# Mise en œuvre d’un capteur à ultrason

## image1) Présentation du module HC-SR04 :

Les caractéristiques techniques du module sont indiquées dans la datasheet du capteur.

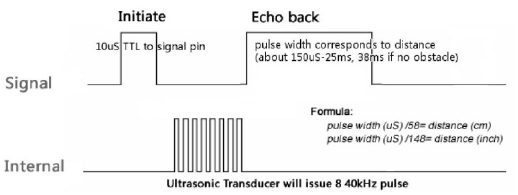
Relevez les caractéristiques :

* Alimentation : 5v
* Consommation en utilisation : 15mA
* Gamme de distance : 2cm-400cm
* Résolution : 3cm
* Précision : 3cm
* Angle de mesure : 15°

Décrivez le principe de fonctionnement du module :

le capteur emet un signal en ultrason pour ensuite attendre le retour du signal réfléchi sur l’obstacle

Voici une représentation graphique de la séquence de fonctionnement du module :



**Questions :**

1. Suite à l’étude de la datasheet du capteur HC-SR04, dites pourquoi l’obstacle est-il signalé au milieu de l’impulsion Echo ?

Le signal Echo est déclenché au moment où le capteur détecte que l'onde réfléchie atteint le capteur. Cela se produit au milieu de l'impulsion Echo parce que le capteur mesure le temps écoulé depuis l'émission du signal ultrasonore jusqu'à la détection de son écho.

*Rappel : Célérité du son dans l’air en m.s-1 : c = 331,2 + 0,607 θ, θ étant la température en °C.*

1. Quelle est la célérité des ultrasons dans l’air à une température de 25 °C ?

346.375m/s

1. Exprimer la formule donnant la distance d en cm pour une durée ΔT en µs de l’impulsion Echo.

ΔT/58=d

1. Application Numérique pour ΔT1 = 150 µs, ΔT2 = 1 ms, ΔT3 = 60 ms.

2.5cm | 17.24cm | hors distance car 60ms sup à 38ms

# 2) Mise en œuvre du module HC-SR04, premières mesures

Réalisez le montage suivant :

Une image contenant Ingénierie électronique, Appareils électroniques, circuit, Composant de circuit

Description générée automatiquement

Pour utiliser le module HC-SR04, notre programme devra suivre la séquence suivante :

* Créer une impulsion de 20 µs (>10 µs) sur la patte Trig
* Mesurer la durée ΔT de l’impulsion Echo en µs
* Calculer la distance d en cm correspondante
* Afficher sur le moniteur série la durée ΔT et la distance d
* Attendre 200 ms avant de réaliser une nouvelle mesure

*Fonction à utiliser :* ***pulseIn****(broche, valeur, delai\_sortie)*

* *broche: le numéro de la broche sur laquelle vous voulez lire la durée de l'impulsion. (type int)*
* *valeur: le type d'impulsion à "lire" : soit HIGH (niveau HAUT) ou LOW (niveau BAS). (type int)*
* *delai\_sortie (optionnel): le nombre de microsecondes à attendre pour début de l'impulsion. La valeur par défaut est 1 seconde. (type unsigned long)*

*Valeur renvoyée :la durée de l'impulsion (en microsecondes) ou 0 si aucune impulsion n'a démarré avant le délai de sortie. (type unsigned long)*

