



LED 分析仪编程手册

版本号	修订人员	修订时间	修订内容
V1.1	NEWTON	2016-06-18	增加 r_xy,r_cct 等指令
V1.5	NEWTON	2018-08-18	增加 r_chroma,flick_mode 等指令
V19.3	NEWTON	2019-03-03	增加 offset,r_cd_mm 等指令
V19.6	NEWTON	2019-06-06	增加 adc_samp,flick_chroma 等指令
V19.9	NEWTON	2019-09-09	增加 Red_6,system 等指令
V20.10	NEWTON	2020-10-01	增加 sdcmm,flick_flow 等指令
V21.05	NEWTON	2021-05-01	增加 flick_edge 等指令
V21.06	NEWTON	2021-06-01	增加目录页
V22.03	NEWTON	2022-02-01	增加 net 网络通信说明
V23.06	NEWTON	2023-06-28	增加 flick_color, AutoWB 等指令
V23.091	NEWTON	2023-09-19	修改 w_target_type=1,2,3 的定义,并增加 offset 测试用例
V23.111	NEWTON	2023-12-12	修改 w_target_type 的 WBKY 定义,并统一了结束符“\r\n” 增加了 wl_whitebalance 指令
新版本兼容旧版本---当前版本 V23.111			

目录

LED 分析仪编程手册	1
新版本兼容旧版本---当前版本 V23.111	1

一，通信概述：	5
二，指令格式：	5
三， 指令详解：	5
---读取仪器 IDN 信息(idn):	5
---查询仪器状态指令(state):	5
---写入模块的地址(w_id)	5
---读取模块的地址(r_id):	5
---写入模块的波特率(w_baud):	6
---读取模块的波特率(r_baud):	6
---系统复位初始化(w_system_reset):	6
---用户参数保存参数到 flash(save_to_flash):	6
---恢复出厂默认设置(default):	6
---读取系统的 ADC 采样模式(r_system_samp):	6
---写入系统的 ADC 采样模式(w_system_samp):	7
---读取传感器的硬件编号(r_system_sens):	7
---写入每个通道的采样时间索引号(w_ft):	7
---读取每个通道的采样时间的索引号(r_ft):	7
---写入每个通道的增益值索引号(w_gain):	8
---读取每个通道的增益索引号(r_gain):	8
---读取每个通道的增益值(r_gainx) :	8
---读取可见光强度(亮度/照度/流明)数据(r_lux):	8
---写入可见光亮度(照度)补偿系数(w_k_lux):	8
---读取可见光亮度(照度)补偿系数(r_k_lux):	8
---写入非可见光功率补偿系数(w_k_uw):	9
---读取非可见光功率补偿系数(r_k_uw):	9
---读取光功率辐射强度数据(r_uw_cm):	9
---读取流明数据(r_lm):	9
---读取发光强度值(坎德拉-流明)数据(r_cd_lm):	9
---读取发光亮度值(坎德拉/平方米)(r_cd_mm):	9
---读取传感器 Raw Data ADC 数据(rgbw):	10
---读取 RGBI(U8)颜色亮度数据(r_rgbi):	10
---读取每个通道的 HSLI 数据(r_hsl):	10
---读取(CIE-1931 色坐标)xy 数据(r_xy):	11
---读取每个通道的 Yxy 数据(r_Yxy):	11
---读取(CIE-1976 色坐标)uv 数据(r_uv):	11
---读取 CCT(相关色温)数据(r_cct):	11
---读取色温黑体偏离量(r_cctd): //V1.5 版本	11
---读取主波长数据(r_doWave):	11
---读取主波长+饱和度+亮度(r_wavesi): (V1.4 版本)	11
---选择待测 LED 的颜色类型(w_target_type):	12
---读取待测 LED 的类型(r_target_type):	12
---读取 r_chroma(色度)数据: (V1.5 版本及以后)	12
---写入色容差类型(w_sdcn_type):	13

---读取色容差的类型(r_sdcn_type):	13
---读取色容差的数据(r_sdcn_data):	13
---读取 flick 阈值的参考模式(r_flick_mode):	14
---写入 flick 的颜色阈值上限(w_flick_color_max):	14
---写入 flick 的颜色阈值下限(w_flick_color_min):	14
---读取 flick 的颜色上限阈值(r_flick_color_max):	15
---读取 flick 的颜色下限阈值(r_flick_color_min):	15
---写入 flick 的亮灭阈值(w_flick_limit):	15
---读取 flick 的亮灭阈值(r_flick_limit):	15
---启动 flick 捕获功能(w_flick_ts):	15
---读取 flick 周期(r_flick_ts):	15
---启动 LED 流水灯捕获功能(w_flick_flow): //V20.123 版本	15
---启动 LED 边沿捕获功能(w_flick_edge): //V21.051 版本	16
---读取 LED 多次亮灭边沿时间戳(r_flick_edge): //V21.051 版本	17
---读取 flick 的照度数据(r_flick_lx): //V1.5 版本	17
---读取 flick 的色度数据(r_flick_chroma):	17
---读取 flick 的 rgbi 数据(r_flick_rgbi):	17
---读取 flick 的 hsli 数据(r_flick_hsli):	18
---读取 flick 的 Raw Data ADC 数据(r_flick_rgbc):	18
---读取被测 LED 的亮灭状态(r_led_chl)	18
---写入 LED 发射光源的亮灭状态(w_led_disp)	18
---读取 LED 发射光源的亮灭状态(r_led_disp)	18
---使能 Offset 组别(w_offset_en)	19
---写入 Offset 组别的补偿值 dx(w_offset_dx)	19
---写入 Offset 组别的补偿值 dy(w_offset_dy)	19
---写入 Offset 组别的亮度值系数 kl(w_offset_kl)	19
---清零 offset 所有数据(w_offset_clear)	19
---保存 offset 所有数据到 flash (w_offset_save)	19
---读取 offset 使能组别(r_offset_en)	20
---读取 offset 补偿值 dx (r_offset_dx)	20
---读取 offset 补偿值 dy(r_offset_dy)	20
---读取 offset 补偿值 kl(r_offset_kl)	20
---读取 DI 单通道输入状态(r_inbit)	20
---读取 DI1-8 通道输入状态 U8(r_in_u8)	20
---写入 DO 多个通道输出状态(w_outbit)	21
---写入 net 的工作模式(w_net_mode)	21
---读取 net 的工作模式(r_net_mode)	21
---写入 net 的仪器的源 IP 地址(w_net_sip)	21
---读取 net 的仪器的源 IP 地址(r_net_sip)	21
---写入 net 的上位机的目标 IP 地址(w_net_dip)	22
---读取 net 的上位机的目标 IP 地址(r_net_dip)	22
---写入 net 的网关 Gateway IP 地址(w_net_gip)	22
---读取 net 的网关 Gateway IP 地址(r_net_gip)	22

---写入 net 的子网掩码 Subnet Mask IP 地址(w_net_mip)	22
---读取 net 的子网掩码 Subnet Mask IP 地址(r_net_mip)	22
---写入 net 的 LED 仪器的源端口号(w_net_sport)	22
---读取 nnet 的 LED 仪器的源端口号(r_net_sport)	22
---写入 net 的上位机服务器的目标端口号(w_net_dport)	22
---读取 net 的上位机服务器的目标端口号(r_net_dport)	23
---读取 net 的 MAC 地址(r_net_mac)	23
---读取 net 的所有配置参数(r_net_all)	23
---试运行 net 网口通信连接(w_net_run)	23
---写入某几个通道的 Red 的白平衡系数(wr_whitebalance)	23
---写入某几个通道的 Green 的白平衡系数(wg_whitebalance)	23
---写入某几个通道的 Blue 的白平衡系数(wb_whitebalance)	23
---写入某几个通道的 WBxx 的 Lux 亮度系数(wl_whitebalance)	24
---读取某几个通道的 WBxx 的 Lux 亮度系数(rl_whitebalance)	24
---读取某几个通道的 Red 的白平衡系数(rr_whitebalance)	24
---读取某几个通道的 Green 的白平衡系数(rg_whitebalance)	24
---读取某几个通道的 Blue 的白平衡系数(rb_whitebalance)	24
---保存白平衡系数(save_whitebalance)	24
---写入 Sensor 的原始数据的 AutoWB 系数(w_autowb_sens)	25
---读取 Sensor 的原始数据的 AutoWB 系数(w_autowb_sens)	25
四：指令表汇总：	26
五：软件开发流程图举例	29
---单色常亮 LED 数据采集流程	29
---单色闪烁 Flick-LED 数据采集流程	30
六：测试用例-指令读写实例	31
---读取 CH1 的黄绿灯的主波长(ID=1):	31
---读取 CH1 的白光灯的 xy 色坐标(ID=1):	31
---读取 CH1 的 RED-LED-flick 闪烁频率(ID=1):	31
---offset 并读取 4CH 的双色灯(ID=1):	32
---1 转 3 光纤如何 offset 并测量双色灯	33
七：编程注意事项-必看！	36
1, 配置参数保存 save_to_flash	36
2, 增益 gain 和采样时间 ft	36
3, 采样模式 w_system_samp:	36
4, LED 光源类型选择 w_target_type	36
5, 频率流水灯捕获指令 flick/flow:	36
6, r_rgb 与 r_chroma 颜色识别指令:	36
7, Net 通信说明:	36
8, 关于 PWM 驱动高亮 LED 测量说明:	37
9, 单个 CHL 测多个 LED(多色 LED)时如何补偿光学参数:	37
八：手册声明:	37

一，通信概述：

- 1,USB,RS485 两种接口,都遵循串口硬件协议, 8b-byte, 1b-stop, none 校验;
- 2,USB,RS485 两种接口通信协议一样, 字符串通信;
- 3,网口 net 通信, 底层是 TCP/IP 协议, 但顶层通信指令和串口指令完全一样;

二，指令格式：

0,假设当前仪器的地址为“001”，且系统中只有一个该仪器：

- 1,主机发送指令格式: “:001*****\r\n” --- (“:”+“ID”+“功能指令”+“\r\n”);
- 2,仪器返回指令格式: “:001#####\r\n” --- (“:”+“ID”+“返回信息”+“\r\n”);
- 3,其中“:”为起始字符, “\r\n”为结束字符(串口通信时至少要有个结束符'\n'),
ID 为 3 位整数(%03d),字符串内无空格;
- 4,所有指令全为英文字符,无 CRC 校验;
- 5,当发送错误命令时, 仪器统一返回":001ERR_CMD\r\n"或者无任何字符返回;
- 6,ID="000"是广播地址, 无论仪器自身 ID 是多少, 只要是合法指令, 都会应答;
- 7,RS485 通信时, 发送和读取需预留总线转换时间(>3ms);
- 8,数据读写指令全部支持任意单通道和连续多通道操作, 通道号必须递增格式;

三，指令详解：

---读取仪器 IDN 信息(idn):

询问模块的型号信息;

发送指令格式: “:001idn\r\n”

返回指令格式: “:001#####\r\n”

指令解释: 返回的指令以实际字符为准,

(具体根据模块反馈信息而定,该信息内容没有特定的格式规律至少有“HanOpticSens”关键字)

---查询仪器状态指令(state):

询问模块的工作状态是 idle 还是 busy;

发送指令格式: “:001state\r\n”

返回指令格式: “:001idle\r\n”或者为“:001busy\r\n”;

指令解释: 当模块 busy 时, 模块不会执行接受到的其他新指令;

---写入模块的地址(w_id)

发送指令格式: “:000w_id=002\r\n”或“:001w_id=002\r\n”(该参数立即生效, 自动保存在 flash)

返回指令格式: “:002w_id=002\r\n”;

指令解释:

写入 ID 成功后, 返回指令立即采用新的 ID 返回信息;

该指令会擦写 flash, 寿命小于 10 万次, 不要在主程序内频繁操作;

---读取模块的地址(r_id):

发送指令格式: “:000r_id\r\n”或“:001r_id\r\n”

返回指令格式: “:001r_id=001\r\n”;

指令解释：模块返回自己当前的 ID="001";

---写入模块的波特率(w_baud):

发送指令格式: ":001w_baud1=6\r\n"或 ":001w_baud2=6\r\n"(该参数立即生效, 自动保存在 flash, 掉电不丢失)

返回指令格式: ":001w_baud1=6\r\n"或 ":001w_baud2=6\r\n"

指令解释:

模块先按照原来的 baud 返回指令, 再配置自己的 baud;

"="后面的值是波特率编号(0-9),默认值是 6;

baud[10]={2400,4800,9600,19200,38400,57600,115200,230400,460800,921600};

RS485 接口的波特率最大是 460800;

在 LBB 多通道系列仪器中, baud1 代表 USB-RS232 接口, baud2 代表 RS485 接口;

在 PROBE 单通道系列仪器中, 只有 baud1, 表征 USB-RS232 或者 RS485 接口;

该指令会擦写 flash, 寿命小于 10 万次, 不要在主程序内频繁操作;

---读取模块的波特率(r_baud):

发送指令格式: ":001r_baud1\r\n"或 ":001r_baud2\r\n"

返回指令格式: ":001r_baud1=6\r\n"或 ":001r_baud2=6\r\n"

指令解释: baud1 代表 USB-RS232, baud2 代表 RS485(RS232), 返回接口波特率编号(0-9);

---系统复位初始化(w_system_reset):

发送指令格式: ":001w_system_reset\r\n"

返回指令格式: ":001w_system_reset\r\n"

指令解释:

让产品在上电状态下从新复位, 并不是恢复出厂设置;

在外置 TrueS 探头系列产品中可以在探头热插拔后, 从新恢复正常工作状态;

---用户参数保存参数到 flash(save_to_flash):

发送指令格式: ":001save_to_flash\r\n"(该指令立即生效-不建议频繁执行)

返回指令格式: ":001save_to_flash\r\n";

指令解释:

把 gain,ft,target_type,system_samp,flick,net 等用户可以修改的参数保存到 flash, 掉电不丢失;

该指令会擦写 flash, 寿命小于 10 万次, 不要在主程序内频繁操作;

---恢复出厂默认设置(default):

发送指令格式: ":001default\r\n" (该指令立即生效-不建议频繁执行)

返回指令格式: ":001default\r\n";

指令解释:

把 id,baud,gain,ft,target_type,system_samp,flick,net 等参数全部恢复出厂默认, 并保存到 flash,

掉电不丢失; 该指令会擦写 flash, 寿命小于 10 万次, 不要在主程序内频繁操作;

---读取系统的 ADC 采样模式(r_system_samp):

发送指令格式: ":001r_system_samp\r\n"

返回指令格式: ":001r_system_samp=0\r\n"

指令解释:

"="后面的值是 adc_samp 的工作模式, 0:连续采样模式 continuous(默认) ,

1:单次采样模式 single sample;

---写入系统的 ADC 采样模式(w_system_samp):

发送指令格式: “:001w_system_samp=1\r\n”

返回指令格式: “:001w_system_samp=1\r\n”

指令解释: (V19.6-该参数立即生效, 暂存于 RAM 中, 也可以保存, 掉电不丢失)

“=”后面值是 adc_samp 工作模式, 0:连续采样模式 continuous, 1:单次采样模式 single sample;

0: 所有通道都处于同步连续采样模式, 数据更新周期等于采样时间(FT);

但不管 FT 是多大, 当模块接收到串口指令后, 会立即返回数据, 但返回的是上一个采样周期的数据, 所以读取时要注意: 必须让 LED 点亮时间大于 2 倍的 FT, 再去读取数据, 才能保证读取到的是 LED 点亮稳定后的数据; 这个指令的优点在于串口返回数据不等待, 可节约上位机等待时间; 对于 LED 常亮型或者对测试时间要求不高的话, 建议采用该模式;

1: 发送一次读取指令, 传感器立即从新采样一次数据, 模块串口需等待 FT 时间后才能返回采集到的数据; 虽然串口等待, 但可以测量到短暂的脉冲光源: 点亮 LED 后, 立即发送一次读取, 只要 LED 点亮时间比 FT 时间长, 那么读取到的就是 LED 点亮的稳定数据;

一般仪器的出厂默认配置是 0:连续采样模式 continuous(默认);

---读取传感器的硬件编号(r_system_sens):

发送指令格式: “:001r_system_sens01-02\r\n” (V23.031 及以后的版本才支持)

返回指令格式: “:001r_system_sens=2,2\r\n”

指令解释:

“01-02”就是需要读取的通道号, “=”后面的值是 sensor 的编号 2, 大于 0 就是对应的 sensor 编号, 不同编号的 sensor, 光学原始数据会有一定的偏差;

如果数据=0:表示无传感器, 或者该通道的传感器损坏;

通道读写指令全部支持任一通道和连续多通道操作, 格式必须严格执行:

“01-02”代表从 01(%02d)通道到 02(%02d)通道, 必须是从小到大, 或者是相等(idn 仪器信息中只有带“HF40”关键字的仪器才支持最大通道为 40, 其他仪器最大通道均为 20, 不能发送超过该仪器支持的最大通道数, 一旦超过, 仪器需要从新上电才能复位);

---写入每个通道的采样时间索引号(w_ft):

发送指令格式: “:001w_ft01-20=1\r\n”(该参数立即生效, 暂存于 RAM 中)

返回指令格式: “:001w_ft01-20=1\r\n”

指令解释:

“=”后面的值是 ft 编号(0-15), 编号越大, ft 越大, 采样速度越慢, 相同光照下, 读取到的原始 ADC 数据 Rawdata 越大(读取 rawdata 数据指令是 rgbw); 由于不同型号的仪器, 支持的 ft 范围不一样, 因此当配置 ft 大于实际支持的采样时间时, 就等于仪器能支持的最大采样时间; 在频闪流水测试时, 根据实际速度大小选择合适的 ft;

---读取每个通道的采样时间的索引号(r_ft):

发送指令格式: “:001r_ft03-06\r\n”

返回指令格式: “:001r_ft=1,1,1,1,\r\n”

指令解释:

“=”后面的值依次是 03-06 通道的 ft 索引号;

---读取每个通道的真实采样时间(r_ftms):

发送指令格式: “:001r_ftms01-02\r\n”

返回指令格式: “:001r_ftms=20,123,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道 ft 真实 ms 时间, 该值与写入的 ft 索引号对应, 实际数据以仪器返回的数据为准, 对频闪测试有参考意义;

---写入每个通道的增益值索引号(w_gain):

发送指令格式: “:001w_gain01-20=1\r\n”(该参数立即生效, 暂存于 RAM 中)

返回指令格式: “:001w_gain01-20=1\r\n”

指令解释:

“=”后面的值是 gain 编号(0-15), 编号越大, gain 越大, 相同光照下, 读取到的原始 ADC 数据 Rawdata 越大(读取 rawdata 数据指令是 rgbw); 由于不同型号的仪器, 支持的 gain 范围不一样, 因此当配置 gain 大于实际支持的增益时, 就等于仪器能支持的最大增益;

修改该参数, 是为了让传感器工作在最佳量程范围, 或者提高灵敏度, 提高分辨率;

一般情况下, 尽量固定用较大的 ft 采样时间, 根据光强的大小修改 gain 来改善传感器的硬件工作状态, 处于最合理的范围内, 即 ADC 值不要太小, 也不要太大;

---读取每个通道的增益索引号(r_gain):

发送指令格式: “:001r_gain03-06\r\n”

返回指令格式: “:001r_gain=1,1,1,1,\r\n”

指令解释:

“=”后面的值依次是 03-06 通道的增益编号;

---读取每个通道的增益值(r_gainx) :

发送指令格式: “:001r_gainx01-02\r\n”

返回指令格式: “:001r_gainx=5,1,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的增益值, 该值与写入的 gain 索引号对应, 实际数据以仪器返回的数据为准;

//////////光强数据-//////////

---读取可见光强度(亮度/照度/流明)数据(r_lux):

发送指令格式: “:001r_lux01-02\r\n”

返回指令格式: “:001r_lux=123.12,234.12,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的照度值, 数据格式是浮点字符串%f,

在测量照度的产品中, 该值单位是 lx; 在流明系列仪器中, 该值单位是 lm; 在光纤系列产品中, 该值代表正比于 LED 发光强度的一个参考值, 需要二次标定才能得到真实的照度/亮度/流明值;

---写入可见光亮度(照度)补偿系数(w_k_lux):

发送指令格式: “:001w_k_lux01-08=1.001\r\n”

返回指令格式: “:001w_k_lux=1.001\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的补偿系数, 精确到 0.001, 内部相乘关系, 出厂默认是 1.0, 可以用在亮度一致性补偿, 也可以用作二次标定亮度相关的真实单位;

---读取可见光亮度(照度)补偿系数(r_k_lux):

发送指令格式: “:001r_k_lux01-02\r\n”

返回指令格式: “:001r_k_lux=1.001,1.001\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的补偿系数, 精确到 0.001;

---写入非可见光功率补偿系数(w_k_uw):

发送指令格式: “:001w_k_uw01-08=1.001\r\n”

返回指令格式: “:001w_k_uw=1.001\r\n”

指令解释:

“=”后面的值依次是 01-08 通道的补偿系数, 精确到 0.001, 内部是相乘关系, 出厂默认是 1.0, 主要用于非可见光 IR/UV 强度测量; 可以用在光功率一致性补偿, 也可以用作二次标定光功率相关的真实单位;

---读取非可见光功率补偿系数(r_k_uw):

发送指令格式: “:001r_k_uw01-02\r\n”

返回指令格式: “:001r_k_uw=1.001,1.001\r\n”

指令解释:

“=”后面的值依次是 01-08 通道的补偿系数, 精确到 0.001;

---读取光功率辐射强度数据(r_uw_cm):

发送指令格式: “:001r_uw_cm01-02\r\n”

返回指令格式: “:001r_uw_cm=123.1,234.2,\r\n”

指令解释:

“=”后面值依次是 01-02 通道的光功率, 该数据只用于非可见光功率判断;

在测量辐射照度的非光纤产品中, 该值单位是 uw/cm^2 ; 在光纤系列产品中, 该值代表正比于 LED 发光强度的一个参考值, 需要二次标定才能得到真实的光功率值;

---读取流明数据(r_lm):

发送指令格式: “:001r_lm01-02=1.0\r\n”

返回指令格式: “:001r_lm=1.123,0.123,\r\n”

指令解释: //选用积分球探头测量, 该指令读取的数据比较准确

“=”后面的值依次是 01-02 通道的流明 $\text{LM}(\%0.3f), (\%0.3f),$;

“1.0(%)”是补偿系数 k(相乘关系), 该指令是可以实时配置的流明系数, 内部关系是 $r_{lm} = r_{lux} * k$;

---读取发光强度值(坎德拉-流明)数据(r_cd_lm):

发送指令格式: “:001r_cd_lm01-01=00=50.5\r\n”//只适用于标准朗伯体光源

返回指令格式: “:001r_cd_lm=1.123,2.123,\r\n”

指令解释:

“00(%02d)”是探头相对于 LED 法线的平面夹角 $\theta(0-89^\circ\text{C})$,

“50.5(%0.1f)”是探头距离 LED 的直线距离 d(单位:mm);

“=”后面的值依次是 01-01 通道的探测角度的坎德拉, 以及 LED 的最大流明 $(\%0.3f, \%0.3f, \text{cd}, \text{lm})$;

为了保证测量数据准确性, 必须选用 Trues 和 PROBE 系列的色温照度探头,

测量时尽量保证 LED 向空间无遮拦, 无反射, 无聚散透镜, 自由发散, 尽量在 LED 法线方向上测量, 即夹角等于 0; 探头尽量远离光源, $d > 20$ 倍光源直径; 该指令不适合光纤系列仪器;

---读取发光亮度值(坎德拉/平方米)(r_cd_mm):

发送指令格式: “:001r_cd_mm01-02\r\n”

返回指令格式: “:001r_cd_mm=123,125,\r\n”

指令解释: // 只适用于 V19.3 之后版本

“=”后面的值依次是 01-02 通道的亮度值, 单位是“坎德拉/平方米”(cd/m²),

为了保证测量数据绝对值准确性, 面光源尽可能大, 如果光源面积小于探头感光面积, 读取到的亮度值小于实际值; 在专门测量亮度数据的探头系列中, 该数据等于 r_lux, 改变 w_k_lux 依然可以改变 r_cd_mm 该指令适合非光纤类产品;

//////////////////////////////////HSI/RGB 颜色数据-适合 HSI(RGB)系列产品//////////////////////////////////

---读取传感器 Raw Data ADC 数据(rgbw):

发送指令格式: “:001rgbw01-02\r\n”

返回指令格式: “:001rgbw=123,234,345,678,11123,11234,11345,22678,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“r(%d),g(%d),b(%d),w(%d),”, 属于传感器原始 ADC 数据,rgbw 的比例代表颜色, 绝对值代表相对强度; 实际编程时, 不建议用该数据作为判断标准, 但该数据可以反应传感器工作状态是否合理, 数据太小, 接近传感器灵敏度下限, 导致不准确, 过大可能会造成数据饱和, 数据失真;

距离数据饱和比例处在 1%-60%是最佳工作区间; 修改 gain 或 ft 参数, 会更改 Raw Data;

---读取 RGBI(U8)颜色亮度数据(r_rgb):

发送指令格式: “:001r_rgb01-01\r\n”

返回指令格式: “:001r_rgb=255,244,105,50.00\r\n”

“=”后面的值依次是 01-01 通道的“r(%d),g(%d),b(%d),i(%0.2f),”; 该 RGB 范围为 0-255;

I 是相对于最大测量强度的百分比(0.00-100.00), I 可以作为相对光强的数据,

但同一光强下, 修改 Gain 参数, 会改变 I 的值; I 接近 100.00%时, 说明传感器已饱和;

该 RGBI 仅是颜色亮度的相对分量值, 没有标准可以参考, 只能一致性相对比较;

该 RGB 与相机的 RGB 和显示器图片 RGB 数据原理还是存在一定的差异, 不能完全对等;

---读取每个通道的 HSLI 数据(r_hsl):

发送指令格式: “:001r_hsl01-02\r\n”

返回指令格式: “:001r_hsl=300,80,40,10.01,300,80,40,10.01,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“H(%0.0f),S(%0.0f),L(%0.0f),I(%0.2f),”;

H(0-360), S(0%-100%), L(0%-100%), I(0.00%-100.00%) ,

I 是相对于最大测量强度的百分比, I 接近 100%时, 说明传感器已饱和与 r_rgb 中的 i 相同;

该 HSLI 仅是颜色亮度的相对分量值, 没有标准可以参考, 只能一致性相对比较, 其中 L 与 HSV 种的 V 相同;

该 HSL 与相机的 HSL 和显示器图片 HSL 数据原理还是存在一定的差异, 不能完全对等;

////////CIE-Chroma 色度数据//适合 XYZ/CIE/TrueS/Probe 系列产品////////

---读取(CIE-1931 色坐标)xy 数据(r_xy):

发送指令格式: “:001r_xy01-02\r\n”

返回指令格式: “:001r_xy=0.3333,0.4333,0.3666,0.3111,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“x(0.4F%),y(0.4F%),”, 该 xy 代表 CIE-1931 色度图坐标;

---读取每个通道的 Yxy 数据(r_Yxy):

发送指令格式: “:001r_Yxy01-02\r\n”

返回指令格式: “:001r_Yxy=323.5,0.2345,0.3145,678.5,0.5234,0.1434,\r\n”

指令解释: “01-02 代表从 01(%02d)通道到 02(%02d)通道, 必须是从小到大, 或者是相等(通道最大值为 8(16)); “=”后面的值依次是 01-02 通道的“Yxy(f%),(f%),(f%),”, Y 等于照度值 r_lux, xy 是色坐标; 该指令中的第一个‘Y’是大写‘Y’;

---读取(CIE-1976 色坐标)uv 数据(r_uv):

发送指令格式: “:001r_uv01-02\r\n”

返回指令格式: “:001r_uv=0.3333,0.4333,0.6666,0.1111,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“u(0.4F%),v(0.4F%),”, 该 uv 代表 CIE-1976 色度图坐标;

---读取 CCT(相关色温)数据(r_cct):

发送指令格式: “:001r_cct01-02\r\n”

返回指令格式: “:001r_cct=5438,6457,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“CCT(d%),(d%),”, 该 CCT 数据根据 CIE-1931(xy)色度图坐标计算得出;

---读取色温黑体偏离量(r_cctd): //V1.5 版本

发送指令格式: “:001r_cctd01-01\r\n”

返回指令格式: “:001r_cctd=5438,0.00601,\r\n”

指令解释:

“=”后面的值依次是 01-01 通道的“CCT(0%0.0f),duv(%0.6f),”, 该数据根据 CIE-1976(uv)色度图坐标计算得出;

---读取主波长数据(r_doWave):

发送指令格式: “:001r_dowave01-02\r\n”

返回指令格式: “:001r_dowave=438.5,617.5,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“(%0.1f),(%0.1f),”;

---读取主波长+饱和度+亮度(r_wavesi): (V1.4 版本)

发送指令格式: “:001r_wavesi01-01\r\n”

返回指令格式: “:001r_wavesi=555.5,99.9,123.4,\r\n”

指令解释:

