中航异步全彩_固定节目内容替换协议 V3.0.8

V3.0.0	加节目切换等单命令
V3.0.1	1. 补充 CRC 2. 加汉字发送实例及 Java 转码 code
V3.0.2	1E 协议支持替换字体
V3.0.3	新增协议 0x0044,0x0045,0x0046
V3.0.4	修改发送图片视频
V3.0.5	修改屏幕亮度协议
V3.0.6	增加协议内容样例
V3.0.7	支持 1E 协议换行文本左对齐
V3.0.8	加字符转 HEX 工具
V3.0.9	修改总包长的描述，包括 crc 校验位的长度
V3.0.10	修改简单命令协议长度

1. 文档使用说明

此文档使用对象为二次开发编程人员，具备 C 语言或者更高级语言编程的能力。文档中的一些专业术语也同为面向专业人员，不对公众使用。

2. 系统工作原理

以计算机通过以太网和中航异步全彩控制系统通信来说明通信原理和方法，请参见以下结构示意图。

图 1 全彩异步系统

2.1 预置字符分区到全彩异步系统

计算机通过本地网络，使用 PC 软件 LEDPlayer 给 全彩异步播放盒 发送分区工程文件，包括字符分区或添加其它分区。注意发送完成后，关闭 LEDPlayer，这样保证 LEDPlayer 和全彩异步播放盒的 TCP 连接断开，不会占用通信连接和端口；

2.2 设置固定 IP 到 全彩异步播放盒

计算机通过本地网络，使用 PC 软件 LEDPlayer 给 全彩异步播放盒 设置固定 IP，方便后面发送字符卡协议。注意设置完成后，关闭 LEDPlayer，这样保证 LEDPlayer 和播放盒的 TCP 连接断开，不会占用通信连接和端口；

2.3 TCP 通讯更新字符到全彩异步播放盒

运行客户自己开发的应用程序，通过 TCP 连接 全彩异步播放盒 的 20003 端口，建立连接后会话会保持 30 秒，需要在 30 秒内更新分区内容，超过 30 秒需要重现建立 TCP 链接(如需维持 TCP 长连接，需要先登陆成功后，发送心跳包，参照下文登录 0x0010,心跳 0x0001 通讯协议)，TCP 连接建立后，按照协议发

送数据来更新指定分区中内容。

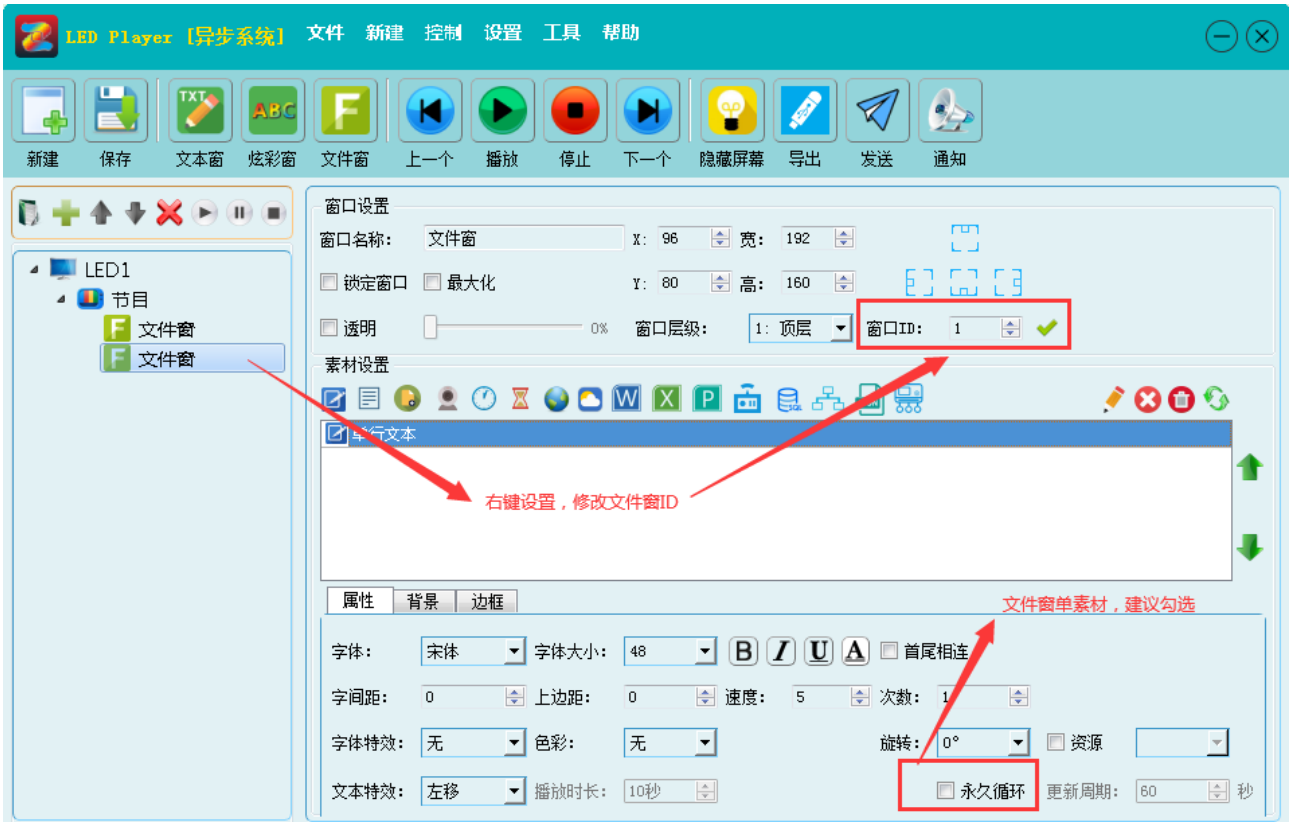
2.4 更新大图片视频到全彩异步系统

客户如果有更新文件窗图片或视频的需求，需要增加 http_server 模块，通过 TCP 连接全彩异步系统的 20003 端口，建立连接后，需要按协议 0x1E 发送图片或视频的 url 路径，达到更新图片的目的。

