Conductivity

**EZO™**

**Circuit**

**Reads**

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

**Range**

0.07 – 500,000+ µS/cm

**Accuracy**

+/- 2%

**Max rate**

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.

Written by Jordan Press
Designed by Noah Press

PATENT PROTECTED

Revised 2/16/18

---
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

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit dimensions</td>
<td>4</td>
</tr>
<tr>
<td>Power consumption</td>
<td>4</td>
</tr>
<tr>
<td>Absolute max ratings</td>
<td>4</td>
</tr>
<tr>
<td>EZO™ circuit identification</td>
<td>5</td>
</tr>
<tr>
<td>Conductivity probe range</td>
<td>6</td>
</tr>
<tr>
<td>Resolution</td>
<td>7</td>
</tr>
<tr>
<td>Operating principle</td>
<td>8</td>
</tr>
<tr>
<td>Output units</td>
<td>9</td>
</tr>
<tr>
<td>Calibration theory</td>
<td>10</td>
</tr>
<tr>
<td>Power and data isolation</td>
<td>12</td>
</tr>
<tr>
<td>Correct wiring</td>
<td>14</td>
</tr>
<tr>
<td>Available data protocols</td>
<td>15</td>
</tr>
<tr>
<td>UART mode</td>
<td>17</td>
</tr>
<tr>
<td>Default state</td>
<td>18</td>
</tr>
<tr>
<td>Receiving data from device</td>
<td>19</td>
</tr>
<tr>
<td>Sending commands to device</td>
<td>20</td>
</tr>
<tr>
<td>LED color definition</td>
<td>21</td>
</tr>
<tr>
<td>UART quick command page</td>
<td>22</td>
</tr>
<tr>
<td>LED control</td>
<td>23</td>
</tr>
<tr>
<td>Find</td>
<td>24</td>
</tr>
<tr>
<td>Continuous reading mode</td>
<td>25</td>
</tr>
<tr>
<td>Single reading mode</td>
<td>26</td>
</tr>
<tr>
<td>Calibration</td>
<td>27</td>
</tr>
<tr>
<td>Export/import calibration</td>
<td>28</td>
</tr>
<tr>
<td>Setting the probe type</td>
<td>29</td>
</tr>
<tr>
<td>Temperature compensation</td>
<td>30</td>
</tr>
<tr>
<td>Enable/disable parameters</td>
<td>31</td>
</tr>
<tr>
<td>Naming device</td>
<td>32</td>
</tr>
<tr>
<td>Device information</td>
<td>33</td>
</tr>
<tr>
<td>Response codes</td>
<td>34</td>
</tr>
<tr>
<td>Reading device status</td>
<td>35</td>
</tr>
<tr>
<td>Sleep mode/low power</td>
<td>36</td>
</tr>
<tr>
<td>Change baud rate</td>
<td>37</td>
</tr>
<tr>
<td>Protocol lock</td>
<td>38</td>
</tr>
<tr>
<td>Factory reset</td>
<td>39</td>
</tr>
<tr>
<td>Change to I²C mode</td>
<td>40</td>
</tr>
<tr>
<td>Manual switching to UART</td>
<td>41</td>
</tr>
<tr>
<td>I²C mode</td>
<td>43</td>
</tr>
<tr>
<td>Sending commands</td>
<td>44</td>
</tr>
<tr>
<td>Requesting data</td>
<td>45</td>
</tr>
<tr>
<td>Response codes</td>
<td>46</td>
</tr>
<tr>
<td>LED color definition</td>
<td>47</td>
</tr>
<tr>
<td>I²C quick command page</td>
<td>48</td>
</tr>
<tr>
<td>LED control</td>
<td>49</td>
</tr>
<tr>
<td>Find</td>
<td>50</td>
</tr>
<tr>
<td>Taking reading</td>
<td>51</td>
</tr>
<tr>
<td>Calibration</td>
<td>52</td>
</tr>
<tr>
<td>Export/import calibration</td>
<td>53</td>
</tr>
<tr>
<td>Setting the probe type</td>
<td>54</td>
</tr>
<tr>
<td>Temperature compensation</td>
<td>55</td>
</tr>
<tr>
<td>Enable/disable parameters</td>
<td>56</td>
</tr>
<tr>
<td>Device information</td>
<td>57</td>
</tr>
<tr>
<td>Reading device status</td>
<td>58</td>
</tr>
<tr>
<td>Sleep mode/low power</td>
<td>59</td>
</tr>
<tr>
<td>Protocol lock</td>
<td>60</td>
</tr>
<tr>
<td>I²C address change</td>
<td>61</td>
</tr>
<tr>
<td>Factory reset</td>
<td>62</td>
</tr>
<tr>
<td>Change to UART mode</td>
<td>63</td>
</tr>
<tr>
<td>Manual switching to UART</td>
<td>64</td>
</tr>
<tr>
<td>Circuit footprint</td>
<td>65</td>
</tr>
<tr>
<td>Datasheet change log</td>
<td>66</td>
</tr>
<tr>
<td>Warranty</td>
<td>68</td>
</tr>
</tbody>
</table>
EZO™ circuit dimensions

