**Архитектура компьютера**

**Отчёт по лабораторной работе №4**

**Лю Сяо**

**1 Цель работы**

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM

**2 Задание**

1) Создать программу Hello world

2) Работа с транслятором NASM

3) Работа с расширенным синтаксисом командой строки NASM

4) Работа с компоновщиком LD

5) Запуск исполняемого файла

6) Выполнение заданий для самостоятельной работы

**3 Теоретическое введение**

Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора (ЦП) входят следующие устройства: • арифметико-логическое устройство (АЛУ) — выполняет логические и арифметические действия, необходимые для обработки информации, хранящейся в памяти; • устройство управления (УУ) — обеспечивает управление и контроль всех устройств компьютера; • регистры — сверхбыстрая оперативная память небольшого объёма, входящая в состав процессора, для временного хранения промежуточных результатов выполнения инструкций; регистры процессора делятся на два типа: регистры общего назначения и специальные регистры 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 7map. • 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. В качестве примера приведем названия основных регистров общего назначения (именно эти регистры чаще всего используются при написании программ): • RAX, RCX, RDX, RBX, RSI, RDI — 64-битные • EAX, ECX, EDX, EBX, ESI, EDI — 32-битные • AX, CX, DX, BX, SI, DI — 16-битные • AH, AL, CH, CL, DH, DL, BH, BL — 8-битные (половинки 16-битных регистров). Например, AH (high AX) — старшие 8 бит регистра AX, AL (low AX) — младшие 8 бит регистра AX. В состав ЭВМ также входят периферийные устройства, которые можно разделить на: • устройства внешней памяти, которые предназначены для долговременного хранения больших объёмов данных (жёсткие диски, твердотельные накопители, магнитные ленты); • устройства ввода-вывода, которые обеспечивают взаимодействие ЦП с внешней средой. В основе вычислительного процесса ЭВМ лежит принцип программного управления.Это означает, что компьютер решает поставленную задачу как последовательность действий, записанных в виде программы. Программа состоит из машинных команд, которые указывают, какие операции и над какими данными (или операндами), в какой последовательности необходимо выполнить. Набор машинных команд определяется устройством конкретного процессора. Коды команд представляют собой многоразрядные двоичные комбинации из 0 и 1. В коде машинной команды можно выделить две части: операционную и адресную. В операционной части хранится код команды, которую необходимо выполнить. В адресной части хранятся данные или адреса данных, которые участвуют в выполнении данной операции. При выполнении каждой команды процессор выполняет определённую последовательность стандартных действий, которая называется командным циклом процессора. В самом общем виде он заключается в следующем: 1. формирование адреса в памяти очередной команды; 2. считы8вание кода команды из памяти и её дешифрация; 3. выполнение команды; 4. переход к следующей команде.

**4 Выполнение лабораторной работы**

1) Создаю рекурсивно вложенные в папку work папки arch-pc и lab04, проверяю их создание

