

# **Лабораторная работа №4**

Жукова Арина Александровна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
2.1	Программа Hello world! . . . . .	6
2.2	Транслятор NASM . . . . .	7
2.3	Расширенный синтаксис командной строки NASM . . . . .	7
2.4	Компоновщик LD . . . . .	9
2.5	Запуск исполняемого файла . . . . .	10
<b>3</b>	<b>Задание для самостоятельной работы</b>	<b>11</b>
<b>4</b>	<b>Выводы</b>	<b>15</b>

# Список иллюстраций

2.1	Создание каталога . . . . .	6
2.2	Создание текстового файла . . . . .	6
2.3	Внесение изменений в файл . . . . .	7
2.4	Компиляция файла . . . . .	7
2.5	Компиляция файла . . . . .	8
2.6	Список команд . . . . .	8
2.7	Подробная информация . . . . .	8
2.8	Компановка файла hello.o . . . . .	9
2.9	Компановка файла obj.o . . . . .	9
2.10	Формат командной строки . . . . .	9
2.11	Подробная информация . . . . .	10
2.12	Запуск исполняемого файла . . . . .	10
3.1	Создание копии . . . . .	11
3.2	Создание копии . . . . .	12
3.3	Запуск исполняемого файла . . . . .	12
3.4	Копирование файлов в локальный репозиторий . . . . .	13
3.5	Загрузка файлов на Github 1 . . . . .	13
3.6	Загрузка файлов на Github 2 . . . . .	13
3.7	Github . . . . .	14

## **Список таблиц**

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

### 2.1 Программа Hello world!

Создадим каталог для работы с программами на языке ассемблера NASM, перейдём в созданный каталог (рис.[2.1])

```
aazhukova1@dk5n52 ~ $ mkdir -p ~/work/arch-pc/lab04
aazhukova1@dk5n52 ~ $ cd ~/work/arch-pc/lab04
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ touch hello.asm
```

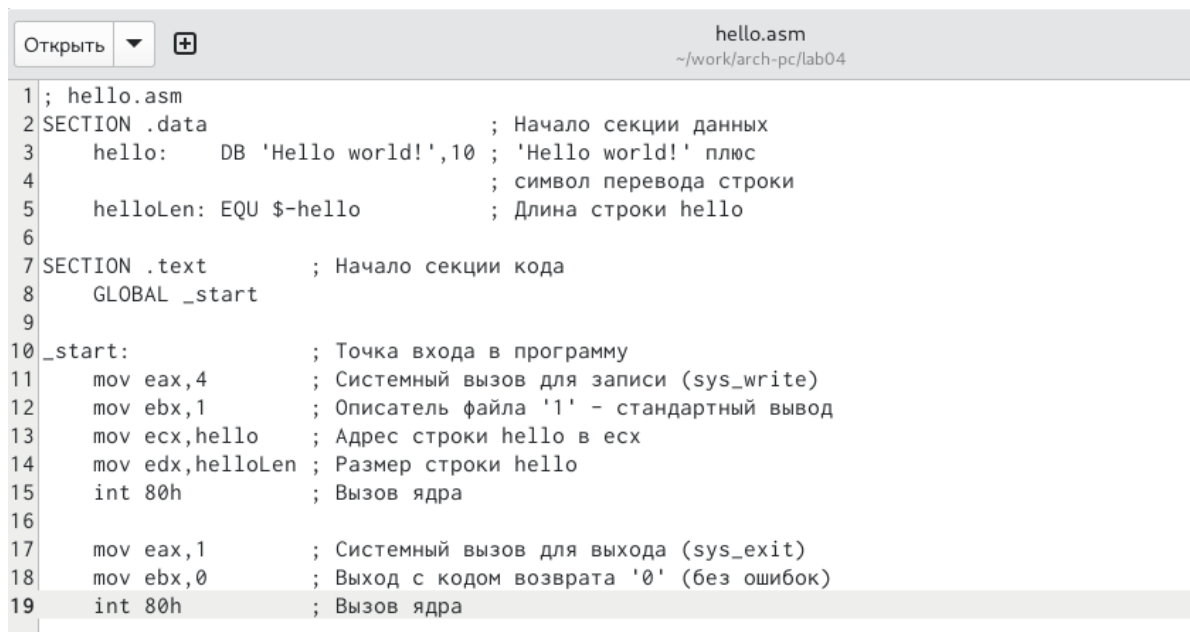
Рис. 2.1: Создание каталога

Создадим текстовый файл с именем hello.asm, откроем этот файл при помощи текстового редактора gedit (рис. [2.2]).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ touch hello.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 2.2: Создание текстового файла

Изменим содержимое текстового файла (рис.[2.3])

