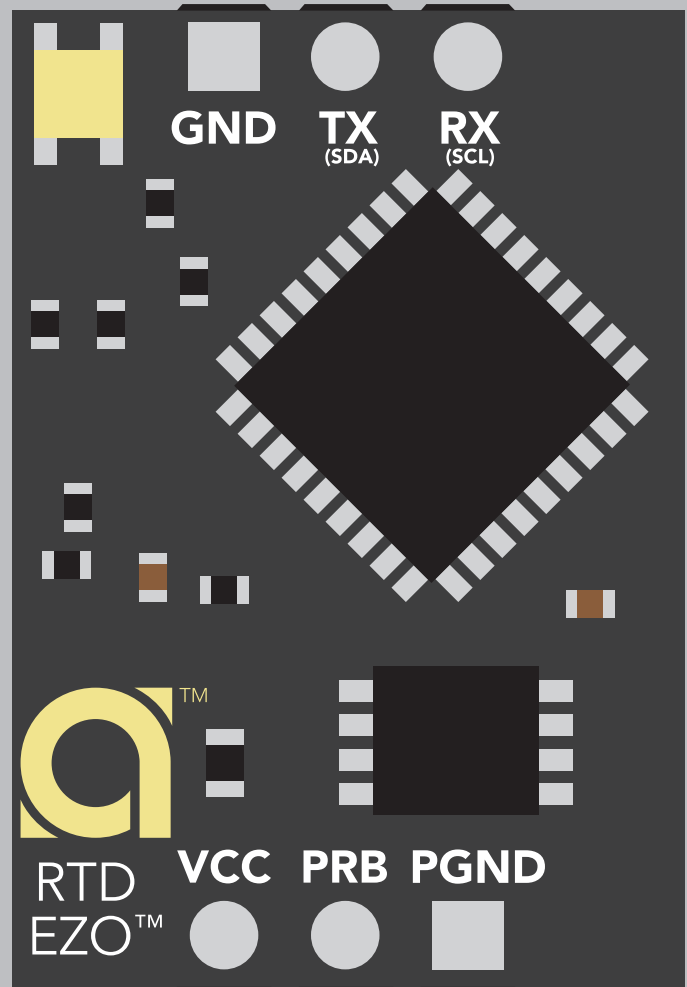


RTD Temperature EZO™

Circuit

Range	-126.000 °C – 1254 °C
Resolution	0.001
Accuracy	$\pm(0.10^{\circ}\text{C} + 0.0017 * T)$
Speed	1 reading per sec
Supported probes	Any type & brand PT-100 or PT-1000 RTD
Calibration	Single point
Temperature output	°C, °K, or °F
Data protocol	UART & I²C
Operating voltage	3.3V – 5.5V
Data format	ASCII
Onboard Data Logger	50 Readings



Electrical Isolation not needed





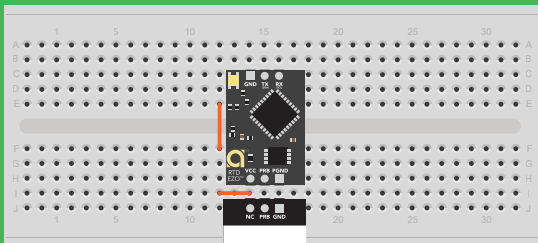
STOP

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!

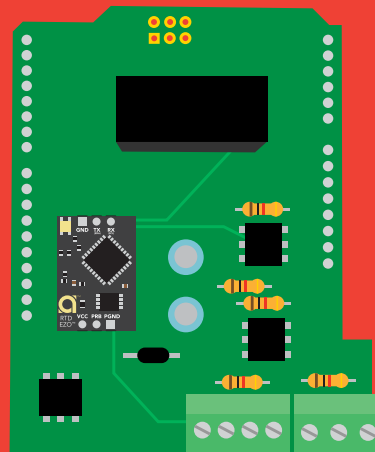


Table of contents

Circuit dimensions	4	Using other brand PT-100/PT-1000	7
Power consumption	4	Operating principle	8
Absolute max ratings	4	Calibration theory	9
Temperature circuit range	5	On board data logger	10
Temperature circuit accuracy	5	Correct wiring	11
Atlas Scientific PT-1000 probe	6	Available data protocols	12

UART

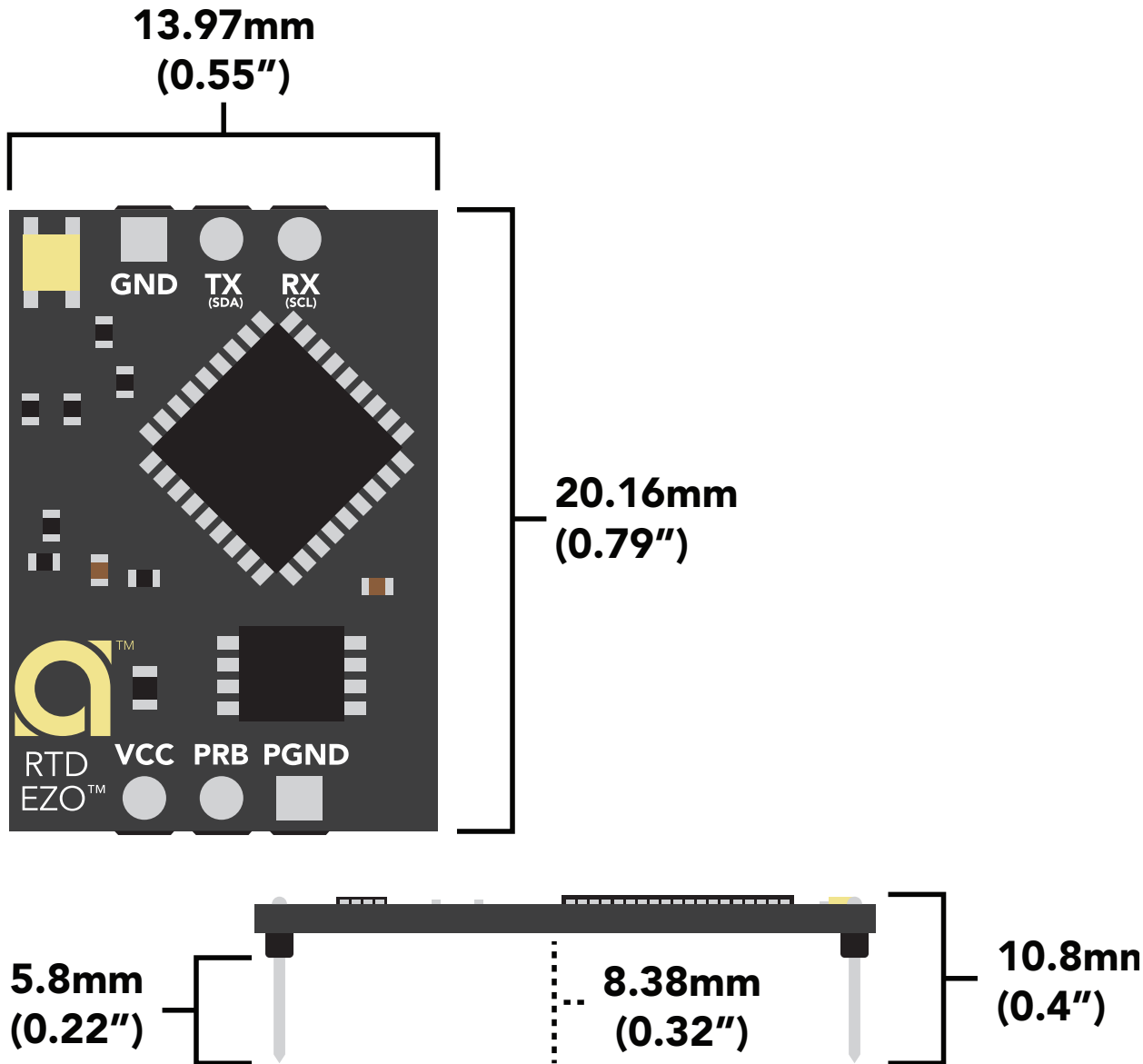
UART mode	14
Default state	15
Receiving data from device	16
Sending commands to device	17
LED color definition	18
UART quick command page	19
LED control	20
Continuous reading mode	21
Single reading mode	22
Calibration	23
Temperature scale	24
Enable/disable data logger	25
Memory recall	26
Memory clear	27
Naming device	28
Device information	29
Response codes	30
Reading device status	31
Sleep mode/low power	32
Change baud rate	33
Protocol lock	34
Factory reset	35
Change to I ² C mode	36
Manual switching to I ² C	37

I²C

I ² C mode	39
Sending commands	40
Requesting data	41
I ² C data processing delay	42
LED color definition	43
I²C quick command page	44
LED control	45
Taking reading	46
Calibration	47
Temperature scale	48
Enable/disable data logger	49
Memory recall	50
Memory clear	51
Device information	52
Reading device status	53
Sleep mode/low power	54
Protocol lock	55
I ² C address change	56
Factory reset	57
Change to UART mode	58

Circuit footprint	57
Datasheet change log	58
Warranty	59

EZO™ circuit dimensions



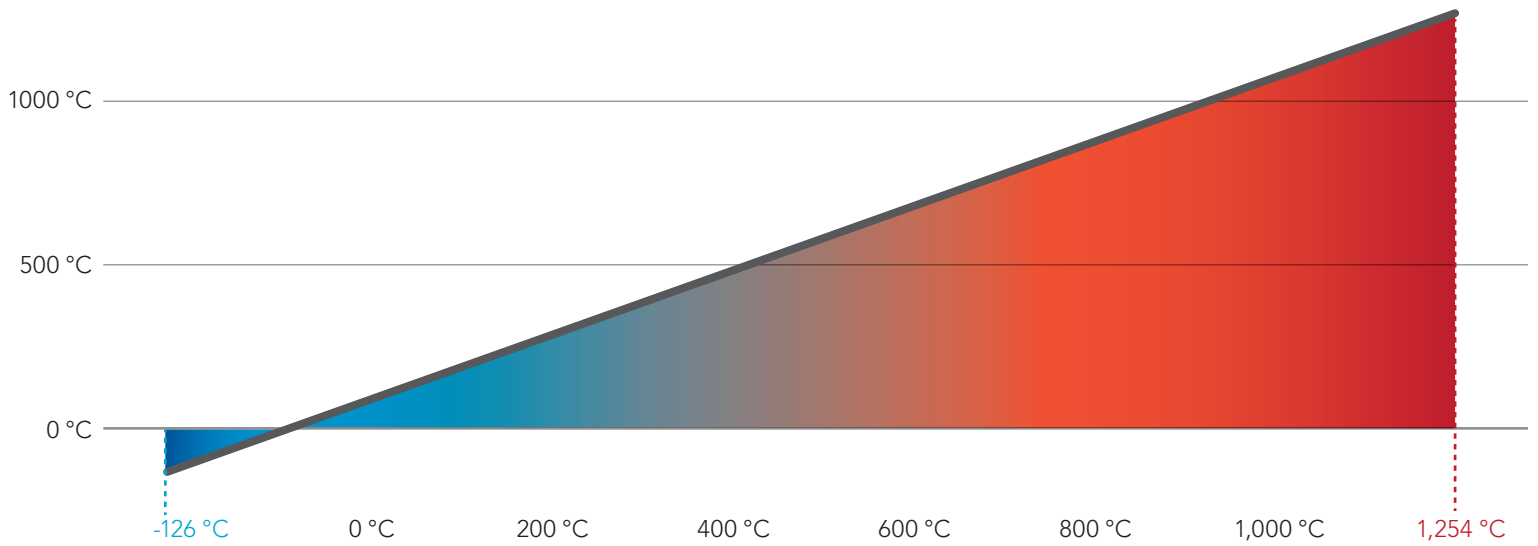
Power consumption

	LED	MAX	STANDBY	SLEEP
5V	ON	16 mA	15.4 mA	3.00 mA
	OFF	15.3 mA	15 mA	
3.3V	ON	14.3 mA	13.8 mA	1.46 mA
	OFF	14 mA	13.6 mA	

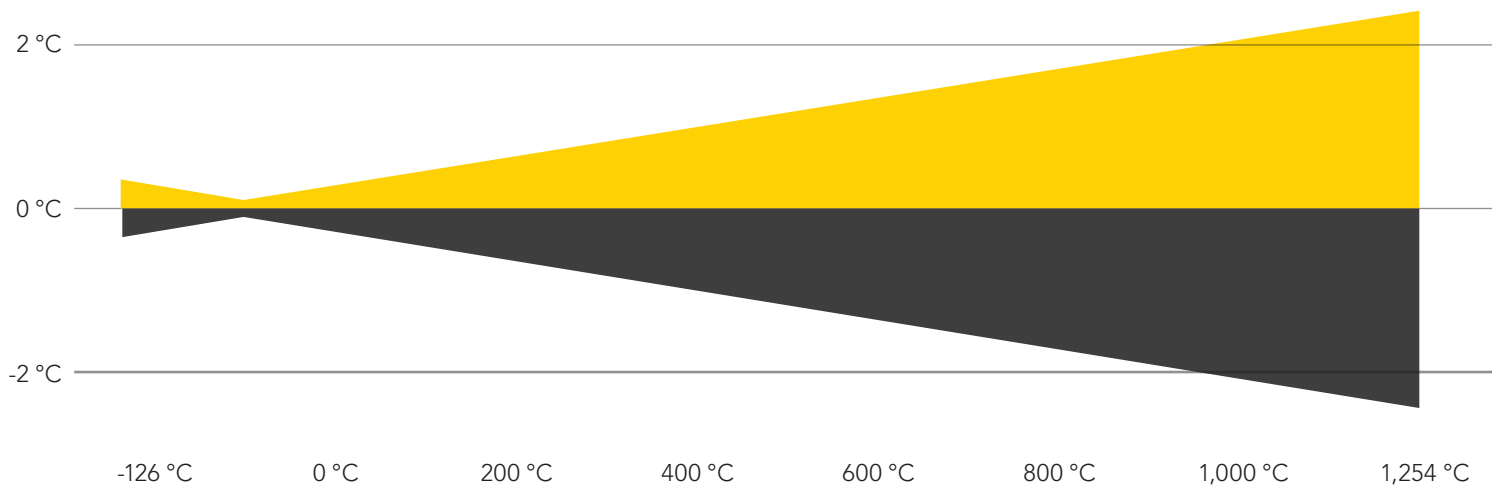
Absolute max ratings

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 circuit range



EZO™ RTD temperature circuit 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



