**EZODO™**
Embedded Dissolved Oxygen Circuit

**Reads**

**Dissolved Oxygen**

- 0.01 – 100+ mg/L
- 0.1 – 400+ % saturation

**Accuracy**

+/- 0.05 mg/L

**Response time**

1 reading per sec

**Supported probes**

Any galvanic probe

**Calibration**

- 1 or 2 point
- Yes

**Data protocol**

- UART & I²C
- 97 (0x61)

**Default I²C address**

- 3.3V – 5V

**Operating voltage**

- ASCII

---

This is an evolving document, check back for updates.
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
Operating principle 5
Power and data isolation 6
Correct wiring 8
Calibration theory 9
Preserve calibration solution 12
Available data protocols 13

UART

UART mode 15
Default state 16
Receiving data from device 17
Sending commands to device 18
LED color definition 19
UART quick command page 20
LED control 21
Find 22
Continuous reading mode 23
Single reading mode 24
Calibration 25
Export calibration 26
Import calibration 27
Temperature compensation 28
Salinity compensation 29
Pressure compensation 30
Enable/disable parameters 31
Naming device 32
Device information 33
Response codes 34
Reading device status 35
Sleep mode/low power 36
Change baud rate 37
Protocol lock 38
Factory reset 39
Change to I²C mode 40
Manual switching to I²C 41

I²C

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

Circuit footprint 67
Datasheet change log 68
Warranty 71
Power consumption

<table>
<thead>
<tr>
<th></th>
<th>5V</th>
<th>3.3V</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED</td>
<td>MAX</td>
<td>STANDBY</td>
</tr>
<tr>
<td>ON</td>
<td>13.5 mA</td>
<td>13.1 mA</td>
</tr>
<tr>
<td>OFF</td>
<td>12.7 mA</td>
<td>12.7 mA</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™ D.O.)</td>
<td>-65 °C</td>
<td></td>
<td>125 °C</td>
</tr>
<tr>
<td>Operational temperature (EZO™ D.O.)</td>
<td>-40 °C</td>
<td>25 °C</td>
<td>85 °C</td>
</tr>
<tr>
<td>VCC</td>
<td>3.3V</td>
<td>5V</td>
<td>5.5V</td>
</tr>
</tbody>
</table>
Operating principle

A galvanic dissolved oxygen probe consists of a Polytetrafluoroethylene membrane, an anode bathed in an electrolyte and a cathode. Oxygen molecules diffuse through the probes membrane at a constant rate (without the membrane the reaction happens too quickly). Once the oxygen molecules have crossed the membrane they are reduced at the cathode and a small voltage is produced. If no oxygen molecules are present, the probe will output 0 mV. As the oxygen increases so does the mV output from the probe. Each probe will output a different voltage in the presence of oxygen. The only thing that is constant is that \(0\text{mV} = 0\text{ Oxygen}\). (A galvanic dissolved oxygen probe can also be used to detect the Oxygen content in gases).

The Atlas Scientific™ EZO™ Dissolved Oxygen circuit works with:

- **X** Optical probe  
  Slow response, requires external power, expensive.
- **X** Polar Graphic probe  
  Requires external power, output in μA.
- **✓ Galvanic probe**  
  Requires no external power, output in mV.

Flow Dependence

One of the drawbacks from using a galvanic probe is that it consumes a **VERY** small amount of the oxygen it reads. Therefore, a small amount of water movement is necessary to take accurate readings. **Approximately 60 ml/min.**
Power and data isolation

The Atlas Scientific EZO™ Dissolved Oxygen circuit is a very sensitive device. This sensitivity is what gives the Dissolved Oxygen circuit its accuracy. This also means that the Dissolved Oxygen 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 Dissolved Oxygen 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 Dissolved Oxygen probe in a cup of water by itself. The readings should stabilize quickly, confirming that electrical noise was the issue.

When reading Dissolved Oxygen and Conductivity together, it is strongly recommended that the EZO™ Dissolved Oxygen circuit is electrically isolated from the EZO™ Conductivity circuit.

