European Commission

JRC TECHNICAL REPORTS

AirSensEUR: an open data/software /hardware multi-sensor platform for air quality monitoring.

Part B: Host, influx datapush and assembling of AirSensEUR

Michel Gerboles, Laurent Spinelle, Marco Signorini and Alexander Kotsev

2016

EUR 28054 EN

Joint Research Centre

This publication is a Technical report by the Joint Research Centre, the European Commission's in-house science service. It aims to provide evidence-based scientific support to the European policy-making process. The scientific output expressed does not imply a policy position of the European Commission. Neither the European Commission nor any person acting on behalf of the Commission is responsible for the use which might be made of this publication.

JRC Science Hub
https://ec.europa.eu/jrc

JRC102703

EUR 28054 EN

ISBN 978-92-79-60980-0 (PDF)

ISSN 1831-9424 (online)

doi:10.2790/214743 (online)

How to cite: Michel Gerboles, Laurent Spinelle, Marco Signorini and Alexander Kotsev; AirSensEUR: an open data/software /hardware multi-sensor platform for air quality monitoring - Part B: Host, influx datapush and assembling of AirSensEUR; EUR 28054 EN; doi:10.2790/214743

Table of contents

Foreword

This report belongs to a series of technical reports about the setup of the AirSensEUR platforms for air quality monitoring.

The A report presented the sensor shield design, software, configuration and evaluation of set of sensors.

The current B report presents the AirSensEUR host, the whole platform assembly with sensor box plus the InFlux dataPush. The SOS-T datapush consistent with the INSPIRE Directive and the correction algorithms of sensor data will be presented in the subsequent AirSensEUR EUR reports.

Abstract

AirSensEUR is an open source platform developed by the Joint Research Centre and LiberaIntentio, a SME specialized in Internet of Things, for monitoring ambient air quality by specialized laboratories, public bodies or citizens using low-cost sensors. All development aspects owned by JRC about AirSensEUR are made freely available through the use of public licenses.

AirSensEUR has the capacity to behave as a node within a network of multi sensors assuring interoperability and compliance with the INSPIRE Directive. It consists of a sensor shield that holds the sensors for gaseous and ancillary measurements and a host (electronic board) that integrates functionalities shared by sensor shields including datapush, GPS positioning, CPU and programing languages.

The first AirSensEUR sensor shield was presented in a previous report of the series of AirSensEUR reports. This report presents the host development, the AirSensor assembly with sensor boxes and InFlux dataPush.

1 Introduction

The use of low-cost sensors by laboratories in charge of ambient air monitoring for regulatory purposes or by citizens is quickly developing. At the same time, low-cost sensors raise strategic questions including the assessment of the quality of the information provided by those devices, the localization and identification of the micro-environments being sampled and the automatic reporting of this information so that it is available for all stakeholders using near to real time web services.

Although several sensor systems have appeared in the last years, generally the developers do not have the capacity to tackle the whole set of open questions regarding sensor systems. Currently air pollution using low cost sensors involves fast growing companies and/or public institutes specialized in digital technologies. Unfortunately, a great deal of effort is wasted by repeating the same research by several companies/institutes or within research projects. Moreover when a solution for one of the problem listed above is found, it is generally subjected to the control of the intellectual property rights.

In order to facilitate the use of sensors by diminishing the operational and development cost, the Joint Research Centre (JRC) is developing a multi-sensor platform, called AirSensEUR, for the monitoring of air pollution at low concentration levels. All development aspects owned by JRC about AirSensEUR are made freely available through the use of public licenses. AirSensEUR is developed as an open software/open hardware object that has the capacity to behave as a node within a network of multi sensors assuring interoperability and compliance with the INSPIRE Directive [1]. We strongly believe in such a common and participative plan for the establishment of a scientific community both of interested users and/or developers joining forces to build a validated sensor system.

The JRC objective is to build both the AirSensEUR host platform able to simultaneously control several sensor shields and to develop the first sensor shield for air pollution sensors to be connected to the host.

The host integrates functionalities shared by sensor shields including GPS positioning, Linux operating system and programming languages to control a list of I/O ports used by the shields (COM, USB, SPI, I2C, PWM …) with sufficient RAM, CPU computing capacity and a micro-SD storage up to 64Gb. Twined with the host platform we designed the 1st AirSensEUR sensor shield for air pollution monitoring based on electrochemical sensors. Figure 1 gives the flow chart of the AirSensEUR platform with the whole set of functionalities that is planned to develop.

The development of the sensor shield (block A1) is now complete (version 2.2). The 1st full prototype of the host including sensors, shield and host with data transmission of the web client has been tested in field in Dec. 2015 and Jan. 2016. Version 2 of the host (likely final version) with a few electronic and software changes is now tested at field sites. The shield handling box has been designed and we are finalising the host box that should be available over the next week. The CAD files are distributed under public license.

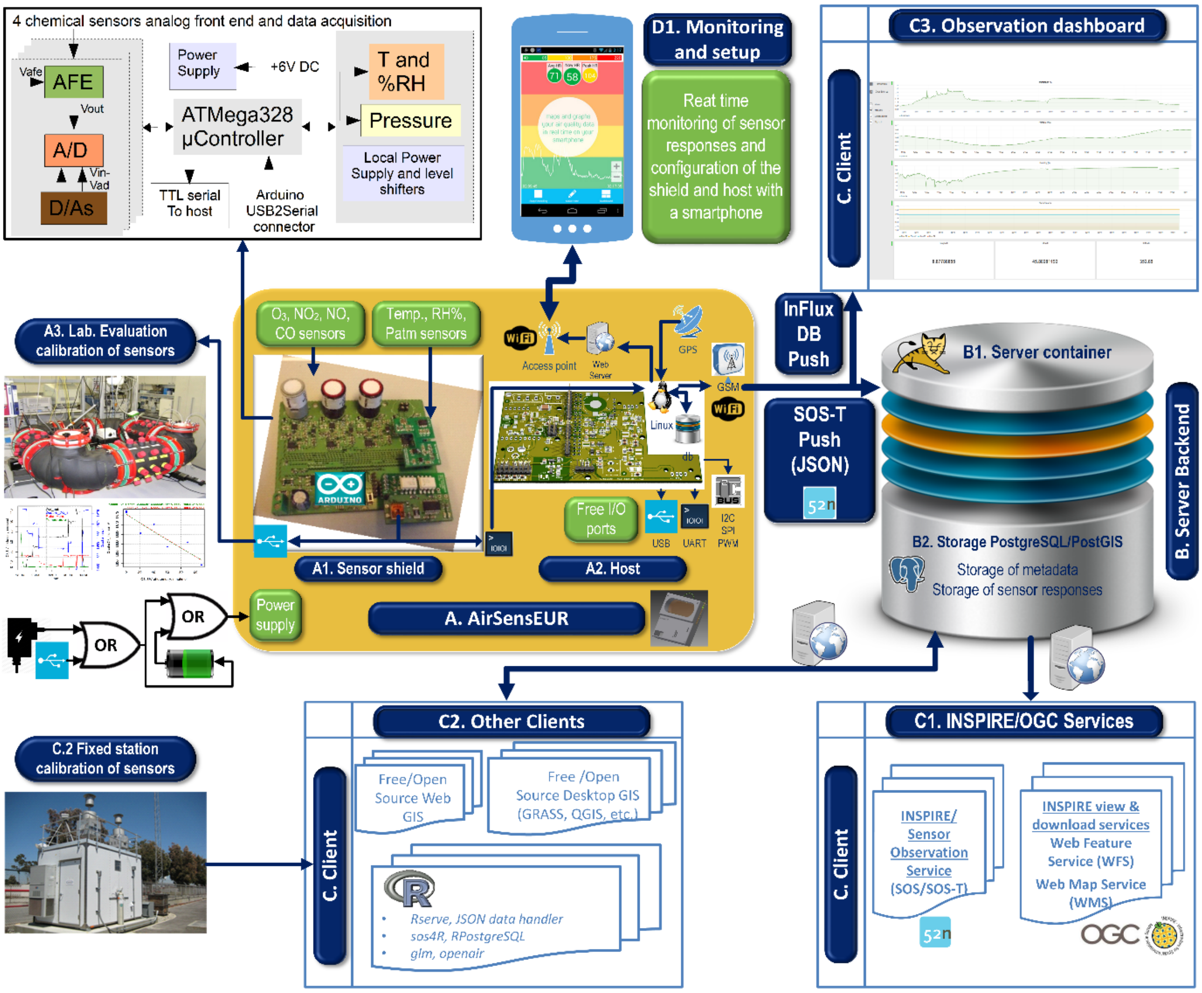

4 chemical sensors analog front end and data acquisition
Vafe
AFE
Vout
A/D
Vin-Vad
D/As
Power Supply
+6V DC
ATMega328 µController
TTL serial To host
Arduino USB2Serial connector
T and %RH
Pressure
Local Power Supply and level shifters
D1. Monitoring and setup
Real time monitoring of sensor responses and configuration of the shield and host with a smartphone
C3. Observation dashboard
C. Client
InFlux DB Push
SOS-T Push (JSON)
B1. Server container
B2. Storage PostgreSQL/PostGIS
Storage of metadata
Storage of sensor responses
B. Server Backend
A3. Lab. Evaluation calibration of sensors
O3, NO2, NO, CO sensors
Temp., RH%, Patm sensors
Access point
Web Server
GPS
GSM
Linux
db
Free I/O ports
USB
UART
I2C SPI PWM
A1. Sensor shield
A2. Host
Power supply
A. AirSensEUR
OR
OR
C.2 Fixed station calibration of sensors
C2. Other Clients
Free/Open Source Web GIS
Free /Open Source Desktop GIS (GRASS, QGIS, etc.)
Rserve, JSON data handler
sos4R, RPostgreSQL
glm, openair
C1. INSPIRE/OGC Services
INSPIRE/ Sensor Observation Service (SOS/SOS-T)
INSPIRE view & download services
Web Feature Service (WFS)
Web Map Service (WMS)
OGC

Figure 1: Schematic diagram of the AirSensEUR platform

The data transfer to a server backend (blocks B1 and B2 and C3) has been developed in compliance with the requirements of the INSPIRE Directive [1] using the Sensor Observation Services (SOS) interface standard [2] and it is in validation stage. The remote transfer functionalities of AirSensEUR will be published in the host report together with a cook book for the configuration of the backend server.

The architecture of AirSensEUR based on well-established free and open source software, also includes an INSPIRE view (through a Web Map Service - WMS), download services (block C1) and an Influx datapush with a graphana web observation dashboard (block C3) that gives the possibility to visualize time series plots of data.

The block C2 in Figure 1 is about the development of algorithms for the correction of sensor measurements based on the reference measurements of automatic stations or using the results of laboratory experiments in exposure chamber (block A3). Other possibility as the chemical filtering of interference and on-board calibration possibilities will be studied. The final solution of this module is expected to be ready by the end of 2016.

The host will have the capacity to act as an access point for a short time in order to save the battery autonomy. A local web server will be created on the host allowing configuring each sensor channel data acquisition and monitoring the sensor shield responses and the data stored in the host data base (block D1). The web server and access point have also been developed and are described in this report.

The addition of other functionalities as for example building a smartphone app for the near-to-real-time mapping of AirSensEUR observations, automatic detection of indoor/outdoor location, estimating population exposure based on data in the server backend database are currently discussed and/or being tested.

This report presents the host development, InFlux dataPush plus the sensor box. Additionally, it presents a LabView executable that can be used for data acquisition of laboratory tests of the sole sensor shield. The SOS data push and data correction will be presented in another EUR report likely end 2016 for the SOS data push and the algorithm of corrections.

2 Copyright notices

2.1 Electronic diagrams and AirSensEUR 3D box diagrams

The electronic diagrams of the AirSensEUR host (3.1) and the technical drawings of the AirSensEUR box (4.1) are distributed under the Creative Commons Attribution Share Alike Attribution, the CC BY SA 4.0 International (see 9). This license lets others remix, tweak and build upon the work being done even for commercial purposes, as long as credit is given as stated here below. New creations (any new works based on the current one) shall be licensed under the identical terms, so any derivatives will also allow commercial use. Additionally, any adaptation/improvement by third parties shall also be made available under the same license preventing from private appropriation of improvements.

The copyright notice for both types of drawings is as follows:

2.2 Software on the host

The JAVA software , shell scripts, C applications, drivers and configuration files developed for the AirSensEUR Host is released under the European Public License, an open source license for software. The copyright notice is as follows:

License

Date: 02/04/2015

Authors

- Michel Gerboles, michel.gerboles@jrc.ec.europa.eu; Laurent Spinelle, laurent.spinelle@jrc.ec.europa.eu, and Alexander Kotsev, alexander.kotsev@jrc.ec.europa.eu - European Commission - Joint Research Centre, and

- Marco Signorini, marco.signorini@liberaintentio.com

2.3 Software on the server

Data collected by the AirSensEUR Host can be sent to an external server using SOS transactions, or using an Influx data push mechanism. In this paper we set up a CentOS 6 based server, running InfluxDB. Data can be retrieved and graphs can be generated using Grafana. The following links give details about the licenses when implementing a solution with CentOS, Influx data pushing and Graphana dashboard.

CentOS 6: https://www.centos.org

InfluxDB: https://docs.influxdata.com/influxdb/v0.9/

Grafana: http://grafana.org

3 Technical description of the AirSensEUR host

The AirSensEUR host is a Linux based platform running on ARM core. The main hardware block, containing the ATMEL ARM9 CPU, 256MB DDR2 RAM, a micro SD cardholder and a bunch of other auxiliary items, is implemented by a commercial module named "Arietta G25" produced by Acme Systems Srl (www.acmesystems.it/arietta). The host board provides all is needed to customize a stock AriettaG25 for the AirSensEUR purposes.

The AirSensEUR Host board should be able to operate battery powered for several hours. It should be able to connect to a Sensor Shield via serial line, to retrieve GPS positioning information, to provide WiFi connectivity with external hosts. It should be able to push collected data to external servers through GPRS mobile network and, last but not least, it should allow for user interaction through a small operating panel.

A battery manager module, built on top of a Linear Technology LTC4156 chip, is connected through the CPU via I2C bus. The LTC4156 provides a fully programmable LiFePO4 battery charger and step-up converter. The module receives the current needed to charge the battery (or to power up the host board when no battery is connected) via a dedicated USB connector or a 5V 4A power supply adapter (those lines are protected from voltage overload and polarity reversal). The battery manager module is equipped with a fuel gauge IC LTC2942 made by Linear Technology. Connected to the same I2C bus, this IC allows the CPU to know the battery charge status.

The output of the battery manager is feed to IC6, a Maxim MAX8315A step-up converter that generates the 5V needed by the CPU module. The same applies to IC5 and IC8, dedicated to generate 5V@1A for, respectively, a pair of external GPRS (USB connector X1) and WiFi (USB connector X6) dongles.

The CPU module generates the 3.3V line needed by the on board GPS module, for the front panel and for the serial data line translator IC3. This is implemented by a NXP Semiconductor NVT2002 and is required to convert the Arietta serial line, operating a 3.3V, with the TTL serial line present on the Sensor Shield. A dedicated step-up converter, based on Texas Instruments LM4510, generates a 7V line for the Sensor Shield.

A Microchip MCP23017 I2C 16 lines GPIO expander implements the front panel with 8 software general purpose LEDs and 8 pushbuttons. A Microchip PIC12F1571, together with IC6 and the power-on pushbutton, implements the power up/down signals for the whole Host board.

The step-up converters, except IC6, are driven by a set of GPIO pins coming from the CPU. This technical solution allows the software to power up the needed peripherals only, thus limiting battery stresses.

Arietta G25 provides a debug header where a standard serial line, running at 115200bps 8N1 (3.3V only and NOT 5V tolerant), can be used to read the boot logs and to execute shell commands when no WiFi connection is available.

In the following pages, the electronic diagrams and bills of material of the host, control panel, battery manager and battery holder are given. The .pdf files can be downloaded at airsenseur.org and https://webgate.ec.europa.eu/CITnet/stash/projects/AIRSENSEUR

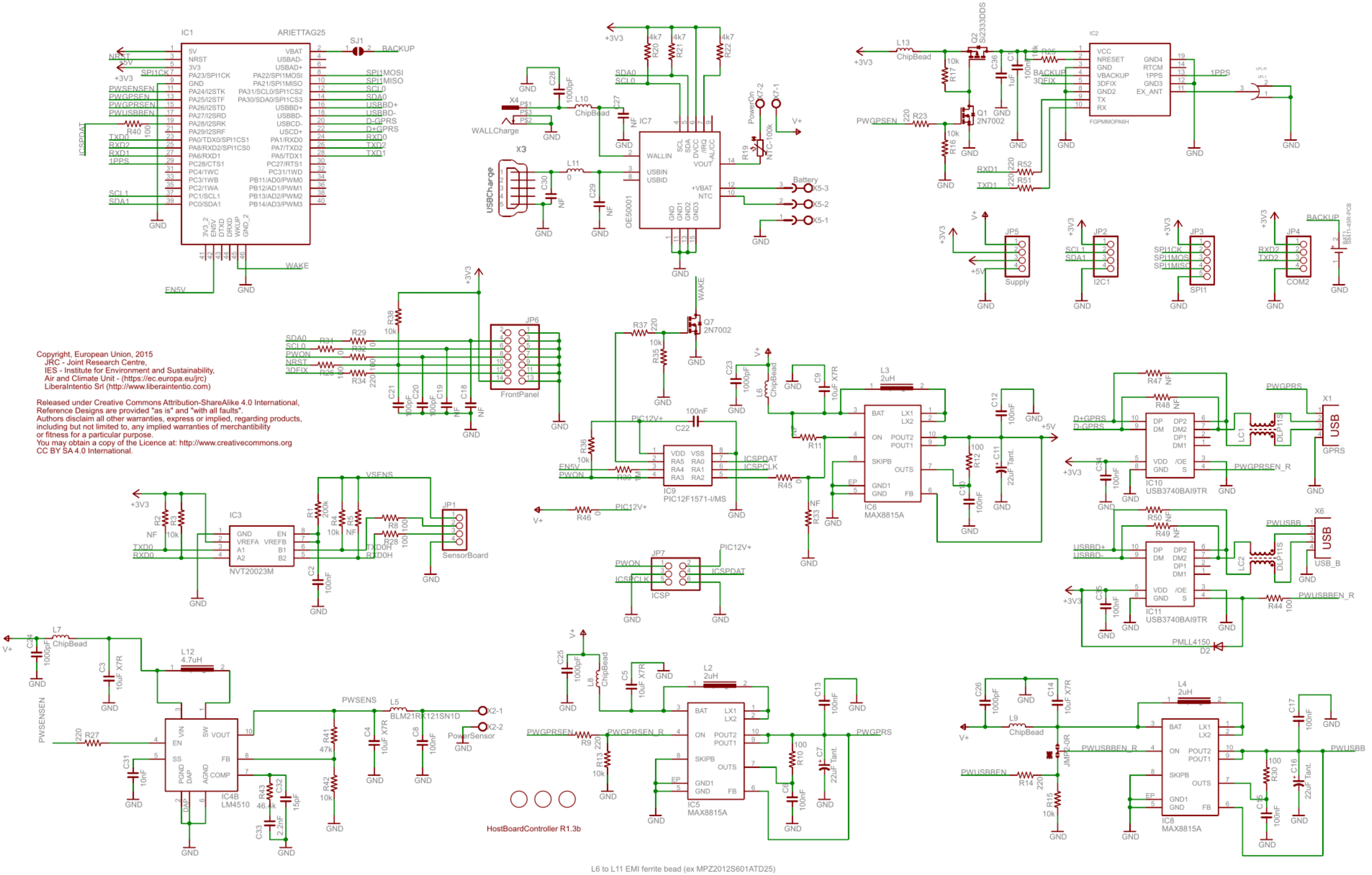


Figure 2: Electronic diagram of the host

Table 1: Bills of material of the host

Quantity	Value	Device	Parts
1		SJ	SJ1
1	0	L_0805	L11
4	0	R_0805	R29, R31, R45, R46
1	1M	R_0805	R39
1	1uF	C_0805	C36
1	2.2nF	C_0805	C33
2	2N7002	BSS123	Q1, Q7
3	2uH	SRN4018	L2, L3, L4
1	4.7uH	WE-TPC2828	L12
3	4k7	R_0805	R20, R21, R22
11	10k	R_0805	R3, R4, R13, R15, R16, R17, R25, R35, R36, R38, R42
1	10nF	C_0805	C31
5	10uF X7R	C_0805	C3, C4, C5, C9, C14
1	15pF	C_0805	C32
3	22uF Tant.	CP_3528	C7, C11, C16
1	46.4k	R_0805	R43
1	47k	R_0805	R41
9	100	R_0805	R8, R10, R12, R26, R28, R30, R32, R40, R44
12	100nF	C_0805	C1, C2, C6, C8, C10, C12, C13, C15, C17, C22, C34, C35
2	100pF	C_0805	C20, C21
1	200k	R_0805	R1
8	220	R_0805	R9, R14, R23, R27, R34, R37, R51, R52
5	1000pF	C_0805	C23, C24, C25, C26, C28
1	ARIETTAG25	ARIETTAG25	IC1
1	BLM21RK121SN1	D L_0805	L5
1	Battery	W232-3	X5
1	COM2	PINHD-1X4	JP4
6	ChipBead	L_0805	L6, L7, L8, L9, L10, L13
2	DLP11S	DLP11S	LC1, LC2
1	FGPMMOPA6H	FGPMMOPA6H	IC2
3	FIDUCIALMOUNT	FIDUCIALMOUNT	FM1, FM2, FM3
1	FrontPanel	PINHD-2X7	JP6
1	GPRS	PN87520	X1
1	I2C1	PINHD-1X4	JP2
1	ICSP	PINHD-2X3	JP7
1	LM4510	LM4510	IC4B
3	MAX8815A	MAX8815A	IC5, IC6, IC8
5	NF	C_0805	C18, C19, C27, C29, C30
8	NF	R_0805	R2, R5, R11, R33, R47, R48, R49, R50
1	NTC-100k	PTC-	R19
1	NVT20023M	NVT20023M	IC3
1	OE50001	OE50001	IC7

Quantity	Value	Device	Parts
1	PIC12F1571-I/	MS PIC12F1571/2-	MS IC9
1	PMLL4150	PMLL4150	D2
1	PowerOn	W237-02P	X7
1	PowerSensor	W237-02P	X2
1	S8411-45R-PCB	S8411-45R-PCB	BAT1
1	SPI1	PINHD-1X5	JP3
1	SensorBoard	PINHD-1X4	JP1
1	Si2333DDS	BSS84	Q2
1	Supply	PINHD-1X4	JP5
1	UFL-R	UFL-R	UFL1
2	USB3740BAI9TR	USB3740BAI9TR	IC10, IC11
1	USBB AON	0R-JUMPB	JMP2
1	USBCharge	MINI-USB-UX60	-MB-5ST X3
1	USB_B	PN87520	X6
1	WALLCharge	PJ-050AH	X4

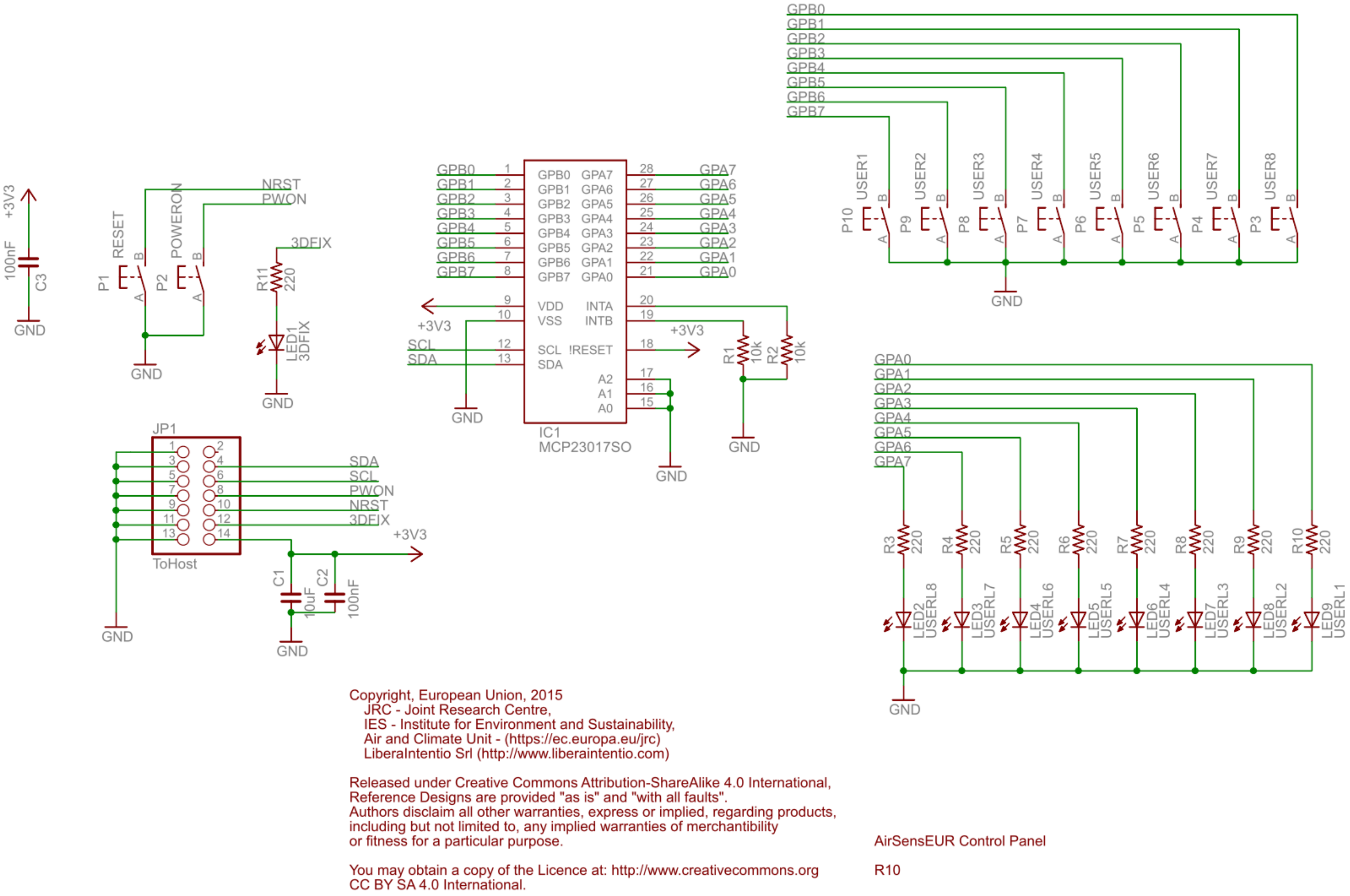
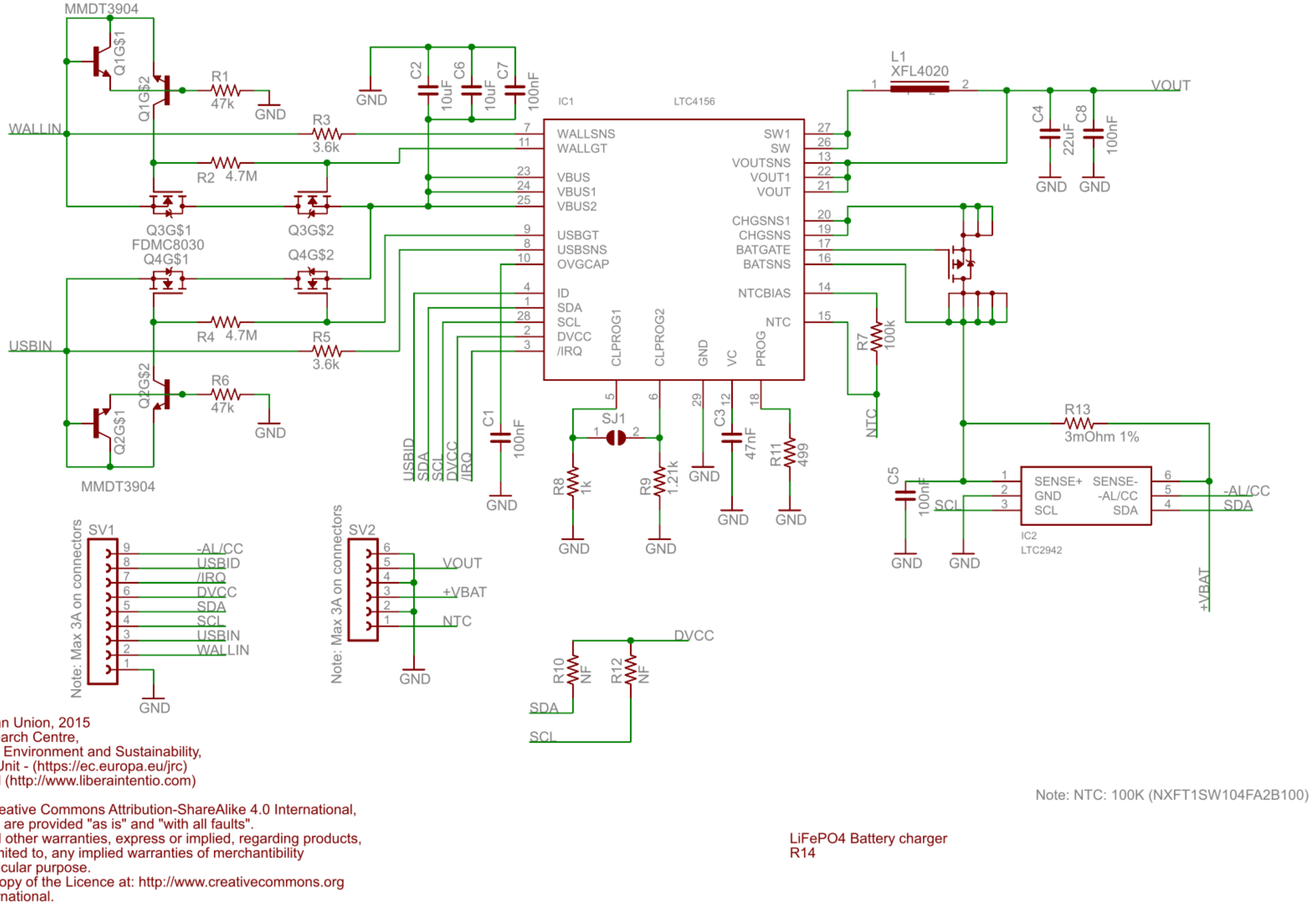


Figure 3: Electronic design of the Control panel

Table 2: Bills of material of the control panel

Quantity	Value	Device	Parts
1	3DFIX	LEDCHIP-LED0805	LED1
2	10k	R_0805	R1, R2
1	10uF	C_0805	C1
2	100nF	C_0805	C2, C3
9	220	R_0805	R3, R4, R5, R6, R7, R8, R9, R10, R11
1	MCP23017SO	MCP23017SO	IC1
1	POWERON	147873-1E	P2
1	RESET	147873-1E	P1
1	ToHost	PINHD-2X7	JP1
1	USER1	147873-1E	P10
1	USER2	147873-1E	P9
1	USER3	147873-1E	P8
1	USER4	147873-1E	P7
1	USER5	147873-1E	P6
1	USER6	147873-1E	P5
1	USER7	147873-1E	P4
1	USER8	147873-1E	P3
1	USERL1	LEDCHIP-LED0805	LED9
1	USERL2	LEDCHIP-LED0805	LED8
1	USERL3	LEDCHIP-LED0805	LED7
1	USERL4	LEDCHIP-LED0805	LED6
1	USERL5	LEDCHIP-LED0805	LED5
1	USERL6	LEDCHIP-LED0805	LED4
1	USERL7	LEDCHIP-LED0805	LED3
1	USERL8	LEDCHIP-LED0805	LED2

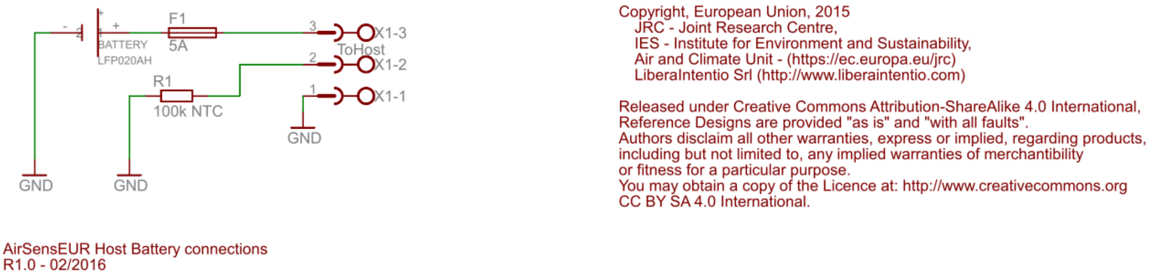


LiFePO4 Battery charger
R14

Figure 4: Electronic diagram of the Battery Manger

Table 3: Bills of material of the battery manager

Quantity	Value	Device	Parts
1		FE06-1	SV2
1		FE09-1	SV1
1		SJ	SJ1
1	1.21k	R_0805	R9
1	1k	R_0805	R8
2	3.6k	R_0805	R3, R5
1	3mOhm 1%	R_0805	R13
2	4.7M	R_0805	R2, R4
2	10uF	C_0805	C2, C6
1	22uF	C_0805	C4
2	47k	R_0805	R1, R6
1	47nF	C_0805	C3
1	100k	R_0805	R7
4	100nF	C_0805	C1, C5, C7, C8
1	499	R_0805	R11
2	FDMC8030	FDMC8030	Q3, Q4
1	LTC2942	LTC2941	IC2
1	LTC4156	LTC4156	IC1
2	MMDT3904	MMDT3904	Q1, Q2
2	NF	R_0805	R10, R12
1	SI7123DN	SI7123DN	Q5
1	XFL4020	XFL4020	L1



AirSensEUR Host Battery connections
R1.0 - 02/2016

Figure 5: Electronic diagram of the battery holder

Table 4: Bills of material of the battery holder

Quantity	Value	Device	Parts
1	5A	SH22,5A	F1
1	100k NTC	R-EU_0207/5V	R1
1	LFP020AH	LFP020AH	BATTERY
1	ToHost	W232-3	X1

3.1 Software on the Host

The AirSensEUR Host is a complex system running a Debian Wheezy custom distribution. Due to their complexity, a lot of applications are involved and a set of well-known services already provides some basic operations. This is the case, for example, for the whole network layers/services like the IP stack, DHCP, SSH, NTP, HTTP, GPSd daemons. The benefits of Linux are extended thanks to the availability of a large set of hardware and middle-layer drivers for the most used interfaces like, for example, I2C, SPI, UART, USB, LEDs, power supplies, keyboard event handlers and other.

The availability of a headless Java Runtime for the ARM CPU reduced the overall development time due to a large availability of libraries implementing high level services like, for example, client/server RPC, JSON serializers and so on.

The AirSensEUR Host implements several applications to perform their tasks. They're mainly written in Java, except for a small application handling the buttons in the user's panel, which is written in plain C, and a jsReact + PHP based web configuration panel. Those applications are detailed in next paragraphs.

3.2 The EventMonitor

The EventMonitor is a small 'C' application that is automatically started up by the boot process and runs forever in background. They handle events generated by the AirSensEUR control panel.

The height software configurable buttons located in the AirSensEUR Host user panel are connected to a Microchip MCP23017 I2C GPIO expander. A Linux kernel driver, that exposes the physical lines as a set of mapped GPIO ports, supports this integrated circuit. This driver can be logically connected to a second Linux kernel driver that implements a virtual keyboard, thus resulting in a Linux kernel event each time a panel button is pressed. The same logic is applied for the GPIO button present to the CPU board.

The EventMonitor application waits for those events and generates a private command shell able to execute a specific shell script stored to the predefined path /etc/eventmonitor.

This technical solution allows developers to easily react, with a simple script implementation, to button pressure (and release) events. The AirSensEUR reference package tree provides a set of standard scripts targeting each button present in the AirSensEUR Host.

3.3 The AirSensEURHost

AirSensEURHost is a Java application able to communicate with an external AirSensEUR Shield. It also provides a JSON RPC server abstraction layer for other processes. This process is started up/down through a specific EventMonitor script. The process takes care to connecting to the Shield via a serial line. It implements the AirSensEUR Shield protocol in order to continuously poll the Shield for new samples. Each time a new sample is detected and retrieved from the Shield, it is stored in a temporary memory and let it available in JSON format to whatever process needs to read it.

This technical solution allows multiple processes to consume samples generated by the AirSensEUR Shield without concerning of protocol communication details.

The AirSensEURHost is also responsible for applying a mathematical function, proper to each Sensor Shield channel, to each incoming sample, thus converting a raw digital value in a human readable and/or calibrated format.

The main operating parameters, like, for example, the serial line name, the polling time period, the mathematical functions and other, are stored on the text file /usr/local/etc/sensor.properties for easy customization.

3.4 The AirSensEURDataAggregator

The AirSensEURDataAggregator is the application ‘consuming’ the data generated by the AirSensEURHost and the GPSd daemon.

This process is started up/down through a specific EventMonitor script, together with the AirSensEURHost companion application. The process takes care to connect to the AirSensEURHost and the GPSd daemon in order to retrieve new samples coming from the AirSensEUR Shield and aggregate them with positional information coming from the GPS subsystem.

Each sample is stored into local filesystem as a .csv file and to a local sqlite compatible database for further usage. All those files can be found under /usr/local/airsenseur/AirSensEURDataAggregation path.

The main operating parameters like, for example, the AirSensEURHost and GPSd daemon IP address and port, are stored on the text file /usr/local/etc/aggregator.properties for easy customization.

3.5 The AirSensEURDataPush

The AirSensEURDataPush is the application taking care of pushing collected data to an external server. Two operating modes are provided: Influx and SOS52North.

The process is started up/down through a specific script periodically called by a cron job. The AirSensEURDataPush loads data from the sqlite database generated by the AirSensEURDataAggregator.

The main operating parameters are stored in configuration files located in the /usr/local/etc/ folder (there are two different configuration files, one for each operating mode) for easy customization.

3.6 The Web Management Console tool

The Web Management Console tool is a jsReact + PHP client/server web application running on the AirSenEUR Host. The main goal of the Web Management Console is to provide an easy way to configure and test the AirSensEUR Host parameters. Their “Live” page can be used to read the latest samples taken from the sensor shield and, for this reason, it’s very useful for an in place functionality check.

The Web Management Console can be reach via (mobile) browser through the WiFi connection provided by the AirSensEUR Host.

4 Assembling AirSensEUR

4.1 Selecting socket of the shield according to the sensors

Once you have selected the sensors (see report A[1]) that you want to use, select and weld the corresponding sensor sockets on the shield according to the following table:

Sensor Brand	Sensor model type	Sockets	External diameter of pin/socket (through PCB)	Diameter of the hole on PCB
CityTech	Sensoric	Mill-Max Model TO-5 model 917-XX-104-41-005000. https://www.mill-max.com/assets/pdfs/metric/136M.PDF	5.08 mm	
	Serie 4	Cambion.co.uk, model 450-3326-01-xx-00 http://www.cambion.co.uk/catalogue_page.aspx?ID=11	2.49 mm	
	Serie 7	Cambion.co.uk, model 450-5301-01-xx-00	1.88 mm	
Membrapor	miniature	Mill-Max model 9801-0-15-15-23-27-10-0 https://www.mill-max.com/products/rec/9801/9801-0-15-15-23-27-10-0	2.54 mm	
	compact	Mill-Max model: 0322-0-15-15-34-27-10-0 https://www.mill-max.com/products/rec/0322/0322-0-15-15-34-27-10-0 ,	2.16 mm	
AlphaSense	A,	Mill-Max: 0364-0-15-80-13-80-10-0 https://www.mill-max.com/products/rec/0364/0364-0-15-80-13-80-10-0 https://www.mill-max.com/assets/pdfs/metric/189M.pdf	2.49 mm	2.54 mm
	B,	Mill-MaX: 0327-0-15-80-34-80-10-0 https://www.mill-max.com/assets/pdfs/metric/183M.pdf	1.83 mm	1.91 mm
SGX-Sensotech	EC4	Andon Electronics, Model: R530-0403-02N-440P55-R27-L14	2.49 mm	

[1] Gerboles M., et al, AirSensEUR: an open data/software /hardware multi-sensor platform for air quality monitoring. Part A: sensor shield, doi: 10.2788/30927, http://publications.jrc.ec.europa.eu/repository/handle/JRC97581

Sensor Brand	Sensor model type	Sockets	External diameter of pin/socket (through PCB)	Diameter of the hole on PCB
		http://sgx.cdistore.com/datasheets/andon/R530-0403-02N-440P55-R27-L14.pdf		
	7	Andon Electronics http://sgx.cdistore.com/datasheets/andon/Andon_%20Loose_Pin_Drawing.pdf mod -285E mod -295E	? 1.52 mm	

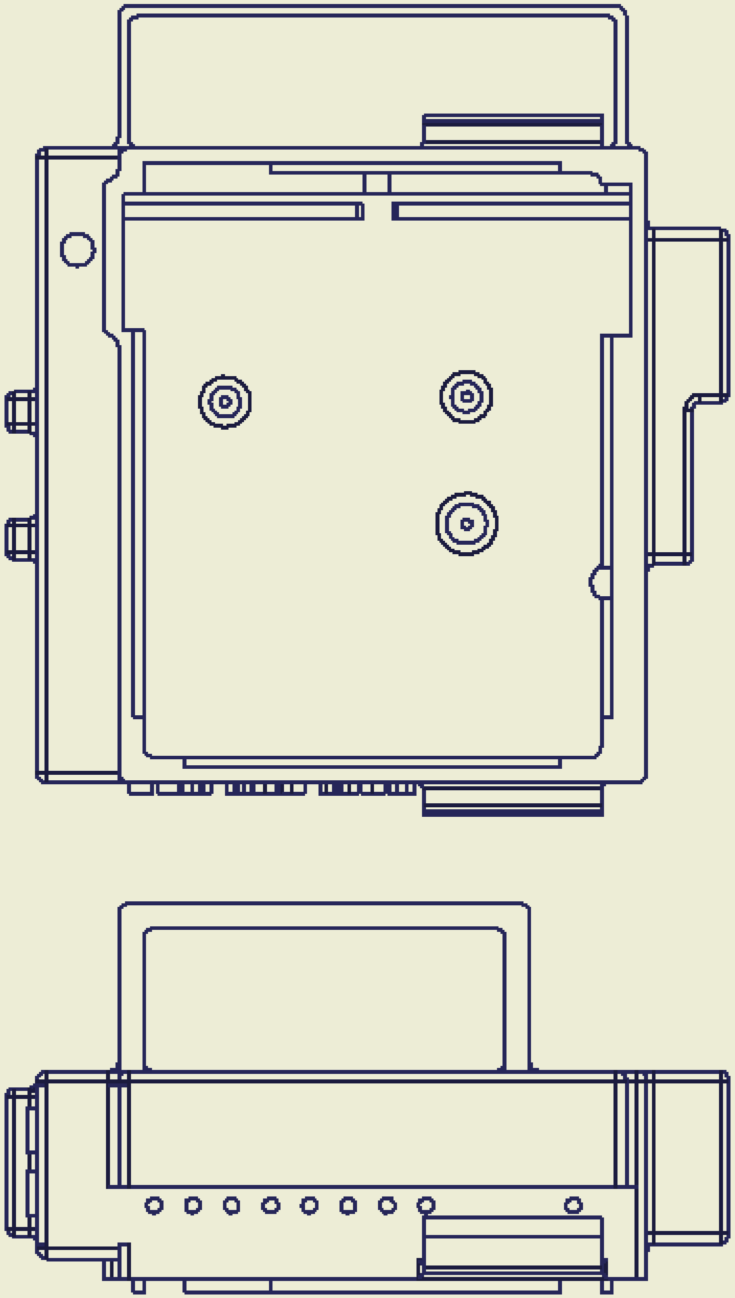
4.2 3D Printing of the sensor Box

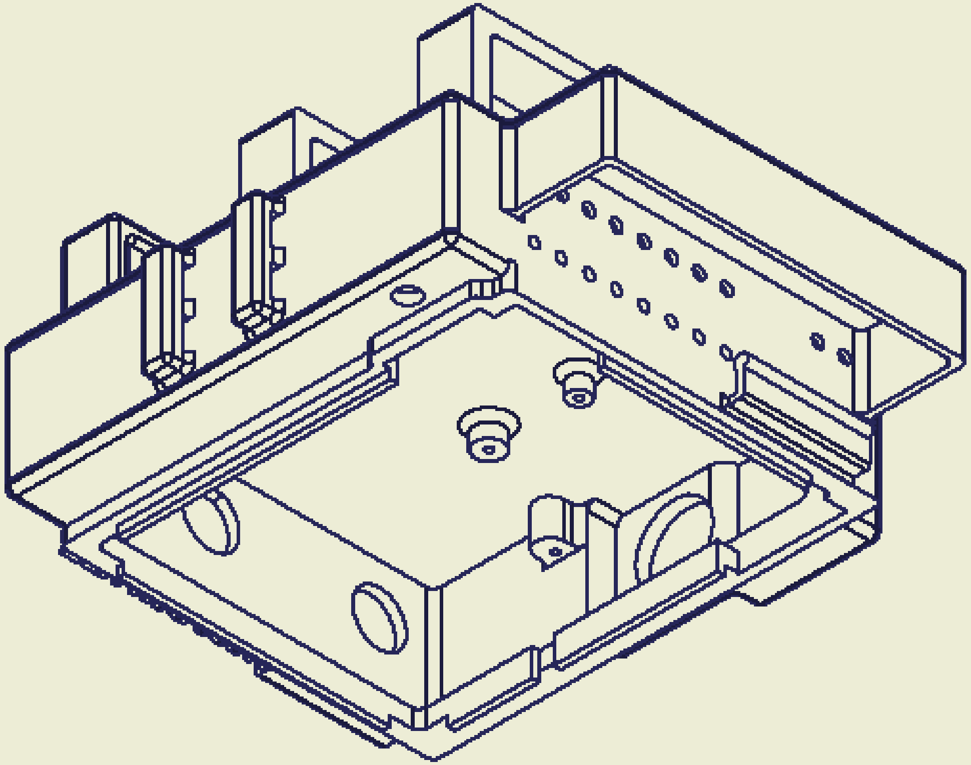
Prior to assemble the air Sensor platform, the sensor box shall be 3D printed. The sensor box consists of 5 elements:

- The box for the sensor shield
- The box for the host
- The box for the board performing the ancillary measurements (T-P-RH)
- If needed a sensor cover according to the size of the selected sensors. A cover is not needed if 2 sensors type Alphasense A or Membrapor Miniature plus 2 sensors type Alphasense B or Membrapor compact are selected.
- Push buttons (10)

The 3 boxes should be printed using inert material (for example ISO-Poly-Carbonate manufactured by Stratasys or equivalent). For a better quality of the box, thread diameter TP 16 should be used for the boxes of the host and sensor shield. The ancillary measurements (T-RH-P R13 and box cap) should be printed with thread diameter Tp 12.

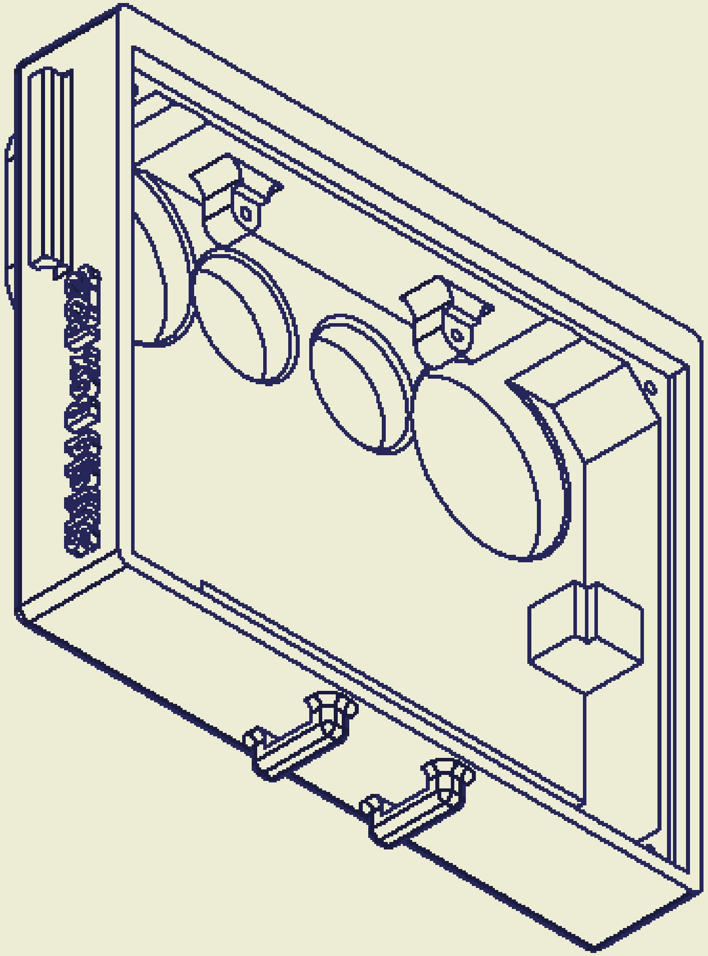
In the following pages, technical drawing or the boxes are given. The .stl files to be used for 3D printing can be downloaded at airsenseur.org and https://webgate.ec.europa.eu/CITnet/stash/projects/AIRSENSEUR

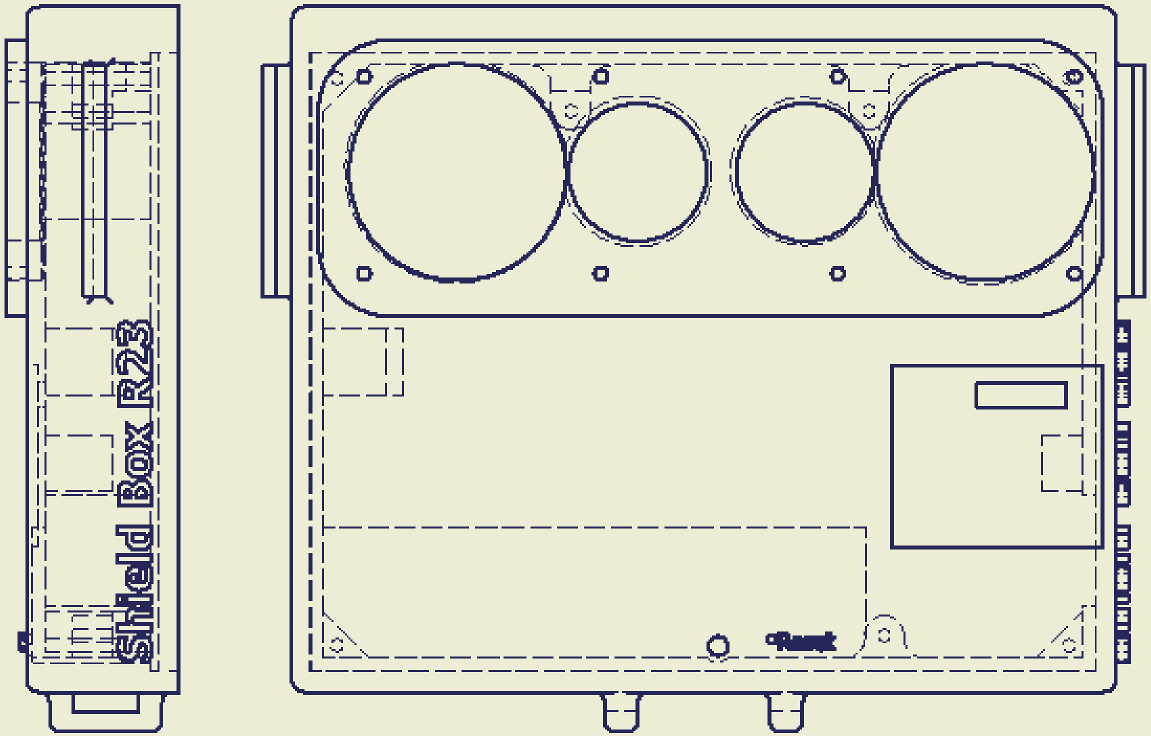




1	1	Box for Host version 1.2 and 1.3	Polycarbonate-ISO, R16
Id.	Qty		

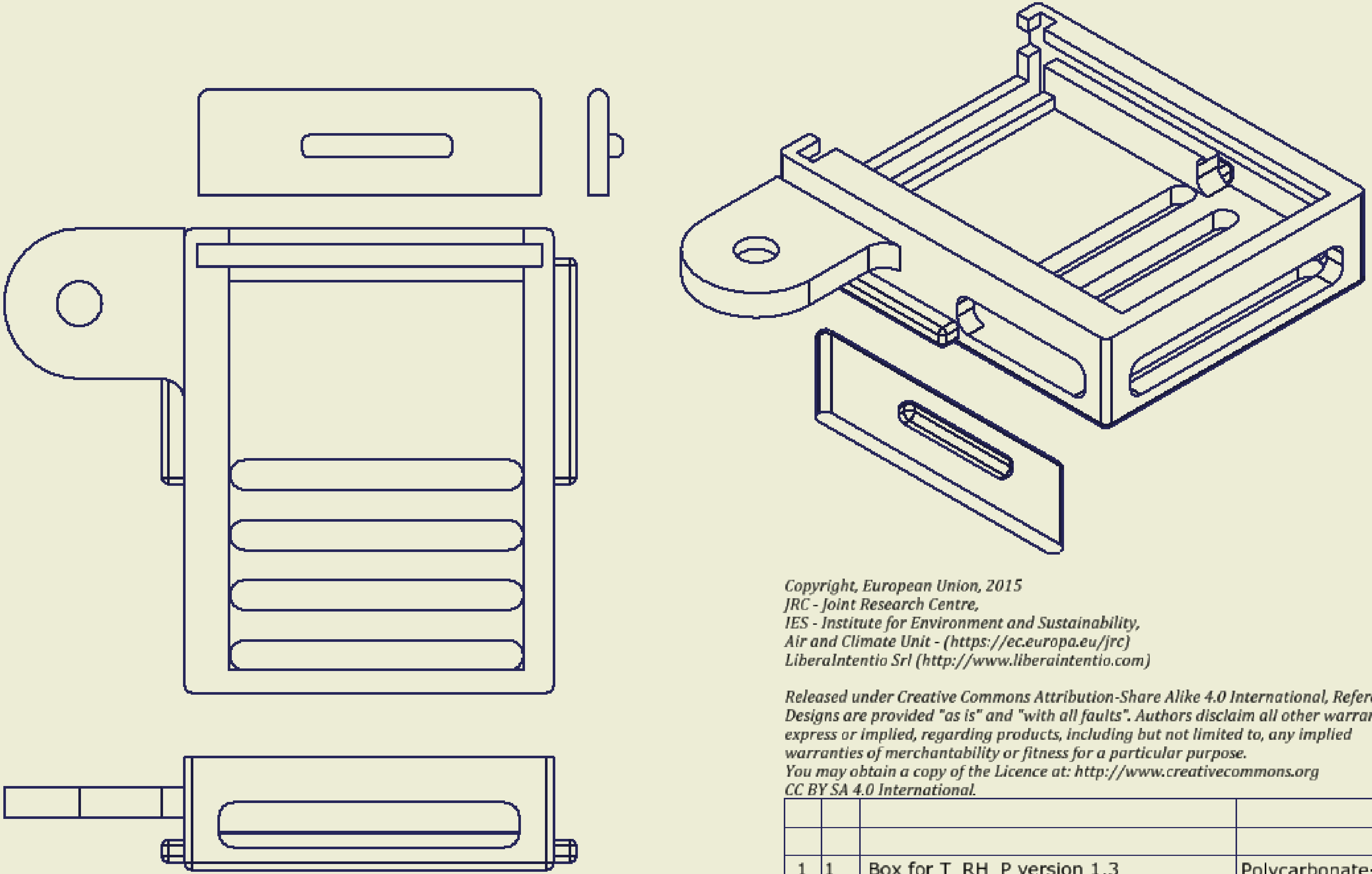
European Commission JRC - Ispra	DESIGNED Michel Gerboles		SCALE 1.5:1	REV. 1,1	DATE 2016-05-21
	AirSensEUR Box for host		AirSensEUR 160521 Box Host R12.ipt		





1	1	Box for Shield version 2.2 and 2.3	Polycarbonate-ISO, R16
Id.	Qty		

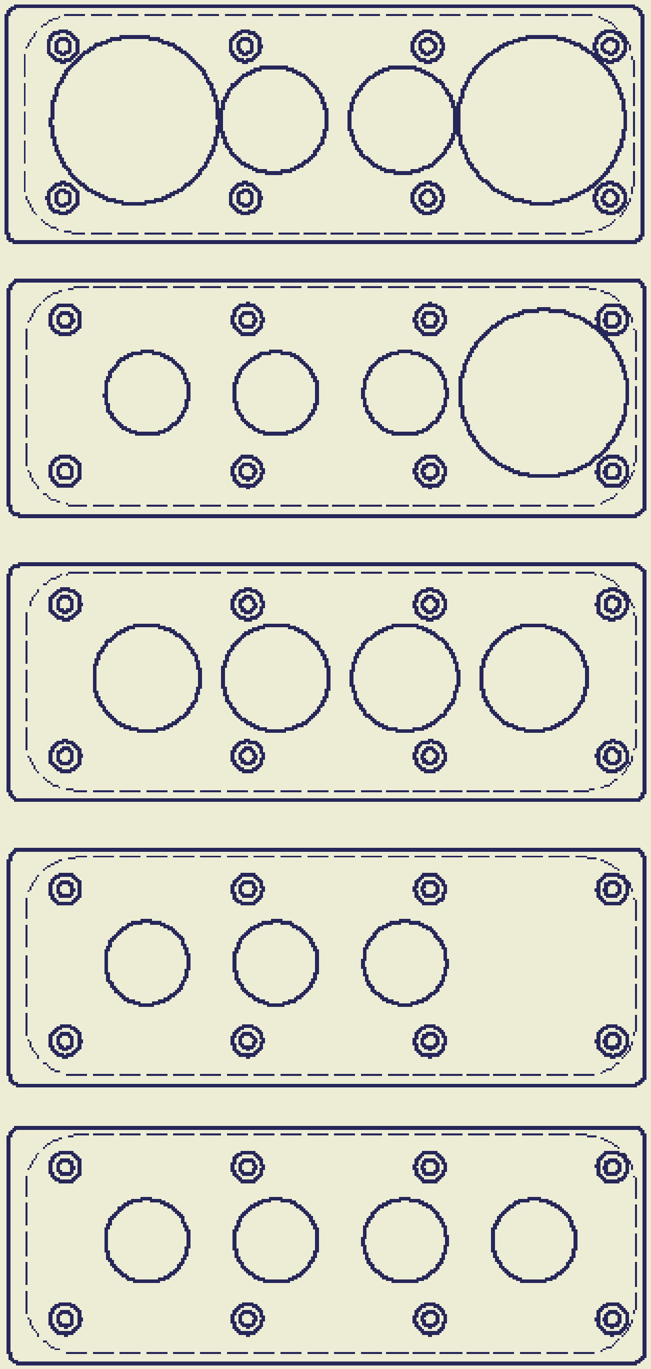
European Commission JRC - Ispra	DESIGNED Michel Gerboles		SCALE 1.4:1	REV. 1.0	DATE 2016-05-22
	AirSensEUR Box for shield		AirSensEUR		
			160522 Box shield R23.ipt		

