

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**по лабораторной работе № 5**

дисциплина: Архитектура компьютера

Студент: Волгин А.А.

Группа: НПИбд-01-22

**МОСКВА**

2022 г.

**Цель работы:**

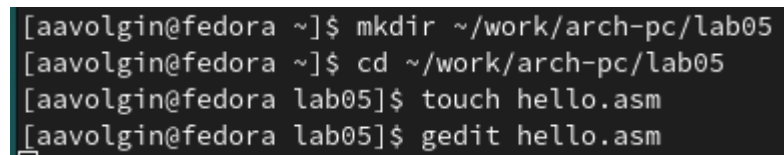
Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

**Порядок выполнения лабораторной работы:**

**1. Программа Hello world!**

Рассмотрим пример простой программы на языке ассемблера NASM. Традиционно первая программа выводит приветственное сообщение Hello world! на экран.

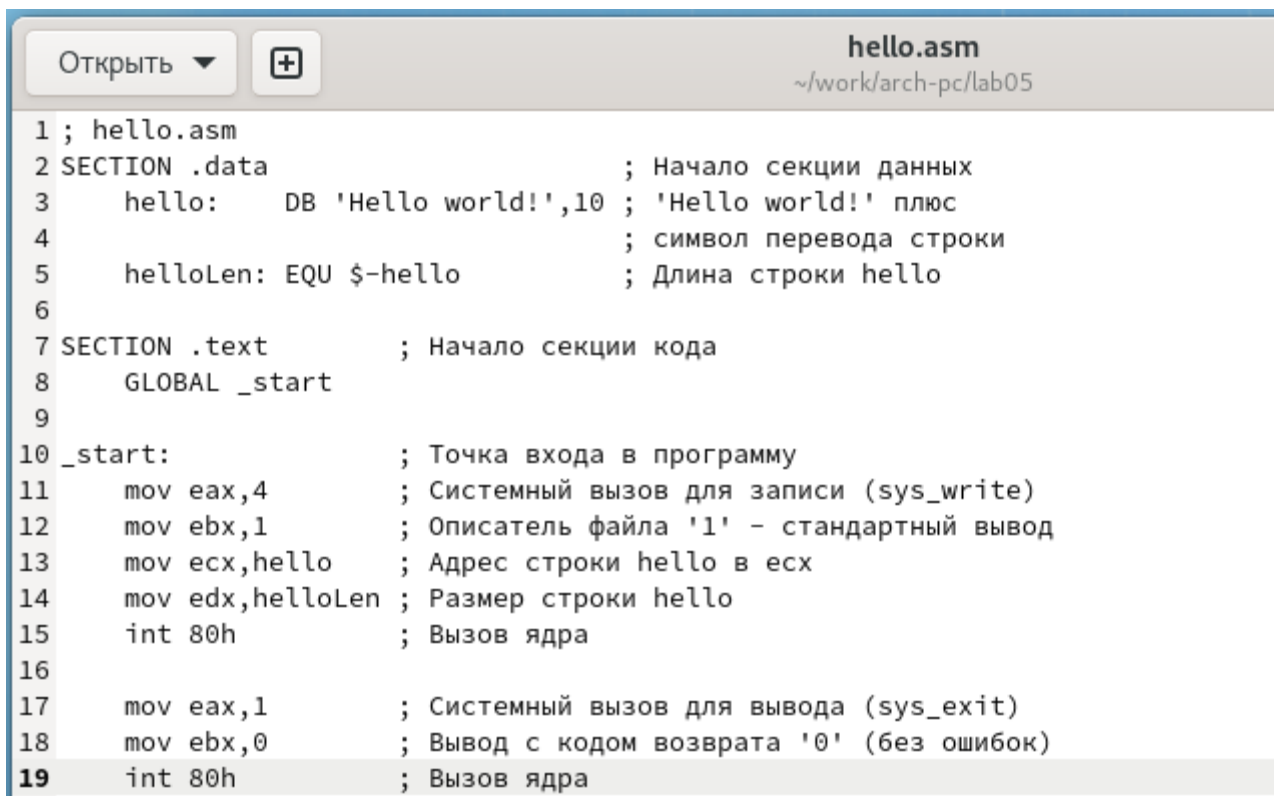
Создадим каталог для работы с программами на языке ассемблера NASM, перейдем в него, создадим текстовый файл с именем hello.asm и откроем его (рис. 1).



```
[aavolgin@fedora ~]$ mkdir ~/work/arch-pc/lab05
[aavolgin@fedora ~]$ cd ~/work/arch-pc/lab05
[aavolgin@fedora lab05]$ touch hello.asm
[aavolgin@fedora lab05]$ gedit hello.asm
```

Рис. 1. Создание файла hello.asm

Введём в него следующий текст (рис. 2).

