第五章

设备管理

主 要 内 容

5.1 I/O硬件系统

5.2 I/O软件系统

5.3 磁盘存储器管理

5.4 UNIX字符块设备管理

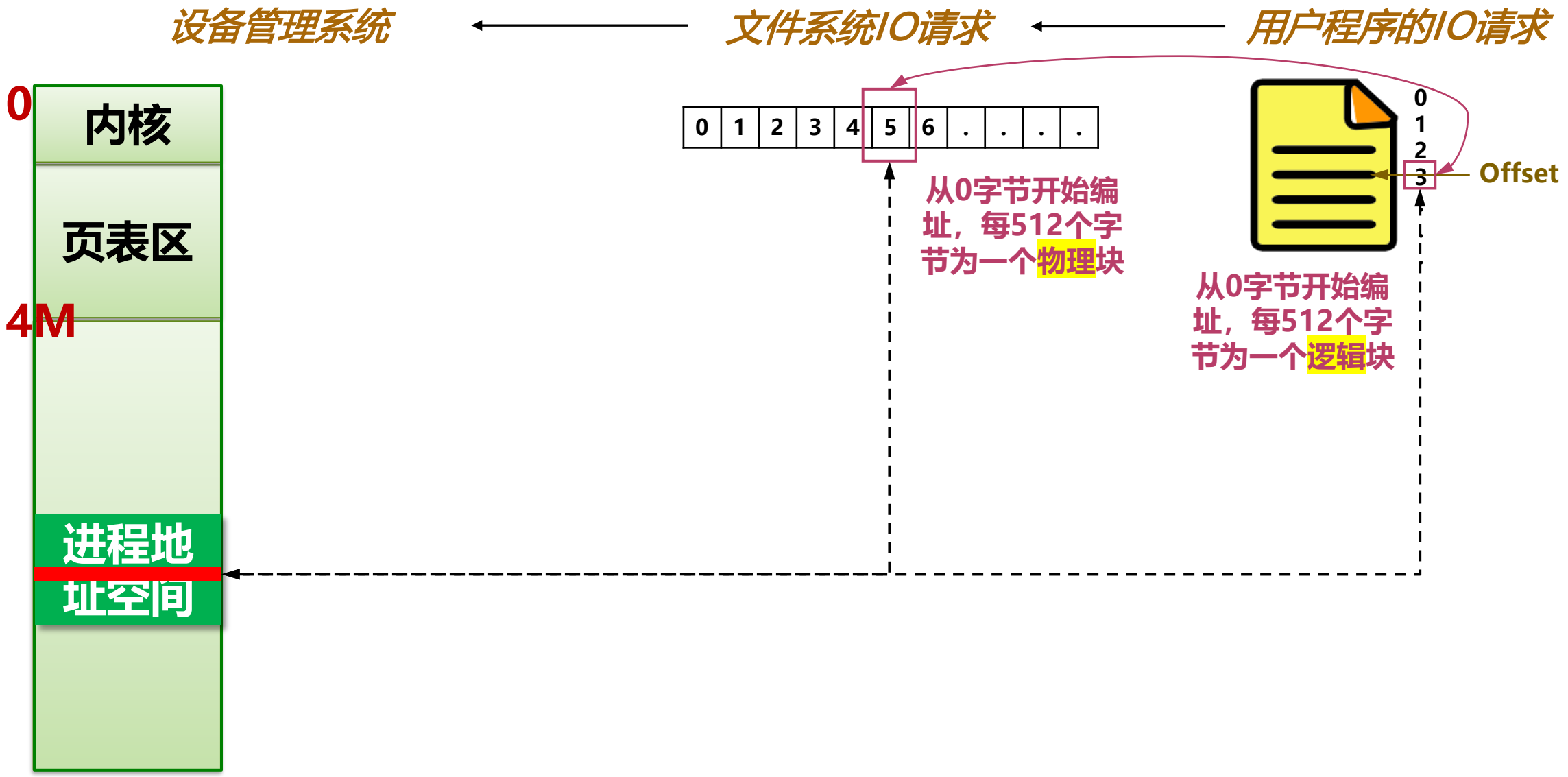
- 缓存管理
- 块设备读写过程

UNIX块设备的读写过程

通过缓存完成读写操作

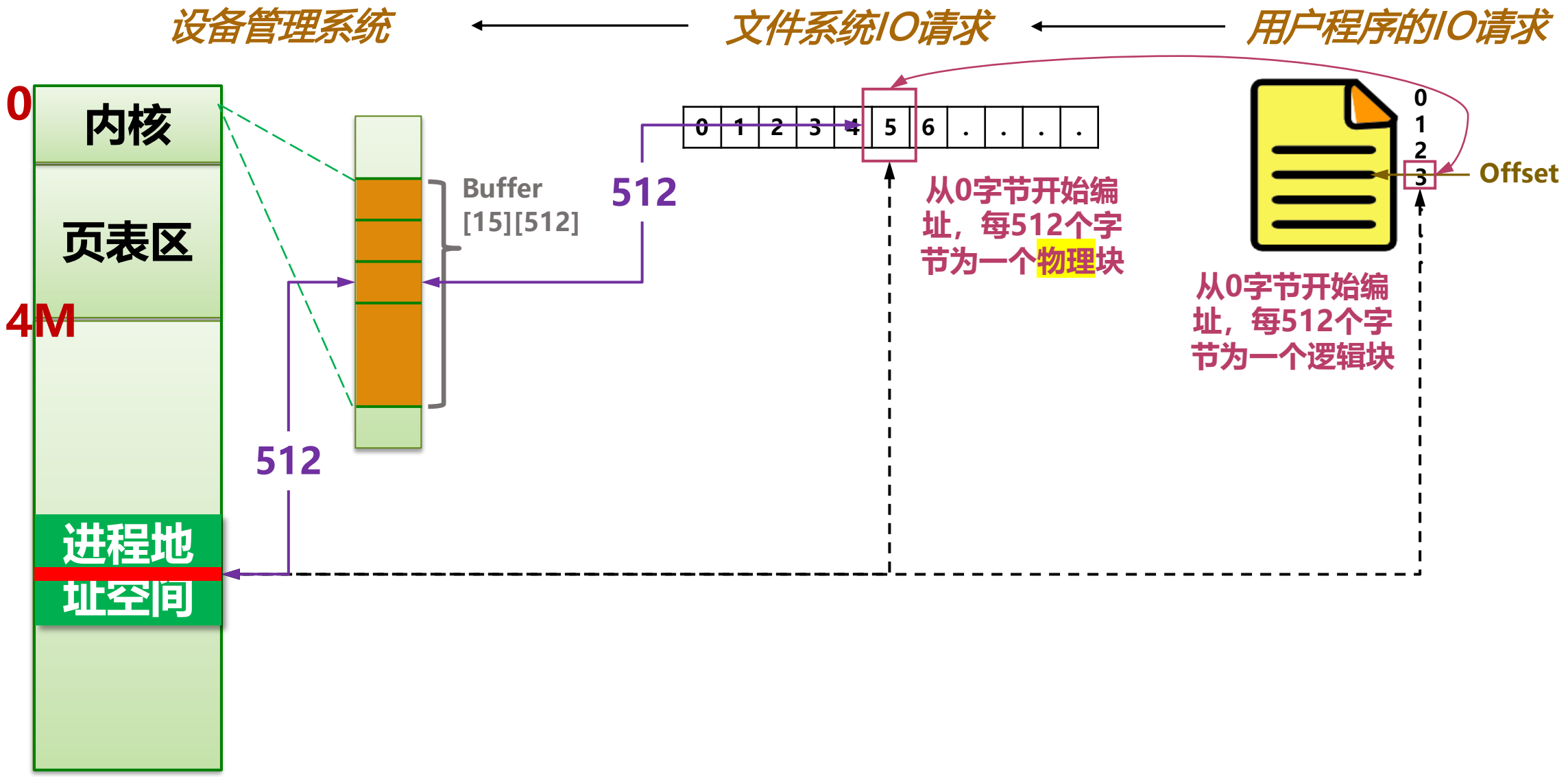
UNIX块设备的读写过程

通过缓存完成读写操作



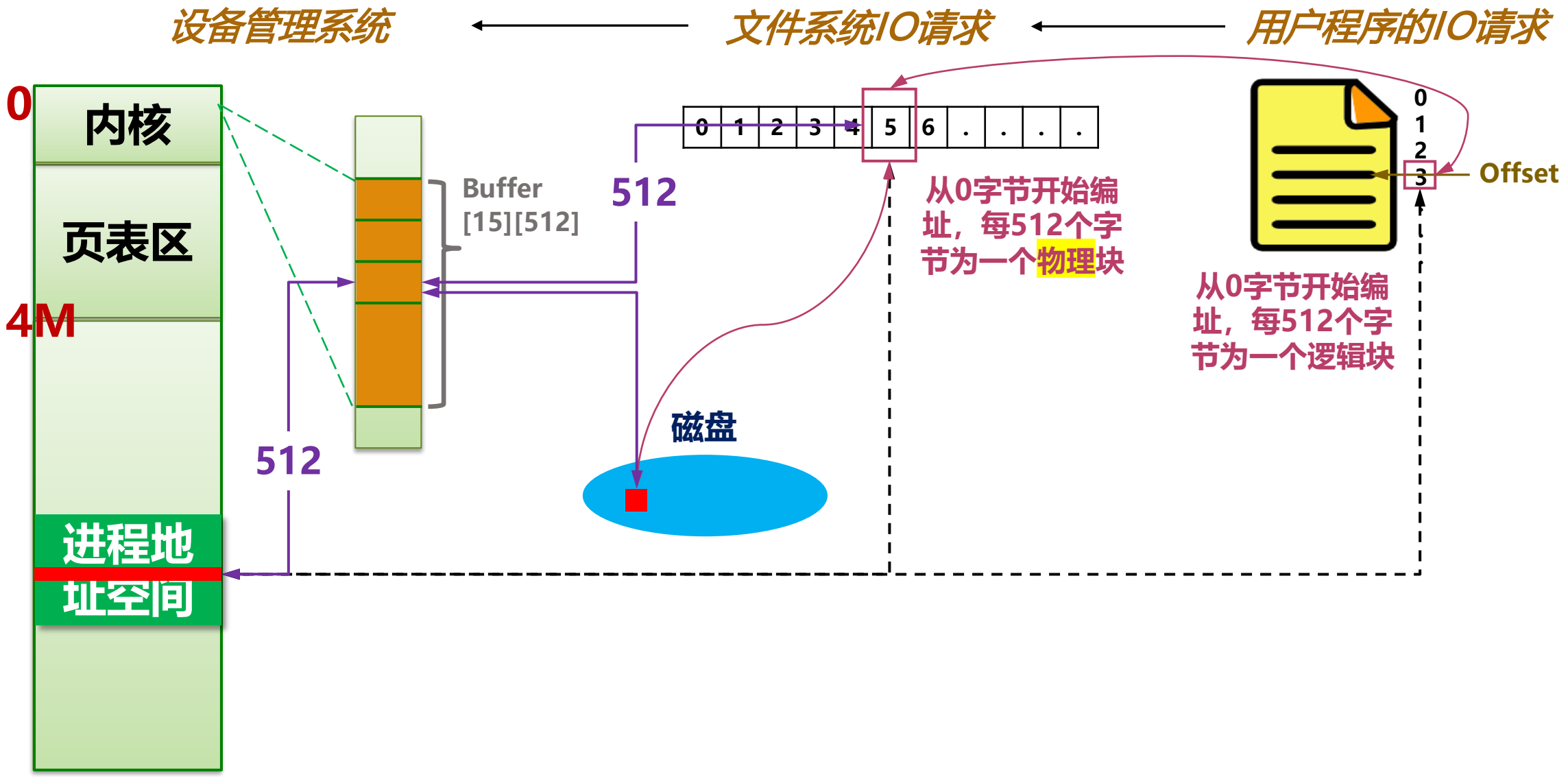
UNIX块设备的读写过程

通过缓存完成读写操作



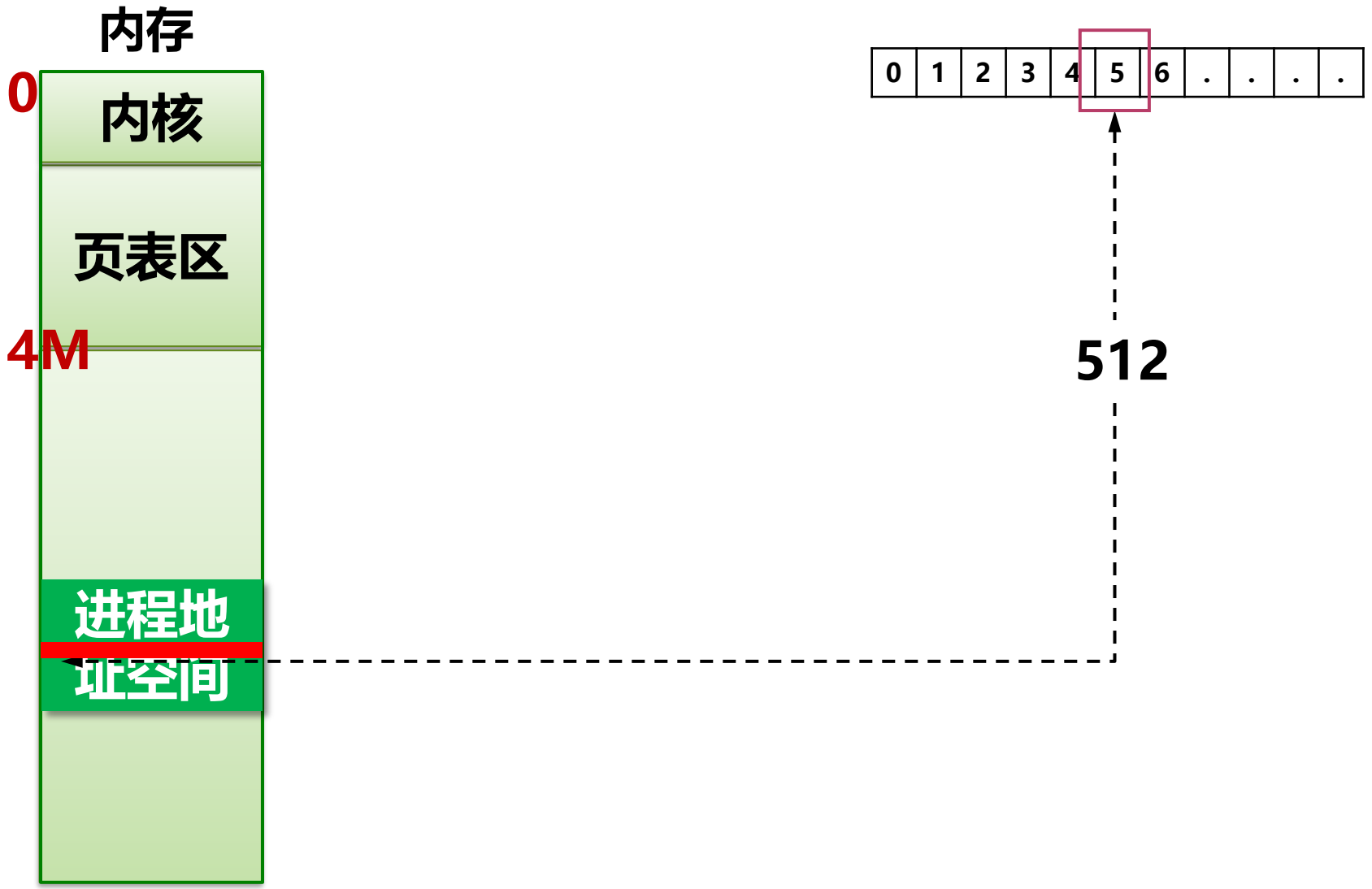
UNIX块设备的读写过程

通过缓存完成读写操作



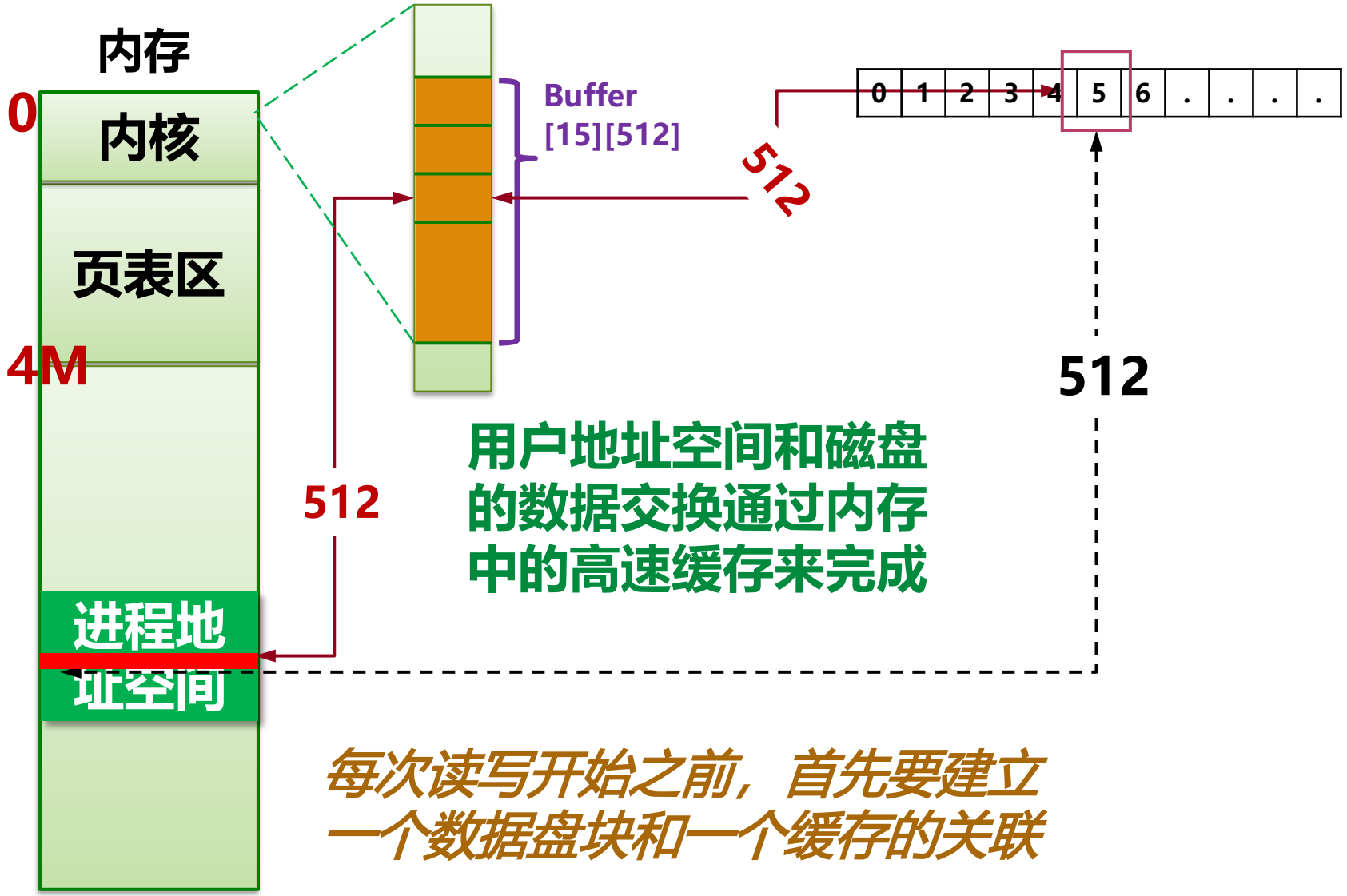
UNIX块设备缓存管理

通过缓存完成读写操作



UNIX块设备缓存管理

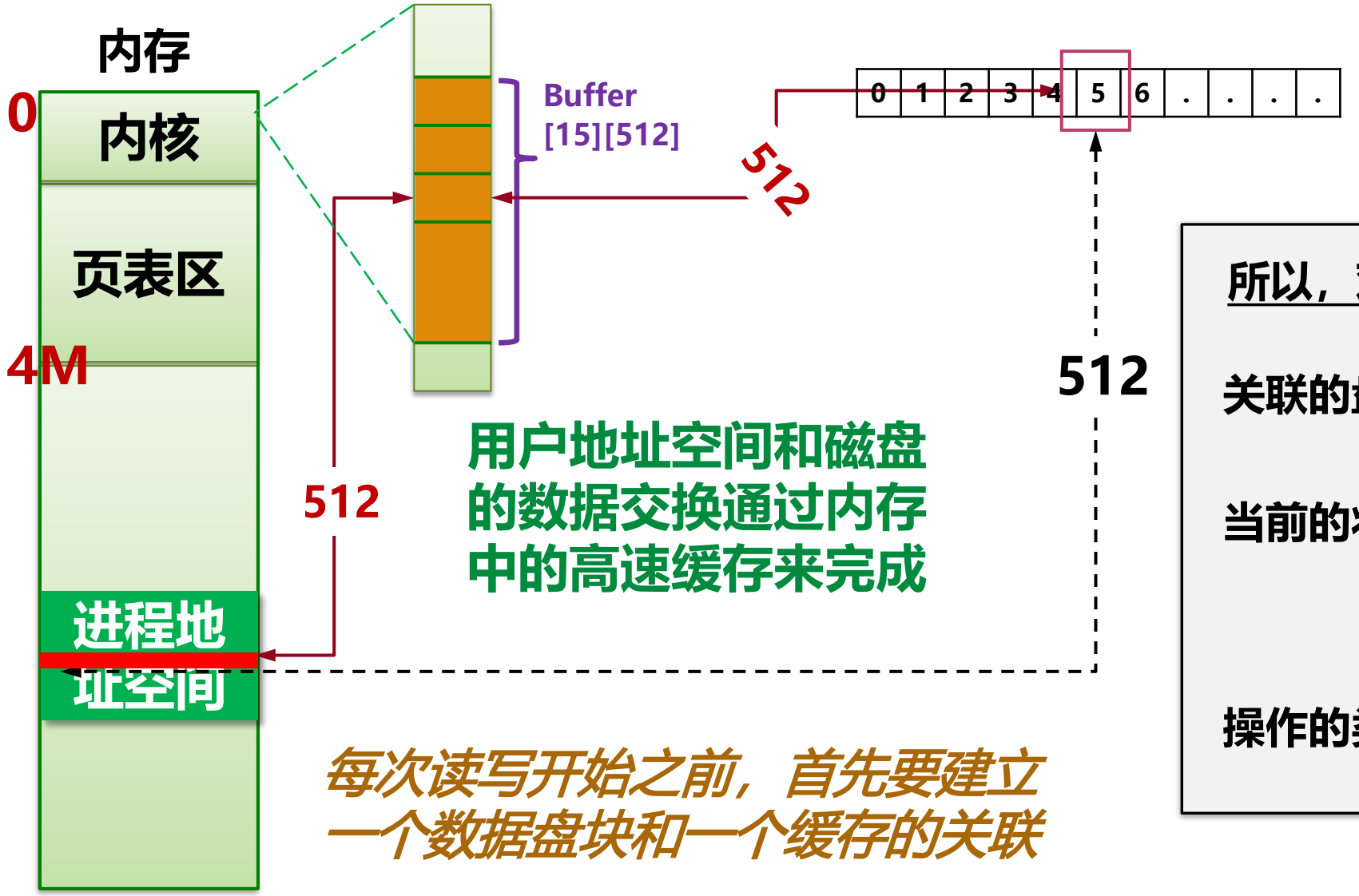
通过缓存完成读写操作



每次读写开始之前，首先要建立一个数据盘块和一个缓存的关联

UNIX块设备缓存管理

通过缓存完成读写操作



所以，对每一块缓存，需记录：

关联的盘块：　盘块地址

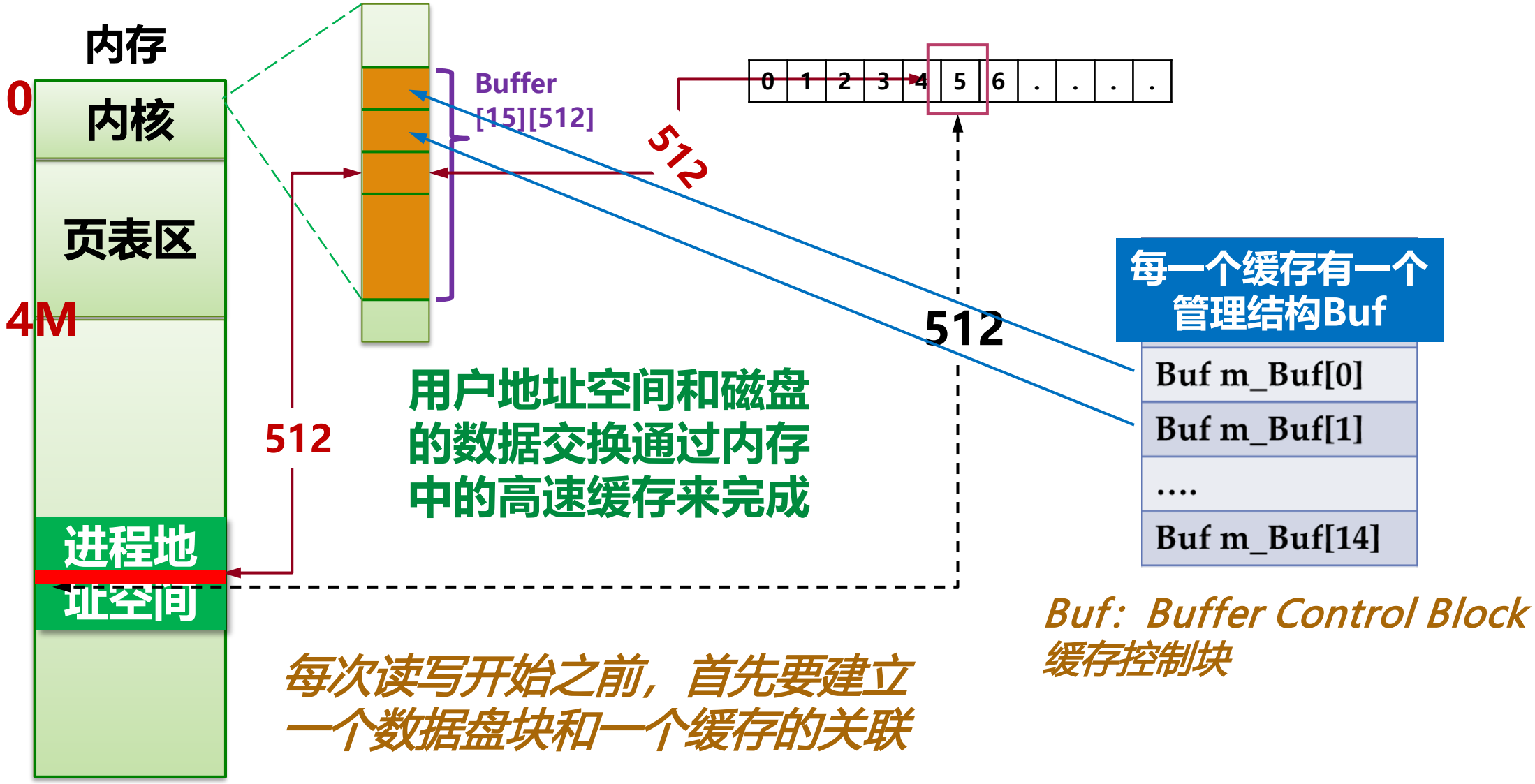
当前的状态：　忙？闲？
闲了多久？

操作的类型：　读？　写？

每次读写开始之前，首先要建立一个数据盘块和一个缓存的关联

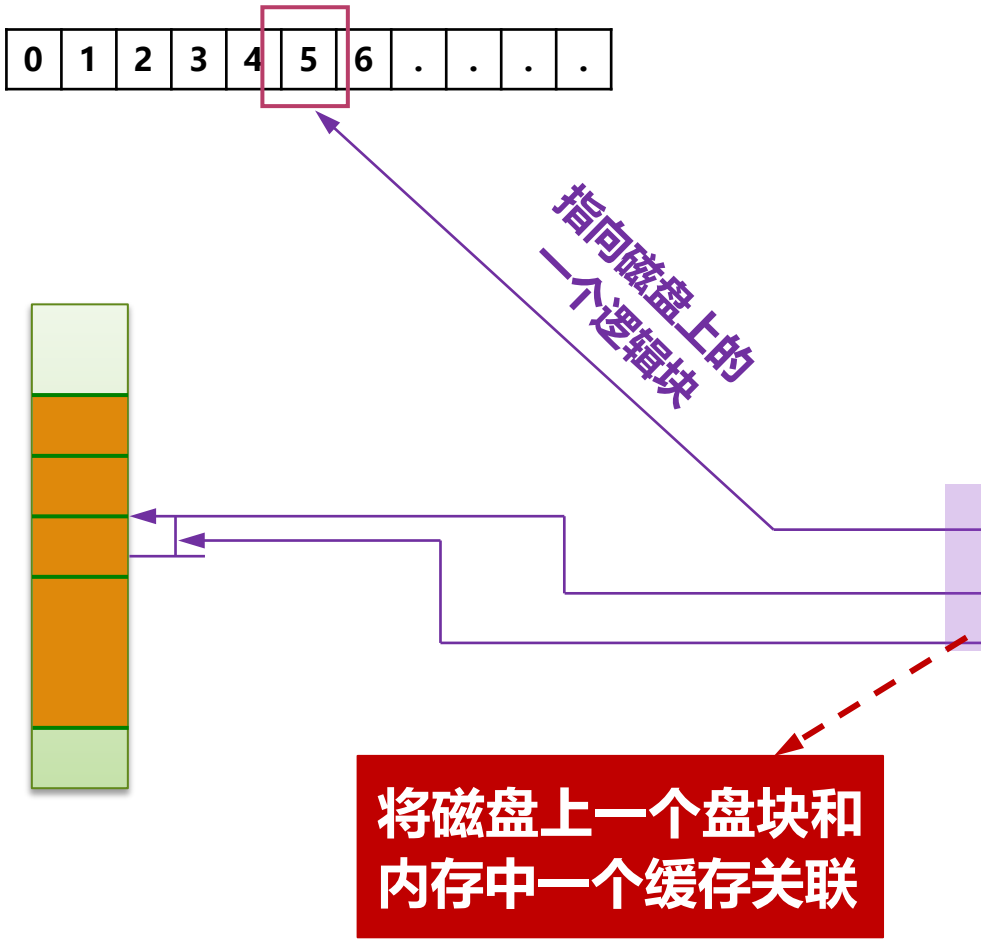
UNIX块设备缓存管理

通过缓存完成读写操作



UNIX块设备缓存管理

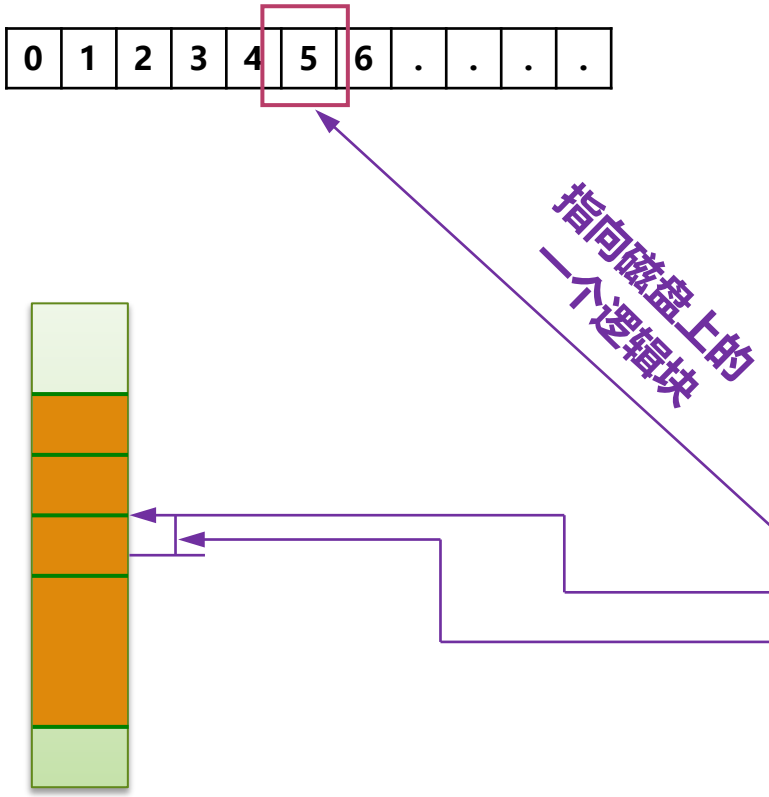
缓存控制块




```
class Buf
{
public:
    enum BufFlag              /* b_flags中标志位 */
