AKIO TV basic LoRa guide

This document explains some more details of the setup featured in the video “what is LoRa?”. It includes wiring diagrams, code and general useful information.

Components involved

- 2x ESP8266 or other 3.3V microcontroller (3.3V because LoRa modules operate at this voltage. 5V microcontrollers can be used with a level converter, or if the LoRa module is integrated into a bigger component e.g. an Arduino LoRa shield)
- 2x RFM95 LoRa module (868MHz, EU frequency)

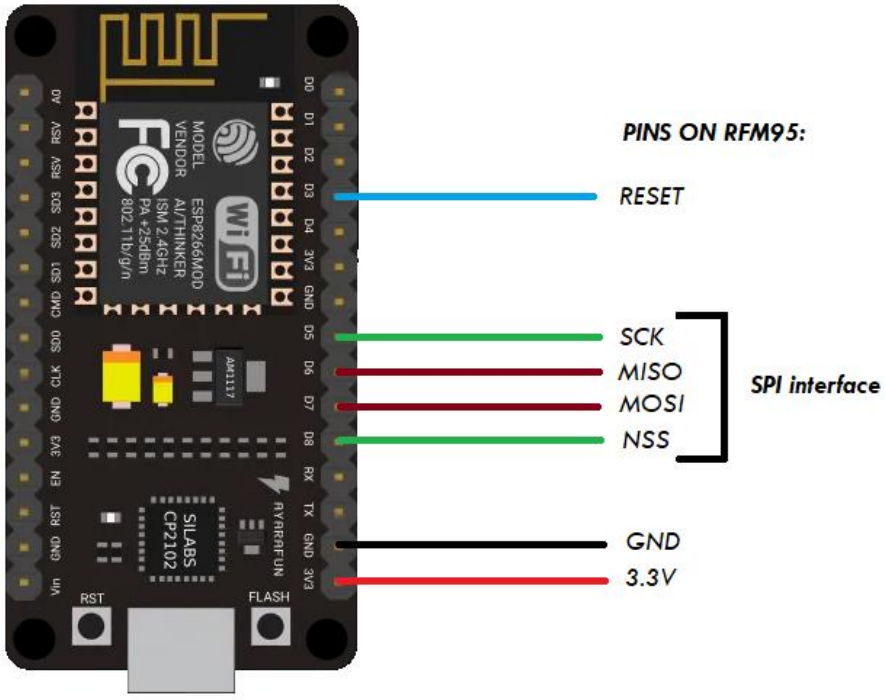
Basic principle

The setup in featured in the video is a simple one-way system. One unit is the transmitter, the other is the receiver. The transmitter sends out the test message “TEST @868MHz TXp 20dBm” every 5 seconds, the receiver picks up this message and sends it to an attached computer. The microcontroller is connected to the LoRa module using an SPI interface.

A two-way setup is entirely possible and may be built by modifying the code.

Hardware

A wiring diagram is shown below:



You must also wire up an antenna to the antenna pin of the LoRa module. If you don’t do this you may destroy the module, depending on the transmit power used. The recommended size for an 868MHz antenna is about 8cm.

Software

The setup code for both the transmitter and receiver is identical:

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

int counter = 0;

void setup() {
  Serial.begin(9600);


  Serial.println("LoRa Sender");

  LoRa.setPins(15, 0);
  LoRa.setTxPower(20);
  LoRa.setSpreadingFactor(12);
  LoRa.setSignalBandwidth(15.6E3);
  LoRa.setPreambleLength(40);
  LoRa.setSyncWord(0x31);


  LoRa.begin(868E6);

}
```

In summary:

- Include the required libraries
- Begin serial communication (for sending data to the laptop)
- Set the NSS pin to 15 and the reset pin to 0 (as seen in wiring diagram)
- Set SF, SB, PL for best range and worst capacity
- Set sync word to 0x31: This allows you to ignore other LoRa transmitters. Our transmitter will send with word 0x31, and our receiver will ignore transmitters that use other sync words or no sync word at all. The sync word is arbitrary and you can pick one you like.
- Begin LoRa at 868MHz (868E6 or 868 x 10 ^6 Hz)

The following code is used to transmit data:

```
void loop() {
  Serial.print("Sending packet: ");
  Serial.println(counter);

  // send packet
  LoRa.beginPacket();
  LoRa.print("TEST @868MHz TXp:20dBm ");
  LoRa.print(counter);
  LoRa.endPacket();

  counter++;

  delay(5000);
}
```

The following code is used to receive data:

```
void loop() {
  // try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize) {

    Serial.print("Received packet '");

    while (LoRa.available()) {
      Serial.print((char)LoRa.read());
    }

    Serial.print("' with RSSI ");
    Serial.println(LoRa.packetRssi());
  }
}
```

I didn't come up with all this code from scratch, there is a lot of useful example code on the GitHub page for the Arduino LoRa library. Go take a look at it here: https://github.com/sandeepmistry/arduino-LoRa