驱动相关的一些数据结构

**一、背景介绍**

在驱动程序内，有下面几个重要的数据结构必须要了解。

file\_operations：将设备和对设备的操作关联起来。主要由很多函数指针构成。

file：对应每个打开的文件，主要由文件的属性描述构成。

inode：描述内核中的文件，具体的文件，关联到具体的设备。

cdev：字符型设备。指针存储在inode结构中。

**二、知识点：**

1.dev\_id

设备id，其实是个32bit的数，高12bit存储主设备号，低20位存储次设备号。在<linux/types.h>中声明。

2.申请和释放设备id

以下两个函数在<linux/fs.h>中声明。

alloc\_chrdev\_region(dev\_t \*dev, 0, int minor\_Count,char \*DeviceName);

unregister\_chrdev\_region(dev\_t dev, int minorCount);

设备注册成功之后，使用cat /proc/devices命令查看。

然后可在/dev目录下创建对应的设备节点。

3.主设备号和次设备号

主设备号对应驱动程序，

次设备号对应该驱动具体实现的设备。

MAJOR(dev\_t dev); //可从设备id中析取出主设备号。

MINOR(dev\_t dev); //可从设备id中析取出次设备号。

4.填充设备

注册到dev\_id之后，就要实现设备，并让设备完成具体的功能，即处理资源，并提供足够的接口让用户层程序可以操作。此部分后面依次介绍。

5.让设备活起来