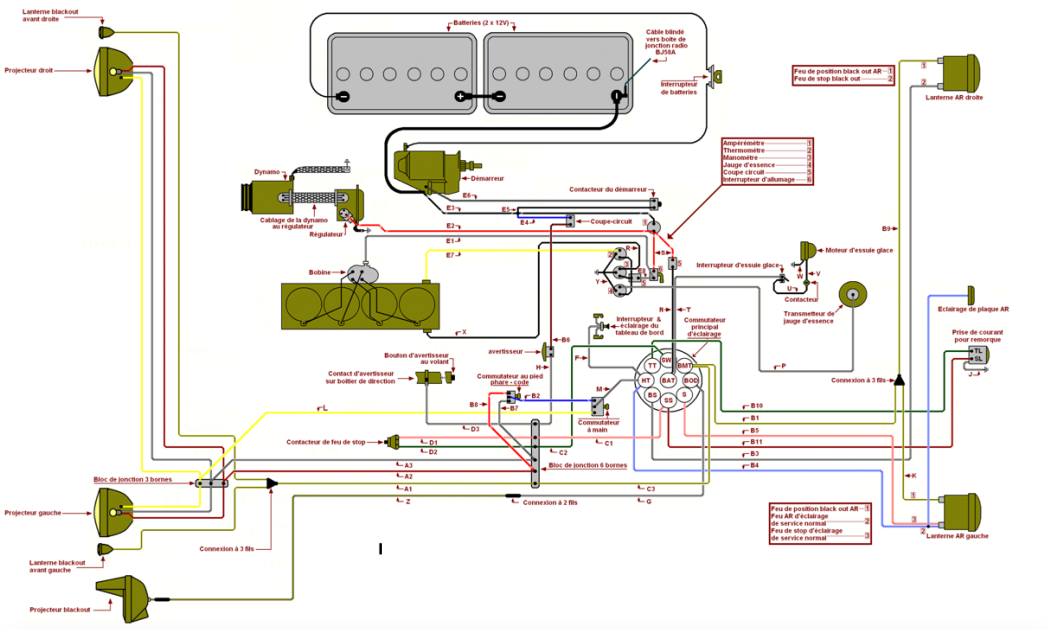
Projet 4A MOC : Électronique embarqué

Conception de l'électronique d'une voiture



Objectif :

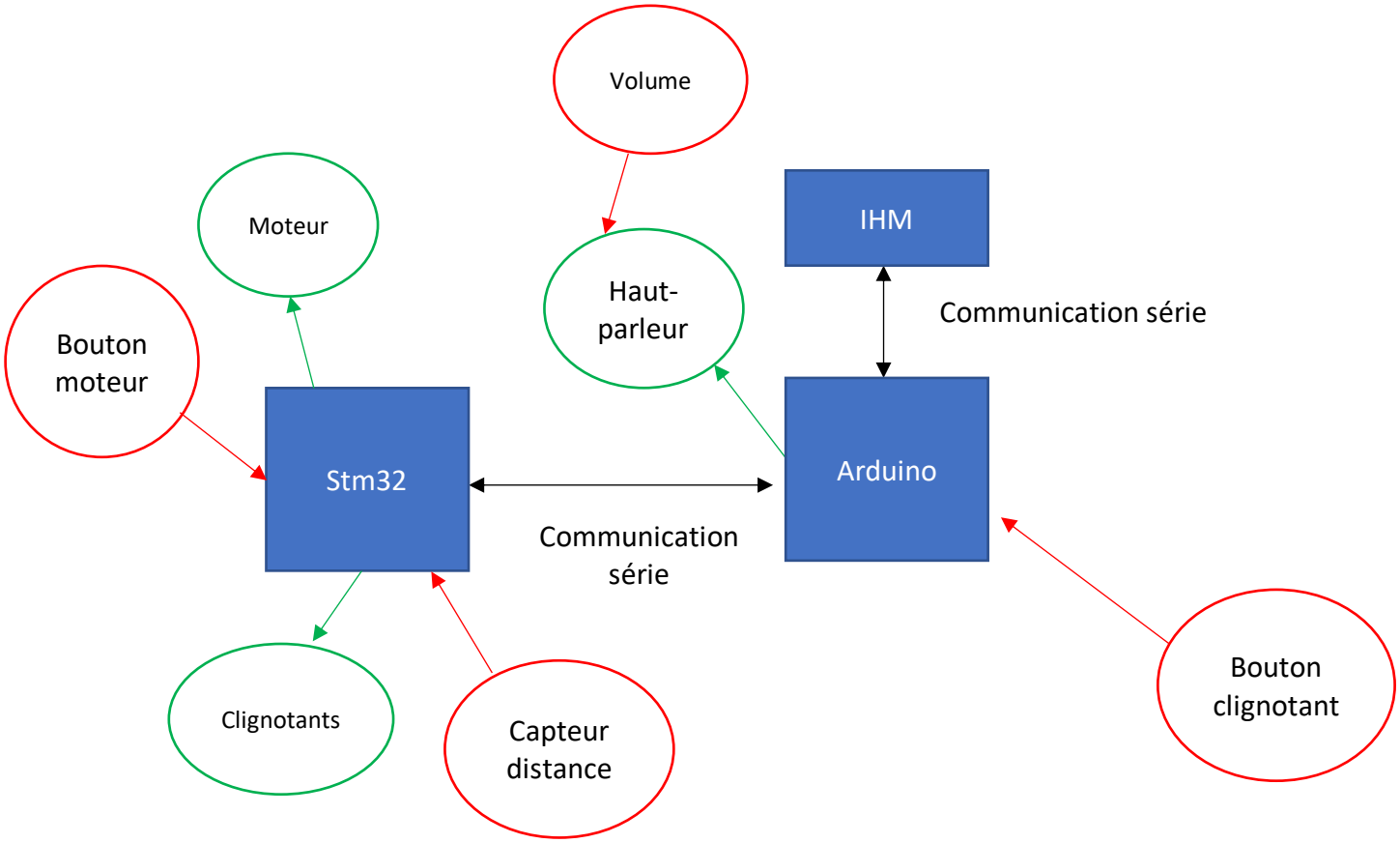
L'objectif de ce projet est de mettre en place un système d'électronique embarqué complet, c'est-à-dire un système qui contient un ou plusieurs circuits analogiques interfaçant des capteurs et/ou actionneurs, un ou plusieurs microcontrôleurs pilotant ces circuits analogiques et une *IHM* (interface homme machine) qui sert de lien entre l'utilisateur et le système. Une *IHM* peut être un site *Web*, un logiciel ou une application mobile par exemple.

Dans le cadre de ce projet vous devez réaliser une partie de l'électronique d'une voiture incluant la réception des commandes utilisateur, l'affichage d'information sur un tableau de bord, le pilotage des capteurs/actionneurs…

1- Cahier des charges :

Depuis un tableau de bord composé de trois boutons, plusieurs *LEDs* et une *IHM*, l'utilisateur doit être capable d'allumer et éteindre un moteur, de piloter des clignotants, de connaitre la position des obstacles environnant via un signal sonore et de régler le volume sonore des sons joués. Toutes les informations sur l'état du véhicule seront affichées sur l'*IHM* (distance de l'obstacle, état du moteur etc..).

Voici le schéma complet du système :



- Actionneurs
- Capteurs

2- Réalisation :

Afin de faciliter la réalisation du système complet il est conseillé de le diviser en fonctionnalités. Une fonctionnalité correspond à une action que peut faire l'utilisateur sur le système. L'utilisateur peut par exemple allumer un clignotant ou encore être tenu au courant de la distance d'un obstacle via un signal sonore. Il est préférable de développer les fonctionnalités une à une puis de les intégrer ensemble par la suite. Chaque fonctionnalité utilise un ou plusieurs modules. Un module est une brique élémentaire du système qui peut fonctionner indépendamment du reste. Pour chaque module on utilise du matériel précis. Un *TP* est généralement associé à un module.
On assemble donc les modules entre eux pour réaliser les fonctionnalités. Par exemple la fonctionnalité « *pilotage du moteur via bouton* » dépendra des modules « *moteur CC* » et « *Boutons de contrôle du moteur* ». En effet pour allumer les moteurs on appuiera sur un bouton et on activera le moteur. Pour réaliser une fonctionnalité il vaut mieux maitriser au préalable les modules nécessaires associés.

