EEEN301 Embedded systems

Lecture 18 2022

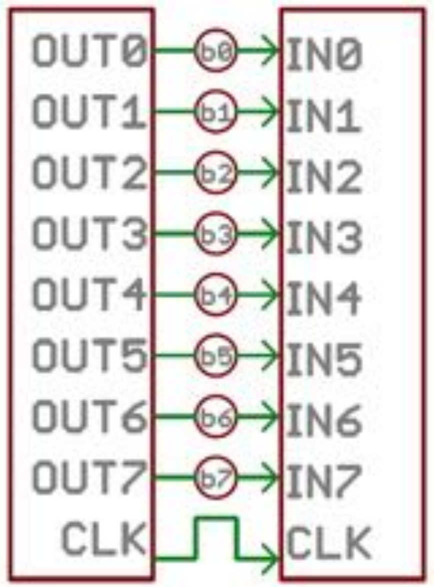
Interfaces

Serial vs. Parallel interfaces

Parallel interfaces transfer multiple bits at the same time. They usually require buses of data: transmitting across 8, 16, or more conductors.

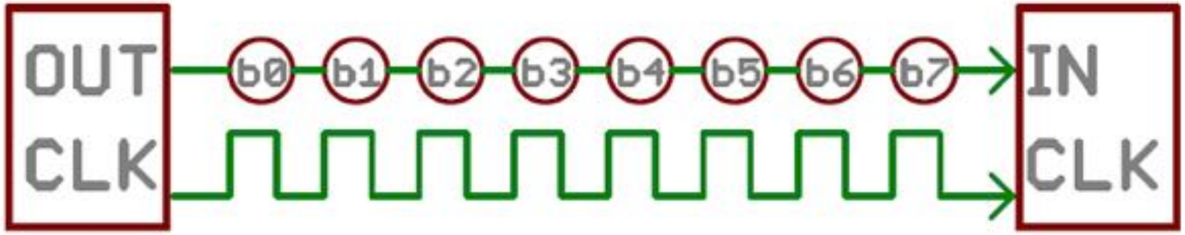
Example: printer port on old PCs (uses DB-25 connector) is a parallel connector with 8 data signals (and a number of other controls such as Select, Error, Reset, etc.).

Largely superseded by USB.



A simple clocked 8-bit parallel connection

Serial interfaces stream data one single bit per clock cycle.



Some common serial connectors:

- PS/2: mini-DIN connecter used to connect keyboard and mouse to PCs with one data pin and one clock pin, plus power and ground.
- USB: Differential data signals transmitting on a twisted pair, plus ground and power.
- MIDI: DIN-5 cable with only 2 pins in use to transfer the signal, plus one ground pin.

PARALLEL: Transmission of multiple bits of data at the same time generally means faster communication.

However, in long haul transmissions, crosstalk and skew issues become significant and can require mitigation schemes.

Crosstalk, skew and the need for sync lead to slower speeds.

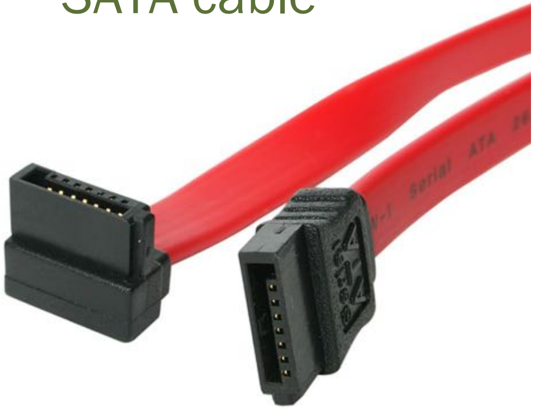
SERIAL: Data transmission at the rate of one bit at a time should generally mean slower transmission.

However, in practice, lack of crosstalk and skew between conductors gives an advantage to serial transmission in terms of speed.

Now, even in high performance inter-IC communications, SATA (Serial AT Attachment) is used to connect host bus adapters to mass storage devices. It replaces PATA (Parallel ATA). Just like the PCIe bus replaces the parallel ISA bus.