Without isolation, Conductivity readings will effect Dissolved Oxygen accuracy.
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 3.9V 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

- Bread board
- Extended leads
- Sloppy setup
- Perfboards or Protoboards

Incorrect wiring

- Extended leads
- Sloppy setup
- Perfboards or Protoboards

Never use Perfboards or Protoboards

*Only after you are familiar with EZO™ circuits operation

Part # COM-104
Part # ISCCB
Part # USB-ISO

Copyright © Atlas Scientific LLC
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.

The Atlas Scientific EZO™ Dissolved Oxygen circuit, has a flexible calibration protocol, allowing for single point or dual point (optional) calibration.

Calibrate first, compensate later. Temperature, salinity and pressure compensation values have no effect on calibration.
Single point calibration

Carefully pull off and discard the cap from the Dissolved Oxygen probe. Let the Dissolved Oxygen probe sit, exposed to air until the readings stabilize. (small movement from one reading to the next is normal).

Do not unscrew!
Cap is only used to protect the probe during shipping.

Once the readings have stabilized, issue the calibration command. "cal"

After calibration is complete, you should see readings between 9.09 – 9.1X mg/L. (only if temperature, salinity and pressure compensation are at default values)
Dual point calibration (optional)

Only perform this calibration if you require accurate readings below 1.0 mg/L

After you have calibrated the EZO™ Dissolved Oxygen circuit using the "Cal" command; Place the probe into the Zero Dissolved Oxygen calibration solution and stir the probe around to remove trapped air (which could cause readings to go high). Let the probe sit in Zero D.O. calibration solution until readings stabilize. (small movement from one reading to the next is normal).

Once the readings have stabilized, issue the calibration command. "cal,0"

![Image showing calibration process and readings]
How to preserve the Zero D.O. calibration solution

Oxygen is everywhere. The Zero D.O. calibration solution has been designed to chemically absorb oxygen. Once the bottle has been opened the test solution has been exposed to oxygen and will slowly stop working.

Inside each bottle of the calibration solution is a small amount of nitrogen gas that helps displace oxygen out of the bottle during the filling process. When the Dissolved Oxygen probe is removed from the bottle, oxygen will enter the bottle and begin to dissolve into the solution.

In order slow down this process, fill the void space of the bottle with any gas (other than oxygen) to preserve the calibration solution. Gas from a lighter works great if other gases are currently unobtainable.
<table>
<thead>
<tr>
<th>Available data protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>UART (Default)</td>
</tr>
<tr>
<td>I²C</td>
</tr>
<tr>
<td>SPI</td>
</tr>
<tr>
<td>Analog</td>
</tr>
<tr>
<td>RS-485</td>
</tr>
<tr>
<td>Mod Bus</td>
</tr>
<tr>
<td>4–20mA</td>
</tr>
</tbody>
</table>
## UART mode

<table>
<thead>
<tr>
<th>Settings that are retained if power is cut</th>
<th>Settings that are NOT retained if power is cut</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud rate</td>
<td>Find</td>
</tr>
<tr>
<td>Calibration</td>
<td>Pressure compensation</td>
</tr>
<tr>
<td>Continuous mode</td>
<td>Salinity compensation</td>
</tr>
<tr>
<td>Device name</td>
<td>Sleep mode</td>
</tr>
<tr>
<td>Enable/disable parameters</td>
<td>Temperature compensation</td>
</tr>
<tr>
<td>Enable/disable response codes</td>
<td></td>
</tr>
<tr>
<td>Hardware switch to I²C mode</td>
<td></td>
</tr>
<tr>
<td>LED control</td>
<td></td>
</tr>
<tr>
<td>Protocol lock</td>
<td></td>
</tr>
<tr>
<td>Software switch to I²C mode</td>
<td></td>
</tr>
</tbody>
</table>
UART mode

