

# EZO-RGB™

## Embedded Color Sensor

Reads

**RGB (24-bit)**  
**CIE (xyY)**  
**LUX (0 – 65535)**

Features

**onboard LEDs**  
**programmable color matching**

Connector

**5 lead data cable**

Response time

**1 reading per 400 milliseconds**

Sensing area

**15° half angle**

Cable length

**1 meter**

Water resistant/dust proof

**IP67**

Data protocol

**UART & I<sup>2</sup>C**

Default I<sup>2</sup>C address

**112 (0x70)**

Data format

**ASCII**

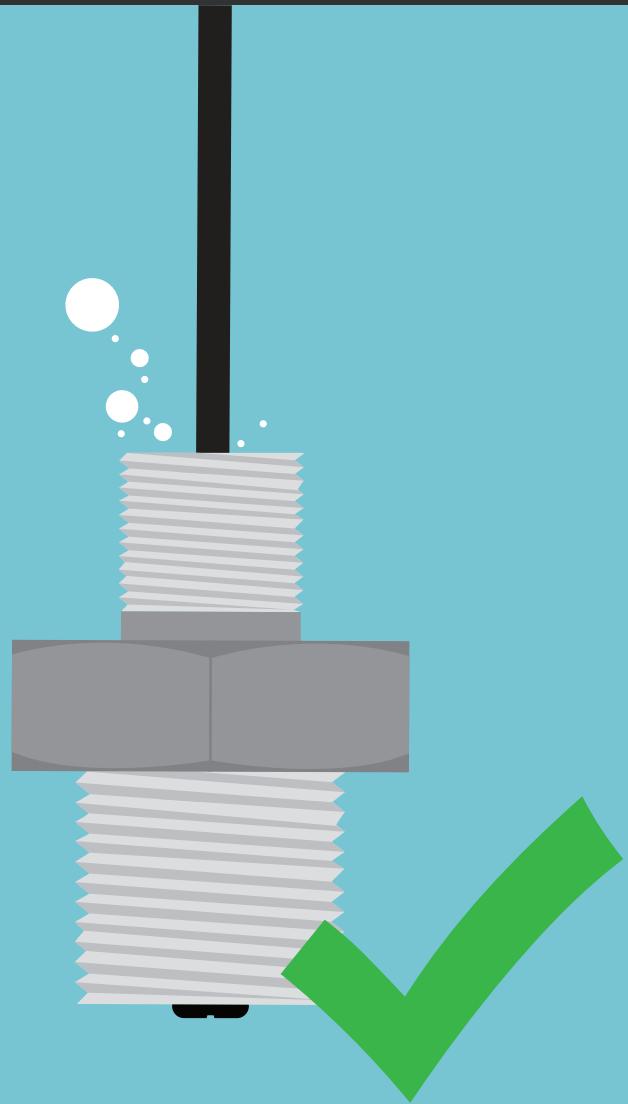
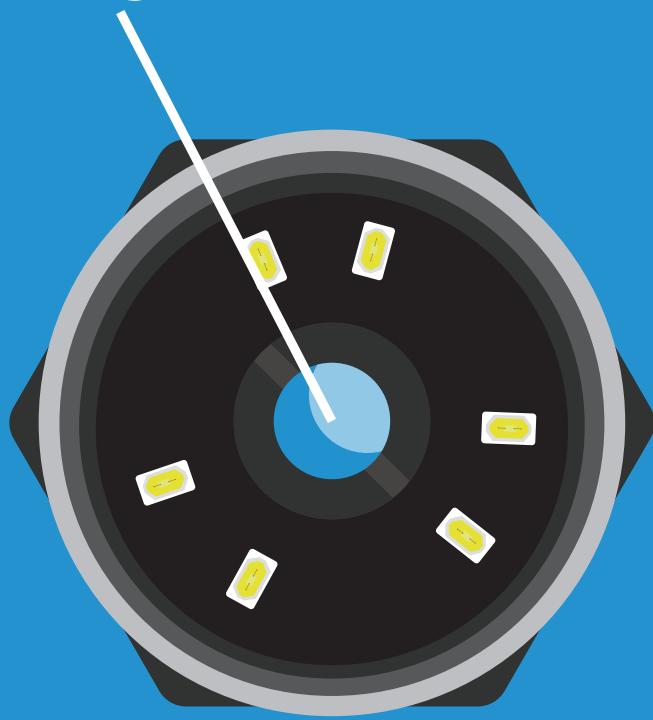
Operating voltage

**3.3V – 5V**

# New Feature

The EZO-RGB™ Embedded Color Sensor  
is now IP67 waterproof – up to 1 meter

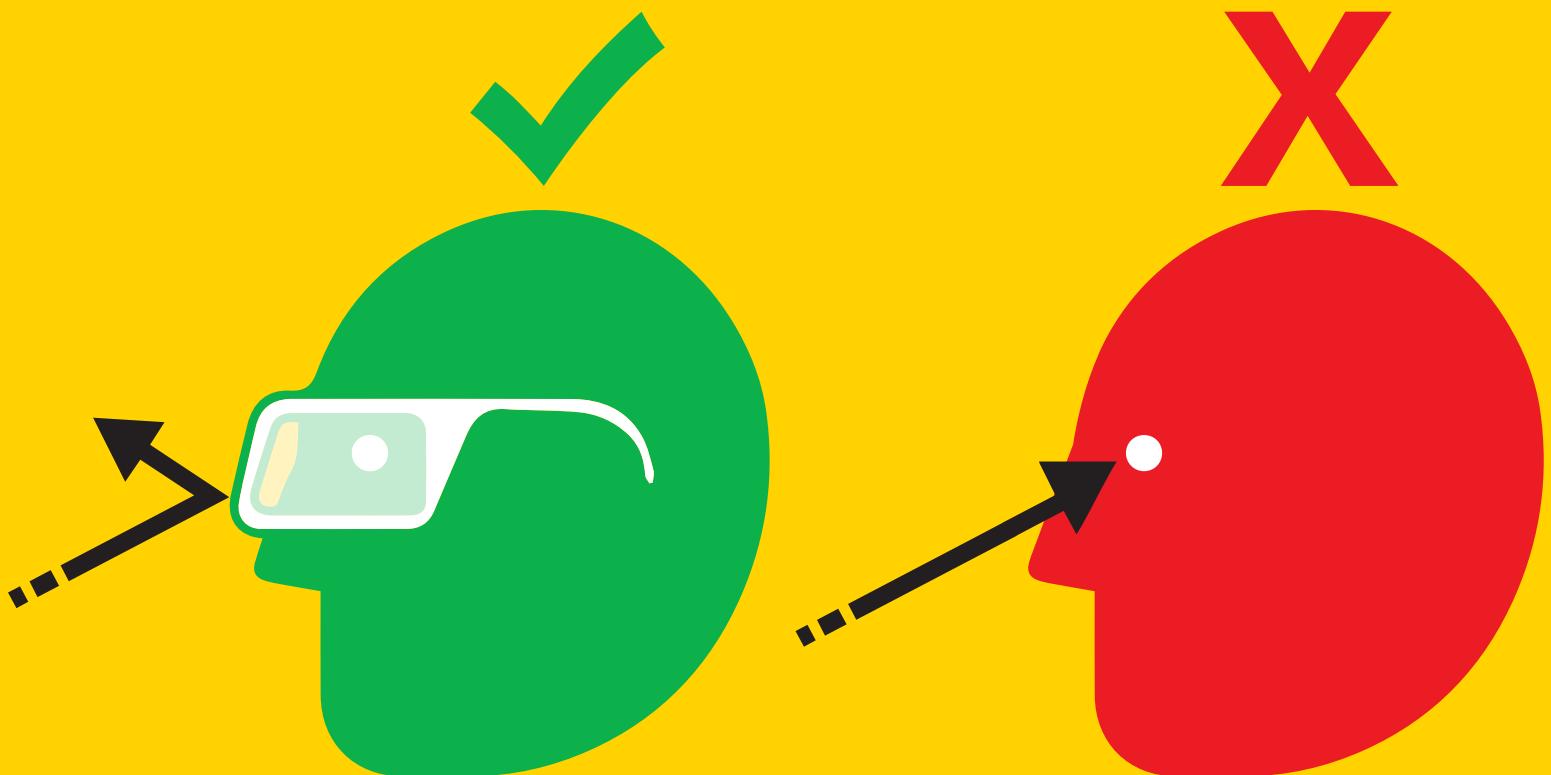
Strong Epoxy  
coating on lens



All EZO-RGB™ Embedded Color Sensors purchased after November 13th 2020, will be IP67 waterproof.

# Caution

At full power the onboard LEDs are **VERY** bright.  
Do not look directly at the light without eye protection!



Minimum brightness = ~400 Lux

Maximum brightness = ~40,000 Lux at 5V (**36,000 Lux at 3.3V**)

# Table of contents

Physical properties	5	Data output	12
Sensor properties	6	CIE 1931 color space	12
Target LED properties	7	Lux	13
Pin out	8	Color matching	14
Performance testing	9	Default state	15
Sensitivity	10	Available data protocol	16
Calibration theory	11		

## UART

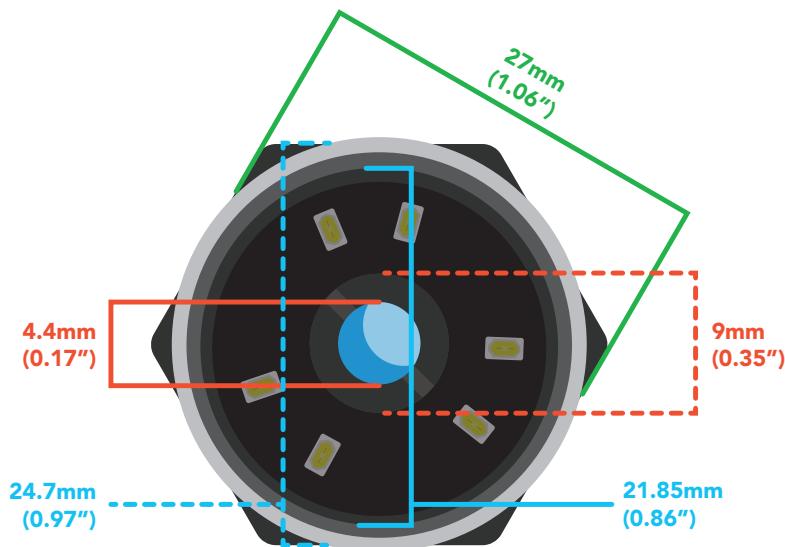
UART mode	18
Receiving data from device	19
Sending commands to device	20
LED color definition	21
<b>UART quick command page</b>	<b>22</b>
Target LED control	23
Indicator LED control	24
Find	25
Continuous mode	26
Single reading mode	27
Calibration	28
Automatic color matching	29
Gamma correction	30
Enable/disable parameters	31
Naming device	32
Device information	33
Response codes	34
Reading device status	35
Sleep mode/low power	36
Change baud rate	37
Protocol lock	38
Factory reset	39
Change to I <sup>2</sup> C mode	40
Manual switching to I <sup>2</sup> C	41

## I<sup>2</sup>C

I <sup>2</sup> C mode	43
Sending commands	44
Requesting data	45
Response codes	46
Processing delay	46
LED color definition	47
<b>I<sup>2</sup>C quick command page</b>	<b>48</b>
Target LED control	49
Indicator LED control	50
Find	51
Taking reading	52
Calibration	53
Gamma correction	54
Enable/disable parameters	55
Naming device	56
Device information	57
Reading device status	58
Sleep mode/low power	59
Protocol lock	60
I <sup>2</sup> C address change	61
Factory reset	62
Change to UART mode	63
Manual switching to UART	64

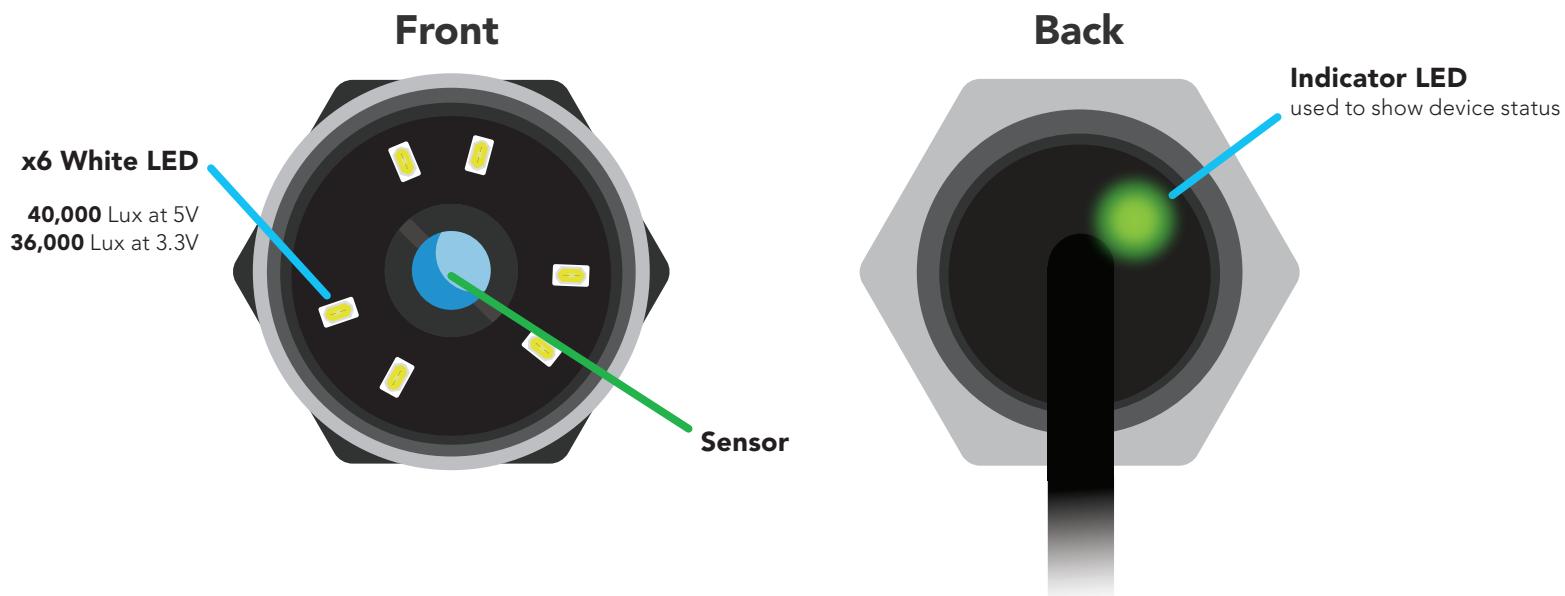
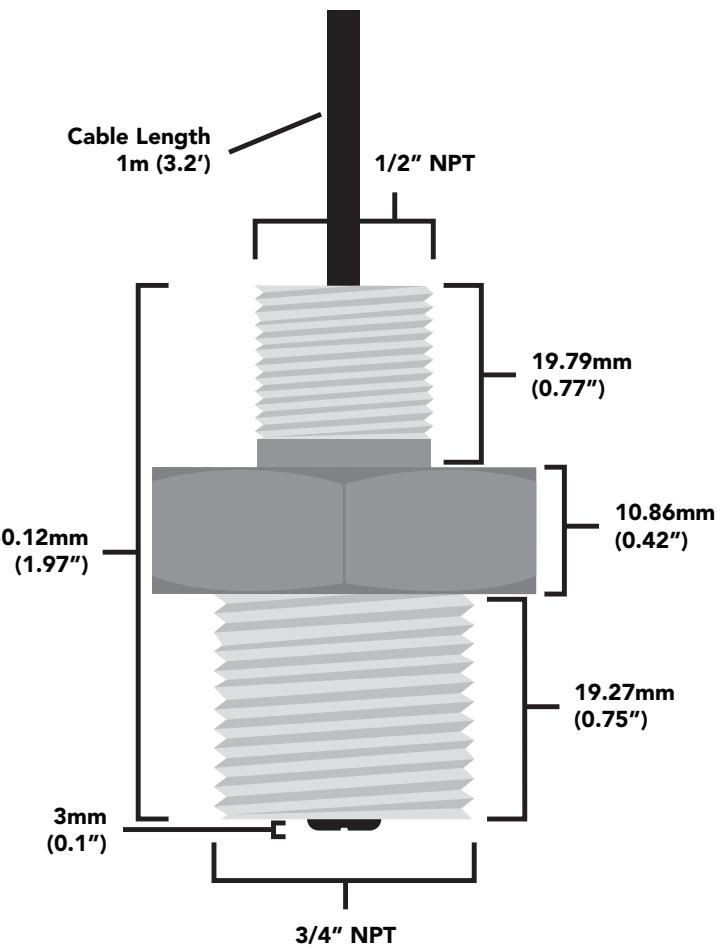
Datasheet change log	65
Firmware updates	66
Warranty	67