PATA cable

SATA cable

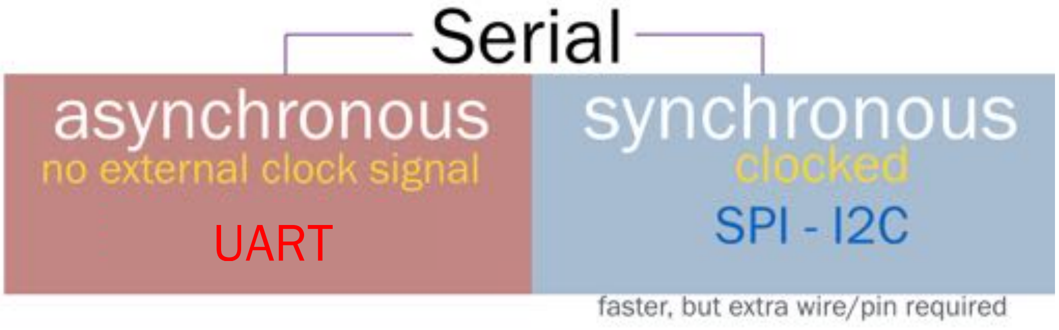


PATA is relatively slow: up to 133Mbit/s. The synchronisation of the parallel signals becomes a significant limiting factor in increasing the signaling rate for PATA.

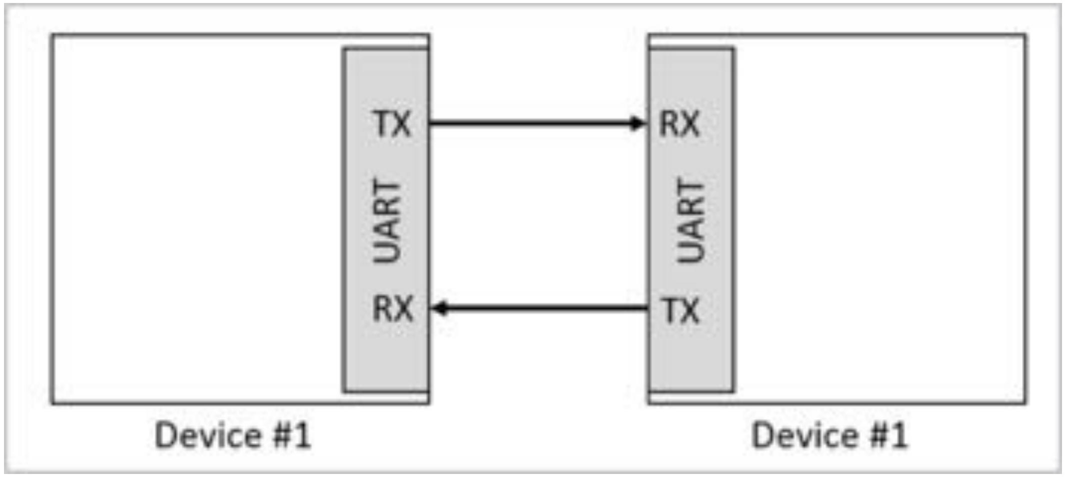
Higher signaling rates of SATA (up to many Gbit/s) and the reduced cable size make it faster, more efficient, and more cost-effective.

Serial	Parallel
Transmission at a rate of one bit at a time (one conductor).	Transmission of multiple bits at the same time (multiple conductors).
Much cheaper cabling (due to less material). Significantly cheaper in long-range communications.	More expensive in terms of cabling (more material). Cable cost becomes significant in long haul transmissions.
Conversion required: serial to parallel and back to serial (extra hardware cost).	Straightforward implementation (fewer hardware overheads).
No crosstalk or skew (only one data signal at a time) leads to faster long haul transmissions.	Signal interference, crosstalk and skew issues in long ranges (caused by multiple adjacent cables).

Typical usage:
Parallel: Inter-chip communication on a circuit board.
Serial: Off-board communication.



