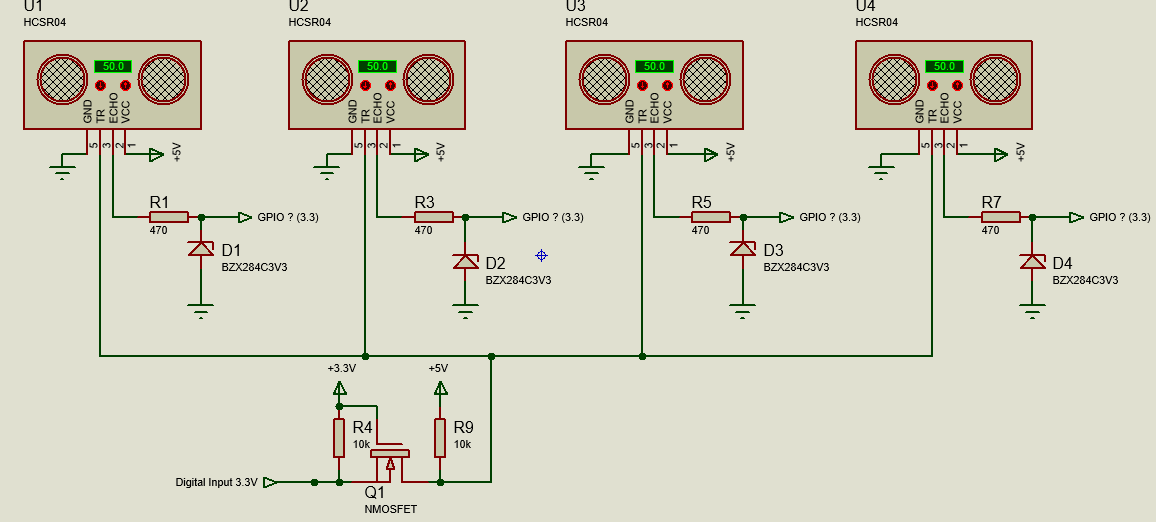
# **Résumé HCSR04 x4**

1. **HC-SR-04**
2. La tension d’alimentation est 5V
3. La communication est 5V. On vient ajuster les tensions à l’aide d’un circuit.
4. Code + Circuit



**Code d’exemple « HC-SR-04-Wallee.ino » (avec fonctions générales)**

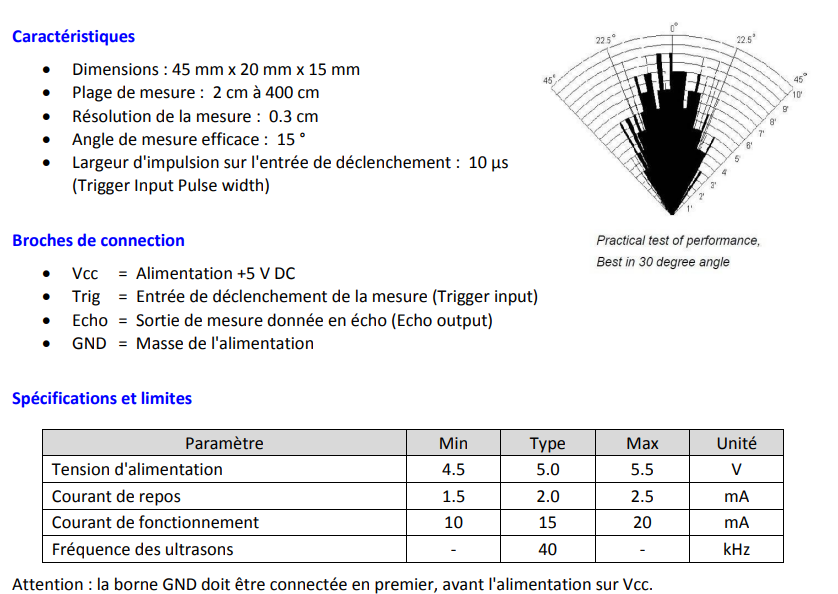
**Code propre pour utilisation sur programme final : « HC-SR-04-wallee-propre »**