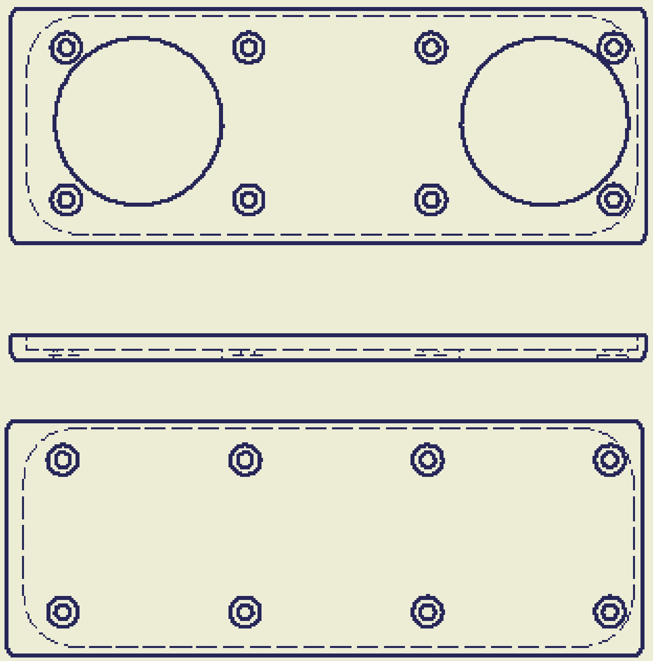


Copyright, European Union, 2015
JRC - Joint Research Centre,
IES - Institute for Environment and Sustainability,
Air and Climate Unit - (https://ec.europa.eu/jrc)
LiberaIntentio Srl (http://www.liberaintentio.com)

1	1	Box for T_RH_P version 1.3	Polycarbonate-ISO, R12
Id.	Qty		

European Commission JRC - Ispra	DESIGNED Michel Gerboles		SCALE 4:1	REV. 2.0	DATE 2016-07-29
	AirSensEUR Box for T_P_RH		AirSensEUR 160409 T-RH-P Box cap R23p.ipt		





1	1	Covers for Shield version 2.2_2.3	Polycarbonate-ISO, R16
Id.	Qty		

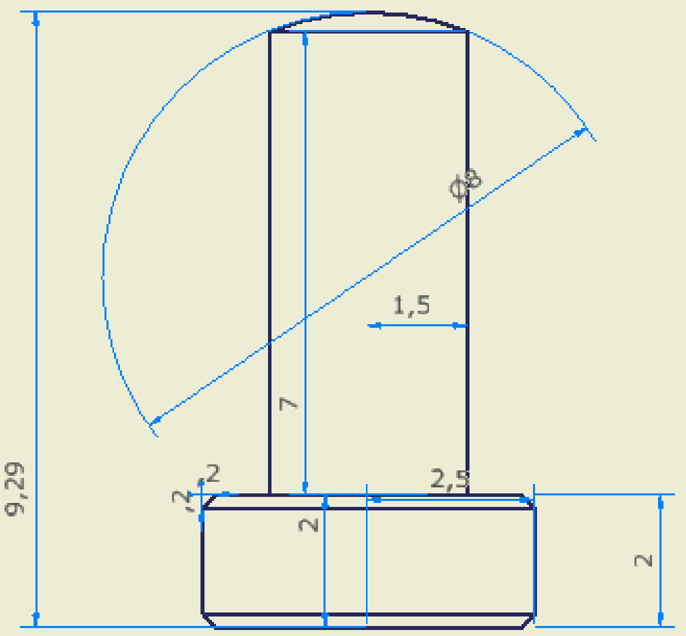
	DESIGNED Michel Gerboles		SCALE 1:1	REV. 1,0	DATE 2016-05-05

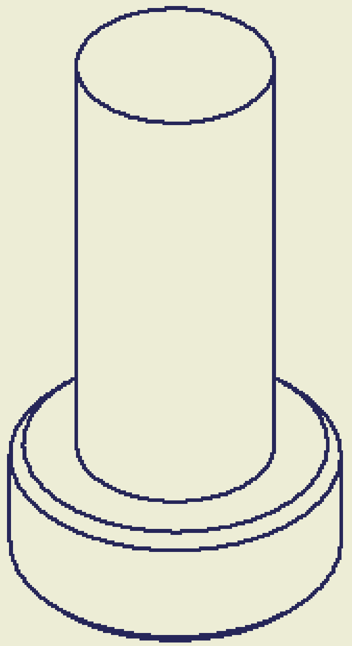
European Commission
JRC - Ispra

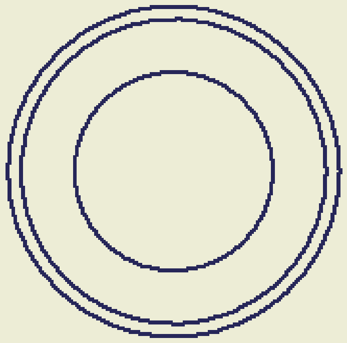
Sensors covers

AirSensEUR

160409 Sensor cover B Compact - None - None - B_Compact R23.ipt

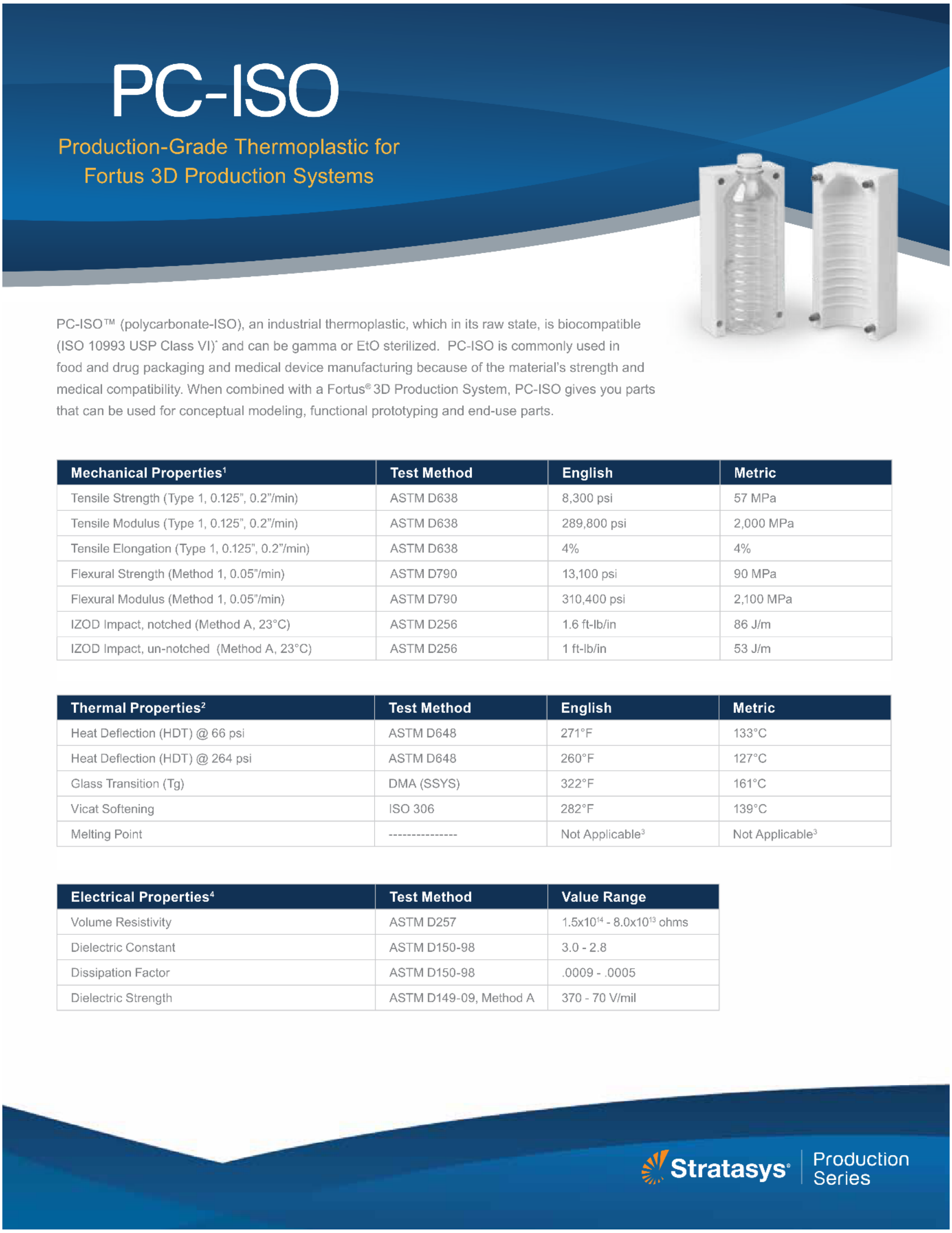






1	1	Button for Host version 1.2 and 1.3	Polycarbonate-ISO, R16
Id.	Qty		

European Commission JRC - Ispra	DESIGNED Michel Gerboles		SCALE 13:1	REV. 1,0	DATE 2016-05-05
	AirSensEUR button for host		AirSensEUR		
			160512 button.ipt		



PC-ISO

Production-Grade Thermoplastic for Fortus 3D Production Systems

PC-ISO™ (polycarbonate-ISO), an industrial thermoplastic, which in its raw state, is biocompatible (ISO 10993 USP Class VI)* and can be gamma or EtO sterilized. PC-ISO is commonly used in food and drug packaging and medical device manufacturing because of the material's strength and medical compatibility. When combined with a Fortus® 3D Production System, PC-ISO gives you parts that can be used for conceptual modeling, functional prototyping and end-use parts.

Mechanical Properties[1]	Test Method	English	Metric
Tensile Strength (Type 1, 0.125", 0.2"/min)	ASTM D638	8,300 psi	57 MPa
Tensile Modulus (Type 1, 0.125", 0.2"/min)	ASTM D638	289,800 psi	2,000 MPa
Tensile Elongation (Type 1, 0.125", 0.2"/min)	ASTM D638	4%	4%
Flexural Strength (Method 1, 0.05"/min)	ASTM D790	13,100 psi	90 MPa
Flexural Modulus (Method 1, 0.05"/min)	ASTM D790	310,400 psi	2,100 MPa
IZOD Impact, notched (Method A, 23°C)	ASTM D256	1.6 ft-lb/in	86 J/m
IZOD Impact, un-notched (Method A, 23°C)	ASTM D256	1 ft-lb/in	53 J/m

Thermal Properties[2]	Test Method	English	Metric
Heat Deflection (HDT) @ 66 psi	ASTM D648	271°F	133°C
Heat Deflection (HDT) @ 264 psi	ASTM D648	260°F	127°C
Glass Transition (Tg)	DMA (SSYS)	322°F	161°C
Vicat Softening	ISO 306	282°F	139°C
Melting Point	---------------	Not Applicable[3]	Not Applicable[3]

Electrical Properties[4]	Test Method	Value Range
Volume Resistivity	ASTM D257	$1.5x10^{14}$ - $8.0x10^{13}$ ohms
Dielectric Constant	ASTM D150-98	3.0 - 2.8
Dissipation Factor	ASTM D150-98	.0009 - .0005
Dielectric Strength	ASTM D149-09, Method A	370 - 70 V/mil

Stratasys® | Production Series

Other[2]	Test Method	Value
Specific Gravity	ASTM D792	1.2
Flame Classification	UL 94	HB

System Availability	Layer Thickness Capability	Support Structure	Available Colors
Fortus 380mc™	0.013 inch (0.330 mm)	Breakaway	□ White
Fortus 400mc™	0.010 inch (0.254 mm)		▨ Translucent Natural
Fortus 450mc™	0.007 inch (0.178 mm)		
Fortus 900mc™			

The information presented are typical values intended for reference and comparison purposes only. They should not be used for design specifications or quality control purposes. End-use material performance can be impacted (+/-) by, but not limited to, part design, end-use conditions, test conditions, etc. Actual values will vary with build conditions. Tested parts were built on Fortus 400mc @ 0.010" (0.254 mm) slice. Product specifications are subject to change without notice.

The performance characteristics of these materials may vary according to application, operating conditions, or end use. Each user is responsible for determining that the Stratasys material is safe, lawful, and technically suitable for the intended application, as well as for identifying the proper disposal (or recycling) method consistent with applicable environmental laws and regulations. Stratasys makes no warranties of any kind, express or implied, including, but not limited to, the warranties of merchantability, fitness for a particular use, or warranty against patent infringement.

**It is the responsibility of the finished device manufacturer to determine the suitability of all the component parts and materials used in their finished products.*

[1] *Build orientation is on side long edge.*

[2] *Literature value unless otherwise noted.*

[3] *Due to amorphous nature, material does not display a melting point.*

[4] *All Electrical Property values were generated from the average of test plaques built with default part density (solid). Test plaques were 4.0 x 4.0 x 0.1 inches (102 x 102 x 2.5 mm) and were built both in the flat and vertical orientation. The range of values is mostly the result of the difference in properties of test plaques built in the flat vs. vertical orientation.*

Stratasys | www.stratasys.com | info@stratasys.com

7665 Commerce Way
Eden Prairie, MN 55344
+1 888 480-3548 (US Toll Free)
+1 952 937-3000 (Intl)
+1 952 937-0070 (Fax)

2 Holtzman St.,
Science Park, PO Box 2496
Rehovot 76124, Israel
+972 74 745-4000
+972 74 745-5000 (Fax)

At the core: Advanced FDM Technology

Fortus systems are based on Stratasys® FDM® (fused deposition modeling) technology. FDM is the industry's leading additive manufacturing technology, and the only one that uses production-grade thermoplastics, enabling the most durable parts.

Fortus systems use a wide range of thermoplastics with advanced mechanical properties so your parts can endure high heat, caustic chemicals, sterilization and high-impact applications.

No special facilities needed

You can install a Fortus 3D Production System just about anywhere. No special venting is required because Fortus systems don't produce noxious fumes, chemicals or waste.

No special skills needed

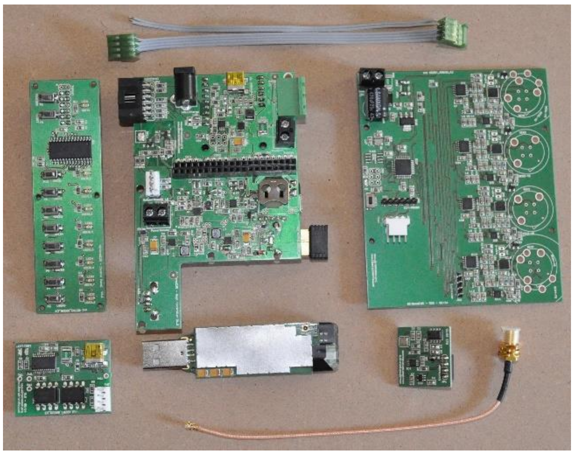
Fortus 3D Production Systems are easy to operate and maintain compared to other additive fabrication systems because there are no messy powders to handle and contain. They're so simple, an operator can be trained to operate a Fortus system in less than 30 minutes.

Get your benchmark on the future of manufacturing

Fine details. Smooth surface finishes. Accuracy. Strength. The best way to see the advantages of a Fortus 3D Production System is to have your own part built on a Fortus system. Get your free part at: stratasys.com.

4.3 Assembling an AirSensEUR platform with the AirSensEUR kit

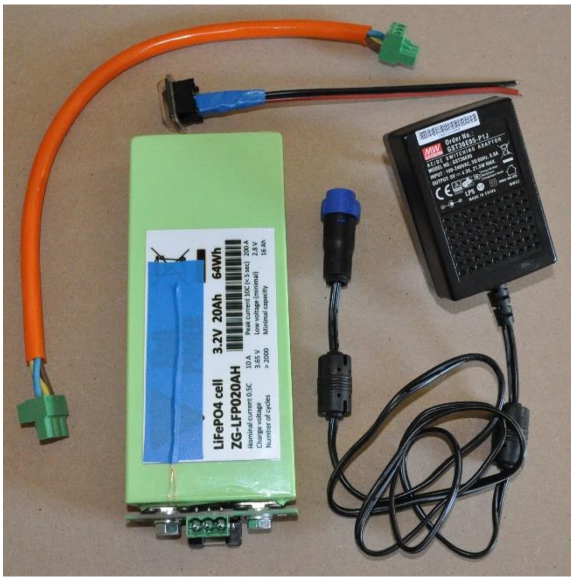
4.4 Content of the AirSensEUR kit:

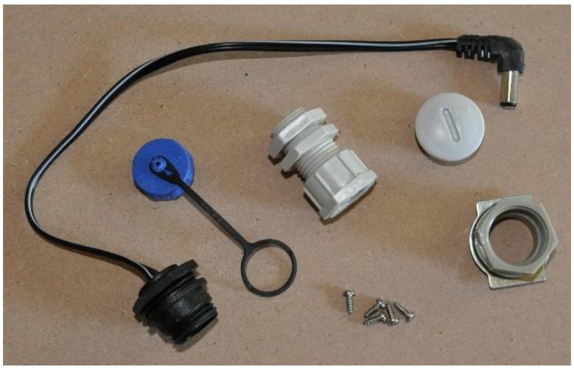


From left to right:

- Control Panel
- USB2Serial connector
- Host
- GPRS dongle
- Sensor Shield with sensor sockets (MillMax 9801-0-15-15-23-27-10-0 or Mouser 575-9801015152327 and MillMax 0322-0-15-15-34-27-10-0 or Mouser 575-032201515)
- Temperature/Humidity/Pressure board
- Flat wire to connect Host and Shield (top)
- GPS wire (bottom)

- Waterproof battery wire (orange)
- LiFePO4 20A 64Wh battery with holder
- ON/OFF Switch
- Power Supply with connector Bulgin PX0410/02P/5560 (RS: 426-0896) with pin Bulgin SA3350/1 (RS: 426-1338)



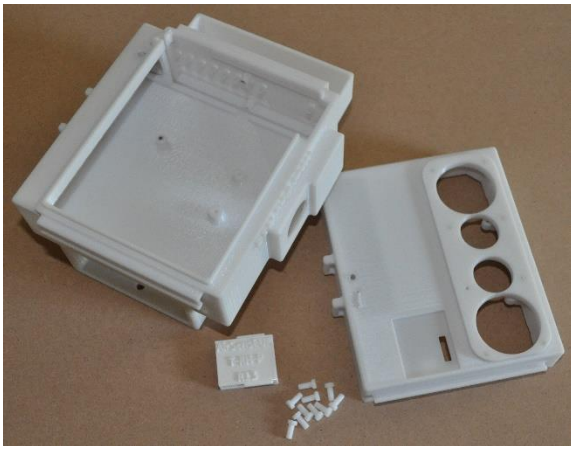


- Power supply cable (2.1mm 900 adaptor cable; RS: 6563850)with connector Bulgin PX0412/02S (RS: 426-1057) with pin Bulgin SA3349/1 (RS: 426-1350)
- Waterproof tap (blue) Bulgin PX0480/1 (RS: 426-1259)
- Cable Gland PG9 IP66 Gewiss GW52002
- Cable Grommet PG 98311
- Tap for Cable Grommet
- 5 x Flat head screw (bottom)

4.5 Tools and other parts:

Tools needed:

- Flat head screwdriver
- Bent long nose pliers
- Pliers
- Coin (2 cents works fine)
- A multimeter



You also need the 3D printed box parts:

- Host Box
- T-RH-P box with cap
- 10 x Press-button
  (diam 5 x2; diam 3 x 5)
- Shield Box

Some others parts are also needed but are not display in this document:

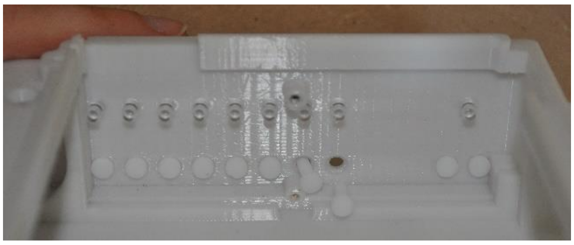
- 9 x Light pipes Mentor 1265.2001 (RS: 879-6030)
- 1 x Light pipe Mentor 1265.2005 (RS 879-6049)
- 2 x Bosh Lock P/N: 2601320027

4.6 Assembly

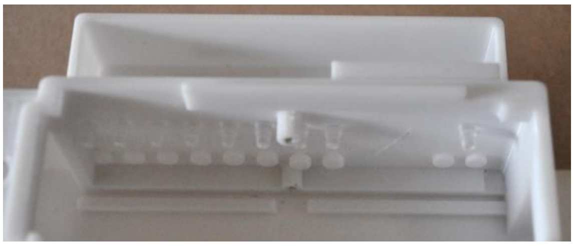
Tips before assembly:

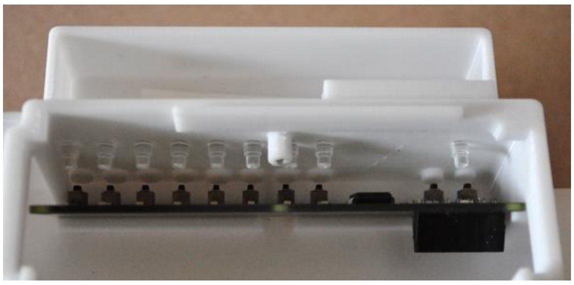
- Cut the Power supply cable to a length of 22 cm (nearly the small edge of an A4 paper format which is 21 cm).
- Match the polarity of the Power supply cable and the Power supply:
  - Black cable of the Power supply cable with the Black cable of the Power supply.
  - White cable of the Power supply cable with the White cable of the Power supply.
- Use the female pin in the female Bulgin connector (box mount).
- Use the male pin in the male Bulgin connector (power supply).
- Cut 1 cm of the Waterproof battery wire on the PCB side to ease the installation.
- Use insulating adhesive tape to protect the temperature sensor on the battery.

Assembly:

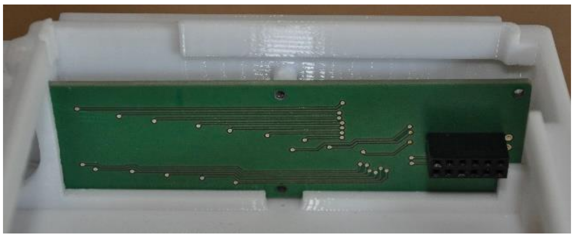


1. Insert the small Lightpipe (x 9) and the push button (x 10) in the hole of the Host Box.

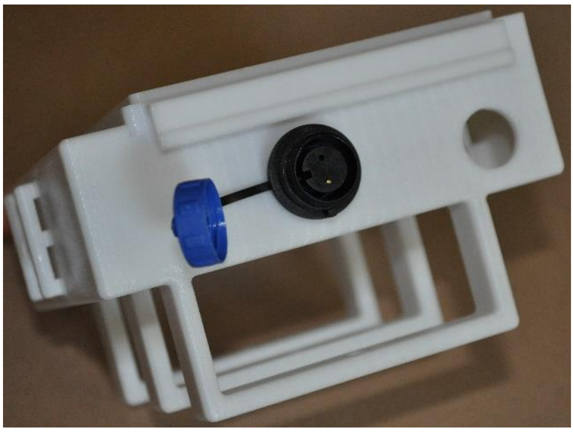




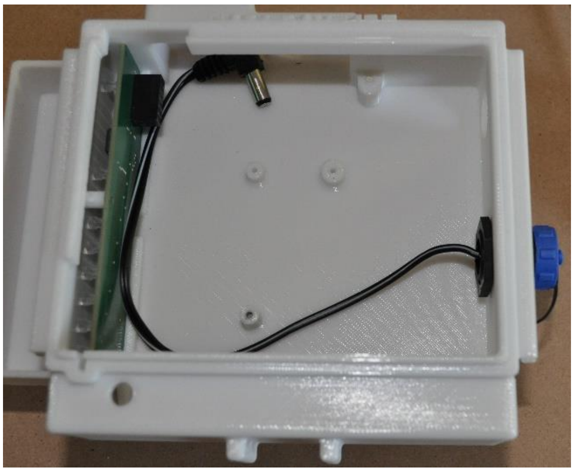
2. Insert the Control Panel. First slide the bottom of the PCB in the groove in front of the push button. Then bring up the top of the PCB against the screw hole.



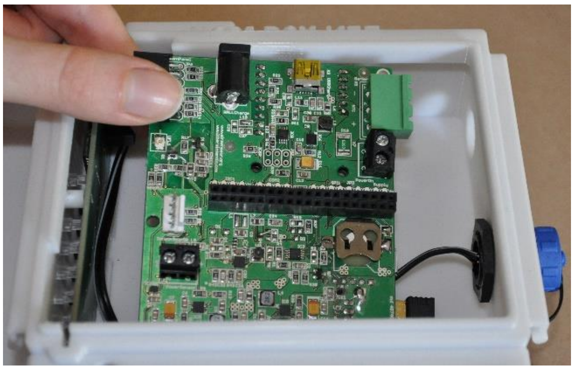
3. The Control Panel should slide in the bottom slot as shown on the picture.

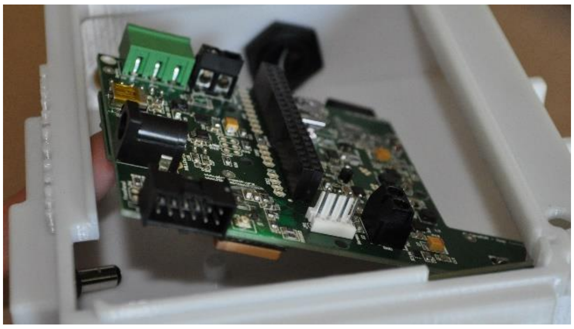


4. Mount the Power supply cable with waterproof connector with the waterproof tap (blue) on the opposite side of the box.

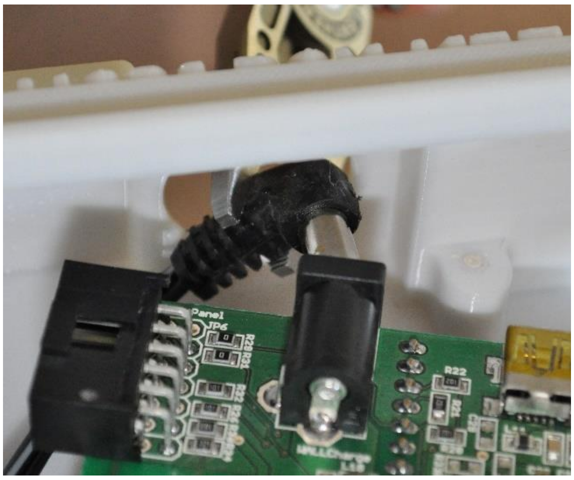


5. Place the cable in the box as shown here in order ease the mounting of the Host PCB in the next step.

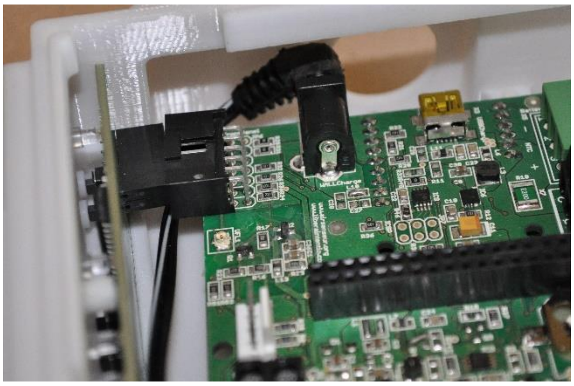




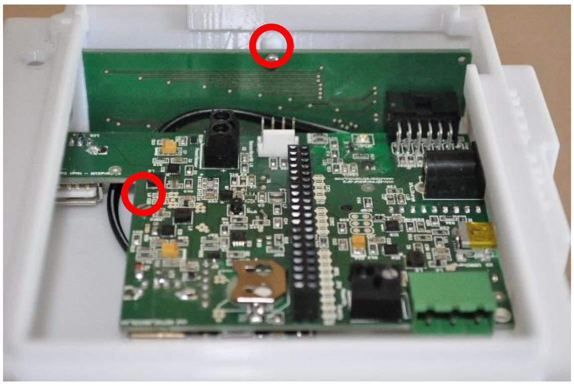
6. First, insert the Host PCB by the USB plug as shown in the pictures.



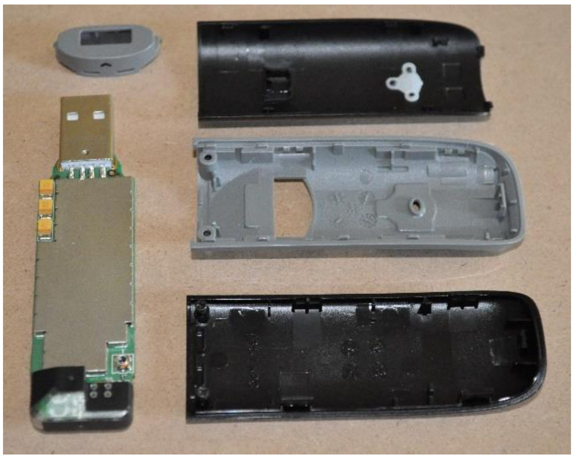
7. Hold the connector with the plier through the hole and insert it in the wall charge plug.



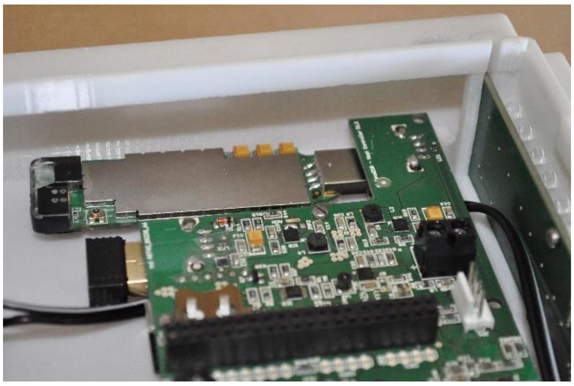
8. Connect the Host PCB to the Control Panel.



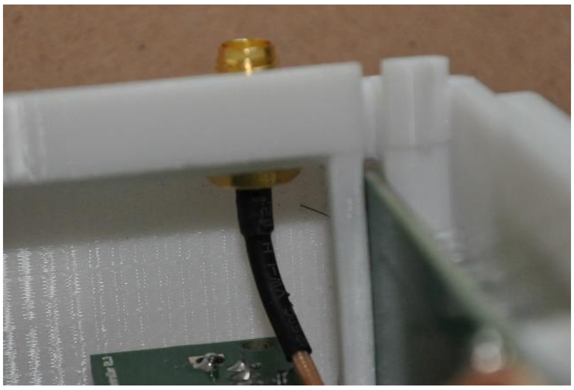
9. Fix the Host PCB and the Control Panel using 2 screws:
    - 1 for the Host PCB.
    - 1 for the Control Panel.



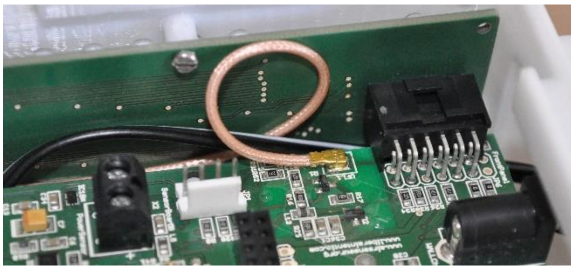
10. Unmount the GPRS dongle cover.
11. Insert a SIM card with internet data subscription



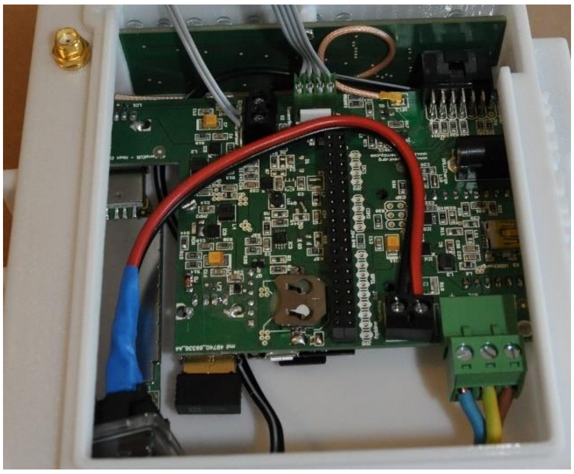
12. Insert the GPRS dongle in the remaining USB plug.



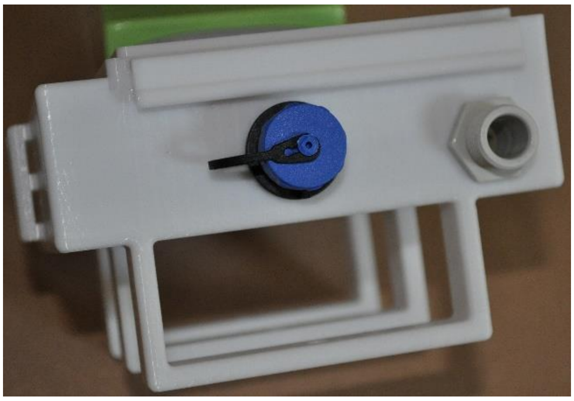
13. Insert the GPS wire in the hole situated above the GPRS USB plug and tight the nut using the bent long nose plier and the plier.

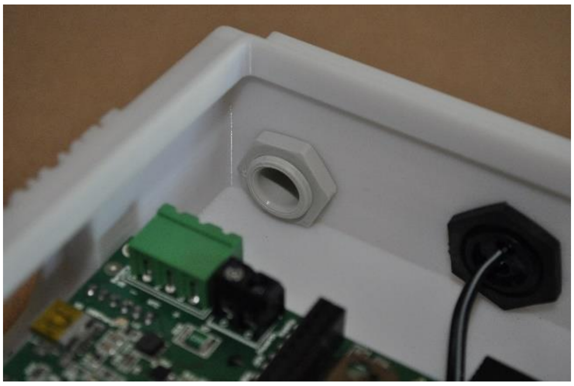


14. Connect the other end of the GPS wire to the Host PCB near the Control Panel plug (see picture)

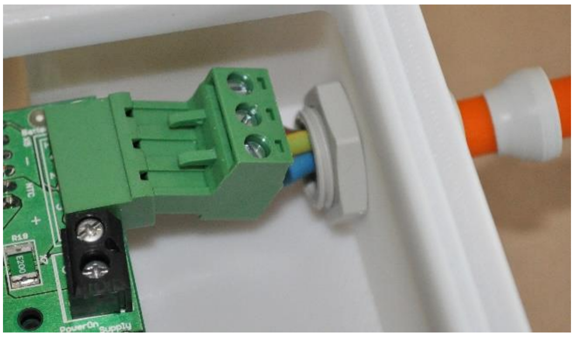


15. Optional: Connect the ON/OFF Switch to the PowerOn plug. In case you do not need the ON/OFF Switch, use a small piece of wire to short circuit the PowerOn plug.

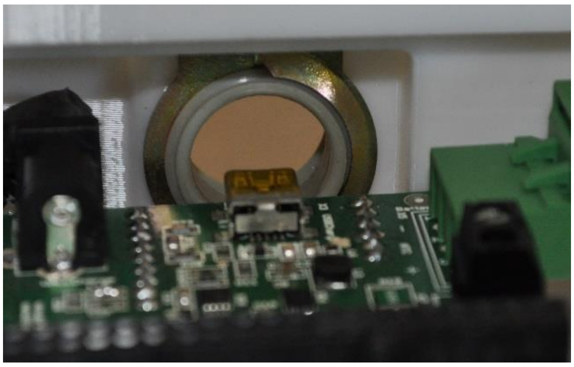




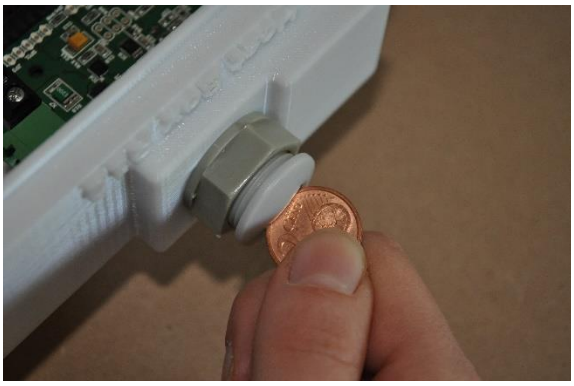
16. Mount the Cable Gland in the hole facing the Battery plug.



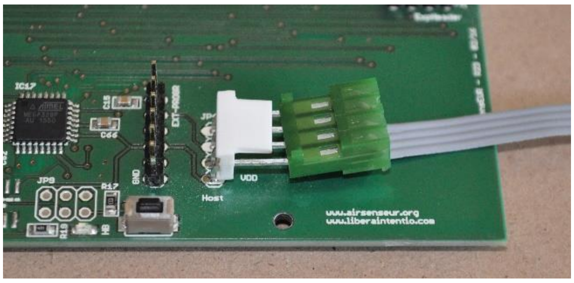
17. Wire the Waterproof battery cable and connect it to the Host PCB.

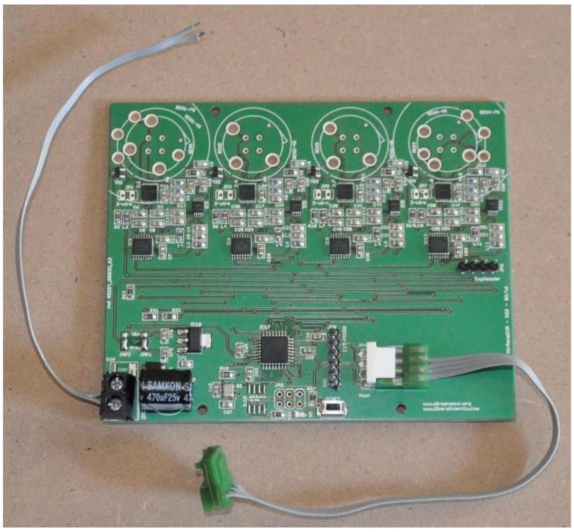


18. Mount the Cable grommet in front of the mini-USB plug of the Host PCB.

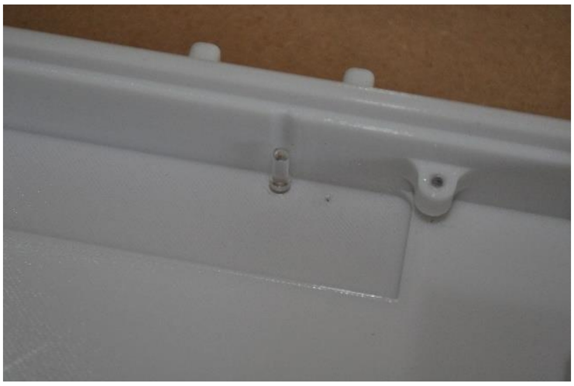


19. Use a 2 cents coin to screw the tap in the Cable grommet.

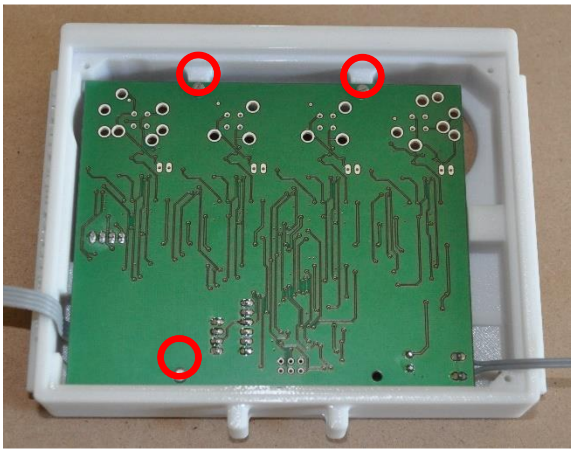




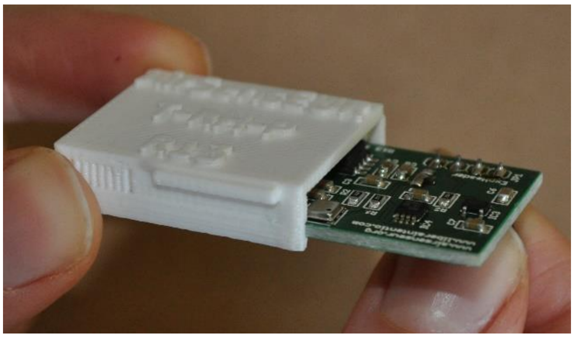
20. Separate the flat wire.
21. Connect the flat wire (green connector) in the JP4 plug of the Shield.
22. Connect the two remaining wires in the black power plug as shown in the picture.
23. Check the connection between the flat wire and the plug using the multimeter.
24. Mount the sensor on the top part of the shield (not shown on the picture).



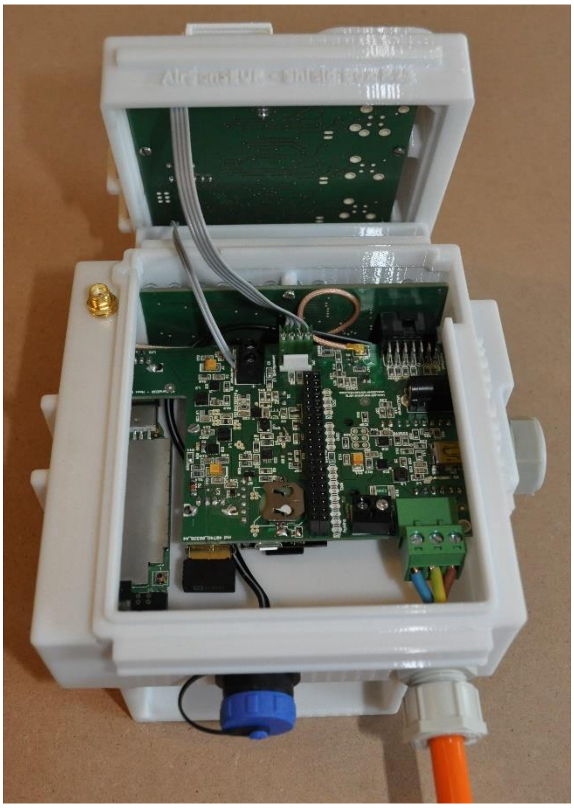
25. Insert the longest Lightpipe in the Shield box



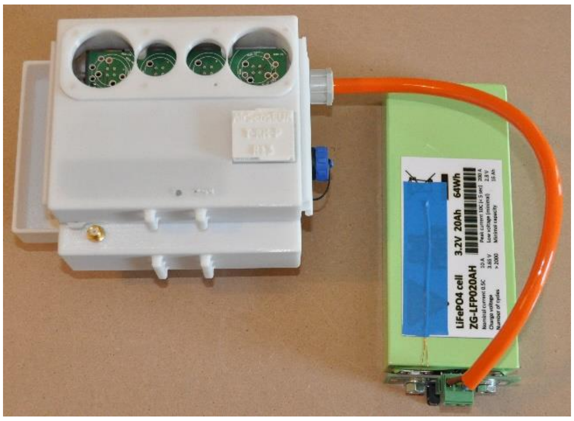
26. Fix the Shield PCB to the box using 3 screws:
    - 2 at the top.
    - 1at the bottom left.



27. Insert the Temperature/Humidity/ Pressure board in its own 3D printed box and install it on the Shield.



28. Connect the small flat wire in the black power plug as shown in the picture.
29. Connect the other end of the flat wire (green connector) in the JP1 plug of the Host.



30. Fit the Shield and Host box together.

4.7 First start of the of the AirSensEUR platform

WARNING – AirSensEUR Host can operate with single cell LiFePO4. Even if the LiFePO4 technology is safer than other Lithium based batteries, cautions should be taken when using high capacity LiFePO4 batteries. Please carefully read this document before you connect batteries to the AirSensEUR Host and start to use it.

DO NOT USE batteries based on different technology than LiFePO4!

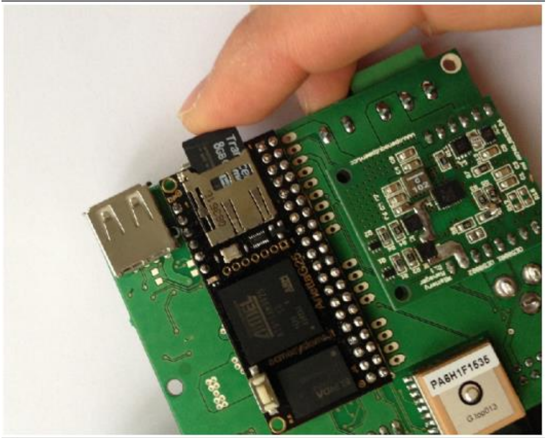
DO NOT CHARGE LiFePO4 batteries with battery chargers not compatible with LiFePO4 technology!

DO NOT SHORT battery terminals and/or DO NOT REMOVE the protection fuse installed on the battery holder. If needed, the fuse should be replaced with a new item matching the same electrical characteristics.

DO NOT REVERSE battery connections to AirSensEUR Host.

DO NOT USE connection cables longer than 30 cm and with section less than 0,75 mmq to connect the battery to the AirSensEUR Host.

DO NOT MODIFY AirSensEUR battery charger voltage and current unless you know what you're doing.

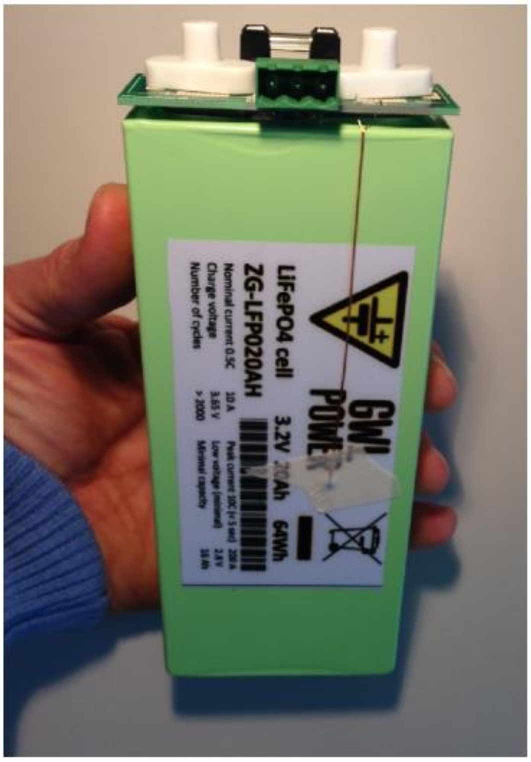


1. Make sure that the microSD and dongles are installed in the host

Install the provided microSD in the microSD holder located on the back of the host PCB. Install the provided WiFi dongle in the near USB connector. The GPRS dongle can be installed in the remaining type-A USB connector located near the control panel.

2. Install LiFePO4 battery

Remove the protection caps on the battery and install the battery holder PCB through the provided bolts and whashers (two on the bolts side, two on the back side). The red terminal on the battery, also labelled with a + sign on the battery case, should be matched with + sign on the PCB.



PAY ATTENTION TO POLARITY!

REVERSING BATTERY WILL PERMANENTLY DAMAGE AIRSENSEUR.

With adhesive tape, firmly block the small temperature sensor to the battery.

NOTE: When purchasing from liberaIntentio s. r. l., the provided battery is already charged following the 1st charge supplier recommendations and is ready to be used. No other actions are required. The first charge shall be carried out with 3.8 V while all following battery charges are carried out at 3.6 V.

3. Turn on the AirSensEUR Host

If installed, turn on the power switch. Press the pushbutton P2, located on the front control panel, labelled “POWERON” and hold for half a second. In a few seconds, the LED3 in the control panel should start blinkink with a 50/50 pattern thus identifying the booting phase. Please note that, for the very first time the Host is turned on, this procedure could take some minutes because the system should calculate some unique cryptographic keys associated to the Secure Shell services. As soon as the process terminates, and the system is ready to run, the LED 3 will blink with an “heartbeat” like pattern.

4. Install the CR2012 battery

AirSensEUR Host implements a Real Time Clock. This service should be able to operate even when the board is not powered up and/or is left unconnected to the main LiFePO4 battery or power supply. For that reason, a small CR2012 battery should be installed in the holder located on the microSD opposite side. This needs to be done when the system is up and running. Take the provided CS2012 battery and insert it by gently slide-in from the holder side. Be sure to match the + pole as labeled on the battery holder (+ up side).

5. Start/stop the WiFi host

To start/stop the WiFi interface, press the P9 (USER2) pushbutton located on the control panel. The LED8 will start blinking for a few seconds, then will be full on indicating the WiFi interface is active. Press P9 to shut down the WiFi interface each time is required. When the WiFi interface is on, it will be visible on the list of available WiFi connection with the “AirSensEUR” name. The password required to pair to this interface is “AirSensEUR”. AirSensEUR default Host IP is set to 192.168.100.1

6. Setup current time

Current time is constantly updated through information retrieved by GPS and/or GPRS. For the very first time, it is useful to manually set-up the current time. This can be done through a Secure Shell connection to the board (in the future this will be simplified and implemented in a different way).

Provided the WiFi interface is on and a PC/Mac is connected to the AirSensEUR WiFi network, from a Linux/MAC shell, issue the following command: *ssh root@192.168.100.1* (PC users should refer to the free PUTTY SSH client and specify 192.168.100.1 in the host IP)

Answer “yes” to the warning message shown in the shell, then type the ‘*password’* password. A message should confirm you're successfully entered in the AirSensEUR command line shell.

Issue the command: *date MMDDhhmmYYYY*

where: MM is the current month, DD is the current day, hh and mm are, respectively, hours and minutes to be set. YYYY is the complete year. When done, type *exit* to exit from the command shell.

7. Accessing to the Web Manager Console and configuring GPRS

Provided that you have the WiFi connection working and a PC/Smartphone/Tablet connected to the AirSensEUR network, point your browser to http://192.168.100.1 A Web panel will be shown. Go to the GPRS Config option on the top menu bar.

In the initialization panel shown, identify the string labelled with “Init3” and replace the contents with something valid for your GPRS provider. We're not able to provide full initialization strings for all international providers but you can find it on Internet by googling for *wvdial configuration* option for your operator. For some operators, different other keys may be modified/added. Please refer also to the wvDial on the Linux man page: http://linux.die.net/man/5/wvdial.conf. Some valid examples are:

H3G in Italy: Init3 = AT+CGDCONT=1,"ip","tre.it",,0,0

Vodafone, in Italy: Init3 = AT+CGDCONT=1,"ip","web.omnitel.it"

8. Other useful configuration options

AirSensEUR Open Platform is a work-in-progress system. Lot of configuration options are available through configuration files that can be modified through an SSH connection. AirSensEUR runs over a standard Debian Wheezy distribution and all AirSensEUR related configuration files and script are stored in the microSD as shown in the following table.

Function / Subsystem	**Path**
AirSensEUR services configuration	/usr/local/etc
AirSensEUR scripts and JAVA binaries	/usr/local/airsenseur
AirSensEUR Logs	/var/log

Most important processes and services configuration files are reported in the following table

Process / Function	**File**
AirSensEURHost (read Shield and publish data)	/usr/local/etc/sensor.properties
AirSensEURDataAggregator (aggregate data from GPS, Shield and store in a local database)	/usr/local/etc/aggregator.properties
AirSensEURDataPush (read data from local database and pushes to external servers)	/usr/local/etc/influxpush.properties (InfluxDB mode) /usr/loca/etc/datapushsosdb.properties (52North SOS mode)
Data Push main script (called by a cron job)	/urs/local/airsenseur/runDataPush

For more information on that, please refer to
https://webgate.ec.europa.eu/CITnet/stash/projects/AIRSENSEUR or www.airsenseur.org.

9. Control panel LEDs and pushbutton assignment

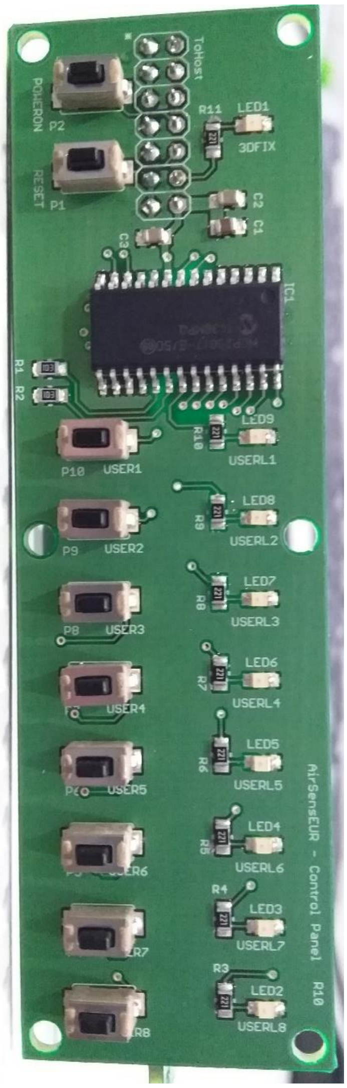
Here is the current control panel LEDs and pushbutton assignment valid for current software release. Please note that the assignment can be changed in next software releases.

The AirSensEUR Host panel implements 8 software controllable LEDs and pushbuttons plus one LED and two pushbuttons statically assigned to specific functions.

Statically assigned functions are related to:

- LED1: 3DFix. This LED is directly connected to the GPS module. When off it indicates that the GPS module is not powered; when blinking in a 50/50 pattern it indicates that the GPS module is powered on but no valid 3DFix is available. As soon as the 3DFix is available the LED will flash periodically with a short blinks.
- P2: POWERON. This pushbutton handles the main board power status. When in off status, pressing for more than a second will power up the Host; when in on status, pressing for more than a second will start a graceful shutdown of the Host.
- P1: RESET. This pushbutton forces a CPU reset.

Software controllable LEDs and pushbuttons are operational only when the Linux kernel has been started. Being software based, their functionality can be modified by kernel parameters and/or bash shell scripts. Some functions were already defined:



- P10 and LED9: Sampling on/off. Pressing the button will start/stop the sampling process. This involves turning on/off the on-board GPS module. The LED blinks when the operation is in progress but is steady on when the sampling process has been started; off when no sampling is running
- P9 and LED8: WiFi Access Point functionality on/off. Pressing the button will start/stop the Access Point WiFi functionality. The LED blinks when the operation is in progress but is steady on when the WiFi Access Point functionality is turned on (the Host is broadcasting their name and ready to accept clients connections).
- P8 and LED7: Data synchronization process scheduler on/off. Pressing the button will enable/disable the data synchronization scheduler. When the LED is on, the data synchronization process is triggered based on the scheduled program. When the LED is off, the scheduler is disabled i.e. no synchronization with external server will be done. The LED blinks when the synchronization is taking place.
- LED3: CPU Heartbeat. This LED blinks in a 50/50 pattern when the system is in the boot/shutdown status. The LED blinks in a heartbeat like pattern when the system is ready and operating. The pulse frequency is proportional to the CPU load status.
- LED2: Battery charge status. The LED is off when Host is powered by battery. The LED flashes when the battery is charging but is steady on when the battery is charged.
- P3: Stealth mode. Pressing this pushbutton toggles the stealth mode. When the stealth mode is on, all software configurable LEDs (except for the battery charge status) are forced off thus reducing the battery power consumption. Exiting from stealth mode restore the software configurable LEDs status.

For up to date information and customization settings, please visit www.airsenseur.org and the AirSensEUR channel on YouTube.

10. Switch off and Install sensors

- At this point you can turn OFF the data acquisition pressing button P10 or even the whole process pressing P2.
- You can now install the sensors. Please make sure that you take off any clip placed on the sensor electrodes before inserting the sensors on the socket of the AirSensEUR shield.
- A typical setup of the sensor with the AlphaSense NO2-B43F and NO-B4 sensors and for the Membrapor O3/M-5 and CO/MF-200 sensors will look as in the photo below:

5 Software

The AirSensEUR Host applications run on a customized Linux Wheezy Debian distribution. The Linux file system is stored in the micro SD connected to the CPU module, together with a couple of external files needed to properly boot-up the system. Setting up a Linux distribution is not an easy task and the details are out of the scope of this document. The AirSensEUR framework provides a set of build scripts and configuration files that simplify the process. A convenient micro SD image is also available for easier start-up with AirSensEUR Host; please refer to https://webgate.ec.europa.eu/CITnet/stash/projects/AIRSENSEUR or www.airsenseur.org for more information. People looking for build up a Linux image can go through the details provided in next chapters.

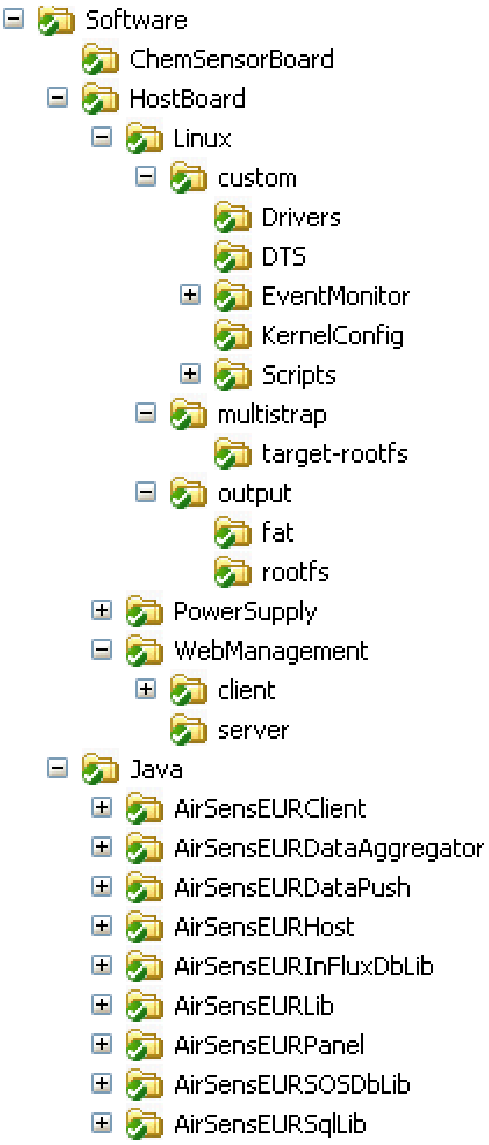
5.1 Setting up the cross-compiler machine

AirSensEUR is based on ARM9 CPU architecture thus imposing to have a compiler for this processor. Common Linux distributions are easily available for i386 or x64 architectures but binaries are not compatible with AirSensEUR. In order to generate Linux kernel image, driver modules and other applications binaries, a cross-compiler is needed. A cross-compiler host is a custom configured, dedicated machine running on a standard i386 or x64 desktop PC that is able to compile and execute/emulate code for different architectures. We suggest to setup a virtual machine starting from a fresh Ubuntu i386 distribution. In the current implementation, we started with a new ubuntu-14.10-desktop-i386.iso image.

As soon as the Ubuntu Linux is set-up and running, it is necessary to install a set of optional packages by issuing the following commands on a terminal window (answer “Yes” each time it is required).

```
sudo apt-get install multistrap dpkg-cross qemu qemu-user-static

sudo apt-get install binfmt-support

sudo apt-get install libc6-armel-cross libc6-dev-armel-cross

sudo apt-get install libncurses5-dev dosfstools wget

sudo apt-get install binutils-arm-linux-gnueabi

sudo apt-get install gcc-arm-linux-gnueabi g++-arm-linux-gnueabi
```

It is required to download and extract the AirSensEUR reference source tree package in a local folder. The package tree has the structure reported in the following picture:



```
⊟ Software
      ChemSensorBoard
  ⊟ HostBoard
    ⊟ Linux
      ⊟ custom
            Drivers
            DTS
        ⊞ EventMonitor
            KernelConfig
        ⊞ Scripts
      ⊟ multistrap
            target-rootfs
      ⊟ output
            fat
            rootfs
    ⊞ PowerSupply
    ⊟ WebManagement
      ⊞ client
          server
  ⊟ Java
    ⊞ AirSensEURClient
    ⊞ AirSensEURDataAggregator
    ⊞ AirSensEURDataPush
    ⊞ AirSensEURHost
    ⊞ AirSensEURInFluxDbLib
    ⊞ AirSensEURLib
    ⊞ AirSensEURPanel
    ⊞ AirSensEURSOSDbLib
    ⊞ AirSensEURSqlLib
```

The Linux subfolder contains patches and scripts required to generate a Debian based custom distribution for the AirSensEUR Host. **In the next sections that describe the successive configuration steps, a command shell prompt that is located on the current local folder will be used**. At the end of the process, the output subfolder will contain the results to be copied in the micro SD.

The Java subfolder, instead, contains all the sources needed to build the Java based applications that will run on the Host (except for the AirSensEURPanel that can be run on a PC to setup and configure the AirSensEUR Shield). The WebManagement folder contains either server than client side sources for the Web Manager Console tool.

5.2 Compiling the bootloader

The bootloader is a small program that runs at the very beginning of the boot process. It's responsible for loading the compressed kernel image and the kernel configuration file from the micro SD boot partition, place them in a well-known RAM memory address, explode the kernel then start their execution. The bootloader is strictly connected to the hardware where it should run and, for setting up the AirSensEUR Host, is mandatory to download the code for the ATMEL ARM9 CPU and to apply the patches provided by the Arietta G25 module producer.

This can be easily done by following the procedure available at www.acmesystems.it/compile_at91bootstrap and summarized hereafter.

From a terminal window, issue the sequence of following commands:

```
wget https://github.com/linux4sam/at91bootstrap/archive/v3.7.zip

unzip v3.7.zip

cd at91bootstrap-3.7

wget
https://raw.githubusercontent.com/AcmeSystems/acmepatches/master/at91boots
trap-3.7.patch

patch -p1 < at91bootstrap-3.7.patch

make arietta-256m_defconfig

cd ..

./buildat91bootstrap
```

The last script available on the AirSensEUR reference package, cross-compiles the bootloader and copies it to the proper output folder.

5.3 Compiling the kernel and the configuration file tree

For the purposes of this report we used a Linux 4.2.6 kernel version. Like the bootloader, the kernel binaries are very much connected to the hardware architecture details and a set of patches should be downloaded from the Arietta G25 module producer. Other patches are connected to the AirSensEUR Host details and are available on the reference package.

The kernel should be cross-compiled from sources available at the kernel.org website. Open a command window into the Hostboard/Linux folder generated by the AirSensEUR package tree. Download and deflate the kernel sources with the following command:

```
wget https://www.kernel.org/pub/linux/kernel/v4.x/linux-4.2.6.tar.xz

tar -xvf linux-4.2.6.tar.xz
```

To target the sources for the Arietta module download and apply a patch file:

```
cd linux-4.2.6
wget https://raw.githubusercontent.com/AcmeSystems/acmepatches/master/linux-4.2.6.patch
patch -p1 < linux-4.2.6.patch
cd ..
```

The last needed steps can easily be done by mean of the scripts available in the AirSensEUR package tree:

```
./preparekernel
./builddtb
./buildkernel
```

Those scripts apply the patches required to customize the kernel for the AirSensEUR Host and starts a compile process. At the end of the process the kernel image and binary configuration tree are copied in the output/fat folder; kernel modules are copied in the /output/rootfs/lib folder ready to be copied into the target micro SD, as explained on next paragraphs.

5.4 Compiling C/C++ utilities

The AirSensEUR operating panel requires a mix of kernel modules and one user application program. Since kernel modules have been already compiled in the previous paragraph it's now time to cross-compile the EventMonitor application program.

The application sources are located in the HostBoard/Linux/custom/EventMonitor folder from the AirSensEUR reference package. It's time to create a folder suitable to contain all AirSensEUR Host executables, and to compile the EventMonitor application. Open a command window in the Linux folder and issue the following statements:

```
mkdir hostapps
cd custom/EventMonitor
arm-linux-gnueabi-gcc -o eventmonitor main.c
cp eventmonitor ../../hostapps/
cd ../..
```

The second application is optional and is used by AirSensEUR Host to synchronize the built-in NTP server with a precise Pulse-Per-Second (PPS) signal generated by the GPS when a 3DFIX event is found. This allows precise real time clock synchronization when no network connection is available for long periods. The application, written by Folkert van Heusden (folkert@vanheusden.com) and released under GPLv2, can be downloaded from Github and compiled by the following commands:

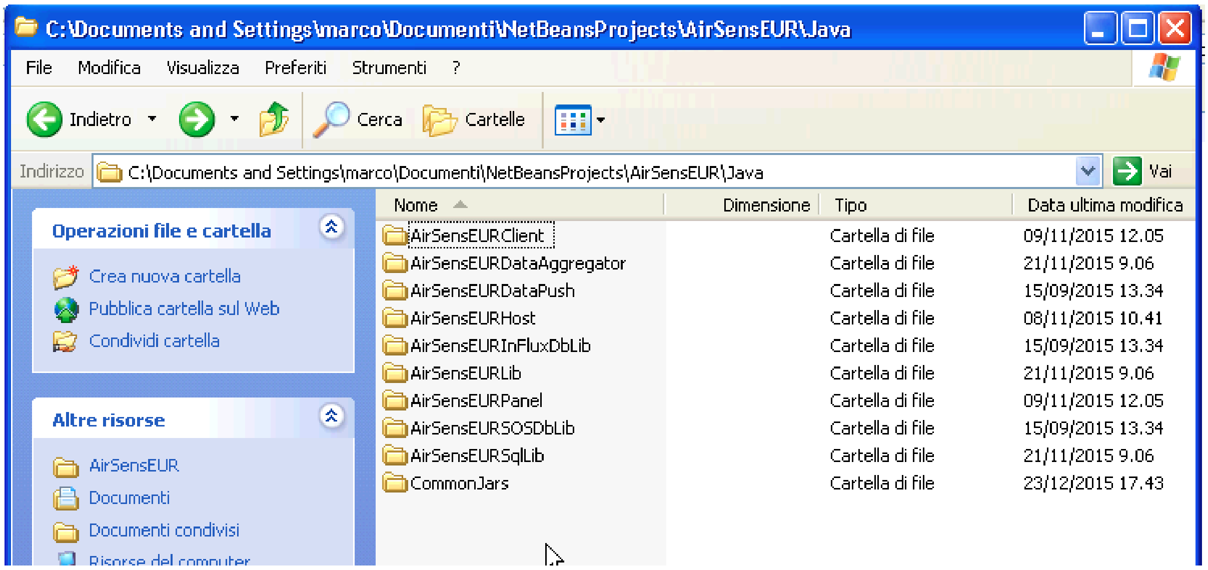
```
wget https://github.com/flok99/rpi_gpio_ntpd/archive/master.zip
unzip master.zip
cd rpi_gpio_ntpd-master
make CC=arm-linux-gnueabi-gcc
cp rpi_gpio_ntp ../hostapps/
```

cd ../

5.5 JAVA

The main AirSensEUR Host applications are Java based. They can be built on whatever machine with Netbeans IDE, and then copied to the development machine into the hostapps folder. For the scope of this paper we address to two applications: AirSensEURHost and AirSensEURDataAggregator.

Some dependencies should be satisfied to properly compile those applications: all required libraries should be downloaded and extracted on the folder CommonJars that should be created in the Java directory, as shown in the picture below.



C:\Documents and Settings\marco\Documenti\NetBeansProjects\AirSensEUR\Java
File
Modifica
Visualizza
Preferiti
Strumenti
?
Indietro
Cerca
Cartelle
Indirizzo
C:\Documents and Settings\marco\Documenti\NetBeansProjects\AirSensEUR\Java
Vai
Operazioni file e cartella
Crea nuova cartella
Pubblica cartella sul Web
Condividi cartella
Altre risorse
AirSensEUR
Documenti
Documenti condivisi
Risorse del computer

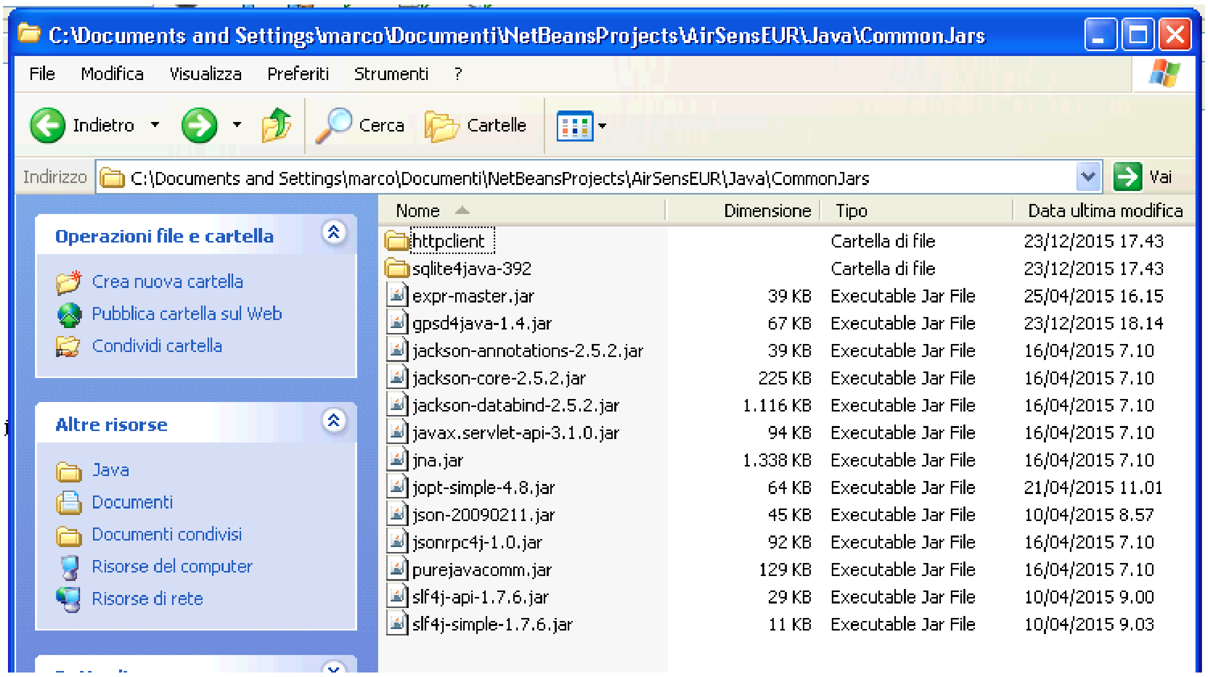
Nome	Dimensione	Tipo	Data ultima modifica
AirSensEURClient		Cartella di file	09/11/2015 12.05
AirSensEURDataAggregator		Cartella di file	21/11/2015 9.06
AirSensEURDataPush		Cartella di file	15/09/2015 13.34
AirSensEURHost		Cartella di file	08/11/2015 10.41
AirSensEURInFluxDbLib		Cartella di file	15/09/2015 13.34
AirSensEURLib		Cartella di file	21/11/2015 9.06
AirSensEURPanel		Cartella di file	09/11/2015 12.05
AirSensEURSOSDbLib		Cartella di file	15/09/2015 13.34
AirSensEURSqlLib		Cartella di file	21/11/2015 9.06
CommonJars		Cartella di file	23/12/2015 17.43

Libraries can be downloaded for from the Maven repository (http://search.maven.org) or directly from their developer website. For best results, libraries versions, if specified, should be matched. jar files can easily be generated with Netbeans if only source code is available.

The list of required libraries includes:

- expr-master.jar: https://github.com/darius/expr
- gpsd4djava-1.4.jar: http://taimos.github.io/GPSd4Java/
- jackson-annotations-2.5.2.jar: (Maven) https://github.com/FasterXML/jackson-annotations
- jackson-core-2.5.2.jar: (Maven) https://github.com/FasterXML/jackson-core
- jackson-databind-2.5.2.jar: (Maven) https://github.com/FasterXML/jackson-databind
- javax.servlet-api-3.1.0.jar: (Maven) https://java.net/projects/servlet-spec/
- jna-3.4.0.jar: https://github.com/java-native-access/jna
- jopt-simple-4.8.jar: (Maven) https://pholser.github.io/jopt-simple/index.html
- json-20090211.jar: (Maven) http://mvnrepository.com/artifact/org.json/json/20090211
- jsonrpc4j-1.0.jar: (Maven) https://github.com/briandilley/jsonrpc4j
- purejavacomm-1.0.0.jar: https://github.com/nyholku/purejavacomm
- slf4j-api-1.7.6.jar: (Maven) http://www.slf4j.org
- slf4j-simple-1.7.6.jar: (Maven) http://www.slf4j.org
- sqlite4java.jar: (Maven). Refer to version 1.0.392 from com.almworks.sqlite4java https://code.google.com/archive/p/sqlite4java/
- httpclient-4.4.1.jar , httpcore-4.4.1.jar , http://hc.apache.org
- commons-logging-1.2.jar: (Maven) https://commons.apache.org/proper/commons-logging/

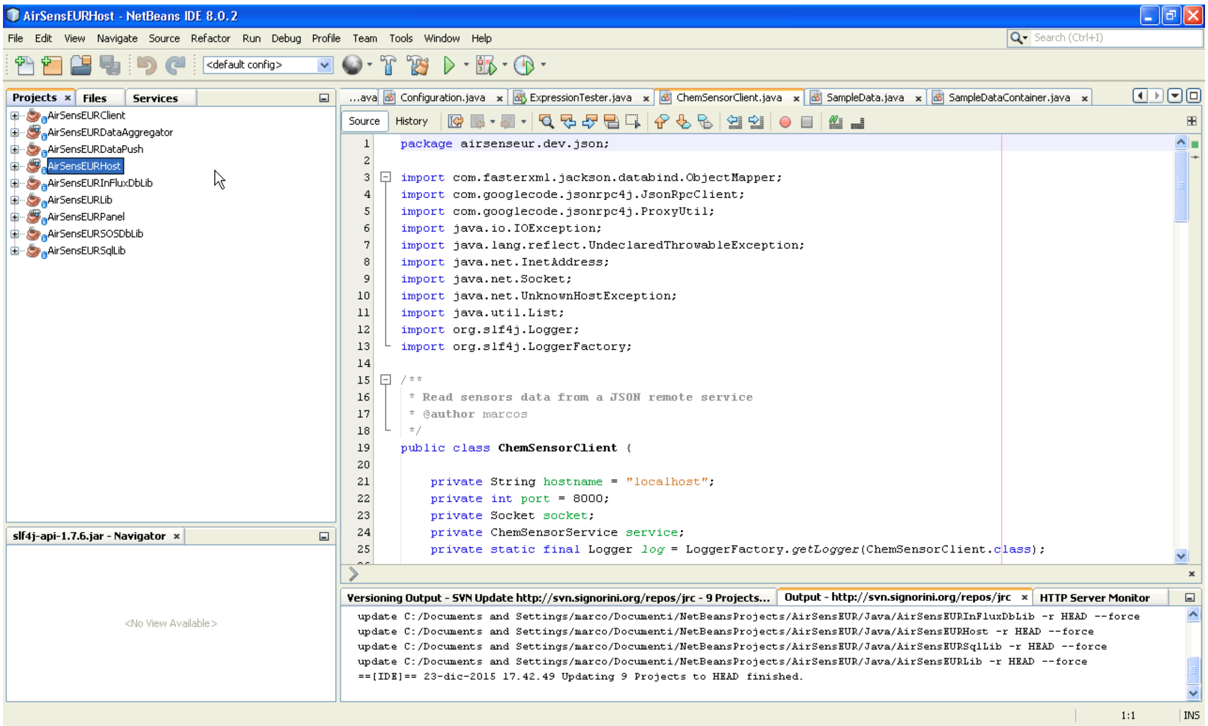
We expect to have all the jar files in the CommonJars folder except for the sqlite4java.jar that should be placed in the sqlive4java-392 subfolder and the httpclient-4.4.1.jar, httpcore-4.4.1.jar and commons-logging-1.2.jar that should be grouped together into the httpclient subfolder. The expected CommonJars folder should be like the one shown in the picture below.



Indirizzo: C:\Documents and Settings\marco\Documenti\NetBeansProjects\AirSensEUR\Java\CommonJars

Nome	Dimensione	Tipo	Data ultima modifica
httpclient		Cartella di file	23/12/2015 17.43
sqlite4java-392		Cartella di file	23/12/2015 17.43
expr-master.jar	39 KB	Executable Jar File	25/04/2015 16.15
gpsd4java-1.4.jar	67 KB	Executable Jar File	23/12/2015 18.14
jackson-annotations-2.5.2.jar	39 KB	Executable Jar File	16/04/2015 7.10
jackson-core-2.5.2.jar	225 KB	Executable Jar File	16/04/2015 7.10
jackson-databind-2.5.2.jar	1.116 KB	Executable Jar File	16/04/2015 7.10
javax.servlet-api-3.1.0.jar	94 KB	Executable Jar File	16/04/2015 7.10
jna.jar	1.338 KB	Executable Jar File	16/04/2015 7.10
jopt-simple-4.8.jar	64 KB	Executable Jar File	21/04/2015 11.01
json-20090211.jar	45 KB	Executable Jar File	10/04/2015 8.57
jsonrpc4j-1.0.jar	92 KB	Executable Jar File	16/04/2015 7.10
purejavacomm.jar	129 KB	Executable Jar File	16/04/2015 7.10
slf4j-api-1.7.6.jar	29 KB	Executable Jar File	10/04/2015 9.00
slf4j-simple-1.7.6.jar	11 KB	Executable Jar File	10/04/2015 9.03

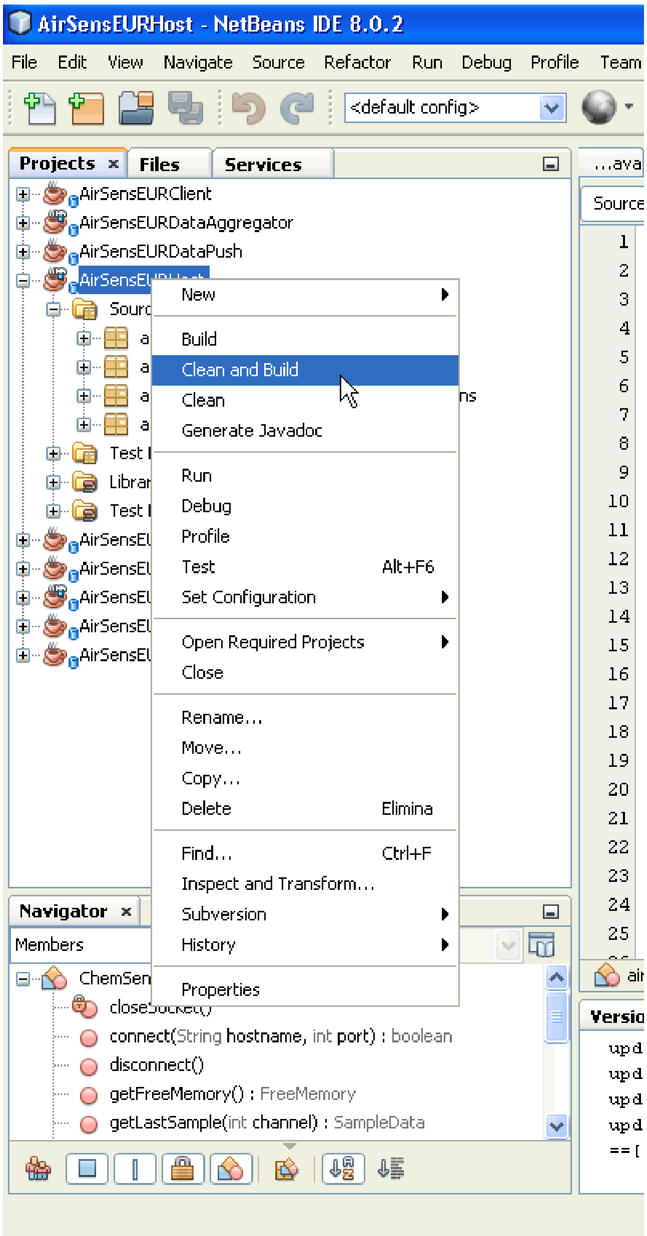
The AirSensEUR reference source package contains a set of Netbeans projects that can be used to compile the Host applications. They are all stored in the Java subfolder. Opening all those projects with Netbeans IDE will result in something similar to what is reported in the picture below.



```
package airsenseur.dev.json;

import com.fasterxml.jackson.databind.ObjectMapper;
import com.googlecode.jsonrpc4j.JsonRpcClient;
import com.googlecode.jsonrpc4j.ProxyUtil;
import java.io.IOException;
import java.lang.reflect.UndeclaredThrowableException;
import java.net.InetAddress;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.List;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * Read sensors data from a JSON remote service
 * @author marcos
 */
public class ChemSensorClient {

    private String hostname = "localhost";
    private int port = 8000;
    private Socket socket;
    private ChemSensorService service;
    private static final Logger log = LoggerFactory.getLogger(ChemSensorClient.class);
```

```
update C:/Documents and Settings/marco/Documenti/NetBeansProjects/AirSensEUR/Java/AirSensEURInFluxDbLib -r HEAD --force
update C:/Documents and Settings/marco/Documenti/NetBeansProjects/AirSensEUR/Java/AirSensEURHost -r HEAD --force
update C:/Documents and Settings/marco/Documenti/NetBeansProjects/AirSensEUR/Java/AirSensEURSqlLib -r HEAD --force
update C:/Documents and Settings/marco/Documenti/NetBeansProjects/AirSensEUR/Java/AirSensEURLib -r HEAD --force
==[IDE]== 23-dic-2015 17.42.49 Updating 9 Projects to HEAD finished.
```

Right click on the AirSensEURHost project listed on the left bar, then select "Clean and Build" to start the building process.



The build process stores the result binaries, together with the required libraries, in the dist subfolder located on the project folder.

Copy the airsenseurhost file, located into the AirSensEURHost folder, to the dist subfolder. Copy the dist directory in the hostapps folder, located in the Linux working directory, then rename it to AirSensEURHost.

Repeat the same procedure for the AirSensEURDataAggregator project (compile, copy the airsenseurdataaggregator file into the dist folder, move the dist folder in the hostapps then rename it to AirSensEURDataAggregator). To enable Java to use sqlite on the ARM CPU, a library wrapper is needed. This library can be installed by simply downloading the binary file from http://sqlite4java.googlecode.com/files/libsqlite4java-linux-arm.so then moving the libsqlite4java-linux-arm.so file into hostapps/AirSensEURDataAggregator/lib subfolder.

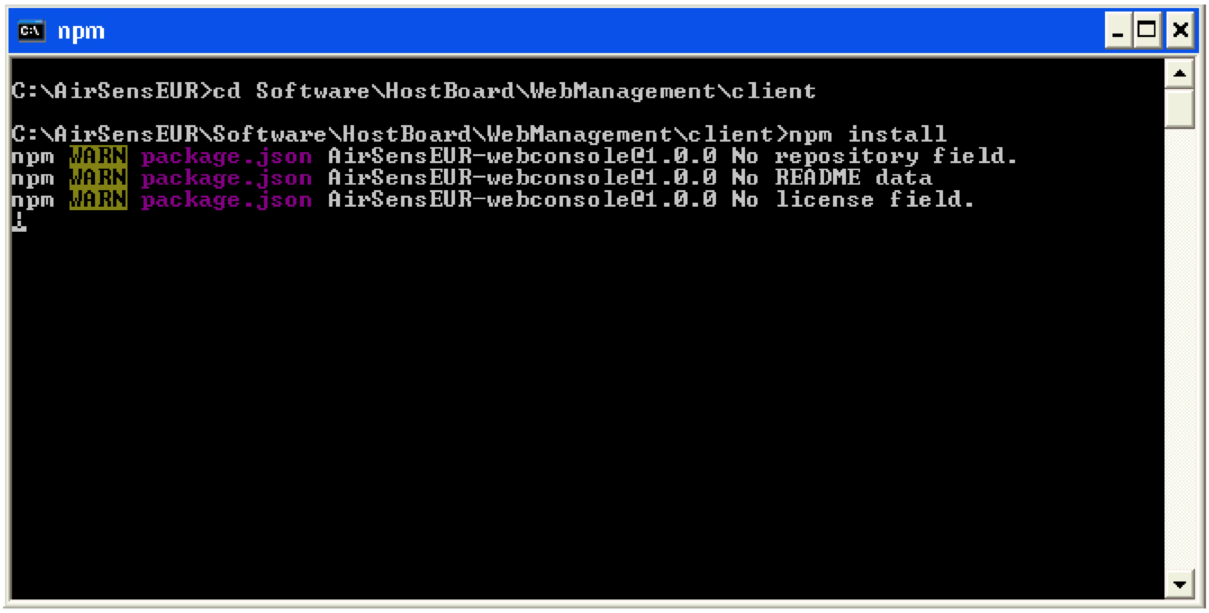
To compile the AirSensEURDataPush project, right click on the AirSensEURDataPush project listed on the left bar, then select "Clean and Build" to start the building process.

Move the dist folder in the hostapps, then rename it to AirSensEURDataPush. To enable Java to use sqlite on the ARM CPI, copy the libsqlite4java-linux-arm.so file downloaded on the previous step to the hostapps/AirSensEURDataPush/lib subfolder too.

5.6 Build the Web Management Console application

We expect to have NodeJS already installed on the production machine. If you need to install, download it from the NodeJS website https://nodejs.org/en/ and install in your development machine.

Open a command shell and change the working folder into WebManagement/client subfolder you can found in the AirSensEUR package tree.



```
C:\AirSensEUR>cd Software\HostBoard\WebManagement\client

C:\AirSensEUR\Software\HostBoard\WebManagement\client>npm install
npm WARN package.json AirSensEUR-webconsole@1.0.0 No repository field.
npm WARN package.json AirSensEUR-webconsole@1.0.0 No README data
npm WARN package.json AirSensEUR-webconsole@1.0.0 No license field.
```

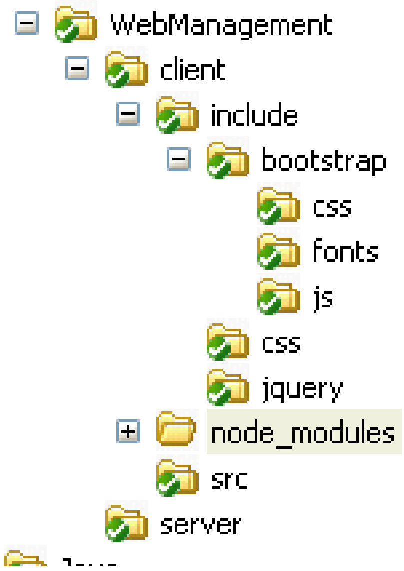
Install the project dependencies by issuing the following command:

```
npm install
```

Dependencies are downloaded from the Network and installed in a local node_modules folder. As soon as the process terminates, compile the jsReact application with the following command:

```
npm run build
```

This generates the AirSensEURBundle.js file implementing the Web Management Console client side application. It's time to copy some redistributable code from the node_modules. To do this, copy the contents of node_modules/bootstrap/dist folder into the include/bootstrap folder, and the contents of node_modules/jquery/dist folder into the include/jquery folder. You should now have a client folder populated as shown in the following screenshot:



```
⊟ WebManagement
    ⊟ client
        ⊟ include
            ⊟ bootstrap
                css
                fonts
                js
              css
              jquery
        ⊞ node_modules
          src
      server
```

The compiled client side Web Management Console binaries should be copied, together with the related PHP server side files, in a proper location in order to include them in the AirSensEUR Host filesystem.

- Create an AirSensEURWebManager named folder in the HostBoard/Linux/hostapps folder
- Copy the whole contents of HostBoard/WebManagement/client folder (excluding the node_modules folder) into the HostBoard/Linux/hostapps/AirSensEURWebManager folder
- Copy the whole contents of HostBoard/WebManagement/server folder into the HostBoard/Linux/hostapps/AirSensEURWebManager folder.

5.7 Build a Linux root file system

AirSensEUR Host applications run on a Wheezy Debian based custom distribution. Kernel images and user applications are only a piece of the puzzle. To have a working Linux distribution, a root file system tree is needed (rootfs). There are many ways to generate a Linux root file system but, for the scope of this paper, we use the Multistrap tool. Multistrap allows defining a set of application packages to be included in the target file system, thus simplifying the Debian customization process.

The AirSensEUR reference source package contains a set of scripts that generate a custom Linux Wheezy in a very easy way. Those scripts are located in the multistrap folder.

To generate the minimal rootfs, simply run the following command:

```
cd multistrap

./buildrootfs

cd ../
```

The script asks for the development machine root password then starts to download and install a long series of packages.

The process prompts for several questions:

- Use dash as the default system shell? Reply with “No”

- What is the local time zone? Select the most appropriated from the menu list
- Insert the system root password: this will be the AirSensEUR Host root password. Insert the one you prefer (and don't forget it)

At the end of the process, another script merges the kernel images and the rootfs generated files:

```
./copytorootfs
```

This terminates the building process. It's now time to prepare the micro SD to be inserted in the AirSensEUR Host.

5.8 Format and populate the target micro SD

Whatever you opted for building by yourself the whole Linux system, or you downloaded a .tar file from the AirSensEUR website, this last step is needed to properly format and populate a new micro SD to be used by the AirSensEUR Host. In this paper we used a 8GB micro SD but the same procedure applies for memories with different sizes.

Supposing to have a new FAT formatted micro SD card, connect it to the Linux development machine (a USB SD card reader best fits our needs). In seconds you should see a new file window pointing to the SD contents. Unmount (eject) the new device then open a command shell window then type:

```
sudo dmesg
```

You should see a set of traces reporting the SD card has been inserted and identified by something like /dev/sdb. Take note of this path as it is needed to properly refer to the micro SD in the next steps.

We need to create two partitions. The first partition is used by AirSensEUR at the very begin of the boot process to read the bootloader and the kernel; the second partition, instead, will contain the whole root filesystem for the Linux distribution.

