

Шаблон отчёта по лабораторной работе №5

Дисциплина: архитектура компьютера

Пронякова Ольга Максимовна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Создание программы Hello world!	9
4.2	Работа с транслятором NASM	10
4.3	Работа с расширенным синтаксисом командной строки NASM . .	11
4.4	Работа с компоновщиком LD	11
4.5	Запуск исполняемого файла	12
5	Задание для самостоятельной работы	13
6	Выводы	15
	Список литературы	16

Список иллюстраций

4.1	Создание пустого файла	9
4.2	Открытие файла в текстовом редакторе	9
4.3	Заполнение файла	10
4.4	Комптация текста программы	10
4.5	Комптация текста программы	11
4.6	Передача объектного файла на обработку компоновщику	11
4.7	Передача объектного файла на обработку компоновщику	12
4.8	Запуск исполняемого файла	12
5.1	Создание копии файла	13
5.2	Компация текста программы	13
5.3	Передача объектного файла на обработку компоновщику	13
5.4	Запуск исполняемого файла	14

Список таблиц

1 Цель работы

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

2 Задание

1. Создание программы Hello world!
2. Работа с транслятором NASM
3. Работа с расширенным синтаксисом командной строки NASM
4. работа с компоновщиком LD
5. Запуск исполняемого файла
6. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Основными функциональными элементами любой электронно-вычислительной машины (ЭВМ) являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской (системной) плате. Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора (ЦП) входят следующие устройства: • арифметико-логическое устройство (АЛУ) — выполняет логические и арифметические действия, необходимые для обработки информации, хранящейся в памяти; • устройство управления (УУ) — обеспечивает управление и контроль всех устройств компьютера; • регистры — сверхбыстрая оперативная память небольшого объёма, входящая в состав процессора, для временного хранения промежуточных результатов выполнения инструкций; регистры процессора делятся на два типа: регистры общего назначения и специальные регистры. Для того, чтобы писать программы на ассемблере, необходимо знать, какие регистры процессора существуют и как их можно использовать. Большинство команд в программах написанных на ассемблере используют регистры в качестве операндов. Практически все команды представляют собой преобразование данных хранящихся в регистрах процессора, это например пересылка данных между регистрами или между регистрами и памятью, преобразование (арифме-

тические или логические операции) данных хранящихся в регистрах. Доступ к регистрам осуществляется не по адресам, как к основной памяти, а по именам. Каждый регистр процессора архитектуры x86 имеет свое название, состоящее из 2 или 3 букв латинского алфавита. В качестве примера приведем названия основных регистров общего назначения (именно эти регистры чаще всего используются при написании программ):

- RAX, RCX, RDX, RBX, RSI, RDI — 64-битные
- EAX, ECX, EDX, EBX, ESI, EDI — 32-битные
- AX, CX, DX, BX, SI, DI — 16-битные
- AH, AL, CH, CL, DH, DL, BH, BL — 8-битные (половинки 16-битных регистров).

Например, AH (high AX) — старшие 8 бит регистра AX, AL (low AX) — младшие 8 бит регистра AX.

4 Выполнение лабораторной работы

4.1 Создание программы Hello world!

С помощью `cd` перемещаюсь в каталог, в котором буду работать. Создаю в текущем каталоге текстовый файл `hello.asm` с помощью `touch` (рис. 4.1).

```
olga@olga-VirtualBox:~$ cd ~/work/study/2022-2023/"Архитектура компьютера"/arch-  
pc/labs/lab05  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ touch hello.asm
```

