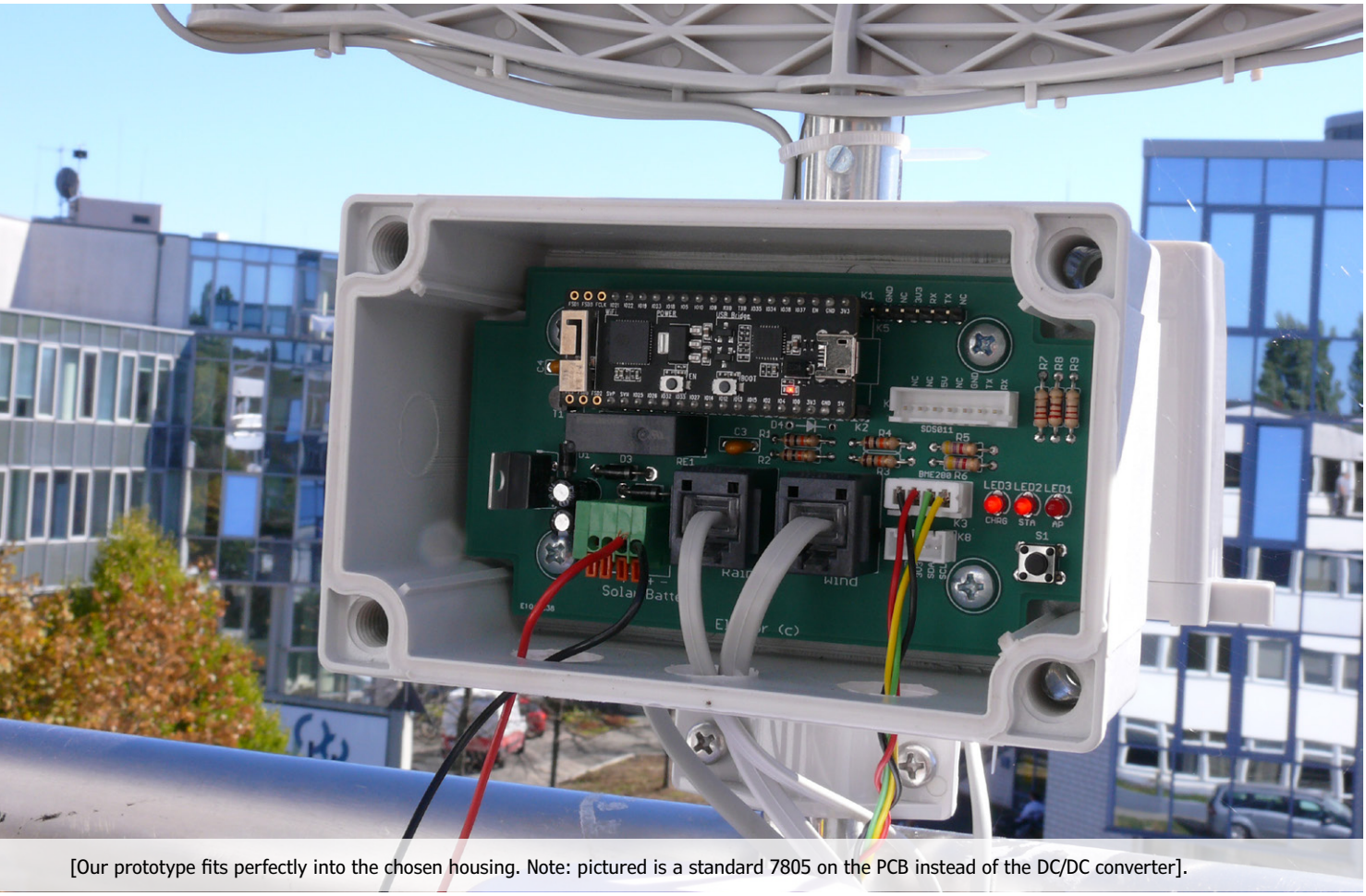
eLABS PROJECT

ESP32 Weather Station

Reading sensor data online

By **Roy Aarts** (Elektor Labs) and **Thijs Beckers** (Editorial)

The ESP32 board is a versatile and very affordable development platform that is quite suitable for all sorts of home automation projects. Here we present a specific application with a weather station built around the ESP32, which makes all data directly available online.



