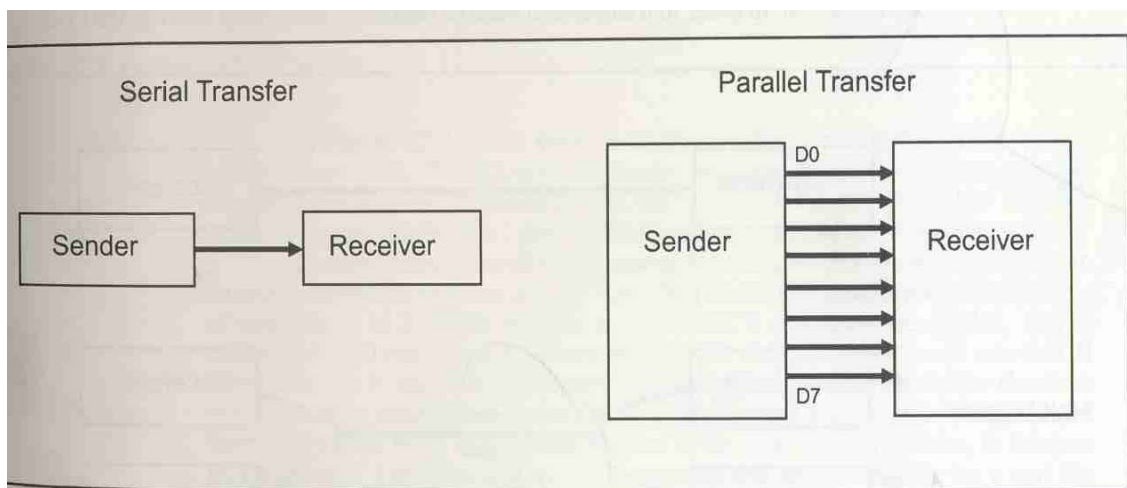


Serial Communication Interface

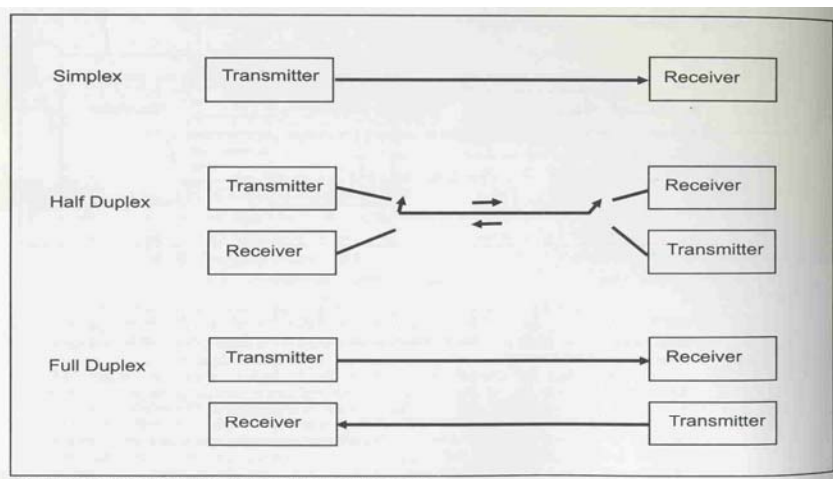
CPU based systems transfer data in two ways: parallel and serial.

In parallel data transfers, often 8 or more lines (wire conductors) are used to transfer data to a device that is only a few feet away. Examples of parallel transfer are printers and hard disks using cables with many wire strips. Although in such cases a lot of data can be transferred in a short amount of time by using many wires in parallel, the distance cannot be great. To transfer to a device located many meters away, the serial method is used. In serial communication, the data is sent one bit at a time, in contrast to parallel communication in which the data is sent a byte or more at a time.



Half-and full-duplex transmission

In data transmission if the data can be transmitted and received, it is a duplex transmission. This is in contrast to **simplex** transmissions such as printers, in which the computer only sends data. Duplex transmissions can be half or full duplex, depending on whether or not the data transfer can be simultaneous. If data is transmitted one way at a time, it is referred to as **half duplex**. If the data can go both ways at the same time, it is **full duplex**. Of course, full duplex requires two wire conductors for the data lines (in addition to ground), one for transmission and one for reception, in order to transfer and receive data simultaneously. See Figure



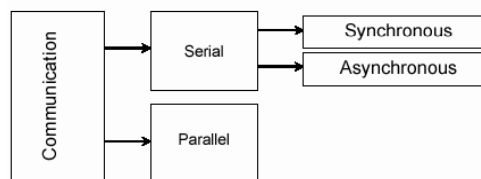
The data coming in at the receiving end of the data line in a serial data transfer is all 0s and 1s; it is difficult to make sense of the data unless the sender and receiving agree on a set of rules, a protocol, on how the data is packed, how many bits constitute a character, and when the data begins and ends.

For serial data communication to work, the byte of data must be grabbed from the 8-bit data bus of the microprocessor and converted to serial bits using a parallel-in-serial-out shift register; then it can be transmitted over a single data line. This also means that at the receiving end there must be a serial-in-parallel-out shift register to receive the serial data and pack them into a byte and present it to the system at the receiving end.

Serial data communication uses two method,

Asynchronous : This method transfers a single byte at time

Synchronous : This method transfers a block of data (characters) at a time (common clock pulse)



It is possible to write software to use either of these methods, **but the programs can be tedious and long. For this reason, there are special IC chips** made by many manufactures for serial data communications. These chips are commonly referred to as **UART (universal asynchronous receiver-transmitter)** and **USART (universal synchronous receiver-transmitter)**.

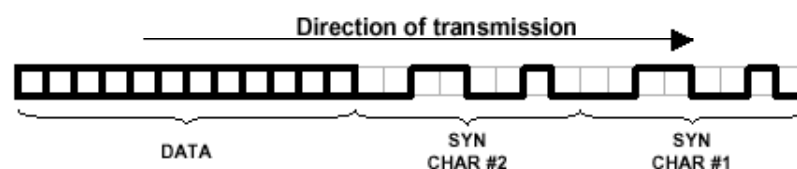
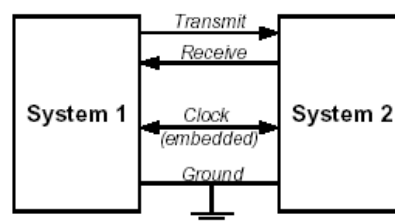
In synchronous communication, the receiver and transmitter sections must run synchronously. As shown in fig, the interface must include a [clock line](#). The format used for synchronous communication of data is shown in figure.

- 1- The transmitter first sends out synchronization characters to receiver
- 2- The receiver reads the synchronization bit pattern and compares it to a known sync pattern
- 3- The receiver begins to read character data of the data line

Example:

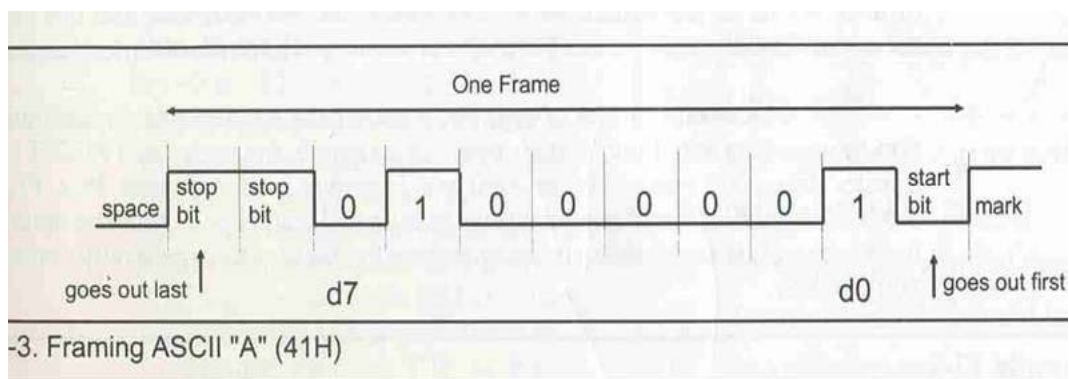
I²C

SPI



Asynchronous serial ;Start and stop bits

Asynchronous serial data communication is widely used for **character-oriented** transmissions, and block-oriented data transfers use the synchronous method. In the **asynchronous method, each character is put in between start and stop bits**. This is called framing. In data framing for asynchronous communications, the data, such as ASCII characters, are packed in between a start bit and a stop bit. The start bit is always one bit but the stop bit can be one or two bits. The start bit is always a 0 (low) and the stop bit(s) is 1 (high). For example, look at Figure where the ASCII character "A", binary 0100 0001, is framed in between the start bit and 2 stop bits. Notice that the LSB is out first. When there is no transfer the signal is 1 (high), which is referred to as mark. The 0 (low) is referred to as space. Notice that the transmission begins with a start bit followed by D0, the LSB, then the rest of the bits until the MSB (D7), and finally, the 2 stop bits indicating the end of the character "A".



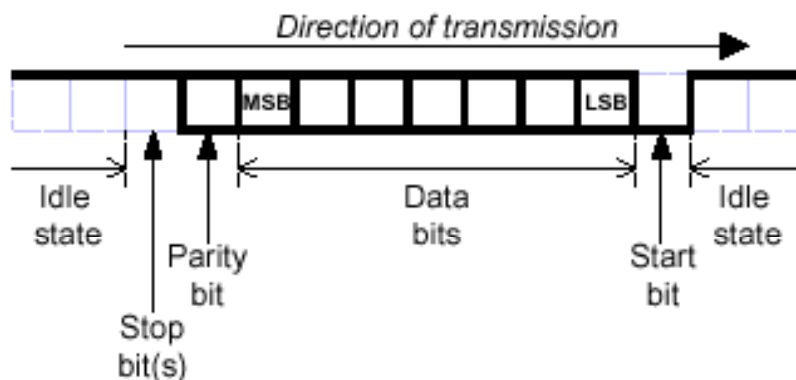
The format of a typical asynchronous character is the following ;

Start bit (1 bit) : a logical 0 (also called a space) that is used to indicate the beginning of a character

Data bits (7- 8 bits) : they contain the information that is being transmitted. The data wordlength can be either 7 or 8 bits

Parity bit (1- 0 bits) : optional, used for error checking. If odd parity is used, the parity bit is set to 1 or 0 such that the total number of 1- bits, including the data bits and the parity bit, is odd.

Stop bit (1- 2 bits) : a logical 1 (also called a mark) that is used to indicate the end of the transmission



Example 1:

Calculate the total number of bits used in transferring **5 pages**, each with 80x25 characters. **Assume 8 bits per character and 1 stop bit.** **Solution:**

For each character, a total of 10 bits is used, 8 bits for each character, **1 stop bit, and 1 start bit**. There for, the total number of bits is **$80 \times 25 \times 10 = 20,000$** bits per. For 5 pages, 100,000 bits will be transferred.

Data transfer rate

The rate of data transfer in serial data communication is stated in bps (bits per second). Another widely used terminology for bps is baud rate. However, the **baud and bps rates are not necessarily equal**. This is due to the fact that baud rate is the modem terminology and is defined as number of signal changes per second. In modems, there are occasions when a single change of signal transfers several bits of data. As far as the conductor wire is concerned, the baud rate and bps are the same,

Bits per second (bps)

Measures the rate at which information flows across a data link. Bit rate reveals the actual number of data bits transferred in a given period of time

Example 2:

Calculate the time it takes to transfer the entire 5 pages of data in last example: for 2400 bps:

$$100,000/2400 = 41.67 \text{ seconds}$$

Programming the 8251 (Programmable communication interface controller)

Control register of 8251:

7	6	5	4	3	2	1	0
Number of Stop bits		Parity	Parity Enable	Character Length		Baud rate	
00-invalid 01-1 bit 10-invalid 11-2 bits		0-odd 1-even	0-disable 1-enable	00-5 bits 01-6 bits 10-7 bits 11-8 bits		00-syn mode 01-x 1 clock 10-x 16 clock 11- x 64 clock	

Setting the serial port parameters, When setting up a serial port, several parameters must be specified. The most common are:

1- Bit rate selection

2- Character length:The number of data bits/ character, usually five to eight.

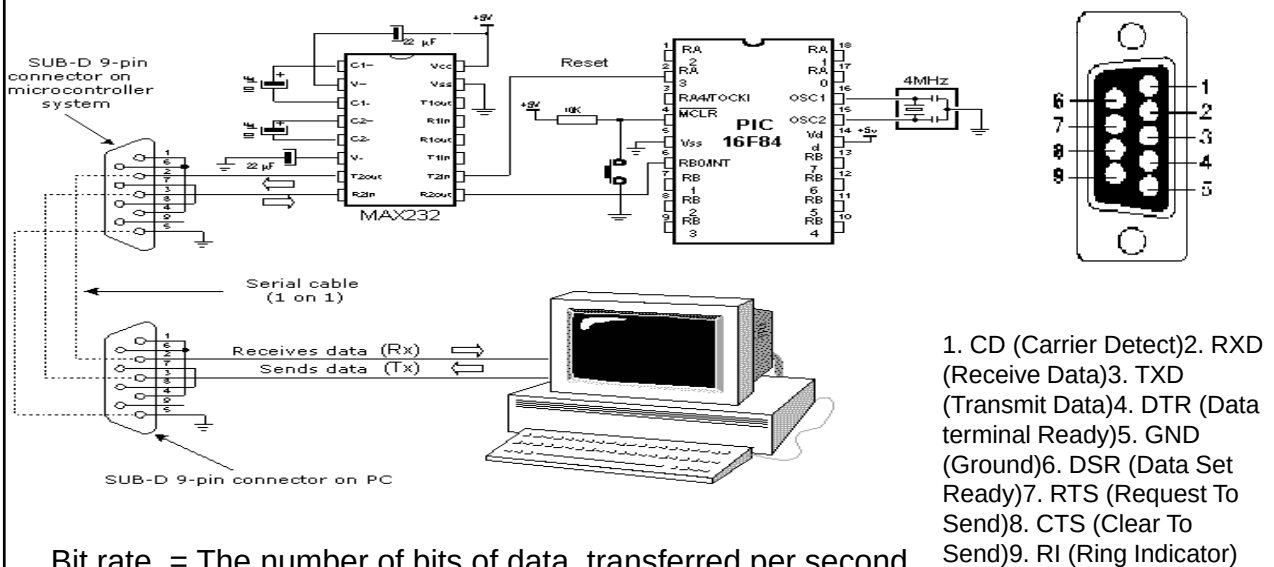
3-Parity selection and generation (odd, even, or no parity)

4- Number of stop bits (usually one).

A simple connection for interface of microcontroller and PC

MAX 232 is used for voltage level converter: (RS232 to TTL logic converter)

In RS 232 data communication: Logic 1 level is -3 to -25 and Logic 0 voltage level is +3 to +25



Bit rate = The number of bits of data transferred per second,

9600 bit rate

Example: The total time is observed for data transfer in a. Communication is 0.833 ms what is the baud rate? Solution = $1/0.833 = 1200$ bps