A LOWER-COST PM/NO2 AIR QUALITY MEASUREMENT UNIT WITH A LOW-COST AIR DRYER FOR STATIONARY OUTDOOR USE

Bernd Laquai*, Ioannis Chourdakis, Miriam Chacon, Abdul Samad, Nour Tamim, Grecia Solis, Ulrich Vogt
Institute of Combustion and Power Plant Technology, University of Stuttgart, Germany
*Corresponding author (for contact): e-mail: Bernd.Laquai@ifk.uni-stuttgart.de

February 6th, 2020

ABSTRACT

An outdoor measurement unit for particulate matter and NO_2 is presented, suitable for stationary use and local data storage on an SD-card. In contrast to professional equipment, it uses lower-cost sensors that reduce the cost per unit to less than 500Euro. The choice of lower-cost sensors in contrast to the very low-cost sensors result from a compromise between measurement accuracy and cost. This compromise appeared to be inevitable when scientific measurements must be executed under high variations of humidity and temperature in the outdoor environment and a matching of results to professional equipment is expected. A field measurement experiment using a prototype unit during a pollution episode in the city of Stuttgart showed a maximum deviation of +/-10ug/m^3 for the measured PM2.5 concentration compared to reference instrumentation and +/-7.1ppb maximum deviation for the measured NO_2 concentration.

INTRODUCTION

With the upcoming of low-cost PM sensors, a lot of commodity devices appeared on the market, mainly in Asia. These devices are mostly used as low-cost indicators of air pollution and assist citizens to identify health risks. In western countries, where high-quality air monitoring with spot measurements in larger cities was already in place, the idea popped up to use these low-cost devices for spatial pollution mapping either in scientific applications or in citizen-science crowd-mapping projects /12, 13, 14/. Due to the different focus on PM2.5 in Asia rather than on PM10 in Europe, a first difficulty popped up: the low-cost devices aren't able to correctly include particle fractions of more than 3um diameter in their PM10 result, they rather extrapolate to obtain a PM10 value assuming a standard size distribution /8/.

However, one sensor supplier in UK so far, came up with lower-cost spectrometer-based sensors, capable to correctly measure particle mass concentrations for size distributions even exceeding 5um. The cost of these sensors is by a factor of about 10 higher compared to the low-cost sensors. Nevertheless, these sensors can still be called "lower-cost sensors" in comparison to existing professional equipment. In addition, they are less power consuming and less bulky and don't use a noisy pump. These lower-cost PM sensors seem to meet a reasonable compromise between measurement accuracy even for PM10 and cost.

These lower-cost sensors had already been combined with electrochemical gas sensors for environmentally important toxic gases such as NO, NO_2, O_3 and CO to create air pollution measurement units for city authorities or larger companies as target customers. However, it quickly became evident, that these measurement units aren't well prepared for outdoor meteorological conditions including high temperature and humidity variations or even fog events. During a fog event, most of the laser scattering PM sensors used in these units just measure the mass concentration of fog water droplets.

In the meantime, it is well understood that the measurement accuracy of almost all low-cost air quality sensors are severely affected by meteorological influences of the outdoor environment. This is also the case for low-cost gas sensors. Unfortunately, some of the influences, such as fog droplets erroneously counted by laser scattering PM sensors, hardly can be compensated by smart software algorithms. The legally accepted measurement of particulate matter (PM) typically refers to the amount of the dry dust mass per air volume, whereas the low-cost laser scattering PM sensor determine the mass from the particle size irrespectively of the aerosol being wet, humid or dry.

Another PM measurement error that frequently occurs during episodes of high humidity is the growth of particle size due to the hygroscopic behavior of salts contained in the chemical particle composition. Since the growth factor strongly depends on the type of salts contained in particles and their specific deliquescence points as well as on the particle size, the growth effect is difficult to predict for compensation with software.

Low-cost electrochemical sensors for toxic gases such as NO_x or ozone also show a significant cross-sensitivity to water vapor and temperature. Low-cost metal-oxide (MOX) toxic gas sensors are extremely sensitive to temperature changes as well. In consequence, the specified range of operation for the low-cost gas sensors guaranteed by the manufacturers often does not match the humidity and temperature variations of the outdoor environment.

Furthermore, in order to make a smart sensor useful for crowd-mapping and scientific applications, a data logging capability is additionally required to record the measured data autonomously for a longer period. Even though IoT (Internet-of-Things) devices allow data to be transmitted into a cloud for logging, data safety and data privacy issues often prevent this option to be used for data logging. Additionally, data logging into the cloud often needs the presence of a wireless network service which is not always available in all locations of interest. Therefore, a local storage often is inevitable and needs to be added to the sensor device.

In our project, aiming at comparing indoor and outdoor PM and NO_2 gas concentrations, we decided to make use of lower-cost sensors and tried to not exceed a total cost goal 500 Euro for an outdoor measurement unit including a gas drying system. First, we started to develop an indoor measurement unit without gas drying system which we later extended for outdoor use by adding an additional simple and low-cost thermal dryer for the air provided to the PM sensor. Later it turned out that the PM dryer also has a very positive effect in limiting the humidity influence on the electrochemical gas sensor. Aside of this, the architecture of the outdoor unit is equal to the indoor unit but uses a more robust and water protected housing.

The gas sensor we used in our project is a NO2-B43F sensor for nitric dioxide from Alphasense, UK with its individual sensor board (ISB) provided by the manufacturer. The PM sensor we used is the spectrometer-based laser scattering sensor OPC-R1 from Alphasense, UK. We also added a temperature/humidity sensor to regulate the PM gas drying system (HYT221 from IST-AG, Switzerland).

In order to control the sensors and the dryer and to also provide the local data logging feature, we added two Arduino microcontroller boards, an external 16bit AD-converter to read in the gas sensor signals and an additional MOSFET device (metal-oxide-semiconductor field-effect transistor) acting as an electronic power switch for the electrical power supply to the dryer.

In addition to the software for the two microcontrollers, we also developed various software algorithms in Matlab for postprocessing the data. The postprocessing includes a calibration procedure based on a multi-variate regression of the NO_2 sensor data and the data of the temperature and humidity signals of the HYT221 sensor with respect to the data of a Model 405 nm NO_2/NO/NO_x Monitor from 2B Technologies used as reference instrument. The PM sensor data were compared against a Grimm EDM 180 dust monitor as reference instrument using a Nafion dryer.

In total, we built 3 prototype outdoor units named "SMbox" which we tested in a field experiment. The validation period ranged from November 2019 to January 2020, including the Silvester event, where air pollution typically is very high due to the public new year fireworks.

ARCHITECTURAL CONCEPT

The architectural concept of the measurement unit was derived from a historically grown concept that was originally designed as a mobile PM measuring unit for educational purposes in the subject of air quality monitoring at the University of Stuttgart. The design not only pursued the goal to train students for the use of the measurement unit for pollution monitoring but also to teach the construction of the unit with a maximally simple application of electronics that simultaneously kept the cost low. This design contained already an Alphasense optical particle counter (OPC-N3) combined with a GPS unit to allow geo-referenced measurements as well as a SD-card data logging unit (both Adafruit Industries, USA). All 3 units were individually controlled by Arduino microcontrollers, an open-source platform originally developed for educational purposes in Italy. An asynchronous serial communication link is used to transmit the measurement data from the OPC and the GPS units to the data logging unit that additionally gathered the data from the I^2C-based temperature/humidity sensor (HYT 221). Status information and a subset of data was displayed on a serial LCD-display. The modular approach of using 3 Arduino controller boards with a simple communication link particularly helped to develop the software per controller and makes the unit very flexible for changes.

Since this modular and educationally focused architecture seemed to be beneficial also for the unit to be designed for stationary outdoor use, we removed the GPS and the LCD display and added an Alphasense NO_2 sensor with its interface board (ISB) and an ADS1115 16-bit AD-converter (Adafruit Industries, USA) to convert the data to digital. For cost reasons, we exchanged the OPC-N3 PM sensor with the cheaper OPC-R1 from Alphasense. Finally, we developed a thermal dryer consisting of a 50cm tube heated by a resistive coil that we mounted on the inlet of the PM sensor. The dryer is controlled digitally using pulse width modulation (PWM) via a high current MOSFET switch (BUZ11A, ON Semiconductor, USA) by the Arduino board used for data logging. The modular concept of the initial educational measurement unit and the experience we had gathered with this unit greatly helped to quickly modify and extend the architecture to meet the requirements of the outdoor unit.

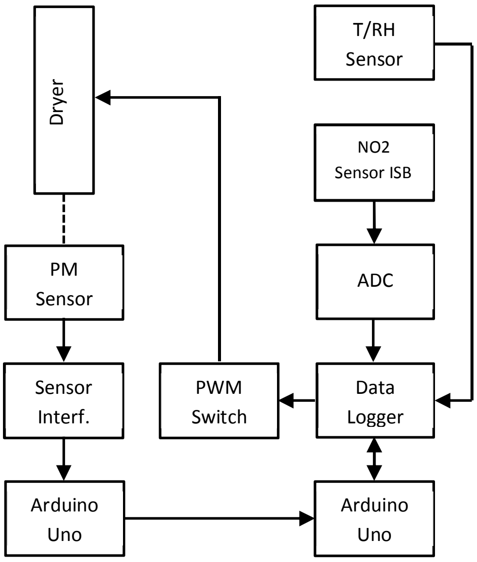


Fig. 1: Overall block diagram of the measurement unit showing the main data flow

The OPC-R1 requires a 5V supply, mainly for the fan but internally uses a 3.3V supply for its logic interface and electronics. The Arduino microcontroller that controls the OPC however, has 5V IO's, therefore a special OPC interface board was developed to provide power to the OPC and to translate the logic levels. An Arduino Uno is connected via a SPI interface to send control commands and to continuously receive the data measured by the OPC at a rate of 1 sample per second. The OPC sensor data transmitted to the data logging unit also contains the temperature and humidity of the input gas measured by an OPC internal sensor.

The NO_2 sensor attached to the ISB board outputs an analog voltage proportional to the NO_2 gas concentration (working electrode, WE). The ISB board additionally outputs an analog voltage signal from an auxiliary electrode (AE) not exposed to gas that reflects the sensor drift and temperature variations. It can be used to compensate these parasitic effects. Both signals WE and AE need to be converted from analog to digital with high resolution and precision. The ADS1115 analog digital converter (ADC) provides 2 differential channels that can be converted in sequence. These 2 channels are used to convert the working and auxiliary electrode signals with a resolution of 16 bits. These data are passed to the data logger.

