Отчет по лабораторной работе №4

Создание и процесс обработки программ на языке ассемблера NASM

Богданюк Анна Васильевна

Содержание

# Цель работы

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

# Задание

1.Выполнение лабораторной работы

2.Задания для самостоятельной работы

# 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). Используемые мнемоники обычно одинаковы для всех процессоров одной архитектуры или семейства архитектур (среди широко известных — мнемоники процессоров и контроллеров x86, ARM, SPARC, PowerPC,M68k). Таким образом для каждой архитектуры существует свой ассемблер и, соответственно, свой язык ассемблера. Наиболее распространёнными ассемблерами для архитектуры x86 являются: • для DOS/Windows: Borland Turbo Assembler (TASM), Microsoft Macro Assembler (MASM) и Watcom assembler (WASM); • для GNU/Linux: gas (GNU Assembler), использующий AT&T-синтаксис, в отличие от большинства других популярных ассемблеров, которые используют Intel-синтаксис.

В процессе создания ассемблерной программы можно выделить четыре шага: • 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. • Трансляция — преобразование с помощью транслятора, например nasm, текста программы в машинный код, называемый объектным. На данном этапе также может быть получен листинг программы, содержащий кроме текста программы различную дополнительную информацию, созданную транслятором. Тип объектного файла — o, файла листинга — lst. • 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. • 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.

# Выполнение лабораторной работы

1.1. Программа Hello world!

Создаю каталог для работы с программами на языке ассеблера NASM (рис. [1](#fig:001)).

Figure 1: Создаю каталог

Figure 1: Создаю каталог

Перехожу в созданный каталог (рис. [2](#fig:002)).

Figure 2: Перехожу в каталог

Figure 2: Перехожу в каталог

Создаю текстовый файл с именем hello.asm (рис. [3](#fig:003)).

Figure 3: Создаю hello.asm

Figure 3: Создаю hello.asm

Открываю этот файл с помощью nano (рис. [4](#fig:004)).

Figure 4: Использую nano

Figure 4: Использую nano

Ввожу следующий текст (рис. [5](#fig:005)).

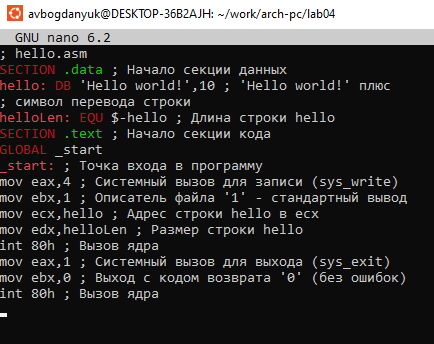


Figure 5: Ввожу текст

1.2. Транслятор NASM

Компилирую текст в объектный код (рис. [6](#fig:006)).

Figure 6: Компилирую текст

Figure 6: Компилирую текст

Проверяю с помощью ls, что был создан объектный файл (рис. [7](#fig:007)).

Figure 7: Файлы в каталоге

Figure 7: Файлы в каталоге

1.3. Расширенный синтаксис командной строки NASM

Компилирую файл hello.asm в obj.o, а также создаю файл листинга list.lst (рис. [8](#fig:008)).

Figure 8: Компилирую и создаю файлы

Figure 8: Компилирую и создаю файлы

Проверяю, что файлы были созданы (рис. [9](#fig:009)).

Figure 9: Файлы в каталоге

Figure 9: Файлы в каталоге

1.4. Компоновщик LD

Передаю объектный файл на обработку компоновщику(рис. [10](#fig:010)).

Figure 10: Передаю файл на обработку

Figure 10: Передаю файл на обработку

Проверяю, что файл был создан (рис. [11](#fig:011)).

Figure 11: Файлы в каталоге

Figure 11: Файлы в каталоге

Выполняю следующую команду. Созданый файл будет называться main, объектный файл, из которого собран этот исполняемый файл, называется obj.o (рис. [12](#fig:012)).

