Front matter

title: "Шаблон отчёта по лабораторной работе 4" subtitle: "Простейший вариант" author: "Абдуллахи Шугофа"

Generic otions

lang: ru-RU toc-title: "Содержание"

Bibliography

bibliography: bib/cite.bib csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

Pdf output format

toc: true # Table of contents toc-depth: 2 lof: true # List of figures lot: true # List of tables fontsize: 12pt linestretch: 1.5

papersize: a4 documentclass: scrreprt

I18n polyglossia

polyglossia-lang: name: russian options: - spelling=modern - babelshorthands=true polyglossia-otherlangs: name:

english

I18n babel

babel-lang: russian babel-otherlangs: english

Fonts

mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions: Ligatures=TeX

romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase monofontoptions:

Scale=MatchLowercase,Scale=0.9

Biblatex

biblatex: true biblio-style: "gost-numeric" biblatexoptions:

- parentracker=true

- backend=biber • hyperref=auto • language=auto
- autolang=other*
- citestyle=gost-numeric

Pandoc-crossref LaTeX customization

figureTitle: "Рис." tableTitle: "Таблица" listingTitle: "Листинг" lofTitle: "Список иллюстраций" lotTitle: "Список

таблиц" lolTitle: "Листинги"

Misc options

indent: true header-includes: -\usepackage{indentfirst} -\usepackage{float}

keep figures where there are in the text -\floatplacement{figure}{H} # keep figures where there are in the text

Цель работы

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

Задание

Рассмотрим пример простой программы на языке ассемблера NASM. Традиционно первая программа выводит приветственное сообщение Hello world! на экран.Создайте каталог для работы с программами на языке ассемблера NASM: Создал каталог для работы с программами на языке ассемблера NASM.

```
shogofa@shogofa-VirtualBox:~$ mkdir -p ~/work/arch-pc/lab04
shogofa@shogofa-VirtualBox:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos  work
shogofa@shogofa-VirtualBox:~$
```

2-Перейдите в созданный каталог:

```
shogofa@shogofa-VirtualBox:~$ cd ~/work/arch-pc/lab04
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
```

3- Создайте текстовый файл с именем hello.asm

Создал текстовый файл с именем hello.asm

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ touch hello.asm
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
hello.asm
```

4-откройте этот файл с помощью любого текстового редактора, например, gedit

Открыл этот файл с помощью gedit

и введите в него следующий текст: ; hello.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)

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ gedit hello.asm
Gtk-Message: 13:49:01.861: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try t
o not load it.

(gedit:77343): GLib-GIO-WARNING **: 13:49:02.854: Error creating IO channel for /proc/self/mountinfo: Permission denied
 (g-file-error-quark, 2)

** (gedit:77343): WARNING **: 13:49:28.543: atk-bridge: get_device_events_reply: unknown signature
```

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

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
```

5-NASM превращает текст программы в объектный код. Например, для компиляции приве-дённого выше текста программы «Hello World» необходимо написать: nasm -f elf hello.asm

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ nasm -f elf hello.asm
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
```

6-Выполните следующую команду: nasm -o obj.o -f elf -g -l list.lst hello.asm

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.asm
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
```

7-Как видно из схемы на рис. 4.3, чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику:

ld -m elf_i386 hello.o -o hello

С помощью команды ls проверьте, что исполняемый файл hello был создан. Компоновщик ld не предполагает по умолчанию расширений для файлов, но принято использовать следующие расширения:

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

8-Ключ -o с последующим значением задаёт в данном случае имя создаваемого исполняе-мого файла. Выполните следующую команду:

ld -m elf_i386 obj.o -o main

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
```

9-Какое имя будет иметь исполняемый файл? Какое имя имеет объектный файл из которого собран этот исполняемый файл? Формат командной строки LD можно увидеть, набрав ld --help. Для получения более подробной информации см. man ld. Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге,можно, набрав в командной строке:

./hello

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ./hello
Hello world!
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$
```

Задание для самостоятельной работы

1- В каталоге ~/work/arch-pc/lab04 с помощью команды cp создайте копию файла hello.asm с именем lab4.asm

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ cd ..
shogofa@shogofa-VirtualBox:~/work/arch-pc$ cp lab04/hello.asm lab04/lab4.asm
shogofa@shogofa-VirtualBox:~/work/arch-pc$ cd lab04
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
```

2- С помощью любого текстового редактора внесите изменения в текст программы в файле lab4.asm так, чтобы

вместо Hello world! на экран выводилась строка с вашими фамилией и именем.

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ gedit hello.asm lab4.asm
Gtk-Message: 14:00:29.136: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try t
o not load it.

(gedit:78413): GLib-GIO-WARNING **: 14:00:30.071: Error creating IO channel for /proc/self/mountinfo: Permission denied
 (g-file-error-quark, 2)

** (gedit:78413): WARNING **: 14:00:55.712: atk-bridge: get_device_events_reply: unknown signature
```

```
;hello.asm

SECTION .data                       ; Начало секции данных

      hello:   DB 'shogofa Abdullahi',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
```

3-Оттранслируйте полученный текст программы lab4.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.

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

4- Скопируйте файлы hello.asm и lab4.asm в Ваш локальный репозиторий в ката- лог ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/. Загрузите файлы на Github.

```
shogofa@shogofa-VirtualBox:~/work/arch-pc/lab04$ cd
shogofa@shogofa-VirtualBox:~$ cp ~/work/arch-pc/lab04/hello.asm ~/work/study/2023-2024/Архитектура\ компьютера/study-2023-2024-arh-pc/labs/lab04
shogofa@shogofa-VirtualBox:~$ cp ~/work/arch-pc/lab04/lab4.asm ~/work/study/2023-2024/Архитектура\ компьютера/study-2023-2024-arh-pc/labs/lab04
shogofa@shogofa-VirtualBox:~$
```

```
shogofa@shogofa-VirtualBox:~$ cd ~/work/study/2023-2024/Архитектура\ компьютера/study-2023-2024-arh-pc/labs/lab04
shogofa@shogofa-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study-2023-2024-arh-pc/labs/lab04$ git add .
shogofa@shogofa-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study-2023-2024-arh-pc/labs/lab04$ git commit
-am 'shogofa'
[master 3d27a8c] shogofa
 2 files changed, 50 insertions(+)
 create mode 100644 labs/lab04/hello.asm
 create mode 100644 labs/lab04/lab4.asm
shogofa@shogofa-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/study-2023-2024-arh-pc/labs/lab04$ git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 909 bytes | 227.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:Shogofa21/study-2023-2024-arh-pc.git
   0434f75..3d27a8c  master -> master
```

Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. @tbl:std-dir приведено краткое описание стандартных каталогов Unix.

: Описание некоторых каталогов файловой системы GNU Linux {#tbl:std-dir}

Имя каталога	Описание каталога
`/`	Корневая директория, содержащая всю файловую
`/bin`	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
`/etc`	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
`/home`	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
`/media`	Точки монтирования для сменных носителей
`/root`	Домашняя директория пользователя `root`
`/tmp`	Временные файлы
`/usr`	Вторичная иерархия для данных пользователя

Более подробно об Unix см. в [@gnu-doc:bash;@newham:2005:bash;@zarrelli:2017:bash;@robbins:2013:bash;@tannenbaum:arch pc:ru;@tannenbaum:modern-os:ru].

Выполнение лабораторной работы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. @fig:001).

Название рисунка{#fig:001 width=70%}

Выводы

Здесь кратко описываются итоги проделанной работы.

Список литературы{.unnumbered}

::: {#refs} :::