EZO-EC™
Embedded Conductivity Circuit

Reads
- Conductivity = µS/cm
- Total dissolved solids = ppm
- Salinity = PSU (ppt) 0.00 – 42.00
- Specific gravity (sea water only) = 1.00 – 1.300

Range
- 0.07 – 500,000+ µS/cm

Accuracy
- +/- 2%

Response time
- 1 reading per sec

Supported probes
- K 0.1 – K 10 any brand

Calibration
- 1 or 2 point

Temp compensation
- Yes

Data protocol
- UART & I2C

Default I2C address
- 100 (0x64)

Operating voltage
- 3.3V – 5V

Data format
- ASCII

This is an evolving document, check back for updates.
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
- Power consumption 4
- Absolute max ratings 4
- Conductivity probe range 5
- Resolution 6
- Operating principle 7
- Output units 8
- Power and data isolation 9
- Correct wiring 11
- Calibration theory 12
- Default state 17
- Available data protocols 18

## UART

- UART mode 20
- Receiving data from device 21
- Sending commands to device 22
- LED color definition 23
- UART quick command page 24
- LED control 25
- Find 26
- Continuous reading mode 27
- Single reading mode 28
- Calibration 29
- Export calibration 30
- Import calibration 31
- Setting the probe type 32
- Temperature compensation 33
- Enable/disable parameters 34
- Naming device 35
- Device information 36
- Response codes 37
- Reading device status 38
- Sleep mode/low power 39
- Change baud rate 40
- Protocol lock 41
- Factory reset 42
- Change to I²C mode 43
- Manual switching to I²C 44

## I²C

- I²C mode 46
- Sending commands 47
- Requesting data 48
- Response codes 49
- LED color definition 50
- I²C quick command page 51
- LED control 52
- Find 53
- Taking reading 54
- Calibration 55
- Export calibration 56
- Import calibration 57
- Setting the probe type 58
- Temperature compensation 59
- Enable/disable parameters 60
- Device information 61
- Reading device status 62
- Sleep mode/low power 63
- Protocol lock 64
- I²C address change 65
- Factory reset 66
- Change to UART mode 67
- Manual switching to UART 68

- Circuit footprint 69
- Datasheet change log 70
- Warranty 73
**EZO™ circuit dimensions**

![EZO™ circuit dimensions diagram]

**Power consumption**

<table>
<thead>
<tr>
<th>Voltage</th>
<th>LED</th>
<th>MAX</th>
<th>STANDBY</th>
<th>SLEEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>ON</td>
<td>50 mA</td>
<td>18.14 mA</td>
<td>0.7 mA</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>45 mA</td>
<td>15.64 mA</td>
<td></td>
</tr>
<tr>
<td>3.3V</td>
<td>ON</td>
<td>35 mA</td>
<td>16.85 mA</td>
<td>0.4 mA</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>34 mA</td>
<td>15.85 mA</td>
<td></td>
</tr>
</tbody>
</table>

**Absolute max ratings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>MIN</th>
<th>TYP</th>
<th>MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage temperature (EZO™ Conductivity)</td>
<td>-60 °C</td>
<td>150 °C</td>
<td></td>
</tr>
<tr>
<td>Operational temperature (EZO™ Conductivity)</td>
<td>-40 °C</td>
<td>25 °C</td>
<td>125 °C</td>
</tr>
<tr>
<td>VCC</td>
<td>3.3V</td>
<td>5V</td>
<td>5.5V</td>
</tr>
</tbody>
</table>

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

**K 0.01** → **K 10**

Atlas Scientific™ has tested three different K value probe types:

- **K 0.1**
  - **accurate reading range**: 0.07μS/cm – 50,000μS/cm
  - TDS (ppm) 0 – 25,000
  - Salinity (ppt) 0 – 33

- **K 1.0**
  - **accurate reading range**: 5μS/cm – 200,000+μS/cm
  - TDS (ppm) 2 – 100,000
  - Salinity (ppt) 0 – 42

- **K 10**
  - **accurate reading range**: 10μS/cm – 1S/cm
  - TDS (ppm) 5 – 500,000
  - Salinity (ppt) 0 – 42

*salinity scale cannot go any higher

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

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

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

<table>
<thead>
<tr>
<th>Range</th>
<th>Resolution (μS/cm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.07 – 99.99</td>
<td>0.01</td>
</tr>
<tr>
<td>100.1 – 999.9</td>
<td>0.1</td>
</tr>
<tr>
<td>1,000 – 9,999</td>
<td>1.0</td>
</tr>
<tr>
<td>10,000 – 99,990</td>
<td>10</td>
</tr>
<tr>
<td>100,000 – 999,900</td>
<td>100</td>
</tr>
</tbody>
</table>
Operating principle

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

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

By default, EZO™ Conductivity circuits with firmware version 2.10 and above will only output EC. To enable these parameters see page 34 for UART, and 60 for I²C.

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

- Conductivity = μS/cm
- Total dissolved solids = ppm
- Salinity = PSU (ppt) 0.00 – 42.00
- Specific gravity (sea water only) = 1.00 – 1.300

These parameters must be individually enabled within the device. See page 34 to enable each parameter in UART mode, and on page 60 for I²C mode.