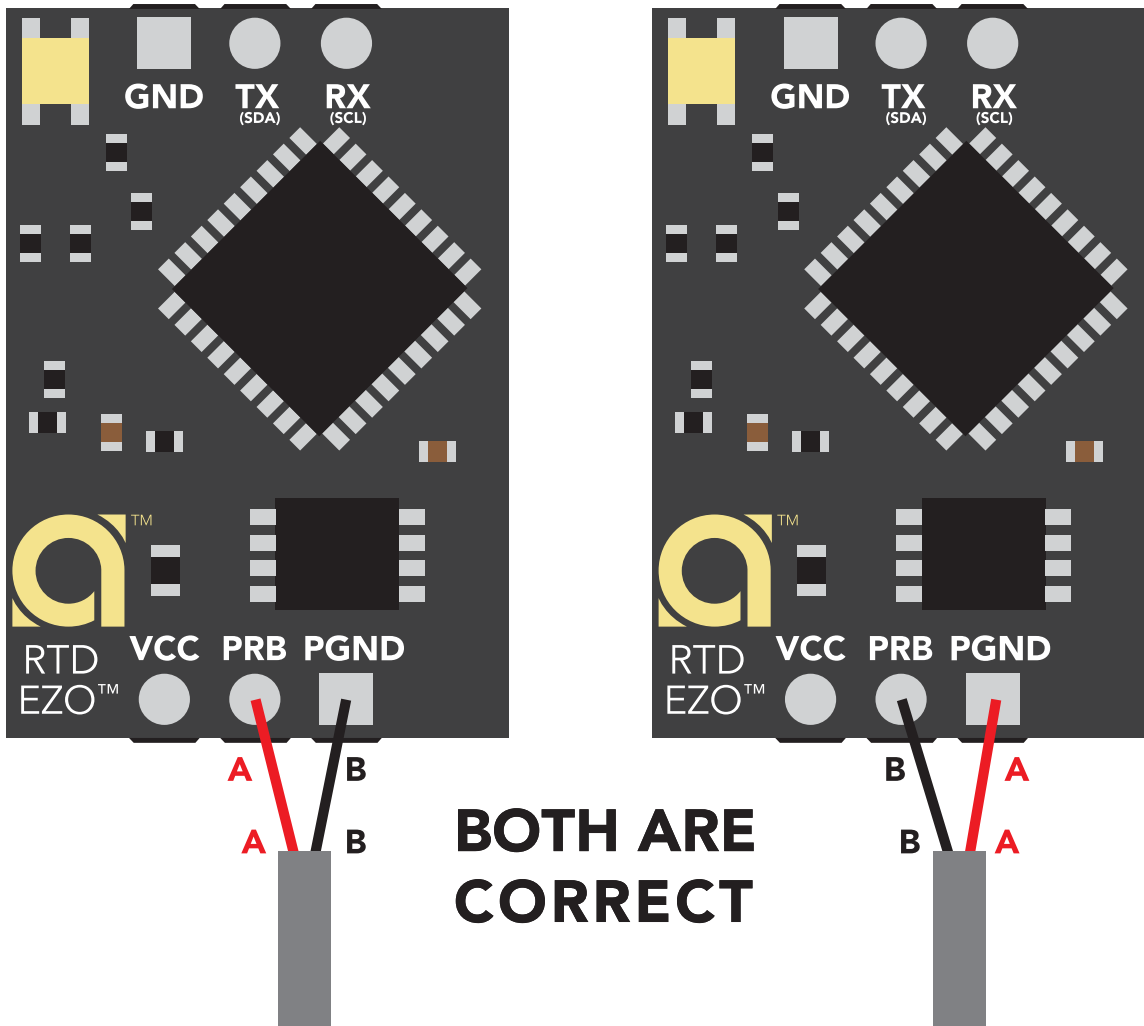
30mm Temperature Thermowell

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	$\pm(0.10^{\circ}\text{C} + 0.0017 * T)$
A	$\pm(0.15^{\circ}\text{C} + 0.002 * T)$
B	$\pm(0.3^{\circ}\text{C} + 0.005 * T)$
C	$\pm(0.6^{\circ}\text{C} + 0.01 * T)$

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



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.

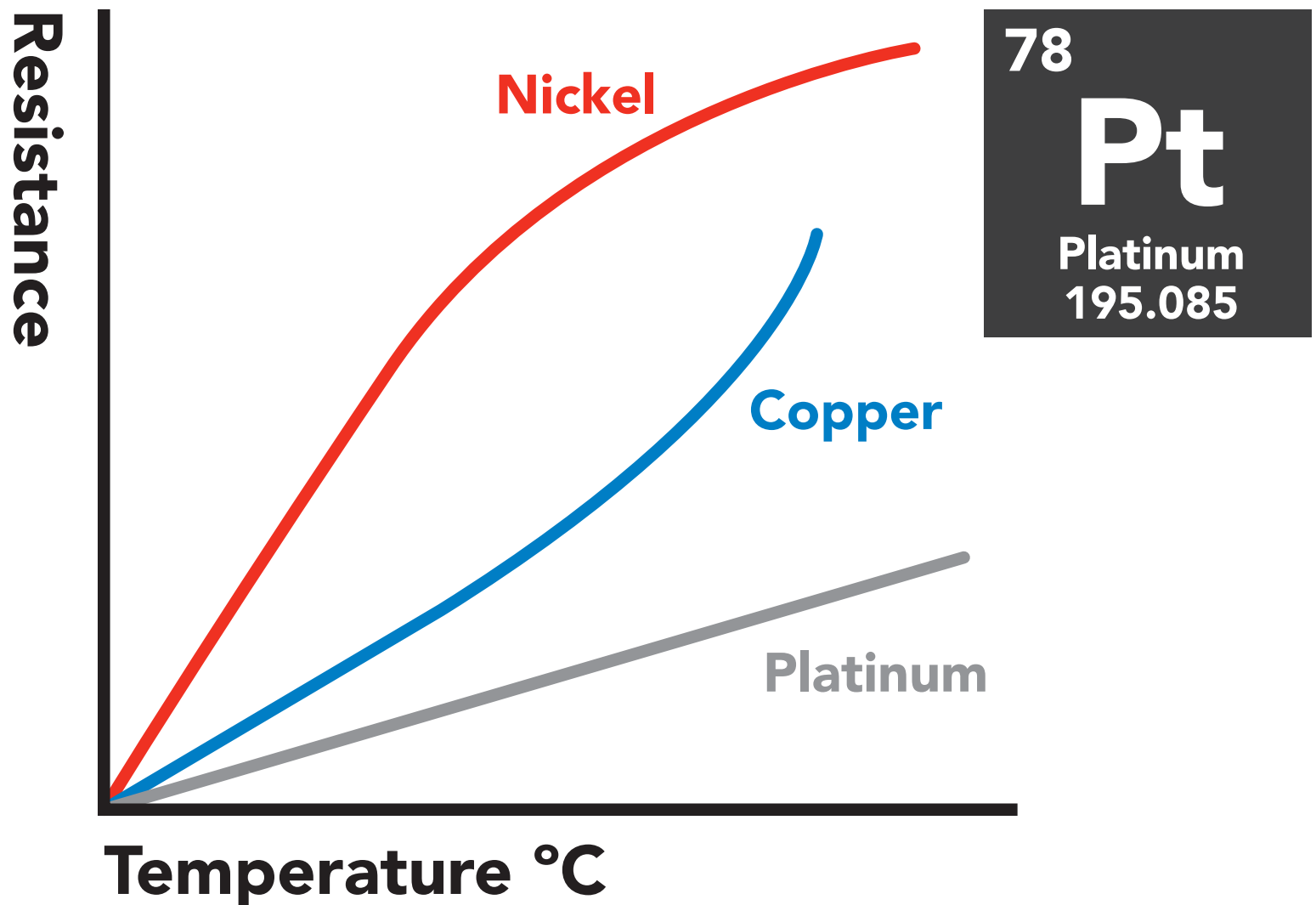
RTD = Resistance Temperature Detector

PT = Platinum

PT-100 = 100 Ω at 0°C

PT-1000 = 1k Ω at 0°C

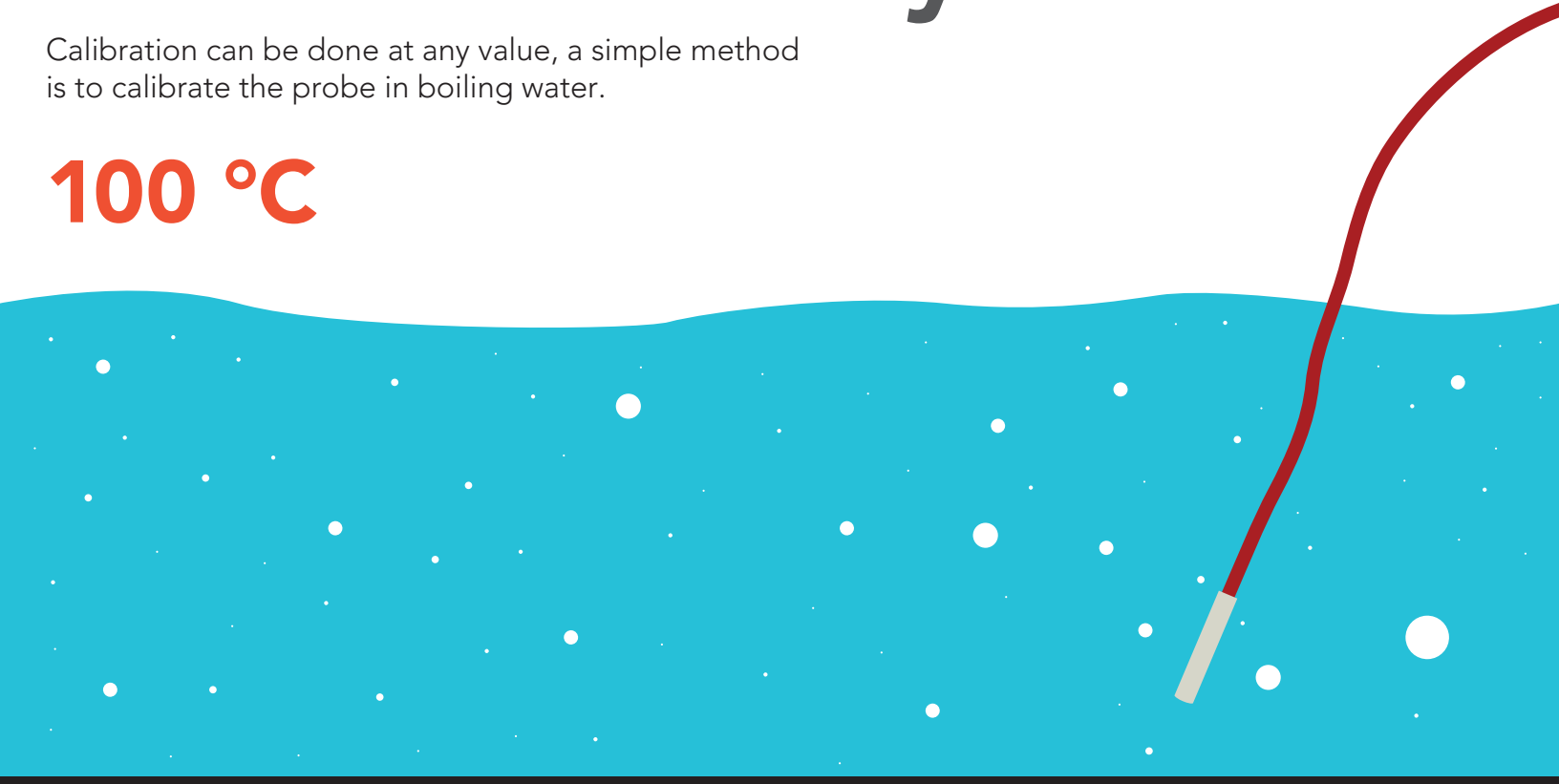
Unlike any other material, platinum's correlation between resistance and temperature seems to be woven into the fabric of the universe. It is for this reason, that the platinum RTD temperature sensor is the industrial standard for temperature measurement.



