Linux操作系统编程

Linux标准I/O

标准I/O库

- **为什么要设计标准I/O库?**
  - 直接使用API进行文件访问时，需要考虑许多细节问题，例如：read、write时，缓冲区的大小该如何确定，才能使效率最优
  - read和write等底层系统调用函数进行输入输出时，在用户态和内核态之间来回切换，每次读出或写入的数据量较少，导致频繁的I/O操作，增加了系统开销
- **标准I/O库是ANSI C规范的一部分，函数原型在文件stdio.h中定义，对底层I/O系统调用进行了封装，为程序员提供了带有格式转换功能的输入输出操作，并在用户空间增加了缓冲区管理**

标准I/O库

- **分离了应用程序空间和实际的物理设备**
- **减少了直接读盘次数，提高性能**
  - 读取前查看是否已存在页缓存中，如果已经存放在了页缓存中，数据立即返回给应用程序
  - 写数据前先写到页缓存中，如果用户采用的是同步写机制（synchronous writes），那么数据会立即被写回到磁盘上，应用程序会一直等到数据被写完为止；如果用户采用的是延迟写机制（deferred writes），那么应用程序就完全不需要等到数据全部被写回到磁盘，数据只要被写到页缓存中去就可以了。

fopen函数

- **fopen函数功能**
  打开一个指定文件
- **函数原型**
  FILE *fopen(const char *restrict pathname, const char *restrict type);
- **参数**
  pathname：要打开的文件名　　type：指定文件的读、写方式

type	说明
r或rb	为读而打开
w或wb	使文件长度为0，或为写而创建
a或ab	添加；为在文件尾写而打开，或为写而创建
r+或r+b或rb+	为读和写而打开
w+或w+b或wb+	使文件长度为0，或为读和写而打开
a+或a+b或ab+	为在文件尾读和写而打开或创建

fopen函数

限制	r	w	a	r+	w+	a+
文件必须存在	√			√		
删除文件以前内容		√			√	
流可以读	√			√	√	√
流可以写		√	√	√	√	√
流只在尾端处写			√			√

setbuf和setvbuf函数

- **setbuf和setvbuf函数功能**
  打开和关闭缓冲机制
- **函数原型**
  void setbuf(FILE *steam, char *buf);
  void setvbuf(FILE *steam, char *buf, int mode, size_t size);

函数	mode	buf	缓存及长度	缓存的类型
setbuf		nonnull	长度为BUFSIZ的用户缓存	全缓存
		NULL	（无缓存）	不带缓存
setvbuf	_IOFBF	nonnull	长度为size的用户缓存	全缓存
		NULL	合适长度的系统缓存	
	_IOLBF	nonnull	长度为size的用户缓存	行缓存
		NULL	合适长度的系统缓存	
	_IONBF	忽略	无缓存	不带缓存

fdopen函数

- **fdopen函数功能**
  取一个现存的文件描述符，并使一个标准I/O流与该描述符相结合
- **头文件**
  #include<stdio.h>
- **函数原型**
  FILE *fdopen(int fd, const char *type);
- **fdopen常用于由创建管道及网络通信通道函数返回的描述符。**
  - 这些特殊类型的文件，不能用fopen打开
  - 因此必须先调用设备专用函数以获得一个文件描述符，然后再用fdopen使一个标准I/O流与该描述符相关联
- **对于fdopen函数，type参数的意义稍有区别**
  - 因为该描述符已被打开，所以fdopen为写而打开并不截短该文件
  - 不能用于创建该文件（因为如若一个描述符引用一个文件，则该文件一定已经存在）

fdopen函数

Type值	操作文件类型	是否新建文件	是否清空原文件	可读	可写	读写开始位置
r	文本文件	NO	NO	YES	NO	文件开头
r+	文本文件	YES	NO	YES	YES	文件开头
w	文本文件	YES	YES	NO	YES	文件开头
w+	文本文件	YES	YES	YES	YES	文件开头
a	文本文件	NO	YES	NO	YES	文件结尾
a+	文本文件	NO	YES	YES	YES	文件结尾
rb	二进制文件	NO	NO	YES	NO	文件开头
r+b或rb+	二进制文件	YES	NO	YES	YES	文件开头
wb	二进制文件	YES	YES	NO	YES	文件开头
w+b或wb+	二进制文件	YES	YES	YES	YES	文件开头
ab	二进制文件	NO	YES	NO	YES	文件结尾
a+b或ab+	二进制文件	NO	YES	YES	YES	文件结尾

fdopen函数示例

```
FILE *fp;
int fd;
if ((fp = fopen("hello.txt", "w+")) == NULL) {
    printf("fopen file error\n");
    return 0;}
fprintf(fp, "hello word\n");
fclose(fp);
if ((fd = open("hello.txt", O_RDWR)) == -1) {
    printf("open file fail\n");
    return 0;}
if ((fp = fdopen(fd, "a+")) == NULL) {
    printf("fdopen open\n");
    return 0;
}
fprintf(fp, "linux c program");
fclose(fp);
```