



INTEL-IRRIS

Intelligent Irrigation System for Low-cost Autonomous Water Control
in Small-scale Agriculture



INTELLIGENT IRRIGATION SYSTEM FOR LOW-COST AUTONOMOUS WATER CONTROL IN SMALL-SCALE AGRICULTURE



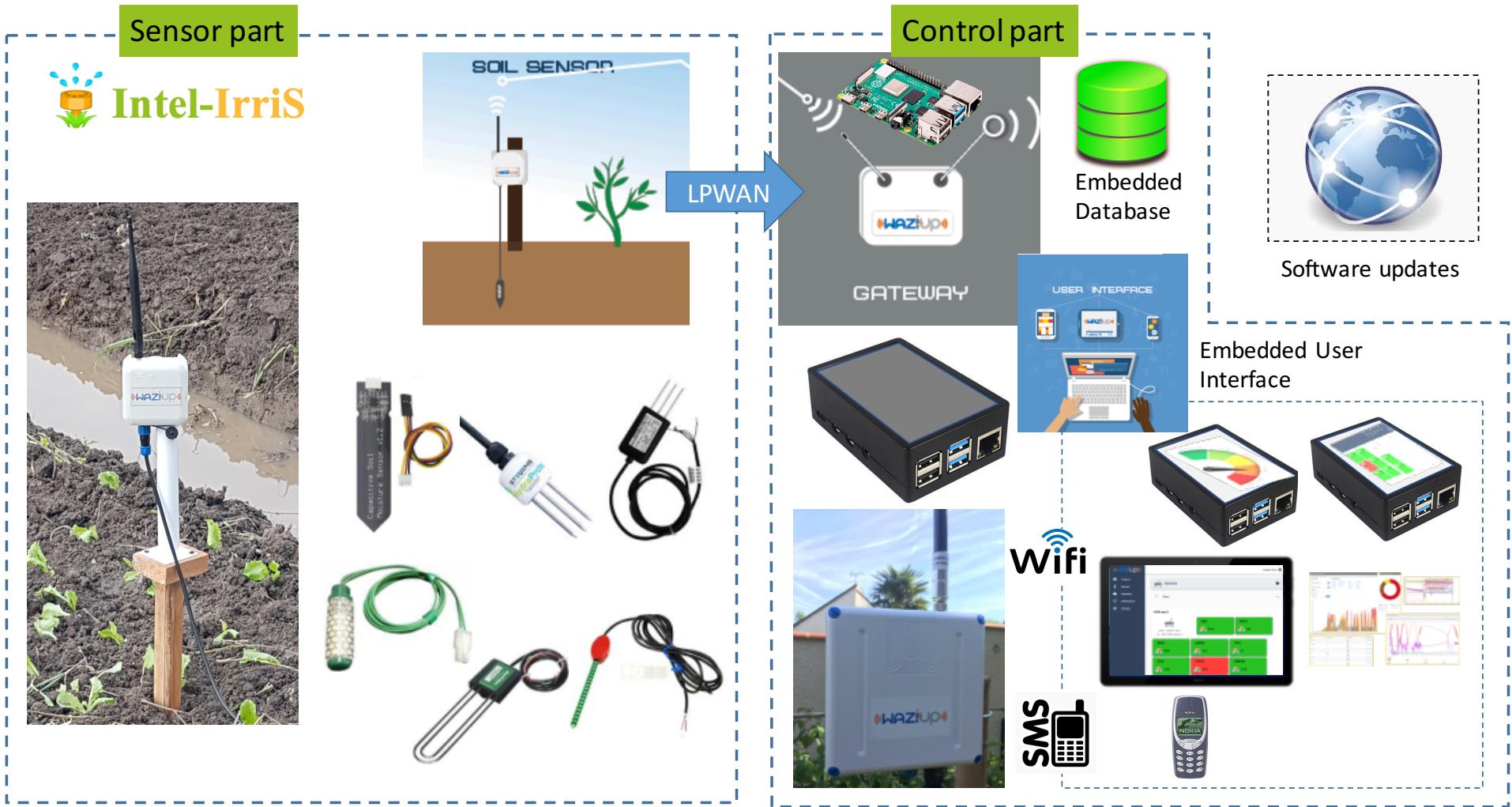
Building the Intel-IrriS LoRa IoT platform Part 2: edge-enabled gateway (WaziGate)



Prof. Congduc Pham
<http://www.univ-pau.fr/~cpham>
Université de Pau, France



Review: Technology components



Review: Low-cost sensors



- Build on low-cost, low-power IoT expertise
- Increase accuracy of low-cost sensors by automatic and remotely controlled procedures for advanced calibration
- Enable deployment of several complementary low-cost sensors
- Include agricultural models / knowledge with corrective & predictive analytics

Review: Smart embedded control

- Build on low-cost embedded & open IoT gateway expertise
- Implement the “Intelligent Irrigation in-the-box” with "plug-&-sense" approach
- Model complex water-soil-plant interaction
- Embed Decision Support System (DSS) and disruptive Artificial Intelligence (AI)
- Integration of various knowledge streams
- Fully autonomous



INTEL-IRRIS starter-kit

- "Intelligent Irrigation in-the-box", "plug-&-sense"
- From idea to reality!



WaziGate

- WaziGate is an IoT LoRa Gateway developed by WAZIUP
- WaziGate implements the edge-enabled IoT gateway approach
 - customized applications can be directly hosted in the gateway
 - the gateway can easily work without Internet connectivity
 - data are available to end-users in an embedded database
 - web-based visualization module provides graphical user interface
- You can find all the WaziGate documentation on the [WaziGate documentation page](#). There are 4 main sections describing the generic WaziGate main features:
 - Quick start: https://www.waziup.io/documentation/wazigate/v2/quick_start/
 - Installation: <https://www.waziup.io/documentation/wazigate/v2/install/>
 - LoRaWAN: <https://www.waziup.io/documentation/wazigate/v2/lorawan/>
 - WaziApps: <https://www.waziup.io/documentation/wazigate/v2/waziapps/>
 - Look at the generic installation video: <https://youtu.be/DvGdmdsGZHA>

Install your WaziGate (1)

● **INTEL-IRRIS WaziGate distribution (RPI3B/3B+/4B)**

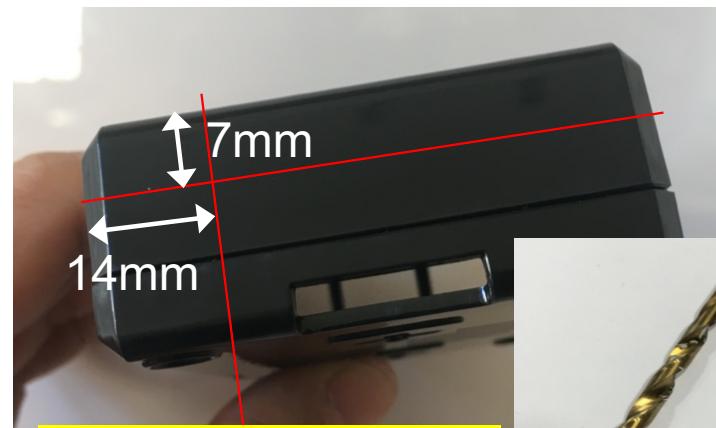
- comes pre-configured with a soil sensor device
- will work out-of-the box with the INTEL-IRRIS soil sensor device
- will be updated to host the INTEL-IRRIS irrigation application
- **Download the INTEL-IRRIS WaziGate SD card image from project website**
- <http://intel-iris.eu/results>
- Image uses EU433 frequency band
- EU868 can be configured afterwards
- Flash the SD card image on 8GB SD card class 10



