

Transmission de l'information:

Objectif : configurer le protocole de la liaison série .
Simuler une liaison séries pour visualiser une trame.

Présentation du matériel.

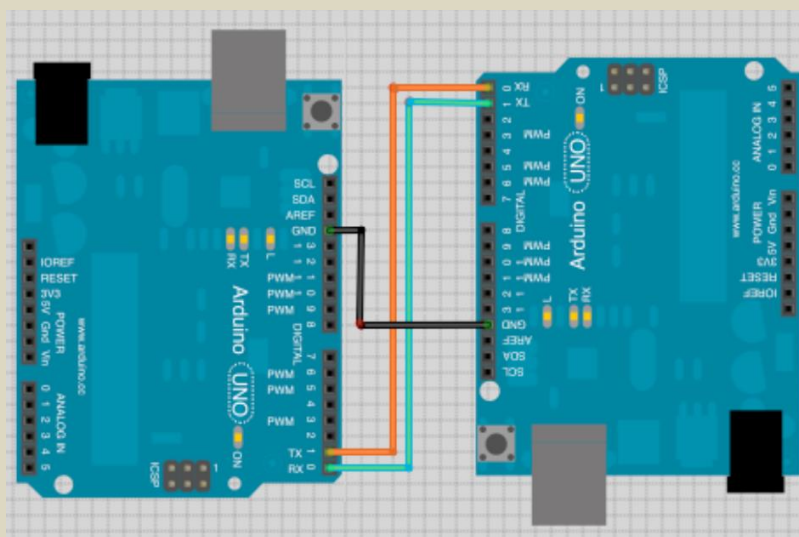


I – Généralités

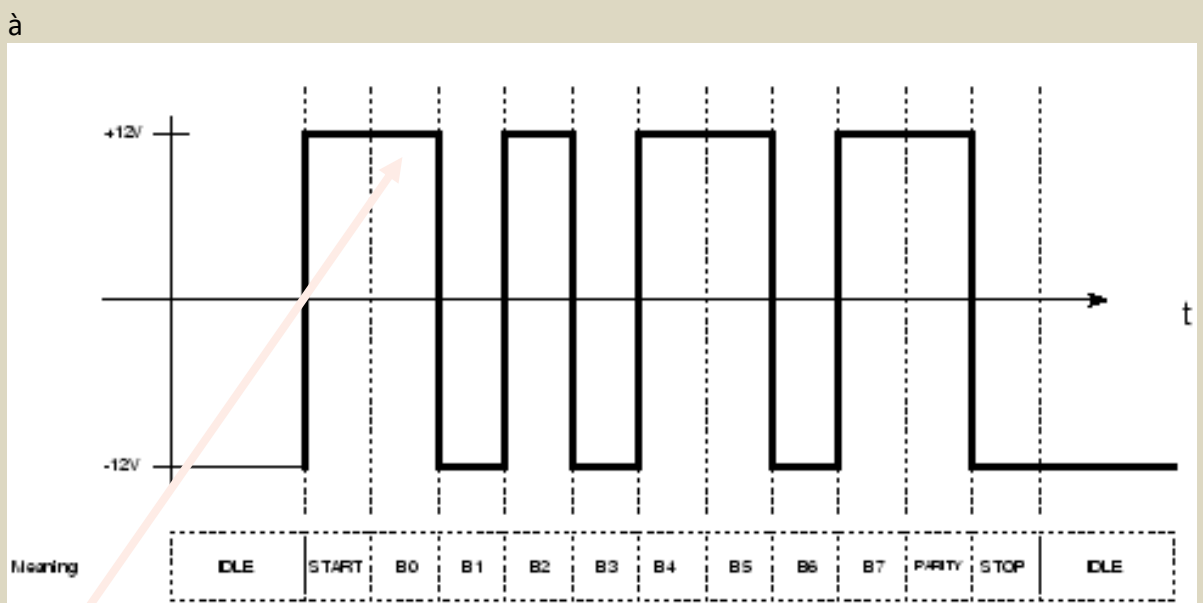
- La liaison série entre la carte Arduino et l'ordinateur est établie à travers le port USB. En fait, ce port USB n'est pas utilisé avec le protocole USB, mais avec celui de la liaison série RS232.

Ceci est donc géré par la carte Arduino et il n'y a rien à paramétrer.

