# HanOpticSens-LED Serial Communication Protocol-V23.111

| Vre | writer | Updated date | note |
|---|---|---|---|
| V19.6 | NEWTON | 2019-06-06 | |
| V23.111 | NEWTON | 2023-11-11 | Multiple instructions were added |
| At the bottom is the list of instructions，New version is compatible with the old version | | | |

## First，Communication overview

1,The USB and RS485 interfaces follow RS232 hardware protocols: 8b-byte, 1b-stop, and none;

2,The communication protocol of USB and RS485 interface is the same, and the protocol format is MODBUS-ASCII;

## Second，Instruction format

0, Suppose the current module's address is "001", and there is only one module in the system;

1, the host sends the instruction format: ": 001******\r\n" --- ("+" + ID "+" function instruction + "\r\n");

2, the module returns the instruction format: ": 001 ###### \ r \ n" --- (":" + "ID" + "return information" + "\ r \ n");

3, where ":" is the starting character, "\r\n" is the end character (at least one terminator is' \n '), and ID is a 3 bit integer (%03d),There is no space;

4, all instructions are character string (ASCII), no CRC check;

5, when the instruction sent is wrong, the module returns uniformly: "001ERR_CMD\r\n"";

6, ID= "000" is the broadcast address, and all modules will respond;

7, RS485 communication, send and read to be reserved bus conversion time (> 2ms);

## Third，Instruction detail

1, **query the module status command:** ask whether the module is in idle or busy;

Send instruction format: ": 001state\n"

Return instruction format: ": 001idle\r\ n" or ": 001busy\r\n";

Instruction Explanation: When the module is busy, the module does not execute the new instruction;

2, **query module status instructions:** query module model information;

Send instruction format: ": 001idn \ r \ n"

Return instruction format: ": 001LBB_RS********* \ r \ n" (depending on the module feedback information)

Instruction Explanation: The returned instruction is based on the actual character;

**3, write the module ID:** (the parameters immediately take effect, automatically saved in the flash)

Send instruction format: "000w_id=002\r\n" or ": 001w_id=002\r\n""

Return instruction format: ": 002w_id = 002 \ r \ n";

Instruction Explanation: After the ID is written successfully, the return instruction immediately uses the new ID to return the information;

**4, read the module ID:**

Send instruction format: ": 000r_id\r\ n" or ": 001r_id\r\n"

Return instruction format: ": 001r_id = 001\r\n";

Instruction Explanation: The module returns its current ID = "001";

**5, write the baud rate of the module (baud):** (the parameters immediately take effect, automatically saved in the flash)

Send instruction format: ": 001w_baud1 = 6\r\n" or ": 001w_baud2 = 6\ r\ n"

Return instruction format: ": 001w_baud1 = 6\r\n" or ": 001w_baud2 = 6\ r\n"

Instruction Explanation: baud1 represents the baud rate of USB-RS232 interface, baud2 represents the baud rate of RS485 interface;

Module first in accordance with the original baud return instructions, and then configure their own baud;

The value following the equal sign is the baud rate number (0-9), the default value is 6;

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

**6, read the module baud rate:**

Send instruction format: ": 001r_baud1 \r\n" or ": 001r_baud2\ r\ n"

Return instruction format: ": 001r_baud1 = 6 \r\n" or ": 001r_baud2 = 6\ r\ n"

Instruction Explanation: baud1 represents the baud rate of the USB-RS232 interface, baud2 represents the baud rate of the RS485 (RS232) interface and returns the baud rate number (0-9) of the interface;

**7, write the gain value of each channel:** (the parameters immediately take effect, temporarily stored in RAM)

Send instruction format: ": 001w_gain01-08 = 1\ n"

Return instruction format: ": 001w_gain01-08 = 1\ n"

Instruction Explanation: "01-08" represents from 01 (%02d) channels to 08 (%02d) channels, which must be from small to large, or equal (the maximum channel is 8 (16));

The value following the equal sign is the gain value number (0-3), the default value is 1, Gain[4]={X1, X4, X16, X64}, the greater the gain, the greater the read RGBW data;

**8, read the gain of each channel:**

Send instruction format: ": 001r_gain03-06 \ n"

Return instruction format: ": 001r_gain = 1,1,1,1,\ n"

Instruction Explanation: "03-06" represents the channel from 03 (% 02d) to 06 (% 02d), must be from small to large, or equal (channel maximum 8 (16));

The value following the equal sign is the gain number of the 03-06 channel;

**-- Read the gain of each channel (r_gainx) :**

Send instruction format: ":001r_gainx01-02\r\n"

Return instruction format: ":001r_gainx=5,1,\r\n"

Instructions explained:

The value after "=" is the gain value of channel 01-02 in turn, which corresponds to the written gain index number. The actual data is subject to the data returned by the instrument.

**9, write the samp time of each channel (exposure time) (ft):** (the parameters take effect immediately, temporarily in RAM)

Send instruction format: ": 001w_ft01-08 = 1\n"

Return instruction format: ": 001w_ft01-08 = 1\r\n"

Instruction Explanation: "01-08" represents from 01 (%02d) channels to 08 (%02d) channels, which must be from small to large, or equal (the maximum channel is 8 (16));

The value following the equal sign is the gain value number (0-4), the default value is 1; in the strobe test, according to the actual frequency size select the appropriate ft;

ft[5]={2.5ms，25ms，100ms，150ms，600ms}；The larger the sampling time, the greater the read RGBW data;

**10, read the samp time of each channel (sampling period) (ft):**

Send instruction format: ": 000r_ft03-06\n" or ": 001r_ft03-06\n"

Return instruction format: ": 001r_ft = 1,1,1,1,\ r\ n"

Instruction Explanation: "03-06" represents the channel from 03 (% 02d) to 06 (% 02d), must be from small to large, or equal (channel maximum 8 (16));

The value following the equal sign is the ft number of the 03-06 channel;

-- **Read the true sampling time (r_ftms) for each channel:**

Send instruction format: ":001r_ftms01-02\r\n"

Return instruction format: ":001r_ftms=20,123,\r\n"

Instructions explained:

The value after "=" is the real ms time of 01-02 channel ft, which corresponds to the ft index number written. The actual data is subject to the data returned by the instrument, which has reference significance for strobe testing.

**11, configure ADC sampling mode of the system :**

(this parameter takes effect immediately and is temporarily stored in RAM. It can also be saved without power loss)

Send instruction format: ": 001w_system_samp =1\n"

Return instruction format: ": 001w_system_samp =1\n"

Instruction interpretation:

The value after "=" is the working mode of ADC_samp, 0: Continuous sampling mode, 1: Single sample;

0: All channels are in synchronous continuous sampling mode, and the data update cycle is equal to the sampling time (FT);

However, no matter how big FT is, when the module receives the serial port instruction, it will immediately return the data, but the data of the previous sampling period is returned.

Therefore, attention should be paid when reading: The LED lighting time must be more than twice FT before reading data, so as to ensure that the data read after LED lighting is stable;The advantage of this instruction is that the serial port returns data without waiting, which can save the waiting time of the upper computer.If the LED is always bright or the test time is not high, it is recommended to use this mode.

1: Once reading instruction is sent, the sensor immediately takes a new sample of data, and the module serial port can only return the collected data after waiting for FT time;

Although the serial port waits, the short pulse light source can be measured: after the LED is lit, it is immediately sent for a read. As long as the LED is lit for a longer time than FT, stable data of LED lighting can be read.

