Отчет по выполнению лабораторной работы №4

Дисциплина: Архитектура компьютера

Ефремова Полина Алекснадровна

Содержание

# 1 Цель работы

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

с основными возможностями разметки Markdown.

# 2 Задание

1. Знакомство с теоретической информацией (работа с языком “Ассемблер”)
2. Насписание команды ‘Hello world!’
3. Научиться работать с шаблоном, чтобы создавать собственные команды.

# 3 Теоретическое введение

Ассемблер (англ. «Assembler») — это низкоуровневый язык программирования, который представляет собой промежуточное звено между машинным кодом и высокоуровневыми языками программирования. Он используется для написания программ, которые управляют компьютером или другими устройствами на более низком уровне, непосредственно взаимодействуя с аппаратным обеспечением. Код, написанный на этом языке, обычно сохраняется с помощью расширения ASM.

Директивы

В языке ассемблера директивы — это специальные инструкции. Они используются для предоставления дополнительной информации ассемблеру или компоновщику, а не выполняются как часть программы. Директивы обычно обозначают специальным символом, например точкой или решеткой.

SECTION: эта директива нужна для определения разделов программы, которые используют для группировки связанного кода и данных вместе.

ORG: чтобы установить исходный или начальный адрес программы или раздела.

EQU: чтобы определить константы или символы, которые используют во всей программе.

DB, DW, DD: для определения значений данных байтов, слов или двойных слов в памяти.

ALIGN: для выравнивания ячейки памяти следующей инструкции или значения данных с указанной границей.

EXTERN, GLOBAL: чтобы указать, определяется ли символ внешне или глобально. Эту информацию использует компоновщик для разрешения ссылок на символы в разных объектных файлах.

INCLUDE: для включения файла кода на языке ассемблера в текущую программу.

Директивы помогают управлять структурой и организацией программы на языке ассемблера, указывать дополнительную информацию для создания конечной исполняемой программы.

Команды

Команды языка ассемблера — основные строительные блоки программ. Эти инструкции используют, чтобы сообщить процессору, какие операции следует выполнять. В одних архитектурах сотни или тысячи различных инструкций, в других может быть всего несколько десятков.

Основные:

Команды перемещения данных. Перемещают данные между регистрами или ячейками памяти: MOV, PUSH и POP.

Арифметические команды. Выполняют арифметические операции с данными в регистрах или ячейках памяти: ADD, SUB и MUL.

Логические команды. Выполняют логические операции с данными в регистрах или ячейках памяти: AND, OR и XOR.

Команды ветвления. Управляют путем перехода к другому разделу кода: JMP, JZ и JE.

Команды стека. Управляют стеком — областью памяти для хранения данных — и управляющей информацией во время вызовов функций и возвратов: PUSH и POP.

Системные вызовы. Позволяют программам на ассемблере взаимодействовать с операционной системой или другими системными функциями, такими как INT, которые запускают программное прерывание.

Программирование на языке assembler не так распространено, как раньше. Но всё еще есть отрасли и приложения, где он нужен. Например, встроенные системы, разработка операционных систем и реверс-инжиниринг.

# 4 Выполнение лабораторной работы

Создаю каталог для работы с программами на языке ассемблера (рис. -fig. 1)

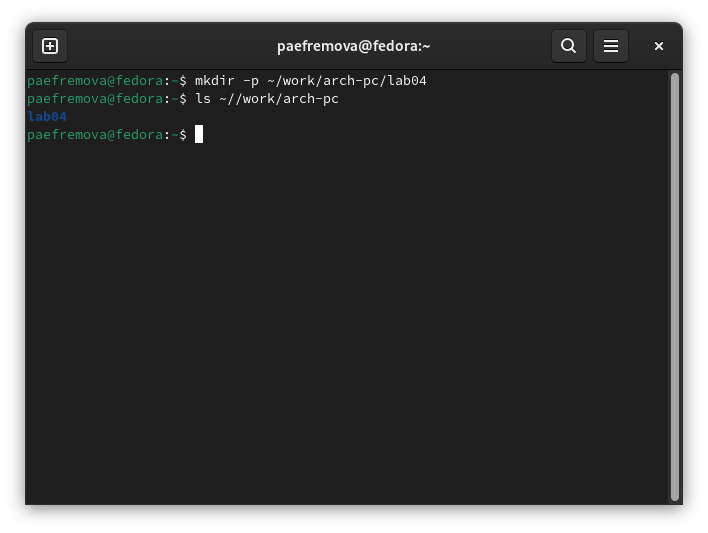


Рис. 1: Новый каталог

Перехожу в созданный каталог и создаю текстовый файл с именем hello.asm (рис. -fig. 2)

