第二十二章

1. 核心转储文件：

核心转储文件是内含进程终止时内存印象的一个文件。可由特定信号引发。

介绍了不产生转储文件的原因。

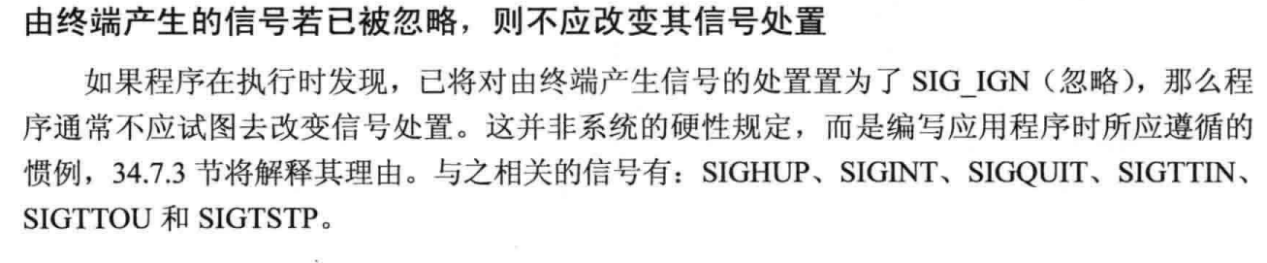
以及如何为转储文件命名的方法。

1. 传递处置和处理的特殊情况：

SIGKILL和SIGSTOP的默认行为无法改变，也不能阻塞。

SIGCONT不受阻塞和忽略的影响。

遇到的问题：



1. 可中断和不可中断进程的睡眠状态

上述第二点SIGKILL和SIGSTOP立竿见影的效果有一个前提，即进程处于可中断的休眠状态。

TASK\_INTERRUPTIBLE和TASK\_UNINTERRUPTIBLE，前者如等待信号量，终端输入。后者如磁盘IO。为了防止后者出现问题只能导致重启解决，出现了TASK\_KILLABLE。类似于后者，但是收到致命信号会唤醒。

