

Отчёт по лабораторной работе №5

Создание и процесс обработки программ на языке ассемблера NASM

Горайнова Алёна Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Выводы	11
	Список литературы	12

Список иллюстраций

4.1	Создание простой программы, выводящей Hello world!	9
4.2	Текст измененной программы	10
4.3	Запуск программы	10

Список таблиц

1 Цель работы

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

2 Задание

Рассмотреть пример простой программы на языке ассемблера NASM.

3 Теоретическое введение

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.

Следует отметить, что процессор понимает не команды ассемблера, а последовательности из нулей и единиц — машинные коды. До появления языков ассемблера программистам приходилось писать программы, используя только лишь машинные коды, которые были крайне сложны для запоминания, так как представляли собой числа, записанные в двоичной или шестнадцатеричной системе счисления. Преобразование или трансляция команд с языка ассемблера в исполняемый машинный код осуществляется специальной программой транслятором — Ассемблер.

Программы, написанные на языке ассемблера, не уступают в качестве и скорости программам, написанным на машинном языке, так как транслятор просто

переводит мнемонические обозначения команд в последовательности бит (нулей и единиц).

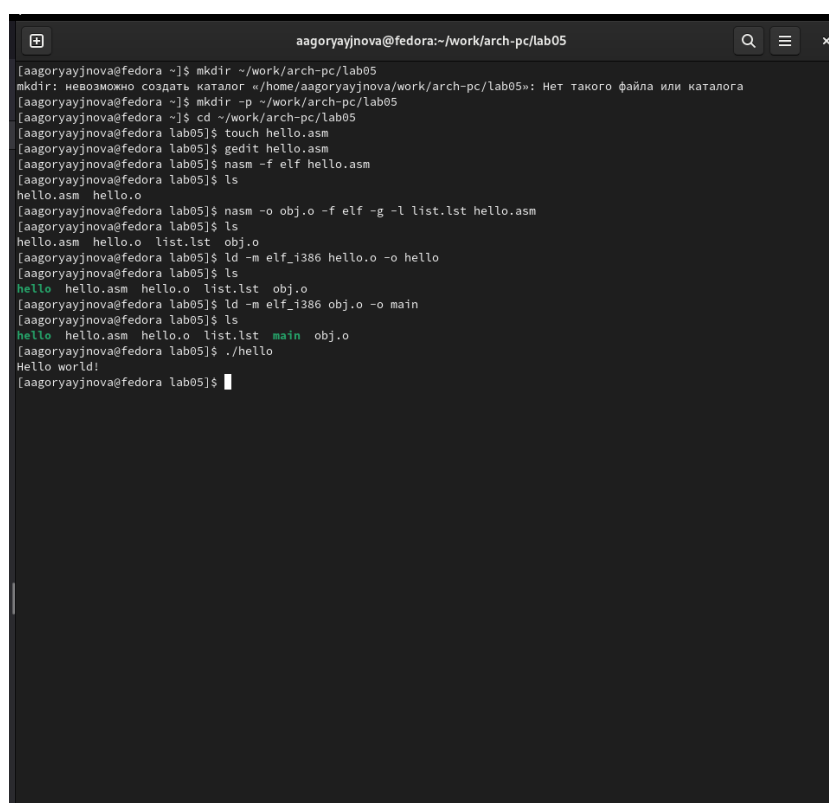
Используемые мнемоники обычно одинаковы для всех процессоров одной архитектуры или семейства архитектур (среди широко известных — мнемоники процессоров и контроллеров x86, ARM, SPARC, PowerPC, M68k). Таким образом для каждой архитектуры существует свой ассемблер и, соответственно, свой язык ассемблера.

Наиболее распространёнными ассемблерами для архитектуры x86 являются:

- для DOS/Windows: Borland Turbo Assembler (TASM), Microsoft Macro Assembler (MASM) и Watcom assembler (WASM);
- для GNU/Linux: gas (GNU Assembler), использующий AT&T-синтаксис, в отличие от большинства других популярных ассемблеров, которые используют Intel-синтаксис.