2) Перехожу в созданную папку



Рис. 4.2: Переход в созданную папку

3) Создаю файл hello с разрешением asm и проверяю его создание

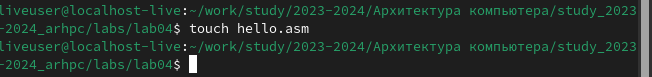


Рис. 4.3: Создание файла

4) Открываю этот файл в nano и копирую туда код из задания лабораторной работы

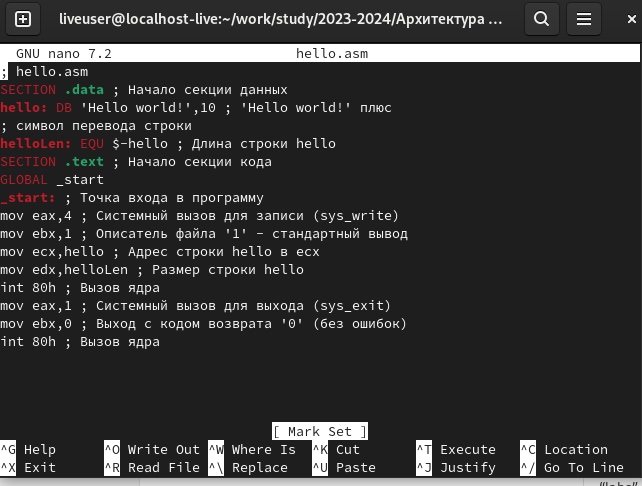


Рис. 4.4: Заполнение файла

5) Скачиваю nasm

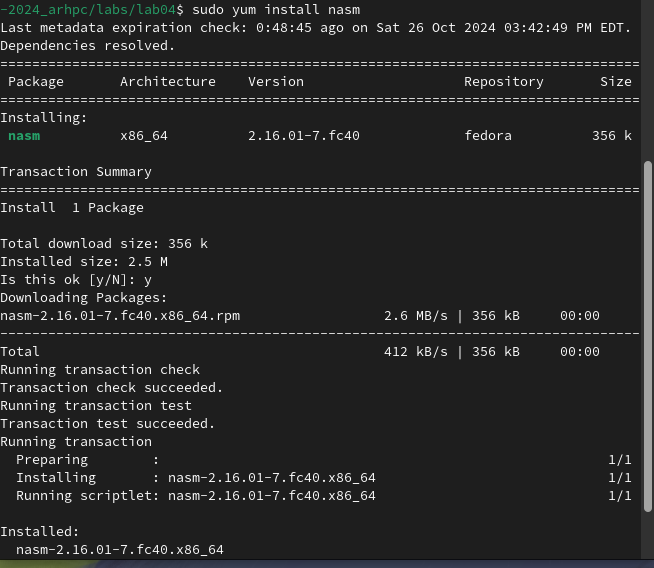


Рис. 4.5: Скачивание

6) Преобразовываю файл hello.asm в объектный код, записанный в файл hello.o. Проверяю, был ли создан файл

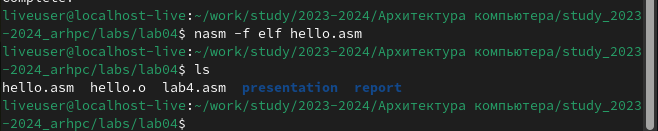


Рис. 4.6: Преобразование файла в объектный код

7) Преобразую файл hello.asm в obj.o с помощью опции -o, которая позволяет задать имя объекта. Из-за elf -g формат выходного файла будет elf, и в него удут включены символы для отладки, а так же будет создан файл листинга list.lst, благодаря -l. Проверяю созданные файлы

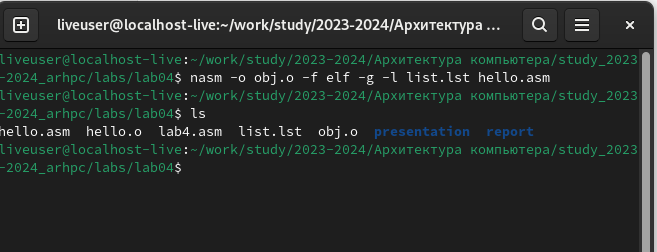


Рис. 4.7: Преобразование файла

8) Передаю файл компановщику с помощью ld. Проверяю, создан ли исполняемый файл

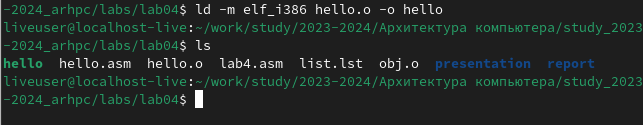


Рис. 4.8: Передача файла на обработку

9) Передаю компановщику файл obj.o и называю скомпанованный файл main. запуская сначала код для предыдущего файла(1), а затем для созданного сейчас(2)