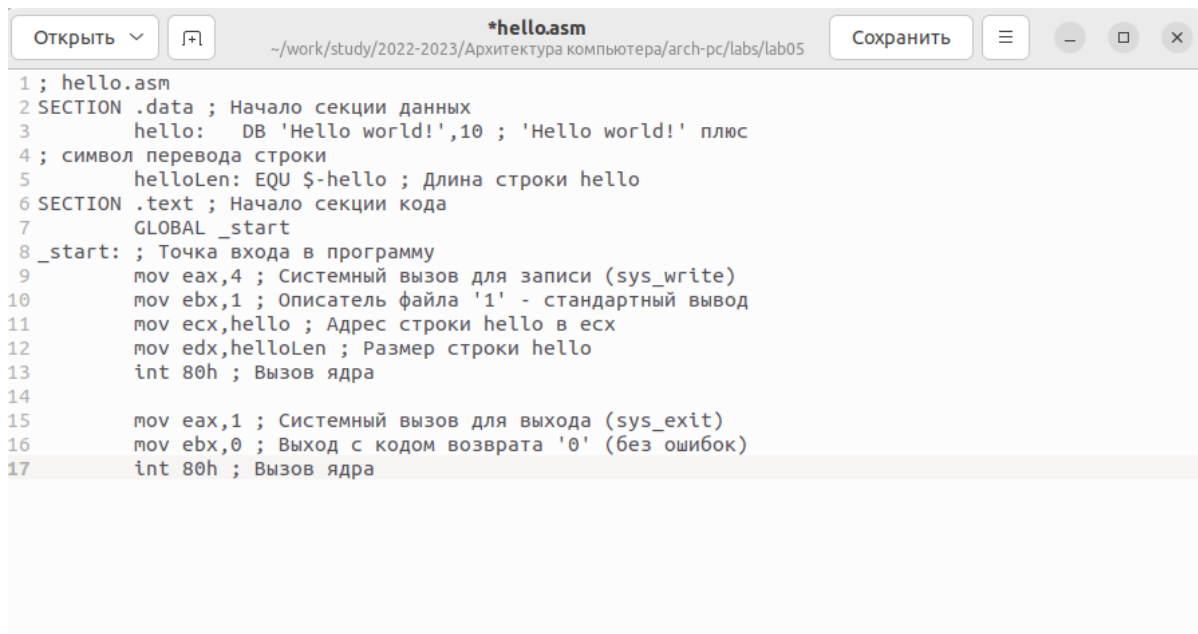
Рис. 4.1: Создание пустого файла

Открываю созданный файл в текстовом редакторе `gedit` (рис. 4.2).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ gedit hello.asm
```

Рис. 4.2: Открытие файла в текстовом редакторе

Заполняю файл, вставляя в него программу для вывода “Hello world!” (рис. 4.3).

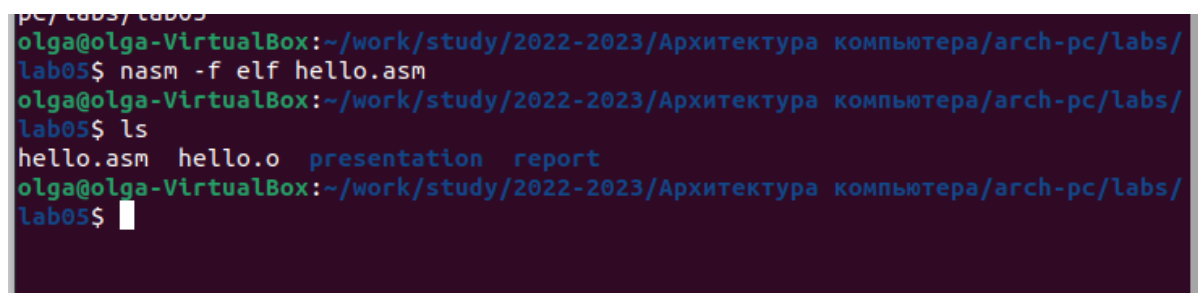


```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3     hello:    DB 'Hello world!',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5     helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7     GLOBAL _start
8 _start: ; Точка входа в программу
9     mov eax,4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

Рис. 4.3: Заполнение файла

4.2 Работа с транслятором NASM

Превращаю текст программы в объектный код. Для компиляции текста программы «Hello World» использую команду `nasm -f elf hello.asm`. Далее проверяю правильность выполнения команды с помощью `ls` (рис. 4.4).



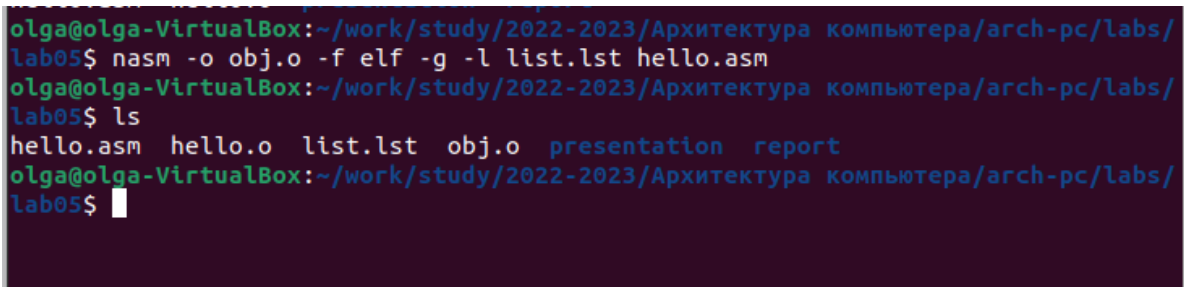
```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ nasm -f elf hello.asm
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ ls
hello.asm  hello.o  presentation  report
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$
```

Рис. 4.4: Компиляция текста программы

4.3 Работа с расширенным синтаксисом командной строки

NASM

Ввожу команду, которая скомпилирует файл `hello.asm` в файл `obj.o`, также с помощью `-l` будет создан файл `list.lst`. Проверяю правильность выполнения команды с помощью `ls` (рис. 4.5).

