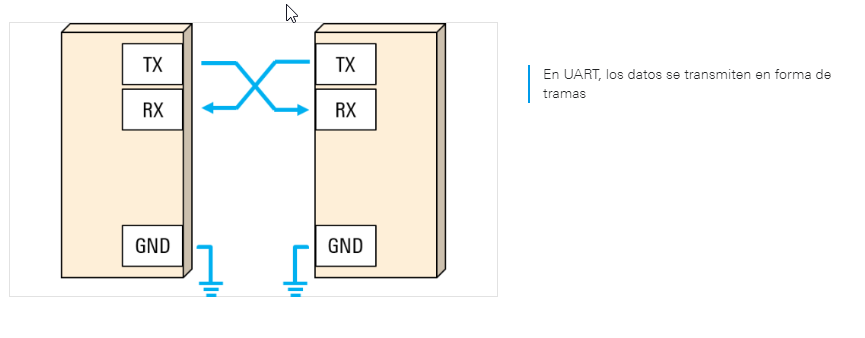
**Que es una transmisión serie o UART**

**UART significa transmisor-receptor asíncrono universal y define un protocolo o un conjunto de reglas, para intercambiar datos en serie entre dos dispositivos. El UART es muy simple y solo utiliza dos cables entre el transmisor y receptor para transmitir y recibir en ambas direcciones.**

Qué es UART?

UART (**universal asynchronous receiver / transmitte**r, por sus siglas en inglés), define un protocolo o un conjunto de normas para el intercambio de datos en serie entre dos dispositivos. UART es sumamente simple y utiliza solo dos hilos entre el transmisor y el receptor para transmitir y recibir en ambas direcciones. Ambos extremos tienen una conexión a masa. La comunicación en UART puede ser **simplex** (los datos se envían en una sola dirección), **semidúplex** (cada extremo se comunica, pero solo uno al mismo tiempo), o **dúplex completo** (ambos extremos pueden transmitir simultáneamente). En UART, los datos se transmiten en forma de tramas. A continuación se describe y explica el formato y el contenido de estas tramas de forma sucinta.

****

En UART, los datos se transmiten en forma de tramas

## ¿Dónde se utiliza UART?

UART fue uno de los primeros protocolos en serie. Los puertos en serie, que en su día proliferaron a gran escala, se basan casi siempre en el protocolo UART, y los dispositivos que utilizan interfaces RS-232, módems externos, etc. son ejemplos típicos de la aplicación de UART.  
En los últimos años la popularidad de UART ha disminuido, y se ha ido sustituyendo por protocolos como SPI e I2C para la comunicación entre chips y componentes. En lugar de comunicarse por un puerto en serie, la mayoría de los ordenadores y periféricos modernos utilizan hoy tecnologías como Ethernet y USB. Sin embargo, UART se sigue utilizando para aplicaciones de baja velocidad y bajo rendimiento, puesto que es muy simple, económico y fácil de integrar.

## Temporización y sincronización de protocolos UART

Una de las mayores ventajas de UART es que es asíncrono: el transmisor y el receptor no comparten la misma señal de reloj. Si bien esto simplifica en gran medida el protocolo, plantea determinados requisitos en el transmisor y el receptor. Puesto que no comparten un reloj, ambos extremos deben transmitir a la misma velocidad, previamente concertada, con el fin de mantener la misma temporización de los bits. Las velocidades en baudios más habituales en UART que se utilizan actualmente son 4800, 9600, 19,2 K, 57,6 K, y 115,2 K. Además de tener la misma velocidad en baudios, ambos extremos de una conexión UART deben utilizar también la misma estructura y parámetros de trama. La forma más sencilla de entender esto es observando una trama UART.

**Resumen**

* UART (universal asynchronous receiver/transmitter, por sus siglas en inglés), es un protocolo simple de dos hilos para el intercambio de datos en serie.
* Asíncrono quiere decir que los extremos no comparten un reloj, de modo que para que UART funcione debe estar configurada la misma velocidad de bits o baudios en ambos extremos de la conexión.
* Los bits de inicio y de parada se utilizan para indicar dónde empiezan y terminan los datos de usuario, o para «enmarcar» los datos.
* Se puede utilizar un bit de paridad opcional para detectar errores de bit únicos.
* UART sigue utilizándose a gran escala como protocolo de datos en serie, pero se ha ido reemplazando en los últimos años en algunas aplicaciones por tecnologías como SPI, I2C, USB y Ethernet.

En una comunicación serial las características más importantes son:

. la velocidad de transmisión

. El número de bits de datos

. El número de bits de paro

. Y si cuenta con bit de paridad

Para que dos puertos se puedan comunicar, es necesario que las características sean iguales.

**Velocidad de transmisión (baud rate)**

● Indica el número de bits por segundo que se transfieren, y se mide en baudios (bauds). Por ejemplo, 300 baudios representa 300 bits por segundo.

● Cuando se hace referencia a los ciclos de reloj se está hablando de la velocidad de transmisión.

Por ejemplo, si el protocolo hace una llamada a 4800 ciclos de reloj, entonces el reloj está corriendo a 4800 Hz, lo que significa que el puerto serial está muestreando las líneas de transmisión a 4800 Hz.

