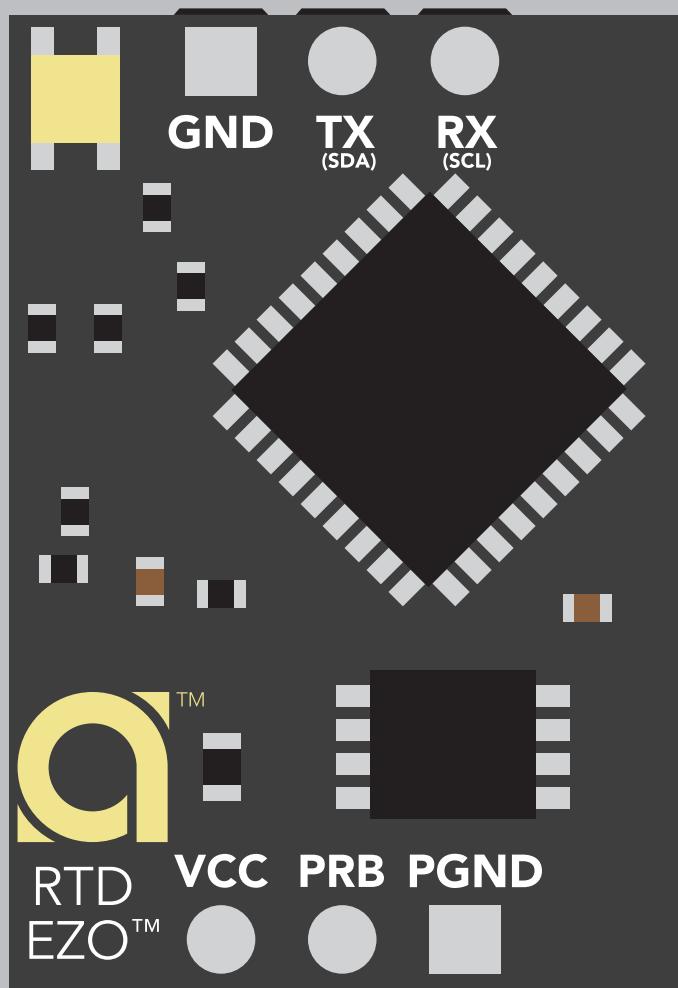


# EZO-RTD™

Embedded Temperature Circuit

Reads	<b>Temperature</b>
Range	<b>-126.000 °C – 1254 °C</b>
Resolution	<b>0.001</b>
Accuracy	<b>+/- (0.1 + 0.0017 x °C)</b>
Response time	<b>1 reading per sec</b>
Supported probes	<b>Any type &amp; brand PT-100 or PT-1000 RTD</b>
Calibration	<b>Single point</b>
Temperature output	<b>°C, °K, or °F</b>
Data protocol	<b>UART &amp; I<sup>2</sup>C</b>
Default I <sup>2</sup> C address	<b>102 (0x66)</b>
Operating voltage	<b>3.3V – 5.5V</b>
Data format	<b>ASCII</b>
Onboard Data Logger	<b>50 Readings</b>



**Electrical Isolation not needed**





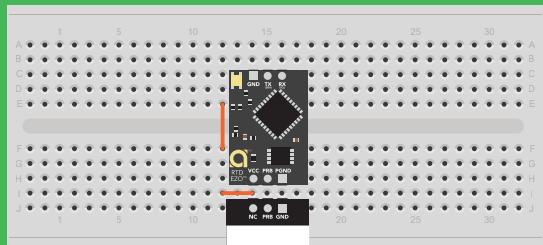
# STOP

**SOLDERING THIS DEVICE VOIDS YOUR WARRANTY.**

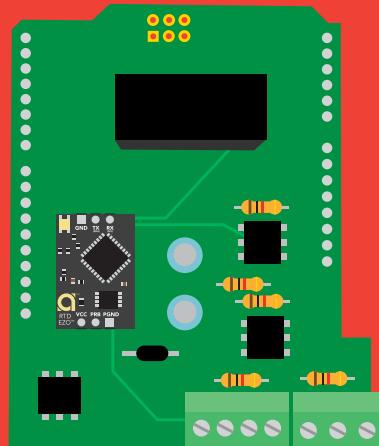
**This is sensitive electronic equipment. Get this device working in a solderless breadboard first. Once this device has been soldered it is no longer covered by our warranty.**

**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	Using other brand PT-100/PT-1000	7
Power consumption	4	Operating principle	9
Absolute max ratings	4	Correct wiring	11
Temperature circuit range	5	Calibration theory	12
Temperature circuit accuracy	5	On board data logger	13
Atlas Scientific PT-1000 probe	6	Default state	14
		Available data protocols	15

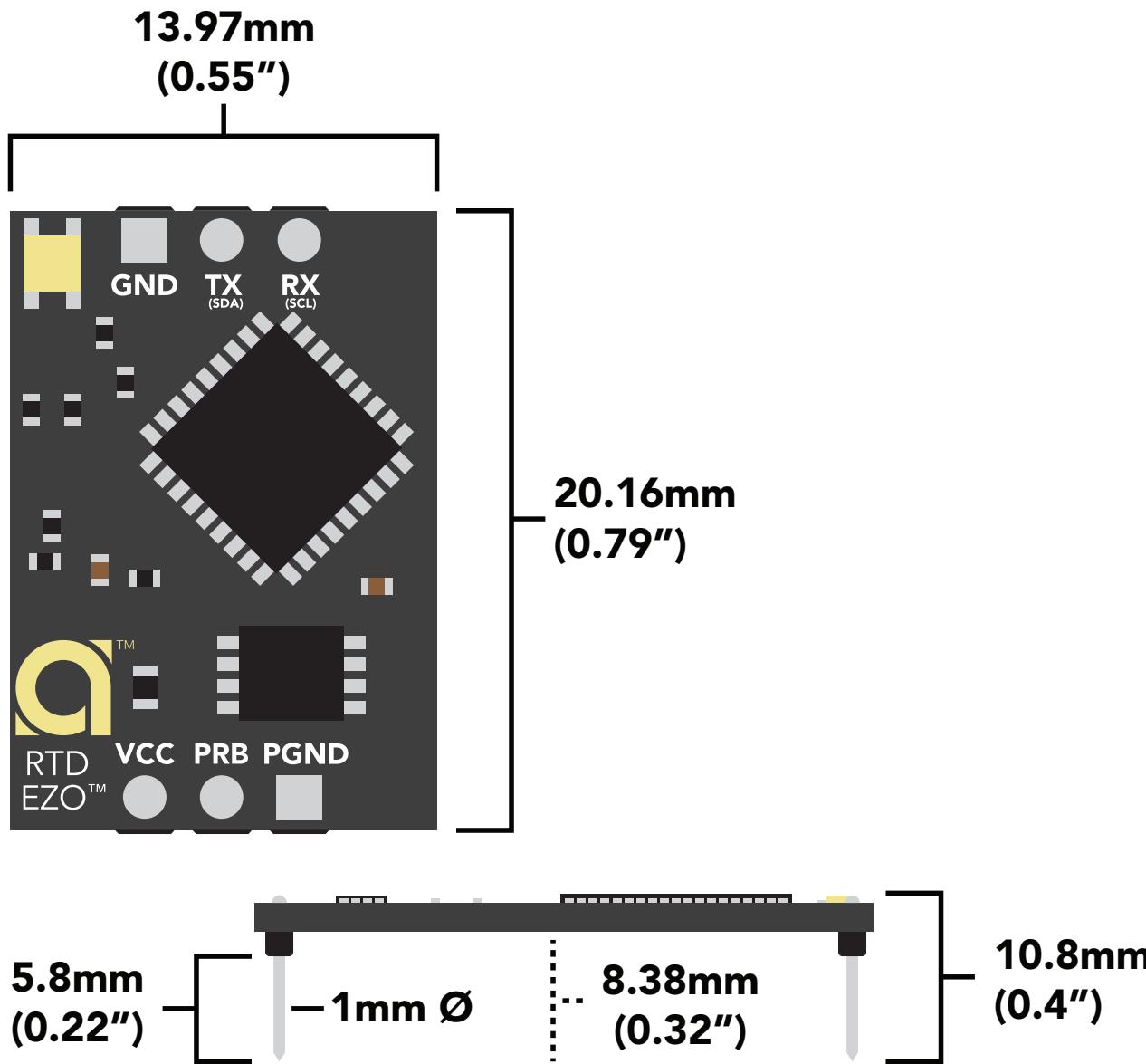
## UART

UART mode	17
Receiving data from device	18
Sending commands to device	19
LED color definition	20
<b>UART quick command page</b>	<b>21</b>
LED control	22
Find	23
Continuous reading mode	24
Single reading mode	25
Calibration	26
Export calibration	27
Import calibration	28
Temperature scale	29
Enable/disable data logger	30
Memory recall	31
Memory clear	32
Naming device	33
Device information	34
Response codes	35
Reading device status	36
Sleep mode/low power	37
Change baud rate	38
Protocol lock	39
Factory reset	40
Change to I <sup>2</sup> C mode	41
Manual switching to I <sup>2</sup> C	42

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

I <sup>2</sup> C mode	44
Sending commands	45
Requesting data	46
Response codes	47
LED color definition	48
<b>I<sup>2</sup>C quick command page</b>	<b>49</b>
LED control	50
Find	51
Taking reading	52
Calibration	53
Export calibration	54
Import calibration	55
Temperature scale	56
Enable/disable data logger	57
Memory recall	58
Memory clear	59
Naming device	60
Device information	61
Reading device status	62
Sleep mode/low power	63
Protocol lock	64
I <sup>2</sup> C address change	65
Factory reset	66
Change to UART mode	67
Manual switching to UART	68
Circuit footprint	69
Datasheet change log	70
Warranty	72

# EZO™ circuit dimensions



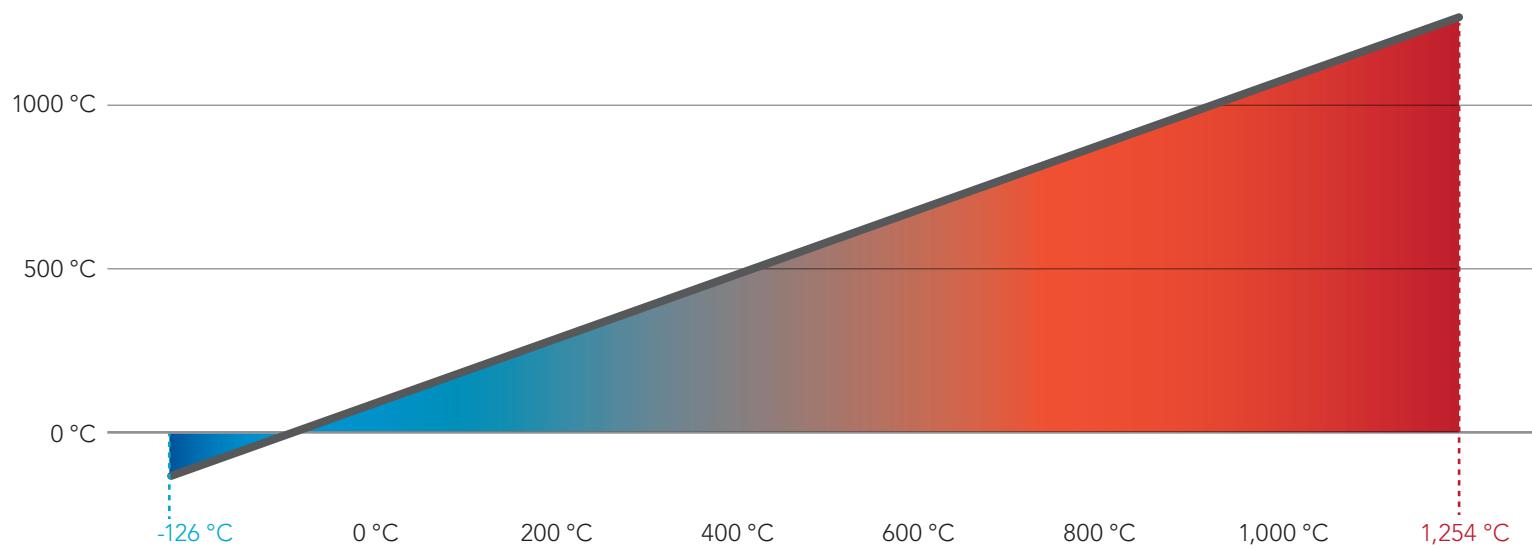
## Power consumption

	LED	MAX	STANDBY	SLEEP
5V	ON	16 mA	15.4 mA	0.4 mA
	OFF	15.3 mA	15 mA	
3.3V	ON	14.3 mA	13.8 mA	0.09 mA
	OFF	14 mA	13.6 mA	

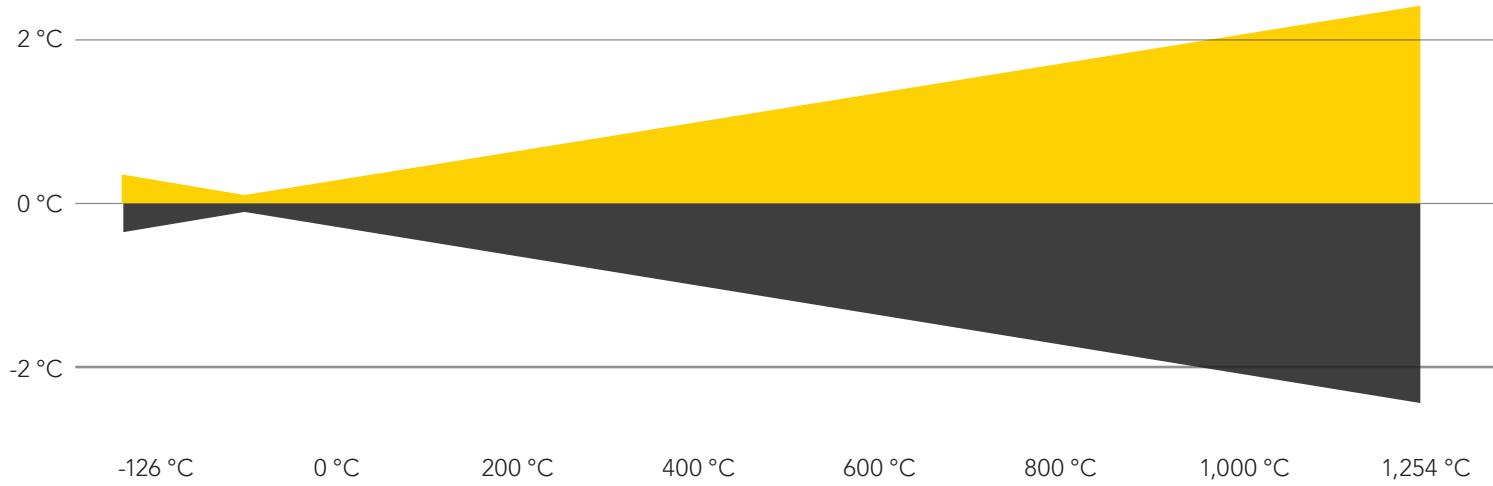
## Absolute max ratings

Parameter	MIN	TYP	MAX
Storage temperature (EZO™ RTD)	-65 °C		125 °C
Operational temperature (EZO™ RTD)	-40 °C	25 °C	85 °C
VCC	3.3V	5V	5.5V

# EZO™ RTD temperature sensing range



# EZO™ RTD temperature sensing accuracy



# Atlas Scientific PT-1000 probe

- Accuracy +/- (0.15 + (0.002\*t))
- Probe type: class A platinum, RTD
- Cable length: 81cm (32")
- Cable material: silicone rubber
- 30mm sensing area (304 SS)
- 6mm diameter
- BNC connector
- Reaction time: 90% value in 13 seconds
- Probe output: analog
- Full sensing range -200 °C to 850 °C
- Cable max temp 125 °C
- Cable min temp -55 °C

The Atlas Scientific EZO™ RTD Temperature circuit only works with PT-100 and PT-1000 probes.



To read temperatures above, or below the max cable temperature, an additional probe housing (thermowell) is needed to protect the cable.



100mm Temperature Thermowell



50mm Temperature Thermowell



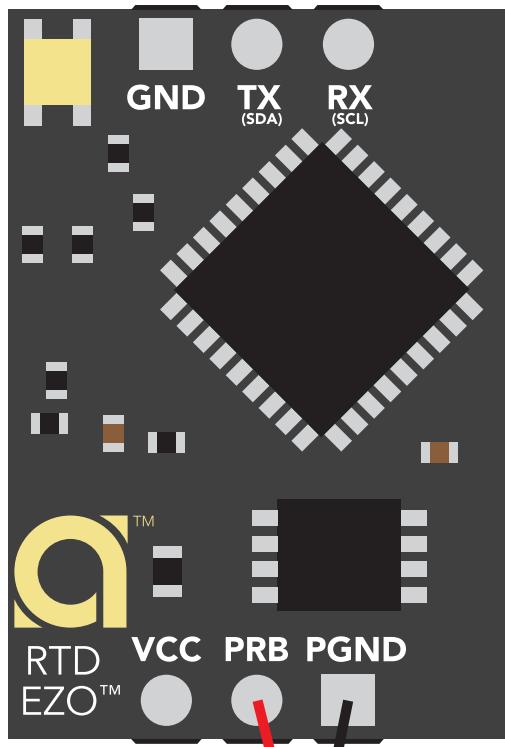
30mm Temperature Thermowell

# Using other brand PT-100/PT-1000

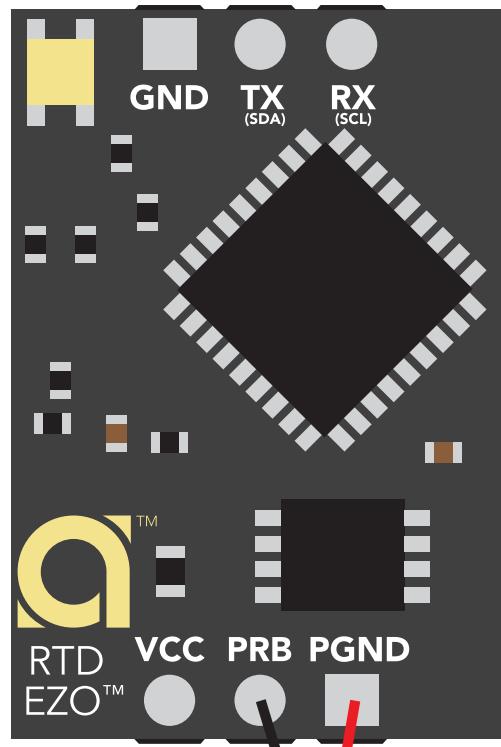
The EZO™ RTD Temperature circuit will auto-detect if the connected probe is PT-100 or PT-1000.

