RKNanoD (Olive) SHUFFLE问题说明

20160528

1. **WALKMAN B-183(NANOC)的shuffle功能定义如下：**

*当选择播放或者更改SHUFFLE模式时，第一首歌曲不算，从下一首歌曲算起*：

例如：当选择第n首歌曲播放时，依次会播放如下序列

n，X,X,X.....

如果当前歌曲是第m首，从非SHUFFLE模式换成SHUFFLE模式时，依次会播放如下序列：

m, X,X,X......

对应API说明如下：

Int CreateRandomList(unsigned int num, const unsigned int orgSeed0, const unsigned int orgSeed1)

其中：num: 当前总歌曲数，orgSeed0：初始种子， orgSeed1: 无意义

unsigned int randomGenerator(DIRECTION\_TYPE direction, const unsigned int num)

direction：方向，num: 当前总歌曲数

由于内存限制我们使用的线性同余等概率伪随机算法，以下简称算法，算法特点：给定一个初始值，后续序列完全相同（RK早期SDK使用的算法），由于不符合183 shuffle的定义，需要更改，客户提出SHUFFLE实现方法，但是由于内存的限制，实现不了，最终商讨一个这种的方案，在第一首歌曲不算的情况，我们利用算法来拟合客户的需求，做法如下：

1. 将算法随机集合扩增到6万首以上
2. 任给一个歌曲种子，产生一个歌曲总个数的子集
3. 配合上层逻辑，可以基本上满足客户的需求。

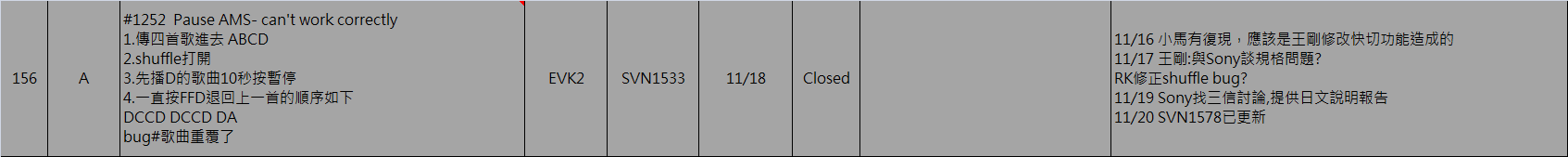
但是只要有耐心在歌曲较少的情况（测试是5首），还是能推算出下一首歌曲。

1. **NanoD （Olive）的处理方式**

NANOD项目需要解决此问题，让用户推算不出下一首歌曲是什么，因此必须引入新的算法，新的算法需要内存需求，大小为16K，为支持的最大歌曲数 \* 2， 由于内存不足，后来采用了一个折中的办法：就是128首歌曲以内，消耗256字节内存，使用新的算法，128歌曲以上使用老算法。因为新算法接口定义上不同，初始种子不在有意义，种子是在产生随机序列的时候随机给出的，为了接口的一致性，这里把128以上使用的老算法的初始种子给固定为12了，也就是说老的算法出来的序列是：

12（初始种子）, 25, 38, 45, 67, 203, 109, 199, 210...

**其实这应为BUG，本来可以在一次修正新算法带来的BUG时，可以将其修正，后来考虑新算法稳定性问题，被叫停，所以这个BUG一直未被解决**。



因此大于128数据出现序列为

1st time

1. 初始种子, 25, 38, 45, 67, 203, 109, 199, 210,...[此序列是以12为初始种子的生成的]

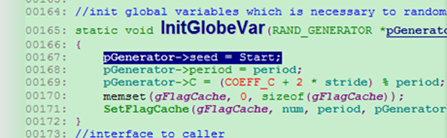
2nd time

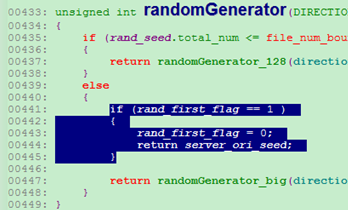
1. 初始种子, 25, 38, 45, 67, 203, 109, 199, 210,... [此序列是以12为初始种子的生成的]

3rd time

1. 初始种子, 25, 38, 45, 67, 203, 109, 199, 210,... [此序列是以12为初始种子的生成的]

问题代码截图：





1. **解决方案如下2种：**
2. 使用WALKMAN B-183(NANOC)算法和规格。
3. 将256字节扩大到16K内存，取消128分水岭。