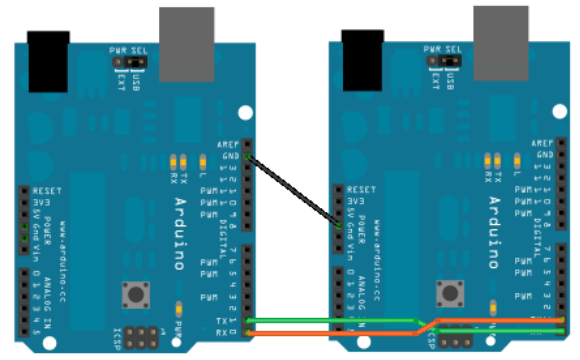
|  |  |  |
| --- | --- | --- |
| Lycée Joliot-Curie  Rennes | **TRANSMISSION DE L’INFORMATION** | **Séq. 5** |
| **Liaison série RS-232** | TP |

***Mise en situation***

Comme vous le savez, il est possible de transmettre des informations entre une carte Arduino et un PC. Cette liaison série est de type USB.

Il est également possible par exemple, de transmettre des informations entre deux cartes Arduino. Cette liaison série asynchrone est de type TTL (proche du RS-232). Elle se fait à partir du pin1 : Tx (transmissions) et du pin0 : RX (réception).

**Liaison série type TTL (proche du RS-232)**



**Liaison série type USB**

***Présentation de la norme RS-232***

La norme RS-232 est un protocole de transmission série qui permet des transmissions full duplex, half duplex ou même simplex. Les signaux transmis sont codés en ±12V.

Le "1" logique est équivalent à un niveau -12V, le "0" logique est équivalent à un niveau +12V.

Le nombre de bits de start : 1 bit à l’état « 0 »

Le nombre de bits de données : Il peut varier de 4 à 8 bits.

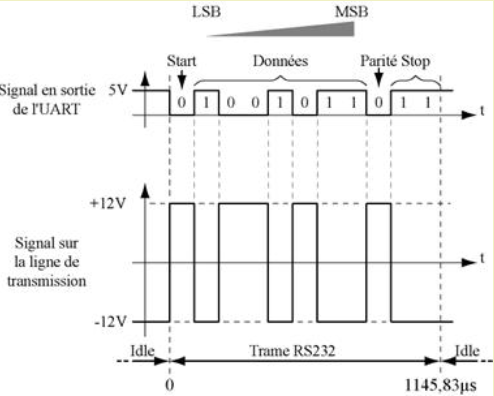
Le nombre de bits de stop : Il peut être de 1 ou 2 (à l’état « 1 » lorsque la ligne est inoccupée)

Le type de contrôle de la parité (gestion des erreurs). Il peut être choisi parmi trois possibilités :

* pas de contrôle de parité;
* parité paire (le message contient un nombre pair de '1');
* parité impaire (le message contient un nombre impair de '1').

Les bits sont en fait des niveaux de tension électrique. La norme RS-232 définit quelles tensions doivent être utilisées. Les niveaux de tension imposés par la norme sont définis dans le tableau ci-dessous :

|  |  |  |
| --- | --- | --- |
|  | Niveau logique 0 | Niveau logique 1 |
| Tension électrique minimale | + 3V | -3 V |
| Tension électrique maximale | + 25 V | -25 V |



Les niveaux de tension transmis sont ensuite traités par un UART (Universal Asynchronous Receiver Transmitter) qui va les transformer en niveaux de tension adaptés au microprocesseur (voir la représentation ci-contre).

Les codes transmis sont définis selon la **table ASCII (voir document ressource).**

***Travail demandé***

**1 /** Brancher un oscilloscope sur les broches GND et TX de la carte Arduino UNO.

**2 /** Téléverser le code ci-dessous dans la carte. Vous trouverez les détails sur l’instruction « Serial.begin » au lien suivant : <https://www.arduino.cc/en/Serial/Begin>

*void setup()*

*{*

*Serial.begin(1200,SERIAL\_7E1); // Initialise le port série à 1200 bauds, 7 bits de données, parité paire (1 bit), 1 bit de stop*

