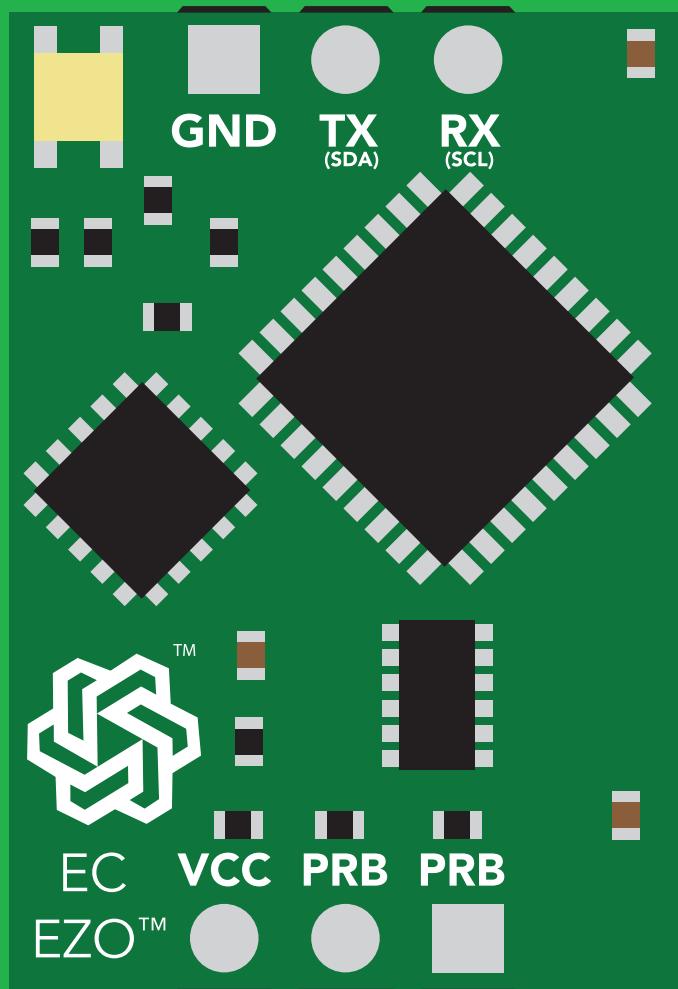


EZO-EC™

Embedded Conductivity Circuit

Reads	Conductivity = $\mu\text{S}/\text{cm}$ Total dissolved solids = ppm Salinity = PSU Specific gravity (sea water only) = 1.00 – 1.300
Range	0.07 – 500,000+ $\mu\text{S}/\text{cm}$
Accuracy	+/- 2%
Response time	1 reading per sec
Supported probes	K 0.1 – K 10 any brand
Calibration	2 or 3 point
Temp compensation	Yes
Data protocol	UART & I²C
Default I ² C address	100 (0x64)
Operating voltage	3.3V – 5V
Data format	ASCII





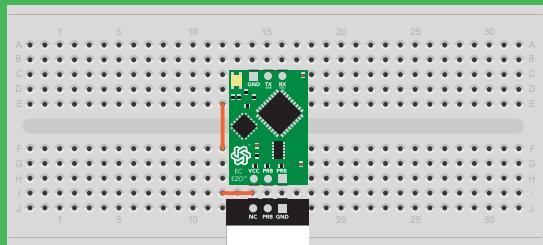
STOP

SOLDERING THIS DEVICE VOIDS YOUR WARRANTY.

This is sensitive electronic equipment. Get this device working in a solderless breadboard first. Once this device has been soldered it is no longer covered by our warranty.

This device has been designed to be soldered and can be soldered at any time. Once that decision has been made, Atlas Scientific no longer assumes responsibility for the device's continued operation. The embedded systems engineer is now the responsible party.

Get this device working in a solderless breadboard first!



Do not embed this device without testing it in a solderless breadboard!

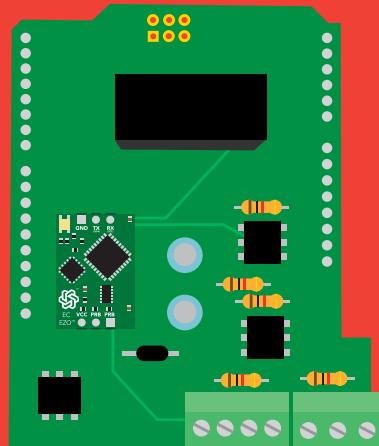


Table of contents

Circuit dimensions	4	Operating principle	8
Power consumption	4	Output units	9
Absolute max ratings	4	Calibration theory	10
EZO™ circuit identification	5	Power and data isolation	12
Conductivity probe range	6	Correct wiring	14
Resolution	7	Available data protocols	15

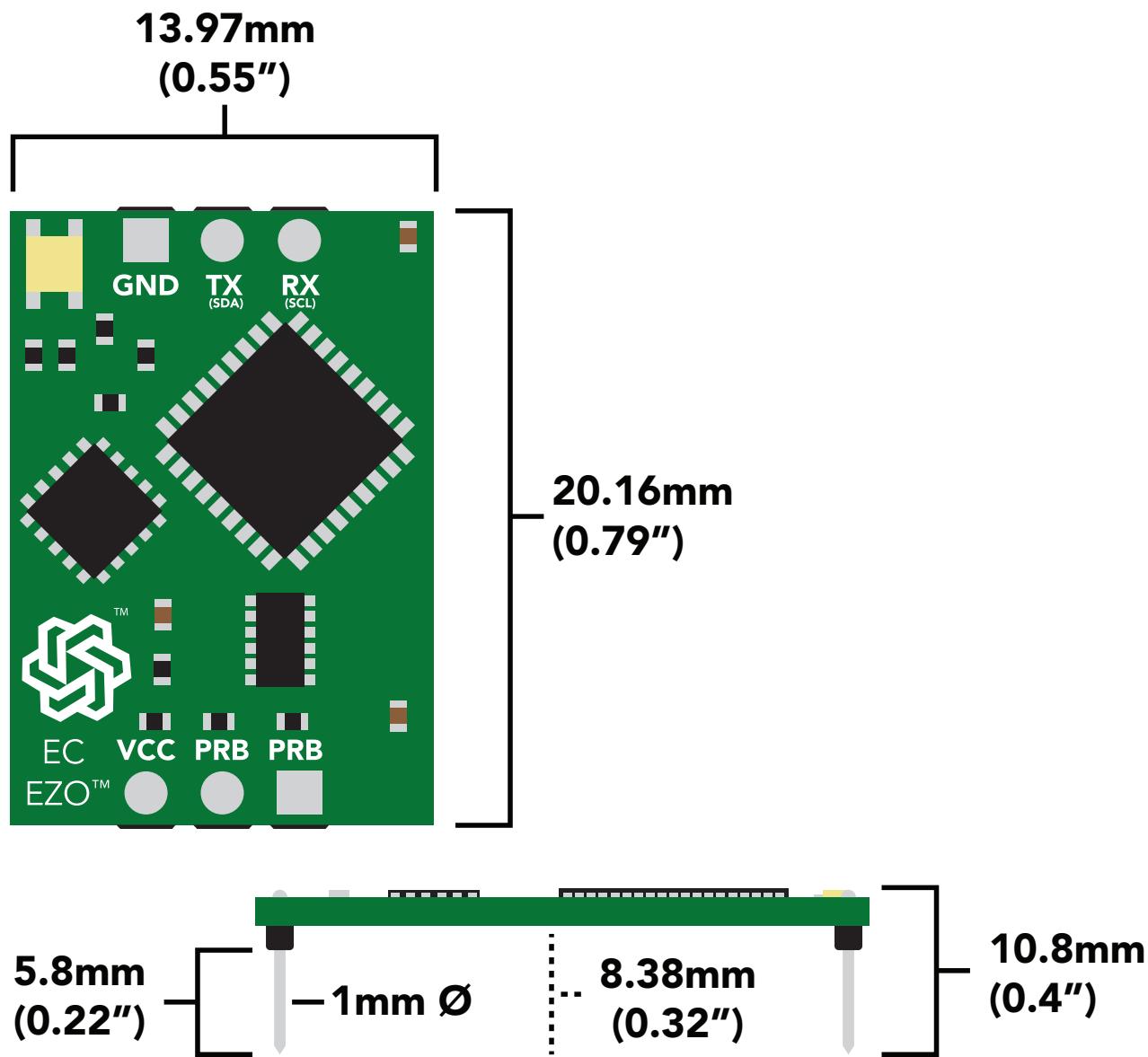
UART

UART mode	17
Default state	18
Receiving data from device	19
Sending commands to device	20
LED color definition	21
UART quick command page	22
LED control	23
Find	24
Continuous reading mode	25
Single reading mode	26
Calibration	27
Export/import calibration	28
Setting the probe type	29
Temperature compensation	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 ² C mode	40
Manual switching to I ² C	41

I²C

I²C mode	43
Sending commands	44
Requesting data	45
Response codes	46
LED color definition	47
I²C quick command page	48
LED control	49
Find	50
Taking reading	51
Calibration	52
Export/import calibration	53
Setting the probe type	54
Temperature compensation	55
Enable/disable parameters	56
Device information	57
Reading device status	58
Sleep mode/low power	59
Protocol lock	60
I ² C address change	61
Factory reset	62
Change to UART mode	63
Manual switching to UART	64
Circuit footprint	65
Datasheet change log	66
Warranty	69

EZO™ circuit dimensions



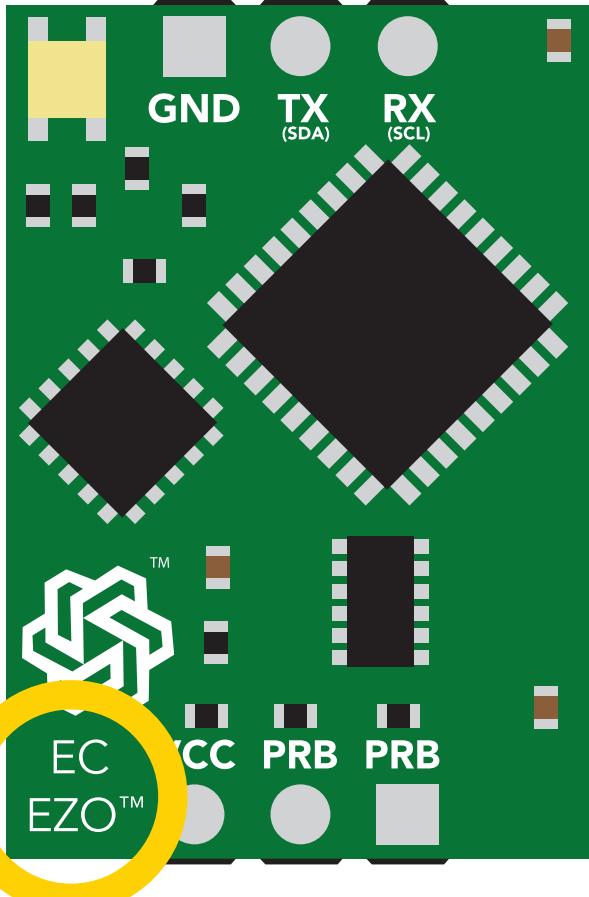
Power consumption

Absolute max ratings

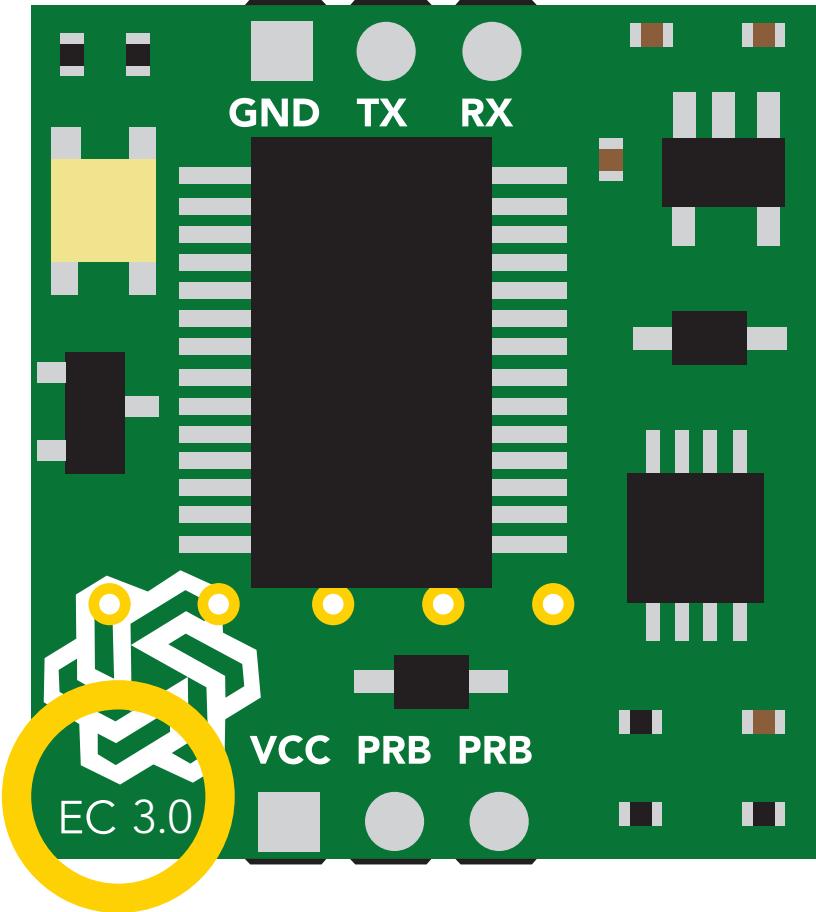
	LED	MAX	STANDBY	SLEEP
5V	ON	50 mA	18.14 mA	0.7 mA
	OFF	45 mA	15.64 mA	
3.3V	ON	35 mA	16.85 mA	0.4 mA
	OFF	34 mA	15.85 mA	

Parameter	MIN	TYP	MAX
Storage temperature (EZO™ Conductivity)	-60 °C		150 °C
Operational temperature (EZO™ Conductivity)	-40 °C	25 °C	125 °C
VCC	3.3V	5V	5.5V

EZO™ circuit identification



EZO™ Conductivity circuit



Legacy Conductivity circuit



Viewing correct datasheet



Viewing incorrect datasheet

[Click here to view legacy datasheet](#)

Conductivity probe range

The EZO™ Conductivity circuit is capable of connecting to any two-conductor conductivity probe, ranging from:

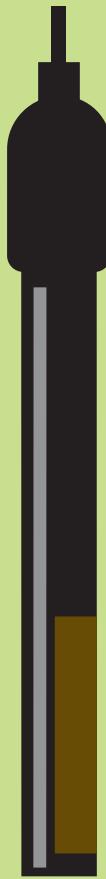
K 0.1



K 10

Atlas Scientific™ has tested 3 different K value probe types

K 0.1



K 1.0



K 10



accurate reading range

0.07µS – 50,000µS

accurate reading range

5µS – 200,000+µS

accurate reading range

10µS – 1S

Atlas Scientific™ does not know what the accurate reading range would be for conductivity probes, other than the above mentioned values. Determining the accurate reading range of such probes, i.e. **K 2.6**, or **K 0.66**, is the responsibility of the embedded systems engineer.

Resolution

The EZO™ Conductivity circuit, employs a method of scaling resolution. As the conductivity increases the resolution between readings decreases.

The EZO™ Conductivity circuit will output conductivity readings where the first **4 digits** are valid and the others are set to 0. This excludes conductivity readings that are less than 9.99. In that case, only 3 conductivity digits will be output.