**12, Read the ADC sampling mode of the system:**

Send instruction format: ": 001r_system_samp\n"

Return instruction format: ": 001r_system_samp =0\n"

Instruction interpretation:

The value after "=" is the working mode of ADC_SAMP,

0: Continuous sampling mode (default), 1: Single sample;

**13,System reset initialization**

Send instruction format: ": 001w_system_reset\n"

Return instruction format: ":001w_system_reset\n"

Instruction interpretation:

Let the product reset when it is powered on, but not restore factory Settings;

**14,Save the parameters to flash:** (the command takes effect immediately)

Send instruction format: ": 001save_to_flash\ n"

Return instruction format: ": 001save_to_flash\ r\ n";

Instructions Explanation: the route, gain, ft, d_lower and other parameters saved to flash, power loss is not lost;

**15,Restore factory default settings:** (the command takes effect immediately)

Send instruction format: ": 001default\ n"

Return instruction format: ": 001default\ r\ n";

Instructions Explanation: the id, baud, route, gain, ft, d_lower and other parameters to restore the factory default, and save to flash, power loss is not lost;

/ / / / / / / / / / / / / / / **RGB color data - suitable for RGB series products /** / / / / / / / / / / / / / / / / / /

**-- Read sensor Raw Data ADC data (rgbw) :**

Send instruction format: ":001rgbw01-02\r\n"

Return instruction format: ": 001 RGBW = 123234345678111, 23112, 34113, 45226, 78, \ r \ n"

Instructions explained:

The value after "=" is "r(%d),g(%d),b(%d),w(%d)," of the 01-02 channel, which belongs to the original ADC data of the sensor. The ratio of rgbw represents the color, and the absolute value represents the relative intensity. In actual programming, it is not recommended to use this data as a judgment standard, but the data can reflect whether the working state of the sensor is reasonable, the data is too small, close to the lower limit of sensor sensitivity, resulting in inaccuracy, too large may cause data saturation and data distortion;

The best working range is 1%-60% distance from data saturation ratio. Changing the gain or ft parameter changes the Raw Data;

**-- Read RGBI(U8) color intensity data (r_rgbi) :**

Send instruction format: ":001r_rgbi01-01\r\n"

Return instruction format: ":001r_rgbi=255,244,105,50.00\r\n"

The value after "=" is "r(%d),g(%d),b(%d),i(%0.2f)," for channel 01-01; The RGB range is 0-255;

I is the percentage relative to the maximum measured intensity (0.00-100.00), I can be used as the relative light intensity data,However, under the same light intensity, changing the Gain parameter will change the value of I; When I is close to 100.00%, the sensor is saturated.The RGBI is only the relative component value of color brightness. There is no standard to refer to, and it can only be compared with consistency.

There are still some differences between the RGB data principle of the camera RGB and the RGB data principle of the display picture RGB, which cannot be completely equivalent.

**-- Read HSLI data for each channel (r_hsli) :**

Send instruction format: ":001r_hsli01-02\r\n"

Return instruction format: ": 001 r_hsli,80,40,10.01 = 300, 300,80,40,10.01, \ r \ n"

Instructions explained:"=" values, in turn, is behind the channel 01-02 "H (% 0.0 f), S (% 0.0 f), L (% 0.0 f), I (% 0.2 f),";H(0-360), S(0%-100%), L(0%-100%), I(0.00%-100.00%),

I is the percentage relative to the maximum measured intensity. When I is close to 100%, it means that the sensor is saturated, which is the same as i in r_rgbi.The HSLI is only the relative component value of color brightness, there is no standard to refer to, only consistency relative comparison, where L is the same as V of HSV species.

There are still some differences between the HSL of the camera and the HSL of the display picture, and the data principle of the HSL is not completely equivalent.

**-- Read visible light intensity (brightness/illumination/lumen) data (r_lux) :**

Send instruction format: ":001r_lux01-02\r\n"

Return instruction format: ":001r_lux=123.12,234.12,\r\n"

Instructions explained:

The value after "=" is the illuminance value of channel 01-02 in the form of floating point string %f.

In products measuring illuminance, this value is in lx; In lumen series instruments, the unit is lm; In the optical fiber series products, this value represents a reference value proportional to the LED luminous intensity, and requires secondary calibration to obtain the true illuminance/brightness/lumen value.----read the non-visible intensity of each channel intensity (uw / cm ^ 2) data:

Send instruction format: ": 001**r_uw_cm**01-08 \ r \ n"

Return instruction format: ": 001r_uw_cm = 123,234,345,678,11123,11234,11345,22678, \ r \ n"

Instruction Explanation: "01-08" represents from 01 (%02d) channels to 08 (%02d) channels, which must be from small to large, or equal (the maximum channel is 8 (16)); The value of the equal sign is followed by the light intensity value of the 01-08 channel. This data is only used for light intensity judgment. This instruction is only applicable to non-visible light measurement module;

////////////////////////**Chroma data**//////////////////////////////////////

-- Select the color type of the LED to be tested (**w_target_type**) :

Send instruction format: "**:001w_target_type01-02=0\r\n**" (effective immediately, temporarily stored in RAM)

Return instruction format: "**:001w_target_type01-02=0\r\n**"

Instruction explanation:

The value after "=" is the type number (0-30), the default value is 0, any mode can read out the chromaticity value, but in order to obtain more accurate luminosity parameters, you need to choose the appropriate mode; Only XYZ and above series products support this directive;

**0:** White balance mode WB0, any light source can use this mode, but not necessarily the best configuration, more suitable for white light;

**3:** green light (500nm-560nm) or white LED(2000k-50000K), defined according to the instrument model;

**4:** Single-core lamp beads "orange (about 600nm)" or "yellow-green (567nm)" or "cyan (about 500nm)", And any of the two or three core mixed lights, such as the network port orange/yellow green indicator;

**5:** RGB light :Red(about 630nm),Green(about 525nm),Blue(about 461nm), Monochrome lights and any combination of two or three of them; For example, RGB atmosphere lights;

**6:** Red monochrome light, the main wavelength between (600-641nm) red light;

//// Configurations numbered 1,2 and greater than 9 are supported by versions 2023-5-1 and later than V23.033

**1:** White balance mode WB1, more suitable for low color warm white light;

**2:** White balance mode WB2, more suitable for high color temperature cold white light;

**10:** AutoWB mode, is an automatic white balance mode, the instrument will automatically match the color of the LED, select the most reasonable white balance coefficient to participate in the calculation, the matched white balance coefficient is limited to the number of 11~16 range;

**11:** RedWB mode, the factory default is a mode dedicated to measuring red light;

**12:** GreenWB mode, the factory default is to measure the mode of green light;

**13:** BlueWB mode, the factory default is dedicated to measuring blue light mode;

**14:** YellowWB mode, the factory default is dedicated to measuring yellow light mode;

**15:** C/R_WB mode, the factory default is dedicated to measuring the green light (Cyan), some instruments are super Red (Red) mode;

**16:** WhiteWB mode, the factory default is to measure the white light mode, the white balance coefficient is equal to the number of 0 coefficient;

Note: When measuring some special light sources, some of these modes will be occupied, and the supporting product manual shall prevail;

Before the V23.101 version, the instruction can only change the chroma data, but in the V23.101 version and later, you can change the brightness and chroma at the same time, especially suitable for WB calibration operation of chroma brightness using official debugging software;

**-- Read the target_type of LED to be tested for each channel:**

Send instruction format: ": 001r_target_type01-02 \n"

Return instruction format: ":001r_target_type=0,0\n"

Instruction interpretation:

The value after "=" is the number of the LED type configuration;

**-- Read r_chroma data (V1.5 and later)**

Send instruction format: ":001**r_chroma**01-01\r\n"

Return instruction format: ": 001 r_chroma = 1000.0, 0.3333, 0.4444, 555.5, 85.2, 6500,0.00123, \ r \ n"

Command to explain: "=" values, in turn, is behind the 01-01 channel "lux (% 0.1 f), x (% 0.4 f), y (% 0.4 f), dowave (% 0.1 f), duty (% 0.1 f), CCTS (% 0.0 f), fd (% 0.5 f),"

This instruction can efficiently obtain the LED chromaticity data;

lux=1000.0, same as r_lux instruction;

x=0.3333,y=0.4444, same as r_xy, is CIE1931 xy coordinate;

dowave=555.5nm,duty=85.2%, similar to r_wavesi;

cct=6500K, similar to the r_cct instruction;

fd=0.00123, this floating point number is a reserved parameter, and the value is different in different products. After V24.011, the default is i in r_rgbi,i is the saturation degree of the sensor, but when SDCM mode measurement is enabled, fd is SDCM data (V20.101 and later are supported);

**-- Read R_wavesi ( domianwavelength + saturation + lux) data:**

Send instruction format: ": 001r_wavesI01-01 \n"

Return instruction format: ": 001r_wavesi =555.5,99.9,123.4,\n"

Instruction interpretation:

The following values of "=" are in turn "(%0.1f)," (%0.1f), "of 01-01 channel, 555.5 represents the domain wavelength(unit nm), 99.9% represents saturation, and the range is 0%-100%;123.4 is the illumination, the unit of LX, which is the same as the instruction R_LUX to read the data;

**-- Read the Yxy data of each channel:**

Send instruction format: ": 001R_Yxy01-02 \n"

Return instruction format: ": 001 r_Yxy = 323.5, 0.2345, 0.3145, 678.5, 0.5234, 0.1434, \ n"

Command to explain: "=" values, in turn, is behind the channel "01-02 Yxy (f %), (f %), (f %),", Y is equal to the illuminance value Lux, xy is color coordinates;

