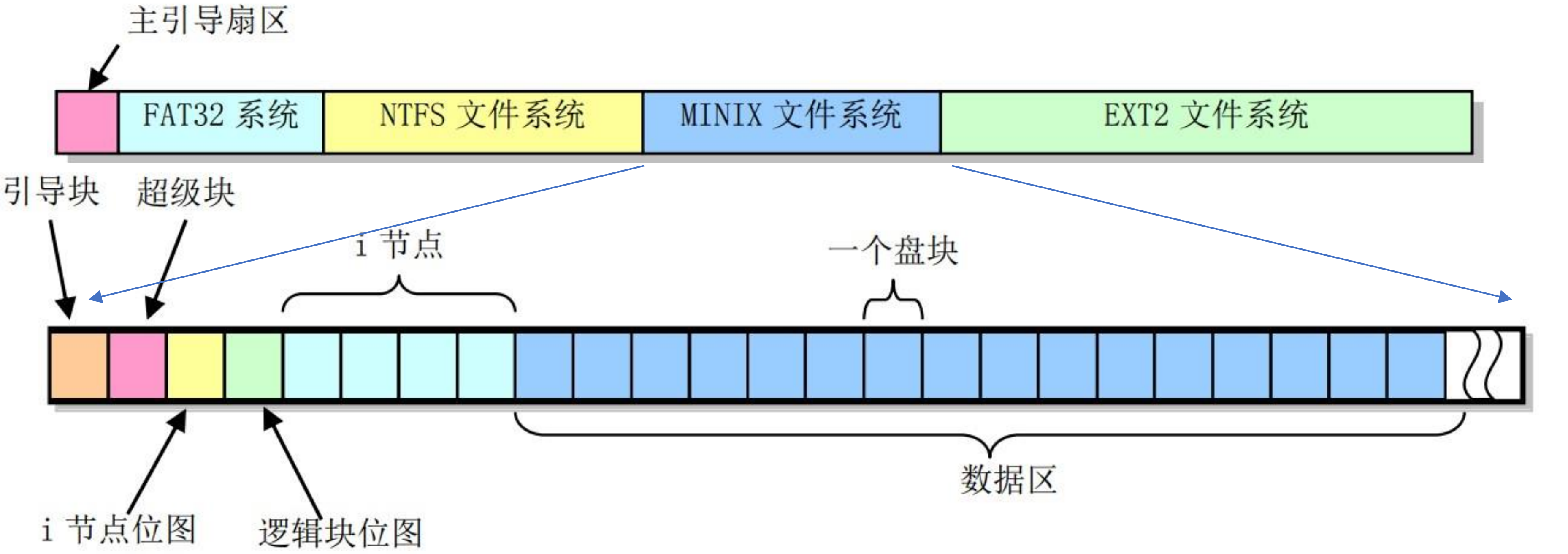
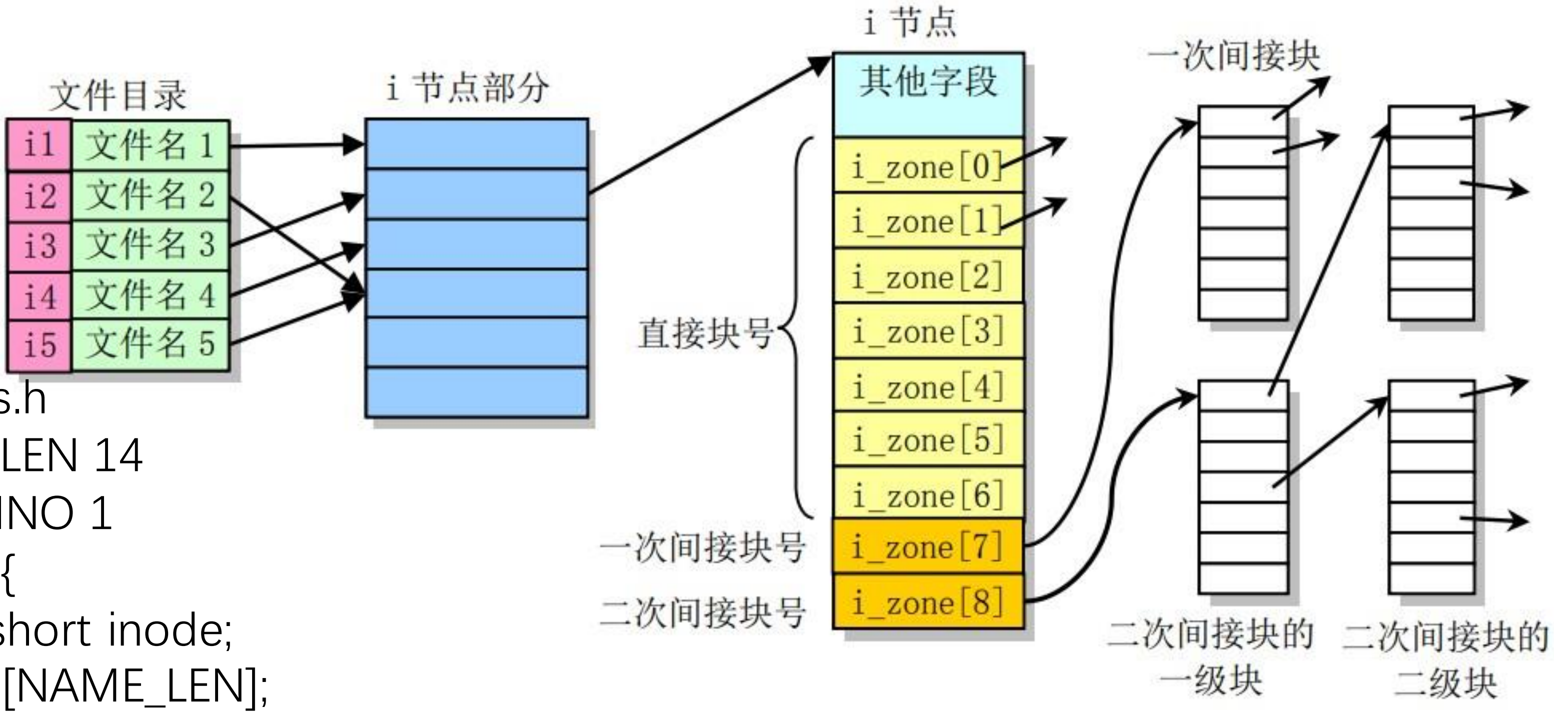
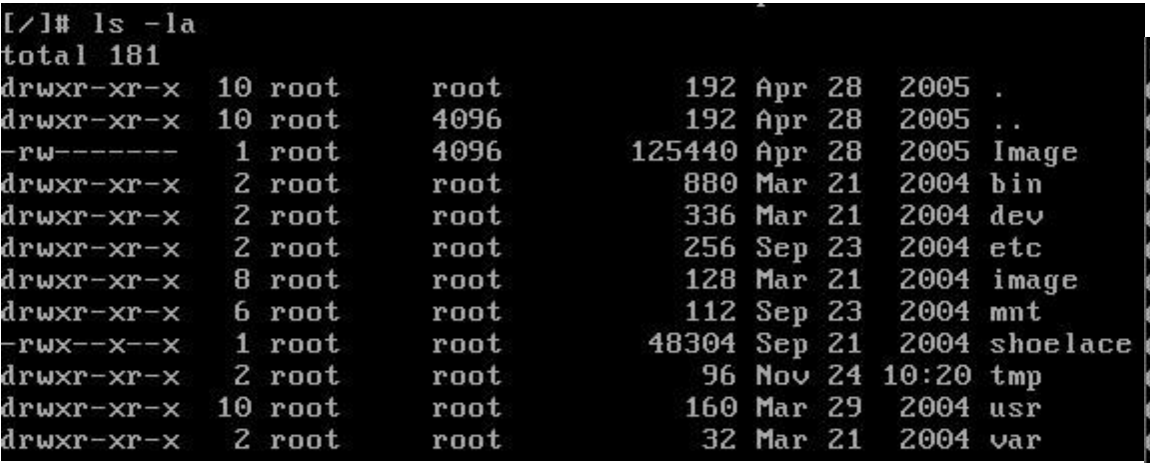
/fs