In the command window type:

```
sudo fdisk /dev/sdb
```

Type, then, the following sequence of commands:

```
o (enter)
```

this will erase the whole partition set; then:

```
n (enter)

Partition type: p (enter)

Partition number: 1 (enter)

First sector: 2048 (enter)

Last sector: +128M (enter)

t (enter)

6 (enter)
```

This creates a 128MB FAT filesystem for the boot process; then:

```
n (enter)

Partition type: p (enter)

Partition number: 2 (enter)
```

```
First sector: press enter for the default value

Last sector: press enter for the default value
```

This creates a second partition for the root filesystem by filling the overall left space in the micro SD card. Confirm writing the new partitions by pressing w (enter). Fdisk exits. The two partitions shuld be formatted and mounted by issuing the following:

```
sudo mkfs.vfat -n boot /dev/sdb1

sudo mkfs.ext4 -L rootfs /dev/sdb2

sudo mkdir /media/boot

sudo mkdir /media/rootfs

sudo mount /dev/sdb1 /media/boot

sudo mount /dev/sdb2 /media/rootfs
```

It’s now time to copy the bootloader, the kernel and the root filesystem.
The AirSensEUR framework scripts used to cross-compile the kernel and to generate the root filesystem placed all binaries into two folders named fat and rootfs located in the output folder of your Linux working path. These folders should be copied to the micro SD by issuing the following commands:

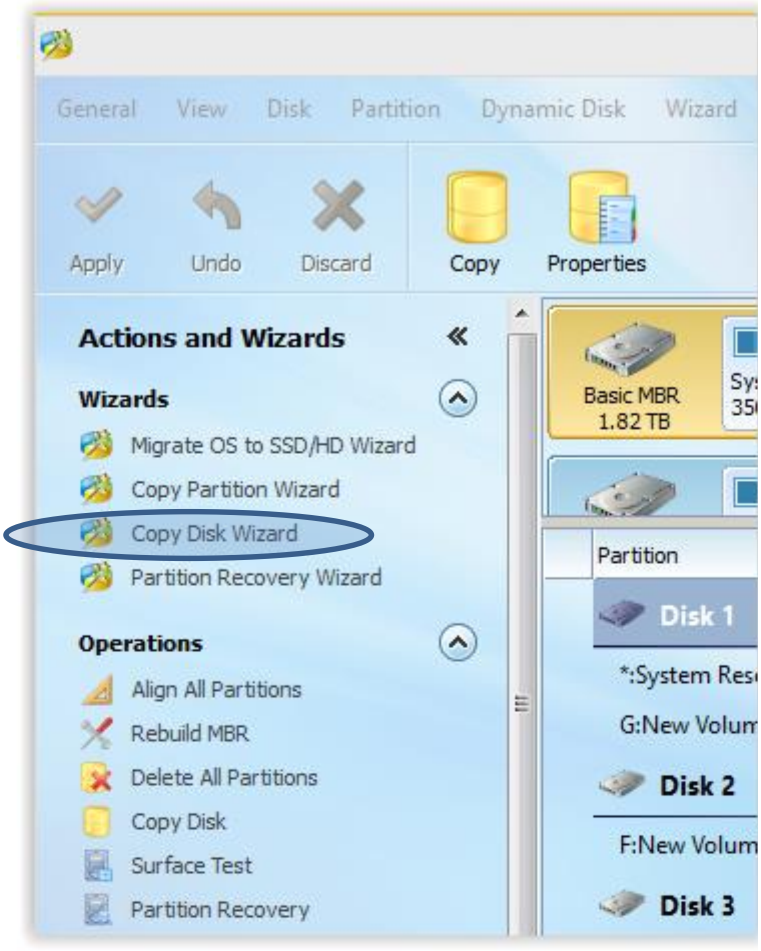
```
sudo cp output/fat/* /media/boot/.
sudo rsync -axHAX --progress output/rootfs/ /media/rootfs/
sudo umount /media/boot
sudo umount /media/rootfs
```

The micro SD is ready to be inserted in the AirSensEUR Host. The very first boot will take some time to create some cryptographic keys and install the Java JVM. The user LED7 flashing with a heartbeat like pattern signals the boot process termination.

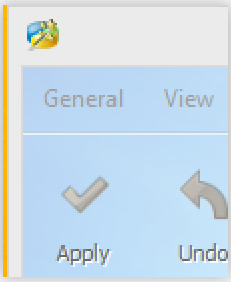
5.9 Alternative cloning of a working SD

As an alternative of the whole process presented in the previous sections, if you already have a working SD, you can clone it as follows:

- Install MiniTool Partition Wizard version 9.0 (http://www.partitionwizard.com/)
- Plug the SD1 to be cloned in your PC
- Plug the SD2 that will receive the SD1 image in your PC
- Launch MiniTool partition wizard with administrative right
- Delete all partitions on SD2 (left menu under Operations “Delete All Partitions”)



-



-

- Select Copy Disk Wizard. In the window “Select Disk to Copy”, select SD1 and in the window “Select target Disk, select SD2. In the last window “Review the changes under Copy options, select 2. Copy partitions without resize and uncheck Force to align partitions to 1MB. Accept the proposed windows until the end of this process
- Click on button apply

6 Running an AirSensEUR

AirSensEUR can operate with or without batteries. When no batteries are installed, it is mandatory to remove the external NTC connected to the battery socket. Batteries should be connected or unconnected only when the Host is not operating, the main power switch is turned off and no power supplies are connected to USB or wall adapter power supply ports.

When operating with no batteries, the AirSensEUR Host can be powered through the wall adapter port but not through the USB charger port.

The main power switch should be used to disconnect the power supply and the battery when maintenance is required on the Host board, thus preventing short circuits that could damage the system.

To power up the AirSensEUR, the main power is switch on using the main power switch (I) then the P2 (POWERON) pushbutton should be pressed for more than a second. After some seconds the LED3 starts blinking in a 50/50 pattern indicating the system is booting the Linux kernel. The system is ready as soon as the LED3 changes its blinking pattern with a heartbeat like sequence.

Pressing P2 (POWERON) pushbutton for more than a second starts a graceful shutdown. This is signalled by LED3 blinking in a 50/50 pattern until the complete shutdown of the system.

The AirSensEUR Host is equipped with a CR1203 Lithium backup battery that preserves the RTC, to speed-up the GPS fix and to provide the required wake up signals to the CPU when powered off. The backup battery should last several months when the system is not powered. Since an efficient battery is necessary for the RTC to properly operate, it should be changed when necessary.

6.1 Start/stop the sampling processes

When the system is up and ready, the sampling process can be started by pressing the P10 pushbutton. The LED9 starts blinking to signal this event. At the same time the GPS module is powered up as confirmed by LED1. The sampling process is handled by two Java applications: AirSensEURHost and AirSensEURDataAggregator plus a Linux daemon that interfaces to the GPS module. The LED9 continues to blink until these processes are running and, when done, it changes to a steady on status. After several seconds the LED located on the AirSensEUR Shield starts blinking thus signalling that new samples are ready to be consumed by the Host.

All samples are stored locally either into a sqlite database then in a set of .csv files located in the /usr/local/airsenseur/AirSensEURDataAggregator folder. These files can be retrieved by removing the micro SD card and installing on a Linux PC or, more conveniently, through a secure copy via a WiFi connection.

Pressing the P10 pushbutton stops the sampling process. This event shuts down the GPS and is signalled by LED9 changing to unlit status.

6.2 Connect to AirSensEUR through WiFi

When a Ralink Technology, Corp. RT5370 based Wireless Adapter is connected to USB_B socket, it can be used in access point mode to connect to WiFi clients like mobile phones, tablets or notebooks. This feature is by default off at power-up but can be turned on/off by mean of the P9 pushbutton. LED8 blinks when the operation is in progress and changes to steady on when the access point feature is ready.

When enabled, the Host is reachable as a WiFi private network named AirSensEUR (default password AirSensEUR). The client should be enabled to receive dynamically an IP address as the AirSensEUR Host acts a DHCP server.

As soon as the connection is established, the client should receive an IP in the 192.168.100.x network. AirSensEUR Host has a default static 192.168.100.1 address.

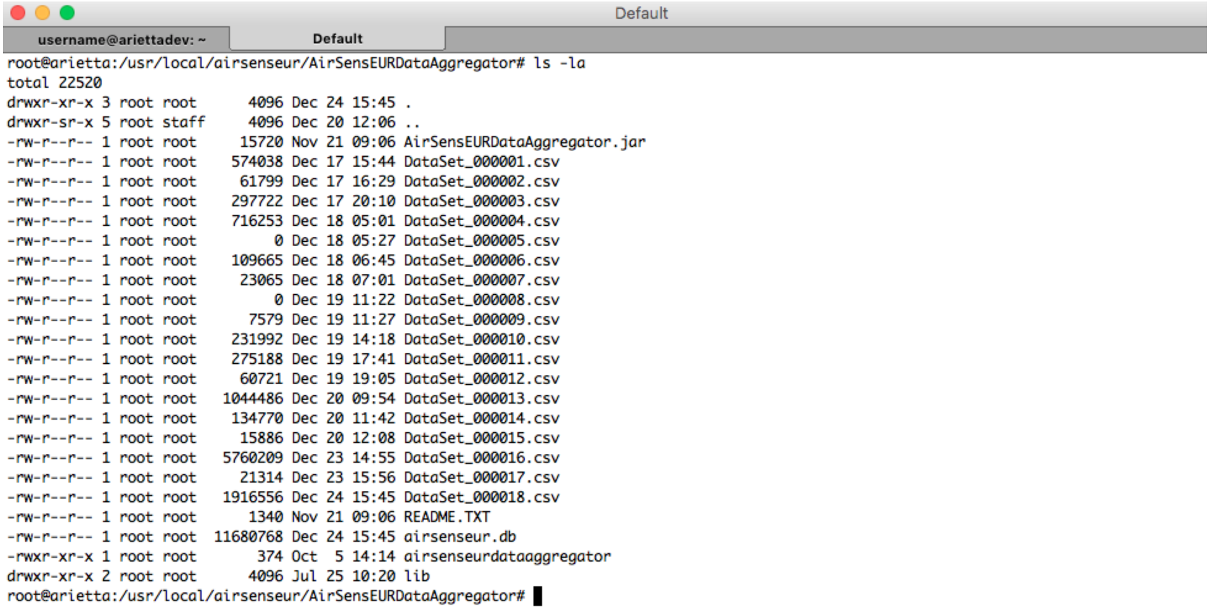
AirSensEUR implements an SSH server to make it possible to open a remote encrypted command shell through any SSH client (for example putty.exe on Windows based machines, or a stock ssh command shell under Linux and Mac OS).

From a command line window try with:

```
ssh root@192.168.100.1

password: the password you choose when building the rootfs
```

root@airsenseur#

The picture below shows, as example, the contents of a remote shell when asking for the list of files presents into the /usr/local/airsenseur/AirSensEURDataAggregator folder.

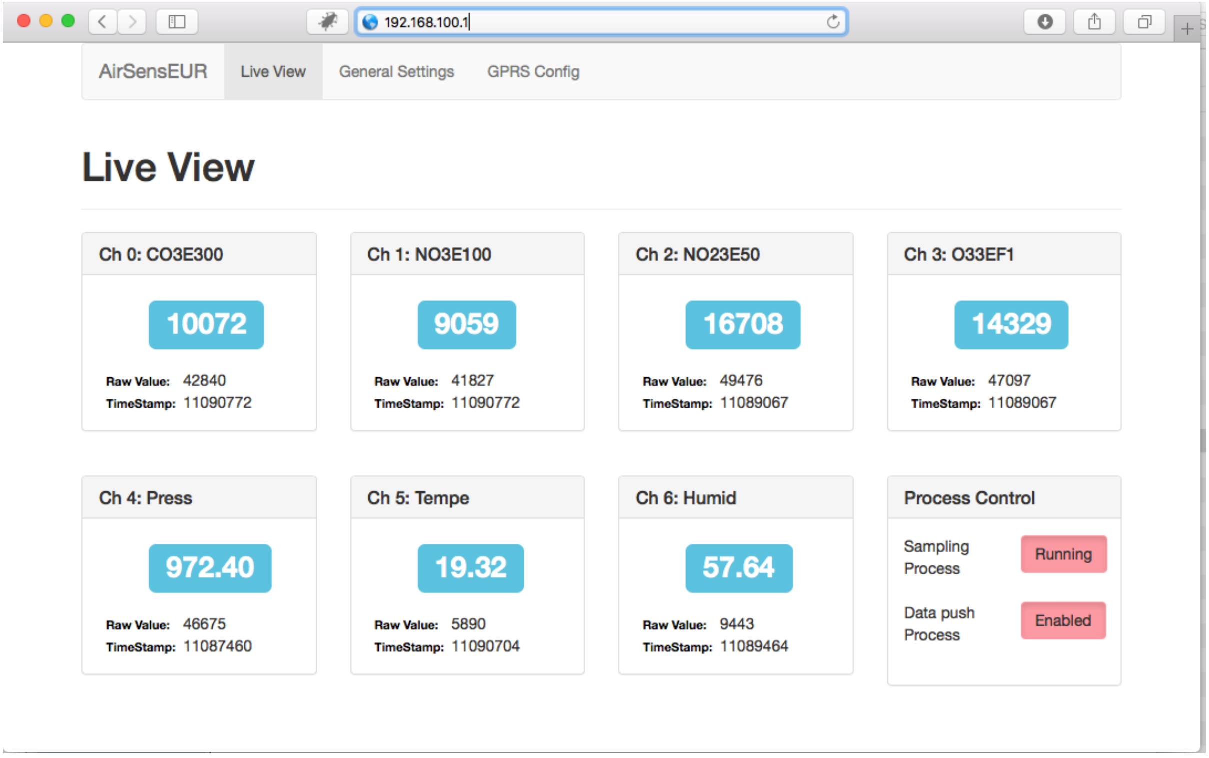


```
root@arietta:/usr/local/airsenseur/AirSensEURDataAggregator# ls -la
total 22520
drwxr-xr-x 3 root root      4096 Dec 24 15:45 .
drwxr-sr-x 5 root staff     4096 Dec 20 12:06 ..
-rw-r--r-- 1 root root     15720 Nov 21 09:06 AirSensEURDataAggregator.jar
-rw-r--r-- 1 root root    574038 Dec 17 15:44 DataSet_000001.csv
-rw-r--r-- 1 root root     61799 Dec 17 16:29 DataSet_000002.csv
-rw-r--r-- 1 root root    297722 Dec 17 20:10 DataSet_000003.csv
-rw-r--r-- 1 root root    716253 Dec 18 05:01 DataSet_000004.csv
-rw-r--r-- 1 root root         0 Dec 18 05:27 DataSet_000005.csv
-rw-r--r-- 1 root root    109665 Dec 18 06:45 DataSet_000006.csv
-rw-r--r-- 1 root root     23065 Dec 18 07:01 DataSet_000007.csv
-rw-r--r-- 1 root root         0 Dec 19 11:22 DataSet_000008.csv
-rw-r--r-- 1 root root      7579 Dec 19 11:27 DataSet_000009.csv
-rw-r--r-- 1 root root    231992 Dec 19 14:18 DataSet_000010.csv
-rw-r--r-- 1 root root    275188 Dec 19 17:41 DataSet_000011.csv
-rw-r--r-- 1 root root     60721 Dec 19 19:05 DataSet_000012.csv
-rw-r--r-- 1 root root   1044486 Dec 20 09:54 DataSet_000013.csv
-rw-r--r-- 1 root root    134770 Dec 20 11:42 DataSet_000014.csv
-rw-r--r-- 1 root root     15886 Dec 20 12:08 DataSet_000015.csv
-rw-r--r-- 1 root root   5760209 Dec 23 14:55 DataSet_000016.csv
-rw-r--r-- 1 root root     21314 Dec 23 15:56 DataSet_000017.csv
-rw-r--r-- 1 root root   1916556 Dec 24 15:45 DataSet_000018.csv
-rw-r--r-- 1 root root      1340 Nov 21 09:06 README.TXT
-rw-r--r-- 1 root root  11680768 Dec 24 15:45 airsenseur.db
-rwxr-xr-x 1 root root       374 Oct  5 14:14 airsenseurdataaggregator
drwxr-xr-x 2 root root      4096 Jul 25 10:20 lib
root@arietta:/usr/local/airsenseur/AirSensEURDataAggregator# █
```

In the list, the .csv files and the airsenseur.db were generated and updated by the sampling process as explained in the previous paragraph.

When not required and if operating with batteries, the WiFi access point should be disabled to reduce power consumption.

To access to the Web Management Console, open a browser and point to the URL http://192.168.100.1. The Live View page should be shown.


192.168.100.1

AirSensEUR | Live View | General Settings | GPRS Config

Live View

Ch 0: CO3E300
10072
Raw Value: 42840
TimeStamp: 11090772

Ch 1: NO3E100
9059
Raw Value: 41827
TimeStamp: 11090772

Ch 2: NO23E50
16708
Raw Value: 49476
TimeStamp: 11089067

Ch 3: O33EF1
14329
Raw Value: 47097
TimeStamp: 11089067

Ch 4: Press
972.40
Raw Value: 46675
TimeStamp: 11087460

Ch 5: Tempe
19.32
Raw Value: 5890
TimeStamp: 11090704

Ch 6: Humid
57.64
Raw Value: 9443
TimeStamp: 11089464

Process Control
Sampling Process: Running
Data push Process: Enabled

6.3 Functionalities of buttons and LEDs

The AirSensEUR Host panel implements 8 software controllable LEDs and pushbuttons plus one LED and two pushbuttons statically assigned to specific functions.

Statically assigned functions are related to:

- LED1: 3DFix. This LED is directly connected to the GPS module. When off it indicates that the GPS module is not powered; when blinking in a 50/50 pattern it indicates that the GPS module is powered on but no valid 3DFix is available. As soon as the 3DFix is available the LED will flash periodically with a short blinks.
- P2: POWERON. This pushbutton handles the main board power status. When in off status, pressing for more than a second will power up the Host; when in on status, pressing for more than a second will start a graceful shutdown of the Host.
- P1: RESET. This pushbutton forces a CPU reset.

Software controllable LEDs and pushbuttons are operational only when the Linux kernel has been started. Being software based, their functionality can be modified by kernel parameters and/or bash shell scripts. Some functions have already been defined:

- P10 and LED9: Sampling on/off. Pressing the button will start/stop the sampling process. This involves turning on/off the on-board GPS module. The LED blinks when the operation is in progress but is steady on when the sampling process has been started; off when no sampling is running
- P9 and LED8: WiFi Access Point functionality on/off. Pressing the button will start/stop the Access Point WiFi functionality. The LED blinks when the operation is in progress but is steady on when the WiFi Access Point functionality is turned on (the Host is broadcasting their name and ready to accept clients connections).
- P8 and LED7: Data synchronization process scheduler on/off. Pressing the button will enable/disable the data synchronization scheduler. When the LED is on, the data synchronization process is triggered based on the scheduled program (see next chapters on

how to define the scheduler plan). When the LED is off, the scheduler is disabled i.e. no synchronization with external server will be done. The LED blinks when the synchronization is taking place.

- LED3: CPU Heartbeat. This LED blinks in a 50/50 pattern when the system is in the boot/shutdown status. The LED blinks in a "heartbeat" like pattern when the system is ready and operating. The pulse frequency is proportional to the CPU load status.
- LED2: Battery charge status. The LED is off when Host is running battery powered. The LED flashes when the battery is charging but is steady on when the battery is charged.
- P3: Stealth mode. Pressing this pushbutton toggles the stealth mode. When the stealth mode is on, all software configurable LEDs (except for the battery charge status) are forced off thus reducing the battery power consumption. Exiting from stealth mode restore the software configurable LEDs status.

6.4 Configuring the AirSensEUR Host

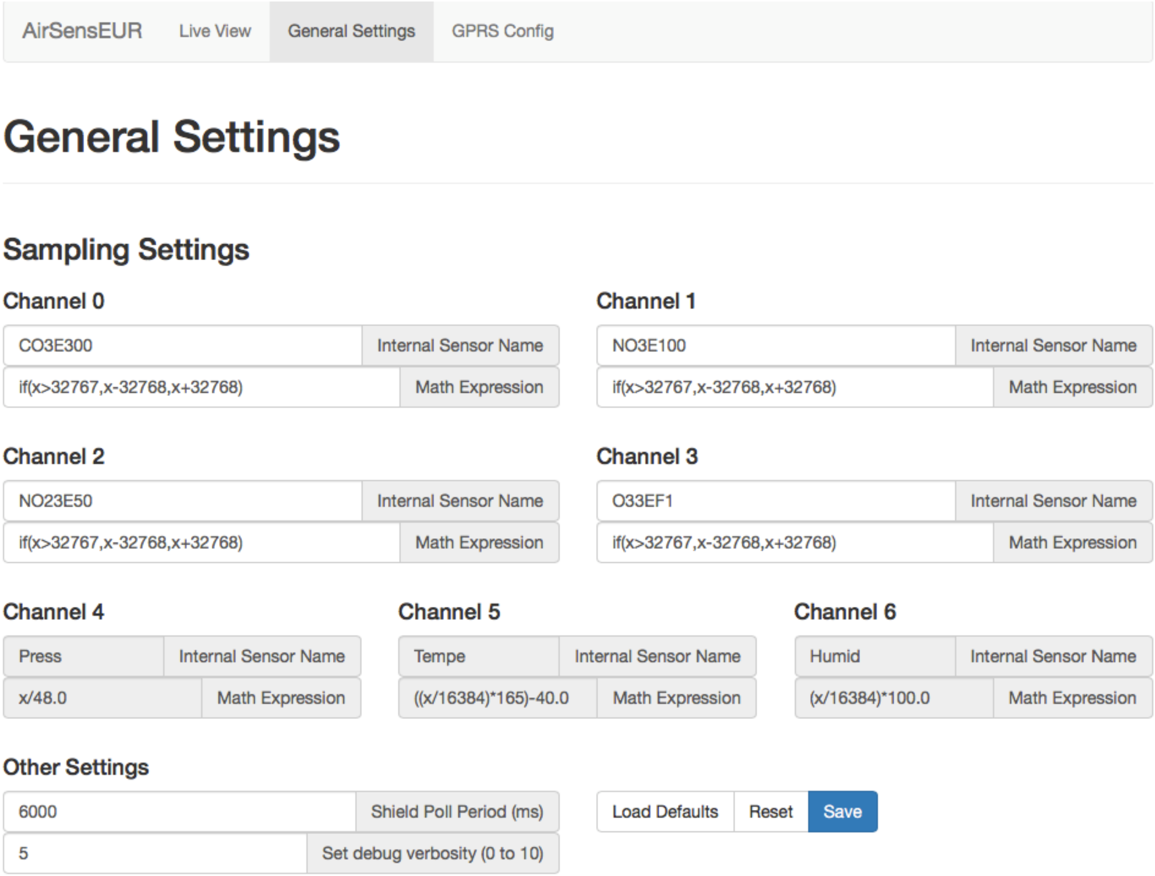
The configuration of a new AirSensEUR Host involves a set of steps. There are different areas where the configuration should take place. The steps are summarised below:

- Define the name of installed sensors and an optional math expression to evaluate and/or calibrate the read samples
- Configure the remote server parameters (remote host IP, port, database identifier etc.)
- Set up the proper operations to be done to perform data synchronization
- Schedule the data synchronization process
- Configure the GPRS parameters to adapt to a specific GPRS public operator

At the time of writing, not all settings can be defined through the Web Management Console but requires an SSH console. This will not be required in the future releases.

6.5 Name and math expression for sensors

The "General Settings" page of the Web Management Console provides a convenient way to set the internal sensor name (the name will be stored in the local database together with samples) and an optional mathematical expression.



General Settings

Sampling Settings

Channel 0
CO3E300 — Internal Sensor Name
if(x>32767,x-32768,x+32768) — Math Expression

Channel 1
NO3E100 — Internal Sensor Name
if(x>32767,x-32768,x+32768) — Math Expression

Channel 2
NO23E50 — Internal Sensor Name
if(x>32767,x-32768,x+32768) — Math Expression

Channel 3
O33EF1 — Internal Sensor Name
if(x>32767,x-32768,x+32768) — Math Expression

Channel 4
Press — Internal Sensor Name
x/48.0 — Math Expression

Channel 5
Tempe — Internal Sensor Name
((x/16384)*165)-40.0 — Math Expression

Channel 6
Humid — Internal Sensor Name
(x/16384)*100.0 — Math Expression

Other Settings
6000 — Shield Poll Period (ms)
5 — Set debug verbosity (0 to 10)

The mathematical expression can be used to calibrate and/or pre-process samples read from the sensor. The syntax should follow the design guidelines published at https://github.com/darius/expr document folder.

NOTE: Chemical sensors outputs are sampled in two-complements values; take care of it in the associated mathematical expression.

6.6 Configure the remote server parameters

Influx DB remote server parameters can be changed by modifying, though a text editor via SSH connection, the /usr/local/etc/influxpush.properties file.

The relevant entries, in this file, are:

```
Influxdbhost=your.remote.server.ip
influxdbport=remote_server_port
influxdbname=airsenseur00
influxdbdataset=HomeData
influxdbuser=airsenseur
influxdbpasswd=airsenseur
```

Their values should reflect the Influx DB target server.

6.7 Configure the synchronization process

Synchronization is done though a script located in /usr/local/airsenseur/runDataPush. The default behaviour of this script is to:

- Power up the GPRS modem
- Start a PPP connection
- Push local data to a remote server
- Shut down the PPP connection
- Power down the GPRS modem

Being AirSensEUR Host an open framework is possible to modify the default behaviour in order to target to several server typologies.

Multiple servers can be updated, for example, by simply concatenating several AirSensEURDataPush runs with different parameters, as reported below:

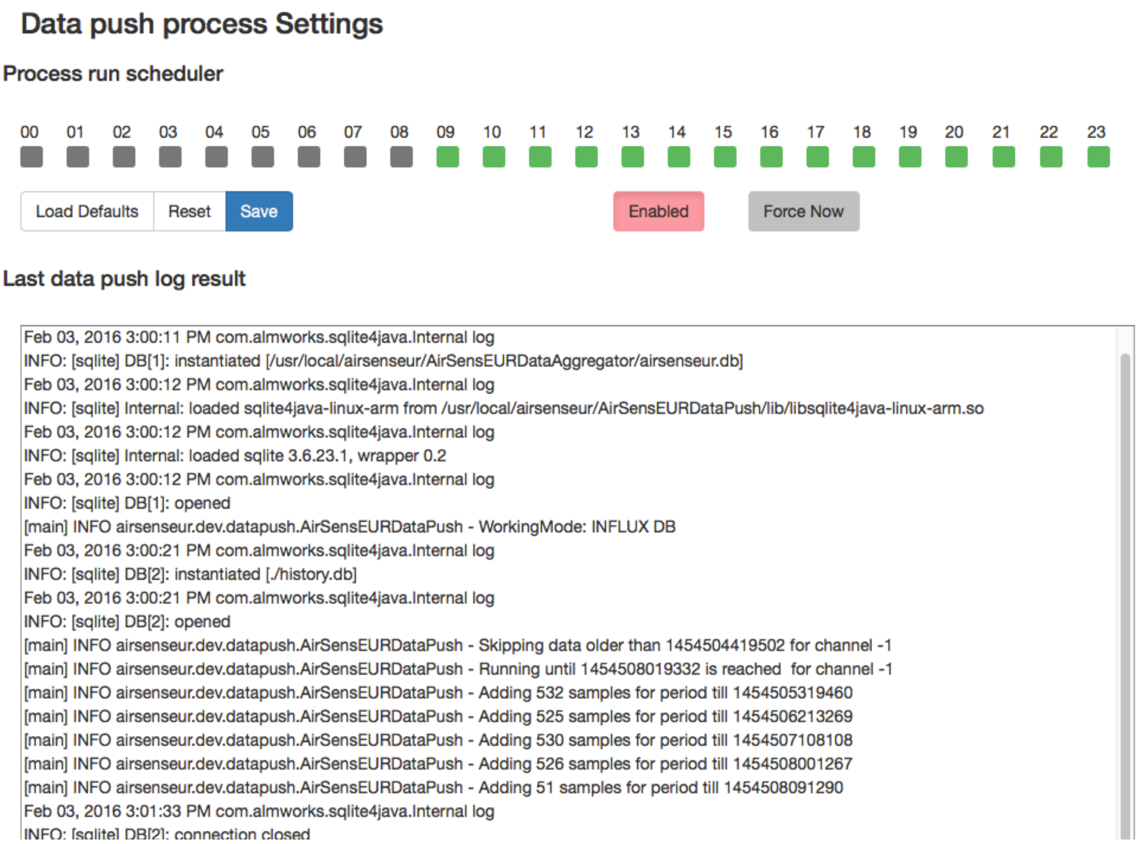
```
# Push updates

cd $SCRIPTBASE/AirSensEURDataPush

java -jar AirSensEURDataPush.jar ../../etc/influxpush.properties

java -jar AirSensEURDataPush.jar ../../etc/datapushsosdb.properties

cd ..
```

6.8 Schedule the data synchronization process

The data synchronization process is triggered following a schedule plan set through the Web Management Console.

The "General Settings" bottom page reports a convenient way to update the scheduler and a feedback on last synchronization process log.

Synchronization slots (represented with a set of squares, one for each hour a day) can be enables/disabled by clicking on it. Green squares identify slots where the synchronization will take place. The scheduled plan can be updated by clicking on the "Save" button.



