

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Компьютерные и информационные науки

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

дисциплина: Архитектура компьютера

Студент: Нечаева Кира

Группа: НКАбд-04-23

МОСКВА

2023__ г.

Содержание

1. Цель работы.....	3
2. Задание.....	4
3. Выполнение лабораторной работы.....	5
4. Выводы.....	11
5. Источники.....	12

1 Цель работы

Целью работы является освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

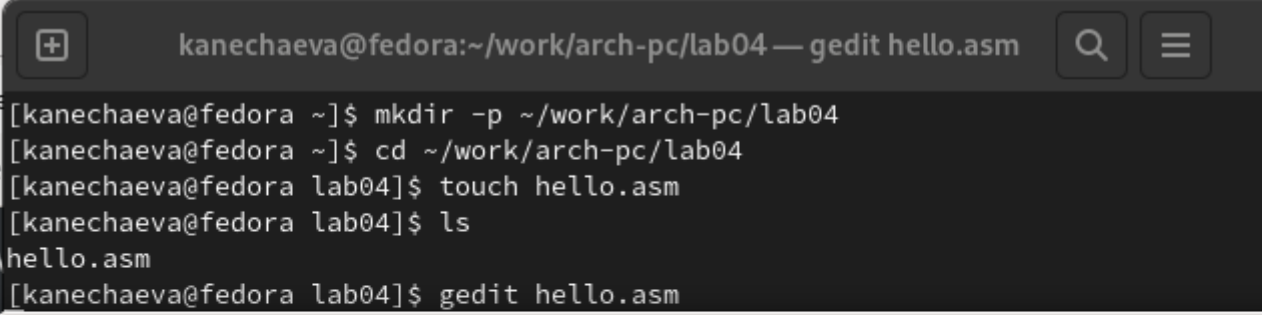
2 Задание

1. Программа Hello world!
2. Транслятор NASM
3. Расширенный синтаксис командной строки NASM
4. Компоновщик LD
5. Запуск исполняемого файла
6. Задание для самостоятельной работы

