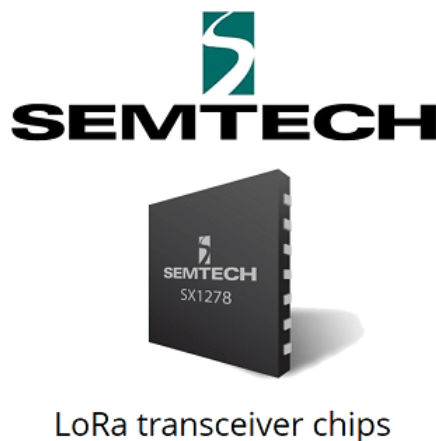
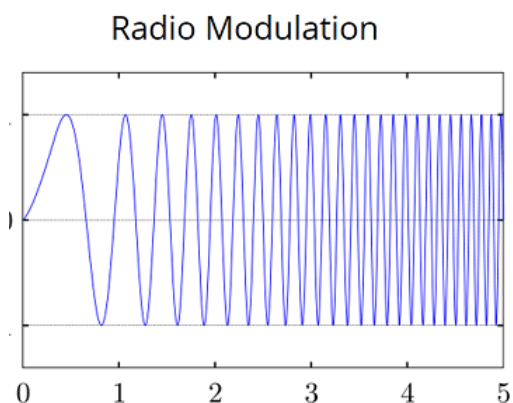


ESP32 avec LoRa utilisant Arduino IDE - Prise en main

Dans ce didacticiel, nous explorerons les principes de base de LoRa et comment il peut être utilisé avec l'ESP32 pour les projets IoT à l'aide de l'IDE Arduino. Pour vous aider à démarrer, nous vous montrerons également comment créer un simple émetteur LoRa et un récepteur LoRa avec le module émetteur-récepteur RFM95.

Qu'est-ce que LoRa?

LoRa est une technologie de communication de données sans fil qui utilise une technique de modulation radio pouvant être générée par des puces d'émetteur-récepteur Semtech LoRa.



Cette technique de modulation permet une communication à longue distance de petites quantités de données (ce qui signifie une faible bande passante), une immunité élevée aux interférences, tout en minimisant la consommation d'énergie. Ainsi, il permet une communication longue distance avec de faibles besoins en énergie.



Long distance
communication



Small amounts of data
(low bandwidth)



High immunity
to interference



Low power
consumption

Fréquences LoRa

LoRa utilise des fréquences sans licence disponibles dans le monde entier. Voici les fréquences les plus utilisées:

868 MHz pour l'Europe

915 MHz pour l'Amérique du Nord

Bande 433 MHz pour l'Asie

Parce que ces bandes ne sont pas sous licence, n'importe qui peut les utiliser librement sans payer ni avoir à obtenir une licence.

Applications LoRa

Les fonctions LoRa longue portée et faible consommation le rendent parfait pour les capteurs alimentés par batterie et les applications à faible puissance dans:

Internet des objets (IoT)

Maison intelligente

Communication de machine à machine

Et beaucoup plus...

Ainsi, LoRa est un bon choix pour les nœuds de capteurs fonctionnant sur une cellule de bobine ou alimentés par l'énergie solaire, qui transmettent de petites quantités de données.

Gardez à l'esprit que LoRa ne convient pas aux projets qui:

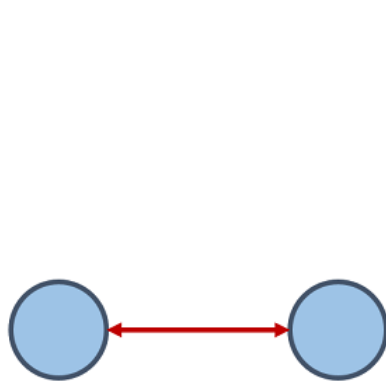
Nécessite une transmission à haut débit;

Besoin de transmissions très fréquentes;

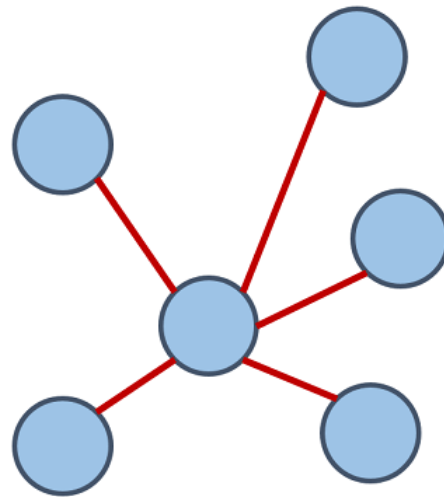
Ou sont dans des réseaux très peuplés.

Topologies LoRa

Vous pouvez utiliser LoRa dans:



Point to point
communication



Network

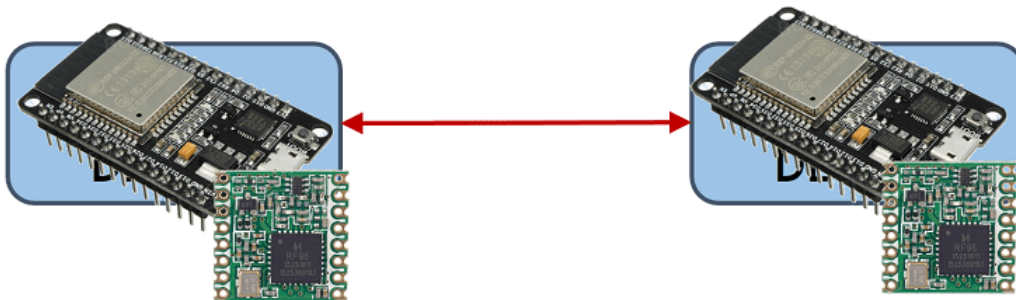
Communication point à point

Ou construisez un réseau LoRa (en utilisant LoRaWAN par exemple)

