BC7215AC

Arduino

万能空调遥控库

V4.x

目录

简介

BC7215AC Arduino 万能空调遥控库（以下简称空调遥控库），是设计专门用于 Arduino 系统的遥控码库，它基于 c 语言版 BC7215A 离线空调遥控码库，并根据 Arduino 系统的特点进行了封装。驱动库须配合 BC7215A 芯片使用，故本驱动库实际包含了 BC7215 芯片驱动库和万能空调遥控两个库，用户仅需进行 1 次安装。目前 V4.x 已经覆盖全部绝大多数空调品牌型号，且还在快速迭代更新中。

有别于其它基于样本数据库的码库，本空调码库使用使用基于编码规则的方式，设置时不是通过品牌型号来选择，而是通过解码红外信号自动识别其编码规则，从而实现不依赖数据库的万能空调控制。市场上空调的品牌繁多，累计不同品牌和型号，有数千种之多，但所使用的遥控协议则仅有百余种，不同品牌产品，甚至不同大品牌产品间使用相同红外遥控协议的情况非常普遍。因此基于规则的万能遥控方案具有以下优点:

- 不依赖网络和数据库，完全离线，可运行于低配置单片机上。
- 未来型号天然支持。空调红外遥控器功能基本固定，新产品大概率会沿用现有的控制协议。
- 越少见品牌被支持概率越大，小品牌通常会选用大厂家的成熟 OEM 方案，而不是自行开发。

本空调遥控库无需联网除 BC7215A 芯片外不依赖任何外部资源，只要程序存储器和 RAM 足够，可运行于任何 Arduino 系统上。随库提供的例子包括在 ESP8266 和 ESP32 处理器上的应用。

本空调遥控库充分利用了 BC7215A 的万能解码功能，实现了基于空调遥控编码规则的通用驱动，而不是简单的波形数据库，这也是其能做到完全离线而体积小巧的原因。驱动库提供 4 个基本的空调控制功能:

- 控制温度
- 控制工作模式
- 控制风力大小
- 控制电源开关

空调遥控协议简介

空调的红外遥控，总体上分为两种，一种是固定码遥控，其特点是遥控器上每个按键发射的红外编码固定，判断特征是这类遥控器上没有液晶显示屏。这类遥控器和普通影音设备的红外遥控器本质一样，仅需简单复制其红外信号即可，使用 BC7215(A)的基本功能即可实现，无需专门的驱动库，用户可参考 BC7215 芯片 Arduino 驱动库中提供的"学习型遥控器"例子。

另外一种是最常见的，可变编码遥控，其特点是遥控信号的长度很长，每一帧的遥控信号中，都包含了空调控制的完整信息，如设定温度，工作模式，风力大小等，判断特征也很简单，即这类的空调遥控器上，都会带有一个液晶显示屏。这类的空调遥控器，同一个按键，每次按下时发出的红外编码根据当前的机器工作状态不同是不一样的，而且不同的空调厂家和型号的空调的编码方式，也是不一样的，产生这样的空调遥控器信号，就需要借助本空调遥控库来实现。

BC7215AC Arduino 空调遥控库使用方法

注意：因空调遥控库依赖于BC7215A 芯片，建议阅读下面内容前，用户先对BC7215A 的工作原理及输入输出数据格式，以及BC7215 芯片驱动库的函数有所了解，可以利于对以下内容的理解。相关资料可以在本遥控库的文档目录中找到。

本驱动库的使用非常简单，仅需 3 个步骤，即可完成任意空调的控制。

第一步：采样原空调遥控器信号

利用 BC7215A 的解码功能，接收解码被控空调遥控器的特定信号（制冷模式，25°C/77°F），采样的数据用以供驱动库分析获知原空调遥控信号的格式。

注意：

> **不要使用“万能遥控器”作为信号源，因为该类遥控器通常每次按键会连续发出多种不同协议的遥控信号，以达到广泛适用的效果，如果实际起效的遥控信号不是第一个发出，则 BC7215A 会采集到错误的信号。**

