**波特率：**115200

**PC端数据格式(整个包最大只能4200个数据)：**

0x55(包头分界符) +命令（1字节）+ n（2字节,低8位,高8位）+数据（n字节）+ checksum(1字节，0-整个包相加)+0xaa（包尾分界符）。

例如：0x55+0x01+0x02+0x00+0x66+0x99+0xff+0xaa

**IO端数据格式(整个包最大只能20个数据)：**

0x55(包头分界符) +pc对应命令（1字节）+ n（2字节,低8位,高8位）+数据（n字节）+ checksum(1字节，0-整个包相加)+0xaa（包尾分界符）。

例如：0x55+0x01+0x02+0x00+0x00+0x00+0xfe+0xaa

**命令含义解释：**

**0x00 :** **初始化端口**

PC->IO：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 端口 | 灯珠数 | 灯珠类型 | 段数 | 亮度 | 检验 | 帧尾 |
| 55 | 00 | 4 | 0~6 | 0~255 | 见下表 | 0~255 | 0~255 | 0-Sum | aa |
| 例如：初始化第一个端口，灯珠总数为20颗，灯珠类型是RBG，分成4段，亮度最亮255  55 00 05 00 00 14 06 04 FF DF AA  注意：如果灯珠红绿蓝颜色对不上，必须选择合适的灯珠类型。  **端口选择：**  enum {  Port1,  Port2,  Port3,  Port4,  Port5,  Port6,  Port7,  Port\_MAX //!< MaxProt  };  **灯珠类型选择：**  // RGB: W R G B  #define NEO\_RGB ((0<<6) | (0<<4) | (1<<2) | (2)) ///< Transmit as R,G,B  #define NEO\_RBG ((0<<6) | (0<<4) | (2<<2) | (1)) ///< Transmit as R,B,G  #define NEO\_GRB ((1<<6) | (1<<4) | (0<<2) | (2)) ///< Transmit as G,R,B  #define NEO\_GBR ((2<<6) | (2<<4) | (0<<2) | (1)) ///< Transmit as G,B,R  #define NEO\_BRG ((1<<6) | (1<<4) | (2<<2) | (0)) ///< Transmit as B,R,G  #define NEO\_BGR ((2<<6) | (2<<4) | (1<<2) | (0)) ///< Transmit as B,G,R  // RGBW NeoPixel permutations; all 4 offsets are distinct  // RGBW : W R G B  #define NEO\_WRGB ((0<<6) | (1<<4) | (2<<2) | (3)) ///< Transmit as W,R,G,B  #define NEO\_WRBG ((0<<6) | (1<<4) | (3<<2) | (2)) ///< Transmit as W,R,B,G  #define NEO\_WGRB ((0<<6) | (2<<4) | (1<<2) | (3)) ///< Transmit as W,G,R,B  #define NEO\_WGBR ((0<<6) | (3<<4) | (1<<2) | (2)) ///< Transmit as W,G,B,R  #define NEO\_WBRG ((0<<6) | (2<<4) | (3<<2) | (1)) ///< Transmit as W,B,R,G  #define NEO\_WBGR ((0<<6) | (3<<4) | (2<<2) | (1)) ///< Transmit as W,B,G,R  #define NEO\_RWGB ((1<<6) | (0<<4) | (2<<2) | (3)) ///< Transmit as R,W,G,B  #define NEO\_RWBG ((1<<6) | (0<<4) | (3<<2) | (2)) ///< Transmit as R,W,B,G  #define NEO\_RGWB ((2<<6) | (0<<4) | (1<<2) | (3)) ///< Transmit as R,G,W,B  #define NEO\_RGBW ((3<<6) | (0<<4) | (1<<2) | (2)) ///< Transmit as R,G,B,W  #define NEO\_RBWG ((2<<6) | (0<<4) | (3<<2) | (1)) ///< Transmit as R,B,W,G  #define NEO\_RBGW ((3<<6) | (0<<4) | (2<<2) | (1)) ///< Transmit as R,B,G,W  #define NEO\_GWRB ((1<<6) | (2<<4) | (0<<2) | (3)) ///< Transmit as G,W,R,B  #define NEO\_GWBR ((1<<6) | (3<<4) | (0<<2) | (2)) ///< Transmit as G,W,B,R  #define NEO\_GRWB ((2<<6) | (1<<4) | (0<<2) | (3)) ///< Transmit as G,R,W,B  #define NEO\_GRBW ((3<<6) | (1<<4) | (0<<2) | (2)) ///< Transmit as G,R,B,W  #define NEO\_GBWR ((2<<6) | (3<<4) | (0<<2) | (1)) ///< Transmit as G,B,W,R  #define NEO\_GBRW ((3<<6) | (2<<4) | (0<<2) | (1)) ///< Transmit as G,B,R,W  #define NEO\_BWRG ((1<<6) | (2<<4) | (3<<2) | (0)) ///< Transmit as B,W,R,G  #define NEO\_BWGR ((1<<6) | (3<<4) | (2<<2) | (0)) ///< Transmit as B,W,G,R  #define NEO\_BRWG ((2<<6) | (1<<4) | (3<<2) | (0)) ///< Transmit as B,R,W,G  #define NEO\_BRGW ((3<<6) | (1<<4) | (2<<2) | (0)) ///< Transmit as B,R,G,W  #define NEO\_BGWR ((2<<6) | (3<<4) | (1<<2) | (0)) ///< Transmit as B,G,W,R  #define NEO\_BGRW ((3<<6) | (2<<4) | (1<<2) | (0)) ///< Transmit as B,G,R,W | | | | | | | | | |