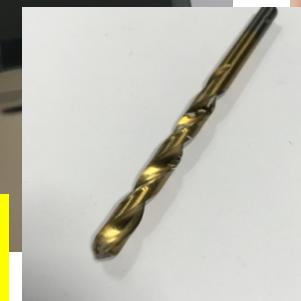
Look at the INTEL-IRRIS WaziGate video
Video n°4 at t=124s: <https://youtu.be/j-1Nk0tv0xM?t=124>

Install your WaziGate (2)

- Recommended RPI model is RPI3B
 - plug in the radio LoRa hat
 - The OLED screen is optional but highly recommended
- Drill a hole for the SMA connector on the RPI enclosure, at the SD card slot side

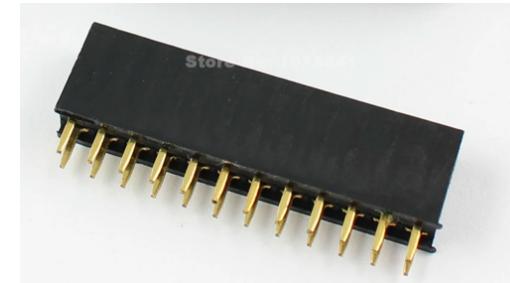
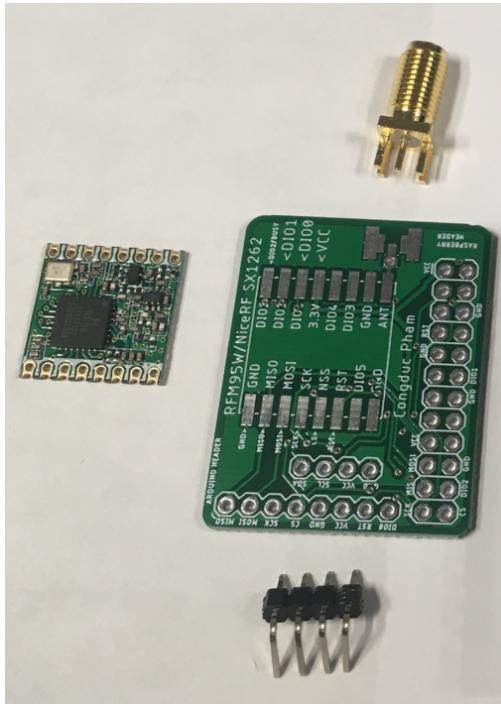


Use from 8mm to 12mm
drill bit for metal



The LoRa radio hat

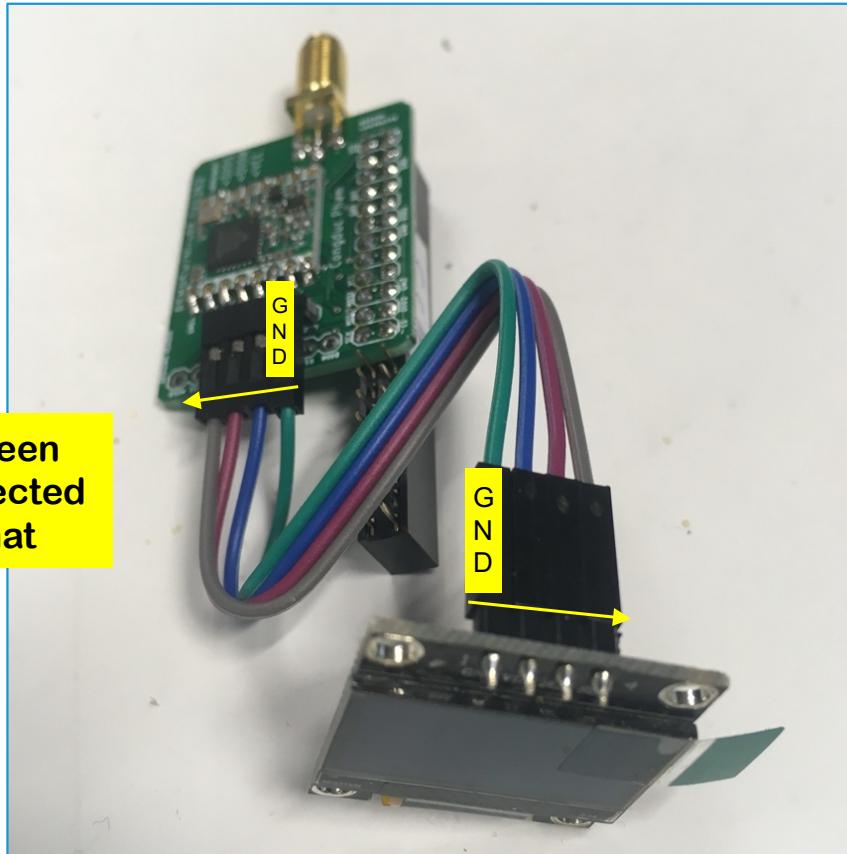
<https://github.com/CongducPham/PRIMA-Intel-Irris/blob/main/Tutorials/Intel-Irris-PCB.pdf>



2-row 12-pin header
or
2 X 1-row 12-pin header



Installing OLED (recommended)

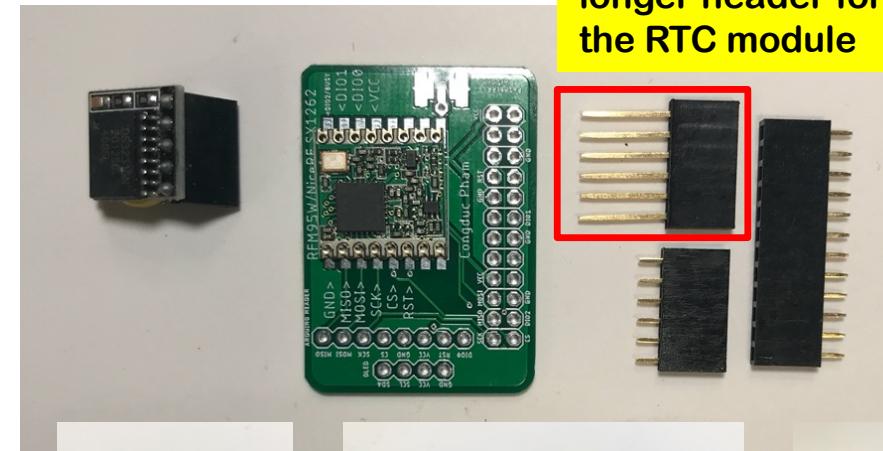


Connecting an RTC module (1)

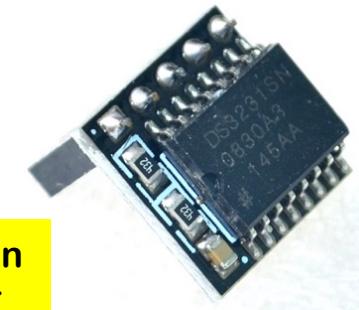
- Several possibilities depending on availability of components



Solder a header with longer pins to connect the RTC module. You can then cut the remaining pins if you want



