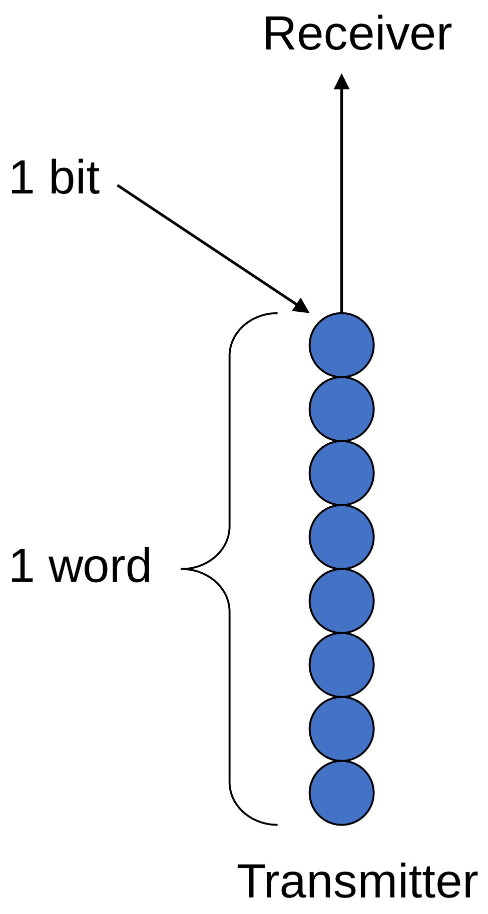
Serial Communication

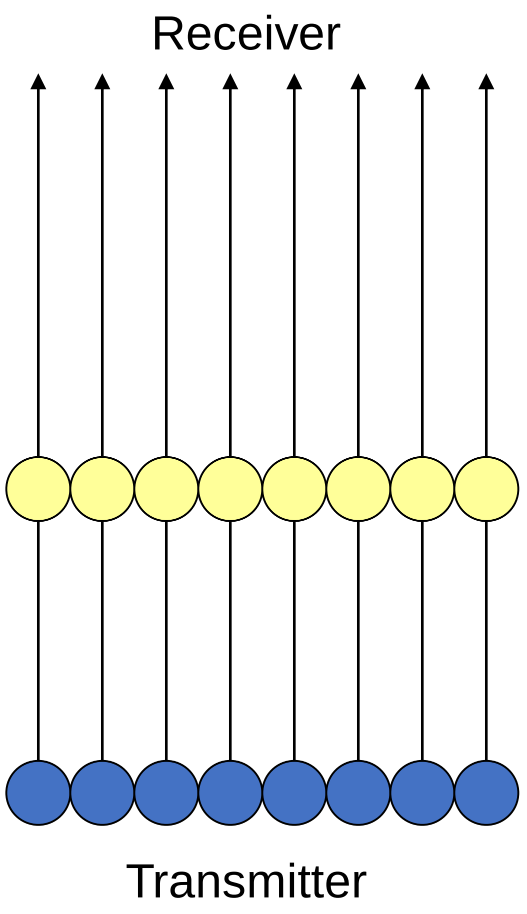
Wired

Transmission Illustration

Serial



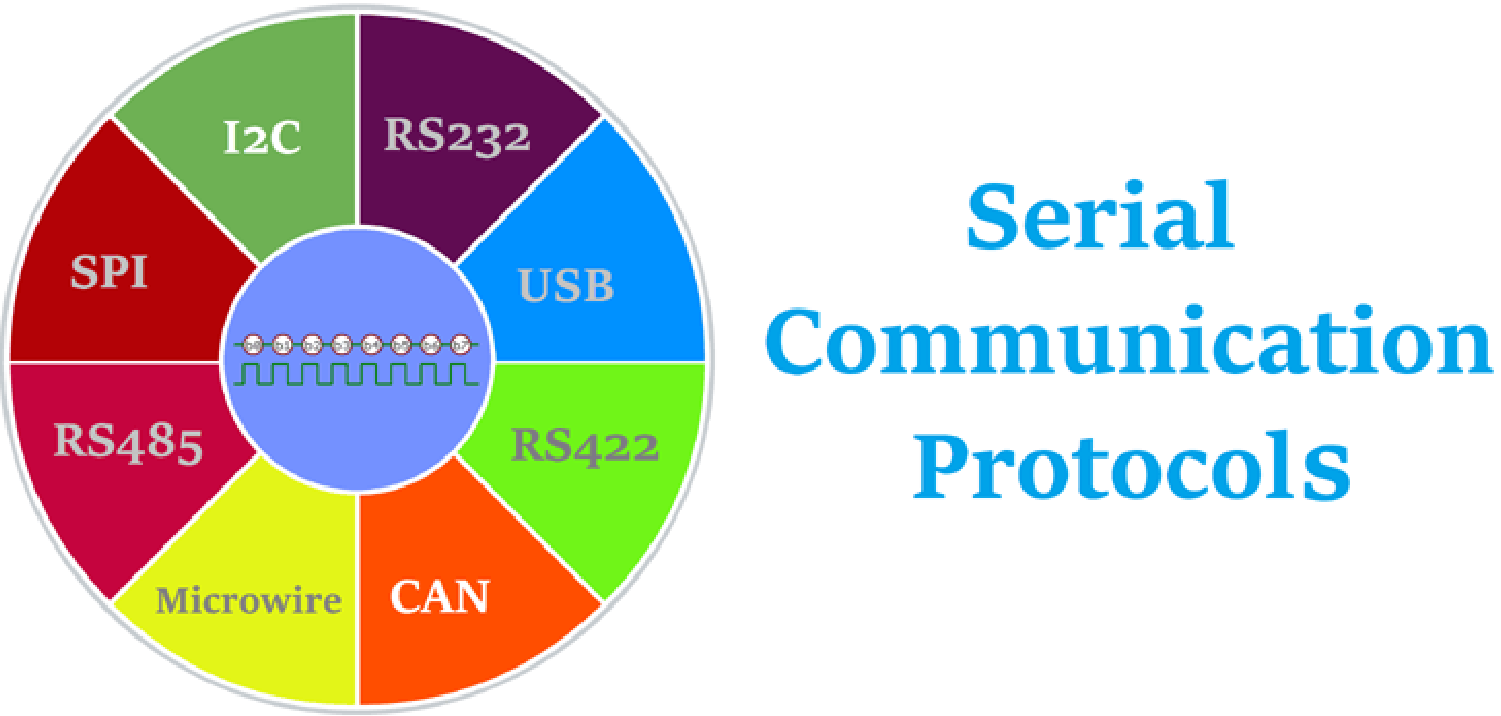
Parallel



Data Transmission

	Serial	**Parallel**
Cost	Cheap	Expensive
Speed	Slow	Fast
Transmission Amount	Single bit	8 bits (8 data lines) Transmitter & Receiver
Transmission Lines	One line to transmit one to receive	8 lines for simultaneous transmission
Transmission Distance	Long distance	Short distance (synchronization)
Example	Modem	Printer Connection

https://circuitdigest.com/tutorial/serial-communication-protocols



