

实验 8 串行通信技术

一、实验目的

1. 了解异步串行通信原理；
2. 掌握 MSP430 异步串行通信模块及其编程控制方法。

二、实验任务

1. 单片机串口与PC机USB虚拟串口连接后，PC机串口的检测

因外部32.768KHz晶振在扩展电路板上，需把扩展板接到单片机板上，才能使用该晶振，注意把JP8处晶振相关的跳线接至晶振侧，参见实验5的图5-1。


如图8-1，先将USB转串口模块的发送TxD、接收信号RxD、GND分别与扩展板上msp430G2553单片机串口的接收引脚P1.1(RxD)、发送引脚P1.2(TxD)、GND相连，**注意不要接错了!**，再将USB转串口模块插入PC侧的USB接口中，构成PC机利用USB虚拟串口与单片机串口进行通信的硬件接线。

由于串口助手软件有时会在运行中出现异常，造成不正常工作情况。故在做与单片机串口通信的实验中，还会需时不时通过自发自收检测PC机的USB转串口，在串口助手的控制下能否正确工作。

麻烦的是测试PC机USB转串口的自发自收，需要断开与单片机的连线，然后将USB转串口的TxD、RxD信号短接，测试完在断开短接线，然后再和单片机的串口接线。

此时可以用下面方法来测试PC侧USB转串口的自发自收：

如图8-2，在扩展板上，单片机的引脚P1.1/RxD和P1.2/TxD共有两处引出。其中一处以在图8-1中用于USB转串口收发信号TxD和RxD对接。如图8-3，只要将另一处中的P1.1和P1.2短接，则在硬件连线上，相当于USB转串口收发信号TxD和RxD短接了。接好线后，在CCS下进入DEBUG后，此时处于暂停状态，由于单片机上电或复位时P1.1和P1.2的初始设置为基本输入功能，这样在图8-3中，只有USB转串口模块的TxD向连接点发送数据(输出)，其他的引脚均从连接点为接收数据(输入)，满足电气上的信号方向不冲突。这样任何时刻，在DEBUG下，只要按菜单栏上的复位按钮，短接P1.1和P1.2，就可以按照任务1中给出方法测试PC串口自发自收，且单片机与USB转串口的线接上后也能随时检测。

当PC侧USB转串口检测成功后，断开图8-3中P1.1和P1.2间的短接线，回到图8-1单片机串口与PC机USB虚拟串口收发对接状态，进行PC机与单片机两者之间的串行通信。注意，在DEBUG下如果运行了的自己编写的串口控制程序，会将P1.1、P1.2设置为单片机的串口引脚，此时即使短接了P1.1和P1.2，当前信号关系参看图8-4，不满足电气上的信号关系，不能做PC侧串口自发自收的检测接线。如果想重新检测PC串口的自发自收，只需在CCS下按复位restart按钮，将P1.1、P1.2恢复到复位时的基本输入状态，然后可以检测。改线期间以及之后，不要断开USB数据线与单片机的连线，否则都需重测PC机USB虚拟串口的自发自收，确保PC机侧串口正常受控串口助手。

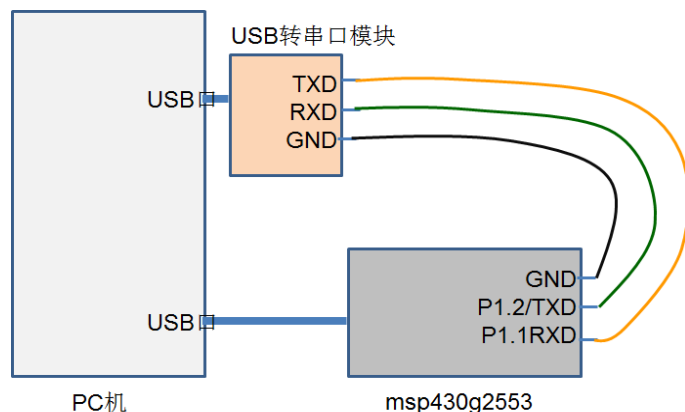


图8-1 单片机与PC机利用USB转串口进行通信

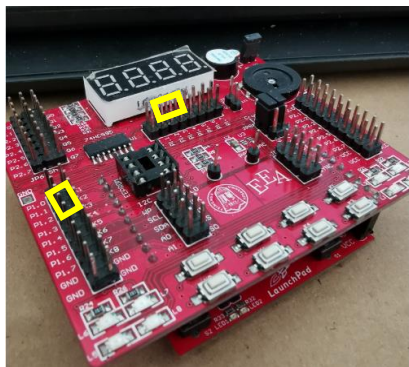


图8-2 扩展板上两处P1.1/RxD、P1.2/TxD

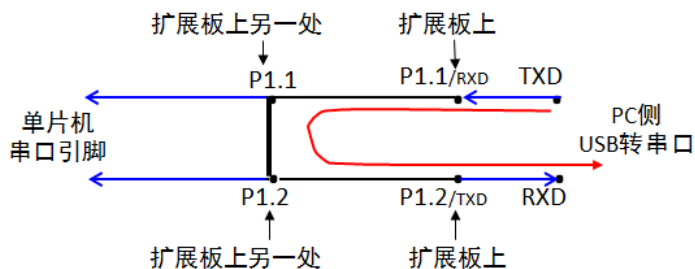


图8-3 短接**设为基本输入**的单片机P1.1、P1.2引脚与USB虚拟串口的TxD、RxD对接时的信号关系

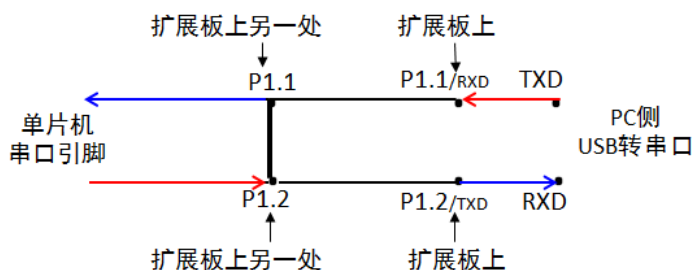


图8-4 短接**设为串行接口**的单片机P1.1、P1.2引脚与USB虚拟串口的TxD、RxD对接时的信号关系

2. 控制单片机与PC机实现异步串行通信

掌握了PC侧USB转串口模块是否正堂工作的检测方法后，按图8-1接好单片机与PC机的串口通信连线，卸下引脚P1.1与P1.2间的短线块。**完成下面内容：**

- 1) 阅读程序L8_TestSCI.c(提供电子版)，调试完成程序功能：设置数据格式为波特率9600bps、无校验、8位数据、先低后高、1个停止位，利于用单片机异步串行接口，**采用查询方式**，实现单片机与PC机之间的数据通信，掌握单片机串口通信编程和调试的基本方法。

步骤：

在PC机上运行串口助手，打开USB转串口的端口号，在CCS下将程序L8_TestSCI.c编译连接，进入到DEBUG下，按任务2的方法检测PC侧USB虚拟串口的自发自收成功后，运行程序。在串口助手的发送窗口输入5个字符，比如“12345”，**并按回车**，然后点击串口助手上的“手动发送”，发送区输入的数据经单片机串口接收后，又发送至PC机，并在串口助手显示出来。注意加上回车操作中包含的回车和换行两个字符（\r\n），实际输入了7个字符。串口助手发送的是这些字符对应的ASCII，即发送的是 0x31, 0x32, 0x33, 0x34, 0x35, 0x0d, 0x0a，其中0x0d、0x0a分别是回车符和换行符的ASCII码。如果在串口助手上，选择用十六进制发送，则应输入 31 32 33 34 35 0d 0a，中间需用空格分开，然后点击手动发送。选择用十六进制显示的话，看到接收到的数据应是 31 32 33 34 35 0d 0a，有时会遇到个别误码的情况，即收到的数据与发送的数据不一致，这要细查原因。

注意:

实验中单片机与PC机通信不成功,有时可能是PC侧的串口助手未能正常工作。可通过检测PC机的自发自收成功,确认PC机的串口受串口助手控制。PC侧正常工作,然后再细查单片机侧的问题。

- 2) 在当前接线基础上,添加接线,编程完成:在实验板上按下K1按键,单片机向PC机发送字符串“Get Ready!”;然后,当PC机向单片机分别发送单个字符“F”、“B”、“L”、“R”、“S”,实验板上的L1、L2两个发光二极管分别全亮、全闪、L1单个闪、L2单个闪、全灭。如PC机发送单个字符“F”,L1、K2全亮,如此类推。
- 3) **(提高)**在任务2)的基础上,加上小车的控制接线。当PC机向单片机分别发送单个字符“F”、“B”、“L”、“R”、“S”时,控制小车前进、后退、左转、右转、停转。

3. 利用蓝牙模块实现单片机与手机蓝牙通信

在任务2)的基础上,将单片机的串口引脚与蓝牙模块连接,改用手机上的蓝牙串口助手,利用蓝牙技术,完成手机与单片机的数据传送,将任务2)的有线通信变为无线通信。步骤:

- 1) 先断开msp430G2553实验小板与PC机的USB数据线;
- 2) 断开USB转串口模块与MSP430G2553单片机的连接,改用HC-05蓝牙模块的收/发信号与实验板上单片机的异步串口发/收信号对接,连线如图8-5;
- 3) 连接msp430G2553实验板与PC机的USB数据线;
- 4) 打开手机蓝牙,运行手机蓝牙串口助手,并与HC-05蓝牙模块配对;
- 5) 在单片机上分别运行任务2)的1)、2)、**(提高)**3)的程序,实现单片机与手机的蓝牙通信。

说明:单片机用HC-05蓝牙模块进行蓝牙通信时,可先按任务2)的方法,调试完成单片机与计算机的异步串行通信,然后再接上蓝牙模块,利用蓝牙模块的透传功能,完成将有线的异步串行通信,改为无线的蓝牙通信。通信出现问题时,按照实验7的任务和本次实验中介绍的调试方法,去查找各环节可能存在的问题。

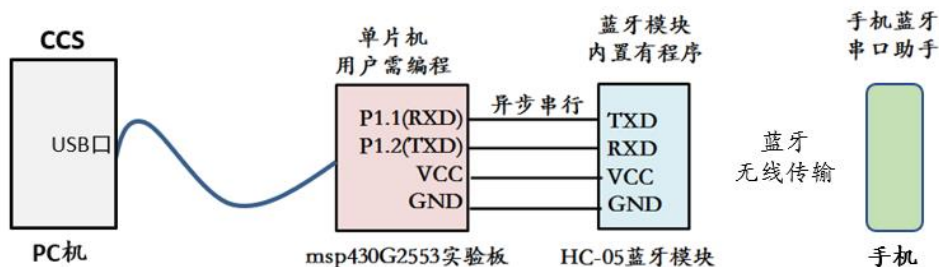


图8-5 单片机与手机蓝牙通信

4. (选做) 利用蓝牙模块实现单片机与电脑蓝牙通信

如实验7的选做任务4,如果手机蓝牙无法与蓝牙模块通信,电脑上有蓝牙的话,可以用电脑上的蓝牙代替手机上的蓝牙做任务3。接线如图8-6。步骤与上面的任务3相似,不同的只是用电脑蓝牙与HC-05蓝牙模块配对,用单片机串口程序控制蓝牙模块与电脑蓝牙通信。

