KING'S College LONDON

Distance sensing with ultrasonic sensor and Arduino

Plugging in the sensor

- VCC -> Arduino +5V pin
- GND -> Arduino GND pin
- Trig -> Arduino Digital Pin 2
- Echo -> Arduino Digital Pin 2

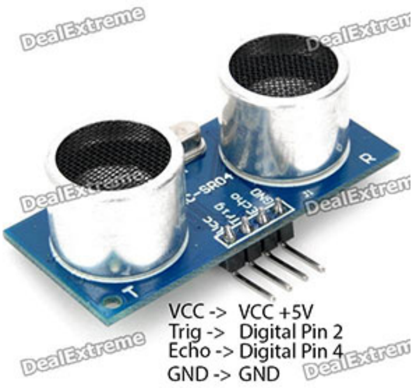


Figure 1(A): Sonar sensor

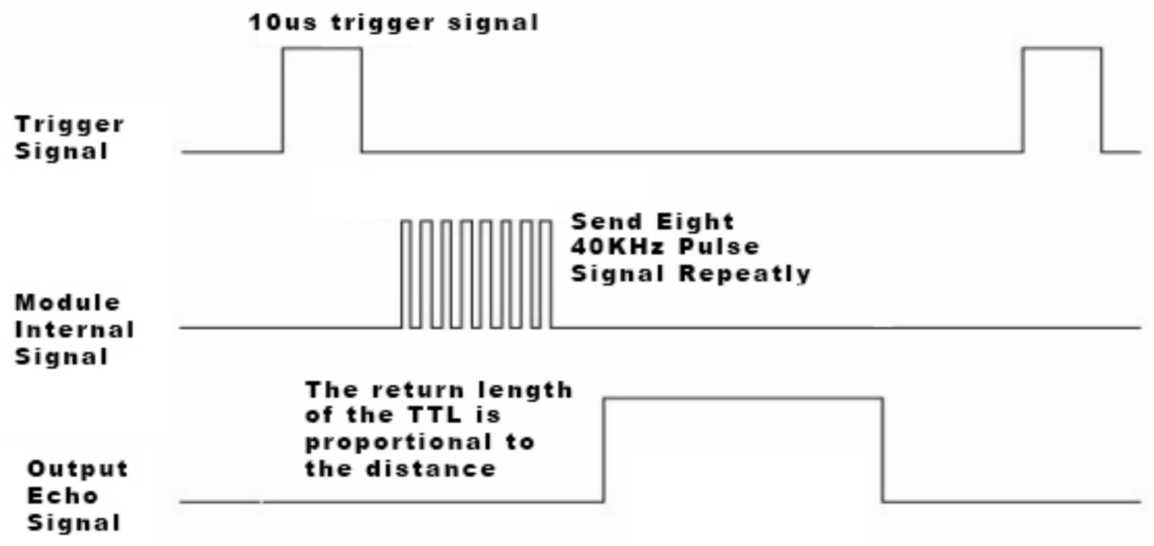

10us trigger signal
Trigger Signal
Send Eight 40KHz Pulse Signal Repeatly
Module Internal Signal
The return length of the TTL is proportional to the distance
Output Echo Signal

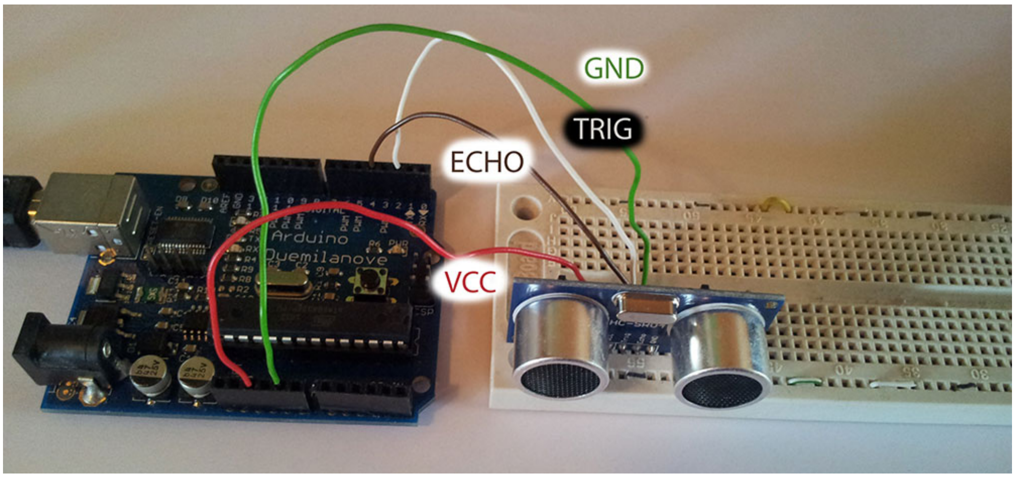
Figure 1(B): Timing diagram.

A sonar sensor works like this: when your program sends a trigger signal through a digital pin (figure 1(B) first row), the sensor emits a ultrasound burst shown in the second row of figure 1(B), and sets the voltage of the echo pin to a HIGH (5V) value, which you can read from a digital pin on the Arduino board.

When the ultrasound burst returned after bouncing on an obstacle, the sensor sets the voltage of the echo pin to a LOW (0V).

The `Trig` pin will be used to send the signal and the `Echo` pin will be used to listen for returning signal.

The assembly looked like this:



Getting data from the sensor

The code is fairly straightforward. I took the original example code for Ping))) sensor and just modified the OUTPUT pin to be `Digital Pin 2`, the INPUT pin to be `Digital Pin 4` and trigger duration to 10 us.