Calibration theory

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

100 °C



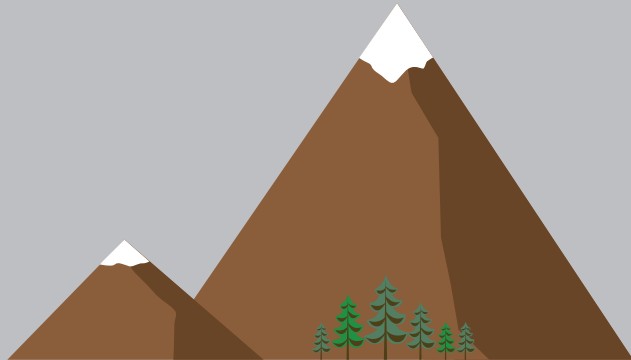
Elevation and Boiling Point table

Elevation in meters

-152
-76
0
76
152
229
305

Boiling point

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

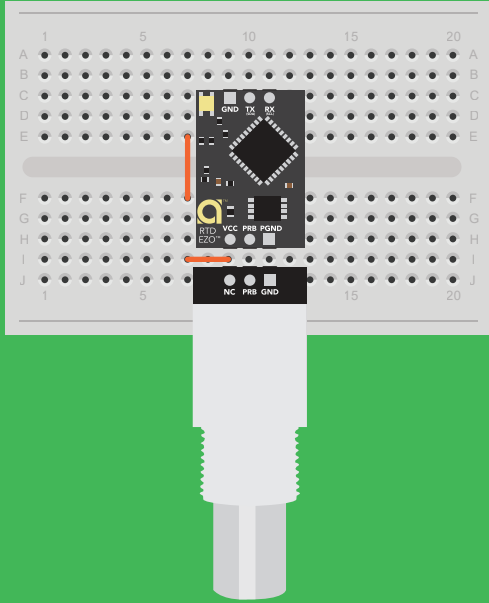
Maximum – 320,000 seconds

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

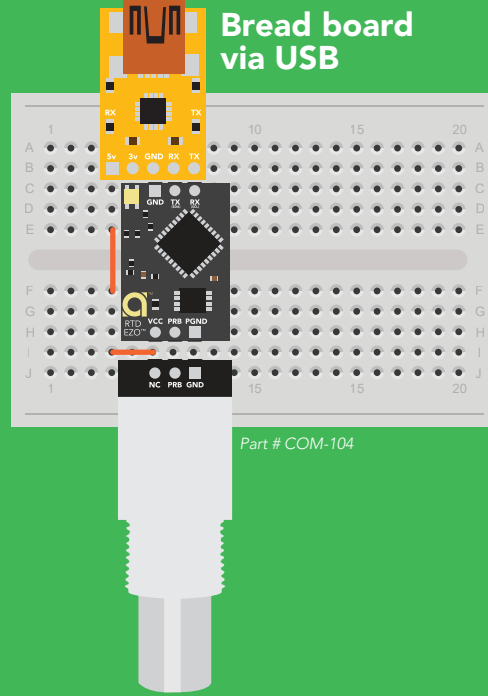


✓ Correct wiring

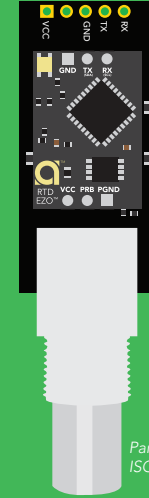
Bread board



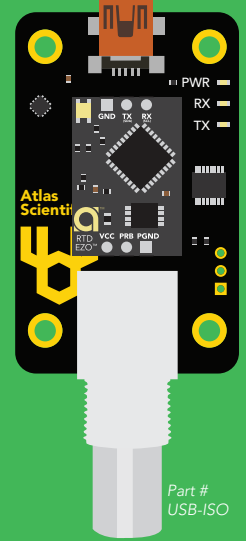
Bread board via USB



Carrier board

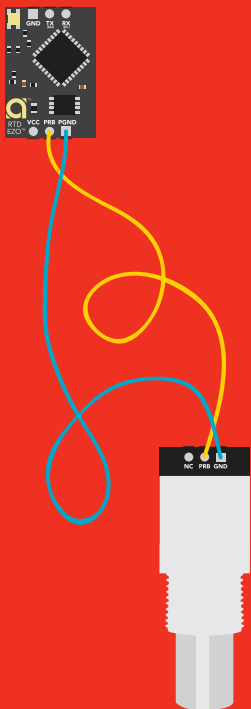


USB carrier board

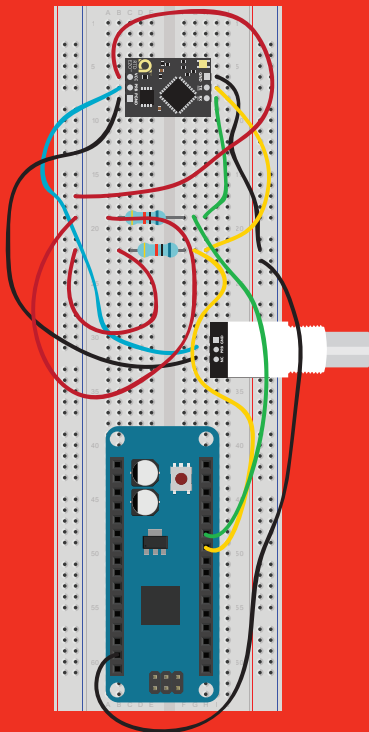


X Incorrect wiring

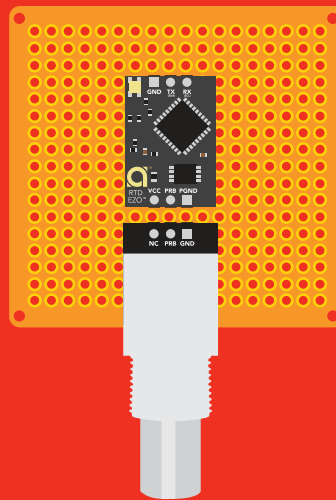
Extended leads



Sloppy setup

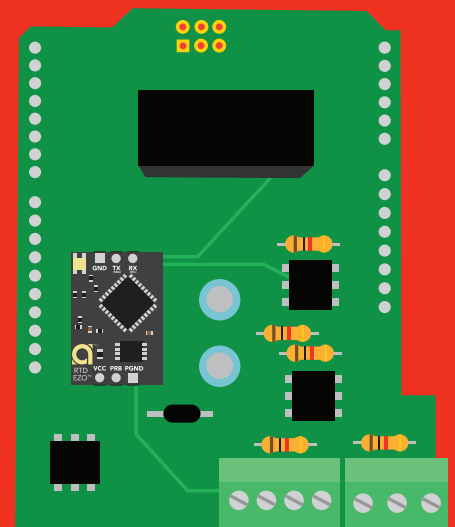


Perfboards or Protoboards



NEVER
use Perfboards
or Protoboards

*Embedded into your device



*Only after you are familiar
with EZO™ circuits operation

✓ Available data protocols

UART

Default

I²C

X Unavailable data protocols

SPI

Analog

RS-485

Mod Bus

4–20mA

UART mode

Any settings that have been changed will be retained even if the power has been cut from the device.

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



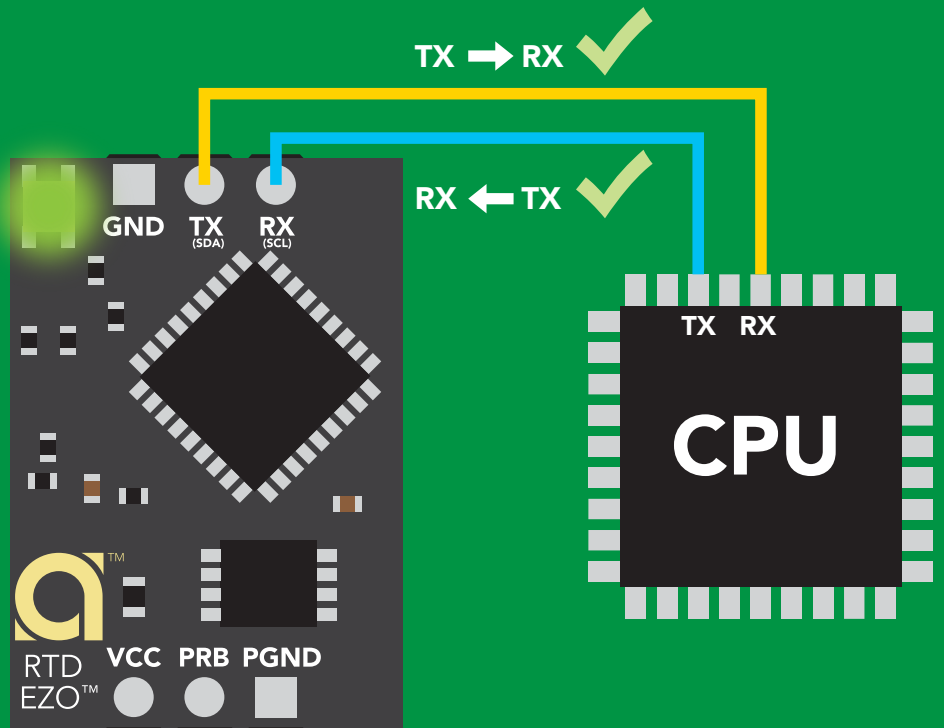
TX
Data out



Vcc 3.3V – 5.5V



0V VCC 0V

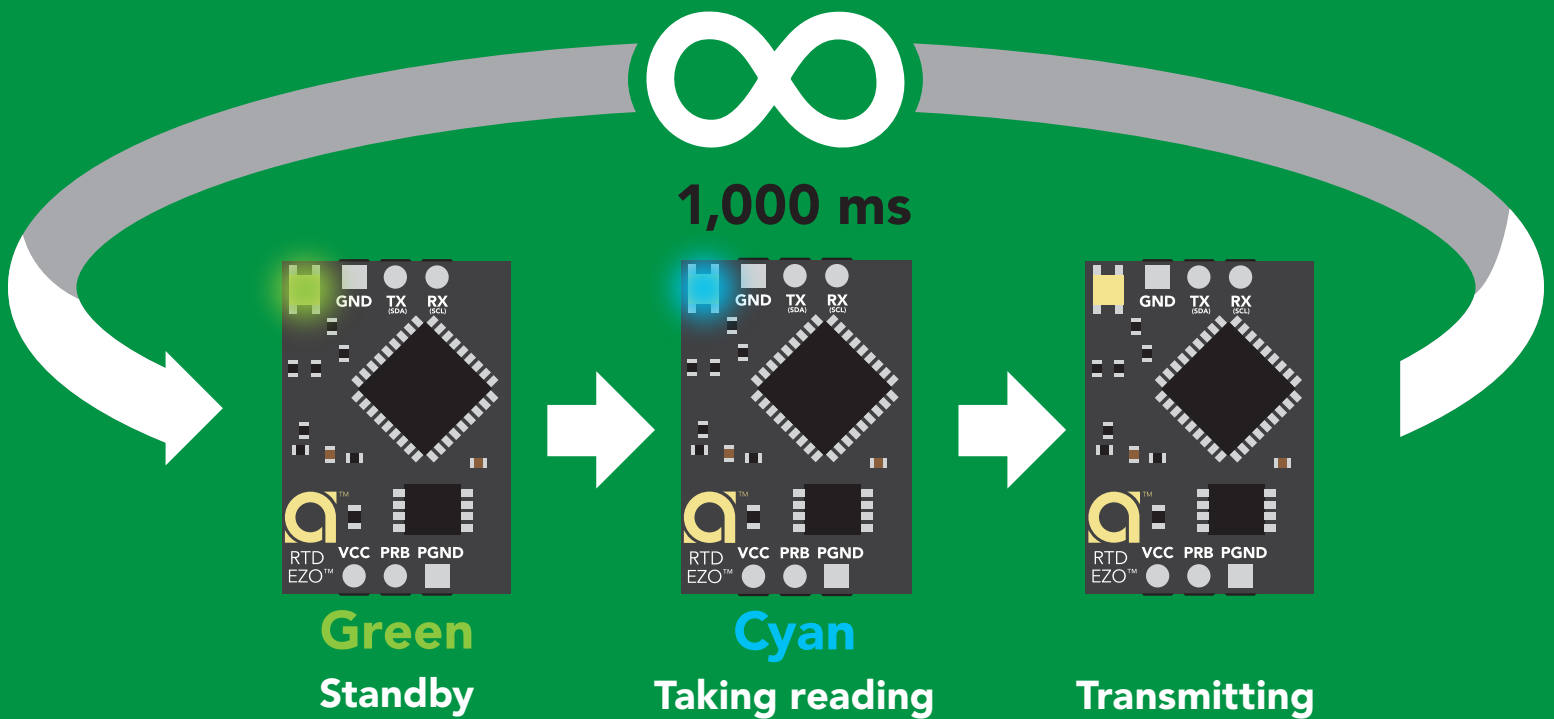


Data format

Reading	temperature	Data type	floating point
Units	°C, °K, or °F	Decimal places	3
Encoding	ASCII	Smallest string	4 characters
Format	string	Largest string	399 characters
Terminator	carriage return		

Default state

Mode	UART
Baud	9,600
Temperature	°C
Readings	continuous
Speed	1 reading per second
With probe	ttt.ttt
Without probe	-1023.000
LED	on



Receiving data from device

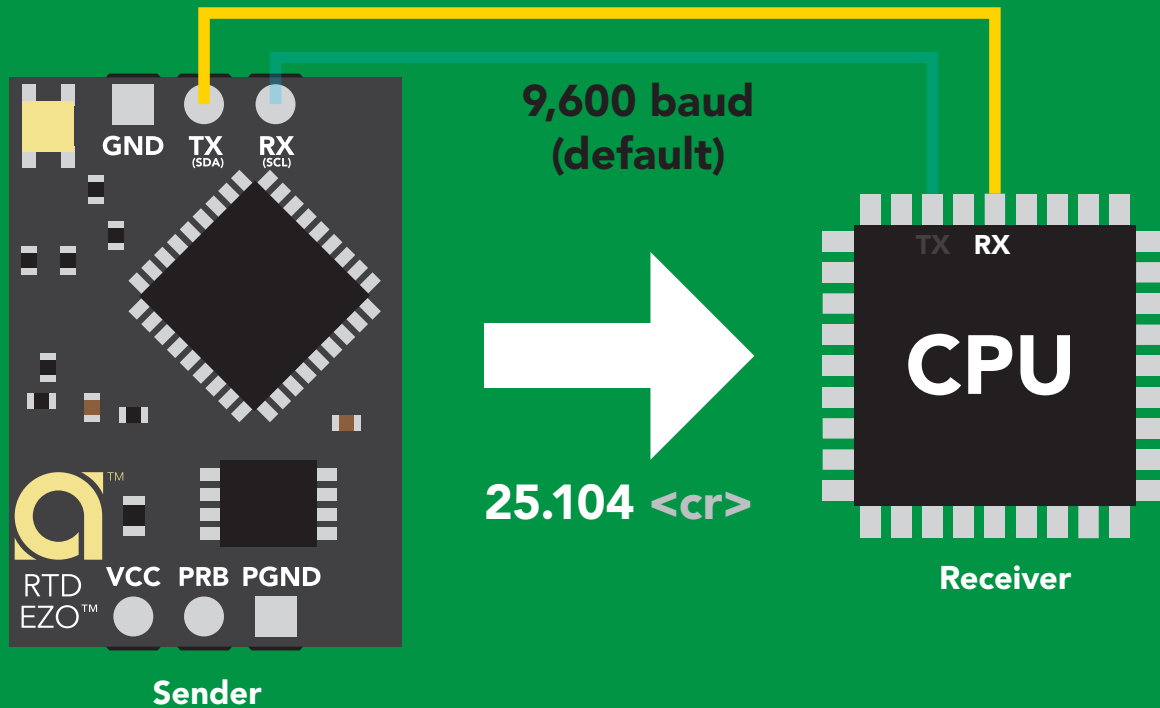
2 parts

ASCII data string

Command

Carriage return <cr>

Terminator



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

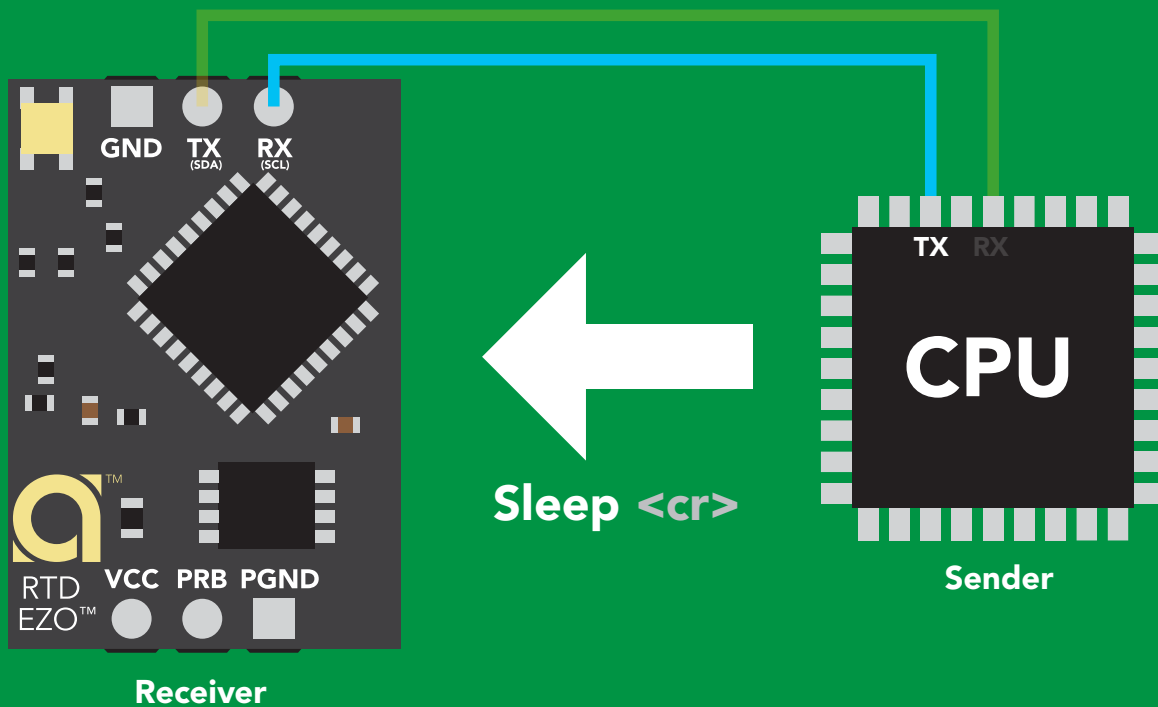
2 parts

Command (not case sensitive)

