**NSS，从设备片选，可以由软件控制，内部的NSS电平由SPI\_CR2->SSI控制了。**

**Master Output Slave Input**

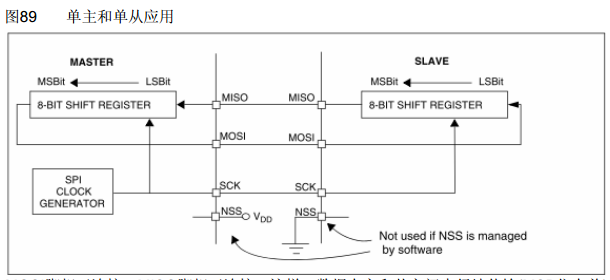
**Master Input Slave Output**

**SR**

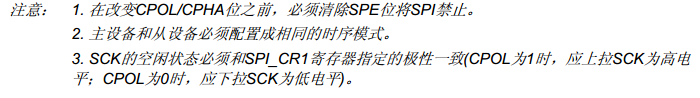
**ICR**

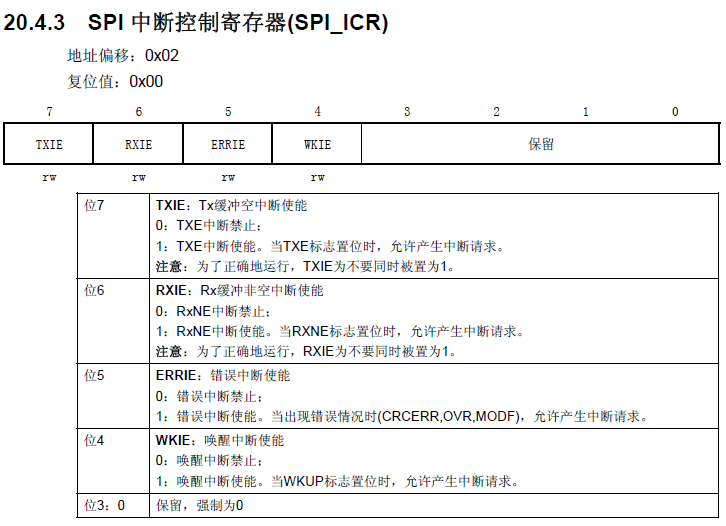
**CR2**

**CR1**

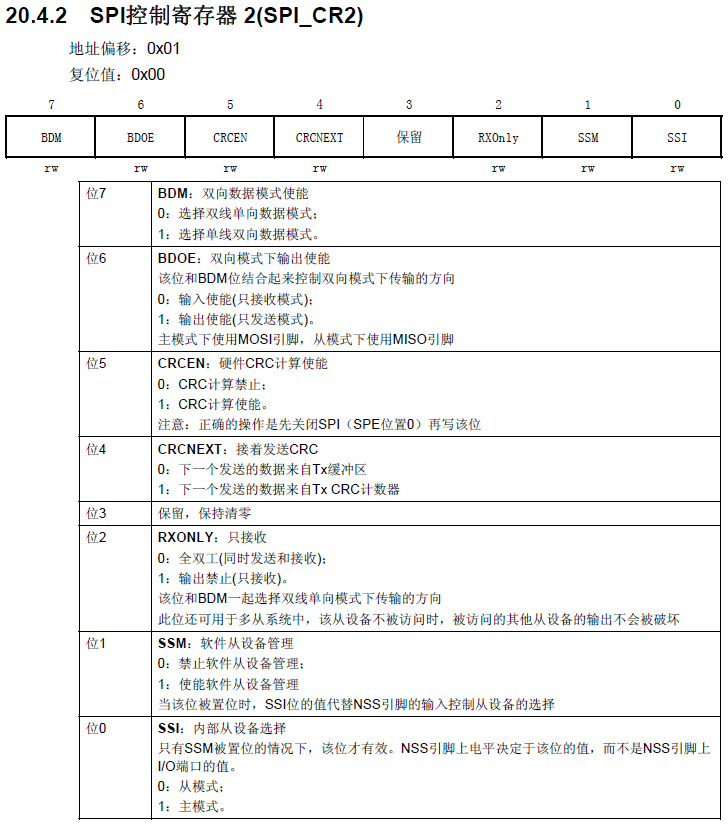




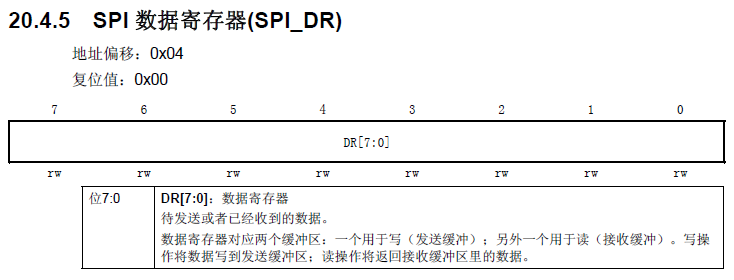


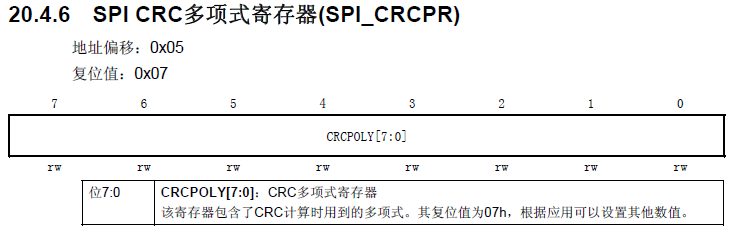


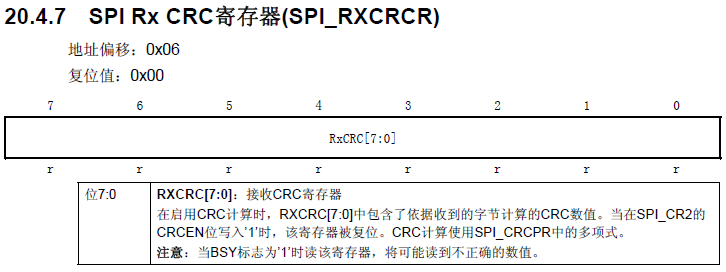


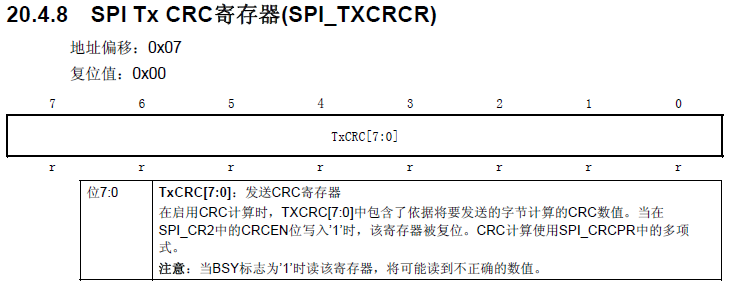


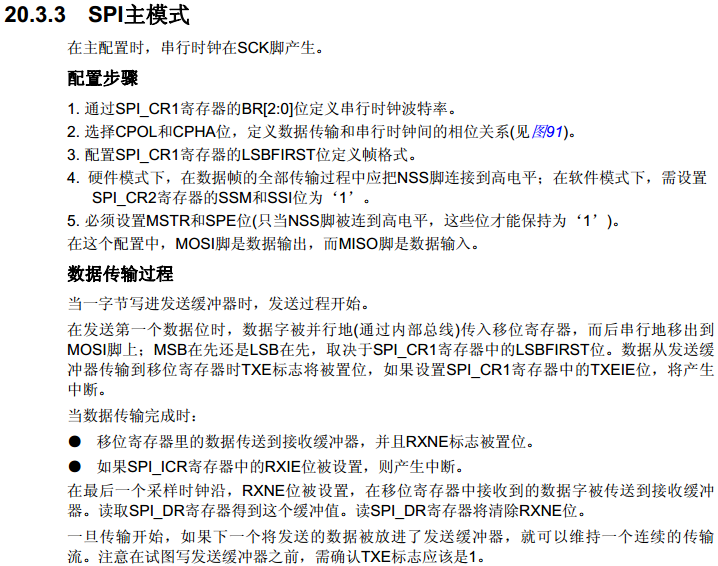












全部SPI配置demo、

void SPI\_conf**()**

**{**

SPI\_CR1 **=** **(**0**<<**7**)|(**0**<<**6**)|(**0**<<**3**)|(**1**<<**2**)|(**1**<<**1**)|(**1**<<**0**);**

/\* 先发送MSB 先禁止SPI 波特率设为 fbus/2 设置为主模式

空闲状态时SCK为高电平 数据从第二个时钟边沿开始采样 \*/

SPI\_CR2 **=** **(**0**<<**7**)|(**0**<<**5**)|(**0**<<**4**)|(**0**<<**2**)|(**1**<<**1**)|(**1**<<**0**);**

