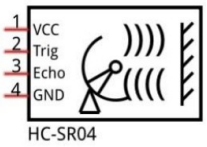
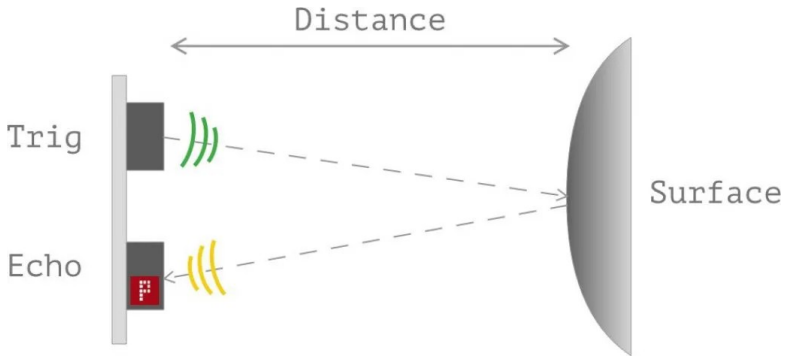
Université Tunis El-Manar	**Année Universitaire** : 2023-2024
Faculté des Sciences de Tunis	**Module :** Conception des objets connectés
Section : LIOT3	**Enseignant :** C.A. ABID

TD 1

Exercice 1.
On se propose d'utiliser le capteur HC-SR0 pour réaliser des mesures de distance assez précise.



Le HC-SR04 est un capteur qui utilise deux signaux numériques. Le premier signal appelé Trig (Trigger) génère un faisceau d'ultrasons. Les ultrasons sont réfléchis par la surface de l'objet (ou la personne) dont on souhaite connaitre la distance. C'est le signal Echo que l'on récupère sur la seconde broche du capteur.



Connaissant le temps entre l'émission du signal généré en excitant la sortie Trig et la réception Echo ainsi que la vitesse de propagation du son dans l'air on peut en déduire la distance à l'aide de la formule suivante :

Distance = temps / 2 * vitesse du son dans l'air, où la vitesse de son est 340m/s

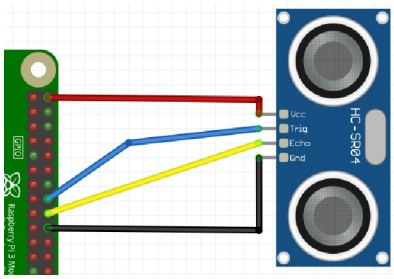
Caractéristiques techniques du HC-SR04

- Plage de mesure de distance : 2cm à 200cm
- Précision de mesure : 0,3cm
- Tension d'alimentation : 5V
- Sortie numérique : PWM

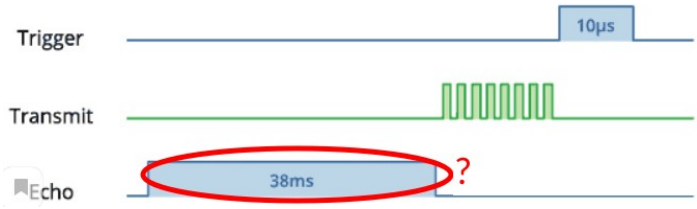
•Poids : 9g

•Dimensions : 45mm x 20mm x 18mm

Nous considérons le montage de la figure ci-après en connectant la broche Trig (resp. Echo) du capteur à la broche BCM23 (resp. BCM24) de la raspberry Pi 3.



1) Écrire la fonction uint32_t pulseIn(int pin, uint32_t timeout) qui mesure et renvoie la durée en micro-secondes de la réception d'une pulsation sur l'entrée pin (connectée à Echo). Elle renvoie 0, si le temps de mesure dépasse timeout.



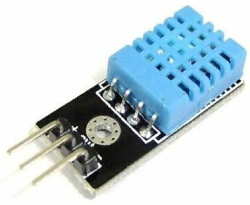
NB. : on utilise la fonction uint32_t gpioTick() qui renvoie le temps courant en microsecondes

2) Écrire la fonction float getSonar() qui renvoie en centimètre la distance entre le capteur et on objet

3) Écrire le cors de la fonction main permettant d'afficher la distance entre un objet et le capteur en centimètre

Exercice 2.

Le capteur DHT11 est lui capable de mesurer des températures de 0 à +50°C avec une précision de +/- 2°C et des taux d'humidité relative de 20 à 80% avec une précision de +/- 5%. Une mesure peut être réalisée toutes les secondes.



Les capteurs DHTxx communiquent avec le microcontrôleur via une unique broche d'entrée / sortie.

Le brochage du capteur est le suivant :

•La broche n°1 est la broche d'alimentation (5 volts ou 3.3 volts).

•La broche n°2 est la broche de communication. Celle-ci doit impérativement être reliée à l'alimentation via une résistance de tirage de 4.7K ohms (il s'agit d'une sortie à collecteur ouvert).

•La broche n°3 n'est pas utilisée et ne doit pas être câblée.

•La broche n°4 est la masse du capteur (GND).

