

Лабораторная работа №4

Архитектура компьютера

Казначеева Кристина Никитична

Содержание

1	Цель работы	3
2	Задание	4
3	Выполнение лабораторной работы	5
4	Выводы	9

1 Цель работы

Изучение процесса компиляции и сборки программ на ассемблере NASM.

2 Задание

Этот практикум посвящен освоению языка ассемблера NASM. В ходе лабораторной работы мы познакомимся с основами программирования на ассемблере NASM. Мы создадим программу “Hello world!”, узнаем, как работает транслятор NASM, освоим расширенные возможности командной строки NASM, научимся использовать компоновщик LD и запускать получившийся исполняемый файл.

3 Выполнение лабораторной работы

Создадим каталог для работы с программами на языке ассемблера NASM, затем перейдём в него (рис. 3.1).

```
knkaznacheeva@dk8n68 ~ $ mkdir -p ~/work/arch-pc/lab04
knkaznacheeva@dk8n68 ~ $ cd ~/work/arch-pc/lab04
```

Рис. 3.1: Создание каталога и переход в него

Создадим текстовый файл с именем hello.asm (рис. 3.2).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ touch hello.asm
```

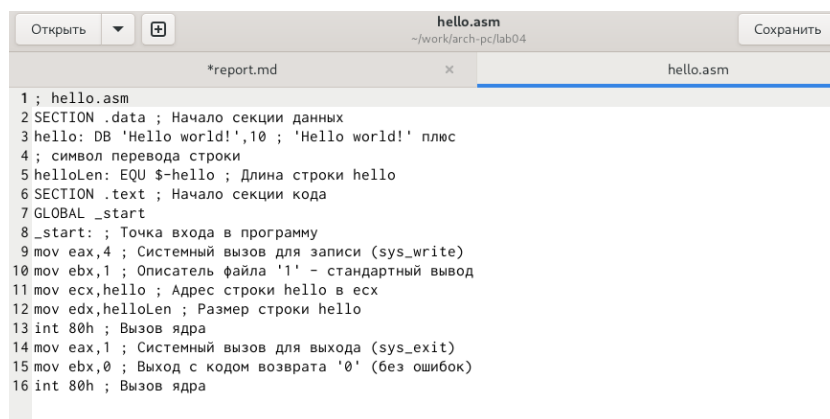
Рис. 3.2: Создание текстового файла

Откроем этот файл с помощью gedit (рис. 3.3).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 3.3: Открытие файла с gedit

Введём в созданный текстовый файл текст (рис. 3.4).



```
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. 3.4: Ввод текста

Компилируем текст программы «Hello World» (рис. 3.5).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
```

Рис. 3.5: Компиляция текста

Скомпилируем исходный файл hello.asm в obj.o (рис. 3.6).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
```

Рис. 3.6: Компиляция файла в obj.o

Передадим на обработку компоновщику объектный файл необходимо, чтобы получить исполняемую программу (рис. 3.7).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
```

Рис. 3.7: Обработка файла компоновщиком

Ключ -o с последующим значением задаёт имя создаваемого исполняемого файла. Выполним следующую команду (рис. 3.8):

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
```

Рис. 3.8: Выполнение команды с ключом -o

Запустим на выполнение созданный исполняемый файл, находящийся в текущем каталоге (рис. 3.9).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

Рис. 3.9: Запуск файла на выполнение

В каталоге ~/work/arch-pc/lab04 с помощью команды `cp` создадим копию файла `hello.asm` с именем `lab4.asm` (рис. 3.10).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ cp ~/work/arch-pc/lab04/hello.asm ~/work/arch-pc/lab04/lab4.asm
```

Рис. 3.10: Создание копию файла

С помощью `gedit` внесём изменения в текст программы в файле `lab4.asm` так, чтобы вместо `Hello world!` на экран выводилась строка с фамилией и именем (рис. 3.11).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ gedit lab4.asm
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
```

Рис. 3.11: Изменение текста программы

Оттранслируем полученный текст программы `lab4.asm` в объектный файл (рис. 3.12). Выполняем компоновку объектного файла и запустите получившийся исполняемый файл (рис. 3.13).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst lab4.asm
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4.asm  lab4.o  list.lst  main  obj.o
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab4.o -o lab4
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab4  lab4.asm  lab4.o  list.lst  main  obj.o
```

Рис. 3.12: Оттранслирование текста программы

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ ./lab4
Казначеева Кристинаknkaznacheeva@dk8n68 ~/work/arch-
```

Рис. 3.13: Запуск получившегося исполняемого файла

Скопируем файлы hello.asm и lab4.asm в локальный репозиторий в каталог ~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/ (рис. 3.14).

```
knkaznacheeva@dk8n68 ~/work/arch-pc/lab04 $ cd ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/labs/lab04/
knkaznacheeva@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab04 $ ls
hello.asm lab4.asm presentation report
```

Рис. 3.14: Копирование файла в локальный репозиторий

Загрузим файлы на Github (рис. 3.15) и (рис. 3.16).

```
knkaznacheeva@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc
] $ git add .
knkaznacheeva@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc
] $ git commit -am 'feat(main): lab4'
[master 7d32556] feat(main): lab4
23 files changed, 4149 insertions(+)
create mode 100644 labs/lab02/report 2.zip
```

Рис. 3.15: Загрузка файлов на Github

```
knkaznacheeva@dk8n68 ~/work/study/2024-2025/Архитектура компьютера/arch-pc
] $ git push
Перечисление объектов: 18, готово.
Подсчет объектов: 100% (18/18), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (12/12), готово.
Запись объектов: 100% (12/12), 2.88 МиБ | 3.16 МиБ/с, готово.
Total 12 (delta 6), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (6/6), completed with 4 local objects.
```

Рис. 3.16: Загрузка файлов на Github

4 Выводы

В ходе лабораторной работы мы получили практические навыки работы с ассемблером NASM, создали программу “Hello world!”, изучили процесс компиляции и сборки с помощью транслятора NASM и компоновщика LD, а также разобрались с расширенными возможностями командной строки NASM.