Отчет по лабораторной работе 4

Создание и процесс обработки программ на языке ассемблера NASM

Грасимилде М.В.Нанке

Содержание

# Цель работы

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

# Задание

1. Программа Hello world!
2. Транслятор NASM
3. Расширенный синтаксис командной строки NASM
4. Компоновщик LD
5. Запуск исполняемого файла
6. Задание для самостоятельной работы

# Теоретическое введение

Процесс создания ассемблерной программы можно изобразить в виде следующей схемы.



Процесс создания ассемблерной программы

В процессе создания ассемблерной программы можно выделить четыре шага:

1. Набор текста программы в текстовом редакторе и сохранение её в отдельном файле. Каждый файл имеет свой тип (или расширение), который определяет назначение файла. Файлы с исходным текстом программ на языке ассемблера имеют тип asm.
2. Трансляция — преобразование с помощью транслятора, например nasm, текста про- граммы в машинный код, называемый объектным. На данном этапе также может быть получен листинг программы, содержащий кроме текста программы различную допол- нительную информацию, созданную транслятором. Тип объектного файла — o, файла листинга — lst.
3. Компоновка или линковка — этап обработки объектного кода компоновщиком (ld), который принимает на вход объектные файлы и собирает по ним исполняемый файл. Исполняемый файл обычно не имеет расширения. Кроме того, можно получить файл карты загрузки программы в ОЗУ, имеющий расширение map.
4. Запуск программы. Конечной целью является работоспособный исполняемый файл. Ошибки на предыдущих этапах могут привести к некорректной работе программы, поэтому может присутствовать этап отладки программы при помощи специальной программы — отладчика. При нахождении ошибки необходимо провести коррекцию программы, начиная с первого шага.

Из-за специфики программирования, а также по традиции для создания программ на язы- ке ассемблера обычно пользуются утилитами командной строки (хотя поддержка ассемблера есть в некоторых универсальных интегрированных средах).

# Выполнение лабораторной работы

## Программа Hello world!

Для начала, использую команды для установки nasm через терминал:(рис. [-@fig:001]).

sudo dnf install -y nasm



Установка nasm через терминал

Рассмотрим пример простой программы на языке ассемблера NASM. Традиционно первая программа выводит приветственное сообщение Hello world! на экран.

Создаю каталог для работы с программами на языке ассемблера NASM:(рис. [-@fig:002]).

mkdir -p ~/work/arch-pc/lab04

Создание каталога для работы на языке NASM

Создание каталога для работы на языке NASM

Перехожу в созданный каталог (рис. [-@fig:003]).

cd ~/work/arch-pc/lab04

Перехожу в созданный каталог

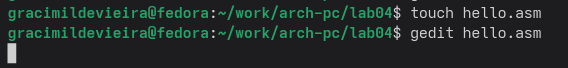
Перехожу в созданный каталог

Создаю текстовый файл с именем hello.asm (рис. [-@fig:004]).

touch hello.asm

и открываю этот файл с помощью текстового редактора gedit (рис. [-@fig:004]).

gedit hello.asm



Создание файла и редактирование в gedit

и ввожу в него следующий текст:

; hello.asm

SECTION .data ; Начало секции данных

hello: DB ‘Hello world!’,10 ; ‘Hello world!’ плюс

; символ перевода строки

helloLen: EQU $-hello ; Длина строки hello

SECTION .text ; Начало секции кода

GLOBAL \_start

\_start: ; Точка входа в программу

mov eax,4 ; Системный вызов для записи (sys\_write)

mov ebx,1 ; Описатель файла ‘1’ - стандартный вывод

mov ecx,hello ; Адрес строки hello в ecx

mov edx,helloLen ; Размер строки hello

int 80h ; Вызов ядра

mov eax,1 ; Системный вызов для выхода (sys\_exit)

mov ebx,0 ; Выход с кодом возврата ‘0’ (без ошибок)

int 80h ; Вызов ядра

## Транслятор NASM

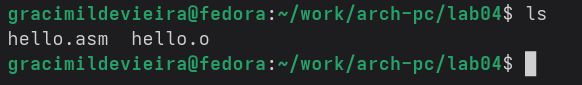
NASM превращает текст программы в объектный код. Например, для компиляции приведённого выше текста программы «Hello World» прописываю: (рис. [-@fig:005]).

