INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Projet MOSH

Réalisation d'un robot "intelligent" : ***QuanticAnt***

BAUDEAU Nicolas - MARTINEZ Quentin - YJJOU Soufian

Sommaire

Présentation du projet

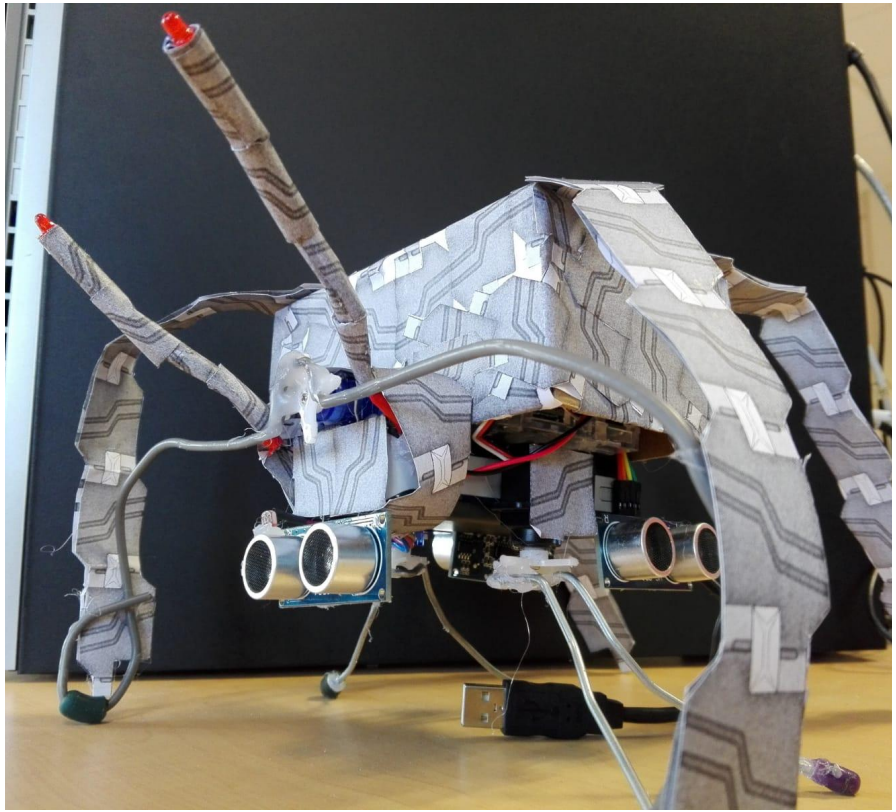
Objectifs

Idée de départ : Robot imitant le comportement d'un insecte qui a peur de la lumière

Fonctions principales de QuanticAnt :

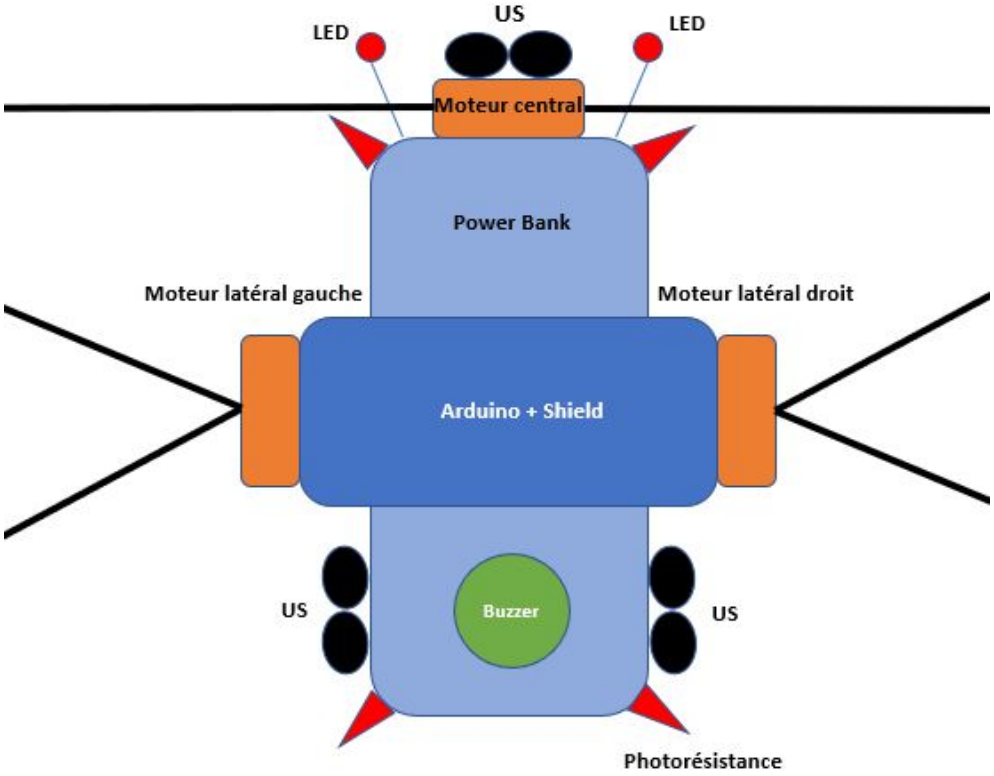
- ➔ Éviter les obstacles proches
- ➔ Gestion de la luminosité (photosensible)
- ➔ Déplacement aléatoire sur pattes mécaniques
- ➔ S'immobilise et chante en cas de faible luminosité (mode veille)