ASCII data string

Carriage return <cr>

Terminator



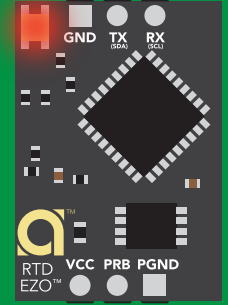
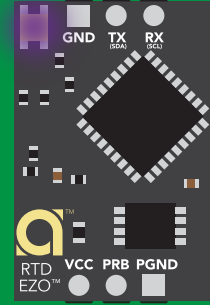
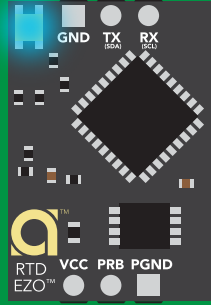
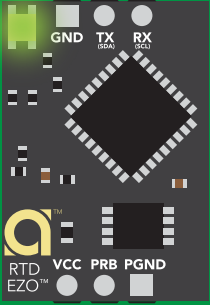
Advanced

ASCII: **S** **I** **e** **e** **p** **<cr>**

Hex: **53** **6C** **65** **65** **70** **0D**

Dec: **83** **108** **101** **101** **112** **13**

LED color definition



Green

UART standby

Cyan

Taking reading

Purple

**Changing
baud rate**

Red

**Command
not understood**

5V

LED ON
+0.4 mA

3.3V

+0.2 mA

UART mode

command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function		Default state
Baud	change baud rate	pg. 33	9,600
C	enable/disable/state of continuous reading	pg. 21	enabled
Cal	performs calibration	pg. 23	n/a
D	enable/disable data logger	pg. 25	disabled
Factory	enable factory reset	pg. 35	n/a
i	device information	pg. 29	n/a
I2C	change to I ² C mode	pg. 36	not set
L	enable/disable/state of LED	pg. 20	enabled
M	memory recall/clear	pg. 26	n/a
Name	set/show name of device	pg. 28	not set
Plock	enable/disable protocol lock	pg. 34	disabled
R	returns a single reading	pg. 22	n/a
S	temperature scale (°C, °K, °F)	pg. 24	celsius
Sleep	enter sleep mode/low power	pg. 32	n/a
Status	retrieve status information	pg. 31	n/a
*OK	enable/disable/state of response codes	pg. 30	enable

LED control

Command syntax

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

L,0 <cr> LED off

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

Example

Response

L,1 <cr>

*OK <cr>

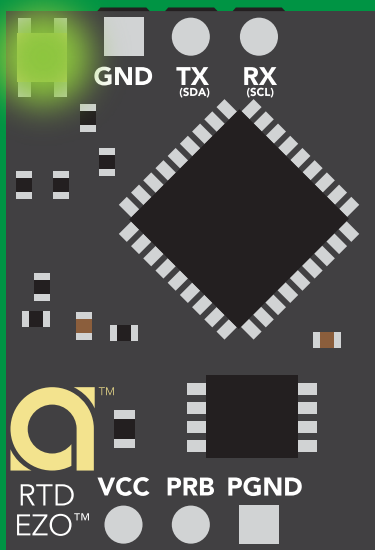
L,0 <cr>

*OK <cr>

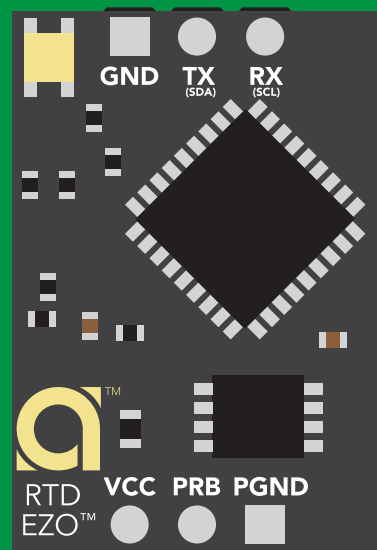
L,? <cr>

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

*OK <cr>



L,1



L,0

Continuous reading mode

Command syntax

- C,1 <cr>** enable continuous readings once per second **default**
- C,0 <cr>** disable continuous readings
- C,? <cr>** continuous reading mode on/off?

Example

Response

C,1 <cr>

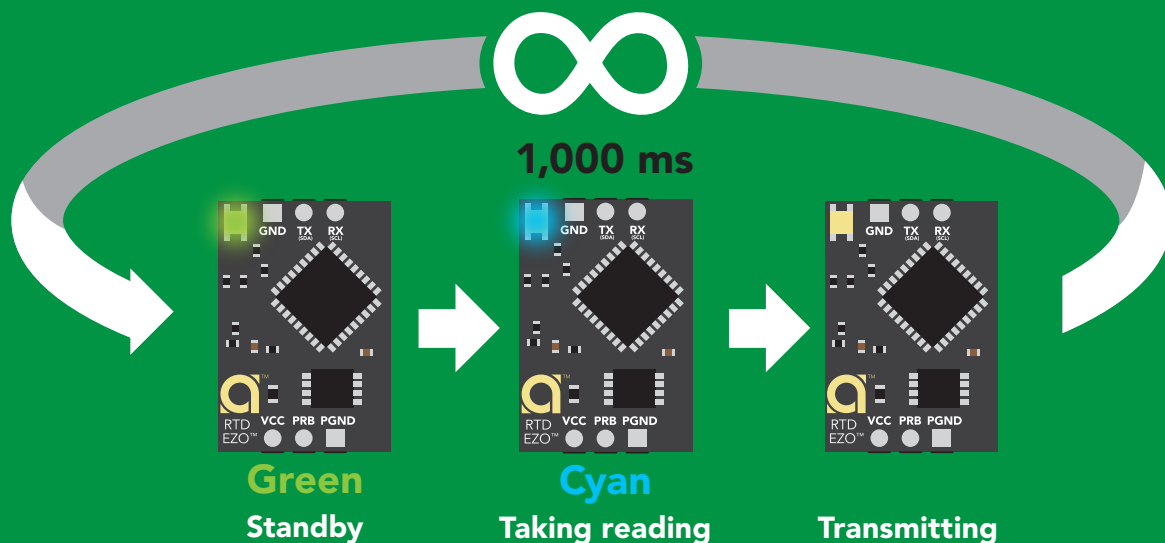
***OK <cr>**
°C (1 sec) <cr>
°C (2 sec) <cr>
°C (n sec) <cr>

C,0 <cr>

***OK <cr>**

C,? <cr>

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



Single reading mode

Command syntax

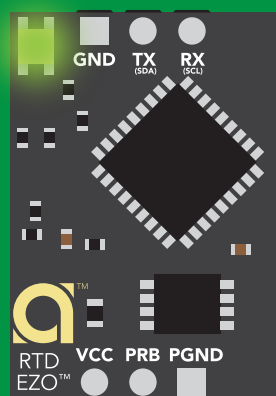
R <cr> takes single reading

Example

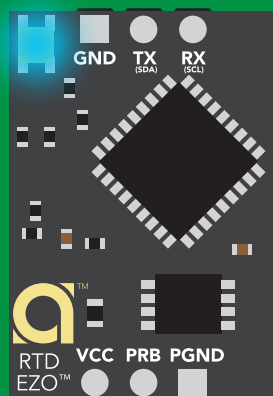
R <cr>

Response

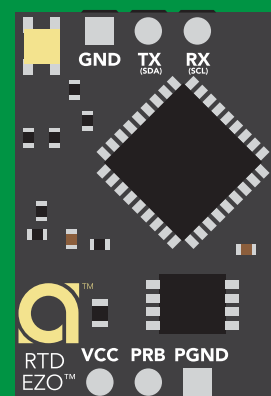
25.104 <cr>
*OK <cr>



Green
Standby



Cyan
Taking reading



Yellow
Transmitting



1,000 ms

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

Response

Cal,100.00 <cr>

***OK** <cr>

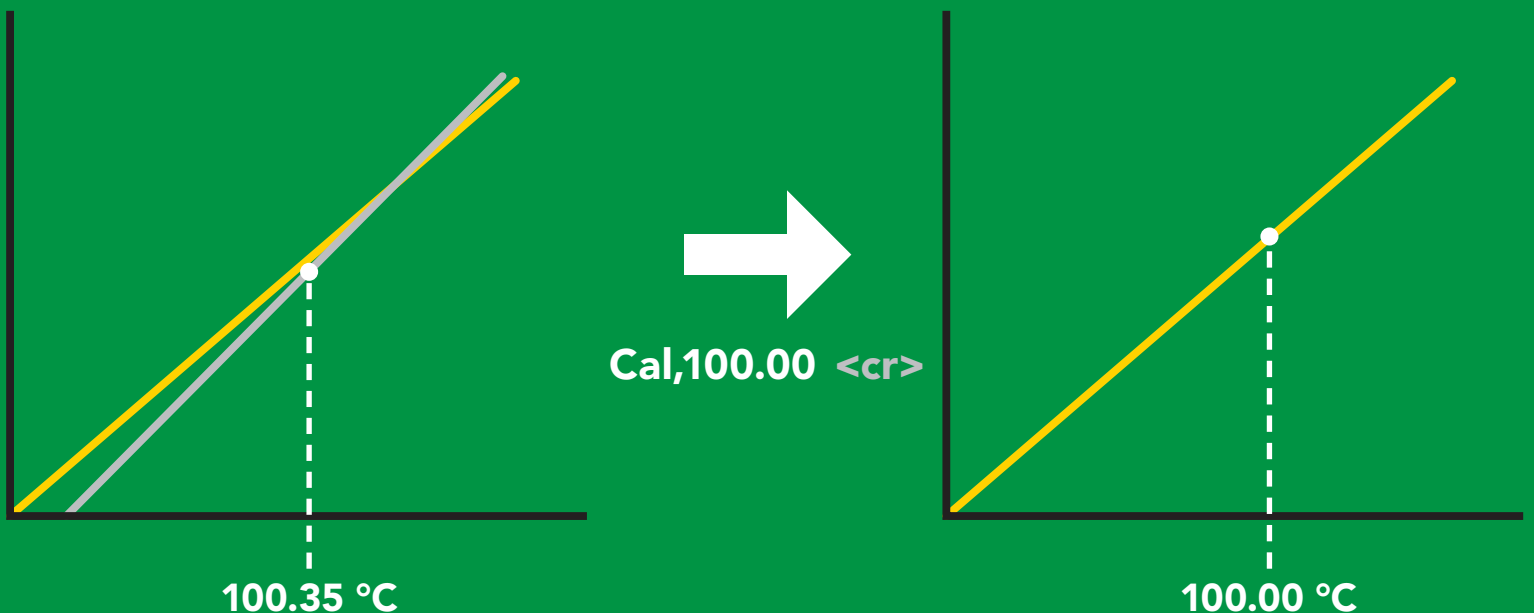
Cal,clear <cr>

***OK** <cr>

Cal,? <cr>

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

***OK** <cr>



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

Response

S,c <cr>

*OK <cr>

S,k <cr>

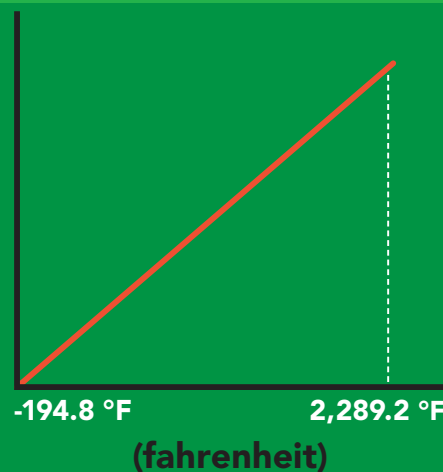
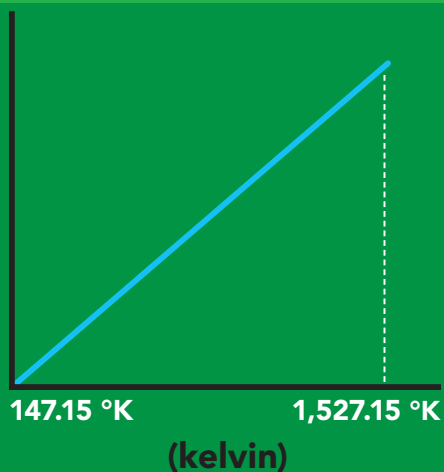
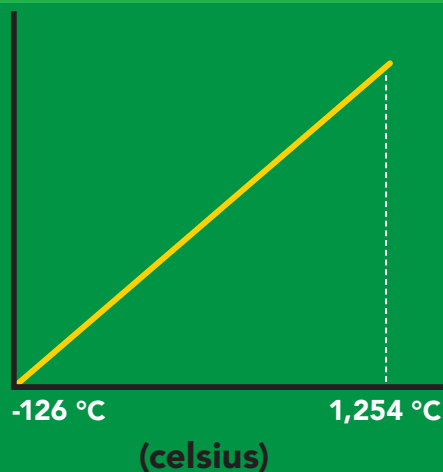
*OK <cr>

S,f <cr>

*OK <cr>

S,? <cr>

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



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

Response

D,6 <cr>

*OK <cr>

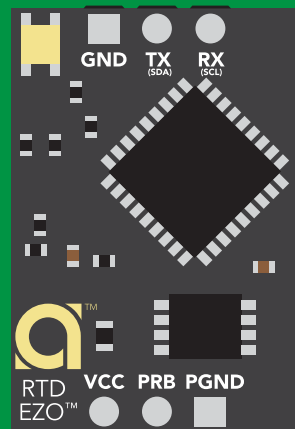
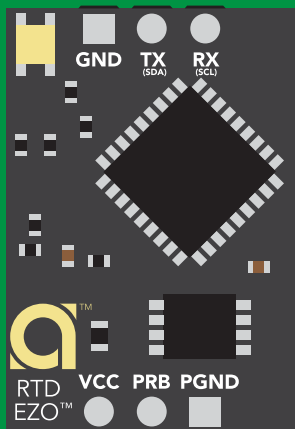
D,0 <cr>

*OK <cr>

D,? <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>

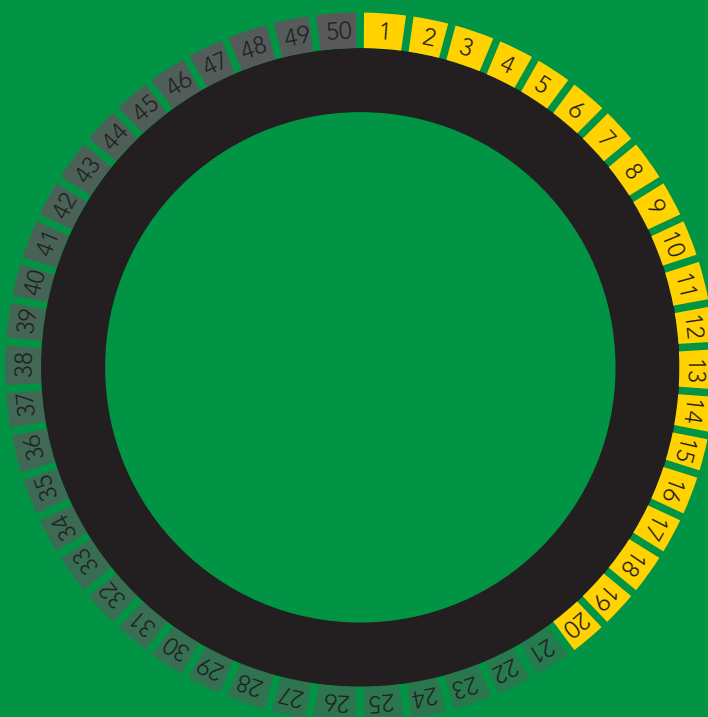
1,100.00 <cr> **2,104.00** <cr> ***OK** <cr>

