

EZO-HUMTM

Embedded Humidity sensor

Reads	Relative humidity Dew point Air temperature
Range	0 – 100%
Calibration	Factory calibrated
Response time	1 reading per second <small>(UART mode)</small> 1 reading per 300 milliseconds <small>(I2C mode)</small>
Accuracy	+/- 2%
Connector	5 lead data cable
Cable length	1 meter
Data protocol	UART & I²C
Default I2C address	111 (0x6F)
Data format	ASCII
Operating voltage	3.3V – 5V
IP rating	IP67
Life expectancy	10 years

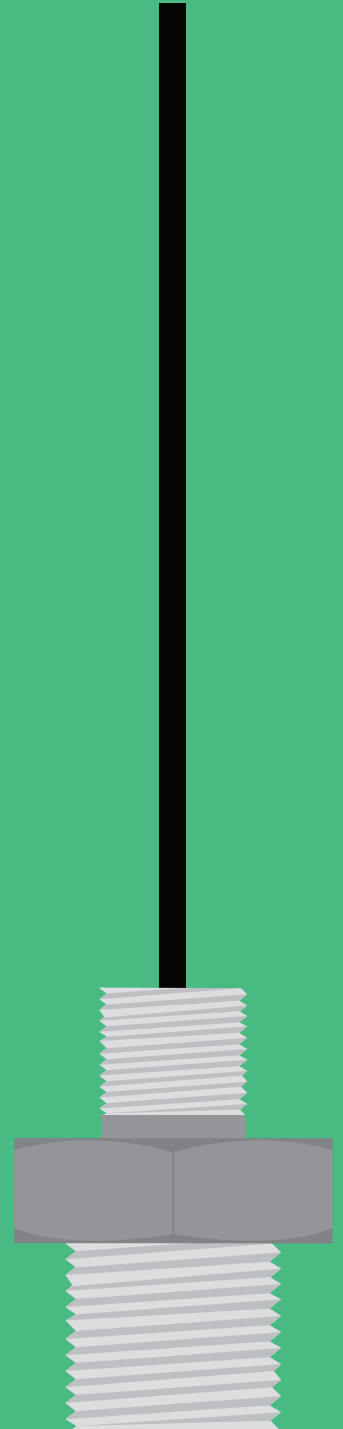


Table of contents

Physical properties	4	Absolute max ratings	5
Pin out	4	Calibration theory	5
Power consumption	5	Custom calibration	5

UART

UART mode	9
Receiving data from device	10
Sending commands to device	11
LED color definition	12
UART quick command page	13
LED control	14
Find	15
Continuous mode	16
Single reading mode	17
Auto monitor	18
Enable/disable parameters	19
Naming device	20
Device information	21
Response codes	22
Reading device status	23
Sleep mode/low power	24
Change baud rate	25
Protocol lock	26
Factory reset	27
Change to I ² C mode	28
Manual switching to I ² C	29

I²C

I ² C mode	31
Sending commands	32
Requesting data	33
Response codes	34
Processing delay	34
LED color definition	35
I²C quick command page	36
LED control	37
Find	38
Taking reading	39
Auto monitor	40
Enable/disable parameters	41
Naming device	42
Device information	43
Reading device status	44
Sleep mode/low power	45
Protocol lock	46
I ² C address change	47
Factory reset	48
Change to UART mode	49
Manual switching to UART	50

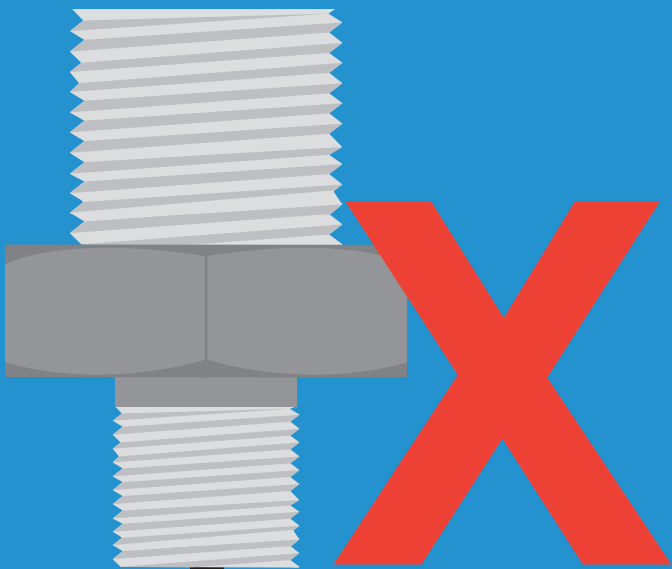
Datasheet change log	51
Firmware updates	51
Warranty	52

Attention

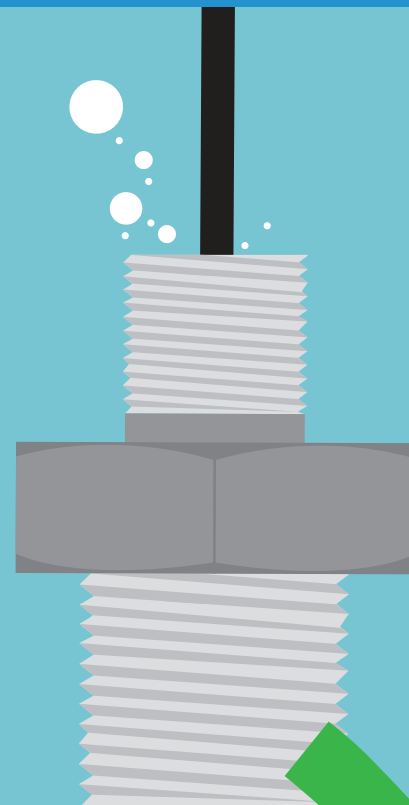
The EZO-HUM™ is 100% operational out of the box.
CALIBRATION IS UNNECESSARY

Direct sunlight will heat the sensor above the air temperature, making the readings incorrect.

Can the sensor get wet?

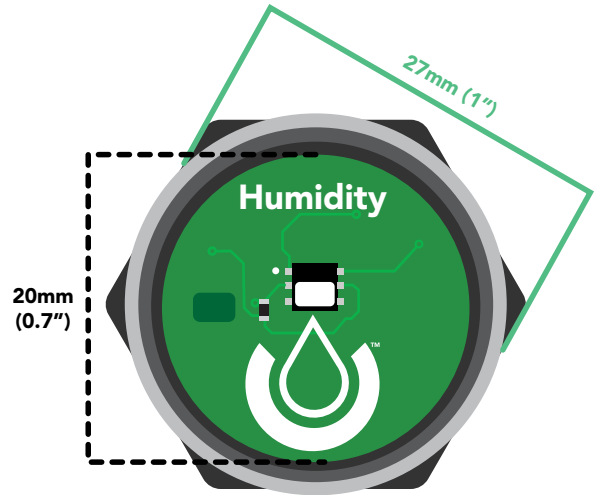
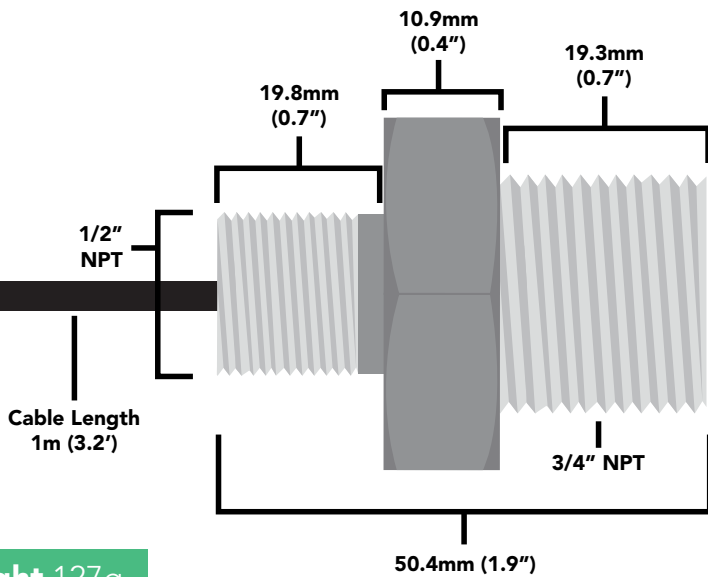


Don't do that



Yes, readings will be $>100\%$ when wet and will return to normal once dry.

Physical properties



Weight 127g

Body 316 Stainless Steel



Pin out

Data and power cable pinout

White - RX/SCL
Green - TX/SDA
Black - GND
Red - VCC
Blue - AUTO



The auto monitor pin will go high when a set humidity has been reached.

57.38%

0V

VCC

*Auto monitor set to 57.38%

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

See page 18 to enable auto-monitoring in UART mode.

See page 40 to enable auto-monitoring in I2C mode.