1. 硬件产生的信号：

在硬件异常的情况下，进程从此类信号处理函数中返回，或忽略或阻塞信号，会导致未定义的行为。正确的处理方式是：默认行为或者信号处理函数里终止进程。

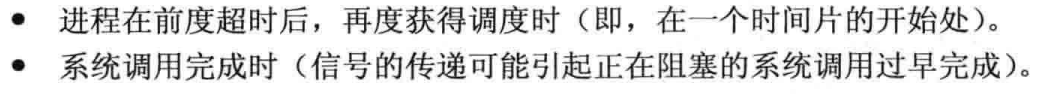
1. 信号同步和异步：

自己发送给自己或者硬件异常为同步，其它为异步。同步信号会立即传递。

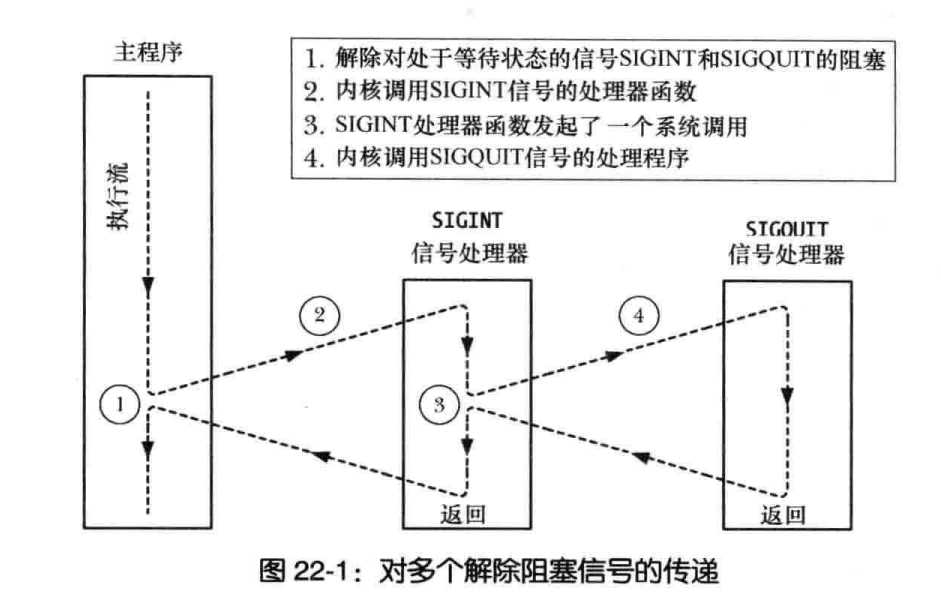
1. 传递信号的时机和顺序

对于时机而言，同步信号会立即传递，异步信号实际产生和传递可能存在时延，处于短暂的等待状态。内核将等待信号传递给进程是在：该进程正在执行，且发生由内核态到用户态的下一次切换时。