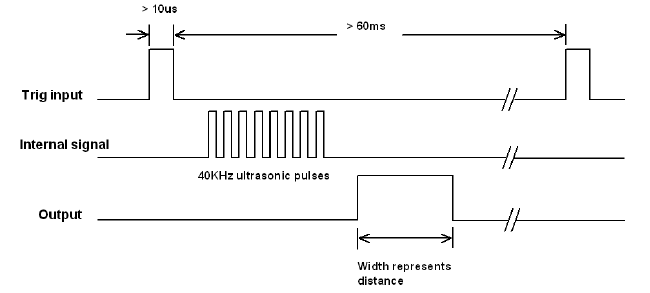
**Code de temps d’éxecution « HCSR -temps-execution.ino » (programme de temps d’éxecution)**

1. **Principe de fonctionnement**

Ecriture à l’état haut sur le trigger (commun à tous les capteurs) : émission 8 signaux ultrasons, on peut alors lire le capteur qui renvoi une durée d’impulsion (echo) qui détermine la distance parcourue.

Super article : <https://www.gotronic.fr/pj2-hc-sr04-utilisation-avec-picaxe-1343.pdf>





1. **Pins**

|  |  |
| --- | --- |
| 1. **+5V** | **+5V** |
| 1. **Trig** | **26 (ESP) 5 (ARDUINO)** |
| 1. **Echo** | **35,34,39,36 (ESP) 6, 7, 8, 9 (ARDUINO)** |
| 1. **GND** | **GND** |

**Librairies à utiliser**

**#include <HCSR04.h>**

**Initialisation HC-SR04 x4(ARDUINO)**

**#define HC\_trig 26** //On définit les broches des capteurs

**#define HC1 35**

**#define HC2 34**

**#define HC3 39**

**#define HC4 36**

**HCSR04 hc\_front(HC\_trig, HC1);** //On crée 4 objets correspondants aux capteurs

**HCSR04 hc\_right(HC\_trig, HC2);**

**HCSR04 hc\_left (HC\_trig, HC3);**

**HCSR04 hc\_back (HC\_trig, HC4);**

**int erreur\_ultrason=0;** //Nombre d’erreur de mesure

**int lim\_err\_HC=5;** //Limite d'erreur avant reset des capteurs

**int front=0; int right=1; int left=2; int back=3;** //Rend intuitif l’appel des fonction: « ultrason\_read(right) »

**int measure\_time=60;** //Délai entre 2 mesures, recommandé à 60ms

**int decharge\_time=100;** //Délai pour décharger les capteurs à l’initialisation/si trop d’erreur

**Initialisation LED (ARDUINO)**

**#define led\_verte 32**

**#define led\_rouge 33**

**#define led\_blanche 25**

**Fonctions à utiliser**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nom fonction** | **Rôle** | **Pire temps d’exécution (déconnecté)** | **Meilleur temps d’exécution** |
| void Ultrason\_Setup() | Setup des 4 capteurs | **120ms/120ms** | **100ms** |
| double Ultrason\_Read(front/left/right/back) | Lecture distance 1 capteur | **199ms/1120ms** | **1ms** |
| int Ultrason\_Indication(front/left/right/back) | Lecture indication 1 capteur | **199ms/1120ms** | **1ms** |
| double\* Ultrason\_All\_Read() | Lecture distance 4 capteurs | **2318ms/4180ms** | **184ms** |
| int\* Ultrason\_All\_Indication () | Lecture indication 4 capteur | **1517ms/4300ms** | **184ms** |
| void ultrason\_ERREUR() | Réinitialise les capteurs + allume LED verte et rouge  (commandé par Indication) | **125ms/125ms** | **120ms** |

**Fonction Setup - Fonction à utiliser – Fonction interne/peu probable à utiliser**

**NOTE IMPORTANTE –**

Après avoir utilisé « Ultrason\_All\_Read() » il faut « **delete** distante » pour libérer l’espace mémoire

Après avoir utilisé « Ultrason\_All\_Indication » il faut « **delete** indication » pour libérer l’espace mémoire.

Il faut espacer deux mesures sur un même capteur de 60ms ! C’est la variable « measure\_time ».

Les fonctions « Ultrason\_Read » et « Ultrason\_All\_Read » ne peuvent pas lancer la fonction d’erreur automatiquement. Seules les fonctions « Ultrason\_Indication » et « Ultrason\_All\_Indication » le font.