

Отчёта по лабораторной работе №4

Дисциплина: Архитектура компьютера

Камалиева Лия Дамировна.

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
4.1	1.1 Программа Hello world!	7
4.2	1.2 Транслятор NASM	8
4.3	1.3 Расширенный синтаксис командной строки NASM	8
4.4	1.4 компоновщик LD	9
4.5	1.5 Запуск исполняемого файла	10
4.6	1.6 Задание для самостоятельной работы	10
	4.6.1 №1	10
	4.6.2 №2	10
	4.6.3 №3	11
	4.6.4 №4	11
5	Выводы	14

Список иллюстраций

4.1	lab04 рис.4.1	7
4.2	создание файла hello.asm рис.4.2	7
4.3	открываем в mousepad рис.4.3	7
4.4	программа hello world! рис.4.4	8
4.5	компилируем код рис.4.5	8
4.6	команда ls рис.4.6	8
4.7	компиляция и проверка рис.4.7	9
4.8	передача объектного файла рис.4.8	9
4.9	ls рис.4.9	9
4.10	создание файла main рис.4.10	9
4.11	запускаем на выполнение файл рис.4.11	10
4.12	создание lab4.asm рис.4.12	10
4.13	проверка рис.4.13	10
4.14	замена рис.4.14	11
4.15	настройка программы рис.4.15	11
4.16	репозиторий рис.4.16	12
4.17	github рис.4.17	13

1 Цель работы

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

2 Задание

1.1 Программа Hello world! 1.2 Транслятор NASM 1.3 Расширенный синтаксис командной строки NASM 1.4 компоновщик LD 1.5 Запуск исполняемого файла 1.6 Задание для самостоятельной работы

3 Теоретическое введение

NASM (Netwide Assembler) — свободный (LGPL и лицензия BSD) ассемблер для архитектуры Intel x86. Используется для написания 16-, 32- и 64-разрядных программ.

4 Выполнение лабораторной работы

4.1 1.1 Программа Hello world!

Шаг 1. Создаем новый каталог lab04

```
ldkamalieva@ldkamalieva-VirtualBox:~$ mkdir -p ~/work/arch-pc/lab04
```

Рис. 4.1: lab04 рис.4.1

Шаг 2. При помощи команды touch создаем файл hello.asm

```
ldkamalieva@ldkamalieva-VirtualBox:~$ cd ~/work/arch-pc/lab04  
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ touch hello.asm
```

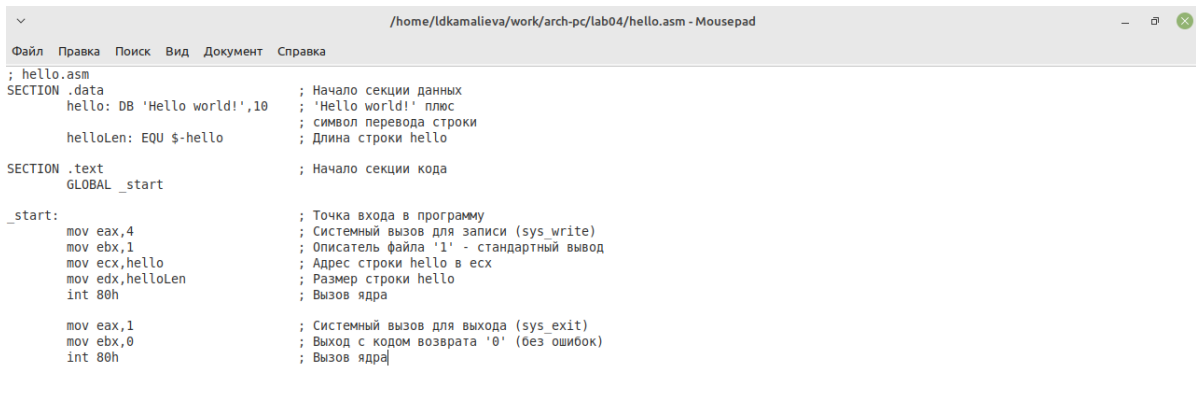
Рис. 4.2: создание файла hello.asm рис.4.2

