Nodo_HumedadSuelo_Doxy

AUTHOR
Versión 1.0.0

Índice de clases

Lista de clases

Lista de clases, estructuras, uniones e interfaces con breves descripciones:

Índice de archivos

Lista de archivos

Lista de todos los archivos con breves descripciones:

Documentación de clases

Referencia de la estructura soil_message_t

Estructura de datos que contiene el porcentaje de humedad del suelo.

Atributos públicos

- float **soil**

Descripción detallada

Estructura de datos que contiene el porcentaje de humedad del suelo.

El valor se calcula mapeando la lectura analógica cruda a un rango 0-100%%.

Definición en la línea **36** del archivo **Emisor_HumedadSuelo.ino**.

Documentación de datos miembro

float soil_message_t::soil

Humedad de suelo en porcentaje

Definición en la línea **37** del archivo **Emisor_HumedadSuelo.ino**.

La documentación de esta estructura está generada del siguiente archivo:

- C:/Users/Andres Fernandez/Documents/Arduino/sketch_may7a/Nodo_HumedadSuelo/Emisor_HumedadSuelo/**Emisor_HumedadSuelo.ino**

Documentación de archivos

Referencia del archivo C:/Users/Andres Fernandez/Documents/Arduino/sketch_may7a/Nodo_HumedadSuelo/Emisor_HumedadSuelo/Emisor_HumedadSuelo.ino

```
#include <esp_now.h>
#include <WiFi.h>
#include "esp_wifi.h"
```

Clases

struct **soil_message_t***Estructura de datos que contiene el porcentaje de humedad del suelo.*

defines

- #define **SOIL_PIN** 34
  Pin analógico conectado al sensor de humedad de suelo.

- #define **VALOR_SECO** 2590
  Valor de sensor correspondiente a suelo completamente seco.

- #define **VALOR_MOJADO** 1200
  Valor de sensor correspondiente a suelo completamente mojado.

Funciones

- void **OnDataSent** (const uint8_t *mac_addr, esp_now_send_status_t status)
  Callback llamado tras enviar un paquete ESP-NOW.

- void **SoilSensorTask** (void *parameter)
  Tarea que lee el sensor de humedad de suelo y envía los datos periódicamente.

- void **setup** ()
  Función de configuración inicial.

- void **loop** ()
  Bucle principal (no utilizado en este diseño multihilo).

Variables

- const char * **ssid** = "iPhone"
  Credenciales WiFi para sincronizar canal y obtener la configuración de WiFi.

- const char * **password** = "25310tati"
- uint8_t **receiverMAC** [] = {0x5C, 0x01, 0x3B, 0x72, 0xF2, 0xCC}
  Dirección MAC del nodo receptor al que se envían los datos.

Documentación de «define»

#define SOIL_PIN 34

Pin analógico conectado al sensor de humedad de suelo.

Definición en la línea **19** del archivo **Emisor_HumedadSuelo.ino**.

#define VALOR_MOJADO 1200

Valor de sensor correspondiente a suelo completamente mojado.

Definición en la línea **25** del archivo **Emisor_HumedadSuelo.ino**.

#define VALOR_SECO 2590

Valor de sensor correspondiente a suelo completamente seco.

Definición en la línea **22** del archivo **Emisor_HumedadSuelo.ino**.

Documentación de funciones

void loop ()

Bucle principal (no utilizado en este diseño multihilo).

Definición en la línea **148** del archivo **Emisor_HumedadSuelo.ino**.

void OnDataSent (const uint8_t * mac_addr, esp_now_send_status_t status)

Callback llamado tras enviar un paquete ESP-NOW.

Imprime en Serial si el envío fue exitoso o falló, mostrando el código de error.

Parámetros

mac_addr	Dirección MAC del peer destino
status	Estado del envío (ESP_NOW_SEND_SUCCESS o fallo)

Definición en la línea **51** del archivo **Emisor_HumedadSuelo.ino**.

void setup ()

Función de configuración inicial.

- Inicia Serial para depuración.
- Configura WiFi en modo estación y sincroniza canal.
- Inicia ESP-NOW y registra callback de envío.
- Agrega el peer receptor.
- Crea la tarea SoilSensorTask en el Core 1.

Definición en la línea **101** del archivo **Emisor_HumedadSuelo.ino**.

void SoilSensorTask (void * parameter)

Tarea que lee el sensor de humedad de suelo y envía los datos periódicamente.

- Lee el valor crudo del pin analógico.
- Convierte la lectura a porcentaje entre 0 y 100%%.
- Envía la estructura **soil_message_t** al nodo receptor.
- Espera 10 segundos antes de la siguiente lectura.

Parámetros

parameter	Parámetro de tarea no usado

Definición en la línea **68** del archivo **Emisor_HumedadSuelo.ino**.

