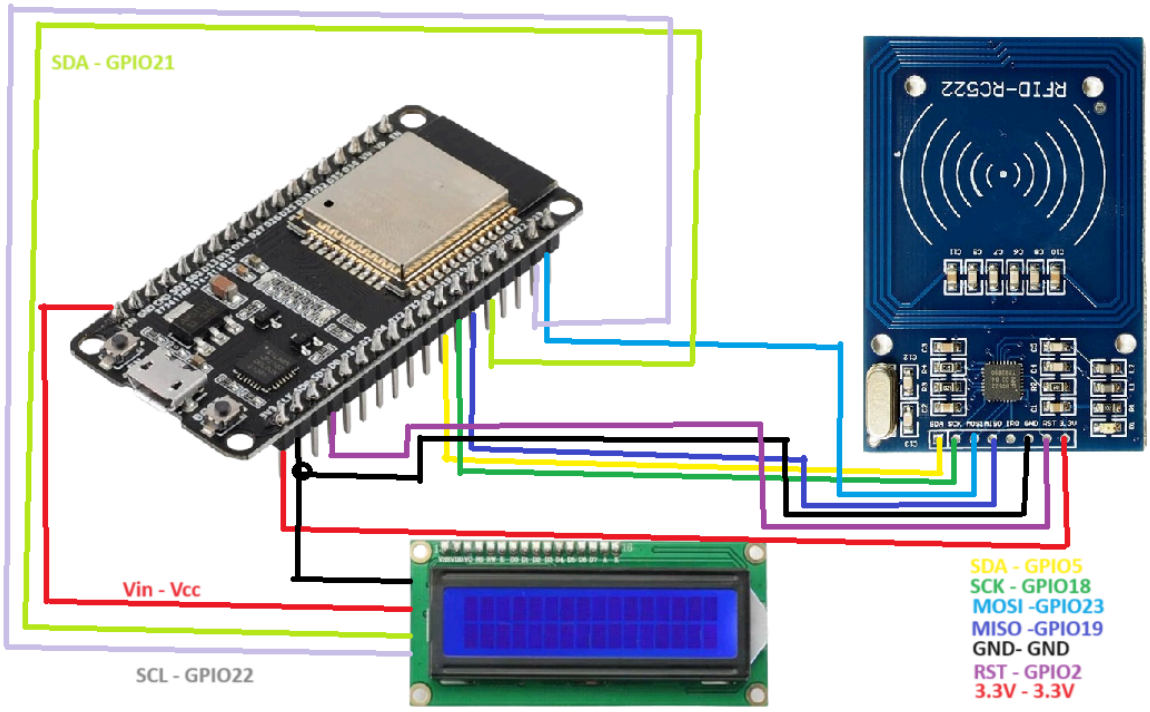
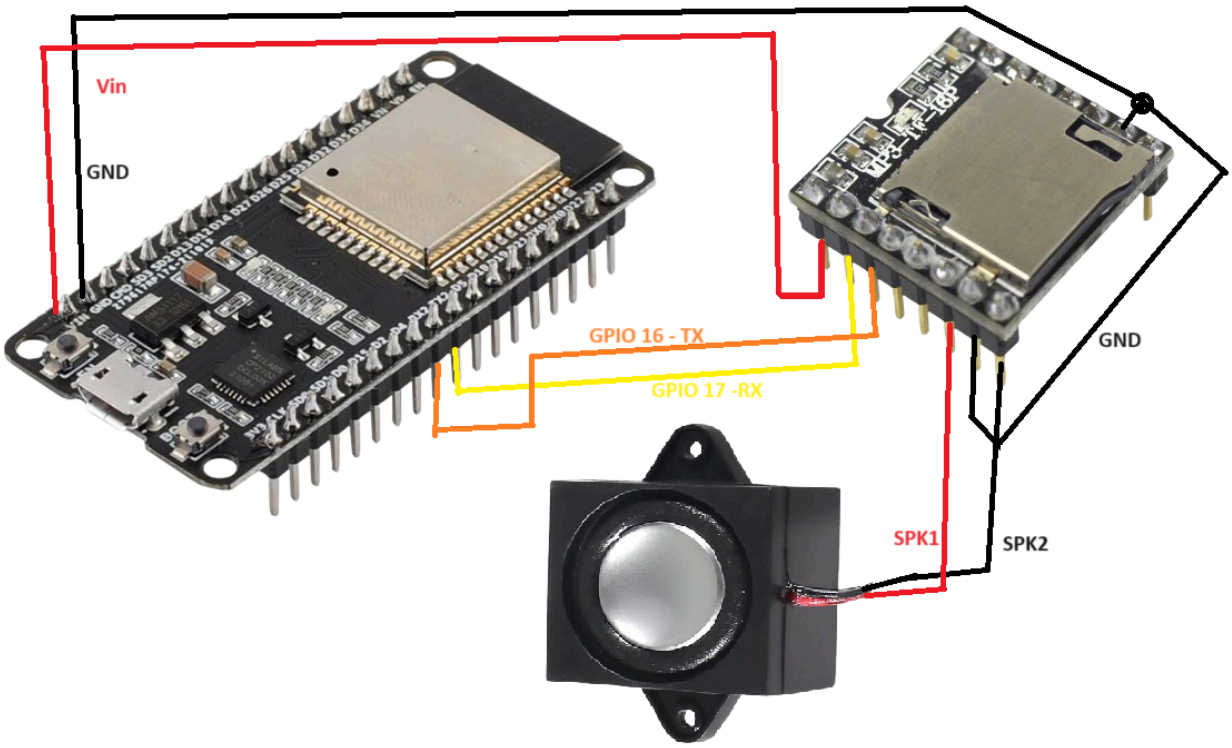
DIY Wireless RFID Module with ESP32 | ESP-Now & DFPlayer Mini Integration

