1 概述

GM812X 系列串口扩展芯片可为用户提供最简单和高性能的通用串口扩展方案，该系列芯片子串口最高波特率达 38400bps。该芯片提供两种工作模式，用户可根据需要灵活选择。该芯片母串口和子串口的工作波特率可由软件调节，而不需要修改外部电路和晶振频率。

GM812X 系列芯片的外部控制少，应用灵活，编程使用简单，适合于大多数需要多串口扩展的应用场合。

2 应用说明

2.1 硬件接口

GM812X 系列的典型应用如图 1 所示：

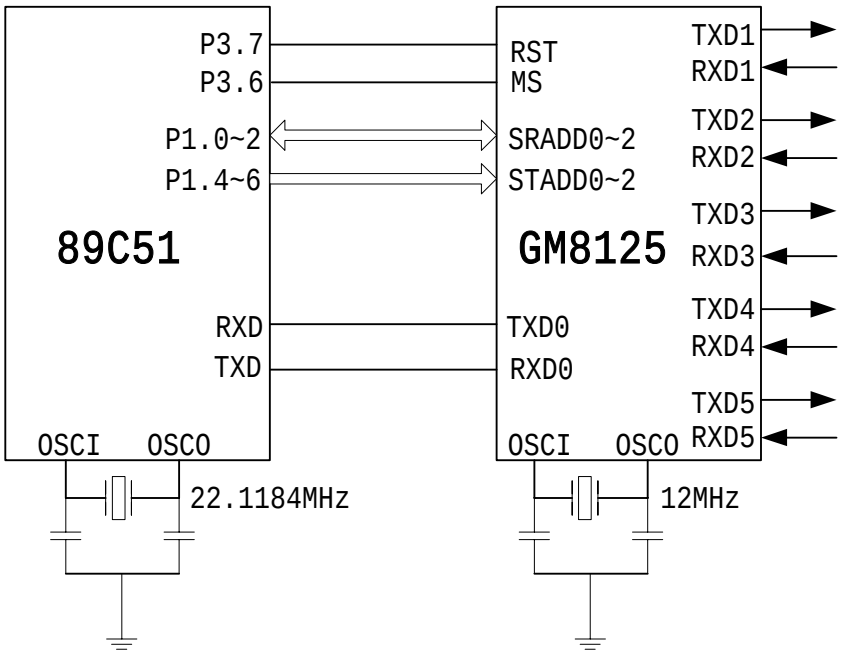

图 1 典型硬件接口电路

图 1 中选用 89C51 作为系统的主机，通过 GM8125 扩展了 5 个子串口，5 个子串口可以与 5 个从机相接。如果用户只需要扩展 3 个串口，则可采用 GM8123，硬件连接方法和 GM8125 相同。

2.2 程序示例

2.2.1 单通道工作模式程序示例

此程序应用的环境是 5 个从机分别以 1200、2400、4800、9600、19200bps 的波特率工作，并且主机与 5 个从机的数据通讯采用分时的方式，即每次只对一个从机发送和接收数据。程序以 C51 为例：

/***/

```
/*CPU：AT89C51                                                    */
/*晶体频率：22.1184MHz                                            */
/*机器周期：0.54uS                                          */
/*语言：C51                                    */
/*************************************************************/
#include <reg51.h>
#define    DELAY_TIME      60000                //Delay time
/***********************I/O 定义***********************/
sbit MS=P3^6;                                   //GM8125 工作模式控制
sbit RESET=P3^7;                                //GM8125 复位引脚控制
/******************数据变量定义*****************/
unsigned char SendBuff[5]={0xaa,0x45,0x67,0xbc,0xc9};
unsigned char ReceiveBuff[5]={0x00,0x00,0x00,0x00,0x00};
unsigned char i=0 ,j=0 ,k=1,c=0;
/***********************************************/
/*函数名称：delay.C                                    */
/*函数功能： 延时程序                          */
/***********************************************/
void delay(unsigned int m)
{
   unsigned int n;
   n=0;
   while(n < m)
   {n++;}
   return;
}
/*************************************************************/
/*函数名称： MAIN.C                                    */
/*函数功能： 主机主程序                                */
/***********************************************/
void main(void)
{
    TMOD = 0x20;       //指定定时器 1 工作在方式 2
    IE = 0x90;          //开串行口中断
    SCON=0xc0;       //串行口工作在方式 3

    for(c=0;c<5;c++)          //选择 5 个子串口以 5 种不同波特率工作
          {
             switch(c)
                   {
                       case 0:{
                                 TH1=0xd0;     //装入定时器 1 初值,设置工作波特率为 1200bps
                                 TL1=0xd0;
                                 PCON=0x00;
                                 P1=0x11;                  //选择 8125 子通道 1 工作
```

```
                    break;
                    }
                case 1:{
                    TH1 = 0xe8;       //装入定时器 1 初值,设置工作波特率为 2400bps
                    TL1 = 0xe8;
                    PCON=0x00;
                    P1=0x22;                    //选择 8125 子通道 2 工作
                    break;
                    }
                case 2:{
                    TH1 = 0xf4;       //装入定时器 1 初值,设置工作波特率为 4800bps
                    TL1 = 0xf4;
                    PCON=0x00;
                    P1=0x33;                    //选择 8125 子通道 3 工作
                    break;
                    }
                case 3:{
                    TH1 = 0xfa;       //装入定时器 1 初值,设置工作波特率为 9600bps
                    TL1 = 0xfa;
                    PCON=0x00;
                    P1=0x44;                    //选择 8125 子通道 4 工作
                    break;
                    }
                case 4:{
                    TH1 = 0xfd;     //装入定时器 1 初值,设置工作波特率为 19200bps
                    TL1 = 0xfd;
                    PCON=0x00;
                    P1=0x55;                    //选择 8125 子通道 5 工作
                    break;
                   }
                default:
                    break;
            }

        TR1=1;                  //启动定时器 1
        MS=1;               //GM8125 工作在单通道工作模式下

/*主控 MCU 发送/接收程序*/
        SBUF=SendBuff[i];
        while(TI==0);
        TI=0;
        i++;

        REN = 1;
        while(j!= k);           //等待接收完成
```

```
            REN = 0;          //停止接收
            k++;
            TR1=0;             //T1 停止
           }
       }


  void CommReceive(void) interrupt 4
{
    if(RI)
    {
        ReceiveBuff[j] = SBUF;
        RI = 0;
        j++;
    }
 }
```

