**ESP32 with LoRa using Arduino**

Using and RFM95 transceiver module.

LoRa is a wireless data communication technology that uses a radio modulation technique that can be generated by Semtech LoRa transceiver chips.

This modulation technique allows

* long range communication
* small amounts of data (which means a low bandwidth),
* high immunity to interference,
* while minimizing power consumption.

So, it allows long distance communication with low power requirements.

**LoRa Frequencies**

LoRa uses unlicensed frequencies that are available worldwide. These are the most widely used frequencies:

* 868 MHz for Europe
* 915 MHz for North America
* 433 MHz band for Asia

Keep in mind that LoRa is not suitable for projects that:

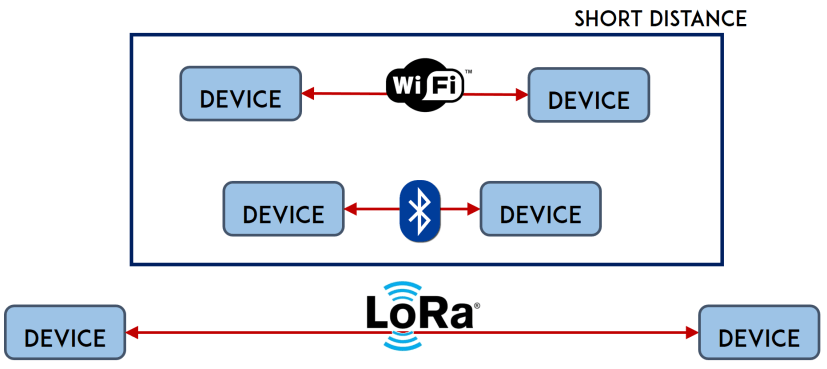
* Require high data-rate transmission;
* Need very frequent transmissions;
* Or are in highly populated networks.

**LoRa Topologies**

You can use LoRa in:

* Point to point communication
* Or build a LoRa network (using LoRaWAN for example)

Unlike Wi-Fi or Bluetooth that only support short distance communication, two LoRa devices with a proper antenna can exchange data over a long distance.

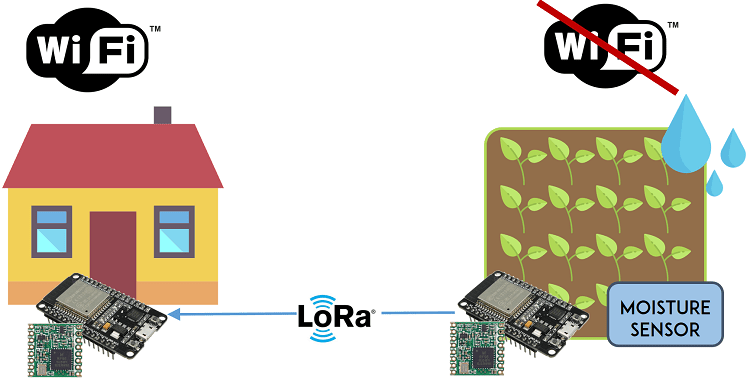


There are also other LoRa solutions that easily have a range of more than 30Km.

**LoRaWAN**

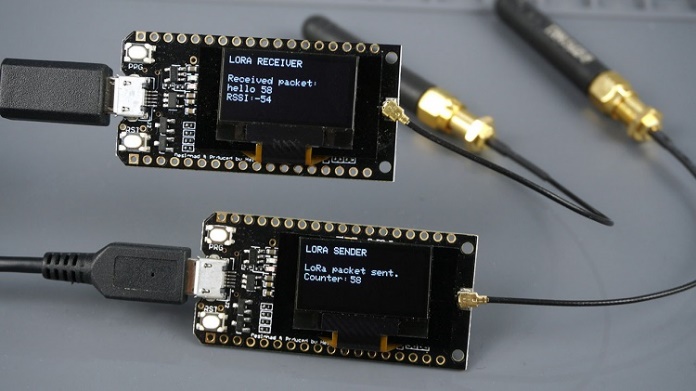
You can also build a LoRa network using LoRaWAN.