即：



当解除信号阻塞的时候，Linux默认顺序是按照信号增序传递信号，与发送顺序无关。不能对标准信号顺序产生依赖。而实时信号顺序得到保障。

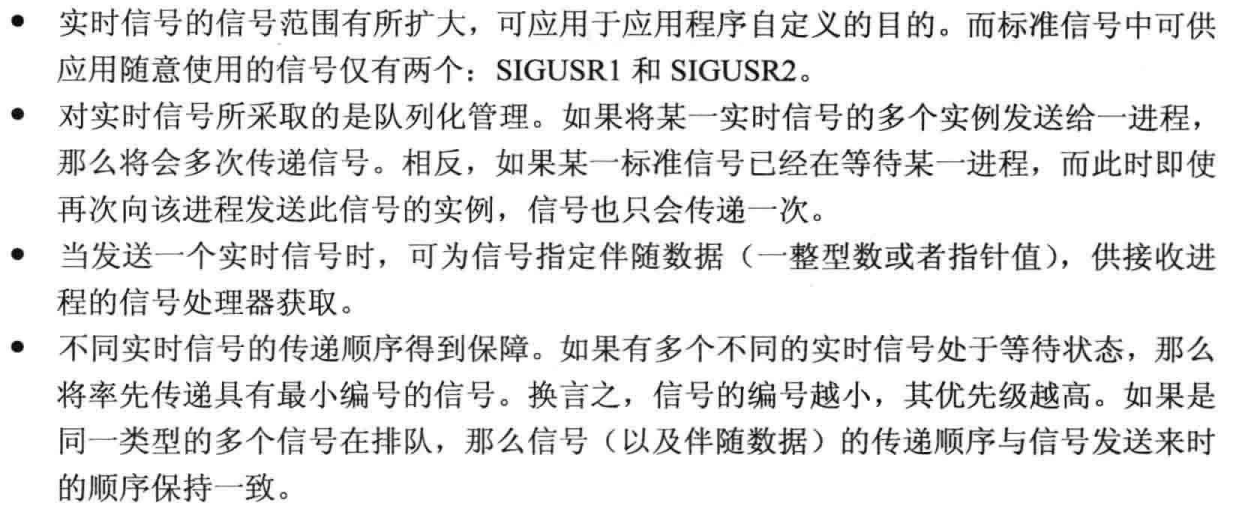


1. Signal实现及可移植性

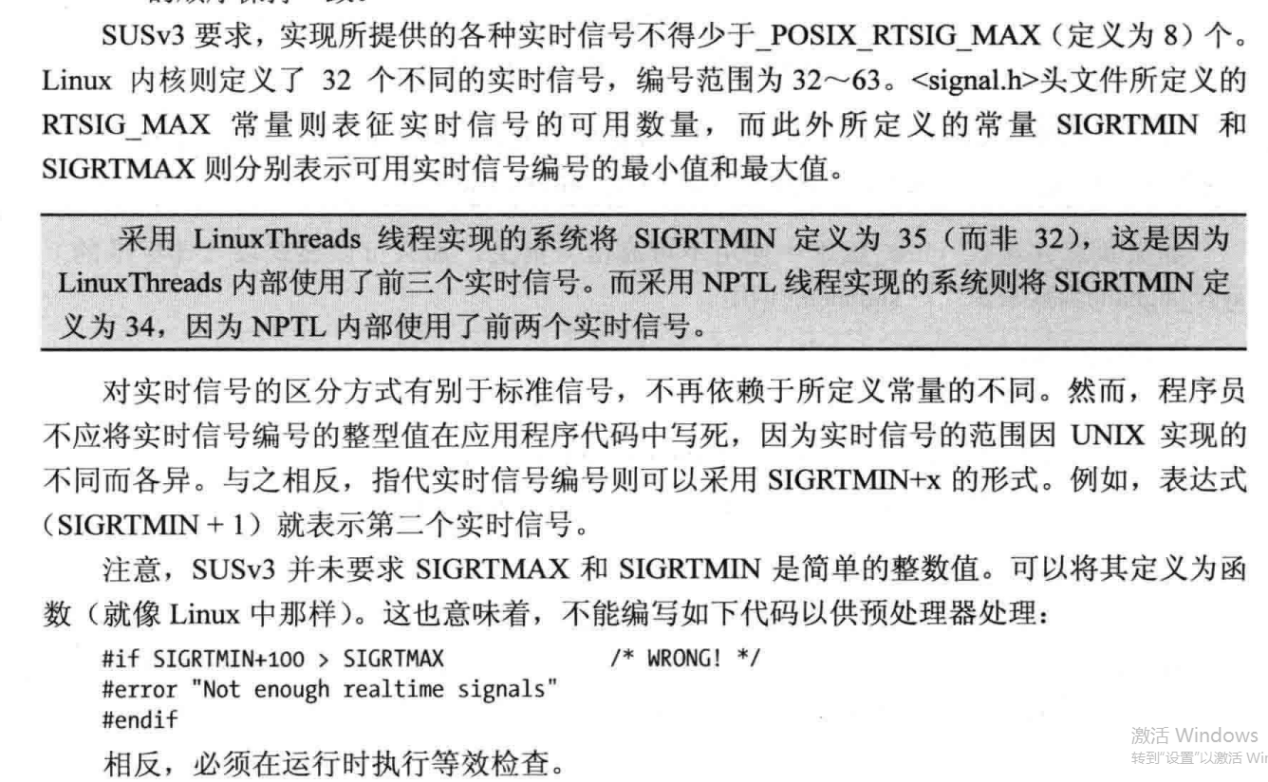
Signal由于历史原因，导致具有不可移植性，因此设置信号处理函数的时候使用sigaction比较好。当然，SIG\_DEF,SIG\_IGN仍然可以继续使用。