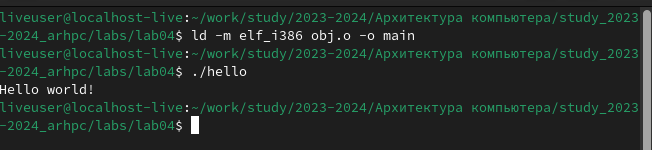


Рис. 4.9: 1

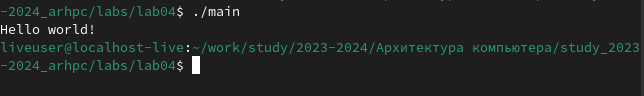


Рис. 4.9 : 2

**5 Выполнение заданий для самостоятельной работы**

1) Копирую hello.asm с названием lab4.asm !Копирование файла](image/10.jpg){#ffg:001 width=70%}

2) С помощью nano изменяю текст кода так, чтобы он выводил моё имя и фамилию

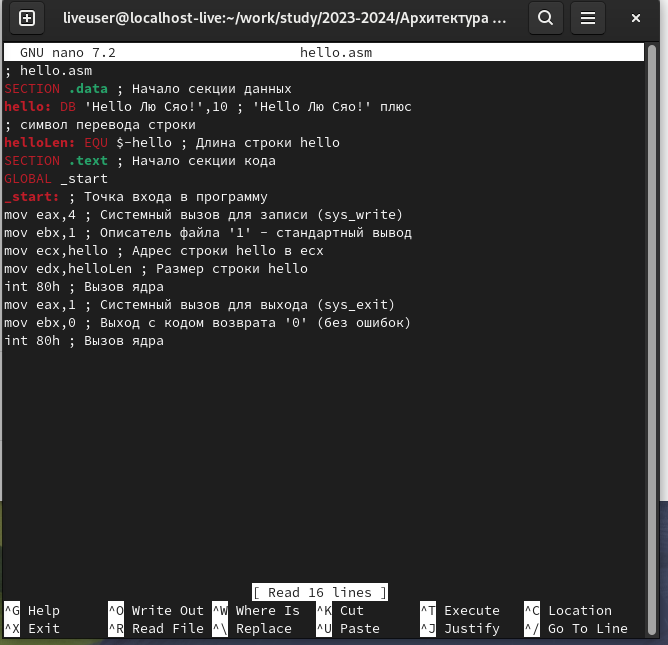


Рис. 5.1: Файл в nano

3) Транслирую файл lab4.asm в объектный



Рис. 5.2: Транслирую файл

4) Выполняю компановку и запускаю исполняемый файл

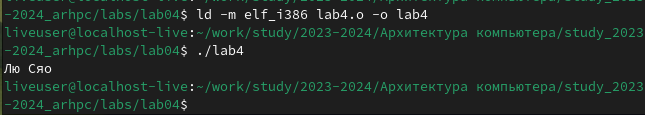


Рис. 5.3: Компановка и исполение

5) Выгружаю изменения на GitHub

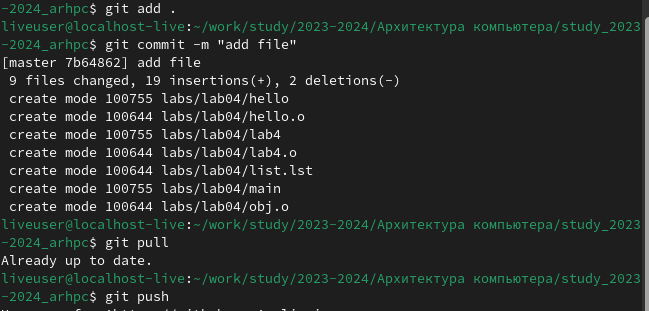


Рис. 5.5: Выгружаю изменения

6) Копирую файл с отчётом и начинаю его заполнять

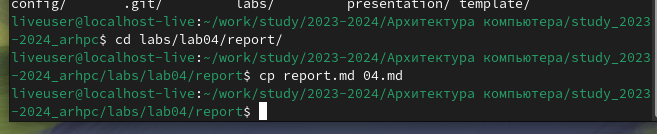


Рис. 5.6: Копирование файла с отчётом

**6Выводы**

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