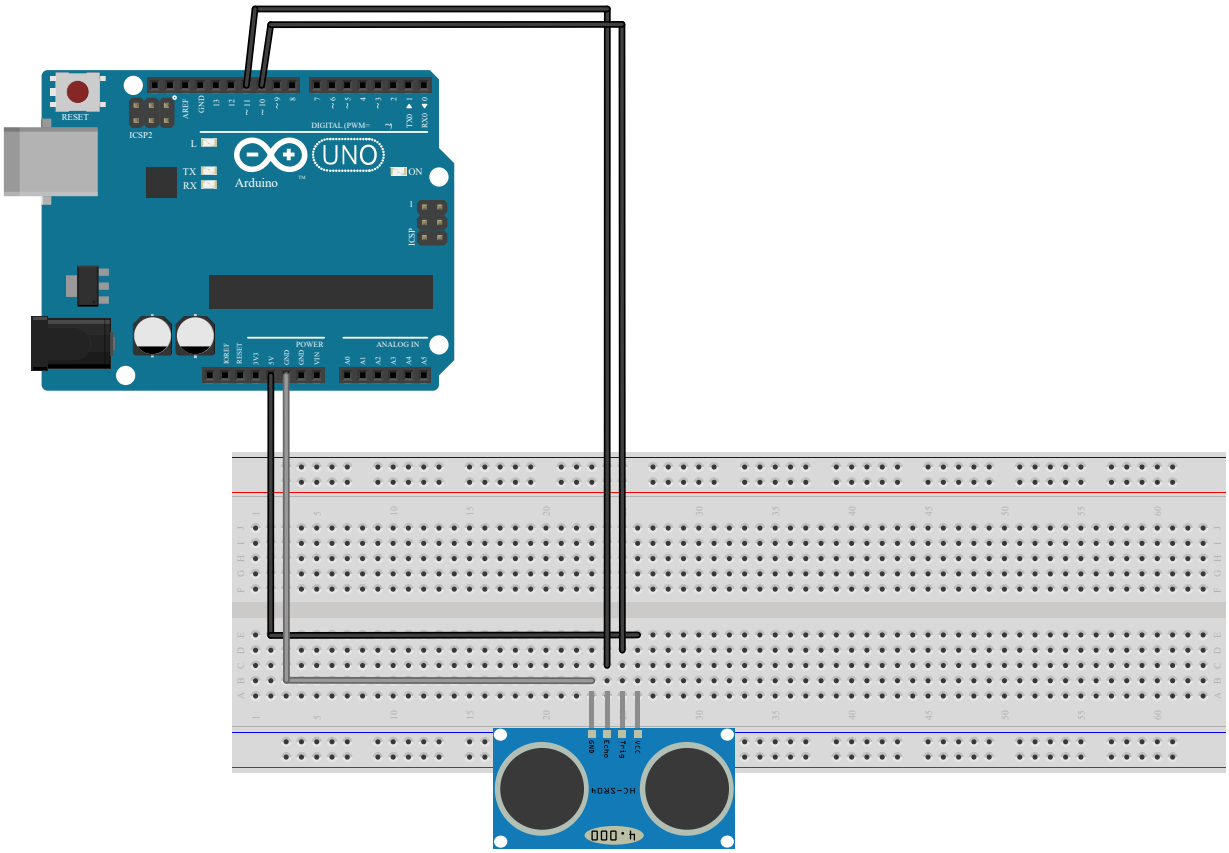
Compte rendu du projet de SI

Imperator Dogoni & Aulys, 1G1, 03/2022

I. Capteur : câblage et code

Exemple de câblage possible avec une carte Arduino UNO et un capteur HC-SR04 :



- Alimentation : `5V` de la carte Arduino → `VCC` du capteur
- Fil de terre : `GND` de la carte → `GND` du capteur (représenté en gris sur les schémas)
- Envoi du signal : `D2` de la carte → `TRIGGER` du capteur
- Réception du signal : `D3` de la carte → `ECHO` du capteur

Programme général :

Programme de base, permettant d'afficher simplement la distance d'un objet par rapport au capteur à ultrasons.

```
#define trigPin 13
#define echoPin 12

void setup()
{
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
}

void loop()
{
    long duration, distance;
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    duration = pulseIn(echoPin, HIGH);

    distance = duration * 340 / (2 * 10000);

    Serial.print(distance);
    Serial.println("cm");

    delay(100);
}
```

Explication du calcul `distance = duration * 340 / (2 * 10000)` :

On sait que $v = \frac{d}{\Delta t}$, avec v la vitesse, d la distance parcourue par le signal, et Δt la durée de l'étude.

$v = \frac{d}{\Delta t}$ donc $d = v \times \Delta t$.

Or, la vitesse du son dans l'air, pour des conditions de températures et de pression considérées comme moyennes et invariables, est d'environ $340\ m.s^{-1}$.

Les ultrasons émis par le capteur parcourent une distance d, puis reparcourent cette distance dans le sens opposé pour revenir vers le capteur. Ainsi, la distance réelle entre le capteur et l'objet est de $\frac{d}{2}$.

Également, on souhaite mesurer la distance en centimètres et la mesure de Δt par le capteur se fait en microsecondes. $340\ m.s^{-1} = 340 \times 10^{-2}$ et $1\ \mu s = 10^{-6}\ s$.

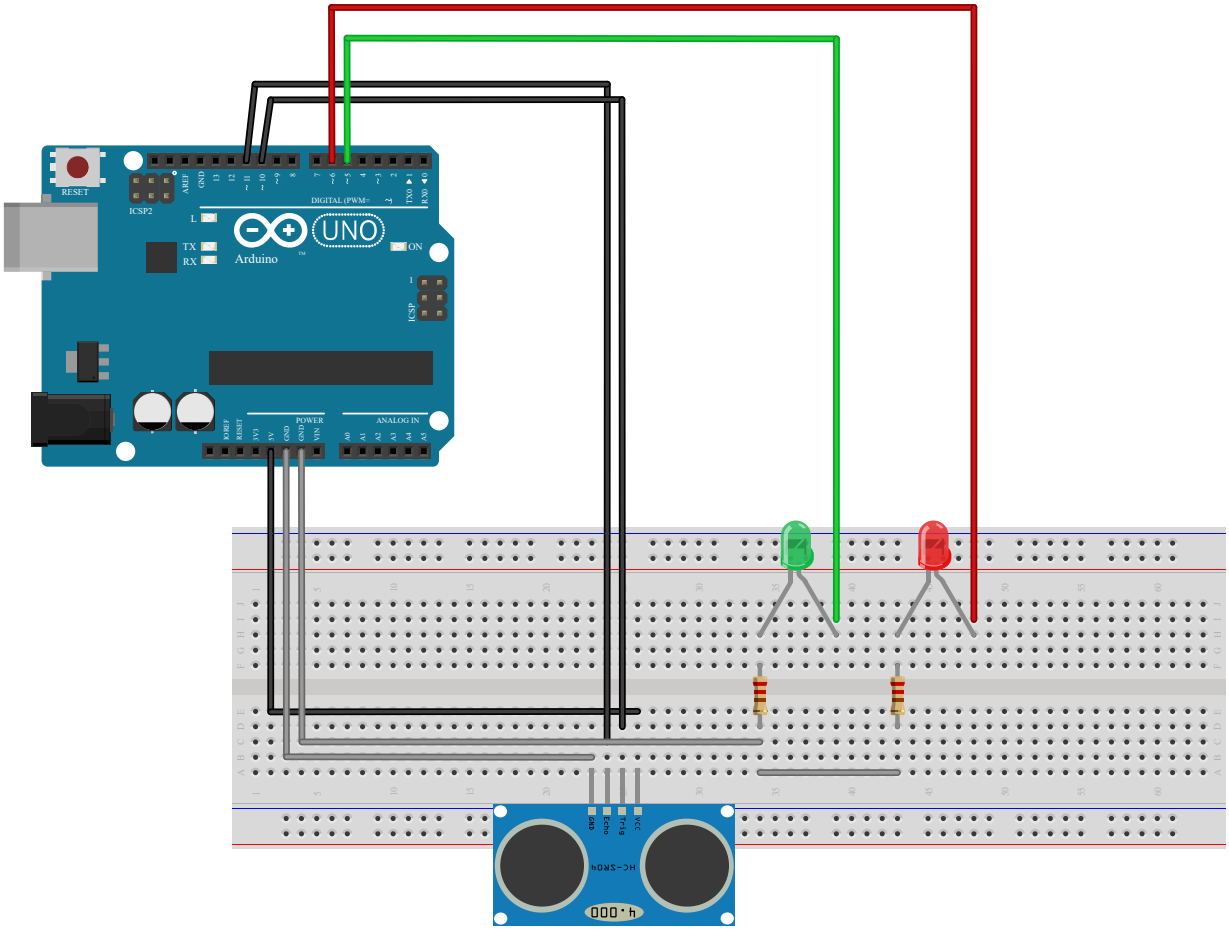
Or, $\frac{10^{-2}}{10^{-6}} = 10000$.

On divise donc par 10^{-5} ainsi on obtient : $d = \Delta t \times (\frac{v}{2 \times 10000})$.

d correspond à `distance` ; Δt correspond à `duration`. On obtient donc `distance = duration * 340 / (2 * 10000)`.

II. Leds rouge et verte

Schéma câblage :



Programme :

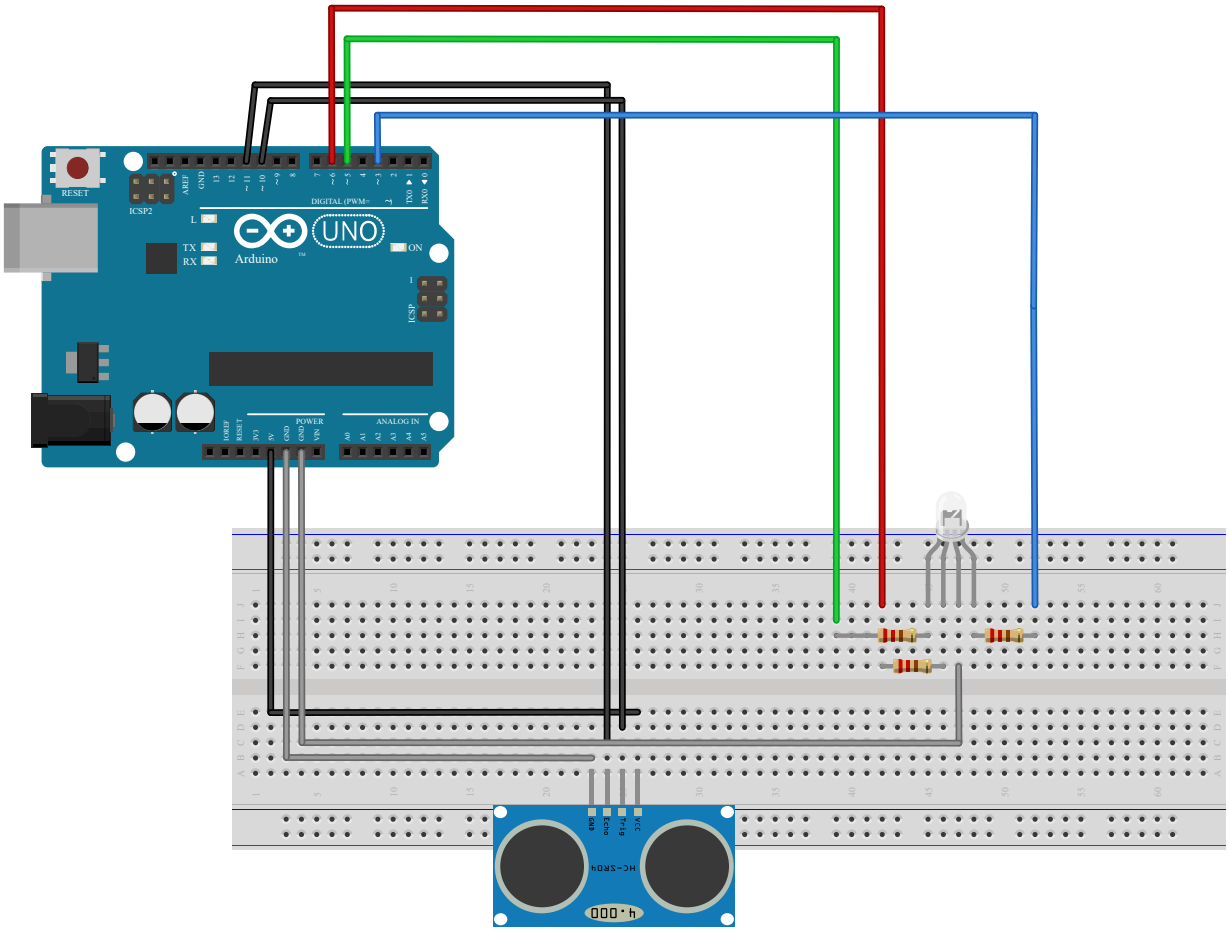
Permet d'allumer une led verte pour `d < 10` et une led rouge pour `d > 10`.

```
#define trigPin 13
#define echoPin 12
#define ledVerte 5
#define ledRouge 6

void setup()
{
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(ledVerte, OUTPUT);
    pinMode(ledRouge, OUTPUT);
}
```

```
void loop()
{
    long duration, distance;
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    duration = pulseIn(echoPin, HIGH);

    distance = duration * 340 / (2 * 10000);

    if (distance < 10)
    {
        digitalWrite(ledVerte, LOW);
        digitalWrite(ledRouge, HIGH);
    }
    else
    {
        digitalWrite(ledVerte, HIGH);
        digitalWrite(ledRouge, LOW);
    }
    Serial.print(distance);
    Serial.println("cm");

    delay(100);
}
```

