

Отчёт по лабораторной работе 4

Архитектура компьютера

Хаоладар Шаханеоядж НПИ-01-24

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Программа Hello world!	6
2.2	Транслятор NASM	7
2.3	Расширенный синтаксис командной строки NASM	8
2.4	Компоновщик LD	8
2.5	Запуск исполняемого файла	9
2.6	Задание для самостоятельной работы	9
3	Выводы	11

Список иллюстраций

2.1	Создан каталог для работы и файл для программы	6
2.2	Программа в файле hello.asm	7
2.3	Трансляция программы	7
2.4	Трансляция программы с дополнительными опциями	8
2.5	Компоновка программы	8
2.6	Компоновка программы	9
2.7	Запуск программы	9
2.8	Программа в файле lab4.asm	10
2.9	Проверка программы lab4.asm	10

Список таблиц

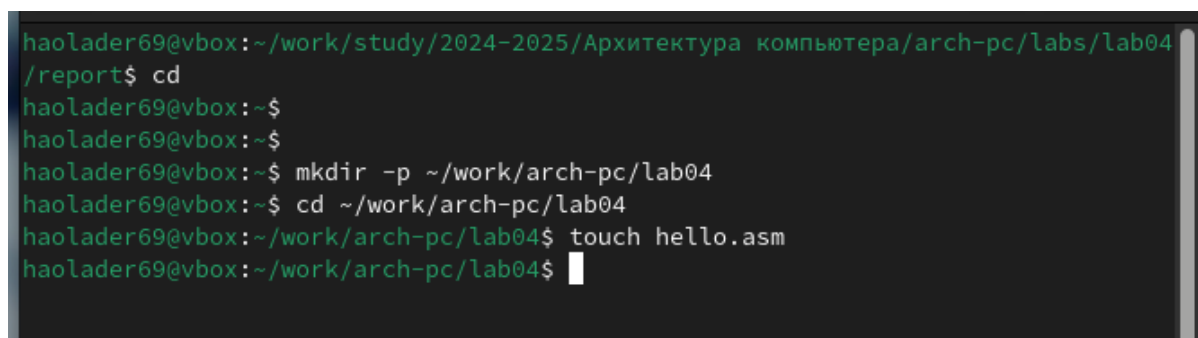
1 Цель работы

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

2 Выполнение лабораторной работы

2.1 Программа Hello world!

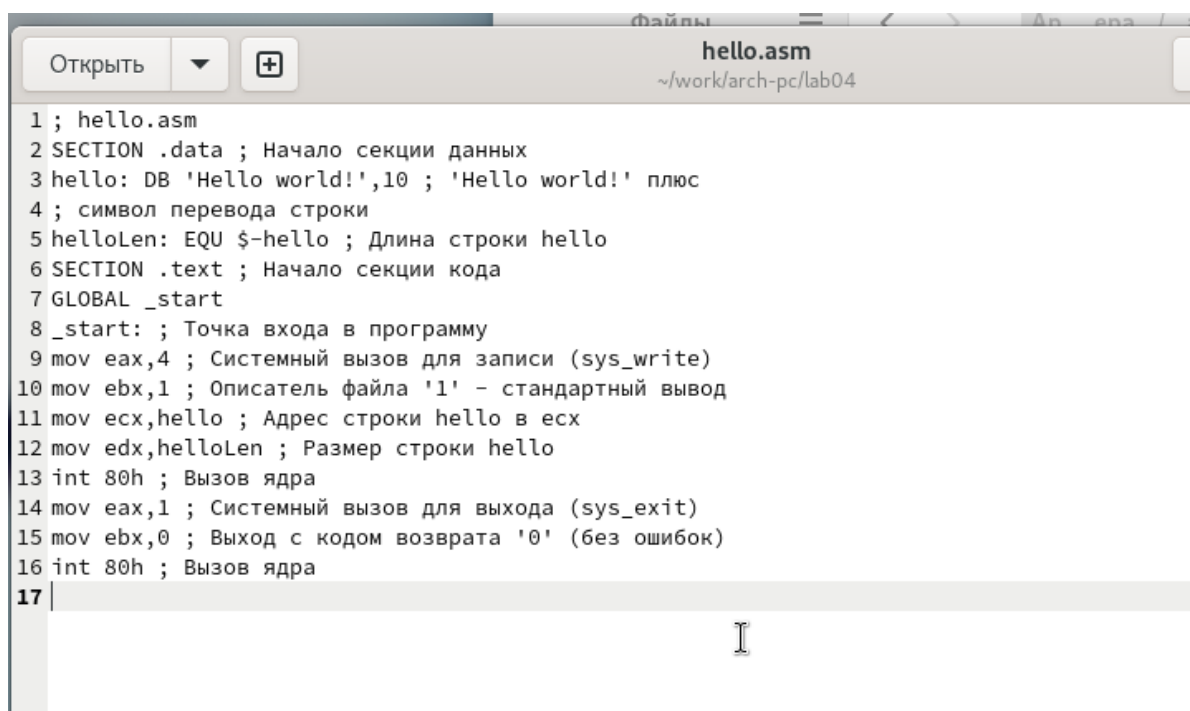
Создал каталог lab04 командой `mkdir`, перешел в него с помощью команды `cd` и создал файл `hello.asm`, в который напишу программу. Убеждаюсь с помощью команды `ls`, что создал файл.



```
haolader69@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04/report$ cd
haolader69@vbox:~$
haolader69@vbox:~$
haolader69@vbox:~$ mkdir -p ~/work/arch-pc/lab04
haolader69@vbox:~$ cd ~/work/arch-pc/lab04
haolader69@vbox:~/work/arch-pc/lab04$ touch hello.asm
haolader69@vbox:~/work/arch-pc/lab04$
```

