指向获取的文件信息将要存放的结构的指针，该结构的各个域如图10-28所示。系统调用fstat的作用和stat一样，惟一不同的是，fstat针对一个打开的文件（文件名可能未知）进行操作，而不是一个路径名。

存储文件的设备
i节点号（哪个文件在设备上）
文件模式（包括保护信息）
指向文件的连接数
文件所有者的标识
文件所属的组
文件大小（单位是字节）
创建时间
最近访问的时间
最近修改的时间

图10-28 stat系统调用返回的域

pipe系统调用用来创建一个shell管线。它创建了一种伪文件（pseudo-file），用于缓冲管线通信的数据，并给缓冲区的读写都返回文件描述符。以下面的管线为例：

```
sort <in | head –30
```

在执行sort的进程中，文件描述符1（标准输出）被设置为写入管道，执行head的进程中，文件描述符0（标准输入）被设置为从管道读取。通过这种方式，sort只是从文件描述符0（被设置为文件in）读取，写入到文件描述符1（管道），甚至不会觉察到它们已经被重定向了。如果它们没有被重定向，sort将会自动从键盘读取数据，而后输出到显示器（默认设备）。同样地，当head从文件描述符0中读取数据时，它读取到的是sort写入到管道缓冲区中的数据，head甚至不知道自己使用了管道。这个例子清晰地表明了一个简单的概念（重定向）和一个简单的实现（文件描述符0和1）如何实现一个强大的工具（以任意方式连接程序，而不需要去修改它们）。

图10-27列举的最后一个系统调用是fcntl。fcntl用于加锁和解锁文件，应用共享锁和互斥锁，或者是执行一些文件相关的其他操作。

现在我们开始关注与目录及文件系统整体更加相关，而不是仅和单个文件有关的系统调用，图10-29列举了一些这样的系统调用。可以使用mkdir和rmdir创建和删除目录，但需要注意：只有目录为空时才可以将其删除。

系统调用	描述
s=mkdir (path, mode)	建立新目录
s=rmdir (path)	删除目录
s=link (oldpath, newpath)	创建指向已有文件的链接
s=unlink (path)	取消文件的链接
s=chdir (path)	改变工作目录
dir=opendir (path)	打开目录
s=closedir (dir)	关闭目录
dirent=readdir (dir)	读取一个目录项
rewinddir (dir)	回转目录使其再次被读取

图10-29 与目录相关的一些系统调用。如果发生错误，那么返回值s是-1，dir是一个目录流，dirent是一个目录项。参数的含义是自解释的

如图10-24所示，创建一个指向已有文件的链接时创建了一个目录项（directory entry）。系统调用link用于创建链接，它的参数是已有文件的文件名和链接的名称，使用unlink可以删除目录项。当文件的最后一个链接被删除时，这个文件会被自动删除。对于一个没有被链接的文件，对其使用unlink也会让它从目录中消失。

使用chdir系统调用可以改变工作目录，工作目录的改变会影响到相对路径名的解释。

图10-29给出的最后四个系统调用是用于读取目录的。和普通文件类似，它们可被打开、关闭和读取。每次调用readdir都会以固定的格式返回一个目录项。用户不能对目录执行写操作（为了保证文件系统的完整性），但可以使用creat或link在文件夹中创建一个目录，或使用unlink删除一个目录。同样地，用户不能在目录中查找某个特定文件，但是可以使用rewinddir作用于一个打开的目录，使得它能再次从头读取。

10.6.3 Linux文件系统的实现

在本节中，我们首先研究虚拟文件系统（Virtual File System，VFS）层支持的抽象。VFS对高层进程和应用程序隐藏了Linux支持的所有文件系统之间的区别，以及文件系统是存储在本地设备，还是需