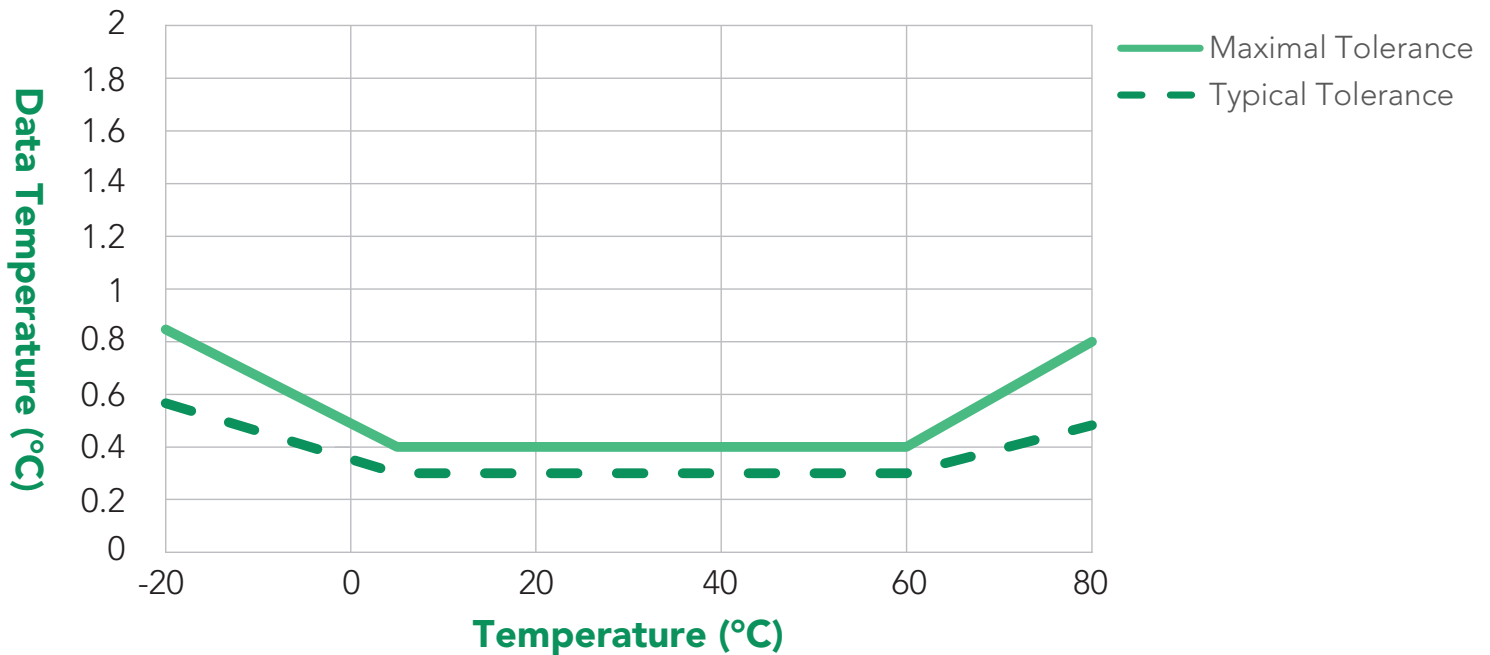
Power consumption

	LED	MAX	SLEEP
5V	ON	2.6 mA	0.5 mA
	OFF	2.4 mA	
3.3V	ON	2.2 mA	0.3 mA
	OFF	2.0 mA	

Absolute max ratings

Parameter	MIN	TYP	MAX
Storage temperature	-30 °C		75 °C
Operational temperature	-20 °C	25 °C	80 °C
VCC	3.3V	3.3V	5.5V

Air temperature



Calibration theory

The Atlas Scientific EZO-HUM™ Embedded Humidity Sensor comes pre-calibrated. The factory calibration data is permanently stored in the circuit and cannot be erased.

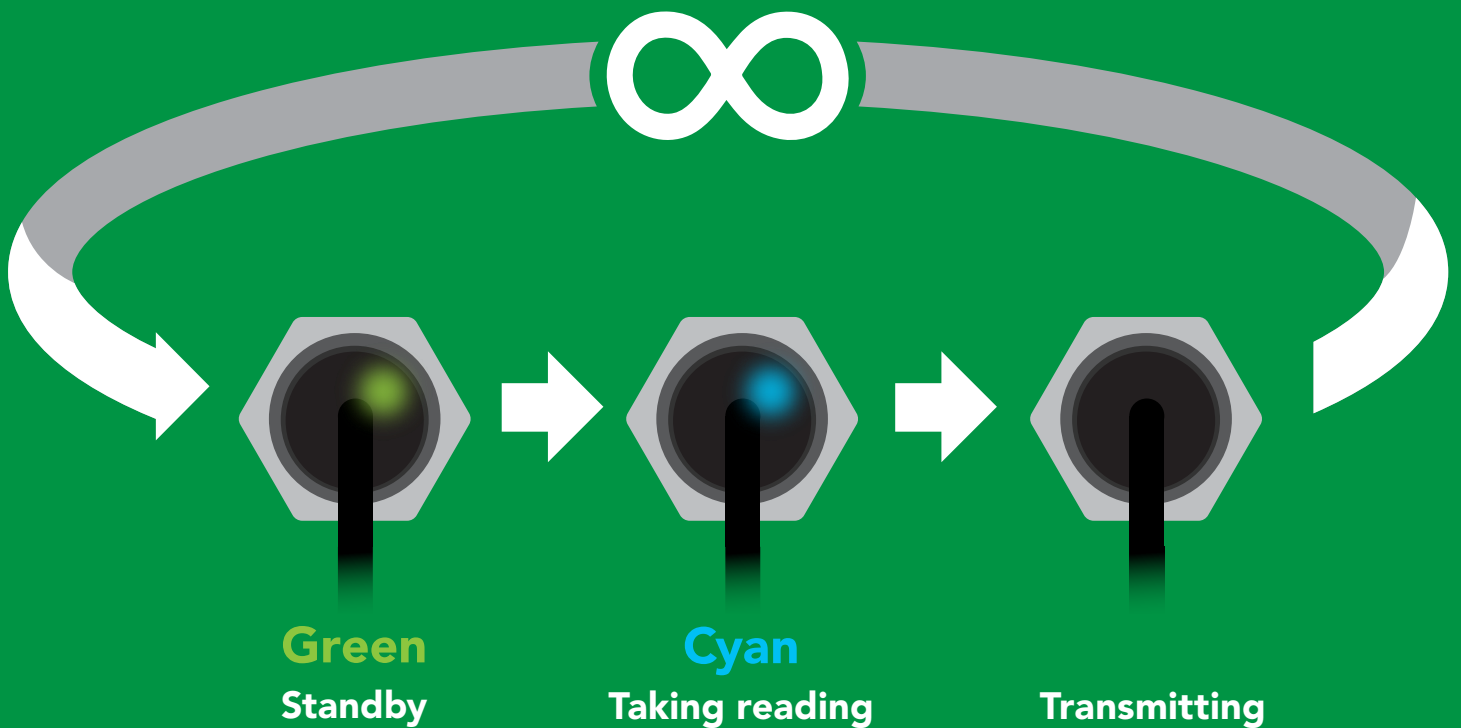
Custom calibration

This circuit does not require recalibration, and does not offer onboard custom calibration.

Default state

UART mode

Baud	9,600
Readings	continuous
Speed	1 second
LED	on



✓ Available data protocols

UART

default

I²C

✗ Unavailable data protocols

SPI

Analog

RS-485

Mod Bus

4–20mA

UART mode

Settings that are retained if power is cut

- Auto monitor
- Baud rate
- Continuous mode
- Device name
- Enable/disable parameters
- Enable/disable response codes
- Hardware switch to I2C mode
- LED control
- Protocol lock
- Software switch to I2C mode

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

- Sleep mode

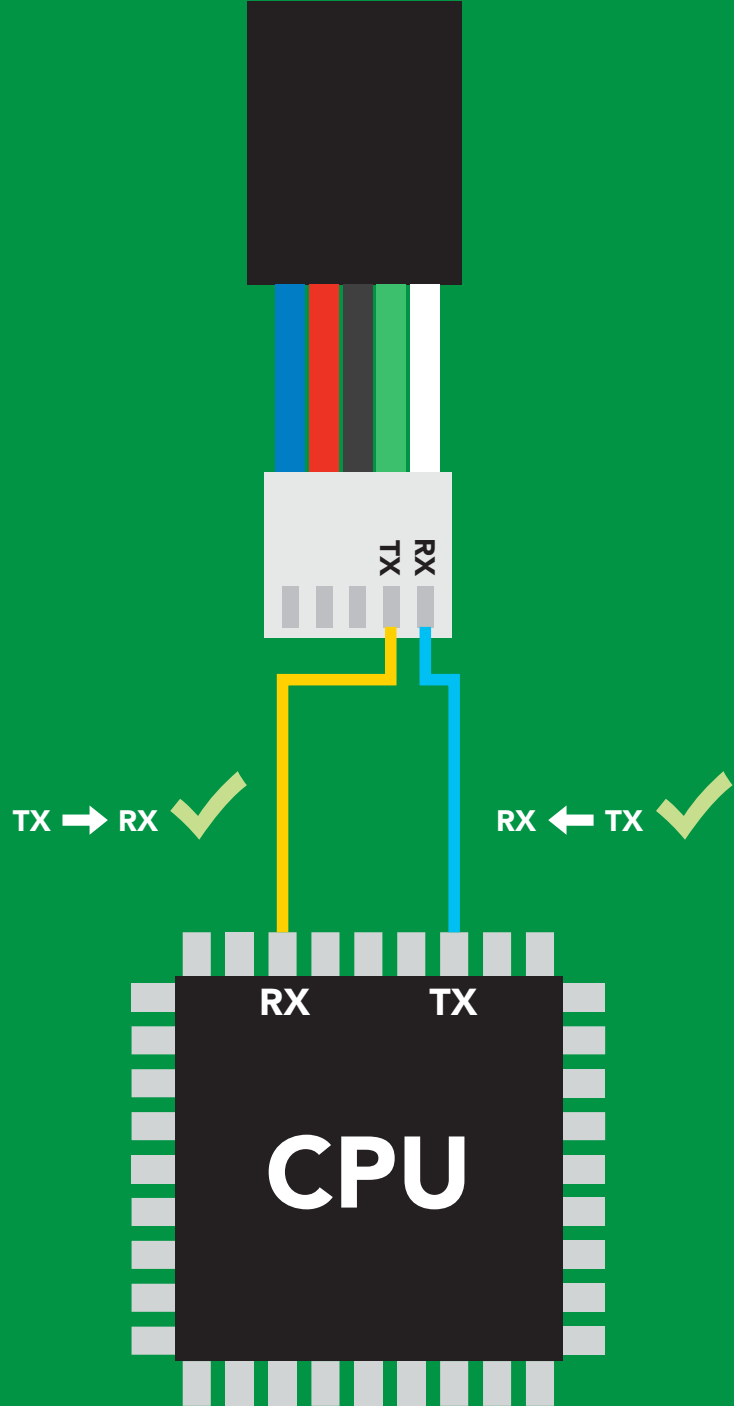
UART mode

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



Vcc 3.3V – 5V



Data format

Reading Humidity
Air Temperature
Dew point

Units % Relative humidity
Air Temperature °C (when enabled)
Dew point Temperature °C (when enabled)

Encoding ASCII (CSV string if temp/
dew point enabled)

