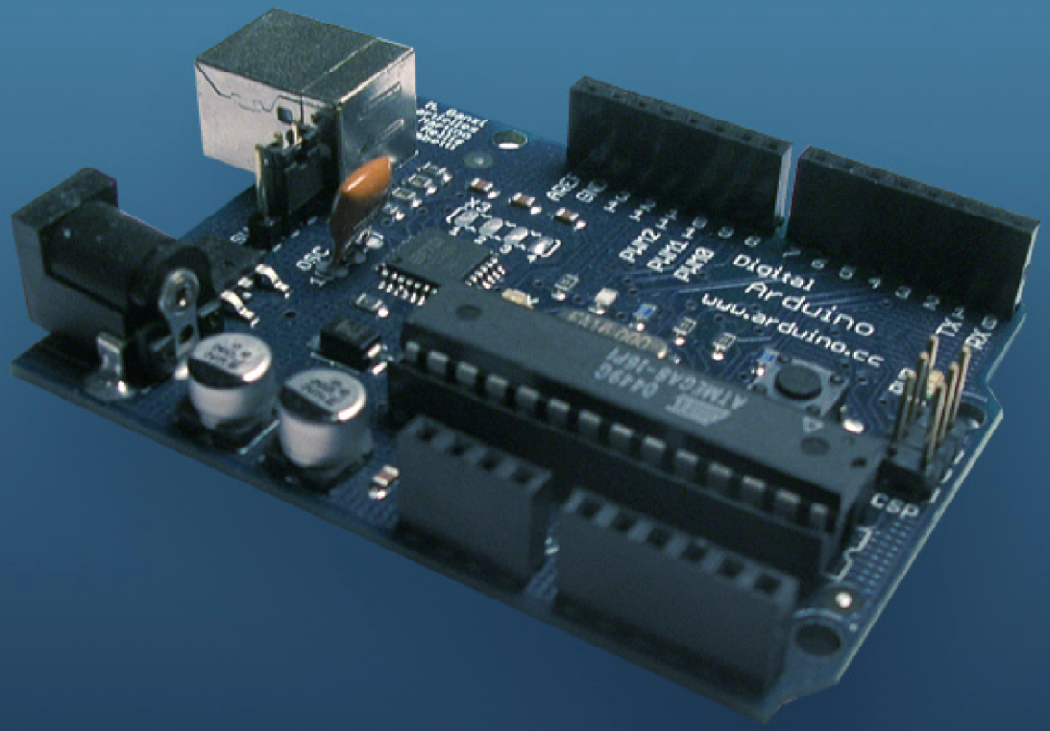
Arduino

Physical Computing I/O board

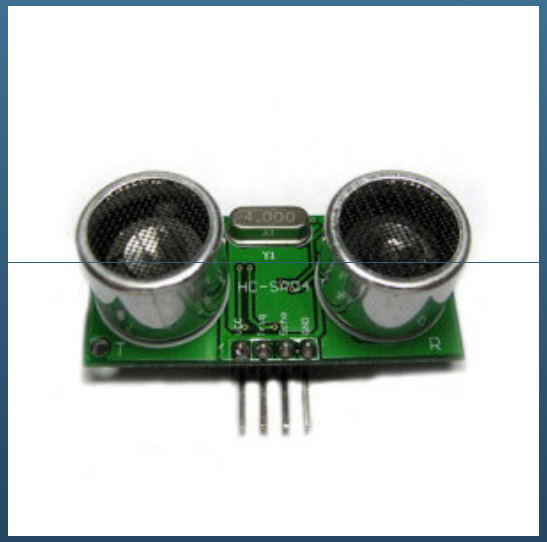


14^ parte : Sensore a ultrasuoni HC SR04 e Arduino

Author: Ing. Sebastiano Giannitto (ITIS “M.BARTOLO” –PACHINO)

La GPRS Shield

Un sensore a ultrasuoni come il HC SR 04 misura il tempo impiegato dalle onde sonore emesse da una sorgente a ritornare dopo aver incontrato un ostacolo che le riflette. Il fascio di onde sonore emesso ha forma conica e lo stesso vale per le onde riflesse da un ostacolo, questo fa si che il sensore riceva molte riflessioni da diversi oggetti, ciò rende il sensore, da solo, incapace di distinguere un oggetto da un altro o aperture negli oggetti troppo piccoli.