Probe class	Accuracy
AA	+/- (0.1 + 0.0017 × °C)
A	+/- (0.15 + 0.002 × °C)
B	+/- (0.3 + 0.005 × °C)
C	+/- (0.6 + 0.01 × °C)

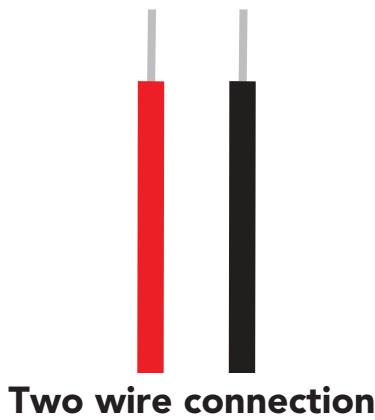
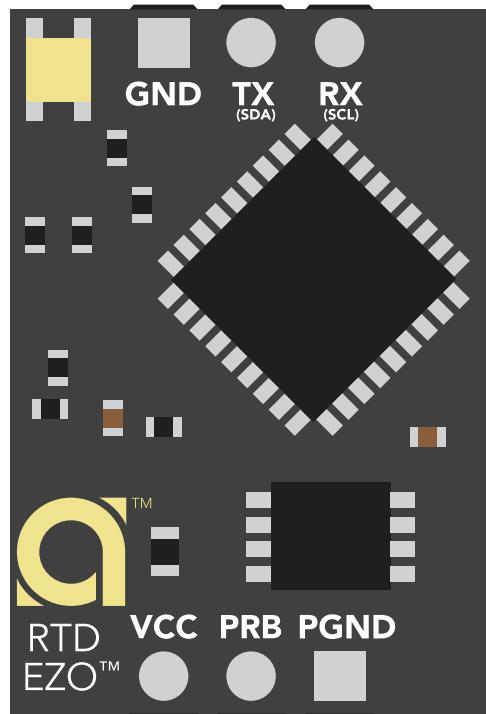
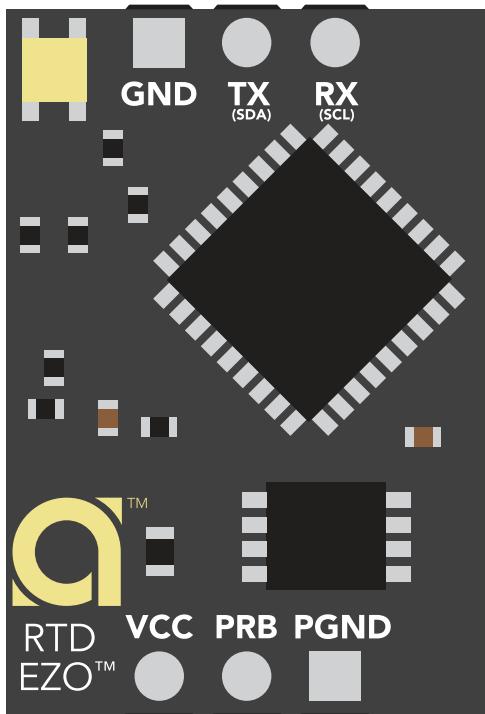
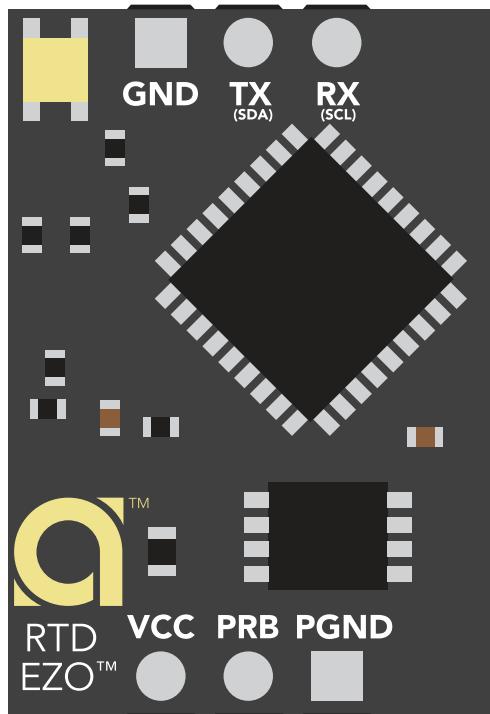
It makes no difference which lead of the temperature probe is connected to the two probe pins.



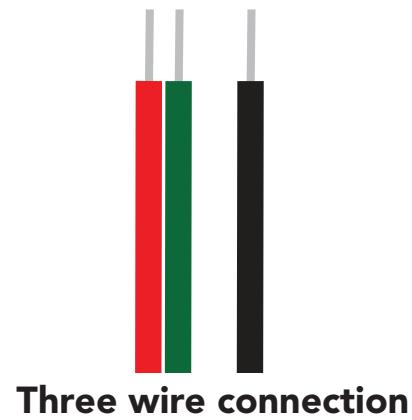
**BOTH ARE CORRECT**



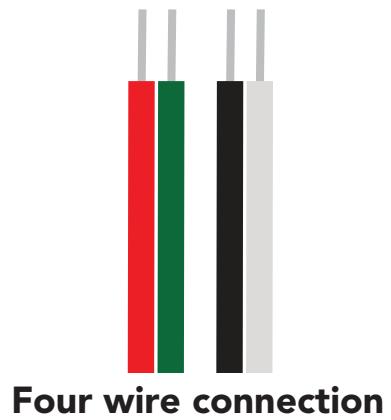
Any off the shelf PT-100 or PT-1000 temperature probe can be used with the Atlas Scientific EZO™ RTD Temperature circuit. The PT-100 or PT-1000 temperature probe can be a 2, 3 or 4 wire probe.



**Two wire connection**



**Three wire connection**



**Four wire connection**

# Operating principle

The Atlas Scientific EZO™ RTD Temperature circuit is a small footprint computer system that is specifically designed to be used in robotic applications where the embedded systems engineer requires accurate and precise measurements of temperature through a generic PT-100/PT-1000 temperature probe.

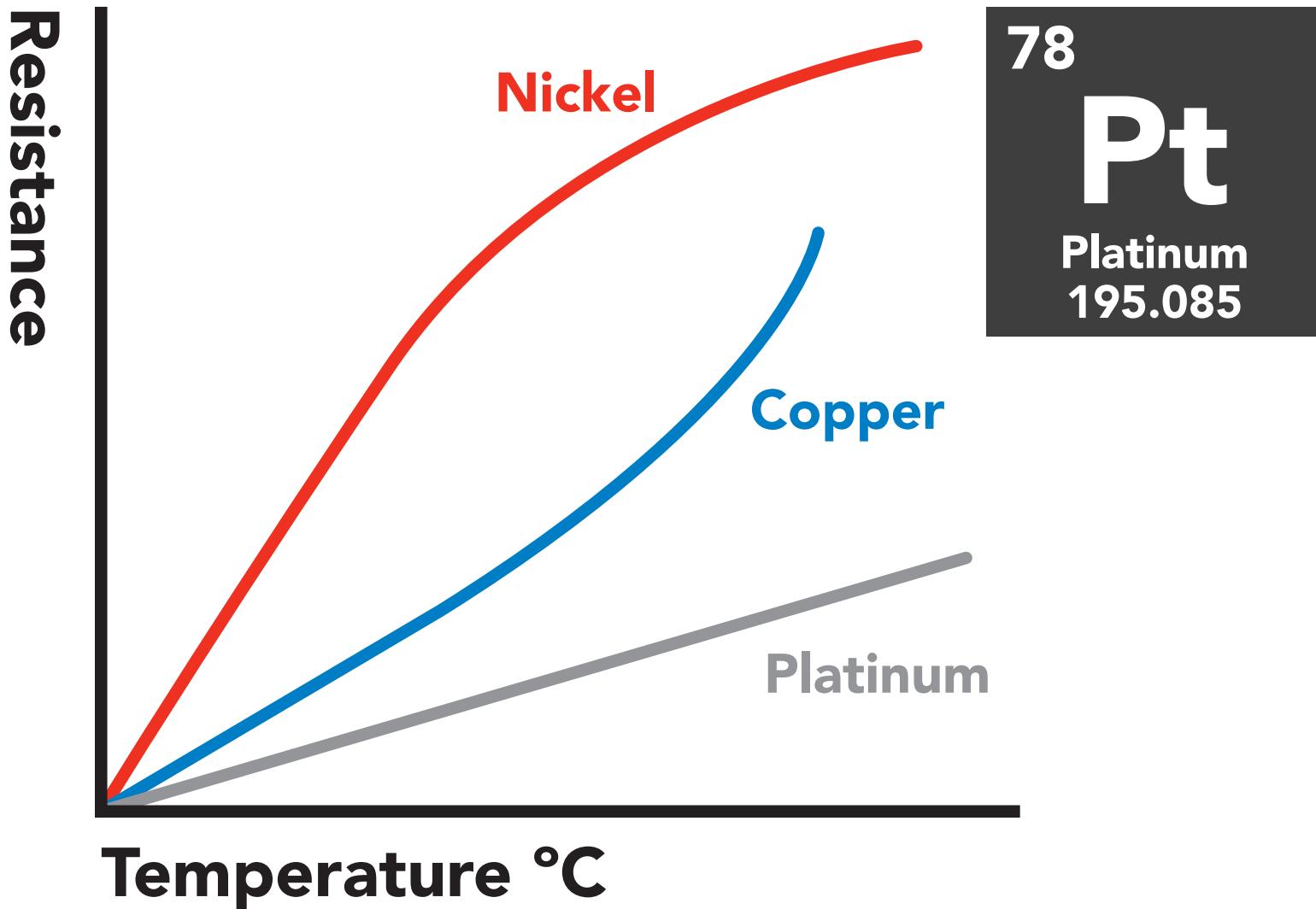
**RTD = Resistance Temperature Detector**

**PT = Platinum**

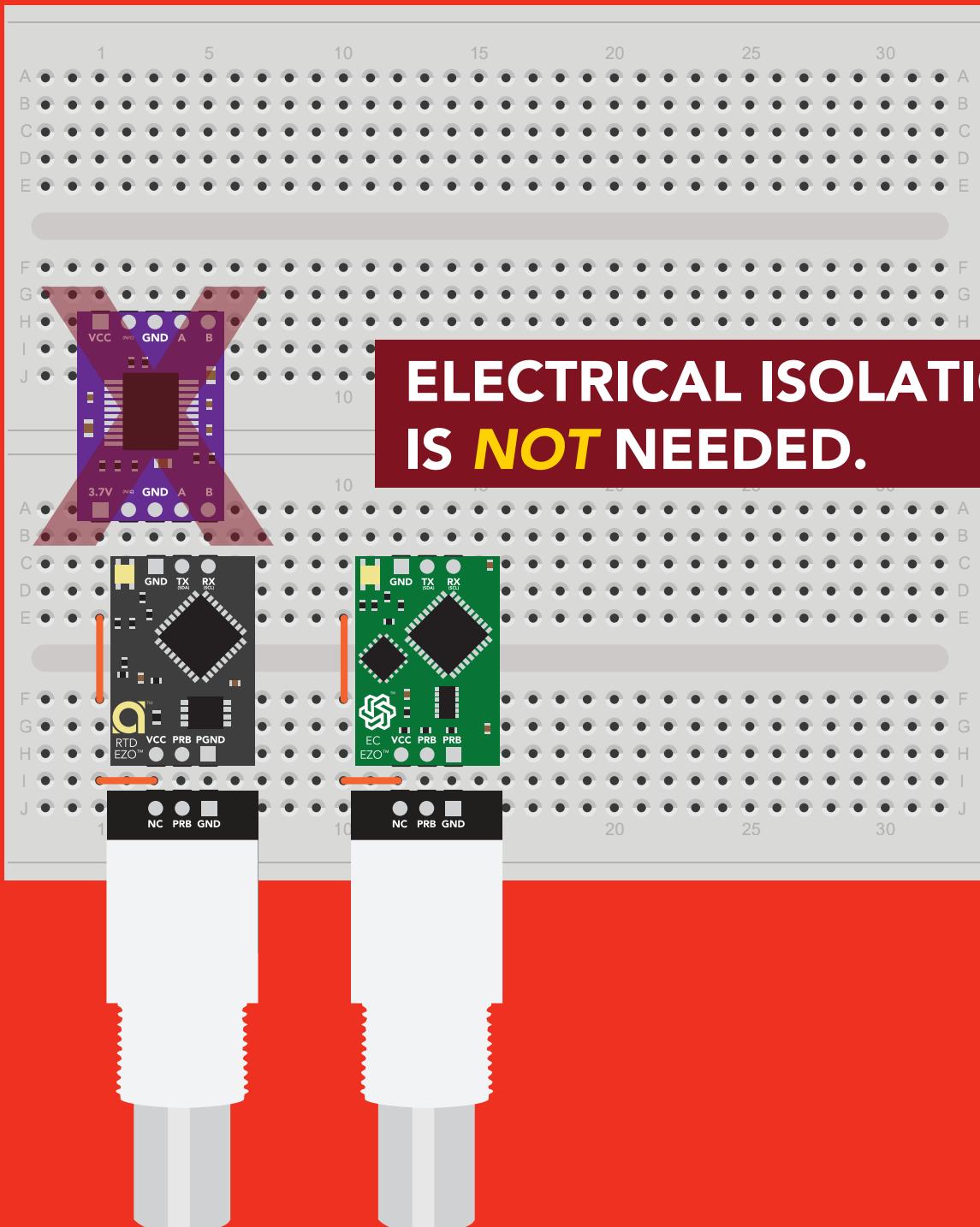
**PT-100 = 100 Ω at 0°C**

**PT-1000 = 1k Ω at 0°C**

Unlike any other material, platinums correlation between resistance and temperature seems to be woven into the fabric of the universe. It is for this reason, that the platinum RTD temperature sensor is the industrial standard for temperature measurement.

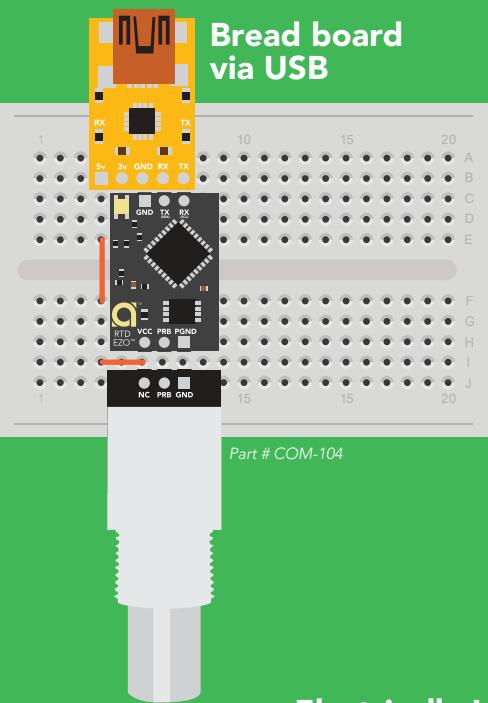
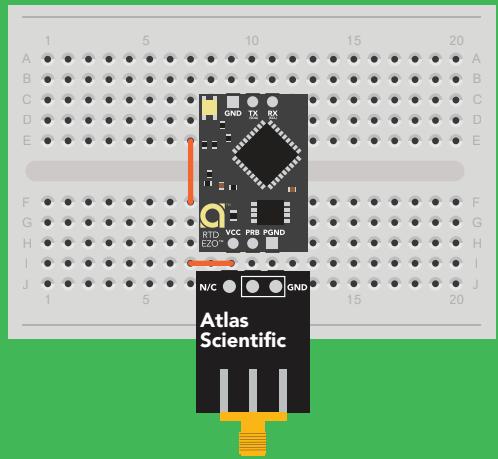


# Power and data isolation



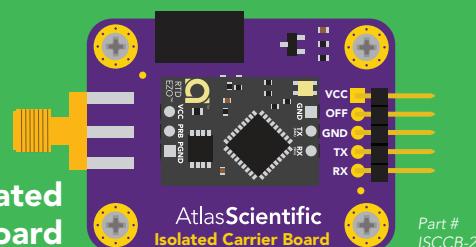
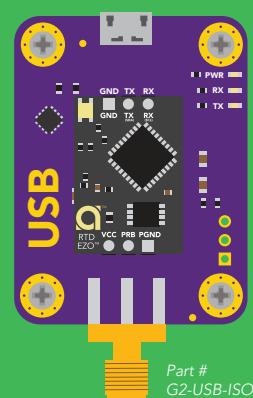
# ✓ Correct wiring