Résultat



Hardware

Vue globale de QuanticAnt



Moteurs latéraux ➡ Déphasage pour permettre le mouvement

Moteur central ➡ Synchronisé pour permettre le réarmement des pattes latérales

Module ultrason ➡ Détection d'obstacles

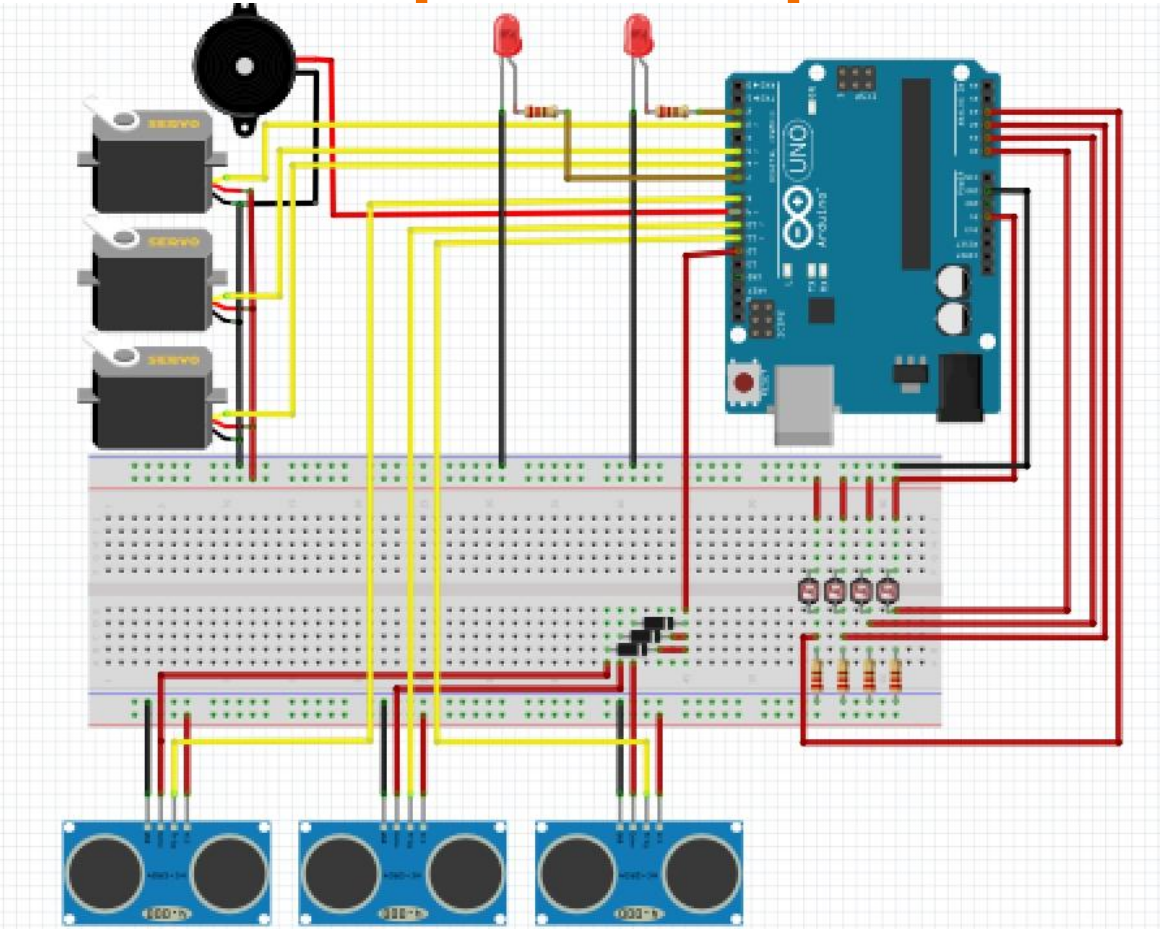
Photorésistance ➡ Détection de lumière