0.07 – 99.99

Resolution = **0.01 μ S**

100.1 – 999.9

Resolution = **0.1 μ S**

1,000 – 9,999

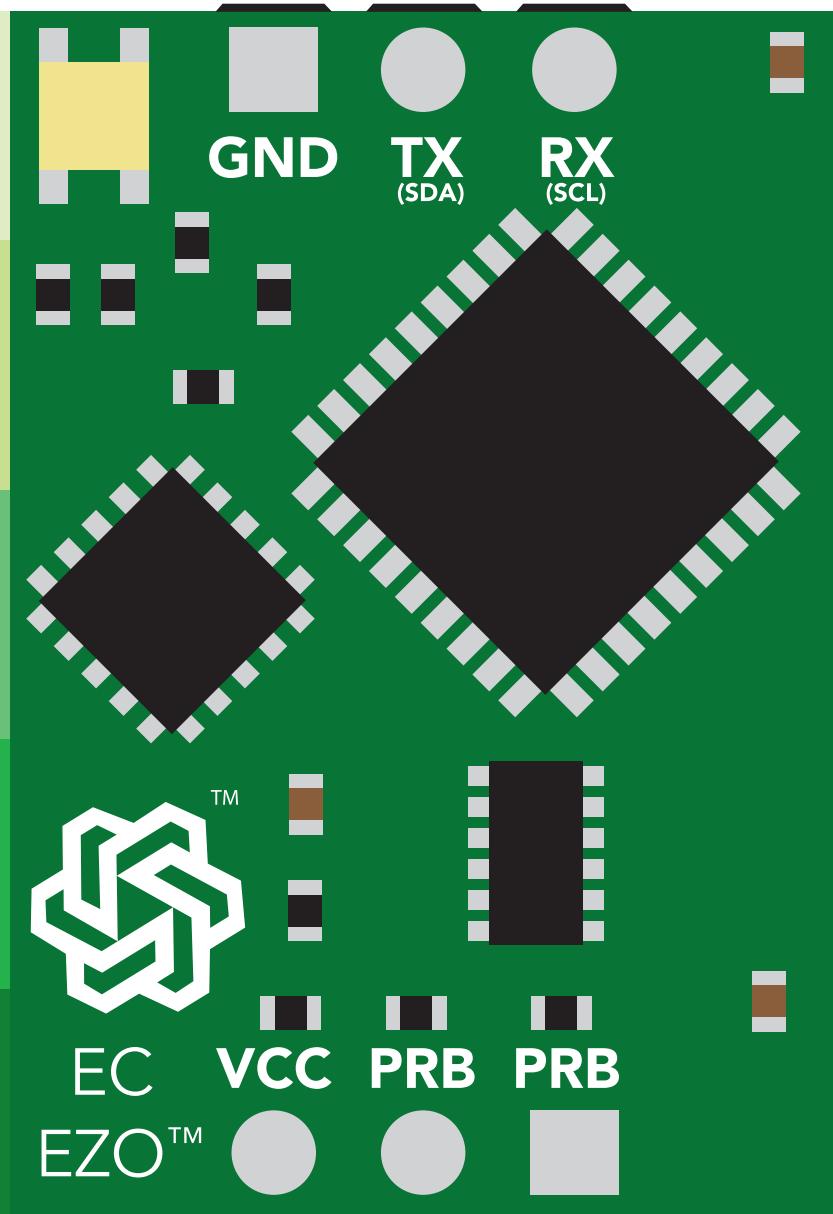
Resolution = **1.0 μ S**

10,000 – 99,990

Resolution = **10 μ S**

100,000 – 999,900

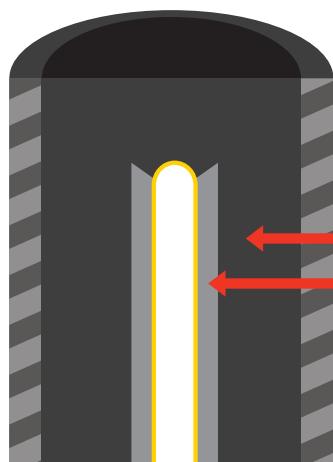
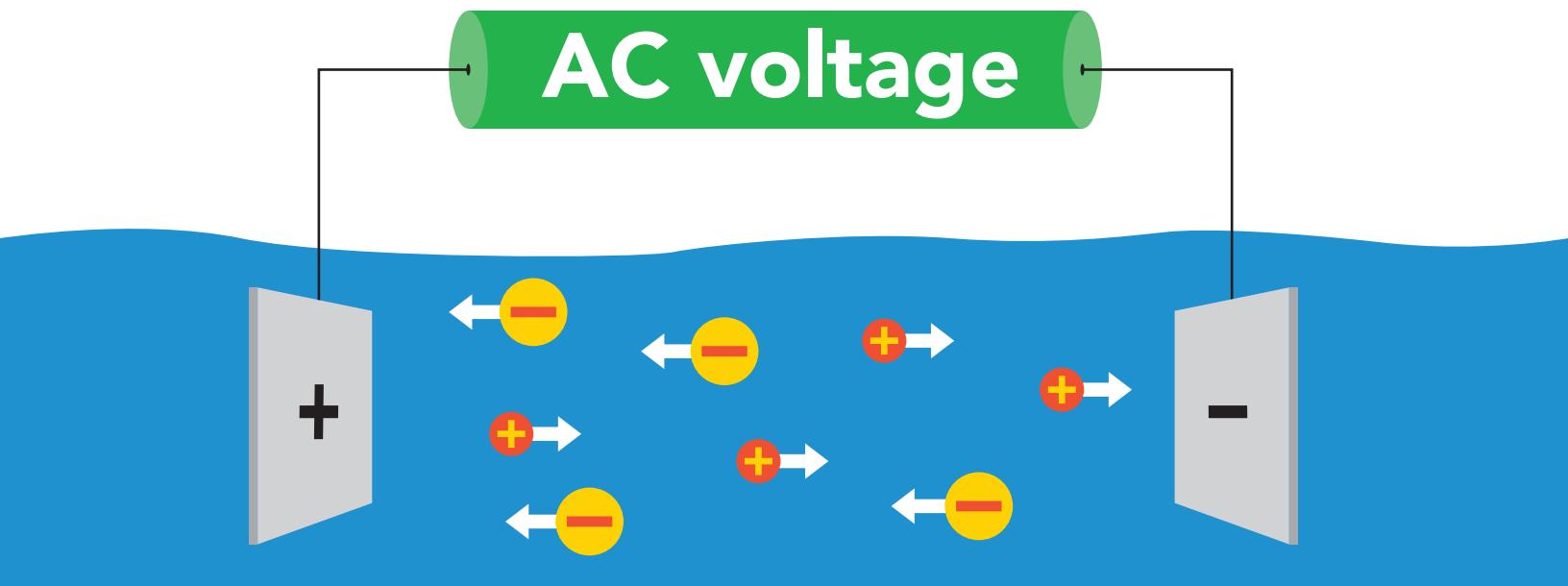
Resolution = **100 μ S**



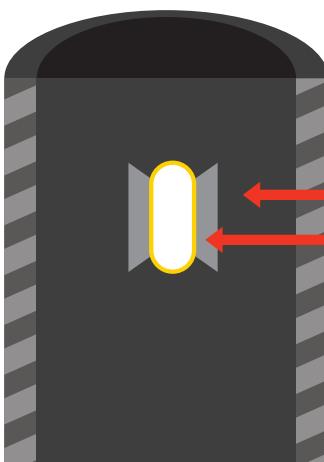
Operating principle

An E.C. (**electrical conductivity**) probe measures the electrical conductivity in a solution. It is commonly used in hydroponics, aquaculture and freshwater systems to monitor the amount of nutrients, salts or impurities in the water.

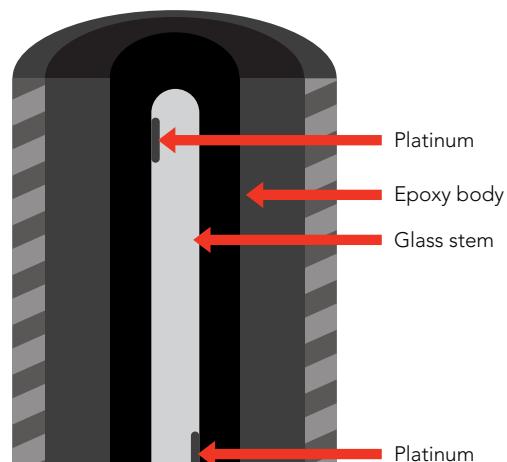
Inside the conductivity probe, two electrodes are positioned opposite from each other, an AC voltage is applied to the electrodes causing cations to move to the negatively charged electrode, while the anions move to the positively electrode. The more free electrolyte the liquid contains, the higher the electrical conductivity.



K 0.1
Graphite electrode



K 1.0
Graphite electrode



K 10
Platinum electrode

Output units

By default, EZO™ Conductivity circuits with firmware version 2.10 and above will **only output EC**. To enable these parameters see page 31 for UART, and 56 for I²C.

The EZO™ Conductivity circuit also has the capability to read:

Conductivity = $\mu\text{S}/\text{cm}$

Total dissolved solids = ppm

Salinity = PSU

Specific gravity (sea water only) = 1.00 – 1.300

These parameters must be individually enabled within the device. See page 31 to enable each parameter in UART mode, and on page 56 for I²C mode.

Once these parameters have been enabled, output will be a CSV string.

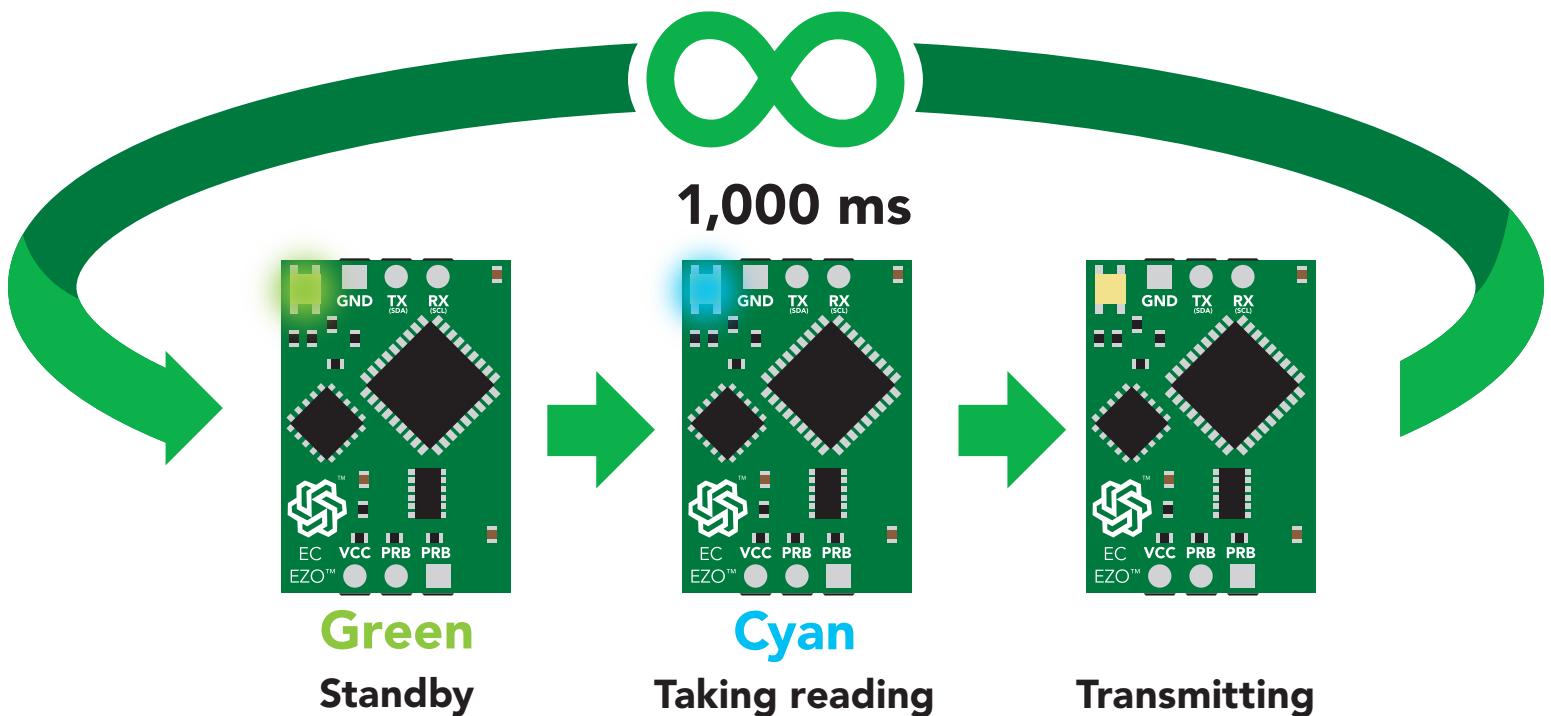
Example

EC,TDS,SAL,SG

Default LED blink pattern

This is the LED pattern for Continuous Mode (default state)

This can only happen when the device is in **UART** mode.

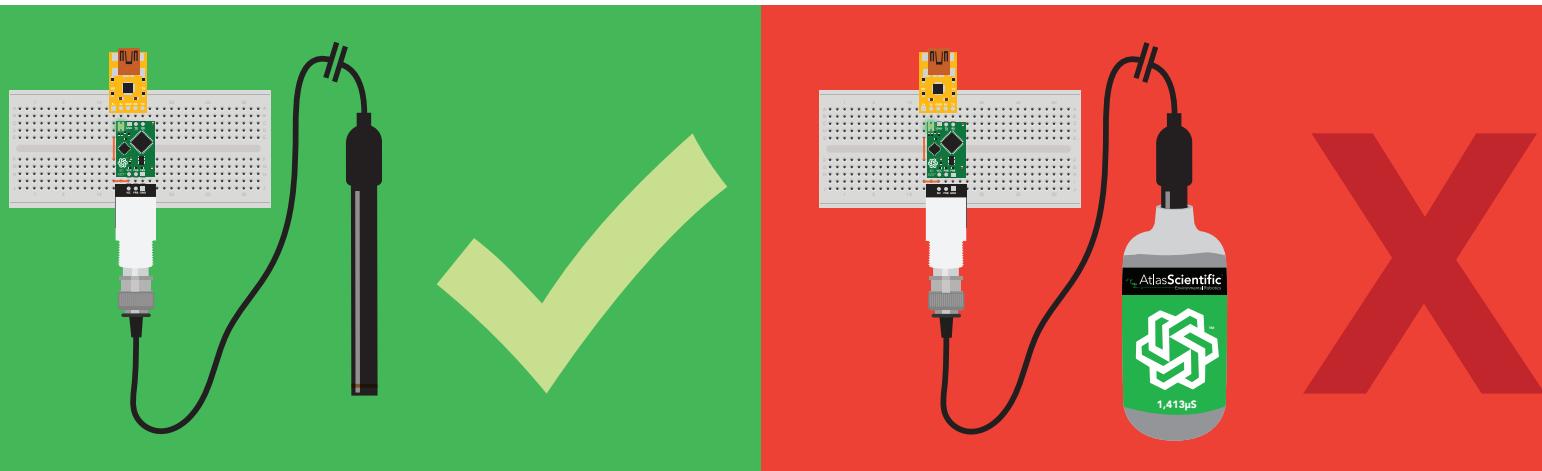


Calibration theory

The most important part of calibration is watching the readings during the calibration process. It's easiest to calibrate the device in its default state (UART mode, continuous readings). Switching the device to I²C mode after calibration **will not** affect the stored calibration. If the device must be calibrated in I²C mode be sure to request readings continuously so you can see the output from the probe.

Pre-calibration setup

First, take readings from dry conductivity probe.



Set probe type

If you are not using a K 1.0 conductivity probe (*default*), you need to set the probe type by using the "**K,n**" command. (where n = K value of your probe)

Dry calibration

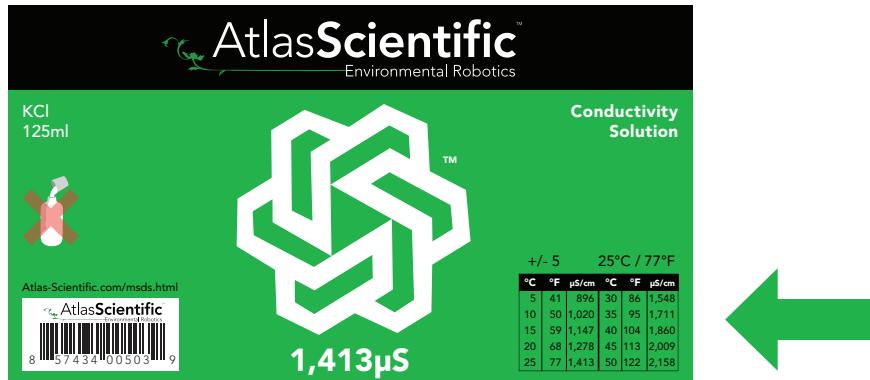
Issuing the "**Cal,dry**" command fine tunes the internal electrical properties of the device. This calibration only needs to be done once. Even though you may see reading of 0.00 before issuing the "**Cal,dry**" command, it is still a necessary component of calibration.

