École Polytechnique Fédérale de Lausanne

All-Sky Camera & Weather Station

USER MANUAL

VERSION 1.0

Olivier Völlmin

Laboratory of Astrophysics

Project Supervisors:
Professor Jean-Paul Kneib
Romain Lucchesi

Lausanne, Fri 9th Jul, 2021

Contents

1 Introduction

This document provides necessary information for anyone trying to work with or improve the weather station and the All-Sky camera. It is a complementary document to the Github repository providing the code referenced below:

`https://github.com/OlivierVollmin/TELESTO-weather-allsky`

It is assumed that the user has a basic knowledge on the working of Linux system and Python.

2 Users & Credentials

The raspberry pi runs on a headless[1] Raspian 10 (Buster) operating system.

- The default user is "**pi**"
- The password is "**TELESTO**"
- The fixed IP address on the local network is **10.194.64.26**
- The hostanme is **raspberrytelesto**

One can simply access the Raspberry Pi using SSH by typing the following command in the terminal or command prompt of his computer:

```
$ ssh pi@raspberrytelesto
```

If it is the first time logging in, the user will be asked to accept the connection. The password for the remote connection is "**TELESTO**".

To navigate directories the user can use **cd** and **ls** commands, the main files of interest are located in:

```
$ cd /var/www/html/
$ cd /home/pi/
```

The `html` directory contains all the files for the server whereas the `pi` directory contains all the scripts, shell scripts and executable to run the weather station and All-Sky camera.

[1]Headless means that the graphical interface has been deactivated, the user interacts the Raspberry Pi via SSH over the terminal. To learn more about headless Raspberries have a look at `https://www.raspberrypi.org/documentation/configuration/wireless/headless.md`

3 MariaDB

A MariaDB database has been setup to store the weather measures and CPU temperatures efficiently. The installation and initialization of the database has been partly inspired from the following instructions on this page:

`https://projects.raspberrypi.org/en/projects/build-your-own-weather-station`

The two MariaDB databases are "**weatherDB**" and "**CPU_tempDB**", they both share the same password "**TELESTO**". They also both have a column "**ID**" that is automatically incremented, a column called "**CREATED**" with the timestamp of the values entry and their entries older than 6 months automatically dropped by scheduled events.

```
MariaDB [weatherDB]> show columns from WEATHER_MEASUREMENT;
+-----------------------+--------------+------+-----+---------------------+----------------+
| Field                 | Type         | Null | Key | Default             | Extra          |
+-----------------------+--------------+------+-----+---------------------+----------------+
| ID                    | bigint(20)   | NO   | PRI | NULL                | auto_increment |
| AMBIENT_TEMPERATURE   | decimal(6,2) | NO   |     | NULL                |                |
| AIR_PRESSURE          | decimal(6,2) | NO   |     | NULL                |                |
| HUMIDITY              | decimal(6,2) | NO   |     | NULL                |                |
| WIND_DIRECTION        | decimal(6,2) | NO   |     | NULL                |                |
| CREATED               | timestamp    | NO   |     | current_timestamp() |                |
| WIND_SPEED            | decimal(6,2) | NO   |     | NULL                |                |
| WIND_GUST_SPEED       | decimal(6,2) | NO   |     | NULL                |                |
| RAINFALL              | decimal(6,2) | NO   |     | NULL                |                |
| DEW_POINT_TEMPERATURE | decimal(6,2) | NO   |     | NULL                |                |
| IS_RAINING            | tinyint(1)   | NO   |     | NULL                |                |
+-----------------------+--------------+------+-----+---------------------+----------------+
11 rows in set (0.004 sec)
```

Figure 1: Columns of `WEATHER_MEASUREMENT` from `weatherDB`

```
MariaDB [cpu_tempDB]> show columns from CPU_TEMP_TABLE;
+-------------+--------------+------+-----+---------------------+----------------+
| Field       | Type         | Null | Key | Default             | Extra          |
+-------------+--------------+------+-----+---------------------+----------------+
| ID          | bigint(20)   | NO   | PRI | NULL                | auto_increment |
| CPU_TEMP    | decimal(6,2) | NO   |     | NULL                |                |
| CREATED     | timestamp    | NO   |     | current_timestamp() |                |
| CAMERA_TEMP | decimal(6,2) | NO   |     | 0.00                |                |
+-------------+--------------+------+-----+---------------------+----------------+
4 rows in set (0.003 sec)
```

Figure 2: Columns of `CPU_TEMP_TABLE` from `CPU_tempDB`

To access the **weatherDB** database from the terminal on the Raspberry Pi, one can enter the following two commands:

```
$ sudo mysql
$ use weatherDB
```

Be extra careful when working in the database as the user "**pi**" has all permissions granted, included the permission to drop or delete multiples entries from the table at once irreversibly.

To access the database from a python script one should use the following library:

```python
import mysql.connector as mariadb
```

Example on how to read or write from/to the database can be found in `Measure.py` and `plot.py`, additionnal help can be found in the MySQL/MariaDB documentation:

`https://dev.mysql.com/doc/connector-python/en/connector-python-reference.html`

4 Weather Station

4.1 Main Script

The code running the weather station works as following (`Measure.py`):

```
while forever do
    Reset wind speed, direction array;
    Reset rain bucket tipping counter;
    for time interval of 5 minutes do
        Measure wind speed, wind direction;
        Count rain bucket tipping;
    Compute average wind speed, wind direction;
    Find max wind speed;
    Convert the number of buckets tipping into rainfall;
    Read temperature, humidity, relative pressure;
    Compute dew point temperature;
    Read capacitive rain sensor;
    if temperature below freezing OR it is raining then
        Set capacitive rain sensor heaters on;
    Add new row to WEATHER_MEASUREMENT (database) with current timestamp;
```

There is a main infinite loop and two encapsulated inner loops. The first inner loop has a duration of 5 minutes and is used to average the wind direction and speed, while the second inner loop is hidden in the "wind measure". This loop is created by the `time.sleep(timeout)` command, which freezes the script for the duration of timeout. However, interrupts are not affected by the pause and each time the anemometer does a full rotation, an signal change is sensed by the GPIO pin and interpreted as an interrupt by the Raspberry Pi. The interrupt handler is called for this specific interrupt and increments a counter variable. At the end of the pause, a wind speed is inferred from the number of rotation on the counter variable.

4.2 Interface Settings & User Groups

For several sensors, (BME280, reed relays, etc.) the GPIO, I2C, SPI peripherals of the Raspberry Pi are used. To be able to read them properly, one must enable them individually in the `Interface Options` menu in the Raspberry Pi configuration using the following command:

```
$ sudo raspi-config
```

Furthermore, the access to the peripheral is by default restricted to small number of user on the Raspberry Pi for safety reasons. If a user other than `pi`, for instance `www-data` wants to run a python script to read the current status of a GPIO pin, the access will be denied. Therefore specific access must be granted by adding the `www-data` user to the gpio group. Granting all access to the `www-data` **must be avoided at all costs**, as it introduces critical safety issues. Usually `www-data` is used to run web servers, malicious user could inject potentially dangerous commands and have root accesses.

4.3 `gpiozero` particularity

Unfortunately the `gpiozero` module does not allows for simultaneous access to GPIO pins by Python scripts. Continuous reading of the capacitive rain sensor in `Measure.py` and periodic reading requested by the web page are thus not possible, as both scripts required access to the same GPIO. A direct consequence is the creation of the `RainPHP.py` script, which is called in `Measure.py` as well as by the web page each time the user refreshes the page. This was a quick workaround to only have one script attached to one pin.

There exists more complex libraries that allow shared reading of GPIO pins, but their compatibility with other libraries need to be verified and tested.

5 All-Sky

The settings of the All-Sky camera can be modified either directly on the web page or via SSH in the config files.

For all the settings explications please consult the GitHub of Thomas Jacquin and read his README.md file.

`https://github.com/thomasjacquin/allsky`

6 Cron Tab

Cron tab is used on the Raspberry Pi to manage all the different scripts that need to be run.

```
@reboot /usr/bin/python3 /home/pi/weather/Measure.py >> /home/pi/weather/Measure.log 2>&1
*/5 * * * * /usr/bin/python3 /home/pi/weather/plot.py >> /home/pi/weather/plot.log 2>&1
*/5 * * * * cp -fr /home/pi/weather/weather_plot.png /var/www/html/weather_plot.png
*/5 * * * * cp -fr /home/pi/weather/windrose.png /var/www/html/windrose.png
*/5 * * * * cp -fr /home/pi/weather/rainfall.png /var/www/html/rainfall.png
12 10 * * * /sbin/shutdown -r now
*/5 * * * * /usr/bin/python3 /home/pi/weather/CPU_temp.py >> /home/pi/weather/CPU_temp.log 2>&1
*/5 * * * * cp -fr /home/pi/weather/cpu_temp.png /var/www/html/cpu_temp.png
```

Figure 3: List of cronjobs

From Figure 3 we see that `Measure.py` is started right after rebooting and logs its errors to `Measure.log` in the same directory. There are two other scripts that are called every 5 minutes, to plot the weather data and to read the temperatures and generate the subsequent plot. Last but not least, the Cron utility is used to force copy (overwriting existing images) in the `/var/www/html` directory with the latest plots. Finally, a daily reboot has been scheduled at 10:12 am to restart crashed scripts or catch any other unseen error.

7 Lighttpd Server

To be able to query data and display it properly on the webpage using PHP code, a module is required, it can be downloaded using :

```
$ sudo apt-get install php-cgi php-fpm php-mysql
```

The web server is based on Thomas Jacquin's project[2] and has only been slightly modified, part of the code from `index.php` has been commented out to allow access without password restriction to the web page. Few lines have been added to link and display three additional pages. The php code of those pages has been included in the `/var/www/html/includes` directory.

1. `liveweather.php` this code is used to display the 3 plots, the current reading of the rain sensor and a small table of the latest database entries.
2. `api.ph` this code queries the weather database and returns a list of the last 1440 database entries in a JSON syntax.
3. `api2.ph` this code queries the CPU temperature database and returns a list of the last 14 days database entries in a JSON syntax.

The images (cpu_temp.png, rainfall.png, weather_plot and windrose.png) displayed are stored in `/var/www/html` and are overwritten by the Cron utility approximately every 5 minutes.

The images (rain.png and sun.png) are used to display either a green dot or a red dot to inform the user that the capacitive rain sensor is On or Off. The path to the image in the php file is conditional, it depends on the return value of `RainPHP.py`, hence changing accordingly to the current reading of the sensor.

8 Electrical Cabinet Keys

The keys to open the box are stored in the electrical workshop with other cabinet keys. The one for the All-Sky box is the right one made out of plastic on the Figure 4 below.

Figure 4: Key Chain with All-Sky cabinet key (right plastic "FIBOX" key)

[2] All-Sky webportal : `https://github.com/thomasjacquin/allsky-portal`

9 Wiring

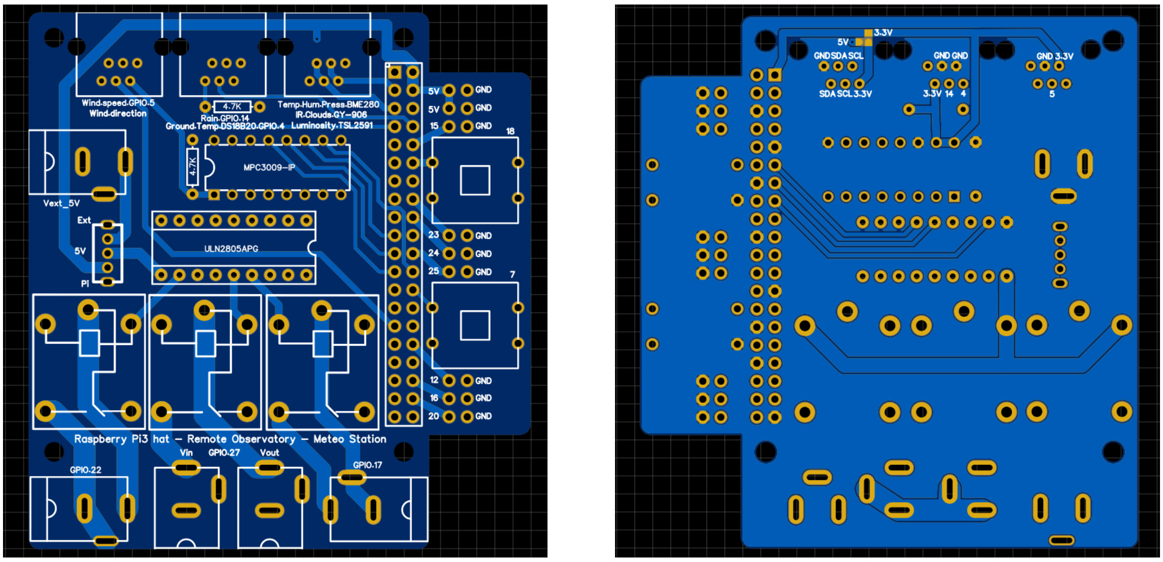
The wiring is relatively simple thanks to a custom PCB designed by Romain Lucchesi (Fig. 6). There are three registered jack connectors on the board. The cables used for this project all have a six-position connectors jacks, but the cables do not always carry 6 conductors. Not all position in the connector jacks are connected on the PCB neither. As a result, some cable can carry more than one sensor/device while others are limited.

There is a small solid state relay (SSR) used to commute the heater in the capacitive rain sensor, hence the additional wire from the Raspberry Pi.

Device A	PIN A	PIN B	Cable	Device B
Raspberry Pi 4 via Custom board	3.3V	VIN	RJ14 cable A	BME280 Adafruit Sensor over I2C
	GND	GND		
	GPIO3 (SCL)	SCK		
	GPIO2 (SDA)	SDI		
	GPIO4	RJ11 - 6	RJ14 cable B	Capacitve Rain Sensor
	GND	RJ11 - 5		
	GPIO16	HZ1	two-wire	SSR for heater commuting
	GND	HZ2		
	USB-C Fem.	USB-C Male		Raspberry Pi 4 Power Supply
	GND	RJ11 - 3	RJ14 cable C	Anemometer
	GPIO5	RJ11 - 4		
	3.3V	RJ11 - 1		Wind Vane
	-> see MCP3008	RJ11 - 2		
	GND	RJ11 - 3	RJ14 cable B	Bucket Rain Sensor
	GPIO14	RJ11 - 4		
	-> see MCP3008	CH0	PCB socket	MCP3008 ADC
	GND	Vdd		
	3.3V	Vref		
	3.3V	AGND		
	GPIO11 (SCLK)	CLK		
	GPIO9 (MISO)	Dout		
	GPIO10 (MOSI)	Din		
	GPIO8 (CE0)	CS/SHDN		
	GND	DGND		
	USB3	USB3	USB3	All Sky Camera
MCP3008 ADC	CH0	RJ11 - 2	RJ14 cable C	Wind Vane
Rain Detector REGME config				
Switch mode signal generator (dry)	S3	Contact		
Common contact to S3 and S1	S2			
Switch mode signal generator (wet)	S1			
Signal generator ground	GND			
Signal generator output	BUZ			
Switch mode relay (dry)	T3			
Common contact to T3 and T1	T2	Contact		
Switch mode relay (wet)	T1			
Heating	HZ1	SSR		Solide State Relay (SSR)
Heating	HZ2	SSR		
Capacitif Rain Sensor	AC/GND	GND		12V, 600mA Power Supply
	AC/DC+	12V		
	REL CO	GND		Raspberry Pi 4 via Custom board
	REL NC			
	REL NO	GPIO4		

Figure 5: Wiring table for the Raspberry Pi 4B, custom PCB and the sensors. There are 3 RJ cables denoted A,B&C, some carry multiple sensors. Details about the inner connections (S1-3, GND, BUZ, T1-3, HZ1 & HZ2) of the B+B Thermo-Technik rain sensor.