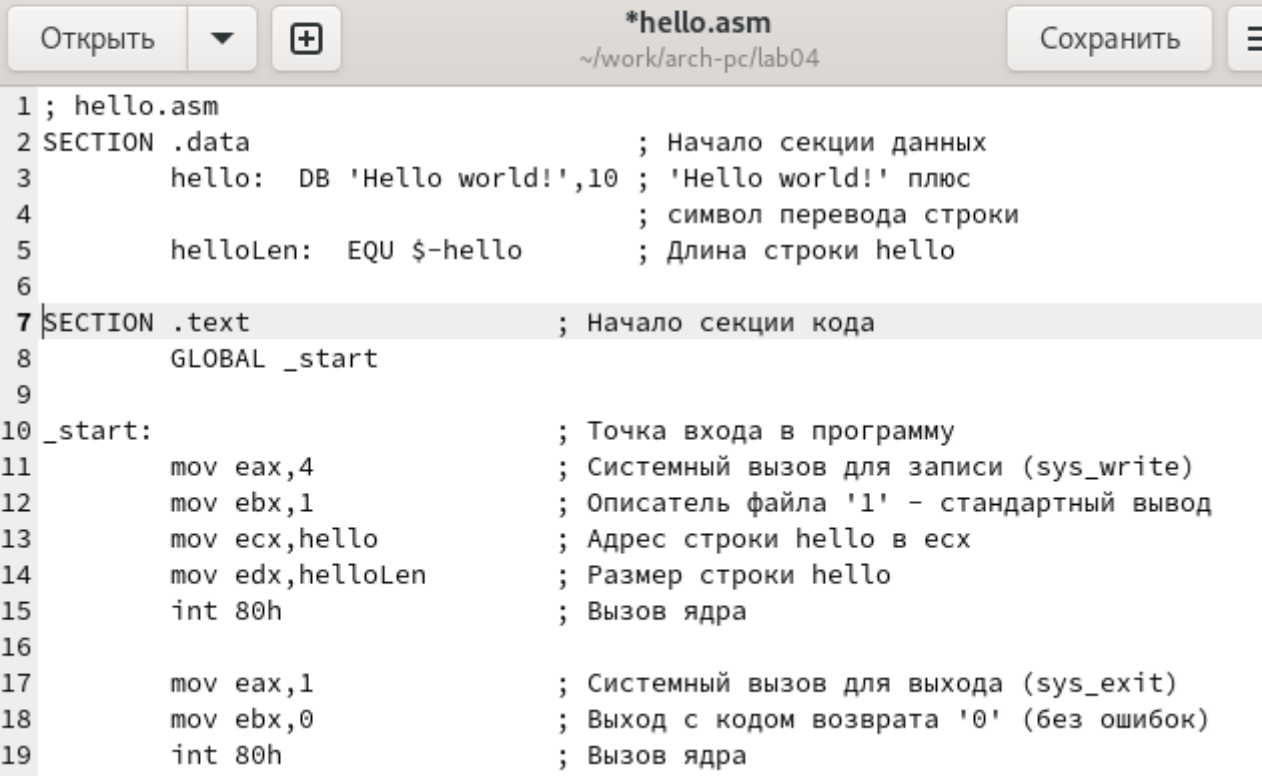
3 Выполнение лабораторной работы

1. Программа Hello world!

Для начала создаю каталог для работы с программами на языке ассемблера NASM, используя команду `mkdir` и ключ `-p`. Перехожу в созданный каталог. Затем создаю текстовый файл с именем `hello.asm` и проверяю правильность выполненных действий с использованием утилиты `ls`. Открываю файл с помощью текстового редактора `gedit` и ввожу текст, который можно увидеть на рис. 1.



```
[kanechaeva@fedora ~]$ mkdir -p ~/work/arch-pc/lab04
[kanechaeva@fedora ~]$ cd ~/work/arch-pc/lab04
[kanechaeva@fedora lab04]$ touch hello.asm
[kanechaeva@fedora lab04]$ ls
hello.asm
[kanechaeva@fedora lab04]$ gedit hello.asm
```

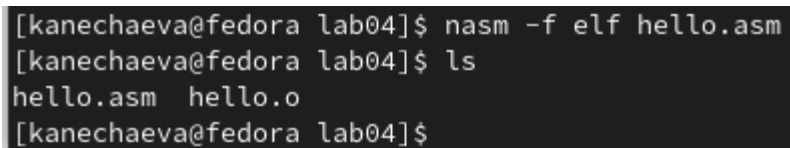
```
1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3     hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4                                     ; символ перевода строки
5     helloLen: EQU $-hello      ; Длина строки hello
6
7 SECTION .text                ; Начало секции кода
8     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. 1. Работа с файлом `hello.asm`

2. Транслятор NASM

Для компиляции приведённого выше текста программы «Hello World» необходимо написать: `nasm -f elf hello.asm`. Ключ `-f` указывает транслятору, что требуется создать бинарные файлы в формате ELF.

С помощью команды `ls` проверяю, что объектный файл был создан. Он имеет название `hello.o`. Это означает, что текст программы был набран без ошибок и транслятор преобразовал текст программы из файла `hello.asm` в объектный код, который записался в файл `hello.o`. (рис. 2)

A terminal window with a dark background and light-colored text. The prompt is [kanechaeva@fedora lab04]\$. The first command is nasm -f elf hello.asm. The second command is ls. The output of ls is hello.asm hello.o. The prompt returns to [kanechaeva@fedora lab04]\$.

```
[kanechaeva@fedora lab04]$ nasm -f elf hello.asm
[kanechaeva@fedora lab04]$ ls
hello.asm  hello.o
[kanechaeva@fedora lab04]$
```

Рис. 2. Создание файла `hello.o`

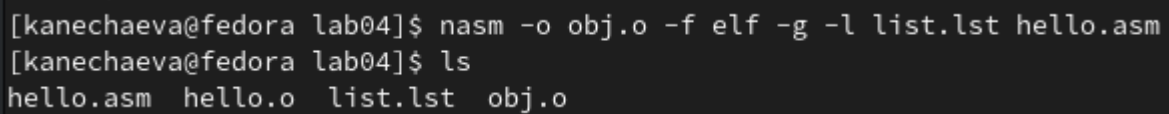
3. Расширенный синтаксис командной строки NASM

Мне нужно выполнить команду:

```
nasm -o obj.o -f elf -g -l list.lst hello.asm
```

Эта команда компилирует исходный файл `hello.asm` в `obj.o` (ключ `-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 при помощи опции `-g`. Кроме того, будет создан файл `list.lst` (опция `-l`).

Проверяю, что все файлы были созданы.



