

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

дисциплина: Архитектура компьютера

Студент: Армихос Гонзалез Карла

Группа: НКАбд-02-24

МОСКВА

2024 г.

Содержание

1. Цель работы.....	4
2. Задание.....	5
3. Технические введение.....	6
4. Порядок выполнения лабораторной работы.....	8
4.1 Программа Hello world!.....	8
4.2 Транслятор NASM.....	8
4.3 Расширенный синтаксис командной строки NASM.....	9
4.4 компоновщик LD.....	9
4.5 Запуск исполняемого файла.....	10
5. Задание для самостоятельной работы.....	11
5.1 Создать копия.....	11
6. Выводы.....	13
7. Список используемой литературы.....	14

Список иллюстраций

рис. 4.1.1 новый каталог.....	7
рис. 4.1.2 Создание .asm файла.....	7
рис. 4.1.3.....	8
рис. 4.2.1 транслятор nasm.....	8
рис. 4.3.1.....	8
рис. 4.4.1.....	8
рис. 4.4. ключ.....	9
рис. 4.5.1.....	9
рис. 5.1 Копия.....	10
рис. 5.2 Редактировать текст.....	11
рис. 5.3 Запустить программу.....	11

1. Цель работы

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

2. Задание

1. Создание программы Hello world!
2. Основные концепты работы с файловым менеджером "mc".
3. Структура приложения на языке ассемблера NASM.
4. Процесс подключения внешнего файла.
5. Выполнение заданий для самостоятельной работы.

3. Технические введение

Основные отличия ассемблерных программ от программ на языках высокого уровня заключаются в уровне абстракции и сложности. Ассемблерные программы пишутся для конкретного процессора и оперируют напрямую его инструкциями, что позволяет максимальный контроль над аппаратной частью. Программы на языках высокого уровня, таких как C или Python, скрывают аппаратные детали, предлагая более понятный и читаемый код, но уступают в скорости и контроле.

Инструкция на языке ассемблера — это команда, которую процессор выполняет непосредственно, например, операция сложения или перемещения данных. Директива — это команда, предназначенная для ассемблера, помогающая ему организовать код, но которая не превращается в машинный код, как, например, определение секций данных. (рис.1)

Основные правила оформления программ на ассемблере включают использование комментариев для пояснения кода, правильное отступление и группировку кода в секции данных, текстовую секцию и стек. Это облегчает чтение и поддержку кода.



рис. 1.

Этапы получения исполняемого файла включают написание исходного кода, трансляцию (ассемблирование), компоновку и, наконец, запуск программы. Эти шаги позволяют перевести код из понятного для человека вида в машинный код.

Назначение этапа трансляции — преобразовать ассемблерный код в машинный код. На этом этапе создается объектный файл, содержащий машинные инструкции, которые соответствуют ассемблерным командам.

Назначение этапа компоновки заключается в объединении всех объектных файлов и библиотек в один исполняемый файл, включая установку всех ссылок и адресов.

При трансляции программы могут создаваться различные файлы, такие как объектные файлы (.o) и исполняемые файлы. По умолчанию, в результате трансляции создается объектный файл, а при компоновке — исполняемый файл.

Форматы файлов для `nasm` и `ld` обычно включают объектные файлы в формате ELF (Executable and Linkable Format) на Linux или COFF (Common Object File Format) на Windows.

4. Порядок выполнения лабораторной работы

4.1 Программа Hello world!

Создать новый каталог а затем войдите в созданную папку.(рис. 4.1.1)

```
gkarmikhos@fedora:~$ mkdir -p ~/work/arch-pc/lab04
gkarmikhos@fedora:~$ cd ~/work/arch-pc/lab04
```

рис. 4.1.1 новый каталог.