8 data bits  no parity
1 stop bit  no flow control

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

RX
Data in

TX
Data out

Vcc
3.3V – 5.5V

0V

Data format

Reading  D.O.
Units  mg/L & (% sat)  [when enabled]
Encoding  ASCII
Format  string  [CSV string when % sat is enabled]
Terminator  carriage return

Data type
- floating point

Decimal places
mg/L = 2
% sat = 1

Smallest string
4 characters

Largest string
16 characters
Default state

Mode: UART

Baud: 9,600

Readings: continuous

Speed: 1 reading per second

Temperature compensation: 20 °C

Salinity compensation: 0 (Fresh water)

Pressure compensation: 101.3 kPa (Sea level)

LED: on

1,000 ms

Green

Cyan

Transmitting

Standby

Taking reading
Receiving data from device

2 parts

ASCII data string

Command

Carriage return <cr>

Terminator

Advanced

ASCII: 7 . 8 2 <cr>

Hex: 37 2E 38 32 0D

Dec: 55 46 56 50 13

CPU

Receiver
Sending commands to device

2 parts

Command (not case sensitive)

ASCII data string

Carriage return <cr>

Terminator

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>+0.4 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>+0.2 mA</td>
</tr>
</tbody>
</table>
## 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, 9,600</td>
</tr>
<tr>
<td>C</td>
<td>enable/disable continuous reading</td>
<td>pg. 23, enabled</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>pg. 25, n/a</td>
</tr>
<tr>
<td>Export</td>
<td>export calibration</td>
<td>pg. 26, n/a</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>pg. 39, n/a</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>pg. 22, n/a</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>pg. 33, n/a</td>
</tr>
<tr>
<td>I2C</td>
<td>change to I²C mode</td>
<td>pg. 40, not set</td>
</tr>
<tr>
<td>Import</td>
<td>import calibration</td>
<td>pg. 27, n/a</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>pg. 21, enabled</td>
</tr>
<tr>
<td>Name</td>
<td>set/show name of device</td>
<td>pg. 32, not set</td>
</tr>
<tr>
<td>O</td>
<td>enable/disable parameters</td>
<td>pg. 31, mg/L</td>
</tr>
<tr>
<td>P</td>
<td>pressure compensation</td>
<td>pg. 30, 101.3 kPa</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>pg. 38, disabled</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>pg. 24, n/a</td>
</tr>
<tr>
<td>S</td>
<td>salinity compensation</td>
<td>pg. 29, n/a</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>pg. 36, n/a</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>pg. 35, n/a</td>
</tr>
<tr>
<td>T</td>
<td>temperature compensation</td>
<td>pg. 28, 20°C</td>
</tr>
<tr>
<td>*OK</td>
<td>enable/disable response codes</td>
<td>pg. 34, enable</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</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>
**Find**

**Command syntax**

<table>
<thead>
<tr>
<th>Find &lt;cr&gt;</th>
<th>LED rapidly blinks white, used to help find device</th>
</tr>
</thead>
</table>

**Example** | **Response**
| Find <cr> | *OK <cr> |

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

---

**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
- Pressure compensation
- Salinity compensation
- Sleep mode
- Temperature compensation

---

**Example Response**

```
r 0.1
```
## 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>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>C,1</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>DO (1 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>DO (2 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>DO (3 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td>C,30</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>DO (30 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>DO (60 sec) &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>DO (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>
<tbody>
<tr>
<td>R &lt;cr&gt;</td>
<td>7.82 &lt;cr&gt; *OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Green Standby</th>
<th>Cyan Taking reading</th>
<th>Transmitting</th>
</tr>
</thead>
</table>

600 ms
## Calibration

### Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal</td>
<td>calibrate to atmospheric oxygen levels</td>
</tr>
<tr>
<td>Cal,0</td>
<td>calibrate device to 0 dissolved oxygen</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</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,0</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>Cal,clear</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>
</tbody>
</table>

#### Example Response

- `?Cal,0` or `?Cal,1` or `?Cal,2`  
  - single point
  - two point

### Calibration Graphs

- **9.53 mg/L**  
  - **Cal**  
- **8.82 mg/L**  
  - **Cal**
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>
<th>Response breakdown</th>
</tr>
</thead>
</table>
| Export,? <cr> | 10,120 <cr> | 10, 120
|           |          | # of strings to export # of bytes to export |
| 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: Use this command to download calibration settings