Buzzer ➡ Génération du bruit

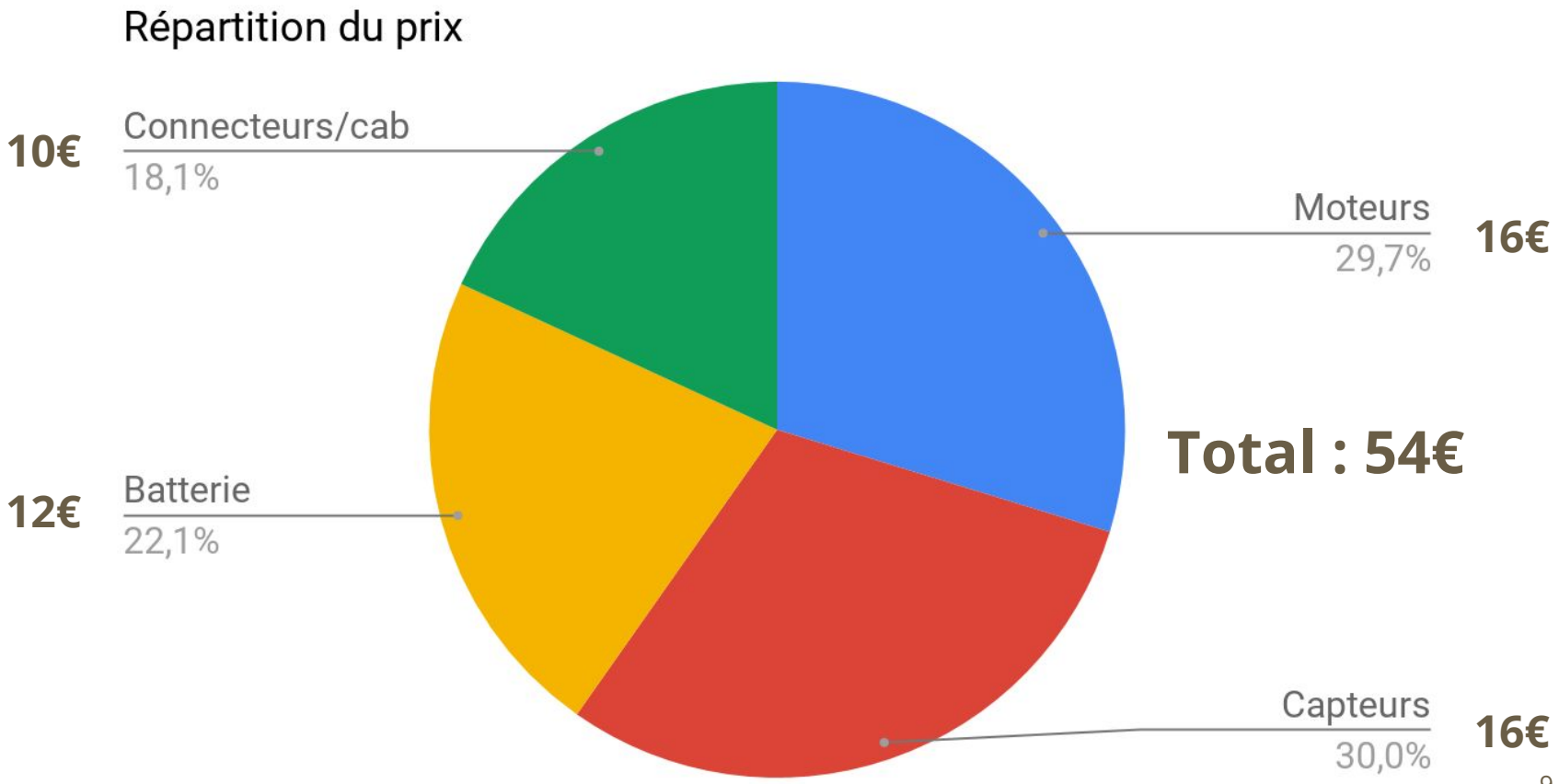
Led ➡ Antennes clignotantes

Batterie ➡ Alimentation Arduino UNO + Alimentation moteur

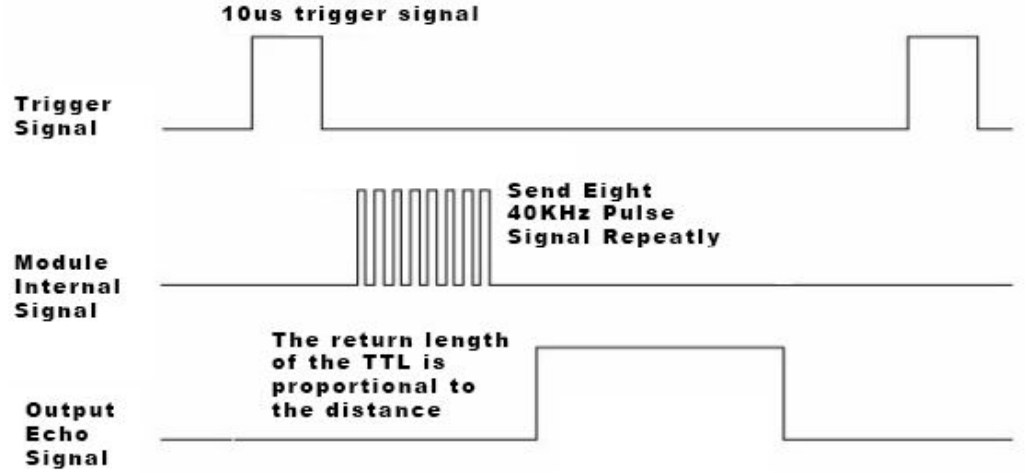
Composants et prix



Composants et prix



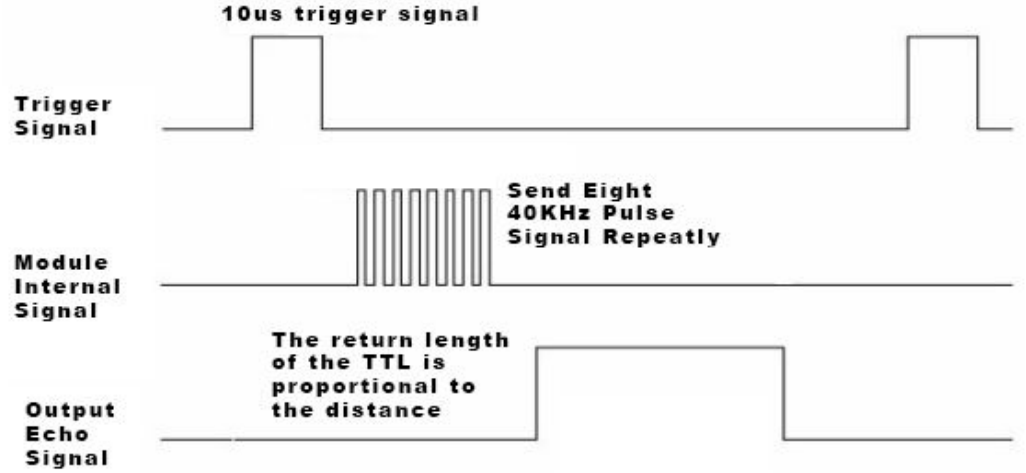
Utilisation des modules ultrason



2 pins logiques par capteur.

- Un déclencheur (TRIG)
- Un echo (ECHO)

Utilisation des modules ultrason



2 pins logiques par capteur.

- Un déclencheur (TRIG)
- Un echo (ECHO)

Distance donnée par la durée du signal Echo et la vitesse du son.

