

# Отчёт по лабораторной работе 5

## Создание и процесс обработки программ на языке ассемблера NASM.

Львов Сергей НПИбд-02-22

### Содержание

1	Цель работы:.....	1
2	Порядок выполнения лабораторной работы:.....	1
3	Порядок выполнения самостоятельной работы:.....	3
4	Вывод:.....	4

### 1 Цель работы:

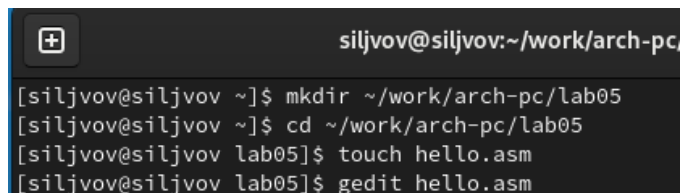
Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

### 2 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.

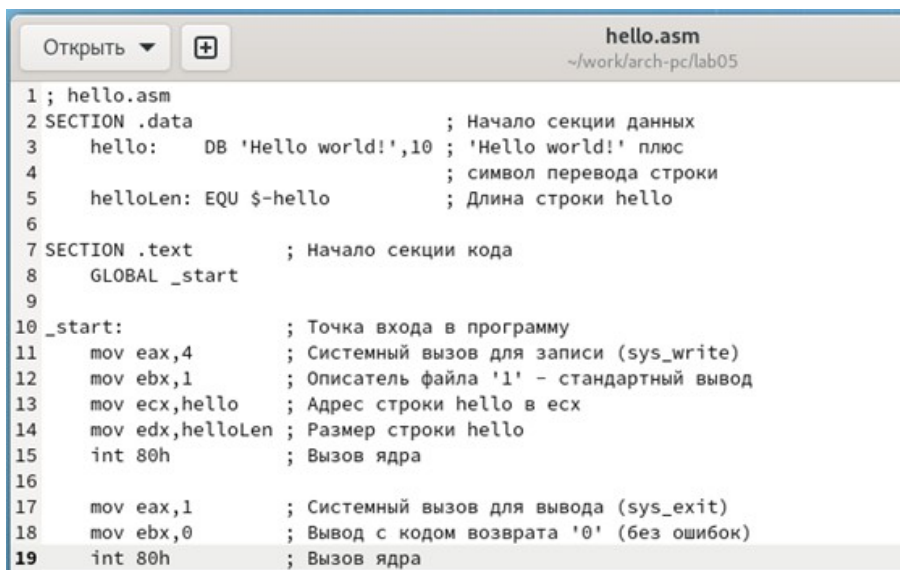
Создадим каталог для работы с программами на языке ассемблера NASM, перейдем в него, создадим текстовый файл с именем hello.asm и откроем его (рис. 1).



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

Рис. 1. Создание файла hello.asm

Введём в него следующий текст (рис. 2).

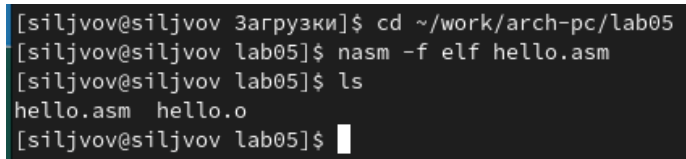


```
1 ; hello.asm
2 SECTION .data          ; Начало секции данных
3     hello:  DB 'Hello world!',10 ; 'Hello world!' плюс
4             ; символ перевода строки
5     helloLen: EQU $-hello      ; Длина строки hello
6
7 SECTION .text          ; Начало секции кода
8     GLOBAL _start
9
10 _start:                ; 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).



