**通讯数据格式：（V1.0）**

蓝牙名称：HZDY-ILIGHT-A01

蓝牙初始密码：1234

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 纵向冗余校验 | 结束 |
| 1字符 | 1字符 | 1字符 | n字符 | 2字符 | 2字符 |

起始码：3A(:)

结束码：0D 0A (回车换行)

使用ASCII码发送，灯上的单片机处理完数据后，返回应答信号给手机。

1. **查询灯的状态**

命令：（手机发送数据给灯）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 纵向冗余校验 | 结束 |
| 3A(:) | 00 | 01 | 00 | 2字符 | (回车换行) |

应答：（灯返回数据）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 数据 | 数据 | 数据 |
| 未命名.bmp3A(:) | 01 | 07 | 灯的状态：  （1是开，0是关） | 灯的亮度：  （0到255） | 当前灯的场景 | 当前灯的档位（0到4） |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据 | 数据 | 数据 | 纵向冗余校验 | 结束 |
| 当前音量的大小：（0到0F） | 当前音乐的场景 | 当前音乐的档位（0到4） | 2字符 | (回车换行) |

1. **开关灯**

命令：（手机发送数据给灯）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 纵向冗余校验 | 结束 |
| 3A(:) | 01 | 01 | 开关标志位  1开，0关 | 2字符 | (回车换行) |

应答：（灯返回数据）

OK

1. **发送灯的信息：**

命令：（手机发送数据给灯）

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 数据 | 数据 | 数据 | 数据 | 数据 | 数据 |
| 未命名.bmp3A(:) | 03 | 0E | R0 | G0 | B0 | R1 | G1 | B1 | R2 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数据 | 数据 | 数据 | 数据 | 数据 | 数据 | 数据 | 数据 | 数据 | 纵向冗余校验 | 结束 |
| G2 | B2 | R3 | G3 | B3 | 变化速度  （1、5、0A、0F、14） | 跳变/渐变  （0渐变， 1跳变，2不变化） | 当前场景 | 呼吸灯  （1开，0关） | 2字符 | 回车换行 |

应答：（灯返回数据）

OK

1. **设置灯的亮度：**

命令：（手机发送数据给灯）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 纵向冗余校验 | 结束 |
| 3A(:) | 04 | 01 | 灯的亮度信息（0到255） | 2字符 | 回车换行 |

应答：（灯返回数据）

OK

1. **选择音乐**

命令：（手机发送数据给灯）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 数据 | 纵向冗余校验 | 结束 |
| 3A(:) | 05 | 02 | 音乐场景 | 音乐档位  （0到4） | 2字符 | 回车换行 |

应答：（灯返回数据）

OK

1. **选择音量大小**

命令：（手机发送数据给灯）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 开始 | 功能 | 数据长度 | 数据 | 纵向冗余校验 | 结束 |
| 3A(:) | 05 | 01 | 音量大小  （0到0F） | 2字符 | 回车换行 |

应答：（灯返回数据）

OK

**纵向冗余(LRC)校验**  
 LRC域是一个包含一个8位二进制值的字节。LRC值由传输设备来计算并放到消息帧中，接收设备在接收消息的过程中计算LRC，并将它和接收到消息中LRC域中的值比较，如果两值不等，说明有错误。   
 LRC校验比较简单，它在ASCII协议中使用，检测了消息域中除开始的冒号（16进制是3A）及结束的换行号（16进制是0A）外的内容。它仅仅是把每一个需要传输的数据按字节叠加后取反加1即可。下面是它对应的代码：   
  
BYTE GetCheckCode(const char \* pSendBuf, int nEnd)//获得校验码   
{   
BYTE byLrc = 0;   
char pBuf[4];   
int nData = 0;   
for(i=1; i<end; i+=2) //i初始为1，避开“开始标记”冒号   
{   
//每两个需要发送的ASCII码转化为一个十六进制数   
pBuf [0] = pSendBuf [i];   
pBuf [1] = pSendBuf [i+1];   
pBuf [2] = '\0';   
sscanf(pBuf,"%x",& nData);   
byLrc += nData;   
}   
  
byLrc = ~ byLrc;   
byLrc ++;   
return byLrc;   
}