/\* 设为全双工模式 使能软件从设备管理 内部从设备选择为主模式 \*/

SPI\_ICR **=** **(**0**<<**7**)|(**0**<<**6**)|(**0**<<**5**)|(**0**<<**4**);**

/\* 禁止所有中断 \*/

SPI\_CR1 **|=** **(**1**<<**6**);**

/\* 开启SPI模块 \*/

**}**

unsigned char SPI\_SendByte**(**unsigned char byte**)**

**{**

**while(!(**SPI\_SR **&** 0x02**));**

/\* 等待发送寄存器为空 \*/

SPI\_DR **=** byte**;**

/\* 将发送的数据写到数据寄存器 \*/

**while(!(**SPI\_SR **&** 0x01**));**

/\* 等待接受寄存器满 \*/

**return** SPI\_DR**;**

/\* 读数据寄存器 \*/

**}**

STM8的SPI例程为SPI控制flash，所以很多是针对flash的编程：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT 风驰iCreate嵌入式开发工作室 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 文件名 ：w25x16.c

\* 描述 ：w25x16读写函数库

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "w25x16.h"

#include "spi.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名：SPI\_FLASH\_Init

\* 描述 ：使能W25X16 FLASH芯片的片选引脚--PE6

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_Init**()**

**{**

PE\_DDR **|=** **(**1**<<**6**);** /\* 设置为输出引脚 \*/

PE\_CR1 **|=** **(**1**<<**6**);** /\* 设置推挽输出--查看STM8寄存器.pdf P88 \*/

PE\_CR2 **|=** **(**1**<<**6**);** /\* 设置输出频率 1为10M，0为2M--查看STM8寄存器.pdf P89 \*/

FLASH\_CS\_HIGH**();** /\* 初始化Flash芯片片选引脚所接的I/O引脚，并输出高电平 \*/

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名：SPI\_FLASH\_WriteEnable

\* 描述 ：对FLASH的写使能 按照芯片的时序来写 W25X16.pdf P18

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_WriteEnable**(**void**)**

**{**

/\* 按照时序图，先拉低片选线 \*/

FLASH\_CS\_LOW**();**

/\* 传送写使能命令--06H \*/

SPI\_SendByte**(**WREN**);** /\* 这个函数是在spi.c中实现的 \*/

/\* 再拉高片选线 这样Flash就写使能了 \*/

FLASH\_CS\_HIGH**();**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名：SPI\_FLASH\_WaitForWriteEnd

\* 描述 ：读Flash芯片状态寄存器的值并且循环读取标志位直到写周期结束

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_WaitForWriteEnd**(**void**)**