The LoRaWAN protocol is a Low Power Wide Area Network (LPWAN) specification derived from LoRa technology standardized by the LoRa Alliance. We won’t explore LoRaWAN in this tutorial, but for more information you can check the [LoRa Alliance](https://lora-alliance.org/) and [The Things Network](https://www.thethingsnetwork.org/) websites.



|  |  |
| --- | --- |
|  | [2x ESP32 DOIT DEVKIT V1 Board](https://makeradvisor.com/tools/esp32-dev-board-wi-fi-bluetooth/)  [2x LoRa Transceiver modules (RFM95)](https://makeradvisor.com/tools/rfm95-lora-transceiver-module/)  RFM95 LoRa breakout board (optional)  [Jumper wires](https://makeradvisor.com/tools/jumper-wires-kit-120-pieces/)  [Breadboard](https://makeradvisor.com/tools/mb-102-solderless-breadboard-830-points/) or [stripboard](https://makeradvisor.com/tools/prototyping-circuit-board-stripboard/)  Alternative:  [2x TTGO LoRa32 SX1276 OLED](https://makeradvisor.com/tools/ttgo-lora32-sx1276-esp32-oled/) |

If you have one of those boards, you can follow: [TTGO LoRa32 SX1276 OLED Board: Getting Started with Arduino IDE](https://randomnerdtutorials.com/ttgo-lora32-sx1276-arduino-ide/).

[](https://i2.wp.com/randomnerdtutorials.com/wp-content/uploads/2019/10/ESP32-board-with-built-in-lora-OLED.jpg?quality=100&strip=all&ssl=1)

**Installing the LoRa Library**

Open your Arduino IDE, and go to **Sketch** > **Include Library** > **Manage Libraries** and search for “**LoRa**“. Select the LoRa library highlighted in the figure below, and install it.



**Getting LoRa Tranceiver Modules**

To send and receive LoRa messages with the ESP32 we’ll be using the [RFM95 transceiver module](https://makeradvisor.com/tools/rfm95-lora-transceiver-module/). All LoRa modules are transceivers, which means they can send and receive information. You’ll need 2 of them.

|  |  |
| --- | --- |
|  | You can also use other compatible modules like Semtech SX1276/77/78/79 based boards including: RFM96W, RFM98W, etc… |
|  | Alternatively, there are ESP32 boards with LoRa and OLED display built-in like the [ESP32 Heltec Wifi Module](http://shrsl.com/ujni), or the [TTGO LoRa32 board](https://randomnerdtutorials.com/ttgo-lora32-sx1276-arduino-ide/). |

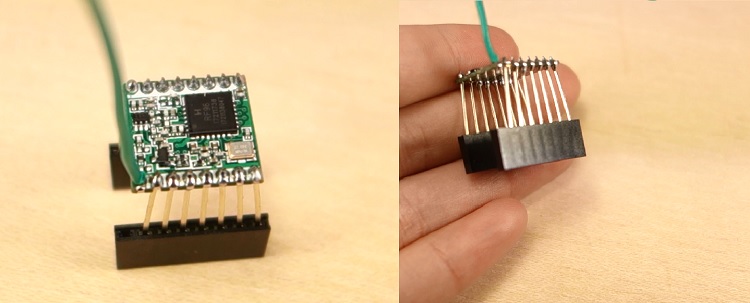
Make sure you check the correct frequency for your location. You can visit the following web page to learn more about [RF signals and regulations according to each country](https://www.thethingsnetwork.org/docs/lorawan/frequencies-by-country.html). For example, in Portugal we can use a frequency between 863 and 870 MHz or we can use 433MHz. For this project, we’ll be using an RFM95 that operates at 868 MHz.

If you have an ESP32 development board with LoRa built-in, you can skip this step.

The RFM95 transceiver isn’t breadboard friendly. A common row of 2.54mm header pins won’t fit on the transceiver pins. The spaces between the connections are shorter than usual.

|  |  |
| --- | --- |
|  | There are a few options that you can use to access the transceiver pins.   * You may solder some wires directly to the transceiver; * Break header pins and solder each one separately; * Or you can buy a breakout board that makes the pins breadboard friendly. |

We’ve soldered a header to the module as shown in the figure below.



**Antenna**

The RFM95 transceiver chip requires an external antenna connected to the ANA pin.

|  |  |  |
| --- | --- | --- |
|  | You can connect a “real” antenna, or you can make one yourself by using a conductive wire as shown in the figure below. Some breakout boards come with a special connector to add a proper antenna. |  |

The wire length depends on the frequency:

* 868 MHz: 86,3 mm (3.4 inch)
* 915 MHz: 81,9 mm (3.22 inch)
* 433 MHz: 173,1 mm (6.8 inch)

For our module we need to use a 86,3 mm wire soldered directly to the transceiver’s ANA pin. Note that using a proper antenna will extend the communication range.

**Important:** you MUST attach an antenna to the module.

**Wiring the RFM95 LoRa Transceiver Module**

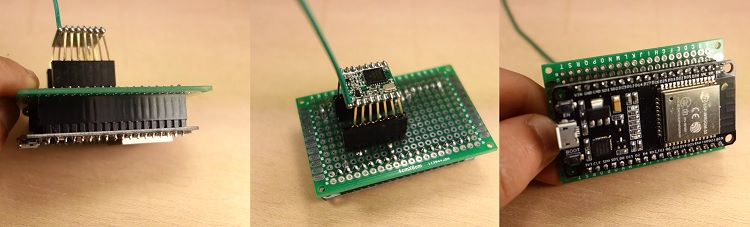
The RFM95 LoRa transceiver module communicates with the ESP32 using SPI communication protocol. So, we’ll use the ESP32 default SPI pins. Wire both ESP32 boards to the corresponding transceiver modules as shown in the next schematic diagram:

|  |  |
| --- | --- |
|  | * ANA: Antenna * GND: GND * DIO3: don’t connect * DIO4: don’t connect * 3.3V: 3.3V * DIO0: GPIO 2 * DIO1: don’t connect * DIO2: don’t connect * GND: don’t connect * DIO5: don’t connect * RESET: GPIO 14 * NSS: GPIO 5 * SCK: GPIO 18 * MOSI: GPIO 23 * MISO: GPIO 19 * GND: don’t connect |

Here’s the connections between the RFM95 LoRa transceiver module and the ESP32:

**Note:** the RFM95 transceiver module has 3 GND pins. It doesn’t matter which one you use, but you need to connect at least one.

For practical reasons we’ve made this circuit on a stripboard. It’s easier to handle, and the wires don’t disconnect. You may use a breadboard if you prefer.



**The LoRa Sender Sketch**

It transmits messages every 10 seconds using LoRa. It sends a “hello” followed by a number that is incremented in every message.

#include <SPI.h>

#include <LoRa.h>

//define the pins used by the transceiver module

#define ss 5

#define rst 14

#define dio0 2

int counter = 0;

void setup() {

Serial.begin(115200);

while (!Serial);

Serial.println("LoRa Sender");

//setup LoRa transceiver module

LoRa.setPins(ss, rst, dio0);

//433E6 for Asia

//866E6 for Europe

//915E6 for North America

while (!LoRa.begin(866E6)) {

Serial.println(".");

delay(500);

}

// Change sync word (0xF3) (0-0xFF) to match the receiver

// Sync word assures you don't get messages from other transceivers

LoRa.setSyncWord(0xF3);

Serial.println("LoRa Initializing OK!");

}

void loop() {

Serial.print("Sending packet: ");

Serial.println(counter++);

//Send LoRa packet to receiver

LoRa.beginPacket();

LoRa.print("hello ");

LoRa.print(counter);

LoRa.endPacket();

delay(10000);

}

**The LoRa Receiver Sketch**

void loop() {

int packetSize = LoRa.parsePacket();

if (packetSize) {

// read packet

while (LoRa.available()) {

String LoRaData = LoRa.readString();

Serial.print(LoRaData);

}

Serial.print("' with RSSI ");

Serial.println(LoRa.packetRssi());

}

}