## Bread board



Carrier board

USB carrier board

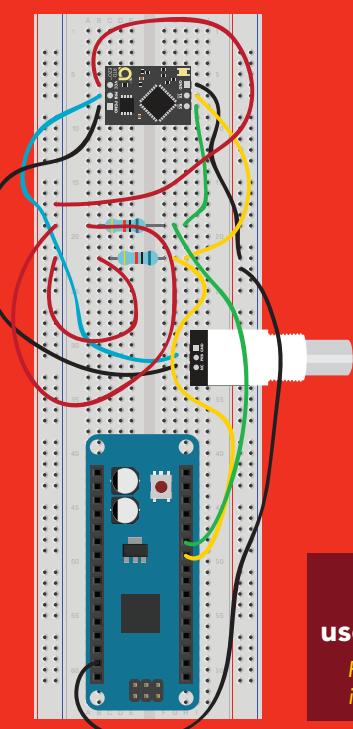


# ✗ Incorrect wiring

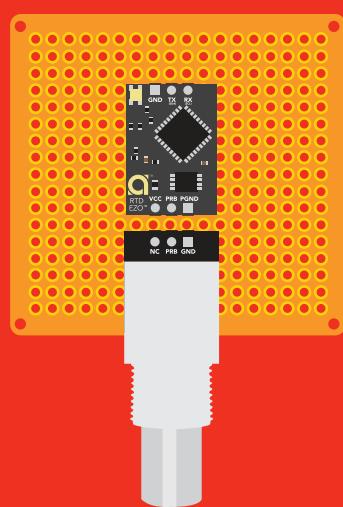
## Extended leads



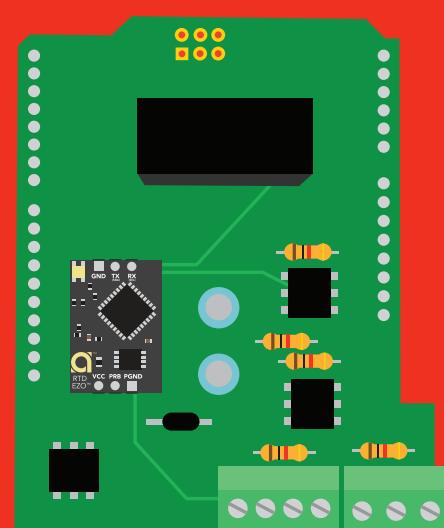
## Sloppy setup



## Perfboards or Protoboards



\*Embedded into your device



**NEVER**  
use Perfboards or Protoboards  
*Flux residue and shorting wires make it very hard to get accurate readings.*

\*Only after you are familiar with EZO™ circuits operation

# 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 I<sup>2</sup>C mode after calibration **will not** affect the stored calibration. If the device must be calibrated in I<sup>2</sup>C mode be sure to request readings continuously so you can see the output from the probe.

Calibration can be done at any value, a simple method is to calibrate the probe in boiling water.

**100 °C**

Atlas Scientific recommends calibration be done every three years.

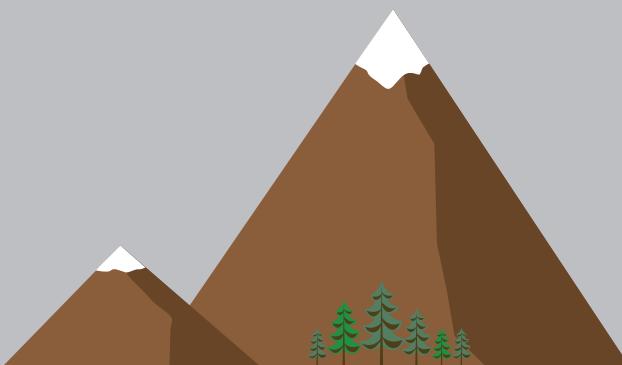
## Elevation and boiling point table

### Elevation in meters

305  
229  
152  
76  
0  
-76  
-152

### Boiling point

98.9 °C  
99.2 °C  
99.5 °C  
99.7 °C  
100 °C  
100.3 °C  
100.5 °C



## Use purified/distilled water

For accurate calibration using different temperature values, you must use a tool called a "dry block calibrator."

# On board data logger

- 50 readings
- Programmable storage interval

Minimum – 10 seconds

Maximum – 320,000 seconds

Temperature readings that are stored to the data logger will be retained even if the power is cut.



# Default state

# UART mode

Baud **9,600**

Temperature **°C**

Readings **continuous**

Speed **1 reading per second**

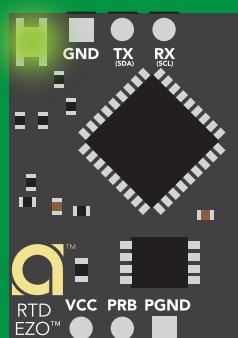
With probe **ttt.ttt**

Without probe **-1023.000**

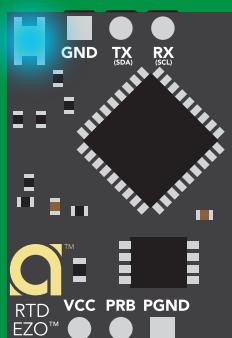
LED **on**



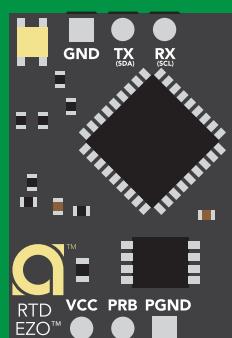
**1,000 ms**



**Green**  
**Standby**



**Cyan**  
**Taking reading**



**Transmitting**

 Available data protocols

**UART**

**Default**

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

 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  
Temperature scale  
Device name  
Enable/disable response codes  
Hardware switch to I<sup>2</sup>C mode  
LED control  
Protocol lock  
Software switch to I<sup>2</sup>C mode

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

Find  
Sleep mode

# UART mode

8 data bits      no parity  
1 stop bit      no flow control

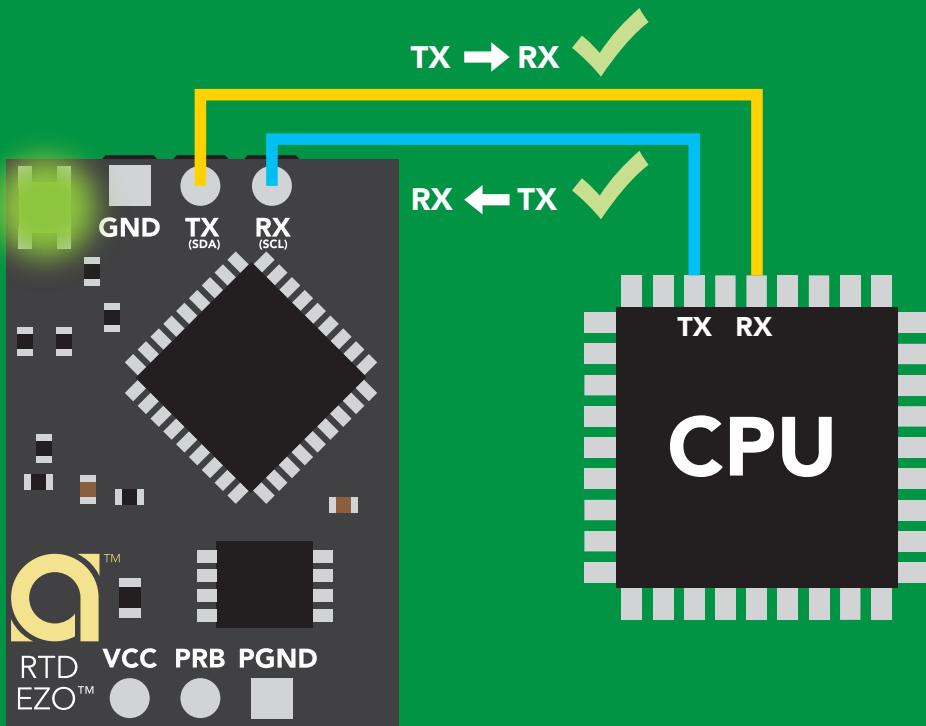
Baud 300  
1,200  
2,400  
**9,600 default**  
19,200  
38,400  
57,600  
115,200

**RX** Data in

**TX** Data out

**Vcc** 3.3V – 5.5V

0V  0V



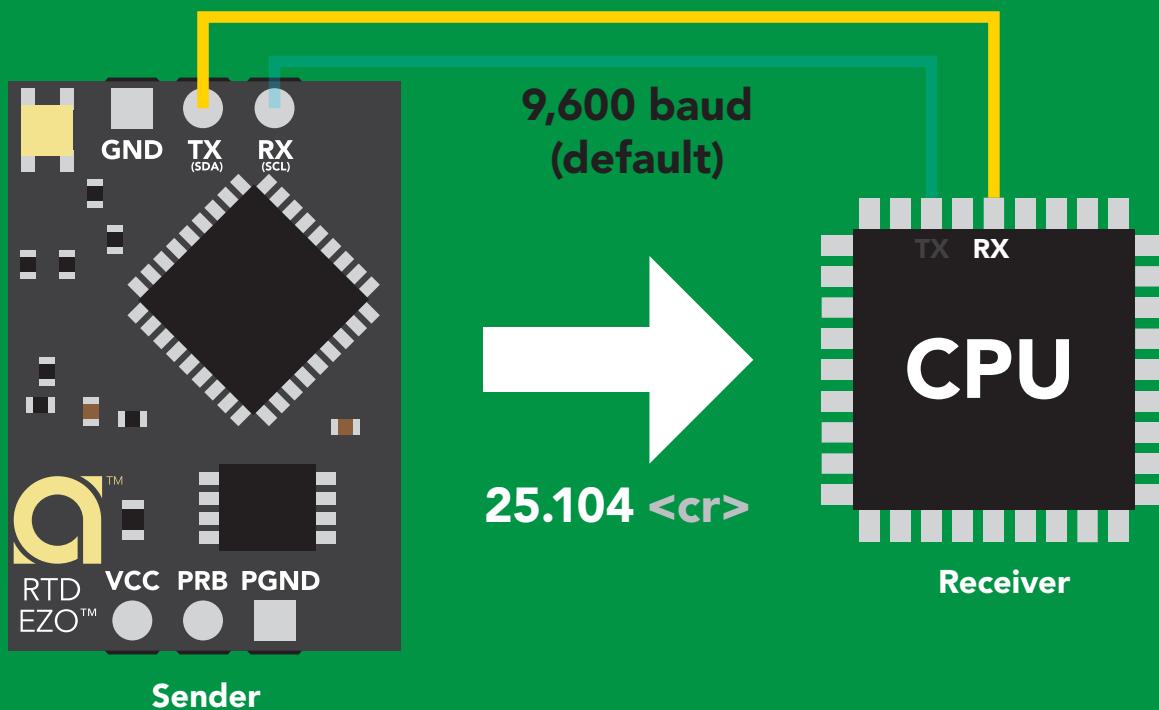
## Data format

**Reading** temperature  
**Units** °C, °K, or °F  
**Encoding** ASCII  
**Format** string  
**Terminator** carriage return

**Data type** floating point  
**Decimal places** 3  
**Smallest string** 4 characters  
**Largest string** 40 characters

# Receiving data from device

2 parts

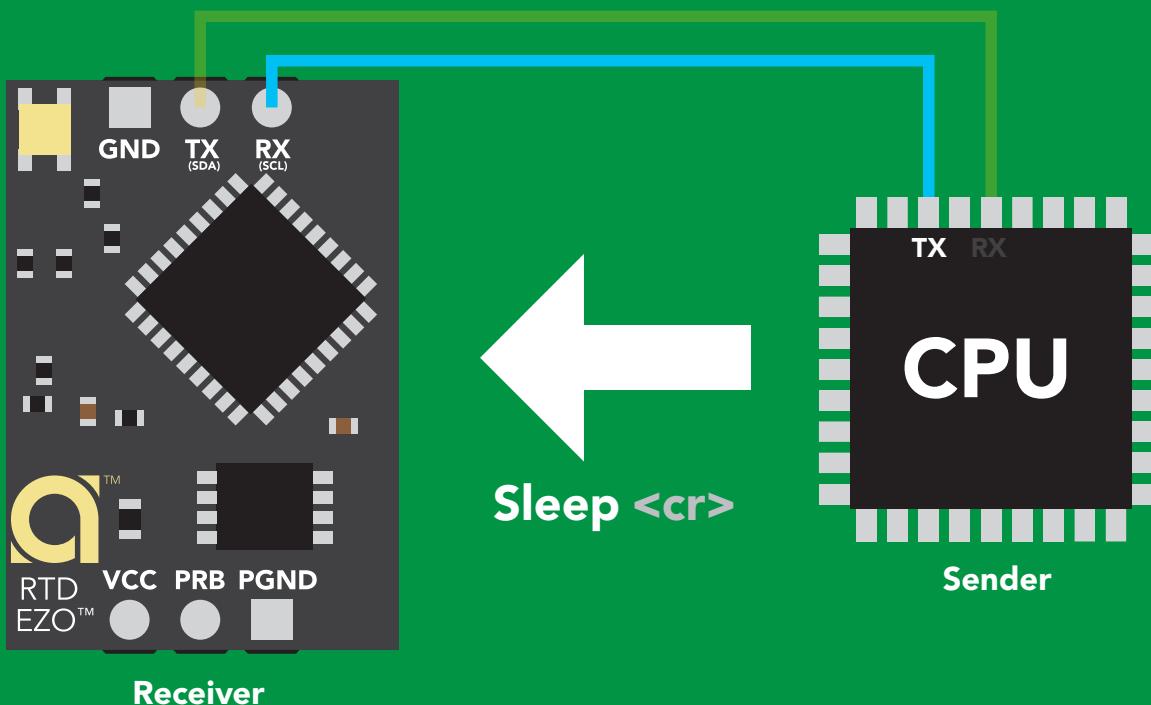


## Advanced

ASCII:	2	5	.	1	0	4	<cr>
Hex:	32	35	2E	31	30	34	0D
Dec:	50	53	46	49	48	52	13

# Sending commands to device

2 parts



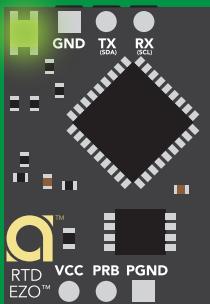
## Advanced

ASCII: S I e e p <cr>

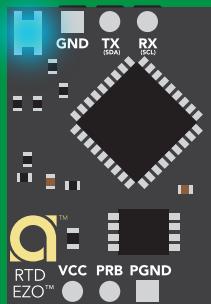
Hex: 53 6C 65 65 70 0D

Dec: 83 108 101 101 112 13

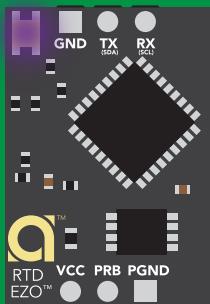
# LED color definition



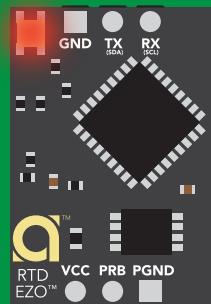
**Green**  
UART standby



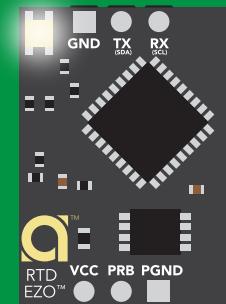
**Cyan**  
Taking reading



**Purple**  
Changing baud rate



**Red**  
Command  
not understood



**White**  
Find

<b>5V</b>	LED ON <b>+0.4 mA</b>
<b>3.3V</b>	<b>+0.2 mA</b>

# UART mode

## command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function	Default state
Baud	change baud rate	pg. 38 9,600
C	enable/disable continuous reading	pg. 24 enabled
Cal	performs calibration	pg. 26 n/a
D	enable/disable data logger	pg. 30 disabled
Export	export calibration	pg. 27 n/a
Factory	enable factory reset	pg. 40 n/a
Find	finds device with blinking white LED	pg. 23 n/a
i	device information	pg. 34 n/a
I2C	change to I <sup>2</sup> C mode	pg. 41 not set
Import	import calibration	pg. 28 n/a
L	enable/disable LED	pg. 22 enabled
M	memory recall/clear	pg. 31 n/a
Name	set/show name of device	pg. 33 not set
Plock	enable/disable protocol lock	pg. 39 disabled
R	returns a single reading	pg. 25 n/a
S	temperature scale (°C, °K, °F)	pg. 29 celsius
Sleep	enter sleep mode/low power	pg. 37 n/a
Status	retrieve status information	pg. 36 n/a
*OK	enable/disable response codes	pg. 35 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>

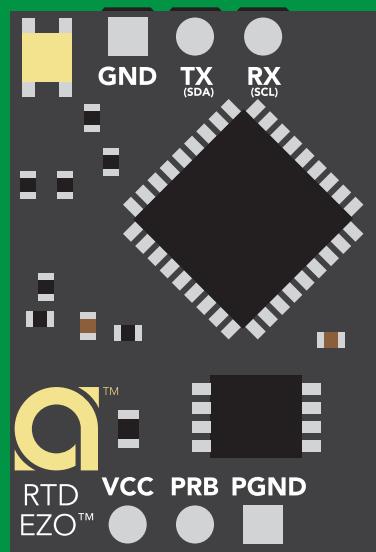
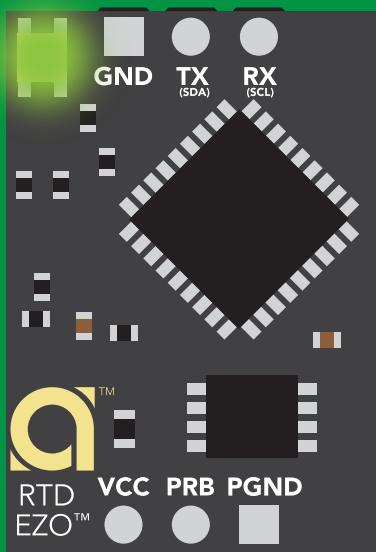
\*OK <cr>