nasm -f elf hello.asm

Команда для компиляции

Команда для компиляции

Транслятор преобразовывает текст программы из файла hello.asm в объектный код, который записывается в файл hello.o. С помощью команды ls проверяю, что объектный файл был создан. Созданный объектный файл имеет имя hello.o (рис. [-@fig:006]).



Проверка созданного файла

## Расширенный синтаксис командной строки NASM

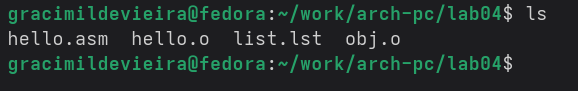
Выполняю следующую команду: (рис. [-@fig:007]).

nasm -o obj.o -f elf -g -l list.lst hello.asm

Выполнение команды

Выполнение команды

С помощью команды ls проверяю, что файлы были созданы.



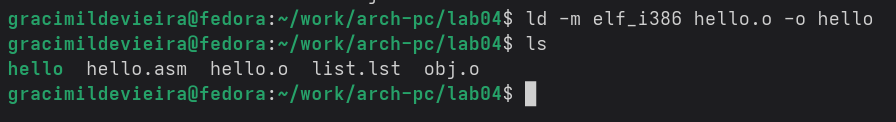
Проверка созданных файлов

## Компоновщик LD

Чтобы получить исполняемую программу, объектный файл передаю на обработку компоновщику:

ld -m elf\_i386 hello.o -o hello

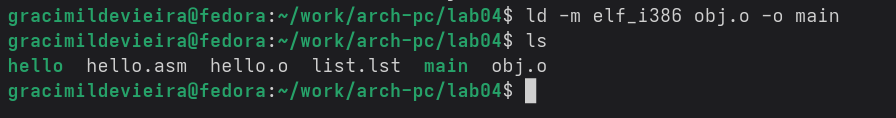
и с помощью команды ls проверяю, что исполняемый файл hello был создан.



Передача объектного файла компоновщику и проверка

Выполняю следующую команду:

ld -m elf\_i386 obj.o -o main



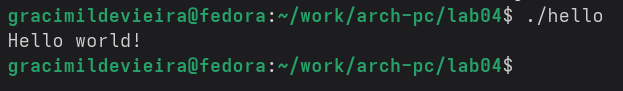
Исполнение команды

Исполняемый файл имеет имя hello. Объектный файл из которого собран этот исполняемый файл имеет имя hello.o

## Запуск исполняемого файла

Запускаю созданный исполняемый файл, находящийся в текущем каталоге, набрав в командной строке:

./hello



Запуск созданного файла

## Задание для самостоятельной работы

В каталоге ~/work/arch-pc/lab04 с помощью команды cp создаю копию файла (рис. [-@fig:012]).