Power consumption

<table>
<thead>
<tr>
<th>5V</th>
<th>LED</th>
<th>MAX</th>
<th>STANDBY</th>
<th>SLEEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>50 mA</td>
<td>18.14 mA</td>
<td>0.7 mA</td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td>45 mA</td>
<td>15.64 mA</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.3V</th>
<th>LED</th>
<th>MAX</th>
<th>STANDBY</th>
<th>SLEEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>35 mA</td>
<td>16.85 mA</td>
<td>0.4 mA</td>
<td></td>
</tr>
<tr>
<td>OFF</td>
<td>34 mA</td>
<td>15.85 mA</td>
<td></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>
EZO™ circuit identification

EZO™ Conductivity circuit

Legacy Conductivity circuit

Viewing correct datasheet

Viewing incorrect datasheet

Click here to view legacy datasheet
Conductivity probe range

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

<table>
<thead>
<tr>
<th>K 0.1</th>
<th>K 1.0</th>
<th>K 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.07µS – 50,000µS</td>
<td>5µS – 200,000+µS</td>
<td>10µS – 1S</td>
</tr>
</tbody>
</table>

Atlas Scientific™ has tested 3 different K value probe types

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</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.07 – 99.99</td>
<td>0.01µS</td>
</tr>
<tr>
<td>100.1 – 999.9</td>
<td>0.1µS</td>
</tr>
<tr>
<td>1,000 – 9,999</td>
<td>1.0µS</td>
</tr>
<tr>
<td>10,000 – 99,990</td>
<td>10µS</td>
</tr>
<tr>
<td>100,000 – 999,900</td>
<td>100µS</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 31 for UART, and 56 for I2C.

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

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

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

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

Example
EC,TDS,SAL,SG

Default LED blink pattern

This is the LED pattern for Continuous Mode (default state)
This can only happen when the device is in UART mode.
### Calibration theory

The most important part of calibration is watching the readings during the calibration process. It’s easiest to calibrate the device in its default state (UART mode, continuous readings). Switching the device to I^2C mode after calibration will not affect the stored calibration. If the device must be calibrated in I^2C mode be sure to request readings continuously so you can see the output from the probe.

### Pre-calibration setup

First, take readings from dry conductivity probe.

### Set probe type

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

### Dry calibration

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

```
17.00 → "Cal,dry" → 0.00  ✔ Correct

00.00 → "Cal,dry" → 0.00  ✔ Correct
```
Temperature compensation

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

Low point/single point calibration

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

Once the readings stabilize, issue the low point or single point calibration command.
Low point calibration: "Cal,low,1413" (Readings will NOT change)
Single point calibration: "Cal,1413" (Readings will change, calibration complete).

High point calibration

Shake the probe to remove trapped air and adjust the temperature as done in the previous step. Once the readings have stabilized issue the high point calibration command.
High point calibration: "Cal,high,12880" (Readings will change, calibration complete).
Power and data isolation

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

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

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

Correct

Incorrect

Basic EZO™
Inline Voltage Isolator

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

This technology works by using tiny transformers to induce the voltage across an air gap. PCB layout requires special attention for EMI/EMC and RF Control, having proper ground planes and keeping the capacitors as close to the chip as possible are crucial for proper performance. The two data channels have a 4.7kΩ 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 R7) this produces a voltage of 3.7V regardless of your input voltage.

