Documentation OpenAir

Executive Summary

Contains information concerning structure, setup and operation of the OpenAir Hardware Prototype and Firmware.

Version History

Version	Datum	Author	Description
1.0	2019-05-23	MKR, TBE	Initial creation
1.1	2019-06-07	TBE	Updates to sensor values, host software
1.2	2019-07-01	TBE	Corrections, new tags for second bme sensor, footprint expansion connector, numbering of quadsense
1.3	2019-09-06	TBE	Config menu filtering explained

Contents

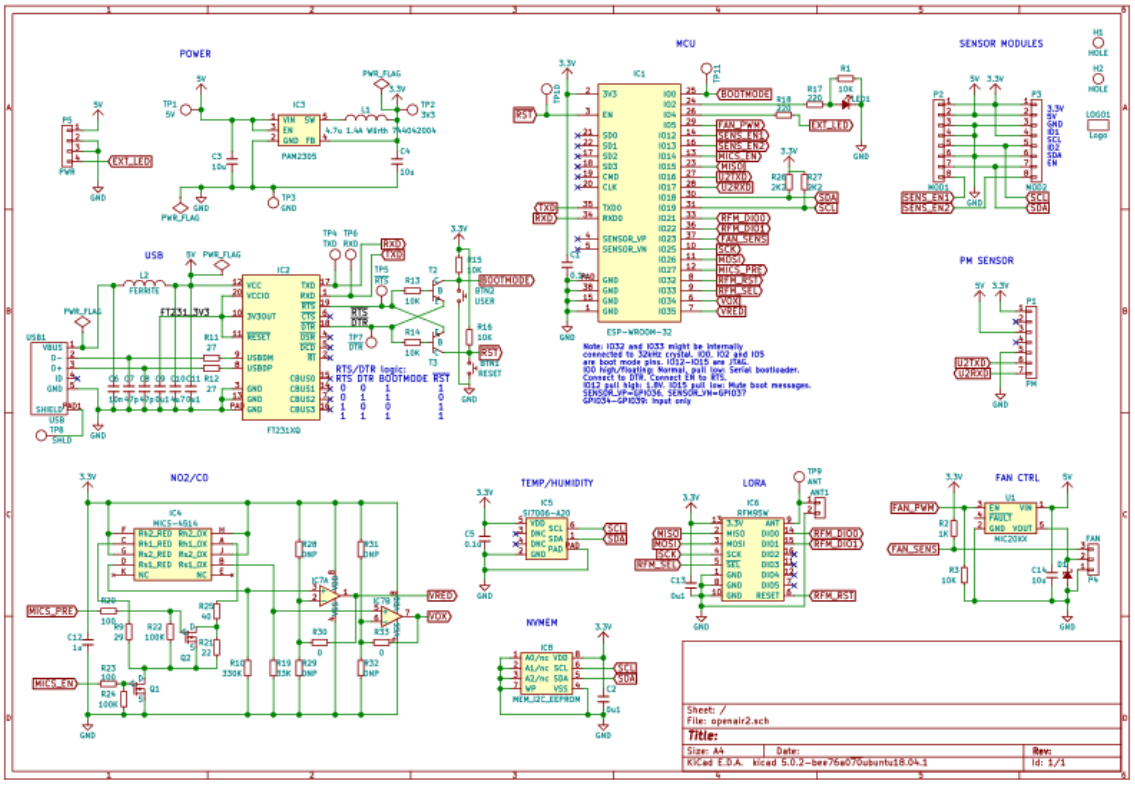
Introduction

This documentation is intended as an operating manual for the OpenAir air quality sensor boards, specifically in the configuration provided to Forschungszentrum Jülich.

Hardware

Board Layout

Below is a copy of the copy of the schematics of the OpenAir Board, provided for easy reference. A high resolution PDF of the schematic should be part of the documentation package containing this document. The OpenAir Board is Open Hardware and the KiCAD source files of the board are available in the project repository [OpenAir PCB Github]



Core of the board is a ESP32 chip on a ESP32-WROOM-32 module described in the following section.

Module Pinout

PIN / Name	Signal	Type	Comment

0 / PAD	GND	P	Ground
1 / GND	GND	P	Ground
2 / 3V3	3V3	P	Supply Power
3 / EN	!RST	I	Module Enable, active high. Default HIGH, may be pulled low by BTN1
4	NC	NC	
5	NC	NC	
6 / IO34	VOX	I	MICS Sensor Input VOx
7 / IO35	VRED	I	MICS Sensor Input VRed
8 / IO32	RFM_RST	I/O	RFM95 LORA Module
9 / IO35	RFM_SEL	I/O	RFM95 LORA Module
10 / IO25	SCK	I/O	SPI serial clock, RFM95
11/ IO26	MOSI	I/O	SPI MOSI, RMF95
12 / IO27	MICS_PRE	I/O (O)	MICS Sensor Preheater
13 / IO14	MICS_EN	I/O (O)	MICS Sensor Enable
14 / IO12	SENS_EN1	I/O (O)	Enable external sensor module
15 / GND	GND	P	Ground
16 / IO13	SENS_EN2	I/O (O)	Enable external sensor module
17	NC	I/O(NC)	
18	NC	I/O(NC)	
20	NC	I/O(NC)	
21	NC	I/O(NC)	
22	NC	I/O(NC)	
23 / IO15	MISO	I/O (I)	SPI MISO, RFM95
24 / IO2	LED	I/O (O)	LED
25 / IO0	BOOTMODE	I/O (I)	BOOTMODE selector, typically HIGH, may be pulled low by BTN2
26 / IO4	EXT_LED	I/O (O)	External LED connector
27 / IO16	U2TXD	I/O (O)	TX of 2nd UART (typically SDS 1011 particulate sensor)

28 / IO17	U2RXD	I/O (I)	RX of 2nd UART
29 / IO5	FAN_PWM	I/O (O)	Control of fan speed
30 / IO18	SDA	I/O	I2C SDA
31 / IO19	SCL	I/O (O)	I2C clock
32	NC	NC	
33 / IO21	RFM_DIO0	I/O	RFM95 LORA Module
34 / RXD0	RXD	I/O	Primary UART, connected to FTDI for host programming
35 / TXD0	TXD	I/O	Primary UART
36 / IO22	RFM_DIO0	I/O	RFM95 LORA Module
37 / IO23	FAN_SENSE	I/O (I)	Fan signal pin
38	GND	P	Ground

Pin numbers refer to pins on the module. Programming APIs typically expect the pin names. E.g. configuring the FAN_SENSE would require 23 (IOS23) as configuration parameter not 37 (the position on the ESP32 module).

Type is: NC=not connected, P=Power, I=Input, I/O=Input/Output, first value are possible configurations, usage on OpenAir board are in (parentheses).

Onboard Peripherals

USB transceiver

The board contains a micro USB port connected via an FT231 USB Serial Adapter to one of the ESP32 module's UART ports for programming the device without the use of additional programming adapters. Besides programming and debugging, the serial connection can also be used for communication between the OpenAir board and a PC-based host.

MICS-4514

The board contains a footprint for an optional MICS 4514 NO2/CO Sensor [MICS 4514].

SI7006-A20

The board contains a footprint for an optional Si 7006 A20 [Si 7006 A20] Humidity and Temperature sensor.

NV Memory

The board contains a SO-8 footprint for external EEPROM memory. E.g. [CAT24C64-D].

LORA

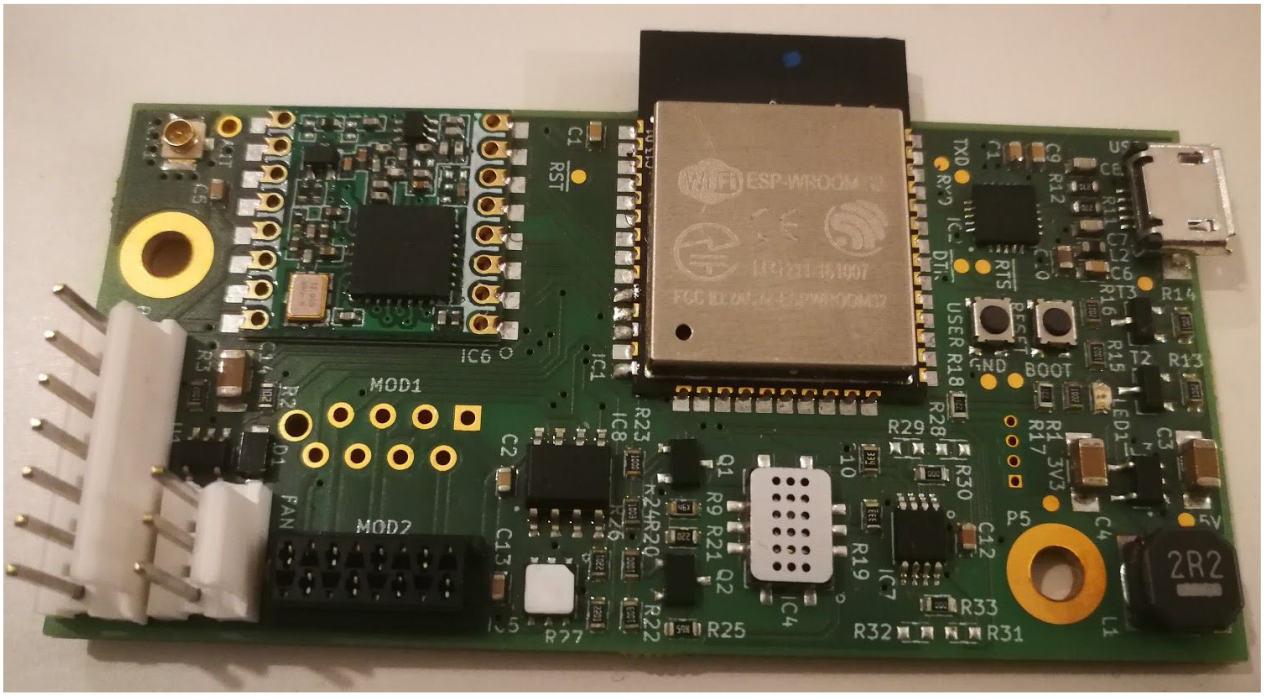
The board contains a footprint for an optional HopeRF RFM95W [RFM95w] LoRa transceiver.

Fan driver

The board contains driver circuitry for 5V fans. A high side switch can be used to enable and disable the fan or to specify fan power by a low frequency PWM signal. A “tachometric” pin can be connected to provide fan speed feedback to the system. A flyback diode protects the rest of thee circuit from inductive motor loads.

Misc

Furthermore, the board contains an onboard LED as well as a connection for an external LED, two buttons (one of which may be used, the other is connected to the “module enable” pin to provide reset.) The board contains an external UART, mainly for connecting a SDS011 PM sensor, and two expansion board connectors, described in the following section.



OpenAir board with connectors, left to right

- LoRa antenna u.fl connector (top left), mounting hole for solder whip antenna
- 2.54mm pitch 7 pin UART/SDS011 connector (bottom left)
- 2.54mm pitch 3 pin fan connector
- 2 x 8 pin module connector (one populated, one unpopulated)
- 4 pin 1.27mm pitch external power and LED connector (bottom right, unpopulated)

- Micro USB connector (top right)

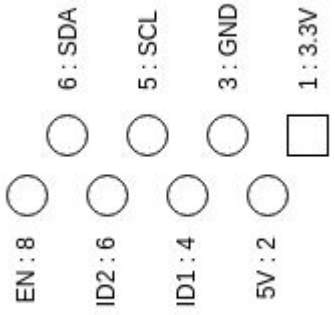
The ESP32 module featured in the photo contains an onboard PCB antenna for 2.4GHz communication (WIFI / Bluetooth LE). It can alternatively be populated with an ESP32 module with an u.fl antenna connector instead of the built-in antenna.

Expansion Boards

Module Connector

The OpenAir board contains two 8-pin module connectors, intended to provide a general purpose interface for connecting additional sensors. The pinout is as follows:

1	3.3V	Power	3.3V power supply from OpenAir to module
2	5V	Power	Unregulated 5V power supply from OpenAir to module
3	GND	Power	Common power and signal ground
4	ID1	Out	Module connector identification
5	SCL	Out	I2C clock
6	ID2	Out	Module connector identification / RFU
7	SDA	IO	I2C data
8	EN	Out	Module enable



The module communicates with the OpenAir board via I2C. The OpenAir board acts as bus master, the module contains one or more I2C slaves. The ID1 pin is tied to GND on module connector 1 and to 3.3V on module connector 2. Modules may use this signal to change their I2C addresses based on the connector they plugged in, avoiding address collisions when two identical or similar modules are connected simultaneously. ID2 is currently tied to GND, but may be used in future updates to either provide up to four different module identification options or for another purpose. The enable signal

may be used by modules to power down circuitry when not in use. This is especially useful for sensors with high power requirements to reduce dissipated heat or in battery powered applications.

The pinout also allows for a four pin connection with a standard 2.54mm header. In this case, only pins 1,3,5 and 7 are used. This may be useful for small, low cost, low power sensors.

Note that the provided power rails cannot be used directly for low noise analog circuitry. The 3.3V rail is shared with OpenAir digital circuitry and the 5V rail is directly forwarded from the OpenAir power input. For precise voltage references and low noise circuits, further filtering or voltage generation is required.

Mechanical connectors are small form factor, IDC-style plugs that mate with standard 1.27mm ribbon cable. Besides the electrical connection, the module connector consists of a recommended mechanical size (100 x 90mm) and recommended mounting holes (M3, 84mm square).

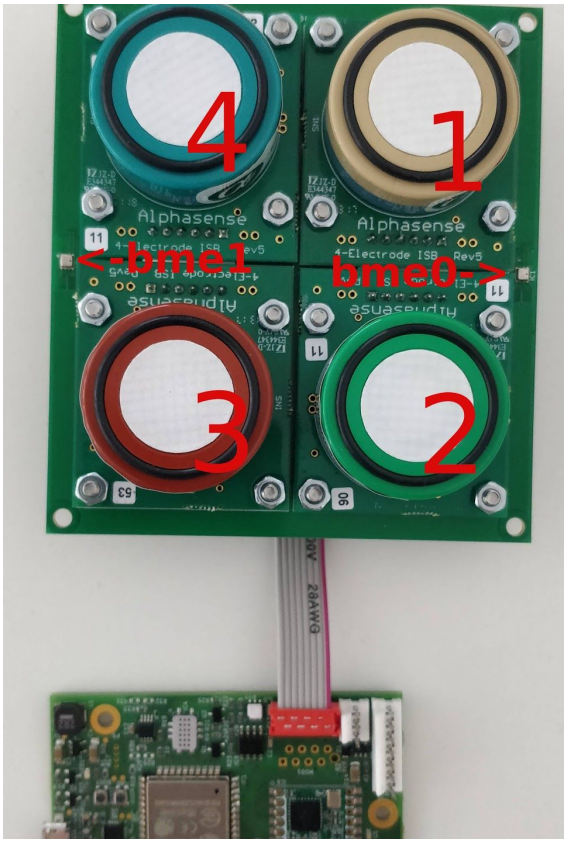
Quadsense

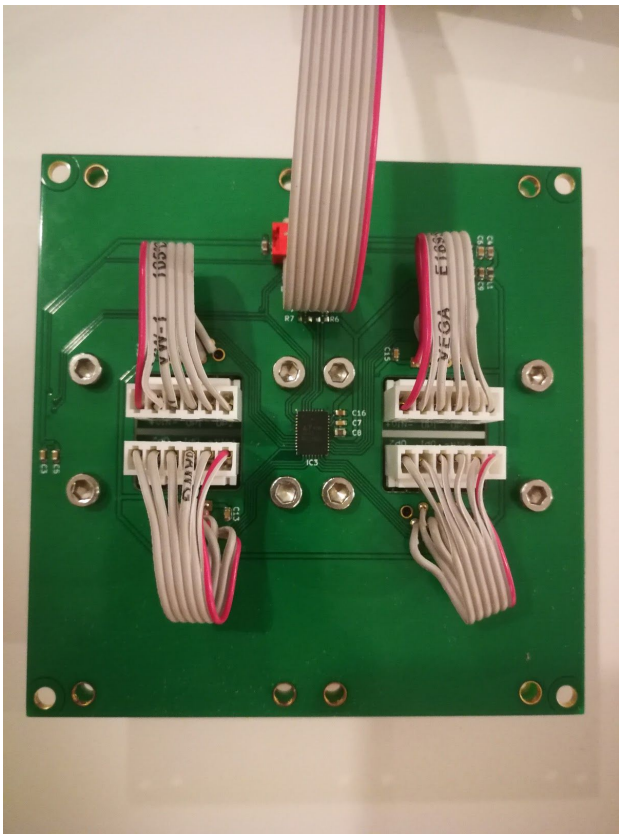
The Quadsense expansion board contains a Bosch BME280 humidity / pressure / temperature sensor, an Analog Devices LTC2497 low noise eight channel 16 bit differential analog to digital converter [LTC2497], four Alphasense B4-type gas sensors with Alphasense ISB support boards and a separate 4.11V voltage regulator. The voltage is used to generate a low noise power supply for the LTC2497. It is further filtered down in order to provide a clean analog power supply for the Alphasense potentiostat circuits.

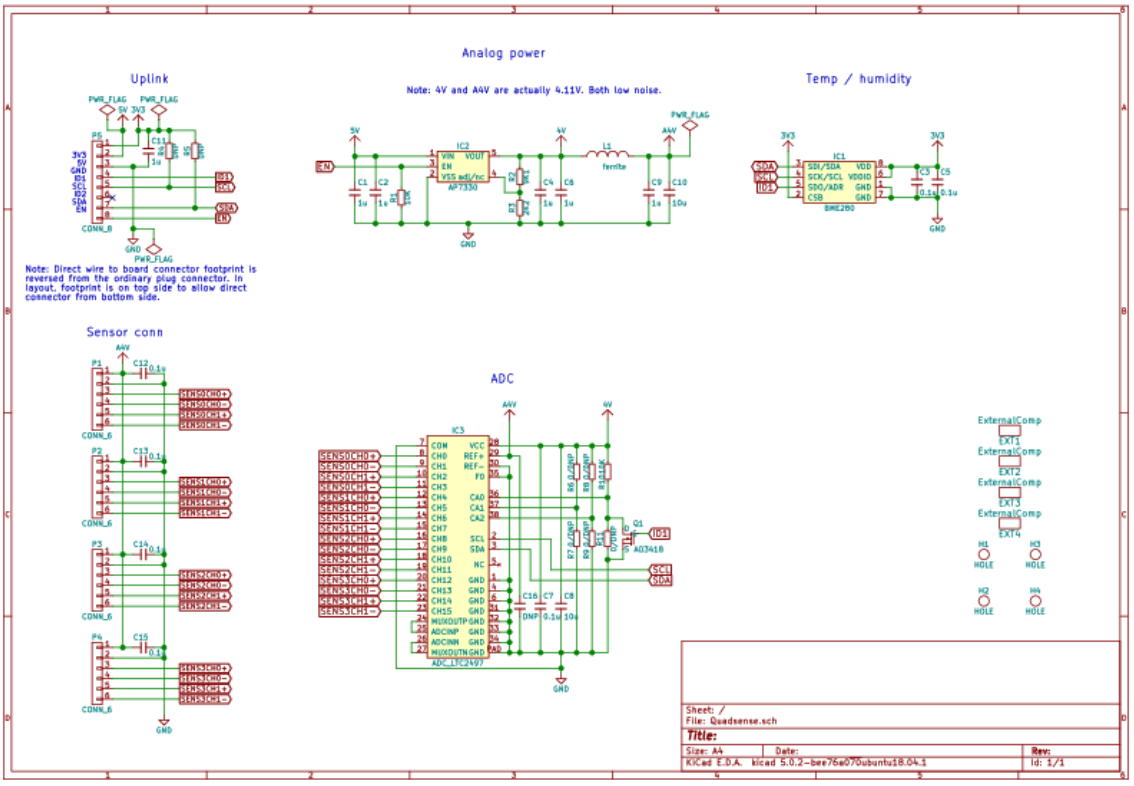
Besides hosting the support circuitry, the board doubles as a mechanical mount for the Alphasense sensors. To protect the circuitry from environmental factors, the board is mainly populated on the back side (with the exception of the BME280 sensor).