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

Example
EC,TDS,SAL,SG

Default LED blink pattern

This is the LED pattern for Continous Mode (default state)
This can only happen when the device is in UART mode.
Power and data isolation

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

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

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

Without isolation, Conductivity readings will effect each other.
This schematic shows exactly how we isolate data and power using the ADM3260 and a few passive components. The ADM3260 can output isolated power up to 150 mW and incorporates two bidirectional data channels.

This technology works by using tiny transformers to induce the voltage across an air gap. PCB layout requires special attention for EMI/EMC and RF Control, having proper ground planes and keeping the capacitors as close to the chip as possible are crucial for proper performance. The two data channels have a 4.7kΩ pull up resistor on both the isolated and non-isolated lines (R1, R2, R3, and R4) The output voltage is set using a voltage divider (R5, R6, and R,7) this produces a voltage of 3.9V regardless of your input voltage.

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

VCC = 3.0v − 5.5v
Correct wiring

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

Copyright © Atlas Scientific LLC
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, with continuous readings enabled).

Switching the device to I²C mode after calibration will not affect the stored calibration. If the device must be calibrated in I²C mode be sure to continuously request readings so you can see the output from the probe.

1. Pre-calibration setup

Connect the dry conductivity probe and take continuous readings.

2. Set probe type

If your probe ≠ K 1.0 (default), then set the probe type by using the "K,n" command. (where n = K value of your probe) for more information, see page 32 or 58.
3. Dry calibration

Perform a dry calibration using the command "Cal,dry". Even though you may see a reading of 0.00 before issuing the "Cal,dry" command, it is still a necessary part of calibration.

0.00 → “Cal,dry” → 0.00  ✔ Correct

17.00 → “Cal,dry” → 0.00  ✔ Also correct

4. Single point or Two point calibration

No calibration

Single point calibration

Two point calibration

Narrow range of accuracy

Wide range of accuracy

Recommended calibration points

When calibrating, Atlas Scientific recommends using the above µS values. However, you can use any µS values you want.
Two point calibration - low point

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

Trapped air in sensing area (shake to remove)

![Image of probe with trapped air](image)

12,880µS

7,728µS – 18,032µS

+/− 40%

0.00µS

check probe connection, you cannot calibrate to 0.

Once the readings stabilize, issue the low point calibration command: "cal,low,12880" (Readings will NOT change)

Two point calibration - high point

- Rinse off the probe before calibrating to the high point.
- Pour a small amount of the high point calibration solution into a cup.
- Shake the probe to remove trapped air.
- Readings may be off by +/- 40%
- Wait for readings to stabilize.

Once the readings stabilize, issue the high point calibration command: "cal,high,80000" (Readings will change, calibration complete).
Single point calibration

- Pour a small amount of calibration solution into a cup (µS value of your choice).
- Shake the probe to remove trapped air.
- Readings may be off by +/- 40%
- Wait for readings to stabilize.

Once the readings stabilize, issue the single point calibration command. "cal,n" where n = any value. (Readings will change, calibration complete).

Temperature compensation during calibration

Temperature has a significant effect on conductivity readings. The EZO™ Conductivity circuit has its temperature compensation set to 25°C as the default. **At no point should you change the default temperature compensation during calibration.**

If the solution is +/- 5°C (or more), refer to the chart on the bottle, and calibrate to that value.
Temperature compensation example

For this example, we brought the temperature of the solution down to 10°C. Referring to chart on the bottle, you can see the value you should calibrate to is 9,330µS.

Over time, the readings will normalize as the solution warms to 25°C.

See pages 33 or 59 for more information.
Default state

UART mode

Baud
9,600

Readings
continuous

Units
μS/cm

Speed
1 reading per second

LED
on

1,000 ms

Green
Standby

Cyan
Taking reading

Transmitting
Available data protocols

UART

I²C

Unavailable data protocols

SPI
Analog
RS-485
Mod Bus
4–20mA
UART mode

**Settings that are retained if power is cut**

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

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

- Find
- Sleep mode
- Temperature compensation
**UART mode**

<table>
<thead>
<tr>
<th>Baud</th>
<th>300</th>
<th>1,200</th>
<th>2,400</th>
<th><strong>9,600 default</strong></th>
<th>19,200</th>
<th>38,400</th>
<th>57,600</th>
<th>115,200</th>
</tr>
</thead>
</table>

**Data format**

**Reading**

- **Conductivity** = $\mu$S/cm
- **Total dissolved solids** = ppm
- **Salinity** = PSU (ppt) 0.00 – 42.00
- **Specific gravity** (sea water only) = 1.00 – 1.300

**Units**

EC, TDS, SAL, SG

**Terminator**

- carriage return

**Data type**

- floating point

**Decimal places**

- 3

**Smallest string**

- 3 characters

**Largest string**

- 40 characters

**Encoding**

- ASCII

**Format**

- string
Receiving data from device

2 parts

ASCII data string

Command

Carriage return <cr>

Terminator

Advanced

ASCII: 1, 4, 1, 3 <cr>

Hex: 31, 2C, 34, 31, 33, 0D

Dec: 49, 44, 52, 49, 51, 13
Sending commands to device

2 parts

Command (not case sensitive)
ASCII data string

Carriage return <cr>
Terminator

Sender
Sleep <cr>

Receiver

Advanced

ASCII: Sleep <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

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>+2.5 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>+1 mA</td>
</tr>
</tbody>
</table>

LED ON
### UART mode

**command quick reference**

All commands are ASCII strings or single ASCII characters.

