Topic: 1.2.1 Data transmission

TRANSMISSION OF DATA

Data transmission refers to the movement of data in the form of bits between two or more digital devices. This transfer of data takes place via some form of transmission media (for example, coaxial cable, fiber optics etc.)

Because ordinary telephone circuits pass signals that fall within the frequency range of voice communication (about 300–3,500 hertz), the high frequencies associated with data transmission suffer a loss of amplitude and transmission speed. Data signals must therefore be translated into a format compatible with the signals used in telephone lines. Digital computers use a modem to transform outgoing digital electronic data; a similar system at the receiving end translates the incoming signal back to the original electronic data. Specialized data-transmission links carry signals at frequencies higher than those used by the public telephone network.

Types of Data Transmission

- **Serial transmission of data**
- **Parallel transmission of data**

Topic: 1.2.1 Data transmission

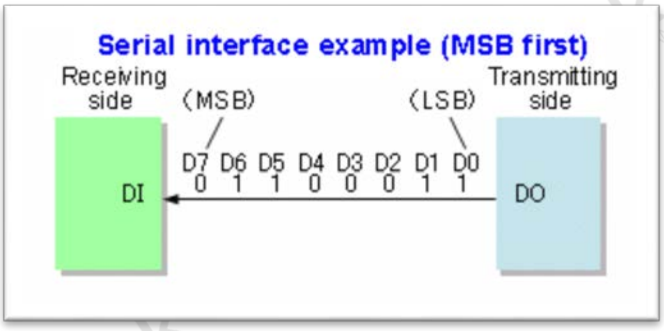
USES OF SERIAL AND PARALLEL DATA TRANSMISSION:

In digital communications systems, there are 2 methods for data transfer: parallel and serial. Parallel communication uses multiple wires running parallel to each other, and can transmit data on all the wires simultaneously. Serial communication on the other hand, uses a single wire to transfer the data bits one at a time.

SERIAL TRANSMISSION

Serial transmission occurs over a single cable, **one bit at a time**. This type of communication is named "serial" not simply because data travels one bit at a time, but also because these bits must be organized in a particular way so that transmission can be organized and considered trustworthy. For example, a single transmission from a peripheral device using serial data might take only 8 bits, so the serial mechanism has a way to dictate how to signal things like start of a new transmission of data.

Data is transmitted in the form of bits down the wire, so an 8 bit byte, which stands for a single character, will be transmitted in 8 parts, one signal for each bit.



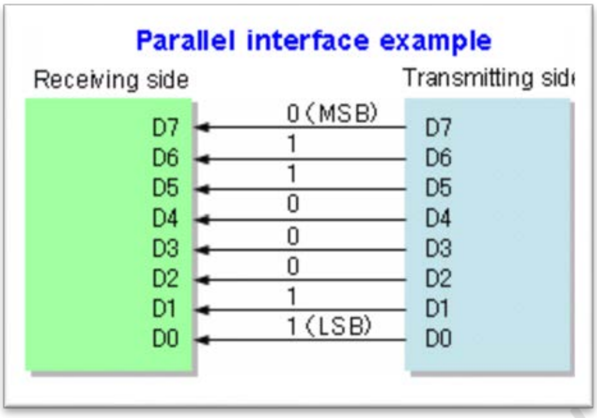
The diagram shows the data byte 01100011 being transmitted. As there is only one wire, only one bit can be transmitted at a time.

Serial transmission is simple and reliable because the next bit is not transmitted until the current one has arrived at its destination. However, because only one bit can be transmitted at a time, the speed of transmission is slow.

Topic: 1.2.1 Data transmission

PARALLEL TRANSMISSION:

Let us assume that the devices are linked by more than one wire. This means that more bits can be sent at once. A sensible number of wires would be 8, because then a whole byte could be sent at once instead of having to send one bit at a time.



This type of data transfer is called "Parallel" transmission. Parallel transmission of data is obviously faster than serial because all the bits are travelling at the same time, but because of the fine tolerances in the transmission, it is less reliable as the bits can become mixed-up.

Topic: 1.2.1 Data transmission

SERIAL TRANSMISSION VERSUS PARALLEL TRANSMISSION

Serial transmission is *slower* than parallel transmission given the *same signal frequency*. With a parallel transmission you can transfer one word per cycle (e.g. 1 byte = 8 bits) but with a serial transmission only a fraction of it i.e. 1 bit.

The reason modern devices use serial transmission is because:

- You cannot increase the signal frequency for a parallel transmission without limit, because, by design, all signals from the transmitter need to arrive at the receiver at *the same time*. This cannot be guaranteed for high frequencies, as you cannot guarantee that the **signal transit time** is equal for all signal lines. The higher the frequency, the more tiny differences matter. Hence the receiver has to wait until all signal lines are settled thus lowering the transfer rate.