Calcolo della distanza

La velocità del suono nell'aria alla temperatura di 20° è di circa 343,4 m/s, qui useremo la velocità approssimata di 340 m/s per semplicità, se è necessaria una maggior precisione si può usare la seguente legge in funzione della temperatura *t*:

*V = 331.4 + 0.62*t*

Il sensore restituisce il tempo impiegato per andare e tornare dalle onde sonore in microsecondi, inoltre è comodo avere la misura in cm, quindi bisogna convertire la velocità del suono da m/s in cm/microsecondo:

1 m/s = 10^2/10^6 cm/microsec = 10^-4 cm/microsec => 340 m/s = 3,4*10^-2 cm/microsec

Il tutto va ancora diviso per 2 in quanto il tempo che abbiamo convertito è quello impiegato per andare e tornare indietro dalle onde, mentre per calcolare la distanza dall'oggetto ci basta metà di questo tempo, la formula finale, dove *t* è il tempo restituito dal sensore in *cm/microsec* è: ***S = 1.7 * 10^-2 * t cm***

La GPRS Shield

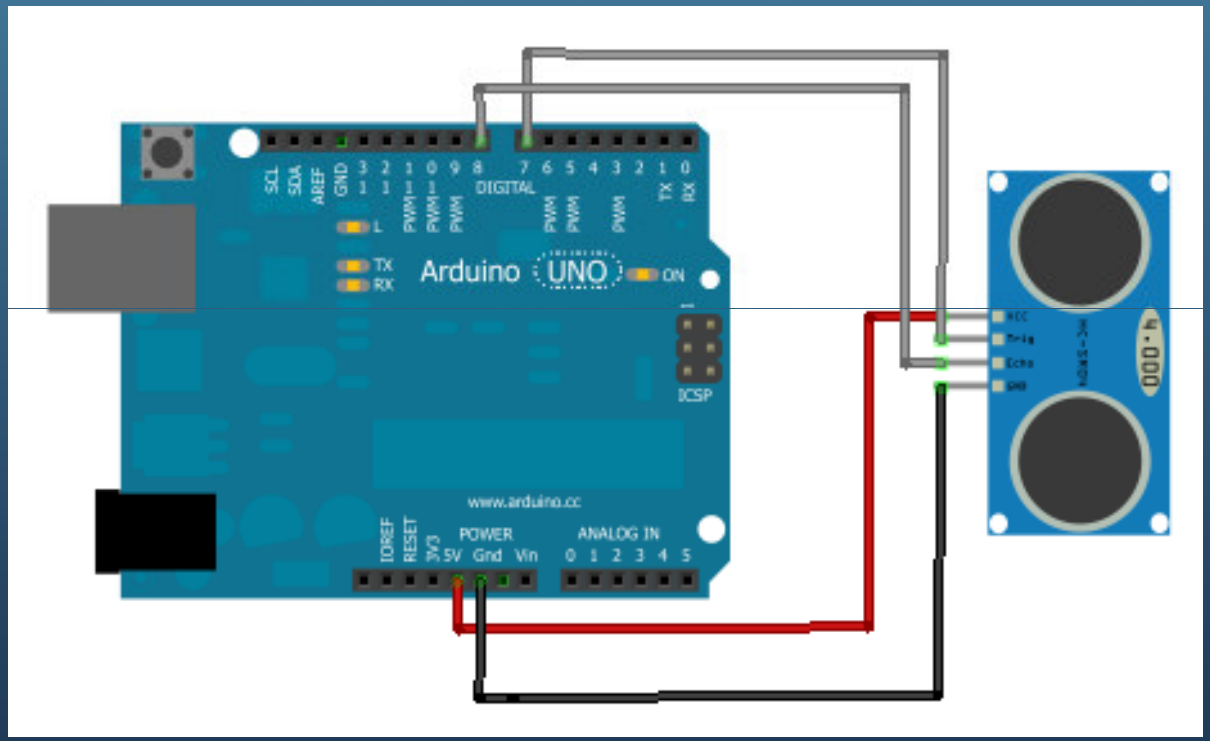
HC SR 04

Il sensore dispone di 4 pin: Vcc (+5V), Trigger, Echo, GND. Si invia un impulso alto sul pin Trigger per almeno 10 microsec, a questo punto il sensore invierà il ping sonoro e aspetterà il ritorno delle onde riflesse, il sensore risponderà sul pin Echo con un impulso alto della durata corrispondente a quella di viaggio delle onde sonore, dopo 38 millisec si considera che non sia stato incontrato alcun ostacolo. Per sicurezza si aspettano in genere 50-60 millisec per far si che non vi siano interferenze con la misura successiva.

Per maggiori informazioni c'è il datasheet.

Connettere Arduino e il sensore

Basta connettere il sensore a +5V e GND di arduino rispettivamente ai pin Vcc e GND del sensore, e Trigger ed Echo a due porte qualsiasi di Arduino(o anche una sola se si fosse a corto di uscite). Lo schema è il seguente:

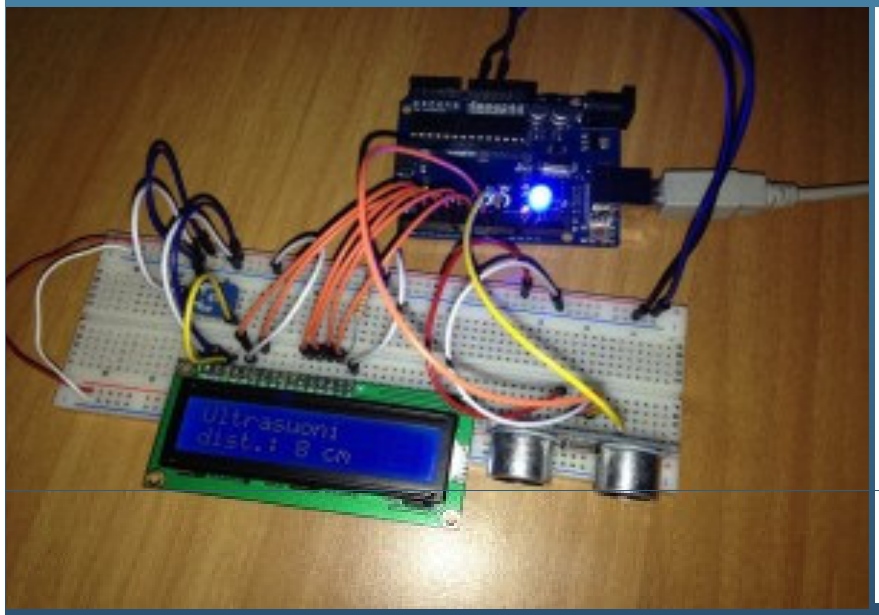


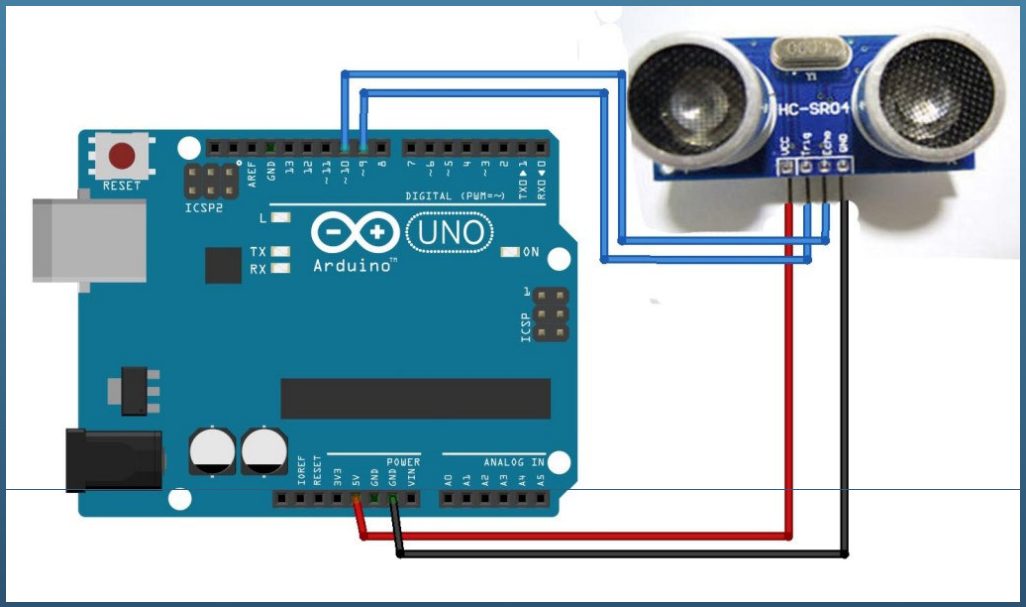
Lo sketch di prova

```
//HC RS04 Sensore ultrasuoni
int triggerPort = 7;
int echoPort = 8;

void setup()
{
   pinMode( triggerPort, OUTPUT );
   pinMode( echoPort, INPUT );
  Serial.begin( 9600 );
  Serial.println( "Sensore ultrasuoni: ");
 }

void loop() {
//porta bassa l'uscita del trigger
digitalWrite( triggerPort, LOW );

//invia un impulso di 10microsec su trigger
digitalWrite( triggerPort, HIGH );
delayMicroseconds( 10 );
digitalWrite( triggerPort, LOW );
long duration = pulseIn( echoPort, HIGH );
long r = 0.034 * duration / 2;
Serial.print( "durata: " );
Serial.print( duration );
Serial.print( " , " );
Serial.print( "distanza: " );

//dopo 38ms è fuori dalla portata del sensore
if( duration > 38000 ) Serial.println( "fuori portata");
else { Serial.print( r );
Serial.println( "cm" );}
//aspetta 1.5 secondi
delay( 1500 );
}
```

Video
https://www.youtube.com/watch?v=3rSUEg2ZzsE

