The speed of data transmission and reception of both devices (tx and rx) must be same to avoid the loss of the data

Synchronization: when a device sends data to a system it needs to notify the receiver that it is sending some data and receiver must send the acknowledgement that it received the data

Types of communication protocol:

1.Synchronous communication protocol: the devices which are communicating with each other share the same clock pulses

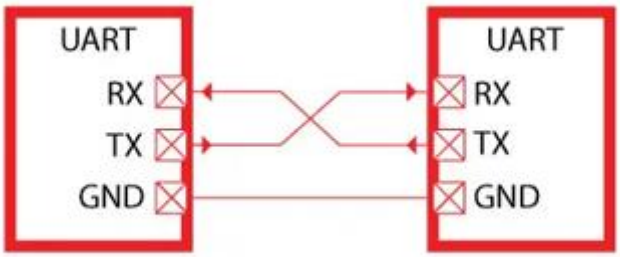
- In the embedded system chips, there are d flip flops as transmitter and receiver the output of d flip flop at transmitter is connected to the input of the d flip flop of receiver.
- After sending data the transmitter sends the closing bit to notify the receiver that it has sent all the information
- When receiver receives the data it sends the acknowledgement bit. But if the receiver doesn't send the ack bit then the transmitter will understand that receiver did not get the packet of data and it will send the same data stream to the receiver again. This cycle keeps on repeating.
- Examples: SPI, I2C

2.Asynchronous communication protocol: here there is no need of clock synchronization

- **Baud rate of communication**: Transmission speed of data packet sent by the transmitter
- The d flip flops of the transmitter and receiver does not have common clock pulse, but they have internal clock signals which are generated by their timers
- Before communication we will configure the same baud rate for the transmitter and receiver and there is no ack signal coming from receiver
- Examples: UART, CAN

UART protocol

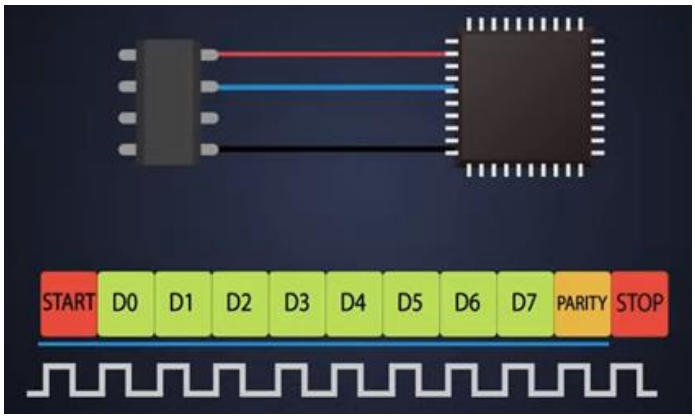
- Universal Asynchronous Receiver Transmitter
- When two devices need to communicate with each other, there needs to be at least 2 connections, one is for transmission and other is for reception line
- It is a character-oriented protocol that means data is sent byte by byte
- **Bit frame:** the sequence of bits which indicate the receiver about the starting and ending of the data communication



- Data frame:

The clock pulses provided to the transmitter shift register

It contains:



1.start bit: When a transmitting device wants to start the communication, it pulls the tx line to low (1 to 0) stays low for one clock pulse due to this receiver will understand that tx wants to send the data

2.main data: After that there is a frame of 8-bit data which needs to be delivered at the receiver

1 = 5v

0 = 0v

3.parity bit – optional: it is used only if the devices need to check the error present in the data stream

4.stop bit: it notifies rx that is about end of communication

The stop bit is actually high, it stays high for one clock pulse and further more

Initially the transmission line and reception line of the UART are high which indicates that line is idle and there is no data transmission

UART data frame is a 10-bit data frame including start and stop bit

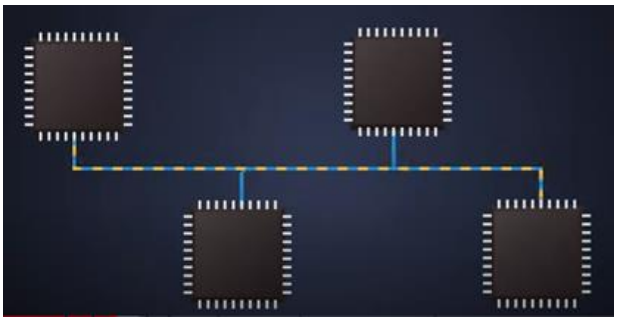
TOPOLOGIES IF COMMUNICATION

1.bus topology: there is a bus of communication wire all of the devices get connected to this bus and they communicate with each other through this platform itself so if a device needs to talk with each other, it will just call it they will communicate and

if a device needs to talk to all other devices, it will just broadcast the message

if any of the devices gets disconnected due to any reason still other devices will communicate

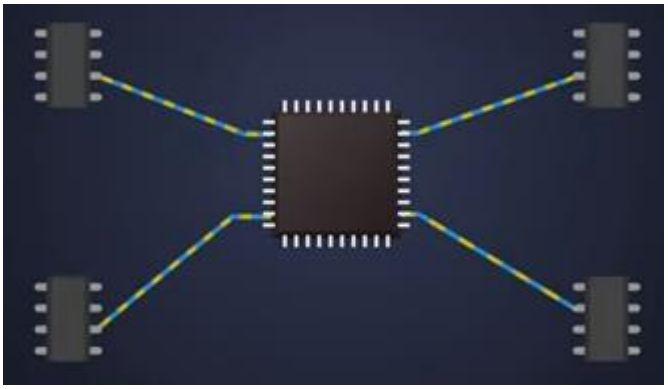
this protocol is used in CAN communication



2.star topology: it has one master and others as slaves

This master can only talk to all the slaves

If the master is not there then there will be no communication



3.ad-hoc/peer to peer topology

Only two devices need to connect and talk where UART communication is a default choice for easy interface

There is no master or slave concepts in this topology, it is an ad-hoc type