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Default state</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud</td>
<td>change baud rate</td>
<td>pg. 40</td>
</tr>
<tr>
<td>C</td>
<td>enable/disable continuous reading</td>
<td>pg. 27</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>pg. 29</td>
</tr>
<tr>
<td>Export</td>
<td>export calibration</td>
<td>pg. 30</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>pg. 42</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>pg. 26</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>pg. 36</td>
</tr>
<tr>
<td>I2C</td>
<td>change to I²C mode</td>
<td>pg. 43</td>
</tr>
<tr>
<td>Import</td>
<td>import calibration</td>
<td>pg. 31</td>
</tr>
<tr>
<td>K</td>
<td>Set probe type</td>
<td>pg. 32</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>pg. 25</td>
</tr>
<tr>
<td>Name</td>
<td>set/show name of device</td>
<td>pg. 35</td>
</tr>
<tr>
<td>O</td>
<td>enable/disable parameters</td>
<td>pg. 34</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>pg. 41</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>pg. 28</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>pg. 39</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>pg. 38</td>
</tr>
<tr>
<td>T</td>
<td>temperature compensation</td>
<td>pg. 33</td>
</tr>
<tr>
<td>*OK</td>
<td>enable/disable response codes</td>
<td>pg. 37</td>
</tr>
</tbody>
</table>
# LED control

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>L,1</td>
<td>LED on default</td>
</tr>
<tr>
<td>L,0</td>
<td>LED off</td>
</tr>
<tr>
<td>L,?</td>
<td>LED state on/off?</td>
</tr>
</tbody>
</table>

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>L,1</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>L,0</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>L,?</td>
<td>?L,1 &lt;cr&gt; or ?L,0 &lt;cr&gt; *OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

---

![Image of LED control](image-url)
Find

Command syntax

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

Example

Response

Find <cr>

*OK <cr>

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

Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C,1</td>
<td>enable continuous readings once per second <strong>default</strong></td>
</tr>
<tr>
<td>C,n</td>
<td>continuous readings every n seconds (n = 2 to 99 sec)</td>
</tr>
<tr>
<td>C,0</td>
<td>disable continuous readings</td>
</tr>
<tr>
<td>C,?</td>
<td>continuous reading mode on/off?</td>
</tr>
</tbody>
</table>

Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>C,1 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>EC,TDS,SAL,SG (1 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>EC,TDS,SAL,SG (2 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>EC,TDS,SAL,SG (3 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td>C,30</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>EC,TDS,SAL,SG (30 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>EC,TDS,SAL,SG (60 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>EC,TDS,SAL,SG (90 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td>C,0</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>C,?</td>
<td>?C,1 &lt;cr&gt; or ?C,0 &lt;cr&gt; or ?C,30 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>
**Single reading mode**

**Command syntax**

R <cr> takes single reading

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
</table>
| R <cr>  | 1,413 <cr> *
|         | OK <cr>  |

- **Green**
  - Standby
- **Cyan**
  - Taking reading
- **Transmitting**

600 ms
**Calibration**

**Command syntax**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,dry</td>
<td>dry calibration</td>
</tr>
<tr>
<td>Cal,n</td>
<td>single point calibration, where n = any value</td>
</tr>
<tr>
<td>Cal,low,n</td>
<td>low end calibration, where n = any value</td>
</tr>
<tr>
<td>Cal,high,n</td>
<td>high end calibration, where n = any value</td>
</tr>
<tr>
<td>Cal,clean</td>
<td>delete calibration data</td>
</tr>
<tr>
<td>Cal,?</td>
<td>device calibrated?</td>
</tr>
</tbody>
</table>

*Dry calibration must always be done first!*

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,dry</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,84</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,low,12880</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,high,80000</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,clean</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,?</td>
<td>?CAL,0 &lt;cr&gt; or ?CAL,1 &lt;cr&gt; or ?CAL,2 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

**One point calibration:**
Step 1. "cal,dry"
Step 2. "cal,n"
Calibration complete!

**Two point calibration:**
Step 1. "cal,dry"
Step 2. "cal,low,n"
Step 3. "cal,high,n"
Calibration complete!
Export calibration

**Command syntax**

Export,? <cr> calibration string info
Export <cr> export calibration string from calibrated device

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export,? &lt;cr&gt;</td>
<td>10,120 &lt;cr&gt;</td>
</tr>
</tbody>
</table>

**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> 10,120 <cr> *DONE

Export: Use this command to download calibration settings

---

Atlas Scientific Environmental Robotics

Copyright © Atlas Scientific LLC
## Import calibration

### Command syntax

| 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>

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

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>K,n &lt;cr&gt;</td>
<td>n = any value; floating point in ASCII</td>
</tr>
<tr>
<td>K,? &lt;cr&gt;</td>
<td>probe K value?</td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>K,10 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>K,? &lt;cr&gt;</td>
<td>?K,10 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

