Serial Communication
(Chapter 10)

MBSD, 6th Semester

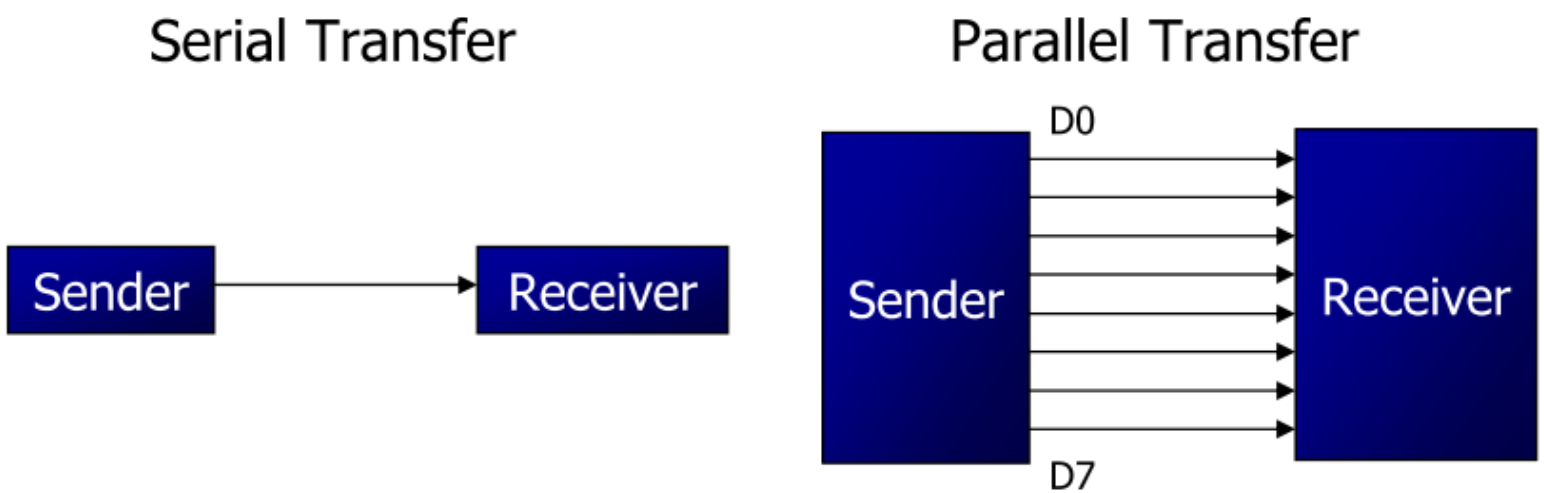
DCSE, UET Peshawar

Bilal Habib

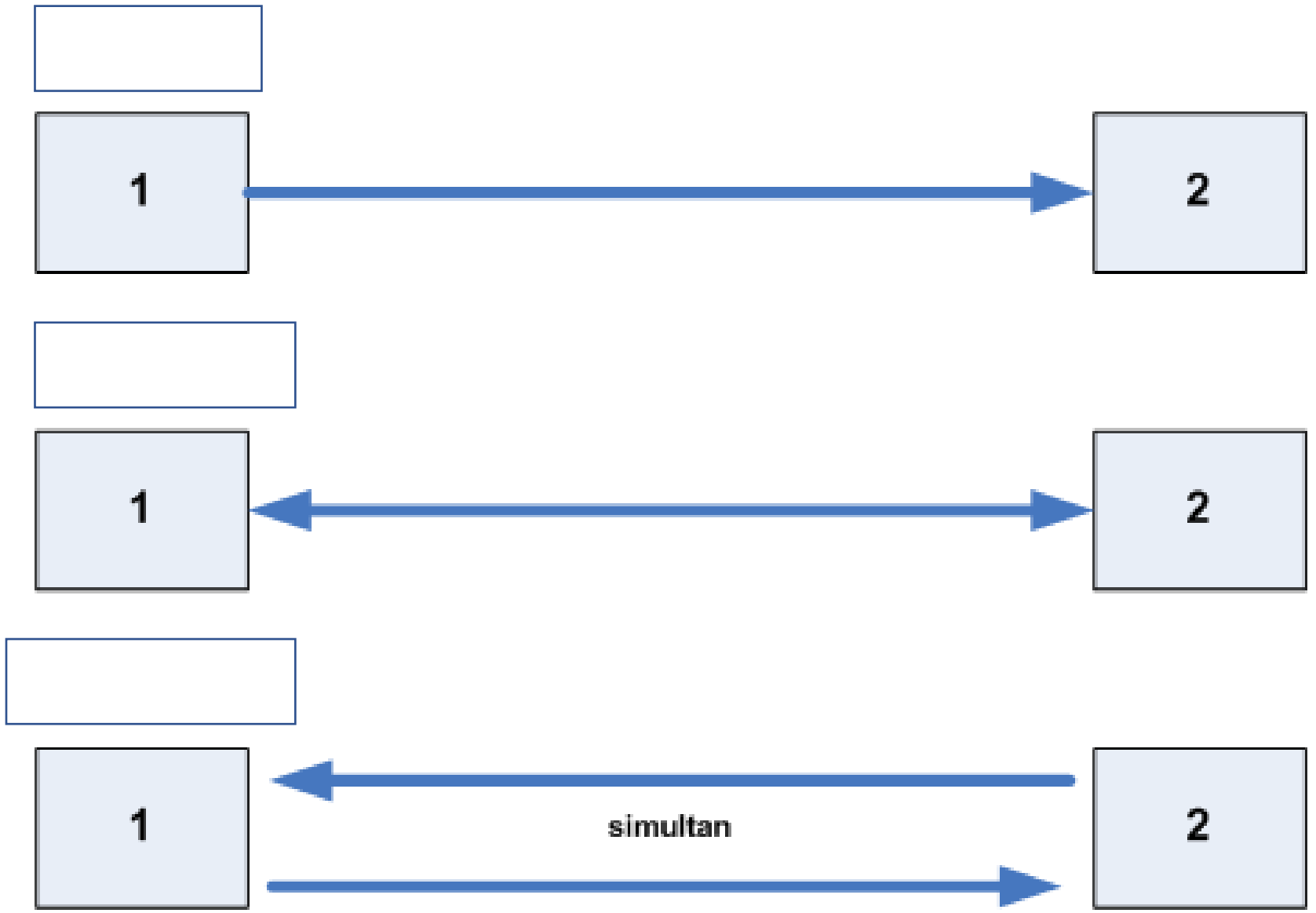
BASICS OF SERIAL COMMUNICA-TION

Computers transfer data in two ways:

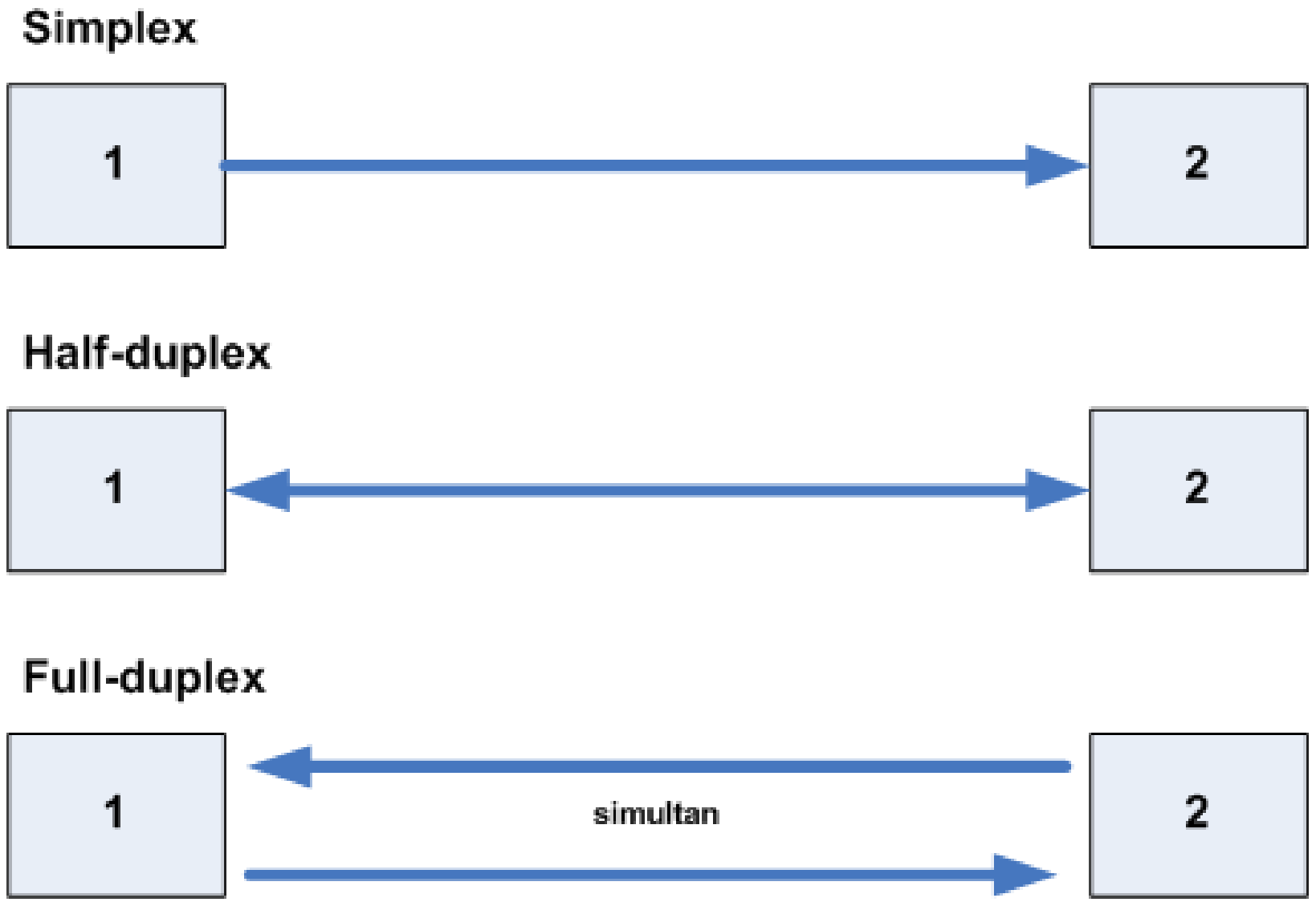
- Parallel
  - Often 8 or more lines (wire conductors) are used to transfer data to a device that is only a few feet away
- Serial
  - To transfer to a device located many meters away, the serial method is used
  - The data is sent one bit at a time



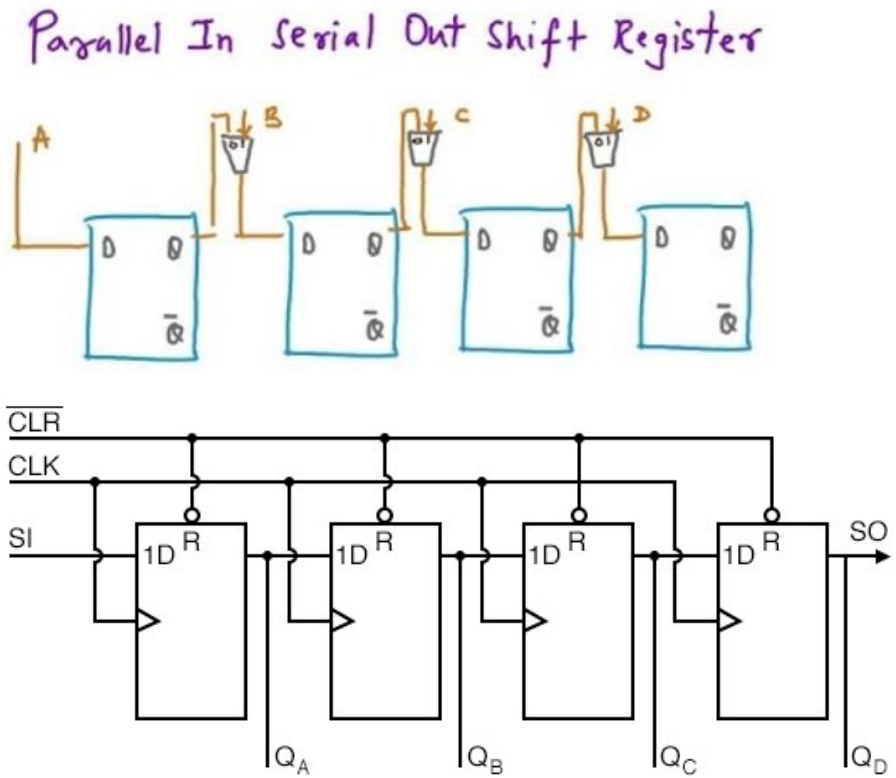
Transmission Modes



Transmission Modes



- At the transmitting end, the byte of data must be converted to serial bits using parallel-in-serial-out shift register
- At the receiving end, there is a serial-in-parallel-out shift register to receive the serial data and pack them into byte
- When the distance is short, the digital signal can be transferred as it is on a simple wire and requires no modulation
- If data is to be transferred on the telephone line, it must be converted from 0s and 1s to audio tones
  - This conversion is performed by a device called a *modem*, "Modulator/demodulator"



Serial-in/ Parallel-out shift register details

BASICS OF SERIAL COMMUNICA-TION

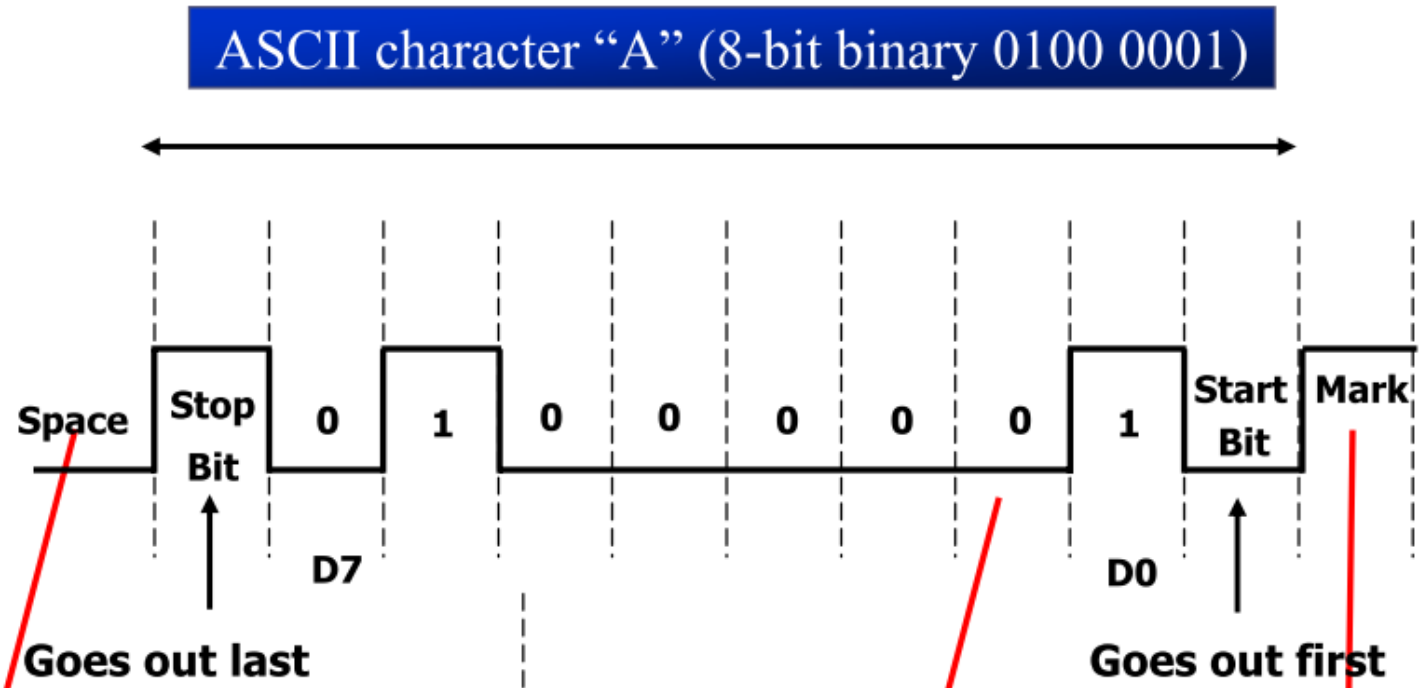
- Serial data communication uses two methods
  - *Synchronous* method transfers a block of data at a time
  - *Asynchronous* method transfers a single byte at a time
- It is possible to write software to use either of these methods, but the programs can be tedious and long
  - There are special IC chips made by many manufacturers for serial communications
    - UART (universal asynchronous Receiver-transmitter)
    - USART (universal synchronous-asynchronous Receiver-transmitter)

BASICS OF SERIAL COMMUNICA-TION

- A *protocol* is a set of rules agreed by both the sender and receiver on
  - How the data is packed
  - How many bits constitute a character
  - When the data begins and ends
- Asynchronous serial data communication is widely used for character-oriented transmissions
  - Each character is placed in between start and stop bits, this is called *framing*
  - Block-oriented data transfers use the synchronous method
- The start bit is always one bit, but the stop bit can be one or two bits

BASICS OF SERIAL COMMUNICA-TION

- The start bit is always a 0 (low) and the stop bit(s) is 1 (high)



The 0 (low) is referred to as *space*

The transmission begins with a start bit followed by D0, the LSB, then the rest of the bits until MSB (D7), and finally, the one stop bit indicating the end of the character

When there is no transfer, the signal is 1 (high), which is referred to as *mark*

BASICS OF SERIAL COMMUNICA-TION

- Due to the extended ASCII characters, 8-bit ASCII data is common
  - In older systems, ASCII characters were 7-bit
- In modern PCs the use of one stop bit is standard
  - In older systems, due to the slowness of the receiving mechanical device, two stop bits were used to give the device sufficient time to organize itself before transmission of the next byte

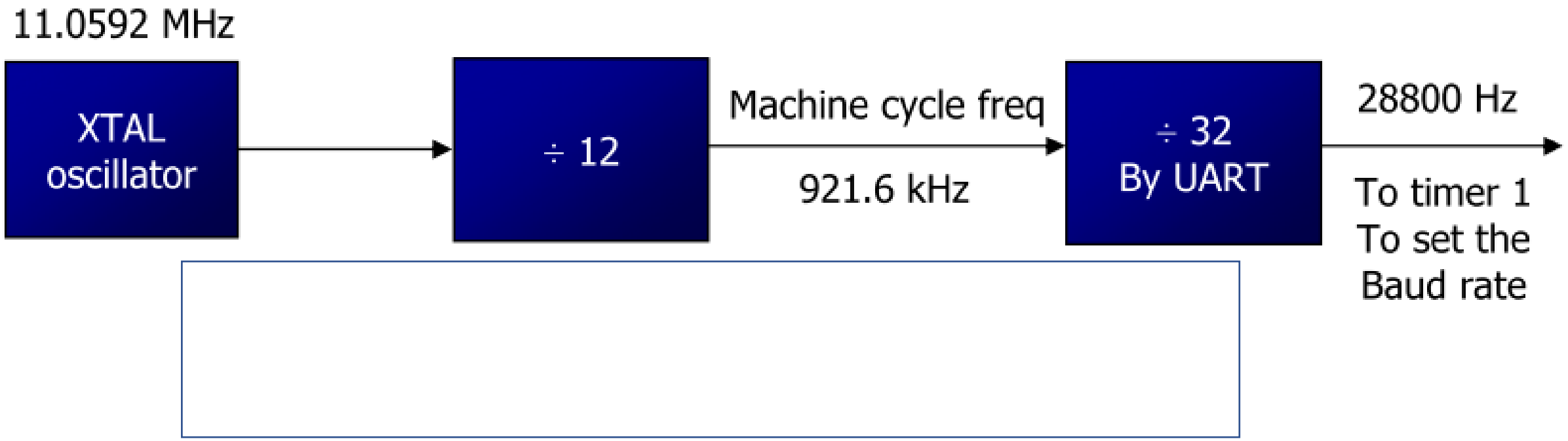
BASICS OF SERIAL COMMUNICA-TION

- Assuming that we are transferring a text file of ASCII characters using 1 stop bit, we have a total of 10 bits for each character
  - This gives 20% overhead, i.e. each 8-bit character with an extra 2 bits
- In some systems in order to maintain data integrity, the parity bit of the character byte is included in the data frame
  - UART chips allow programming of the parity bit for odd-, even-, and no-parity options

BASICS OF SERIAL COMMUNICA-TION

- 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*
  - It is modem terminology and is defined as the 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, and we use the terms interchangeably

- The data transfer rate of given computer system depends on communication ports incorporated into that system
  - IBM PC/XT could transfer data at the rate of 100 to 9600 bps
  - Pentium-based PCs transfer data at rates as high as 56K bps
  - In asynchronous serial data communication, the baud rate is limited to 100K bps



/12 to get machine cycles
and then /32 by UART

BASICS OF SERIAL COMMUNICA-TION

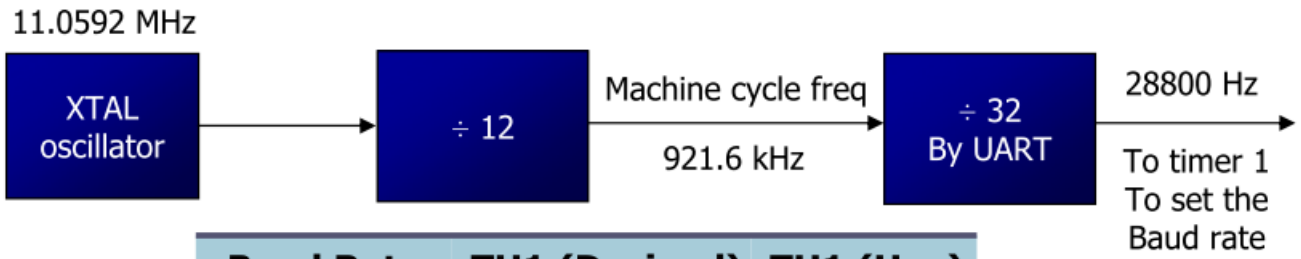
With XTAL = 11.0592 MHz, find the TH1 value needed to have the following baud rates. (a) 9600 (b) 2400 (c) 1200

Solution:

The machine cycle frequency of 8051 = 11.0592 / 12 = 921.6 kHz, and 921.6 kHz / 32 = 28,800 Hz is frequency by UART to timer 1 to set baud rate.

(a) 28,800 / 3 = 9600 where -3 = FD (hex) is loaded into TH1
(b) 28,800 / 12 = 2400 where -12 = F4 (hex) is loaded into TH1
(c) 28,800 / 24 = 1200 where -24 = E8 (hex) is loaded into TH1

Notice that dividing 1/12 of the crystal frequency by 32 is the default value upon activation of the 8051 RESET pin.



Baud Rate	TH1 (Decimal)	TH1 (Hex)
9600	-3	FD
4800	-6	FA
2400	-12	F4
1200	-24	E8

TF is set to 1 every 12 ticks, so it functions as a frequency divider

BASICS OF SERIAL COMMUNICA-TION

- SBUF is an 8-bit register used solely for serial communication
  - For a byte data to be transferred via the TxD line, it must be placed in the SBUF register
    - The moment a byte is written into SBUF, it is framed with the start and stop bits and transferred serially via the TxD line
  - SBUF holds the byte of data when it is received by 8051 RxD line
    - When the bits are received serially via RxD, the 8051 deframes it by eliminating the stop and start bits, making a byte out of the data received, and then placing it in SBUF

- SCON is an 8-bit register used to program the start bit, stop bit, and data bits of data framing, among other things

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial port mode specifier
SM1	SCON.6	Serial port mode specifier
SM2	SCON.5	Used for multiprocessor communication
REN	SCON.4	Set/cleared by software to enable/disable reception
TB8	SCON.3	Not widely used
RB8	SCON.2	Not widely used
TI	SCON.1	Transmit interrupt flag. Set by HW at the begin of the stop bit mode 1. And cleared by SW
RI	SCON.0	Receive interrupt flag. Set by HW at the begin of the stop bit mode 1. And cleared by SW

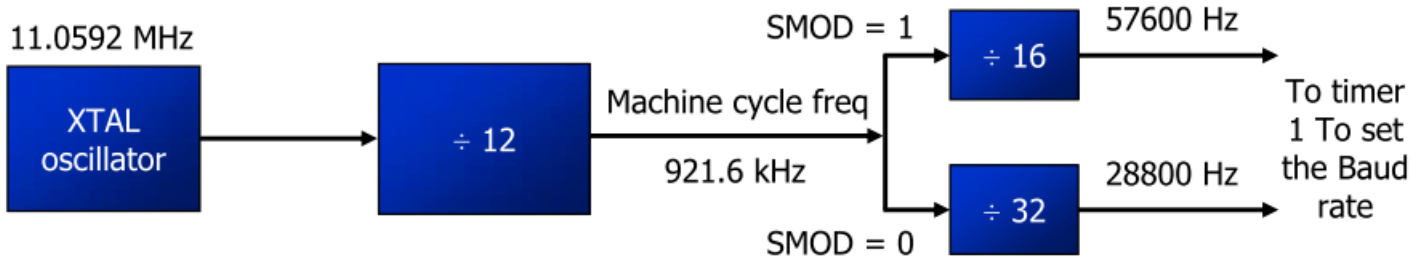
Note: *Make SM2, TB8, and RB8 =0*

- SM0, SM1
  - They determine the framing of data by specifying the number of bits per character, and the start and stop bits

SM0	SM1	
0	0	Serial Mode 0
0	**1**	**Serial Mode 1, 8-bit data, 1 stop bit, 1 start bit**
1	0	Serial Mode 2
1	1	Serial Mode 3

Only mode 1 is of interest to us

- REN (receive enable)
  - It is a bit-adressable register
    - When it is high, it allows 8051 to receive data on RxD pin
    - If low, the receiver is disable
- TI (transmit interrupt)
  - When 8051 finishes the transfer of 8-bit character
    - It raises TI flag to indicate that it is ready to transfer another byte
    - TI bit is raised at the beginning of the stop bit
- RI (receive interrupt)
  - When 8051 receives data serially via RxD, it gets rid of the start and stop bits and places the byte in SBUF register
    - It raises the RI flag bit to indicate that a byte has been received and should be picked up before it is lost
    - RI is raised halfway through the stop bit



Baud Rate comparison for SMOD=0 and SMOD=1

TH1	(Decimal)	(Hex)	SMOD=0	SMOD=1
	-3	FD	9600	19200
	-6	FA	4800	9600
	-12	F4	2400	4800
	-24	E8	1200	2400

SERIAL COMMUNICA-TION PROGRAMMING

Doubling Baud Rate

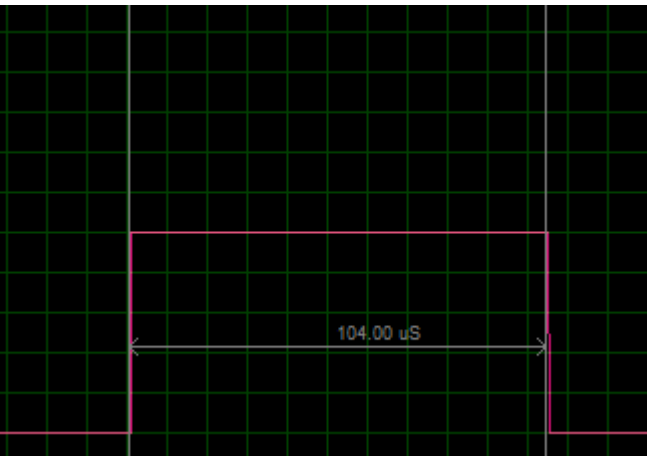
- There are two ways to increase the baud rate of data transfer
  - To use a higher frequency crystal
  - To change a bit in the PCON register

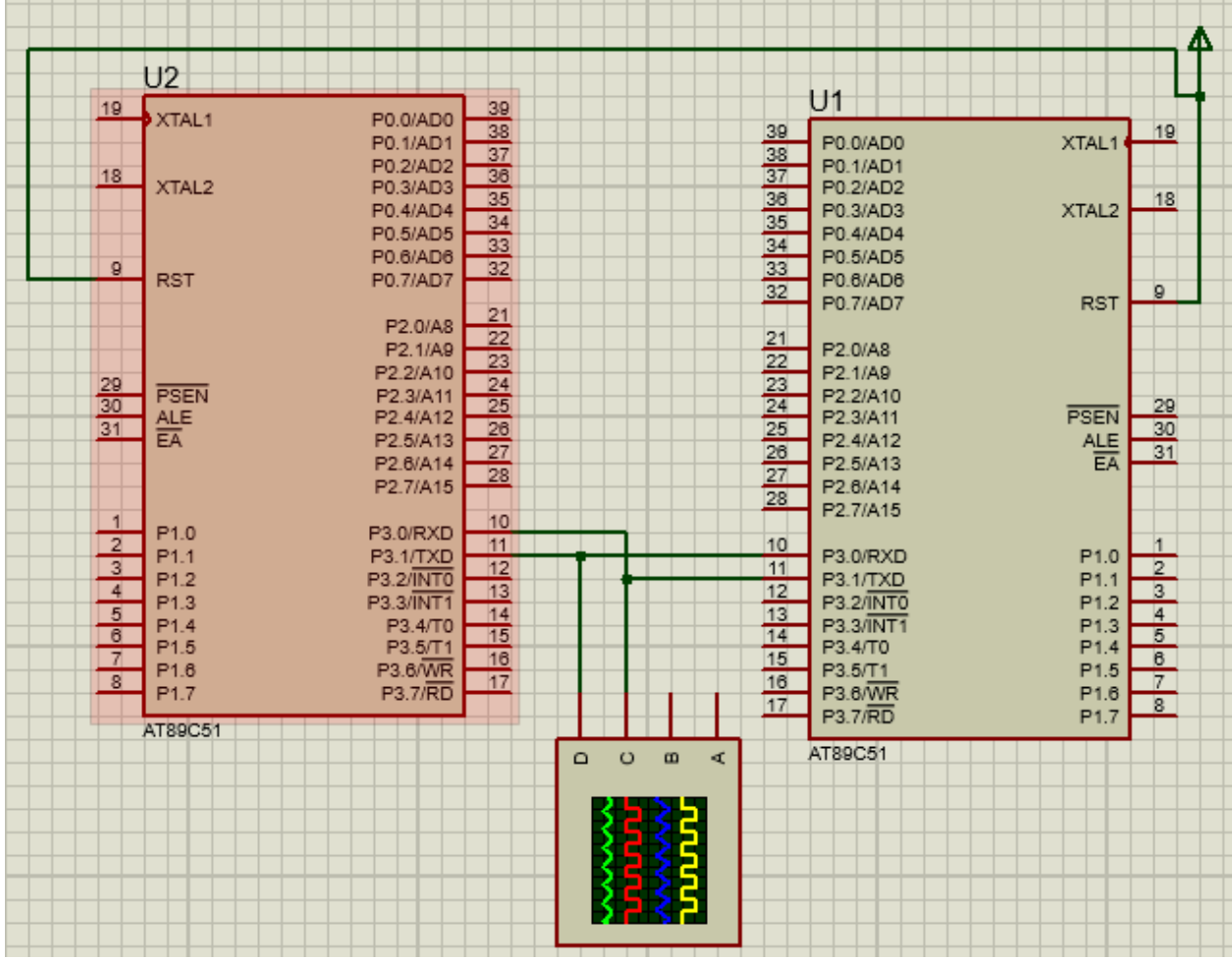
The system crystal is fixed

- PCON register is an 8-bit register
  - When 8051 is powered up, SMOD is zero
  - We can set it to high by software and thereby double the baud rate

SMOD	--	--	--	GF1	GF0	PD	IDL

9600 bps Transmit example



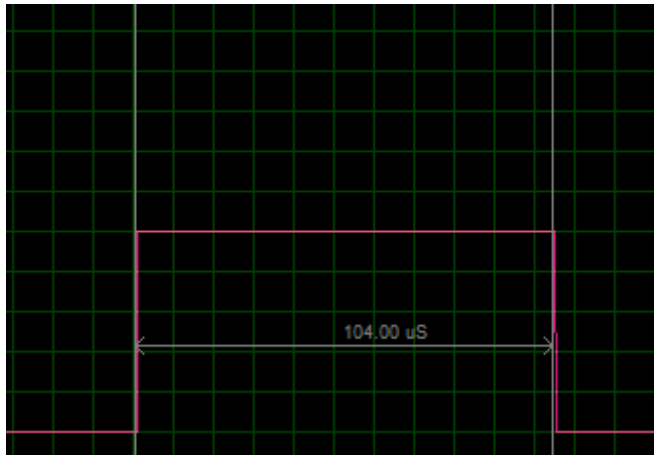


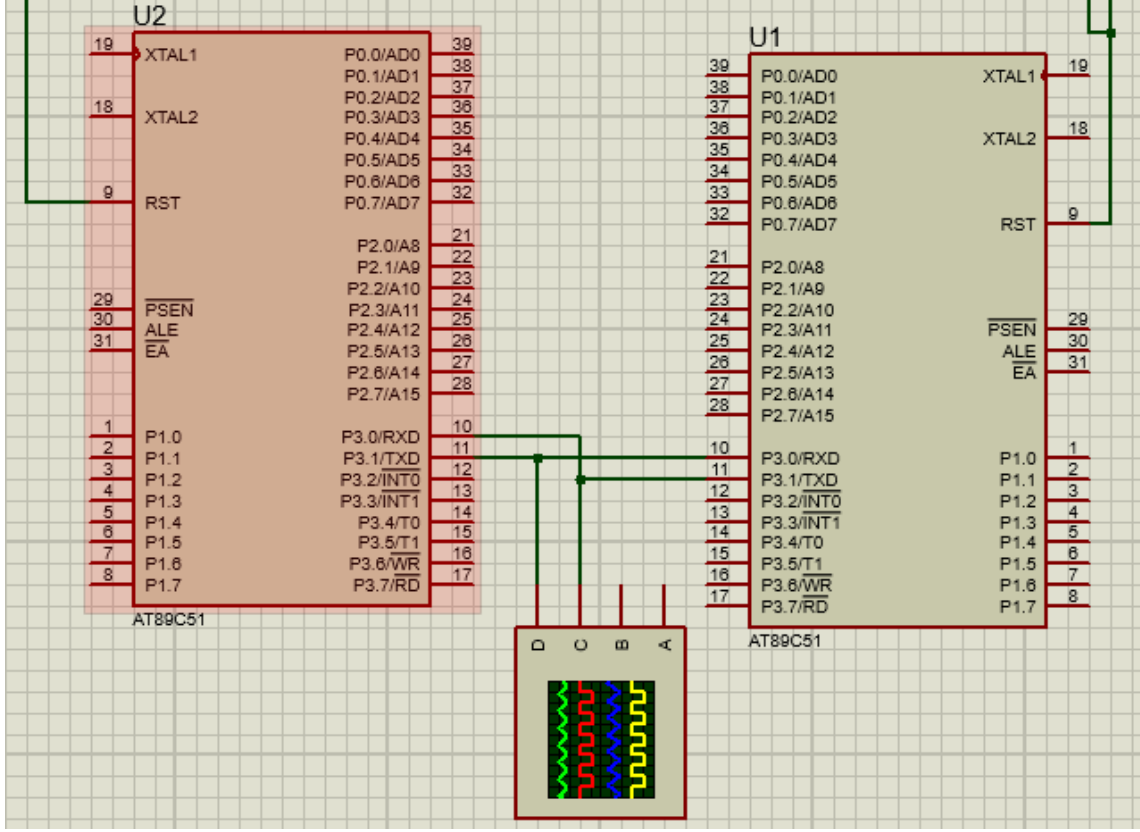

```
unsigned int x = 0x0;

void main(void)
 {
    TMOD = 0x20;      //Timer 1, auto Reload
    TH1 = 0xFD;       //9600 bps
    SCON = 0x50;      // 1 start bit, 1 stop bit (total 10 bits, overhead is 2 bits, data is 8 bits) and REN
    PCON = 0x00;      // SMOD = 0, transmission rate will be 1x

    TR1 = 1;          //Start Timer 1
    while(1){
        SBUF = x;     // We want to transmit x
        while(TI==0);
        TI = 0;
    }
```

9600 bps Receive example






```c
#include <reg51.h>
#include <stdio.h>
unsigned int y = 0;
void main(void)
 {
    TMOD = 0x20;     //Timer 1, auto Reload
    TH1 = 0xFD;      //9600 bps
    SCON = 0x50;     // 1 start bit, 1 stop bit (total 10 bits, overhead is 2 bits, data is 8 bits) and REN
    PCON = 0x00;     // SMOD = 0, transmission rate will be 1x
    TR1 = 1;         // Start Timer 1
    while(1){
        while(RI==0);
        RI = 0;      // clear by SW
        y = SBUF;

        SBUF = y;
        while(TI==0);
        TI = 0;
    }
```