# Physical properties



**Weight** 145g

**Body** 316 Stainless Steel

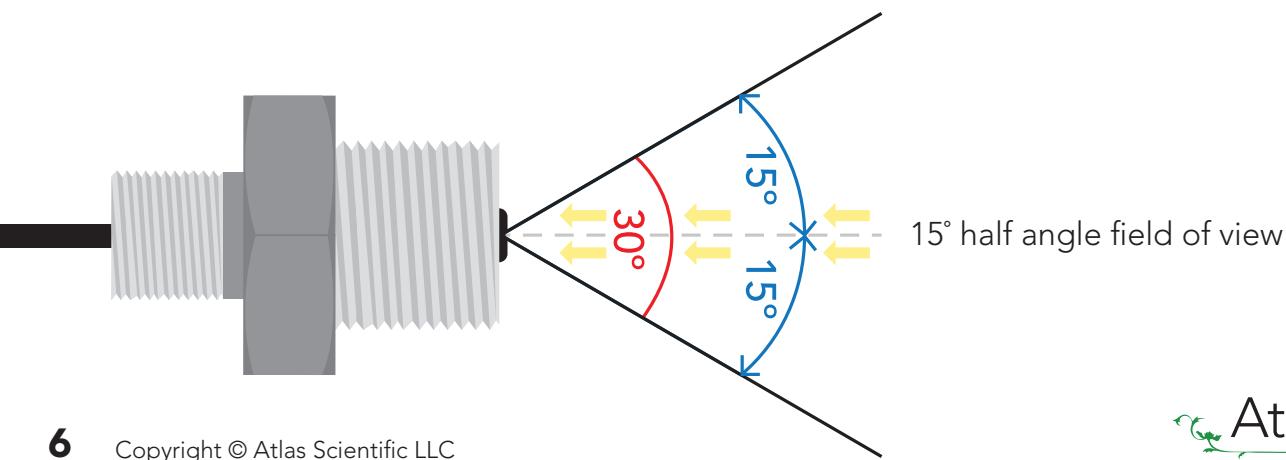
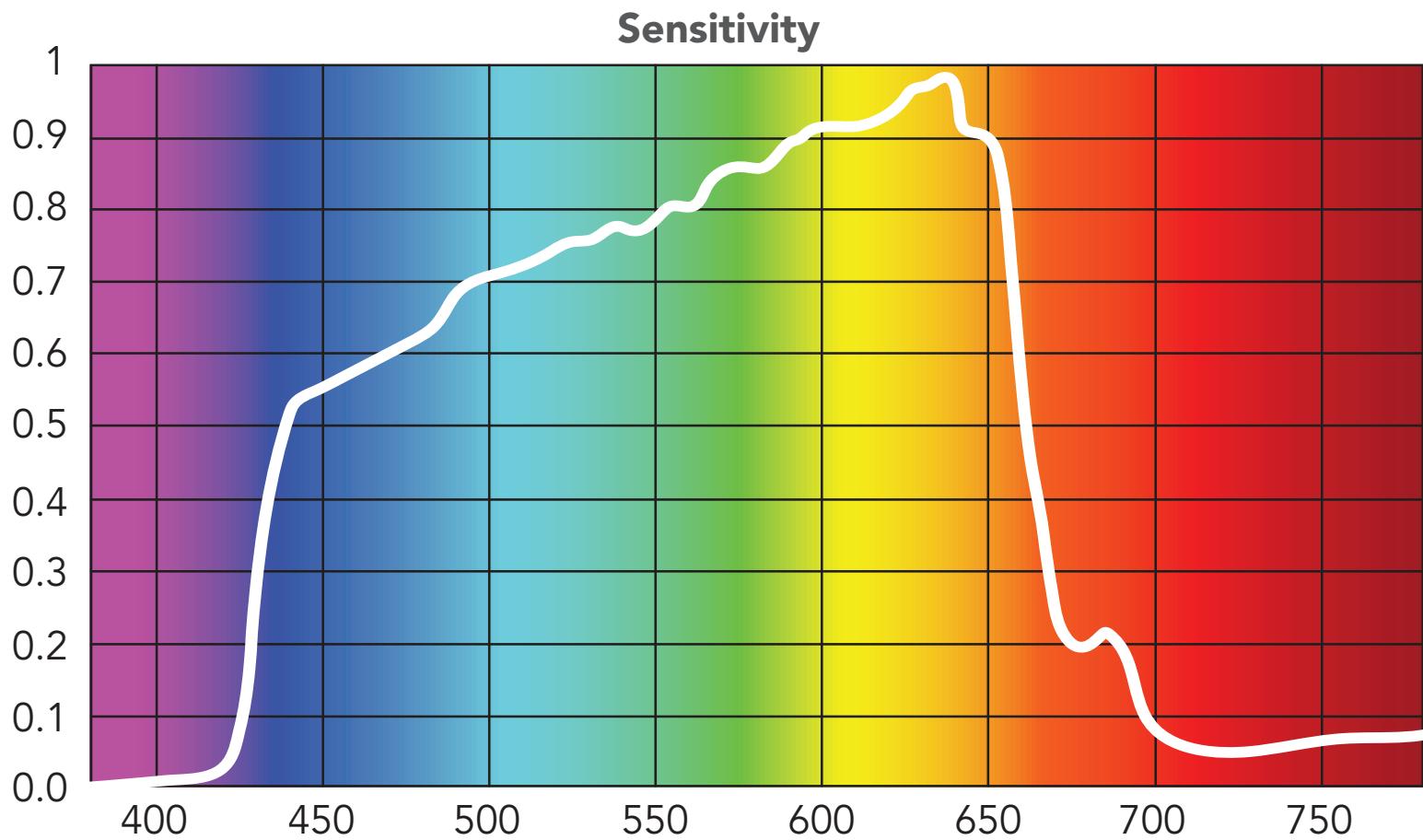


# Sensor properties



**Sensor**

The sensor detects colored light in the red, green and blue spectrum. It is least sensitive to blue light and most sensitive to red light.



# Target LED properties



x6 White LED (5000K color temperature)

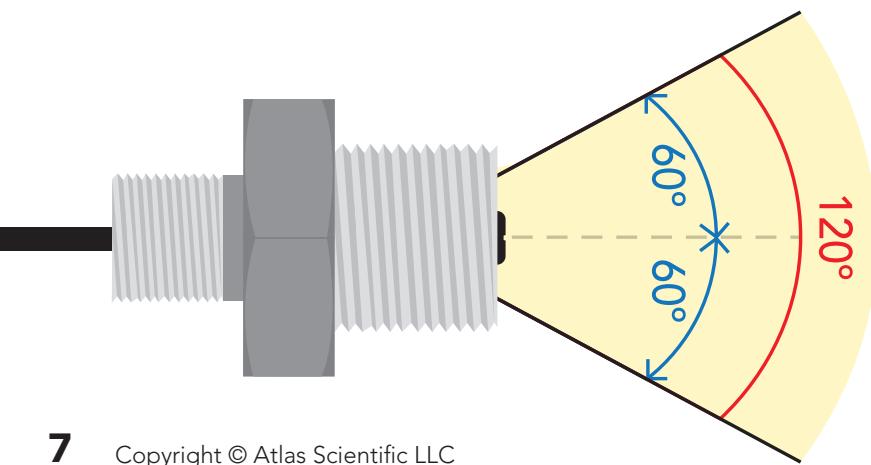
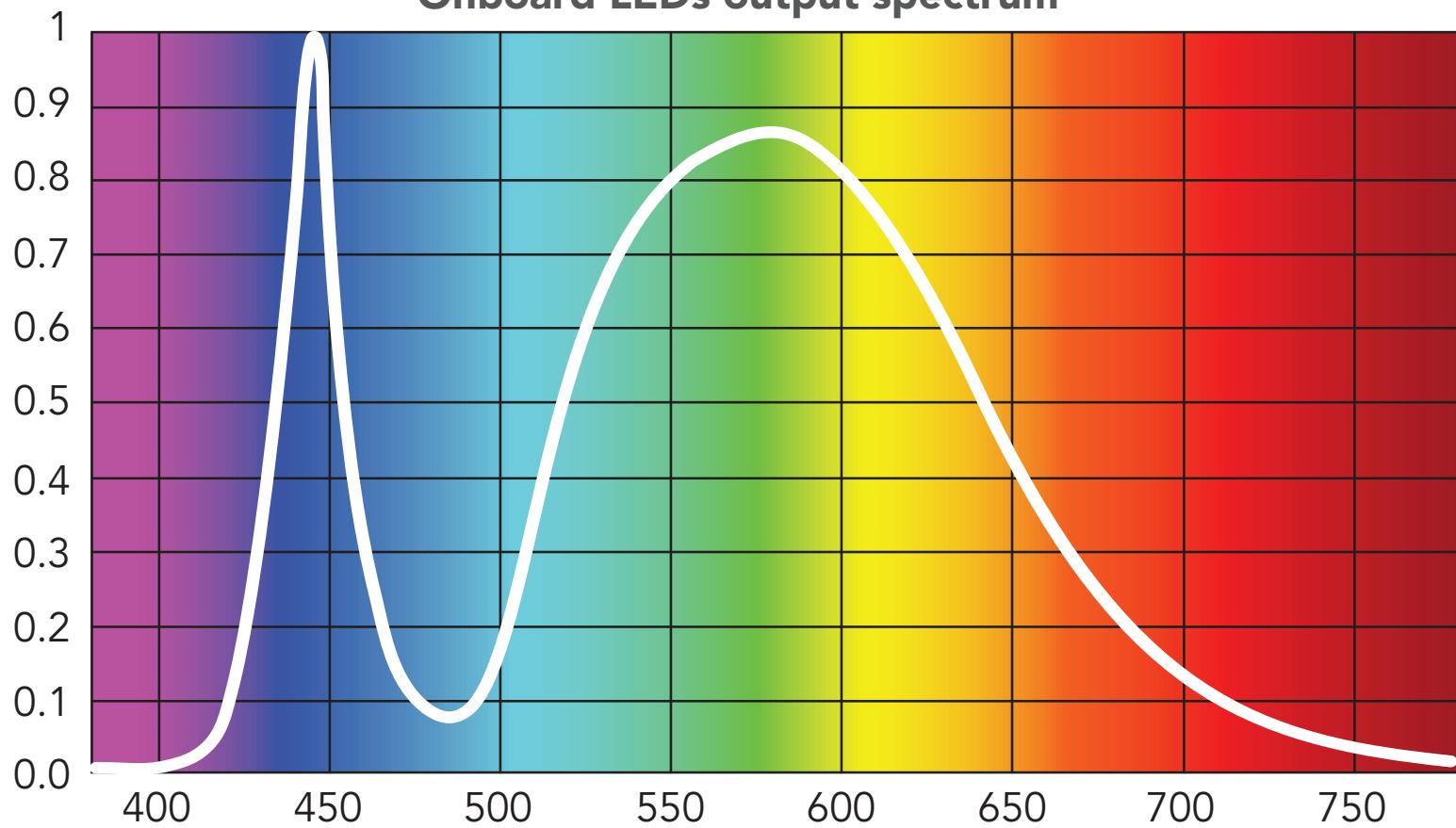
The spectrum output by the six onboard target LEDs is strongest in the blue spectrum and weakest in the red spectrum. This is the opposite of the color sensors sensitivity giving it the best possible color sensing performance.

## Target LED brightness

Minimum ~400 Lux

Maximum ~40,000 Lux

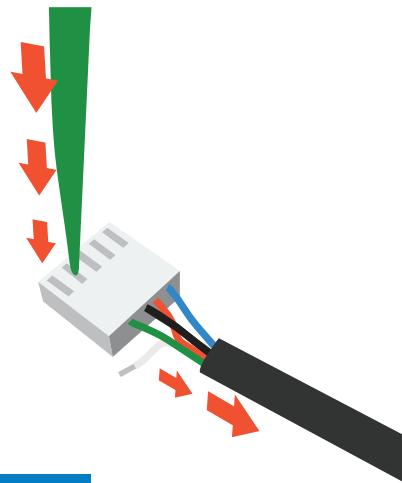
### Onboard LEDs output spectrum



120° angle of illumination

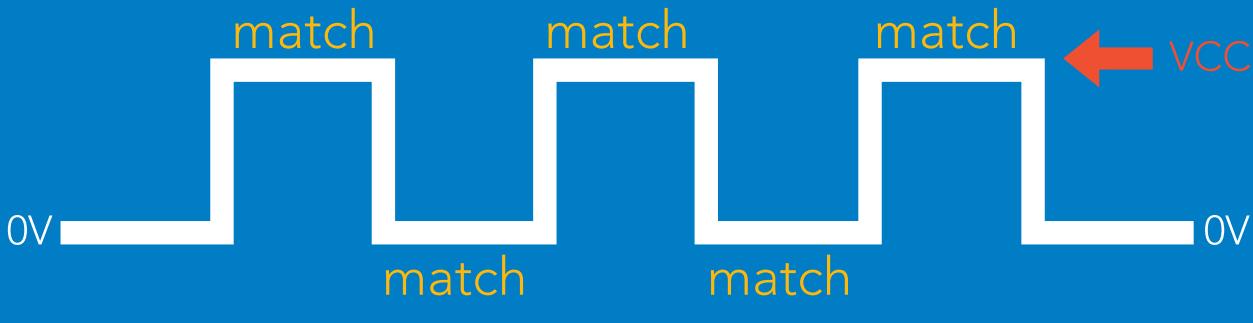
# Pin out

Data and power cable pinout



Should you need to remove this connector from the data cable, follow the provided illustration.

The interrupt pin will change its state when a color match has been detected.



If unused leave **INT** floating. Do not connect **INT** to **VCC** or **GND**.

See page 29 to enable automatic color matching in UART mode.

## Power consumption

	LED	MAX	SLEEP
5V	ON 100%	275 mA	
	ON 1%	15 mA	0.40 mA
	OFF	13 mA	
3.3V	ON 100%	100 mA	
	ON 1%	15 mA	0.14 mA
	OFF	12 mA	

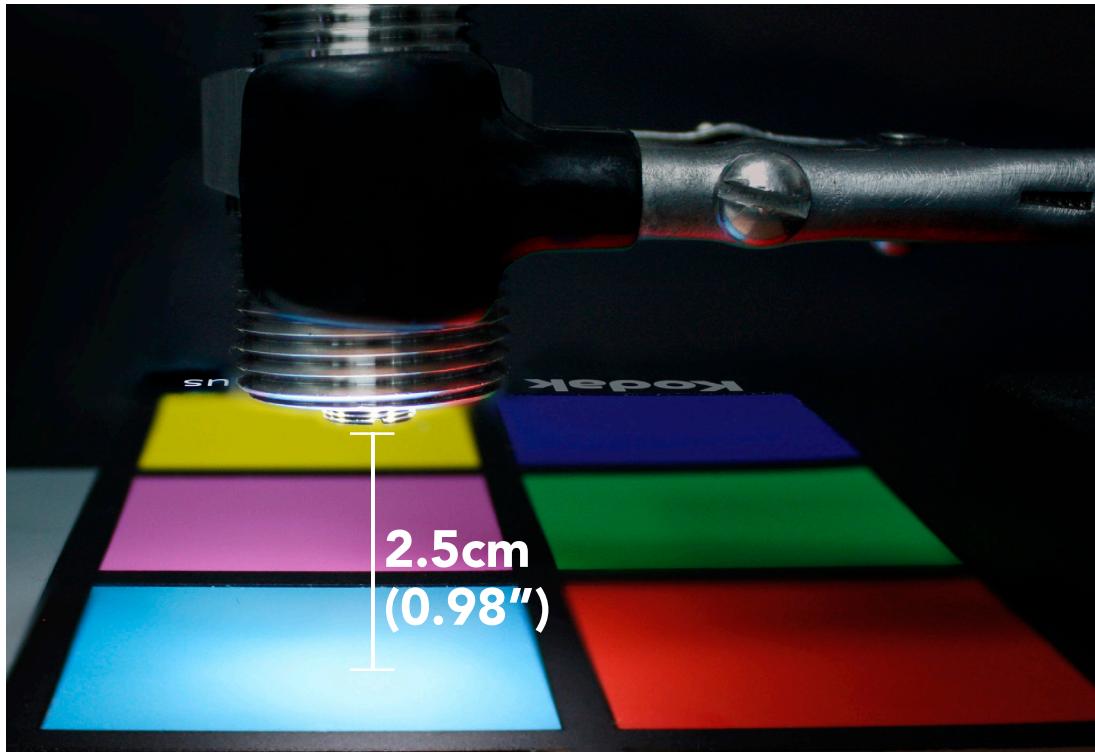
## Absolute max ratings

Parameter	MIN	TYP	MAX
Storage temperature	-65 °C		125 °C
Operational temperature	-40 °C	25 °C	85 °C
VCC	3.3V	3.3V	5.5V
Pressure			1379kPa (200 PSI)

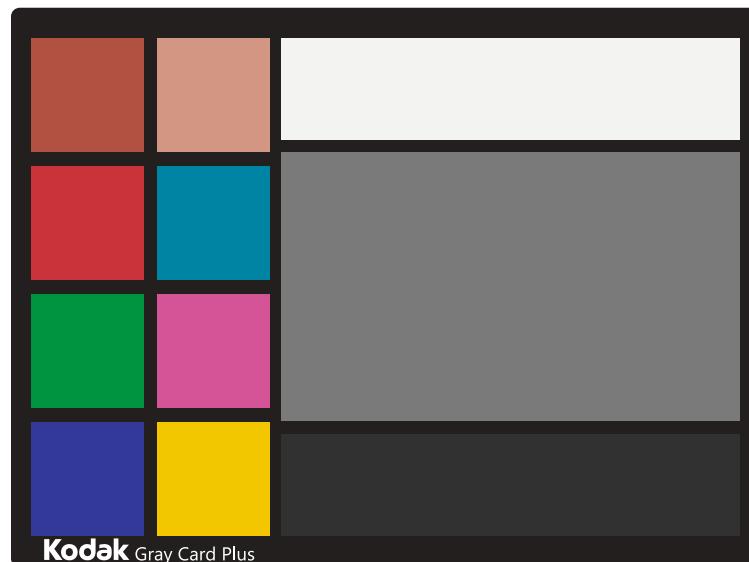
# Performance testing

**Color Sample** Kodak™ Gray Card Plus  
**Distance** 2.5cm  
**On-board LEDs** 100% power  
**VCC** 5V

The color readings were displayed using the free software on the Atlas Scientific™ website located [HERE](#).



