

块设备驱动

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149



块设备驱动程序的介绍

块设备驱动框架

块设备驱动编写

块设备实例之用内存模拟

测试

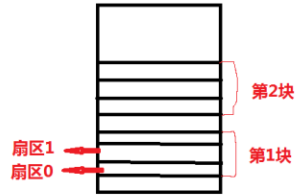
块设备驱动程序的引入

对于块设备，不能像字符设备那样直接提供读函数、写函数。

为什么呢？



块设备驱动程序的介绍



假设: 写扇区0和扇区1

原先是这样写扇区0:

原先是这样写扇区1:

a. 读出整块到Buffer

a. 读出整块到Buffer

b. 修改Buffer里的扇区0

b. 修改Buffer里的扇区1

c. 擦除整块

c. 擦除整块

d. 烧写整块

d. 烧写整块

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备驱动程序的引入

优化后是这样写扇区0/1

1. 读出整块
2. 修改两个扇区(0和1)
3. 擦除整块
4. 烧写整块

这样是不是减少了很多重复的动作呢？

宗旨就是: 先不执行，放入队列，优化后再执行

块设备驱动程序就是扮演优化后顺序在读写块设备的角色
所以，块设备驱动和字符设备驱动并不一样。

块设备驱动程序的框架

应用程序: **open/read/write "1.txt"**

----- 文件的读写
文件系统: vfat, ext2, ext3, yaffs2, jffs2 (把文件的读写转换为扇区的读写)

----- **ll_rw_block** ----- 扇区的读写

1. 把"读写"放入队列
2. 调用队列的处理函数(优化/调顺序/合并)

块设备驱动程序

硬件: 硬盘, flash

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备驱动程序的框架

```
分析ll_rw_block
    for (i = 0; i < nr; i++) {
        struct buffer_head *bh = bhs[i];
        submit_bh(rw, bh);
        struct bio *bio; // 使用bh来构造bio (block input/output)
        submit_bio(rw, bio);
        // 通用的构造请求: 使用bio来构造请求(request)
        generic_make_request(bio);
        __generic_make_request(bio);
        request_queue_t *q = bdev_get_queue(bio->bi_bdev); // 找到队列
        // 调用队列的"构造请求函数"
        ret = q->make_request_fn(q, bio);
        // 默认函数是__make_request
        __make_request
        // 先尝试合并
        elv_merge(q, &req, bio)
        // 如果合并不成, 使用bio构造请求
        init_request_from_bio(req, bio);
        // 把请求放入队列
        add_request(q, req);

        // 执行队列
        __generic_unplug_device(q);
        // 调用队列的"处理函数"
        q->request_fn(q);
    }
```

这是在内核2.6.22.6分析的, 不同版本的内核稍有差异, 但大体步骤是一样的

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备驱动程序的编写步骤

A. 使用`alloc_disk`分配`gendisk`结构体

B. 设置`gendisk`

B.1 `blk_init_queue`(分配/设置)队列`request_queue_t`

B.2 设置`gendisk`其他信息 // 它提供属性: 如容量

C. 注册: `add_disk`

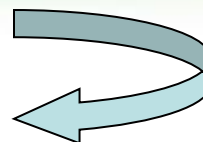
电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备实例之用内存模拟硬盘

1. 设置分配一个gendisk结构体



```
ramblock_disk = alloc_disk(16) //次设备号个数/分区个数+1
```

/* 2. 分配/设置队列: 提供读写能力 */

```
ramblock_queue = blk_init_queue(do_ramblock_request, &ramblock_lock);  
ramblock_disk->queue = ramblock_queue;
```

do_ramblock_request: 是一个void函数指针，用它来处理请求

ramblock_lock: 是一个自旋锁，用于对临界资源的保护

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

do_ramblock_request

1. 设置数据传输的3要素：目的/源 + 长度
2. 用电梯调度算法处理读写请求
3. 如果有请求：先判断是读还是写？，然后在调用 **memcpy** 进行数据的拷贝

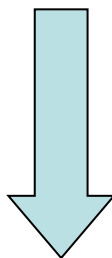
电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备实例之用内存模拟硬盘

3. 设置其他属性: 比如容量



```
set_capacity(ramblock_disk, RAMBLOCK_SIZE  
/ 512)
```

RAMBLOCK_SIZE : 分配的大小,单位是扇区

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备实例之用内存模拟硬盘

4. 设置gendisk的操作函数

```
ramblock_disk->fops      = &ramblock_fops;
```

设置gendisk的主次设备号

```
ramblock_disk->major      = major;  
ramblock_disk->first_minor = 0;
```

major 是register_blkdev注册后的返回值

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

块设备实例之用内存模拟硬盘

3. 分配并注册

```
ramblock_buf=kzalloc(RAMBLOCK_SIZE,GFP_  
KERNEL)
```

```
add_disk(ramblock_disk);
```

ramblock_buf : unsigned char的指针

GFP_KERNEL : 分配的标志，一般是这个

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149

测试

在开发板上:

1. insmod ramblock.ko
2. 格式化: mkdosfs /dev/ramblock
3. 挂接: mount /dev/ramblock /tmp/
4. 读写文件: cd /tmp, 在里面vi文件
5. cd /; umount /tmp/
6. cat /dev/ramblock > /mnt/ramblock.bin
7. 在PC上查看ramblock.bin
sudo mount -o loop ramblock.bin /mnt

电话: 0755-86200561 淘宝地址: 100ask.taobao.com

地址: 广东省深圳市龙岗区布吉中海怡翠山庄13-2-2B

版权所有: 深圳百问网科技有限公司, 群84174029 28664149