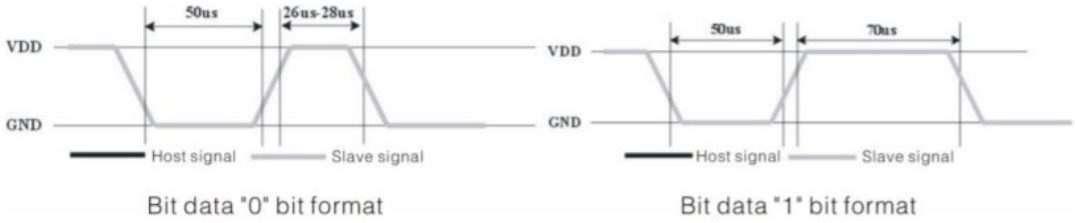
La communication avec un capteur DHTxx se fait en 3 étapes :

• Tout d'abord, le microcontrôleur maître réveille le capteur en plaçant la ligne de données à LOW pendant au moins 18ms pour le DHT11. Durant ce laps de temps, le capteur va se réveiller et préparer une mesure de température et d'humidité. Une fois le temps écoulé, le maître va libérer la ligne de données et passer en écoute. Le timeout pour recevoir une réponse est de 1000 us.

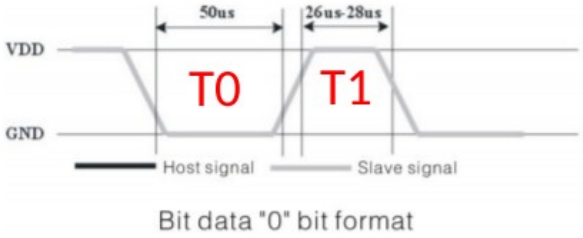
• Une fois la ligne de données libérée, le capteur répond au maître (pour montrer qu'il est bien réveillé) en maintenant la ligne de données à LOW pendant 80µs puis à HIGH pendant 80µs.

• Le capteur va ensuite transmettre une série de 40 bits (5 octets). Les deux premiers octets contiennent la mesure de l'humidité. Les deux octets suivants contiennent la mesure de la température et le cinquième octet contient une somme de contrôle qui permet de vérifier que les données lues sont correctes.

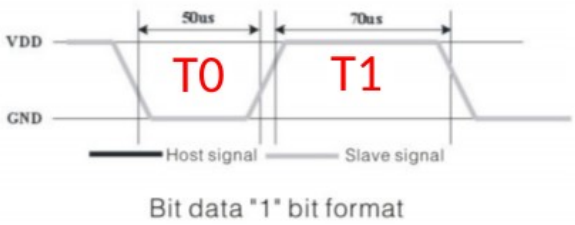
Le DHT envoie la valeur de l'humidité et celle de température sur 40 en série en commençant par le bit le plus significatif. Chaque bit de données commence par le niveau de tension basse 50us et la longueur du signal de niveau de tension élevé suivant détermine si le bit de données est à "0" ou "1" comme l'illustre la figure ci-après.



Comme le montre la figure ci-après, pour identifier le bit 0, il suffit de vérifier que la durée T1<T0.



De même manière, pour identifier le bit 1 provenant du capteur de température, il suffit de vérifier que T1>T0.



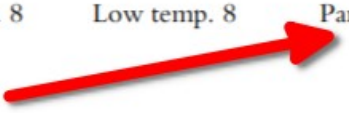
Le format d'une mesure envoyée par le capteur est illustrée par l'exemple suivant :

Example .: 40 data is received:

0011 0101	0000 0000	0001 1000	0000 0000	0100 1101
High humidity 8	Low humidity 8	High temp. 8	Low temp. 8	Parity bit

Calculate:

0011 0101+0000 0000+0001 1000+0000 0000= 0100 1101



On utilise la fonction uint32_t gpioTick() qui renvoie le temps courant en microsecondes.

On se propose d'écrire un programme qui affiche à chaque seconde la température et l'humidité capturées à partir du capteur DHT11.

1) Développer chacune des fonctions ci-après sachant qu'on ne s'intéresse pas au calcul de la parité.

```
/*
 * Indiquer au capteur qu'on va lire une donnée (40 bits)
 */
void SendStartSignal();
/*
 * Lancer une mesure de la température et de l'humidité
 */
std::tuple<uint8_t, uint8_t> Measure();

/*
 * Attendre que la ligne donnée soit mise à 0 par le capteur. Elle retourne le       *
temps d'attente en microsecondes
 */
int WaitForLow();
/*
 * Attendre que la ligne donnée soit mise à 1 par le capteur. Elle retourne le       *
temps d'attente en microsecondes
 */
int WaitForHigh();
/*
* À partir d'une donnée renvoyée par le capteur, elle retourne sous forme d'un      *
couple la valeur de la température et la valeur de l'humidité
*/
std::tuple<uint8_t,uint8_t> ProcessData(uint64_t Data);
```

2) Écrire le programme principal pour afficher la température et l'humidité sur la console. On effectue une mesure à chaque trois secondes.