

L'AIRSER



BEDIA Julien, CAPPIELLO Raphaël,
KHAMMA Houssameddine, N'GUYEN VAN KY Nicolas

SOMMAIRE

Introduction	3
Définition du besoin	4
Emetteur	5
Prototype	5
Composants	5
Récepteur	6
Prototype	6
Composants	6
Programmation	7
Arduino émetteur	7
Arduino récepteur	7

Introduction

Le développement de la technologie des drones a révolutionné la manière dont les militaires mènent des opérations sur le terrain. L'utilisation de drones permet aux forces armées de surveiller des zones dangereuses ou inaccessibles, de recueillir des informations sur l'ennemi et de mener des attaques ciblées. Dans le cadre de ce sujet, l'armée dispose de deux drones, l'un en l'air et l'autre au sol, pour effectuer des tirs sur des cibles désignées. Le défi consiste à déterminer quel drone a effectué les tirs, afin de pouvoir attribuer la responsabilité de l'attaque. Dans cet environnement militaire, il est crucial de pouvoir détecter et identifier avec précision les acteurs impliqués dans une attaque, afin de garantir la sécurité des troupes et de minimiser les dommages collatéraux.



Définition du besoin

Nous avons besoin de distinguer deux types de drones, les drones aériens et les drones terrestres, en fonction de la trame de données qu'ils envoient à nos récepteurs. Pour cela, nous aurons besoin d'un émetteur prototype équipé d'un laser pour envoyer la trame de données et d'un récepteur unique prototype qui décodera la trame de données reçue pour identifier le drone..

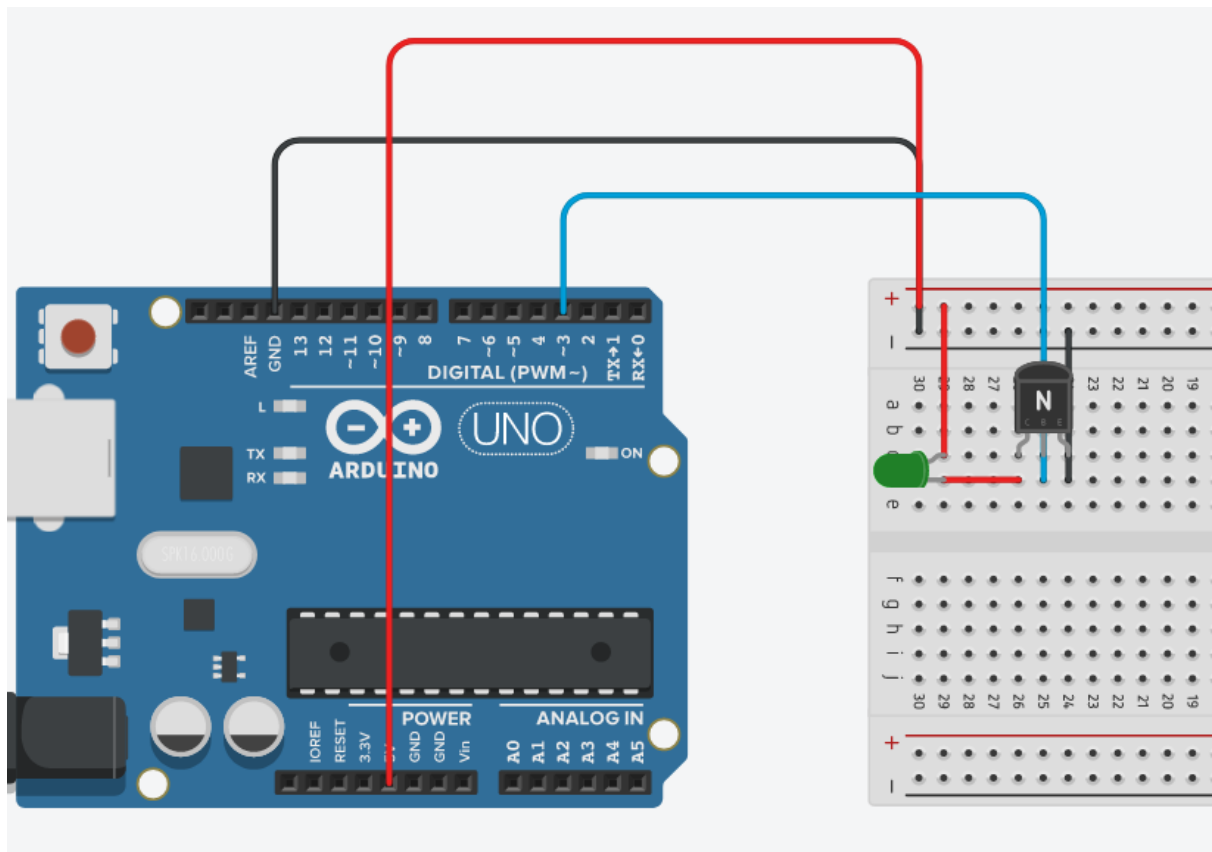
Définition de la solution technique :