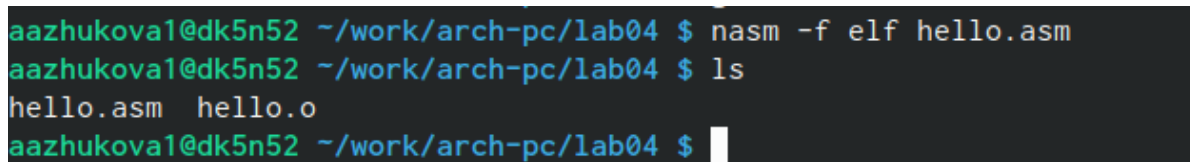


```
1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3     hello:    DB 'Hello world!',10 ; 'Hello world!' плюс
4                                     ; символ перевода строки
5     helloLen: EQU $-hello      ; Длина строки hello
6
7 SECTION .text                ; Начало секции кода
8     GLOBAL _start
9
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
17     mov eax,1              ; Системный вызов для выхода (sys_exit)
18     mov ebx,0              ; Выход с кодом возврата '0' (без ошибок)
19     int 80h                ; Вызов ядра
```

Рис. 2.3: Внесение изменений в файл

## 2.2 Транслятор NASM

Проведём компиляцию файла hello.asm и проверим, что объектный код был записан правильно и файл hello.o был создан (рис. [2.4]).



```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $
```

Рис. 2.4: Компиляция файла

## 2.3 Расширенный синтаксис командной строки NASM

Скомпилируем 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, будет создан файл листинга list.lst (опция -l). Проверим создание файлов (рис.[2.5]).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $
```

Рис. 2.5: Компиляция файла

Чтобы узнать более подробную информацию введём команду `man nasm`. Для получения списка форматов объектного файла `nasm -hf` (рис.[2.6]-[2.7])

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ man nasm
aazhukova1@dk5n52 ~/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
```

Рис. 2.6: Список команд

```
NASM(1)                                The Netwide Assembler Project                                NASM(1)

NAME
  nasm - the Netwide Assembler, a portable 80x86 assembler

SYNOPSIS
  nasm [-@ response file] [-f format] [-o outfile] [-l listfile] [options...] filename

DESCRIPTION
  The nasm command assembles the file filename and directs output to the file outfile if specified. If outfile is not specified, nasm will derive a default output file name from the name of its input file, usually by appending '.o' or '.obj', or by removing all extensions for a raw binary file. Failing that, the output file name will be 'nasm.out'.

OPTIONS
  -@ filename
    Causes nasm to process options from filename as if they were included on the command line.

  -a
    Causes nasm to assemble the given input file without first applying the macro preprocessor.
```

Рис. 2.7: Подробная информация



## 2.4 Компоновщик LD

Передадим объектный файл hello.o на обработку компоновщику, а также проверим, что исполняемый файл hello был создан (рис.[2.8]).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис. 2.8: Компоновка файла hello.o

Передадим объектный файл obj.o на обработку компоновщику, а также проверим, что исполняемый файл main был создан (рис.[2.9]).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
```

Рис. 2.9: Компоновка файла obj.o

Узнаем формат командной строки (рис.[2.10]).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ld --help
Использование ld [параметры] файл...
Параметры:
  -а КЛЮЧЕВОЕ СЛОВО                Управление общей библиотекой для совместимости с HP/UX
  -A АРХИТЕКТУРА, --architecture АРХИТЕКТУРА  Задать архитектуру
  -b ЦЕЛЬ, --format ЦЕЛЬ            Задать цель для следующих входных файлов
  -с ФАЙЛ, --mri-script ФАЙЛ        Прочитать сценарий компоновщика в формате MRI
  -d, -dc, -dp                      Принудительно делать общие символы определёнными
```

Рис. 2.10: Формат командной строки

Просматриваем более подробную информацию man ld (рис.[2.11])

```
LD(1) GNU Development Tools LD(1)
NAME
    ld - The GNU linker
SYNOPSIS
    ld [options] obfile ...
DESCRIPTION
    ld combines a number of object and archive files, relocates their data and ties up symbol references. Usually the last step in compiling a program is to run ld.
    ld accepts Linker Command Language files written in a superset of AT&T's Link Editor Command Language syntax, to provide explicit and total control over the linking process.
```

