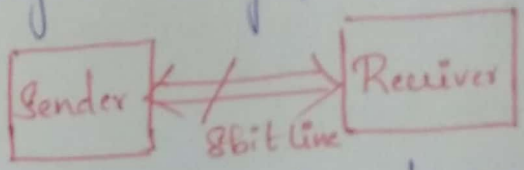
Unit-5

8051 Serial Communication

- Communication means exchange of information between transmitter and receiver. i.e Source & destination.
- Communication are mainly classified into
  1. Serial Communication
  2. Parallel Communication
- Most of microcontroller are designed using parallel Communication.



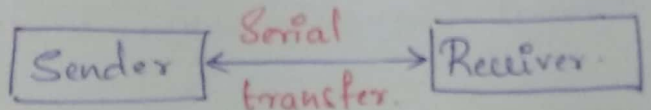
* Parallel Communication :-

- No of lines required to transfer data depends on the number of bits to be transfered.

Eg to transfer 8 bits of data, 8 lines are required and all 8bits are transfered simultaneously.

- Transmitting data over a long distance, using parallel Communication is impractical due to increase in cost of cabling. In such cases we use serial Communication.

* Serial Communication :-



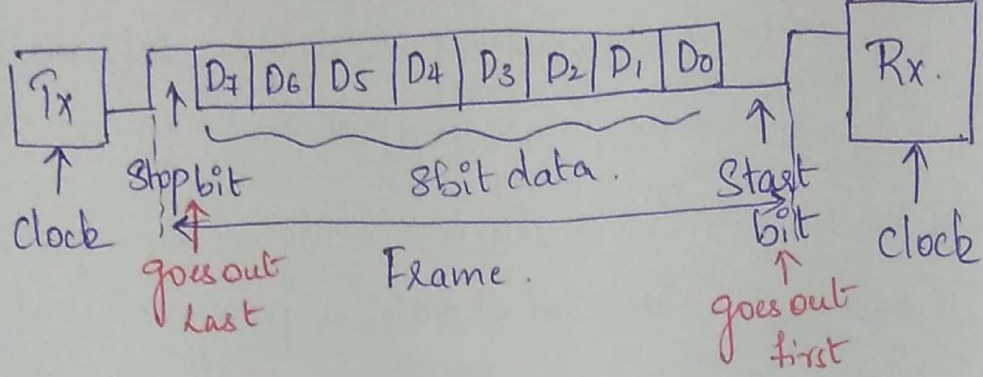
- Serial Communication use a single data line to transfer data.
- Only one bit is transfered at a time over a single data line. It is used for long distance Communication.
- Serial Communication enables two Computers located in two different cities to connect & Communicate over telephone

- It uses single data line instead of 8bit data line hence it is much cheaper when compared to parallel communication.
- In Serial Communication data byte (8bit data) must be converted to serial bits using parallel-in serial out shift registers. and then it can be transmitted over a single
- At the receiver end there must be a serial in parallel out shift register to receive the serial data & pack them into bytes.
- Serial Communication is slower than parallel Comm"
If the data is to be transfered on telephone line, it must be converted into Audio tones from 1's & 0's this conversion is performed by peripheral device called Modem i.e modulator/demodulator.
- When the communication distance is short, the digital signal can be transferred it on a simple wire & requires no modulation.
- the data in Serial Communication is done by two ways i.e

1. Asyncheonous Serial Communication
2. Syncheonous Serial Communication.

* Asyncheonous Serial Communication :-

- Widely used for character-oriented transmission



- Each character is placed blw start and stop bit this is called framing.
- Start bit is always a 0, followed by a character & One or two stop bit. (High always).
- ~~When~~ there is no data transfer, the signal is high (1), which is usually refered as Mark.
- In syncheonous serial Communication, peripheral chips and modem can be programmed for data that is 7bit or 8bit wide.
- In older System ASCII characters were 7bits and in Recent years ASCII characters are 8bits.
- Assuming that we transfer 8bit ASCII characters using 1-stop bit and 1-Start bit i.e total of 10 bits for each character. therefore for each 8bit character there are extra 2bits, which gives 20% overhead.
- In some Systems, parity bit is included in the data frame to maintain data integrity

* Data transfer Rates:-

- the rate of data transfer in serial Communication is Stated in bps (bits per second) also called as Baud rate

- Baud rate is a MODEM terminology which is defined as the number of bit change / signal changes per second.
- 8051 transfers and receives data serially at many different baud rate. The baud rate is programmable.
- A standard crystal frequency = 11.0592 MHz. is assumed to be used to generate the baud rate.
- 8051 divides the crystal frequency by 12 to get machine cycle frequency.