```
[siljvov@siljvov Загрузки]$ cd ~/work/arch-pc/lab05
[siljvov@siljvov lab05]$ nasm -f elf hello.asm
[siljvov@siljvov lab05]$ ls
hello.asm  hello.o
[siljvov@siljvov lab05]$
```

Рис. 3. Компиляция программы

Создался объектный файл, значит компиляция прошла успешно.

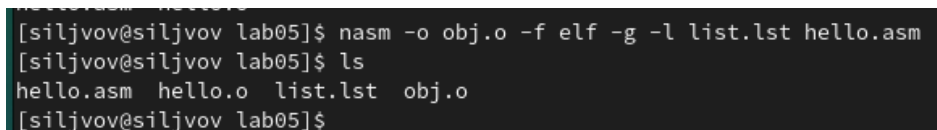
## 3. Расширенный синтаксис командной строки NASM.

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).



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

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

Рис. 6. Исполняемый файл 1

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

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

Рис. 7. Исполняемый файл 2

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

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

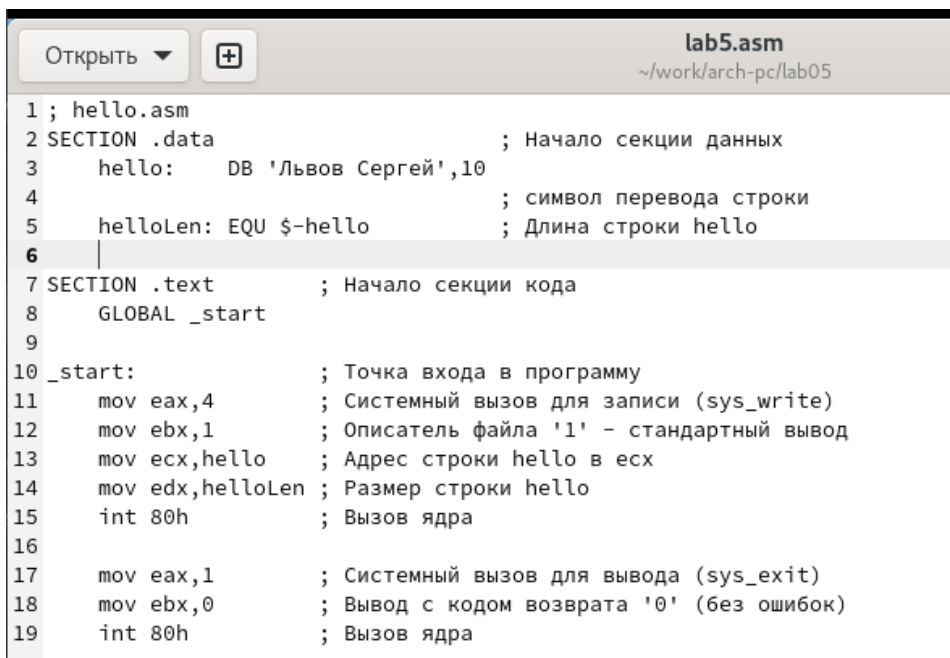
Рис. 8. Запуск программы hello

### 3 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).

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

Рис. 9. Копирование файла hello.asm

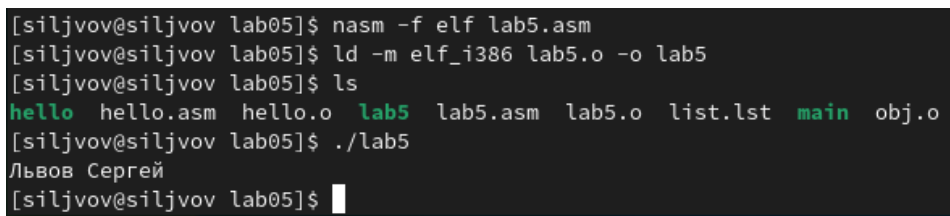


```
lab5.asm
~/work/arch-pc/lab05

1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3     hello:    DB 'Львов Сергей',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 есх
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).



```
[siljvov@siljvov lab05]$ nasm -f elf lab5.asm
[siljvov@siljvov lab05]$ ld -m elf_i386 lab5.o -o lab5
[siljvov@siljvov lab05]$ ls
hello  hello.asm  hello.o  lab5  lab5.asm  lab5.o  list.lst  main  obj.o
[siljvov@siljvov lab05]$ ./lab5
Львов Сергей
[siljvov@siljvov lab05]$
```

Рис. 11. Работа программы lab5

## 4 Вывод:

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