17.00 → "Cal,dry" → 0.00 ✓ Correct

00.00 → "Cal,dry" → 0.00 ✓ Correct

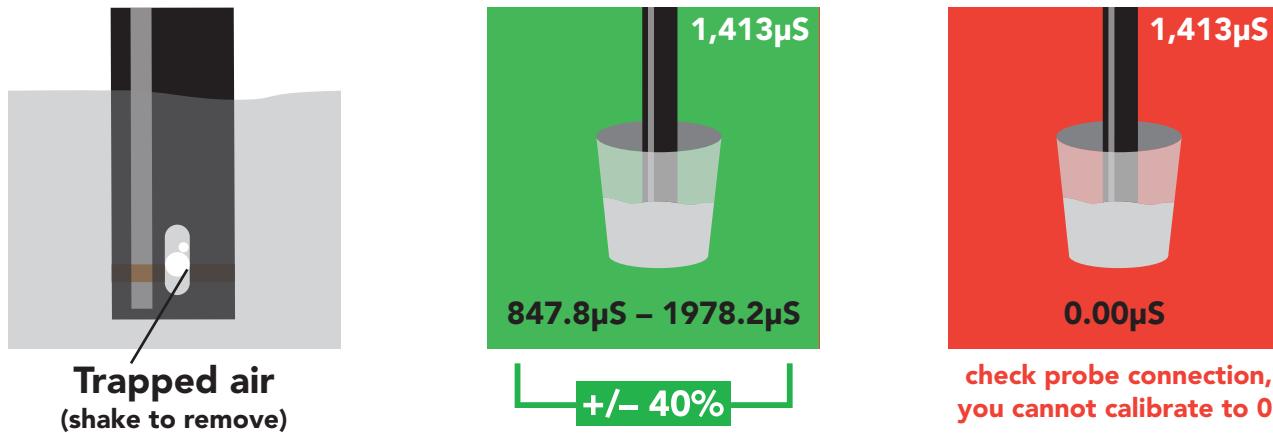
Temperature compensation

Temperature has a significant effect on conductivity readings. The EZO™ Conductivity circuit has its temperature compensation set to 25° C as the default. If the calibration solution is not within 5° of 25° C, check the temperature chart on the side of the calibration bottle, and calibrate to that value.



Low point/single point calibration

Pour a small amount of the calibration solution into a cup. Shake the probe to make sure you do not have trapped air bubbles in the sensing area. You should see readings that are off by **1 – 40%** from the stated value of the calibration solution. Wait for readings to stabilize (small movement from one reading to the next is normal).



Once the readings stabilize, issue the low point or single point calibration command.

Low point calibration: "**Cal,low,1413**" (Readings will **NOT** change)

Single point calibration: "**Cal,1413**" (Readings **will** change, calibration complete).

High point calibration

Shake the probe to remove trapped air and adjust the temperature as done in the previous step. Once the readings have stabilized issue the high point calibration command.

High point calibration: "**Cal,high,12880**" (Readings **will** change, calibration complete).

Power and data isolation

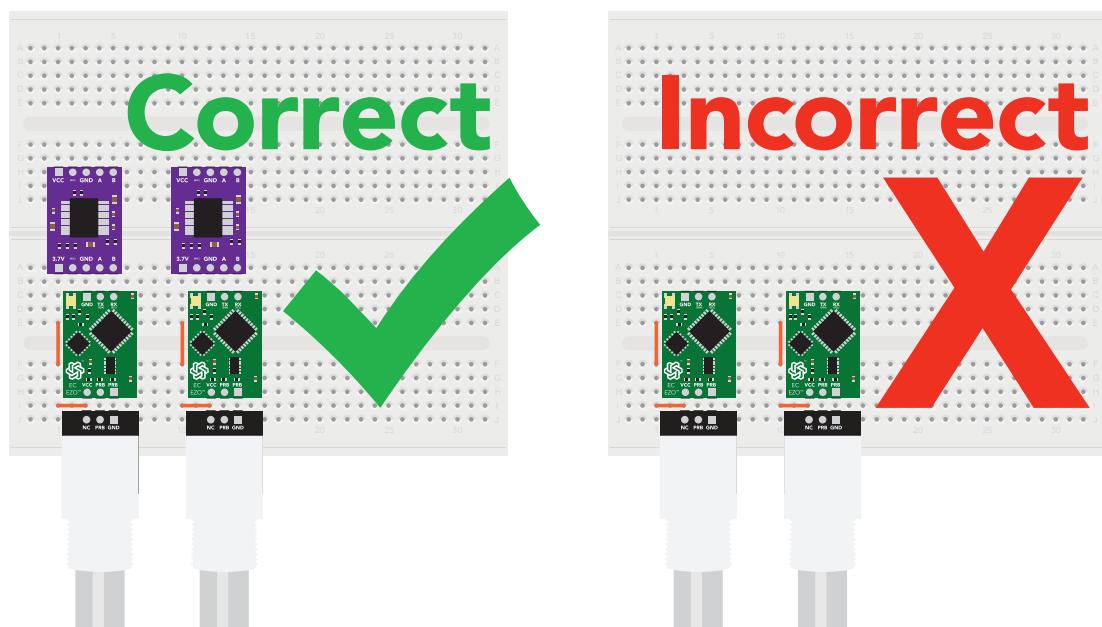
The Atlas Scientific EZO™ Conductivity circuit is a very sensitive device. This sensitivity is what gives the Conductivity circuit its accuracy. This also means that the Conductivity circuit is capable of reading micro-voltages that are bleeding into the water from unnatural sources such as pumps, solenoid valves or other probes/sensors.

When electrical noise is interfering with the Conductivity readings it is common to see rapidly fluctuating readings or readings that are consistently off. To verify that electrical noise is causing inaccurate readings, place the Conductivity probe in a cup of water by itself. The readings should stabilize quickly, confirming that electrical noise was the issue.



When reading from two EZO™ Conductivity circuits, it is **strongly recommended** that they are electrically isolated from each other.

Basic EZO™
Inline Voltage Isolator

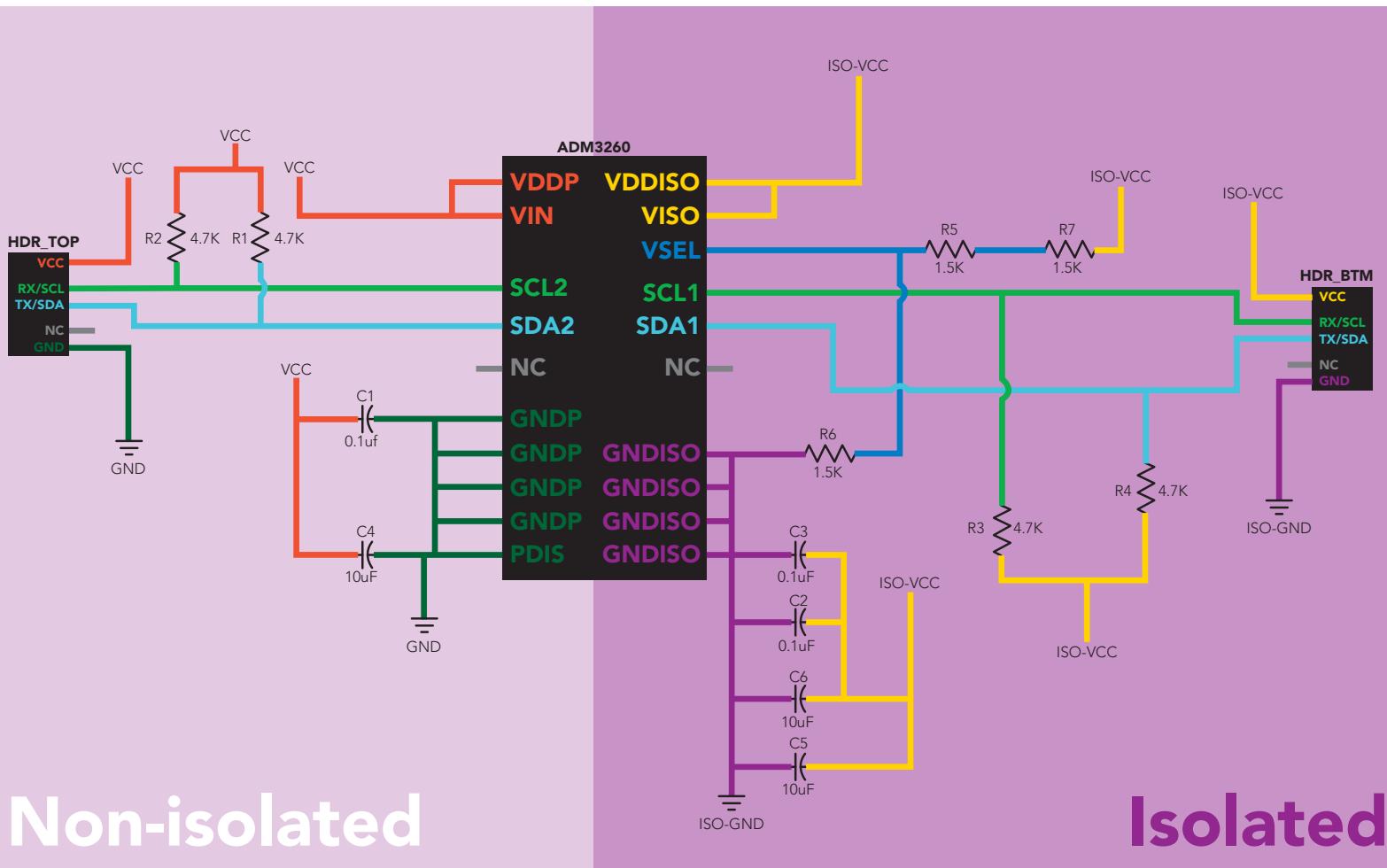


**Without isolation, Conductivity readings
will effect each other.**

This schematic shows exactly how we isolate data and power using the [ADM3260](#) and a few passive components. The ADM3260 can output isolated power up to 150 mW and incorporates two bidirectional data channels.

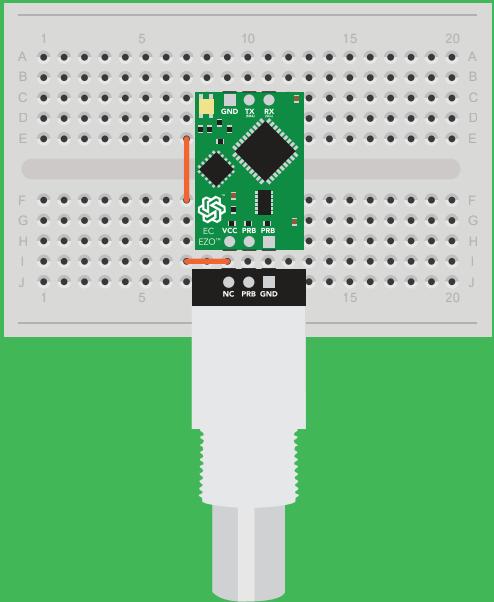
This technology works by using tiny transformers to induce the voltage across an air gap. PCB layout requires special attention for EMI/EMC and RF Control, having proper ground planes and keeping the capacitors as close to the chip as possible are crucial for proper performance. The two data channels have a $4.7\text{k}\Omega$ pull up resistor on both the isolated and non-isolated lines (R1, R2, R3, and R4) The output voltage is set using a voltage divider (R5, R6, and R7) this produces a voltage of 3.7V regardless of your input voltage.

Isolated ground is different from non-isolated ground, these two lines should not be connected together.

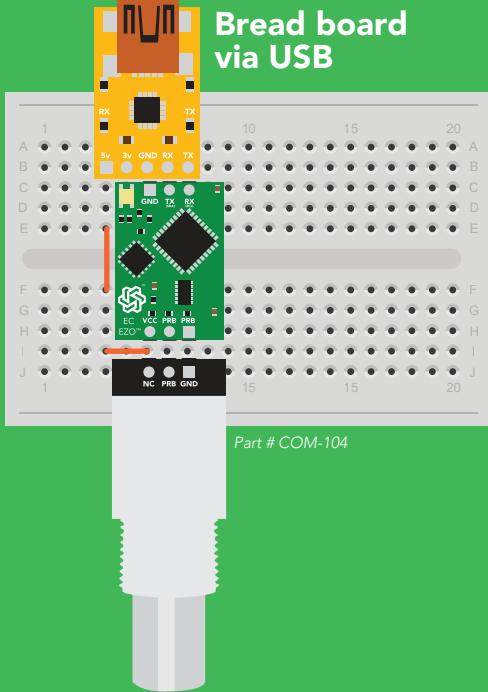


✓ Correct wiring

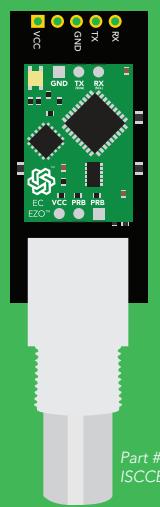
Bread board



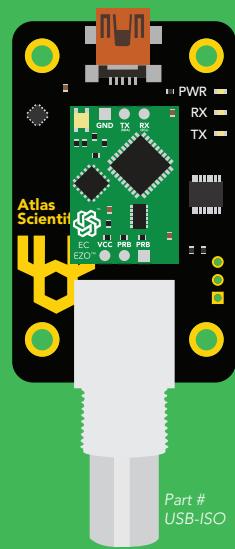
Bread board
via USB



Carrier board

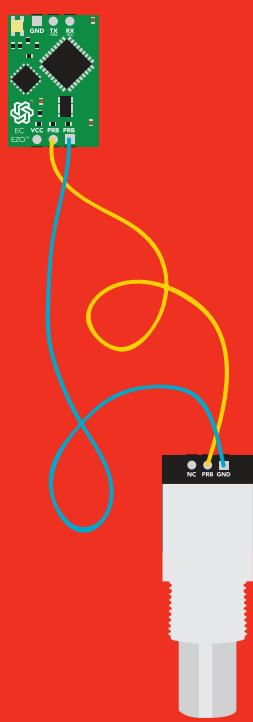


USB
carrier board

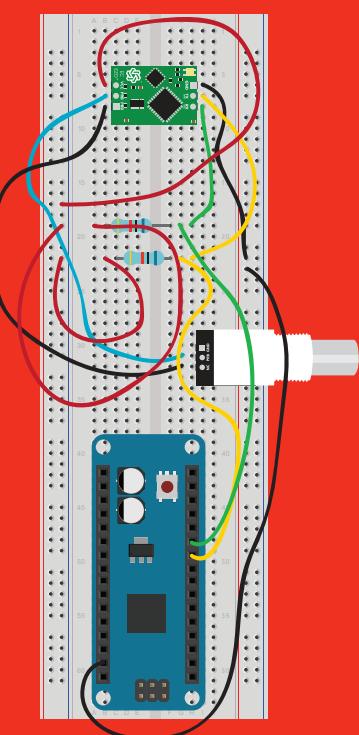


✗ Incorrect wiring

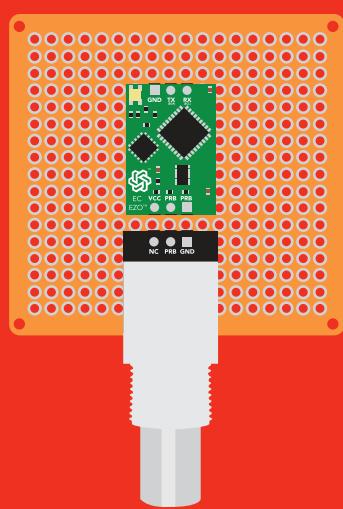
Extended leads



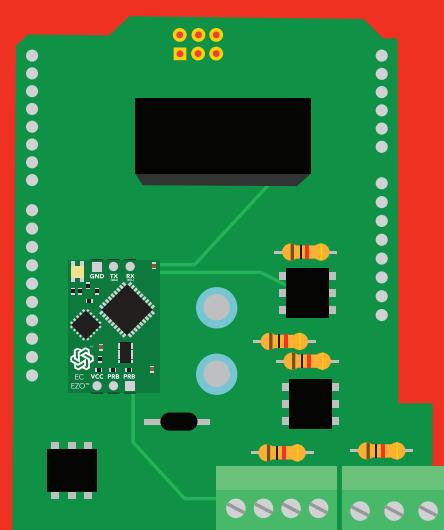
Sloppy setup



Perfboards or Protoboards



*Embedded into your device



NEVER
use Perfboards
or Protoboards

*Only after you are familiar
with EZO™ circuits operation

 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

Baud rate
Calibration
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

Find
Sleep mode
Temperature compensation

UART mode

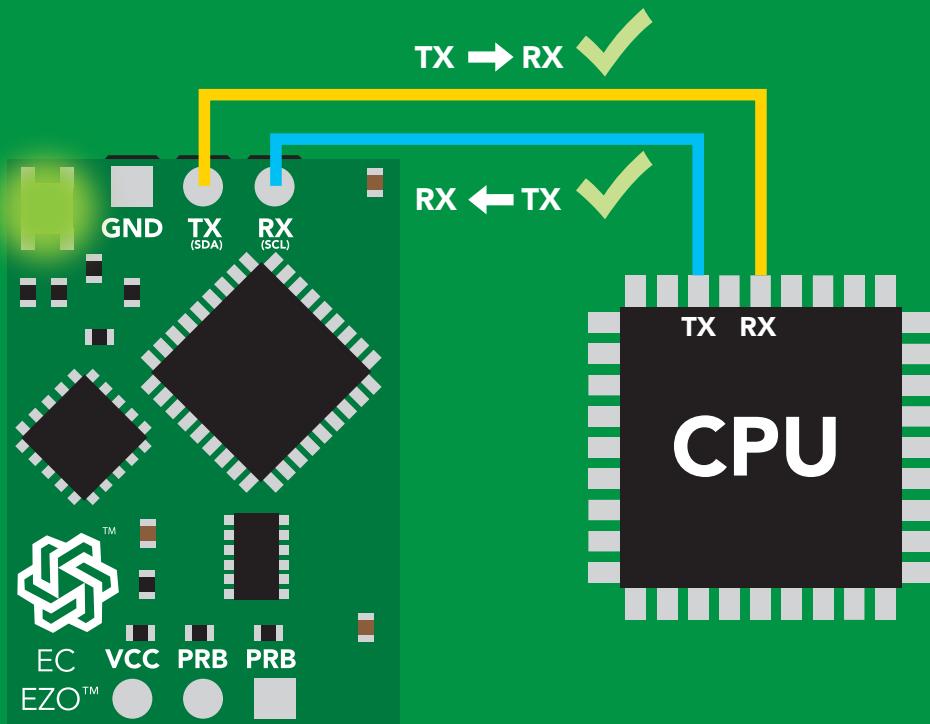
8 data bits no parity
1 stop bit no flow control

