

# **FT60F01X**

## **IR\_Send Application note**

## 目录

1. IR 介绍.....	3
2. IR SEND 相关寄存器的设置 .....	3
3. 应用范例.....	4
联系信息 .....	11

## FT60F01x IR\_Send 应用

### 1. IR 介绍

一个通用的红外遥控系统由发射和接收两大部分组成，如图 1-1 所示：

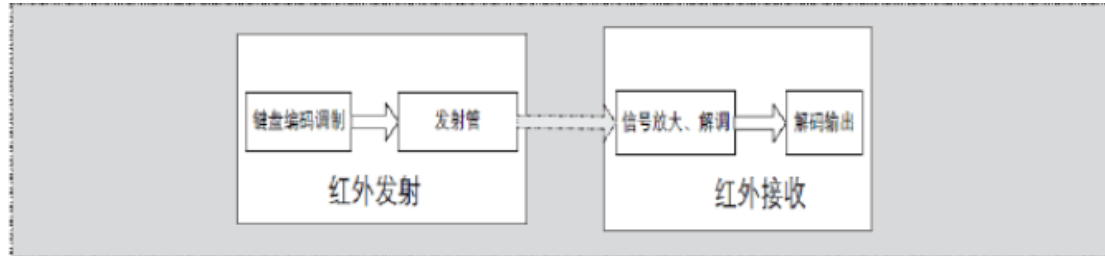


图 1-1

发射部分主要包括键盘矩阵、编码调制、红外发射管；接收部分包括光、电信号的转换以及放大、解调、解码电路。

举例来说，通常我们家电遥控器信号的发射，就是将相应按键所对应的控制指令和系统码（由 0 和 1 组成的序列），调制在 32~56kHz 范围内的载波上（目的为：抗干扰及低功率），然后经放大（接三极管）、驱动红外发射管（透明的头）将信号发射出去。

### 2. IR Send 相关寄存器的设置

本例使用两个定时器，一个是产生 38kHz 载波频率，另一个定时器是做时基，定时时长是 560μs，红外信号的高低电平是 560μs 的整数倍。

**定时器 0** 为 8 位，可配置为计数器或定时器使用，当作为外部事件（T0CKI）计数器时，可以配置为上升沿或者下降沿计数。作为定时器时，其计数时钟为系统时钟的 4 分频，即每一指令周期递增一次。有一个与 WDT 共用的 8 位预分频器，PSA 为 0 时该预分频器分配给定时器 0 使用。

例如在系统时钟和 4T 模式下，定时时长计算公式如下：

$$\text{定时时长} = (1/\text{系统时钟频率}) * 4 * \text{预分频值} * 255$$

**定时器 2** 为 8 位，其时钟源为指令周期，可以作为计数器和定时器使用，当 TMR2 值等于 PR2 时会产生中断，Timer2 具有预分频器和后分频器，预分频比为 1: 1、1: 4 和 1: 16，后分频比为 1: 1~1: 16。

