遥控器 TAG 码约定

V1.1

1. 红外空调 TAG 说明

空调参数很多使用 16 进制的字符串保存对应的字节流，长度用 LEN 表示（LEN 是一个字节，表示后面的多少个字节的数据是一整块数据）。各个状态值有对应的常量定义。

```
public static final byte AC_MODE_COOL = 0;
public static final byte AC_MODE_HEAT = 1;
public static final byte AC_MODE_AUTO = 2;
public static final byte AC_MODE_FAN = 3;
public static final byte AC_MODE_DRY = 4;

public static final byte AC_WIND_SPEED_AUTO = 0;
public static final byte AC_WIND_SPEED_LOW = 1;
public static final byte AC_WIND_SPEED_MEDIUM = 2;
public static final byte AC_WIND_SPEED_HIGH = 3;

public static final byte AC_POWER_ON = 0;
public static final byte AC_POWER_OFF = 1;

public static final byte AC_UD_WIND_DIRECT_SWING = 0;
public static final byte AC_LR_WIND_DIRECT_SWING = 0;
```

状态为 0-N 的整数，这儿使用数据块的偏移量来隐含表示，比如制冷模式的数据块需要放在最开头，后面就是制热模式；自动风速数据块放在第一位，后面是低，中，高，后面的参数数据根据不同的 TAG 值有不同的解析方式。一般的规则就是第一个字节指定空调的数据的第几个字节，接下来的字节 表示要将这个字节设置为的值。举例如下：
1001 = 02 0C 05 04 04 05 0C 05
1001 TAG ID 表示这是第一种开关机参数，第一个 02 表示开机的参数有 2 个字节，0C 05 表示开机的话需要将第 13 个（从 0 开始下标为 12）字节的值设置为 05,04 表示关机的参数有 4 个字节 04 05 表示将第 5 个字节设置为 05， 0C 05 表示将第 13 个字节设置为 05。

格式可以表示为<LEN><BYTE POS><BYTE DATA>...

温度差不多，唯一的区别是 温度数据指定的不是将对应字节设置为多少字节数，而是相对于 16 度需要增加多少。

另外有些不是按 BYTE 指定，而是需要按 BIT 指定。这种需要增加一个 TAG，格式就变成
<LEN><START BIT POS><STOP BIT POS><BYTE DATA>...

注意：这儿的每个<>中的均占 1 个字节，使用 2 个 16 进制字符串表示。

[1000-2000] 保留为空调专用的 TAG,其中[1000-1500] 保留为需要传给 native library 的参数，[1501-2000]为 APP 使用的参数。

1000 空调组合码算法 //用来指定使用哪一个算法

1001 空调开关机参数 1 //用来设置开关机的参数，格式 <LEN><BYTE POS><BYTE DATA>...

1002 空调初始值 1 // 空调所有字节的初始值，格式 <LEN><BYTE DATA>... (每个字节的起始值是多少)

1003 空调温度参数 1 //用来设置温度的参数，格式 <LEN><BYTE POS><BYTE DATA>...

1004 空调模式参数 1 //用来设置模式的参数，格式 <LEN><BYTE POS><BYTE DATA>...

1005 空调风速参数 1 //用来设置风速式的参数，格式 <LEN><BYTE POS><BYTE DATA>...

1006 空调左右风参数 1 //用来设置空调左右风的参数，格式 <LEN><BYTE POS><BYTE DATA>...

1007 空调上下风参数 1 //用来设置空调上下风的参数，格式 <LEN><BYTE POS><BYTE DATA>...

1008 空调校验码参数 //用来设置校验码的参数，格式 <LEN><CHECK SUM ALG><START BYTE POS><END BYTE POS><CHECK SUM BYTE POS><CHECK SUM OFFSET> ...(LEN 字节后面的第一个字节表示使用哪种检验码算法，后面的数据跟校验码相关，一般都是<START BYTE POS><END BYTE POS><CHECK SUM BYTE POS><CHECK SUM OFFSET> 表示起始字节到结束字节之间的所有字节参数校验，结果存放在哪个字节中,最后一个 CHECK SUM OFFSET 是可选字节，如果有的话 表示在 checksum 结果上进行修正，比如如果这个字节是 0x01，算出来的 checksum 为 0x08,那么最终的 checksum 就会设置为 0x09 ）

检验码算法常量定义如下：

const unsigned char CHECKSUM_SUM = 1; //字节累加和

const unsigned char CHECKSUM_SUM_ANTI = 2; //字节累加和取反

const unsigned char CHECKSUM_HALF_BYTE_SUM = 3; //半字节累加和

const unsigned char CHECKSUM_HALF_BYTE_SUM_ANTI = 4; //半字节累加和取反

const unsigned char CHECKSUM_HALF_BYTE_SUM_2= 5; //任意半字节累加和,格式为 05 结果放在第几个半字节 偏移量 累加的半字节位置.....

const unsigned char CHECKSUM_HALF_BYTE_SUM_ANTI_2= 6; //任意半字节累加和,格式为 06 结果放在第几个半字节 偏移量 累加的半字节位置.....

