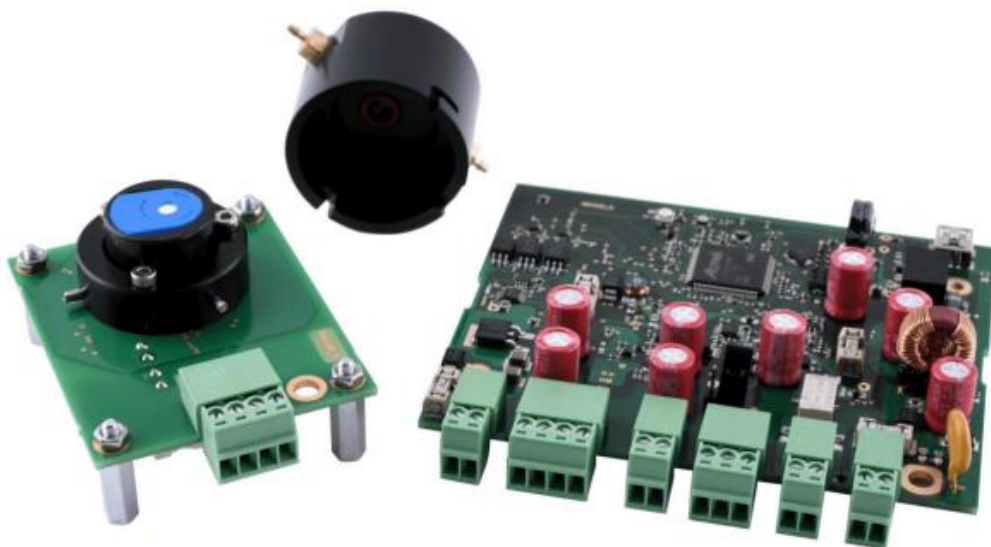




# Sensor Development Kit

Instrument User Manual V1.4



## SDK Quick Start Guide:

### Unboxing the SDK Contents

- 1 X Integration PCB
- 1 X Sensor PCB
- 1 X Gas Delivery Hood

### SDK Software

Download and install the SDK software from the SDK product page:

<https://www.ionscience.com/products/sensor-development-kit/>

### Connecting the SDK

Connect the Integration PCB and the sensor PCB together.

Insert MiniPID 2 into the socket on the sensor PCB.

Connect the integration PCB either via USB Mini-b to a Windows PC or via the RS485 connection to a Modbus master device.

Connect a 12 – 30 V dc +/- 500 mV power supply to the integration PCB.

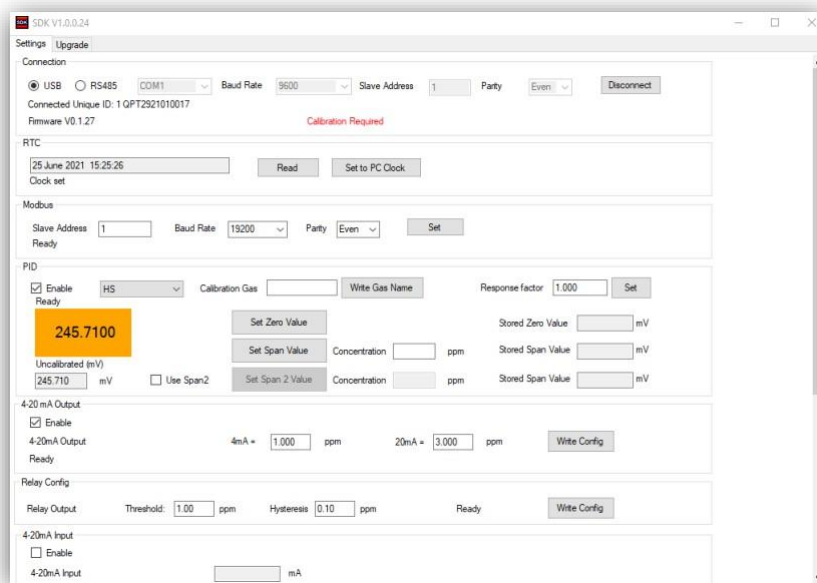
### The SDK Software

Select either the USB or RS485 option for the connection depending upon your connection type (RS485 only: Adjust the COM port, baud rate, slave address and parity)

Set the RTC (real time clock)

Enable the PID

Select the correct MiniPID 2 from the drop-down menu



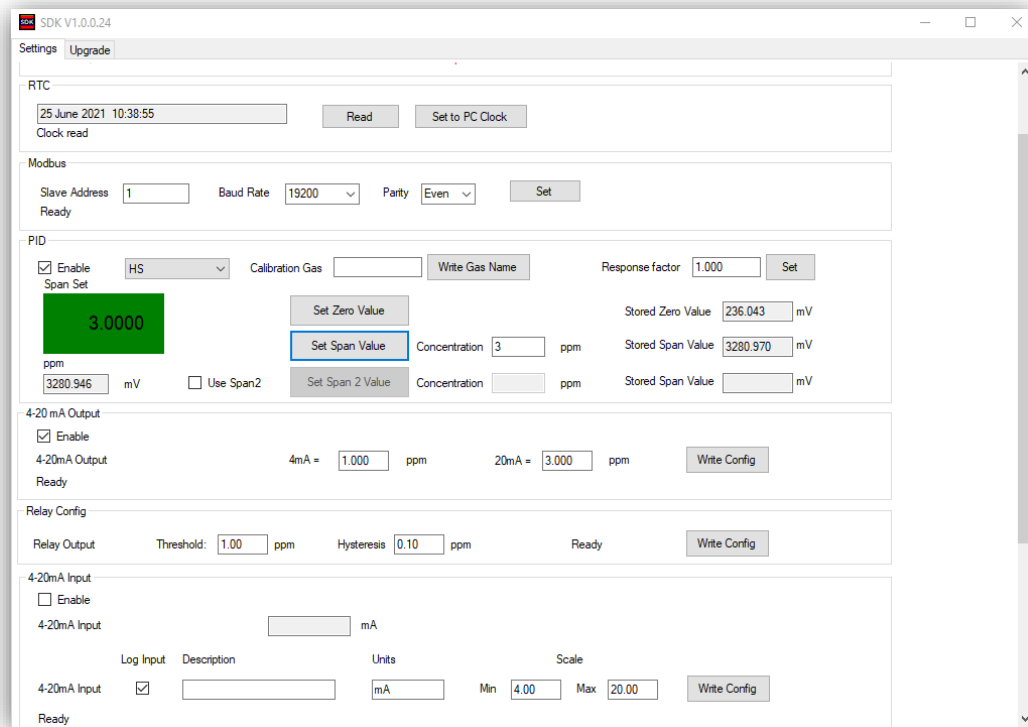
## Calibrating the MiniPID 2

When uncalibrated, the display reports a reading in mV.

Supply zero gas (for example: UHP synthetic air) and once the reading has stabilised, click the 'set zero value'

Supply the span gas (for example: 3 ppm Isobutylene in UHP synthetic air balance) and once the reading has stabilised, click the 'set span value'

The display box will now give a reading in ppm.