IO->PC：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 00 | 00 | 0-Sum | aa |
| 例如：55 00 00 00 01 AA | | | | |

**0x01 :停止端口更新**

PC->IO：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 端口 | 检验 | 帧尾 |
| 55 | 01 | 01 | 0~6 | 0-Sum | aa |
| 例如：停止第一端口的更新  55 01 01 00 00 FF AA  **端口选择：**  enum {  Port1,  Port2,  Port3,  Port4,  Port5,  Port6,  Port7,  Port\_MAX //!< MaxProt  }; | | | | | |

IO->PC：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 01 | 00 | 0-Sum | aa |
| 例如：55 01 00 00 00 AA | | | | |

**0x02 :开启端口更新**

PC->IO：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 端口 | 检验 | 帧尾 |
| 55 | 02 | 01 | 0~6 | 0-Sum | aa |
| 例如：开启第一端口的更新  55 02 01 00 00 FE AA  **端口选择：**  enum {  Port1,  Port2,  Port3,  Port4,  Port5,  Port6,  Port7,  Port\_MAX //!< MaxProt  }; | | | | | |

IO->PC：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 02 | 00 | 0-Sum | aa |
| 例如：55 02 00 00 FF AA | | | | |

**0x03 :** **设置固定颜色表,不配置就用默认的**

PC->IO：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 颜色成分\*12 | | | | 检验 | 帧尾 |
| B | G | R | W |
| 55 | 03 | 12\*4 | 0~255 | 0~255 | 0~255 | 0~255 | 0-Sum | aa |
| 备注：一定要一次性发送48个数据(即12种颜色)，例如只是想改变默认颜色表中的黄色(YELLOW),也要把其他的11种颜色按默认表中的数据，重新发送。  55 03 30 00 00 00 FF 00 00 FF 00 00 FF 00 00 00 FF FF FF 00 00 00 00 00 00 FF FF 00 FF FF 00 00 FF 00 FF 00 80 FF 40 00 00 30 FF 00 93 14 FF 00 10 10 10 00 16 AA  **以下是默认的颜色列表，一般情况下不需要去重新配置：**  // FixedColor\_buf一些常见的颜色  #define RED 0  #define GREEN 1  #define BLUE 2  #define WHITE 3  #define BLACK 4  #define YELLOW 5  #define CYAN 6  #define MAGENTA 7  #define PURPLE 8  #define ORANGE 9  #define PINK 10  #define GRAY 11  #define FixedColor\_Max 12  uint32\_t FixedColor[FixedColor\_Max]={  0x00ff0000,  0x0000FF00,  0x000000ff,  0x00FFFFFF,  0x00000000,  0x00FFFF00,  0x0000FFFF,  0x00FF00FF,  0x00400080,  0x00FF3000,  0x00FF1493,  0x00101010  }; | | | | | | | | |

IO->PC：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 03 | 00 | 0-Sum | aa |
| 例如：55 03 00 00 fe AA | | | | |