Serial Communication Overview
ME4447/6405

Serial Communications

- Synchronous
  - Synchronous Peripheral Interface (SPI)
- Asynchronous
  - Serial Communication Interface (SCI)

Synchrononous - SPI

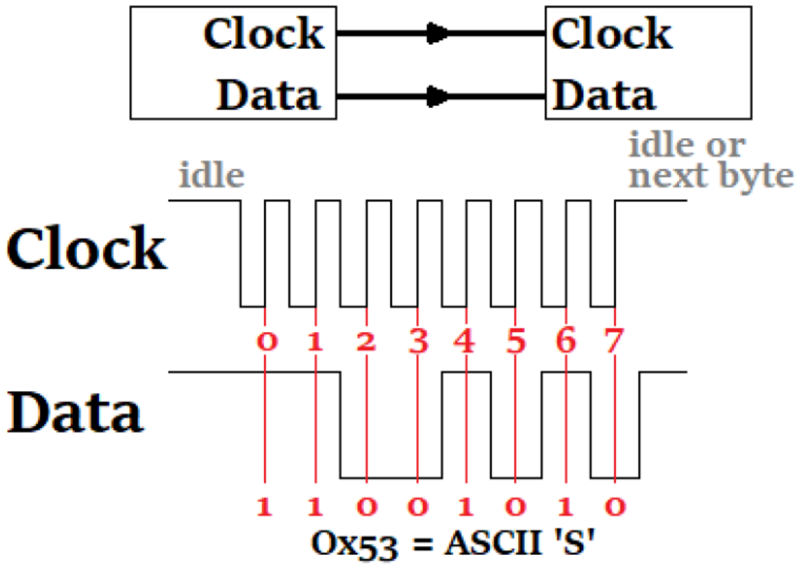
- Constant transmission of data
- Clocks of Transmitter and Receiver must be synchronized
- No safeguard against error or noise
- Data rates depend on clock rates
- Flexible to communication with peripheral devices
  - LCD drivers, A/D converter, other microprocessors
- Simultaneously transmits and receives data
  - Transmission line, Receiving line, and Ground

