MSP430 单片机应用进阶

MSP430 的 Timer_A 实现模拟串口功能例程 (V1.1)

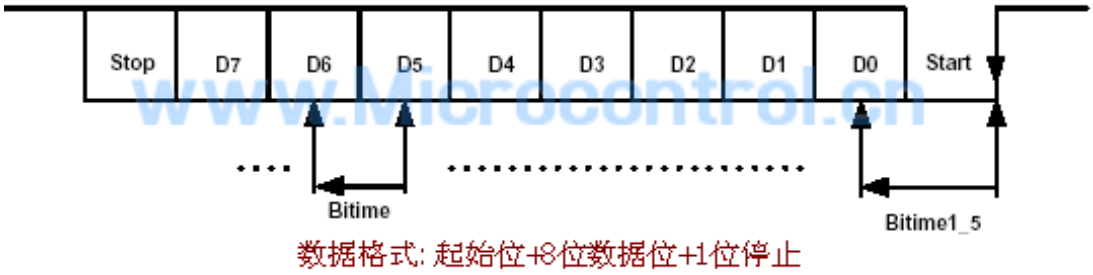
策划：微控设计网 DC

1 简述

本例程是用 MSP430F1121 单片机的 Timer A（以下简称 TA）模块实现简单的串行口功能。有兴趣的朋友们可以将此制作来当一个 430DIY 实验练练功。能让你更进一步了解 TA 模块的功能好处，有实验过程中，如有什么问题可以到微控技术论坛上来论坛 http://bbs.microcontrol.cn

在本例中，波特率发生器是以 32768 晶体振荡器 (ACLK) 作为 TA 的时钟源输入 (ACLK=TACLK)，每位时钟周期约为 30us。若采用 2400B/S 作为波特率收发输出，那么传输出一位的时间周期为：1/2400=417uS。此时需要 TA 产生一个定时约为 417uS 的中断来作波特率发器。