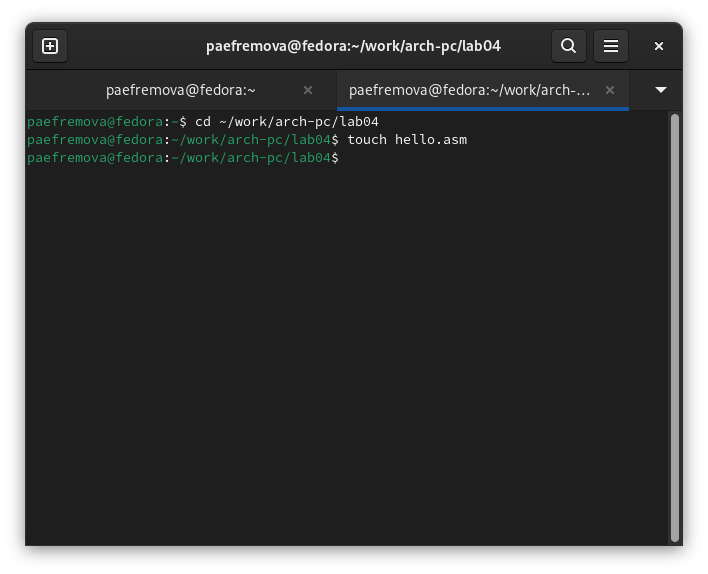


Рис. 2: Новый файл

Открываю созданный файл (рис. -fig. 3)

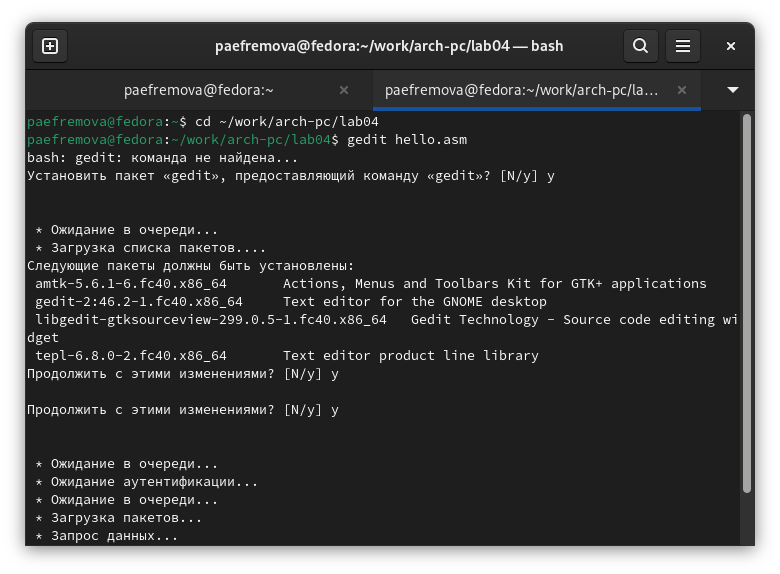


Рис. 3: Открытие файла

(рис. -fig. 4)

