USB dO$_2$ isoPod™ (Model EPU354)

- Software controlled
- Plug and play with Pod-Vu software
- Electrical isolation
- Input ranges, from ±20 nA to ±20 µA
- Offset control to zero background signal

Description

An electrically isolated, compact signal conditioner with a USB/virtual serial port connection, for continuous monitoring of polarographic (Clark) dissolved oxygen electrodes.

Compatibility

This isoPod be used with most types of polarographic oxygen electrode, including the:

- ET1117 Polarographic oxygen electrode
- ET1120 Micro–oxygen electrode

Use with serial compatible software (RS232) or later, on Windows XP or later computers with USB port. A virtual serial port is created. Can be used with serial compatible software such as such as:

- Connect™, www.labtronics.com/DI/RS232_Software.htm
- WinWedge®, www.taltech.com/products/winwedge.html
- Tera Term, http://logmett.com/
- Pod-Vu, http://www.edaq.com/

or with your own software written in LabView, Visual Basic or C++ etc.

Applications

Ideal for chemical, biochemical, or physiological studies where continuous monitoring of an oxygen sensor is required. Electrical isolation minimizes interference with nearby pH, conductivity, ISE and similar sensors used in multi–parametric studies.

Theory of Operation

The dO$_2$ isoPod applies a polarizing voltage to a suitable oxygen sensor and measures the current resulting from the reduction of O$_2$ at the working (cathodic) electrode:

$$O_2 + 4e^- + 2H_2O \longrightarrow 4OH^-$$

Signal values, indicating the current flow, are output. Polarization can be selected between 0 and −1000 mV but usually a value of −800 mV is close to optimal.

The isoPod runs on DC power and can be used inside a Faraday cage for lowest noise operation.

Pod-Vu Software

A demonstration version of eDAQ Pod-Vu software is included on the isoPod Installer USB stick. If you have purchased Pod-Vu then a license code will also have been provided to enable all Pod-Vu features.

Pod-Vu will automatically configure the virtual serial ports and locate all connected USB isoPods. Refer to the Pod-Vu manual for operating instructions.

Pod-Vu will communicate with up to eight isoPod units, calibrate sensors, log data, and graphically display the signals in real time.

Specifications

Input ranges (and resolution):	20 nA　　　(6.25 fA) 200 nA　　(62.5 fA) 2 µA　　　(625 fA) 20 µA　　　(6.25 pA)
Polarization:	0 to −1000 mV
Zero offset:	±20 µA
Typical RMS noise:*	1 pA @ 1/s or slower
Connectors:	BNC (input), mini USB
Output data:	ASCII or 32 bit binary IEEE floating point
Output data:	Up to 100 /s
COM port settings:	115200 baud; 8 bits; 1 stopbit; no parity; flow NONE
Power:	< 50 mA from USB connector
Dimensions (l x w x h):	108 x 58 x 35 mm (4.25" x 2.28" x 1.38")
Weight:	200 g (7 oz)
Operating conditions:	0 – 40°C, 0 – 90% humidity (non condensing)

* On 20 nA range, with 1 Gohm load, inside a Faraday cage.

eDAQ Pty Ltd reserves the right to alter these specifications at any time.

www.**eDAQ**.com

E-mail: info@edaq.com

Serial Communication

The isoPod is fitted with a USB port. This can then be used as a virtual serial port with Windows, Mac OSX, and Linux computers. A Windows USB driver for most Windows computers is located on the isoPod Installer USB stick. Otherwise you can download a suitable driver from

http://www.ftdichip.com/Drivers/VCP.htm

Software can then be written to communicate with the isoPod as a serial (RS232) device, for example using LabView, Visual Basic, or C++.

The serial port number must first be established. Serial Ports 1 and 2 on Windows computers are reserved for the mother board. Thus it is common to find that the isoPod is located on serial port 3 (COM3) or greater.

The virtual serial port should be configured as 115200 baud, 8 bits, 1 stopbit, no parity. Set the flow control to NONE.

When correctly configured the unit will send the prompt **EPU354>** to indicate that a new command can be sent.

Interactive Communication

You can use terminal emulation software (eg Tera Term) to manually interact with the isoPod:

1. Download the Tera Term installer from http://logmett.com

2. Install Tera Term choosing the 'Compact Installation' option to reduce unnecessary extras.

3. Connect the isoPod and provide the computer with the isoPod Installer USB stick if a USB driver is requested.

4. Open the Windows device manager and find the isoPod and its corresponding COM port listed under 'Ports (COM & LPT)'. On first use there is no way to know in advance what COM port will be assigned to the isoPod, except that it will be COM3 or greater.

5. Start the Tera Term software, go to the Serial port... command in the Setup menu and configure the serial port as above. Click OK to return to the main Tera Term window and use the Return key to get the **EPU355>** prompt.

Once interactive communication is established you can be confident that the serial communications are working and can proceed to write your own software.

Serial Protocol

EPU354> help
EPU354> ?
Returns a list of commands.

EPU355> set range <r>
Sets the range in nanoamperes where **<r>** = 20, 200, 2000, or 20000. A confirmation string is returned.

EPU355> get range
Returns the range setting.

EPU355> set Vex <amount>
Set the excitation/poise voltage to **<amount>** between 0 and –1.000 V. A confirmation string is returned.

EPU355> get Vex
Returns the excitation/poise voltage in volts.

EPU355> set offset auto
Automatically set the incoming signal to zero nanoamperes.

EPU355> set offset <amount>
Applies an offset of **<amount>** nA up to 2000 nA to the signal.

EPU355> set offset 0
EPU355> set offset off
Removes any applied offset.

EPU355> set units
Sets the unit name, to be used after calibration. For example %saturation, ppmO2, nM, ng/L. A confirmation string is returned.

EPU355> get units
Returns the unit name.

EPU355> r
Return a single reading. Can be repeated at up to 10 Hz.

EPU355> v
Return a single value. That is, the reading is returned as a number, without extraneous text such as unit name etc. Can be repeated at up to 10 Hz.

EPU355> sample ascii <freq> [N]
EPU355> sample binary <freq> [N]
Return readings at a frequency of **<freq>**, an integer between 1 and 100 Hz. Readings are returned as ASCII or 32 bit binary IEEE floating point data. Specify an optional integer, **[N]**, to return that number of samples. Send ! to exit this mode. If **<freq>** is the character # then a single sample is returned every time # is sent.

EPU355> interval ascii <time> [N]
EPU355> interval binary <time> [N]
Return one sample every **<time>** seconds. Specify an optional integer, **[N]**, to return a fixed number of samples. Send ! to exit this mode.

EPU355> cal set <N> <value>
Defines the Nth calibration point, **<N>**, by taking a signal reading and assigning it to the **<value>** which is in the defined units. Note that **<N>** has a range of 1 – 15. If there is only **<N>** = 1 (single point calibration) then it is assumed that zero signal corresponds to zero amount. Linear regression is used to determine the line of best fit as the response curve.

EPU355> cal set <N> <value> <current>
Defines the Nth calibration point, **<N>**, by assigning a signal value of **<current>** in nanoamperes to a **<value>** which is in the defined units. Note that **<N>** has a range of 1 – 15. If there is only **<N>** = 1 (single point calibration) then it is assumed that zero signal corresponds to zero amount. Linear regression is used to determine the line of best fit as the response curve.

EPU355> cal remove <N>
Remove the Nth calibration point, **<N>**. Any subsequent calibration points are decremented. Calibration is redetermined with existing calibration points.

EPU355> cal remove all
Remove all calibration points.

EPU355> cal get
Returns all calibration information.

EPU355> res
Takes a reading of the current signal, then uses it to divide Vex and calculate resistance. Returns a string showing this calculation.

EPU355> version
Returns the firmware version number.

EPU355> prompt off
Turns off the 'EPU355>' prompt.

prompt on
Reinstates the 'EPU355>'prompt.

How to start

If you intend to write your own software using these protocols then it is a good idea to commence by sending each command manually to the isoPod using a terminal emulation program for example Tera Term, see above. This will confirm the responses expected, so that you can be sure that any recent changes in the protocols will be accounted for.