- Pour relier deux cartes Arduino en liaison série, il suffit de connecter les broches Tx et Rx ensemble, de cette manière :



- ✓ Donner le rôle des broches Rx et Tx dans une liaison série ?
- l'ordinateur utilise des niveaux de -12V à +12V (de manière habituelle, mais ils sont en réalité entre -3/-24V et +3/+24V). Les niveaux "positifs" représentent un état bas (un '0' logique), alors qu'un niveau haut (le '1' logique) est représenté par les tensions négatives.
- L'arduino utilise les niveaux de tensions 0V et 5V (niveau TTL)
- La carte Arduino, plutôt que d'être branché sur un port série classique sera donc branché sur l'USB. Les niveaux seront donc toujours du 5V maximum. Ensuite, un composant intégré à Arduino se chargera de simuler une voie série et tout devient transparent pour votre ordinateur. Il vous suffit donc juste d'utiliser le câble USB et de le relier.
- ✓ Soit l'image ci-dessous, donnez le code transmit en binaire.



0	1	0	1	0	0	1	0		
---	---	---	---	---	---	---	---	--	--

Code en binaire = 01001010 = 82 = \$52

- ✓ Retrouver le code ASCII ainsi que le caractère envoyé. 52 c'est la lettre R
- ✓ Expliquer le rôle du bit de parité.
Pour détecter des erreurs de transmission



II – Notion de classe

Dans les langages de programmation actuels, les concepteurs ont imaginé la possibilité de pouvoir rassembler des fonctions ensemble lorsqu'elles s'appliquent à une même fonctionnalité.

Tout comme une fonction, une classe aura un nom et pour distinguer une classe d'une fonction, le nom d'une classe commencera par une MAJUSCULE.

Le langage Arduino ou ses bibliothèques comporte plusieurs classes et il faut comprendre ce concept :

- Toutes les fonctions qui gèrent la communication avec le port série USB sont rassemblées dans une classe appelée Serial,
- La classe LiquidCrystal pour la gestion d'un afficheur LCD,
- La classe Servo pour la gestion d'un servomoteur,
- Etc...

En pratique, pour utiliser une fonction d'une classe du langage Arduino, on utilisera le nom de la classe + un point + le nom de la fonction.

NomClasse.fonction1()

La classe Serial

Les fonctions de la classe Serial sont a nombre d'une dizaine.

Nous allons découvrir les 4 fonctions qui permettent d'écrire un programme pour afficher des messages vers le PC.

- `begin()` : fonction d'initialisation de la communication USB
- `print()` : fonction d'affichage d'un message sans saut de ligne
- `println()` : fonction d'affichage d'un message avec saut de ligne
- `read()` : permet de lire et de renvoyer le premier caractère présent dans le port série
- `available()` : renvoie le nombre d'octets présents en réception sur le port série (true si caractère présent et false sinon...)
- `flush()` : vide la file d'attente en réception du port série.

Dans le cas de la classe Serial, on fera :

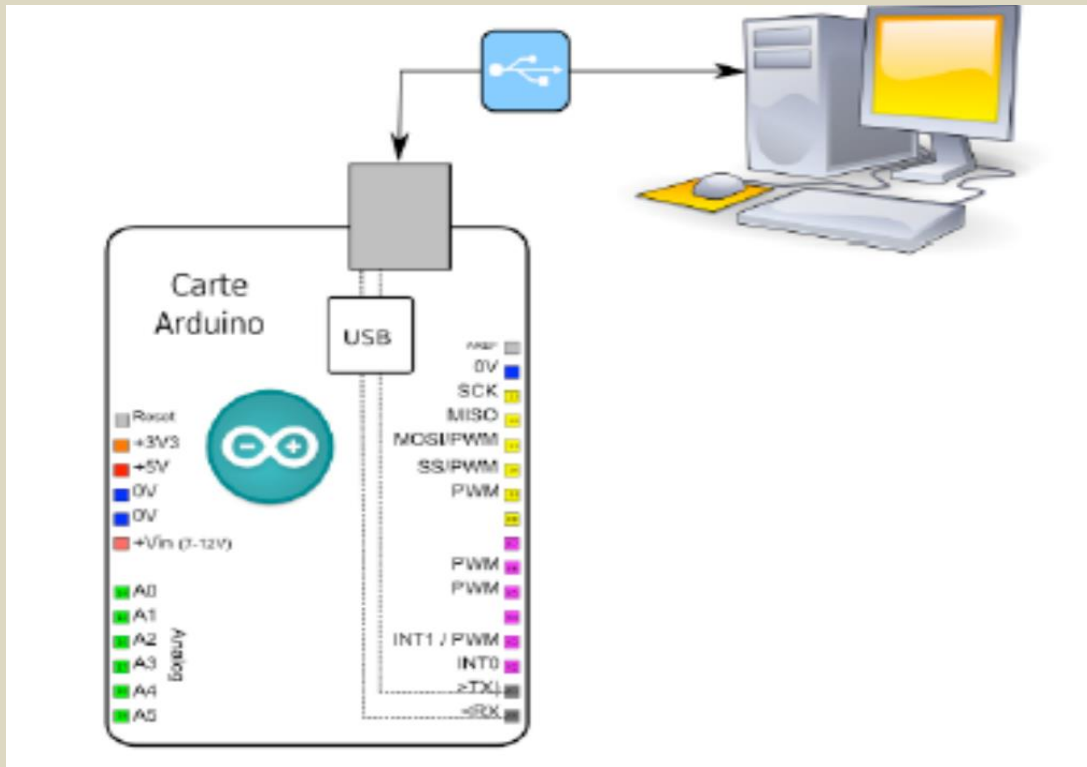
- `Serial.begin()` pour appeler la fonction `begin()`,
- `Serial.print()` pour appeler la fonction `print()`,
- `Serial.println()` pour appeler la fonction `println()`,
- `Serial.read()` pour appeler la fonction `read()`,
- `Serial.available()` pour appeler la fonction `available()`