“=”后面的值依次是 01-01 通道的“(%0.1f),(%0.1f),”, 555.5 代表主波长, 单位 nm, 99.9%代表饱和度, 色纯度, 范围 0%-100%; 123.4 是与指令 r_lux 读取数据相同;

---选择待测 LED 的颜色类型(w_target_type):

发送指令格式: “:001w_target_type01-02=0\r\n”(立即生效,暂存于 RAM 中)

返回指令格式: “:001w_target_type01-02=0\r\n”

指令解释:

“=”后面的值是类型编号(0-30),默认值是 0,任何模式都可以读取出来色度值,但为了获取更准确的光度参数,需要选择合适的模式;只有 XYZ 及以上系列产品支持该指令;

0: 白平衡模式 WB0,任何光源都可以采用该模式,但不一定是最优配置,比较适合白光;

3: 绿灯(500nm-560nm)或白色 LED(2000k-5000K),具体根据仪器型号来定义;

4: 单芯灯珠发“橙色(600nm 左右)”或“黄绿(567nm)”或“青色(500nm 左右)”,
以及任意其中两芯或三芯混合而成的灯光,比如网口橙/黄绿指示灯;

5: RGB 灯:Red(630nm 左右),Green(525nm 左右),Blue(461nm 左右),
单色以及任意其中两芯或三芯混合而成的灯光;比如 RGB 氛围灯;

6: Red 单色灯,主波长在(600-641nm)之间的红灯;

////编号 1,2 以及大于 9 的配置是 2023-5-1 以后的版本才支持的,晚于 V23.033 版本/////

1: 白平衡模式 WB1,比较适合低色温暖白光;

2: 白平衡模式 WB2,比较适合高色温暖白光;

10: AutoWB 模式,是自动白平衡模式,仪器会自动匹配 LED 的颜色,选择最合理的白平衡系数参与计算,匹配的白平衡系数仅限于编号为 11~16 的范围;

11: RedWB 模式,出厂默认是专门测量红光的模式;

12: GreenWB 模式,出厂默认是专门测量绿光的模式;

13: BlueWB 模式,出厂默认是专门测量蓝光的模式;

14: YellowWB 模式,出厂默认是专门测量黄光的模式;

15: C/R_WB 模式,出厂默认是专门测量青灯(Cyan),某些仪器是超级红灯(Red)的模式;

16: WhiteWB 模式,出厂默认是专门测量白光的模式,白平衡系数等于编号 0 的系数;

备注: 在一些特殊光源测量时,会占用其中某些模式,以配套产品手册为准;

该指令在 V23.101 版本之前,只能改变色度数据,但在 V23.101 版本及以后,可以同时改变亮度和色度,特别适合采用官方调试软件进行色度亮度的 WB 校准操作;

---读取待测 LED 的类型(r_target_type):

发送指令格式: “:001r_target_type01-02\r\n”

返回指令格式: “:001r_target_type=0,0\r\n”

指令解释: “=”后面的值是 LED 类型配置的编号;

---读取 r_chroma(色度)数据: (V1.5 版本及以后)

发送指令格式: “:001r_chroma01-01\r\n”

返回指令格式: “:001r_chroma=1000.0,0.3333,0.4444,555.5,85.2,6500,0.00123,\r\n”

指令解释: “=”后面的值依次是 01-01 通

道”lux(%0.1f),x(%0.4f),y(%0.4f),dowave(%0.1f),duty(%0.1f),cct(%0.0f),fd(%0.5f),”

该指令可以高效获取 LED 的色度数据;

lux=1000.0,与 r_lux 指令相同;

x=0.3333,y=0.4444,与 r_xy 相同,是 CIE1931 的 xy 坐标;

dowave=555.5nm,duty=85.2%,与 r_wavesi 功能相似;

cct=6500K,与 r_cct 指令类似;

fd=0.00123,该浮点数是预留参数,不同产品该值不一样,V24.011 之后默认是 r_rgbi 中的 i,i

是传感器饱和程度,但启用 SDCM 模式测量时, fd 是 SDCM 数据(支持 V20.101 及以后);

---写入色容差类型(w_sdcn_type):

发送指令格式: “:001w_sdcn_type01-02=0\r\n”

返回指令格式: “:001w_sdcn_type01-02=0\r\n”

指令解释: (立即生效, 暂存于 RAM 中, 只适用 XYZ 及以上版本 V20.101)

“=”后面的值是类型编号(0-99),默认值是 0,disable,

类型编号	功能描述	备注
0	Disabale, 不启用 SDCM 测量功能	
1	预留, 暂时不用	后续可能会添加成全自动匹配模式
2	AUTO-SDCM_ErpF	
3	AUTO-SDCM_ANSI	
4	AUTO-SDCM_customized	可根据客户的要求定制
5,6,7,8,9	预留	
10-19	SDCM_ErpF (0,1,2,3,4,5,6,7,8,9,)	
20-29	SDCM_ANSI (0,1,2,3,4,5,6,7,8,9,)	
30-39	SDCM_customized (0,1,2,3,4,5,6,7,8,9,)	

---读取色容差的类型(r_sdcn_type):

发送指令格式: “:001r_sdcn_type01-02\r\n”

返回指令格式: “:001r_sdcn_type=0,0\r\n”

指令解释:

“=”后面的值是 sdcn 类型配置的编号;

---读取色容差的数据(r_sdcn_data):

发送指令格式: “:001r_sdcn_data01-02\r\n”

返回指令格式: “:001r_sdcn_data=3.2,2.3\r\n”

指令解释:

“=”后面的值是 01-02 通道的 sdcn 数据, 精确度 0.1f%;

---读取色容差+光照度的数据(r_sdcn_lux):

发送指令格式: “:001r_sdcn_lux01-01\r\n”

返回指令格式: “:001r_sdcn_lux=100.1,3.2,21\r\n”

指令解释:

“=”后面是 01 通道的数据, 100.1 是 lux 照度数据, 3.2 是 SDCM 值, 21 是 SDCM 的光源对标类型 ANSI 3000K;

//////////flick 指令通用，但不同型号仪器的最小采样速度不同//////////

---写入 flick 阈值的参考模式(w_flick_mode):

发送指令格式: “:001w_flick_mode01-08=0\r\n”(该参数立即生效，暂存在于 RAM 中)

返回指令格式: “:001w_flick_mode01-08=0\r\n”;

指令解释:

Mode	参数含义	备注
0(默认)	Lux	参考 r_lux 指令,只有 lux 与 flick_limit 亮灭对比捕获频闪
1	R_adc	参考指令 rgbw, 是传感器的原始 ADC 数据,采用该指令时为了提高 Flick 捕获速度,每个通道相比 mode0 可节省约 0.2ms 的计算时间; 就是采用其中一个 ADC 值与 flick_limit 亮灭对比捕获频闪;
2	G_adc	
3	B_adc	
4	W_adc	
5~9	预留	
10	Hue+Lux	参考 r_hsl 中的色调 H, 在颜色比对时, 相比 mode12,13 可节省 0.1ms 计算时间
11	Satura+Lux	参考 r_hsl 中的色调 S, 在颜色比对时, 相比 mode12,13 可节省 0.1ms 计算时间
12	Dowave_Lux	参考 r_chroma 中的主波长 wave, 即将 wave 作为颜色比对的参考值与 flick_color_min,flick_color_max 进行通过测试, 进而捕获频率
13	CCT+Lux	参考 r_chroma 中的色温 CCT, 即将 CCT 作为颜色比对的参考值与 flick_color_min,flick_color_max 进行通过测试, 进而捕获频率
颜色参与比对时, 相比 mode0, 每个通道速度会增加 0.2ms, 适用于单颗多色灯交替闪烁的品牌捕获编号大于 4 的配置是 2023-5-1 以后的版本才支持的, 晚于 V23.033 版本//////////		

---读取 flick 阈值的参考模式(r_flick_mode):

发送指令格式: “:001r_flick_mode01-02\r\n”

返回指令格式: “:001r_flick_mode=0,0\r\n”;

指令解释:

“=”后面的模式索引范围是 0-20(%d);

---写入 flick 的颜色阈值上限(w_flick_color_max):

发送指令格式: “:001w_flick_color_max01-08=1\r\n”(该参数立即生效，暂存在于 RAM 中)

返回指令格式: “:001w_flick_color_max01-08=1\r\n”; (V23.033 以后版本)

指令解释: “=”后面的阈值是浮点(%f),如果是 dowave 模式, 支持正负值, 默认值是 360;

当采集到的选定数据(见 flick_mode)大于 w_flick_color_min 时, 且小于 w_flick_color_max, 该颜色的 LED 灯点亮, 否则该颜色的 LED 属于熄灭状态, 该值是判断频闪测试的颜色阈值, 可独立配置每个通道;

---写入 flick 的颜色阈值下限(w_flick_color_min):

发送指令格式: “:001w_flick_color_min01-08=1\r\n”(该参数立即生效，暂存在于 RAM 中)

返回指令格式: “:001w_flick_color_min01-08=1\r\n”; (V23.033 以后版本)

指令解释: “=”后面的阈值是浮点(%f),如果是 dowave 模式, 支持正负值, 默认值是 0;

当采集到的选定数据(见 flick_mode)大于 w_flick_color_min 时, 且小于 w_flick_color_max, 该颜色的 LED 灯点亮, 否则该颜色的 LED 属于熄灭状态, 该值是判断频闪测试的颜色阈值, 可

独立配置每个通道;

---读取 flick 的颜色上限阈值(r_flick_color_max):

发送指令格式: ":001r_flick_color_max01-02\r\n"(V23.033 以后版本)

返回指令格式: ":001r_flick_color_max=360,360,\r\n";

指令解释: "="后面是返回的阈值,支持浮点,如果是 dowave 模式,支持正负值;

---读取 flick 的颜色下限阈值(r_flick_color_min):

发送指令格式: ":001r_flick_color_min01-02\r\n"(V23.033 以后版本)

返回指令格式: ":001r_flick_color_min=0,0,\r\n";

指令解释: "="后面是返回的阈值,支持浮点,如果是 dowave 模式,支持正负值;

---写入 flick 的亮灭阈值(w_flick_limit):

发送指令格式: ":001w_flick_limit01-08=10\r\n"(该参数立即生效,暂存在于 RAM 中)

返回指令格式: ":001w_flick_limit01-08=10\r\n";

指令解释: "="后面的阈值范围是 0-1000000(%d),必须大于 0,默认值是 20;

当采集到的选定数据(见 flick_mode)小于该值时,认为是灯灭,大于该阈值,灯点亮,该值是判断频闪测试的亮灭阈值,可独立配置每个通道;

---读取 flick 的亮灭阈值(r_flick_limit):

发送指令格式: ":001r_flick_limit01-02\r\n"

返回指令格式: ":001r_flick_limit=20,20,\r\n";

指令解释: "="后面是返回的阈值(0-1000000:%d);

---启动 flick 捕获功能(w_flick_ts):

发送指令格式: ":001w_flick_ts01-02=09\r\n"

返回指令格式: ":001w_flick_ts01-02=09\r\n";

指令解释: "09"的取值范围是 01-50 秒,设置采样时间长度,单位是秒;

须提前设置好相应通道的硬件参数(ft&gain&w_flick_limit),该指令下达后,模块立即执行频闪测试,测试时间持续 9s(设定值);测试期间,模块不响应其他指令,但可以发送 state 指令"查询模块状态,待模块返回"idle",才可发送新的读写指令;模块支持所有通道同时进行频闪测试;测量结果会一直保持在 RAM 中,直到下次指令下达从新转换更新;

---读取 flick 周期(r_flick_ts):

发送指令格式: ":001r_flick_ts01-02\r\n"(数据一直保持,直到下次测试更新)

指令解释: 读取 01-02 通道的频闪测试结果;

返回指令格式: ":001r_flick_ts=1.00,990,1000,500,5, 2.00,490,501,100,8\r\n";

指令解释:

"="后面值分别代表(频率 1.00Hz,亮-亮间隔 990ms,灭到灭间隔 1000ms,持续点亮时间 500ms,亮脉冲次数 5)

Fre1(Hz)"%0.2f",Tup-up1(ms)%d ",Tdw-dw1(ms)"%d ",Tduty1(ms)"%d",Cnt1_Puls"%d",

Fre2(Hz)"%0.2f",Tup-up2(ms)%d ",Tdw-dw2(ms)"%d ",Tduty2(ms)"%d",Cnt1_Puls"%d",

//////////flow,edge-指令通用,但不同型号仪器的最小采样速度不同//////////

---启动 LED 流水灯捕获功能(w_flick_flow): //V20.123 版本

发送指令格式: ":001w_flick_flow01-16=03\r\n" //只适合顺序点亮 LED 流水检测,例如汽车流水转向灯

指令解释: 同时启动 1-16chl 流水捕获功能,捕获时间是 3 秒;

返回指令格式: “:001w_flick_flow01-16=03\r\n”;

指令解释: “=03”的取值范围是 01-55 秒, 设置捕获时间长度, 单位是秒;

须提前设置好相应通道的硬件参数(ft&gain&w_flick_limit), 该指令下达后, 模块立即执行 flow 测试, 测试时间持续 3s(设定值); 测试期间, 模块不响应其他指令, 但可以发送 state 指令”查询模块状态, 待模块返回”idle”, 才可发送新的读写指令; 模块支持所有通道同时进行 flow 测试; 测量结果会一直保持在 RAM 中, 直到下次指令下达从新转换更新; 该指令收到后, 会先立即返回”:001w_flick_flow01-16=03\r\n”, 再进行 flow 测量;

在多个模块 RS485 并行测量多个流水灯时, 可以发送把地址改为 0, 例如发

送”:000w_flick_flow01-16\r\n”指令, 同时启动多个仪器捕获, 由于是 485 并联, 仪器会同时响应广播地址 0, 只有 ID=1 的仪器会正常返回值”:000w_flick_flow01-16\r\n”, 其他 ID 仪器不回复任何指令, 以免造成总线冲突;

---读取 LED 流水灯亮熄灭时间戳(r_flick_flow): //V20.123 版本

发送指令格式: “:001r_flick_flow01-02\r\n”(数据一直保持, 直到下次测试更新)

指令解释: 读取 01-02 通道的 flow 测试结果;

返回指令格式: “:001r_flick_flow=1000,1500,500,1050,1450,400,\r\n”;

指令解释:

“=”后面值是 ms, 全部整数, 返回数据个数 CNT=CHL*3;

1000: 代表 1CH 第一次被点亮的时间,

1500: 1CH 点亮后第一次熄灭的时间,

500: 是预留的参数, 暂定是前后时间差 1500-1000=500;