Note: The footprint of the BME280 sensor is mirrored on the first revision of the board, so it is not directly soldered to the board but connected through wires instead.

The following photos show the Quadsense board, top and bottom. Most of the space is obviously used for mounting and holding the Alphasense sensor boards. Their bottom facing connectors connect through the Quadsense board.


4
1
3
2
<-bme1
bme0->
Alphasense
4-Electrode ISB Rev5
40AWG 28AWG




Uplink
Analog power
Note: 4V and A4V are actually 4.11V. Both low noise.
Temp / humidity
Note: Direct wire to board connector footprint is reversed from the ordinary plug connector. In layout, footprint is on top side to allow direct connector from bottom side.
Sensor conn
ADC
ExternalComp
Sheet: /
File: Quadsense.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 5.0.2-bee76e0~ubuntu18.04.1
Id: 1/1

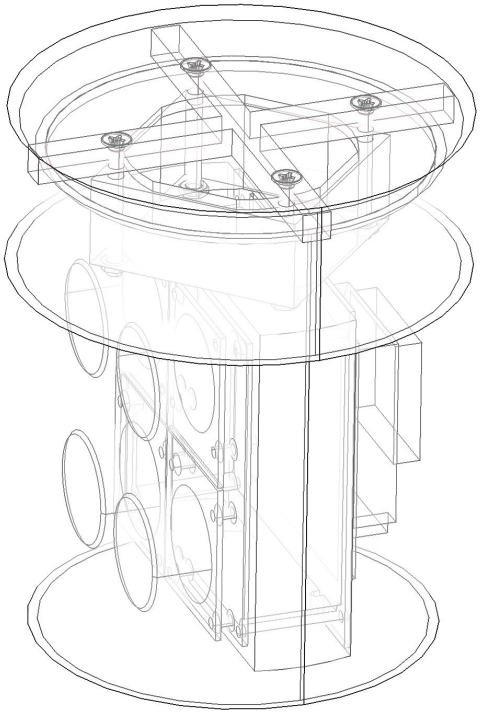
UART / Nova SDS 011

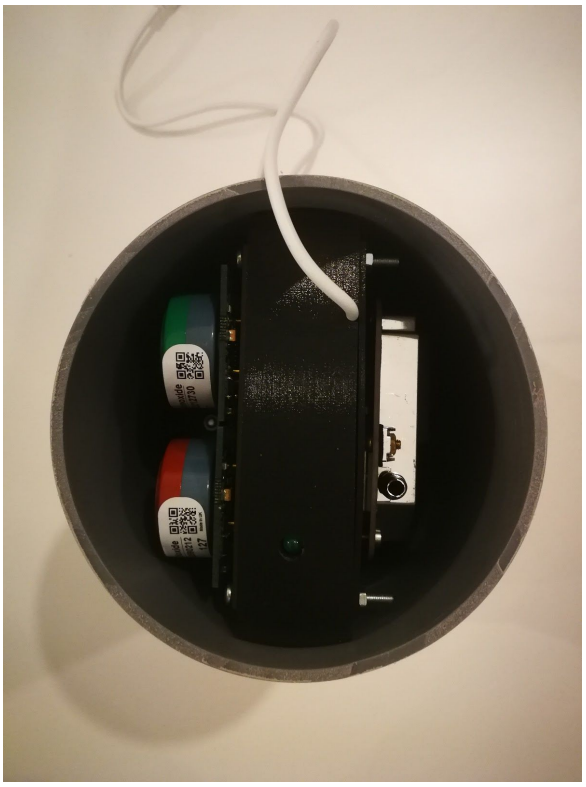
The OpenAir board contains a 7-pin connector with 5V power supply and UART. The pinout corresponds directly to the Nova SDS011 particle matter sensor. The OpenAir board mounting holes are designed so that it can be mechanically connected back-to-back to the SDS011.

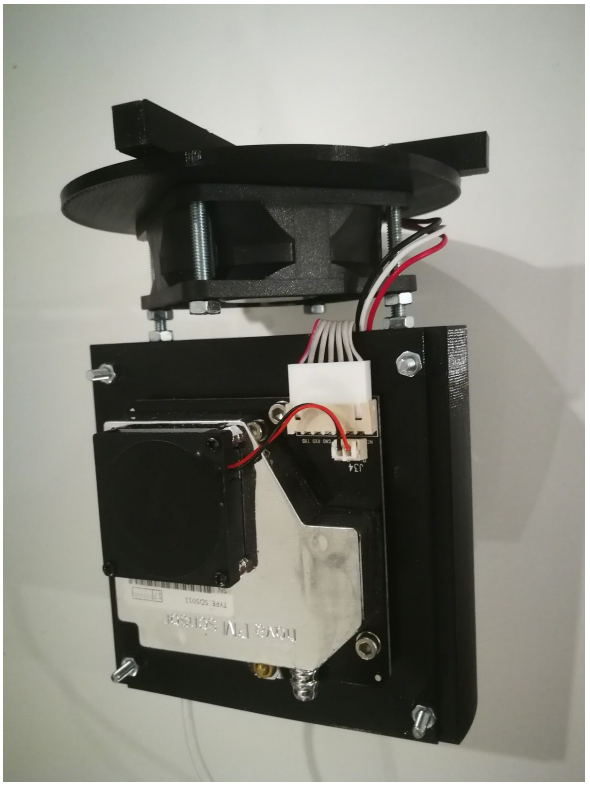
Enclosure

The OpenAir board can be mounted together with an expansion board and a SDS011 sensor into a compact, waterproof enclosure. Outer parts of the enclosure are made from plastic construction tubing (~15cm DN110 tube and a DN125 end cap). Inner parts are 3d printed, but can also be made with other production techniques, e.g. laser cut or injection molding.

The OpenAir controller itself sits in a frame that forms an enclosure together with two extension modules. The enclosure splits the DN110 tube into two circle segments, exposed to air flowing from bottom to top, exposing the sensors to ambient air close to the intake. A barrier with a fan above the main enclosure ensures a steady air flow. A cross-shaped spacer at the top of the inner unit extends beyond the DN110 diameter but fits into the DN125 cap. It holds the inner unit in place, centers the DN125 cap above it and forms an air exhaust path.







Firmware

Mongoose OS

The OpenAir device firmware is based on Mongoose OS [Mongoose]. The firmware source code is available [OpenAir Firmware] under the Apache 2 Open Source License.

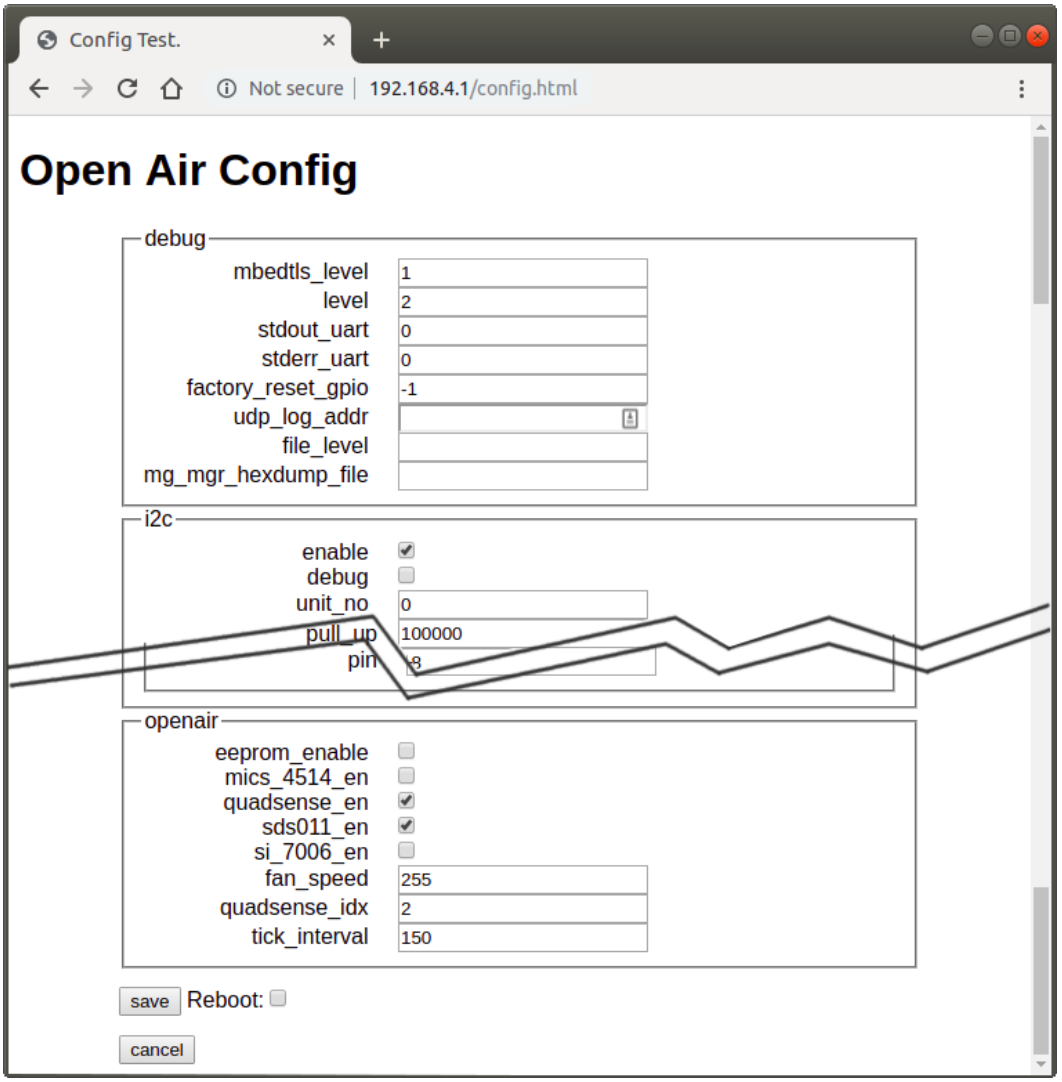
Initial Provisioning

The OpenAir board boots automatically as soon as it is powered. Power is delivered to the board via the onboard micro USB connector. In case no WIFI Access Point (AP) to connect to is configured, the device itself will deploy an Access Point[1]. The SSID of the AP will be `OpenAir_XXXXXXXXXXXX` (Xs will be replaced by the devices MAC address). Connecting an external device to this AP will allow you to connect to web interface of the OpenAir board via HTTP using the url:

```
http://192.168.4.1/config.html
```

The web interface is a rudimentary interface that allows for configuration of the device as well as access to the last values read by the sensors. The OpenAir specific configuration values are described in the Configuration section below.

[1] Currently (2019-05) the AP will *always* be available for convenience, even if an external access point is configured.



The configuration page of the web interface allows configuration of all the system's properties. The "cancel" button resets the current state of the interface (i.e. it does not revert to factory settings!). Most configurations require a reset to take effect (of the OpenAir entries, only the fan_speed setting is honored immediately.) Check the 'reboot' box in case the device should automatically reboot after saving the configuration.