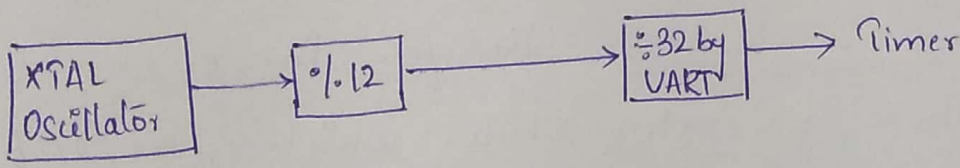
$$\therefore \frac{11.0592M}{12} = 921.6\,KHz.$$

- the 8051's Serial Communication i.e UART circuitry divides the machine cycle frequency by 32 and then feeds it to Timer 1 to set the baud rate.

$$\therefore \frac{\text{Machine Cycle freq}}{32} = \frac{921.6K}{32} = 28,800\,Hz.$$

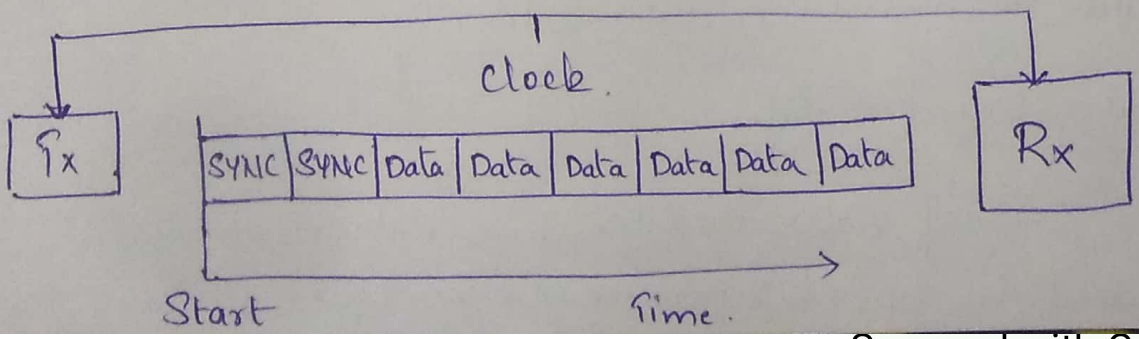
↑ fed to timer 1.

- 8051 baud rate is set by Timer 1 using mode 2 (8bit auto reload).



* Synchronous Serial Communication:-

- this method is used to transfer a block of data at a time.



* Start and stop bits in each frame of asynchronous format represents wasted overhead byte that reduces the overall character rate. this can be eliminated by Synchronizing Rx and Tx by having a common clock signal.

- Synchronous transmission sends Sync bits instead of Stop and start bits.

* Comparison between Asyncheonous and Syncheonous Serial Communication

Asyncheonous Serial Comm"	Syncheonous Serial Comm"
1. Tx and Rx are not Synced by clock.	1. Syncheonized by a common clock.
2. Bits of data are transmitted at constant rate	2. Data bits are transmitted with Syncheonization of clock.
3. Character may arrive at any rate at receiver.	3. Character is received at constant rate.
4. Data transfer is character oriented.	4. Data transfer takes place in blocks.
5. Start and stop bits are required to establish Comm" of each character.	5. Synchronization bits are required to transfer the data block.
6. Used in low-speed transmission at about speed less than 20 Kbps.	6. Used in high speed transmission.

Baud rate & Transmission Rate:-

- Baud rate is defined as number of bits transmitted per second.

- Transmission Rate = $\frac{1 \text{ sec}}{\text{Baud rate}}$.