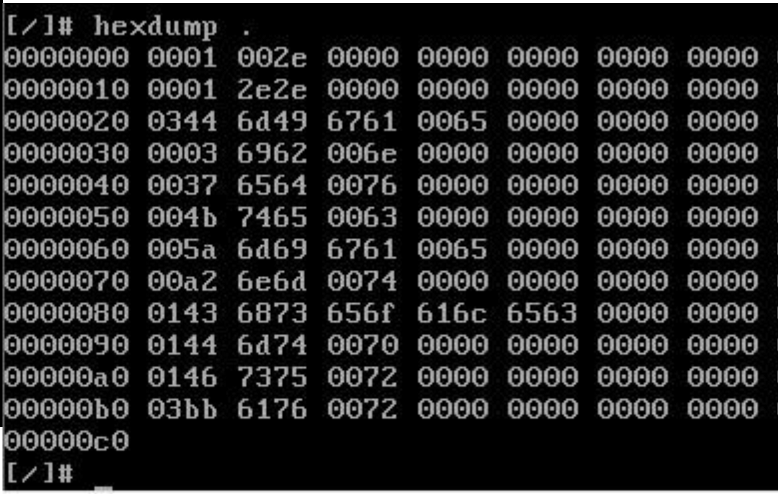




```
/include/linux/fs.h
#define NAME_LEN 14
#define ROOT_INO 1
struct dir_entry {
        unsigned short inode;
        char name[NAME_LEN];
};
```

