Title

*LoRa communications*.

Hardware Required

* 2 Arduino or Genuino Board
* 2 breadboard
* 2 Module LoRa
* 1 LED
* 1 Resistor
* 1 TMP

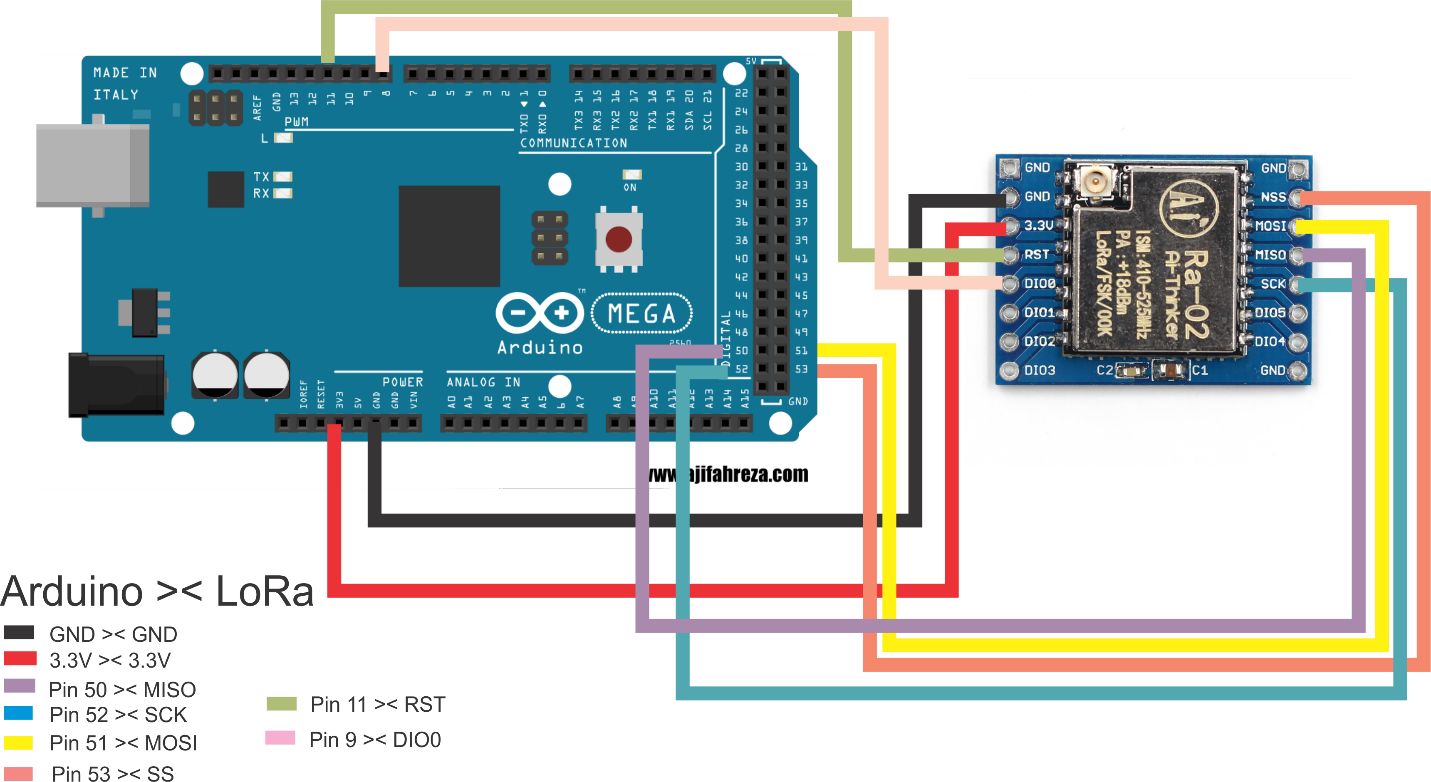
Theory

*-2 stations, each station = Arduino + LoRa module.*

*- Connect 2 stations*

*- Display value of sensor received from sensor of another station.*

Circuit



Code

Station1

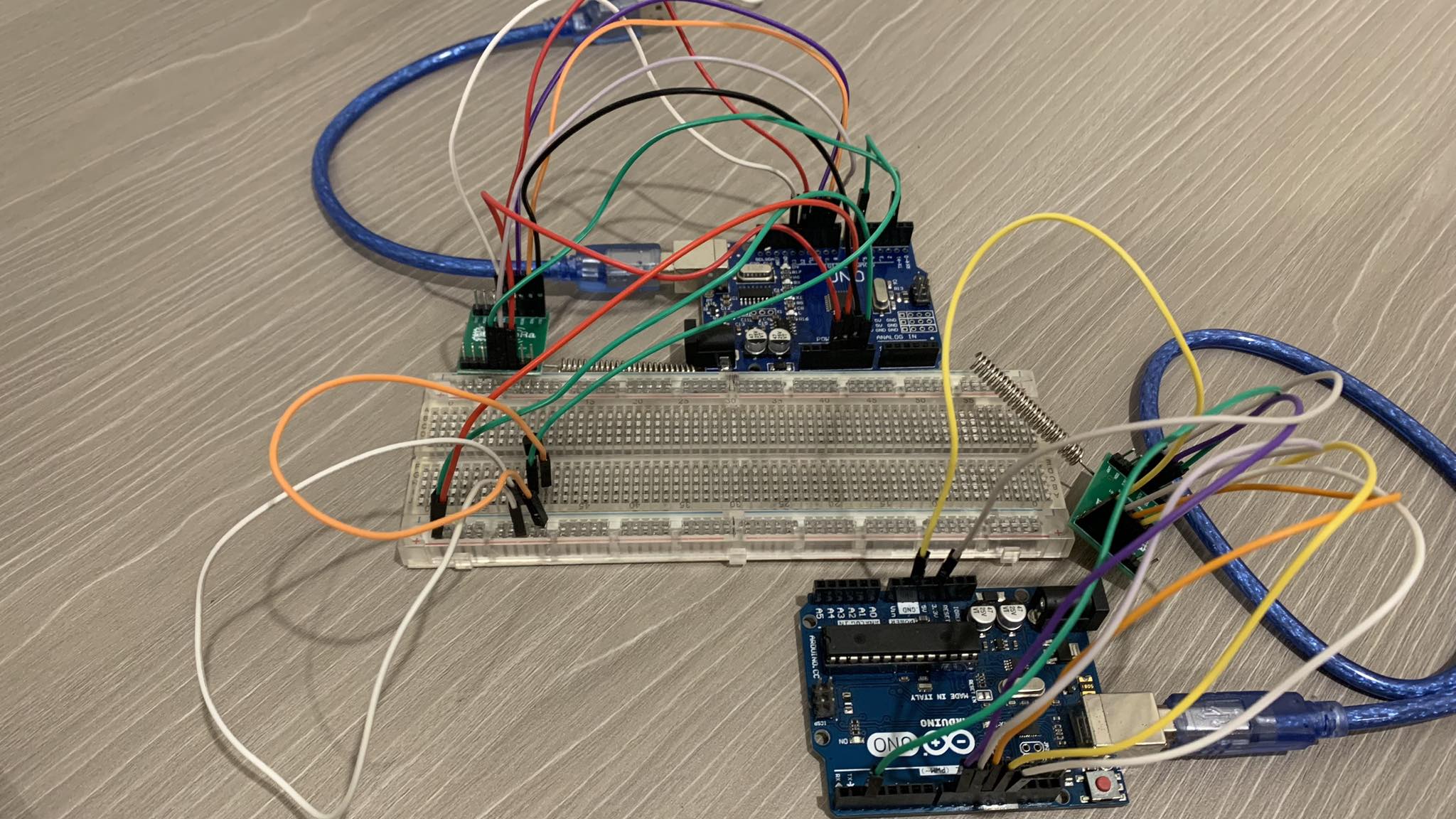
|  |
| --- |
| #include <SPI.h> // include libraries  #include <LoRa.h>  //  //const int csPin = 7; // LoRa radio chip select  //const int resetPin = 6; // LoRa radio reset  //const int irqPin = 1; // change for your board; must be a hardware interrupt pin  #define NSS\_PIN 10  #define RESET\_PIN 9  #define DIO0\_PIN 2  #define tempSensor 6  String outgoing; // outgoing message  byte msgCount = 0; // count of outgoing messages  byte localAddress = 0xBB; // address of this device  byte destination = 0xFF; // destination to send to  long lastSendTime = 0; // last send time  int interval = 2000; // interval between sends  int countPacket = 0;  void setup() {  LoRa.setPins(NSS\_PIN, RESET\_PIN, DIO0\_PIN);  Serial.begin(9600); // initialize serial  while (!Serial);  Serial.println("LoRa Duplex");  // override the default CS, reset, and IRQ pins (optional)  // LoRa.setPins(csPin, resetPin, irqPin);// set CS, reset, IRQ pin  if (!LoRa.begin(433E6)) { // initialize ratio at 915 MHz  Serial.println("LoRa init failed. Check your connections.");  while (true); // if failed, do nothing  }  Serial.println("LoRa init succeeded.");  }  void loop() {  if (millis() - lastSendTime > interval) {  int val = analogRead(tempSensor);  float cel = ((val/1023.0)\*5000)/10;  String celString = "";  sendMessage(String(cel));  lastSendTime = millis(); // timestamp the message  interval = random(2000) + 1000; // 2-3 seconds  }  // parse for a packet, and call onReceive with the result:  onReceive(LoRa.parsePacket());  }  void sendMessage(String outgoing) {  LoRa.beginPacket(); // start packet  LoRa.write(countPacket++);  LoRa.write(destination); // add destination address  LoRa.write(localAddress); // add sender address  LoRa.write(msgCount); // add message ID  LoRa.write(outgoing.length()); // add payload length  LoRa.print(outgoing); // add payload  LoRa.endPacket(); // finish packet and send it  msgCount++; // increment message ID  }  void onReceive(int packetSize) {  if (packetSize == 0) return; // if there's no packet, return  // read packet header bytes:  int recipient = LoRa.read(); // recipient address  byte sender = LoRa.read(); // sender address  byte incomingMsgId = LoRa.read(); // incoming msg ID  byte incomingLength = LoRa.read(); // incoming msg length  String incoming = "";  while (LoRa.available()) {  incoming += (char)LoRa.read();  }  if (incomingLength != incoming.length()) { // check length for error  // Serial.println("error: message length does not match length");  return; // skip rest of function  }  // if the recipient isn't this device or broadcast,  if (recipient != localAddress && recipient != 0xFF) {  // Serial.println("This message is not for me.");  return; // skip rest of function  }  // if message is for this device, or broadcast, print details:  Serial.println("Message: " + incoming);  Serial.println();  } |

Station2

|  |
| --- |
| /\*  LoRa Duplex communication  Sends a message every half second, and polls continually  for new incoming messages. Implements a one-byte addressing scheme,  with 0xFF as the broadcast address.  Uses readString() from Stream class to read payload. The Stream class'  timeout may affect other functuons, like the radio's callback. For an  created 28 April 2017  by Tom Igoe  \*/  #include <SPI.h> // include libraries  #include <LoRa.h>  #define NSS\_PIN 10  #define RESET\_PIN 9  #define DIO0\_PIN 2  String outgoing; // outgoing message  byte msgCount = 0; // count of outgoing messages  byte localAddress = 0xBB; // address of this device  byte destination = 0xFF; // destination to send to  long lastSendTime = 0; // last send time  int interval = 2000; // interval between sends  const int buttonPin = 5;  int buttonState = 0;    void setup() {  Serial.begin(9600); // initialize serial  while (!Serial);  Serial.println("LoRa Duplex");  // override the default CS, reset, and IRQ pins (optional)  LoRa.setPins(NSS\_PIN,RESET\_PIN,DIO0\_PIN);  if (!LoRa.begin(433E6)) { // initialize ratio at 915 MHz  Serial.println("LoRa init failed. Check your connections.");  while (true); // if failed, do nothing  }  Serial.println("LoRa init succeeded.");  pinMode(buttonPin, INPUT);    }  void sendMessage(String outgoing) {  LoRa.beginPacket(); // start packet  LoRa.write(destination); // add destination address  LoRa.write(localAddress); // add sender address  LoRa.write(msgCount); // add message ID  LoRa.write(outgoing.length()); // add payload length  LoRa.print(outgoing);  LoRa.endPacket(); // finish packet and send it  msgCount++; // increment message ID  }  int receiveCounter = 0;  void onReceive(int packetSize) {  if (packetSize == 0) return; // if there's no packet, return  // read packet header bytes:  int counter = LoRa.read();  int recipient = LoRa.read(); // recipient address  byte sender = LoRa.read(); // sender address  byte incomingMsgId = LoRa.read(); // incoming msg ID  byte incomingLength = LoRa.read(); // incoming msg length  String incoming = "";  while (LoRa.available()) {  incoming += (char)LoRa.read();  }  if (incomingLength != incoming.length()) { // check length for error  //Serial.println("error: message length does not match length");  return; // skip rest of function  }  // if the recipient isn't this device or broadcast,  if (recipient != localAddress && recipient != 0xFF) {  //Serial.println("This message is not for me.");  return; // skip rest of function  }  receiveCounter++;  Serial.print("Message: " + incoming);  Serial.println();  }  void loop() {  if (millis() - lastSendTime > interval) {  String message = "Lora2 xin chao";  sendMessage(message);  // Serial.println("Sending " + message);  lastSendTime = millis(); // timestamp the message  interval = random(2000) + 1000; // 2-3 seconds  }  // parse for a packet, and call onReceive with the result:  onReceive(LoRa.parsePacket());  } |

Demonstrations

*(Demonstrations with photos of experiments)*



References

<https://github.com/sandeepmistry/arduino-LoRa/blob/master/examples/LoRaDuplex/LoRaDuplex.ino>