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

Bread board

Incorrect wiring

Extended leads
Sloppy setup
Perfboards or Protoboards
*Embedded into your device

NEVER use Perfboards or Protoboards

*Only after you are familiar with EZO™ circuits operation
Available data protocols

UART

Default

Unavailable data protocols

I²C

SPI

Analog

RS-485

Mod Bus

4–20mA
UART mode

**Settings that are retained if power is cut**
- Baud rate
- Calibration
- Continuous mode
- Device name
- Enable/disable parameters
- Enable/disable response codes
- Hardware switch to I²C mode
- LED control
- Protocol lock
- Software switch to I²C mode

**Settings that are NOT retained if power is cut**
- Find
- Sleep mode
- Temperature compensation
UART mode

8 data bits  no parity
1 stop bit   no flow control

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

**Vcc**
- 3.3V – 5.5V

**Data format**

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

**Units**
- EC, TDS, SAL, SG

**Encoding**
- ASCII

**Format**
- string

**Terminator**
- carriage return

**Data type**
- floating point

**Decimal places**
- 3

**Smallest string**
- 3 characters

**Largest string**
- 40 characters
Default state

Mode
UART

Baud
9,600

Readings
continuous

Units
µS/cm

Speed
1 reading per second

LED
on

1,000 ms

Green
Standby

Cyan
Taking reading

Transmitting
Receiving data from device

2 parts

ASCII data string  Carriage return <cr>
Command  Terminator

Hex: 31 49 34 52
Dec: 1 4 0D
ASCII: 1 4

Advanced

ASCII: 1 , 4 1 3 <cr>
Hex: 31 2C 34 31 33
Dec: 49 44 52 49 51

9,600 baud (default)

1,413 <cr>

CPU

Receiver
Sending commands to device

2 parts

- Command (not case sensitive)
- Carriage return <cr>

ASCII data string
Terminator

Advanced

ASCII: S l e e p <cr>
Hex: 53 6C 65 65 70 0D
Dec: 83 108 101 101 112 13
LED color definition

- **Green**: UART standby
- **Cyan**: Taking reading
- **Purple**: Changing baud rate
- **Red**: Command not understood
- **White**: Find

**LED ON**

- **5V**: +2.5 mA
- **3.3V**: +1 mA
# 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. 37</td>
</tr>
<tr>
<td>C</td>
<td>enable/disable continuous reading</td>
<td>pg. 25</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>pg. 27</td>
</tr>
<tr>
<td>Export/import</td>
<td>export/import calibration</td>
<td>pg. 28</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>pg. 39</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>pg. 24</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>pg. 33</td>
</tr>
<tr>
<td>I2C</td>
<td>change to I²C mode</td>
<td>pg. 40</td>
</tr>
<tr>
<td>K</td>
<td>Set probe type</td>
<td>pg. 29</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>pg. 23</td>
</tr>
<tr>
<td>Name</td>
<td>set/show name of device</td>
<td>pg. 32</td>
</tr>
<tr>
<td>O</td>
<td>enable/disable parameters</td>
<td>pg. 31</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>pg. 38</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>pg. 26</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>pg. 36</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>pg. 35</td>
</tr>
<tr>
<td>T</td>
<td>temperature compensation</td>
<td>pg. 30</td>
</tr>
<tr>
<td>*OK</td>
<td>enable/disable response codes</td>
<td>pg. 34</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 <strong>default</strong></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>
</tbody>
</table>
| L,?     | ?L,1 <cr> or ?L,0 <cr>  
          | *OK <cr> |

---

**Diagram**

- **L,1**: LED on
- **L,0**: LED off
Find

Command syntax

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

Example

Response

Find <cr>  

*OK <cr>

*This command is only available for firmware version 2.10 and above.
# 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</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>

*This command is only available for firmware version 2.10 and above.