- One needs to consider **crosstalk** with parallel signal lines. The higher the frequency, the more pronounced crosstalk gets and with it, the higher the probability of a corrupted word and the need to retransmit it.

Parallel transmission; however has its own merits:

- Multiple bits of information are sent at the same time.
- At identical clock speeds, parallel transfers are faster than serial transfers because more data is being transferred.

However, parallel transfers also have problems:

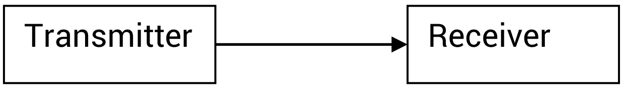
- Many wires or traces (wire-like connections on the motherboard or expansion cards) are needed, leading to interference concerns and thick, expensive cables.
- Excessively long parallel cables or traces can cause data to arrive at different times. This is referred to as “signal skew". Parallel cables that are too long can cause signal skew, allowing the parallel signals to become "out of step" with each other.
- Differences in voltage between wires or traces can cause jitter.

Topic: 1.2.1 Data transmission

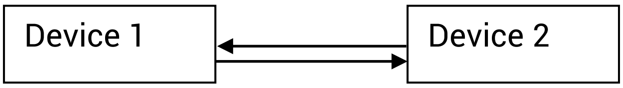
Modes of Transmission

If data is to be transmitted between devices, there are three different modes of transmission possible.

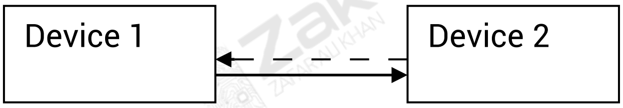
a) Simplex mode. If data can only travel in one direction then it is known as a SIMPLEX transmission. A good example of a simplex transmission of data is teletext information which is passed to a television receiver, but there is no way to send data in the other direction.



b) Duplex mode. When data can pass in both directions at the same time it is known as DUPLEX transmission of data. An example of duplex transmission would be a telephone conversation as both users can speak and be heard at the same time.



c) Half duplex mode. When data can pass in both directions, but only in one direction at a time, the transmission is called HALF DUPLEX. An example of half duplex is a CB radio system in which each handset can either be set to receive mode or send mode.



Topic: 1.2.1 Data transmission

ERROR DETECTION AND CORRECTION IN DATA TRANSFER:

In networking, error detection refers to the techniques used to detect noise or other impairments introduced into data while it is transmitted from source to destination. Error detection ensures reliable delivery of data across vulnerable networks.

Error detection minimizes the probability of passing incorrect frames to the destination, known as undetected error probability.

The amount of additional information sent is usually the same for a given amount of data, and the error detection information will have a relationship to the data that is determined by the application of an algorithm of some kind to the data itself.

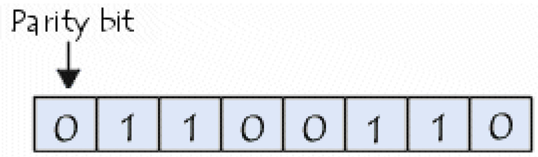
The receiver applies the same algorithm to the data it receives to obtain its own version of the error correction code, and then compares that version with the error correction code it has received. If the two codes match, the receiver can be reasonably sure that the data is correct. If not, it will assume that an error has occurred and respond in the appropriate manner.

PARITY BITS AS A METHOD FOR ERROR DETECTION:

The movement of digital data from one location to another can result in transmission errors, the receiver not receiving the same signal as transmitted by the transmitter as a result of electrical noise in the transmission process. Sometimes a noise pulse may be large enough to alter the logic level of the signal. For example, the transmitted sequence 1001 may be incorrectly received as 1101. In order to detect such errors, a parity bit is often used. A parity bit is an extra "0" or "1" bit attached to a code group at transmission. In the even parity method, the value of the bit is chosen so that the total number of 1s in the code group, including the parity bit, is an even number.

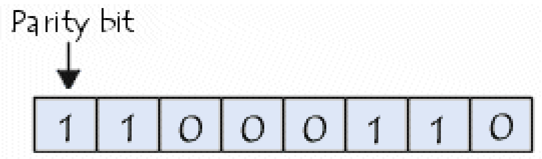
For example, in transmitting 1001 the parity bit used would be 0 to give 01001, and thus an even number of 1s. In transmitting 1101 the parity bit used would be 1 to give 11101, and thus an even number of 1s. With odd parity, the parity bit is chosen so that the total number of 1s, including the parity bit, is odd. Thus if at the receiving end, the number of 1s in a code group does not give the required parity, the receiver will know that there is an error and can request that the code group be retransmitted.

Consider the following example:

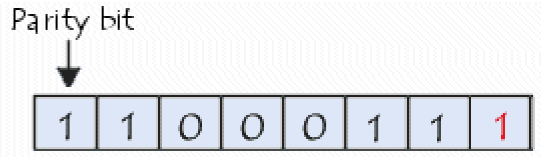


Topic: 1.2.1 Data transmission

In this example, the number of *1* data bits is even, so the parity bit is set to *0*. By contrast, in the example below, the data bits are odd, so the parity bit becomes 1:

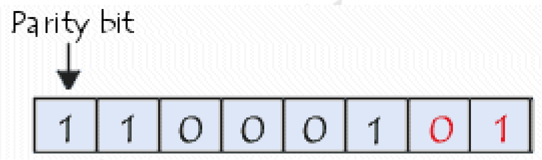


Let's pretend that after being transmitted, the lowest-weighted bit of the previous byte (the one on the far right) had fallen victim to interference:



The parity bit, in this case, no longer corresponds to the byte's parity: **an error has been detected**.

However, if two bits (or an even number of bits) had simultaneously changed as the signal was being sent, no error would have been detected.



As the parity control system can only detect an odd number of errors, it can only detect 50% of all errors. This error-detection mechanism also has the major downside of being unable to correct the errors it finds (the only way to fix it is to request that the erroneous byte be retransmitted)

Topic: 1.2.1 Data transmission

INTEGRATED CIRCUITS:

In the beginning all data sent to and from disks traveled in serial form—one bit was sent right after another, using just a single channel or wire. With advancement in IC technology, it became feasible and cheap to put multiple devices on a single piece of silicon, and the parallel interface was born.

The decreasing cost of integrated circuits, combined with greater consumer demand for speed and cable length, has led to parallel communication links becoming deprecated in favor of serial links. Typically, it used eight channels for transmission, allowing eight bits (one byte) to be sent simultaneously, which was faster than straight serial connections. The standard parallel interface used a bulky and expensive 36-wire cable.

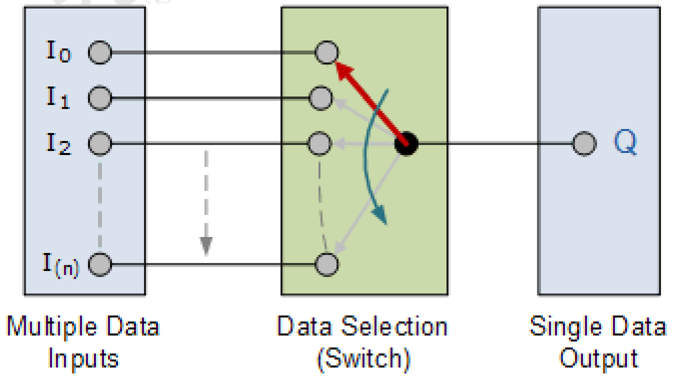
Serial or parallel data transmission is used in many Integrated circuits today. Multiplexer is one such example of an IC where both serial and parallel data transmission is used.

MULTIPLEXERS:

The *multiplexer* or "MUX" is a combinational logic circuit designed to switch one of several input lines through to a single common output line by the application of a control signal. Multiplexers operate like very fast acting multiple position rotary switches connecting or controlling multiple input lines called "channels" one at a time to the output.

Multiplexers, or MUX's, can be either digital circuits made from high speed logic gates used to switch digital or binary data or they can be analogue types using transistors, MOSFET's or relays to switch one of the voltage or current inputs through to a single output.

The most basic type of multiplexer device is that of a one-way rotary switch as shown.



As you can see, initially inputs are provided via parallel transmission technique but the output is given out serially.

Topic: 1.2.1 Data transmission

Universal Serial Bus (USB):

The USB, which stands for “Universal Serial Bus”, is a high-speed serial standard that, along with FireWire, has almost totally replaced the ordinary serial and parallel standards that have been used to connect peripheral devices to PCs since the 1980s.

In theory, up to 127 devices can be connected to a single USB port on a computer's motherboard, using only one of a computer's 16 or 23 Interrupt Requests (IRQs) by using a USB hub, which itself has a number of USB ports.

A USB hub can have its own power source, or draw its power through the computer to which it is attached. To avoid power problems, a powered hub is the best choice. Seven peripherals can be attached to each USB hub. One of these peripherals can be a second hub to which up to another seven peripherals can be connected, and the second hub can be attached to a third hub, etc.