

# Gases 2.0

## Technical Guide



Document version: v4.8 - 11/2014  
© Libelium Comunicaciones Distribuidas S.L.

# INDEX

<b>1. General .....</b>	<b>5</b>
1.1. General and safety information .....	5
1.2. Conditions of use .....	5
<b>2. Waspote Plug &amp; Sense!.....</b>	<b>6</b>
2.1. Features .....	6
2.2. Sensor Probes.....	6
2.3. Solar Powered .....	7
2.4. Programming the Nodes.....	8
2.5. Radio Interfaces .....	9
2.6. Program in minutes.....	10
2.7. Data to the Cloud.....	10
2.8. Meshlium Storage Options.....	11
2.9. Meshlium Connection Options.....	11
2.10. Models .....	12
2.10.1. Smart Environment.....	13
<b>3. Hardware.....</b>	<b>15</b>
3.1. General Description .....	15
3.2. Specifications .....	15
3.3. Electrical Characteristics.....	16
<b>4. Sensors .....</b>	<b>17</b>
4.1. General considerations in the use of the sensors.....	17
4.2. Starting with the gas sensors .....	18
4.2.1. Sensor calibration .....	18
4.2.2. Gain and load resistor configuration.....	18
4.2.3. Converting the read data .....	19
4.2.4. Calibrated sensors performance.....	19
4.3. Humidity Sensor – 808H5V5 .....	20
4.3.1. Specifications.....	20
4.3.2. Measurement Process.....	20
4.4. Temperature Sensor – MCP9700A .....	21
4.4.1. Specifications.....	21
4.4.2. Measurement Process.....	21
4.5. Atmospheric Pressure Sensor - MPX4115A .....	23
4.5.1. Specifications.....	23
4.5.2. Measurement Process.....	23

4.6. Carbon Monoxide (CO) Sensor – TGS2442.....	24
4.6.1. Specifications.....	24
4.6.2. Measurement Process.....	24
4.7. Carbon Dioxide (CO <sub>2</sub> ) Sensor – TGS4161.....	25
4.7.1. Specifications.....	25
4.7.2. Measurement Process.....	26
4.8. Molecular Oxygen (O <sub>2</sub> ) Sensor – SK-25.....	27
4.8.1. Specifications.....	27
4.8.2. Measurement Process.....	27
4.9. Nitrogen Dioxide (NO <sub>2</sub> ) Sensor - MiCS-2710.....	28
4.9.1. Specifications.....	28
4.9.2. Measurement Process.....	28
4.10. Nitrogen Dioxide (NO <sub>2</sub> ) Sensor - MiCS-2714.....	29
4.10.1. Specifications.....	29
4.10.2. Measurement process.....	29
4.11. Ammonia (NH <sub>3</sub> ) sensor – TGS2444.....	30
4.11.1. Specifications.....	30
4.11.2. Measurement Process.....	31
4.12. Methane (CH <sub>4</sub> ) sensor – TGS2611.....	32
4.12.1. Specifications.....	32
4.12.2. Measurement Process.....	32
4.13. Liquefied Petroleum Gas Sensor - TGS2610.....	33
4.13.1. Specifications.....	33
4.13.2. Measurement Process.....	33
4.14. Air Contaminants Sensor - TGS2600.....	34
4.14.1. Specifications.....	34
4.14.2. Measurement Process.....	34
4.15. Air Contaminants Sensor - TGS2602.....	36
4.15.1. Specifications.....	36
4.15.2. Measurement Process.....	36
4.16. Solvent Vapors Sensor - TGS2620.....	37
4.16.1. Specifications.....	37
4.16.2. Measurement Process.....	37
4.17. Ozone (O <sub>3</sub> ) Sensor - MiCS-2610.....	38
4.17.1. Specifications.....	38
4.17.2. Measurement Process.....	39
4.18. Ozone (O <sub>3</sub> ) Sensor - MiCS-2614.....	40
4.18.1. Specifications.....	40
4.18.2. Measurement process.....	40
4.19. VOC's Sensor - MiCS-5521.....	41
4.19.1. Specifications.....	41
4.19.2. Measurement Process.....	41
4.20. VOC's Sensor - MiCS-5524.....	42
4.20.1. Specifications.....	42
4.20.2. Measurement process.....	43

---

4.21. Design and connections .....	44
4.21.1. Connector 1 .....	44
4.21.2. Connector 2 .....	45
4.21.3. Connector 3 .....	46
4.21.4. Connector 4 .....	47
4.21.5. Sockets for casing .....	48
<b>5. Board configuration and programming .....</b>	<b>51</b>
5.1. Hardware configuration .....	51
5.2. API.....	51
<b>6. Consumption .....</b>	<b>55</b>
6.1. Power control .....	55
6.2. Consumption table.....	55
6.3. Low Consumption Mode.....	55
<b>7. API Changelog .....</b>	<b>56</b>
<b>8. Documentation changelog .....</b>	<b>57</b>
<b>9. Maintenance .....</b>	<b>58</b>
<b>10. Disposal and Recycle.....</b>	<b>59</b>

# 1. General

## 1.1. General and safety information

In this section, the term “Waspote” encompasses both the Waspote device itself and its modules and sensor boards.

- Read through the document “General Conditions of Libelium Sale and Use”.
- Do not allow contact of metallic objects with the electronic part to avoid injuries and burns.
- NEVER submerge the device in any liquid.
- Keep the device in a dry place and away from any liquid which may spill.
- Waspote consists of highly sensitive electronics which is accessible to the exterior, handle with great care and avoid bangs or hard brushing against surfaces.
- Check the product specifications section for the maximum allowed power voltage and amperage range and consequently always use a current transformer and a battery which works within that range. Libelium is only responsible for the correct operation of the device with the batteries, power supplies and chargers which it supplies.
- Keep the device within the specified range of temperatures in the specifications section.
- Do not connect or power the device with damaged cables or batteries.
- Place the device in a place only accessible to maintenance personnel (a restricted area).
- Keep children away from the device in all circumstances.
- If there is an electrical failure, disconnect the main switch immediately and disconnect that battery or any other power supply that is being used.
- If using a car lighter as a power supply, be sure to respect the voltage and current data specified in the “Power Supplies” section.
- If using a battery in combination or not with a solar panel as a power supply, be sure to use the voltage and current data specified in the “Power supplies” section.
- If a software or hardware failure occurs, consult the Libelium Web **Development section**.
- Check that the frequency and power of the communication radio modules together with the integrated antennas are allowed in the area where you want to use the device.
- Waspote is a device to be integrated in a casing so that it is protected from environmental conditions such as light, dust, humidity or sudden changes in temperature. The board supplied “as is” is not recommended for a final installation as the electronic components are open to the air and may be damaged.

## 1.2. Conditions of use

- Read the “General and Safety Information” section carefully and keep the manual for future consultation.
- Use Waspote in accordance with the electrical specifications and the environment described in the “Electrical Data” section of this manual.
- Waspote and its components and modules are supplied as electronic boards to be integrated within a final product. This product must contain an enclosure to protect it from dust, humidity and other environmental interactions. In the event of outside use, this enclosure must be rated at least IP-65.
- Do not place Waspote in contact with metallic surfaces; they could cause short-circuits which will permanently damage it.

Further information you may need can be found at <http://www.libelium.com/development/waspote>

The “General Conditions of Libelium Sale and Use” document can be found at:  
[http://www.libelium.com/development/waspote/technical\\_service](http://www.libelium.com/development/waspote/technical_service)

## 2. Waspote Plug & Sense!

The new Waspote Plug & Sense! line allows you to easily deploy wireless sensor networks in an easy and scalable way ensuring minimum maintenance costs. The new platform consists of a robust waterproof enclosure with specific external sockets to connect the sensors, the solar panel, the antenna and even the USB cable in order to reprogram the node. It has been specially designed to be scalable, easy to deploy and maintain.

**Note:** For a complete reference guide download the “Waspote Plug & Sense! Technical Guide” in the **Development section** of the **Libelium website**.

### 2.1. Features

- Robust waterproof IP65 enclosure
- Add or change a sensor probe in seconds
- Solar powered with internal and external panel options
- Radios available: ZigBee, 802.15.4, WiFi, 868MHz, 900MHz, LoRa, 3G/GPRS and Bluetooth Low Energy
- Over the air programming (OTAP) of multiple nodes at once
- Special holders and brackets ready for installation in street lights and building fronts
- Graphical and intuitive programming interface
- External, contactless reset with magnet
- External SIM connector for GPRS or 3G models

### 2.2. Sensor Probes

Sensor probes can be easily attached by just screwing them into the bottom sockets. This allows you to add new sensing capabilities to existing networks just in minutes. In the same way, sensor probes may be easily replaced in order to ensure the lowest maintenance cost of the sensor network.



Figure : Connecting a sensor probe to Waspote Plug & Sense!

## 2.3. Solar Powered

Battery can be recharged using the internal or external solar panel options.

The external solar panel is mounted on a 45° holder which ensures the maximum performance of each outdoor installation.



*Figure : Waspote Plug & Sense! powered by an external solar panel*

For the internal option, the solar panel is embedded on the front of the enclosure, perfect for use where space is a major challenge.



*Figure : Internal solar panel*



Figure : Waspote Plug & Sense! powered by an internal solar panel

## 2.4. Programming the Nodes

Waspote Plug & Sense! can be reprogrammed in two ways:

The basic programming is done from the USB port. Just connect the USB to the specific external socket and then to the computer to upload the new firmware.

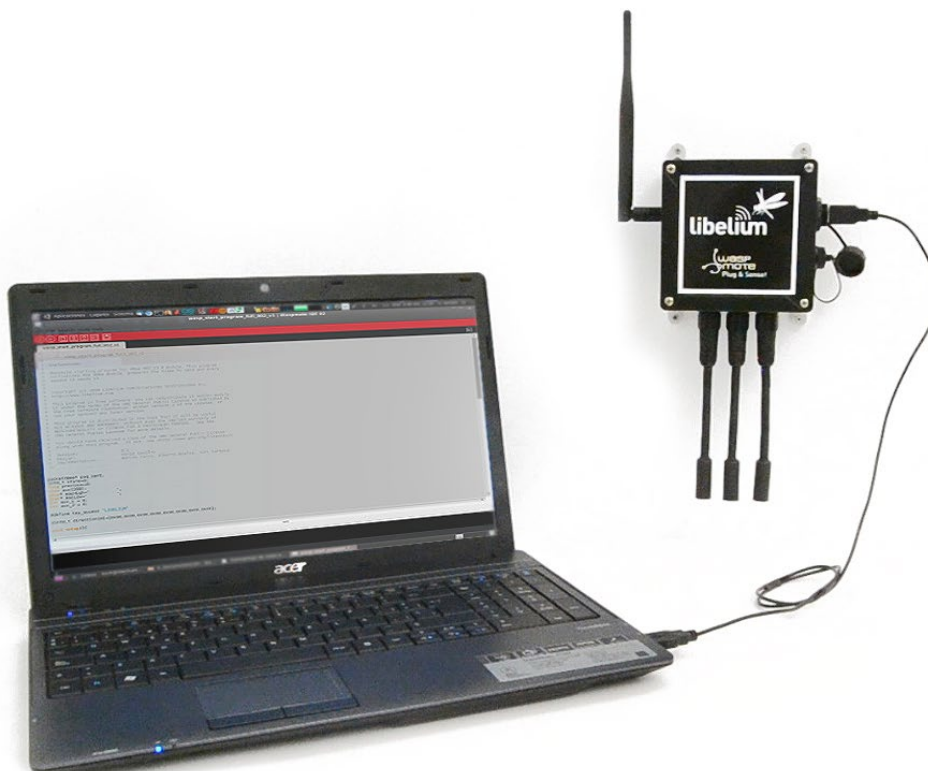


Figure : Programming a node



Over the Air Programming is also possible once the node has been installed. With this technique you can reprogram wirelessly one or more Waspote sensor nodes at the same time by using a laptop and the Waspote Gateway.

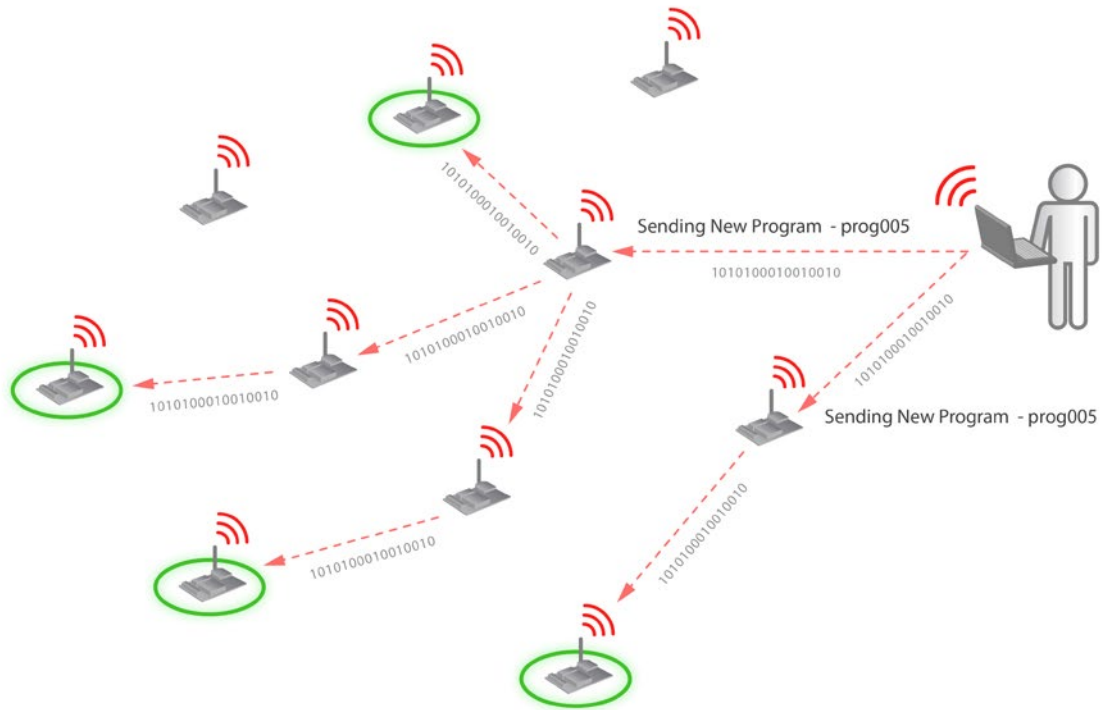


Figure : Typical OTAP process

## 2.5. Radio Interfaces

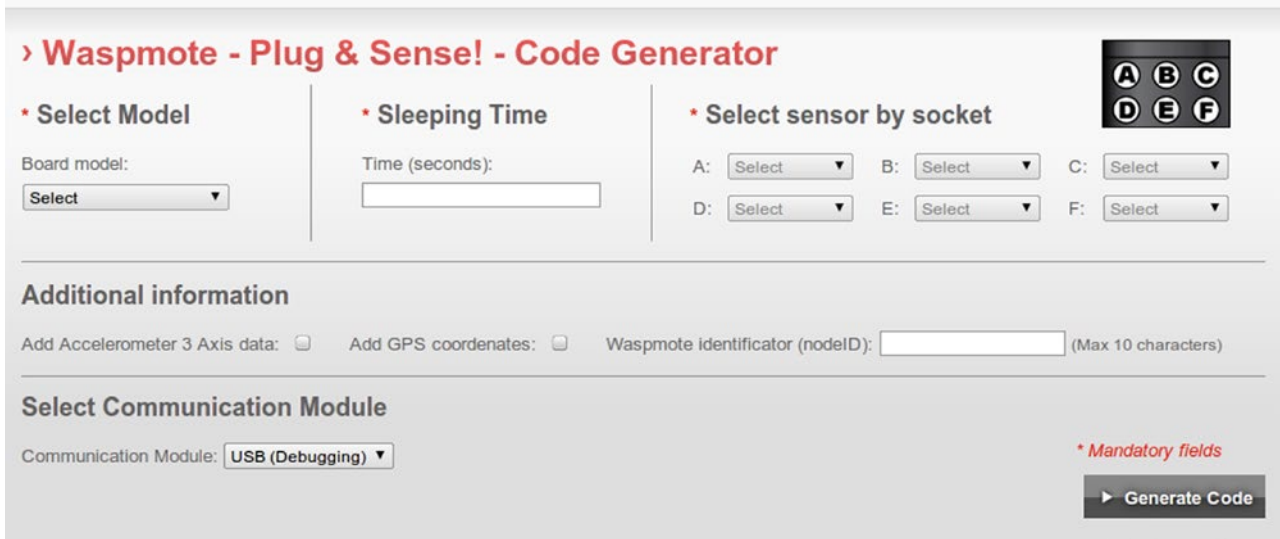
Model	Protocol	Frequency	txPower	Sensitivity	Range *
XBee-802.15.4-Pro	802.15.4	2.4GHz	100mW	-100dBm	7000m
XBee-ZB-Pro	ZigBee-Pro	2.4GHz	50mW	-102dBm	7000m
XBee-868	RF	868MHz	315mW	-112dBm	12km
XBee-900	RF	900MHz	50mW	-100dBm	10Km
LoRa	RF	868 and 900MHz	14dBm	-137dBm	22Km
WiFi	802.11b/g	2.4GHz	0dBm - 12dBm	-83dBm	50m-500m
GPRS Pro and GPRS+GPS	-	850MHz/900MHz/ 1800MHz/1900MHz	2W(Class4) 850MHz/900MHz, 1W(Class1) 1800MHz/1900MHz	-109dBm	- Km - Typical carrier range
3G/GPRS	-	Tri-Band UMTS 2100/1900/900MHz Quad-Band GSM/EDGE, 850/900/1800/1900 MHz	UMTS 900/1900/2100 0,25W GSM 850MHz/900MHz 2W DCS1800MHz/PCS1900MHz 1W	-106dBm	- Km - Typical carrier range
Bluetooth Low Energy	Bluetooth v.4.0 / Bluetooth Smart	2.4GHz	3dBm	-103dBm	100m

\* Line of sight, Fresnel zone clearance and 5dBi dipole antenna.

## 2.6. Program in minutes

In order to program the nodes an intuitive graphic interface has been developed. Developers just need to fill a web form in order to obtain the complete source code for the sensor nodes. This means the complete program for an specific application can be generated just in minutes. Check the Code Generator to see how easy it is at:

[http://www.libelium.com/development/plug\\_&\\_sense/sdk\\_and\\_applications/code\\_generator](http://www.libelium.com/development/plug_&_sense/sdk_and_applications/code_generator)



The screenshot shows a web form titled "Wasp mote - Plug & Sense! - Code Generator". It is divided into several sections:

- Select Model:** A dropdown menu labeled "Board model:" with a "Select" button.
- Sleeping Time:** A text input field labeled "Time (seconds):".
- Select sensor by socket:** Six dropdown menus labeled A, B, C, D, E, and F, each with a "Select" button. To the right of these is a small keypad with buttons A, B, C, D, E, and F.
- Additional information:** Three checkboxes: "Add Accelerometer 3 Axis data:", "Add GPS coordinates:", and "Wasp mote identifier (nodeID):" followed by a text input field (Max 10 characters).
- Select Communication Module:** A dropdown menu labeled "Communication Module:" with "USB (Debugging)" selected.
- Generate Code:** A button labeled "Generate Code" with a right-pointing arrow.