III. Led RGB (permière partie)

Schéma câblage :



Le programme est le même que pour les leds rouge et verte. On pensera à vérifier les ports des différentes pattes de la led RGB, ou les changer dans le programme.

IV. Led RGB (deuxième partie)

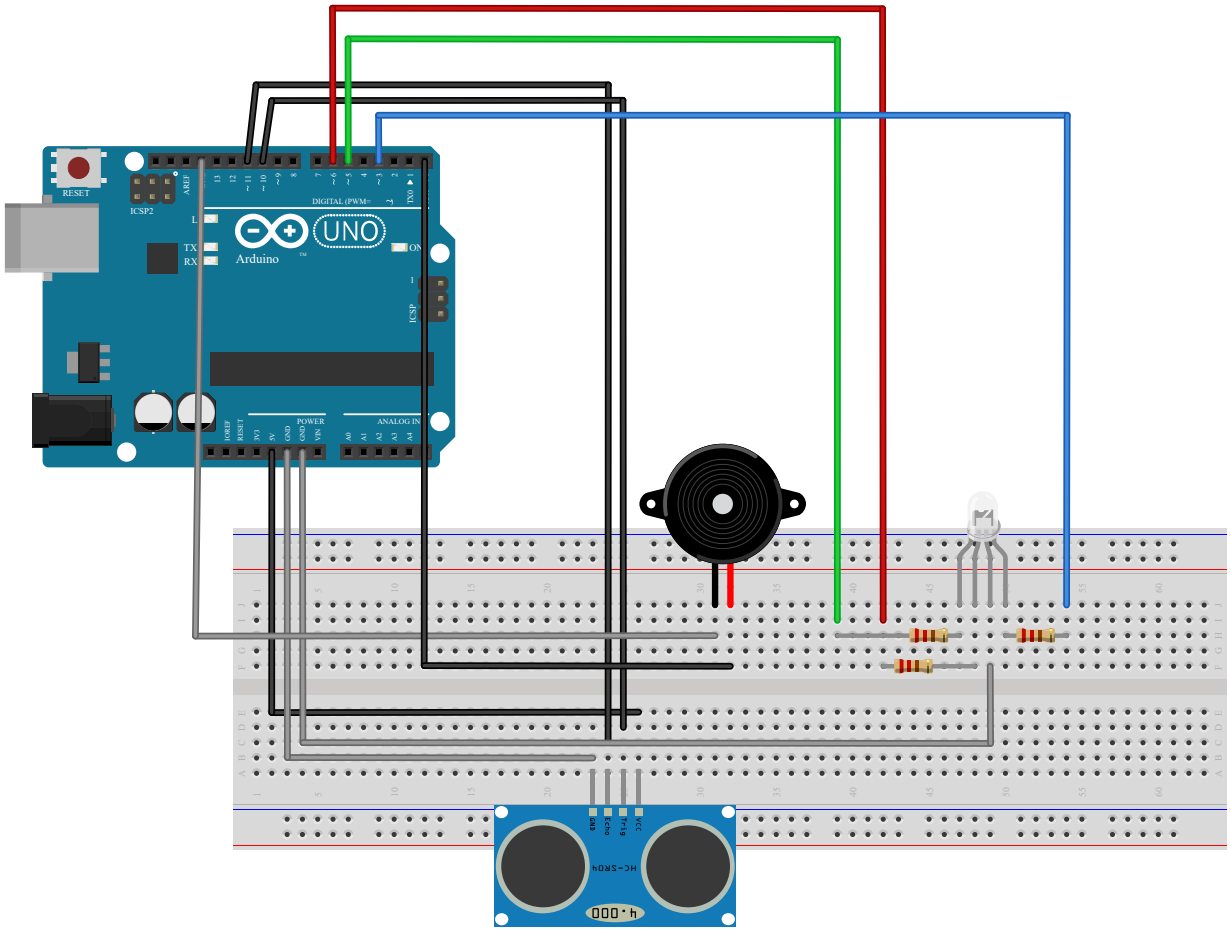
Le câblage n'a pas changé

```
#define trigPin 13
#define echoPin 12
#define ledVerte 5
#define ledRouge 6

int rou = 0;
int ver = 0;

void setup()
{
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(ledVerte, OUTPUT);
    pinMode(ledRouge, OUTPUT);
}
void ledRVBpwm(int pwmRouge, int pwmVerte)
{
    analogWrite(ledRouge, pwmRouge);
```

```
    analogWrite(ledVerte, pwmVerte);
}
void loop()
{
    long duration, distance;
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    duration = pulseIn(echoPin, HIGH);

    distance = duration340 / (210000);

    if (distance < 10)
    {
        digitalWrite(ledVerte, LOW);
        digitalWrite(ledRouge, HIGH);
    }

    else if (distance > 10 && distance < 30)
    {
        rou = 255 - ver;
        ver = 255 * (distance - 10) / 30;

        ledRVBpwm(rou, ver);
    }
    else
    {
        digitalWrite(ledVerte, HIGH);
        digitalWrite(ledRouge, LOW);
    }
    Serial.print(distance);
    Serial.println(" cm ");
    delay(100);
}
```

V. Buzzer