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
</table>
| C,1      | *OK <cr>  
EC,TDS,SAL,SG (1 sec) <cr>  
EC,TDS,SAL,SG (2 sec) <cr>  
EC,TDS,SAL,SG (3 sec) <cr> |
| C,30     | *OK <cr>  
EC,TDS,SAL,SG (30 sec) <cr>  
EC,TDS,SAL,SG (60 sec) <cr>  
EC,TDS,SAL,SG (90 sec) <cr> |
| C,0      | *OK <cr>  |
| C,?      | ?C,1 <cr> or ?C,0 <cr> or ?C,30 <cr>  
*OK <cr> |
Single reading mode

Command syntax

R <cr> takes single reading

Example

R <cr>

Response

1,413 <cr> *OK <cr>

Green
Standby

Cyan
Taking reading

Transmitting

600 ms
## Command syntax

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

---

*This command is only available for firmware version 2.10 and above.

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,dry <code>&lt;cr&gt;</code></td>
<td>*OK <code>&lt;cr&gt;</code></td>
</tr>
<tr>
<td>Cal,84 <code>&lt;cr&gt;</code></td>
<td>*OK <code>&lt;cr&gt;</code></td>
</tr>
<tr>
<td>Cal,low,1413 <code>&lt;cr&gt;</code></td>
<td>*OK <code>&lt;cr&gt;</code></td>
</tr>
<tr>
<td>Cal,high,12880 <code>&lt;cr&gt;</code></td>
<td>*OK <code>&lt;cr&gt;</code></td>
</tr>
<tr>
<td>Cal,clear <code>&lt;cr&gt;</code></td>
<td>*OK <code>&lt;cr&gt;</code></td>
</tr>
<tr>
<td>Cal,? <code>&lt;cr&gt;</code></td>
<td>?CAL,0 <code>&lt;cr&gt;</code> or ?CAL,1 <code>&lt;cr&gt;</code> or ?CAL,2 <code>&lt;cr&gt;</code></td>
</tr>
</tbody>
</table>
# Export/import calibration

## Command syntax

Export:
- `<cr>` export calibration string from calibrated device*
- `<cr>` import calibration string to new device*
- `<cr>` calibration string info*

Import:
- `<cr>` calibration string info*

---

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Export,</strong> n</td>
<td><strong>Response breakdown</strong></td>
</tr>
<tr>
<td><code>&lt;cr&gt;</code> Export, ?</td>
<td><code>10,120 &lt;cr&gt;</code></td>
</tr>
<tr>
<td><strong>Response breakdown</strong></td>
<td></td>
</tr>
<tr>
<td># of strings to export</td>
<td># of bytes to export</td>
</tr>
<tr>
<td>10</td>
<td>120</td>
</tr>
<tr>
<td>Export strings can be up to 12 characters long, and is always followed by <code>&lt;cr&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;cr&gt;</code> Export</td>
<td><code>59 6F 75 20 61 72 &lt;cr&gt;</code> (1 of 10)</td>
</tr>
<tr>
<td><code>&lt;cr&gt;</code> Export</td>
<td><code>65 20 61 20 63 6F &lt;cr&gt;</code> (2 of 10)</td>
</tr>
<tr>
<td>(7 more)</td>
<td></td>
</tr>
<tr>
<td><code>&lt;cr&gt;</code> Export</td>
<td><code>6F 6C 20 67 75 79 &lt;cr&gt;</code> (10 of 10)</td>
</tr>
<tr>
<td><code>&lt;cr&gt;</code> Export</td>
<td><em>DONE</em></td>
</tr>
</tbody>
</table>

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

---

*This command is only available for firmware version 2.10 and above.*
## Setting the probe type

### Command syntax

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

- **K, ?**  
  - probe K value?

### Example

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

---

K 1.0 is the default value

---

<table>
<thead>
<tr>
<th>K 0.1</th>
<th>K 1.0</th>
<th>K 10</th>
</tr>
</thead>
</table>
Temperature compensation

Command syntax

\texttt{T,n <cr> } n = \text{any value; floating point or int} \\
\texttt{T,? <cr> } \text{compensated temperature value?}

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{T,19.5 &lt;cr&gt;}</td>
<td>\texttt{*OK &lt;cr&gt;}</td>
</tr>
</tbody>
</table>
| \texttt{T,? <cr>} | \texttt{?T,19.5 <cr>}  \\
| | \texttt{*OK <cr>} |

