Архитектура ЭВМ

**Лабораторная работа №5.Создание и процесс обработки программ на языке ассемблера NASM**

# Цель работы

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

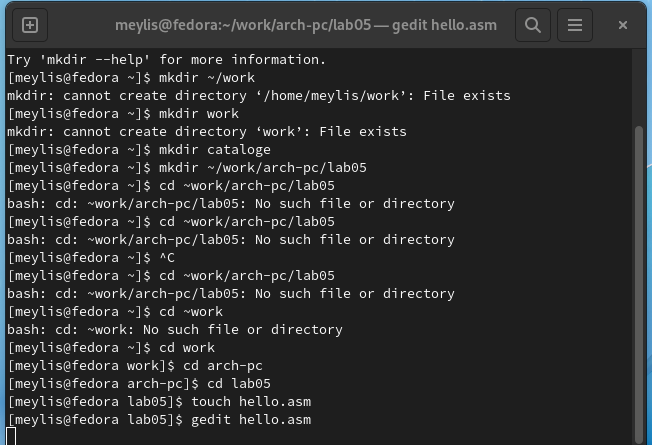
# 5.3. Порядок выполнения лабораторной работы

**5.3.1. Программа Hello world!**

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

на экран.

Создайте каталог для работы с программами на языке ассемблера NASM: mkdir ~/work/arch-pc/lab05



Перейдите в созданный каталог cd ~/work/arch-pc/lab05

Создайте текстовый файл с именем hello.asm touch hello.asm откройте этот файл с помощью любого текстового редактора, например, gedit gedit hello.asm Демидова А. В.

Архитектура ЭВМ

и введите 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

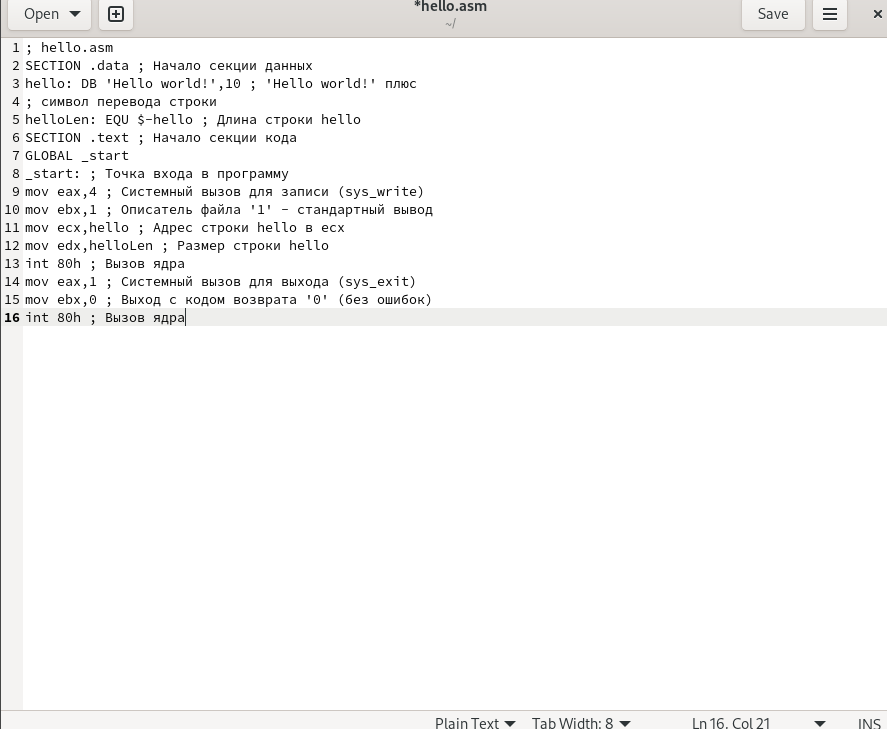
**SECTION** .text ; Начало секции кода

**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

mov eax,1 ; Системный вызов для выхода (sys\_exit) mov ebx,0 ; Выход с кодом возврата '0' (без ошибок) int 80h ; Вызов ядра



В отличие от многих современных высокоуровневых языков программирования, в ассемблерной программе каждая команда располагается на *отдельной строке*. Размещение нескольких команд на одной строке **недопустимо**. Синтаксис ассемблера NASM является *чувствительнымкрегистру*,т.е. есть разница между большими и малыми буквами.