## Technical Support

If you require technical support with any aspect of the SDK, please contact

[sensors@ionscience.com](mailto:sensors@ionscience.com)

## Contents

<b>Quick Start Guide</b>	<b>2</b>
<b>Contents</b>	<b>4</b>
<b>Symbols</b>	<b>5</b>
<b>Recycling and Disposal</b>	<b>5</b>
<b>Statements</b>	<b>6</b>
Validity of this Manual .....	6
Responsibility for Correct Use .....	6
Warnings .....	6
Quality Assurance.....	6
Disposal .....	7
Legal Notice.....	7
Warranty .....	7
Contact details .....	8
<b>Introduction to the Sensor Development Kit (SDK)</b>	<b>9</b>
Technical Specification .....	10
Sensor PCB .....	10
Integration PCB .....	10
Gas Delivery Hood .....	10
Compatible MiniPID 2 Sensors.....	10
<b>System Description</b>	<b>10</b>
<b>Outputs and Communications</b>	<b>10</b>
Location Requirements .....	11
Power Requirements.....	11
Cable Requirements .....	11
Dimensions for Installation.....	11
Connections .....	12
<b>Installing the configuration software</b>	<b>13</b>
<b>Configuring the Integration PCB</b>	<b>13</b>
Connection .....	13
RTC.....	14
Modbus .....	14
PID Setup and Calibration.....	14
Procedure.....	15
Relay Configuration .....	16
4-20 mA Output .....	17
4-20 mA Input .....	17
Logging.....	17
<b>Fault Codes</b>	<b>18</b>
<b>LED Information</b>	<b>19</b>
<b>Firmware Update procedure</b>	<b>19</b>
<b>Spare Parts</b>	<b>20</b>
<b>Manual Log</b>	<b>21</b>
<b>Modbus Interface</b>	<b>22</b>
Introduction .....	22
Modbus .....	22
Interface .....	22
Function Codes.....	22
Coil Addresses .....	22
Error Returns.....	22
Function Code 03 (Read Holding Registers).....	23
Function Code 04 (Read Input Registers) .....	24
Function Code 16 (Preset Multiple Registers) .....	26
EEPROM Memory Use.....	28
Datalog Store .....	28
Calibration Store .....	30
Log Configuration .....	30
User Calibration.....	30

## Symbols



### WARNING!

USED TO INDICATE DANGER WARNINGS WHERE THERE IS A RISK OF INJURY OR DEATH.



### WARNING! - DANGER OF ELECTRIC SHOCK

USED TO INDICATE DANGER WARNINGS WHERE THERE IS A RISK OF INJURY OR DEATH FROM ELECTRIC SHOCK.



### CAUTION

USED TO INDICATE A CAUTION WHERE THERE IS A RISK OF DAMAGE TO EQUIPMENT.



### PROHIBITED ACTION

USED TO INDICATE ACTIONS THAT ARE NOT PERMITTED; E.G. 'YOU MUST NEVER'.



### INFORMATION

IMPORTANT INFORMATION OR USEFUL HINTS ABOUT USAGE.

## Recycling and Disposal



### RECYCLING

RECYCLE ALL PACKAGING.



### WEEE REGULATIONS

ENSURE THAT WASTE ELECTRICAL EQUIPMENT IS DISPOSED OF CORRECTLY.

## Statements

### Validity of this Manual

This User Manual gives information and procedures for the firmware version shown in the manual update log later in this manual.

If you have different versions of firmware, please obtain the correct User Manual.

### Responsibility for Correct Use

ION Science Ltd accepts no responsibility for incorrect adjustments, configurations or installations that cause harm or damage to people or property.

Use the equipment in accordance with this manual, and compliance with local safety standards.

Reduced performance of gas detection might not be obvious, so equipment must be inspected and maintained regularly. ION Science recommends:

- You use a schedule of regular checks to ensure it performs within calibration limits.
- you keep a record of calibration check data.

ION Science Ltd SDK boards are not authorized for use in safety-critical applications where a failure of the ION Science Ltd product would reasonably be expected to cause severe personal injury or death. Safety-critical applications include, without limitation, life support devices and systems, equipment or systems for operation in Hazardous Areas or Zones. ION Science Ltd products are neither designed nor intended for use in military applications. The Customer acknowledges and agrees that any such use of ION Science Ltd products is solely at the Customer's risk and that the Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

### Warnings

1. Read and understand this Manual fully before you install or operate the SDK.
2. For safety, the SDK must only be installed by qualified personnel.
3. All electrical work must be only carried out by competent people.
4. The substitution of components can result in unsafe conditions and will invalidate the warranty.
5. It is the responsibility of the user to ensure that PCBs are adequately protected from sources of EMI.
6. It is the responsibility of the user to ensure that PCBs are adequately protected from unsafe voltage conditions that could be transmitted into any connected equipment or systems.
7. It is the user's responsibility to assess risk when working with toxic and flammable compounds. Relevant protective equipment must be used.

### Quality Assurance

The SDK is manufactured using business systems complying with the ISO 9001 standard. That ensures that the equipment is:

- Designed and assembled reproducibly, from traceable components.
- Calibrated to the stated standards before it leaves the factory.

## **Disposal**

Dispose of the SDK and its components in accordance with all local and national safety and environmental requirements. This includes the European WEEE (Waste Electrical and Electronic Equipment) directive. ION Science Ltd offers a take-back service. Please contact us for more information.

## **Legal Notice**

Whilst every attempt is made to ensure the accuracy of the information contained in this manual, Ion Science accepts no liability for errors or omissions, or any consequences deriving from the use of information contained herein. It is provided "as is" and without any representation, term, condition or warranty of any kind, either expressed or implied. To the extent permitted by law, Ion Science shall not be liable to any person or entity for any loss or damage which may arise from the use of this manual. We reserve the right at any time and without any notice to remove, amend or vary any of the content which appears herein.

## **Warranty**

ION Science Ltd warrants that its products will conform to the Specifications.

This warranty lasts for one (1) year from the date of the sale.

ION Science Ltd shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or any products that have been altered or modified in any way by the Customer. Moreover, ION Science Ltd shall not be liable for any defects that result from the Customer's design, specifications, installations or instructions or incorrect interfacing for such products. Testing and other quality control techniques are used to the extent ION Science Ltd deems necessary.

If any ION Science Ltd product fails to conform to the warranty set forth above, ION Science Ltd.'s sole liability shall be to replace such products. ION Science Ltd.'s liability should be limited to products that are determined by ION Science Ltd not to conform to such a warranty. If ION Science Ltd elects to replace such products, ION Science Ltd shall be given a reasonable time to provide replacements. Replaced products shall be warranted for a new full warranty period. ION Science Ltd shall not be liable for any freight costs incurred.

In no event shall ION Science Ltd be liable to the Customer or any third party for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether ION Science Ltd has been advised of the possibility of such damages. This indemnity will survive the termination of the warranty period.

## Contact details

### UK Head Office

Ion Science Ltd  
The Hive, Butts Lane  
Fowlmere  
Cambridge  
SG8 7SL  
UNITED KINGDOM

Tel: +44 (0)1763 208503  
Fax: +44 (0) 1763 208814  
Email: [info@ionscience.com](mailto:info@ionscience.com)  
Web: [www.ionscience.com](http://www.ionscience.com)

### Italy Office

Ion Science Italia  
Via Emilia 51/c  
40011 Anzola Emilia  
Bologna  
ITALY

Tel: +39 051 0561850  
Fax: +39 051 0561851  
Email: [info@ionscience.it](mailto:info@ionscience.it)  
Web: [www.ionscience.it](http://www.ionscience.it)

### USA Corporate HQ

Ion Science Inc  
4153 Bluebonnet Drive  
Stafford  
TX 77477  
USA

Tel: +1 (877) 864 7710  
Email: [info@ionscienceusa.com](mailto:info@ionscienceusa.com)  
Web: [www.ionscienceusa.com](http://www.ionscienceusa.com)

### China Office

Ion Science China Ltd  
1101, Building B  
Far East International Plaza  
No. 317 Xiaxia Road  
Shanghai  
CHINA

Tel: +86 21 52545988  
Fax: +86 21 52545986  
Email: [info@ionscience.cn](mailto:info@ionscience.cn)  
Web: [ionscience.cn](http://ionscience.cn)

### Germany Office

Ion Science Messtechnik GMBH  
Laubach 30  
Metmann-Neandertal  
40822  
GERMANY

Tel: +49 2104 14480  
Fax: +49 2104 144825  
Email: [info@ism-d.de](mailto:info@ism-d.de)  
Web: [www.ism-d.de](http://www.ism-d.de)

### India Office

Ion Science India Pvt. Ltd  
#1-90/B/C/3/1, G-10 Charmy, Ganesh  
Nilayam, Vittal Rao Nagar  
Image Hospital Lane, Madhapur,  
Hyderabad – 500 081  
Telangana State  
INDIA

Tel: +91 40 48536129  
Email: [kschari@ionscience.com](mailto:kschari@ionscience.com)  
Web: [www.ionscience-india.com](http://www.ionscience-india.com)



## Introduction to the Sensor Development Kit (SDK)

The ION Science SDK consists of two PCBs designed to enable users to test and integrate MiniPID 2 VOC sensors into their applications.