如果需要一个窗口放三种素材，文本、图片、视频，替换后可以用素材切换命令切换到要显示的素材。

字符分区的制作

PC 上位机软件编辑节目，设置窗口 ID(1，2，3...)，以 ID 区别指定窗口，每个窗口中同一种素材只能放一个(本例中只放一个单行文本素材)，如果窗口内只有一个素材，请勾选素材属性中的“永久循环”。



下载地址：http://www.zhonghangled.com/e/action/ListInfo/?classid=93　　同异步全彩

3. 通信协议说明

1、全部为数字的(如 123)：为十进制数，大端字节序(发送时先发高字节，后发低字节)。

2、以 0x 开头的(如 0xABC)：为十六进制数，大端字节序(发送时先发高字节，后发低字节)。

3、以单引号括起来的(如'A')：为 ASCII 码字符。

4、以双引号括起来的(如"ABCDE")：为 ASCII 码字符串。

4. 命令帧和应答帧结构

字段	字节数	描述
包头(STX)	2	0xAA99，大端网络字节序
命令(CMD)	2	大端网络字节序，eg: 0x001E
版本(VER)	4	0x0000，无 CRC 校验 0x8000，有 CRC16 校验
包长(LEN)	4	总包长，从包头到校验位的总长度,大端网络字节序 2+2+4+4+n+2
数据(DATA)	n	帧数据，长度为总包长减去前 12 字节和 crc 的 2 个字节
校验(CRC16)	2	VER: 0x0000，无 CRC 校验 VER: 0x8000，有 CRC16 校验 计算范围包括 STX，CMD，VER，LEN 和 DATA

表 1 计算机至全彩异步系统的命令帧结构

字段	字节数	描述
包头(STX)	2	0xAA99，大端网络字节序
命令(CMD)	2	大端网络字节序，回包的命令字为 0x8000+CMD，eg: 0x801E
版本(VER)	4	0x0000，回包无 CRC 校验
包长(LEN)	4	总包长，从包头到数据部分的长度
数据(DATA)	n	帧数据，长度为总包长减去前 12 字节

表 2 全彩异步系统至计算机的应答帧结构

5. 命令帧类型定义

	命令代码	定义
无需登录	0x001C	获取节目列表
	0x001E	更新单个文件窗的素材
	0x003C	同时更新多个文件窗素材
	0x003D	TCP 数据流更新图片
	0x003E	修改文件窗属性(坐标，宽高，背景，边框)
	0x003F	增加删除修改节目
	0x0040	增加删除修改文件窗
	0x0041	增加删除修改属性
	0x0042	(TCP)上传文件到播放盒
	0x0043	(TCP)从播放盒下载文件
需要登录		
	0x0001	心跳包，维持 TCP 长连接的心跳协议
	0x0010	登陆验证，建立 TCP 长连接前的身份验证
	0x0007	亮度，开关屏
	0x002C	切换上一个节目/素材
	0x002D	切换下一个节目/素材
	0x001D	处理指定节目
	0x003B	重启设备

表 3 命令帧类型定义

6. 命令 0x0001 心跳包

AA 99 00 01 00 00 00 00 00 00 0C
请求 Data:
无
应答 Data:

名称	长度(字节)	说明
设备 ID 长度	1	

设备 ID	N	

表 4 心跳包帧类型定义

注：心跳包主要用于登录验证成功后，维持 TCP 长连接通讯。

7. 命令 0x0010 登录验证（需要维持长连接的做登录验证）

AA 99 00 10 00 00 00 00 00 00 0E 00 00
请求 Data:

名称	长度(字节)	说明
user_len	1	未设置填 0
user	n1	未设置填 null
pwd_len	1	未设置填 0
pwd	n2	未设置填 null

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；

表 5 登录验证帧类型定义

注：登录验证主要为了身份确认，保证 TCP 长连接时，通讯主机与全彩异步系统只允许有一个 TCP 会话，不允许其他主机建立连接。

8. 命令 0x001E 更新单个文件窗的素材(无需登录)

限制：只能发一个节目，每个窗口中素材类型不能重复(即一个窗口如果加文本素材，则最多只能加一个)
请求 Data:

名称	长度(字节)	说明
item_type	1	单行文本: 0x0C 图片:0x0B 视频: 0x15
item_action	4	覆盖文本：0x00 (**清空**:发送""; **换行**:设置动画静止 403，换行位置加[###]；**换行左对齐**：动画类型静止居左 500，换行位置加[###]) 追加文本: 0x01 删除文本: 0x02 自动换行(静态文本): 0x03
window_id	2	窗口 id
content_length	2	文本：内容字节数组长度 图片: URL 字节数组长度 (Http) 视频: URL 字节数组长度 (Http)
content	n1	文本：内容 图片: URL 内容 (http) 视频: URL 内容 (http)
attributes_length	2	文本：属性字符串转字节后的长度 图片视频: md5 值转字符后的长度(包含后缀)
Attributes	n2	**文本属性键值对(**形如 Key=Value;以分号分割) animation_type=402; (见附录 A.2) keep_time=2000; (时间：毫秒) word_color=#ff0000; (字体颜色) word_size=24; (字体大小) word_bold=0; (1：粗体)

		word_italic=0; (1: 斜体) word_underline=0; (1: 下划线) word_font="xxx.ttc"(字体) **图片视频 md5 值(包含后缀)** eg: f405697cf6377e51.mp4

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证; -2 设备已占用; 0 成功处理返回;

表 6 设置更新单窗口素材

1E 命令发送文本替换：窗口 ID:01

"欢迎光临"

AA99 001E 00000000 00000023

0C

00000000

0001

000c

E6ACA2 E8BF8E E58589 E4B8B4

0000

1E 命令发送图片替换：窗口 ID:01

http://download.zoehoo.com/demo/image.jpeg， name: 98d6f733df463b49.jpg

AA 99 00 1E 00 00 00 00 00 00 00 55

0B

00000000

0001

002A

687474703a2f2f646f776e6c6f61642e7a6f65686f6f2e636f6d2f64656d6f2f696d6167652e6a706567

0014

393864366663733336466343633623 4392e6a7067

1E 命令发送视频替换：窗口 ID:01

http://download.zoehoo.com/demo/video.mp4 name: 98d6f733df463b49.mp4

AA 99 00 1E 00 00 00 00 00 00 00 54

15

00000000

0001

0029

687474703a2f2f646f776e6c6f61642e7a6f65686f6f2e636f6d2f64656d6f2f766964656f2e6d7034

0014

393864366637333364663436336234392e6d7034

9. 命令 0x003C 更新多个文件窗素材(无需登录)

请求 Data:

名称	长度(字节)	说明
count	2	素材数量
item_pkg_len1	2	素材包长度 (1+4+2+2+n1+2+n2) (Item_type ->Attributes)
item_type	1	文本: 0x0C 图片:0x0B 视频: 0x15
item_action	4	0 (0: 覆盖；1: 追加；2 删除；3: 单行文本自动换行)
windown_id	2	窗口 id
content_length	2	文本: 内容字节数组长度 图片: URL 字节数组长度 (Http) 视频: URL 字节数组长度 (Http)
content	n1	文本: 内容 图片:URL 内容 (http) 视频:URL 内容 (http)
attributes_length	2	文本:属性字符串转字节后的长度 图片视频:md5 值转字符后的长度(包含后缀)
attributes	n2	**文本属性键值对(**形如 Key=Value;以分号分割) animation_type=402; (见附录 A.2) keep_time=2000; (时间: 毫秒) word_color=#ff0000; (字体颜色) word_size=24; (字体大小) word_bold=0; (1: 粗体) word_italic=0; (1: 斜体) word_underline=0; (1: 下划线) word_font="md5.ttf" **图片视频 md5 值(包含后缀)** eg: f405697cf6377e51.mp4
item_pkg_len2	2	素材包长度(1+4+2+2+n1+2+n2) (Item_type ->Attributes)
…		

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证; -2 设备已占用; 0 成功处理返回;

表 7 更新多个文件窗素材

注: 通过 0x3C 字符协议，同时更新素材到多个文件窗，协议里的文件窗 ID 和素材类型(文本:0x0C，图片:0x0B，视频:0x15)务必填写正确，pkg_length 和对应的 item_pkg_len 务必准确。

10. 命令 0x003D TCP 数据流更新文件(无需登录,只能传小图片)

请求 Data:

名称	长度(字节)	说明

item_type	1	图片:0x0B 视频: 0x15
item_action	4	填 0
windown_id	2	窗口 id
image_name_len	2	图片名称长度
image_name	n1	图片名称: md5 值+后缀 Eg: f405697cf6377e51.png
image_len	4	图片大小
image_data	n2	图片内容(TCP 字节流)

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；

表 8 TCP 数据流更新图片

注: 通过 0x3D 协议，设备接收数据后，按照图片信息保存至设备，之后更新图片到指定文件窗显示，协议里的文件窗 ID，pkg_length 和素材类型(图片:0x0B)务必填写正确。

11. 命令 0x0042，向播放盒上传文件（无需登录,上传到/Resource/，比如字体文件）

上传文件时，如文件长度超过 2048 字节，必须把文件分割成等于 2048 字节的若干段（最后一段为 0-2047 字节），每段依次发送。组帧时，文件指针偏移从 0 开始，每帧依次加 2048，如文件长度正好为 2048 的整数倍，最后也必须发送文件内容为 0 字节的一帧。

根据文件名的不同，上传文件可以实现不同的功能。如文件名为 "uuid.zhpro"，是更改的当前播放列表；文件名为 "003.bmp"，则是上传编号为 003 的一幅位图。

播放盒收到一帧后，若文件内容保存成功，则返回执行结果为 0，ret_info 为写入的文件字节数；否则执行结果为 1，错误信息为一不定长的 ASCII 码字符串，如 "cannot open file XXX for writing"。

接收 Data:

名称	长度（字节）	说明
file_name_len	2	名称长度（md5.jpg，即 md5 值加扩展名长度）
file_name	n1	名称（md5.jpg，即 md5 值加扩展名）
file_offset	4	文件指针偏移（2G 文件）
file_data	n2	一段文件内容