Terminator carriage return

Data type floating point

Decimal places 2

Smallest string 4 characters

Largest string 24 characters

Receiving data from device

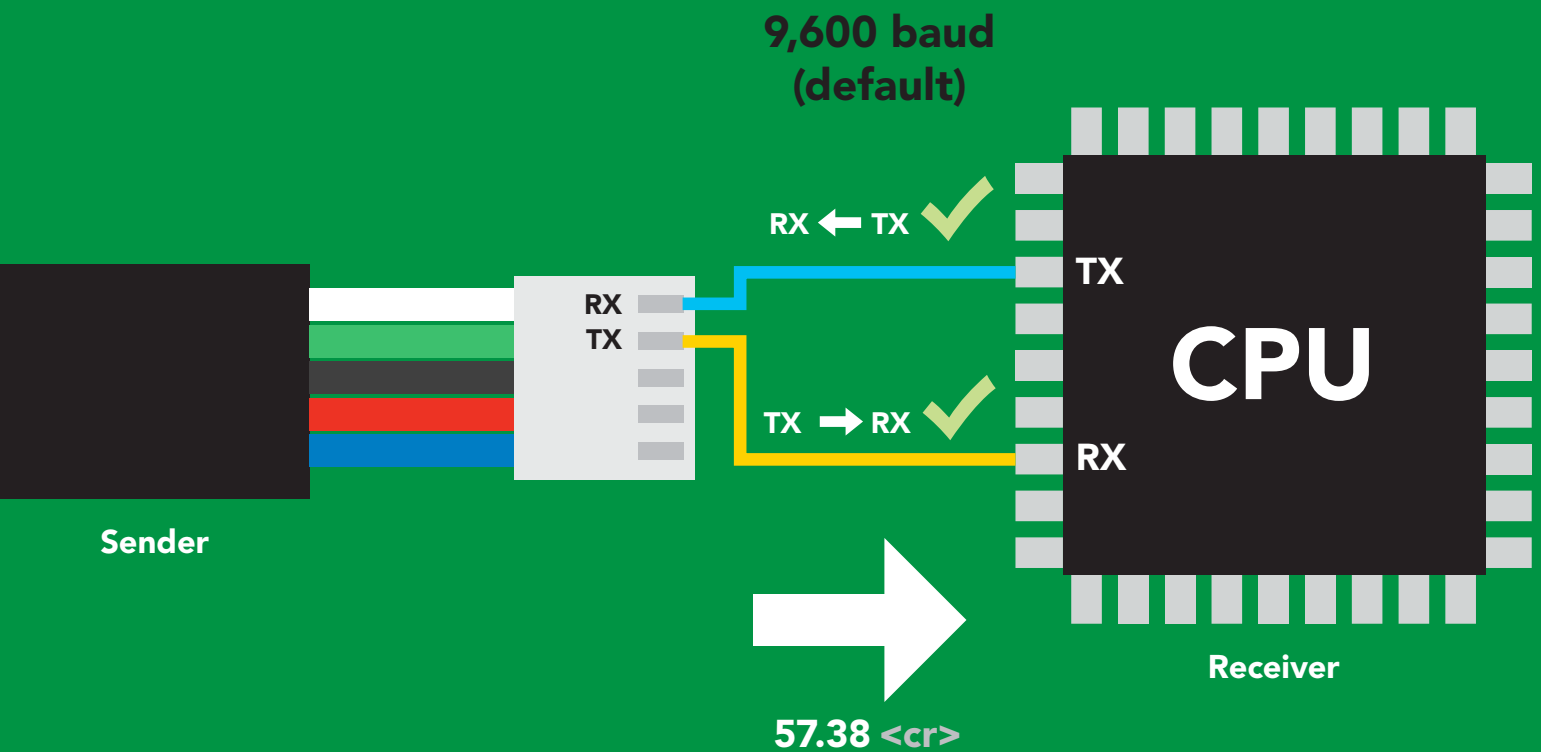
2 parts

ASCII data string

Command

Carriage return <cr>

Terminator



Advanced

ASCII: 5 7 . 3 8 <cr>

Hex: 35 37 2E 33 38 0D

Dec: 53 55 46 51 56 13

Sending commands to device

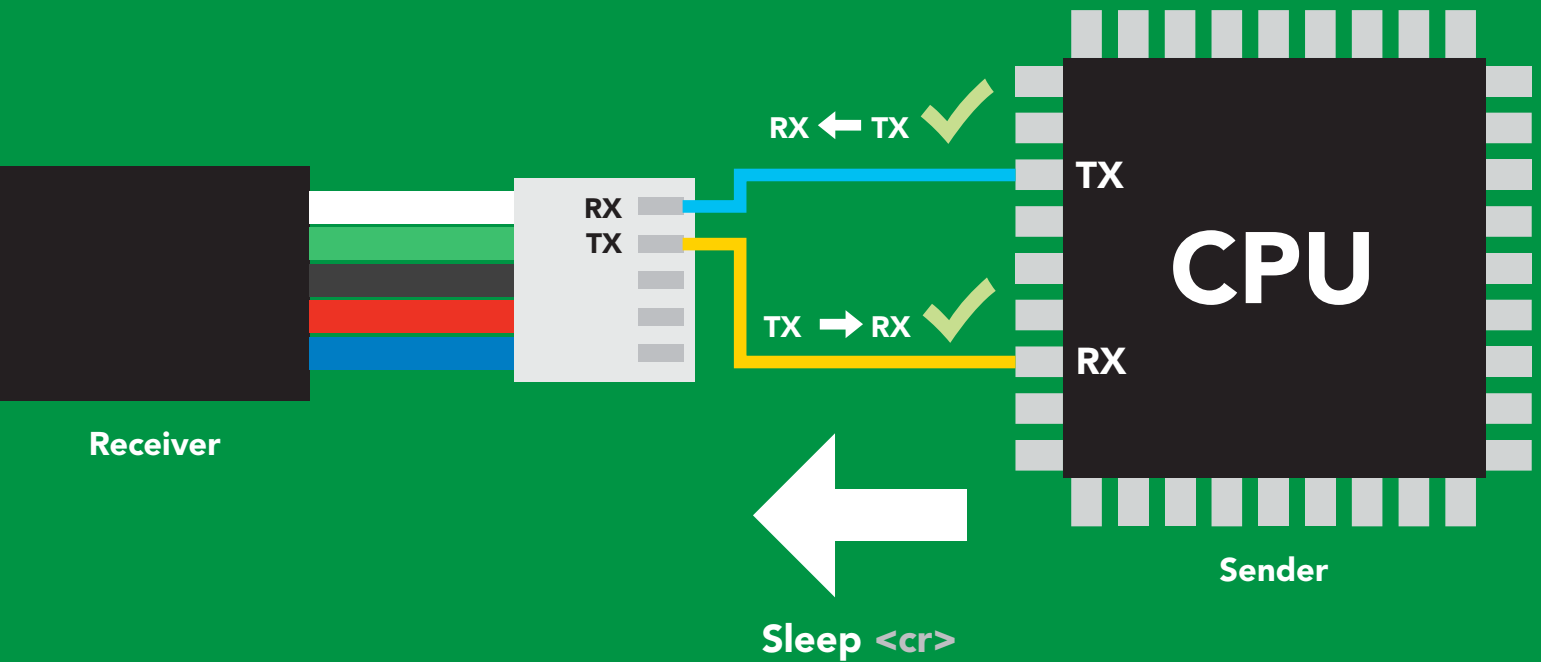
2 parts

Command (not case sensitive)

ASCII data string

Carriage return <cr>

Terminator



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



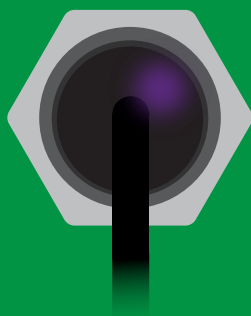
Green

UART standby



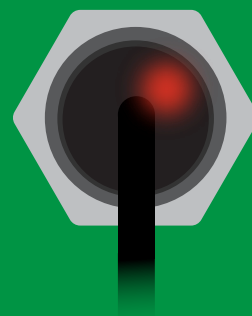
Cyan

Taking reading



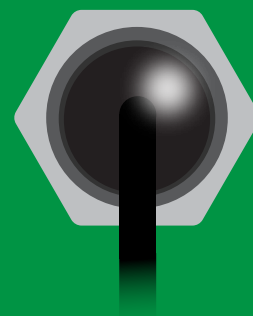
Purple

Changing
I²C address



Red

Command
not understood



White

Find

5V

LED ON
+0.2 mA

3.3V

+0.2 mA

UART mode

command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function		Default state
Auto	enable/disable auto monitor	pg. 18	disabled
Baud	change baud rate	pg. 25	9,600
C	enable/disable continuous mode	pg. 16	enabled
Factory	enable factory reset	pg. 27	n/a
Find	finds device with blinking white LED	pg. 15	n/a
i	device information	pg. 21	n/a
I2C	change to I ² C mode	pg. 28	not set
L	enable/disable LED	pg. 14	enabled
Name	set/show name of device	pg. 20	not set
O	enable/disable parameters	pg. 19	HUM
Plock	enable/disable protocol lock	pg. 26	n/a
R	returns a single reading	pg. 17	n/a
Sleep	enter sleep mode/low power	pg. 24	n/a
Status	Retrieve status information	pg. 23	n/a
*OK	enable/disable response codes	pg. 22	n/a

LED control

Command syntax

L,1 <cr> LED on **default**

L,0 <cr> LED off

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

Example

Response

L,1 <cr>

*OK <cr>

L,0 <cr>

*OK <cr>

L,? <cr>

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



L,1



L,0

Find

Command syntax

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

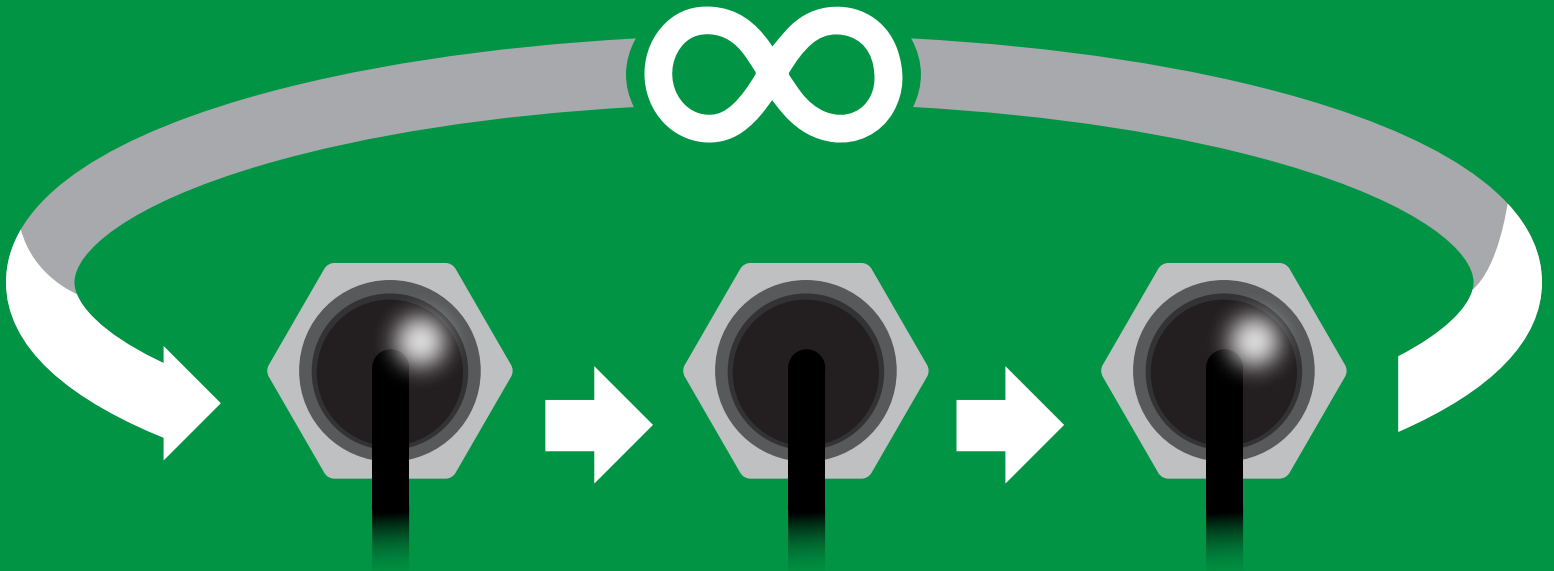
Find <cr> LED rapidly blinks white, used to help find device

