

Отчёт по лабораторной работе №4

Дисциплина: Архитектура компьютера

Абакумов Тимофей Александрович

Содержание

1	Цель работы.....	1
2	Задание	1
3	Выполнение лабораторной работы.....	2
3.1	Создание программы Hello world!.....	2
3.2	Работа с транслятором NASM.....	2
3.3	Работа с расширенным синтаксисом командной строки NASM.....	3
3.4	Работа с компоновщиком LD	3
3.5	Запуск исполняемого файла	3
4	Задание для самостоятельной работы	3
5	Выводы	5

1 Цель работы

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

2 Задание

Порядок выполнения лабораторной работы

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

Задание для самостоятельной работы

1. В каталоге ~/work/arch-pc/lab04 с помощью команды cp создайте копию файла hello.asm с именем lab4.asm
2. С помощью любого текстового редактора внесите изменения в текст программы в файле lab4.asm так, чтобы вместо Hello world! на экран выводилась строка с вашими фамилией и именем.

3. Оттранспируйте полученный текст программы lab4.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.
4. Скопируйте файлы hello.asm и lab4.asm в Ваш локальный репозиторий в каталог ~/work/study/2024-2025/“Архитектура компьютера”/arch-pc/labs/lab04/. Загрузите файлы на Github.

3 Выполнение лабораторной работы

3.1 Создание программы Hello world!

С помощью утилиты cd переместимся в каталог, в котором будем работать (рис. 1).

```
taabakumov@dk8n68 ~ $ cd ~/work/study/2024-2025/“Архитектура компьютера”
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера $ cd arch-pc/labs/lab04
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 1: Перемещение между директориями

Создадим в текущем каталоге пустой текстовый файл hello.asm с помощью утилиты touch (рис. 2).

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ touch hello.asm
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 2: Создание текстового файла

Откроем созданный файл в текстовом редакторе gedit и вставим туда текст из Лабораторной работы (рис. 3).

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ gedit hello.asm
```

Рис. 3: Работа с текстовым файлом

3.2 Работа с транслятором NASM

Превратим текст программы для вывода “Hello world!” в объектный код с помощью транслятора NASM, используя команду `nasm -f elf hello.asm`, ключ `-f` указывает транслятору nasm, что требуется создать бинарный файл в формате ELF (рис. 4).

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf hello.asm
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $
```

Рис. 4: Работа с транслятором NASM

Далее проверяю правильность выполнения команды с помощью утилиты ls: действительно, создан файл “hello.o”.

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm hello.o presentation report
```

Рис. 5: Проверка правильности

3.3 Работа с расширенным синтаксисом командной строки NASM

Введём 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 `-l` будет создан файл листинга `list`. Далее проверим с помощью утилиты `ls` правильность выполнения команды.

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o  presentation  report
```

Рис. 6: Работа с расширенным синтаксисом командной строки NASM

3.4 Работа с компоновщиком LD

Передадим объектный файл `hello.o` на обработку компоновщику `LD`, чтобы получить исполняемый файл `hello` (рис. 7). Ключ `-o` задает имя создаваемого исполняемого файла. Далее проверим с помощью утилиты `ls` правильность выполнения команды.

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 hello.o -o hello
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o  presentation  report
```

Рис. 7: Работа с компоновщиком LD

Выполним следующую команду (рис. 8). Исполняемый файл будет иметь имя `main`, т.к. после ключа `-o` было задано значение `main`. Объектный файл, из которого собран этот исполняемый файл, имеет имя `obj.o`

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 obj.o -o main
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  list.lst  main  obj.o  presentation  report
```

Рис. 8: Передача объектного файла на обработку компоновщику

3.5 Запуск исполняемого файла

Запускаю на выполнение созданный исполняемый файл `hello` (рис. 9).

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ./hello
Hello world!
```

Рис. 9: Запуск исполняемого файла

4 Задание для самостоятельной работы

С помощью утилиты `cp` создаю в текущем каталоге копию файла `hello.asm` с именем `lab4.asm` (рис. 10).

```
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ cp hello.asm lab4.asm
taabakumov@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o  presentation  report
```