1050: 2CH 第一次被点亮的时间,

1450: 2CH 点亮后第一次熄灭的时间,

400: 是预留的参数, 暂定是前后时间差 1450-1050=400;

从数据上来看, 第 2CH 的点亮时间比第 1CH 晚了 1050-1000=50ms, 但第 2CH 的熄灭亮时间比第 1CH 早了 50ms(1450-1500 = -50ms); 说明流水灯是从 1 到 2 顺序点亮, 然后再反序熄灭; 测量的时间参考 0 点, 是发送 w_flick_flow 启动指令后, 仪器返回 w_flick_flow 后, 计为 0ms;

---启动 LED 边沿捕获功能(w_flick_edge): //V21.051 版本

发送指令格式: “:001w_flick_edge01-16=03\r\n”

指令解释: 同时启动 1-16ch 边沿捕获功能, 捕获时间是 3 秒;

适用多次流水或多次闪烁的场合, 比如汽车迎宾灯, 流水转向灯, 是频闪中最通用的一条指令;

返回指令格式: “:001w_flick_edge01-16=03\r\n”;

指令解释: “=03”的取值范围是 01-55 秒, 设置捕获时间长度, 单位是秒;

须提前设置好相应通道的硬件参数(ft&gain&w_flick_limit), 该指令下达后, 模块立即执行 edge 测试, 测试时间持续 3s(设定值); 测试期间, 模块不响应其他指令, 但可以发送 state 指令”查询模块状态(查询频率过快, 会降低底层采集速度), 待模块返回”idle”, 才可发送新的读写指令; 模块支持所有通道同时进行 edge 测试; 测量结果会一直保持在 RAM 中, 直到下次指令下达从新转换更新; 该指令收到后, 会先立即返回”:001w_flick_edge01-16=03\r\n”, 再进行 flow 测量;

在多个模块 RS485 并行测量多个流水灯时, 可以发送把地址改为 0, 例如发

送”:000w_flick_edge01-16\r\n”指令, 同时启动多个仪器捕获, 由于是 485 并联, 仪器会同时响应广播地址 0, 只有 ID=1 的仪器会正常返回值”:001w_flick_edge01-16\r\n”, 其他 ID 仪器不回复任何指令, 以免造成总线冲突;

---读取 LED 多次亮灭边沿时间戳(r_flick_edge): //V21.051 版本

发送指令格式: “:001r_flick_edge01-02=2\r\n”(数据一直保持, 直到下次测试更新)

指令解释:

读取 01-02 通道的 edge 测试结果,“=2”是每个通道读取 EDGE 次(2 次)点亮熄灭的时间戳, 最多是 10 次;

返回数据个数 CNT 等于通道数 CHL 乘以边沿次数 EDGE, 再乘以 2(上升沿和下降沿),

$CNT=CHL*EDGE*2$;

测量的时间参考 0 点, 是发送 w_flick_flow 启动指令后, 仪器返回 w_flick_flow 后, 计为 0ms;

返回指令格式: “:001r_flick_edge=1000,1100,1200,1300,2000,2100,2200,2300,\r\n”;

指令解释:

“=”后面值是 ms, 全部整数, 前面 4 个数据是 1CHL 时间戳, 后面 4 个数据是 2CHL 时间戳,

1000: 1CHL 第一次点亮的时间,

1100: 1CHL 点亮后第一次熄灭的时间,

1200: 1CHL 第二次点亮的时间,

1300: 1CHL 第二点亮后再次熄灭的时间,

2000: 2CHL 第一次点亮的时间,

2100: 2CHL 点亮后第一次熄灭的时间,

2200: 2CHL 第二次点亮的时间,

2300: 2CHL 第二点亮后再次熄灭的时间,

//////////下面是读取 flick 期间捕获的光度数据//////////

---读取 flick 的照度数据(r_flick_lx): //V1.5 版本

发送指令格式: “:001r_flick_lx01-02\r\n”(数据一直保持, 直到下次测试更新)

返回指令格式: “:001r_flick_lx=123,234,\r\n”

指令解释:

“=”后面的值依次是 01-02 通道的“lx1(%0.0f),lx2(%0.0f),”;该数据是 LED 点亮期间 lux 的最大值, 该值的大小可以反映 LED 的光照度(光亮度);

---读取 flick 的色度数据(r_flick_chroma):

发送指令格式: “:001r_flick_chroma01-01\r\n”(V19.6 版-数据一直保持, 直到下次测试更新)

返回指令格式: “:001r_flick_chroma=1000.0,0.3333,0.4444,555.5,85.2,6500,0.00123,\r\n”

指令解释:

“=”后面的值依次是 01-01 通

道“lux(%0.1f),x(%0.4f),y(%0.4f),dowave(%0.1f),duty(%0.1f),cct(%0.0f),fd(%0.5f),”

该指令可以高效获取 LED 的色度数据;

lux=1000.0,与 r_lux 指令相同;

x=0.3333,y=0.4444,与 r_xy 相同,是 CIE1931 的 xy 坐标;

dowave=555.5nm,duty=85.2%,与 r_wavesi 功能相似;

cct=6500K,与 r_cct 指令类似;

fd=0.00123,该浮点数是预留参数, 默认是 ADC 饱和程度 i%(参考 r_flick_rgbi 中的 i 值);

---读取 flick 的 rgbi 数据(r_flick_rgbi):

发送指令格式: “:001r_flick_rgbi01-01\r\n”(V19.6 版-数据一直保持, 直到下次测试更新)

返回指令格式: “:001r_flick_rgbi=255,244,105,50.00\r\n”

“=”后面的值依次是 01-01 通道的“r(%d),g(%d),b(%d),i(%0.2f),”; 该 RGB 范围为 0-255;

I 是相对于最大测量强度的百分比(0.00-100.00), I 可以作为相对亮度的数据(对比测试比较实用),但同一光强下, 修改 Gain 参数, 会改变 I 的值;I=100.00%时, 说明传感器已完全曝光;该指令与 r_rgb 指令读取效果一样;

---读取 flick 的 hsl 数据(r_flick_hsl):

发送指令格式: ":001r_flick_hsl01-01\r\n"(V23.033 版-数据一直保持, 直到下次测试更新)

返回指令格式: ":001r_flick_hsl=3,94,50,26.8,\r\n"

指令解释:

"="后面的值依次是 01-01 通道的"h(%d),s(%d),l(%d),i(%0.3f)," , h 是色调(0-360), s 是色纯度(0-100), L 是颜色明亮程度((0-100)一般不用), I 值的大小可以表征 LED 的光强; 与 r_hsl 指令读取结果一样;

---读取 flick 的 Raw Data ADC 数据(r_flick_rgbc):

发送指令格式: ":001r_flick_rgbc01-01\r\n"(V19.6 版-数据一直保持, 直到下次测试更新)

返回指令格式: ":001r_flick_rgbc=123,234,345,678,\r\n"

指令解释:

"="后面的值依次是 01-01 通道的"r(%d),g(%d),b(%d),w(%d)," , 属于传感器 rgbw 原始 ADC 数据;该数据是 LED 点亮期间 rgbw 的最大值, 该值的大小可以表征 LED 的光强, 比例可以定性判断 LED 的颜色; 与 rgbw 指令读取结果一样;

////////////////////////////////////

---读取被测 LED 的亮灭状态(r_led_ch1)

发送指令格式: ":001r_led_ch101-02\r\n" //可快速判断 LED 亮灭状态, 也可用于数码管的段显识别

返回指令格式: ":001r_led_ch1=0,1,\r\n"

指令解释:

发送: "01 代表 01(%02d)通道 CH1;

返回: 1 代表点亮, 0 代表熄灭;

可以选择某一种光强数据与 limit_flick 数据比较, 比较规则和 flick 逻辑一样, 参看 flick_mode 指令;

---写入 LED 发射光源的亮灭状态(w_led_disp)

发送指令格式: ":001w_led_disp01-01=1\r\n" //只适用于自带 LED 发射端口的产品

返回指令格式: ":001w_led_disp01-01=1\r\n"

指令解释:

发送: "01 代表 01(%02d)led 输出通道 CH1, 最多是 8CH;

---读取 LED 发射光源的亮灭状态(r_led_disp)

发送指令格式: ":001r_led_disp01-02\r\n" //只适用于自带 LED 发射端口的产品

返回指令格式: ":001r_led_disp=0,1,\r\n"

指令解释:

发送: "01 代表 01(%02d)led 输出通道 CH1;

返回: 1 代表点亮, 0 代表熄灭;

//////////////////////////////////offset 功能-----适用 V19.3 及以后版本////////////////////////////////////

---使能 Offset 组别(w_offset_en)

发送指令格式: “:001w_offset_en01-02=8\r\n” 或 “:001w_offset_en01-02=0\r\n”

返回指令格式: “:001w_offset_en01-02=8\r\n” 或 “:001w_offset_en01-02=0\r\n”

指令解释:

发送: 01-02(%02d)是 led 连续通道 CH1-CH2;

08(%02d):使能 CH1-CH2 的第 8 组别的 offset 参数, 如果该值是 0, 就是关闭 CH1-CH2 的 offset 功能;

---写入 Offset 组别的补偿值 dx(w_offset_dx)

发送指令格式: “:001w_offset_dx01-08=0.0001\r\n”

返回指令格式: “:001w_offset_dx01-08=0.0001\r\n”

指令解释:

发送: 01(%02d)是 led 通道 CH1;

08(%02d)是第一通道,第 8 组的 dx, 一共支持设置 8 组;

测量时可以随意打开其中一组 dx 参与运算, 也可以关闭所有组的 dx;

0.0001 是 dx 的数据,取值范围是(-1.0000 ~ 1.0000),在模块内部是与 x 坐标相加关系;

该参数使能后可以自动修正主波长 d_{wave} 和色温 CCT 等色度参数;

---写入 Offset 组别的补偿值 dy(w_offset_dy)

发送指令格式: “:001w_offset_dy01-08=0.0001\r\n”

返回指令格式: “:001w_offset_dy01-08=0.0001\r\n”

指令解释:

发送: 01(%02d)是 led 通道 CH1;

08(%02d)是第一通道,第 8 组的 dy, 一共支持设置 8 组;

测量时可以随意打开其中一组 dx 参与运算, 也可以关闭所有组的 dy;

0.0001 是 dy 的数据,取值范围是(-1.0000 ~ 1.0000),在模块内部是与 y 坐标相加关系;

该参数使能后可以自动修正主波长 d_{wave} 和色温 CCT 等色度参数;

---写入 Offset 组别的亮度值系数 kl(w_offset_kl)

发送指令格式: “:001w_offset_kl01-08=1.001\r\n”

返回指令格式: “:001w_offset_kl01-08=1.001\r\n”

指令解释:

发送: 01(%02d)是 led 通道 CH1;

08(%02d)是第 1 通道,第 8 组的 kl, 一共支持设置 8 组;

测量时可以随意打开其中一组 kl 参与运算, 也可以关闭所有组的 kl;

1.001 是 kl 的数据,取值范围(0.001-32.000),在内部与 lux 相乘关系,该参数只能改变 lux 数据;

---清零 offset 所有数据(w_offset_clear)

发送指令格式: “:001w_offset_clear\r\n”

返回指令格式: “:001w_offset_clear\r\n”

指令解释: 清除 offset 相关所有通道所有组别数据, en 组别=0, dx=0,dy=0,kl=1.0; 清零是对内存参数进行清零, 如果要保存, 还需要发送 w_offset_save 保存指令

---保存 offset 所有数据到 flash (w_offset_save)

发送指令格式: “:001w_offset_save\r\n”

返回指令格式: “:001w_offset_save\r\n”

指令解释: 该指令只能保存 offset 相关参数, 其他保存指令也不能保存 offset 参数;
该指令会擦写 flash, 寿命小于 10 万次, 保存时间大于 1 秒, 尽量不要在主程序内频繁操作;

---读取 offset 使能组别(r_offset_en)

发送指令格式: “:001r_offset_en01-02\r\n”

返回指令格式: “:001r_offset_kl=8,8\r\n”

指令解释:

发送: 01-02(%02d)是 led 连续通道 CH1-CH2;

8,8:使能了 CH1-CH2 的第 8 组别的 offset, 如果该值是 0,就是关闭了 CH1-CH2 的 offset 功能;

---读取 offset 补偿值 dx (r_offset_dx)

发送指令格式: “:001r_offset_dx01-02\r\n”

返回指令格式: “:001r_offset_dx=-0.0001\r\n”

指令解释:

发送: 01(%02d)是 led 通道 CH1;

02(%02d)是读取第一通道,第 2 组的 dx, 一共支持设置 8 组,-0.0001 是 dx 的数据;

---读取 offset 补偿值 dy(r_offset_dy)

发送指令格式: “:001r_offset_dy01-02\r\n”

返回指令格式: “:001r_offset_dy=-0.0001\r\n”

指令解释:

发送: 01(%02d)是 led 通道 CH1;

02(%02d)是读取第一通道,第 2 组的 dy, 一共支持设置 8 组,-0.0001 是 dy 的数据;

---读取 offset 补偿值 kl(r_offset_kl)

发送指令格式: “:001r_offset_kl01-02\r\n”

返回指令格式: “:001r_offset_kl= 1.001\r\n”

指令解释:

发送: 01(%02d)是 led 通道 CH1;

02(%02d)是读取第一通道,第 2 组的 kl, 一共支持设置 8 组,1.001 是 kl 的数据;

////////DIO 指令仅限含 DIO 模块的产品使用, 部分 LBB 系列可选配 16DIO 模块////////

---读取 DI 单通道输入状态(r_inbit)

发送指令格式: “:001r_inbit01\r\n”

返回指令格式: “:001r_inbit=0\r\n”或“:001r_inbit=1\r\n”

指令解释:

发送: “01 代表 01(%02d)通道(DIN1),每次读取一路输入点;

返回: 1 代表有输入(光耦导通), 0 代表无输入(光耦无导通);

---读取 DI1-8 通道输入状态 U8(r_in_u8)

发送指令格式: “:001r_in_u8\r\n”

返回指令格式: “:001r_in_u8=128\r\n”

指令解释:

发送: “u8”代表 DIN1-DIN8 从低到高组成的 uint8 字节数据(模块默认支持 8DI);

返回: DIN1-DIN8 依次由低到高位组合成一个字节数据, 128(1000 0000)代表 DIN8 有输入,

其他无输入;

---写入 DO 多个通道输出状态(w_outbit)

发送指令格式: “:001w_outbit01-02=1\r\n”

返回指令格式: “:001w_outbit01-02=1\r\n”

指令解释:

发送: “01-02 代表从 01(%02d)通道到 02(%02d)通道, 必须是从小到大, 或者是相等(通道最大值为 16);