应答 Data:

名称	长度（字节）	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；(文件大小 offset) 1 失败处理返回；(失败原因)
ret_info	n	0：返回 int offset 1：执行错误信息

12. 命令 0x0043，从播放盒下载文件（无需登录）

与上传文件一样，下载文件时也必须把文件分割成等于 2048 字节的若干段，每段依次接收。组帧时，文件指针偏移从 0 开始，每帧依次加 2048，当监控计算机收到文件内容为 0-2047 字节的一帧时，则认为此文件下载完毕。

根据文件名的不同，上传文件可以实现不同的功能。如文件名为 "uuid.zhpro"，是获取播放盒的节目文件；文件名为 "003.bmp"，则是获取编号为 003 的一幅位图。

请求 Data:

名称	长度（字节）	说明

file_name_len	2	名称长度（md5.jpg，即 md5 值加扩展名长度）
file_name	n1	名称（md5.jpg，即 md5 值加扩展名）
file_offset	4	文件指针偏移

应答 Data:

名称	长度（字节）	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回； 1 失败处理返回；
file_size	4	文件总长度
file_data	n	一段文件内容

13. 命令 0x003E 修改文件窗属性(无需登录)

请求 Data(坐标大小):

名称	长度(字节)	说明
item_type	1	window:0x09
item_action	1	0x00:修改坐标 0x01:修改宽高 0x02:修改坐标和宽高
window_count	2	更改窗口的总数量
windown_id_1	2	窗口 1 的 ID
X	2	窗口的 x 坐标
Y	2	窗口的 y 坐标
width	2	窗口的宽度
height	2	窗口的高度
…		(window_id ->height)

请求 Data(不启用背景/边框):

名称	长度(字节)	说明
item_type	1	window:0x09
item_action	1	0x10:不启用背景 0x20:不启用边框
window_count	2	更改窗口的总数量
windown_id_1	2	窗口 1 的 ID
windown_id_2	2	窗口 2 的 ID
…		(window_id)

请求 Data(背景颜色):

名称	长度(字节)	说明
item_type	1	window:0x09
item_action	1	0x11:修改背景颜色
window_count	2	更改窗口的总数量

window_id_1	2	窗口 1 的 ID
window_color	4	int color = 0xff00ff00; 高位在前
…		(window_id -> window_color)

请求 Data(背景图片):

名称	长度(字节)	说明
item_type	1	window:0x09
item_action	1	0x12: 修改背景图片
window_count	2	更改窗口的总数量
windown_id_1	2	窗口 1 的 ID
image_name_len	2	图片名称长度
image_name	n1	背景图片名称: md5 值+后缀 Eg: f405697cf6377e51.png
image_len	4	图片大小(最大支持 2048m 图片)
image_data	n2	图片内容(TCP 字节流)
…		(window_id -> image_data)

请求 Data(修改边框颜色):

名称	长度(字节)	说明
Item_type	1	Window:0x09
Item_action	1	0x21:修改边框颜色
window_count	2	更改窗口的总数量
windown_id_1	2	窗口 1 的 ID
border_Left_Right_width	1	左右边框的宽度
border_Top_Bottom_height	1	上下边框的高度
border_color	4	int color = 0xff00ff00; 高位在前
…		(window_id -> border_color)

请求 Data(修改边框图片):

名称	长度(字节)	说明
Item_type	1	Window:0x09
Item_action	1	0x22:修改边框图片
window_count	2	更改窗口的总数量
windown_id_1	2	窗口 1 的 ID
border_Left_Right_width	1	左右边框的宽度
border_Top_Bottom_height	1	上下边框的高度

image_name_len	2	图片名称长度
image_name	n1	边框图片名称: md5 值+后缀 Eg: f405697cf6377e51.png
image_len	4	图片大小(最大支持 2048m 图片)
image_data	n2	图片内容 (TCP 字节流)
…		(window_id -> image_data)

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；

14. 命令 0x002C 切换上一个节目/素材 （需要先发登录命令）

AA 99 00 2C 00 00 00 00 00 00 00 0F 00 00 00

请求 Data:

名称	长度(字节)	说明
type	1	0: program (节目) 1: item (素材)

应答 Data:

名称	长度(字节)	说明
status	2	处理结果 0 成功处理返回; -1 表示身份未验证; -2 设备已占用;

15. 命令 0x002D 切换下一个节目/素材 （需要先发登录命令）

AA 99 00 2D 00 00 00 00 00 00 00 0F 00 00 00

请求 Data:

名称	长度(字节)	说明
type	1	0: program (节目) 1: item (素材)

应答 Data:

名称	长度(字节)	说明
status	2	处理结果 0 成功处理返回; -1 表示身份未验证;

		-2 设备已占用;

16. 命令 0x0044，播放指定序号的节目 (无需登录)

AA 99 00 44 00 00 00 00 00 00 00 10 00 00 00 01(切换到第 2 个普通节目)

请求 Data:

名称	长度(字节)	说明
program_type	2	0: 普通节目 1: 全局节目 2: 通知
program_index	2	顺序从 0 开始

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；

17. 命令 0x0045，播放指定序号的素材 (无需登录)

AA 99 00 45 00 00 00 00 00 00 00 14 00 00 00 00 00 00 00 01

请求 Data:

名称	长度(字节)	说明
program_type	2	0: 普通节目 1: 全局节目 2: 通知
program_index	2	顺序从 0 开始
window_index	2	窗口序号，从 0 开始
media_index	2	素材索引，从 0 开始

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；

18. 命令 0x0046，更新素材内容到文件窗，根据节目序号，窗口序号(无需登录)

限制: 窗口中素材类型不能重复

请求 Data:

名称	长度(字节)	说明
count	2	素材数量
pkg_len	2	素材包长度(14+n1+2+n2) (program_type->Attributes)
program_type	2	0: 普通节目 1: 全局节目 2: 通知
program_index	2	节目序号 从 0 开始
window_index	2	窗口序号 从 0 开始
item_type	2	0x0C: 文本 0x0B: 图片 0x15: 视频
item_action	2	0x00: 覆盖文本 0x01: 追加文本 0x02: 删除文本 0x03: 自动换行(静态文本)
content_length	2	类型为 0x0C 时: 文本长度 类型为 0x15/0x0B 时: URL 长度

content	n1	类型为 0x0C 时: 文本内容 类型为 0x15/0x0B 时: URL
attributes_length	2	下面 attributes 内容的总长
attributes	n2	格式: 以分号符分割的键值对，Key=Value;Key=Value; **类型为 0x15 或 0x0B 时**: 文件名+扩展名 例: file_name=f405697cf6377e51.jpg; 视频静音 video_mute=1; (0 有声音) **类型为 0x0C 时**: 属性键值对 animation_type=402; (左移: 402，右移: 404，静止多行显示: 403，上移: 405，下移: 406) keep_time=2000: (停留时间: 单位/毫秒，停留时间仅在静止多行显示时有效) word_color=#ff0000;(字体颜色) word_size=24; (字体大小) word_bold=0; (1: 粗体) word_italic=0; (1: 斜体) word_underline=0;(1: 下划线)

应答 Data:

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回； 1 返回下载进度；
progress	4	注: status 为 1 此域存在，下载进度(0-100)

19. 命令 0x0015，清空节目 （需要先发登录命令）

AA 99 00 15 00 00 00 00 00 00 00 0E 00 00

请求 Data

名称	长度(字节)	说明
Type	1	0:(按 id 处理) 1: (按 Level 处理)
用户 ID 长	1	
用户 ID	n	
权限	2	

应答 Data

名称	长度(字节)	说明
status	2	-1 表示身份未验证； -2 设备已占用； 0 成功处理返回；

注: data.len==12 -->不指定 id 和 level，默认清空节目和资源

20. 命令 0x003B，重启设备 （需要先发登录命令）

AA 99 00 3B 00 00 00 00 00 00 00 0E 00 00 （没有 crc 最后 00 需要发送）

发送:

名称	长度(字节)	说明

应答:

名称	长度(字节)	说明
status	2	处理结果 0 成功处理返回; -1 表示身份未验证; -2 设备已占用;

21. 命令 0x0007，亮度/开关屏协议 （需要先发登录命令）

发送:

名称	长度(字节)	说明
pkg	n	参考亮度，开关屏协议

应答:

名称	长度(字节)	说明
status	2	处理结果 0 成功处理返回; -1 表示身份未验证; -2 设备已占用;

- **亮度指令协议**:

AA 99 00 07 00 00 00 00 00 00 00 19 AA 88 0D 00 FF FF 00 00 91 00 FF F1 01 // 亮度 FF(255) F1 01 是 AA88

整个协议作为调屏参数中的 data，用命令字 0x0007 发送给播放盒，调节亮度协议组成: 协议头 + 数据内容 + crc16 校验和

0xAA	0x88	0x0d	0x00	0x0f	0xff	0x00	0x00	0x91	0x00	value	CRC_L	CRC_H

- 亮度指令校验和:

校验和为协议头和数据内容的 crc16 校验和，即 0xAA~value。最终算出来的校验和在组包时，低字节在前，高字节在后

- **开关屏指令协议:**

整个协议作为调屏参数中的 data，用命令字 0x0007 发送给播放盒，调节亮度协议组成: 协议头 + 数据内容 + crc16 校验和

0xAA	0x88	0x0d	0x00	0xff	0xff	0x00	0x00	0x96	0x00	value	CRC_L	CRC_H

AA 99 00 07 00 00 00 00 00 00 00 19 AA 88 0D 00 FF FF 00 00 96 00 01 C1 40 // 开屏
AA 99 00 07 00 00 00 00 00 00 00 19 AA 88 0D 00 FF FF 00 00 96 00 02 81 41 // 关屏

- 开关屏指令数据内容(value，一个字节无符号数):

0x01: 开屏
0x02: 关屏

- 开关屏指令校验和:

校验和为协议头和数据内容的 crc16 校验和，即 0xAA~value，最终的校验和在组包时，低字节在前，高字节在后。

附录 A

A.1 帧协议数据包样例

由于字符更新协议复杂，先提供每个协议的简单样例以供参考。这些通信命令包括:

1、**心跳包**: AA 99 00 01 00 00 00 00 00 00 00 0C

2、**登陆**: AA 99 00 10 00 00 00 00 00 00 00 0E 00 00

3、**更新单素材**(0x1E):

- **数字 eg: "123456"**

AA 99 00 1E 00 00 00 00 00 00 00 1D 0C 00 00 00 00 00 01 00 06 31 32 33 34 35 36 00 00

->:

AA99001E000000000000001D

0C

00000000

0001

0006

313233343536

0000

- eg: "123456"&& word_color=#FF0000

AA 99 00 1E 00 00 00 00 00 00 00 30 0C 00 00 00 00 00 01 00 06 31 32 33 34 35 36 00 13 77 6F 72 64 5F 63 6F 6C 6F 72 3D 23 66 66 30 30 30 30 3B

- 汉字 eg: "欢迎光临" (utf8 字符串转 16 进制)

  AA99001E0000000000000023

  0C

  00000000

  0001

  000c

  E6ACA2E8BF8EE58589E4B8B4

  0000

  - 在线测试链接 https://www.bejson.com/convert/ox2str/
  - Java 代码

```
publicstatic String bytesToHexString(byte[] src) {
        StringBuilder stringBuilder = new StringBuilder("");
        if (src == null || src.length<= 0) {
            returnnull;
        }
        for (inti = 0; i<src.length; i++) {
            intv = src[i] & 0xFF;
            String hv = Integer.toHexString(v);
            if (hv.length() < 2) {
                stringBuilder.append(0);
            }
            stringBuilder.append(hv);
        }
        returnstringBuilder.toString();
}

publicstaticvoid main(String[] args) {
        String s = "欢迎光临";
        try {
            byte[] data = s.getBytes("utf8");
```

```
            System.out.println(bytesToHexString(data));
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }
    }
```

16 进制到文本字符串的转换，16 进制-BeJSON.com

(3)图片视频示例

eg:

Item_type:0x0B; Windown_id:3;

Content :" http:/192.168.9.117:8185/d:/meigui.png";

Attributes:" 5d2e2f201446db9280f9cddfe4c222c8.png")

->:

AA99001E 00000000 00000061 0B 00000000 0003

004c

687474703a2f3139322e3136382e392e3131373a383138352f643a2f6d65696775692e706e67

0048

3564326532663230313434366462393238306639636464666534633232323263382e706e67

4、更新多素材(0x3C)

eg: window_id=1:"123456" window_id=2:"654321"

AA 99 00 3C 00 00 00 00 00 00 00 34 00 02

00 11 0C 00 00 00 00 00 01 00 06 31 32 33 34 35 36 00 00

00 11 0C 00 00 00 00 00 02 00 06 36 32 34 33 32 31 00 00

4、修改文件窗属性(0x3E)

eg:

(1)更改背景颜色

AA 99 00 3E 00 00 00 00 00 00 00 1C 09 11 00 02 00 01 00 FF 00 00 00 02 00 00 FF 00

->

AA 99 00 3E 00 00 00 00 00 00 00 1C

09 11

00 02

00 01 00 FF 00 00

00 02 00 00 FF 00

(2)更改边框颜色

AA 99 00 3E 00 00 00 00 00 00 00 20 09 21 00 02 00 01 05 05 00 FF 00 00 00 02 08 08 00 00

FF 00

->

AA 99 00 3E 00 00 00 00 00 00 00 1C

09 21

00 02

00 01 05 05 00 FF 00 00

00 02 08 08 00 00 00 FF

A.2 素材类型和特效枚举

(1)素材类型

```
enum{
    Fragment = 4,
    GlobalFragment = 5,
    NotifyFragment = 6,
    GlogalMutilWindow = 8,
    NormalMutilWindow = 9,
    ZHImage = 11,
    SingleText = 12,
    MultiText = 13,
    Gif = 14,
    Clock = 15,
    Word = 16,
    Excel = 17,
    PPt = 18,
    RTF = 19,
    Flash = 20,
    Video = 21,
    Table = 23,
    Txt = 24,
    Weather = 28,
    Web = 29,
    ColorfulText = 33,
    SimpleTextWindow = 34,
    ColoredTextWindow = 35,
```

```
    GlobalSimpleTextWindow = 36,
    GlobalColoredTextWindow = 37,
    LEDScreen = 38,
    Templet = 39,
    Camera = 40,
    SportScore = 41,
    Sensor = 42,
    DBTable = 43,
    JSON = 44,
    WebVideo = 45,
    NewSensor = 46,
    ModbusTableItem = 47
}
```

