Part 2

# lesson 9

## Ultrasonic Sensor Module

# Overview

The ultrasonic sensor is great for all kinds of projects that need distance measurements or avoiding obstacles for example. The HC-SR04 is inexpensive and easy to use since we will be using a Library specifically designed for these sensor.

## Component Required:

(1) x Elegoo ESP32

(1) x Ultrasonic sensor module

(4) x F-M wires (Female to Male DuPont wires)


# Component Introduction

### Ultrasonic sensor

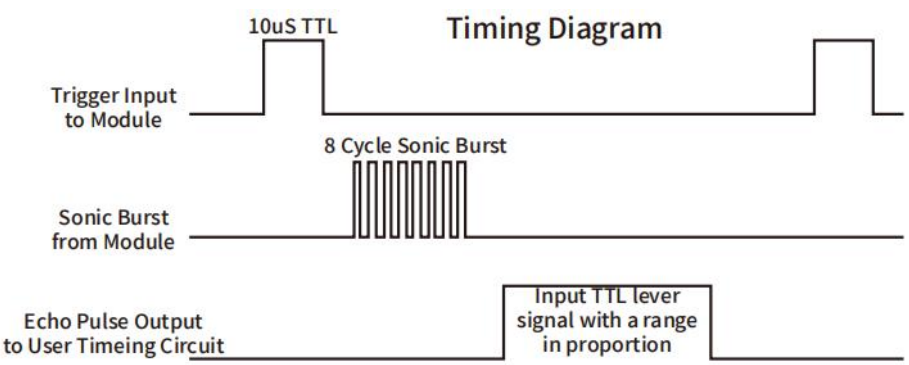
**Ultrasonic** sensor module HC-SR04 provides 2cm-400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:

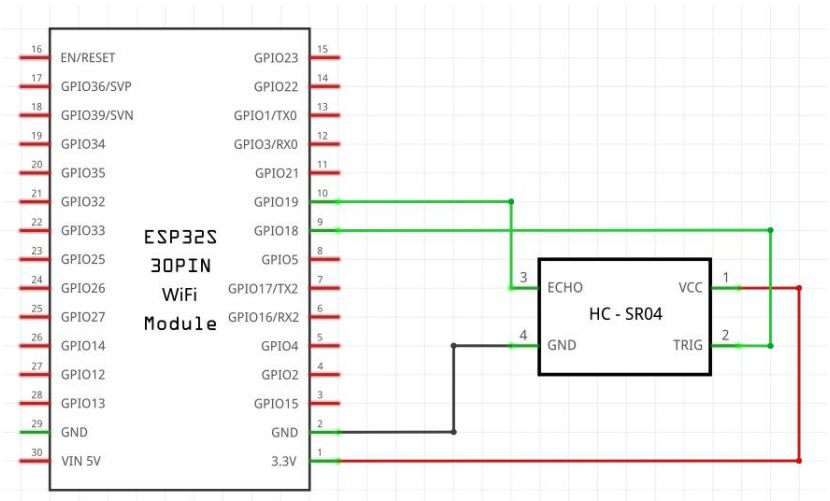
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic tore turning.

**Test** distance = (high level time × velocity of sound (340m/s) /2

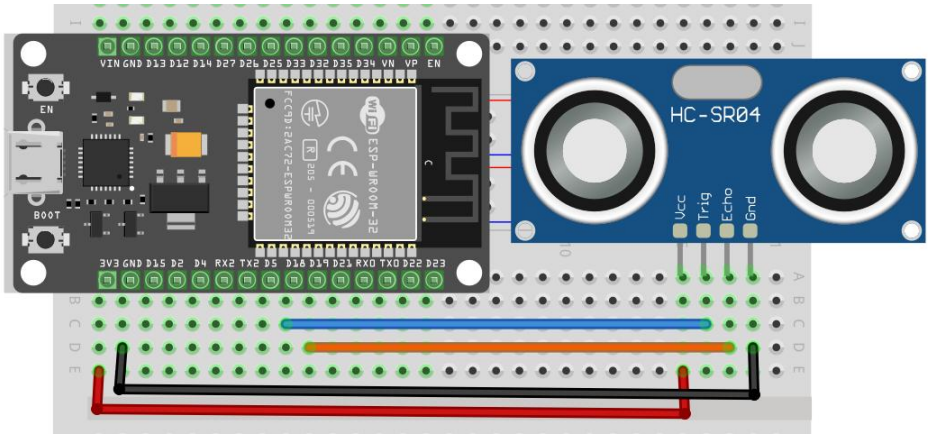
**The** Timing diagram is shown below. You only need to supply a short 10us pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo pulse width is proportional to the distance of the object, or range. You can calculate through the time interval between sending trigger signal and receiving echo signal. Formula: us / 58 = centimeters or us / 148 =inch; or:
the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.