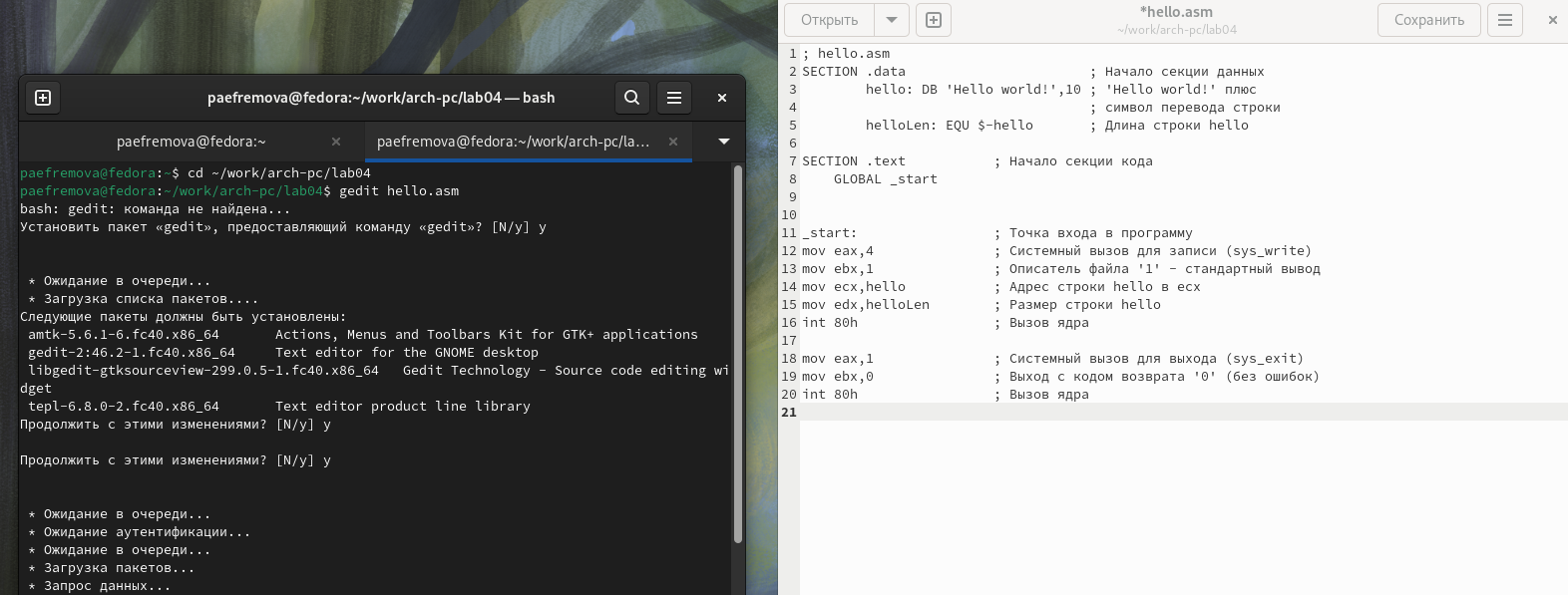


Рис. 4: Открытие файла

Для компиляции приведённого выше текста программы «Hello World» использую команду (рис. -fig. 5)

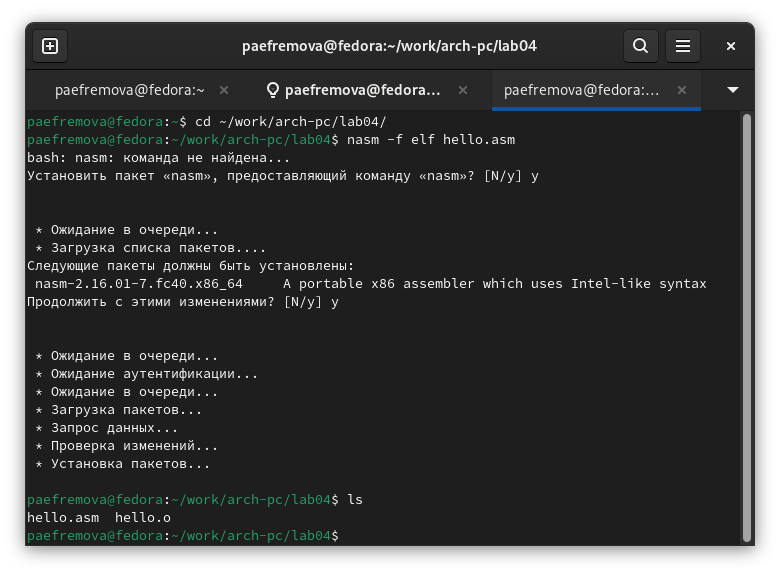


Рис. 5: Компиляция текста программы

Компилирую исходный файл hello.asm в obj.o (рис. -fig. 6)

