Usage of LoRa Radio Shield 433MHz /868MHz

The LoRa Shield allows the user to send data and reach extremely long ranges at low data-rates. It provides ultra-long range spread spectrum communication and high interference immunity whilst minimising current consumption.

Pins usage on Arduino

- D2 - RFM95_INT
- D9 - RFM9x_RS
- D10 - RFM9x_CS
- D11 - RFM9x_MOSI
- D12 - RFM9x_MISO
- D13 - RFM9x_SCK

Arduino Library

Download the demo code form our website:

433MHz:

868MHz:

Place the RadioHead library folder your arduino sketch folder/libraries/ folder.

Hardware connection:
With this test, you need two Aruduino main boards. Just plug it to you Arduino or maduino. And upload the code, it will work ok.

**Basic RX & TX example**

Let’s get a basic demo going, where one Arduino transmits and the other receives.

Transmitter example code
Once uploaded you should see the following on the serial console

```cpp
#include <SPI.h>
#include <RF_RF95.h>

#define RFM95_CS 10
#define RFM95_RST 9
#define RFM95_INT 2

// Change to 434.0 or other frequency, must match RX's freq!
#define RF95_FREQ 434.0    // 868.0

// Singleton instance of the radio driver
RF_RF95 rf95 (RFM95_CS, RFM95_RST);
```
Now open up another instance of the Arduino IDE - this is so you can see the serial console output from the TX Arduino while you set up the RX Arduino.

Receiver example code
Upload the code to receiver maduino.

Now open up the Serial console on the receiver, while also checking in on the transmitter's serial console. You should see the receiver is...well, receiving packets
You can see that the library example prints out the hex-bytes received: 48 65 6C 6C 6F 20 57 6F 72 6C 64 20 23 35 37 32 0 20 20 0. As well as the ASCII 'string' Hello World. Then it will send a reply.

And, on the transmitter side, it is now printing that it got a reply after each transmission And hello back to you because it got a reply from the receiver.