

Отчёт по лабораторной работе №5.

Дисциплина: архитектура компьютера

Тимофеева Екатерина Николаевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Задания для самостоятельной работы	10
6	Выводы	12
	Список литературы	13

Список иллюстраций

4.1	Создание текстового файла	8
4.2	Написание текста	8
4.3	Переход от текста к объектному коду	9
4.4	Компиляция файла	9
4.5	Получение исполняемого файла	9
4.6	Запуск исполняемого файла	9
5.1	Создание копии	10
5.2	Внесение изменений	10
5.3	Запуск файла	11

Список таблиц

1 Цель работы

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

2 Задание

1. Создание программы Hello world!
2. Работа с транслятором NASM
3. Работа с расширенным синтаксисом командной строки NASM
4. Работа с компоновщиком LD
5. Запуск исполняемого файла
6. Выполнение заданий для самостоятельной работы.

3 Теоретическое введение

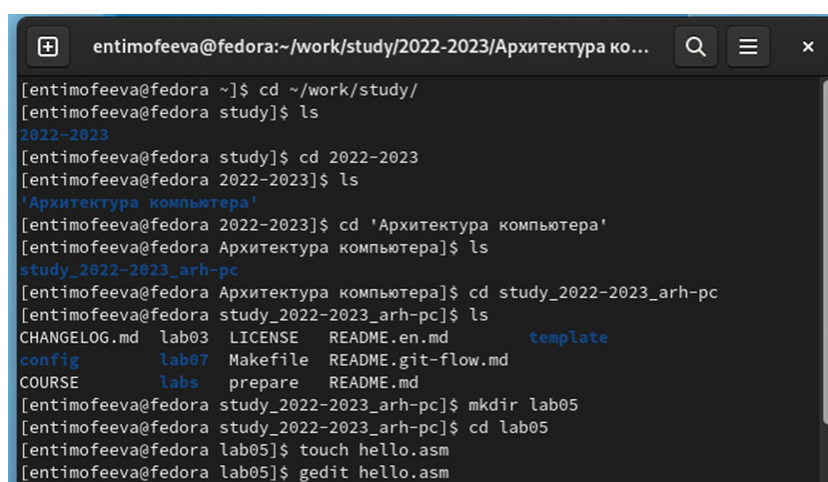
Основными функциональными элементами любой ЭВМ являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской плате. Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора входят следующие устройства:

1. арифметико-логическое устройство (АЛУ) – выполняет логические и арифметические
2. устройство управления (УУ) – обеспечивает управление и контроль всех устройств
3. регистры – сверхбыстрая оперативная память небольшого объёма, входящая в состав
4. RAX, RCX, RDX, RBX, RSI, RDI – 64-битные
5. EAX, ECX, EDX, EBX, ESI, EDI – 32-битные
6. AX, CX, DX, BX, SI, DI – 16-битные
7. AH, AL, CH, CL, DH, DL, BH, BL – 8-битные

Другим важным узлом ЭВМ является оперативное запоминающее устройство (ОЗУ).