**0x04 :** **设置灯光模式**

PC->IO：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 端口 | 段号 | 开始灯珠 | 结束灯珠 | 模式 | 颜色1 | 颜色2 | 颜色3 | 速度 | 选项 | 检验 | 帧尾 |
| 55 | 04 | 11 | 0~6 | 0~255 | 0~255 | 1~255 | 0~72 | 0~11 | 0~11 | 0~11 | 1~65535 |  | 0-Sum | aa |
| 例如：灯带接在第一端口，共分成了4段，假设设置第0段的开始灯珠从第9颗开始，第20颗灯结束，设置在闪烁模式，红色和白色闪烁，周期1S(500MS红，500ms白)，需要伽马校正  55 04 0A 00 00 00 08 13 01 00 03 02 0A 08 00 C0 AA  注意：  1、段号0必须要先使用，后面才可使用别的段号(例如初始化时设置位5个段号，段号0必须先配置，后面才可以其他的段号)  2、如果使用FX\_MODE\_FLIPBOOK模式，需先执行05指令将要显示的内容存入灯板上的Flash，然后再调用执行06指令，把需要显示的内容调取出来存入相应的端口，再执行FX\_MODE\_FLIPBOOK模式(需要注意的是一个端口只能使用一个FX\_MODE\_FLIPBOOK模式)。  **端口选择：**  enum {  Port1,  Port2,  Port3,  Port4,  Port5,  Port6,  Port7,  Port\_MAX //!< MaxProt  };  **段号选择：**  根据00指令(初始化指令)中的段数进行选择  **模式选择：**  #define FX\_MODE\_STATIC 0 //静态模式，灯光保持固定颜色不变  #define FX\_MODE\_BLINK 1 //闪烁模式，灯光以一定频率闪烁  #define FX\_MODE\_BREATH 2 //呼吸模式，灯光亮度像呼吸一样逐渐变亮再变暗  #define FX\_MODE\_COLOR\_WIPE 3 //单色擦除模式，一种颜色从一端逐渐擦除到另一端  #define FX\_MODE\_COLOR\_WIPE\_INV 4 //反向单色擦除模式，单色擦除方向与COLOR\_WIPE相反  #define FX\_MODE\_COLOR\_WIPE\_REV 5 //反转单色擦除模式，颜色顺序反转的单色擦除  #define FX\_MODE\_COLOR\_WIPE\_REV\_INV 6 //反转反向单色擦除模式，结合了反转和反向的单色擦除  #define FX\_MODE\_COLOR\_WIPE\_RANDOM 7 //随机单色擦除模式，随机颜色进行擦除  #define FX\_MODE\_RANDOM\_COLOR 8 //随机颜色模式，灯光随机变换颜色  #define FX\_MODE\_SINGLE\_DYNAMIC 9 //单动态模式，单个灯光动态变化  #define FX\_MODE\_MULTI\_DYNAMIC 10 //多动态模式，多个灯光同时动态变化  #define FX\_MODE\_RAINBOW 11 //彩虹模式，灯光呈现彩虹般的色彩渐变效果  #define FX\_MODE\_RAINBOW\_CYCLE 12 //彩虹循环模式，彩虹效果循环滚动  #define FX\_MODE\_SCAN 13 //扫描模式，灯光像扫描仪一样来回扫描  #define FX\_MODE\_DUAL\_SCAN 14 //双重扫描模式，两个扫描光束同时进行扫描  #define FX\_MODE\_FADE 15 //渐变模式，灯光颜色或亮度逐渐变化(跟2差不多)  #define FX\_MODE\_THEATER\_CHASE 16 //剧院追逐模式，灯光像剧院灯光一样追逐闪烁  #define FX\_MODE\_THEATER\_CHASE\_RAINBOW 17 //剧院追逐彩虹模式，结合了剧院追逐和彩虹效果  #define FX\_MODE\_RUNNING\_LIGHTS 18 //流动灯光模式，灯光像流水一样向前流动  #define FX\_MODE\_TWINKLE 19 //闪烁模式，灯光像星星一样闪烁  #define FX\_MODE\_TWINKLE\_RANDOM 20 //随机闪烁模式，灯光随机位置随机闪烁  #define FX\_MODE\_TWINKLE\_FADE 21 //渐变闪烁模式，闪烁时带有渐变效果  #define FX\_MODE\_TWINKLE\_FADE\_RANDOM 