CH-1092 Belmont, Switzerland
Tel +41 21 728-6156
Email info@didel.com, www.didel.com

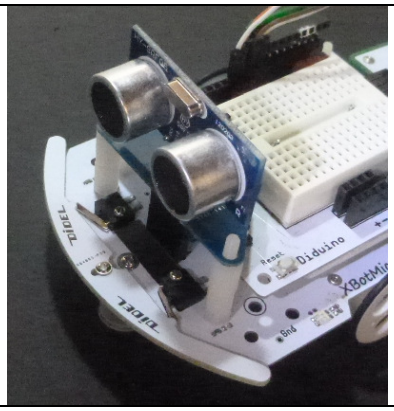
https://www.didel.com/prof/Xuson.pdf

# Capteur de distance ultrasons xUson

***Aspects pédagogiques***

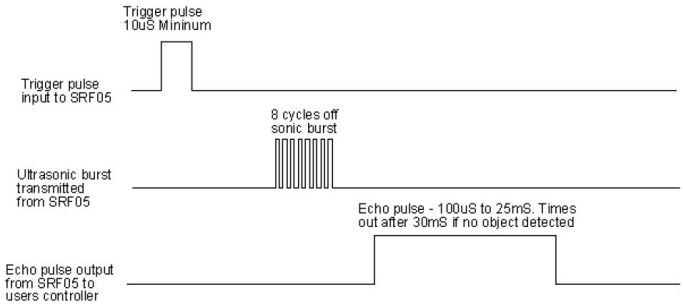
*Liens aux docs sous prof/Xuson.html*

Le circuit avec son capteur SR05 se fixe sur le connecteur avant. Il est très simple à utiliser, mais ses limitations doivent être comprises.
La fonction Arduino est bloquante, parfois même assez longtemps, et nous a encouragé à documenter une machine d'état appelée par interruption toutes les 58 microsecondes. Cette valeur particulière donne la distance en centimètres sans calcul de conversion. Cette jolie approche n'a probablement pas sa place au début de l'utilisation du Xbot. Il y a assez à mesurer et à réfléchir : effets parasites des échos, angle de vision, réaction du robot.



Le circuit SR05 a 5 broches: Gnd, Vcc (3.5V à 5.5V), Trig ( une impulsion de 10 us déclenche l'envoi d'un train d'impulsions à 40 kHz) et Echo (impulsion positive pendant le temps de vol, durée max en principe 200 ms s'il n'y a pas d'echo).
Sur le Xbot, Trig est connecté à la pin 15(A1) et Echo à la pin 14(A0) en mode digital.
Une impulsion sur Trig déclenche l'envoi de 8 impulsions à 40 kHz et active le signal Echo, qui est désactivé quand l'écho revient (min 2cm 100 us). La durée d'un aller-retour pour une distance de 25cm est environ 1,3 ms.

Si la distance est trop grande, le signal reste à 1. Lorsque Echo est retourné à zéro, il faut attendre au moins 30-40 ms avant de réactiver Trig. Si les mesures ne sont pas stables, augmenter ce temps à 50, voire 100ms.
Pour mesurer la durée de l'Echo, Arduino a une fonction bloquante qui rend cette valeur et est valable pour n'importe quelle pin.
`pulseIn(Pin,HIGH);` rend la durée pendant laquelle la pin est HIGH.




Le programme est facile à comprendre. La fonction GetUson() rend la durée en microsecondes. Elle est trop simple pour la mettre dans un fichier inclus.

```
//XusonTest1.ino
#define Trig A1
#define Echo A0
#include "Tell.h"
void setup() {
  pinMode(Echo, INPUT);
  pinMode(Trig, OUTPUT);
  Serial.begin(9600);
  SetupTell();
}
```

*Note: si Tell n'est pas installé cela ne perturbe pas l'affichage su le terminal.*

```
int GetSonar () {
  int temp;
  digitalWrite(Trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig,LOW);
  temp= pulseIn(Echo,HIGH);
  return temp;
}
int dist;
void loop() {
    delay (100) ; // l'écran perd aussi du temps
    dist= GetSonar ();
    Serial.println(dist);
    Tell(dist);
}
```

Quelles cibles utiliser pour les tests? Ne pas oublier de vérifier l'effet miroir.

## Oled

Si le Oled est utilisé, voir OledUson.pdf. Un exercice non mentionné est de dessiner le plan de la pièce, robot au centre. Il n'y a pas d'odométrie; un premier exercice est des déterminer la durée d'un tour à partir du minimum et/ou maximum de la distance.

## Asservissement

Comme exemple de programme à inventer, maintenir le robot à une distance donnée d'un mur permet de comprendre le problème des asservissements. On joue avec des zones de réglage et de vitesse et observe les oscillations ou erreurs de positionnement. Voir LibXUson.pdef et le programme KeepDistOledUson.ino dans le Zip

## Références – cliquables dans *Xdist2IrProf.html*

| | |
|---|---|
| Xdist2IrProf.zip | Les programmes de cette notice |
| DistSonar.pdf | Documentation 2015 |
| OledUson.pdf | Plus complete avecOled 2017 |
| LibXUson.pdf | Utilisation de plusieurs librairies 2015 |
| TestSonar.pdf | Premier document périmé 2013 |

Jdn 180813