Astuce d’utilisation des modules ultrason

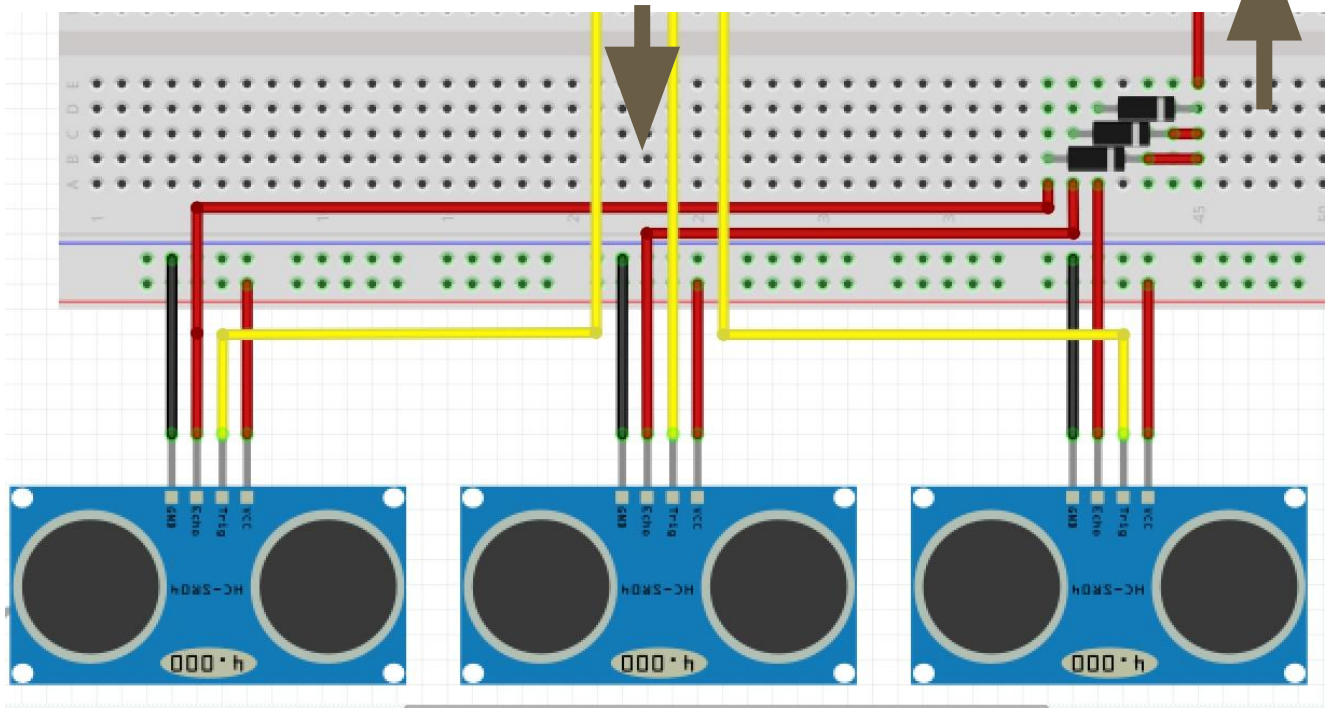
6 pins logiques sur l’arduino…

4 pins logiques sur l’arduino…

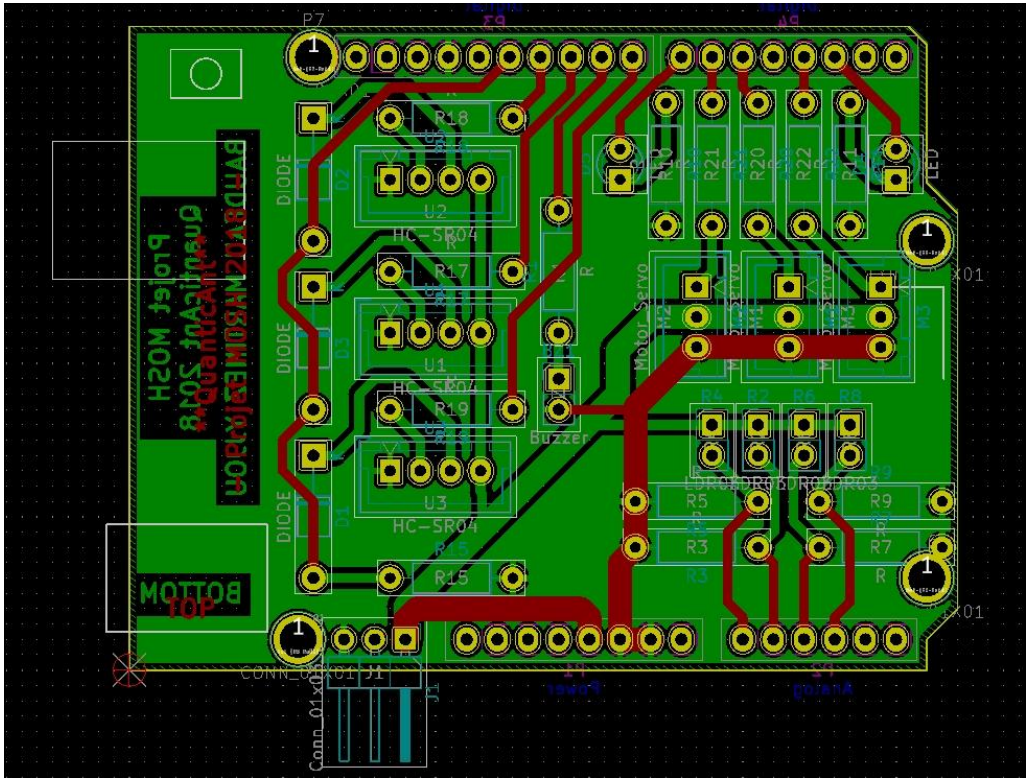
Astuce d'utilisation des modules ultrason

3 TRIG

1 seul ECHO commun vers arduino (PullDown) !



