

Отчёт по лабораторной работе №4

Дисциплина: архитектура компьютера

Новиков Никита Владимирович

Содержание

1 Цель работы

Цель данной лабораторной работы - освоить процедуры компиляции и сборки программ, написанных на ассемблере

2 Задание

1. Создание программы Hello world!
2. Работа с транслятором NASM
3. Работа с расширенным синтаксисом командной строки
4. Работа с компоновщиком LD
5. Запуск исполняемого файла
6. Выполнение заданий для самостоятельной работы

3 Выполнение лабораторной работы

3.1 Создание программы Hello world!

С помощью команды `mkdir` и ключа `-p` создаю директорию `arch-pc/lab04` в `work`.

```
(nvnovikov1@nvnovikov1)-[~]  
$ mkdir -p ~/work/arch-pc/lab04
```

Перехожу в созданную директорию при помощи утилиты `cd`.

```
(nvnovikov1@nvnovikov1)-[~]  
$ cd ~/work/arch-pc/lab04
```

Создаю в текущем каталоге пустой текстовый файл `hello.asm`.

```
(nvnovikov1@nvnovikov1)-[~/work/arch-pc/lab04]  
$ touch hello.asm
```

Открываю созданный файл при помощи текстового редактора mousepad.

```
*~/work/arch-pc/lab04/hello.asm - Mousepad
File Edit Search View Document Help

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
15    mov eax,1 ; Системный вызов для выхода (sys_exit)
16    mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
17    int 80h ; Вызов ядра
```

3.2 Работа с транслятором NASM

Превращаю текст программы для вывода “Hello World!” в объектный файл с помощью транслятора NASM.

```
(nvnovikov1@nvnovikov1)-[~/work/arch-pc/lab04]
$ nasm -f elf hello.asm
```

При помощи команды ls проверяю правильность выполнения предыдущих

действий.

```
(nvnovikov1@nvnovikov1)-[~/work/arch-pc/lab04]
$ ls
hello.asm hello.o
```

3.3 Работа с расширенным синтаксисом командной строки

Ввожу команду, которая компилирует файл hello.asm в объектный файл obj.o. Потом проверяю правильность выполнения команд утилитой ls.

```
(nvnovikov1@nvnovikov1)-[~/work/arch-pc/lab04]
$ nasm -o obj.o -f elf -g -l list.lst hello.asm

(nvnovikov1@nvnovikov1)-[~/work/arch-pc/lab04]
$ ls
hello.asm hello.o list.lst obj.o
```

3.4 Работа с компоновщиком LD

Передаю файл hello.o на обработку компоновщику LD, дабы получить исполняемый файл hello. Далее проверяю с помощью утилиты ls

правильность выполнения команд.

```
(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ld -m elf_i386 hello.o -o hello

(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ls
hello hello.asm hello.o list.lst obj.o
```

Выполняю следующую команду. При её исполнении файл будет иметь имя main, т.к. после ключа -o мы задали значение main.

```
(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ld -m elf_i386 obj.o -o main

(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ls
hello hello.asm hello.o list.lst main obj.o
```

3.5 Запуск исполняемого файла

Запускаю созданный исполняемый файл hello.

```
(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ./hello
Hello world!
```

3.6 Выполнение заданий для самостоятельной работы

С помощью утилиты cp создаю в текущем каталоге копию файла hello.asm с именем lab04.asm.

```
(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ cp hello.asm lab04.asm
```

С помощью текстового редактора mousepad открываю файл lab04.asm и вношу изменения в программу так, чтобы она выводила мои имя и фамилию.

Компилирую текст программы в объектный файл и проверяю правильность создания файла lab04.o

```
(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ nasm -f elf lab04.asm

(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ls
hello hello.asm hello.o lab04.asm lab04.o list.lst main obj.o
```

Передаю объектный файл на компоновку, чтоб получить исполняемый

```
(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ld -m elf_i386 lab04.o -o lab04

(nvnikov1@nvnikov1)-[~/work/arch-pc/lab04]
$ ls
hello hello.asm hello.o lab04 lab04.asm lab04.o list.lst main obj.o
```

файл lab04.

Запускаю исполняемый файл lab04.

```
(nvnovikov1@nvnovikov1)-[~/work/arch-pc/lab04]  
$ ./lab04  
Novikov Nikita
```

При помощи команды `git add .` и `git commit` добавляю файлы на github.

```
(nvnovikov1@nvnovikov1)-[~/../Архитектура компьютера/arch-pc/labs/lab04]  
$ git add .  
  
(nvnovikov1@nvnovikov1)-[~/../Архитектура компьютера/arch-pc/labs/lab04]  
$ git commit -m "Add files for lab04"  
[master cbab367] Add files for lab04  
2 files changed, 34 insertions(+)  
create mode 100644 labs/lab04/hello.asm  
create mode 100644 labs/lab04/lab04.asm
```

Отправляю файлы на сервер с помощью команды `git push`.

```
(nvnovikov1@nvnovikov1)-[~/../Архитектура компьютера/arch-pc/labs/lab04]  
$ git push  
Enumerating objects: 9, done.  
Counting objects: 100% (9/9), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (6/6), done.  
Writing objects: 100% (6/6), 960 bytes | 960.00 KiB/s, done.  
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0  
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.  
To github.com:nvnovikov1/study_2023-2024_arh-pc.git  
237c74e..cbab367 master -> master
```

4 Выводы

При выполнении данной лабораторной работы я освоил процедуры компиляции и сборки программ, написанных на языке ассемблер.

5 Список литературы

<https://esystem.rudn.ru/mod/resource/view.php?id=1030552>