Example

Response

Find <cr>

*OK <cr>



Continuous mode

Command syntax

- C,1 <cr>** enable continuous readings once per second **default**
- C,n <cr>** continuous readings every n seconds (n = 2 to 99 sec)
- C,0 <cr>** disable continuous readings
- C,? <cr>** continuous mode settings

Example

Response

C,1 <cr>

***OK <cr>**
HUM (1 sec) <cr>
HUM (2 sec) <cr>
HUM (n sec) <cr>

C,30 <cr>

***OK <cr>**
HUM (30 sec) <cr>
HUM (60 sec) <cr>
HUM (90 sec) <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

R <cr>

Response

57.38 <cr>

*OK <cr>



Green
Standby



Cyan
Taking reading



Transmitting



1 second

Auto monitor

Command syntax

When enabled, the sensor will continuously monitor the readings and set the auto monitor pin high when your value has been reached. When Auto Monitor is enabled, it is not necessary to actively take readings (continuous mode can be disabled).

- Auto,en, [0,1,2]** <cr> 0 = disable, 1= Enable for humidity, 2= Enable for dew point
- Auto,n** <cr> The value that will set the alarm pin
- Auto,tol,n** <cr> The value that will reset the alarm pin
- Auto,?** <cr> Auto monitor settings

Example

Response

Auto,en,1 <cr>

***OK** <cr> Enable humidity automonitoring

Auto,57.38 <cr>

***OK** <cr> Set alarm to go off at 57.38% humidity

Auto,tol,1.2 <cr>

***OK** <cr> The humidity must fall 1.2 percentage points below set point for alarm to reset.

Auto,? <cr>

?,auto,57.38,1.20,1 <cr> if all are enabled



Enable/disable parameters from output string

Command syntax

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

Example

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

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

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

O,? <cr>

Response

*OK <cr> enable / disable humidity

*OK <cr> enable / disable temperature

*OK <cr> enable / disable dew point

?,O,HUM,T,Dew <cr> if all enabled

Parameters

Hum Humidity
T Air temperature in °C
Dew Dew point

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

Name, <cr> clears name

Name,? <cr> show name

n =

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Up to 16 ASCII characters

Example

Response

Name, <cr>

*OK <cr> name has been cleared

Name,zzt <cr>

*OK <cr>

Name,? <cr>

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

Name,zzt <cr>



*OK <cr>

Name,? <cr>



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

Device information

Command syntax

```
i <cr> device information
```

Example

```
i <cr>
```

Response

```
?i,HUM,1.0 <cr>  
*OK <cr>
```

Response breakdown

```
?i, HUM, 1.0  
    ↑    ↑  
  Device Firmware
```

Response codes

Command syntax

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

Example

Response

R <cr>

57.38 <cr>
***OK** <cr>

***OK,0** <cr>

no response, ***OK** disabled

R <cr>

57.38 <cr> ***OK** disabled

***OK,?** <cr>

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

Other response codes

- *ER** unknown command
- *OV** over volt ($VCC \geq 5.5V$)
- *UV** under volt ($VCC \leq 3.1V$)
- *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

```
Status <cr>
```

Response

```
?Status,P,5.038 <cr>  
*OK <cr>
```

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

Send any character or command to awaken device.

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

Example

Response

`Sleep <cr>`

`*OK <cr>`

`*SL <cr>`

`Any command`

`*WA <cr>` wakes up device

5V

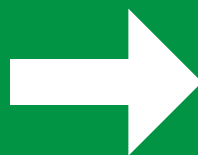
MAX
2.6 mA

SLEEP
0.5 mA

3.3V

2.2 mA

0.4 mA



`Sleep <cr>`



Change baud rate

Command syntax

Baud,n <cr> change baud rate

Example

Baud,38400 <cr>

Response

*OK <cr>

Baud,? <cr>

?Baud,38400 <cr>

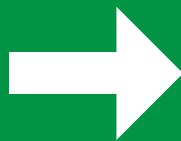
*OK <cr>

n =

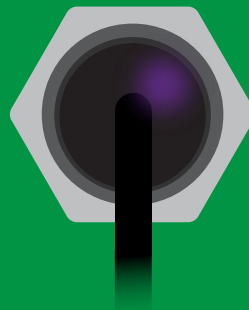
- 300
- 1200
- 2400
- 9600 default**
- 19200
- 38400
- 57600
- 115200



Standby

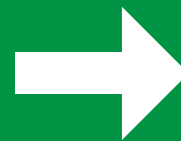


Baud,38400 <cr>



Changing
baud rate

*OK <cr>



(reboot)



Standby

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

Response

`Plock,1 <cr>`

`*OK <cr>`

`Plock,0 <cr>`

`*OK <cr>`

`Plock,? <cr>`

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

`Plock,1`



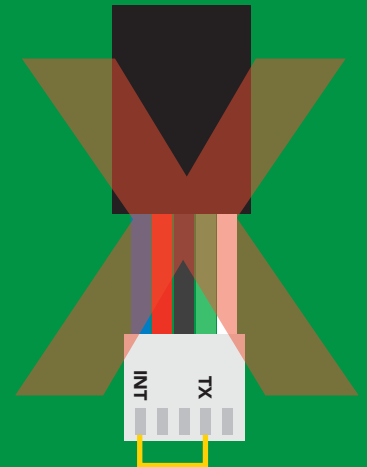
`*OK <cr>`

`I2C,100`



cannot change to I²C

`*ER <cr>`



cannot change to I²C

Factory reset

Command syntax

Factory <cr> enable factory reset

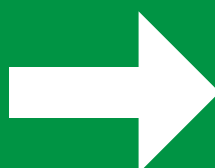
Example

Factory <cr>

Response

*OK <cr>

Factory <cr>



(reboot)



*OK <cr>

*RS <cr>
*RE <cr>

Baud rate will not change

Change to I²C mode

Command syntax

Default I²C address 111 (0x6F)

I2C,n <cr> sets I²C address and reboots into I²C mode

n = any number 1 – 127

Example

Response

I2C,100 <cr>

*OK (reboot in I²C mode)

Wrong example

Response

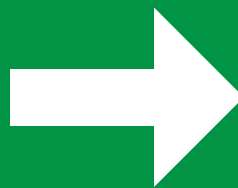
I2C,139 <cr> n ≠ 127

*ER <cr>

I2C,100



Green
*OK <cr>



(reboot)



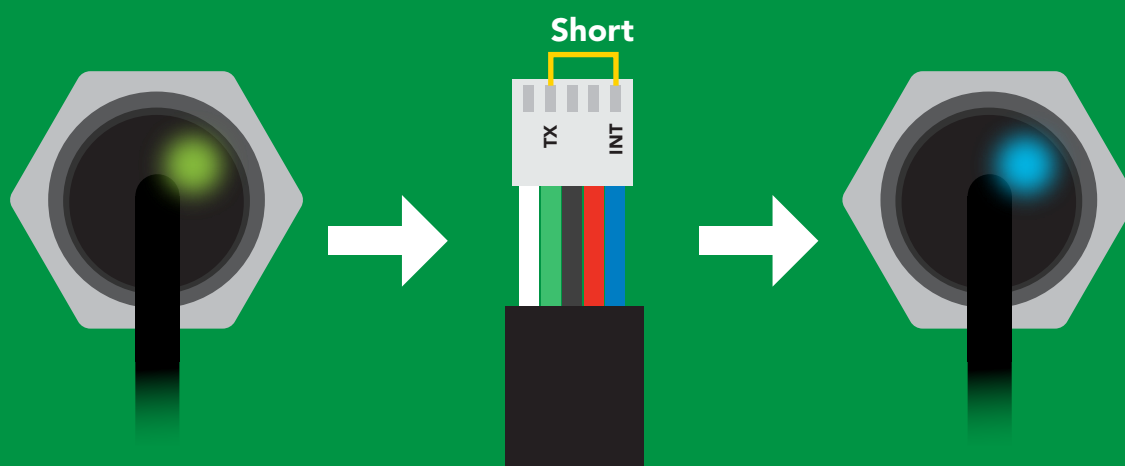
Blue
now in I²C mode

Manual switching to I²C

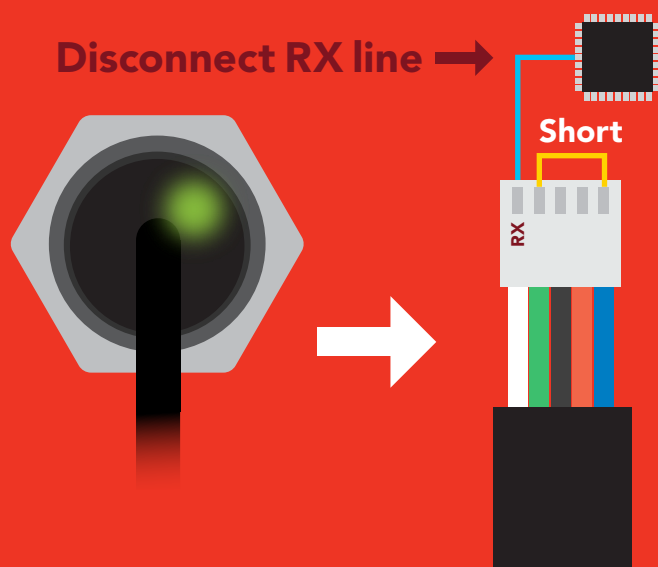
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to INT
- Confirm RX is disconnected
- Connect ground (power on)
- Wait for LED to change from **Green** to **Blue**
- Disconnect ground (power off)
- Reconnect all data and power

Manually switching to I²C will set the I²C address to 111 (0x6F)

Example



Wrong Example



I²C mode

The I²C 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 I²C mode click [here](#)

Settings that are retained if power is cut

- Calibration
- Change I²C address
- Hardware switch to UART mode
- LED control
- Protocol lock
- Software switch to UART mode

