

# Getting Started with ESP-NOW

Will guide you to communicate between ESP32 controllers by using ESP NOW protocol.

 Difficulté Facile

 Durée 1 heure(s)

 Catégories Électronique

 Coût 10 USD (\$)

## Sommaire

Introduction

Étape 1 - Get PCBs for Your Projects Manufactured

Étape 2 - How is it different from existing protocols?

Max Distance:

Maximum nodes:

Applications:

Étape 3 - Setting Up the Receiver

Étape 4 - Setting Up the Transmitter

Étape 5 - Testing the Connection

Étape 6 - Wrap Up:

Commentaires

## Introduction

ESP-NOW is a wireless communication protocol based on the data-link layer that enables the direct, quick, and low-power control of smart devices without the need for a router. Espressif defines it and can work with Wi-Fi and Bluetooth LE. ESP-NOW provides flexible and low-power data transmission to all interconnected devices. It can also be used as an independent protocol that helps with device provisioning, debugging, and firmware upgrades.

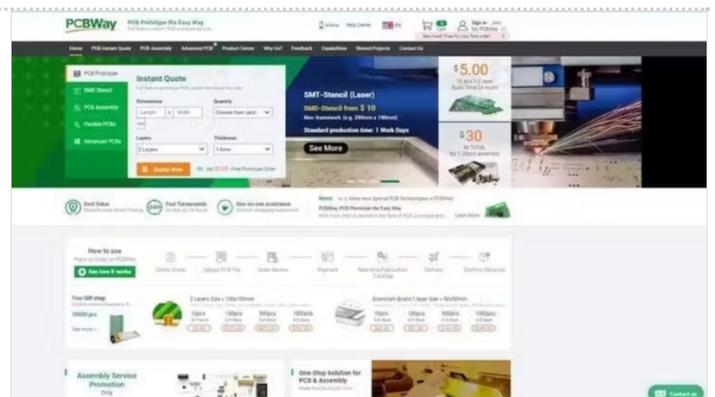
ESP-NOW is a connectionless communication protocol developed by Espressif that features short packet transmission. This protocol enables multiple devices to talk to each other in an easy way. It is a fast communication protocol that can be used to exchange small messages (up to 250 bytes) between ESP32 or ESP8266 boards. ESP-NOW supports the following features: Encrypted and unencrypted unicast communication; Mixed encrypted and unencrypted peer devices; Up to 250-byte payload can be carried; Sending callback function that can be set to inform the application layer of transmission success or failure.

## Matériaux

### Étape 1 - Get PCBs for Your Projects Manufactured

You must check out PCBWAY for ordering PCBs online for cheap! You get 10 good-quality PCBs manufactured and shipped to your doorstep for cheap. You will also get a discount on shipping on your first order. Upload your Gerber files onto PCBWAY to get them manufactured with good quality and quick turnaround time. PCBWay now could provide a complete product solution, from design to enclosure production. Check out their online Gerber viewer function. With reward points, you can get free stuff from their gift shop.

## Outils



---

## Étape 2 - How is it different from existing protocols?

ESP-NOW is a wireless communication protocol that is different from Wi-Fi and Bluetooth in that it reduces the five layers of the OSI model to only one<sup>1</sup>. Additionally, ESP-NOW occupies fewer CPU and flash resources than traditional connection protocols while co-exists with Wi-Fi and Bluetooth LE.

Bluetooth is used to connect short-range devices for sharing information, while Wi-Fi is used for providing high-speed internet access<sup>2</sup>. Wi-Fi provides high bandwidth because the speed of the internet is an important issue.

### Max Distance:

The range of ESP-NOW is up to 480 meters when using the ESP-NOW protocol for bridging between multiple ESP32s<sup>1</sup>. The range can be further increased by enabling long-range ESP-NOW. When enabled, the PHY rate of ESP32 will be 512Kbps or 256Kbps.

### Maximum nodes:

ESP-NOW supports various series of Espressif chips, providing a flexible data transmission that is suitable for connecting “one-to-many” and “many-to-many” devices.