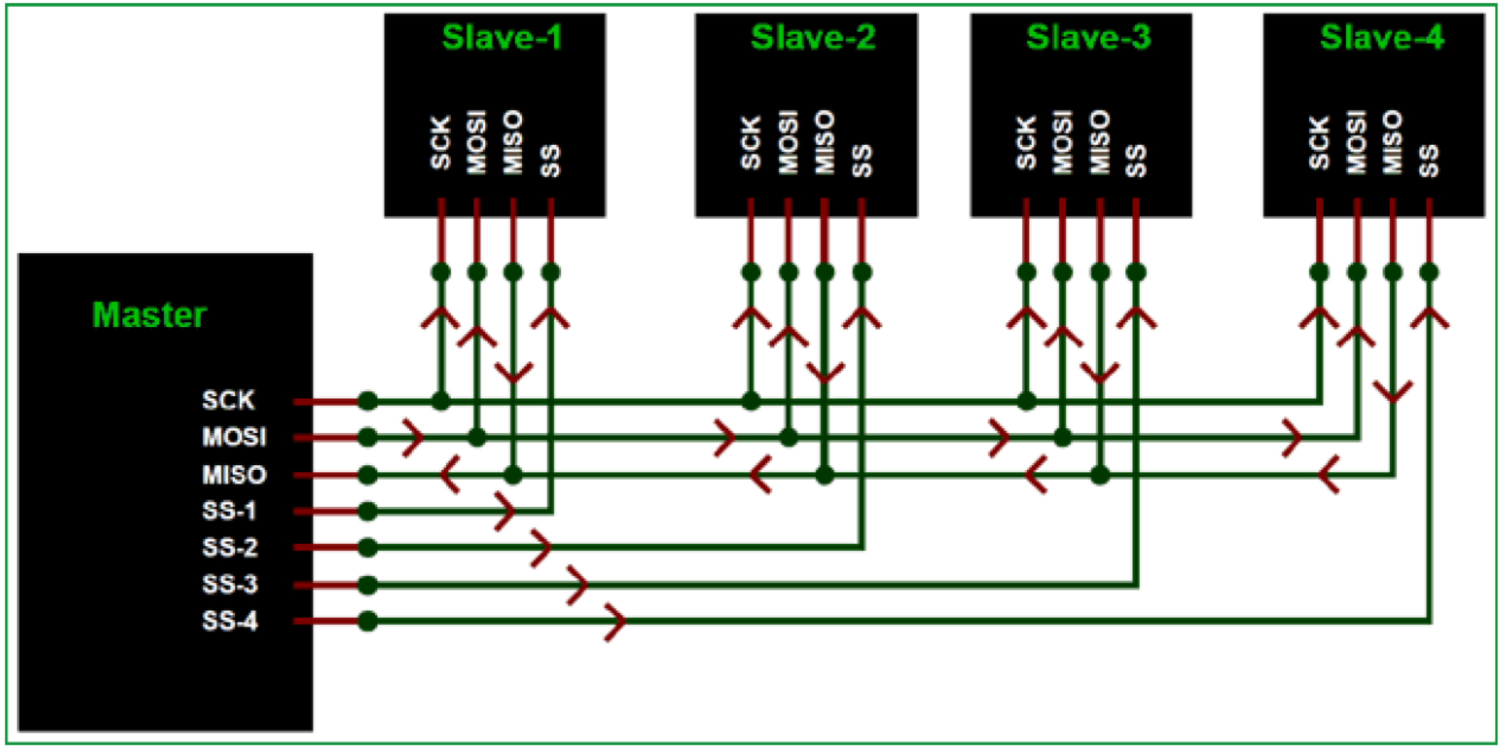
SYNCHRONOUS

- **USART** (Universal Synchronous-Asynchronous Receiver-Transmitter)
- **USB** (Universal Serial Bus)
- **RS232 () :** The **RS-232** spec provides for both **synchronous** and **asynchronous** but PC's use a **UART chip** such as a 16450, 16550A, or 16650 and can't deal with **sync**.
- **I2C or I^{2}C** (Inter-Integrated Circuit, pronounced as "I squared C" or "I two C")
- **SPI** (Serial Peripheral Interface)

Synchronous Serial Interface:

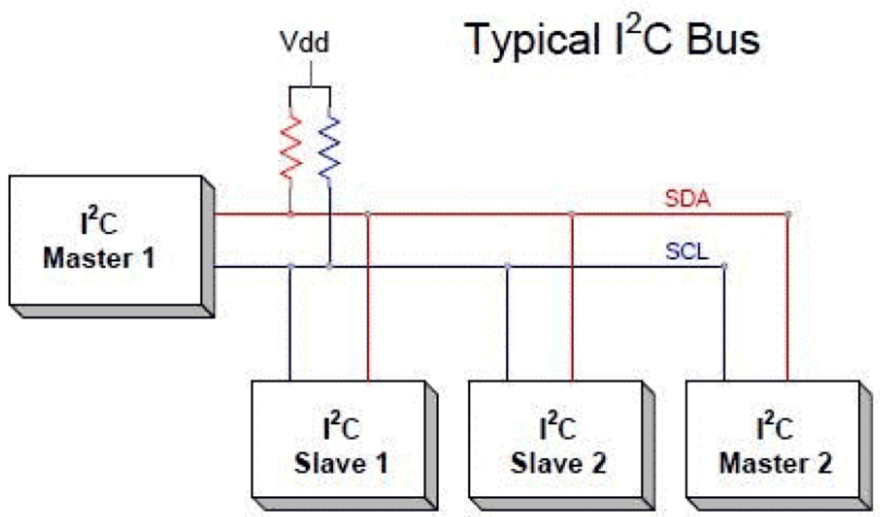
It is a point-to-point connection from a master to slave. In this type of interface, all the devices use single CPU bus to share data and clock. The data transmission becomes faster **with same bus to share clock** and data. Also there is no mismatch inbaud rate in this interface. In transmitter side, there is a shift of the data onto serial line providing the clock as a separatesignal as there is no start, stop and parity bits are added to data. In receiver side, the data is being extract using the clockprovided by the transmitter and converts the serial data back to the parallel form. **The well-known examples are I2C and SPI.**





Most of the microcontrollers have inbuilt support for SPI and can be directly connected SPI supported device:

I2C Serial Communication



Inter integrated circuit (I2C) two-line communication between different ICs or modules where two lines are SDA (Serial Data Line) and SCL (Serial Clock Line). Both the lines must be connected to a positive supply using a pull up resistor. I2C can deliver speed up to 400Kbps and it uses 10 bit or 7 bit addressing system to target a specific device on the i2c bus so it can connect up to 1024 devices. It has limited length communication and is ideal for onboard communication. I2C networks are easy to setup since it uses only two wires and new devices can simply be connected to the two common I2C bus lines. Same like SPI, microcontroller generally have I2C pins to connect any I2C device:

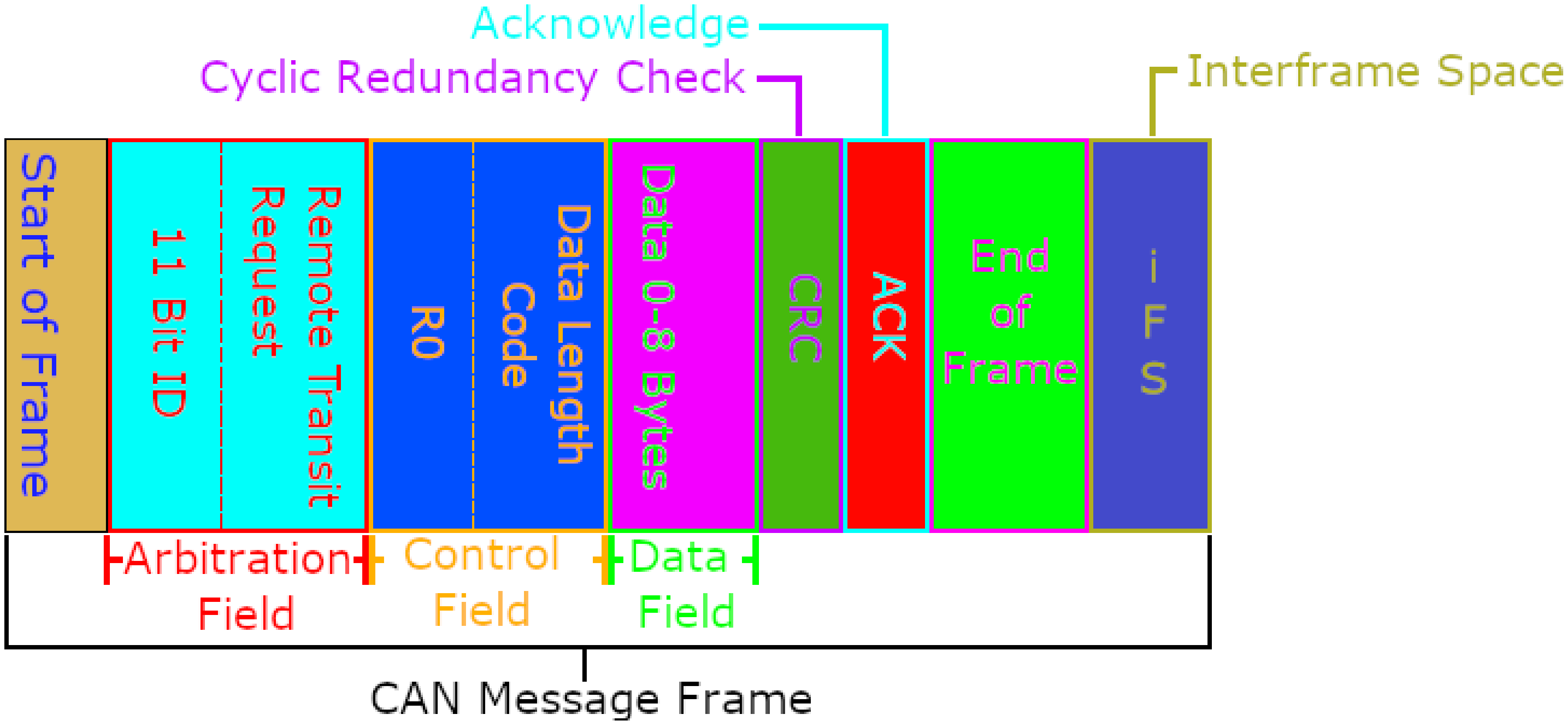
USB

USB (Universal Serial Bus) is widely protocol with different versions and speeds. A maximum of 127 peripherals can be connected to a single USB host controller. USB acts as "plug and play" device. The USB are used in almost devices such as keyboards, printers, media devices, cameras, scanners and mouse. It is designed for easy installation, faster data rated, less cabling and hot swapping. It has replaced the bulkier and slower serial and parallel ports. USB uses differential signalling to reduce interference and allow high-speed transmission over a long distance.