Assume baud rate = 1200

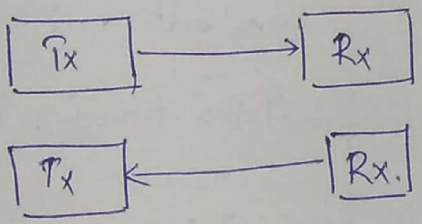
Transmission Rate = $\frac{1 \text{ sec}}{1200}$ = 0.83msec.

∴ 0.83msec is the delay Between two bits.

* Serial data transmission classification:-

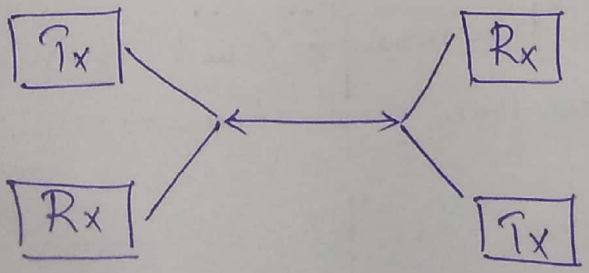
Serial data transmission can be classified on the basis of how transmission occurs:-

1. Simplex:- data is transmitted in only one direction. there is no possibility of data transfer in other direction.

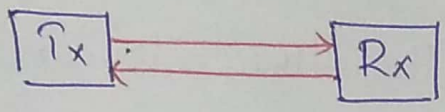


one transfer at a time.

2. Half Duplex:- It allows the data transfer in both direction, but not simultaneously. Eg walkie-talkie



3. Full Duplex: allows the data transfer in both direction simultaneously. & transmission through telephone lines.



* SCON (Serial Control Register):-

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Mode Selector — Transmit Receive bit 8 — Interrupt flags.

- SM0, SM1 : Serial mode specifier.

SM0	SM1	Mode	Description	Baud rate	
0	0	Mode 0	Shift Register	f/12	
0	1	Mode 1	8bit UART	Variable	Set by mode 1/2
1	0	mode 2	9bit UART	f/12 of f/32	
1	1	mode 3	9bit UART	Variable.	Set by mode 1/2

f = Crystal frequency of microcontroller.

- SM2 - this bit is used for multiprocessor Communication In case multiprocessor is not used SM2 = 0.

- REN (Receive enable bit)

REN = 1 - enables reception.

REN = 0 - disables reception.

- TB8 (Transmit bit 8)

Set/cleared by the hardware to determine the state of 9th data bit transmited in 9 bit UART

- RB8 (Receive bit 8)

* Set / cleared by hardware to indicate the state of 9th data bit received (Used in mode 2 and mode 3)
* If 8051 is used, then these two bits are cleared.

- TI (Transmit Interrupt flag):-

* Set by the hardware at the begining of the Stop bit in mode 1. (Set by the hardware whenever byte is transmitted)
* TI is cleared by Software.

- RI (Receive Interrupt flag):-

* Set by hardware at the begining of the Stop bit in mode 1 (Set by the hardware whenever byte is received).
* RI must be cleared by Software.

* PCON (Power mode Control) Special function register:-

SMOD	-	-	-	GP1	GP0	PD	IDL

- PCON is not bit addressable register.

SMOD: Serial Baud Rate Modify bit.

SMOD = 1 by program to double the baud rate using Timer 1 for mode 1, 2, 3.

SMOD = 0 by program to use Timer 1 Baud rate

- Bit 6-4 Not implemented.
- GF1 :- General purpose user flag bit 1. } Set/cleared.
- GF0 :- General purposer user flag bit 0 } by program.
- PD :- Power down bit

  Set to 1 by program to enter power down Configuration for microprocessor.
- IDL :- Idle mode bit

  Set to 1 by program to enter idle mode Configuration for microprocessor.

