

Отчёт по лабораторной работе №4

Дисциплина: Архитектура компьютеров

Лазарев Даниил Михайлович

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
5	Выполнение самостоятельной работы	10
6	Выводы	12

Список иллюстраций

4.1	Создание каталога для работы с ассемблером	7
4.2	Создание файла “hello.asm”	7
4.3	Вставленный текст в редакторе	8
4.4	Компиляция программы	8
4.5	Компиляция с помощью полной команды	8
4.6	Компановка программы	9
4.7	Задаем имя исполняемого файла	9
4.8	Выполнение исполняемого файла	9
5.1	Создание копии файла	10
5.2	Вывод измененного файла	10
5.3	Файлы на Github	11

1 Цель работы

Целью работы является освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Задание

1. Выполнение лабораторной работы №4
2. Заполнение отчета по выполнению лабораторной работы №4 с помощью языка разметки Markdown
3. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язык низкого уровня. Можно считать, что он больше любых других языков приближен к архитектуре ЭВМ и её аппаратным возможностям, что позволяет получить к ним более полный доступ, нежели в языках высокого уровня, таких как C/C++, Perl, Python и пр.

4 Выполнение лабораторной работы

Открываем терминал, создаем папку “lab04” в папке “work” для работы с программами на языке ассемблера NASM и переходим в нее. (рис. 4.1)

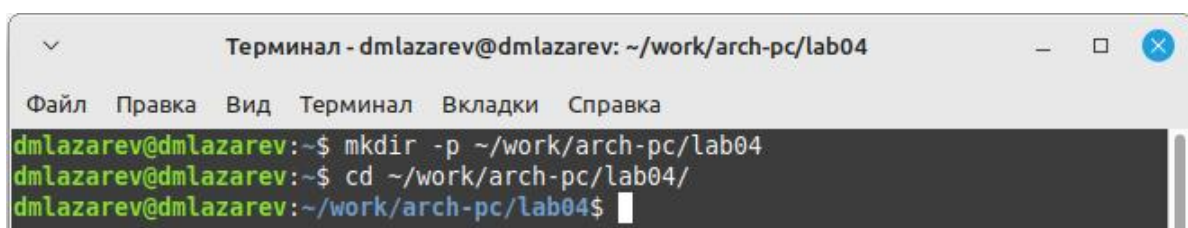


Рис. 4.1: Создание каталога для работы с ассемблером

После того, как перешли в созданный нами каталог, создадим файл “hello.asm” и откроем его с помощью текстового редактора “gedit” (рис. 4.2)

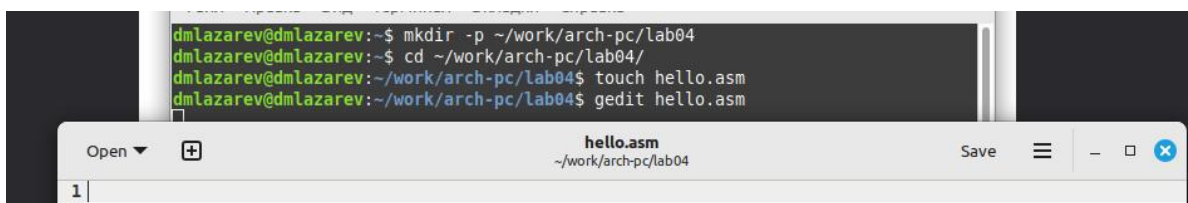


Рис. 4.2: Создание файла “hello.asm”

После того, как файл был открыт с помощью текстового редактора, вставим в него заранее заготовленный текст на языке ассемблера. (рис. 4.3)

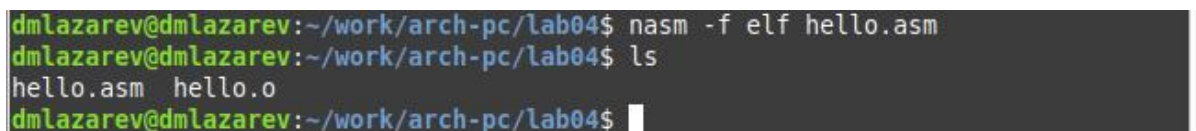


```
1 |;hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5 helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для 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. 4.3: Вставленный текст в редакторе

В отличие от многих современных высокоуровневых языков программирования, в ассемблерной программе каждая команда располагается на отдельной строке. Так же синтаксис чувствителен к регистру.

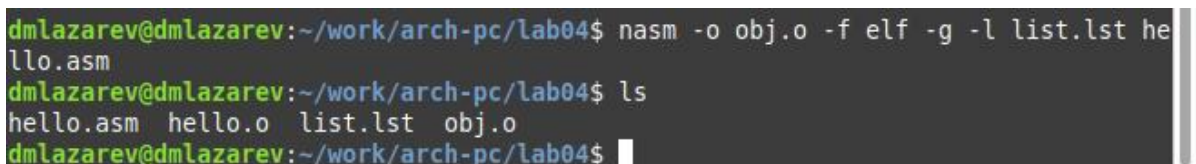
С помощью команды “nasm” скомпилируем текст программы “Hello world”. Текст был вставлен без ошибок, поэтому после, при проверке через “ls”, у нас появился необходимый файл - “hello.o” (рис. 4.4)