Temperature is always in Celsius

\texttt{SDA} (TX) (RX) \texttt{SCL} \\
\texttt{SDA} (TX) (RX) \texttt{SCL} \\
\texttt{SDA} (TX) (RX) \texttt{SCL} \\
\texttt{SDA} (TX) (RX) \texttt{SCL}

\texttt{T,19.5 <cr>} \\
\texttt{1,400\mu S} \quad \texttt{1,413\mu S} \\
\texttt{T,19.5 <cr>}

Copyright © Atlas Scientific LLC

AtlasScientific
Environmental Robotics
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] &lt;cr&gt;</td>
<td>enable or disable output parameter</td>
</tr>
<tr>
<td>O,? &lt;cr&gt;</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 / O,EC,0 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt; enable / disable conductivity</td>
</tr>
<tr>
<td>O,TDS,1 / O,TDS,0 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt; enable / disable total dissolved solids</td>
</tr>
<tr>
<td>O,S,1 / O,S,0 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt; enable / disable salinity</td>
</tr>
<tr>
<td>O,SG,1 / O,SG,0 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt; enable / disable specific gravity</td>
</tr>
<tr>
<td>O,? &lt;cr&gt;</td>
<td>?,O,EC,TDS,S,SG &lt;cr&gt; if all are enabled</td>
</tr>
</tbody>
</table>

**Parameters**

- **EC**  conductivity
- **TDS** total dissolved solids
- **S** salinity
- **SG** specific gravity

**Followed by 1 or 0**

- 1  enabled
- 0  disabled

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

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name,n &lt;cr&gt;</td>
<td>set name</td>
</tr>
<tr>
<td>Name,? &lt;cr&gt;</td>
<td>show name</td>
</tr>
</tbody>
</table>

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

**Up to 16 ASCII characters**

## Example

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

## Example Response

**Name,zzt**

- *OK <cr>

**Name,?**

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

---

**Taking reading**

- 1,000 ms
- Green
- Standby
- Cyan

**Transmitting**

- 1,000 ms
- Green
- Standby
- Cyan
## Device information

### Command syntax

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

<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

```
?i, EC, 2.10
```

- **Device**: up
- **Firmware**: up

---

Copyright © Atlas Scientific LLC
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

<table>
<thead>
<tr>
<th>Status &lt;cr&gt;</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>?Status,P,5.038 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

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

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

Sleep <cr>

Standby 18.14 mA

Sleep 0.7 mA
## Change baud rate

### Command syntax

Baud,\texttt{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>

### Response

| n = | 300 | 1200 | 2400 | 9600 default | 19200 | 38400 | 57600 | 115200 |

### Default baud rates

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

### Example Response

```
*Baud,38400
*OK
```

### Command syntax details

- \texttt{Baud,n}  change baud rate

### Default baud rates

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

### Baud rate options

- \texttt{Baud,38400}
- \texttt{Baud,?}
- \texttt{?Baud,38400

### Example

1. **Standby**
2. **Baud,38400**
3. **Changing baud rate**
4. **(reboot)**
5. **Standby**

### 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 (default)</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>
<tbody>
<tr>
<td>Plock,1</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Plock,0</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Plock,?</td>
<td>?Plock,1 &lt;cr&gt; or ?Plock,0 &lt;cr&gt;</td>
</tr>
</tbody>
</table>

**Example Response**

- **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

Example

Factory <cr>

Response

*OK <cr>

(reboot)

*OK <cr>

*RS <cr>

*RE <cr>

Baud rate will not change
# Change to I²C mode

## Command syntax

I²C, n \(<\text{cr}>\) sets I²C address and reboots into I²C mode

\[ n = \text{any number 1 – 127} \]

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>I²C,100 (&lt;\text{cr}&gt;)</td>
<td>*OK (reboot in I²C mode)</td>
</tr>
</tbody>
</table>

## Wrong example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>I²C,139 (&lt;\text{cr}&gt;)</td>
<td>*ER (&lt;\text{cr}&gt;)</td>
</tr>
</tbody>
</table>

## Diagram

**I²C,100**

- **Green**: *OK \(<\text{cr}>\)*
- **Blue**: (reboot) now in I²C mode

---

Copyright © Atlas Scientific LLC
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