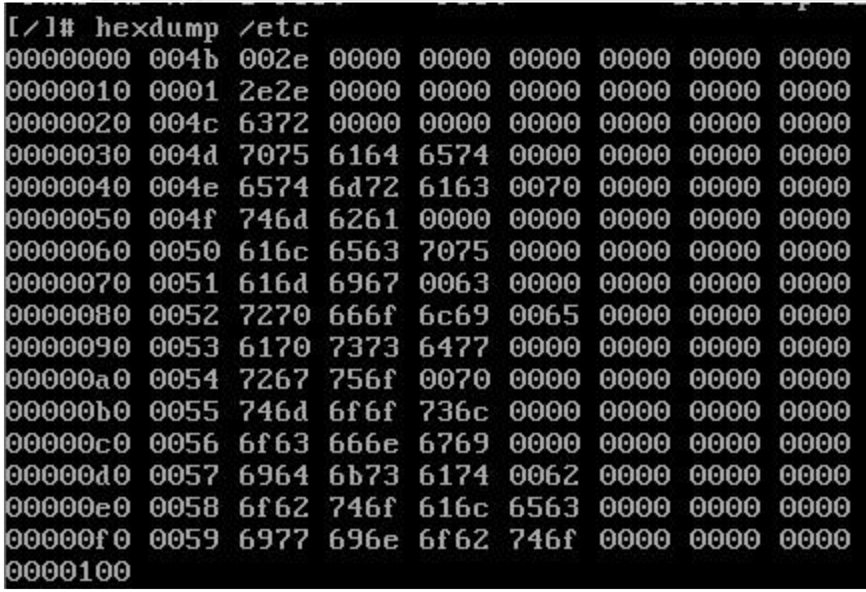


```
[/]# ls -la
total 181
drwxr-xr-x  10 root     root          192 Apr 28  2005 .
drwxr-xr-x  10 root     4096          192 Apr 28  2005 ..
-rw-------   1 root     4096       125440 Apr 28  2005 Image
drwxr-xr-x   2 root     root          880 Mar 21  2004 bin
drwxr-xr-x   2 root     root          336 Mar 21  2004 dev
drwxr-xr-x   2 root     root          256 Sep 23  2004 etc
drwxr-xr-x   8 root     root          128 Mar 21  2004 image
drwxr-xr-x   6 root     root          112 Sep 23  2004 mnt
-rwx--x--x   1 root     root        48304 Sep 21  2004 shoelace
drwxr-xr-x   2 root     root           96 Nov 24 10:20 tmp
drwxr-xr-x  10 root     root          160 Mar 29  2004 usr
drwxr-xr-x   2 root     root           32 Mar 21  2004 var
```

