

Отчёт по лабораторной работе №4

Создание и процесс обработки программ на языке ассемблера NASM

Исупов Олег Денисович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Задания для самостоятельной работы	9
4	Выводы	12

Список иллюстраций

2.1	Создание каталога	6
2.2	Переход в каталог	6
2.3	Создание текстового файла	6
2.4	Открытие файла	6
2.5	Ввод текста	7
2.6	Компиляция	7
2.7	Выполнение команды	7
2.8	Передача на обработку	8
2.9	Проверка	8
2.10	Выполнение команды	8
2.11	Запуск файла	8
3.1	Создание копии	9
3.2	Открытие файла	9
3.3	Редактирование	10
3.4	Компиляция	10
3.5	Выполнение команды	10
3.6	Передача на обработку	10
3.7	Запуск файла	11
3.8	Копирование файлов	11
3.9	Загрузка файлов на Github	11

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

1. Создайте каталог для работы с программами на языке ассемблера NASM:

```
odisupov@odisupov-VirtualBox:~$ mkdir -p ~/work/arch-pc/lab04
```

Рис. 2.1: Создание каталога

2. Перейдите в созданный каталог

```
odisupov@odisupov-VirtualBox:~$ cd ~/work/arch-pc/lab04
```

Рис. 2.2: Переход в каталог

3. Создайте текстовый файл с именем hello.asm

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ touch hello.asm
```

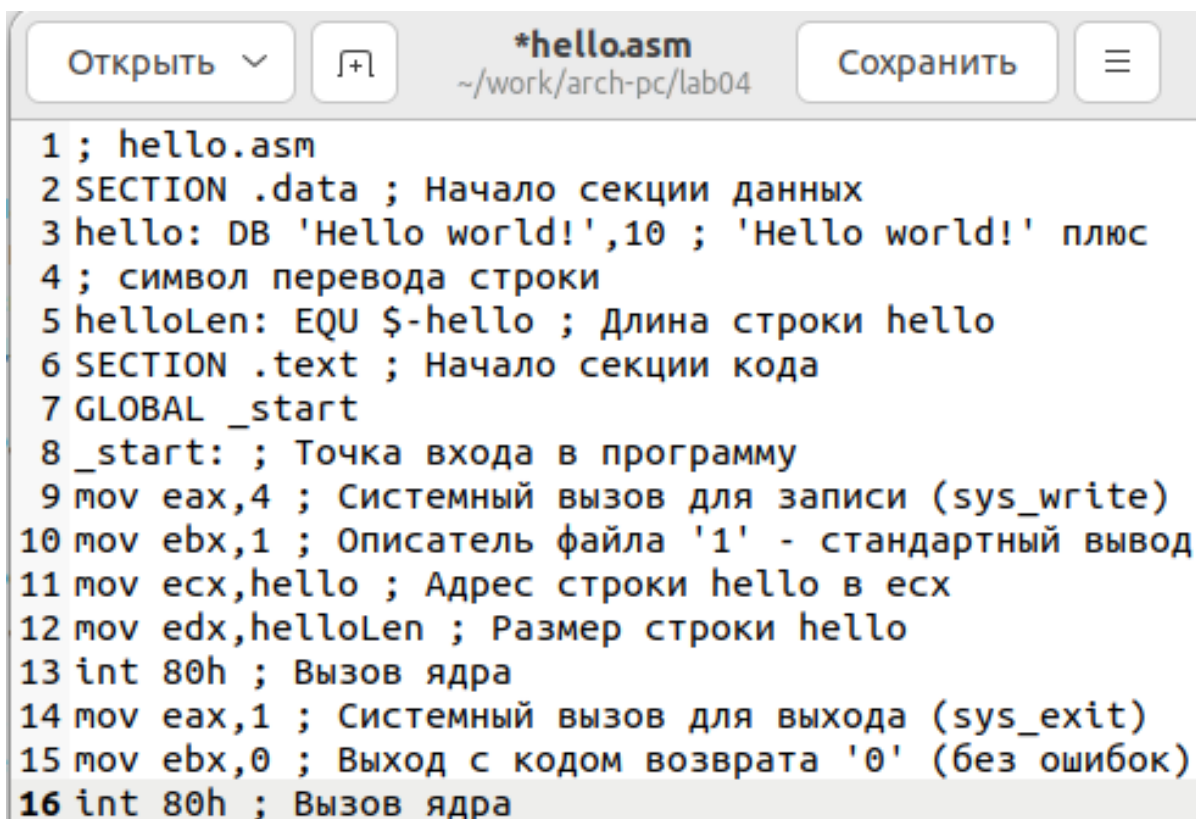
Рис. 2.3: Создание текстового файла

4. Откройте этот файл с помощью любого текстового редактора, например, gedit

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ gedit hello.asm
```