Рис. 2.11: Подробная информация

## 2.5 Запуск исполняемого файла

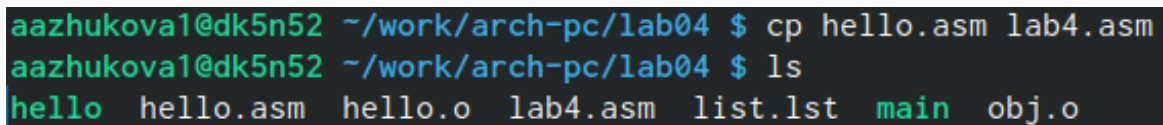
Запускаем созданный исполняемый файл, находящийся в текущем каталоге (рис.[2.12])

```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

Рис. 2.12: Запуск исполняемого файла

### 3 Задание для самостоятельной работы

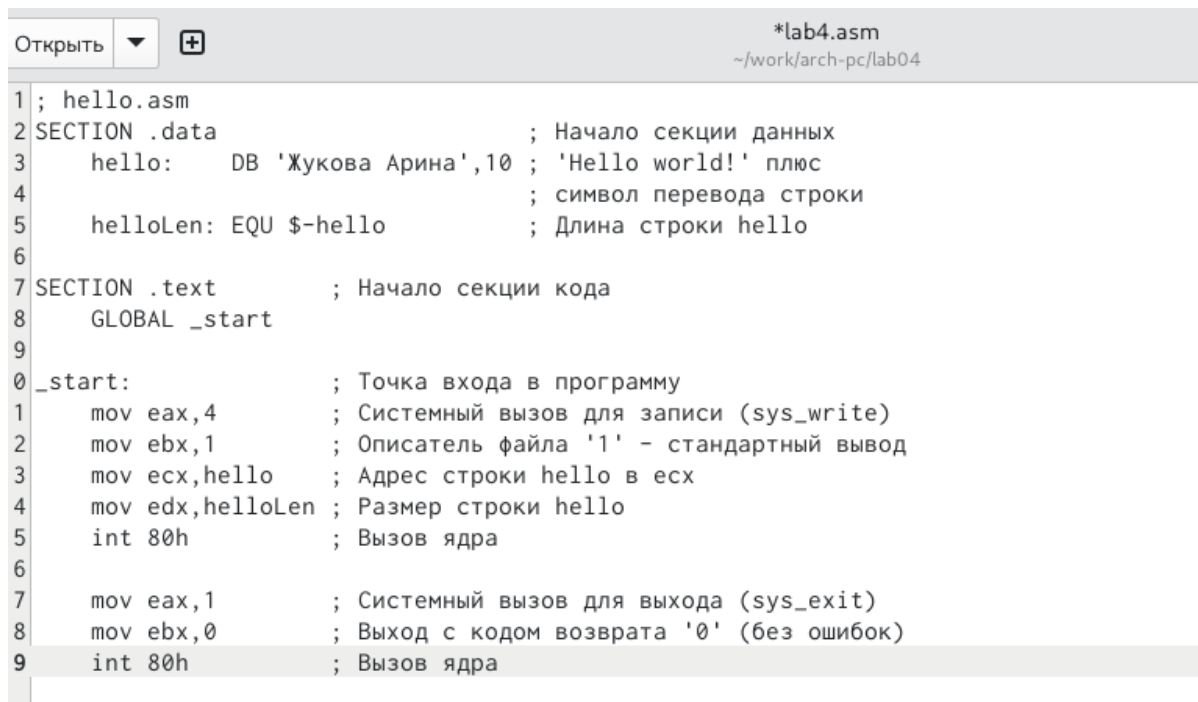
1. Создаём копию файла `hello.asm` с именем `lab4.asm` в каталоге `~/work/arch-pc/lab04` (рис.[3.1])



```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
```

Рис. 3.1: Создание копии

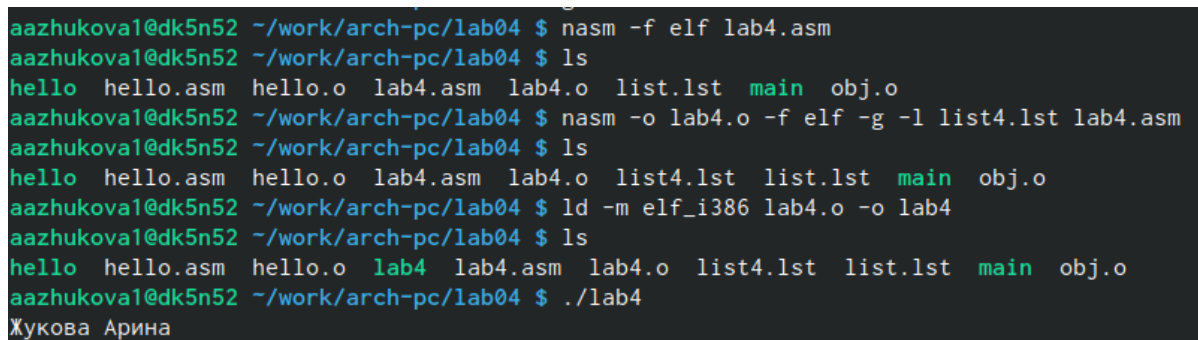
2. Вносим изменения в файл `lab4.asm` при помощи текстового редактора `gedit`, чтобы вместо `Hello world!` на экран выводилась строка с моими фамилией и именем (рис.[3.2]).