K 1.0 is the default value
Temperature compensation

**Command syntax**

- **T,n**  
  - n = any value; floating point or int
- **T,?**  
  - Compensated temperature value?
- **RT,n**  
  - Set temperature compensation and take a reading*

*This is a new command for firmware V2.13

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

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>T,19.5</td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>
| RT,19.5 | *OK <cr>  
  | 8.91 <cr> |
| T,? | ?T,19.5 <cr>  
  | *OK <cr> |
Enable/disable parameters from output string

**Command syntax**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>O, [parameter],[1,0]</td>
<td>enable or disable output parameter</td>
</tr>
<tr>
<td>O,?</td>
<td>enabled parameter?</td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>O,EC,1</td>
<td>*OK</td>
</tr>
<tr>
<td>/</td>
<td>enable / disable conductivity</td>
</tr>
<tr>
<td>O,EC,0</td>
<td></td>
</tr>
<tr>
<td>O,TDS,1</td>
<td>*OK</td>
</tr>
<tr>
<td>/</td>
<td>enable / disable total dissolved solids</td>
</tr>
<tr>
<td>O,TDS,0</td>
<td></td>
</tr>
<tr>
<td>O,S,1</td>
<td>*OK</td>
</tr>
<tr>
<td>/</td>
<td>enable / disable salinity</td>
</tr>
<tr>
<td>O,S,0</td>
<td></td>
</tr>
<tr>
<td>O,SG,1</td>
<td>*OK</td>
</tr>
<tr>
<td>/</td>
<td>enable / disable specific gravity</td>
</tr>
<tr>
<td>O,SG,0</td>
<td></td>
</tr>
<tr>
<td>O,?</td>
<td></td>
</tr>
<tr>
<td>?</td>
<td>,O,EC,TDS,S,SG</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>conductivity</td>
</tr>
<tr>
<td>TDS</td>
<td>total dissolved solids</td>
</tr>
<tr>
<td>S</td>
<td>salinity</td>
</tr>
<tr>
<td>SG</td>
<td>specific gravity</td>
</tr>
</tbody>
</table>

Followed by 1 or 0

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>enabled</td>
</tr>
<tr>
<td>0</td>
<td>disabled</td>
</tr>
</tbody>
</table>

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

*OK <cr> enable / disable conductivity

*OK <cr> enable / disable total dissolved solids

*OK <cr> enable / disable salinity

*OK <cr> enable / disable specific gravity

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

Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set name</td>
<td>Name,n &lt;cr&gt;</td>
<td>Name,zzt</td>
</tr>
<tr>
<td>Show name</td>
<td>Name,? &lt;cr&gt;</td>
<td>Name,zzt</td>
</tr>
</tbody>
</table>

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Syntax</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set name</td>
<td>Name,zzt</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Show name</td>
<td>Name,?</td>
<td>?Name,zzt  *OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

Up to 16 ASCII characters
# Device information

## Command syntax

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

## Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>i &lt;cr&gt;</td>
<td>?i,EC,2.10 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

## Response breakdown

<table>
<thead>
<tr>
<th>?i,</th>
<th>EC,</th>
<th>2.10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>Firmware</td>
<td></td>
</tr>
</tbody>
</table>
Response codes

Command syntax

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

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>R &lt;cr&gt;</td>
<td>1,413 &lt;cr&gt; *OK &lt;cr&gt;</td>
</tr>
<tr>
<td>*OK,0 &lt;cr&gt;</td>
<td>no response, *OK disabled</td>
</tr>
<tr>
<td>R &lt;cr&gt;</td>
<td>1,413 &lt;cr&gt; *OK disabled</td>
</tr>
<tr>
<td>*OK,? &lt;cr&gt;</td>
<td>?*OK,1 &lt;cr&gt; or ?*OK,0 &lt;cr&gt;</td>
</tr>
</tbody>
</table>

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

| Status <cr> | ?Status,P,5.038 <cr> |

| *OK <cr> |

## Response breakdown

<table>
<thead>
<tr>
<th>?Status,</th>
<th>P,</th>
<th>5.038</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑Reason for restart</td>
<td>↑Voltage at Vcc</td>
<td></td>
</tr>
</tbody>
</table>

## Restart codes

<table>
<thead>
<tr>
<th>P</th>
<th>powered off</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>software reset</td>
</tr>
<tr>
<td>B</td>
<td>brown out</td>
</tr>
<tr>
<td>W</td>
<td>watchdog</td>
</tr>
<tr>
<td>U</td>
<td>unknown</td>
</tr>
</tbody>
</table>
## Sleep mode/low power

### Command syntax

**Sleep** \(<\text{cr}>\)  enter sleep mode/low power

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep (&lt;\text{cr}&gt;)</td>
<td>*OK (&lt;\text{cr}&gt;)</td>
</tr>
<tr>
<td></td>
<td>*SL (&lt;\text{cr}&gt;)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Any command</th>
<th>Response</th>
<th>wakes up device</th>
</tr>
</thead>
<tbody>
<tr>
<td>*WA (&lt;\text{cr}&gt;)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Standby</th>
<th>Sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>18.14 mA</td>
<td>0.7 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>16.85 mA</td>
<td>0.4 mA</td>
</tr>
</tbody>
</table>