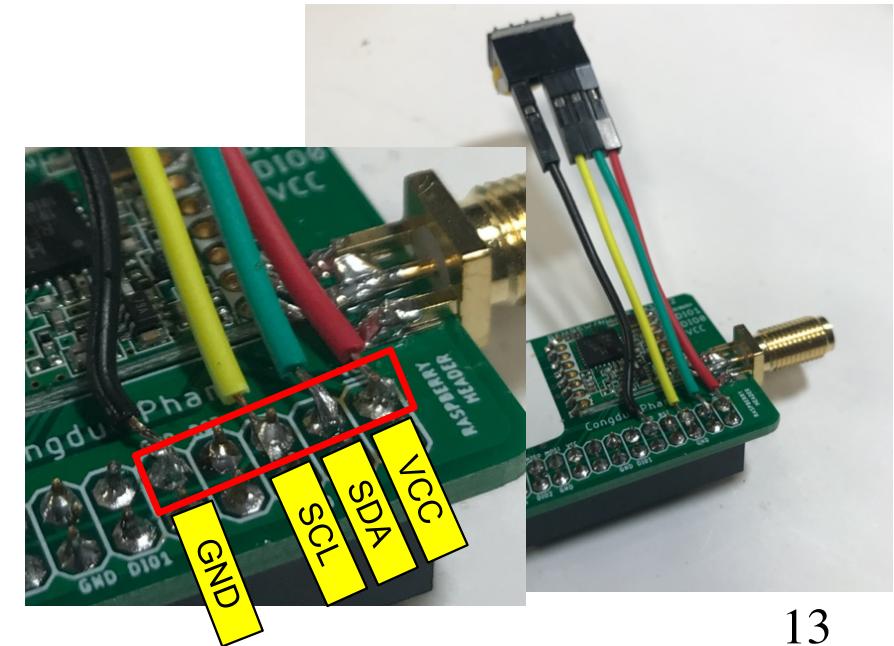
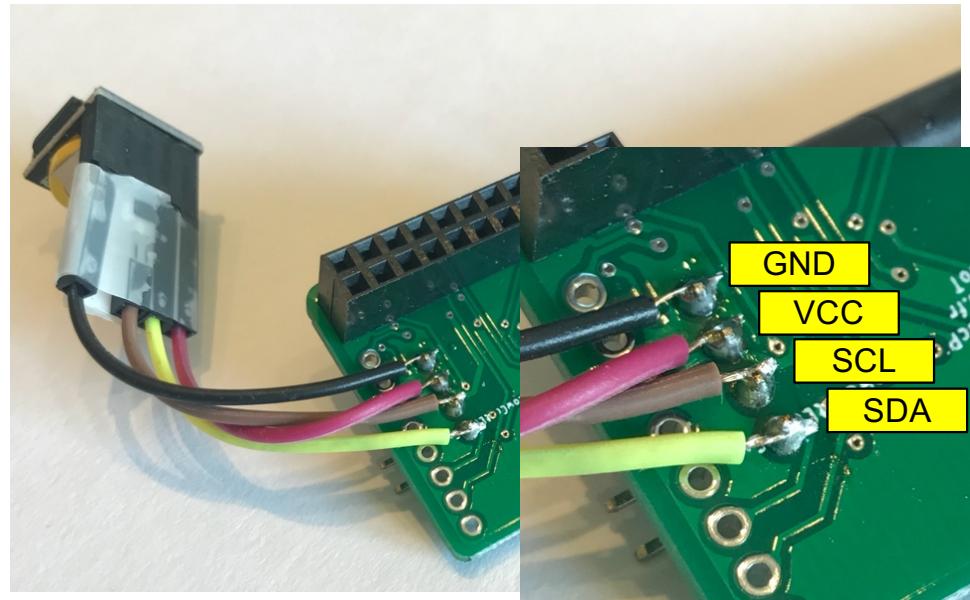
Or use only a 6-pin longer header for the RTC module



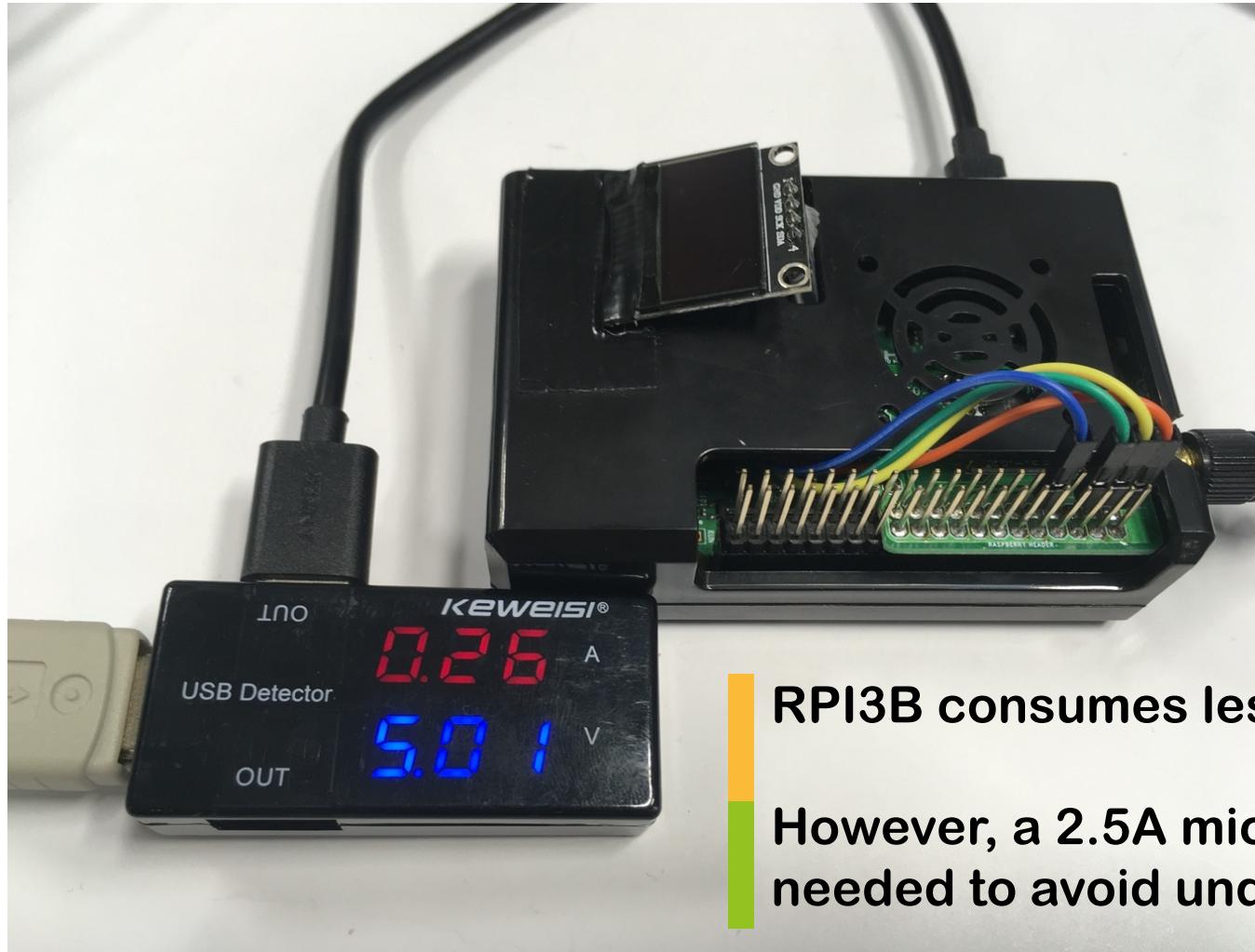
Connecting an RTC module (2)