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

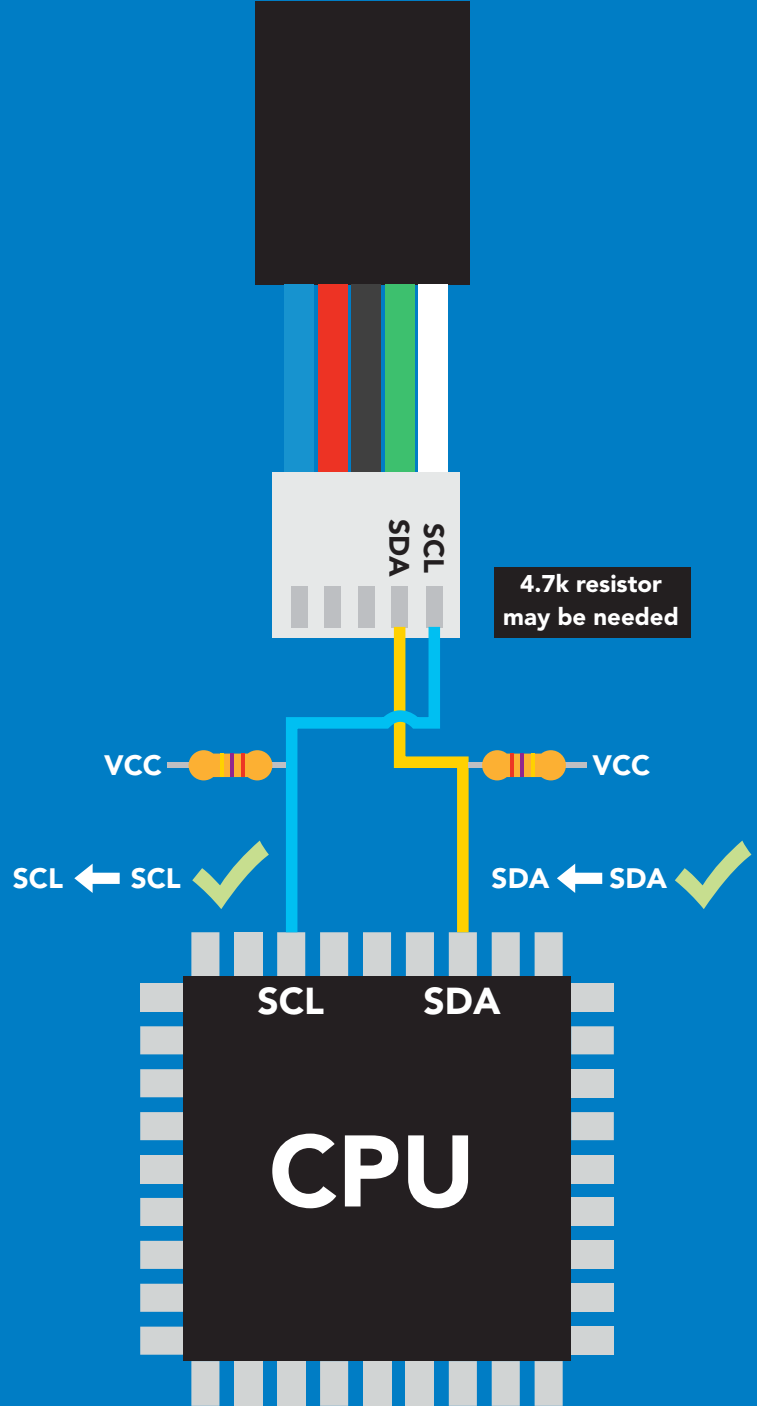
- Sleep mode

I²C mode

I²C address (0x01 – 0x7F)
111 (0x6F) default

Vcc 3.3V – 5.5V

Clock speed 100 – 400 kHz



Data format

Reading Humidity
 Air Temperature
 Dew point

Units % Relative humidity
 Air Temperature °C (when enabled)
 Dew point Temperature °C (when enabled)

Encoding ASCII (CSV string if temp/
 dew point enabled)

Data type floating point

Decimal places 2

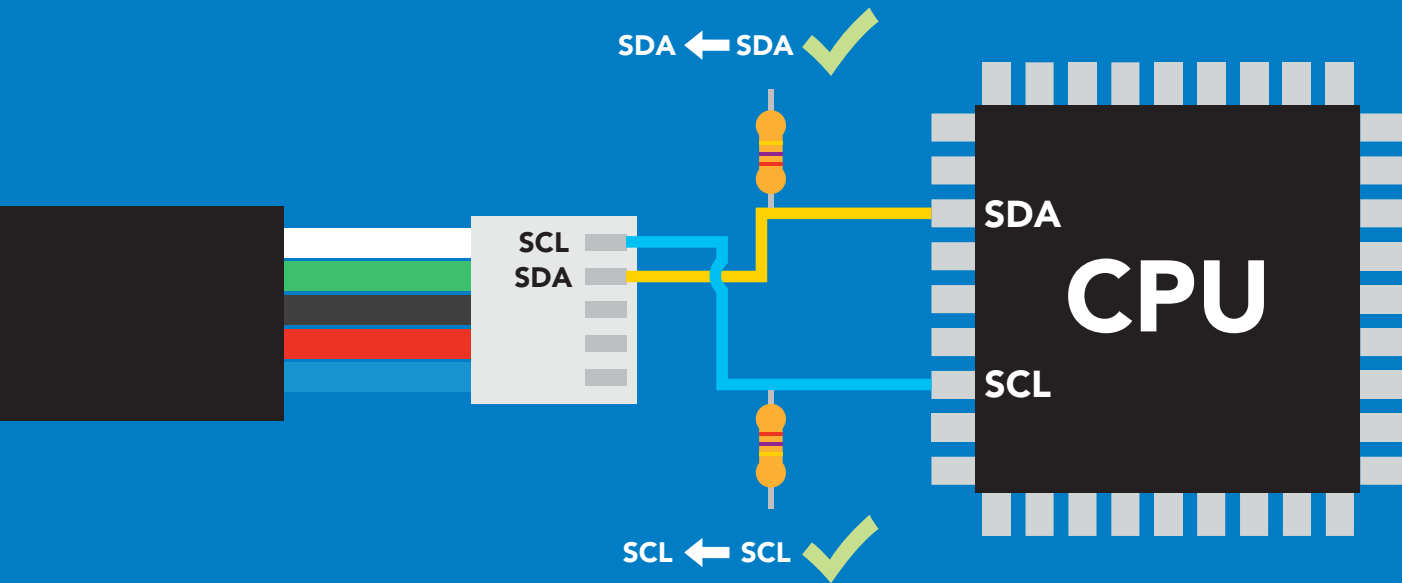
Smallest string 4 characters

Largest string 24 characters

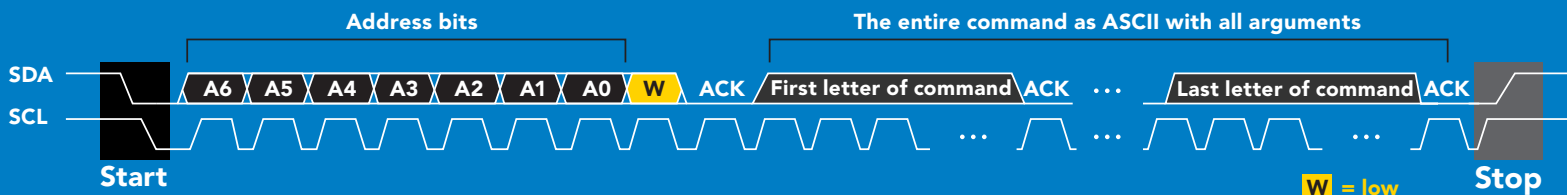
Sending commands to device



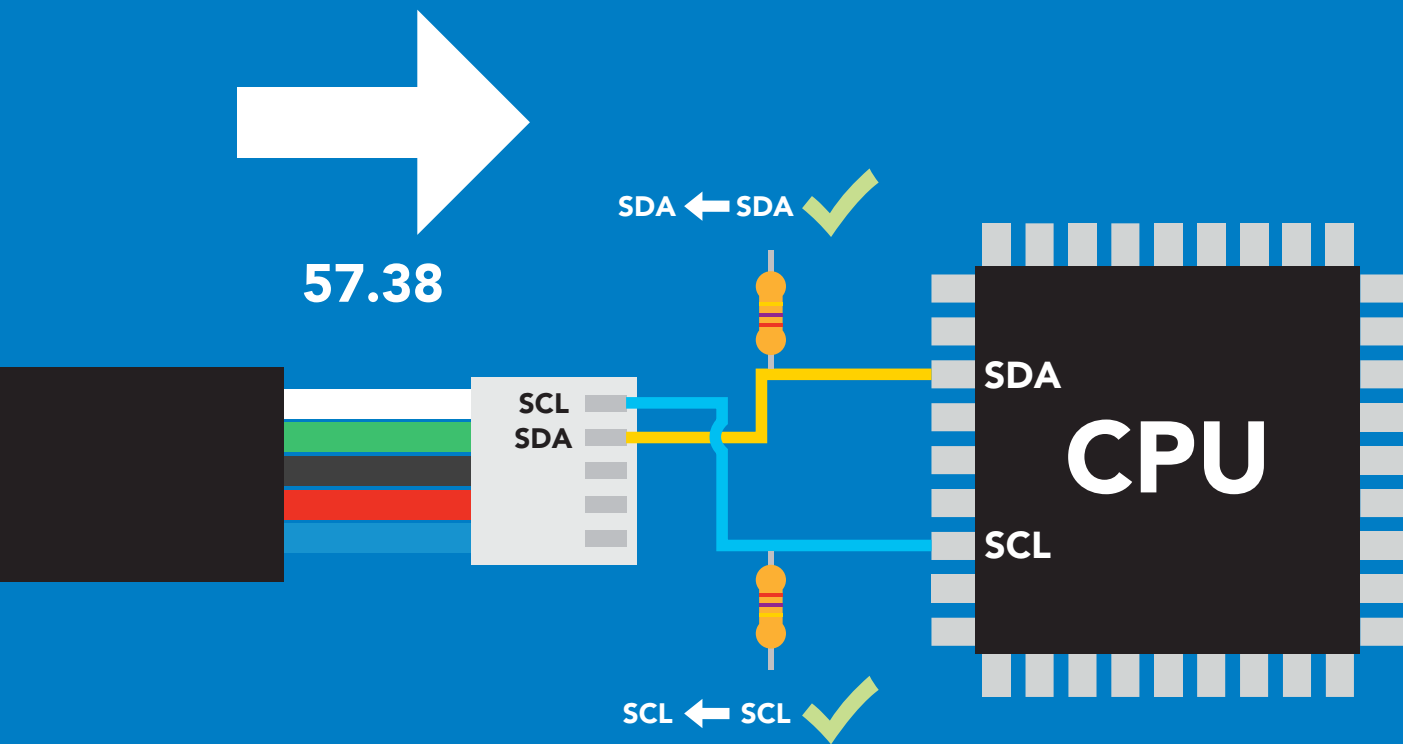
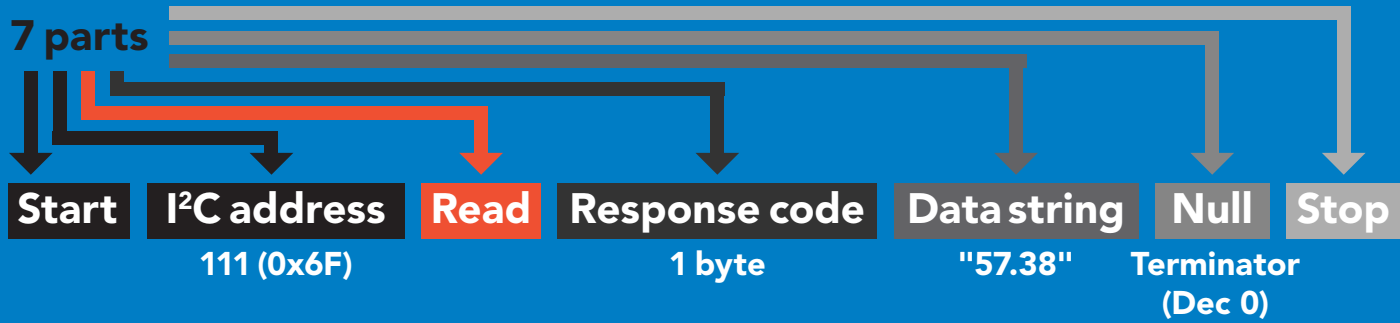
Example