☐ **Transmitter circuit diagram**



☐ **Receiver circuit diagram**



☐ Wireless Transmitter code

```
void setup() {

  // put your setup code here, to run once:


}
```

```
void loop() {#include <SPI.h>                    // Include the SPI library
for communication with RFID

#include <MFRC522.h>              // Include the MFRC522 library for RFID
reader module

#include <Wire.h>                 // Include the Wire library for I2C
communication

#include <esp_now.h>              // Include ESP-NOW library for wireless
communication
```

```
#include <WiFi.h>                 // Include Wi-Fi library for ESP32

// Define RFID module pins
#define SS_PIN 5                  // Set Slave Select pin for RFID module
#define RST_PIN 2                 // Set Reset pin for RFID module

// Replace this with your receiver's MAC address
uint8_t broadcastAddress[] = {0x70, 0xB8, 0xF6, 0x5B, 0xED, 0xB0}; // MAC address of the receiver

// Define the structure of the data packet to be sent
struct __attribute__((packed)) dataPacket {
  bool state;                     // A boolean variable to indicate the card authorization state
};

// Create a variable to store peer information
esp_now_peer_info_t peerInfo;

// Callback function to handle the result of data transmission
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery Fail");
```

```
}


// Create an instance of the MFRC522 class with the specified SS and RST
pins

MFRC522 rfid(SS_PIN, RST_PIN);


// Predefined UID for the authorized RFID card

byte authorizedUID[] = {0x0B, 0x23, 0x9B, 0x15}; // Example UID; replace
with your actual card's UID


void setup() {

  Serial.begin(9600);              // Start serial communication at a baud
rate of 9600


  SPI.begin();                     // Initialize the SPI bus for RFID
communication

  rfid.PCD_Init();                 // Initialize the MFRC522 RFID reader
module


  WiFi.mode(WIFI_STA);             // Set the device to Wi-Fi Station mode
(required for ESP-NOW)


  // Initialize ESP-NOW communication

  if (esp_now_init() != ESP_OK) {

    Serial.println("Error initializing ESP-NOW");
```

```
    return;                         // If initialization fails, exit the
setup
  }

  esp_now_register_send_cb(OnDataSent); // Register the callback function
to track the status of data sent

  // Set up the peer information for ESP-NOW communication
  memcpy(peerInfo.peer_addr, broadcastAddress, 6); // Copy the receiver's
MAC address
  peerInfo.channel = 0;                             // Set the
communication channel to 0
  peerInfo.encrypt = false;                         // Disable encryption
for this example

  // Add the receiver as a peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Failed to add peer");           // Print an error if
adding the peer fails
    return;
  }
}

void loop() {
  // Check if a new RFID card is present and read its UID
  if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial()) {
```

```
    return;                       // Exit the loop if no new card is
detected
  }

  Serial.print("UID tag: ");     // Print the detected card's UID
  String content = "";           // Initialize a string to store the UID
  for (byte i = 0; i < rfid.uid.size; i++) {
    Serial.print(rfid.uid.uidByte[i] < 0x10 ? " 0" : " ");
    Serial.print(rfid.uid.uidByte[i], HEX); // Print each byte of the UID
in hexadecimal format
    content.concat(String(rfid.uid.uidByte[i] < 0x10 ? " 0" : " "));
    content.concat(String(rfid.uid.uidByte[i], HEX)); // Append each byte
to the string
  }
  Serial.println();
  Serial.println();

  // Check if the detected UID matches the authorized UID
  bool authorized = true;
  for (byte i = 0; i < rfid.uid.size; i++) {
    if (rfid.uid.uidByte[i] != authorizedUID[i]) {
      authorized = false;         // Set authorized to false if any byte
doesn't match
      break;
    }
```

```
  }

  dataPacket packet;              // Create an instance of the dataPacket
struct

  if (authorized) {
    packet.state = 0;             // Set state to 0 for authorized UID
    // lcd.print("Welcome Home     "); // (Optional) Display message on
LCD for authorized card
  } else {
    packet.state = 1;             // Set state to 1 for unauthorized UID
    // lcd.print("Wrong identity..."); // (Optional) Display message on
LCD for unauthorized card
  }

  // Send the packet to the receiver using ESP-NOW
  esp_now_send(broadcastAddress, (uint8_t *) &packet, sizeof(packet));
  Serial.println(packet.state);  // Print the state of the packet (0 or 1)

  delay(30);                      // Short delay to prevent rapid
re-triggering

  rfid.PICC_HaltA();              // Halt the communication with the RFID
card
}
```

```
  // put your main code here, to run repeatedly:


}
```

Explanation Summary:

1. **Header files:** Libraries for RFID, I2C, Wi-Fi, and ESP-NOW are included.
2. **Pin definitions:** Defines SS and RST pins for the RFID module.
3. **MAC address:** Replace with the actual MAC address of the receiver device.
4. **Data packet:** A structure is defined to send a boolean value indicating authorization status.
5. **ESP-NOW setup:** Initializes ESP-NOW and registers the receiver as a peer.
6. **RFID processing:** Checks for a new RFID card, reads its UID, and compares it with a predefined UID.
7. **Packet transmission:** Sends a packet indicating whether the card is authorized or not.
8. **Callback function:** Reports the status of the data transmission.