Kodak™ Gray Card Plus

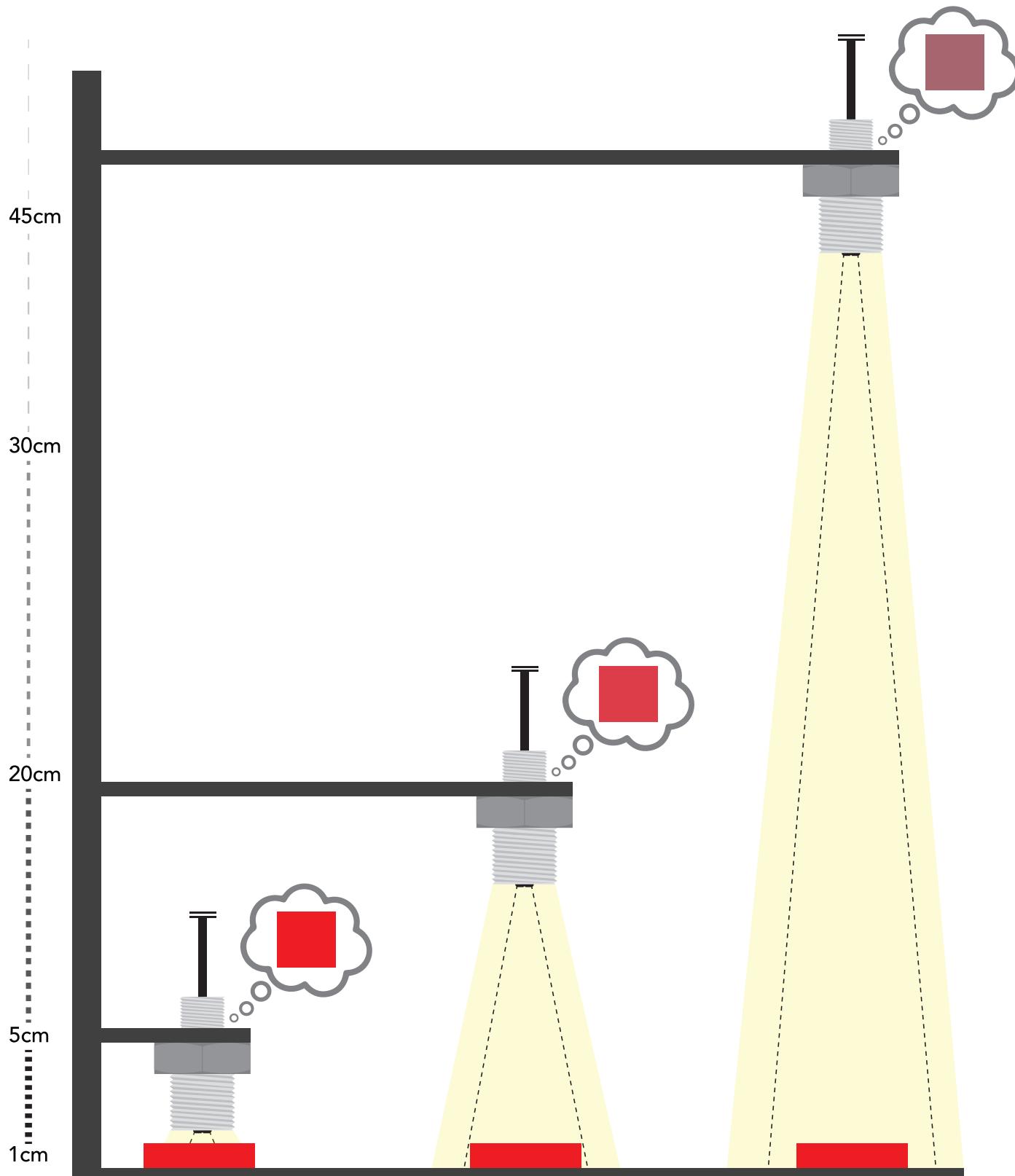


Color output from the EZO-RGB™



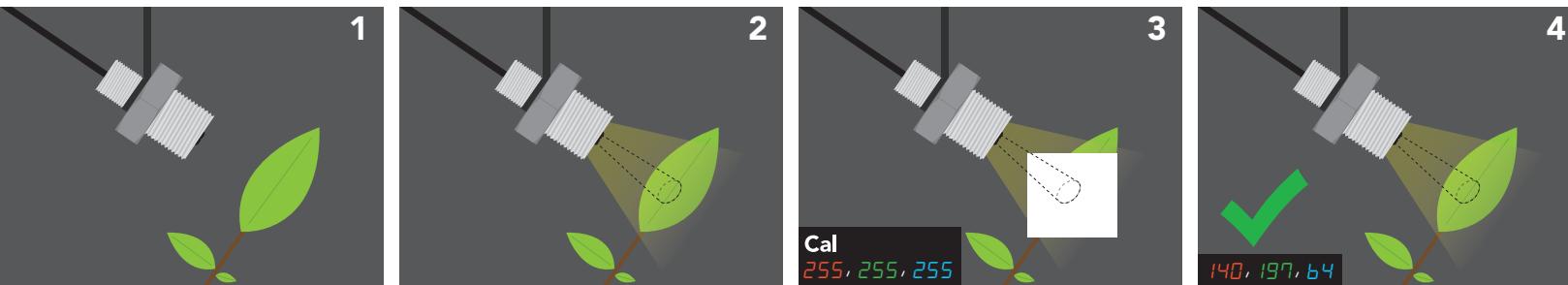
# Sensitivity

As the EZO-RGB™ color sensor is placed further away from the target object, its ability to detect color is diminished. At distances greater than **45cm** most colors become varying shades of gray.

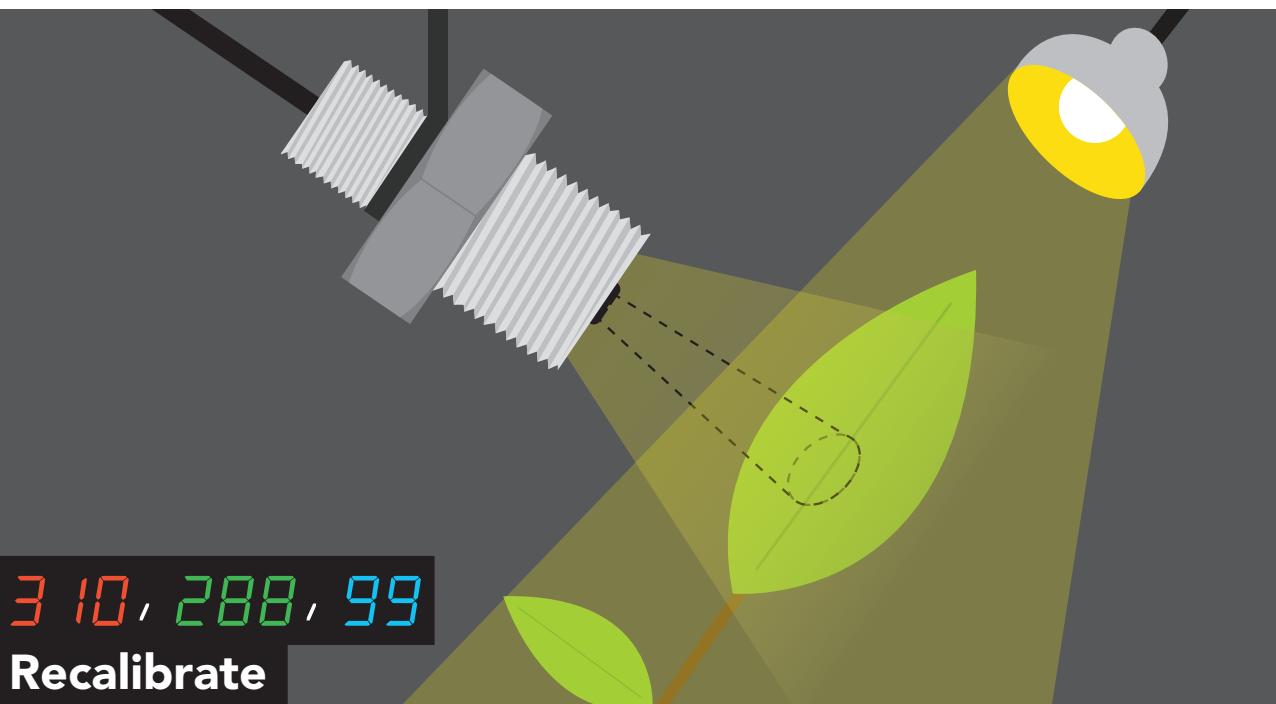


# Calibration theory

The EZO-RGB™ color sensor is designed to be calibrated to a white object at the maximum brightness the object will be viewed under. In order to get the best results Atlas Scientific strongly recommends that the sensor is mounted into a fixed location. Holding the sensor in your hand during calibration will decrease performance.



1. Embed the EZO-RGB™ color sensor into its intended use location.
2. Set LED brightness to the desired level.
3. Place a white object in front of the target object and issue the calibration command "Cal".
4. A single color reading will be taken and the device will be fully calibrated.



The RGB output has a three comma separated value, ranging from 0–255. However, it is possible to get RGB readings where one, or all of the values are greater than 255. This is because brightness is encoded in a RGB reading, if the subject being viewed is brighter than the calibrated brightness, the RGB values can go above 255. If this happens, the EZO-RGB™ Embedded Color Sensor needs to be re-calibrated for the correct brightness.

# Data output

RGB

Default output

8-bit Red

8-bit Green

8-bit Blue

8-bit color graphics

} 24 bits in total

Color pallet

Output frequency

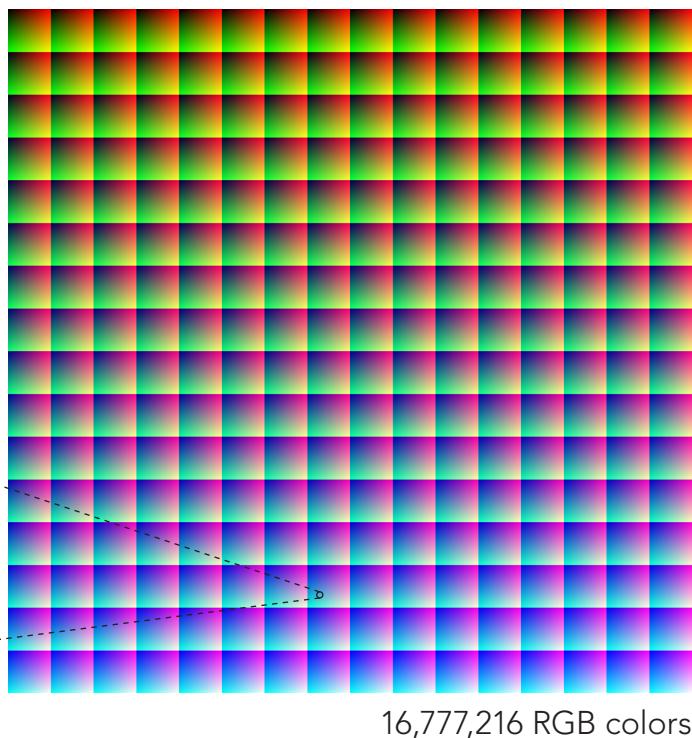
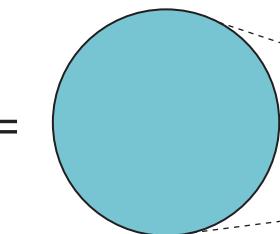
Output format

16,777,216 colors (24 Bit)

1 reading every 400ms

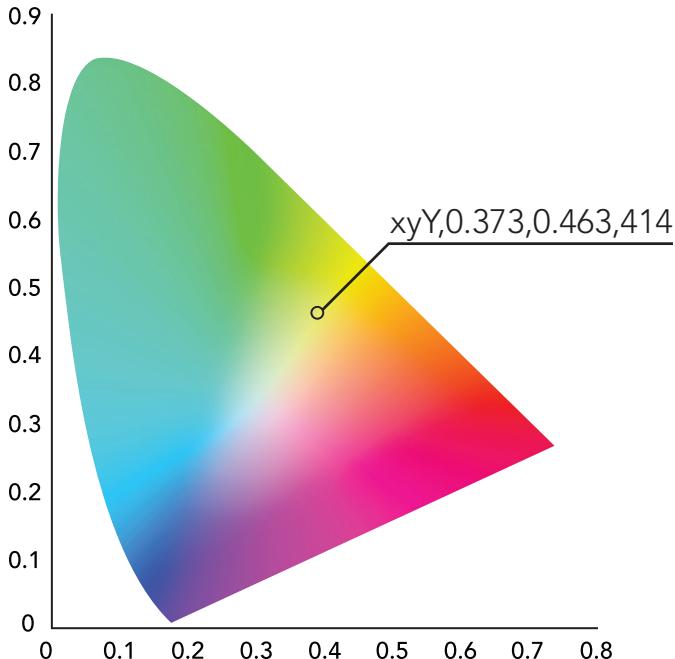
CSV string 24 bits

122, 196, 211  
8-bit Red      8-bit Green      8-bit Blue



## CIE 1931 color space

Human perception of color is not the same as a sensors perception of color. The CIE output is a representation of human color perception, while the RGB output is a representation of machine perception. While the two are close, they are not the same.



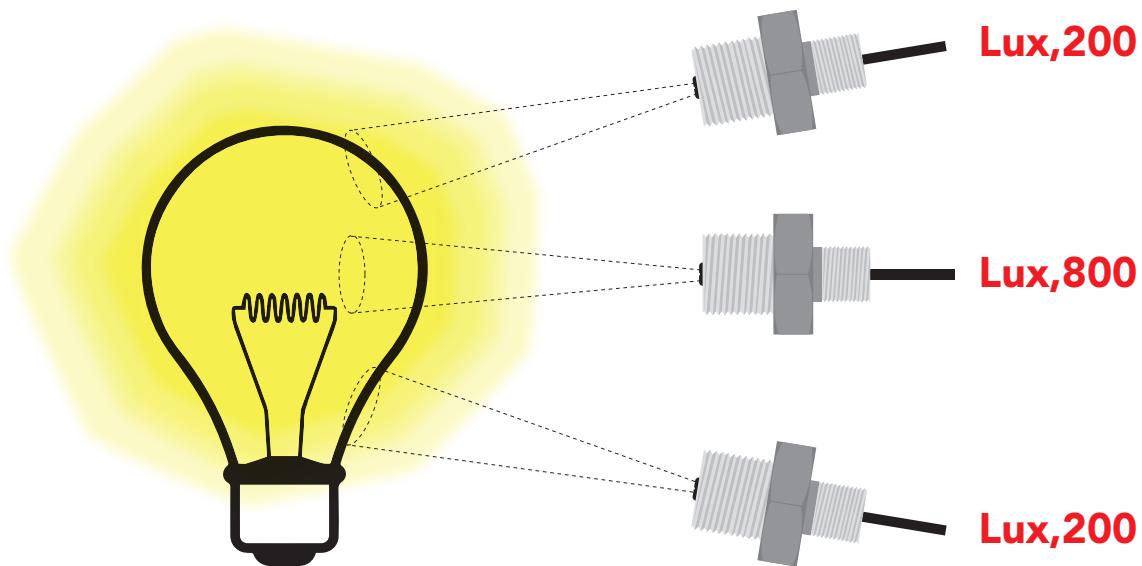
Identifier    x                  y                  Y  
xyY, 0.373, 0.463, 414

xy = coordinates

Y = luminance

# Lux

Lux is a measure of light intensity as perceived by the human eye. The lux output has a comma separated identifier “**Lux**” followed by a single integer value from 0 – 65535. Lux readings will be effected by the sensors position.



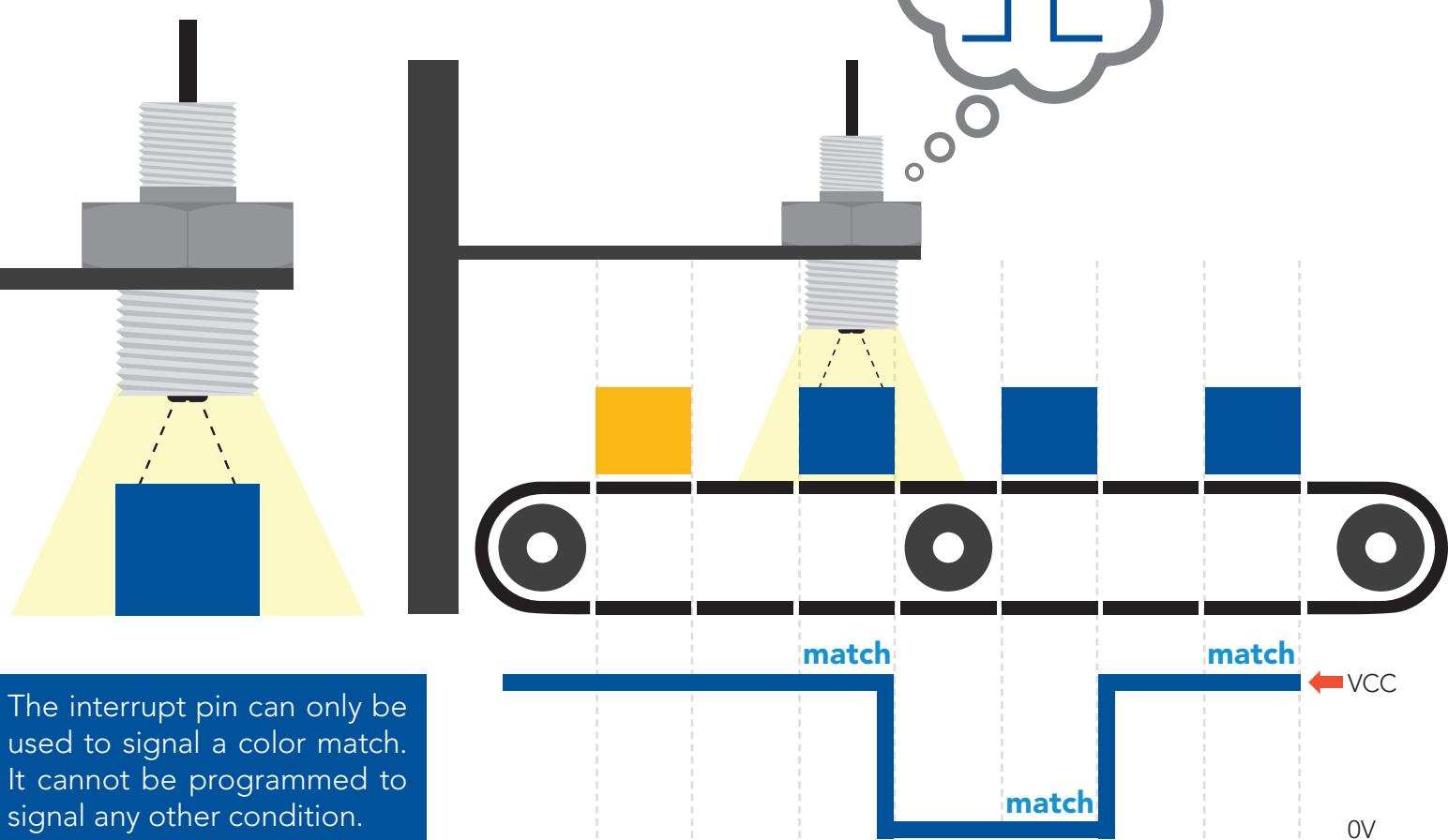
# Color matching

The EZO-RGB™ can indicate when a preset color is detected.

**Place object of any color under the sensor.**

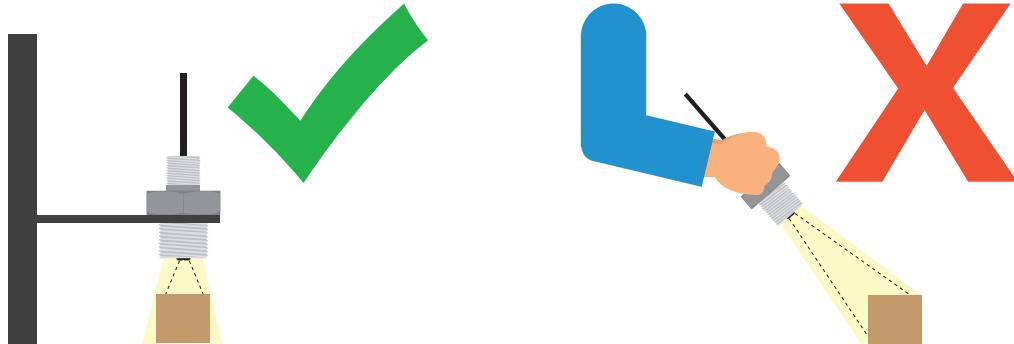
**Issue command "M,1".**

**Color matching has been enabled.**



When a color match has been detected the reading will be appended with "**\*M**" and the interrupt pin will change its state.

