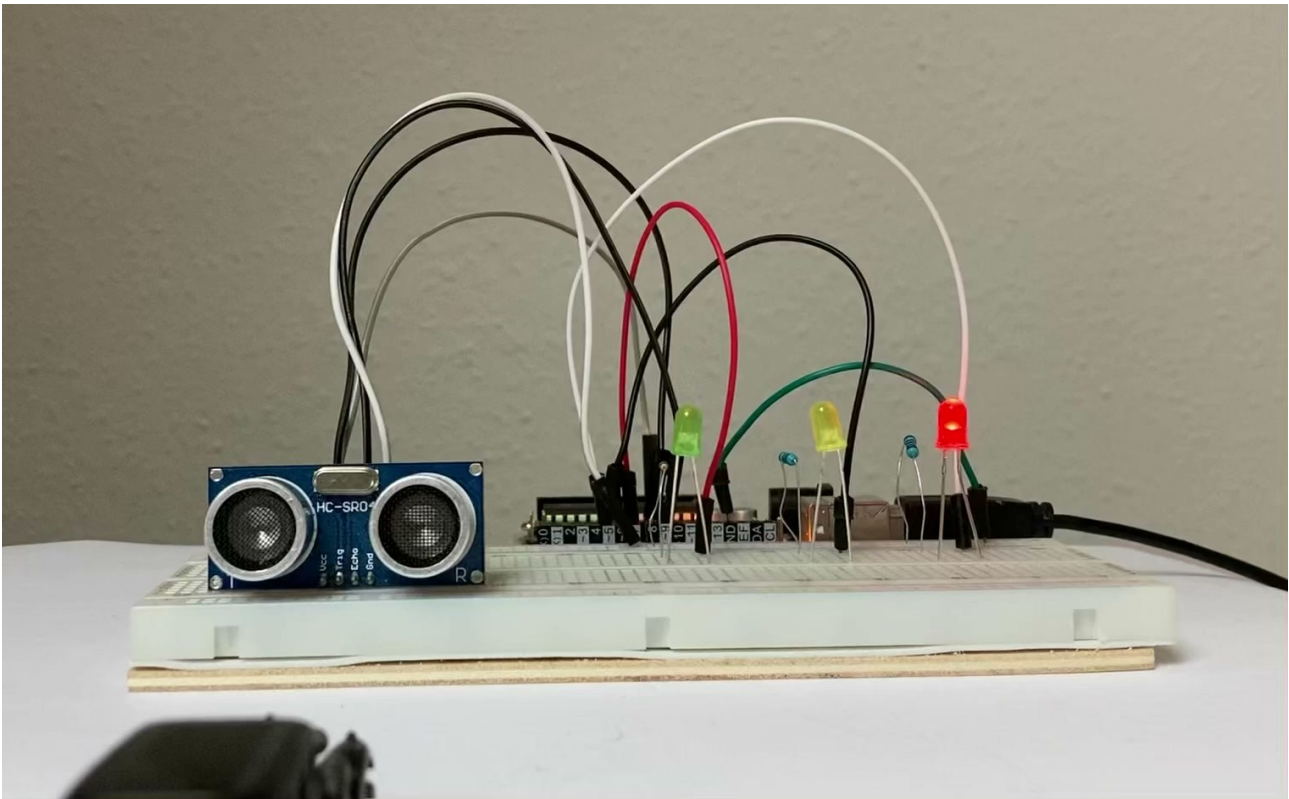
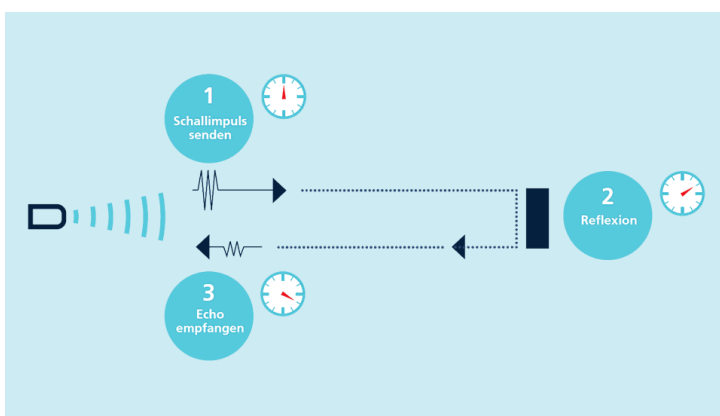


Die Ampel zeigt zunächst rot. Wenn die Entfernung kleiner als 10 cm ist, soll sie nach 1 Sekunde Wartezeit erst auf rot/gelb für eine Sekunde, dann auf grün für drei Sekunden springen. Anschließend folgt wieder eine Sekunde gelb und dann wieder rot.



Der Ultraschallsensor arbeitet nach einem einfachen Prinzip:



In regelmäßigen Abständen wird eine Schallwelle gesendet. Trifft sie auf einen Gegenstand wird die Schallwelle reflektiert und gelangt als Echo zurück. Mithilfe der Zeitspanne zwischen dem Aussenden des Signals und dem Wiedereintreffen lässt sich die Entfernung berechnen.

<https://www.microsonic.de/de/service/ultraschallsensoren/prinzip.htm>

Der Sender schickt das Signal und es wird vom Echo empfangen. Die Geschwindigkeit des Schalls in der Luft beträgt 343,2 m/s. Der Arduino bestimmt die Zeit in Millisekunden (1 Sekunde = 1000 Millisekunden), deshalb muss der Wert in cm pro Millisekunde berechnet werden:

Umrechnung in cm: $343,2 \text{ m} \cdot 100 = 34.320 \text{ cm}$

Strecke pro Sekunde: $34.320 : 1.000 = 34,32 \text{ cm}$

Strecke pro Millisekunde: $34,3 : 1.000 = 0,03432 \text{ cm}$

Benötigte Bauteile:

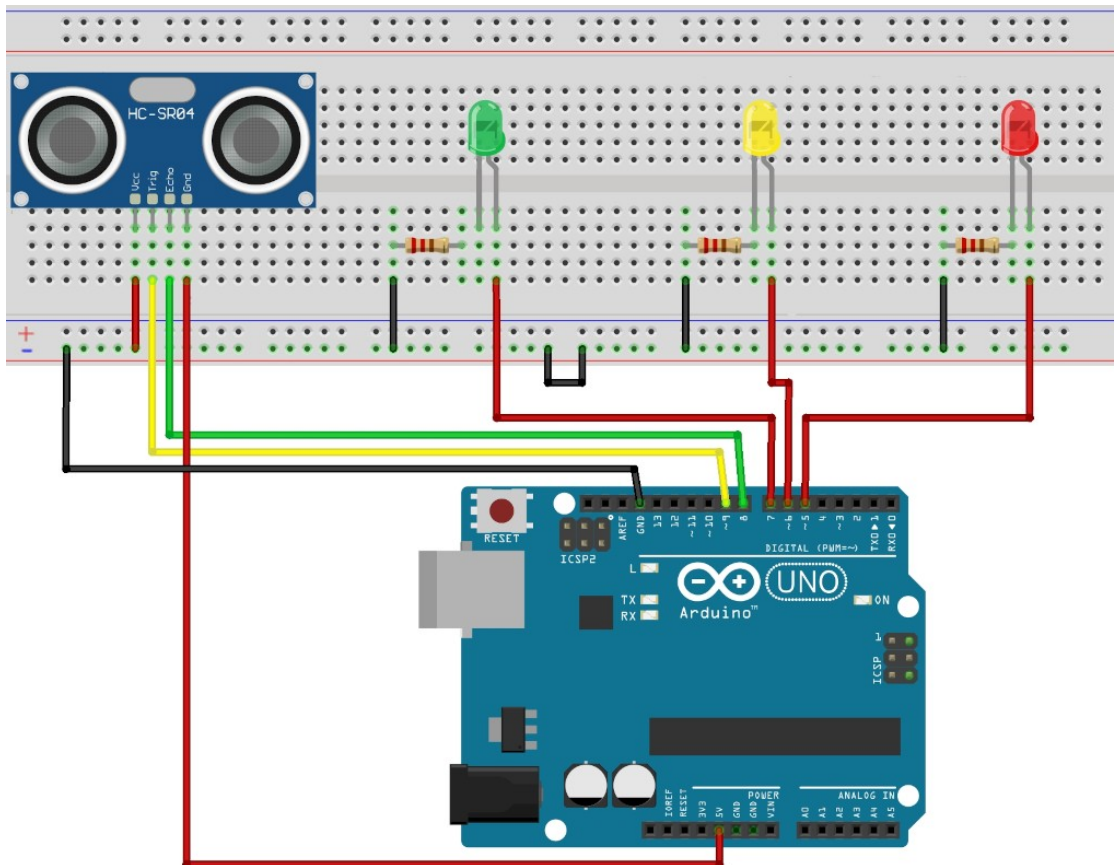
- Ultraschallsensor HC-SR04
- 3 LEDs (rot, gelb, grün)
- 3 Widerstände 220 Ω
- Leitungsdrähte