    {
        B_WRITE = 0x1,        /* 写操作。将缓存中的信息写到硬盘上去 */
        B_READ= 0x2,          /* 读操作。从盘读取信息到缓存中 */
        B_DONE= 0x4,          /* I/O操作结束 */
        B_ERROR= 0x8,         /* I/O因出错而终止 */
        B_BUSY= 0x10,         /* 相应缓存正在使用中 */
        B_WANTED = 0x20,      /* 有进程正在等待使用该buf管理的缓存*/
        B_ASYNC= 0x40,        /* 异步I/O，不需要等待其结束 */
        B_DELWRI = 0x80       /* 延迟写 */
    };
public:
    short b_dev;              /* 高、低8位分别是主、次设备号 */
    int b_blkno;              /* 磁盘逻辑块号 */
    unsigned char* b_addr;    /* 指向该缓存控制块管理的缓冲区首地址 */
    int b_wcount;             /* 需传送的字节数 */
    unsigned int b_flags;     /* 缓存控制块标志位 */
    int b_error;              /* I/O出错时信息 */
    int b_resid;              /* I/O出错时尚未传送的剩余字节数 */
    int padding;              /* 4字节填充，否则强制转换会出错。 */
    Buf* b_forw;              /* 缓存控制块队列勾连指针 */
    Buf* b_back;
    Buf* av_forw;
    Buf* av_back;
};
```