A red asterisk and the text "\* Mandatory fields" are located to the right of the "Communication Module" dropdown.

Figure : Code Generator

## 2.7. Data to the Cloud

The Sensor data gathered by the Wasp mote Plug & Sense! nodes is sent to the Cloud by **Meshlium**, the Gateway router specially designed to connect Wasp mote sensor networks to the Internet via Ethernet, WiFi and 3G interfaces.

Thanks to Meshlium's new feature, the Sensor Parser, now it is easier to receive any frame, parse it and store the data into a local or external Data Base.



Figure : Meshlium

## 2.8. Meshlium Storage Options

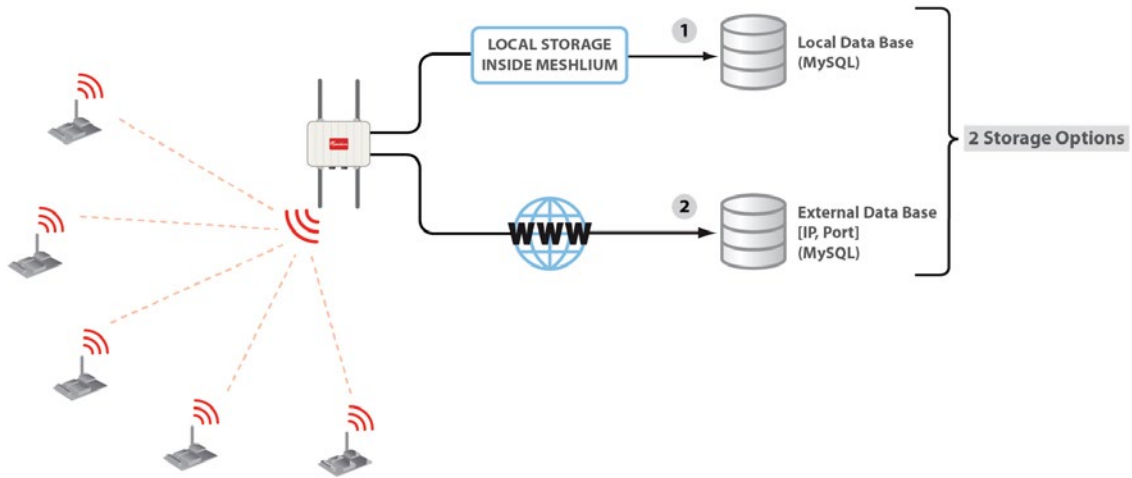


Figure : Meshlium Storage Options

- Local Data Base
- External Data Base

## 2.9. Meshlium Connection Options

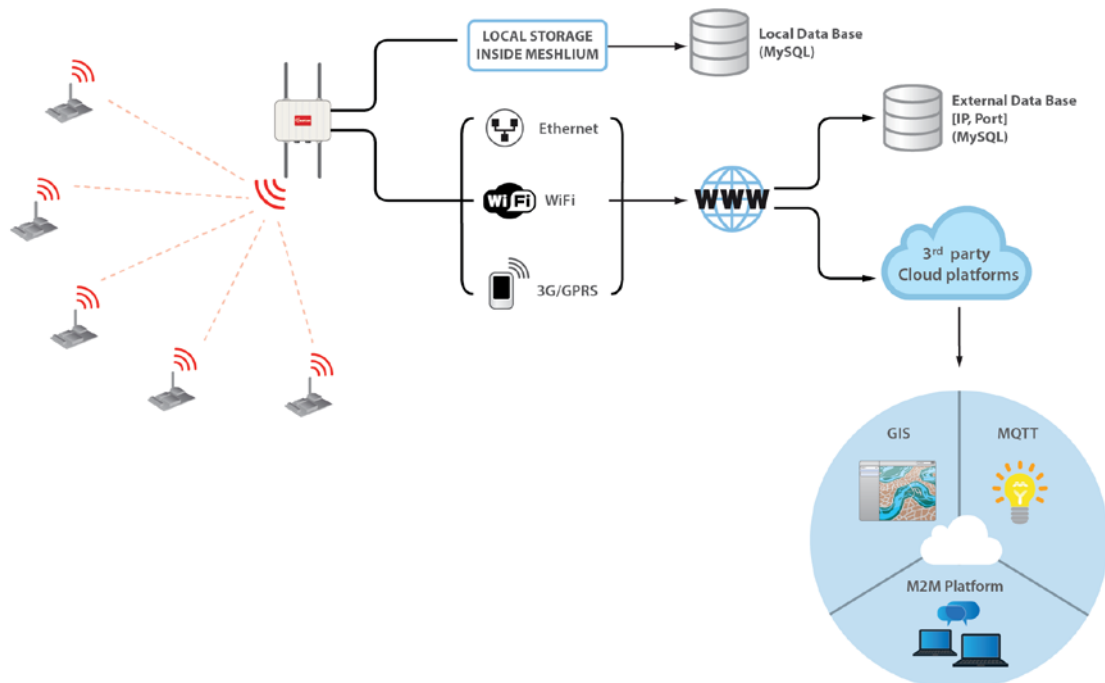


Figure : Meshlium Connection Options

- XBee / LoRa / GPRS / 3G / WiFi → Ethernet
- XBee / LoRa / GPRS / 3G / WiFi → WiFi
- XBee / LoRa / GPRS / 3G / WiFi → 3G/GPRS

## 2.10. Models

There are some defined configurations of Waspote Plug & Sense! depending on which sensors are going to be used. Waspote Plug & Sense! configurations allow to connect up to six sensor probes at the same time.

Each model takes a different conditioning circuit to enable the sensor integration. For this reason each model allows to connect just its specific sensors.

This section describes each model configuration in detail, showing the sensors which can be used in each case and how to connect them to Waspote. In many cases, the sensor sockets accept the connection of more than one sensor probe. See the compatibility table for each model configuration to choose the best probe combination for the application.

It is very important to remark that each socket is designed only for one specific sensor, so **they are not interchangeable**. Always be sure you connected probes in the right socket, otherwise they can be damaged.

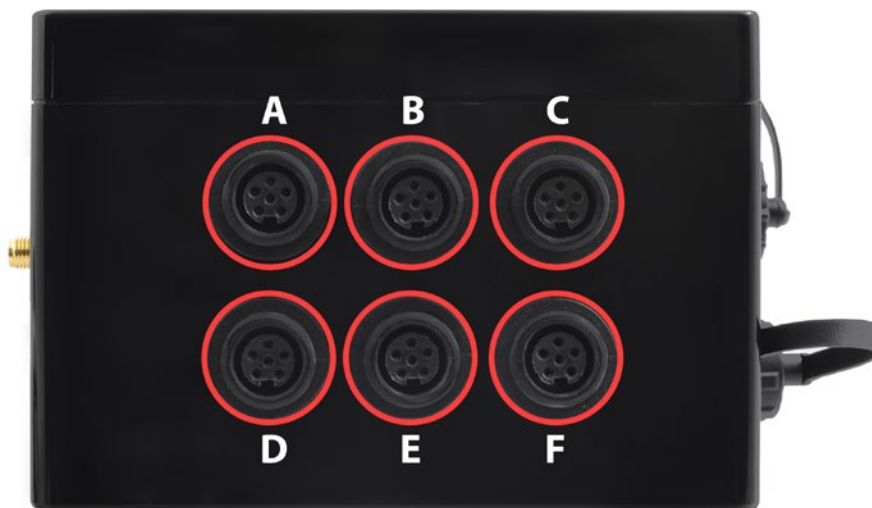


Figure : Identification of sensor sockets

## 2.10.1. Smart Environment

Smart Environment model is designed to monitor environmental parameters such as temperature, humidity, atmospheric pressure and some types of gases. The main applications for this Waspote Plug & Sense! configuration are city pollution measurement, emissions from farms and hatcheries, control of chemical and industrial processes, forest fires, etc. Go to the application section in the [Libelium website](#) for a complete list of services.



Figure : Smart Environment Waspote Plug & Sense! model

Sensor sockets are configured as shown in the figure below.

Sensor Socket	Sensor probes allowed for each sensor socket	
	Parameter	Reference
A	Temperature	9203
	Carbon monoxide - CO	9229
	Methane - CH <sub>4</sub>	9232
	Ammonia – NH <sub>3</sub>	9233
	Liquefied Petroleum Gases: H <sub>2</sub> , CH <sub>4</sub> , ethanol, isobutene	9234
	Air pollutants 1: C <sub>4</sub> H <sub>10</sub> , CH <sub>3</sub> CH <sub>2</sub> OH, H <sub>2</sub> , CO, CH <sub>4</sub>	9235
	Air pollutants 2: C <sub>6</sub> H <sub>5</sub> CH <sub>3</sub> , H <sub>2</sub> S, CH <sub>3</sub> CH <sub>2</sub> OH, NH <sub>3</sub> , H <sub>2</sub>	9236
	Alcohol derivates: CH <sub>3</sub> CH <sub>2</sub> OH, H <sub>2</sub> , C <sub>4</sub> H <sub>10</sub> , CO, CH <sub>4</sub>	9237
B	Humidity	9204
	Atmospheric pressure	9250
C	Carbon dioxide - CO <sub>2</sub>	9230
D	Nitrogen dioxide - NO <sub>2</sub>	9238 and 9238-B
E	Ozone - O <sub>3</sub>	9258 and 9258-B
	Hydrocarbons - VOC	9201 and 9201-B
	Oxygen - O <sub>2</sub>	9231
F	Carbon monoxide - CO	9229
	Methane - CH <sub>4</sub>	9232
	Ammonia – NH <sub>3</sub>	9233
	Liquefied Petroleum Gases: H <sub>2</sub> , CH <sub>4</sub> , ethanol, isobutene	9234
	Air pollutants 1: C <sub>4</sub> H <sub>10</sub> , CH <sub>3</sub> CH <sub>2</sub> OH, H <sub>2</sub> , CO, CH <sub>4</sub>	9235
	Air pollutants 2: C <sub>6</sub> H <sub>5</sub> CH <sub>3</sub> , H <sub>2</sub> S, CH <sub>3</sub> CH <sub>2</sub> OH, NH <sub>3</sub> , H <sub>2</sub>	9236
	Alcohol derivates: CH <sub>3</sub> CH <sub>2</sub> OH, H <sub>2</sub> , C <sub>4</sub> H <sub>10</sub> , CO, CH <sub>4</sub>	9237

Figure : Sensor sockets configuration for Smart Environment model

**Note:** For more technical information about each sensor probe go to the **Development section** in Libelium website.

### 3. Hardware

#### 3.1. General Description

The Waspmote Gases 2.0 board has been designed to monitor environmental parameters such as temperature, humidity, atmospheric pressure and 14 different types of gases. It allows the inclusion of 7 gases sensors at the same time, the regulation of their power through a system of solid state switches and the amplification of the output signal of each one of them through a non-inverting amplification stage of a maximum gain of 101 controlled by a digital potentiometer configurable through the Inter-Integrated Circuit Bus, I2C).

The gases which can be monitored are:

- Carbon Monoxide – CO
- Carbon Dioxide – CO<sub>2</sub>
- Molecular Oxygen – O<sub>2</sub>
- Methane – CH<sub>4</sub>
- Molecular Hydrogen – H<sub>2</sub>
- Ammonia – NH<sub>3</sub>
- Isobutane – C<sub>4</sub>H<sub>10</sub>
- Ethanol – CH<sub>3</sub>CH<sub>2</sub>OH
- Toluene – C<sub>6</sub>H<sub>5</sub>CH<sub>3</sub>
- Hydrogen Sulphide – H<sub>2</sub>S
- Nitrogen Dioxide – NO<sub>2</sub>
- Ozone – O<sub>3</sub>
- Volatile Organic Compounds (VOC's)
- Hydrocarbons

#### 3.2. Specifications

**Weight:** 20gr

**Dimensions:** 73.5 x 51 x 1.3 mm

**Temperature Range:** [-20°C, 65°C]

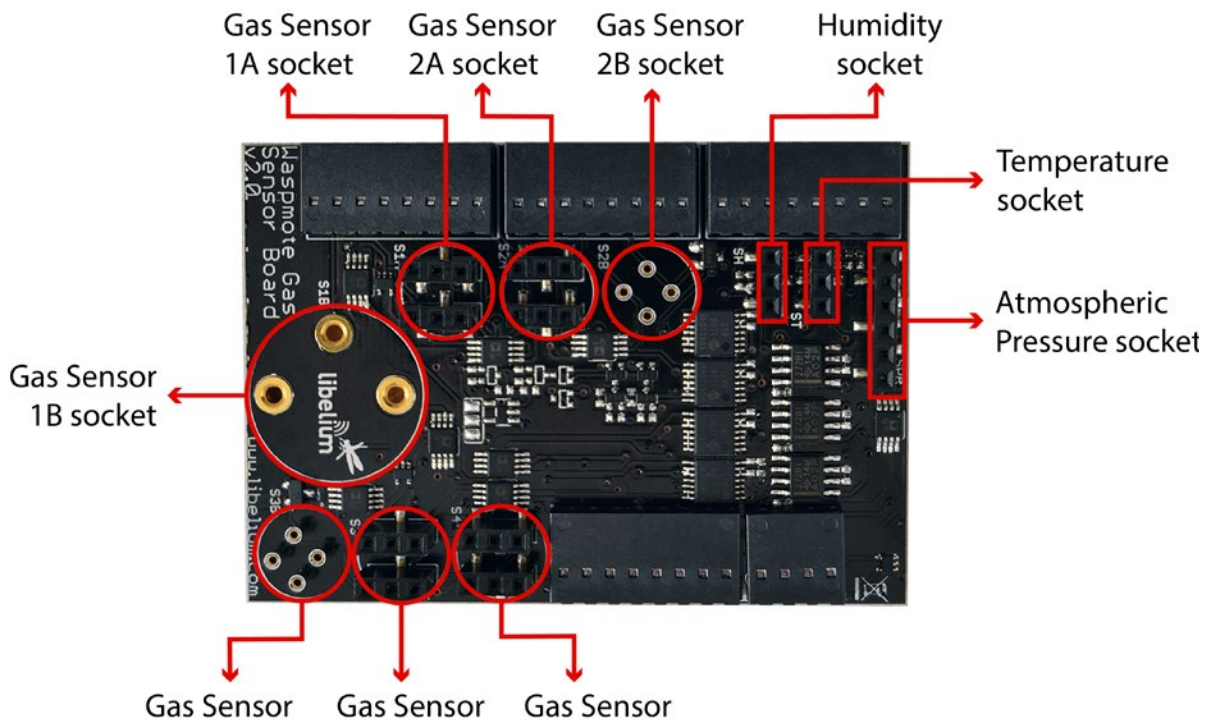


Figure: Upper side

Sensors compatibility:

- There is a dedicated socket for each of the following sensors:
  - temperature
  - humidity
  - air pressure
  - CO<sub>2</sub> (socket 1A)
  - O<sub>2</sub> (socket 1B)
  - NO<sub>2</sub> (socket 3B)
  - O<sub>3</sub> or VOC (socket 2B)
- Sockets 2A, 3 and 4 are available for the rest of the sensors.
- CO and NH<sub>3</sub> sensors can only be placed in the sockets 3 or 4.

### 3.3. Electrical Characteristics

**Board power voltage:** 3.3V and 5V

**Sensor power voltage:** 5V

**Maximum admitted current (continuous):** 200mA

**Maximum admitted current (peak):** 400mA



## 4. Sensors

### 4.1. General considerations in the use of the sensors

A very similar structure has been installed for all the connectors for the gas sensors: a load resistance at the output of each sensor, except in connectors 1A and 1B where it is not necessary, combined with an amplification stage of maximum gain 101. Connectors 1A and 1B, 2A and 2B and 3A and 3B share the same amplification stage and the same output to the microprocessor. For more details on the connectors, see the Design and connections section in this manual.

The choice of amplification stage gain and of the sensor's load resistance can be carried out according to two parameters: the specific sensor available, since there may be significant variations between two different sensors of the same model, and the value and range of concentrations of gas to be monitored.

**Important:** when selecting load resistance and amplification it must be remembered that, although the sensors must be powered by a voltage of 5V to function appropriately, the Waspote allows input between 0 and 3.3V, so it will be necessary to calculate the resistance, load and gain values to adapt the measurement range of the sensor to the Waspote input.

The amplification stage gain and load resistance of a connector can be configured through a simple group of commands available in the SensorGasv20 library, created to facilitate handling of the board from the Waspote mote. For more information on the library instructions and steps to follow for the configuration of the sensors, consult the API section of this manual.

The accuracy which can be obtained in the sensor's output value will be dependent on the way in which it is supplied. This way, the longer the power time or duty cycle, as appropriate, the better accuracy will be obtained. The disadvantage of prolonged power is an increase in the mote's consumption, with the consequent decrease of the battery's life, so adjusting the power of each sensor to the requirements of the specific application being developed is recommended to optimize the equipment's performance.

The calculation of the sensor's resistance, from which the concentration of gas value can be obtained using the graphs included in this manual and in the sensors' data sheets, can be made using the following equation:

$$R_s = (V_c * R_l) / V_{out} - R_l$$

In which  $R_s$  is the sensor's output resistance,  $V_c$  its power (5V for any sensor except socket 3B for the  $NO_2$ , which is powered at 1.8V, and socket 2B for VOC and  $O_3$  sensors, which is powered at 2.5V),  $V_{out}$  is the output voltage measurement and  $R_l$  the load resistance which has been defined.

When a sensor remains without power for a prolonged period it is possible that it shows an unstable output.

This stability is regained after spending time switched on or after many consecutive cycles of power supply.

Sensitivity of the sensor may vary when the device is subjected to large variations in temperature or humidity, for example in outdoor conditions. To compensate for these variations, use the tables and graphs used in the sensors' data sheets.

## 4.2. Starting with the gas sensors

In this section we are going to explain the first steps to start with the sensors used in the Gases 2.0 Board, including the most common doubts, such as what parameters configure for the adaptation stage, which calibration process must be followed or how to transform the read voltage or resistance value into a concentration.

Take into account that developing a robust application for gases detection or measurement may take an important effort of testing and knowing the sensors.

