e-PG Pathshala

Subject: Computer Science

Paper: Embedded System
Module: Serial Port Programming in Embedded C
Module No: CS/ES/21

Quadrant 1 – e-text

In this lecture basics of serial port communication in 8051 will be discussed. The Control register in 8051 will be visited. Embedded C program for Serial port communication will be discussed in detail with examples.

1. Serial Vs Parallel

Data transferred by computer occurs in two ways. They are parallel and serial communication. Parallel Communication has 8 or more lines. It is used for transfer of data to a device that is only a short distance, while serial communication is used to transfer data over a long distance. The data is sent one bit at a time in serial communication, whereas multiple bits are sent simultaneously in parallel communication. The difference between serial communication and parallel communication is shown in Figure 1.

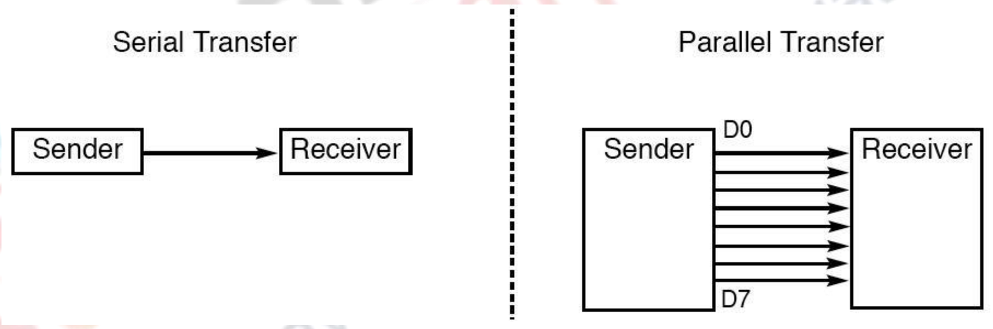


Figure 1. Serial Vs Parallel Communication

1.1 Serial Communication

The transmitter converts a byte of data into serial bits using parallel-in-serial-out shift register, while the receiver has a serial-in-parallel-out shift register to receive the serial data and pack them into a byte. For a short distance, the digital signal can be transferred on a simple wire without modulation. To transfer data over a long distance, some form of modulation may be required. For example, to transfer data on the telephone line, it must be converted from 0s and 1s to audio tones. The device which performs conversion of 0s and 1s to audio tones, and back, is called a modem, "Modulator/demodulator".

The serial communication uses two techniques for communication. They are

- Synchronous- transfer block of data(characters) at a time
- Asynchronous- transfer single byte at a time

This communication can be done by the following hardware:

- USART (Universal Synchronous-Asynchronous Receiver-Transmitter)

- UART (Universal Asynchronous Receiver Transmitter)

The 8051 chip has a built-in UART.

Asynchronous communication is used for character-oriented transmissions, but each character is placed in between start and stop bits; this is called framing. The start bit is always one bit and the stop bit can be one or two bits. The start bit is always a 0 (low), while the stop bit(s) is 1 (high) and the LSB is sent out first. Block-oriented data transfers use the synchronous method. Data transfer rate in serial data communication is mentioned in bps (bits per second) or baud rate.

1.2 RS232 standards

It is the most widely used serial I/O interfacing standard for serial communication. The input and output voltage levels are not TTL compatible. The bit-value of 1 is represented as -3 to -25 V and the value0 is represented as +3 to +25 V, while -3 to +3 is undefined. To connect RS232 to a microcontroller system, it must use voltage converters such as MAX232 to convert the TTL logic levels to the RS232 voltage levels, and vice versa. MAX232 IC chips are commonly referred to as line drivers.

1.3 8051 connections to RS232

8051 has two pins used for transferring and receiving data serially. They are TxD and RxD pins. TxD and RxD are part of the port 3 group (P3.0 and P3.1), while pin 11 (P3.1) of 8051 is assigned to TxD. The pin 10 (P3.0) of 8051 is assigned to RxD. These pins are TTL compatible. They require a line driver to make them RS232 compatible. The driver used for this is MAX232 chip.

1.4 Serial Communication Registers

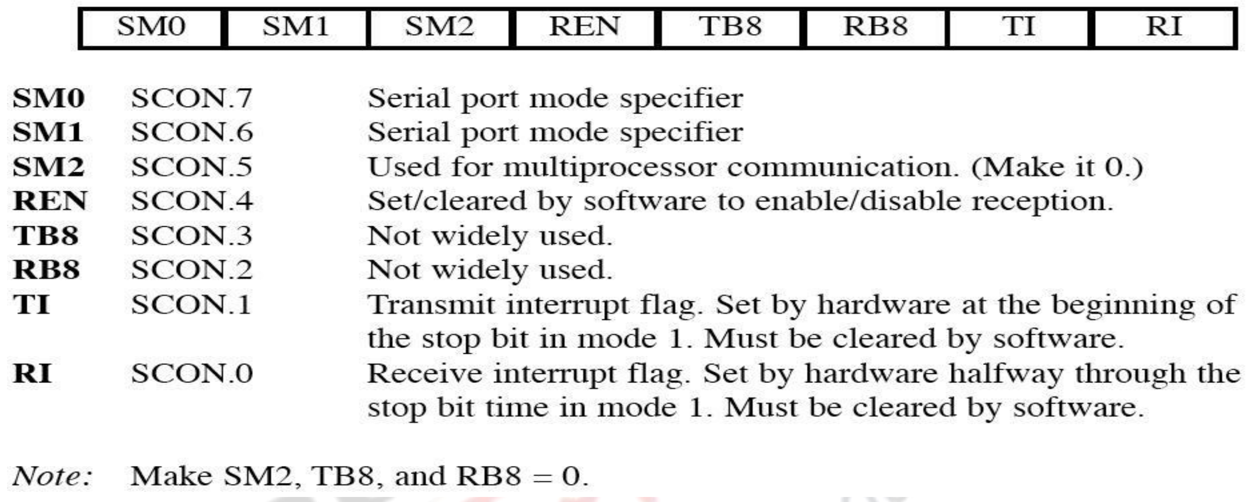
SBUF Register:

It is an 8-bit register used for serial communication. To transmit a byte of data via the TxD line, it must be placed in the SBUF register. When 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, 8051 deframes it by removing the stop and start bits, making a byte out of the data received, and then places it in SBUF register.

SCON (serial control) register:

It is an 8-bit register used to program start bit, stop bit, and data bits of data for framing, among other things. In this SM0 and SM1 are used to determine the mode. If (SM0,SM1) is (0,0) then it is mode 0, in this mode the serial port function as half duplex serial port with fixed baud rate.If (SM0,SM1) is (0,1) it is mode1,in this mode the serial port function as full duplex serial port with variable baud rate. If (SM0,SM1) is (1,0) then it is mode 2, in this mode the serial port function as full duplex serial port with a baud rate of either 1/32 or 1/64 of the oscillator frequency. If (SM0,SM1) is (1,1) then it is mode 3.The mode-3 is same as mode-2, except the baud rate.In all these modes, only mode 1 is important. When mode 1 is chosen, the following bits are compatible with the COM port of PCs. They are 8 bits of data with 1 stop bit, and 1 start bit. The mode 1 allows the baud rate to be variable and is set by Timer 1 of the 8051. For each

character a total of 10 bits are transferred, where the first bit is the start bit, followed by 8 bits of data, and finally 1 stop bit.
REN is the receive enable. If REN=1, it allows 8051 to receive data on the RxD. If 8051 is to both transfer and receive data, REN must be set to 1. If REN=0, the receiver is disabled and TB8 is used for serial modes 2 and 3. The figure 2 shown below specifies the bits of the SCON register.



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. (Make it 0.)
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 hardware at the beginning of the stop bit in mode 1. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware halfway through the stop bit time in mode 1. Must be cleared by software.

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

Figure 2 SCON register

1.5 Use of Baud rate in 8051

Baud rate in 8051 is programmable. It is done with the help of Timer 1. Timer 1 must be programmed in mode 2, that is, 8-bit, auto-reload. We must make sure that the baud rate of the 8051 system matches the baud rate of the system used for communication.

Relationship between the crystal frequency and the baud rate in the 8051 is as follows:

- 8051 divides the crystal frequency by 12 to get the machine cycle frequency.
- Load FD/FA/F4/E8 in TH1 to get 9600/4800/2400/1200 baud rate.

1.6 Steps to send data serially in serial port communication

The following are the steps to send the data serially in serial port communication:

1. Set baud rate by loading TMOD register with the value 20H; this indicates 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 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. Transmit Interrupt(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 to see if the character has been transferred completely.
8. To transfer the next byte, go to step 5.

1.7 Importance of the TI flag:

The following steps are the importance of TI flag in 8051 serial port communication

- 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 instructions like “CLR TI”.
- Writing a byte into SBUF before the TI flag bit is raised, may lead to loss of a portion of the byte being transferred.

1.8 Steps to receive data serially in serial port communication

The following are the steps to receive the data serially in the serial port communication:

1. Set baud rate by loading TMOD register with the value 20H; this indicates setting of 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. Receive Interrupt(RI) is cleared by CLR RI instruction.
6. The RI flag bit is monitored 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.

1.9 Importance of the RI flag:

The following point to the importance of RI flag in 8051 serial port communication:

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

We will now look at programming these using Embedded C.

2. 8051 Embedded C Programming for Serial Port Communication

SFR registers of 8051 are accessible directly in 8051 C compilers by using reg51.h. It has already been described in the previous modules. Example 1 shows how to program the serial port in 8051 using Embedded C. For testing operation, HyperTerminal is used for this example.

Example 1

Write a C program for 8051 to transfer the letter “A” serially at 4800 baud continuously.

