SYSTEM SOFTWARE
DEVICE DRIVER

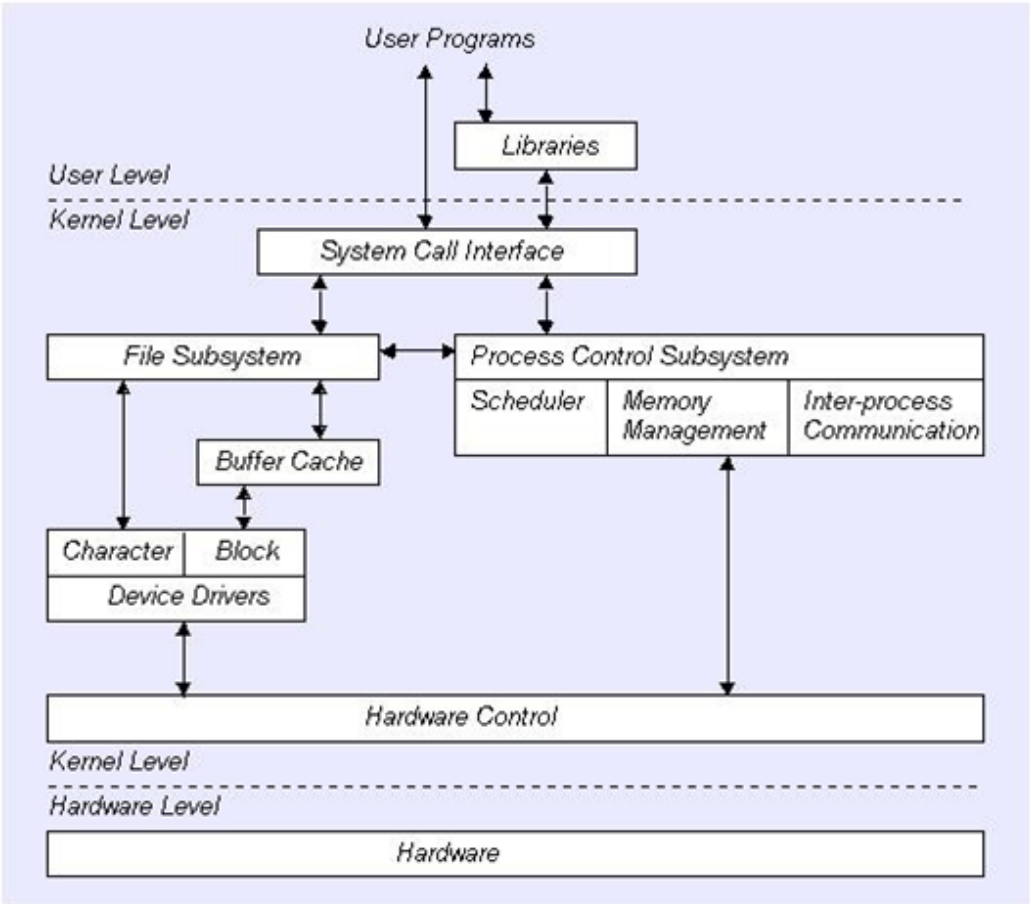
Nguyen Huu Duc

School of Information and Communication Technology
Hanoi University of Science and Technology

Overview of device driver

- Is a kind of system software
- Control peripherals connecting to a computer (Initialize, handle interrupt, exchange data)
- Communicate with users (read, write, ioctl)
- Device driver in UNIX/Linux
  - Each device is seen as a file by users
  - Each device has a driver which is built as a kernel module (LKM - Loadable Kernel Module)
  - Two main device type
    - Block device (HDD, CD-ROM,...)
    - Character device (keyboard, modem, printer,...)

Architecture



Loadable Kernel Module (LKM)

- Device driver
- File system driver (ext2, ext3, MSDOS FAT32,...)
- System call
- Network devices driver
- tty driver
- Interpreter

Loadable Kernel Module (LKM)

- Each LKM has at least two functions
  - `int init_module(void) ...`
  - `int cleanup_module(void) ...`

Example

```
#include <linux/init.h>
#include <linux/module.h>
MODULE_LICENSE("Dual BSD/GPL");
static int hello_init(void)
{
    printk(KERN_ALERT "Hello, world\n");
    return 0;
}
static void hello_exit(void)
{
    printk(KERN_ALERT "Goodbye, cruel world\n");
}
module_init(hello_init);
module_exit(hello_exit);
```