```
[kanechaeva@fedora lab04]$ nasm -o obj.o -f elf -g -l list.lst hello.asm
[kanechaeva@fedora lab04]$ ls
hello.asm  hello.o  list.lst  obj.o
```

Рис. 3. Полный вариант командной строки `nasm`

4. Компоновщик LD

Передаю объектный файл на обработку компоновщику, чтобы получить исполняемую программу.

С помощью команды `ls` проверяю, что исполняемый файл `hello` был создан. Ключ `-o` задаёт в данном случае имя создаваемого исполняемого файла.

Далее выполняю команду `ld -m elf_i386 obj.o -o main`. Исполняемый файл будет иметь имя `main`, т.к. после ключа `-o` было задано значение `main`. Объектный файл, из которого собран этот исполняемый файл, имеет имя `obj.o`. (рис. 4)

```
[kanechaeva@192 ~]$ cd /home/kanechaeva/work/arch-pc/lab04
[kanechaeva@192 lab04]$ ld -m elf_i386 hello.o -o hello
[kanechaeva@192 lab04]$ ls
hello hello.asm hello.o list.lst obj.o
[kanechaeva@192 lab04]$ ld -m elf_i386 obj.o -o main
[kanechaeva@192 lab04]$ ls
hello hello.asm hello.o list.lst main obj.o
[kanechaeva@192 lab04]$
```

Рис. 4. Передача объектного файла на обработку компоновщику

5. Запуск исполняемого файла

Запускаю исполняемый файл `hello`. (рис. 5)

```
[kanechaeva@192 lab04]$ ./hello
Hello world!
[kanechaeva@192 lab04]$
```

Рис. 5. Запуск файла `hello`

6. Задание для самостоятельной работы

1. В каталоге ~/work/arch-pc/lab04 с помощью команды `cp` создаю копию файла `hello.asm` с именем `lab4.asm`. (рис. 6)