Pour chaque module, il n'est pas précisé dans le matériel requis l'ensemble des résistances et autres composants nécessaires au bon fonctionnement du système.

L'ensemble de l'électronique sera réalisé proprement sur une plaque d'essai.

Les modules et fonctionnalités peuvent être réalisées dans l'ordre de votre choix.

Vous devez donc réaliser le plus de fonctionnalités possibles pour ce projet.

3- Matériel :

- Une carte *Stm32*
- Une carte *Arduino*
- *Un moteur à courant continu* :
  https://docs.rs-online.com/88b2/A700000008739998.pd
- *Un relai* :
  https://wiki.seeedstudio.com/Grove-Relay/
- *Un haut-parleur* :
  https://docs.rs-online.com/b943/0900766b8157fda9.pdf
- *Un amplificateur opérationnel* :

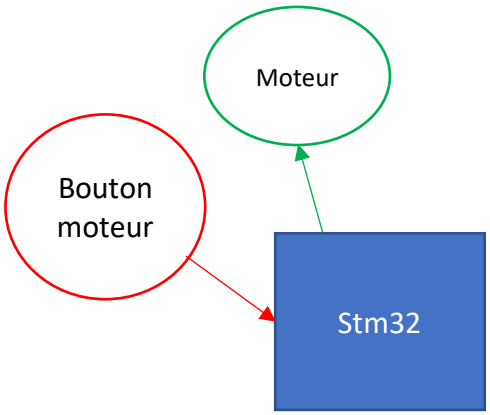
  https://pdf1.alldatasheet.com/datasheet-pdf/view/17981/PHILIPS/LM358AN.html
- *Un potentiomètre* :
  https://docs.rs-online.com/1e37/0900766b80fe1658.pdf
- *Un capteur ultrason* :
  https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
- *Deux LEDs*
- *Trois boutons poussoirs*
- *Une pile 9V*
- *Une plaque d'essaie*
- *Des fils*
- *Des résistances*

4- Fonctionnalités

L'ensemble des fonctionnalités ci-dessous doivent fonctionner en parallèles. Les modules associés doivent donc être implémentés le plus possible en mode interruption.

Fonctionnalité : Pilotage du moteur via bouton

Le but de cette fonctionnalité est de pouvoir allumer ou éteindre le moteur implémenté sur la carte *Stm32* depuis un bouton implémenté en interruption sur cette même carte *Stm32*.

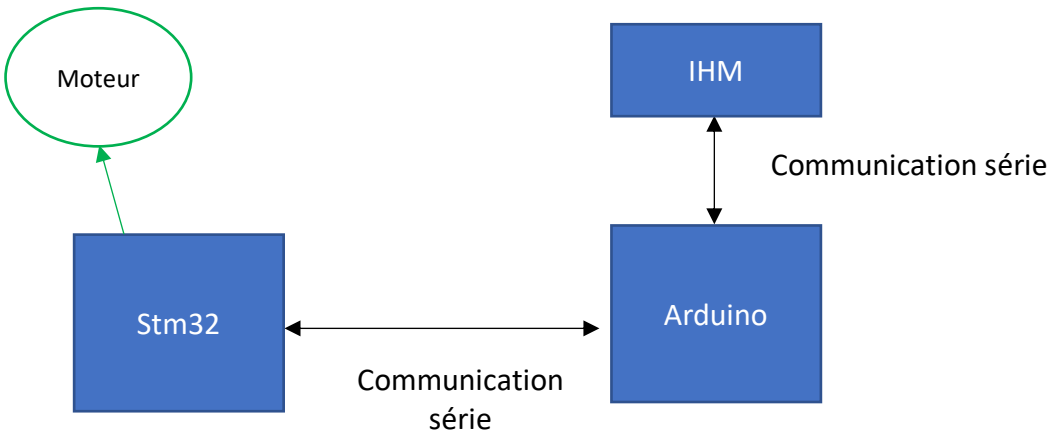


Modules associés :

- *Moteur CC*
- *Boutons de contrôle du moteur*

Fonctionnalité : Pilotage du moteur via *IHM*

Le but de cette fonctionnalité est de pouvoir allumer ou éteindre le moteur implémenté sur la carte *Stm32* depuis l'*IHM* (en ligne de commande ou depuis un bouton d'une interface graphique). On transmettra donc une commande via les ports série selon un protocole.

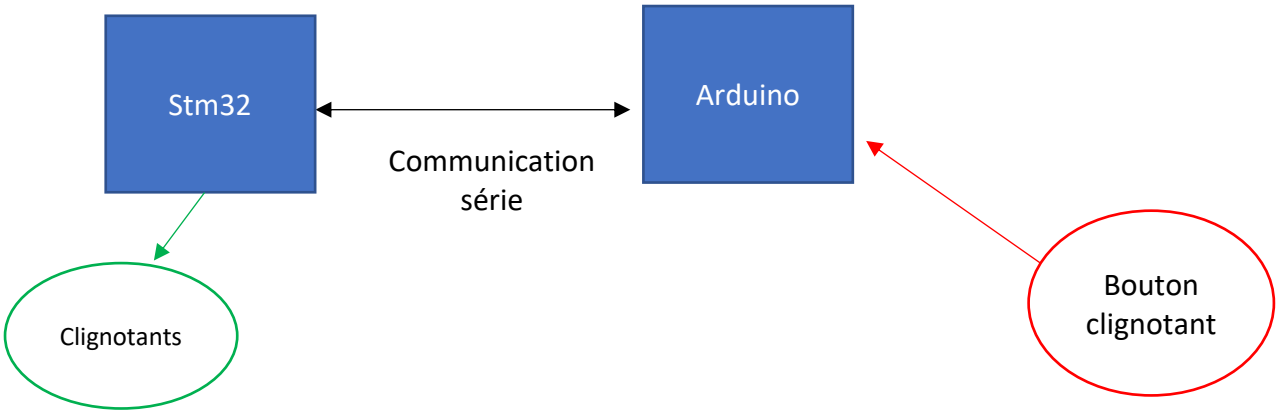


Modules associés :

- *Moteur CC*
- *Communication série Stm32/Arduino*
- *Communication série IHM/Arduino*
- *Protocole de communication*

Fonctionnalité : Pilotage clignotant via boutons

Le but de cette fonctionnalité est de pouvoir allumer ou éteindre un clignotant (gauche ou droite) en fonction de l'état des boutons branchés sur la carte *Arduino*. Il y aura donc un bouton pour le clignotant gauche et un bouton pour le clignotant droite. Les deux clignotants ne peuvent pas être allumés en même temps. On transmettra la commande via le port série connectant la *Stm32* et l'*Arduino*.

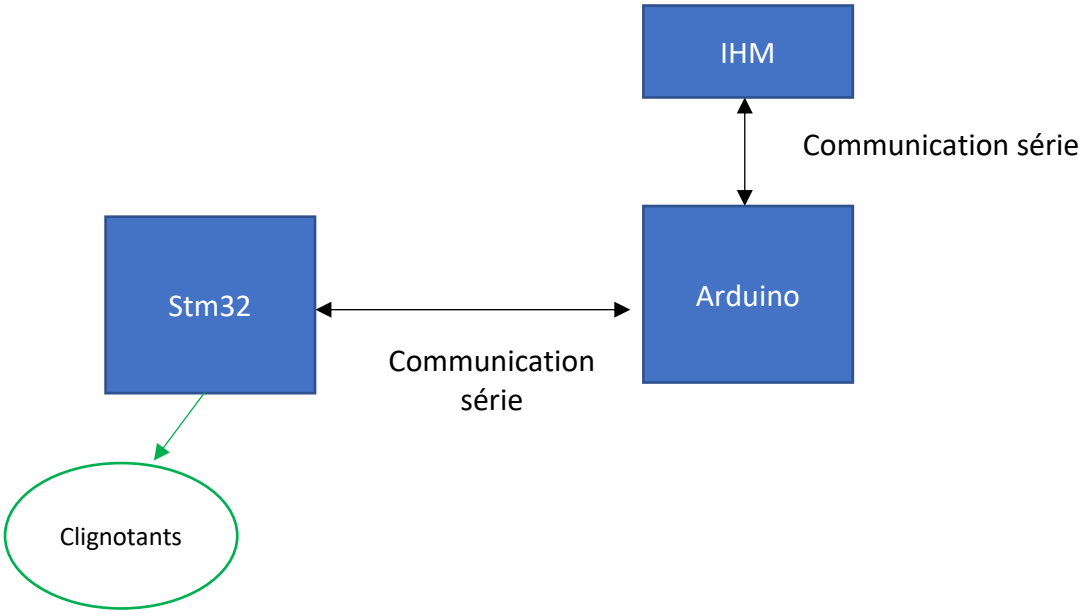


Modules associés :

- *Boutons de contrôle des clignotants*
- *Communication série Stm32/Arduino*
- *Protocole de communication*
- *Clignotants*

Fonctionnalité : Pilotage clignotant via IHM

Le but de cette fonctionnalité est de pouvoir allumer ou éteindre un clignotant (gauche ou droite) via l'*IHM*. Pour cela l'interface graphique de l'*IHM* pourra avoir *2* boutons.

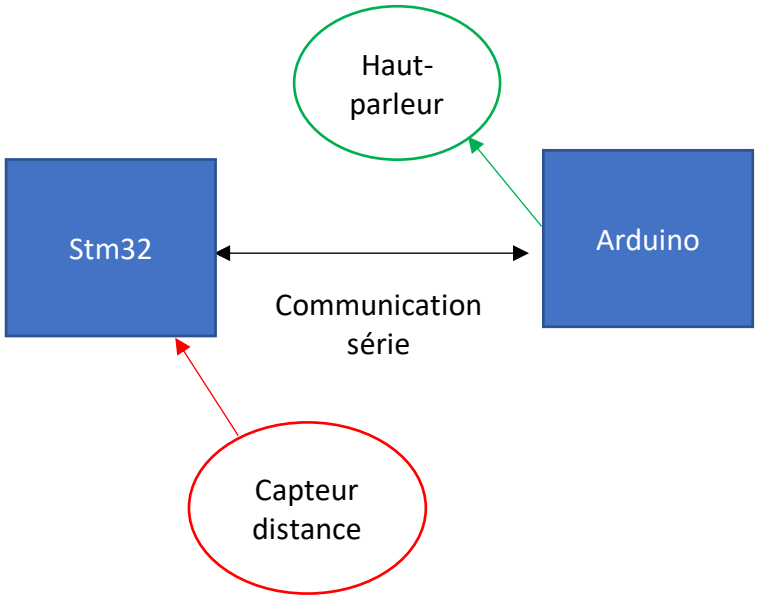


Modules associés :

- Communication série Stm32/Arduino
- Communication série IHM/Arduino
- Protocole de communication
- Clignotants

Fonctionnalité : Détection d'obstacle sonore

Le but de cette fonctionnalité est de récupérer l'information de distance relevé par le capteur ultrason sur la carte *Stm32*, de l'envoyer sur la carte Arduino afin de jouer un son de plus en plus rapide sur le haut-parleur plus la distance est faible.