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

... (remaining strings)

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

**Response**

*OK <cr>
*OK <cr>

... (remaining responses)

*OK <cr>

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

---

27 Copyright © Atlas Scientific LLC
# Temperature compensation

## Command syntax

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

## Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>T,19.5 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>RT,19.5 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>T,? &lt;cr&gt;</td>
<td>?T,19.5 &lt;cr&gt;</td>
</tr>
</tbody>
</table>

*This is a new command for firmware V2.13*

---

**Default temperature = 20°C**

**Temperature is always in Celsius**

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

---

![Graph showing temperature compensation](image)

- **T,19.5 <cr>**
  - **8.82**
  - **8.91**
Salinity compensation

**Command syntax**

- `S,n`  
  `n = any value in microsiemens`
- `S,n,ppt`  
  `n = any value in ppt`
- `S,?`  
  `compensated salinity value?`

**Default value = 0 μs**

If the conductivity of your water is less than 2,500 μS this command is irrelevant

**Example**

<table>
<thead>
<tr>
<th>Command Syntax</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>S,50000 &lt;cr&gt;</code></td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td><code>S,37.5,ppt &lt;cr&gt;</code></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>
| `S,? <cr>` | ?S,50000,μS <cr> or ?S,37.5,ppt <cr>  
  *OK <cr> |

**Example Response**

```
?S,50000,μS       or ?S,37.5,ppt
*OK
```

```
8.91 mg/L
```

```
8.01 mg/L
```

29 Copyright © Atlas Scientific LLC
Pressure compensation

Command syntax

\[ \text{P,} n \text{ <cr> } n = \text{any value in kPa} \]
\[ \text{P,} ? \text{ <cr> compensated pressure value?} \]

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>P,90.25 &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td>P,? &lt;cr&gt;</td>
<td>? , P,90.25 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>? , P,90.25 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

Default value = 101.3 kPa
This parameter can be omitted if the water is less than 10 meters deep

Example Response

| P,90.25 <cr>                  | *OK <cr>                        |
| 8.01 mg/L                     | 6.94 mg/L                       |

P,90.25 <cr>
# Enable/disable parameters from output string

## Command syntax

- **Enable or disable output parameter**

  \[ \texttt{O, [parameter],[1,0]} \quad \texttt{<cr>} \]

- **Enabled parameter?**

  \[ \texttt{O,?} \quad \texttt{<cr>} \]

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{O,mg,1} / \texttt{O,mg,0} \texttt{&lt;cr&gt;}</td>
<td>*\texttt{OK &lt;cr&gt;} enable / disable mg/L</td>
</tr>
<tr>
<td>\texttt{O,%,1} / \texttt{O,%,0} \texttt{&lt;cr&gt;}</td>
<td>*\texttt{OK &lt;cr&gt;} enable / disable percent saturation</td>
</tr>
<tr>
<td>\texttt{O,?} \texttt{&lt;cr&gt;}</td>
<td>\texttt{?,O,%,mg &lt;cr&gt;} if both are enabled</td>
</tr>
</tbody>
</table>

## Parameters

- \texttt{mg} \hspace{1em} mg/L
- \texttt{%} \hspace{1em} percent saturation

## Followed by 1 or 0

- 1 \hspace{1em} enabled
- 0 \hspace{1em} 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>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name, n</td>
<td><code>&lt;cr&gt;</code> set name</td>
<td>Set the device name to <code>n</code></td>
</tr>
<tr>
<td>Name, ?</td>
<td><code>&lt;cr&gt;</code> show name</td>
<td>Show the current device name</td>
</tr>
</tbody>
</table>

Example Response

- **Name,zzt `<cr>`**
  - **Response:** *OK `<cr>`

- **Name,? `<cr>`**
  - **Response:** ?Name,zzt `<cr>`
    - **Response:** *OK `<cr>`

