

FT60F01X

SPI Application note

目录

1. SPI 相关寄存器的设置	3
2. 应用范例	5
联系信息	10

FT60F01x SPI 应用

1. SPI 相关寄存器的设置

SPI是串行外设接口（Serial Peripheral Interface）的缩写。SPI，是一种高速的，全双工，同步的通信总线，以主从方式工作，这种模式通常有一个主设备和一个或多个从设备，需要至少4根线，事实上3根也可以（单向传输时）。也是所有基于SPI的设备共有的，它们是：

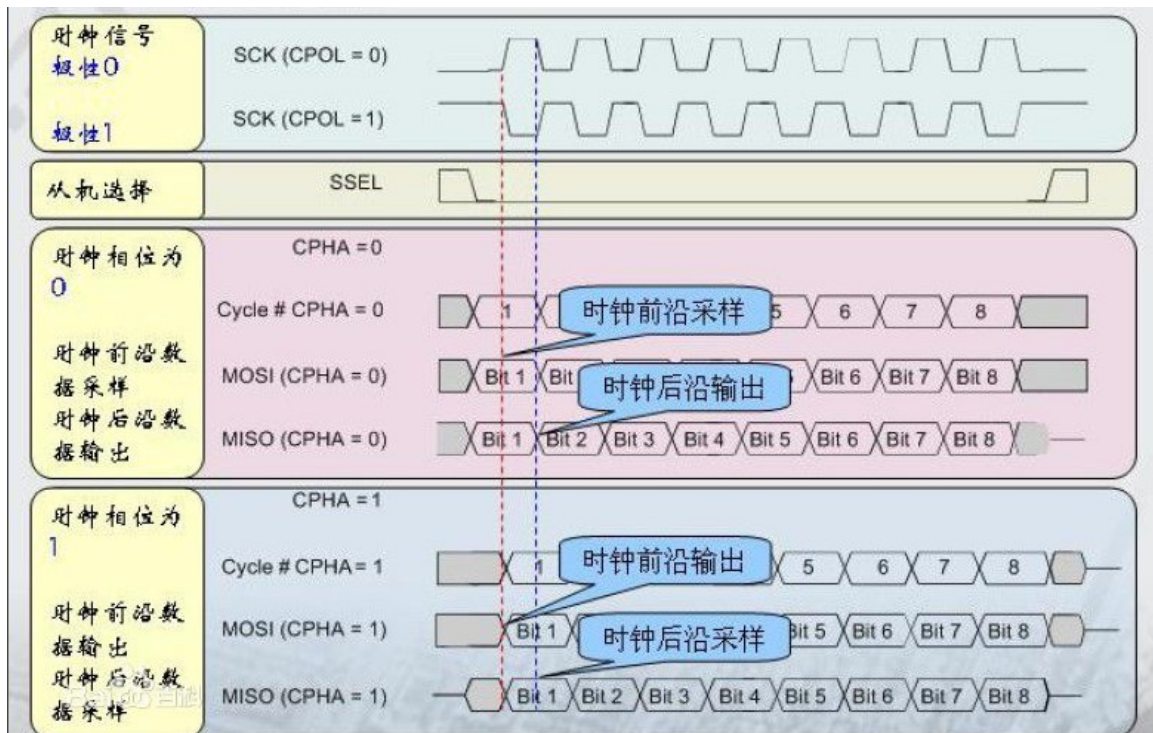
SDO/MOSI ----- 主设备数据输出，从设备数据输入；

SDI/MISO ----- 主设备数据输入，从设备数据输出；

SCLK ----- 时钟信号，由主设备产生；

CS -----片选，从设备使能信号，由主设备控制。

SPI通信有4种不同的模式，不同的从设备可能在出厂是就是配置为某种模式，这是不能改变的；但我们的通信双方必须是工作在同一模式下，所以我们可以对我们的主设备的SPI模式进行配置，通过CPOL（时钟极性）和CPHA（时钟相位）来控制我们主设备的通信模式



Mode0: CPOL=0, CPHA=0

Mode1: CPOL=0, CPHA=1

Mode2: CPOL=1, CPHA=0

Mode3: CPOL=1, CPHA=1

本程序采用mode0的工作模式，以FT60F011A SOP8为例，四根数据线所对应的IO引脚：

```
#define MISO PORTA,4  
#define MOSI PORTA,2  
#define SCK PORTA,5  
#define CS PORTA,0
```

2. 应用范例

```
//*****
/* 文件名: TEST_60F01x_SPI.c
* 功能:    FT60F01x_SPI 功能演示
* IC:      FT60F011A SOP8
* 晶振:    16M/4T
* 说明:    此演示程序为 60F01x_SPI 的演示程序.
*          该程序读取(25C64)0x12 地址的值,取反后存入 0x13 地址
*
*          FT60F011A  SOP8
*          -----
*  VDD-----|1(VDD)  (GND)8|-----GND
*  MOSI -----|2(PA2)  (PA4)7|-----MISO
*  NC-----|3(PA1)  (PA5)6|-----SCK
*  NC-----|4(PA3)  (PA0)5|-----CS
*
*          -----
*/
//*****
#include "SYSCFG.h"
//***** 宏定义 *****
#define unchar    unsigned char
#define uint      unsigned int

#define MISO      PA4
#define MOSI      PA2
#define SCK       PA5
#define CS        PA0

unchar  SPIReadData;
/*-----
*  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 = 0B01110000;    //IRCF=111=16MHz/4T=4MHz, 0.25μs
    INTCON = 0;             //暂禁止所有中断
    PORTA = 0B00000000;
    TRISA = 0B00010000;    //PA 输入输出 0-输出 1-输入
    WPUA = 0B00010000;    //PA 端口上拉控制 1-开上拉 0-关上拉
    OPTION = 0B00001000;    //Bit3=1, WDT MODE, PS=000=WDT RATE 1:1
    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: init_25c64_io
 * 功能:   25C64 初始化
 * 输入:   无
 * 输出:   无
-----*/
void init_25c64_io(void)
{
    CS = 1;
    SCK = 0;
    MOSI = 0;
}
/*-----
 * 函数名: SPI_RW
 * 功能:   主机输出以及输入一个字节
 * 输入:   data
 * 输出:   根据接收的 data 输出给从机一个字节
-----*/
unchar SPI_RW(unchar data)
{
    unchar i;
    for(i=0;i<8;i++)
    {
        if(data&0x80)
            MOSI = 1;
        else
            MOSI = 0;
        NOP();
        data<<=1;
        SCK = 1;
        NOP();
        if(MISO)
            data |= 0x01;
        else
            data &= 0xFE;
        NOP();
        SCK = 0;
    }
    return data;
}
```

```
/*-----
 * 函数名: WriteEnable
 * 功能:   写允许 (将 WEN 置位)
 *-----*/
void WriteEnable(void)
{
    CS=0;
    SPI_RW(0x06);
    CS=1;
}
/*-----
 * 函数名: WriteDisable
 * 功能:   写禁止 (将 WEN 复位)
 *-----*/
void WriteDisable (void)
{
    CS=0;
    SPI_RW(0x04);
    CS=1;
}
/*-----
 * 功能:   读取 25C64 芯片的状态。
 * 返回值: 状态寄存器数据字节
 * 注:     25C64 内部状态寄存器第 0 位=0 表示空闲, 0 位=1 表示忙。
 *-----*/
uchar SPI_ReadStatus(void)
{
    uchar status=0;
    CS=0;
    SPI_RW(0x05);                //0x05 读取状态的命令字
    status = SPI_RW(0x00);
    CS=1;                        //关闭片选
    return status;
}
/*-----
 * 程序名: SPI_WriteStatus
 * 功能:   写 25C64 芯片的状态寄存器。
 *         只有 BP1、BP0 (bit7、3、2)可以写、
 * 注:     25c64 内部状态寄存器第 0 位=0 表示空闲, 0 位=1 表示忙。
 *-----*/
void SPI_WriteStatus(uchar Status)
{
    CS=0;
    SPI_RW(0x01);                //0x01 读取状态的命令字
```

```
SPI_RW(Status);                //写入一个字节
CS=1;                          //关闭片选
}
/*-----
* 程序名: SPI_Read
* 输入:   16 位的地址
* 返回:   读取的数据
* 说明:   从 25c64 指定的地址读取一个字节
-----*/
uchar SPI_Read(uint addr)
{
    uchar spidata;
    while(SPI_ReadStatus() & 0x01);    //判断是否忙
    CS=0;                               //使能器件
    SPI_RW(0x03);                       //发送读取命令
    SPI_RW((unsigned char)((addr)>>8));
    SPI_RW((unsigned char)addr);
    spidata = SPI_RW(0x00);             //读出数据
    CS=1;
    return spidata;
}
/*-----
* 程序名: SPI_Write
* 输入:   地址, 字节数据
* 说明:   将一个字节写入指定的地址
-----*/
void SPI_Write(uint addr, uchar dat)
{
    while(SPI_ReadStatus() & 0x01);    //判断是否忙
    WriteEnable();                     //SET WEL
    CS=0;                               //使能器件
    SPI_RW(0x02);                       //发送写命令
    SPI_RW((uchar)((addr)>>8));
    SPI_RW((uchar)addr);

    SPI_RW(dat);
    CS=1;                               //关闭片选
    WriteDisable();
    while(SPI_ReadStatus() & 0x01);
}
/*-----
* 函数名: main
* 功能:   主函数
* 输入:   无
```


* 输出: 无

-----*/

void main()

```
{
    POWER_INITIAL();           //系统初始化
    init_25c64_io();
    SPIReadData = SPI_Read(0x0012); //读取 0x12 地址 EEPROM 值
    SPI_Write(0x0013,~SPIReadData); //取反写入地址 0x13
    while(1)
    {
        NOP();
    }
}
```

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.