本例所实现串口数据格式如下图：



上图是本程的串口数据格式：分别由 1 位起始位、8 位数据位、1 位停止位组成。

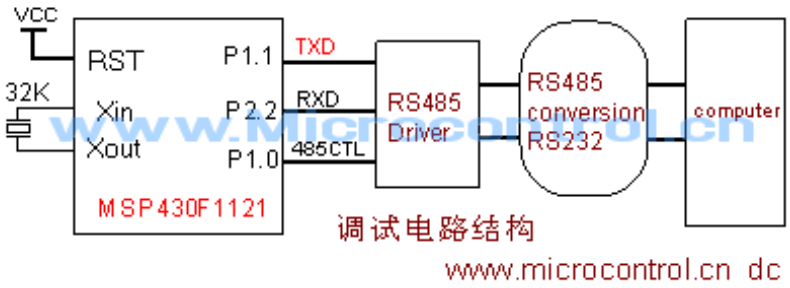
起始位：是用于识别一个串行数据的首要位标识。

数据位：8 位数据。

停止位：表示结束串行数据。

在 F1121 中用 TA 作波特率发生器以及利用捕获比较寄存器实现边沿捕获功能等软件来组合实现的。从而可以体现出 TA 模块的灵活性。

2 电路图



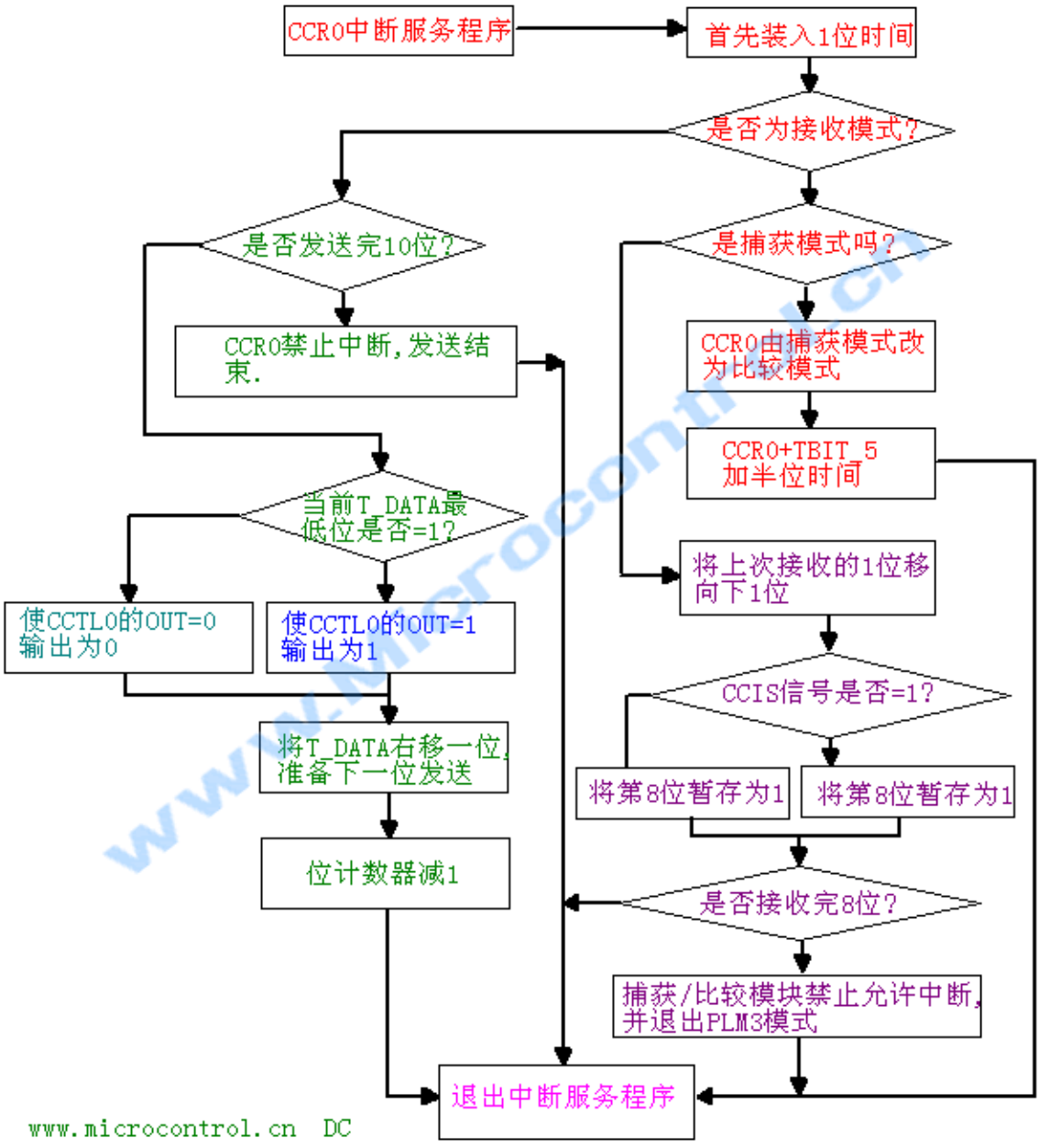
本次实验总体结构图

本实验时我用上了，RS485 作为驱动器与 PC 的通讯，在软件上与 RS232 并没有太大的区别，所以这部分不将多说。

3 主要软件流程图

在程序中，主程先将接收准备程序设置好，进入低功耗模式中等待串行口发来的起始位信号进行唤醒。在接收中主要利用 CCR0 的捕获功能和比较功能实现；而在发送中主要利用 CCR0 的比较功能实现。此时 CCR0 的中断就当波特率发生器之用。在软件中，只要设置好相应的初始化，大部分是由中断程序完成的。