Petit cours sur envoyer et recevoir des données sur la voie série .

<https://eskimon.fr/tuto-arduino-302-envoyer-et-recevoir-des-donn%C3%A9es-sur-la-voies%C3%A9rie>

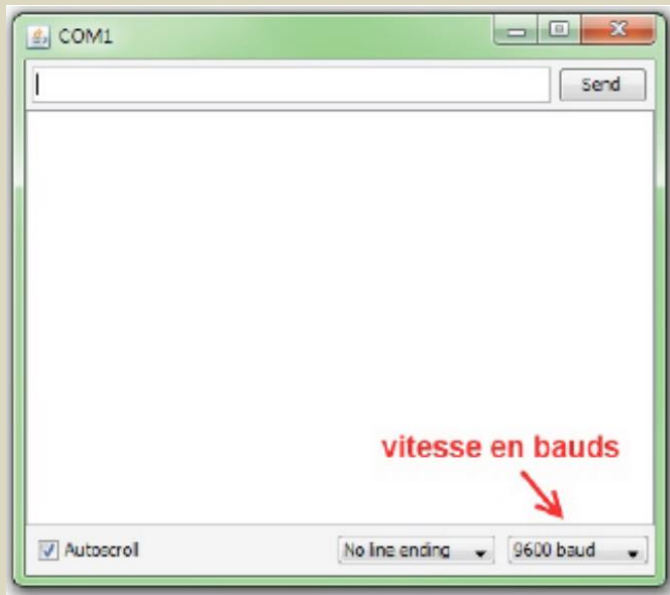
III– Envoyer et recevoir des données.



Pour pouvoir utiliser la communication de l'ordinateur, l'environnement de développement Arduino propose de base un outil pour communiquer. Pour cela, il suffit de cliquer sur le bouton dans la barre de menu pour démarrer l'outil.

Une nouvelle fenêtre s'ouvre, c'est le terminal série :





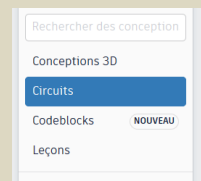
Dans cette fenêtre, vous allez pouvoir envoyer des messages sur la liaison série de votre ordinateur (qui est émulée par l'Arduino) ; recevoir les messages que votre Arduino vous envoie ; et régler des paramètres tels que la vitesse de communication avec l'Arduino et l'autoscroll qui fait défiler le texte automatiquement.

