2.1 Serial

DSM

Unidad II

Febrero 2021

UTM

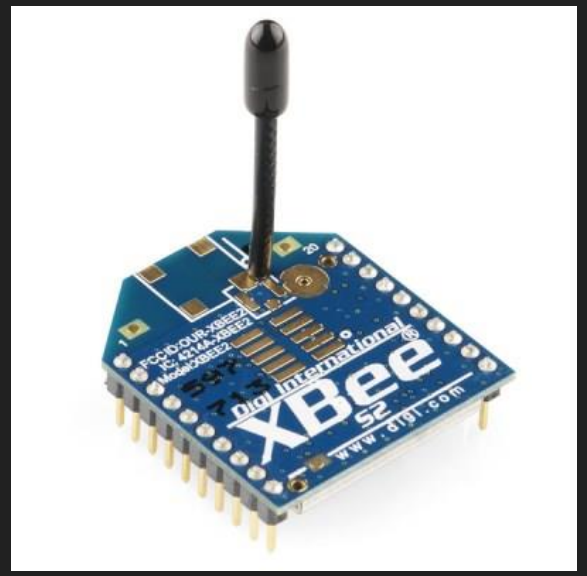
Agenda

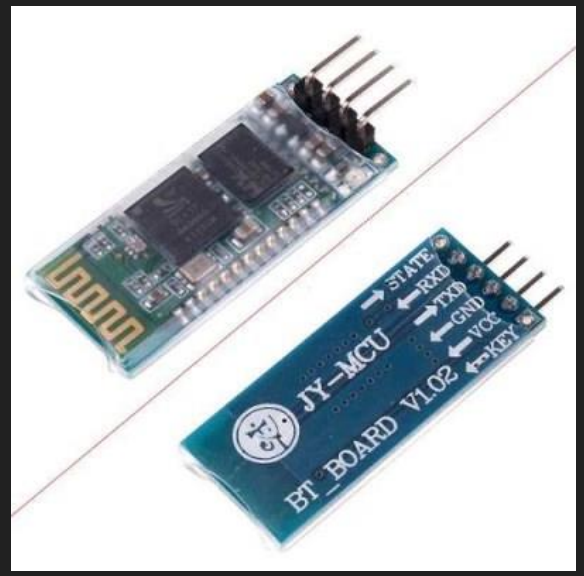
- Introducción
- Comunicación serie o serial
- Conceptos
- Puerto serie en arduino
- Reference serial programming
- SoftwareSerial library
- Conclusión

Introducción

- La comunicación serie o serial es importante porque gran parte de los protocolos utilizados actualmente son serie y además muchos dispositivos de comunicación inalámbrica usan la comunicación serie para “hablar” con Arduino como los módulos bluetooth y los módulos Xbee, módulos Sigfox, etc..

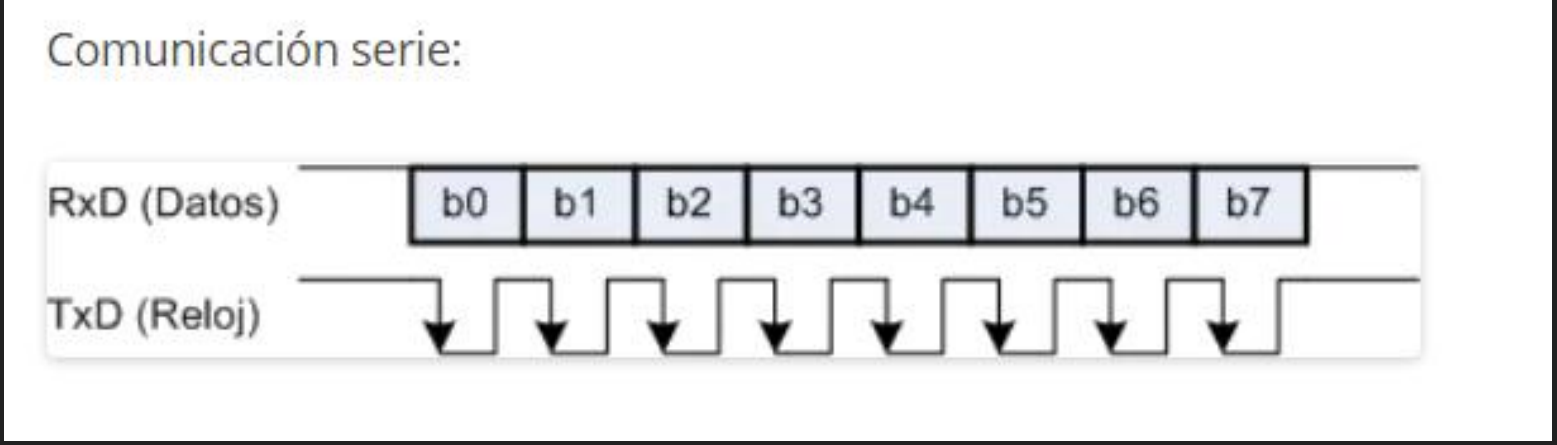
Breakouts / módulos comm



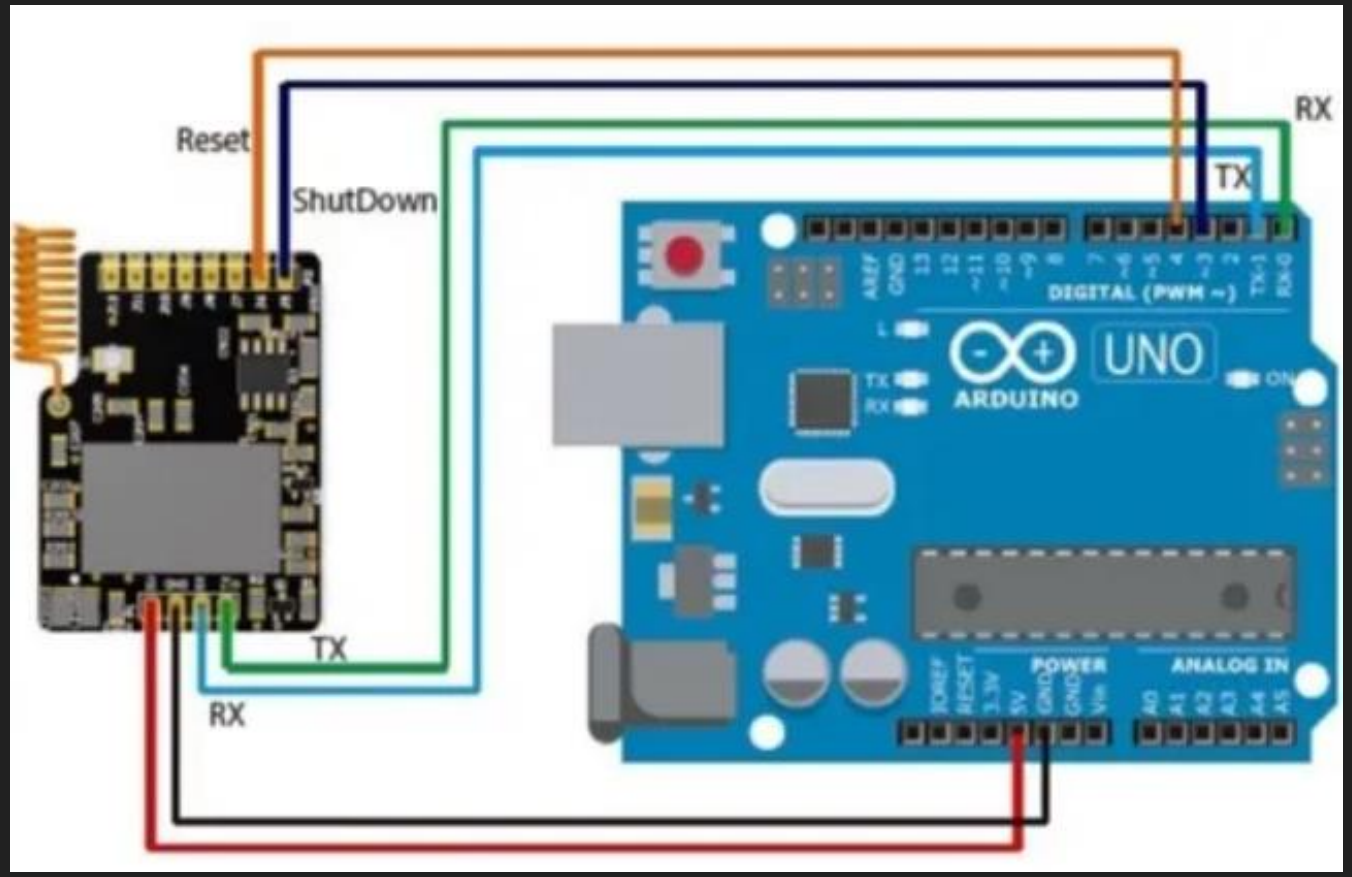


Comunicación serie

- También la comunicación serie es la que se usa para comunicar el Arduino con la computadora a través del cable USB.

