Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа по СВВ №2
Основы написания драйверов устройств с использованием операционной системы

вариант: 1

Выполнил: Галиуллин Рашит Дамирович

Группа: P3334

Санкт-Петербург, 2025г

Цель

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

Задание

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

Вариант

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

Описание функций

my_write (обработчик записи):

Назначение:

Обрабатывает данные, записанные в устройство. Вычисляет длину полученной строки и сохраняет результат в буфер results.

Логика работы:

- Проверяет, что длина данных не превышает размер буфера (BUF_LEN - 1).
- Копирует данные из пользовательского пространства в ядро с помощью copy_from_user.
- Удаляет символ перевода строки \n в конце (если есть).
- Вычисляет количество символов в строке (без \n).
- Добавляет результат в формате "<длина_строки>\n" в буфер results, используя snprintf.
- Накопительный буфер: каждый вызов дописывает новую запись в конец results.

Пример:

При записи "Hello" → в results добавится строка "5\n".

Возвращаемое значение:

- 👍 При успехе: исходная длина данных (len).
- 👎 При ошибках: -EINVAL (неверная длина) или -EFAULT (ошибка копирования).

```c
static ssize_t my_write(struct file *f, const char __user *buf, size_t len, loff_t *off)
{
    char tmp[BUF_LEN];
    if (len > BUF_LEN - 1)
        return -EINVAL;

    if (copy_from_user(tmp, buf, len))
        return -EFAULT;

    tmp[len] = '\0';

    if (len > 0 && tmp[len - 1] == '\n') {
        tmp[len - 1] = '\0';
    }

    int chars = strlen(tmp);
    count += snprintf(results + count, BUF_LEN - count, "%d\n", chars);

    return len;
}
```

my_read (обработчик чтения):

Назначение:

Возвращает содержимое буфера results пользователю.

Логика работы:

- Использует simple_read_from_buffer для безопасного копирования данных из results в пользовательский буфер.
- Данные читаются с текущего смещения (off). После чтения смещение обновляется автоматически.
- Буфер results содержит все ранее записанные длины строк в формате:

5\n
10\n

Новые операции записи продолжают дополнять буфер results.

Возвращаемое значение:

Количество скопированных байт или код ошибки (если возникла).

```
static ssize_t my_read(struct file *f, char __user *buf, size_t len, loff_t *off)
{
    return simple_read_from_buffer(buf, len, off, results, strlen(results));
}
```

Вывод

В ходе выполнения лабораторной работы был разработан драйвер символьного устройства для ОС Linux. Драйвер успешно решает поставленную задачу. Полученный опыт формирует базу для создания более сложных модулей ядра.