



# LOW-COST LORA IOT DEVICE: A STEP-BY-STEP TUTORIAL

RESSACS'2016  
IRD, BONDY

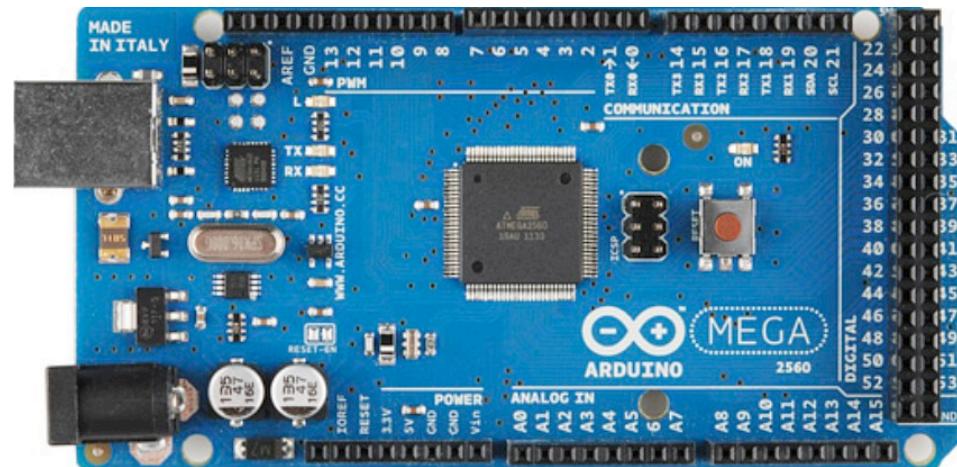
MAY 10 TH, 2016



PROF. CONG DUC PHAM  
[HTTP://WWW.UNIV-PAU.FR/~CPHAM](http://www.univ-pau.fr/~cpham)  
UNIVERSITÉ DE PAU, FRANCE

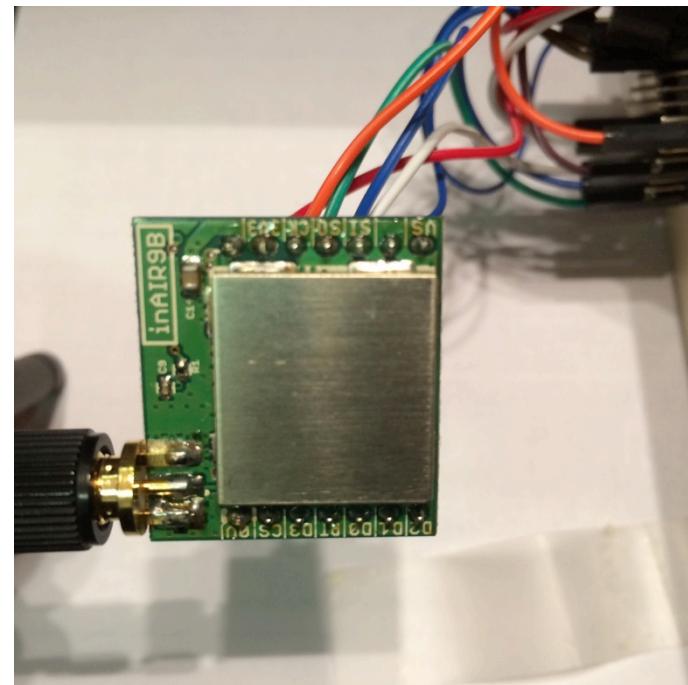
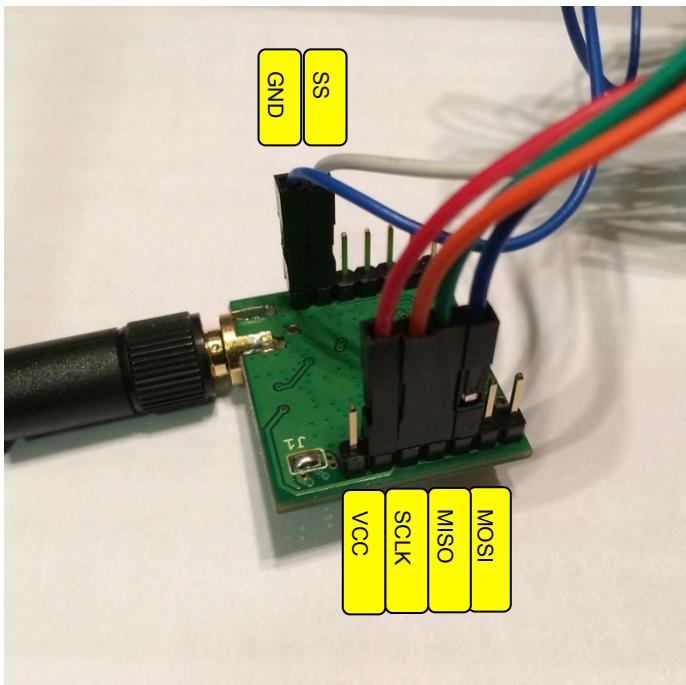


# PREPARE THE BOARD



You either have an Arduino UNO or an Arduino MEGA2650

# NOW THE RADIO MODULE: MODTRONIX inAIR9

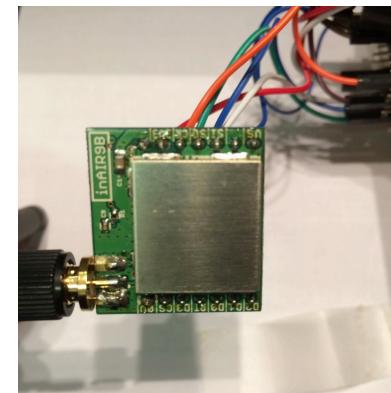
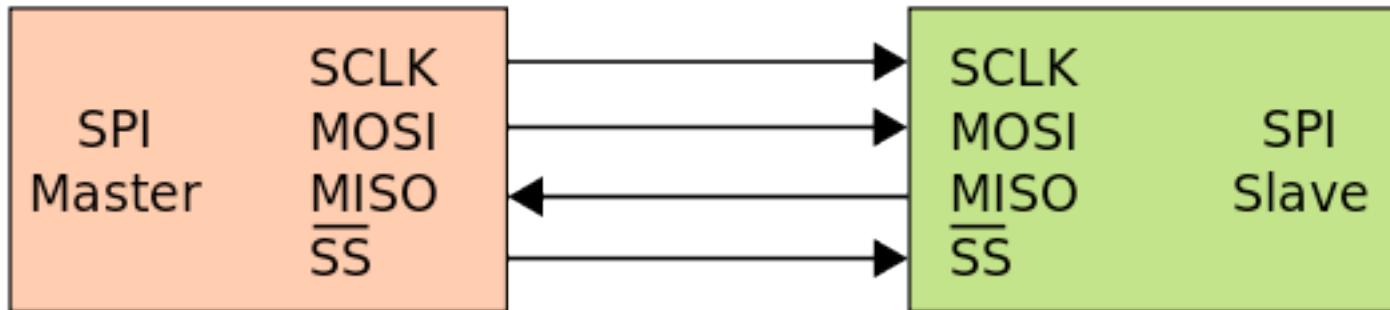


Connect the SPI pins (MOSI, MISO, SCLK) with F/F wires and the SS pin with a F/M wire. Try to use different colors. I use the following colors: MOSI (blue), MISO (green), SS (white), SCLK (orange). Then connect also the radio VCC (red) with a F/M wire and the radio GND (black or any other dark color) also with a F/M wire.