```
[kanechaeva@192 lab04]$ cp hello.asm lab4.asm
[kanechaeva@192 lab04]$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
```

Рис. 6. Создание копии файла

2. С помощью текстового редактора `gedit` вношу изменения в текст программы так, чтобы вместо `Hello world!` на экран выводилась строка с моими фамилией и именем. (рис. 7)

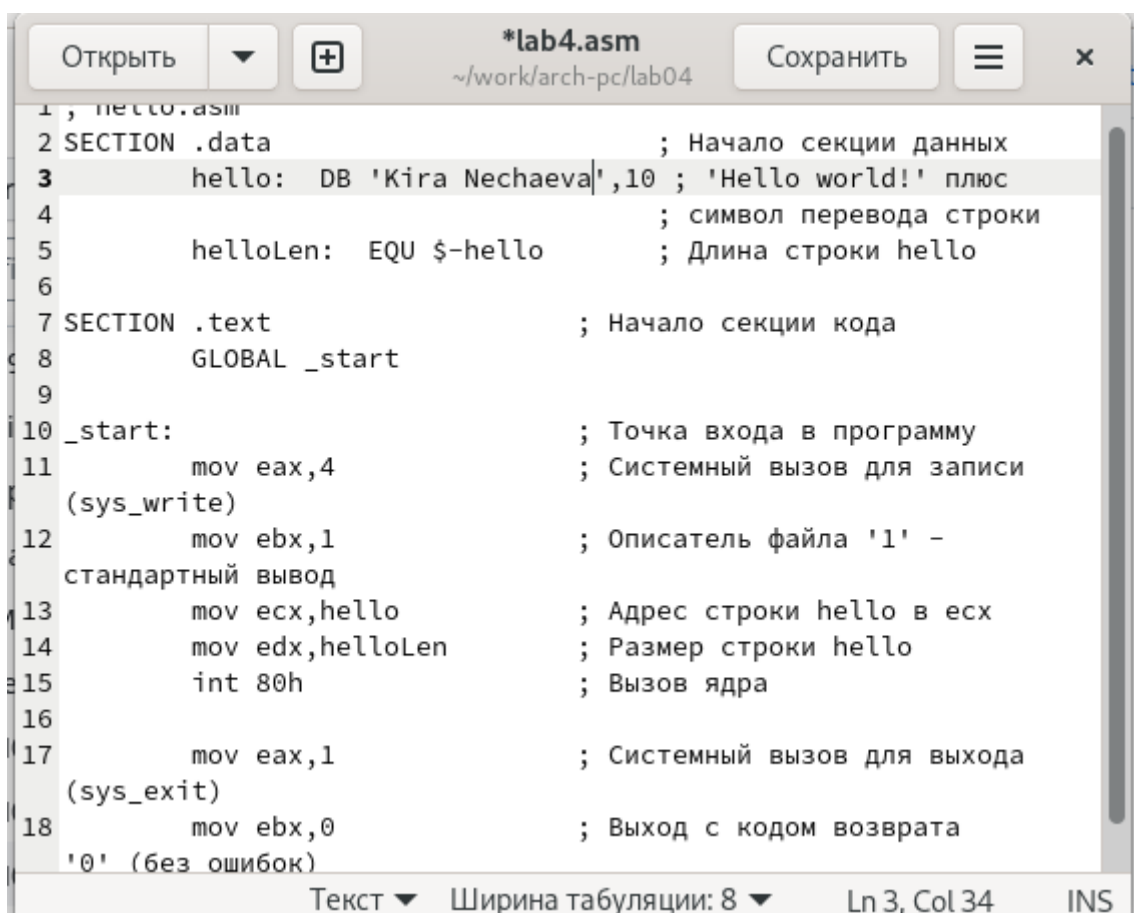


Рис. 7. Изменение программы

3. Транслирую полученный текст программы `lab4.asm` в объектный файл. Выполняю компоновку объектного файла и запускаю получившийся исполняемый файл. (рис. 8)

```

[kanechaeva@192 lab04]$ nasm -f elf lab4.asm
[kanechaeva@192 lab04]$ ls
hello hello.asm hello.o lab4.asm lab4.o list.lst main obj.o
[kanechaeva@192 lab04]$ ld -m elf_i386 lab4.o -o lab4
[kanechaeva@192 lab04]$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o
[kanechaeva@192 lab04]$ ./lab4
Kira Nechaeva
[kanechaeva@192 lab04]$

```

Рис. 8. Трансляция текста программы, компоновка объектного файла и запуск исполняемого файла

4. Копирую файлы hello.asm и lab4.asm в Ваш локальный репозиторий в каталог `~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/`.

Загружаю файлы на Github. (рис. 9)

```

[ks/ lab04
[kanechaeva@192 lab04]$ cp hello.asm /home/kanechaeva/work/study/2023-2024/archcomp/study_2023-2024_arhpc/l
abs/lab04
[kanechaeva@192 lab04]$ cp lab4.asm /home/kanechaeva/work/study/2023-2024/archcomp/study_2023-2024_arhpc/la
bs/lab04
[kanechaeva@192 lab04]$ cd /home/kanechaeva/work/study/2023-2024/archcomp/study_2023-2024_arhpc/labs/lab04
[kanechaeva@192 lab04]$ git add .
[kanechaeva@192 lab04]$ git commit -am 'feat(main): add files lab-4'
[master b38fefd] feat(main): add files lab-4
2 files changed, 38 insertions(+)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
[kanechaeva@192 lab04]$ git push

```

Рис. 9. Копирование файлов и загрузка их на Github

4 Вывод

При выполнении данной лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.

5 Источники

1. ТУИС – Архитектура ЭВМ – [Электронный ресурс] -
<https://esystem.rudn.ru/mod/resource/view.php?id=1030552>