UNIX块设备缓存管理

缓存控制块



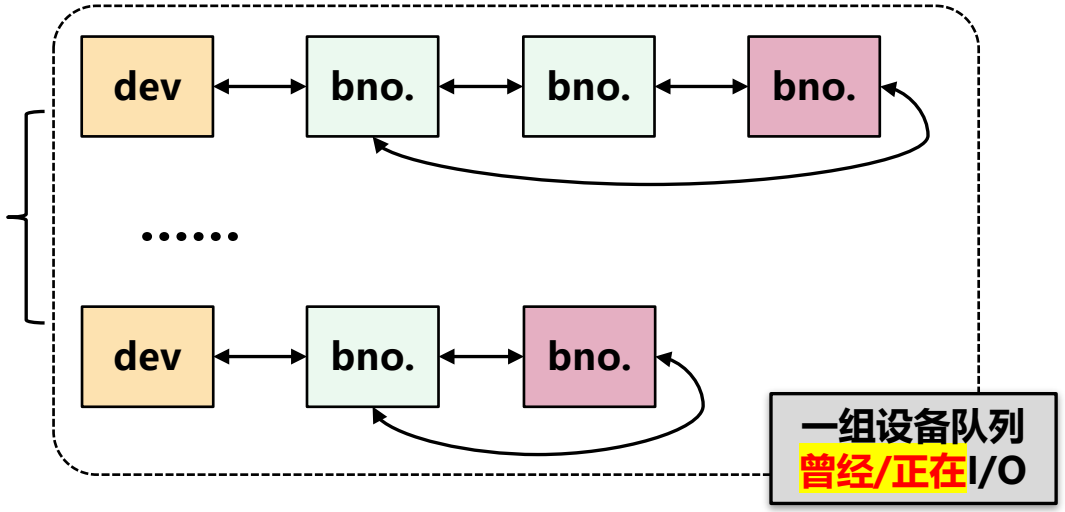

```
class Buf
{
public:
    enum BufFlag              /* b_flags中标志位 */
    {
        B_WRITE = 0x1,        /* 写操作。将缓存中的信息写到硬盘上去 */
        B_READ= 0x2,          /* 读操作。从盘读取信息到缓存中 */
        B_DONE= 0x4,          /* I/O操作结束 */
        B_ERROR= 0x8,         /* I/O因出错而终止 */
        B_BUSY= 0x10,         /* 相应缓存正在使用中 */
        B_WANTED = 0x20,      /* 有进程正在等待使用该buf管理的缓存*/
        B_ASYNC= 0x40,        /* 异步I/O，不需要等待其结束 */
        B_DELWRI = 0x80       /* 延迟写 */
    };
public:
    short b_dev;              /* 高、低8位分别是主、次设备号 */
    int b_blkno;              /* 磁盘逻辑块号 */
    unsigned char* b_addr;    /* 指向该缓存控制块管理的缓冲区首地址 */
    int b_wcount;             /* 需传送的字节数 */
    unsigned int b_flags;     /* 缓存控制块标志位 */
    int b_error;              /* I/O出错时信息 */
    int b_resid;              /* I/O出错时尚未传送的剩余字节数 */
    int padding;              /* 4字节填充，否则强制转换会出错。 */
    Buf* b_forw;              /* 缓存控制块队列勾连指针 */
```

7	6	5	4	3	2	1	0
B_DELWRI	B_ASYNC	B_WANTED	B_BUSY	B_ERROR	B_DONE	B_READ	B_WRITE

```
};
```

UNIX块设备缓存管理

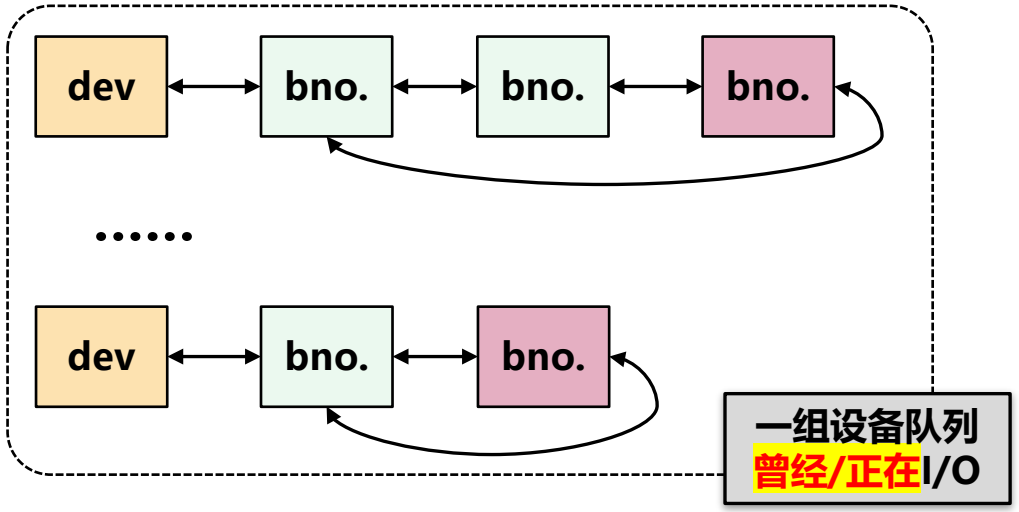
缓存队列



UNIX块设备缓存管理

缓存队列

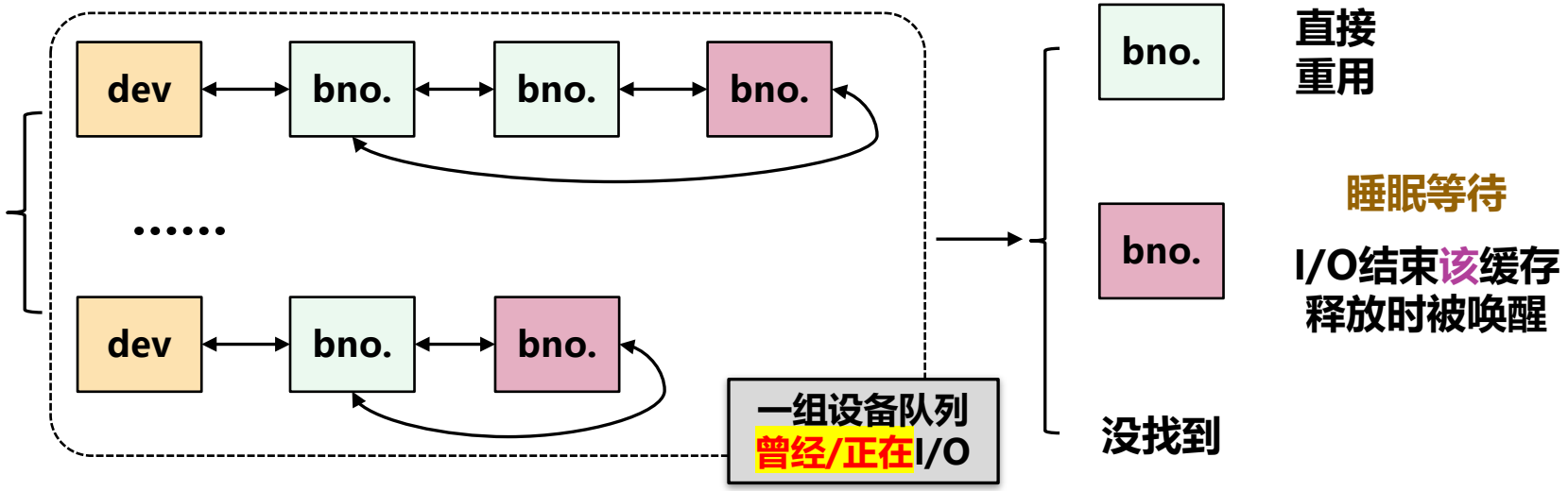
1 可以重用么？ *dev，blkno*



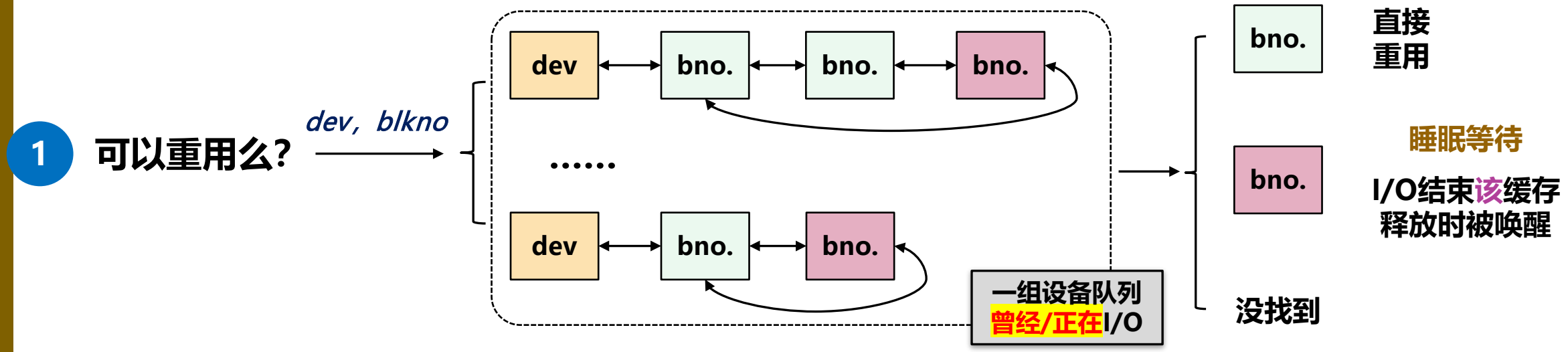
UNIX块设备缓存管理

缓存队列

1 可以重用么？ *dev，blkno*

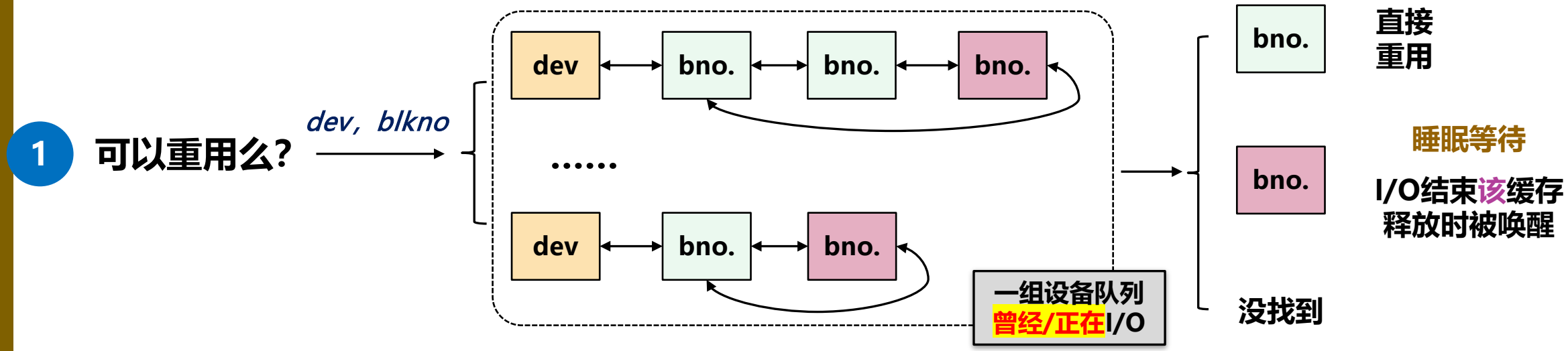


UNIX块设备缓存管理



	7 B_DELWRI	6 B_ASYNC	5 B_WANTED	4 B_BUSY	3 B_ERROR	2 B_DONE	1 B_READ	0 B_WRITE
bno.	“延迟写”时有	“异步”时有	“等重用”时有	无	出错时有	有	二者有其一	
bno.	无	“异步”时有	“等重用”时有	有	无	无	二者有其一	

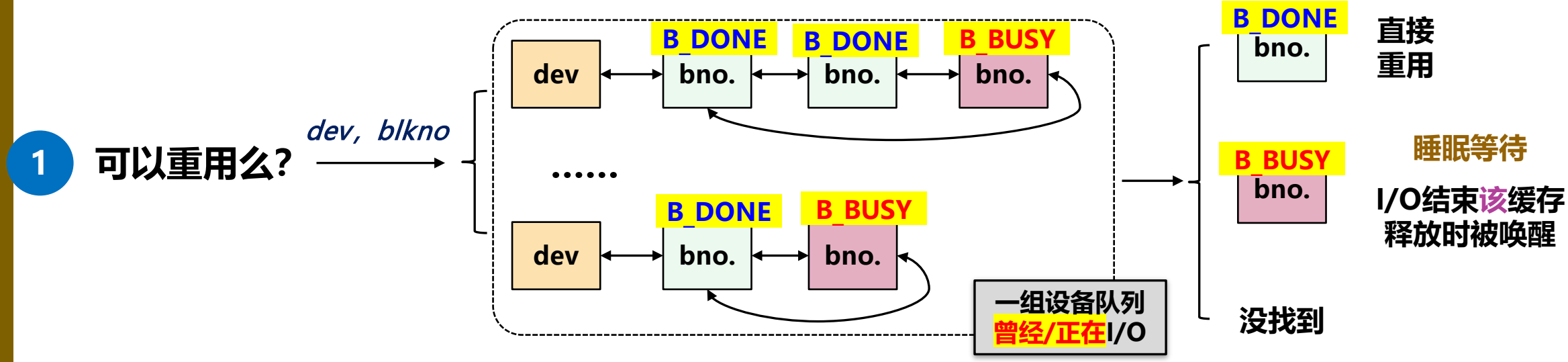
UNIX块设备缓存管理



	7 B_DELWRI	6 B_ASYNC	5 B_WANTED	4 B_BUSY	3 B_ERROR	2 B_DONE	1 B_READ	0 B_WRITE
bno.	“延迟写”时有	“异步”时有	“等重用”时有	无	出错时有	有	二者有其一	
bno.	无	“异步”时有	“等重用”时有	有	无	无	二者有其一	

