其实就是增加一个标志位flag: tx\_preemt\_enable，其值=1表示允许发送preemt帧，=0表示不允许发preemt帧。

默认情况下，flag：tx\_preemt\_enable=1允许发送preemt帧。

mac软件进入powersave模式的时候将这个标志位置0不允许发送preemt帧，mac软件退出powersave模式的时候将这个标志位置1允许发送preemt帧。

需要仲裁的情况：mac软件处于非powersave模式，如果wifi、BT同时请求，造成冲突的时候需要仲裁。

分两种情况，

①仲裁给WiFi，则WiFi正常使用。此时WiFi感觉不到BT的存在，就像是WiFi独占一样；

②仲裁给BT，那么mac逻辑走preemt流程，t3\_timer\_out时刻pta逻辑输出tx\_abort高电平有效给mac逻辑，mac逻辑产生abort\_start中断上报到软件，同时mac逻辑启动发preemt帧null/qosnull/cts-self(ps=1)流程，preemt帧发送完成后产生abort\_end中断，软件收到abort\_end中断后设置flag:tx\_preemt\_enable=0。在t4\_timer\_out时刻切天线给BT，t1\_timer\_out时刻天线稳定，BT进行收发。假设蓝牙收发时间长度为Tbt。

又分为两种情况，a.假设蓝牙正常使用完毕，Tbt时间后拉低tx\_abort信号，mac逻辑产生abort\_end中断，软件设置flag:tx\_preemt\_enable=1，同时mac逻辑启动发preemt帧null/qosnull/cf-end(ps=0)流程；b.假设Tbt时间段的中间，WiFi来了请求，再分为两种情况，如果仲裁给BT，那么BT继续使用(此时由于flag:tx\_preemt\_enable=0不允许发preemt帧，因此不会再进行preemt过程)，如果仲裁给WiFi，那么这个时候怎么处理？因为在之前的流程中已经发了preempt(ps=1)帧，此时WiFi不应该再发送数据了。(这个时候原来的逻辑是怎么处理的？问李胜)

--我认为，mac逻辑启动发preemt帧null/qosnull/cts-self(ps=1)之后，BT一直使用，WiFi不可中断BT收发，直到BT使用完毕主动释放使用权，正常拉低tx\_abort信号、产生abort\_end中断。

不需要仲裁的情况：mac软件进入powersave模式，BT可以直接使用，不需要仲裁。

**wl\_tx\_abort**

t3\_timer

t2\_timer

t4\_timer

Tbt

t1\_timer

abort\_end中断

post preemt(ps=0)

tx\_preemt\_en=1

abort\_done中断

tx\_preemt\_en=0

abort\_start中断

发preemt(ps=1)

tx\_preemt\_en=1

bt\_status在t2\_timer out之前表示bt\_priority，在t2\_timer out之后表示bt收发状态bt tx/rx。

因此在t2\_timer out这一时刻将bt\_status锁存为bt优先级，在这以后的时间里都以这个值作为bt优先级。

最早在t3\_timer out这一时刻给出仲裁结果，如果仲裁给bt则拉高wl\_tx\_abort信号，进行preempt的过程。