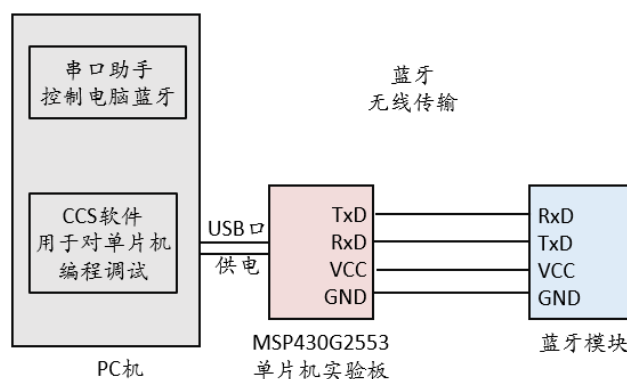


图8-6 单片机与电脑蓝牙通信

5. (提高) 中断方式控制串行通信的收发

编程：采用中断方式接收，查询方式发送完成任务2的LED控制。(可参看讲义中断例子)

6. (提高) 简单通信协议练习

将任务2的单个字符命令，放在以字符“AT+”开头和回车符之间，比如，发出“F”命令，改为输入“AT+F回车”。单片机侧接收到回车换行符后，才会分析收到的命令，判断接收到的字符串前3个字符为“AT+”，第5、6字符为回车换行符，再根据第4个字符的内容，完成相应的控制操作。即增加了简单的通信协议，单片机只将接收在“AT+”和回车换行符间的字符当作命令，而其他在“AT+”和回车换行符外的字符不予处理，避免了PC机误发、或通信误码，造成单片机误判情况。

7. (提高) 时钟信号对通信的影响

假设SMCLK上电复位频率为1MHz（DCO上电复位时钟值），改写L8_testSCI.c，选用上电复位后的SMCLK作为串口波特率时钟源，运行L8_testSCI.c，看通信是否成功。如果不成功，改用十六进制方式发送和显示，观察现象，说明原因。

L8_TestSCI.c

```
#include "msp430.h"
void UARTA0_init( );
char buffer[20],string[30]="Please input 7 characters:\r\n\0";
unsigned char j;
int main ( void )
{   WDTCTL = WDTPW + WDTHOLD;           //关闭看门狗
//做实验时，如果有示波器
//可在P1.0、P1.4引脚输出时钟ACLK、SMCLK，以便观察时钟频率,修改波特率寄存器设置
//   P1SEL |= BIT0+BIT4;
//   P1SEL2 |= ~( BIT0+BIT4);
//   P1DIR |= BIT0+BIT4;

    UARTA0_init( );                     //初始化串口
    while(1)
    {   j=0;
        while(string[j]!='\0')          //输出提示信息
        {   while((IFG2&UCA0TXIFG)==0);  //检测发送缓冲是否空
            UCA0TXBUF=string[j];         //取一个数据发送
            j++;
        };

        for (j=0; j<7; j++)              //接收字符串
        {   while((IFG2&UCA0RXIFG)==0);  //检测接收缓冲器是否满
            buffer[j]= UCA0RXBUF;         //接收一个数据并保存
        };

        for (j=0;j<7;j++)                //发送字符串
        {   while((IFG2&UCA0TXIFG)==0);  //检测发送缓冲是否空
            UCA0TXBUF=buffer[j];         //取一个数据发送
        };
    };

}

void UARTA0_init( )
{   UCA0CTL1|= UCSWRST;                  //置软件复位位swrst为1
    P1SEL |= BIT1+BIT2;                  //置P1.1、P1.2为串行接口收、发引脚功能
    P1SEL2 |= BIT1+BIT2;

    //数据格式选用上电复位设置：无校验，8位数据，1个停止位，异步串行通信
    UCA0CTL1|=UCSSEL0+UCRXEIE; //波特率时钟选择上电复位时的ACLK，32.768KHz，对错均收
    UCA0BR0 =3;                          //波特率9600
    UCA0BR1 = 0;
    UCA0MCTL=UCBRF_0+UCBRS_3;
    UCA0CTL1 &=~UCSWRST;                  //置软件复位位swrst为0，串口设置完毕
}
```