Example Image:

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

- **Name,**
  - **Response:** Name,zzt `<cr>`
    - **Response:** *OK `<cr>`
# Device information

## Command syntax

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

## Example

<table>
<thead>
<tr>
<th>i &lt;cr&gt;</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>i &lt;cr&gt;</td>
<td>?i, D.O., 1.98 &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;cr&gt;</td>
</tr>
</tbody>
</table>

## Response breakdown

```
?i, D.O., 1.98
```

<table>
<thead>
<tr>
<th>Device</th>
<th>Firmware</th>
</tr>
</thead>
<tbody>
<tr>
<td>?i,</td>
<td>D.O.,</td>
</tr>
<tr>
<td>1.98</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>7.82 &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>7.82 &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

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

### Response breakdown

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

### Restart codes

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

---

Copyright © Atlas Scientific LLC
Sleep mode/low power

Command syntax

Sleep <cr> enter sleep mode/low power

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep &lt;cr&gt;</td>
<td>*OK &lt;cr&gt;</td>
</tr>
<tr>
<td></td>
<td>*SL &lt;cr&gt;</td>
</tr>
<tr>
<td>Any command</td>
<td>*WA &lt;cr&gt;  wakes up device</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Standby</th>
<th>Sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>13.1 mA</td>
<td>0.66 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>12 mA</td>
<td>0.3 mA</td>
</tr>
</tbody>
</table>
Change baud rate

Command syntax

Baud,n <cr>  change baud rate

Example          Response

Baud,38400 <cr> *OK <cr>

Baud,? <cr>

?Baud,38400 <cr>
*OK <cr>

n =

300
1200
2400
9600 default
19200
38400
57600
115200

*OK <cr>

(reboot)

Standby

Baud,38400 <cr>

Changing baud rate

*OK <cr>

Standby
## Protocol lock

### Command syntax

- **Plock,1** <cr> enable Plock
- **Plock,0** <cr> disable Plock
- **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 Response

- **Plock,1**
  - *OK <cr>
- **I2C,100**
  - cannot change to I²C
  - *ER <cr>
- **Short**
  - cannot change to I²C

---

*Locks device to UART mode.*
Factory reset

Command syntax

Factory <cr> enable factory reset

Example

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

## Wrong example

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

## Response

- **Green**
  - *OK <cr>
- **Blue**
  - now in I²C mode

**Default I²C address 97 (0x61)**

---

I2C,100

---

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 PGND
- 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 97 (0x61)

Example

Wrong Example

Disconnect RX line
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
- Pressure compensation
- Salinity compensation
- Sleep mode
- Temperature compensation
I²C mode

I²C address  (0x01 – 0x7F)

97 (0x61) default

Vcc  3.3V – 5.5V

Clock speed  100 – 400 kHz

SDA

SCL

0V

VCC

4.7k resistor may be needed

Data format

Reading  D.O.

Units  mg/L & (% sat) when enabled

Encoding  ASCII

Format  string (CSV string when % sat is enabled)

Data type  floating point

Decimal places  mg/L = 2

% sat = 1

Smallest string  4 characters

Largest string  16 characters
Sending commands to device

5 parts

Start  I²C address  Write  Command (not case sensitive)  Stop

97 (0x61)

ASCII command string

Example

Start  97 (0x61)  Write  Sleep  Stop

I²C address  Command

Advanced

Address bits

The entire command as ASCII with all arguments

Start

... W = low

Stop
Requesting data from device

7 parts

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

97 (0x61) | 1 byte | "7.82" | Terminator (Dec 0)

Advanced

CPU

7.82

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

A6 – A0 | ACK | Data | ACK | Data | N | ACK | Null | ACK | Null | NACK | Stop

1 55 46 56 50 0 = 7.82

Dec | ASCII | Dec

Copyright © Atlas Scientific LLC
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