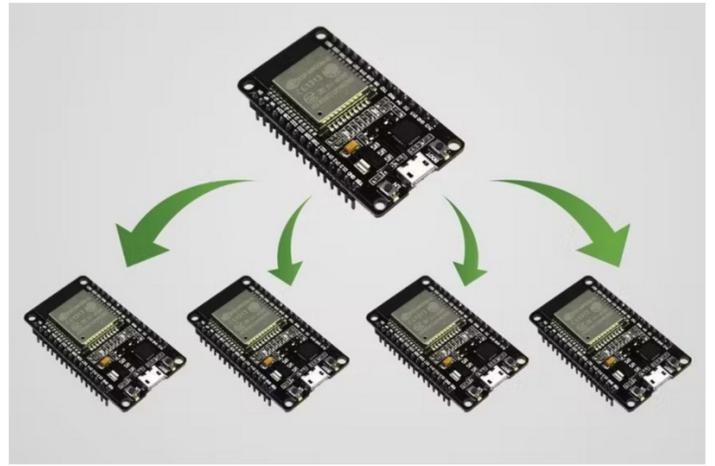
### Applications:

ESP-NOW is widely used in

- smart-home appliances,
- remote controlling,
- sensors, etc.

In this tutorial, will see how to implement a basic ESP NOW communication between ESP32 Microcontrollers.

---



## Étape 3 - Setting Up the Receiver

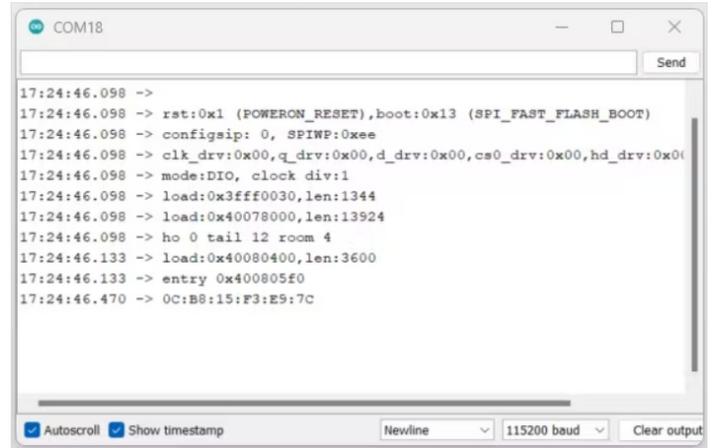
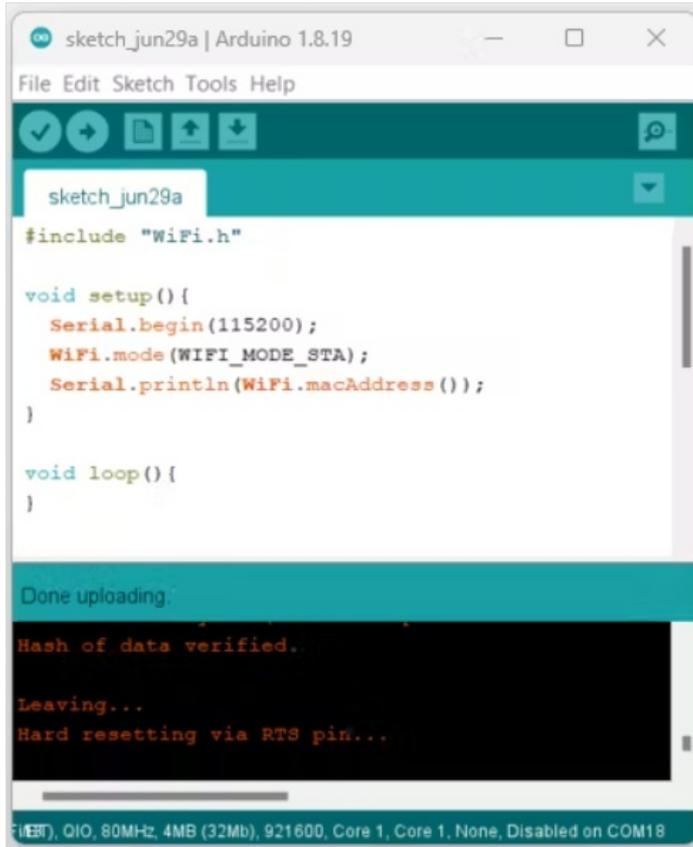
ESPNOW communication works based on the MAC address of the nodes. So, we need to find the Mac address of our slave or receiver node. ]For that just upload the following sketch to the ESP32 board and look for the Mac address in the serial monitor.

```
#include "WiFi.h"

void setup(){
  Serial.begin(115200);
  WiFi.mode(WIFI_MODE_STA);
  Serial.println(WiFi.macAddress());
}

void loop(){
}
```

Once you uploaded the code, press the EN button and wait for the serial monitor results. It will show you the Mac address. Note that.