(2)特效类型

```
  enum{
    Nothing = 0,  //无转场效果
    Random = 1,       //随机效果
    Waitting = 2,     //等待
    MoveUpConnect = 3, //连续上移
    //移动
    MoveUp = 20,
    MoveDown,
    MoveLeft,
    MoveRight,
    MoveUpperLeftCorner,
    MoveUpperRightCorner,
    MoveBottomLeftCorner,
    MoveBottomRightCorner,
    MoveFlash,
    MoveUp_continuous,
    MoveDown_continuous,
    MoveLeft_continuous,
    MoveRight_continuous,
```

```
//渐变效果，淡入淡出
GradientFade = 40,
//缩放效果
ZoomCenter = 60,
ZoomUpperLeft,
ZoomUpperRight,
ZoomBottomLeft,
ZoomBottomRight,
//覆盖效果
CoverTop = 80,
CoverBottom,
CoverLeft,
CoverRight,
CoverUpperLeftDiagonal,
CoverUpperRightDiagonal,
CoverBottomLeftDiagonal,
CoverBottomRightDiagonal,
CoverUpperLeftAngle,
CoverUpperRightAngle,
CoverBottomLeftAngle,
CoverBottomRightAngle,
//时钟效果
ClockCW360 = 100,
ClockCCW360,
ClockCW180,
ClockCCW180,
ClockCW90,
ClockCCW90,
//拉幕效果
CurtainLeftOff = 120,
CurtainClosedLeft,
CurtainDownOff,
CurtainClosedDown,
```

```
//百叶窗效果
BliendsHorizontal = 140,
BliendsVertical,
//马赛克效果
MosaicSuperSmall = 160,
MosaicSmall,
MosaicMiddle,
MosaicLarge,
//形状效果
ShapeRectangularExpand = 180,
ShapeRectangularContraction,
ShapeDiamondExpand,
ShapeDiamondContraction,
ShapeCrossExpand,
ShapeCrossContraction,
ShapeRoundExpand,
ShapeCircularContraction,
//////////////////文字动画//////////////////
TextFragmentsSmall = 200,   //小碎片
TextFragmentsMiddle,
TextFragmentsLarge,
TextType = 400, //打字
TextFlop = 401, //跌落
TextMoveLeft = 402,//左移
TextStatic = 403,//静止
TextMoveRight= 404, //右移
TextMoveUp= 405,//上移
TextMoveDown = 406,//下移
TextStaticLeft = 500; // 静止居左
TextStaticRight = 501; // 静止居右
TextTwinkle,
TextContinuityLeft, //单行文本连续左移
TextContinuityRight,//单行文本连续右移
```

```
        TextContinuityUp,    //单行文本连续上移
        TextContinuityDown, //单行文本连续下移
        TextLRCenter,
        TextUDCenter
    };
```

附录 B

（规范性附录）

16 位 CRC 算法校验

B.1 C 语言实现

```
unsigned short get_crc(const unsigned char * buffer, int buffer_length)
{
    unsigned short CRCFull = 0xFFFF;
    unsigned char CRCLSB;
    int i = 0, j = 0;
    unsigned char *mess = (unsigned char *) buffer;

    for (i = 0; i < buffer_length; i++)//
    {
        CRCFull = (unsigned short)(CRCFull ^ mess[i]);
        for (j = 0; j < 8; j++)
        {
            CRCLSB = (unsigned char)(CRCFull & 0x0001);
            CRCFull = (unsigned short)((CRCFull >> 1) & 0x7FFF);

            if (CRCLSB == 1)
                CRCFull = (unsigned short)(CRCFull ^ 0xA001);
        }
    }
    return CRCFull;
}
```

