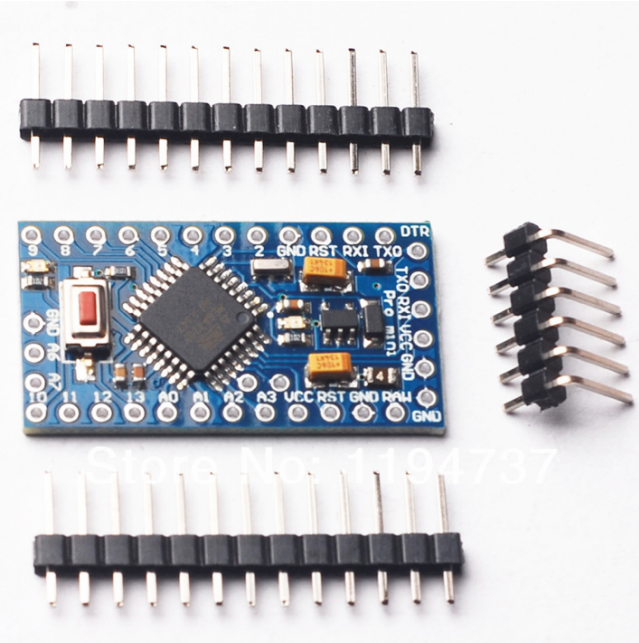
# Low-cost LoRa WaterSense IoT device: a step-by-step tutorial


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

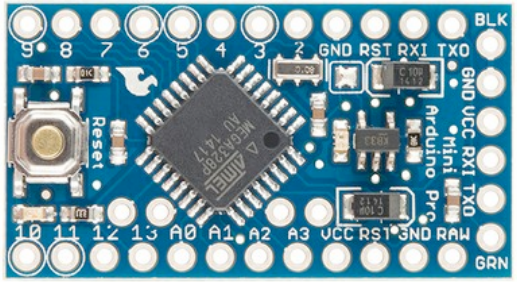
# Contents

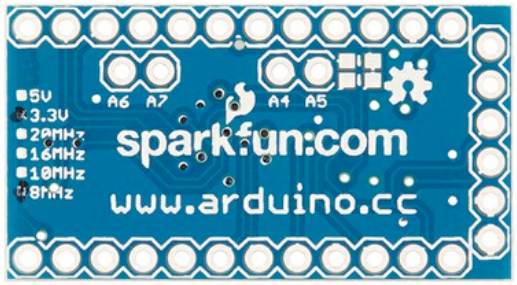
- We will show how to build a low-cost IoT device for sensing applications using LoRa radio technology
- The gateway part will be shown in a separate tutorial
- The target hardware platform is an Arduino Pro Mini in its 3.3v, 8MHz version: the original from Sparkfun or a clone from various providers
- Let's get started…

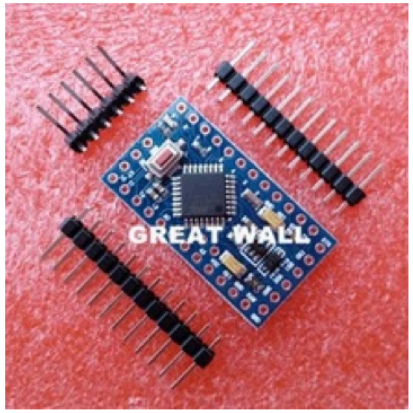
# Assembling the hardware



# Get the Arduino board




With the bootloader 1pcs **pro mini** atmega328 **Pro Mini** 328 **Mini** ATMEGA328 3.3V/8MHz for **Arduino**

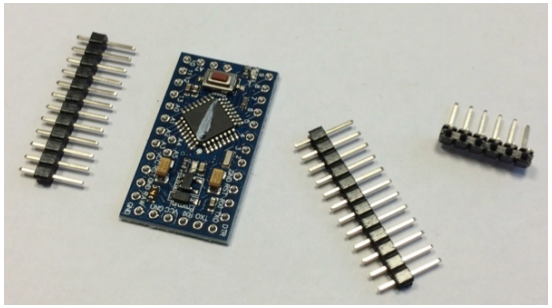
GREAT WALL Electronics Co., Ltd.

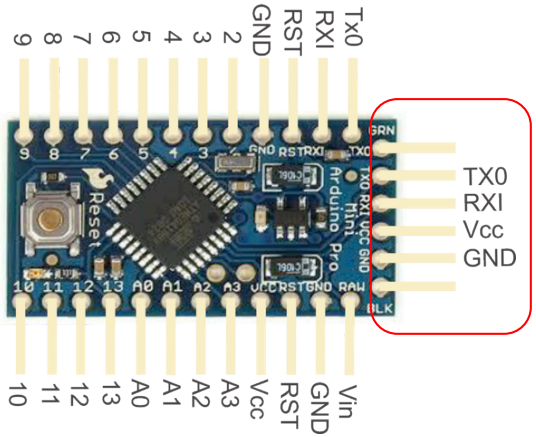
Chat now!

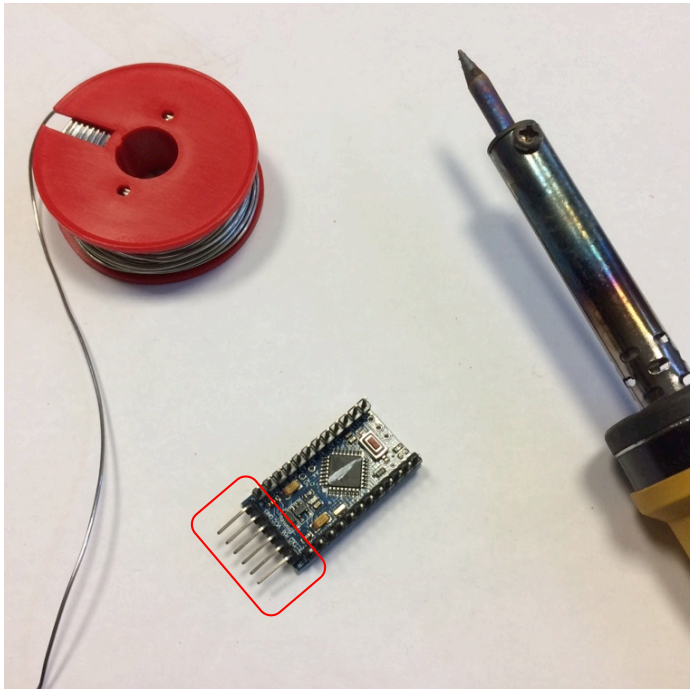
Be sure to get the 3.3v and 8MHz version

You can get the original board designed by Sparkfun or get one of the various clones available mainly from Chinese manufacturer. The last solution is very cost-effective as the Pro Mini board can be obtained for less than 2€ a piece. Some boards may not be working as reported by some people but in my own experience, all the boards I got from Chinese manufacters have been working great.

