Pseudo Char Driver:-

Every driver is a module , but not vice versa

Char Devices

Block Devices

Device Driver:-

* Char Drivers
* Block Drivers
* Network Drivers
* Misc / sub systems

Device Special Files (Device Node Files):-

Interfacing drivers with userspace

```
ls /dev
```

```
/dev/ttyS0        ==> Regular UART Driver
/dev/ttyUSB0    ==> USB-UART
/dev/i2c-dev
/dev/spidev
```

Books:-

* Linux Device Drivers (LDD), 3/e, by Rubini
* Linux Kernel Development(LKD), 3/e, Robert Love

```
ls -l /dev/ttyS0
ls -l /dev/lp0

ls -l /dev/sda*        # Internal HDD, SATA
ls - /dev/sdb*         # USB Storage/Pen Drive
ls -l /dev/mmcblk0     # SD Card
```

First letter in "ls -l" output

```
stat /dev/ttyS0
stat /dev/sda1
```

Device ID   ==> Major number + Minor number

```
cat /proc/devices
```

Activity:-

* Driver code upto Step-3
* User space code
* System call impl
* Pre-read list and kfifo APIs
(for list links given in Yammer)

```
cscope - vi
      :q

alloc_chrdev_region
copy_to_user
cdev_init

struct task_struct {
struct file_operations {
struct inode {

sched.h
```

memset, memcpy, memcmp, bzero

```
fd=open("/dev/psample", O_RDWR);
if(fd<0) {
        perror("open");
}

char str[]="abcdxyz";
nbytes=write(fd,str,7);
if(nbytes<0) {
        perror("write");
}

char buf[64];
int maxlen=64;
nbytes=read(fd,buf,maxlen);
if(nbytes<0) {
        perror("write");
}

write(1,buf,maxlen); (or) buf[nbytes]='\0'; puts(buf);

close(fd);
```

MAJOR, MINOR, MAKEDEV

Step-4:-

class_create
alloc_chrdev_region

kmalloc

cdev_init
kobject_set_name
cdev_add

device_create

----

write:-
1) wr_offset : 1024
2) wr_offset : 1000
    user req : 40
3) wr_offset : 900
    user_req : 40

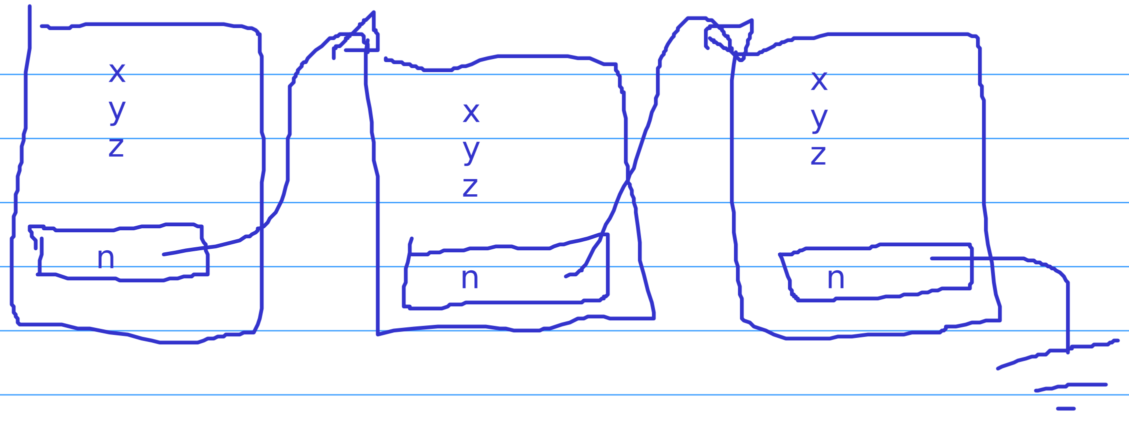
read:-
1) buflen : 0
2) buflen : 10, user : 15
3) buflen : 20, user : 12

wr_offset ==> tail/rear
rd_offset ==> head/front

```
fd=open("/dev/psample", O_RDWR);

char wbuf[32];
//fill wbuf with some data, hint:- memset
nbytes=write(fd, wbuf, len);
//fill wbuf with some data, hint:- memset
nbytes=write(fd, wbuf, len); //Assume 36 bytes

char rbuf[32]; int maxlen=10;

while(1) {
     nbytes=read(fd, rbuf, maxlen);
     //err hadnling
     //print rbuf
     if(nbytes==0) break;
}

close(fd);
```

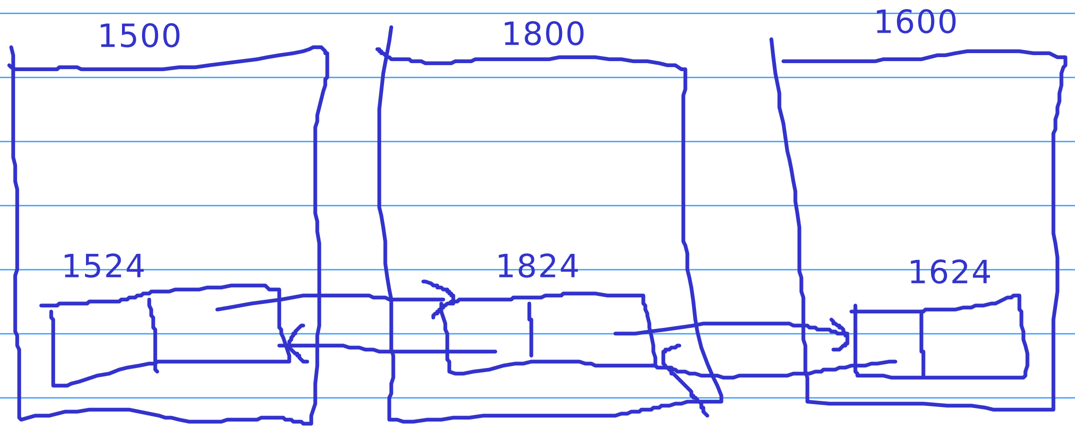
Normal Approach:-

```
struct sample {
   int x;      //+0
   int y;      //+4
   int z;      //+8
   struct sample *pnext; //+12
};
```



Kernel Approach:-

```
struct list_head {
   struct list_head *prev;
   struct list_head *nex;
};
struct sample {
   int x;
   int y;
   int z;
   struct list_head entry;
};
```



container_of macro
==> base address of structure
as per address of member

stddef.h:-

```
offsetof(struct sample, y)
offsetof(struct sample, z)
```

list_add_tail
list_for_each
list_for_each_safe
list_for_each_entry
list_for_each_entry_safe

Activity:-
* Step-4, Step-5 of pseudo driver (plain buffer, kfifo)
* Userspace code
* List API demo & simple example
* Basic IOCTL example (Driver + Userspace code)