Шаг 3. Открываем его в текстовом редакторе

```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ mousepad hello.asm
```

Рис. 4.3: открываем в mousepad рис.4.3

Шаг 4. Вводим в mousepad программу hello world!



```
File  Правка  Поиск  Вид  Документ  Справка
; hello.asm
SECTION .data
hello: DB 'Hello world!',10      ; Начало секции данных
                                   ; 'Hello world!' плюс
                                   ; символ перевода строки
helloLen: EQU $-hello           ; Длина строки hello

SECTION .text
GLOBAL _start                   ; Начало секции кода

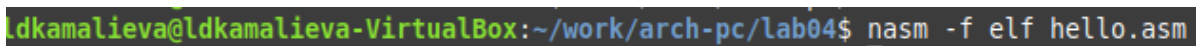
_start:                          ; Точка входа в программу
    mov eax,4                   ; Системный вызов для записи (sys_write)
    mov ebx,1                   ; Описатель файла '1' - стандартный вывод
    mov ecx,hello               ; Адрес строки hello в ecx
    mov edx,helloLen            ; Размер строки hello
    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. 4.4: программа hello world! рис.4.4

4.2 1.2 Транслятор NASM

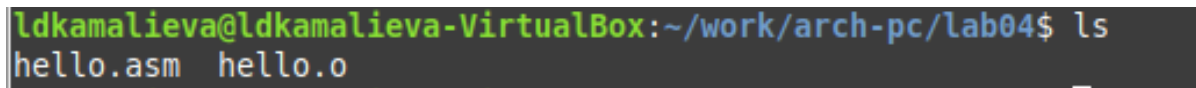
Шаг 1. Компилируем наш код при помощи команды `nasm -f elf hello.asm`



```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ nasm -f elf hello.asm
```

Рис. 4.5: компилируем код рис.4.5

Шаг 2. Сделаем проверку



```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
```

Рис. 4.6: команда ls рис.4.6

4.3 1.3 Расширенный синтаксис командной строки NASM

Шаг 1. скомпилируем наш файл `hello.asm` в `obj.o` при помощи команды `nasm -o obj.o -f elf -g -l list.lst hello.asm` и сделаем проверку с помощью команды `ls`


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

Рис. 4.7: компиляция и проверка рис.4.7

4.4 1.4 Компоновщик LD

Шаг 1. Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику при помощи команды `ld -m elf_i386 hello.o -o hello`

```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o
-o hello
```

Рис. 4.8: передача объектного файла рис.4.8

Шаг 2. Проверяем созданся ли файл `hello`

```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис. 4.9: ls рис.4.9

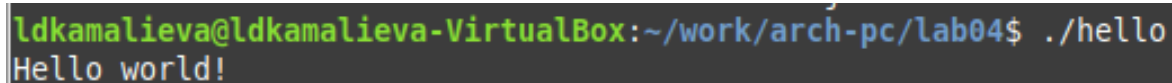
Шаг 3. Далее вводим команду `ld -m elf_i386 obj.o -o main` и создаем файл `main`, который также проверяем с помощью команды `ls`

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

Рис. 4.10: создание файла `main` рис.4.10

4.5 1.5 Запуск исполняемого файла

Шаг 1. Запускаем на выполнение созданный файл, находящийся в текущем каталоге, при помощи команды `./hello`



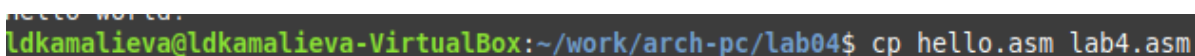
```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ ./hello
Hello world!
```

Рис. 4.11: запускаем на выполнение файл рис.4.11

4.6 1.6 Задание для самостоятельной работы

4.6.1 №1

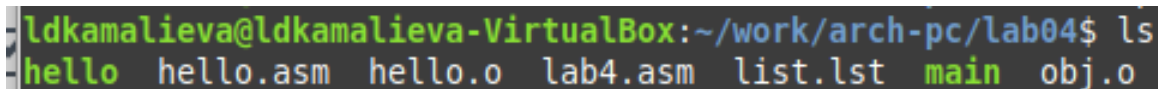
Шаг 1. В каталоге `~/work/arch-pc/lab04` с помощью команды `cp` создаем копию файла `hello.asm` с именем `lab4.asm`



```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
```