**In order for color matching to work the EZO-RGB™ must be securely mounted and remain a fixed distance from its target.**



# Default state UART mode

Baud

9,600

Readings

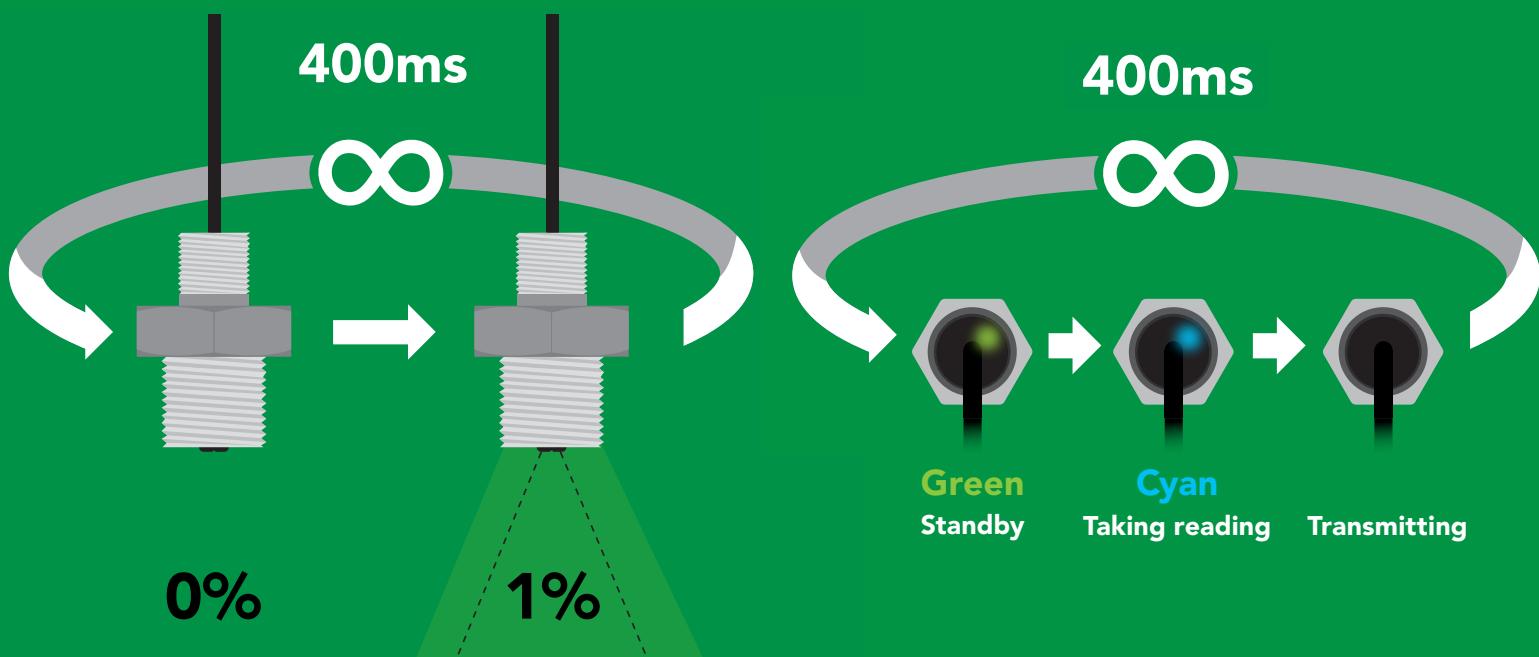
continuous

Speed

400 milliseconds

LED

on, when taking reading



 Available data protocols

**UART**

**default**

**I<sup>2</sup>C**

**X Unavailable data protocols**

**SPI**

**Analog**

**RS-485**

**Mod Bus**

**4–20mA**

# UART mode

## Settings that are retained if power is cut

Automatic color matching  
Baud rate  
Calibration  
Continuous mode  
Device name  
Enable/disable parameters  
Enable/disable response codes  
LED control

## Settings that are **NOT** retained if power is cut

Sleep mode

# UART mode

8 data bits      no parity  
1 stop bit      no flow control

Baud 300  
1,200  
2,400  
**9,600 default**  
19,200  
38,400  
57,600  
115,200

**RX**  
Data in

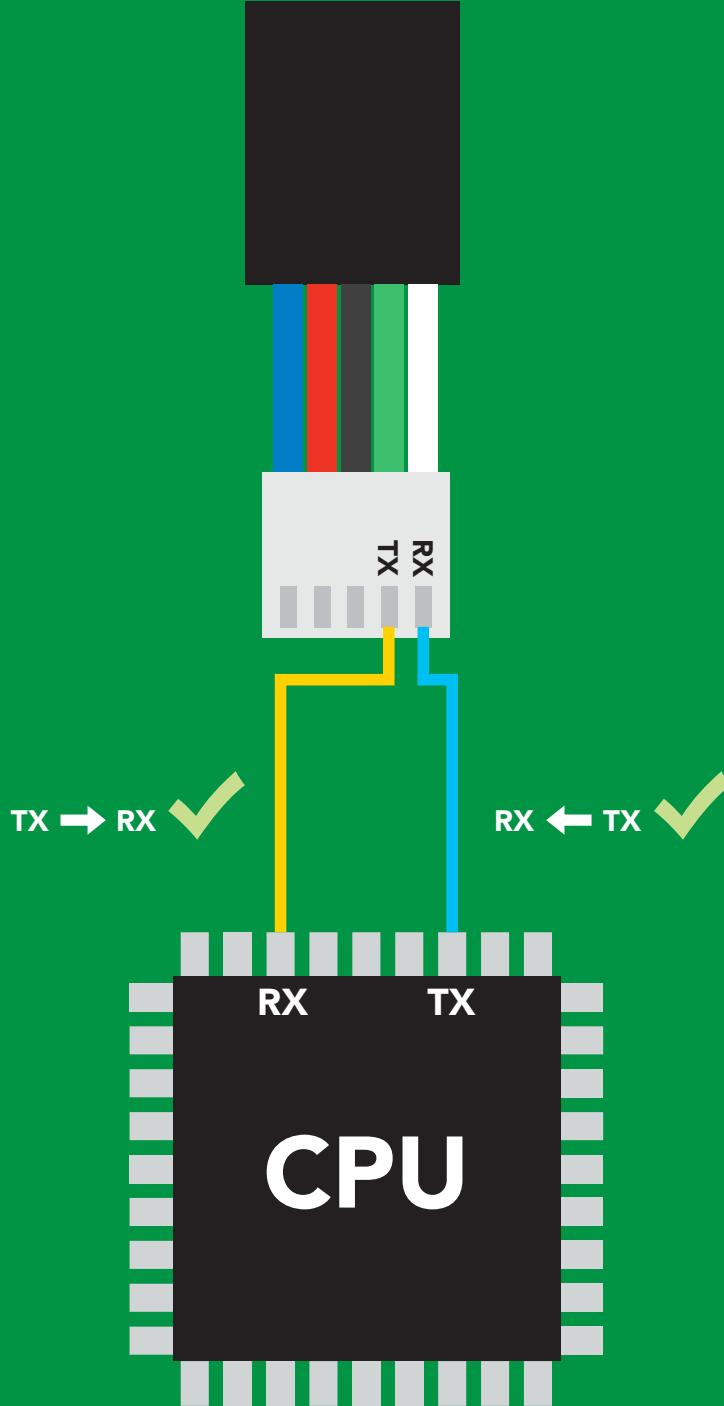


**TX**  
Data out



**Vcc** 3.3V – 5V

0V      VCC      0V

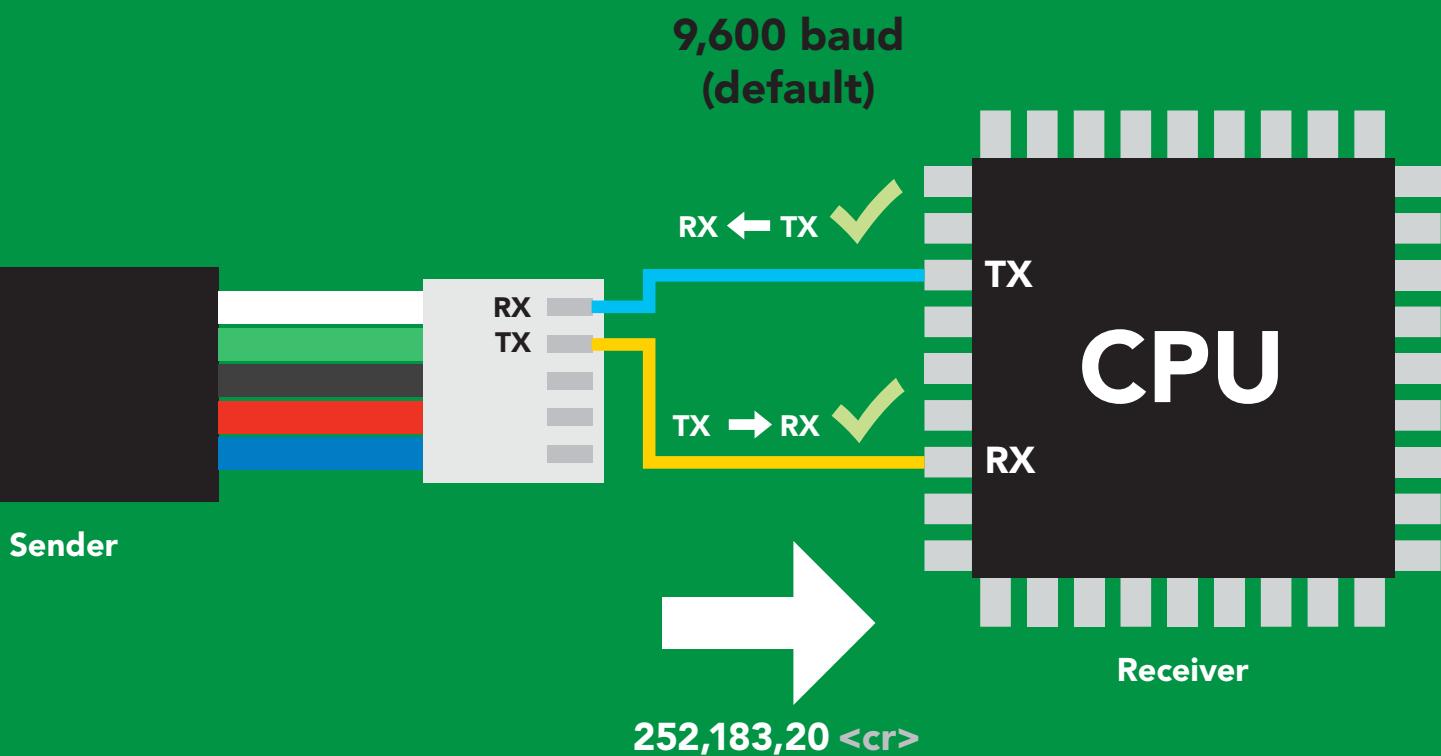


## Data format

Units	RGB, LUX, & CIE	Data type	integer & floating point
Encoding	ASCII		
Format	string	Decimal places	3
Terminator	carriage return	Smallest string	4 characters
		Largest string	52 characters

# Receiving data from device

2 parts



## Advanced

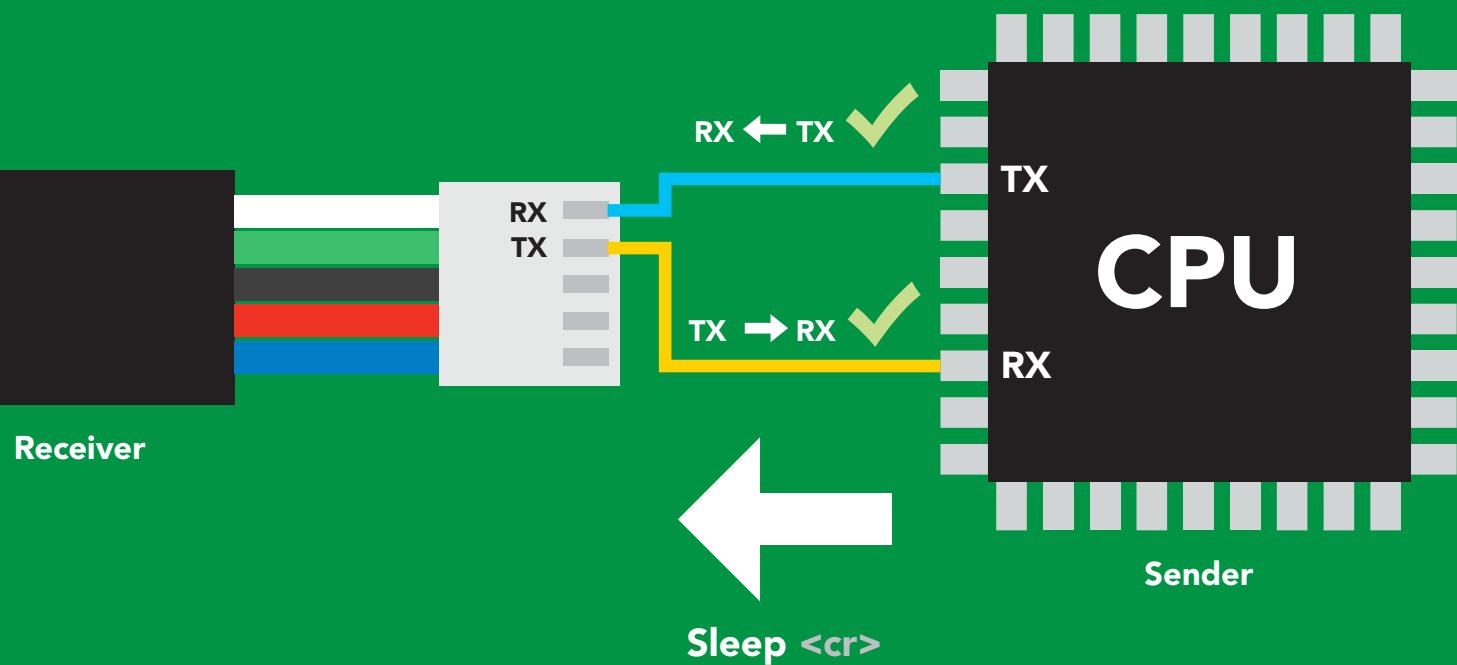
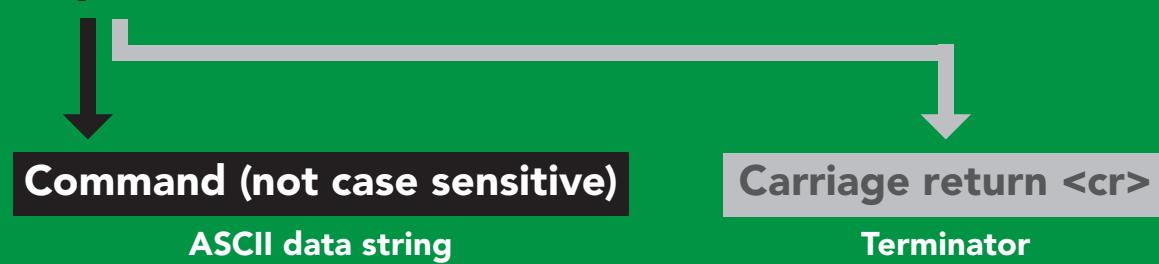
ASCII: 2 5 2 , 1 8 3 , 2 0 <cr>

Hex: 32 35 32 2C 31 38 33 2C 32 30 0D

Dec: 50 53 50 44 49 56 51 44 50 48 13

# Sending commands to device

2 parts



## Advanced

ASCII: S I e e p <cr>

Hex: 53 6C 65 65 70 0D

Dec: 83 108 101 101 112 13

# Indicator LED definition



**Green**

UART standby



**Cyan**

Taking reading



**Purple**

Changing  
I<sup>2</sup>C address



**Red**

Command  
not understood



**White**

Find

	LED ON <b>+2.5 mA</b>
<b>5V</b>	
<b>3.3V</b>	<b>+1 mA</b>

# UART mode

## command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function	Default state
Baud	change baud rate	pg. 37 9,600
C	enable/disable continuous mode	pg. 26
Cal	performs calibration	pg. 28 n/a
Factory	enable factory reset	pg. 39 n/a
Find	finds device with blinking white LED	pg. 25 n/a
G	gamma correction	pg. 30 n/a
i	device information	pg. 33 n/a
iL	enable/disable indicator LED	pg. 24 enabled
I2C	change to I <sup>2</sup> C mode	pg. 40 not set
L	enable/disable target LED	pg. 23 enabled
M	automatic color matching	pg. 29 enabled
	set/show name of device	pg. 32 not set
O	enable/disable parameters	pg. 31 RGB
Plock	enable/disable protocol lock	pg. 38 n/a
R	returns a single reading	pg. 27 n/a
Sleep	enter sleep mode/low power	pg. 35 n/a
Status	retrieve status information	pg. 40 n/a
*OK	enable/disable response codes	pg. 34 n/a

# Target LED control

## Command syntax

% represents the percentage of target LED brightness. (any number from 0–100)

L,% <cr> set target LED brightness

L%,T <cr> set target LED brightness/trigger target LED only when a reading is taken (*power saving*)

L,? <cr> target LED state on/off?

## Example Response

L,32 <cr>

\*OK <cr> target LED set to 32% brightness.

L,14,T <cr>

\*OK <cr> target LED set to 14% brightness, and will only turn on when a reading is taken.