Documentación de variables

const char* password = "25310tati"

Definición en la línea **16** del archivo **Emisor_HumedadSuelo.ino**.

uint8_t receiverMAC[] = {0x5C, 0x01, 0x3B, 0x72, 0xF2, 0xCC}

Dirección MAC del nodo receptor al que se envían los datos.

Definición en la línea **28** del archivo **Emisor_HumedadSuelo.ino**.

const char* ssid = "iPhone"

Credenciales WiFi para sincronizar canal y obtener la configuración de WiFi.

Definición en la línea **15** del archivo **Emisor_HumedadSuelo.ino**.

Emisor_HumedadSuelo.ino

Ir a la documentación de este archivo.

```
00001
00009
00010 #include <esp_now.h>
00011 #include <WiFi.h>
00012 #include "esp_wifi.h"
00013
00015 const char* ssid     = "iPhone";
00016 const char* password = "25310tati";
00017
00019 #define SOIL_PIN     34
00020
00022 #define VALOR_SECO   2590
00023
00025 #define VALOR_MOJADO 1200
00026
00028 uint8_t receiverMAC[] = {0x5C, 0x01, 0x3B, 0x72, 0xF2, 0xCC};
00029
00036 typedef struct {
00037   float soil;
00038 } soil_message_t;
00039
00041 static soil_message_t sensorSend;
00042
00051 void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
00052   Serial.printf("[ESP-NOW] Envío a %02X:%02X:%02X:%02X:%02X:%02X -> %s\n",
00053                 mac_addr[0], mac_addr[1], mac_addr[2],
00054                 mac_addr[3], mac_addr[4], mac_addr[5],
00055                 status == ESP_NOW_SEND_SUCCESS ? "OK" : "FALLÓ");
00056 }
00057
00068 void SoilSensorTask(void *parameter) {
00069   (void) parameter; // Evitar advertencia de parámetro no usado
00070
00071   for (;;) {
00072     int raw = analogRead(SOIL_PIN);
00073     // Mapear valor crudo a porcentaje
00074     int pct = map(raw, VALOR_SECO, VALOR_MOJADO, 0, 100);
00075     pct = constrain(pct, 0, 100);
00076
00077     // Imprimir valores en Serial
00078     Serial.printf("[Suelo] RAW=%4d -> %3d%%\n", raw, pct);
00079
00080     // Preparar mensaje
00081     sensorSend.soil = pct;
00082     esp_err_t res = esp_now_send(receiverMAC, (uint8_t *)&sensorSend,
sizeof(sensorSend));
00083     if (res != ESP_OK) {
00084       Serial.printf("[Error] Fallo al enviar datos (err %d)\n", res);
00085     }
00086
00087     // Esperar 10 segundos antes de la próxima lectura
00088     vTaskDelay(pdMS_TO_TICKS(10000));
00089   }
00090 }
00091
00101 void setup() {
00102   // Iniciar Serial
00103   Serial.begin(115200);
00104   delay(100);
00105
00106   // Modo estación WiFi y mostrar MAC
00107   WiFi.mode(WIFI_STA);
00108   Serial.printf("[Setup] MAC STA: %s\n", WiFi.macAddress().c_str());
00109
00110   // Configurar canal para ESP-NOW
00111   uint8_t canal = WiFi.channel();
00112   esp_wifi_set_channel(canal, WIFI_SECOND_CHAN_NONE);
00113   Serial.printf("[Setup] Canal WiFi: %d\n", canal);
00114
00115   // Iniciar ESP-NOW
```

```
00116   if (esp_now_init() != ESP_OK) {
00117     Serial.println("[Error] No se pudo iniciar ESP-NOW");
00118     return;
00119   }
00120   // Registrar callback
00121   esp_now_register_send_cb(OnDataSent);
00122
00123   // Configurar peer receptor
00124   esp_now_peer_info_t peerInfo = {};
00125   memcpy(peerInfo.peer_addr, receiverMAC, 6);
00126   peerInfo.channel = canal;
00127   peerInfo.encrypt = false;
00128   if (esp_now_add_peer(&peerInfo) != ESP_OK) {
00129     Serial.println("[Error] Fallo al agregar peer receptor");
00130     return;
00131   }
00132
00133   // Crear tarea de sensor de suelo en Core 1
00134   xTaskCreatePinnedToCore(
00135     SoilSensorTask,     /* Función de la tarea */
00136     "SoilSensorTask",  /* Nombre de la tarea */
00137     2048,               /* Tamaño de stack */
00138     NULL,               /* Parámetro */
00139     1,                  /* Prioridad */
00140     NULL,               /* Handle */
00141     1                   /* Núcleo */
00142   );
00143 }
00144
00148 void loop() {
00149   vTaskDelay(portMAX_DELAY);
00150 }
```