TYPE A	**TYPE B**	**TYPE C**
USB A-Type	USB B Type	USB C Type
USB 3.0 A-Type	USB 3.0 B-Type	
Micro USB A	Micro USB B	
	USB Mini-b (5 pins)	
	USB 3.0 Micro B	
	USB Mini-b (4 pins)	

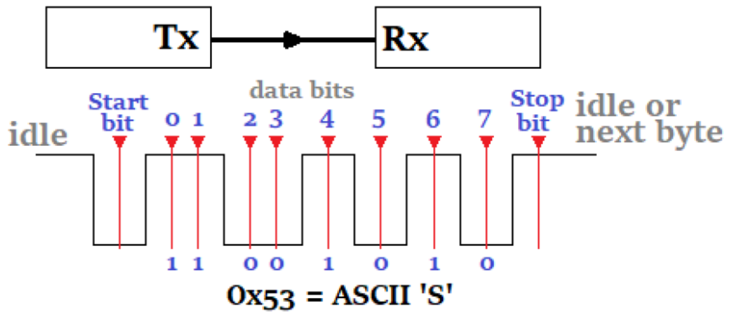
CAN

The Controller Area Network (CAN) is used in e.g. **automotive** to allow communication between ECUs (Engine Control Units) and sensors. The CAN protocol is robust, low-cost and message based and covers in many applications - e.g. cars, trucks, tractors, industrial robots. The CAN bus system allows for central error diagnosis and configuration across all ECUs. CAN messages are prioritized via IDs so that the highest priority IDs are non-interrupted.



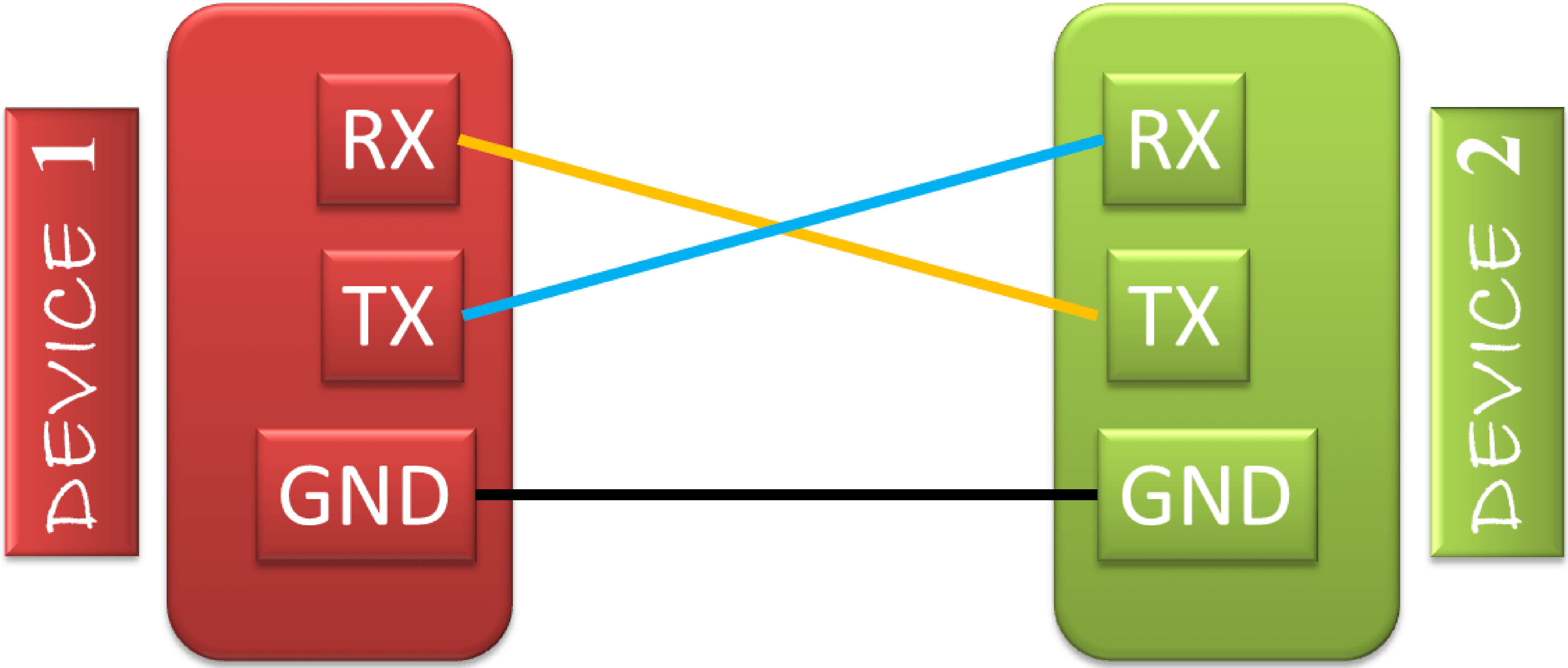
Asynchronous Serial Interface:
In asynchronous Serial Interface, the external clock signal is absent. The Asynchronous Serial Interfaces can be seen in mostly in long distance applications and are a perfect fit for the stable communication. In asynchronous Serial Interface the absence
of external Clock Source makes it rely on several parameters such as Data Flow Control, Error Control, Baud Rate Control.