TP1 : Programme bonjour

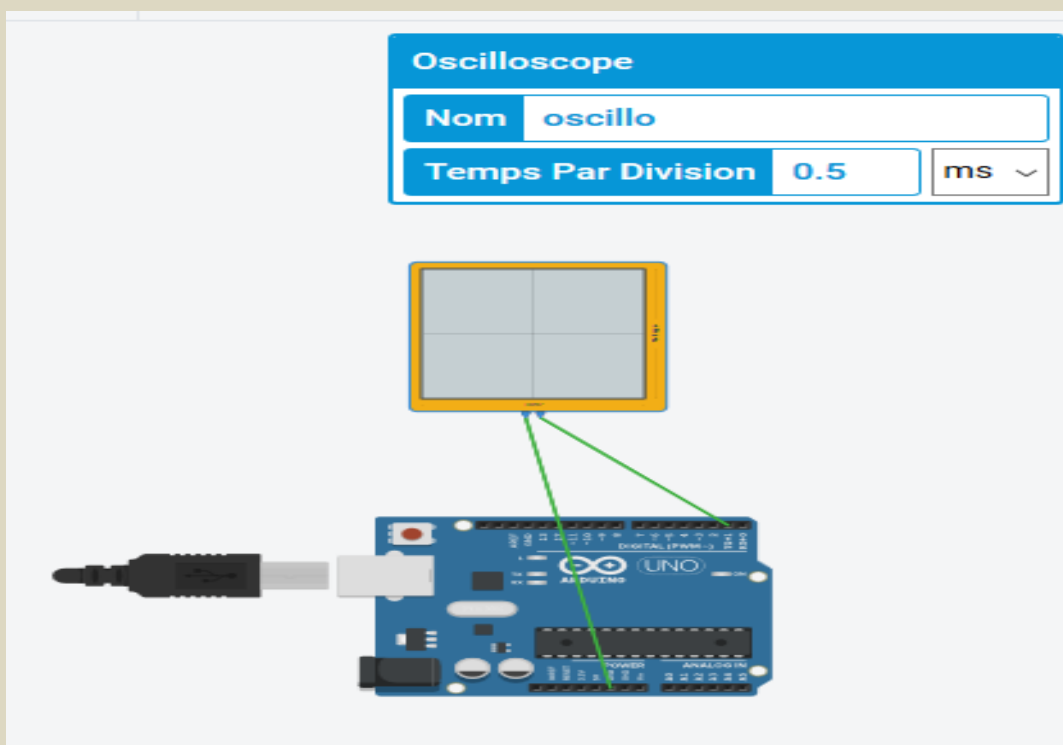
Pour toute nos simulation, nous allons utiliser un simulateur en ligne : <https://www.tinkercad.com>



Cliquez sur l'onglet circuits :



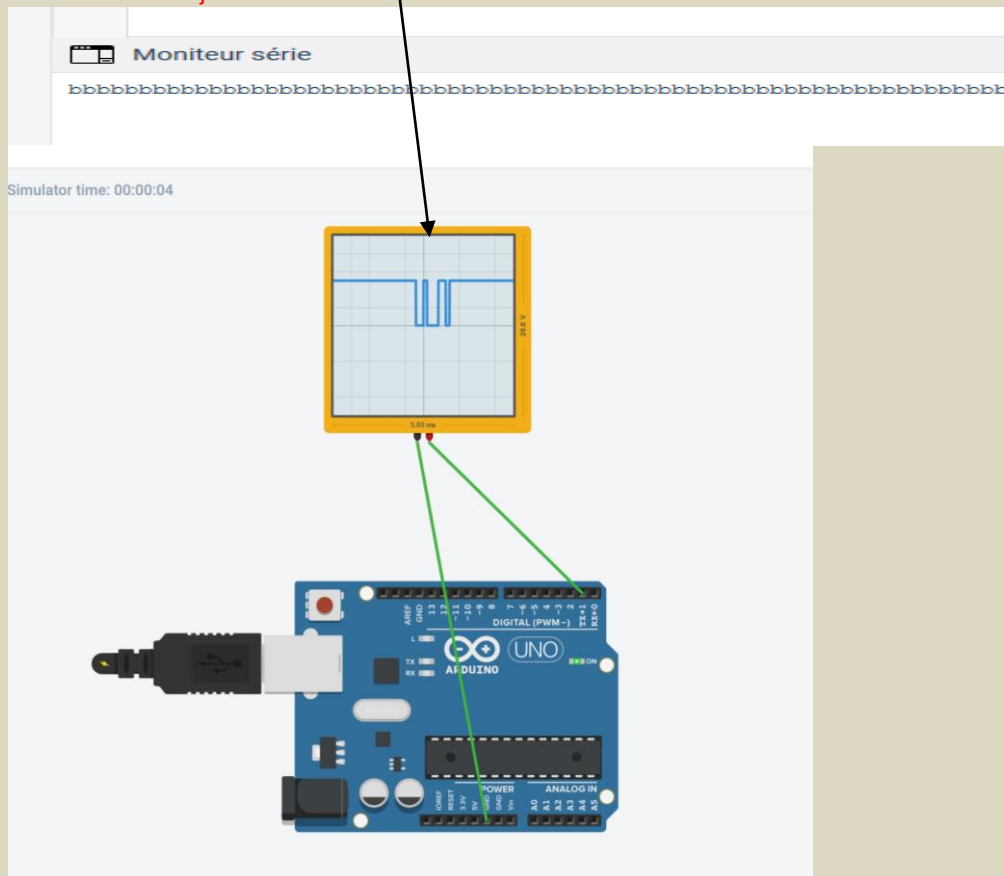
Réaliser le schéma suivant :