L,? <cr>

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

\*OK <cr>

### Response



L,1

L,0

# Find

## Command syntax

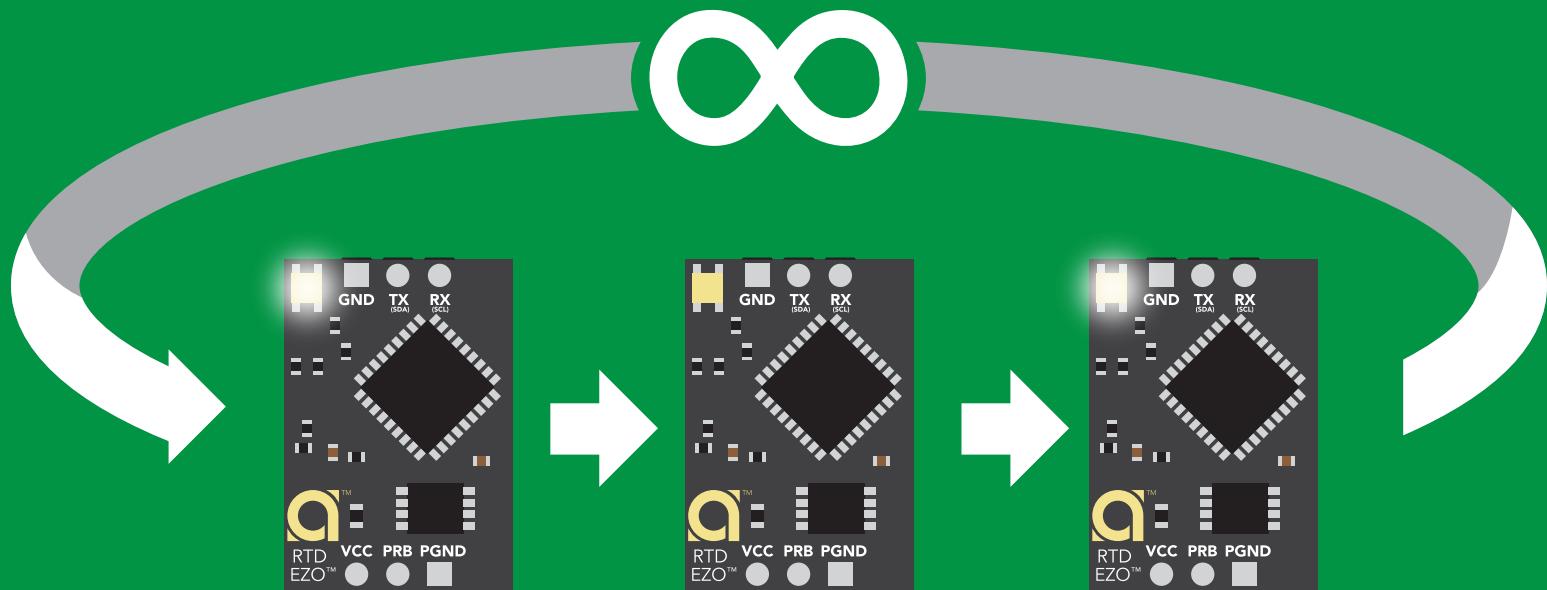
This command will disable continuous mode  
Send any character or command to terminate find.

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>  
°C (1 sec) <cr>  
°C (2 sec) <cr>  
°C (n sec) <cr>

C,30 <cr>

\*OK <cr>  
°C (30 sec) <cr>  
°C (60 sec) <cr>  
°C (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

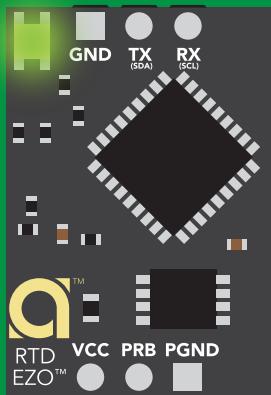
R <cr> takes single reading

### Example Response

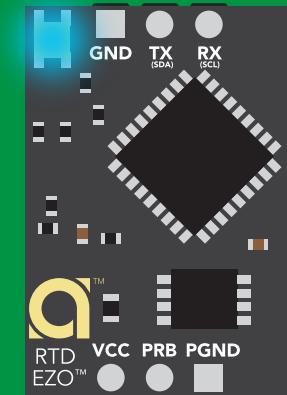
R <cr>

25.104 <cr>

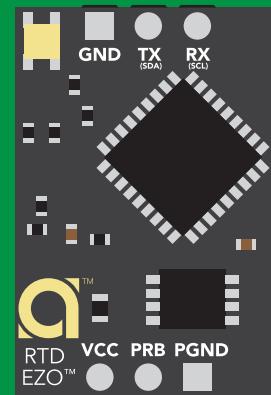
\*OK <cr>



Green  
Standby



Cyan  
Taking reading



Transmitting



# Calibration

## Command syntax

The EZO™ RTD circuit uses single point calibration.

**Cal,t <cr>** t = any temperature

**Cal,clear <cr>** delete calibration data

**Cal,? <cr>** device calibrated?

## Example

**Cal,100.00 <cr>**

**\*OK <cr>**

**Cal,clear <cr>**

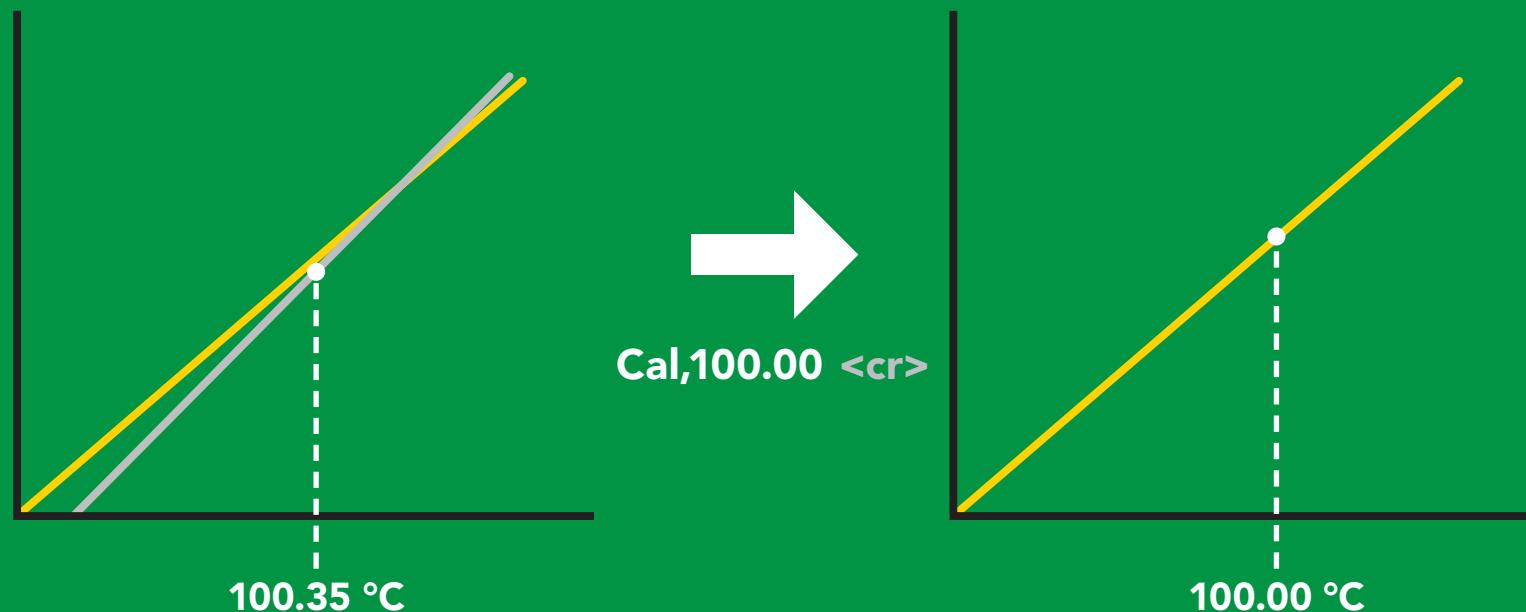
**\*OK <cr>**

**Cal,? <cr>**

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

**\*OK <cr>**

## Response



# Export calibration

## Command syntax

Export: Use this command to download calibration settings

**Export,? <cr>** calibration string info

**Export <cr>** export calibration string from calibrated device

## 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>

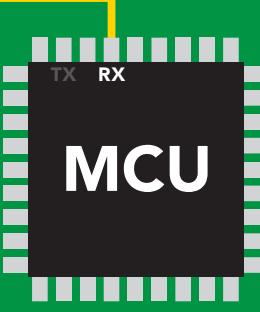
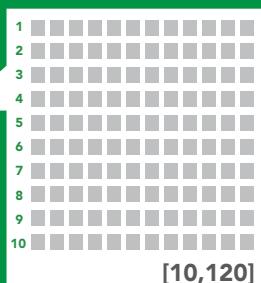
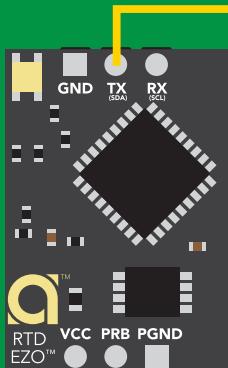
6F 6C 20 67 75 79 <cr> (10 of 10)

Export <cr>

\*DONE

Disabling \*OK simplifies this process

Export <cr>



# Import calibration

## Command syntax

Import: Use this command to upload calibration settings to one or more devices.

**Import,n <cr> import calibration string to new device**

## Example

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

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

⋮

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

## Response

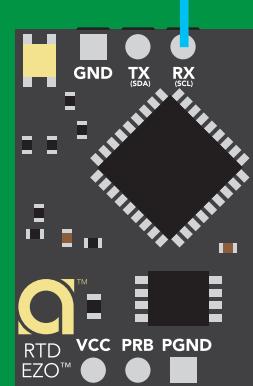
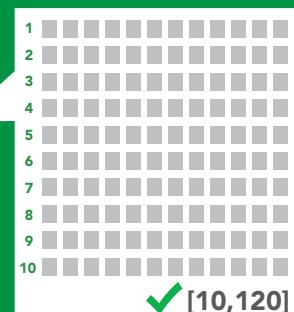
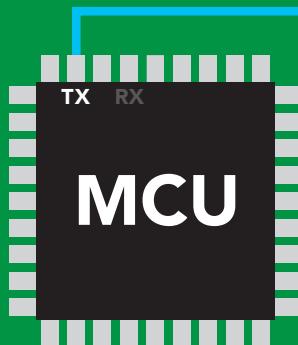
\*OK <cr>

\*OK <cr>

⋮

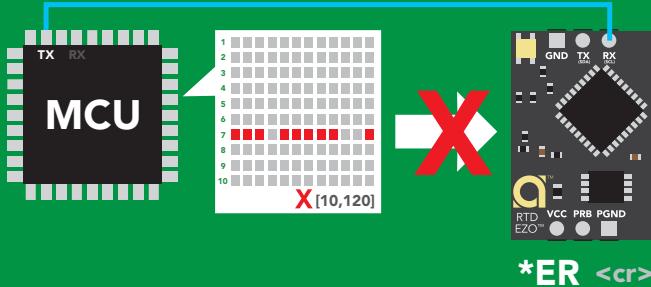
\*OK <cr>

**Import,n <cr>**



\*OK <cr>

system will reboot



\* If one of the imported strings is not correctly entered, the device will not accept the import, respond with \*ER and reboot.

# Temperature scale (°C, °K, °F)

## Command syntax

**S,c <cr> celsius** **default**  
**S,k <cr> kelvin**  
**S,f <cr> fahrenheit**  
**S,? <cr> temperature scale?**

## Example

**S,c <cr>**

**\*OK <cr>**

**S,k <cr>**

**\*OK <cr>**

**S,f <cr>**

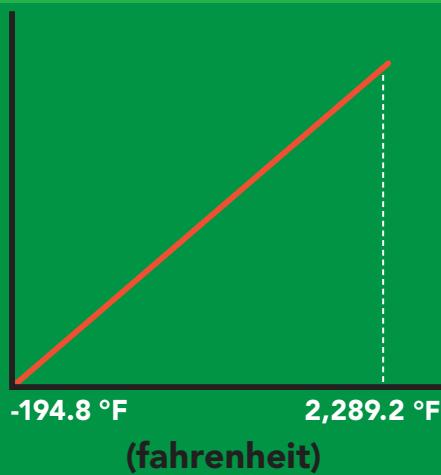
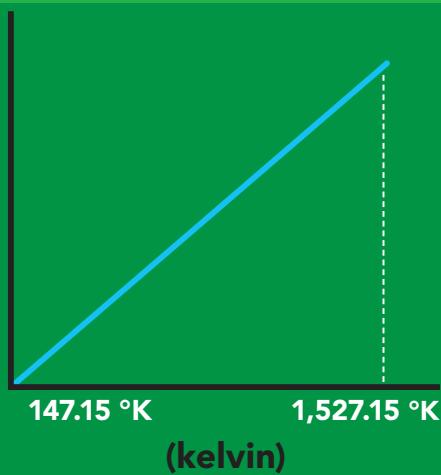
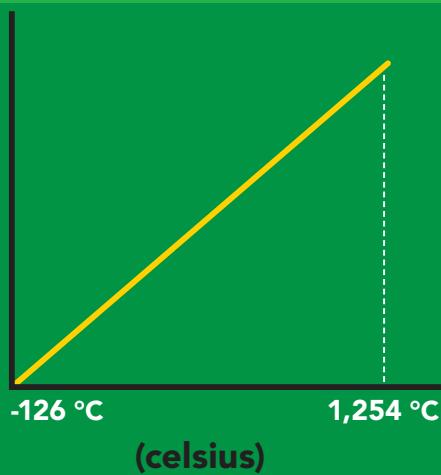
**\*OK <cr>**

**S,? <cr>**

**?S,c <cr> or ?S,k <cr> or ?S,f <cr>**

**\*OK <cr>**

## Response



# Enable/disable data logger

## Command syntax

The time period (n) is in 10 second intervals and can be any value from 1 to 32,000.

D,n <cr> n = (n x 10 seconds)

D,0 <cr> disable default

D,? <cr> data logger storage interval?

## Example

D,6 <cr>

D,0 <cr>

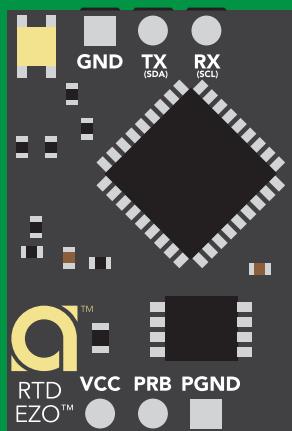
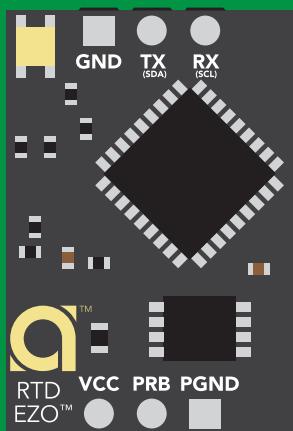
D,? <cr>

## Response

\*OK <cr>

\*OK <cr>

?D,6 <cr>  
\*OK <cr>



D,6



60 seconds

\* <cr>

\* indicates reading has been logged

# Memory recall

# Command syntax

**Disable data logger to recall memory.**

**M <cr> recall 1 sequential stored reading**

**M,all <cr> recall all readings in a CSV string**

**M,? <cr> display memory location of last stored reading**

# Example

# Response

M <cr>

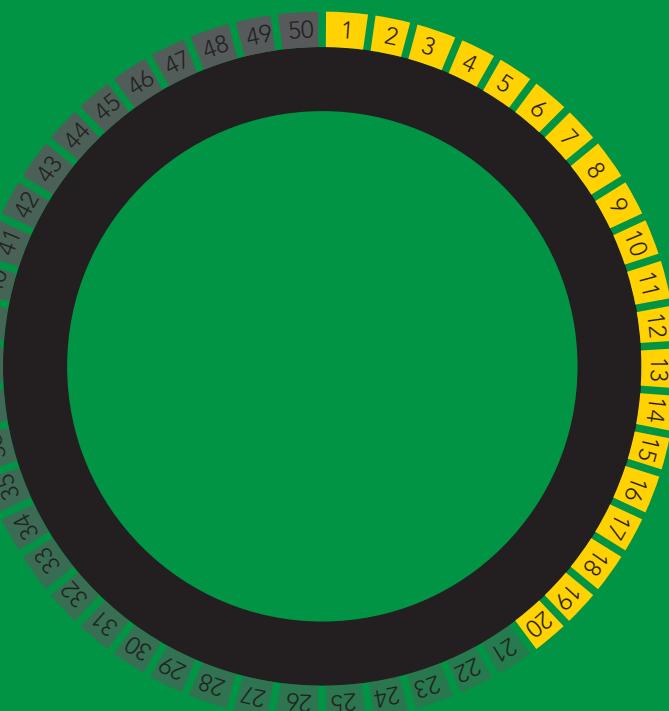
**1,100.00 <cr> 2,104.00 <cr> \*OK <cr>**

