

# **отчёт по лабораторной работе № 5**

**Нкнбд-05-2023**

Диого Элизеу Луиж Музумбо

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
3.1	Программа Hello world! . . . . .	6
3.2	Транслятор NASM . . . . .	7
3.3	Расширенный синтаксис командной строки NASM . . . . .	8
3.4	Компоновщик LD . . . . .	9
3.5	Запуск исполняемого файла . . . . .	10
3.6	Задание для самостоятельной работы . . . . .	10
<b>4</b>	<b>Выводы</b>	<b>13</b>

## Список иллюстраций

3.1	Созданный каталог . . . . .	6
3.2	Переход в каталог . . . . .	6
3.3	gedit . . . . .	7
3.4	Компиляция . . . . .	7
3.5	Созданный объектный файл . . . . .	8
3.6	obj.o . . . . .	8
3.7	Созданные файлы . . . . .	9
3.8	Работа компоновщика . . . . .	9
3.9	Созданный файл hello . . . . .	9
3.10	Компоновка файла . . . . .	9
3.11	Проверка названий файлов . . . . .	9
3.12	ld -help . . . . .	10
3.13	Выполнение файла . . . . .	10
3.14	cp lab5.asm . . . . .	10
3.15	ls lab05 . . . . .	10
3.16	Изменения в тексте программы . . . . .	11
3.17	lab5.o . . . . .	11
3.18	lab5.o . . . . .	11
3.19	lab5 запуск . . . . .	11
3.20	hello.asm . . . . .	12
3.21	lab5.asm . . . . .	12
3.22	Загрузка файлов на Github . . . . .	12

# 1 Цель работы

Освоение процедуры компиляции и сборки программ, написанных на, машинноориентированный языке низкого уровня, ассемблере NASM.

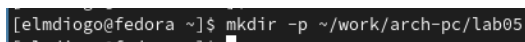
## 2 Задание

1. Создать файлы расширения .asm.
2. Отредактировать .asm файлы.
3. Оттранслировать .asm файлы в объектные.
4. С помощью компоновщика создать исполняемые файлы и запустить.

## 3 Выполнение лабораторной работы

### 3.1 Программа Hello world!

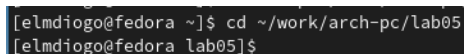
Рассмотрели пример простой программы на языке ассемблера NASM. Традиционно первая программа выводит приветственное сообщение Hello world! на экран. Создали каталог для работы с программами на языке ассемблера NASM: (рис. 3.1)



```
[elmdiego@fedora ~]$ mkdir -p ~/work/arch-pc/lab05
```

Рис. 3.1: Созданный каталог

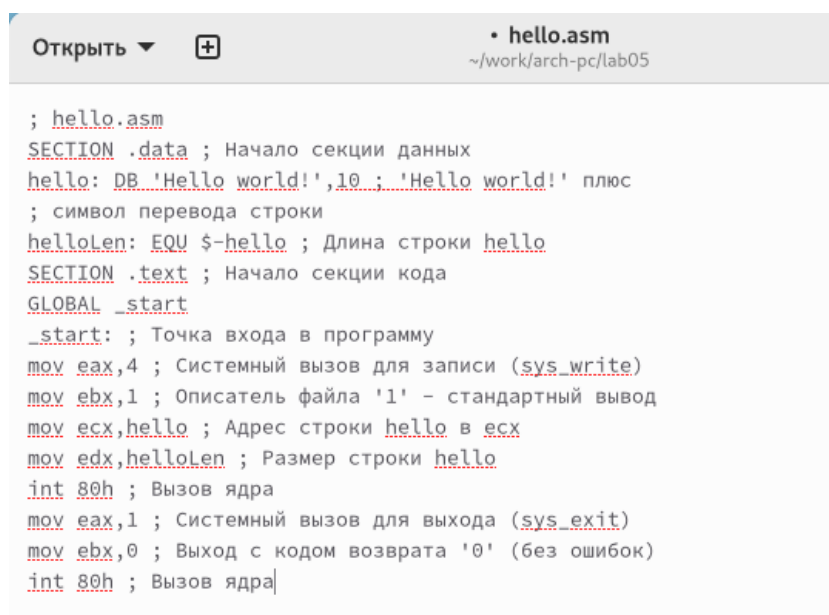
Перешли в созданный каталог. (рис. 3.2)



```
[elmdiego@fedora ~]$ cd ~/work/arch-pc/lab05  
[elmdiego@fedora lab05]$
```