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 – 5.5V

Data format

Reading

Conductivity = $\mu\text{S}/\text{cm}$
Total dissolved solids = ppm
Salinity = PSU
Specific gravity (sea water only) = 1.00 – 1.300

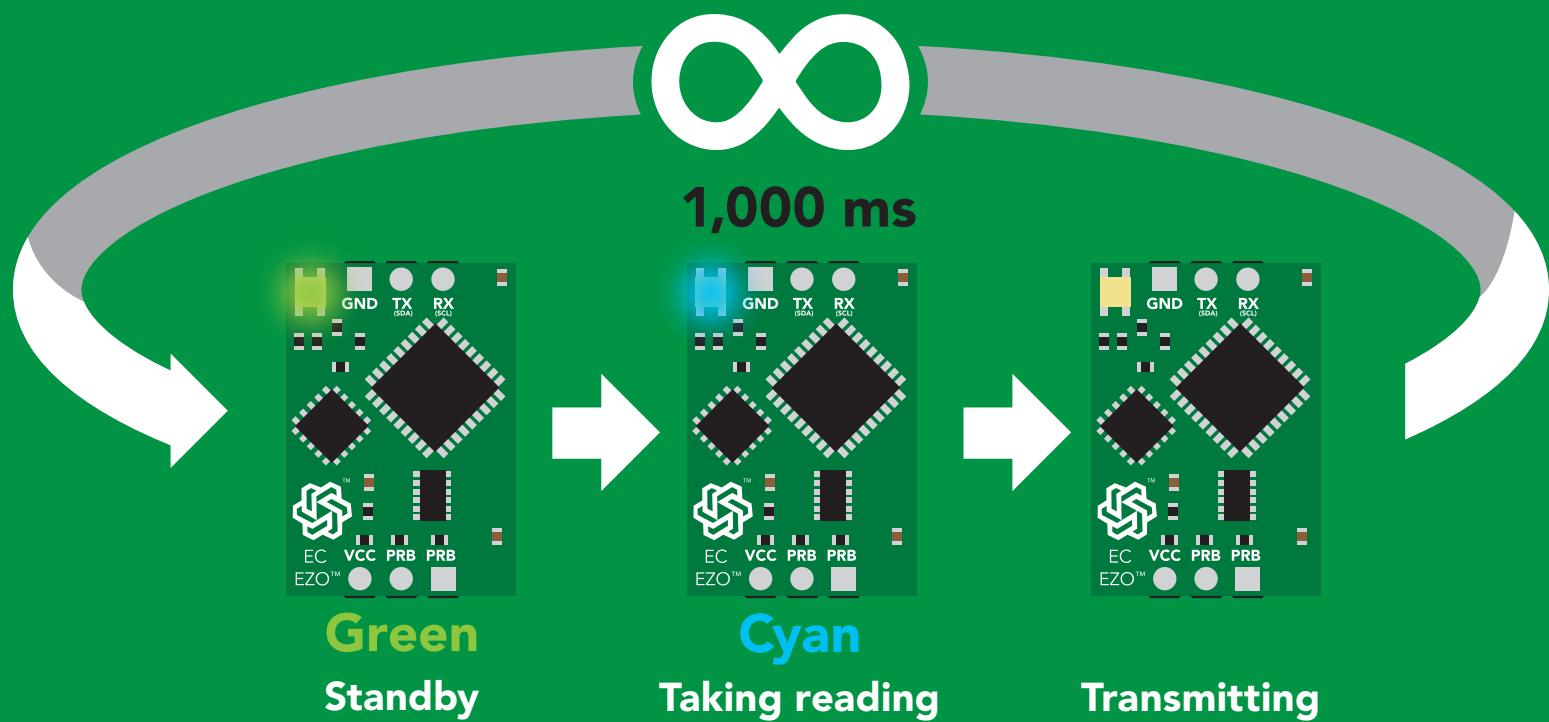
Units EC,TDS,SAL,SG
Encoding ASCII
Format string

Terminator

carriage return
floating point
Decimal places 3
Smallest string 3 characters
Largest string 40 characters

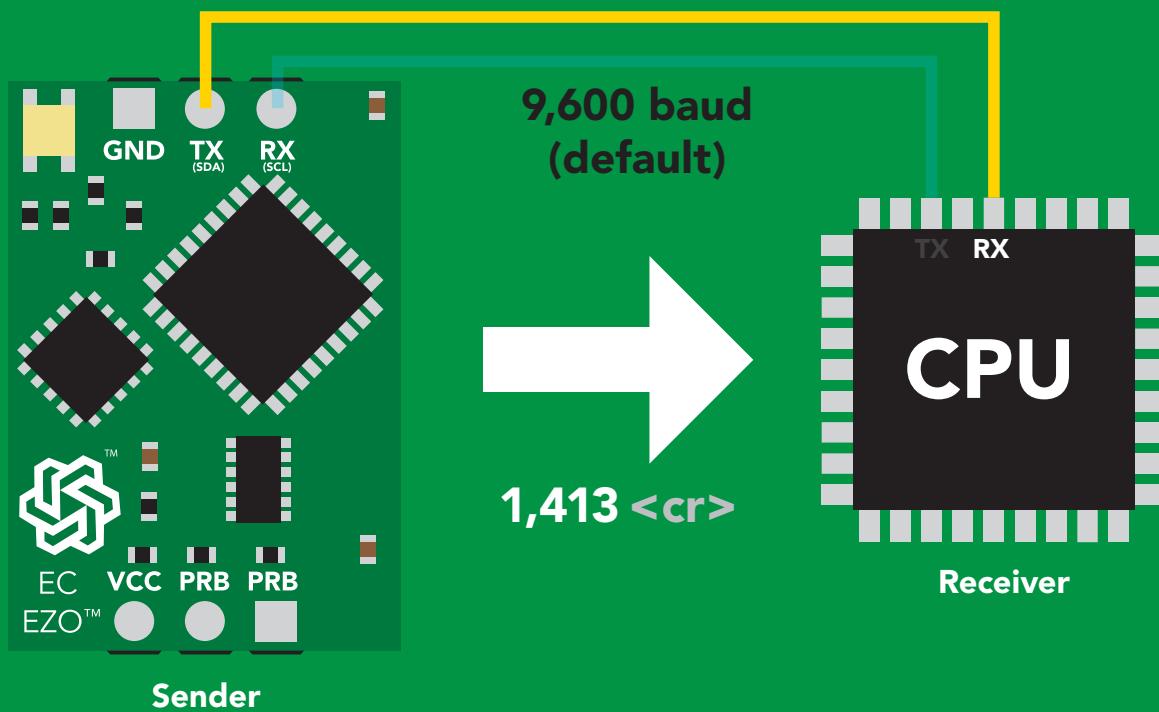
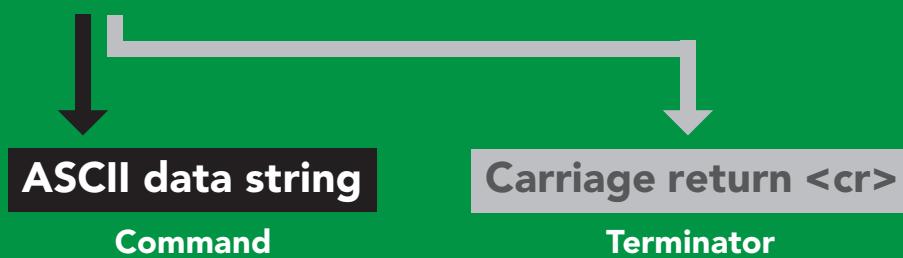
Default state

Mode	UART
Baud	9,600
Readings	continuous
Units	µS/cm
Speed	1 reading per second
LED	on



Receiving data from device

2 parts



Advanced

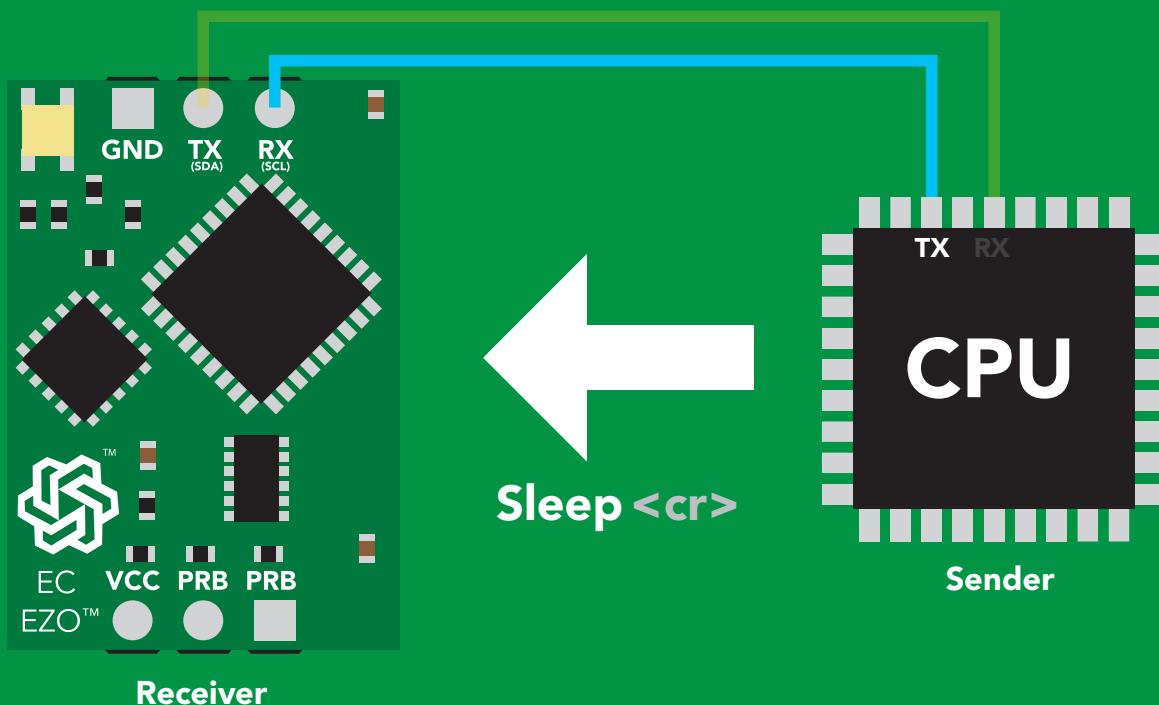
ASCII: 1 , 4 1 3 <cr>

Hex: 31 2C 34 31 33 0D

Dec: 49 44 52 49 51 13

Sending commands to device

2 parts



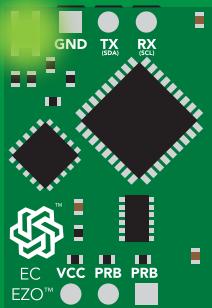
Advanced

ASCII: S I e e p <cr>

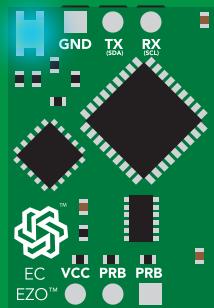
Hex: 53 6C 65 65 70 0D

Dec: 83 108 101 101 112 13

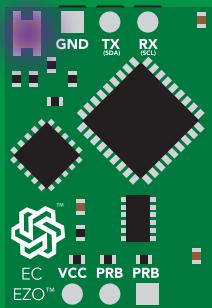
LED color definition



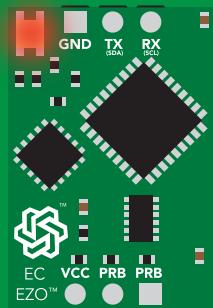
Green
UART standby



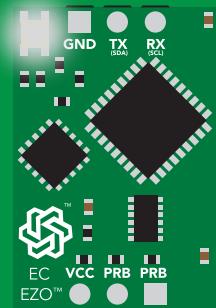
Cyan
Taking reading



Purple
Changing baud rate



Red
Command
not understood



White
Find

5V	LED ON +2.5 mA
3.3V	+1 mA

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 reading	pg. 25 enabled
Cal	performs calibration	pg. 27 n/a
Export/import	export/import calibration	pg. 28 n/a
Factory	enable factory reset	pg. 39 n/a
Find	finds device with blinking white LED	pg. 24 n/a
i	device information	pg. 33 n/a
I2C	change to I ² C mode	pg. 40 not set
K	Set probe type	pg. 29 K 1.0
L	enable/disable LED	pg. 23 enabled
Name	set/show name of device	pg. 32 not set
O	enable/disable parameters	pg. 31 all enabled
Plock	enable/disable protocol lock	pg. 38 disabled
R	returns a single reading	pg. 26 n/a
Sleep	enter sleep mode/low power	pg. 36 n/a
Status	retrieve status information	pg. 35 enable
T	temperature compensation	pg. 30 25°C
*OK	enable/disable response codes	pg. 34 enable