=后面的“1”代表打开 DO1-DO2, “0”代表关闭通道 DO1-DO2

返回: 如果指令正常, 返回值就和发送字符一样;

---写入 DO-16 通道输出 U16(w_out_u16)

发送指令格式: “:001w_out_u16=4\r\n”

返回指令格式: “:001w_out_u16=4\r\n”

指令解释:

发送: “u16”代表 DO1-DO16 从低到高组成的 uint16 整形数据;

=后面的“4”(0000 0000 0000 0100)代表打开 DO3, 其他通道全部关闭,

返回: 如果指令正常, 返回值就和发送字符一样;

////////2022 年 2 月之后的仪器才支持 net 指令---V22.011 以后版本////////

---写入 net 的工作模式(w_net_mode)

发送指令格式: “:001w_net_mode=1\r\n”

返回指令格式: “:001w_net_mode=1\r\n”

发送指令解释: net_mode 取值范围是 0:Disable,1:TCP_Server,1:TCP_Client,3:UDP;

默认是关闭网口的, 一旦打开网口功能, 仪器初始化会有 5 秒钟网络连接时间, 如果需要网口通信, 建议 LED 分析仪工作在 1:TCP_Server 模式下; 在没有启动网口之前, 所有 net 参数都是通过 USB/RS485/RS232 串口协议配置的, 所有 net 参数都是可以通过 save_to_flash 指令保存的;

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 net 的工作模式(r_net_mode)

发送指令格式: “:001r_net_mode\r\n”

返回指令格式: “:001r_net_mode=1\r\n”

---写入 net 的仪器的源 IP 地址(w_net_sip)

发送指令格式: “:001w_net_sip=192,168,0,100\r\n”

返回指令格式: “:001w_net_sip=192,168,0,100\r\n”

发送指令解释: sip 是 LED 分析仪网口的 IP 地址, 该网络所有 ip 满足 IPV4 格式;

为了兼容其他解析数字指令, IP 地址数字间隔是用逗号(',')分割, 不是用点('.');

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 net 的仪器的源 IP 地址(r_net_sip)

发送指令格式: “:001r_net_sip\r\n”

返回指令格式: “:001r_net_sip=192,168,0,100,\r\n”

---写入 net 的上位机的目标 IP 地址(w_net_dip)

发送指令格式: “:001w_net_dip=192,168,0,10\r\n”

返回指令格式: “:001w_net_dip=192,168,0,10\r\n”

发送指令解释: dip 是上位机电脑的网口的 IP 地址,

只有 LED 分析仪网口工作在 TCP_Client 模式下, dip 配置才能用到;

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 net 的上位机的目标 IP 地址(r_net_dip)

发送指令格式: “:001r_net_dip\r\n”

返回指令格式: “:001r_net_dip=192,168,0,10,\r\n”

---写入 net 的网关 Gateway IP 地址(w_net_gip)

发送指令格式: “:001w_net_gip=192,168,0,1\r\n”

返回指令格式: “:001w_net_gip=192,168,0,1\r\n”

发送指令解释: gip 是网口的网关 Gateway IP 地址,

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 net 的网关 Gateway IP 地址(r_net_gip)

发送指令格式: “:001r_net_gip\r\n”

返回指令格式: “:001r_net_gip=192,168,0,1,\r\n”

---写入 net 的子网掩码 Subnet Mask IP 地址(w_net_mip)

发送指令格式: “:001w_net_mip=255,255,255,0\r\n”

返回指令格式: “:001w_net_mip=255,255,255,0\r\n”

发送指令解释: gip 是网口的网关 Gateway IP 地址,

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 net 的子网掩码 Subnet Mask IP 地址(r_net_mip)

发送指令格式: “:001r_net_mip\r\n”

返回指令格式: “:001r_net_mip=255,255,255,0,\r\n”

---写入 net 的 LED 仪器的源端口号(w_net_sport)

发送指令格式: “:001w_net_sport=8000,8001,8002,8003,8004,8005,8006,8007\r\n”

返回指令格式: “:001w_net_sport=8000,8001,8002,8003,8004,8005,8006,8007\r\n”

发送指令解释: sport 是 LED 分析仪网口端口号, 在 TCP_Server 模式下, 支持 8 个同时通信;
在 TCP_Client 模式下, 只采用第一个端口号(8000)访问上位机服务器;

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 nnet 的 LED 仪器的源端口号(r_net_sport)

发送指令格式: “:001r_net_sport\r\n”

返回指令格式: “:001r_net_sport=8000,8001,8002,8003,8004,8005,8006,8007,\r\n”

---写入 net 的上位机服务器的目标端口号(w_net_dport)

发送指令格式: “:001w_net_dport=10000\r\n”

返回指令格式: “:001w_net_dport=10000\r\n”

发送指令解释: dport 是 PC 上位机作为 TCPServer 时端口号, 只有 LED 分析仪在 TCP_Client 模式下, 该端口号才有作用;

返回指令解释: 如果指令正常, 返回值就和发送字符一样;

---读取 net 的上位机服务器的目标端口号(r_net_dport)

发送指令格式: “:001r_net_dport\r\n”

返回指令格式: “:001r_net_dport=10000\r\n”

---读取 net 的 MAC 地址(r_net_mac)

发送指令格式: “:001r_net_mac\r\n”

返回指令格式: “:001r_net_mac=101,102,103,104,105,106,\r\n”

指令解释: mac 格式也是按照十进制字符串返回的, 不是按照 16 进制字符串格式;

---读取 net 的所有配置参数(r_net_all)

发送指令格式: “:001r_net_all\r\n”

返回指令格式: “.....\r\n”

指令解释: 该指令适合在串口调试助手下, 手动读取网络配置信息,

网口功能启动后, LED 分析仪在复位初始化时, 会从 USB 接口打印出来网口所有的配置信息;

---试运行 net 网口通信连接(w_net_run)

发送指令格式: “:001w_net_run\r\n”

返回指令格式: “:001w_net_run\r\n”

指令解释: 网线插上后, net 参数配置完成后, 可以先发送该指令试运行一下网络连接, 该指令发送后, LED 分析仪上的 D9 指示灯会连续闪烁 4 秒连接网络, 期间 USB 接口会自动发送网络参数信息到上位机, RJ45 上的橙,绿色灯也会点亮; 一旦网络通信连接正常, 说明配置参数正确, 再发送”:001save_to_flash\r\n”保存起来, 后续上电后网口会自动启动网口功能;

////////WhiteBalance 白平衡系数配置指令-V23.031 及以后的版本支持 20230301////////

////建议用官方调试软件简单高效配置 whitebalance 系数, 不建议用该指令配置/////

---写入某几个通道的 Red 的白平衡系数(wr_whitebalance)

发送指令格式: “:001wr_whitebalance01-02=0.5,11\r\n”

返回指令格式: “:001wr_whitebalance01-02=0.5,11\r\n”

指令解释: “wr”是 Write Red 的缩写, “01-02”是第 1CH 到第 2CH, “=0.5”是写入 R 的白平衡系数 Kr=0.5,系数范围是 0.001~30 建议 Krgb 中最大系数写 1.0; 最后的”,11”是指 w_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 11, 12, 13,14,15,16,如果最后的编号不写, 或者写入错误, 那就强制为编号 0;正常接收到指令后, 返回相同的指令;

白平衡系数, 可以简单理解成给传感器底层的 rgb 或 XYZ 原始值各乘以一个系数, 对于 LED 单点校准比较实用;

---写入某几个通道的 Green 的白平衡系数(wg_whitebalance)

发送指令格式: “:001wg_whitebalance01-02=0.5,11\r\n”

返回指令格式: “:001wg_whitebalance01-02=0.5,11\r\n”

指令解释: “wg”是 Write Green 的缩写, “01-02”是第 1CH 到第 2CH, “=0.5”是写入 R 的白平衡系数 Kg=0.5,系数范围是 0.001~30 建议 Krgb 中最大系数写 1.0; 最后的”,11”是指 w_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 11, 12, 13,14,15,16,如果最后的编号不写, 或者写入错误, 那就强制为编号 0;正常接收到指令后, 返回相同的指令;

---写入某几个通道的 Blue 的白平衡系数(wb_whitebalance)

发送指令格式: “:001wb_whitebalance01-02=0.5,11\r\n”

返回指令格式: “:001wb_whitebalance01-02=0.5,11\r\n”

指令解释: “wb”是 Write Blue 的缩写, “01-02”是第 1CH 到第 2CH, “=0.5”是写入 R 的白平衡系数 Kb=0.5,系数范围是 0.001~30, 建议 Krgb 中最大系数写 1.0; 最后的”,11”是指

w_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 11, 12, 13, 14, 15, 16, 如果最后的编号不写, 或者写入错误, 那就强制为编号 0; 正常接收到指令后, 返回相同的指令;

---写入某几个通道的 WBxx 的 Lux 亮度系数(wl_whitebalance)

发送指令格式: ":001wl_whitebalance01-02=0.5,11\r\n"

返回指令格式: ":001wl_whitebalance01-02=0.5,11\r\n"

指令解释: "wl" 是 Write Lux 的缩写, "01-02" 是第 1CH 到第 2CH, "=0.5" 是写入 Lux 的亮度系数 KL=0.5, 系数范围是 0.001~30; 最后的 "11" 是指 w_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 3, 4, 5, 6, 11, 12, 13, 14, 15, 16, 如果最后的编号不写, 或者写入错误, 那就强制为编号 0; 正常接收到指令后, 返回相同的指令; 该指令仅支持仪器 V23.111 及以后的版本;

---读取某几个通道的 WBxx 的 Lux 亮度系数(rl_whitebalance)

发送指令格式: ":001rl_whitebalance01-02=11\r\n"

返回指令格式: ":001rl_wb=0.5,0.6\r\n"

指令解释: "rl" 是 Read Lux 的缩写, "01-02" 是第 1CH 到第 2CH, 最后的 "=11" 是指 r_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 3, 4, 5, 6, 11, 12, 13, 14, 15, 16, 如果最后的编号不写, 或者写入错误, 那就强制为编号 0; 返回指令做了简化, 0.5 是第 1CH 第 11 组的 KL 系数, 0.6 是第 2CH 的第 11 组 KL 系数; 该指令仅支持仪器 V23.111 及以后的版本;

---读取某几个通道的 Red 的白平衡系数(rr_whitebalance)

发送指令格式: ":001rr_whitebalance01-02=11\r\n"

返回指令格式: ":001rr_wb=0.5,0.6\r\n"

指令解释: "rr" 是 Read Red 的缩写, "01-02" 是第 1CH 到第 2CH, 最后的 "=11" 是指 r_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 11, 12, 13, 14, 15, 16, 如果最后的编号不写, 或者写入错误, 那就强制为编号 0; 返回指令做了简化, 0.5 是第 1CH 第 11 组的 Kr 系数, 0.6 是第 2CH 的第 11 组 Kr 系数;

---读取某几个通道的 Green 的白平衡系数(rg_whitebalance)

发送指令格式: ":001rg_whitebalance01-02=11\r\n"

返回指令格式: ":001rg_wb=0.5,0.6\r\n"

指令解释: "rg" 是 Read Green 的缩写, "01-02" 是第 1CH 到第 2CH, 最后的 "=11" 是指 r_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 11, 12, 13, 14, 15, 16, 如果最后的编号不写, 或者写入错误, 那就强制为编号 0; 返回指令做了简化, 0.5 是第 1CH 第 11 组的 Kg 系数, 0.6 是第 2CH 的第 11 组 Kg 系数;

---读取某几个通道的 Blue 的白平衡系数(rb_whitebalance)

发送指令格式: ":001rb_whitebalance01-02=11\r\n"

返回指令格式: ":001rb_wb=0.5,0.6\r\n"

指令解释: "rb" 是 Read Blue 的缩写, "01-02" 是第 1CH 到第 2CH, 最后的 "=11" 是指 r_target_type 指令中的 WB 编号 11, 范围是 0, 1, 2, 11, 12, 13, 14, 15, 16, 如果最后的编号不写, 或者写入错误, 那就强制为编号 0; 返回指令做了简化, 0.5 是第 1CH 第 11 组的 Kb 系数, 0.6 是第 2CH 的第 11 组 Kb 系数;

---保存白平衡系数(save_whitebalance)

发送指令格式: ":001save_whitebalance\r\n"

返回指令格式: ":001save_whitebalance\r\n"

白平衡系数写入后会立即生效, 因此只有当配置完成后, 确认可以正常测量后, 再保存即可, 做到掉电不丢失; 保存时间一般较长(非光纤系列仪器可能会超过 8 秒, 光纤系列主机一般在 3S 以内), 而且内部 Flash 存在擦写寿命限制, 不要频繁发送该指令; 该保存指令

和"save_to_flash"的保存对象不一样；

/////调整传感器原始 rgb 或 XYZ 数据 AutoWB 比例关系-V23.031 及以后版本支持////////

///只有当仪器默认配置的系数不满足测量要求时，再配置该指令！！////////

---写入 Sensor 的原始数据的 AutoWB 系数(w_autowb_sens)

发送指令格式: ":001w_autowb_sens02_k01=3.5,10\r\n"

返回指令格式: ":001w_autowb_sens02_k01=3.5,10\r\n"

指令解释: "sens02"中的 02 是传感器的硬件编号，可以通过 r_system_sens 指令读取，该指令与通道没有关系，只与传感器硬件编号有关；

接收到指令后，返回相同的指令；

---读取 Sensor 的原始数据的 AutoWB 系数(r_autowb_sens)

发送指令格式: ":001r_autowb_sens02_k01\r\n"

返回指令格式: ":001r_autowb_sens02_k01=3.5,10,\r\n"

