Source : http://bts2m.free.fr/TP_Arduino/01_UART.html

## La liaison série asynchrone

**Un UART, pour Universal Asynchronous Receiver Transmitter, est un émetteur-récepteur asynchrone universel. En voici le principe pour un Arduino :**

- Transmission de données d'un équipement 1 (un microcontrôleur Arduino) à un équipement 2 (PC, GPS, émetteur Bluetooth, microcontrôleur ....) .
- Données à transmettre existent sous forme parallèle (octet) et sont transmises sous forme série (LSB en premier)
- Données reçues sous forme série (LSB en premier ...:-) puis reconditionnées sous forme d'octet.
- Pour permettre une liaison plus rapide les données sont stockées dans un buffer (mémoire tampon) d'une capacité de 64 octets.
- Entre 2 équipements les fils sont croisés : Tx1 relié à Rx2 et Tx2 relié à Rx1 (voir figure)
- Les niveaux de tension sont de type TTL soit 0 V pour le niveau bas et +5V pour le niveau haut.
- Asynchrone car aucune horloge (bit clock) n'est transmise entre l'emetteur et le récepteur. Le recepteur ignore quand il va recevoir une donnée.
- Afin de faciliter l'interopérabilité entre périphériques des vitesses de transmission sont normalisées par multiples et sous-multiples de 9600 baud,
- l'unité baud correspondant à une vitesse de transmission de un bit par seconde.

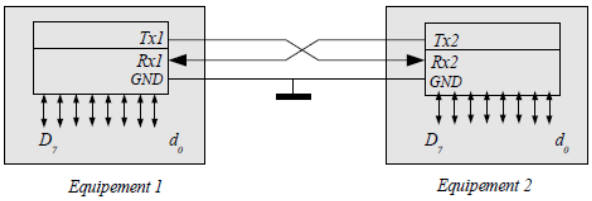
**Avantages** : standardisée, universelle, pas cher
3 fils suffisent (émission Tx, réception Rx, masse GND) et souvent l'alimentation + 5 V

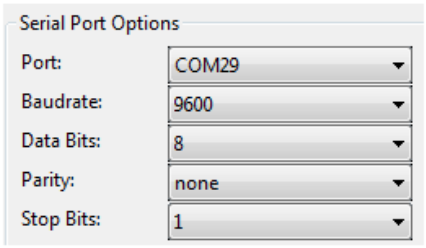
**Inconvénients** : Assez lent. (maximum pour un Arduino : 115200 bauds)
Une liaison UART ne permet que de relier 2 équipements.

**Constitution d'une trame UART Arduino :**

- un bit de *start* toujours à 0 : servant à la synchronisation du récepteur
- les données : pour un code Ascii étendu 8 bits (un octet) (cf code ascii sur wikipedia)
- et un bit de *stop* toujours à 1

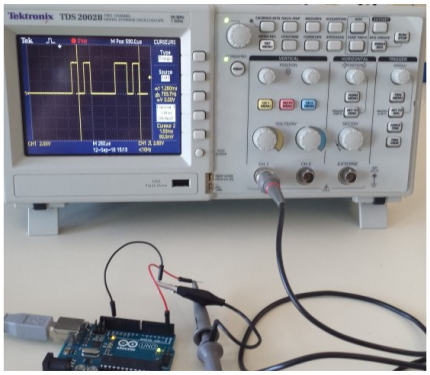
Le niveau logique de repos est le 1.

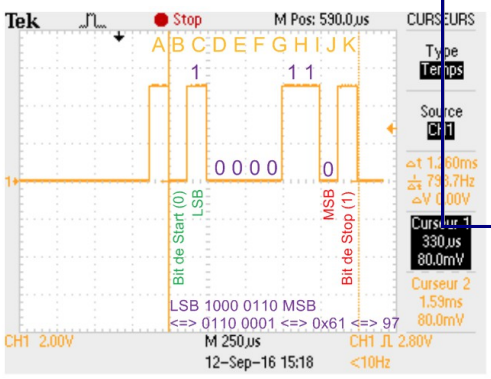


## Test de la liaison UART

Nous avons voulu savoir comment fonctionnait physiquement la liaison UART.
Pour cela nous avons transmis le caractère "a" et observé à l'oscilloscope la patte Tx d'un Arduino Uno.






| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | NUL | DLE | space | 0 | @ | P | ` | p |
| 1 | SOH | DC1 XON | ! | 1 | A | Q | a | q |
| 2 | STX | DC2 | " | 2 | B | R | b | r |
| 3 | ETX | DC3 XOFF | # | 3 | C | S | c | s |
| 4 | EOT | DC4 | $ | 4 | D | T | d | t |
| 5 | ENQ | NAK | % | 5 | E | U | e | u |
| 6 | ACK | SYN | & | 6 | F | V | f | v |
| 7 | BEL | ETB | ' | 7 | G | W | g | w |
| 8 | BS | CAN | ( | 8 | H | X | h | x |
| 9 | HT | EM | ) | 9 | I | Y | i | y |
| A | LF | SUB | * | : | J | Z | j | z |
| B | VT | ESC | + | ; | K | [ | k | { |
| C | FF | FS | , | < | L | \ | l | \| |
| D | CR | GS | - | = | M | ] | m | } |
| E | SO | RS | . | > | N | ^ | n | ~ |
| F | SI | US | / | ? | O | _ | o | del |

Le code ASCII du caractère a est 97, soit 0x 61 en hexadécimal, soit 0b 0110 0001 en binaire.

$$0b\ 0110\ 0001 = 0x\ 61 = 6*16^1 + 1*16^0 = 96 + 1 = 97$$

*Commentaire de l'oscillogramme obtenu :*

- A : Bit de Stop de la trame précédente
- B : Bit de start de la nouvelle trame a = (MSB)0110 0001(LSB),
  soit à l'envers (LSB)1000 0110(MSB)
- les 4 bits de poids faible : C (LSB=1) D (0) E (0) F (0)
- les 4 bits de poids fort : G (0) H (1) I (1) J (MSB=0)
- K : Bit de Stop de la trame étudiée

1-a) Pour réaliser ce test, copier le programme ci-dessous. Puis téléversez-le sur un Arduino uno.
Visualiser à l'oscilloscope la transmission du caractère x
Sauvegarder la trame, l'imprimer et la mettre dans votre compte rendu.

1-b) Sur la trame, repérer le niveau logique de chaque bit (ils seront notés sur le chronogramme) puis déduire son code ASCII et vérifier ce résultat dans la table ASCII ci-dessus.

Le baudrate de 8000 bit/s n'est pas standard.
Mais la durée de transmission d'un bit est alors de 1/8000 = 125 µs ce qui correspond à une demi-division de l'oscilloscope (calibre 250 µs/div).

```
// Visualisation d'une trame série à l'oscilloscope

void setup() {
 Serial.begin(8000);
// Oscilloscope entre Tx->1 et GND calibre : 250 µs:div
}
void loop() {
  //Nous envoyons le caractère "a" code ASCII : 97 soit 0x61 soit 0b01100001
  // suivi d'un caractère nul pour bien isoler le "a" à l'oscilloscope
  Serial.write(97); // idendique à Serial.print("a"); ou à Serial.write(0x61);
  // Trame de  10 bits : Start : 0/LSB:1/0/0/0/0/1/1/MSB:0/Stop:1
  // durée de transmission d'un bit : 1/8000 = 0,125 ms = 125 µs soit 1/2 div
  // durée de la trame : 10*0.125ms = 1,25 ms   pour un octet soit 800 octets à
la seconde
  Serial.write(0);
  }
```

# ANNEXE 1

**La liaison série asynchrone**

Caractéristiques de la liaison UART Arduino

Un UART, pour Universal Asynchronous Receiver Transmitter, est un émetteur-récepteur asynchrone universel. En voici le principe pour un Arduino :

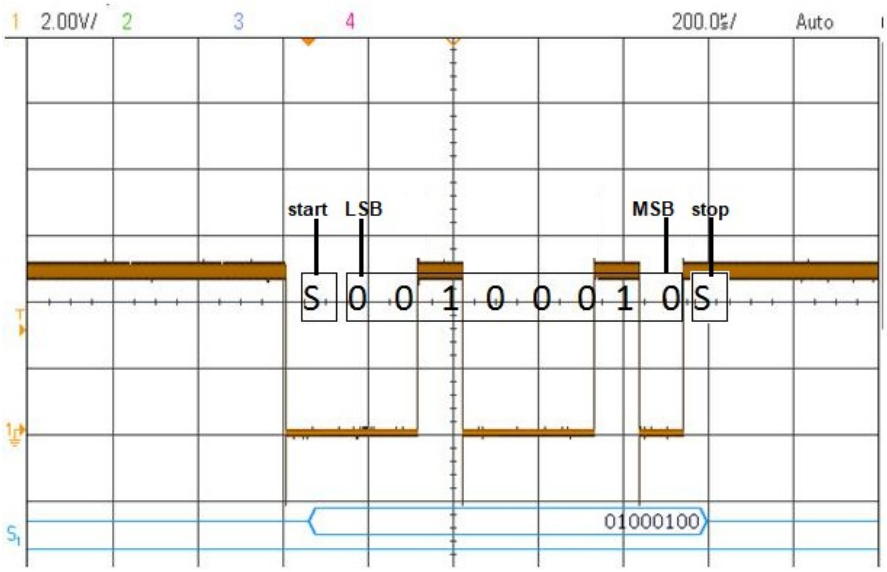
- Les données à transmettre et sont transmises sous forme série (LSB en premier).
- Les données reçues sous forme série (LSB en premier ...) puis reconditionnées sous forme d'octet.
- Les niveaux de tension sont de type TTL.
- Asynchrone car aucune horloge (bit clock) n'est transmise entre l'emetteur et le récepteur. Le recepteur ignore quand il va recevoir une donnée.

Constitution d'une trame UART Arduino

- un bit de *start* toujours à 0 : servant à la synchronisation du récepteur
- les données : pour un code Ascii étendu 8 bits (un octet)
- et un bit de *stop* toujours à 1

Le niveau logique de repos est le 1.

Exemple de décodage d'une trame UART Arduino (1 caractère Ascii transmis)




On visualise bien les 10 bits :

- bit de start à 0
- 8 bits de données LSB first
- bit de sop à 1

# ANNEXE 2

**Table des codes « ASCII étendu »**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | nul | | | 0 | @ | P | ` | p | Ç | É | á | ▒ | └ | ╨ | α | ≡ |
| 1 | | | ! | 1 | A | Q | a | q | ü | æ | í | ▓ | ┴ | ╤ | ß | ± |
| 2 | | | " | 2 | B | R | b | r | é | Æ | ó | ▓ | ┬ | ╥ | Γ | ⩾ |
| 3 | | | # | 3 | C | S | c | s | â | ô | ú | │ | ├ | ╙ | π | ⩽ |
| 4 | | | $ | 4 | D | T | d | t | ä | ö | ñ | ┤ | ─ | ╘ | Σ | ⌠ |
| 5 | | | % | 5 | E | U | e | u | à | ò | Ñ | ╡ | ┼ | ╒ | σ | ⌡ |
| 6 | | | & | 6 | F | V | f | v | â | û | ª | ╢ | ╞ | ╓ | µ | ÷ |
| 7 | bel | | ' | 7 | G | W | g | w | ç | ù | º | ╖ | ╟ | ╫ | τ | ≈ |
| 8 | bs | | ( | 8 | H | X | h | x | ê | ÿ | ¿ | ╕ | ╚ | ╪ | Φ | ° |
| 9 | tab | | ) | 9 | I | Y | i | y | ë | Ö | ⌐ | ╣ | ╔ | ┘ | Θ | ∘ |
| A | lf | | * | : | J | Z | j | z | è | Ü | ¬ | ║ | ╩ | ┌ | Ω | · |
| B | vt | esc | + | ; | K | [ | k | { | ï | ¢ | ½ | ╗ | ╦ | █ | δ | √ |
| C | ff | | , | < | L | \ | l | \| | î | £ | ¼ | ╝ | ╠ | ▄ | ∞ | ⁿ |
| D | cr | | - | = | M | ] | m | } | ì | ¥ | ¡ | ╜ | ═ | ▌ | Ø | ² |
| E | | | . | > | N | ^ | n | ~ | Ä | ₧ | « | ╛ | ╬ | ▐ | ε | ■ |
| F | | | / | ? | O | _ | o | | Å | ƒ | » | ┐ | ╧ | ▀ | ∩ | |

Exemples :

@ est codé par $40.
τ est codé par $E7.
> est codé par $3E.