1009 空调独立发码 1 //用来表示哪些按键需要单独发码，格式 <LEN><FUNCTION>.... (每个 functionid 由 1 个字节表示，空调预留了 0-30functionId ）

1010 空调按键参数 1 //用来定制化每个按键的数据，<LEN><FUNCTION><BYTE POS><BYTE DATA>....,这儿考虑可能 function 会比较多，因此没有采用下标的格式，而是将 FUNCTION 作为一个字节显示的表示。

```
public static final int FUNCTION_POWER = 1;
public static final int FUNCTION_MODE = 2;
public static final int FUNCTION_TEMPERATURE_UP = 3;
public static final int FUNCTION_TEMPERATURE_DOWN = 4;
public static final int FUNCTION_WIND_SPEED = 5;
public static final int FUNCTION_UD_WIND_MODE_SWING = 6;
public static final int FUNCTION_UD_WIND_MODE_FIX = 7;
```

1011 空调温度参数 2 //用来设置温度的参数，格式 <LEN><START BIT POS><STOP BIT POS><BIT RANGE DATA>...
1012 空调模式参数 2 //用来设置模式的参数，格式<LEN><START BIT POS><STOP BIT POS><BIT RANGE DATA>...
1013 空调风速参数 2 //用来设置风速式的参数，格式 <LEN><START BIT POS><STOP BIT POS><BIT RANGE DATA>...
1014 空调左右风参数 2 //用来设置空调左右风的参数，格式 <LEN><START BIT POS><STOP BIT POS><BIT RANGE DATA>...
1015 空调上下风参数 2 //用来设置空调上下风的参数，格式 <LEN><START BIT POS><STOP BIT POS><BIT RANGE DATA>...
1016 空调按键参数 2 //用来定制化每个按键的数据，<LEN><FUNCTION><START BIT POS><STOP BIT POS><BIT RANGE DATA>....,这儿考虑可能 function 会比较多，因此没有采用下标的格式，而是将 FUNCTION 作为一个字节显示的表示。

1501 空调制冷模式功能
1502 空调制热模式功能
1503 空调自动模式功能
1504 空调送风模式功能
1505 空调除湿模式功能

这 5 个 TAG 分别描述空调对应的模式不支持的功能：
比如 1503 = T|S&1,2,3 表示在自动模式下没有温度（T），风量没有低中高（1,2,3），也就是只有自动风量。
1503=NA 表示没有自动模式，因此如果只有制冷制热功能的空调，需要配置 1503=NA, 1504=NA, 1505=NA

注意：1501-1505 描述的是没有的功能。因此如果对应的 TAG 不出现，表示该模式下为全功能

1506 空调上下风向功能
~~1507 空调左右风向功能~~

1506 和 1507 描述方式稍有不同，描述的是拥有的方向状态，因此 1506=0,1,2 表示上下风向状态为扫风以及两个固定风向，注意：风向所有的 0 代表扫风（自动风），大于 0 的数字代表固定风向的不同风向状态。如果 1506 不出现，则默认 0,1。如果配置 1 则表示不区分扫风和固定风向。

1508 重发次数 //表示帧重发几次
1509 空调最后字节位数 //表示空调的最后一个字节有效的位数是多少，有些空调最后一个字节比如只有 4bit
1510 空调定制波形 //主要用于有些遥控器的某几个键的码值是和其他键不同格式的固定的码。配置格式<FUNCTION>&<STATUS>&<WAVE>|.... 比如 1&1&123,456|6&&567,897|5&0,1&345,123 表示按了关机键 发送 123,456 按了扫风键 发送 567,897, 按了风量的自动和低风量发送 345,123
1511 空调定制脚本 //主要用于没规律的格式，在经过 SO 处理后，再使用脚本进行最终的修正

红外通用 TAG 说明：

300 引导码 //,分隔的数字字符串表示高低电平的时长，单位 us

301 0 表示数据 //,分隔的数字字符串表示高低电平的时长，单位 us

302 1 表示数据 //,分隔的数字字符串表示高低电平的时长，单位 us

303 延迟码 // 格式为 <END BYTE>&<,分隔的数字字符串表示高低电平的时长>|<END BYTE>&<,分隔的数字字符串表示高低电平的时长>，比如 6&6488,4567|9&7898|-1&3456 表示在第 6 个 BYTE 后加上 6488,4567,在第 9 个字节后加上 7898，在结尾加上 3456

304 帧长度 //固定一帧有多长，跟延迟码理论上只能出现一个。

305 红外格式，使用的是哪种红外格式编码: 38K OR 56K，暂时不使用

306 编码输出字节序：默认或者 0 都表示大端输出，高位先输出，1 表示小端，低位先输出

307 取值 1 表示结尾不增加 1 的高电平(默认结尾会增加一个 1 的高电平 ,比如如果 1 用 560 ,1680 表示，则如果 307 的 TAG 没有配置或者配置值不等于 1，则生成波形码的时候会在数据区域最后增加 560 高电平)

2. 蓝牙主机遥控 TAG 说明

100 受控设备蓝牙（外设）名称，字符串

101 受控设备是否需要连接握手，1/0，字符

102 名称有效长度，字符串

103 名字特征长度，字符串

104 遥控器识别唯一受控设备的方式，0 表示使用唯一广告名进行识别，1 表示使用 MAC 地址进行识别

300 1 字节命令分段数，[1 字节长度 + 2 字节发送 Handle + N 字节命令码]，循环格式

200 Key0，1 字节命令分段数，[1 字节长度 + 2 字节发送 Handle + N 字节命令码]，循环格式

201 Key1，1 字节命令分段数，[1 字节长度 + 2 字节发送 Handle + N 字节命令码]，循环格式

202 Key2，1 字节命令分段数，[1 字节长度 + 2 字节发送 Handle + N 字节命令码]，循环格式

…

214 Key14，1 字节命令分段数，[1 字节长度 + 2 字节发送 Handle + N 字节命令码]，循环格式