

Отчёт по лабораторной работе №4

Дисциплина: Архитектура компьютера

Кижваткина Анна Юрьевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	14
	Список литературы	15

Список иллюстраций

4.1	Создание каталога	8
4.2	Перемещение в каталог	8
4.3	Создание текстового файла	8
4.4	Проверка создания файлов	9
4.5	Открытие файла с помощью gedit	9
4.6	Редактирование файла	9
4.7	Вводим команду <code>nasm -f hello.asm</code>	10
4.8	Проверка правильности выполнения команды	10
4.9	Команды <code>nasm -o obj.o -f elf -g -l list.lst hello.asm</code> и <code>ls</code>	10
4.10	Обработка файла при помощи компоновщика LD	10
4.11	Команды <code>ld -m elf_i386 obj.o -o main</code> и <code>ls</code>	11
4.12	Запуск файла	11
4.13	Копирование файла	11
4.14	Команда <code>ls</code>	11
4.15	Редактирование текста	11
4.16	Компиляция текста и команда <code>ls</code>	12
4.17	Компоновка файла	12
4.18	Запуск получившегося файла	12
4.19	Копирование получившихся файлов в <code>~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/</code>	12
4.20	Команда <code>ls</code>	12
4.21	Команды <code>git add</code> и <code>git commit</code>	13
4.22	Команда <code>git push</code>	13

Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . . .	7
-----	---	---

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. Транслятор NASM.
3. Работа с расширенным синтаксисом командной строки NASM.
4. Работа с компоновщиком LD.
5. Запуск исполняемого файла.
6. Выполнение задания для самостоятельной работы.

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую систему
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [1–4].

4 Выполнение лабораторной работы

4.1. Создание программы Hello world!

Создаем каталог для работы с программами на языке ассемблера NASM. (рис. 4.1)

```
aykizhvatkina@dk3n35 ~ $ mkdir -p ~/work/arch-pc/lab04
```

Рис. 4.1: Создание каталога

Переходим в созданный каталог. (рис. 4.2).

```
aykizhvatkina@dk3n35 ~ $ cd ~/work/arch-pc/lab04  
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $
```

Рис. 4.2: Перемещение в каталог

Создаем текстовый файл с именем hello.asm. (рис. 4.3)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ touch hello.asm
```

Рис. 4.3: Создание текстового файла

Проверяем созданся ли файл. (рис. 4.4)

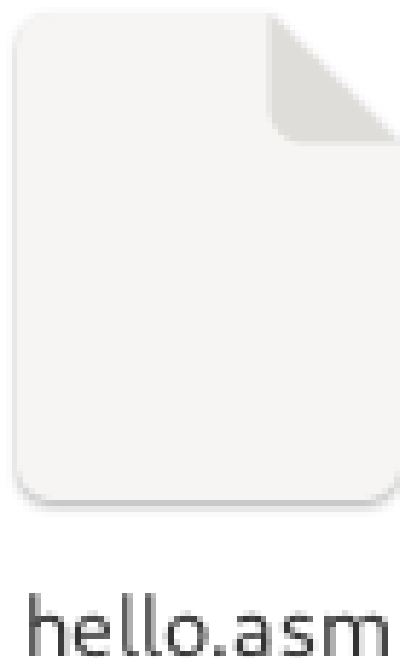


Рис. 4.4: Проверка создания файлов

Открываем файл с помощью текстового редактора gedit. (рис. 4.5)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 4.5: Открытие файла с помощью gedit

Вводим необходимый текст.(рис. 4.6)

```
*hello.asm
~/work/arch-pc/lab04
Сохранить

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
7 SECTION .text ; Начало секции кода
8     GLOBAL _start
9
10 _start: ; Точка входа в программу
11     mov eax,4 ; Системный вызов для записи (sys_write)
12     mov ebx,1 ; Описатель файла '1' - стандартный вывод
13     mov ecx,hello ; Адрес строки hello в ecx
14     mov edx,helloLen ; Размер строки hello
15     int 80h ; Вызов ядра
16
17     mov eax,1 ; Системный вызов для выхода (sys_exit)
18     mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
19     int 80h ; Вызов ядра
```

Рис. 4.6: Редактирование файла

4.2. Транслятор NASM.

Превращаем текст программы в объектный код с помощью транслятора NASM, используя команду `nasm -f elf hello.asm`. (рис. 4.7)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
```

Рис. 4.7: Вводим команду `nasm -f elf hello.asm`

С помощью команды `ls` проверяем правильность выполнения команды. (рис. 4.8)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o
```

Рис. 4.8: Проверка правильности выполнения команды

4.3. Работа с расширенным синтаксисом командной строки NASM.

Вводим команду `nasm -o obj.o -f elf -g -l list.lst hello.asm`, которая компилирует исходный файл `hello.asm` в `obj.o`. Также будет создан файл листинга `list.lst`. С помощью команды `ls` проверяем правильность выполнения команды. (рис. 4.9).

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
```

Рис. 4.9: Команды `nasm -o obj.o -f elf -g -l list.lst hello.asm` и `ls`

4.4. Работа с компоновщиком LD.

Обрабатываем файл `hello.o` при помощи компоновщика LD, чтобы получить исходный файл `hello`. С помощью `ls` проверяем правильность выполнения команды. (рис. 4.10)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst obj.o
```

Рис. 4.10: Обработка файла при помощи компоновщика LD

Выполняем следующую команду `ld -m elf_i386 obj.o -o main` и проверяем правильность выполнения команды. (рис. 4.11)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
```

Рис. 4.11: Команды `ld -m elf_i386 obj.o -o main` и `ls`

4.5. Запуск исполняемого файла.

Запускаем на выполнение созданный исполняемый файл `hello`. (рис. 4.12)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

Рис. 4.12: Запуск файла

4.6. Выполнение задания для самостоятельной работы.

В каталоге `~/work/arch-pc/lab04` с помощью команды `cp` создаем копию файла `hello.asm` с именем `lab4.asm`. (рис. 4.13)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
```

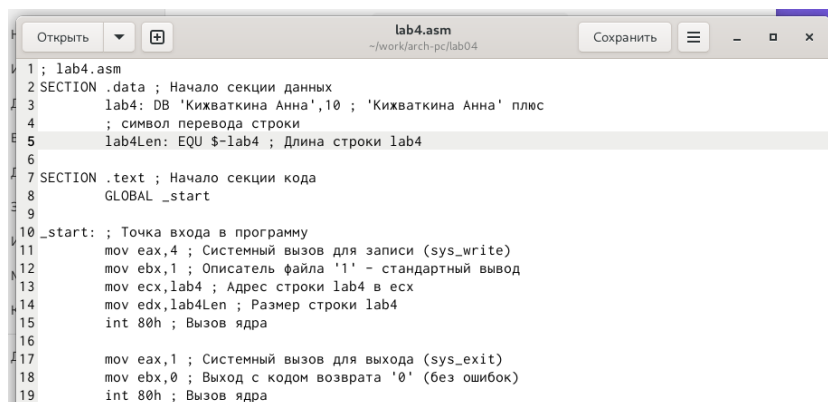
Рис. 4.13: Копирование файла

Проверяем правильность выполнения команды с помощью `ls`. (рис. 4.14)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
```

Рис. 4.14: Команда `ls`

С помощью текстового редактора вносим изменения в текст программы в файле `lab4` так, чтобы вместо `hello world!` на экран выводилось `Кижваткина Анна`. (рис. 4.15)



```
1; lab4.asm
2 SECTION .data ; Начало секции данных
3     lab4: DB 'Кижваткина Анна',10 ; 'Кижваткина Анна' плюс
4     ; символ перевода строки
5     lab4Len: EQU $-lab4 ; Длина строки lab4
6
7 SECTION .text ; Начало секции кода
8     GLOBAL _start
9
10 _start: ; Точка входа в программу
11     mov eax,4 ; Системный вызов для записи (sys_write)
12     mov ebx,1 ; Описатель файла '1' - стандартный вывод
13     mov ecx,lab4 ; Адрес строки lab4 в ecx
14     mov edx,lab4Len ; Размер строки lab4
15     int 80h ; Вызов ядра
16
17     mov eax,1 ; Системный вызов для выхода (sys_exit)
18     mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
19     int 80h ; Вызов ядра
```

Рис. 4.15: Редактирование текста

Компилируем текст программы в объектный файл. Проверяем правильность выполнения с помощью ls. (рис. 4.16)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  lab4.o  list.lst  main  obj.o
```

Рис. 4.16: Компиляция текста и команда ls

Выполняем компоновку объектного файла. (рис. 4.17)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab4.o -o lab4
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.o  list.lst  main  obj.o
```

Рис. 4.17: Компоновка файла

Запускаем получившийся исполняемый файл. (рис. 4.18)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ ./lab4
Кижваткина Анна
```

Рис. 4.18: Запуск получившегося файла

Копируем файлы hello.asm и lab4.asm в Ваш локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/. (рис. 4.19)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ cp hello.asm ~/work/study/2023-2024/“Архитектура компьютера”/study_2023-2024_arh--pc/labs/lab04/
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ cp lab4.asm ~/work/study/2023-2024/“Архитектура компьютера”/study_2023-2024_arh--pc/labs/lab04/
```

Рис. 4.19: Копирование получившихся файлов в ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/

Проверяем правильность выполнения с помощью ls. (рис. 4.20)

```
aykizhvatkina@dk3n35 ~/work/arch-pc/lab04 $ cd
aykizhvatkina@dk3n35 ~ $ cd ~/work/study/2023-2024/“Архитектура компьютера”/study_2023-2024_arh--pc/labs/lab04/
aykizhvatkina@dk3n35 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh--pc/labs/lab04 $ ls
hello.asm  lab4.asm  presentation  report
```

Рис. 4.20: Команда ls

Загружаем файлы на Github при помощи команд git add, git commit и git push. (рис. 4.21 и рис. 4.22)

```
aykizhvatkina@dk3n35 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh--pc/labs/lab04 $ git add .
aykizhvatkina@dk3n35 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh--pc/labs/lab04 $ git commit -m "add files"
[master 43442ae] add files
24 files changed, 39 insertions(+), 1 deletion(-)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab04/report/image/1.png
create mode 100644 labs/lab04/report/image/10.png
```

Рис. 4.21: Команды git add и git commit

```
aykizhvatkina@dk3n35 ~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh--pc/labs/lab04 $ git push
Перечисление объектов: 36, готово.
Подсчет объектов: 100% (36/36), готово.
При скатин изменения используется до 4 потоков
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (30/30), 891.52 Киб | 8.10 Миб/с, готово.
Total 30 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To github.com:Anyakizh/study_2023-2024_arh--pc.git
073e630..43442ae master -> master
```

Рис. 4.22: Команда git push

5 Выводы

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

Список литературы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.
2. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.