☐ Wireless Receiver Arduino code

```
#include <esp_now.h>

#include <WiFi.h>

#include "Arduino.h" // Include the core Arduino library

#include "DFRobotDFPlayerMini.h" // Include the DFRobot DFPlayer Mini library


#ifdef ESP32

  #define FPSerial Serial1  // For ESP32, use hardware serial port 1
```

```
#else
  #include <SoftwareSerial.h> // Include SoftwareSerial library for non-ESP32 boards
  SoftwareSerial FPSerial(26, 27); // Define SoftwareSerial on pins GPIO26 (RX) and GPIO27 (TX)
#endif

DFRobotDFPlayerMini myDFPlayer; // Create an instance of the DFRobotDFPlayerMini class

// Structure to hold received data
struct __attribute__((packed)) dataPacket {
  bool state; // State variable (on/off)
};

// Variables to handle timing and state
unsigned long previousMillis = 0;
const long interval = 1000; // Interval to wait (1 second)
bool playStarted = false; // Track if the play command was issued
bool dataReceived = false; // Track if data was received
dataPacket receivedPacket; // Store the received packet

// Callback function that executes when data is received via ESP-NOW
void OnDataRecv(const esp_now_recv_info* info, const uint8_t *incomingData, int len) {
```

```
  memcpy(&receivedPacket, incomingData, sizeof(receivedPacket)); // Copy received data into receivedPacket
  Serial.print("button: ");
  Serial.println(receivedPacket.state); // Print the state to Serial Monitor
  dataReceived = true;  // Set flag to indicate data was received
}

void setup() {
  // Initialize Serial communication for DFPlayer Mini
  #ifdef ESP32
  FPSerial.begin(9600, SERIAL_8N1, 26, 27); // For ESP32, use hardware serial on pins 26 (RX) and 27 (TX)
  #else
  FPSerial.begin(9600); // For other boards, use SoftwareSerial at 9600 baud rate
  #endif

  Serial.begin(115200); // Start the Serial monitor at 115200 baud
  Serial.println(F("DFRobot DFPlayer Mini Demo"));
  Serial.println(F("Initializing DFPlayer ... (May take 3~5 seconds)"));

  // Initialize DFPlayer Mini
  if (!myDFPlayer.begin(FPSerial)) {
    Serial.println(F("Unable to begin:"));
```

```
    Serial.println(F("1. Please recheck the connection!"));
    Serial.println(F("2. Please insert the SD card!"));
    while (true); // Stop execution if initialization fails
  }
  Serial.println(F("DFPlayer Mini online."));

  myDFPlayer.volume(30);  // Set the volume to maximum (30)

  // Set the ESP32 to Wi-Fi station mode
  WiFi.mode(WIFI_STA);

  // Initialize ESP-NOW and check for errors
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Register callback function for receiving data
  esp_now_register_recv_cb(OnDataRecv);

  myDFPlayer.play(1); // Play the first track at startup
}
```

```
void loop() {
  // Check if data was received
  if (dataReceived) {
    dataReceived = false; // Reset the flag

    if (receivedPacket.state == 0) {
      // If state is 0, play the second track
      playStarted = true; // Set playStarted to true
      previousMillis = millis(); // Record the current time
      myDFPlayer.play(2); // Play track 2
    } else {
      // If state is not 0, play the first track
      playStarted = true; // Set playStarted to true
      previousMillis = millis(); // Record the current time
      myDFPlayer.play(1); // Play track 1
    }
  }

  // Check if playStarted is true and if 1 second has passed
  if (playStarted && (millis() - previousMillis >= interval)) {
    playStarted = false; // Reset the flag after 1 second
    // You can add additional actions here if needed after the delay
  }
```

}

Key Features:

1. **ESP-NOW Communication:**
   The code uses ESP-NOW for wireless communication, receiving a simple `bool` state (on/off).
2. **DFPlayer Mini Control:**
   Depending on the received state:
   - Track 1 plays when the state is `1`.
   - Track 2 plays when the state is `0`.
3. **Timing Logic:**
   The `millis()` function is used to introduce a delay of 1 second after each play command to prevent rapid triggering.

☐ You cand Find the code and the circuit diagrams on github:

https://github.com/Valentim-bot/DIY-Wireless-RFID-Module-with-ESP32-ESP-Now-DFPlayer-Mini-Integration

How to put the audios inside the sd card?

1- Your micro SD card should support FAT16 and FAT32 file systems

2 - Ordinary folders must be renamed as 01, 02, 03.....99, and the audio files must be renamed as
001.mp3/001.wav, 002.mp3/002.wav, 003.mp3/003.wav, ...255.mp3/255.wav. It is also possible to keep the
original name when you rename a file. For example, the original name is “Yesterday Once More.mp3”, then you can
rename it as “001Yesterday Once More.mp3”.