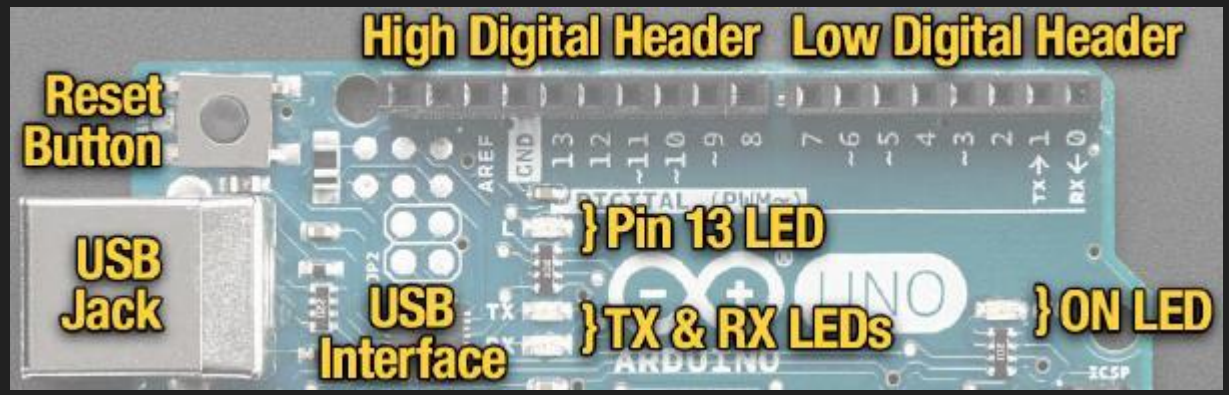


Comunicación serie..