在系统时钟和 4T 模式下，定时时长计算公式如下：

$$\text{定时时长} = (1/\text{系统时钟频率}) * 4 * \text{预分频值} * \text{后分频值} * \text{PR2}$$

本讲解以 IC FT60F011A SOP8 为示范，每 5 秒钟会发出一次信号，信号的码为 IRData[4] = {0x00,0xff,0x40,0xBf}

本程序 IR 接收与 LED 所对应的 IO 引脚：

```
#define SendIO    PA4
```

### 3. 应用范例

```
//*****
/* 文件名: TEST_60F01x_IR_Send.c
* 功能:    FT60F01x-红外发射 功能演示
* IC:      FT60F011A SOP8
* 晶振:    16M/4T
* 说明:    演示程序中,IR 红外是采用 6122 协议, 起始信号是 9ms 低电平, 到 4.5ms 高电平, 再到低
*          8 位用户识别码, 到高 8 位的用户识别码, 8 位数据码, 8 位数据码的反码。SendIO (PA4)
*          定时 (5 秒钟) 发送一次, 接收端收到遥控器发过来的数据后, 校验数据互为补码, LED 会
*          开关。
*
*          FT60F011A  SOP8
*          -----
*  VDD-----|1(VDD)   (GND)8|-----GND
*  NC-----|2(PA2)   (PA4)7|-----SendIO
*  NC-----|3(PA1)   (PA5)6|-----NC
*  NC-----|4(PA3)   (PA0)5|-----NC
*          -----
*/
//*****
#include "SYSCFG.h";
#include "FT60F01X.h";
//***** 宏定义 *****
#define uchar      unsigned char
#define uint       unsigned int

#define IRSendIO    PA4                //串口的发送脚

#define IRSend_HIGH_1  1                //560μs
#define IRSend_LOW_1   3                //1680μs

#define IRSend_HIGH_0  1                //560μs
#define IRSend_LOW_0   1                //560μs

#define IRSend_PIN_1   T0IE = 1        //发送数据 开启定时器 0
#define IRSend_PIN_0   T0IE = 0        //关闭定时器 0

#define Status_NOSEND  0                //不发送的状态
#define Status_Head    1                //发送引导码的状态
#define Status_Data     2                //发送数据的状态

uchar  IRSendStatus;                    //发送状态, 是发送引导码还是数据
uchar  IRSendData;                      //发送的数据中转变量
uchar  TxBit=0,TxTime=0;
```

```

uchar  Sendbit = 0;
uchar  level0,level1;                                //一位数据里发送与关闭的时间值
bit     SendLastBit = 0;
uchar  SaveLastBit = 0;
uint    SYSTime5S = 0;                                //系统时间，5S 发送一次

uchar  IRData[4] = {0x00,0xff,0x40,0xBf};            //需要发送的 4 个数据
/*-----
 * 函数名: POWER_INITIAL
 * 功能:   系统初始化
 * 输入:   无
 * 输出:   无
-----*/
void POWER_INITIAL(void)
{
    OSCCON = 0B01110000;                                //IRCF=111=16MHz/4T=4MHz,0.25μs
    INTCON = 0;                                          //暂禁止所有中断
    OPTION = 0B00001000;                                //Bit3=1,WDT MODE,PS=000=WDT RATE 1:1

    PORTA = 0B00000000;
    TRISA = 0B00000000;                                //PA 输入输出 0-输出 1-输入
                                                    //PA4-OUT
    WPUA = 0B00000000;                                //PA 端口上拉控制 1-开上拉 0-关上拉

    MSCKCON = 0B00000000;
    //Bit4=0,禁止 LVR(60F01x O 版之前)
    //Bit4=0,LVREN 使能时,开启 LVR(60F01x O 版及 O 版之后)
    //Bit4=1,LVREN 使能时,工作时开启 LVR,睡眠时自动关闭 LVR(60F01x O 版及 O 版后)
}
/*-----
 * 函数名: TIMER0_INITIAL
 * 功能:   初始化设置定时器 0
 * 说明:   38KHz 发生器, 1000000/38000=26.3μs .由于定时太短, 频繁进定时器, 时间有一定的
 *         误差, 239 并不是直接算出来的, 是用示波器看的。
 * 设置 TMR0 定时时长=1/系统时钟频率*4*预分频值*26
                =(1/16000000)*4*2*26=13μs
-----*/
void TIMER0_INITIAL (void)
{
    OPTION = 0B00000000;    //时钟源为指令时钟, 预分频比为 1:2
    //Bit5: T0CS Timer0 时钟源选择
    //      1-外部引脚电平变化 T0CKI 0-内部时钟(FOSC/4)
    //Bit4: T0SE 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

```

```

//Bit[2:0]: PS 8 位预分频比 000 - 1:2
TMR0 = 239;
TOIF = 0;                                //清空 T0 软件中断
}
/*-----
* 函数名: TIMER2_INITIAL
* 功能:   初始化设置定时器 1
* 设置 TMR2 定时时长=1/系统时钟频率*4*预分频值*后分频值*PR2
              =(1/16000000)*4*4*4*140=560μs
-----*/
void TIMER2_INITIAL(void)
{
    T2CON = 0B00011001;
    //Bit[6:3]: 定时器 2 后分频比 0011-1:4
    //Bit2:   定时器 2 使能位 0-关闭 1-使能
    //Bit[1:0]: 定时器 2 预分频比 01-1:4
    TMR2 = 0;                                //TMR2 赋初值
    PR2 = 140;

    TMR2IF = 0;                                //清 TMER2 中断标志
    TMR2IE = 1;                                //使能 TMER2 的中断
    TMR2ON = 1;                                //使能 TMER2 启动
    PEIE=1;                                    //使能外设中断
}
/*-----
* 函数名: SendCtrl
* 功能:   发送数据函数
* 输入:   无
* 输出:   无
-----*/
void SendCtrl(void)
{
    if (IRSendStatus == Status_NOSEND)        //不发送的状态
    {
        IRSend_PIN_0;
        Sendbit = 0;
        TxTime = 0;
    }
    else if (IRSendStatus == Status_Head)     //发送引导码
    {
        TxTime++;
        if (TxTime < 17)                      //发送 9ms 信号
        {