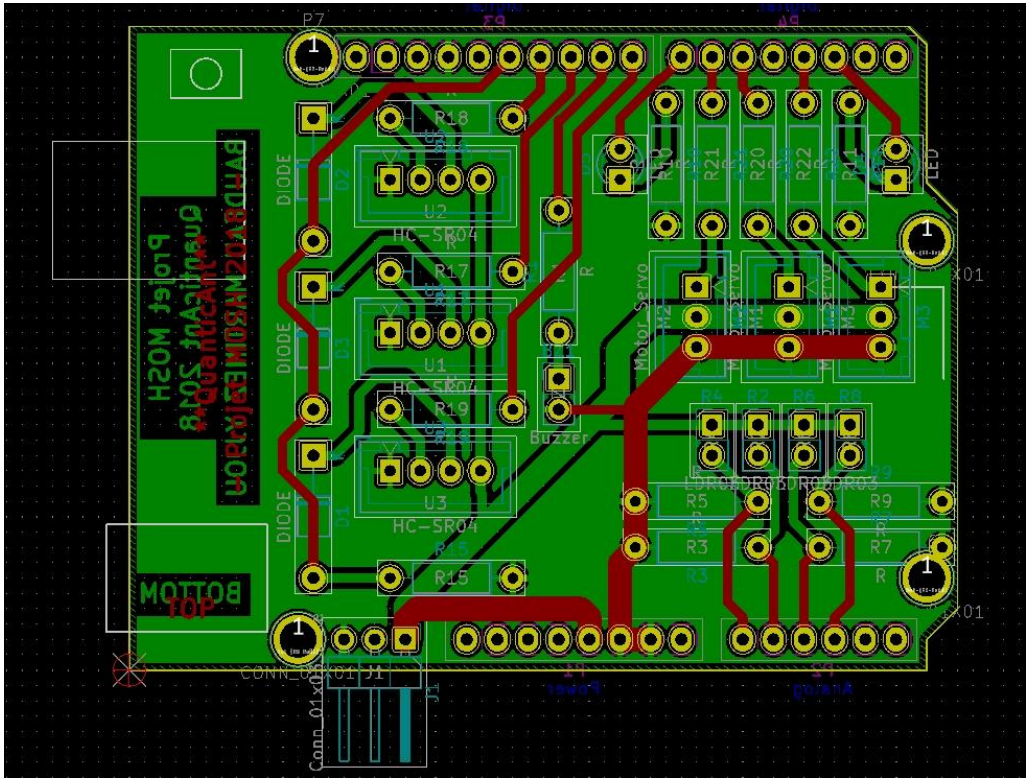
La conception du PCB (Kicad)



PCB double face pour :

- Connecter les actionneurs / capteurs
- Contenir les ponts diviseurs des capteurs
- Le système d'Echo commun

La conception du PCB (Kicad)



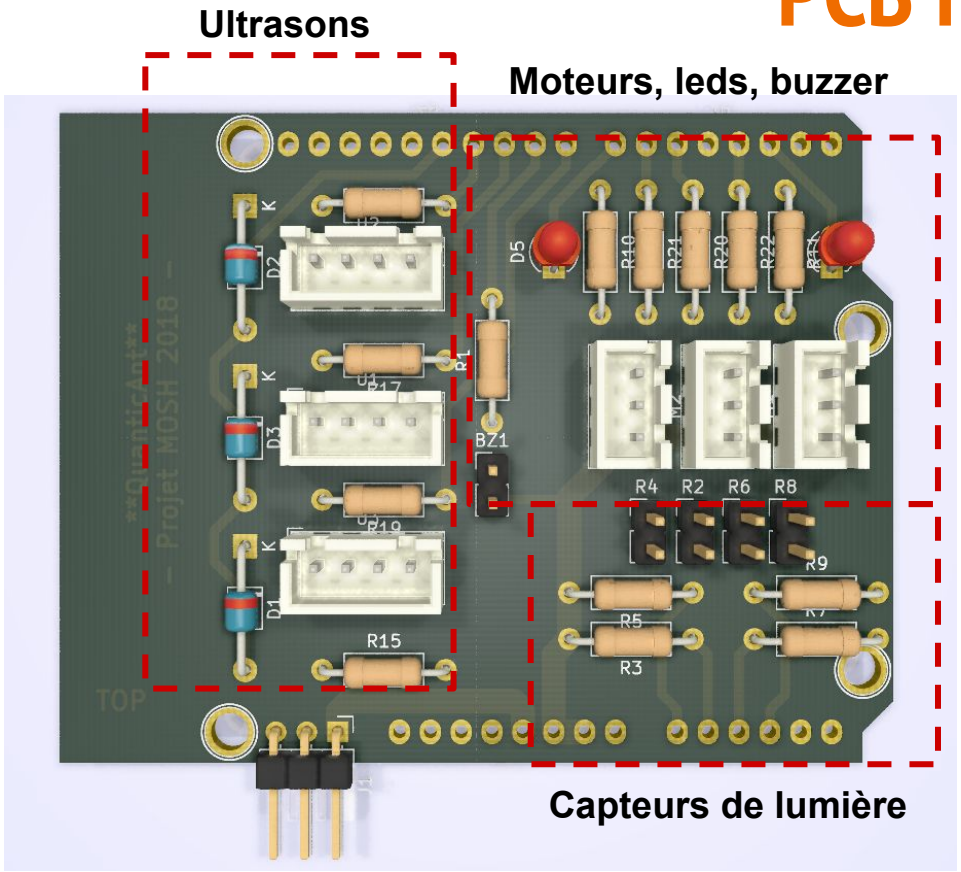
PCB double face pour :

- Connecter les actionneurs / capteurs
- Contenir les ponts diviseurs des capteurs
- Le système d'Echo commun

