UART - An Asynchronous Serial Interface

- UART stands for Universal Asynchronous Receiver Transmitter
- Used ubiquitously in serial ports such as RS-232 and in embedded systems

- UART is **asynchronous** — *no explicit clock signal sent from Tx to Rx*
- UART is **point-to-point** (i.e., *one Tx device and one Rx device*)
- UART data is sent in **data frames with specific frame format**

Pad	Data	Pad

↖ there is some data padding

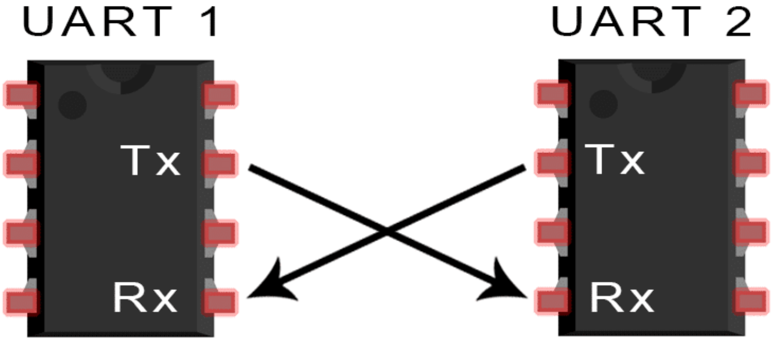
- Prerequisite for UART communication is that Tx and Rx must agree on:
  - Data rate (baud rate in bps)
  - Data frame format

baud rate : symbol rate (rate at which symbols fly over wire connecting Tx, Rx)

baud rate = bits per sec, for our case

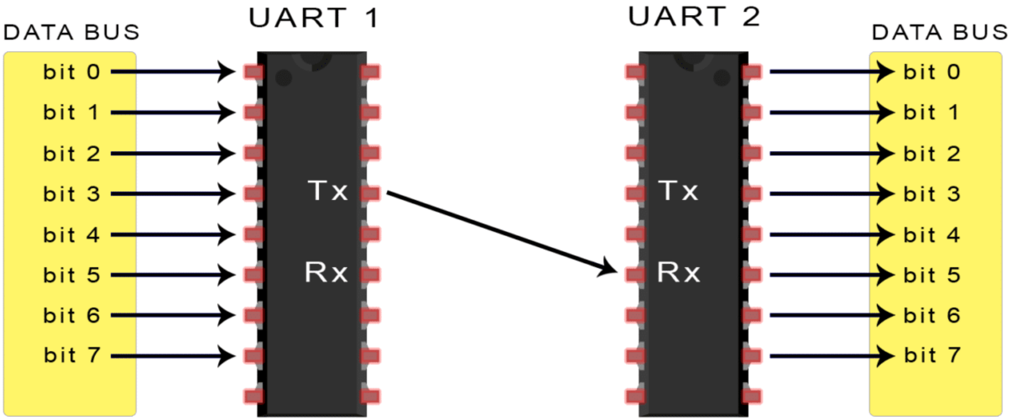
- **Key Idea:** Use the pre-agreed upon data rate and frame format to do **local clock recovery** (i.e., generate a clock signal at the Rx to sample bits on the line)

