

Отчёт по лабораторной работе 4

Архитектура компьютеров

Ел Вакил Марьям Махмоудовна НБИбд-03-23

Содержание

1	Цель работы	5
2	Задания	6
3	Выполнение лабораторной работы	7
3.1	Программа Hello world	7
3.2	Транслятор NASM	8
3.3	Компоновщик LD	9
3.4	Самостоятельная работа	10
4	Выводы	12

Список иллюстраций

3.1	Создание файла	7
3.2	Код программы	8
3.3	Трансляция программы	9
3.4	Компоновка программы и запуск	10
3.5	Копирование программы	10
3.6	Код программы	10
3.7	Проверка программы lab4.asm	11

Список таблиц

1 Цель работы

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

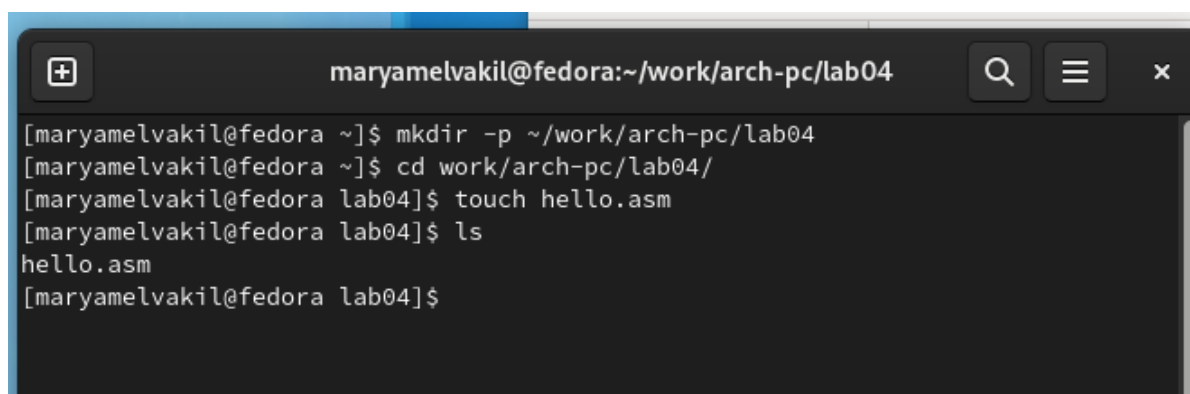
2 Задания

1. Изучить пример программы Hello world
2. Опробовать и освоить процесс компиляции программы
3. Внести изменения в программу

3 Выполнение лабораторной работы

3.1 Программа Hello world

Я создала каталог lab04, используя команду `mkdir`, затем перешла в него с помощью команды `cd`. Внутри этого каталога я создала файл `hello.asm`.

A screenshot of a terminal window with a dark background. The window title bar shows the user 'maryamelvakil' on a 'fedora' machine, with the current directory being '~ /work/arch-pc/lab04'. The terminal contains the following commands and their outputs:

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

Рис. 3.1: Создание файла

Далее, я открыла файл в редакторе и написала код программы в соответствии с заданием.

Открыть ▾

+

hello.asm
~/work/arch-pc/lab04

```
; 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. 3.2: Код программы

3.2 Транслятор NASM

Затем я транслировала файл с помощью команды `nasm`, указав опцию `-f`. Ключ `-f` указывает транслятору, что нужно создать бинарные файлы в формате ELF.

Если текст программы был набран без ошибок, то транслятор преобразует текст программы из файла `hello.asm` в объектный код, который будет записан в файл `hello.o`. Таким образом, имена всех файлов формируются из имени входного файла и расширения по умолчанию. В случае возникновения ошибок, объектный файл не будет создан, а при запуске транслятора появятся сообщения об ошибках или предупреждениях.

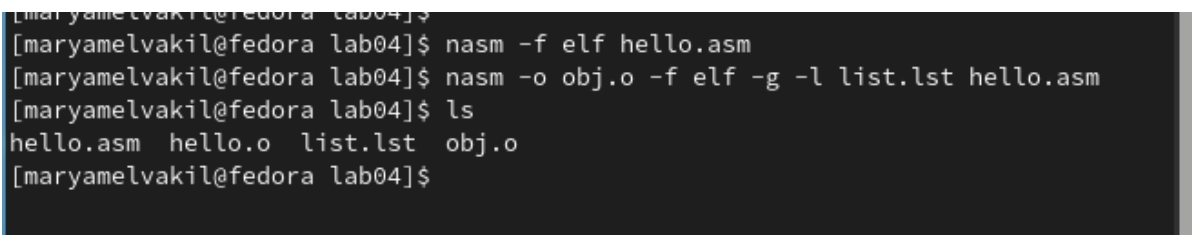
В результате, я получила объектный файл `hello.o`.

Полный вариант командной строки `nasm` выглядит следующим образом:

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

Затем транслировала файл с помощью команды `nasm` и дополнительными опциями: `-o`, `-g`, `-l`. Опция `-o` позволяет задать имя объектного файла. Опция `-g` добавляет отладочную информацию. Опция `-l` создает файл листинга.

В итоге, я получила файл листинга `list.lst`, объектный файл `obj.o`, и в программу была добавлена отладочная информация.



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

Рис. 3.3: Трансляция программы

3.3 Компоновщик LD

Выполнила компоновку командой `ld` и получила исполняемый файл.

Еще раз выполнила компоновку для объектного файла `obj.o` и получила исполняемый файл `main`.

Затем я запустила исполняемые файлы.