Transmission Control and Reception Control. On the transmitter side, there is a shifting of parallel data onto the serial line using its own clock. Also it adds the start, stop and parity check bits. On the receiver side, the receiver extracts the data using its own clock and convert the serial data back to the parallel form after stripping off the start, stop, and parity bits. The well- known examples are RS-232 (https://circuitdigest.com/article/rs232-serial-communication-protocol-basics-specifications),
RS-422 and RS-485 (https://circuitdigest.com/tags/rs485).



Asynchronous - SCI

- Transmission of data through “words”
- Continuous transmission unnecessary
- Built-in safeguards against noise and error
- Transmitter and Receiver operate independently
- Requires start and stop bit for each byte of data
  - Sends constant ‘1’ for idle
  - Sends a ‘0’ for start and “1” for stop bits
- Very reliable data reception


DEVICE 1
RX
TX
GND
RX
TX
GND
DEVICE 2

Bit Types

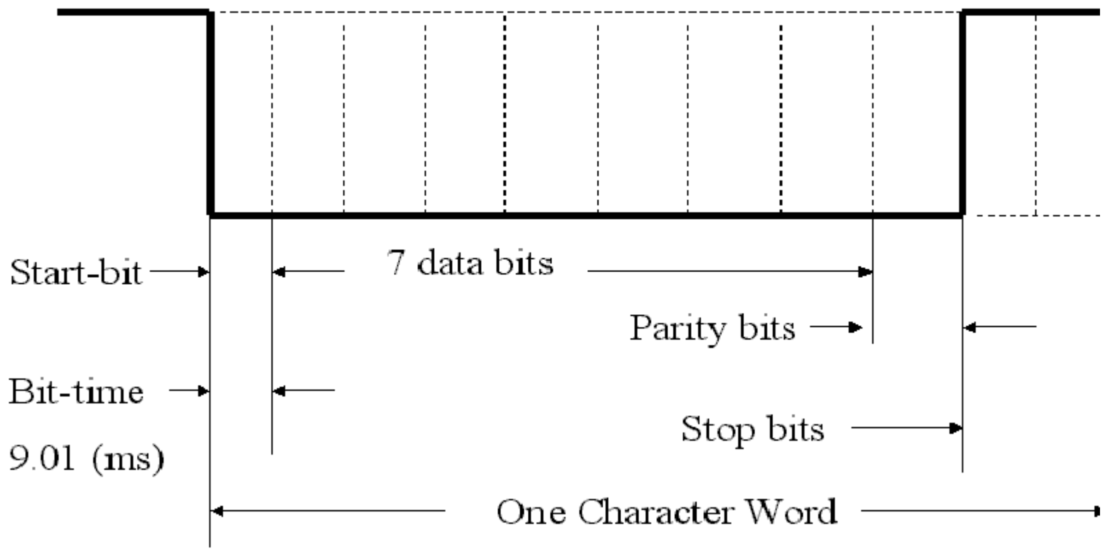


Figure 1. Role of stop, start and parity bits.