Рис. 10: Создание копии файла

С помощью утилиты `mc` открываю файл `lab4.asm` и вношу изменения в программу так, чтобы она выводила мои имя и фамилию. (рис. 11).

```
lab04:mc — Konsole

[Новая вкладка] [Разделить окно] [Копировать]

/afs/.dk.sci.pfu.edu.ru/home/t/a/taabakumov/work/study/2024-2025/ Архитектура компьютера/arch-pc/labs/1
; lab4.asm

SECTION .data ; Начало секции данных

lab4: DB 'Абакумов Тимофей',10 ; 'Абакумов Тимофей' плюс символ перевода строки

lab4Len: EQU $-lab4 ; Длина строки lab4

SECTION .text ; Начало секции кода

GLOBAL _start

_start: ; Точка входа в программу

mov eax,4 ; Системный вызов для записи (sys_write)

mov ebx,1 ; Описатель файла '1' - стандартный вывод

mov ecx,lab4 ; Адрес строки lab4 в ecx

mov edx,lab4Len ; Размер строки lab4

int 80h ; Вызов ядра

mov eax,1 ; Системный вызов для выхода (sys_exit)

mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)

int 80h ; Вызов ядра
```

Рис. 11: Изменение программы

Компилирую текст программы в объектный файл. Затем передадим объектный файл `lab4.o` на обработку компоновщику `LD`, чтобы получить исполняемый файл `lab4`. Запустим исполняемый файл `lab4` (рис. 12).

```
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf lab4.asm
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o obj.o -f elf -g -l list.lst lab4.asm
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 lab4.o -o lab4
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ./lab4
Абакумов Тимофей
```

Рис. 12: Компиляция текста программы

С помощью команд `git add .` и `git commit` добавляю файлы на GitHub, комментируя действие как добавление файлов для лабораторной работы №5. После этого отправляем файлы на сервер с помощью команды `git push` (рис. 13).

```
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ git add .
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ git commit -am 'feat(main): make course structure'
[main 2c8eade] feat(main): make course structure
9 files changed, 96 insertions(+)
create mode 100755 labs/lab04/hello
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/hello.o
create mode 100755 labs/lab04/lab4
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/lab4.o
create mode 100644 labs/lab04/list.lst
create mode 100755 labs/lab04/main
create mode 100644 labs/lab04/obj.o
taabakumov@dk8n58 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ git push
Перечисление объектов: 100% (16/16), готово.
Подсчет объектов: 100% (16/16), готово.
При скатии изменений используется до 6 потоков
Сжатие объектов: 100% (12/12), готово.
Запись объектов: 100% (13/13), 3.63 КБ | 3.63 МБ/с, готово.
Total 13 (delta 7), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (7/7), completed with 2 local objects.
To github.com:Timofey2210/study_2024-2025_arch-pc.git
e0894a..2c8eade master -> master
```

Рис. 13: Отправка файлов

Проверим на GitHub все файлы(рис. 14)

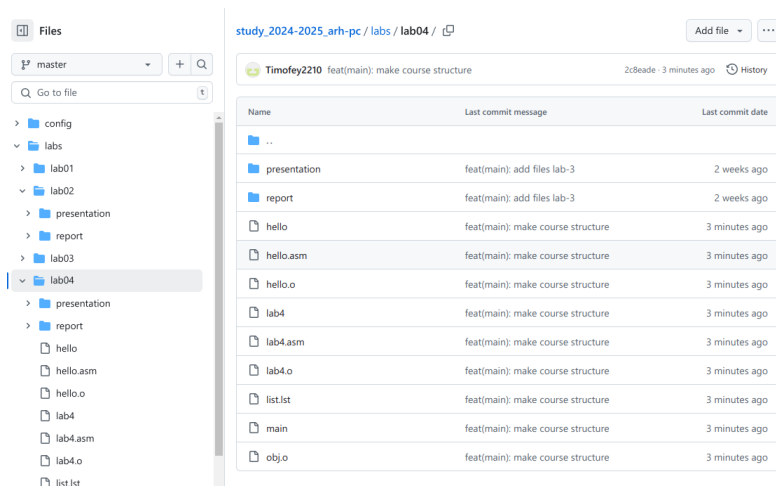


Рис. 14: Проверка файлов

5 Выводы

При выполнении данной лабораторной работы я освоил процедуры компиляции и сборки программ, написанных на ассемблере NASM.