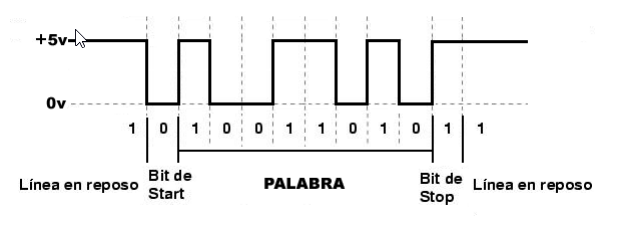
**Bits de datos**

● Se refiere a la cantidad de bits (palabra) en la transmisión.

● Cuando la computadora envía un paquete de información, el tamaño de ese paquete no necesariamente será de 8 bits.

● Las cantidades más comunes de bits por paquete son 5, 7 y 8 bits.

● El número de bits que se envía depende en el tipo de información que se transfiere.

****

● Por ejemplo, la representación de caracteres ASCII estándar tiene un intervalo de valores que va de 0 a 127, es decir, utiliza 7 bits.

● Para ASCII extendido es de 0 a 255, lo que utiliza 8 bits.

● Si el tipo de datos que se está transfiriendo es texto simple (ASCII estándar), entonces es suficiente con utilizar 7 bits por paquete para la comunicación.

● Un paquete se refiere a una transferencia de un byte, incluyendo los bits de inicio/paro, bits de datos, y paridad. Debido a que el número actual de bits depende en el protocolo que se seleccione, el término paquete se usar para referirse a todos los casos.

**Bits de paro**

● Usado para indicar el fin de la comunicación de un solo paquete.

● Los valores típicos son 1, 1.5 o 2 bits.

● Debido a la manera como se transfiere la información a través de las líneas de comunicación y que cada dispositivo tiene su propio reloj, es posible que los dos dispositivos no estén sincronizados. Por lo tanto, los bits de paro no sólo indican el fin de la transmisión sino además dan un margen de tolerancia para esa diferencia de los relojes.

● Mientras más bits de paro se usen, mayor será la tolerancia a la sincronía de los relojes, sin embargo la transmisión será más lenta.

**Paridad**

● Es una forma sencilla de verificar si hay errores en la transmisión serial.

● Existen cuatro tipos de paridad:

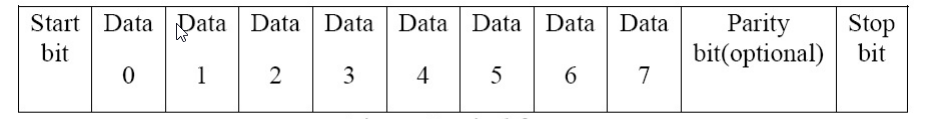
● par,

● impar,

● marcada y

● espaciada.

● La opción de no usar paridad alguna también está disponible.



En caso de habilitar la paridad par o impar, el puerto serial fijará el bit de paridad (el último bit después de los bits de datos) a un valor para asegurarse que la transmisión tenga un número par o impar de bits en estado lógico alto.

Por ejemplo, si la información a transmitir es 011 y la paridad es par, el bit de paridad sería 0 para mantener el número de bits en estado alto lógico como par.

● Si la paridad seleccionada fuera impar, entonces el bit de paridad sería 1, para tener 3 bits en estado alto lógico.

● La paridad marcada y espaciada en realidad no verifican el estado de los bits de datos; simplemente fija el bit de paridad en estado lógico alto para la marcada, y en estado lógico bajo para la espaciada.

● Esto permite al dispositivo receptor conocer de antemano el estado de un bit, lo que serviría para determinar si hay ruido que esté afectando de manera negativa la transmisión de los datos, o si los relojes de los dispositivos no están sincronizados.

# XON/XOFF

En [Informática](https://es.wikipedia.org/wiki/Inform%C3%A1tica), se trata de un [protocolo](https://es.wikipedia.org/wiki/Protocolo_de_red) para el control del flujo de datos entre dispositivos informáticos (ordenadores, impresoras, etc.)

Básicamente este protocolo consiste en usar estos dos caracteres para controlar el flujo de caracteres. En la tabla [ASCII](https://es.wikipedia.org/wiki/ASCII), el carácter XON es el código 17, mientras que el XOFF es el 19. Así el carácter XON se denomina algunas veces DC1, y el carácter XOFF se le conoce también por DC3. Desde el punto de vista del teclado de un ordenadores estos caracteres se suelen corresponder con control-S (XOFF) y con control-Q (XON).

## Funcionamiento del protocolo

Cuando el receptor del mensaje desea que el emisor detenga el flujo de datos, manda carácter XOFF (carácter de pausa) y el emisor al recibirlo detiene la emisión del mensaje. Hay que tener en cuenta que desde que se manda el carácter XOFF hasta que se interrumpe la emisión de datos, aún pueden llegar algunos datos. Por lo tanto no se debe esperar a tener el buffer totalmente lleno para mandar el XOFF, sino que lo habitual es mandarlo cuando, por ejemplo, está a un 75% de su capacidad.

Para que el flujo se reanude, el emisor debe recibir un carácter XON. Este carácter lo manda el receptor cuando tiene suficiente espacio en su buffer de recepción, por ejemplo cuando su nivel de llenado es del 25%.