BÀI 21: TRAO ĐỔI DỮ LIỆU USER - KERNEL

copy_to_user & copy_from_user

🎯 Mục tiêu bài học

1. **Vấn đề bộ nhớ:** Tại sao Kernel không được truy cập trực tiếp con trỏ của User? (User Space vs Kernel Space).
2. **Giải pháp:** Sử dụng hàm `copy_to_user` và `copy_from_user`.
3. **An toàn:** Kiểm tra tính hợp lệ của con trỏ để tránh Crash hệ thống.

1. Ranh giới User - Kernel

Linux chia bộ nhớ ảo làm 2 phần:

- **User Space (0 - 3GB):** Nơi ứng dụng chạy. Có thể bị swap ra ổ cứng.
- **Kernel Space (3GB - 4GB):** Nơi Kernel chạy. Luôn nằm trong RAM vật lý.

> **Nguy hiểm:** Nếu Driver truy cập trực tiếp địa chỉ của User (ví dụ `0x1234`), có thể địa chỉ đó đang không tồn tại trong RAM (Page Fault) -> Gây **Kernel Panic** (Treo máy).

2. Hàm `copy_from_user`

Dùng trong hàm `my_write` của Driver.

```
// Lấy dữ liệu từ User App chép vào Kernel Buffer
unsigned long copy_from_user(void *to, const void __user *from, unsigned long n);
```

- **to:** Con trỏ bộ nhớ Kernel (đích).
- **from:** Con trỏ bộ nhớ User (nguồn).
- **n:** Số byte cần chép.
- **Return:** Số byte KHÔNG chép được (0 là thành công).

3. Hàm `copy_to_user`

Dùng trong hàm `my_read` của Driver.

```
// Lấy dữ liệu từ Kernel Buffer trả về cho User App
unsigned long copy_to_user(void __user *to, const void *from, unsigned long n);
```

> **Lưu ý:** Hai hàm này có cơ chế kiểm tra lỗi bộ nhớ và có thể "ngủ" (sleep) nếu dữ liệu user đang nằm ở ổ cứng (swap). Do đó, **không được dùng trong Interrupt Handler**.

🛠️ PHẦN THỰC HÀNH (LAB 21)

Driver lưu trữ dữ liệu (Echo Driver)

Yêu cầu

1. Cải tiến driver bài 20.
2. Khai báo một bộ đệm trong Kernel: `char kernel_buf[1024]` .
3. **Hàm Write:** Khi User ghi chuỗi "Hello", driver copy vào `kernel_buf` .
4. **Hàm Read:** Khi User đọc, driver trả `kernel_buf` ngược lại cho User.
5. **Test:**

```
echo "Hello Linux" > /dev/dummy_driver
cat /dev/dummy_driver  # Kết quả mong đợi: "Hello Linux"
```

Q & A

Hẹn gặp lại ở Bài 22: GPIO Driver!