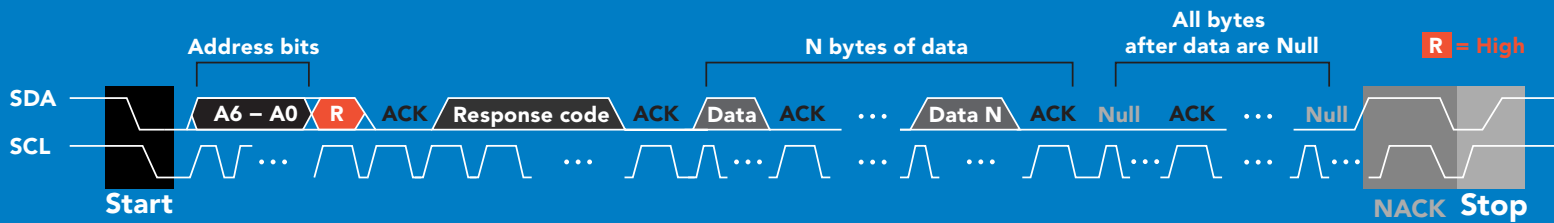
Advanced



Requesting data from device



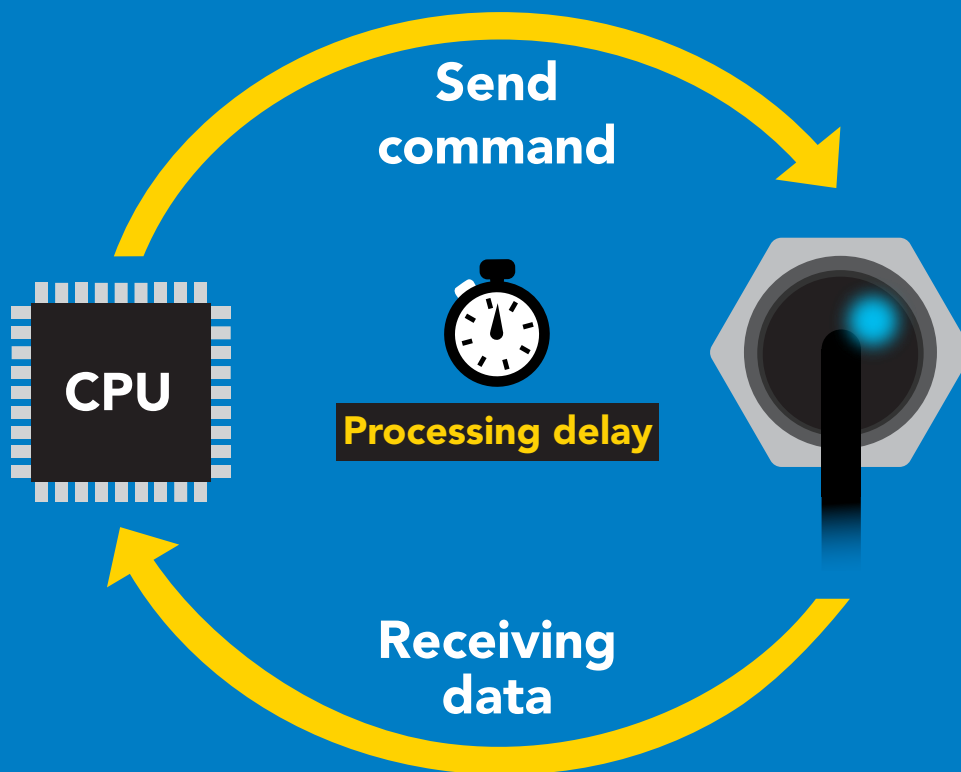
Advanced



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;
```

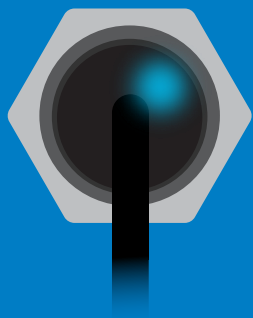
If there is no processing delay or the processing delay is too short, the response code will always be 254.

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



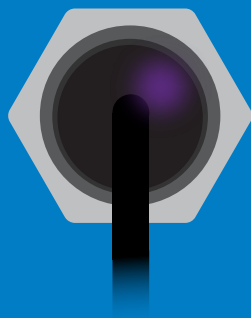
Blue

I²C standby



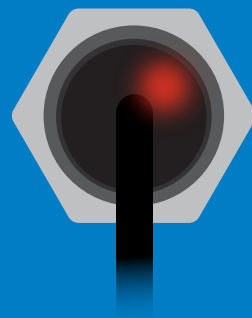
Green

Taking reading



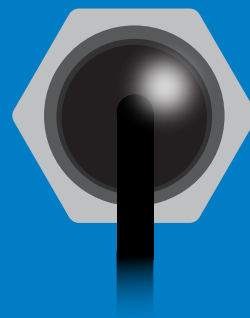
Purple

Changing
I²C address



Red

Command
not understood



White

Find

5V

+0.2 mA

3.3V

+0.2 mA

I²C mode

command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function	
Auto	enable/disable auto monitor	pg. 40
Baud	switch back to UART mode	pg. 49
Factory	enable factory reset	pg. 48
Find	finds device with blinking white LED	pg. 38
i	device information	pg. 43
I2C	change I ² C address	pg. 47
L	enable/disable LED	pg. 37
Name	set/show name of device	pg. 42
O	enable/disable parameters	pg. 41
Plock	enable/disable protocol lock	pg. 46
R	returns a single reading	pg. 39
Sleep	enter sleep mode/low power	pg. 45
Status	retrieve status information	pg. 44

LED control

Command syntax

300ms  processing delay

L,1 LED on **default**

L,0 LED off

L,? LED state on/off?

Example

Response

L,1

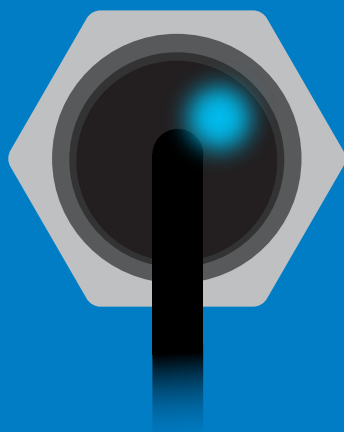

Wait 300ms **1** **0**
Dec Null

L,0

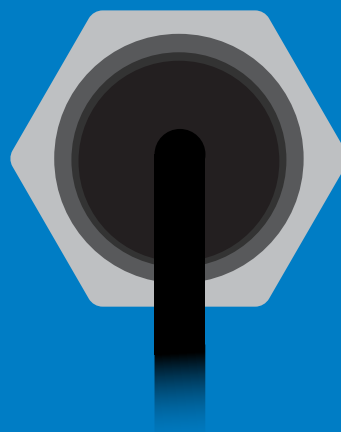

Wait 300ms **1** **0**
Dec Null

L,?


Wait 300ms **1** **?L,1** **0** or 
Dec ASCII Null Wait 300ms **1** **?L,0** **0**
Dec ASCII Null



L,1



L,0

Find

Command syntax

300ms  processing delay

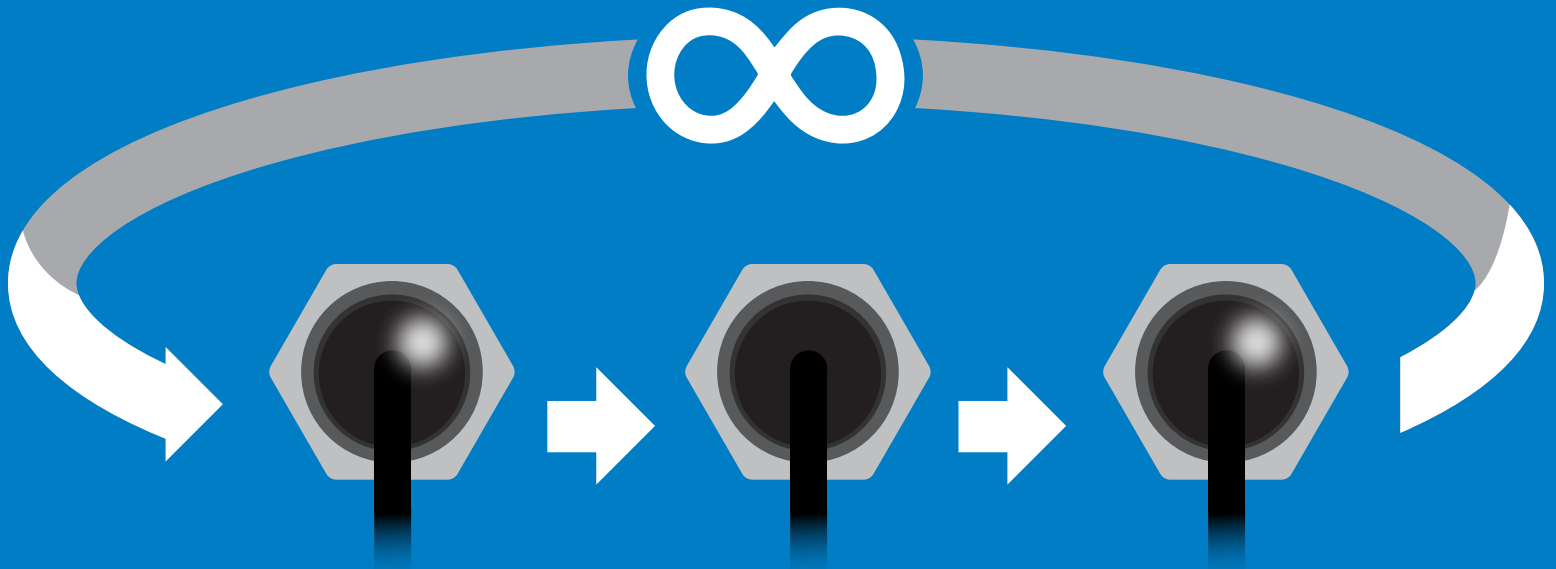
Find LED rapidly blinks white, used to help find device

Example

Response

Find

 Wait 300ms **1** Dec **0** Null



Taking reading

Command syntax

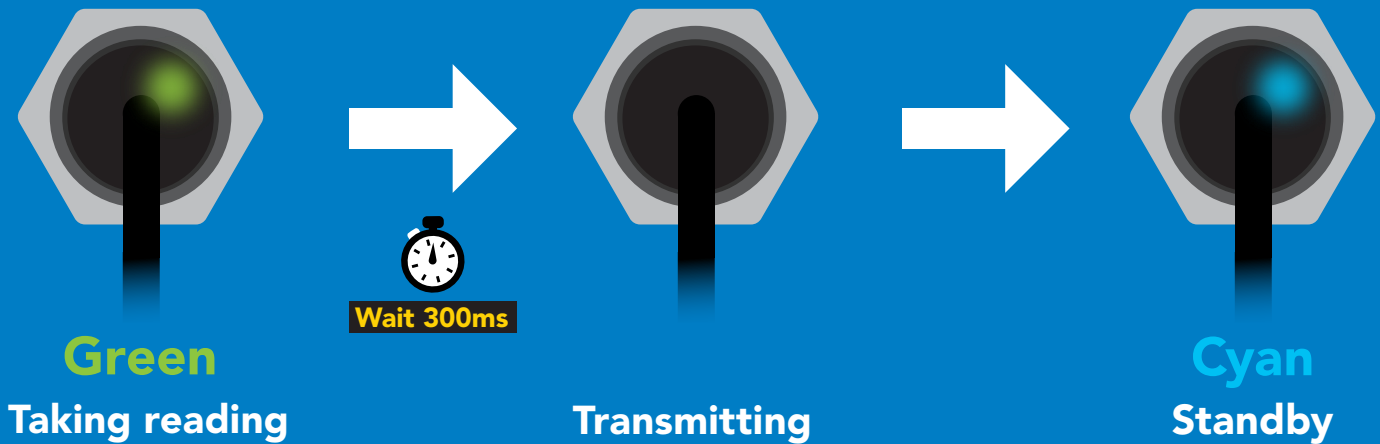
300ms  processing delay

R return 1 reading

Example

Response

R  **1** **57.38** **0**
Wait 300ms Dec ASCII Null



Auto monitor

300ms  processing delay

Command syntax

When enabled, the sensor will continuously monitor the readings and set the auto monitor pin high when your value has been reached. When Auto Monitor is enabled, it is not necessary to actively take readings (continuous mode can be disabled).

Auto,en, [0,1,2]

0 = disable, 1= Enable for humidity, 2= Enable for dew point

Auto,n

The value that will set the alarm pin