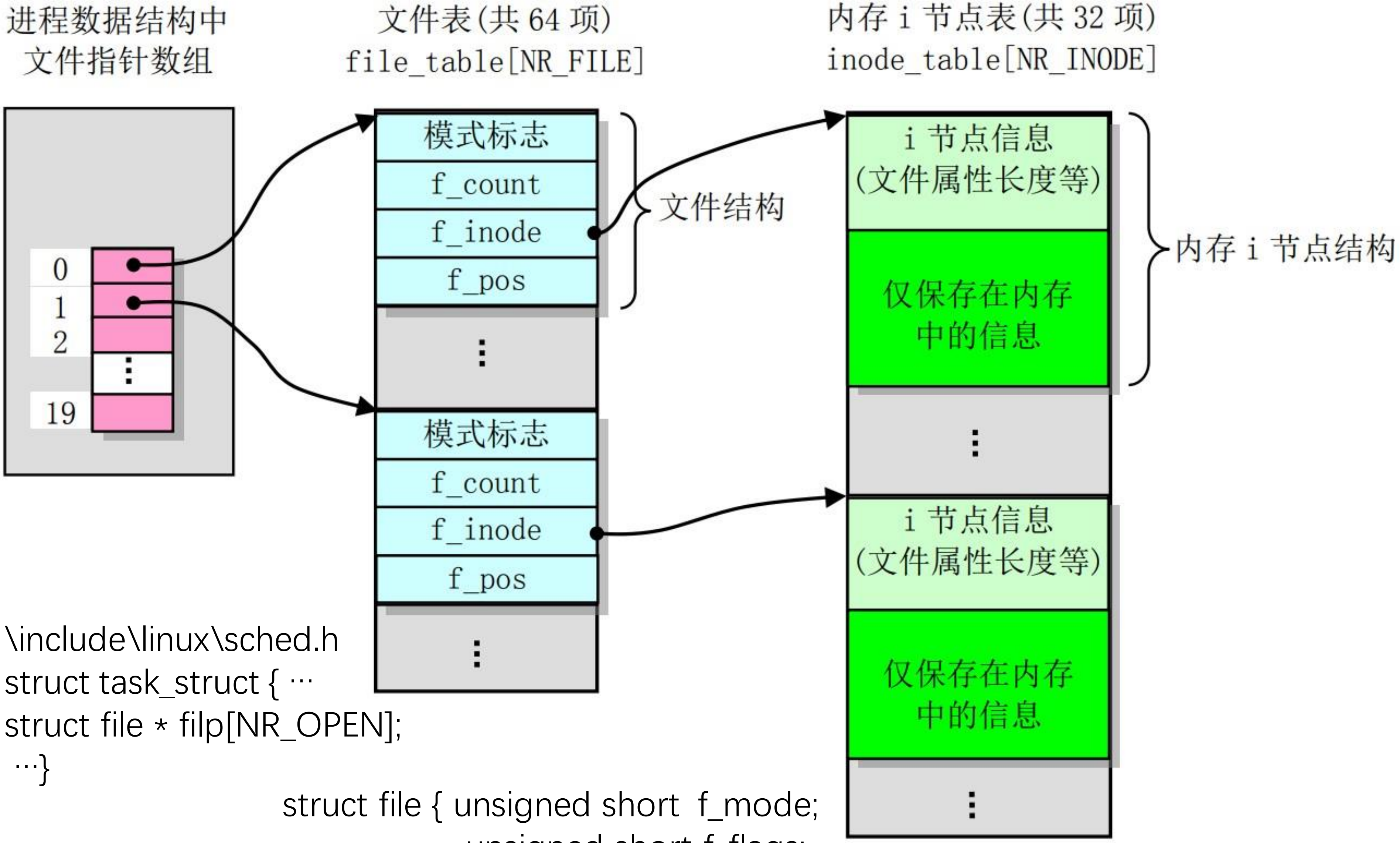


```
[/]# hexdump .
0000000 0001 002e 0000 0000 0000 0000 0000
0000010 0001 2e2e 0000 0000 0000 0000 0000
0000020 0344 6d49 6761 0065 0000 0000 0000
0000030 0003 6962 006e 0000 0000 0000 0000
0000040 0037 6564 0076 0000 0000 0000 0000
0000050 004b 7465 0063 0000 0000 0000 0000
0000060 005a 6d69 6761 0065 0000 0000 0000
0000070 00a2 6e6d 0074 0000 0000 0000 0000
0000080 0143 6873 656f 616c 6563 0000 0000
0000090 0144 6d74 0070 0000 0000 0000 0000
00000a0 0146 7375 0072 0000 0000 0000 0000
00000b0 03bb 6176 0072 0000 0000 0000 0000
00000c0
[/]#
```

```
/include/linux/fs.h
#define NAME_LEN 14
#define ROOT_INO 1
struct dir_entry {
        unsigned short inode;
        char name[NAME_LEN];
};
```



```
[/]# hexdump /etc
0000000 004b 002e 0000 0000 0000 0000 0000 0000
0000010 0001 2e2e 0000 0000 0000 0000 0000 0000
0000020 004c 6372 0000 0000 0000 0000 0000 0000
0000030 004d 7075 6164 6574 0000 0000 0000 0000
0000040 004e 6574 6d72 6163 0070 0000 0000 0000
0000050 004f 746d 6261 0000 0000 0000 0000 0000
0000060 0050 616c 6563 7075 0000 0000 0000 0000
0000070 0051 616d 6967 0063 0000 0000 0000 0000
0000080 0052 7270 666f 6c69 0065 0000 0000 0000
0000090 0053 6170 7373 6477 0000 0000 0000 0000
00000a0 0054 7267 756f 0070 0000 0000 0000 0000
00000b0 0055 746d 6f6f 736c 0000 0000 0000 0000
00000c0 0056 6f63 666e 6769 0000 0000 0000 0000
00000d0 0057 6964 6b73 6174 0062 0000 0000 0000
00000e0 0058 6f62 746f 616c 6563 0000 0000 0000
00000f0 0059 6977 696e 6f62 746f 0000 0000 0000
0000100
```



进程数据结构中
文件指针数组

0
1
2
⋮
19

文件表(共 64 项)
file_table[NR_FILE]