**--- read each channel xy (CIE-1931 color coordinates) data:**

Send instruction format: ": 001r_xy001-02 \ r \ n"

Return instruction format: ": 001r_xy = 0.3333,0.4333,0.3666,0.3111, \ r \ n"

Instruction Explanation: "01-02 represents the channel from 01 (% 02d) to 02 (% 02d), which must be from small to large, or equal (channel maximum 8 (16)); The value following the equal sign is the value of the 01-02 channel "x (0.4F%), y (0.4F%)," which represents the CIE-1931 chromaticity coordinates;

**--- read the UV (CIE-1976 color coordinate) data for each channel:**

Send instruction format: ": 001r_uv001-02 \ r \ n"

Return instruction format: ": 001r_uv = 0.3333,0.4333,0.6666,0.1111, \ r \ n"

Instruction Explanation: "01-02 represents the channel from 01 (% 02d) to 02 (% 02d), which must be from small to large, or equal (channel maximum 8 (16)); The value following the equal sign is the value of the 01-02 channel "u (0.4F%), v (0.4F%)," which represents the CIE-1976 chromaticity coordinates;

**--- read the CCT (color temperature) data for each channel:**

Send instruction format: ": 001r_cct01-02 \ r \ n"

Return instruction format: ": 001r_cct = 5438,6457, \ r \ n"

Instruction Explanation: "01-02 represents the channel from 01 (% 02d) to 02 (% 02d), which must be from small to large, or equal (channel maximum 8 (16)); The values following the equal sign is the value of the 01-02 channel "CCT (d%), (d%)," the CCT data is calculated from the CIE-1931 (xy) chromaticity coordinates;

**-- Write the color tolerance type (w_sdcm_type) :**

Send instruction format: ":001w_sdcm_type01-02=0\r\n"

Return instruction format: ":001w_sdcm_type01-02=0\r\n"

Instruction explanation: (effective immediately, temporarily stored in RAM, only applicable to XYZ and above V20.101)

The value after "=" is the type number (0-99), the default is 0,disable,

| MODE | Functional Description | REMARK |
|---|---|---|
| 0 | Disabale | The SDCM measurement feature is not enabled |
| 1 | reservation | |
| 2 | AUTO-SDCM_ErpF | |
| 3 | AUTO-SDCM_ANSI | |
| 4 | AUTO-SDCM_customized | Customizable |
| 5,6,7,8,9 | reservation | |
| 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,) | |

**-- Read the type of color tolerance (r_sdcm_type) :**

Send instruction format: ":001r_sdcm_type01-02\r\n"

Return instruction format: ":001r_sdcm_type=0,0\r\n"

Instructions explained:

The value after the "=" is the number of the sdcm type configuration;

**-- Read the color tolerance data (r_sdcm_data) :**

Send instruction format: ":001r_sdcm_data01-02\r\n"

Return instruction format: ":001r_sdcm_data=3.2,2.3\r\n"

Instructions explained:

The value after "=" is the sdcm data of channel 01-02 with 0.1f% accuracy;

**-- Read the color tolerance + illumination data (r_sdcm_lux) :**

Send instruction format: ":001r_sdcm_lux01-01\r\n"

Return instruction format: ":001r_sdcm_lux=100.1,3.2,21\r\n"

Instructions explained:

After "=" is the data of 01 channel, 100.1 is the lux illuminance data, 3.2 is the SDCM value, 21 is the light source reference type ANSI 3000K of SDCM;

/ / / / / / / / / / / / / / / / / / / / **flick--flick-flick/** / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /

**-- write the reference channel mode to the flick threshold :**

(this parameter takes effect immediately and is stored temporarily in RAM)

Send instruction format: ":001w_flick_mode01-08=0\n"

Return instruction format: ":001w_flick_mode01-08=0\n";

Instruction interpretation:

| Mode | explain | remark |
|---|---|---|
| 0(default) | Lux | reference to the “r_lux” and “flick_limit” light on and off contrast capture stroboscopic |
| 1 | R_adc | The reference instruction “rgbw” is the original ADC data of the sensor. In order to improve the capture speed of the Flick, the calculation time of each channel can be saved by about 0.2ms compared with mode0. One of the ADC values is used to compare flick_limit on and off to capture the stroboscope. |
| 2 | G_adc | |
| 3 | B_adc | |
| 4 | W_adc | |
| 5~9 | reserve | |
| 10 | Hue+Lux | Referring to the hue H in r_hsli, when comparing colors, compared with mode12,13 can save 0.1ms of calculation time |
| 11 | Satura+Lux | Referring to the color saturation S in r_hsli, compared with mode12,13 can save 0.1ms in the calculation of color comparison |
| 12 | Dowave_Lux | Refer to the main wavelength wave in r_chroma, that is, wave is used as the reference value of color comparison with “flick_color_min” and “flick_color_max” are tested to capture the frequency |
| 13 | CCT+Lux | Reference the color temperature CCT in r_chroma, that is, CCT as the reference value of color comparison with “flick_color_min” and “flick_color_max” , and then capture the frequency |
| When color is involved in comparison, the speed of each channel will be increased by 0.2ms compared to mode0, which is suitable for brand capture with a single multicolor light flashing alternately; CVersions with numbers greater than 4 are supported by versions 2023-5-1 and later than V23.033 | | |

**-- Reference channel mode for reading flick threshold:**

Send instruction format: ": 001r_flick_mode01-02 \n"

Return instruction format: ":001r_flick_mode=0,0\n";

Instruction interpretation:

The range of schema indexes after "=" is 0-9(%d);LUX / 1:R /2:G /3:B /4:W;

**-- write to the flick threshold :**

(this parameter takes effect immediately and is stored temporarily in RAM)

Send instruction format: ": 001w_flick_limit01-08 =10\n"

Return instruction format: ": 001w_flick_limit01-08 =10\n";

The threshold range after "=" is 0-1000000(%d), must be greater than 0, the default value is 20;

When the selected data collected (see Flick_mode) is less than this value, the light is considered to be off; when it is greater than this threshold, the light is on, which is the threshold for judging the strobe test, and each channel can be configured independently;

**-- Read the minimum threshold of LED indicator light:**

Send instruction format: ": 001r_limit_limit01-02 \n"

Return instruction format: ":001r_flick_limit=20,20,\n";

Instruction interpretation: "=" is followed by the returned threshold (0-1000000:%d);

**-- Write the upper color threshold of flick (w_flick_color_max) :**

Send instruction format: ":001**w_flick_color_max**01-08=1\r\n" (This parameter takes effect immediately and temporarily exists in RAM)

Return instruction format: ":001w_flick_color_max01-08=1\r\n"; (V23.033 or later)

Instruction explanation: "=" after the threshold is floating point (%f), if it is dowave mode, support positive and negative, the default value is 360;

When the selected data collected (see flick_mode) is greater than w_flick_color_min and less than w_flick_color_max, the LED light of the color is on; otherwise, the LED of the color is in the off state. This value is the color threshold for judging the strobe test, and each channel can be configured independently.

**-- Write the lower limit of the flick color threshold (w_flick_color_min) :**

Send instruction format: ":001**w_flick_color_min**01-08=1\r\n" (This parameter takes effect immediately and temporarily exists in RAM)

Return instruction format: ":001w_flick_color_min01-08=1\r\n"; (V23.033 or later)

Instruction explanation: "=" after the threshold is floating point (%f), if it is dowave mode, support positive and negative values, the default value is 0;

When the selected data collected (see flick_mode) is greater than w_flick_color_min and less than w_flick_color_max, the LED light of the color is on; otherwise, the LED of the color is in the off state. This value is the color threshold for judging the strobe test, and each channel can be configured independently.

**-- Read flick's upper color threshold (r_flick_color_max) :**

Send instruction format: ":001**r_flick_color_max**01-02\r\n" (V23.033 or later)

Return instruction format: ":001r_flick_color_max=360,360,\r\n";

Instruction explanation: "=" followed by the returned threshold, support floating-point, if it is dowave mode, support positive and negative;

**-- Read flick's lower color threshold (r_flick_color_min) :**

Send instruction format: ":001**r_flick_color_min**01-02\r\n" (V23.033 or later)

Return instruction format: ":001r_flick_color_min=0,0,\r\n";

Instruction explanation: "=" followed by the returned threshold, support floating-point, if it is dowave mode, support positive and negative;

**-- Start LED flashing frequency function:**

Send instruction format: ": 001w_flick_ts01-02 =09\n"

Return instruction format: ": 001w_flick_ts01-02 =09\n";

Instruction interpretation: the value range of "09" is 01-99 seconds. Set the sampling time length in seconds.