The data logger contains a real-time clock (RTC) that provides date and time in terms of a timestamp. During the execution of the main measurement loop, program execution waits until data from the OPC Arduino are available, then the RTC is polled for obtaining a time stamp, the ADC is requested to perform a conversion of working and auxiliary electrode data, and finally the temperature/humidity data are polled from the HYT221 sensor. Together with the status signal from the dryer (on/off), this forms a sample data set assigned to a single time stamp which gets stored on SD-card as one line of text.

After storing the data, the dryer control is updated. For this purpose, the humidity of the HYT221 sensor exposed to outside air is evaluated. In case the humidity is above a given threshold, the temperature of the inlet gas temperature is evaluated by the internal sensor of the OPC and in case it is within a given window, the dryer switch is activated that turns the dryer-on. The dryer tube is imposed on the OPC-R1 sampling inlet and heats the air that is sucked into the OPC-R1. After the dryer control is updated, the main measurement loop repeats.

ELECTRONIC CONTROL

The electronic control is mainly designed around the Arduino microcontroller boards. For cost reasons, the cheapest Arduino board named "UNO" was chosen. It is using an Atmel 328P microcontroller that can be programmed via an open source development platform using the C++ programming language. The Arduino microcontroller boards are readily available from various manufacturers. The choice of the Arduino platform mainly determined the way the peripherals are attached to arrive at the desired overall architecture.

HARDWARE ARCHITECTURE

An important design aspect when using the Arduino UNO microcontroller board is the fact that an UNO board just provides one SPI interface (serial peripheral interface). Since the OPC as well as the SD-card use an SPI bus exclusively, two UNO boards are required, one for the OPC and one for the data logging unit.

Furthermore, since the OPC fan was expected to generate noise and requires a high inrush current when it turns on, the PM sensor supply of 5V was separated from the Arduinos and from the other electronics. The OPC was therefore equipped with its own regulator attached to the main 12V supply. This regulator together with a level converter for the SPI logic levels was assembled on an individual OPC interface board. The level converter converts between 3.3V used by the OPC logic and the 5V Arduino IO's and is implemented as a passive resistive divider.

The data logging board, also containing the RTC, is readily available as a so-called shield, a PCB that can directly be attached to the IO pin headers of the Arduino UNO board. When stackable pin headers are also assembled on the data logging shield, a further board can be stacked upon it. We used this approach to attach an additional board on the data logging shield on which we mounted the ADC break out board for the ADS1115 AD-converter. The NO_2 sensor plugged onto the Alphasense ISB interface board as well as the HYT221 temperature/humidity sensor are attached with cables to the PCB boards, since they need to be mounted into holes of the housing with the sensor surface exposed to the outside air. The HYT221 sensor as well as the ADS1115 ADC use a standardized I^2C bus for communication, an addressed bus for which the Arduino provides a dedicated interface. The gate of the MOSFET used as PWM switch is directly controlled by an Arduino digital output pin. For the detailed circuit diagrams, see the appendix A.

Low-Cost Dryer

The dryer was constructed from a 50cm brass tube with a 9mm inner diameter. The outer surface of the tube was first covered with a ceramic foil to improve the heat transfer from the thermal heating. It additionally provides electrical isolation. The heating is achieved by a resistive wire with a specific resistance of 0.975Ohms per meter. A total length of about 15m of wire is wound around the tube to form a coil with approx. 500 windings. Using a power supply of 12V the electrical power dissipated is around 10Watt when the PWM duty cycle is 100%.

The coil is covered to the outside with a foam for thermal isolation. The thermally isolated tube is then inserted into a plastic tube with 32mm diameter for protection and to achieve mechanical stability.

On one end the brass tube is soldered to a copper plate that is mounted on the OPC-R1 such that the inlet nozzle dives into the tube. On the other end of the tube, a rain protection roof was mounted, and a fine wire grid was attached to protect insects and larger objects from entering the tube.

During our initial tests, we monitored the sample flow rate as measured by the OPC while attaching tubes with different diameters to ensure that the used tube does not visibly cause a drop of the sample flow rate due to a high flow resistance. According to our research no influence is visible starting at inner tube diameters larger than 7mm. On the other hand, a larger diameter than 10mm would unnecessarily increase the heating power to achieve the same drying effectiveness.

In order to validate the dryer effectiveness, we used an ultrasonic fog generator (Boneco, Switzerland) to generate small water droplets (maximum of the mass size distribution ≈2um). We used an OPC-R1 with dryer and compared the PM2.5 results to an OPC-N3 without dryer after calibrating the OPC-R1 to the OPC-N3. Having reached an initial concentration in this measurement, we first let the larger droplets sediment. After more than an hour, we turned the dryer-on and after a further hour we turned the dryer off again.

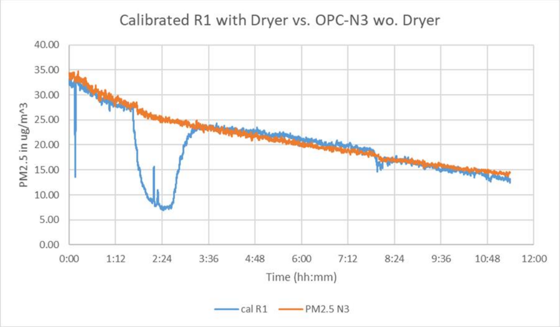


Fig. 2: Validation of the dryer effectiveness

As it can be seen from fig. 2, the dryer reduces the particle mass concentration by a factor of about 3.5, when the dryer is on and after it has stabilized thermally. This effectiveness appeared to be enough and was not increased further, since also care must be taken not to overheat particles during the drying process. Many particles contain volatile carbon hydrates that also easily evaporate when the gas temperature increases too much. To our experience the thermal energy to revert the hygroscopic growth effect requires much less energy compared to evaporating fog droplets. On the other hand, fog events seem to be less frequent compared to episodes with high humidity and small amounts of liquid water content in the air.

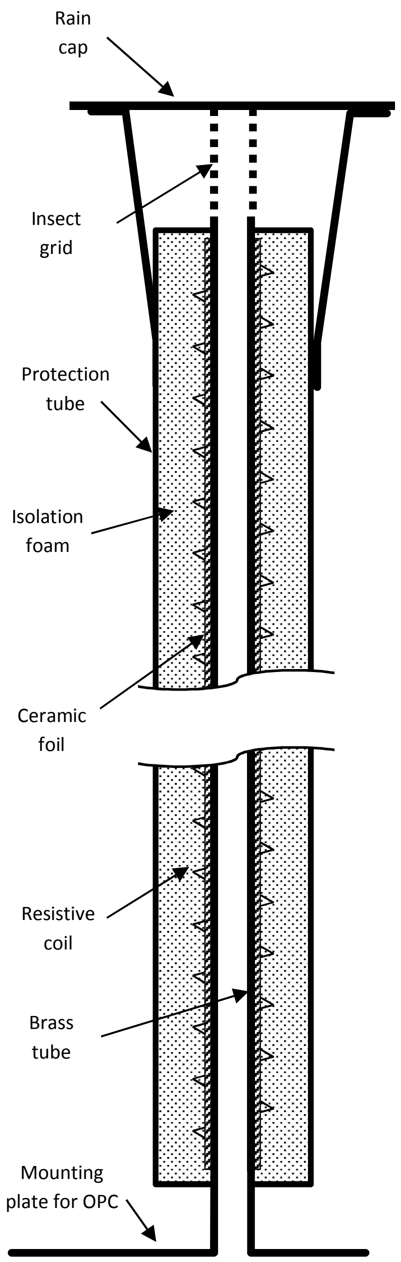


Fig. 3: Construction of the thermal dryer

HOUSING

As a housing, we used a 200x300x150 mm plastic enclosure with a hinged door (type KS 1423.500 from Rittal GmbH & Co. KG, Germany). It is made of fiberglass-reinforced unsaturated polyester and is specified for an IP 66 water protection category. Inside the housing, a 145x250mm mounting plate was installed on which to plates are mounted perpendicularly, one carries the electronics boards and the other one is used to attach the dryer tube. The AD-converter board and the datalogging board is mounted with stackable headers on the data logging Arduino, the PWM switch board and the OPC interface board were designed and manufactured using a PCB shop and are mounted to the left and the right of the Arduino boards.

The installation of the NO_2 sensor and the HYT221 humidity/temperature sensor with the sensor membrane faced to outside air requires the arrangement of holes in one side wall of the housing. Since the thermal heating of the dryer inside the housing will also affect the NO_2 sensor performance, the temperature/humidity sensor was installed right aside the NO_2 sensor. Further holes need to be arranged for the exhaust air of the PM sensor. In order to maintain rain and spray water protection, the exhaust air holes were arranged nearby the NO_2 sensor and the temperature/humidity sensor and a side-wall roof was mounted to cover this area (fig. 5) The influence of the small amounts of exhaust air as well as the reactive behavior of the copper material of the side wall roof on the NO_2 gas sensor performance proved to be negligible.

At the same side wall, a water protected power connector was installed to supply the 12V voltage. The total power consumption turned out to be around 12Watt during the on state of the dryer.

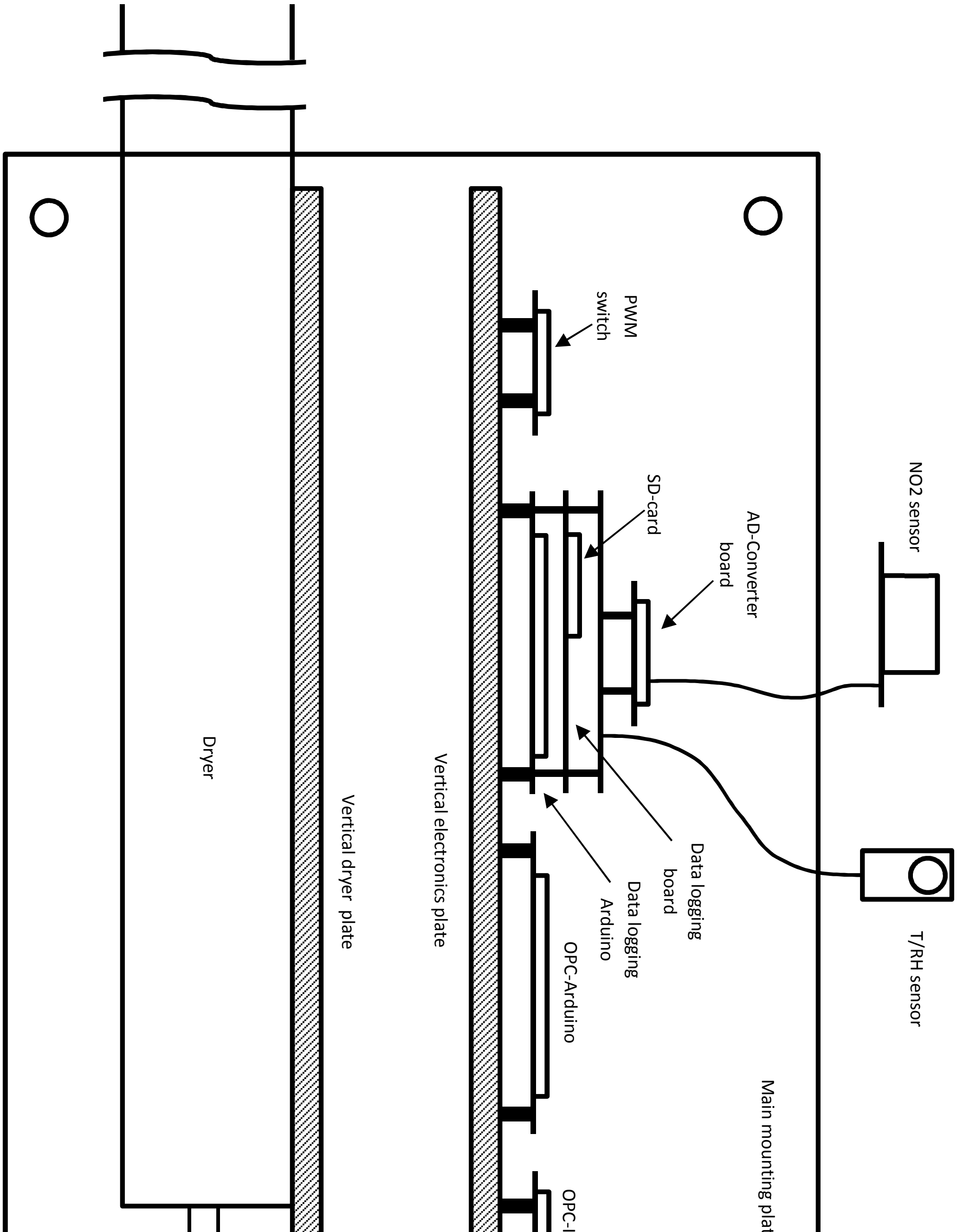


Fig. 4: Assembly of the mounting plate that carries the electronic boards and the dryer

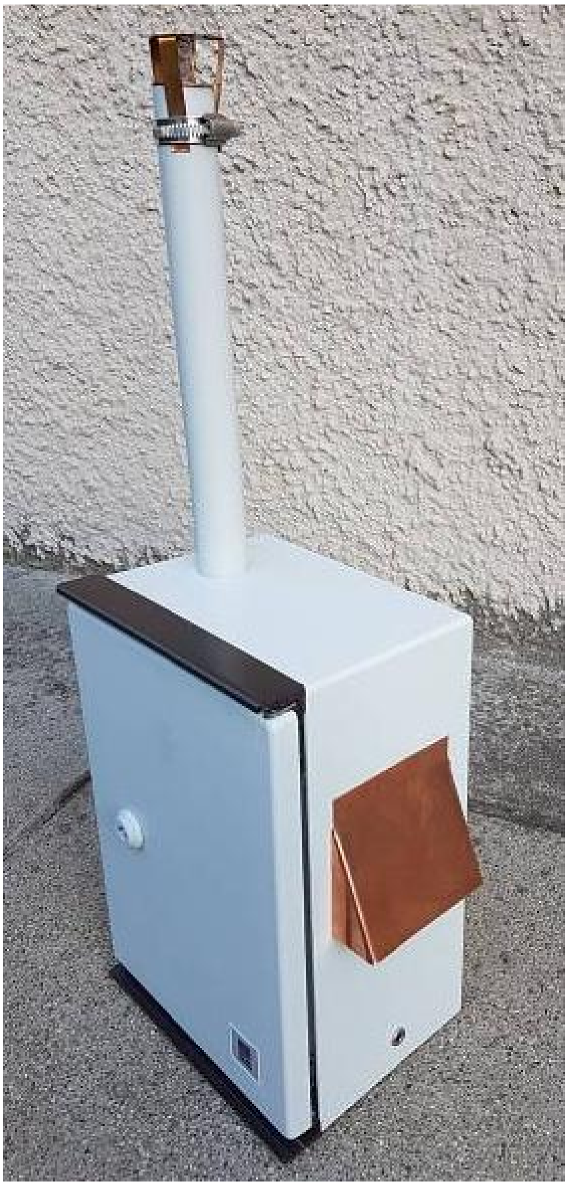


Fig. 5: One prototype of the SMbox measurement unit, on the side wall the copper roof is visible covering the NO_2 and temperature/humidity sensor as well as the OPC sensor exhaust

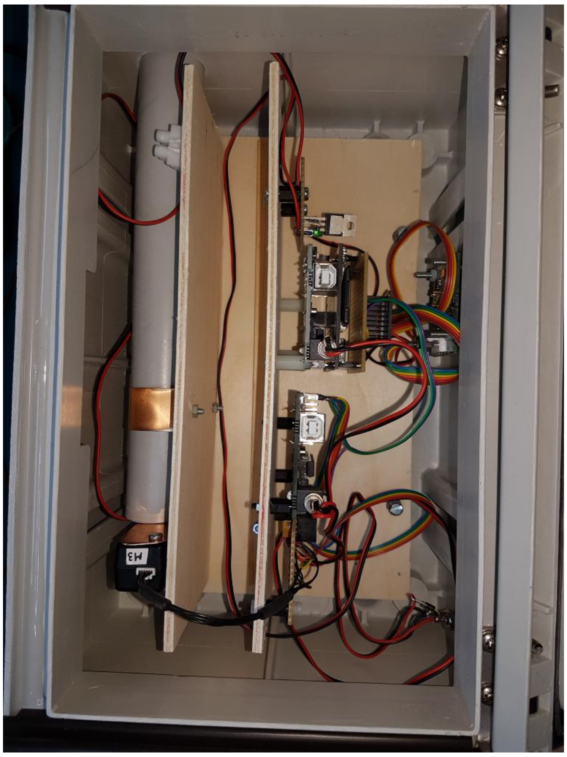


Fig. 6: The interior of one measurement unit prototype

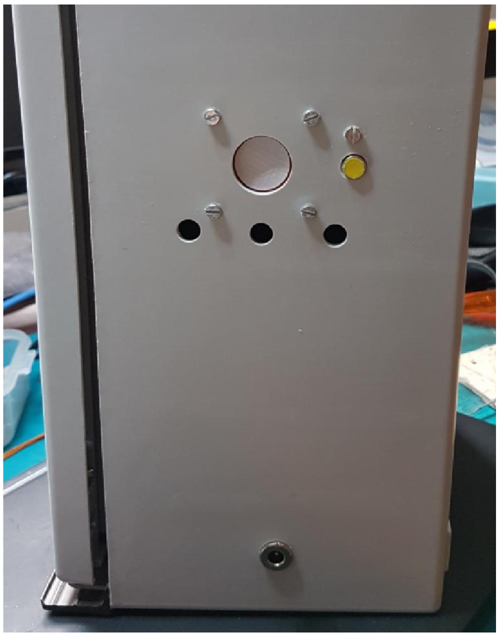


Fig. 7: PM sensor exhaust holes and the installed NO_2 and humidity/temperature sensor, at the bottom the 12V power supply connection

SOFTWARE ARCHITECTURE

The processor hardware of the measurement unit represents a multi-processor architecture distributed across two Arduino microcontrollers with independent processor clocks. Therefore, the software running in parallel on the two boards requires a synchronization mechanism, that governs the overall software architecture. Furthermore, it requires a communication channel to exchange data between the two boards.

The synchronization mechanism is based on a master-slave system. The OPC-Arduino acts as the pace-making master that communicates with the OPC-R1 sensor via its SPI interface for control and to request the measurement data. Upon receipt of a data set from the OPC-R1 sensor device, it uses an asynchronous serial communication protocol (EasyTransfer Arduino Library see /11/) to send out a subset of the data in a burst to the data logging Arduino that acts as a slave board. The data logging Arduino runs in a loop waiting for data from the OPC-Arduino. As soon as the serial communication link indicates a complete data reception from the OPC-Arduino, it executes a data logging cycle.

A data logging cycle starts with reading the real-time clock (RTC) available on the data logging shield. A sample set written to the SD-card forms a single data line of ASCII text, with the different data values of the set being separated by tabulator characters. Using date and time provided from the RTC, a time stamp is created which is written as first data item in a sample set to the SD-card, followed with the data values from the OPC-R1, the data values from the gas sensor, the values from the external temperature/humidity sensor (HYT221) and the dryer-on signal sent to the dryer control board. Date and time are written in the format “dd.mm.yy hh:mm:ss” with date and time being separated with a space. A sample set stored in one text line on the SD-card finally consists of the items shown in tab. 1.

Tab. 1: Data items of a sample set logged on the SD-card

Date-Time
Bin1 OPC-R1
Bin2 OPC -R1
…
Bin16 OPC-R1
T OPC-R1
RH OPC-R1
WE NO2
AE NO2
PM1 OPC-R1
PM2.5 OPC-R1
PM10 OPC-R1
T HYT221
RH HYT221

Before data can be logged, a complete set of OPC data must be available. Therefore, the waiting time caused by looping the data logging unit without action is responsible for the synchronization. The software running on the data logging Arduino waits until the serial link signals the complete reception of OPC data, then the temperature and humidity sensor as well as the AD-converter must be read for obtaining the respective data. To obtain the control signal for the dryer, the data of the OPC and the data from the temperature must be evaluated and a decision must be made whether to turn on the dryer or not.

The data logged to the SD-card are also output to the serial monitor of the Arduino integrated development environment (IDE) to assist debugging. For the software code see appendix B.