```
Открыть  + *lab4.asm
~/work/arch-pc/lab04
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello:    DB 'Жукова Арина',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
17     mov eax,1 ; Системный вызов для выхода (sys_exit)
18     mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
19     int 80h ; Вызов ядра
```

Рис. 3.2: Создание копии

3. Оттранслируем полученный текст программы lab4.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл (рис.[3.3]).



```
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o lab4.asm lab4.o list.lst main obj.o
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ nasm -o lab4.o -f elf -g -l list4.lst lab4.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o lab4.asm lab4.o list4.lst list.lst main obj.o
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab4.o -o lab4
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list4.lst list.lst main obj.o
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ./lab4
Жукова Арина
```

Рис. 3.3: Запуск исполняемого файла

4. Скопируйте файлы hello.asm и lab4.asm в Ваш локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/ (рис.[3.4]).

```

aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ cp ~/work/arch-pc/lab04/hello.asm ~/work/study/2023-2024/Архитектура\ компьютера/study_2023-2024_arh-pc/labs/lab04
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.o  list4.lst  list.lst  main  obj.o
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ cp ~/work/arch-pc/lab04/lab4.asm ~/work/study/2023-2024/Архитектура\ компьютера/study_2023-2024_arh-pc/labs/lab04
aazhukova1@dk5n52 ~/work/arch-pc/lab04 $ cd
aazhukova1@dk5n52 ~ $ cd ~/work/study/2023-2024/Архитектура\ компьютера/study_2023-2024_arh-pc/labs/lab04
aazhukova1@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab04 $ ls
hello.asm  lab4.asm  presentation  report
aazhukova1@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab04 $

```

Рис. 3.4: Копирование файлов в локальный репозиторий

Выгрузим файлы на GitHub (рис.[3.5]-[3.6]).

```

aazhukova1@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab04 $ git add .
aazhukova1@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab04 $ git commit -am 'feat(main): add files lab-4'
[master 6fd2e34] feat(main): add files lab-4
21 files changed, 43 insertions(+), 6 deletions(-)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/report/image/Снимок экрана от 2023-10-12 09-21-40.png

```

Рис. 3.5: Загрузка файлов на Github 1

```

aazhukova1@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab04 $ git push
Перечисление объектов: 39, готово.
Подсчет объектов: 100% (39/39), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (39/39), готово.

```

Рис. 3.6: Загрузка файлов на Github 2

Проверим наличие файлов на Github (рис.[3.7]).

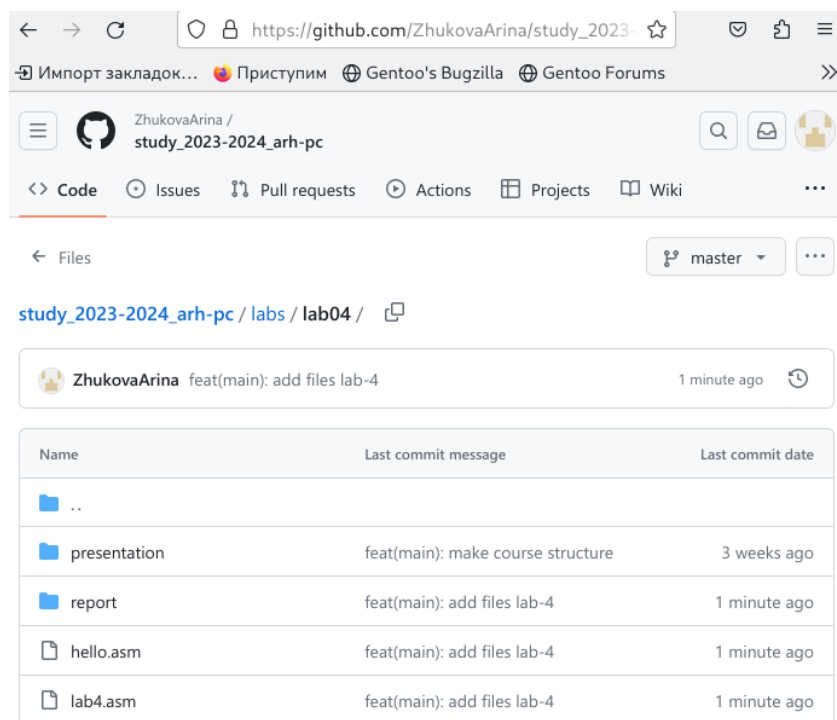


Рис. 3.7: Github

## **4 Выводы**

При выполнении данной лабораторной работы мною были освоены процедуры компиляции и сборки программ, написанных на ассемблере NASM.