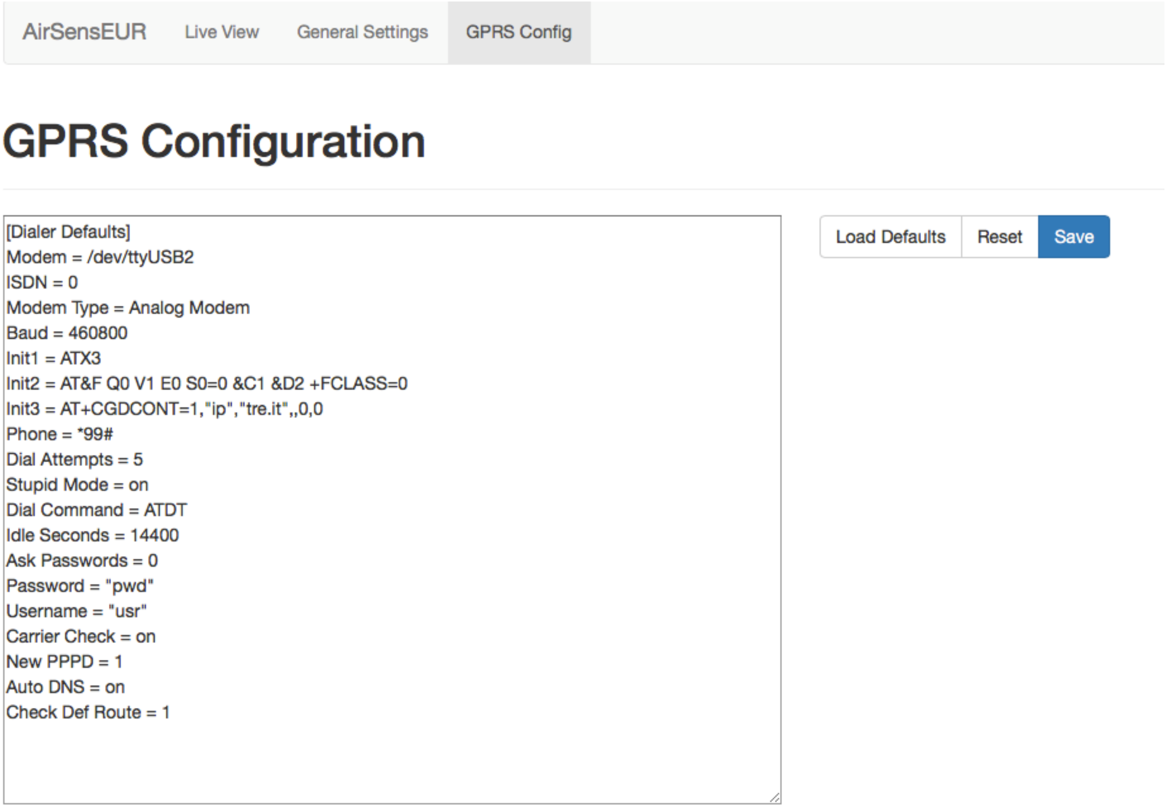
```
Feb 03, 2016 3:00:11 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[1]: instantiated [/usr/local/airsenseur/AirSensEURDataAggregator/airsenseur.db]
Feb 03, 2016 3:00:12 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] Internal: loaded sqlite4java-linux-arm from /usr/local/airsenseur/AirSensEURDataPush/lib/libsqlite4java-linux-arm.so
Feb 03, 2016 3:00:12 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] Internal: loaded sqlite 3.6.23.1, wrapper 0.2
Feb 03, 2016 3:00:12 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[1]: opened
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - WorkingMode: INFLUX DB
Feb 03, 2016 3:00:21 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[2]: instantiated [./history.db]
Feb 03, 2016 3:00:21 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[2]: opened
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Skipping data older than 1454504419502 for channel -1
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Running until 1454508019332 is reached  for channel -1
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Adding 532 samples for period till 1454505319460
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Adding 525 samples for period till 1454506213269
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Adding 530 samples for period till 1454507108108
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Adding 526 samples for period till 1454508001267
[main] INFO airsenseur.dev.datapush.AirSensEURDataPush - Adding 51 samples for period till 1454508091290
Feb 03, 2016 3:01:33 PM com.almworks.sqlite4java.Internal log
INFO: [sqlite] DB[2]: connection closed
```

The overall synchronization scheduler can be enabled/disabled by clicking on the "Enable" button (this button reflects the status of the synchronization LED in the user panel). An extra "forced" synchronization can be triggered whenever needed through the "Force Now" pushbutton.

6.9 Configure the GPRS parameters

A TPLink MA180 USB GPRS Modem has been used to provide GPRS connection. Insert a GPRS Data enabled SIM card on the GPRS Modem and connect the GPRS Modem in the lower USB socket of the host. Be sure to disable any lock code protecting the SIM before inserting it in the GPRS Modem. The AirSensEUR Host should be configured with an APN appropriated for your SIM provider. This can be done by editing a specific configuration file, as explained in the next chapters.

GPRS related parameters could be modified thought the GPRS Config page in the Web Management Console.



```
[Dialer Defaults]
Modem = /dev/ttyUSB2
ISDN = 0
Modem Type = Analog Modem
Baud = 460800
Init1 = ATX3
Init2 = AT&F Q0 V1 E0 S0=0 &C1 &D2 +FCLASS=0
Init3 = AT+CGDCONT=1,"ip","tre.it",,0,0
Phone = *99#
Dial Attempts = 5
Stupid Mode = on
Dial Command = ATDT
Idle Seconds = 14400
Ask Passwords = 0
Password = "pwd"
Username = "usr"
Carrier Check = on
New PPPD = 1
Auto DNS = on
Check Def Route = 1
```

This page reports the wvDial parameters used by AirSensEUR to connect to the GPRS public data. For more information on the syntax and token values, please refer to the wvDial on the Linux man page: http://linux.die.net/man/5/wvdial.conf
All those settings are highly dependent on the public mobile network but we found the most relevant settings are related to the "Modem" key, which identify the physical modem connection device file, and the "Init3" key. This last key defines the public APN to be used by the GPRS modem to subscribe to the public mobile network.

6.10 Changing SSID of AirSensEUR

The SSID of Any AirSensEUR can be change using the following procedure:

- Connect to AirSenSEUR in WiFi mode
- Launch and ssh session
- Open the following file in edit mode /etc/hostapd/hostapd.conf and change the following key with the desired SSID: ssid=AirSensEUR
- Save the file and reconnect to AirSensEUR WiFi with the new SSID

It is also possible to edit the radio channel of the AirSensEUR WiFi to avoid interferences when several AirSensEUR are in vicinity by chaining the following key: channel=6. The valid values range between 3 to 11.

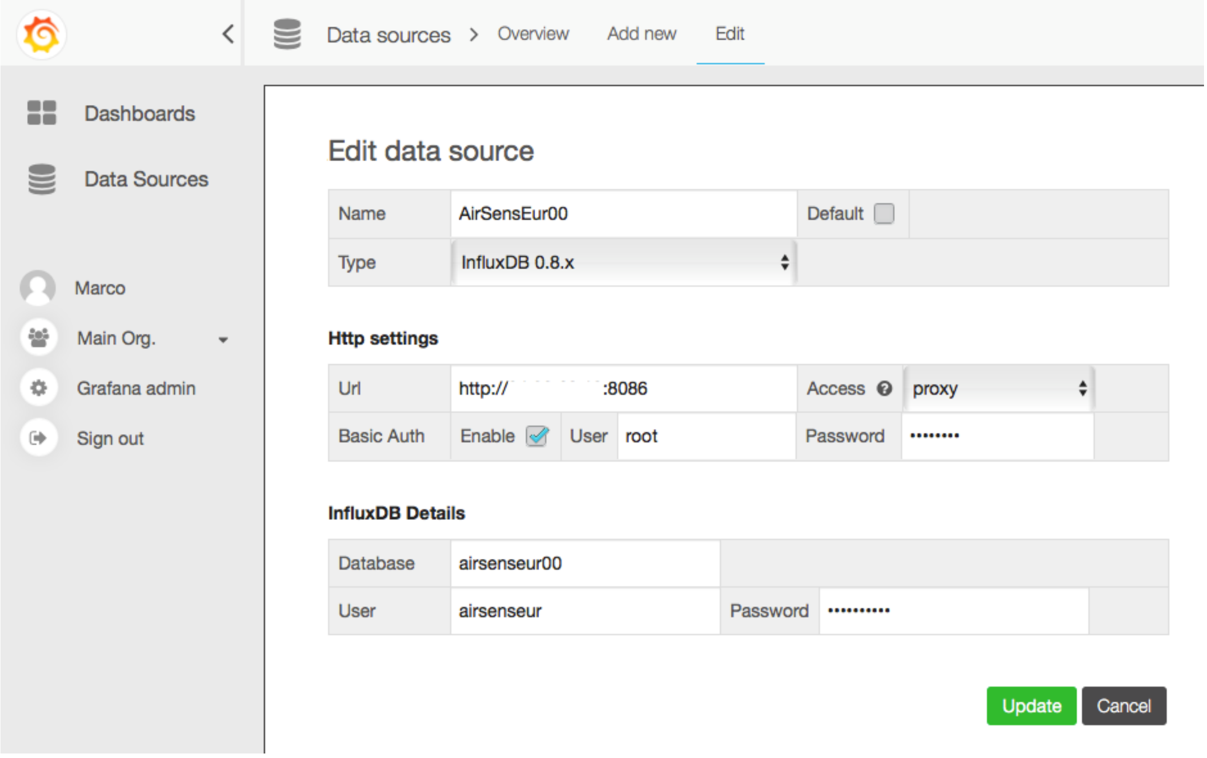
7 Managing of the remote data server

AirSensEUR Host can push collected data to an external server where they can be aggregated to generate more complex statistics. The AirSensEUR framework is open to several technologies but, in this paper, we opted for the Grafana + InfluxDB solution mainly because they're well known, easy to deploy and stable services targeted to IoT sensors. Grafana and Influx DB runs on top of a CentOS 6 Linux distribution.

Installation and configuration of Influx DB and Grafana are out of the scope of this paper. More information can be found on online tutorials at https://docs.influxdata.com/influxdb/v0.9/introduction/installation/ and http://docs.grafana.org/installation/

As soon as the services are installed, a new data source should be configured in the Grafana administration panel.

Click on "DataSources" and "Add new" and provide the required parameters.



Edit data source

Name	AirSensEur00	Default □	
Type	InfluxDB 0.8.x		

Http settings

Url	http:// :8086			Access	proxy
Basic Auth	Enable ☑	User	root	Password	••••••••

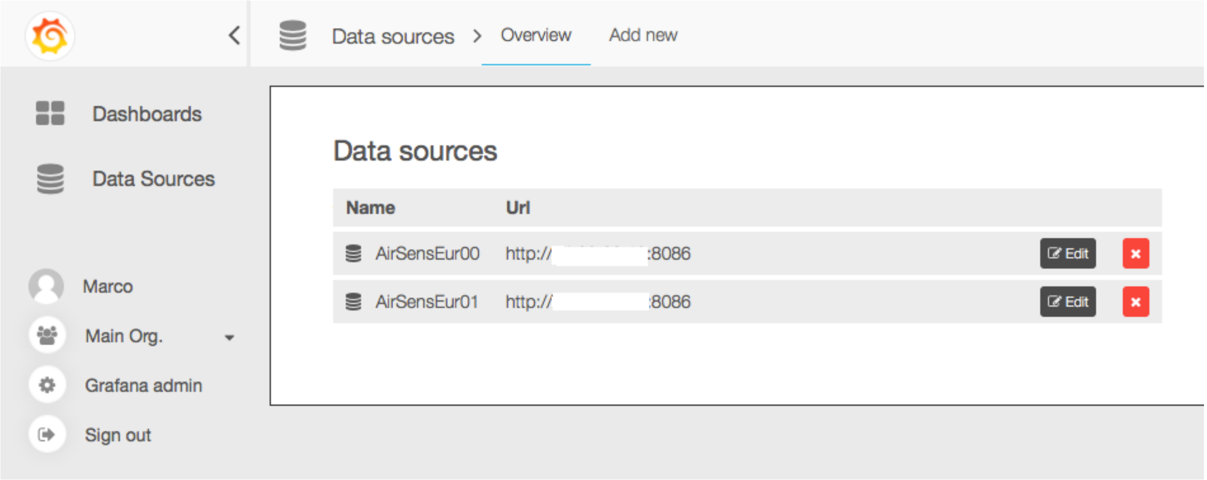
InfluxDB Details

Database	airsenseur00		
User	airsenseur	Password	••••••••••

In our example we defined a local datasource to the *airsenseur00* database for user *airsenseur* and related password. We named it as *AirSensEUR00*

Note that those parameters should match what already defined in the AirSensEURDataPush configuration file and should reflect the name associated to the InfluxDB database you need to generate through the InfluxDB administrative web-pages.

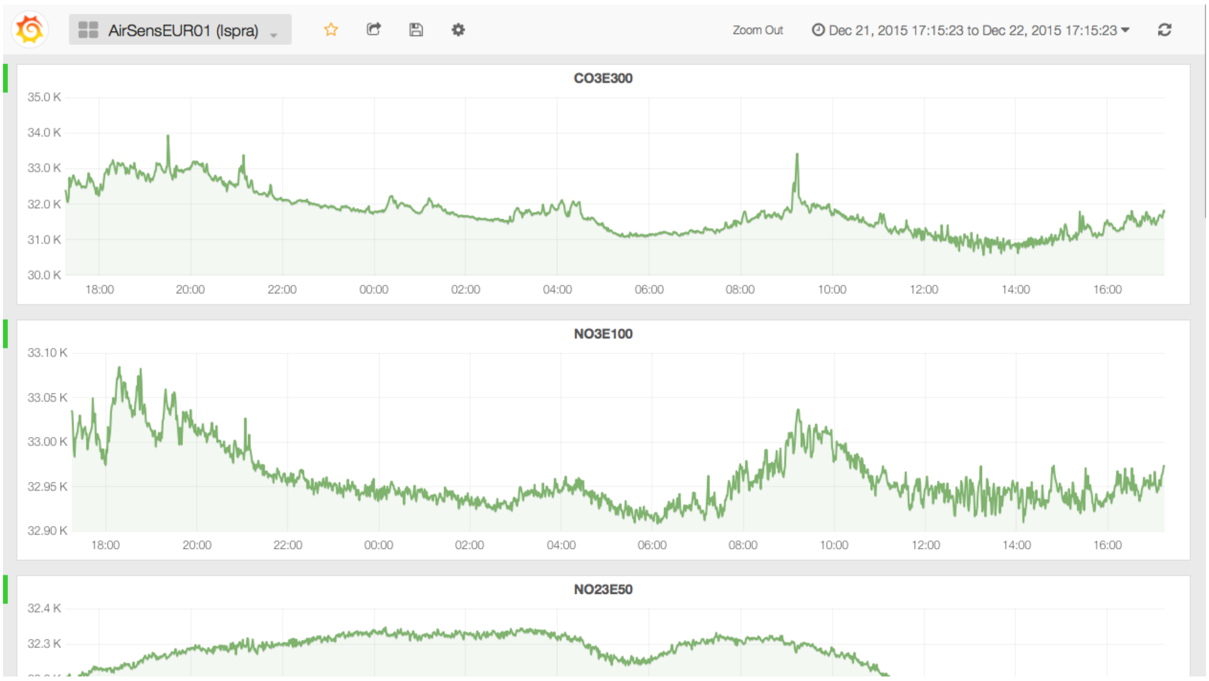
Several datasources can be defined if more than an AirSensEUR Host should be monitored. In our example we defined two different datasources targeted to two different AirSensEUR Hosts.



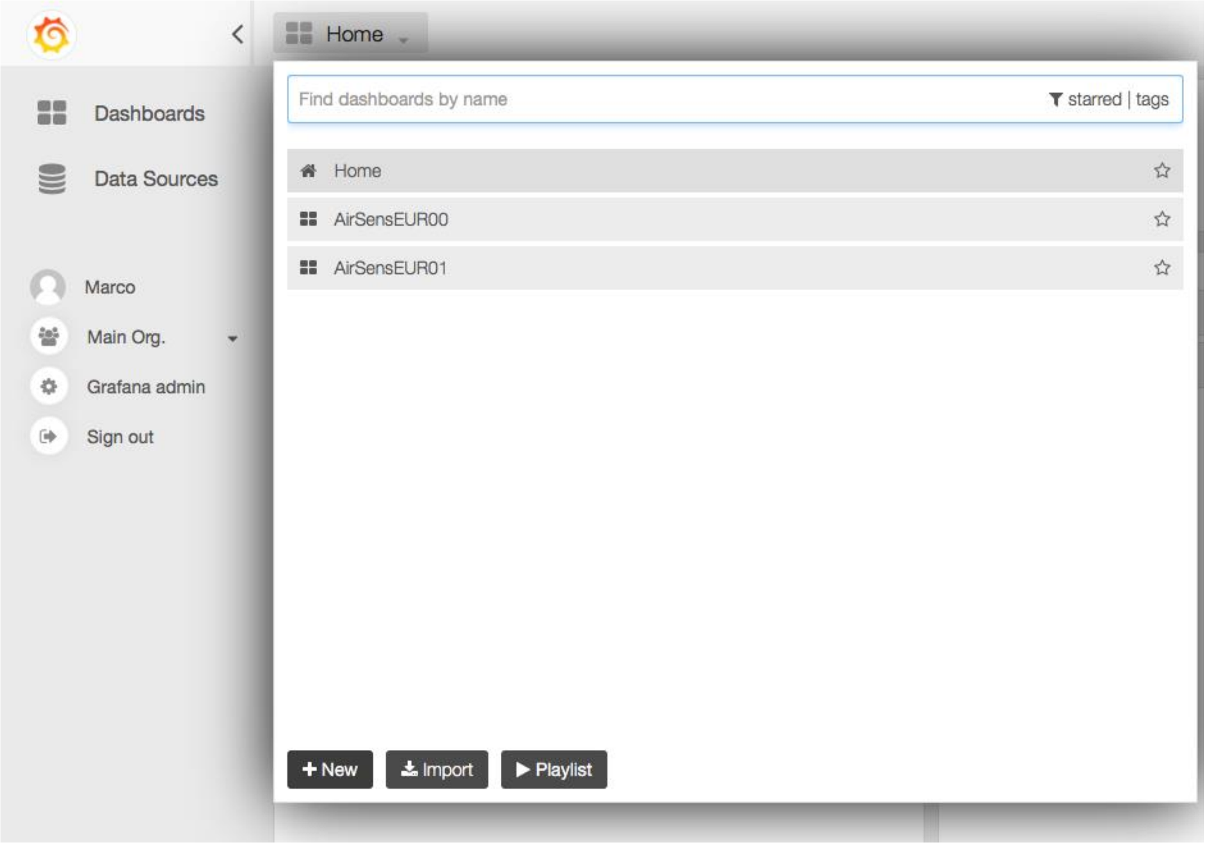
Data sources

Name	Url		
AirSensEur00	http://[redacted]:8086	Edit	✖
AirSensEur01	http://[redacted]:8086	Edit	✖

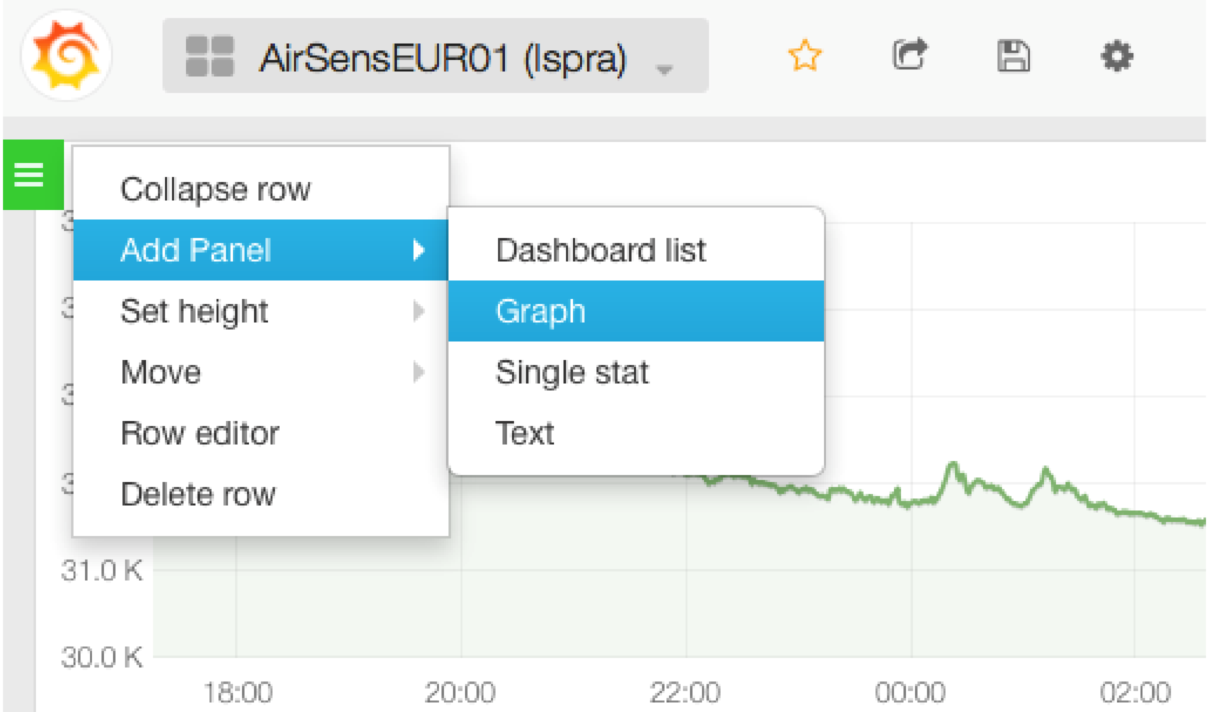
Data sources can be used to retrieve data from the database and produce graphs. Graphs are organized in dashboards for easy access.
In the example below we generated a panel reporting the sampled values for a single AirSensEUR Unit.



Adding a dashboard is very easy. Click on the "Home" button in the top of the Graphana page and a new panel will be shown with known dashboards. Click on the "New" button to create a new dashboard.

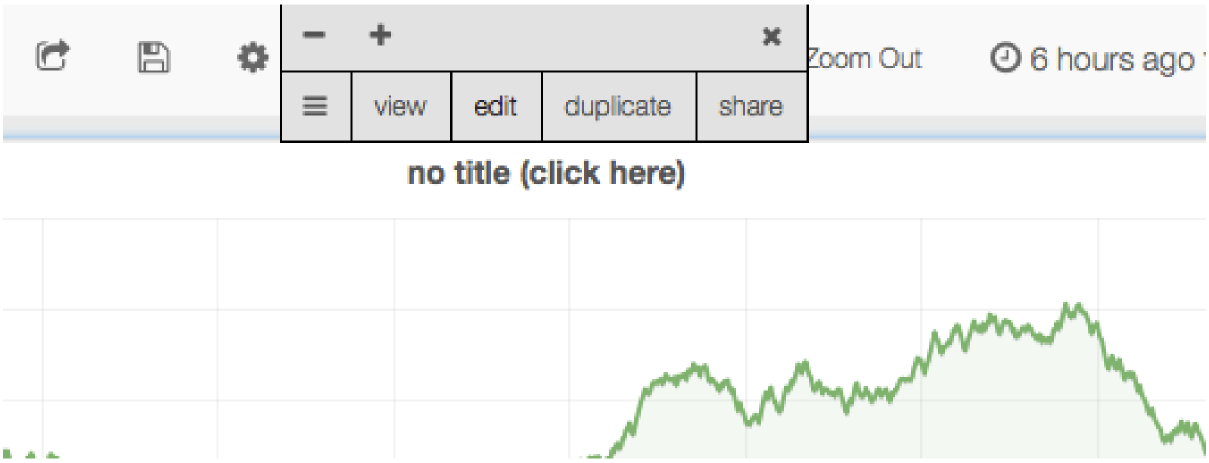


A dashboard is composed of a set of rows. Each row in the dashboard contains a green box. Clicking on this, a dropdown menu is shown. To add a new graph, select "Add Panel" and then "Graph".

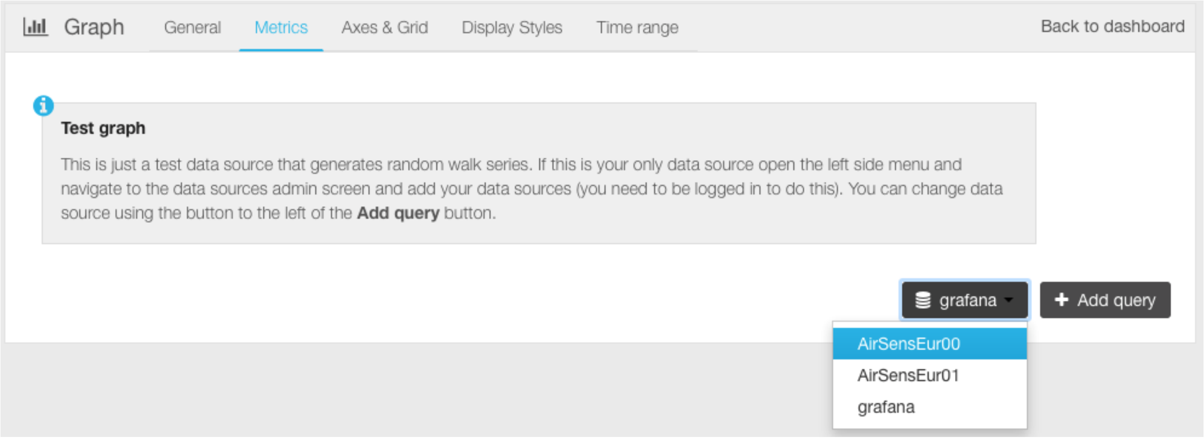


A default graph is shown in the dashboard. This is not connected with any datasource.

Click on the title, then on the "edit" button.



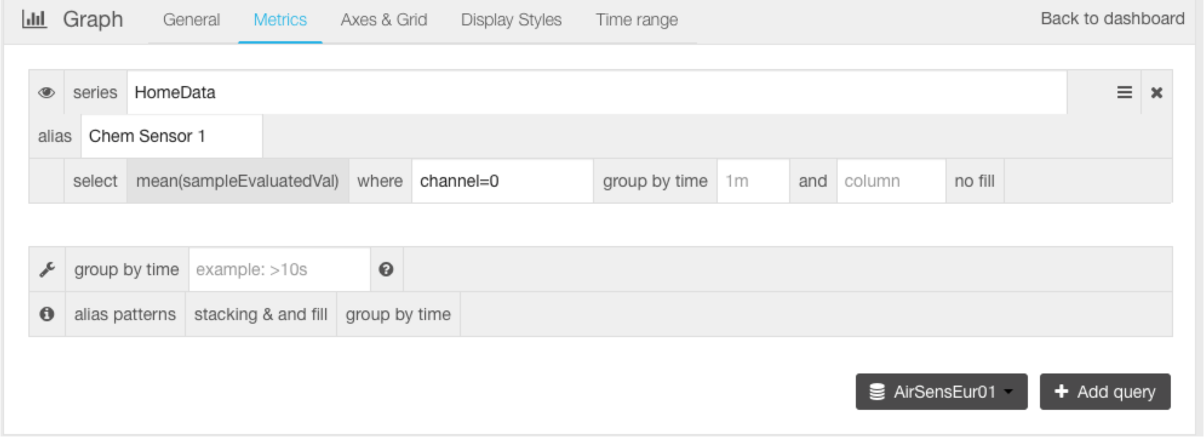
On the bottom of the page select the proper datasource; a new empty query is added by default.



Test graph

This is just a test data source that generates random walk series. If this is your only data source open the left side menu and navigate to the data sources admin screen and add your data sources (you need to be logged in to do this). You can change data source using the button to the left of the **Add query** button.

To show the average values for samples related to the first chemical sensor in the AirSensEUR Shield, the following query parameters has been added for our test setup.



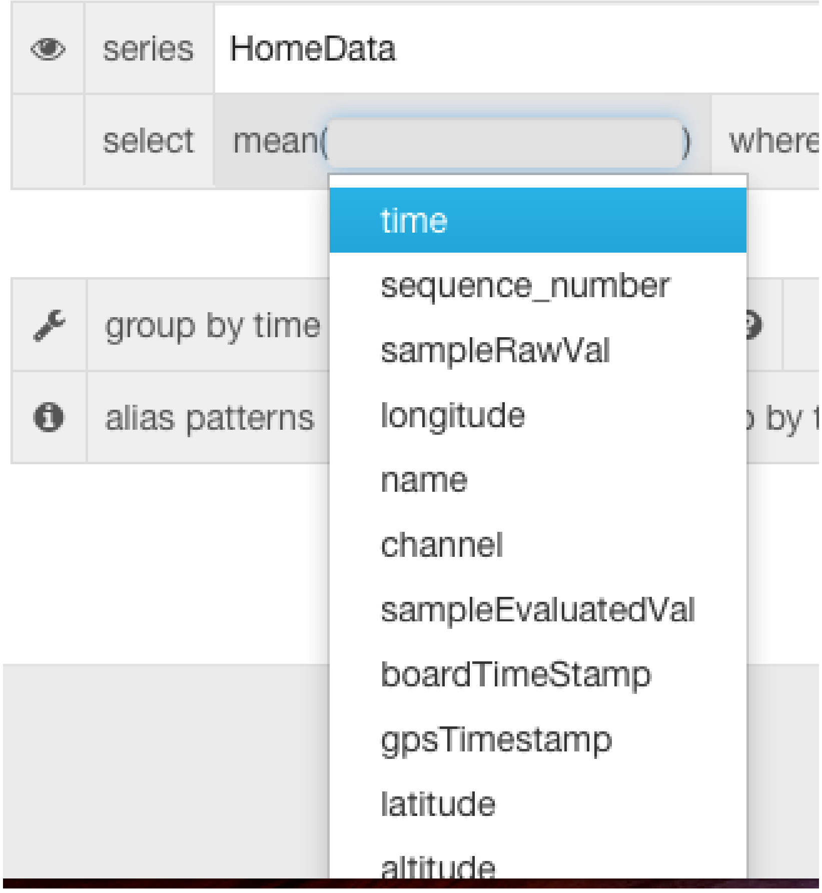
series HomeData

alias Chem Sensor 1

select mean(sampleEvaluatedVal) where channel=0 group by time 1m and column no fill

group by time example: >10s

The query refers to the HomeData series in the AirSensEUR01 datasource and generates a graph of "mean" values of "sampleEvaluatedVal" column in the database. Note that "sampleEvaluatedVal" label is strictly dependent on how the data are transmitted to the Influx DB by the AirSensEUR Host. Columns labels, available for the query, are suggested when typing the query, as shown in the picture below:



series HomeData

select mean( )

The query can be executed by clicking on "Back to dashboard" link located on the query panel top right.