Ecrire un programme que vous nommerez bonjour qui va :

- ✚ Initialiser la communication à 9600 bauds,
 - ✚ Afficher le message « b » dans le terminal toutes les 2 secondes , avec retour à la ligne.
 - ✚ Avec un oscilloscope récupérer le code ascii de b. Décoder la trame série.
- ```
void setup()
{
 Serial.begin(9600);
}

void loop()
{
 Serial.print ("b"); // on théorie sérial.println ("b") mais ça rajoute l'information
 delay (10); retour chario dans le code.
}
```



- ✚ Afficher le message « bonjour » dans le terminal toutes les 2 secondes , avec retour à la ligne.

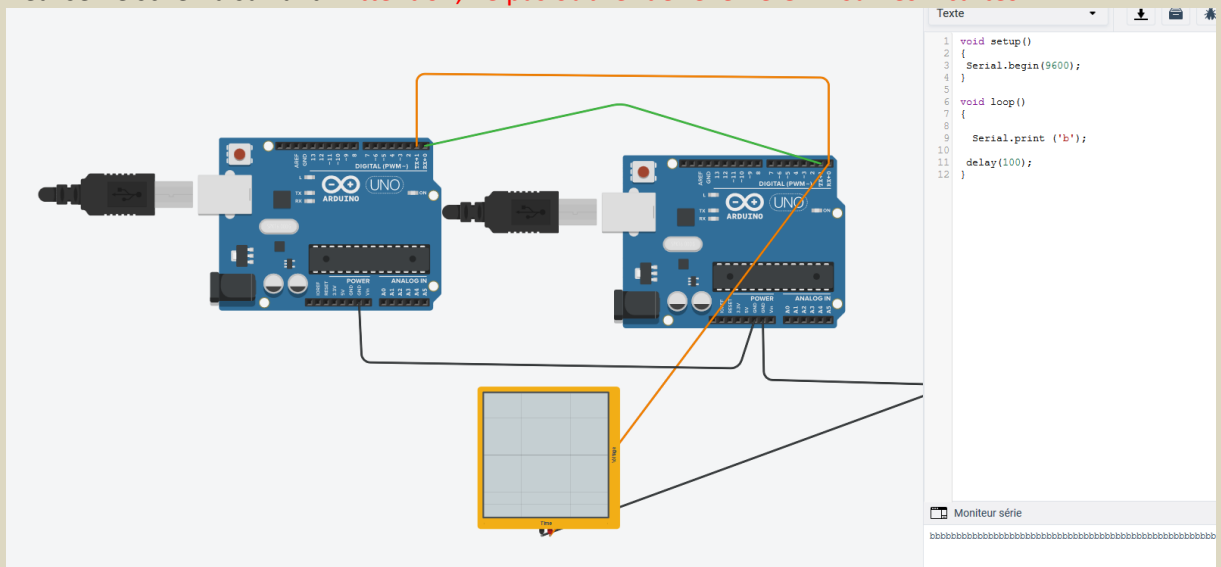
```

✚ void setup()
✚ {
✚ Serial.begin(9600);
✚ }
✚
✚ void loop()
✚ {
✚ Serial.print ("bonjour"); // on théorie sérial.println ("b") mais ça rajoute
 l'information retour chario dans le code.
✚ delay (2000);
✚ }

```

## TP2 : Programme caractère\_reçu

Réaliser le schéma suivant : **Attention, ne pas oublier de relier le GND sur les 2 cartes.**



La 1<sup>er</sup> carte Arduino garde le programme du TP1.

La 2eme carte arduino a le programme suivant.

Ecrire un programme que vous nommerez caractère\_reçu dont la structure est la suivante :

- ✚ Entête déclarative
  - On déclare une variable int pour stocker en réception (code ASCII)
  - On déclare une variable char pour stocker le caractère correspondant
- ✚ Fonction setup()
  - on initialise la communication série avec une vitesse de 9600 bauds.
- ✚ Fonction loop()



- On écoute le port série en testant l'arrivée d'un caractère
- Si un octet est reçu, on affiche successivement :
  - o Sa valeur numérique (code ASCII du caractère reçu)
  - o Puis le caractère correspondant.

Valider le fonctionnement du programme en simulant ce schéma et en ouvrant le moniteur série de la carte 2.

```
int val,T;
void setup() {
 // Begin the Serial at 9600 Baud

 Serial.begin(9600);
}

void loop() {
 delay(1000);
 int v = Serial.read();
 delay(1000);
 Serial.println(v);

 delay(1000);
}
```