## Étape 4 - Setting Up the Transmitter

Next, we need to prepare the transmitter, for that use this example sketch which can send multiple data types of data to the particular slave node.

```
// Register peer
memcpy(peerInfo.peer_addr, broadcastAddress, 6);
peerInfo.channel = 0;
peerInfo.encrypt = false;

// Add peer
if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
}

void loop() {
    // Set values to send
    strcpy(myData.a, "I'm alive");
    myData.b = random(1,20);
    myData.c = 1.2;
    myData.d = false;

    // Send message via ESP-NOW
    esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));

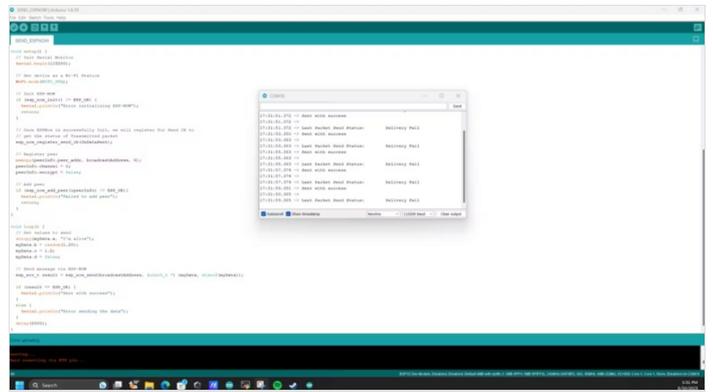
    if (result == ESP_OK) {
        Serial.println("Sent with success");
    }
    else {
        Serial.println("Error sending the data");
    }
    delay(2000);
}
```

Note: Change the Mac Address here

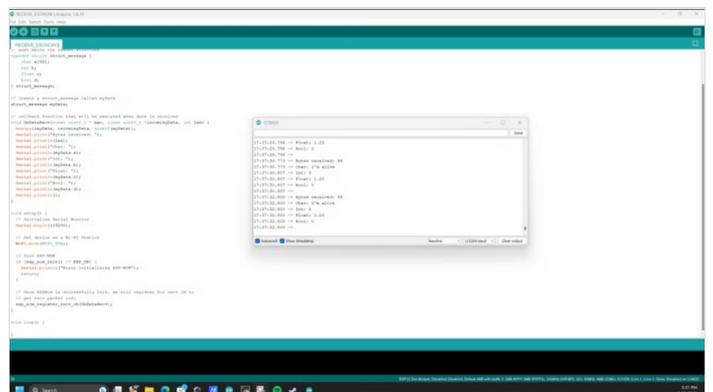
Here are the serial monitor results, it show sent success but not delivered. Because we don't have the receiver. Let's try to implement the receiver.

## Étape 5 - Testing the Connection

With the help of below example sketch, you can receive the data from the master and it will print that into the serial monitor.



The screenshot shows an IDE window with a code editor containing the transmitter code from the previous block. A serial monitor window is open, showing the output: "Sent with success". The code in the editor is the same as the code block above.



The screenshot shows an IDE window with a code editor containing the receiver code. A serial monitor window is open, showing the output: "Received data: I'm alive". The code in the editor is the receiver code described in the text below.

```

#include <esp_now.h>
#include <WiFi.h>

// Structure example to receive data
typedef struct struct_message {
    char a[32];
    int b;
    float c;
    bool d;
} struct_message;

// Create a struct_message called myData
struct_message myData;

// callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
    memcpy(&myData, incomingData, sizeof(myData));
    Serial.print("Bytes received: ");
    Serial.println(len);
    Serial.print("Char: ");
    Serial.println(myData.a);
    Serial.print("Int: ");
    Serial.println(myData.b);
    Serial.print("Float: ");
    Serial.println(myData.c);
    Serial.print("Bool: ");
    Serial.println(myData.d);
    Serial.println();
}

void setup() {
    // Initialize Serial Monitor
    Serial.begin(115200);

    // Set device as a Wi-Fi Station
    WiFi.mode(WIFI_STA);

    // Init ESP-NOW
    if (esp_now_init() != ESP_OK) {
        Serial.println("Error initializing ESP-NOW");
        return;
    }

    // get recv packer info
    esp_now_register_recv_cb(OnDataRecv);
}

void loop() {
}

```

Serial monitor results.

---

## Étape 6 - Wrap Up:

We have seen how to implement the ESP NOW in ESP32 microcontroller, in upcoming tutorials will see how to transmit sensor data via ESPNOW.

---