hello.asm с именем lab4.asm

Создаю копию файла с новым именем

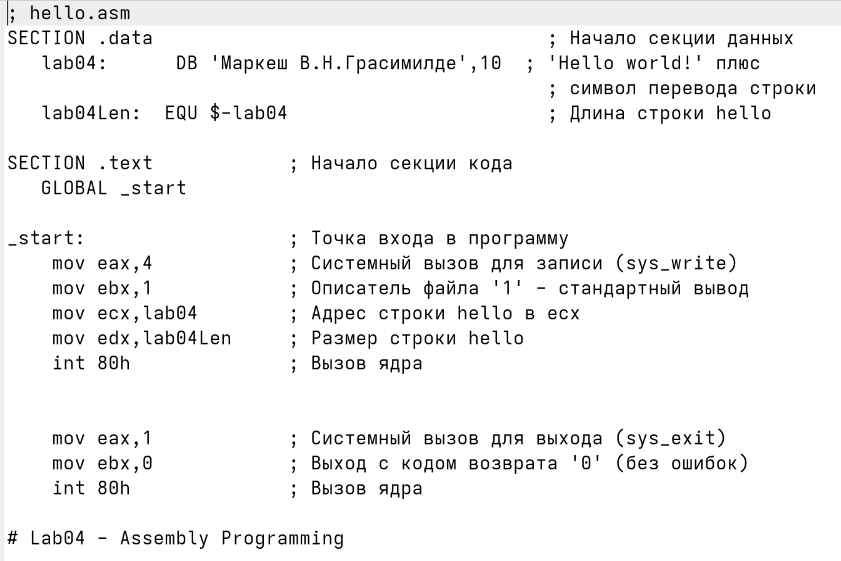
Создаю копию файла с новым именем

Открываю текстовый редактор gedit (рис. [-@fig:013]).

Открываю gedit

Открываю gedit

Вношу изменения в текст программы в файле lab4.asm так, чтобы вместо Hello world! на экран выводилась строка с моим фамилией и именем. (рис. [-@fig:014]).



Вношу свои имя и фамилию

Компилирую текст программы в объектный файл (рис. [-@fig:015]).

Компиляция объектного файла

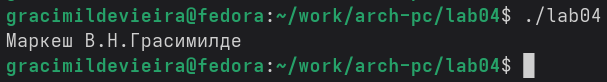
Компиляция объектного файла

Передаю объектный файл lab4.o на обработку компоновщику LD, чтобы получить исполняемый файл lab4 (рис. [-@fig:016]).

Передача объектного файла компоновщику

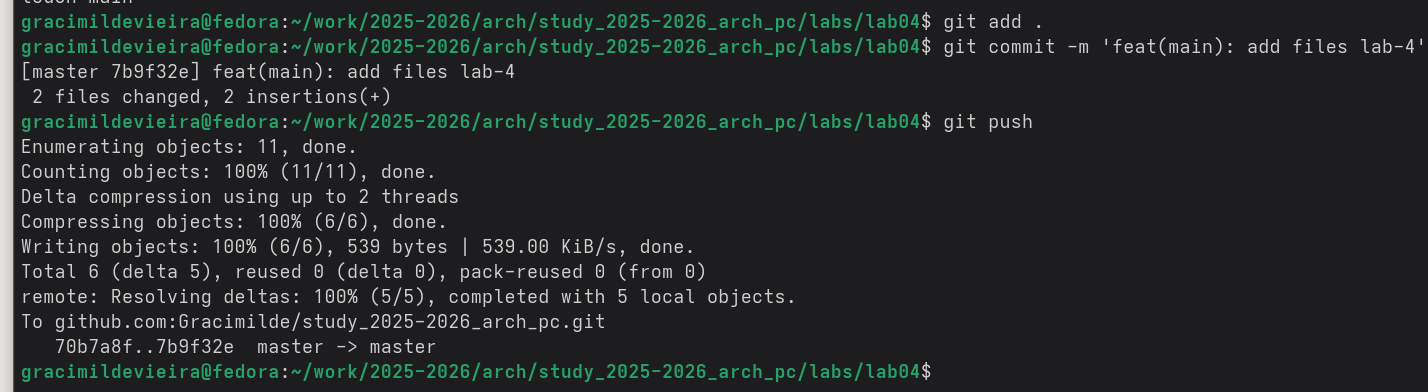
Передача объектного файла компоновщику

Запускаю исполняемый файл, на экран действительно выводятся мои имя и фамилия (рис. [-@fig:017]).



Запуск программы

С помощью команд git add . и git commit добавляю файлы на гитхаб и отправляю файлы на сервер с помощью команды git push (рис. [-@fig:018]).



Отправка на гитхаб

# Выводы

При выполнении лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.