第二步：初始化遥控库

使用解码出的数据作为参数，调用本驱动库的初始化函数，使驱动库识别所控制空调的编码格式 。

第三步：设置需要的温度、模式及风力大小

使用驱动库的设置函数，即可发射红外信号将空调设置为指定的状态。

其它：电源开/关

驱动库提供了空调开关函数，调用开关函数，即可控制空调开关机。

最佳实践

实际应用中，初始化成功后，可以将初始化数据保存起来，每次启动后直接用所保存数据完成初始化，这样不必每次执行采样操作，做到开机即用，方法请参阅例程。

系统要求

原理上可用于任何有足够程序和 RAM 空间的 Arduino 系统，示例程序使用了 ESP8266 和 ESP32 作为演示。

空调遥控库详解

组成

BC7215AC 空调遥控库由两个独立的驱动库组成，分为为 BC7215(A)芯片的驱动库，完成基本的对 BC7215(A)芯片的控制操作，BC7215(A)芯片驱动库提供了 4 个演示程序，分别是：

- 任意遥控器解码
- 学习型遥控器
- 2 路红外遥控开关
- 红外数据传输

用于空调控制时，须使用 BC7215A 芯片，不带 A 的 BC7215 芯片不适于用于解码空调信号。
BC7215AC 空调遥控库专用于空调控制，提供了 8 个演示程序，分别为：

- ESP8266 演示阻塞版(使用软件串口与 BC7215A 通讯，并利用 Arduino IDE 的串口监视器作为用户交互渠道，软件为阻塞式设计，逻辑简单)
- ESP8266 演示非阻塞版(使用软件串口与 BC7215A 通讯，并利用 Arduino IDE 的串口监视器作为用户交互渠道，软件为非阻塞式设计)
- ESP32 演示串口监视器英文版(ESP8266 演示的 ESP32 移植)
- ESP32 演示串口监视器中文版(ESP8266 演示的 ESP32 移植)
- ESP32 演示 LCD 英文版(使用 LILYGO T-Display 的板载按键和 LCD 屏)
- ESP32 演示 LCD 中文版(使用 LILYGO T-Display 的板载按键和 LCD 屏)
- ESP32 演示 MQTT 英文版(在 LCD 版基础上增加了联网功能，可利用 MQTT 协议进行空调的联网控制，以及将空调状态进行网络上报)
- ESP32 演示 MQTT 中文版(在 LCD 版基础上增加了联网功能，可利用 MQTT 协议进行空调的联网控制，以及将空调状态进行网络上报)

硬件连接

实现空调控制的核心器件 BC7215A 芯片，采用串口和 Arduino 控制连接，同时还需要 2 个辅助信号 MOD 和 BUSY，MOD 为 Arduino 的输出信号，控制红外信号是发射还是接收模式，BUSY 是 BC7215A 芯片的输出信号，用来控制串口的数据传送。

接口函数

下面讲解 BC7215AC 空调遥控库的接口函数，关于驱动库的接口函数使用，请参阅 BC7215 Arduino 驱动库的说明书。

1. 开始/停止采样函数

```
startCapture();

stopCapture();
```

空调遥控库的初始化前，需要采集特定的空调遥控器信号，这两个函数用来控制 BC7215A 芯片进入和退出接收状态。

2. 采样状态查询函数

```
bool signalCaptured();

bool signalCaptured(bc7215DataVarPkt_t* targetData,  bc7215FormatPkt_t* targetFormat);
```

用来查询采样是否已经完成（接收到了完整空调遥控信号）。此函数有两种形态，第一种不带任何参数的，采样的结果将存入遥控库内部默认的缓冲存储，第二种函数则将采样结果存入用户指定的存储位置。

此函数返回为 bool 值，采样完成时，返回 true，否则返回 false.

3. 初始化函数

```
bool init();

bool init(const bc7215DataMaxPkt_t& data, const bc7215FormatPkt_t& format);

bool init(const bc7215DataVarPkt_t* data, const bc7215FormatPkt_t* format);

bool init(uint8_t cnt, bc7215DataMaxPkt_t const data[], bc7215FormatPkt_t const
format[]);
```

该函数用于利用采样的数据初始化遥控库。该函数有 4 种形式，最简单的无任何参数的形式，也是最常用的方式，直接使用刚刚采样完存在内部缓冲区的数据完成初始化。第 2,3 种形式显式地输入遥控信号的原始数据和格式信息，通常用于读取事前保存的初始化信息后初始化遥控库, 直接传递数据和格式信息的变量或者其指针作为参数。最后一种形式比较特殊也很少用到，仅在遇到某些特殊的空调遥控协议格式时，才会用到，详情见后文 extraSample()函数。