The sensor PCB holds an ION Science MiniPID 2 sensor and is designed to be fitted with a gas delivery hood to allow easy introduction of gas to the sensor.

The integration PCB supplies power to the sensor PCB. The integration PCB can be configured to provide a 4 - 20 mA analogue output, a 4 - 20 mA input and a digital output via USB to the MS Windows based SDK application. The SDK can also be used as a Modbus slave device via RS485. The integration PCB stores calibration values for the sensor and will also log and store data from all inputs.

## Technical Specification

### Sensor PCB

Dimensions	50mm x 62mm
Weight	40g (72g when fitted with hood & PID)
Nominal Voltage	5 VDC $\pm$ 500mV
Supply Cables	0.5 to 1.5mm <sup>2</sup>
Operating Humidity:	0 – 99 RH% (non-condensing)
Operating Temperature	-20°C to +60°C

### Integration PCB

Dimensions	99mm x 82mm
Weight	70g
Nominal Voltage	12V to 30Vdc $\pm$ 500mV
Typical Power	<200mA when connected to a PID via the sensor PCB
Supply Cables	0.5 to 1.5mm <sup>2</sup>
Maximum Contact Load	100 Vac / 2A
Operating Humidity:	0 – 99 RH% (non-condensing)
Operating Temperature	-20°C to +60°C

### Gas Delivery Hood

Dimensions	Total height including PCB 40mm
Pipe Connection	1/16" OD barb push fit 1/16" ID/1/8" OD fluorinated tubing recommended
Seal material	Viton
Flow Rate (max)	<300 ml/min
Pressure (max)	<50mBar

## Compatible MiniPID 2 Sensors

Sensor Type	Part Number
MiniPID 2 PPM (3.6 – 10.0 V)	MP3SMLLCU2
MiniPID 2 PPM (3.6 – 18.0 V)	MP3SMLLNU2
MiniPID 2 PPM WR (3.6 – 10.0 V)	MP3SWMLLCU2
MiniPID 2 PPM WR (3.6 – 18.0 V)	MP3SWMLLNU2
MiniPID 2 PPB (3.6 – 10.0 V)	MP3SBLBCU2
MiniPID 2 PPB (3.6 – 18.0 V)	MP3SBLBNU2
MiniPID 2 PPB WR (3.6 – 10.0 V)	MP3SWBLBCU2
MiniPID 2 PPB WR (3.6 – 18.0 V)	MP3SWBLBNU2
MiniPID 2 HS (3.6 – 10.0 V)	MP3SHLHSCU2
MiniPID 2 HS (3.6 – 18.0 V)	MP3SHLHSNU2
MiniPID 2 10.0 eV (3.6 – 10.0 V)	MP3SBL0CU2
MiniPID 2 10.0 eV (3.6 – 18.0 V)	MP3SBL0NU2
MiniPID 2 11.7 eV (3.6 – 10.0 V)	MP3SB7BCU2
MiniPID 2 11.7 eV (3.6 – 18.0 V)	MP3SB7BNU2

## System Description

### Output and Communications

The integration PCB has:

- 1 x 4-20mA current loop output.
- 1 x 4-20mA current loop input.
- 1 x USB Interface.
- 1 x RS485 Modbus® channel.
- 1 x Programmable relay.

The 4-20mA output is an analogue representation of the calibrated sensor value.

You can program a normally open relay to operate at a chosen concentration of gas. This is set up in the companion software, available from [www.ionscience.com](http://www.ionscience.com). The relay can switch to a maximum of 250VAC / 2A maximum load.

The integration PCB can be configured as a Modbus® RS485 slave. For further details of Modbus® instructions, see the Modbus Interface section in this manual.

### Location Requirements

It is recommended that the integration PCB is housed in a metallic enclosure to shield against EMI from external sources.

### Power Requirements

The integration PCB must be powered from a stable 12-30VDC supply.

The Sensor PCB must be powered from a stable 5VDC supply.

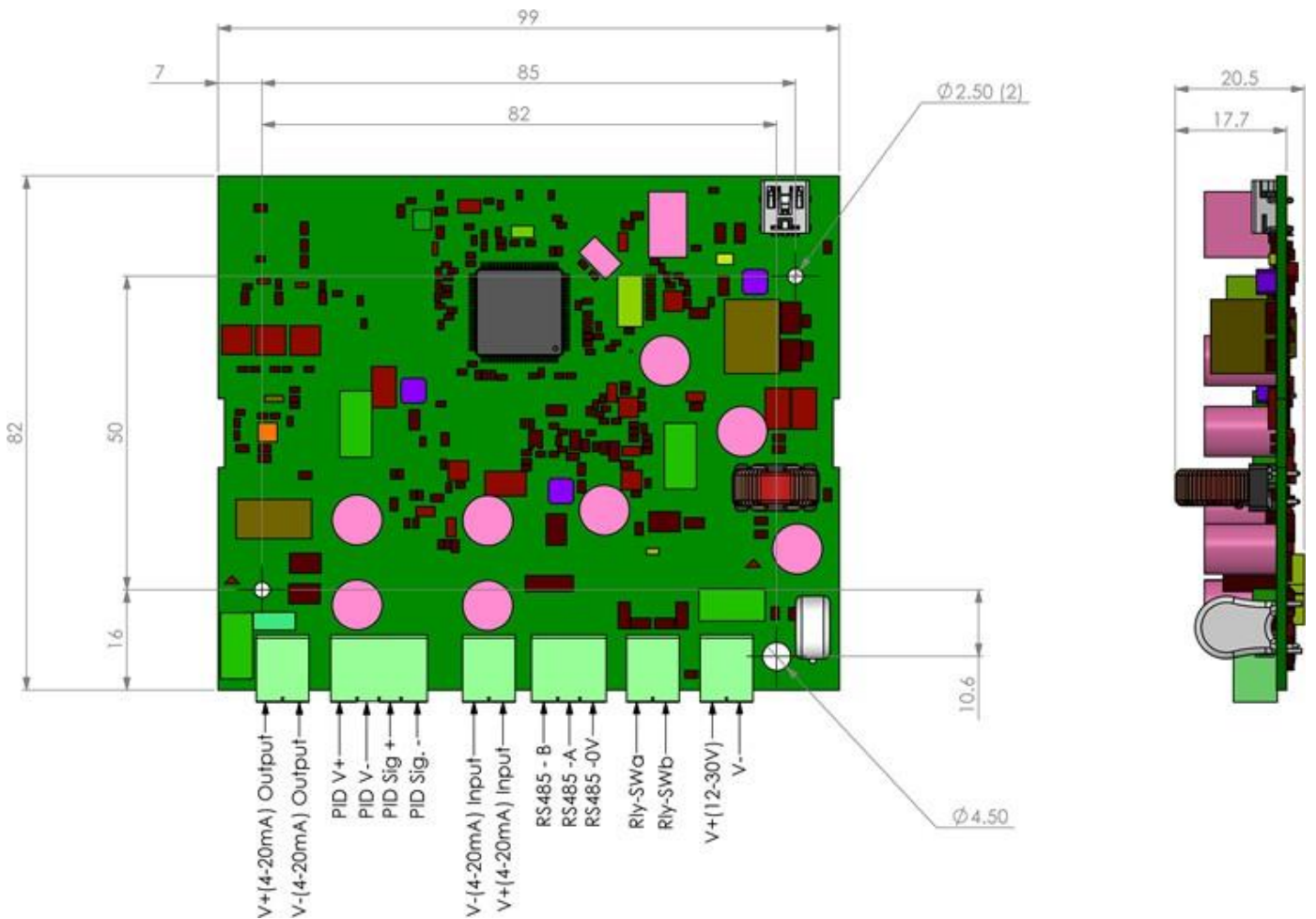
Modbus ADCMV field granularity voltage - LTC2485 using a 4.5V reference  $\sim 0.134\mu\text{V}$

