FT6XF2XX

Application note

目录

FT6XF2XX UART 应用

1 UART 介绍

UART是一种通用串行数据总线，用于异步通信。该总线双向通信，可以实现全双工传输和接收。

UART 协议，其中各位的意义如下：

起始位：先发出一个逻辑”0”的信号，表示传输字符的开始。

资料位：紧接着起始位之后，资料位的个数可以是 4、5、6、7、8 等，构成一个字符。通常采用 ASCII 码，从最低位开始传送，靠时钟定位。

奇偶校验位：资料位加上这一位后，使得“1”的位数应为偶数(偶校验)或奇数(奇校验)，以此来校验资料传送的正确性。

停止位：它是一个字符数据的结束标志。可以是 1 位、1.5 位、2 位的高电平。 由于数据是在传输线上定时的，并且每一个设备有其自己的时钟，很可能在通信中两台设备间出现了小小的不同步。因此停止位不仅仅是表示传输的结束，并且提供计算机校正时钟同步的机会。适用于停止位的位数越多，不同时钟同步的容忍程度越大，但是数据传输率同时也越慢。

空闲位：处于逻辑“1”状态，表示当前线路上没有资料传送。

波特率：是衡量资料传送速率的指标。表示每秒钟传送的符号数（symbol）。一个符号代表的信息量（比特数）与符号的阶数有关。例如资料传送速率为 120 字符/秒，传输使用 256 阶符号，每个符号代表 8bit，则波特率就是 120baud，比特率是 120*8=960bit/s。

2 UART 相关设置

本例子是模拟UART发送部分，使用定时器让TXIO产生与设定好的波特率一样频率的信号；接收部分，使用RXIO电平转换中断识别起始信号，然后再根据定时器扫描数据识别数据内容。

讲解以IC FT60F011A SOP8为示范，在收到电脑端的串口助手发来的数据后，又把同样的数据发出去。

本程序数据线TX与RX所对应的IO引脚：

```
#define   TXIO        RA4

#define   RXIO        RA2
```