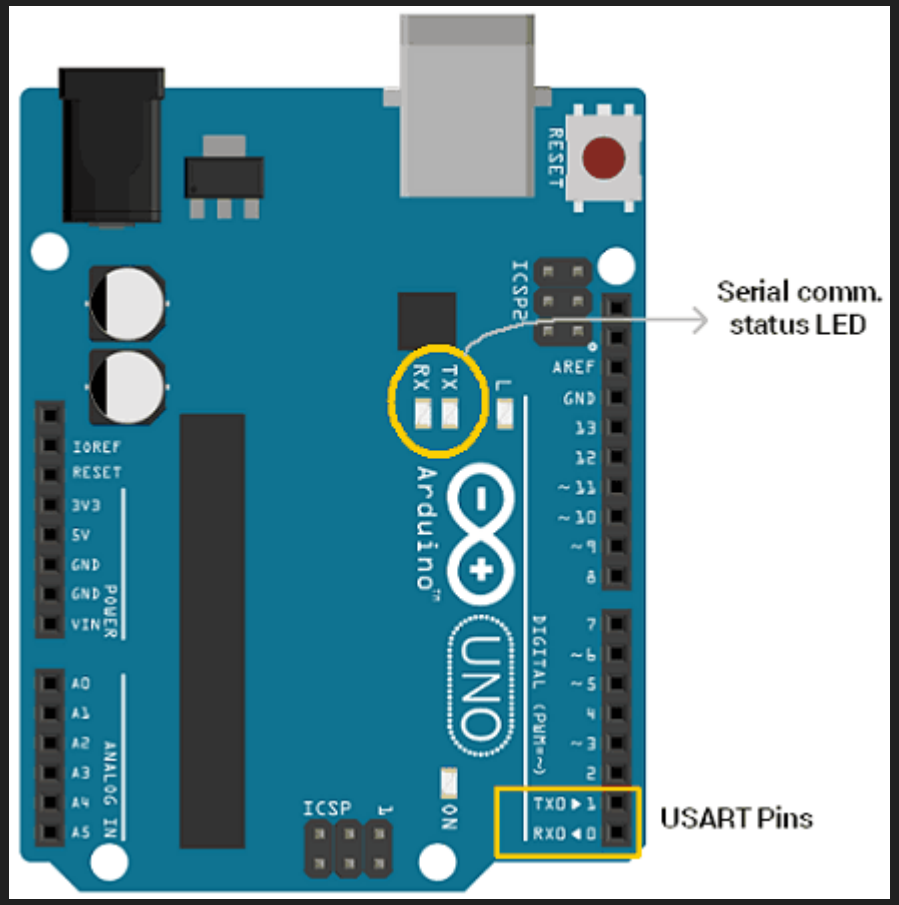


Puerto serie en Arduino

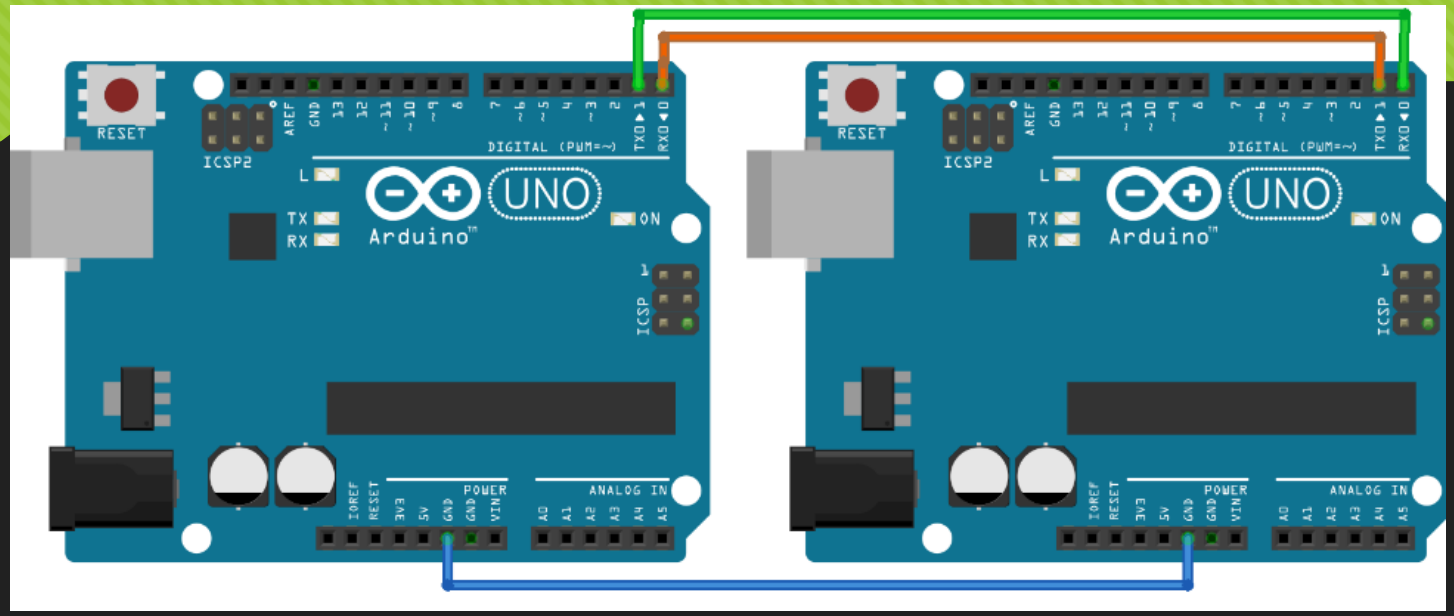
- Todas las placas Arduino tienen al menos un puerto serie disponible en los pines digitales 0 (RX) y 1 (TX) compartido con el USB.
- Por lo tanto no se recomienda usar estos pines como entradas/salidas digitales para otros sensores o usos que no sea comm.



Puerto serie en Arduino..



Puerto serie en Arduino..



- Otro caso. Comunicar dos dispositivos. Estas líneas se deben cruzar para comunicar dos dispositivos, es decir, el Tx del dispositivo 1 debe conectarse al Rx del dispositivo 2.
- El Tx del dispositivo 2 debe conectarse al Rx del dispositivo 1.
- Además ambos dispositivos deben compartir una masa común.
- Warning: Para fines prácticos usar otros pins, ya que corre riesgo de daño al equipo.

