进程间通信（IPC，InterProcess Communication）

概念：就是进程和进程之间交换信息。

常用通信方式

无名管道（pipe）

有名管道 （fifo）

信号（signal）

共享内存(mmap)

套接字（socket）

过时的IPC通信方式

System V IPC

共享内存（share memory）

消息队列（message queue）

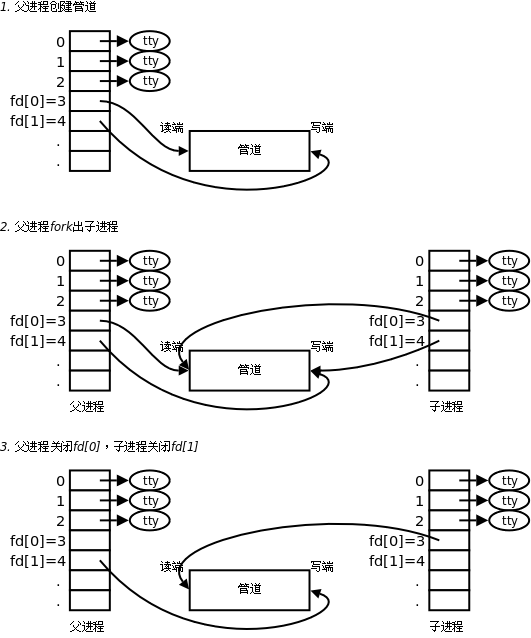
信号灯集（semaphore set）

无名管道

int pipe(int pfd[2]); 成功：0；失败：-1，设置errno

pfd[0] 为读描述符

pfd[1] 为写描述符



**无名管道注意事项：**

1. 只能用于亲缘关系的进程间通信（父子进程，兄弟进程）
2. 管道通信是单工的，一端读，一端写（程序实现设计好）。
3. 数据自己读不能自己写
4. 管道可以用于大于2个进程共享

**无名管道的读写特性：**

1. 读管道：

1. 管道中有数据，read返回实际读到的字节数。

2. 管道中无数据：

(1) 管道写端被全部关闭，read返回0 (好像读到文件结尾)

(2) 写端没有全部被关闭，read阻塞等待(不久的将来可能有数据递达，此时会让出cpu)

1. 写管道：

1. 管道读端全部被关闭， 进程异常终止(也可使用捕捉SIGPIPE信号，使进程不终止)

2. 管道读端没有全部关闭：

(1) 管道已满，write阻塞。（管道大小64K）

1. 管道未满，write将数据写入，并返回实际写入的字节数。

有名管道（命名管道）

创建管道

#include <sys/types.h>

#include <sys/stat.h>

int mkfifo(const char \*filename, mode\_t mode);

open(const char \*path, O\_RDONLY);//1

open(const char \*path, O\_RDONLY | O\_NONBLOCK);//2

open(const char \*path, O\_WRONLY);//3

open(const char \*path, O\_WRONLY | O\_NONBLOCK);//4

特点：

1有名管道可以使非亲缘的两个进程互相通信

2通过路径名来操作，在文件系统中可见，但内容存放在内存中

3 文件IO来操作有名管道

4 遵循先进先出规则

5 不支持leek操作

6 单工读写

注意事项：

1 就是程序不能以O\_RDWR(读写)模式打开FIFO文件进行读写操作，而其行为也未明确定义，因为如一个管道以读/写方式打开，进程可以读回自己的输出，同时我们通常使用FIFO只是为了单向的数据传递

2 第二个参数中的选项O\_NONBLOCK，选项O\_NONBLOCK表示非阻塞，加上这个选项后，表示open调用是非阻塞的，如果没有这个选项，则表示open调用是阻塞的

3 对于以只读方式（O\_RDONLY）打开的FIFO文件，如果open调用是阻塞的（即第二个参数为O\_RDONLY），除非有一个进程以写方式打开同一个FIFO，否则它不会返回；如果open调用是非阻塞的的（即第二个参数为O\_RDONLY | O\_NONBLOCK），则即使没有其他进程以写方式打开同一个FIFO文件，open调用将成功并立即返回。

对于以只写方式（O\_WRONLY）打开的FIFO文件，如果open调用是阻塞的（即第二个参数为O\_WRONLY），open调用将被阻塞，直到有一个进程以只读方式打开同一个FIFO文件为止；如果open调用是非阻塞的（即第二个参数为O\_WRONLY | O\_NONBLOCK），open总会立即返回，但如果没有其他进程以只读方式打开同一个FIFO文件，open调用将返回-1，并且FIFO也不会被打开。

4.数据完整性,如果有多个进程写同一个管道，使用O\_WRONLY方式打开管道，如果写入的数据长度小于等于PIPE\_BUF（4K），那么或者写入全部字节，或者一个字节都不写入，系统就可以确保数据决不会交错在一起。