<table>
<thead>
<tr>
<th>LED ON</th>
<th>5V</th>
<th>+0.4 mA</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3V</td>
<td>+0.2 mA</td>
<td></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>change back to UART mode</td>
<td>65</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>52</td>
</tr>
<tr>
<td>Export</td>
<td>export calibration</td>
<td>53</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>64</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>59</td>
</tr>
<tr>
<td>I2C</td>
<td>change I²C address</td>
<td>63</td>
</tr>
<tr>
<td>Import</td>
<td>import calibration</td>
<td>54</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>49</td>
</tr>
<tr>
<td>O</td>
<td>removing parameters</td>
<td>58</td>
</tr>
<tr>
<td>P</td>
<td>pressure compensation</td>
<td>57</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>62</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>51</td>
</tr>
<tr>
<td>S</td>
<td>salinity compensation</td>
<td>56</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>61</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>60</td>
</tr>
<tr>
<td>T</td>
<td>temperature compensation</td>
<td>55</td>
</tr>
</tbody>
</table>
LED control

Command syntax

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

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>

300ms processing delay

Example Response

- 300ms processing delay
- Wait 300ms

Copyright © Atlas Scientific LLC
Find

Command syntax

Find  LED rapidly blinks white, used to help find device

Example | Response
---|---
Find | 1 0
Wait 300ms
Dec Null

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

300ms processing delay

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

Command syntax

R  return 1 reading

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td><img src="image1" alt="Image" /></td>
</tr>
<tr>
<td></td>
<td>Wait 600ms</td>
</tr>
<tr>
<td></td>
<td>Dec</td>
</tr>
</tbody>
</table>

600ms processing delay

Example Response

Wait 600ms

R return 1 reading

Example

Response
# Calibration

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal</td>
<td>calibrate to atmospheric oxygen levels</td>
</tr>
<tr>
<td>Cal,0</td>
<td>calibrate device to 0 dissolved oxygen</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 Response**

- **Cal**
  - Wait 1300ms
  - 1 Dec 0 Null
- **Cal,0**
  - Wait 1300ms
  - 1 Dec 0 Null
- **Cal,clear**
  - Wait 300ms
  - 1 Dec 0 Null
- **Cal,**
  - Wait 300ms
  - 1 ?Cal,0 0 Dec ASCII Null or 1 ?Cal,1 0 Dec ASCII Null or 1 ?Cal,2 0 Dec ASCII Null

The EZO™ Dissolved Oxygen circuit uses single and/or two point calibration.
# Export calibration

**Response breakdown**
- # of strings to export: 10
- # of bytes to export: 120

## Command syntax
- **Export,?** calibration string info
- **Export** export calibration string from calibrated device

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export,?</td>
<td>![Wait 300ms] 1 10,120 0 Response breakdown 10, 120 Export strings can be up to 12 characters long</td>
</tr>
<tr>
<td>Export</td>
<td>![Wait 300ms] 1 59 6F 75 20 61 72 0 (1 of 10)</td>
</tr>
<tr>
<td>Export</td>
<td>![Wait 300ms] 1 65 20 61 20 63 6F 0 (2 of 10)</td>
</tr>
<tr>
<td>(7 more)</td>
<td>![Wait 300ms]</td>
</tr>
<tr>
<td>Export</td>
<td>![Wait 300ms] 1 6F 6C 20 67 75 79 0 (10 of 10)</td>
</tr>
<tr>
<td>Export</td>
<td>![Wait 300ms] 1 <em>DONE</em> 0</td>
</tr>
</tbody>
</table>
**Import calibration**

**Command syntax**

Import, \texttt{import calibration string to new device}

**Example**

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

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

---

300ms processing delay

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

#### Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T,n</td>
<td>n = any value; floating point or int</td>
</tr>
<tr>
<td>T,?</td>
<td>compensated temperature value?</td>
</tr>
<tr>
<td>RT,n</td>
<td>set temperature compensation and take a reading*</td>
</tr>
</tbody>
</table>