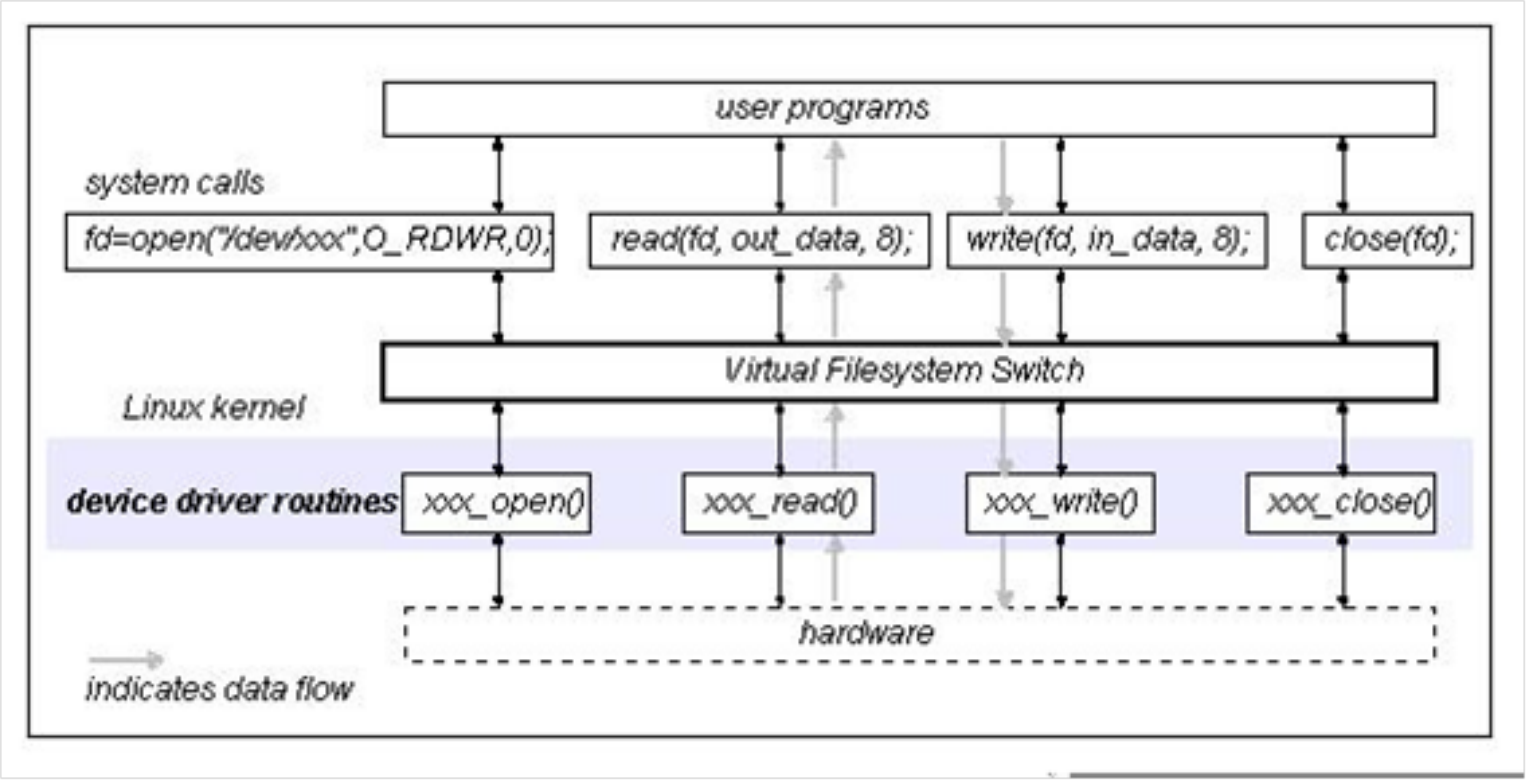
Add and delete modules

```
% make
make[1]: Entering directory ‘/usr/src/linux-2.6.10’
  CC [M]  /home/ldd3/src/misc-modules/hello.o
  Building modules, stage 2.
  MODPOST
  CC      /home/ldd3/src/misc-modules/hello.mod.o
  LD [M]  /home/ldd3/src/misc-modules/hello.ko
make[1]: Leaving directory ‘/usr/src/linux-2.6.10’
% su
root# insmod ./hello.ko
Hello, world
root# rmmod hello
Goodbye cruel world
root#
```