# Prepare the board (1)

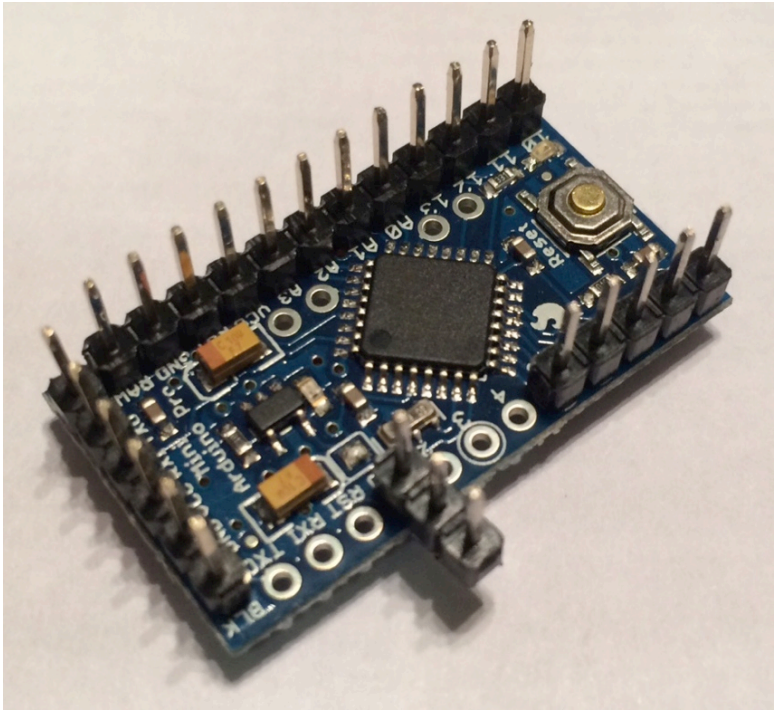


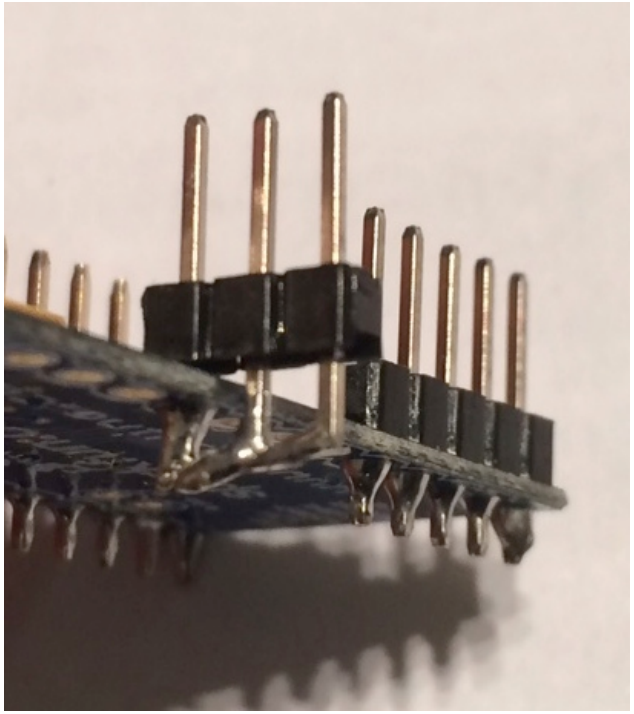


When you receive the board, it will probably come with the appropriate header pins that must be soldered to the board. Just use a regular soldering station to solder the header pins to the board. The 6-pin header on one side of the board (see red rectangle) will be used to connect an FTDI cable to program the board. This will be explained in the « software » section.

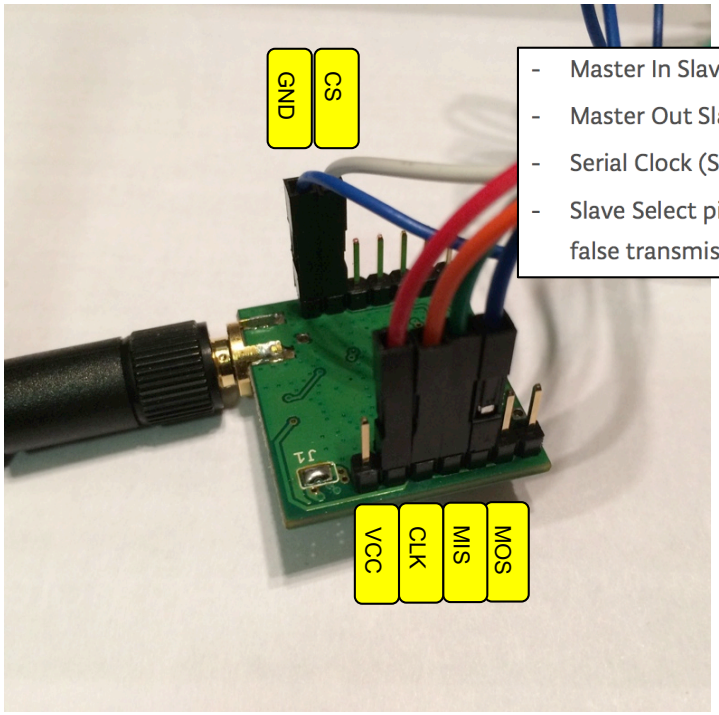
# Prepare the board (2)



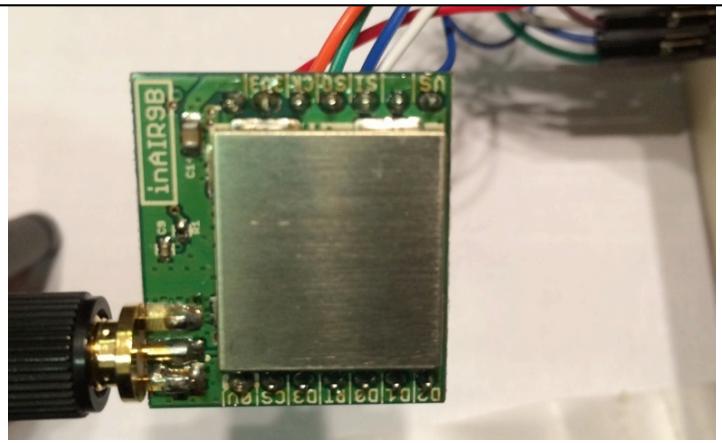


You can solder a row of header pins (left) on the ground pin (GND) in order to have several ground pins for the various sensors that will be connected to the device. But don't forget to link all the pins together to get the GND signal on all the pins (right).

# Now the radio module

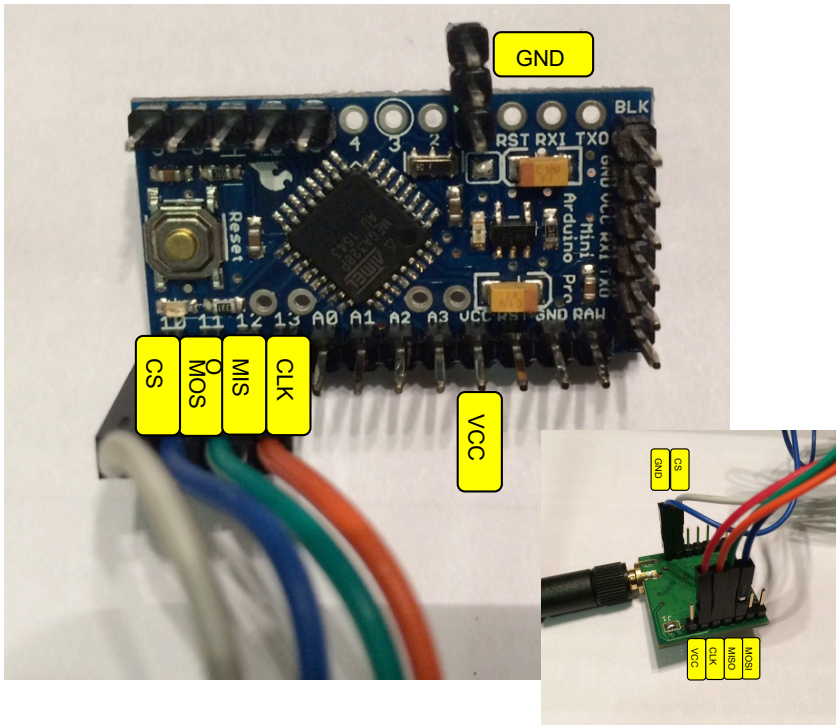



- Master In Slave Out (MISO) - The Slave line for sending data to the master,
- Master Out Slave In (MOSI) - The Master line for sending data to the peripherals,
- Serial Clock (SCK) - The clock pulses which synchronize data transmission generated by the master, and
- Slave Select pin - allocated on each device which the master can use to enable and disable specific devices and avoid false transmissions due to line noise.



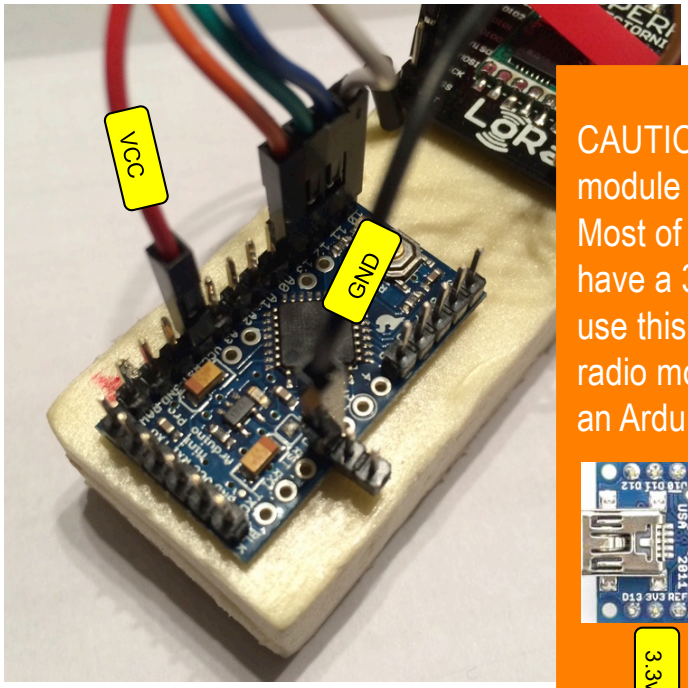

If you go for the inAir9 from Modtronix, then the header pins can come fully assembled. Order with the 6mm header pins to have enough length to connect F/F breadboard cables (left). Connect the SPI pins with the F/F cables. Try to use different colors. I use the following colors: MOSI (blue), MISO (green), CS (white), CLK (orange). Then connect also the VCC (red) and the GND (black or any other dark color) of the radio board.

# Connecting the radio module









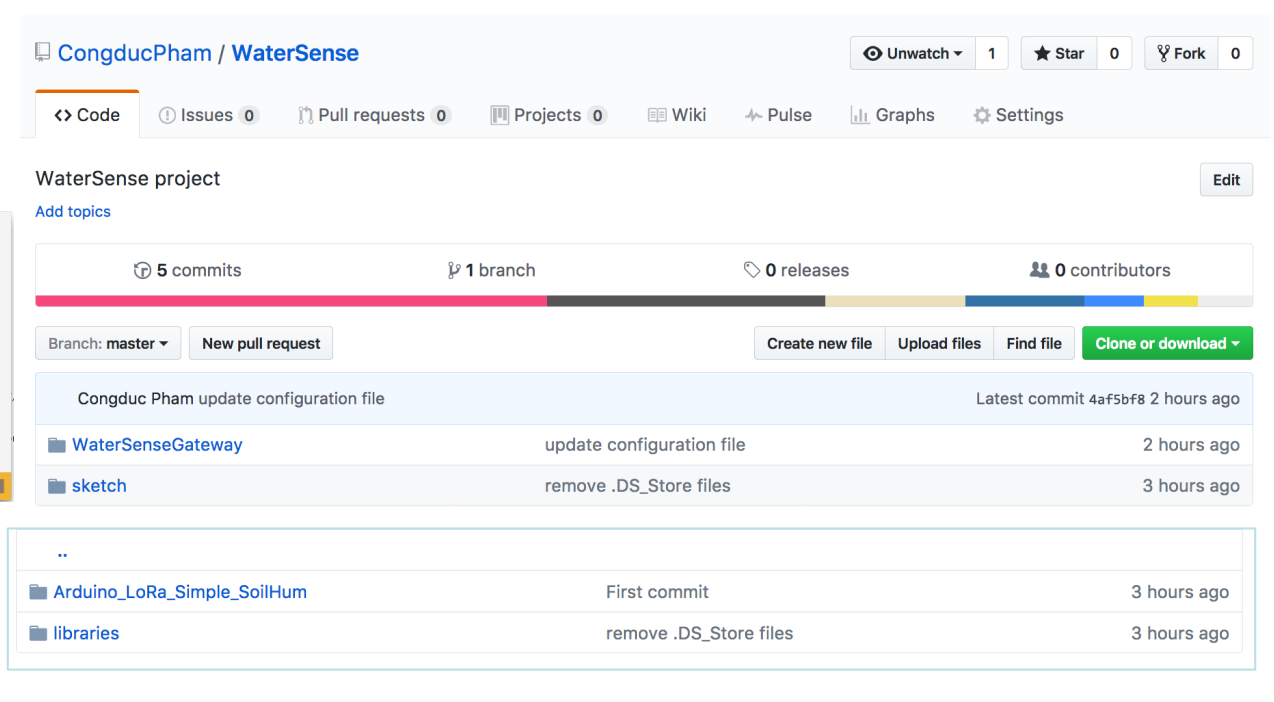

CAUTION: the radio module needs 3.3v, not 5v! Most of Arduino boards have a 3.3v pin so you can use this pin to power the radio module. Example wth an Arduino Nano:

3.3v

Now, just connect the corresponding SPI pins of the radio module to the SPI pins on the board. MOSI (blue) is pin 11, MISO (green) is pin 12, CS (white) is pin 10 and CLK (orange) is pin 13 (left picture). Then connect also the VCC (red) and the GND (black) of the radio board to the VCC and the GND of the board (right picture). The VCC of the board gets 3.3v from the on-board voltage regulator.

# Getting, compiling & uploading the software

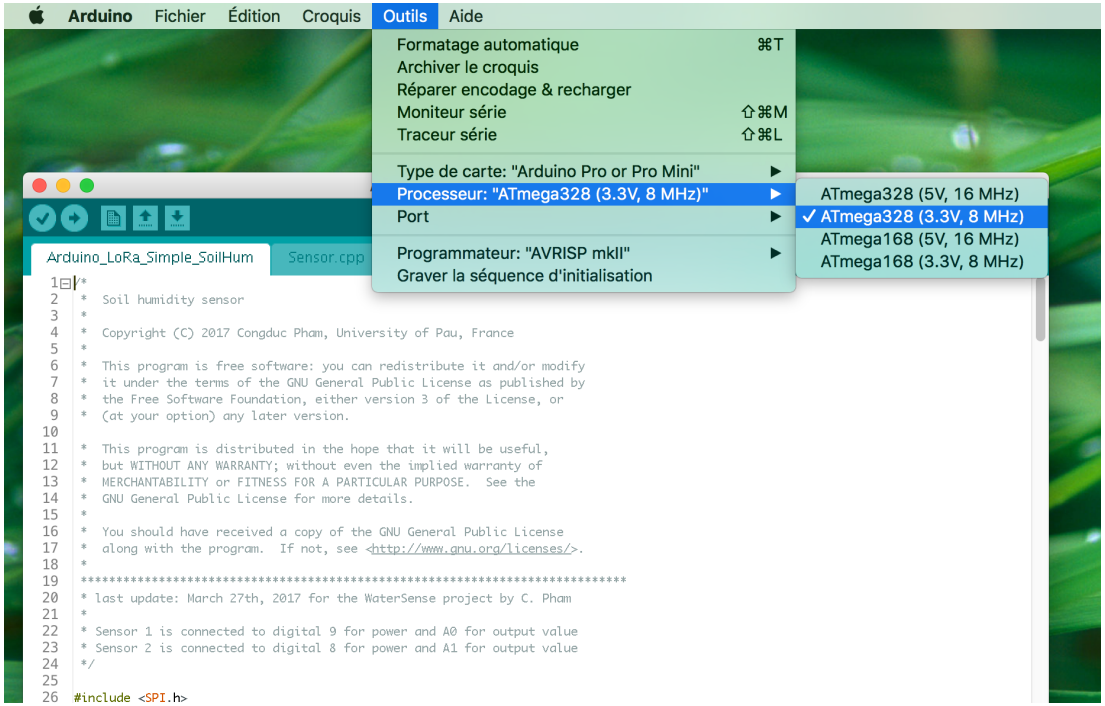
# Getting the software


Fisrt, you will need the Arduino IDE 1.6.6 or later (left). Then get the LoRa library from our github: https://github.com/CongducPham/WaterSense (right).

Get into the sketch folder and get both Arduino_LoRa_Simple_SoilHum and libraries folder. Copy both folders into your “sketch” folder.

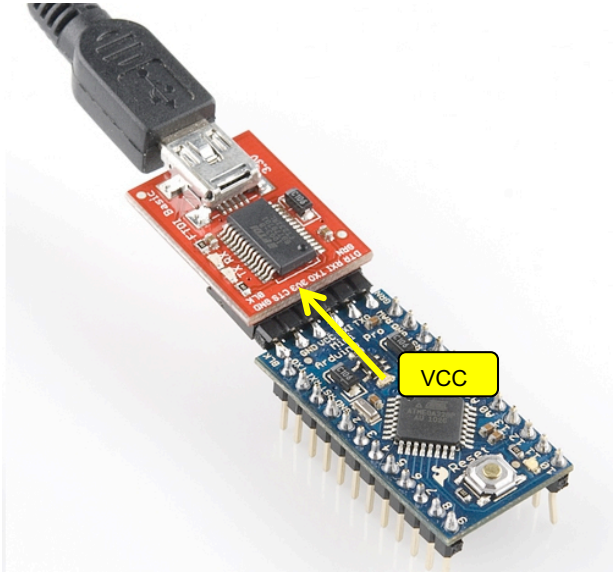
# Compiling

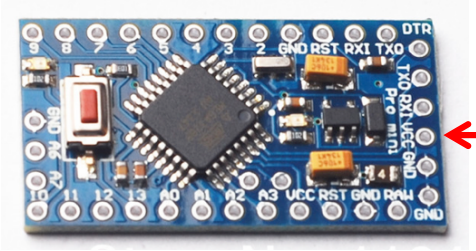

Open the Arduino_LoRa_Simple_SoilHum sketch and select the Arduino Pro Mini board with its 3.3V & 8MHz version.

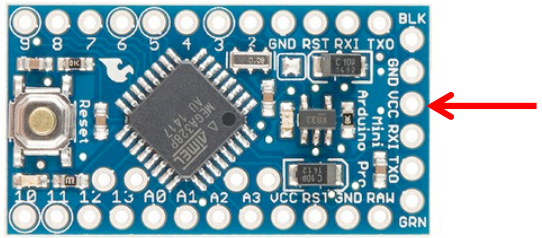
Then, click on the « verify » button 

# Uploading (1)




Some clone version, check the VCC pin

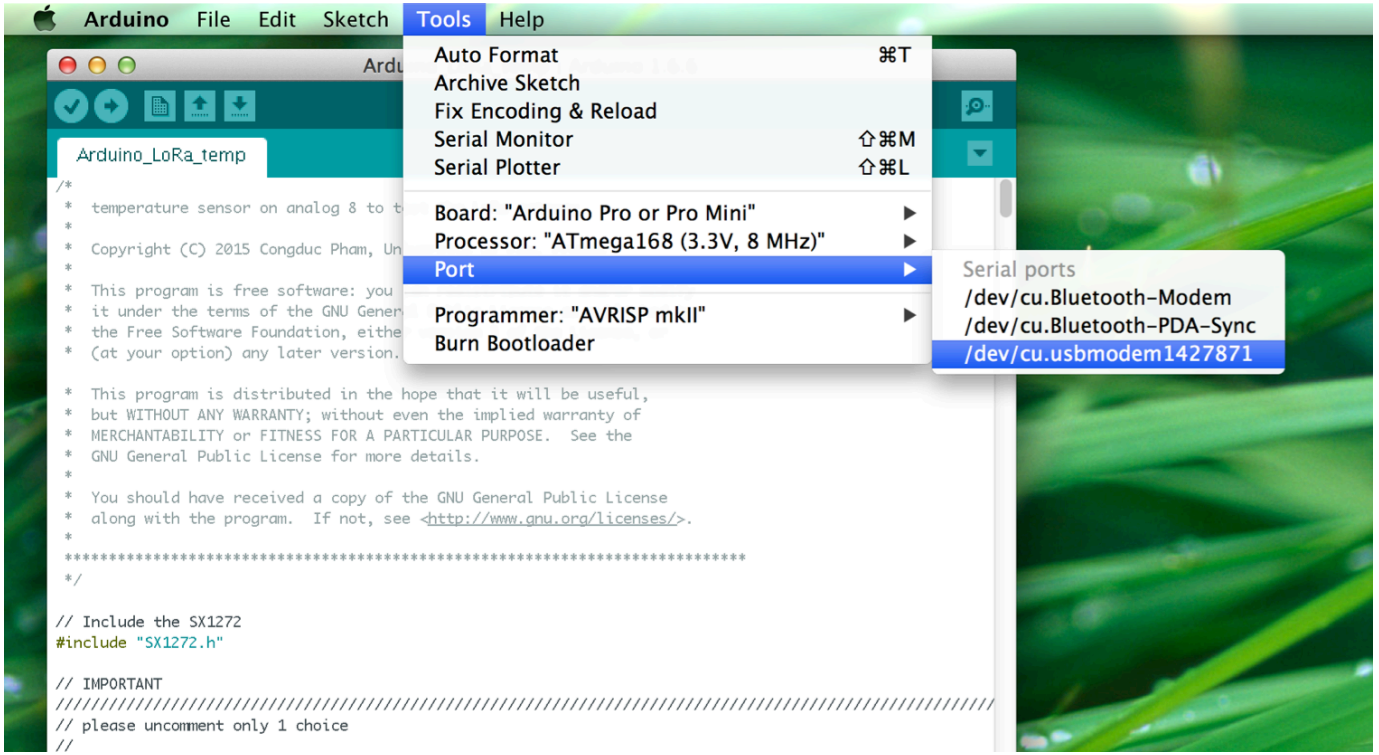



Original Sparkfun version

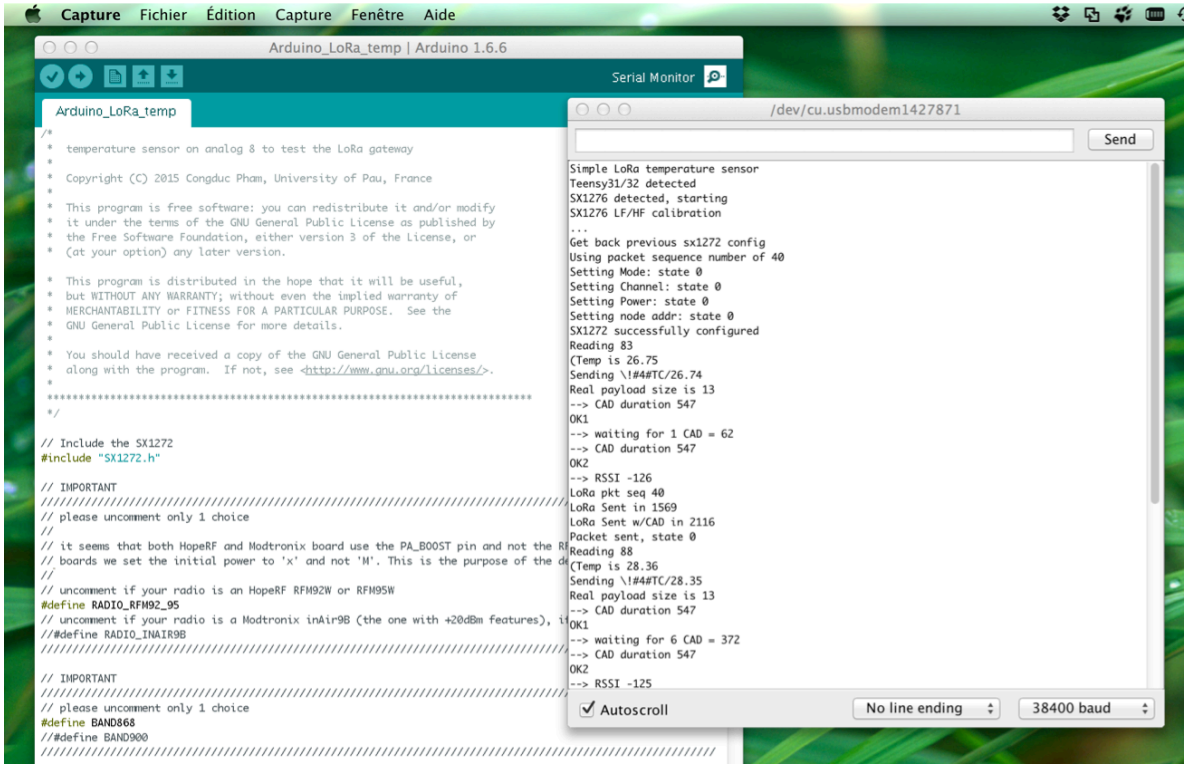
For the Pro Mini, you need to have an FTDI breakout cable working at 3.3v level (there is also 5v version but our advised Pro Mini version is running at 3.3v to reduce energy consumption). Be careful, on some low-cost Pro Mini version (Chinese manufacturer for instance) the pins may be in reversed order. The simplest way in to check the VCC pin and make it to correspond to the VCC pin of the FTDI breakout.

# Uploading (2)



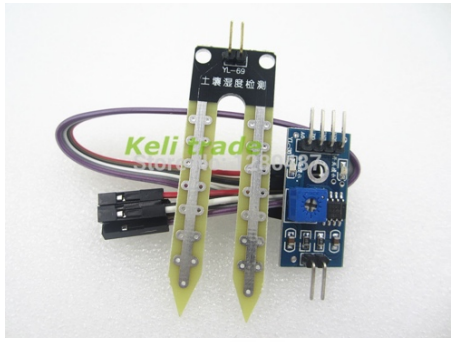

Connect the USB end to your computer and the USB port should be detected in the Arduino IDE. Select the serial port for your device. It may have another name than what is shown in the example. Then click on the « upload » button

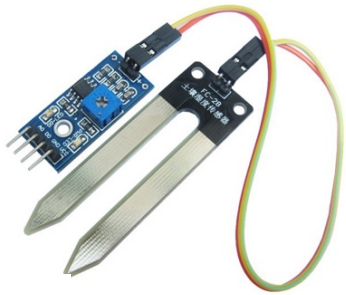
# Serial monitor



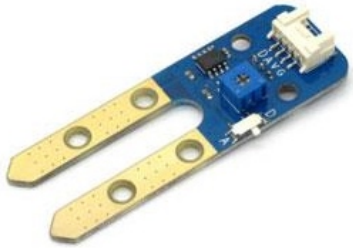

You can see the output from the sensor if it is connected to your computer. Use the Arduino IDE « serial monitor » to get such output, just to verify that the sensor is running fine, or to debug new code. Be sure to use 38400 baud.

# Adding a Soil Humidity sensor

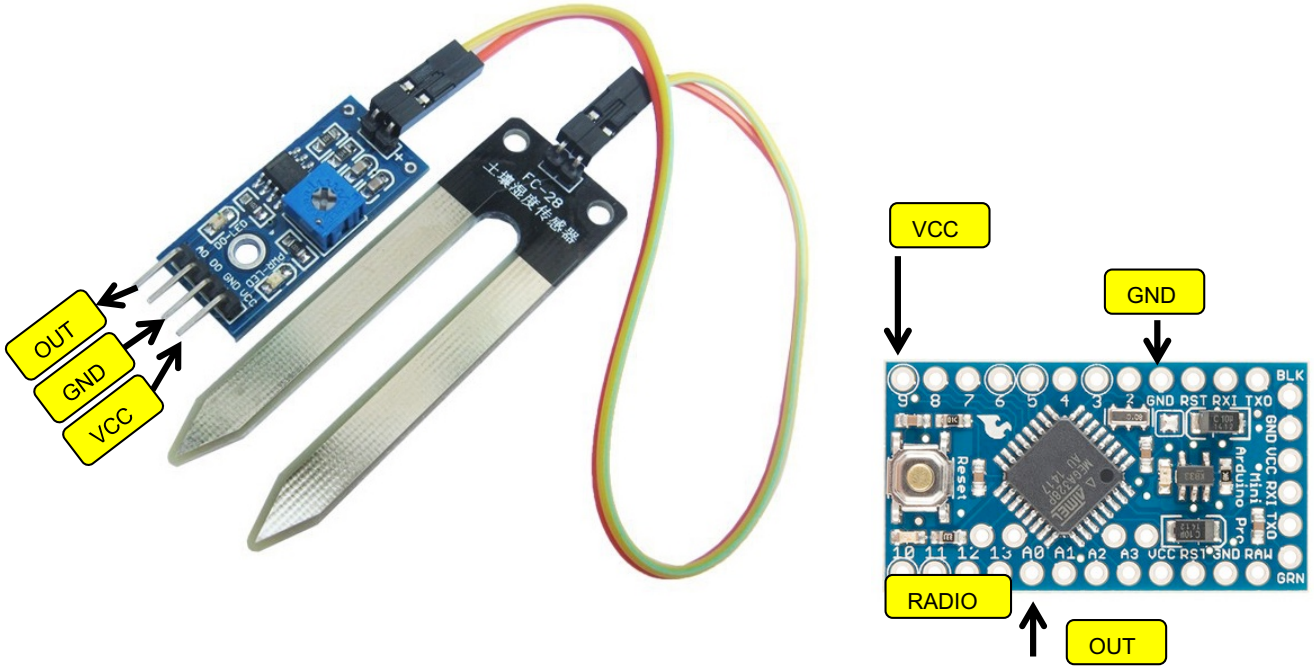




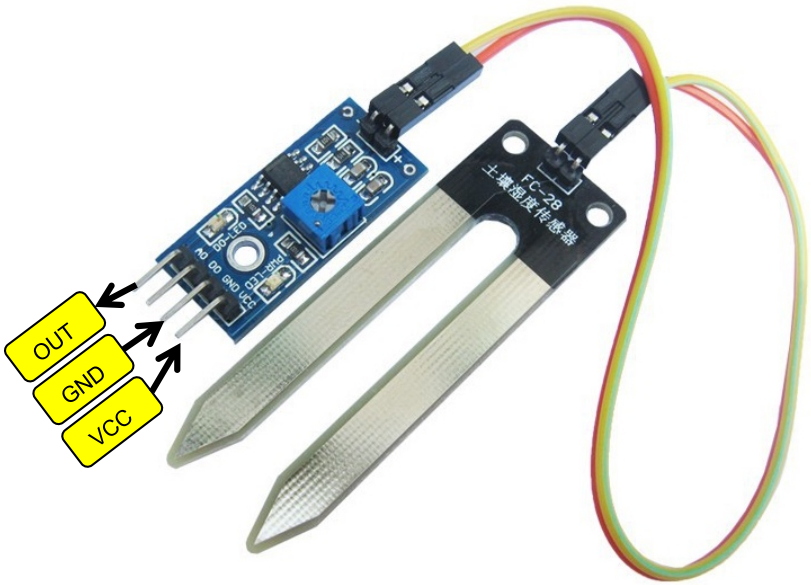


# Soil humidity sensor




For the moment, there is no physical sensor connected to the board, so you will probably get random value when running the sensor. The Arduino_LoRa_Simple_SoilHum example uses low-cost analog soil humidity sensor. The GND should be connected to one of the board's GND, the VCC should be connected to the digital pin 9 and the OUT pin should be connected to the analog A0 pin.

# Understanding analog output




Vcc is 3.3V (the output of digital 9 to power the sensor)

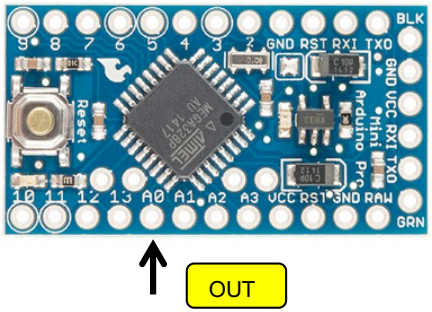
The soil humididy sensor will provide on its OUT pin a voltage between 0 and 3.3v, depending on the humidity level (based on conductivity measure)

This voltage value will be converted into a numerical value between 0 (0v) and 1023 (3.3v) as the A/D circuit is on 10 bits.

The maximum value is when there is no humidity and the minimum value is where there is a lot of humidity (after a rain)

Some calibration on soil may be needed to know the typical minimum and maximum values. For instance, minimum can be 300 (very wet) and maximum can be 1004 (very dry).

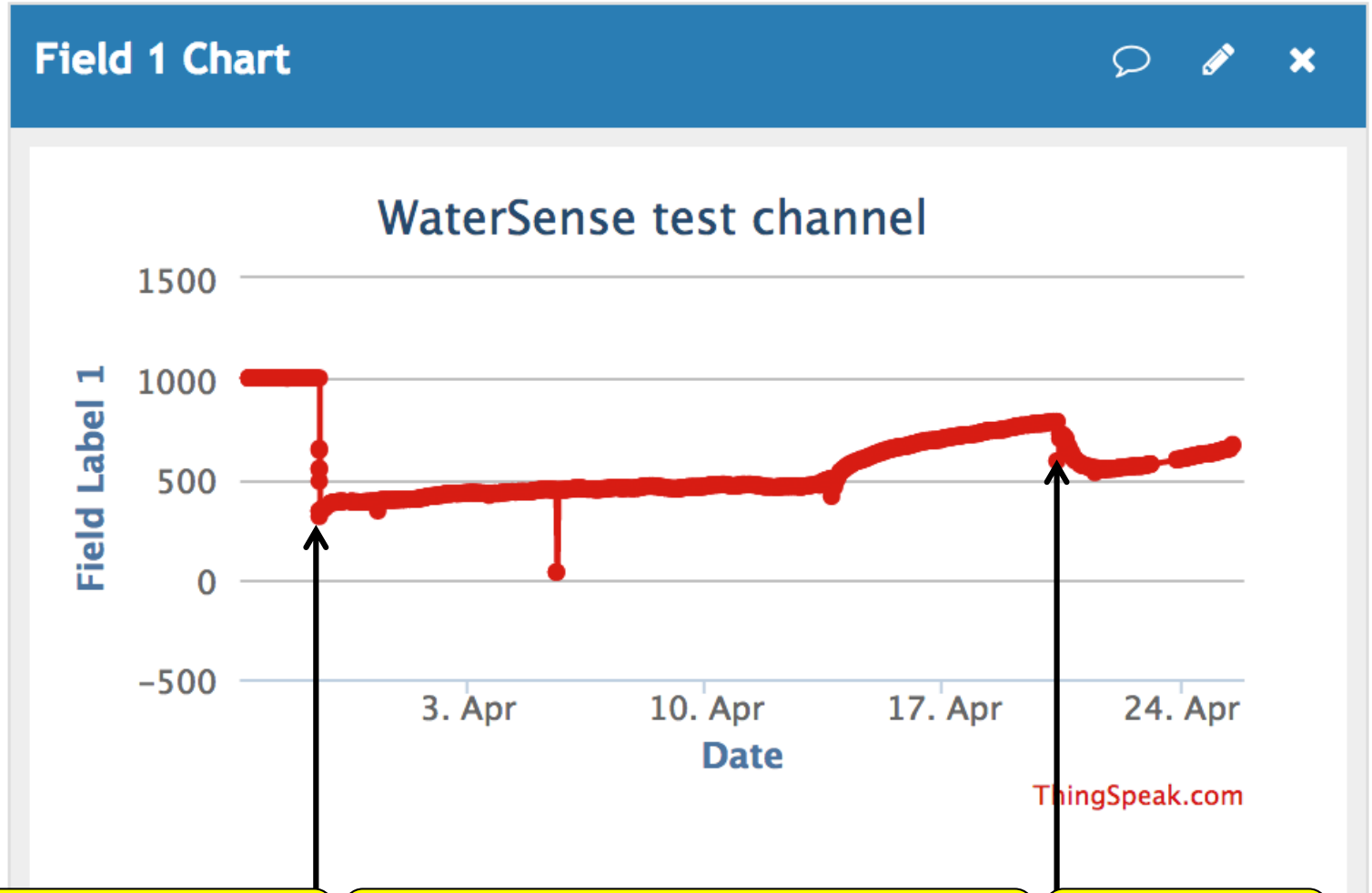
# Reading analog pin value




```
// sensor output connected to A0 analog pin

shu = analogRead(A0);

// we can provide high-level conversion here
// or rely on the end application

// now process and transmit the data
```

- 2 soil humidity sensors can be connected to the board
- Digital pins 9 & 8 will be used to power the sensors
- Analog pins A0 & A1 will be used to get the output values

# Example from ThingSpeak channel



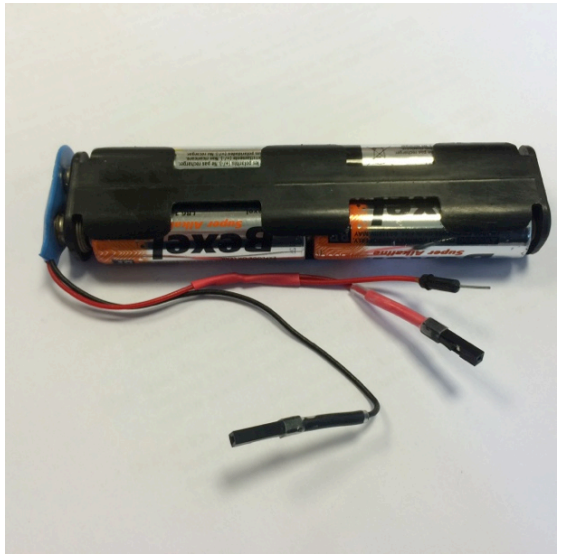

Sensor is plug into very wet soil

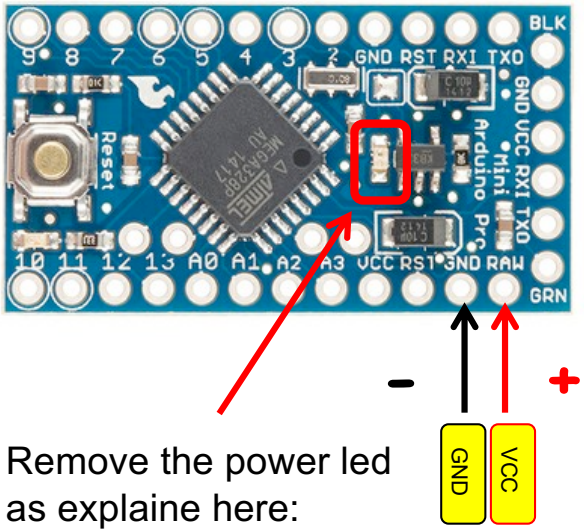
Soil getting dryer and dryer
Sensed value getting closer to max

New water added

# Running on battery & using low-power mode

# Connecting a battery






Remove the power led as explaine here:

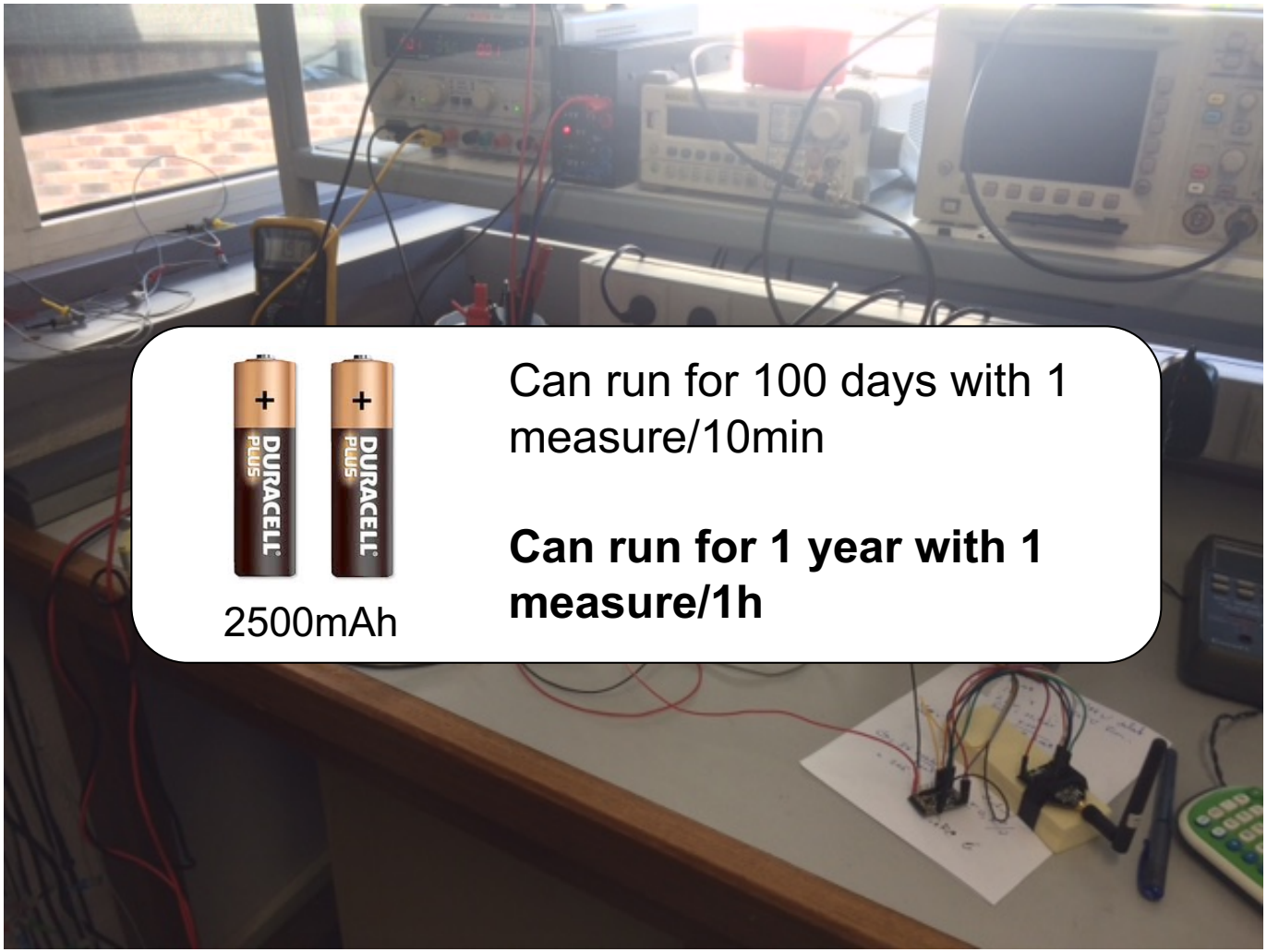
http://www.home-automation-community.com/arduino-low-power-how-to-run-atmega328p-for-a-year-on-coin-cell-battery/

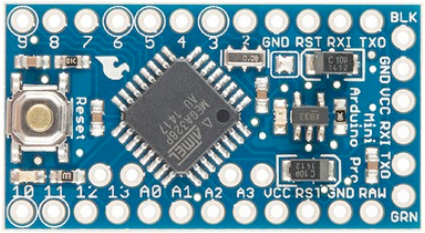
The advantage of using the Arduino Pro Mini running at 3.3v is that energy consumption can be low especially when the board is put in « sleep mode » for low-power operation. Remove the power led to further reduce consumption.

You can use 4 AA batteries to provide 4x1.5v=6v. Using only 3 batteries may lead to insufficient voltage difference.This will be injected into the RAW pin of the board, therefore using the on-board voltage regulator to get the 3.3v.

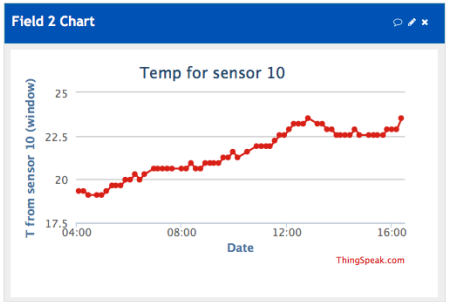
# Running for 1 year!

**Low-Power library from RocketScream**




Wakes-up every 10min, take a measure (temp) and send to GW




**120µA in deep sleep mode, 93mA when active and sending**

Thanks to T. Mesplou and P. Plouraboué for their help

# Compiling for low-power

```
//////////////////////////////////////////////////////////////////
// COMMENT OR UNCOMMENT TO CHANGE FEATURES.
// ONLY IF YOU KNOW WHAT YOU ARE DOING!!! OTHERWISE LEAVE AS IT IS
#if not defined _VARIANT_ARDUINO_DUE_X_ && not defined __SAMD21G18A__
#define WITH_EEPROM
#endif
#define WITH_APPKEY
#define LOW_POWER
#define LOW_POWER_HIBERNATE
//#define WITH_ACK
////////////////////////////////////////////////////////////////
```

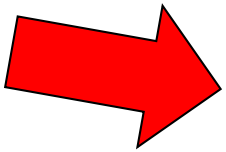
Uncomment the « #define LOW_POWER » statement, compile and upload

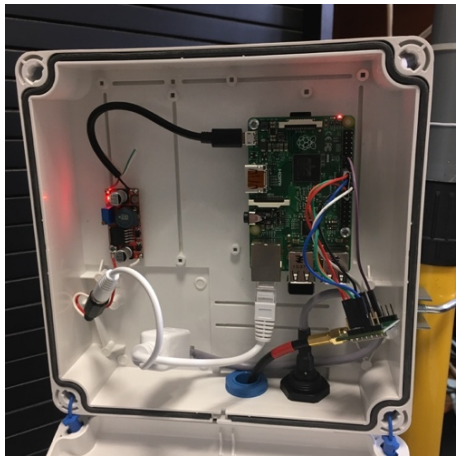
```
//////////////////////////////////////////////////////////////////
// CHANGE HERE THE TIME IN MINUTES BETWEEN 2 READING & TRANSMISSION
unsigned int idlePeriodInMin = 60;
//////////////////////////////////////////////////////////////////
```

Change the value of idlePeriodInMin to X minutes if needed

# Sending to a gateway: Default configuration

\!SM1/567/SM2/678



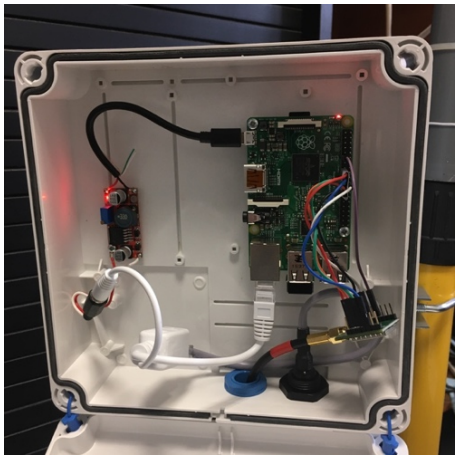


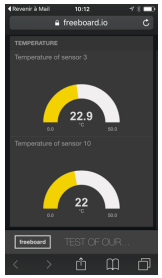
```
#define DEFAULT_DEST_ADDR 1
#define LORAMODE  1
#define node_addr 2
```

The default configuration in the Arduino_LoRa_Simple_SoilHum example is:

Send packets to the gateway (one or many if in range)
LoRa mode 1, 433.3MHz
Node short address is 2

# Gateway to cloud

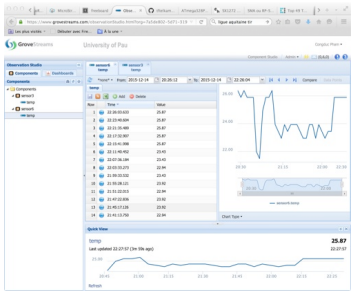


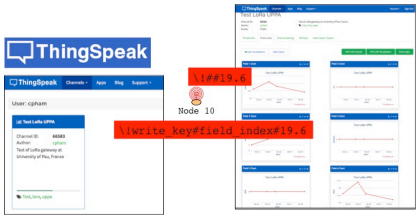



Dropbox


freeboard

FIWARE






GroveStreams

SensorCloud™

Data received at the gateway will be pushed to IoT clouds. We provide python script examples for many IoT cloud platforms. Most of clouds with REST API can be easily integrated.

**Now, have a look at the « Low-cost LoRa WaterSense gateway: a step-by-step tutorial »**