1. Proposez un algorithme permettant de faire le programme correspondant au cahier des charges :

Mettre le pin trig low

attendre 10µs

Mettre le pin trig high

attendre 10µs

Mettre le pin trig low

mesurer la durée de l’impulsion

distance = duree\*0.034/2

renvoyer la distance calculée sur le port série

1. Recopiez le programme répondant à ce cahier des charges, puis le tester.

**/\***

**Pour utiliser le module HC-SR04, notre programme devra suivre le séquentiel suivant :**

**Créer une impulsion de 20 µs (>10 µs) sur la patte Trig**

**Mesurer la durée ΔT de l’impulsion Echo en µs**

**Calculer la distance d en cm correspondante**

**Afficher sur le moniteur série la durée ΔT et la distance d**

**Attendre 200 ms avant de réaliser une nouvelle mesure**

**\*/**

**/\* 1) Zone 1 : la partie déclaration des variables (optionnelle) \*/**

**float Temp = 25; // température en °C**

**int Echo = 2;**

**int Trig = 5;**

**unsigned long dt;**

**float d,c;**

**/\* 2) Zone 2 : la partie initialisation et configuration des entrées/sorties : la fonction setup ()\*/**

**void setup() {**

**pinMode(Trig, OUTPUT); // Initialiser la patte Trig en sortie**

**pinMode(Echo, INPUT); // Initialiser la patte Echo en entrée**

**Serial.begin(9600); // Initialiser la liaison série à 9600 baud**

**}**

**/\* 3) Zone 3 : la partie principale qui s'exécute en boucle : la fonction loop () \*/**

**void loop() {**

**// Créer une impulsion de 20 µs (>10 µs) sur la patte Trig**

**digitalWrite(Trig, HIGH);**

**delayMicroseconds(20);**

**digitalWrite(Trig, LOW);**

**// Mesurer la durée ΔT de l’impulsion Echo en µs**

**dt = pulseIn(Echo, HIGH);**

**// Valeur renvoyée : 0 si aucune impulsion n'a démarrée avant le délai de sortie (1 s par défaut)**

**if (dt==0) Serial.println("Mesure hors limite");**

**else {**

**// Calculer la distance d en cm correspondante**

**c=331.2+0.607\*Temp;   // Caldcul de la vitesse du son avec correctif Température**

**d=c/2\*dt/10000;      // c en m/s dt en µs d en cm**

**// Afficher sur le moniteur série la durée ΔT et la distance d**

**Serial.print("dt = ");Serial.print(dt);Serial.println(" µs");**

**Serial.print("d = ");Serial.print(d,1);Serial.println(" cm");**

**}**

**// Attendre 200 ms avant de réaliser une nouvelle mesure**

**delay(200);**

**}**

1. Visualiser sur l’oscilloscope les signaux des pattes Trig et Echo.

Mettre les captures de l’oscilloscope des voies A (Trig) et B (Echo)

1. Mesurer la durée des impulsions Trig et Echo. Comparer ces durées à celles attendues.

Width trig : 200µs

width echo @140cm : 8.4ms

1. Comparer la mesure d à la distance mesurée par une règle. Justifiez si nécessaire la différence de mesure.

La mesure est correspondante à environ 1cm

# 3) Etape 2 : afficher la distance sur un écran LCD

1. Afficher sur l’écran LCD la distance d, ou le texte : « Mesure hors limite ».

Voir le TP1 – I2C pour le branchement et le programme à utiliser.

Proposez un montage de l’ensemble Capteur HC-SR04 / Arduino / Ecran LCD :

# 4) Précision du capteur

On souhaite mesurer la précision du capteur pour se faire, nous allons procéder à plusieurs mesures avec un obstacle. Remplissez le tableau suivant :

|  |  |  |  |
| --- | --- | --- | --- |
| **Distance en mm** | **Mesure affichée** | **Erreur en mm** | **Erreur en %** |
| **10** |  |  |  |
| **20** |  |  |  |
| **30** |  |  |  |
| **40** |  |  |  |
| **50** |  |  |  |
| **60** |  |  |  |
| **70** |  |  |  |
| **80** |  |  |  |
| **90** |  |  |  |
| **100** |  |  |  |
| **150** |  |  |  |
| **200** |  |  |  |
| **250** |  |  |  |
| **300** |  |  |  |

Tracez les courbes « théorique » et des mesures (les valeurs mesurées en abscisses et les valeurs théoriques en ordonnées).

Tracez la courbe des erreurs en pourcentage (les distances en abscisses et les pourcentages en ordonnées).

Analysez les graphiques :

le pourcentage d’erreur diminue avec la distance