The hardware parameters of the corresponding channel (FT&Gain &w_flick_limit) must be set in advance. After this instruction is issued, the module will perform strobe test immediately, lasting 9 seconds.During the test, the module is busy, but a "state\n" instruction can be sent to query the status of the module, and a read instruction can only be sent when the module returns "idle".The module supports simultaneous strobe testing of all channels.The measurement results will be kept in RAM until the next instruction is issued and updated;

**-- flick frequency cycles :(data is maintained until the next test update)**

Send instruction format: ": 001r_flick_ts01-02 \n"

Instruction interpretation: read the stroboscopic test results of 01-02 channel;

Return instruction format: ":001r_flick_ts=1.00,990,1000,500,5, 2.00,490,501,100,8\n";

Instruction interpretation:

Description of face value after "=" (frequency: 1.00Hz, bright-light interval: 990ms, switch-off interval: 1000ms, duration: 500ms, number of light pulses: 5)

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

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

**Capture simple flow function:**

Send instruction format: ":001w_flick_flow01-16=03\r\n" // only suitable for sequential lighting of LED water detection, such as car flow water turn signal

Instruction interpretation: At the same time start 1-16chl flow water capture function, capture time is 3 seconds;

Return instruction format: ":001w_flick_flow01-16=03\r\n";

Instruction explanation: "=03" value range is 01-50 seconds, set the capture time length, the unit is second;

The hardware parameters of the corresponding channel (ft&gain&w_flick_limit) must be set in advance. After the instruction is issued, the module immediately executes the flow test, and the test time lasts for 3s(set value). During the test, the module does not respond to other commands, but it can send the state command "to query the module status. When the module returns" idle ", new read and write commands can be sent. The module supports simultaneous flow testing of all channels; The measurement results will remain in the RAM until the next command is updated. After receiving this instruction, it will immediately return ":001w_flick_flow01-16=03\n", and then perform flow measurement;

When multiple modules RS485 measure multiple flow lights in parallel, you can send the address to 0, such as sending ":000w_flick_flow01-16\n" instruction, and start multiple instrument capture at the same time, because it is 485 parallel, the instrument will respond to broadcast address 0 at the same time. Only the ID=1 instrument will return the normal value of ":000w_flick_edge01-16\n", other ID instruments do not reply to any instructions, so as not to cause bus conflict;

**-- Read the LED flow water light off time stamp (r_flick_flow) : //V20.123 version**

Send instruction format: "**:001r_flick_flow01-02\r\n**" (data will be kept until the next test update)

Instruction interpretation: Read the flow test results of the 01-02 channel;

Return instruction format: ": 001 r_flick_flow = 1000150, 0500105, 0145, 0400, \ r \ n";

Instruction explanation:

The literal value after "=" is ms, all integers, the number of returned data CNT=CHL*3;

1000: represents the time 1CH was first lit,

1500: The time when 1CH is first extinguished after being lit,

500: indicates the reserved parameter. The tentative time difference is 1500-1000=500.

1050:2CH first lit time,

1450: The time when 2CH is first extinguished after being lit,

400: indicates the reserved parameter. The tentative time difference is 1450-1050=400.

From the data point of view, the lighting time of 2CH is 1050-1000=50ms later than that of 1CH, but the lighting time of 2CH is 50ms earlier than that of 1CH (1450-1500 = -50ms). The water lamp is lit from 1 to 2, and then extinguished in reverse order;

The time reference 0 point of measurement is 0ms after sending w_flick_flow start instruction and the instrument returns w_flick_flow;

**--LED edge capture function (w_flick_edge) : //V21.051 version**

Send instruction format: "**:001w_flick_edge01-16=03\r\n**"

Instruction interpretation: Start 1-16chl edge capture function at the same time, capture time is 3 seconds;

Suitable for multiple times of water or multiple flashing occasions, such as car welcome lights, water turn signals, is the most common instruction in stroboscope;

Return instruction format: ":001w_flick_edge01-16=03\r\n";

Instruction explanation: "=03" value range is 01-55 seconds, set the capture time length, the unit is second;

The hardware parameters of the corresponding channel should be set in advance (ft&gain&w_flick_limit). After the instruction is issued, the module will immediately perform edge test, and the test time will last for 3s(set value). During the test, the module does not respond to other commands, but can send the state command to query the module status (the query frequency is too fast, which will reduce the underlying collection speed).

After the module returns "idle", new read and write commands can be sent. The module supports simultaneous edge testing of all channels; The measurement results will remain in the RAM until the next command is updated.

After receiving this command, it will immediately return ":001w_flick_edge01-16=03\n", and then perform flow measurement;

When multiple modules RS485 are measuring multiple flow lights in parallel, the address can be changed to 0, such as sending ":000w_flick_edge01-16\n" instruction, and multiple instrument capture can be started at the same time. Since it is 485 in parallel, the instrument will respond to broadcast address 0 at the same time. Only instruments with ID=1 will return the normal value "001w_flick_edge01-16\n", other instruments with ID do not reply to any instructions, so as not to cause bus conflict;

**-- Read the LED multiple on-off edge timestamp (r_flick_edge) : //V21.051 version**

Send instruction format: "**:001r_flick_edge**01-02=2\r\n" (data will be kept until the next test update)

Instruction explanation:

Read the edge test results of channel 01-02. "=2" is the time stamp for each channel to read the EDGE on and off (2 times), up to 10 times;

The number of returned data CNT is equal to the number of channels CHL multiplied by the number of edges EDGE, then multiplied by 2(rising EDGE and falling edge), CNT=CHL*EDGE*2;

The time reference 0 point of measurement is 0ms after sending w_flick_flow start instruction and the instrument returns w_flick_flow;

Return instruction format: ": 001 r_flick_edge = 1000110, 0120, 0130, 0200, 0210, 0220, 0230, \ r \ n";

Instruction explanation:

The value after "=" is ms, all integers, the first 4 data is 1CHL timestamp, the last 4 data is 2CHL timestamp,

1000:1CHL The time of the first light,

1100:1CHL light after the first time to go out,

1200:1CHL second lighting time,

1300:1CHL second light and then extinguished again time,

2000:2CHL first lit time,

2100: The time when 2CHL is first extinguished after being lit,

2200:2CHL second lighting time,

2300:2CHL second lit and then extinguished again time,

**-- Read LX data of LED flicker: //(the data is kept until the next test update)**

Send instruction format: ":001r_flick_lx01-02\n"

Return instruction format: ":001r_flick_lx=123,234,\n"

Instruction interpretation:

"=" followed by "lx1(%0.0f),lx2(%0.0f)" of the 01-02 channel, in turn;This data is the maximum value of Lux during LED lighting, which can reflect the illumination of LED (brightness);

**-- Read r_flick_Chroma (chroma) data lit by strobe on each channel: (data remains until next test update)**

Send instruction format: ": 001r_flick_chroma01-01 \n"

Return instruction format: ": 001 r_flick_chroma = 1000.0, 0.3333, 0.4444, 555.5, 85.2, 6500,0.00123, \ n"

Instruction interpretation:

Value, in turn, is behind the "=" 01-01 channel "lux (% 0.1 f), x (% 0.4 f), y (% 0.4 f), dowave (% 0.1 f), duty (% 0.1 f), CCT (% 0.0 f), fd (% 0.5 f),"

This instruction can get the color data of LED efficiently.

Lux =1000.0, same as r_Lux instruction;

X =0.3333,y=0.4444, the same as R_xy, is the XY coordinate of CIE1931;

Dowave =555.5nm, Duty =85.2%, similar to r_wavesi;

CCT =6500K, similar to r_CCT instruction;

Fd =0.00123, the floating-point number is a reserved parameter,  the default is the i%(r_flick_rgbi);

**-- Read r_flick_RGBI data: (data remains until next test update)**

Send instruction format: ": 001r_flick_rgbi01-01 \n"

Return instruction format: ": 001r_flick_rgbi =255,244,105,50.00\n"

"=" is followed by "r(%d),g(%d), B (%d), I (%0.2f)," of the 01-01 channel.The RGB range is 0-255;

I is the percentage relative to the maximum intensity (0.00-100.00), I can be used as the data of relative brightness,But under the same light intensity, the change of Gain parameter will change the value of I.When I=100.00%, it means that the sensor has been fully exposed;

This instruction has the same reading effect as r_RGBI instruction;

**---Read the flick's hsli data (r_flick_hsli) :**

Send instruction format: ":001r_flick_hsli01-01\r\n" (V23.033 - data will be kept until the next test update)

Return instruction format: ":001r_flick_hsli=3,94,50,26.8,\r\n"

Instructions explained:

Value, in turn, is behind the "=" 01-01 channel "h (% d), s (% d), l (% d), I (% 0.3 f),", h is tonal (0-360), s is color purity (0-100), l is color bright degree ((0-100) generally don't have to), I value can represent the size of the LED light intensity; The same as the r_hsli instruction reads the result;

**-- read Raw Data when LED flashes :(Data is kept until the next test update)**

Send instruction format: ": 001r_flick_rgbc01-01 \n"

Return instruction format: ": 001r_flick_rgbc =123,234,345,678,\n"

Instruction interpretation:

Value, in turn, is behind the "=" 01-01 channel "r (% d), g (% d), b (% d), w (% d)", belong to the original sensor RGBW ADC data;This data is the maximum value of RGBW during LED lighting. The value can represent the light intensity of LED and the ratio can qualitatively judge the color of LED.