4 Выполнение лабораторной работы

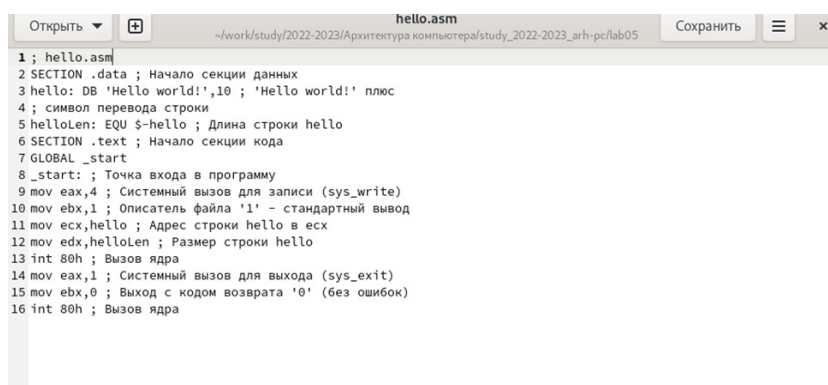
Создаём каталог для работы с программами на языке ассемблера NASM, переходим в созданный каталог, создаём в текущем каталоге пустой текстовый файл `hello.asm`, открываем созданный файл в текстовом редакторе. (рис. 4.1)



```
entimofeeva@fedora:~/work/study/2022-2023/Архитектура ко...
[entimofeeva@fedora ~]$ cd ~/work/study/
[entimofeeva@fedora study]$ ls
2022-2023
[entimofeeva@fedora study]$ cd 2022-2023
[entimofeeva@fedora 2022-2023]$ ls
'Архитектура компьютера'
[entimofeeva@fedora 2022-2023]$ cd 'Архитектура компьютера'
[entimofeeva@fedora Архитектура компьютера]$ ls
study_2022-2023_arh-pc
[entimofeeva@fedora Архитектура компьютера]$ cd study_2022-2023_arh-pc
[entimofeeva@fedora study_2022-2023_arh-pc]$ ls
CHANGELOG.md lab03 LICENSE README.en.md template
config lab07 Makefile README.git-flow.md
COURSE labs prepare README.md
[entimofeeva@fedora study_2022-2023_arh-pc]$ mkdir lab05
[entimofeeva@fedora study_2022-2023_arh-pc]$ cd lab05
[entimofeeva@fedora lab05]$ touch hello.asm
[entimofeeva@fedora lab05]$ gedit hello.asm
```

Рис. 4.1: Создание текстового файла

Введём в него текст.(рис. 4.2)



```
Открыть ▾ + hello.asm
~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab05 Сохранить ≡ x

1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5 helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для записи (sys_write)
10 mov ebx,1 ; Описатель файла '1' - стандартный вывод
11 mov ecx,hello ; Адрес строки hello в ecx
12 mov edx,helloLen ; Размер строки hello
13 int 80h ; Вызов ядра
14 mov eax,1 ; Системный вызов для выхода (sys_exit)
15 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16 int 80h ; Вызов ядра
```

Рис. 4.2: Написание текста

Превращаем текст программы для вывода “Hello world!” в объектный код с помощью транслятора NASM. Далее проверяем правильность выполнения команды с помощью утилиты ls; действительно, создан файл.(рис. 4.3)

```
[entimofeeva@fedora lab05]$ nasm -f elf hello.asm
[entimofeeva@fedora lab05]$ ls
hello.asm hello.o
```

Рис. 4.3: Переход от текста к объектному коду

Вводим команду, которая скомпилирует файл hello.asm в файл obj.o при этом в файл будут включены символы для отладки (ключ -g), также с помощью ключа будет создан файл листинга. Проверяем правильность выполнения команды.(рис. 4.4)

```
[entimofeeva@fedora lab05]$ nasm -o obj.o -f elf -g -l list.lst hello.asm
[entimofeeva@fedora lab05]$ ls
hello.asm hello.o list.lst obj.o
```

Рис. 4.4: Компиляция файла

Передаём объектный файл hello.o на обработку компоновщику LD, чтобы получить исполняемый файл. Ключ -o задает имя создаваемого исполняемого файла. Далее проверяем с помощью утилиты ls правильность выполнения команды.(рис. 4.5)

```
[entimofeeva@fedora lab05]$ ld -m elf_i386 hello.o -o hello
[entimofeeva@fedora lab05]$ ls
hello hello.asm hello.o list.lst obj.o
```

Рис. 4.5: Получение исполняемого файла

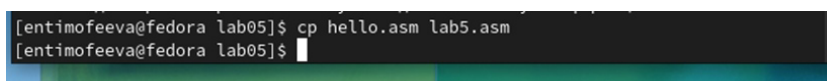
Выполняем следующую команду и запускаем созданный исполняемый файл.(рис. 4.6)

```
[entimofeeva@fedora lab05]$ ld -m elf_i386 obj.o -o main
[entimofeeva@fedora lab05]$ ls
hello hello.asm hello.o list.lst main obj.o
[entimofeeva@fedora lab05]$ ./hello
Hello world!
[entimofeeva@fedora lab05]$
```