下面是 CCR0 中断程序流程图，供大家参考：



4 源程序

```
//file mane: serial.c ver:1.0
//利用定时器 A 作串行口波特率发生器用,利用捕捉比较功能实现异步串行通信。
//芯片型号：MSP430F1121  P1.1---TXD  P2.2---RXD  (485 时,P1.0 为 485 收发控制)
//Timer_A TACLK=ACLK
//波特率为 @4800BIT/S TBIT1=7 TBIT_5=3 ,  @2400BIT/S TBIT1=14 TBIT_5=6
//软件软件环境：IAR WE430 3.4A
//编写:http://www.microcontrol.cn  DC

//采用 1 位停止位+8 位数据位+1 位起始位(左最低位)=10 位方式发送.
#include "msp430x11x1.h"                    //头文件

#define TBIT1 14;                           //TBIT1 为 1 位时间
#define TBIT_5 6;                           //TBIT_5 为半位时间
#define TXD 0x0002
#define RXD 0x0004
#define TEN 0x0001                          //发送充许位
#define DE485 0x0001                        //485 充许发送控制

unsigned char GBIT=0x00;                    //通用位区
unsigned int TR_COUNT;                      //发收计数器
unsigned int T_DATA=0x00;                   //发送缓冲器
unsigned int R_DATA=0x00;                   //接收缓冲器

void init(void);                            //初始化
void txd (unsigned char byte);              //发送一字节数据
void rxd(void);                             //接收准备子程序

//*************************************************************************************
//主程序
main()
{
 init();                                    //初始化
 P2DIR|=BIT0;//P2.0 输出
 P2OUT|=BIT0;//先关 LED
while(1)
 {
 rxd();                                     //准备接收
 LPM3;
 txd(R_DATA);
 }
}

//*************************************************************************************
void init(void)                             //初始化
{
 WDTCTL=WDTPW+WDTHOLD;      //停止 WDT

 P1DIR|=BIT0;                               //P1.0 设为输出
```

```
  P1OUT&=~BIT0;                    //设 DE485=0;--485 接收充许

  P1DIR|=TXD;                      //P1.1 设为输出--TXD
  P1SEL|=TXD;                      //P1.1 设为使用外围模块
  P2DIR&=~RXD;                     //P2.2 设为输入--RXD
  P2SEL|=RXD;                      //P2.2 设为使用外围模块

  CCTL0|=OUT;                      //CCR0 的 OUT 设 1 (OUT 定义为 0x0004)

  TACTL|=TASSEL0+MC1;              //设定时器 A 时钟源 ACLK.
                                   //设定时器为连续模式,同时启动计时
  _EINT();                         //开中断允许
}


//*************************************************************************************
//发送一字节数据子程序
//输入参量:unsigned char 类型字节
void txd(unsigned char byte)
{     T_DATA=byte;
      CCR0=TAR;                     //将 TAR 时间存入 CCR0,确定第一位的长度。
      CCR0=CCR0+TBIT1;              //将每 1 位时间周期加入 CCR0。
      T_DATA=T_DATA<<1;             //将字节数据向左移一位,构造最低位为起始位。
      T_DATA=T_DATA|0x0200;         //将字的第 10 位设为 1,以作停止位使用。
      TR_COUNT=10;                  //发送计数器。
      P1OUT|=DE485;                 //P1.0=1,使 485 驱动器为发送。

      CCTL0=OUTMOD0+CCIE;           //重新设置 CCTL0(CCIS1-0=00)
                                    //捕获/比较模块输出模式 1,充许模块中断。
      while(CCIE&CCTL0);            //等待 CCIE 是否为 0?为 0 则表示发送完数据。

}
//*************************************************************************************
//接收准备子程序
//依赖 TA0 中断来接收一字节数据。
void rxd(void)
{TR_COUNT=8;                              //接收数据位的位数
 CCTL0=OUTMOD0+CCIE+CM1+CAP+CCIS0+SCS; //设置比较捕获控制寄存器 0 的设置
 //比较捕获为输出模式+比较捕获模块为中断允许+下降沿捕获+设置为捕获模式
 //+选择 CCI0B 为捕获源+同步捕获
 P1OUT&=~DE485;                     //485 接收充许  DE485=0,RS232 使用时可以不使用
}
//*************************************************************************************
#pragma vector=TIMERA0_VECTOR     //TIMER_A 中断函数
__interrupt void cc10int(void)
{
 CCR0=CCR0+TBIT1;                  //重装下一位时间(当前时间+1 位时间)

 //-------------------------接收处理程序段
```

```
if(CCIS0&CCTL0)                    //是处于接收中还是发送中?
  {if(CCTL0&CAP)                   //是捕获模式还是比较模式?
    { CCTL0&=~CAP;                 //是-开始捕获,将捕获功能改为比较功能
      CCR0=CCR0+TBIT_5;            //开始捕获位再加半位时间
    }
   else
    {R_DATA=R_DATA>>1;             //处于比较功能,将前面的那位向低位移.
     if(CCTL0&SCCI)                //捕获/比较块输入信号 SCCI 位是 1 还是 0?
        R_DATA|=0x80;              //SCCI 位是 1,将第 8 位置 1.否则第 8 位为 0.
     TR_COUNT--;                   //计数器减 1
     if(TR_COUNT==0)               //是否接收完 8 位?
       {
         CCTL0&=~CCIE;             //是接收完,捕获/比较块停止中断充许
         LPM3_EXIT;                //退出低功率模式(一般,在进入 LPM3 时才使用)
       }
    }
  }                                //接收结束

  //-------------------------------------------------------------
 else                                //开始发送程序段
  {
   if(TR_COUNT==0)
     CCTL0&=~CCIE;                 //关捕获/比较块中断,发送结束.
   else
     {if(T_DATA&0x0001)            //状态寄存器 C 位是 1,还是 0?
        CCTL0&=~OUTMOD2;           //状态寄存器 C 位为 1,发送 1.
      else
        CCTL0|=OUTMOD2;            //状态寄存器 C 位为 0,发送 0.
      T_DATA=T_DATA>>1;            //将字节数据向右移一位
      --TR_COUNT;                  //位计数器减 1.
     }
  }
}
```