22 //随机渐变闪烁模式，随机位置渐变闪烁  #define FX\_MODE\_SPARKLE 23 //闪烁模式，灯光像火花一样闪烁  #define FX\_MODE\_FLASH\_SPARKLE 24 //闪光闪烁模式，闪烁时带有闪光效果  #define FX\_MODE\_HYPER\_SPARKLE 25 //超级闪烁模式，更强烈的闪烁效果  #define FX\_MODE\_STROBE 26 //频闪模式，灯光快速闪烁产生频闪效果  #define FX\_MODE\_STROBE\_RAINBOW 27 //彩虹频闪模式，频闪时呈现彩虹颜色  #define FX\_MODE\_MULTI\_STROBE 28 //多重频闪模式，多个频闪光束同时出现  #define FX\_MODE\_BLINK\_RAINBOW 29 //彩虹闪烁模式，闪烁时呈现彩虹颜色  #define FX\_MODE\_CHASE\_WHITE 30 //白色追逐模式，白色灯光进行追逐效果  #define FX\_MODE\_CHASE\_COLOR 31 //彩色追逐模式，彩色灯光进行追逐效果  #define FX\_MODE\_CHASE\_RANDOM 32 //随机追逐模式，随机颜色灯光进行追逐  #define FX\_MODE\_CHASE\_RAINBOW 33 //彩虹追逐模式，彩虹颜色灯光进行追逐  #define FX\_MODE\_CHASE\_FLASH 34 //追逐闪光模式，追逐时带有闪光效果  #define FX\_MODE\_CHASE\_FLASH\_RANDOM 35 //随机追逐闪光模式，随机颜色追逐时闪光  #define FX\_MODE\_CHASE\_RAINBOW\_WHITE 36 //彩虹白追逐模式，彩虹与白色灯光结合追逐  #define FX\_MODE\_CHASE\_BLACKOUT 37 //熄灭追逐模式，灯光在追逐过程中有熄灭效果  #define FX\_MODE\_CHASE\_BLACKOUT\_RAINBOW 38 //彩虹熄灭追逐模式，彩虹灯光在追逐时有熄灭效果  #define FX\_MODE\_COLOR\_SWEEP\_RANDOM 39 //随机颜色扫过模式，随机颜色像扫帚一样扫过灯光  #define FX\_MODE\_RUNNING\_COLOR 40 //流动颜色模式，颜色像水流一样在灯光上流动  #define FX\_MODE\_RUNNING\_RED\_BLUE 41 //红蓝流动模式，红蓝两色灯光依次流动  #define FX\_MODE\_RUNNING\_RANDOM 42 //随机流动模式，随机颜色灯光依次流动  #define FX\_MODE\_LARSON\_SCANNER 43 //拉尔森扫描模式，灯光像扫描仪一样来回扫描，常用于科幻效果  #define FX\_MODE\_COMET 44 //彗星模式，灯光像彗星尾巴一样流动  #define FX\_MODE\_FIREWORKS 45 //烟花模式，灯光像烟花一样绽放  #define FX\_MODE\_FIREWORKS\_RANDOM 46 //随机烟花模式，烟花绽放的位置和颜色随机  #define FX\_MODE\_MERRY\_CHRISTMAS 47 //圣诞模式，呈现圣诞主题的灯光效果，如闪烁的彩灯等  #define FX\_MODE\_FIRE\_FLICKER 48 //火焰闪烁模式，灯光像火焰一样闪烁  #define FX\_MODE\_FIRE\_FLICKER\_SOFT 49 //柔和火焰闪烁模式，火焰闪烁效果更柔和  #define FX\_MODE\_FIRE\_FLICKER\_INTENSE 50 //强烈火焰闪烁模式，火焰闪烁效果更强烈  #define FX\_MODE\_CIRCUS\_COMBUSTUS 51 //马戏团火焰模式，呈现马戏团表演中的火焰效果  #define FX\_MODE\_HALLOWEEN 52 //万圣节模式，呈现万圣节主题的灯光效果，如幽灵、南瓜灯等  #define FX\_MODE\_BICOLOR\_CHASE 53 //双色追逐模式，两种颜色的灯光进行追逐  #define FX\_MODE\_TRICOLOR\_CHASE 54 //三色追逐模式，三种颜色的灯光进行追逐  #define FX\_MODE\_TWINKLEFOX 55 //闪烁狐模式，灯光像狐狸眼睛一样闪烁  #define FX\_MODE\_BLOCK\_DISSOLVE 56 //方块溶解模式，灯光像方块一样逐渐溶解消失  #define