Schéma câblage :



Programme :

(Un peu buggé mais marche quand même)

```
#define trigPin 13
#define echoPin 12
#define ledVerte 5
#define ledRouge 6
#define buzzer 10

int rou = 0;
int ver = 0;

void setup()
{
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(ledVerte, OUTPUT);
    pinMode(ledRouge, OUTPUT);
}
void ledRVBpwm(int pwmRouge, int pwmVerte)
{
    analogWrite(ledRouge, pwmRouge);
    analogWrite(ledVerte, pwmVerte);
}
void loop()
```

```
{
    long duration, distance;
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);
    distance = duration340(210000);

    if (distance < 10)
    {
        digitalWrite(ledVerte, LOW);
        digitalWrite(ledRouge, HIGH);

        void loop();

        tone(buzzer, 1000);
        delay(20);
        noTone(buzzer);
        delay(20);
    }

    else if (distance > 10 && distance < 30)
    {
        rou = 255 - ver;
        ver = 255 * (distance - 10) / 30;
        ledRVBpwm(rou, ver);

        void loop();

        tone(buzzer, 1000);
        delay(ver);
        noTone(buzzer);
        delay(ver);
    }

    else
    {
        digitalWrite(ledVerte, HIGH);
        digitalWrite(ledRouge, LOW);

        void loop();

        tone(buzzer, 1000);
        delay(1000);
        noTone(buzzer);
        delay(1000);
    }
```