5 理解

所用到的子程分别如下:

```
void init(void);                   //初始化
void txd (unsigned char byte);     //发送一字节数据
void rxd(void);                    //接收准备子程序
```

在 init()中,其中用于设置 MSP430 的 IO 引脚模块功能.

```
 P1DIR|=TXD;                       //P1.1 设为输出--TXD
 P1SEL|=TXD;                       //P1.1 设为使用外围模块
 P2DIR&=~RXD;                      //P2.2 设为输入--RXD
 P2SEL|=RXD;                       //P2.2 设为使用外围模块

 CCTL0|=OUT;                       //CCR0 对应的输出功能输出端为 1
```

TACTL|=TASSEL0+MC1; //设置 Timer_A 的工作模式

在设定时器为连续模式，并于 CCR0 为计算周期,此时也启动定时器工作开始工作，此时定时器 A 虽然工作了，但还没有初值并且还未能允许定时中断.在此,初始化完成。
从主程序可以看出,程序先等待一个字符的接收到来然后再发回出去。

```
main()
{
 init();                    //初始化

 while(1)
 {
  rxd();                    //准备接收
  LPM3;                     //进入低功耗，等待字符接收
  txd(R_DATA);              //发送一个字符
 }
}
```

RXD()解释：
在 while 结构中，开始调用 RXD()程序.此程序主要用于设置接收串口所发来的数据做准备。
具体如下：
TR_COUNT=8; //接收数据位的位数
用 TR_COUNT 变量来存放计数串行口应接收多位的数据,通常为 8 位.所以初值设置为 8。

