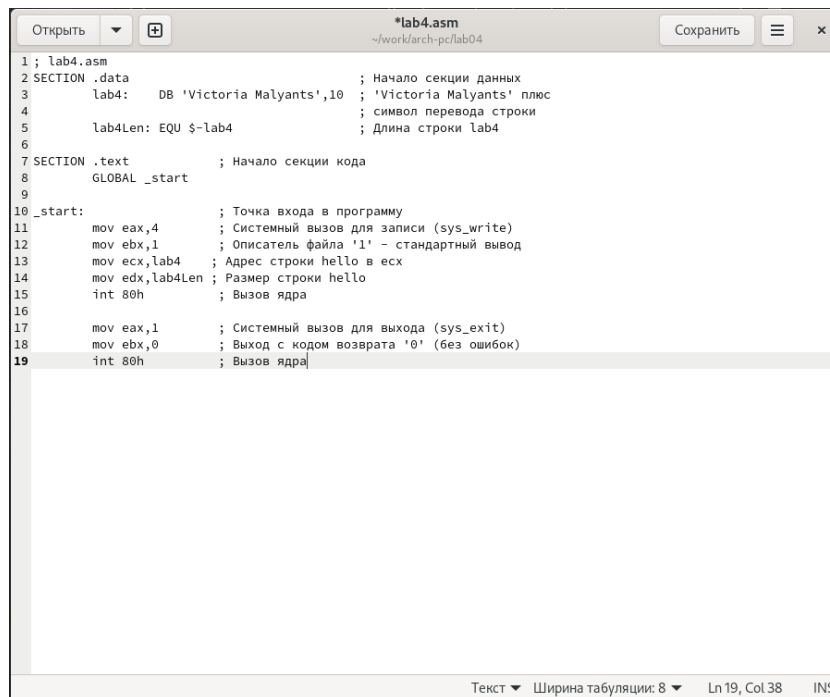
```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
```

Рис. 3.12: Создание копии файла

С помощью текстового редактора `gedit` открываю файл `lab4.asm`. В нем вношу изменения в программу так, чтобы вместо `Hello world!` на экран выводилась строка с моими именем и фамилией (рис. 3.13) (рис. 3.14).

```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ gedit lab4.asm
```

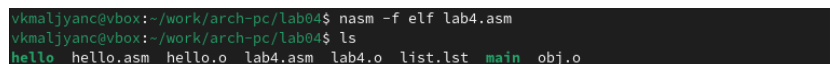
Рис. 3.13: Открытие файла `lab4.asm` в текстовом редакторе `gedit` через терминал



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

Рис. 3.14: Изменение программы

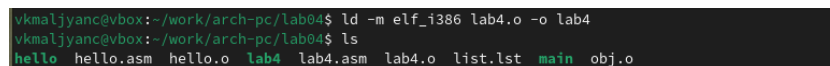
Компилирую текст программы в объектный файл. Проверяю с помощью команды `ls`, что файл `lab4.o` создан (рис. 3.15).



```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
vkmaljyanc@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  lab4.o  list.lst  main  obj.o
```

Рис. 3.15: Компиляция текста программы

Передаю объектный файл `lab4.o` на обработку компоновщику `LD`, чтобы получить исполняемый файл `lab4`. С помощью команды `ls` проверяю, что файл `lab4` был создан (рис. 3.16).



```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
vkmaljyanc@vbox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.o  list.lst  main  obj.o
```

Рис. 3.16: Компиляция текста программы

Запускаю исполняемый файл `lab4` (рис. 3.17).



```
vkmaljyanc@vbox:~/work/arch-pc/lab04$ ./lab4
Victoria Malyants
```

Рис. 3.17: Запуск исполняемого файла

Копирую файл `hello.asm` в каталог `~/work/study/2024-2025/“Архитектура компьютера”/arch-pc/labs/lab04/` (рис. 3.18).

```
vkmaljyanc@vbox: ~/work/arch-pc/lab04$ cp hello.asm ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/labs/lab04/
```

Рис. 3.18: Создание копии файла в каталоге

Копирую файл `lab4.asm` в каталог `~/work/study/2024-2025/“Архитектура компьютера”/arch-pc/labs/lab04/` (рис. 3.19).

```
vkmaljyanc@vbox: ~/work/arch-pc/lab04$ cp lab4.asm ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/labs/lab04/
```

Рис. 3.19: Создание копии файла в каталоге

Перехожу в каталог `~/work/study/2024-2025/“Архитектура компьютера”/arch-pc/labs/lab04/` (рис. 3.20).

```
vkmaljyanc@vbox: ~/work/arch-pc/lab04$ cd ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/labs/lab04/
vkmaljyanc@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$
```

Рис. 3.20: Переход в каталог

С помощью команды `ls` убеждаюсь в том, что файлы `hello.asm` и `lab4.asm` скопированы в каталог (рис. 3.21).

```
vkmaljyanc@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ ls
hello.asm  lab4.asm  presentation  report
```

Рис. 3.21: Файлы `hello.asm` и `lab4.asm` в каталоге

Добавляю изменения на Github с помощью команды `git add .` (рис. 3.22).

```
vkmaljyanc@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git add .
```

Рис. 3.22: Добавление файлов

Сохраняю изменения на Github с помощью команды `git commit -m` (рис. 3.23).

```
vkmaljyanc@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git commit -m
"Add files for lab04"
[master 0965788] Add files for lab04
4 files changed, 38 insertions(+)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/report/report.docx
create mode 100644 labs/lab04/report/report.pdf
```

Рис. 3.23: Сохранение изменений

Отправляю все произведенные изменения локального дерева в центральный репозиторий (рис. 3.24).

```
vkmaljanc@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04$ git push
Перечисление объектов: 13, готово.
Подсчет объектов: 100% (13/13), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (9/9), готово.
Запись объектов: 100% (9/9), 539.16 КиБ | 831.00 КиБ/с, готово.
Total 9 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To github.com:victoriamalyants/study_2024-2025_arh-pc.git
   0958d2e..0965788  master -> master
```

Рис. 3.24: Отправка файлов

4 Выводы

Я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.