- If you already have the LoRa hat, you can also just solder 4 wires for the RTC module. 2 possibilities
 - On the back side of the I2C header (left figure)
 - On the RPI GPIO header (right figure)
- Check carefully to avoid short circuits

Pin#	NAME
01	3.3v DC Power
03	GPIO02 (SDA1 , I ^C)
05	GPIO03 (SCL1 , I ^C)
07	GPIO04 (GPIO_GCLK)
09	Ground
11	GPIO17 (GPIO_GEN0)
13	GPIO27 (GPIO_GEN2)



WaziGate power consumption

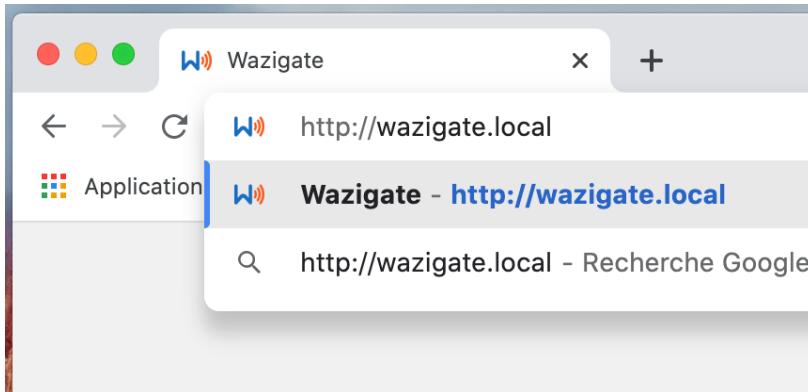
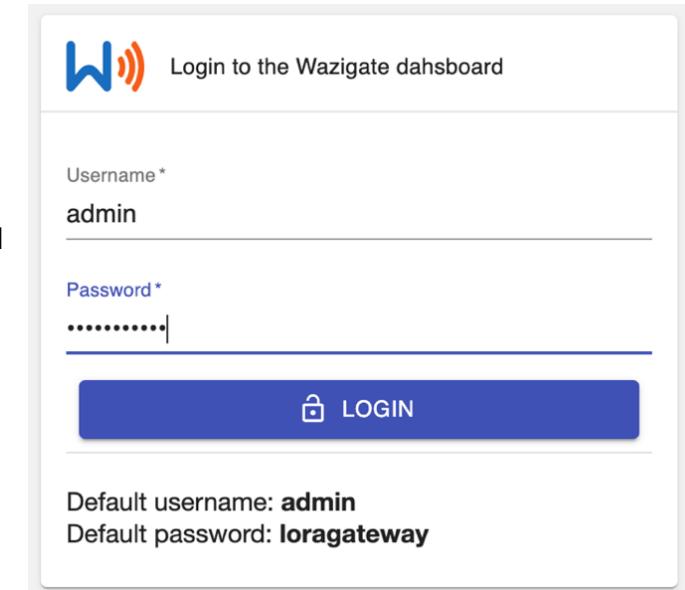


RPI3B consumes less than 300mA

However, a 2.5A micro USB charger is needed to avoid undervoltage

Accessing your WaziGate

- Power the WaziGate, no Internet is required, wait 3-4mins (boot)
- Connect to **WAZIGATE_XXXXXXXXXXXXXX** WiFi network
 - default WiFi password is loragateway
- Open web navigator. Go to <http://wazigate.local> or <http://10.42.0.1>

Login to the Wazigate dashboard

Username*
admin

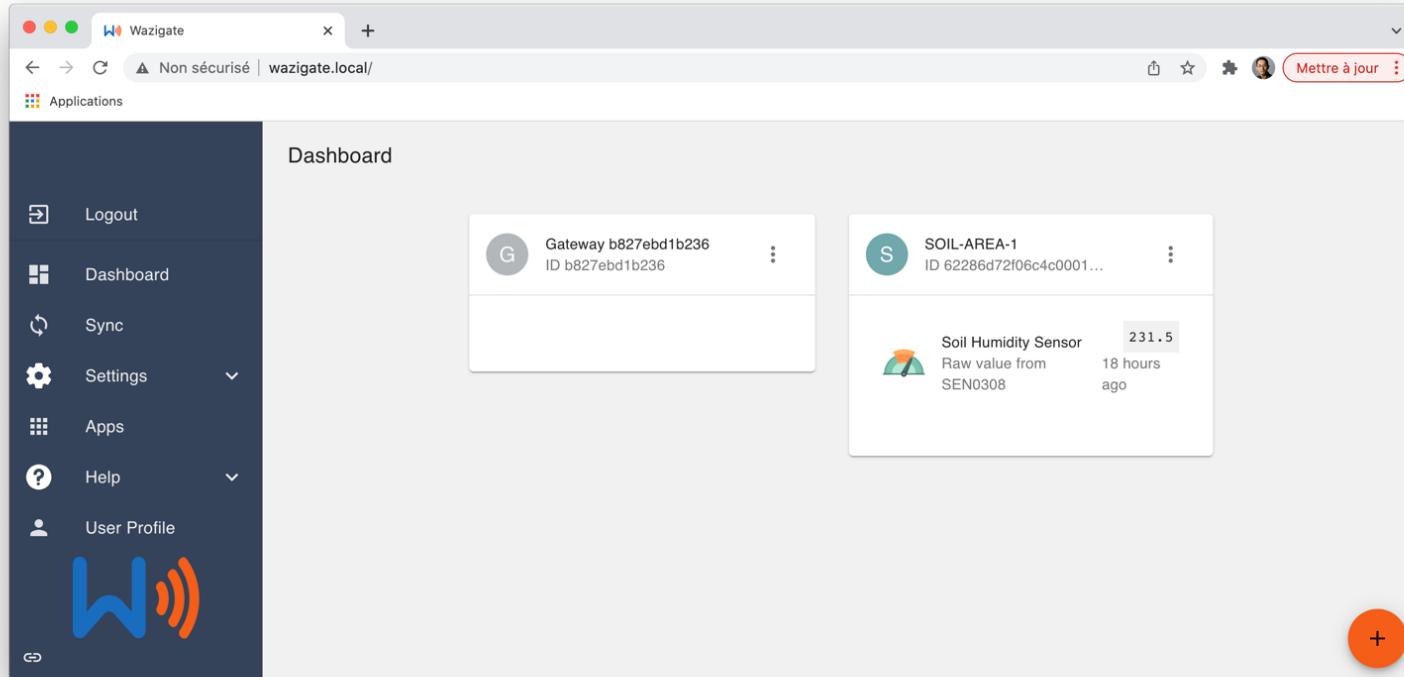
Password*
.....