## 5.3.2. Транслятор NASM

NASMпревращаеттекстпрограммывобъектныйкод.Например,длякомпиляции приведённого выше текста программы «Hello World» необходимо написать:

Демидова А. В.

nasm -f elf hello.asm

Если текст программы набран без ошибок,то транслятор преобразует текст программы из файла hello.asm в объектный код, который запишется в файл hello.o. Таким образом, имена всех файлов получаются из имени входного файла и расширения по умолчанию. При наличии ошибок объектный файл не создаётся, а после запуска транслятора появятся сообщения об ошибках или предупреждения.

С помощью команды ls проверьте, что объектный файл был создан. Какое имя имеет объектный файл?

NASM не запускают без параметров. Ключ -f указываеттранслятору, что требуется создать бинарные файлы в формате ELF. Следует отметить, что формат elf64 позволяет создавать исполняемый код,работающий под 64-битными версиями Linux. Для 32-битных версий ОС указываем в качестве формата просто elf.

NASM всегда создаёт выходные файлы в **текущем** каталоге.

## 5.3.3. Расширенный синтаксис командной строки NASM

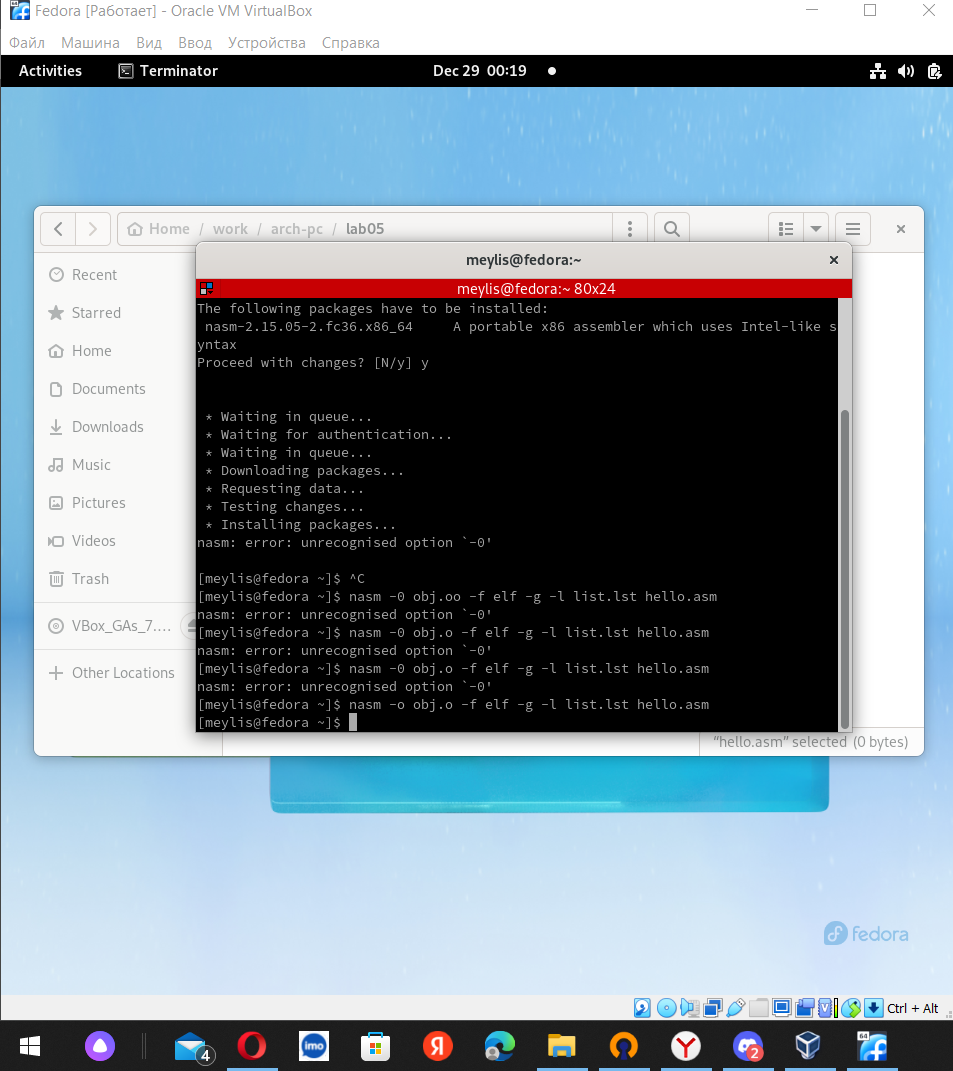
Полный вариант командной строки nasm выглядит следующим образом:

nasm [-@ косвенный\_файл\_настроек] [-o объектный\_файл] [-f

↪ формат\_объектного\_файла] [-l листинг] [параметры...] [--]

↪ исходный\_файл

Выполните следующую команду: nasm -o obj.o -f elf -g -l list.lst hello.asm



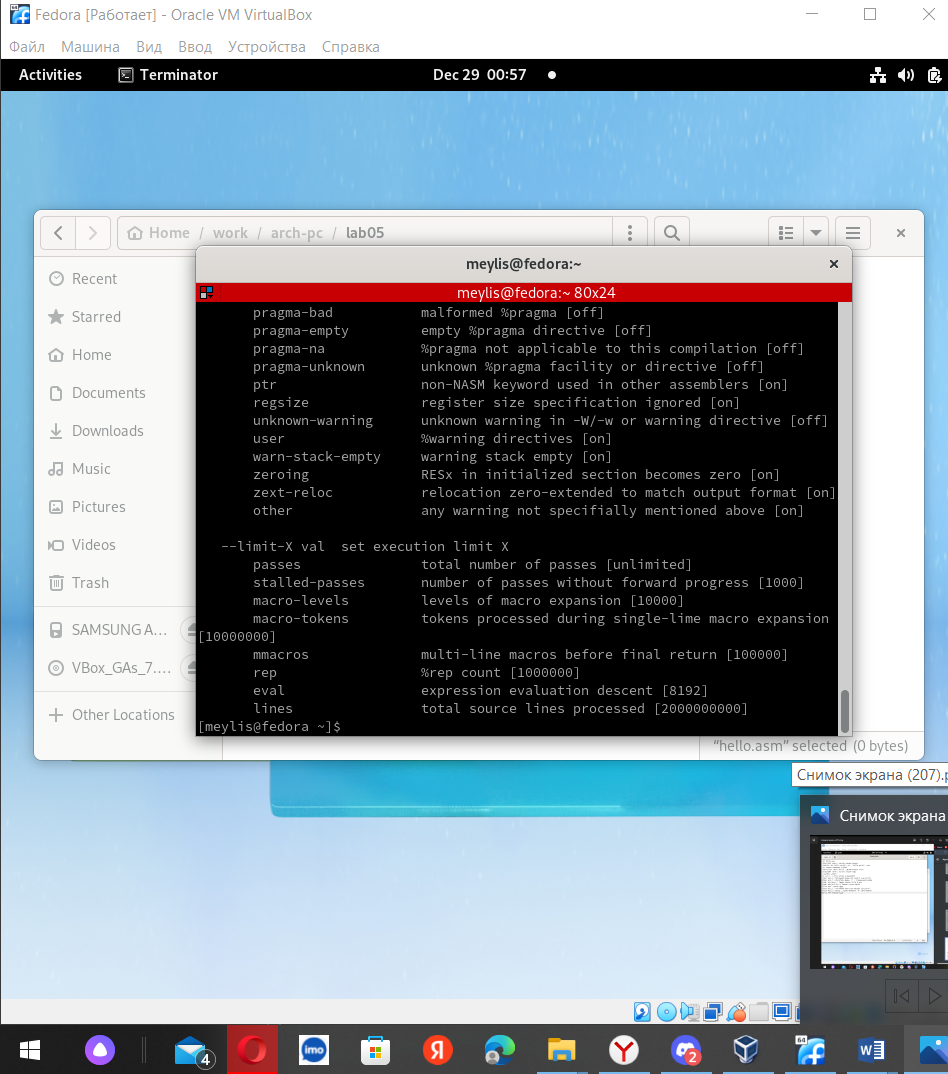
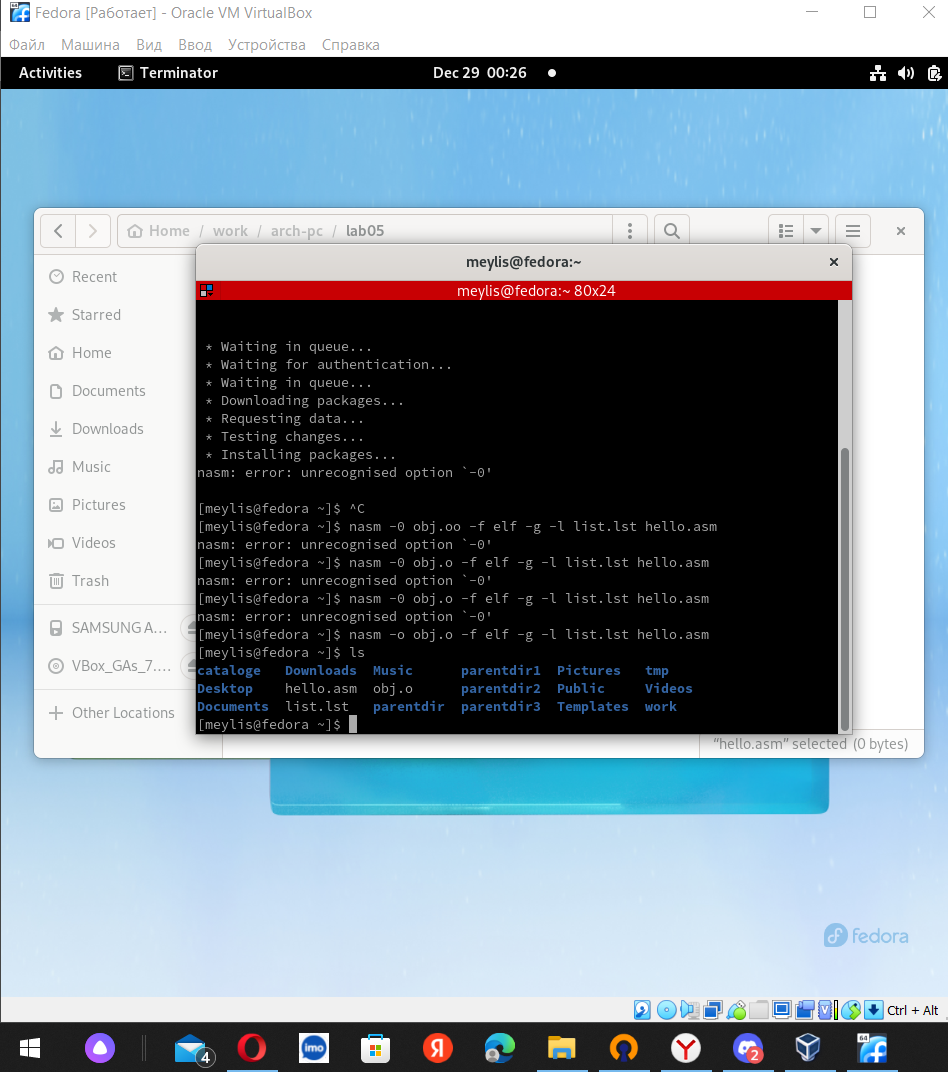
Данная команда скомпилирует исходный файл hello.asm в obj.o (опция o позволяет задать имя объектного файла, в данном случае obj.o), при этом формат выходного файла будет elf, и в него будут включены символы для отладки (опция -g), кроме того, будет создан файл листинга list.lst (опция

-l).

Архитектура ЭВМ

С помощью команды ls проверьте, что файлы были созданы.

Для более подробной информации см. man nasm. Для получения списка форматов объектного файла см. nasm -hf.