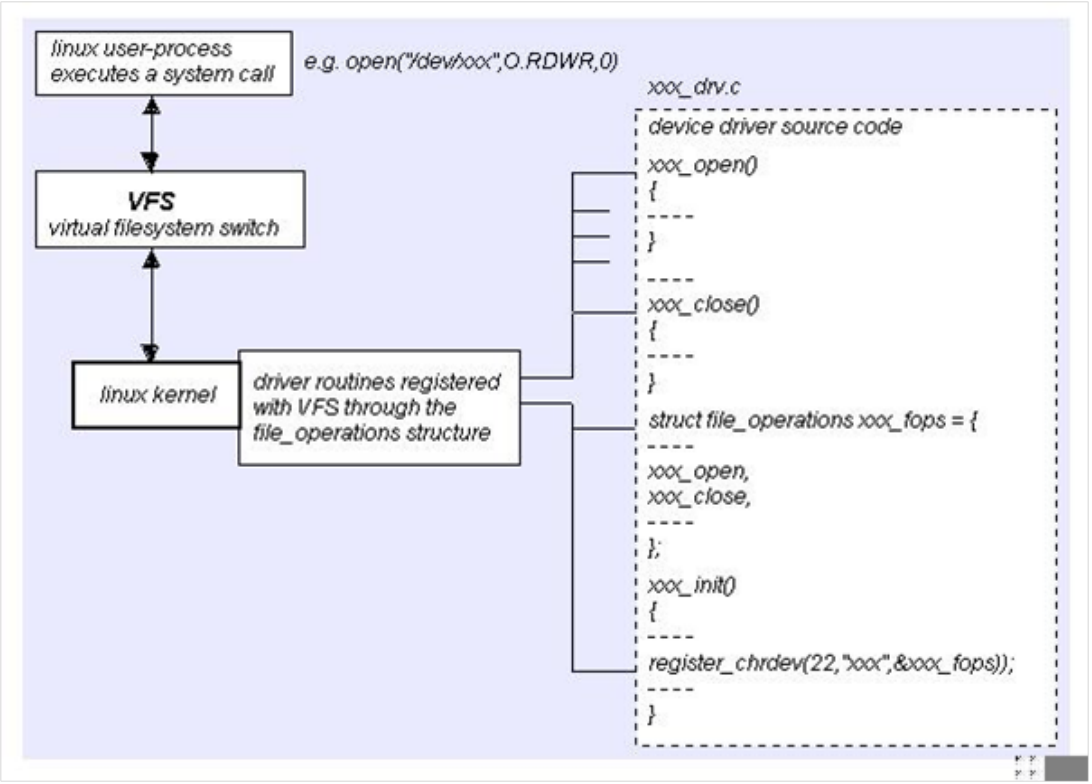
Device driver

- Is a module packaging a set of API which is used to communicate with devices
- Hidden hardware operation via file
  - Each device is referred to a file of a form of `/dev/device`
- Coordinate data flows between user and device

Device driver interface



Interface between device driver and kernel

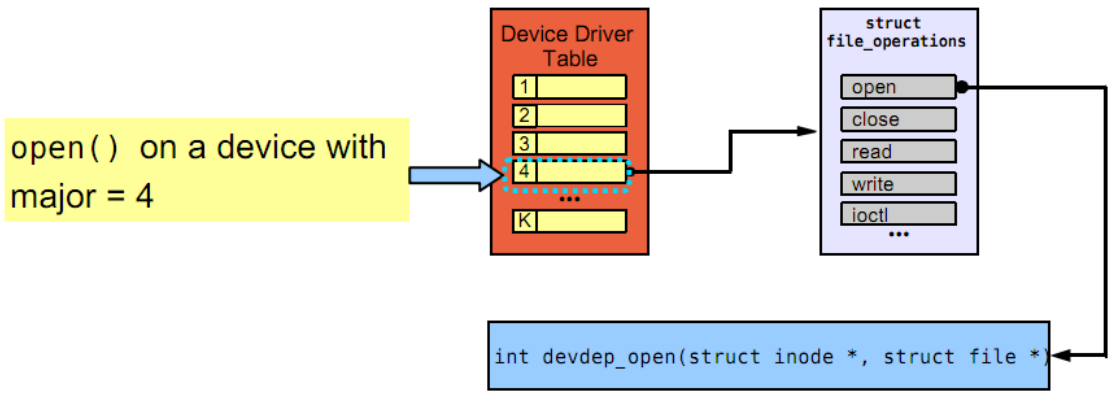


Structure of file_operations

```
// Each component is for a service
struct file_operations {
    int (*lseek) ( );
    int (*read) ( );
    int (*write) ( );
    int (*ioctl) ( );
    int (*open) ( );
    void (*release) ( );
    ....
};
```

Major/Minor number

- Major number is used to select a device driver
  - Is an index of the device driver in the device driver table.
- Minor number is used to select a device which is handled by the device driver



Register and unregister a device

```
int init_module(void) /*used for all initialition stuff*/
{
 /* Register the character device (atleast try) */
 Major = register_chrdev(0,DEVICE_NAME,&Fops);
  ...
}
void cleanup_module(void) /*used for a clean shutdown*/
{
 ret = unregister_chrdev(Major, DEVICE_NAME);
  ...
}
```

Register

- Compile

```
-Wall -DMODULE -D__KERNEL__ -DLINUX -DDEBUG
-I /usr/include/linux/version.h
-I/lib/modules/‘uname -r‘/build/include
```

- Install: `insmod module.o`
- List: `lsmod`
- Find major number: `more /proc/devices`
- Create a device file:

```
mknod /dev/device_name c major minor
```

Basic structure

- `xxx_init()`: Initialize a device
- `xxx_open()`: Open a device
- `xxx_read()`: Read from a device
- `xxx_write()`: Write to a device
- `xxx_release()`: Close a device
- `init_module()`: Initialize (load module)
- `cleanup_module()`: Finalize (unload module)