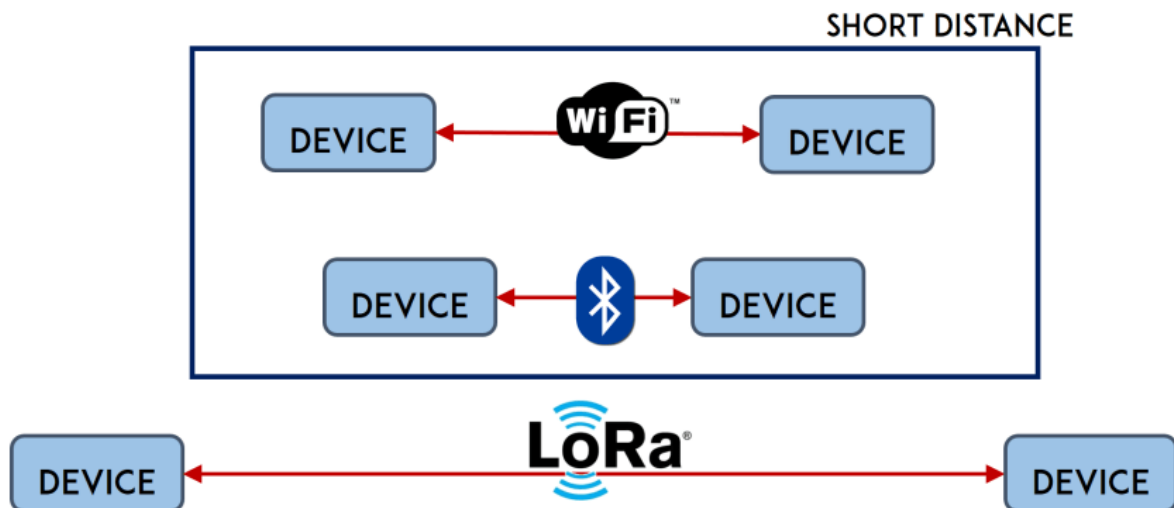
Communication point à point

Dans la communication point à point, deux appareils compatibles LoRa communiquent entre eux à l'aide de signaux RF.

Par exemple, cela est utile pour échanger des données entre deux cartes ESP32 équipées de puces d'émetteur-récepteur LoRa qui sont relativement éloignées l'une de l'autre ou dans des environnements sans couverture Wi-Fi.



Contrairement au Wi-Fi ou au Bluetooth qui ne prennent en charge que les communications à courte distance, deux appareils LoRa avec une antenne appropriée peuvent échanger des données sur une longue distance.



Vous pouvez facilement configurer votre ESP32 avec une puce LoRa pour transmettre et recevoir des données de manière fiable à plus de 200 mètres de distance (vous pouvez obtenir de meilleurs résultats en fonction de votre environnement et des paramètres LoRa). Il existe également d'autres solutions LoRa qui ont facilement une portée de plus de 30 km.

LoRaWAN

Vous pouvez également créer un réseau LoRa en utilisant LoRaWAN.

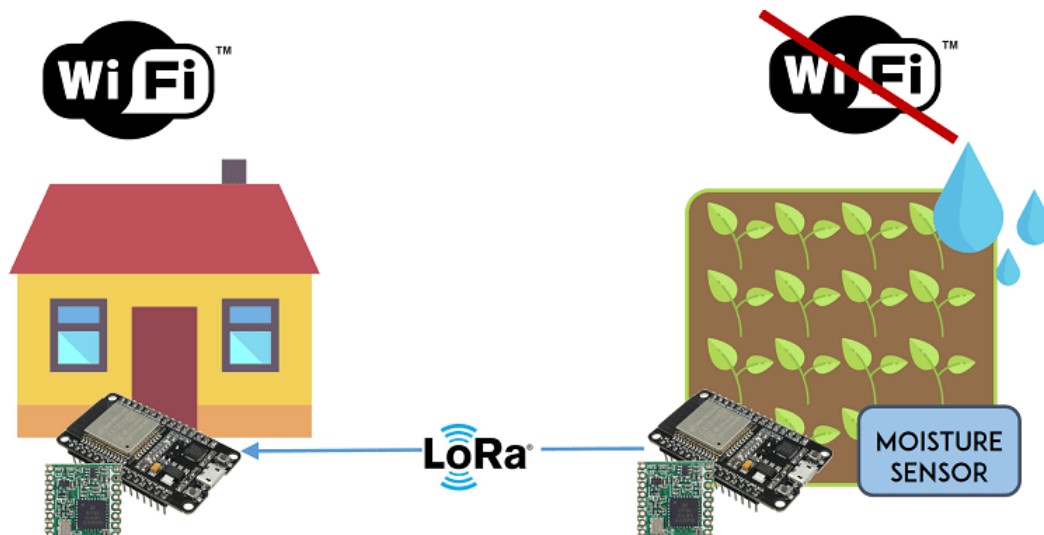


Le protocole LoRaWAN est une spécification LPWAN (Low Power Wide Area Network) dérivée de la technologie LoRa normalisée par la LoRa Alliance. Nous n'explorerons pas LoRaWAN dans ce didacticiel, mais pour plus d'informations, vous pouvez consulter les sites Web de LoRa Alliance et The Things Network.

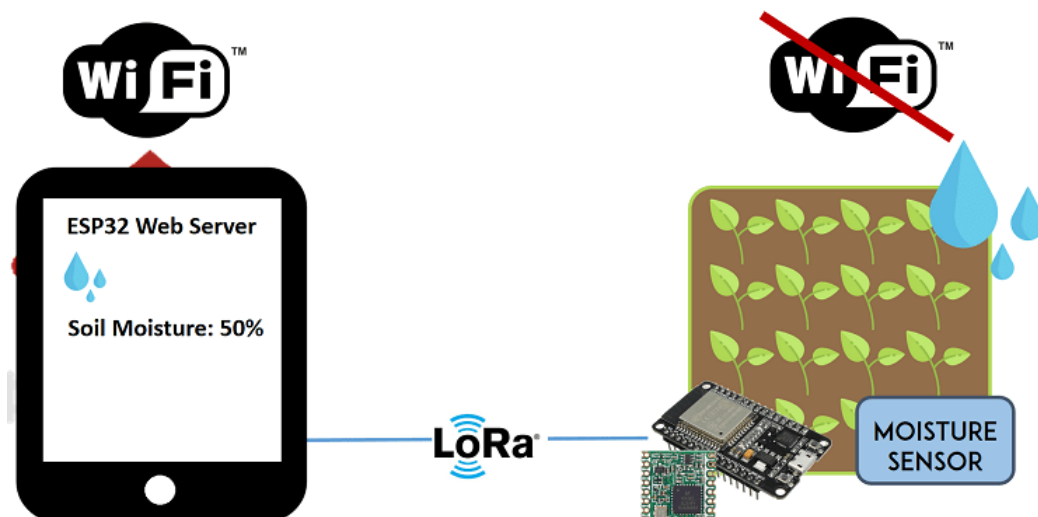
Comment LoRa peut-il être utile dans vos projets domotiques?