Same as “rgbw” instruction read result;

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

**-- Read the on-off state of multi-channel LEDS // To quickly determine the on-off state of leds**

Send instruction format: ": 001r_led_chl01-02 \n"

Return instruction format: ":001r_led_chl=0,1,\n"

Instruction interpretation:

Send: "01 represents 01(% 02d) CHL1;

Return: 1 for light, 0 for extinguish;

You can select one of the intensities data to compare with the limit_flick data. The comparison rules are the same as the flick logic, as shown in the” Flick_mode” directive;

**-- Write the on-off state of LED emitting light source // only applicable to products with LED emitting port**

Send instruction format: “:001w_led_disp01-01=1\n”

Return instruction format: “:001w_led_disp01-01=1\n”

Instruction interpretation:

Send: "01 represents 01(% 02d) LED output channel CH1;

**-- Read the on-off state of LED emitting light source // Only applicable to products with LED emitting port**

Send instruction format: “:001r_led_disp01-02\n”

Return instruction format: ":001r_led_disp=0,1,\n"

Instruction interpretation:

Send: "01 represents 01(% 02D) LED output channel CHL1;

Return: 1 for light, 0 for extinguish;

////////////////////////**DIO//**//////////////////////////

**---read single channel DI (only with DIO module can be used)**

Send instruction format: ": 001r_inbit01 \ r \ n"

Return instruction format: ": 001r_inbit = 0 \ r \ n" or ": 001r_inbit = 1 \ r \ n"

Instruction Explanation:

Send: "01 stands for 01 (% 02d) channel (DIN1);

Return: 1 on behalf of the input (optocoupler conduction), 0 represents no input (optocoupler without conduction);

**---once read 8 (16) channel DI** (only with DIO module can be used)

Send instruction format: ": 001r_in_u8 \ r \ n" or ": 001r_in_u16 \ r \ n"

Return instruction format: ": 001r_in_u8 = 128 \ r \ n" or ": 001r_in_u16 = 128 \ r \ n"

Instruction Explanation:

Send: "u8" on behalf of DIN1-DIN8 from low to high composed of uinit8 bytes of data (module default support

8DI); "U16" represents DIN1-DIN16 from low to high composition of uinit16 shaping data (reserved);

Return: DIN1-DIN8 in turn from low to high combination into a byte data, 128 (1000 0000) on behalf of DIN8 input, the other no input;

**----- write single-channel DO** (only with DIO module can be used)

Send instruction format: ": 001w_outbit01-02 = 1 \ r \ n" or ": 001w_outbit01-16 = 0 \ r \ n"

Return instruction format: ": 001w_outbit01-02 = 1 \ r \ n" or ": 001w_outbit01-16 = 0 \ r \ n"

Instruction Explanation:

Send: "01-02 represents the channel from 01 (% 02d) to 02 (% 02d), must be from small to large, or equal (channel maximum 16); The "1" on the back of the equal sign represents DO1-DO2, and "0" represents the closed channel DO1-DO16

Return: if the instruction is normal, the return value is the same as the sending character;

**-----write 16CHL channel DO** (only with DIO module can be used)

Send instruction format: ": 001w_out_u16 = 4 \ r \ n"

Return instruction format: ": 001w_out_u16 = 4 \ r \ n"

Instruction Explanation:

Send: "u16" on behalf of DO1-DO16 from low to high composition of the uinit16 plastic data; The "4" (0000 0000 0000 0100) after the equal sign indicates that DO3 is turned on and all other channels are closed;

Return: if the instruction is normal, the return value is the same as the sending character;

/////////////////////////**net Communication configuration**/////////////////////

///// Instruments after February 2022 support the net directive -V22.011 and later ////////

**-- Write net working mode (w_net_mode)**

Send command format: ":001w_net_mode=1\r\n"

Return command format: ":001w_net_mode=1\r\n"

net_mode The value ranges from 0:Disable,1:TCP_Server,1:TCP_Client,3:UDP.

The default is to turn off the network port, once the network port function is turned on, the instrument initialization will have 5 seconds of network connection time, if you need network port communication, it is recommended that the LED analyzer work in 1:TCP_Server mode; Before the boot network port, all net parameters are configured by USB/RS485/RS232 serial port protocol, and all net parameters can be saved by save_to_flash command.

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

**-- Read the working mode of net (r_net_mode)**

Send command format: ":001r_net_mode\r\n"

Return command format: ":001r_net_mode=1\r\n"

**-- Source IP address of the instrument written to net (w_net_sip)**

Send command format: ":001w_net_sip=192,168,0,100\r\n"

Return instruction format: ":001w_net_sip=192,168,0,100\n"

sip is the IP address of the network port of the LED analyzer, and all ip addresses of the network meet the IPV4 format;

To be compatible with other parsing instructions, the IP address number interval is separated by a comma (', '), not a dot ('. ');

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

**-- Source IP address of the instrument reading net (r_net_sip)**

Send instruction format: ":001r_net_sip\r\n"

Return instruction format: ":001r_net_sip=192,168,0,100,\r\n"

**-- Target IP address of the host computer writing net (w_net_dip)**

Send command format: ":001w_net_dip=192,168,0,10\r\n"

Return instruction format: ":001w_net_dip=192,168,0,10\r\n"

Send instruction explanation: dip is the IP address of the network port of the upper computer,

The dip configuration can be used only when the LED analyzer network port works in TCP_Client mode.

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

**-- Target IP address of the host computer reading net (r_net_dip)**

Send command format: ":001r_net_dip\r\n"

Return instruction format: ":001r_net_dip=192,168,0,10,\r\n"

**-- Write the Gateway IP address of net (w_net_gip).**

Send instruction format: ":001w_net_gip=192,168,0,1\r\n"

Return instruction format: ":001w_net_gip=192,168,0,1\r\n"

Send instruction explanation: gip is the Gateway IP address of the network port,

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

**-- Read the Gateway IP address of net (r_net_gip).**

Send instruction format: ":001r_net_gip\r\n"

Return instruction format: ":001r_net_gip=192,168,0,1,\r\n"

Write net Subnet Mask IP address (w_net_mip).

Send command format: ":001w_net_mip=255,255,255,0\r\n"

Return instruction format: ":001w_net_mip=255,255,255,0\r\n"

Send instruction explanation: gip is the Gateway IP address of the network port,

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

Read the Subnet Mask. Subnet Mask IP address (r_net_mip).

Send instruction format: ":001r_net_mip\r\n"

Return instruction format: ":001r_net_mip=255,255,255,0,\r\n"

**-- Source port number of LED instrument written to net (w_net_sport)**

Send instruction format: ": 001 w_net_sport = 8000800 1800 2800 3800 4800 5800 6800 7 \ r \ n"

Return instruction format: ": 001 w_net_sport = 8000800 1800 2800 3800 4800 5800 6800 7 \ r \ n"

Send instruction explanation: sport is the LED analyzer network port number, in TCP_Server mode, support 8 simultaneous communication; In TCP_Client mode, only the first port number (8000) is used to access the host server.

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

**-- Read source port number of nnet LED instrument (r_net_sport)**

Send command format: ":001r_net_sport\r\n"

Return instruction format: ": 001 r_net_sport = 8000800 1800 2800 3800 4800 5800 6800 7, \ r \ n"

**-- Target port number of the host server writing to net (w_net_dport)**

Send command format: ":001w_net_dport=10000\r\n"

Return instruction format: ":001w_net_dport=10000\r\n"

Sending instruction explanation: dport is the port number when the PC host is used as the TCPServer, and the port number is only effective when the LED analyzer is in TCP_Client mode.

