

Отчёта по лабораторной работе №5

Создание и процесс обработки программ на языке ассемблера NASM

Акопян Сатеник Манвеловна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Задание для самостоятельной работы	10
5	Выводы	13
	Список литературы	14

Список иллюстраций

3.1	рисунок 1	7
3.2	рисунок 2	7
3.3	рисунок 3	7
3.4	рисунок 4	8
3.5	рисунок 5	8
3.6	рисунок 6	8
3.7	рисунок 7	8
3.8	рисунок 8	8
3.9	рисунок 9	9
4.1	рисунок 10	10
4.2	рисунок 11	10
4.3	рисунок 12	11
4.4	рисунок 13	11
4.5	рисунок 14	11
4.6	рисунок 15	11
4.7	рисунок 16	11
4.8	рисунок 17	12

Список таблиц

1 Цель работы

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

2 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.

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язык низкого уровня. Можно считать, что он больше любых других языков приближен к архитектуре ЭВМ и её аппаратным возможностям, что позволяет получить к ним более полный доступ, нежели в языках высокого уровня, таких как C/C++, Perl, Python и пр. Заметим, что получить полный доступ к ресурсам компьютера в современных архитектурах нельзя, самым низким уровнем работы прикладной программы является обращение напрямую к ядру операционной системы. Именно на этом уровне и работают программы, написанные на ассемблере. Но в отличие от языков высокого уровня ассемблерная программа содержит только тот код, который ввёл программист. Таким образом язык ассемблера — это язык, с помощью которого понятным для человека образом пишутся команды для процессора.

3 Выполнение лабораторной работы

1. Создаём текстовый файл с именем hello.asm (рис. 3.1)

```
[smakopyan@fedora lab05]$ touch hello.asm  
[smakopyan@fedora lab05]$
```

Рис. 3.1: рисунок 1

2. Открываем этот файл с помощью gedit (рис. 3.2)

```
[smakopyan@fedora lab05]$ touch hello.asm  
[smakopyan@fedora lab05]$ gedit hello.asm
```

Рис. 3.2: рисунок 2

3. Вводим в него текст, данный в лабораторной работе (рис. 3.3)



```
Открыть ▾ + *hello.asm  
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/  
1 ; hello.asm  
2 SECTION .data  
3     hello: DB 'Hello world',10  
4  
5     helloLen: EQU $-hello  
6  
7 SECTION .text  
8     GLOBAL _start  
9  
10 _start:  
11     mov eax,4  
12     mov ebx,1 ;  
13     mov ecx,hello |  
14     mov edx,helloLen  
15     int 80h  
16     mov eax,1  
17     mov ebx,0  
18     int 80h
```

Рис. 3.3: рисунок 3

4. Компилируем текст программы “Hello world” (рис. 3.4)

```
[smakopyan@fedora lab05]$ nasm -f elf hello.asm  
[smakopyan@fedora lab05]$
```

Рис. 3.4: рисунок 4

5. С помощью команды ls проверяем, что объектный файл был создан (рис. 3.5). Объектный файл имеет имя hello.o

```
[smakopyan@fedora lab05]$ ls  
hello.asm hello.o presentation report  
[smakopyan@fedora lab05]$
```

Рис. 3.5: рисунок 5

6. Скомпилируем исходный файл hello.asm в obj.o и с помощью команды ls проверяем, что файлы были созданы.(рис. 3.6)

```
[smakopyan@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst hello.asm  
[smakopyan@fedora lab05]$ ls  
hello.asm hello.o list.lst obj.o presentation report
```

Рис. 3.6: рисунок 6

7. Далее чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику, с помощью команды ls проверяем что файл hello был создан. (рис. 3.7)

```
[smakopyan@fedora lab05]$ ld -m elf_i386 hello.o -o hello  
[smakopyan@fedora lab05]$ ls  
hello hello.asm hello.o list.lst obj.o presentation report
```

Рис. 3.7: рисунок 7

8. Выполняем следующую команду, которая создаёт файл main (рис. 3.8)

```
[smakopyan@fedora lab05]$ ld -m elf_i386 obj.o -o main  
[smakopyan@fedora lab05]$ ls  
hello hello.asm hello.o list.lst main obj.o presentation report  
[smakopyan@fedora lab05]$
```

Рис. 3.8: рисунок 8

9. Запускаем на выполнение созданный исполняемый файл, находящийся в текущем каталоге (рис. 3.9)

A terminal window with a dark background. The prompt is [smakopyan@fedora lab05]\$. The command ./hello is entered. The output Hello world is displayed. The prompt [smakopyan@fedora lab05]\$ is shown again with a white cursor.

```
[smakopyan@fedora lab05]$ ./hello
Hello world
[smakopyan@fedora lab05]$
```

