FT62F21X

Application note

目录

1 EEPROM 相关寄存器的设置 ····· 3

2 应用范例 ····· 7

FT62F21x EEPROM 应用

1 EEPROM 相关寄存器的设置

片内集成有256个字节的EEPROM，通过EEADR进行寻址访问。软件可通过EECON1和EECON2对EEPROM进行编程操作，硬件实现了擦除和编程的自定时功能，无需软件查询，节省有限的代码空间，同时利用此特性，启动编程周期之后可以进入睡眠模式，以降低功耗。

数据EEPROM在使用（无论是读还是写）之前必须进行以下初始化操作：在未使用到的EEPROM某个单元写两次0xAA，后续程序不要对此单元操作。如：

```
SYSTEM_INIT:
……
……
LDWI    0x55
STR     EEPROM_ADDR
LDWI    0xAA
STR     EEPROM_DATA
LCALL   EEPROM_write
LCALL   EEPROM_write
……
```

◆编程数据EEPROM步骤

a. 把INTCON的GIE位清0；
b. 判断GIE是否为1，是则重复a步骤，否则可以进行下一步；
c. 往EEADR写入目标地址；
d. 往EEDAT写入目标数据；
e. 把位WREN3/WREN2/WREN1全部置1；
f. 把位WR置1（EECON2.0，此后WR会维持高）；
g. 写过程不能改变WREN3/2/1的值，否则编程终止；
h. 等大概2ms之后编程自动完成，WR自动清0，WREN3、WREN2、WREN1自动清0；
i. 如果想再次编程，重复步骤c~h即可；

例子 1：

```
BCR INTCON, GIE
BTSC INTCON, GIE
LJUMP $-2
BANKSEL EEADR
LDWI 55H
STR EEADR               ;地址为 0x55
STR EEDAT               ;数据为 0x55
LDWI 34H
STR EECON1                  ;WREN3/2/1 同时置 1
BSR EECON2, 0           ;启动写
BSR INTCON, GIE         ;把 GIE 置 1
```

例子 2：

```
BCR INTCON, GIE
BTSC INTCON, GIE
LJUMP $-2
BANKSEL EEADR
LDWI 55H
STR EEADR              ;地址为 0x55
STR EEDAT              ;数据为 0x55
LDWI 34H
STR EECON1             ;WREN3/2/1 同时置 1
NOP                    ;这里 NOP 可以换成其他指令
BSR EECON2, 0          ;启动写，实际上硬件不会启动编程 EEPROM 操作

BCR EECON1, WREN1      ;先清 WREN1，使得 WREN3/2/1 不同时为 1
BSR EECON1, WREN1      ;重新置位 WREN1，令 WREN3/2/1 同时为 1
BSR EECON2, 0          ;启动写，这次硬件将对 EEPROM 编程
BSR INTCON, GIE
```

注意：

1. **以上步骤的 E、F 两步必须是连续的两条指令周期完成，不能错开（如例子 2），否则编程操作不会启动，其中 WREN3、WREN2 和 WREN1 可以不是同一条指令置 1，比如可以用 BSR 指令分开对各位置 1；**
2. **如果 E、F 两步被错开执行，要想启动下一次编程操作，必须在 E、F 之前加入一步，把 WREN3、WREN2 或者 WREN1 任意一位清 0，如例子 2；**
3. **编程过程中读操作无效；**

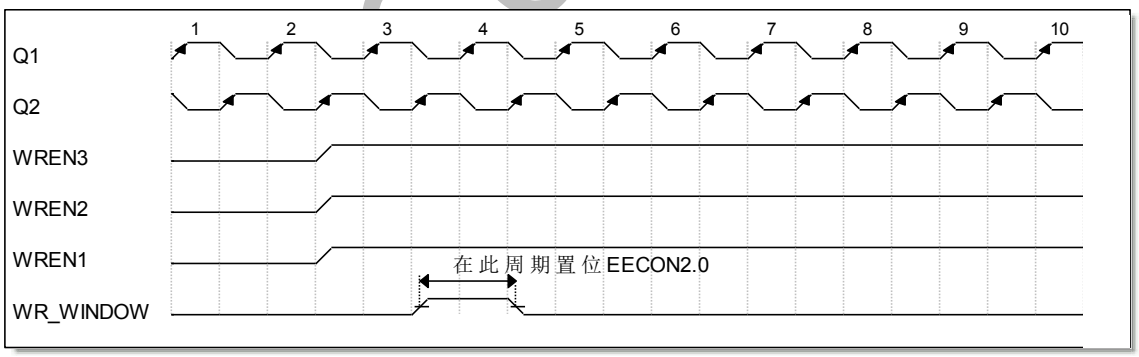

图 13.1 软件编程数据 EEPROM 时序

◆读数据EEPROM

要读取数据存储单元，用户必须将地址写入EEADR寄存器，然后将EECON1寄存器的控制位RD置1。在紧接着的下一周期，EEDAT寄存器就被EEPROM数据写入。因此该数据可由下一条指令读取。EEDAT将保持这个值直到用户下一次从该单元读取或向该单元写入数据时（在写操作过程中）。