Disconnect RX line
I²C mode

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

To set your EZO™ device into I²C mode click here

Settings that are retained if power is cut:
- Calibration
- Change I²C address
- Enable/disable parameters
- Hardware switch to UART mode
- LED control
- Protocol lock
- Software switch to UART mode

Settings that are NOT retained if power is cut:
- Find
- Sleep mode
- Temperature compensation
**I²C mode**

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

**Vcc**  
3.3V – 5.5V

**Clock speed**  
100 – 400 kHz

---

**SDA**  
[Diagram]

**SCL**  
[Diagram]

---

**Data format**

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

**Units**  
EC, TDS, SAL, SG

**Encoding**  
ASCII

**Format**  
string

**Data type**  
floating point

**Decimal places**  
3

**Smallest string**  
3 characters

**Largest string**  
399 characters
Sending commands to device

5 parts

Start | I2C address | Write | Command (not case sensitive) | Stop
100 (0x64) | ASCII command string

Example

Start | 100 (0x64) | Write | Sleep | Stop
I2C address | Command

Advanced

Address bits

Start
A6 A5 A4 A3 A2 A1 A0 W ACK

The entire command as ASCII with all arguments

First letter of command | ACK | ... | Last letter of command | ACK

W = low

Stop

44 Copyright © Atlas Scientific LLC
Requesting data from device

7 parts

Start | I²C address | Read | Response code | Data string | Null | Stop
---|---|---|---|---|---|---
1 00 (0x64) | 1 byte | "1,413" | Terminator (Dec 0)

Advanced

CPU

1,413

Requesting data from device

Advanced

CPU

1,413

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

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

Response codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>255</td>
<td>no data to send</td>
</tr>
<tr>
<td>254</td>
<td>still processing, not ready</td>
</tr>
<tr>
<td>2</td>
<td>syntax error</td>
</tr>
<tr>
<td>1</td>
<td>successful request</td>
</tr>
</tbody>
</table>
**LED color definition**

- **Blue**: I²C standby
- **Green**: Taking reading
- **Purple**: Changing I²C ID#
- **Red**: Command not understood
- **White**: Find

**LED ON**

<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>
## 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>63</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>52</td>
</tr>
<tr>
<td>Export/import</td>
<td>export/import calibration</td>
<td>53</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>62</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>50</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>57</td>
</tr>
<tr>
<td>I2C</td>
<td>change I²C address</td>
<td>61</td>
</tr>
<tr>
<td>K</td>
<td>Set probe type</td>
<td>54</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>49</td>
</tr>
<tr>
<td>O</td>
<td>enable/disable parameters</td>
<td>56</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>60</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>51</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>59</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>58</td>
</tr>
<tr>
<td>T</td>
<td>temperature compensation</td>
<td>55</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>1 0 Dec Null</td>
</tr>
<tr>
<td></td>
<td>Wait 300ms</td>
</tr>
<tr>
<td>L,0</td>
<td>1 0 Dec Null</td>
</tr>
<tr>
<td></td>
<td>Wait 300ms</td>
</tr>
<tr>
<td>L,?</td>
<td>1 ?L,1 0 Dec ASCII Null or 1 ?L,0 0 Dec ASCII Null</td>
</tr>
<tr>
<td></td>
<td>Wait 300ms</td>
</tr>
</tbody>
</table>
Find

Command syntax

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

Example Response

Find <cr>

Send any character or command to terminate find.

This command will disable continuous mode

*This command is only available for firmware version 2.10 and above.
**Taking reading**

**Command syntax**

```
R  return 1 reading
```

**Example**

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>![Response Image]</td>
</tr>
</tbody>
</table>

**Response**

- **1** Dec
- **1,413** ASCII
- **0** Null

**Wait 600ms processing delay**

**Example Images**

- **Green**
  - Taking reading
- **Blue**
  - Standby
- **Transmitting**
# Calibration