Рис. 2.1: Создан каталог для работы и файл для программы

Написал программу по заданию на языке ассемблера.



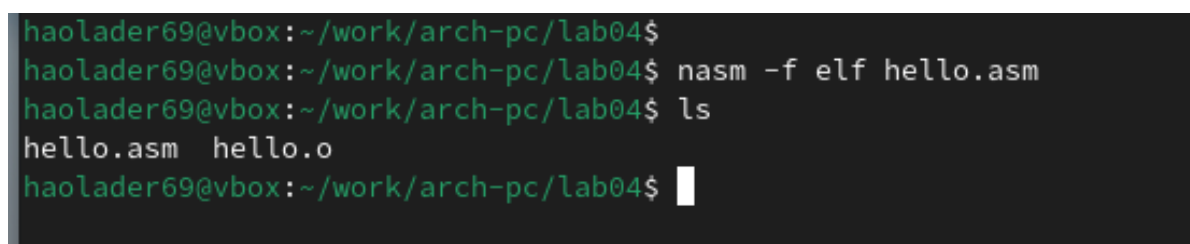
```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5 helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для записи (sys_write)
10 mov ebx,1 ; Описатель файла '1' - стандартный вывод
11 mov ecx,hello ; Адрес строки hello в ecx
12 mov edx,helloLen ; Размер строки hello
13 int 80h ; Вызов ядра
14 mov eax,1 ; Системный вызов для выхода (sys_exit)
15 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16 int 80h ; Вызов ядра
17
```

Рис. 2.2: Программа в файле hello.asm

2.2 Транслятор NASM

NASM превращает текст программы в объектный код. Если текст программы набран без ошибок, то транслятор преобразует текст программы из файла hello.asm в объектный код, который запишется в файл hello.o.

Транслировал файл командой `nasm`. Получился объектный файл hello.o.



```
haolader69@vbox:~/work/arch-pc/lab04$
haolader69@vbox:~/work/arch-pc/lab04$ nasm -f elf hello.asm
haolader69@vbox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
haolader69@vbox:~/work/arch-pc/lab04$
```

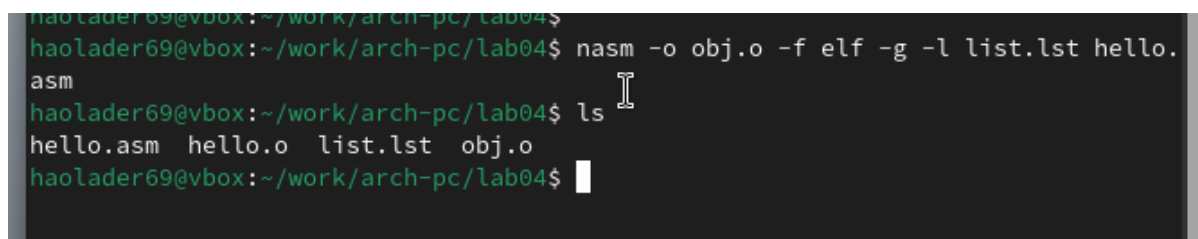
Рис. 2.3: Трансляция программы

2.3 Расширенный синтаксис командной строки NASM

Полный вариант командной строки `nasm` выглядит следующим образом:

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

Транслировал файл командой `nasm` с дополнительными опциями. С опцией `-l` Получил файл листинга `list.lst`, с опцией `-f` объектный файл `obj.o`, с опцией `-g` в программу добавилась отладочная информация.



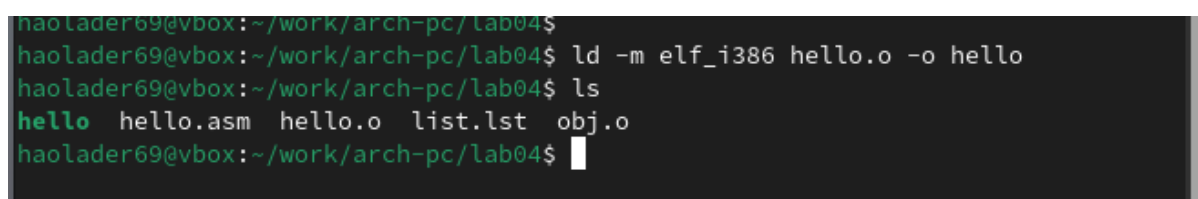
```
hao1ader69@vbox:~/work/arch-pc/lab04$  
hao1ader69@vbox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm  
hao1ader69@vbox:~/work/arch-pc/lab04$ ls  
hello.asm hello.o list.lst obj.o  
hao1ader69@vbox:~/work/arch-pc/lab04$
```

Рис. 2.4: Трансляция программы с дополнительными опциями

2.4 Компоновщик LD

Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику.

Выполнил команду `ld` и получил исполняемый файл `hello` из объектного файла `hello.o`.



```
hao1ader69@vbox:~/work/arch-pc/lab04$  
hao1ader69@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello  
hao1ader69@vbox:~/work/arch-pc/lab04$ ls  
hello hello.asm hello.o list.lst obj.o  
hao1ader69@vbox:~/work/arch-pc/lab04$
```

Рис. 2.5: Компоновка программы

Еще раз выполнил команду `ld` для объектного файла `obj.o` и получил исполняемый файл `main`.


```
haolader69@vbox:~/work/arch-pc/lab04$  
haolader69@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main  
haolader69@vbox:~/work/arch-pc/lab04$ ls  
hello hello.asm hello.o list.lst main obj.o  
haolader69@vbox:~/work/arch-pc/lab04$
```

Рис. 2.6: Компоновка программы

2.5 Запуск исполняемого файла

Запустил исполняемые файлы.