下面是读取EEPROM的一段示例程序：

```
BANKSEL EEADR
LDWI    dest_addr
STR     EEADR
```

```
BSR     EECON1, RD
LDR     EEDAT, W
```

◆关于编程周期

启动数据EEPROM的编程操作后，2ms的编程计时开始，在这段时间内，CPU并不会暂停，而是继续执行程序

◆相关寄存器的各个位定义如下：

1） EEDAT 寄存器

Bit	7	6	5	4	3	2	1	0
Name	EEDAT[7:0]							
Reset	0	0	0	0	0	0	0	0

Bit7~Bit0：EEPROM 数据寄存器

2） EEADR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	EEADR[7:0]							
Reset	0	0	0	0	0	0	0	0

Bit7~ Bit6：EEPROM 地址寄存器

3） EECON1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	-	-	WREN3	WREN2	WRERR	WREN1	-	RD
Reset	-	-	0	0	x	0	-	0

Bit7~Bit6：保留位,读 0

Bit5：数据 EEPROM 写使能 3，和 WREN2、WREN1 结合使用

Bit4：数据 EEPROM 写使能 2，和 WREN3、WREN1 结合使用

Bit3：数据 EEPROM 写错误标志位

1：在 EEPROM 编程周期发生了看门狗或外部复位，中止

0：在 EEPROM 编程周期正常完成

Bit2：数据 EEPROM 写使能 1

WREN3-1=111：允许软件对 EEPROM 编程，编程完成后各位自动清 0

WREN3-1=其他值：禁止软件对 EEPROM 编程

Bit1：保留位，读 0

Bit0：数据 EEPROM 读控制位

此位是只写，读永远返回 0

写 1：启动一次数据 EEPROM 读周期

写 0：不启动读

4） EECON2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	-	-	-	WR
Reset	-	-	-	-	-	-	-	0

Bit7~Bit1：保留位，读 0

Bit0：数据 EEPROM 写控制位

读操作：

1：数据EEPROM编程周期进行中

0：数据EEPROM不处于编程周期

写操作：

1：启动一次数据EEPROM编程周期

0：无意义

5） PIE1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	EEIE	CKMIE	LVDIE	-	-	OSFIE	TMR2IE	-
Reset	0	0	0	-	-	0	0	-

Bit7：EE 写中断使能位

1：使能 EE 写操作完成中断

0：关闭 EE 写操作完成中断

Bit6：快时钟测量慢时钟操作完成中断使能位

1：使能快时钟测量慢时钟操作完成中断

0：关闭快时钟测量慢时钟操作完成中断

Bit5：按键中断使能位

1：使能按键中断

0：禁止按键中断

Bit4~Bit3：保留位，读 0

Bit2：振荡器故障中断允许位

1：允许振荡器故障中断

0：禁止振荡器故障中断

Bit1：Timer2 与 PR2 比较相等中断使能位

1：使能 timer2 的值等于 PR2 中断

0：关闭 timer2 的值等于 PR2 中断

Bit0：保留位，读 0

6） PIR1

Bit	7	6	5	4	3	2	1	0
Name	EEIF	CKMIF	LVDIF	-	-	-	TMR2IF	-
Reset	0	0	0	-	-	-	0	-

Bit7：EEIF 写中断标志位

1：EE 写操作完成

0：EE 写操作未完成，或已经由软件清 0

Bit6：快时钟测量慢时钟操作完成中断标志位

1：快时钟测量慢时钟操作完成

0：快时钟测量慢时钟未完成，或已经由软件清 0

Bit5：LVD 中断标志位

1：LVD 检测电压低于所设置阈值

0：LVD 检测电压高于所设置阈值，或已经由软件清 0

Bit4~Bit2：保留位，读 0

Bit1：Timer2 与 PR2 比较相等中断标志位
1：timer2 的值等于 PR2
0：timer2 的值不等于 PR2，或已经由软件清 0
Bit0：保留位，读 0

7） INTCON 寄存器

Bit	7	6	5	4	3	2	1	0
Name	GIE	PEIE	T0IE	INTE	PAIE	T0IF	INTF	PAIF
Reset	0	0	0	0	0	0	0	0

