ORP EZO™

Circuit

Reads ORP

Range 
-1019.9mV – 1019.9mV

Accuracy
+/- 1mV

Max rate
1 reading per sec

Supported probes
Any type & brand

Calibration
Single point

Temp compensation
N/A

Data protocol
UART & I²C

Default I²C address
98 (0x62)

Operating voltage
3.3V – 5V

Data format
ASCII

PATENT PROTECTED

This is an evolving document, check back for updates.
STOP

SOLDERING THIS DEVICE VOIDS YOUR WARRANTY.

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

This device has been designed to be soldered and can be soldered at any time. Once that decision has been made, Atlas Scientific no longer assumes responsibility for the device’s continued operation. The embedded systems engineer is now the responsible party.

Get this device working in a solderless breadboard first!

Do not embed this device without testing it in a solderless breadboard!
# Table of contents

<table>
<thead>
<tr>
<th>Circuit dimensions</th>
<th>4</th>
</tr>
</thead>
<tbody>
<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>Operating principle</td>
<td>6</td>
</tr>
<tr>
<td>Calibration theory</td>
<td>8</td>
</tr>
<tr>
<td>Power and data isolation</td>
<td>9</td>
</tr>
<tr>
<td>Correct wiring</td>
<td>11</td>
</tr>
<tr>
<td>Available data protocols</td>
<td>14</td>
</tr>
</tbody>
</table>

## UART

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

## I²C

- I²C mode | 39 |
- Sending commands | 40 |
- Requesting data | 41 |
- Response codes | 42 |
- LED color definition | 43 |
- I²C quick command page | 44 |
- LED control | 45 |
- Find | 46 |
- Taking reading | 47 |
- Calibration | 48 |
- Export/import calibration | 49 |
- Device information | 50 |
- Reading device status | 51 |
- Sleep mode/low power | 52 |
- Protocol lock | 53 |
- I²C address change | 54 |
- Factory reset | 55 |
- Change to UART mode | 56 |
- Manual switching to UART | 57 |

- Circuit footprint | 58 |
- Datasheet change log | 59 |
- Warranty | 61 |

Copyright © Atlas Scientific LLC
**Power consumption**

<table>
<thead>
<tr>
<th>Voltage</th>
<th>LED</th>
<th>MAX</th>
<th>STANDBY</th>
<th>SLEEP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>5V</strong></td>
<td>ON</td>
<td>18.3 mA</td>
<td>16 mA</td>
<td>1.16 mA</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>13.8 mA</td>
<td>13.8 mA</td>
<td></td>
</tr>
<tr>
<td><strong>3.3V</strong></td>
<td>ON</td>
<td>14.5 mA</td>
<td>13.9 mA</td>
<td>0.995 mA</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>13.3 mA</td>
<td>13.3 mA</td>
<td></td>
</tr>
</tbody>
</table>

**Absolute max ratings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>MIN</th>
<th>TYP</th>
<th>MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage temperature (EZO™ ORP)</td>
<td>-65 °C</td>
<td>125 °C</td>
<td></td>
</tr>
<tr>
<td>Operational temperature (EZO™ ORP)</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>
EZO™ circuit identification

EZO™ ORP circuit

Legacy ORP circuit

Viewing correct datasheet

Viewing incorrect datasheet

Click here to view legacy datasheet
Operating principle

ORP stands for oxidation/reduction potential. Oxidation is the loss of electrons and reduction is the gain of electrons. The output of the probe is represented in millivolts and can be positive or negative.

Just like a pH probe measures hydrogen ion activity in a liquid; an ORP probe measures electron activity in a liquid. The ORP readings represent how strongly electrons are transferred to or from substances in a liquid. Keeping in mind that the readings do not indicate the amount of electrons available for transfer.
When reading the ORP of a liquid that has very few electrons available for transfer ORP readings can appear to be inconsistent.

The water is unreactive and has only trace amounts of electron movement. These readings are equivalent to the readings you see with an unconnected multimeter.