返回参数为一个 bool 值，为 true 时，表示初始化成功，false 时表示初始化失败。

初始化的输入采样数据，必须是空调在“制冷模式，25°C(77°F)”时的数据，如果输入的数据不是这个指定状态下的数据，将造成初始化失败。可以先将空调遥控器设置到制冷 25°C，然后按“风力调节”按键采集信号，以确保工作模式和温度设置保持不变。

4.空调设置函数

```
const bc7215DataVarPkt_t* setTo(int tempC, int mode = -1, int fan = -1, int key = 0);
```

初始化成功后，使用此函数设置空调的状态。

函数具有 4 个输入参数，前三个分别为温度、工作模式，以及风力，第 4 个参数为按键，部分空调的红外编码中，带有按键的信息，比如同样 25°C，从 26°C 按“温度-”按键和从 24°C 按“温度+”按键，虽然最终空调状态一样，但遥控器发射的编码是不同的。这种情况出现在个别品种的空调上，如果不确定是否需要，一般可设置为“保持不变”或者“温度+”。

除温度外，其他参数有默认值，即可以按顺序省略，如 setTo(26), setTo(26, 1), setTo(26, 1, 2)这样的方式，分别代表设置到 26 度其他不变；设置到 26 度制冷模式，其它不变，设置到 26 度制冷模式风力中。

所有的输入参数，均为**8 位有符号整数**，每个参数均有各自的取值范围，超出取值范围的输入值，均视为该项设置保持不变。

呼叫此函数后，将直接控制 BC7215A 芯片发送相应的红外信号，同时函数的返回值为指向所发送的红外数据的指针。

温度temp:

取值范围 16-30, 分别对应温度值 16°C - 30°C，如果使用华氏温度，需用户自行换算为摄氏温度。超出这个范围的参数值，视为现有设置“保持不变”

模式mode:

取值范围 0-4, 分别对应为：

0 - 自动(Auto)

1 - 制冷(Cool)

2 - 制热(Heat)

3 - 除湿(Dry)

4 - 通风(Fan)

超出这个范围的参数值，视为现有设置“保持不变”。

风力fan:

取值范围 0-3, 分别对应为：

0 - 自动(Auto)

1 - 低(Low)

2 - 中(Med)

3 - 高(High)

超出这个范围的参数值，视为现有设置“保持不变”。

按键key:

取值范围 0-3，分别对应为：

0 - 温度+ 键

1 - 温度- 键

2 - 模式按键

3 - 风力按键

超出这个范围的参数值，视为现有设置“保持不变”。

注意：

1. **BC7215AC 空调遥控库是基于编码规则的码库，仅负责产生复合编码规则的数据包，但不会检查其设置是否有效。在各种设置的组合中，有很多是实际空调无法达到的，各种机型会有各自的限制，比如，多数机型可能在“通风”模式下设置温度无效，有的机型可能最低温度只能设置到 18°C 等等，使用本码库时应该以所控制机型的实际限制为准，如果设置超出该机型的允许范围，空调机的反应将不可预知。**

5. 空调开关函数

```
const bc7215DataVarPkt_t* on(void);

const bc7215DataVarPkt_t* off(void);
```

这两个函数用来控制空调的开/关。呼叫后，将直接发射相应红外信号，同时，返回的指针指向实际所发射的数据。

多数空调无需专门的开机指令，在关机状态下，向空调发送设置状态的指令，空调就会开机，对于这样的机型，调用 on()函数，将空调设置到最后一次调用 setTo()函数时的状态，或者初始化采样时的状态。有个别型号开机需要专门的指令，这时，on()函数呼叫后，仅意味着空调会开机，开机后的状态并不确定，需要再使用 setTo()将空调设置为用户需要的状态。

6. 寻找下一匹配函数

```
bool matchNext();
```

初始化时，码库会根据采集的数据，自动匹配空调所用的编码协议，不过，因为市面上空调品牌型号众多，会有不同型号的空调编码协议很接近，但又有细微差异的情况，会有初始化后，空调机的实际动作和设置不相符的情况，如设置风速为高，实际却为低风速等。如果发生这种情况，可以尝试调用此函数，查找库是否有其他和被控空调相符的协议。此函数无输入参数，返回也是一个 bool 值，意义和初始化函数中一致。