# 5.4. Компоновщик LD

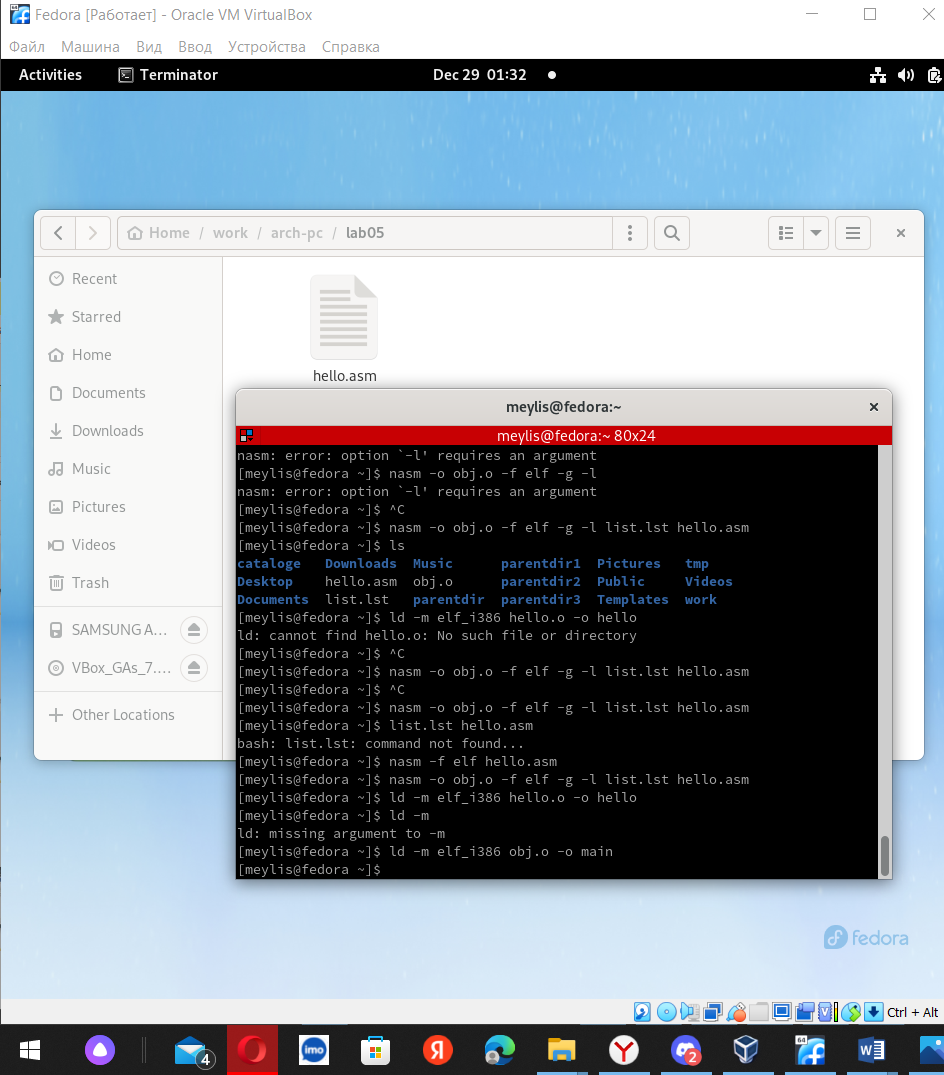
Как видно из схемы на рис. 5.3, чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику: ld -m elf\_i386 hello.o -o hello

С помощью команды ls проверьте, что исполняемый файл hello был создан.

Компоновщик ld не предполагает по умолчанию расширений для файлов, но принято использовать следующие расширения:

* o –для объектных файлов;
* без расширения –для исполняемых файлов; • map –для файлов схемы программы;
* lib –для библиотек.

Ключ -o с последующим значением задаёт в данном случае имя создаваемого исполняемого файла.