LOGIN

Default username: **admin**
Default password: **loragateway**

- Use default login to connect
 - User: admin
 - Password: loragateway

WaziGate dashboard



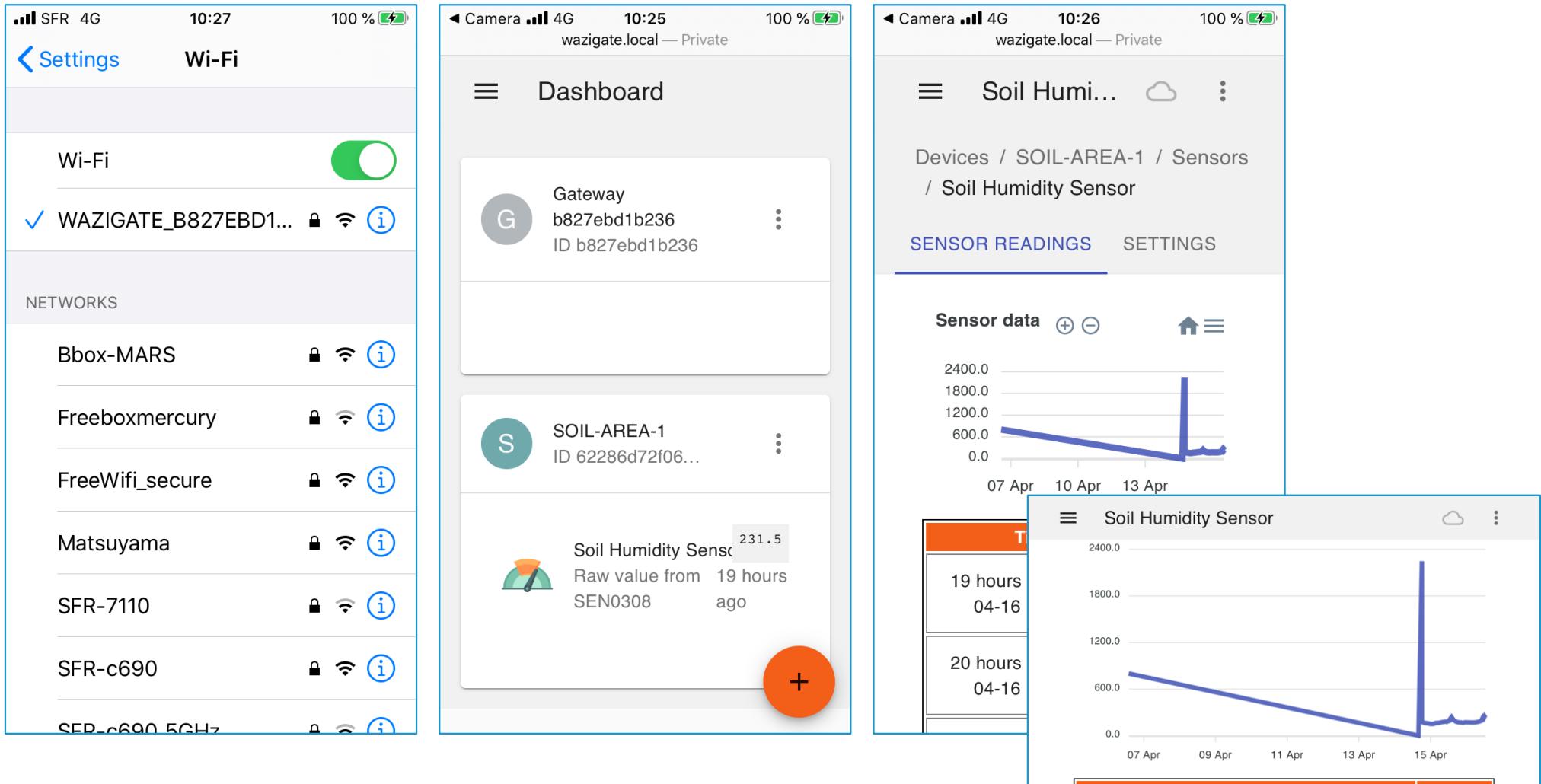
Pre-configured soil sensor device (SOIL-AREA-1) with a capacitive SEN0308 soil humidity sensor ready to receive data

The last received value is displayed in the device block

Display sensor data



All these steps with a smartphone



The figure consists of three screenshots of a mobile application interface:

- Screenshot 1: WiFi Settings**
 Shows the device's connectivity status (SFR 4G, 10:27, 100% battery). Under "Wi-Fi", "WAZIGATE_B827EBD1..." is selected. A list of available networks includes "Bbox-MARS", "Freeboxmercury", "FreeWiFi_secure", "Matsuyama", "SFR-7110", "SFR-c690", and "SFR_c690_5GHz".
- Screenshot 2: Dashboard**
 Shows a "Gateway" entry (b827ebd1b236, ID b827ebd1b236) and a "SOIL-AREA-1" entry (ID 62286d72f06...). Below it, a "Soil Humidity Sensor" card shows a raw value of 231.5 from SEN0308 19 hours ago. A large orange "+" button is at the bottom right.
- Screenshot 3: Sensor Data**
 Shows "Sensor data" for a "Soil Humidity Sensor". A graph plots values from April 7 to April 15. Two data series are shown: one for "19 hours 04-16" and another for "20 hours 04-16". Both show a sharp drop from approximately 600 to 0 around April 14.

Configure soil device for WaziGate

Intelirris_Soil_Sensor | Arduino 1.8.13

```

  Intelirris_Soil_Sensor DS18B20.cpp DS18B20.h RadioSettings.h SX126X_RadioSettings.h SX127X_RadioSettings.h SX128X_RadioSettings.h

27 /*
28
29 */
30 ****
31 /--\ /--\ /--\
32 | / \ | / \ | / \
33 | | / \ | / \ | / \
34 | | \ / \ | / \ | / \
35 \--\ / \ | / \ | / \
36 | | / \ | / \ | / \
37 | |
38 ****
39
40 /////////////////
41 // sends data to INTEL-IRRIS WaziGate edge-gateway
42 #define TO_WAZIGATE
43
44 /////////////////
45 // Frequency band - do not change in SX127X_RadioSettings.h anymore
46 //##define BAND868
47 //##define BAND900
48 #define BAND433
49
50 ///////////////
51 // Test device
52 ##define TEST_DEVICE_RANDOM
53
54 ///////////////
55 // uncomment to have a soil tensiometer watermark sensor
56 //##define WITH_WATERMARK
57