Рис. 4.12: создание lab4.asm рис.4.12

Шаг 2. Делаем проверку

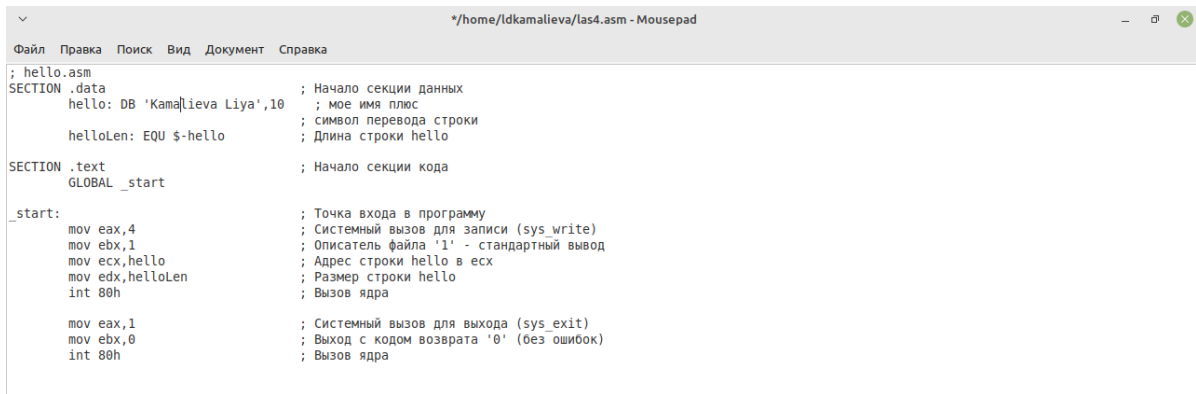


```
ldkamalieva@ldkamalieva-VirtualBox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
```

Рис. 4.13: проверка рис.4.13

4.6.2 №2

Шаг 1. С помощью текстового редактора в программе `Hello world!` заменяем на `Kamaliev Liya`



```

; hello.asm
SECTION .data
    hello: DB 'Kamaliev Liya',10
            ; Начало секции данных
            ; мое имя плюс
            ; символ перевода строки
            helloLen: EQU $-hello
            ; Длина строки hello
SECTION .text
    GLOBAL _start
            ; Начало секции кода
_start:
            ; Точка входа в программу
            mov eax,4
            ; Системный вызов для записи (sys_write)
            mov ebx,1
            ; Описатель файла '1' - стандартный вывод
            mov ecx,hello
            ; Адрес строки hello в ecx
            mov edx,helloLen
            ; Размер строки hello
            int 80h
            ; Вызов ядра

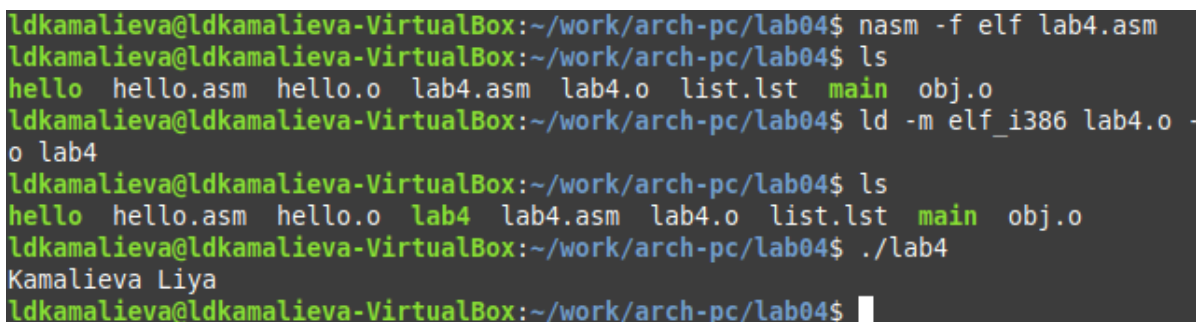
            mov eax,1
            ; Системный вызов для выхода (sys_exit)
            mov ebx,0
            ; Выход с кодом возврата '0' (без ошибок)
            int 80h
            ; Вызов ядра

