Ultrasonic ranging module

# Introduction:

The HC-SR04 Ultrasonic Sensor is a very affordable proximity/distance sensor that has been used mainly for object avoidance in various robotics projects . It essentially gives your Arduino eyes / spacial awareness and can prevent your robot from crashing or falling off a table. It has also been used in turret applications, water level sensing, and even as a parking sensor. This simple project will use the HC-SR04 sensor with an Arduino and a Processing sketch to provide a neat little interactive display on your computer screen.

# Parts Required:

* Arduino.
* HC-SR04 Ultrasonic Sensor
* Wires to connect it all together

# Diagram

# https://2.bp.blogspot.com/-EBxJWG31r20/UJ9LQgmFycI/AAAAAAAAASg/SSWcoQUi9iY/s1600/HC_SR04+sketch2.jpg Functions used

## pinMode()

[Digital I/O]

### Description

Configures the specified pin to behave either as an input or an output. See the description of ([digital pins](http://arduino.cc/en/Tutorial/DigitalPins)) for details on the functionality of the pins.

As of Arduino 1.0.1, it is possible to enable the internal pullup resistors with the mode INPUT\_PULLUP. Additionally, the INPUT mode explicitly disables the internal pullups.

### Syntax

pinMode(pin, mode)

### Parameters

pin: the number of the pin whose mode you wish to set

mode: INPUT, OUTPUT, or INPUT\_PULLUP. (see the ([digital pins](http://arduino.cc/en/Tutorial/DigitalPins)) page for a more complete description of the functionality.

## begin()

### Description

Sets the data rate in bits per second (baud) for serial data transmission. For communicating with the computer, use one of these rates: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200. You can, however, specify other rates - for example, to communicate over pins 0 and 1 with a component that requires a particular baud rate.

An optional second argument configures the data, parity, and stop bits. The default is 8 data bits, no parity, one stop bit.

### Syntax

Serial.begin(speed)  
Serial.begin(speed, config)

## digitalWrite()

[Digital I/O]

### Description

Write a HIGH or a LOW value to a digital pin.

If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.

If the pin is configured as an INPUT, digitalWrite() will enable (HIGH) or disable (LOW) the internal pullup on the input pin. It is recommended to set the pinMode() to INPUT\_PULLUP to enable the internal pull-up resistor. See the digital pins tutorial for more information.

If you do not set the pinMode() to OUTPUT, and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

### Syntax

digitalWrite(pin, value)

### Parameters

pin: the pin number

value: HIGH or LOW

## delayMicroseconds()

[Time]

### Description

Pauses the program for the amount of time (in microseconds) specified as parameter. There are a thousand microseconds in a millisecond, and a million microseconds in a second.

Currently, the largest value that will produce an accurate delay is 16383. This could change in future Arduino releases. For delays longer than a few thousand microseconds, you should use delay() instead.

### Syntax

delayMicroseconds(us)

### Parameters

us: the number of microseconds to pause (unsigned int)