```

42 Arduino Pro or Pro Mini, ATmega328P (3.3V, 8 MHz) on /dev/cu.usbserial-AK05C49Q

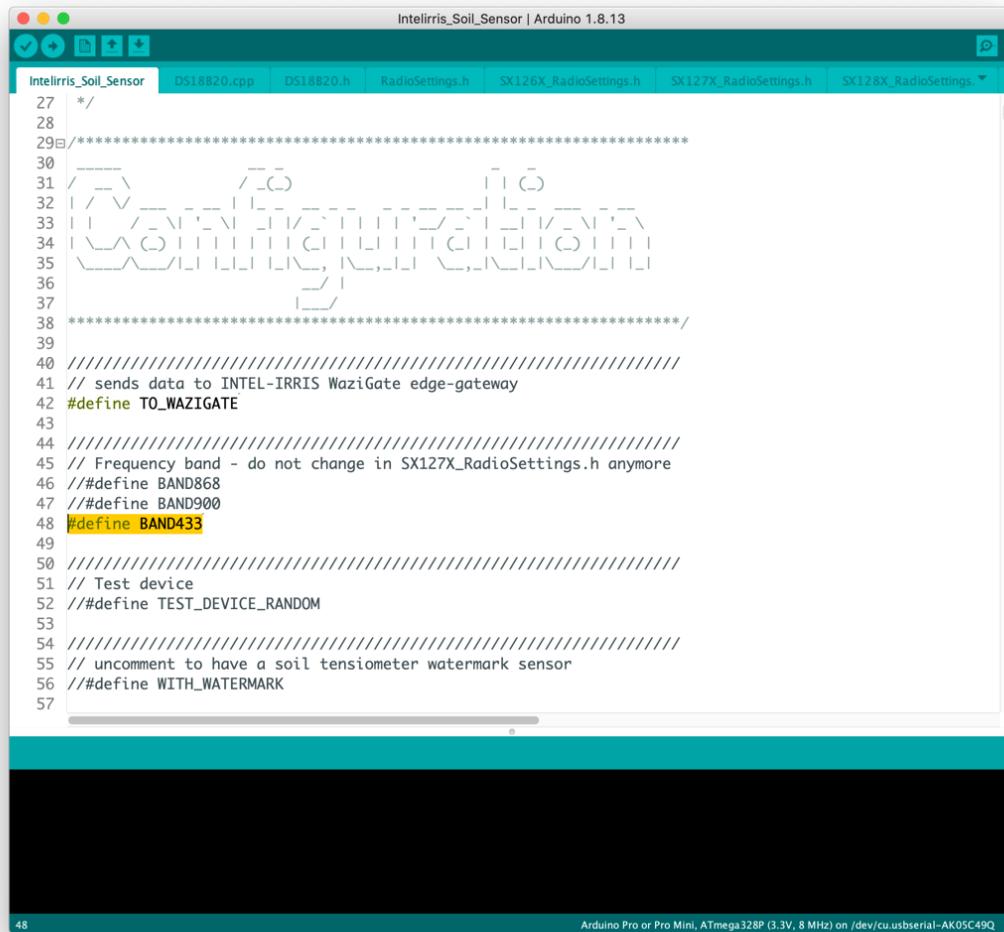
Be sure that

#define TO_WAZIGATE

is uncommented

Configuring for EU433 band

Intelirris_Soil_Sensor | Arduino 1.8.13



```

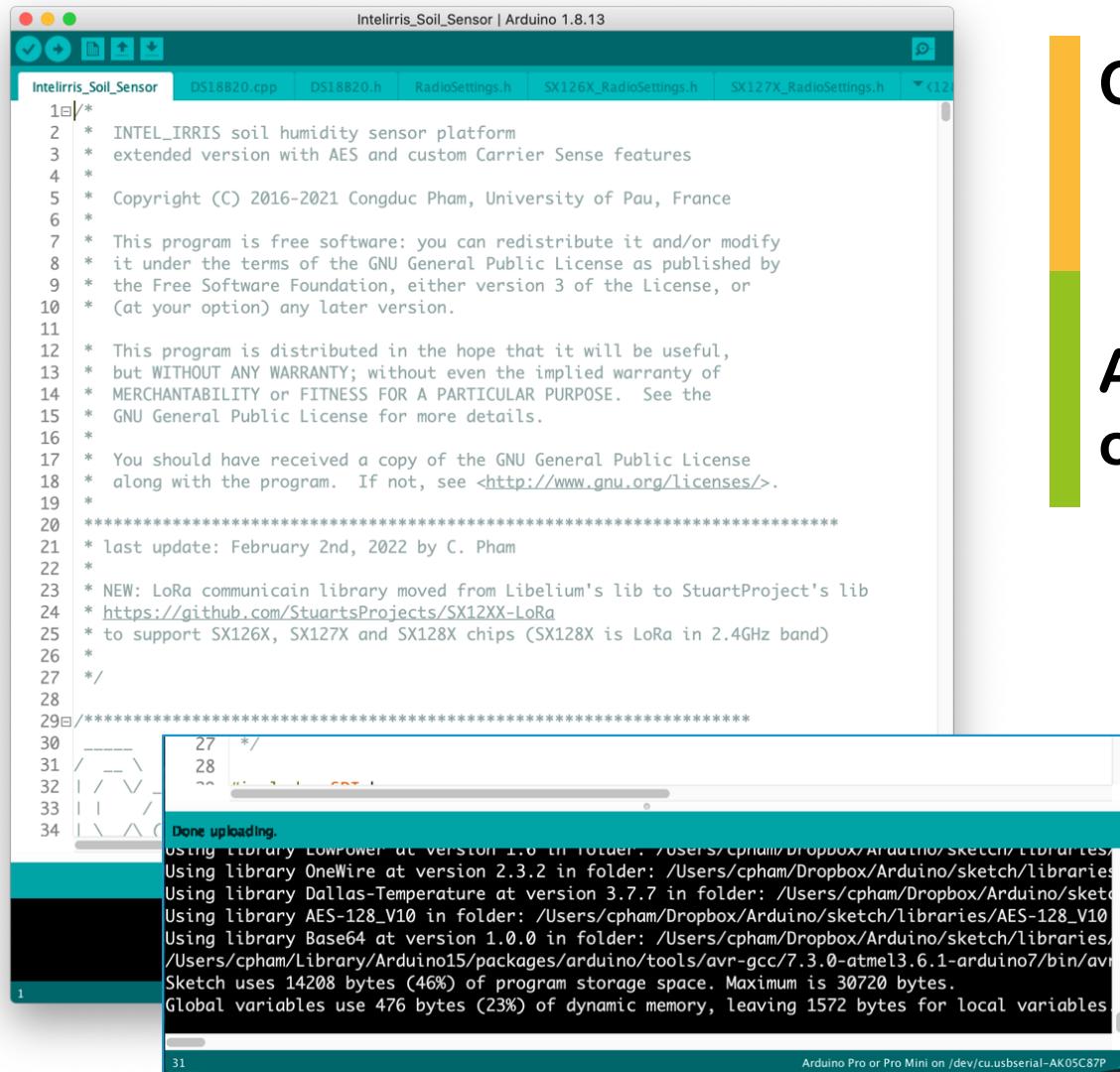
27 */
28
29 ****
30 -----
31 / _ \   / _ \   / _ \
32 / \ _ _ \ / \ _ _ \ / \ _ \
33 / \ _ _ \ / \ _ _ \ / \ _ _ \
34 / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ \
35 / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ \
36 / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ _ \
37 / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ _ \ / \ _ _ \
38 ****
39
40 // sends data to INTEL-IRRIS WaziGate edge-gateway
41 #define TO_WAZIGATE
42
43
44
45 // Frequency band - do not change in SX127X_RadioSettings.h anymore
46 //#define BAND868
47 //#define BAND900
48 #define BAND433
49
50
51 // Test device
52 //#define TEST_DEVICE_RANDOM
53
54
55 // uncomment to have a soil tensiometer watermark sensor
56 //#define WITH_WATERMARK
57

```

48 Arduino Pro or Pro Mini, ATmega328P (3.3V, 8 MHz) on /dev/cu.usbserial-AK05C49Q

If you use the EU433 band
 make sure that
`#define BAND433`
 is the only uncommented
 band option