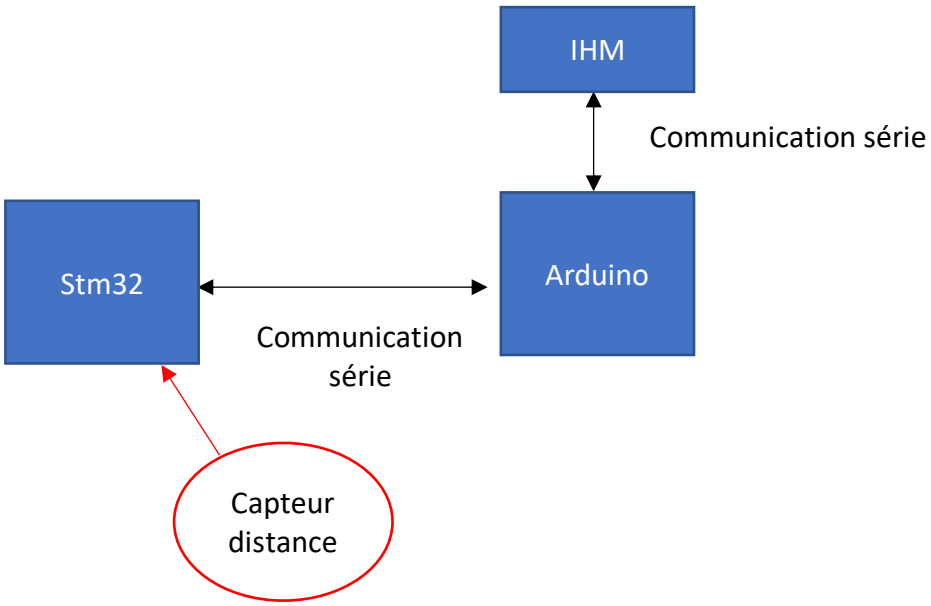


Modules associés :

- Communication série *Stm32/Arduino*
- Protocole de communication
- Capteur distance
- Haut-parleur

Fonctionnalité : Détection d'obstacle via IHM

Le but de cette fonctionnalité est d'afficher la distance du capteur à ultrason sur l'*IHM*.

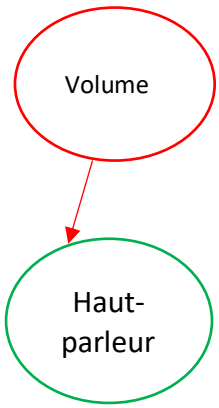


Modules associés :

- *Communication série Stm32/Arduino*
- *Communication série IHM/Arduino*
- *Protocole de communication*
- *Capteur distance*

Fonctionnalité : Gestion du volume

Le but de cette fonctionnalité est de pouvoir régler le volume sonore du haut-parleur.



Modules associés :

- Gestion du volume sonore

5- Modules

Chaque module ci-dessous est associé à un *TP*. Vous devez reprendre vos travaux pour mener à bien chaque module. Utilisez ces modules pour réaliser des fonctionnalités.

Module : Moteur

Matériel :

- Un moteur *CC*
- Un relai
- Une pile *9V*

Objectif :

Allumer ou d'éteindre le moteur depuis la carte *Stm32*. On utilisera un relai piloté par la carte *Stm32* pour ouvrir ou fermer le circuit de puissance alimenté en *9V*.

TP associé :

TP4 Moteur CC

Module : Clignotants

Matériel :

- Deux *LEDs*
- Trois boutons

Objectif :

Faire clignoter deux *LEDs* depuis la *Stm32* de la même manière que les clignotants d'une voiture. Les deux clignotants ne peuvent pas être activés en même temps, si j'active le clignotant gauche celui de droite doit être éteint.

TP associé :

Partie *3* et *4* du *TP3 Découverte du Stm32*.

Module : Capteur distance

Matériel :

- Un capteur ultrason

Objectif :

Le but de ce module est de récupérer sur la carte *Stm32*, la distance d'un objet devant le capteur. Ce module devra être réalisé en interruption pour ne pas gêner les fonctionnements des autres modules.

TP associé :

TP 7 capteur à ultrason.

Module : Haut-parleur

Matériel :

- Un haut-parleur

Objectif :

Jouer un son sur le haut-parleur piloté par une carte *Arduino*.

TP associé :

Parties *1* à *4* du *TP 5*.