Рис. 3.2: Переход в каталог

Создали текстовый файл с именем hello.asm. Открыли этот файл с помощью текстового редактора gedit. Введите в него следующий текст: (рис. 3.3)



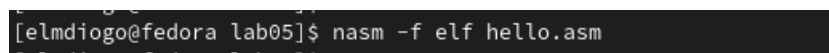
```
• hello.asm
~/work/arch-pc/lab05

; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
```

Рис. 3.3: gedit

## 3.2 Транслятор NASM

NASM превращает текст программы в объектный код. Для компиляции текста программы «Hello World» написали: (рис. 3.4)



```
[elmdiego@fedora lab05]$ nasm -f elf hello.asm
```

Рис. 3.4: Компиляция

С помощью транслятора преобразовали текст программы из файла hello.asm в объектный код, который записали в файл hello.o С помощью команды ls проверили, что объектный файл был создан. Объектный файл имеет имя hello.o (рис. 3.5)



hello.o

Рис. 3.5: Созданный объектный файл

### 3.3 Расширенный синтаксис командной строки NASM

Выполнили следующую команду: (рис. 3.6)

```
[elmdiego@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst hello.asm
```

Рис. 3.6: obj.o

Данная команда скомпилирует исходный файл `hello.asm` в `obj.o` (опция `-o` позволяет задать имя объектного файла, в данном случае `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`).

С помощью команды `ls` проверьте, что файлы были созданы. (рис. 3.7)



```
[elmdiago@fedora lab05]$ ls
hello.asm hello.o list.lst obj.o
```

Рис. 3.7: Созданные файлы

## 3.4 Компоновщик LD

Для получения исполняемой программы, объектный файл передали на обработку компоновщику: (рис. 3.8)

```
[elmdiago@fedora lab05]$ ld -m elf_i386 hello.o -o hello
```

Рис. 3.8: Работа компоновщика

С помощью команды `ls` проверьте, что исполняемый файл `hello` был создан. (рис. 3.9)

```
[elmdiago@fedora lab05]$ ls
hello hello.asm hello.o list.lst obj.o
```

Рис. 3.9: Созданный файл `hello`

Выполните следующую команду: (рис. 3.10)

```
[elmdiago@fedora lab05]$ ld -m elf_i386 obj.o -o main
```

Рис. 3.10: Компоновка файла

Исполняемый файл будет иметь имя `main`. Объектный файл, из которого собран этот исполняемый файл, будет иметь имя `main.o` (рис. 3.11)

```
[elmdiago@fedora lab05]$ ls
hello hello.asm hello.o list.lst main obj.o
```

Рис. 3.11: Проверка названий файлов

Формат командной строки LD увидели, набрав `ld -help`. (рис. 3.12)

```
[elmdiago@fedora lab05]$ ld --help
Использование ld [параметры] файл...
Параметры:
  -a КЛЮЧЕВОЕ СЛОВО                Управление общей библиотекой для совместимости
P/UX
  -A АРХИТЕКТУРА, --architecture АРХИТЕКТУРА  Задать архитектуру
  -b ЦЕЛЬ, --format ЦЕЛЬ              Задать цель для следующих входных файлов
  -c ФАЙЛ, --mri-script ФАЙЛ         Прочитать сценарий компоновщика в формате MRI
  -d, -dc, -dp                      Принудительно делать общие символы определённым
  --dependency-file ФАЙЛ             Write dependency file
  --force-group-allocation           Принудительно удалить членов группы из групп
  -e АДРЕС, --entry АДРЕС            Задать начальный адрес
  -E, --export-dynamic               Экспортировать все динамические символы
  --no-export-dynamic                Отменить действие --export-dynamic
  --enable-non-contiguous-regions    Enable support of non-contiguous memory regions
```

Рис. 3.12: ld -help

## 3.5 Запуск исполняемого файла

Запустили на выполнение созданный исполняемый файл, находящийся в текущем каталоге. (рис. 3.13)

```
[elmdiago@fedora lab05]$ ./hello
Hello world!
```

Рис. 3.13: Выполнение файла

## 3.6 Задание для самостоятельной работы

1. В каталоге ~/work/arch-рс/lab05 с помощью команды cp создали копию файла hello.asm с именем lab5.asm (рис. 3.14), (рис. 3.15)

