**有名信号量一般用于进程间同步或互斥**。

所需头文件：

#include <fcntl.h>

#include <sys/stat.h>

#include <semaphore.h>

**创建一个有名信号量**

当有名信号量文件存在时使用：

sem\_t \*sem\_open(const char \*name, int oflag);

当有名信号量文件不存在时使用（当然，如果存在也会正常返回）：

sem\_t \*sem\_open(const char \*name, int oflag, mode\_t mode, unsigned int value);

功能：

创建一个有名信号量。

参数：

**name**：信号量文件名。注意，不能指定路径。因为有名信号量，默认放在**/dev/shm** 里，

**flags**：sem\_open() 函数的行为标志。

**mode**：文件权限(可读、可写、可执行)的设置。

**value**：信号量初始值。

返回值：

成功：信号量的地址

失败：SEM\_FAILED

**关闭有名信号量**

int sem\_close(sem\_t \*sem);

功能：

关闭有名信号量。

参数：

**sem**：指向信号量的指针。

返回值：

成功：0

失败：-1

**删除有名信号量文件**

int sem\_unlink(const char \*name);

功能：

删除有名信号量的文件。

参数：

**name**：有名信号量文件名。

返回值：

成功：0

失败：-1

**信号量 PV 操作**

[用法和POSIX 无名信号量](http://blog.csdn.net/tennysonsky/article/details/46496201)一样

**有名信号量实现进程间互斥功能示例**

#include <stdio.h>

#include <semaphore.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/stat.h>

#include <sys/types.h>

#include <sys/wait.h>

void printer(sem\_t \*sem, char \*str)

{

    sem\_wait(sem);  //信号量减一

    while(\*str!='\0')

    {

        putchar(\*str);

        fflush(stdout);

        str++;

        sleep(1);

    }

    printf("\n");

    sem\_post(sem);  //信号量加一

}

int main(int argc, char \*argv[])

{

    pid\_t pid;

    sem\_t \*sem = NULL;

    pid = fork(); //创建进程

    if(pid<0){ //出错

        perror("fork error");

        return 0;

    }

    else if(pid == 0){

        //子进程

        // 从信号量文件获取信号量，不存在则创建 值为 1

        sem = sem\_open("name\_sem", O\_CREAT|O\_RDWR, 0666, 1);

        //有名信号量创建失败

        if(sem == SEM\_FAILED){

            perror("sem\_open");

            return -1;

        }

        char \*str1 = "hello";

        printer(sem, str1); //打印

        sem\_close(sem); //关闭有名信号量

        \_exit(1);

    }

    else if(pid > 0){

        //父进程

        // 从信号量文件获取信号量，不存在则创建 值为 1

        sem = sem\_open("name\_sem", O\_CREAT|O\_RDWR, 0666, 1);

        //有名信号量创建失败

        if(sem == SEM\_FAILED){

            perror("sem\_open");

            return -1;

        }

        char \*str2 = "world";

        printer(sem, str2); //打印

        sem\_close(sem); //关闭有名信号量

        waitpid(pid, NULL, 0); //等待子进程结束

    }

    sem\_unlink("name\_sem");//删除有名信号量

    return 0;

}

运行结果如下：