UART Basics



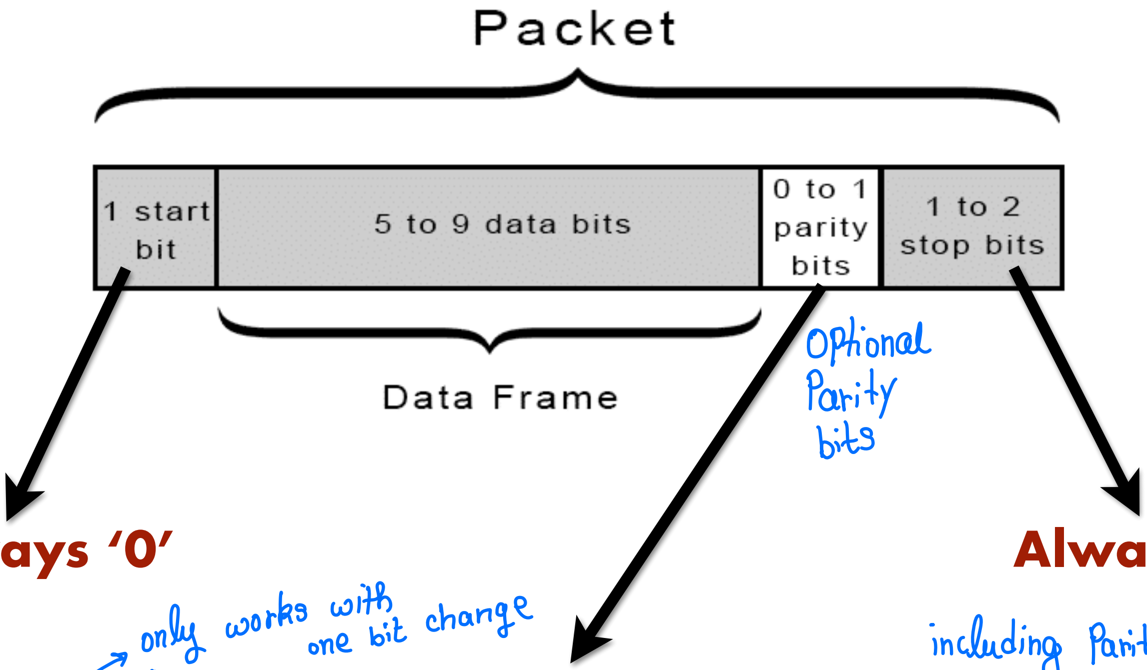
duplex

Tx pin of device 1 connected to Rx pin of device 2



Parallel data serialized at device 1 and de-serialized at device 2

UART Basics - Data Frame Format



Optional Parity bits

Always '0'

Always '1'

only works with one bit change

Odd parity (1): Total number of 1's in data frame including Parity should be odd

Even parity (0): Total number of 1's in data frame should be even

Stop bit,
start bit need to have different polarities

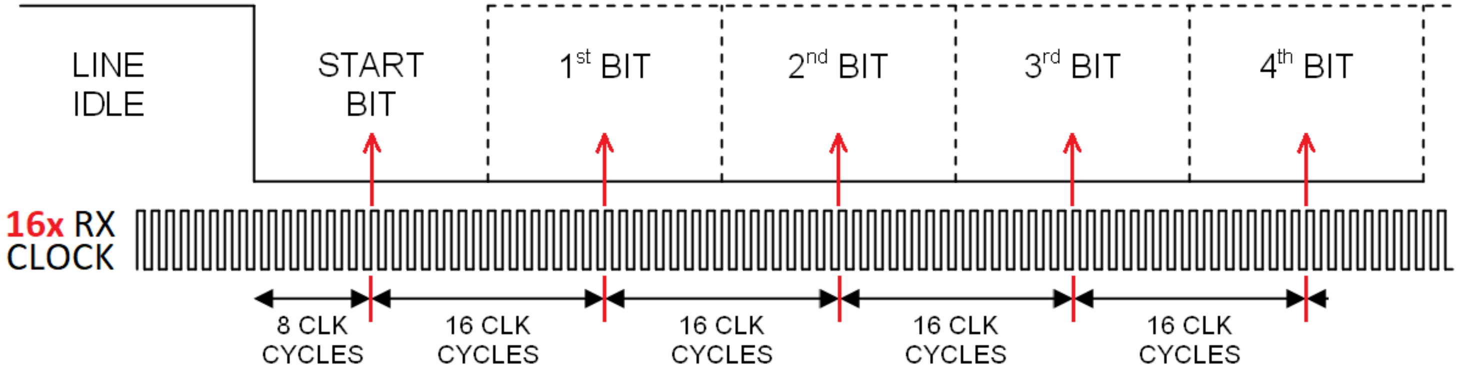
Why always '0'?
- holding the wire at logic-high

UART Basics - Synchronization

- Receiver device generates a local clock with frequency that is a multiple of the baud rate (typically 8x the baud rate or 16x the baud rate)

- **Key step for UART to work is that the data line is held HIGH (i.e., logic 1) when there is no data being sent**

- When a START bit is sent (remember, a START bit is always 0), there will be a 1 -> 0 transition (i.e., negative edge) on the data line)



- After locking on to the middle of a the start bit, the Rx device samples every 16 bits to get to the middle of the next bit

when stop bit ≠ 1, then there's
Tx, Rx different baud rates → Framing Error

UART Basics

- A common data format is 8-N-1 (8 data bits, No parity bits, 1 stop bit), commonly used to transmit ASCII character data
- With this format, character transmission rate is the 1/10 of baud rate

- Can be full-duplex (independent transmit chain from device 2 to device 1)

