

Отчет по лабораторной работе № 4

Архитектура компьютера

Исаханян Армен Артурович

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	8
4	Выполнение самостоятельной работы	9
5	Выводы	11
	Список литературы	12

Список иллюстраций

3.1	Создание каталога и переход в него	8
3.2	Создание и открытие файла	8
3.3	Компиляция исходного файла и текста, передача файла компоновщику, задание имени файла	8
4.1	Создание копии, открытие редактора	9
4.2	Копирование файлов	9
4.3	Загрузка на GitHub	10

Список таблиц

1 Цель работы

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

2 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).

3 Выполнение лабораторной работы

Создаём каталог для работы с программами на языке ассемблера NASM, переходим в созданный каталог (рис. [3.1]).

```
aaisakhanyan@dk3n40 ~ $ mkdir -p ~/work/arch-pc/lab04
aaisakhanyan@dk3n40 ~ $ cd ~/work/arch-pc/lab04
```

Рис. 3.1: Создание каталога и переход в него

Создаём файл с именем hello.asm с помощью команды touch, затем открываем его с помощью команды gedit (рис. 3.2)

```
aaisakhanyan@dk3n40 ~/work/arch-pc/lab04 $ touch hello.asm
aaisakhanyan@dk3n40 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 3.2: Создание и открытие файла

Для компиляции текста программы “Hello world” написал: `nasm -f elf hello.asm`, скомпилировал исходный файл hello.asm в obj.o с помощью команды `nasm -o obj.o -f elf -g -l list.lst hello.asm`, передал объектный файл на обработку компоновщику с помощью команды `ld -m elf_i386 hello.o -o hello`, `ld -m elf_i386 obj.o -o main` - задал имя создаваемого исполняемого файла (рис. 3.3)

```
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

Рис. 3.3: Компиляция исходного файла и текста, передача файла компоновщику, задание имени файла

4 Выполнение самостоятельной работы

В каталоге `~/work/arch-pc/lab04` создал копию файла `hello.asm` с именем `lab04.asm`, открыл редактор, чтобы внести изменения. (рис. 4.1)

```
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ cp hello.asm lab04.asm
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ gedit lab04.asm
```

Рис. 4.1: Создание копии, открытие редактора

Оттранслировал полученный текст программы `lab04.asm` в объектный файл. Выполнил компоновку объектного файла и запустил получившийся файл. (рис. ??).

![Компиляция исходного файла и текста, передача файла компоновщику, задание имени файла(image/image5.jpg)]{#fig:005 width=70%}

Скопировал оба файла в свой локальный репозиторий в каталог `~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/lab04/`(рис. 4.2)

```
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ mv hello.asm ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/
aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ mv lab04.asm ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/
```

Рис. 4.2: Копирование файлов

Загрузил файлы на GitHub (рис. 4.3)

```

aaisakhanyan@dk4n62 ~/work/arch-pc/lab04 $ cd ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/
aaisakhanyan@dk4n62 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git add .
aaisakhanyan@dk4n62 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git commit -am 'feat(main): add
[master 7b038a9] feat(main): add files lab04
2 files changed, 36 insertions(+)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab04.asm
aaisakhanyan@dk4n62 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04 $ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 781 байт | 781.00 КиБ/с, готово.
Всего 6 (изменений 2), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:Armen20053443/study_2023-2024_arh--pc.git
5eebcf6..7b038a9 master -> master

```

Рис. 4.3: Загрузка на GitHub

5 Выводы

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

Список литературы