M,all <cr>

**100.00,104.00,108.00,112.00 <cr>**

M,? <cr>

?M,4 <cr>  
\*OK <cr>



# Memory clear

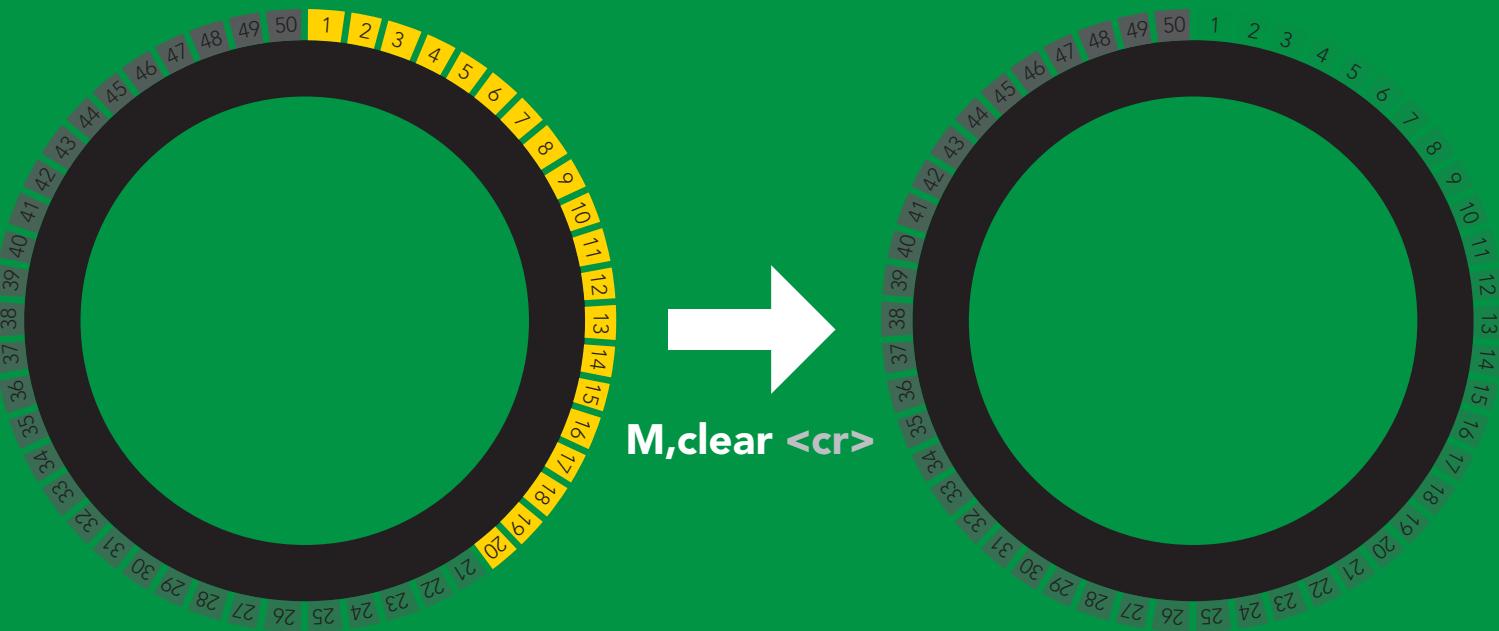
## Command syntax

**M,clear <cr>** clear all stored memory

### Example Response

**M,clear <cr>**

**\*OK <cr>**



# Naming device

## Command syntax

Do not use spaces in the name

**Name,n <cr> set name**

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

**Name, <cr> clears name**

Up to 16 ASCII characters

**Name,? <cr> show name**

## Example

Name, <cr>

Name,zzt <cr>

Name,? <cr>

## Response

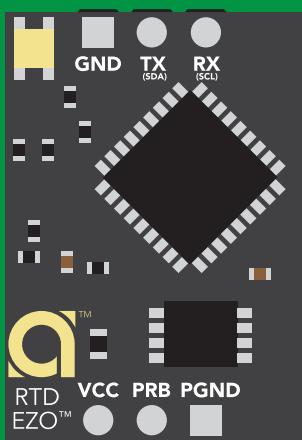
\*OK <cr> name has been cleared

\*OK <cr>

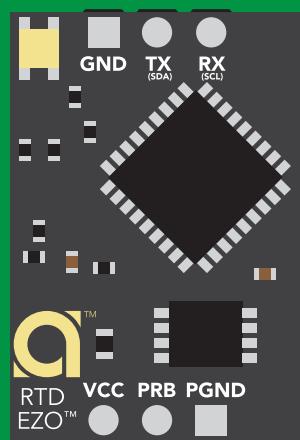
?Name,zzt <cr>

\*OK <cr>

Name,zzt



Name,?



\*OK <cr>

?Name,zzt <cr>

\*OK <cr>

# Device information

## Command syntax

**i <cr> device information**

### Example      Response

i <cr>

?i,RTD,2.01 <cr>  
\*OK <cr>

### Response breakdown

?i, RTD, 2.01  
↑            ↑  
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>

**25.104 <cr>**

**\*OK <cr>**

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

**no response, \*OK disabled**

R <cr>

**25.104 <cr> \*OK disabled**

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

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

## Other response codes

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

These response codes  
cannot be disabled

# Reading device status

## Command syntax

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

### Example      Response

Status <cr>

?Status,P,5.038 <cr>

\*OK <cr>

### Response breakdown

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

#### Restart codes

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

# Sleep mode/low power

## Command syntax

Send any character or command to awaken device.

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

### Example

**Sleep <cr>**

### Response

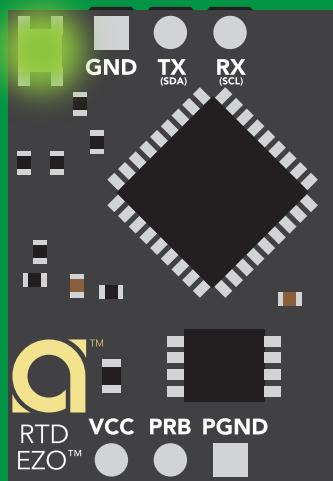
**\*OK <cr>**  
**\*SL <cr>**

### Any command

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

	STANDBY	SLEEP
<b>5V</b>	<b>15.40 mA</b>	<b>0.4 mA</b>

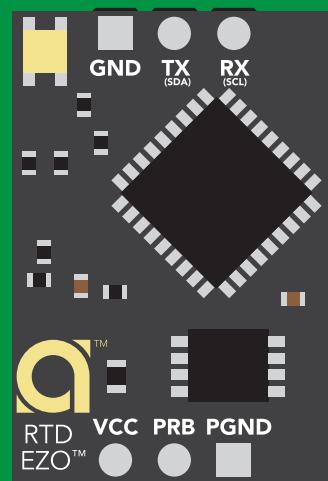
<b>3.3V</b>	<b>13.80 mA</b>	<b>0.09 mA</b>
-------------	-----------------	----------------



**Standby  
15.40 mA**



**Sleep <cr>**



**Sleep  
3.00 mA**

# Change baud rate

## Command syntax

Baud,n <cr> change baud rate

### Example

Baud,38400 <cr>

### Response

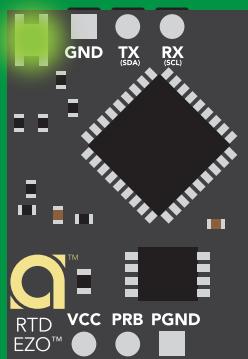
\*OK <cr>

Baud,? <cr>

?Baud,38400 <cr>

\*OK <cr>

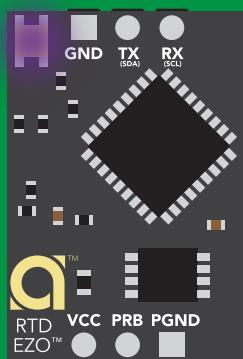
n = [ 300  
1200  
2400  
**9600 default**  
19200  
38400  
57600  
115200 ]



Standby



Baud,38400 <cr>

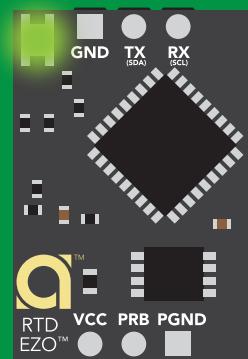


Changing  
baud rate

\*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>

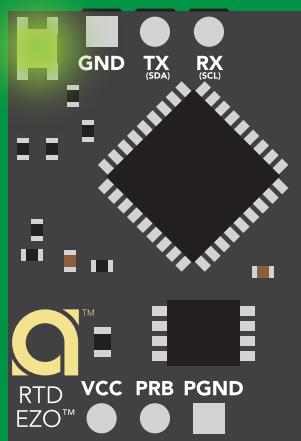
\*OK <cr>

Plock,? <cr>

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

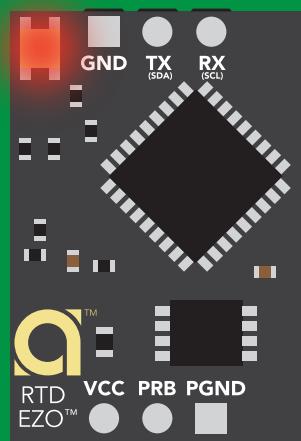
## Response

Plock,1



\*OK <cr>

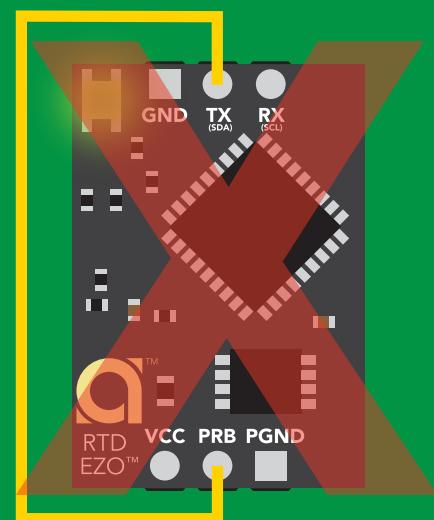
I2C,100



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

\*ER <cr>

Short



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

# Factory reset

## Command syntax

Clears calibration  
LED on  
"\*OK" enabled  
Clears data logger

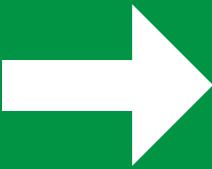
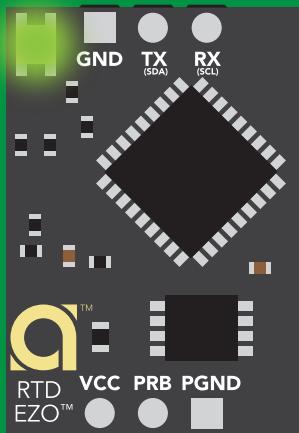
Factory <cr> enable factory reset

## Example Response

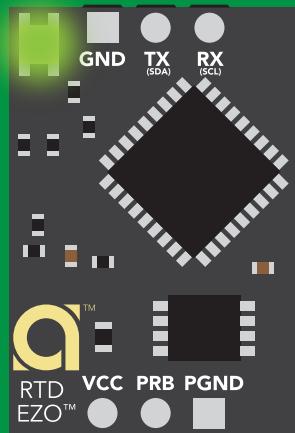
Factory <cr>

\*OK <cr>

Factory <cr>



(reboot)



\*OK <cr>

\*RS <cr>

\*RE <cr>

Baud rate will not change

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

## Command syntax

Default I<sup>2</sup>C address 102 (0x66)

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

n = any number 1 – 127

## Example      Response

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

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

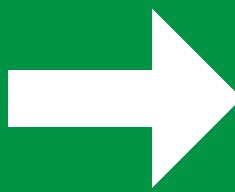
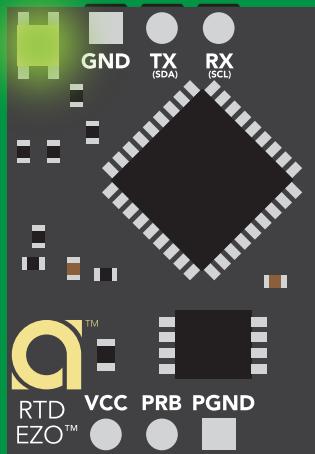
## Wrong example

I<sup>2</sup>C,139 <cr> n ≠ 127

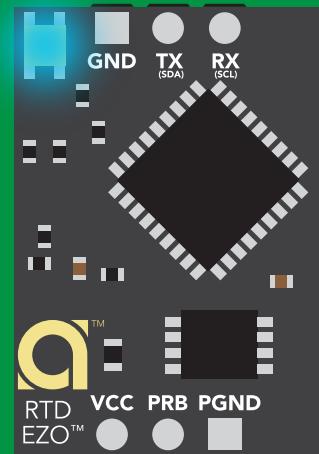
## Response

\*ER <cr>

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



(reboot)



Green  
\*OK <cr>

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

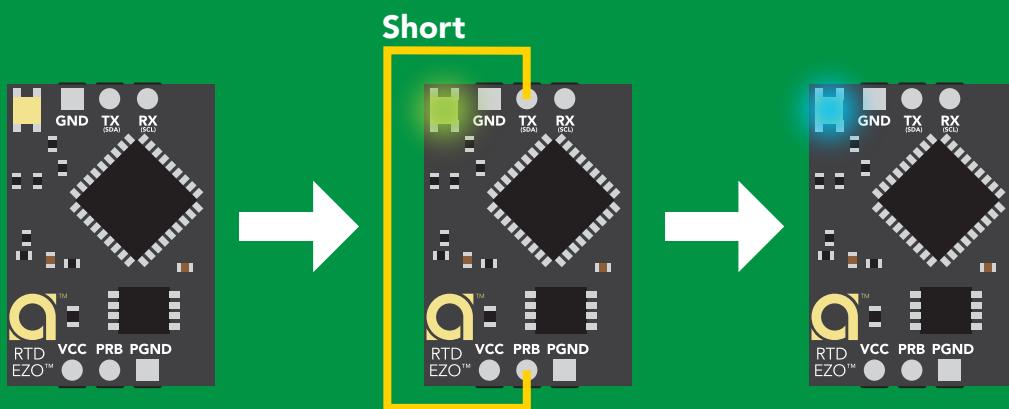
# Manual switching to I<sup>2</sup>C

- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to 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

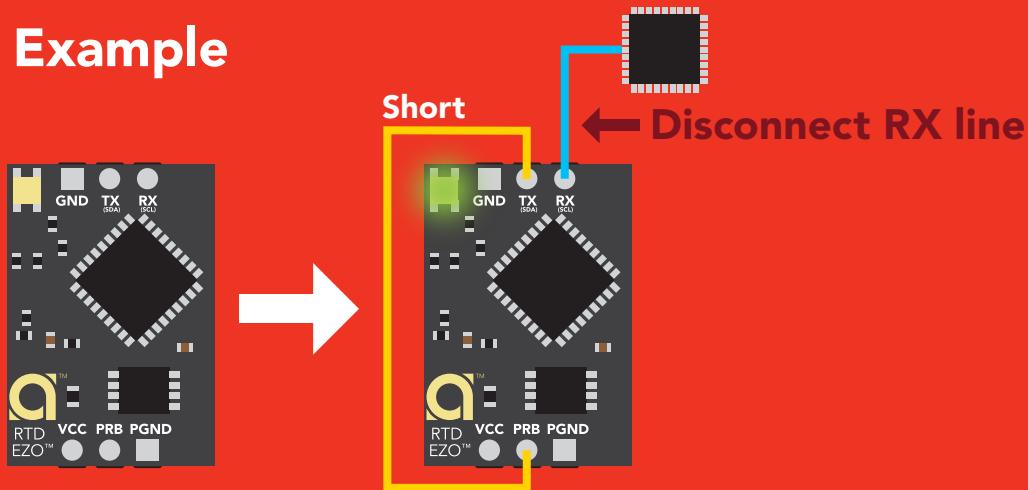
Connecting TX to PRB only works for the EZO-RTD™ and the EZO-FLO™ circuits

Manually switching to I<sup>2</sup>C will set the I<sup>2</sup>C address to 102 (0x66)

## Example



## Wrong Example



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

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

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

## Settings that are retained if power is cut

- Calibration
- Change I<sup>2</sup>C address
- Temperature scale
- Hardware switch to UART mode
- LED control
- Protocol lock
- Software switch to UART mode

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

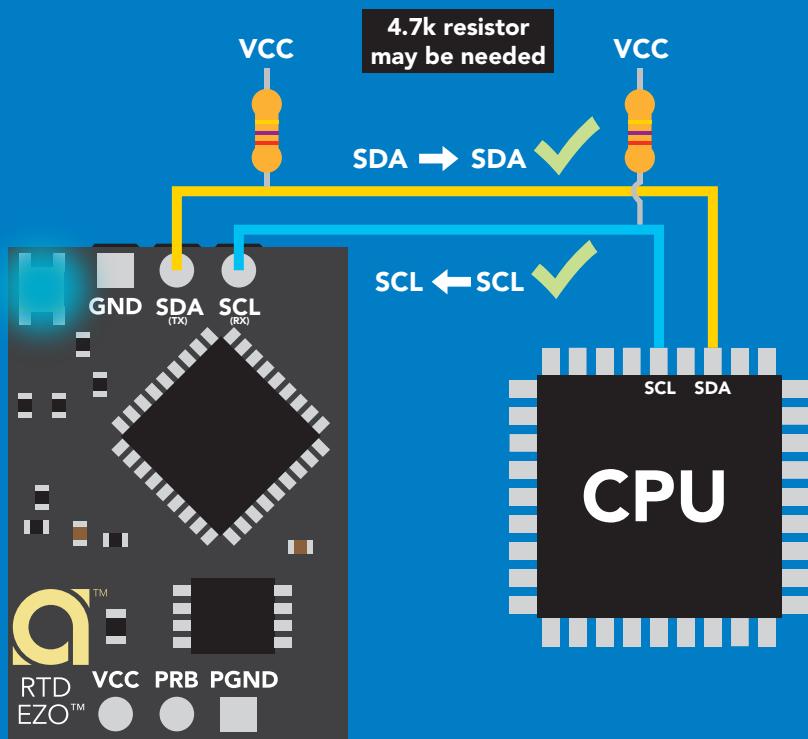
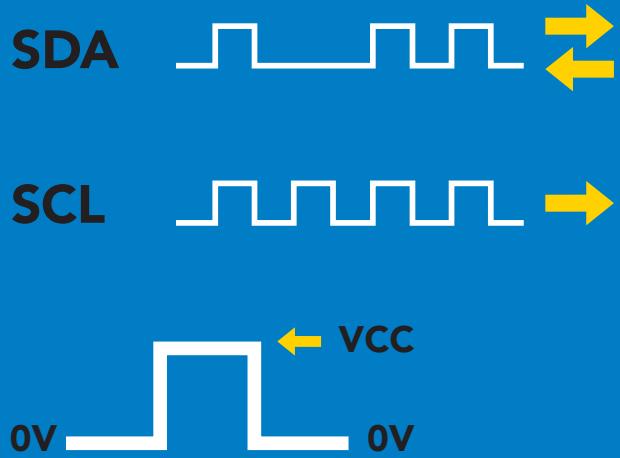
- Find
- Sleep mode

