Práctica: ESP32 y sensor ultrasónico

Objetivo:

- Emplear hardware externo para poder ampliar las capacidades de nuestro ESP32
- Determinar distancias a través de un sensor ultrasónico

Introducción

Un sensor ultrasónico mide la distancia mediante el uso de ondas, específicamente el cabezal emite una onda ultrasónica y recibe la onda reflejada que rebota desde el objeto. Los sensores ultrasónicos miden la distancia al objeto contando el tiempo entre la emisión y la recepción.

Por otro lado, el protocolo de comunicación empleado por nuestro hardware es el I^2C que se destaca por tener dos líneas de señal, una que es la encargada de transportar los datos mientras que la otra solamente se encarga del reloj o *clock*.

Descripción:

Hoy en día existen muchas variantes de sensores ultrasónicos, sin embargo, el que haya tantas variantes tanto de distintas empresas como de las mismas empresas, provoca que existan distintas versiones de bibliotecas para el uso de estos, es por eso, que para el desarrollo de esta práctica se planteó no usar ninguna biblioteca o recurso extra.

Por otro lado, para el desarrollo de esta práctica se planteó trabajar con el sensor ultrasónico HC-SR04, el cual solamente tiene 4 pines, uno de alimentación y otro de tierra (GND), y 2 principales que llevan toda la información del sensor denominados *trigger* y *echo*.

Tabla de entradas y salidas:

A continuación, se muestra una imagen con las entradas y salidas del ESP32-CAM:

Figura 1. Características generales del ESP-32

Tabla 1. Puertos empelados del ESP-32

Puerto	Tipo	Descripción
GPIO 4	Input	Se empleará para adquirir el echo de nuestro sensor
GPIO 2	Output	Se empleará para adquirir el trigger de nuestro sensor

NOTA: Recuerda que para poder grabar en memoria el programa es necesario tener conectado nuestro programador FTDI (Revisa el diagrama de conexiones de la práctica)

Diagrama de conexiones:

A continuación, se muestra el diagrama de conexiones, donde podemos observar como es que se ha integrado el sensor ultrasónico respecto a la configuración de la práctica previa

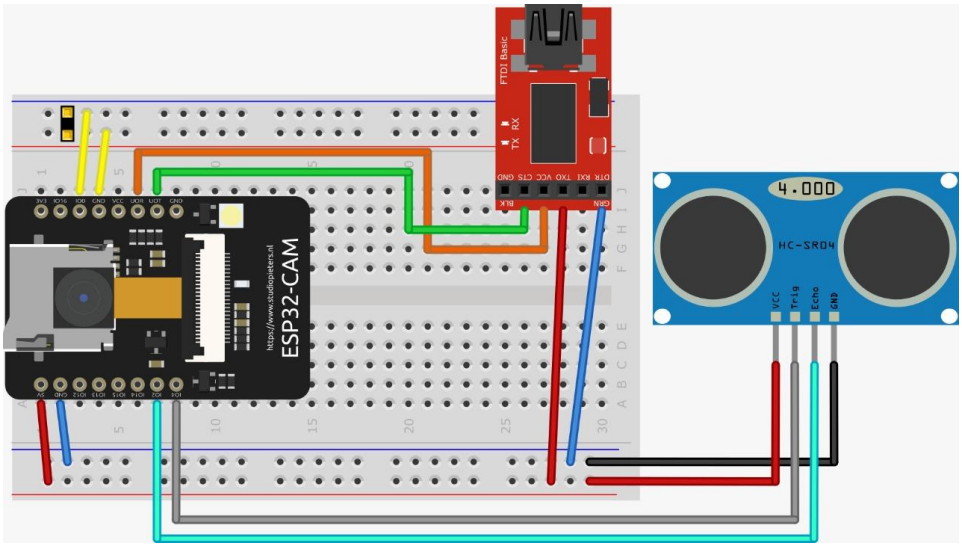

Figura 2. Diagrama de conexiones entre el ESP-32 y nuestro sensor HC-SR04

Es necesario tomar en cuenta una alimentación no mayor a 5 V para el caso del sensor ultrasónico.

Listado del programa y descripción:

A continuación, se muestra paso a paso cada una de las partes que conforman al código encargado del funcionamiento de esta práctica, cabe destacar que para conocer la información adquirida y procesada por nuestro microcontrolador se desplegará a través del monitor serial de nuestro Arduino IDE

1. *Definición de puertos y configuración de estos*

```
#define echoPin 2 // Echo Pin
#define trigPin 4 // Trigger Pin

float duration, distance;

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}
```

En este apartado se declararon en primera instancia los puertos de nuestro sensor para posteriormente ser configurados en nuestra función void setup(), a su vez también se declararon 2 variables globales que serán posteriormente usadas para saber por un lado la duración que tarda una onda en regresar , es decir la diferencia de tiempo que le tomó en recorrer la distancia del sensor al objeto con el que rebotó, por otro lado, otra variable flotante que será la encargada de mostrar la distancia una vez ya procesada.

2. *Lógica principal de programa y procesamiento de las señales adquiridas por el sensor*

```
void loop() {

  // Write a pulse to the HC-SR04 Trigger Pin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the response from the HC-SR04 Echo Pin
  duration = pulseIn(echoPin, HIGH);

  // Determine distance from duration
  // Use 343 metres per second as speed of sound
  distance = (duration / 2) * 0.0343;

  // Send results to Serial Monitor

  Serial.print("Distance = ");
  if (distance >= 400 || distance <= 2) {
     Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
    delay(500);
  }
  delay(500);
}
```

Lo primero que debemos entender es que a través de las señales adquiridas por nuestro sensor

es como haremos una comparación de las señales o pulsos, lo cual podemos apreciar específicamente con la función **digitalWrite()** , una vez ya determinado estas comparaciones es como sabemos que mediante los pulsos en alto de nuestro puerto echo determinaremos la duración que tardará una onda en recorrer una determinada distancia.

Posteriormente, mediante este dato y considerando la velocidad del sonido en metros por segundo es como finalmente determinaremos la distancia a la que se encuentra un objeto respecto a nuestro sensor.

```
// Determine distance from duration
// Use 343 metres per second as speed of sound
distance = (duration / 2) * 0.0343;

// Send results to Serial Monitor
```

Por último, una vez que hemos determinado la distancia de los objetos, mostraremos mediante el monitor serial nuestros resultados, sin embargo, aquí hacemos hincapié en que debido a las limitaciones del hardware no podemos exceder los 4 metros de distancia y tampoco podemos resolver la distancia de objetos que estén a menos de 2 centímetros de cercanía del sensor.

A continuación, se muestra el resultado obtenido una vez descargado en nuestro microcontrolador y empleando el monitor serial de nuestro ESP32 con el IDE de arduino:

Figura 3. Resultados obtenidos mediante el puerto serial y desplegados con el monitor serial del IDE de arduino

Notas y reglas de funcionamiento

No olvides que, una vez descargado el programa, antes de oprimir el botón de reset es necesario desconectar el GPIO0 del GND (Amarillo), en caso de no hacerlo en el monitor serial de nuestro IDE se mostraría un mensaje de “programa por descargar”.

Recursos extras:

Video de funcionamiento de la presente práctica:
https://youtu.be/syB4jWV5IXQ