如果此函数返回为 true，则表示空调遥控库已经按照新协议初始化成功，用户可以尝试控制空调看是否能够正常操作，如果仍不正常，还可以继续再次调用，直至返回为 false，表示已经没有更多能够匹配的编码协议。

此函数仅可以在初始化正确后调用，调用后，如果想恢复使用初始化后的编码协议，必须重新调用初始化函数。

7. 预定义协议

有极少数的几种空调遥控协议，无法由 BC7215A 芯片直接解码，用户须获取红外波形数据后通过使用在线的转换工具完成转换。BC7215A 空调遥控库随库提供已经经过转换的预置数据，供用户直接调取。当发现 BC7215A 无法完成数据采集，或者采集到的信号无法正常初始化时，可以尝试使用预定义数据看是否可以工作。

有关预定义数据相关的函数主要有：

获取预定义协议数量

```
uint8_t cntPredef();
```

返回值为库中内置的预定义协议数据的数量。

获取预定义协议的名称

```
const char* getPredefName(uint8_t index);
```

输入参数为预定义协议的序号，而返回为一个字符串，为该预置协议的助记名称，帮助用户区分。

使用预定义协议初始化

```
bool initPredef(uint8_t index);
```

使用空调遥控库的内置协议数据初始化。输入参数为预定义协议的序号。

8. 获取版本信息函数

```
const char* getLibVer();
```

该函数无需输入参数，返回值为一个字符串，包含当前空调遥控库的版本信息。

9. 检查协议是否需要额外采样

初始化成功后，可以调用

```
uint8_t extraSample();
```

函数，该函数返回一个 8 位整数，指示当前初始化的空调协议是否有指令使用和普通指令不同的特殊格式，如果有，则需要针对该指令再做一次额外的采样。返回结果可能是 0, 1, 2, 3，4 中的一个值:

0. 无需特殊格式
1. 温度设置指令需特殊格式
2. 模式设置指令需特殊格式
3. 风力设置指令需特殊格式
4. 该协议使用多段红外信号，需用使用多个采样数据进行初始化

10. 保存额外格式信息

如果上面函数得到非 0 的结果，则用户需要再对需要特殊格式的指令再多做一次采样，以便驱动库了解特殊格式的信息。针对该特殊指令进行采样后，使用

```
bool saveExtra(const bc7215DataVarPkt_t* data, const bc7215FormatPkt_t* format);
```

来将其存入空调遥控库。函数中的输入数据，在采样时通过

```
bool signalCaptured(bc7215DataVarPkt_t* targetData,  bc7215FormatPkt_t* targetFormat);
```

函数获得。

保存成功，则返回结果为 true, 否则返回 false;

如果 extraSample()函数返回结果为 4, 则说明该空调遥控协议使用了多段红外数据格式，需要重新使用多段采样数据进行初始化，详情请见后文第 14 小节。

11. 获取基础数据包及获取基础格式包

```
const bc7215DataVarPkt_t* getDataPkt();

const bc7215FormatPkt_t*  getFormatPkt();
```

当初始化成功后，希望将初始化数据保存起来，以便下次开机可直接用于初始化时，可以通过这两个命令来获取当前所使用的初始化数据。

12. 获取额外格式信息

作用和上面获取基础信息类似，当需要保存初始化信息以便下次使用，而当前的协议格式又包含额外格式信息时，可以用此函数获取相应的额外格式信息。具体的返回数据格式为

```
typedef struct {
    uint16_t    bitLen;     /**< Bit length placeholder, always set to 0 */
    union {
        uint8_t data[1];    /**< Raw data array */
        struct {
            const bc7215FormatPkt_t*    fmt;        /**< Pointer to format packet */
            const bc7215DataVarPkt_t*   datPkt;     /**< Pointer to data packet */
        } msg;       /**< Message structure containing format and data packets */
    } body;          /**< Union containing either raw data or structured message */
} bc7215CombinedMsg_t;
```