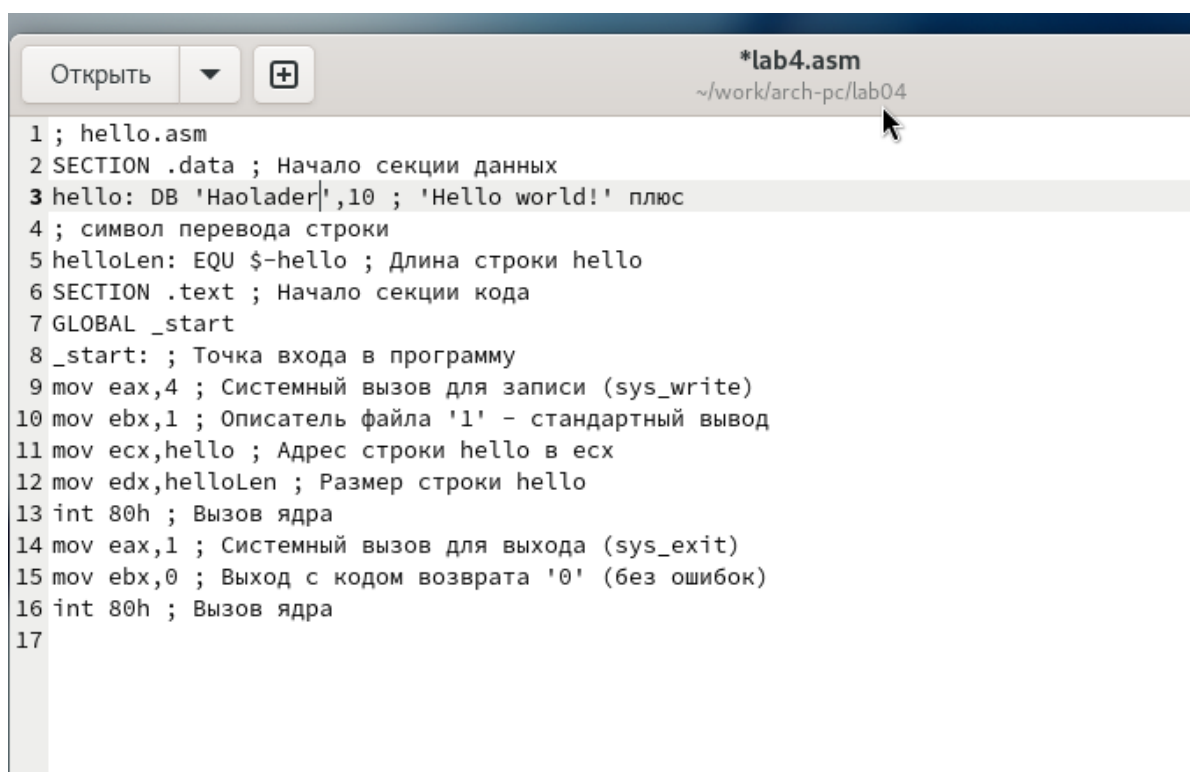
```
haolader69@vbox:~/work/arch-pc/lab04$  
haolader69@vbox:~/work/arch-pc/lab04$ ./hello  
Hello world!  
haolader69@vbox:~/work/arch-pc/lab04$
```

Рис. 2.7: Запуск программы

2.6 Задание для самостоятельной работы

Скопировал файл hello.asm в файл lab4.asm.

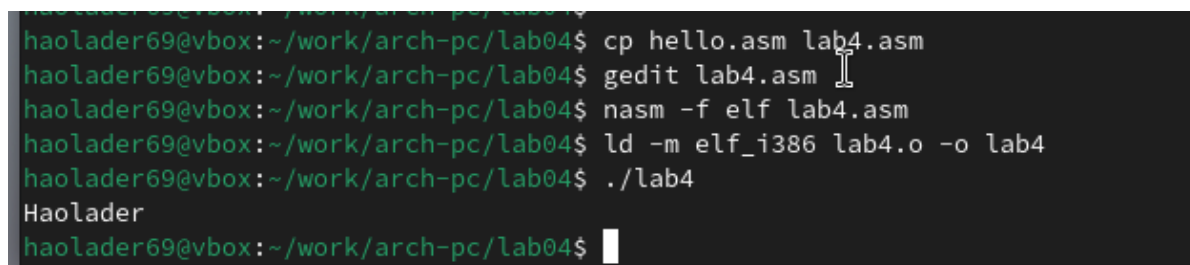
Изменил сообщение Hello world на свое имя.



```
Открыть ▼ + *lab4.asm
~/work/arch-pc/lab04
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Haolader|',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5 helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для записи (sys_write)
10 mov ebx,1 ; Описатель файла '1' - стандартный вывод
11 mov ecx,hello ; Адрес строки hello в ecx
12 mov edx,helloLen ; Размер строки hello
13 int 80h ; Вызов ядра
14 mov eax,1 ; Системный вызов для выхода (sys_exit)
15 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16 int 80h ; Вызов ядра
17
```

Рис. 2.8: Программа в файле lab4.asm

Запустил программу и проверил.



```
haolader69@vbox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
haolader69@vbox:~/work/arch-pc/lab04$ gedit lab4.asm
haolader69@vbox:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
haolader69@vbox:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
haolader69@vbox:~/work/arch-pc/lab04$ ./lab4
Haolader
haolader69@vbox:~/work/arch-pc/lab04$
```

Рис. 2.9: Проверка программы lab4.asm

3 Выводы

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