Figure 12: Создаю исполняемый файл

Figure 12: Создаю исполняемый файл

1.5. Запуск исполняемого файла

Запускаю на выполнение созданный исполняемый файл (рис. [13](#fig:013)).

Figure 13: Запускаю файл

Figure 13: Запускаю файл

1. Задание для самостоятельной работы

Копирую файл hello.asm с именем lab4.asm (рис. [14](#fig:014)).

Figure 14: Копирую файл

Figure 14: Копирую файл

Изменяю текст, чтобы вместо ‘Hello World!’, было ‘Богданюк Анна’ (рис. [15](#fig:015)).

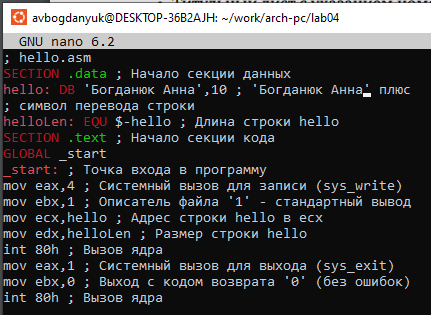


Figure 15: Измененый текст

Оттрансилую полученный файл в объектный файл и проверяю, что файл был создан (рис. [16](#fig:016)).

Figure 16: Команда

Figure 16: Команда

Выполняю компоновку объектного файла и проверяю, что файл был создан(рис. [17](#fig:017)).

Figure 17: Выполняю компоновку

Figure 17: Выполняю компоновку

Запускаю получившийся исполняемый файл (рис. [18](#fig:018)).

Figure 18: Результат

Figure 18: Результат

Копирую файл hello.asm и lab4.asm в мой локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/ (рис. [19](#fig:019)).

Figure 19: Копирую файлы

Figure 19: Копирую файлы

Загружаю файлы на GitHub (рис. [20](#fig:020)).

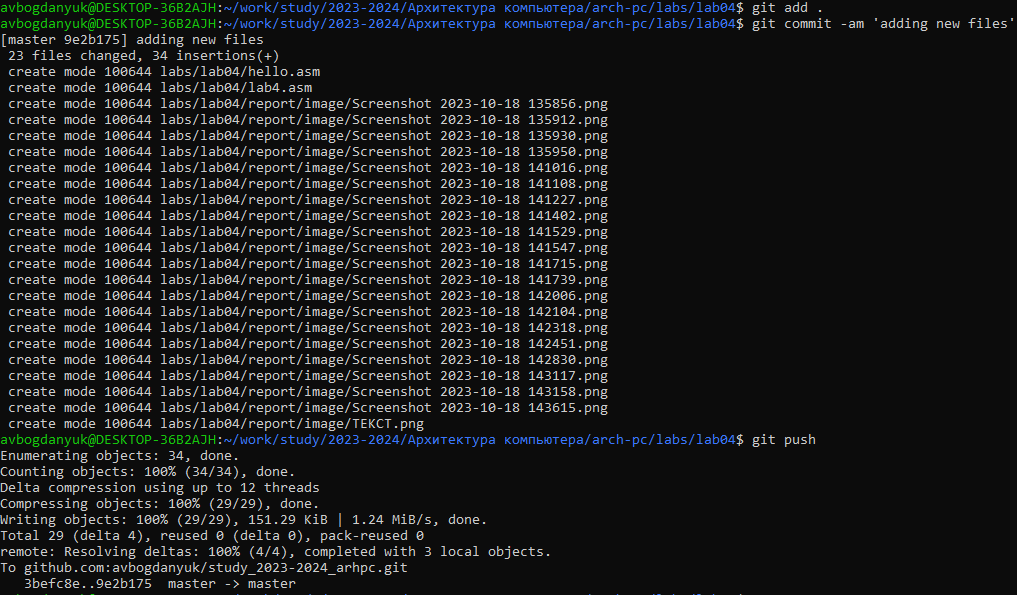


Figure 20: Загружаю файлы

# Выводы

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

# Список литературы