进程与线程

1. 进程间的通信方式

- ### 管道

管道是一种把两个进程之间的标准输入和标准输出连接起来的机制，可分为无名管道和有名管道，用于具有亲缘关系进程间的通信。有名管道克服了没有名字的限制，因此允许无亲缘关系进程间的通信。

特点

> ①只能在有亲缘关系的进程之间进行通信，也就是在父子进程之间通信
>
> ②单向通信，一个读端，一个写端。如果要双向通信就要建立两个管道
>
> ③接收数据流，与数据格式无关
>
> ④一般而言，进程退出时管道即被释放，因此管道的生命周期会随进程的关闭而消亡
>
> ⑤同步互斥原则，内核会对管道操作进行同步和互斥

无名管道参考

有名管道**读**操作参考

有名管道**写**操作参考

- ### 信号

信号机制是UNIX系统中最为古老的进程之间通信机制。它用于一个进程或多个进程之间传递异步信号，来通知进程有某种事件发生，除了用于进程间通信外，使用信号还可以使进程发送信号给进程本身。

信号参考

- ### 消息队列

消息队列是内核地址空间中的消息的链表，通过Linux内核在进程之间传递内容。消息顺序地发送到消息队列中，并以几种不同的方式从队列中获取，每个消息队列可以用IPC标识符唯一的进行标识。不同的消息队列之间是相互独立的，每个消息队列中的消息构成一个独立的链表。有写权限的进程可以对消息队列添加消息，有读权限的进程可以对消息队列读取信息

特点

> ①消息队列克服了信号承载信息量少，管道只能承载无格式字节流以及缓冲区大小受限的缺点。消息队列 是基于消息的，而管道是基于字节流的，且消息队列的读取不一定先入先出
>
> ②消息队列中每个消息的最大长度是有上限的，每个消息队列的总的字节数也是有上限的，系统上消息队 列的总数也有一个上限

消息队列**读**参考

消息队列**写**参考

- ## 共享内存

共享内存是在多个进程之间共享一块内存区域的一种进程间的通信方式，它是通过在多个进程之间对内存段进行映射的方式来实现共享内存的，是进程间通信最快捷的方式，之所以最快，是因为共享内存方式的通信没有中间过程，而管道、消息队列等方式则是需要将数据通过中间机制进行转换。共享内存往往与其他通信机制（如信号量）结合使用，来达到进程间的同步及互斥

特点

> ①共享内存就是允许两个不相关的进程访问同一个内存
>
> ②共享内存是两个正在运行的进程之间共享和数据的最有效的方式
>
> ③共享内存不提供任何互斥和同步机制，一般用信号量对临界资源进行保护
>
> ④不同进程之间共享的内存通常安排为同一段物理内存

共享内存**读**参考

共享内存**写**参考

- ## 信号量

信号量是一种计数器，用来控制对多个进程共享的资源所进行的访问，它们常常被用作一个锁机制，在某个进程正在对特定资源进行操作时，信号量可以防止另一个进程去访问它。简单来说就是信号量是一个特殊的变量，程序对其访问都是原子操作，且只允许对它进行P（申请）V（释放）操作

- ## socket

最为一般的进程间通信机制。可用于不同机器之间的不同进程之间的通信

2. 线程同步方法

- ## 互斥锁（mutex）

通过锁机制实现线程间的同步

```
1、初始化互斥锁；
int pthread_mutex_init(pthread_mutex_t *mutex,const pthread_mutex attr_t
*mutexattr);
//mutex：锁容器； mutexattr：锁的属性NULL；

2、申请锁，如果锁不可用，阻塞等待
int pthread_mutex_lock(pthread_mutex_t *mutex);
//mutex：你要申请的锁      返回值:0成功-1失败

3、测试锁    如果锁不可用，立即返回
int pthread_mutex_trylock(pthread_mutex_t *mutex);

4、释放互斥锁 ，打开锁
int pthread_mutex_unlock(pthread_mutex_t *mutex);
```

```
5、销毁锁
int pthread_mutex_destroy(pthread_mutex_t *mutex);
```

互斥锁参考代码

- ## 信号量（sem）

```
1、初始化信号量
int sem_init(sem_t *sem, int pshared, unsigned int value);
//sem：一个存放信号量的容器    pshared：使用范围 0：线程间使用  !0 进程间使用
//value：信号量的初始值
//返回值：0成功   -1失败

2、P操作   申请资源
int sem_wait(sem_t *sem);
//sem：要申请的资源类型，容器  返回值：0成功     -1失败

3、v操作   释放资源
int sem_post(sem_t *sem);
//sem：要释放的资源类型，容器

4、查看信号量的值
int sem_getvalue(sem_t *sem, int *sval);
//sval：存放信号量数值的容器
```

信号量参考例子

3. 进程线程区别

进程由3部分构成 进程控制块， 程序段，数据段，进程是操作系统进程资源分配的基本单元。 **线程**是进程的一个实体，是被系统独立调度和分派的基本单元，线程自己不具有系统资源，只拥有一点在运行中必不可少的资源。线程与同属一个进程的其他线程共享进程多拥有的所有资源。一个线程可以创建和撤销另一个线程，同一进程中多个线程之间可以并发执行。

- 一个程序至少有一个进程，一个进程至少有一个线程
- 进程在执行过程中所拥有独立内存单元，而多个线程共享进程所拥有的内存
- 进程可以独立运行，但线程不能独立执行，必须依存在进程中，由使用该进程的应用程提供多个线程执

4. 死锁[解决方式:顺序加锁 获取时限]

死锁