[Our prototype fits perfectly into the chosen housing. Note: pictured is a standard 7805 on the PCB instead of the DC/DC converter].

Features

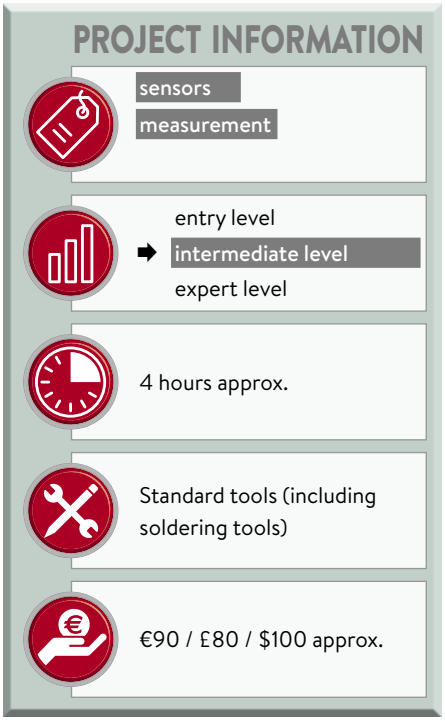
- Measures temperature, wind direction and wind speed, humidity, air pressure and precipitation
- Optional sensor for fine particles: Nova Fitness SDS-011
- Additional ports for Grove sensors or other devices
- Supports Thingspeak and senseBox
- Can be configured on the internal web page of the ESP32
- Operates from a solar panel, 12 V battery, and/or 8-28 V DC adapter

The ESP32 Weather Station measures the usual weather parameters: temperature, wind direction and wind speed, humidity, air pressure and precipitation. With a suitable sensor, such as the Nova Fitness SDS-011 [1] (PM2.5, resolution 0.3 µg/m^3), it can also measure the concentration of fine particles. A Bosch FME280 sensor [2] is used to measure the tem-

perature, humidity and air pressure. For wind speed, wind direction and precipitation we use a weather station kit that is available in the Elektor Store (see the '@ www.elektor.com' inset).
The ESP32 makes the measurements and can upload the resulting data to Thingspeak [3] or senseBox [4]. Thingspeak is an online database operated by Mathworks, where you can upload data and view it nicely plotted in charts. The data can also be processed with Matlab.
SenseBox acts as a sort of open-source synoptic weather map. Users who have a senseBox kit or other senseBox compatible device can upload their measurements, which are then displayed on a world map and visible to everyone.

Hardware

The weather station is built around the ESP32 Pico Kit V4 (see the schematic in **Figure 1**, which handles all the necessary tasks. To make sensor connection and fitting in a waterproof enclosure (a Fibox PC 100/60 HT) a bit easier, we designed a carrier board. The wind and rain sensors are connected to the carrier board through RJ45 connectors, while the BME280 and SDS011 sensors are connected through JST XH connectors. The BME280 sensor uses the I2C bus, and the SDS011 sensor uses a UART port. We also implemented two additional ports on the board to allow supplementary sensors (or other sensors) or peripheral devices to be read. The first additional port consists of a Grove I²C connector for Grove modules from Seeed studio [5, 6]. The second is an FTDI port that feeds out a UART connection.
The circuit can be powered from a 12 V lead-acid battery. If you would like to use a 12 V solar panel to charge the battery, we have a handy built-in feature for you: the ESP32 can close a relay when the battery is low and open it when the battery is again fully charged. The ESP32 uses a voltage divider with resistor values of 470 kΩ and 100 kΩ to measure the battery voltage, so you can adapt the start and stop points for battery charging in the software if necessary. LED3 indicates the battery charging status. We use a simple FET (T1) as a buffer to drive the relay. Diode D3 acts as a snubber to suppress voltage spikes from the relay coil and protect T1 against premature failure. Resistor R10 limits the base current, while R7 limits the current through



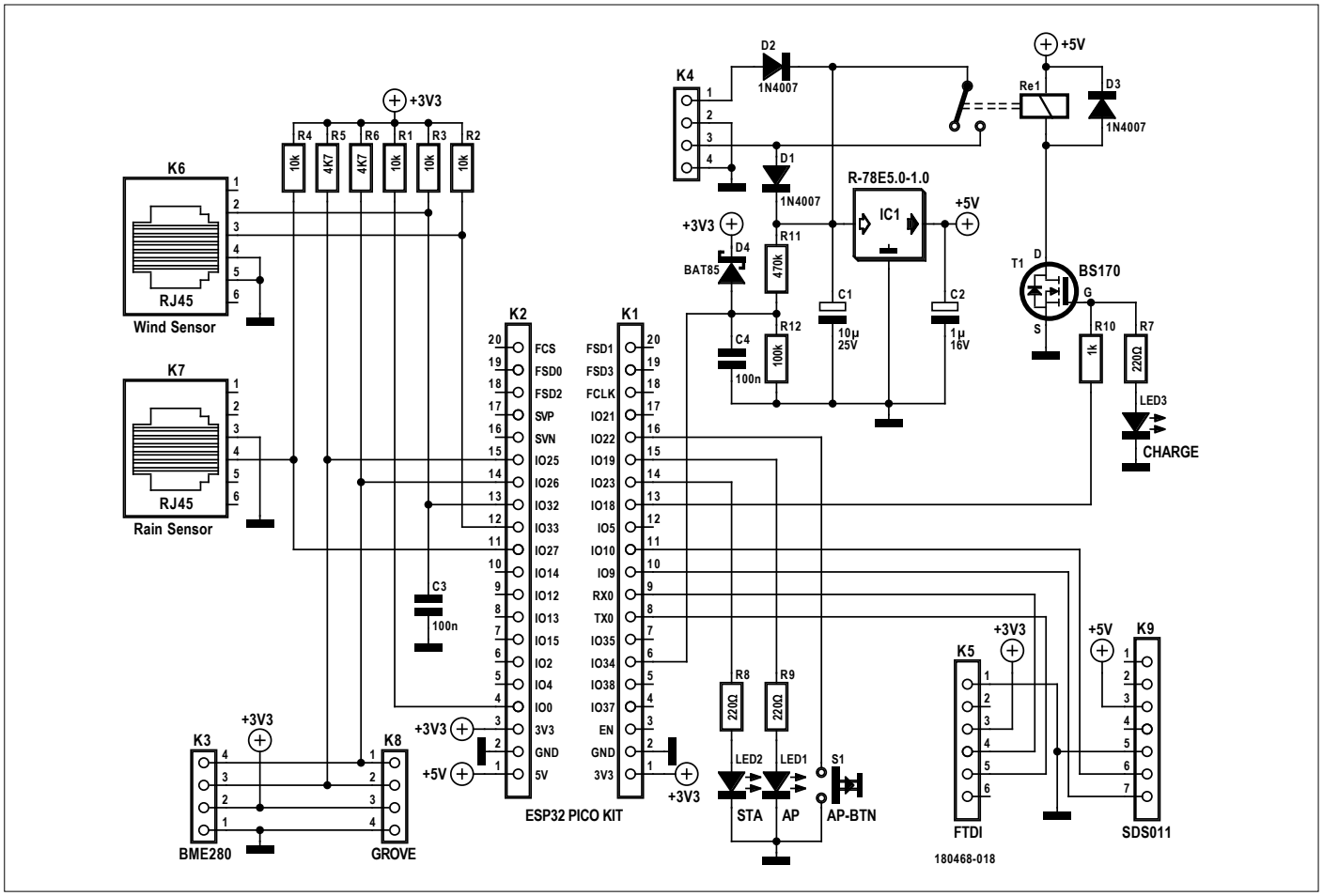


Figure 1. As you can see from the schematic, the ESP32 is the heart of the weather station.

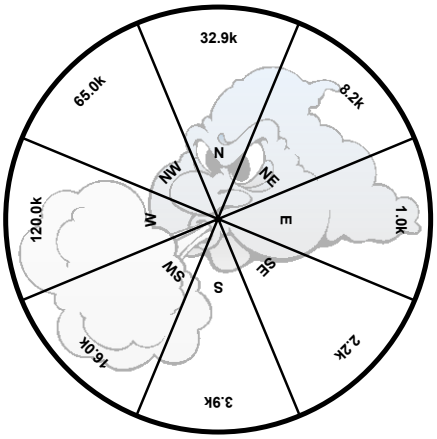


Figure 2. The internal resistance of the wind vane depends on the wind direction.

Sensor data accessible worldwide

LED3, as do resistors R8 and R9 for LED2 and LED1, respectively.

A DC/DC converter (IC1) reduces the power supply voltage to 5 V. This converter is a good deal more efficient than a standard 7805 voltage regulator, so it is a better choice for a battery-powered circuit. On the ESP32 board, the 5 V supply voltage is converted to 3.3 V by a type 1117 voltage regulator IC, and this voltage is available on the board's connector for use elsewhere (that's where the 3V3 in the schematic comes from).

Diode D4 protects the ESP32 against any overvoltage from the battery (via voltage divider R11/R12), while capacitor C4 suppresses noise spikes to foster stable readout of the battery voltage.

Resistors R4 through R6 are pull-up resistors that are necessary for proper operation of the corresponding signal lines.

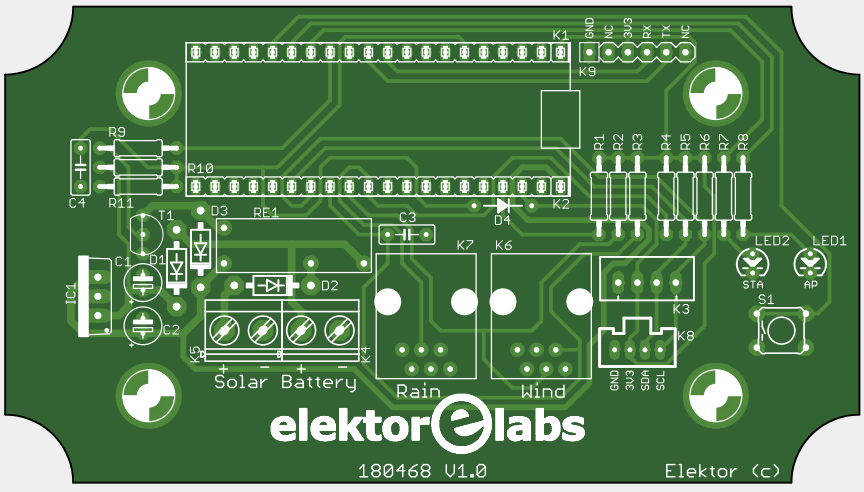
COMPONENT LIST

Resistors
R1-R4 = 10kΩ
R5,R6 = 4.7kΩ
R7-R9 = 220Ω
R10 = 1kΩ
R11 = 470kΩ
R12 = 100kΩ

Capacitors
C1 = 10µF, 50V
C2 = 1µF, 10V
C3,C4 = 100nF

Semiconductors
D1-D3 = 1N4007
D4 = BAT85
IC1 = 5V DC/DC converter, 5W, R-78E5.0-1.0 Recom Power
LED1, LED2 = yellow, 3mm
LED3 = red, 3mm
T1 = BS170

Miscellaneous
K1, K2 = 17-pin (1×17) stacking header, 0.1" pitch
K3 = 4-pin pinheader, 0.1" pitch
K4 = 4-way barrier strip terminal block, 0.1" pitch (Würth 691403900004B)
K5 = 6-pin pinheader, 0.1" pitch
K6, K7 = RJ11 jack, Molex RJ11 6P V
K8 = Grove connector, vertical, TWIG-4P-2.0-2.0
K9 = 7-pin pinheader, 0.1" pitch, JST-XH-07-PIN-VER
RE1 = relay ALDP105
S1 = pushbutton, 6x6mm
BME280 breakout board, I²C version
Wind and precipitation sensor
Enclosure PC 100/60 HT, Fibox
PCB 180468-1
Optional: fine particle sensor, Nova Fitness SDS-011



Rain and wind sensors

The rain sensor consists of a small tray that fills with rainwater. When it is full, it tips and briefly closes a reed switch contact that is monitored by the ESP32. According to the datasheet, each tip of the tray corresponds to 0.33 mm of precipitation.

The wind sensor consists of a wind vane for measuring the wind direction and an anemometer for measuring the wind speed. The internal resistance of the wind vane changes depending on the wind direction. **Figure 2** shows the layout for the various compass directions. We combined a 10 kΩ resistor (R2) with this wind direction dependent resistance to form a voltage divider. The resulting voltage changes when the wind vane sensor resistance changes, and it can be measured by the ESP32.

The anemometer has a reed switch that generates two pulses for each rotation. According to the data sheet, the wind speed in m/s is equal to the pulse frequency (in Hz) multiplied by 0.33. For determination of the rotation frequency, the data sheet says that the time interval between two pulses (in seconds) should be multiplied by 2 (since the sensor generates two pules per rotation), and then the rotation frequency can be calculated as the inverse of this time interval. If you multiply that value by 0.66, you get the wind speed in metres per second.

The wind speed in km/h can be calculated by multiplying the speed in m/s by 3.6. In other words, 1 rps corresponds to a wind speed of 2.4 km/h.

Uploading the software

In order to upload the software to the ESP32, you first have to install the ESP32 Arduino core [7]. Then you have to install the SPIFFS download tool in order to

download the web page to the file system of the ESP32 [8]. Once everything is correctly installed, you can select the ESP32 Pico Kit in the boards menu and then download the web page and the sketch.

Software

As previously mentioned, the weather station is controlled by the ESP32. You can configure it on a web page hosted directly by the ESP32. When the ESP32 starts up, it tries to connect to the configured network. If that fails, the ESP32 launches the web server that hosts the configuration page. The web server can also be launched manually by pressing the button on the board while switching on power, or by pressing the EN button on the ESP32 module. If the user has not done anything on the configuration page for ten minutes, the ESP32 restarts and connects to the configured network. You can suppress this by keeping the button pressed or bypassing the button.

On the configuration page you can view the current measurements as well as the measured battery voltage. You can also configure the network settings and the upload settings. The network settings include the SSID (name) of the network, which can be selected from a list, and the password. The upload settings include the required API keys for Thingspeak and senseBox, as well as the upload interval.

To obtain access to the web page, you have to connect to the ESP32 network provided by the ESP32. Then you can open the web page at the IP address 192.168.4.1. Note that with smartphones (e.g. Android) you must temporarily disable mobile data, because smartphones often switch automatically to mobile data if they do not detect an valid Internet connection via WiFi. In that case the configuration page will not be available.

The web page consists of a single HTML file, which in turn consists of three sections. The <style> element at the top defines the appearance of the page, the <body> element contains the structure and content of the page, and the <script> element at the bottom defines the functionality of the page. For the web page to work properly, javascript must be enabled in the web browser.

The communication between the web page and the ESP32 is done via an 'asynchronous XMLHttpRequest' on the web page. When the page is loaded, an http

Thingspeak

Thingspeak is a free online database from Mathworks to which data can be uploaded. For this, a channel and associated field must be created for each measured parameter (see **Figure 3**). After the channel is created, you receive a write API key (**Figure 4**). This key must be entered on the configuration page of the weather station.

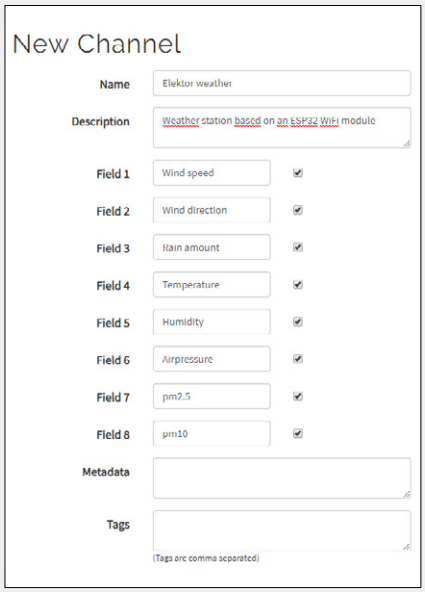


Figure 3. At Thingspeak, you create a new channel for your application and enter the data types in the corresponding boxes.

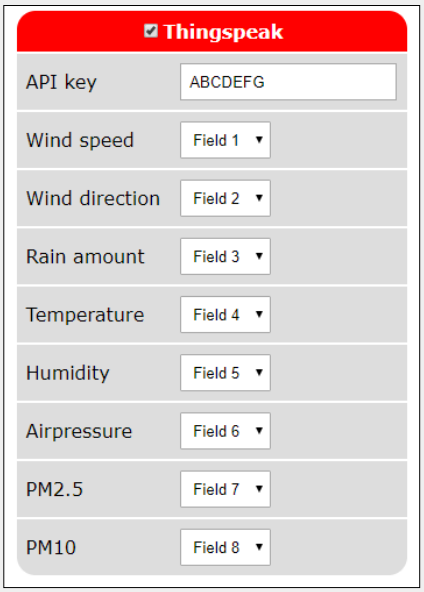


Figure 4. Then you receive an API key to be used in the software of your application.

request is generated and sent to the ESP32. The ESP32 receives this request and sends an answer back to the web page with the requested settings or values. For example, when you press a Submit button on the web page, a similar request is sent that now contains the values you entered.

Practical matters

The only thing left to describe is how to set up the device and connect the sensors. In order to upload the data, it is of course necessary that the ESP32 can access your home network (or some other network). For setting up the sensors, you should also take into account the specific requirements of the sensors concerned. For example, putting the precipitation sensor close to a wall is not a good choice. For this you should be guided by your common sense.

180468-02

senseBox

A senseBox kit makes it easy to set up your own sensor station. The measured parameters can be uploaded to opensensemap.org, where the measurements from all senseBoxes or devices compatible with senseBox can be viewed. To register your own weather station, you first create an account at opensensemap.org and then you can add a senseBox. You can select your specific senseBox type on the account creation screen, or you can configure your measurement station yourself by entering the measured parameters, the measurement units and the sensor types (see **Figure 5**). You have to create an account for the EPS32 weather station manually. After you register the weather station, you receive an ID for the station and a sensor ID for each sensor. Then you enter these IDs on the configuration page of the EPS32 weather station. After that it can upload the measured data.

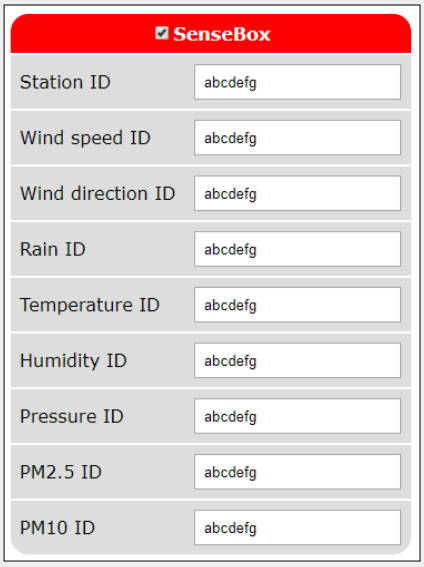
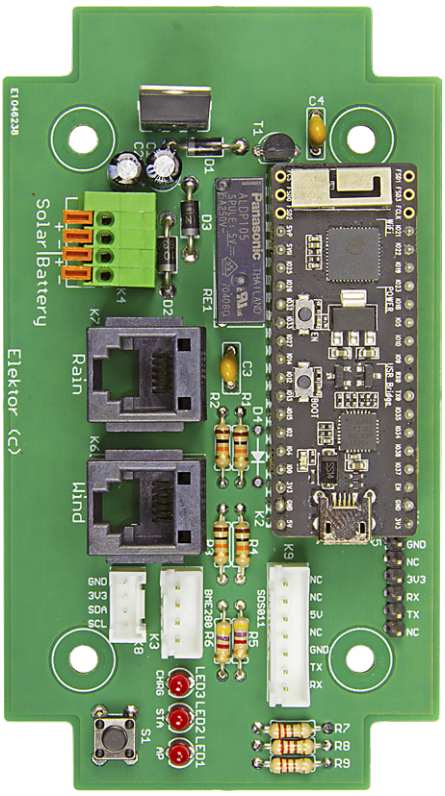


Figure 5. At opensensemap.org you can create your own sensors for the ESP32 weather station.



Web Links

[1] Inovafitness: www.inovafitness.com/en/a/chanpinzhongxin/95.html

[2] BME280 (Mouser): www.elektor.com/bme280-mouser-intel-i2c-version-160109-91

[3] Thingspeak: https://thingspeak.com

[4] Sensebox: https://sensebox.de/en

[5] Seeedstudio products at Elektor: www.elektor.com/seeedstudio

[6] Seeedstudio Grove: www.seeedstudio.com/s/grove.html

[7] Github for Arduino-Esp32: https://github.com/espressif/arduino-esp32

[8] Github for Arduino Esp32fs plugin: https://github.com/me-no-dev/arduino-esp32fs-plugin