Auto,tol,n

The value that will reset the alarm pin

Auto,?

Auto monitor settings

Example

Response

Auto,en,1

 **1** **0** Enable humidity automonitoring
Wait 300ms Dec Null


Auto,57.38

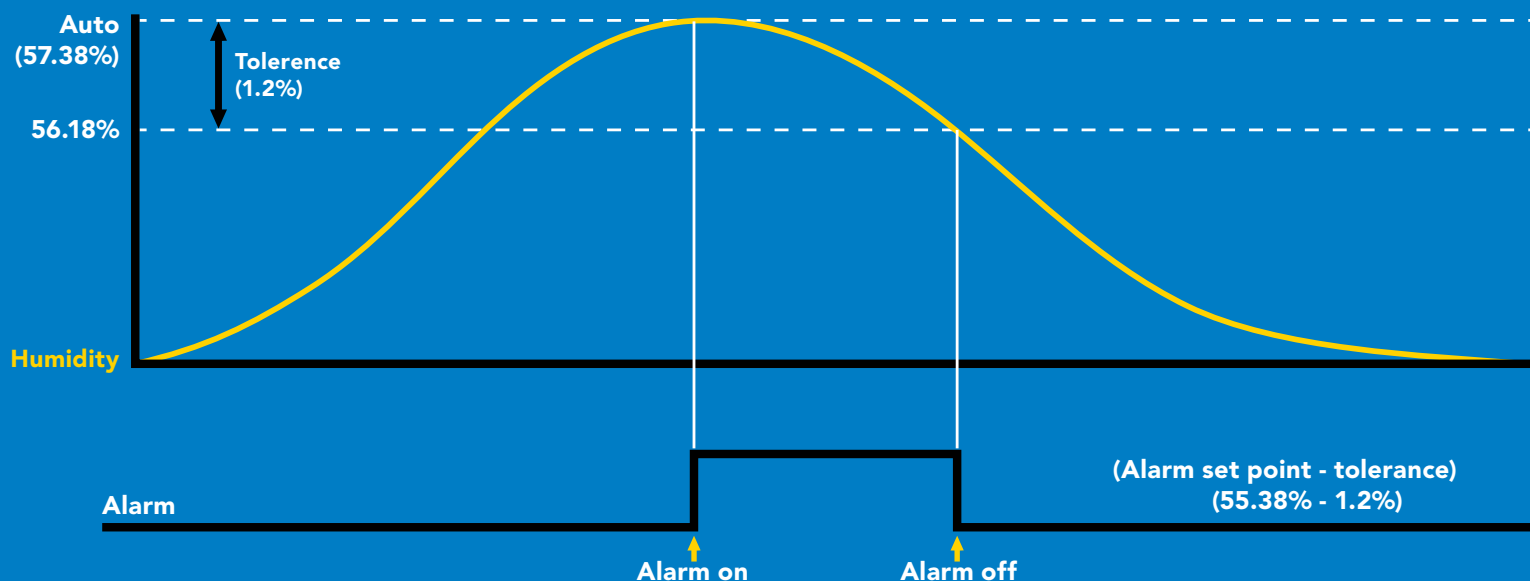
 **1** **0** Set alarm to go off at 55.38% humidity
Wait 300ms Dec Null

Auto,tol,1.2

 **1** **0** The humidity must fall 1.2 percentage points below set point for alarm to reset.
Wait 300ms Dec Null

Auto,?

 **1** **?,auto,57.38,1.20,1** **0** if all are enabled
Wait 300ms Dec ASCII Null



Enable/disable parameters from output string

Command syntax

O, [parameter],[1,0] enable or disable output parameter
O,? enabled parameter?

Example

O,HUM,1 / O,HUM,0

Response

 **1** **0** enable / disable humidity
Wait 300ms Dec Null

O,T,1 / O,T,0

 **1** **0** enable / disable temperature
Wait 300ms Dec Null

O,Dew,1 / O,Dew,0

 **1** **0** enable / disable dew point
Wait 300ms Dec Null

O,?

 **1** **? , O, HUM, T, Dew** **0** if all enabled
Wait 300ms Dec ASCII Null

Parameters

Hum Humidity
T Air temperature in °C
Dew Dew point

Followed by 1 or 0

1 enabled
0 disabled

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

Naming device

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	16	
Name,	clears name																		
Name,?	show name																		

Up to 16 ASCII characters

Example

Response

Name,



1 0
Dec Null

name has been cleared

Name,zzt



1 0
Dec Null

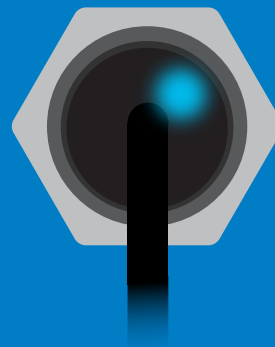
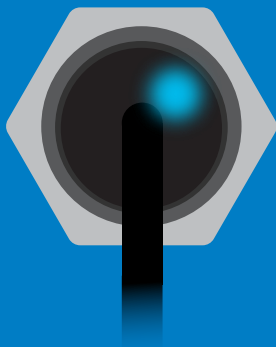
Name,?



1 ?Name,zzt 0
Dec ASCII 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,HUM,1.0

ASCII

0

Null

Response breakdown

?i, HUM, 1.0
↑ ↑
Device Firmware

Reading device status

Command syntax

300ms  processing delay

Status voltage at Vcc pin and reason for last restart

Example

Response

Status

 **1** **?Status,P,5.038** **0**
Wait 300ms Dec ASCII 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

5V

MAX
2.6 mA

SLEEP
0.5 mA

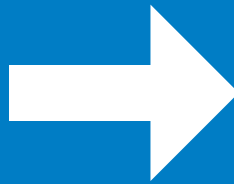
3.3V

2.2 mA

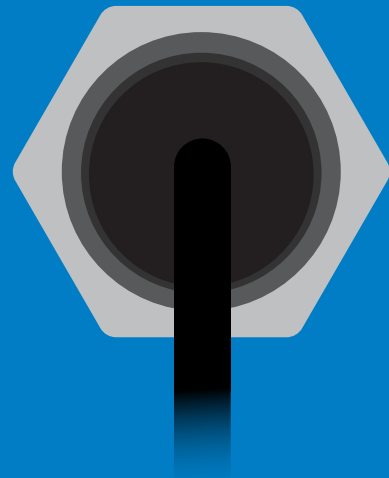
0.4 mA



Standby



Sleep



Sleep

Protocol lock

Command syntax

300ms  processing delay

Plock,1 enable Plock

Plock,0 disable Plock **default**

Plock,? Plock on/off?

Locks device to I²C mode.

Example

Response


Plock,1


Wait 300ms **1** **0**
Dec Null

Plock,0

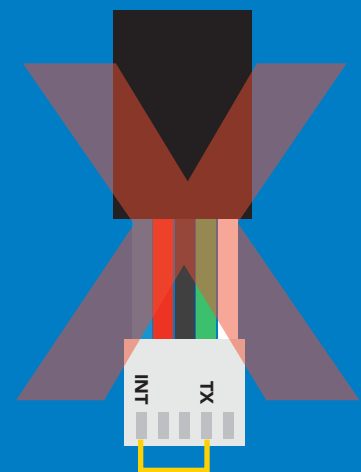
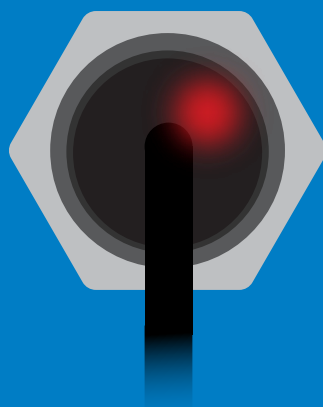
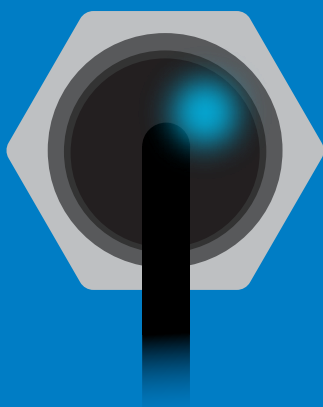