首先,我们要知道串口在空闲时电平是处于高电平的.那么串行口的数据到来时,第一个到达的是起始位.此起始位为低电平。 那么,单片机要通过边沿捕获的方法来捕获到此下边沿的到来。在这里,我们可以通过 Timer_A 模块中的 CCR0 子模块功能来捕获此下边沿.具体设置如下:
CCTL0=OUTMOD0+CCIE+CM1+CAP+CCIS0+SCS; //设置比较捕获控制寄存器 0 的设置
OUTMOD0：捕获比较寄存器 CCR0 的子模块输出方式为输出模式
CCIE ：捕获比较寄存器 CCR0 子模块为中断允许
CM1 ：捕获比较寄存器 CCR0 子模块为下降沿捕获
CAP ：捕获比较寄存器 CCR0 子模块为捕获模式
CCIS0 ：捕获比较寄存器 CCR0 子模块选择 CCI0B 为捕获源(P2.2)
SCS ：捕获比较寄存器 CCR0 子模块为定时器时钟同步捕获

到此为止,MSP430 单片机已设置好准备接收字符数据功能。同时,程序进入低功耗模式 LPM3;此时，CCR0 模块接收到下边沿信号就可以进入中断进入识别处理。

```
#pragma vector=TIMERA0_VECTOR
__interrupt void cc10int(void)
```

TIMER_A 中断函数的接收处理解释:
当 CCR0 子模块产生中断(CCR0 有独立中断向量)时,会将一位时间装入 CCR0 中.表示不管怎样定时器 A 最低程度也要以 1 位时间产生中断.
CCR0=CCR0+TBIT1; //重装下一位时间(当前时间+1 位时间)
CCR0+TBIT1 的做法原因是以当前 CCR0 的时间值加下一位中断的时间值.因为 CCR0 一直都在计数的,所以必需 CCR0+TBIT1.

```
if(CCIS0&CCTL0)        //CCIS0 是否为 1?(是否 CCI0B 为捕获源?)
```

用 CCIS0 位来识别此时属于接收还是发送中。

处于接收字符:
if(CCTL0&CAP)
此句用于识别此时是不是为边沿捕获功能.如果是则认为此时串行口的RXD信号上出现第一次下降边沿.那么,此位就是串行口数据的初始位。
此时下降沿到了,也就是说在接下来过 1.5 倍的位时间后就是数据位了。

```
CCTL0&=~CAP;                //由边沿捕获改为比较功能
CCR0=CCR0+TBIT_5;           //从一位时间再加半位时间,因为起始位时间为 1.5 倍.
```

如果为比较模式,则此时接收的都是数据.

```
 else
    {R_DATA=R_DATA>>1;        //处于比较功能,将上次的那位移向低 1 位.
     if(CCTL0&SCCI)           //捕获/比较块输入信号 SCCI 位是 1 还是 0?
        R_DATA|=0x80;         //SCCI 位是 1,将第 8 位置 1.否则第 8 位为 0.
     TR_COUNT--;              //收接计数器指针减 1,用于指示位的数据
     if(TR_COUNT==0)          //是否接收完 8 位?
       {
         CCTL0&=~CCIE;        //是接收完,捕获/比较模块停止中断
         LPM3_EXIT;           //退出低功率模式
       }
    }
```

退出低功耗模式,TXD()程序将发送回字符出去:

```
    T_DATA=byte;
    CCR0=TAR;                       //将 TAR 当前时间存入 CCR0
    CCR0=CCR0+TBIT1;                //将每 1 位时间周期加入 CCR0 中
    T_DATA=T_DATA<<1;               //将字节数据向左移一位,构造最低位为起始位
    T_DATA=T_DATA|0x0200;           //将字的第 10 位设为 1,以作停止位使用
    TR_COUNT=10;                    //发送计数器
```