- **Start Bit –**
  - **Signals the transmission of a word.**
  - **Transition from "1" to "0". ("Mark-to-space")**
  - **First bit to be transmitted.**

Baud Rate Example

Baud rate > Bit rate

What is the bit rate for a 2400 baud rate using a parity bit and two stop bits per data word?

$$\text{bit rate} = \text{baud rate}\ \frac{8 \text{ data bits/word}}{12 \text{ total bits/word}}$$

$$\text{bit rate} = 2400(2/3) = 1600 \text{ bps}$$

$$\text{bit rate} = \text{baud rate}\ \frac{8 \text{ data bits/word}}{10 \text{ total bits/word}}$$

$$\text{bit rate} = 2400(4/5) = 1920 \text{ bps}$$

https://resources.altium.com/p/comparing-all-serial-communications-protocols

The Serial Communications Protocols Comparison

Mark Harris

| Created: August 17, 2021 | Updated: July 19, 2022

Protocol/ Standard	Max. Nodes	Max. Bit Rate [kbps]	Max. Length [m]	Advantages	Disadvantages	Notes
UART	2			1, Full-duplex 2. uses only two wires 3. Can provide both synchronous and asynchronous communication	1. Can connect only two devices 2. controller should adjust settings to the controlled device 3. Usually slower compared to I2C and SPI	1. Configurable baud rate 2. Can use only one wire if one-sided communication is required 3. Uses internal IC clock system 4. Basic IC to IC or device to device communication
I2C	127 or 1023	5000	Undefined	1. Uses only two wires 2. Multi-master and multi slave 3. More slaves do not require more pins 4. More reliable than UART	1. Slower compared to SPI 2. More complex hardware than SPI 3. Slaves should have addresses defined	1. Requires pull-up resistors 2. Basic IC to IC communication
SPI	Depends on SS pins	Up to 10000	Undefined	1. Full-duplex 2. very simple drivers 3. More reliable than UART 4. Very fast serial communication 5. Multi slave communication	1. Uses three or more wires (traces) 2. More slaves increase pin count 3. Only one master 4. Slaves cannot communicate freely with each other.	1. Basic IC to IC communication 2. Low-resolution display/image data
1-Wire	2^48	16.3	300	1. Requires only one wire 2. Parasitic power configuration 3. Very low power consumption 4. Very simple few pin devices	1. Only Maxim devices have this serial communication	1. Requires Pull-up
CAN	128	1000	500	1. Very robust 2. Multi slave 3. Error detection	1. Quite expensive	1. 120 Ω impedance transmission line 2. Uses differential pair 3. Used primarily on automotive electronics
LIN	16	19.2	40	1. Very cheap 2. Requires only one wire 3. Can have up to 15 slaves	1. Slow data rate	1. Used primarily on automotive electronics
RS-485	256	Up to 10000	1330	1. High achievable data rate 2. High achievable communication distance 3. High receiver sensitivity	1. Higher power consumption 2. Pretty complex hardware	1. Uses one or two differential pairs 2. Device to device communication
RS-232	2	128	15	1. Cheap	1. Slow data rate 2. Modern devices rarely have this connection or do not use this standard 3. Low receiver sensitivity	1. It can be seen often in many previous-generation devices 2. There are a lot of converters 3. Device to device communication

UART	USART
Standalone Communication Protocol	It can support multiple protocols like LIN, RS-485, IrDA, etc
Commonly used for low-speed applications	**More suitable for high-speed applications**
The data rate is relatively low	**The data rate is much higher**
The clock is generated locally and both devices are configured to run at the same baud rate	The clock signal is generated by the transmitter and sent to the receiver during data transmission
The baud rate of the receiver module must be known prior to any communication	The baud rate is useless as the receiver module will be using the clock coming from the transmitter device
Offer reduced low energy footprint even in sleep mode	Operates at high energy consumption modes

PIC24FJ128GB204 FAMILY

Three 3-Wire/4-Wire SPI modules:
- Support four Frame modes
- Variable FIFO buffer
- I2S mode
- Variable width from 2-bit to 32-bit

• **Two I2C™ modules Support Multi-Master**/ Slave mode and 7-Bit/10-Bit Addressing

• **Four UART modules**:
- **Support RS-485, RS-232 and LIN/J2602**
- On-chip hardware encoder/decoder for IrDA®
- Smart Card ISO 7816 support on UART1 and UART2 only:
- T = 0 protocol with automatic error handling
- T = 1 protocol
- Dedicated Guard Time Counter (GTC)
- Dedicated Waiting Time Counter (WTC)
- Auto-wake-up on Auto-Baud Detect (ABD)
- 4-level deep FIFO buffer

• Programmable 32-Bit **Cyclic Redundancy Check** (CRC) Generator