4 Выполнение лабораторной работы

Создала каталог для работы с программами на языке ассемблера NASM, перешла в него, создала и открыла текстовый файл `hello.asm`. Ввела текст из лабораторной работы. Далее я скомпилировала исходный файл `hello.asm` в `obj.o`, передала на обработку компоновщику и проверила всё это с помощью команды `ls`. И наконец запустила на выполнение созданный файл. (рис. 4.1)



```
aagoryaynova@fedora:~/work/arch-pc/lab05
[aagoryaynova@fedora ~]$ mkdir ~/work/arch-pc/lab05
mkdir: невозможно создать каталог «/home/aagoryaynova/work/arch-pc/lab05»: Нет такого файла или каталога
[aagoryaynova@fedora ~]$ mkdir -p ~/work/arch-pc/lab05
[aagoryaynova@fedora ~]$ cd ~/work/arch-pc/lab05
[aagoryaynova@fedora lab05]$ touch hello.asm
[aagoryaynova@fedora lab05]$ gedit hello.asm
[aagoryaynova@fedora lab05]$ nasm -f elf hello.asm
[aagoryaynova@fedora lab05]$ ls
hello.asm  hello.o
[aagoryaynova@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst hello.asm
[aagoryaynova@fedora lab05]$ ls
hello.asm  hello.o  list.lst  obj.o
[aagoryaynova@fedora lab05]$ ld -m elf_i386 hello.o -o hello
[aagoryaynova@fedora lab05]$ ls
hello  hello.asm  hello.o  list.lst  obj.o
[aagoryaynova@fedora lab05]$ ld -m elf_i386 obj.o -o main
[aagoryaynova@fedora lab05]$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
[aagoryaynova@fedora lab05]$ ./hello
Hello world!
[aagoryaynova@fedora lab05]$
```

Рис. 4.1: Создание простой программы, выводящей Hello world!

Приступила к выполнению заданий для самостоятельной работы, создала копию

файла hello.asm с именем lab5.asm, поменяла текст программы. Оттранслировала полученный текст программы lab5.asm в объектный файл, выполнила компоновку объектного файла и запустила получившийся исполняемый файл. (рис. 4.2, 4.3)

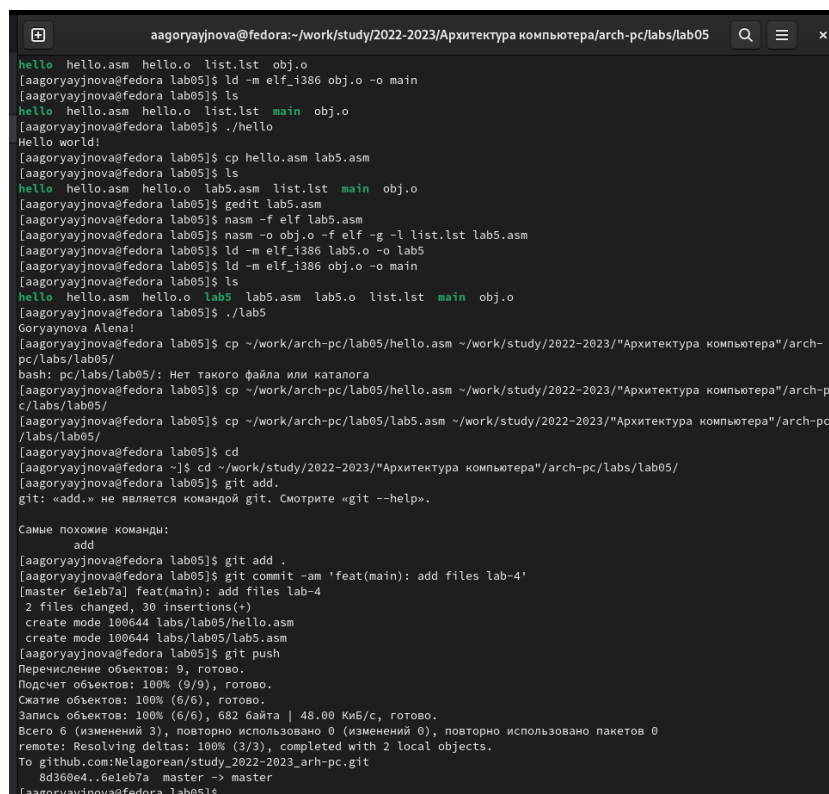


```

1 SECTION .data
2 hello: DB 'Goryaynova Alena!',10
3 helloLen: EQU $-hello
4 SECTION .text
5 GLOBAL _start
6 _start:
7     mov eax,4
8     mov ebx,1
9     mov ecx,hello
10    mov edx,helloLen
11    int 80h
12
13    mov eax,1
14    mov ebx,0
15    int 80h