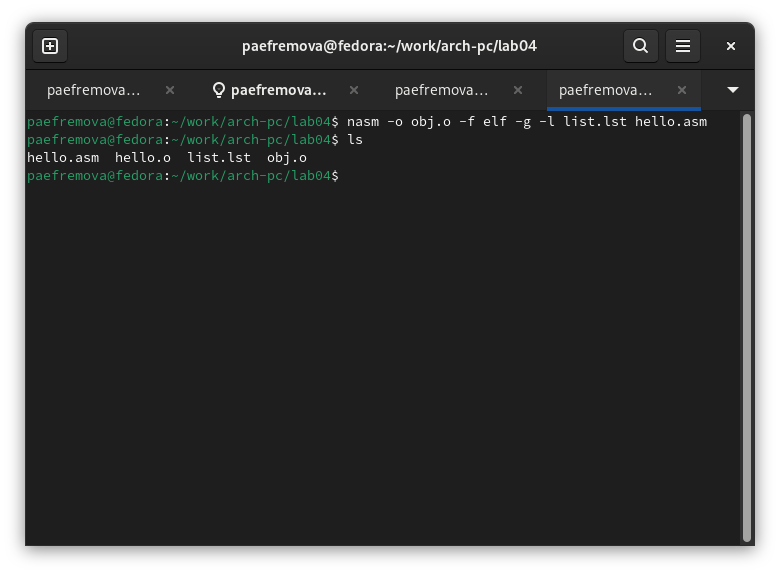


Рис. 6: Компиляция в obj.o

Передача объектного файла на работу компановщику (рис. -fig. 7)

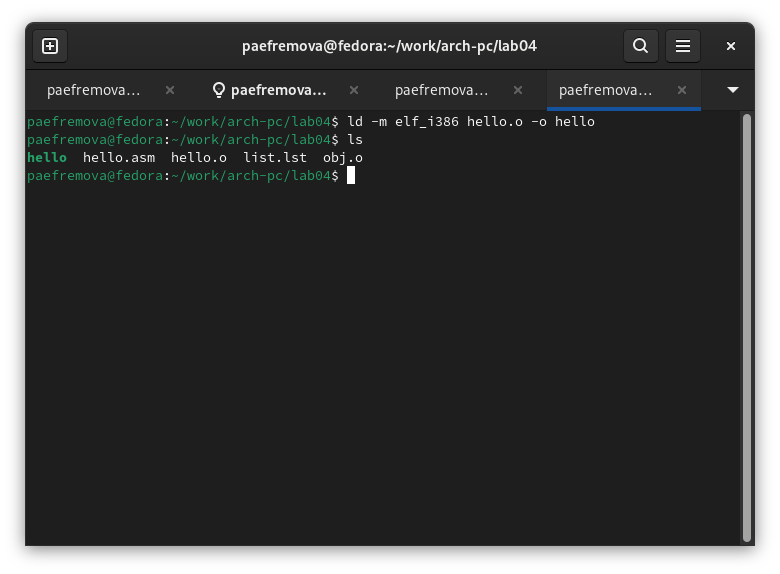


Рис. 7: Передача файла

Выполняю еще одну команду (рис. -fig. 8)