The overall functionality of the software running on the datalogging Arduino can be visualized in the following flow graph:

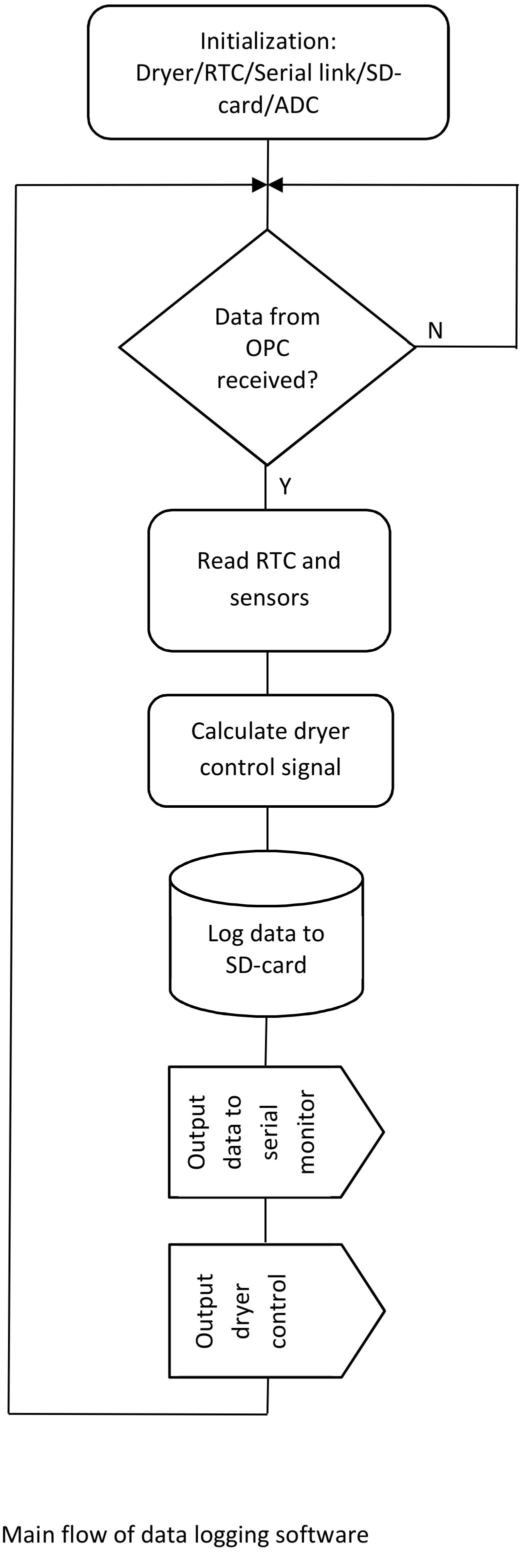


Fig. 8: Main flow of data logging software

PM SENSOR SYSTEM

The software running on the OPC-R1 is specific to the sensor and was provided by the manufacturer. It was just extended slightly to transmit a subset of the data gathered from the sensor via the asynchronous serial link using the Easy Transfer library /11/. The data structure for the data transfer is defined by a struct-element:

```
SoftEasyTransfer ET;
struct SEND_DATA_STRUCTURE{
  unsigned int hist[16];
  float temp;
  float humid;
  float PM[3];
};
SEND_DATA_STRUCTURE mydata;
```

The data containers of this structure are filled with the void PrintData() method with statements like “mydata.hist[k] = *pUInt16” that transfers the 16bit integer values of the histogram as an example. The actual sending of the data is accomplished with the “ET.sendData()” method in the main loop of the program.

The actual sensor data are simply transferred from the variables used in the manufacturer’s example code as for example in:

```
pUInt16 = (unsigned int *)&SPI_in[40];
mydata.temp = ConvSTtoTemperature(*pUInt16);
```

In this code snippet, the first line extracts an integer from what was received on the SPI bus interface. The second line converts the data with a method given by the manufacturer and transfers the result to the Easy Transfer SEND_DATA_STRUCTURE. This methodology is used for all data used from the OPC-R1 such as OPC-temperature, OPC-humidity, histogram and PM data.

NO_2 SENSOR SYSTEM

The NO_2 sensor system makes use of the ADS1115 AD-converter and is controlled by the data logging Arduino. The ADS1115 converter has a programmable gain amplifier that is set to maximum gain (0.0078125mV per LSB) with a call to the “ads.setGain(GAIN_SIXTEEN)” method during initialization. For reading the data, a differential connection is established to suppress noise on the ground line using the “ads.readADC_Differential_0_1()” method to read the working electrode signal on channel 0 and 1 and “ads.readADC_Differential_2_3()” to read the auxiliary electrode signal on channel 2 and 3. The analog signals are read directly from the respective ISB outputs for the working and auxiliary electrode.

DRYER REGULATION

The software implements a simple algorithm to regulate the temperature of the air sucked in by the OPC-R1 through the low-cost dryer system. For this purpose, the temperature sensed by the OPC internal temperature sensor is evaluated. However, the dryer heating is only activated when the humidity seen by the external HYT221 temperature/humidity sensor is exceeding the threshold value of 70% relative humidity to avoid overheating particles and to save electrical power.

The regulation algorithm is implemented with the following code segment:

```
//dryer control
if (hum > 70) {
  if (mydata.temp < dryerMinT) {
    dryerOn = true;
    overtemp = false;
  }
  else {
    if (mydata.temp > dryerMaxT) {
      dryerOn = false;
      overtemp = true;
    }
    else{
      if (overtemp) {
        dryerOn = false;
      }
      else {
        dryerOn = true;
      }
    }
  }
}
else {
  dryerOn = false;
}
```

Two state variables are used to describe the dryer status: dryerOn and overtemp. The main if-clause evaluates the humidity data of the external humidity sensor. When the humidity is > 70%, a further evaluation of the OPC-temperature data is done. If this is not the case, the dryer will be kept off by setting the dryerOn status signal to false. In case of high humidity, the regulation of the OPC inlet air temperature is activated. This regulation scheme can be perceived as a two-point control that keeps the air temperature sensed by the OPC between dryerMinT and dryerMaxT. When dryerMaxT is reached after turning the heating on, the overtemp status is set and the dryer is turned off. When the dryer is turned off in the overtemp state, the tube starts cooling down with its thermal time constant. As soon as the dryerMinT is reached, the overtemp status is reset and heating is turned on again. During the field experiment the temperature thresholds were set to dryerMinT = 34°C and dryerMaxT = 35°C.

Provided the high humidity condition lasts for a certain period, the inlet air temperature oscillations between dryerMinT and dryerMaxT show a periodicity that depends on the thermal capacity of the tube and the thermal leakage of the thermal isolation. For the first prototype the periodicity ranged between 30 and 60min, therefore the difference between dryerMaxT and dryerMinT was selected as small as 1°C to keep the amplitude of the oscillations small enough.

NO_2 SENSOR HUMIDITY REGULATION

Since electrochemical gas sensors suffer from cross-sensitivity to temperature and humidity, the initial strategy was to also record temperature and humidity and to use this data for compensation by postprocessing. However, it quickly turned out, that the regulation of the inlet air temperature of the OPC-R1 also has a very positive regulation effect on the relative humidity seen by the NO_2 sensor.

The reason is, that the humidity limit for the dryer regulation becoming active is sensed by the external temperature/humidity sensor placed in close vicinity to the NO_2 sensor. Consequently, during high humidity phases the relative humidity sensed near the NO_2-sensor gets stabilized to the turn-on threshold of the dryer. When the dryer is activated, the heating warms up the interior and the walls of the box and thus reduces the humidity under the rain protection roof that covers the NO_2-sensor as well as the external temperature/humidity sensor. The effect can clearly be observed in fig. 9, when comparing the humidity of the HYT221 sensor placed adjacent to the NO_2 sensor in the wall of the SMbox measurement unit. The humidity was also measured by the Grimm reference instrument that is equipped with an external meteorological sensor.

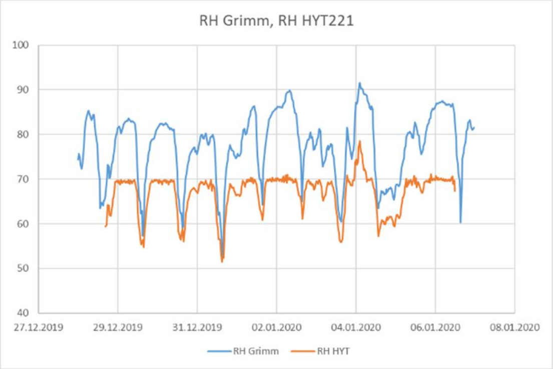


Fig. 9: Relative humidity recorded by the external sensor of the measurement unit and the Grimm reference instrument for PM

The data in fig. 9 from the field experiment show the typical daily fluctuations of a cold but sunny winter day. During night, the ground cools down by thermal radiation and the relative humidity increases to high values. Due to the low position of the sun at the horizon, the ground can’t store much thermal energy during daytime that would prevent the temperatures to drop at night. Therefore, the blue trace of the Grimm sensor in the graphs just shows short phases of low relative humidity during the afternoon and a strong increase during nighttime that exceeds the 70% threshold each day and reaches maxima of almost 90%. The orange trace of the HYT 221 sensor in the SMbox measurement unit, however, appears to be clamped to a maximum of 70%, which is the turn-on threshold of the PM dryer.

Nevertheless, when humidity gets very high, the thermal coupling is too weak to maintain the humidity seen by the sensor to the threshold value. But even under this condition, the dryer heating still has a beneficial effect.

Calibration and Data Postprocessing

In many cases, the idea behind low-cost or lower-cost sensors is to compensate the initially poor measurement accuracy by smart software. In contrast, achieving high initial accuracy would require a costly production precision and control. When compensation algorithms can be implemented in software, the development of the code may also be costly but when the same software is used for many sensors, the development cost is divided up among the quantity of the sensors manufactured.

Lower-cost PM sensors as well as electrochemical gas sensors show a high variation in their sensitivity with respect to professional equipment. Therefore, a reasonable approach is to improve accuracy by a calibration to reference equipment. Due to the fact, that the measurement signal is a mixture between a deterministic signal generated by the desired working principle and an unwanted but inevitable random noise, a statistical method such as a regression calculation is favorable for compensation of sensitivity variations. The assumption with a standard linear regression model is a linear relationship between data measured with the sensor and the data measured with the reference instrument. For the OPC-R1 used as PM sensor in the measurement unit, this assumption turned out to be a feasible method to compensate the sensitivity variations.

However, since the sample volume of the OPC-R1 used for counting the particles is small, a significant averaging across many sample intervals is required to reduce the statistical noise. The statistical noise is the larger, the larger the particles are, since typically the number count significantly drops for larger particles. Therefore, an averaging of about 800 sample sets equivalent to a measurement interval of 30min was applied to evaluate the field experiments. Also, for the NO_2 measurements an averaged measurement interval of 30min turned out to be reasonable. This additionally helps to decimate the amount of data required for further post-processing and graphical representation.

Even though it would have been beneficial to do the averaging already in the hardware, the strategy to sample and store at high rates without averaging gives more flexibility in optimizing the evaluations during post-processing. Consequently, we decided not to do any postprocessing inside the measurement.