Le système de détection de drones consiste en un émetteur équipé d'un laser pour envoyer une trame de données et un récepteur unique qui décode la trame de données reçue pour identifier le drone qui a tiré. La trame de données envoyée par le drone aérien commence par les bits (110) et est suivie par les bits (10), tandis que la trame de données envoyée par les drones terrestres commence par les bits (110) et est suivie par les bits (01).

Le récepteur décodera la trame de données reçue et identifiera le type de drone à cibler en fonction des bits suivants les bits (110). Le système sera conçu pour une utilisation en extérieur et devra être capable de fonctionner dans des conditions météorologiques favorables.

Emetteur

Prototype



Prototype de l'émetteur laser qui va permettre d'émettre une trame différente en fonction du type de drone (aérien ou au sol) à notre cible.

Composants

Arduino ESP 32

Émettre la trame en fonction du type de drone

Drone aérien : 11001

Drone au sol : 11010

Laser

Longueur d'onde : environ 520nm (vert)

Transistor

Référence : NPN Bipolaire

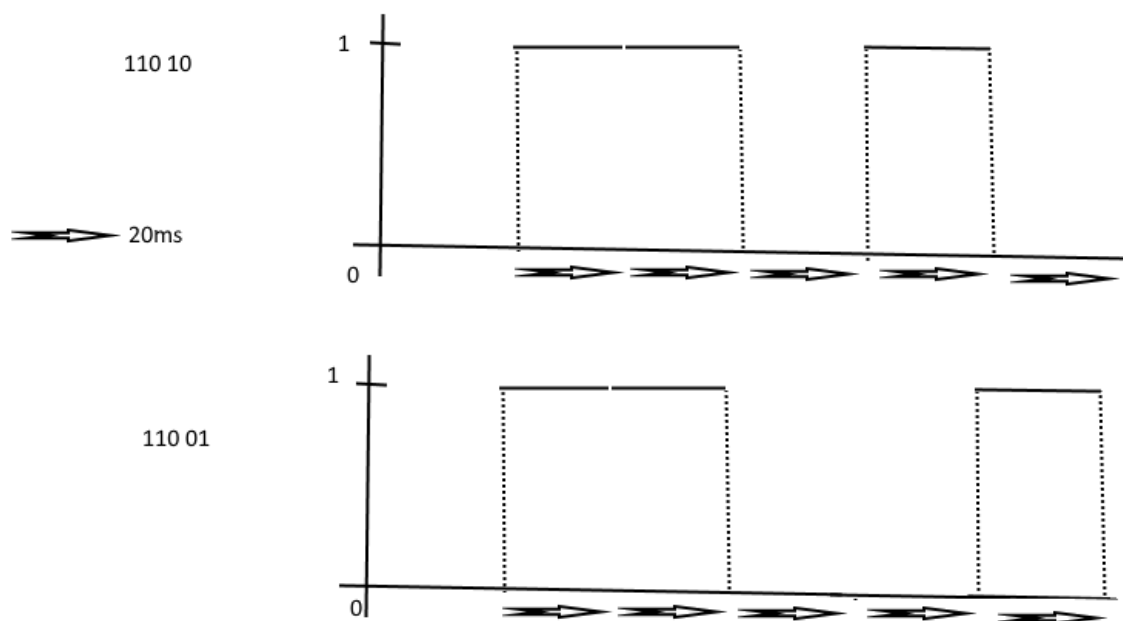
Le transistor va permettre de faire passer une tension de 5V et il nous permet de faire la variation de la tension via un PIN sur l'arduino et ainsi émettre la donnée (1 ou 0).

La lentille va nous permettre de concentrer le rayon du signal laser sur une longue distance dans un point plus précis de notre récepteur, notamment sur la photoresistance.

Programmation

La programmation des 2 arduinos va nous permettre de configurer l'émetteur pour envoyer 2 trames différentes dans un espace de temps égal à : ...

Et configurer le récepteur pour réceptionner la trame émise en 100ms (20ms par bit).