## Command syntax

**Cal,dry**  
dry calibration

**Cal,n**  
single point calibration, where n = any value*

**Cal,low,n**  
low end calibration, where n = any value

**Cal,high,n**  
high end calibration, where n = any value

**Cal,clear**  
delete calibration data

**Cal,?**  
device calibrated?

---

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cal,dry</strong></td>
<td>![Wait 600ms](1 Dec 0 Null)</td>
</tr>
<tr>
<td><strong>Cal,84</strong></td>
<td>![Wait 600ms](1 Dec 0 Null)</td>
</tr>
<tr>
<td><strong>Cal,low,1413</strong></td>
<td>![Wait 600ms](1 Dec 0 Null)</td>
</tr>
<tr>
<td><strong>Cal,high,12880</strong></td>
<td>![Wait 600ms](1 Dec 0 Null)</td>
</tr>
<tr>
<td><strong>Cal,clear</strong></td>
<td>![Wait 300ms](1 Dec 0 Null)</td>
</tr>
<tr>
<td><strong>Cal,</strong></td>
<td>![Wait 300ms](1 ASCII 0 Null or 1 ASCII point 0 or 1 ASCII two point 0)</td>
</tr>
</tbody>
</table>

---

*This command is only available for firmware version 2.10 and above.*

---

600ms processing delay

Dry calibration must always be done first!
## Export/import calibration

### Command syntax

**Export**
- export calibration string from calibrated device*

**Import**
- import calibration string to new device*

**Export,**
- calibration string info*

---

### 300ms processing delay

*This command is only available for firmware version 2.10 and above.

---

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Example</th>
<th>Response Breakdown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export,?</td>
<td>Wait 300ms</td>
<td>10, 120</td>
</tr>
<tr>
<td>Export</td>
<td>Wait 300ms</td>
<td>59 6F 75 20 61 72</td>
</tr>
<tr>
<td>Export</td>
<td>Wait 300ms</td>
<td>65 20 61 20 63 6F</td>
</tr>
<tr>
<td>Export</td>
<td>Wait 300ms</td>
<td>*DONE</td>
</tr>
<tr>
<td>Import, n</td>
<td>Import, 59 6F 75</td>
<td>(1 of 10)</td>
</tr>
</tbody>
</table>

---

*Export strings can be up to 12 characters long.
Setting the probe type

**Command syntax**

- $K_n$  $n$ = any value; floating point in ASCII
- $K?$  probe $K$ value?

**Example Response**

- $K,10$
  - Wait 300ms
  - 1 Dec 0 Null

- $K,?$
  - Wait 600ms
  - 1 Dec $K,10$ 0 ASCII Null
## Temperature compensation

### Command syntax

- `T,n`  
  n = any value; floating point or int

- `T,?`  
  compensated temperature value?

### Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>T,19.5</code></td>
<td><img src="1400%C2%B5S" alt="Wait 300ms" /> 1 Dec 0 Null</td>
</tr>
<tr>
<td><code>T,?</code></td>
<td><img src="1413%C2%B5S" alt="Wait 300ms" /> 1 Dec ASCII 0 Null</td>
</tr>
</tbody>
</table>

- `1,400µS`  
- `1,413µS`

### Example Response

```
1
Dec
Null
```

- `1,400µS`  
- `1,413µS`
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 / O,EC,0</td>
<td>enable / disable conductivity</td>
</tr>
<tr>
<td>O,TDS,1 / O,TDS,0</td>
<td>enable / disable total dissolved solids</td>
</tr>
<tr>
<td>O,SG,1 / O,SG,0</td>
<td>enable / disable specific gravity</td>
</tr>
</tbody>
</table>

Example Response

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>O,?</td>
<td>if all are enabled</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”.

Copyright © Atlas Scientific LLC
## Device information

### Command syntax

| i | device information |

### Example

#### i

| Wait 300ms | 1 | ASCII | 0 | Null |

### Response breakdown

| ?i, EC, 2.10 | 2.10 |

| Device | Firmware |
# Reading device status

## Command syntax

- **Wait 300ms**

## Status voltage at Vcc pin and reason for last restart

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Status</strong></td>
<td><strong>Response</strong></td>
</tr>
<tr>
<td><img src="image" alt="Wait 300ms" /></td>
<td><img src="image" alt="1" /> <img src="image" alt="?Status,P,5.038" /> <img src="image" alt="0" /> <img src="image" alt="Null" /></td>
</tr>
</tbody>
</table>

## 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

Example | Response
---------|---------
Sleep    | no response
Any command | wakes up device

Send any character or command to awaken device.