---

`Sleep <cr>`

Standby
18.14 mA

Sleep
0.7 mA

---

Copyright © Atlas Scientific LLC

AtlasScientific
Environmental Robotics
**Change baud rate**

**Command syntax**

Baud,n <cr> change baud rate

**Example**

<table>
<thead>
<tr>
<th>Baud,38400 &lt;cr&gt;</th>
<th>*OK &lt;cr&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud,? &lt;cr&gt;</td>
<td>?Baud,38400 &lt;cr&gt; *OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

**n =**

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

Baud,38400 <cr>

Changing baud rate

*OK <cr>

(reboot)

Standby

Changing baud rate

*OK <cr>

Standby
Protocol lock

Command syntax

Plock,1 <cr> enable Plock
Plock,0 <cr> disable Plock default
Plock,? <cr> Plock on/off?

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plock,1 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Plock,0 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Plock,? &lt;cr&gt;</td>
<td>?Plock,1 &lt;cr&gt; or ?Plock,0 &lt;cr&gt;</td>
</tr>
</tbody>
</table>

Example

Plock,1

*OK <cr>

I2C,100
cannot change to I2C

*ER <cr>

Short
cannot change to I2C
Factory reset

Command syntax

Factory <cr> enable factory reset

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factory &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

Factory <cr>

(reboot)

*OK <cr>

*RS <cr>

*RE <cr>

Baud rate will not change
Change to I²C mode

Command syntax

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

n = any number 1 – 127

Example

<table>
<thead>
<tr>
<th>I²C,100  &lt;cr&gt;</th>
<th>*OK (reboot in I²C mode)</th>
</tr>
</thead>
</table>

Wrong example

<table>
<thead>
<tr>
<th>I²C, 139  &lt;cr&gt;</th>
<th>n ≠ 127</th>
<th>*ER  &lt;cr&gt;</th>
</tr>
</thead>
</table>

Example

I²C,100

*OK  <cr>

(reboot)

Green

Blue

now in I²C mode
Manual switching to I²C

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

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

Example

Wrong Example
I²C mode

The I²C protocol is *considerably more complex* than the UART (RS–232) protocol. Atlas Scientific assumes the embedded systems engineer understands this protocol.

To set your EZO™ device into I²C mode click here

<table>
<thead>
<tr>
<th>Settings that are retained if power is cut</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration</td>
</tr>
<tr>
<td>Change I²C address</td>
</tr>
<tr>
<td>Enable/disable parameters</td>
</tr>
<tr>
<td>Hardware switch to UART mode</td>
</tr>
<tr>
<td>LED control</td>
</tr>
<tr>
<td>Protocol lock</td>
</tr>
<tr>
<td>Software switch to UART mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Settings that are <strong>NOT</strong> retained if power is cut</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find</td>
</tr>
<tr>
<td>Sleep mode</td>
</tr>
<tr>
<td>Temperature compensation</td>
</tr>
</tbody>
</table>
I²C mode

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

Vcc  
3.3V – 5.5V

Clock speed  
100 – 400 kHz

SDA  
SCL

Data format

Reading  
Conductivity = μS/cm  
Total dissolved solids = ppm  
Salinity = PSU (ppt) 0.00 – 42.00  
Specific gravity (sea water only) = 1.00 – 1.300

Units  
EC, TDS, SAL, SG

Encoding  
ASCII

Format  
string

Data type  
floating point

Decimal places  
3

Smallest string  
3 characters

Largest string  
40 characters

4.7k resistor may be needed
Sending commands to device

5 parts

- **Start**
- **I²C address**
- **Write**
- **Command (not case sensitive)**
- **Stop**

ASCII command string

Example

**Start** 100 (0x64) **Write** **Sleep** **Stop**

I²C address

Command

Advanced

Address bits

The entire command as ASCII with all arguments

<table>
<thead>
<tr>
<th>SDA</th>
<th>A6</th>
<th>A5</th>
<th>A4</th>
<th>A3</th>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>W</th>
<th>ACK</th>
<th>First letter of command</th>
<th>ACK</th>
<th>Last letter of command</th>
<th>ACK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>W = low</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Stop</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Requesting data from device**

7 parts

- Start
- I²C address
- Read
- Response code
- Data string
- Null
- Stop

- Start: 100 (0x64)
- I²C address
- Read
- Response code: 1 byte
- Data string: "1,413"
- Null
- Terminator: (Dec 0)