### 4.2.1. Sensor calibration

**Note:** *The calibrated gas sensor line launched in 2014 was discontinued.*

When dealing with most of the sensors used in the Gases 2.0 board it is highly recommended to calibrate the sensor in order to get an accurate value, since, as explained in section “General considerations in the use of the sensors”, its normal resistance and sensitivity may vary from one unit to another in a wide range. This calibration may not be necessary in all applications, for example if the sensor is going to be used in a gas detection, where monitoring the variation of the sensor output may be enough to have it working properly, and the normal operation conditions may be replicated without an specific equipment.

The calibration procedure requires the capture of the sensor response under different concentrations of gas in the target operation range (that should be comprised in the operation range of the sensor), and, depending on the conditions of the application to be implemented, under controlled temperature and humidity. The larger the number of calibration points in that range the more accurate the calibration will be, a logarithmic approximation to the response of the sensor for the intermediate values should be applied (in one or two axis, depending on the sensors, as shown in their respective response graphs). Take into account that gas sensors are sensitive, in different degree, to several target gases, which may affect the measurement both in the field and in the laboratory. The recreation of these conditions may require of specific equipment, so it may be necessary the help of a specialized laboratory.

### 4.2.2. Gain and load resistor configuration

The gain and load resistor configured for a given stage will depend on two main aspects: first of all, the sensor connected to the stage. Since in most cases there is a high variability in the parameters that rule the behavior of the sensor (sensor's initial resistance and sensitivity, taking this as the variability of the sensor resistance with the gas concentration), every single unit may show important differences in operation, so it will be necessary a specific configuration for each one. Secondly, the application conditions under which the sensor is supposed to be operating will also have to be taken into account, in order to maintain the output voltage of the adaptation stage in the voltage range of the input of the mote's microcontroller (between 0 and 3.3V). It will also be recommended to set both parameters to have an output voltage of around 1.6V (the middle of the voltage input range) when the gas concentration is in the middle of the desired operation range.

The most adequate way of configuring the adaptation stage is by determining the load resistance and gain necessary from the calibration data and the application requirements and test it, adjusting the definitive values with the results obtained. In other cases, in which these parameters cannot be that clearly defined owing to the lack of calibration or to the indeterminacy of the application conditions, for example when handling the sensors for the first time, you can follow the advice provided below:

As a general rule, gain will be fixed at 1 in almost every application, only in very specific situations, such as operation in the limits of the sensor range, it will be necessary a different value. There are two exceptions to this rule, that are sensors for CO<sub>2</sub> (TGS4161) and O<sub>2</sub> (SK-25) which output a voltage that has to be amplified to adjust it to the Wasp mote's analog-to-digital converter. In the case of the O<sub>2</sub> sensor the extremely low voltage advises a gain of 100, whilst for the CO<sub>2</sub> sensor, with a higher (and more variable) output, a gain between 7 and 10 will be adequate.

In the case of the load resistor it is not that simple, since, as said before, it will be different from one sensor to another. If calibration has not been performed it will be necessary a wide testing based on trial and error to achieve the correct resistance value. It is recommended in this case to start with the lowest value indicated as initial sensor resistance in the documentation, and adjust it to get the desired output voltage (usually 1.6V, the middle of the analog-to-digital converter range, if it is not specified by another restriction). When carrying out this adjustment beware not to configure a resistance below the minimum load resistance indicated in the corresponding section of the documentation, since it could damage the sensor.

As indicated in section “General considerations in the use of the sensors”, beware when operating with sensors that have not been used for a long time, since they may require of some time to get to work properly. Applying a “burning” process, having them on with a continuous reading for at least about 12 hours, is highly recommended before starting using the sensors.

### 4.2.3. Converting the read data

The gas concentration value in ppm or ppb, depending on the sensor, can be obtained from the ratio between the read resistance of the sensor (that can be obtained using function `calculateResistance`, described in section “API”, or by direct calculation) and the initial resistance of the sensor, except in the case of the O<sub>2</sub> (SK-25) and CO<sub>2</sub> (TGS4161), which as said before output a voltage proportional to the gas concentration.

Again, the most accurate response will be obtained if sensors are calibrated previously to their introduction in the definitive application. Constructing a conversion function or graph out of the calibration results and translating the read values into it will lead to the best performance.

In case the calibration process cannot be performed, the concentration may be estimated using the provided output graphs, which correspond to the typical response of the sensors. Even though, it will be necessary to calculate the initial resistance of the sensor, whose range is given in the specifications section of the sensor. In some cases it may be determined from a normal environment measurement, since information may be obtained from other sources such as detectors or studies about the gas presence in the operation environment. Anyway, take into account that the data obtained following this method will be highly inaccurate.

Finally, it is necessary to keep in mind that in the case of some sensors and applications the extraction of the concentration value may not be that straightforward, since other parameters may interfere in the behavior of the sensor. In some sensors other gases different from the target can cause a significant change in the resistance value, so it will be necessary a high knowledge of the application conditions to be able to determine precisely the gas concentration. The combination of several sensors is a suitable option to avoid this situations. Also, in applications in which high variations of temperature and humidity are present it may be required to perform a compensation. For this, a temperature and a humidity sensor have been integrated in the Gases Sensor boards, so all the necessary information can be obtained. Please refer to the sensors data sheet and application notes for more information about the temperature and humidity compensation of the sensors.

Instead of using this function it can be applied a more simple linear approximation between each of the points. This method will generally be more accurate when the output is very close to a calibration point since the curve resulting from the fitting may not be adjusted to pass by that exact point, but it will lead to more inaccurate results for points in the middle of the range.

### 4.2.4. Calibrated sensors performance

In optimal working conditions the accuracy of the measurement has been calculated to be of the  $\pm 5\%$  of the measurement, including the error owed to the electronics and the calibration. However, when operating with the calibrated sensors there are several additional issues regarding the accuracy of the measurement that will have to be taken into account when developing a new application:

First of all, as mentioned before, the accuracy will be highly dependent on the heating time applied to the sensor. For proper operation sensors must be powered until they reach their operating temperature, which may take several minutes depending on the sensor, the environmental temperature and the duty cycle of the application. If a completely stable measurement has not been achieved an additional error to that of the sensor and the electronics itself will be added.

Secondly, the environmental temperature will also affect the behavior of the sensor, being necessary a temperature correction to achieve the maximum accuracy. This correction should be performed from the temperature dependence graphs and tables provided by the manufacturer of the sensor in its data sheet or application note.

The last problem encountered with these sensors (which generally affects all chemical sensors) is recalibration. **After some operation time, the wear of the sensing layer owing to different effects will lower the accuracy of the sensor, so it will have to be replaced with a new calibrated probe.** The more accuracy required or the more harass environment sensors will operate in, the more frequent replacement will be required, but **as a general rule probes should be always replaced every six months.**

## 4.3. Humidity Sensor – 808H5V5

### 4.3.1. Specifications

**Measurement range:** 0 ~ 100%RH

**Output signal:** 0.8 ~ 3.9V (25°C)

**Accuracy:**  $<\pm 4\%$  RH (at 25°C, range 30 ~ 80%),  $<\pm 6\%$  RH (range 0 ~ 100)

**Supply voltage:** 5V DC  $\pm 5\%$

**Operating temperature:** -40 ~ +85°C

**Response time:** <15 seconds

**Typical consumption:** 0.38mA

**Maximum consumption:** 0.5mA



Figure : Image of the 808H5V5 sensor

### 4.3.2. Measurement Process

This is an analog sensor which provides a voltage output proportional to the relative humidity in the atmosphere. As the sensor's signal range is outside of that permitted to the Wasp mote's input, a voltage divider has been installed which converts the output voltage to values between 0.48 ~ 2.34V.

The sensor remains powered provided that the board's 5V supply is switched on, so that for its reading it is only necessary to execute the capture command of the analog value of the pin to which the sensor is connected (ANALOG4).

Reading code:

```
{
  SensorGasv20.0N();
  delay(15000);
  float value;
  value = SensorGasv20.readValue(SENS_HUMIDITY);
}
```

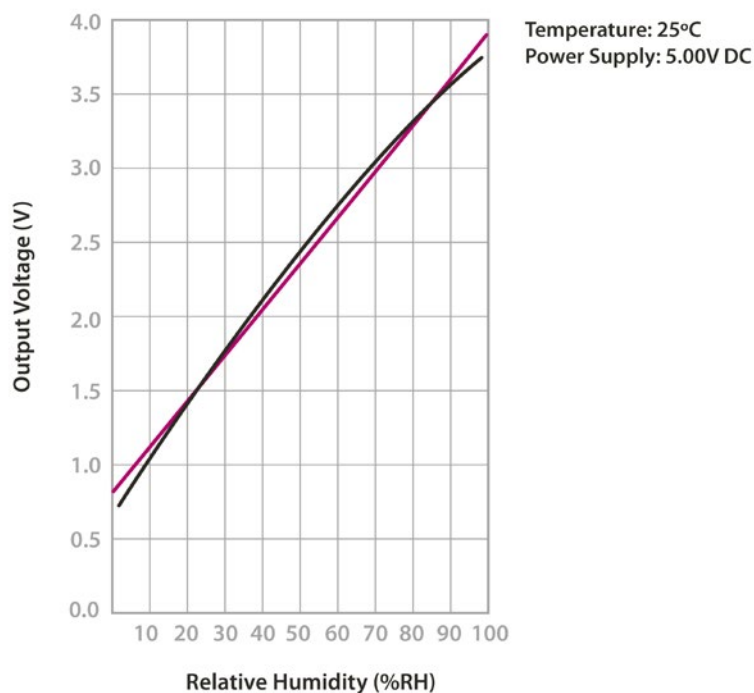


Figure : 808H5V5 Humidity sensor output taken from the Sencera Co. Ltd sensor data sheet

You can find a complete example code for reading the humidity sensor in the following link:

<http://www.libelium.com/development/waspmote/examples/ga-2-humidity-sensor-reading>

This sensor has its own connector on the Waspote Gases 2.0 board:

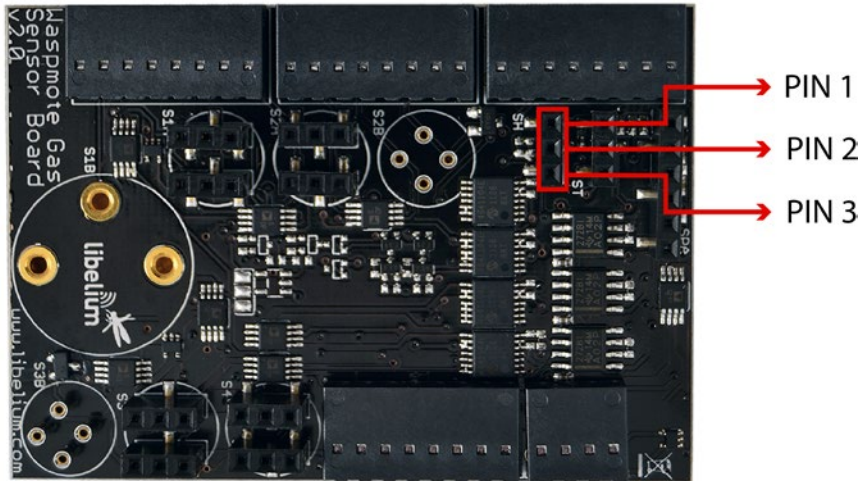


Figure : Image of the sensor connector on the Waspote Gases 2.0 board

## 4.4. Temperature Sensor – MCP9700A

### 4.4.1. Specifications

**Measurement range:** [-40°C, +125°C]

**Output voltage (0°C):** 500mV

**Sensitivity:** 10mV/°C

**Accuracy:** ±2°C (range 0°C ~ +70°C), ±4°C (range -40 ~ +125°C)

**Supply voltage:** 2.3 ~ 5.5V

**Response time:** 1.65 seconds (63% response from +30 to +125°C).

**Typical consumption:** 6µA

**Maximum consumption:** 12µA



Figure : Image of the MCP9700A temperature sensor

### 4.4.2. Measurement Process

The MCP9700A is an analog sensor which converts a temperature value into a proportional analog voltage. The range of output voltages is between 100mV (-40°) and 1.75V (125°C), resulting in a variation of 10mV/°C, with 500mV of output for 0°C.

The output can thus be read directly from Waspote through the capture command of the pin's analog value to which it is connected (ANALOG1).

Reading code:

```
{
  SensorGasv20.ON();
  delay(100);
  float value;
  value = SensorGasv20.readValue(SENS_TEMPERATURE);
}
```

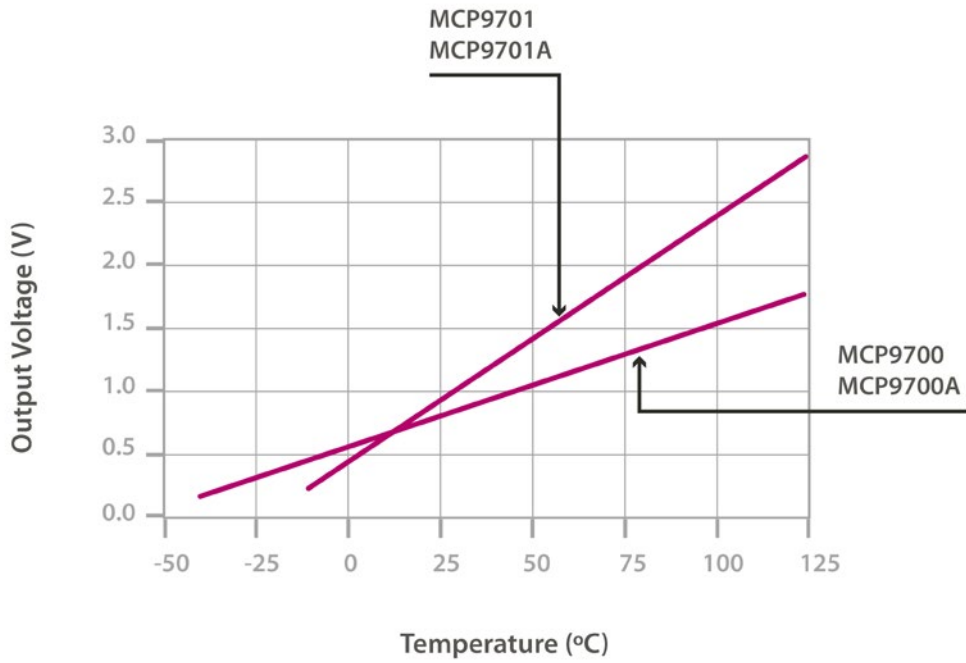


Figure : Graph of the MCP9700A sensor output voltage with respect to temperature, taken from the Microchip sensor's data sheet

This sensor has its own connector on the Wasp mote Gases 2.0 board:

You can find a complete example code for reading the temperature sensor in the following link:

<http://www.libelium.com/development/waspote/examples/ga-1-temperature-sensor-reading>

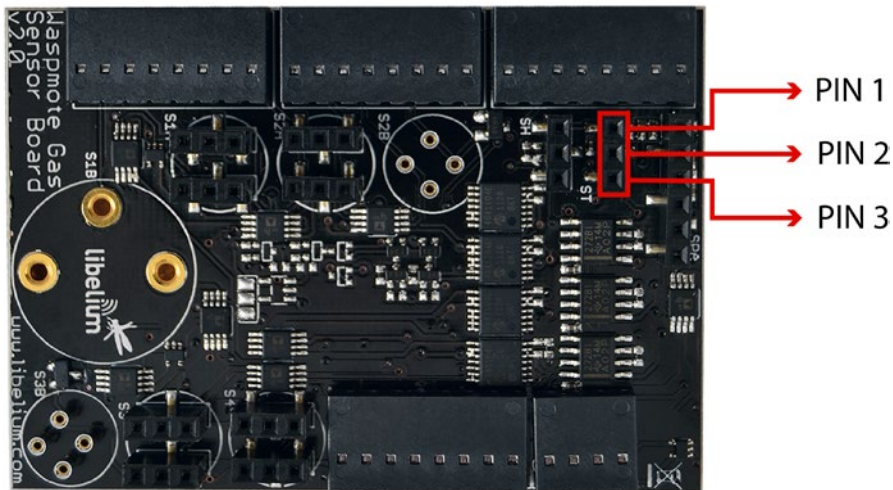


Figure : Image of the temperature sensor connector of the Wasp mote Gases 2.0 board

## 4.5. Atmospheric Pressure Sensor - MPX4115A

### 4.5.1. Specifications

**Measurement range:** 15 ~ 115kPa

**Output signal:** 0,2 ~ 4,8V (0 ~ 85°C)

**Sensitivity:** 46mV/kPa

**Accuracy:** <math>\pm 1,5\%V</math> (0 ~ 85°C)

**Typical consumption:** 7mA

**Maximum consumption:** 10mA

**Supply voltage:** 4.85 ~ 5.35V

**Operation temperature:** -40 ~ +125°C

**Storage temperature:** -40 ~ +125°C

**Response time:** 20ms



Figure : MPX4115A Sensor

### 4.5.2. Measurement Process

The MPX4115A sensor converts atmospheric pressure to an analog voltage value in a range covering between 0.2V and 4.8V. As this is a range which exceeds the maximum value admitted by Wasp mote, its output has been adapted to fit in a range between 0.12V and 2.88V.

To read the sensor it is sufficient to capture the analog value in its input (ANALOG5) via the corresponding command.

