**Trasmissione seriale su Arduino**

*Scopo di questa attività è analizzare la struttura di un segnale seriale*

**Fase 1:** analisi del segnale con oscilloscopio (in coppia)

*Analizzare il segnale prodotto da Arduino mediante un print*

1. Sviluppare un semplice programma su Arduino che scriva su seriale (a velocità 300 bps) un singolo carattere (sempre lo stesso) ogni mezzo secondo; verificarne il funzionamento con la finestra del monitor seriale
2. Analizzare il contenuto della pagina <https://www.vincenzov.net/tutorial/rs232/seriale.htm>
3. Analizzare con l’oscilloscopio il segnale presente sul piedino ***tx***

**Fase 2:** ricezione del segnale seriale da programma Arduino (in coppia)

*Sviluppare una coppia di programmi, uno che occupi dell’invio e l’altro della ricezione*

1. **Invio**
   1. il programma di invio in realtà si occupa di ricevere un dato da PC via seriale e reinviarlo sulla seriale stessa, alla quale sarà però connesso un secondo Arduino
   2. per inviare un dato da PC ad Arduino si può usare una qualsiasi delle applicazioni di lettura/scrittura su seriale, ad esempio *Putty* o lo stesso monitor seriale all’interno dell’IDE di Arduino. (vedi a titolo di esempio [Serial.read()](https://www.arduino.cc/reference/en/language/functions/communication/serial/read/))
   3. il dato letto e reinviato dovrà essere un singolo carattere non seguito dal alcun *newline* o *carriage return* (selezionare l’opzione opportuna sul monitor seriale)
   4. la velocità consigliata di trasmissione è 1200 bps
   5. connettere (oltre a *GND)* il piedino ***tx*** dell’Arduino dove gira il programma di invio con un input digitale del secondo Arduino dove girerà il programma di ricezione
2. **Ricezione**
   1. Il programmadi ricezione dovrà monitorare in continuazione l’input digitale al quale è connesso il *tx* dell’Arduino di invio
   2. La linea è normalmente nello stato alto
   3. Il *bit di start* è un bit a 0, quindi appena viene monitorato tale bit significa che a partire dal quel momento stanno iniziando ad arrivare gli 8 bit del dato preceduti appunto dal *bit di start*
   4. A partire da tale momento il programma inizierà a campionare la linea per leggere il valore degli 1+8 bit in arrivo (il primo sempre a zero)
   5. Conviene effettuare il campionamento al centro di ciascun bit per essere certi di non trovarsi su di un fronte
   6. Il calcolo delle temporizzazioni deve essere effettuato in modo preciso: usare [delayMicroseconds](https://www.arduino.cc/reference/en/language/functions/time/delaymicroseconds/)
   7. Memorizzare il valore dei bit ricevuti all’interno di un array e procedere alla stampa una volta che tutto il byte è stato ricevuto: prestare attenzione al fatto che i bit vengono inviati a partire dal bit più basso (destra)