# SERIAL PERIPHERAL INTERFACE (SPI)



WIKIPEDIA  
The Free Encyclopedia



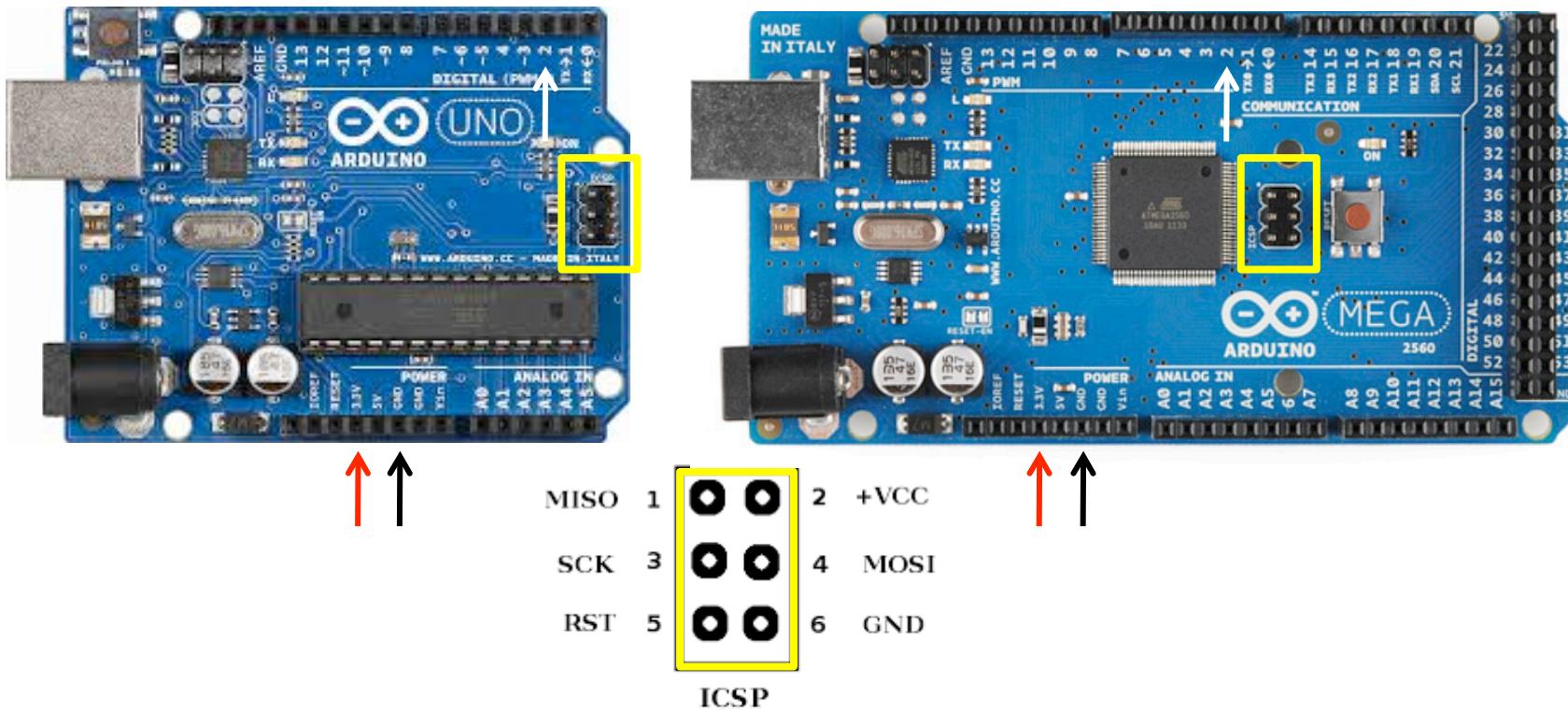
The SPI bus specifies four logic signals:



WIKIPEDIA  
The Free Encyclopedia

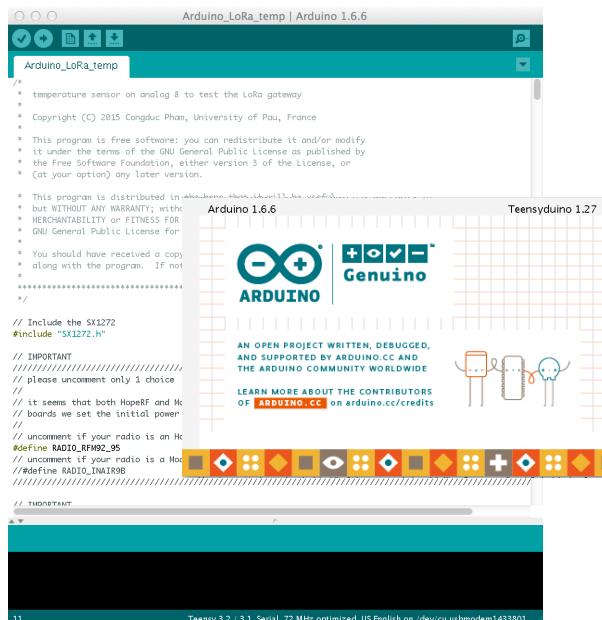
- SCLK : Serial Clock (output from master).
- MOSI : Master Output, Slave Input (output from master).
- MISO : Master Input, Slave Output (output from slave).
- SS : Slave Select (**active low**, output from master).

# CONNECTING THE RADIO MODULE



Now, just connect the corresponding SPI pins of the radio module to the SPI pins on the board which are on the ICSP header. SS goes to digital 2. Then connect also the VCC (red) and the GND (black) of the radio board to the 3.3v and the GND of the board. The 3.3v pin of the board gets 3.3v from the on-board voltage regulator.