LED control

Command syntax

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

L,0 <cr> LED off

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

Example

L,1 <cr>

*OK <cr>

L,0 <cr>

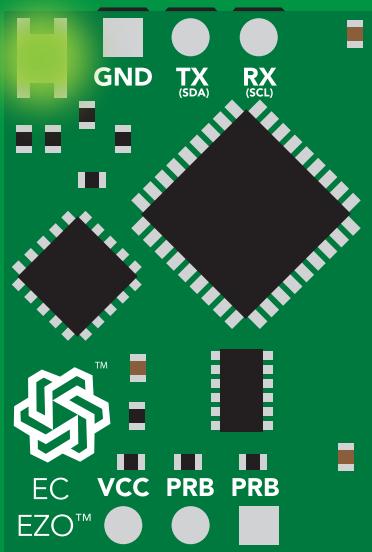
*OK <cr>

L,? <cr>

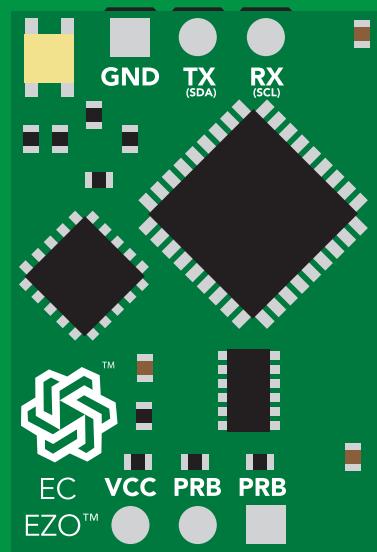
?L,1 <cr> or ?L,0 <cr>

*OK <cr>

L,1



L,0



Find

Command syntax

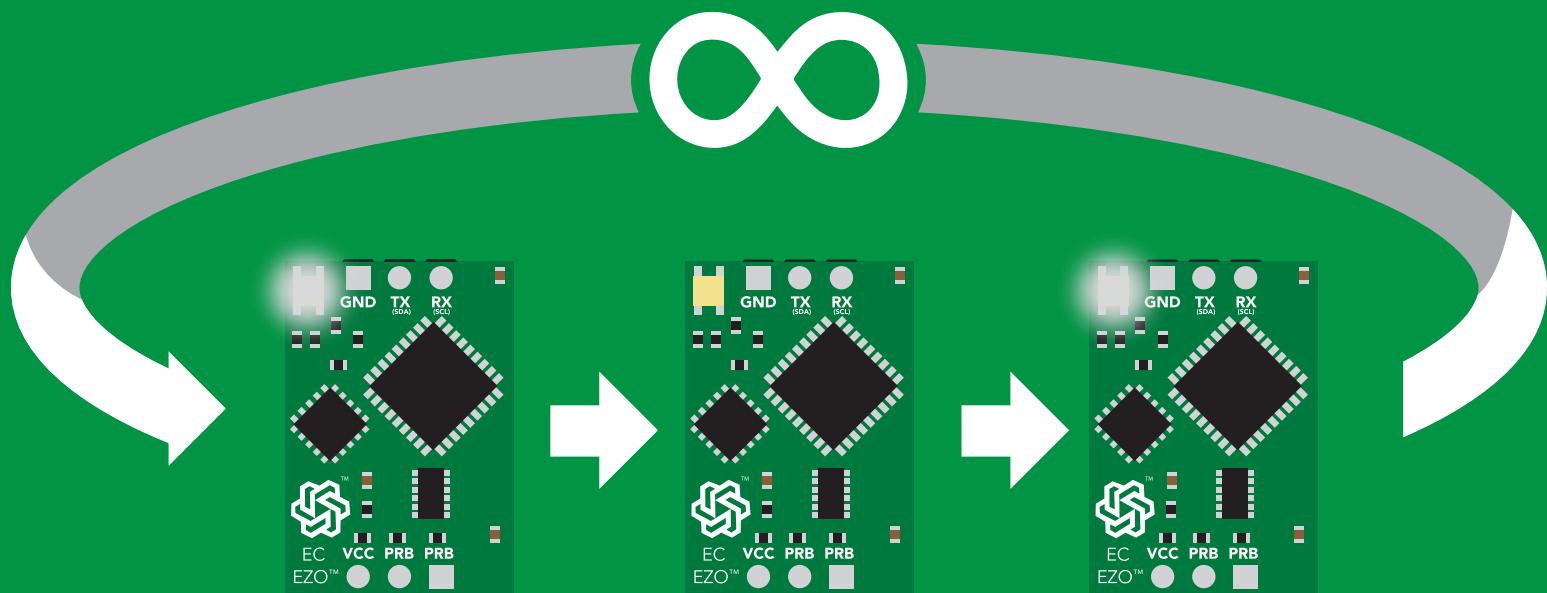
This command will disable continuous mode
Send any character or command to terminate find.

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

Example Response

Find <cr>

*OK <cr>



Continuous reading 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 reading mode on/off?

Example Response

C,1 <cr>	*OK <cr> EC,TDS,SAL,SG (1 sec) <cr> EC,TDS,SAL,SG (2 sec) <cr> EC,TDS,SAL,SG (3 sec) <cr>
C,30 <cr>	*OK <cr> EC,TDS,SAL,SG (30 sec) <cr> EC,TDS,SAL,SG (60 sec) <cr> EC,TDS,SAL,SG (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

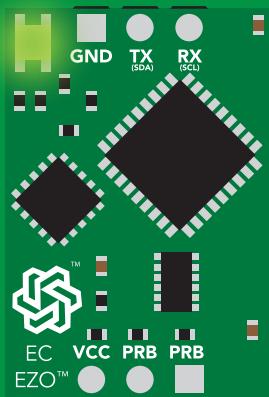
R <cr> takes single reading

Example Response

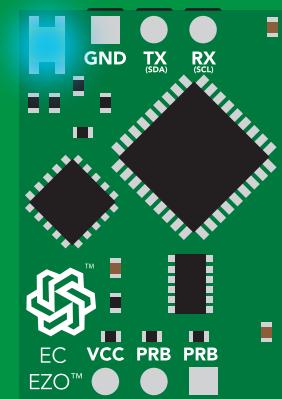
R <cr>

1,413 <cr>

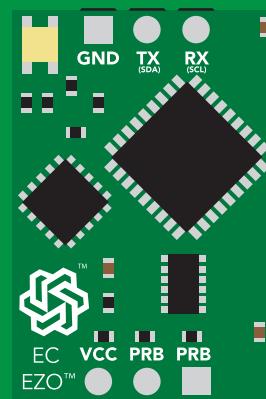
*OK <cr>



Green
Standby



Cyan
Taking reading



Transmitting



Calibration

Command syntax

Dry calibration must always be done first!

Cal,dry	<cr> dry calibration
Cal,n	<cr> single point calibration, where n = any value
Cal,low,n	<cr> low end calibration, where n = any value
Cal,high,n	<cr> high end calibration, where n = any value
Cal,clear	<cr> delete calibration data
Cal,?	<cr> device calibrated?

Example

Response

Cal,dry <cr>	*OK <cr>
Cal,84 <cr>	*OK <cr>
Cal,low,12880 <cr>	*OK <cr>
Cal,high,80000 <cr>	*OK <cr>
Cal,clear <cr>	*OK <cr>
Cal,? <cr>	?CAL,0 <cr> or ?CAL,1 <cr> or ?CAL,2 two point three point *OK <cr>

Two point calibration:

Step 1. "cal,dry"

Step 2. "cal,n"

Calibration complete!

Three point calibration:

Step 1 "cal,dry"

Step 2 "cal,low,n"

Step 3 "cal,high.n"

Calibration complete!

Export/import calibration

Command syntax

Export: Use this command to save calibration settings
Import: Use this command to load calibration settings to one or more devices.

Export <cr> export calibration string from calibrated device
Import <cr> import calibration string to new device
Export,? <cr> calibration string info

Example

Export,? <cr>

Response

10,120 <cr>

Response breakdown

10, 120

of strings to export

of bytes to export

Export strings can be up to 12 characters long, and is always followed by <cr>

Export <cr>

59 6F 75 20 61 72 <cr> (1 of 10)

Export <cr>

65 20 61 20 63 6F <cr> (2 of 10)

(7 more)

⋮

Export <cr>

6F 6C 20 67 75 79 <cr> (10 of 10)

Export <cr>

*DONE

Disabling *OK simplifies this process

Import, n
(FIFO)

Import, 59 6F 75 20 61 72 <cr> (1 of 10)

Setting the probe type

Command syntax

K 1.0 is the default value

K,n <cr> n = any value; floating point in ASCII

K,? <cr> probe K value?

Example

K,10 <cr>

Response

*OK <cr>

K,? <cr>

?K,10 <cr>

*OK <cr>



K 0.1



K 1.0



K 10

Temperature compensation

Command syntax

Default temperature = 25°C
Temperature is always in Celsius
Temperature is not retained if power is cut

T,n <cr> n = any value; floating point or int

T,? <cr> compensated temperature value?

RT,n <cr> set temperature compensation and take a reading*

This is a new command
for firmware V2.13

Example

T,19.5 <cr>

Response

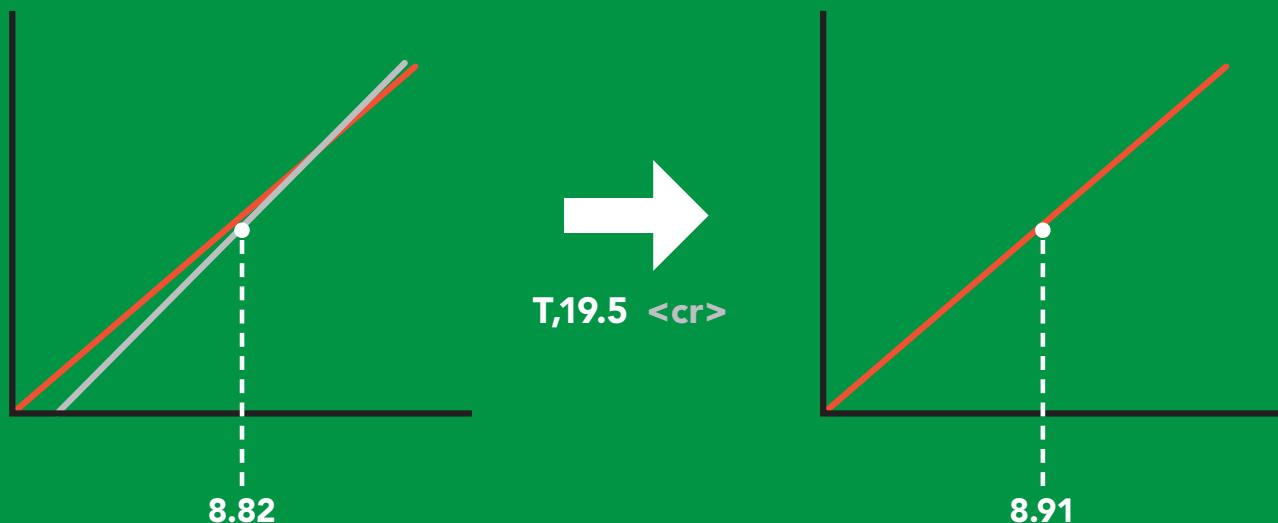
*OK <cr>

RT,19.5 <cr>

*OK <cr>
8.91 <cr>

T,? <cr>

?T,19.5 <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,EC,1 / O,EC,0 <cr>

Response

*OK <cr> enable / disable conductivity

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

*OK <cr> enable / disable total dissolved solids

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

*OK <cr> enable / disable salinity

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

*OK <cr> enable / disable specific gravity

O,? <cr>

?O,EC,TDS,S,SG <cr> if all are enabled

Parameters

EC conductivity

TDS total dissolved solids

S salinity

SG specific gravity

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

Name,n <cr> set name

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

Name,? <cr> show name

Up to 16 ASCII characters

Example

Name,zzt <cr>

*OK <cr>

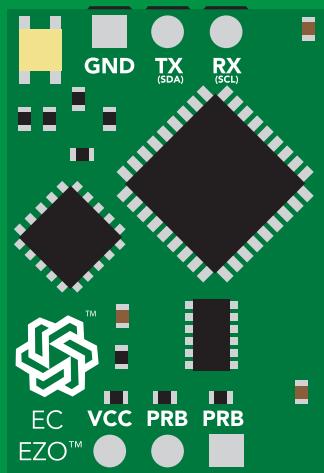
Name,? <cr>

?Name,zzt <cr>

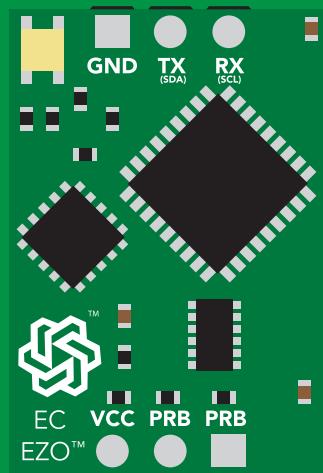
*OK <cr>

Response

Name,zzt



Name,?



*OK <cr>

Name,zzt <cr>
*OK <cr>

Device information

Command syntax

i <cr> device information

Example Response

i <cr>

**?i,EC,2.10 <cr>
*OK <cr>**

Response breakdown

?i, EC, 2.10

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>

1,413 <cr>
*OK <cr>

*OK,0 <cr>

no response, *OK disabled

R <cr>

1,413 <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

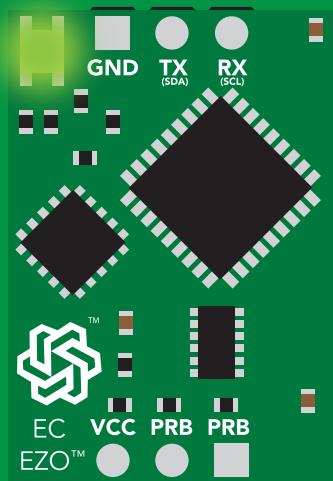
***SL**

Any command

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

	STANDBY	SLEEP
5V	18.14 mA	0.7 mA

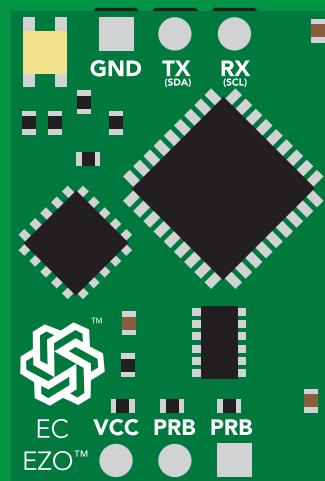
3.3V	16.85 mA	0.4 mA
-------------	-----------------	---------------



**Standby
18.14 mA**



Sleep <cr>



**Sleep
0.7 mA**

Change baud rate

Command syntax

Baud,n <cr> change baud rate

Example

Baud,38400 <cr>

Response

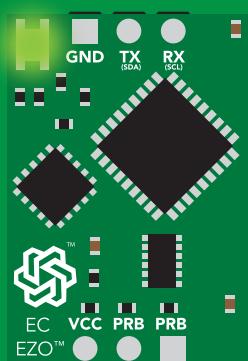
*OK <cr>

Baud,? <cr>

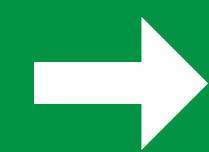
?Baud,38400 <cr>

*OK <cr>

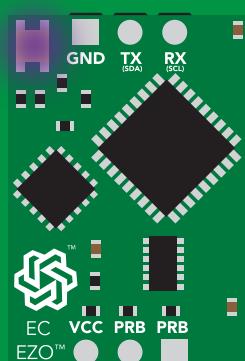
n = [300
1200
2400
9600 default
19200
38400
57600
115200]



Standby



Baud,38400 <cr>

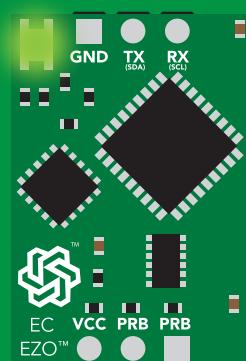


Changing baud rate

*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

Plock,1 <cr>

*OK <cr>

Plock,0 <cr>

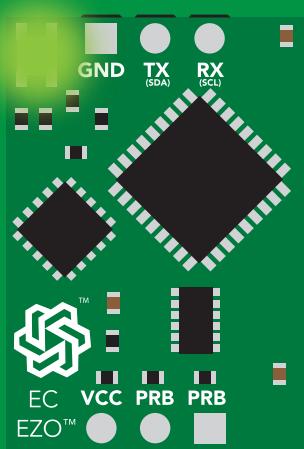
*OK <cr>

Plock,? <cr>

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

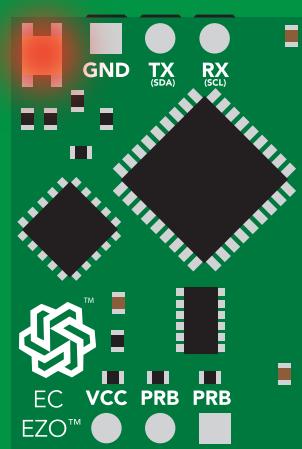
Response

Plock,1



*OK <cr>

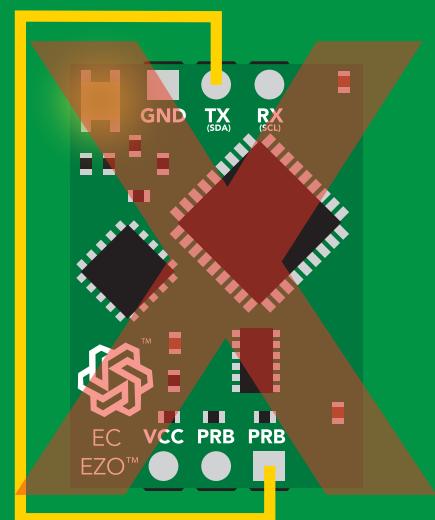
I²C,100



cannot change to I²C

*ER <cr>

Short



cannot change to I²C

Factory reset

Command syntax

Clears calibration
LED on
"*OK" enabled

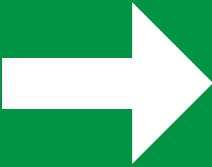
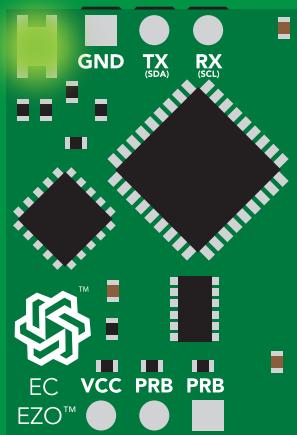
Factory <cr> enable factory reset