FX\_MODE\_ICU 57 //重症监护模式，呈现类似医院重症监护室的灯光效果  #define FX\_MODE\_DUAL\_LARSON 58 //双重拉尔森模式，两个拉尔森扫描光束同时出现  #define FX\_MODE\_RUNNING\_RANDOM2 59 //另一种随机流动模式，与RUNNING\_RANDOM有所不同  #define FX\_MODE\_FILLER\_UP 60 //填充上升模式，灯光从底部向上逐渐填充  #define FX\_MODE\_RAINBOW\_LARSON 61 //彩虹拉尔森模式，拉尔森扫描光束呈现彩虹颜色  #define FX\_MODE\_TRIFADE 62 //三色渐变模式，三种颜色之间相互渐变  #define FX\_MODE\_HEARTBEAT 63 //心跳模式，灯光像心跳一样有节奏地闪烁  #define FX\_MODE\_VU\_METER 64 //音量表模式，灯光根据音量大小变化，常用于音响设备  #define FX\_MODE\_BITS 65 //比特模式，灯光像二进制比特一样闪烁  #define FX\_MODE\_MULTI\_COMET 66 //多彗星模式，多个彗星尾巴效果同时出现  #define FX\_MODE\_FLIPBOOK 67 //翻页书模式，灯光像翻页书一样快速切换效果  #define FX\_MODE\_POPCORN 68 //爆米花模式，灯光像爆米花一样随机爆开  #define FX\_MODE\_OSCILLATOR 69 //振荡器模式，灯光像振荡器一样有规  #define UFX\_MODE\_COMET1 70 //彩色流星雨扫来扫去  #define UFX\_MODE\_COMET2 71 //彩色流星雨扫来扫去  #define UFX\_MODE\_COMET3 72 //彩色流星雨扫来扫去  **颜色选择：**  根据03指令(设置固定颜色表指令)中的颜色编号。在某些模式中，此值是无效的  **速度：**  此值为灯光变化的速度，在某些模式中，此值是无效的，例如模式2，模式15.  **选项：**   1. 当没有特殊的设置时，可以选择“NO\_OPTIONS”。 2. 针对流水灯效果的，可以通过“REVERSE”来设置方向。。 3. 针对带流星效果、烟花、心跳效果的可以使用褪色速率FADE\_ XFAST~ FADE\_GLACIAL。 4. 如果需要对色彩进行伽马校正的，可以通过“GAMMA”来设置。 5. 非整段同时亮一种颜色的，都可以使用SIZE\_SMALL~ SIZE\_XLARGE来设置亮起颜色的长度。 6. 需要多个同时配置时，可以或起来后使用，例如REVERSE | FADE\_FAST | SIZE\_SMALL。     #define NO\_OPTIONS (uint8\_t)B00000000 //没有任何选项被设置。  #define REVERSE (uint8\_t)B10000000 //动画反转选项。如果设置了这个位，动画将反向播放  #define FADE\_XFAST (uint8\_t)B00010000 //这些宏定义了不同的褪色速率（fade rate），从极快（FAST）到极慢（GLACIAL）。每个宏定义了不同的位模式，用于设置 LED 段的褪色速率。  #define FADE\_FAST (uint8\_t)B00100000  #define FADE\_MEDIUM (uint8\_t)B00110000  #define FADE\_SLOW (uint8\_t)B01000000  #define FADE\_XSLOW (uint8\_t)B01010000  #define FADE\_XXSLOW (uint8\_t)B01100000  #define FADE\_GLACIAL (uint8\_t)B01110000  #define GAMMA (uint8\_t)B00001000 //开启伽马校正选项。如果设置了这个位，LED 的颜色将应用伽马校正。  #define SIZE\_SMALL (uint8\_t)B00000000 //这些宏定义了段的大小选项，从小型（SMALL）到超大型（XLARGE）。每个宏定义了不同的位模式，用于设置 LED 段的大小。  #define SIZE\_MEDIUM (uint8\_t)B00000010  #define SIZE\_LARGE (uint8\_t)B00000100  #define SIZE\_XLARGE (uint8\_t)B000001100 | | | | | | | | | | | | | | |