* Note

1. formula to Compute BAUD Rate :-

$$\text{Baud rate} = \frac{\text{Crystal frequency}}{12 \times 32} \times \frac{1}{256 - TH1}$$

2. Formula to Compute Initial value for particular Baud rate.

$$TH1 = 256 - \frac{\text{Crystal frequency}}{12 \times 32 \times \text{Baud rate}}$$

Doubling the Baud rate in 8051 :-

there are two ways to increase the baud rate

1. Use a high frequency crystal
2. Change a bit value in the PCON register.

- when 8051 is powered up, SMOD bit (D7) of the PCON register is zero.
- We can set it to high by Software and thereby double the baud rate.

SMOD = 0 : Baud rate = $\frac{\text{Crystal frequency}}{12 \times 32} \times \frac{1}{256 - TH1}$

SMOD = 1 : Baud rate = $\frac{\text{Crystal frequency}}{12 \times 16} \times \frac{1}{256 - TH1}$

Set of instruction used to set SMOD of PCON register is as given below.

```
MOV A, PCON    ; Place a copy of PCON in A
SETB ACC.7     ; Make D7 = 1
MOV PCON, A    ; SMOD = 1, without changing any
                 other bits.
```

* SBUF Register :-

- SBUF is an 8bit register used only for serial Comm'
- whenever 8051 wants a byte of data to be transfered via TxD line, it must be placed in SBUF register.
- SBUF holds the byte of data when it is received by 8051's RxD line.
- It can be accessed like any other register in 8051.

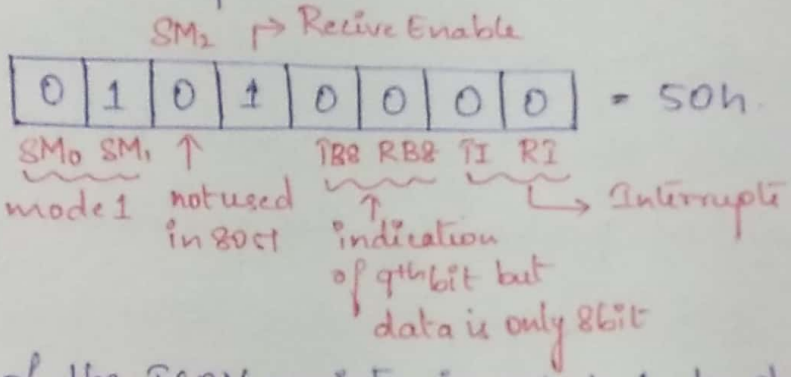
Eg
```
MOV SBUF, #'D'
MOV SBUF, A.
```

Note :- 1. When a byte is written into SBUF, it is framed with the start and stop bits and transfered serially via TxD pin.

2. When bits are received serially via RxD, the 8051 deframes it by eliminating the stop & start bits and then data received is placed on SBUF.

* Procedure to program 8051 to transmit data Serially

1. The TMOD register is loaded with the value 20h, indicating the use of Timer 1 in mode 2 to set the baud rate.
2. The TH1 is loaded with the values to set baud rate for serial data transfer (Assuming XTAL = 11.0592MHz)
3. SCON register is loaded with 50h, indicating Serial mode 1, where an 8bit data is framed with Start and stop bits



4. TR1 of the TCON register is set to 1 to start the Timer1.
5. TI is cleared by the "CLR TI" instruction.
6. The character byte to be ~~transmitted~~ transferred serially is written into SBUF register.
7. The TI flag bit is monitored with the use of the instruction "JNB TI, lable" to see if the character has been transfered Completely.
8. To transfer the next character, go to step 5.

* Importance of the TI flag:-

To understand the importance of the role of TI, look at the following sequence of steps that the 8051 goes through in transmitting a character via TxD.

1. the byte character to be transmitted is written into SBUF register.
2. The start bit is transfered.
3. the 8bit character is transfered one bit at a time.
4. the stop bit is transferred. It is during the transfer of stop bit that the 8051 raises the TI flag (TI=1), indicating that the last character was transmitted and it is ready to transfer next character.
5. By monitoring the TI flag, we make sure that we are not overloading the SBUF register. If we write another byte into SBUF register before TI is raised, the untransmitted portion of the previous byte will be lost. In other words, when 8051 finishes transferring a byte, it raises the TI flag to indicate it is ready for the next character.
6. After SBUF is loaded with a new byte, the TI flag bit must be forced to 0 by the "CLR TI" instruction in order for this new byte to be transferred.

* Procedure to program the 8051 to Receive data Serially.

1. TMOD register is loaded with the value 20H, indicating the use of Timer 1 in mode 2 (8bit auto reload) to set the baud rate.
2. The TH1 is loaded with one of the value to set Baud rate for serial data transfer.

3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8bit data is framed with start and stop bit.
4. TR1 is set to 1 to start timer 1.
5. RI is cleared by "CLR RI" instruction.
6. the RI flag bit is monitored with the use of the transmission "JNB RI, lable" to see if an entire character has been received yet.
7. When RI is raised, SBUF has the byte. Its contents are moved into a safe place.
8. To receive the next character, go to step 5.

* Importance of RI flag:-

In Receiving bits via RxD pin, the 8051 goes through the following steps.

1. It receives the start bit indicating that the next bit is first bit of the character byte it is about to receive.
2. the 8bit character is received one bit at a time. When the last bit is received, a byte is about to receive.
3. the stop bit is received. When receiving the stop bit the 8051 makes RI = 1, indicating that an entire character byte has been received and must be placed up before it gets overwritten by an incoming character.

4. By checking the RI flag bit when it is raised, we know that a character has been received and is sitting in SBUF register. We copy the SBUF contents to a safe place in some other register or memory before it is lost.

5. After SBUF content are copied into a safe place, the RI flag bit must be forced to 0 by the "CLR RI" instruction in order to allow the next received character byte to be placed in SBUF.

Failure to do this causes loss of the received character.

1. Write a program to transfer a letter 'Y' serially at 9600 baud continuously, and also to send a letter 'N' through port 0, which is connected to a display device.

Sol: Baud rate = 9600

Step 1: TMOD = 20h Timer 1 mode 2

Step 2: TH1 = value to set baudrate.

$$TH1 = 256 - \frac{\text{crystal freq.}}{12 \times 32 \times \text{Baud rate}}$$

$$= 256 - \frac{11.0592 \times 10^6}{12 \times 32 \times 9600}$$

$$= 256 - 3 = 253d.$$

$$= FDh$$

3. SCON = 50h – Serial mode 1
   8bit data framed with start & stopbit
4. TR1 = 1 – Start the timer 1.
5. clear the TI flag
6. SBUF = 'Y' – character transfered serially.
7. Monitor TI to check if character has been transfered
8. Send 'N' to port P0
9. jump back to step 5. to transfer again.

Program :-

```
        Org 00h
        MOV TMOD, #20h  ; timer 1 mode 2
        MOV TH1, #Fdh   ; Set baud rate = 9600
        MOV SCON, #50h  ; 8bit, 1stop, REN enable
        SETB TR1        ; Start timer 1.
again:  CLR TI          ; Clear TI for transmission.
        MOV SBUF, #'Y'  ; transfer 'Y' serially
here:   JNB TI, here    ; wait for transmission to get over.
        MOV P0, #'N'    ; move 'N' to P0 for parallel transfer
        Sjmp again      ; Repeat.
        END.
```

2. Write a 8051 assembly level programs to transfer the message "hello" serially at 9600 baud rate, 8bit data, 1 stop bit continuously.

```
Org 00h
MOV TMOD, #20h ; timer 1 mode 2
MOV TH1, #FDh ; Baud rate 9600, 8bit data 1stop
MOV SCON, #50h ; 8bit, 1stop, REN enable.
SETB TR1 ; Start timere.
Start: MOV A, #'H' ; Character → A
       ACall trans ; Serial transmission. Subroutine
       MOV A, #'E'.
       ACall trans
       MOV A, #'L'
       ACall trans
       MOV A, #'L'
       ACall trans
       MOV A, #'O'
       ACall trans.
       Sjmp Start ; Repeat next transmission.
trans: MOV SBUF, A ; transfer character in A serially
Here: JNB TI, Here ; wait for completion of transfer.
      CLR TI ; clear TI for next transmission.
      RET ; Return to main program.
      End.
```

3. Take a data in through port 0, 1, 2, one after the other and transfer this data serially and continuously with a Baud rate of 4800.

$$TH1 = 256 - \frac{\text{Crystal frequency}}{12 \times 32 \times \text{Baud rate}}$$

$$= 256 - \frac{11.0592 \times 10^6}{12 \times 32 \times 4800}$$

TH1 = 256 − 6 = 250d.

= FAh.

Program:

```
0x9000h
MOV TMOD, #20h  ; timer 1 mode2
MOV TH1, #FAh   ; 4800 baud rate
MOV SCON, #50h  ; 8bit data, 1stop bit, REN enable
MOV P0, #0FFh
MOV P1, #0FFh   } data to the port P0, P1, P2
MOV P2, #0FFh
SETB TR1        ; Start the timer.
again: MOV A, P0    ; Content of P0 → A
       ACall send   ; call subroutine to transmit
       MOV A, P1
       ACall send.
       MOV A, P2
       ACall send.
       Sjmp again ; Repeat for next set of transmission
Send: MOV SBUF, A   ; Content of A → SBUF
here: JNB TI, here  ; Wait for completion of transfer.
      Clear TI      ; Clear the TI flag
      Ret           ; Return to main program.
      End.
```

4. Port 0 of an 8051 is used to monitor a parameter in an industrial environment. If the parameter gives the reading above 0Fh, a message 'Hi' is to be sent serially. Otherwise, a message 'Ok' is to be sent. The word Hi and Ok are burned into program ROM location. (Baudrate = 9600)

Program:

```
        Org 00h
        MOV P0,#0FFh  ; make P0 as i/p port
        MOV TMOD, #20h ; Timer 1 mode2
        MOV TH1, #FDh ; 9600 Baud rate
        MOV SCON, #50h ; 8bit data, 1stop, REN=1
        SETB TR1 ; Start timer.
Check : MOV A, P0 ; P0 → A
        CJNE A, #0Fh, test → if A≠0Fh jmp to test
        Sjmp OK → A=0Fh jump to OK.
Test : JNC Hi → jump to hi lable.
OK : MOV DPTR, #00A0h → DPTR points to OK msg location
     Acall Access → Call Subroutine to access msg.
     Sjmp check → Continue monitoring P0
Hi : MOV DPTR, #0090h → DPTR points to Hi msg
     Acall Access → Subroutine location
     Sjmp check → Continue monitoring P0.
```

* Subroutine to access the message ROM where msg is stored.

```
Access : CLR A
         MOVC A, @DPTR+A
         ACall Send
         INC DPTR
         CLR A
         MOVC A, @DPTR+A
         ACall Send.
         RET
```

* Subroutine to send data serially.

```
Send : MOV SBUF, A
here : JNB TI, here
```

```
        CLR TI
        RET
        ORG 0090h
mes1 : DB, "HI"
        ORG 00A0h
mes2 : DB "Ok"
        End.
```

5. Generate a Square wave at pin P1.2. this Square wave is to be sent to a receiver Connected in Serial form to Soct. Write the program. Assume Baudrate is 9600.

Timer 0 mode 2 is used to generate the Square wave on P1.2. whenever this pin is high a data FFh is transmitted Serially, and when this pin is low, data 00h is transmitted. this data can be Converted into parallel form at receiver side to regenerate Square wave.

```
        Org 0000h
        MOV TMOD, #22h  ; timer 0, mode2 & timer 1 mode2
        MOV TH1, #FDh   ; Baud rate 9600
        MOV SCON, #50h  ; 8bit, 1stop bit, REN=enable
        MOV TH0, #00h   ; Count value of timer 0
        SETB TR1        ; Start timer 1 for transfer of data
        MOV A, #00h     ; A = 00
        CLR P1.2        ; clear Port pin P1.2
```

```
Rept: SetB TF0, Start  ; Start timer 0 for pulse generation.
Back: JNB TF0, Back ; wait till timer 0 overflows
      CPL A  ; Complement A
      CPL P1.2 ; Complement P1.2
      MOV SBUF, A ; Content A → SBUF for Serial transmission.
      CLR TR0 ; Stop timer
      CLR TF0 ; clear timer flag.
Here: JNB TI, here ; wait for transmission completion
      CLR TI
      Sjmp Rept ; repeat next transmission.
      End.
```

6. Write a program to send the text string "Hello" to Second Serial port. Set the baud rate at 9600, 8-bit data and 1 stop bit.