**Advanced**

- Address bits
- N bytes of data
- All bytes after data are Null

- SDA
- SCL

- Start

- 1,413

- ASCII: 49, 44, 52, 49, 51

= 1,413
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);
```

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

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

**Response codes**

Single byte, not string

- **255**: no data to send
- **254**: still processing, not ready
- **2**: syntax error
- **1**: successful request
LED color definition

Blue
I²C standby

Green
Taking reading

Purple
Changing I²C address

Red
Command not understood

White
Find

5V
LED ON
+2.5 mA

3.3V
+1 mA
# I²C mode command quick reference

All commands are ASCII strings or single ASCII characters.

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud</td>
<td>switch back to UART mode</td>
<td>67</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>55</td>
</tr>
<tr>
<td>Export</td>
<td>export calibration</td>
<td>56</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>66</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>53</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>61</td>
</tr>
<tr>
<td>I2C</td>
<td>change I²C address</td>
<td>65</td>
</tr>
<tr>
<td>Import</td>
<td>import calibration</td>
<td>57</td>
</tr>
<tr>
<td>K</td>
<td>Set probe type</td>
<td>58</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>52</td>
</tr>
<tr>
<td>O</td>
<td>enable/disable parameters</td>
<td>60</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>64</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>54</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>63</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>62</td>
</tr>
<tr>
<td>T</td>
<td>temperature compensation</td>
<td>59</td>
</tr>
</tbody>
</table>
## LED control

### Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L,1</td>
<td>LED on (default)</td>
</tr>
<tr>
<td>L,0</td>
<td>LED off</td>
</tr>
<tr>
<td>L,?</td>
<td>LED state on/off?</td>
</tr>
</tbody>
</table>

#### 300ms processing delay

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>L,1</td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td>L,0</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td>L,?</td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
</tbody>
</table>

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>L,1</td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td>L,0</td>
<td><img src="image5.png" alt="Image" /></td>
</tr>
<tr>
<td>L,?</td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
</tbody>
</table>

---

Copyright © Atlas Scientific LLC
Find

**Command syntax**

Find  LED rapidly blinks white, used to help find device

**Example**

Find

**Response**

Find

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

Find

300ms processing delay

Send any character or command to terminate find.

Find

LED rapidly blinks white, used to help find device
# Taking reading

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>return 1 reading</td>
<td></td>
</tr>
</tbody>
</table>

## Example

<table>
<thead>
<tr>
<th>R</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait 600ms</td>
<td></td>
</tr>
</tbody>
</table>

## Response

<table>
<thead>
<tr>
<th>Time</th>
<th>Value 1 (Dec)</th>
<th>Value 2 (ASCII)</th>
<th>Value 3 (Null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>600ms</td>
<td>1</td>
<td>1,413</td>
<td>0</td>
</tr>
</tbody>
</table>

---

- **Green**: Taking reading
- **Transmitting**
- **Blue**: Standby

---

**Note**: 600ms processing delay

---

**Copyright © Atlas Scientific LLC**
# Calibration

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,dry</td>
<td>dry calibration</td>
</tr>
<tr>
<td>Cal,n</td>
<td>single point calibration, where ( n = ) any value</td>
</tr>
<tr>
<td>Cal,low,n</td>
<td>low end calibration, where ( n = ) any value</td>
</tr>
<tr>
<td>Cal,high,n</td>
<td>high end calibration, where ( n = ) any value</td>
</tr>
<tr>
<td>Cal,clear</td>
<td>delete calibration data</td>
</tr>
<tr>
<td>Cal,?</td>
<td>device calibrated?</td>
</tr>
</tbody>
</table>

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,dry</td>
<td></td>
</tr>
<tr>
<td>Cal,84</td>
<td></td>
</tr>
<tr>
<td>Cal,low,12880</td>
<td></td>
</tr>
<tr>
<td>Cal,high,80000</td>
<td></td>
</tr>
<tr>
<td>Cal,clear</td>
<td></td>
</tr>
<tr>
<td>Cal,?</td>
<td></td>
</tr>
</tbody>
</table>

## Processing delay

- **600ms**

## Example response

- **One point calibration:**
  - Step 1: "cal,dry"
  - Step 2: "cal,n"
  - **Calibration complete!**

- **Two point calibration:**
  - Step 1: "cal,dry"
  - Step 2: "cal,low,n"
  - Step 3: "cal,high,n"
  - **Calibration complete!**