相关使用方法，请参见例子程序。

13. 查询 BC7215A 芯片状态

```
bool isBusy();
```

如果 BC7215A 芯片处于接收或者发送红外信号过程中，则返回为 true. 当发射红外信号时，红外信号发送的速度通常比较慢，setTo()和 on(), off()函数返回时，红外信号仍处在发射的过程中，如果需要发送下一个红外信号，必须等待当前的信号完全发送结束，且间隔一段时间(通常需>100ms)后才可以发送下一个信号，否则会造成接收端(空调机)不能正确接收和识别指令。

14. 多段数据初始化

当 extraSample()函数返回值为 4,说明该遥控格式和通常的遥控格式不同，为多段格式，必须重新进行采样，采集多段信号。采样时，必须使用

```
bool signalCaptured(bc7215DataVarPkt_t* targetData, bc7215FormatPkt_t* targetFormat);
```

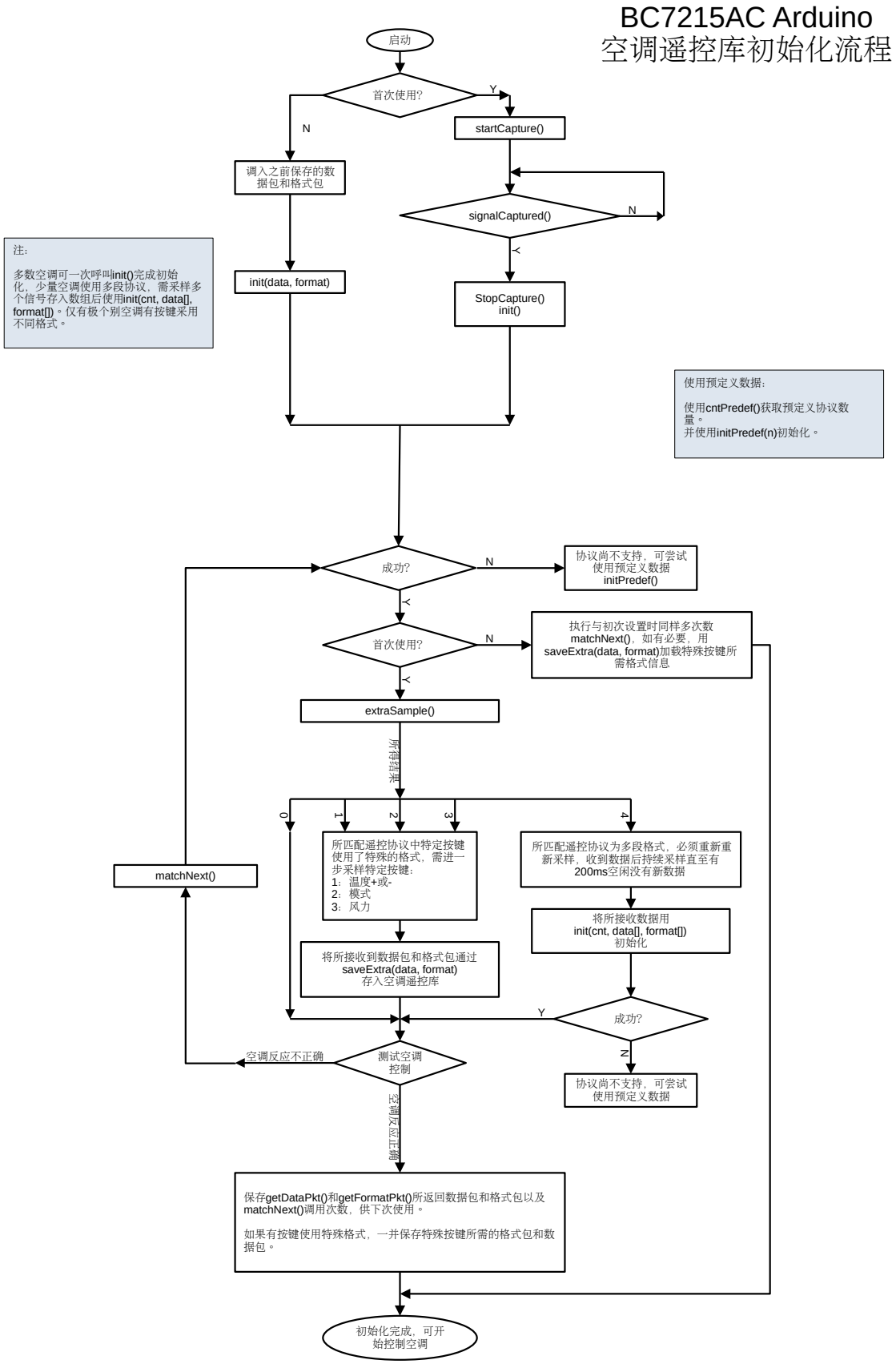
进行查询，将采样的数据和格式信息各存入一个数组，然后用

```
bool init(uint8_t cnt, bc7215DataMaxPkt_t const data[], bc7215FormatPkt_t const
format[]);
```