Return instruction explanation: If the instruction is normal, the return value is the same as the sent character;

**-- The target port number of the host server for reading net (r_net_dport)**

Send instruction format: ":001r_net_dport\r\n"

Return instruction format: ":001r_net_dport=10000\r\n"

**-- Read the MAC address of net (r_net_mac)**

Send instruction format: ":001r_net_mac\r\n"

Return instruction format: ":001r_net_mac=101,102,103,104,105,106,\r\n"

mac format is also returned in accordance with decimal string, not in accordance with hexadecimal string format;

**-- Read all configuration parameters of net (r_net_all)**

Send instruction format: ":001r_net_all\r\n"

Return instruction format: "... \r\n"

Instruction explanation: This instruction is suitable for reading network configuration information manually under the serial debugging assistant.

After the network port function is enabled, the LED analyzer will print out all the configuration information of the network port from the USB interface when it is reset and initialized;

**-- Test run the net network port communication connection (w_net_run)**

Send command format: ":001w_net_run\r\n"

Return instruction format: ":001w_net_run\r\n"

Instruction explanation: After the network cable is plugged in, after the net parameter configuration is completed, you can send the instruction to test run the network connection. After the instruction is sent, the D9 indicator light on the LED analyzer will blink continuously for 4 seconds to connect to the network, during which the USB interface will automatically send the network parameter information to the upper computer, and the orange and green lights on the RJ45 will also be lit. Once the network connection is normal, it indicates that the configuration parameters are correct. Then send ":001save_to_flash\r\n" to save it. After the network port is powered on, the network port function will be automatically enabled.

**/ / // configuration instructions - V23.031 WhiteBalance coefficient and later supports 20230301 / / / / / /**

It is recommended to use the official debug software to configure whitebalance coefficients simply and efficiently. It is not recommended to configure with this directive;

**--Write the whitebalance coefficient of Red for a few channels (wr_whitebalance)**

Send instruction format: ":001**wr_whitebalance**01-02=0.5,11\r\n"

Return instruction format: ":001wr_whitebalance01-02=0.5,11\r\n"

"wr" is the abbreviation of Write Red, "01-02" is the 1th ch to 2th ch, "=0.5" is the white balance coefficient Kr=0.5, coefficient range is 0.001~30 recommended maximum coefficient in Krgb write 1.0; The last ",11 "refers to the WB number 11 in the w_target_type instruction, which is in the range 0, 1, 2, 11, 12, 13,14,15,16. If the last number is not written, or is written incorrectly, the number is forced to be 0; When an instruction is received normally, the same instruction is returned.

White balance coefficient can be simply understood as multiplying the original rgb or XYZ value of the sensor by a coefficient, which is more practical for single-point calibration of LED.

**-- Write the Green whitebalance for certain channels (wg_whitebalance)**

Send instruction format: ":001wg_whitebalance01-02=0.5,11\r\n"

Return instruction format: ":001wg_whitebalance01-02=0.5,11\r\n"

Instruction explanation: "wg" is the abbreviation of Write Green, "01-02" is the 1th ch to 2th ch, "=0.5" is the white balance coefficient Kg=0.5, coefficient range is 0.001~30 recommended maximum coefficient in Krgb write 1.0; The last ",11 "refers to the WB number 11 in the w_target_type instruction, which is in the range 0, 1, 2, 11, 12, 13,14,15,16. If the last number is not written, or is written incorrectly, the number is forced to be 0; When an instruction is received normally, the same instruction is returned.

**-- Write Blue's whitebalance coefficient for certain channels (wb_whitebalance)**

Send instruction format: ":001wb_whitebalance01-02=0.5,11\r\n"

Return instruction format: ":001wb_whitebalance01-02=0.5,11\r\n"

Instruction explanation: "wb" is the abbreviation of Write Blue, "01-02" is the 1th ch to 2th ch, "=0.5" is the white balance coefficient to write R Kb=0.5, coefficient range is 0.001~30, the maximum coefficient in Krgb is recommended to write 1.0; The last ",11 "refers to the WB number 11 in the w_target_type instruction, which is in the range 0, 1, 2, 11, 12, 13,14,15,16. If the last number is not written, or is written incorrectly, the number is forced to be 0; When an instruction is received normally, the same instruction is returned.

**--Writing the Lux luminance coefficients of WBxx for a few channels (wl_whitebalance)**

Send instruction format: ":001wl_whitebalance01-02=0.5,11\r\n"

Return instruction format: ":001wl_whitebalance01-02=0.5,11\r\n"

Instruction explanation: "wl" is the abbreviation of Write Lux, "01-02" is the 1th ch to the 2th ch, "=0.5" is the brightness coefficient of write Lux KL=0.5, coefficient range is 0.001~30; The last ",11 "refers to the WB number 11 in the w_target_type instruction, which is in the range 0, 1, 2, 3,4,5,6, 11, 12, 13,14,15,16. If the last number is not written, or is written incorrectly, the number is forced to be 0; When an instruction is received normally, the same instruction is returned. This directive only supports instrument V23.111 and later;

**-- Read the Lux luminance coefficients of WBxx for a few channels (rl_whitebalance)**

Send instruction format: ":001rl_whitebalance01-02=11\r\n"

Return instruction format: ":001rl_wb=0.5,0.6\r\n"

Instructions explained: "rl" is short for "Read Lux," 01-02 "is the 1CH to 2CH, the last" =11 "refers to the WB number 11 in the r_target_type instruction, the range is 0, 1, 2, 3,4,5,6, 11, 12, 13,14,15,16, if the last number is not written, or the writing error, then the number is forced to be 0; The return instruction is simplified, 0.5 is the KL coefficient of the 1CH and the 11th group, and 0.6 is the 11th group of the 2CH; This directive only supports instrument V23.111 and later;

**--Read the whitebalance of Red for a few channels (rr_whitebalance)**

Send instruction format: ":001rr_whitebalance01-02=11\r\n"

Return instruction format: ":001rr_wb=0.5,0.6\r\n"

Instructions explained: "rr" is short for "Read Red," 01-02 "is the 1CH to 2CH, the last" =11 "refers to the WB number 11 in the r_target_type instruction, the range is 0, 1, 2, 11, 12, 13,14,15,16, if the last number is not written, or the writing error, then the number is forced to be 0; The return instruction is simplified, 0.5 is the Kr coefficient of the 11th group of the 1CH, 0.6 is the Kr coefficient of the 11th group of the 2CH;

**--Read Green's whitebalance for certain channels (rg_whitebalance)**

Send instruction format: ":001rg_whitebalance01-02=11\r\n"

Return instruction format: ":001rg_wb=0.5,0.6\r\n"

Instructions explained: "rg" is short for "Read Green," 01-02 "is the 1CH to 2CH, the last" =11 "refers to the WB number 11 in the r_target_type instruction, the range is 0, 1, 2, 11, 12, 13,14,15,16, if the last number is not written, or the writing error, then the number is forced to be 0; The return command is simplified, 0.5 is the Kg coefficient of the 11th group of the 1CH, and 0.6 is the 11th group of the 2CH;

**--Read Blue's whitebalance for a few channels (rb_whitebalance)**

Send instruction format: ":001rb_whitebalance01-02=11\r\n"

Return instruction format: ":001rb_wb=0.5,0.6\r\n"

Instructions explained: "rb" is short for "Read Blue," 01-02 "is the 1CH to 2CH, the last" =11 "refers to the WB number 11 in the r_target_type instruction, the range is 0, 1, 2, 11, 12, 13,14,15,16, if the last number is not written, or the writing error, then the number is forced to be 0; The return instruction is simplified; 0.5 is the 1CH and 11th group Kb coefficient, and 0.6 is the 2CH and 11th group Kb coefficient.

**-- save the whitebalance (save_whitebalance)**

Send command format: ":001save_whitebalance\r\n"

Return instruction format: ":001save_whitebalance\r\n"

The white balance coefficient will take effect immediately after writing, so only when the configuration is completed, confirm that it can be measured normally, and then save it, so that the power is not lost; The storage time is generally long (non-fiber series instruments may exceed 8 seconds, fiber series host is generally within 3S), and the internal Flash has erase and write life limit, do not send the instruction frequently; This save directive is not the same as the save object for "save_to_flash";

**/ /// adjust the sensor raw rgb ADC data AutoWB proportional relationship - V23.031 and later suppor// / /**

//// Only configure this command if the default configured coefficients of the instrument do not meet the measurement requirements!! //////////

**--AutoWB coefficient of raw data written to Sensor (w_autowb_sens)**

Send command format: ":001w_autowb_sens02_k01=3.5,10\r\n"

Return instruction format: ":001w_autowb_sens02_k01=3.5,10\r\n"

Instruction interpretation: The 02 in "sens02" is the hardware number of the sensor, which can be read by the r_system_sens instruction, which has no relationship with the channel, only with the sensor hardware number; When an instruction is received, the same instruction is returned.