Do not read status byte after issuing sleep command.

<table>
<thead>
<tr>
<th></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>

Standby

Sleep
Protocol lock

Command syntax

Plock,1  enable Plock
Plock,0  disable Plock  default
Plock,?  Plock on/off?

300ms processing delay

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plock,1</td>
<td><img src="wait_300ms.png" alt="" /> <img src="Dec" alt="1" /> <img src="Null" alt="0" /></td>
</tr>
<tr>
<td>Plock,0</td>
<td><img src="Dec" alt="1" /> <img src="Null" alt="0" /></td>
</tr>
<tr>
<td>Plock,?</td>
<td><img src="Dec" alt="1" /> <img src="ASCII" alt="?Plock,1" /> <img src="Null" alt="0" /></td>
</tr>
</tbody>
</table>

300ms processing delay

Example Response

300ms processing delay

Cannot change to UART

Copyright © Atlas Scientific LLC
**I²C address change**

**Command syntax**

I²C,n sets I²C address and reboots into I²C mode

---

### Example

<table>
<thead>
<tr>
<th>I²C,n</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).

---

I²C address change command:

- **Example Response:**
  - I²C,101
  - device reboot

**I²C,101**

(reboot)
**Factory reset**

**Command syntax**

- Factory enable factory reset

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

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

- I²C address will not change

- Clears calibration
- LED on
- Response codes enabled

**Factory**

![Image showing the process of factory reset](image-url)

(reboot)
## Change to UART mode

### Command syntax

Baud,\(n\) switch from I\(^2\)C to UART

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud,9600</td>
<td>reboot in UART mode</td>
</tr>
</tbody>
</table>

\(n = \{300, 1200, 2400, 9600, 19200, 38400, 57600, 115200\}\)

---

![Diagram showing the transition from I²C to UART mode](image-url)
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
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 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 & 52.

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

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>Initial release (April 17, 2014)</td>
<td></td>
</tr>
<tr>
<td>V1.1</td>
<td>(June 2, 2014)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Change specific gravity equation to return 1.0 when the uS reading is &lt; 1000 (previously returned 0.0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Change accuracy of specific gravity from 2 decimal places to 3 decimal places</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Don't save temperature changes to EEPROM</td>
<td></td>
</tr>
<tr>
<td>V1.2</td>
<td>(Aug 1, 2014)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Baud rate change is now a long, purple blink</td>
<td></td>
</tr>
<tr>
<td>V1.5</td>
<td>Baud rate change (Nov 6, 2014)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Change default baud rate to 9600</td>
<td></td>
</tr>
<tr>
<td>V1.6</td>
<td>I2C bug (Dec 1, 2014)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Fix I2C bug where the circuit may inappropriately respond when other I2C devices are connected.</td>
<td></td>
</tr>
<tr>
<td>V1.8</td>
<td>Factory (April 14, 2015)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Changed “X” command to “Factory”</td>
<td></td>
</tr>
<tr>
<td>V1.95</td>
<td>Plock (March 31, 2016)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Added protocol lock feature “Plock”</td>
<td></td>
</tr>
<tr>
<td>V1.96</td>
<td>EEPROM (April 26, 2016)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Fixed glitch where EEPROM would get erased if the circuit lost power 900ms into startup</td>
<td></td>
</tr>
<tr>
<td>V2.10</td>
<td>(April 12, 2017)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Added &quot;Find&quot; command.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Added &quot;Export/import&quot; command.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Modified continuous mode to be able to send readings every &quot;n&quot; seconds.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Default output changed from CSV string of 4 values to just conductivity; Other values must be enabled.</td>
<td></td>
</tr>
<tr>
<td>V2.11</td>
<td>(April 28, 2017)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Fixed &quot;Sleep&quot; bug, where it would draw excessive current.</td>
<td></td>
</tr>
<tr>
<td>V2.12</td>
<td>(May 9, 2017)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Fixed glitch in sleep mode, where circuit would wake up to a different I²C address.</td>
<td></td>
</tr>
</tbody>
</table>
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.