Variante con display

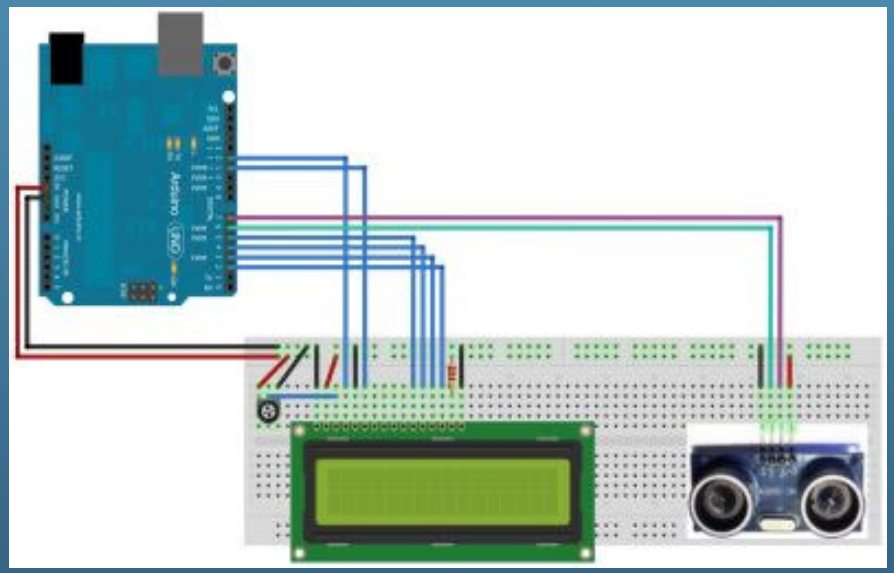




```
//HC RS04 Sensore ultrasuoni

const int triggerPort = 9;
const int echoPort = 10;
const int led = 13;

void setup()
{
 pinMode(triggerPort, OUTPUT);
pinMode(echoPort, INPUT);
pinMode(led, OUTPUT);
Serial.begin(9600);
Serial.print( "Sensore Ultrasuoni: ");
}
void loop() {

//porta bassa l'uscita del trigger
digitalWrite( triggerPort, LOW );
//invia un impulso di 10microsec su trigger
digitalWrite( triggerPort, HIGH );
delayMicroseconds( 10 );
digitalWrite( triggerPort, LOW );
 long durata = pulseIn( echoPort, HIGH );
 long distanza = 0.034 * durata / 2;
 Serial.print("distanza: ");
//dopo 38ms è fuori dalla portata del sensore
if( durata > 38000 ){
Serial.println("Fuori portata   ");
}
else{
Serial.print(distanza);
Serial.println(" cm     ");
}

if(distanza < 10){
digitalWrite(led, HIGH);
}
else{
digitalWrite(led, LOW);
}

//Aspetta 1000 microsecondi
delay(1000);
}
```

