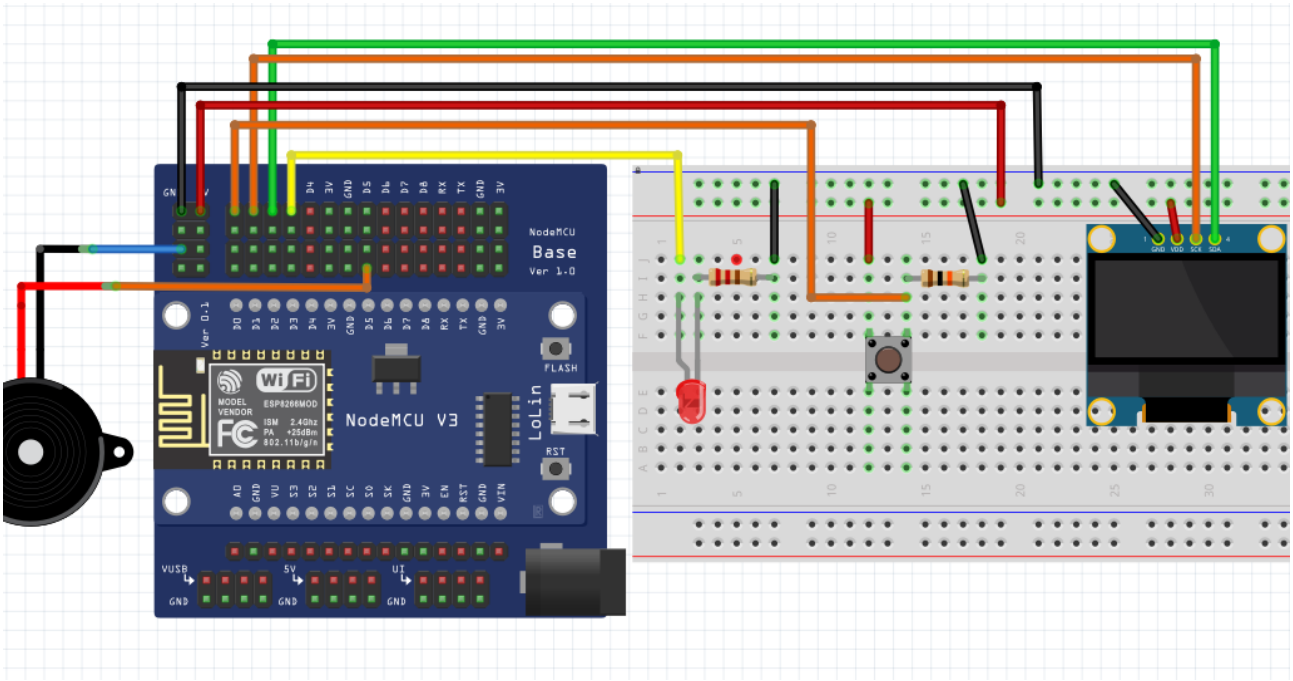
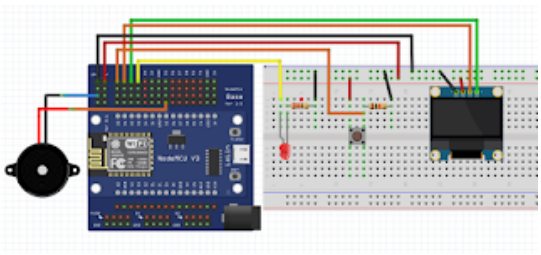
# NodeMCU: 09_ESPNOW (Comunicación bidireccional) Mejorando la central. FASE 1

agrportfolioeducativo.blogspot.com/2020/04/nodemcu-09espnow-comunicacion.html




## ESQUEMA

La central conserva el mismo esquema que la estación.



