一、IPC介绍：

PIC 允许一个硬件线程中断其他的硬件线程，这种方式被称为核间中断(Inter-Processor Interrupts,IPI）。PIC 拥有一个宽度为 32 位的核间中断寄存器IPIBase，该寄存器包含目的线程的编号、中断向量及中断类型(是否中断多个硬件线程）等内容。核间中断可以通过向这个寄存器写入需要的值来产生。若硬件线程 A 想要发送一个核间中断给硬件线程 B，它只需要向寄存器 IPIBase 中写入 B的 Thread ID、中断向量、中断类型等值就可以了，PIC 会通知 B 所在的核挂起它当前的执行序列，并根据中断向量跳转到中断服务例程 ISR 的入口。使用 IPI 进行核间通信的关键在于要利用中断服务例程 ISR 去读取一个事先约好的共享内存区域。发起方首先将消息写到一块共享内存中，然后发起核间中断。被中断的硬件线程在中断服务例程中读取该内存，以获得发起方通知的消息。为防止多核间的竞争导致消息被改写，使用这种方式必须利用锁机制来确保消息的完整性。

二、

2.1 初始化函数ICC\_Initialize( )：

注册核间中断处理函数ICC\_Handle\_IPI，并使能。

2.2 核间中断函数 ICC\_Handle\_IPI( )

1、通过CP15协处理器获取当前cpu\_id；

2、清除当前CPU的IPI中断值；

3、获取当前ICC\_Data[cpu\_id]的spinlock；

4、通过ICC\_Data[cpu\_id].ipi\_mask获取当前核间中断的消息掩码，并保存入msg\_mask；

5、清0掩码ipi\_mask；

6、释放spinlock

7、对msg\_mask从高位到低位的置位值依次遍历，目前处理以下几位：

bit0：ICC\_SCHEDULE\_MSG，核间调度消息

仅置位主动切换请求标志

bit1：ICC\_SHUTDOWN\_MSG， 关闭当前核的电源，

需要配置为支持电源管理，暂不支持

bit2：ICC\_PAUSE\_MSG，暂停消息

同电源管理，暂不支持

bit3：ICC\_RPC\_MSG，留空

bit4：ICC\_DATA\_MSG，留空

bit5：ICC\_SYNC\_MSG，尚不确定用途

2.3 ICC\_Send\_With\_Data

未实现

2.4 ICC\_Receive

未实现

2.5 ICC\_Send(ESAL\_CPU\_MASK \* cpu\_mask, UNSIGNED msg\_type)

1、检查参数，msg\_type应小于6，cpu\_mask不应为0，否则返回错误

2、遍历所有核，如果为cpu\_mask标记的核，则：

获取自旋锁ICC\_Data[cpu\_index].lock

按照msg\_type设置ICC\_Data[cpu\_index].ipi\_mask

释放自旋锁

3、调用ESAL\_PR\_SMP\_IPI\_Send(ESAL\_CPU\_MASK \* cpu\_mask)

设置核间中断寄存器值，结构如下：

bit0-bit15：IPI中断向量在以AR中断向量定位符开始的编号，可理解为内部中断号

bit16-bit23：cpu\_mask，即要发送的CPU掩码值

bit24-bit31：IPI目标掩码值，此处为0

将以上值设写入软中断寄存器0xF8F01F00中，触发IPI中断处理。

一：

MCAPI是一个针对在密集分布系统（closely distributed systems）中的处理器之间进行通讯的标准规范，MCAPI满足基本的消息传递和多核系统同步的需求。

Nucleus 中的MCAPI实现是基于由多核联盟（MCA）发布的V1.063的MCAPI标准来实现的。该实现提供了一个移植层用于新OS或平台的代码迁移。

二：

MCAPI拓扑结构

数据收发过程

阻塞和非阻塞的使用

2.1 拓扑结构

MCAPI拓扑由通过端点（endpoints）来交换数据的独立结点(nodes)组成。各个结点在编译时预先决定，而端点可以在编译或运行时配置。

NODE:

每个node都是一个单独的实例，初始化时分配一个系统范围内唯一的ID，运行中可通过该ID对其进行引用。node之间可通过路径进行连接，并交换数据。

node操作原语：

mcapi\_initialize

mcapi\_finalize

mcapi\_get\_node\_id  
 Endpoints: