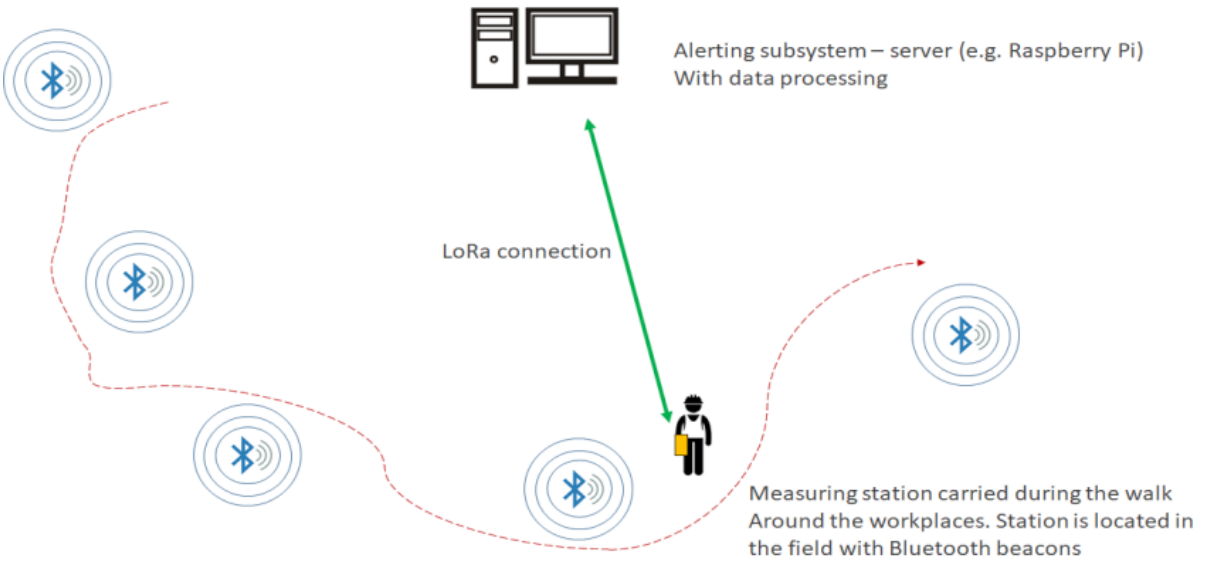
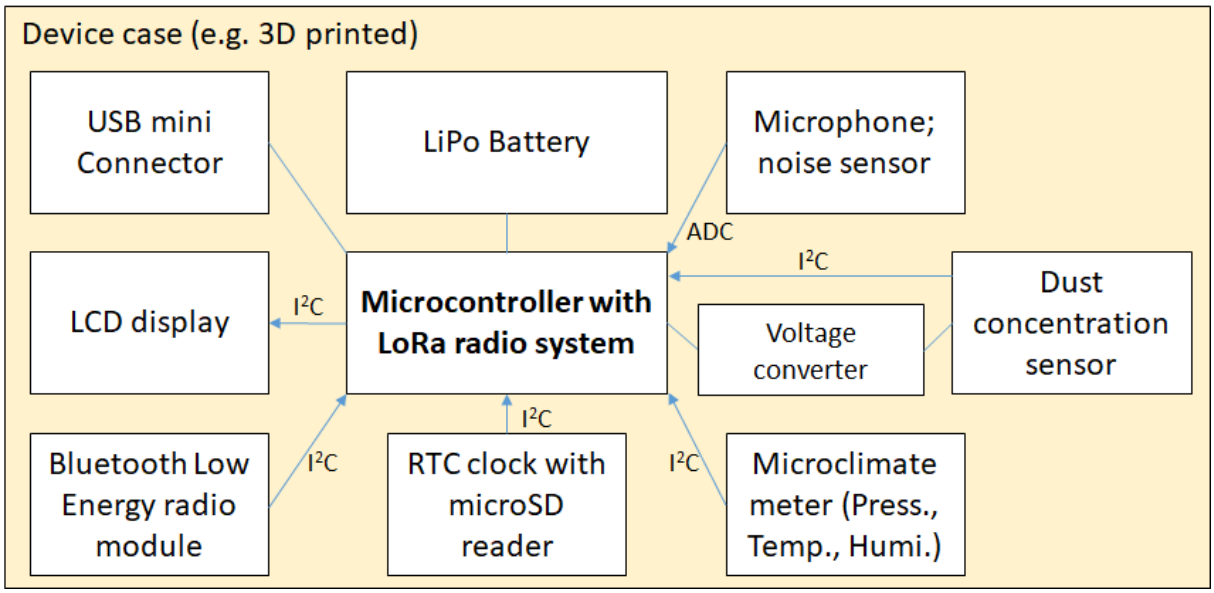
**Design of a near real-time dust and acoustic hazard monitoring system based on wireless data transmission and a portable measurement station**

The proposed solution consists of two subsystems: observation (portable, measuring station) and alerting. The structure of the system is presented in a simplified way:




**Observation subsystem**

Main idea for portable, measuring station:



**Alerting subsystem**

The function of receiving and archiving information could be performed by the acquisition unit (server, e.g. based on Raspberry Pi), and the information and warning function will be performed by a computer program (e.g. written in Python). The server should wirelessly receive data from the observation system, and then send the measurement results to a dedicated program. The software could enable the visualization of data superimposed on the network map (hazards mapping), analysis of the observation results and comparing them with alarm levels (manually set).

Details:

# 1. Observation subsystem

The observation subsystem consists of two basic elements: a portable measurement station with environmental sensors and Bluetooth LE beacons.

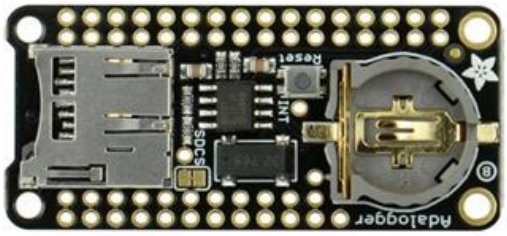
For this project we have chosen:

**Microcontroller with the LoRa radio module**
Adafruit Feather M0  with RFM95 LoRa Radio:

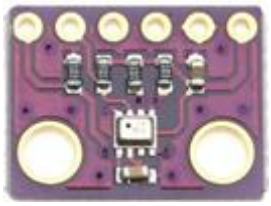

**Station power**
Akyga battery LiPo lp903759 (2200 mAh, 3,7 V):

**Dust sensor**
Sensirion SPS30:

**Real time clock and woth SD slot**
Adalogger FeatherWing with the RTC clock and MicroSD slot.




**Microclimate sensor**
BME280 for pressure, temperature, humidity sensor:

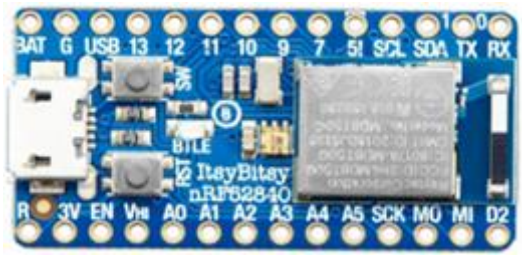


**Local data reading**
LCD display DFRobot Gravity:

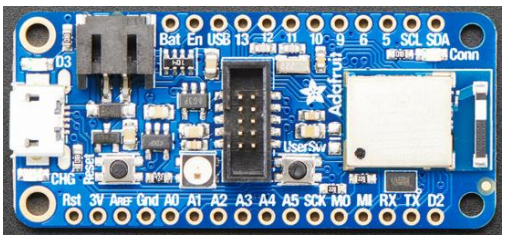

**Converter**
Step-up converter 5V, 1.4A


**Noise sensor**
For this project we have chosen a custom made, original microphone (analog noise meter with constant gain), but probably any cheap mike could work.

**Location using the Bluetooth Low Energy module**
Adafruit ItsyBitsy nRF52840 Express with Bluetooth LE connected to the portable station:




**Bluetooth LE beacons:**
ItsyBitsy nRF52840 Express that could work on battery or connected to the main power.




# 2. Alerting subsystem

The function of receiving and archiving data could be performed by an acquisition unit. The results can be collected by a Raspberry Pi4 based server or a larger PC station based server. Second LoRa radio module should be connected to the server for receiving the data from the portable station.

The project was not carried out at this stage.

Raspberry Pi 4 model B with 4GB RAM, Dual Band WiFi

---

**Diagram of connecting electronic components:**

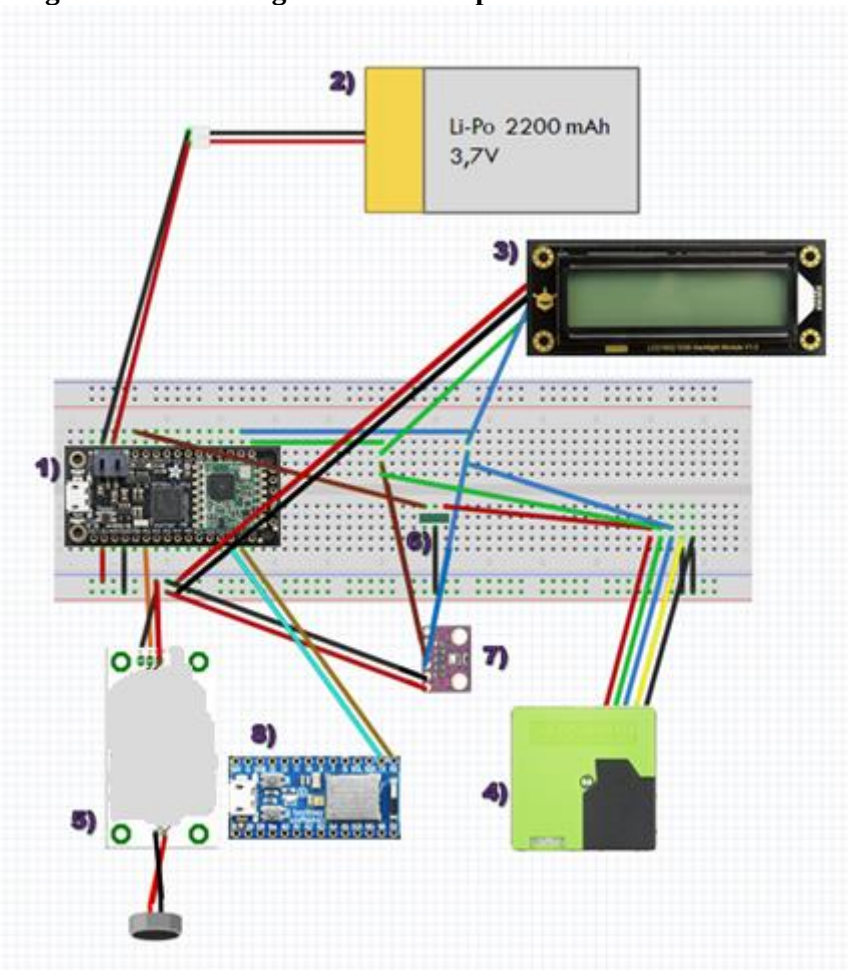



Diagram of connecting electronic components in a portable measuring station: 1) microcontroller with the LoRa radio, underneath there is a plate with RTC clock and SD slot, 2) LiPo battery, 3) LCD display, 4) dust concentration meter, 5) noise meter, 6) voltage converter, 7) microclimate meter, 8) Bluetooth Le radio module.

Well, that diagram is a little bit messy, but you can get the idea.

# 3. Arduino codes

There are 4 codes in this repository:

- Portable_station (LoRa, addressing for LoRa, BLE readings, BME, RTC, SD, Battery check, LCD, SPS30, mike),
- BLE_portable,
- BLE_beacon,
- LoRa_for_server.

We get most of the codes from Adafruit examples, some tutorials etc. and stitch them together. BLE beacon is just an advertising code, BLE connected to portable station has its own code which is get through second serial. Both LoRa portable and LoRa connected to the server has their unique addressing.

The code isn't clean and perfect, but it's working so far. If you want to make the system better, you should add the button to switch on the LCD (it should not work constantly, because it's using too much power). SD card should write the data once in a while, same for SPS30 – it should switch on for 1 measurement per minute every 15 or 30 seconds, but not constantly – it is also using too much power. We also don't trust the battery check, it should be carefully tested.

But, the station is reading the data, showing them on the LCD, writing on the SD card and sending them to the LoRa receiver (server) which is displaying all the information on the serial.