指令解释: "sens02"中的 02 是传感器的硬件编号，k01 中的 01 是第一组颜色；

返回值表示 sens=2 的第 1 组颜色比例关系是 K1=3.5, K2=10；

写入(读取)指令	替代指令	说明
w_autowb_sens02_k01=3.5,10 (r_autowb_sens02_k01)	w_autowb_r_rg_rb02=3.5,10 (r_autowb_r_rg_rb02)	对 SENS=2 的传感器写入 K1=r/g,K2=r/b 两个系数，范围是 0.001~1000,目的是写入筛选第 1 组(k01)颜色(Red 红灯)的比例关系，在 w_target_type=10(AUTOWB)模式下内部判断逻辑是：当 K1>3.5 且 K2>10 时，被认定为红灯，仪器会自动代入 RedWB(11)的白平衡系数参与计算；
w_autowb_sens02_k02=10,2 (r_autowb_sens02_k02)	w_autowb_g_gr_gb02=10,2 (r_autowb_g_gr_gb02)	写入筛选第 2 组(k02)Green 绿灯的比例关系，当 K1(g/r)>10 且 K2(g/b)>2 时，被认定为绿灯，仪器会自动代入 GreenWB(12)的白平衡系数参与计算；
w_autowb_sens02_k03=30,4 (r_autowb_sens02_k03)	w_autowb_b_br_bg02=30,4 (r_autowb_b_br_bg)	写入筛选第 3 组(k03)Blue 蓝灯的比例关系，当 K1(b/r)>30 且 K2(b/g)>4 时，被认定为绿灯，仪器会自动代入 BlueWB(13)的白平衡系数参与计算；
w_autowb_sens02_k04=2.5,5.5 (r_autowb_sens02_k04)	w_autowb_y_rb_gb02=2,5.5 (r_autowb_y_rb_gb02)	写入筛选第 4 组(k04)黄橙色灯的比例关系，当 K1(r/b)>2 且 K2(g/b)>5.5 时，被认定为黄橙灯，仪器会自动代入 YellowWB(14)的白平衡系数参与计算；
w_autowb_sens02_k05=10,10 (r_autowb_sens02_k05)	w_autowb_c_gr_br02=10,10 (r_autowb_c_gr_br02)	写入筛选第 5 组(k05)Cyan 青色灯的比例关系，当 K1(g/r)>10 且 K2(b/r)>10 时，被认定为青色灯，仪器会自动代入 C/R_WB(15)的白平衡系数参与计算；
当 w_target_type=10(AUTOWB)模式下，k01~k05 前 5 组都匹配不到时，就判定为白光 WhiteWB(16)，仪器会自动代入 WhiteWB(16)的白平衡系数参与计算； 如果 w_target_type!=10，那么该比例系数就不会参与对比筛选了； autowb 相关参数的保存指令也是"save_to_flash"；		

四：指令表汇总：

串口指令表(地址为 001)			
指令分类	指令关键字串	功能描述	备注
状态查询	":001state\r\n"	查询模块状态 " :001idle\r\n"或者":001busy\r\n"	"
模块信息	:001idn	读取模块产品信息	
模块 ID	:000r_id	读取模块的 ID,不管 USB/RS485/LAN 接口, 都要用模块 ID(保持协议格式不变)	
	:001w_id	写入模块 ID(%03d),立即生效并自动保存该参数	
波特率 Baud	w_baud1	设置 USB-RS232 波特率, 立即生效并自动保存	Baud 范围 0-9
	w_baud2	设置 RS485 端口波特率, 立即生效并自动保存 {2400,4800,9600,19200,38400,57600,115200,230400,460800,921600}	
配置 ADC 采样模式	w_system_samp	配置 ADC 采样模式 0: 连续模式; 1: 单次采样	
读取 ADC 采样模式	r_system_samp	读取 ADC 采样模式(V19.6 版)	
复位系统	w_system_reset	系统从新复位初始化, 不是恢复出厂设置	
读取 Sensor 编号	r_system_sens	读取仪器传感器硬件编号	(V23.031)
参数保存	save_to_flash	把用户配置在 RAM 中参数保存到 flash, 掉电不丢失, 不建议频繁保存	
恢复出厂设置	default	所有用户配置的参数恢复出厂设置	
采样周期	w_ft	设置传感器采样周期, 所有通道同步采样, 立即执行, 暂存在 RAM	ft 越大, 数据越大, 采样周期越长
	r_ft	读取传感器采样时间索引号	
	r_ftms	读取传感器采样时间 ms(V1.5 版本)	
ADC 增益	w_gain	设置传感器硬件增益, 立即执行, 暂存在 RAM	增益越大, 数据越大
	r_gain	读取传感器增益索引号	
	r_gainx	读取传感器增益实际值(V1.5 版本)	
RGB 数据 (只有 RGB 和 XYZ 系列才有此功能)	rgbw	读取传感器 RGBW 原始 ADC 值	
	r_rgbi	读取换算到 0-255 范围的 RGB 和相对功率强度 I(0-100%)	
	r_hsli	读取色度/饱和度/亮度/相对亮度 I(0-100%)	
Lux 照度/功率	w_k_lux	写入 lux 补偿系数,立即执行, 暂存于 RAM 中	
	r_k_lux	读取 lux 补偿系数 k(%0.3f);	
	w_k_uw	写入光功率补偿系数,立即执行, 暂存于 RAM 中	
	r_k_uw	读取光功率补偿系数 k(%0.3f)默认为 1.0;	
	r_lux	读取照度值 lux, 或者是亮度值 mcd/m^2,	

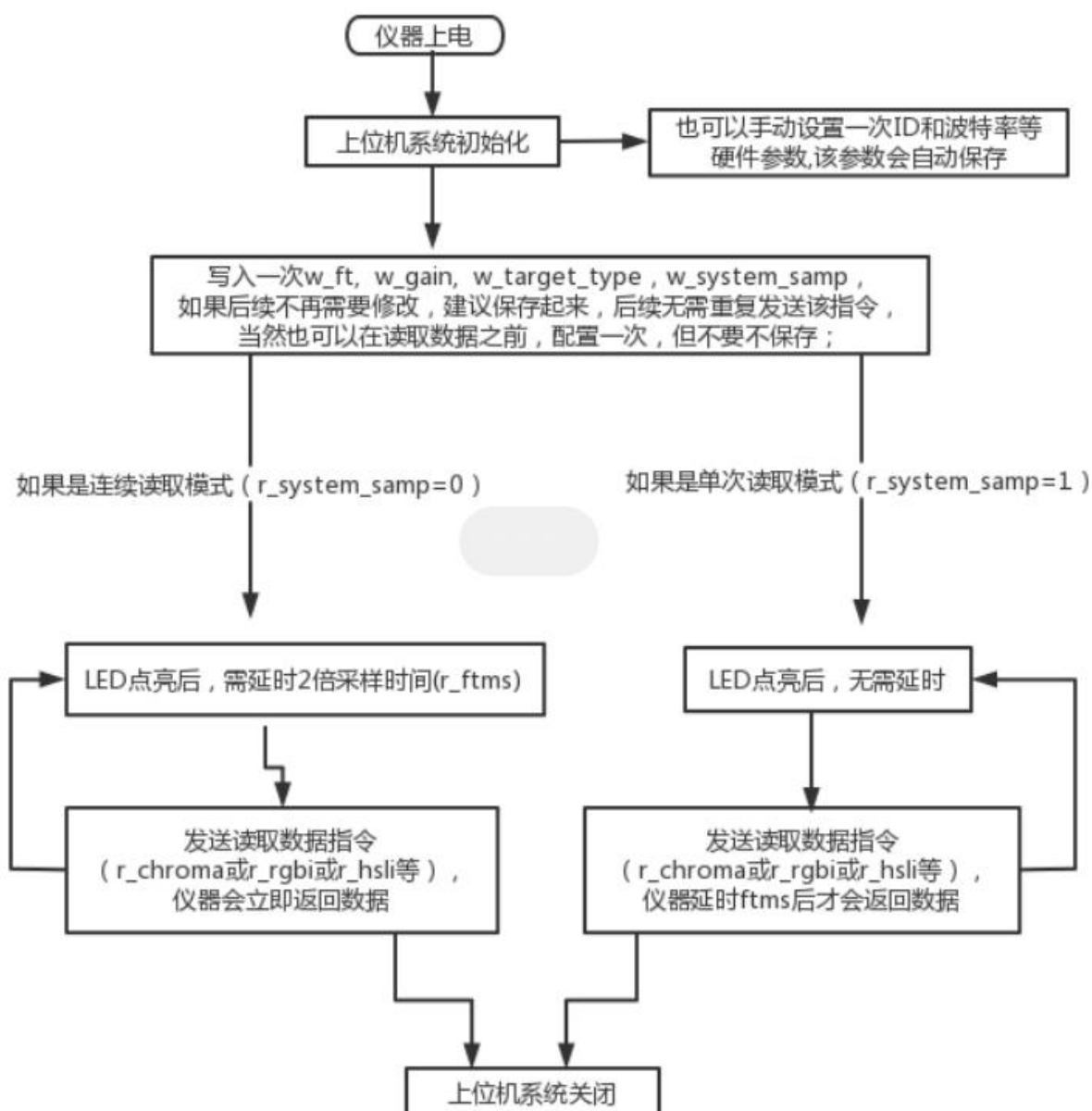
	r_uw_cm	读取光功率 uw/cm ² ,一般用于表征非可见光,某些产品不适用	
	r_cd_mm	读取亮度 cd/m ² ,面光源亮度数据, 适合非光纤类产品 (V19.3 版本)	
流明 lm	r_lm	读取积分球探头的流明值, 流明探头出厂前已校准, 与 lux 成正比关系	积分球反射率和面积 cm ²
坎德拉 cd	r_cd_lm	读取 LED 的坎德拉 cd 和流明 (适用于朗伯体光源)	探测角度和距离
Flick 频闪功能			
	w_flick_mode	设置 LUX-R-G-B-W 为对比对象-立即执行	
	r_flick_mode	读取 flick 对比通道模式- (V1.5 版本)	
	w_flick_limit	设置 flick 亮度门限值,立即执行,暂存于 RAM 中	
	r_flick_limit	读取 flick 亮度门限值	
	w_flick_color_max	设置 flick 颜色上限值,立即执行,暂存于 RAM 中	
	w_flick_color_min	设置 flick 颜色下限值,立即执行,暂存于 RAM 中	
	r_flick_color_max	读取 flick 颜色上限值	
	r_flick_color_min	读取 flick 颜色下限值	
	w_flick_ts	设置并立即执行 flick 测试- (V1.5 版本)	
	r_flick_ts	读取 flick 结果(Cnt,fre,Tup,Tdw,Tduty,LUXmax)	
	r_flick_lx	读取 flick 期间的点亮时的 lux 数据-	
	r_flick_chroma	读取 flick 期间的点亮时的 chroma 数据	
	r_flick_rgbi	读取 flick 期间的点亮时的 rgbi 数据	
	r_flick_rgbc	读取 flick 期间的点亮时的 rgbc 原始数据	
	r_flick_hsli	读取 flick 期间的点亮时的 hsli 数据	
	w_flick_flow	设置并立即执行 flick_flow 测试- (V20.101 版本)	
	r_flick_flow	读取 flick_flow 结果 (V20.101 版本)	
	w_flick_edge	设置并立即执行 flick_flow 测试- (V21.051 版本)	
	r_flick_edge	读取 flick_edge 结果 (V21.051 版本)	
读取 LED 亮灭状态	r_led_ch1	快速读取每个通道是否有 led 亮灭, 适合数码管测量	
chroma 色度数据 (只有 XYZ 模块读取的数据是准确的, 为了兼容通信协议, 这些指令也适用于 RGB 系列产品, 但获取的数据不准确)	w_target_type	选择待测目标 LED 的类型, 立即执行, 暂存于 RAM	选择合适的类型可以提高色坐标的测量精度
	r_target_type	读取当前的 led 测试类型	
	r_Yxy	读取(CIE1931)三刺激色坐标和亮度	
	r_xy	读取 xy(CIE1931)色坐标数据	
	r_uv	读取 uv(CIE1976)色坐标数据	
	r_cct	读取 CCT(CIE1931)色温数据	
	r_cctd	读取 CCT 及 Duv 数据 (V1.5 版本)	
	r_dowave	读取主波长(CIE1931)数据	
	r_wavesi	读取主波长, 色纯度(色饱和度), lux 数据	

	r_chroma	读取 lux, x, y, dowave, duty%, cct, duv 色度数据;	
	w_sdcn_type	选择 sdcn 类型, 立即执行, 暂存于 RAM	
	r_sdcn_type	读取当前通道的 sdcn 类型	
	r_sdcn_data	读取读取当前通道的 sdcn	
	r_sdcn_lux	读取读取当前通道的 lux, sdcn, sdcn_type	
LED 发射光源	w_led_disp	打开 LED 的亮灭状态, 只适合带 LED 发射端口的产品	
	r_led_disp	读取 LED 的亮灭状态, 只适合带 LED 发射端口的产品	
DIO 读写 (配 DIO 的模块才可使用)	r_inbit	读取一位输入信号	光耦输入
	r_in_u8	读取 8 位输入组成一个字节	
	w_outbit	写入指定地址的 DO 状态	ULN2803-NPN 晶体管输出 (200MA)
	w_out_u16	写入 16 位 DO 状态	
Offset (V19.3 版本)	w_offset_en	读写某些通道的某个组别功能	r_offset_en
	w_offset_dx	读写 cie1931-x 的 offset_dx 参数	r_offset_dx
	w_offset_dy	读写 cie1931-x 的 offset_dy 参数	r_offset_dy
	w_offset_kl	读写 lux 的 offset_kl 参数	r_offset_kl
	w_offset_save	保存所有 offset 相关参数	
	w_offset_clear	清零所有 offset 相关参数	
Net 网络参数配置指令 -适用于 2022-2 月之后 带 LAN 网口的新仪器	w_net_mode	读写网口的工作模式	r_net_mode
	w_net_sip	读写仪器的源 ip 地址	r_net_sip
	w_net_dip	读写上位机的目标 ip 地址, 一般是电脑的 ip	r_net_dip
	w_net_gip	读写网络的网关 Gateway IP 地址	r_net_gip
	w_net_mip	读写网络的子网掩码 Subnet Mask IP 地址	r_net_mip
	w_net_sport	读写仪器网口的源端口号, 支持 8 个 ports	r_net_sport
	w_net_dport	读写上位机的网口目标端口号, 只有一个	r_net_dport
	r_net_mac	读取网口的 MAC 地址	
	w_net_run	软件初始化一次网口, 但网络参数并没有保存到 flash	
	r_net_all	读取所有网络配置信息, 只用在串口调试助手上	
强烈建议采用官方调试软件, 通过 USB 或 RS485 配置 net 参数, 方便省事!			
Whitebalance 白平衡系数读写 V23.031 WBxx 对应的 Lux 系数 Autowb 自动颜色匹配比例关系 V23.031	wr_whitebalance rr_whitebalance V23.111 版本及以后 w_autowb_sens r_autowb_sens	wg_whitebalance rg_whitebalance wl_whitebalance/rl_whitebalance w_autowb_r_rg_rb r_autowb_r_rg_rb	wb_whitebalance rb_whitebalance w_autowb_g_gr_gb r_autowb_g_gr_gb

