Friday, June 19, 2020 tom.muller1@gmail.com

Building an automated weather station and pressure sensor

Objectives

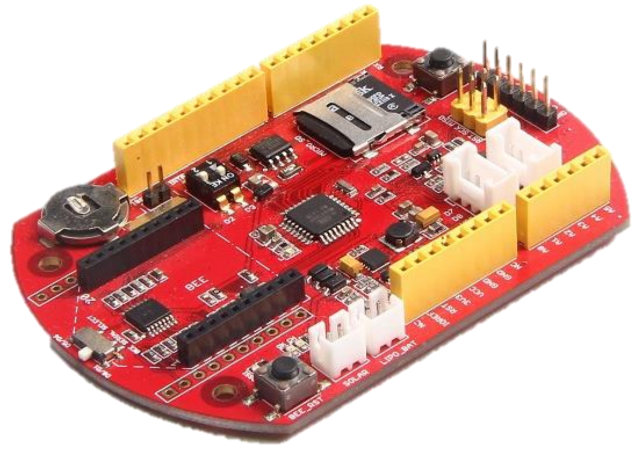
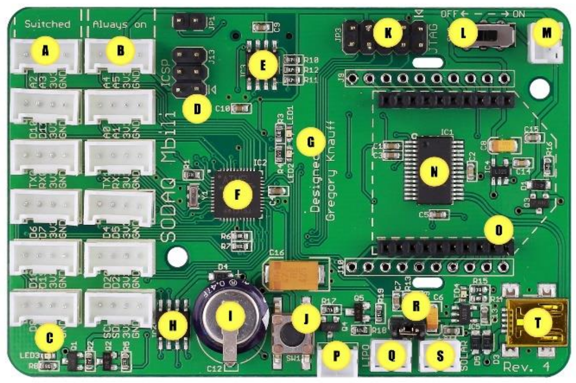
1. Find a DIY datalogger with low power consumption that can run on a small LiPo battery (3.7V) with SD card slot, internal clock (RTC) and solar panel (and optionally a bee socket);
2. Build a precise waterproof pressure sensor ;
3. Build the AWS ;
4. Code the interface to record data correctly ;
5. Adding other sensors ;
6. Test and calibrate the sensor ;
7. Add a LoRa network to send data online (bonus)

1. DIY datalogger

The idea is to base our station on the Arduino interface, which promotes DIY and is supported by a large community, allowing to find many tutorials and help on forum across the internet.

However, the standard Arduino board is composed of a very simple design, only composed of a microcontroller and digital and analog input/output pins. This is a good way to start learning about how to control small device or read a sensor but doesn't fulfil our needs of objective 1.

Many companies have however developed their own board to provide low-power autonomous devices. After many trials I am currently working with 2 boards:

Seeeduino Stalker v3.1 : A cheap board that contains all necessary features but has a relatively low internal memory (Atmega328P) and sometimes some small bug, but overall very satisfying. Imported from the US. You will need a USB-UART converter to communicate with the board. 	SODAQ Mbili : A quite similar board with a larger memory (ATmega 1284P), a bit more flexible and plug-and-play grove sockets. A bit more expensive and imported from the Netherland (Europe). ***Caution : If you use mbili, jumper SJ8 should be soldered to use external RTC interrupt*** 

To be noted, a more expensive board is also very popular in the US and seems a robust alternative, with a strong community around it : the Mayfly Data Logger Hardware from envirodiy.org. This board is not chipped to Europe and I did not test it.

2. Waterproof pressure sensor

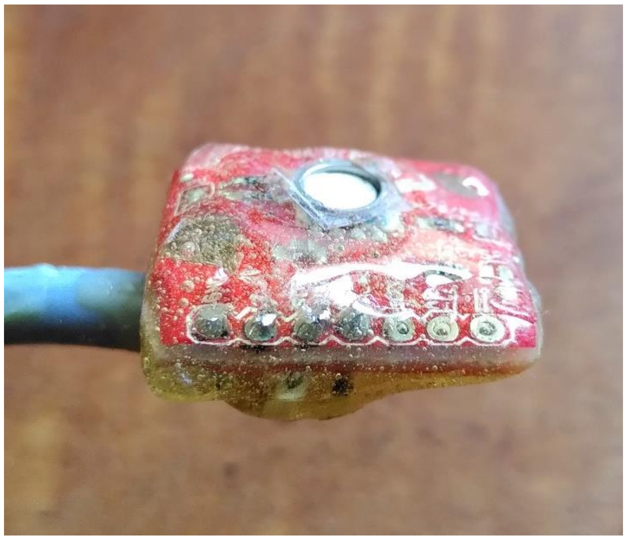
This sensor was inspired from this post of enviroDIY : https://www.envirodiy.org/construction-of-water-level-monitoring-sensor-station/

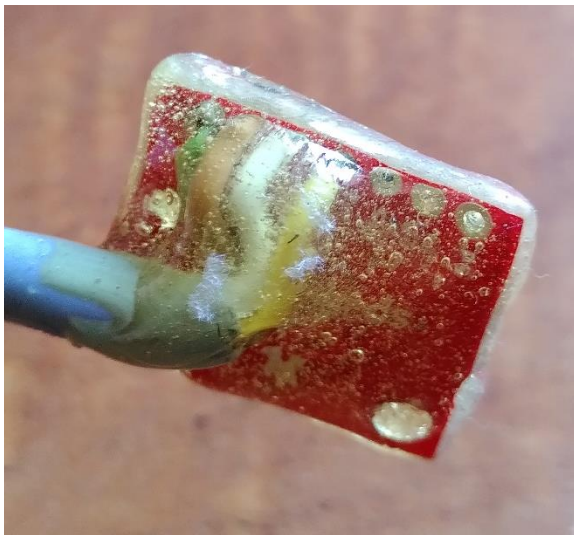
The idea is to use a pressure valve (MS5803-14BA), solder a cable to the pins and envelop it with epoxy glue. All info can be found on the link above.

My current design is composed of the following components :

- 1x MS5803-14BA : https://www.mouser.ch/ProductDetail/SparkFun/SEN-12909?qs=WyAARYrbSnbe%2FsfkO8iqRw%3D%3D
- 2.5 meters Mutlicore cable (4x0.5mm2) : https://www.distrelec.ch/en/multicore-cable-yy-pvc-4x-5mm-grey-50m-lapp-00100024-50/p/30049250?no-cache=true&track=true
- Epoxy glue : https://www.reichelt.com/de/en/wiko-epoxy-adhesive-5-min-wiko-epo5-s25-p98449.html?r=1
- Acrylic lacquer : https://www.reichelt.com/de/en/acrylic-protective-lacquer-400-ml-rnd-605-00135-p211619.html?r=1

You basically solder the cable, spray epoxy (but care to protect the white pressure valve), apply epoxy arond the circuit board. My design is very rough and can clearly be improved though.

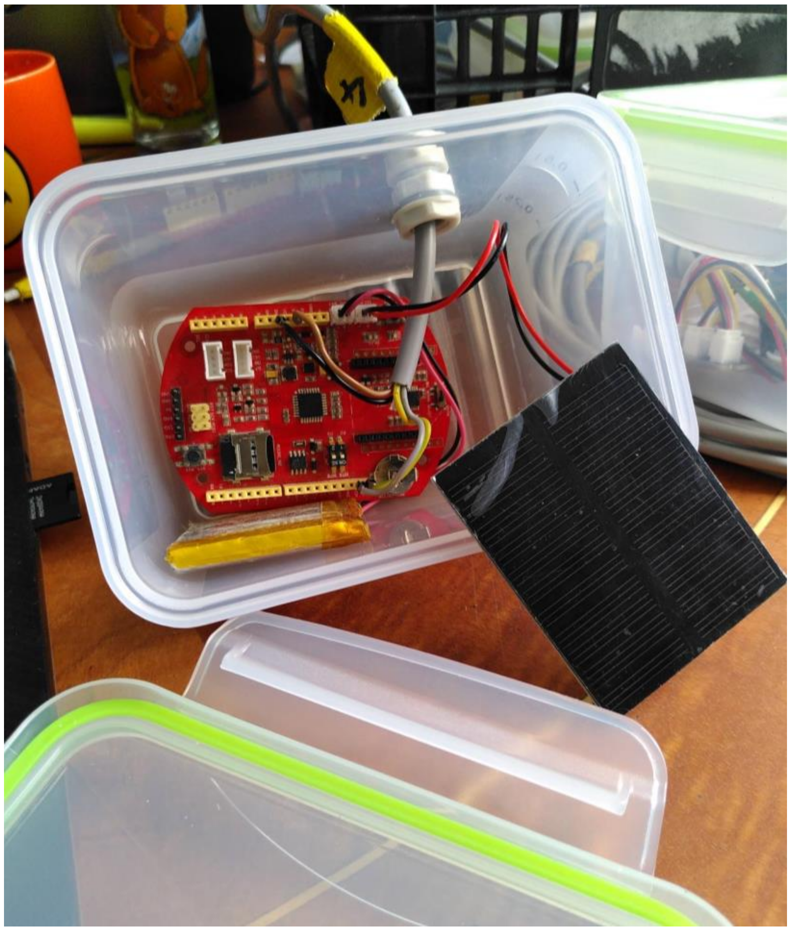




3. Building the automated weather station

We need to pack everything in a waterproof box and add sd card, solar panel and batteries. I use the following parts :