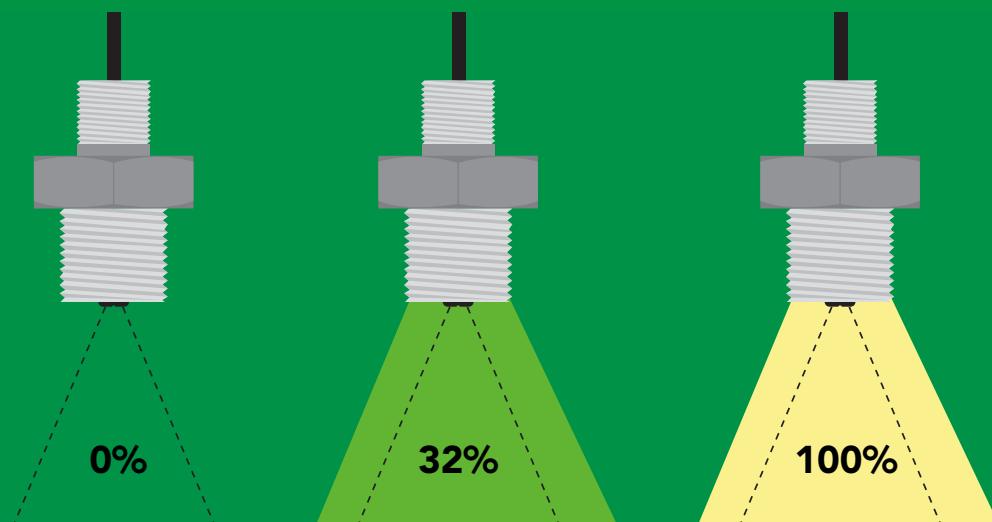
L,? <cr>

?L, %, [T] <cr>  
\*OK <cr>

L,0 <cr>

L,32 <cr>

L,100 <cr>



# Indicator LED control

## Command syntax

iL,1 <cr> indicator LED on **default**

iL,0 Indicator LED off

iL,? <cr> Indicator LED state on/off?

### Example

iL,1 <cr>

iL,0 <cr>

iL,? <cr>

### Response

\*OK <cr>

\*OK <cr>

?iL,1 <cr> or ?iL,0 <cr>

\*OK <cr>



iL,1



iL,0

# Find

## Command syntax

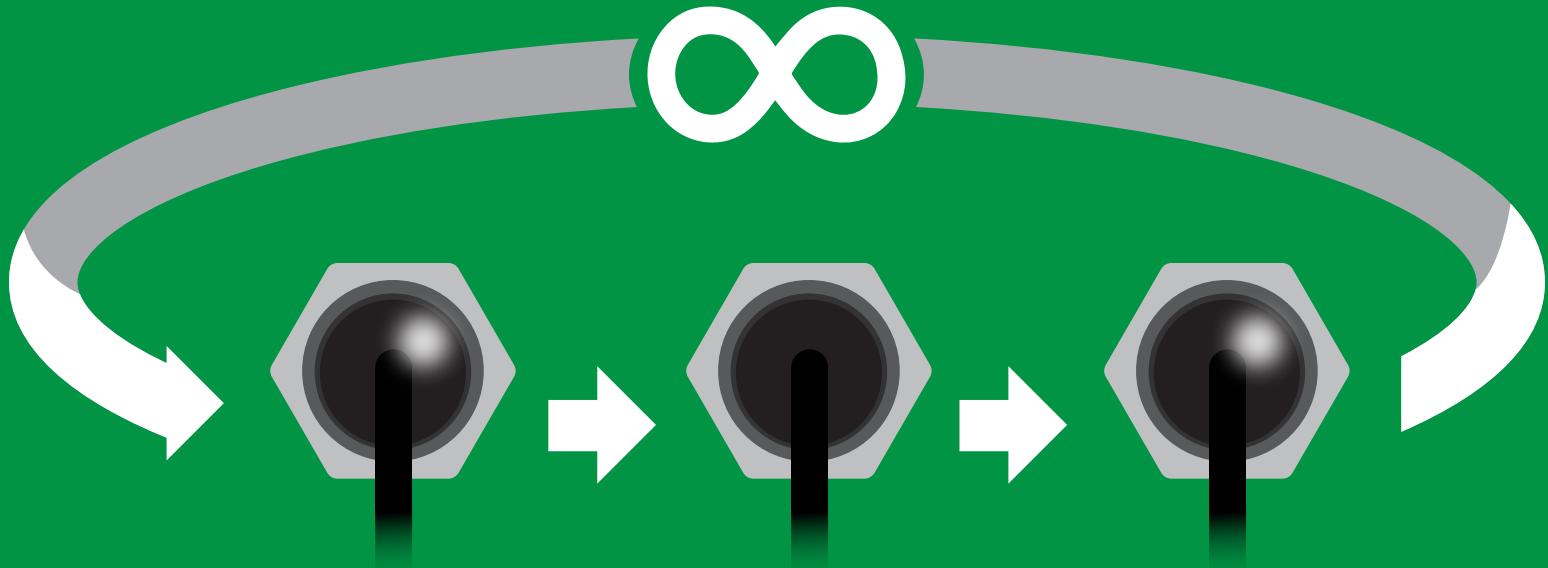
This command will disable continuous mode  
Send any character or command to terminate find.

Find <cr> LED rapidly blinks white, used to help find device

### Example      Response

Find <cr>

\*OK <cr>



# Continuous mode

## Command syntax

- C,1 <cr> enable continuous readings once per 400ms **default**
- C,n <cr> continuous readings every n x 400ms (n = 2 to 99)
- C,0 <cr> disable continuous readings
- C,? <cr> continuous reading mode on/off?

## Example      Response

C,1 <cr>

\*OK <cr>  
R,G,B (400ms) <cr>  
R,G,B (800ms) <cr>  
R,G,B (1200ms) <cr>

C,30 <cr>

\*OK <cr>  
R,G,B (12,000ms) <cr>  
R,G,B (24,000ms) <cr>  
R,G,B (36,000ms) <cr>

C,0 <cr>

\*OK <cr>

C,? <cr>

?C,1 <cr> or ?C,0 <cr> or ?C,30 <cr>  
\*OK <cr>

# Single reading mode

## Command syntax

R <cr> takes single reading

### Example      Response

R <cr>

R,G,B <cr>  
\*OK <cr>



Green  
Standby



Cyan  
Taking reading



Transmitting



# Calibration

## Command syntax

**Cal <cr>** calibrates the EZO-RGB™

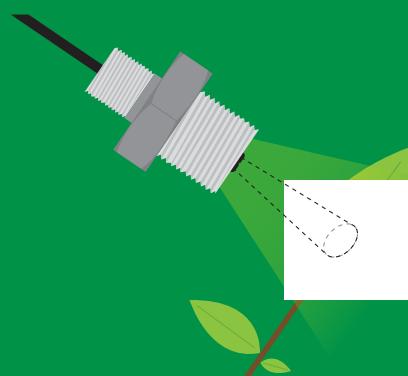
1. place white object (such as a piece of paper) in front of target
2. Issue "cal" command

## Example Response

**Cal <cr>**

**\*OK <cr>**

**Cal <cr>**



# Automatic color matching

## Command syntax

M,1 <cr> enables automatic color matching

M,0 <cr> disables automatic color matching

M,? <cr> color matching on/off?

## Example

M,1 <cr>

M,0 <cr>

M,? <cr>

## Response

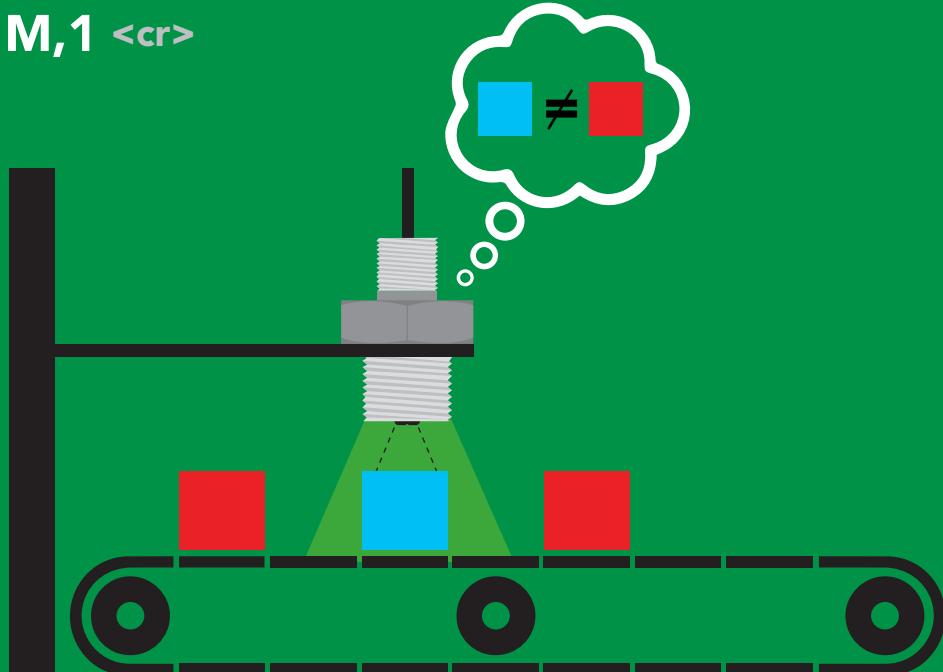
\*OK <cr>

\*OK <cr>

?M,1 <cr> or ?M,0 <cr>

\*OK <cr>

M,1 <cr>



# Gamma correction

## Command syntax

**G,n <cr> set gamma correction**

where n = a floating point number from 0.01 – 4.99

**G,? <cr> gamma correction value?**

Adjusting the gamma correction helps adjust the color seen by the sensor.

The default gamma correction is 1.00 which represents no correction at all.  
A gamma correction factor is a floating point number from 0.01 to 4.99.

## Example

**G,1.99 <cr>**

## Response

**\*OK <cr>**

**G,? <cr>**

**?G,1.99 <cr>**

**\*OK <cr>**

# Enable/disable parameters from output string

## Command syntax

O, [parameter],[1,0] <cr> enable or disable output parameter

O,? <cr> enabled parameter?

### Example

O,RGB,1 / O,RGB,0 <cr>

### Response

\*OK <cr> enable / disable RGB

O,LUX,1 / O,LUX,0 <cr>

\*OK <cr> enable / disable lux

O,CIE,1 / O,CIE,0 <cr>

\*OK <cr> enable / disable CIE

O,? <cr>

? ,O,RGB,LUX,CIE <cr> if all enabled

### Parameters

RGB red, green, blue

LUX illuminance

CIE CIE 1931 color space

### Followed by 1 or 0

1 enabled

0 disabled

\* If you disable all possible data types  
your readings will display "no output".

# Naming device

## Command syntax

Do not use spaces in the name

Name,n <cr> set name

n =    1 2 3 4 5 6 7 8    —    10 11 12 13 14 15 16

Name, <cr> clears name

Up to 16 ASCII characters

Name,? <cr> show name

## Example

## Response

Name, <cr>

\*OK <cr> name has been cleared

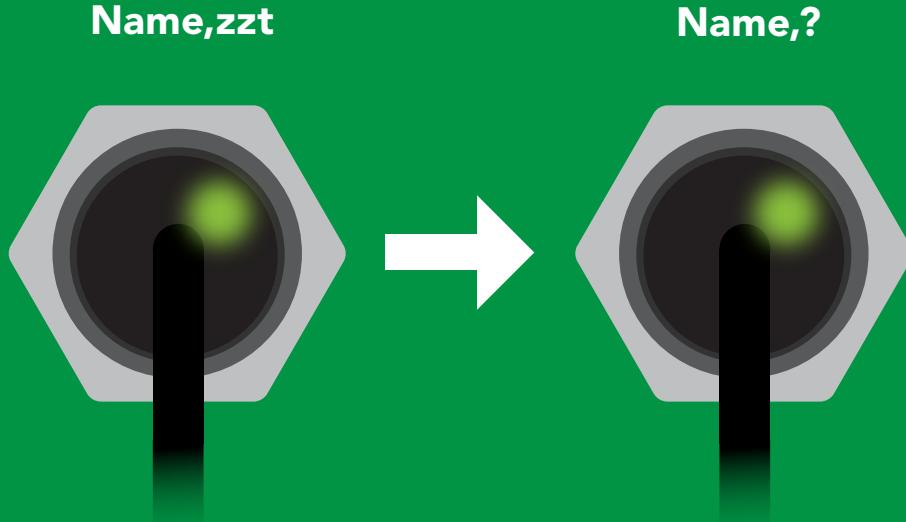
Name,zzt <cr>

\*OK <cr>

Name,? <cr>

?Name,zzt <cr>

\*OK <cr>



\*OK <cr>

?Name,zzt <cr>  
\*OK <cr>

# Device information

## Command syntax

i <cr> device information

### Example      Response

i <cr>

?i,RGB,2.1 <cr>  
\*OK <cr>

### Response breakdown

?i,    RGB,    2.1  
     ↑           ↑  
    Device    Firmware

# Response codes

## Command syntax

- \*OK,1 <cr> enable response **default**
- \*OK,0 <cr> disable response
- \*OK,? <cr> response on/off?

## Example

R <cr>

**140,197,64 <cr>**  
**\*OK <cr>**

**\*OK,0 <cr>**

**no response, \*OK disabled**

R <cr>

**140,197,64 <cr> \*OK disabled**

**\*OK,? <cr>**

**?\*OK,1 <cr> or ?\*OK,0 <cr>**

## Other response codes

- \*ER unknown command
- \*OV over volt (VCC>=5.5V)
- \*UV under volt (VCC<=3.1V)
- \*RS reset
- \*RE boot up complete, ready
- \*SL entering sleep mode
- \*WA wake up

**These response codes  
cannot be disabled**

# Reading device status

## Command syntax

Status <cr> voltage at Vcc pin and reason for last restart

### Example      Response

Status <cr>

?Status,P,5.038 <cr>

\*OK <cr>

### Response breakdown

?Status, P,  
↑  
Reason for restart      5.038  
                         ↑  
                         Voltage at Vcc

#### Restart codes

P	powered off
S	software reset
B	brown out
W	watchdog
U	unknown

# Sleep mode/low power

## Command syntax

Send any character or command to awaken device.

**Sleep <cr>** enter sleep mode/low power

### Example

**Sleep <cr>**

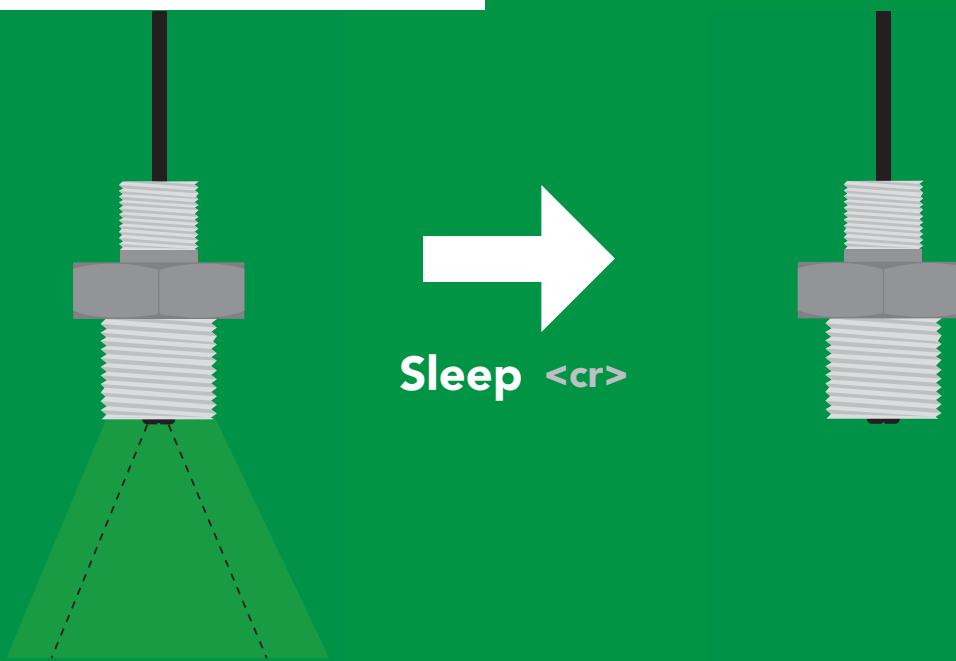
### Response

**\*OK <cr>**  
**\*SL <cr>**

### Any command

**\*WA <cr>** wakes up device

	MAX	SLEEP
<b>5V</b>	<b>175 mA</b>	<b>0.40 mA</b>
<b>3.3V</b>	<b>138 mA</b>	<b>0.18 mA</b>



# Change baud rate

## Command syntax

Baud,n <cr> change baud rate

### Example      Response

Baud,38400 <cr>

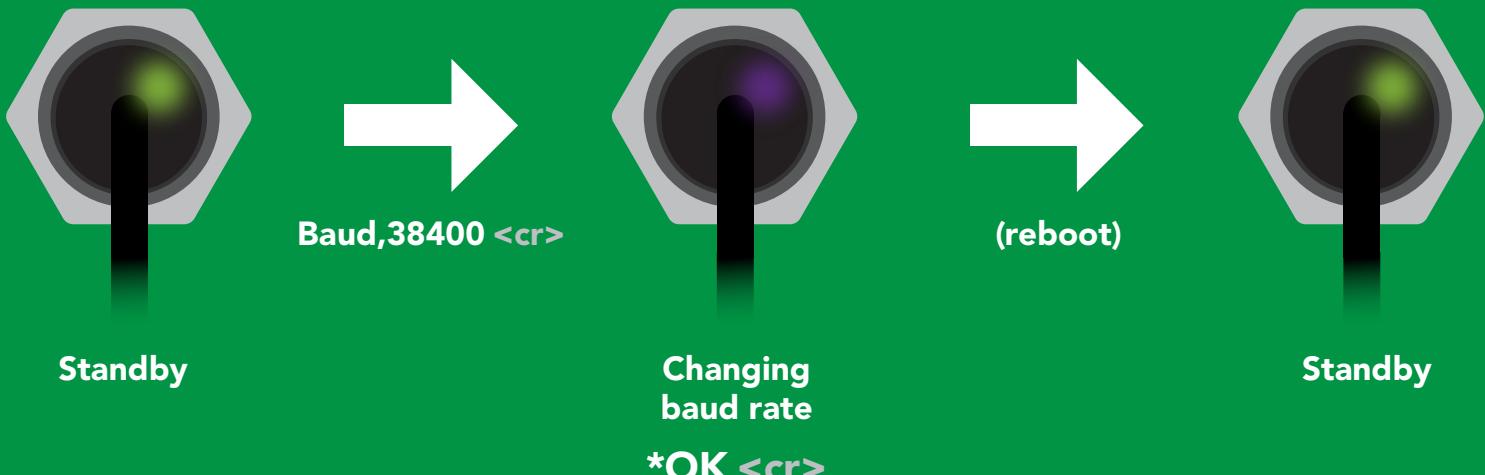
\*OK <cr>

Baud,? <cr>

?Baud,38400 <cr>

\*OK <cr>

n = [ 300  
1200  
2400  
**9600 default**  
19200  
38400  
57600  
115200 ]



# Protocol lock

## Command syntax

Locks device to UART mode.

Plock,1 <cr> enable Plock

Plock,0 <cr> disable Plock **default**

Plock,? <cr> Plock on/off?