For the PM sensor, the major cross-sensitivity to the humidity was substantially reduced by the additional dryer system. For the NO_2 sensor, we indeed achieved a mitigation of influence due to the indirect thermal regulation. Nevertheless, the cross-sensitivity to temperature was still visible and the cross-sensitivity to humidity was just mitigated and not completely removed.

Therefore, we decided to implement a field calibration strategy based on a multi-variate regression including the humidity and temperature signals measured by the external temperature/humidity sensor (HYT221).

From various experiments in the lab and from reviewing the literature, it became obvious that temperature and humidity variations show significant influencing factors on the raw working electrode signal (WE) of the gas sensor. Whereas the auxiliary electrode (AE) not exposed to gas can be used to a certain extend to compensate temperature influences, humidity influences can hardly be compensated, since the AE is not exposed to humidity.

Therefore, during a field calibration phase, the sensor was intentionally exposed not only to NO_2 concentration changes but also to significant temperature and humidity changes that covered the targeted measurement range. Using a linear model of superimposed influences from WE, AE, temperature and humidity as independently measured parameters together with an offset to obtain the final NO_2 signal, a multi-variate regression was deployed to determine the respective influencing coefficients for each parameter. Applying these previously determined influencing coefficients in an inverse sense for compensation during the actual validation phase showed, that the coefficients determined during the calibration phase are stable enough over a time frame of at least a month to achieve enough compensation of the temperature and humidity influences.

FIELD VALIDATION RESULTS

The prototype measurement unit was operated in a field measurement experiment in the garden of a public school at Hauptstätter Straße 139 (B14), a road heavily loaded with traffic in downtown Stuttgart, Germany (N48.765457, E9.170234).

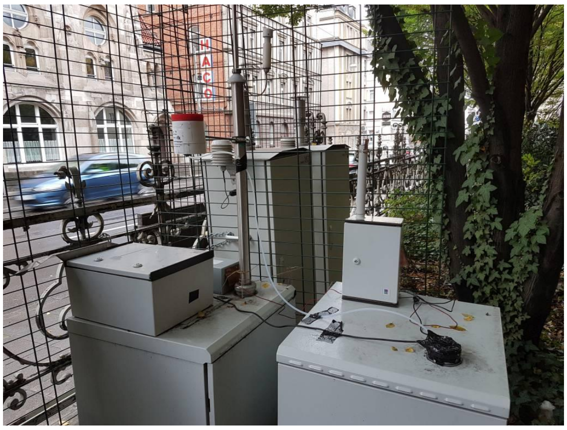


Fig. 10: The prototype unit placed on the right equipment container in the garden of Marienschule towards the roadside of Bundesstrasse B14

The field measurement was conducted from November 2019 to January 2020 in several phases from which two representative phases were selected: the validation phase ranging from 28.12.19 - 06.01.2020 (including Silvester) which is referenced to the phase ranging from 02.12.19-16.12.19 considered as calibration phase.

The SMbox measurement unit was placed nearby a Grimm EDM 180 dust monitor that acted as reference instrument for particulate matter (PM) and a 2B Technologies Model 405 nm $NO_2/NO/NO_x$ monitor as reference for NO_2 (see fig 10).

The basic idea of the field measurement was to run the SMbox measurement unit in at least two phases. One is considered as calibration phase and serves for extracting the compensation coefficients to achieve a best as possible matching between low-cost sensor and reference instrument using the regression method. The other phase, where the compensation coefficients are applied and the results are compared to the reference equipment, represents the actual validation phase.

Particularly during 31st of December and 2nd of January, the weather conditions were well suited to study the dryer performance. During these 3 days, a high-pressure area existed with inversion conditions above the land pan of Stuttgart. The boundary of the inversion was clearly visible with the camera installed at the TV tower 370m above the geographic height (260m) of Stuttgart downtown (fig. 11). This inversion layer kept the pollution concentration high until 3rd of January when the inversion collapsed.

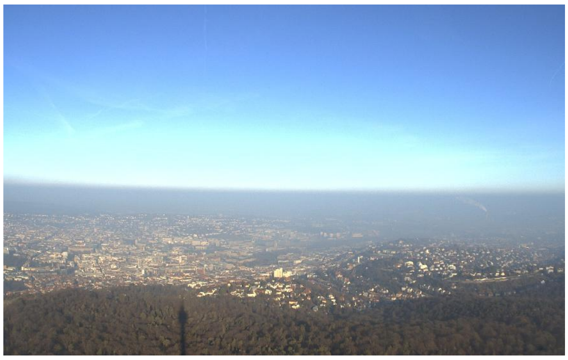


Fig. 11: Inversion conditions on January 1st, 2020 10:53h above Stuttgart

In the morning of 2nd of January fog evolved and increased the relative humidity up to 90%. However, the lower boundary of fog did not touch the ground level in downtown Stuttgart. Nevertheless, the high humidity caused significant hygroscopic growth of the particles and let many sensors of the OK Lab sensor network /14/ measure very high mass concentrations for PM10 exceeding 50ug/m^3 because their laser scattering sensors (SDS011, Nova Fitness, China) do not use a drying system.

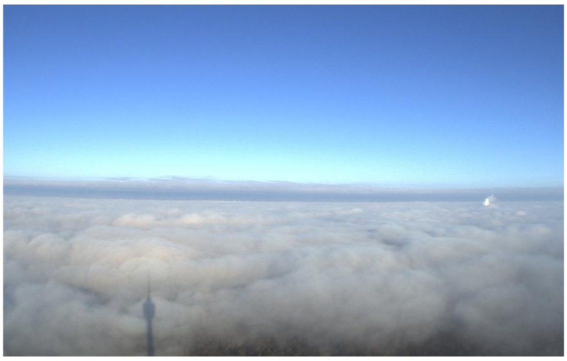


Fig. 12: Inversion conditions on January 2nd, 2020 10:35h above Stuttgart downtown

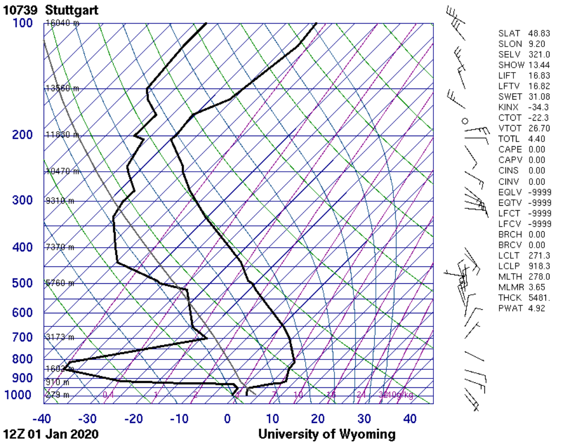
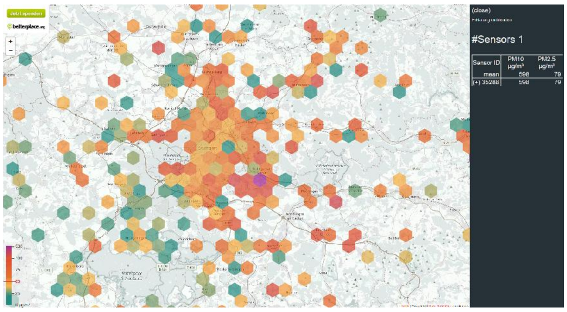


Fig. 13: Vertical profile of dew point (left) and temperature (right) on January 1st, 2020 clearly indicating the inversion



(close)

#Sensors 1

Sensor ID mean	PM10 µg/m³ 598	PM2.5 µg/m³ 79
(+) 35288	598	79

Fig. 14: Low-cost PM10 sensors of the OK Lab network without dryer showing far too high particle concentrations on January 2nd, 2020 due to the humidity influence

PM-Results

During the validation phase of 28.12.-06.01. the measurement unit shows a large concentration span of PM10, PM2.5 and PM1. Since all PM fractions were dominated by small particle sizes, we concentrated on PM2.5 for the analysis. PM2.5 was measured by the OPC-R1 lower than 1ug/m^3 on January 4th and higher than 500ug/m^3 on Silvester.

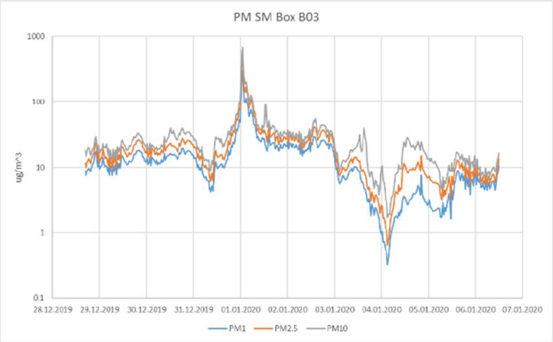


Fig. 15: Uncompensated PM results of the measurement unit, validation phase (28.12. - 06.01.)

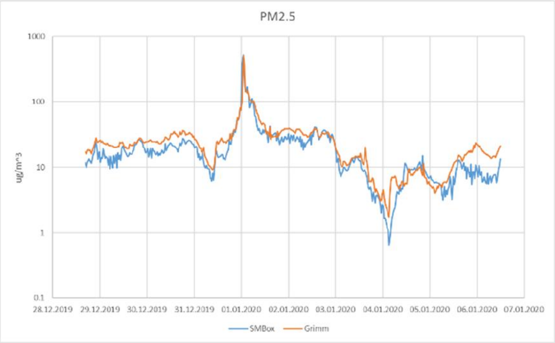


Fig. 16: Uncompensated PM2.5 results of the measurement unit, compared to the reference instrument (28.12. - 06.01.)

Fig. 15 shows that the particle spectrum is dominated by small particles.

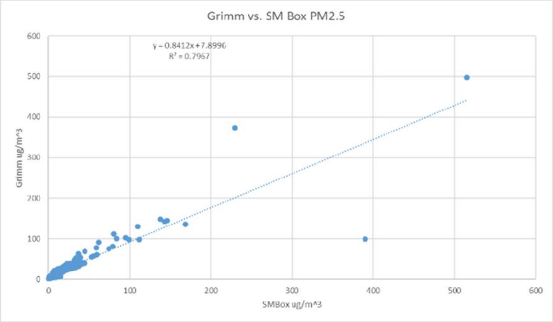


Fig. 17: Correlation between Grimm EDM data and OPC-R1 raw data

Even when not compensated, the data of the OPC-R1 match quite well the data of the Grimm EDM (see fig. 16). When a linear one-dimensional regression is applied to determine compensation coefficients and the Silvester event is excluded, the difference between SMbox data and reference instrument stays between +/-10ug/m^3 for the PM2.5 fraction. During Silvester, due to the high particle concentrations caused by the firework and the poor mixing of the air, the deviation gets much stronger. This may be also being caused due to coincidence errors in the low-cost sensors and due to lack of synchronism between the systems with respect to the highly local fluctuations of the particle concentrations.