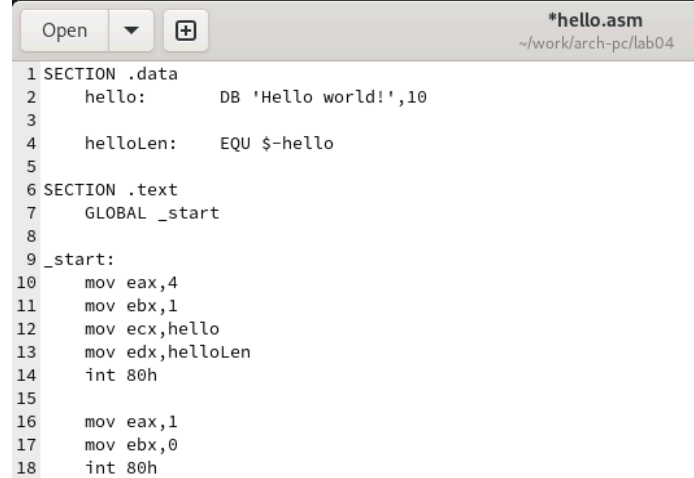
Создайте текстовый файл с именем hello.asm (рис. 4.1.2)

```
gkarmikhos@fedora:~/work/arch-pc/lab04$ touch hello.asm
gkarmikhos@fedora:~/work/arch-pc/lab04$
```

рис. 4.1.2 Создание .asm файла

Открываем файл в текстовом редакторе gedit и пишем программу(рис.4.1.3)

```
gkarmikhos@fedora:~/work/arch-pc/lab04$ gedit hello.asm
```



```
1 SECTION .data
2     hello:      DB 'Hello world!',10
3
4     helloLen:   EQU $-hello
5
6 SECTION .text
7     GLOBAL _start
8
9 _start:
10    mov eax,4
11    mov ebx,1
12    mov ecx,hello
13    mov edx,helloLen
14    int 80h
15
16    mov eax,1
17    mov ebx,0
18    int 80h
```

рис. 4.1.3 файл

4.2 Транслятор NASM

Преобразование текста программы в код, затем для подтверждения вводим команду ls (рис. 4.2.1)


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

рис. 4.2.1 транслятор nasm

4.3 Расширенный синтаксис командной строки NASM

Скомпилируйте файл и убедитесь, что он создан правильно (рис. 4.3.1)

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

рис. 4.3.1

4.4 Компоновщик LD

Связать объектный файл для обработки(рис. 4.4.1)

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

рис. 4.4.1

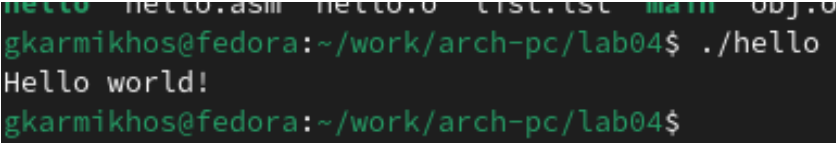
Выполнить ключ (рис. 4.4.2)

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

рис. 4.4. ключ

4.5 Запуск исполняемого файла

Запустить созданный каталог(рис. 4.5.1)

A terminal window with a dark background and green text. The prompt is 'gkarmikhos@fedora:~/work/arch-pc/lab04\$'. The user enters './hello' and the output is 'Hello world!'.

