Отчёт по лабораторной работе №4

Создание и процесс обработки программ на языке ассемблера NASM

Маев Даниил Егорович

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

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

2 Задание

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

3 Выполнение лабораторной работы

1. Создайте каталог для работы с программами на языке ассемблера NASM:

```
demaev@dk8n60 ~ $ mkdir -p ~/work/arch-pc/lab04
```

Рис. 3.1: Создание каталога

2. Перейдём в созданный каталог:

```
demaev@dk8n60 ~ $ cd ~/work/arch-pc/lab04
```

Рис. 3.2: Переход в каталог

3. Создадим текстовый файл с именем hello.asm:

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ touch hello.asm
```

Рис. 3.3: Создание текстового файла

4. Откроем этот файл с помощью текстового редактора

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ gedit hello.asm
```

Рис. 3.4: Открытие файла

5. Введём в него текст:

```
 1
 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. 3.5: Ввод текста

6. Скомпилируем данный текст

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
```

Рис. 3.6: Компиляция текста

7. Проверим, что объектный файл был создан:

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o
```

Рис. 3.7: проверка, что объектный файл был создан

8. Скомпилируем исходный файл hello.asm в obj.o и создадим файл листинга list.lst

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
```

Рис. 3.8: Создание файлов

9. Проверим, что файлы были созданы.

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o
```

Рис. 3.9: Проверка, что файлы были созданы.

10. Передадим объектный файл на обработку компоновщику.

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
```

Рис. 3.10: Передача файла на компоновку

11. Проверим, что исполняемый файл hello был создан.

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис. 3.11: Проверка, что исполняемый файл hello был создан

12. Зададим имя создаваемого исполняемого файла.

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
```

Рис. 3.12: Зададим имя создаваемого исполняемого файла

13. Запустим на выполнение созданный исполняемый файл, находящийся в текущем каталоге.

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ ./hello
Hello world!
```

Рис. 3.13: Запуск на выполнение созданный исполняемый файл

14. Создадим копию файла hello.asm с именем lab04.asm

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ cp hello.asm lab04.asm
```

Рис. 3.14: Создание копии файла с именем lab4.asm

15. Внесём изменения в текст программы в файле lab04.asm

```
 1
 2 SECTION .data ; Начало секции данных
 3 hello: DB 'Daniil Maev',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. 3.15: Внесение изменения в текст программы

16. Оттранслируем полученный текст программы lab5.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл.

```
demaev@dk8n60 ~/work/arch-pc/lab04 $ gedit lab04.asm
demaev@dk8n60 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst lab04.asm
demaev@dk8n60 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab04.o -o lab04
ld: невозможно найти lab04.o: Нет такого файла или каталога
demaev@dk8n60 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o lab04
demaev@dk8n60 ~/work/arch-pc/lab04 $ ./lab04
Daniil Maev
```

Рис. 3.16: Оттранслирование, компоновка, запуск

17. Скопируем файлы hello.asm и lab04.asm в локальный репозиторий и загрузим файлы на Github.

4 Выводы

В ходе выполнения работы, я освоил процедуры компиляции и сборки программ, написанных на ассемблере NASM.