

Отчёта по лабораторной работе 5

Создание и процесс обработки программ на языке ассемблера NASM

Агоссоу Вигнон Тримегистре Разиел НФИбд-05-22

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	10
5	Вопросы для самопроверки	11

Список иллюстраций

3.1	Файл hello.asm	7
3.2	Работа программы hello	8
3.3	Файл lab05.asm	8
3.4	Работа программы lab05	9

Список таблиц

1 Цель работы

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

2 Задание

1. Изучите программу HelloWorld и скомпилируйте ее.
2. С помощью любого текстового редактора внесите изменения в текст программы так, чтобы вместо Hello world! на экран выводилась строка с вашими фамилией и именем.
3. Скомпилируйте новую программу и проверьте ее работу.
4. Загрузите файлы на GitHub.

3 Выполнение лабораторной работы

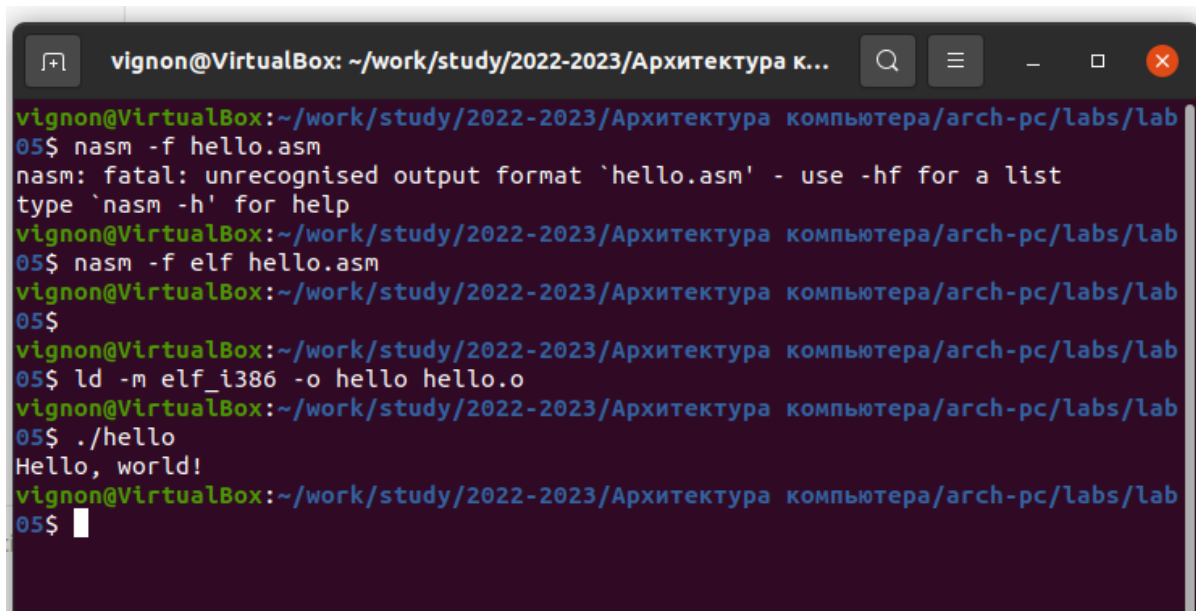
1. Создали каталог lab05 командой `mkdir`, перешел в него с помощью команды `cd`, скачал с ТУИС файл `hello.asm` и положил в папку. (рис. [3.1])
2. Открыли файл и изучили текст программы (рис. [3.1])



```
1 SECTION .data
2     hello:      db "Hello, world!",0xa
3     helloLen:   equ $ - hello
4 SECTION .text
5     global _start
6
7 _start:
8     mov eax, 4
9     mov ebx, 1
10    mov ecx, hello
11    mov edx, helloLen
12    int 0x80
13
14    mov eax, 1
15    mov ebx, 0
16    int 0x80
```

Рис. 3.1: Файл `hello.asm`

2. Транслировали файл командой `nasm`
3. Выполнили линковку командой `ld` и получили исполняемый файл и запустили его (рис. [3.2])



```
vignon@VirtualBox: ~/work/study/2022-2023/Архитектура к...
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ nasm -f hello.asm
nasm: fatal: unrecognized output format 'hello.asm' - use -hf for a list
type 'nasm -h' for help
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ nasm -f elf hello.asm
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ ld -m elf_i386 -o hello hello.o
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$ ./hello
Hello, world!
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05$
```

Рис. 3.2: Работа программы hello

4. Изменили сообщение Hello world на свое имя и запустили файл еще раз (рис. [3.3], [3.4])



```
name.asm
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05
hello.asm
name.asm
1 SECTION .data
2     hello:      db "Agossou Vignon",0xa
3     helloLen:   equ $ - hello
4 SECTION .text
5     global _start
6
7 _start:
8     mov eax, 4
9     mov ebx, 1
10    mov ecx, hello
11    mov edx, helloLen
12    int 0x80
13
14    mov eax, 1
15    mov ebx, 0
16    int 0x80
```

Рис. 3.3: Файл lab05.asm


```
05$  
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab  
05$ nasm -f elf name.asm  
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab  
05$ ld -m elf_i386 -o name name.o  
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab  
05$ ./name  
Agossou Vignon  
vignon@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab  
05$
```

Рис. 3.4: Работа программы lab05

4 Выводы

Освоили процесс компиляции и сборки программ, написанных на ассемблере `nasm`.

5 Вопросы для самопроверки

1. Какие основные отличия ассемблерных программ от программ на языках высокого уровня? - Ассемблер позволяет работать с ресурсами компьютера на уровне ядра ОС. Это язык низкого уровня, в котором с помощью кодовых инструкций пишутся команды прямо для процессора и регистров.
2. В чём состоит отличие инструкции от директивы на языке ассемблера? - Инструкции выполняются прямо процессором как машинные команды. Директивы не выполняются как команды, а обрабатываются транслятором в инструкции
3. Перечислите основные правила оформления программ на языке ассемблера.
- Типичный формат записи команд NASM имеет вид: [метка:] мнемокод [операнд {, операнд}] [; комментарий]
4. Каковы этапы получения исполняемого файла? - Написание кода программы, трансляция кода в объектный файл, линковка объектного файла в исполняемый.
5. Каково назначение этапа трансляции? - — преобразование с помощью транслятора, например `nasm`, текста программы в машинный код, называемый объектным
6. Каково назначение этапа компоновки? - этап обработки объектного кода компоновщиком (`ld`), который принимает на вход объектные файлы и собирает по ним исполняемый файл.

7. Какие файлы могут создаваться при трансляции программы, какие из них создаются по умолчанию? - Создается объектный файл .o и можно получить файл листинга .lst.
8. Каковы форматы файлов для nasm и ld? - для nasm на вход подается текст программы в формате .asm. для ld подается объектный файл, полученный от nasm, в формате .o