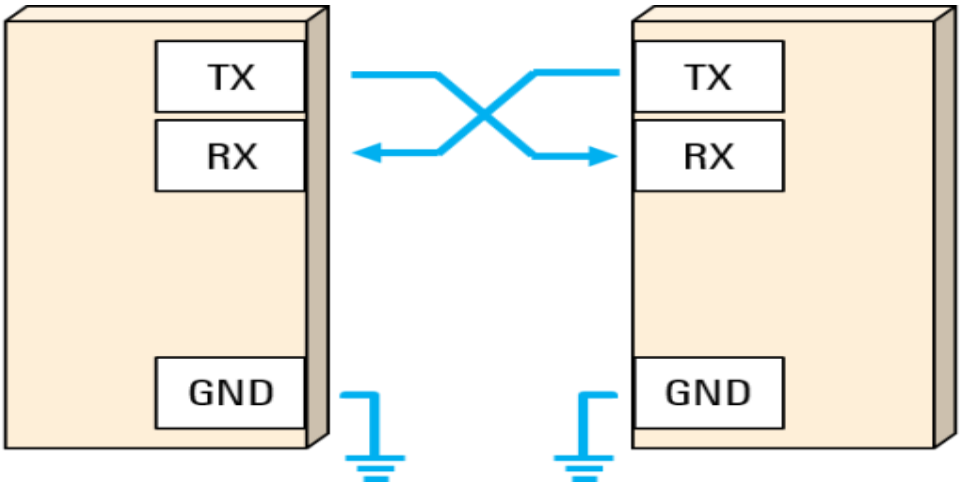
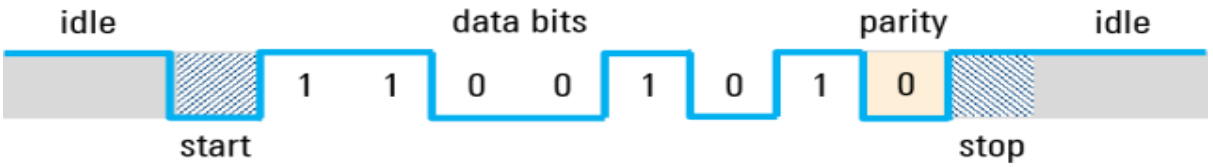
What is UART?

UART stands for universal asynchronous receiver / transmitter and defines a protocol, or set of rules, for exchanging serial data between two devices. UART is very simple and only uses two wires between transmitter and receiver to transmit and receive in both directions. Both ends also have a ground connection. Communication in UART can be simplex (data is sent in one direction only), half-duplex (each side speaks but only one at a time), or full-duplex (both sides can transmit simultaneously). Data in UART is transmitted in the form of frames.



UART frame format



Start and stop bits

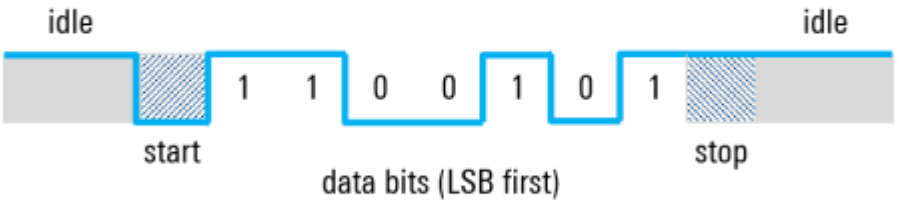
Because UART is asynchronous, the transmitter needs to signal that data bits are coming. This is accomplished by using the start bit. The start bit is a transition from the idle high state to a low state, and immediately followed by user data bits.
After the data bits are finished, the stop bit indicates the end of user data. The stop bit is either a transition back to the high or idle state or remaining at the high state for an additional bit time. A second (optional) stop bit can be configured, usually to give the receiver time to get ready for the next frame, but this is uncommon in practice.

Data bits

The data bits are the user data or “useful” bits and come immediately after the start bit. There can be 5 to 9 user data bits, although 7 or 8 bits is most common. These data bits are usually transmitted with the least significant bit first.

Example:

If we want to send the capital letter “S” in 7-bit ASCII, the bit sequence is 1 0 1 0 0 1 1. We first reverse the order of the bits to put them in least significant bit order, that is 1 1 0 0 1 0 1, before sending them out. After the last data bit is sent, the stop bit is used to end the frame and the line returns to the idle state.



Parity bit

A UART frame can also contain an optional parity bit that can be used for error detection. This bit is inserted between the end of the data bits and the stop bit. The value of the parity bit depends on the type of parity being used (even or odd):

In even parity, this bit is set such that the total number of 1s in the frame will be even

In odd parity, this bit is set such that the total number of 1s in the frame will be odd

Example:Capital “S” (1 0 1 0 0 1 1) contains a total of three zeros and 4 ones. If using even parity, the parity bit is zero because there already is an even number of ones. If using odd parity, then the parity bit has to be one in order to make the frame have an odd number of 1s. The parity bit can only detect a single flipped bit. If more than one bit is flipped, there’s no way to reliably detect these using a single parity bit.