Asynchronous serial protocols:



Asynchronous Serial Communications Example

Asynchronous protocols form the basis for many legacy communications protocols (RS-232, etc). Devices employing an asynchronous serial protocol use a UART (Universal Asynchronous Receiver/Transmitter) to communicate.

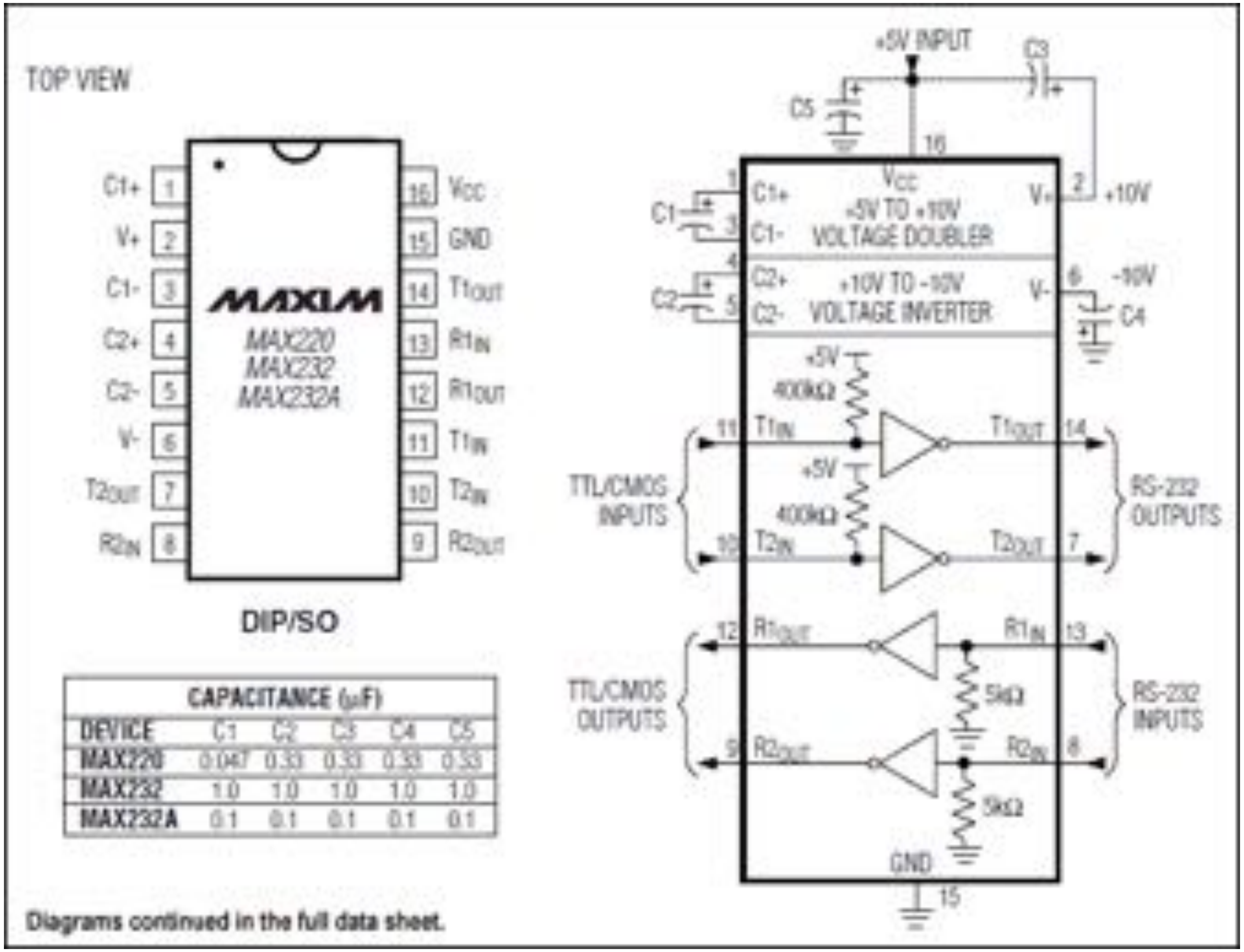
- Data is output at a fixed baud rate which must be agreed upon beforehand by sender and receiver.
- “Start bits” are also added to signal the receiver when a transmission has started.
- The UART typically oversamples incoming data to ensure that all is received.
- UART can be implemented in hardware or software.
- Many microcontrollers include one or more hardware UARTs
- Asynchronous protocols use a variety of voltages:

The legacy RS-232 protocol represents a logic HIGH with a negative voltage (from -3 to -25 V) and a LOW with a positive voltage from 3 V to 25 V.

RS-232 connector

Most microcontrollers’ UARTs work from 0 V for LOW and +5 V (or +3.3 V) for HIGH. Level shifter circuitry helps to address this.

The Maxim IC, MAX232 is a popular monolithic solution, making use of a voltage doubler and inverter, allowing the RS-232 protocol to be interfaced with common +5 V DC power supplies.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Diagrams continued in the full data sheet.

The Maxim MAX232

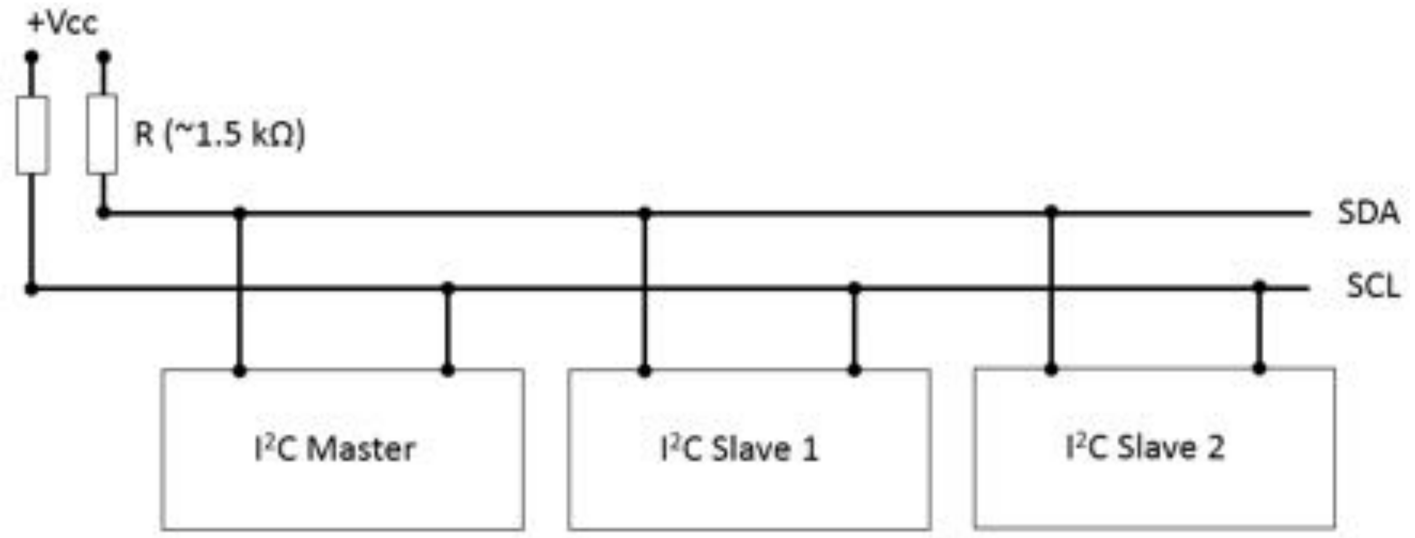
Synchronous serial protocols

I2C and SPI are relatively simple synchronous serial protocols where a clock signal is sent on a separate line in addition to one or more data lines. (Embedded clocking is the alternative and is used in more sophisticated protocols such as Ethernet and PCIe).

Advantages compared to asynchronous serial:

- Generally simpler hardware. Don’t need to oversample.
- Often a higher data rate (due to no over-sampling).
- less dead time and greater data density per clock. UART requires start bits for every byte and a delay between bytes (often called ”stop bits”).

