问题背景

录音播音实际需求

1、随时终止

2、录音并非文件，而是形成rtp发送

3、播音并非源于文件，而是源于rtp

因此Waveform audio使用的buffer较小，不断的装载/发送 buffer,终止的时候Reset并且close.

大致如下调用的循环

录音

waveInUnprepareHeader

waveInPrepareHeader

waveInAddBuffer

播音

waveOutUnprepareHeader

waveOutPrepareHeader

waveOutWrite

循环周期40ms，我采用的是回调函数。问题是有时候调用waveInReset/waveOutReset会形成死锁，调用waveInReset/waveOutReset的线程与回调函数所在的线程死锁在一块了。

原因分析

这方面网上有文章提到，就是调用waveInReset/waveOutReset的同时调用了录音/播音循环调用的某个函数会形成死锁。我再稍作解释下，我们知道buffer满了或是调用Reset都会触发消息（回调函数方式的话就是MM\_WOM\_DONE/MM\_WIM\_DATA），由于调用waveInReset/waveOutReset所在的线程，与回调函数所在的线程不是一个线程，因此很容易撞车，也就是说，你调用reset的时候，另一个线程正好在处理MM\_WOM\_DONE/MM\_WIM\_DATA，于是就这样死锁了。

解决方案

方案一

先加上标记(假设标记bReset:bool)，令bReset为true;

标记作用如下

if(!bReset)

{

录音

waveInUnprepareHeader

waveInPrepareHeader

waveInAddBuffer

播音

waveOutUnprepareHeader

waveOutPrepareHeader

waveOutWrite

}

延时调用waveInReset/waveOutReset，延时时间长度以循环周期为妙，我这个例子中也就是采用40ms。

当然也可以采用临界保护。

方案二

换一个角度去考虑问题，之所以死锁，是因为两个线程冲突了的缘故，所以可以建立一个线程

录音

waveInUnprepareHeader

waveInPrepareHeader

waveInAddBuffer

播音

waveOutUnprepareHeader

waveOutPrepareHeader

waveOutWrite

与waveInReset/waveOutReset都放到这个线程去处理，自然不会发生死锁了。