<table>
<thead>
<tr>
<th>Reading A</th>
<th>Reading B</th>
</tr>
</thead>
<tbody>
<tr>
<td>-234.6</td>
<td>24.2</td>
</tr>
</tbody>
</table>

Add just a drop of bleach (which is an oxidizing agent)

An ORP probe has a platinum tip that is connected to a silver wire, surrounded by silver chloride. That silver wire is then connected to a KCL reference solution. Because platinum is an unreactive metal it can “silently observe” the electron activity of the liquid without becoming apart of whatever reaction is occurring in the liquid.
Calibration theory

Calibration should be done at least once per year.
If the ORP that’s being read is continuously on the extremes of the scale (around -900mV or +900mV) calibration may have to be done more often. The exact frequency of calibration will have to be determined by your engineering team.

The Atlas Scientific EZO™ class ORP circuit has a flexible calibration protocol, allowing single point calibration to any off the shelf calibration solution.

Single point calibration

1. Remove soaker bottle and place probe in ORP calibration solution.
2. Let the probe sit in calibration solution until readings stabilize (10 – 60 seconds).
3. Calibrate to the value of the calibration solution using the command "Cal,n".

(If you are using the Atlas Scientific ORP calibration solution, calibrate to 225mV; "Cal,225").
Power and data isolation

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

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

Correct

Incorrect

Basic EZO™ Inline Voltage Isolator

Without isolation, Conductivity and Dissolved Oxygen readings will effect ORP 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 3.7V regardless of your input voltage.

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

Incorrect wiring

Bread board

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

Bread board via USB

Part # COM-104

Carrier board

USB carrier board

Part # ISCCB

Part # USB-ISO

NEVER use Perfboards or Protoboards

*Only after you are familiar with EZO™ circuits operation
NEVER EXTEND THE CABLE WITH CHEAP JUMPER WIRES!

DO NOT CUT THE PROBE CABLE WITHOUT REFERING TO THIS DOCUMENT!
DO NOT MAKE YOUR OWN UNSHIELDED CABLES!

ONLY USE SHIELDED CABLES. REFER TO THIS DOCUMENT!
Available data protocols

UART

I²C

Unavailable data protocols

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

#### Settings that are retained if power is cut
- Baud rate
- Calibration
- Continuous mode
- Device name
- Enable/disable 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
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

**RX**
- Data in

**TX**
- Data out

**Terminator**
- carriage return

**Data format**

- **Reading**: ORP
- **Units**: mV
- **Encoding**: ASCII
- **Format**: string
- **Data type**: floating point
- **Decimal places**: 1
- **Smallest string**: 2 characters
- **Largest string**: 40 characters
Default state

Mode

UART

Baud

9,600

Readings

continuous

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
Command

Carriage return <cr>
Terminator

Advanced

ASCII: 2 0 9 . 6 <cr>
Hex: 32 30 39 2E 36 0D
Dec: 50 48 57 46 54 13
Sending commands to device

2 parts

Command (not case sensitive)
ASCII data string

Carriage return <cr>
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

<table>
<thead>
<tr>
<th>Voltage</th>
<th>LED On</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>+2.2 mA</td>
<td></td>
</tr>
<tr>
<td>3.3V</td>
<td>+0.6 mA</td>
<td></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>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud</td>
<td>change baud rate</td>
<td>9,600</td>
<td>33</td>
</tr>
<tr>
<td>C</td>
<td>enable/disable continuous reading</td>
<td>enabled</td>
<td>24</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>n/a</td>
<td>26</td>
</tr>
<tr>
<td>Export/import</td>
<td>export/import calibration</td>
<td>n/a</td>
<td>27</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>n/a</td>
<td>35</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>n/a</td>
<td>23</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>n/a</td>
<td>29</td>
</tr>
<tr>
<td>I2C</td>
<td>change to I²C mode</td>
<td>not set</td>
<td>36</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>enabled</td>
<td>22</td>
</tr>
<tr>
<td>Name</td>
<td>set/show name of device</td>
<td>not set</td>
<td>28</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>disabled</td>
<td>34</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>n/a</td>
<td>25</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>n/a</td>
<td>32</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>n/a</td>
<td>31</td>
</tr>
<tr>
<td>*OK</td>
<td>enable/disable response codes</td>
<td>enable</td>
<td>30</td>
</tr>
</tbody>
</table>
LED control