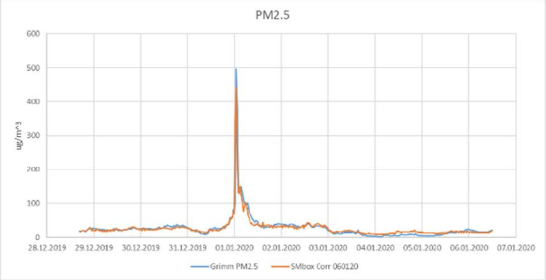


Fig. 18: result of the calibration of the measurement unit to the PM reference instrument for PM2.5 (linear scale) using calibration coefficients extracted from the validation phase of 28.12.20-06.01.20

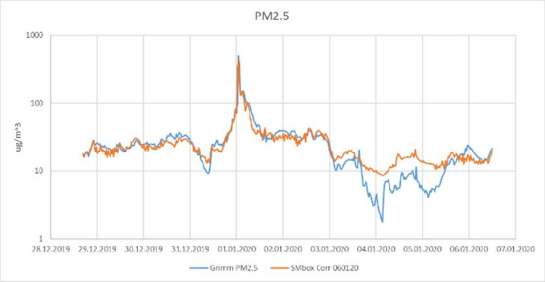


Fig. 19: result of the calibration of the measurement unit to the PM reference instrument for PM2.5 (log scale) using calibration coefficients extracted from the validation phase of 28.12.20-06.01.20

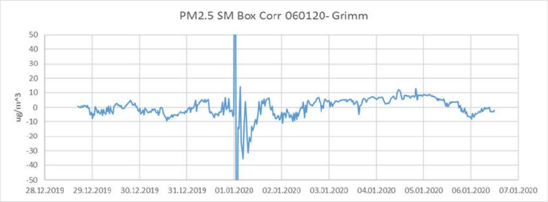


Fig. 20: Difference between the measurement unit and PM reference instrument (Grimm EDM) using calibration coefficients extracted from the validation phase of 28.12.20-06.01.20

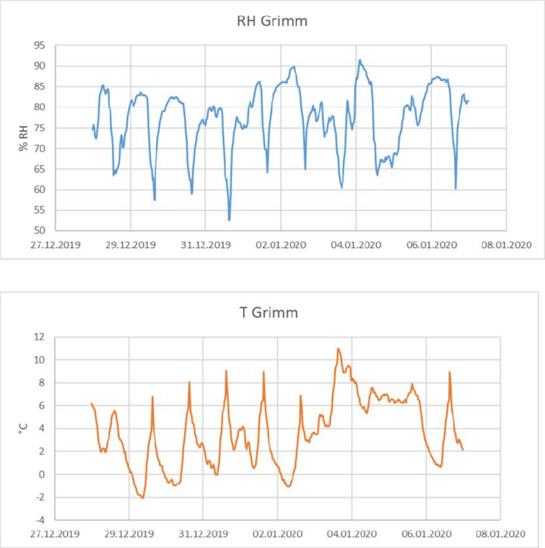


Fig. 21 a, b: Humidity and temperature during the validation phase of 28.12.20-06.01.20

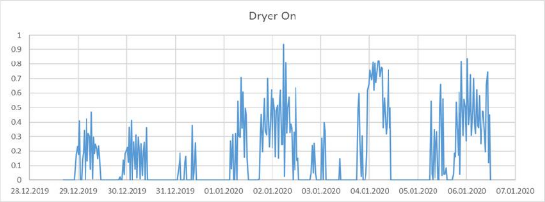


Fig. 22: The “dryer-on” control signal during the validation phase of 28.12.20-06.01.20

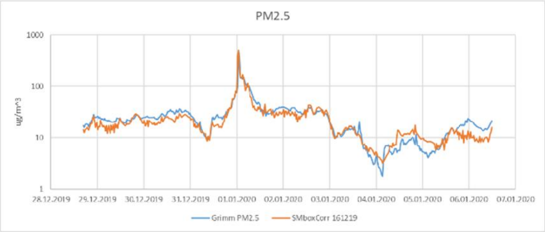


Fig. 23: Result of the calibration of the measurement unit to the PM reference instrument for PM2.5 (log scale) using calibration coefficients extracted from the calibration phase 02.12.19-16.12.19

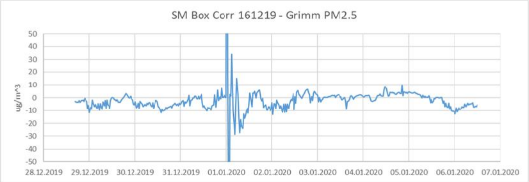


Fig. 24: Difference between the measurement unit and PM reference instrument (Grimm EDM) using calibration coefficients extracted from the calibration phase 02.12.19-16.12.19

Of major importance, however, is the fact that compensation coefficients extracted during the calibration phase 02.12.19-16.12.19 and finally applied to the validation phase 28.12.19-06.01.20, change the difference between the SMbox measurement unit and the reference instrument only negligibly. The extracted compensation coefficients in the calibration phase are 0.989 for the slope and 2.593ug/m^3 for the offset, whereas during the validation phase the slope is 0.841 and the offset is 7.9ug/m^3 (Grimm = SMbox*slope + offset). This means, a field calibration with a reference instrument preceding the actual validation phase would result in an acceptable measurement uncertainty when no reference is available anymore later.

In the above discussion, the data sampled by the SMbox and Grimm EDM at a rate of one sample per 6 seconds, was averaged to an interval of 30min. The assessment of the dryer effectiveness, however, requires a higher time resolution since the dryer is controlled by a fast oscillating pulse width modulation of the electrical energy supplied. Therefore, a dedicated evaluation with a 1-minute averaging time was additionally conducted.

The temperature and humidity sensor of the Grimm, installed at a separate sampling head in enough distance to the instrument, particularly recorded very high humidity values in the morning of January 1st around 9:00h (85% rh) and January 2nd around 9:00h (90% rh). During these phases the low-cost dryer was working with high duty cycle. In contrast, the low-cost dryer turned off around 15:00h on both days, since the humidity decreased to below 60% under the influence of the sun. When the data are analyzed at high temporal resolution the variation of the growth factor reduction under the influence of the oscillating thermal power of the dryer becomes visible as a fine structure of the PM2.5 sensor signal. The temperature inside the OPC-R1 monitoring the temperature of the incoming air shows a temperature variation between 15 and 28°C that is well correlated to the dryer-on signal sent to the PWM switch turning on and off the dryer. Consequently, we can assume that the effectiveness of the dryer is quite high.

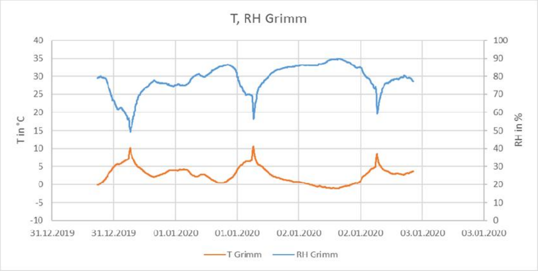


Fig. 25: Humidity and temperature during the validation phase of 28.12.20-06.01.20

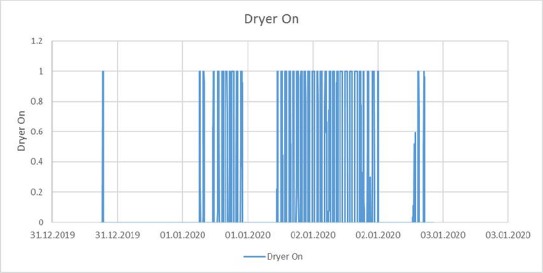


Fig. 26 a, b: The "dryer-on" control signal during the validation phase of 28.12.20-06.01.20 shown with 1min time resolution

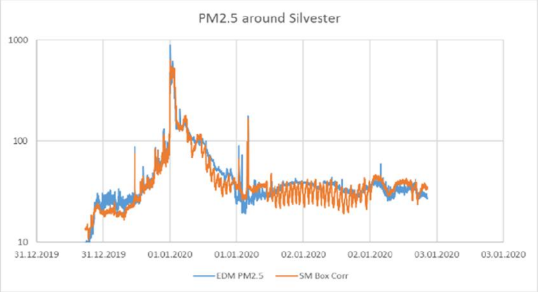


Fig. 27: result of the calibration of the measurement unit to the PM reference instrument for PM2.5 (log scale) displayed with 1min time resolution

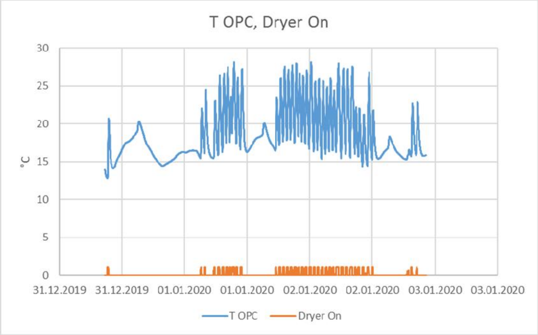


Fig. 28: The temperature of the sucked in air as monitored by the OPC-R1 internal sensor together with the "dryer-on" control signal shown with 1min time resolution

NO_2-Results

The Alphasense electrochemical NO_2 sensor also showed a good correlation to the 2B Technologies reference instrument. Initially, the same method of a linear, one dimensional correlation-based compensation technique was used as with the PM sensor. The NO_2 sensor system in the measurement unit outputs AD-converter samples in LSBs and therefore need a conversion to a ppb unit.