Example Response

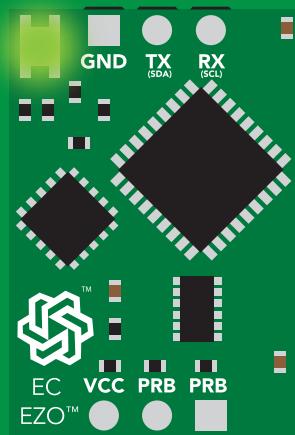
Factory <cr>

*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 100 (0x64)

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)

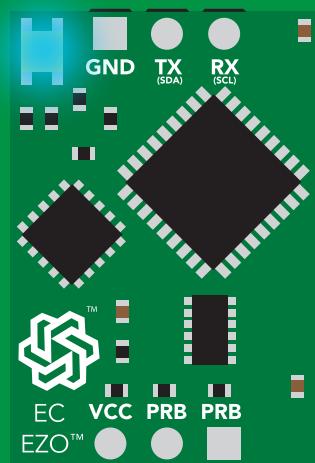
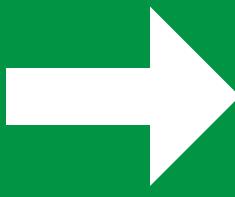
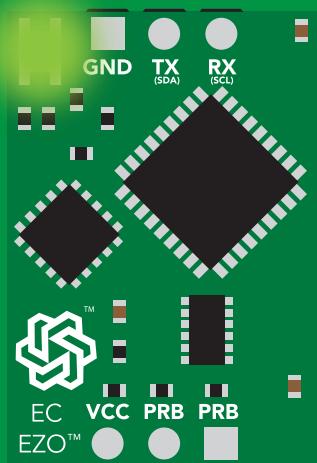
Wrong example

I²C,139 <cr> n ≠ 127

Response

*ER <cr>

I²C,100



Green
*OK <cr>

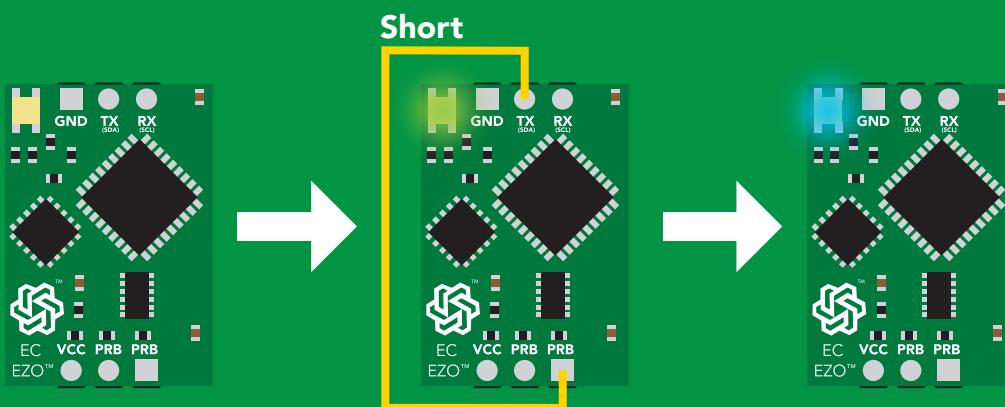
Blue
now in I²C mode

Manual switching to I²C

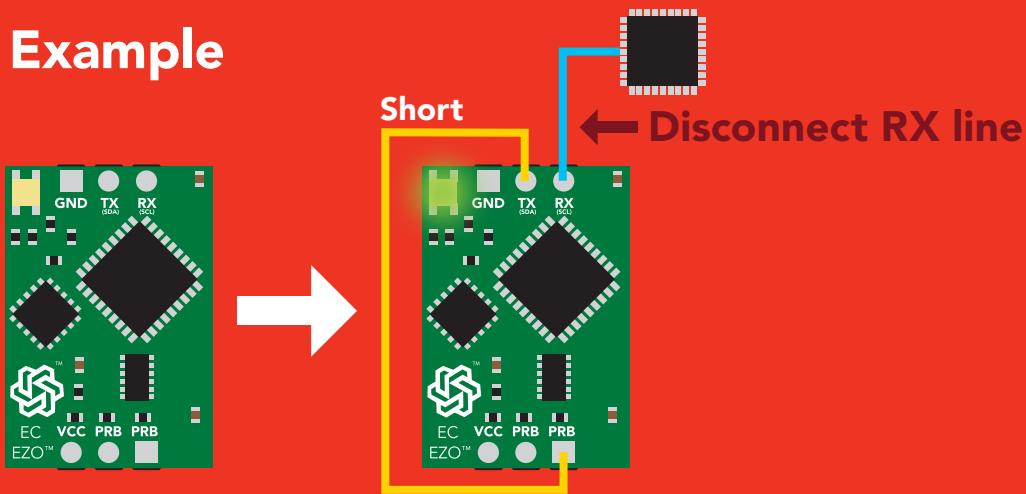
- Make sure Plock is set to 0
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to the right PRB
- Confirm RX is disconnected
- Connect ground (power on)
- Wait for LED to change from **Green** to **Blue**
- Disconnect ground (power off)
- Reconnect all data and power

Manually switching to I²C will set the I²C address to 100 (0x64)

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
- Enable/disable parameters
- Hardware switch to UART mode
- LED control
- Protocol lock
- Software switch to UART mode

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

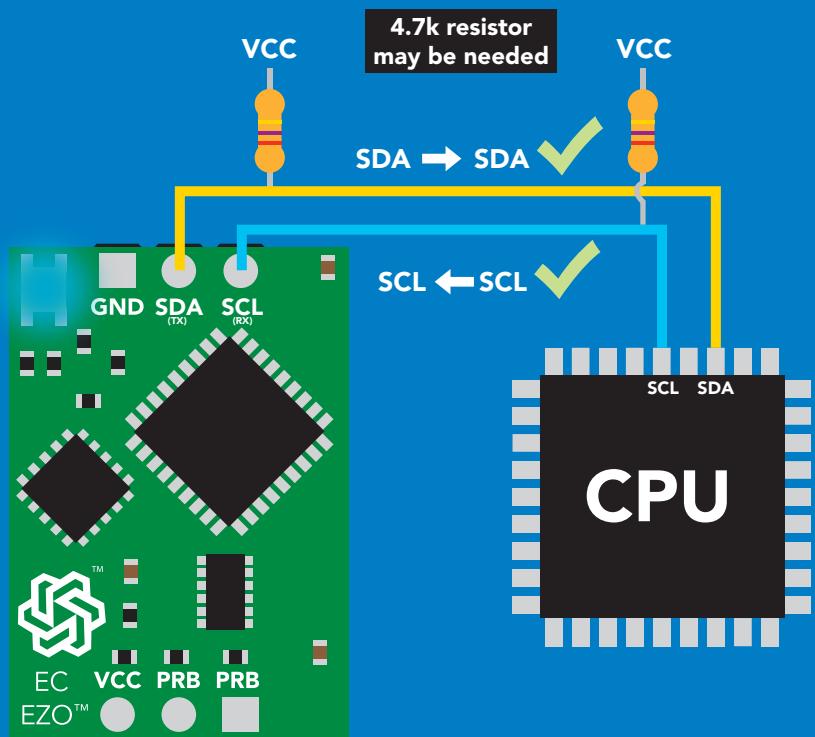
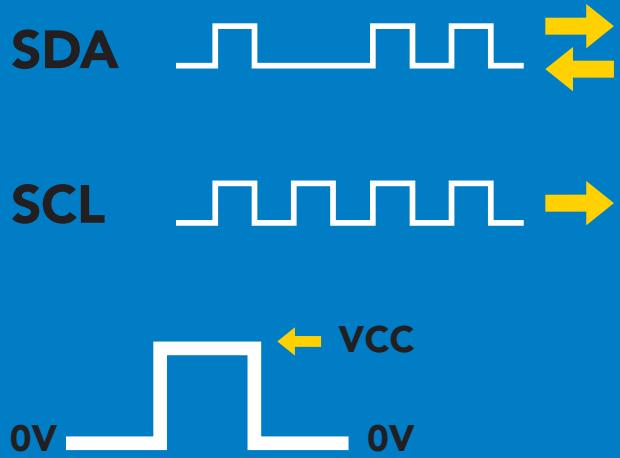
- Find
- Sleep mode
- Temperature compensation

I²C mode

I²C address (0x01 – 0x7F)
100 (0x64) default

V_{cc} 3.3V – 5.5V

Clock speed 100 – 400 kHz



Data format

Reading Conductivity = $\mu\text{S}/\text{cm}$
Total dissolved solids = ppm
Salinity = PSU
Specific gravity
(sea water only) = 1.00 – 1.300

Units EC, TDS, SAL, SG

Encoding ASCII

Format	string
Data type	floating point
Decimal places	3
Smallest string	3 characters
Largest string	399 characters

Sending commands to device

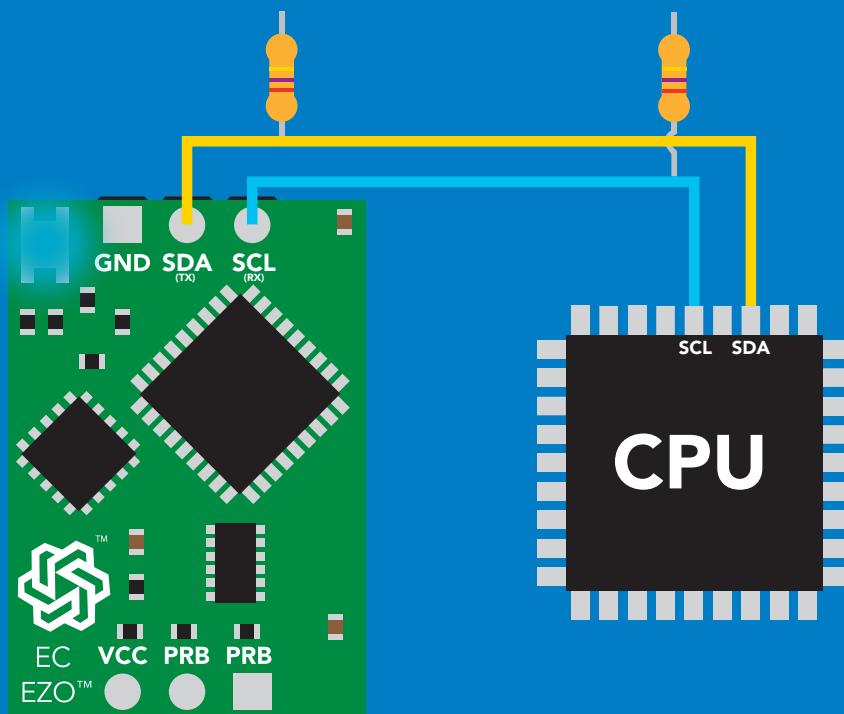
5 parts



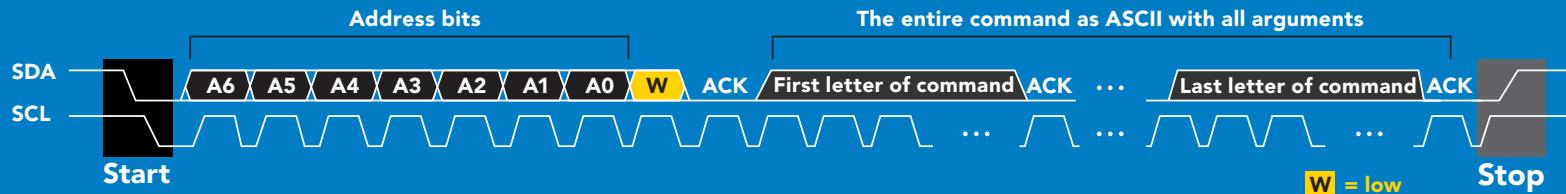
Example

Start 100 (0x64) Write Sleep Stop

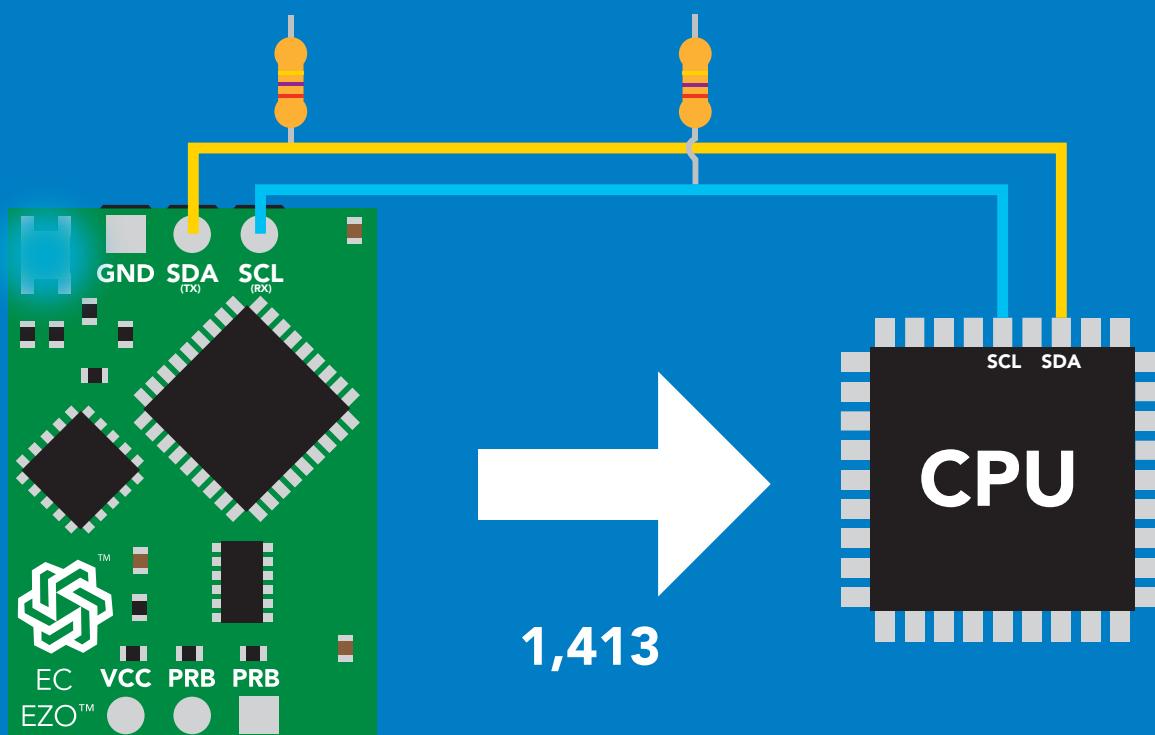
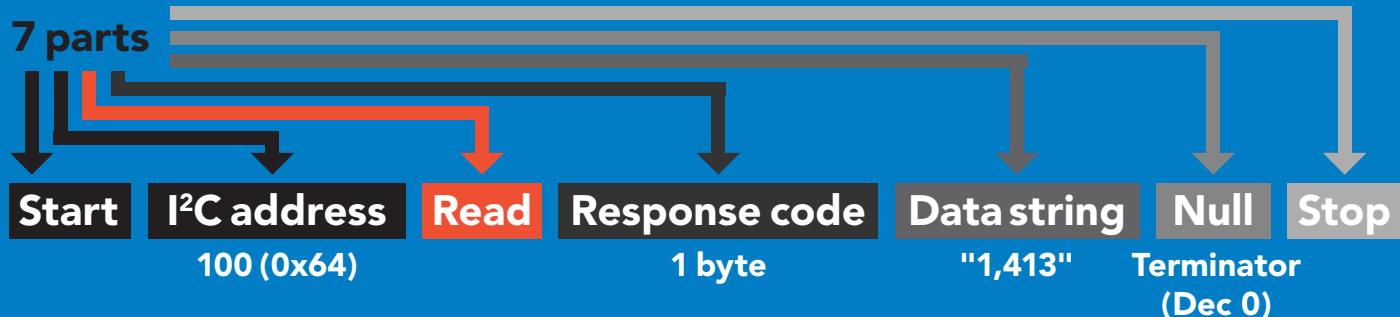
I²C address Command