1. 实时信号

实时信号弥补了标准信号的诸多限制：

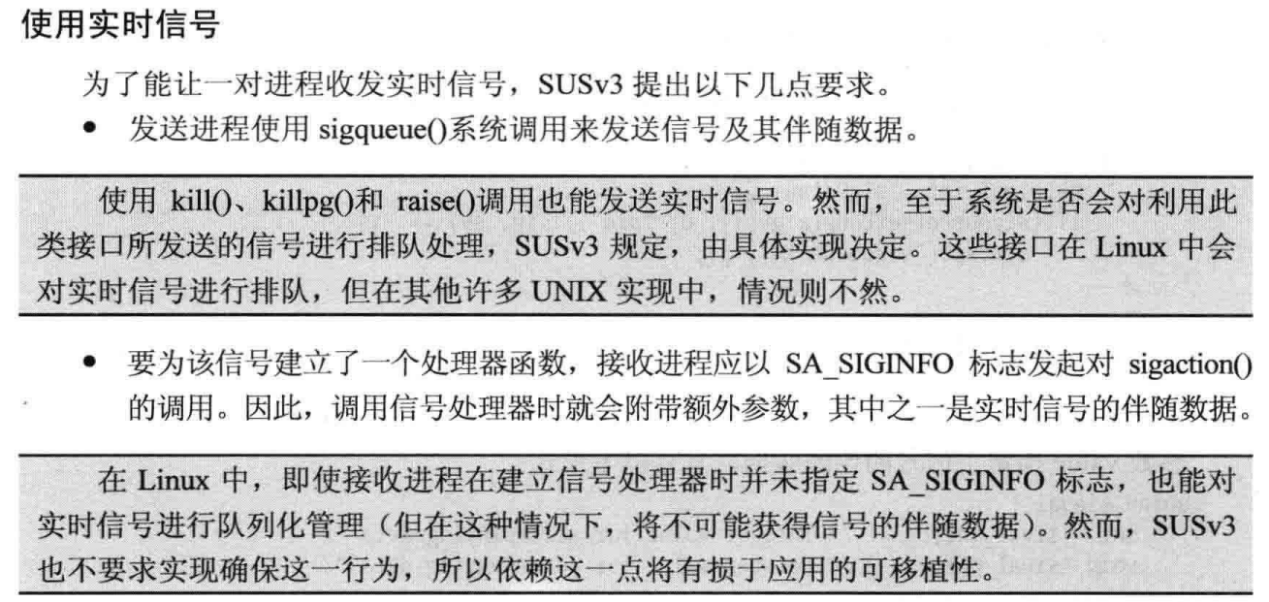


实时信号个数：

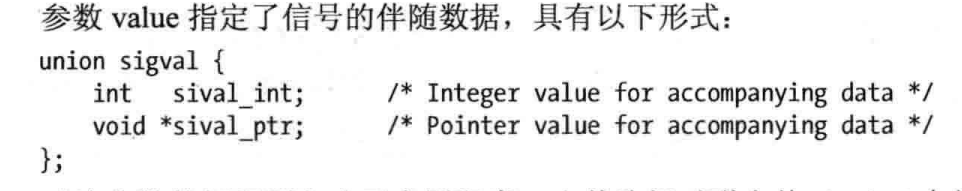


由于需要排队，所以需要数据结构维护，因此会对其大小限制。具体查看书本。

如何使用实时信号呢？

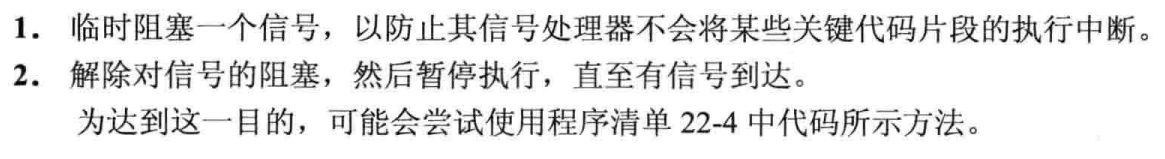




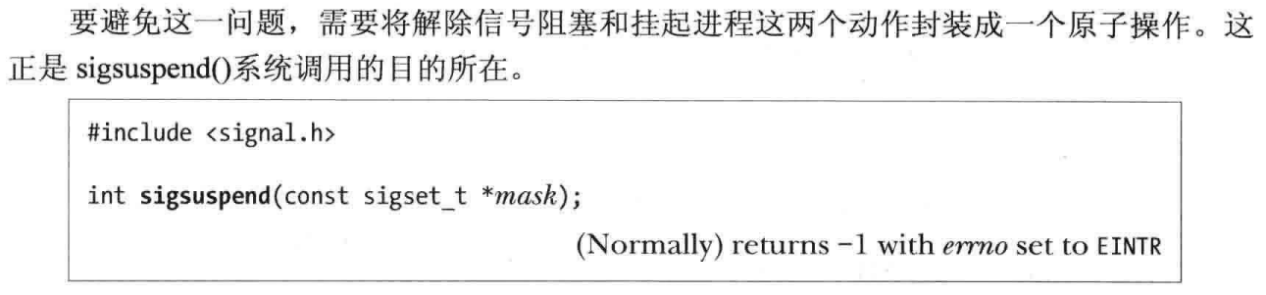