```
dmLazarev@dmLazarev:~/work/arch-pc/lab04$ nasm -f elf hello.asm
dmLazarev@dmLazarev:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
dmLazarev@dmLazarev:~/work/arch-pc/lab04$
```

Рис. 4.4: Компиляция программы

После введем полный вариант команды “nasm”, который скомпилирует нашу программу в файл “obj.o”. После выполнения проверим правильность выполнения с помощью команды “ls”. (рис. 4.5)



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

Рис. 4.5: Компиляция с помощью полной команды

Теперь передаем исполняемую программу на обработку компоновщику, используя команду “ld”. Проверив выполнение с помощью команды “ls” необходимый нам исполняемый файл “hello”. (рис. 4.6)

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

Рис. 4.6: Компоновка программы

Опять используем компоновщика, чтобы задать имя создаваемого исполняемого файла. Проверяем правильность выполнения. (рис. 4.7)

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

Рис. 4.7: Задаем имя исполняемого файла

Выполним созданный нами исполняемый файл с помощью команды “./hello” (рис. 4.8)

```
dmlazarev@dmlazarev:~/work/arch-pc/lab04$ ./hello
Hello world!
dmlazarev@dmlazarev:~/work/arch-pc/lab04$
```

Рис. 4.8: Выполнение исполняемого файла

5 Выполнение самостоятельной работы

1. Переходим в каталог `~/work/arch-pc/lab04` с помощью `cd`, после с помощью команды `cp` создаем копию файла `hello.asm` с именем `lab4.asm` (рис. 5.1)

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

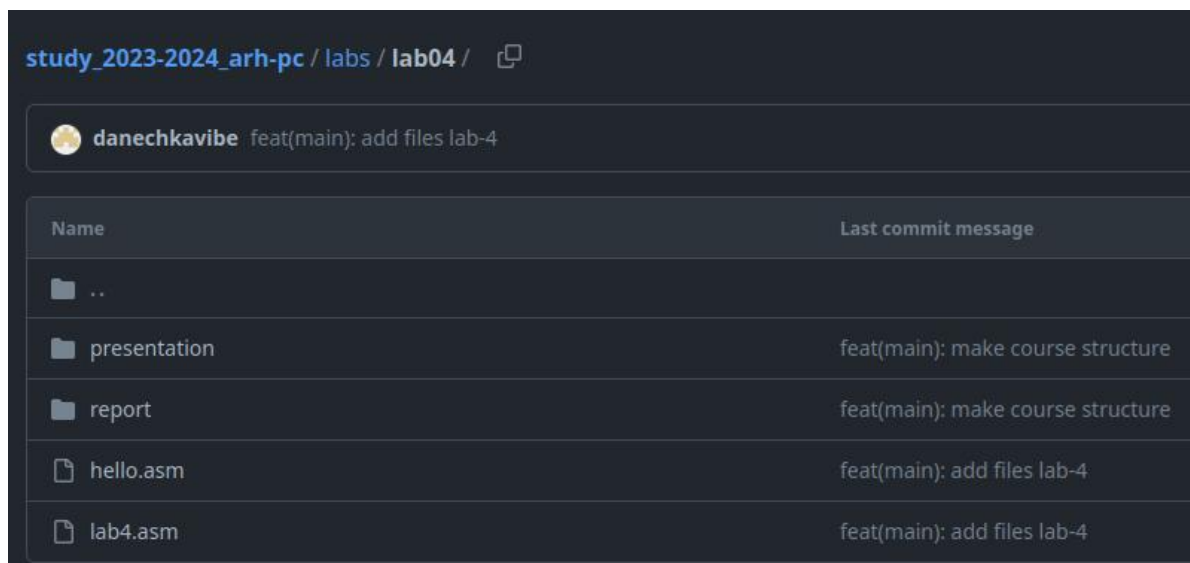
Рис. 5.1: Создание копии файла

2. С помощью текстового редактора `gedit` внесем изменения в текст программы так, чтобы вместо фразы `hello world` выводились мои имя и фамилия. После конвертируем полученный текст в файле `lab4.asm` в объектный файл, выполним компоновку и запустим получившийся исполняемый файл (рис. 5.2)

```
dmlazarev@dmlazarev:~/work/arch-pc/lab04$ ls
hello      hello.o  lab4.asm  list4.lst  main      obj.o
hello.asm  lab4     lab4.o    list.lst   main4
dmlazarev@dmlazarev:~/work/arch-pc/lab04$ ./lab4
Лазарев Даниил
dmlazarev@dmlazarev:~/work/arch-pc/lab04$
```

Рис. 5.2: Вывод измененного файла

3. Скопируем все файлы в локальный репозиторий и выгрузим на Github (рис. 5.3)



The screenshot shows a GitHub repository interface. At the top, the breadcrumb path is 'study_2023-2024_arh-pc / labs / lab04'. Below this, a commit by user 'danechkavibe' is shown with the message 'feat(main): add files lab-4'. A table lists the files and folders in the repository, along with the commit message for each.

Name	Last commit message
..	
presentation	feat(main): make course structure
report	feat(main): make course structure
hello.asm	feat(main): add files lab-4
lab4.asm	feat(main): add files lab-4

Рис. 5.3: Файлы на Github

6 Выводы

В ходе лабораторной работы мы освоили процедуры компиляции и сборки программ на языке ассемблера.