## Example

Plock,1 <cr>

\*OK <cr>

Plock,0 <cr>

\*OK <cr>

Plock,? <cr>

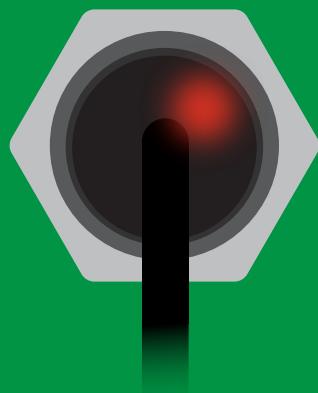
?Plock,1 <cr> or ?Plock,0 <cr>

Plock,1

I2C,100

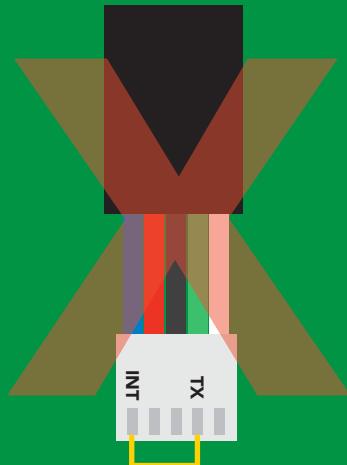


\*OK <cr>



cannot change to I<sup>2</sup>C

\*ER <cr>



cannot change to I<sup>2</sup>C

# Factory reset

## Command syntax

Clears calibration  
Reset target LED brightness to 1%  
Reset output to RGB  
"\*OK" enabled

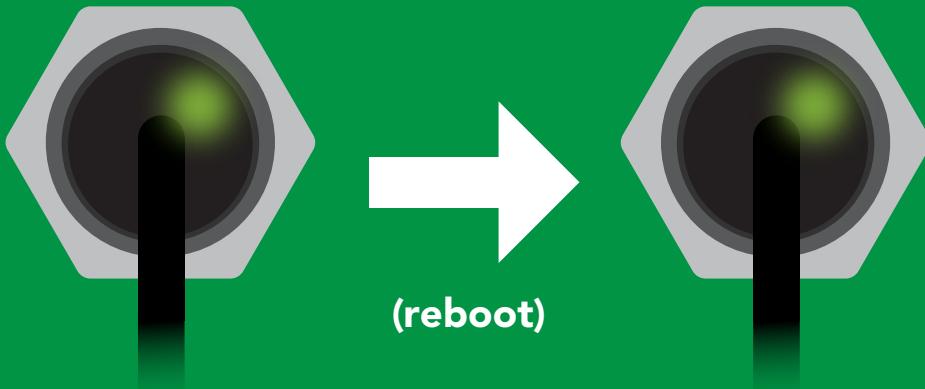
Factory <cr> enable factory reset

## Example Response

Factory <cr>

\*OK <cr>

Factory <cr>



\*OK <cr>

\*RS <cr>  
\*RE <cr>

Baud rate will not change

# Change to I<sup>2</sup>C mode

## Command syntax

Default I<sup>2</sup>C address 112 (0x70)

I<sup>2</sup>C,n <cr> sets I<sup>2</sup>C address and reboots into I<sup>2</sup>C mode

n = any number 1 – 127

## Example      Response

I<sup>2</sup>C,100 <cr>

\*OK (reboot in I<sup>2</sup>C mode)

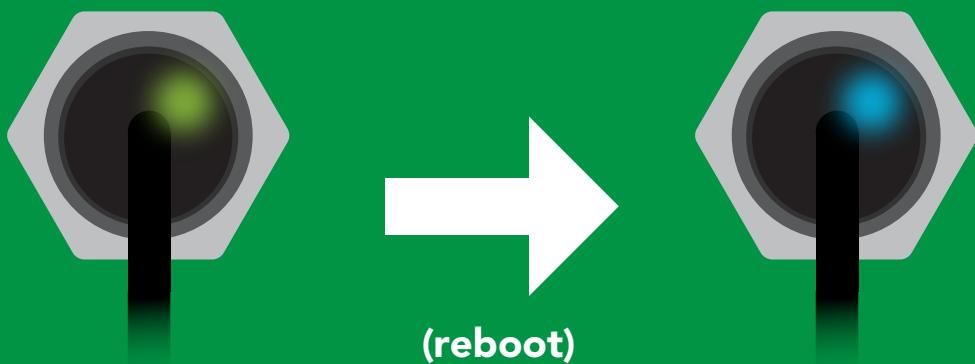
## Wrong example

I<sup>2</sup>C,139 <cr> n > 127

## Response

\*ER <cr>

I<sup>2</sup>C,100



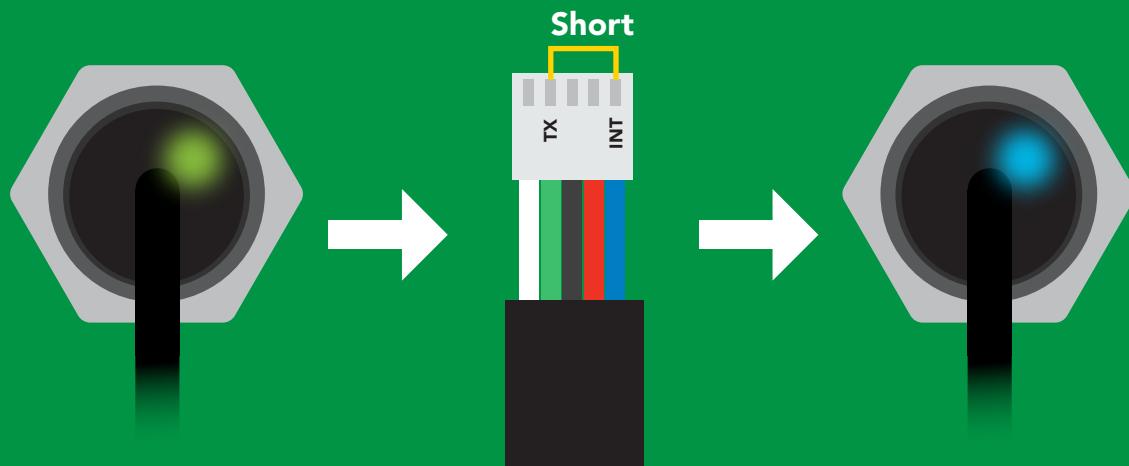
Green  
\*OK <cr>

Blue  
now in I<sup>2</sup>C mode

# Manual switching to I<sup>2</sup>C

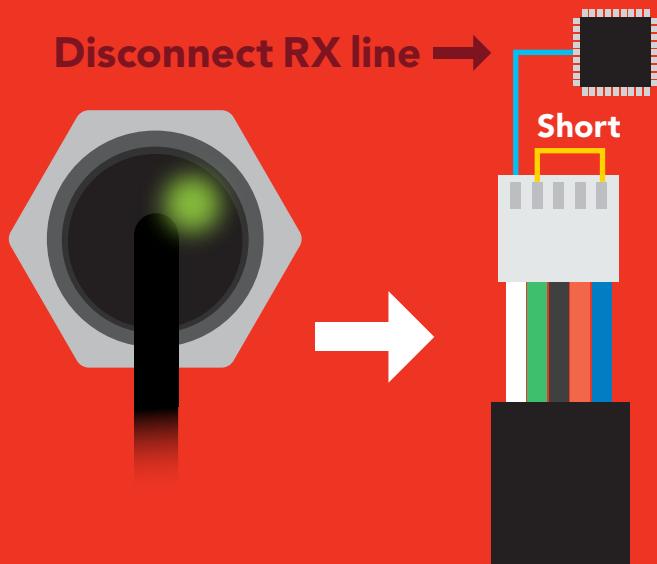
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to INT
- Confirm RX is disconnected
- Connect ground (power on)
- Wait for LED to change from Green to Blue
- Disconnect ground (power off)
- Reconnect all data and power

Manually switching to I<sup>2</sup>C will set the I<sup>2</sup>C address to 112 (0x70)



## Wrong Example

Disconnect RX line →



# I<sup>2</sup>C mode

The I<sup>2</sup>C protocol is **considerably more complex** than the UART (RS-232) protocol. Atlas Scientific assumes the embedded systems engineer understands this protocol.

To set your EZO™ device into I<sup>2</sup>C mode [click here](#)

## Settings that are retained if power is cut

Automatic color matching  
Calibration  
Change I<sup>2</sup>C address  
Hardware switch to UART mode  
LED control  
Protocol lock  
Software switch to UART mode

## Settings that are **NOT** retained if power is cut

Sleep mode

# I<sup>2</sup>C mode

I<sup>2</sup>C address (0x01 – 0x7F)  
112 (0x70) default

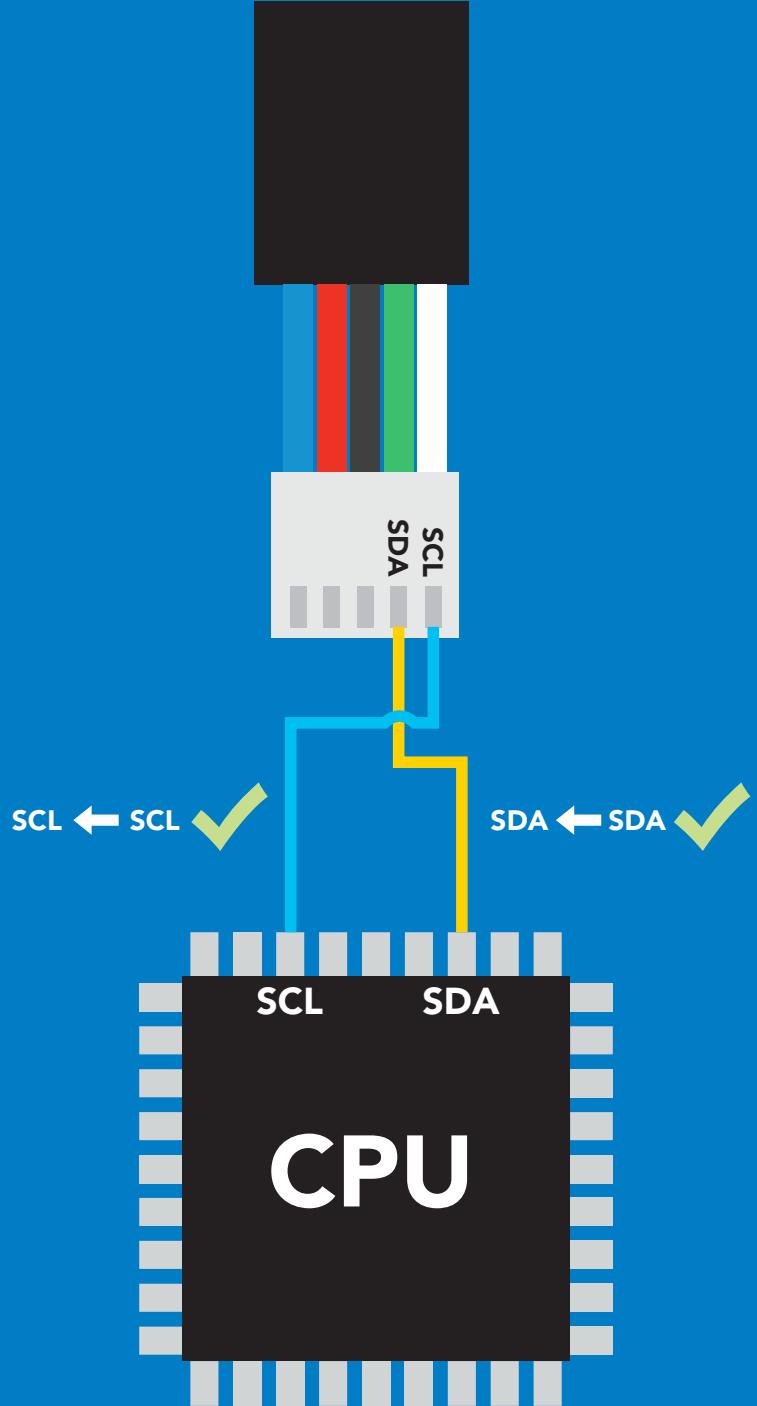
V<sub>cc</sub> 3.3V – 5.5V

Clock speed 100 – 400 kHz

SDA

SCL

0V VCC



## Data format

Units RGB, LUX, & CIE  
Encoding ASCII  
Format string  
Terminator carriage return

Data type integer & floating point  
Decimal places 3  
Smallest string 4 characters  
Largest string 52 characters

# Sending commands to device

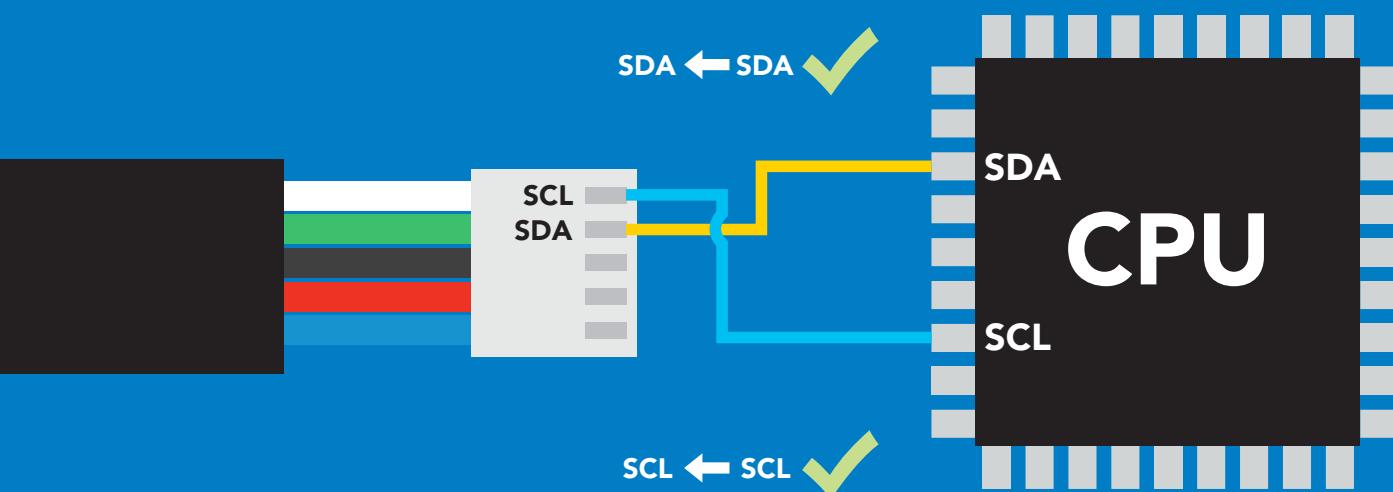
5 parts



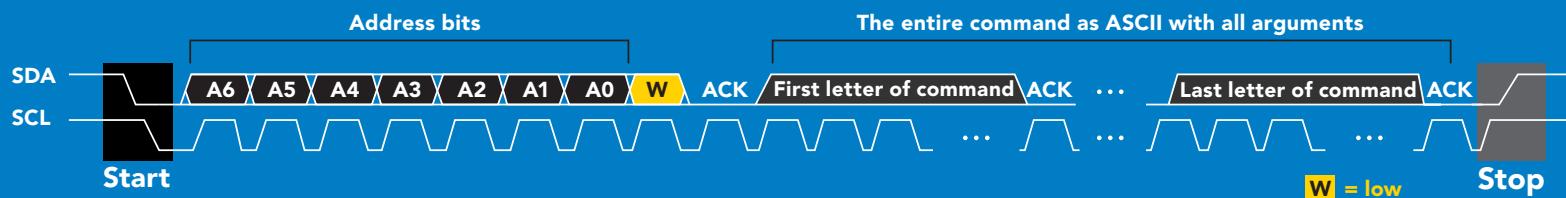
## Example

**Start**    **112 (0x70)**    **Write**    **Sleep**    **Stop**

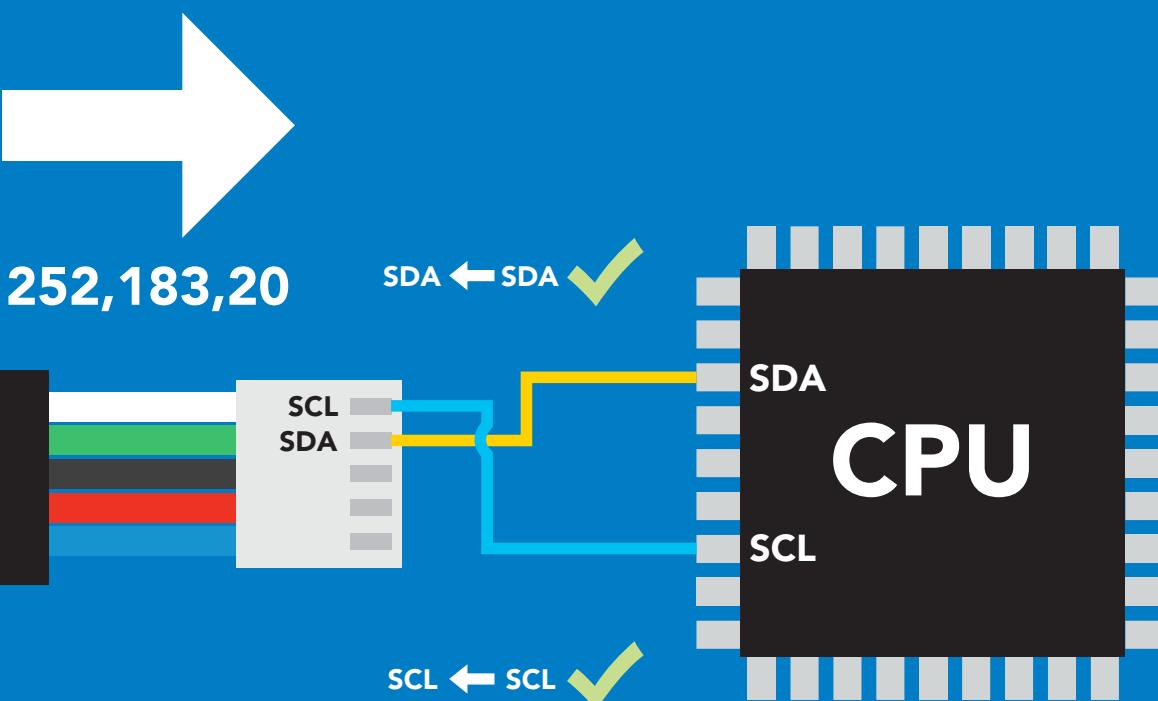
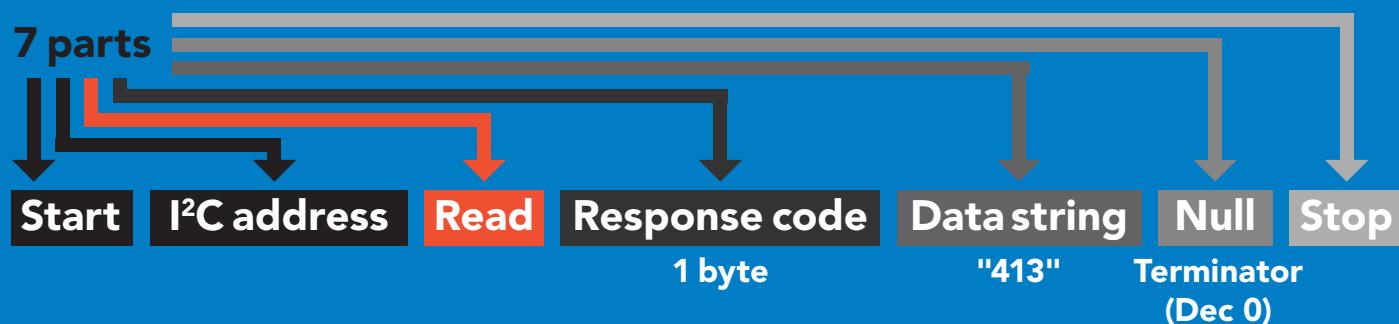
I<sup>2</sup>C address                      Command