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

I<sup>2</sup>C address (0x01 – 0x7F)  
102 (0x66) default

V<sub>cc</sub> 3.3V – 5.5V

Clock speed 100 – 400 kHz



## Data format

Reading temperature  
Units °C, °K, or °F  
Encoding ASCII  
Format string

Data type floating point  
Decimal places 3  
Smallest string 4 characters  
Largest string 40 characters

# Sending commands to device

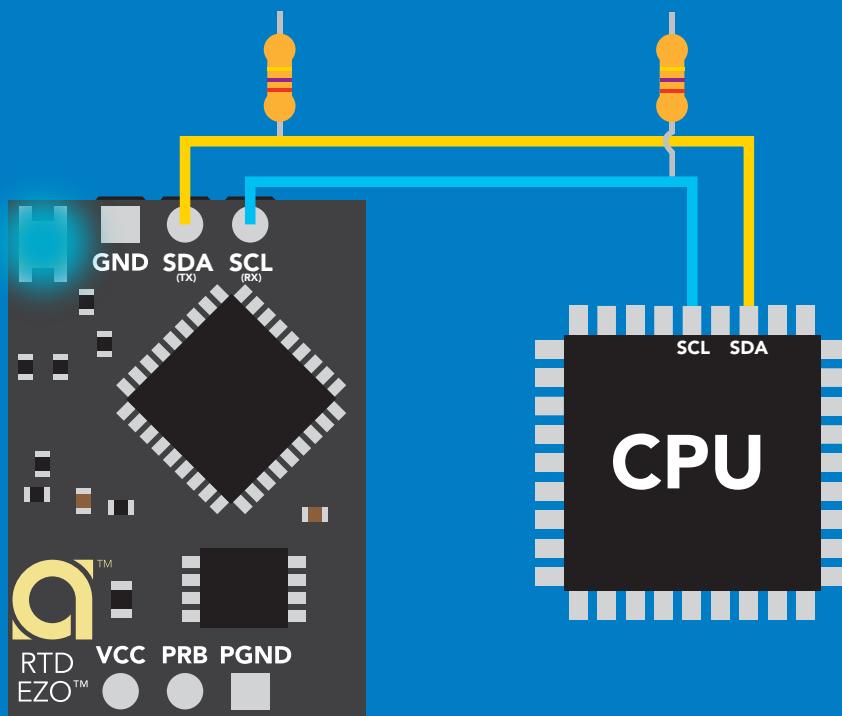
5 parts



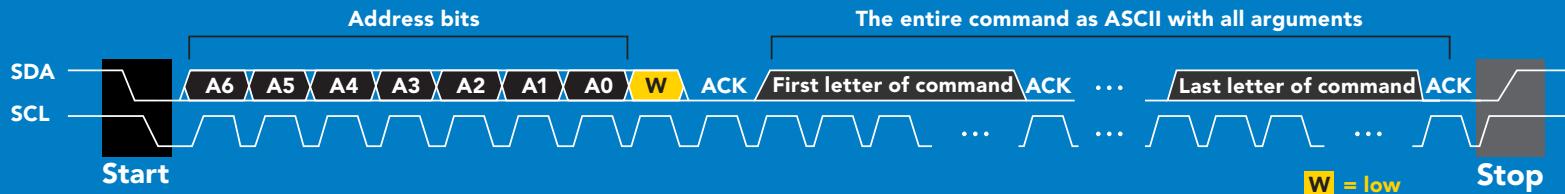
## Example

**Start**    **102 (0x66)**    **Write**    **Sleep**    **Stop**

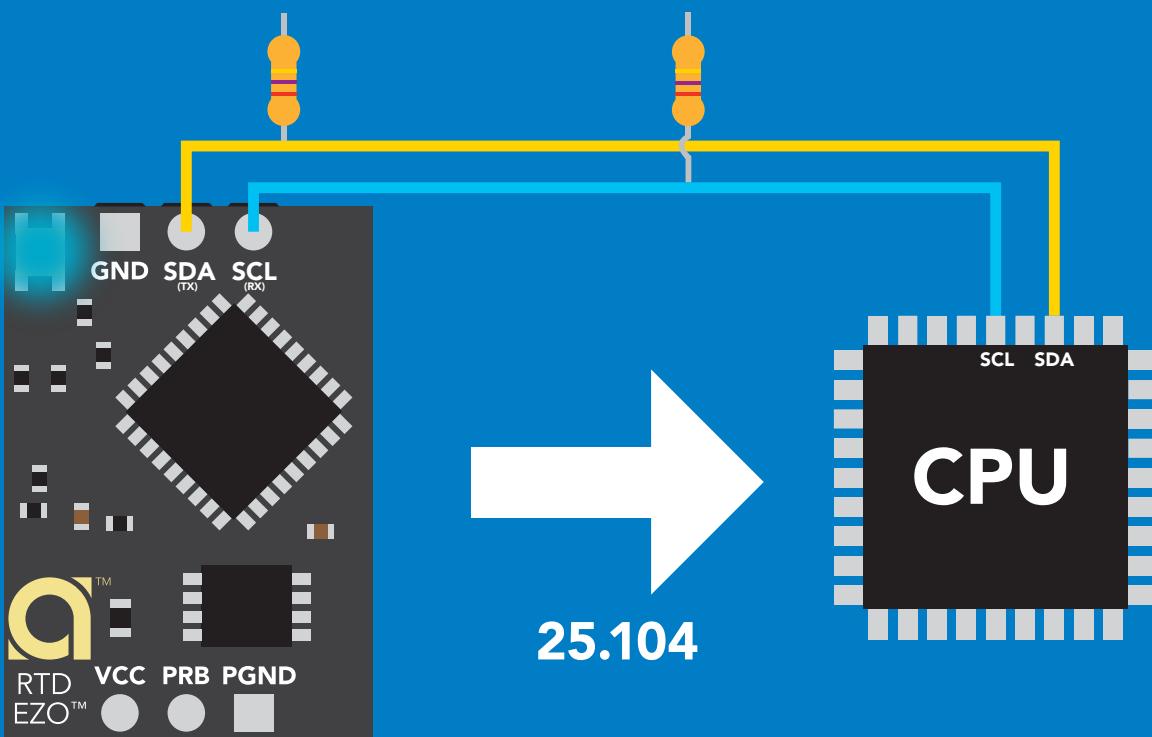
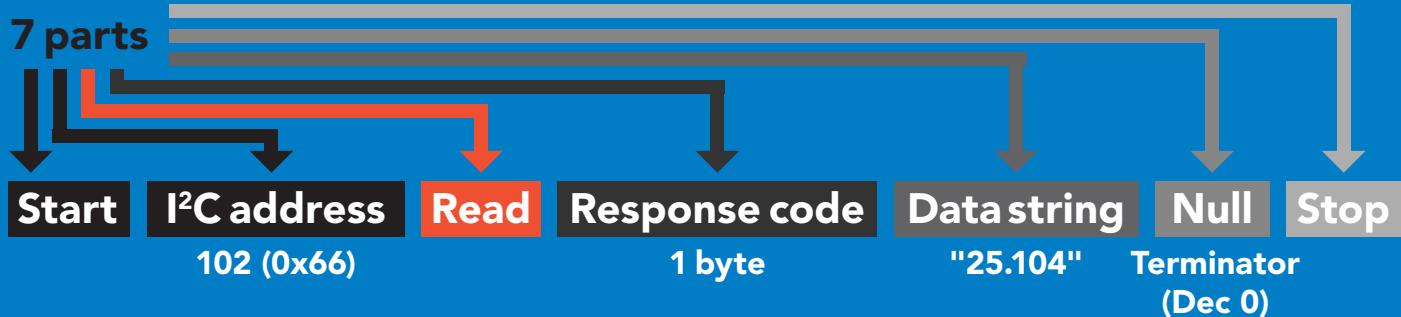
I<sup>2</sup>C address                      Command