**Default temperature = 20°C**  
Temperature is always in Celsius  
Temperature is not retained if power is cut

**300ms processing delay**

---

#### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>T,19.5</td>
<td><img src="image" alt="Example" /></td>
</tr>
<tr>
<td>RT,19.5</td>
<td><img src="image" alt="Example" /></td>
</tr>
<tr>
<td>T,?</td>
<td><img src="image" alt="Example" /></td>
</tr>
</tbody>
</table>

---

*This is a new command for firmware V2.13*
## Salinity Compensation

### Command Syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S,n</td>
<td>n = any value in microsiemens</td>
</tr>
<tr>
<td>S,n,ppt</td>
<td>n = any value in ppt</td>
</tr>
<tr>
<td>S,?</td>
<td>compensated salinity value?</td>
</tr>
</tbody>
</table>

### Example Response

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>S,50000</td>
<td><code>1 0</code></td>
</tr>
<tr>
<td>S,37.5,ppt</td>
<td><code>1 0</code></td>
</tr>
<tr>
<td>S,?</td>
<td><code>1 ?S,50000,μS 0</code></td>
</tr>
<tr>
<td></td>
<td>or</td>
</tr>
<tr>
<td></td>
<td><code>1 ?S,37.5,ppt 0</code></td>
</tr>
</tbody>
</table>

*If the conductivity of your water is less than 2,500μS this command is irrelevant.*
## Pressure compensation

### Command syntax

- `P,n`  
  - `n` = any value in kPa

- `P,?`  
  - compensated pressure value?

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>P,90.25</code></td>
<td></td>
</tr>
</tbody>
</table>
  - `Wait 300ms`
  - `1` Dec
  - `0` Null |

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>P,?</code></td>
<td></td>
</tr>
</tbody>
</table>
  - `Wait 300ms`
  - `1` Dec
  - `?`,P,90.25` ASCII `0` Null |

### Example Response

This parameter can be omitted if the water is less than 10 meters deep.

- `57` mg/L

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.01</td>
<td>mg/L</td>
</tr>
<tr>
<td>6.94</td>
<td>mg/L</td>
</tr>
</tbody>
</table>

### Diagram

- Graph showing data points before and after pressure compensation.
  - Before compensation: 8.01 mg/L
  - After compensation: 6.94 mg/L
## Enable/disable parameters from output string

### Command syntax

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

### Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>O,mg,1</code> / <code>O,mg,0</code></td>
<td><code>1 0</code> Dec Null</td>
</tr>
<tr>
<td><code>O,%,1</code> / <code>O,%,0</code></td>
<td><code>1 0</code> Dec Null</td>
</tr>
<tr>
<td><code>O,?</code></td>
<td><code>1 ?O,%,,mg 0</code> Dec ASCII Null</td>
</tr>
</tbody>
</table>

### Parameters

- `mg` mg/L
- `%` percent saturation

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>i</th>
<th>device information</th>
</tr>
</thead>
</table>

### Example

#### i

#### Wait 300ms

#### Dec

<table>
<thead>
<tr>
<th>?i,D.O.,1.98</th>
</tr>
</thead>
</table>

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

| 0 |

### Response breakdown

<table>
<thead>
<tr>
<th>?i, D.O., 1.98</th>
</tr>
</thead>
</table>

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

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

---

300ms processing delay
# 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>

**300ms processing delay**

**Example**

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

**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>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>
Sleep mode/low power

Command syntax

**Sleep** enter sleep mode/low power

Send any character or command to awaken device.

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

<table>
<thead>
<tr>
<th>Voltage</th>
<th>STANDBY</th>
<th>SLEEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>13.1 mA</td>
<td>0.66 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>12 mA</td>
<td>0.3 mA</td>
</tr>
</tbody>
</table>

Do not read status byte after issuing sleep command.

Sleep

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>

300ms processing delay

Locks device to I²C mode.