模式标志
f_count
f_inode
f_pos
⋮
模式标志
f_count
f_inode
f_pos
⋮

文件结构

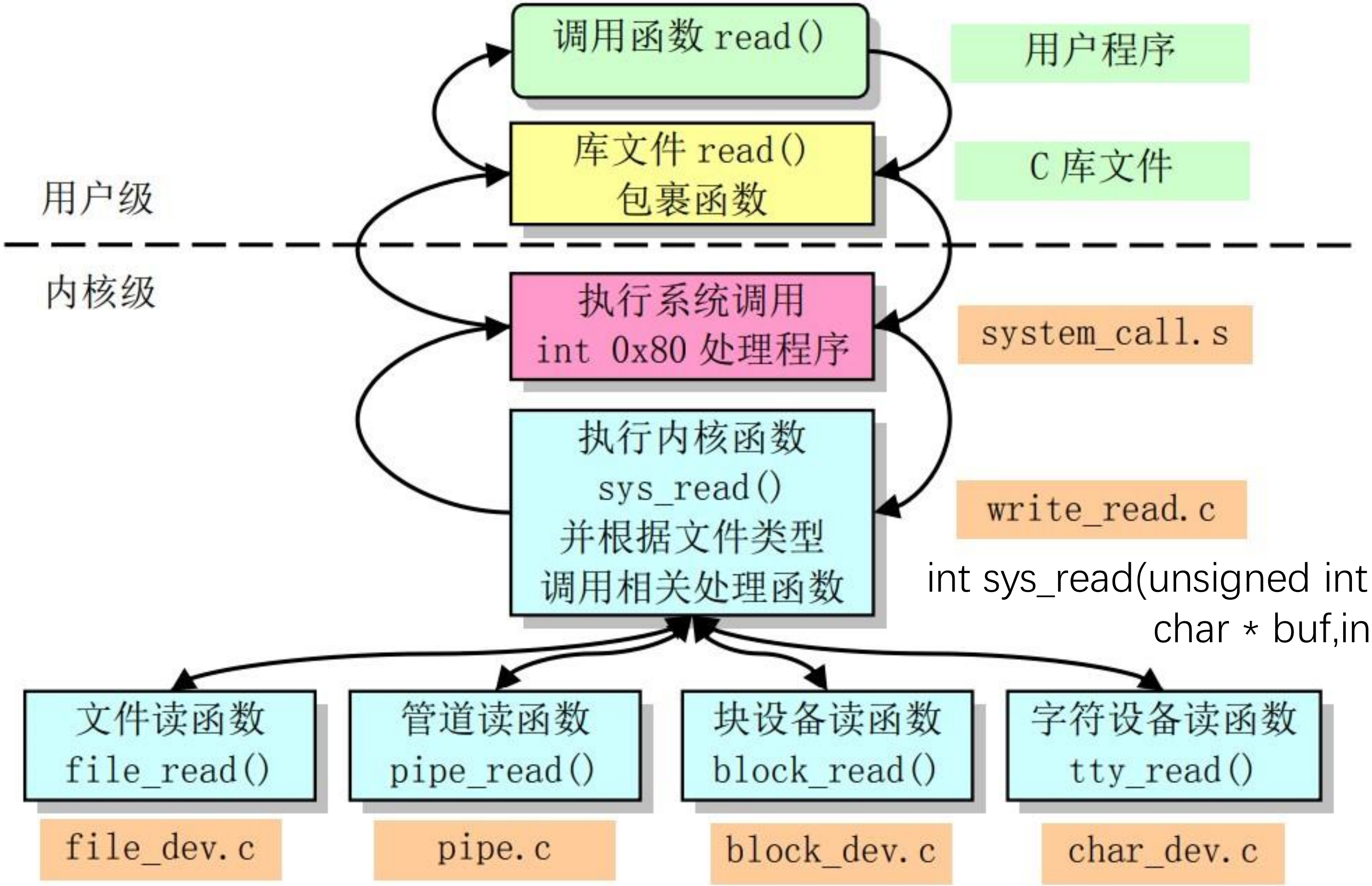
内存 i 节点表(共 32 项)
inode_table[NR_INODE]

i 节点信息
(文件属性长度等)
仅保存在内存
中的信息
⋮
i 节点信息
(文件属性长度等)
仅保存在内存
中的信息
⋮

内存 i 节点结构

```
\include\linux\sched.h
struct task_struct { …
struct file * filp[NR_OPEN];
 …}
```

```
struct file { unsigned short  f_mode;
              unsigned short f_flags;
              unsigned short f_count;
              struct m_inode * f_inode
              off_t f_pos; };
struct file file_table[NR_FILE]
```



调用函数 read()

用户程序

库文件 read()
包裹函数

C 库文件

用户级

内核级

执行系统调用
int 0x80 处理程序

system_call.s

执行内核函数
sys_read()
并根据文件类型
调用相关处理函数

write_read.c

int sys_read(unsigned int fd,
char * buf,int count)

文件读函数
file_read()

file_dev.c

管道读函数
pipe_read()

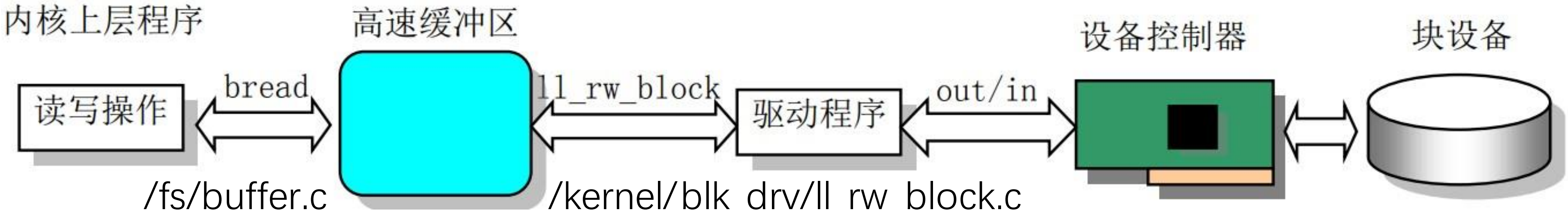
pipe.c

块设备读函数
block_read()

block_dev.c

字符设备读函数
tty_read()