```

```

        IRSend_PIN_1;
    }
    else if (TxTime < 24)                //4.5ms 不发送
    {
        IRSend_PIN_0;
    }
    else
    {
        TxTime = 0;
        IRSendStatus = Status_Data;
    }
    IRSendData = IRData[0];
    TxBit = 0x01;
}
else if (IRSendStatus == Status_Data)    //发送数据
{
    if (IRSendData & TxBit)                //1, 是 1:3 的时间
    {
        level1 = IRSend_HIGH_1;
        level0 = IRSend_LOW_1;
    }
    else                                    //0, 是 1:1 的时间
    {
        level1 = IRSend_HIGH_0;
        level0 = IRSend_LOW_0;
    }
    TxTime++;
    if (TxTime <= level1)                    //发送信号
    {
        IRSend_PIN_1;
    }
    else if (TxTime <= (level0+level1))    //不发送信号
    {
        IRSend_PIN_0;
    }
    else if (Sendbit < 4)                    //发送 4 位数据未完成
    {
        TxTime = 1;
        IRSend_PIN_1;
        SaveLastBit = IRSendData & TxBit;
        TxBit <<= 1;
        if (TxBit == 0x00)                //发送完一个字节
        {
            TxBit = 0x01;

```

```

        Sendbit++;
        IRSendData = IRData[Sendbit];
        if (Sendbit > 3)                //最后一位要注意，因为发送完了还要一个脉冲
        {
            SendLastBit = 1;
        }
    }
}
else                                    //数据完成了，要补脉冲
{
    if(SendLastBit)
    {
        TxTime++;
        if(SaveLastBit)
        {
            if(TxTime < 3)
            {
                IRSend_PIN_0;
            }
            else if(TxTime < 4)
            {
                IRSend_PIN_1;
            }
            else
            {
                IRSend_PIN_0;
                IRSendStatus = Status_NOSEND;
                IRSend_PIN_0;
                SendLastBit = 0;
                TxBit = 0;
                TxTime = 0;
            }
        }
    }
    else
    {
        if(TxTime < 5)
        {
            IRSend_PIN_0;
        }
        else if(TxTime < 6)
        {
            IRSend_PIN_1;
        }
        else
    }
}

```



```

        {
            IRSend_PIN_0;
            IRSendStatus = Status_NOSEND;
            IRSend_PIN_0;
            SendLastBit = 0;
            TxBit = 0;
            TxTime = 0;
        }
    }
}

/*-----*/
* 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
/*-----*/

void interrupt ISR(void)
{
    //定时器 0 的中断处理
    if(T0IE && T0IF)                //13μs
    {
        TMR0 = 239;                //注意:对 TMR0 重新赋值 TMR0 在两个周期内不变化
        T0IF = 0;
        IRSendIO = ~IRSendIO;      //翻转电平产生 38KHz 信号
    }

    //定时器 2 的中断处理
    if(TMR2IE && TMR2IF)            //560μs 中断一次 红外每一位都是 560μs 的倍数
    {
        TMR2IF = 0;
        SendCtrl();
        SYSTime5S++;
    }
}

/*-----*/
* 函数名: main
* 功能:   主函数
* 输入:   无
* 输出:   无
/*-----*/

void main(void)

```

```
{
    POWER_INITIAL();
    TIMER0_INITIAL();
    TIMER2_INITIAL();
    GIE = 1;                                //开中断
    while(1)
    {
        if(SYSTime5S >10000)                //定时 5s
        {
            SYSTime5S = 0;
            IRTSendStatus = Status_Head;
        }
    }
}
```

## 联系信息

### **Fremont Micro Devices Corporation**

#5-8, 10/F, Changhong Building  
Ke-Ji Nan 12 Road, Nanshan District,  
Shenzhen, Guangdong, PRC 518057

Tel: (+86 755) 8611 7811

Fax: (+86 755) 8611 7810

### **Fremont Micro Devices (HK) Limited**

#16, 16/F, Block B, Veristrong Industrial Centre,  
34-36 Au Pui Wan Street, Fotan, Shatin, Hong Kong SAR

Tel: (+852) 2781 1186

Fax: (+852) 2781 1144

<http://www.fremontmicro.com>

\* Information furnished is believed to be accurate and reliable. However, Fremont Micro Devices Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties, which may result from its use. No license is granted by implication or otherwise under any patent rights of Fremont Micro Devices Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. Fremont Micro Devices Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of Fremont Micro Devices Corporation. The FMD logo is a registered trademark of Fremont Micro Devices Corporation. All other names are the property of their respective owners.