指导

16.wait\_on\_buffer函数中为什么不用if（）而是用while（）？

1、getblk函数中，申请空闲缓冲块的标准就是b\_count为0，而申请到之后，为什么在wait\_on\_buffer(bh)后又执行if（bh->b\_count）来判断b\_count是否为0？

2、b\_dirt已经被置为1的缓冲块，同步前能够被进程继续读、写？给出代码证据。

1. 分析panic函数的源代码，根据你学过的操作系统知识，完整、准确的判断panic函数所起的作用。假如操作系统设计为支持内核进程（始终运行在0特权级的进程），你将如何改进panic函数？

6、操作系统如何利用b\_uptodate保证缓冲块数据的正确性？new\_block (int dev)函数新申请一个缓冲块后，并没有读盘，b\_uptodate却被置1，是否会引起数据混乱？详细分析理由。

7、add\_reques（）函数中有下列代码

if (!(tmp = dev->current\_request)) {

dev->current\_request = req;

sti();

(dev->request\_fn)();

return;

}

其中的

if (!(tmp = dev->current\_request)) {

dev->current\_request = req;

是什么意思？

8、do\_hd\_request()函数中dev的含义始终一样吗？

9、read\_intr（）函数中，下列代码是什么意思？为什么这样做？

if (--CURRENT->nr\_sectors) {

do\_hd = &read\_intr;

return;

}

10、bread（）函数代码中为什么要做第二次if (bh->b\_uptodate)判断？

if (bh->b\_uptodate)

return bh;

ll\_rw\_block(READ,bh);

wait\_on\_buffer(bh);

if (bh->b\_uptodate)

return bh;

11、getblk（）函数中，两次调用wait\_on\_buffer（）函数，两次的意思一样吗？

12、getblk（）函数中

do {

if (tmp->b\_count)

continue;

if (!bh || BADNESS(tmp)<BADNESS(bh)) {

bh = tmp;

if (!BADNESS(tmp))

break;

}

/\* and repeat until we find something good \*/

} while ((tmp = tmp->b\_next\_free) != free\_list);

说明什么情况下执行continue、break。

13、make\_request（）函数

if (req < request) {

if (rw\_ahead) {

unlock\_buffer(bh);

return;

}

sleep\_on(&wait\_for\_request);

goto repeat;

其中的sleep\_on(&wait\_for\_request)是谁在等？等什么？

为什么 static inline \_syscall0(type,name)中需要加上关键字 inline？