LoRa Module Setup with ESP32

What is LoRa?

LoRa (Long Range) is a wireless communication technology that enables long-range communication with extremely low power consumption. Chips such as SX1276 / SX1278 support this protocol.
- In open areas, it can reach up to 15 km or more with a proper antenna and clear line of sight.
- In urban environments, the range is typically 2–5 km.

Frequency bands: 433MHz, 868MHz, 915MHz
The module communicates with microcontrollers via SPI protocol.

Common Use Cases

- Internet of Things (IoT): Transmitting temperature, humidity, pressure from remote sensors
- Smart Agriculture: Monitoring soil and water in farms
- Smart City: Monitoring trash bins, air quality
- Robotics: Communication between mobile units without WiFi
- Private Networks: Creating independent networks without internet

Hardware Connections

ESP32 Pin	LoRa Pin
3.3V	VCC
GND	GND
GPIO18	SCK
GPIO23	MOSI
GPIO19	MISO
GPIO5	SS (NSS/CS)
GPIO14	RESET
GPIO2	DIO0

Antenna Specification: BW433FNX75-35B1

A spring-type antenna designed for 433 MHz wireless modules (LoRa, RF, etc.)

Frequency Range: 433 MHz
Type: Spring Antenna
Gain: ~2–5 dBi

By Saba Soleimanzadeh

Impedance: 50 ohms
Length: 3.5 cm

Libraries & Functions

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

Initialization

LoRa.begin(433E6);
LoRa.setPins(csPin, resetPin, DIO0);

Sending Data

LoRa.beginPacket();
LoRa.print("Hello");
LoRa.endPacket();

Receiving Data

int packetSize = LoRa.parsePacket();
int data = LoRa.read();
String msg = LoRa.readString();

Advanced Communication Settings

LoRa.setSpreadingFactor(7);
LoRa.setSignalBandwidth(125E3);
LoRa.setCodingRate4(5);
LoRa.setTxPower(14);

Signal Metrics

LoRa.packetRssi();
LoRa.packetSnr();
LoRa.available();

Power Modes

LoRa.idle();
LoRa.receive();
LoRa.sleep();

By Saba Soleimanzadeh

Using LoRa.setSyncWord(0x12)

The Sync Word is an 8-bit identifier (0–255) acting as a network ID.
Only devices with the same Sync Word can communicate with each other.

Challenges & Solutions

One-way Communication:
Solution: Modify both ESP32 to handle sending and receiving using flags and LoRa.parsePacket().

No Response to Received Messages:
Solution: Implemented reply mechanism using 'ack' messages.

Why is LoRa.receive() not used in the transmitter?

Unless using interrupt-based reception, LoRa.receive() isn't required.
The library automatically handles the reception mode.

Using esp_timer in ESP32 for Periodic Sending

esp_timer is a low-level, high-precision timer managed by FreeRTOS.

esp_timer_create_args_t and esp_timer_start_periodic() allow periodic tasks without delay().

Receiver Behavior with Interrupts

attachInterrupt(digitalPinToInterrupt(DIO0), onReceiveInterrupt, RISING);
DIO0 goes HIGH when data is received – triggers an interrupt.

DIO Pins Overview

DIO0: End of TX or RX (Interrupt)
DIO1: Preamble detected or timeout
DIO2: FIFO level or frequency hopping
DIO3: CAD done (Channel Activity Detection)
DIO4/5: Configurable for special functions

By Saba Soleimanzadeh