Conceptos

- Serial (TTL).- Los pines digitales 0 y 1 son los pines seriales del Arduino Uno. Son utilizados por el módulo USB integrado.
- UART.- Universal Asynchronous Receiver-Transmitters, Para enviar los datos por las líneas debemos usar un hardware que esté diseñado para llevar a cabo esa tarea y en este caso lo llamamos UART. Esta se encargará de leer datos cuando llegan, generar y gestionar interrupciones, enviar datos y gestionar los tiempos de bit

Conceptos

- SPI.- Serial Peripheral Interface, es un protocolo de datos en serie que utilizan los microcontroladores para comunicarse con uno o más dispositivos externos en una conexión tipo bus. Se pueden encontrar en los pines digitales 10-13 del Arduino
- I2C.- Es un protocolo de datos referido como "I2C bus", El protocolo I2C fue diseñado para permitir la comunicación entre componentes en una sola placa de circuito. Con I2C hay 2 cables denominados SCL y SDA. En el Arduino Uno se encuentran en los pines analógicos A4 y A5.

Hardware serial v.s. Software serial

- La mayoría de los microcontroladores tienen hardware diseñado para comunicarse con otros dispositivos en serie. El procesador utiliza software serial para simular puertos serie adicionales.
- El único inconveniente del software serial es que requiere más procesamiento y no puede admitir las mismas altas velocidades que el hardware en serie.

Reference serial

- Para manejar el puerto serie en Arduino, debemos leer a fondo la referencia de Arduino: http://arduino.cc/en/Reference/Serial
- Con las funciones que tenemos disponibles para trabajar con el puerto serie.

Reference serial..

Functions

if(Serial)
available()
availableForWrite()
begin()
end()
find()
findUntil()
flush()
parseFloat()
parseInt()
peek()
print()
println()
read()
readBytes()
readBytesUntil()
readString()
readStringUntil()
setTimeout()
write()
serialEvent()

Funciones

- begin().- establece la velocidad de la UART en baudios para la transmisión serie, también es posible configurar el número de bits de datos, la paridad y los bits de stop, por defecto es 8 bits de datos, sin paridad y un bit de stop. https://www.arduino.cc/en/Serial/Begin
- read() – lee el primer byte entrante del buffer serie. https://www.arduino.cc/en/Serial/Read
- write() – escribe datos en binario sobre el puerto serie. El dato es enviado como un byte o serie de bytes.
- print() – imprime datos al puerto serie como texto ASCII, también permite imprimir en otros formatos. https://www.arduino.cc/en/Serial/Print
- available() – da el número de bytes (caracteres) disponibles para leer en el puerto serie, son datos que han llegado y se almacenan en el buffer serie que tiene un tamaño de 64 bytes. https://www.arduino.cc/en/Serial/Available
- end() – deshabilita la comunicación serie permitiendo a los pines RX y TX ser usado como pines digitales. https://www.arduino.cc/en/Serial/End

Funciones..

- if(Serial) – especifica si el puerto serie está listo. https://www.arduino.cc/en/Serial/IfSerial
- find() – lee datos del buffer serie hasta encontrar el string buscado. https://www.arduino.cc/en/Serial/Find
- parseInt() – busca el siguiente entero válido en el stream de datos del puerto serie. https://www.arduino.cc/en/Serial/ParseInt
- readBytes() – lee datos del buffer serie y retorna el número de bytes en el buffer. https://www.arduino.cc/en/Serial/ReadBytes
- setTimeout() – configura el máximo de milisegundos de espera para la lectura del puerto serie. Por defecto es un segundo. https://www.arduino.cc/en/Serial/SetTimeout
- readBytesUntil() – lee caracteres del buffer serie y los guarda en un array de caracteres, la función termina si el carácter determinado es encontrado o la longitud determinada se ha leído o ha alcanzado el timeout. https://www.arduino.cc/en/Serial/ReadBytesUntil

Funciones..

- serialEvent() – llamado cuando hay datos disponibles. https://www.arduino.cc/en/Reference/SerialEvent
- flush() – espera hasta la transmisión completa de los datos salientes. https://www.arduino.cc/en/Serial/Flush
- peek() – devuelve el siguiente carácter del buffer serie pero sin borrarlo de él. https://www.arduino.cc/en/Serial/Peek
- readString() – lee caracteres del buffer serie y los guarda en un string. La función termina cuando se produce un timeout. https://www.arduino.cc/en/Serial/ReadString
- Como vimos, estas funciones se encuentran en el core del IDE de Arduino.