Рис. 4.6: Запуск исполняемого файла

5 Задания для самостоятельной работы

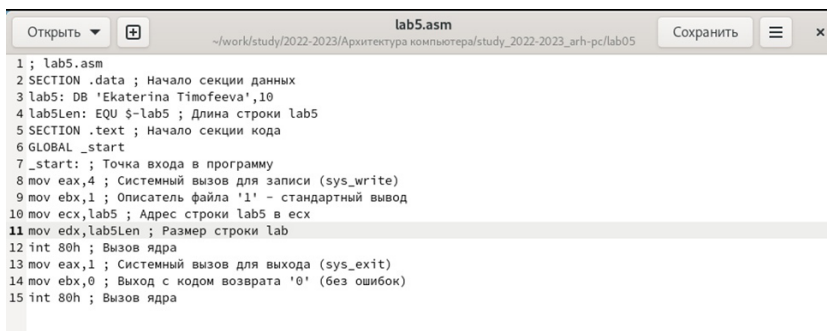
Создаём в текущем каталоге копию файла `hello.asm` с именем `lab5.asm` (рис. 5.1)



```
[entimofeeva@fedora lab05]$ cp hello.asm lab5.asm
[entimofeeva@fedora lab05]$
```

Рис. 5.1: Создание копии

С помощью текстового редактора открываем файл `lab5.asm` и вносим изменения в программу так, чтобы она выводила мои имя и фамилию.(рис. 5.2)



```
lab5.asm
~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab05
Сохранить

1 ; lab5.asm
2 SECTION .data ; Начало секции данных
3 lab5: DB 'Ekaterina Timofeeva',10
4 lab5Len: EQU $-lab5 ; Длина строки lab5
5 SECTION .text ; Начало секции кода
6 GLOBAL _start
7 _start: ; Точка входа в программу
8 mov eax,4 ; Системный вызов для записи (sys_write)
9 mov ebx,1 ; Описатель файла '1' - стандартный вывод
10 mov ecx,lab5 ; Адрес строки lab5 в ecx
11 mov edx,lab5Len ; Размер строки lab
12 int 80h ; Вызов ядра
13 mov eax,1 ; Системный вызов для выхода (sys_exit)
14 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
15 int 80h ; Вызов ядра
```

Рис. 5.2: Внесение изменений

Компилируем текст программы в объектный файл, передаём объектный файл `lab5.o` на обработку компоновщику LD, чтобы получить исполняемый файл `lab5` и запускаем исполняемый файл.(рис. 5.3)

```

[entimofeeva@fedora study_2022-2023_arh-pc]$ cd lab05
[entimofeeva@fedora lab05]$ ls
hello hello.asm hello.o lab5.asm list.lst main obj.o
[entimofeeva@fedora lab05]$ cd hello.asm
bash: cd: hello.asm: Это не каталог
[entimofeeva@fedora lab05]$ gedit hello.asm
[entimofeeva@fedora lab05]$ cp hello.asm
cp: после 'hello.asm' пропущен операнд, задающий целевой файл
По команде «cp --help» можно получить дополнительную информацию.
[entimofeeva@fedora lab05]$ cp hello.asm lab5.asm
[entimofeeva@fedora lab05]$ gedit lab5.asm
[entimofeeva@fedora lab05]$ nasm -f elf lab5.asm
[entimofeeva@fedora lab05]$ ls
hello hello.asm hello.o lab5.asm lab5.o list.lst main obj.o
[entimofeeva@fedora lab05]$ ld -m elf_i386 lab5.o -o lab5
[entimofeeva@fedora lab05]$ ls
hello hello.asm hello.o lab5 lab5.asm lab5.o list.lst main obj.o
[entimofeeva@fedora lab05]$ ./lab5
Ekaterina Timofeeva
[entimofeeva@fedora lab05]$

```

Рис. 5.3: Запуск файла

Добавляем файлы на GitHub.

6 Выводы

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

Список литературы

Демидова А.В. “Лабораторная работ №5” - методический материал