I²C



An example I²C bus with two slave devices

I2C (often called 2-Wire) is a very popular way of connecting peripherals to microcontrollers.
It is a “multidrop network” (like a part line) where multiple devices are connected to shared lines which are pulled up using resistors.
In high speed mode it can transmit at up to 3.4 Mbps.

Used on many devices: Accelerometers, pressure sensors, motor controllers, ADCs, RAM, etc.

- I2C allows for a large number of devices to be connected using few microcontroller pins.
- I2C devices all share a bus and are each assigned a unique address (either at the factory or by the developer). The bus has two wires: SDA (Serial Data Line) and SCL (Serial Clock Line).
- I2C is half-duplex.

I^{2}C EEPROM (24LC256)

I^{2}C DAC (12-bit, up to 3.4 Mbps)

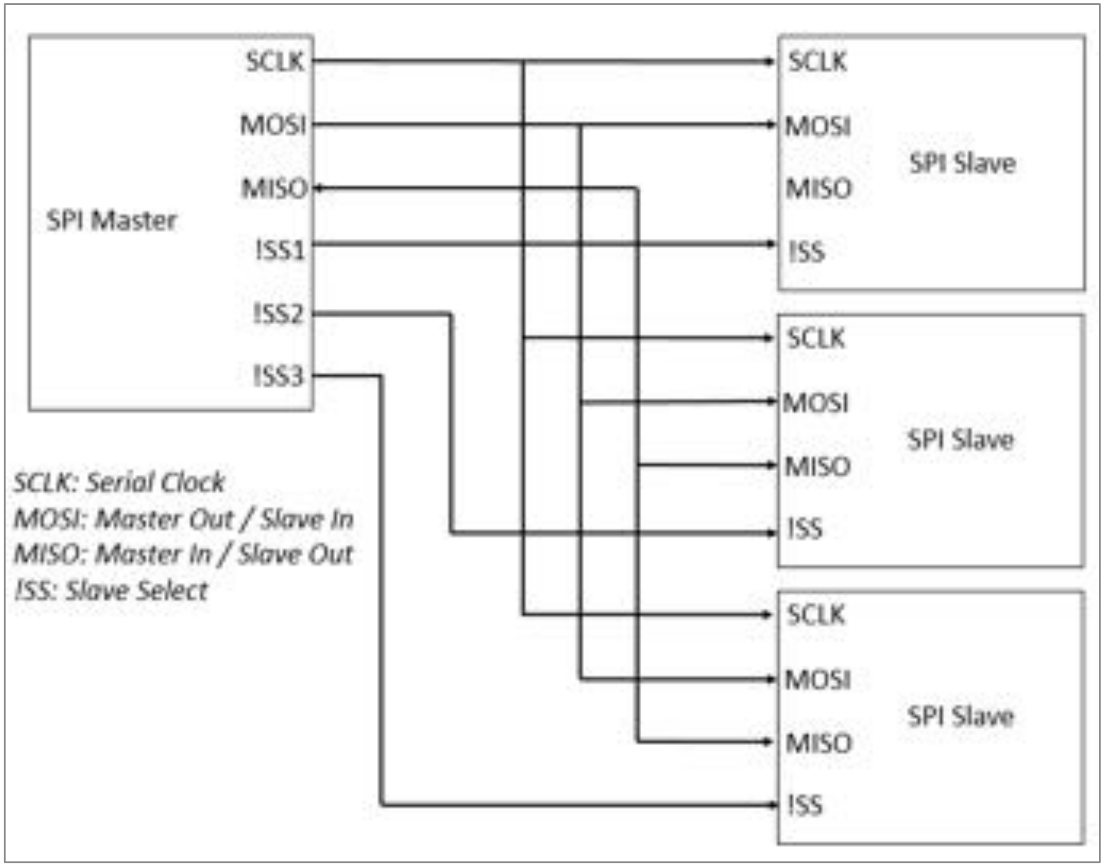
LIDAR-Lite (miniature I^{2}C laser rangefinder)

I^{2}C pressure sensor (BMP180)

Some devices that use I^{2}C

SPI (Serial Peripheral Interface) is a full-duplex protocol that is popular for its simplicity and high data rate (Upwards of 10 Mbps, typically 50Mbps).