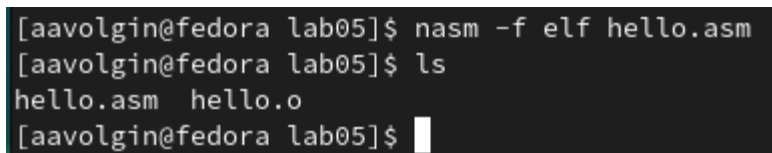


```
1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3     hello:    DB 'Hello world!',10 ; 'Hello world!' плюс
4                ; символ перевода строки
5     helloLen: EQU $-hello      ; Длина строки hello
6
7 SECTION .text                ; Начало секции кода
8     GLOBAL _start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

Рис. 2. Код программы hello

## 2. Транслятор NASM.

Затем скомпилируем программу Hello world! (рис. 3).



```
[aavolgin@fedora lab05]$ nasm -f elf hello.asm
[aavolgin@fedora lab05]$ ls
hello.asm  hello.o
[aavolgin@fedora lab05]$
```

Рис. 3. Компиляция программы

Как видим, образовался объектный файл, значит компиляция прошла успешно.

## 3. Расширенный синтаксис командной строки NASM.

Полный вариант командной строки nasm выглядит следующим образом (рис. 4).

```
nasm [-@ косвенный_файл_настроек] [-o объектный_файл] [-f
↪ формат_объектного_файла] [-l листинг] [параметры...] [--]
↪ исходный_файл
```

Рис. 4. Командная строка nasm

Выполним следующую команду, а затем проверим, что файлы были созданы (рис. 5).

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

Рис. 5. Команда nasm

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).

#### 4. Компоновщик LD.

Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику, затем проверим, что исполняемый файл был создан (рис. 6).

```
[aavolgin@fedora lab05]$ ld -m elf_i386 hello.o -o hello
[aavolgin@fedora lab05]$ ls
hello hello.asm hello.o list.lst obj.o
[aavolgin@fedora lab05]$
```

Рис. 6. Исполняемый файл 1

Затем создадим еще один исполняемый файл, как видим, его название стало main (рис. 7).