用于区分两类缓存

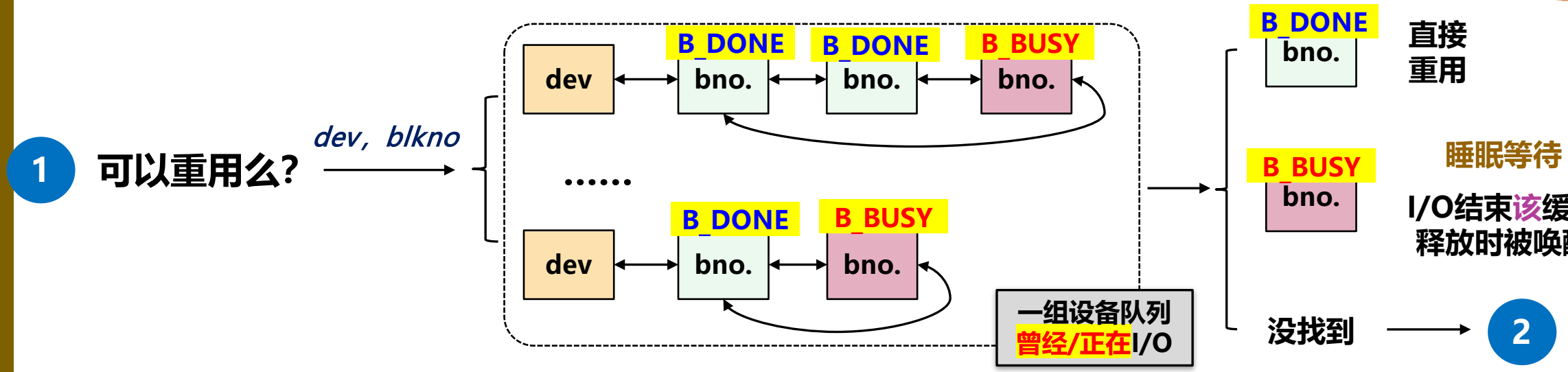
UNIX块设备缓存管理



	7 B_DELWRI	6 B_ASYNC	5 B_WANTED	4 B_BUSY	3 B_ERROR	2 B_DONE	1 B_READ	0 B_WRITE
bno.	“延迟写”时有	“异步”时有	“等重用”时有	无	出错时有	有	二者有其一	
bno.	无	“异步”时有	“等重用”时有	有	无	无	二者有其一	

用于区分两类缓存

UNIX块设备缓存管理



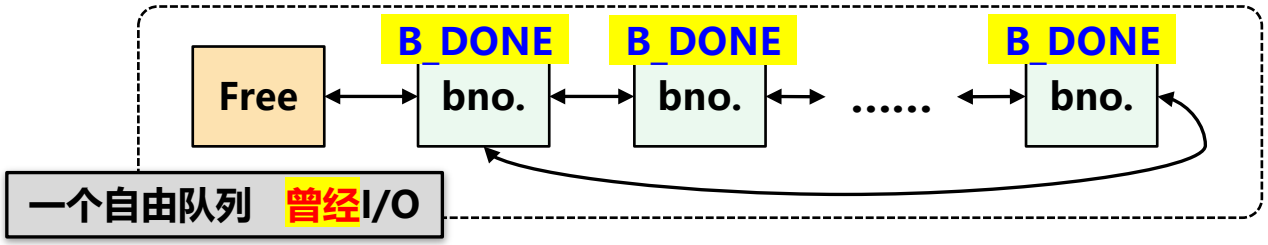
	7 B_DELWRI	6 B_ASYNC	5 B_WANTED	4 B_BUSY	3 B_ERROR	2 B_DONE	1 B_READ	0 B_WRITE
bno.	“延迟写”时有	“异步”时有	“等重用”时有	无	出错时有	有	二者有其一	
bno.	无	“异步”时有	“等重用”时有	有	无	无	二者有其一	

用于区分两类缓存

UNIX块设备缓存管理

缓存队列

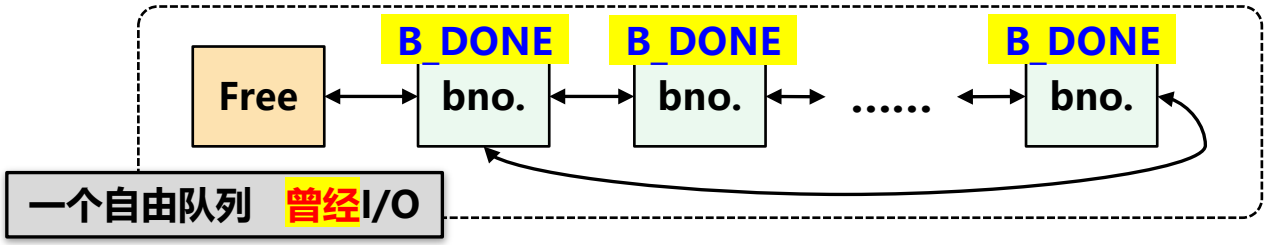
2 找可以置换的缓存



UNIX块设备缓存管理

缓存队列

2 找可以置换的缓存

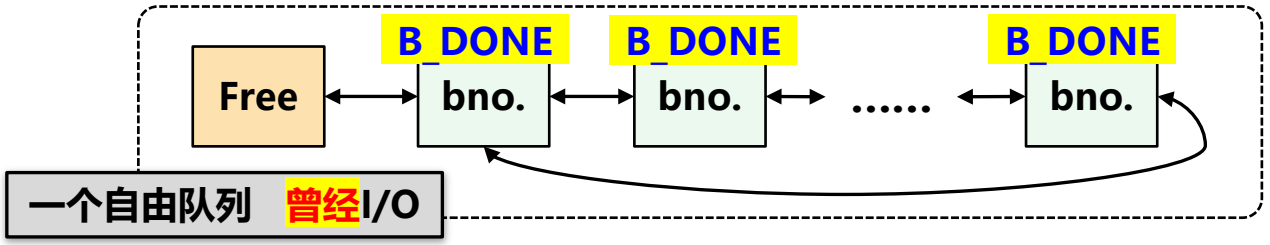


怎么找到最久未使用的缓存?

UNIX块设备缓存管理

缓存队列

2 找可以置换的缓存



怎么找到最久未使用的缓存?

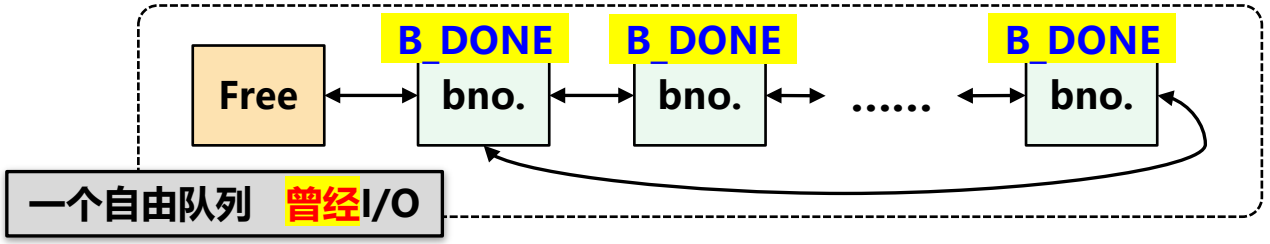
每次缓存使用（重新分配，重用）完毕，从队尾重新排队

每次分配缓存，从队头开始分配

UNIX块设备缓存管理

缓存队列

2 找可以置换的缓存



怎么找到最久未使用的缓存?

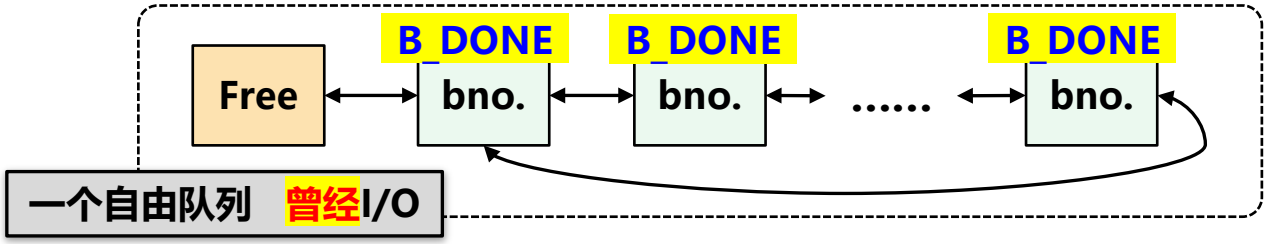
每次缓存使用（重新分配，重用）完毕，从队尾重新排队

每次分配缓存，从队头开始分配

读写时： 先找本设备队列，找到重用，找不到再从自由队首分配

UNIX块设备缓存管理

2 找可以置换的缓存



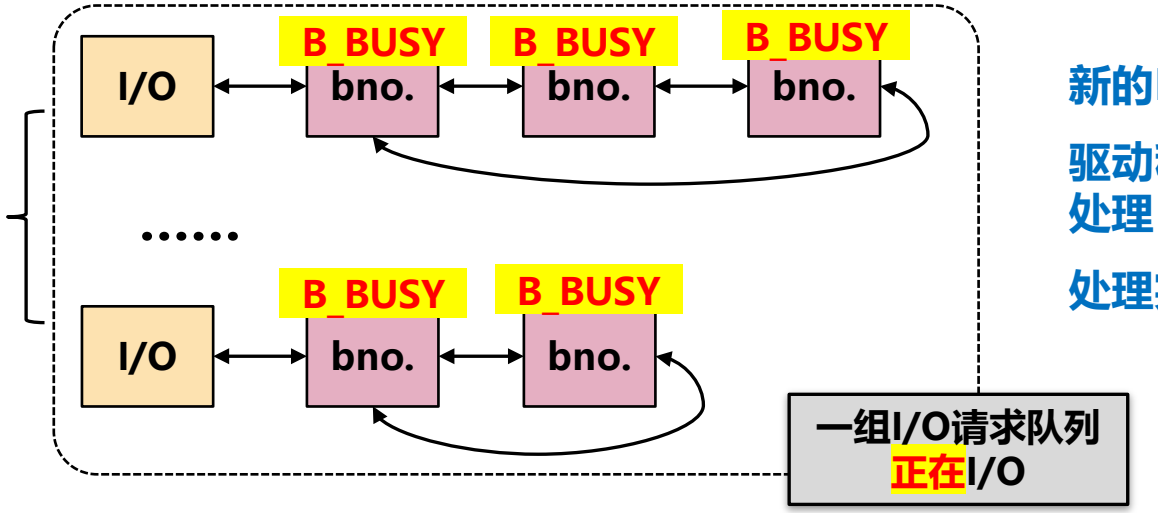
怎么找到最久未使用的缓存?

每次缓存使用（重新分配，重用）完毕，从队尾重新排队

每次分配缓存，从队头开始分配

读写时： 先找本设备队列，找到重用，找不到再从自由队首分配

3 为了I/O操作管理的方便



新的I/O操作挂在队尾

驱动程序从对头开始依次处理

处理完摘下

UNIX块设备缓存管理

缓存队列

```
Buf* b_forw;
Buf* b_back;
Buf* av_forw;
Buf* av_back;
```

两对指针可以分别将一个Buf对象插入两个队列中

```
class Buf
{
public:
    enum BufFlag              /* b_flags中标志位 */
    {
        B_WRITE = 0x1,        /* 写操作。将缓存中的信息写到硬盘上去 */
        B_READ= 0x2,          /* 读操作。从盘读取信息到缓存中 */
        B_DONE= 0x4,          /* I/O操作结束 */
        B_ERROR= 0x8,         /* I/O因出错而终止 */
        B_BUSY= 0x10,         /* 相应缓存正在使用中 */
        B_WANTED = 0x20,      /* 有进程正在等待使用该buf管理的缓存*/
        B_ASYNC= 0x40,        /* 异步I/O，不需要等待其结束 */
        B_DELWRI = 0x80       /* 延迟写 */
    };
public:
    short b_dev;              /* 高、低8位分别是主、次设备号 */
    int b_blkno;              /* 磁盘逻辑块号 */
    unsigned char* b_addr;    /* 指向该缓存控制块管理的缓冲区首地址 */
    int b_wcount;             /* 需传送的字节数 */
    unsigned int b_flags;     /* 缓存控制块标志位 */
    int b_error;              /* I/O出错时信息 */
    int b_resid;              /* I/O出错时尚未传送的剩余字节数 */
    int padding;              /* 4字节填充，否则强制转换会出错。 */
    Buf* b_forw;              /* 缓存控制块队列勾连指针 */
    Buf* b_back;
    Buf* av_forw;
    Buf* av_back;
};
```

UNIX块设备缓存管理

缓存队列

一共有四个队列：

- NODEV队列：与设备无关的Buf
- 自由队列： 空闲，目前没有用于I/O

整个系统一个

- 设备队列： 正在 或 曾经 用于该设备
- I/O请求队列： 正在用于进行I/O

一个设备一个

UNIX块设备缓存管理

缓存队列

```
class BufferManager {
        ……;
private:
        Buf bFreeLis
        ……;
};
```

自由队列和NODEV队列的队头

bFreeList

b_flags
…
*b_forw
*b_back
*av_forw
*av_back

一共有四个队列：

- NODEV队列：与设备无关的Buf
- 自由队列： 空闲，目前没有用于I/O

整个系统一个

- 设备队列： 正在 或 曾经 用于该设备
- I/O请求队列： 正在用于进行I/O

一个设备一个

UNIX块设备缓存管理

缓存队列

```
class BufferManager {
        ……;
private:
        Buf bFreeLis
        ……;
};
```

自由队列和NODEV队列的队头

bFreeList

b_flags
…
*b_forw
*b_back
*av_forw
*av_back

```
class Devtab {
        ……;
public:
        ……;
        Buf* b_forw;
        Buf* b_back;
        Buf* d_actf;
        Buf* d_actl;
};
```

设备队列和I/O请求队列的队头

Devtab

…
*b_forw
*b_back
*d_catf
*d_actl

一共有四个队列：

- NODEV队列：与设备无关的Buf
- 自由队列： 空闲，目前没有用于I/O

整个系统一个

- 设备队列： 正在 或 曾经 用于该设备
- I/O请求队列： 正在用于进行I/O

一个设备一个

UNIX块设备缓存管理

缓存队列

```
class BufferManager {
        ……;
private:
        Buf bFreeLis
        ……;
};
```

自由队列和NODEV队列的队头

bFreeList

b_flags
…
*b_forw
*b_back
*av_forw
*av_back

```
class Devtab {
        ……;
public:
        ……;
        Buf* b_forw;
        Buf* b_back;
        Buf* d_actf;
        Buf* d_actl;
};
```