IO->PC：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 04 | 00 | 0-Sum | aa |
| 例如：55 04 00 00 fd AA | | | | |

**0x05 :按动态显示的数据存入flash**

PC->IO：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 块号 | 扇区 | **存入Flash数据(见下面说明)** | | | | 检验 | 帧尾 |
| 55 | 05 |  | 0-255 | 0~15 | 帧数（有几帧画面） | 行数 | 列数 | 颜色数据 | 0-Sum | aa |
| Len：每次传输最多不能大于4098(减掉块区和扇区，也就是每次存入Flash不能超过4096)  **块：每一种动态显示效果存入同一块。**  **扇区：每一块里面有0~15个扇区，必须从0扇区开始存入，不能跨越。**  **存入Flash数据说明：**  用户需先将帧数、行数、列数、先存入然后再存入颜色数据，，  帧数（有几帧画面）：1~0xff；  行数：1~0xff；**，****例如下表，行数是12**  列数：1~0xff；**，例如下表，列数是4**  颜色数据：**(一颗灯珠由RGBW组成,有些灯带是不带W的)**：0~0xffffffff，**先发低位再发高位，颜色数据按下表的灯珠1~48发送。**  **第一帧数据：**   |  |  |  |  | | --- | --- | --- | --- | | **1红** | **24灰** | **25红** | **48灰** | | **2绿** | **23粉** | **26绿** | **47粉** | | **3蓝** | **22橙** | **27蓝** | **46橙** | | **4白** | **21紫** | **28白** | **45紫** | | **5黑** | **20紫红** | **29黑** | **44紫红** | | **6黄** | **19青** | **30黄** | **43青** | | **7青** | **18黄** | **31青** | **42黄** | | **8紫红** | **17黑** | **32紫红** | **41黑** | | **9紫** | **16白** | **33紫** | **40白** | | **10橙** | **15蓝** | **34橙** | **39蓝** | | **11粉** | **14绿** | **35粉** | **38绿** | | **12灰** | **13红** | **36灰** | **37红** |   **第二帧数据：**   |  |  |  |  | | --- | --- | --- | --- | | **1黑** | **24黑** | **25黑** | **48黑** | | **2黑** | **23黑** | **26黑** | **47黑** | | **3黑** | **22黑** | **27黑** | **46黑** | | **4黑** | **21黑** | **28黑** | **45黑** | | **5黑** | **20黑** | **29黑** | **44黑** | | **6黑** | **19黑** | **30黑** | **43黑** | | **7黑** | **18黑** | **31黑** | **42黑** | | **8黑** | **17黑** | **32黑** | **41黑** | | **9黑** | **16黑** | **33黑** | **40黑** | | **11黑** | **15黑** | **34黑** | **39黑** | | **11黑** | **14黑** | **35黑** | **38黑** | | **12黑** | **13黑** | **36黑** | **37黑** |   55 05 85 01 00 00 02 0C 04 00 00 FF 00 00 FF 00 00 FF 00 00 00 FF FF FF 00 00 00 00 00 00 FF FF 00 FF FF 00 00 FF 00 FF 00 80 FF 40 00 00 30 FF 00 93 14 FF 00 10 10 10 00 00 00 FF 00 00 FF 00 00 FF 00 00 00 FF FF FF 00 00 00 00 00 00 FF FF 00 FF FF 00 00 FF 00 FF 00 80 FF 40 00 00 30 FF 00 93 14 FF 00 10 10 10 00 00 00 FF 00 00 FF 00 00 FF 00 00 00 FF FF FF 00 00 00 00 00 00 FF FF 00 FF FF 00 00 FF 00 FF 00 80 FF 40 00 00 30 FF 00 93 14 FF 00 10 10 10 00 00 00 FF 00 00 FF 00 00 FF 00 00 00 FF FF FF 00 00 00 00 00 00 FF FF 00 FF FF 00 00 FF 00 FF 00 80 FF 40 00 00 30 FF 00 93 14 FF 00 10 10 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 84 AA  **注意：1、一个块只能存入一种动态效果，如果有多种动态显示，需存入不同的块，同时记住每个块对应的动态效果，后面在执行06指令的时候，可以调用。**  **2、当第一次传入0扇区的数据超过4096后，第二次的传输不需要再带入帧数、行数、列数，直接传入颜色数据即可。**  3、灯珠数建议不要超过110颗，每种动态显示数据不能超过16,720 字节(约16.4k)。也就是在110颗的情况下，不能超过38帧。 | | | | | | | | | | |

IO->PC：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 状态 | 检验 | 帧尾 |
| 55 | 05 | 01 | 00：正常  非00：不正常 | 0-Sum | aa |
| 例如：55 05 01 00 00 FB AA | | | | | |