By default only the configuration sections wifi, openair and quadsense are displayed in order to reduce clutter. The search string "advanced" may be appended to the interface URL to allow access to allow configuration entries:

```
http://192.168.4.1/config.html?advanced
```

WiFi Configuration

Once the SSID and password of an external access point have been configured, the device is capable of general IP based access. The IP address assigned to the device *should* be resolvable via using the device.id , i.e. `esp32_XXXXXX` (with X being the final digits of the mac address) configuration value as

the hostname. This is highly dependant on the configuration of the access point the device is connected to. At the moment the device remains in dual AP and station mode. While this is practical for local testing, it is not suitable for production use. All endpoints described in this document refer to `http://192.168.4.1/xyz.html` but would be available in the local network using the assigned ip address.

Up to three different SSIDs may be configured for the device in station mode.

Configuration

OpenAir Specific Configuration

Configuration Name	Type	Default	Comment
eeprom_en	bool	false	Indicate whether EEPROM is available on board
fan_speed	number	255	Speed of fan 0=off 255=full
mics_4514_en	bool	false	Indicate whether a MICS sensor is available on the board
mics4514_interval	number	1000	Number of milliseconds between subsequent mics measurements
quadsense_en	bool	true	Indicate availability of quadsense expansion board.
quadsense_interval	number	1000	Number of milliseconds between subsequent gas sensor measurements
quadsense_idx	number	2	The expansion port the quadsense module is installed on
quadsense_bme0_en	bool	true	Enable first bme temp/humidity/pressure sensor on quadsense
quadsense_bme1_interval	number	1000	Number of milliseconds between subsequent BME measurements
quadsense_bme1_en	bool	true	Enable second bme temp/humidity/pressure sensor on quadsense
quadsense_bme1_interval	number	1000	Number of milliseconds between subsequent BME measurements
rfm95_en	bool	false	Indicate whether RFM95w is available on board.
sds011_en	bool	true	Indicate whether SDS011

			particulate sensor is installed.
sds011_interval	number	1000	Number of milliseconds between subsequent SDS measurements
si_7006_en	bool	false	Indicate whether a the Silicon Labs humidity & temperature sensor is installed
si7006_interval	number	1000	Number of milliseconds between subsequent SI7006 measurements
tick_interval	number	150	Tick interval in ms, internal config
firehose_en	bool	true	Whether to dump all sensor readings to the network. See Host Software below
firehose_addr	string	192.168.4.2:7531	The IP address of the host to send sensor readings to. The default is the first address assigned to connected host if operating in AP mode. This may be updated to correspond to a local ip or dns name if connecting as STA

General Configuration

The following table contains an extract of available configuration entries for the Mongoose operating system that are particularly relevant. Please refer to [Mongoose] for a full description of all configuration entries.

Configuration Name	**Type**	**Default**	**Comment**
debug.level	number	2	The log level describing the granularity of console logs: LL_NONE = -1, LL_ERROR = 0, LL_WARN = 1, LL_INFO = 2, LL_DEBUG = 3, LL_VERBOSE_DEBUG = 4
wifi.ap.pass	string	openair123	The password of the access point opened by the OpenAir board.
wifi.ap.keep_enablled	bool	true	Whether to keep AP enabled when station is on
wifi.staX			Up to 3 external APs can be configured to be used in station mode. The configuration of each is

			identical. Configuration is in order of priority: STA then STA1 then STA2. Apart from the SSID and password, general IP information can be configured.
wifi.staX.enable	bool	false	Whether to use this configuration.
wifi.staX.ssid	string	*empty*	SSID of AP to attempt to connect to
wifi.staX.pass	string	*empty*	Password of AP

Interface

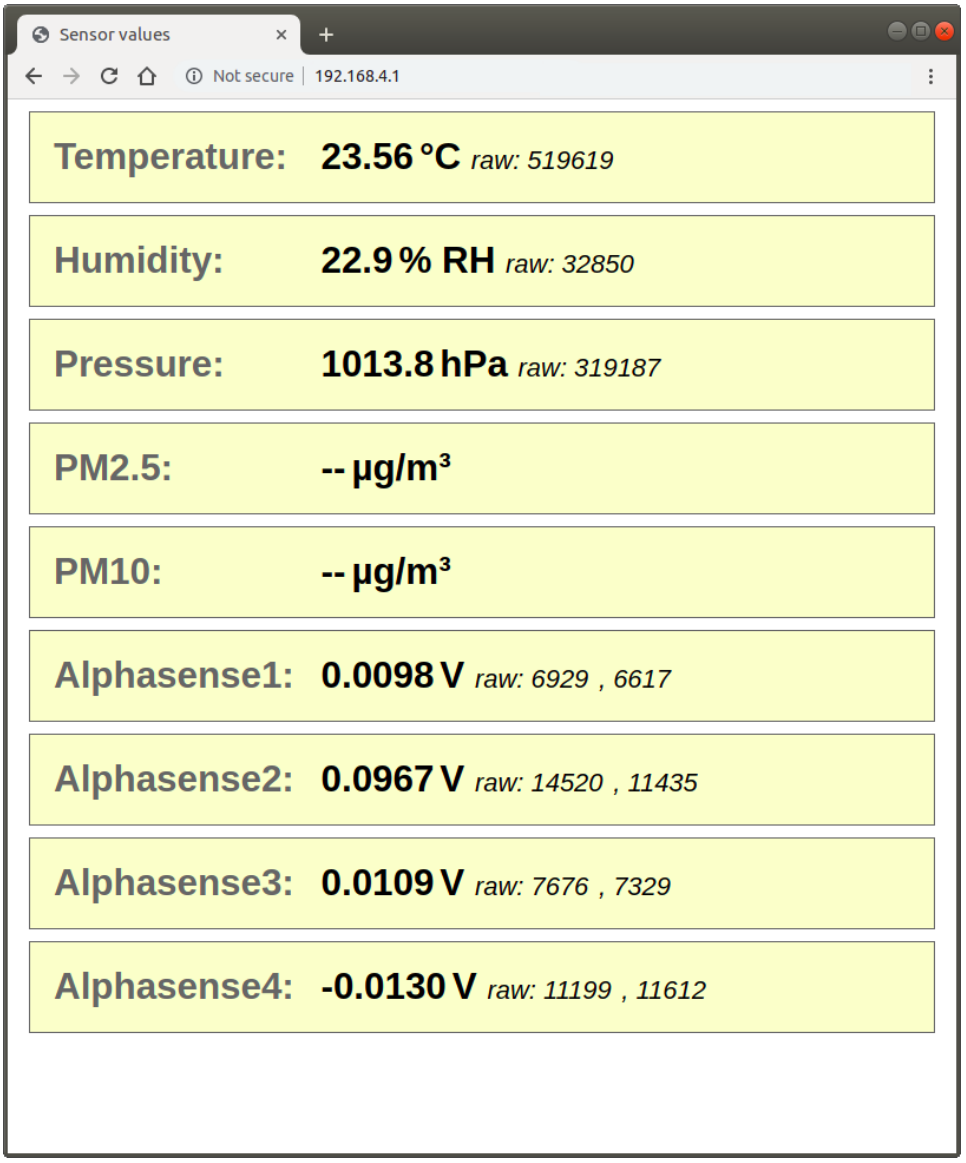
The OpenAir device currently support three main functional interface: retrieval of sensor data, configuration of settings and finally firmware updates. Each functional group has an http endpoint assigned to it, as well as automated mechanisms:

Function	**URL**	**Comment**
sensor data	http://192.169.4.1/index.html	Retrieval of the last read sensor values
configuration	http://192.169.4.1/config.html	Configuration page, described above.
firmware update	http://192.169.4.1/fw.html	Firmware upload.

Mongoose OS provides a tool called mos that is available for Windows, macOS and Linux which may be used to access the device directly. The mos tool is beyond the scope of this documentation. Instructions to install the tool may be found at [mos tool].

Sensor Data

Sensor data may be retrieved via the web interface (`/index.html`):



Furthermore a REST endpoint exists to retrieve data in machine readable form. This endpoint can be called via the URL:

`http://192.168.4.1/rpc/openair.current`

Calls to this URL contain JSON, as follows:

```
$ curl http://192.168.4.1/rpc/openair.current
{
   "device_id" : "esp32_0AAEAC",
   "ts" : 2537,
   "entries" : [
      {
         "ts" : 0,
         "tag" : 0,
         "value" : 0
      },
```

```
      {
         "value" : 6938,
         "ts" : 2536,
         "tag" : 1
      },
(...)
    {
         "value" : 0,
         "ts" : 0,
         "tag" : 16
      }
   ]
}
```

Values reflect the last performed measurement. The ts value is the number of seconds since the device was booted. The tags value are described in the following table:

Tag Value	Tag Name[2]	Description
0	`oa_time`	Reserved for time (retrieved from server)
1	`oa_alpha_1`	Alphasense 1.1 (depends on configuration of sensors on the Quadsense board)
2	`oa_alpha_2`	Alphasense 1.2
3	`oa_alpha_3`	Alphasense 2.1
4	`oa_alpha_4`	Alphasense 2.2
5	`oa_alpha_5`	Alphasense 3.1
6	`oa_alpha_6`	Alphasense 3.2
7	`oa_alpha_7`	Alphasense 4.1
8	`oa_alpha_8`	Alphasense 4.2
9	`oa_bme0_pressure_raw`	
10	`oa_bme0_pressure`	Derived PA value
11	`oa_bme0_temp_raw`	
12	`oa_bme0_temp`	Derived millikelvin value
13	`oa_bme0_humidity_raw`	
14	`oa_bme0_humidity`	Per mille rh

[2] Tag values are extracted from their definition in `src/broker.h` of the firmware source code [OpenAir firmware].

15	`oa_bme1_pressure_raw`	
16	`oa_bme1_pressure`	Derived PA value
17	`oa_bme1_temp_raw`	
18	`oa_bme1_temp`	Derived millikelvin value
19	`oa_bme1_humidity_raw`	
20	`oa_bme1_humidity`	Per mille rh
21	`oa_sds_pm25`	PM2.5 in ng/m^3
22	`oa_sds_pm10`	PM10 in ng/m^3
23	`oa_si7006_temp_raw`	Temperature measurement of on board si7006 sensor
24	`oa_si7006_temp`	
25	`oa_si7006_rh_raw`	humidity measurement of on board si7006 sensor
26	`oa_si7006_rh`	
27	`oa_mics4514_vred`	Mics 4514 raw vred measurement
28	`oa_mics4514_vox`	Mics 4514 raw vox measurement

Finally, an option exists to dump all sensor values onto a tcp stream. See Host Software section, below.

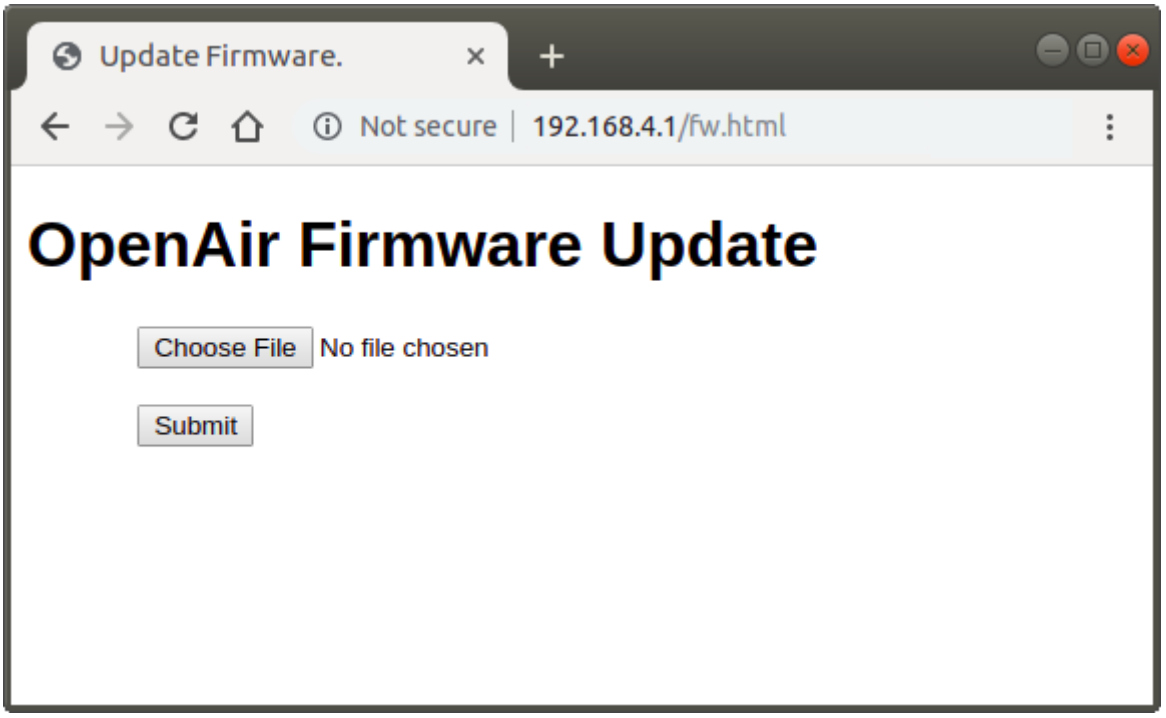
Configuration

Configuration via web interface is described in the Provisioning section above. Configuration is maintained via the [Mongoose] configuration system and may be modified by http calls or using the `mos` tool. (`mos get-config` and `mos set-config`)

