

Лабораторная работа №4

Архитектура вычислительных систем

Зарифбеков Амир Пайшанбиевич

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Выводы	10

Список иллюстраций

3.1	пример простой программы на языке ассемблера	6
3.2	переход в каталог	6
3.3	создание текстового файла	6
3.4	открытие файла с помощью текстового редактора	7
3.5	вводим текст	7
3.6	напишем текст программы	7
3.7	проверяем созданся ли файл	7
3.8	выполнение команды и проверка её создания	7
3.9	передача объектного файла компоновщику и проверка его передачи	8
3.10	выполняем команду	8
3.11	видим формат командной строки	8
3.12	Запустим на выполнение созданный исполняемый файл	9
3.13	создадим копию файла hello.asm с именем lab5.asm	9
3.14	выводится наша фамилия и имя	9
3.15	Оттранслируем полученный текст программы	9

1 Цель работы

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM

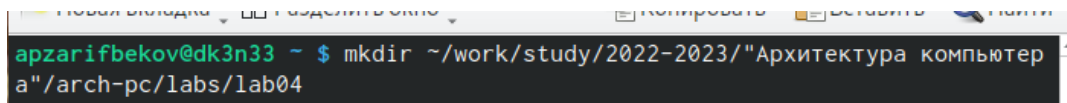
2 Задание

1. В каталоге `~/work/arch-рс/lab05` с помощью команды `ср` создайте копию файла `hello.asm` с именем `lab5.asm`
2. С помощью любого текстового редактора внесите изменения в текст программы в файле `lab5.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с вашими фамилией и именем.
3. Оттранслируйте полученный текст программы `lab5.asm` в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.
4. Скопируйте файлы `hello.asm` и `lab5.asm` в Ваш локальный репозиторий в каталог `~/work/study/2022-2023/“Архитектура компьютера”/arch-рс/labs/lab05/`. Загрузите файлы на Github.

3 Выполнение лабораторной работы

4.3.1. Программа Hello world!

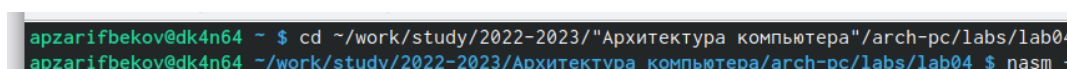
1. рассмотрим пример прослой программы на языке ассемблера NASM. Создадим каталог для работы с программами на языке ассемблера NASM. Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 3.1)



```
apzarifbekov@dk3n33 ~ $ mkdir ~/work/study/2022-2023/"Архитектура компьютера"
a"/arch-pc/labs/lab04
```

Рис. 3.1: пример простой программы на языке ассемблера

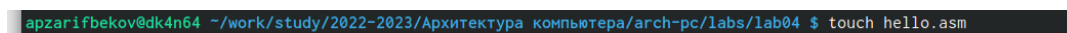
2. Перейдём в созданный каталог



```
apzarifbekov@dk4n64 ~ $ cd ~/work/study/2022-2023/"Архитектура компьютера"/arch-pc/labs/lab04
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f
```

Рис. 3.2: переход в каталог

3. Создадим текстовый файл с именем hello.asm



```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ touch hello.asm
```

Рис. 3.3: создание текстового файла

4. Откроем этот файл с помощью любого текстового редактора, например gedit

```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ gedit hello.asm
^C
```

Рис. 3.4: открытие файла с помощью текстового редактора

и введём в него следующий текст:

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

Рис. 3.5: вводим текст

4.3.2. Транслятор NASM 1. Напишем , необходимый для компеляции приведённого выше текст программы “Hello World”

```
^C
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf hello.asm
```

Рис. 3.6: напишем текст программы

2. С помощью ls проверим , что объектный файл был создан

```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm hello.o presentation report
```

Рис. 3.7: проверяем создался ли файл

4.3.3. Расширенный синтаксис командной строки NASM

1. Выпнлим следующую команду и спомощью ls проверим был ли создан файл

```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm hello.o list.lst obj.o presentation report
```

Рис. 3.8: выполнение команды и проверка её создания

4.4. Компоновщик JD

1. Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику с помощью ls проверим что был создан файл hello

```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386
hello.o -o hello
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o  presentation  report
```

Рис. 3.9: передача объектного файла компоновщику и проверка его передачи

2. Выполним следующую команду :

```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386
obj.o -o main
```

Рис. 3.10: выполняем команду

3. Формат командной строки LD можно увидеть набрав ld-help:

```
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ld - -help
Использование ld [параметры] файл...
Параметры:
  -a КЛЮЧЕВОЕ СЛОВО                Управление общей библиотекой для совместимости с HP/UX
  -A АРХИТЕКТУРА, --architecture АРХИТЕКТУРА  Задать архитектуру
  -b ЦЕЛЬ, --format ЦЕЛЬ            Задать цель для следующих входных файлов
  -c ФАЙЛ, --mri-script ФАЙЛ        Прочитать сценарий компоновщика в формате MRI
  -d, -dc, -dp                      Принудительно делать общие символы определёнными
  --dependency-file ФАЙЛ             Write dependency file
  --force-group-allocation           Принудительно удалить членов группы из групп
  -e АДРЕС, --entry АДРЕС           Задать начальный адрес
  -E, --export-dynamic              Экспортировать все динамические символы
  --no-export-dynamic               Отменить действие --export-dynamic
  --enable-non-contiguous-regions    Enable support of non-contiguous memory regions
```

Рис. 3.11: видим формат командной строки

4.4.1. Запуск исполняемого файла

1. Запустим на выполнение созданный исполняемый файл, находящийся в текущем каталоге, набрав в командной строке :


```

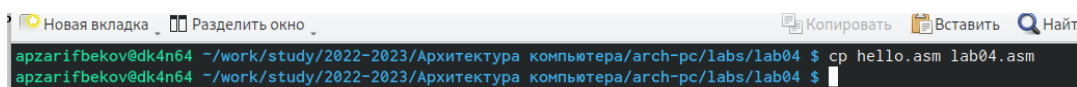
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ./hello
Hello world!
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $

```

Рис. 3.12: Запустим на выполнение созданный исполняемый файл

4.5 Задание для самостоятельной работы

1. В каталоге ~/work/arch-pc/lab05 с помощью команды `cp` создайте копию файла `hello.asm` с именем `lab5.asm`



```

Новая вкладка Разделить окно Копировать Вставить Найти
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ cp hello.asm lab04.asm
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $

```

Рис. 3.13: создадим копию файла `hello.asm` с именем `lab5.asm`

2. С помощью любого текстового редактора внесите изменения в текст программы в файле `lab5.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с вашими фамилией и именем.

```

apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ nasm -f elf -g -l list.lst lab04.asm
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ld -m elf_i386 lab04.o -o lab04
apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $

```

Рис. 3.14: выводится наша фамилия и имя

3. Оттранслируйте полученный текст программы `lab5.asm` в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.

```

apzarifbekov@dk4n64 ~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab04 $ ./lab04
Zarifbekov Amir

```

Рис. 3.15: Оттранслируем полученный текст программы

4 Выводы

Освоил процедуру компиляции и сборки программ, написанных на ассемблере NASM.