Лабораторная работа 2

По дисциплине “Системы ввода-вывода”

“Основы написания драйверов устройств с использованием операционной системы”

Вариант 4

Выполнил:

Баянов Равиль Динарович
Кузнецов Даниил Александрович

Поток 1.4

Санкт-Петербург

2025 г.

Оглавление

Цель

Познакомится с основами разработки драйверов устройств с использованием операционной системы на примере создания драйверов символьных устройств под операционную систему Linux.

Задачи

1. Написать драйвер символьного устройства, удовлетворяющий требованиям:

• должен создавать символьное устройство /dev/varN, где N – это номер варианта

• должен обрабатывать операции записи и чтения в соответствии с вариантом задания

2. Оформить отчет по работе в электронном формате

Вариант

При записи текста в файл символьного устройства должен осуществляться подсчет введенных цифр. Последовательность полученных результатов (количество цифр) с момента загрузки модуля ядра должна выводиться при чтении файла.

Описание функций

Функция чтения из устройства:

Возвращает количество введённых цифр при чтении из устройства.

snprintf формирует строку с количеством цифр.

copy_to_user копирует данные из пространства ядра в пространство пользователя.

Если данные уже были прочитаны (*off > 0), функция завершает чтение, что реализует поведение "прочитать один раз".

```
static ssize_t my_read(struct file *f, char __user *buf, size_t len,
loff_t *off)
{
  char str[100];
  ssize_t str_len;

  str_len = snprintf(str, sizeof(str), "Total digits entered: %d\n",
digits_counter);

  if (len < str_len) {
    return -EINVAL;
  }

  if (*off > 0) {
    return 0;
  }

  if (copy_to_user(buf, str, str_len)) {
    return -EFAULT;
  }

  *off += str_len;
  return str_len;
}
```

Функция записи в устройство:

Записывает данные из пространства пользователя в буфер ядра.

Считает количество цифр в записанных данных и увеличивает digits_counter.

copy_from_user копирует данные из пространства пользователя в пространство ядра.

printk логирует количество принятых байтов и цифр.

```
static ssize_t my_write(struct file *f, const char __user *buf,
size_t len, loff_t *off)
{
  size_t bytes_to_write = (len > BUF_SIZE) ? BUF_SIZE : len;

  if (copy_from_user(device_buffer, buf, bytes_to_write)) {
    return -EFAULT;
  }
```

```
  device_buffer[bytes_to_write] = '\0';
  data_len = bytes_to_write;
  printk(KERN_INFO "Driver: wrote %zu bytes\n", bytes_to_write);

  char* input_str = device_buffer;
  int i = 0;
  while (input_str[i] != '\0') {
    if (input_str[i] >= '0' && input_str[i] <= '9') {
      digits_counter++;
    }
    i++;
  }
  printk(KERN_INFO "Driver: wrote %zu digits\n", digits_counter);
  device_buffer[0] = '\0';
  return bytes_to_write;
}
```

Вывод

Выполнив данную лабораторную работу, мы реализовали простой драйвер устройств с помощью ОС. Мы изучили как работают драйверы, работающие под операционную систему Linux.