

Отчёт по лабораторной работе №3

дисциплина: Архитектура компьютера

Аносов Даниил Игоревич

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Задание для самостоятельной работы	11
5	Выводы	13

Список иллюстраций

3.1	Переход в каталог курса	7
3.2	Редактирование кода в hello.asm	8
3.3	NASM installation on Arch Linux	8
3.4	Компиляция программы	9
3.5	Компоновка объектного файла	9
3.6	Проверка удаления файлов report.pdf и report.docx	9
3.7	Копирование файла с кодом	10
4.1	Редактирование кода	11
4.2	Переход в каталог с отчётом по лабораторной работе №2	12
4.3	Запуск итогового кода	12
4.4	Копирование файлов в каталог лабораторной работы	12

Список таблиц

1 Цель работы

Приобретение навыков работы с процедурой оформления отчётов с помощью языка разметки Markdown.

2 Задание

3 Выполнение лабораторной работы

Откроем терминал и перейдём в каталог курса. (рис. 3.1). Создадим каталог lab04 для работы с NASM, тронем в нём файл *hello.asm*

```
[dianosov@archlinux study]$ ls
arch-pc
[dianosov@archlinux study]$ cd arch-pc/
CHANGELOG.md
config/
COURSE
.git/
.gitattributes
.gitignore
.gitmodules
HI_QiYsKILxRpg3hIP6sJ7fM7Pq1ONvZlMIXxw.woff2
labs/
LICENSE
Makefile
prepare
presentation/
README.en.md
README.git-flow.md
README.md
README.pdf
README.tex
template/
[dianosov@archlinux study]$ cd arch-pc/
[dianosov@archlinux arch-pc]$ mkdir -p ~/study/arch-pc/lab04
[dianosov@archlinux arch-pc]$ cd ~/study/arch-pc/lab04
[dianosov@archlinux lab04]$ touch hello.asm
[dianosov@archlinux lab04]$
```

Рис. 3.1: Переход в каталог курса

В файл *hello.asm* поместим код, предложенный в документе, приложенном к лабораторной работе (рис. 3.2).

```

SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
~
~
~
~
~
~
~
~
~
~
"hello.asm" [noeol] 16L, 800B

```

Рис. 3.2: Редактирование кода в hello.asm

Заметим, что в отличие от многих современных высокоуровневых языков программирования, в ассемблерной программе каждая команда располагается на отдельной строке. Размещение нескольких команд на одной строке недопустимо.

Пакетным менеджером *pacman* установим компилятор ассемблера *nasm*. (рис. 3.3).

```

[dianosov@archlinux lab04]$ vim hello.asm
[dianosov@archlinux lab04]$ sudo pacman -S nasm
[sudo] password for dianosov:
resolving dependencies...
looking for conflicting packages...

Packages (1) nasm-2.16.03-1
Total Download Size: 0.33 MiB
Total Installed Size: 2.45 MiB

:: Proceed with installation? [Y/n]
:: Retrieving packages...
nasm-2.16.03-1-x86_64 341.2 KiB 1161 KiB/s 00:00 [#####] 100%
(1/1) checking keys in keyring [#####] 100%
(1/1) checking package integrity [#####] 100%
(1/1) loading package files [#####] 100%
(1/1) checking for file conflicts [#####] 100%
(1/1) checking available disk space [#####] 100%
:: Processing package changes...
(1/1) installing nasm [#####] 100%
:: Running post-transaction hooks...
(1/1) Arming ConditionNeedsUpdate...
[dianosov@archlinux lab04]$

```

Рис. 3.3: NASM installation on Arch Linux

Проведём компиляцию программы, убедимся, что выходные файлы появились в директории (рис. 3.4).


```
[dianosov@archlinux lab04]$ nasm -f elf hello.asm
[dianosov@archlinux lab04]$ ls
hello.asm  hello.o
[dianosov@archlinux lab04]$
```

Рис. 3.4: Компиляция программы

Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику. Прделаем это, как показано на картинке (рис. 3.5).

```
[dianosov@archlinux lab04]$ ld -m elf_i386 hello.o -o hello
[dianosov@archlinux lab04]$ ls
hello  hello.asm  hello.o
[dianosov@archlinux lab04]$
```

Рис. 3.5: Компоновка объектного файла

Как видно, появился исполняемый файл **hello**

Проверим, что программа исполняется корректно (рис. 3.6).

```
[dianosov@archlinux lab04]$ ld -m elf_i386 hello.o -o hello
[dianosov@archlinux lab04]$ ls
hello  hello.asm  hello.o
[dianosov@archlinux lab04]$ ./hello
Hello world!
[dianosov@archlinux lab04]$
```

Рис. 3.6: Проверка удаления файлов report.pdf и report.docx

Скопируем файл с кодом и назовём новый экземпляр *lab04.asm*, проверим, что весь код на месте (рис. 3.7).

```

[dianosov@archlinux lab04]$ cp hello.asm lab04.asm
[dianosov@archlinux lab04]$ ls
hello hello.asm hello.o lab04.asm
[dianosov@archlinux lab04]$ cat lab04.asm

SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра[dianosov@archlinux lab04]$ █

```

Рис. 3.7: Копирование файла с кодом

4 Задание для самостоятельной работы

Откроем файл с помощью редактора *vim*. В коде вместо *Hello world* напомним фамилию и имя автора лабораторной работы. Для семантики также поменяем *hello* на *name* (рис. 4.1).

```
SECTION .data ; Начало секции данных
name: DB 'Daniil Anosov',10 ; 'Daniil Anosov' плюс
; символ перевода строки
nameLen: EQU $-name ; Длина строки name
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,name ; Адрес строки name в ecx
mov edx,nameLen ; Размер строки name
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
~
~
~
~
~
~
~
~
~
~
~
```

Рис. 4.1: Редактирование кода

Скомпилируем код и обработаем его компоновщиком (аналогично предыдущему примеру) (рис. 4.2).

```
[dianosov@archlinux lab04]$ nasm -f elf lab04.asm
[dianosov@archlinux lab04]$ ld -m elf_i386 lab04.o -o lab04
[dianosov@archlinux lab04]$ ls
hello hello.asm hello.o lab04 lab04.asm lab04.o
[dianosov@archlinux lab04]$
```

Рис. 4.2: Переход в каталог с отчётом по лабораторной работе №2

Запустим полученный на выходе исполняемый файл **lab04** и убедимся, что код работает корректно (рис. 4.3).

```
[dianosov@archlinux lab04]$ ls
hello hello.asm hello.o lab04 lab04.asm lab04.o
[dianosov@archlinux lab04]$ ./lab04
Daniil Anosov
[dianosov@archlinux lab04]$
```

Рис. 4.3: Запуск итогового кода

```
[dianosov@archlinux lab04]$ cp . ~/study/arch-pc/labs/lab04/report/asm-dir
cp: -r not specified; omitting directory '.'
[dianosov@archlinux lab04]$ cp . ~/study/arch-pc/labs/lab04/report/asm-dir -r
[dianosov@archlinux lab04]$
```

Рис. 4.4: Копирование файлов в каталог лабораторной работы

Скопируем всё содержимое текущего каталога в `~/study/arch-pc/labs/lab04/report/asm-dir` (рис. 4.4).

Загрузим на Github всю новую версию проекта. add, commit
push

5 Выводы

В ходе выполнения данной лабораторной работы была освоена процедура оформления отчётов с помощью языка разметки Markdown.