Ejemplo 1

```
1  byte byteRead;
2  void setup() {
3    Serial.begin(300);
4    Serial.write("Soy espejo y me reflejo\n");
5  }
6  void loop() {
7    if( Serial.available() )
8    {
9      byteRead = Serial.read();
10     Serial.write(byteRead);
11   }
12 }
```

Buffer serial

- Los puertos serie de los microcontroladores tienen un buffer que se va llenando hasta que nosotros lo vamos leyendo con la función read() que lo vamos vaciando, es una pila FIFO.
- El tamaño del buffer serie en el Arduino Uno es de 64 bytes, cuando se llena ese buffer el resto de elementos recibidos se pierden.

Software serial library

- ¿Y que pasa, si necesitas más puertos serie que los disponibles en un Arduino?
  - Cada microcontrolador tiene un número de puertos serie hardware (UART), pero se ha desarrollado la librería SoftwareSerial para permitir la comunicación serie sobre otros pines digitales de Arduino, usando software para replicar las funcionalidades de la comunicación serie.
- Es posible tener varios "puertos software" serial con velocidades de hasta 115200 bps.
- Ver funciones de la librería SoftwareSerial en: http://arduino.cc/en/Reference/SoftwareSerial

Software serial library..

- SoftwareSerial library ha sido desarrollada para permitir la comunicación serie en otros pines digitales del Arduino, usando el software para replicar la funcionalidad (de ahí el nombre de "SoftwareSerial").
- La versión de SoftwareSerial incluido en la versión 1.0 y posteriores se basa en la biblioteca NewSoftSerial escrita por Mikal Hart.

Limitaciones conocidas:

- Si se usa con puertos serie de programas múltiples, sólo uno puede recibir datos a la vez.
- No todos los pines en las placas Mega y Mega 2560 soportan interrupciones de cambio de nivel, por lo que solamente los siguientes se pueden utilizar para RX: 10, 11, 12, 13, 14, 15, 50, 51, 52, 53, A8 (62), A9 (63), A10 (64), A11 (65), A12 (66), A13 (67), A14 (68), A15 (69).
- En Arduino o Genuino 101 la velocidad máxima actual de RX es de 57600bps
- En Arduino o Genuino 101 RX no funciona en el pin 13

Funciones de SoftwareSerial library

- **SoftwareSerial(rxPin, txPin, inverse_logic).-** SoftwareSerial se utiliza para crear una instancia de un objeto SoftwareSerial, cuyo nombre es necesario proporcionar al igual que en el ejemplo siguiente:
- #include <**SoftwareSerial**.h>

```
const byte rxPin = 2;
const byte txPin = 3;

// configura un nuevo objeto serie
SoftwareSerial mySerial (rxPin, txPin);
```

- Ver más https://www.arduino.cc/en/Reference/SoftwareSerialConstructor

Funciones de SoftwareSerial library..