char_dev.c



内核上层程序

读写操作

bread

高速缓冲区

ll_rw_block

驱动程序

out/in

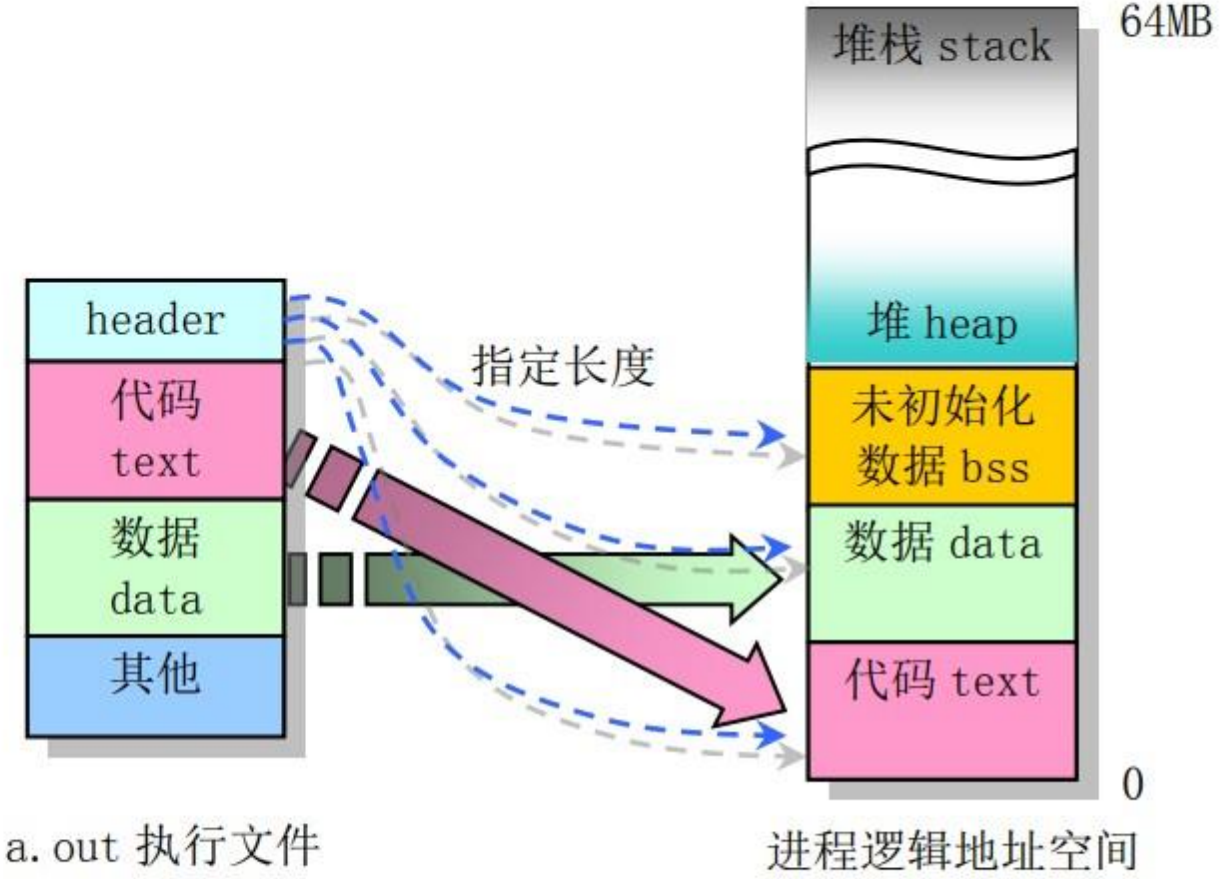
设备控制器

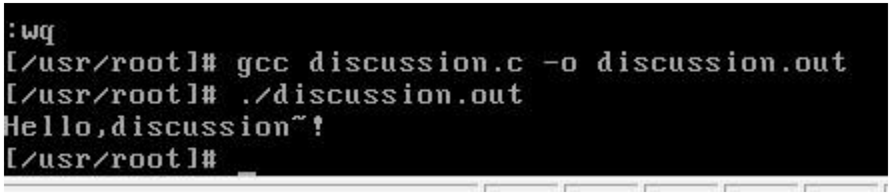
块设备

/fs/buffer.c

/kernel/blk_drv/ll_rw_block.c

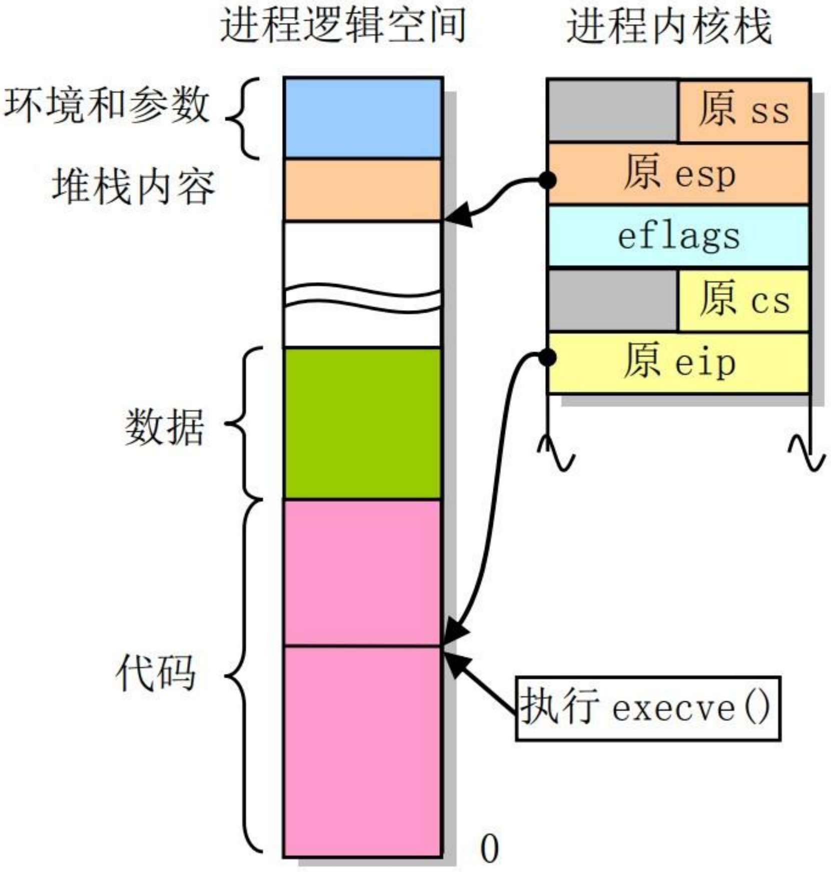
```
/linux/include/a.out.h
struct exec {
        unsigned long a_magic
        unsigned long a_text
        unsigned long a_data
        unsigned long a_bss
        unsigned long a_syms
        unsigned long a_entry
        unsigned long a_trsize
        unsigned long a_drsize
};
```