*}*

*void loop()*

*{*

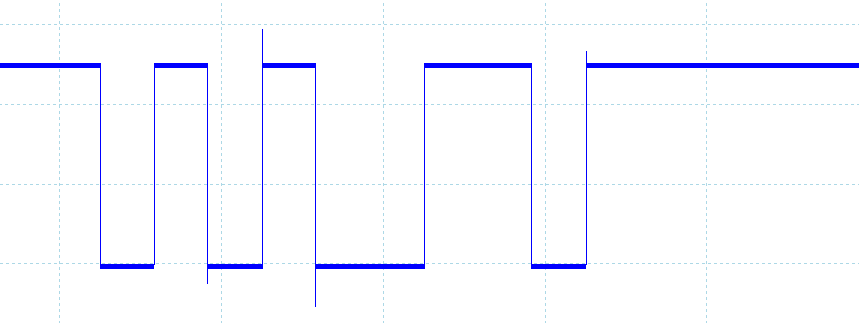
*Serial.print("e"); // Transmet la trame correspondant à la lettre « e »*

*delay(50); // Temps de repos avant répétition de la trame*

*}*

**3 /** Calculer le temps de transmission d’un bit puis de la trame complète correspondante à la lettre « e ».

**4 /** Relever l’ensemble de la trame à l’oscilloscope et identifier les informations bits à bits. Représenter les temps calculés à la question Q3.



**5 /** Tracer la représentation des signaux ±12V sur la ligne de transmission au format RS-232 (ne représenter que les informations de la trame).

Tension (V)

+ 12 V

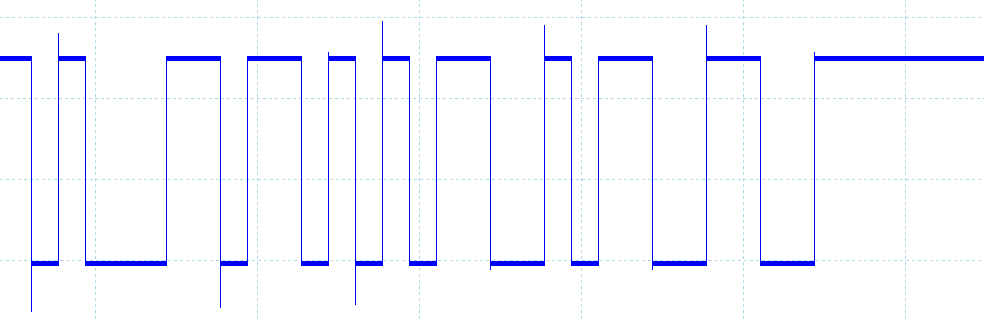
Temps (ms)

- 12 V

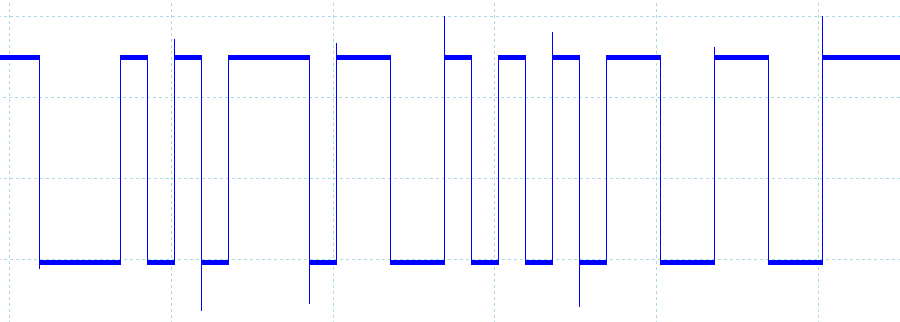
**6 /** Vérifier à l’oscilloscope les temps de transmissions calculés précédemment.

**7 /** Modifier dans le programme, le débit de la transmission à 9600 bauds. Que constatez-vous ? Déterminer le temps de transmission de la trame.

**8 /** Refaire les questions Q3, Q4 et Q5 pour la transmission du nombre « 153 ».

****

**9 /** Déterminer le message transmis par la trame suivante. Tracer en dessous, la représentation des signaux ±12V sur la ligne de transmission au format RS-232.



Tension (V)

+ 12 V

Temps (ms)

- 12 V

**10 /** Vérifier l’ensemble des vos résultats en utilisant le simulateur RS-232.

**DOCUMENT RESSOURCE**

**Table ASCII**