Implementation is relatively simple in either hardware or software, requiring only shift registers, buffers, and master-generated clock and slave select signals.



SPI ADC (MCP3008)

SPI FRAM (64 Kb Fujitsu MB85RS64V)

SPI Real Time Clock (DS3234)

SPI Wireless Transceiver (RFM12B-S2, 915 MHz)

Some devices that use SPI.

SD cards and some ICs use a multibit version of SPI.

SPI vs I^2C

	SPI	I^2C
Data Rate	>10 Mbps	3.3 Mbps in high speed mode
Wiring	Shared MOSI & MISO. Each device requires a slave select line. (Can add up to lots of connections!)	Two wires (SDA & SCL)
Hardware complexity	Simple (easy to implement in software if no hardware SPI interface is available)	Relatively complicated
Configuration	Simple in hardware or software.	More complicated.

Many devices are now available in either SPI or I2C format (some can even support both I2C and SPI).

ADXL345 Accelerometer (I^2C and SPI)

Embedded clocking

I2C, SPI or any other serial protocol that uses a separate clock line can suffer from cross talk and skew so this does potentially limit the maximum length of the wires.

Single wire communication protocols overcome this issue by embedding the clock within the data signal. The receiver unit is able to synchronize it’s own clock so we achieve synchronous communication.

One of the key methods to embed the clock signal is known as 8b/10b Encoding.

8b/10b Encoding was developed in 1983 by A.X. Widmer and P.A. Franaszek from IBM.

The method encodes an 8-bit data byte (256 unique data words) into a 10-bit symbol, hence the 8b/10b designation. An additional 12 special (or K) characters are also included for management functions.

The characteristics of the code scheme make it ideally suited for high-speed local area networks, computer links, or any serial data link. The code scheme is DC-balanced, which is of particular benefit for active gain, threshold setting and equalization of optical receivers.

The code insures a limited run length, no more than 5 consecutive ones or zeros, and a guaranteed transition density, which permits clock recovery from the data stream. The special (K) characters are useful as packet delimiters.

Because of its many features, this method has been used in the physical layer (PHY) of a number of current and emerging standards, including Fibre Channel, Gigabit Ethernet, Rapid I/O, Thunderbolt, HDMI, USB3, and PCIe.

Examples of high speed synchronous serial protocols:

GigaBit Ethernet: Ethernet originally started as a multidrop network using coaxial cable. Now multiple twisted pairs are used for Tx and Rx. Also a dedicated cable between each device and router.

HDMI: Video transmission method. 3 serial lanes for RGB data and a separate lower frequency pixel clock.