设备队列和I/O请求队列的队头

Devtab

…
*b_forw
*b_back
*d_catf
*d_actl

一共有四个队列：

- NODEV队列：与设备无关的Buf
- 自由队列： 空闲，目前没有用于I/O

整个系统一个

- 设备队列： 正在 或 曾经 用于该设备
- I/O请求队列： 正在用于进行I/O

一个设备一个

```
Buf* b_forw;
Buf* b_back;
Buf* av_forw;
Buf* av_back;
```

不用于I/O　曾经/正在I/O

将Buf插入NODEV队列或某一设备队列

曾经I/O　正在I/O

将Buf插入自由队列或某一I/O请求队列

UNIX块设备缓存管理

缓存队列

每个Buf 同时位于两个队列中：

初始时：自由队列 和 NODEV队列
读写时：设备队列 和 I/O请求队列
结束时：设备队列 和 自由队列

读写时： 先找本设备队列，找到利用。找不到再从自由队首分配。
（*从其他设备队列中摘下*）

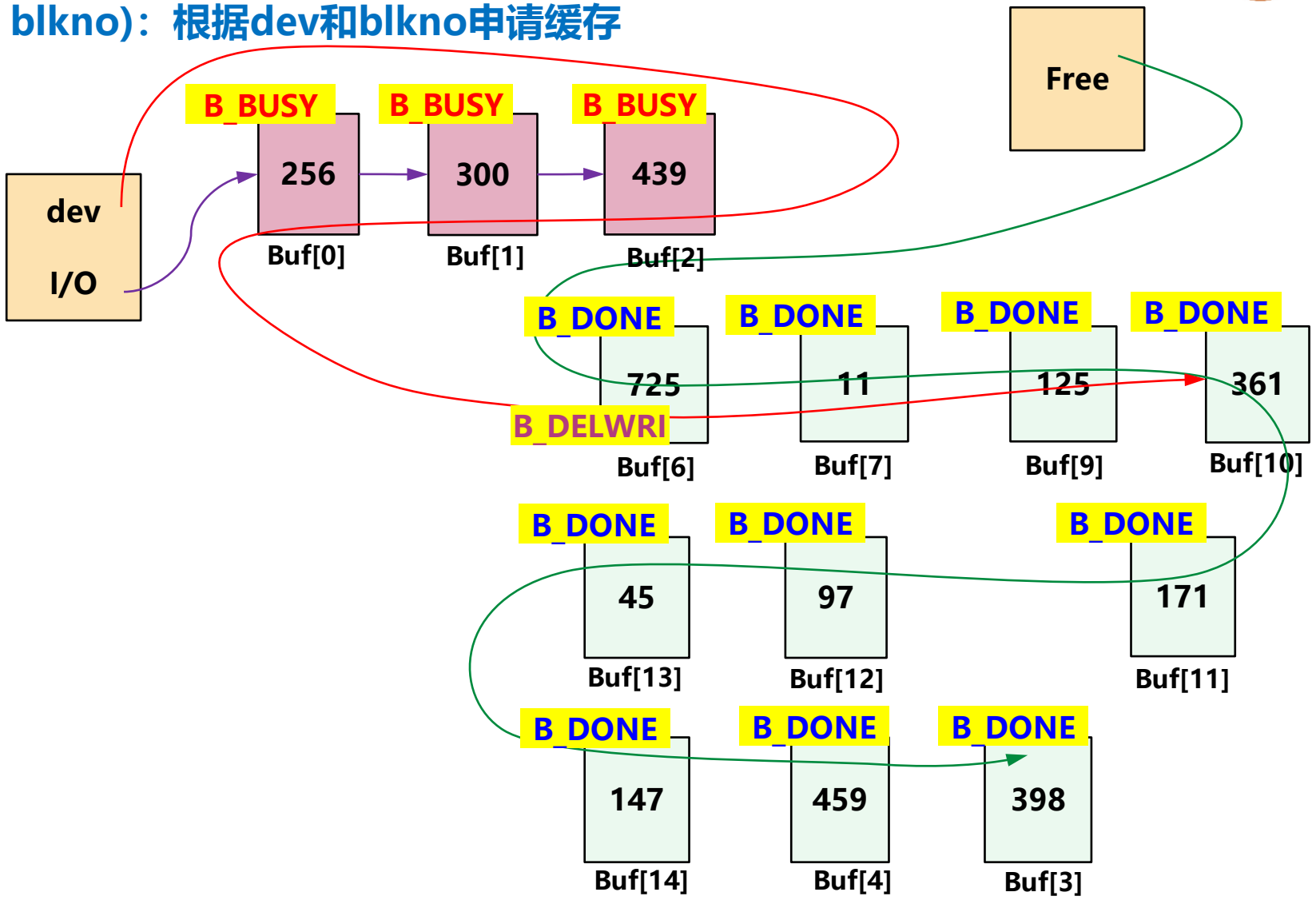
读写完毕：从I/O请求队列中摘下，释放到自由队尾。
（*但仍留在原设备队列中*）

只要未重分配就保持原内容
FIFO保证自由队列中的缓冲区尽可能长时间的保持原状

UNIX块设备缓存管理

缓存分配过程

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存



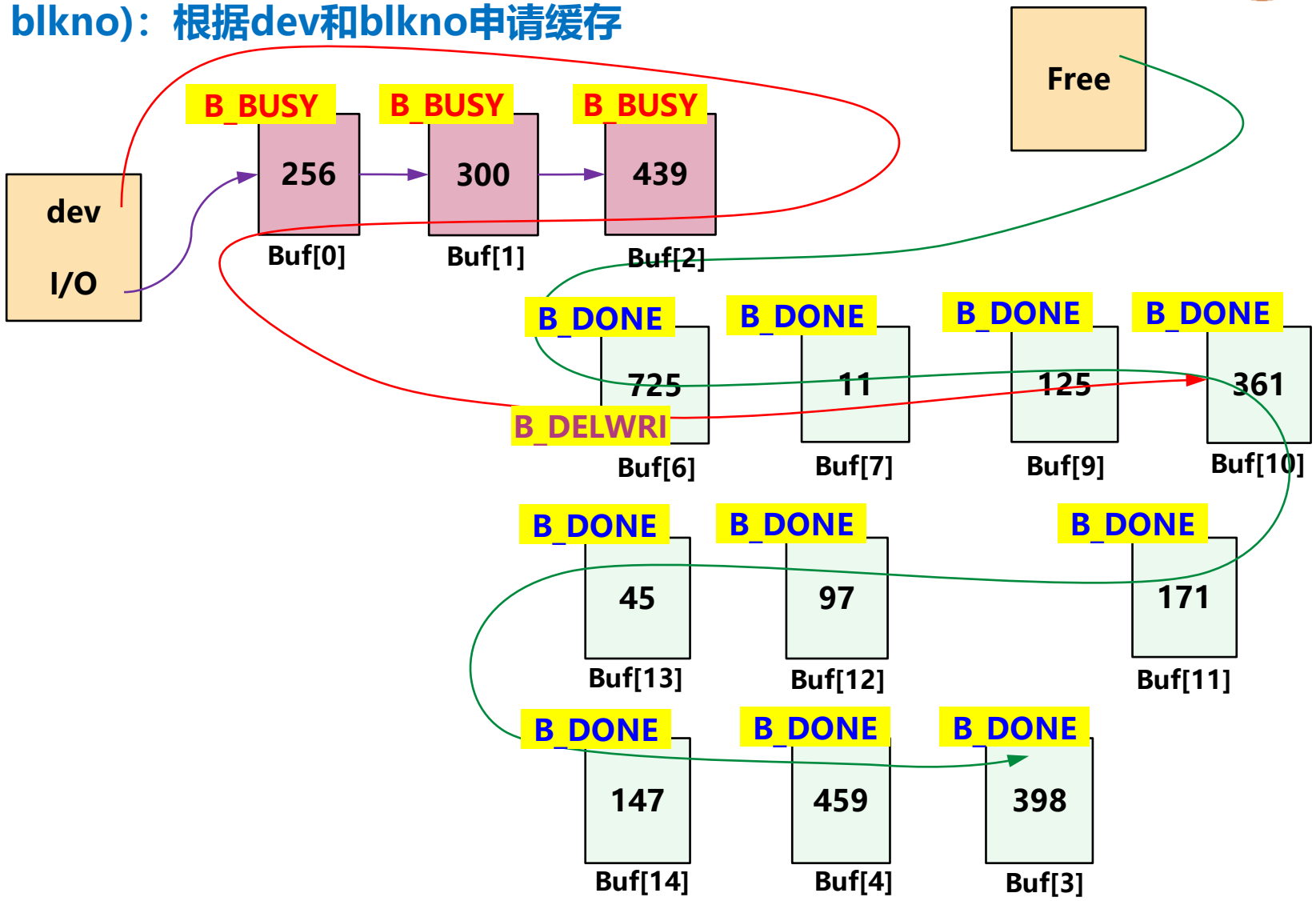
UNIX块设备缓存管理

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

缓存分配过程

1. GetBlk (0, 125);

在设备队列中找与
dev, blkno相同者



UNIX块设备缓存管理

缓存分配过程

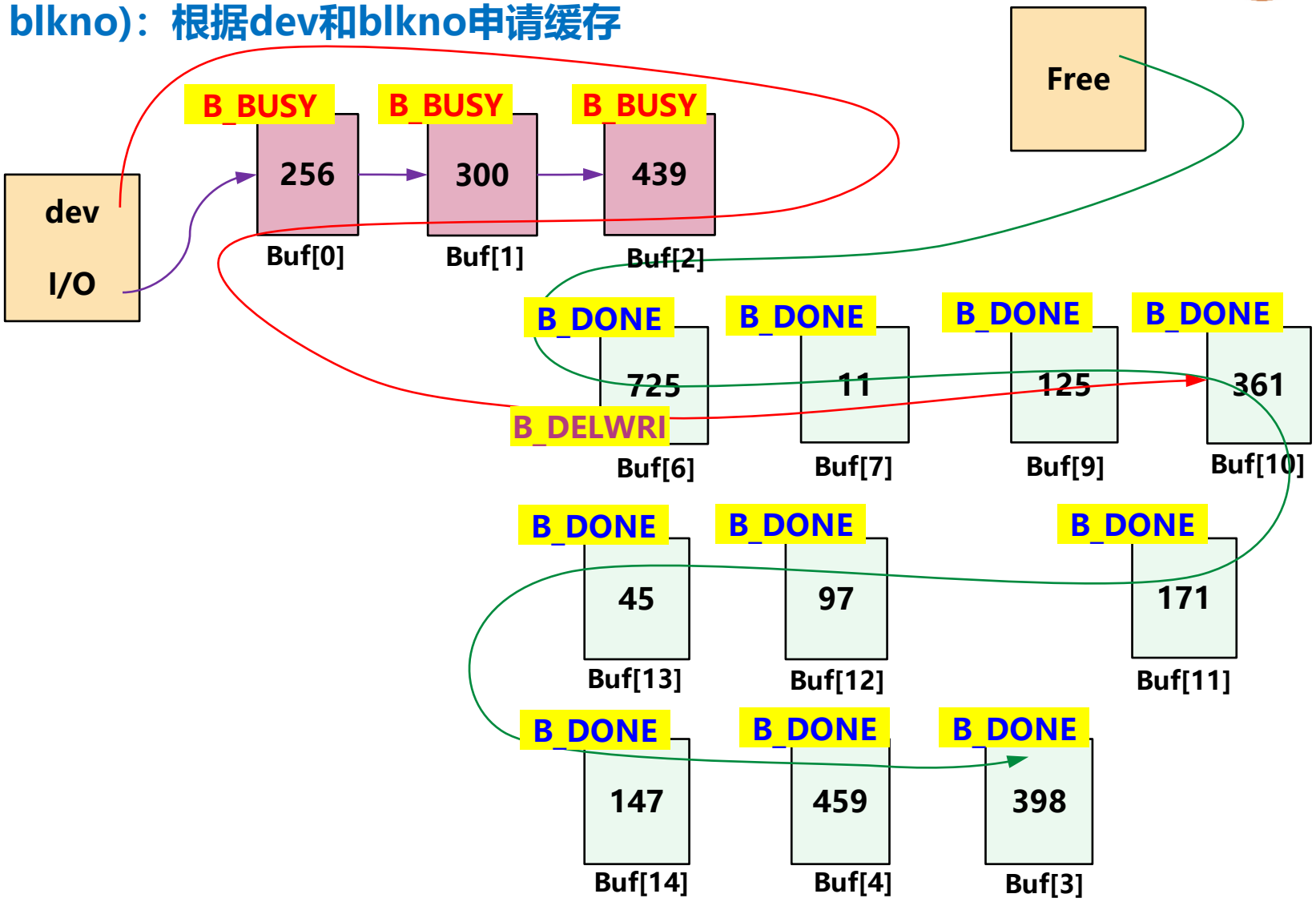
BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

1. GetBlk (0, 125);

在设备队列中找与
dev, blkno相同者

找到

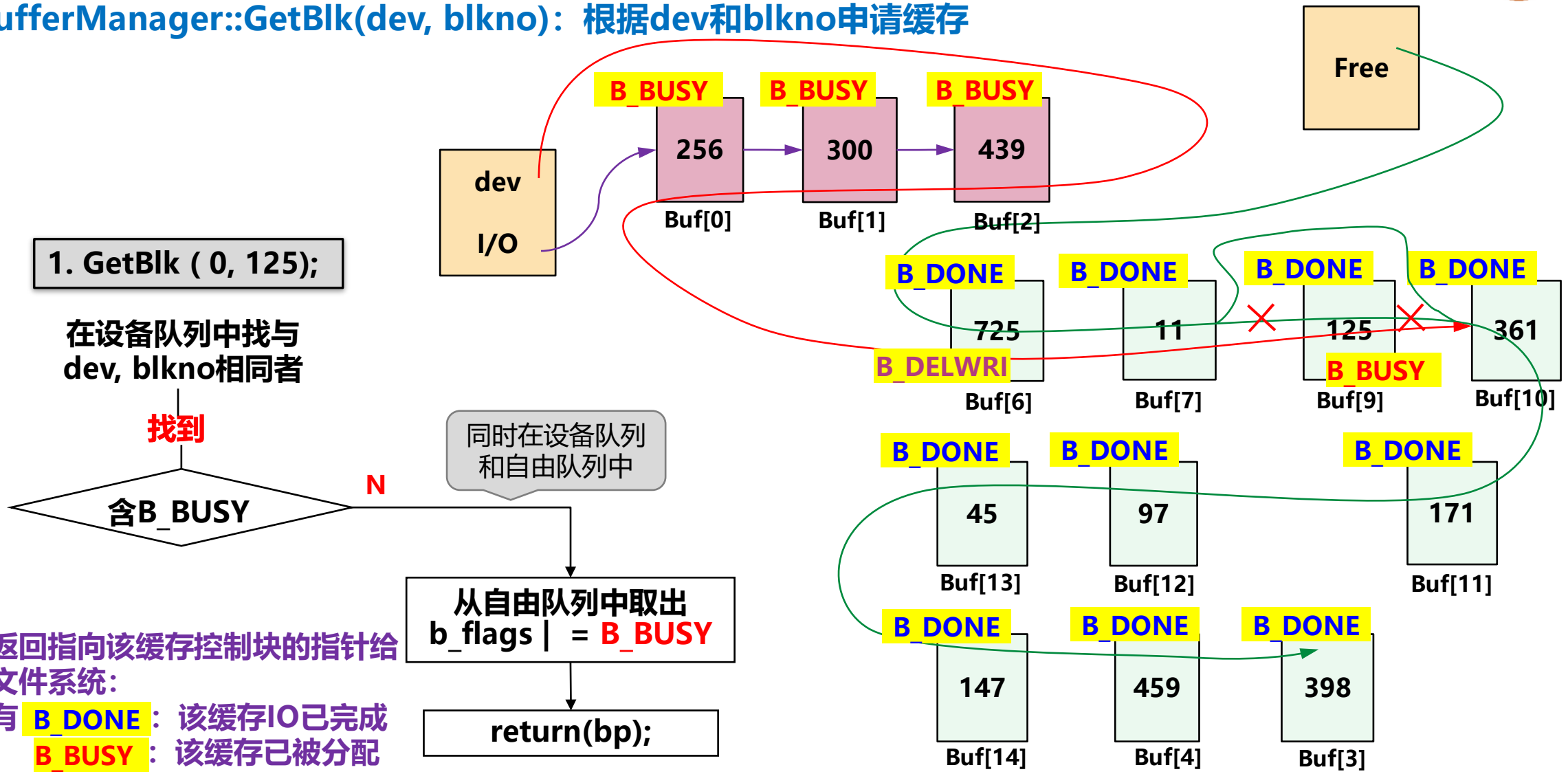
含B_BUSY



UNIX块设备缓存管理

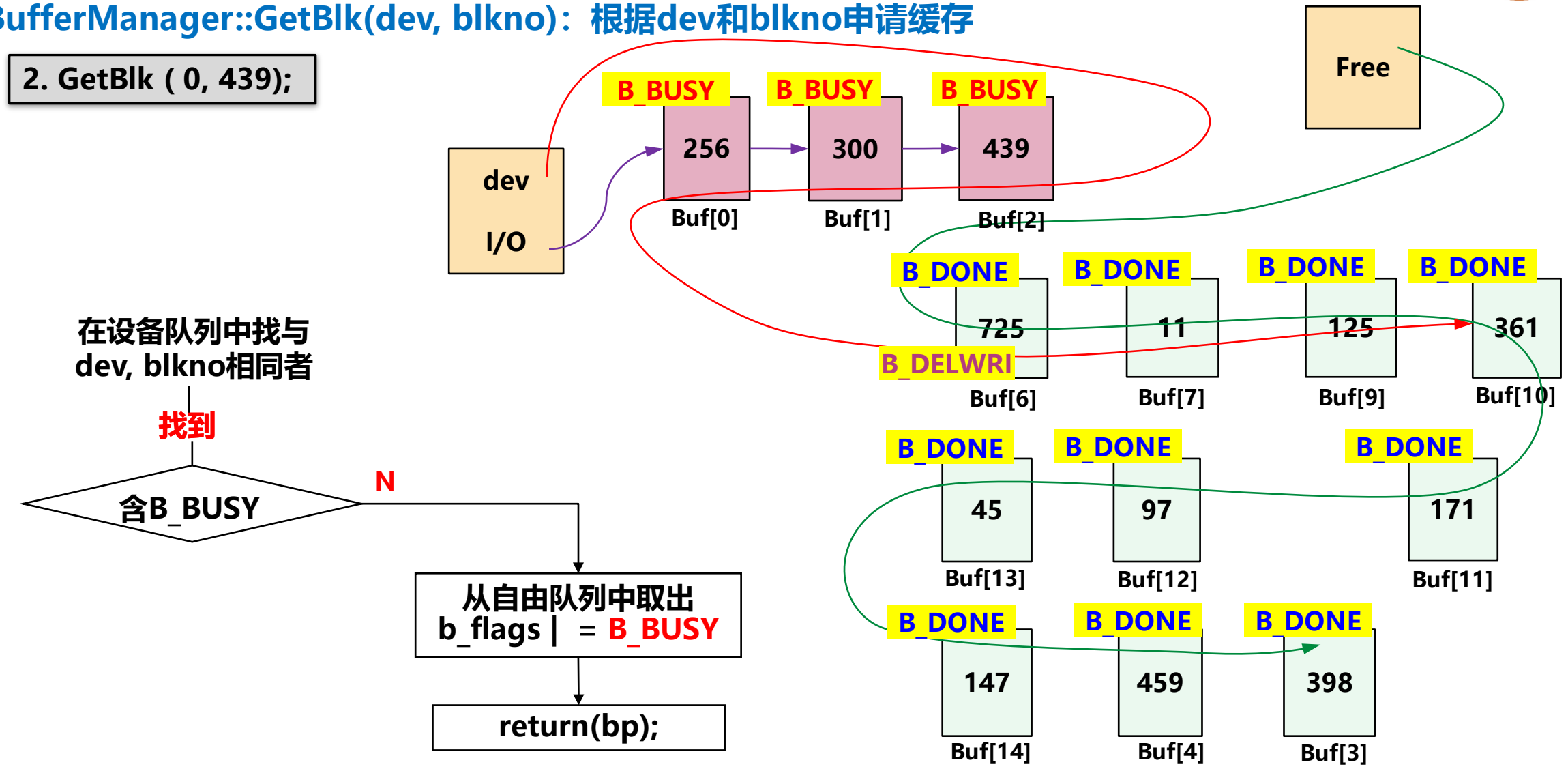
缓存分配过程

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存



UNIX块设备缓存管理

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

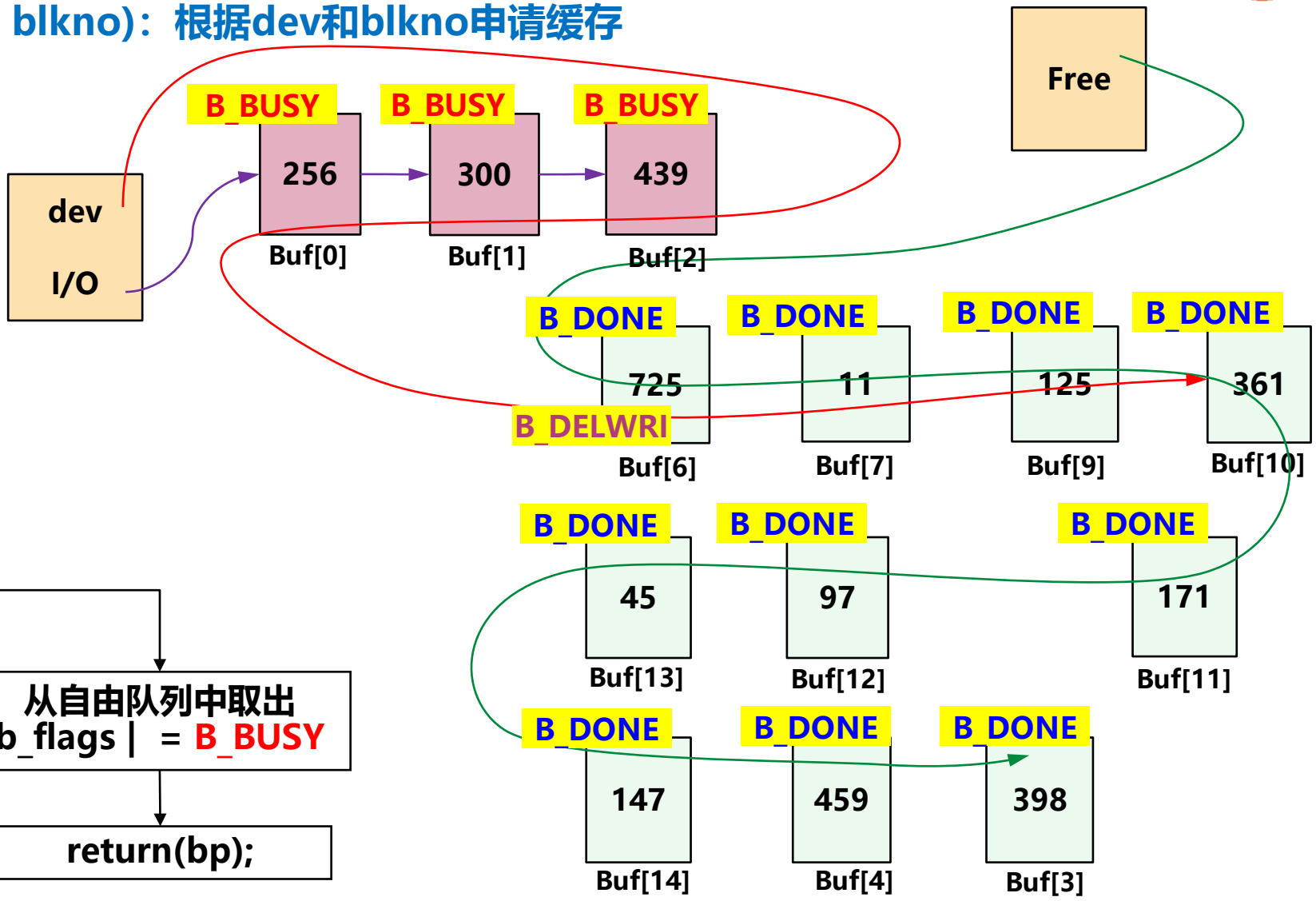


缓存分配过程

UNIX块设备缓存管理

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

2. GetBlk (0, 439);



在设备队列中找与dev, blkno相同者

找到

含B_BUSY

N

Y

b_flags | = B_WANTED
Sleep(bp, PRIBIO)

睡眠原因：指向该缓存控制块的指针 ❶

从自由队列中取出
b_flags | = B_BUSY

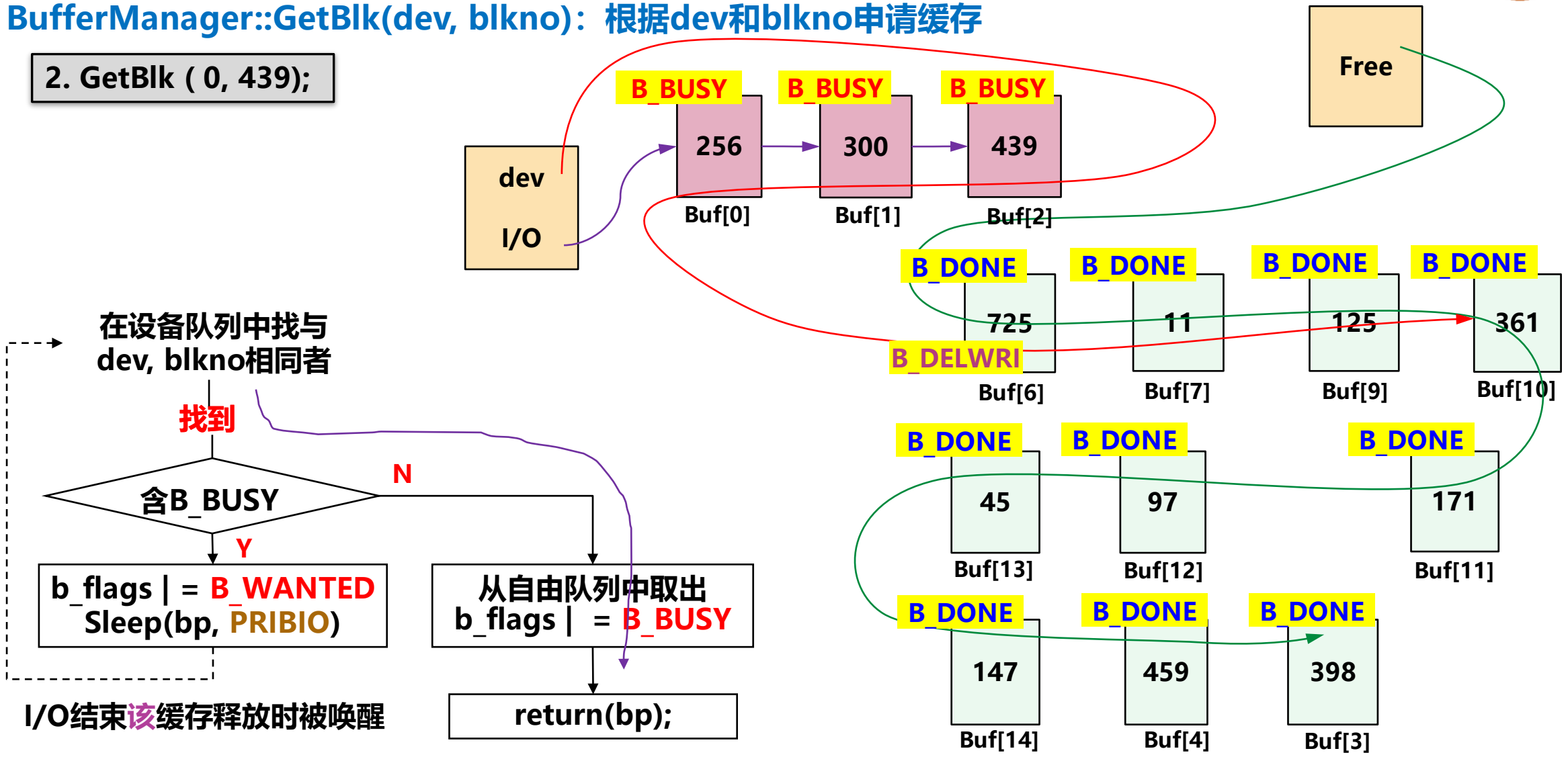
return(bp);

UNIX块设备缓存管理

缓存分配过程

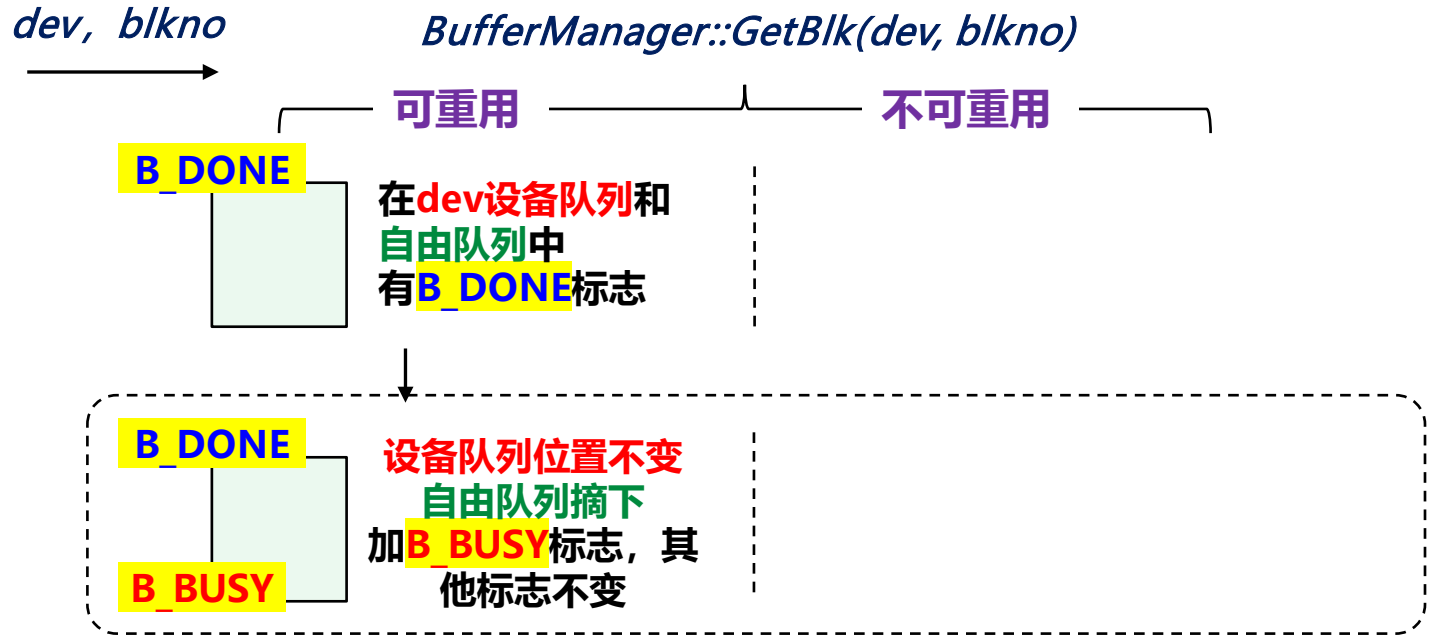
BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