Рис. 2.4: Открытие файла

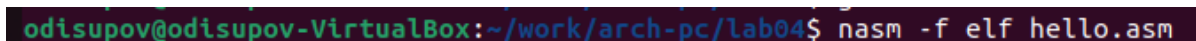
5. И введите в него следующий текст:



```
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
```

Рис. 2.5: Ввод текста

6. NASM превращает текст программы в объектный код. Например, для компиляции приведённого выше текста программы «Hello World» необходимо написать:



```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ nasm -f elf hello.asm
```

Рис. 2.6: Компиляция

7. Выполните следующую команду:



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

Рис. 2.7: Выполнение команды

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

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
```

Рис. 2.8: Передача на обработку

9. С помощью команды `ls` проверьте, что исполняемый файл `hello` был создан

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ ls  
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис. 2.9: Проверка

10. Выполните следующую команду:

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
```

Рис. 2.10: Выполнение команды

11. Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге, можно, набрав в командной строке:

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

Рис. 2.11: Запуск файла

3 Задания для самостоятельной работы

1. В каталоге ~/work/arch-pc/lab04 с помощью команды cp создайте копию файла hello.asm с именем lab4.asm

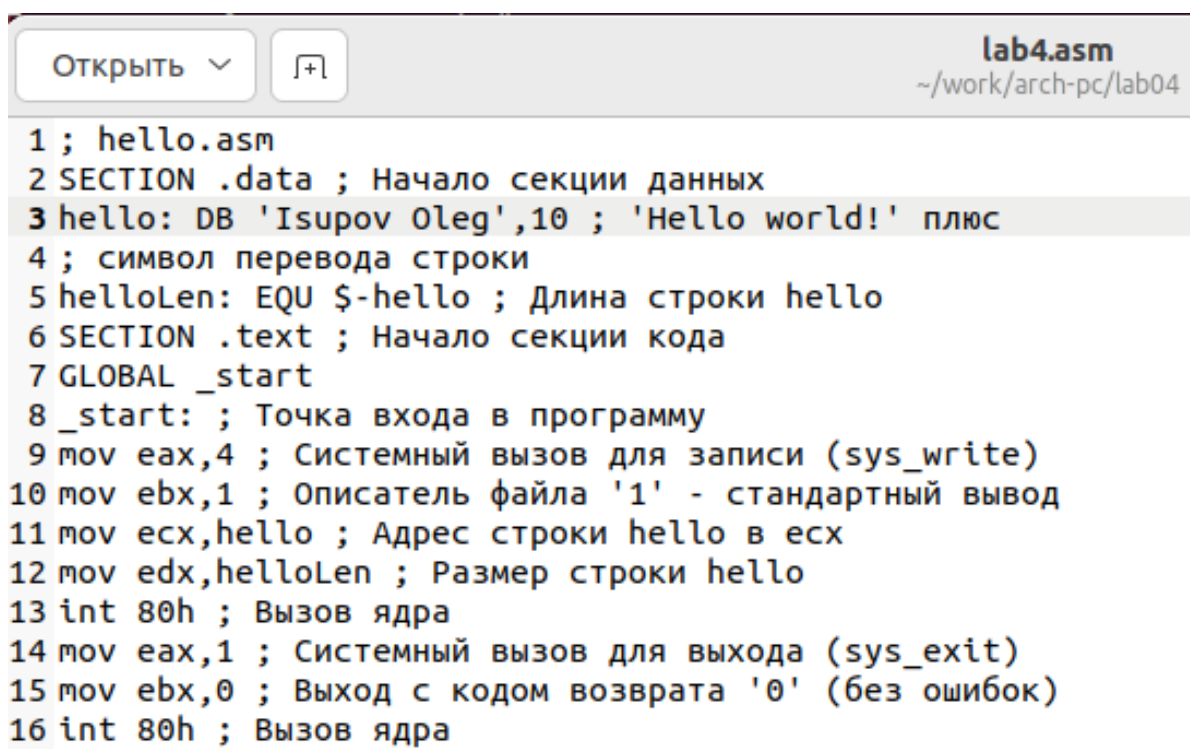
```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
```