**x06 :读取flash动态显示数据**

PC->IO：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 端口 | 块号 | 检验 | 帧尾 |
| 55 | 06 | 2 | 0~6 | 0-255 | 0-Sum | aa |
| **端口选择：**  enum {  Port1,  Port2,  Port3,  Port4,  Port5,  Port6,  Port7,  Port\_MAX //!< MaxProt  };  块号：在0x05指令中存入的块号  注意：  **在04指令中调用**FX\_MODE\_FLIPBOOK模式时，需要先调用该指令，把显示数据调取出来。 | | | | | | |

IO->PC：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 状态 | 检验 | 帧尾 |
| 55 | 06 | 01 | 00：正常  非00：不正常 | 0-Sum | aa |
| 例如：55 06 01 00 00 Fa AA | | | | | |

**x07 :控制数据传入**

PC->IO：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 端口 | 段号 | extDataSrc | extDataCnt | 检验 | 帧尾 |
| 55 | 07 |  | 0~6 | 0-255 | 见下面解释 | 1~255 | 0-Sum | aa |
| **端口选择：**  enum {  Port1,  Port2,  Port3,  Port4,  Port5,  Port6,  Port7,  Port\_MAX //!< MaxProt  };  extDataSrc和extDataCnt在不同模式下，有不同的作用，具体如下：  1、FX\_MODE\_VU\_METER：extDataSrc实现模拟音量计效果,范围是0~255, extDataCnt是设置为多少段，例如用于显示该模式的灯珠为50颗，extDataCnt设为10，即每一段为5颗灯珠，假设现在传入的extDataSrc=100，即亮的灯珠为(extDataSrc /(255/extDataCnt))\*(50/extDataCnt)约等于20颗。  2、FX\_MODE\_BITS： extDataSrc是一个数组，把需要显示的二进制数填入, extDataCnt是extDataSrc数组的大小，例如想显示一个B11000001, extDataSrc[0]=1, extDataSrc[1]=1, extDataSrc[2]=0……extDataSrc[7]=1, extDataCnt=8.  注意：灯珠数必须是extDataCnt的2倍，例如上面的例子，要显示B11000001,需要16颗灯珠。  3、FX\_MODE\_MULTI\_COMET：extDataSrc是一个数组，填入的是流星大小(灯珠数), extDataCnt是有几个流星,即是extDataSrc数组的大小。  4、FX\_MODE\_POPCORN：  typedef struct{  float position; //位置  float velocity; //速率  uint32\_t color;//颜色  } Popcorn;  按上面的数据结构传入extDataSrc，extDataCnt是传入了多少个数据结构。  5、FX\_MODE\_OSCILLATOR：  typedef struct{  uint8\_t size;//表示振荡器的大小，即它影响的 LED 数量。  int16\_t pos;//表示振荡器在 LED 灯带上的位置。  int8\_t speed;//表示振荡器移动的速度，正值表示向右移动，负值表示向左移动。  } Oscillator;  按上面的数据结构传入extDataSrc，extDataCnt是传入了多少个数据结构。 | | | | | | | | |

IO->PC：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len |  | 检验 | 帧尾 |
| 55 | 07 | 00 |  | 0-Sum | aa |
| 例如：55 07 00 00 FB AA | | | | | |

**0x08 :** **获取软件版本号**

PC->IO：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 08 | 00 | 0-Sum | aa |
| 例如：55 08 00 ?? AA | | | | |

IO->PC：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | **版本号** | 检验 | 帧尾 |
| 55 | 08 |  |  | 0-Sum | aa |
| 例如：VER.0.0.1\_2503  **55 08 0F 56 45 52 2E 30 2E 30 2E 31 5F 32 35 30 33 00 ?? AA**  **注意：版本号要转换为字符串** | | | | | |

**0x09 :** **获取PCB版本号**

PC->IO：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | 检验 | 帧尾 |
| 55 | 09 | 00 | 0-Sum | aa |
| 例如：55 81 00 80 AA | | | | |

IO->PC：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | **版本号** | 检验 | 帧尾 |
| 55 | 09 |  |  | 0-Sum | aa |
| 例如：PCB\_Ver\_1.1  **注意：版本号要转换为字符串** | | | | | |

**0x0A :** **错误信息打印**

IO->PC：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 帧头 | Cmd | Len | **错误信息** | 检验 | 帧尾 |
| 55 | 0a |  |  | 0-Sum | aa |
| 例如：  **注意：错误信息要转换为字符串** | | | | | |