Jetons un coup d'œil à une application pratique.

Imaginez que vous vouliez mesurer l'humidité dans votre champ. Bien qu'il ne soit pas loin de votre maison, il n'a probablement pas de couverture Wi-Fi. Ainsi, vous pouvez créer un nœud de capteur avec un ESP32 et un capteur d'humidité, qui envoie les lectures d'humidité une ou deux fois par jour à un autre ESP32 en utilisant LoRa.



Le dernier ESP32 a accès au Wi-Fi et peut exécuter un serveur Web qui affiche les lectures d'humidité.



Ceci est juste un exemple qui illustre comment vous pouvez utiliser la technologie LoRa dans vos projets ESP32.

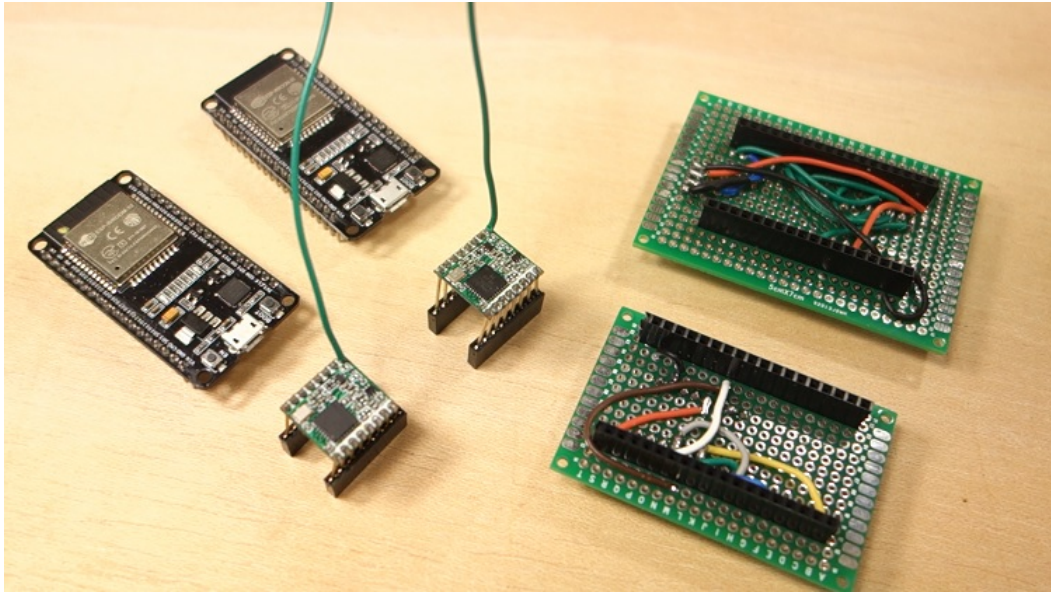
Remarque: nous enseignons comment construire ce projet sur notre cours « Apprendre ESP32 avec Arduino IDE ». Il s'agit du projet 4 sur la table des matières: **Surveillance du capteur à longue portée LoRa - Rapport des lectures du capteur de l'extérieur: humidité et température du sol.**



ESP32 avec LoRa

Dans cette section, nous vous montrerons comment démarrer avec LoRa avec votre ESP32 en utilisant Arduino IDE. À titre d'exemple, nous allons créer un simple expéditeur LoRa et un récepteur LoRa.

L'expéditeur LoRa enverra un message «bonjour» suivi d'un compteur à des fins de test. Ce message peut être facilement remplacé par des données utiles telles que des lectures de capteur ou des notifications.



Pour suivre cette partie, vous avez besoin des composants suivants:

[2x carte ESP32 DOIT DEVKIT V1](#)

[2 modules émetteur-récepteur LoRa \(RFM95\)](#)

Carte de dérivation RFM95 LoRa (en option)

[Fils de cavalier](#)

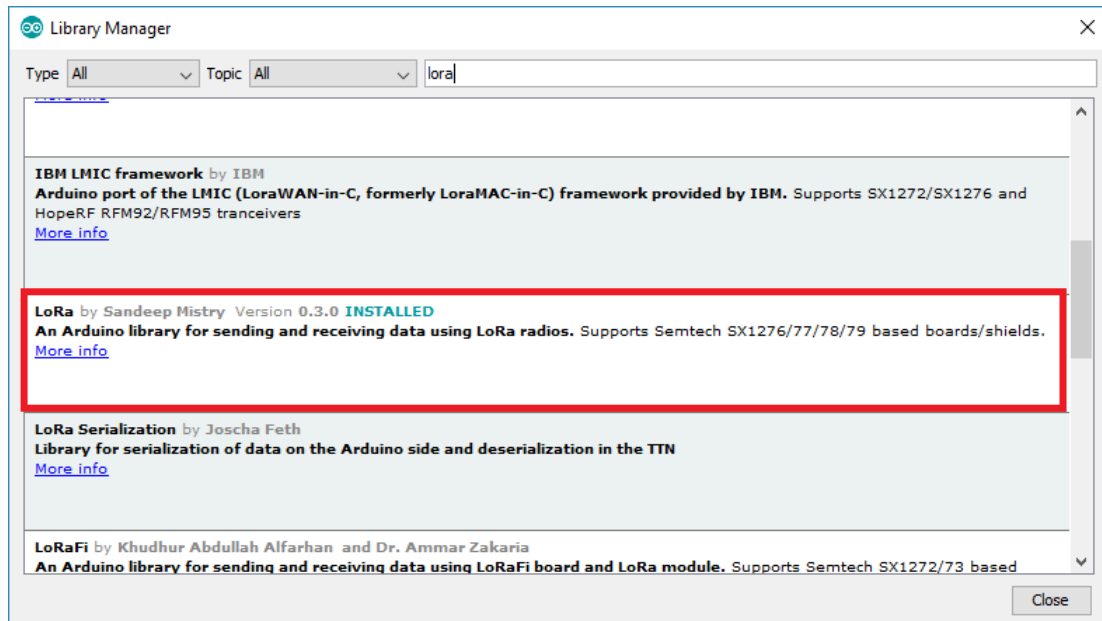
[Planche à pain](#) ou [stripboard](#)

Préparation de l'IDE Arduino

Installation de la bibliothèque LoRa

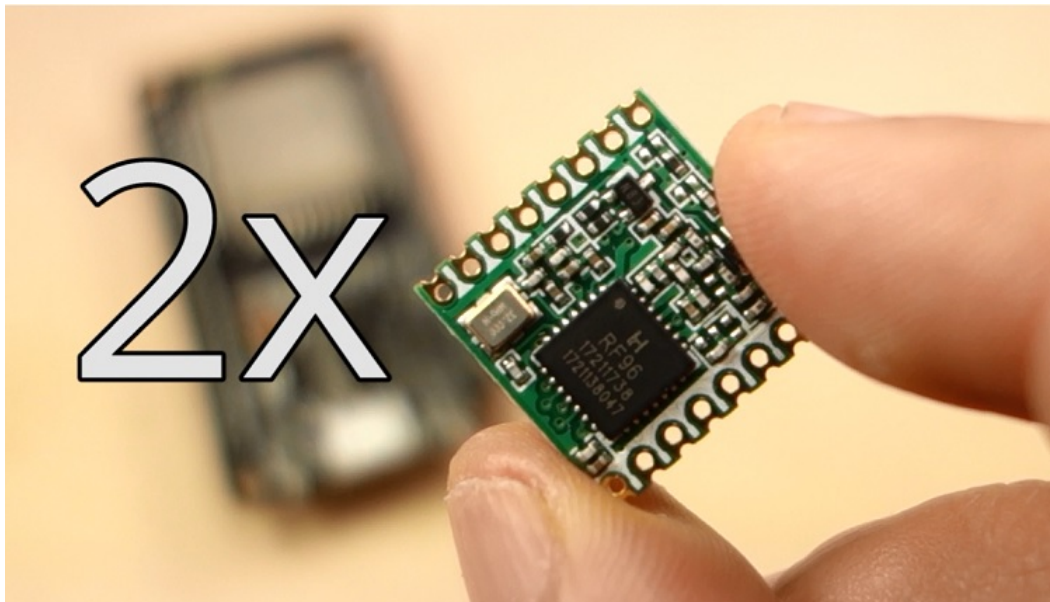
Il existe plusieurs bibliothèques disponibles pour envoyer et recevoir facilement des paquets LoRa avec l'ESP32. Dans cet exemple, nous utiliserons la [bibliothèque arduino-LoRa de sandeep mistry](#).

Ouvrez votre IDE Arduino, allez dans **Sketch > Inclure la bibliothèque > Gérer les bibliothèques** et recherchez « **LoRa** ». Sélectionnez la bibliothèque LoRa mise en évidence dans la figure ci-dessous et installez-la.



Obtention des modules LoRa Tranceiver

Pour envoyer et recevoir des messages LoRa avec l'ESP32, nous utiliserons le [module émetteur-récepteur RFM95](#) . Tous les modules LoRa sont des émetteurs-récepteurs, ce qui signifie qu'ils peuvent envoyer et recevoir des informations. Vous aurez besoin de 2 d'entre eux.



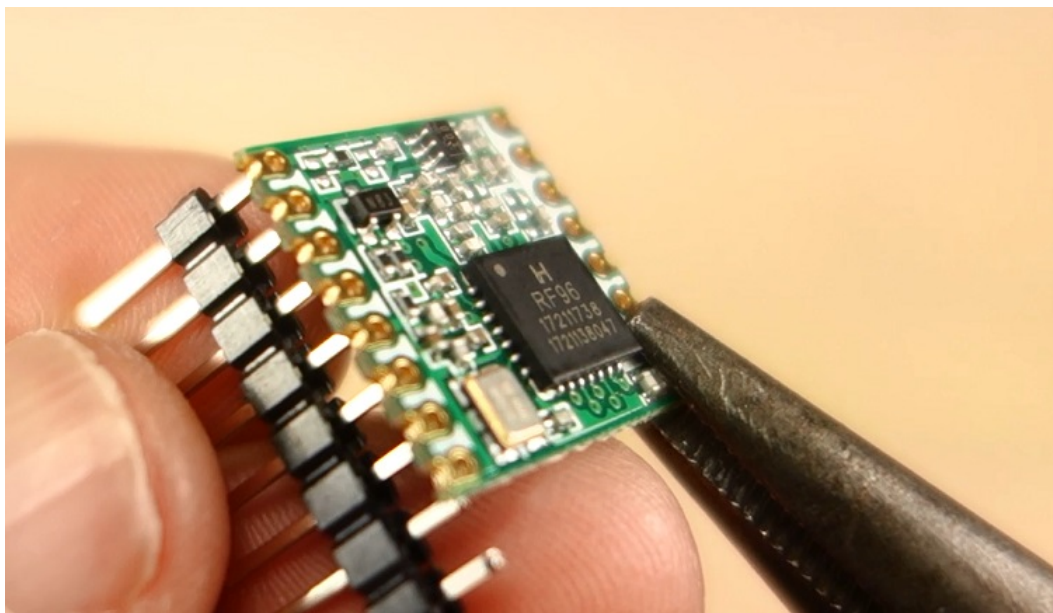
Avant de vous procurer votre module émetteur-récepteur LoRa, assurez-vous de vérifier la fréquence correcte pour votre emplacement. Vous pouvez visiter la page Web suivante pour en savoir plus sur [les signaux RF et les réglementations en fonction de chaque pays](#) . Par exemple, au Portugal,

nous pouvons utiliser une fréquence comprise entre 863 et 870 MHz ou nous pouvons utiliser 433 MHz. Pour ce projet, nous utiliserons un RFM95 fonctionnant à 868 MHz.

Préparation du module émetteur-récepteur RFM95

Si vous disposez d'une carte de développement ESP32 avec LoRa intégré, vous pouvez ignorer cette étape.

L'émetteur-récepteur RFM95 n'est pas compatible avec la maquette. Une rangée commune de broches d'en-tête de 2,54 mm ne convient pas aux broches de l'émetteur-récepteur. Les espaces entre les connexions sont plus courts que d'habitude.



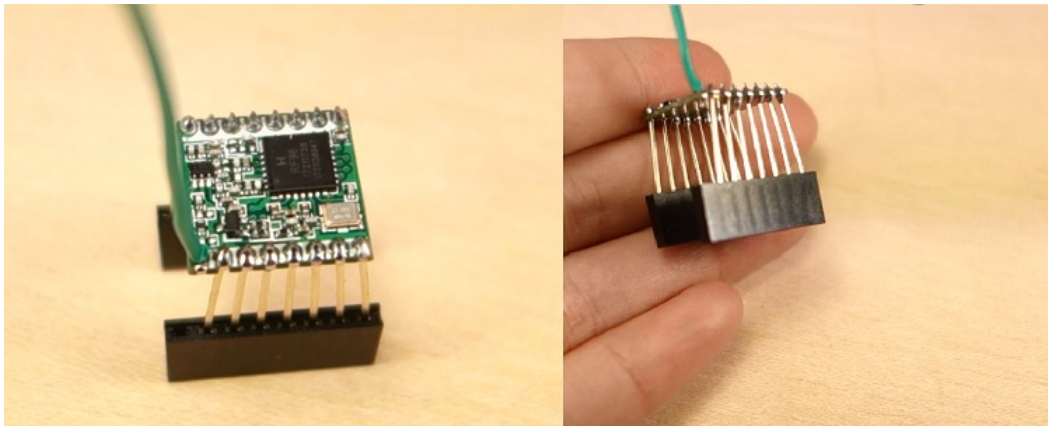
Il existe quelques options que vous pouvez utiliser pour accéder aux broches de l'émetteur-récepteur.

Vous pouvez souder certains fils directement sur l'émetteur-récepteur;

Cassez les broches d'en-tête et soudez chacune séparément;

Ou vous pouvez acheter une carte de dérivation qui rend la planche à pain de broches conviviale.

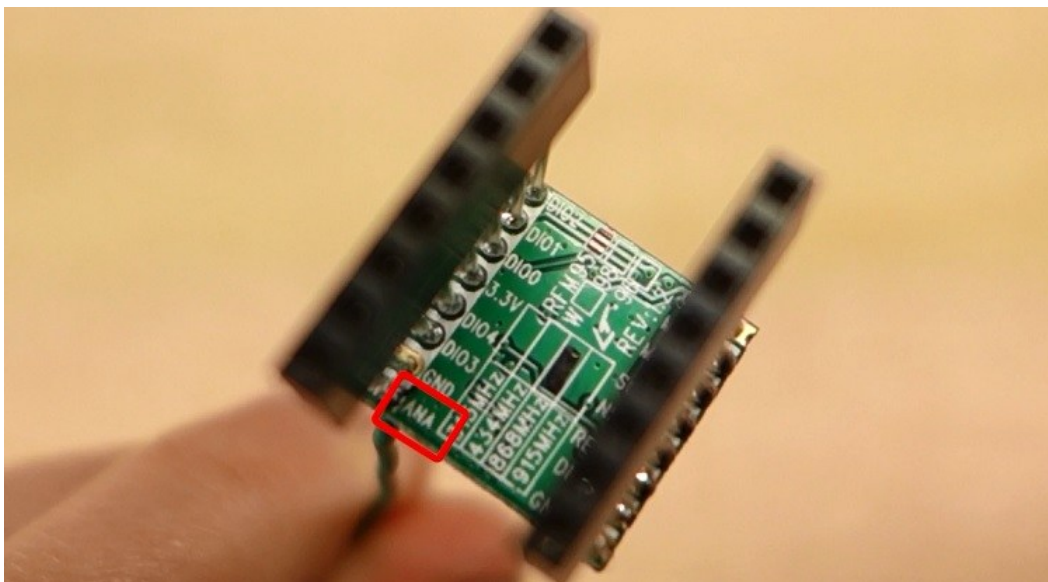
Nous avons soudé un en-tête au module comme indiqué dans la figure ci-dessous.



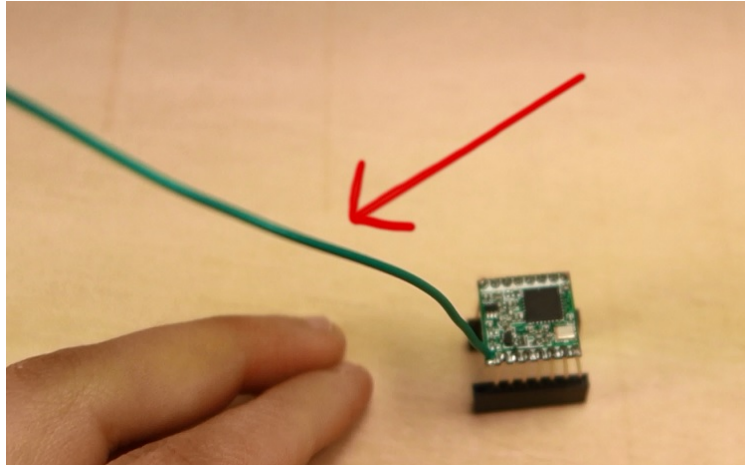
De cette façon, vous pouvez accéder aux broches du module avec des fils de cavalier réguliers, ou même mettre des broches d'en-tête pour les connecter directement à une bande ou une planche à pain.

Antenne

La puce émetteur-récepteur RFM95 nécessite une antenne externe connectée à la broche ANA.



Vous pouvez connecter une antenne «réelle», ou vous pouvez en fabriquer une vous-même en utilisant un fil conducteur comme indiqué dans la figure ci-dessous. Certaines cartes de dérivation sont livrées avec un connecteur spécial pour ajouter une antenne appropriée.



La longueur du fil dépend de la fréquence:

868 MHz: 86,3 mm (3,4 pouces)

915 MHz: 81,9 mm (3,22 pouces)

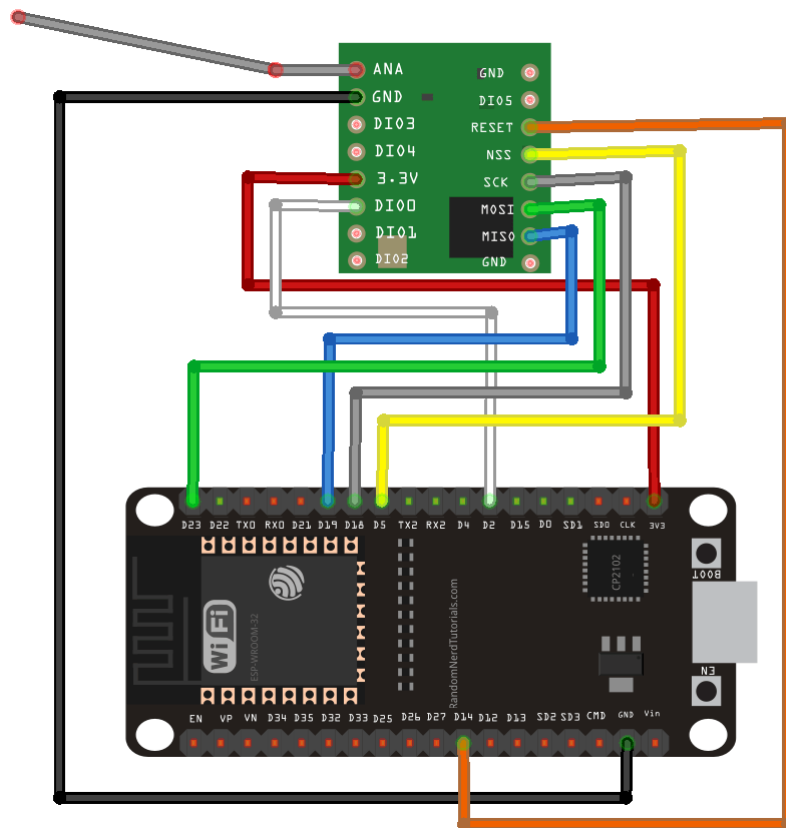
433 MHz: 173,1 mm (6,8 pouces)

Pour notre module, nous devons utiliser un fil de 86,3 mm soudé directement à la broche ANA de l'émetteur-récepteur. Notez que l'utilisation d'une antenne appropriée étendra la portée de communication.

Important: vous DEVEZ attacher une antenne au module.

Câblage du module émetteur-récepteur RFM95 LoRa

Le module émetteur-récepteur RFM95 LoRa communique avec l'ESP32 à l'aide du protocole de communication SPI. Nous allons donc utiliser les broches SPI par défaut de ESP32. Câblez les deux cartes ESP32 aux modules émetteurs-récepteurs correspondants comme indiqué dans le schéma de principe suivant:



Voici les connexions entre le module émetteur-récepteur RFM95 LoRa et l'ESP32:

ANA: Antenne

GND: **GND**

DIO3: ne pas se connecter

DIO4: ne pas se connecter

3,3 V: **3,3 V**

DIO0: **GPIO 2**

DIO1: ne pas se connecter

DIO2: ne pas se connecter

GND: ne pas se connecter

DIO5: ne pas se connecter

RÉINITIALISER: **GPIO 14**

NSS: **GPIO 5**

SCK: **GPIO 18**

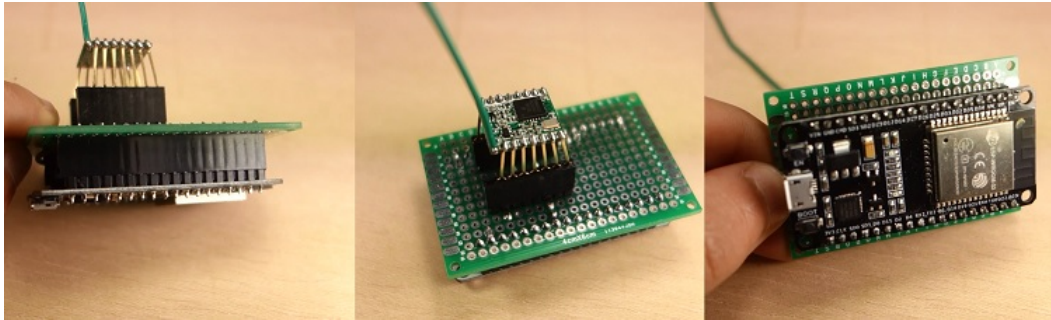
MOSI: **GPIO 23**

MISO: **GPIO 19**

GND: ne pas se connecter

Remarque: le module émetteur-récepteur RFM95 possède 3 broches GND. Peu importe celui que vous utilisez, mais vous devez en connecter au moins un.

Pour des raisons pratiques, nous avons réalisé ce circuit sur un stripboard. C'est plus facile à manipuler et les fils ne se déconnectent pas. Vous pouvez utiliser une maquette si vous préférez.



L'esquisse de l'expéditeur LoRa

Ouvrez votre IDE Arduino et copiez le code suivant. Cette esquisse est basée sur un exemple de la bibliothèque LoRa. Il transmet des messages toutes les 10 secondes en utilisant LoRa. Il envoie un «bonjour» suivi d'un nombre qui est incrémenté dans chaque message.

```
/******
```

```
Modified from the examples of the Arduino LoRa library
```

```
More resources: https://randomnerdtutorials.com
```

```
*****/
```

```
#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() {
```

```
//initialize Serial Monitor
```

```

Serial.begin(115200);

while (!Serial);

Serial.println("LoRa Sender");


//setup LoRa transceiver module

LoRa.setPins(ss, rst, dio0);


//replace the LoRa.begin(---E-) argument with your location's frequency
//433E6 for Asia
//866E6 for Europe
//915E6 for North America
while (!LoRa.begin(866E6)) {
  Serial.println(".");
  delay(500);
}

// Change sync word (0xF3) to match the receiver
// The sync word assures you don't get LoRa messages from other LoRa transceivers
// ranges from 0-0xFF
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();
}

```

```

counter++;

delay(10000);
}

```

Jetons un coup d'œil au code.

Il commence par inclure les bibliothèques nécessaires.

```

#include <SPI.h>
#include <LoRa.h>

```

Ensuite, définissez les broches utilisées par votre module LoRa. Si vous avez suivi le schéma précédent, vous pouvez utiliser la définition de broche utilisée dans le code. Si vous utilisez une carte ESP32 avec LoRa intégré, vérifiez les broches utilisées par le module LoRa de votre carte et effectuez la bonne affectation des broches.

```

#define ss 5
#define rst 14
#define dio0 2

```

Vous initialisez le `compteur` variable qui commence à 0;

```
int counter = 0;
```

dans le `installer()`, vous initialisez une communication série.

```

Serial.begin(115200);
while (!Serial);

```

Définissez les broches du module LoRa.

```
LoRa.setPins(ss, rst, dio0);
```

Et initialisez le module émetteur-récepteur avec une fréquence spécifiée.

```

while (!LoRa.begin(866E6)) {
    Serial.println(".");
    delay(500);
}

```

Vous devrez peut-être modifier la fréquence pour qu'elle corresponde à la fréquence utilisée dans votre région. Choisissez l'une des options suivantes:

433E6

866E6

915E6

Les modules émetteurs-récepteurs LoRa écoutent les paquets dans sa portée. Peu importe d'où viennent les paquets. Pour vous assurer de ne recevoir que les paquets de votre expéditeur, vous pouvez définir un mot de synchronisation (compris entre 0 et 0xFF).

```
LoRa.setSyncWord(0xF3);
```

Le destinataire et l'expéditeur doivent utiliser le même mot de synchronisation. De cette façon, le récepteur ignore tous les paquets LoRa qui ne contiennent pas ce mot de synchronisation.

Ensuite, dans le `boucle()` vous envoyez les paquets LoRa. Vous initialisez un paquet avec le `beginPacket()` méthode.

```
LoRa.beginPacket();
```

Vous écrivez des données dans le paquet en utilisant le `impression()` méthode. Comme vous pouvez le voir dans les deux lignes suivantes, nous envoyons un message bonjour suivi du compteur.

```
LoRa.print("hello ");
```

```
LoRa.print(counter);
```

Ensuite, fermez le paquet avec le `endPacket()` méthode.

```
LoRa.endPacket();
```

Après cela, le message du compteur est incrémenté de un dans chaque boucle, ce qui se produit toutes les 10 secondes.

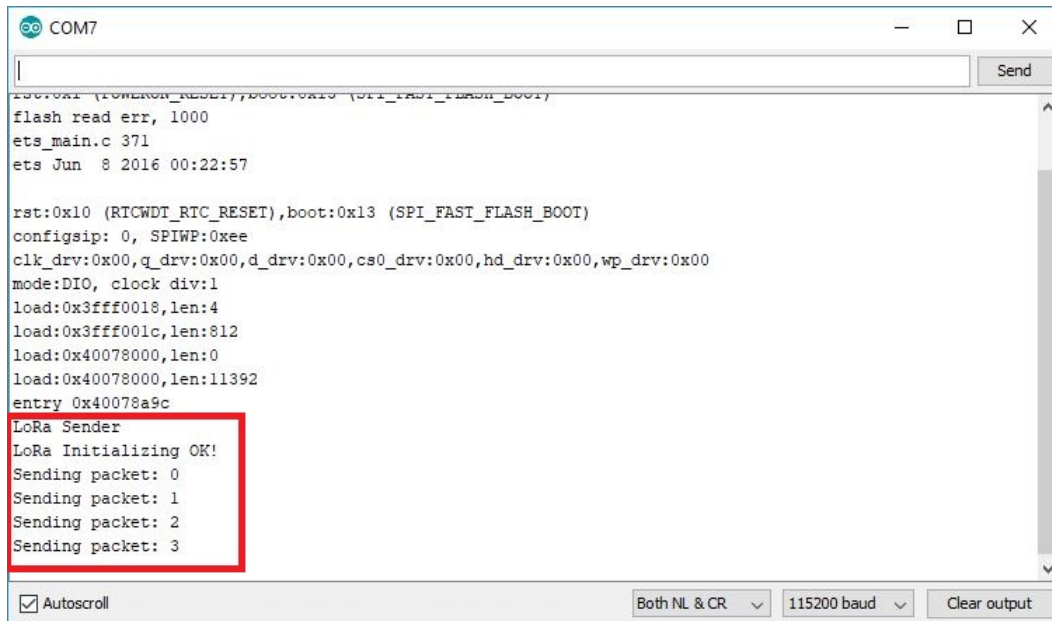
```
counter++;
```

```
delay(10000);
```

Test de l'esquisse de l'expéditeur

Téléchargez le code sur votre carte ESP32. Assurez-vous que vous avez sélectionné la bonne carte et le bon port COM.

Après cela, ouvrez le moniteur série et appuyez sur le bouton d'activation ESP32. Vous devriez voir un message de réussite comme indiqué dans la figure ci-dessous. Le compteur doit être incrémenté toutes les 10 secondes.



```
COM7
flash read err, 1000
ets_main.c 371
ets Jun  8 2016 00:22:57

rst:0x10 (RTCWDT_RTC_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:812
load:0x40078000,len:0
load:0x40078000,len:11392
entry 0x40078a9c
LoRa Sender
LoRa Initializing OK!
Sending packet: 0
Sending packet: 1
Sending packet: 2
Sending packet: 3

Autoscroll Both NL & CR 115200 baud Clear output
```

L'esquisse du récepteur LoRa

Maintenant, prenez un autre ESP32 et téléchargez le croquis suivant (le croquis du récepteur LoRa). Cette esquisse écoute les paquets LoRa avec le mot de synchronisation que vous avez défini et imprime le contenu des paquets sur le moniteur série, ainsi que le RSSI. Le RSSI mesure la force relative du signal reçu.

```
/******
```

Modified from the examples of the Arduino LoRa library

More resources: <https://randomnerdtutorials.com>

```
*****/
```

```
#include <SPI.h>
```

```
#include <LoRa.h>
```

```
//define the pins used by the transceiver module
```

```
#define ss 5
```

```
#define rst 14
```

```
#define dio0 2
```

```
void setup() {
```

```

//initialize Serial Monitor

Serial.begin(115200);

while (!Serial);

Serial.println("LoRa Receiver");


//setup LoRa transceiver module

LoRa.setPins(ss, rst, dio0);


//replace the LoRa.begin(---E-) argument with your location's frequency
//433E6 for Asia
//866E6 for Europe
//915E6 for North America
while (!LoRa.begin(866E6)) {

  Serial.println(".");

  delay(500);

}

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

// The sync word assures you don't get LoRa messages from other LoRa transceivers

// ranges from 0-0xFF

LoRa.setSyncWord(0xF3);

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

}


void loop() {

  // try to parse packet

  int packetSize = LoRa.parsePacket();

  if (packetSize) {

    // received a packet

    Serial.print("Received packet ");

    // read packet

```

```

while (LoRa.available()) {

    String LoRaData = LoRa.readString();

    Serial.print(LoRaData);

}

// print RSSI of packet

Serial.print(" with RSSI ");

Serial.println(LoRa.packetRssi());

}
}

```

Cette esquisse est très similaire à la précédente. Seulement `loop ()` est différent.

Vous devrez peut-être modifier la fréquence et le mot synoptique pour qu'ils correspondent à ceux utilisés dans l'esquisse de l'expéditeur.

dans le `loop ()` le code vérifie si un nouveau paquet a été reçu en utilisant le `parsePacket ()` méthode.

```
int packetSize = LoRa.parsePacket();
```

S'il y a un nouveau paquet, nous lisons son contenu tant qu'il est disponible.

Pour lire les données entrantes, utilisez le `readString ()` méthode.

```

while (LoRa.available()) {

    String LoRaData = LoRa.readString();

    Serial.print(LoRaData);

}

```

Les données entrantes sont enregistrées sur le `LoRaData` variable et imprimé dans le moniteur série.

Enfin, les deux lignes de code suivantes impriment le RSSI du paquet reçu en dB.

```

Serial.print(" with RSSI ");

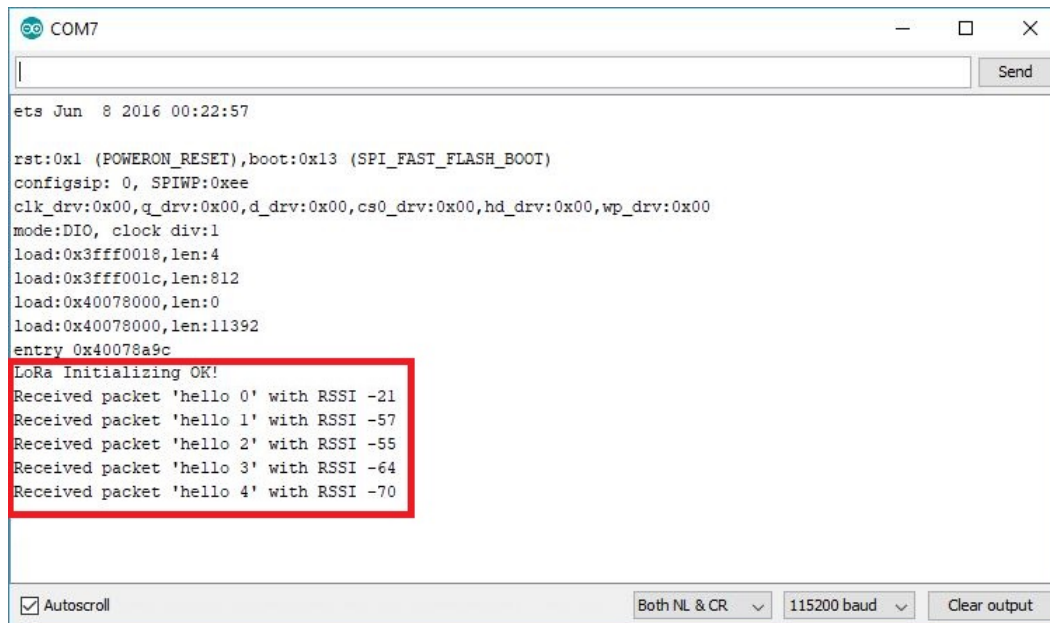
Serial.println(LoRa.packetRssi());

```

Test de l'esquisse du récepteur LoRa

Téléchargez ce code sur votre ESP32. À ce stade, vous devriez avoir deux cartes ESP32 avec des croquis différents: l'expéditeur et le récepteur.

Ouvrez Serial Monitor pour le récepteur LoRa et appuyez sur le bouton d'activation de LoRa Sender. Vous devriez commencer à recevoir les paquets LoRa sur le récepteur.



The screenshot shows a Serial Monitor window titled "COM7". The output text is as follows:

```
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:812
load:0x40078000,len:0
load:0x40078000,len:11392
entry 0x40078a9c
LoRa Initializing OK!
Received packet 'hello 0' with RSSI -21
Received packet 'hello 1' with RSSI -57
Received packet 'hello 2' with RSSI -55
Received packet 'hello 3' with RSSI -64
Received packet 'hello 4' with RSSI -70
```

At the bottom of the window, there are controls: a checked "Autoscroll" checkbox, a "Both NL & CR" dropdown menu, a "115200 baud" dropdown menu, and a "Clear output" button.