bkc_pasync 模块设计

Cond 要支持的操作
A1 create/destroy 操作
a2. wait 操作
A3. signal 操作
A4. 打印等待在 cond 上的所有线程（后续要支持的操作，可以先不考虑，因为 cond 可以通过 mutex 来表达）

Mutex 要支持的操作
A1. create/destroy 操作
A2. lock 操作
A3. unlock 操作

Semaphor 要支持的操作
A1. create/destroy 操作
A2. p 操作
A3. v 操作

Basic async rules
>>
async 的三个基本原操作
1. 唤醒指定的执行体（如线程）来执行 (wake-up)
2. 使得指定的执行体（如线程）休眠 ,(hang-on)
3. 栅栏操作（可由硬件指令来实现）: 比如 testandset
对于 testandset: 语义可以如下：

testandset(a, b)
如果条件为 a, 则设置为 b 继续运行下去
如果条件为 b, 则反复测试此条件 .

这种语义很容易实现锁
比如
testandset(1, 0)->lock
如果条件为 1 则设置为 0, 继续运行下去 .
如果条件为 0 则反复测试此条件 .
...

testandset(0, 1)->unlock
如果条件为 0, 则设置为 1 继续运行下去
如果条件为 1, 则反复测试此条件 .
...

Semaphor 的语义实现
>>
semaphor 两级选择控制，p 的时候进行级别判断运行，v 的时候进行级别判断唤醒 .

>>
semaphor 的实现语义

1.
p(semaphor) 操作：

tag1:barrier
把自己放入备选择运行的队列；
按照预定的选择策略选择一个来分配资源；
把自己移出备选择运行队列；

如果发现选择的不是自己：
    把自己放到 semaphor 的等待队列中，登记
  unbarrier
    等待 (hang-on(self))
  ...（直到 hang-on(self) 结束，本执行体又被执行）
  goto tag1
如果发现选择的就是自己：
    判断 res 是否够用
    够用
      占用一个或指定数目的资源，做好记录，
      继续下去 (continue p operation)
  unbarrier
    不够用
      把自己放到 semaphor 的等待队列中，登记
  unbarrier
      等待 (hang-on(self))
  ...（直到 hang-on(self) 结束，本执行体又被执行）
  goto tag1

2.
v(semaphor) 操作：

barrier
归还得到的资源，做好记录
判断是否有执行体在等待 res
    有
      按照策略选择一个，把它加入备选运行队列，
       唤醒它 (wake-up(her)),
    无
      继续下去
unbarrier

异步关系