2. GetBlk (0, 439);



UNIX块设备缓存管理

缓存分配过程



UNIX块设备缓存管理

缓存分配过程

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

3. GetBlk (0, 100);

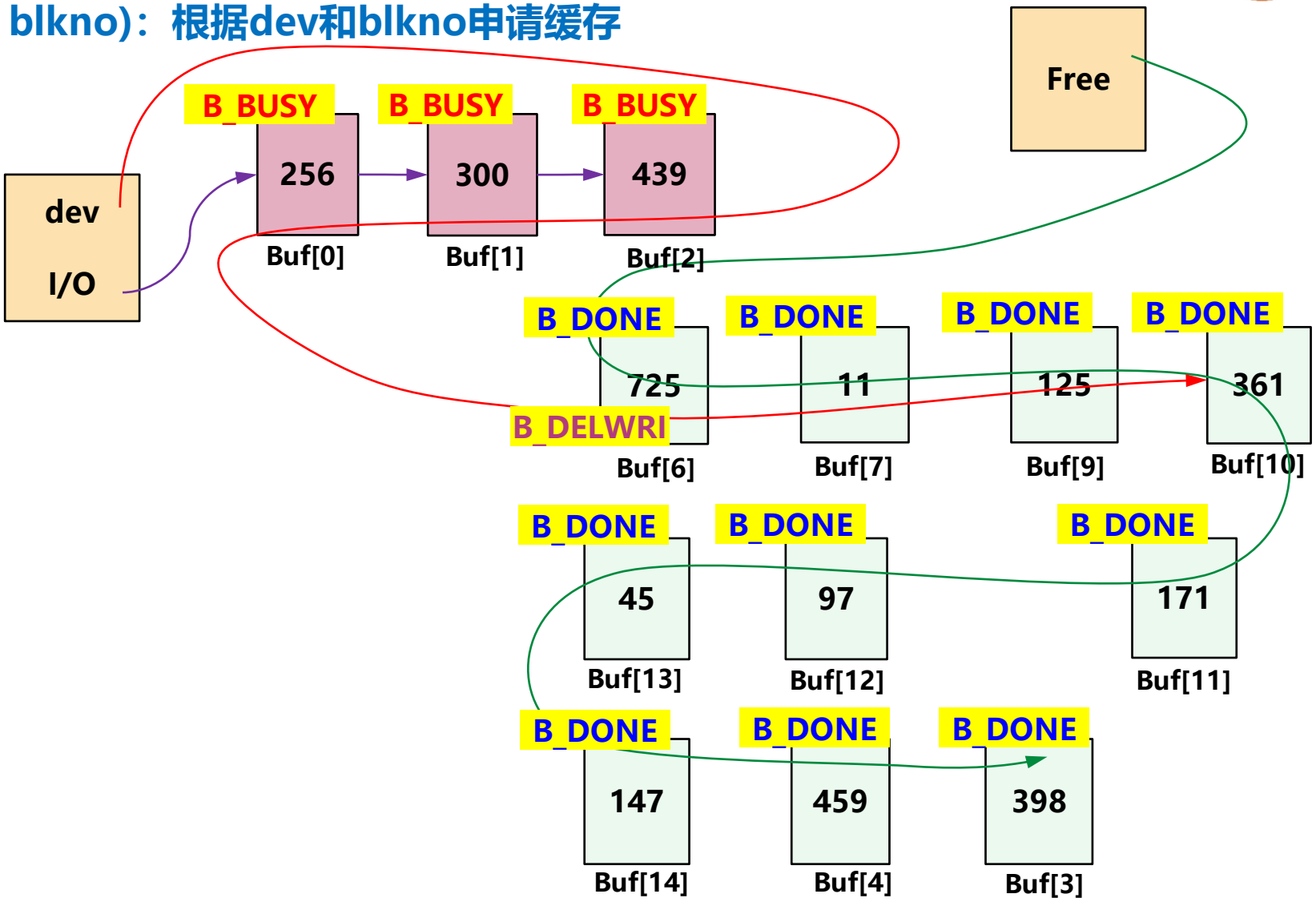
在设备队列中找与dev, blkno相同者

找不到

自由队列空

N

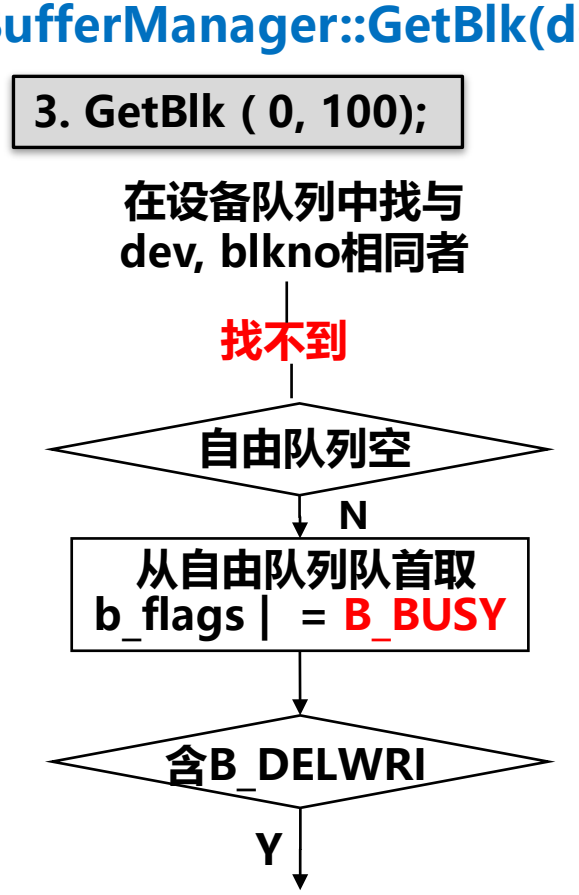
从自由队列队首取
b_flags | = B_BUSY

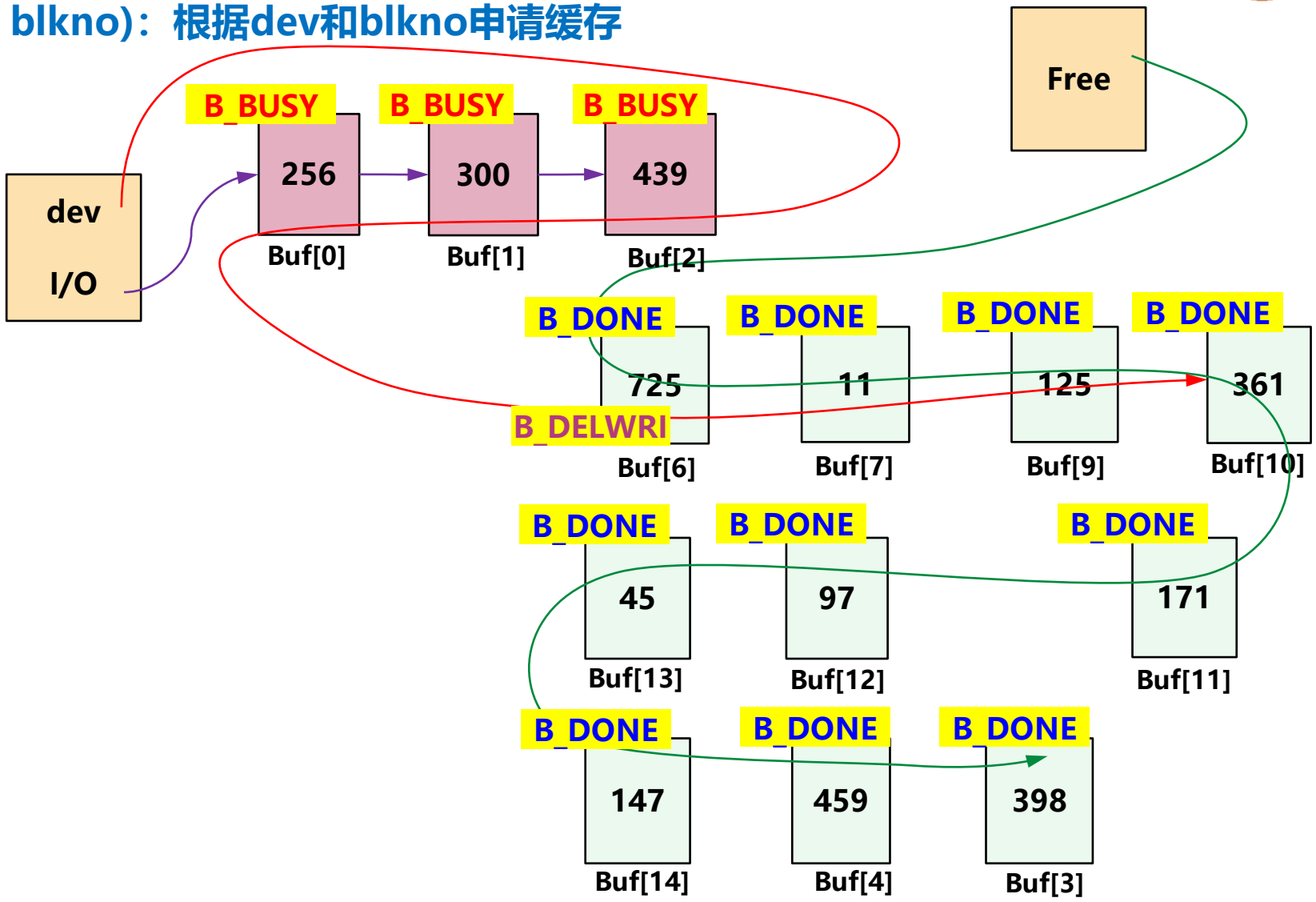


UNIX块设备缓存管理

缓存分配过程

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

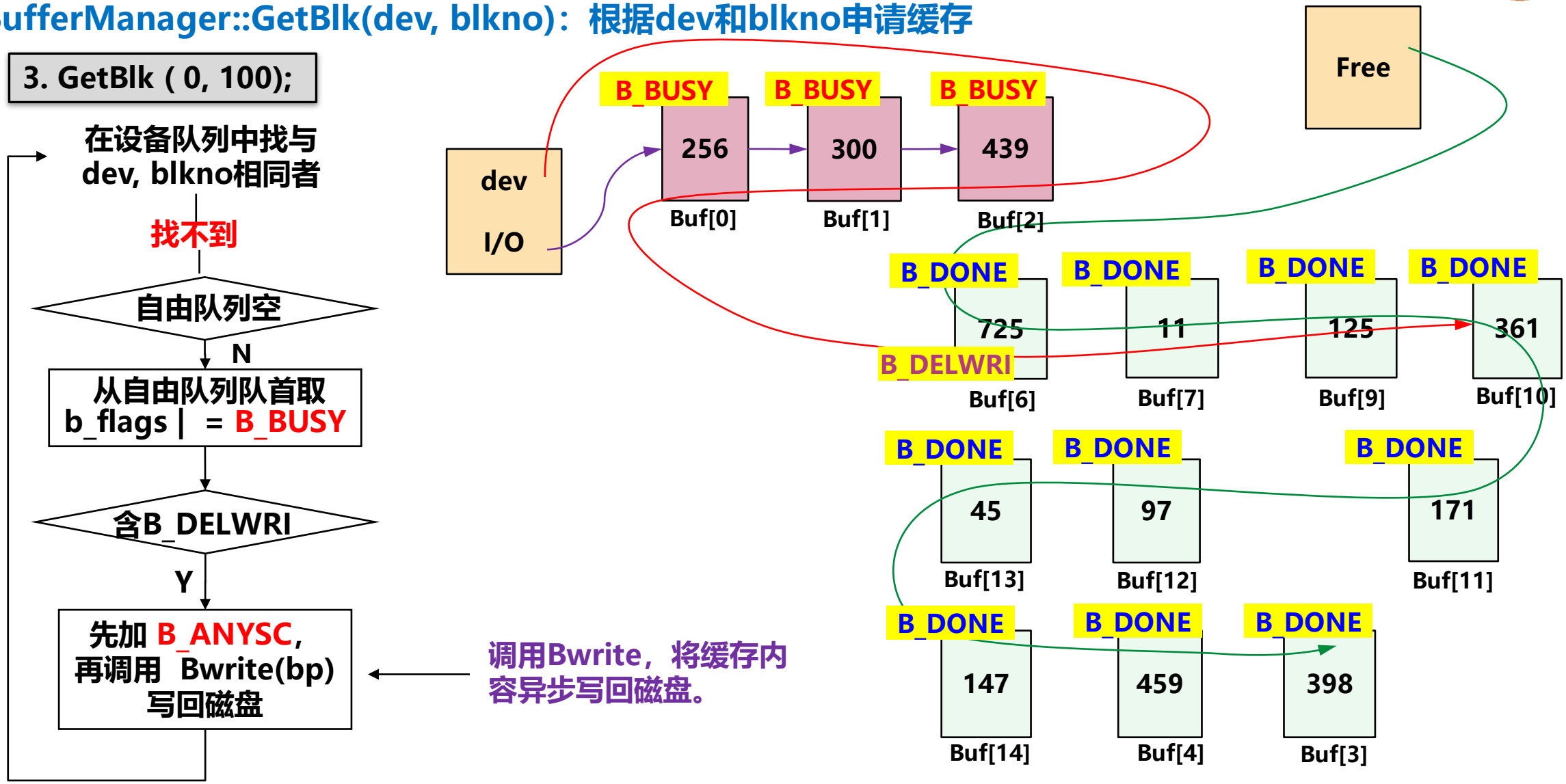




UNIX块设备缓存管理

缓存分配过程

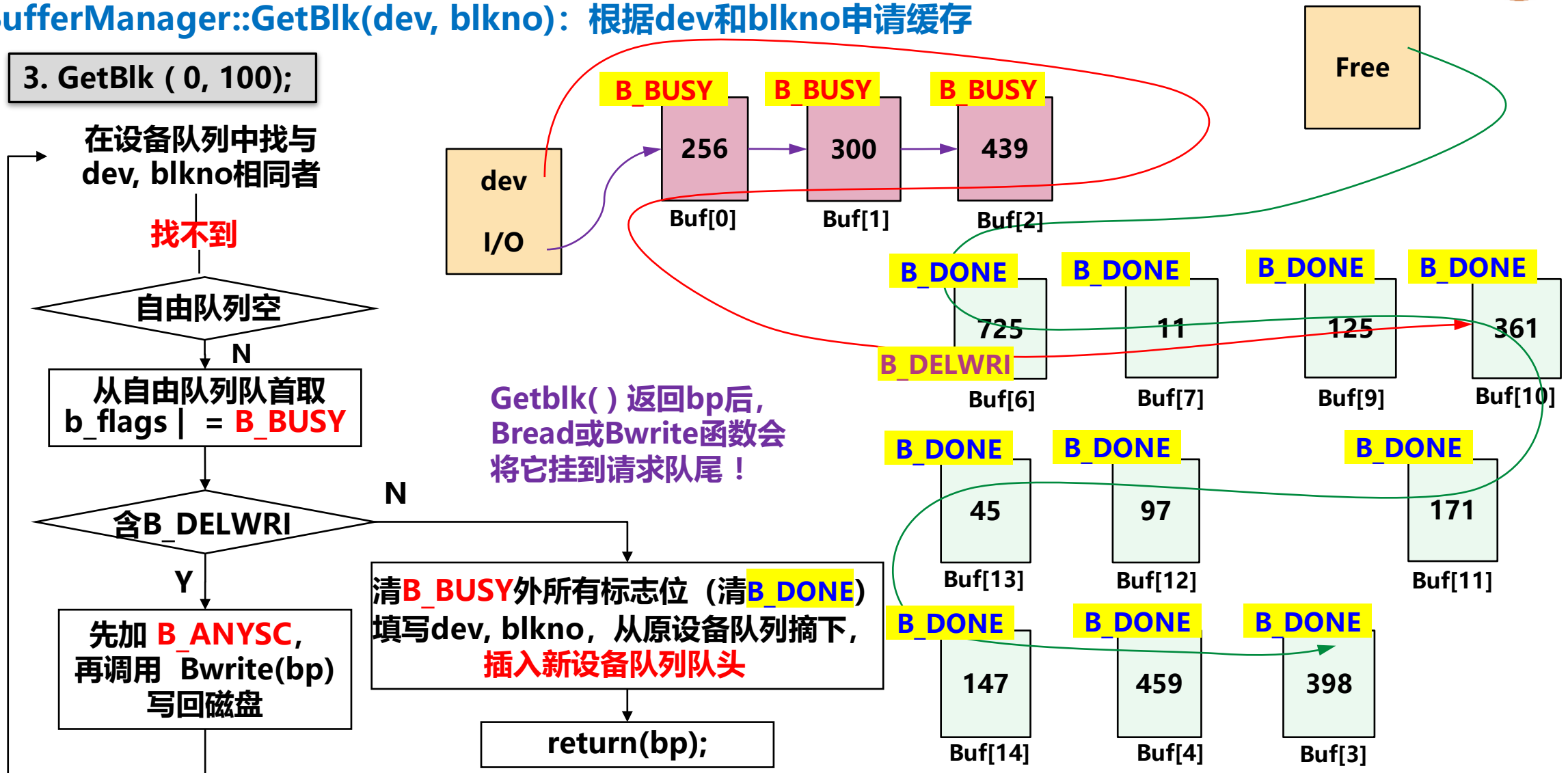
BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存