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plock,1</td>
<td><img src="image" alt="Wait 300ms" /></td>
</tr>
<tr>
<td>Plock,0</td>
<td><img src="image" alt="Wait 300ms" /></td>
</tr>
<tr>
<td>Plock,?</td>
<td><img src="image" alt="Wait 300ms" /></td>
</tr>
</tbody>
</table>

## Example Response

- **Plock,1**: Wait 300ms
- **Plock,0**: Wait 300ms
- **Plock,?**: Wait 300ms

**Example**: Plock,1 Baud, 9600

- **GND**: cannot change to UART
- **SDA**: cannot change to UART
- **SCL**: cannot change to UART

Copyright © Atlas Scientific LLC
I²C address change

Command syntax

I²C,n  sets I²C address and reboots into I²C mode

Example

I²C,100

Response

device reboot

Warning!

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

Default I²C address is 97 (0x61).

n = any number 1 – 127
Factory reset

Command syntax

Factory enable factory reset

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

I2C address will not change

Example

Response

Factory

device reboot

Clears calibration
LED on
Response codes enabled

Factory

(reboot)
## Change to UART mode

### Command syntax

**Baud,n** switch from I²C to UART

### Example

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

### Baud Options

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

---

---
Manual switching to UART

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

![Diagram showing manual switching process]

Wrong Example

![Diagram showing incorrect manual switching process]
In your CAD software place a 8 position header.

Place a 3 position header at both top and bottom of the 8 position. The two 3 position headers are now 17.78mm (0.7”) apart from each other.
Datasheet change log

Datasheet V 5.1
Revised response for the sleep command in UART mode on pg 36.

Datasheet V 5.0
Revised calibration theory on page 9, and added more information on the Export calibration and Import calibration commands.

Datasheet V 4.9
Corrected temperature compensation typo on pages 26 & 52.

Datasheet V 4.8
Revised isolation schematic on pg. 10

Datasheet V 4.7
Added new command:
"RT,n" for Temperature compensation located on pages 26 (UART) & 52 (I²C).
Added firmware information to Firmware update list.

Datasheet V 4.6
Added more information about temperature compensation on pages 26 & 52.

Datasheet V 4.5
Changed "Max rate" to "Response time" on cover page.

Datasheet V 4.4
Removed note from certain commands about firmware version.

Datasheet V 4.3
Added information to calibration theory on pg 7.

Datasheet V 4.2
Revised definition of response codes on pg 44.
Datasheet change log

Datasheet V 4.1
Updated firmware changes on pg. 66.

Datasheet V 4.0
Revised Enable/disable parameters information on pages 29 (UART) & 55 (I²C).

Datasheet V 3.9
Revised information on cover page.

Datasheet V 3.8
Update firmware changes on pg. 66.

Datasheet V 3.7
Revised Plock pages to show default value.

Datasheet V 3.6
Added new commands:
"Find" pages 21 (UART) & 48 (I²C).
"Export/Import calibration" pages 25 (UART) & 51 (I²C).
Added new feature to continuous mode "C,n" pg 22.

Datasheet V 3.5
Added accuracy range on cover page, and revised isolation info on pg. 10.

Datasheet V 3.4
Added manual switching to UART information on pg. 59.

Datasheet V 3.3
Updated firmware changes to reflect V1.99 update.

Datasheet V 3.2
Revised entire datasheet.
## Firmware updates

V1.1 – Initial release (Oct 30, 2014)
- Change output to mg/L, then percentage (was previously percentage, then mg/L).

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.

V1.7 – 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.

V1.97 – EEPROM (Oct 10, 2016)
- Fixed bug in the cal clear command, improves how it calculates the DO, adds calibration saving and loading.

V1.98 – EEPROM (Nov 14, 2016)
- Updated firmware for new circuit design.

V1.99 – (Feb 2, 2017)
- Revised “O” command to accept mg.

V2.10 – (April 12, 2017)
- Added "Find" command.
- Added "Export/import" command.
- Modified continuous mode to be able to send readings every "n" seconds.

V2.11 – (Sept 28, 2017)
- Fixed bug where the temperature would default to 0 on startup.

V2.12 – (Dec 19, 2017)
- Improved accuracy of dissolved oxygen equations.

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

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

The debugging phase

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

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