M,all <cr>

100.00,104.00,108.00,112.00 <cr>
Oldest Newest

M,? <cr>

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



Memory clear

Command syntax

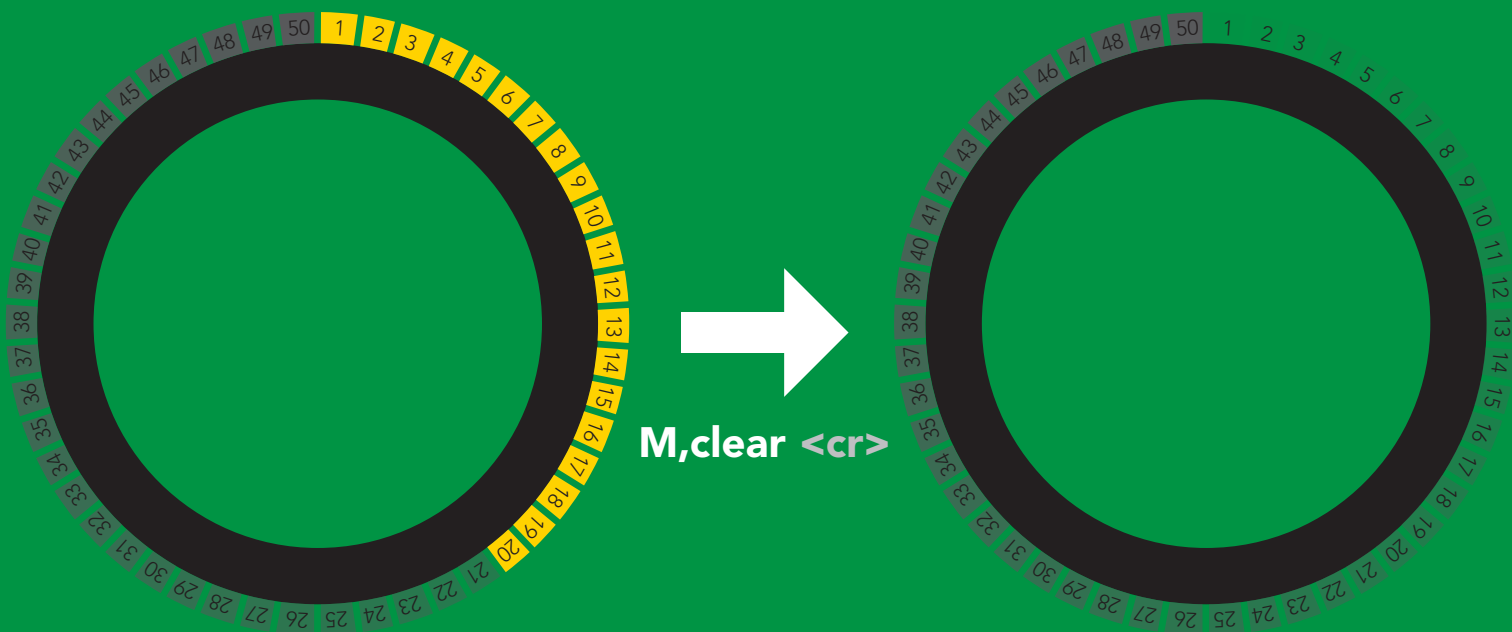
M,clear <cr> clear all stored memory

Example

M,clear <cr>

Response

***OK** <cr>



Naming device

Command syntax

Name,n <cr> set name

Name,? <cr> show name

n =

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

Up to 16 ASCII characters

Example

Name,zzt <cr>

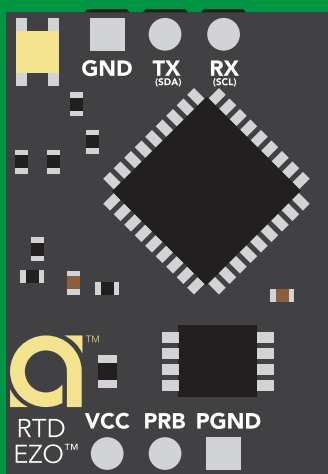
Response

*OK <cr>

Name,? <cr>

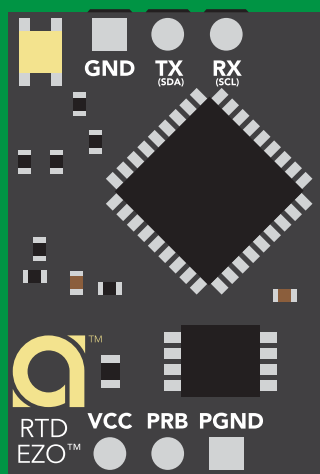
?Name,zzt <cr>
*OK <cr>

Name,zzt



*OK <cr>

Name,?



Name,zzt <cr>
*OK <cr>

Device information

Command syntax

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

Example

```
i <cr>
```

Response

```
?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 ($VCC \geq 5.5V$)
***UV** under volt ($VCC \leq 3.1V$)
***RS** reset
***RE** boot up complete, ready
***SL** entering sleep mode
***WA** wake up

These response codes
cannot be disabled

Reading device status

Command syntax

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

Example