UNIX块设备缓存管理

缓存分配过程

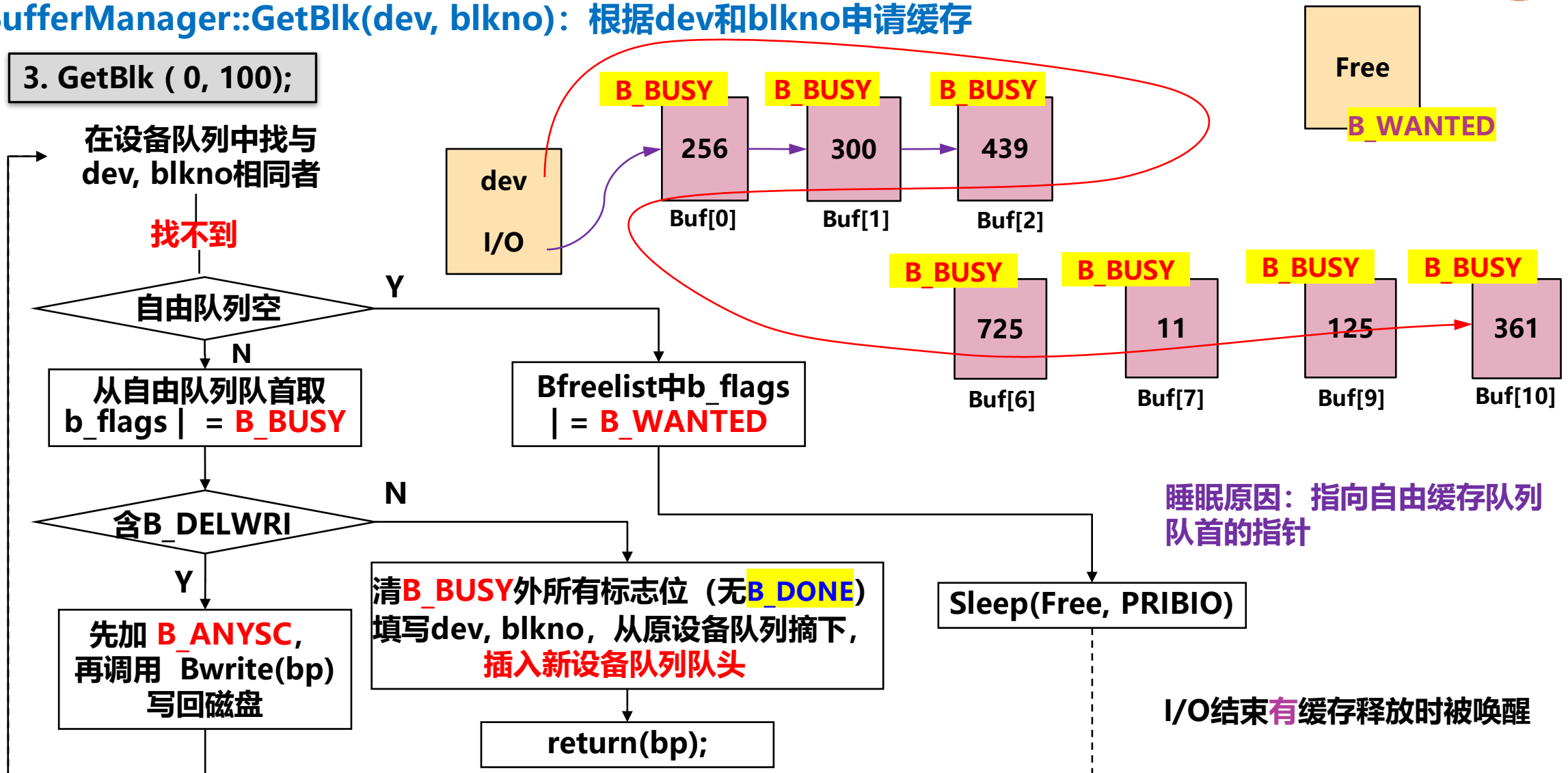
BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存



UNIX块设备缓存管理

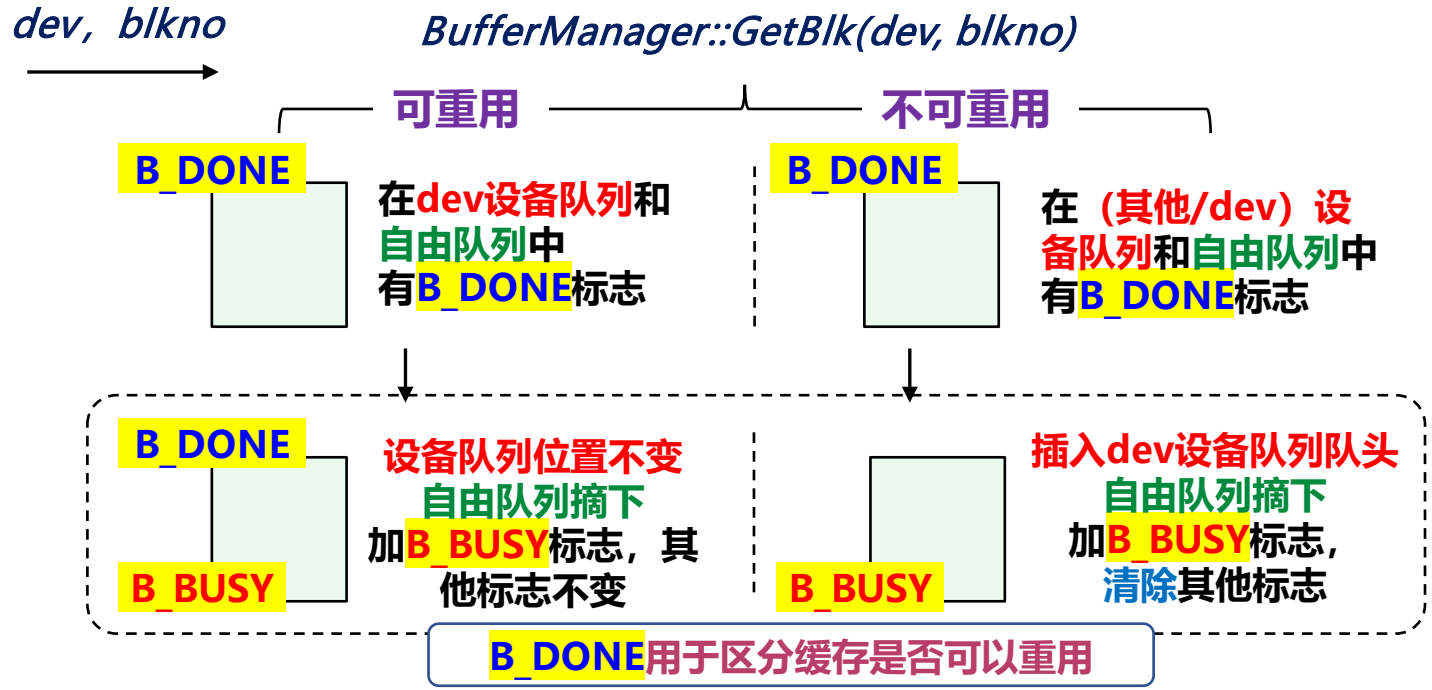
缓存分配过程

BufferManager::GetBlk(dev, blkno)：根据dev和blkno申请缓存

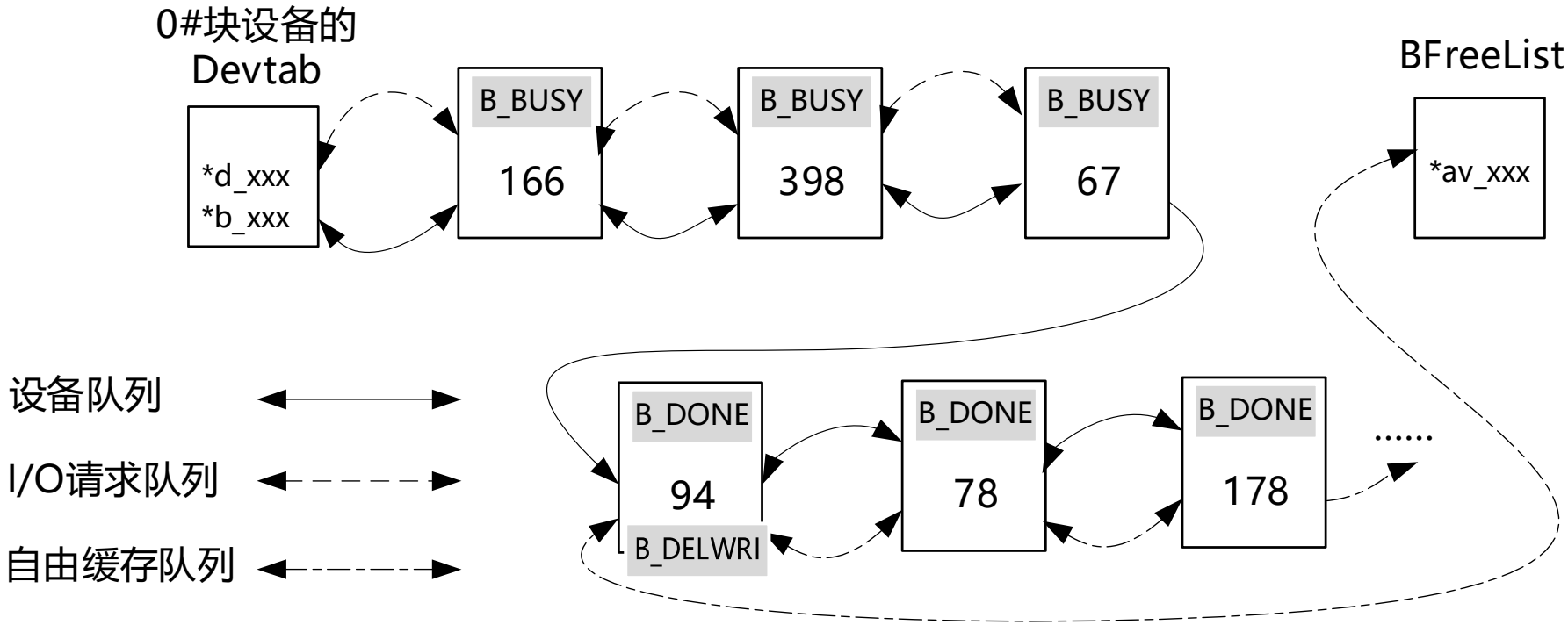


UNIX块设备缓存管理

缓存分配过程

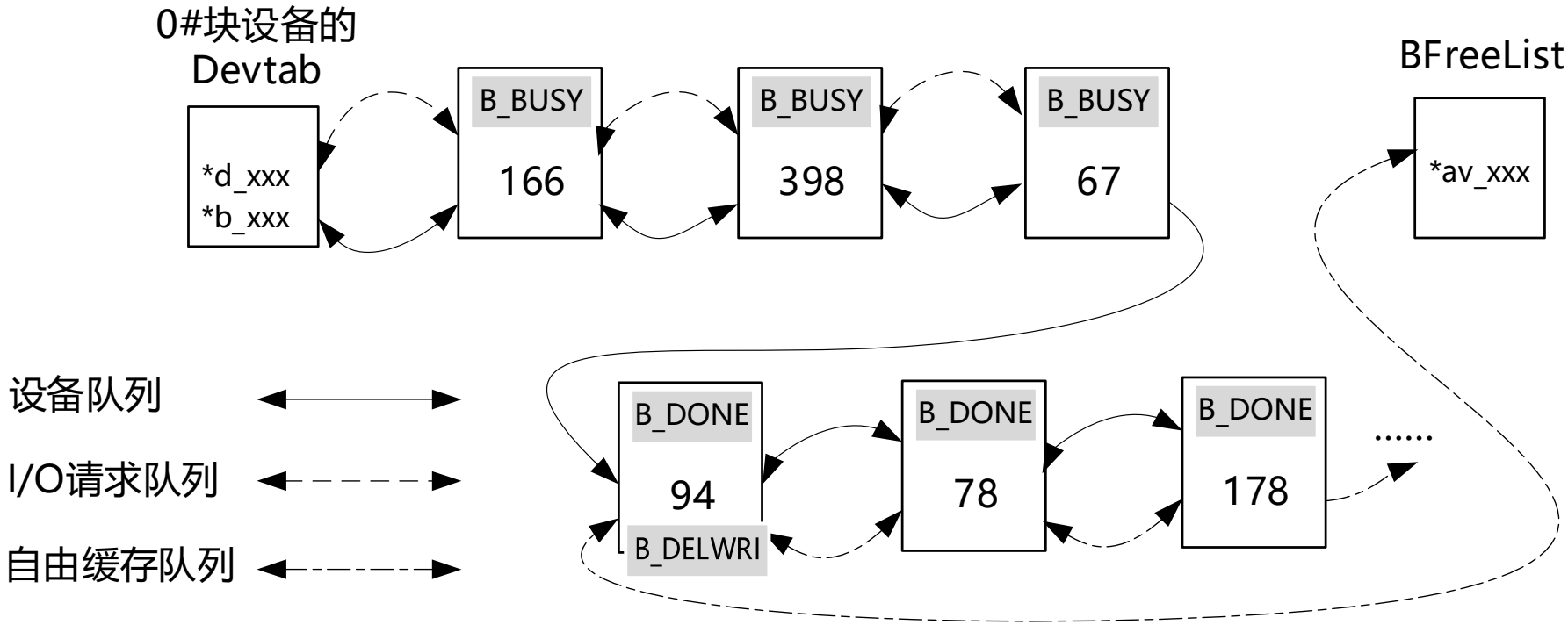


UNIX块设备缓存管理



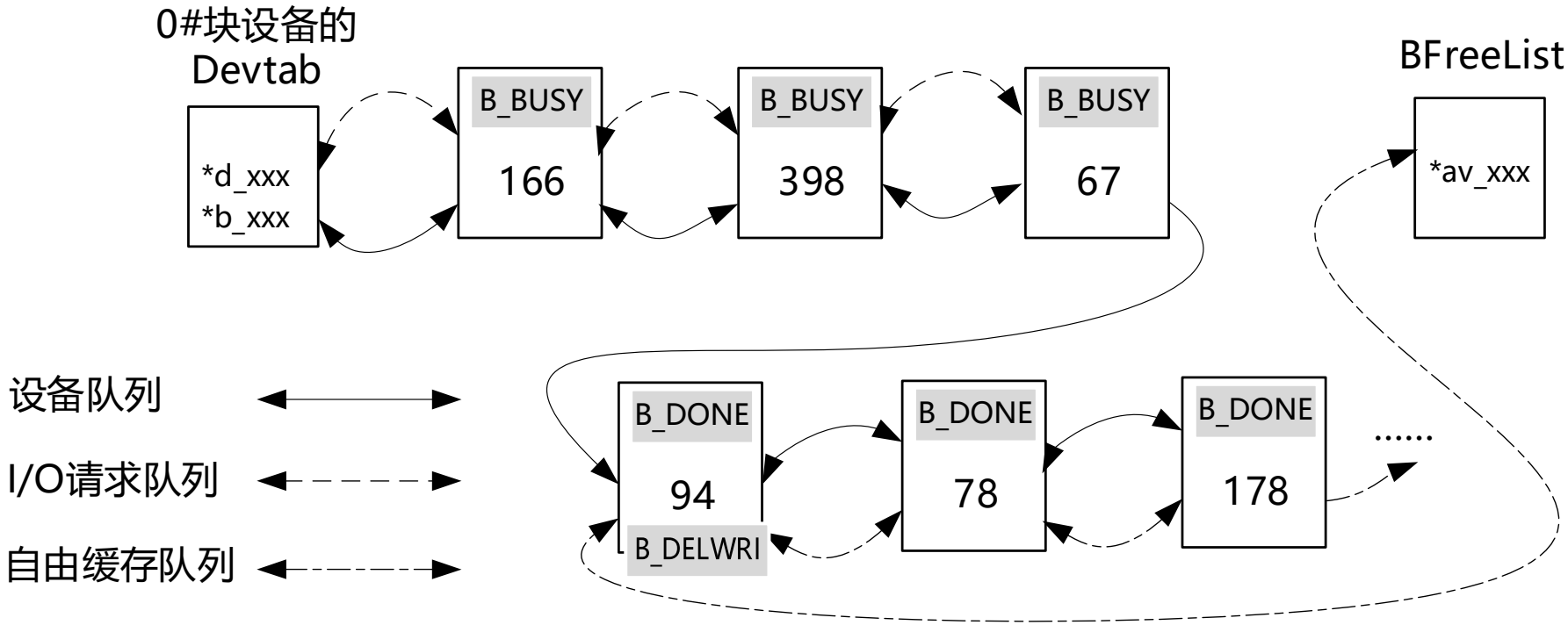
（1）如果此时进程pa读取该设备上的398号数据块，。。。

UNIX块设备缓存管理



（1）如果此时进程pa读取该设备上的94号数据块，。。。

UNIX块设备缓存管理



（1）如果此时进程pa读取该设备上的100号数据块，。。。

本节小结

1　掌握UNIX的块设备缓存队列设置

2　掌握UNIX的块设备缓存分配过程

阅读讲义：228页 ~ 235页；242页 ~ 248页