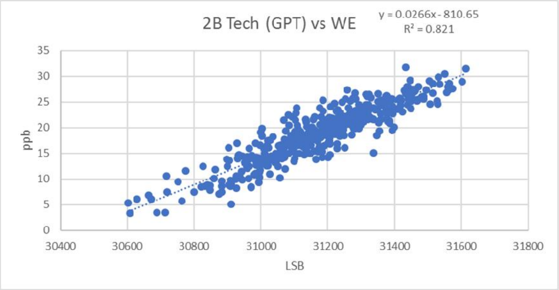


Fig. 29: Correlation between the working electrode signal of the NO_2 gas sensor and the 2B Technologies reference instrument during validation phase

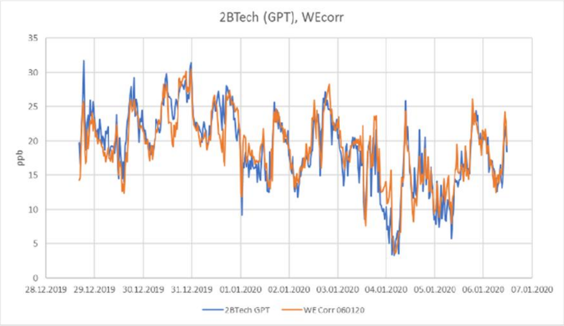


Fig. 30: NO_2 results of the measurement unit (working electrode) and the 2B Technologies NO_2 reference instrument after a standard regression-based calibration during validation phase

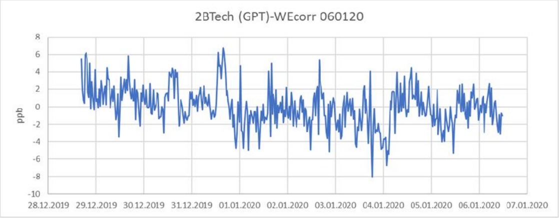


Fig. 31: Difference between the measurement unit and the 2B technologies NO_2 reference instrument after a standard regression-based calibration using calibration coefficients extracted from the validation phase of 28.12.20-06.01.20

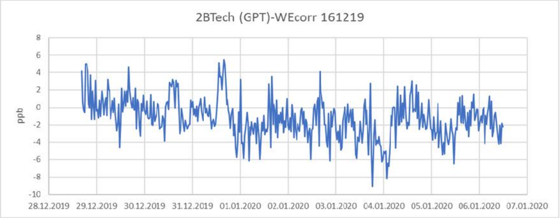


Fig. 32: Difference between the measurement unit and the 2B technologies NO_2 reference instrument after a standard regression-based calibration using calibration coefficients extracted from the calibration phase 02.12.19-16.12.19

When compensating the values of the validation phase from 28.12.-6.1. with coefficients extracted from the same phase, the maximum difference between the SMbox measurement unit and the reference instrument turns out to be 8ppb.

The compensation yields the following compensation coefficients:

Tab. 2 a, b: Compensation coefficients and results from a one-dimensional regression

Coefficients extracted from validation phase 28.12.-6.1. and applied to same validation phase		
Offset	-810.652	ppb
WE sensitivity	0.026607	ppb/LSB
Max. difference	8.005657	ppb

Coefficients extracted from calibration phase 2.12.-16.12. and applied to validation phase 28.12.-6.1.		
Offset	-781.9998	ppb
WE sensitivity	0.025727	ppb/LSB
Max. difference	9.068822	

When compensating the values of the validation phase from 28.12.19-6.1.20 with coefficients obtained from the calibration phase 2.12.19-16.12.19 then the maximum difference is 9ppb. This result shows the calibration strategy used for PM2.5 also works in general with the electrochemical NO_2 sensor.

The Alphasense NO_2 sensor is a 4-electrode sensor and uses a 4th electrode not exposed to gas, called auxiliary electrode, to allow compensation at least for the temperature influence and long-term sensor drift. Therefore, also a multi-variate regression including the humidity and temperature variation, or humidity and AE signal variation was checked for improvement of the matching.

Using the WE signal, humidity and temperature indeed resulted in a maximum of the difference of only 5.3ppb when the compensation coefficients were extracted from the same phase. When using AE instead of temperature, the maximum difference is 6ppb. When the coefficients were extracted from the calibration phase and are applied to the validation phase the maximum difference increases to 7.1ppb using the WE signal, humidity and temperature). As a conclusion, the use of a multi-variate regression including humidity and temperature data from the HYT221 sensor and optionally AE data from the 4 electrode sensor results in a notable improvement.

Tab. 3 a, b, c: Compensation coefficients and results from a multi-variate regression

Coefficients extracted from validation phase 28.12.-6.1. and applied to the same phase		
Offset	-741.686000	ppb
WE sensitivity	0.025037	ppb/LSB
RH influence	-0.278644	ppb/%RH
Temp influence	-0.288314	ppb/°C
Max. difference	5.3	ppb

Coefficients extracted from validation phase 28.12.-6.1. and applied to same phase		
Offset	-692.18500	ppb
WE sensitivity	0.02870	ppb/LSB
RH influence	-0.30063	ppb/%RH
AE influence	-0.00562	ppb/°C
Max. difference	6.0	ppb

Coefficients extracted from calibration phase 2.12.-16.12. and applied to validation phase 28.12.-6.1.		
Offset	-754.51700	ppb
WE sensitivity	0.02552	ppb/LSB
RH influence	-0.30456	ppb/%RH
Temp influence	-0.12772	ppb/°C
Max. difference	7.1	ppb

SUMMARY

As it can be seen from the field measurement results for particulate matter of type PM2.5, the maximum deviation of the measurement unit from reference instrumentation stays within +/-10ug/m^3 when we exclude the high pollution event caused by the new year firework during the field experiment. A simple linear, one-dimensional regression-based compensation for the sensor sensitivity differences is enough to achieve a good matching between the lower cost measurement unit and the reference instrument. The application of a low-cost thermal gas dryer with a simple two-point regulation scheme proved to be essential to effectively remove the hygroscopic growth effect induced by high humidity phases. In laboratory experiments the lower cost dryer system also proved to be highly efficient to even remove fog droplets of small diameter. Therefore, we are convinced the low-cost dryer is responsible for a substantial improvement of the overall PM measurement accuracy.

For NO_2, the maximum deviation of the measurement unit from reference instrumentation stays within +/-10ppb during the field experiment using a simple linear one-dimensional regression for compensating just the working electrode sensitivity of the gas sensor. A compensation of humidity and temperature influences by postprocessing the data based on a multi-variate regression showed substantial improvement. In both cases, it was possible to derive the calibration coefficients from a calibration phase and to deploy them later to a validation phase. For NO_2 it was possible to further minimize the maximum deviation by using the multi-variate regression that includes the temperature and humidity influences to +/-7.1ppb.

With these results, we believe that our lower-cost measurement unit is a suitable solution for many cost-sensitive outdoor applications where many spatially distributed measurements are executed in parallel and cloud-based data-logging using wireless networks is not available or data-safety issues prevent the use of a network based data storage.

LITERATURE

/1/ Environmental Changes: Temperature, Pressure, Humidity; Alphasense Application Note Issue 12; Alphasense Ltd.

/2/ Piotr Czernicki and Mathias Kallmert; Evaluation of a heated inlet to reduce humidity induced error in low-cost particulate matter sensors; Department of Design Sciences Faculty of Engineering LTH, Lund University;2019

/3/ Michel Gerboles, Laurent Spinelle, Manuel Aleixandre; OPEN SESSION COST on New Sensing Technologies for Air Quality Monitoring Brescia, Italy; September 2014

/4/ Michel Gerboles, Laurent Spinelle, Marco Signorini; AirSensEUR: an open data/software/hardware multi-sensor platform for air quality monitoring. Part A: sensor shield;JRC Technical Report; 2015

/5/ Christian Kjær Jensen: Assessing the applicability of low-cost electrochemical gas sensors for urban air quality monitoring. Master’s thesis. January 2016.

/6/ Karagulian, F., Gerboles, M., Barbiere, M., Kotsev, A., Lagler, F., Borowiak, A.; Review of sensors for air quality monitoring; JRC Technical Report; 2019

/7/ Anumita Roychowdhury, Vivek Chattopadhyaya and Shambhavi Shukla; Reinventing Air Quality Monitoring - Potential of low-cost alternative monitoring methods; Centre for Science and Environment; New Delhi

/8/ Matthias Budde et al.; Potential and Limitations of the Low-Cost SDS011 Particle Sensor for Monitoring Urban Air Quality; 3rd International Conference on Atmospheric Dust; DUST 2018

/9/ Ron Williams, Amanda Kaufman, Tim Hanley, Joann Rice, Sam Garvey; Evaluation of Field-deployed Low Cost PM Sensors; Evaluation of Field-deployed Low Cost PM Sensors; United States Environmental Protection Agency; EPA/600/R-14/464 | December 2014

/10/ Rohan Jayaratne, Xiaoting Liu, Phong Thai, Matthew Dunbabin, Lidia Morawska; The Influence of Humidity on the Performance of Low-Cost Air Particle Mass Sensors and the Effect of Atmospheric Fog; Atmos. Meas. Tech. Discuss.; April 2018

/11/ Bill Porter; EasyTransfer Arduino Library v2.1; http://www.billporter.info/easytransfer-arduino-library

/12/ Aboubakr Benabbas et al.; Measure particulate matter by yourself: data-quality monitoring in a citizen science project; J. Sens. Sens. Syst., 8, 317–328, 2019

/13/ Matthias Budde et al.; SmartAQnet: remote and in-situ sensing of urban air quality; SPIE Remote Sensing, 2017, Warsaw, Poland

/14/ Luftdaten Selber Messen; OK-Lab Stuttgart https://luftdaten.info

/15/ Philipp Schneider et al.; Toward a Unified Terminology of Processing Levels for Low-Cost Air-Quality Sensors; Environ. Sci. Technol. 2019, 53, 8485−8487

APPENDIX A: CIRCUIT DIAGRAMS

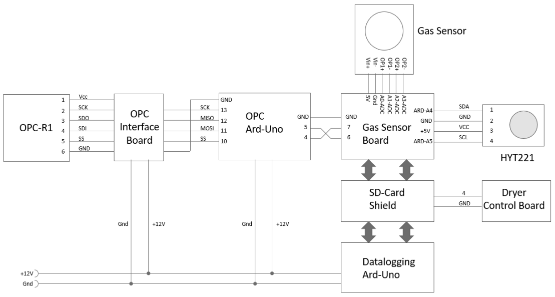


Fig. 33: Circuit diagram of the main measurement unit

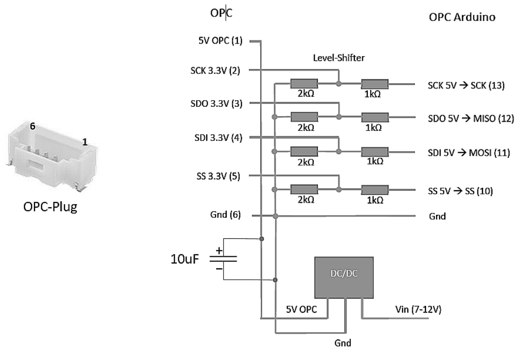


Fig. 34: Circuit diagram of the OPC interface board

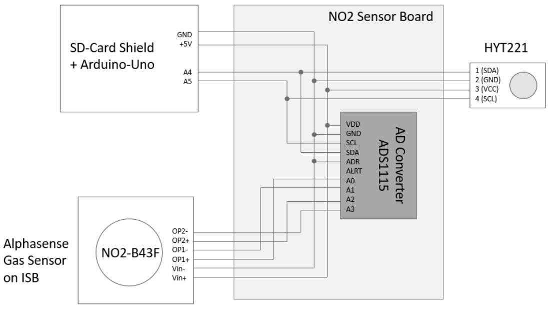


Fig. 35: Circuit diagram of the gas sensor board containing the AD-converter board