```

[maryamelvakil@fedora lab04]$
[maryamelvakil@fedora lab04]$ ld -m elf_i386 hello.o -o hello
[maryamelvakil@fedora lab04]$ ls
hello hello.asm hello.o list.lst obj.o
[maryamelvakil@fedora lab04]$ ld -m elf_i386 obj.o -o main
[maryamelvakil@fedora lab04]$ ls
hello hello.asm hello.o list.lst main obj.o
[maryamelvakil@fedora lab04]$ ./hello
Hello world!
[maryamelvakil@fedora lab04]$ ./main
Hello world!
[maryamelvakil@fedora lab04]$

```

Рис. 3.4: Компоновка программы и запуск

3.4 Самостоятельная работа

Скопировала программу в файл lab4.asm.

```

[maryamelvakil@fedora lab04]$
[maryamelvakil@fedora lab04]$ cp hello.asm lab4.asm
[maryamelvakil@fedora lab04]$ ls
hello hello.asm hello.o lab4.asm list.lst main obj.o
[maryamelvakil@fedora lab04]$

```

Рис. 3.5: Копирование программы

Изменила сообщение Hello world на свое имя.

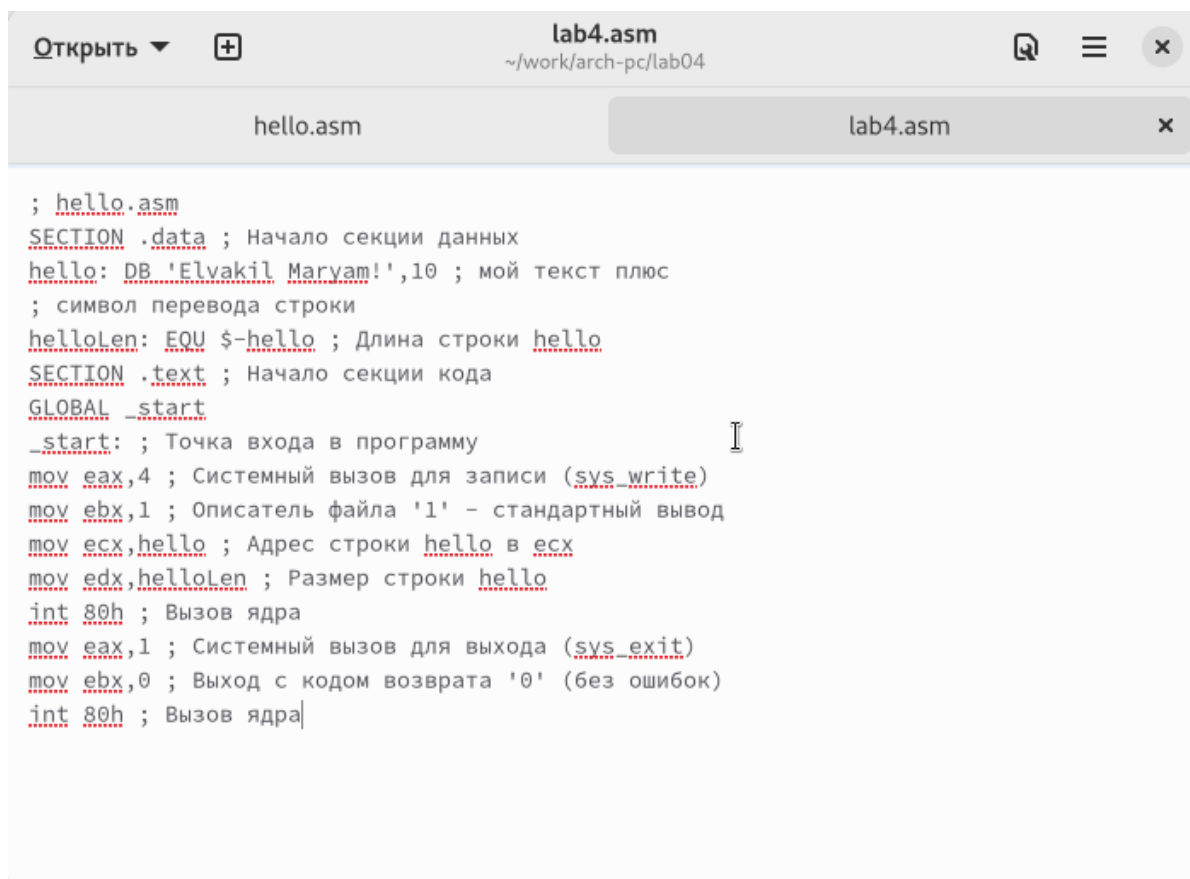
```

[maryamelvakil@fedora lab04]$
[maryamelvakil@fedora lab04]$ cp hello.asm lab4.asm
[maryamelvakil@fedora lab04]$ ls
hello hello.asm hello.o lab4.asm list.lst main obj.o
[maryamelvakil@fedora lab04]$

```

Рис. 3.6: Код программы

Оттранслировала полученный текст программы lab4.asm в объектный файл. Выполнила компоновку объектного файла и запустила получившийся исполняемый файл.



```
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Elvakil Maryam!',10 ; мой текст 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. 3.7: Проверка программы lab4.asm

4 Выводы

Освоили процесс компиляции и сборки программ, написанных на ассемблере `nasm`.