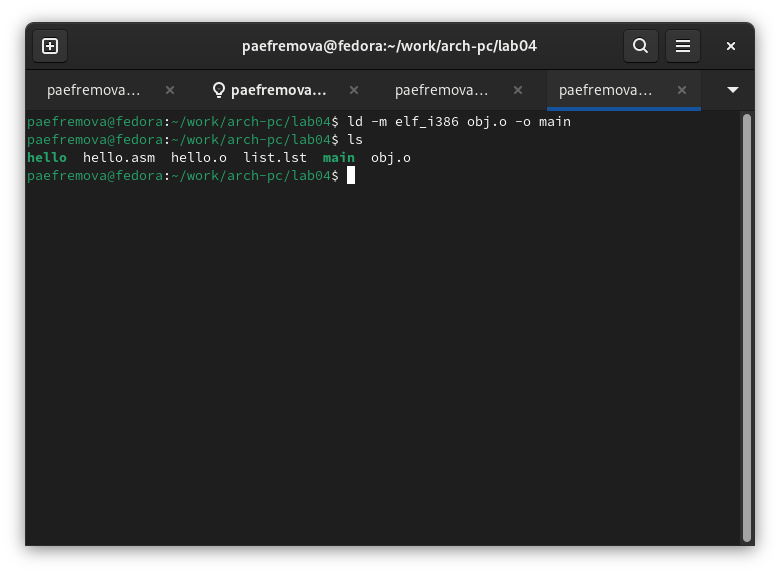


Рис. 8: Выполнение команды

Запускаю на выполнение созданный файл (рис. -fig. 9)

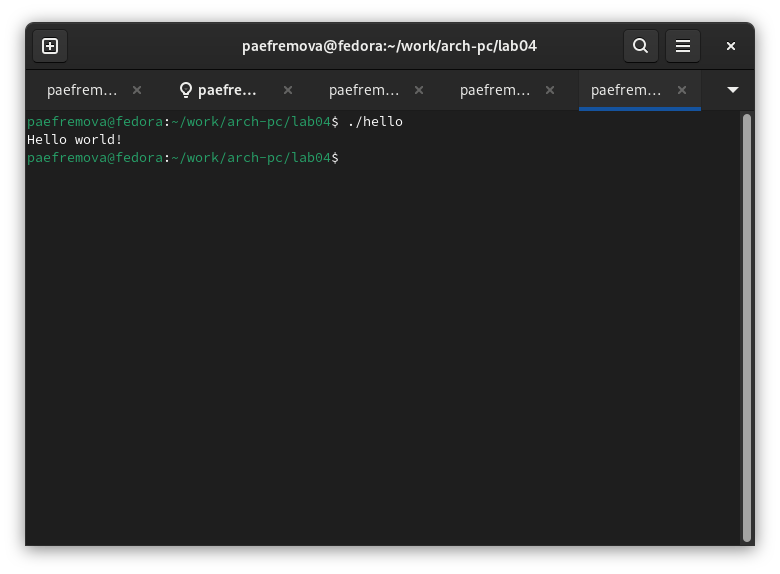


Рис. 9: Выполнение команды

# 5 Задание для самостоятельной работы

Перехожу в нужный каталог (рис. -fig. 10)

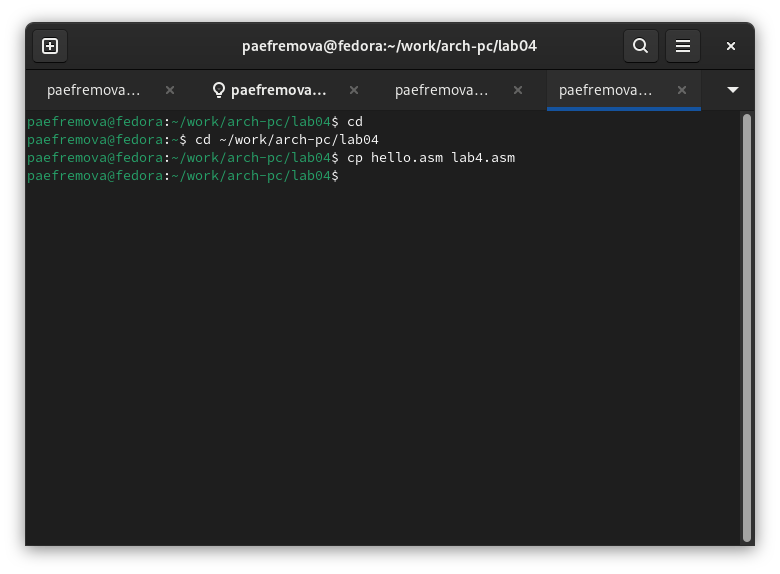


Рис. 10: Переход

Создайю копию файла hello.asm с именем lab4.asm и начинаю его редактировать (рис. -fig. 11)