程序是先将要发送的数据发到数据缓冲器 T_DATA,然后将当前的 TA 时间+1 位的时间当作为下一次定时中断的时间。将缓冲器 T_DATA 左移一位是将产生第一位为 0.此位用作串口发送时的起始位.缓冲器 T_DATA 与 0x0200 相或的目的是为了在数据的第 10 位产生一个位为 1 的停止位.
TR_COUNT 是用来计数发送时要发送多少位位数据。

```
    CCTL0=OUTMOD0+CCIE;         //重新设置 CCTL0(CCIS1-0=00)
                                //捕获/比较模块输出模式,充许模块中断
    while(CCIE&CCTL0);          //等待 CCIE 是否为 0?为 0 则表示发送完数据
```

主要重新设置了 CCR0 模块的为输出模式,同时允许 CCR0 为比较中断.接着程序将要等待一个数据的发送完成.
发送完成时将退出中断.
在发送时,通过中断定时来发送每一位数据.通过 if(CCIS0&CCTL0)语句来进入发送处理.

```
if(TR_COUNT==0)
      CCTL0&=~CCIE;             //关捕获/比较块中断允许,发送结束.
   else
     {if(T_DATA&0x0001)         //状态寄存器 C 位是 1,还是 0?
        CCTL0&=~OUTMOD2;        //状态寄存器 C 位为 1,发送 1.
      else
```

```
    CCTL0|=OUTMOD2;          //状态寄存器 C 位为 0,发送 0.
T_DATA=T_DATA>>1;            //将字节数据向右移一位
--TR_COUNT;                  //位计数器减 1.
```

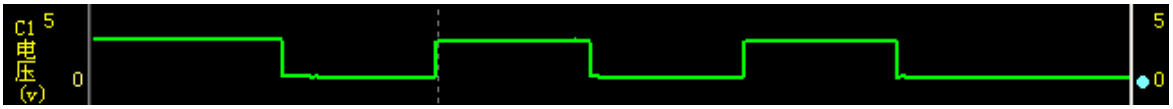
发送主要将 T_DATA 数据每一位的值通过 CCR0 的 OUT 引脚发送出去，其每一位时间间隔都是固定的，就这样,直到发送完十位为止。

6 实验结果

以下是用虚拟示波器所测量到的调试部分信号波形：

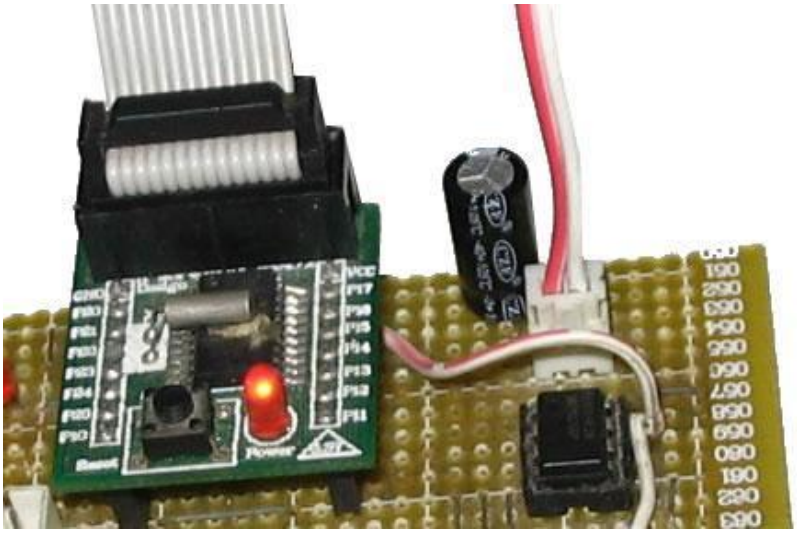
RXD 信号

TXD 信号



使用虚拟示波对串口所生产的波形进行了分析，结果表明是符合程序要求的。
MSP430F1121 模拟串口程序通过上位机串口调试软件的调试并证实收发数据正确，到此实验基础完成。

以下是此次 DIY 实验的实物图：



主要由 MSP430F1121 小最系统板与 SN75176 RS485 驱动芯片电路组成