### Cable Requirements

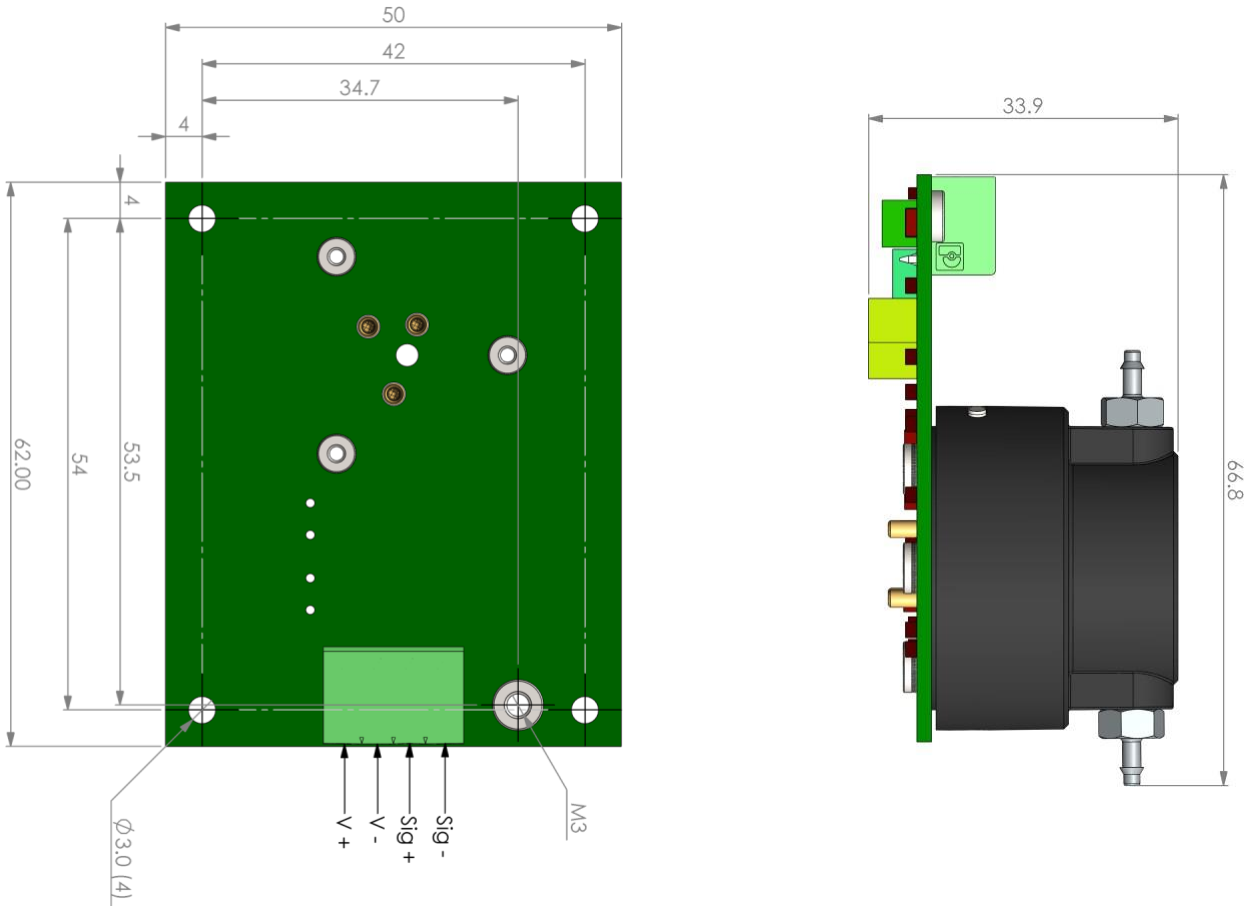
It is recommended that you use screened cables to protect against EMI.

### Dimensions for Installation

#### Integration PCB (Top and Side View)



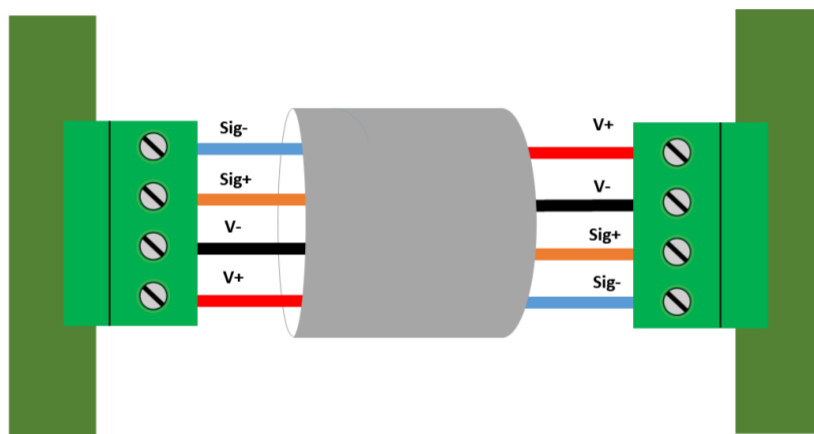
### Sensor PCB (Top and Side View)



**Note: maximum conductor size is 1.5mm<sup>2</sup>**

### Connections

The sensor PCB is connected to the integration PCB in the manner shown below. It is required that the cabling between the boards is shielded.



## Installing the configuration software

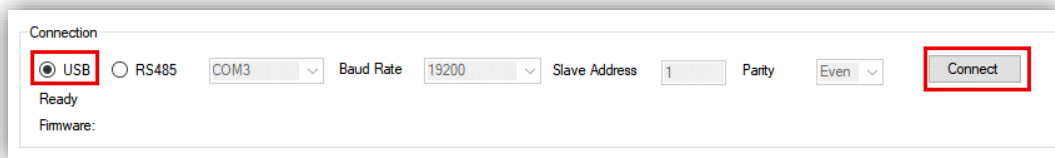
The integration PCB is configured using a Windows® based software tool. The software is available for download from [www.ionscience.com](http://www.ionscience.com). The software is compatible with Windows® 7, 8, 10 and 11.

1. Download the software.
2. Open the .zip file that contains the software.
3. Double click the setup.exe file.
4. The setup wizard will begin.
5. On the setup wizard dialogue, click “Next >”.
6. Use the “Browse” button to select a location for the wizard to install the integration PCB software or accept the default “C:\IonScience” folder and click “Next >”.
7. Choose a folder for the program shortcut and click “Next >”.
8. Select or deselect the tick box to create a desktop icon and click “Next >”.
9. Check the summary screen and if you are happy with the proposed structure, click “Install”.
10. Your software is now installed.

## Configuring the Integration PCB

### Connection

1. Before connecting the integration PCB to your PC, please ensure you have connected the sensor PCB, fitted with a MiniPID 2 sensor.
2. Supply power to the integration PCB using a suitable DC power supply (12-30Vdc).
3. Connect the integration PCB to your computer with a USB A to USB mini-B cable.



4. Open the configuration software and at the top of the page, select USB and click “Connect”.
5. When the integration PCB has connected successfully, you will see the Unique ID and firmware version of the board will appear. It is also possible to connect to the board via an RS485 adaptor.

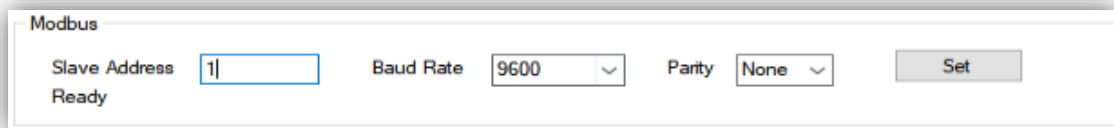


## RTC

- Set the real-time clock on the integration PCB (this is important for data logging accurately). To do this, in the RTC section, click “Set to PC clock”. You will see confirmation and the clock on the board will be synchronized with the clock on your PC.