Dashboards should be saved and shared through the appropriate buttons, located on the top bar.

8 AirSensEURPanel Rev 0.2 changelog

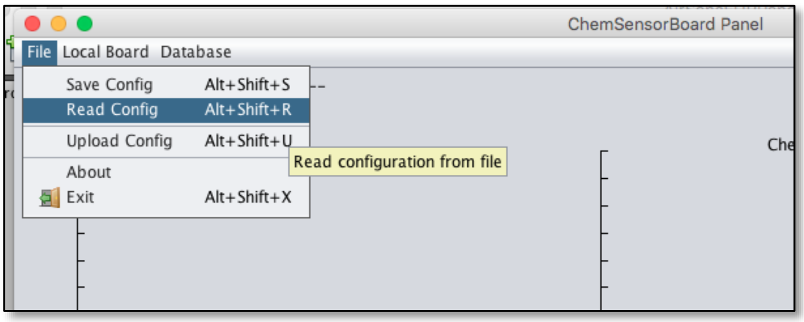
8.1 New features

The AirSensEURPanel software is used for the configuration of the sensor on the sensor shield. The application was already presented in the AirSensEUR report A [1]. Several new features were added in this revision. Below is their description and basic explanations on how to use them.

8.2 Persistent configuration via file

AirSensEUR Shield configuration can be persisted through files. This allows to easily sharing configuration information between several boards and/or users. In order to generate a configuration file, as soon as the overall configuration parameters are set through the proper panels, the user should select the "File->Save Config" option in the main menu.

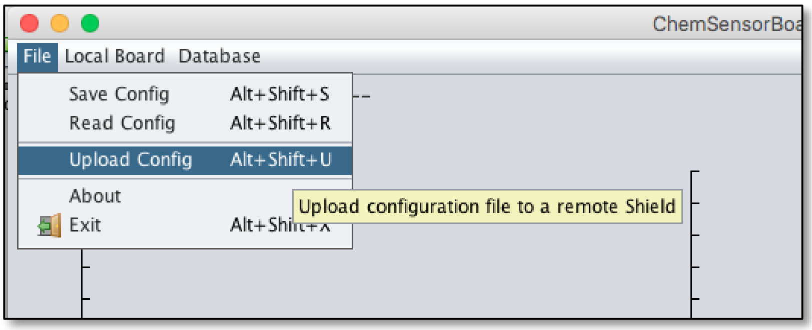
A file picker will be open, so, the user will be able to specify the file name and location. The file is created as soon as the user confirms this dialog.



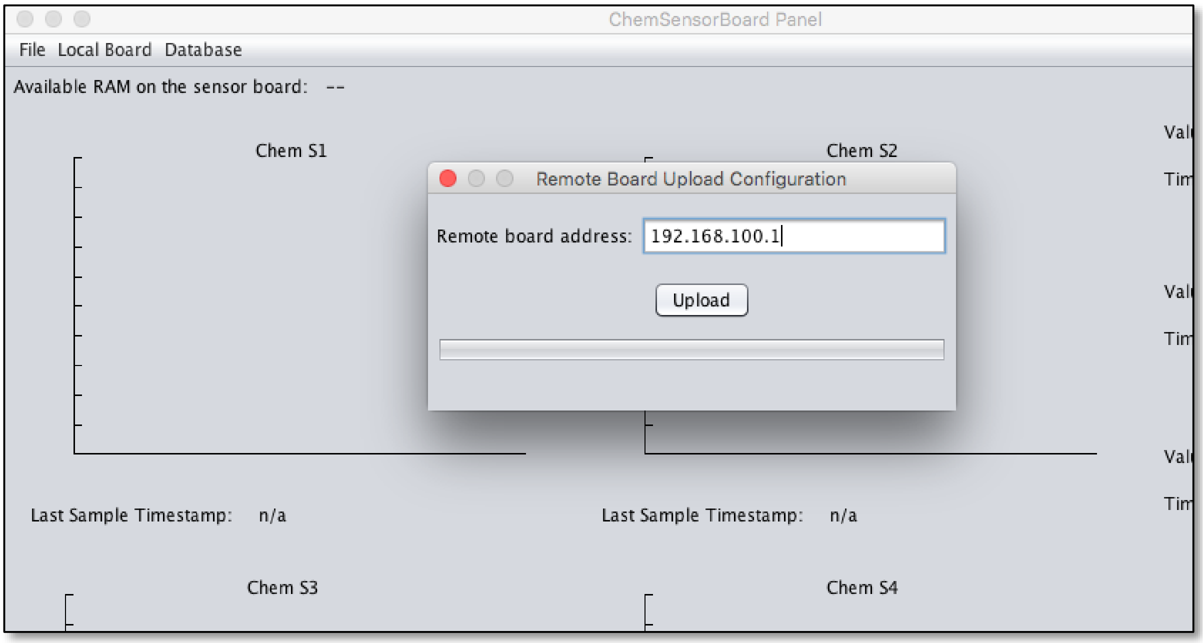
To load a previously saved configuration, the user should select the option "File -> Read Config" from the menu, then should select the appropriated file though the file picker that will be shown.

8.3 Upload configuration to a remote AirSensEUR Shield

When the AirSensEUR Shield is connected to an AirSensEUR Host device, is sometimes useful to be able to update their configuration without connecting to the PC running the AirSensEUR Panel. This is what the "File -> Upload Config" option is for.



When selecting this option, a new popup window will be shown. This is represented on next following picture.


ChemSensorBoard Panel
File
Local Board
Database
Available RAM on the sensor board: --
Chem S1
Chem S2
Remote Board Upload Configuration
Upload
Last Sample Timestamp: n/a
Last Sample Timestamp: n/a
Chem S3
Chem S4
Val
Tim
Val
Tim
Val
Tim

Remote board address: 192.168.100.1

The popup is already populated with the standard IP address associated to an AirSensEUR Host when running in ad hoc mode through WiFi but it can be modified, if required. To start the transfer, click on the “Upload” button. In a few seconds the actual configuration will be transferred to the remote AirSensEUR Shield and applied.

The panel is not modal, so, it can be left open when changing configuration, and used to update the remote board several times.

TIPS: if you want to update a remote AirSensEUR Shield with a configuration stored in a specific configuration file, load the configuration file then upload the read configuration through this panel.

8.4 Sensor Database

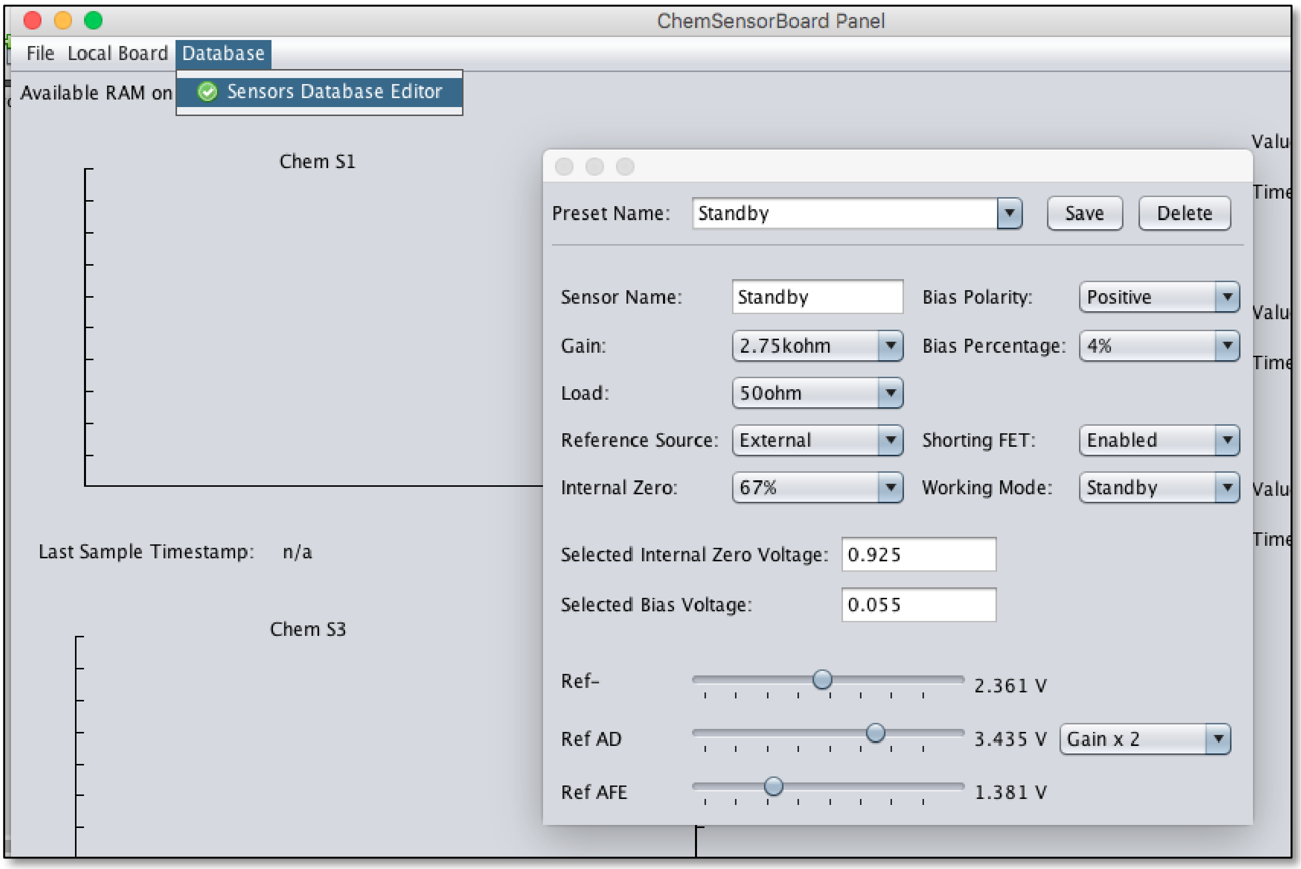
Each sensor brand and typology has its associated front end and DAC options. To easily set-up for a specific sensor type/brand, a sensor database has been implemented. The database can be edited through a specific panel shown by clicking on the "Database->Edit ->Sensors Database Editor" menu option. The dialog is shown in the following picture.

The sensors database editor dialog is split in two parts. The top row allows loading, saving, updating and deleting a specific sensor "preset". Each available preset is listed on an editable dropdown. A preset can be recalled by selecting it through the dropdown; it can be saved and/or deleted by clicking on the "Save" and/or "Delete" buttons.

Settings for a specific preset are defined in the lower side of the dialog where parameters, related to the Analog Front End and reference voltages applied by the AirSensEUR Shield hardware, are shown.

To create a new preset:

1. Select the appropriate AFE and reference settings through the GUI items in the panel
2. Edit a suitable preset name in the top dropdown
3. Click on the "Save" button



Preset Name: Standby

Sensor Name: Standby

Bias Polarity: Positive

Gain: 2.75kohm

Bias Percentage: 4%

Load: 50ohm

Reference Source: External

Shorting FET: Enabled

Internal Zero: 67%

Working Mode: Standby

Selected Internal Zero Voltage: 0.925

Selected Bias Voltage: 0.055

Ref- 2.361 V

Ref AD 3.435 V Gain x 2

Ref AFE 1.381 V

Last Sample Timestamp: n/a

To update a known preset:

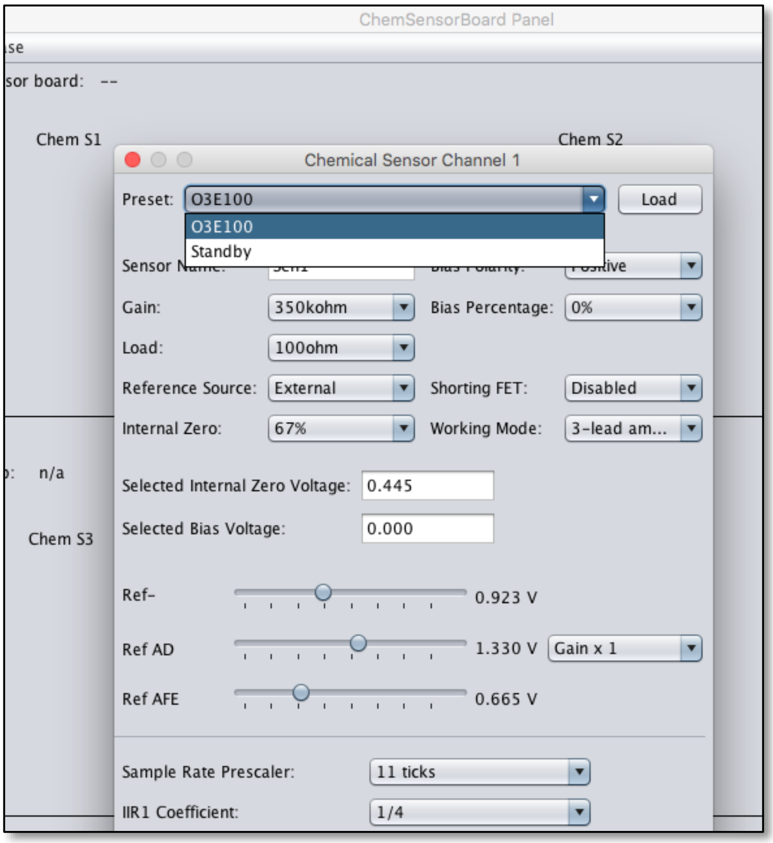
1. Select the preset you want to change through the dropdown. The preset will be load and GUI items will reflect the preset parameters
2. Modify the parameters you want
3. Click on the "Save" button then confirm you want to override the preset

To delete a known preset:

1. Select the preset you want to change through the dropdown. The preset will be load and GUI items will reflect the preset parameters
2. Click on the "Delete" button then confirm you want to delete the preset

Presets are stored in a local file called "SensorDatabase.txt" for easy backup and/or sharing with other users. The file is located in the application's working folder.

All known presets are listed in the chemical sensor channel configuration panels and can be loaded by simply selecting it through the dropdown and clicking on the "Load" button. Channel parameters are still editable, so you can use a preset as a starting point for following parameter modifications.



sor board: --

Preset: O3E100

Sensor Name:

Gain: 350kohm

Bias Percentage: 0%

Load: 100ohm

Reference Source: External

Shorting FET: Disabled

Internal Zero: 67%

Working Mode: 3-lead am...

n/a

Selected Internal Zero Voltage: 0.445

Selected Bias Voltage: 0.000

Ref- 0.923 V

Ref AD 1.330 V Gain x 1

Ref AFE 0.665 V

Sample Rate Prescaler: 11 ticks

IIR1 Coefficient: 1/4

8.5 Bug fixes

Some fixes has been applied:

8.6 Hide configuration panels

A bug was preventing to close all channel configuration panels through the standard icon on each panel window title. This is now fixed so each panel follows the standard behaviour rules.

8.7 Sticky dropdowns

A bug was preventing to properly select some options in the dropdowns. This happened when external configuration was loaded either “from board” or “from file”. The problem has been fixed rewriting the base data model on each dropdown in the current version.

9 Installation and use of AirSensEUR DEMO ver. 1.0

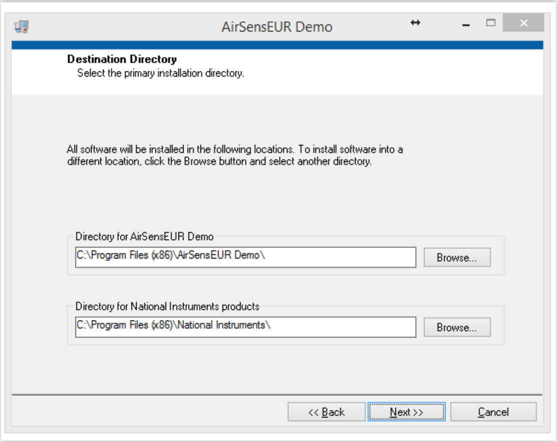
9.1 License

AirSEnsEUR Demo is delivered as a freeware, it is made available free of charge. It is not copyrighted, there is no control of its distribution. Users are not allowed to modify it and sell it. It is distributed without its source code, thus preventing modification by users.

The Software is provided "as is" without warranty of any kind, either express or implied, including without limitation any implied warranties of condition, uninterrupted use, merchantability, fitness for a particular purpose, or non-infringement. The software was installed on Windows 7 and Windows 8. The software can be downloaded at https://webgate.ec.europa.eu/CITnet/stash/projects/AIRSENSEURat or airsenseur.org

9.2 Installation of AirSensEUR DEMO ver. 1.0

1. Run setup.exe in …\20151120 builds AirSensEURDemo\AirSensEUR Demo Installer\Volume
2. Choose destination directory both for AirSensEURDEMO application and National Instruments (NI) products. It is advised to choose the Program files directory for 32 bit applications (see example below with window 8.1)



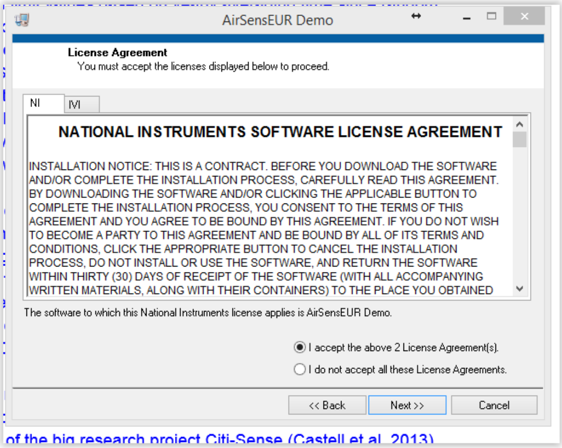
Destination Directory
Select the primary installation directory.

All software will be installed in the following locations. To install software into a different location, click the Browse button and select another directory.

Directory for AirSensEUR Demo: C:\Program Files (x86)\AirSensEUR Demo\

Directory for National Instruments products: C:\Program Files (x86)\National Instruments\

3. Accept the license agreement for the NI libraries



License Agreement
You must accept the licenses displayed below to proceed.

NATIONAL INSTRUMENTS SOFTWARE LICENSE AGREEMENT

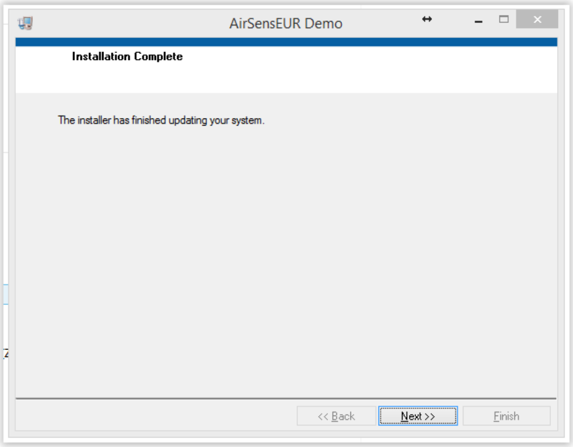
INSTALLATION NOTICE: THIS IS A CONTRACT. BEFORE YOU DOWNLOAD THE SOFTWARE AND/OR COMPLETE THE INSTALLATION PROCESS, CAREFULLY READ THIS AGREEMENT. BY DOWNLOADING THE SOFTWARE AND/OR CLICKING THE APPLICABLE BUTTON TO COMPLETE THE INSTALLATION PROCESS, YOU CONSENT TO THE TERMS OF THIS AGREEMENT AND YOU AGREE TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT WISH TO BECOME A PARTY TO THIS AGREEMENT AND BE BOUND BY ALL OF ITS TERMS AND CONDITIONS, CLICK THE APPROPRIATE BUTTON TO CANCEL THE INSTALLATION PROCESS, DO NOT INSTALL OR USE THE SOFTWARE, AND RETURN THE SOFTWARE WITHIN THIRTY (30) DAYS OF RECEIPT OF THE SOFTWARE (WITH ALL ACCOMPANYING WRITTEN MATERIALS, ALONG WITH THEIR CONTAINERS) TO THE PLACE YOU OBTAINED

The software to which this National Instruments license applies is AirSensEUR Demo.

● I accept the above 2 License Agreement(s).
○ I do not accept all these License Agreements.

of the big research project Citi-Sense (Castell et al. 2013)

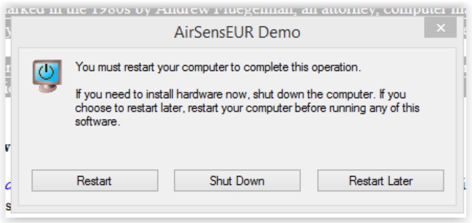
4. You get the following message when the software is installed:



Installation Complete

The installer has finished updating your system.

5. You need to re-boot your system to run AirSensEURDEMO:

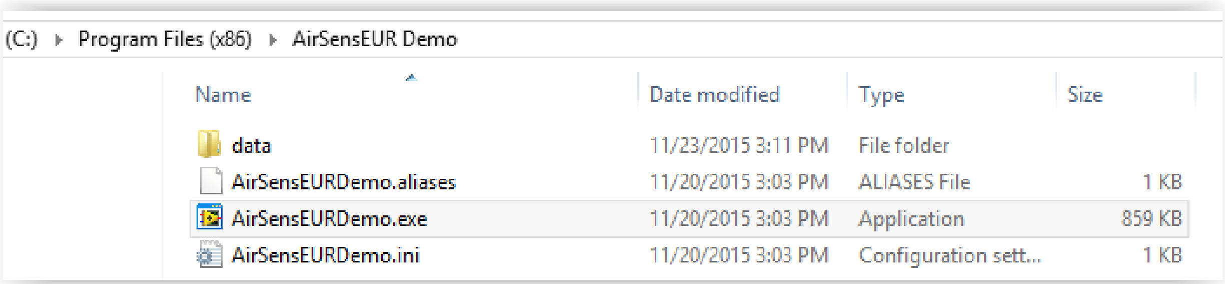


You must restart your computer to complete this operation.

If you need to install hardware now, shut down the computer. If you choose to restart later, restart your computer before running any of this software.

9.3 Installation of AirSensEUR DEMO vs 1.0

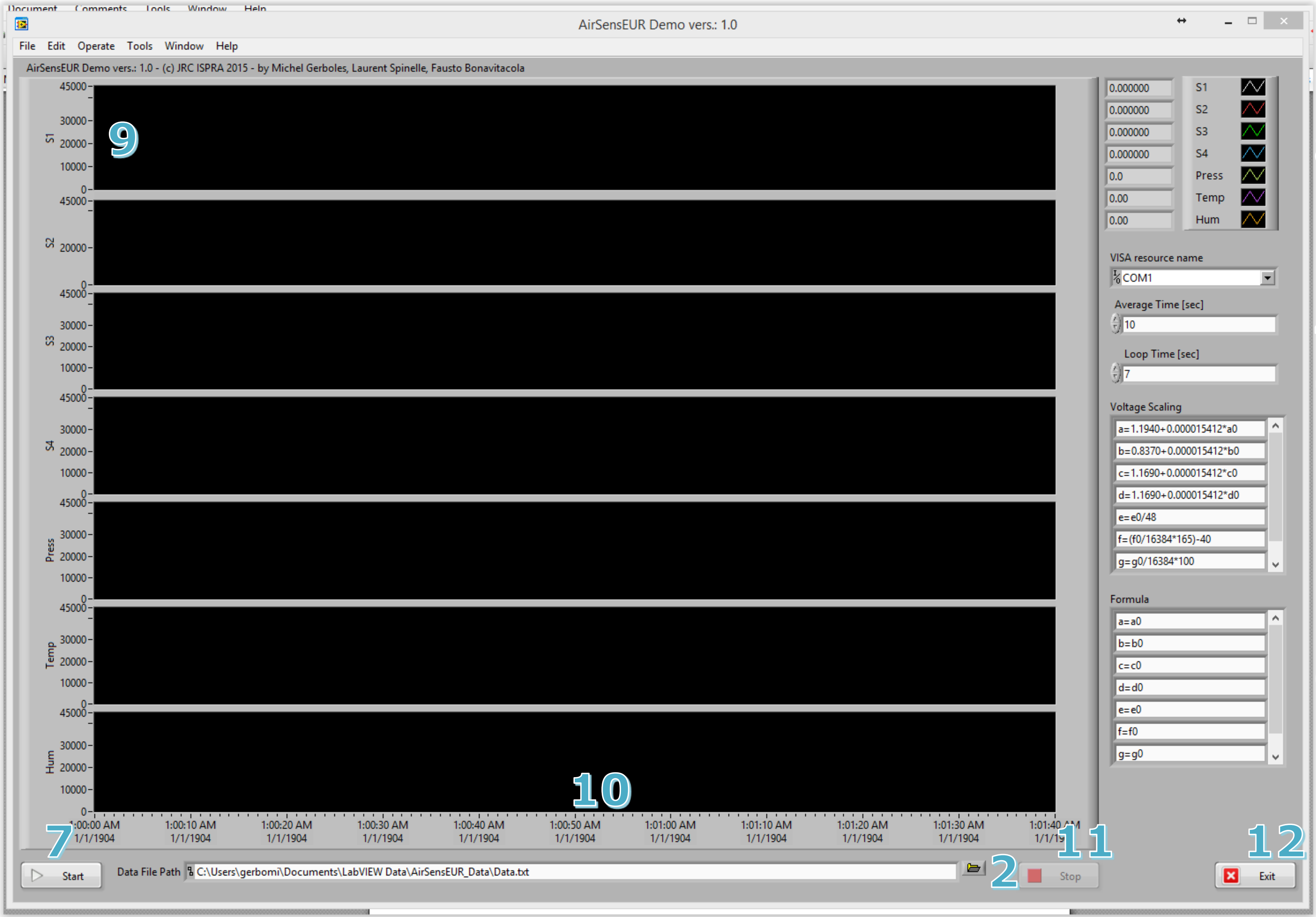
1. Connect the daughter board of the AirSensEUR shield to the PC with a USB cable. Goes to device manager to detect the COM number of the serial port observe the COM number.
2. Run AirSensEURDemo.exe in C:\Program Files (x86)\AirSensEUR Demo (this link will depend on the directory set during Installation in step 2 and the operative system).



(C:) ▸ Program Files (x86) ▸ AirSensEUR Demo

Name	Date modified	Type	Size
data	11/23/2015 3:11 PM	File folder	
AirSensEURDemo.aliases	11/20/2015 3:03 PM	ALIASES File	1 KB
AirSensEURDemo.exe	11/20/2015 3:03 PM	Application	859 KB
AirSensEURDemo.ini	11/20/2015 3:03 PM	Configuration sett...	1 KB

3. The AirSensEUR Demo main window opens as shown next page:



AirSensEUR Demo vers.: 1.0

AirSensEUR Demo vers.: 1.0 - (c) JRC ISPRA 2015 - by Michel Gerboles, Laurent Spinelle, Fausto Bonavitacola

0.000000 S1
0.000000 S2
0.000000 S3
0.000000 S4
0.0 Press
0.00 Temp
0.00 Hum

VISA resource name: COM1

Average Time [sec]: 10

Loop Time [sec]: 7

Voltage Scaling

a=1.1940+0.000015412*a0
b=0.8370+0.000015412*b0
c=1.1690+0.000015412*c0
d=1.1690+0.000015412*d0
e=e0/48
f=(f0/16384*165)-40
g=g0/16384*100

Formula

a=a0
b=b0
c=c0
d=d0
e=e0
f=f0
g=g0

Data File Path: C:\Users\gerbomi\Documents\LabVIEW Data\AirSensEUR_Data\Data.txt

9.4 Running AirSensEUR DEMO vs 1.0

1. Select the correct COM port
2. Select the directory and file name for the data file
3. Minimum data acquisition periodicity >= 7 sec
4. Minimum averaging time >= 10 sec
5. These are the equations for the conversion from analogic to digitally converted values to sensor voltages for the O3 E1F (S1), NO2 E50 (S2), NO E100 (S3) and CO E300 (S4) sensors (in this order) with the configuration parameters given in Gerboles M., et al, AirSensEUR: an open data/software /hardware multi-sensor platform for air quality monitoring. Part A: sensor shield, doi: 10.2788/30927, http://publications.jrc.ec.europa.eu/repository/handle/JRC97581
6. Use this equations for calibration of the sensors S1-S4. The parameters with 0 subscript, as for example a_0, represents the values converted in point 5 for each channel. You can for example apply a correction of NO2 on O3 or apply a temperature, humidity correction.
7. Click to start the data acquisition
8. After a few averaging time, data will appear
9. The 7 graphs with start displaying (S1 to S4 are for the 4 sensors). These graphs behave as LabView graphs, you can right clicks the y-axis and change their layout.
10. You can also right click the x-axis to change the layout of the plots.
11. Click to stop data acquisition
12. Click to exit AirSensEUR DEMO

10 Conclusion

With the “Part B: Host, influx datapush and assembling of AirSensEUR” report, all information has be made public about the electronic boards, the internal software and the assembly of the AirSensEUR Platform together with the technical drawings of the sensor box.

A lot of information has also been provided about the electronic design, the structure of the internal software, the procedure for the first start-up of AirSensEUR and its configuration using the available web server application.

The AirSensEURPanel, the software used for the configuration of the sensor shield and data acquisition was revised in order to simplify the whole process, while the high level of flexibility remains.

A LabView executable software has been developed for the simple calibration of the sensor shield either in lab or in field.

The first datapush protocol (Influx) has been documented together with an example of web dashboard design using the graphana environment.

The next reports of the AirSensEUR series will (i) describe the data push according to the transactional Sensor Observation Service protocol (SOS-T), an implementation of the INSPIRE Directive, and (ii) will propose a sensor calibration method.

List of figures

List of tables

LD-NA-28054-EN-N

JRC Mission

As the Commission's in-house science service, the Joint Research Centre's mission is to provide EU policies with independent, evidence-based scientific and technical support throughout the whole policy cycle.

Working in close cooperation with policy Directorates-General, the JRC addresses key societal challenges while stimulating innovation through developing new methods, tools and standards, and sharing its know-how with the Member States, the scientific community and international partners.

Serving society
Stimulating innovation
Supporting legislation

Publications Office

doi:10.2790/214743

ISBN 978-92-79-60980-0