3 应用范例

```
//************************************************************
/*   文件名：Test_62F21X_UART.c
*    功能：       FT62F21X_UART 功能演示
*    IC:          FT62F21X SOP8
*    晶振：       16M/4T
```

```
*     说明：    演示程序中波特率为 9600，RXIO（PA0）每次收到外部串口发过来的数据后，TXIO(PA4)
*               把收到的数据再发送出去。接收起始位时是用电平变化中断识别，后面关闭电平变化中断。?
* Memory: Flash 1KX14b, EEPROM 128X8b, SRAM 64X8b
*                                   FT62F21X SOP8
*                                   ----------------
*   DemoPortOut ---------- |1(PA4)              (PA3)8 |----------NC
*   NC---------------------|2(TKCAP)            (PA0)7 |----------NC
*   NC---------------------|3(VDD)              (PA1)6 |----------NC
*   NC---------------------|4(VSS)              (PA2)5 |----------DemoPortIn
*                                   ----------------
*/
//============================================================
//============================================================
#include  "SYSCFG.h";
#include  "FT62F21X.h";

// 宏定义
#define   uchar         unsigned char
#define   uint          unsigned int
#define   ulong         unsigned long

#define   TXIO          RA4     //串口的发送脚
#define   RXIO          RA2     //串口的接收脚

#define   Bord          49      //通过定时器提供波特率

//============================================================
//
//    系统时钟
//============================================================
#define OSC_16M   0X70
#define OSC_8M    0X60
#define OSC_4M    0X50
#define OSC_2M    0X40
#define OSC_1M    0X30
#define OSC_500K 0X20
#define OSC_250K 0X10
#define OSC_32K   0X00

//============================================================
//变量定义
//============================================================
```

```
uchar RXFLAG = 0;
uchar ReadAPin;

/*---------------------------------------------
 *   函数名：  interrupt ISR
 *   功能：    中断处理，包括定时器 0 中断和外部中断
 *   输入：    无
 *   输出：    无
 *   说明：    定时器产生 104uS 中断，对应 9600 的波特率  1000000÷9600=104
 ---------------------------------------------*/
void interrupt ISR(void)
{

  //定时器 0 的中断处理**********************
    if(T0IE && T0IF)              //104us
    {
        TMR0 = Bord;              //注意:对 TMR0 重新赋值 TMR0 在两个周期内不变化

        T0IF = 0;
        T0IE = 0;
    }

    //PA 电平变化中断**********************
    if(PAIE && PAIF)
    {
        ReadAPin = PORTA;     //读取 PORTA 数据清 PAIF 标志
        PAIF = 0;             //清 PAIF 标志位
        if(RXIO == 0)
        {
            PAIE = 0;         //暂先禁止 PA 电平变化中断
            IOCA2 =0;         //禁止 PA2 电平变化中断
            RXFLAG = 1;
        }
    }
}
/*---------------------------------------------
 *   函数名：POWER_INITIAL
 *   功能：    上电系统初始化
 *   输入：    无
 *   输出：    无
 ---------------------------------------------*/
void POWER_INITIAL (void)
{
    OSCCON = OSC_16M;             //bit7 Timer2  选择 LIRC 为时钟源时  LIRC 的频率选择
```

```
                                    //0:32KHz      1:256KHz
                                    //bit[6:4]  系统频率选择
                                    //bit[2]高速内部时钟状态    1:ready   0:not ready
                                    //bit[1]低速内部时钟状态    1:ready   0:not ready

    INTCON = 0;                     //暂禁止所有中断
    OPTION = 0;
    TRISA   = 1<<2;                 //1:输入  0:输出
    PSRCA   = 0;                    //00: 4mA 01/10:8mA    11:28mA
                                    //bit[3:2]控制 PA5 源电流 bit[1:0]控制 PA4 源电流
    PSINKA = 0;                     //bit[1:0]  控制 PA5 和 PA4   0:灌电流最小 1 灌电流最大
    PORTA   = 0;                    //1:PAx 输出高电平      0:PAx 输出低电平
    WPUA    = 1<<2;                 //1:  使能 PA 口上拉    0:关闭 PA 口上拉

}
/*-------------------------------------------------
 *   函数名：TIMER0_INITIAL
 *   功能：    初始化设置定时器
 *   输入：    无
 * 输出：    无
 *   说明：    设置 TMR0 定时时长 104us=(1/16000000)*4*2*208(16M-2T-PSA  1:2-  TMR0=255
溢出)
 -------------------------------------------------*/
void TIMER0_INITIAL (void)
{
    OPTION = 0B00000000;            //Bit5 T0CS Timer0 时钟源选择
                                    //1-外部引脚电平变化 T0CKI 0-内部时钟(FOSC/2)
                                    //Bit4 T0CKI 引脚触发方式  1-下降沿  0-上升沿
                                    //Bit3 PSA  预分频器分配位  0-Timer0 1-WDT
                                    //Bit2:0 PS2 8 个预分频比  000 - 1:2
    TMR0 = Bord;
    T0IF = 0;                       //清空 T0 软件中断
    T0ON = 1;
}

/*-------------------------------------------------
 * 函数名: PA0_Level_Change_INITIAL
 *   功能：    PA 端口(PA2)电平变化中断初始化
 * 输入：    无
 * 输出：    无
-------------------------------------------------*/
void PA2_Level_Change_INITIAL(void)
{
    TRISA2 =1;                      //SET PA2 INPUT
```

```
    ReadAPin = PORTA;              //清 PA 电平变化中断
    PAIF =0;                       //清 PA INT 中断标志位
    IOCA2 =1;                      //使能 PA2 电平变化中断
    PAIE =1;                       //使能 PA INT 中断
  //GIE =1;                        //使能全局中断
}

//
/*-------------------------------------------------
 * 函数名：  WaitTF0
 * 功能：    查询定时器溢出后，在中断里关闭定时器后，再次打开定时器
 * 输入：    无
 * 输出：    无
 -------------------------------------------------*/
void WaitTF0( void )
{
     while(T0IE);
     T0IE=1;
}

/*-------------------------------------------------
 * 函数名：  WByte
 * 功能：    UART 发送一个字节
 * 输入：    input
 * 输出：    无
 -------------------------------------------------*/
void WByte(uchar input)
{
                                   //发送起始位
    uchar i=8;
    TXIO = 1;
    TMR0 = Bord;
    T0IE = 1;
    WaitTF0();
    TXIO=0;
    WaitTF0();
                                   //发送 8 位数据位
    while(i--)
    {
        if(input&0x01)             //先传低位
        {
            TXIO=1;
        }
        else
```

```
        {
            TXIO = 0;
        }
        WaitTF0();
        input=input>>1;
    }
                                        //发送校验位(无)
                                        //发送结束位
    TXIO=(bit)1;
    T0IE=0;
}

/*-------------------------------------------------
 * 函数名：  RByte
 * 功能：    UART 接收一个字节
 * 输入：    无
 * 输出：    Output
 -------------------------------------------------*/
uchar RByte()
{
    uchar Output=0;
    uchar i=8;
    T0IE=1;                         //启动 Timer0
    TMR0 = Bord;
    WaitTF0();
    T0IE=1;                         //启动 Timer0
    TMR0 = Bord;
    WaitTF0();                      //等过起始位
                                    //发送 8 位数据位
    while(i--)
    {
        Output >>=1;
        if(RXIO)
        {
            Output|=0x80;           //先收低位
        }
        WaitTF0();                  //位间延时
    }
    T0IE=0;                         //停止 Timer0
    return Output;
}

/*-------------------------------------------------
 * 函数名: main
```

```
 * 功能：    主函数
 * 输入：    无
 * 输出：    无
 -----------------------------------------------------*/
void main()
{
    uchar rdata = 0;

    POWER_INITIAL();

    TIMER0_INITIAL();
    PA2_Level_Change_INITIAL();
    GIE = 1;                        //开中断
    T0IE = 1;                       //开定时器/计数器 0 中断
    while(1)
    {
        if(RXFLAG)                  //外部中断下降沿触发了
        {
            rdata = RByte();
            WByte(rdata);

            IOCA2 =1;               //使能 PA2 电平变化中断
            PAIE =1;                //使能 PA INT 中断
            RXFLAG = 0;
        }
    }
}
//=========================================================
```

Fremont Micro Devices (SZ) Limited

#5-8, 10/F, Changhong Building, Ke-Ji Nan 12 Road, Nanshan District, Shenzhen, Guangdong 518057
Tel: (86 755) 86117811
Fax: (86 755) 86117810

Fremont Micro Devices (Hong Kong) Limited

#16, 16/F, Blk B, Veristrong Industrial Centre, 34-36 Au Pui Wan Street, Fotan, Shatin, Hong Kong
Tel: (852) 27811186
Fax: (852) 27811144

Fremont Micro Devices (USA), Inc.

42982 Osgood Road Fremont, CA 94539
Tel: (1-510) 668-1321
Fax: (1-510) 226-9918

Web Site: http://www.fremontmicro.com/