# Proyecto: Detección de Ruido con Semáforo y Sensor KY-038 usando ESP32

Autores: Mario Andrés Testa, Juan David Pineda, Juan David Alfonso Maussa

## 1. Introducción

Este proyecto consiste en crear un sistema de detección de ruido usando un sensor de sonido KY-038 y un ESP32, con el objetivo de monitorear el nivel de ruido en un ambiente y reflejar el estado del mismo a través de un semáforo de dos colores (verde y rojo). El semáforo cambiará de verde a rojo cuando el sensor registre niveles elevados de ruido durante un periodo de 5 a 10 minutos de manera continua. El estado del semáforo también se podrá visualizar en tiempo real a través de una página web conectada mediante WebSocket, lo que facilita la supervisión en entornos ruidosos.

## 2. Objetivos del proyecto

1. Detectar ruido en tiempo real: Utilizar el sensor KY-038 para captar variaciones de sonido en el ambiente. El ESP32 monitoreará continuamente los niveles de sonido.  
2. Cambiar el estado del semáforo: Cada vez que el ruido detectado persista por más de 5 minutos, el semáforo cambiará de verde a rojo. Si los niveles de ruido bajan, el semáforo volverá a verde.  
3. Visualizar el estado en tiempo real: Implementar una conexión WebSocket para reflejar el estado del semáforo en una interfaz web en tiempo real, brindando una herramienta de monitoreo visual.  
4. Aplicación en ambientes ruidosos: Este sistema es ideal para ambientes donde el ruido excesivo es un problema, como en aulas, oficinas o espacios públicos, permitiendo monitorear y tomar medidas preventivas.

## 3. Funcionamiento del proyecto

El sistema de monitoreo de ruido se compone de un sensor de sonido KY-038 conectado a un ESP32 que mide la presencia de ruido de manera continua. El sistema se comporta de la siguiente manera:  
- Captura de sonido: El sensor KY-038 detecta cambios en las ondas sonoras del ambiente. Este sensor puede percibir si el nivel de sonido supera un umbral determinado, indicando ruido en el ambiente.  
- Monitoreo continuo: El ESP32 verifica el estado del sensor en cada momento. Cuando el sensor detecta ruido, el ESP32 acumula el tiempo de duración de ese sonido. Si el nivel de ruido es constante por más de 5 minutos, el sistema activa el semáforo en color rojo.  
- Cambio de estado: Cuando el ruido persiste durante 5 minutos o más, el semáforo cambia de verde a rojo, indicando que se ha alcanzado un nivel de ruido sostenido. Si el ruido disminuye, el semáforo vuelve a verde.  
- Visualización en la interfaz web: El ESP32 envía el estado del semáforo a través de WebSocket a una página web, permitiendo el monitoreo remoto en tiempo real.

## 4. Aplicaciones del proyecto

Este sistema es útil en situaciones donde es importante controlar el nivel de ruido, tales como:  
- Aulas y bibliotecas: Para mantener un ambiente adecuado para el estudio.  
- Lugares públicos: Para monitorear y regular el ruido en eventos o espacios concurridos.

## 5. Materiales necesarios

- ESP32  
- Sensor de sonido KY-038  
- LEDs (uno verde y otro rojo)  
- Resistencias para los LEDs  
- Computadora con Arduino IDE instalado

## 6. Diagrama de conexión

- Conectar el pin de salida analógico del sensor KY-038 a un pin GPIO del ESP32 (por ejemplo, el GPIO 34).  
- Conectar el LED verde a otro GPIO (por ejemplo, GPIO 18) y el LED rojo a otro (por ejemplo, GPIO 19) con sus respectivas resistencias.

## 7. Código del ESP32

El código permite que el ESP32 se conecte a la red Wi-Fi, lea el nivel de ruido del sensor y, si el ruido es continuo durante 5 minutos, cambia el semáforo de verde a rojo. El estado del semáforo también se envía a una página web usando WebSocket.  
  
```cpp  
#include <WiFi.h>  
#include <WebSocketsServer.h>  
  
const char\* ssid = "TU\_SSID";  
const char\* password = "TU\_CONTRASEÑA";  
  
WebSocketsServer webSocket = WebSocketsServer(81);  
  
const int soundSensorPin = 34; // Pin del sensor KY-038  
const int greenLedPin = 18; // LED verde  
const int redLedPin = 19; // LED rojo  
  
unsigned long soundStartTime = 0;  
unsigned long currentMillis = 0;  
bool highSoundDetected = false;  
  
const int soundThreshold = 500; // Ajusta este valor según la sensibilidad  
const int soundDurationLimit = 300000; // 5 minutos en milisegundos  
  
void setup() {  
 Serial.begin(115200);  
 pinMode(soundSensorPin, INPUT);  
 pinMode(greenLedPin, OUTPUT);  
 pinMode(redLedPin, OUTPUT);  
  
 WiFi.begin(ssid, password);  
 while (WiFi.status() != WL\_CONNECTED) {  
 delay(500);  
 Serial.print(".");  
 }  
 Serial.println("Conectado al WiFi");  
  
 webSocket.begin();  
 webSocket.onEvent(webSocketEvent);  
  
 digitalWrite(greenLedPin, HIGH); // Empieza en verde  
 digitalWrite(redLedPin, LOW);  
}  
  
void loop() {  
 webSocket.loop();  
  
 int soundValue = analogRead(soundSensorPin);  
 currentMillis = millis();  
  
 if (soundValue > soundThreshold) {  
 if (!highSoundDetected) {  
 highSoundDetected = true;  
 soundStartTime = currentMillis;  
 } else if (currentMillis - soundStartTime >= soundDurationLimit) {  
 changeTrafficLightState(false); // Cambia a rojo  
 }  
 } else {  
 highSoundDetected = false;  
 changeTrafficLightState(true); // Mantiene en verde  
 }  
}  
  
void changeTrafficLightState(bool isGreen) {  
 if (isGreen) {  
 digitalWrite(greenLedPin, HIGH);  
 digitalWrite(redLedPin, LOW);  
 webSocket.broadcastTXT("GREEN");  
 } else {  
 digitalWrite(greenLedPin, LOW);  
 digitalWrite(redLedPin, HIGH);  
 webSocket.broadcastTXT("RED");  
 }  
}  
  
void webSocketEvent(uint8\_t num, WStype\_t type, uint8\_t\* payload, size\_t length) {  
 if (type == WStype\_TEXT) {  
 Serial.printf("[%u] Mensaje: %s\n", num, payload);  
 }  
}  
```

## 8. Conclusiones

Este proyecto demuestra cómo se puede utilizar el ESP32 con un sensor de sonido para monitorear el nivel de ruido en el ambiente y reflejar su estado a través de un semáforo visual y una interfaz web. Este tipo de sistema es aplicable en diversas áreas, como la gestión de ruido en aulas, oficinas o lugares públicos, proporcionando una herramienta práctica de monitoreo visual en tiempo real.