

Отчёт по лабораторной работе №4

Создание и процесс обработки программ на языке ассемблера NASM

Югай Александр Витальевич

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Программа Hello World!	7
3.2	Транслятор NASM	8
3.3	Расширенный синтаксис командной строки NASM	9
3.4	Компоновщик LD	9
3.5	Запуск исполняемого файла	10
3.6	Задание для самостоятельной работы	10
4	Выводы	13

Список иллюстраций

3.1	Создание каталога	7
3.2	Переходим в созданный каталог	7
3.3	Создание текстового файла	7
3.4	Открываем файл с помощью gedit и вводим программу	8
3.5	Используем команду <code>nasm</code>	8
3.6	Используем команду <code>ls</code>	9
3.7	Преобразуем файл <code>hello.asm</code> в <code>obj.o</code>	9
3.8	Используем команду <code>ls</code>	9
3.9	Используем команду <code>ld</code>	9
3.10	Используем команду <code>ls</code>	10
3.11	Используем команду <code>ld</code> для создания файла <code>main</code>	10
3.12	Используем команду <code>ls</code>	10
3.13	Используем команду <code>./hello</code>	10
3.14	Используем команду <code>cp</code>	11
3.15	Используем gedit и редактируем	11
3.16	Используем команды для работы файла и запускаем программу .	12
3.17	Копируем файлы в каталог <code>lab04</code>	12
3.18	Загружаем файлы	12

Список таблиц

1 Цель работы

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

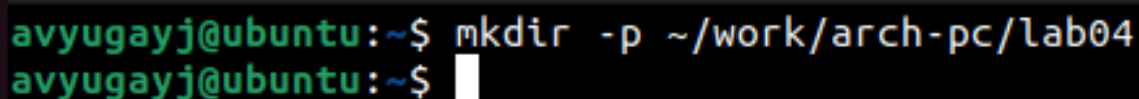
2 Задание

Написать программу на Ассемблере с выводом “Hello World!” и своего ФИО

3 Выполнение лабораторной работы

3.1 Программа Hello World!

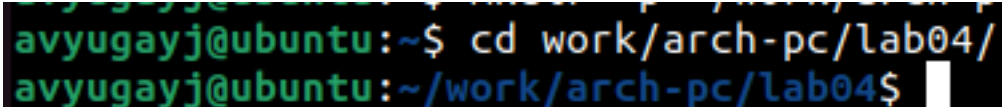
Создайте каталог для работы с программами на языке ассемблера NASM



```
avyugayj@ubuntu:~$ mkdir -p ~/work/arch-pc/lab04
avyugayj@ubuntu:~$
```

Рис. 3.1: Создание каталога

Перейдите в созданный каталог



```
avyugayj@ubuntu:~$ cd work/arch-pc/lab04/
avyugayj@ubuntu:~/work/arch-pc/lab04$
```

Рис. 3.2: Переходим в созданный каталог

Создайте текстовый файл с именем hello.asm



```
avyugayj@ubuntu:~/work/arch-pc/lab04$ touch hello.asm
```

Рис. 3.3: Создание текстового файла

Откройте этот файл с помощью любого текстового редактора, например, gedit и введите в него следующий текст:

```

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

```

Рис. 3.4: Открываем файл с помощью gedit и вводим программу

3.2 Транслятор NASM

Преобразуем текстовый файл в объектный код

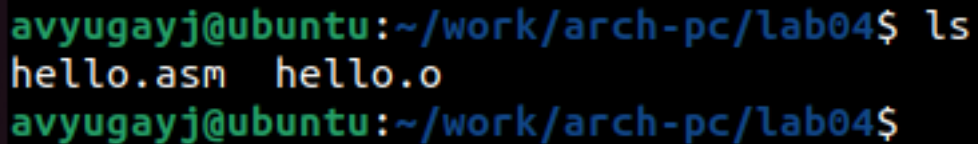
```

avyugayj@ubuntu:~/work/arch-pc/lab04$ nasm -f elf hello.asm

```

Рис. 3.5: Используем команду nasm

Проверяем правильность выполнения команды

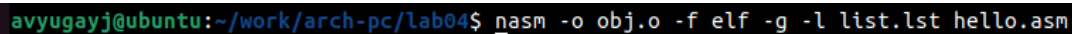


```
avyugayj@ubuntu:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
avyugayj@ubuntu:~/work/arch-pc/lab04$
```

Рис. 3.6: Используем команду ls

3.3 Расширенный синтаксис командной строки NASM

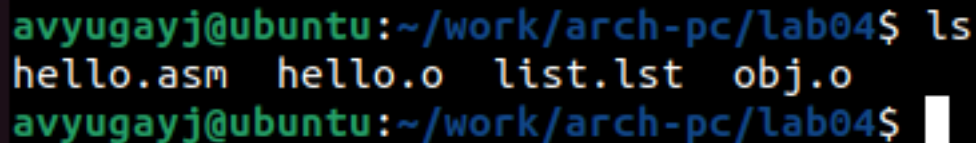
Компилируем исходный файл



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

Рис. 3.7: Преобразуем файл hello.asm в obj.o

Проверяем правильность выполнения команды



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

Рис. 3.8: Используем команду ls

3.4 Компоновщик LD

Передаем объектный файл на обработку компоновщику



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

Рис. 3.9: Используем команду ld

Проверяем созданся ли исполняемый файл

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

Рис. 3.10: Используем команду ls

Передаем объектный файл obj.o на обработку компоновщику

```
avyugayj@ubuntu:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
```

Рис. 3.11: Используем команду ld для создания файла main

Проверяем правильно выполнения команды

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

Рис. 3.12: Используем команду ls

3.5 Запуск исполняемого файла

Запускаем выполняемый файл

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

Рис. 3.13: Используем команду ./hello

3.6 Задание для самостоятельной работы

Копируем файл hello.asm

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

Рис. 3.14: Используем команду cp

Открываем файл и меняем Hello World на свое имя и фамилию

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

Рис. 3.15: Используем gedit и редактируем

Прописываем те же команды, что и с первой программой

```

avyugayj@ubuntu:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
avyugayj@ubuntu:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst lab4.asm
avyugayj@ubuntu:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o hello
avyugayj@ubuntu:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
avyugayj@ubuntu:~/work/arch-pc/lab04$ ./hello
Югай Александр

```

Рис. 3.16: Используем команды для работы файла и запускаем программу

Копируем файлы в локальный репозиторий

```

avyugayj@ubuntu:~/work/arch-pc/lab04$ cp hello.asm ~/work/study/2023-2024/Архитектура\ Компьютера/arch-pc/labs/lab04/
avyugayj@ubuntu:~/work/arch-pc/lab04$ cp lab4.asm ~/work/study/2023-2024/Архитектура\ Компьютера/arch-pc/labs/lab04/

```

Рис. 3.17: Копируем файлы в каталог lab04

Переходим в каталог лабораторных работ и загружаем файлы на github

```

avyugayj@ubuntu:~/work/study/2023-2024/Архитектура Компьютера/arch-pc$ git commit -a
[master a177d15] feat(main): add lab4
2 files changed, 31 insertions(+)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
avyugayj@ubuntu:~/work/study/2023-2024/Архитектура Компьютера/arch-pc$ git push
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 835 байтов | 835.00 КиБ/с, готово.
Всего 6 (изменений 2), повторно использовано 0 (изменений 0), повторно использовано
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:4heavensake/study_2023-2024_arch-pc.git
41e4e37..a177d15 master -> master

```

Рис. 3.18: Загружаем файлы

4 Выводы

Мы освоили процедуры компиляции и сборки программ, написанных на ассемблере NASM