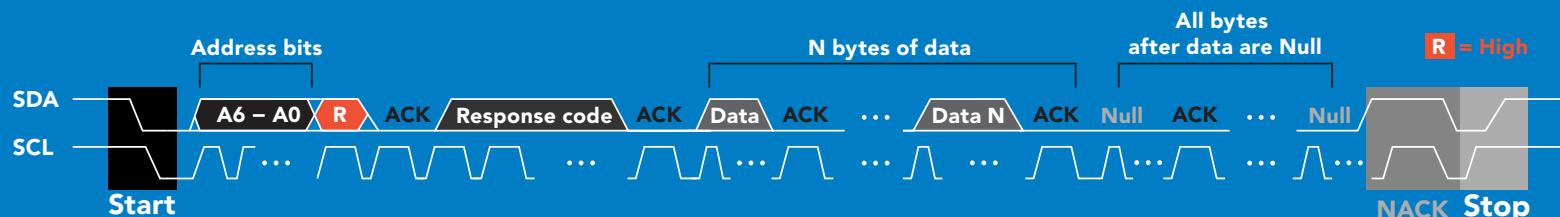
## Advanced



# Requesting data from device



## Advanced

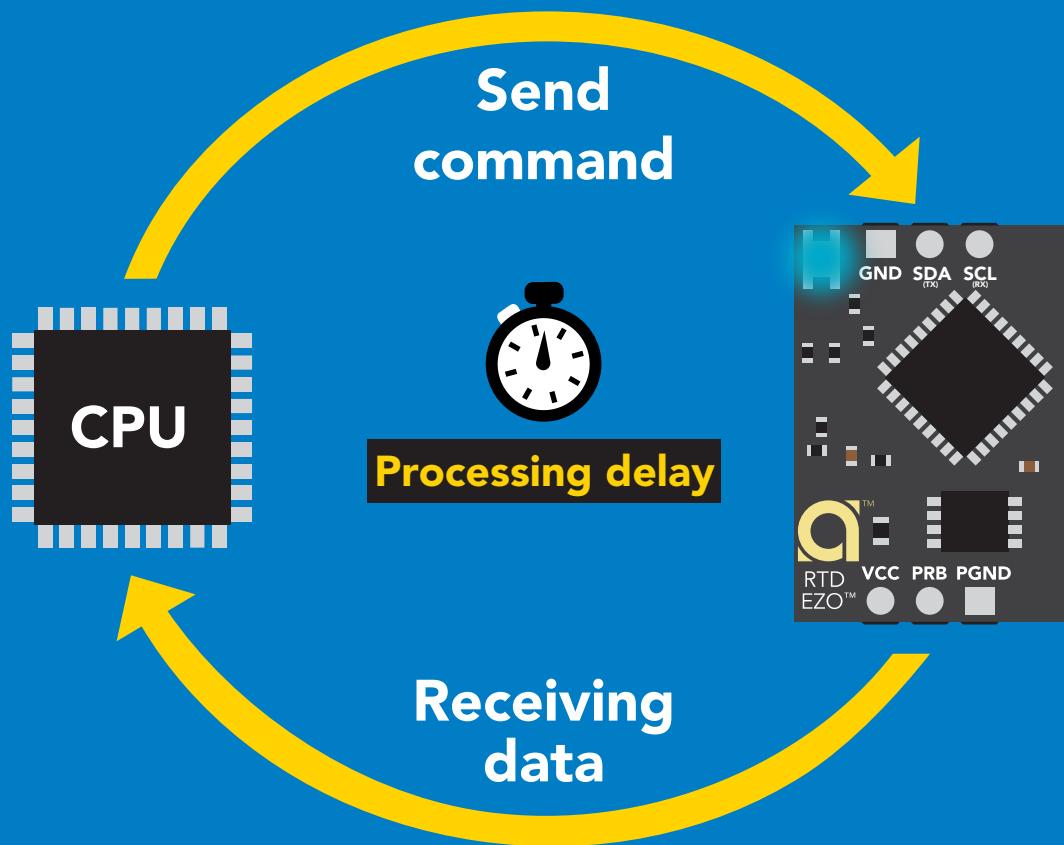


1 50 53 46 49 48 52 0 = 25.104  
 Dec ASCII Dec

# 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;
```

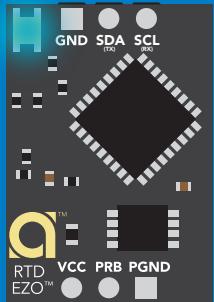
If there is no processing delay or the processing delay is too short, the response code will always be 254.

### Response codes

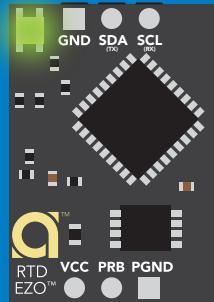
Single byte, not string

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

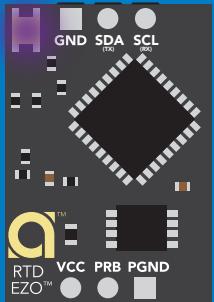
# LED color definition



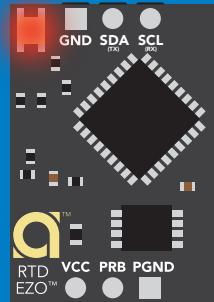
Blue



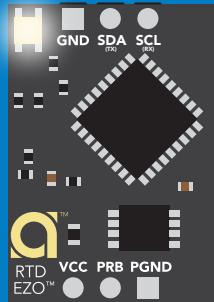
Green



Purple



Red



White

I<sup>2</sup>C standby

Taking reading

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

Command  
not understood

Find

5V

LED ON  
**+0.4 mA**

3.3V

**+0.2 mA**

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

## command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function	
Baud	switch back to UART mode	pg. 67
Cal	performs calibration	pg. 53
D	enable/disable data logger	pg. 57
Export	export calibration	pg. 54
Factory	enable factory reset	pg. 66
Find	finds devices with white blinking LED	pg. 51
i	device information	pg. 61
I2C	change I <sup>2</sup> C address	pg. 65
Import	import calibration	pg. 55
L	enable/disable LED	pg. 50
M	memory recall/clear	pg. 58
Name	set/show name of device	pg. 60
Plock	enable/disable protocol lock	pg. 64
R	returns a single reading	pg. 52
S	temperature scale (°C, °K, °F)	pg. 56
Sleep	enter sleep mode/low power	pg. 63
Status	retrieve status information	pg. 62

# 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      **0**  
Null

L,0

  
Wait 300ms

**1**  
Dec      **0**  
Null

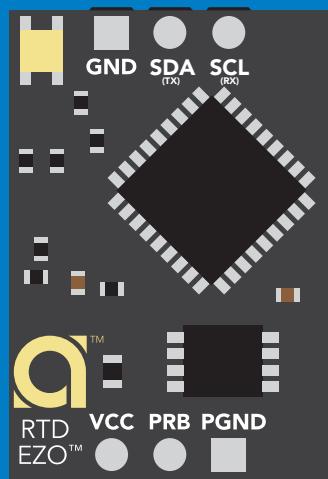
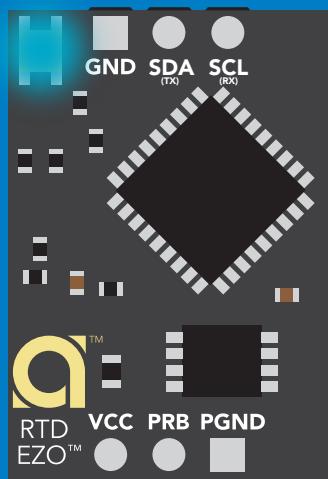
L,?

  
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.

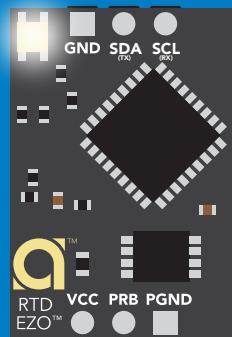
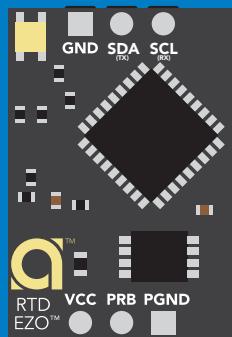
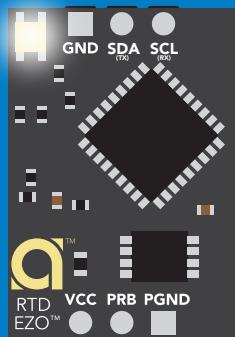
Find    LED rapidly blinks white, used to help find device

## Example      Response

Find <cr>

  
Wait 300ms

1    0  
Dec Null



# Taking reading

## Command syntax

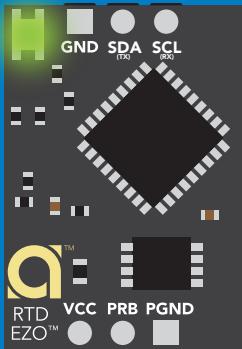
600ms  processing delay

R return 1 reading

## Example Response

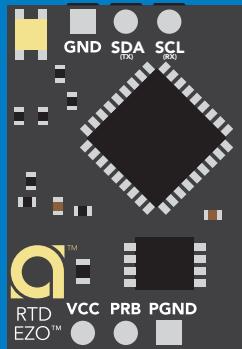
R

 Wait 600ms  
1 Dec 25.104 ASCII 0 Null

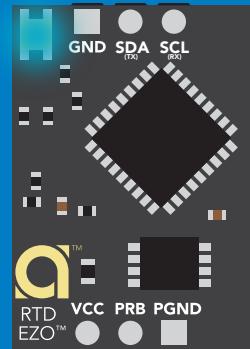


Green

Taking reading



Transmitting



Blue

Standby

# Calibration

## Command syntax

600ms  processing delay

**Cal,t**      t = any temperature

EZO™ RTD circuit uses  
single point calibration.

**Cal,clear** delete calibration data

**Cal,?** device calibrated?

## Example

## Response

**Cal,t**

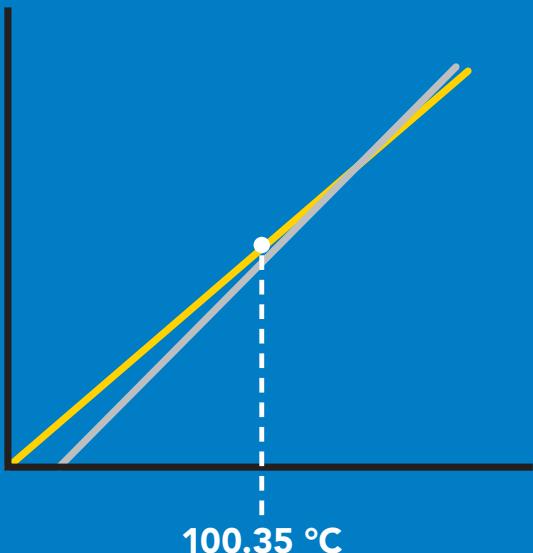
  
Wait 600ms      1 Dec 0 Null

**Cal,clear**

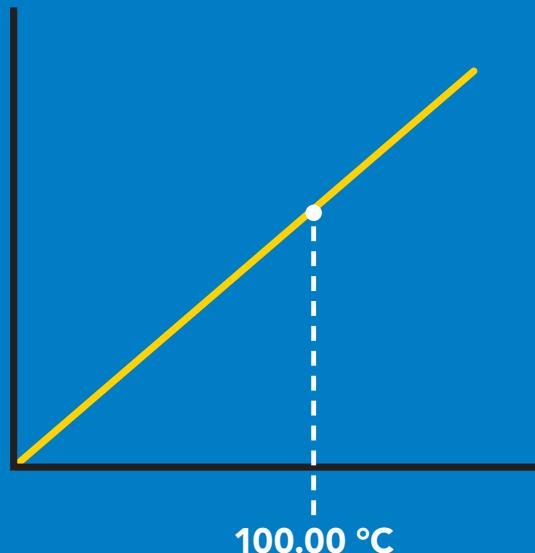
  
Wait 300ms      1 Dec 0 Null

**Cal,?**

  
Wait 300ms      1 Dec ?Cal,1 0 Null      or      1 Dec ?Cal,0 0 Null



Cal,100.00



# Export calibration

300ms  processing delay

## Command syntax

Export: Use this command to download calibration settings

**Export,?** calibration string info

**Export** export calibration string from calibrated device

## Example

**Export,?**



Wait 300ms

1

10,120

0

Dec

ASCII

Null

## Response breakdown

10, 120



# of strings to export

# of bytes to export

Export strings can be up to 12 characters long

**Export**



Wait 300ms

1

59 6F 75 20 61 72

0

Dec

ASCII

Null

(1 of 10)

**Export**



Wait 300ms

1

65 20 61 20 63 6F

0

Dec

ASCII

Null

(2 of 10)

**(7 more)**



**Export**



Wait 300ms

1

6F 6C 20 67 75 79

0

Dec

ASCII

Null

(10 of 10)

**Export**



Wait 300ms

1

\*DONE

0

Dec

ASCII

Null

# Import calibration

300ms  processing delay

## Command syntax

Import: Use this command to upload calibration settings to one or more devices.

Import,n    import calibration string to new device

## Example

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

Import, 65 20 61 20 63 6F    (2 of 10)

⋮

Import, 6F 6C 20 67 75 79    (10 of 10)

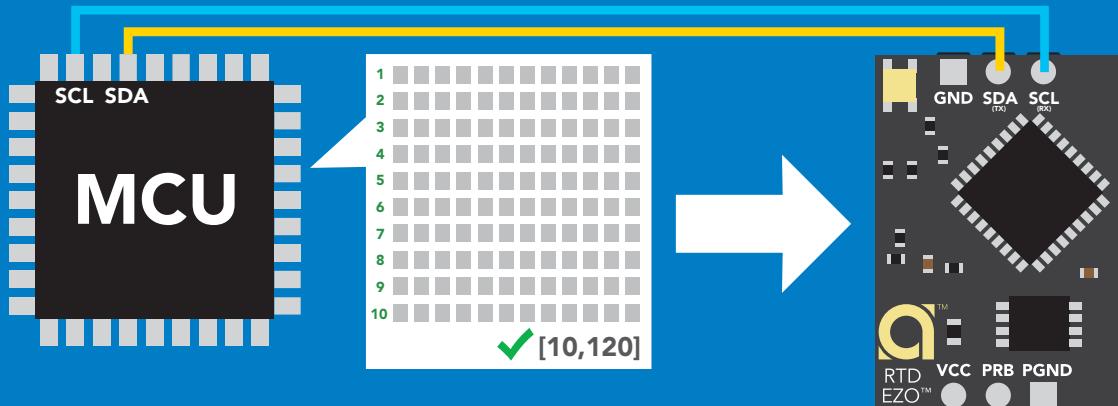
## Response

 1 0 Null  
Wait 300ms

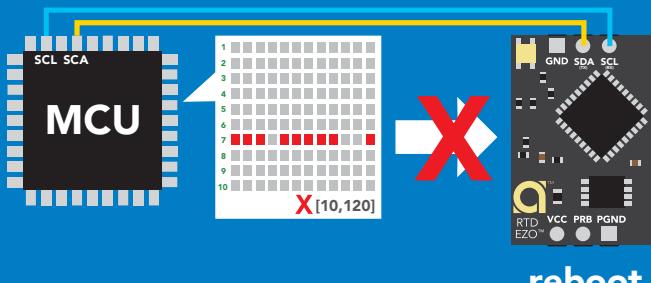
 1 0 Null  
Wait 300ms

⋮  
 1 0 Null  
Wait 300ms

Import,n



system will reboot



\* If one of the imported strings is not correctly entered, the device will not accept the import and reboot.

# Temperature scale ( $^{\circ}\text{C}$ , $^{\circ}\text{K}$ , $^{\circ}\text{F}$ )

## Command syntax

300ms  processing delay

**S,c** celsius **default**

**S,k** kelvin

**S,f** fahrenheit

**S,?** temperature scale?

## Example

**S,c**

 Wait 300ms

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

**S,k**

 Wait 300ms

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

**S,f**

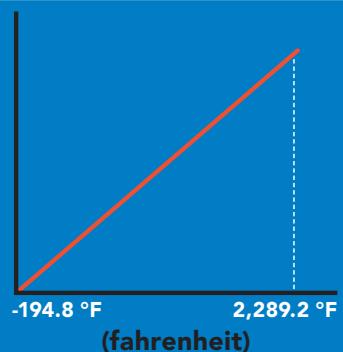
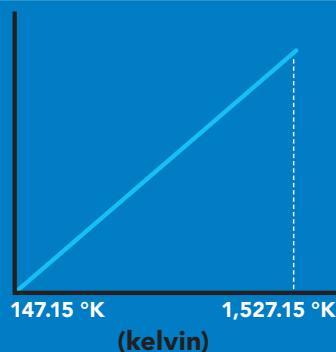
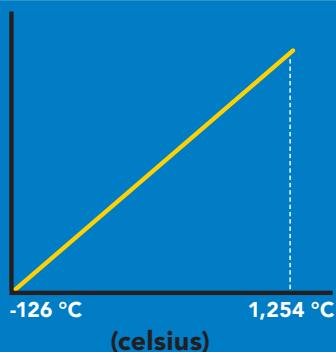
 Wait 300ms

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

**S,?**

 Wait 300ms

**1** Dec **?S,f** **0** or **1** Dec **?S,k** **0** or **1** Dec **?S,k** **0**



# Enable/disable data logger

## Command syntax

D,n n = (n x 10 seconds)

300ms  processing delay

D,0 disable

D,? data logger storage interval?

The time period (n) is in 10 second intervals and can be any value from 1 to 32,000.

## Example

D,6

 Wait 300ms  
1 Dec 0 Null

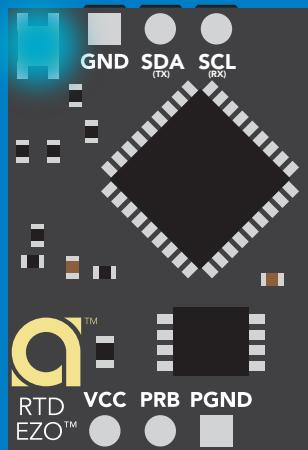
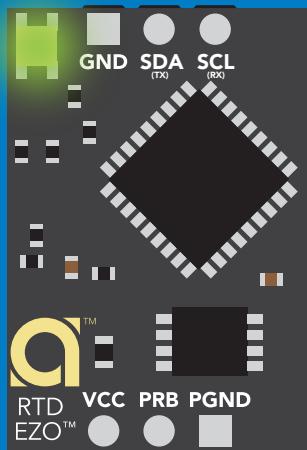
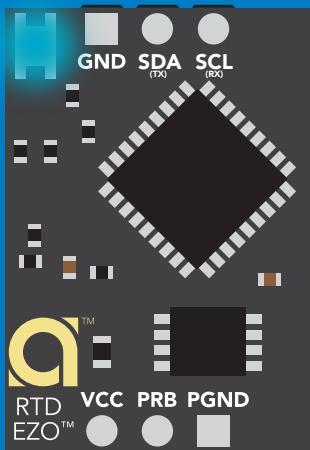
D,0

 Wait 300ms  
1 Dec 0 Null

D,?

 Wait 300ms  
1 Dec ?D,6 ASCII 0 Null

D,6  
(after 60 seconds)



# Memory recall

Disable data logger to recall memory.

## Command syntax

300ms  processing delay

M    recall 1 sequential stored reading

M,?    display memory location of last stored reading

## Example

M

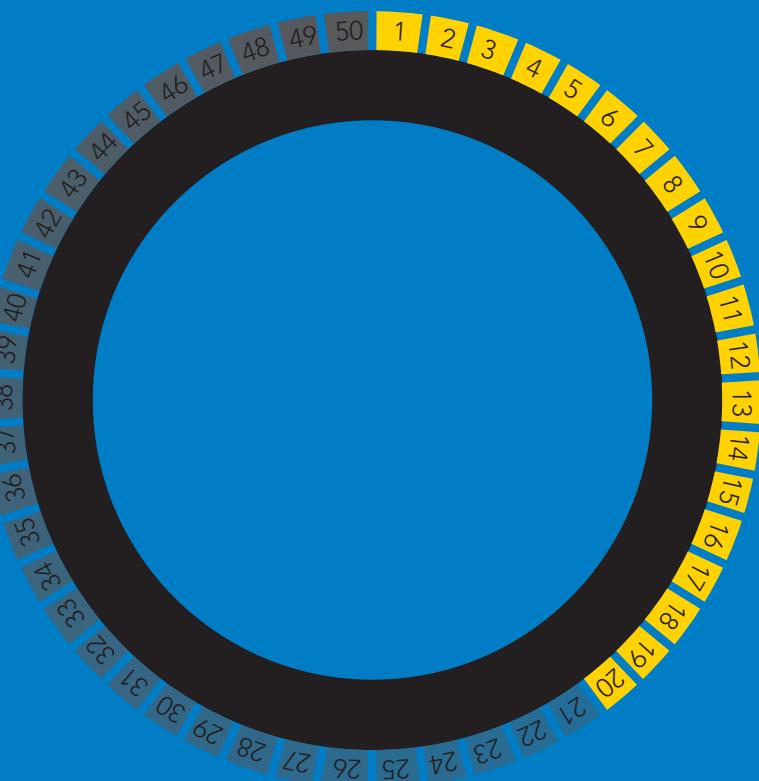


1    1,100.00    0  
Dec    ASCII    Null

M,?



1    4,112.00    0  
Dec    ASCII    Null



# Memory clear

## Command syntax

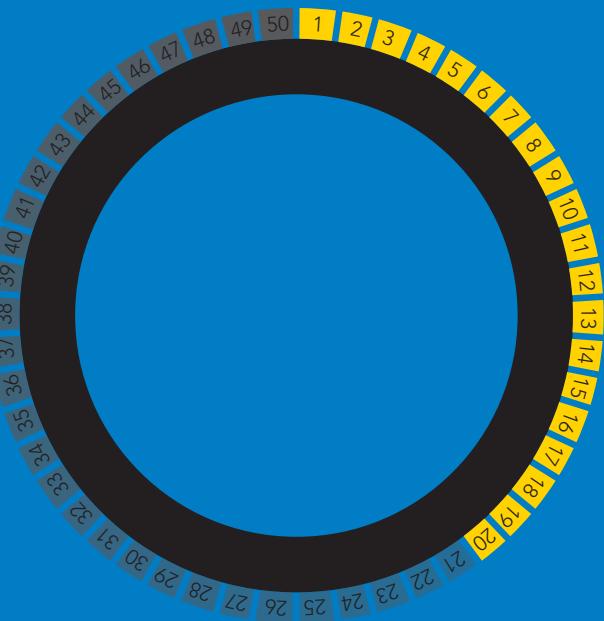
300ms  processing delay

**M,clear** clear all stored memory

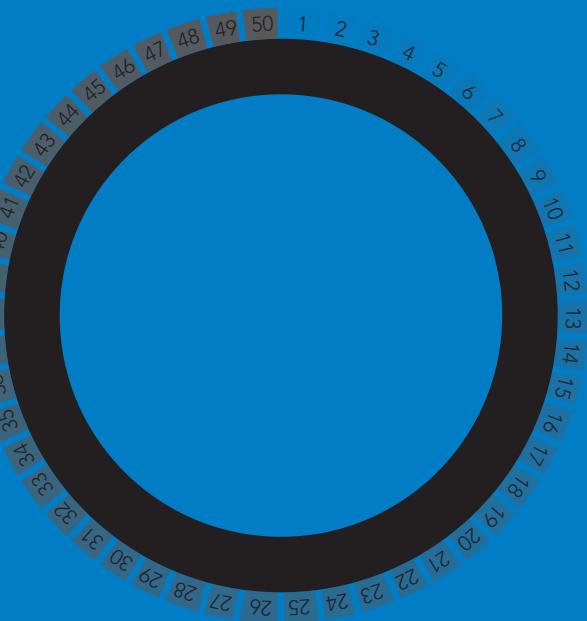
## Example Response

**M,clear**

 Wait 300ms    1 Dec 0 Null



→  
**M,clear**



# 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

Up to 16 ASCII characters

Name,?    show name

## Example

## Response

Name,



1  
Dec  
0  
Null

name has been cleared

Name,zzt



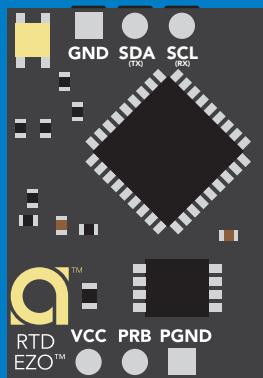
1  
Dec  
0  
Null