Command syntax

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

Example | Response
--- | ---
L,1 <cr> | *OK <cr>
L,0 <cr> | *OK <cr>
L,? <cr> | ?L,1 <cr> or ?L,0 <cr> *OK <cr>
**Find**

**Command syntax**

*LED rapidly blinks white, used to help find device*

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

**Example**

| Find <cr> | *OK <cr> |

**Response**

This command will disable continuous mode.

Send any character or command to terminate find.

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>

*C,? <cr> continuous reading mode on/off?*

---

### Command syntax details

- **C,1** enable continuous readings once per second (default)
- **C,n** continuous readings every n seconds (n = 2 to 99 sec)*
- **C,0** disable continuous readings
- **C,?** continuous reading mode on/off?

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

### Example Response

#### C,1 <cr>

- *OK <cr>*
- ORP (1 sec) <cr>
- ORP (2 sec) <cr>
- ORP (n sec) <cr>

#### C,30 <cr>

- *OK <cr>*
- ORP (30 sec) <cr>
- ORP (60 sec) <cr>
- ORP (90 sec) <cr>

#### C,0 <cr>

- *OK <cr>*

#### C,? <cr>

- ?C,1 <cr> or ?C,0 <cr> or ?C,30 <cr>
- *OK <cr>
Single reading mode

Command syntax

R <cr> takes single reading

Example

Response

R <cr>

209.6 <cr> *OK <cr>

Green

Cyan

Taking reading

Transmitting

800 ms
Calibration

**Command syntax**

- **Cal,n**  <cr>  calibrates the ORP circuit to a set value
- **Cal,clear**  <cr>  delete calibration data
- **Cal,**  <cr>  device calibrated?

**Example** | **Response**
---|---
**Cal,225**  <cr> | ***OK**  <cr>
**Cal,clear**  <cr> | ***OK**  <cr>
**Cal,**  <cr> | **?Cal,0**  <cr>  or  **?Cal,1**  <cr>  
| ***OK**  <cr>

---

The EZO™ ORP circuit can be calibrated to any known ORP value.

---

Example Response:

- **Cal,225**  <cr>
- 209.6 mV
- **225**  <cr>
- **26**
### Export/import calibration

#### Command syntax

Export: Use this command to save calibration settings
Import: Use this command to load calibration settings to one or more devices.

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

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

#### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export,? <code>&lt;cr&gt;</code></td>
<td>10,120 <code>&lt;cr&gt;</code></td>
</tr>
<tr>
<td>Export <code>&lt;cr&gt;</code></td>
<td>59 6F 75 20 61 72 <code>&lt;cr&gt;</code> (1 of 10)</td>
</tr>
<tr>
<td>Export <code>&lt;cr&gt;</code></td>
<td>65 20 61 20 63 6F <code>&lt;cr&gt;</code> (2 of 10)</td>
</tr>
<tr>
<td>(7 more)</td>
<td>:</td>
</tr>
<tr>
<td>Export <code>&lt;cr&gt;</code></td>
<td>6F 6C 20 67 75 79 <code>&lt;cr&gt;</code> (10 of 10)</td>
</tr>
<tr>
<td>Export <code>&lt;cr&gt;</code></td>
<td><em>DONE</em></td>
</tr>
<tr>
<td>Import, n (FIFO)</td>
<td>Import, 59 6F 75 20 61 72 <code>&lt;cr&gt;</code> (1 of 10)</td>
</tr>
</tbody>
</table>

**Response breakdown**

- **10,** `<cr>`: # of strings to export
- **120**, `<cr>`: # of bytes to export

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

Disabling *OK simplifies this process.
Naming device

Command syntax

Name,n  <cr>  set name
Name,?  <cr>  show name

Example Response

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

*OK <cr>
### Device Information

#### Command Syntax

```
i  <CR>  device information
```

#### Example

<table>
<thead>
<tr>
<th></th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>?i, ORP, 1.97 &lt;CR&gt;</td>
</tr>
<tr>
<td></td>
<td>*OK &lt;CR&gt;</td>
</tr>
</tbody>
</table>