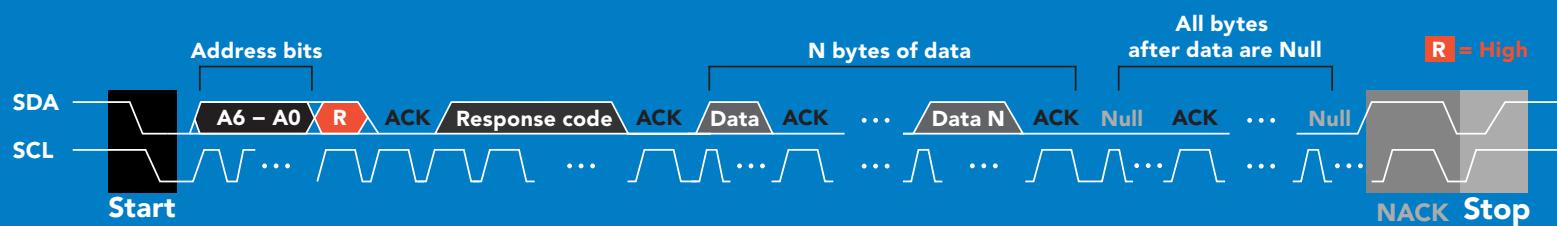
## Advanced



# Requesting data from device



# Advanced

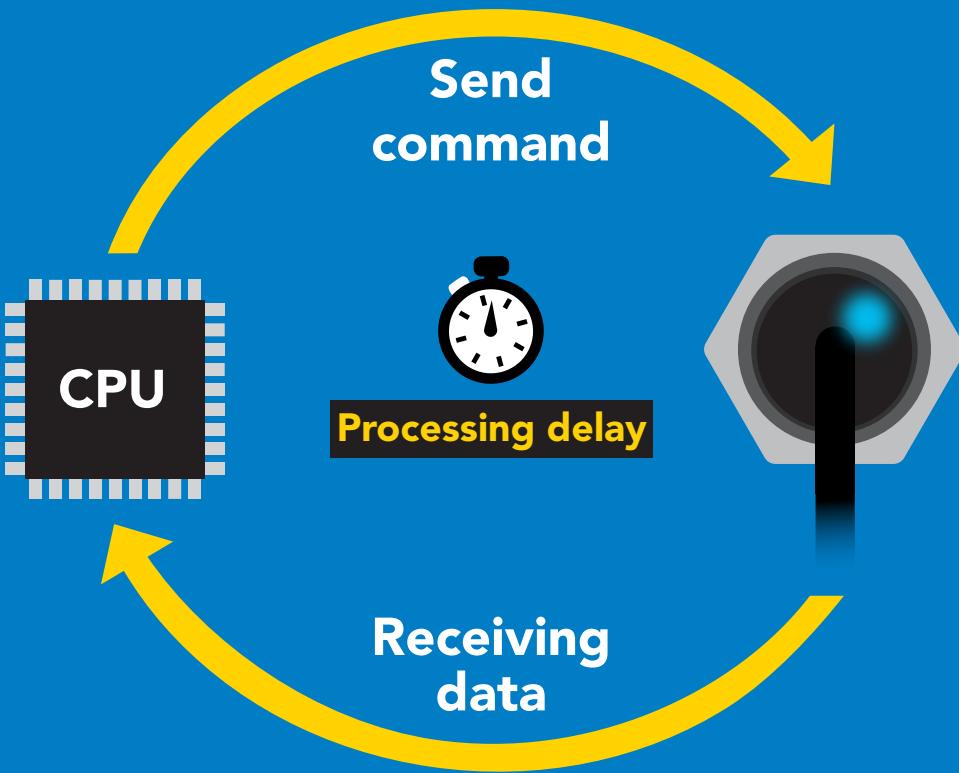


**1 50 53 50 44 49 56 51 44 50 48 0 =252,183,20**

# Response codes & processing delay

After a command has been issued, a 1 byte response code can be read in order to confirm that the command was processed successfully.

*Reading back the response code is completely optional, and is not required for normal operation.*



## Example

```
I2C_start;  
I2C_address;  
I2C_write(EZO_command);  
I2C_stop;
```

```
delay(300); →  Processing delay
```

```
I2C_start;  
I2C_address;  
Char[ ] = I2C_read;  
I2C_stop;
```

If there is no processing delay or the processing delay is too short, the response code will always be 254.

### Response codes

Single byte, not string

255	no data to send
254	still processing, not ready
2	syntax error
1	successful request

# Indicator LED control



**Blue**

I<sup>2</sup>C standby

**Green**

Taking reading

**Purple**

Changing  
I<sup>2</sup>C address

**Red**

Command  
not understood

**White**

Find

**5V**

LED ON  
**+2.5 mA**

**3.3V**

**+1 mA**

# I<sup>2</sup>C mode

## command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function	
Baud	switch back to UART mode	pg. 63
Cal	performs custom calibration	pg. 53
Factory	enable factory reset	pg. 62
Find	finds device with blinking white LED	pg. 51
G	gamma correction	pg. 54
i	device information	pg. 57
iL	enable/disable indicator LED	pg. 50
I2C	change I <sup>2</sup> C address	pg. 61
L	enable/disable target LED	pg. 49
Name	set/show name of device	pg. 56
O	enable/disable parameters	pg. 55
Plock	enable/disable protocol lock	pg. 60
R	returns a single reading	pg. 52
Sleep	enter sleep mode/low power	pg. 59
Status	retrieve status information	pg. 58

# Target LED control

300ms  processing delay

## Command syntax

% represents the percentage of target LED brightness. (any number from 0–100)

L,% set target LED brightness

L,%,T set target LED brightness/trigger target LED only when a reading is taken (*power saving*)

L,? target LED state on/off?

## Example

L,32



1  
Dec 0  
Null

target LED set to 32% brightness.

L,14,T



1  
Dec 0  
Null

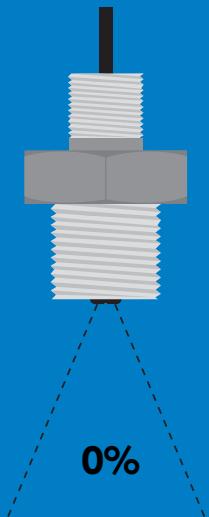
target LED set to 14% brightness, and will only turn on when a reading is taken.

L,?

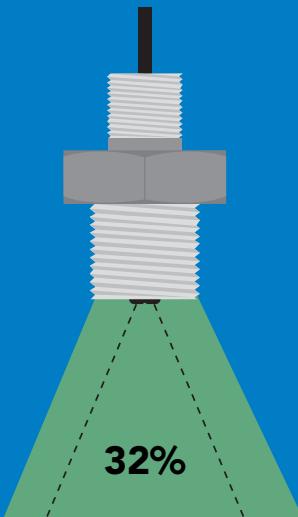


1 ?L, %, [ T ] 0  
Dec ASCII Null

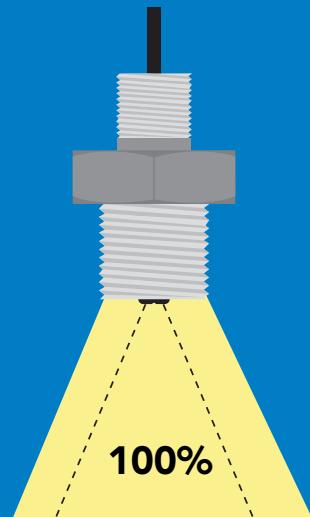
L,0



L,32



L,100



# Indicator LED control

## Command syntax

300ms  processing delay

- iL,1    indicator LED on    default
- iL,0    Indicator LED off
- iL,?    Indicator LED state on/off?

## Example

iL,1

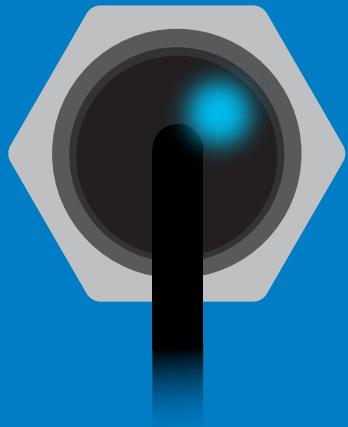
 Wait 300ms  
1 Dec 0 Null

iL,0

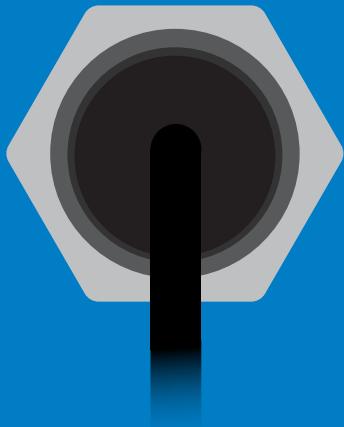
 Wait 300ms  
1 Dec 0 Null

iL,?

 Wait 300ms  
1 Dec ?iL,1 ASCII 0 or  Wait 300ms  
1 Dec ?iL,0 ASCII 0 Null



iL,1



iL,0

# Find

## Command syntax

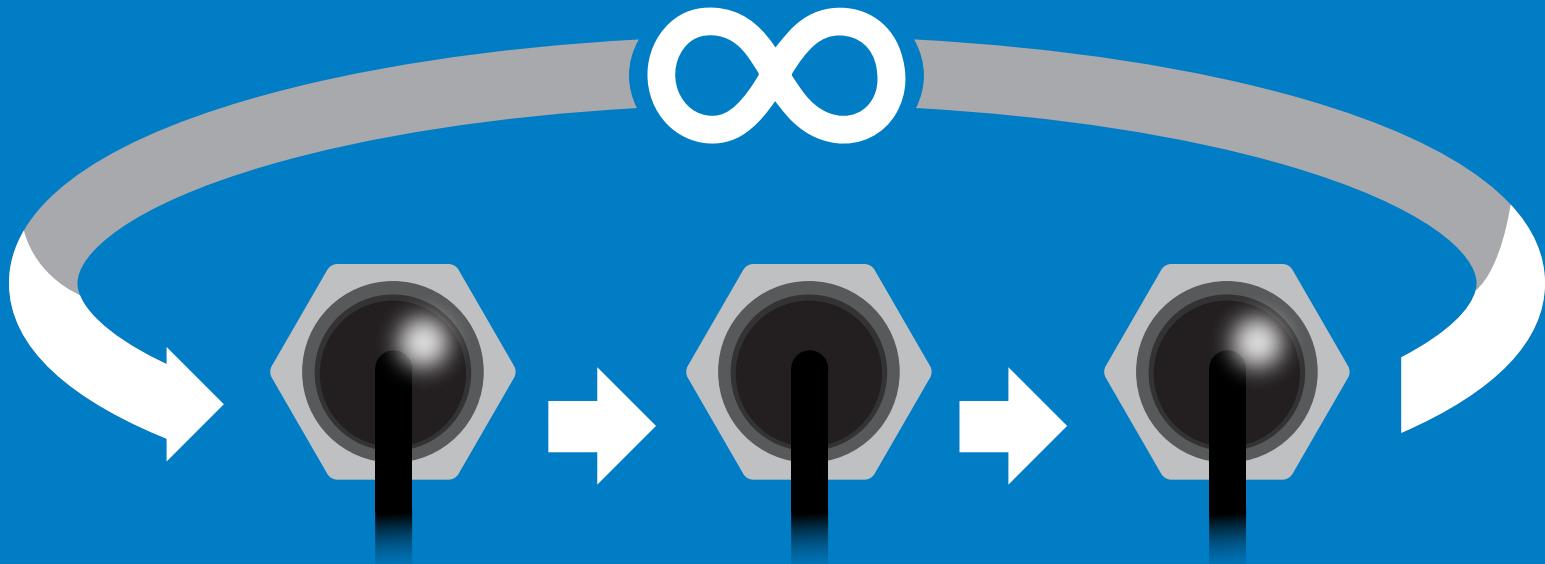
300ms  processing delay

Find LED rapidly blinks white, used to help find device

## Example Response

Find

 Wait 300ms  
1 Dec 0 Null



# Taking reading

Command syntax

300ms  processing delay

R return 1 reading

Example

Response

R



1  
Dec

R,G,B  
ASCII

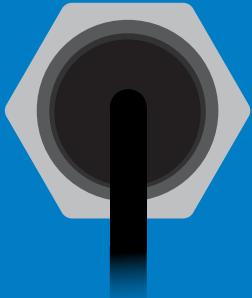
0  
Null

Wait 300ms

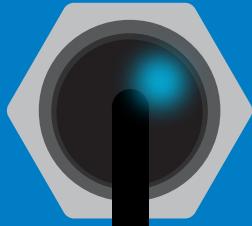


Green

Taking reading



Transmitting



Cyan

Standby

# Calibration

## Command syntax

300ms  processing delay

**Cal** calibrates the EZO-RGB™

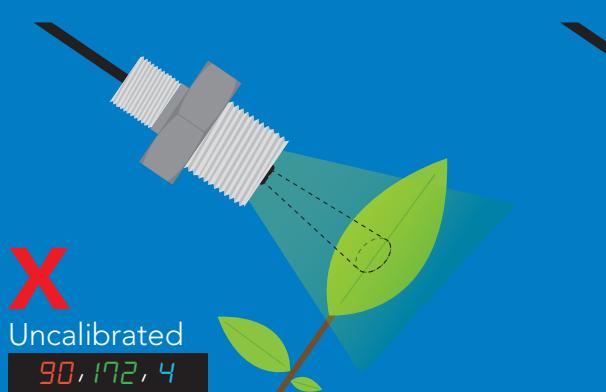
1. place white object (such as a piece of paper) in front of target
2. Issue "cal" command

## Example

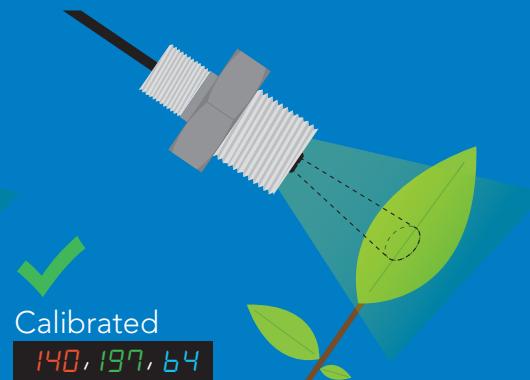
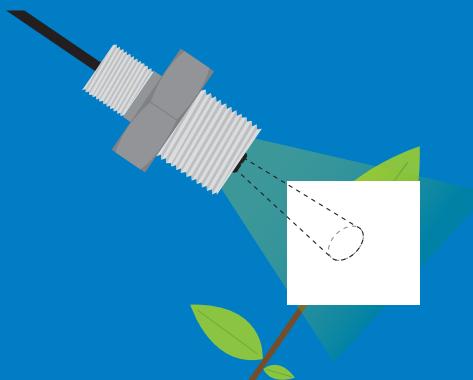
## Response

Cal

 Wait 300ms  
1 Dec 0 Null



Cal



# Gamma correction

300ms  processing delay

## Command syntax

**G,n** set gamma correction

where n = a floating point number from 0.01 – 4.99

**G,?** gamma correction value?

Adjusting the gamma correction helps  
adjust the color seen by the sensor.

The default gamma correction is 1.00 which represents no correction at all.  
A gamma correction factor is a floating point number from 0.01 to 4.99.

## Example

**G,1.99**

## Response

	1	0
Wait 300ms	Dec	Null

	1	?G,1.99	0
Wait 300ms	Dec	ASCII	Null

**G,?**

# Enable/disable parameters from output string

## Command syntax

O, [parameter],[1,0]

enable or disable output parameter

O,?

enabled parameter?

## Example

O,RGB,1 / O,RGB,0

## Response



enable / disable RGB

O,LUX,1 / O,LUX,0



enable / disable lux

O,CIE,1 / O,CIE,0



enable / disable CIE

O,?



if all enabled

## Parameters

RGB red, green, blue

LUX illuminance

CIE CIE 1931 color space

## Followed by 1 or 0

1 enabled

0 disabled

\* If you disable all possible data types your readings will display "no output".

300ms  processing delay

## Command syntax

Do not use spaces in the name

Name,n    set name

n =

— 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 —

Name,    clears name

Up to 16 ASCII characters

Name,?    show name

## Example

## Response

Name,



1  
Dec  
0  
Null

name has been cleared

Name,zzt



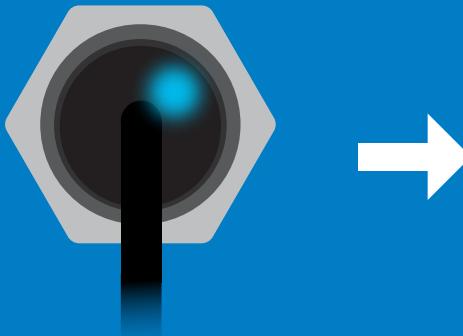
1  
Dec  
0  
Null

Name,?

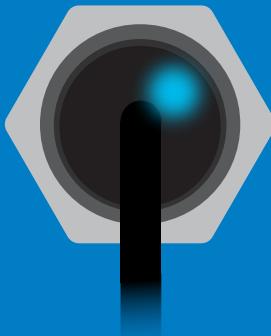


1  
Dec  
?Name,zzt  
ASCII  
0  
Null

Name,zzt



Name,?