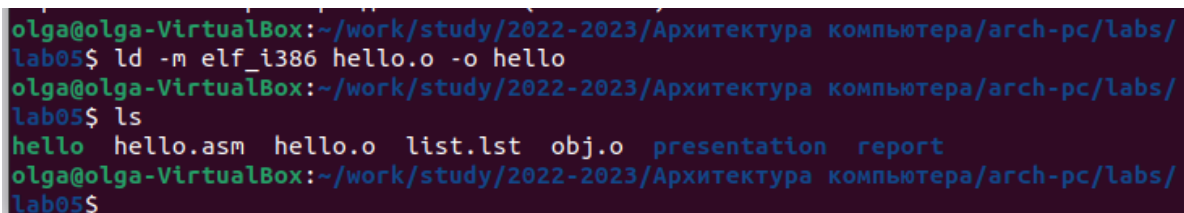


```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ nasm -o obj.o -f elf -g -l list.lst hello.asm
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ ls
hello.asm  hello.o  list.lst  obj.o  presentation  report
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$
```

Рис. 4.5: Компиляция текста программы

4.4 Работа с компоновщиком LD

Передаю объектный файл `hello.o` на обработку компоновщику `LD`, чтобы получить исполняемый файл `hello`. Проверяю правильность выполнения команды с помощью `ls` (рис. 4.6).



```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ ld -m elf_i386 hello.o -o hello
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ ls
hello  hello.asm  hello.o  list.lst  obj.o  presentation  report
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$
```

Рис. 4.6: Передача объектного файла на обработку компоновщику

Выполняю следующую команду. Объектный файл, из которого собран этот исполняемый файл, имеет имя `obj.o`. Проверяю правильность выполнения команды с помощью `ls` (рис. 4.7).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ ld -m elf_i386 obj.o -o main  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ ls  
hello hello.asm hello.o list.lst main obj.o presentation report  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$
```

Рис. 4.7: Передача объектного файла на обработку компоновщику

4.5 Запуск исполняемого файла

Запускаю на выполнение исполняемый файл hello (рис. 4.8).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ ./hello  
Hello world!  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$
```

Рис. 4.8: Запуск исполняемого файла

5 Задание для самостоятельной работы

Создаю с помощью `cp` копию файла `hello.asm` с именем `lab5.asm` и открываю его в текстовом редакторе `gedit` (рис. 5.1).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ cp hello.asm lab5.asm  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ gedit lab5.asm
```

Рис. 5.1: Создание копии файла

Вношу в программу изменения, чтобы она выводила мои имя и фамилию. Далее компилирую текст программы в объектный файл. Проверяю правильность выполнения команды с помощью `ls` (рис. 5.2).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ nasm -f elf lab5.asm  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ ls  
hello      hello.o    lab5.o     main      presentation  
hello.asm  lab5.asm  list.lst  obj.o     report
```

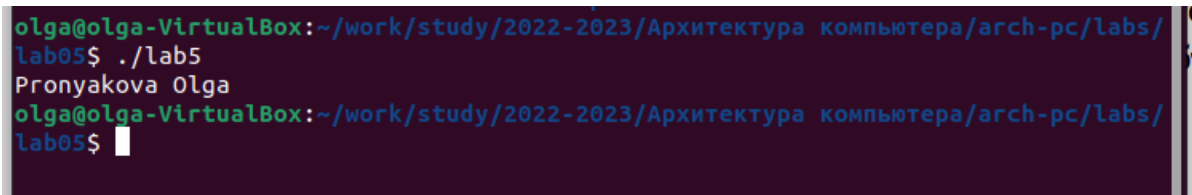
Рис. 5.2: Компиляция текста программы

Передаю объектный файл `lab5.o` на обработку компоновщику `LD`, чтобы получить исполняемый файл `lab5` (рис. 5.3).

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ ld -m elf_i386 lab5.o -o lab5  
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
lab05$ ls  
hello      hello.o    lab5.asm  list.lst  obj.o      report  
hello.asm  lab5       lab5.o    main      presentation
```

Рис. 5.3: Передача объектного файла на обработку компоновщику

Запускаю исполняемый файл (рис. 5.4).

A terminal window with a dark purple background. The prompt is 'olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-рс/labs/lab05\$'. The user enters './lab5'. The output is 'Pronyakova Olga'. The prompt returns to 'olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-рс/labs/lab05\$' with a cursor at the end.

```
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-рс/labs/
lab05$ ./lab5
Pronyakova Olga
olga@olga-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-рс/labs/
lab05$
```

Рис. 5.4: Запуск исполняемого файла

Добавляем файлы на Github.

6 Выводы

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

Список литературы

1. Архитектура ЭВМ