```
Status <cr>
```

Response

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

Response breakdown

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

Restart codes

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

Sleep mode/low power

Command syntax

Send any character or command to awaken device.

Sleep <cr> enter sleep mode/low power

Example

Response

Sleep <cr>

*SL

Any command

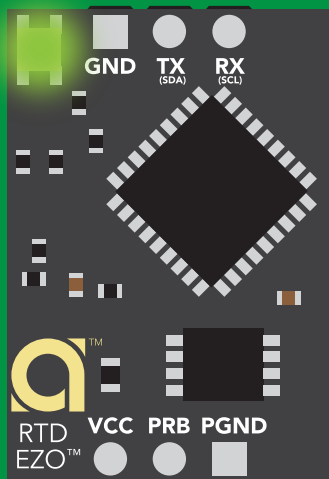
*WA <cr> wakes up device

5V

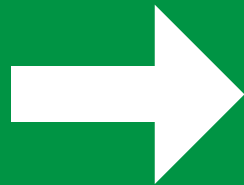
STANDBY	SLEEP
15.40 mA	3.00 mA

3.3V

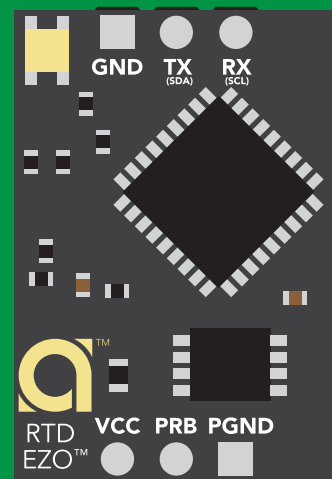
13.80 mA	1.46 mA
----------	---------



Standby
15.40 mA



Sleep <cr>



Sleep
3.00 mA

Change baud rate

Command syntax

Baud,n <cr> change baud rate

Example

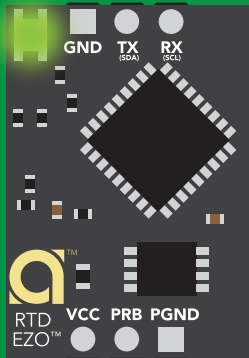
Baud,38400 <cr>

Response

*OK <cr>

n =

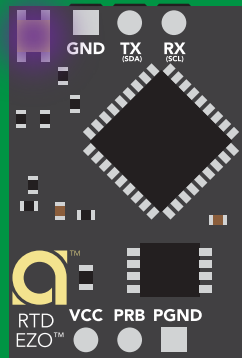
- 300
- 1200
- 2400
- 9600
- 19200
- 38400
- 57600
- 115200



Standby



Baud,38400 <cr>

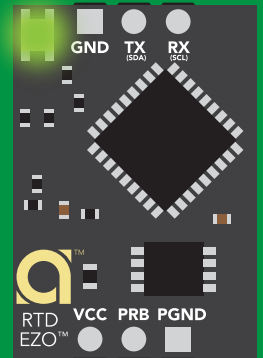


Changing
baud rate

*OK <cr>



(reboot)



Standby

Protocol lock

Command syntax

Locks device to UART mode.

Plock,1 <cr> enable Plock

Plock,0 <cr> disable Plock

Plock,? <cr> Plock on/off?

Example

Response

Plock,1 <cr>

*OK <cr>

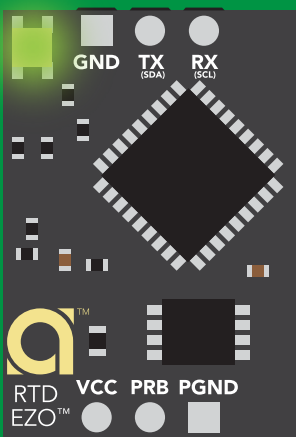
Plock,0 <cr>

*OK <cr>

Plock,? <cr>

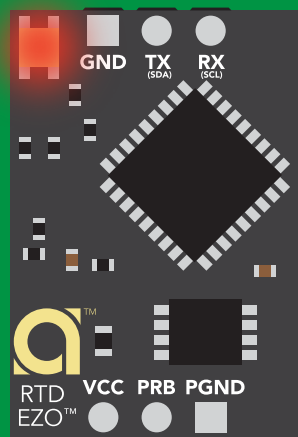
?Plock,1 <cr> or ?Plock,0 <cr>

Plock,1



*OK <cr>

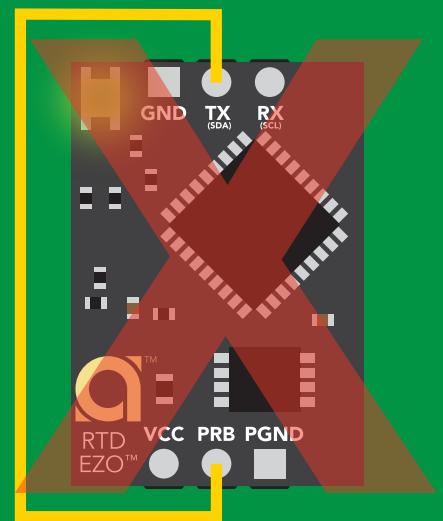
I2C,100



cannot change to I²C

*ER <cr>

Short



cannot change to I²C

Factory reset

Command syntax

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

Factory <cr> enable factory reset

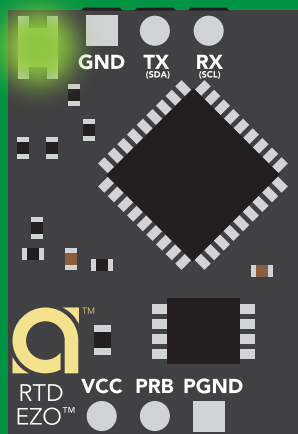
Example

Response

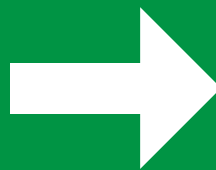
Factory <cr>

*OK <cr>

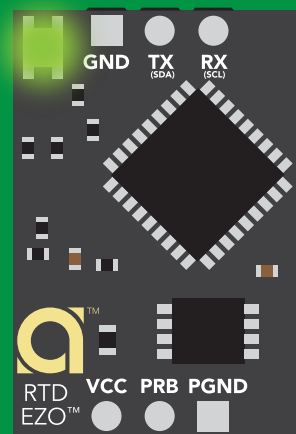
Factory <cr>



*OK <cr>



(reboot)



*RS <cr>

*RE <cr>

Baud rate will not change

Change to I²C mode

Command syntax

I2C,n <cr> switch from UART to I²C

n = any number 1 – 127

Example

I2C,100 <cr>

Response

*OK (reboot in I²C mode)

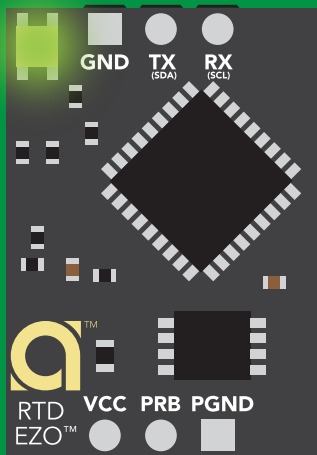
Wrong example

I2C,139 <cr> n ≠ 127

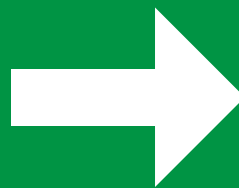
Response

*ER <cr>

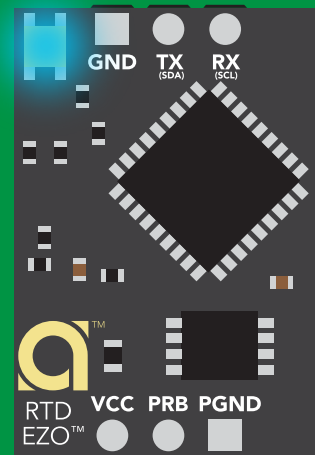
I2C,100



Green
*OK <cr>



(reboot)



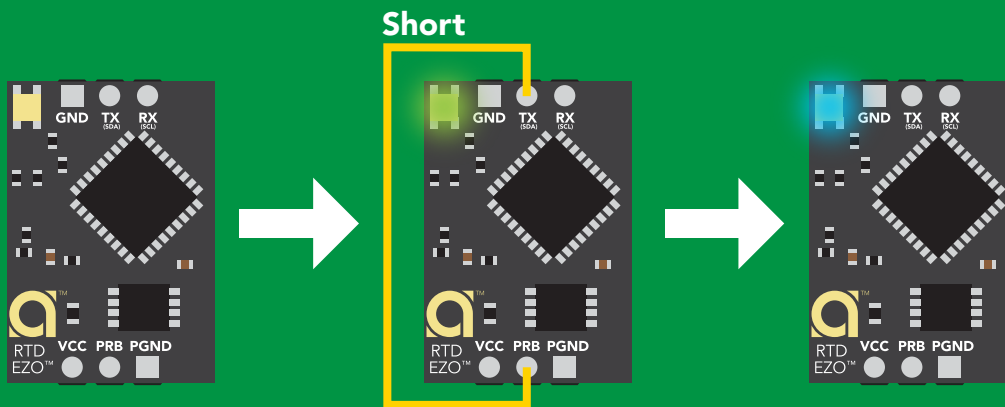
Blue
now in I²C mode

Manual switching to I²C

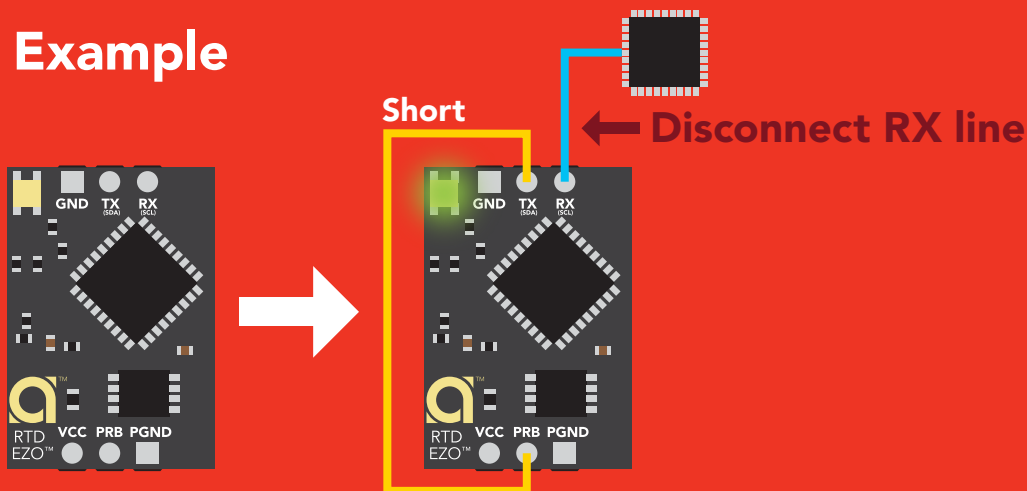
- Make sure Plock is set to 0
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to PRB
- Confirm RX is disconnected
- Connect ground (power on)
- Wait for LED to change from Green to Blue
- Disconnect ground (power off)
- Reconnect all data and power

Manually switching to I²C will set the I²C address to 102 (0x66)

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.

Any settings that have been changed will be retained even if the power has been cut from the device.

To set your EZO™ device into I²C mode [click here](#)

I²C mode

I²C address (0x01 – 0x7F)
0x66 default

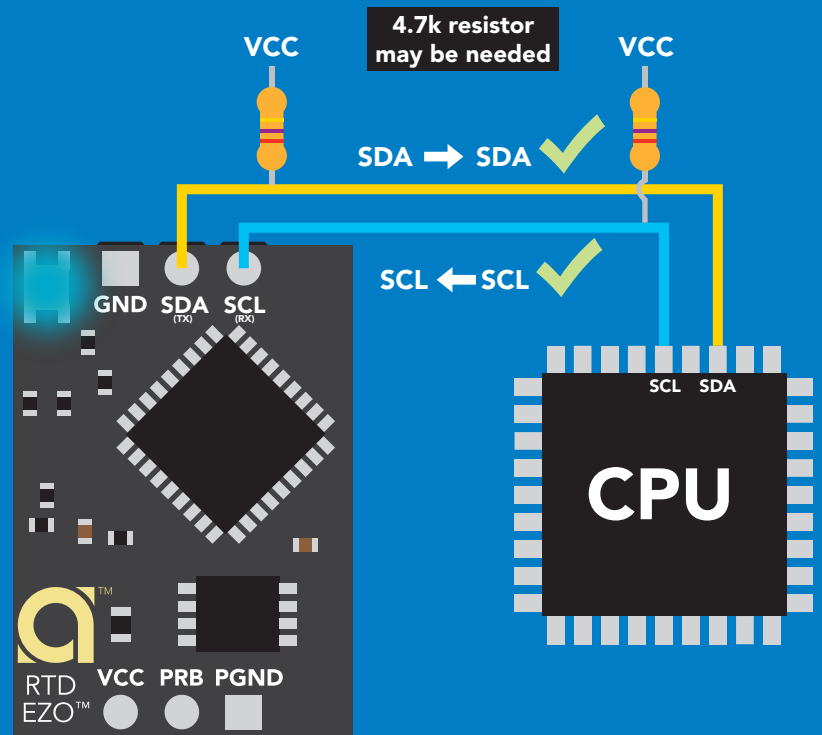
Vcc 3.3V – 5.5V

Clock speed 100 – 400 kHz

SDA 

SCL 


0V VCC 0V



Data format

Reading temperature

Units °C, °K, or °F

Encoding ASCII

Format string

Data type floating point

Decimal places 3

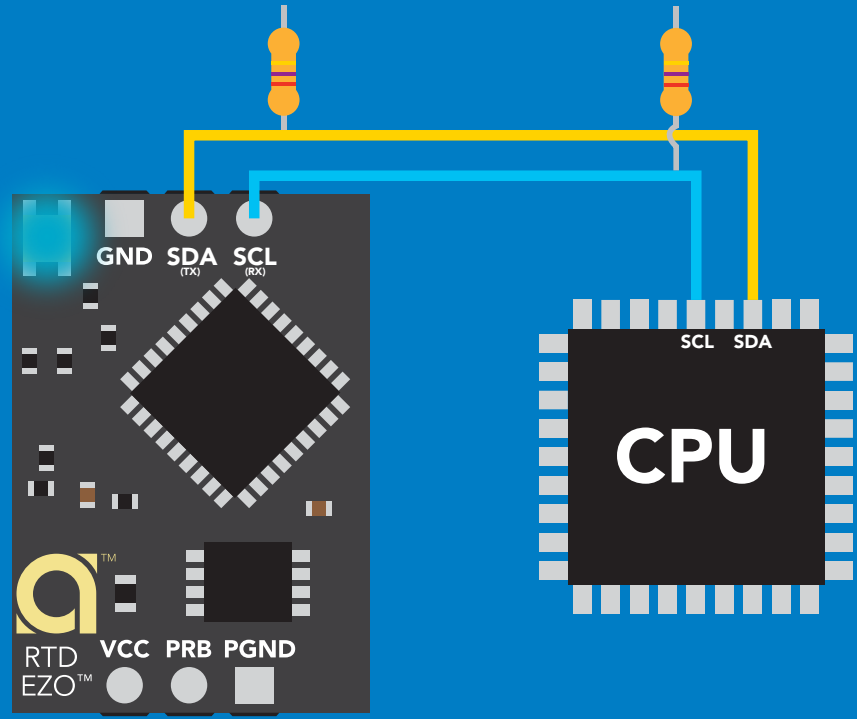
Smallest string 4 characters

Largest string 14 characters

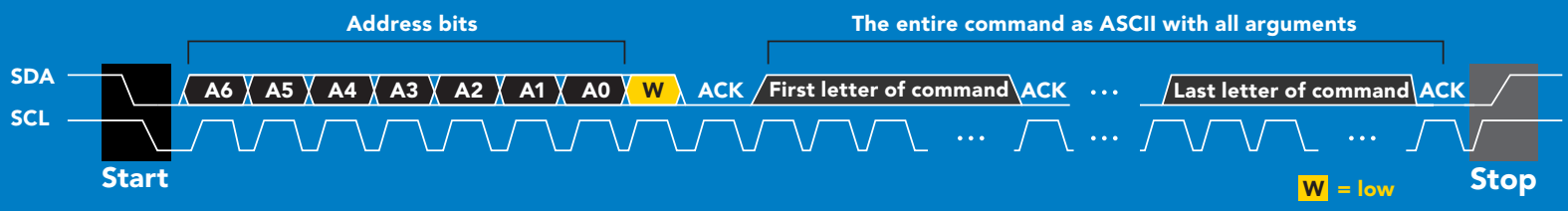
Sending commands to device



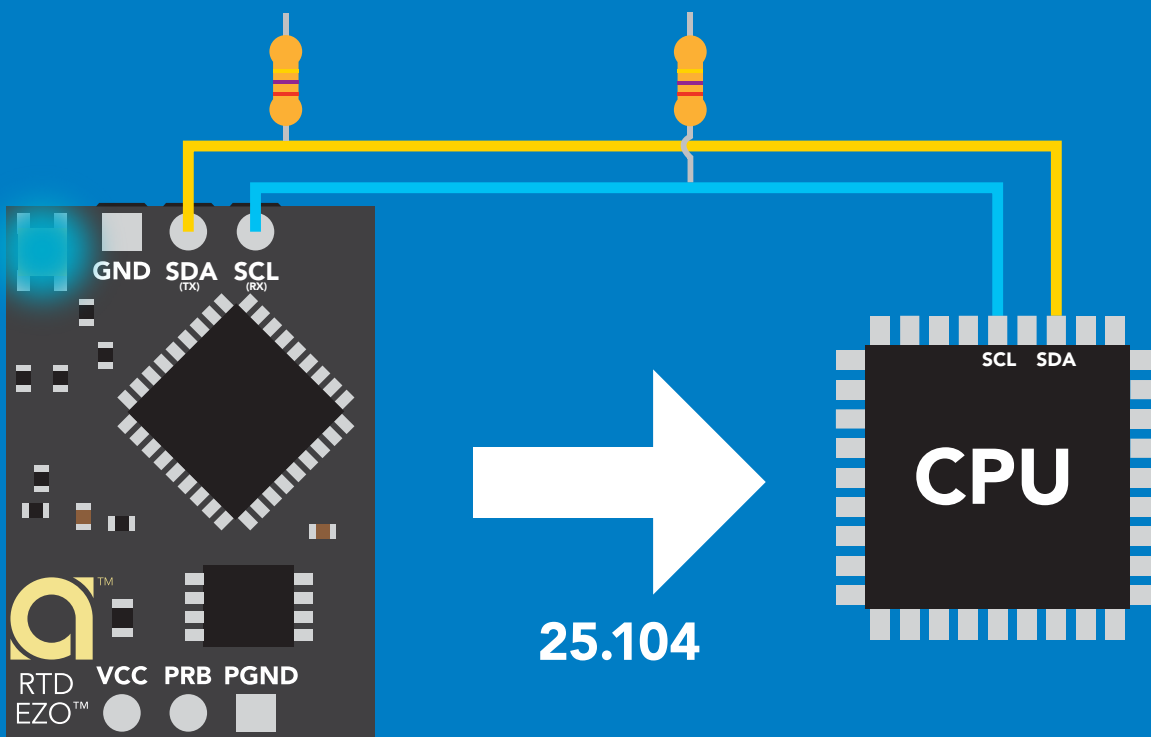
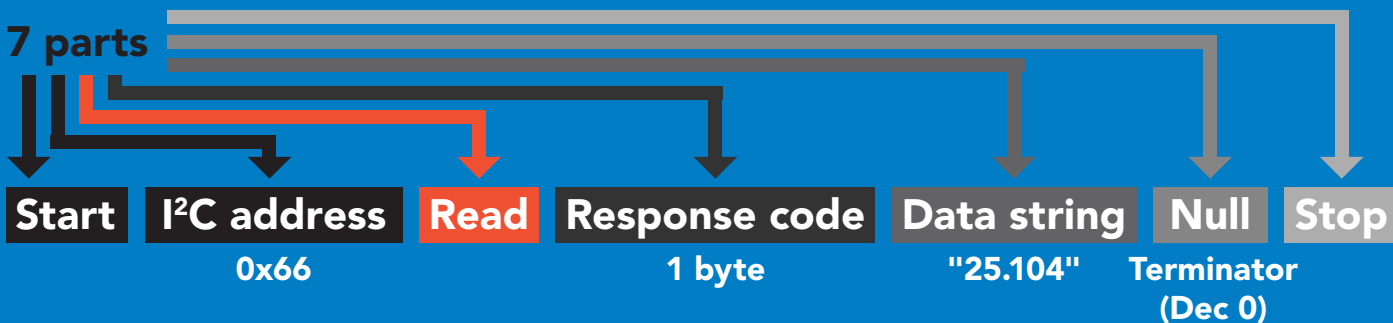
Example



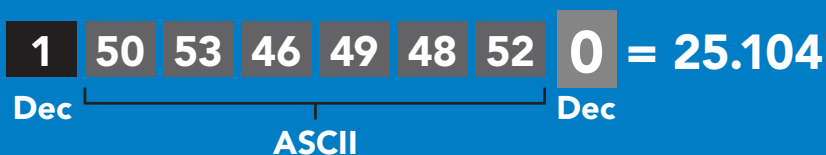
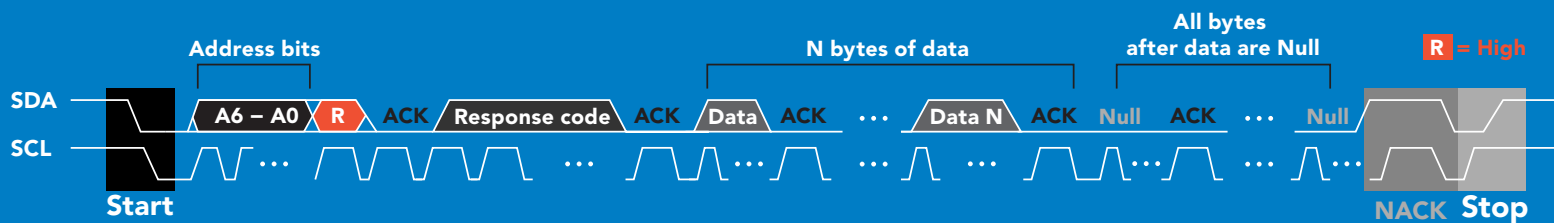
Advanced



Requesting data from device



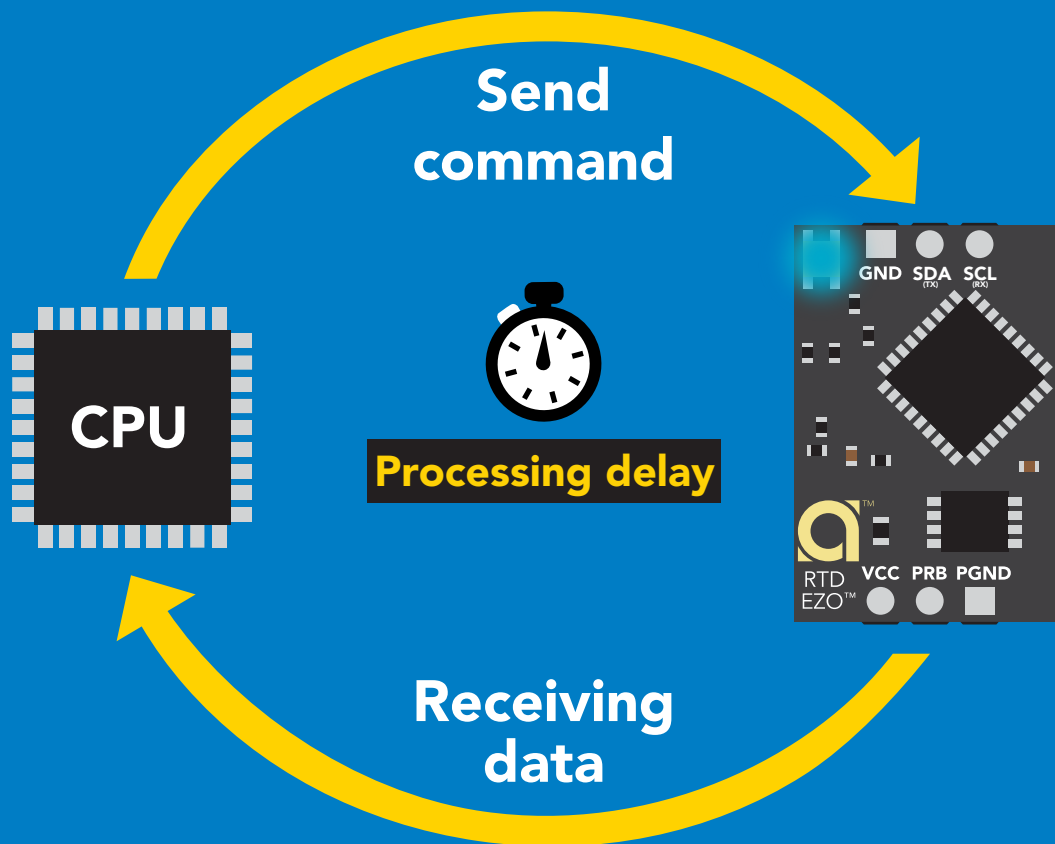
Advanced



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;
```

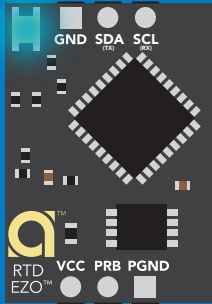
If there is no processing delay or the processing delay is too short, the response code will always be 254.