1 0

1 ?Name,zzt 0

# Device information

## Command syntax

300ms  processing delay

i device information

Example Response

i



Wait 300ms

1  
Dec

?i,RGB,2.1  
ASCII

0  
Null

## Response breakdown

?i, RGB, 2.1  
↑      ↑  
Device Firmware

# Reading device status

## Command syntax

300ms  processing delay

Status    voltage at Vcc pin and reason for last restart

## Example      Response

Status



Wait 300ms

1

?Status,P,5.038

Dec

ASCII

0

Null

## Response breakdown

?Status,      P,      5.038  
↑               ↑  
Reason for restart      Voltage at Vcc

### Restart codes

P	powered off
S	software reset
B	brown out
W	watchdog
U	unknown

# Sleep mode/low power

## Command syntax

**Sleep    enter sleep mode/low power**

Send any character or command to awaken device.

## Example                  Response

**Sleep**

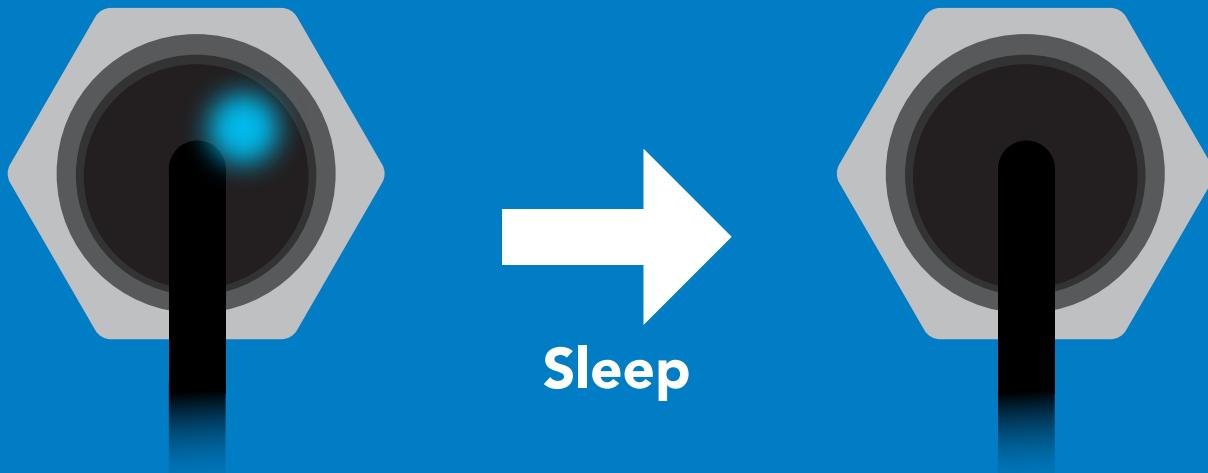
**no response**

Do not read status byte after issuing sleep command.

**Any command**

**wakes up device**

	STANDBY	SLEEP
<b>5V</b>	<b>45 mA</b>	<b>3.4 mA</b>
<b>3.3V</b>	<b>42 mA</b>	<b>3.0 mA</b>



**Standby**

**Sleep**

# Protocol lock

## Command syntax

300ms  processing delay

Plock,1 enable Plock

Locks device to I<sup>2</sup>C mode.

Plock,0 disable Plock **default**

Plock,? Plock on/off?

## Example

Plock,1

 Wait 300ms

**1** Dec **0** Null

Plock,0

 Wait 300ms

**1** Dec **0** Null

Plock,?

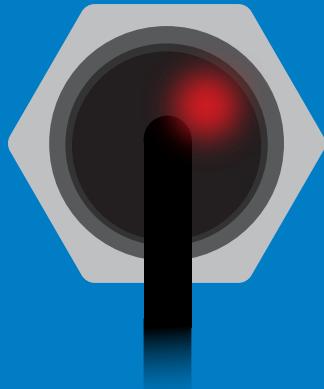
 Wait 300ms

**1** Dec **?Plock,1** ASCII **0** Null

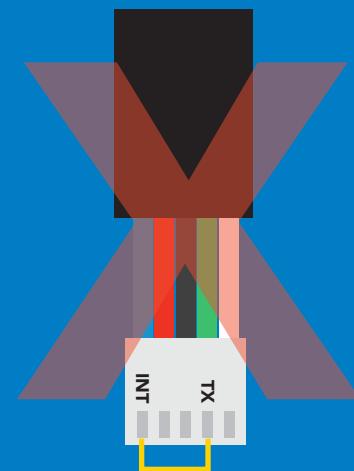
Plock,1



Baud, 9600



cannot change to UART



cannot change to UART

# I<sup>2</sup>C address change

## Command syntax

I2C,n    sets I<sup>2</sup>C address and reboots into I<sup>2</sup>C mode

### Example      Response

I2C,101

device reboot

(no response given)

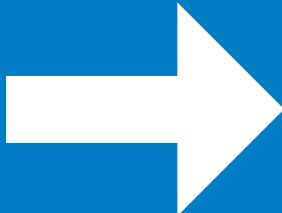
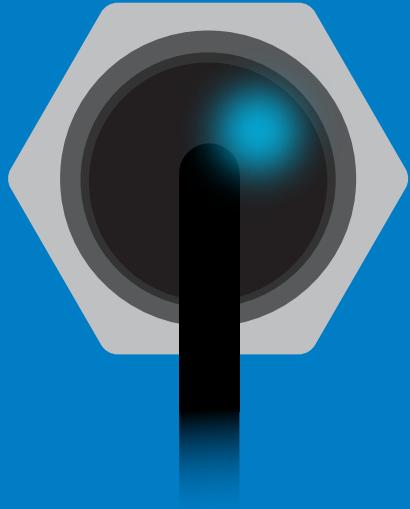
### Warning!

Changing the I<sup>2</sup>C address will prevent communication between the circuit and the CPU until the CPU is updated with the new I<sup>2</sup>C address.

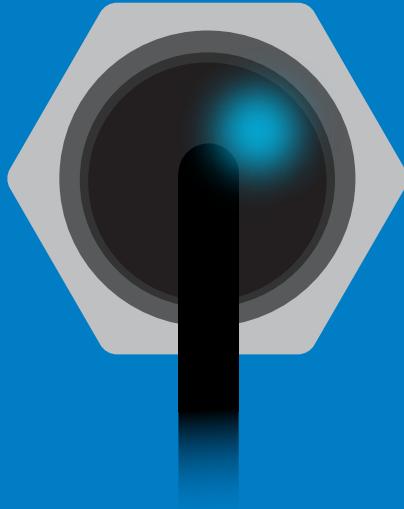
Default I<sup>2</sup>C address is 112 (0x70).

n = any number 1 – 127

I2C,101



(reboot)



# Factory reset

## Command syntax

Factory reset will not take the device out of I<sup>2</sup>C mode.

Factory enable factory reset

I<sup>2</sup>C address will not change

## Example Response

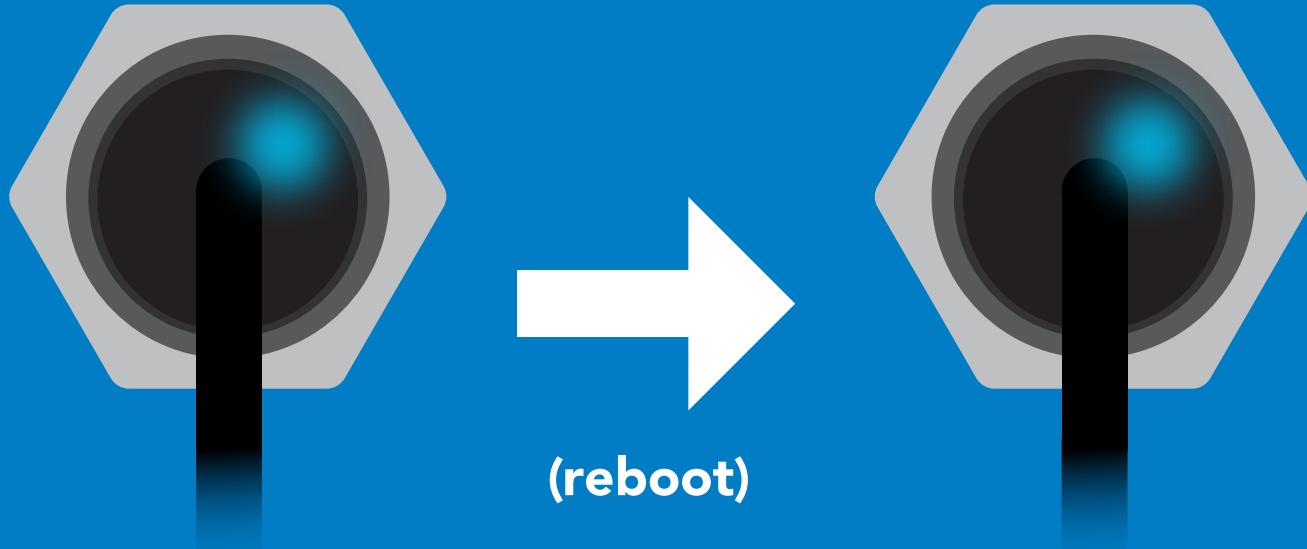
Factory

device reboot

(no response given)

Clears custom calibration  
LED on  
Response codes enabled

Factory



# Change to UART mode

## Command syntax

Baud,n    switch from I<sup>2</sup>C to UART

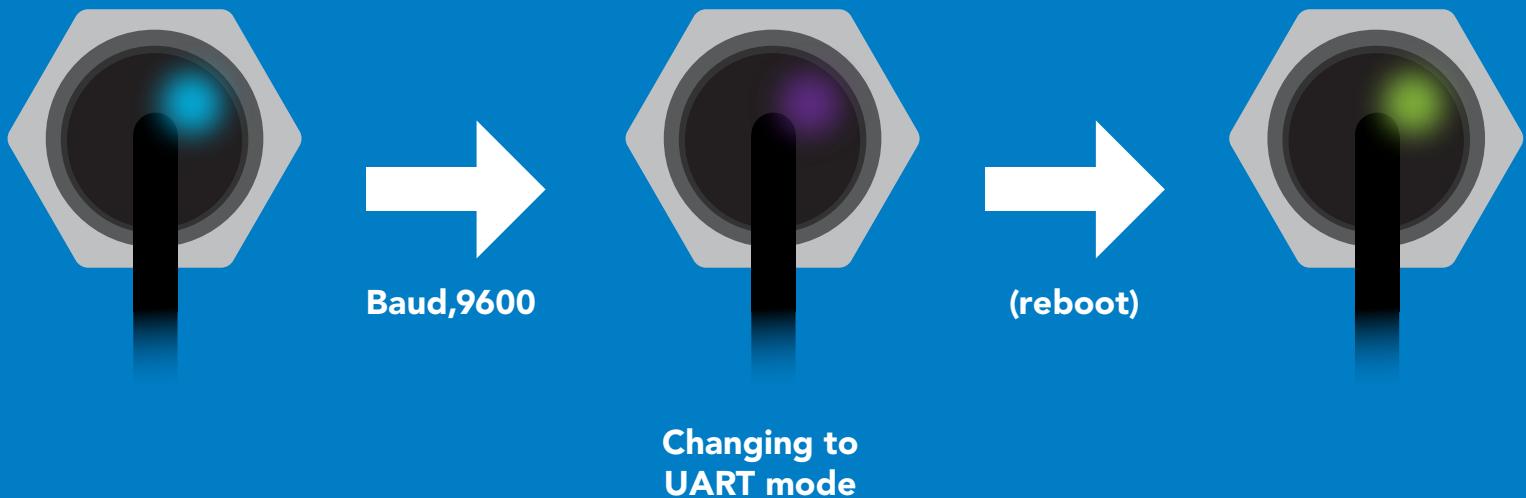
### Example      Response

Baud,9600

reboot in UART mode

(no response given)

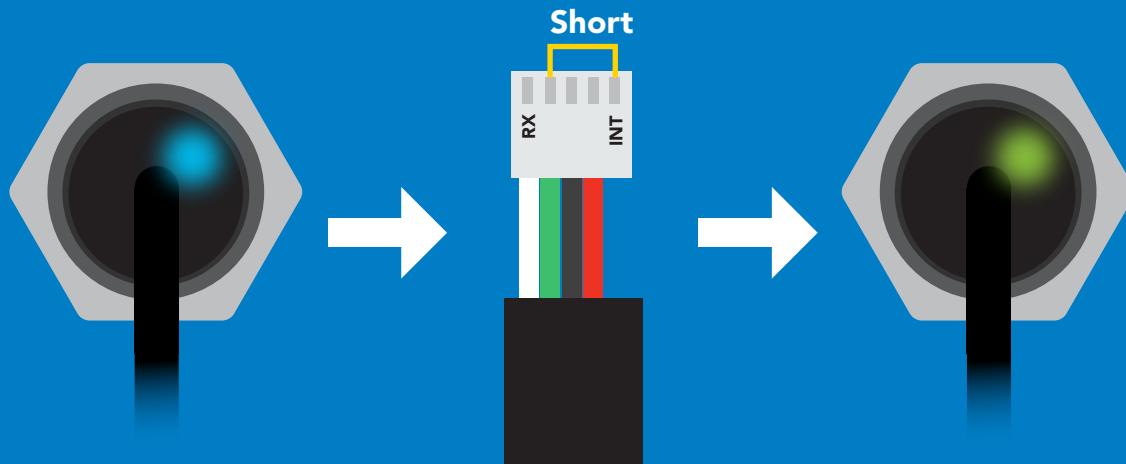
$$n = \left[ \begin{array}{l} 300 \\ 1200 \\ 2400 \\ 9600 \\ 19200 \\ 38400 \\ 57600 \\ 115200 \end{array} \right]$$



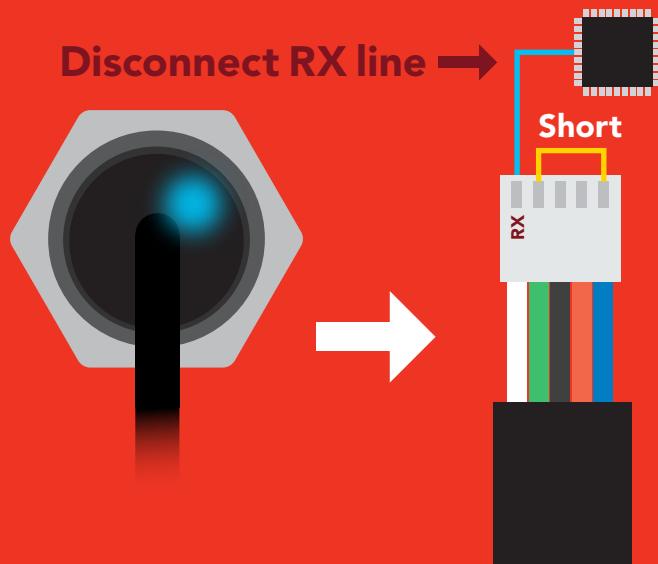
# Manual switching to UART

- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to INT
- Confirm RX is disconnected
- Connect ground (power on)
- Wait for LED to change from **Blue** to **Green**
- Disconnect ground (power off)
- Reconnect all data and power

## Example



## Wrong Example



# Datasheet change log

## Datasheet V 2.9

Revised artwork on page 8.

## Datasheet V 2.8

Revised naming device info on pages 32 & 56.

## Datasheet V 2.7

Removed proximity sensing capabilities from device.

## Datasheet V 2.6

Added new feature info on pg 2.

## Datasheet V 2.5

Corrected typo on pg 54.

## Datasheet V 2.4

Moved Default state to pg 18.

## Datasheet V 2.3

Changed the default I2C Address to 112 (0x70)

## Datasheet V 2.2

Added an I<sup>2</sup>C section to the datasheet.

## Datasheet V 2.1

Revised response for the sleep command in UART mode on pg 39.

## Datasheet V 2.0

Revised entire datasheet

# Firmware updates

V1.10 – (November 7, 2015)

- Fixed sleep mode bug.

V1.15 – (November 30, 2015)

- Fixed threshold bug.

V1.16 – (February 2, 2016)

- Fixed bug where excessive newline characters would be output for every line.

v1.18 - (Sept 19, 2016)

- Updated manufacturing process.

v1.20 - (June 29, 2017)

- Issuing the I<sup>2</sup>C command will return with an error.

v2.00 - (May 1, 2019)

- Added the RGB indicator LED and I<sup>2</sup>C mode, find command, C,n command

v2.10 - (August 23, 2021)

- Proximity sensing capabilities removed (feature was hardly ever used).

# Warranty

Atlas Scientific™ Warranties the EZO-RGB™ Embedded Color Sensor to be free of defect during the debugging phase of device implementation, or 30 days after receiving the EZO-RGB™ Embedded Color Sensor (which ever comes first).

## The debugging phase

The debugging phase as defined by Atlas Scientific™ is the time period when the EZO-RGB™ Embedded Color Sensor is connected into a bread board, or shield. If the EZO-RGB™ Embedded Color Sensor is being debugged in a bread board, the bread board must be devoid of other components. If the EZO-RGB™ Embedded Color Sensor is being connected to a microcontroller, the microcontroller must be running code that has been designed to drive the EZO-RGB™ Embedded Color Sensor exclusively and output the EZO-RGB™ Embedded Color Sensor data as a serial string.

**It is important for the embedded systems engineer to keep in mind that the following activities will void the EZO-RGB™ Embedded Color Sensor warranty:**

- **Soldering any part to the EZO-RGB™ Embedded Color Sensor.**
- **Running any code, that does not exclusively drive the EZO-RGB™ Embedded Color Sensor and output its data in a serial string.**
- **Embedding the EZO-RGB™ Embedded Color Sensor into a custom made device.**
- **Removing any potting compound.**

# Reasoning behind this warranty

Because Atlas Scientific™ does not sell consumer electronics; once the device has been embedded into a custom made system, Atlas Scientific™ cannot possibly warranty the EZO-RGB™ Embedded Color Sensor, against the thousands of possible variables that may cause the EZO-RGB™ Embedded Color Sensor to no longer function properly.

## Please keep this in mind:

- 1. All Atlas Scientific™ devices have been designed to be embedded into a custom made system by you, the embedded systems engineer.**
- 2. All Atlas Scientific™ devices have been designed to run indefinitely without failure in the field.**
- 3. All Atlas Scientific™ devices can be soldered into place, however you do so at your own risk.**

Atlas Scientific™ is simply stating that once the device is being used in your application, Atlas Scientific™ can no longer take responsibility for the EZO-RGB™ Embedded Color Sensor continued operation. This is because that would be equivalent to Atlas Scientific™ taking responsibility over the correct operation of your entire device.