#### Response Breakdown

<table>
<thead>
<tr>
<th>?i,</th>
<th>ORP,</th>
<th>1.97</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>Firmware</td>
<td></td>
</tr>
</tbody>
</table>
Response codes

Command syntax

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

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
</table>
| R <cr>  | 209.6 <cr>  
|        | *OK <cr>    |
| *OK,0 <cr> | no response, *OK disabled |
| R <cr>  | 209.6 <cr>  
|        | *OK disabled |
| *OK,? <cr> | ?*OK,1 <cr> or ?*OK,0 <cr> |

Other response codes

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

These response codes cannot be disabled
## Reading device status

### Command syntax

<table>
<thead>
<tr>
<th>Command Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status &lt;cr&gt;</td>
<td>Voltage at Vcc pin and reason for last restart</td>
</tr>
</tbody>
</table>

### Example

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

### Response breakdown

<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 <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>STANDBY 16 mA</th>
<th>SLEEP 1.16 mA</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3V</td>
<td>13.9 mA</td>
<td>0.995 mA</td>
</tr>
</tbody>
</table>

Standby 16 mA

Sleep <cr>

Sleep 1.16 mA
Change baud rate

Command syntax

Baud, n <CR> change baud rate

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

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

Example Response

*Baud, 38400 <CR>*

Standby → Changing baud rate → *(reboot)* → Standby
Protocol lock

Command syntax

- **Plock,1** (cr)  enable Plock
- **Plock,0** (cr)  disable Plock  default
- **Plock,?** (cr)  Plock on/off?

Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plock,1 (cr)</td>
<td>*OK (cr)</td>
</tr>
<tr>
<td>Plock,0 (cr)</td>
<td>*OK (cr)</td>
</tr>
<tr>
<td>Plock,? (cr)</td>
<td>?Plock,1 (cr) or ?Plock,0 (cr)</td>
</tr>
</tbody>
</table>

Example Response

- Plock,1  Locks device to UART mode.
- I2C,100  cannot change to I2C
  *ER (cr)
Factory reset

Command syntax

Factory <cr> enable factory reset

Example

Factory <cr>  *OK <cr>

Response

Factory <cr>  *OK <cr>  *RS <cr>  *RE <cr>

Baud rate will not change

(reboot)
# Change to I²C mode

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2C,n  &lt;cr&gt;</td>
<td>sets I²C address and reboots into I²C mode</td>
</tr>
</tbody>
</table>

n = any number 1 – 127

## Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2C,100</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</td>
<td>*ER &lt;cr&gt;</td>
</tr>
</tbody>
</table>

n ≠ 127

Default I²C address 98 (0x62)

Example:

I2C,100

Response:

*OK <cr>

(reboot)

now in I²C mode
Manual switching to I²C

- Make sure Plock is set to 0
- Disconnect ground (power off)
- Disconnect TX and RX
- Connect TX to 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 98 (0x62)

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

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

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

I²C address (0x01 – 0x7F)

98 (0x62) default

Vcc

3.3V – 5.5V

Clock speed

100 – 400 kHz

SDA

SCL

VCC

4.7k resistor may be needed

0V

DATA FORMAT

Reading ORP

Units mV

Encoding ASCII

Format string

Data type floating point

Decimal places 1

Smallest string 2 characters

Largest string 399 characters
Sending commands to device

5 parts

Start | I²C address | Write | Command (not case sensitive) | Stop

ASCII command string

Example

Start | 98 (0x62) | Write | Sleep | Stop

I²C address

Command

Advanced

Address bits

The entire command as ASCII with all arguments

Start

40
Requesting data from device

Advanced

50 48 57 46 54 0 = 209.6
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

```c
I2C_start;
I2C_address;
I2C_write(EZO_command);
I2C_stop;

delay(300);
I2C_start;
I2C_address;
Char[] = I2C_read;
I2C_stop;
```