SFR's address External
- SCON 1 → Second Serial port – add = 0C0h
- SBUF 1 → Second Serial SBUF - add = 0C1h
- TI 1 → Second Serial TI flag - add = 0C1h.

Prog :-

```
SCON1  EQU  0C0h   } Second Serial
SBUF1  EQU  0C1h   } SBUF, SCON & TI address location
TI1    EQU  0C1h   }
Org 00h
MOV TMOD, #20h ; timer 1 mode2
MOV TH1, #FDh ; 9600 baud rate
MOV SCON1, #50h ; 8bit, 1 stop, REN=1
```

```
        SetB  TR1     ; Start the timer 1
        MOV  DPTR, #mess  ; display "HELLO"
again:  CLR A
        MOVC  A, @A+DPTR.  ; read the value.
        JZ  S1.    ; check for the end of line.
        ACall  Send  ; Send to serial port.
        INC  DPTR.  ; move to next value.
        Sjmp  again.  ; Repeat
S1:     Sjmp  S1.  ; End.
Send:   MOV SBUF1, A  ; place A value in buffer.
here:   JNB  TI1, here; wait until transmission Couplets
        CLR TI1  ; clear TI flag.
        RET
mess:   DB  "Hello", 0
        End.
```

7. Program the Second Serial port to receive bytes of data serially and o/p them on P1. Set the baud rate at 4800, 8bit data and 1 stop bit.

Program:

```
SBUF1  EQU  0C1h
SCON1  EQU  0C0h
 RI1   EQU  0C0h
```

} Second Serial SBUF, SCON, RI address location

```
        Org  00h
        MOV  TMOD, #20h.; Timer 1, mode2
        MOV  TH1, #FAh ; 4800 baud.
        MOV  SCON1, #50h ; 8bit, 1stop, REN=1
        SetB  TR1  ; Start timer 1
```

```
Here: JNB RI1, here ; Wait here for data to
                      come in and save the
      MOV A, SBUF1. ; data into Acc
      MOV P1, A ; display on P1
      CLR RI1 ; clear RI flag.
      Sjmp here
      End.
```

8. Assume that a switch is connected to pin 2.0. Write a program to monitor the switch and perform to operation as follows.

   1. If SW=0 send message "Hello" to serial #0 port
   2. If SW=1 send message "Good bye" to serial #1 port.

Program:

```
SCON1  EQU  0C0h   } Second serial transmission
TI1    EQU  0C1h   } address locations
SW1    BIT  P2.0   }

Org 00h
MOV TMOD, #20h ; Timer 1, mode2
MOV TH1, #FDh ; 9600 baud
MOV SCON, #50h } 8bit, 1 stop, REN=enable.
MOV SCON1, #50h }
SETB TR1 → Start timer 1.
SETB SW1 → make SW1 an input to port P2.0
S1: JB SW1, next → check SW1=1
    MOV DPTR, #mesc1 → if SW1=0 display "hello"
FN: CLR A
    MOVC A, @A+DPTR ; read value
    JZ S1 ; check for end line
    ACall ~~delay~~ Send1 ; Send msg to serial port
    INC DPTR ; move to next value
    Sjmp FN
```

```
Next : MOV DPTR, #mess2 ; if SW1 = 1 display
                          'Good bye"
LN : CLR A
     MOVC A, @A+DPTR ; read value.
     JZ S1   ; check for end line
     ACall ~~delay~~ Send2 ; Send to serial port
     INC DPTR ; move to next value.
     Sjmp LN.
Send 1 : MOV SBUF, A ; place value in buffer.
here : JNB TI, here ; wait for transmission to complete
       CLR TI ; clear flag.
       RET
Send 2 : MOV SBUF, A
here1 : JNB TI1, here1.
        CLR TI1
        RET
mess 1 : DB "HELLO", 0
mess 2 : DB "Good bye", 0
         End.
```