Отчёт по лабораторной работе №4

Дисциплина: Архитектура компьютера

Осина Виктория Александровна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Целью данной работы является освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Задание

1. Создание программы Hello world!
2. Компиляция текста программы при помощи транслятора NASM
3. Работа с расширенным синтаксисом командной строки NASM
4. Передача объектного файла на обработку компановщику LD
5. Запуска исполняемого файла
6. Задания для самостоятельной работы

3 Теоретическое введение

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язы
Следует отметить, что процессор понимает не команды ассемблера, а последовательно
Программы, написанные на языке ассемблера, не уступают в качестве и скорости прог
Используемые мнемоники обычно одинаковы для всех процессоров одной архитектуры ил
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.

NASM — это открытый проект ассемблера, версии которого доступны под различные опе
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 Программа Hello world!

Создаю каталог для работы с программами на языке ассемблера NASM и перехожу в созданный каталог(рис. [**??**])

Рис. 1. Создание каталога

Создаю текстовый файл с именем hello.asm (рис. [**??**])

Рис. 2. Создание текстового файла

Открываю этот файл с помощью текстового редактора gedit (рис. [**??**])

Рис. 3. Открытие файла в текстовом редакторе

и ввожу в него необходимый текст (рис. [**??**])

Рис. 4. Ввод текста

4.2 Транслятор NASM

Для компиляции текста программы «Hello World» пишу следующую команду (рис. [**??**])

Рис. 5. Компиляция программы

С помощью команды ls проверяю, что объектный файл был создан. Файл создается в формате ELF (рис. [**??**])

Рис. 6. Проверка создания файла

4.3 Расширенный синтаксис командной строки NASM

Выполняю следующую команду: (рис. [**??**])

Рис. 7. Выполнение команды

С помощью команды ls проверяю, что файл был создан (рис. [**??**])

Рис. 8. Проверка создания файла

4.4 Компоновщик LD

Чтобы получить исполняемую программу, передаю объектный файл на обработку компоновщику LD (рис. [**??**])

Рис. 9. Передача файла компановщику

С помощью команды ls проверяю, что исполняемый файл hello был создан. (рис. [**??**])

Рис. 10. Проверка создания файла

Выполняю следующую команду: (рис. [**??**])

Рис. 11. Выполнение команды

Исполняемый файл имеет имя main, объектный файл имеет имя obj.o

4.5 Запуск исполняемого файла

Запускаю на выполнение созданный исполняемый файл, находящийся в текущем каталоге, при помощи команды ./hello (рис. [**??**])

Рис. 12. Запуск созданного файла на выполнение

4.6 Задание для самостоятельной работы

В каталоге ~/work/arch-pc/lab04 с помощью команды cp создаю копию файла hello.asm с именем lab4.asm (рис. [**??**])

Рис. 13. Создание копий файлов

С помощью текстового редактора gedit вношу изменения в текст программы в файле lab4.asm так, чтобы вместо Hello world! на экран выводилась строка с моими фамилией и именем (рис. [**??**])

Рис. 14. Изменений текста программы

Транслирую полученный текст программы lab4.asm в объектный файл (рис. [**??**])

Рис. 15. Трансляция текста в объектный файл

Выполняю компоновку объектного файла (рис. [**??**])

Рис. 16. Выполнение комановки

Запускаю получившийся исполняемый файл (рис. [**??**])

Рис. 17. Запуск получившегося файла

Копирую файлы hello.asm и lab4.asm в мой локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/. (рис. [**??**]), (рис. [**??**]) и (рис. [**??**])

Рис. 18. Создание копии файла hello.asm

Рис. 19. Создание копии файла lab04.asm

Рис. 20. Проверка создания копий

Загружаю файлы на Github (рис. [**??**])

Рис. 21. Загрузка файлов на Github

5 Выводы

Я освоила процедуру компиляции и сборки программ, написанных на ассемблере NASM

6 Список литературы

::: Архитектура ЭВМ - Лабораторная работа №4 :::