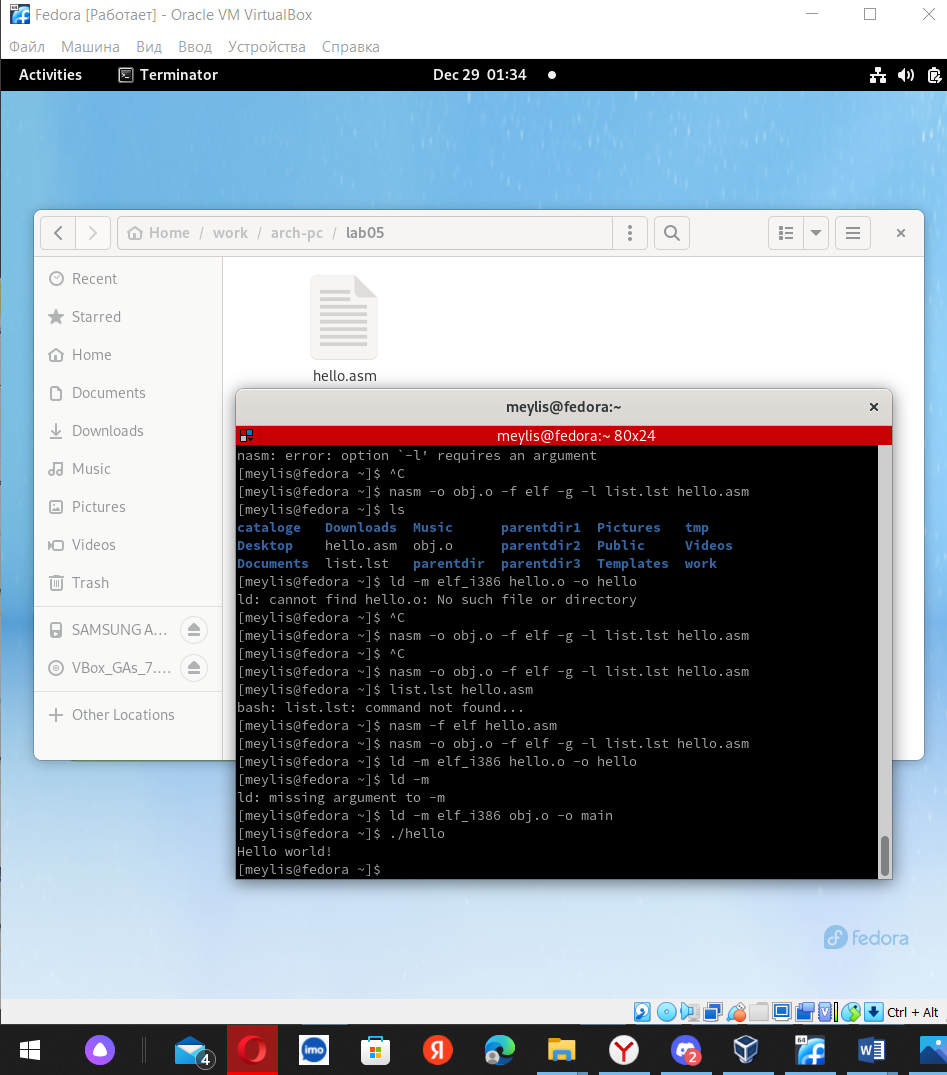
Выполните следующую команду: ld -m elf\_i386 obj.o -o main

Какое имя будет иметь исполняемый файл? Какое имя имеет объектный файл из которого собран этот исполняемый файл?

Формат командной строки LD можно увидеть, набрав ld --help. Для получения более подробной информации см. man ld.

Чарыев М. Г.

## 5.4.1. Запуск исполняемого файла

Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге, можно, набрав в командной строке: ./hello

# 5.5. Задание для самостоятельной работы

1. В каталоге ~/work/arch-pc/lab05 с помощью команды cp создайте копию файла hello.asm с именем lab5.asm
2. С помощью любого текстового редактора внесите изменения в текст программы в файле lab5.asm так, чтобы вместо Hello world! на экран выводилась строка с вашими фамилией и именем.
3. Оттранслируйте полученный текст программы lab5.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.
4. Скопируйте файлы hello.asm и lab5.asm в Ваш локальный репозиторий в каталог ~/work/study/2022-2023/"Архитектура компьютера"/archpc/labs/lab05/. Загрузите файлы на Github.

