

Renzo Mischianti's E220 Articles

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: settings and basic usage](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: library](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: configuration](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: fixed transmission, broadcast, monitor, and RSSI](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: power-saving and sending structured data](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: WOR microcontroller and Arduino shield](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: WOR microcontroller and WeMos D1 shield](#)

[Ebyte LoRa E220 device for Arduino, esp32 or esp8266: WOR microcontroller and esp32 dev v1 shield](#)

[Github Ebyte E220 library](#)

[Mischianti Arduino LoRa shield \(Open source\)](#)

[Mischianti WeMos LoRa shield \(Open source\)](#)

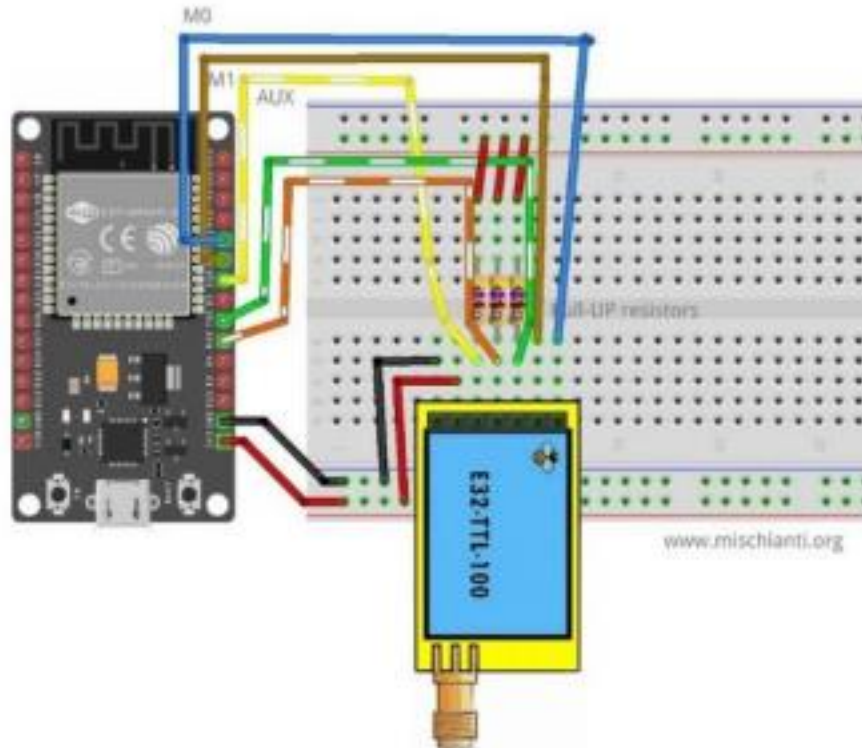
[Mischianti ESP32 DOIT DEV KIT v1 shield \(Open source\)](#)

[Support Forum](#)

Frizing diagram for Doit ESP32 Devkit V1; Full Schema:

esp32

Similar connection schema for esp32, but for RX and TX, we use RX2 and TX2 because, by default, esp32 doesn't have SoftwareSerial but has 3 Serial.



Ebyte LoRa E220 device esp32 dev kit v1 breadboard full connection

[Completed Ebyte E220-900T30D Project Code](#)

Configuration settings:

1. Load and run sender: "01_setConfiguration_WOR_Sender.ino", find file in completed project code. This is from the xReef, E220 Library example –modified to use only ESP32 pins and WOR Sender configuration.
2. Load and run receiver: "01_setConfiguration_WOR_Receiver.ino", find file in completed project code. This is from the xReef, E220 Library example –modified to use only ESP32 pins and WOR Receiver configuration.

Suggestion: Label one breadboard "Sender" and the other breadboard "Receiver."

Connect the breadboard labeled "Sender"; load "E220_Transceiver_Videofeed_Sender.ino" into the Arduino IDE and run.

Connect the breadboard labeled "Receiver"; load "E220_Transceiver_Videofeed_Receiver.ino" into the Arduino IDE and run.

ESP32 Core 2.0.14 was used in development of project.

Development of the E220 project would not have been possible without Renzo Mischianti's E220 library and his articles covering EByte Transceivers. Well, done Mr. Mischianti!

Using dual windows, Arduino IDE Serial Monitors; open "Sender" and run. Open serial monitors in one window and "Receiver" in the other window. Run "receiver" and open serial monitor. Goto static Ip Address "10.0.0.27/relay". Make this window as small as you are able to and still see the refresh icon.

"Sender" should show the web server information and connection result. Receiver should show Starting Deep sleep.

Select the refresh icon. "Sender" will show a countdown timer triggered and battery power turned on. Countdown timer will expire in about two minutes. "Receiver" should show Battery power off and going to deep sleep. Repeat refresh to wake ESP32 by WOR.

No attempt was made to measure ESP32 Devkit V1 deep sleep current; not optimized for lowest current consumption, plus being a development board not well suited for battery power.

[WOR Current meter readings](#) Increased the separation between transceivers.

[EByte, E220 Series WOR Configuration notes](#)