Module : Gestion du volume sonore

Matériel :

- Un AOP
- Un potentiomètre

Objectif :

Régler le volume sonore du son joué sur le haut-parleur à l'aide d'un montage *AOP* amplificateur et d'un potentiomètre.

TP associé :

Partie *5* du *TP 5*.

Module : Boutons de contrôle des clignotants

Matériel :

- Deux boutons poussoir

Objectif :

Mettre en places *2* boutons en interruption afin d'être notifié sur la carte *Arduino* en cas d'appuie utilisateur.

Il faut pour cela utiliser les boutons en mode interruption sur la carte *Arduino*.

Tutoriel :

https://www.lambot.info/interruptions-multiples-sur-arduino/

Module : Boutons de contrôle du moteur

Matériel :

- Buton natif du *Stm32*

Objectif :

Utilisez le bouton natif du *Stm32* en interruption afin d'être notifié dans le code *Stm32* en cas d'appuie utilisateur.

Tutoriel :

http://wiki.labaixbidouille.com/index.php/5_-_Utiliser_un_%22bouton_poussoir%22_avec_un_%22GPIO_in%22

Module : Communication série Stm32/Arduino

Le but de ce module est d'établir une communication série entre la carte *Stm32* et la carte *Arduino*. Vous devrez être capable de transmettre des données via ce canal. Les données envoyées par la carte *Stm32* devront pouvoir être lues par la carte *Arduino* et inversement. Pour cela vous utiliserez la librairie software Serial qui permettra d'implémenter une deuxième connexion série (la première est pour la Stm32)

Tutoriel :

https://docs.arduino.cc/learn/built-in-libraries/software-serial

TP associé :

TP 6 communication série

Module : Communication série Arduino/IHM

Objectif :

Établir une communication série entre la carte *Arduino* et l'*IHM*.. Les données envoyées par la carte Arduino devront pouvoir être lues par l'IHM et inversement.

TP associé :

TP 6 communication série

Module : Protocole Communication

Objectif :

Définir un protocole de communication pour pouvoir transmettre plusieurs commandes sur un même canal série. Par exemple, sur le même port série de la carte *Arduino* vers la carte *Stm32*, on veut pouvoir allumer les clignotants ou les moteurs. On va donc créer une trame suivant un protocole.

Pour allumer le moteur, la trame (c'est enfaite une chaine de caractère, un string) envoyée sur le port série pourra être la suivante :

M1 : *M* pour moteur et *1* pour allumer

Pour l'éteindre la trame pourra être :

M0 : *M* pour moteur et *0* pour éteindre

Pour allumer le clignotant gauche on pourra envoyer la trame :

CG1 : *C* pour clignotant, *G* pour gauche et *1* pour allumer

Pour éteindre le clignotant droite on pourra envoyer la trame :

CD0 : *C* pour clignotant, *G* pour gauche et *1* pour éteindre

Du côté du *Stm32*, il faudra analyser chaque trame reçue sur le port série pour en déduire une action correspondante.

Exemple algorithmique :

(les fonctions sont des exemples et ne correspondent peut-être pas aux fonctions du *Stm32*)

trameReçu = *serial.read*(). (*trameReçu* est un tableau de *char* ou un *String*, par exemple ['M','1'])

```
if trameRecu[0] == M :   //dans ce cas, le premier caractère de la trame indique que l'on veut piloter le moteur
        if trameRecu[1] == '1' :
                Allumer moteur()
        If trameRecu[1] == '0' :
                Éteindre moteur()

if trameRecu[0] == C :   //dans ce cas, le premier caractère de la trame indique que l'on veut piloter un clignotant
        traitement Clignotant
```

TP associé :

TP 6 communication série

Module : IHM

Le but de ce module est de développer une *IHM*. Cette *IHM* pourra piloter les clignotants, le moteur et afficher la distance relevée par le capteur à ultrason. Le plus simple est d'utiliser un langage de programmation de votre choix (*python* par exemple), qui pourra lire et écrire sur le port série de votre ordinateur. Vous pourrez rajouter une interface graphique avec des bouton et un affichage dynamique de l'état de la voiture.