Wait 300ms **1** **0**
Dec Null

Plock,?


Wait 300ms **1** **?Plock,1** **0**
Dec ASCII Null

Plock,1

Baud, 9600



cannot change to UART

cannot change to UART

I²C address change

Command syntax

I2C,n sets I²C address and reboots into I²C mode

Example

I2C,101

Response

device reboot
(no response given)

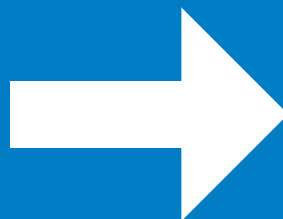
Warning!

Changing the I²C address will prevent communication between the circuit and the CPU until the CPU is updated with the new I²C address.

Default I²C address is 111 (0x6F).

n = any number 1 – 127

I2C,101



(reboot)



Factory reset

Command syntax

Factory reset will not take the device out of I²C mode.

Factory enable factory reset

I²C address will not change

Example

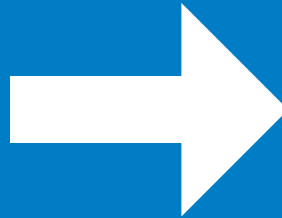
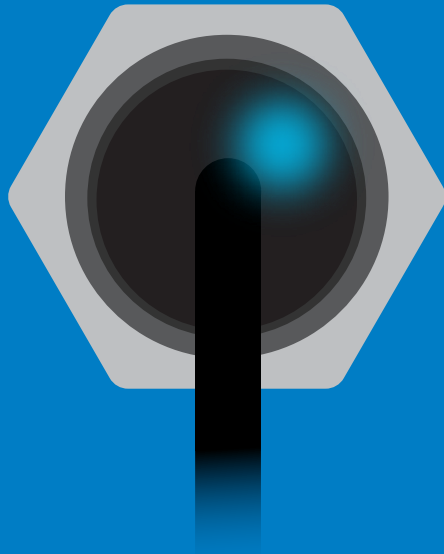
Response

Factory

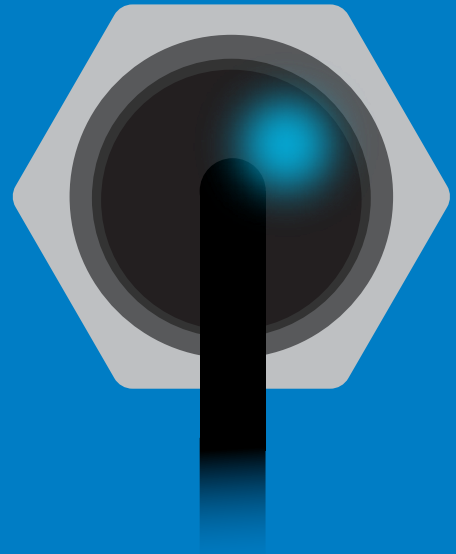
device reboot
(no response given)

Clears custom calibration
LED on
Response codes enabled

Factory



(reboot)



Change to UART mode

Command syntax

Baud,n switch from I²C to UART

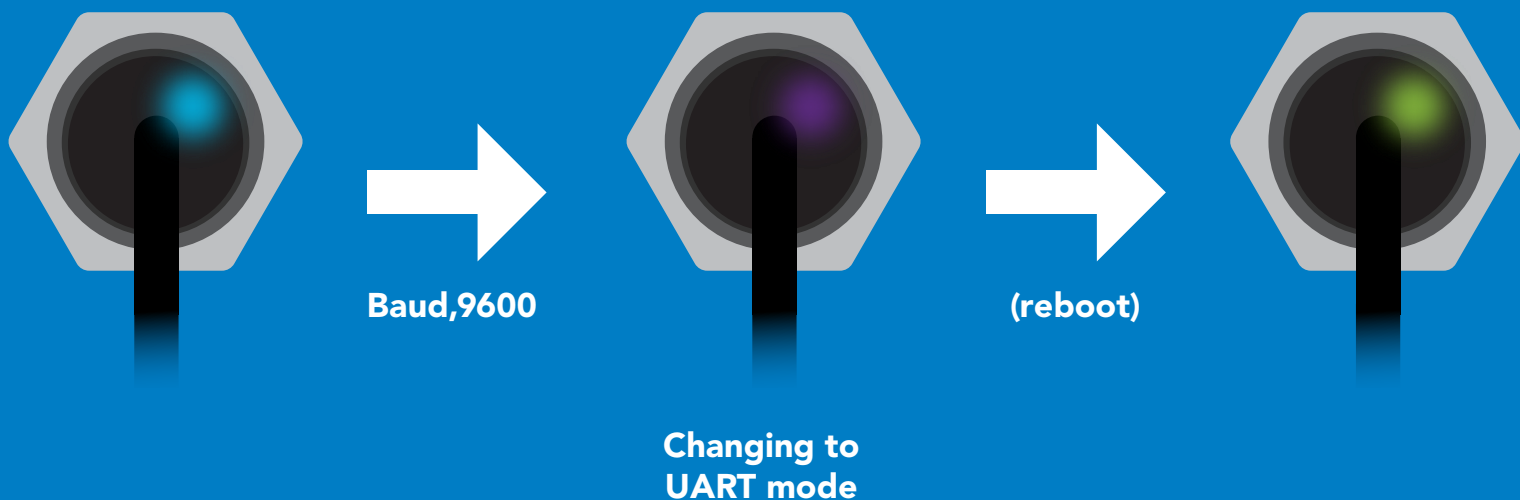
Example

Baud,9600

Response

reboot in UART mode
(no response given)

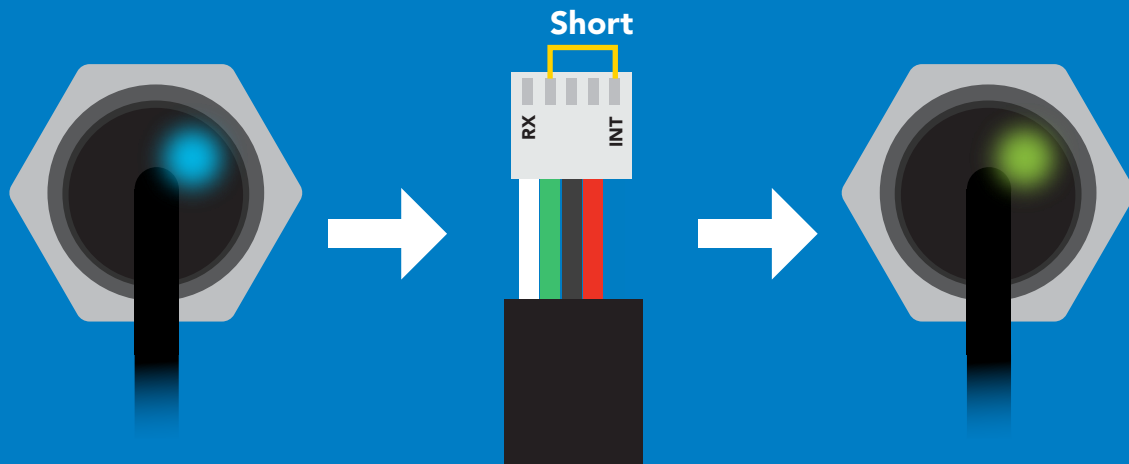
n = [300
1200
2400
9600
19200
38400
57600
115200



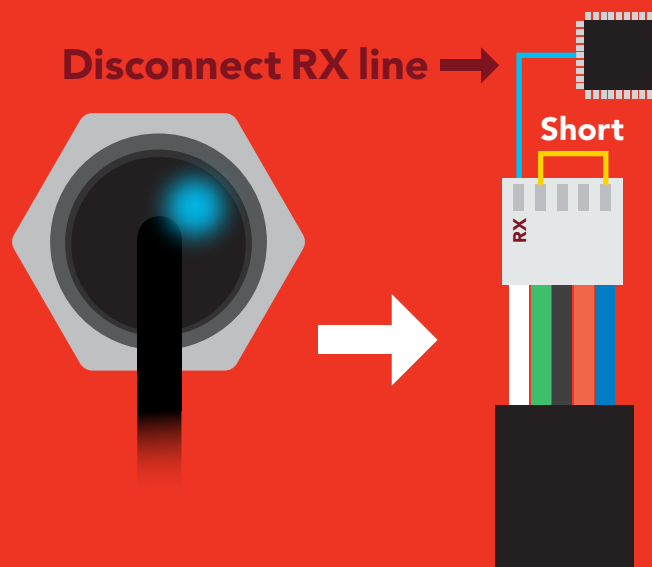
Manual switching to UART

- 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 1.3

Added Air Temperature chart on pg 5.

Datasheet V 1.2

Revised naming device info on pages 20 & 42.

Datasheet V 1.1

Revised the information on pg 3.

Datasheet V 1.0

New datasheet

Firmware updates

V1.0 – Initial release (August 14, 2020)

Warranty

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

The debugging phase

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

- **Soldering any part to the EZO-HUM™ Embedded Humidity Sensor.**
- **Running any code, that does not exclusively drive the EZO-HUM™ Embedded Color Sensor and output its data in a serial string.**
- **Embedding the EZO-HUM™ Embedded Humidity 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-HUM™ Embedded Humidity Sensor, against the thousands of possible variables that may cause the EZO-HUM™ Embedded Humidity 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-HUM™ Embedded Humidity Sensor continued operation. This is because that would be equivalent to Atlas Scientific™ taking responsibility over the correct operation of your entire device.