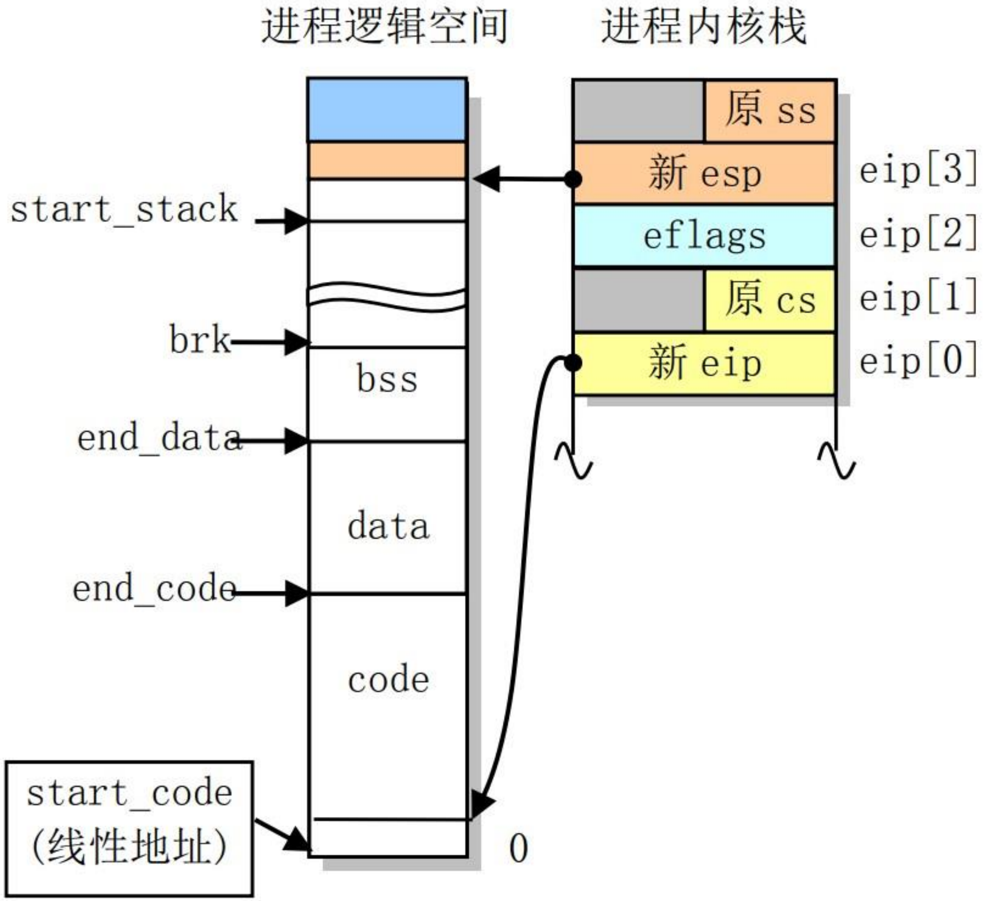


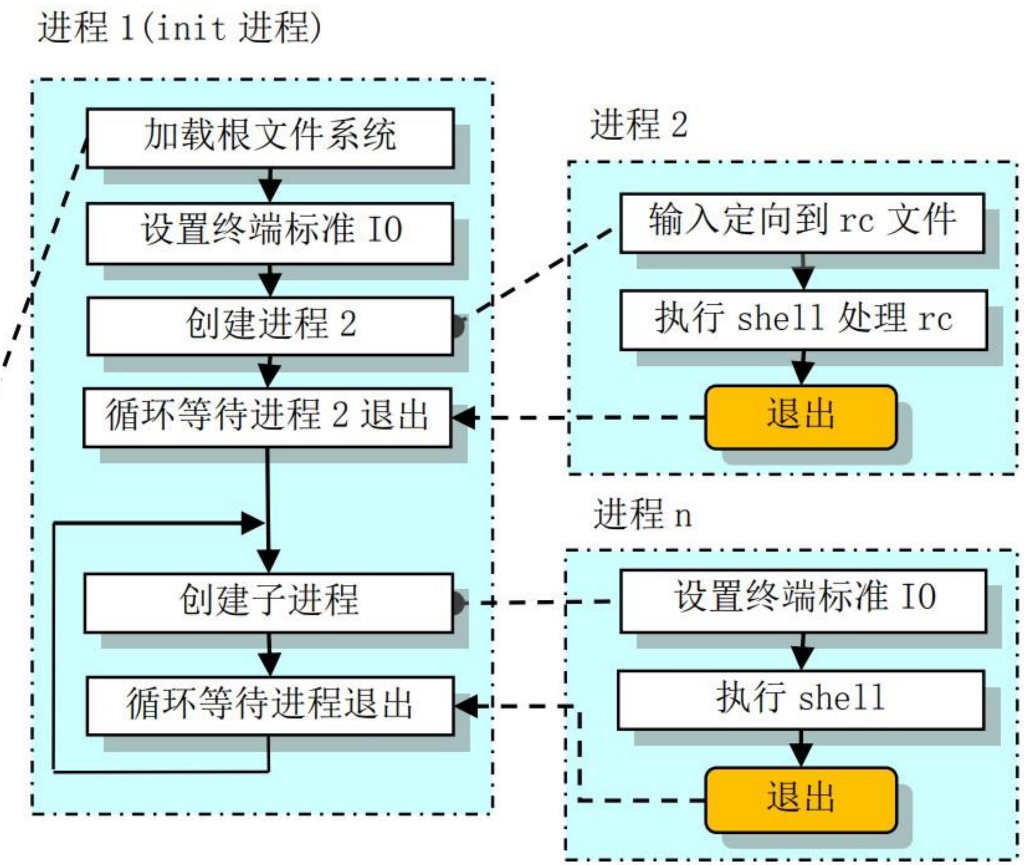
```
:wq
[/usr/root]# gcc discussion.c -o discussion.out
[/usr/root]# ./discussion.out
Hello,discussion~!
[/usr/root]# _
```

程序调用 execve()刚进入系统调用时

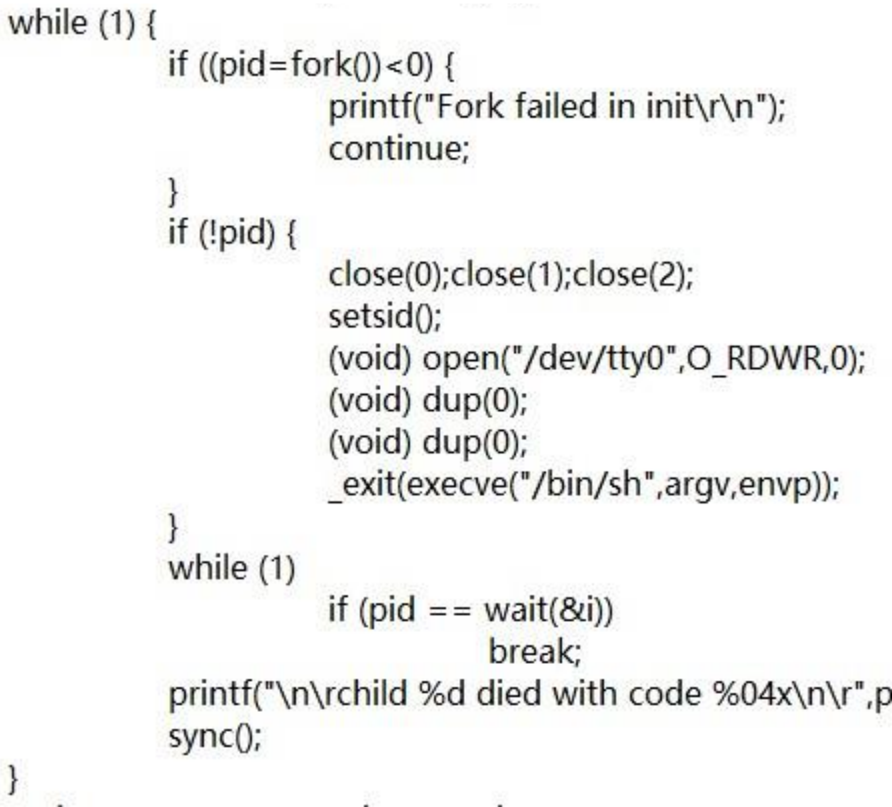


函数 execve()加载新执行文件返回时





/init/main.c



```
while (1) {
        if ((pid=fork())<0) {
                printf("Fork failed in init\r\n");
                continue;
        }
        if (!pid) {
                close(0);close(1);close(2);
                setsid();
                (void) open("/dev/tty0",O_RDWR,0);
                (void) dup(0);
                (void) dup(0);
                _exit(execve("/bin/sh",argv,envp));
        }
        while (1)
                if (pid == wait(&i))
                        break;
        printf("\n\rchild %d died with code %04x\n\r",p
        sync();
}
```