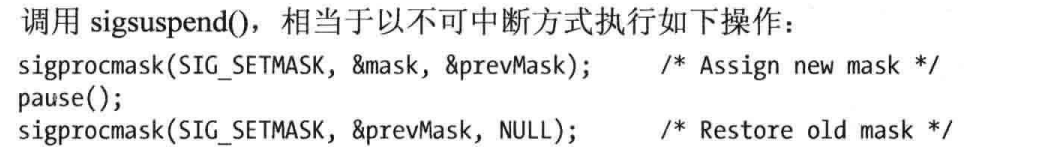
1. 使用掩码来等待信号

编程中遇到的问题：

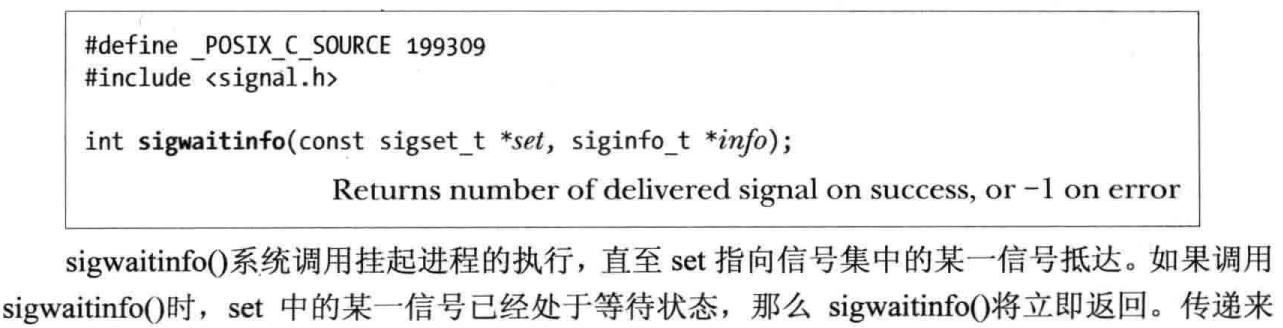


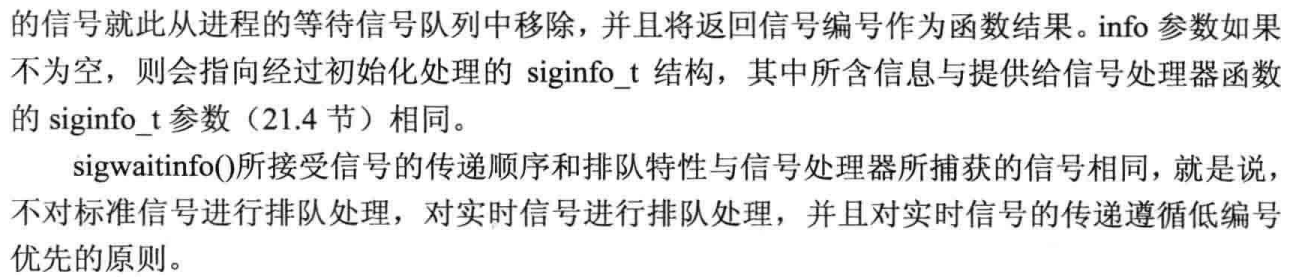
如果单纯使用sigprocmask和pause实现的话，会在后一个sigprocmask和pause之间出现空隙，使得竞争发生。这个时候如果把这两个操作实现为一个序列化，则可以成功。