五：软件开发流程图举例

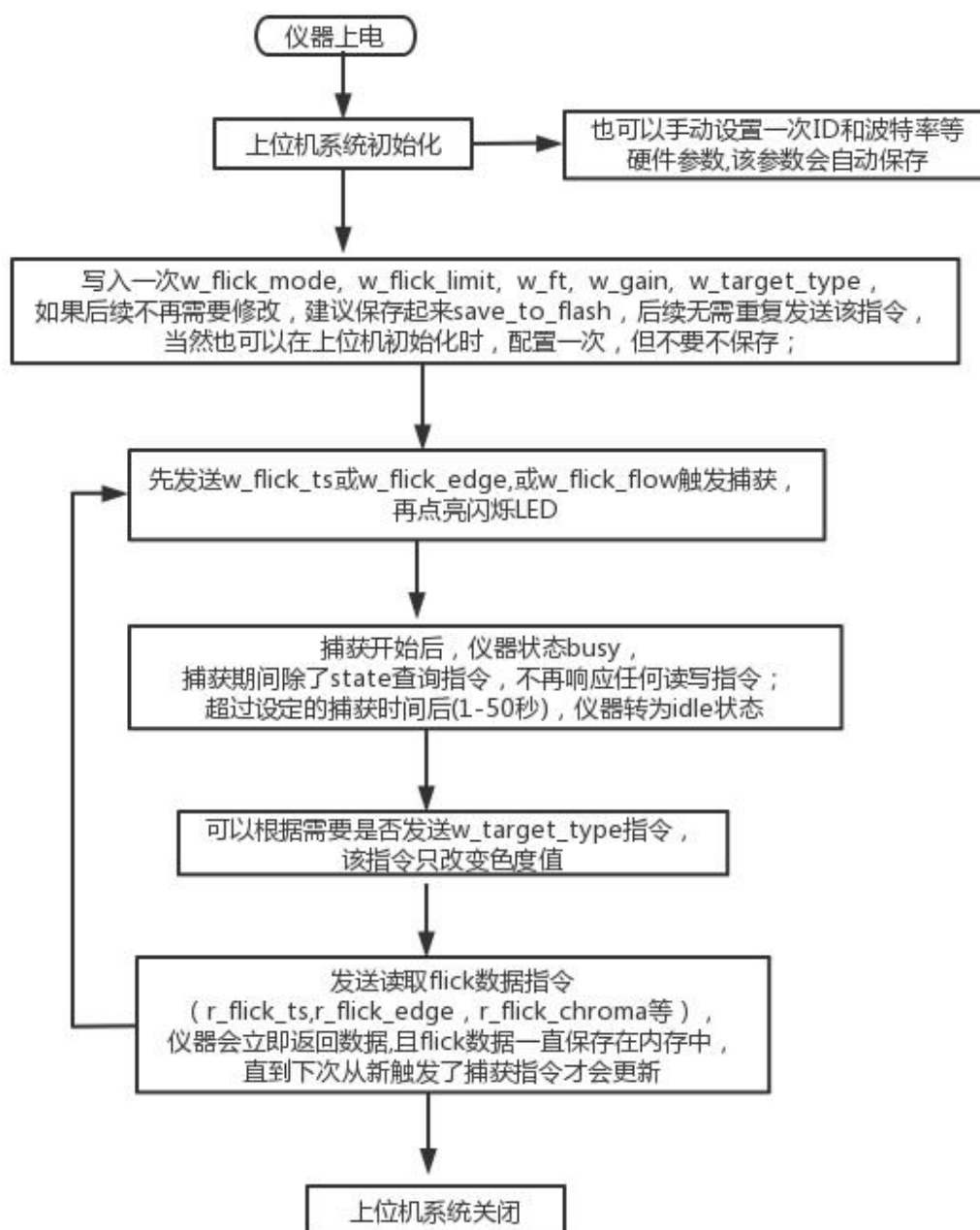
---单色常亮 LED 数据采集流程

单色常亮LED的光学数据采集流程



---单色闪烁 Flick-LED 数据采集流程

单色闪烁Flick-LED的光学数据采集流程



六：测试用例-指令读写实例

---读取 CH1 的 RED-LED 的 RGBI/HSLI(ID=1):

序号	发送指令(C 语言格式)	返回指令	备注
1	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	光越弱, gain, ft 要配置越大, 每款传感器的 gain 和 ft 范围不一样, 但仪器内部有软件防呆功能, 写入非法的序号, 主机不会生效; 一般情况下, 出厂默认配置就可以满足大部分测量要求, 除非 LED 特殊, 一般不需要频繁修改这两个硬件参数
2	“:001r_rgb01-01\n”	“:001r_rgb01=255,50,10,1.23\n”	RGB/HSI 系列仪器建议用其中一条指令判断颜色亮度即可, 不建议用 CIE1931-xy 专业光学参数
3	“:001r_hsl01-01\n”	“:001r_hsl01=5,95,55,1.23\n”	

---读取 CH1 的黄绿灯的主波长(ID=1):

序号	发送指令(C 语言格式)	返回指令	备注
1	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	可以配置一次保存起来(save_to_flash), 不用频繁发送;
2	“:001w_target_type01-01=4\n”	“:001w_target_type01-01=4\n”	选择黄绿色的 LED 读取模式 4, 也可以配置一次保存起来(save_to_flash), 不用频繁发送;
3	“:001r_waves01-01\n”	“:001r_waves01=570.5,80,345.6”	也可以用“:001r_chroma01-01\n”读取

---读取 CH1 的白光灯的 xy 色坐标(ID=1):

序号	发送指令(C 语言格式)	返回指令	备注
1	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	可以配置一次保存起来(save_to_flash), 不用频繁发送;
2	“:001w_target_type01-01=0\n”	“:001w_target_type01-01=0\n”	选择白光的 LED 读取模式 0, 也可以用 3, 建议用 0
3	“:001r_Yxy01-01\n”	“:001r_Yxy=123.4,0.3333,0.3333”	也可以用“:001r_chroma01-01\n”读取

---读取 CH1 的 RED-LED-flick 闪烁频率(ID=1):

序号	发送指令(C 语言格式)	返回指令	备注
1	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	“:001w_gain01-01=4\n” “:001w_ft01-01=4\n”	Ft 的设置采样时间要小于二分之一 LED 频闪期间的点亮时间
2	“:001w_flick_mode01-01=0\n” “:001w_flick_limit01-01=20\n”	“:001w_flick_mode01-01=0\n” “:001w_flick_limit01-01=20\n”	如果单色 LED 是固定的闪烁状态, 可以一次配置后, 保存起来(save_to_flash), 不用频繁发送;
3	“:001w_flick_ts01-01=5\n”	“:001w_flick_ts01-01=5\n”	根据需要也可以用“:001w_flick_edge01-01=3\n”
4	“:001r_flick_ts01-01\n”	“:001r_flick_ts=1.0,1000,1000,500,5\n”	上位机必须延时等待捕获时间到期后, 才可以发送读取指令, 不建议在捕获期间连续发送 state 查询指令, 这样会影响一点主机的处理速度
5	“:001r_flick_rgb01-01\n”	“:001r_flick_rgb01=255,50,10,1.23\n”	

---offset 并读取 4CH 的双色灯(ID=1):

测量场景: 仪器一共 4 个通道, 同时测量 4 颗白红 LED, 一个通道测量一颗白红 LED, 分别测量白红 LED 的亮度 Y(Lux)和色坐标 xy;

数据要求: 4 个通道测量白灯的 Lux/xy 保持一致, 4 个通道测量红灯的 Lux/xy 保持一致;
 假设仪器的 ID=1, 实现编程逻辑如下:

在正常读取数据之前, 调试机台时, 先要对白红 LED 的 Lux/xy 进行补偿 offset, 即对 4 个通道的 White 和 Red LED 分别补偿, 使相同颜色 Lux/xy 保持一致, 假定白灯(White)为 offset_en 的第一组 Group1, 红灯(Red)为 offset_en 的第二组 Group2; 这里规定, 在读取仪器原始数据前, 白灯用模式 0 读取(":001w_target_type01_04=0\n"), 红灯用模式 6 读取(":001w_target_type01_04=6\n");		
功能描述	指令格式(C 代码)	说明
第一次配置时, 建议清除仪器内部所有的补偿系数, 以免内部原来就有配置过系数;	":001w_offset_clear\n"	所有 CH 所有组别 Group 的 kl=1.0, dxy=0.0, en=0;
假设对 CH1 的 White-Lux 写入系数 k=1.1	":001w_offset_kl01_01=1.1\n"	第一个 01 是 CH 编号, 第二个 01 是第一个组别 Group1
假设对 CH2 的 White-Lux 写入系数 k=2.1	":001w_offset_kl02_01=2.1\n"	K=Lux_target_w/Lux_read_w
假设对 CH3 的 White-Lux 写入系数 k=3.1	":001w_offset_kl03_01=3.1\n"	Lux_target_w 是白光对标 Lux
假设对 CH4 的 White-Lux 写入系数 k=4.1	":001w_offset_kl04_01=4.1\n"	Lux_read_w 是未加系数前从仪器读取 Lux
假设对 CH1 的 Red-Lux 写入系数 k=1.2	":001w_offset_kl01_02=1.2\n"	01 是仪器 CH 编号, 02 是第二个组别,
假设对 CH2 的 Red-Lux 写入系数 k=2.2	":001w_offset_kl02_02=2.2\n"	第一个 02 是 CH 编号, 第二个 02 是第二个组别 Group2
假设对 CH3 的 Red-Lux 写入系数 k=3.2	":001w_offset_kl03_02=3.2\n"	
假设对 CH4 的 Red-Lux 写入系数 k=4.2	":001w_offset_kl04_02=4.2\n"	格式 ":001w_offset_kl[CH]_[Group]=1.0\n"
Lux 补偿值范围是 0.001~32.0, 如果 kl 不参与补偿, 就写入 1; 如果 kl>32.0, 那就需要对全局 lux 写入一个较大系数 k, 指令"w_k_lux", 先增大 lux 数值, 然后再计算这个 kl; 仪器通道有多个 lux 系数, 都是相乘关系, 出厂默认都是 1.0;		
假设对 CH1 的 White-xy 写入补偿值 dx=-0.011, dy=0.011	":001w_offset_dx01_01=-0.011\n" ":001w_offset_dy01_01=0.011\n"	第一个 01 是仪器 CH 编号, 第二个 01 是第一个组别 Group1,
假设对 CH2 的 White-xy 写入补偿值 dx=-0.021, dy=0.021	":001w_offset_dx02_01=-0.021\n" ":001w_offset_dy02_01=0.021\n"	dx=x_target_w-x_read_w; dy=y_target_w-y_read_w;
假设对 CH3 的 White-xy 写入补偿值 dx=-0.031, dy=0.031	":001w_offset_dx03_01=-0.031\n" ":001w_offset_dy03_01=0.031\n"	x(y)_target_w 是白灯的 xy 目标值
假设对 CH4 的 White-xy 写入补偿值 dx=-0.041, dy=0.041	":001w_offset_dx04_01=-0.041\n" ":001w_offset_dy04_01=0.041\n"	x(y)_read_w 是白灯在未补偿前的 xy 读取值 (r_chroma 指令)
假设对 CH1 的 Red-xy 写入补偿值 dx=-0.012, dy=0.012	":001w_offset_dx01_02=-0.012\n" ":001w_offset_dy01_02=0.012\n"	
假设对 CH2 的 Red-xy 写入补偿值 dx=-0.022, dy=0.022	":001w_offset_dx02_02=-0.022\n" ":001w_offset_dy02_02=0.022\n"	第一个 02 是仪器 CH 编号, 第二个 02 是第二个组别 Group2,
假设对 CH3 的 Red-xy 写入补偿值 dx=-0.032, dy=0.032	":001w_offset_dx03_02=-0.032\n" ":001w_offset_dy03_02=0.032\n"	
假设对 CH4 的 Red-xy 写入补偿值 dx=-0.042, dy=0.042	":001w_offset_dx04_02=-0.042\n" ":001w_offset_dy04_02=0.042\n"	":001w_offset_dx[CH]_[Group]=0\n" ":001w_offset_dy[CH]_[Group]=0\n"
xy 补偿值范围是-0.9~0.9, 如果 xy 不参与补偿, 就写入 0;		
最后保存所有 CH 所有 Group 补偿值	":001w_offset_save\n"	一定要确保配置完全 OK 后, 再保存, 掉电不丢失
以上是写入 offset 补偿值的操作, 每个 CH 都支持最多 8 组 Group 的 Lux/xy 补偿值, 即使补偿过系数, 如果不调用任何一组补偿值, 对读取的 LUX/xy 就没有任何影响; 如果要在后续正常测量流程中调用某一组 Group 参与运算, 还需要一个使能选择指令 w_offset_en;		

如果要在正常测量过程中，调用某一通道的某一组的 offset 补偿值参与运算，那就需要在读取数据之前(r_chroma/r_Yxy/r_lux/r_xy 等指令)，发送 w_offset_en 选择使能某一组：

功能描述	指令格式(C 代码)	说明
假设读取 CH1-CH4 的 White LED 的数据	":001w_target_type01_04=0\n"	CH1-CH4 都选择模式 0 读取白光
	":001w_offset_en01_04= 1 \n"	第一个"01"是 CH1，"04"是结束通道 CH4 " =1 "是使能选择第一组 Group 的补偿值参与运算
	":001r_chroma01_04\n"	读取 CH1-CH4 的亮度和色度值
	":001w_offset_en01_04=0\n"	再禁用 Disable 任何一组补偿值，这条不是必须的
假设读取 CH1-CH4 的 Red LED 的数据	":001w_target_type01_04=6\n"	CH1-CH4 都选择模式 6 读取 Red 红光
	":001w_offset_en01_04= 2 \n"	" =2 "是使能选择第二组 Group 的补偿值参与运算
	":001r_chroma01_04\n"	读取 CH1-CH4 的亮度和色度值
	":001w_offset_en01_04=0\n"	再禁用 Disable 任何一组补偿值，这条不是必须的
在提前配置 w_offset 时，也要先写入对应 w_target_type 模式，读取未加补偿值的 Lux/xy，除非 w_target_type 都一样且不需要修改		

---1 转 3 光纤如何 offset 并测量双色灯

测量场景：仪器一共 16 个 CH，每个 CH 插入一根 1 转 3 光纤采光头，每根采光头测量 1 颗白红双色 LED，即每个 CH 要测量 6 组 LED 的亮度 Y(Lux)和色坐标 xy；