**-- AutoWB coefficients for reading raw data from Sensor (r_autowb_sens)**

Send command format: ":001r_autowb_sens02_k01\r\n"

Return instruction format: ":001r_autowb_sens02_k01=3.5,10,\r\n"

Instruction interpretation: 02 in "sens02" is the hardware number of the sensor, 01 in k01 is the first set of colors; The first set of color scale relations returned for sens=2 is K1=3.5, K2=10;

| Write (read) instructions | Substitute instruction | Notes |
|---|---|---|
| w_autowb_sens02_k01=3.5,10<br>(r_autowb_sens02_k01) | w_autowb_r_rg_rb02=3.5,10<br>(r_autowb_r_rg_rb02) | For SENS=2 sensor to write K1=r/g,K2=r/b two coefficients, ranging from 0.001 to 1000, the purpose is to write the proportion of screening group 1 (k01) color (Red light), the internal judgment logic in w_target_type=10(AUTOWB) mode is: When K1>3.5 and K2>10, it is identified as a red light, and the instrument will automatically substitute the white balance coefficient of RedWB(11) to participate in the calculation； |
| w_autowb_sens02_k02=10,2<br>(r_autowb_sens02_k02) | w_autowb_g_gr_gb02=10,2<br>(r_autowb_g_gr_gb02) | Write the proportion relationship of the second group (k02)Green light. When K1(g/r)>10 and K2(g/b)>2, it is identified as green light, and the instrument will |

| | | |
|---|---|---|
| | | automatically replace the white balance coefficient of GreenWB(12) to participate in the calculation. |
| w_autowb_sens02_k03=30,4 (r_autowb_sens02_k03) | w_autowb_b_br_bg02=30,4 (r_autowb_b_br_bg) | Write the proportion of Blue light in the third group (k03). When K1(b/r)>30 and K2(b/g)>4, it is identified as green light, and the instrument will automatically replace the white balance coefficient of BlueWB(13) to participate in the calculation; |
| 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) | Write the proportion of Blue light in the third group (k03). When K1(b/r)>30 and K2(b/g)>4, it is identified as green light, and the instrument will automatically replace the white balance coefficient of BlueWB(13) to participate in the calculation; |
| w_autowb_sens02_k05=10,10 (r_autowb_sens02_k05) | w_autowb_c_gr_br02=10,10 (r_autowb_c_gr_br02) | When K1(g/r)>10 and K2(b/r)>10, it is identified as a Cyan lamp, and the instrument will automatically replace the white balance coefficient of C/R_WB(15) to participate in the calculation； |
| In the mode of w_target_type=10(AUTOWB), when the first 5 groups of k01~k05 fail to match, it is judged as white light WhiteWB(16), and the instrument will automatically replace the white balance coefficient of WhiteWB(16) to participate in the calculation; If w_target_type! =10, then the scale factor will not be used in the contrast filter; The save instruction for autoWB-related parameters is also "save_to_flash"; | | |

## Programming considerations - must see!

**1, save the configuration parameter save_to_flash**

The factory default configuration can basically meet most of the measurement occasions, the relevant parameters can be programmed configuration, after the configuration is completed, you can choose to save to the flash eeprom, power is not lost, but it is not recommended to save the instruction frequently, too many times will wipe the internal flash or eeprom;

**2, gain and sampling time ft**

The w_gain and w_ft instructions are the ADC acquisition gain and sampling time of the optical sensor can be configured, and the instructions such as rgbw/r_adc/r_spect can read the original ADC value. Under the same light intensity, the larger the w_gain writing parameter is, the larger the ADC value is, and the larger the w_ft writing parameter is, the larger the ADC value is. Whether the ADC is saturated depends on i(0%-100%) in r_rgbi. Once the ADC is saturated, the measured optical data will be distorted. However, too small gain and ft will lead to too small ADC and reduce the resolution of optical data. Generally speaking, 1%-60% is the most appropriate interval for i;

Gain and ft do not affect the value of lux, but only change the resolution of lux.

gain and ft can be set at the initial time, or directly in advance to save the configuration, power is not lost;

**3, sampling pattern w_system_samp:**

Continuous mode 0(default mode): all channels in the instrument will automatically refresh data according to the specified FT time, and the sampling start time is not controlled by the host computer software. After the LED is lit, the host computer software must delay 2 times the ftms time to read the data, so as to ensure that the complete light intensity data updated after the LED is lit. After the interface communication sends the read instruction, the instrument returns the data immediately. Instead of waiting for a sampling time period to receive the data returned by the instrument, multiple read instructions can be sent continuously to read different optical parameters measured at the same time.

Single mode 1: All channels in the instrument will wait for the command software of the host computer to trigger the acquisition. After the LED is lit, the host computer can understand and send the data reading instruction. The instrument will immediately start collecting data after receiving the instruction, and then send the collected data after waiting for the FT time. This advantage is that the data acquisition time can be reduced when only one optical data is read.

The mode can be set once in the initialization, or directly in advance to save the configuration, power is not lost;

**4. Select w_target_type as the LED light source type**

The purpose of this instruction is to improve the accuracy of optical parameters. Although each mode can measure any light source, choosing the right type can greatly improve the accuracy of optical data, which is not supported by some lower versions of the product (LTD/HSI(RGB) series); If it is a multi-color lamp, the corresponding type can be sent in real time before reading the optical data. If it is a monochrome lamp, it can be set once in the initialization, or directly configured in advance and saved, and the power is not lost.

**5, frequency pipeline lamp capture command flick/flow:**

It is necessary to pay attention to the bright-off threshold w_flick_limit and the sampling time ft. The limit should be set at the lower middle position of the bright-off brightness value, ft must be less than half of the LED flicker period. The smaller ft is, the higher the cycle resolution is, but the smaller ADC is, ft should be reduced as far as possible without affecting the resolution of optical data. Once the capture instruction is executed, the data captured during the strobe period will be temporarily stored in the memory and can be read at any time. It will not be updated until the next time the capture instruction is triggered.

**6, r_rgbi and r_chroma color recognition instructions:**

r_rgbi instruction in rgb is the color ratio value, does not represent the brightness information, i is only represents the relative light intensity value, rgb does not have any photometric reference standard, only qualitative judgment LED color and brightness, generally qualified sample rgbi for the standard value, such as a red light rgb(255,100,30),r=255, But gb is not equal to 0;

The parameters in r_chroma are all the CIE-1931 chromaticity values specified by the International Lighting Association, which can quantitatively judge the chromaticity and brightness of LED. The specific meaning of the parameters needs to understand some of the most basic photometric concepts in advance.

**7, Net communication description:**

Support network communication LED analyzer, the bottom layer is TCP/IP hardware protocol layer, but the top layer of the application layer communication instructions and serial communication instructions are exactly the same, so the string communication instructions on this document are all compatible with network communication instructions;

8. **How to compensate the optical parameters when a single CHL measures multiple leds (multi-color leds) :**

Mode 1: Install the official debugging software of LED tester in advance, which can carry out a simple secondary calibration of WB white balance mode. Version V23.101 and later can simultaneously calibrate 8 groups of color brightness for a single CHL, and the operation is very simple. See w_target_type instruction for logic;

Mode 2: If the official debugging software is not pre-installed, the open offset instruction can be used to carry out up to 8 groups of compensation values for each channel. For the specific instruction logic, please refer to the offset related instruction.