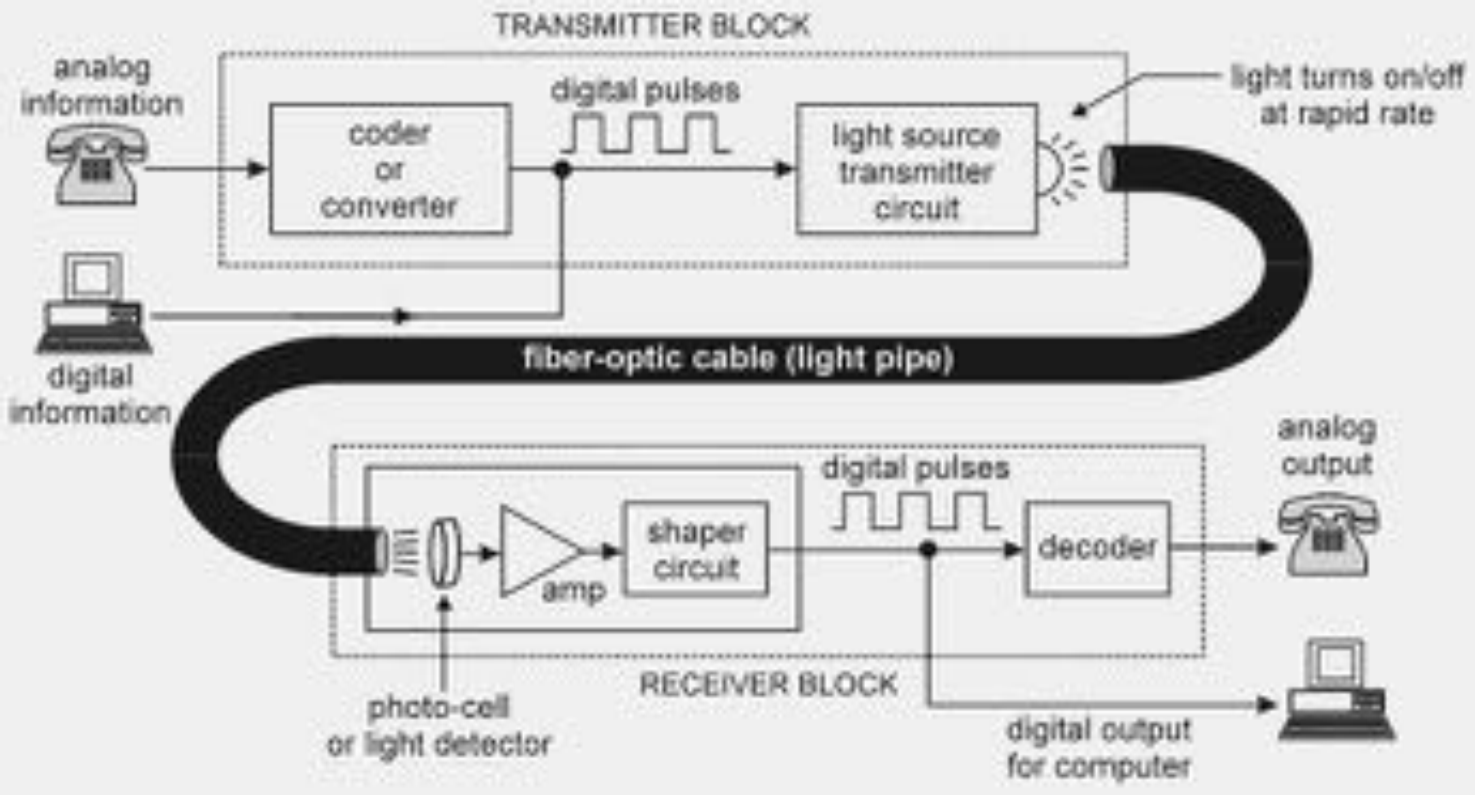
USB3.0: Two “superspeed” differential pairs for Rx/Tx and one lower speed differential pair to implement half-duplex USB2.0.

PCIe: Each lane is made up of two differential pairs for Rx/Tx. PCIe can have 1 to 16 lanes and for version 5 the bandwidth is 32Gbits/s per lane. Multiple lanes provides the advantage of a parallel interface but here each lane works independently. In the transmitter the data is divided up across the lanes. It is later recombined in the receiver.

32Gbits/s per lane and 16 lanes provides a total bandwidth of 63.015 GBytes/s !!

Fibre-Optics

For situations requiring long range and extreme resilience in a high-noise environment, fibre- optic schemes are the very popular solution.



Fibre-Optics transmission

Replaces electrical transmission for higher bandwidth and exceptionally longer distance communications.

- Optical Transmitter: Converts serial bit stream (with embedded clock) electrical signal into optical signal. Laser diode and high speed current driver.

- Cable: optical core, cladding and protective coating

- Optical Receiver: Converts the optical signal back to electrical signal. Usually a high speed photodiode and amplifier.

- Transmission rate: >100 Gbps over 10’s of kilometres.

- For long distances, an amplifier called a “repeater” can be used at midway points to compensate for signal attenuation and distortion.