数据要求：所有 CH 测量 White 白灯的 Lux/xy 保持一致，所有 CH 测量 Red 红灯的 Lux/xy 保持一致；假设

仪器的 ID=1，实现编程逻辑如下：

在正常读取数据之前，调试机台时，先要对每个 CH 的 3 根采光头的白红双色 LED 的 Lux/xy 进行补偿 offset，即每个 CH 要补偿 3 组的 White 和 3 组 Red LED，一个通道有 6 组 Group 需要补偿；假定三颗白灯(White)补偿到 Group1/2/3，三颗红灯(Red)补偿到 Group4/5/6；这里规定，在读取仪器原始数据前，白灯用模式 0 读取("001w_target_type01_16=0\n"),红灯用模式 6 读取("001w_target_type01_16=6\n");		
功能描述	指令格式(C 代码)	说明
第一次配置时，建议清除仪器内部所有的补偿系数，以免内部原来就有配置过系数；	":001w_offset_clear\n"	所有 CH 所有组别 Group 的 kl=1.0, dxy=0.0, en=0;
对 CH1 第一根光纤的 White-Lux 写入系数 k=1.1	":001w_offset_kl01_01=1.1\n"	第一个 01 是 CH 编号，第二个 01 是第 1 组 Group1
对 CH1 第二根光纤的 White-Lux 写入系数 k=1.2	":001w_offset_kl01_02=2.2\n"	K=Lux_target_w/Lux_read_w
对 CH1 第三根光纤的 White-Lux 写入系数 k=1.3	":001w_offset_kl01_03=3.3\n"	Lux_target_w 是白光对标 Lux
.....CH1~CH16.....		Lux_read_w 是未加系数前从仪器读取 Lux (White LED 要提前配置为 w_target_type=0 在去读数据)
对 CH1 第一根光纤的 Red-Lux 写入系数 k=1.4	":001w_offset_kl01_04=1.4\n"	01 是 CH 编号,04 是第 4 组 Group4
对 CH1 第二根光纤的 Red-Lux 写入系数 k=1.5	":001w_offset_kl01_05=1.5\n"	
对 CH1 第三根光纤的 Red-Lux 写入系数 k=1.6	":001w_offset_kl01_06=1.6\n"	格式 ":001w_offset_kl[CH]_[Group]=xxx\n"
.....CH1~CH16.....		(Red LED 要提前配置为 w_target_type=6 在去读数据)
Lux 补偿值范围是 0.001~32.0,如果 kl 不参与补偿，就写入 1；如果 kl>32.0，那就需要对全局 lux 写入一个较大系数 k，指令"w_k_lux",先增大 lux 数值,然后再计算这个 kl；仪器通道有多个 lux 系数，都是相乘关系，出厂默认都是 1.0；		
对 CH1 第一根光纤的 White-xy 写入补偿值 dx=-0.011, dy=0.011	":001w_offset_dx01_01=-0.011\n" ":001w_offset_dy01_01=0.011\n"	第一个 01 是仪器 CH 编号， 第二个 01 是第一个组别 Group1，
对 CH1 第二根光纤的 White-xy 写入补偿值 dx=-0.021, dy=0.021	":001w_offset_dx01_02=-0.021\n" ":001w_offset_dy01_02=0.021\n"	dx=x_target_w-x_read_w; dy=y_target_w-y_read_w;
对 CH1 第三根光纤的 White-xy 写入补偿值 dx=-0.031, dy=0.031	":001w_offset_dx01_03=-0.031\n" ":001w_offset_dy01_03=0.031\n"	x(y)_target_w 是白灯的 xy 目标值 x(y)_read_w 是白灯在未补偿前的 xy 读取值
.....CH1~CH16.....		(White LED 要提前配置为 w_target_type=0 在去读数据)
对 CH1 第一根光纤的 Red-xy 写入补偿值 dx=-0.012, dy=0.012	":001w_offset_dx01_04=-0.012\n" ":001w_offset_dy01_04=0.012\n"	第一个 01 是仪器 CH 编号， 第二个 04 是第 4 个组别 Group4，
对 CH1 第二根光纤的 Red-xy 写入补偿值 dx=-0.022, dy=0.022	":001w_offset_dx01_05=-0.022\n" ":001w_offset_dy01_05=0.022\n"	
对 CH1 第三根光纤的 Red-xy 写入补偿值 dx=-0.032, dy=0.032	":001w_offset_dx01_06=-0.032\n" ":001w_offset_dy01_06=0.032\n"	":001w_offset_dx[CH]_[Group]=xxxx\n" ":001w_offset_dy[CH]_[Group]=xxxx\n"
.....CH1~CH16.....		(Red LED 要提前配置为 w_target_type=6 在去读数据)
xy 补偿值范围是-0.9~0.9,如果 xy 不参与补偿，就写入 0；		
最后保存所有 CH 所有 Group 补偿值	":001w_offset_save\n"	一定要确保配置完全 OK 后，再保存，掉电不丢失
以上是写入 offset 补偿值的操作，每个 CH 都支持最多 8 组 Group 的 Lux/xy 补偿值，即使补偿过系数，如果不调用任何一组补偿值，对读取的 LUX/xy 就没有任何影响；如果要在后续正常测量流程中调用某一组 Group 参与运算，还需要一个使能选择指令 w_offset_en;		

如果要在正常测量过程中，调用某一通道的某一组的 offset 补偿值参与运算，那就需要在读取数据之前(r_chroma/r_Yxy/r_lux/r_xy 等指令)，发送 w_offset_en 选择使能某一组：

功能描述	指令格式(c 代码)	说明
同时读取 CH1 到 CH16 的 第一根 光纤 White LED 的数据	":001w_target_type01_16=0\n"	CH1-CH16 都选择模式 0 读取白光
	":001w_offset_en01_16= 1 \n"	第一个"01"是 CH1，"16"是结束通道 CH16 " =1 "是使能选择 Group1 的补偿值参与运算
	":001r_Yxy01_16\n"	读取 CH1-CH16 的 Lux 和 xy 值
	":001w_offset_en01_16=0\n"	=0 禁用 Disable 任何一组补偿值，这条不是必须的
同时读取 CH1 到 CH16 的 第一根 光纤 Red LED 的数据	":001w_target_type01_16=6\n"	CH1-CH16 都选择模式 6 读取 Red 红光
	":001w_offset_en01_04= 4 \n"	" =4 "是使能选择第 Group4 的补偿值参与运算
	":001r_Yxy01_04\n"	读取 CH1 到 CH16 的 LUX 和 xy
	":001w_offset_en01_16=0\n"	=0 禁用 Disable 任何一组补偿值，这条不是必须的
同时读取 CH1 到 CH16 的 第二根 光纤 White LED 的数据	":001w_target_type01_16=0\n"	CH1-CH16 都选择模式 0 读取白光
	":001w_offset_en01_16= 2 \n"	第一个"01"是 CH1，"16"是结束通道 CH16 " =2 "是使能选择 Group2 的补偿值参与运算
	":001r_Yxy01_16\n"	读取 CH1-CH16 的 Lux 和 xy 值
	":001w_offset_en01_16=0\n"	=0 禁用 Disable 任何一组补偿值，这条不是必须的
同时读取 CH1 到 CH16 的 第二根 光纤 Red LED 的数据	":001w_target_type01_16=6\n"	CH1-CH16 都选择模式 6 读取 Red 红光
	":001w_offset_en01_04= 5 \n"	" =5 "是使能选择第 Group5 的补偿值参与运算
	":001r_Yxy01_04\n"	读取 CH1 到 CH16 的 LUX 和 xy
	":001w_offset_en01_16=0\n"	=0 禁用 Disable 任何一组补偿值，这条不是必须的
同时读取 CH1 到 CH16 的 第三根 光纤 White LED 的数据	":001w_target_type01_16=0\n"	CH1-CH16 都选择模式 0 读取白光
	":001w_offset_en01_16= 3 \n"	第一个"01"是 CH1，"16"是结束通道 CH16 " =3 "是使能选择 Group3 的补偿值参与运算
	":001r_Yxy01_16\n"	读取 CH1-CH16 的 Lux 和 xy 值
	":001w_offset_en01_16=0\n"	=0 禁用 Disable 任何一组补偿值，这条不是必须的
同时读取 CH1 到 CH16 的 第三根 光纤 Red LED 的数据	":001w_target_type01_16=6\n"	CH1-CH16 都选择模式 6 读取 Red 红光
	":001w_offset_en01_04= 6 \n"	" =6 "是使能选择第 Group6 的补偿值参与运算
	":001r_Yxy01_04\n"	读取 CH1 到 CH16 的 LUX 和 xy
	":001w_offset_en01_16=0\n"	=0 禁用 Disable 任何一组补偿值，这条不是必须的
在提前配置 w_offset 时，也要先写入对应 w_target_type 模式，读取未加补偿值的 Lux/xy，除非 w_target_type 都一样且不需要修改		

七：编程注意事项-必看！

1，配置参数保存 save_to_flash

出厂默认配置基本可以满足大部分测量场合，相关参数都是可以程控配置的，配置完成后都是可以选择保存到 flash 或 eeprom，掉电不丢失，但不建议频繁保存该指令，次数太多会擦坏内部 flash 或 eeprom；

2，增益 gain 和采样时间 ft

其中 w_gain 和 w_ft 指令是可以配置光学传感器的 ADC 采集增益和采样时间，

rgbw/r_adc/r_spect 等指令可以读取最原始的 ADC 值，相同光强下，w_gain 写入参数越大，ADC 值越大，w_ft 写入参数越大，ADC 值越大，但不能让 ADC 值饱和，ADC 是否饱和要看 r_rgbi 中的 i(0%-100%)来判断，一旦 ADC 饱和，那么测量的光学数据将失真，但 gain 和 ft 太小导致 ADC 过小，又会降低光学数据的分辨率；一般来说，i 的区间取 1%-60%是最合适；Gain 和 ft 基本不影响 lux 的数值，只是改变了 lux 的分辨率；

可以在初始化设置一次 gain 和 ft，或者直接提前配置保存好，掉电不丢失；

3，采样模式 w_system_samp:

连续模式 0(默认模式):仪器内所有通道会按照指定的 FT 时间自动刷新数据，采样起始时间不受上位机软件控制，LED 点亮后，要上位机软件延时 2 倍的 ftms 时间再去读取数据，才能保证是在 LED 点亮后更新出来的完整光强数据，这个好处在于，接口通信发送读取指令后，仪器立即返回数据，不用再等待一个采样时间周期后收到仪器返回的数据，可以连续发送多条读取指令，读取同一时刻测量的不同的光学参数；

单次模式 1:仪器内所有通道会等待上位机指令软件触发采集，LED 点亮后，上位机可以理解发送数据读取指令，仪器收到指令后会立即开始采集数据，等待 FT 时间后，再发送采集到的数据；这个好处在于，在只读取一条光学数据时，可以缩短数据获取时间；

可以在初始化设置一次模式，或者直接提前配置保存好，掉电不丢失；

4，LED 光源类型选择 w_target_type

这条指令的目的是提高光学参数的准确度，虽然每种模式都可以测量任何光源，但选择合适的类型，可以大大提高光学数据的准确度，一些低版本的产品不支持(LTD/HSI(RGB)系列)；如果是多色灯，可以在读取光学数据之前实时发送对应的类型，如果是单色灯，可以在初始化设置一次，或者直接提前配置保存好，掉电不丢失；

5，频率流水灯捕获指令 flick/flow:

需要关注亮灭的阈值 w_flick_limit，和采样时间 ft，limit 设置在亮灭亮度值的中间偏低位置，ft 一定要小于 LED 闪烁周期的一半以内，ft 越小，周期分辨率越高，但 ADC 越小，在不影响光学数据分辨率的情况下，尽量减小 ft；一旦捕获指令执行完成后，在频闪期间捕获的数据，都会一直暂存于内存中，可以随时去读，直到下次从新触发捕获指令后，才会被更新改变；

6，r_rgbi 与 r_chroma 颜色识别指令:

r_rgbi 指令中的 rgb 是颜色比例值，并不代表亮度信息，i 是仅代表相对光强度值，rgb 也没有任何光度参考标准，只能定性判断 LED 颜色和亮度，一般以待测合格样品的 rgbi 为标准值，比如一个红光的 rgb(255,100,30),r=255，但 gb 并不会等于 0；

r_chroma 中的参数全部是国际照明协会规定的 CIE-1931 的色度值，可以定量判断 LED 的色度和亮度，具体参数代表的含义，需要提前了解一些最基本的光度学概念；

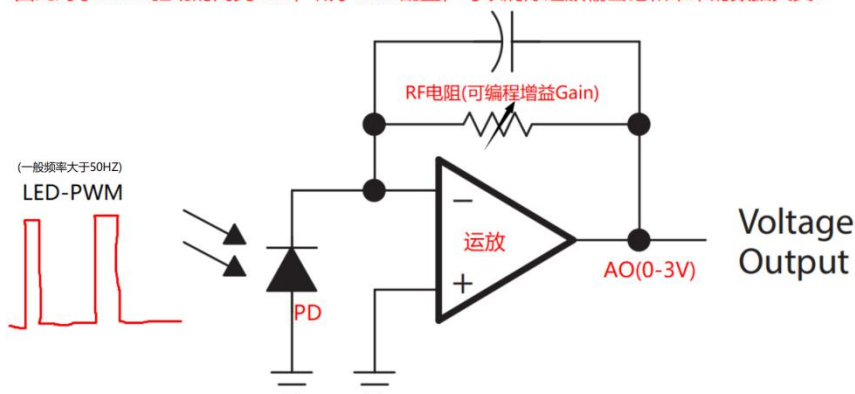
7，Net 通信说明:

支持网口通信的 LED 分析仪，底层是 TCP/IP 硬件协议层，但顶层应用层的通信指令还是和串口通信指令完

全一样，因此本文档上的字符串通信指令全部兼容网口通信指令；

8, 关于 PWM 驱动高亮 LED 测量说明:

即使PWM占空比很小，平均光强不大，只要LED瞬时点亮期间光强足够大，也能导致运放输出饱和！
因此对于PWM驱动的高亮LED，减小Gain配置，可以消除运放输出饱和带来的数据失真！



9, 单个 CHL 测多个 LED(多色 LED)时如何补偿光学参数:

方式 1: 提前装 LED 测试仪官方调试软件，可以进行 WB 白平衡模式的简易二次校准，版本 V23.101 及以后的可同时为单一 CHL 校准 8 组颜色亮度，一键校准，操作非常简单，可参看 `w_target_type` 指令查看逻辑；

方式 2: 如果没有预装官方调试软件，可以采用开放的 `offset` 指令，对每个通道进行多达 8 组的补偿值，具体指令逻辑参看 `offset` 相关指令；

备注: 两种方式都可以实现任意通道任意颜色亮度的 LED 光学数据校准到任何数值；

八: 手册声明:

我司 LED 测量系列产品开发资料会不定期更新，我们会尽可能兼容之前的协议，如有差错，请以最新编程手册作为准；开发资料编写如发现错误，欢迎指正修改；如有疑问，请及时联系我司技术人员协助解答；最终解释权归我司所有！