## Modbus

1. In the Modbus section of the software, you can configure the integration PCB as a Modbus® slave device. This allows you to collect data from the boards remotely.
2. Set the slave device address to a value between 1 – 254. Each integration PCB on a Modbus® network must have a unique slave address.
3. Set the baud rate and parity to suit your network. Please note that all devices on the same Modbus® network must have the same baud and parity settings.
4. When the Modbus® settings are correct click “Set” to save the settings to the integration PCB.



## PID Setup and Calibration

An example of how to calibrate MiniPID 2 is given below:

- 1 x cylinder of ultra-high purity air (zero gas) fitted with a 300ml/min fixed flow regulator.
- 1 or 2 x cylinder of calibration gas (depending on the use of 2- or 3-point calibration) fitted with a 300 ml/min fixed flow regulator.
- Tubes to connect the regulators to the gas delivery hood. The barb on the gas delivery hood is suitable for 1/16” inner diameter tubing (Fluorinated tubing is recommended).
- A connected sensor PCB fitted with a MiniPID 2 sensor.

**Note: If you use a pump, the following must be considered.**

Pump upstream of Gas hood: Choose a pump with suitable materials to not affect the reading on MiniPID 2. Many pumps have soft parts that will adsorb and desorb VOCs.

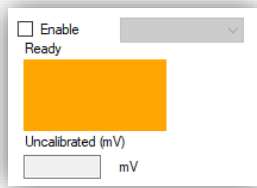
Pump downstream of Gas hood: Any leaks in the pneumatic system upstream of the gas hood will result in dilution of the sample.

The flow rate of the pump must match the flow rate of the fixed flow regulator.

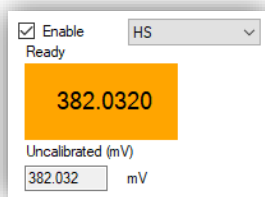
## Procedure

In the PID section of the SDK software, the integration PCB can be set up to read data from a MiniPID 2 sensor. The integration PCB can also be calibrated. This calibration is stored on the integration PCB meaning that the integration PCB will output a calibrated PPM (parts per million) value for all MiniPID 2 sensors.

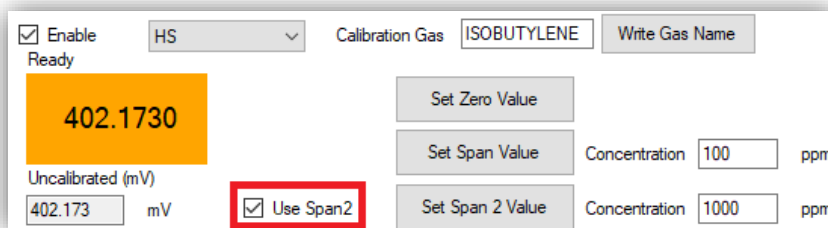
When the SDK is uncalibrated the PID reading box will be displayed as follows:



1. click the “Enable” checkbox, then select the correct sensor type from the drop-down menu. The raw signal in mV from the sensor is now displayed in the PID reading box.



2. If carrying out a 3-point calibration check the ‘Use Span2’ box.



3. Enter the calibration gas, e.g. ISOBUTYLENE. Confirm by pressing the ‘Write Gas Name’ button.
4. Enter the response factor for the gas and sensor combination you are using (for example Isobutylene = 1.0 and click ‘set’.
5. Connect zero air to the gas delivery hood and apply the air. Leave zero air applied to the MiniPID 2 for 2 minutes for the signal to stabilize. Note: If the sensor has been off for extended periods, longer stabilization time may be required.
6. After 2 minutes have elapsed, press the “Set Zero Value” button. This will store zero value in the internal memory on the integration PCB.

7. Enter the gas concentration for the first calibration span referring to the calibration gas cylinder. For example, 1.00 ppm of ISOBUTYLENE.

The screenshot shows the calibration software interface. At the top, there is a status bar with "Enable Ready" checked, a dropdown menu set to "HS", "Calibration Gas" set to "ISOBUTYLENE", a "Write Gas Name" button, and "Response factor" set to "1.000" with a "Set" button. Below this, a large orange box displays the current reading "1332.848". To the right, there are buttons for "Set Zero Value", "Set Span Value", and "Set Span 2 Value". The "Set Span Value" button is highlighted with a red box, and its corresponding input field contains "1.00" ppm. Below the main display, there is a section for "Uncalibrated (mV)" showing "1332.848" mV and a "Use Span2" checkbox which is currently unchecked. On the right side, there are fields for "Stored Zero Value" (344.254 mV), "Stored Span Value" (empty), and "Stored Span 2 Value" (empty).

8. Now, connect your cylinder of span gas to the gas hood and apply the gas. Leave the gas applied for 2 minutes for the signal to stabilize.
9. After 2 minutes have elapsed, press the “Set Span Value” button. This will store the span value in the internal memory on the integration PCB. If only using a two-point calibration, then calibration is complete.
10. If using a three-point calibration, enter the gas concentration for the second span gas calibration point. For example, 3.00 ppm ISOBUTYLENE.
11. Now, connect your span 2 ISOBUTYLENE cylinder to the gas hood and apply the gas. Leave the gas applied for 2 minutes.

The screenshot shows the calibration software interface after the first span value has been set. The "Uncalibrated (mV)" display now shows "234.465" mV. The "Use Span2" checkbox is now checked. The "Set Span 2 Value" button is highlighted with a red box, and its corresponding input field contains "3.00" ppm. The "Stored Span Value" field now contains "2066.081" mV, and the "Stored Span 2 Value" field is empty.

12. After 2 minutes have elapsed, press the “Set Span 2 Value” button. This will store the second span value in the SDK memory. Calibration is now complete.

## Relay Configuration

1. If you wish to use the onboard relay, the settings are found in the Relay Config settings box in the software. It is enabled by specifying the PPM level at which you wish the relay to activate. You also need to specify the level of hysteresis required.

The screenshot shows the Relay Configuration software interface. It features a "Relay Output" label, a "Threshold" input field set to "5.00" ppm, a "Hysteresis" input field set to "1.00" ppm, a "Ready" status indicator, and a "Write Config" button.

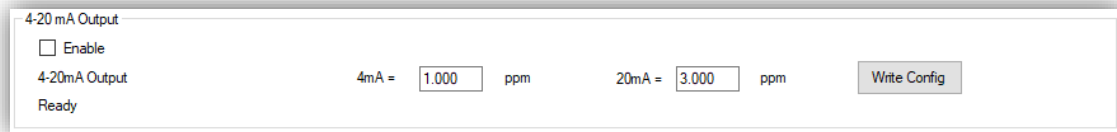
2. Once this is complete, click the “Write config” button. This will save all these settings to the Integration PCB.

*Please note the relay will not operate until the board is actively logging data.*



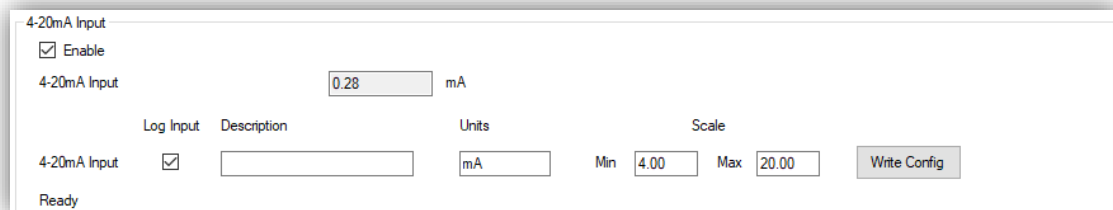
### 4-20 mA Output

The integration PCB has a single 4-20mA output channel which can output a mA signal as the mV signal from the MiniPID changes. The 4-20 mA scale can be defined such that each end of the scale can be between two concentrations. For example: 4 mA = 1 ppm and 20mA = 3 ppm.



### 4-20 mA Input

The integration PCB has a single 4-20 mA input channel that can be used to log data from other sensors and instrumentation. This data can be logged along with data from the PID sensor.