Astuces :

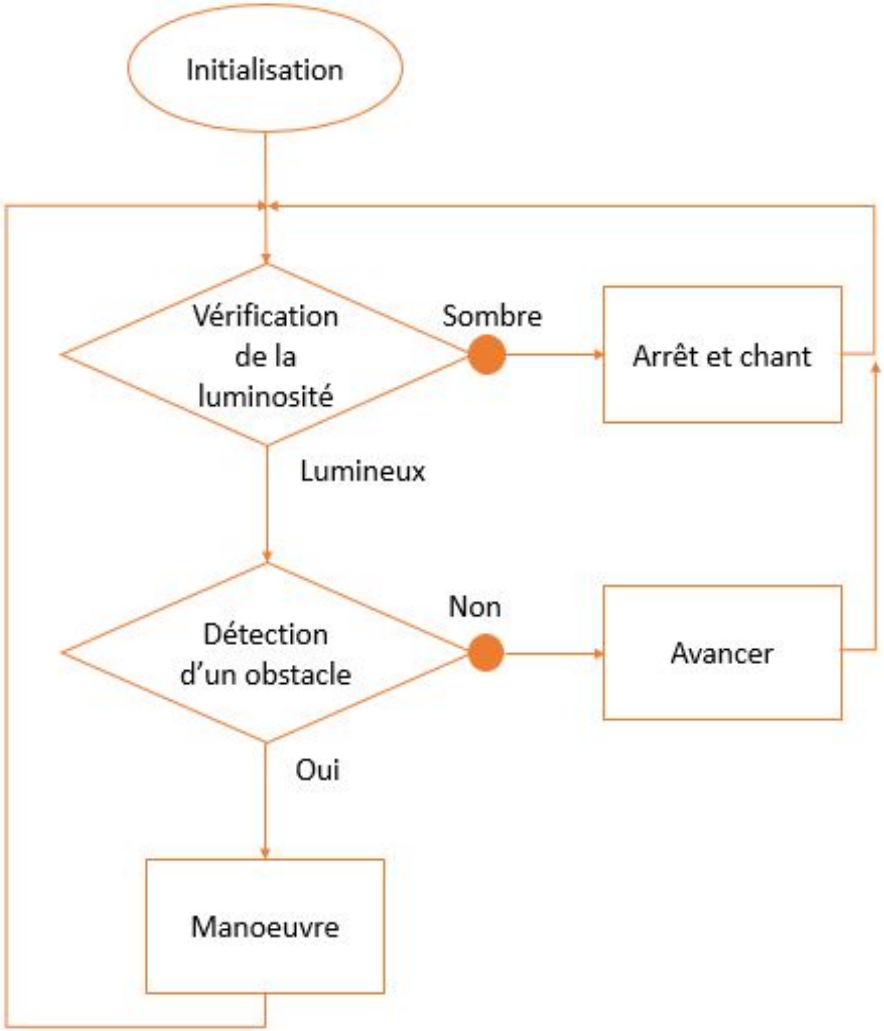
- Indiquer sur les 2 faces "TOP" et "BOTTOM"
- Grossir les pistes des moteurs (puissance)
- Utiliser les pattes des composants comme via

PCB final



Découpage fonctionnel du PCB

Software



Logique de bord

Logigramme général du robot

Détection d'un obstacle

- ➔ Retourne la distance en mm entre le capteur "Broche" et un obstacle à moins de 2m (dure 6ms environ)
- ➔ Signal = Temps (en ms) entre l'envoie de l'ultrason et ping de réception de l'écho
- ➔ Conversion du temps en distance 343 m/s soit 0.0343 mm/s = 1/29.1

```
float DistUS(int Broche)
{
        float Dist = 0.0;

//Impulsion pour le capteur ultrason

        digitalWrite(Broche, LOW);
        delayMicroseconds(5);
        digitalWrite(Broche, HIGH);
        delayMicroseconds(10);
        digitalWrite(Broche, LOW);

// Lecture du signal détecté par le capteur ultrason.

        pinMode(ECHO, INPUT);
        Dist = (pulseIn(ECHO, HIGH, 15000)/2) /
        29.1;

        if(Dist == 0)
        return ( 200.0);
        else
        return (Dist);
}
```

Détection de la luminosité

- ➔ Retourne la luminosité la plus forte parmi les 4 capteurs

- ➔ LUXF = Lumière capté devant le robot
- ➔ LUXR = Lumière capté à droite du robot
- ➔ LUXG = Lumière capté à gauche du robot
- ➔ LUXB = Lumière capté derrière le robot

```
int MaxLux()
{
    int maxL = analogReadN(LUXF, 10);
    if( analogReadN(LUXL, 10) > maxL)
    maxL = analogReadN(LUXL, 10);

    if( analogReadN(LUXR, 10) > maxL)
    maxL = analogReadN(LUXR, 10);

    if( analogReadN(LUXB, 10) > maxL)
    maxL = analogReadN(LUXB, 10);
    return maxL;
}
```