Рис. 3.1: Создание копии

2. С помощью любого текстового редактора внесите изменения в текст программы в файле lab4.asm так, чтобы вместо Hello world! на экран выводилась строка с вашими фамилией и именем.

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ gedit lab4.asm
```

Рис. 3.2: Открытие файла



```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Isupov Oleg',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
```

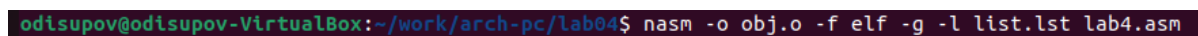
Рис. 3.3: Редактирование

3. Оттранслируйте полученный текст программы lab4.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.



```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
```

Рис. 3.4: Компиляция



```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst lab4.asm
```

Рис. 3.5: Выполнение команды



```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o hello
```

Рис. 3.6: Передача на обработку

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ ./hello
Isupov Oleg
```

Рис. 3.7: Запуск файла

4. Скопируйте файлы hello.asm и lab4.asm в Ваш локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/. Загрузите файлы на Github

```
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ cp hello.asm ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc/labs04
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$ cp lab4.asm ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc/labs04
odisupov@odisupov-VirtualBox:~/work/arch-pc/lab04$
```

Рис. 3.8: Копирование файлов

```
odisupov@odisupov-VirtualBox:~/work/study/2023-2024/Архитектура\ компьютера/arch-pc$ git add .
odisupov@odisupov-VirtualBox:~/work/study/2023-2024/Архитектура\ компьютера/arch-pc$ git commit -am 'feat(main): lab04'
[master a32a96d] feat(main): lab04
23 files changed, 153 insertions(+), 120 deletions(-)
delete mode 100644 labs/lab02/report/.~lock.102_Исупов_Отчёт.doc#
create mode 100644 labs/lab03/report/image/image.zip
create mode 100644 labs/lab04/report/image/1.png
create mode 100644 labs/lab04/report/image/10.png
create mode 100644 labs/lab04/report/image/11.png
create mode 100644 labs/lab04/report/image/12.png
create mode 100644 labs/lab04/report/image/13.png
create mode 100644 labs/lab04/report/image/14.png
create mode 100644 labs/lab04/report/image/15.png
create mode 100644 labs/lab04/report/image/16.png
create mode 100644 labs/lab04/report/image/17.png
create mode 100644 labs/lab04/report/image/18.png
create mode 100644 labs/lab04/report/image/2.png
create mode 100644 labs/lab04/report/image/3.png
create mode 100644 labs/lab04/report/image/4(2).png
create mode 100644 labs/lab04/report/image/4.png
create mode 100644 labs/lab04/report/image/5.png
create mode 100644 labs/lab04/report/image/6.png
create mode 100644 labs/lab04/report/image/7.png
create mode 100644 labs/lab04/report/image/8.png
create mode 100644 labs/lab04/report/image/9.png
rewrite labs/lab04/report/report.md (71%)
create mode 100644 labs04
odisupov@odisupov-VirtualBox:~/work/study/2023-2024/Архитектура\ компьютера/arch-pc$ git push
Перечисление объектов: 44, готово.
Подсчет объектов: 100% (44/44), готово.
При сжатии изменений используется до 7 потоков
Сжатие объектов: 100% (33/33), готово.
Запись объектов: 100% (33/33), 646.75 Киб | 5.34 Миб/с, готово.
Всего 33 (изменений 6), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (6/6), completed with 6 local objects.
To github.com:mxnzter/study_2023-2024_arhpc.git
 a0c483b..a32a96d master -> master
```

Рис. 3.9: Загрузка файлов на Github

4 Выводы

Таким образом мы освоили процедуры компиляции и сборки программ, написанных на ассемблере NASM.