Firmware Updates

Firmware updates can be applied to the device via http using the url:

`http://192.168.4.1/fw.html`



New firmware images will be supplied in the form of a zip archive, select the archive using the file chooser and upload it. In case of a successful update, the page displays a notice (after roughly 1 minute). In case the upload was not success, the device “hangs”, currently the upload mechanism seems to be less reliable than is desirable. In case the upload does not complete reboot the device, the update mechanism ensures that the current firmware remains available until the next version has been successfully installed.

Alternatively, the new firmware may be uploaded manually:

```
$ curl -i -F filedata=@./build/fw.zip  http://192.168.4.1/update
HTTP/1.1 200 OK
Server: Mongoose/6.10
Content-Type: text/plain
Connection: close

Update applied, finalizing
```

or using the mos tool: `mos flash fw.zip`

Host Software

In the current state of the software a host program (`firehose_server`) is provided for Linux, macOS and Windows to collect all sensor readings from the OpenAir device. The software is a command line program with the following options

```
$ ./firehose -help
```

```
Usage of ./firehose:
  -a string
  -addr string
        address for server to listen on (default ":7531")
  -h
  -help
        print usage
  -o string
  -outfile string
        filename to save output to (default "-")
```

By default the software listens to port 7531 on all interfaces, using this option, it can be restricted to a single IP address, e.g. the address assigned by OpenAir if connecting to the AP provided by the device.

The software receives sensor readings (currently as a TCP Stream of JSON packets) and outputs a CSV file for use in Excel, etc. By default the CSV entries are printed to STDOUT, but a filename to save the csv can be provided using the outfile parameter.

The program can receive and process output from multiple OpenAir devices. These devices must be configured to use the same server using the `firehose_addr` configuration parameter.

The format of the CSV is as follows:

```
server_time,timestamp,device_id,tag,value(hex),value(decimal),tag_annotation,value_annotation
1559312722,0,esp32_0AAEAC,0xffffffff,0x00000000,0,OA_Network_Events,CONNECT
1559312722,992,esp32_0AAEAC,0x00000009,0x0004d243,315971,OA_BME_Pressure_Raw,(raw 315971)
1559312722,992,esp32_0AAEAC,0x0000000a,0x00018e28,101928,OA_BME_Pressure,1019.28 hPa
1559312722,992,esp32_0AAEAC,0x0000000b,0x000801dc,524764,OA_BME_Temp_Raw,(raw 524764)
1559312722,992,esp32_0AAEAC,0x0000000c,0x00048d3b,298299,OA_BME_Temp,25.15 C
...
```

Field	Description
`server_time`	Time the package was received by the server (unix timestamp, seconds since 1970-01-01)
`timestamp`	OpenAir timestamp of sensor reading on the device, seconds since device boot
`device_id`	String identifying the OpenAir device
`tag`	Tag identifying the source of the reading. For convenience this value is annotated in the field tag_annotation. A list of all tags is provided in a table below.

`value(hex)`	All sensor reading are currently stored as unsigned 32bit values. This field is a hex representation of the value.
`value(dec)`	This field is a decimal representation of the sensor value. Depending on the nature of the data returned by the sensor, this may not be a sensible representation.
`tag_annotation`	Human readable description of the tag
`value_annotation`	Human readable interpretation of the value for convenience.

In addition to the tags described in the Sensor Data section above, the firehose-server logs the following derived and pseudo-tags:

Tag	**Description**
OA_Network_Events	Value CONNECT and DISCONNECT signify time the OpenAir device established a connection or the connection was lost.
OA_AlphaCalc_1 … OA_Alpha_Calc_4	Precalculated (server side) voltage of the ADC reading. This value is calculated by the formula: V = (ALPHA(X-1) - ALPHA(X)) * ADC_CONST For X in 2, 4, 6, 8 ADC_CONST = 0.000031356811523

References

[BME280] Bosch BME280 combined pressure, temp and humidity sensor datasheet (https://sgx.cdistore.com/datasheets/sgx/0278_Datasheet%20MiCS-4514%20rev%2017.pdf)

[CAT24C64-D] EEPROM Datasheet (https://www.onsemi.com/pub/Collateral/CAT24C64-D.PDF)

[LTC2497] Analog Devices LTC2497 ADC Datasheet (https://www.analog.com/media/en/technical-documentation/data-sheets/2497fb.pdf)

[MICS 4514] MICS 4514 MOS Sensor Datasheet (https://sgx.cdistore.com/datasheets/sgx/0278_Datasheet%20MiCS-4514%20rev%2017.pdf)

[Mongoose] General information and project pages for Mongoose OS are here:

	https://mongoose-os.com/mos.html . The repository containing the source code is available here: https://github.com/cesanta/mongoose-os/ .
[mos tool]	Installation instructions for the Mongoose OS mos tool. https://mongoose-os.com/docs/mongoose-os/quickstart/setup.md#1-download-and-install-mos-tool
[OpenAir PCB Github]	Repository containing the OpenAir schematics and board layout files.
[OpenAir Firmware]	Repository containing the source of the firmware: https://github.com/OpenAirCgn/mongoose-firmware
[RFM95]	LoRa Transceiver datasheet (https://www.hoperf.com/data/upload/portal/20190301/RFM95_96_97_98W.pdf)
[Si7006 A20]	Si7006 A20 humidity and temp sensor datasheet (https://www.silabs.com/documents/public/data-sheets/Si7006-A20.pdf)
[SDS011]	Nova SDS011 Particle concentration sensor datasheet (http://ecksteinimg.de/Datasheet/SDS011%20laser%20PM2.5%20sensor%20specification-V1.3.pdf)
[ESP32 WROOM]	Datasheet of ESP module https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf