Serial Port Programming in Assembly

1. Serial port programming in assembly

Since IBM PC/compatible computers are so widely used to communicate with 8051-based systems, serial communications of the 8051 with the COM port of the PC will be emphasized. To allow data transfer between the PC and an 8051 system without any error, we must make sure that the baud rate of the 8051 system matches the baud rate of the PC‟s COM port.

1.1 Baud rate in the 8051

The 8051 transfers and receives data serially at many different baud rates. Serial communications of the 8051 is established with PC through the COM port. It must make sure that the baud rate of the 8051 system matches the baud rate of the PC's COM port/ any system to be interfaced. The baud rate in the 8051 is programmable. This is done with the help of Timer. When used for serial port, the frequency of timer tick is determined by (XTAL/12)/32 and 1 bit is transmitted for each timer period (the time duration from timer start to timer expiry).

The Relationship between the crystal frequency and the baud rate in the 8051 is that the 8051 divides the crystal frequency by 12 to get the machine cycle frequency which is shown in figure1. Here the oscillator is XTAL = 11.0592 MHz, the machine cycle frequency is 921.6 kHz. 8051's UART divides the machine cycle frequency of 921.6 kHz by 32 once more before it is used by Timer 1 to set the baud rate. 921.6 kHz divided by 32 gives 28,800 Hz. Timer 1 must be programmed in mode 2, that is 8-bit, auto-reload.

1.1.1 Calculation of baud rate:

In serial communication if data transferred with a baud rate of 9600 and XTAL used is 11.0592 then following is the steps followed to find the TH1 value to be loaded.

Clock frequency of timer clock: f = (11.0592 MHz / 12)/32 = 28,800Hz

Time period of each clock tick: T0 = 1/f = 1/28800

Duration of timer : n*T0 (n is the number of clock ticks)

9600 baud ->duration of 1 symbol: 1/9600

1/9600 = n*T0 = n*1/28800

n = f/9600 = 28800/9600 = 3 ->TH1 =-3

Similarly, for baud 2400

n = f/2400 = 12 ->TH1 = -12

Example: set baud rate at 9600

```
MOV TMOD, #20H  ;  timer 1,mode 2(auto reload)
MOV TH1, #-3            ; To set 9600 baud rate
SETB TR1                 ; start timer 1
```

1.1.2 Baud rate selection

Baud rate is selected by timer1 and when Timer 1 is used to set the baud rate it must be programmed in mode 2 that is 8-bit, auto-reload. To get baud rates compatible with the PC, we must load TH1 with the values shown in Table 1.

Table.1 Timer 1 THI register values for different baud rates

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

Note: XTAL = 11.0592 MHz.

1.2 Registers for serial communication

1.2.1 SBUF (serial buffer) register:

It is an 8 bit register used solely for serial communication in the 8051.A byte of data to be transferred via the TxD line must be placed in the SBUF register. SBUF holds the byte of data when it is received by the RxD line. It can be accessed like any other register

```
MOV SBUF, #'D'  ; load SBUF=44H, ASCII for 'D,,

MOV SBUF, A   ; copy accumulator into SBUF

MOV A, SBUF   ; copy SBUF into accumulator
```

when a byte is written, it is framed with the start and stop bits and transferred serially via the TxD pin. when the bits are received serially via RxD, it is deframed by eliminating the stop and start bits, making a byte out of the data received, and then placing it in the SBUF.

1.2.2 SCON (serial control) register:

It is an 8 bit register used to program start bit, stop bit, and data bits of data framing, among other things.

The first two bits are SM0, SM1 which is the serial port mode bits. It is used to specify framing format, how to calculate baud. For example if (SM0, SM1) = (0,1), mode 1: 8-bit data, 1 start bit, 1 stop bit, variable baud rate can be set by timer. The other three modes are rarely used and they are (SM0,SM1) = (0,0), mode 0: fixed baud = XTAL/12,(SM0, SM1) = (1,0), mode 2: 9-bit data, fixed baud,(SM0, SM1) = (1, 1), mode 3: 9-bit data, variable baud. The third bit is used to select the type of processor used for communication. If SM2 is 0 means it is single processor communication. If SM2 is 1, then it is multiprocessor communication. The fourth bit REN is Receive Enable which is used to enable/disable reception. If REN=1,then 8051 will accept incoming data from serial port. If REN=0, then the receiver is disabled. E.g. SETB REN,CLR REN, SETB SCON.4, CLR SCON.4.

The fifth bit is TB8 which is used by modes 2 and 3 for the 8-bit transmission. When mode 1 is used the pin TB8 should be cleared. The sixth bit RB8 is used by modes 2 and 3 for the reception of bit 8. It is used by mode1 to store the stop bit. The seventh bit is TI which is the Transmit Interrupt. When 8051 finishes the transfer of the 8-bit character, it sets TI to ”1” to indicate that it is ready to transfer the next character. The TI is raised at the beginning of the stop bit. The last bit is the RI which is the receive interrupt. When 8051 receives a character,the UART removes start bit and stop bit. The UART puts the 8-bit character in SBUF. RI is set to „1“ to indicate that a new byte is ready to be picked up in SBUF.RI is raised halfway through the stop bit

1.3 Steps to send data serially:

1. Set baud rate by loading TMOD register with the value 20H, this indicating timer 1 in mode2 (8-bit auto-reload) to set baud rate

2. The TH1 is loaded with proper values to set baud rate for serial data transfer

3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8- bit data is framed with start and stop bits

4. TR1 is set to 1 to start timer 1

5. TI is cleared by CLR TI instruction

6. The character byte to be transferred serially is written into SBUF register

7. The TI flag bit is monitored with the use of instruction JNB TI,xx to see if the character has been transferred completely

8. To transfer the next byte, go to step 5

1.3.1 Program to transfer letter "D" serially at 9800baud, continuously:

```
MOV TMOD,#20H ; timer 1,mode 2(auto reload)
MOV TH1, #-3 ; 9600 baud rate
MOV SCON, #50H ; 8-bit, 1 stop, REN enabled
SETB TR1 ; start timer 1
AGAIN: MOV SBUF, #"D" ; letter "D" to transfer
HERE: JNB TI, HERE ; wait for the last bit
CLR TI ;clear TI for next char
SJMP AGAIN ; keep sending A
```

1.3.2 Importance of the TI flag:

Check the TI flag bit, we know whether or not 8051 is ready to transfer another byte. TI flag bit is raised by the 8051 after transfer of data. TI flag is cleared by the programmer by instruction like "CLR TI". When writing a byte into SBUF, before the TI flag bit is raised, it may lead to loss of a portion of the byte being transferred.

1.4 Steps to receive data serially:

1. Set baud rate by loading TMOD register with the value 20H, this indicating timer 1 in mode 2 (8-bit auto-reload) to set baud rate
2. The TH1 is loaded with proper values to set baud rate

3. The SCON register is loaded with the value 50H, indicating serial mode 1, where an 8- bit data is framed with start and stop bits

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 instruction JNB RI,xx 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 next character, go to step 5

1.4.1 Program to receive bytes of data serially, and put them in P2, set the baud rate at 9600, 8-bit data, and 1 stop bit:

```
MOV TMOD, #20H ; timer 1,mode 2(auto reload)

MOV TH1, #-3 ; 9600 baud rate

MOV SCON, #50H ; 8-bit, 1 stop, REN enabled

SETB TR1 ; start timer 1 ; wait for char to come in

HERE: JNB RI, HERE ; saving incoming byte in A

MOV A, SBUF ; send to port 1

MOV P2, A ; get ready to receive next byte

CLR RI   ; keep getting data

SJMP HERE
```

1.4.2 Importance of the RI flag bit:

It receives the start bit, next bit is the first bit of the character about to be received. When the last bit is received, a byte is formed and placed in SBUF. when stop bit is received, it makes RI = 1 indicating entire character byte has been received and can be read before overwritten by next data. When RI=1, received byte is in the SBUF register, copy SBUF contents to a safe place. After the SBUF contents are copied the RI flag bit must be cleared to 0.

1.5 Increasing the baud rate:

Baud rate can be increase by two ways-

1. Increasing frequency of crystal

2. Change bit in PCON register

1.5.1 PCON

It is 8-bit register. When 8051 is powered up, SMOD is zero.By setting the SMOD, baud rate can be doubled.If SMOD = 0 (which is its value on reset), the baud rate is 1/64 the oscillator frequency.If SMOD = 1, the baud rate is 1/32 the oscillator frequency.

1.5.2 Baud rate comparison:

Table.1 Timer 1 THI register values for different baud rates

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

Note: XTAL = 11.0592 MHz.

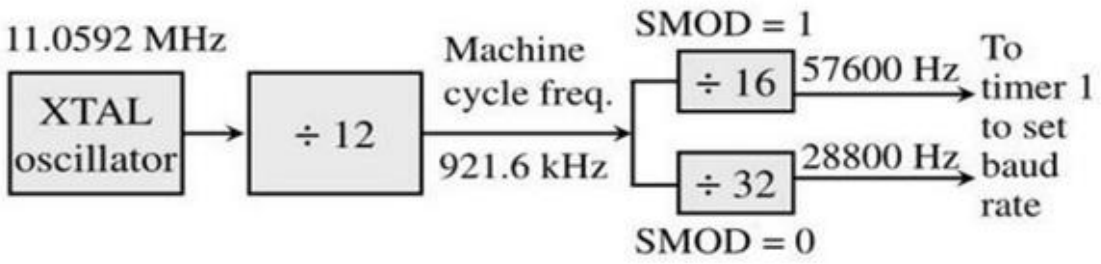

Figure 4. XTAL Oscillator

1.6 Program for Sending and receiving data serially:

Assume that the 8051 serial port is connected to the COM port of IBM PC, and on the PC, we are using the terminal.exe program to send and receive data serially. P1 and P2 of the 8051 are connected to LEDs and switches, respectively. Write an 8051 program to (a) send to PC the message “We Are Ready”, (b) receive any data send by PC and put it on LEDs connected to P1, and (c) get data on switches connected to P2 and send it to PC serially. The program should perform part (a) once, but parts (b) and (c) continuously, use 4800 baud rate.

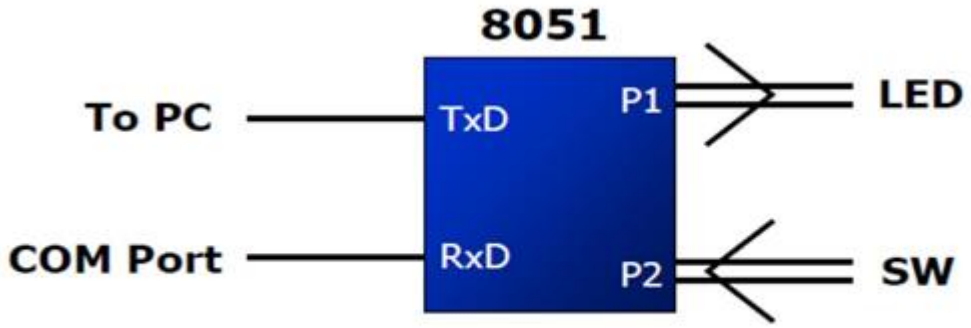

Figure 5. Serial Communication of 8051

```
        ORG 0
        MOV P2, #0FFH           ; make P2 an input port
        MOV TMOD,#20H           ;timer 1, mode 2
        MOV TH1, #0FAH          ;4800 baud rate
        MOV SCON, #50H          ; 8-bit, 1 stop, REN enabled
        SETB TR1                ; start timer 1
        MOV DPTR, #MYDATA       ; load pointer for message
H_1: CLR    A
     MOV A,@A+DPTR              ;get the character
        JZ    B_1               ;if last character get out
        ACALL SEND          ;otherwise call transfer
        INC   DPTR          ;next one
        SJMP H_1                ;stay in loop
B_1:    MOV a,P2                ;read data on P2
        ACALL SEND          ;transfer it serially
        ACALL RECV              ;get the serial data
        MOV P1,A            ;display it on LEDs
         SJMP B_1           ;stay in loop indefinitely
; ---------serial data transfer. ACC has the data----------
        SEND: MOV SBUF,A        ;load the data
H_2:    JNB TI,H_2              ;stay here until last bit gone
        CLR TI                  ; get ready for next char
        RET                     ;return to caller
```

1.7 Serial port programming enhancement

Normal serial data transfer is wasting of the microcontroller's time to poll the TI and RI flags. In order to reduce time, use interrupts instead of polling. New generations of 8051 microcontroller come with two serial ports. It is possible to program second serial port also**.**