Note: Both methods can achieve any channel and any color brightness of the LED optical data calibration to any value;

## Summary of instruction table:

| Serial port instruction table(id=001) | | | |
|---|---|---|---|
| Instruction classification | Instruction key string | Functional Description | remark |
| Status query | ":001state\n" | Returns ":001idle\r\n" or ":001busy\r\n" | |
| Module information | :001idn | Read module product information | |
| ID | :000r_id | Read the ID of the module, ID is the communication protocol software ID | |
| | :001w_id | Write the module ID(%03d), which takes effect immediately and saves the parameter automatically | |
| Baud | w_baud1 | Set USB/RS232 baud rate, effective immediately and automatically saved | Baud:0-9 |
| | w_baud2 | Set RS485 port baud rate, effective immediately and automatically saved | {2400,4800,9600,19200,38400,57600,115200,230400,460800,921600} |
| | | | |
| Configure the ADC sampling mode | w_system_samp | Configure ADC sampling mode 0: continuous mode; 1: Single sampling | |
| Read the ADC sampling pattern | r_system_samp | Read ADC sampling mode (V19.6 and later) | |
| system_reset | w_system_reset | The instrument is initialized from a fresh reset, not a factory reset | |
| Reply to factory Settings | default | All user-configured parameters restore factory Settings | |
| User parameter saving | save_to_flash | The user configuration in RAM parameters saved to flash, power is not lost, it is not recommended to | |

| | | | |
|---|---|---|---|
| | | save frequently | |
| Sensor NO. | r_system_sens | Read the instrument sensor hardware number | |
| Sampling period | w_ft | The sensor sampling period is set, and all channels are sampled synchronously, executed immediately, and temporarily stored in RAM | The larger the ft, the larger the adc data and the longer the sampling period |
| | r_ft | Read the sensor sampling time index number | |
| | r_ftms | Read sensor sampling time ms(V1.5 and later) | |
| ADC gain | w_gain | The sensor hardware gain is set, executed immediately, and temporarily stored in RAM | The larger the gain, the larger the ADC data |
| | r_gain | Read the sensor gain index number | |
| | r_gainx | Read actual sensor gain (V1.5 and later) | |
| | | | |
| RGB/HSL data (LTD series does not support color measurement function) | rgbw | Read the sensor RGBW raw ADC value | |
| | | | |
| | r_rgbi | Read RGB and intensity I(0-100%) converted to 0-255 range | |
| | r_hsli | Read chrominance/saturation/brightness/ intensity I(0-100%) | |
| Lux | w_k_lux | lux compensation coefficients are written, executed immediately, and temporarily stored in RAM | |
| | r_k_lux | Read lux compensation coefficient k(%0.3f), default 1.0 | |
| | w_k_uw | he optical power compensation coefficients are written, executed immediately, and temporarily stored in RAM | |
| | r_k_uw | The optical power compensation coefficient k(%0.3f) , default 1.0 | |
| | r_lux | Read the illumination value/brightness value/light intensity/lumen value | |
| | r_uw_cm | Reading optical power uw/cm^2, generally used to characterize non-visible light, | some products are not applicable |
| | r_cd_mm | Reading brightness cd/m^2, area light source brightness data, suitable for non-fiber products | (V19.3 and later) |
| Lumen lm | r_lm | Read the lumen value of the integral scout, which is proportional to lux | |
| cd/mcd | r_cd_lm | Read Candela cd and lumen of LED (for Lambertian light source) | |
| | | | |

| | | | |
|---|---|---|---|
| Flick | w_flick_mode | Set the LUX-R-G-B-W contrast data mode | immediate effect |
| | r_flick_mode | Read flick Contrast Channel mode -(V1.5 and later) | |
| | w_flick_limit | Set the flick brightness threshold, temporarily store in RAM | immediate effect |
| | r_flick_limit | Read the flick brightness threshold | |
| | w_flick_color_max | Set the upper limit of flick color, temporarily stored in RAM | immediate effect |
| | w_flick_color_min | Set the lower limit of flick color and temporarily store it in RAM | immediate effect |
| | r_flick_color_max | Read the flick color upper limit | |
| | r_flick_color_min | Read the flick color lower limit | |
| | w_flick_ts | Set up and immediately execute the flick_ts Test | V1.5 and later |
| | r_flick_ts | Read flick ts (Cnt, MAC, Tup, Tdw, Tduty, LUXmax) | |
| | r_flick_lx | Reading lux data when lit during flick | |
| | r_flick_chroma | Reading chroma data when lit during flick - | |
| | r_flick_rgbi | Reading rgbi data when lit during flick - | |
| | r_flick_rgbc | Reading RAWadc data when lit during flick - | |
| | r_flick_hsli | Reading hsli data when lit during flick - | |
| | w_flick_flow | Set up and immediately execute the flick_flow Test | V20.101 and later |
| | r_flick_flow | Read flick_flow data | |
| | w_flick_edge | Set up and immediately execute the flick_edge Test | V21.051 and later |
| | r_flick_edge | Read flick_edge data | |
| Read the LED on and off state | r_led_chl | uickly read whether each channel has led on and off | suitable for digital tube measurement |
| chroma chromaticity data (Only the data read by the XYZ module is accurate, In order to be compatible with the communication protocol, these instructions also apply to RGB series products, but the obtained data is not accurate.) | w_target_type | Select the type of the target LED to be tested, execute immediately, and temporarily store in RAM | Choosing an appropriate type can improve the measurement accuracy of color coordinates |
| | r_target_type | Read the current led test type | |
| | r_Yxy | Read (CIE1931) tristimulus color coordinates and luminance | Y=r_lux |
| | r_xy | Read xy(CIE1931) color coordinate data | |
| | r_uv | Read uv(CIE1976) color coordinate data | |
| | r_cct | Read CCT(CIE1931) color temperature data | |
| | r_cctd | Reading CCT and Duv data | (V1.5 and later) |
| | r_dowave | Read domain wavelength (CIE1931) data | |
| | r_wavesi | Read domain wavelength, color purity (color saturation), lux data | i=r_lux |

| | | | |
|---|---|---|---|
| | r_chroma | Read lux, x,y,dowave,duty%,cct,i% data; | |
| | w_sdcm_type | Select the sdcm type, effective immediately, temporarily stored in RAM | |
| | r_sdcm_type | Read the sdcm type of the current channel | |
| | r_sdcm_data | Read the sdcm data of the current channel | |
| | r_sdcm_lux | Read lux/sdcm/sdcm type of the current channel | |
| | | | |
| LED emission light source | w_led_disp | Turn the LED on and off, only suitable for products with LED transmitter port | |
| | r_led_disp | Read the status of the emission light source LED | |
| DIO read and write (Only with DIO modules can be used) | r_inbit | Read one bit input signal | Optocoupler input |
| | r_in_u8 | Read 8-bit input to form a byte | |
| | w_outbit | Write to the DO state at the specified address | ULN2803-NPN Transistor Output (200MA) |
| | w_out_u16 | Write the 16-bit DO output state | |
| Offset (V19.3 and later) | w_offset_en | Read and write certain group functions for certain channels | r_offset_en |
| | w_offset_dx | Read and write the offset dx parameter of cie1931-x | r_offset_dx |
| | w_offset_dy | Read and write the offset dy parameter of cie1931-y | r_offset_dy |
| | w_offset_kl | Read and write the offset kl parameter of lux | r_offset_kl |
| | w_offset_save | Save all offset parameters | |
| | w_offset_clear | Clear all offset related parameters | |
| Net network parameter configuration instructions - Applicable to new devices with LAN network port after 2022-Feb | w_net_mode | Read and write network port working mode | r_net_mode |
| | w_net_sip | Read and write the source ip address of the instrument | r_net_sip |
| | w_net_dip | Read and write the target ip address of the host computer, usually the ip of the computer | r_net_dip |
| | w_net_gip | Read and write the Gateway IP address of the network | r_net_gip |
| | w_net_mip | Read and write the Subnet Mask IP address of the network | r_net_mip |
| | w_net_sport | ead and write the source port number of the instrument network port. 8 ports are supported | r_net_sport |
| | w_net_dport | Read and write the host computer network port target port number, only one | r_net_dport |
| | r_net_mac | Read the MAC address of the network port | |
| | w_net_run | The software initializes the network port once, but the network parameters are not saved to flash | |
| | r_net_all | Read all network configuration information, only on the serial debugging assistant | |
| | **It is strongly recommended to use the official debugging software, configure net parameters through USB or RS485, convenient and save trouble!** | | |
| White balance | wr_whitebalance | wg_whitebalance | wb_whitebalance |

| | | | |
|---|---|---|---|
| coefficient read and write( V23.031)<br>The Lux coefficient corresponding to WBxx | rr_whitebalance<br>V23.111 版本及以后<br>w_autowb_sens<br>r_autowb_sens | rg_whitebalance<br>wl_whitebalance/rl_whitebalance<br>w_autowb_r_rg_rb<br>r_autowb_r_rg_rb | rb_whitebalance<br><br>w_autowb_g_gr_gb<br>r_autowb_g_gr_gb |

**Manual statement:**

The development information of our LED measurement series products will be updated from time to time. We will try our best to comply with the previous agreement. If there is any error, please refer to the latest programming manual. Welcome to correct and revise the development materials if you find any errors; If you have any questions, please contact our technical staff to help answer; The final right of interpretation belongs to our company!