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

Response codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</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>

CPU

SCL  SDA

Receiving data

Send command

Processing delay

[Atlas Scientific] Environmental Robotics
LED color definition

Blue
I²C standby

Green
Taking reading

Purple
Changing
I²C ID#

Red
Command
not understood

White
Find

<table>
<thead>
<tr>
<th>5V</th>
<th>LED ON</th>
<th>+2.2 mA</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3V</td>
<td></td>
<td>+0.6 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>56</td>
</tr>
<tr>
<td>Cal</td>
<td>performs calibration</td>
<td>48</td>
</tr>
<tr>
<td>Export/import</td>
<td>export/import calibration</td>
<td>49</td>
</tr>
<tr>
<td>Factory</td>
<td>enable factory reset</td>
<td>55</td>
</tr>
<tr>
<td>Find</td>
<td>finds device with blinking white LED</td>
<td>46</td>
</tr>
<tr>
<td>i</td>
<td>device information</td>
<td>50</td>
</tr>
<tr>
<td>I²C</td>
<td>change I²C address</td>
<td>54</td>
</tr>
<tr>
<td>L</td>
<td>enable/disable LED</td>
<td>45</td>
</tr>
<tr>
<td>Plock</td>
<td>enable/disable protocol lock</td>
<td>53</td>
</tr>
<tr>
<td>R</td>
<td>returns a single reading</td>
<td>47</td>
</tr>
<tr>
<td>Sleep</td>
<td>enter sleep mode/low power</td>
<td>52</td>
</tr>
<tr>
<td>Status</td>
<td>retrieve status information</td>
<td>51</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>

### Example Response

<table>
<thead>
<tr>
<th>Command</th>
<th>Example Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>L,1</td>
<td>Wait 300ms, Dec 1, Null 0</td>
</tr>
<tr>
<td>L,0</td>
<td>Wait 300ms, Dec 1, Null 0</td>
</tr>
<tr>
<td>L,?</td>
<td>Wait 300ms, Dec 1, ASCII 1, Null 0 or Dec 1, ASCII 1, Null 0</td>
</tr>
</tbody>
</table>

---

**300ms processing delay**

---

![Atlas Scientific Environmental Robotics](image)
**Find**

**Command syntax**

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

300ms processing delay

**Example Response**

Send any character or command to terminate find.

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

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
</table>
| Find <cr> | ![Find LED Blinking](image1)

Wait 300ms

1 Dec 0 Null

![Atlas Scientific Logo](image2)
### Command syntax

**R** return 1 reading

### Example

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td><img src="taking-reading-I.png" alt="Green" /></td>
</tr>
<tr>
<td></td>
<td>Green: Taking reading</td>
</tr>
<tr>
<td></td>
<td>Wait 900ms</td>
</tr>
</tbody>
</table>

### Response

- **R**
- Dec: 1
- ASCII: 209.6
- Null: 0

900ms processing delay
# Calibration

## Command syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,n</td>
<td>calibrates the ORP circuit to a set value</td>
</tr>
<tr>
<td>Cal,clear</td>
<td>delete calibration data</td>
</tr>
<tr>
<td>Cal,?</td>
<td>device calibrated?</td>
</tr>
</tbody>
</table>

300ms processing delay

The EZO™ ORP circuit can be calibrated to any known ORP value

## Example

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal,225</td>
<td><img src="image1.png" alt="Graph showing Cal,225" /></td>
</tr>
<tr>
<td>Cal,clear</td>
<td><img src="image2.png" alt="Graph showing Cal,clear" /></td>
</tr>
<tr>
<td>Cal,?</td>
<td><img src="image3.png" alt="Graph showing Cal,?" /></td>
</tr>
</tbody>
</table>