2.2.2 多通道工作模式程序示例

此程序应用的环境是 5 个从机均以 19200bps 的波特率工作，要求主机对 5 个从机分别发送完数据后要等待从机向主机返回一个数据。程序以 C51 为例：

```
/*************************************************************/
/*CPU：AT89C51                                               */
/*晶体频率：22.1184M                                         */
/*机器周期：0.54us                                           */
/*语言：C51                                                  */
/*************************************************************/
#include <reg51.h>
#define DELAY_TIME  60000          //Delay time
#define DELAY_TIME1  1
/**********************I/O 定义**********************/
sbit MS=P3^6;                      //GM8125 工作模式控制
sbit RESET=P3^7;                   //GM8125 复位引脚控制
/******************数据变量定义*****************/
unsigned char SendBuff[5]={0xaa,0x45,0x67,0xbc,0xc9};
unsigned char ReceiveBuff0;
unsigned char ReceiveBuff1;
unsigned char ReceiveBuff2;
unsigned char ReceiveBuff3;
unsigned char ReceiveBuff4;
unsigned char ReceiveBuff5;
unsigned char Contr_data;
unsigned char ADD;
```

```
unsigned char i=0 ,j1=0 ,j2=0 ,j3=0 ,j4=0 ,j5=0 ,c=0;
/*********************************************/
/*函数名称：delay.C                          */
/*函数功能： 延时程序                        */
/*********************************************/
void delay(unsigned int m)
{
   unsigned int n;
   n=0;
   while(n < m)
   {n++;}
   return;
}
/*****************************************************/
/*函数名称： MAIN.C                                  */
/*函数功能： 主机主程序                              */
/*****************************************************/
void main(void)
{
   TMOD = 0x20;       //指定定时器 1 工作在方式 2
   IE = 0x90;          //开串行口中断
   SCON=0xc0;          //串行口工作在方式 3
   TH1 = 0xf8;         //装入定时器 1 初值,设置主机工作波特率为 7200bps
   TL1 = 0xf8;
   PCON=0x00;

   RESET=0;        //对 GM8125 进行复位操作
   RESET=1;
   delay(DELAY_TIME);

   Contr_data=0xfc;                //装入命令字初值
   TR1=1;          //启动定时器 1
   MS=0;           //GM8125 工作在写命令字模式下
   P1=0x00;        //置 GM8125 命令字地址
   SBUF=Contr_data;       //设置 GM8125 子串口波特率为 19200bps，母串口波特率为 115200bps
   while(TI==0);
   TI=0;
   delay(DELAY_TIME);
   REN=1;
   MS=1;          //读命令字
   delay(DELAY_TIME);
   while (ReceiveBuff0!=Contr_data)        //验证写入的命令字是否正确，不正确则重新写
         {
            MS=0;           //GM8125 工作在写命令字工作模式下
            SBUF=Contr_data;
```

```
            while(TI==0);
            TI=0;
            delay(DELAY_TIME);
            REN=1;
            MS=1;          //读命令字
            delay(DELAY_TIME);
          }
    REN=0;
    MS=0;           //设置 GM8125 工作在多通道工作模式下
    TR1=0;           //定时器 1 停止

    TH1 = 0xff;      //装入定时器 1 初值,设置主控 MCU 工作波特率为 115200bps
    TL1 = 0xff;
    PCON=0x80;
    TR1=1;             //启动定时器 1

/*主控 MCU 发送/接收程序*/
    ADD=0x1f;            //子通道 1 发送地址
    P1=ADD;                    //选择 GM8125 子通道 1 发送
    delay(DELAY_TIME1);        //GM8125 命令字设置更新时间
    SBUF=SendBuff[i];
    while(TI==0);
    TI=0;
    i++;

    ADD=0x2f;            //子通道 2 发送地址
    P1=ADD;                    //选择 GM8125 子通道 2 发送
    SBUF=SendBuff[i];
    while(TI==0);
    TI=0;
    i++;

    ADD=0x3f;            //子通道 3 发送地址
    P1=ADD;                    //选择 GM8125 子通道 3 发送
    SBUF=SendBuff[i];
    while(TI==0);
    TI=0;
    i++;

    ADD=0x4f;            //子通道 4 发送地址
    P1=ADD;                    //选择 GM8125 子通道 4 发送
    SBUF=SendBuff[i];
    while(TI==0);
    TI=0;
    i++;
```

```
    ADD=0x5f;              //子通道 5 发送地址
    P1=ADD;                      //选择 GM8125 子通道 5 发送
    SBUF=SendBuff[i];
    while(TI==0);
    TI=0;
    i++;

    REN = 1;
    while(j1!= 1);          //等待接收完成
    while(j2!= 1);          //等待接收完成
    while(j3!= 1);          //等待接收完成
    while(j4!= 1);          //等待接收完成
    while(j5!= 1);          //等待接收完成
    REN = 0;          //停止接收
}


    void CommReceive(void) interrupt 4
{
      if(RI)
      {
          switch(P1&0x07)
              {
                 case 0:
                      {ReceiveBuff0=SBUF;     //读命令字存入 ReceiveBuff0
                       break;
                       }
                 case 1:
                      {ReceiveBuff1=SBUF;     //子通道 1 接收的数据存入 ReceiveBuff1
                       j1++;
                       break;
                      }
                 case 2:
                      {ReceiveBuff2=SBUF;    //子通道 2 接收的数据存入 ReceiveBuff2
                       j2++;
                       break;
                      }
                 case 3:
                      {ReceiveBuff3=SBUF;    //子通道 3 接收的数据存入 ReceiveBuff3
                       j3++;
                       break;
                      }
                 case 4:
                      {ReceiveBuff4=SBUF;    //子通道 4 接收的数据存入 ReceiveBuff4
```

```
                    j4++;
                    break;
                   }
              case 5:
                   {ReceiveBuff5=SBUF;     //子通道 5 接收的数据存入 ReceiveBuff5
                    j5++;
                    break;
                   }
              default:
                    break;
             }
         RI = 0;
      }
}
```