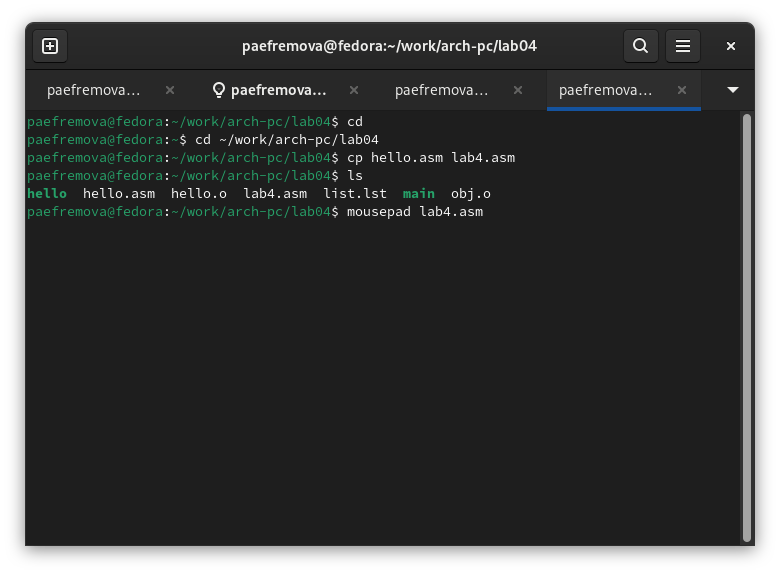


Рис. 11: Копирование

С помощью любого текстового редактора вношу изменения в текст программы в файле lab4 и вывожу полученный результат (рис. -fig. 12)

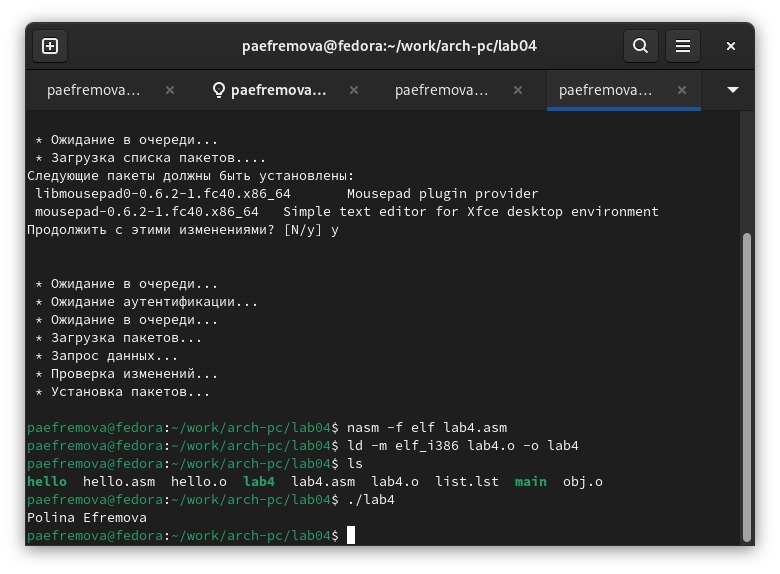


Рис. 12: Вывод собственной команды

Копирую файлы в свой локальный репозиторий (рис. -fig. **¿fig:013?**)

![(image/13.png){#fig:013 width=70%}

Провожу загрузку на GitHub (рис. -fig. **¿fig:014?**)

![(image/14.png){#fig:014 width=70%}

# 6 Выводы

Данная лабораторная работа позволила мне познакомиться с таким языком как Ассемблер. Теперь я знаю как писать простейшие команды на данном языке через консоль. В заключение хочется отметить, что пцсть приведенный язык и не считается популярным, я все же убеждаюсь в его важности.

# Список литературы

[Ассемблер: что это за язык программирования, для чего нужен, пример кода] (https://sky.pro/media/chto-takoe-assembler/).

[Архитектура ЭВМ](https://esystem.rudn.ru/pluginfile.php/2089084/mod_resource/content/0/Лабораторная%20работа%20№4.%20Создание%20и%20процесс%20обработки%20программ%20на%20языке%20ассемблера%20NASM.pdf)

[Что такое язык ассемблера и кому его нужно изучать / Skillbox Media] (https://skillbox.ru/media/code/chto-takoe-assembler/)