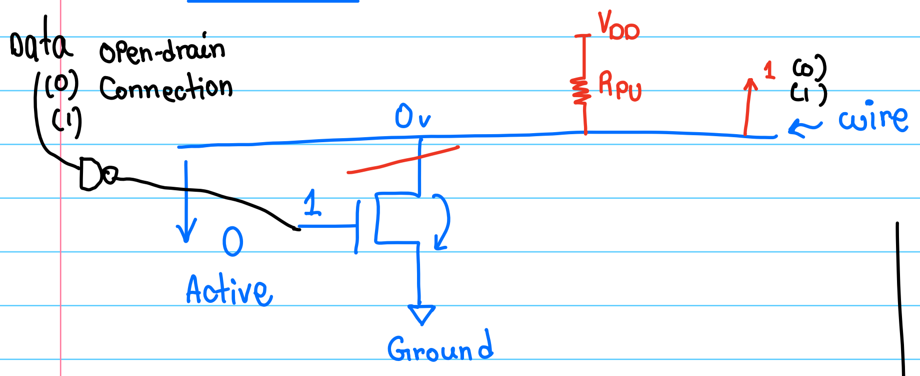
Advantages

- Only 2 wires (Tx and Rx) for two-way communication
- No clock signal needed
- Has basic error checking

Disadvantages

- Needs clock recovery
- Needs serialization and de-serialization of data
- Only point to point (no multiple masters, no multiple slaves)
- Clocks on the devices need to be close to each other (~5%)

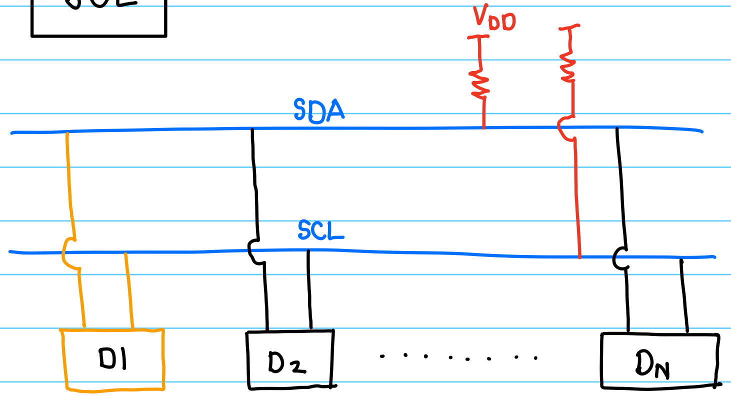
I^2C



Physical layer mechanism of bit transfer in I^2C

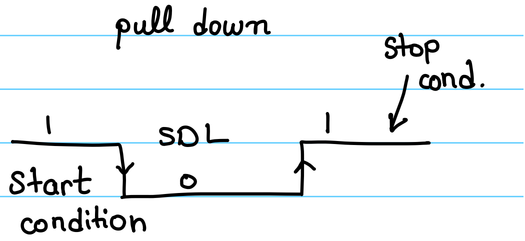
I^2C : syn protocol, serial

SDA
SCL
(2 lines open-drain)

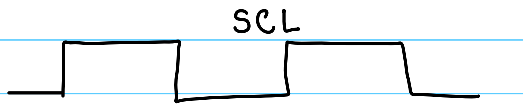


Open Drain

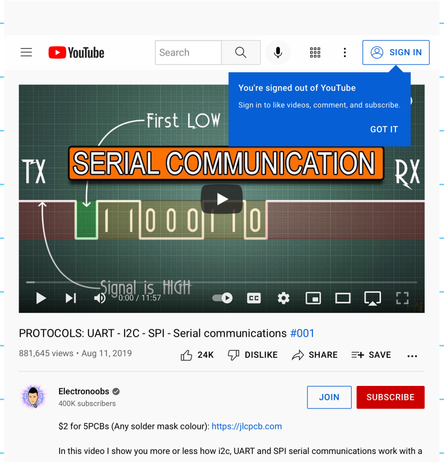
- Resistive
  pull up
- Active
  pull down



no other controller will try to interfere with the transaction until the bus is released



https://youtu.be/lyGwvGzrqp8



You're signed out of YouTube

Sign in to like videos, comment, and subscribe.

GOT IT

PROTOCOLS: UART - I2C - SPI - Serial communications #001

881,645 views • Aug 11, 2019

Electronoobs
400K subscribers

$2 for 5PCBs (Any solder mask colour): https://jlcpcb.com

In this video I show you more or less how i2c, UART and SPI serial communications work with a