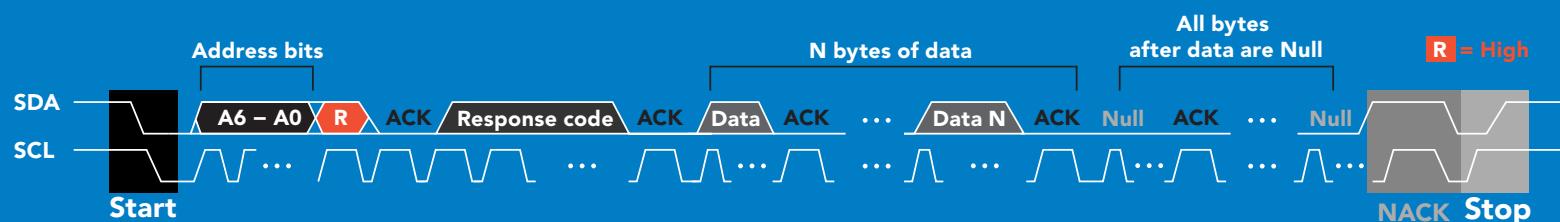
Advanced



Requesting data from device



Advanced

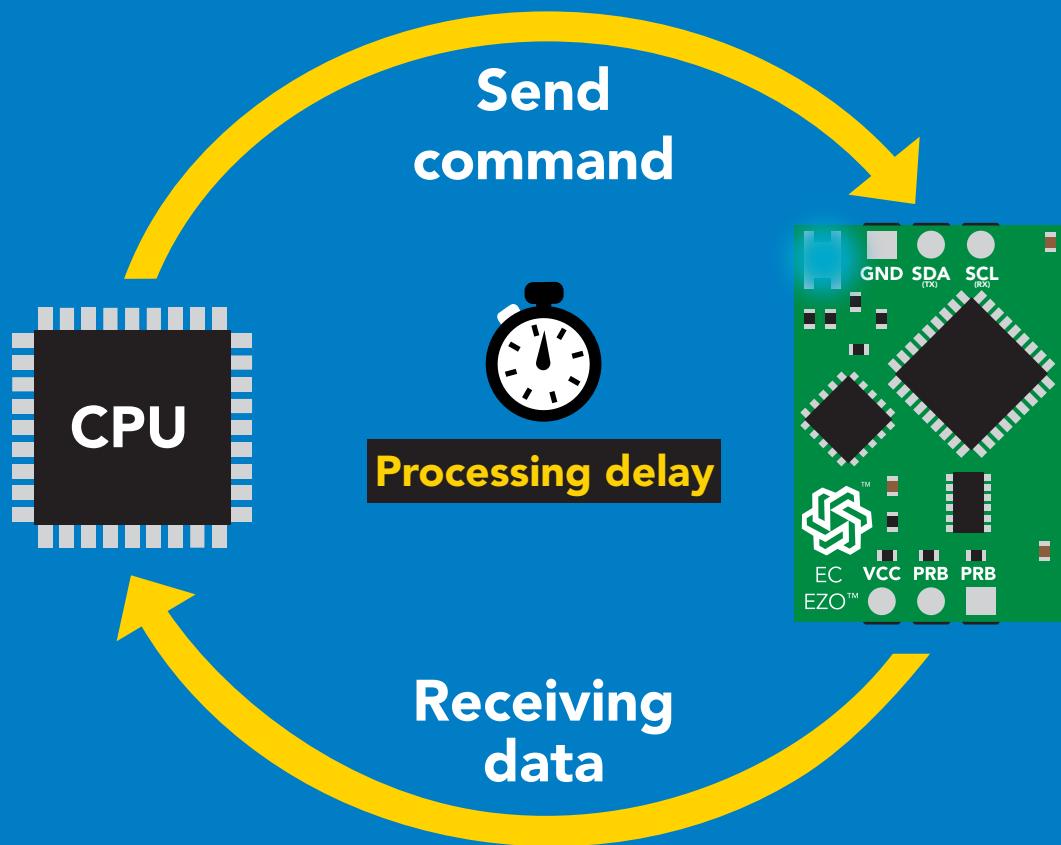


1 49 44 52 49 51 0 = 1,413
Dec Dec
ASCII

Response codes

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;
```

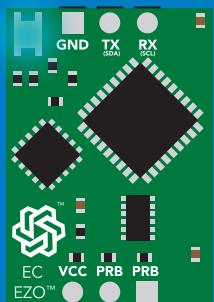
The response code will always be 254, if you do not wait for the processing delay.

Response codes

Single byte, not string

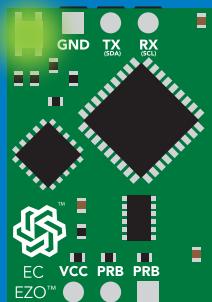
255	no data to send
254	still processing, not ready
2	syntax error
1	successful request

LED color definition



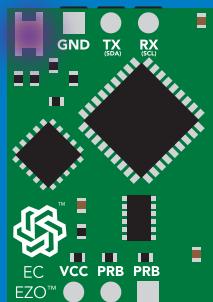
Blue

I²C standby



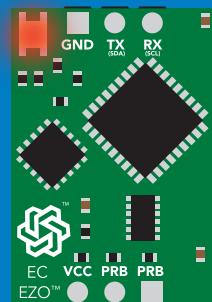
Green

Taking reading



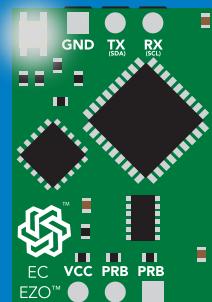
Purple

Changing I²C ID#



Red

Command not understood



White

Find

5V	LED ON +2.5 mA
3.3V	+1 mA

I²C 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 calibration	pg. 52
Export/import	export/import calibration	pg. 53
Factory	enable factory reset	pg. 62
Find	finds device with blinking white LED	pg. 50
i	device information	pg. 57
I2C	change I ² C address	pg. 61
K	Set probe type	pg. 54
L	enable/disable LED	pg. 49
O	enable/disable parameters	pg. 56
Plock	enable/disable protocol lock	pg. 60
R	returns a single reading	pg. 51
Sleep	enter sleep mode/low power	pg. 59
Status	retrieve status information	pg. 58
T	temperature compensation	pg. 55

LED control

Command syntax

300ms  processing delay

L,1 LED on **default**

L,0 LED off

L,? LED state on/off?

Example

L,1


Wait 300ms

1
Dec
Null

L,0


Wait 300ms

1
Dec
Null

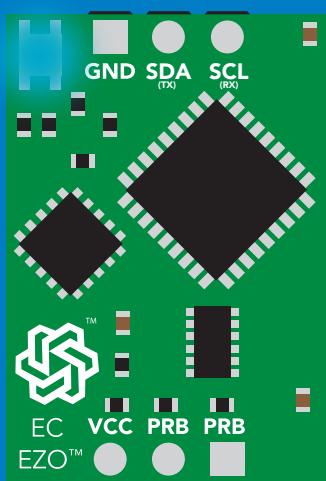
L,?


Wait 300ms

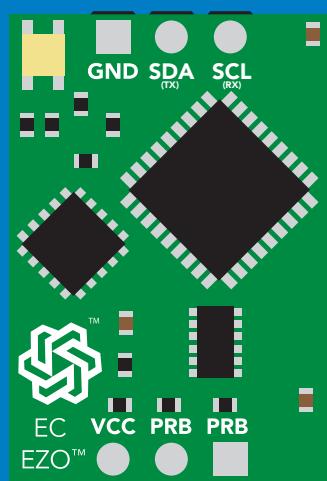
1 **?L,1** **0**
Dec ASCII Null

or

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



L,1



L,0

Find

300ms  processing delay

Command syntax

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

Find LED rapidly blinks white, used to help find device

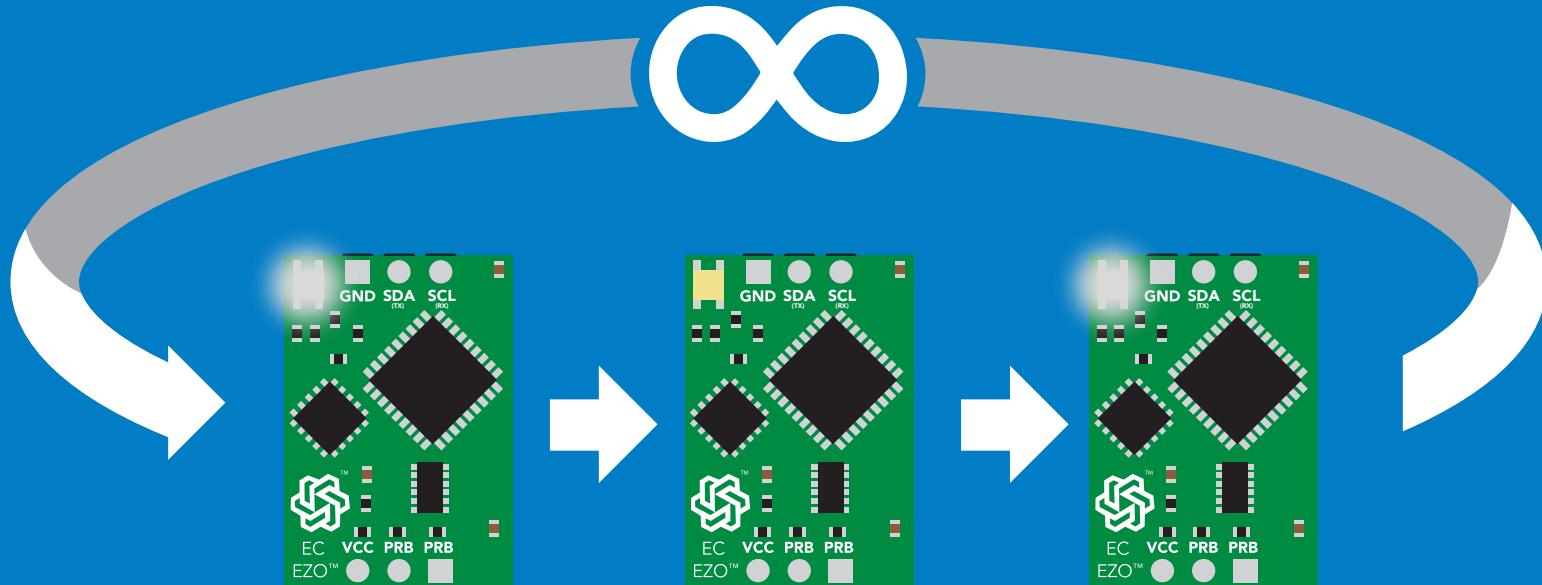
Example Response

Find


Wait 300ms

1
Dec
Null

Response



Taking reading

Command syntax

600ms  processing delay

R return 1 reading

Example

Response

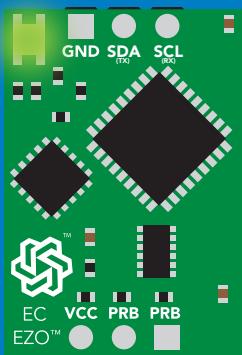
R



1
Dec

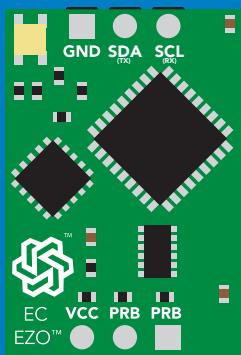
1,413
ASCII

0
Null

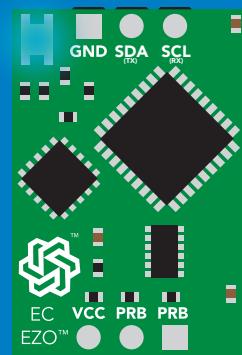


Green

Taking reading



Transmitting



Blue

Standby

Calibration

Command syntax

600ms  processing delay

Dry calibration must always be done first!

Cal,dry	dry calibration
Cal,n	single point calibration, where n = any value
Cal,low,n	low end calibration, where n = any value
Cal,high,n	high end calibration, where n = any value
Cal,clear	delete calibration data
Cal,?	device calibrated?

Example

Response

Cal,dry

 Wait 600ms **1** **0**
Dec Null

Cal,84

 Wait 600ms **1** **0**
Dec Null

Cal,low,12880

 Wait 600ms **1** **0**
Dec Null

Cal,high,80000

 Wait 600ms **1** **0**
Dec Null

Cal,clear

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

Cal,?

 Wait 300ms **1** **?CAL,0** **0** or **1** **?CAL,1** **0** or **1** **?CAL,2** **0**
Dec ASCII Null Dec ASCII Null Dec ASCII Null
two point three point

Two point calibration:

Step 1. "cal,dry"

Step 2. "cal,n"

Calibration complete!

Three point calibration:

Step 1 "cal,dry"

Step 2 "cal,low,n"

Step 3 "cal,high,n"

Calibration complete!

Export/import calibration

Command syntax

Export: Use this command to save calibration settings
Import: Use this command to load calibration settings to one or more devices.

Export

export calibration string from calibrated device

Import

import calibration string to new device

Export,?

calibration string info

300ms  processing delay

Example

Export,?

Response



1 Dec 10,120 ASCII 0 Null

Response breakdown

10, 120
↑ ↑
of strings to export # of bytes to export

Export strings can be up to 12 characters long

Export

(8 more)

Export

Export

Import, n
(FIFO)



1 Dec 59 6F 75 20 61 72 0 Null

(1 of 10)

⋮



1 Dec 65 20 61 20 63 6F 0 Null

(10 of 10)



1 Dec *DONE 0 Null

Import, 59 6F 75 20 61 72 (1 of 10)
ASCII

Setting the probe type

Command syntax

300ms  processing delay

K,n n = any value; floating point in ASCII

K 1.0 is the default value

K,? probe K value?

Example

Response

K,10

 Wait 300ms
1 Dec 0 Null

K,?

 Wait 600ms
1 Dec K,10 ASCII 0 Null



K 0.1



K 1.0



K 10

Temperature compensation

Command syntax

Default temperature = 25°C
Temperature is always in Celsius
Temperature is not retained if power is cut

- T,n n = any value; floating point or int 300ms  processing delay
- T,? compensated temperature value?
- RT,n set temperature compensation and take a reading*

This is a new command
for firmware V2.13

Example

T,19.5

Response

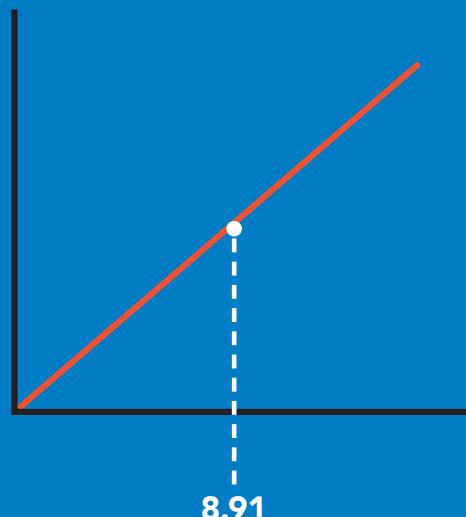
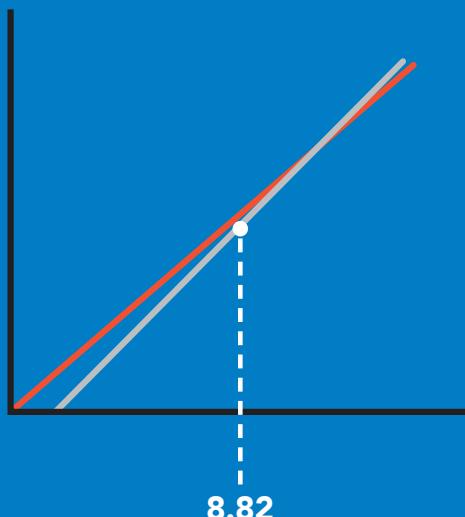
 Wait 300ms 1 Dec 0 Null

RT,19.5

 Wait 900ms 1 Dec 8.91 0 Null

T,?

 Wait 300ms 1 Dec ?T,19.5 0 Null



Enable/disable parameters from output string

Command syntax

300ms  processing delay

O, [parameter],[1,0]

enable or disable output parameter
enabled parameter?

Example

O,EC,1 / O,EC,0

Response

 Wait 300ms	1	Dec	0	enable / disable conductivity
 Wait 300ms	1	Dec	0	enable / disable total dissolved solids
 Wait 300ms	1	Dec	0	enable / disable salinity
 Wait 300ms	1	Dec	0	enable / disable specific gravity
O,?	1	?	O,EC,TDS,S,SG	0 if all are enabled

Parameters

EC conductivity
TDS total dissolved solids
S salinity
SG specific gravity

Followed by 1 or 0

1 enabled
0 disabled

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

Device information

Command syntax

300ms  processing delay

i device information

Example Response