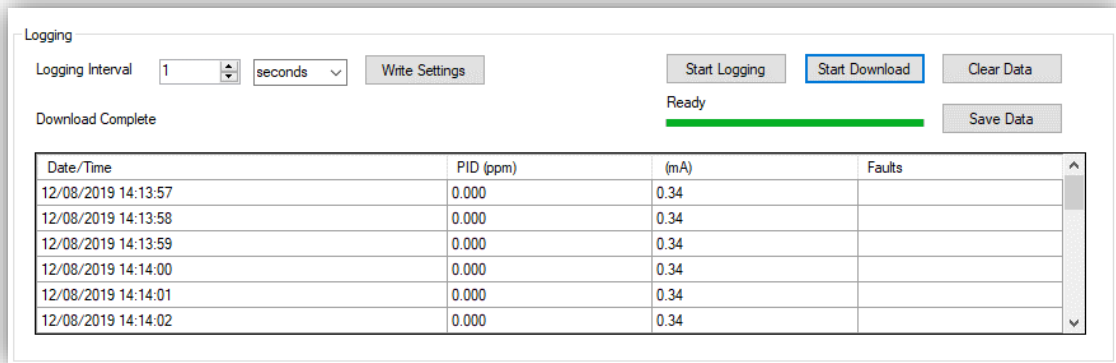
3. In the 4-20 mA Input config section, select “Enable” by clicking the checkbox.
4. Select the “Descriptions” text field and enter the name of the device you are logging.

### Logging

The integration PCB has 256KB of memory. The first 768 bytes are used to store calibration and configuration data, leaving roughly 255KB for data logs. The integration PCB memory runs on a circular buffer, meaning that when the memory becomes full, the oldest data will be overwritten first. The duration of data storage is dependent on the configured interval for data logging. Each data log entry is around 41 bytes.

1. Configure the logging interval by first selecting the units for either minutes or seconds.
2. Now enter the number of minutes or seconds you require the logging interval to be.
3. Click “Write Settings” to save the data logging settings to the integration PCB.
4. To start data logging click the “Start Logging” button.
5. To stop data logging click the “Stop Logging” button. If you wish to view the data logs you can either view them in the software or you can export them to a CSV file.
6. To view data logs within the software, click the “Start Download” button.

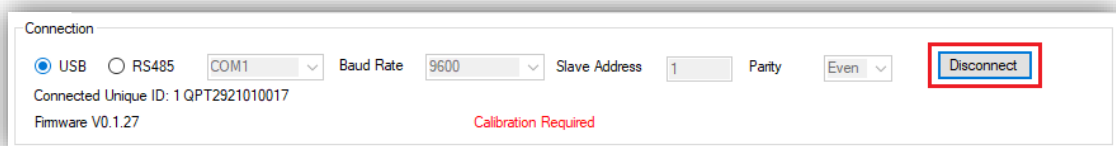
- To export a CSV file with all logged data, click “Save Data”. A save file dialogue will open so you can choose the name and save location of the downloaded CSV file.



### Fault Codes and Troubleshooting

Code	Text	Note
0	RTC Fault	Clock not set
1	Voltage Regulator Fault	PID power
5	PID overrange	PID signal overrange
6	Lamp Fault	Replace lamp
7	Oscillator not working (Error 2)	PID power
8	Oscillator Loaded (Error 1)	PID power
9	Power	PID power removed
10	Internal fault	Replace PID
11	MiniPID Removed	Reseat PID
12	PID Open Circuit	Reseat/ Replace stack
13	PID Short Circuit	Reseat/ Replace stack

If a prompt appears after a calibration has been completed, “Calibration Required”, disconnect, and reconnect to clear. If the message persists, recalibrate, disconnect, and reconnect the SDK software again.

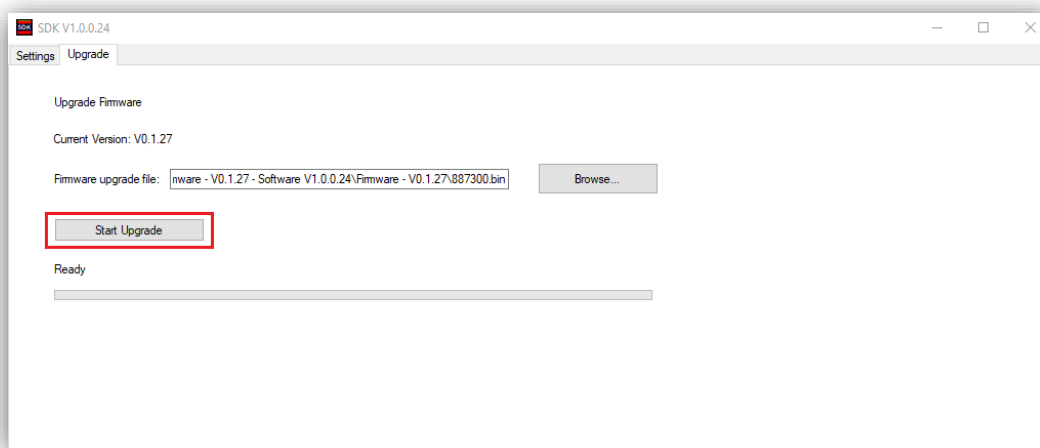
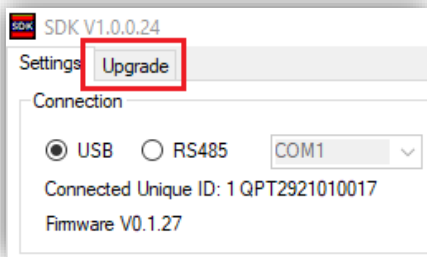


## LED Information

LED Output Table		
LED Location	Function	Description
Integration PCB	Power On	Flashes green, red, green, red for approx. 2 seconds then stops
	Logging	Solid green for duration of logging
	Updating	Rapid red flash upon commencing <1s. Multiple red flashes (approx. x5) followed by multiple green flashes (approx. x5), then green, red upon completion
	Up-Date Failed	Solid white
Sensor PCB	Enabled	Solid blue for duration while PID enabled

## Software and Firmware Update Procedure

For guidance on updating the SDK firmware, please download the latest version of the software and firmware from the Ion Science website. When updating the firmware, click 'Upgrade' and then 'Browse' and navigate to the correct firmware file and click 'Start Upgrade'.



## Spares and Accessories

Part Description	Part Number
MiniPID 2 PPM (3.6 – 10.0 V)	MP3SMLLCU2
MiniPID 2 PPM (3.6 – 18.0 V)	MP3SMLLNU2
MiniPID 2 PPM WR (3.6 – 10.0 V)	MP3SWMLLCU2
MiniPID 2 PPM WR (3.6 – 18.0 V)	MP3SWMLLNU2
MiniPID 2 PPB (3.6 – 10.0 V)	MP3SBLBCU2
MiniPID 2 PPB (3.6 – 18.0 V)	MP3SBLBNU2
MiniPID 2 PPB WR (3.6 – 10.0 V)	MP3SWBLBCU2
MiniPID 2 PPB WR (3.6 – 18.0 V)	MP3SWBLBNU2
MiniPID 2 HS (3.6 – 10.0 V)	MP3SHLHSCU2
MiniPID 2 HS (3.6 – 18.0 V)	MP3SHLHSNU2
MiniPID 2 10.0 eV (3.6 – 10.0 V)	MP3SBL0CU2
MiniPID 2 10.0 eV (3.6 – 18.0 V)	MP3SBL0NU2
MiniPID 2 11.7 eV (3.6 – 10.0 V)	MP3SB7BCU2
MiniPID 2 11.7 eV (3.6 – 18.0 V)	MP3SB7BNU2
MiniPID 2 Gas Delivery Hood	A-886215
MiniPID 2 PPM Electrode Stack (Blue)	A-846486
MiniPID 2 PPB Electrode Stack (White)	A-846267
MiniPID 2 HS Electrode Stack (Red)	A-846695
MiniPID 2 10.0 eV Electrode Stack (White + Gold)	A-846417
MiniPID 2 Electrode Stack Removal Tool	846216
MiniPID 2 Long Life Lamp (3.6V to 18V) 10.6eV	LA4SXL3.6
MiniPID 2 Series 4 Lamp 11.7eV (standard)	LA4SM7STD
MiniPID 2 Lamp Spring	846600
MiniPID 2 Lamp Cleaning Kit	A-31063

## Manual Update Log

Manual Version	Amendment	Issue Date	Instrument Firmware	PC Software
1.0	First Issue	27/11/18	V0.1.22	
1.1	Sensor information added (p8) Contact Details updated (p7)		V0.1.22	
1.2	Updated information for new software and firmware versions LED information (p17)		V.0.1.23	V.1.0.0.21
1.3	Updated information for new functionality, MiniPID 2 variants, software and hardware versions. Max flow rate and calibration information (p14) Extended modus commands added	15/01/21	V0.1.24	V1.0.0.23
1.4	Updated information on new compatible MiniPID 2 variants, software, and firmware versions. Updated information on new software and firmware functionality.	22/03/22	V0.1.27	V1.0.0.24

## Ion Science MiniPID SDK – Modbus Interface

### Introduction

This document details the Modbus interface for the MiniPID OEM Integration PCB. The same interface applies for connection via USB or RS485 interface.