```
[aavolgin@fedora lab05]$ ld -m elf_i386 obj.o -o main
[aavolgin@fedora lab05]$ ls
hello hello.asm hello.o list.lst main obj.o
[aavolgin@fedora lab05]$
```

Рис. 7. Исполняемый файл 2

Затем запустим созданный исполняемый файл с помощью следующей команды (рис. 8).

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

Рис. 8. Запуск программы

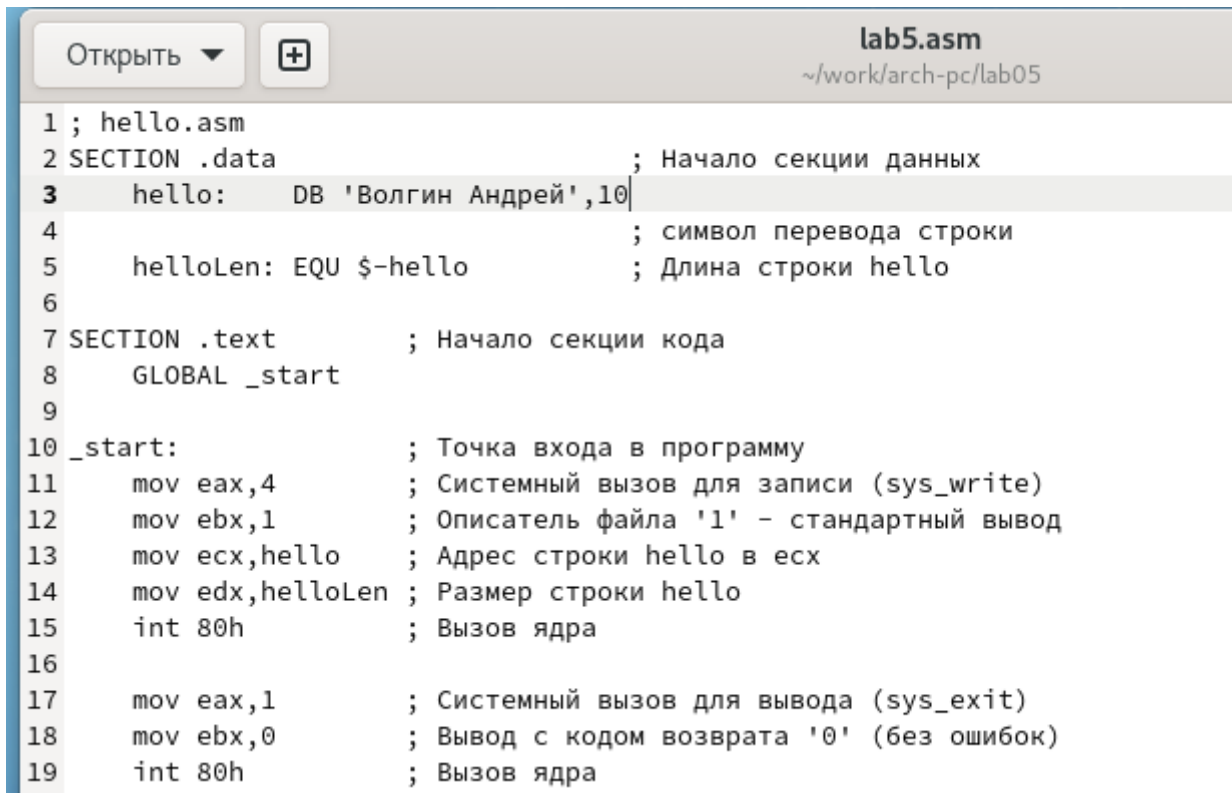
Как видим, все сработало корректно.

#### Порядок выполнения самостоятельной работы:

В том же каталоге создадим копию файла hello.asm с именем lab5.asm и внесем в него изменения, чтобы программа выводила на экран мои фамилию и имя (рис. 9-10).

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

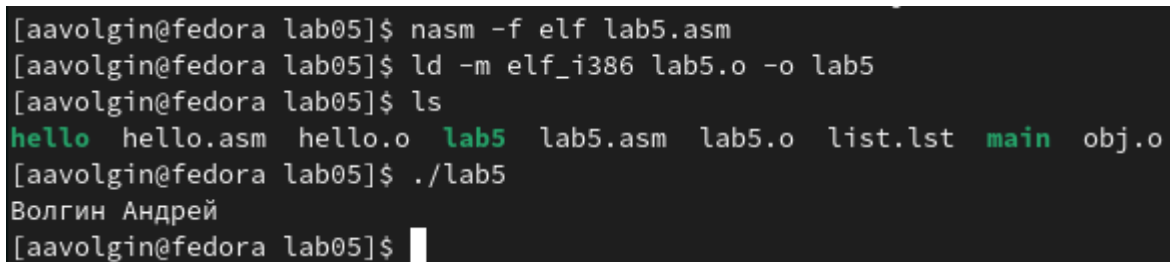
Рис. 9. Копия файла hello.asm



```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello: DB 'Волгин Андрей',10
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 вывода (sys_exit)
18        mov ebx,0 ; Вывод с кодом возврата '0' (без ошибок)
19        int 80h ; Вызов ядра
```

Рис. 10. Изменения в программе

Затем оттранслируем полученный текст программы в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл (рис. 11).



```
[aavolgin@fedora lab05]$ nasm -f elf lab5.asm
[aavolgin@fedora lab05]$ ld -m elf_i386 lab5.o -o lab5
[aavolgin@fedora lab05]$ ls
hello hello.asm hello.o lab5 lab5.asm lab5.o list.lst main obj.o
[aavolgin@fedora lab05]$ ./lab5
Волгин Андрей
[aavolgin@fedora lab05]$
```

Рис. 11. Программа lab5

Как видим, все работает.

Скопируем файлы hello.asm и lab5.asm в локальный репозиторий и загрузим эти файлы на Github (рис. 12-13).

```
[aavolgin@fedora lab05]$ cp hello.asm ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab05/
[aavolgin@fedora lab05]$ cp lab5.asm ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab05/
[aavolgin@fedora lab05]$ cd ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab05/
[aavolgin@fedora lab05]$ ls
hello.asm lab5.asm presentation report
[aavolgin@fedora lab05]$ git push
Everything up-to-date
```

Рис. 12. Копирование файлов

```
[aavolgin@fedora arch-pc]$ git commit -am 'newfiles'
[master edc15ad] newfiles
 2 files changed, 38 insertions(+)
 create mode 100644 labs/lab05/hello.asm
 create mode 100644 labs/lab05/lab5.asm
[aavolgin@fedora arch-pc]$ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 985 байтов | 985.00 КиБ/с, готово.
Всего 6 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:ShooterFromChicago/study_2022-2023_arh-pc.git
 6c45946..edc15ad master -> master
[aavolgin@fedora arch-pc]$
```

Рис. 13. Загрузка на гитхаб

### **Вывод:**

Во время лабораторной работы были освоены процедуры компиляции и сборки программ, написанных на ассемблере NASM.