---

Copyright © Atlas Scientific LLC
## Export calibration

### Command syntax

**Export,?** calibration string info

**Export** export calibration string from calibrated device

### Example

#### Export,?

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait 300ms</td>
<td>1</td>
</tr>
</tbody>
</table>

**Response breakdown**

- **10** strings to export
- **120** bytes to export

Export strings can be up to 12 characters long

#### Export

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait 300ms</td>
<td>1</td>
</tr>
</tbody>
</table>

(1 of 10)

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait 300ms</td>
<td>1</td>
</tr>
</tbody>
</table>

(2 of 10)

(7 more)

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait 300ms</td>
<td>1</td>
</tr>
</tbody>
</table>

(10 of 10)

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait 300ms</td>
<td>1</td>
</tr>
</tbody>
</table>

---

56 Copyright © Atlas Scientific LLC
**Import calibration**

**Command syntax**

Import, n

import calibration string to new device

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import, 59 6F 75 20 61 72</td>
<td>(1 of 10)</td>
</tr>
<tr>
<td>Import, 65 20 61 20 63 6F</td>
<td>(2 of 10)</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Import, 6F 6C 20 67 75 79</td>
<td>(10 of 10)</td>
</tr>
</tbody>
</table>

**Response**

- Wait 300ms
- Dec 1
- Null 0

*If one of the imported strings is not correctly entered, the device will not accept the import and reboot.*
### Setting the probe type

#### Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>K,n</td>
<td><code>n</code> = any value; floating point in ASCII</td>
</tr>
<tr>
<td>K,?</td>
<td>probe K value?</td>
</tr>
</tbody>
</table>

#### Example and Response

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>K,10</td>
<td>1,0 Dec Null Wait 300ms</td>
</tr>
<tr>
<td>K,?</td>
<td>1, K,10 ASCII 0 Dec Null Wait 600ms</td>
</tr>
</tbody>
</table>

**K 1.0 is the default value**
# Temperature compensation

## Command syntax

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

* This is a new command for firmware V2.13

---

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T,19.5$</td>
<td><img src="https://via.placeholder.com/15" alt="Wait 300ms" /> 1 Dec 0 Null</td>
</tr>
<tr>
<td>$RT,19.5$</td>
<td><img src="https://via.placeholder.com/15" alt="Wait 900ms" /> 1 Dec 8.91 ASCII 0 Null</td>
</tr>
<tr>
<td>$T,?$</td>
<td><img src="https://via.placeholder.com/15" alt="Wait 300ms" /> 1 Dec $?T,19.5$ 0 Null</td>
</tr>
</tbody>
</table>

---

**Default temperature = 25°C**

**Temperature is always in Celsius**

**Temperature is not retained if power is cut**

---

**300ms processing delay**

---

---

---

---
## Enable/disable parameters from output string

### Command syntax

<table>
<thead>
<tr>
<th>Command Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>O, [parameter].[1,0]</td>
<td>enable or disable output parameter</td>
</tr>
<tr>
<td>O,?</td>
<td>enabled parameter?</td>
</tr>
</tbody>
</table>

### Example

<table>
<thead>
<tr>
<th>Command Syntax</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>O,EC,1 / O,EC,0</td>
<td>O,EC,1</td>
</tr>
<tr>
<td>O,TDS,1 / O,TDS,0</td>
<td>O,TDS,1</td>
</tr>
<tr>
<td>O,S,1 / O,S,0</td>
<td>O,S,1</td>
</tr>
<tr>
<td>O,SG,1 / O,SG,0</td>
<td>O,SG,1</td>
</tr>
<tr>
<td>O,?</td>
<td>O,?</td>
</tr>
</tbody>
</table>

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>conductivity</td>
</tr>
<tr>
<td>TDS</td>
<td>total dissolved solids</td>
</tr>
<tr>
<td>S</td>
<td>salinity</td>
</tr>
<tr>
<td>SG</td>
<td>specific gravity</td>
</tr>
</tbody>
</table>

Followed by 1 or 0

1 enabled
0 disabled

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

---

300ms processing delay
## Device information

### Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>device information</td>
</tr>
</tbody>
</table>

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>Wait 300ms</td>
</tr>
</tbody>
</table>

### Response breakdown

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?i, EC, 2.10</td>
<td>Device Firmware</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>1</td>
</tr>
<tr>
<td>?i,EC, 2.10</td>
<td>ASCII</td>
</tr>
<tr>
<td>0</td>
<td>Null</td>
</tr>
</tbody>
</table>

### Processing delay

- 300ms
# Reading device status

## Command syntax

<table>
<thead>
<tr>
<th>Status</th>
<th>voltage at Vcc pin and reason for last restart</th>
</tr>
</thead>
</table>

## Example

<table>
<thead>
<tr>
<th>Status</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>?Status,P,5.038</td>
<td>0, Null</td>
</tr>
</tbody>
</table>

## Response breakdown