Fonction motrice

- On contrôle la (phase) des moteurs en degré (0-180°)
- La valeur Dir donne la direction (8 = avancer, 4 et 6 = tourner, 2 = reculer)
- A = moteur de droite
  B = moteur de gauche
  C = moteur central
- La phase A et B oscillent entre 60° et 130° (pas de 4°)

```
void Motrice(int vitesse, int dir)

{
        if(posA<=59)
        sensA =4;
        if( posA >= 130)
        sensA = -4;

        if(posB<=59)
        {
                sensB =4;

//Contrôle de la phase C au moment où B est en butée

                if(dir ==8 || dir == 6)
                posC = 70;
                else // Dans le cas de la marche avant ou rotation Droite on inverse le mouvement
                posC = 110;
        }
```

Fonction motrice

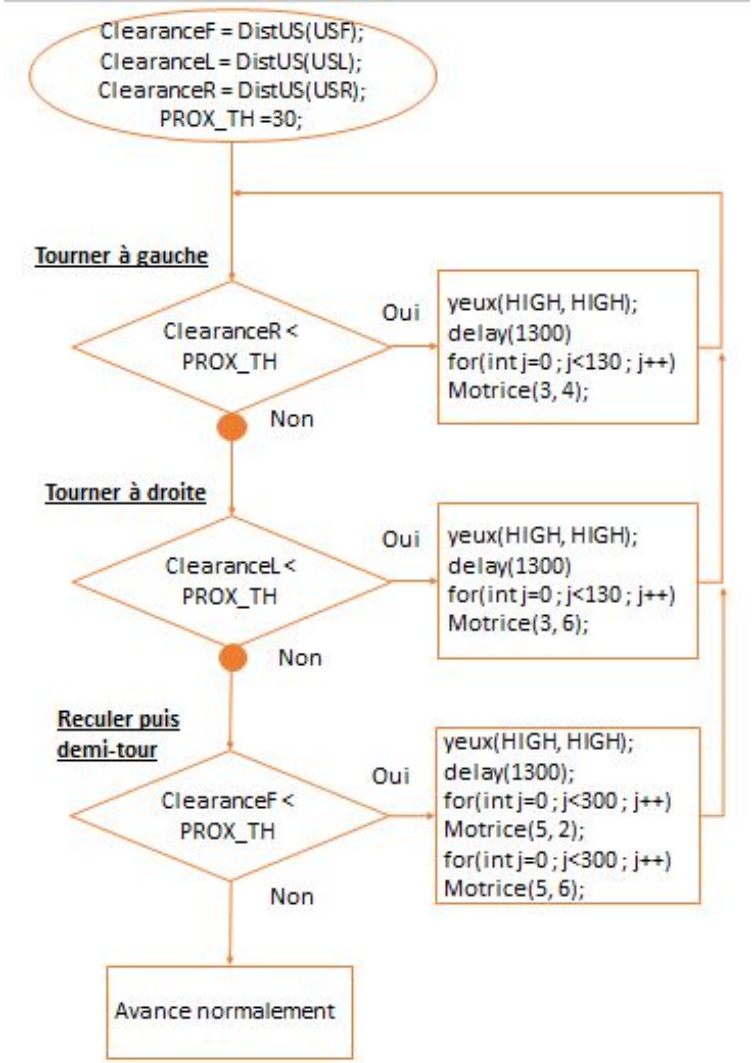
- La direction désirée est obtenue en corrigeant la position des moteurs A et C en fonction de la position du moteur B
- La vitesse est déterminé par le temps d'attente entre deux mouvements

```
        if( posB >= 130)
        {
                sensB = -4;

//Correction du déphasage avec A selon la direction
                if(dir ==8 || dir == 2)
                posA = 59;
                if(dir ==4 || dir == 6)
                posA = 130;

//Correction du déphasage avec C selon la direction
                if(dir == 2 || dir == 4)
                posC = 70;
                else
                posC = 110;
        }

        posA+=sensA;
        posB+=sensB;
        phaseA.write(posA);
        phaseB.write(posB);
        phaseC.write(posC);
        delay(80/vitesse);
}
```



Gestion des collisions

Organigramme de déplacement du robot

Conclusion

Difficultés rencontrées

La mécanique des pattes

L’identification des directions lumineuses dans son environnement (manque de contraste)

Le dimensionnement des moteurs

Perspectives d'évolution

Amélioration de la fonction motrice et de la mécanique des pattes

Création de la fonction de détection de la direction lumineuse (augmentation de la résolution du système)

Impression 3D du corps de QuanticAnt

Merci de votre attention
Démonstration ?