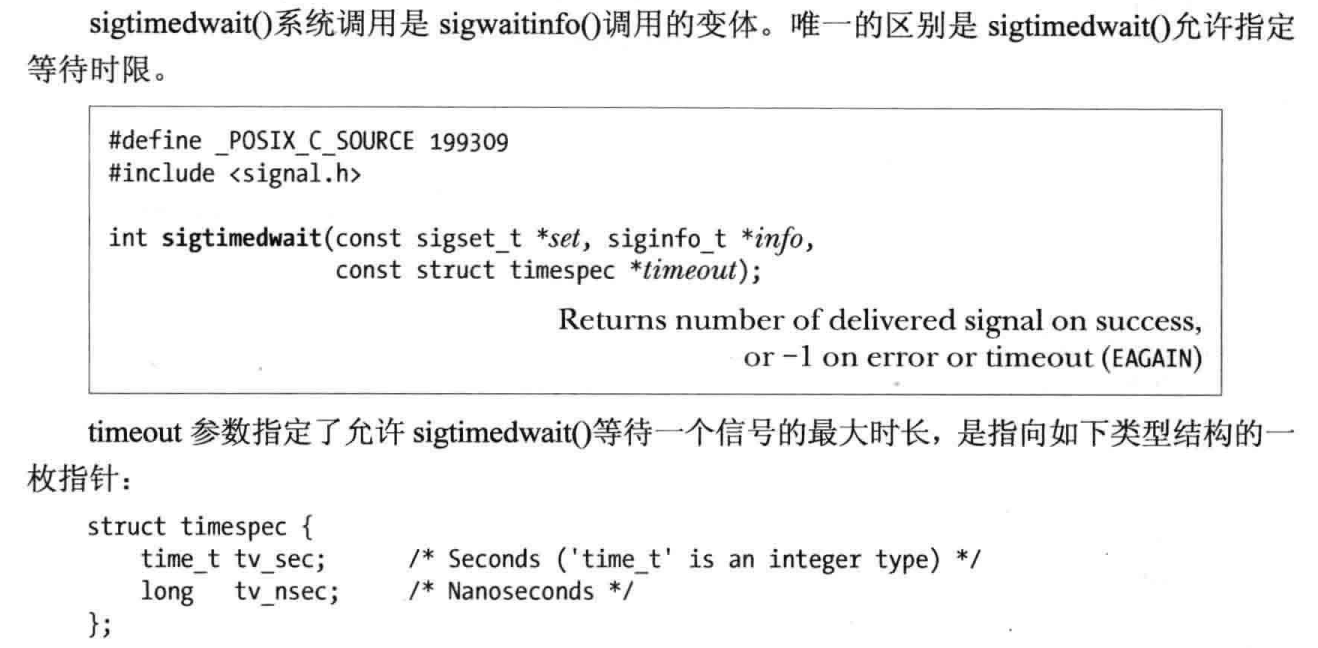


1. 以同步方式等待信号

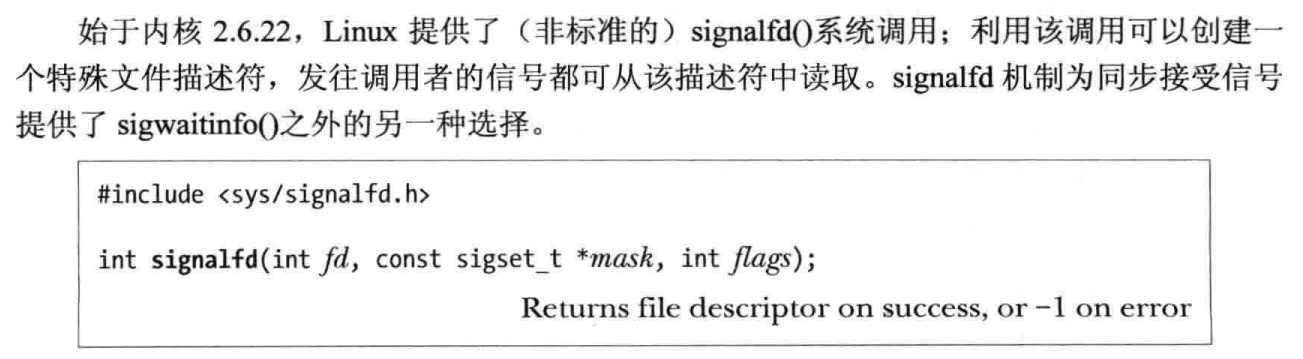




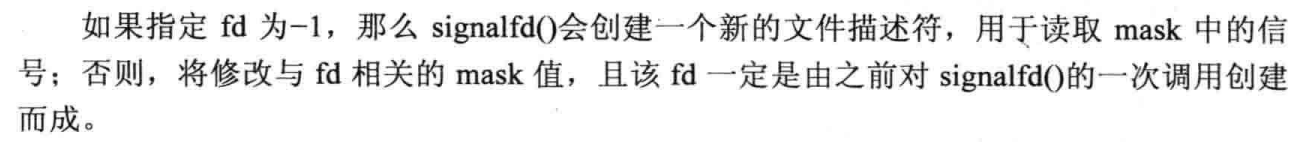