B.2 Java 语言实现

```
publicstatic String bytesToHex(byte[] bytes) {
        StringBuffer sb = new StringBuffer();
        for (inti = 0; i<bytes.length; i++) {
            String hex = Integer.toHexString(bytes[i] & 0xFF);
            if (hex.length() < 2) {
                sb.append(0);
            }
            sb.append(hex);
        }
```

```
        returnsb.toString();
    }

    publicstaticbyte[] calcCrc16(byte[] data) {
        intverLen = data.length - 2;
        intverify = calcCrc16(data, verLen);
        data[verLen] = (byte) (verify& 0xff); // 校验位
        data[verLen + 1] = (byte) (verify>> 8 & 0xff); // 校验位
        returndata;
    }

    publicstaticint calcCrc16(byte[] data, intlen) {
        returncalcCrc16(data, 0, len);
    }

    publicstaticint calcCrc16(byte[] data, intoffset, intlen) {
        returncalcCrc16(data, offset, len, 0xffff);
    }

    privatestaticint calcCrc16(byte[] data, intoffset, intlen, intprevail) {
        intucCRCHi = (prevail& 0xff00) >> 8;
        intucCRCLo = prevail& 0x00ff;
        intiIndex;
        for (inti = 0; i<len; ++i) {
            iIndex = (ucCRCLo ^ data[offset + i]) & 0x00ff;
            ucCRCLo = ucCRCHi ^ crc16_tab_h[iIndex];
            ucCRCHi = crc16_tab_l[iIndex];
        }
        return ((ucCRCHi& 0x00ff) << 8) | (ucCRCLo& 0x00ff) & 0xffff;
    }

    privatestaticbyte[] crc16_tab_h = { (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
```

```
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x00, (byte) 0xC1, (byte) 0x81, (byte) 0x40,
(byte) 0x01, (byte) 0xC0,
            (byte) 0x80, (byte) 0x41, (byte) 0x01, (byte) 0xC0, (byte) 0x80, (byte) 0x41,
(byte) 0x00, (byte) 0xC1,
            (byte) 0x81, (byte) 0x40 };

    privatestaticbyte[] crc16_tab_l = { (byte) 0x00, (byte) 0xC0, (byte) 0xC1, (byte) 0x01,
(byte) 0xC3, (byte) 0x03,
            (byte) 0x02, (byte) 0xC2, (byte) 0xC6, (byte) 0x06, (byte) 0x07, (byte) 0xC7,
(byte) 0x05, (byte) 0xC5,
            (byte) 0xC4, (byte) 0x04, (byte) 0xCC, (byte) 0x0C, (byte) 0x0D, (byte) 0xCD,
(byte) 0x0F, (byte) 0xCF,
            (byte) 0xCE, (byte) 0x0E, (byte) 0x0A, (byte) 0xCA, (byte) 0xCB, (byte) 0x0B,
(byte) 0xC9, (byte) 0x09,
            (byte) 0x08, (byte) 0xC8, (byte) 0xD8, (byte) 0x18, (byte) 0x19, (byte) 0xD9,
(byte) 0x1B, (byte) 0xDB,
            (byte) 0xDA, (byte) 0x1A, (byte) 0x1E, (byte) 0xDE, (byte) 0xDF, (byte) 0x1F,
(byte) 0xDD, (byte) 0x1D,
            (byte) 0x1C, (byte) 0xDC, (byte) 0x14, (byte) 0xD4, (byte) 0xD5, (byte) 0x15,
(byte) 0xD7, (byte) 0x17,
            (byte) 0x16, (byte) 0xD6, (byte) 0xD2, (byte) 0x12, (byte) 0x13, (byte) 0xD3,
(byte) 0x11, (byte) 0xD1,
            (byte) 0xD0, (byte) 0x10, (byte) 0xF0, (byte) 0x30, (byte) 0x31, (byte) 0xF1,
(byte) 0x33, (byte) 0xF3,
            (byte) 0xF2, (byte) 0x32, (byte) 0x36, (byte) 0xF6, (byte) 0xF7, (byte) 0x37,
```

```
(byte) 0xF5, (byte) 0x35,
            (byte) 0x34, (byte) 0xF4, (byte) 0x3C, (byte) 0xFC, (byte) 0xFD, (byte) 0x3D,
(byte) 0xFF, (byte) 0x3F,
            (byte) 0x3E, (byte) 0xFE, (byte) 0xFA, (byte) 0x3A, (byte) 0x3B, (byte) 0xFB,
(byte) 0x39, (byte) 0xF9,
            (byte) 0xF8, (byte) 0x38, (byte) 0x28, (byte) 0xE8, (byte) 0xE9, (byte) 0x29,
(byte) 0xEB, (byte) 0x2B,
            (byte) 0x2A, (byte) 0xEA, (byte) 0xEE, (byte) 0x2E, (byte) 0x2F, (byte) 0xEF,
(byte) 0x2D, (byte) 0xED,
            (byte) 0xEC, (byte) 0x2C, (byte) 0xE4, (byte) 0x24, (byte) 0x25, (byte) 0xE5,
(byte) 0x27, (byte) 0xE7,
            (byte) 0xE6, (byte) 0x26, (byte) 0x22, (byte) 0xE2, (byte) 0xE3, (byte) 0x23,
(byte) 0xE1, (byte) 0x21,
            (byte) 0x20, (byte) 0xE0, (byte) 0xA0, (byte) 0x60, (byte) 0x61, (byte) 0xA1,
(byte) 0x63, (byte) 0xA3,
            (byte) 0xA2, (byte) 0x62, (byte) 0x66, (byte) 0xA6, (byte) 0xA7, (byte) 0x67,
(byte) 0xA5, (byte) 0x65,
            (byte) 0x64, (byte) 0xA4, (byte) 0x6C, (byte) 0xAC, (byte) 0xAD, (byte) 0x6D,
(byte) 0xAF, (byte) 0x6F,
            (byte) 0x6E, (byte) 0xAE, (byte) 0xAA, (byte) 0x6A, (byte) 0x6B, (byte) 0xAB,
(byte) 0x69, (byte) 0xA9,
            (byte) 0xA8, (byte) 0x68, (byte) 0x78, (byte) 0xB8, (byte) 0xB9, (byte) 0x79,
(byte) 0xBB, (byte) 0x7B,
            (byte) 0x7A, (byte) 0xBA, (byte) 0xBE, (byte) 0x7E, (byte) 0x7F, (byte) 0xBF,
(byte) 0x7D, (byte) 0xBD,
            (byte) 0xBC, (byte) 0x7C, (byte) 0xB4, (byte) 0x74, (byte) 0x75, (byte) 0xB5,
(byte) 0x77, (byte) 0xB7,
            (byte) 0xB6, (byte) 0x76, (byte) 0x72, (byte) 0xB2, (byte) 0xB3, (byte) 0x73,
(byte) 0xB1, (byte) 0x71,
            (byte) 0x70, (byte) 0xB0, (byte) 0x50, (byte) 0x90, (byte) 0x91, (byte) 0x51,
(byte) 0x93, (byte) 0x53,
            (byte) 0x52, (byte) 0x92, (byte) 0x96, (byte) 0x56, (byte) 0x57, (byte) 0x97,
(byte) 0x55, (byte) 0x95,
            (byte) 0x94, (byte) 0x54, (byte) 0x9C, (byte) 0x5C, (byte) 0x5D, (byte) 0x9D,
(byte) 0x5F, (byte) 0x9F,
            (byte) 0x9E, (byte) 0x5E, (byte) 0x5A, (byte) 0x9A, (byte) 0x9B, (byte) 0x5B,
(byte) 0x99, (byte) 0x59,
            (byte) 0x58, (byte) 0x98, (byte) 0x88, (byte) 0x48, (byte) 0x49, (byte) 0x89,
(byte) 0x4B, (byte) 0x8B,
            (byte) 0x8A, (byte) 0x4A, (byte) 0x4E, (byte) 0x8E, (byte) 0x8F, (byte) 0x4F,
(byte) 0x8D, (byte) 0x4D,
            (byte) 0x4C, (byte) 0x8C, (byte) 0x44, (byte) 0x84, (byte) 0x85, (byte) 0x45,
(byte) 0x87, (byte) 0x47,
            (byte) 0x46, (byte) 0x86, (byte) 0x82, (byte) 0x42, (byte) 0x43, (byte) 0x83,
(byte) 0x41, (byte) 0x81,
            (byte) 0x80, (byte) 0x40 };
```