Arduino émetteur

Code : Voir le fichier `emetteur.cpp`

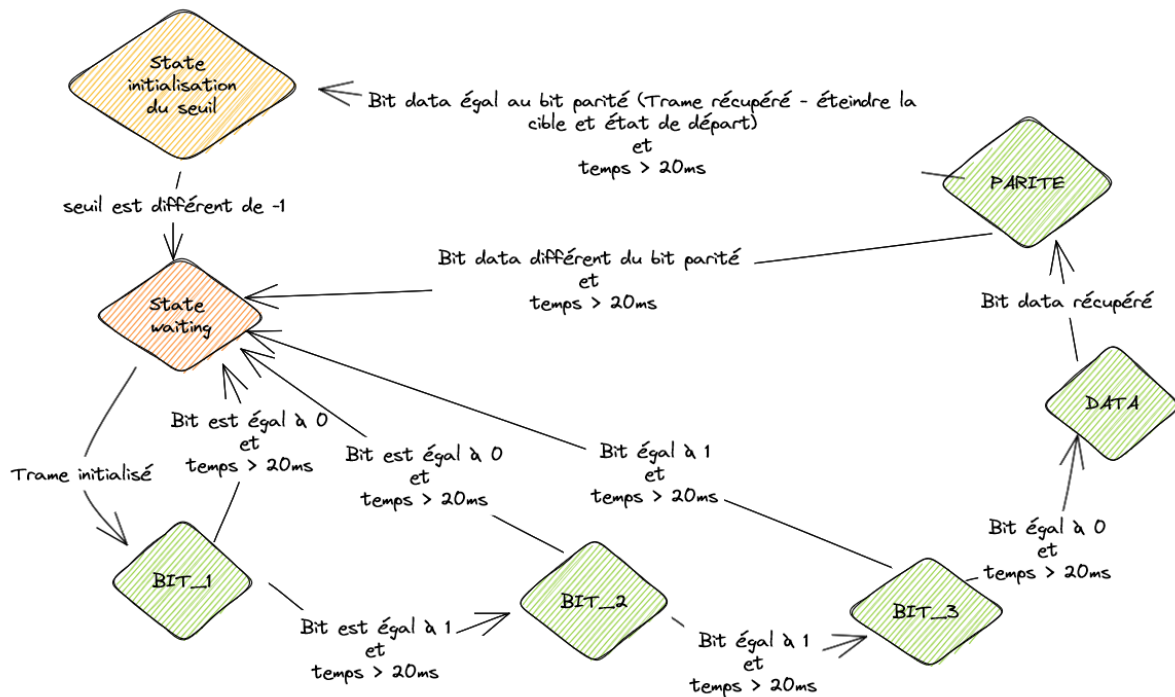
Pour émettre une valeur 1 l'émetteur va allumer le laser pendant 4ms puis s'éteindre pendant 18ms

Pour émettre une valeur 0 l'émetteur va s'éteindre pendant 19ms

Arduino récepteur

Code : Voir le fichier `recepteur.cpp`

Pour réceptionner les valeurs le récepteur va écouter pendant 20ms et déterminer si il y a eu une variation de tension perçu avec la photoresistance et ainsi déterminer si l'émetteur à envoyer une valeur 1.



La Finite State Machine (FSM) est composé de 5 états :

- State INIT_SEUIL : On détermine le seuil de la lumière ambiante
- State WAITING : On initialise la trame et on passe à l'état BIT1
- State BIT 1 : On récupère le premier bit de la trame et si il est égal à 1 on passe en BIT 2 sinon en WAITING
- State BIT 2 : On récupère le second bit de la trame et si il est égal à 1 on passe en BIT 3 sinon en WAITING
- State BIT 3 : On récupère le troisième bit de la trame et si il est égal à 0 on passe en DATA sinon en WAITING
- State DATA : On récupère le bit de data et on passe en PARITE
- State PARITE : On récupère le bit de parité et on vérifie qu'il soit différent au bit data, si c'est le cas on enregistre la trame et on l'affiche sinon on retourne en WAITING

Résultat final

Ci-dessous, nous pouvons retrouver la FSM ainsi que les résultats obtenus dans la console (la trame) lorsque le laser balaye le récepteur.

Nous remarquons que l'on récupère bien la trame que l'on souhaite transmettre avec le laser.

(Ici une distance de 35m environ à l'ombre en utilisant filtre et lentille de fresnel)

D'autres tests sont nécessaires pour espérer toucher une distance de 100m.

[illegible]