Reading Code:

```
{
  SensorGasv20.ON();
  SensorGasv20.setSensorMode(SENS_ON, SENS_PRESSURE);
  delay(30);
  float value;
  value = SensorGasv20.readValue(SENS_PRESSURE);
}
```

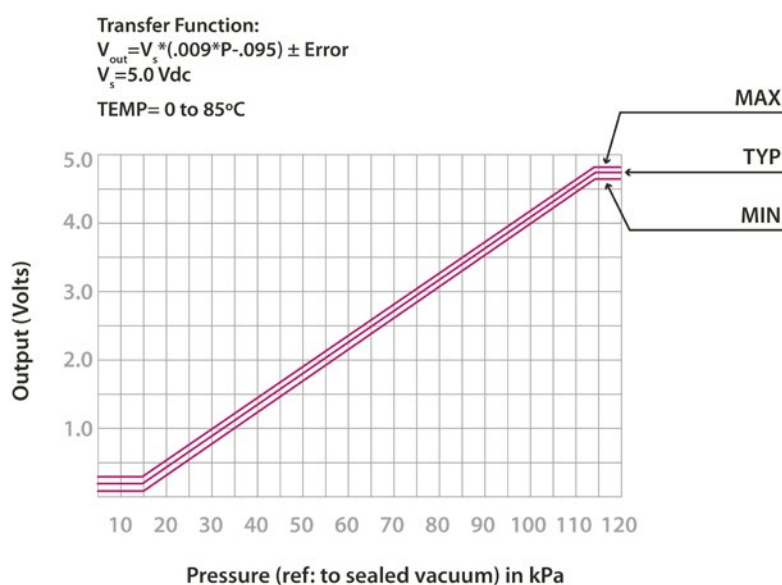


Figure : Graph of the MPX4115A sensor's output voltage with regard to pressure taken from the Freescale sensor's data sheet

You can find a complete example code for reading the atmospheric pressure sensor in the following link:

<http://www.libelium.com/development/waspote/examples/ga-3-atmospheric-pressure-sensor-reading>

This sensor has its own connector on the Wasp mote Gases 2.0 board:

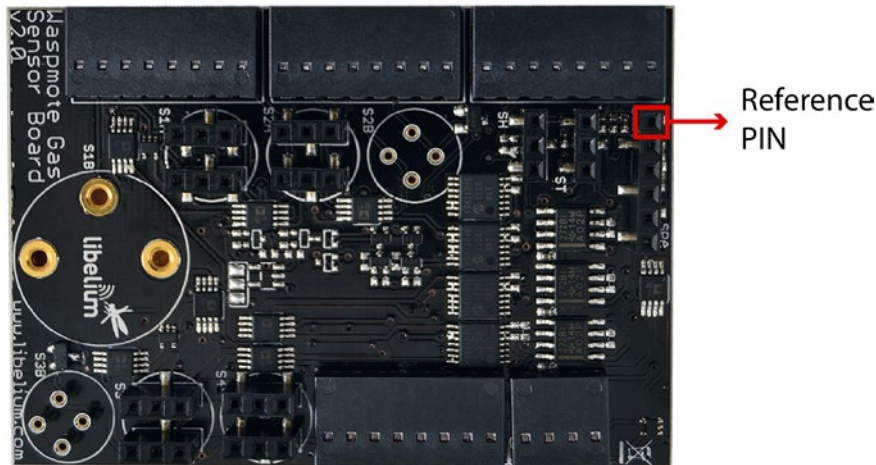


Figure: Image of the socket for the MPX4115A sensor

## 4.6. Carbon Monoxide (CO) Sensor – TGS2442

### 4.6.1. Specifications

**Gases:** CO

**Measurement range:** 30 ~ 1000ppm

**Resistance at 100ppm:** 13.3 ~ 133k $\Omega$

**Sensitivity:** 0.13 ~ 0.31 (ratio between the resistance at 300ppm and at 100ppm)

**Supply voltage:** 5V  $\pm$ 0.2V DC

**Operating temperature:** -10 ~ +50 $^{\circ}$ C

**Response time:** 1 second

**Minimum load resistance:** 10k $\Omega$

**Average consumption:** 3mA (throughout the complete power supply cycle in one second)



Figure: Image of the TGS2442 sensor

### 4.6.2. Measurement Process

The TGS2442 is a resistive sensor sensitive to the changes in concentration of Carbon Monoxide (CO) and, very slightly, Hydrogen ( $H_2$ ), which may be placed in the board's socket 3A and 4, and must be connected in the way indicated in images 51 and 53 of sections "Connector 3" and "Connector 4" about these two connectors. The sensor's resistance would vary according to the graph in figure below these paragraphs, which may present significant variations between two different sensors, so it is recommended to consult the sensor's documentation to choose the load resistance and amplification gain and calibrate it before finally inserting it into the application.

Reading this sensor requires a cycle of one second throughout which two power supply pulses are generated on heat resistance and sensor resistance of 14ms and 5ms each (average consumption throughout the power supply cycle is 3mA). The execution of this cycle and the reading of the sensor can be done automatically using the functions of the SensorGasv20 library.

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}
```



**SENSOR** indicates the socket into which it is to be inserted, for this sensor: **SENS\_SOCKET3CO** or **SENS\_SOCKET4CO**  
**GAIN** indicates the chosen gain for the sensor.  
**RESISTOR** indicates the load resistance to be introduced

You can find a complete example code for reading the TGS2442 sensor placed on socket 4 in the following link:  
<http://www.libelium.com/development/waspote/examples/ga-11-co-sensor-on-socket4-reading>

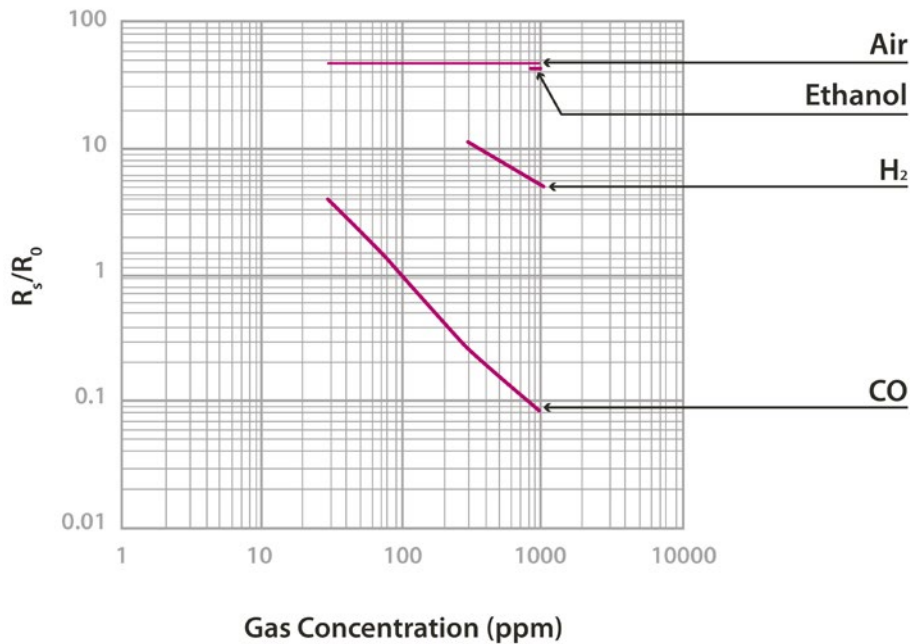


Figure : Graph of the sensitivity of the TGS2442 taken from the Figaro sensor's data sheet

## 4.7. Carbon Dioxide (CO<sub>2</sub>) Sensor – TGS4161

### 4.7.1. Specifications

- Gases:** CO<sub>2</sub>
- Measurement range:** 350 ~ 10000ppm
- Voltage at 350ppm:** 220 ~ 490mV
- Sensitivity:** 44 ~ 72mV (variation between the voltage at 350ppm and at 3500ppm)
- Supply voltage:** 5V ±0.2V DC
- Operating temperature:** -10 ~ +50°C
- Response time:** 1.5 minutes
- Average consumption:** 50mA



Figure : Image of the TGS4161 sensor

## 4.7.2. Measurement Process

The TGS4161 sensor provides a voltage output proportional to the CO<sub>2</sub> concentration in the atmosphere. It shows a value between 220 and 490mV for a concentration of 350ppm (approximately the normal CO<sub>2</sub> concentration in the air) decreasing as the amount of gas increases. Different sensors may show a large variability in the initial voltage values at 350ppm and sensitivity, so it is recommended to calibrate each sensor before including it in the application.

The accuracy that this sensor can offer will vary depending on the time it has remained powered before being measured. A time of 30 seconds is sufficient to detect significant changes in concentration, while a high accuracy measurement will require at least 10 minutes of power.

This sensor must be placed only in connector 1A as indicated in the figure which appears in section "Connector 1". To access the sensor's output value it is enough to execute the SensorGasv20 library command which captures the sensor's analog value in its input pin (ANALOG3).

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_CO2, GAIN);
  SensorGasv20.setSensorMode(SENS_ON, SENS_CO2);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_CO2);
}
```

**GAIN** indicates the chosen gain for the sensor.

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

You can find a complete example code for reading the TGS4161 sensor in the following link:

<http://www.libelium.com/development/waspmote/examples/ga-4-co2-sensor-reading>

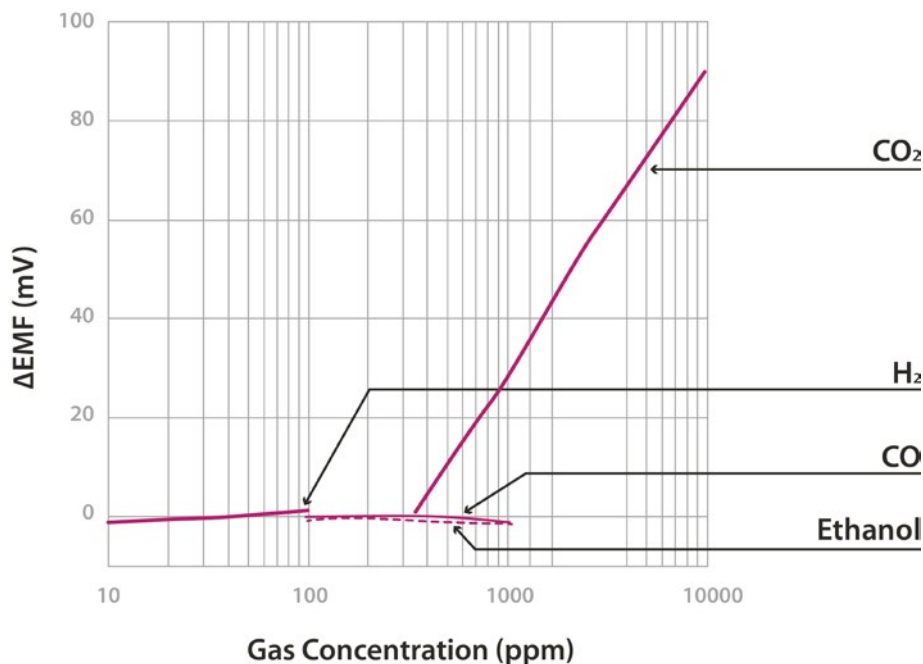


Figure : Graph of sensitivity of the TGS4161 sensor taken from the Figaro sensor's data sheet

## 4.8. Molecular Oxygen (O<sub>2</sub>) Sensor – SK-25

### 4.8.1. Specifications

**Gases:** O<sub>2</sub>

**Measurement range:** 0 ~ 30%

**Output range:** Approximately 0 ~ 10mV

**Initial voltage:** 5.5 ~ 8.8mV

**Operating temperature:** 5 ~ +40°C

**Response time:** 15 seconds

**Consumption:** 0μA



Figure : Image of the SK-25 sensor

### 4.8.2. Measurement Process

The SK-25 is an analog sensor which provides a voltage output proportional to the O<sub>2</sub> concentration in the atmosphere, without needing power and therefore with zero consumption. It shows an output range between 0 and 10mV, with voltage in standard conditions (approximately 21% O<sub>2</sub> concentration) of between 5.5 and 8.8mV. The output response can vary from one sensor to another, so it is recommended to calibrate the sensor before finally inserting it into the application.

Reading code:

```
{
  SensorGasv20.ON();
  delay(10);
  SensorGasv20.configureSensor(SENS_O2, GAIN);
  float value;
  value = SensorGasv20.readValue(SENS_O2);
}
```

**GAIN** indicates the chosen gain for the sensor.

You can find a complete example code for reading the SK25 sensor in the following link:

<http://www.libelium.com/development/waspmote/examples/ga-5-o2-sensor-reading>

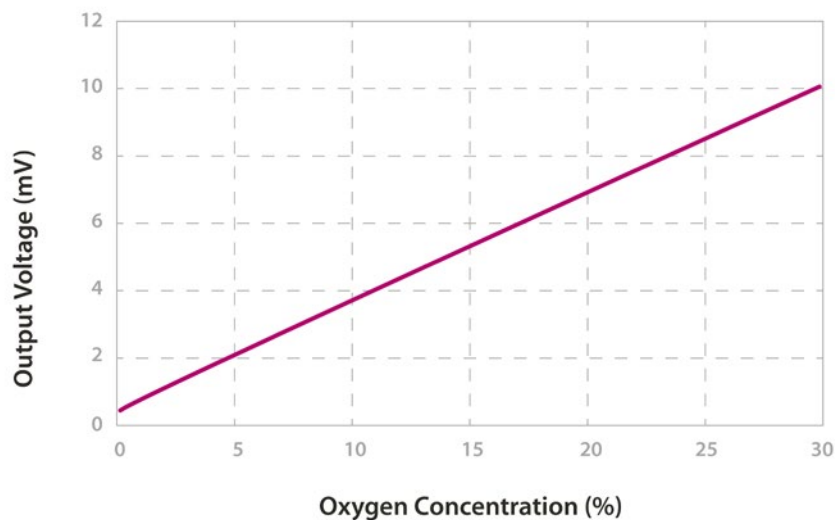


Figure : Graph of the sensitivity of the SK-25 extracted from the Figaro sensor's data sheet

The calibration process for this sensor can be performed by the customer very easily, if we consider a couple of facts:

- In normal conditions,  $O_2$  is present in the atmosphere with a concentration of 20.5%.
- The SK-25 sensor has a linear output:  $y = f(x) = a \cdot x + b$ , where:
  - $y$  has dimensions of  $O_2$  percentage,
  - $x$  has dimensions of volts (it is the voltage read by the function `SensorGasv20.readValue(SENS_02)`),
  - and  $b$  is negligible.

So we get a simple system:  $y = a \cdot x$ . If we want to calibrate the sensor response we just need to obtain the  $a$  term. With a single sample outdoors and a division, the customer can get the  $a$  term in order to output accurate measurements. Make sure the sensor's output is stable in time before getting the sample (a few minutes may be necessary).

## 4.9. Nitrogen Dioxide ( $NO_2$ ) Sensor - MiCS-2710

### 4.9.1. Specifications

**Gases:**  $NO_2$

**Measurement range:** 0.05 ~ 5ppm

**Air resistance:** 0.8 ~ 8k $\Omega$  (typically 2.2k $\Omega$ )

**Sensitivity:** 6 ~ 100 (typically 55, ratio between the resistance at 0.25ppm and in air)

**Supply voltage:** 1.7 ~ 2.5V DC

**Operating temperature:** -30 ~ +85 $^{\circ}C$

**Response time:** 30 seconds

**Average consumption:** 26mA (throughout the complete power supply cycle in one second)



Figure : Image of the MiCS-2710 sensor

### 4.9.2. Measurement Process

The MiCS-2710 is a sensor whose resistance varies in the presence of small concentrations of  $NO_2$ . This value varies between 2k $\Omega$  and 2M $\Omega$  approximately, providing high accuracy throughout the output range. Unlike the rest of the board's gas sensors, which operate at a voltage of 5V, this sensor is powered through a 1.8V voltage regulator, with consumption of approximately 26mA. The sensor's resistance in air, as well as its sensitivity, can vary between different units, so it is recommended to calibrate each one of them before finally inserting them in the application.

This sensor must be connected in socket 3B of the board (its position is indicated in section "Connector 3"), and its output can be read through the capture commands of the SensorGasv20 library.

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_SOCKET3B, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET3B);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_SOCKET3B);
}
```

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

You can find a complete example code for reading the MiCS-2710 sensor in the following link:

<http://www.libelium.com/development/waspote/examples/ga-9-socket3b-sensor-reading>

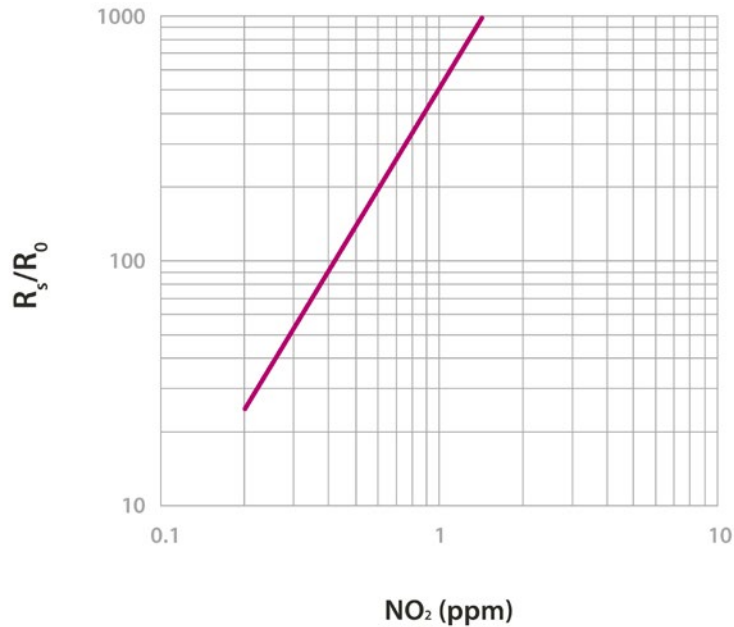


Figure : Graph of the sensitivity of the MiCS-2710 taken from the e2v's sensor data

## 4.10. Nitrogen Dioxide (NO<sub>2</sub>) Sensor - MiCS-2714

### 4.10.1. Specifications

This sensor is the new version for the MiCS-2710 sensor. The new version is provided since June 2014 and has similar specifications:

**Gases:** NO<sub>2</sub>

**Measurement range:** 0.05 ~ 5ppm

**Air resistance:** 0.8 ~ 8kΩ (typically 2.2kΩ)

**Sensitivity:** 6 ~ 100 (typically 55, ratio between the resistance at 0.25ppm and in air)

**Supply voltage:** 1.7 ~ 2.5V DC

**Operating temperature:** -30 ~ +85°C

**Response time:** 30 seconds

**Average consumption:** 26mA (throughout the complete power supply cycle in one second)



Figure : Image of the MiCS-2714 sensor

### 4.10.2. Measurement process

The MiCS-2714 is a sensor whose resistance varies in the presence of small concentrations of NO<sub>2</sub>. This value varies between 2kΩ and 2MΩ approximately, providing high accuracy throughout the output range. Unlike the rest of the board's gas sensors, which operate at a voltage of 5V, this sensor is powered through a 1.8V voltage regulator, with consumption of approximately 26mA. The sensor's resistance in air, as well as its sensitivity, can vary between different units, so it is recommended to calibrate each one of them before finally inserting them in the application.

This sensor must be connected in socket 3B of the board (its position is indicated in section "Connector 3"), and its output can be read through the capture commands of the SensorGasv20 library.

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_SOCKET3B, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET3B);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_SOCKET3B);
}
```

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor (minimum 1K $\Omega$ )

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

You can find a complete example code for reading the MiCS-2714 sensor in the following link:

[www.libelium.com/development/waspmote/examples/ga-9-socket3b-sensor-reading](http://www.libelium.com/development/waspmote/examples/ga-9-socket3b-sensor-reading)

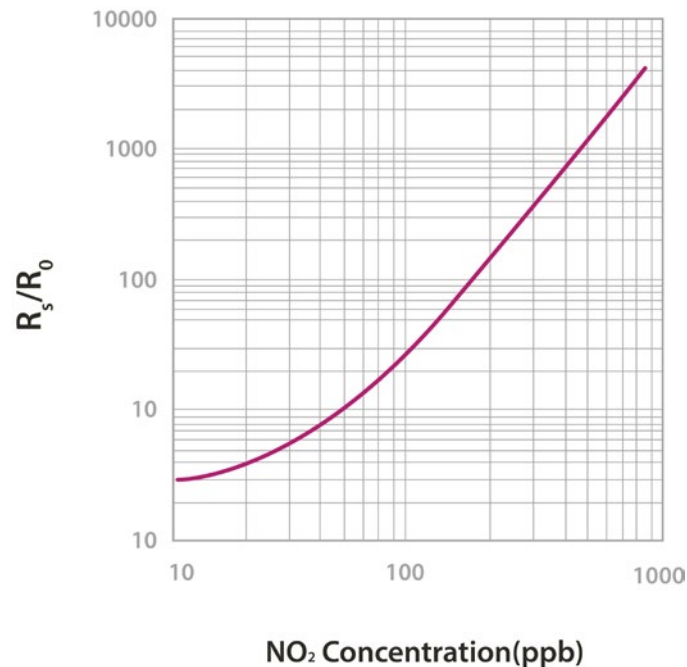


Figure : Graph of the sensitivity of the MiCS-2714 taken from the e2v's sensor data.

## 4.11. Ammonia (NH<sub>3</sub>) sensor – TGS2444

### 4.11.1. Specifications

**Gases:** NH<sub>3</sub>, H<sub>2</sub>S

**Measurement range:** 10 ~ 100ppm

**Resistance at 10ppm:** 3.63 ~ 36.3k $\Omega$

**Sensitivity:** 0,063 ~ 0.63 (ratio between the resistance at 3000 and at 1000ppm)

**Supply voltage:** 5V  $\pm$ 0.2V DC

**Operating temperature:** -10 ~ +50°C

**Response time:** 250ms

**Minimum load resistance:** 8k $\Omega$

**Average consumption:** 12mA (throughout the complete power supply cycle in 250ms)



Figure : Image of the TGS2444 sensor

### 4.11.2. Measurement Process

The TGS2444 sensor is a resistive sensor which is highly sensitive to variations in the concentration of Ammonia ( $\text{NH}_3$ ) and which shows slight sensitivity to hydrogen sulphide ( $\text{H}_2\text{S}$ ) and to a lesser extent, to Hydrogen ( $\text{H}_2$ ) and Ethanol ( $\text{CH}_3\text{CH}_2\text{OH}$ ). Both the sensor's initial resistance (at 10ppm) and its sensitivity vary widely between different sensors of the same model, so it is recommended to calibrate each one of them independently before finally including them in the application.

This sensor can be placed on connectors 3A and 4 following the direction indicated in sections "Connector 3" and "Connector 4". To read it, the necessary instructions are in the SensorGasv20 library:

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}
```

**SENSOR** indicates the socket into which it is to be inserted, for this sensor: **SENS\_SOCKET3NH3** or **SENS\_SOCKET4NH3**

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

You can find a complete example code for reading the TGS2444 sensor placed on socket 3 in the following link:

<http://www.libelium.com/development/waspmote/examples/ga-12-nh3-sensor-on-socket3-reading>

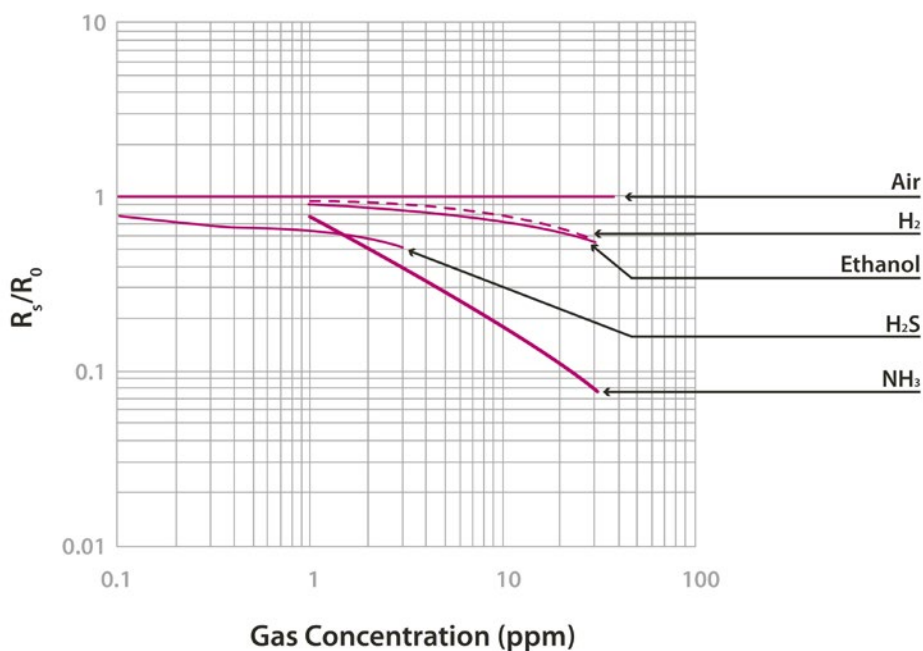


Figure : Graph of the sensitivity of the TGS2444 taken from the Figaro sensor data sheet

## 4.12. Methane (CH<sub>4</sub>) sensor – TGS2611

### 4.12.1. Specifications

**Gases:** CH<sub>4</sub>, H<sub>2</sub>

**Measurement range:** 500 ~ 10000ppm

**Resistance at 5000ppm:** 0.68 ~ 6.8kΩ

**Sensitivity:** 0.6 ± 0.06 (ratio between the resistance at 9000 and at 3000ppm)

**Supply voltage:** 5V ±0.2V DC

**Operating temperature:** -10 ~ +40°C

**Response time:** 30 seconds

**Minimum load resistance:** 0.45kΩ

**Average consumption:** 61mA



Figure : Image of the TGS2611 sensor

### 4.12.2. Measurement Process

The TGS2611 sensor shows a variable resistance with the concentration of CH<sub>4</sub> and to a lesser extent with the concentration of H<sub>2</sub>. The sensor's initial resistance (for 5000ppm) and its sensitivity may show large variations between different sensors of the same model, so it is recommended to consult the manufacturer's documentation and calibrate it before finally inserting it in the application.

This sensor can be used from connectors 2A, 3A and 4, placing it in the position referred to in sections "Connector 2", "Connector 3" and "Connector 4" and its output value can be read using the corresponding function in the SensorGasv20 library:

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENSOR);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}
```

**SENSOR** indicates the socket into which it is to be inserted, for this sensor:

**SENS\_SOCKET2A**, **SENS\_SOCKET3A** or **SENS\_SOCKET4A**

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains switched on before taking the measurement (minimum: 30000)



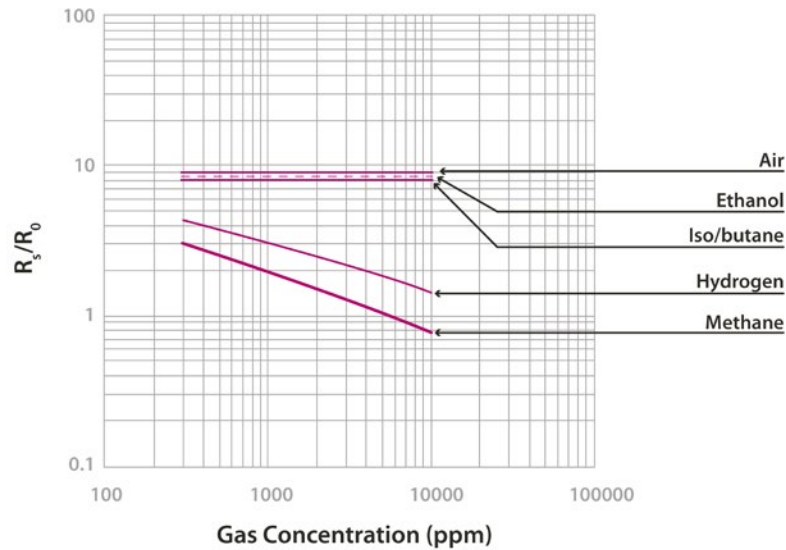


Figure : Graph of sensitivity of the TGS2611 taken from the Figaro sensor's data sheet

## 4.13. Liquefied Petroleum Gas Sensor - TGS2610

### 4.13.1. Specifications

**Gases:**  $\text{CH}_3\text{CH}_2\text{OH}$ ,  $\text{CH}_4$ ,  $\text{C}_4\text{H}_{10}$ ,  $\text{H}_2$

**Measurement range:** 500 ~ 10000ppm

**Resistance at 1800ppm (isobutane)** 0.68 ~ 6.8k $\Omega$

**Sensitivity:**  $0.56 \pm 0.06$  (ratio between the resistance at 3000 and at 1000ppm)

**Supply voltage:** 5V  $\pm 0.2$ V DC

**Operating temperature:** -10 ~ +40°C

**Response time:** 30 seconds

**Minimum load resistance:** 0.45k $\Omega$

**Average consumption:** 61mA



Figure : Image of the TGS2610 sensor

### 4.13.2. Measurement Process

The TGS2610 is a resistive sensor which shows sensitivity to combustible gases and derivatives. Especially reactive to Isobutane ( $\text{C}_4\text{H}_{10}$ ), it is also sensitive to Methane ( $\text{CH}_4$ ), Ethanol ( $\text{CH}_3\text{CH}_2\text{OH}$ ) and Hydrogen ( $\text{H}_2$ ). Because both its resistance and sensitivity show significant variations between different sensors of the same model, it is recommended to consult the manufacturer's documentation and carry out a process of calibration prior to its final inclusion in an application.

This sensor can be inserted in connectors 2A, 3A and 4, and must be placed in the position indicated in sections "Connector 2", "Connector 3" and "Connector 4". The sensor's output value can be read using the corresponding commands in the SensorGasv20 library:

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENSOR);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}
```

**SENSOR** indicates the socket into which it is to be inserted, for this sensor:

**SENS\_SOCKET2A**, **SENS\_SOCKET3A** or **SENS\_SOCKET4A**

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

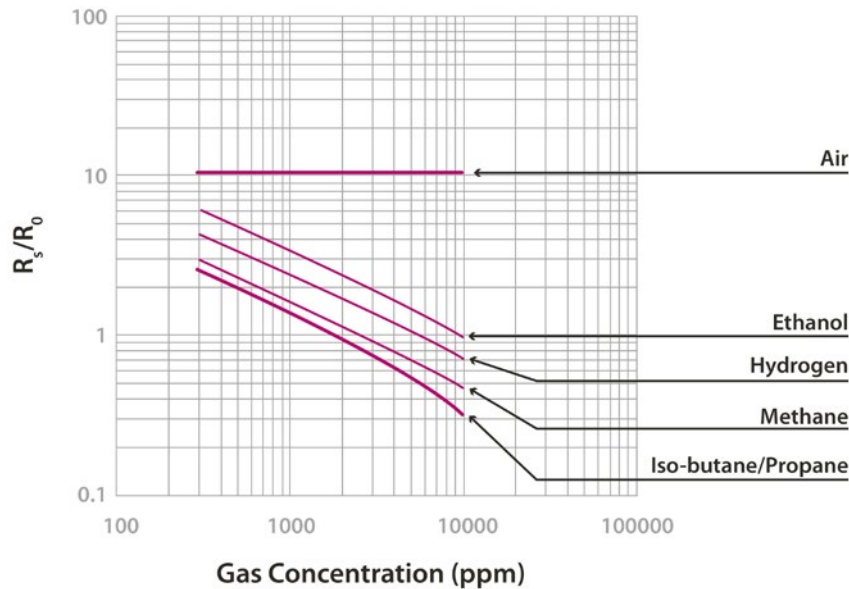


Figure: Graph of the sensitivity of the TGS2610 taken from the Figaro sensor's data sheet

## 4.14. Air Contaminants Sensor - TGS2600

### 4.14.1. Specifications

**Gases:**  $C_4H_{10}$ ,  $CH_3CH_2OH$ ,  $H_2$ ,  $CO$ ,  $CH_4$

**Measurement range:** 1 ~ 100ppm

**Air resistance:** 10 ~ 90k $\Omega$

**Sensitivity:** 0.3 ~ 0.6 (ratio between the resistance in 10ppm of  $H_2$  and in air)

**Supply voltage:** 5V  $\pm$ 0.2V DC

**Operating temperature:** -10 ~ +40°C

**Response time:** 30 seconds

**Minimum load resistance:** 0.45k $\Omega$

**Average consumption:** 46mA



Figure: Image of TGS2600 sensor

### 4.14.2. Measurement Process

The TGS2600 sensor shows sensitivity to the variation of the concentration of numerous gases that are not usually found in the composition of the atmosphere and which are considered contaminants. Amongst these would be mainly, Ethanol ( $CH_3CH_2OH$ ) and Isobutane ( $C_4H_{10}$ ) and, with less response, Carbon Monoxide ( $CO$ ) and Methane ( $CH_4$ ). This sensor is also sensitive to variations in the concentration of Hydrogen ( $H_2$ ). The sensor's resistance in air would vary between 10 and 90k $\Omega$ , with a ratio of sensitivity between 0.3 and 0.6 for an  $H_2$  concentration of 10ppm. Because of this variability it is recommended to calibrate each one of the sensors prior to their use in a final application.

This sensor can be placed on connectors 2A, 3A and 4 following the direction indicated in sections "Connector 2", "Connector 3" and "Connector 4" of this manual. To capture this output voltage value, specific instructions are in the SensorGasv20 library.

Reading code:

```

{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENSOR);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}

```

**SENSOR** indicates the socket into which it is to be inserted, for this sensor:

**SENS\_SOCKET2A**, **SENS\_SOCKET3A** or **SENS\_SOCKET4A**

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

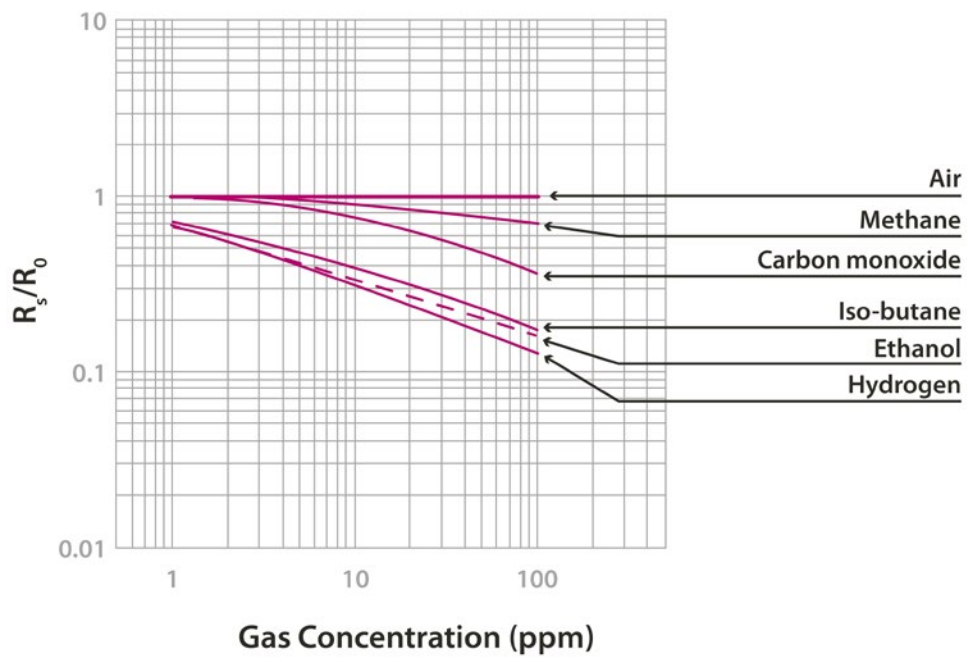


Figure : Graph of the sensitivity of the TGS2600 taken from the Figaro sensor's data sheet

## 4.15. Air Contaminants Sensor - TGS2602

### 4.15.1. Specifications

**Gases:**  $C_6H_5CH_3$ ,  $H_2S$ ,  $CH_3CH_2OH$ ,  $NH_3$ ,  $H_2$

**Measurement range:** 1 ~ 30ppm

**Air resistance:** 10 ~ 100k $\Omega$

**Sensitivity:** 0.15 ~ 0.5 (ratio between the resistance in 10ppm of Ethanol and in air)

**Supply voltage:** 5V  $\pm$ 0.2V DC

**Operating temperature:** +10 ~ +50 $^{\circ}C$

**Storage temperature:** -20 ~ +60 $^{\circ}C$

**Response time:** 30 seconds

**Minimum load resistance:** 0.45k $\Omega$

**Average consumption:** 61mA



Figure : Image of the TGS2602 sensor

### 4.15.2. Measurement Process

The TGS2602 is a sensor similar to the TGS2600 which reacts varying its resistance in the presence of contaminant gases, mainly Toluene ( $C_6H_5CH_3$ ), Hydrogen Sulphide ( $H_2S$ ), Ethanol ( $CH_3CH_2OH$ ), Ammonia ( $NH_3$ ) and to a lesser extent, Hydrogen ( $H_2$ ). In air without contaminants the sensor shows a resistance between 10 and 100k $\Omega$  with a variation ratio between 0.15 and 0.5 between the resistance in 10ppm of  $CH_3CH_2OH$  and this one. This variability makes a calibration of the sensor necessary before using it in a final application.

The TGS2602 sensor can be placed in sockets 2A, 3A and 4 in accordance with the indications shown in sections "Connector 2", "Connector 3" and "Connector 4". The output voltage values of these connectors can be read using the instructions in the SensorGasv20 library:

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENSOR);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}
```

**SENSOR** indicates the socket into which it is to be inserted, for this sensor:

**SENS\_SOCKET2A**, **SENS\_SOCKET3A** or **SENS\_SOCKET4A**

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

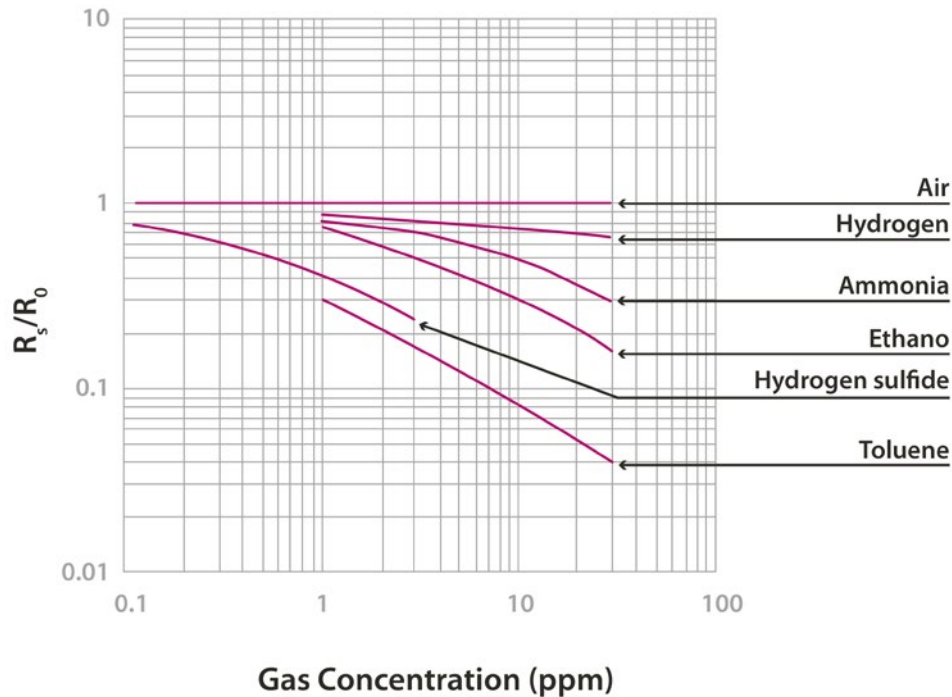


Figure : Graph of the sensitivity of the TGS2602 taken from the Figaro sensor's data sheet

## 4.16. Solvent Vapors Sensor - TGS2620

### 4.16.1. Specifications

**Gases:**  $\text{CH}_3\text{CH}_2\text{OH}$ ,  $\text{H}_2$ ,  $\text{C}_4\text{H}_{10}$ ,  $\text{CO}$ ,  $\text{CH}_4$

**Measurement range:** 50 ~ 5000ppm

**Resistance to 300ppm of Ethanol:** 1 ~ 5k $\Omega$

**Sensitivity:** 0.3 ~ 0.5 (ratio between the resistance at 300ppm and at 50ppm)

**Supply voltage:** 5V  $\pm$ 0.2V DC

**Operating temperature:** -10 ~ +40°C

**Response time:** 30 seconds

**Load minimum resistance:** 0.45k $\Omega$

**Average consumption:** 46mA (throughout the complete power supply cycle in 250ms)



Figure : Image of the TGS2620 Sensor

### 4.16.2. Measurement Process

The TGS2620 sensor allows detection of alcohol and organic gases, mainly Ethanol ( $\text{CH}_3\text{CH}_2\text{OH}$ ), Hydrogen ( $\text{H}_2$ ), Isobutane ( $\text{C}_4\text{H}_{10}$ ), Carbon Monoxide ( $\text{CO}$ ) and Methane ( $\text{CH}_4$ ). The resistance the sensor shows in a 300ppm concentration of Ethanol can vary between 1 and 5k $\Omega$ , while the sensitivity ratio between this and the resistance in 50ppm varies between 0.3 and 0.5. As a consequence of these variations it is necessary to calibrate each sensor before their insertion into a final application.

The sensor can be connected to the board through sockets 2A, 3A and 4, provided that their direction is as that shown in sections "Connector 2", "Connector 3" and "Connector 4". The necessary instructions for the reading of the sensor's output is in the SensorGasv20 library.

Reading code:

```

{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENSOR);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENSOR);
}

```

**SENSOR** indicates the socket into which it is to be inserted, for this sensor:

**SENS\_SOCKET2A**, **SENS\_SOCKET3A** or **SENS\_SOCKET4A**

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for it

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

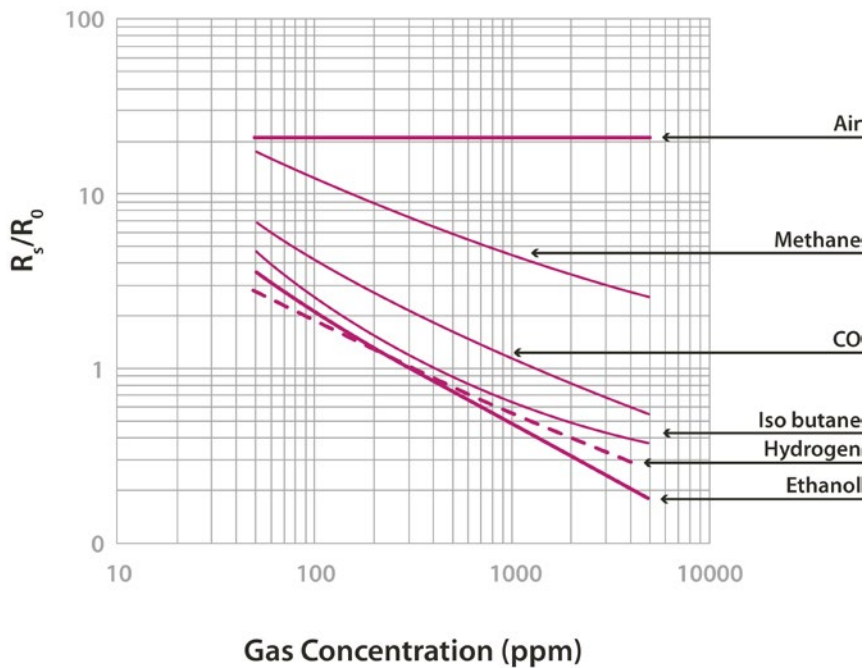


Figure : Graph of the sensitivity of the TGS2620 taken from the Figaro sensor's data sheet

## 4.17. Ozone (O<sub>3</sub>) Sensor - MiCS-2610

### 4.17.1. Specifications

**Gases:** O<sub>3</sub>

**Measurement range:** 10 ~ 1000ppb

**Air resistance:** 3 ~ 60kΩ (typically 11kΩ)

**Sensitivity:** 2 ~ 4 (typically 1.5, ratio between the resistance at 100ppm and at 50ppm)

**Supply voltage:** 1.95 ~ 5V DC

**Operating temperature:** -30 ~ +85°C

**Response time:** 30 seconds

**Average consumption:** 34mA



Figure : Image of the MiCS-2610 sensor

### 4.17.2. Measurement Process

The MiCS-2610 is a resistive sensor that allows to measure the variation of the  $O_3$  concentration between 10ppb and 1000ppb. Its resistance varies between 11k $\Omega$  and 2M $\Omega$  approximately. Unlike the MiCS-2710, this sensor is powered through a 2.5V voltage regulator, with consumption of approximately 34mA. The sensor's resistance in air, as well as its sensitivity, can vary between different units, so it is recommended to calibrate each one of them before finally inserting them in the application.

This sensor must be connected in socket 2B of the board (its position is indicated in section "Connector 2"), and its output can be read through the capture commands of the SensorGasv20 library:

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_SOCKET2B, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2B);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_SOCKET2B);
}
```

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

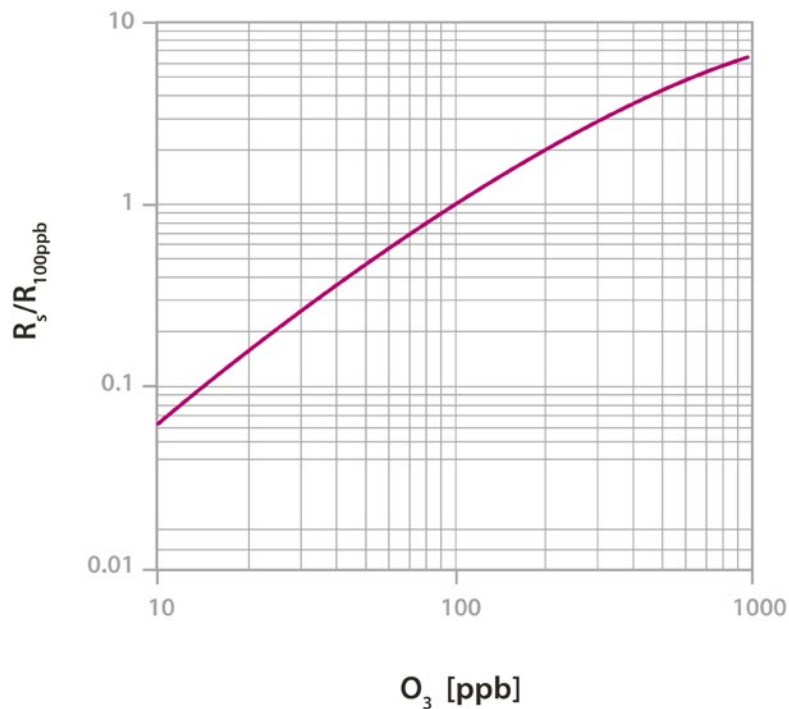


Figure: Graph of the sensitivity of the MiCS-2610 taken from the e2v's sensor data

## 4.18. Ozone (O<sub>3</sub>) Sensor - MiCS-2614

### 4.18.1. Specifications

This sensor is the new version for the MiCS-2610 sensor. The new version is provided since June 2014 and has similar specifications:

**Gases:** O<sub>3</sub>

**Measurement range:** 10 ~ 1000ppb

**Air resistance:** 3 ~ 60kΩ (typically 11kΩ)

**Sensitivity:** 2 ~ 4 (typically 1.5, ratio between the resistance at 100ppb and at 50ppb)

**Supply voltage:** 1.95 ~ 5V DC

**Operating temperature:** -30 ~ +85°C

**Response time:** 30 seconds

**Average consumption:** 34mA



Figure : Image of the MiCS-2614 sensor

### 4.18.2. Measurement process

The MiCS-2614 is a resistive sensor that allows to measure the variation of the O<sub>3</sub> concentration between 10ppb and 1000ppb. It's resistance varies between 11kΩ and 2MΩ approximately. Unlike the MiCS-2714, this sensor is powered through a 2.5V voltage regulator, with consumption of approximately 34mA. The sensor's resistance in air, as well as its sensitivity, can vary between different units, so it is recommended to calibrate each one of them before finally inserting them in the application.

This sensor must be connected in socket 2B of the board (its position is indicated in section "Connector 2"), and its output can be read through the capture commands of the SensorGasv20 library:

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_SOCKET2B, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2B);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_SOCKET2B);
}
```

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor (minimum 1KΩ)

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)



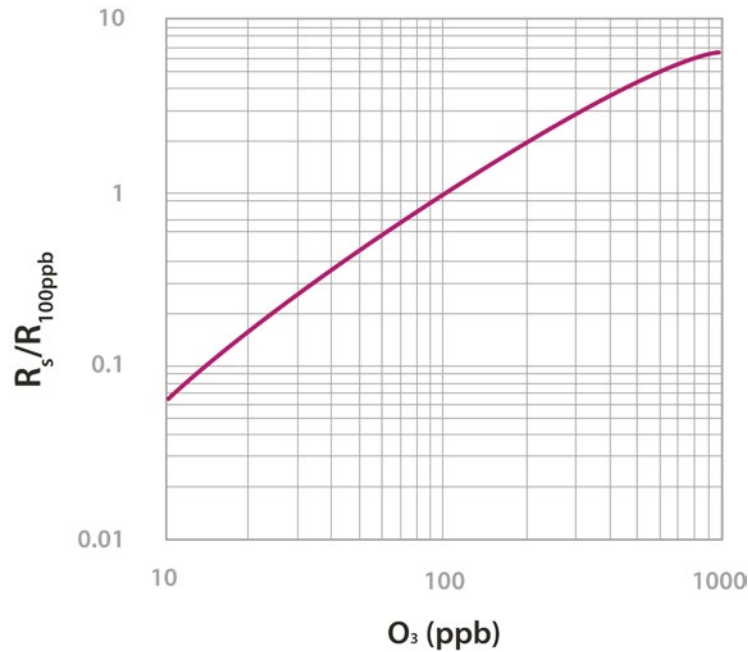


Figure : Graph of the sensitivity of the MiCS-2614 taken from the e2v's sensor data

## 4.19. VOC's Sensor - MiCS-5521

### 4.19.1. Specifications

**Gases:** CO, Hydrocarbons, Volatile Organic Compounds \*

**Measurement range:** 30 ~ 400ppm

**Air resistance:** 100 ~ 1000kΩ

**Sensitivity:** 1.8 ~ 6 (typically 3, ratio between the resistance at 60ppm and at 200ppm of CO)

**Supply voltage:** 2.1 ~ 5V DC

**Operating temperature:** -30 ~ +85°C

**Response time:** 30 seconds

**Average consumption:** 32mA



Figure : Image of the MiCS-5521 sensor

(\*) Chlorinated hydrocarbons, aromatic hydrocarbons, aromatic alcohols, aliphatic alcohols, terpenes, glycols, aldehydes, esters and acids. Detailed list can be found at <http://www.libelium.com/downloads/voc-sensors.xls>

### 4.19.2. Measurement Process

The MiCS-5521 is a resistive sensor that responds to a great variety of gases, such as Carbon Monoxide (CO), Hydrocarbons and Volatile Organic Compounds. Its resistance varies between 1000kΩ and 2kΩ approximately. Like the MiCS-2610, the MiCS-5521 is powered through a 2.5V voltage regulator, with consumption of approximately 32mA. The sensor's resistance in air, as well as its sensitivity, can vary between different units, so it is recommended to calibrate each one of them before finally inserting them in the application.

This sensor must be connected in socket 2B of the board (its position is indicated in section "Connector 2"), and its output can be read through the capture commands of the SensorGasv20 library.

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_SOCKET2B, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2B);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_SOCKET2B);
}
```

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

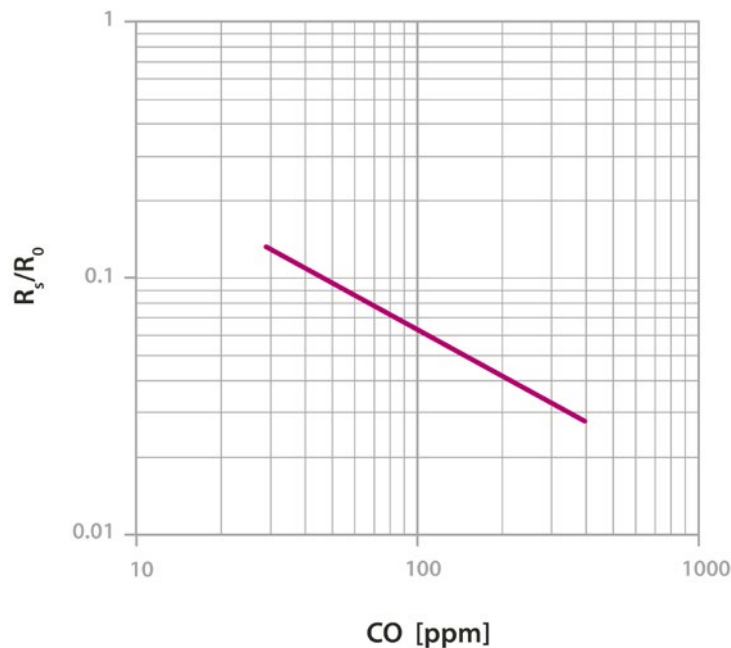


Figure : Graph of the sensitivity of the MiCS-5521 taken from the e2v's sensor data

## 4.20. VOC's Sensor - MiCS-5524

### 4.20.1. Specifications

This sensor is the new version for the MiCS-5521 sensor. The new version is provided since June 2014 and has similar specifications:

**Gases:** CO, Hydrocarbons, Volatile Organic Compounds \*

**Measurement range:** 30 ~ 400ppm

**Air resistance:** 100 ~ 1500k $\Omega$

**Sensitivity:** 1.8 ~ 6 (typically 3, ratio between the resistance at 60ppm and at 200ppm of CO)

**Supply voltage:** 2.1 ~ 5V DC

**Operating temperature:** -30 ~ +85°C

**Response time:** 30 seconds

**Average consumption:** 32mA



Figure : Image of the MiCS-5524 sensor

(\* ) Chlorinated hydrocarbons, aromatic hydrocarbons, aromatic alcohols, aliphatic alcohols, terpenes, glycols, aldehydes, esters and acids. Detailed list can be found at <http://www.libelium.com/downloads/voc-sensors.xls>

## 4.20.2. Measurement process

The MiCS-5524 is a resistive sensor that responds to a great variety of gases, such as Carbon Monoxide (CO), Hydrocarbons and Volatile Organic Compounds. Its resistance varies between 1000k $\Omega$  and 2k $\Omega$  approximately. Like the MiCS-2614, the MiCS-5524 is powered through a 2.5V voltage regulator, with consumption of approximately 32mA. The sensor's resistance in air, as well as its sensitivity, can vary between different units, so it is recommended to calibrate each one of them before finally inserting them in the application.

This sensor must be connected in socket 2B of the board (its position is indicated in section "Connector 2"), and its output can be read through the capture commands of the SensorGasv20 library.

Reading code:

```
{
  SensorGasv20.ON();
  SensorGasv20.configureSensor(SENS_SOCKET2B, GAIN, RESISTOR);
  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2B);
  delay(TIME);
  float value;
  value = SensorGasv20.readValue(SENS_SOCKET2B);
}
```

**GAIN** indicates the chosen gain for the sensor.

**RESISTOR** indicates the load resistance chosen for the sensor (minimum 1K $\Omega$ )

**TIME** is the time in milliseconds that the sensor remains on before taking the measurement (minimum: 30000)

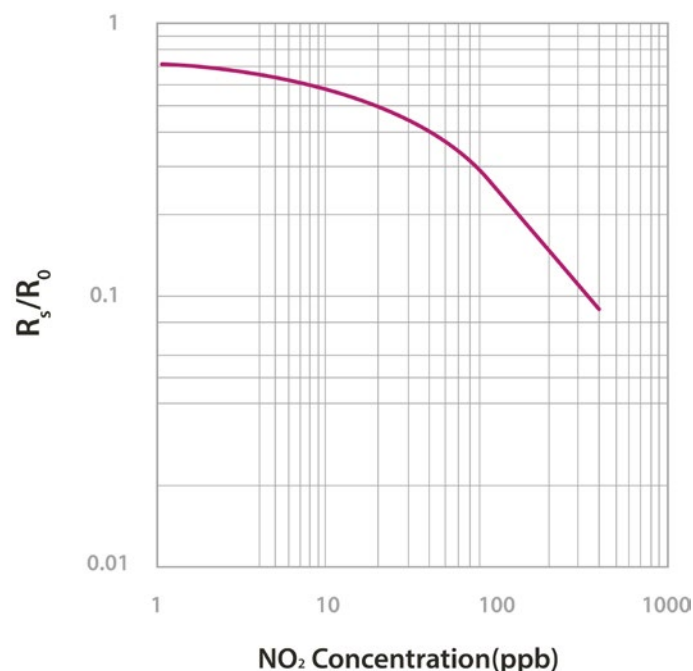


Figure : Graph of the sensitivity of the MiCS-5524 taken from the e2v's sensor data

## 4.21. Design and connections

The different connectors used for the sensors' connection can be used for the integration of different sensors to those previously planned, provided that the organization of the pins is followed, as well as the defined electrical specifications in the Waspote manual. In this sense, two types of different sensors are available:

Firstly, the three connectors for analog sensors, initially those for temperature, humidity and atmospheric pressure sensors (see components diagram in section "Hardware. Specifications"). These three connectors have only one strip of three pins which provide connection to ground, to 5V supply and to one of the microprocessor's analog inputs (ANALOG1, ANALOG4 and ANALOG5 respectively). In the case of the humidity and atmospheric pressure sensors' output, a voltage divider of resistances of 3.3K and 2.2K has been placed between these and the microprocessor's analog input ANALOG4 and ANALOG5 to adapt the sensor's output range (between 0 and 4.5V to that available in the Waspote input (between 0 and 3.3V).

Next the rest of the connectors used for gas sensors are described:

### 4.21.1. Connector 1

Connector 1 (see the components diagram in section "Hardware. Specifications") has been designed to connect 2 different sensors in the board simultaneously: TGS4161 (CO<sub>2</sub>) and SK-25 (O<sub>2</sub>), although its reading and powering is not permitted at the same time. In this way, the connection designed for the CO<sub>2</sub> sensor (which is named connector 1A) has two strips of three 2.54mm pitch female pins, of which two are without connection (the two in the center), one drives the sensor output to the amplification stage, the other the 5V supply controlled by a switch and two are for the circuit's connection to ground; while three isolated pins have been assigned to the O<sub>2</sub> sensor (connector 1B), one connected to ground, another to the sensor's output (accessing the same amplification stage as connector 1A) and another without connections assigned to the sensor's reference pin. The amplification stage of the connectors consists of a voltage follower followed by a non-inverting amplifier, whose gain is controlled through a 100KΩ digital potentiometer, which can be programmed by the user, up to a maximum value of 101. We can see an image with connector 1A and the connected CO<sub>2</sub> sensor, with reference to the placement highlighted in red, in the figure below.

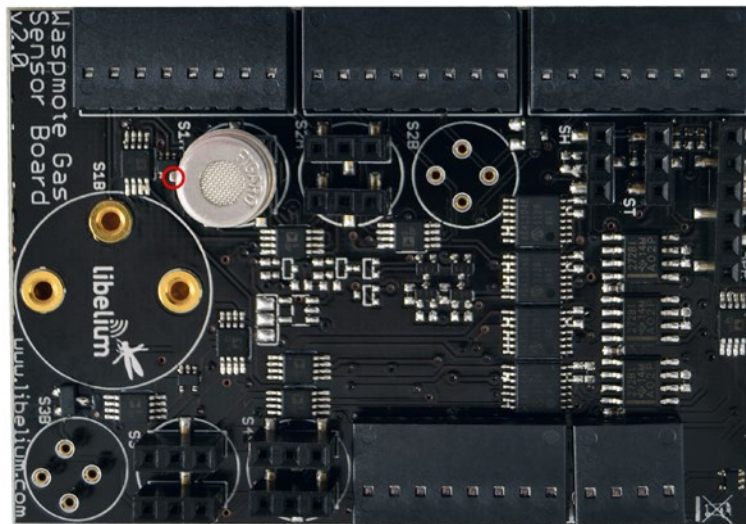


Figure : Image of socket 1A with a connected sensor

In the following links you can find two example codes for reading the CO<sub>2</sub> and O<sub>2</sub> sensors, that can be placed in sockets 1A and 1B respectively:

<http://www.libelium.com/development/waspote/examples/ga-4-co2-sensor-reading>

<http://www.libelium.com/development/waspote/examples/ga-5-o2-sensor-reading>

### 4.21.2. Connector 2

As with connector 1, connector 2 (which will be divided into connector 2A and connector 2B) has been designed to support connection of two different sensors at the same time but without allowing simultaneous reading or powering. In this case, the aim of this structure is to allow coexistence on the board of the MiCS-2610 or MiCS-2614 sensors for O<sub>3</sub> and the MiCS-5521 or MiCS-5524 sensors for VOC's (connector 2B) and the TGS2611 resistive sensors for CH<sub>4</sub>, TGS2600 and TGS2602 for air contaminants, TGS2610 for LPG gases and TGS2620 for solvent vapors (connector 2A). The difference between the last sensors and the TGS2442 for CO and TGS2444 for NH<sub>3</sub>, not compatible with this connector, is that the latter require the independent excitation of the heater supply voltage and of the sensor resistance supply voltage. Given that this connector's power is regulated by a single switch, these two last sensors cannot be inserted into this connector without risk of breakdown.

Thus, for connector 2B there are four isolated pins which provide two connections to the 2.5V power (taken from a regulator which receives the 5V power at its input), a connection to ground and another to a load resistance (100KΩ digital potentiometer) at the input of a non-inverting amplification stage with a maximum gain of 101 controlled through a 100KΩ digital potentiometer. Connector 2A has two strips of 2.54mm pitch female pins which provide the sensor with two connections to 5V supply, controlled by a switch, a connection to ground and another to output, at the same load resistance and amplification stage described previously. The two central pins of both strips, not used, remain without connection. An image of connector 2A can be seen in the figure below with a connected sensor and its reference for placement highlighted, and an image of connector 2B with a connected O<sub>3</sub> sensor and its reference highlighted can be seen in the second figure below.

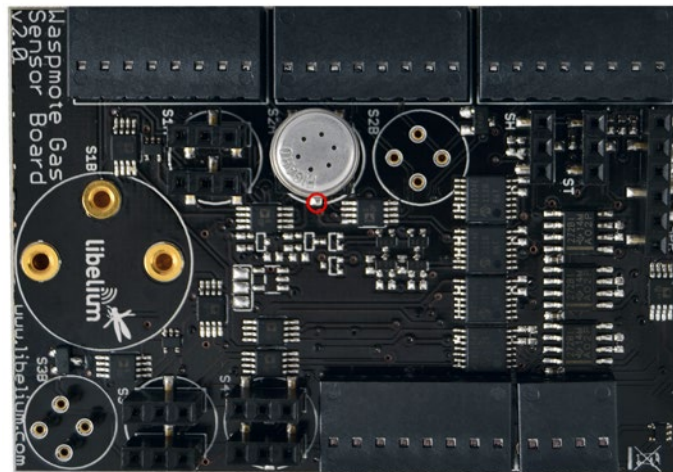


Figure : Image of socket 2A with a connected sensor

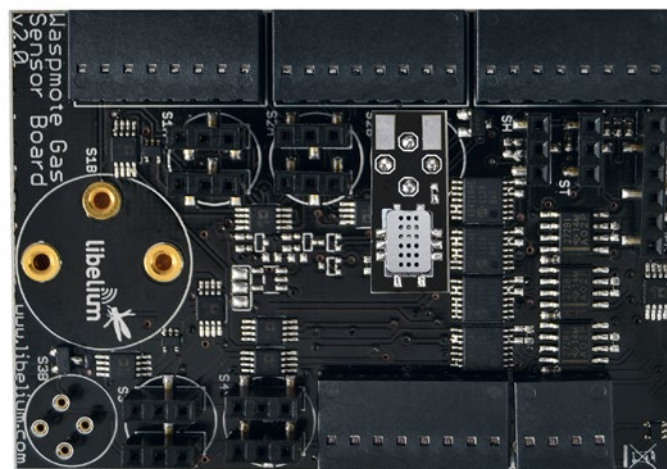


Figure : Image of socket 2B with a connected sensor (new version)

In the following links you can find two example codes for reading the sensors placed on sockets 2A and 2B respectively:

- <http://www.libelium.com/development/waspmote/examples/ga-6-socket2a-sensor-reading>
- <http://www.libelium.com/development/waspmote/examples/ga-7-socket2b-sensor-reading>

### 4.21.3. Connector 3

Connector 3 shows the same structure as connector 2: a socket (3A) composed of two strips of three 2.54mm pitch female pins (whose central pins are not connected to any signal) which provide two connections to 5V power supply and one connection to ground; and a connector (3B) of four pins whose supply voltage is regulated down to 1.8V. Both connectors share the same load resistor, configurable through a 100kΩ digital potentiometer, followed by an amplification stage with a maximum gain of 101 controlled through another digital potentiometer and whose output is connected to the microcontroller’s analog input ANALOG7.

This connector presents two differences respect the former one: first of all, power supply and ground connections in socket 3A are controlled through two switches, activating separately the sensor’s heating resistor and the measurement load resistor power circuits, thus allowing to connect sensors TGS2442 for CO and TGS2444 for NH<sub>3</sub>, as well as any of the sensors admitted by socket 2A (TGS2611 for CH<sub>4</sub>, TGS2600 and TGS2602 for air contaminants, TGS2610 for LPG gases and TGS2620 for solvent vapors). Secondly, connector 3B is endowed with a 1.8V regulator instead of the 2.5V of socket 2B in order to provide connection to the MiCS-2710 or MiCS-2714 sensors for NO<sub>2</sub> instead of the MiCS-2610, MiCS-2614, MiCS-5524 and MiCS-5521 sensors.

Since three different signals are used to control the activation of the sensors on both 3A and 3B sockets, they can be maintained powered at the same time, though to read the sensor on socket 3A it will be necessary to disconnect briefly sensors MiCS-2710 or MiCS-2714 (this process lasts a few milliseconds and is automatically carried out by the [readValue](#) function of the board’s API).

In the next 2 figures we have two images of both connectors with the sensor’s reference pin highlighted.

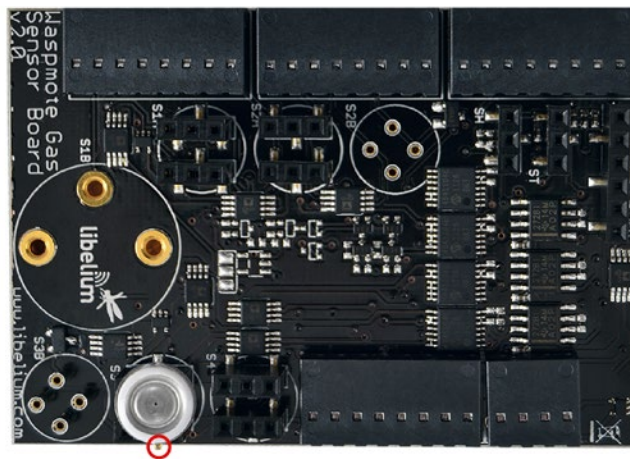


Figure : Image of socket 3A with a connected sensor

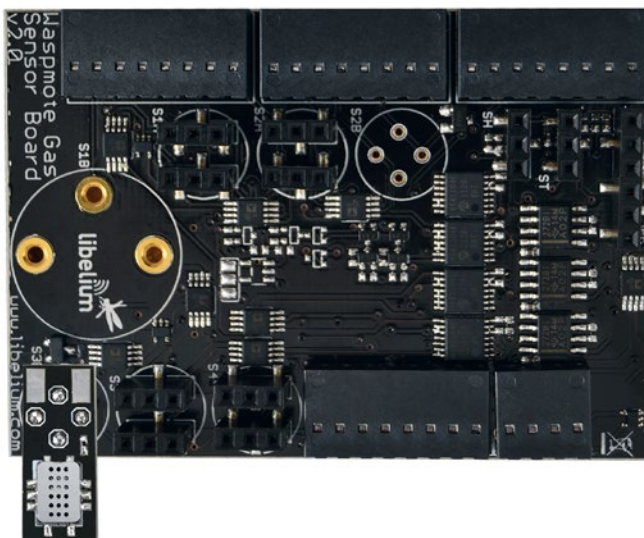


Figure : Image of socket 3B with the MiCS-2710 sensor connected (new version)

In the following links you can find two example codes for reading the NH<sub>3</sub> sensor placed on socket 3A, the remaining sensors that can be placed on that socket and the NO<sub>2</sub> sensor on socket 3B respectively:

<http://www.libelium.com/development/waspmote/examples/ga-12-nh3-sensor-on-socket3-reading>

<http://www.libelium.com/development/waspmote/examples/ga-8-socket3a-sensor-reading>

<http://www.libelium.com/development/waspmote/examples/ga-9-socket3b-sensor-reading>

#### 4.21.4. Connector 4

Socket 4 is the only one of those for gas sensors whose adaptation electronics is not shared by two different sockets. It is composed of two strips of three 2.54mm pitch female pins that connect the sensor to power supply, ground and output to a 100kΩ load variable resistor followed by an amplification stage of maximum gain 101, both controlled by a digital potentiometer. Like connector 3A, power supply circuits for heating and measurement sensor resistances are independently controlled, so the same sensors can be connected (TGS2442 for CO, TGS2444 for NH<sub>3</sub>, TGS2611 for CH<sub>4</sub>, TGS2600 and TGS2602 for air contaminants, TGS2610 for LPG gases and TGS2620 for solvent vapors).

In the figure below we have an image of a sensor placed on the socket with the reference pin highlighted.

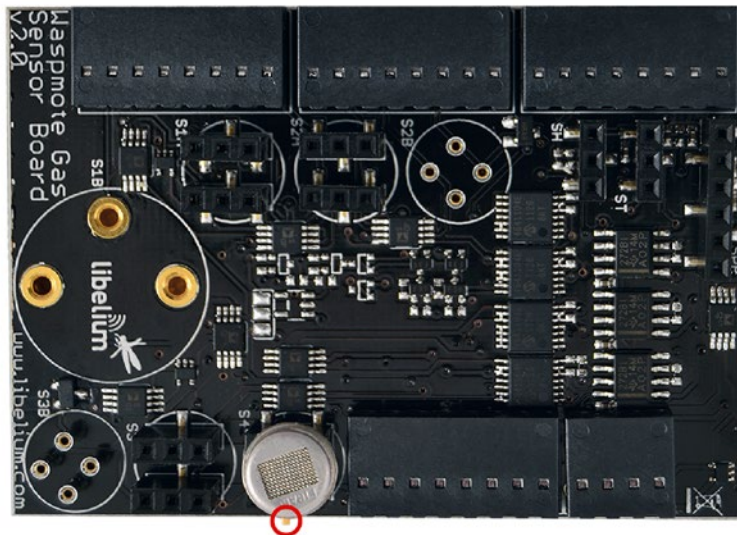


Figure : Image of socket 4 with a connected sensor

In the following link you have two example codes for reading the CO sensor on socket 4 and the remaining sensors placed on that sockets respectively:

<http://www.libelium.com/development/waspmote/examples/ga-11-co-sensor-on-socket4-reading>

<http://www.libelium.com/development/waspmote/examples/ga-10-socket4-sensor-reading>

### 4.21.5. Sockets for casing

In case the Gases 2.0 board is going to be used in an application that requires the use of a casing, such as an outdoors application, a series of sockets to facilitate the connection of the sensors through a probe has been disposed.

These sockets (PTSM from Phoenix Contact) allow to assemble the wires of the probe simply by pressing them into it. To remove the wire press the slot above the input pin and pull off the wire softly.

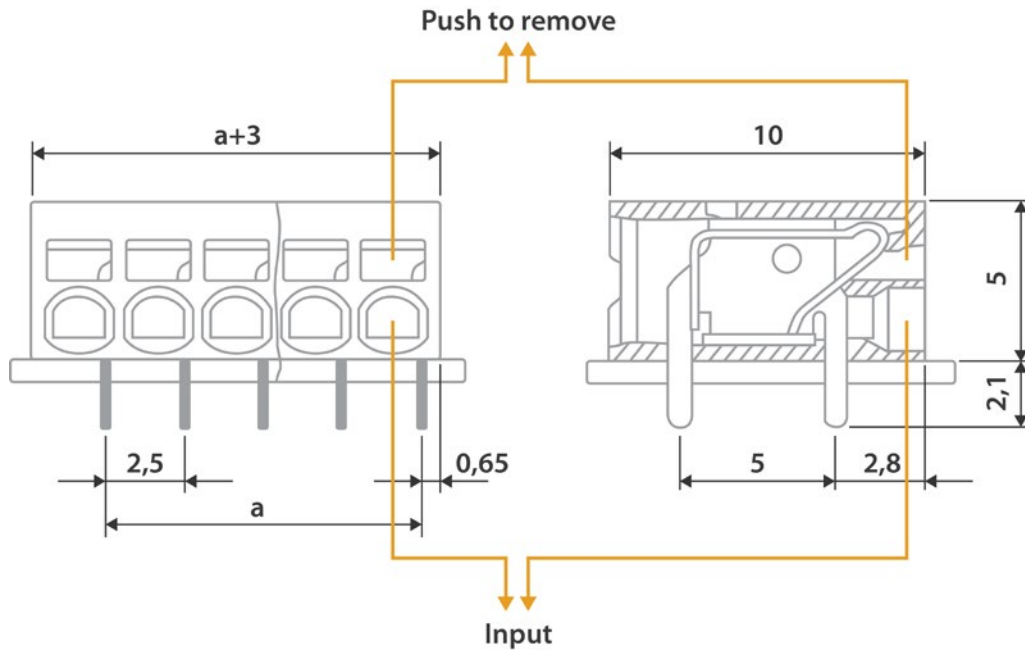


Figure : Diagram of a socket extracted from the Phoenix Contact data sheet



In the figure below an image of the board with the sockets in it and the correspondence between its inputs and the sensor's pins is shown.

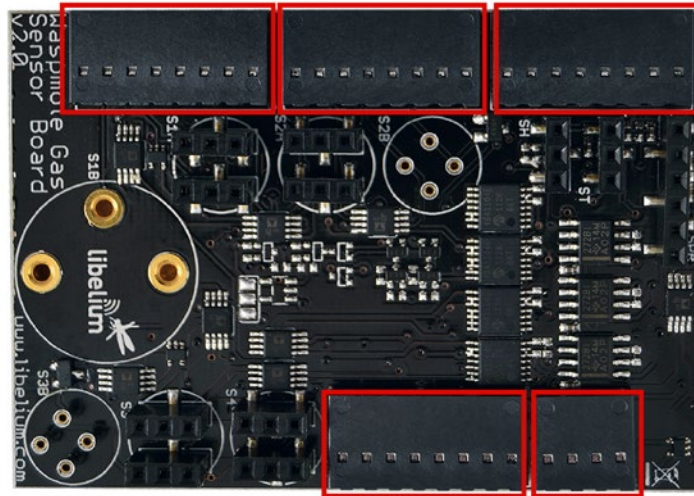


Figure : Image of the sockets for casing applications

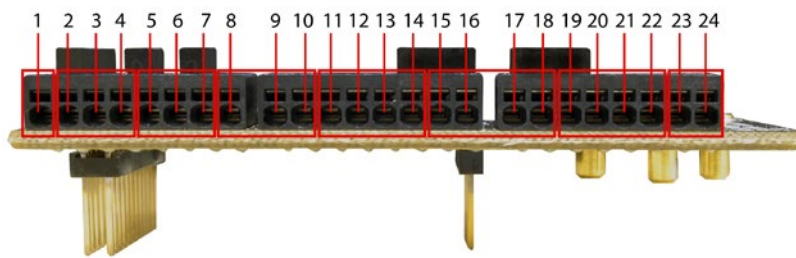


Figure : Image of the pin correspondence between the sockets and the sensors

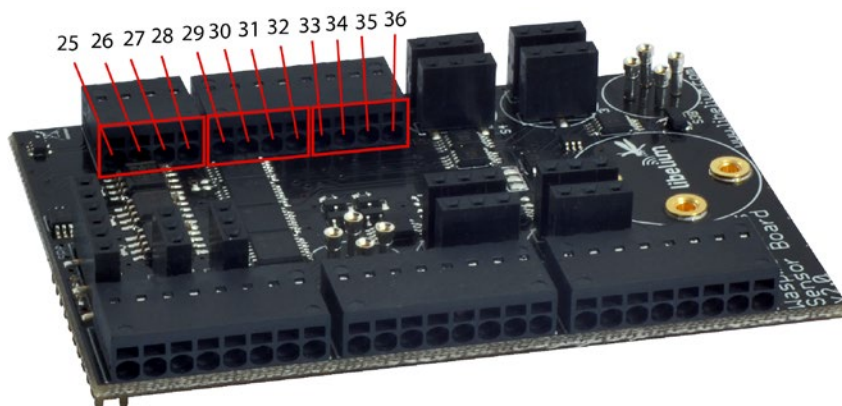


Figure : Image of the pin correspondence between the sockets and the sensors

Sensor	Pin	Function
Not used	1	Unconnected
Atmospheric Pressure Sensor	2	Supply Voltage
	3	Output
	4	Ground
	5	Supply Voltage
Temperature Sensor	6	Output
	7	Ground
	8	Supply Voltage
Humidity Sensor	9	Output
	10	Ground
	11	Supply Voltage
Socket 2B	12	Supply Voltage
	13	Load Resistor
	14	Ground
	15	Load Resistor
Socket 2A	16	Ground
	17	Supply Voltage
	18	Supply Voltage
	19	Output
TGS4161 CO <sub>2</sub> Sensor	20	Supply Voltage
	21	Ground
	22	Ground
	23	Output
SK-25 O <sub>2</sub> Sensor	24	Ground
	25	Ground
MiCS-2710 and MiCS-2714 NO <sub>2</sub> Sensors	26	Supply Voltage
	27	Supply Voltage
	28	Load Resistor
	29	Ground
Socket 4	30	Load Resistor
	31	Heater Supply Voltage
	32	Supply Voltage
	33	Ground
Socket 3A	34	Load Resistor
	35	Supply Voltage
	36	Heater Supply Voltage

## 5. Board configuration and programming

### 5.1. Hardware configuration

The Gases 2.0 board does not require any handling of the hardware by the user except for placing the sensors in their corresponding position. In the section dedicated to each connector we can see an image of each of the sensors inserted into the corresponding socket with the reference to the sensor's direction highlighted.

### 5.2. API

The Gases 2.0 board for Waspote has its own library which contains the set of necessary instructions to easily and intuitively configure and read each one of the sensors which connect to the board. Next each one of the functions is described and the process of configuration detailed for each sensor. The specific configuration which must be applied to each one of the sensors is explained in the specific sensor's section.

When using the Gases Sensor Board v20 on Waspote PRO, remember it is mandatory to include the SensorGasv20 library by introducing the next line at the beginning of the code:

```
#include <WaspSensorGas_v20.h>
```

Next, the different functions that make up the library are described:

**SensorGasv20.ON()**

Turns on the sensor board by activating the 3.3V and 5V supply lines.

**SensorGasv20.OFF()**

Turns off the sensor board by cutting the 3.3V and 5V supply lines.

**SensorGasv20.setBoardMode(MODE)**

This function is used to switch the sensor board power on and off. The variable **MODE** can take values **SENS\_ON**, to switch on the board, and **SENS\_OFF**, to switch it off.

**SensorGasv20.configureSensor(SENSOR, GAIN, RESISTOR)**

The function described in this section is used to establish parameters which affect taking the sensor's measurements: the socket in which the sensor is connected, the amplification stage gain at the output and the load resistance on which the reading is taken.

The value **SENSOR** indicates the connector on which the sensor has been placed. There are 7 possible connectors to choose, three of which are shared by a number of sensors while the three remaining are exclusive for a specific sensor.

The connectors exclusive to one sensor are:

- **SENS\_CO2**, which represents connector 1A, to which only the CO<sub>2</sub> sensor can be connected.
- **SENS\_O2**, which represents connector 1B, to which only the O<sub>2</sub> sensor can be connected.
- **SENS\_SOCKET3B**, which represents connector 3B, to which only the NO<sub>2</sub> sensor can be connected.

The connectors which are shared by a number of sensors are:

- **SENS\_SOCKET2A**, which is used for sensors placed in connectors 2A.
- **SENS\_SOCKET2B**, which is used for sensors placed in connectors 2B.
- **SENS\_SOCKET3A**, which is used for sensors placed in connector 3, when dealing with the CH<sub>4</sub>, LPG, solvent vapors and both air contaminant sensors.
- **SENS\_SOCKET3CO**, which is used when the CO sensor is placed in connector 3.
- **SENS\_SOCKET3NH3**, which is used when the NH<sub>3</sub> sensor is placed in connector 3.
- **SENS\_SOCKET4A**, which is used for sensors placed in connector 4, when dealing with the CH<sub>4</sub>, LPG, solvent vapors and both air contaminant sensors.

- `SENS_SOCKET4CO`, which is used when the CO sensor is placed in connector 4.
- `SENS_SOCKET4NH3`, which is used when the NH<sub>3</sub> sensor is placed in connector 4.

The variable `GAIN` represents the gain chosen for the amplification stage which is accessed from the sensor's output. Integer values can be set between 1 and 101.

The variable `RESISTOR` determines the load resistance on which the sensor acts before the amplification stage, expressed in kilohms. Values between 0 and 100kΩ can be taken (provided that the minimum load resistance shown in the specifications of each sensor is respected). In the case of the O<sub>2</sub> and CO<sub>2</sub> sensors, which do not require load resistance, this parameter may be omitted.

#### `SensorGasv20.setSensorMode(MODE, SENSOR)`

This function, analog to `setBoardMode`, allows independent configuration of the power supply of each sensor.

The variable `MODE` defines the status in which the sensor should be set, which can take the values `SENS_ON`, to connect the power, and `SENS_OFF` to disconnect it.

The variable `SENSOR` represents the connector on which the sensor is placed and the type of sensor used, which is able to take the following values:

- `SENS_CO2`, to control the CO<sub>2</sub> sensor's power and enable its output.
- `SENS_O2` to enable the O<sub>2</sub> sensor's output.
- `SENS_PRESSURE`, to control the atmospheric pressure sensor's power.
- `SENS_SOCKET3B`, to control the power of the NO<sub>2</sub> sensor and enable its output.
- `SENS_SOCKET2A`, to control the power of the sensor placed in connector 2A and enable its output.
- `SENS_SOCKET2B`, to control the power of the sensor placed in connector 2B and enable its output.
- `SENS_SOCKET3A`, to control the power of the sensor placed in connector 3 when one of the CH<sub>4</sub>, LPG, solvent vapors and both air contaminants sensors are connected to it (**CAUTION**: do not use this command when the CO or NH<sub>3</sub> sensors are placed in this connector, these sensors must not be permanently powered).
- `SENS_SOCKET4A`, to control the power of the sensor placed in connector 4 when one of the CH<sub>4</sub>, LPG, solvent vapors and both air contaminant sensors are connected to it (**CAUTION**: do not use this command when the CO or NH<sub>3</sub> sensors are placed in this connector, these sensors must not be permanently powered).

#### `SensorGasv20.readValue(SENSOR)`

The function `readValue` allows capture of the output value of each one of the board's sensors, returning the voltage value in a floating point (`float`) in the variable to which it has been assigned. The sensor whose output we want to read is defined by the variable `SENSOR`, which can take the following values:

- `SENS_TEMPERATURE`, for the reading of the temperature sensor.
- `SENS_HUMIDITY`, for the reading of the humidity sensor.
- `SENS_PRESSURE`, for the reading of the atmospheric pressure sensor.
- `SENS_CO2`, for the reading of the CO<sub>2</sub> sensor.
- `SENS_O2`, for the reading of the O<sub>2</sub> sensor.
- `SENS_SOCKET3B`, for the reading of the NO<sub>2</sub> sensor.
- `SENS_SOCKET2A`, for the reading of the sensor placed in connector 2A.
- `SENS_SOCKET2B`, for the reading of the sensor placed in connector 2B.
- `SENS_SOCKET3A`, for the reading of the sensor placed in connector 3, with the exception of the CO and NH<sub>3</sub> sensors.
- `SENS_SOCKET4A`, for the reading of the sensor placed in connector 4, with the exception of the CO and NH<sub>3</sub> sensors.

The values of the variable `SENSOR` for the reading of the NH<sub>3</sub> and CO sensors are the following:

- `SENS_SOCKET3CO`, to read the CO sensor placed in connector 3.
- `SENS_SOCKET4CO`, to read the CO sensor placed in connector 4.
- `SENS_SOCKET3NH3`, to read the NH<sub>3</sub> sensor placed in connector 3.
- `SENS_SOCKET4NH3`, to read the NH<sub>3</sub> sensor placed in connector 4.

The reason for reading these two sensors in a different way is that they do not normally remain powered, but must be powered by two small pulses of voltage when capturing the output data to avoid them being damaged. The function `readValue` provides these power supply pulses and automatically reads the sensors when one of the previous values is provided in its input variable.

#### **`SensorGasv20.calculateResistance(SENSOR, VALUE, GAIN, LOAD)`**

This function has been implemented to facilitate the calculation of the output of the resistive sensors attached to the board, enabling a direct conversion from the value measured in volts in function of the configuration parameters of the sensor.

The value returned is the resistance of the sensor, expressed in kilohms, and given in a floating point format. The parameters required by this function are the following:

- **SENSOR:** Through this parameter the socket upon which the sensor has been placed is specified. It may take the same values than for function `readValue`.
- **VALUE:** Here it is introduced the output voltage obtained from function `readValue`.
- **GAIN:** This parameter indicates the gain set for the amplification stage. It must be the same value used in function `configureSensor`.
- **LOAD:** It refers to the load resistor configured for the sensor voltage divider. It must be the same value used in function `configureSensor`.

The structure in which the code to read a sensor must be presented is the following:

1. The board is switched on using the function `SensorGasv20.ON`.
2. Configuration of the sensor's amplification gain and load resistance (the latter if necessary) using the function `SensorGasv20.configureSensor`.
3. Turn on the RTC to avoid possible conflicts in the I2C bus using the function `RTC.ON`.
4. The sensor is switched on (with the exception of the CO and NH<sub>3</sub> sensors) using the function `SensorGasv20.setSensorMode`.
5. Wait for the necessary amount of time for the sensor's response depending on the accuracy required. The delay function may be used (see the Waspote API manual).
6. Read the sensor using the function `SensorGasv20.readValue`.
7. If necessary, switch off the sensors and the board using the functions `SensorGasv20.setSensorMode` and `SensorGasv20.OFF`.

A specific example of the reading of each sensor can be found in the section dedicated to each one.

Below, an example code to read the Air Pollutants 2 sensor (TGS2600) placed on socket 2A is shown

```
/* -----Gases 2.0 Sensor Board example-----  
  
www.Libelium.com  
*/  
  
// Inclusion of the Events Sensor Board v20 library  
#include <WaspSensorGas_v20.h>  
  
// Inclusion of the Frame library  
#include <WaspFrame.h>  
  
// Inclusion of the XBee 802.15.4 library  
#include <WaspXBee802.h>  
  
// Pointer to an XBee packet structure  
packetXBee* packet;  
  
// Heating time  
#define TIME 30000
```

```
// Load resistor
#define RESISTOR 50

// Gain
#define GAIN 1

float value;
float resistance;

void setup()
{
  // Switch on the gases sensor board
  SensorGasv20.setBoardMode(SENS_ON);
  delay(100);

  // Init RTC
  RTC.ON();
  delay(100);

  SensorGasv20.configureSensor(SENS_SOCKET2A, GAIN, RESISTOR);

  SensorGasv20.setSensorMode(SENS_ON, SENS_SOCKET2A);
  USB.println("Heating Sensor");
  delay(TIME);
}

void loop()
{
  value = SensorGasv20.readValue(SENS_SOCKET2A);
  resistance = SensorGasv20.calculateResistance(SENS_SOCKET2A, value, GAIN, RESISTOR);

  // Create new frame (ASCII)
  frame.createFrame(ASCII, "Waspmote_Pro");

  // Add the value and the resistance calculated read to the frame composition
  frame.addSensor(SENSOR_AP2, value);
  frame.addSensor(SENSOR_STR, resistance);

  // Init XBee
  xbee802.ON();
  // Set parameters to packet:
  packet=(packetXBee*) calloc(1, sizeof(packetXBee));
  packet->mode=BROADCAST;

  // Set destination XBee parameters to packet
  xbee802.setDestinationParams( packet, "000000000000FFFF", frame.buffer, frame.length);

  // Send XBee packet
  xbee802.sendXBee(packet);

  // Turn off the XBee Module
  xbee802.OFF();
  delay(100);

  // Put the mote to sleep with pluviometer interruptions enabled
  PWR.deepSleep("00:00:10:00", RTC_OFFSET, RTC_ALM1_MODE1, UART0_OFF | UART1_OFF | BAT_
OFF);
}
```

The files of the sensor board itself are: **WaspSensorGas\_v20.cpp, WaspSensorGas\_v20.h**

They can be downloaded from: **[http://www.libelium.com/development/waspote/sdk\\_and\\_applications](http://www.libelium.com/development/waspote/sdk_and_applications)**

## 6. Consumption

### 6.1. Power control

On one side, the control of the Gases 2.0 board power can be carried out using the Waspote's general on/off system for the 3.3V and 5V supply lines, which allows the board to be totally switched on and off (0uA).

On the other hand, specific control mechanisms have been installed inside the sensor board using a system of solid state switches, allowing the independent digital control of each sensor power without the need to physically access the circuit, except for the humidity and temperature sensors, which are powered always the board is on. This way, activation and reading of each sensor can be programmed at the same main code, controlled by the microcontroller. In section "API" where the API libraries related to this board are presented and the use of each of these switches is clearly and precisely described, as well as the correct way to read each sensor.

### 6.2. Consumption table

In the following table the consumption shown by the board when active is detailed, from minimum consumption (constant, fixed by the permanently active components, such as the adaptation electronics and temperature and humidity sensors).

To find out the total consumption of the board with sensors integrated to the consumption of each connector, the consumption of each chosen sensor must be added together. This consumption can be consulted in the section for the sensor itself when all its characteristics are described.

	Switch ON	Switch OFF
<b>Minimum (Constant)</b>	2mA	0mA (Waspote switch)
<b>Connector 1-A (without sensor)</b>	0mA	0mA
<b>Connector 1-B (without sensor)</b>	0mA	0mA
<b>Connector 2-A (without sensor)</b>	0mA	0mA
<b>Connector 2-B (without sensor)</b>	0mA	0mA
<b>Connector 3-A (without sensor)</b>	0mA	0mA
<b>Connector 3-B (without sensor)</b>	0mA	0mA
<b>Connector 4 (without sensor)</b>	0mA	0mA

### 6.3. Low Consumption Mode

From the point of view of optimizing Waspote resources when the Gases 2.0 board is used, it is recommended that the following instructions are followed:

- Keep the board switched off while no measurement is being taken**  
 This is the most efficient method of lowering consumption when none of the parameters are being continually monitored. To completely disconnect the board's power, disable the switches that allow passage of the 3.3V supply, the 5V supply from Waspote (using the `SensorGasv20.setBoardMode` library function) and to the two I2C bus channels (SCL and SDA) using the command shown in the WaspotePWR library (more information on this can be found in the API manual). Do not forget to reconfigure gain and load resistances when switching it on again.
- Optimize the time the sensors are switched on depending on your application**  
 The accuracy of each sensor's measurement which can be obtained will vary depending on the time that it remains switched on or on the power supply cycles which are continually applied, depending on the type of sensor. Knowing the time required to take a measurement in a determined application will allow saving of consumption without losing resolution in the sampled value.
- Simultaneously activate the minimum number of sensors possible**  
 Given that the current allowed in the digital switches' output is limited (about 200mA), it is recommended to not overload them by activating a number of sensors at the same time which in total may surpass this current.

## 7. API Changelog

Keep track of the software changes on this link:

[www.libelium.com/development/waspmote/documentation/changelog/#Gases](http://www.libelium.com/development/waspmote/documentation/changelog/#Gases)



## 8. Documentation changelog

### From v4.7 to v4.8

- References to the new LoRa module
- Link to the new online API changelog

### From v4.6 to v4.7

- Radios table for Plug&Sense! updated
- References to the Calibrated Gas Sensor Line were deleted
- Added process to calibrate the O2 sensor manually

### From v4.5 to v4.6

- Added MiCS-2714 NO<sub>2</sub> sensor, MiCS-2614 O<sub>3</sub> sensor and MiCS-5524 VOC's sensor (new versions)
- Connector 2 and connector 3 for design and connection section updated
- Sockets for casing table updated

### From v4.4 to v4.5:

- Radios table for Plug&Sense! updated

### From v4.3 to v4.4:

- API changelog updated to API v008
- Calibrated sensor list updated

### From v4.2 to v4.3:

- Added section New line of Calibrated Gas Sensors and the API function [calculateConcentration](#)
- API changelog updated to API v006

### From v4.1 to v4.2:

- Added reference to the External SIM socket

### From v4.0 to v4.1:

- Added references to 3G/GPRS Board in section: Radio Interfaces

## 9. Maintenance

- In this section, the term “Waspote” encompasses both the Waspote device itself as well as its modules and sensor boards.
- Take care with the handling of Waspote, do not drop it, bang it or move it sharply.
- Avoid putting the devices in areas of high temperatures since the electronic components may be damaged.
- The antennas are lightly threaded to the connector; do not force them as this could damage the connectors.
- Do not use any type of paint for the device, which may damage the functioning of the connections and closure mechanisms.

## 10. Disposal and Recycle

- In this section, the term “Wasmote” encompasses both the Wasmote device itself as well as its modules and sensor boards.
- When Wasmote reaches the end of its useful life, it must be taken to a recycling point for electronic equipment.
- The equipment has to be disposed on a selective waste collection system, different to that of urban solid waste. Please, dispose it properly.
- Your distributor will inform you about the most appropriate and environmentally friendly waste process for the used product and its packaging.

