## 共享内存

**定义：**

**共享内存就是允许两个不相关的进程访问同一个逻辑内存**。是在两个正在运行的进程之间共享和传递数据的一种非常有效的方式。

不同进程之间共享的内存通常安排为同一段物理内存。进程可以将同一段共享内存连接到它们自己的地址空间中，所有进程都可以访问共享内存中的地址。*如果某个进程向共享内存写入数据，所做的改动将立即影响到可以访问同一段共享内存的任何其他进程*。

注意：

共享内存并未提供同步机制：在第一个进程结束对共享内存的写操作之前，并无自动机制可以阻止第二个进程开始对它进行读取。

**我们通常需要用其他的机制来同步对共享内存的访问**，例如前面说到的信号量，定义一个标识变量

头文件

#include<sys/shm.h >

**函数接口**

**int** shmget(key\_t key, **size\_t** size, **int** shmflg)

获取或者创建key所对应的共享内存不标识，size表示共享内存的大小，shmflg为权限标志。

0644,它表示允许一个进程创建的共享内存被内存创建者所拥有的进程向共享内存读取和写入数据，同时其他用户创建的进程只能读取共享内存。

IPC\_CREAT在共享内存不存在时，执行创建的操作

**void** \*shmat(**int** shm\_id, **const** **void** \*shm\_addr, **int** shmflg)

shmat函数的作用就是用来启动对该共享内存的访问，并把共享内存连接到当前进程的地址空间（创建完共享内存后，还不能立刻访问）

shm\_id是由shmget函数返回的共享内存标识

shm\_addr指定共享内存连接到当前进程中的地址位置，通常为空，表示让系统来选择共享内存的地址

shm\_flg是一组标志位，通常为0。

**调用成功后返回指向共享内存中第一个字节的指针，否则返回-1**

**int** shmdt(**const** **void** \*shmaddr)

将共享内存从当前进程中分离。注意，将共享内存分离并不是删除它，只是使该共享内存对当前进程不再可用。

**int** shmctl(**int** shm\_id, **int** command, **struct** shmid\_ds \*buf)

shm\_id是shmget函数返回的共享内存标识符

command是要采取的操作

* 1. IPC\_STAT:获取共享内存的参数信息，保存到结构体shmid\_ds中
  2. IPC\_SET:如果权限够，则设置共享内存的参数，依据shmid\_ds的参数
  3. IPC\_RMID:删除共享内存段

shmid\_ds结构至少包括以下成员：

struct shmid\_ds

{

*uid\_t shm\_perm.uid;*

*uid\_t shm\_perm.gid;*

*mode\_t shm\_perm.mode;*

}