i



Wait 300ms

1
Dec

?i,EC, 2.10
ASCII

0
Null

Response breakdown

?i, EC, 2.10
↑ ↑
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

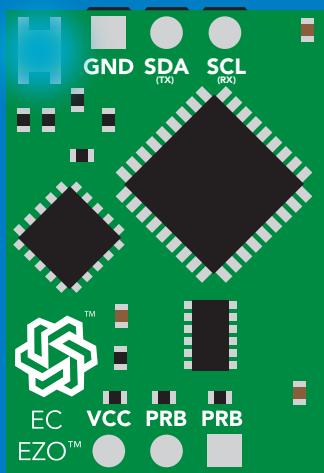
Do not read status byte after issuing sleep command.

Any command

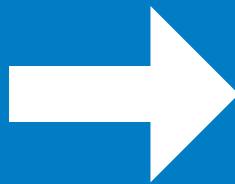
wakes up device

	STANDBY	SLEEP
5V	18.14 mA	0.7 mA

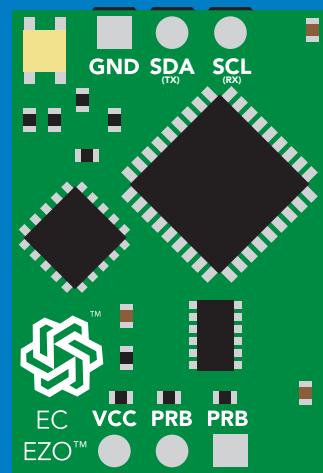
3.3V	16.85 mA	0.4 mA
-------------	-----------------	---------------



Standby



Sleep



Sleep

Protocol lock

Command syntax

300ms  processing delay

Plock,1 enable Plock

Locks device to I²C mode.

Plock,0 disable Plock

default

Plock,? Plock on/off?

Example

Plock,1


Wait 300ms

1
Dec
0
Null

Plock,0

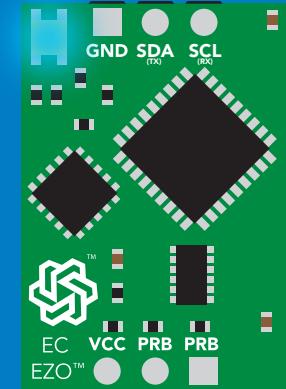

Wait 300ms

1
Dec
0
Null

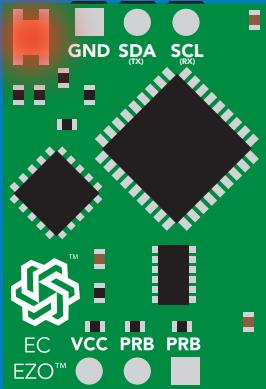
Plock,?


Wait 300ms

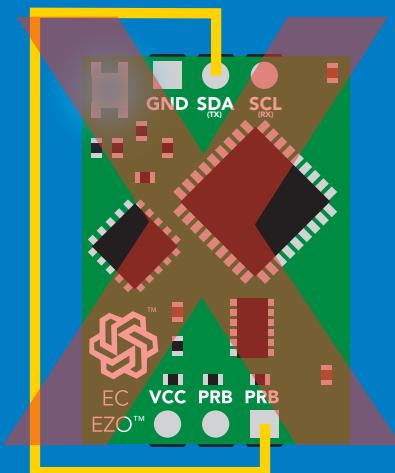
1
Dec
?Plock,1
ASCII
0
Null



Baud, 9600



cannot change to UART



cannot change to UART

I²C address change

Command syntax

300ms  processing delay

I²C,n sets I²C address and reboots into I²C mode

Example Response

I²C,101

device reboot

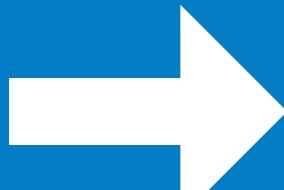
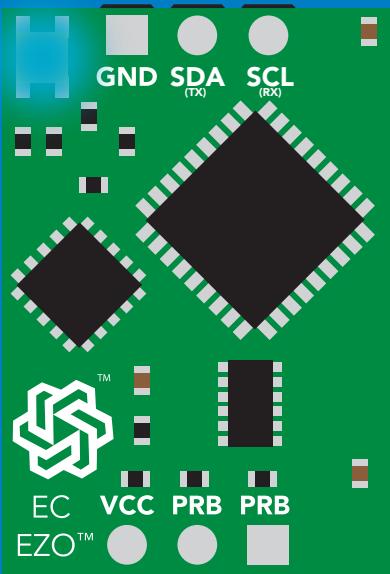
Warning!

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

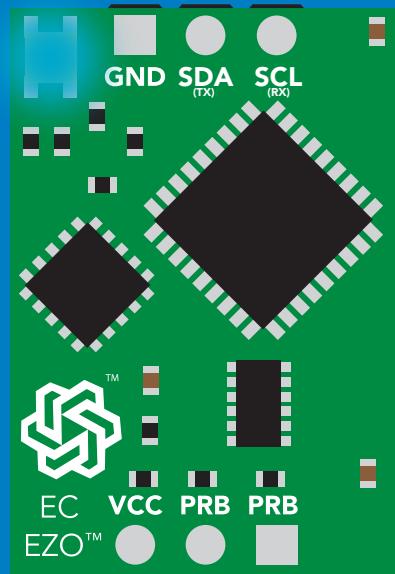
Default I²C address is 100 (0x64).

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

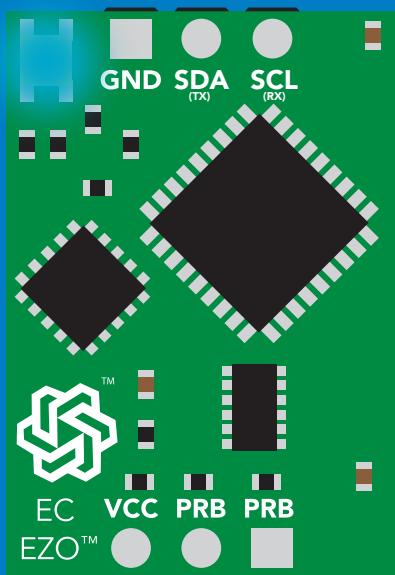
Example Response

Factory

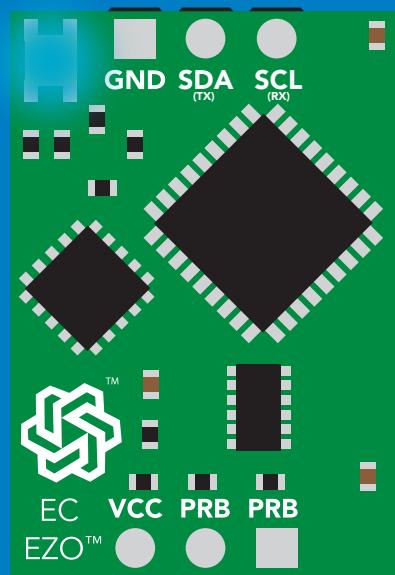
device reboot

Clears calibration
LED on
Response codes enabled

Factory



(reboot)



Change to UART mode

Command syntax

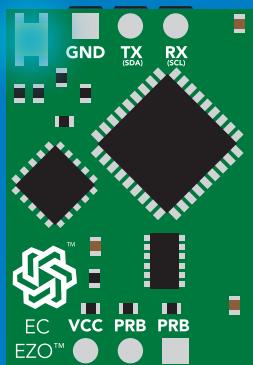
Baud,n switch from I²C to UART

Example Response

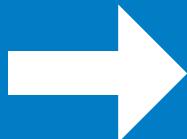
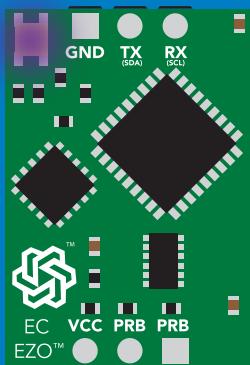
Baud,9600

reboot in UART mode

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

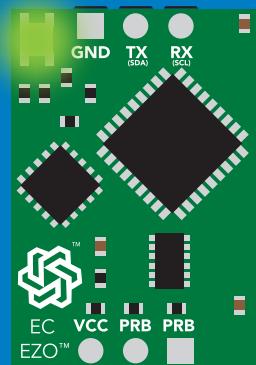


Baud,9600



(reboot)

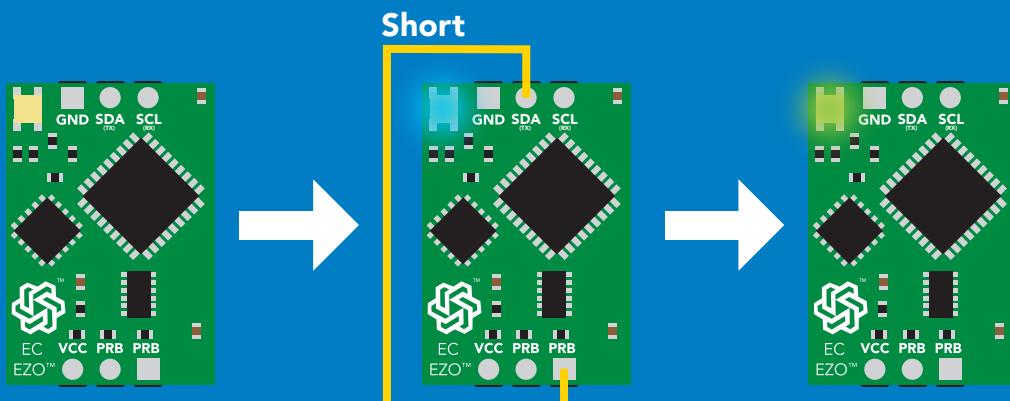
Changing to
UART mode



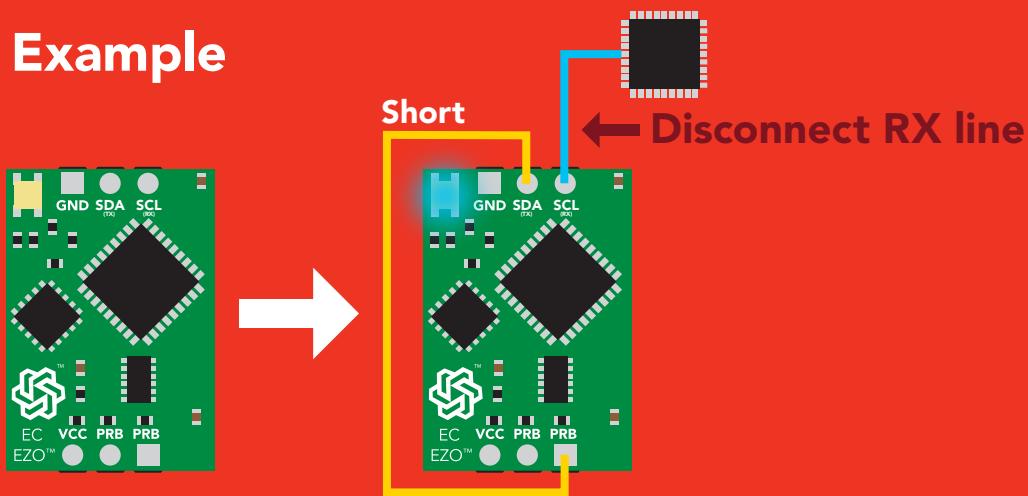
Manual switching to UART

- Make sure Plock is set to 0
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to the right PRB
- 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

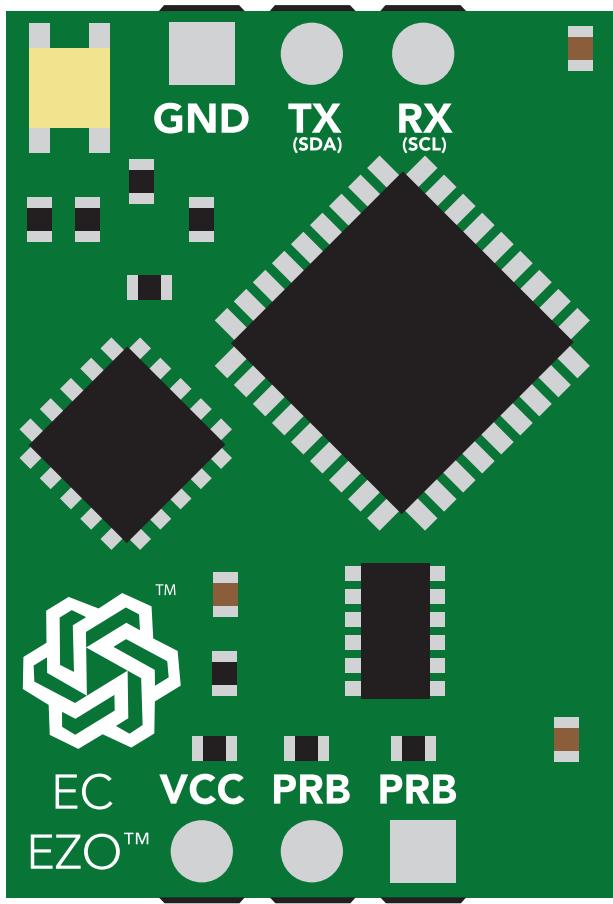
Example



Wrong Example



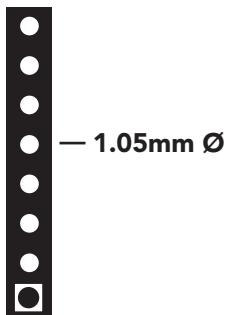
EZO™ circuit footprint



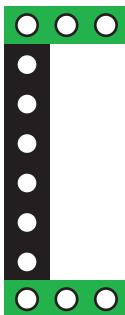
17.78mm
(0.7")

2.54mm
(0.1")

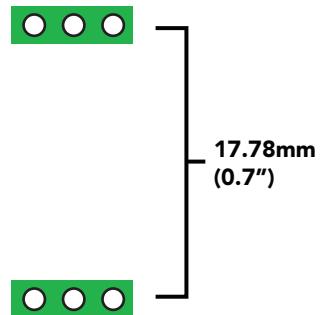
1 In your CAD software,
place a 8 position header.



2 Place a 3 position header at both
top and bottom of the 8 position.



3 Delete the 8 position header. The two 3
position headers are now 17.78mm (0.7")
apart from each other.



Datasheet change log

Datasheet V 5.3

Added new command:

"RT,n" for Temperature compensation located on pages 30 (UART) & 55 (I²C).
Added firmware information to Firmware update list.

Datasheet V 5.2

Revised calibration information on pages 27 & 52.

Datasheet V 5.1

Added more information about temperature compensation on pages 30 & 55.

Datasheet V 5.0

Changed "Max rate" to "Response time" on cover page.

Datasheet V 4.9

Removed note from certain commands about firmware version.
Added steps to calibration command pages 27 (UART) and 52 (I²C).

Datasheet V 4.8

Revised definition of response codes on pg 46.

Datasheet V 4.7

Revised cover page art.

Datasheet V 4.6

Updated calibration processing delay time on pg.52.

Datasheet V 4.5

Revised Enable/disable parameters information on pages 31 & 56.

Datasheet V 4.4

Updated High point calibration info on page 11.

Datasheet change log

Datasheet V 4.3

Updated calibration info on pages 27 (UART) and 52 (I²C).

Datasheet V 4.2

Revised Plock pages to show default value.

Datasheet V 4.1

Corrected I²C calibration delay on pg. 52.

Datasheet V 4.0

Revised entire datasheet.

Firmware updates

V1.0 – Initial release (April 17, 2014)

V1.1 – (June 2, 2014)

- Change specific gravity equation to return 1.0 when the uS reading is < 1000 (previously returned 0.0)
- Change accuracy of specific gravity from 2 decimal places to 3 decimal places
- Don't save temperature changes to EEPROM

V1.2 – (Aug 1, 2014)

- Baud rate change is now a long, purple blink

V1.5 – Baud rate change (Nov 6, 2014)

- Change default baud rate to 9600

V1.6 – I²C bug (Dec 1, 2014)

- Fixed I²C bug where the circuit may inappropriately respond when other I²C devices are connected.

V1.8 – Factory (April 14, 2015)

- Changed "X" command to "Factory"

V1.95 – Plock (March 31, 2016)

- Added protocol lock feature "Plock"

V1.96 – EEPROM (April 26, 2016)

- Fixed glitch where EEPROM would get erased if the circuit lost power 900ms into startup

V2.10 – (April 12, 2017)

- Added "Find" command.
- Added "Export/import" command.
- Modified continuous mode to be able to send readings every "n" seconds.
- Default output changed from CSV string of 4 values to just conductivity; Other values must be enabled.

V2.11 – (April 28, 2017)

- Fixed "Sleep" bug, where it would draw excessive current.

V2.12 – (May 9, 2017)

- Fixed glitch in sleep mode, where circuit would wake up to a different I²C address.

V2.13 – (July 16, 2018)

- Added "RT" command to Temperature compensation.

Warranty

Atlas Scientific™ Warranties the EZO™ class Conductivity circuit to be free of defect during the debugging phase of device implementation, or 30 days after receiving the EZO™ class Conductivity circuit (which ever comes first).

The debugging phase

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

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