注意，上述这个函数一般与sigprocmask函数一起使用。

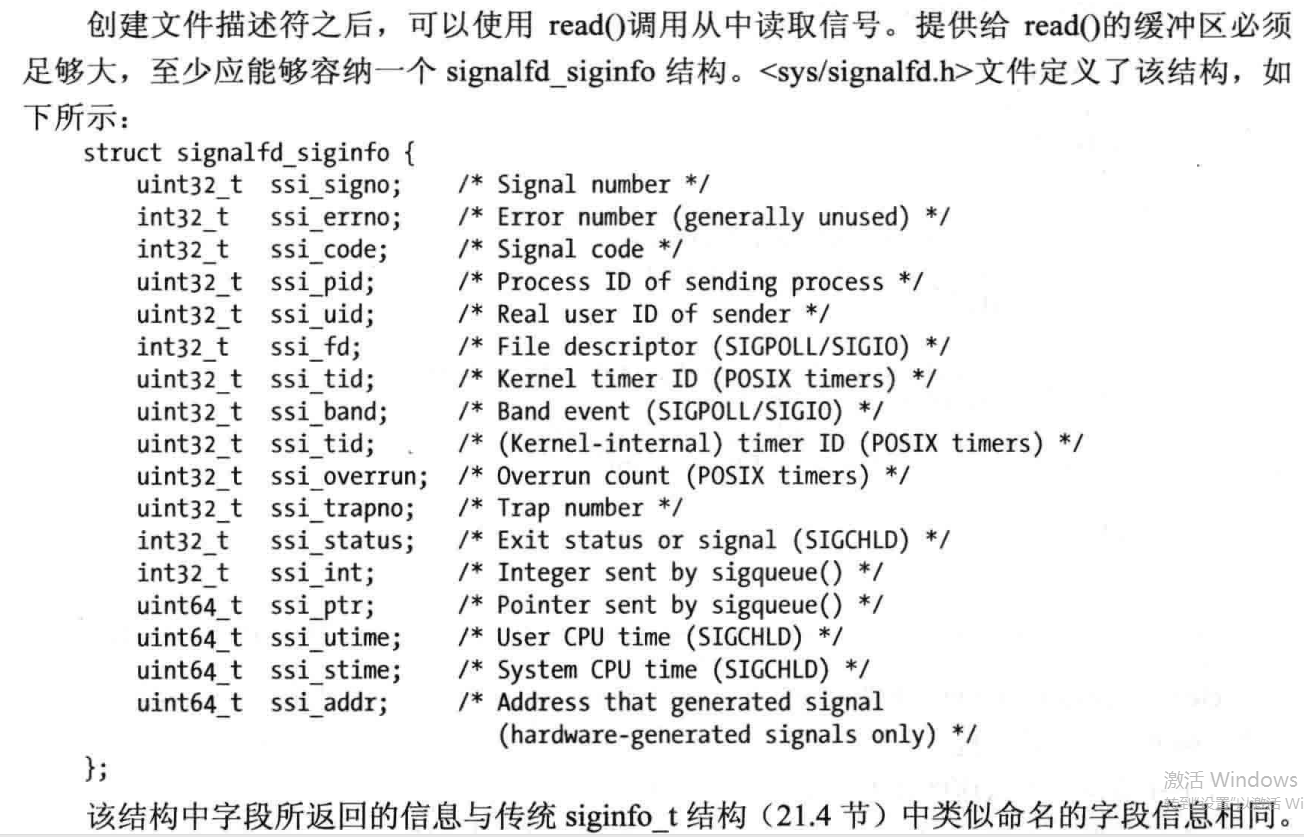


1. 通过文件描述符来获取信号



注意，此函数也要与sigprocmask函数一起使用。





1. 利用信号进行进程间通信

由于信号的诸多限制，一般不用此方法进行进程间的通信。