```
?Status, P, 5.038
```

- **?Status**: Type of status request
- **P**: Reason for restart
- **5.038**: Voltage at Vcc

## Restart codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>powered off</td>
</tr>
<tr>
<td>S</td>
<td>software reset</td>
</tr>
<tr>
<td>B</td>
<td>brown out</td>
</tr>
<tr>
<td>W</td>
<td>watchdog</td>
</tr>
<tr>
<td>U</td>
<td>unknown</td>
</tr>
</tbody>
</table>

---

300ms processing delay

Copyright © Atlas Scientific LLC
# Sleep mode/low power

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
</tr>
</tbody>
</table>

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep</td>
<td>no response</td>
</tr>
<tr>
<td>Any command</td>
<td>wakes up device</td>
</tr>
</tbody>
</table>

### Response

- Send any character or command to awaken device.
- Do not read status byte after issuing sleep command.

### Table

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Standby</th>
<th>Sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>18.14 mA</td>
<td>0.7 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>16.85 mA</td>
<td>0.4 mA</td>
</tr>
</tbody>
</table>

## Diagram

- **Standby**
- **Sleep**

---

Copyright © Atlas Scientific LLC
## Protocol lock

### Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plock, 1</td>
<td>enable Plock</td>
</tr>
<tr>
<td>Plock, 0</td>
<td>disable Plock</td>
</tr>
<tr>
<td>Plock, ?</td>
<td>Plock on/off?</td>
</tr>
</tbody>
</table>

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
</table>
| **Plock, 1** | 🕒 Wait 300ms  
  🔄 Dec 0  
  🔄 ASCII Null |
| **Plock, 0** | 🕒 Wait 300ms  
  🔄 Dec 0  
  🔄 ASCII Null |
| **Plock, ?** | 🕒 Wait 300ms  
  🔄 Dec ?Plock,1 0  
  🔄 ASCII Null |

### Example diagrams

- **Plock, 1**: Cannot change to UART
- **Baud, 9600**: Cannot change to UART
**I²C address change**

**Command syntax**

I²C,n sets I²C address and reboots into I²C mode

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>I²C,101</td>
<td>device reboot</td>
</tr>
</tbody>
</table>

**Warning!**

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

Default I²C address is 100 (0x64).

n = any number 1 – 127
Factory reset

Command syntax

Factory enable factory reset

I²C address will not change

Example

<table>
<thead>
<tr>
<th>Factory</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factory</td>
<td>device reboot</td>
</tr>
</tbody>
</table>

Clears calibration
LED on
Response codes enabled

Factory

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

(reboot)
Change to UART mode

Command syntax

Baud, n  switch from I²C to UART

Example

| Baud, 9600 | reboot in UART mode |

Response

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

Example:

- Baud, 9600
  - Changing to UART mode
  - reboot in UART mode

Response:

- Command not understood
Manual switching to UART

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

Example

Wrong Example

Disconnect RX line
In your CAD software, place a 8 position header. Place a 3 position header at both top and bottom of the 8 position. 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 6.0
Changed the K value range from 0.1 to 0.01 on pg 5.

Datasheet V 5.9
Moved Default state to pg 17.

Datasheet V 5.8
Revised conductivity probe range information on pg 5.

Datasheet V 5.7
Revised response for the sleep command in UART mode on pg 39.

Datasheet V 5.6
Added more information on the Export calibration and Import calibration commands.

Datasheet V 5.5
Revised calibration theory pages, added information on temperature compensation on pg. 15, moved data isolation to pg 9, and correct wiring to pg 11.

Datasheet V 5.4
Revised isolation schematic on pg. 13

Datasheet V 5.3
Added new command:
"RT,n" for Temperature compensation located on pages 30 (UART) & 55 (I²C).
Added firmware information to Firmware update list.

Datasheet V 5.2
Revised calibration information on pages 27 & 52.

Datasheet V 5.1
Added more information about temperature compensation on pages 30 & 55.
Datasheet change log

Datasheet V 5.0
Changed "Max rate" to "Response time" on cover page.

Datasheet V 4.9
Removed note from certain commands about firmware version. Added steps to calibration command pages 27 (UART) and 52 (I²C).

Datasheet V 4.8
Revised definition of response codes on pg 46.

Datasheet V 4.7
Revised cover page art.

Datasheet V 4.6
Updated calibration processing delay time on pg.52.

Datasheet V 4.5
Revised Enable/disable parameters information on pages 31 & 56.

Datasheet V 4.4
Updated High point calibration info on page 11.

Datasheet V 4.3
Updated calibration info on pages 27 (UART) and 52 (I²C).

Datasheet V 4.2
Revised Plock pages to show default value.

Datasheet V 4.1
Corrected I²C calibration delay on pg. 52.

Datasheet V 4.0
Revised entire datasheet.
# Firmware updates

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

**V1.1** – (June 2, 2014)
- Change specific gravity equation to return 1.0 when the uS reading is < 1000 (previously returned 0.0)
- Change accuracy of specific gravity from 2 decimal places to 3 decimal places
- Don’t save temperature changes to EEPROM

**V1.2** – (Aug 1, 2014)
- Baud rate change is now a long, purple blink

**V1.5** – Baud rate change (Nov 6, 2014)
- Change default baud rate to 9600

**V1.6** – I2C bug (Dec 1, 2014)
- Fixed I2C bug where the circuit may inappropriately respond when other I2C devices are connected.

**V1.8** – Factory (April 14, 2015)
- Changed “X” command to “Factory”

**V1.95** – Plock (March 31, 2016)
- Added protocol lock feature “Plock”

**V1.96** – EEPROM (April 26, 2016)
- Fixed bug where EEPROM would get erased if the circuit lost power 900ms into startup

**V2.10** – (April 12, 2017)
- Added "Find" command.
- Added "Export/import" command.
- Modified continuous mode to be able to send readings every "n" seconds.
- Default output changed from CSV string of 4 values to just conductivity; Other values must be enabled.

**V2.11** – (April 28, 2017)
- Fixed "Sleep" bug, where it would draw excessive current.

**V2.12** – (May 9, 2017)
- Fixed bug in sleep mode, where circuit would wake up to a different I2C address.

**V2.13** – (July 16, 2018)
- Added “RT” command to Temperature compensation.

**V2.14** – (Nov 26, 2019)
- The K value range has been extended to 0.01.
Warranty

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

The debugging phase

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

It is important for the embedded systems engineer to keep in mind that the following activities will void the EZO™ class Conductivity circuit warranty:

• Soldering any part of the EZO™ class Conductivity circuit.
• Running any code, that does not exclusively drive the EZO™ class Conductivity circuit and output its data in a serial string.
• Embedding the EZO™ class Conductivity circuit into a custom made device.
• Removing any potting compound.
Reasoning behind this warranty

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

Please keep this in mind:

1. All Atlas Scientific™ devices have been designed to be embedded into a custom made system by you, the embedded systems engineer.

2. All Atlas Scientific™ devices have been designed to run indefinitely without failure in the field.

3. All Atlas Scientific™ devices can be soldered into place, however you do so at your own risk.

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