n






**Connection Schematic**




**Wiring diagram**

## Code

Using a Library designed for these sensors will make our code short and simple. We include the library at the beginning of our code, and then by using simple commands we can control the behavior

of the sensor.

After wiring, please open the program in the code folder-SR04_Example and click UPLOAD to upload the program. See Lesson 5 for details about program uploading if there are any errors.

Before you can run this, make sure that you have installed the <HC-SR04> library or re-install it, if necessary. Otherwise, your code won't work.

```
long a;
```

long()
**[Conversion]**
**Description**

Long keyword indicates a long integer data, which is a basic data type in programming language. It is the abbreviation of long int. it is a signed long integer by default, which contains 4 bytes, and the value range is: - 2 ^ 31 ~ (2 ^ 31-1)

## Syntax

```
long(x)
(long)x (C-style type conversion)
```

### Parameters

**x**: a value. Allowed data types: any type.

### Returns

**Data type**: long.

### Part 1: Library Inclusion, Pin Definition & Variable Declaration

```
#include "SR04.h"                    // Include the HC-SR04 sensor library
#define TRIG_PIN 18                  // Define trigger pin as Arduino pin 18
#define ECHO_PIN 19                  // Define echo pin as Arduino pin 19
SR04 sr04 = SR04(ECHO_PIN, TRIG_PIN);  // Create SR04 object with echo/trigger pins
long a;                              // Variable to store measured distance (unit: cm)
```

**Library Inclusion** (#include "SR04.h"): The SR04.h is a third-party library designed for the HC-SR04 ultrasonic sensor. It encapsulates the low-level logic for distance calculation (based on sound wave travel time)

**Pin Definition** (#define TRIG_PIN 18): #define is used to assign meaningful names to Arduino pins for readability. The HC-SR04 sensor has two key pins:

1.TRIG_PIN (Trigger): Sends ultrasonic pulses (controlled by Arduino).

2ECHO_PIN (Echo): Receives the reflected ultrasonic pulses (feeds signal back to ESP32).

**Object Declaration** (SR04 sr04 = SR04(ECHO_PIN, TRIG_PIN)): SR04 is a class in the library that represents the ultrasonic sensor.

```
void setup() {
  Serial.begin(9600);//Initialization of Serial Port
  delay(1000);
}
```

**Serial.begin(9600)**: Initializes serial communication between the Arduino board and a computer (or serial monitor) at a baud rate of 9600. and the computer's serial monitor must be set to the same baud rate (9600) to correctly receive data.

## Part 2: loop() Function ()

```
void loop() {
  a = sr04.Distance();  // Read distance from sensor and store in variable "a"
  Serial.print(a);      // Print the distance value (without line break)
  Serial.println("cm"); // Print "cm" and add a line break
  delay(1000);          // Wait 1 second before next measurement
}
```

**Distance Measurement** (a = sr04.Distance()): The Distance() function is a built-in method of the SR04 class. When called, it triggers the sensor to send ultrasonic pulses, calculates the distance based on the time it takes for the pulses to reflect off an object and return (using the speed of sound: 343 m/s), and returns the result in centimeters (cm). The measured value is stored in variable a.

**Serial Output::**

1.Serial.print(a): Sends the numerical value of a (distance) to the computer via the serial port. Unlike 2.Serial.println(), it does not add a line break after the output.

Open the monitor then you can see the data as blow:

Click the Serial Monitor button to turn on the serial monitor. The basics about the serial monitor are introduced in details in Lesson 4 of part 2.