|  |  |  |
| --- | --- | --- |
| Санкт-Петербургский политехнический университет Петра Великого  Институт компьютерных наук и технологий | | |
| **Лабораторная работа №8 Драйверы** | | |
|  | | |
| по дисциплине «Системное программное обеспечение GNU/Linux» | | |
| Выполнил | | |
| студент гр.3530904/10004 |  | Д.С. Малахов |
|  | | |
| Руководитель |  | А.В. Петров |
|  | | |
| Санкт-Петербург | | |
| 2022 | | |

**Оглавление**

[**Задание** 3](#_Toc100267006)

[**Ход работы** 4](#_Toc100267007)

[**Возникшие трудности при выполнении** 4](#_Toc100267008)

[**Аппаратная платформа** 5](#_Toc100267009)

[**Программная платформа** 6](#_Toc100267010)

[**Вывод** 7](#_Toc100267011)

# **Задание**

1. Изучите приведённые примеры взаимодействия с драйверами устройств.
2. Выполните одно из следующих заданий (вариант взять у преподавателя). Требуется реализовать драйвер, поддерживающий функции открытия, закрытия, записи, чтения и имеющий документацию. При записи в драйвер могут передаваться команды. Для этого нужно проводить общий анализ передаваемых в него строк, чтобы определять какие команды передаются. Одна из них direction [forward/back] – направление дальнейшего чтения из драйвера. Например последовательность команд с драйвером echo “direction back” > /dev/chardev cp /dev/chardev text приведёт к чтению строки или буфера драйвера в обратном порядке, если драйвер поддерживает работу со строками. То же для файлов.
   1. Драйвер поддерживает чтение и запись сообщений в него через пользовательский процесс: ioctl\_set\_msg(file\_desc, msg). Поддерживает функцию удаления сообщения: ioctl\_msg\_delete.
3. Оформить отчёт, включающий в себя задание, метод выполнения, возникшие сложности и методы решения.

# **Ход работы**

1. Сборка кода из примеров в техзадании
2. Создание файла устройства cd /dev; sudo mknod char\_dev c 100 0
3. Установка модуля sudo insmod L8/module/chardev.ko
4. Написание контроллера ввода вывода.
5. Тестирование
6. Написание отчёта

# **Возникшие трудности при выполнении**

1. Код в примерах был несколько устаревшим. Сейчас в структуре file\_operations поле ioctl заменено на поле unlocked\_ioctl
2. Реализация ввода из потока на C оказалась несколько менее тривиальной задачей, чем я ожидал изначально.

# **Аппаратная платформа**

Architecture: x86\_64

CPU op-mode(s): 32-bit, 64-bit

Byte Order: Little Endian

CPU(s): 4

On-line CPU(s) list: 0-3

Thread(s) per core: 1

Core(s) per socket: 4

Socket(s): 1

NUMA node(s): 1

Vendor ID: GenuineIntel

CPU family: 6

Model: 142

Model name: Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz

Stepping: l2

CPU MHz: 2111.988

BogoMIPS: 4223.99

Hypervisor vendor: KVM

Virtualization type: full

L1d cache: 128K

L1i cache: 128K

L2 cache: 1024K

L3 cache: 24576K

NUMA node0 CPU(s): 0-3

# **Программная платформа**

Linux version 5.10.0-11-amd64 (gcc-10 (Debian 10. 2.1-6) 10.2.1 20210110, GNU ld (GNU binutils for Debian) 2.35.2) #1 SMP Debian 5.10.92-1 (2022-01-18)

# **Вывод**

В целом, реализованный механизм взаимодействия с драйвером меня устраивает. Он решает поставленную в лабораторной работе задачу, переносим между разными дистрибутивами Linux и интегрируем в сторонних программных продуктах при необходимости.