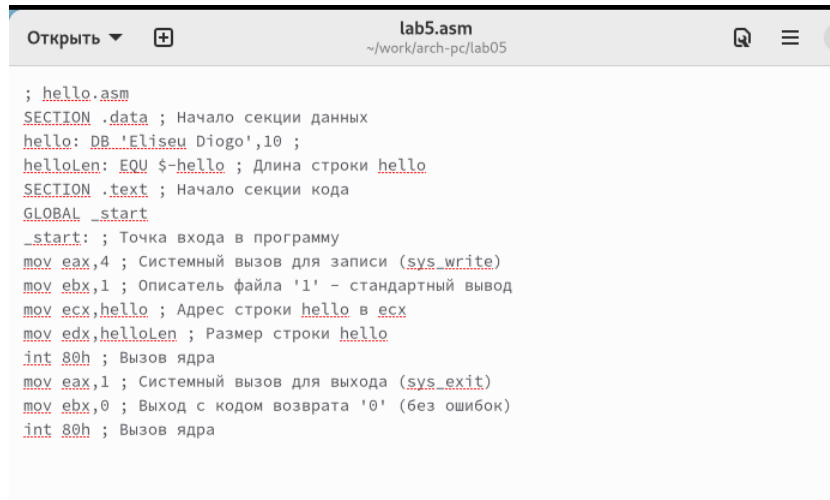
```
[elmdiago@fedora lab05]$ cp hello.asm lab5.asm
```

Рис. 3.14: cp lab5.asm

```
[elmdiago@fedora lab05]$ ls
hello  hello.asm  lab5.asm  list.lst  main  obj.o
```

Рис. 3.15: ls lab05

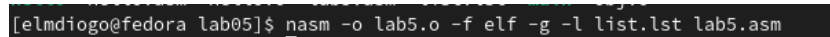
2. С помощью текстового редактора внесли изменения в текст программы в файле lab5.asm так, чтобы вместо Hello world! на экран выводилась строка с фамилией и именем. (рис. 3.16)



```
Открыть + lab5.asm ~/work/arch-pc/lab05
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Eliseu Diogo',10 ;
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
```

Рис. 3.16: Изменения в тексте программы

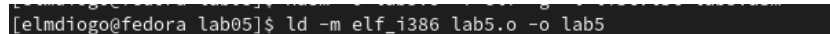
3. Оттранслировали полученный текст программы lab5.asm в объектный файл. (рис. 3.17)



```
[elmdiago@fedora lab05]$ nasm -o lab5.o -f elf -g -l list.lst lab5.asm
```

Рис. 3.17: lab5.o

Выполнили компоновку объектного файла и запустили получившийся исполняемый файл. (рис. 3.18), (рис. 3.19)



```
[elmdiago@fedora lab05]$ ld -m elf_i386 lab5.o -o lab5
```

Рис. 3.18: lab5.o



```
[elmdiago@fedora lab05]$ ./lab5
Eliseu Diogo
[elmdiago@fedora lab05]$
```

Рис. 3.19: lab5 запуск

4. Скопируйте файлы `hello.asm` и `lab5.asm` в Ваш локальный репозиторий в каталог `~/work/study/2022-2023/“Архитектура компьютера”/archpc/labs/lab05/`. (рис. 3.20), (рис. 3.21)

```
[elmdiago@fedora lab05]$ cp hello.asm ~/work/study/2022-2023/"Архитектура компьютера"/study_2022-2023_arh-pc/labs/lab05/
```

Рис. 3.20: `hello.asm`

```
[elmdiago@fedora lab05]$ cp lab5.asm ~/work/study/2022-2023/"Архитектура компьютера"/study_2022-2023_arh-pc/labs/lab05/
```

Рис. 3.21: `lab5.asm`

Загрузите файлы на Github. (рис. 3.22)

```
[elmdiago@fedora lab05]$ cd ~/work/study/2022-2023/"Архитектура компьютера"/study_2022-2023_arh-pc/labs/lab05/
[elmdiago@fedora lab05]$ git add .
[elmdiago@fedora lab05]$ git commit -am "lab5"
[master ff12c57] lab5
124 files changed, 31 insertions(+)
delete mode 100755 labs/lab04/скрин/1/1.png
delete mode 100755 labs/lab04/скрин/1/10.png
delete mode 100755 labs/lab04/скрин/1/11.png
delete mode 100755 labs/lab04/скрин/1/12.png
delete mode 100755 labs/lab04/скрин/1/13.png
delete mode 100755 labs/lab04/скрин/1/14.png
delete mode 100755 labs/lab04/скрин/1/15.png
delete mode 100755 labs/lab04/скрин/1/16.png
delete mode 100755 labs/lab04/скрин/1/17.png
```

Рис. 3.22: Загрузка файлов на Github

## 4 Выводы

В ходе лабораторной работы были освоены процедуры компиляции и сборки программ, написанных на машинноориентированном языке низкого уровня, ассемблере NASM.