Uploading to your board



```

1/* 
2 * INTEL_IRRIS soil humidity sensor platform
3 * extended version with AES and custom Carrier Sense features
4 *
5 * Copyright (C) 2016-2021 Congduc Pham, University of Pau, France
6 *
7 * This program is free software: you can redistribute it and/or modify
8 * it under the terms of the GNU General Public License as published by
9 * the Free Software Foundation, either version 3 of the License, or
10 * (at your option) any later version.
11 *
12 * This program is distributed in the hope that it will be useful,
13 * but WITHOUT ANY WARRANTY; without even the implied warranty of
14 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
15 * GNU General Public License for more details.
16 *
17 * You should have received a copy of the GNU General Public License
18 * along with the program. If not, see <http://www.gnu.org/licenses/>.
19 *
20 ****
21 * last update: February 2nd, 2022 by C. Pham
22 *
23 * NEW: LoRa communicain library moved from Libelium's lib to StuartProject's lib
24 * https://github.com/StuartsProjects/SX12XX-LoRa
25 * to support SX126X, SX127X and SX128X chips (SX128X is LoRa in 2.4GHz band)
26 *
27 */
28
29 ****
30 */
31 */
32 */
33 */
34 */

Done uploading.

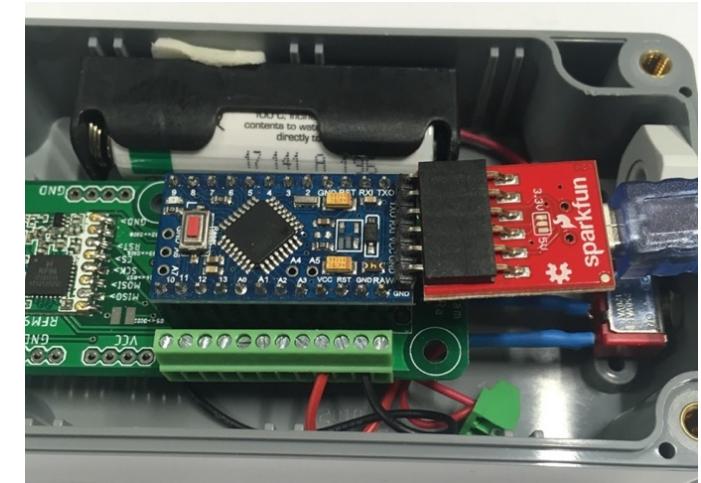
Using library LowPower at version 1.0 in folder: /Users/cpham/Dropbox/Arduino/sketch/LowPower
Using library OneWire at version 2.3.2 in folder: /Users/cpham/Dropbox/Arduino/sketch/libraries/OneWire
Using library Dallas-Temperature at version 3.7.7 in folder: /Users/cpham/Dropbox/Arduino/sketch/libraries/Dallas-Temperature
Using library AES-128_V10 in folder: /Users/cpham/Dropbox/Arduino/sketch/libraries/AES-128_V10
Using library Base64 at version 1.0.0 in folder: /Users/cpham/Dropbox/Arduino/sketch/libraries/Base64
/Users/cpham/Library/Arduino15/packages/arduino/tools/avr-gcc/7.3.0-atmel3.6.1-arduino7/bin/avr-gcc
Sketch uses 14208 bytes (46%) of program storage space. Maximum is 30720 bytes.
Global variables use 476 bytes (23%) of dynamic memory, leaving 1572 bytes for local variables.

  
```

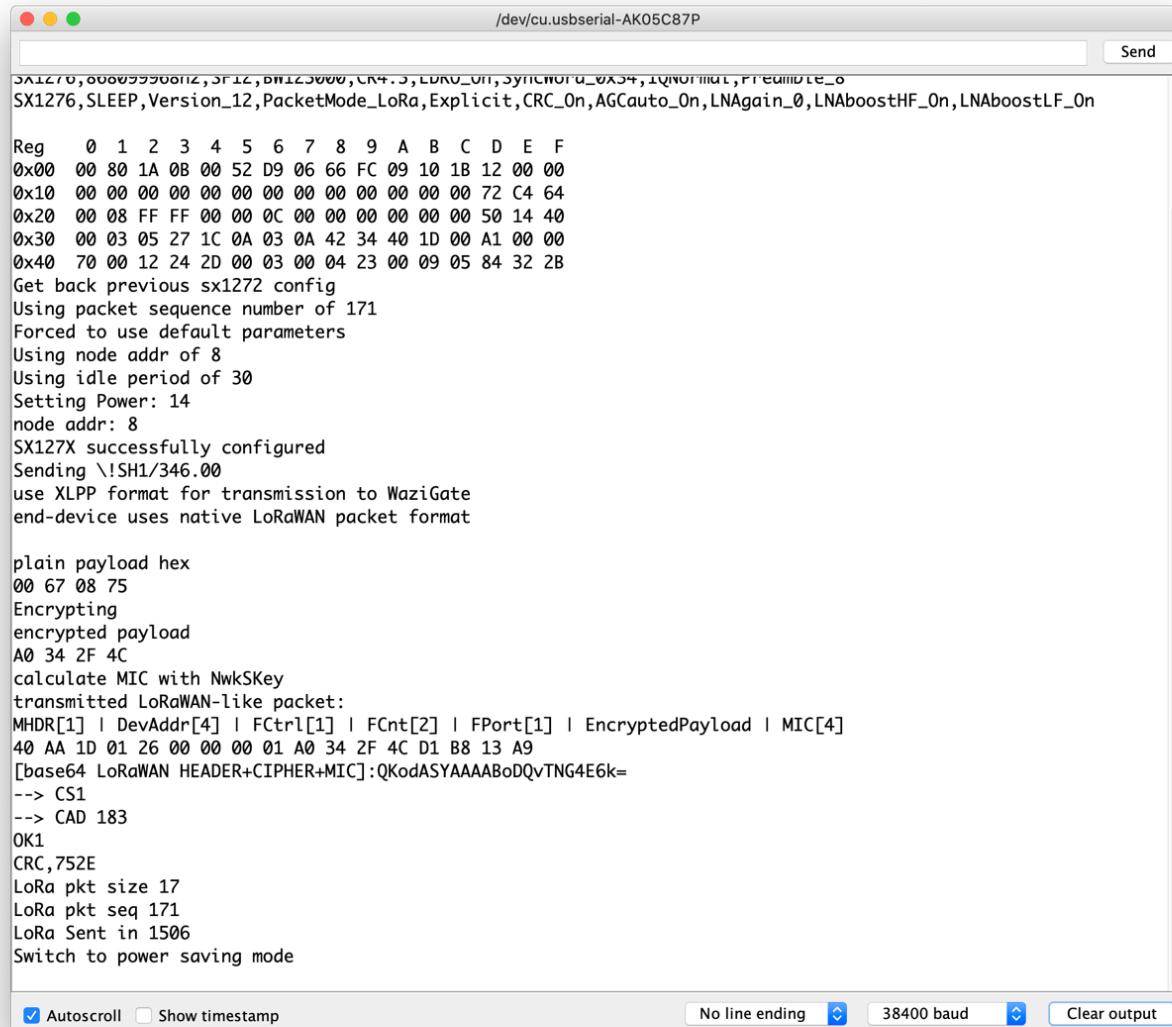
Click on the "upload" button



And wait until upload is completed



Checking that device is operational



```

/dev/cu.usbserial-AK05C87P
Send

SX1276,0000000000000000,3F12,B0123000,CR4.5,LDRU_0H,SYNCHRO_0X34,TQNormal,Preamble_0
SX1276,SLEEP,Version_12,PacketMode_LoRa,Explicit,CRC_On,AGCAuto_On,LNAgain_0,LNAboostHF_On,LNAboostLF_On

Reg 0 1 2 3 4 5 6 7 8 9 A B C D E F
0x00 00 80 1A 0B 00 52 D9 06 66 FC 09 10 1B 12 00 00
0x10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 72 C4 64
0x20 00 08 FF FF 00 00 0C 00 00 00 00 00 00 00 50 14 40
0x30 00 03 05 27 1C 0A 03 0A 42 34 40 1D 00 A1 00 00
0x40 70 00 12 24 2D 00 03 00 04 23 00 09 05 84 32 2B
Get back previous sx1272 config
Using packet sequence number of 171
Forced to use default parameters
Using node addr of 8
Using idle period of 30
Setting Power: 14
node addr: 8
SX127X successfully configured
Sending \!SH1/346.00
use XLPP format for transmission to WaziGate
end-device uses native LoRaWAN packet format

plain payload hex
00 67 08 75
Encrypting
encrypted payload
A0 34 2F 4C
calculate MIC with NwkSKey
transmitted LoRaWAN-like packet:
MHDR[1] | DevAddr[4] | FCtrl[1] | FCnt[2] | FPort[1] | EncryptedPayload | MIC[4]
40 AA 1D 01 26 00 00 00 01 A0 34 2F 4C D1 B8 13 A9
[base64 LoRaWAN HEADER+CIPHER+MIC]:QKodASYAAABoDQvTNG4E6k=
--> CS1
--> CAD 183
OK1
CRC,752E
LoRa pkt size 17
LoRa pkt seq 171
LoRa Sent in 1506
Switch to power saving mode

 Autoscroll  Show timestamp
  No line ending 38400 baud Clear output

```

Open serial monitor

Set baud rate to 38400

See output from board

Check that transmission is OK

Transmission to WaziGate



Parameters for
INTEL-IRRIS WaziGate



SF12BW125
868.1MHz | 433.175MHz
Node id is 26011DAA
1 msg/60mins
1 sensor
XLPP data

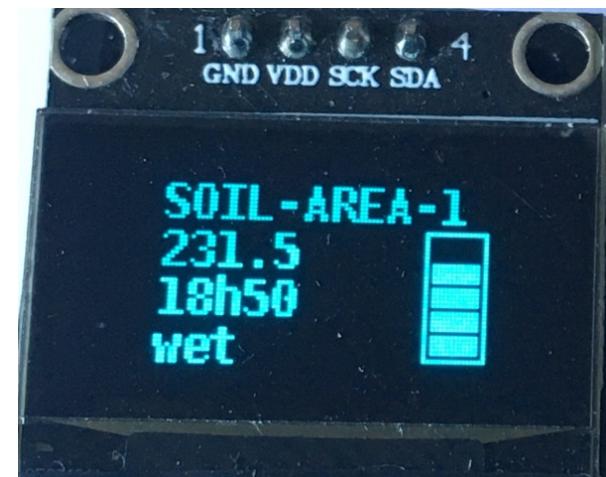


This dedicated video will show all these steps, from connecting the SEN0308 to testing transmission to the WaziGate
Video n°4: <https://youtu.be/j-1Nk0tv0xM>



With the OLED screen

- With a small .96" OLED screen, information summary is displayed for the end-user: the device name, the time of last received data, the sensor raw value and the soil condition
- The main screen is displayed for 6s every 30s. Then a screen saver display will show a shorter version of these information with a 5-bar visual
- 5 bars: very wet | 4 bars: wet
- 3 bars: wet-dry | 2 bars: dry-wet
- 1 bar: dry | 0 bar: very dry



QR code for connecting to WiFi

- The WaziGate WiFi is WAZIGATE_XXXXXXXXXXXX where XXXXXXXXXXXX is the MAC address of the Raspberry
- For instance WAZIGATE_B827EBD1B236
- With the OLED, a QR code for joining the WiFi network is generated dynamically at boot time and displayed for 10s before the main screen so that users can automatically join with a smartphone
- Then, users can scan the static QR code on the WaziGate sticker to connect to the WaziGate's dashboard or the INTEL-IRRIS WaziApp

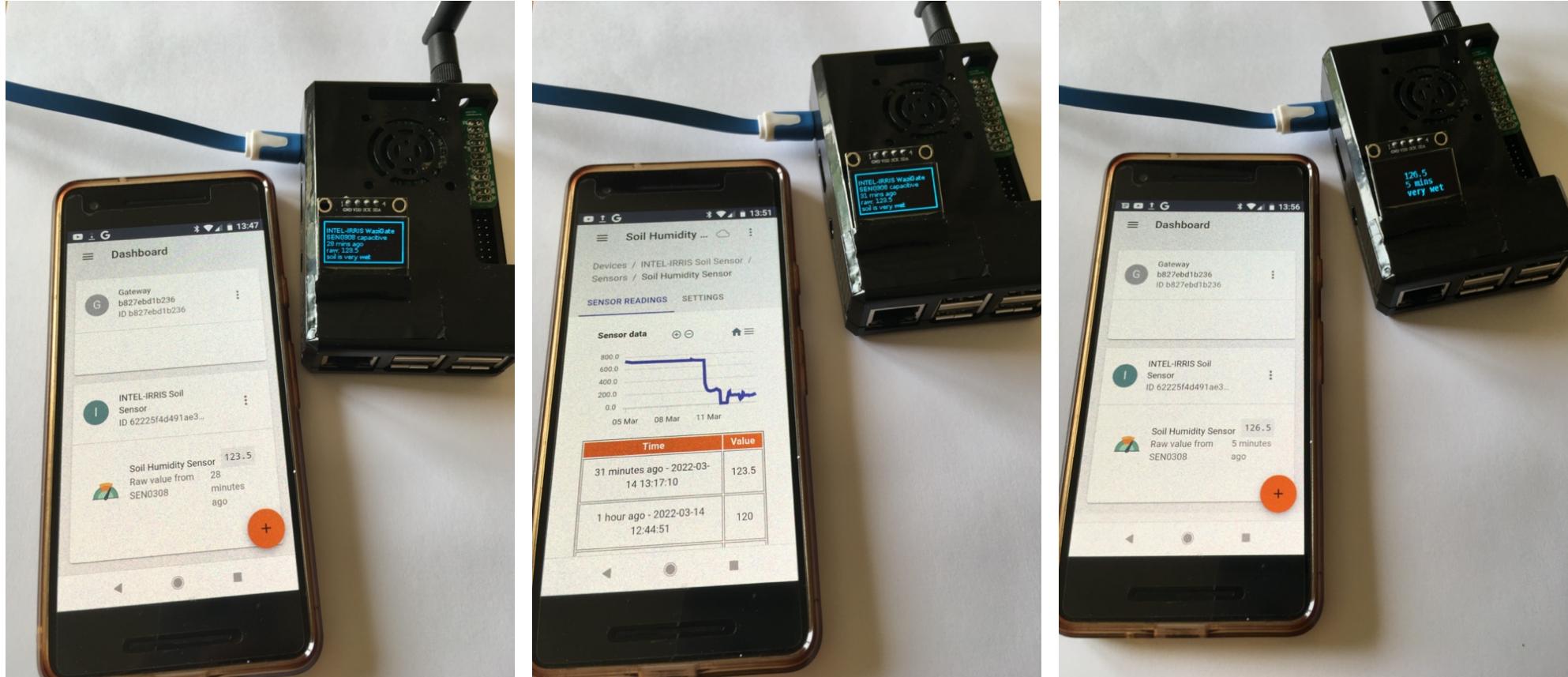
WAZIGATE_B827EBD1B236



WAZIGATE_DCA6325C2A7A



WaziGate User Interface



The WaziGate provides the simple OLED interface but also more advanced features through the WaziGate dashboard and embedded application interface (WaziApps)

A dedicated INTEL-IRRIS Irrigation WaziApps is currently being developed

Advanced configuration

Synching your WaziGate to the cloud

- If you want to sync your WaziGate to the Waziup Cloud, look at this tutorial page
 - <https://www.waziup.io/documentation/wazigate/v2/install/#registration-with-the-cloud>
- You will need an account on Waziup Cloud dashboard
 - If you don't have one, you need to create one first
 - <https://dashboard.waziup.io/>
- Then, just activate sync on your WaziGate which needs to be connected to Internet

