

```

/*
This program serves to read the output of a KY-023 joystick using an ESP-32
microcontroller and transmit it wirelessly to a second receiving ESP-32
microcontroller using the ESP-NOW wireless protocol.
This program was produced by Mr. Kwek E.Y. (MOE, Singapore) by modifying code
provided by:

    Rui Santos & Sara Santos - Random Nerd Tutorials
    Complete project details at
https://RandomNerdTutorials.com/esp-now-esp32-arduino-ide/
    Permission is hereby granted, free of charge, to any person obtaining a copy of
    this software and associated documentation files.
    The above copyright notice and this permission notice shall be included in all
    copies or substantial portions of the Software.
*/
#include <esp_now.h>
#include <WiFi.h>

const int SW_pin = 13; // Joystick button is connected to pin 13 of ESP32 (not
used here)
const int X_pin = 32; // X-output of the joystick is connected to pin 32 of
ESP32.
const int Y_pin = 34; // Y-output of the joystick is connected to pin 34 of
ESP32.

int x;
int y;

// REPLACE THE NUMBERS IN RED WITH YOUR RECEIVER'S UNIQUE MAC Address:
uint8_t broadcastAddress[] = {0x70, 0xb8, 0xf6, 0x5b, 0xf6, 0x54};

// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
    char a[32];    // Test data (not used in this project).
    int b;         // Test data (not used in this project).
    float c;       // Test data (not used in this project).
    bool d;        // Test data (not used in this project).
    int x;         // This stores the x-output of the joystick for transmission.

```

```

    int y;           // This stores the y-output of the joystick for transmission
} struct_message;

// Create a struct_message called myData
struct_message myData;

esp_now_peer_info_t peerInfo;

// callback when data is sent
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");
}

void setup() {
    pinMode(X_pin, INPUT);
    pinMode(Y_pin, INPUT);
    pinMode(SW_pin, INPUT_PULLUP);

    // Init 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;
    }

    // Once ESPNow is successfully Init, we will register for Send CB to
    // get the status of Trasnmitted packet
    esp_now_register_send_cb(esp_now_send_cb_t(OnDataSent));

    // 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() {

    x = analogRead(X_pin);        // Read the x-axis output of the joystick.
    y = analogRead(Y_pin);        // Read the y-axis output of the joystick.

    // Show the x and y outputs of the joysticks on the Serial Monitor of the IDE.
    Serial.print("X-axis: ");
    Serial.print(x);
    Serial.print("    Y-axis: ");
    Serial.println(y);

    // Set values to send
    strcpy(myData.a, "THIS IS A CHAR");        // Test data (not important to
project).
    myData.b = random(1,20);                    // Test data (not important to
project).
    myData.c = 1.2;                             // Test data (not important to
project).
    myData.d = false;                           // Test data (not important to
project).
    myData.x = x;
    myData.y = y;

    // 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(200);    // Delay for 0.200 seconds between transmissions. Feel free to  
shorten this if it makes your robot more responsive, but check that the code  
still works.  
  
}
```