Рис. 3.9: рисунок 9

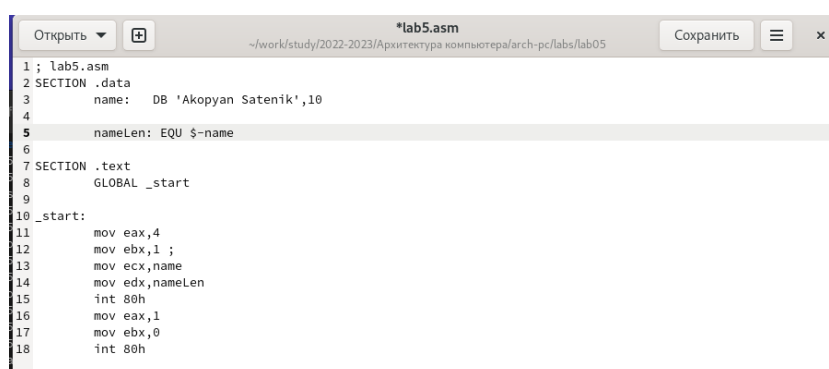
4 Задание для самостоятельной работы

1. В каталоге ~/work/arch-pc/lab05 с помощью команды `cp` создаем копию файла `hello.asm` с именем `lab5.asm` (рис. 4.1)

```
[smakopyan@fedora lab05]$ cp ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab05/hello.asm ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab05/lab5.asm
[smakopyan@fedora lab05]$ ls
hello      hello.o    list.lst  obj.o      report
hello.asm  lab5.asm  main      presentation
```

Рис. 4.1: рисунок 10

2. С помощью `gedit` вносим изменения в текст программы в файле `lab5.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с моей фамилией и именем. (рис. 4.2)



```
*lab5.asm
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05
Сохранить x

1 ; lab5.asm
2 SECTION .data
3     name:    DB 'Akopyan Satenik',10
4
5     nameLen: EQU $-name
6
7 SECTION .text
8     GLOBAL _start
9
10 _start:
11     mov eax,4
12     mov ebx,1 ;
13     mov ecx,name
14     mov edx,nameLen
15     int 80h
16     mov eax,1
17     mov ebx,0
18     int 80h
```

Рис. 4.2: рисунок 11

3. Оттранслируем полученный текст программы `lab5.asm` в объектный файл. (рис. 4.3) (рис. 4.4)

```
[smakopyan@fedora lab05]$ nasm -f elf lab5.asm  
[smakopyan@fedora lab05]$
```

Рис. 4.3: рисунок 12

```
[smakopyan@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst lab5.asm  
[smakopyan@fedora lab05]$
```

Рис. 4.4: рисунок 13

4. Выполняем компоновку объектного файла и запустите получившийся исполняемый файл. (рис. 4.5) (рис. 4.6) (рис. 4.7)

```
[smakopyan@fedora lab05]$ ld -m elf_i386 lab5.o -o lab5  
[smakopyan@fedora lab05]$
```

Рис. 4.5: рисунок 14

```
[smakopyan@fedora lab05]$ ld -m elf_i386 obj.o -o main  
[smakopyan@fedora lab05]$
```

Рис. 4.6: рисунок 15

```
[smakopyan@fedora lab05]$ ./lab5  
Akopyan Satenik  
[smakopyan@fedora lab05]$
```

Рис. 4.7: рисунок 16

5. Копируем файлы hello.asm и lab5.asm в локальный репозиторий в каталог ~/work/study 2022-2023/“Архитектура компьютера”/arch-pc/labs/lab05/. Загружаем файлы на github (рис. 4.8)

```
smakopyan@fedora:~/work/study/2022-2023/Архитектура ...
Akopyan Satenik
[smakopyan@fedora lab05]$ git add .
[smakopyan@fedora lab05]$ git commit -am 'lab5'
[master 2d242d6] lab5
10 files changed, 55 insertions(+)
create mode 100755 labs/lab05/hello
create mode 100644 labs/lab05/hello.asm
create mode 100644 labs/lab05/hello.o
create mode 100755 labs/lab05/lab5
create mode 100644 labs/lab05/lab5.asm
create mode 100644 labs/lab05/lab5.o
create mode 100644 labs/lab05/list.lst
create mode 100755 labs/lab05/main
create mode 100644 labs/lab05/obj.o
delete mode 100644 labs/lab05/report/report.docx
[smakopyan@fedora lab05]$ git push
Перечисление объектов: 100% (18/18), готово.
Подсчет объектов: 100% (18/18), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (14/14), готово.
Запись объектов: 100% (14/14), 2.98 КиБ | 1.49 МиБ/с, готово.
Всего 14 (изменений 6), повторно использовано 0 (изменений 0), повторно использо-
вано пакетов 0
remote: Resolving deltas: 100% (6/6), completed with 3 local objects.
```

Рис. 4.8: рисунок 17

5 Выводы

В результате данной лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.

Список литературы