

Отчёт по лабораторной работе № 5

Создание и процесс обработки программ на языке ассемблера NASM

Давит Оганнисян Багратович

Содержание

1	Цель работы	5
2	Задание	6
2.1	Программа Hello world!	6
2.2	Транслятор NASM	6
2.3	Расширенный синтаксис командной строки NASM	6
2.4	Компоновщик LD	6
2.5	Запуск исполняемого файла	6
2.6	Задание для самостоятельной работы	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Создаем каталог для работы с программами на языке ассемблера NASM, переходим в него и создаем текстовый файл с именем hello.asm. Открываем этот файл с помощью любого текстового редактора, например, gedit и вводим нужный нам текст	8
4.2	Превращение текста программы в объектный код и компилирование hello.asm в obj.o	9
4.3	Передача объектного файла на обработку компоновщику	10
4.4	Создаем также исполняемый файл main	10
4.5	Запуск исполняемого файла	10
4.6	Задание для самостоятельной работы	10
5	Выводы	12

Список иллюстраций

4.1	Создание hello.asm	8
4.2	Текст	9
4.3	Превращение текста программы в объектный код и компилирование hello.asm в obj.o	9
4.4	Передача объектного файла на обработку компоновщику	10
4.5	Создание также исполняемый файл main	10
4.6	Hello world!	10
4.7	Фамилия Имя	10
4.8	Копирование	10
4.9	Загрузка на гитхаб	11

Список таблиц

1 Цель работы

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

2 Задание

2.1 Программа Hello world!

2.2 Транслятор NASM

2.3 Расширенный синтаксис командной строки NASM

2.4 Компоновщик LD

2.5 Запуск исполняемого файла

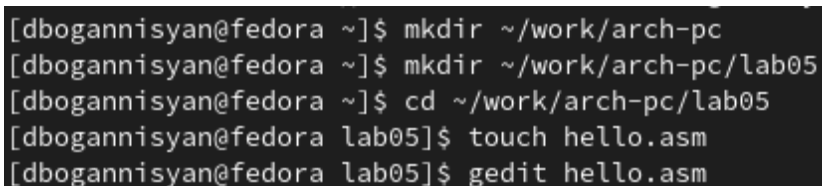
2.6 Задание для самостоятельной работы

3 Теоретическое введение

NASM — это открытый проект ассемблера, версии которого доступны под различные операционные системы и который позволяет получать объектные файлы для этих систем. В NASM используется Intel-синтаксис и поддерживаются инструкции x86-64. Типичный формат записи команд NASM имеет вид: [метка:] мнемокод [операнд {, операнд}] [; комментарий] Здесь мнемокод — непосредственно мнемоника инструкции процессору, которая является обязательной частью команды. Операндами могут быть числа, данные, адреса регистров или адреса оперативной памяти. Метка — это идентификатор, с которым ассемблер ассоциирует некоторое число, чаще всего адрес в памяти. Т.о. метка перед командой связана с адресом данной команды. Допустимыми символами в метках являются буквы, цифры, а также следующие символы: `,`, `$`, `#`, `@`, `~`, `.` и `?`. *Начинаться метка или идентификатор могут с буквы, `.`, и `?`.* Перед идентификаторами, которые пишутся как зарезервированные слова, нужно писать `$`, чтобы компилятор трактовал его верно (так называемое экранирование). Максимальная длина идентификатора 4095 символов. Программа на языке ассемблера также может содержать директивы — инструкции, не переводящиеся непосредственно в машинные команды, а управляющие работой транслятора. Например, директивы используются для определения данных (констант и переменных) и обычно пишутся большими буквами.

4 Выполнение лабораторной работы

4.1 Создаем каталог для работы с программами на языке ассемблера NASM, переходим в него и создаем текстовый файл с именем hello.asm. Открываем этот файл с помощью любого текстового редактора, например, gedit и вводим нужный нам текст

A terminal window with a dark background and light-colored text. It shows a series of five commands being executed in a shell. The prompt is [dbogannisyan@fedora ~]. The commands are: mkdir ~/work/arch-pc, mkdir ~/work/arch-pc/lab05, cd ~/work/arch-pc/lab05, touch hello.asm, and gedit hello.asm. The last command is still being executed, as indicated by the dollar sign at the end of the line.

```
[dbogannisyan@fedora ~]$ mkdir ~/work/arch-pc
[dbogannisyan@fedora ~]$ mkdir ~/work/arch-pc/lab05
[dbogannisyan@fedora ~]$ cd ~/work/arch-pc/lab05
[dbogannisyan@fedora lab05]$ touch hello.asm
[dbogannisyan@fedora lab05]$ gedit hello.asm
```