9.1 Custom PCB



(a) PCB recto

(b) PCB verso

Figure 6: Custom PCB designed by Romain Lucchesi

9.2 Wiring Inside the Box

The wiring inside the box is slightly messy, as 3 different voltages are required (5, 12 and 230V). The 230V lines are clearly separated from the other wires and tightly attached to the plate (Fig.8).

The layout of the wiring is shown on the sketch (Fig.7) below.

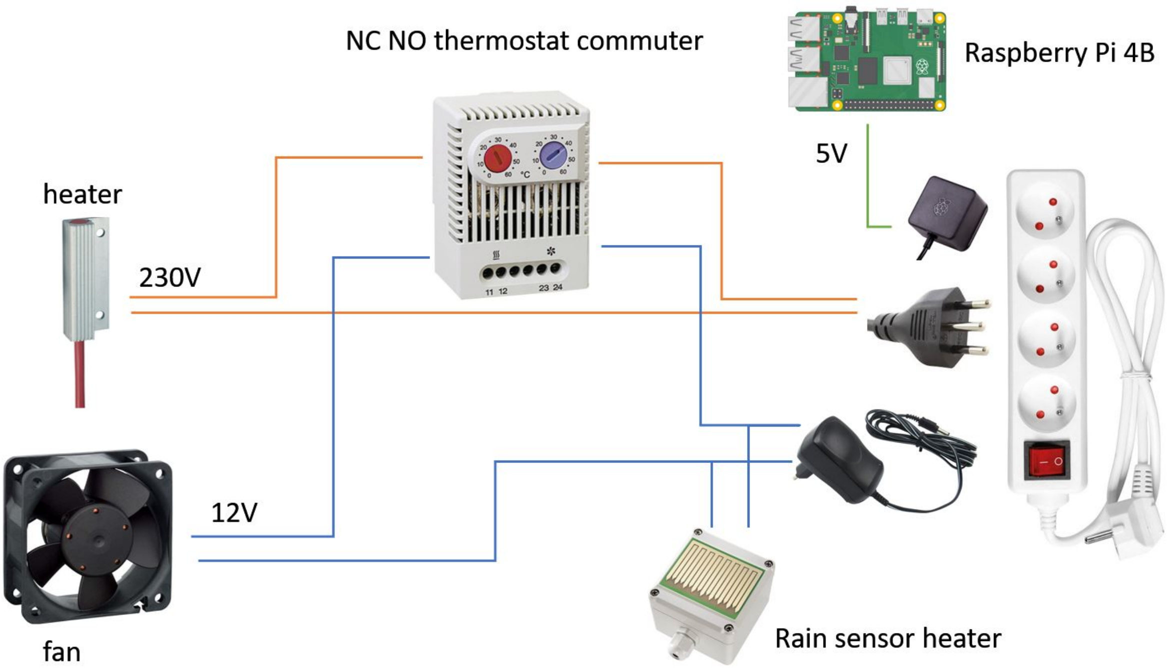


Figure 7: Power distribution sketch

Figure 8: Interior of the Cabinet with all wiring. 230V wiring tightly fastened to the inner plate at the top. DIN rail supporting the NC NO commuting thermostat, WAGO multi clip and the Raspberry Pi 4B with custom shield.

10 Component Documentation Links

10.1 Raspberry Pi 4B

https://www.raspberrypi.org/documentation/

10.2 BME280

https://wiki.seeedstudio.com/Grove-Barometer_Sensor-BME280/

10.3 Weather Meter Kit

https://www.sparkfun.com/products/15901

10.4 Capacitif Rain Sensor

https://www.conrad.ch/fr/p/b-b-thermo-technik-detecteur-de-pluie-1-pc-s-regme-12-v-l-x-l-x-h-85-x-85-x-60-mm-187621.html

10.5 MPC3008

https://www.adafruit.com/product/856

10.6 ASI 178 MM Mono

https://astronomy-imaging-camera.com/product/asi178mm-mono

10.7 Heater

https://www.stego-group.com/products/heating/heaters/rc-016-8-w-to-13-w/

10.8 Fan

https://www.ebmpapst.com/de/de/products/compact-fans/axial-compact-fans/p/612NH.html

10.9 Thermostat

https://www.stego-group.com/index.php?id=1294&L=538