Use 8-bit data and 1 stop bit.
Solution:
```
#include <reg51.h>
void main(void){
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFA; //4800 baud rate
SCON=0x50;
TR1=1;
while (1) {
  SBUF='A'; //place value in buffer
while (TI==0);
  TI=0;
}
}
```

Example 2 described below gives the details about how to transfer the message serially with 9600 baud continuously.

Example 2

Write an 8051 C program to transfer the message "YES" serially at 9600 baud, 8-bit data, 1 stop bit. Do this continuously.

Solution:

```
#include <reg51.h>
void SerTx(unsigned char);
void main(void){
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFD; //9600 baud rate
SCON=0x50;
TR1=1; //start timer
while (1) {
  SerTx('Y');
  SerTx('E');
  SerTx('S');
}
}
void SerTx(unsigned char x){
SBUF=x; //place value in buffer
while (TI==0); //wait until transmitted
TI=0;
}
```

Example 3 described below gives the details about how to receive the data serially and put in Port P1 with 4800 baud and 1 stop bit. Here, Timer 1 in mode 2 is used for stopping the bit while receiving the data. The received data stored in the SBUF is then written to port P1.

Example 3

Program the 8051 in C to receive bytes of data serially and put them in P1. Set the baud rate at 4800; use 8-bit data, and 1 stop bit.

```
#include <reg51.h>
void main(void){
unsigned char mybyte;
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFA; //4800 baud rate
SCON=0x50;
TR1=1; //start timer
while (1) { //repeat forever
   while (RI==0); //wait to receive
            mybyte=SBUF; //save value
   P1=mybyte; //write value to port
   RI=0;
}
}
```

Example 4 described below gives the details about sending two messages to serial port with 28,800 and 56K baud. Here Timer 1 in mode 2 is used. The data is stored in the SBUF and then it waits to transmit to the corresponding destination.

Example 4
Write an 8051 C Program to send the two messages “Normal Speed” and “High Speed” to the serial port. Assuming that SW is connected to pin P2.0, monitor its status and set the baud rate as follows:
SW = 0, 28,800 baud rate
SW = 1, 56K baud rate
Assume that XTAL = 11.0592 MHz for both cases.
Solution:

```
#include <reg51.h>
sbit MYSW=P2^0; //input switch
void main(void){
unsigned char z;
unsigned char Mess1[]=“Normal Speed”;
unsigned char Mess2[]=“High Speed”;
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFF; //28800 for normal
SCON=0x50;
TR1=1; //start timer
if(MYSW==0) {
   for (z=0;z<12;z++) {
            SBUF=Mess1[z]; //place value in buffer
            while(TI==0); //wait for transmit
            TI=0;
   }
}
else {
PCON=PCON|0x80; //for high speed of 56K
   for (z=0;z<10;z++) {
            SBUF=Mess2[z]; //place value in buffer
```

```
             while(TI==0); //wait for transmit
             TI=0;
  }
 }
}
```

Example 5 described below gives the details about transferring the letter A serially with 4,800 baud continuously. Here Timer 1 in mode 2 is used to set the baud rate. The data is stored in the SBUF and new addresses for DS89C4x0 chip are declared using SFR registers using sfr keyword.

Example 5

Write a C program for the DS89C4x0 to transfer the letter "A" serially at 4800 baud continuously. Use the second serial port with 8-bit data and 1 stop bit. (note: We can only use Timer 1 to set the baud rate).

Solution:

```
#include <reg51.h>
sfr SBUF1=0xC1;
sfr SCON1=0xC0;
sbit TI1=0xC1;
void main(void){
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFA; //4800 baud rate
SCON1=0x50; //use 2nd serial port SCON1
TR1=1; //start timer
while (1) {
  SBUF1='A'; //use 2nd serial port SBUF1
  while (TI1==0); //wait for transmit
          TI1=0;
}}
```

Example 6 described below gives the details about receiving the data serially with 9,600 bauds via second serial port. Here Timer 1 in mode 2 is used to set the baud rate.

Example 6

Program the DS89C4x0 in C to receive bytes of data serially via the second serial port and put them in P1. Set the baud rate at 9600, 8-bit data and 1 stop bit. Use Timer 1 for baud rate generation.

Solution:

```
#include <reg51.h>
sfr SBUF1=0xC1;
sfr SCON1=0xC0;
sbit RI1=0xC0;
void main(void){
unsigned char mybyte;
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFD; //9600 baud rate
SCON1=0x50; //use 2nd serial port SCON1
TR1=1; //start timer
while (1) {
  while (RI1==0); //monitor RI1
```

```
   mybyte=SBUF1; //use SBUF1
   P2=mybyte; //place value on port
   RI1=0;
}
}
```

3. Summary

In this lecture, basics of serial communication is discussed. The 8051 control registers for serial communication are shown and the Serial Port programming in Embedded C for 8051 are discussed with examples.

4. References

1. Muhammad Ali Mazidi, Janice Gillispie Mazidi, Rolin D. McKinlay, “The 8051 Microcontroller and Embedded Systems Using Assembly and C -Second Edition”, New Delhi(2000).