Name,?

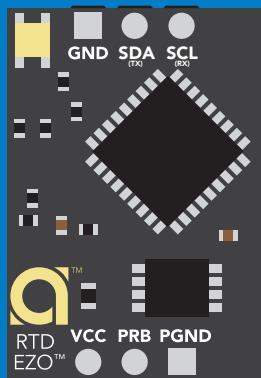


1  
Dec  
?Name,zzt  
ASCII  
0  
Null

Name,zzt



Name,?



1    0

1    ?Name,zzt    0

# Device information

## Command syntax

300ms  processing delay

### i device information

#### Example Response

i



Wait 300ms

1  
Dec

?i,RTD,2.01  
ASCII

0  
Null

#### Response breakdown

?i, RTD, 2.01

↑      ↑  
Device   Firmware

# Reading device status

## Command syntax

300ms  processing delay

Status voltage at Vcc pin and reason for last restart

## Example Response

Status



Wait 300ms

1

?Status,P,5.038

Dec

ASCII

0

Null

## Response breakdown

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

### Restart codes

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

# Sleep mode/low power

## Command syntax

**Sleep** enter sleep mode/low power

Send any character or command to awaken device.

### Example

### Response

**Sleep**

**no response**

Do not read status byte after issuing sleep command.

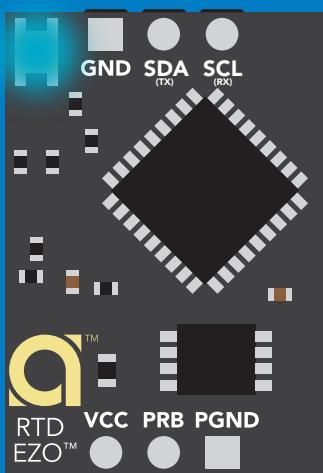
**Any command**

**wakes up device**

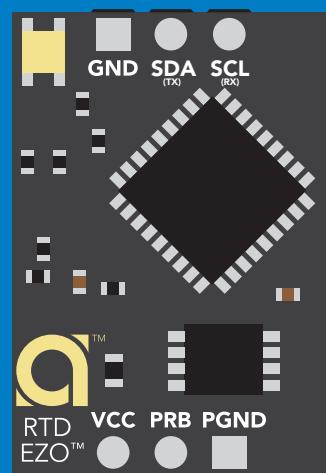
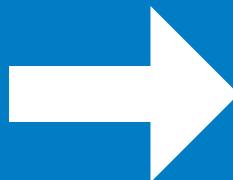
	STANDBY	SLEEP
--	---------	-------

<b>5V</b>	<b>15.40 mA</b>	<b>0.4 mA</b>
-----------	-----------------	---------------

<b>3.3V</b>	<b>13.80 mA</b>	<b>0.09 mA</b>
-------------	-----------------	----------------



Standby



Sleep

# Protocol lock

## Command syntax

300ms  processing delay

Plock,1 enable Plock

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

Plock,0 disable Plock

default

Plock,? Plock on/off?

## Example

Plock,1

 Wait 300ms

1  
Dec  
0  
Null

Plock,0

 Wait 300ms

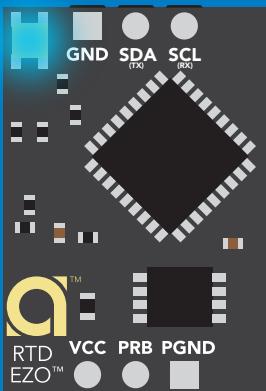
1  
Dec  
0  
Null

Plock,?

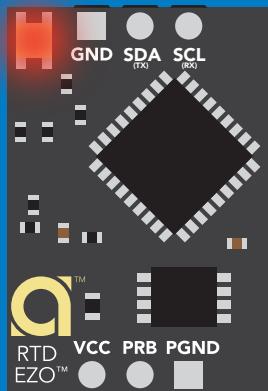
 Wait 300ms

1  
Dec  
?Plock,1  
ASCII  
0  
Null

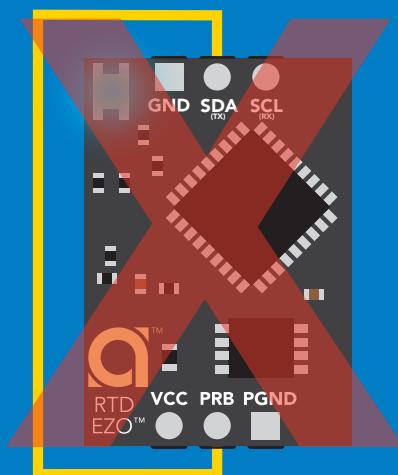
Plock,1



Baud, 9600



cannot change to UART



cannot change to UART

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

## Command syntax

300ms  processing delay

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

## Example Response

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

device reboot

(no response given)

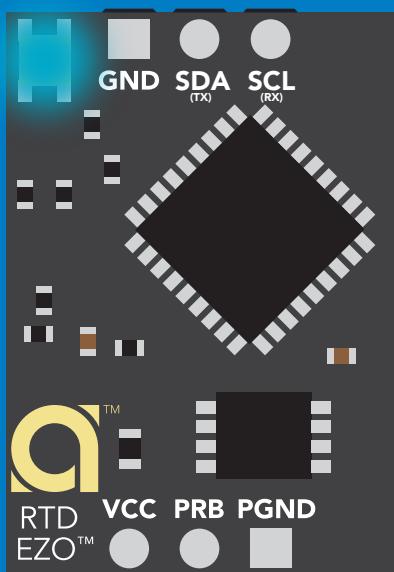
## Warning!

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

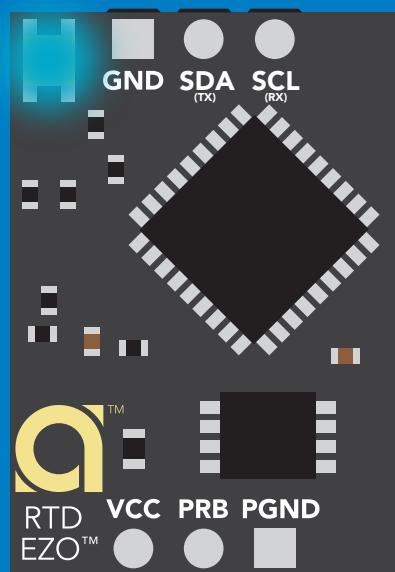
Default I<sup>2</sup>C address is 102 (0x66).

n = any number 1 – 127

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



(reboot)



# Factory reset

## Command syntax

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

Factory enable factory reset

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

## Example Response

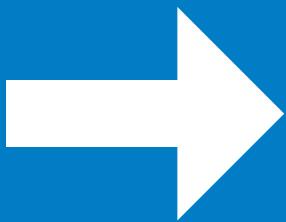
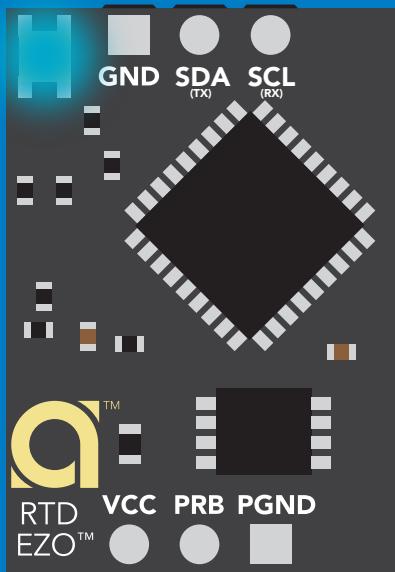
Factory

device reboot

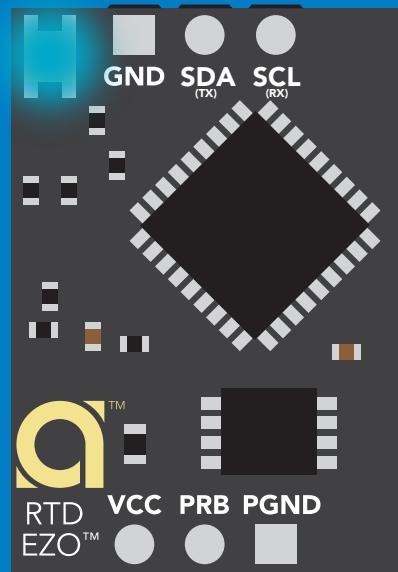
(no response given)

Clears calibration  
LED on  
Response codes enabled  
Clears data logger

Factory



(reboot)



# Change to UART mode

## Command syntax

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

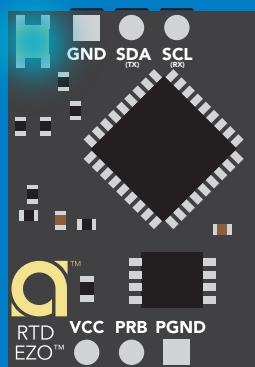
### Example      Response

Baud,9600

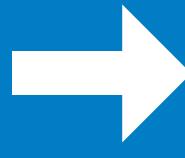
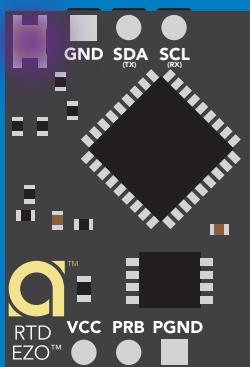
reboot in UART mode

(no response given)

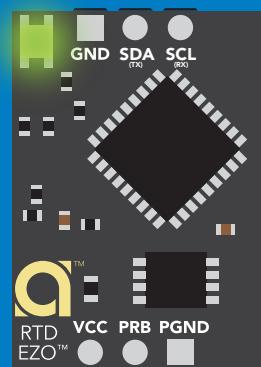
n = [ 300  
1200  
2400  
9600  
19200  
38400  
57600  
115200 ]



Baud,9600



(reboot)



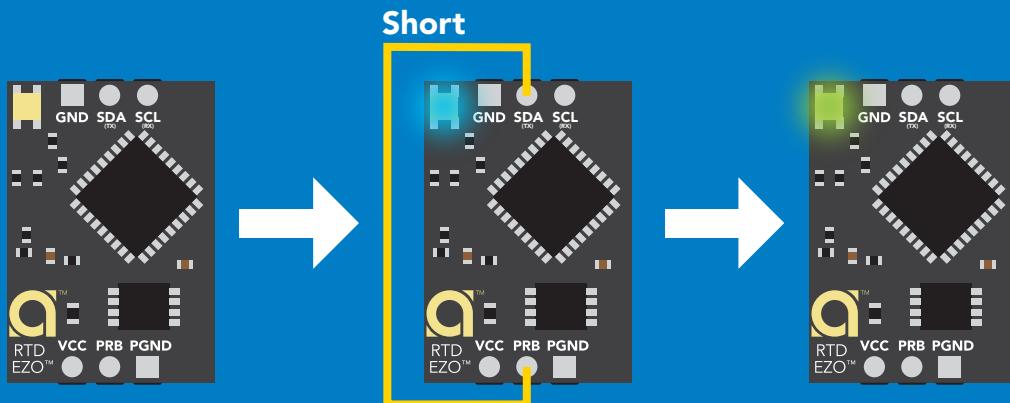
Changing to  
UART mode

# Manual switching to UART

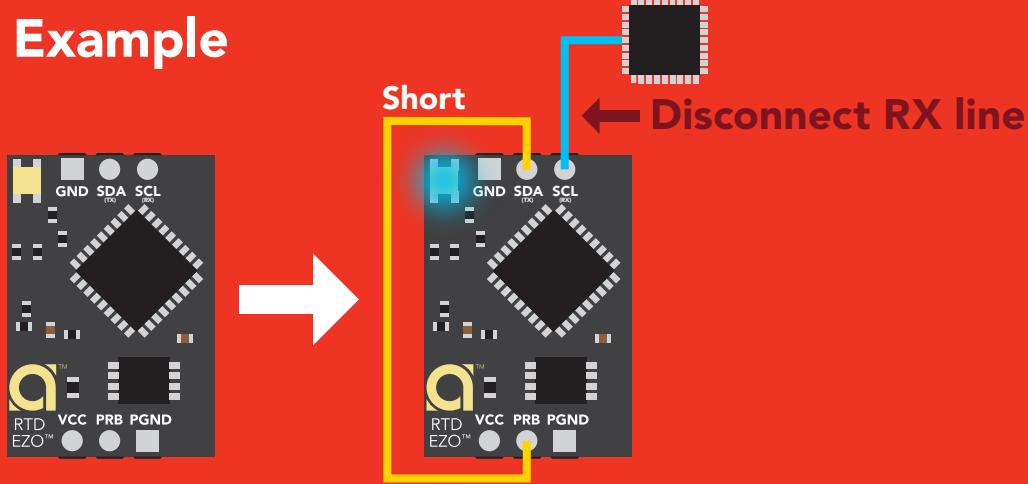
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to PRB
- Confirm RX is disconnected
- Connect ground (power on)
- Wait for LED to change from **Blue** to **Green**
- Disconnect ground (power off)
- Reconnect all data and power

Connecting TX to PRB only works for the EZO-RTD™ and the EZO-FLO™ circuits

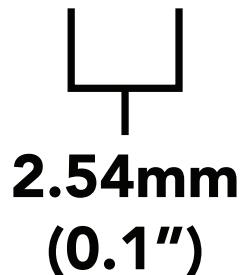
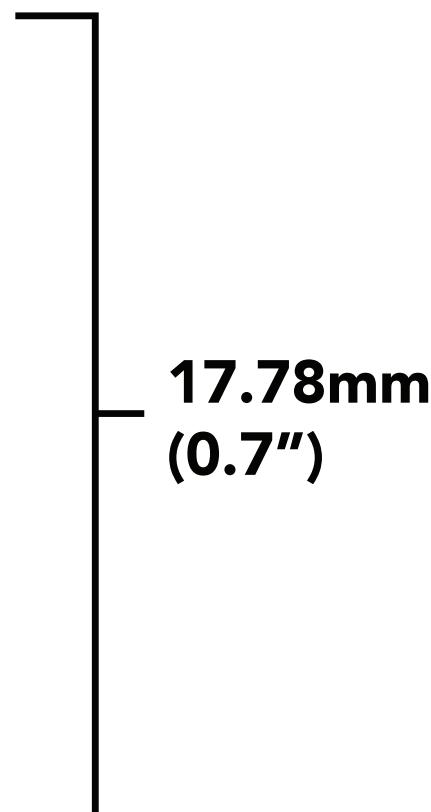
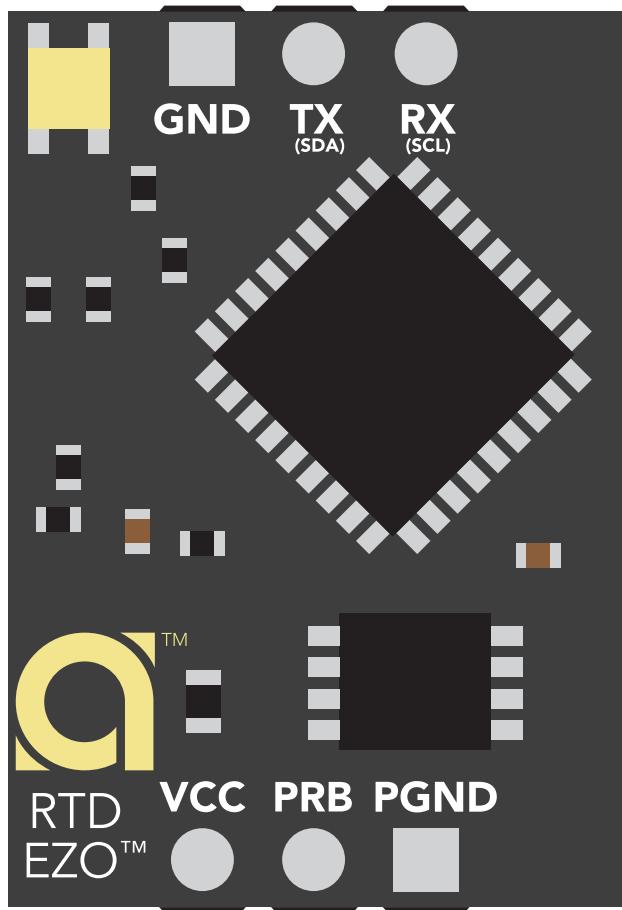
## Example



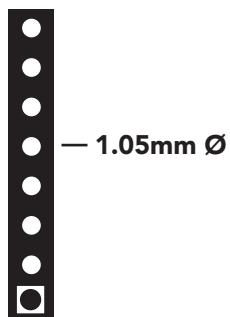
## Wrong Example



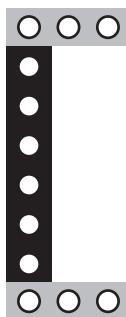
# EZO™ circuit footprint



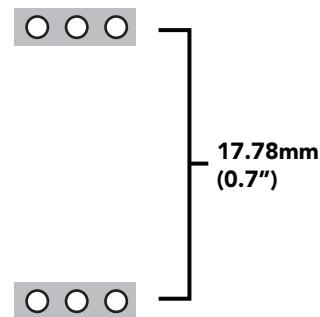
**1** In your CAD software place a 8 position header.



**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 3.6**

Revised naming device info on pages 33 & 60.

## **Datasheet V 3.5**

Added 2 wire, 3 wire, or 4 wire, wiring diagram on pg 7.

## **Datasheet V 3.4**

Revised accuracy equation on pg 7.

## **Datasheet V 3.3**

Moved Default state to pg 13.

## **Datasheet V 3.2**

Revised response for the sleep command in UART mode on pg 36.

## **Datasheet V 3.1**

Added more information on the Export calibration and Import calibration commands.

## **Datasheet V 3.0**

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

## **Datasheet V 2.9**

Removed note from certain commands about firmware version.

## **Datasheet V 2.8**

Added information to calibration theory on pg 9.

## **Datasheet V 2.7**

Revised definition of response codes on pg 45.

## **Datasheet V 2.6**

Updated calibration processing delay time on pg.51.

## **Datasheet V 2.5**

Revised Plock pages to show default value.

# Datasheet change log

## Datasheet V 2.4

### Added new commands:

"Find" pages 22 & 49.

"Export/Import calibration" pages 26 & 52.

Added new feature to continuous mode "C,n" pg 23.

## Datasheet V 2.3

Added manual switching to UART information on pg. 59.

## Datasheet V 2.2

Revised Baud command information on pg. 33.

## Datasheet V 2.1

Revised entire datasheet.

# Firmware updates

V1.02 – Plock (March 31, 2016)

- Added protocol lock feature "Plock"

V1.03 – EEPROM (April 26, 2016)

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

V1.11 – Bug Fix (June 9, 2016)

- Fixed bug where a blank name would result in garbage output

V2.01 – Update (January 1, 2017)

- Replaced command "response" with "\*OK"
- Replaced command "Serial" with "Baud"

V2.02 – Bug Fix (February 16, 2017)

- Fixed bug where calibration would not accept floating point numbers.

V2.10 – (May 9, 2017)

- Added "Find" command.
- Added "Export/import" command.
- Modified continuous mode to be able to send readings every "n" seconds.
- Sleep current is lowered.

V2.11 - Bug Fix (November 19, 2020)

- Fixed bug where the first reading after boot up could be -1024

# Warranty

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

## The debugging phase

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

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