Bit7：全局中断使能位
1：使能全局中断
0：关闭全局中断
Bit6：外设中断使能位
1：使能外设中断
0：关闭外设中断
Bit5：Timer0 定时器溢出中断使能位
1：使能Timer0中断
0：关闭Timer0中断
Bit4：外部中断使能位
1：使能PA2/INT外部中断
0：关闭PA2/INT外部中断
Bit3：PORTA端口状态变化中断使能位
1：使能PORTA端口状态变化中断使能
0：关闭PORTA端口状态变化中断使能
Bit2：Timer0定时器溢出中断标志位
1：Timer0寄存器溢出（必须软件清零）
0：Timer0寄存器无溢出
Bit1：外部中断标志位
1：PA2/INT外部中断发生（必须软件清零）
0：PA2/INT外部中断无发生
Bit0： PORTA端口状态改变中断标志位
1：PORTA<5:0>至少有一个端口状态发生了改变（必须软件清零）
0：PORTA<5:0>没有一个端口发生状态改变

2 应用范例

```
//*************************************************************
/*   文件名：  TEST_FT62F21x_EEPROM.c
*    功能：    FT62F21x-EEPROM 功能演示
*    IC:       FT62F211 SOP8
*    晶振：    16M/4T
*    说明：    该程序读取 0x12 地址的值,取反后存入 0x13 地址
*
*
```

```
*                          FT62F211 SOP8
*                         ----------
*   NC----------|1(PA4)          (PA3)8|----------NC
*   NC----------|2(TKCAP)        (PA0)7|----------NC
*   VDD---------|3(VDD)          (PA1)6|----------NC
*   GND---------|4(VSS)           (PA2)5|----------NC
*                         ----------
*/
//************************************************************
#include  "SYSCFG.h";
#include  "FT62F21X.h";
//************************************************************
#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

#define WDT_256K       0X80
#define WDT_32K        0X00
//************************************************************
//**********************宏定义*******************************
#define   unchar        unsigned char
#define   unint         unsigned int
#define   unlong        unsigned long

unchar EEReadData;
 /*-------------------------------------------
 *   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 = WDT_32K|OSC_16M|0X00;
  //OSCCON = 0B01110000;            //WDT 32KHZ IRCF=111=16MHZ/4=4MHZ,0.25US/T

     INTCON = 0;                     //暂禁止所有中断

     PORTA = 0B00000000;
```

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

    OPTION = 0B00001000;         //Bit3=1 WDT MODE,PS=000=1:1 WDT RATE
                                 //Bit7(PAPU)=0  由 WPUA 决定是否上拉
    MSCON = 0B00000000;
}
/*---------------------------------------------
 *  函数名：EEPROMread
 *  功能：   读 EEPROM 数据
 *  输入：   需要读取数据的地址  EEAddr
 *  输出：   对应地址读出的数据  ReEEPROMread
 ---------------------------------------------*/
unchar EEPROMread(unchar EEAddr)
{
    unchar ReEEPROMread;
    EEADR = EEAddr;
    RD=1;
    ReEEPROMread =EEDAT;
    return ReEEPROMread;
}
/*---------------------------------------------
 *  函数名：EEPROMwrite
 *  功能：   写数据到 EEPROM
 *  输入：   需要读取数据的地址  EEAddr
 *           需要写入的数据 Data
 *  输出：   无
 ---------------------------------------------*/
void EEPROMwrite(unchar EEAddr,unchar Data)
{
    GIE = 0;                     //写数据必须关闭中断
    while(GIE);                  //等待 GIE 为 0
    EEADR = EEAddr;              //EEPROM 的地址
    EEDAT = Data;                //EEPROM 的写数据   EEDATA = Data;
    EEIF = 0;
    EECON1 |= 0x34;              //置位 WREN1,WREN2,WREN3 三个变量.
    WR = 1;                      //置位 WR 启动编程
    while(WR);                   //等待 EE 写入完成
    GIE = 1;
}
/*---------------------------------------------
 *  函数名:   main
 *  功能：   主函数
 *  输入：   无
```

```
 *   输出：    无
 ------------------------------------------------*/
void main()
{
    POWER_INITIAL();                    //系统初始化
    EEPROMwrite(0x55,0xaa);
    EEPROMwrite(0x55,0xaa);             //在未使用到的随意一个地址写两次 0xAA

    EEReadData = EEPROMread(0x12);      //读取 0x12 地址 EEPROM 值
    EEPROMwrite(0x13,~EEReadData);      //取反写入地址 0x13

    while(1)
    {

        NOP();

    }
}
```

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/