### Modbus

The board uses Modbus RTU Mode. This document does not seek to explain the Modbus protocol. This detail is available elsewhere (e.g., [http://modbus.org/docs/PI\\_MBUS\\_300.pdf](http://modbus.org/docs/PI_MBUS_300.pdf)). The board behaves as a Modbus slave in the Master-slave relationship.

### Interface

When using a USB connection, the Modbus slave address of the board is ignored since USB is a single end-to-end connection.

When using RS485 multiple units may be connected provided they are configured with different slave addresses. When using RS485, the board supports the following configuration options:

Baud rate: 9600 or 19200

Parity: Even, Odd, none

Stop bits: 1 if parity is set, 2 if no parity

Slave address: 1 to 247

### Function Codes

The board supports the use of function codes

03 (Read Holding Registers)

04 (Read Input Registers)

16 (Preset multiple registers)

### Coil Addresses

The Coil addresses supported by the board depend upon the Function Code. These are tabulated in the following sections. If a non-existent coil is addressed an illegal address error is returned. Some addresses should be written or read in a block of a certain size in one transaction – where this is so it is noted in the address definition tables.

### Error returns

The board may return the following error codes:

01 Illegal Function

02 Illegal Address

04 Device error

### Function Code 03 (Read Holding Registers)

Name	Address (hex)	Length (decimal)(bytes)	Notes
MEM	1000	256	Reads data from board EEPROM from address specified in MEMADDR
MEMADDR	1100	4	Address of EEPROM to read when reading from MEM
MAX9611	1104	2	Reads directly the current register of the MAX96114-20mA input IC
RTC	1200	32	<p>Read the RTC from the unit. Returns a Date Time structure defined as:</p> <pre>typedef struct {     uint32_t year; // year value     uint32_t month; // month value     uint32_t day; // day value     uint32_t hour; // hour value     uint32_t minute; // minute value     uint32_t second; // second value     uint32_t DOW; // Day of week     uint32_t DOY; // Day of year } DateTime;</pre>
EXT420	1244	2	Returns a value indicating if the 4-20mA power is turned on. Returns 0 if not on, non-zero if on.
DIAGNOSTIC	124A	2	Reads value of diagnostic control. See Function Code 16 table for details.
MODBUSSETTINGS	1250	6	<p>Read the RS485 interface settings. Returns a structure defined as:</p> <pre>typedef struct {     short slave;     short baud;     short parity;     short sum; } MODBUS_SETTINGS;</pre> <p>The 'sum' is a simple sum of bytes the first 6 bytes of the structure. 'parity' = 1 for even, 2 for odd, anything else for no parity. 'baud' is 9600 or 19200.</p>

## Function Code 04 (Read Input Registers)

Name	Address (hex)	Length (decimal)(bytes)	Notes
UID	1000	16	Returns Unique ID of this board.
VERSIONS	1100	16	Returns firmware version string
PIDGOOD	1200	2	Returns 1 if PID Power OK, 0 if not
POWERGOOD	1202	2	Returns 1 if Power OK, 0 if not
READ420	1210	4	Returns 4-20mA calibrated input current as a long in micro-amps.
READ10V	1220	4	Returns 10V calibrated input in millivolts.
READR	1230	4	Returns RTD calibrated input in milli-ohms
READTEMPERATURE	1240	8	Returns structure defined as follows: <pre>struct {     int temperature;     int resistance; } readTemperature_value;</pre> 'temperature' is converted RTD input in milli-degC 'resistance' is calibrated RTD input in milli-ohms
READEMPTYTYPE	1250	4	Returns calibrated RTD type: 1 = Pt100 2 = Pt1000
ADCMV	1260	24	Returns array of 6 x 32 bits values. [0] = RTD - raw ADC input in $\mu$ V [1] = 10V - raw ADC input in $\mu$ V [2] = PID - raw ADC input in $\mu$ V [3] = LIGHT - raw ADC input in $\mu$ V [4] = TEMPERATURE - raw ADC input in $\mu$ V [5] = 4-20 - raw ADC input in ADC units
CALADCMV	1280	24	Returns array of 6 x 32 bits values: [0] = RTD - calibrated input in milli-ohms [1] = 10V - calibrated input in $\mu$ V [2] = PID - calibrated input in $\mu$ V [3] = LIGHT - calibrated input in $\mu$ V [4] = TEMPERATURE - raw ADC input in $\mu$ V [5] = 4-20 - calibrated input in $\mu$ A
COMPADCMV	12A0	24	Returns array of 6 x 32 bits values: [0] = RTD - temp. compensated input in milli-ohms [1] = 10V - temp. compensated input in $\mu$ V [2] = PID - temp. compensated input in $\mu$ V [3] = LIGHT - temp. compensated input in $\mu$ V [4] = TEMPERATURE - raw ADC input in $\mu$ V [5] = 4-20 - temp. compensated input in $\mu$ A
SIGNAL	12C0	24	Returns array of 6 x float values: [0] = RTD - Temperature in degC or degF



			<p>[1] = 10V - Scaled Input          [2] = PID - Signal in ppm          [3] = LIGHT - 1.0 if lamp on, 0.0 if lamp off          [4] = TEMPERATURE - Internal Temperature in degC          [5] = 4-20 - Scaled Input</p> <p>The "Scaled Inputs" are scaled according to the configured user calibration.</p>
LOGSTATE	1400	12	<p>Returns a structure:</p> <pre>typedef struct {     int state;     int counter;     int store_pointer; } LOG_CONTROL;</pre> <p>'state' = logging state          0 = Idle          1 = Starting log          2 = Stopping log          3 = logging          'counter' = Increments once per second counter up to next log interval          'store_pointer' = Incrementing counter of number of items stored during run.</p>
STORE	1500	16	<p>Returns a structure:</p> <pre>struct {     unsigned int store_address;     unsigned int start_store_address;     unsigned int store_length;     unsigned int start_offset; } store_pointers;</pre> <p>'store_address' = address of next space to be written in EEPROM          'start_store_address' = address of start of logged data in EEPROM          'store_length' = maximum space in EEPROM for logged data (bytes)          'start_offset' = lowest EEPROM address to be used for logged data.</p> <p>This data is used to download logged data from EEPROM</p>