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

рис. 4.5.1

5. Задание для самостоятельной работы

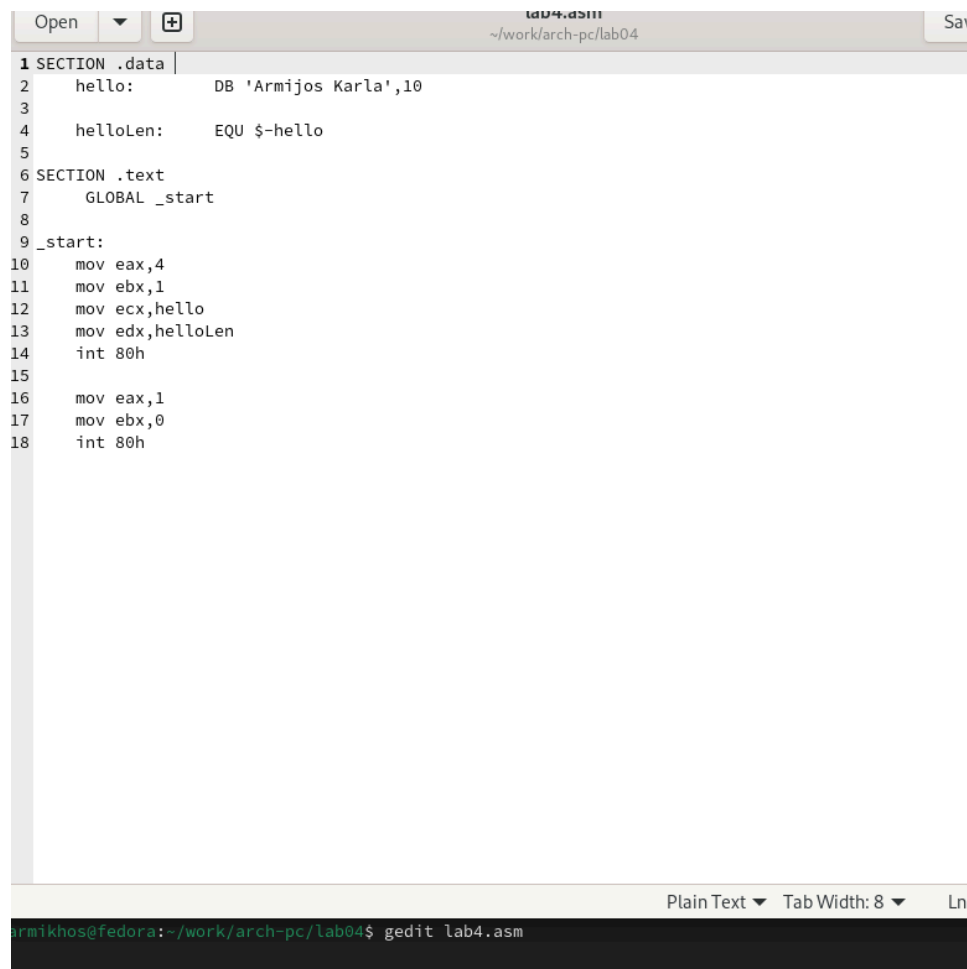
5.1 Создать копия

Создать копию файла с помощью команды `cp`(рис. 5.1)

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

рис. 5.1 Копия

Отредактируйте текст, чтобы при его запуске отображались мои имя и фамилия(рис. 5.2)



```
1 SECTION .data
2     hello:      DB 'Armijos Karla',10
3
4     helloLen:   EQU $-hello
5
6 SECTION .text
7     GLOBAL _start
8
9 _start:
10    mov eax,4
11    mov ebx,1
12    mov ecx,hello
13    mov edx,helloLen
14    int 80h
15
16    mov eax,1
17    mov ebx,0
18    int 80h
```

armikhos@fedora:~/work/arch-pc/lab04\$ gedit lab4.asm

рис. 5.2 Редактировать текст

Воспроизвести отредактированный текст.(рис. 5.3)

```
gkarmikhos@fedora:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
gkarmikhos@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
gkarmikhos@fedora:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab04.asm lab4 lab4.asm lab4.o list.lst main obj.o
gkarmikhos@fedora:~/work/arch-pc/lab04$ ./lab4
Armijos Karla
gkarmikhos@fedora:~/work/arch-pc/lab04$
```

рис. 5.3 Запустить программу

6. Выводы

В ходе выполнения задания были изучены основы работы с языком ассемблера, его специфика и практические применения.

Эта задача также включала сохранение файлов в локальном репозитории и их загрузку на GitHub.

7. Список используемой литературы

- 1) Архитектура ЭВМ