Scketch con display lcd:

```
#include <LiquidCrystal.h>
const int triggerPort = 9;
const int echoPort = 10;
const int led = 13;

LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

void setup() {

pinMode(triggerPort, OUTPUT);
pinMode(echoPort, INPUT);
pinMode(led, OUTPUT);
lcd.begin(16, 2);
lcd.setCursor(0, 0);
lcd.print( "Ultrasuoni ");
}

void loop() {
 //porta bassa l'uscita del trigger
digitalWrite( triggerPort, LOW );
//invia un impulso di 10microsec su trigger
digitalWrite( triggerPort, HIGH );
delayMicroseconds( 10 );
digitalWrite( triggerPort, LOW );
long durata = pulseIn( echoPort, HIGH );
 long distanza = 0.034 * durata / 2;
 lcd.setCursor(0, 1);
lcd.print("dist.: ");

//dopo 38ms è fuori dalla portata del sensore
if( durata > 38000 ){
lcd.setCursor(0, 1);
lcd.println("Fuori portata   ");
}
else{
lcd.print(distanza);
lcd.println(" cm     ");
}

if(distanza < 10){
digitalWrite(led, HIGH);
}
else{
digitalWrite(led, LOW);
}

//Aspetta 1000 microsecondi
delay(1000);
}
```



Close-up of LCD Wiring

0V
5V
Contrast: 1k to Ground and 10k to 5V
RS: to Arduino pin12
R/W: to 0V
E: to Arduino pin11
DB0: n/c
DB1: n/c
DB2: n/c
DB3: n/c

DB4: to Arduino pin 5
DB5: to Arduino pin 4
DB6: to Arduino pin 3
DB7: to Arduino pin 2
Anode: to 100Ω or 220Ω resistor; other end of resistor to 5V
Cathode: to 0V

LCD Module 1602

Solder the 16-pin header (provided) to the module

Use a 100Ω resistor to connect the backlight
Use a 10K potentiometer (or 1K and 10K resistors as shown below) for contrast
Only 4 data pins required in 4 bit mode

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int pingPin = 7;  //output 10us pulse to this pin
 int inPin = 6;  //measure return pulse width from this pin
long duration, inches, cm;
int indec, cmdec;
int inchconv = 147;  // ratio between puls width and inches
int cmconv = 59;  // ratio between pulse width and cm
String s1, s2;  // initialise LCD library and pins

void setup()
{
 lcd.begin(16, 2);
 pinMode(pingPin, OUTPUT);
 pinMode(inPin, INPUT);
 }
```

```
void loop()
{
// Send a short LOW followed by HIGH pulse to Trigger input:
digitalWrite(pingPin, LOW);
delayMicroseconds(2);
digitalWrite(pingPin, HIGH);
delayMicroseconds(10);
digitalWrite(pingPin, LOW);   // read the length of the return pulse on Echo output
duration = pulseIn(inPin, HIGH);   // convert the time into a distance (non-floating point with decimals
inches = microsecondsToInches(duration);
 indec = (duration - inches * inchconv) * 10 / inchconv;
cm = microsecondsToCentimeters(duration);
cmdec = (duration - cm * cmconv) * 10 / cmconv;
 s1 = String(inches) + "." + String(indec) + "in" + " ";
s2 = String(cm) + "." + String(cmdec) + "cm" + " ";
 lcd.setCursor(0, 0); // print inches on top line of LCD
lcd.print(s1);
lcd.setCursor(0,1); // print cm on second line of LCD
 lcd.print(s2);   delay(100);
}
long microsecondsToInches(long microseconds) {   return microseconds / inchconv; }
long microsecondsToCentimeters(long microseconds) {   return microseconds / cmconv;
}
```