```

Рис. 4.14: замена рис.4.14

4.6.3 №3

Шаг 1. Делаем все действия, что, и с программой Hello world!, также делаем проверку



```

ldkamaliev@ldkamaliev-VirtualBox:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
ldkamaliev@ldkamaliev-VirtualBox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4.asm lab4.o list.lst main obj.o
ldkamaliev@ldkamaliev-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
ldkamaliev@ldkamaliev-VirtualBox:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o
ldkamaliev@ldkamaliev-VirtualBox:~/work/arch-pc/lab04$ ./lab4
Kamaliev Liya
ldkamaliev@ldkamaliev-VirtualBox:~/work/arch-pc/lab04$

```

Рис. 4.15: настройка программы рис.4.15

4.6.4 №4

Шаг 1. Скопируем файлы hello.asm и lab4.asm в локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04. Загрузим файлы на Github

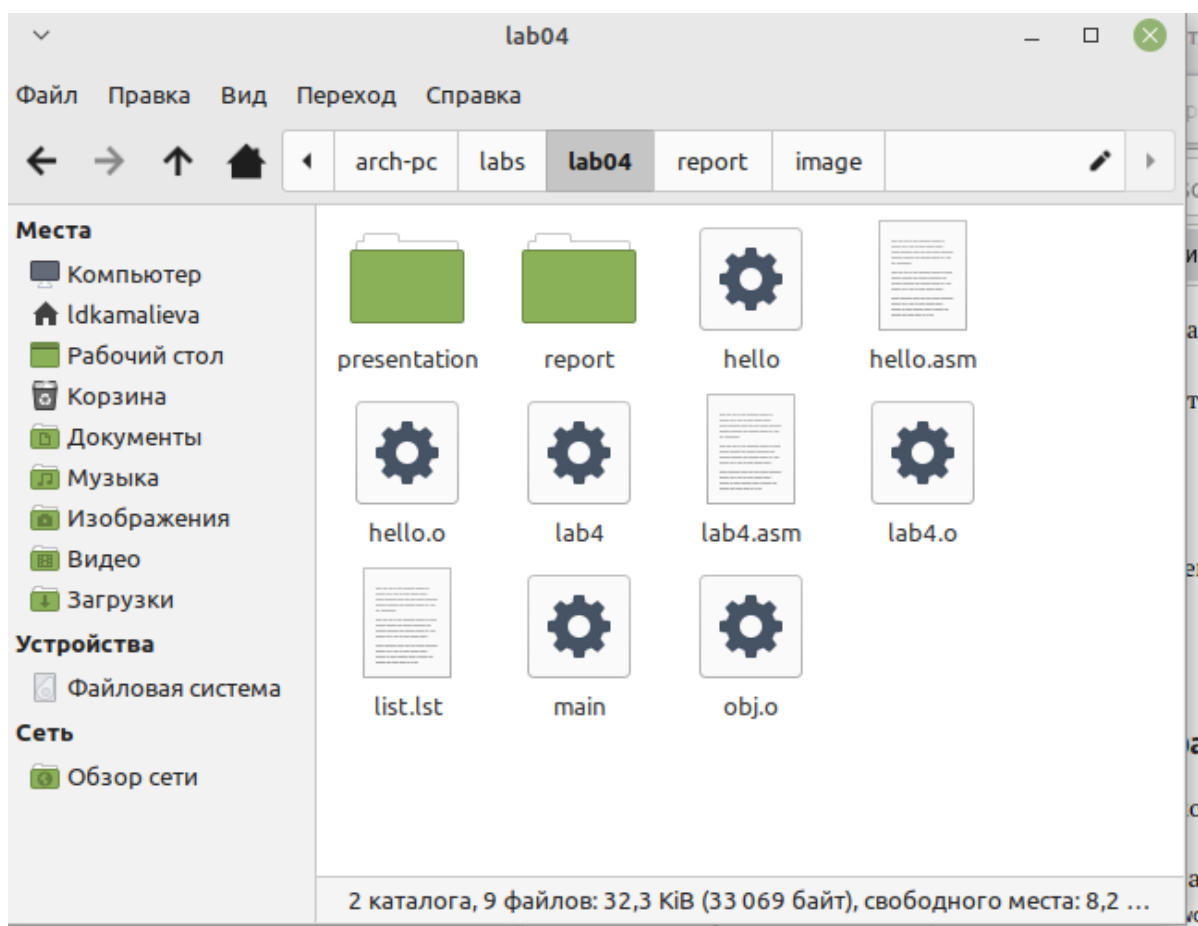


Рис. 4.16: репозиторий рис.4.16

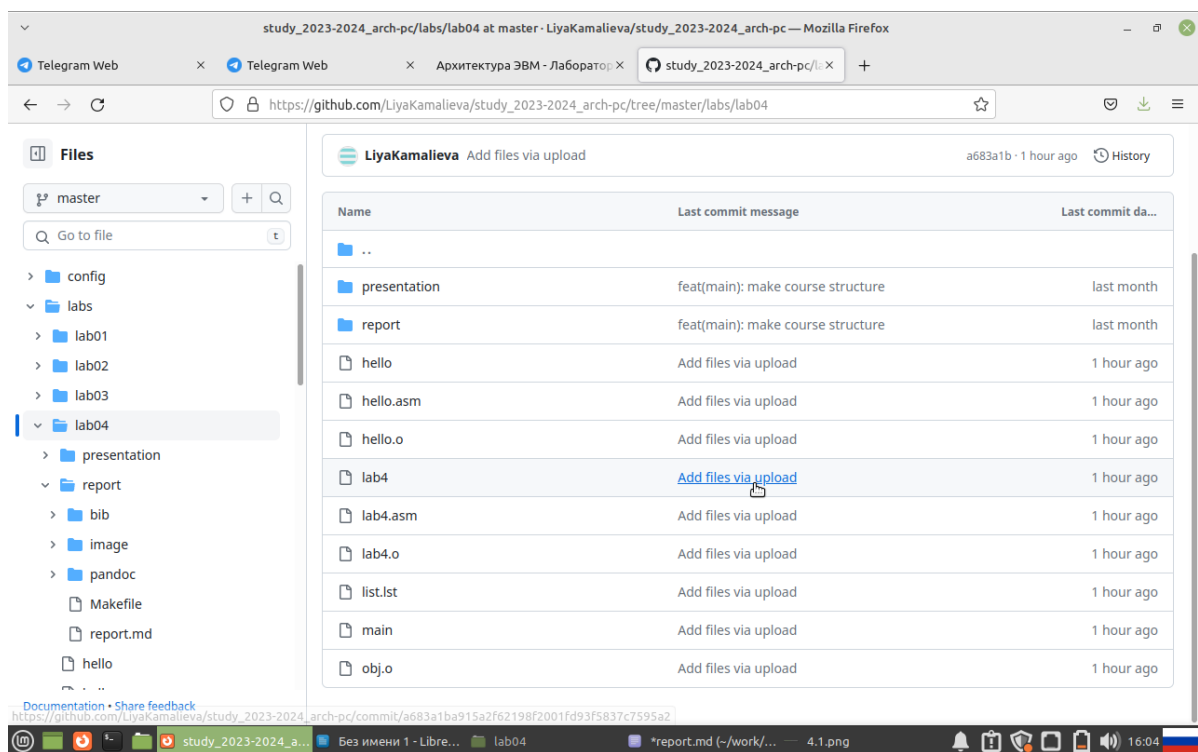


Рис. 4.17: github рис.4.17

5 Выводы

Мы изучили основные команды для работы с языком NASM.