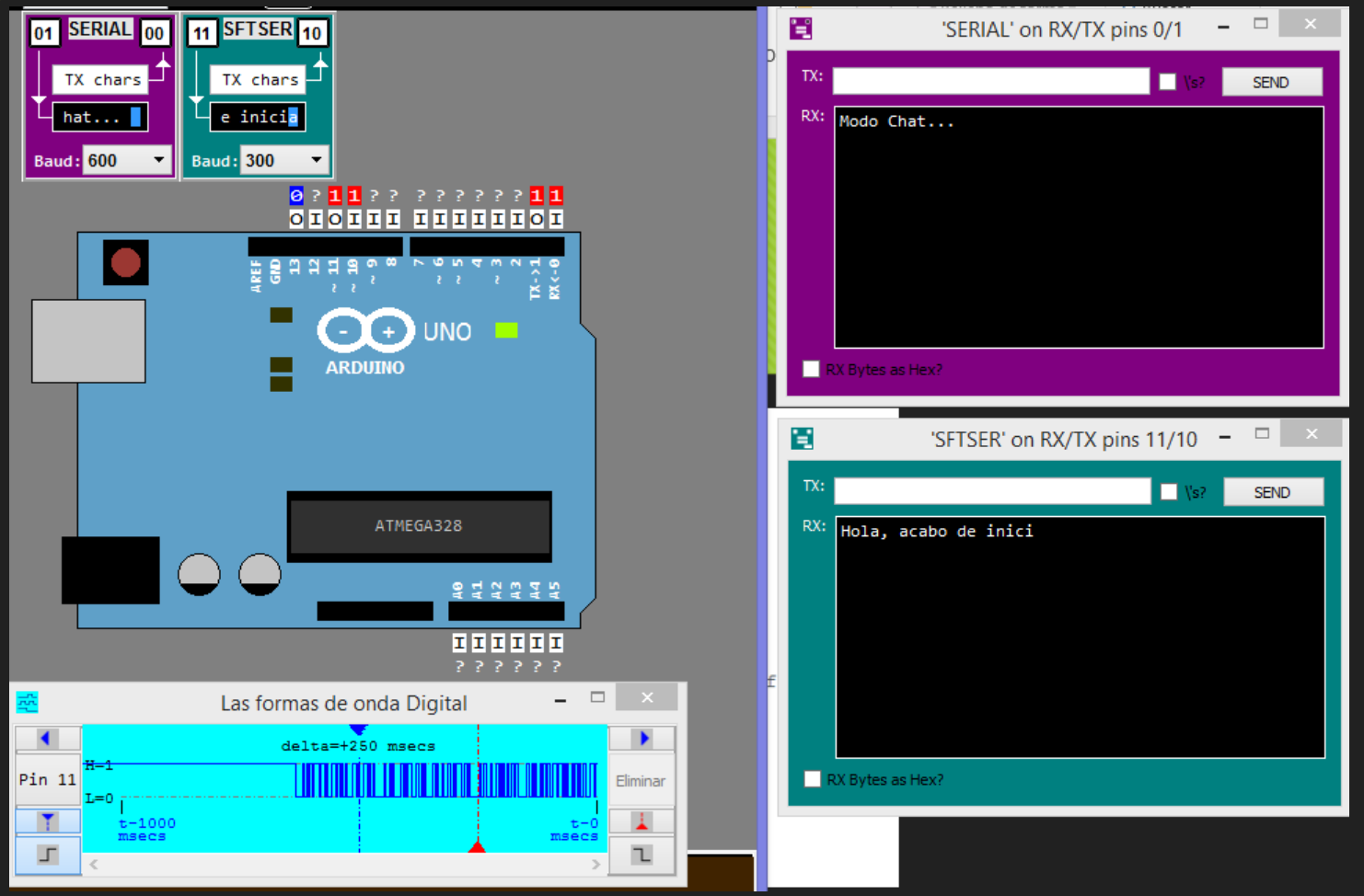
- available()
- begin()
- isListening()
- overflow()
- peek()
- read()
- print(data)
- println(data)
- listen()
- write(data)

Ejemplo 2

```
1  #include <SoftwareSerial.h>
2  SoftwareSerial mySerial(10, 11); // RX, TX
3
4  void setup() {
5    Serial.begin(600);
6    Serial.println("Modo Chat...");
7    mySerial.begin (300); //probar a velocidades diferentes en cada punto.
8    delay(1000);
9    mySerial.println("Hola, acabo de iniciar el chat");
10 }
11 void loop() { //en cada loop leer un caracter si hay en alguno de los buffers
12   if (mySerial.available()) {
13     Serial.print((char)mySerial.read());
14   }
15   if (Serial.available()) {
16     mySerial.print((char)Serial.read());
17   }
18 }
```

- Para ver ejemplos de uso. http://manueldelgadocrespo.blogspot.com/p/biblioteca.html

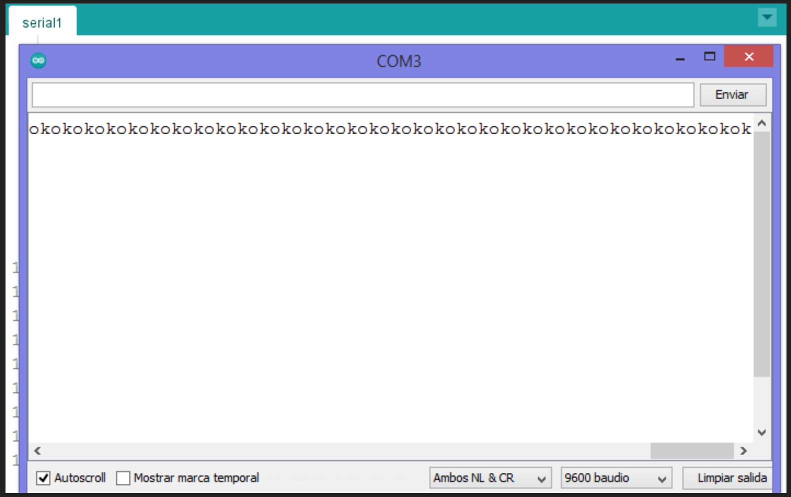
Ejemplo 2..