2.3 应用说明

1、 GM8123 的应用程序可参考 GM8125；
2、 以上程序仅作为用户使用的参考程序，不作为 GM812X 系列芯片的固定使用程序；
3、 用户在使用 GM812X 系列芯片时请注意遵照数据手册中的使用说明进行操作；
4、 根据实际测试经验，用户若使用 51 单片机控制，需要在写入命令字后立即读取命令字，由于 51 单片机串口工作时序是发送停止位的同时将 TI 置 1，如果此时立即将 MS 置 1 读取命令字，将导致芯片还未接收到命令字字节的停止位，芯片就转入读命令字模式，使写入操作无法完成，所以编程时要注意在 TI 置 1 后至少延时 1bit 的时间才能将 MS 置 1，以保证芯片有足够的时间完成写命令字操作；
5、 GM812X 从母串口完成数据接收到将此数据发送到子串口有一个子口位的延时，所以主机发完停止位到子口发完停止位的时间为：一个子口位的时间 + 一个子口帧的时间，计算公式为：（1+dn）/br，其中 br 为当前设置的子串口波特率，dn 为当前设置的子串口一帧的位数（10 或 11）。**用户采用 485 通讯时需要遵照此时间进行控制**；
6、 多通道模式下，根据数据手册上的描述“ 完成命令字的设置之后，必须将 STADD 置为非全 0 的值后，设置才生效 ”，由于从 STADD 置为非全 0 值到设置更新还有一定的延时，如果用户程序是在写完命令字后 STADD 的全 0 状态一直保持到需要发送数据时才将地址线修改，修改地址线状态后立即向芯片发送数据，将有可能出现第一个字节错误的情况，建议在设置完地址线后给予几微秒的延时，以保证芯片有足够的时间完成设置更新操作。或者用户可以在设置完命令字后立即修改 STADD 为非全 0 状态，使新设置在芯片通讯之前完成生效。