## PROGRAMA

```
/* Este ejemplo intenta recopilar lo aprendido de esquemas anteriores
* y comunicará dos NodeMCU station "hablando" entre ellos.
* La acción en uno, establecerá una reacción en el otro.
*/

#include <ESP8266WiFi.h>
extern "C" {
#include <espnow.h>
}
```

```
//***ESTRUCTURA DE LOS DATOS TRANSMITIDOS ENTRE LAS UNIDADES***//
// Se establecerá IGUAL en todas
struct ESTRUCTURA_DATOS {
int estado; // 1 --> operativo, 2 --> activo
String estacion;
};


// variables de elementos hardware
int entrada = D0; // pulsador
int salida = D1; // LED
int pulsado = 0;
int pulsadoAntes = 0;

ESTRUCTURA_DATOS ED;

// *****
// SETUP
// *****

void setup() {

//***INICIALIZACIÓN DEL PUERTO SERIE***//
Serial.begin(115200); Serial.println();Serial.println();

//*** pinMode ***
pinMode(entrada,INPUT);
pinMode(salida,OUTPUT);

//***INICIALIZACIÓN DEL PROTOCOLO ESP-NOW***//
if (esp_now_init()!=0) {
Serial.println("*** ESP_Now init failed");
ESP.restart();
delay(1);
}

//***DATOS DE LAS MAC (Access Point y Station) del ESP***//
// Serial.print("Access Point MAC de este ESP: "); Serial.println(WiFi.softAPmacAddress());
// Serial.print("Station MAC de este ESP: "); Serial.println(WiFi.macAddress());
```

```
//***DECLARACIÓN DEL PAPEL DEL DISPOSITIVO ESP EN LA COMUNICACIÓN***//
//0=OCIOSO, 1=MAESTRO, 2=ESCLAVO y 3=MAESTRO+ESCLAVO
esp_now_set_self_role(3); // Sería como maestro y esclavo a la vez

//***EMPAREJAMIENTO CON EL ESCLAVO***//
// Dirección MAC del ESP con el que se empareja (esclavo)
// Se debe introducir la STA MAC correspondiente
uint8_t mac_addr[6] = {0xA4, 0xCF, 0x12, 0xDF, 0x5A, 0x6B}; // STA MAC esclavo
uint8_t role=2;
uint8_t channel=3;
uint8_t key[0]={}; //no hay clave
//uint8_t key[16] = {0,255,1,1,1,1,1,1,1,1,1,1,1,1,1,1};
uint8_t key_len=sizeof(key);
// Serial.print("Tamaño de *key: "); Serial.println(key_len);
esp_now_add_peer(mac_addr,role,channel,key,key_len);
ED.estacion="C";
}

// *******
// LOOP
// *******
void loop() {

pulsado = digitalRead(entrada);

// ---------------------------------------------------------------------------
// ENVÍO: como Master
// ---------------------------------------------------------------------------
//***DATOS A ENVIAR***//
if (pulsado==1 & pulsadoAntes==0) {
ED.estado = 1-ED.estado;
// Serial.print("Dato pulsado: "); Serial.println(ED.estado);
// Serial.print("Variable pulsado: "); Serial.println(pulsado);
// Serial.print("Variable pulsadoAntes: "); Serial.println(pulsadoAntes);
}
```

```
delay(20);

//***ENVÍO DE LOS DATOS***//
//uint8_t *da=NULL; //NULL envía los datos a todos los ESP con los que está emparejado
uint8_t da[6] = {0xA4, 0xCF, 0x12, 0xDF, 0x5A, 0x6B}; // ¿mismos datos que STA MAC?
uint8_t data[sizeof(ED)]; memcpy(data, &ED, sizeof(ED));
uint8_t len = sizeof(data);
esp_now_send(da, data, len);

delay(1); //Si se pierden datos en la recepción se debe subir este valor

// *** CALLBACK de datos ENVIADOS ***
// *** Verificamos que los datos se han enviado....
esp_now_register_send_cb([](uint8_t* mac, uint8_t status) {
char MACesclavo[6];
//
sprintf(MACesclavo,"%02X:%02X:%02X:%02X:%02X:%02X",mac[0],mac[1],mac[2],mac[3],mac[4],mac[5]);
// Serial.print(". Enviado a ESP MAC: "); Serial.print(MACesclavo);
// Serial.print(". Recepcion (0=0K - 1=ERROR): "); Serial.println(status);
});
// ---------------------------------------------------------------------------

// ---------------------------------------------------------------------------
// Recepción: como Esclavo
// ---------------------------------------------------------------------------
esp_now_register_recv_cb([](uint8_t *mac, uint8_t *data, uint8_t len) {
char MACmaestro[6];
// sprintf(MACmaestro,
"%02X:%02X:%02X:%02X:%02X:%02X",mac[0],mac[1],mac[2],mac[3],mac[4],mac[5]);
// Serial.print("Recepcion desde ESP MAC: "); Serial.print(MACmaestro);

ESTRUCTURA_DATOS ED2;
memcpy(&ED2, data, sizeof(ED2));

Serial.println("Recibiendo de " + ED2.estacion + " // Su estado: "+ (String) ED2.estado);
digitalWrite(salida,ED2.estado-1); // -1 porque recibe un dos cuando está ACTIVADO
});
// ---------------------------------------------------------------------------
```

```
    pulsadoAntes=pulsado;

    }
```

view raw NodeMCU_BOTON_PANICO_CENTRAL.ino hosted with ❤ by GitHub

Por ahora, el programa de la central recibe los datos y los muestra por el monitor serie. Deberá ser capaz, en próximas versiones, de detectar la estación que se lo envía (más de una), mostrar su estado en pantalla.

Además envía una señal de control a la estación. En próximas versiones, esa señal de control deberá apagar la alarma de la misma y volver su estado a espera.