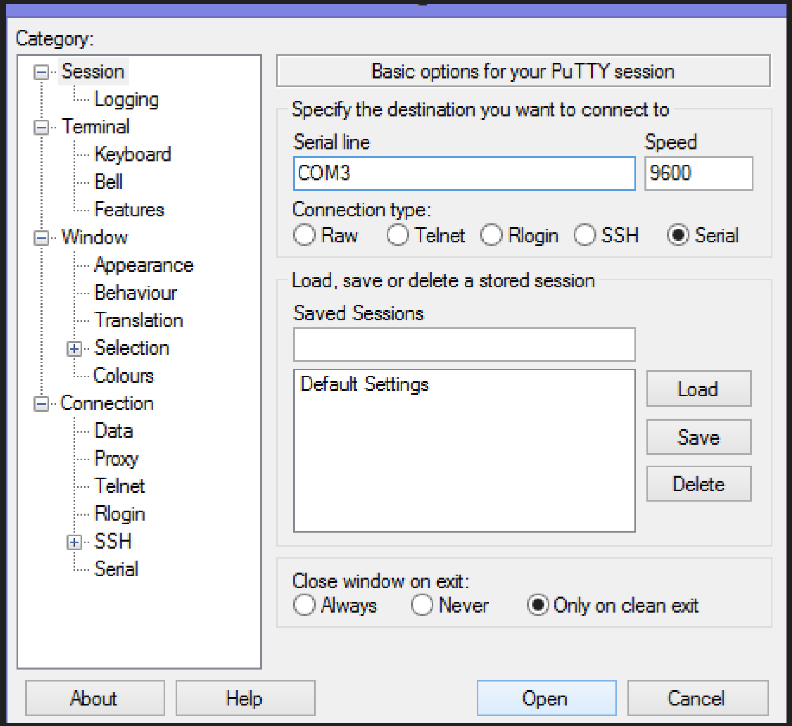

01 SERIAL 00
TX chars
hat...
Baud: 600
11 SFTSER 10
TX chars
e inicia
Baud: 300
UNO
ARDUINO
ATMEGA328
'SERIAL' on RX/TX pins 0/1
TX:
\s?
SEND
RX:
Modo Chat...
RX Bytes as Hex?
'SFTSER' on RX/TX pins 11/10
TX:
\s?
SEND
RX:
Hola, acabo de inici
RX Bytes as Hex?
Las formas de onda Digital
delta=+250 msecs
Pin 11
H=1
L=0
t-1000 msecs
t-0 msecs
Eliminar

Terminal serie

- Una vez visto cómo manejar el puerto serie en Arduino, si queremos comunicarnos con Arduino a través del puerto serie desde la PC debemos usar un emulador de terminal o terminal serie, el propio IDE de Arduino trae uno, pero existen otros más completos.

Terminal serie..





Video

- https://youtu.be/xWG_6SH_ft4
- ¿Cómo funciona el puerto serie?

Conclusión

- En realidad, el término puerto serie no hace referencia exclusivamente a este tipo de comunicación pero con el paso del tiempo se ha terminado asociando a ella.
- La comunicación serie está definida como una forma de comunicaciones punto a punto, es decir, una forma de conectar dos dispositivos entre sí.

Referencias

1. Comm serie Arduino. https://aprendiendoarduino.wordpress.com/2017/06/21/comunicacion-serie-arduino-3/
2. Serial reference. http://arduino.cc/en/Reference/Serial
3. Comm. Serial. https://www.arduino.cc/reference/en/language/functions/communication/serial/
4. SoftwareSerial Library Arduino. http://manueldelgadocrespo.blogspot.com/p/biblioteca.html
5. SoftwareSerial Library Arduino reference. https://www.arduino.cc/en/Reference/SoftwareSerial