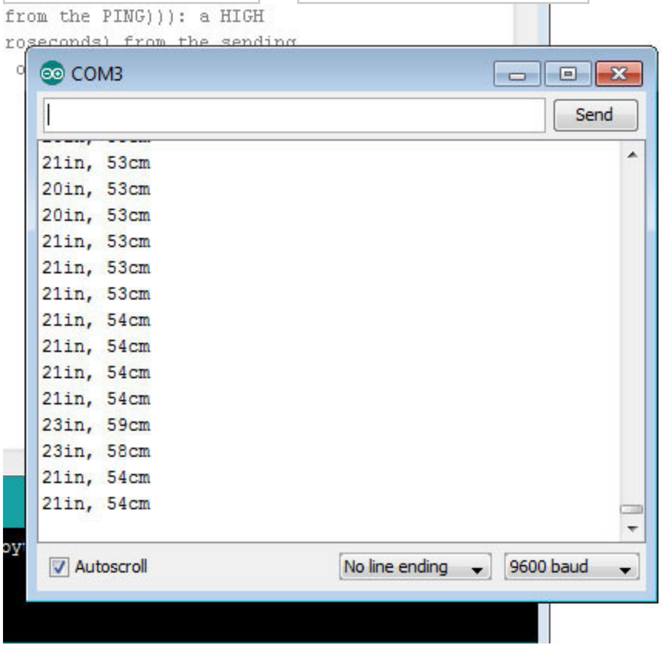
```
/* HC-SR04 Sensor


   https://www.dealextreme.com/p/hc-sr04-ultrasonic-sensor-distance-measuring-module-133696

   This sketch reads a HC-SR04 ultrasonic rangefinder and returns the
   distance to the closest object in range. To do this, it sends a pulse
   to the sensor to initiate a reading, then listens for a pulse
   to return.  The length of the returning pulse is proportional to
   the distance of the object from the sensor.

   The circuit:
       * VCC connection of the sensor attached to +5V
       * GND connection of the sensor attached to ground
       * TRIG connection of the sensor attached to digital pin 2
       * ECHO connection of the sensor attached to digital pin 4
   Original code for Ping))) example was created by David A. Mellis
   Adapted for HC-SR04 by Tautvidas Sipavicius
   This example code is in the public domain.
 */
```

```
const int trigPin = 2; //This can be any pin on the digital port
const int echoPin = 4; //Any pin on the digital port


void setup() {
  // initialize serial communication:
  Serial.begin(9600);
}


void loop()
{
  // establish variables for duration of the ping,
  // and the distance result in inches and centimeters:
  long duration, inches, cm;


  // The sensor is triggered by a HIGH pulse of 10 or more microseconds.
  // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
  pinMode(trigPin, OUTPUT); //Since you use this pin on the Arduino board to send a
  // trigger pulse to the sonar sensor, configure the pin to an OUTPUT
  digitalWrite(trigPin, LOW); //First reset the pin to LOW.
  delayMicroseconds(2); //Wait for 2 micro seconds
  digitalWrite(trigPin, HIGH); //Send the trigger pulse by setting its voltage to HIGH
// HIGH means its voltage will be 5V in transistor-transistor logic (TTL).
  delayMicroseconds(10); //Hold the pulse for 10 microsecond (you can change this)
  digitalWrite(trigPin, LOW); //Then set it back to LOW (0V)


  // Read the signal from the sensor: a HIGH pulse whose
  // duration is the time (in microseconds) from the sending
  // of the ping to the reception of its echo off of an object.
  pinMode(echoPin, INPUT); //Since the voltage of the echo pin of the sensor has to be read
//by this pin, configure it as an INPUT pin.
  duration = pulseIn(echoPin, HIGH); //get the duration this echo pin kept its voltage at HIGH (5V)


  // convert the time into a distance
  inches = microsecondsToInches(duration); //Then use thiss function to compute the distance to
//the object in inches
  cm = microsecondsToCentimeters(duration); //In cm

  Serial.print(inches); //You can display values of variables like this
  Serial.print("in, ");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();

  delay(100); //Use a 100ms delay before going to the next iteration
}


long microsecondsToInches(long microseconds)
{
  // According to Parallax's datasheet for the PING))), there are
  // 73.746 microseconds per inch (i.e. sound travels at 1130 feet per
  // second).  This gives the distance travelled by the ping, outbound
  // and return, so we divide by 2 to get the distance of the obstacle.
  // See: http://www.parallax.com/dl/docs/prod/acc/28015-PING-v1.3.pdf
  return microseconds / 74 / 2;
}


long microsecondsToCentimeters(long microseconds)
```

```
{
  // The speed of sound is 340 m/s or 29 microseconds per centimeter.
  // The ping travels out and back, so to find the distance of the
  // object we take half of the distance travelled.
  return microseconds / 29 / 2;
}
```

view rawhc-sr04.ino hosted with by GitHub

After compiling, uploading and running it, in the `Serial Monitor` (`Tools -> Serial Monitor`or `Ctrl + Shift + M`) the sensor was sending correct data!



```
21in, 53cm
20in, 53cm
20in, 53cm
21in, 53cm
21in, 53cm
21in, 53cm
21in, 54cm
21in, 54cm
21in, 54cm
21in, 54cm
23in, 59cm
23in, 58cm
21in, 54cm
21in, 54cm
```

☑ Autoscroll   No line ending   9600 baud

After a bit of fooling around I found out that the range is approximately 2cm - 300cm