### Example Response

- **Cal,225**
  - Wait 900ms
  - Dec 1, Null 0

- **Cal,clear**
  - Wait 300ms
  - Dec 1, Null 0

- **Cal,?**
  - Wait 300ms
  - Dec 1, ASCII ?Cal,0 0, Null 0 or Dec 1, ASCII ?Cal,1 0, Null 0
**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>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export,?</td>
<td>1 10,120 0</td>
</tr>
</tbody>
</table>

**Response breakdown**

10, 120  |
# of strings to export  # of bytes to export

Export strings can be up to 12 characters long

---

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export</td>
<td>1 59 6F 75 20 61 72 0</td>
</tr>
<tr>
<td>Export</td>
<td>1 65 20 61 20 63 6F 0</td>
</tr>
<tr>
<td>Export</td>
<td>1 <em>DONE</em> 0</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import, n</td>
<td>Import, 59 6F 75 20 61 72</td>
</tr>
</tbody>
</table>
## Device information

### Command syntax

- **i device information**

### Example

<table>
<thead>
<tr>
<th>i</th>
<th>?i, ORP, 19.7</th>
<th>1</th>
<th>ASCII</th>
<th>0</th>
<th>Null</th>
</tr>
</thead>
</table>

### Response breakdown

<table>
<thead>
<tr>
<th>?i, ORP, 1.97</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Firmware</td>
</tr>
</tbody>
</table>
# Reading device status

## Command syntax

| Status voltage at Vcc pin and reason for last restart | 300ms processing delay |

## Example

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

| P | powered off |
| S | software reset |
| B | brown out |
| W | watchdog |
| U | unknown |
# Sleep mode/low power

## Command syntax

<table>
<thead>
<tr>
<th>Sleep</th>
<th>enter sleep mode/low power</th>
</tr>
</thead>
</table>

**Example Response**

<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>

### Example

<table>
<thead>
<tr>
<th>Voltage</th>
<th>STANDBY</th>
<th>SLEEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V</td>
<td>16 mA</td>
<td>1.16 mA</td>
</tr>
<tr>
<td>3.3V</td>
<td>13.9 mA</td>
<td>0.995 mA</td>
</tr>
</tbody>
</table>

**Send any character or command to awaken device.**

**Do not read status byte after issuing sleep command.**
# Protocol lock

## Command syntax

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

### Example

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

### Example Response

- Locks device to I²C mode.
- 300ms processing delay
- Plock,1 Serial, 9600 cannot change to UART
- Cannot change to UART

---

53
**I²C address change**

**Command syntax**

$I2C,n$ sets I²C address and reboots into I²C mode

<table>
<thead>
<tr>
<th>Example</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I2C,100$</td>
<td>device reboot</td>
</tr>
</tbody>
</table>

**Warning!**

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

Default I²C address is 98 (0x62).

$n = \text{any number } 1 - 127$

---

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 the CPU is updated with the new I²C address.

Default I²C address is 98 (0x62).
Factory reset

Command syntax

| Factory enable factory reset | I²C address will not change |

Example

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

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

Clears calibration
LED on
Response codes enabled

Factory

(reboot)
Change to UART mode

**Command syntax**

Baud, n switch from I²C 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

Serial, 9600

Changing to UART mode

(reboot)
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

Wrong Example

Disconnect RX line
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 4.0
Revised definition of response codes on pg 42.

Datasheet V 3.9
Revised isolation information on pg 9.

Datasheet V 3.8
Revised Plock pages to show default value.

Datasheet V 3.7
Added new commands:
"Find" pages 23 & 46.
"Export/Import calibration" pages 27 & 49.
Added new feature to continous mode "C,n" pg 24.

Datasheet V 3.6
Revised circuit illustrations throughout datasheet.

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

Datasheet V 3.4
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)**
- Fix 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 glitch where EEPROM would get erased if the circuit lost power 900ms into startup

**V1.97 – EEPROM (Oct 10, 2016)**
- Fixed glitch in the cal clear command, improves how it calculates the ORP
- Added calibration saving and loading

**V2.10 – (May 9, 2017)**
- Added "Find" command.
- Added "Export/import" command.
- Modified continuous mode to be able to send readings every "n" seconds.
Warranty

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

The debugging phase

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

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