Trg = senden **Echo** = empfangen

Damit der Ultraschallsensor ordnungsgemäß arbeiten kann, musst du ihn so einstecken, dass Sender und Empfänger nach vorn zeigen, damit das Signal ungehindert gesendet und empfangen werden kann. Die Leitungsdrähte werden auf der Rückseite eingesteckt.

Baue die Schaltung auf.



Definiere die benötigten Variablen:

```
# define Senden 9
# define Echo 8
# define ROT 5
# define GELB 6
# define GRUEN 7
long Zeit = 0;
long Entfernung = 0;
```

Der setup-Teil:

```
void setup()
{
  pinMode(Senden, OUTPUT);
  pinMode(Echo, INPUT);
  pinMode(ROT, OUTPUT);
  pinMode(GELB, OUTPUT);
  pinMode(GRUEN, OUTPUT);
}
```



Verwende die Funktion EntfernungMessen():

```
int EntfernungMessen()
{
  long Entfernung = 0;

  // Sender kurz ausschalten um Störungen des Signal zu vermeiden
  digitalWrite(SENDEN, LOW);
  delay(5);

  // Signal senden
  digitalWrite(SENDEN, HIGH);
  delayMicroseconds(10);
  digitalWrite(SENDEN, LOW);

  // pulseIn -> Zeit messen, bis das Signal zurückkommt
  long Zeit = pulseIn(ECHO, HIGH);

  // Entfernung in cm berechnen
  Entfernung = (Zeit / 2) * 0.03432;
  return Entfernung;
}
```

Die Methode `AmpelSchalten()` sorgt dafür, dass die Ampel im festgelegten Rhythmus geschaltet wird (rot/gelb → 1 Sekunde, grün → 3 Sekunden, gelb → 1 Sekunde, rot).

```
void AmpelSchalten()
{
  delay(1000);
  digitalWrite(GELB, HIGH);
  delay(1000);
  digitalWrite(GELB, LOW);
  digitalWrite(ROT, LOW);
  digitalWrite(GRUEN, HIGH);
  delay(3000);
  digitalWrite(GRUEN, LOW);
  digitalWrite(GELB, HIGH);
  delay(1000);
  digitalWrite(GELB, LOW);
  digitalWrite(ROT, HIGH);
}
```

Der loop-Teil:

```
void loop()
{
  digitalWrite(ROT, HIGH);

  // Funktion aufrufen
  Entfernung = EntfernungMessen();

  // Ampel schalten
  if (Entfernung < 10)
  {
    AmpelSchalten();
  }
}
```

Hartmut Waller (hartmut-waller.info/arduino-blog) Letzte Änderung: 10.05.23