Рис. 4.1: Создание hello.asm

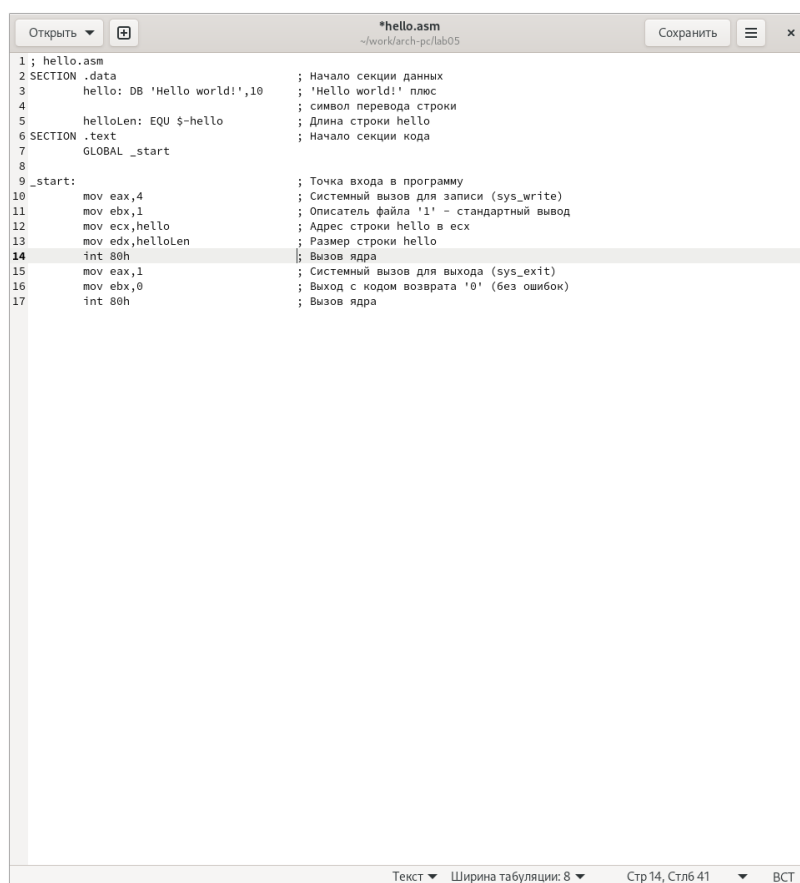


Рис. 4.2: Текст

4.2 Превращение текста программы в объектный код и компилирование hello.asm в obj.o

```

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

```

Рис. 4.3: Превращение текста программы в объектный код и компилирование hello.asm в obj.o

4.3 Передача объектного файла на обработку

компоновщику

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

Рис. 4.4: Передача объектного файла на обработку компоновщику

4.4 Создаем также исполняемый файл main

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

Рис. 4.5: Создание также исполняемый файл main

4.5 Запуск исполняемого файла

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

Рис. 4.6: Hello world!

4.6 Задание для самостоятельной работы

```
[dbogannisyan@fedora lab05]$ cp ~/work/arch-pc/lab05/hello.asm ~/work/arch-pc/lab05/lab5.asm
[dbogannisyan@fedora lab05]$ gedit lab5.asm
[dbogannisyan@fedora lab05]$ nasm -f elf lab5.asm
[dbogannisyan@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst lab5.asm
[dbogannisyan@fedora lab05]$ ld -m elf_i386 lab5.o -o lab5
[dbogannisyan@fedora lab05]$ ./lab5
Давит Оганнисян
```

Рис. 4.7: Фамилия Имя

```
[dbogannisyan@fedora lab05]$ cp ~/work/arch-pc/lab05/hello.asm ~/work/study/2022-2023/"Архитектура компьютера"/study_2022-2023_arh-pc/labs/lab05/hello.asm
[dbogannisyan@fedora lab05]$ cp ~/work/arch-pc/lab05/lab5.asm ~/work/study/2022-2023/"Архитектура компьютера"/study_2022-2023_arh-pc/labs/lab05/lab5.asm
```

Рис. 4.8: Копирование

```

[dbogannisyan@fedora study_2022-2023_arh-pc]$ git add .
[dbogannisyan@fedora study_2022-2023_arh-pc]$ git commit -am 'lab05'
[master 3ed46c7] lab05
4 files changed, 76 insertions(+), 37 deletions(-)
delete mode 100644 labs/lab03/report/./lock.report.docx#
create mode 100644 labs/lab05/hello.asm
create mode 100644 labs/lab05/lab5.asm
[dbogannisyan@fedora study_2022-2023_arh-pc]$ git push
Перечисление объектов: 100% (19/19), готово.
Подсчет объектов: 100% (19/19), готово.
При сжатии изменений используется до 8 потоков
Сжатие объектов: 100% (11/11), готово.
Запись объектов: 100% (11/11), 2.93 КиБ | 1.47 МиБ/с, готово.
Всего 11 (изменений 5), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (5/5), completed with 4 local objects.
To github.com:dbogannisyanNKA/study_2022-2023_arh-pc.git
d70681e..3ed46c7 master -> master
[dbogannisyan@fedora study_2022-2023_arh-pc]$

```

Рис. 4.9: Загрузка на гитхаб

5 Выводы

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