```

Рис. 4.2: Текст измененной программы



```

aagoryayjnova@fedora:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05
hello hello.asm hello.o list.lst obj.o
[aagoryayjnova@fedora lab05]$ ld -m elf_i386 obj.o -o main
[aagoryayjnova@fedora lab05]$ ls
hello hello.asm hello.o list.lst main obj.o
[aagoryayjnova@fedora lab05]$ ./hello
Hello world!
[aagoryayjnova@fedora lab05]$ cp hello.asm lab5.asm
[aagoryayjnova@fedora lab05]$ ls
hello hello.asm hello.o lab5.asm list.lst main obj.o
[aagoryayjnova@fedora lab05]$ gedit lab5.asm
[aagoryayjnova@fedora lab05]$ nasm -f elf lab5.asm
[aagoryayjnova@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst lab5.asm
[aagoryayjnova@fedora lab05]$ ld -m elf_i386 lab5.o -o lab5
[aagoryayjnova@fedora lab05]$ ld -m elf_i386 obj.o -o main
[aagoryayjnova@fedora lab05]$ ls
hello hello.asm hello.o lab5.asm lab5.o list.lst main obj.o
[aagoryayjnova@fedora lab05]$ ./lab5
Goryaynova Alena!
[aagoryayjnova@fedora lab05]$ cp ~/work/arch-pc/lab05/hello.asm ~/work/study/2022-2023/"Архитектура компьютера"/arch-
pc/labs/lab05/
bash: pc/labs/lab05/: Нет такого файла или каталога
[aagoryayjnova@fedora lab05]$ cp ~/work/arch-pc/lab05/hello.asm ~/work/study/2022-2023/"Архитектура компьютера"/arch-p
c/labs/lab05/
[aagoryayjnova@fedora lab05]$ cp ~/work/arch-pc/lab05/lab5.asm ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc
/labs/lab05/
[aagoryayjnova@fedora lab05]$ cd
[aagoryayjnova@fedora lab05]$ cd ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab05/
[aagoryayjnova@fedora lab05]$ git add.
git: «add.» не является командой git. Смотрите «git --help».

Самые похожие команды:
add
[aagoryayjnova@fedora lab05]$ git add .
[aagoryayjnova@fedora lab05]$ git commit -am 'feat(main): add files lab-4'
[master 6e1eb7a] feat(main): add files lab-4
2 files changed, 30 insertions(+)
create mode 100644 labs/lab05/hello.asm
create mode 100644 labs/lab05/lab5.asm
[aagoryayjnova@fedora lab05]$ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 682 байта | 48.00 Киб/с, готово.
Всего 6 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:Nelagorean/study_2022-2023_arh-pc.git
8d360e4..6e1eb7a master -> master
[aagoryayjnova@fedora lab05]$

```

Рис. 4.3: Запуск программы

5 Выводы

Я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.

Список литературы