Response codes

Single byte, not string

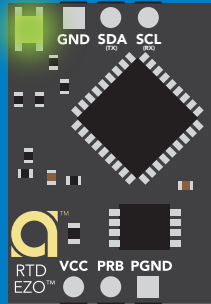
255	no data to send
254	still processing, not ready
2	error
1	successful request

LED color definition



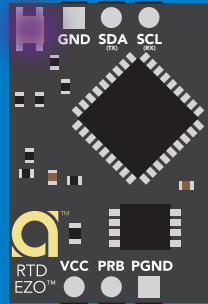
Blue

I²C standby



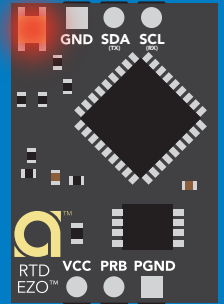
Green

Taking reading



Purple

Changing
I²C ID#



Red

Command
not understood

5V

LED ON
+0.4 mA

3.3V

+0.2 mA

I²C mode

command quick reference

All commands are ASCII strings or single ASCII characters.

Command	Function	
Baud	switch back to UART mode	pg. 58
Cal	performs calibration	pg. 47
D	enable/disable data logger	pg. 49
Factory	enable factory reset	pg. 57
i	device information	pg. 52
I2C	change I ² C address	pg. 56
L	enable/disable/state of LED	pg. 45
M	memory recall/clear	pg. 50
Plock	enable/disable protocol lock	pg. 55
R	returns a single reading	pg. 46
S	temperature scale (°C, °K, °F)	pg. 48
Sleep	enter sleep mode/low power	pg. 54
Status	retrieve status information	pg. 53

LED control

Command syntax

300ms  processing delay

- L,1 LED on **default**
- L,0 LED off
- L,? LED state on/off?

Example

Response

L,1

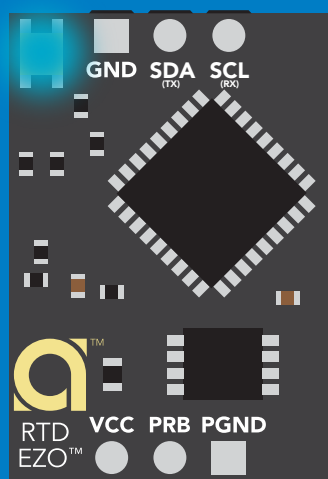
 **Wait 300ms** **1** **0**
Dec Null

L,0

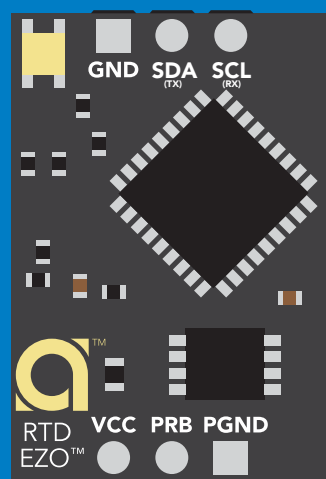
 **Wait 300ms** **1** **0**
Dec Null

L,?

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



L,1



L,0

Taking reading

Command syntax

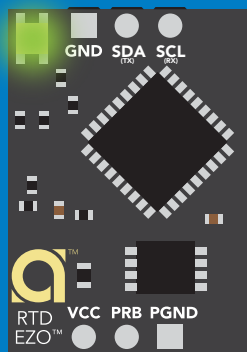
600ms  processing delay

R Return 1 reading

Example

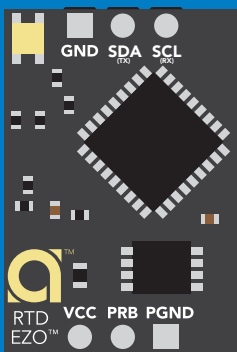
Response

R  **1** **25.104** **0**
Wait 600ms Dec ASCII Null

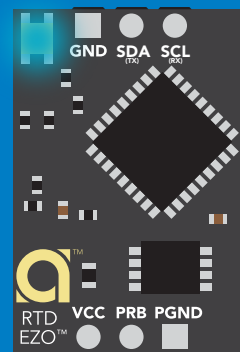
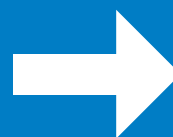


Green

Taking reading



Transmitting



Blue

Standby

Calibration

Command syntax

1000ms  processing delay

Cal,t t = any temperature
Cal,clear delete calibration data
Cal,? device calibrated?

EZO™ RTD circuit uses single point calibration.

Example

Response

Cal,t

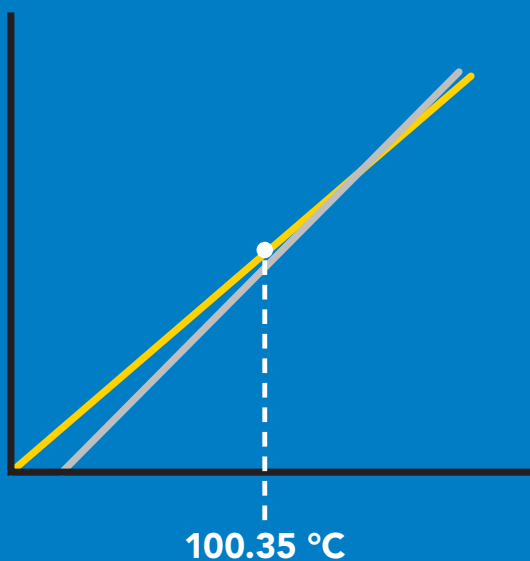
 **Wait 1000ms**
1 **0**
Dec Null

Cal,clear

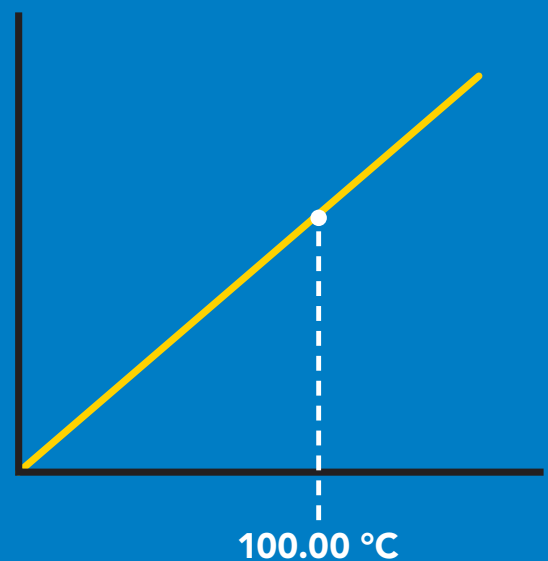
 **Wait 300ms**
1 **0**
Dec Null

Cal,?

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




Cal,100.00



Temperature scale (°C, °K, °F)

Command syntax

300ms  processing delay

S,c celsius **default**

S,k kelvin

S,f fahrenheit

S,? temperature scale?

Example

Response

S,c


Wait 300ms 1 0
Dec Null

S,k

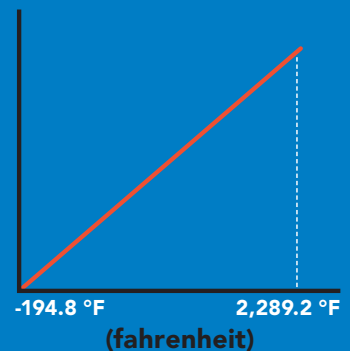
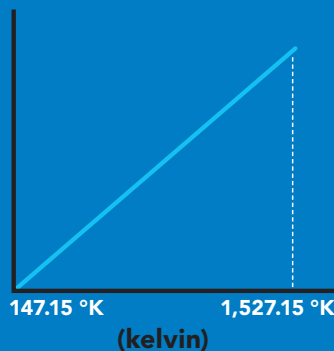
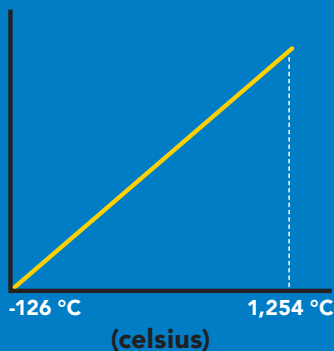

Wait 300ms 1 0
Dec Null

S,f


Wait 300ms 1 0
Dec Null

S,?


Wait 300ms 1 ?S,f 0 or 1 ?S,k 0 or 1 ?S,k 0
Dec ASCII Null Dec ASCII Null Dec ASCII Null



Enable/disable data logger

Command syntax

300ms  processing delay

D,n n = (n x 10 seconds)

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

Response

D,6


Wait 300ms

1	0
Dec	Null

D,0

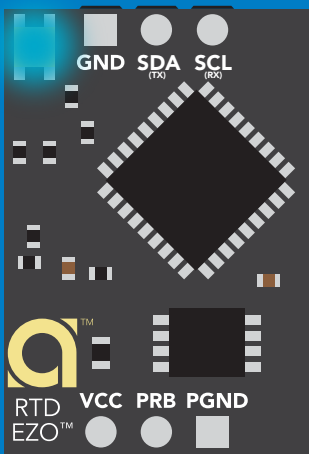

Wait 300ms

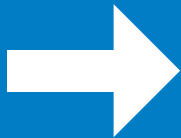
1	0
Dec	Null

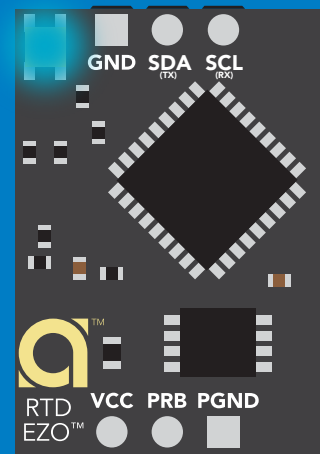
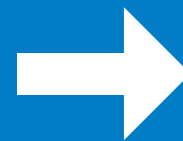
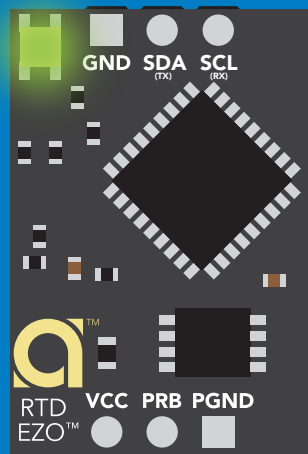
D,?


Wait 300ms

1	?D,6	0
Dec	ASCII	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

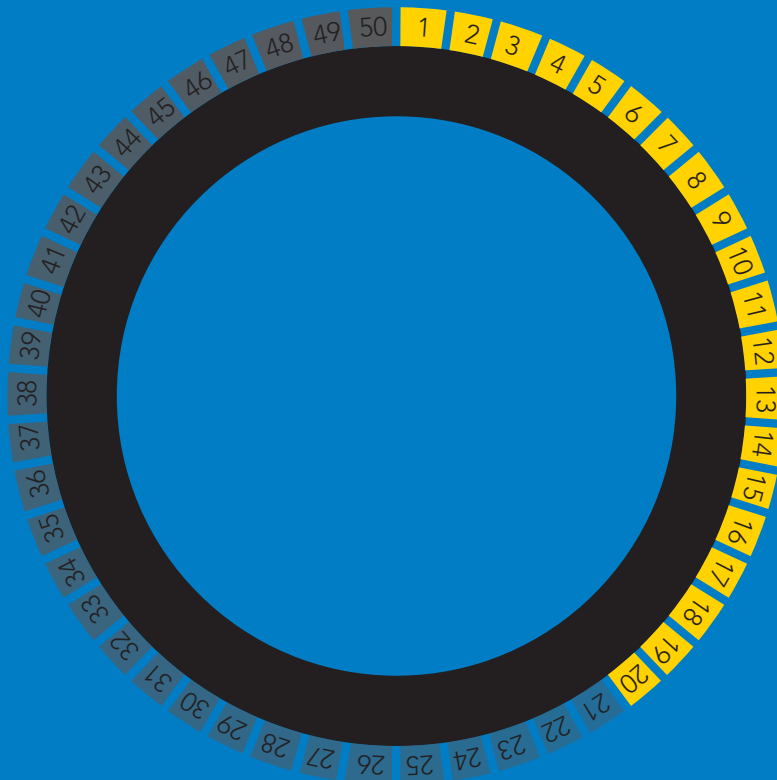
M,? Display memory location of last stored reading

Example

Response

M		1	1,100.00	0
	Wait 300ms	Dec	ASCII	Null

M,?		1	4,112.00	0
	Wait 300ms	Dec	ASCII	Null



Memory clear

Command syntax

300ms  processing delay

M,clear Clear all stored memory

Example

Response

M,clear



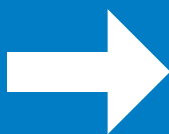
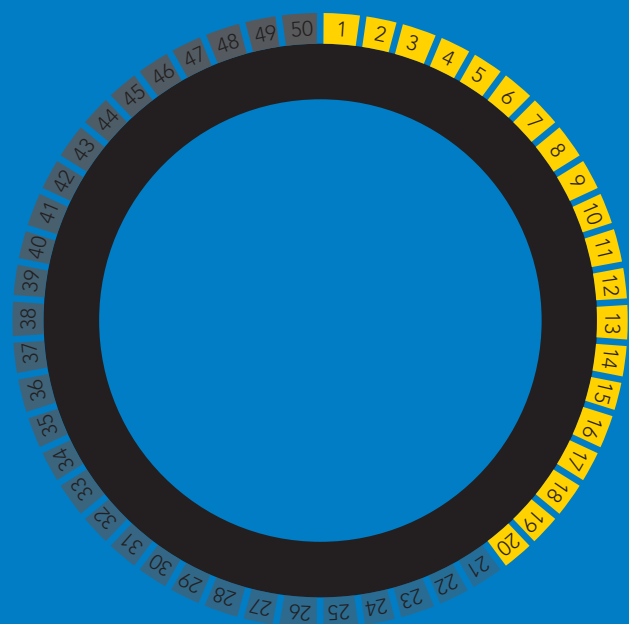
Wait 300ms

1

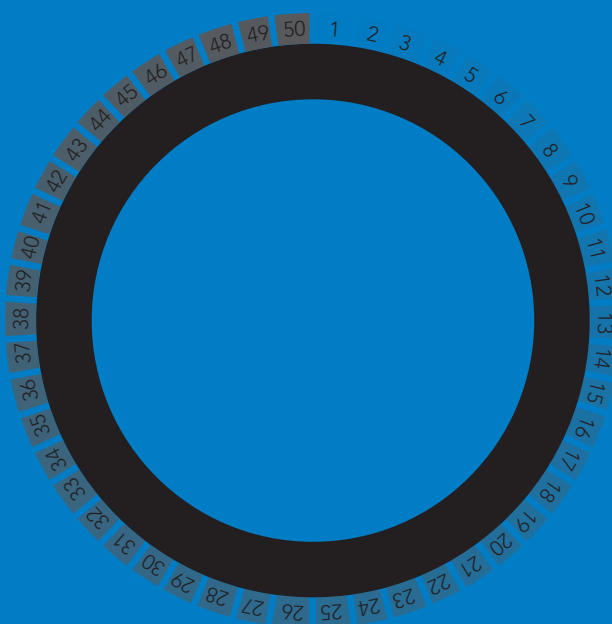
Dec

0

Null



M,clear



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

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

Response breakdown

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

Restart codes

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

Sleep mode/low power

Command syntax

Sleep enter sleep mode/low power

Send any character or command to awaken device.