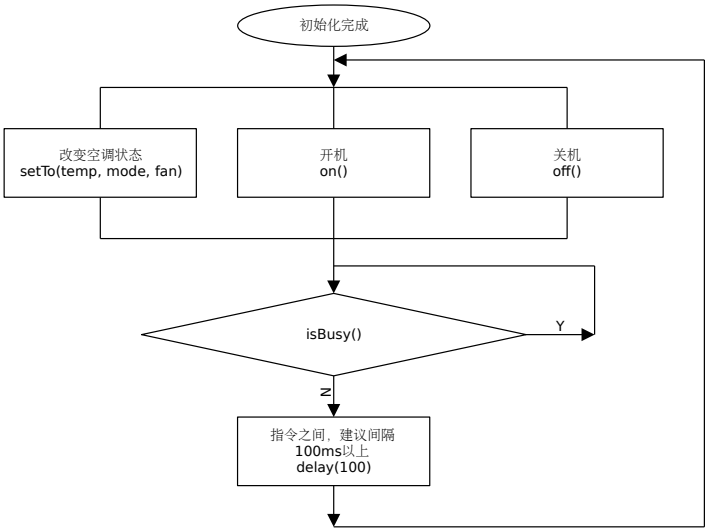
进行初始化。

每采样到一个信号后，程序须开始一个计时，查询 isBusy()的状态，如果在计时 200ms 以内，又收到新的信号，则需保存该信号，然后重新开始计时，知道 isBusy()有连续>200ms 的时间处于 false 的状态，才可停止采样，然后用所有采样到的数据，初始化空调遥控库，cnt 参数为所收到的信号的个数。

编程流程


BC7215AC Arduino
空调遥控库初始化流程
启动
首次使用?
Y
N
startCapture()
调入之前保存的数据包和格式包
signalCaptured()
N
Y
注:
多数空调可一次呼叫init()完成初始化，少量空调使用多段协议，需采样多个信号存入数组后使用init(cnt, data[], format[])。仅有极个别空调有按键采用不同格式。
init(data, format)
StopCapture()
init()
使用预定义数据:
使用cntPredef()获取预定义协议数量。
并使用initPredef(n)初始化。
成功?
N
协议尚不支持，可尝试使用预定义数据
initPredef()
Y
首次使用?
N
执行与初次设置时同样多次数matchNext()，如有必要，用saveExtra(data, format)加载特殊按键所需格式信息
Y
extraSample()
所得结果
0
1
2
3
4
matchNext()
所匹配遥控协议中特定按键使用了特殊的格式，需进一步采样特定按键:
1：温度+或-
2：模式
3：风力
所匹配遥控协议为多段格式，必须重新重新采样，收到数据后持续采样直至有200ms空闲没有新数据
将所接收数据用init(cnt, data[], format[])初始化
将所接收到数据包和格式包通过saveExtra(data, format)存入空调遥控库
Y
成功?
N
空调反应不正确
测试空调控制
协议尚不支持，可尝试使用预定义数据
空调反应正确
保存getDataPkt()和getFormatPkt()所返回数据包和格式包以及matchNext()调用次数，供下次使用。
如果有按键使用特殊格式，一并保存特殊按键所需的格式包和数据包。
初始化完成，可开始控制空调

空调基本控制流程


初始化完成
改变空调状态
setTo(temp, mode, fan)
开机
on()
关机
off()
isBusy()
Y
N
指令之间，建议间隔
100ms以上
delay(100)