- cheap lunch box (5 swiss francs, from Migros shop)
- Seeeduino 0.5W Solar Panel 55x70mm (with 2mm JST connecter)
- 1200 mAH LiPo battery (2mm JST connecter)
- 4 GB sd card
- Cable gland



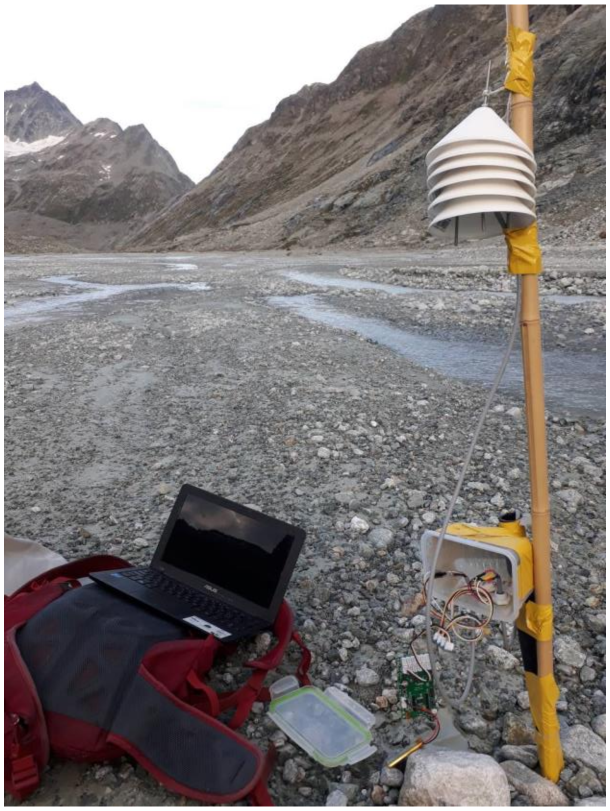
4. Coding

The codes are available here : https://github.com/tomuelle/DIYweatherstation/

I will not go into details about the codes here. They can probably be a bit improved but they worked well for me. They are programmed to read the every X seconds (minimum 60sec), save all data on the SD card in a scv files and then go to deep sleep mode until they wake up (using an external RTC interrupt).

These codes can be easily modified to add other sensors. Usually the manufacturer provides a basic Arduino code to read the data from their sensors so that only a few lines need to be added to the program.

5. Adding other sensors



In order to measure water depth for instance, there is a need to add a second pressure sensor in the air and then calculate the difference between air pressure and water+air pressure. Thus, we need a sensor in the air. This can be achieved using another MS8301-14BA sensor which is left in the air.

If you add two sensors on the same board, you need to change the addressing by solder the x77 jumper on the circuit board and change the addressing in the program (*MS5803 sensor(ADDRESS_LOW);* to *MS5803 sensor(ADDRESS_HIGH)*).

However, other cheaper sensors are available such as Seeeduino BME280 which measures temperature, humidity and air pressure. The sensor should be protected by a radiation/rain shield.

I faced some issue communicating with both BME280 and MS8301 with the seeeduino board so that this set-up was only used with the mbili board (see the combined setting working on the field).

6. Testing and calibrating the sensor

I haven't completely finished testing and calibrating all sensors yet. It would be useful to observe any offsets between sensors as well as checking if temperature changes modifies the measurements. Also long-term evolution should be monitored (especially with the risk of water slowly infiltrating under the epoxy).

7. Sending data online (LoRa network)

This is a long chapter and I will only give an overview.

Both boards have a bee sockets, allowing to plug different bee antennas. In the past, I used the GPRSbee module from SODAQ (only working with the mbili) to connect the sensor. However with the end of the 2G network a better solution needed to be found.

I am now using a LoRa antenna. This network only works if a master antenna is available in the vicinity (range of 5 to 10 km). Fortunately, many free antennas from thethingsnetwork are available in my neighborhood, so that I only needed to create a free account and register my devices. In remote areas, a stand-alone master LoRa gateway should be installed.

I also uploaded a code where my mbili board sends the data to thethingsnetwork where they are recorded. Then these data can be further pushed to another application such as thingspeak.com or to a web server.

I am using a raspberry pi in my home as data server which collects and saves the data and eventually publish them on my web server (but this is a whole different tutorial).

An example of the final result is found here : https://fermedebassenges.ch/meteo/