### Function Code 16 (Preset multiple registers)

Name	Address (hex)	Length (decimal)(bytes)	Notes
MEM	1000	256	Writes data to board EEPROM at address specified in MEMADDR. <b>This should be used with care (if at all) as writing to certain locations will over-write factory calibration data and require the board returning to factory to restore correct operation. See later section for EEPROM memory definition.</b>
MEMADDR	1100	4	Specify address of EEPROM to read when reading or writing from MEM
MAX9611	1104	2	Writes the range directly the control register of the MAX96114-20mA input IC
RTC	1200	32	Set the RTC in the unit. Send a Date Time structure defined as:  <pre>typedef struct {     uint32_t year; // year value     uint32_t month; // month value     uint32_t day; // day value     uint32_t hour; // hour value     uint32_t minute; // minute value     uint32_t second; // second value     uint32_t DOW; // Day of week     uint32_t DOY; // Day of year } DateTime; The DOW and DOY values are ignored.</pre>
LTC2640	1240	4	Write to output DAC. Writes to an array: <pre>static short ltc2640_data[2];</pre> The first short is DAC channel 0 = 4-20mA output, 1 = 10V output. The second short is the DAC value to write (0 to 4095)
EXT420	1244	2	Control Power to 4-20mA input circuit. Send 0 to power off, non-zero to power on.
PIDSUPPLY	1248	2	Control Power to PID. Send 0 to power off, non-zero to power on.
DIAGNOSTIC	124A	2	Set value to diagnostic control. Lower two bits of value written have significance: Bit 0: set to ramp 4-20mA output Bit 1: set to ramp 0-0V output

MODBUSSETTINGS	1250	6	<p>Write the RS485 interface settings. Send a structure defined as:</p> <pre>typedef struct {     short slave;     short baud;     short parity;     short sum; } MODBUS_SETTINGS;</pre> <p>The 'sum' is a simple sum of bytes the first 6 bytes of the structure. 'parity' = 1 for even, 2 for odd, anything else for no parity. 'baud' is 9600 or 19200.</p>
OUT420	1300	2	Output a current on the 4-20mA output. Write the current in $\mu$ A
LOGSTARTSTOP	1400	2	Control logging. Write 0 to stop logging. Write 1 to start logging.
ERASESTORE	1500	2	Erase log store. Write a 1 to carry out erase. Note: You cannot erase store if log state is anything other than Idle.

## EEPROM memory use

The Board has 256Kbytes of EEPROM memory. Most of this is available for holding data logs. The first 768 bytes are used to hold calibration data. **If this area is written incorrectly the board may malfunction and will need returning for repair.**

## Data log store

The data log store uses all the available EEPROM space outside of that used for calibration. Once the store becomes full, the logging wraps around and over-writes the oldest data.

To download the logged data, you first need to find out where the start of the store is (due to possible wrap around).

Use Modbus read STORE to return the information:

```
struct
{
  unsigned int store_address;
  unsigned int start_store_address;
  unsigned int store_length;
  unsigned int start_offset;
} store_pointers;
'store_address' = address of next space to be written in EEPROM
'start_store_address' = address of start of logged data in EEPROM
'store_length' = maximum space in EEPROM for logged data (bytes)
'start_offset' = lowest EEPROM address to be used for logged data
```

You then know to start reading data from EEPROM address '*start\_store\_address*' and finish when you get to '*store\_address*'. Incrementing the address by 256 bytes for each block read. If the address reaches the end address ('*start\_offset*' + '*store\_length*') then reset to '*start\_offset*'. (Note: If the unit is still logging, these might change so it is best to Stop Logging before downloading data if the memory has wrapped).

Data is stored in 256-byte blocks, which is the sector size of the EEPROM and the maximum size of a Read from the EEPROM over Modbus.

To Read a block first write the EEPROM address you want to read to MEMADDR, then do a 256-byte read from MEM.

Each block has its own internal checksum (independent of the checksum used in the Modbus transfer). The last two bytes, taken as a 16-bit value, hold the sum of all the previous 254 bytes in the block. A simple routine to check the sum written in C# is:

```
private bool CheckBlockSum(byte [] block)
{
  ushort sum = 0;
  for(int i=0; i<block.Length - 2; i++)
  {
    sum += (ushort)block[i];
  }
  ushort insum = BitConverter.ToUInt16(block, block.Length - 2);
  return (insum == sum);
}
```

For a used block, the first 16-bits should form the value 0x5AA5. If it does not have this value, it is not a valid log block and should be discarded.

The remaining of the block holds the logged data in the form of several records. The first byte of the record indicates the record content, and by implication, its length. So, by parsing each record in turn the block can be decoded.

The records are as follows:

Record Tag (hex)	Total length (including Tag)	Content
0	1	Block filler – ignore and move on
FE	5	Indicates logging stopped due to power fail. 4-bytes form time logging was stopped as number of seconds since 1/1/2010 00:00:00
FF	5	Indicates logging stopped by user. 4-bytes form time logging was stopped as number of seconds since 1/1/2010 00:00:00
1	5 + (number channels) * 4	Indicates normal log record. Length depends on number of channels configured to log. Bytes 1-4: integer - time of log as number of seconds since 1/1/2010 00:00:00 Bytes 5-8: float - PID signal in ppm Bytes 9-12: float – scaled 10V input signal Bytes 13-16: float – scaled 4-20mA input signal Bytes 17-20: float – scaled RTD Temperature  If any of the channels are not configured for input then the log record will be shorter by 4 bytes for each channel not configured and the subsequent channels will move down.
2	10	Fault record. Bytes 1-4: integer - time of fault as number of seconds since 1/1/2010 00:00:00 Byte 5: integer – fault number Byte 6-9: integer – value associated with the fault
3	10	Fault cleared record Bytes 1-4: integer - time of fault as number of seconds since 1/1/2010 00:00:00 Byte 5: integer – fault number Byte 6-9: integer – value associated with the fault
4	5	Start of logging 4-bytes form time logging was started as number of seconds since 1/1/2010 00:00:00

## Calibration Store

Calibration information is stored in the first 3 blocks of the EEPROM.

**Warning: If this is written incorrectly the board may malfunction and will need returning for repair.**

The only user locations are documented below. Data should **not** be written to any EEPROM locations outside of those described.

## Log Configuration

At address 180 hex is stored in the log configuration block. This is defined by the structure:

```
typedef struct
{
    int version;
    int validity;
    int interval;
    int mode;
    int sum;
} LOG_CONFIG;
```

'version' should be set to 0x55AA0001

'validity' should be set to 1

'interval' is the log interval in seconds

'mode' should be set to 0

'sum' is a sum of the first 16 bytes of the structure.

The log configuration can be read by writing 180hex to MEMADDR and then reading 20 bytes from MEM. Likewise, it can be written by writing 180hex to MEMADDR and then writing the 20 bytes block to MEM.

## User calibration

User calibration data is held at EEPROM address 200 hex.

The information is defined by the following structures:

```
typedef struct
{
    unsigned short valid;
    unsigned short enable;
    float min;
    float max;
    long zero;
    long span;
    int decimals;
    char units[24];
    char label[32];
} CHANNEL_CALIBRATION;
```

'valid': should be set to 0x55AA

'enable': set to 1 to enable logging, set to 0 to disable

'min': is the minimum scaled value of the channel (set to 0.0 for PID)

'max': is the maximum scaled value of the channel (set to the calibration gas ppm for PID)

'zero': is the input channel zero value (0 for 10V input, 4000 for 4-20mA input, zero mV signal for PID)

'span': is the input channel span value (10000000 for 0V input, 20000 for 4-20mA input, span mV signal for PID)

'decimals': is number of decimal points to use for scaled signal

'units': is displayed units for channel (set to 'ppm' for PID)

'label': is channel label

```
typedef struct
{
    unsigned int version;
    unsigned int validity;

    CHANNEL_CALIBRATION CalibPID;
    CHANNEL_CALIBRATION Calib420;
    CHANNEL_CALIBRATION Calib10V;

    unsigned short enable_rtd;
    unsigned short rtd_units;           // 0 = degC, 1 = degF

    unsigned int sum;
} USERCALIBRATION;

'version' should be set to 0x55AA0001
'validity' should be set to 7
'enable_rtd': set to 1 to enable logging RTD channel, set to 0 to disable
'rtd_units': set to 1 for degF, set to 0 for degC
'sum' is sum of the first 252 bytes of the block.
```

The USERCALIBRATION block is exactly 256 bytes long.

You can read the existing user calibration by writing 200 hex to MEMADDR and then reading 256 bytes from MEM. You can write new user calibration by writing 200 hex to MEMADDR then writing the 256-byte block to MEM.