# GETTING, COMPIILING & UPLOADING THE SOFTWARE





# GETTING THE SOFTWARE: A SIMPLE SENDER

```
Arduino_LoRa_temp | Arduino 1.6.6
/*
 * temperature sensor on analog 0 to test the LoRa gateway
 *
 * Copyright (C) 2015 Congduc Pham, University of Pau, France
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in
 * but WITHOUT ANY WARRANTY; without
 * MERCHANTABILITY or FITNESS FOR
 * GNU General Public License for
 *
 * You should have received a copy
 * along with the program. If not
 */
*****  

// Include the SX1272
#include "SX1272.h"  

//  

// IMPORTANT
// please uncomment only 1 choice
// it seems that both HopeRF and M
// boards we set the initial power
// uncomment if your radio is an M
#define RADIO_RFH22_95
// uncomment if your radio is a M
#define RADIO_INA198
//  

// TUDOSTANT  

11  Teensy 3.2 / 3.1, Serial, 72 MHz optimized, US English on /dev/cu.usbmodem143301
```

CongducPham / LowCostLoRaGw

Code Issues 6 Pull requests 0 Pulse Graphs

Low-cost LoRa gateway with SX1272 and Raspberry

11 commits 1 branch 0 releases 0 contributors

Branch: master New pull request New file Find file HTTPS https://github.com/Congdu... Download ZIP

File	Commit Message	Date
Arduino	modified some low-power info	10 days ago
Raspberry	modified some low-power info	10 days ago
.DS_Store	changes in the SX1272 lib, gateway and temperature example	2 months ago
README.md	Congduc Pham added a simpler version of temperature sensor	10 days ago
..		
Arduino_LoRa_Gateway	added a simpler version of temperature sensor	
Arduino_LoRa_Simple_temp	added a simpler version of temperature sensor	
Arduino_LoRa_temp	added a simpler version of temperature sensor	
libraries/SX1272	Added Teensy support	

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

Get into the Arduino folder and get both Arduino\_LoRa\_Gateway and SX1272 folders. Copy Arduino\_LoRa\_Gateway into your “sketch” folder and SX1272 into “sketch/libraries”

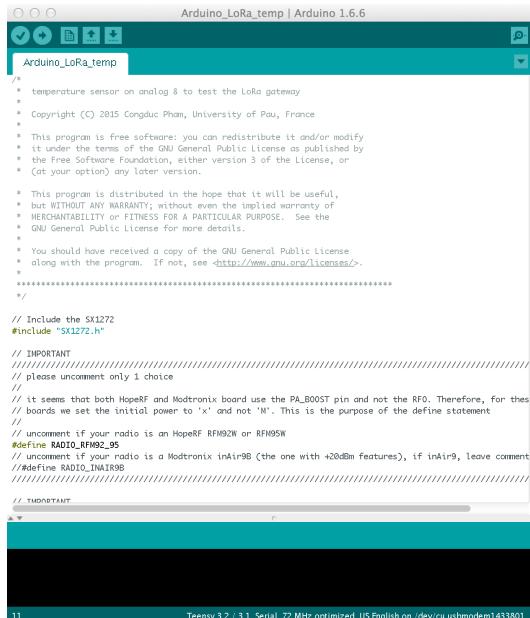
# MAKE IT A SENDER

---

```
//#define RECEIVE_ALL
//#define IS_RCV_GATEWAY
#define IS_SEND_GATEWAY
//#define CAD_TEST
//#define LORA_LAS
//#define WINPUT
//#define WITH_SEND_LED
```

Uncomment the statement #define IS\_SEND\_GATEWAY

# COMPILING



```

/*
 * temperature sensor on analog 8 to test the LoRa gateway
 *
 * Copyright (C) 2015 Congduc Pham, University of Pau, France
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with the program. If not, see <http://www.gnu.org/licenses/>.
 */
*****  

// Include the SX1272
#include "SX1272.h"  

// IMPORTANT  

// please uncomment only 1 choice  

//  

// it seems that both HopeRF and Madtronix board use the PA_BOOST pin and not the RFO. Therefore, for these  

// boards we set the initial power to 'x' and not 'N'. This is the purpose of the define statement  

//  

// uncomment if your radio is an HopeRF RFM92 or RFM95
#define RADIO_RF92_95
// uncomment if your radio is a Madtronix inA9R9B (the one with +20dBm features), if inA9R9, leave comment
#define RADIO_INA9RB  

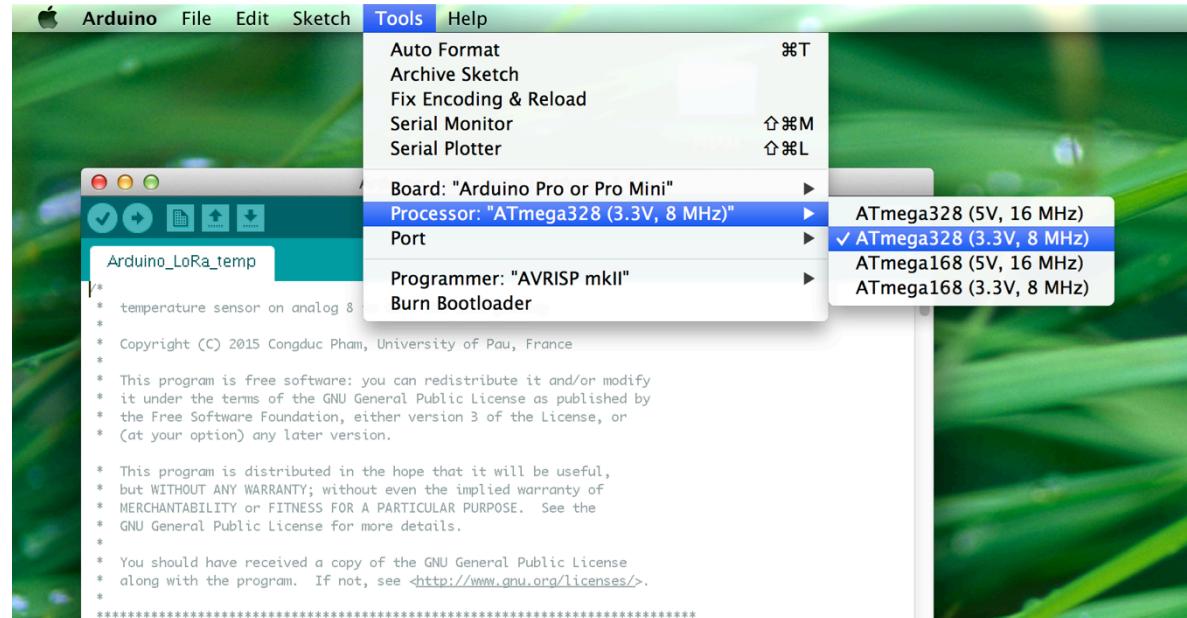
  

// TUDOSTANT
  

11   Teensy 3.2 / 3.1, Serial, 72 MHz optimized, US English on /dev/cu.usbmodem143301

```

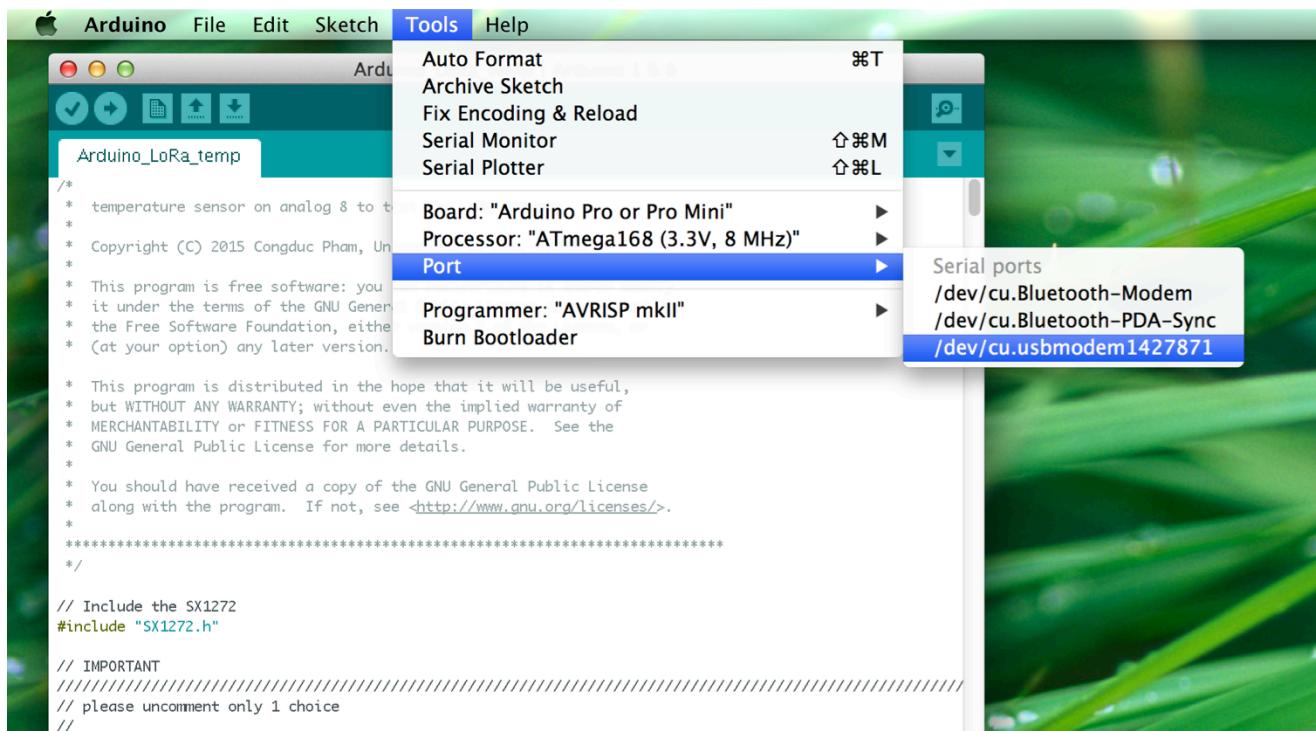


Open the Arduino\_LoRa\_Gateway sketch and select the Arduino UNO or MEGA 2560 depending on the board you have

Then, click on the « verify » button



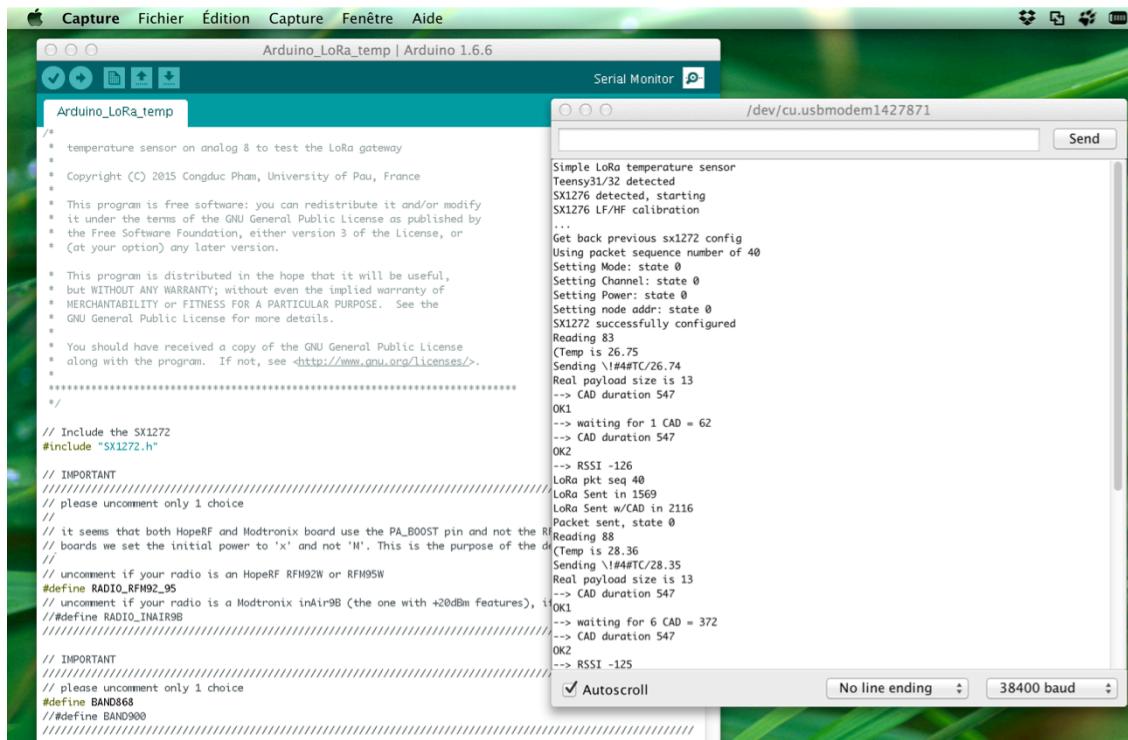
# UPLOADING



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

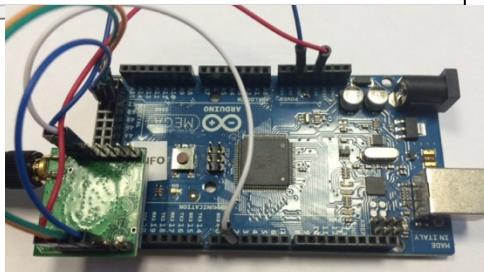


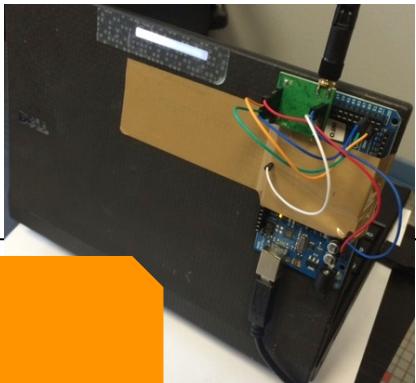
You can see the output from the sensor if it is connected to your computer. Use the Arduino IDE « serial monitor » to get such output, just to verify that the sensor is running fine, or to debug new code. Be sure to use 38400 baud.

# USE THE INTERACTIVE SENDER

## □ Interactive end-device

```
Hello world
/dev/cu.usbmodemFA131 (Arduino/Genuino N
6477 bytes of free memory.
SX1276 detected, starting
SX1276 LF/HF calibration
...
@$*****Power ON: state 0
@$Default sync word: 0x12
$LoRa mode 4
$Setting mode: state 0
$Channel CH_10_868: state 0
$Set LoRa Power to x
$Power: state 0
$Get Preamble Length: state 0
$Preamble Length: 8
$LoRa addr 6: state 0
$SX1272/76 configured as device. Waiting serial input for serial-RF bridge
Rcv serial: hello world
Sending. Length is 11
hello world
Payload size is 11
ToA is w/5B Libelium header 322
Packet number 0
LoRa Sent in 545
LoRa Sent w/CAD in 545
Packet sent, state 0
```





Enter **hello world**  
and press ENTER

Command	Action
/@M1#	set LoRa mode 1
/@C12#	use channel 12
/@PL/H/M/x/X#	set power to Low, High, Max, extreme (PA_BOOST), eXtreme (+20dBm)
/@A9#	set node addr to 9
/@ACK#hello w/ack	sends "hello w/ack" and request an ACK
/@ACKON#	enables ACK (for all messages)
/@ACKOFF#	disables ACK
/@CAD#	performs an SIFS CAD, i.e. 3 or 6 CAD depending on the LoRa mode
/@CADON3#	uses 3 CAD when sending data (normally SIFS is 3 or 6 CAD, DIFS=3SIFS)
/@CADOFF#	disables CAD (IFS) when sending data
/@RSSI#	toggles checking of RSSI before transmission and after CAD
/@EIFS#	toggles for extended IFS wait
/@T5000#	send a message at regular time interval of 5000ms. Use /@T0# to disable periodic sending
/@TR5000#	send a message at random time interval between [2000, 5000]ms.
/@Z200#	sets the packet payload size to 200 for periodic sending
/@S50#	sends a 50B user payload packet filled with '#'. The real size is 55B with the protocol header
/@D56#	set the destination node to be 56, this is permanent, until the next D command
/@D56#hello	send "hello" to node 56, destination addr is only for this message
/@D1#/@M1#	send the command string "/@M1#" to node 1 (i.e. gateway)

/ @ACK#hello



You can get  
SNR of both  
uplink and  
downlink





# GETTING THE SOFTWARE: A

## TEMPERATURE SENSOR

```
Arduino_LoRa_temp | Arduino 1.6.6
/*
 * temperature sensor on analog 0 to test the LoRa gateway
 *
 * Copyright (C) 2015 Congduc Pham, University of Pau, France
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in
 * but WITHOUT ANY WARRANTY; without
 * MERCHANTABILITY or FITNESS FOR
 * GNU General Public License for
 *
 * You should have received a copy
 * along with the program. If not
 */
*****  

// Include the SX1272
#include "SX1272.h"  

//  

// IMPORTANT  

// please uncomment only 1 choice  

// it seems that both HopeRF and M  

// boards we set the initial power  

// uncomment if your radio is an M
#define RADIO_RFM92_95
// uncomment if your radio is a M
#define RADIO_INA109B
//  

// *****  

11  Teensy 3.2 / 3.1, Serial, 72 MHz optimized, US English on /dev/cu.usbmodem143301
```

CongducPham / LowCostLoRaGw

Code Issues 6 Pull requests 0 Pulse Graphs

Low-cost LoRa gateway with SX1272 and Raspberry

11 commits 1 branch 0 releases 0 contributors

Branch: master New pull request New file Find file HTTPS https://github.com/Congdu... Download ZIP

Congduc Pham modified some low-power info Latest commit a46b0f7 10 days ago

Arduino modified some low-power info 10 days ago

Raspberry modified some low-power info 10 days ago

.DS\_Store changes in the SX1272 lib, gateway and temperature example 2 months ago

README.md

Congduc Pham added a simpler version of temperature sensor 10 days ago

..

Arduino\_LoRa\_Gateway added a simpler version of temperature sensor

Arduino\_LoRa\_Simple\_temp added a simpler version of temperature sensor

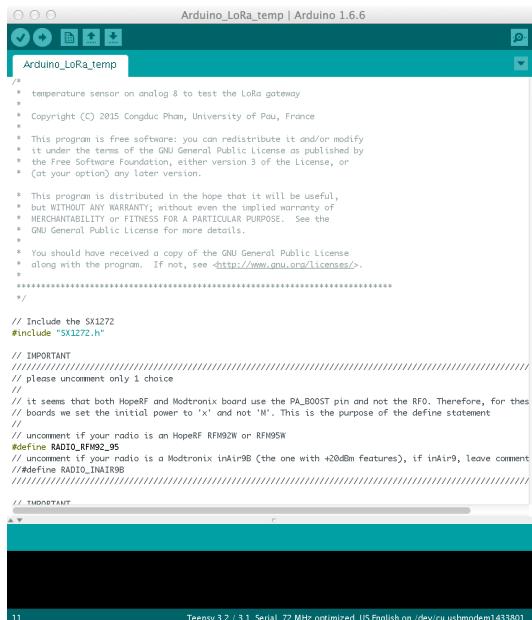
Arduino\_LoRa\_temp added a simpler version of temperature sensor

libraries/SX1272 Added Teensy support

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

Get into the Arduino folder and get `Arduino_LoRa_Simple_temp`. Copy `Arduino_LoRa_Simple_temp` into your “sketch” folder. Compile then Upload as previously.

# COMPILING



```

/*
 * temperature sensor on analog 8 to test the LoRa gateway
 *
 * Copyright (C) 2015 Congduc Pham, University of Pau, France
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with the program. If not, see <http://www.gnu.org/licenses/>.
 */
*****  

// Include the SX1272
#include "SX1272.h"  

// IMPORTANT  

// please uncomment only 1 choice  

//  

// it seems that both HopeRF and Madtronix board use the PA_BOOST pin and not the RFO. Therefore, for these  

// boards we set the initial power to 'x' and not 'N'. This is the purpose of the define statement  

//  

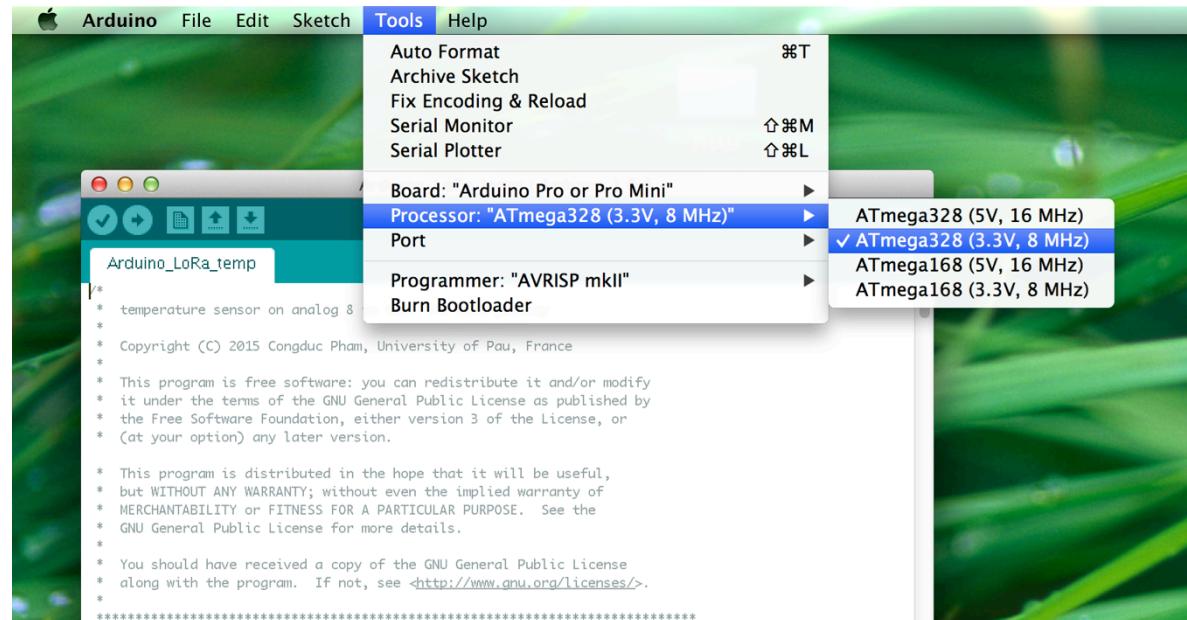
// uncomment if your radio is an HopeRF RFM92 or RFM95
#define RADIO_RF92_95
// uncomment if your radio is a Madtronix inA9R9B (the one with +20dBm features), if inA9R9, leave comment
#define RADIO_INA9RB
//  

// TUDOSTANT
  

11

```

Teensy 3.2 / 3.1, Serial, 72 MHz optimized, US English on /dev/cu.usbmodem143301



Open the Arduino\_LoRa\_Simple\_temp sketch and select the Arduino UNO or MEGA 2560 depending on the board you have

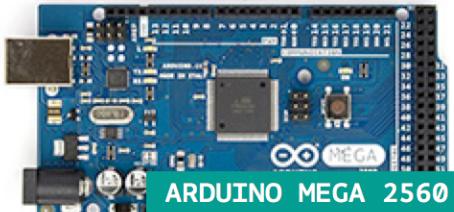
Then, click on the « verify » button



## MORE PLATFORMS ARE SUPPORTED!



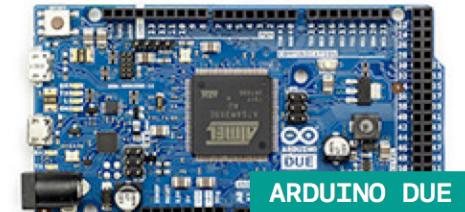
ARDUINO UNO



ARDUINO MEGA 2560



ARDUINO ZERO



ARDUINO DUE



ARDUINO MICRO



ARDUINO PRO MINI



ARDUINO NANO



Ideetron Nexus



Teensy3.1/3.2



LoRa radios that  
our library already  
supports



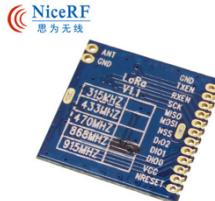
HopeRF  
RFM92W/95W



Libelium LoRa



Modtronix  
inAir9/9B



NiceRF  
LoRa1276

Long-Range communication library  
(mostly sending functions)



LoRa radios that  
our library already  
supports



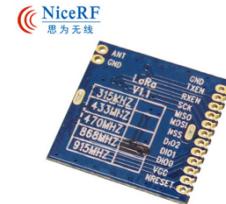
HopeRF  
RFM92W/95W



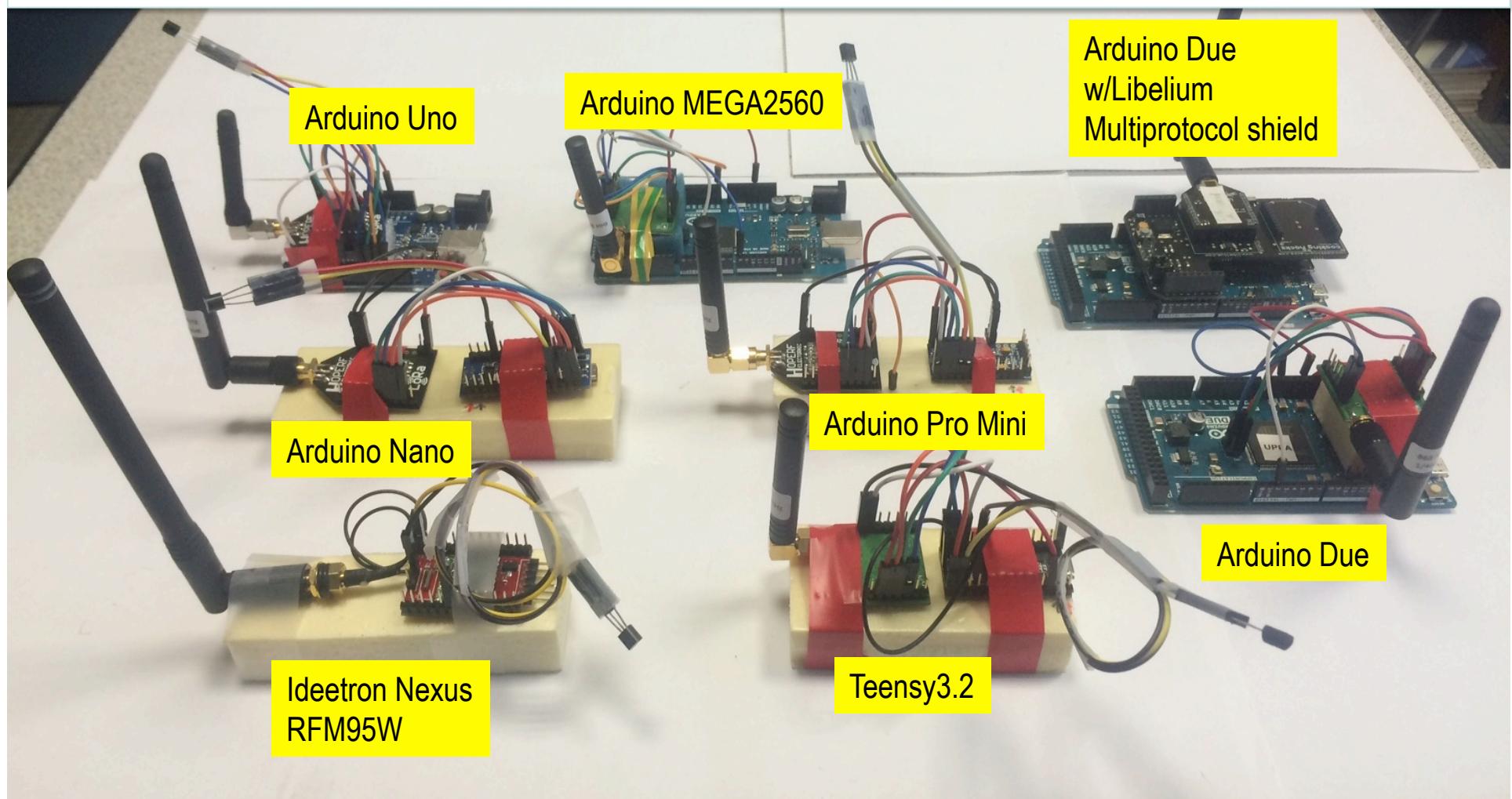
Libelium LoRa



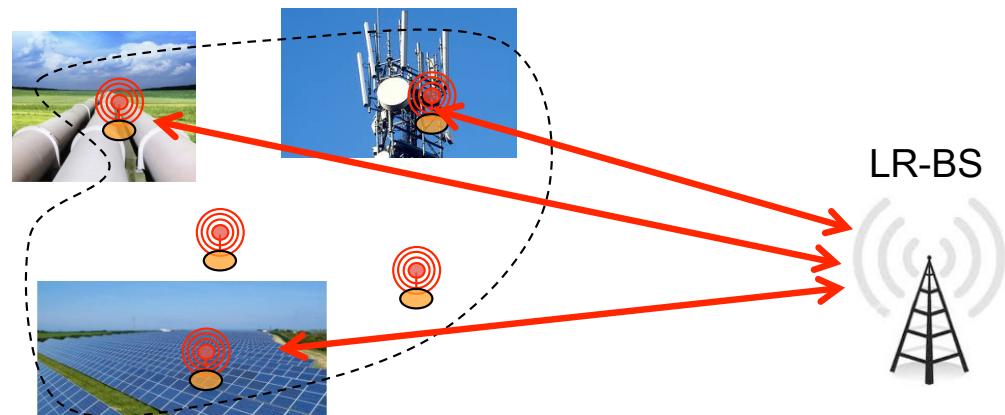
Modtronix  
inAir9/9B



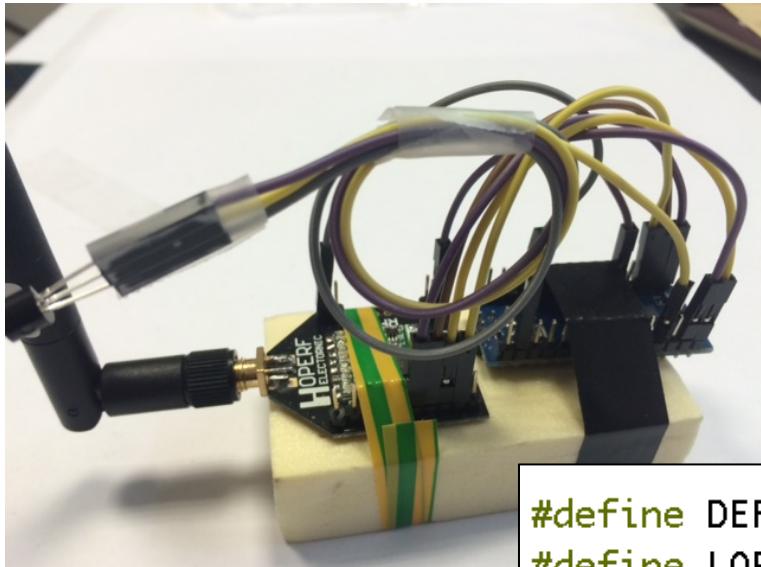
LoRa1276  
NiceRF  
LoRa1276



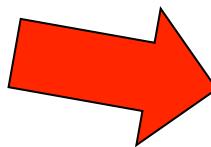
## SENDING TO A GATEWAY



# DEFAULT CONFIGURATION



\!#4#18.5

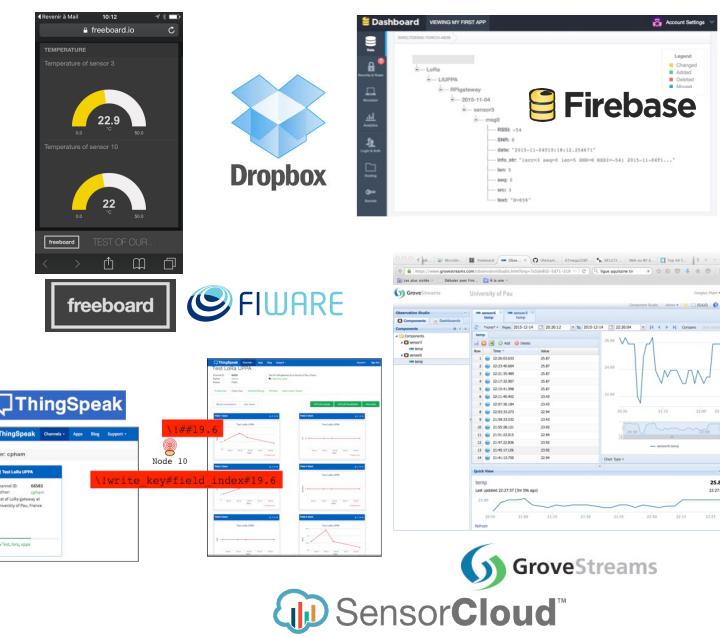
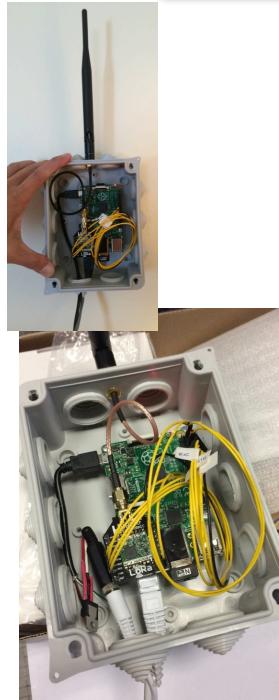


```
#define DEFAULT_DEST_ADDR 1
#define LORAMODE 1
#define node_addr 8
```

The default configuration in the Arduino\_LoRa\_Simple\_temp example is:

Send packets to the gateway (one or many if in range)  
LoRa mode 1  
Node short address is 8

# GATEWAY TO CLOUD

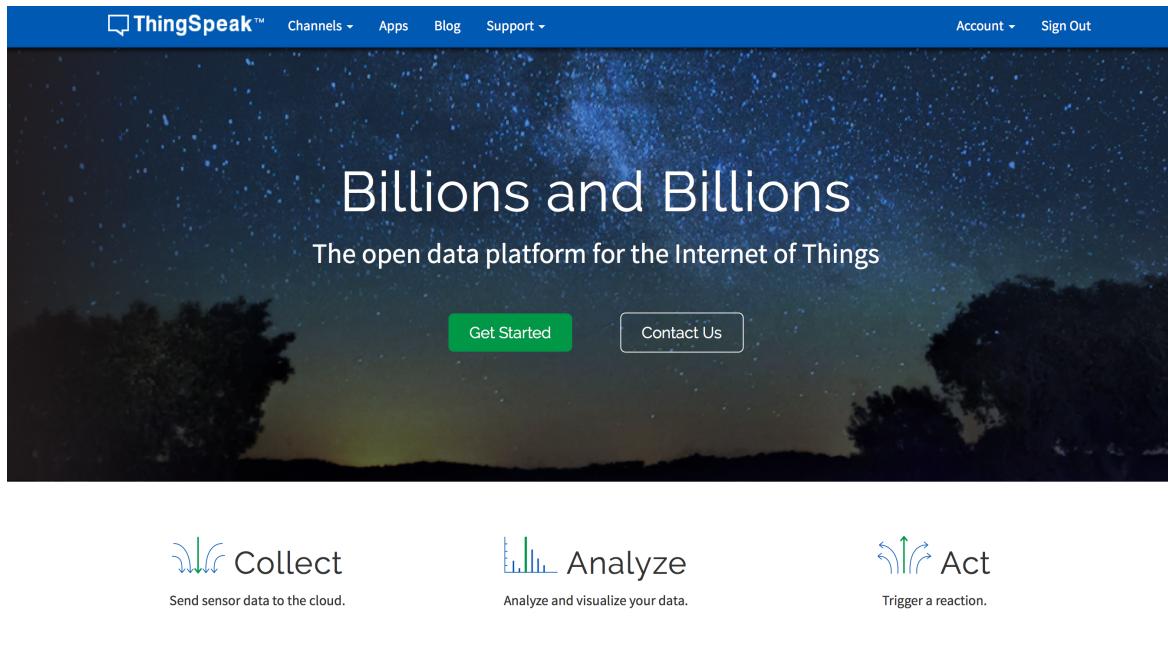


Data received at the gateway will be pushed to IoT clouds. We provide python script examples for many IoT cloud platforms. Most of clouds with REST API can be easily integrated.

In this example, data will be pushed to the ThingSpeak cloud on field index 4.



# USE YOUR OWN CLOUD



The screenshot shows the ThingSpeak homepage. At the top, there is a navigation bar with links for "ThingSpeak™", "Channels", "Apps", "Blog", "Support", "Account", and "Sign Out". The main header features the text "Billions and Billions" and "The open data platform for the Internet of Things" over a background image of a starry night sky and silhouetted trees. Below the header are two buttons: "Get Started" (in green) and "Contact Us". At the bottom of the page, there are three sections: "Collect" (with a cloud icon), "Analyze" (with a bar chart icon), and "Act" (with a gear and arrow icon). Each section has a brief description: "Send sensor data to the cloud.", "Analyze and visualize your data.", and "Trigger a reaction." respectively.

Go to <https://thingspeak.com/>

Create your own ThingSpeak account (Sign Up)

And create a new channel, called it **ressac2016\_lora\_test** for instance



# GET THE API KEY OF YOUR CHANNEL

## Test LoRa Gateway

Channel ID: 66794  
Author: cpham  
Access: Public

Public Channel for Testing yo  
🏷: Test, lora, gateway

Private View    Public View    Channel Settings    API Keys    Data Import /

## Write API Key

Key

SGSH52UGPVAUYG3S

Generate New Write API Key

## Read API Keys

Key

50MG8DESB3PTUX20

The write key is the key that will allow you to write data to your ThingSpeak channel

# CHANGE THE SENSOR TO SEND TO YOUR CHANNEL

```
#ifdef FLOAT_TEMP
    ftoa(float_str,temp,2);

#ifndef NEW_DATA_FIELD
    r_size=sprintf((char*)message+app_key_offset, "\\!#%d#TC/%s", field_index, float_str);
#else
    // this is for testing, uncomment if you just want to test, without a real temp sensor plugged
    //strcpy(float_str, "21.55567");
    r_size=sprintf((char*)message+app_key_offset, "\\!#%d#%s", field_index, float_str);
#endif
```

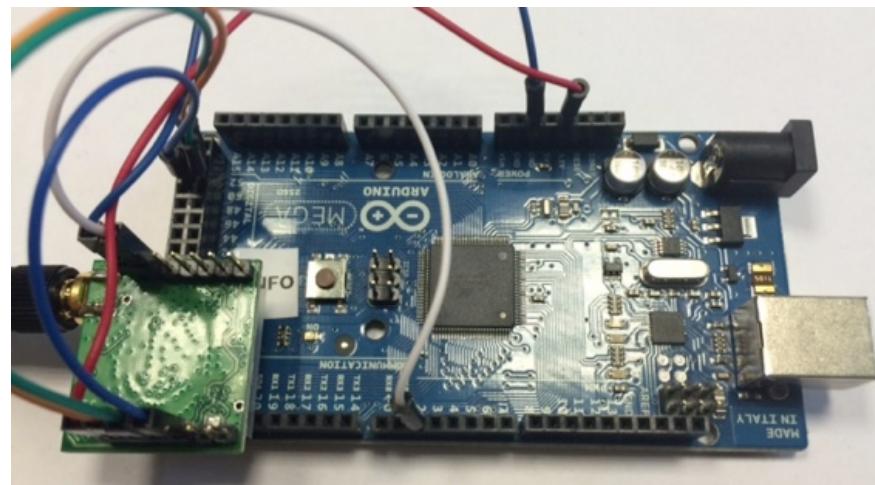
Get your ThingSpeak channel's API write key: i.e. SGSH52UGPVAUYG3S

Change "\\!#%d#TC/%s » into "\\!SGSH52UGPVAUYG3S#%d#TC/%s"

Compile and upload.

# USING THE INTERACTIVE SENDER

```
/dev/cu.usbmodemFA131 (Arduino/Genuino M  
  
6477 bytes of free memory.  
SX1276 detected, starting  
SX1276 LF/HF calibration  
...  
^$*****Power ON: state 0  
^$Default sync word: 0x12  
^$LoRa mode 4  
^$Setting mode: state 0  
^$Channel CH_10_868: state 0  
^$Set LoRa Power to x  
^$Power: state 0  
^$Get Preamble Length: state 0  
^$Preamble Length: 8  
^$LoRa addr 6: state 0  
^$SX1272/76 configured as device. Waiting serial input for serial-RF bridge  
Rcv serial: hello world  
Sending. Length is 11  
hello world  
Payload size is 11  
ToA is w/5B Libelium header 322  
Packet number 0  
LoRa Sent in 545  
LoRa Sent w/CAD in 545  
Packet sent, state 0
```



Assume your API write key is SGSH52UGPVAUYG3S

Enter \!SGSH52UGPVAUYG3S#1#TC/20.5

Try field value from 1 to 8. Try other value: i.e. 31.5