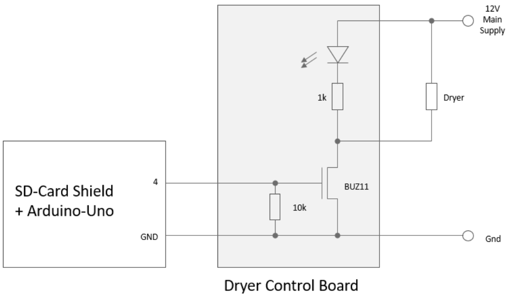


Fig. 36: Circuit diagram of the dryer control board

Appendix B: code

OPC Arduino

(OPC-R1_BL4.ino)

Copyrighted code from sensor manufacturer with some modifications, please contact Alphasense Ltd

Data Logging Arduino

(SmBoxOutdoor_DryerV3.ino)

```
//Alphasense Gas differenial
//HYT221
//OPC-R1
//#define dryerMinT 37 (V1) changed to 34
//#define dryerMaxT 38 (V1) changed to 35
//fixed pinMode dryerPin

#include <Wire.h>
#include <SoftEasyTransfer.h>
#include <SPI.h>
#include <SD.h>
#include <SoftwareSerial.h>
#include <Adafruit_ADS1015.h>
#include "RTClib.h"
#define HYT939_ADDR 0x28
#define TFACTOR 99.2909
#define TDELTA 40.0
#define HFACTOR 163.83
#define dryerPin 4
#define dryerMinT 34
#define dryerMaxT 35

Adafruit_ADS1115 ads(0x48);


RTC_PCF8523 rtc;
//RTC_DS1307 rtc;
DateTime now1;

char fileName[15] = "datalog.txt";
File myFile;

SoftwareSerial opcSerial(6, 7); //OPC rx, tx

//create object
SoftEasyTransfer ETopc;

struct RECEIVE_DATA_STRUCTURE {
  unsigned int hist[16];
  float temp;
  float humid;
  float PM[3];
};
RECEIVE_DATA_STRUCTURE mydata;

bool dryerOn = false;
bool overtemp = false;

void setup(){
  pinMode(dryerPin, OUTPUT);
  Wire.begin();
  pinMode(10, OUTPUT); // SD Card CS

  if (! rtc.initialized()) {  //PCF8523
  //if (! rtc.isrunning()) {    //DS1307
    Serial.println("RTC is NOT running!");
    // following line sets the RTC to the date
& time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__),
F(__TIME__)));
    // This line sets the RTC with an explicit
date & time, for example to set
    // January 21, 2014 at 3am you would call:
    //rtc.adjust(DateTime(2017, 1, 21, 3, 0,
0));
  }
  //rtc.adjust(DateTime(F(__DATE__),
F(__TIME__)));

  opcSerial.begin(9600);
  Serial.begin(9600);
  //start the library, pass in the data
details and the name of the serial port.
  ETopc.begin(details(mydata), &opcSerial);

  if (!SD.begin(10)) {
    Serial.println("SDcard not ready");
    delay(10000);
  }
  else
    Serial.println("SDcard ok");

  if (!SD.exists(fileName)) {
    myFile = SD.open(fileName, FILE_WRITE);
    myFile.println("###");
    myFile.flush();
  }
  else {
    myFile = SD.open(fileName, FILE_WRITE);
    myFile.println("-----");
    myFile.flush();
  }

  // setup adc
  //Serial.println("Getting      single-ended
readings from AIN0");
  //Serial.println("ADC Range: +/- 6.144V (1
bit = 3mV/ADS1015, 0.1875mV/ADS1115)");

  // The ADC input range (or gain) can be
changed via the following
  // functions, but be careful never to exceed
VDD +0.3V max, or to
  // exceed the upper and lower limits if you
adjust the input range!
  // Setting these values incorrectly may
destroy your ADC!
  // ADS1015  ADS1115  -------  -------
  //  ads.setGain(GAIN_TWOTHIRDS);    //  2/3x
gain +/- 6.144V  1 bit = 3mV       0.1875mV
(default)
  // ads.setGain(GAIN_ONE);         // 1x gain
+/- 4.096V  1 bit = 2mV      0.125mV
  // ads.setGain(GAIN_TWO);         // 2x gain
+/- 2.048V  1 bit = 1mV      0.0625mV
  // ads.setGain(GAIN_FOUR);        // 4x gain
+/- 1.024V  1 bit = 0.5mV    0.03125mV
  // ads.setGain(GAIN_EIGHT);       // 8x gain
+/- 0.512V  1 bit = 0.25mV   0.015625mV
```

```
  // ads.setGain(GAIN_SIXTEEN);   // 16x gain
+/- 0.256V  1 bit = 0.125mV  0.0078125mV
  ads.setGain(GAIN_SIXTEEN);      //  16x  gain
+/- 0.256V  1 bit = 0.125mV  0.0078125mV
  ads.begin();

  Serial.println("Setup       done,       now
logging...");
}

void loop(){

    if(ETopc.receiveData()){   //opc received
      now1 = rtc.now();


      Serial.print(now1.day());
      Serial.print('.');
      Serial.print(now1.month());
      Serial.print('.');
      Serial.print(now1.year());
      Serial.print(' ');
      Serial.print(now1.hour());
      Serial.print(':');
      Serial.print(now1.minute());
      Serial.print(':');
      Serial.print(now1.second());
      Serial.print(' ');


      Serial.print(';');
      for (int i = 0; i<16; i++) {
        Serial.print(mydata.hist[i]);
        Serial.print(' ');
      }
      Serial.print(';');
      Serial.print(mydata.temp);
      Serial.print(' ');
      Serial.print(mydata.humid);
      Serial.print(';');
      for (int i = 0; i<3; i++) {
        Serial.print(mydata.PM[i]);
        Serial.print(' ');
      }
      Serial.print(';');

      //myFile.print(timeStr);
      // print time without String

      DateTime now = rtc.now();
      myFile.print(now.day(), DEC);
      myFile.print('.');
      myFile.print(now.month(), DEC);
      myFile.print('.');
      myFile.print(now.year(), DEC);
      myFile.print(' ');
      myFile.print(now.hour(), DEC);
      myFile.print(':');
      myFile.print(now.minute(), DEC);
      myFile.print(':');
      myFile.print(now.second(), DEC);





      myFile.print('\t');
      for (int i = 0; i<16; i++) {
        myFile.print(mydata.hist[i]);
        myFile.print('\t');
      }
```

```
      myFile.print(mydata.temp);
      myFile.print('\t');
      myFile.print(mydata.humid);
      myFile.print('\t');
      for (int i = 0; i<3; i++) {
        myFile.print(mydata.PM[i]);
        myFile.print('\t');
      }

      //HYT
      unsigned int traw;
      unsigned int hraw;

      double temp;
      double hum;
      int i;
      unsigned char buffer[4];

      Wire.beginTransmission(HYT939_ADDR); //
transmit to device #44 (0x2c)
      Wire.endTransmission();     //     stop
transmitting
      //100ms warten
      delay(100);
      //4 Bytes vom Sensor lesen
      Wire.requestFrom(HYT939_ADDR, 4,true);
      i=0;
      while(Wire.available()) {
        char  c  =  Wire.read();  //  receive  a
byte as character
        buffer[i]=c;
        i++;
      }
      //Rohdaten aus Puffer lesen
      traw=buffer[2]*256+buffer[3];
      hraw=buffer[0]*256+buffer[1];
      //Daten laut Datenblatt maskieren
      traw&=0xfffc;
      hraw&=0x3fff;
      traw=traw/4;
      //Rohdaten Umrechnen
      temp=(double)traw/TFACTOR;
      temp=temp-TDELTA;
      hum=(double)hraw/HFACTOR;

      Serial.print("T:");
      Serial.print(temp);
      Serial.print(' ');
      Serial.print("H:");
      Serial.print(hum);
      Serial.print(' ');

      //read ADC
      int16_t adc_0, adc_1, adc_2, adc_3;

      adc_0 = ads.readADC_Differential_0_1();
      adc_1 = 0;
      adc_2 = ads.readADC_Differential_2_3();
      adc_3 = 0;
      //adc_0 = ads.readADC_SingleEnded(0);
      //adc_1 = ads.readADC_SingleEnded(1);
      //adc_2 = ads.readADC_SingleEnded(2);
      //adc_3 = ads.readADC_SingleEnded(3);
      Serial.print("Ch0: ");
      Serial.print(adc_0);
      //Serial.print(" Ch1:");
      //Serial.print(adc_1);
      Serial.print(" Ch2:");
      Serial.print(adc_2);
      Serial.print(";");
      //Serial.print(" Ch3:");
      //Serial.println(adc_3);
      //myFile.print('\t');
      myFile.print(adc_0);
      myFile.print('\t');
      //myFile.print(adc_1);
      //myFile.print('\t');
      myFile.print(adc_2);
      myFile.print('\t');
      //myFile.print(adc_3);
      //myFile.print('\t');

      myFile.print(temp);
      myFile.print('\t');
      myFile.print(hum);
      myFile.print('\t');

      //dryer control
      if (hum > 70) {
        if (mydata.temp < dryerMinT) {
          dryerOn = true;
          overtemp = false;
        }
        else {
          if (mydata.temp > dryerMaxT) {
            dryerOn = false;
            overtemp = true;
          }
          else  {  //  between  dryerMinT  and
dryerMaxT
            if (overtemp) {
              dryerOn = false; //let it cool
below dryerMinT
            }
            else {
              dryerOn = true; //let it reach
dryerMaxT
            }
          }
        }
      }
      else {
        dryerOn = false;
      }

      if (dryerOn) {
          digitalWrite(dryerPin,HIGH);
          Serial.println('1');
          myFile.print('1');
      }
      else {
          digitalWrite(dryerPin,LOW);
          Serial.println('0');
          myFile.print('0');
      }
      myFile.println();
      myFile.flush();

      delay(2000);
    }
}
```