Example

Response

Sleep

no response

Do not read status byte after issuing sleep command.

Any command

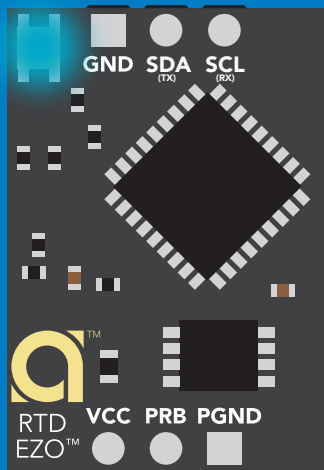
*WA wakes up device

5V

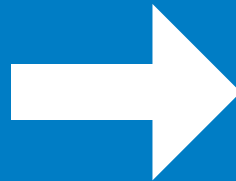
STANDBY	SLEEP
15.40 mA	3.00 mA

3.3V

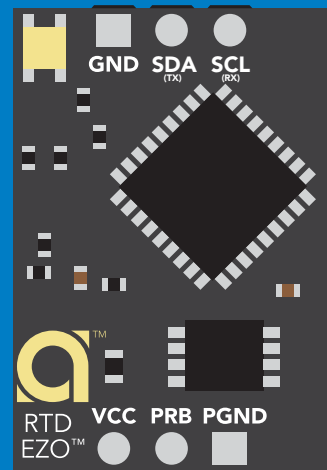
13.80 mA	1.46 mA
----------	---------



Standby



Sleep



Sleep

Protocol lock

Command syntax

300ms  processing delay

- Plock,1 enable Plock
- Plock,0 disable Plock
- Plock,? plock on/off?

Locks device to I²C mode.

Example

Response

Plock,1


Wait 300ms

1	0
Dec	Null

Plock,0


Wait 300ms

1	0
Dec	Null

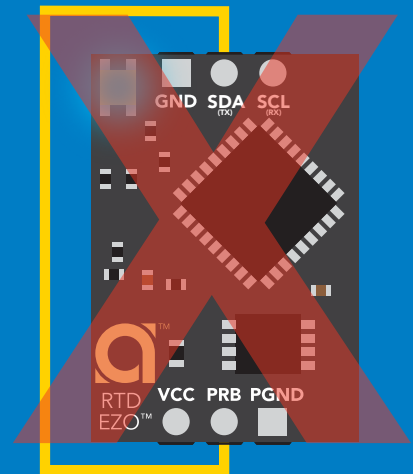
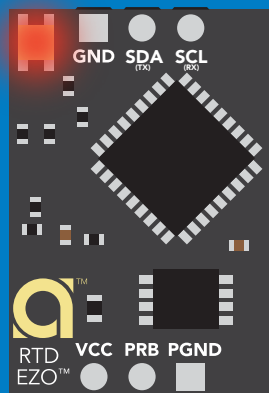
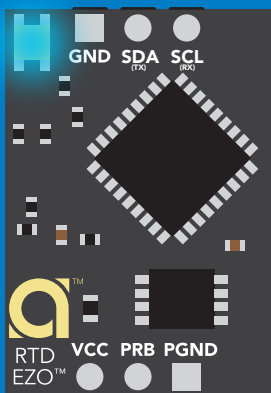
Plock,?


Wait 300ms

1	?Plock,1	0
Dec	ASCII	Null

Plock,1

Serial, 9600



cannot change to UART

cannot change to UART

I²C address change

Command syntax

300ms  processing delay

I2C,n change I²C address

n = any number 1 – 127

Example

Response

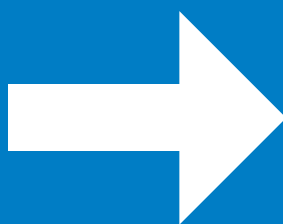
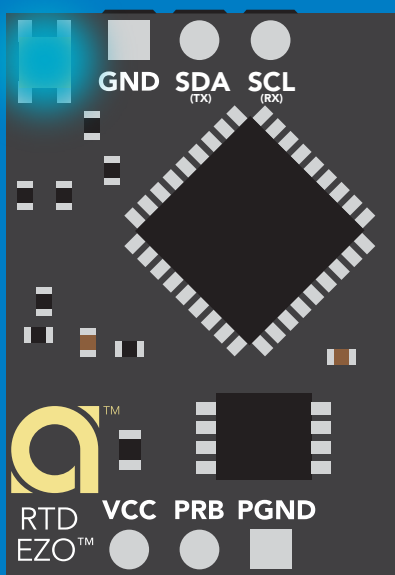
I2C,100

device reboot

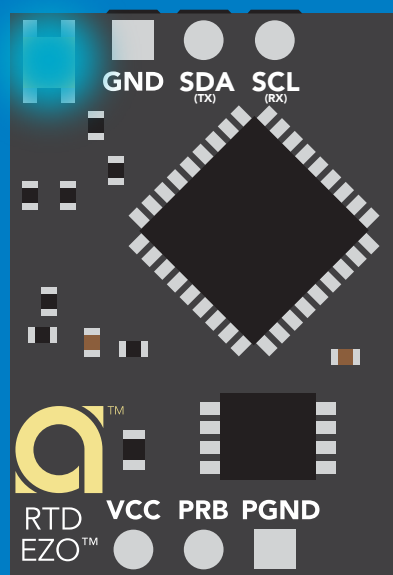
Warning!

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

I2C,100



(reboot)



Factory reset

Command syntax

Factory enable factory reset

I²C address will not change

Example

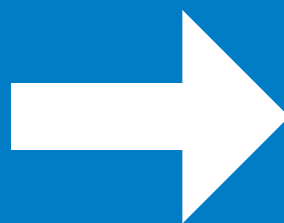
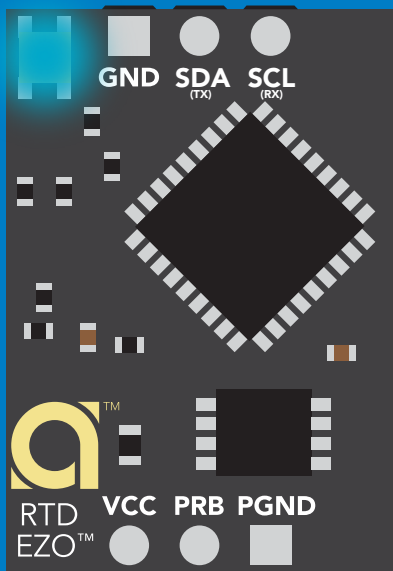
Response

Factory

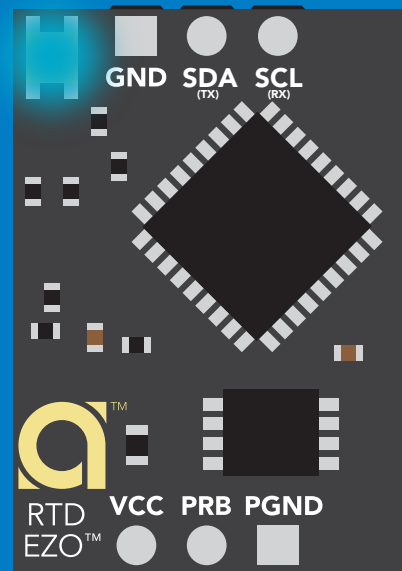
device reboot

Clears calibration
LED on
Response codes enabled
Clears data logger

Factory



(reboot)



Change to UART mode

Command syntax

Baud,n enable factory reset

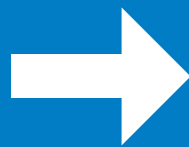
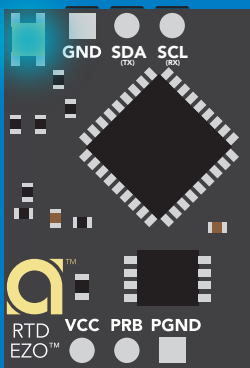
Example

Baud,9600

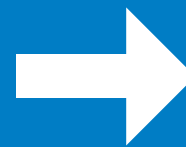
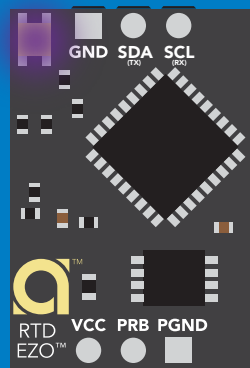
Response

reboot in UART mode

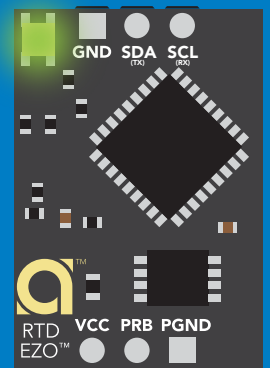
n = [300
1200
2400
9600
19200
38400
57600
115200



Serial,9600

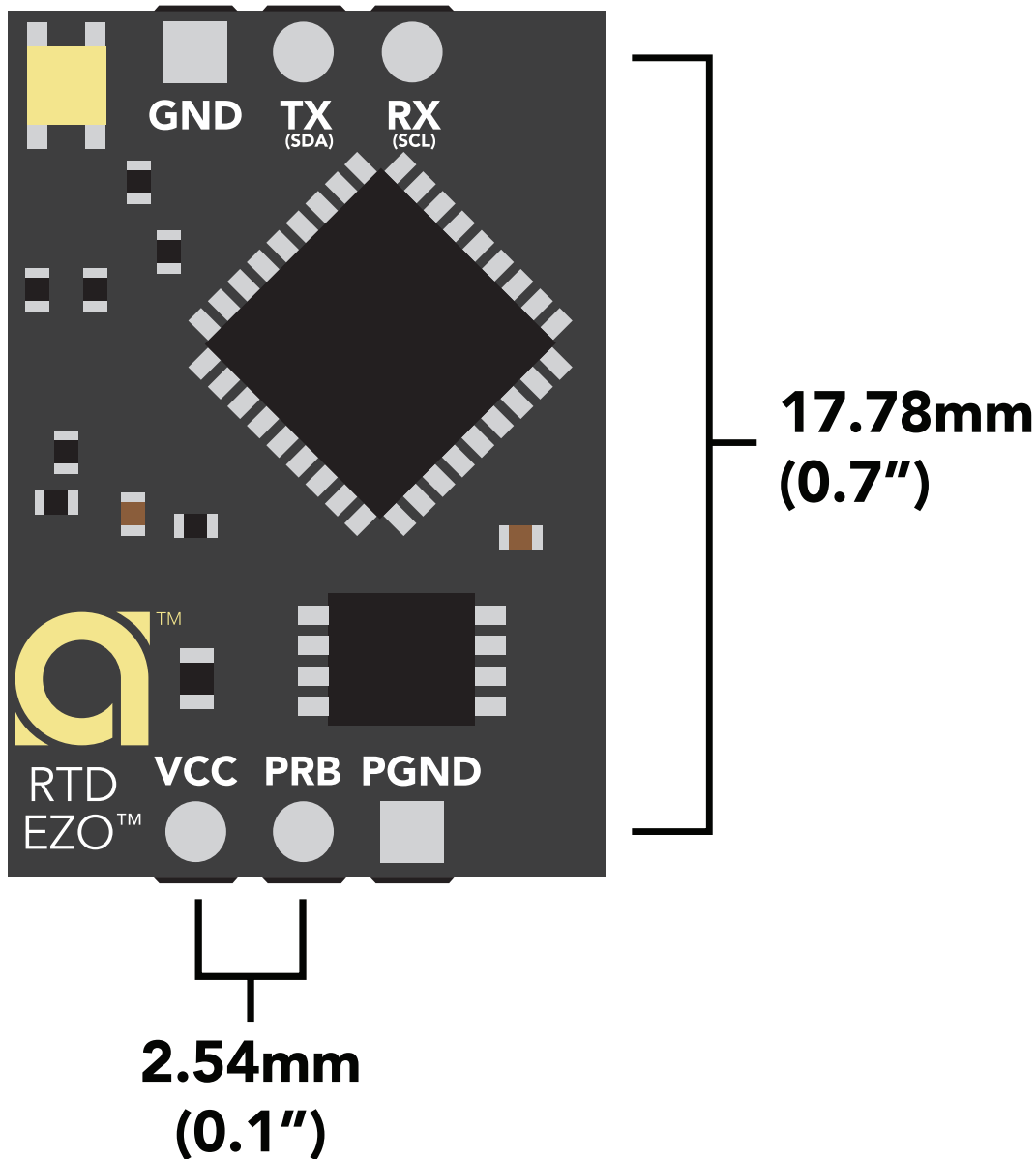


(reboot)



Changing to UART mode

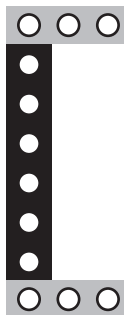
EZO™ circuit footprint



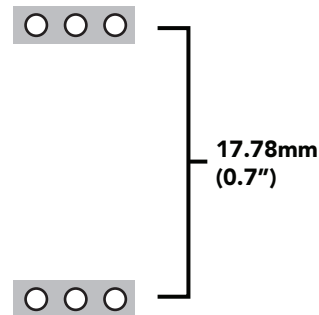
1 In your CAD software place an 8 position header.



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 2.1

Revised entire datasheet

Temperature circuit firmware changes

V1.02 – Plock (March 31, 2016)

- Added protocol lock feature “Plock”

V1.03 – EEPROM (April 26, 2016)

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

V1.11 – Glitch Fix (June 9, 2016)

- Fixed glitch 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”

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.