**{**

u8 FLASH\_Status **=** 0**;**

/\* 拉低/CS 电平 选中芯片 \*/

FLASH\_CS\_LOW**();**

/\* 发送读状态寄存器命令，发送完之后，状态寄存器的值会被传送到STM8 要注意SPI总线下发送与接收是不同的引脚

状态寄存器的具体的数据格式可以参考W25X16.pdf P12 \*/

SPI\_SendByte**(**RDSR**);**

/\* 循环查询标志位 等待写周期结束 \*/

**do**

**{**

/\* 0xA5是一个无意义的指令 发这条指令是为了生成FLASH芯片需要的时钟

并且将FLASH的状态寄存器的值读回STM8 \*/

FLASH\_Status **=** SPI\_SendByte**(**Dummy\_Byte**);**

**}**

**while** **((**FLASH\_Status **&** WIP\_Flag**)** **==** SET**);**

/\* 拉高电平，不选中芯片 \*/

FLASH\_CS\_HIGH**();**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名：SPI\_FLASH\_PageWrite

\* 描述 ：在页写命令下写不止一个字节到W25X16，所写字节数不能超过每一页的限制

\* 输入 ：- pBuffer：指向要写入W25X16中的数据的指针

\* - WriteAddr：将写入的W25X16的地址

\* - NumByteToWrite：要写入W25X16的字节数，不能超过一页所能写入的最大

\* 的字节数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_PageWrite**(**u8**\*** pBuffer**,** u32 WriteAddr**,** u16 NumByteToWrite**)**

**{**

/\* 先使能对FLASH芯片W25X16的操作 \*/

SPI\_FLASH\_WriteEnable**();**

/\* 按照时序图，先拉低片选线 \*/

FLASH\_CS\_LOW**();**

/\* 发送页写命令 \*/

SPI\_SendByte**(**WRITE**);**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* 发送完页写命令后，紧跟着24位的的W25X16地址 先发高最高8位 \*/

SPI\_SendByte**((**WriteAddr **&** 0xFF0000**)** **>>** 16**);**

/\* 再发中间8位 \*/

SPI\_SendByte**((**WriteAddr **&** 0xFF00**)** **>>** 8**);**

/\* 最后发最低的8位地址 \*/

SPI\_SendByte**(**WriteAddr **&** 0xFF**);**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* 发送地址之后紧跟着要写入的数据 \*/

**while** **(**NumByteToWrite**--)**

**{**

/\* 发送要写入W25X16的数据 \*/

SPI\_SendByte**(\***pBuffer**);**

/\* 指向下一个要写入的数据 \*/

pBuffer**++;**

**}**

/\* 释放W25X16芯片 \*/

FLASH\_CS\_HIGH**();**

/\* 等待写操作结束 \*/

SPI\_FLASH\_WaitForWriteEnd**();**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名：SPI\_FLASH\_BufferWrite

\* 描述 ：使用页写命令写多个页的数据到W25X16

\* 输入 ：- pBuffer：指向要写入W25X16中的数据的指针

\* - WriteAddr：将写入的W25X16的地址

\* - NumByteToWrite：要写入W25X16的字节数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_BufferWrite**(**u8**\*** pBuffer**,** u32 WriteAddr**,** u16 NumByteToWrite**)**

**{**

u8 NumOfPage **=** 0**,** NumOfSingle **=** 0**,** Addr **=** 0**,** count **=** 0**,** temp **=** 0**;**

Addr **=** WriteAddr **%** SPI\_FLASH\_PageSize**;** /\* 判断要写入的地址是否页对齐 \*/

count **=** SPI\_FLASH\_PageSize **-** Addr**;**

NumOfPage **=** NumByteToWrite **/** SPI\_FLASH\_PageSize**;** /\* 总共要写几页 \*/

NumOfSingle **=** NumByteToWrite **%** SPI\_FLASH\_PageSize**;** /\* 不足一页的数据字节数 \*/

**if** **(**Addr **==** 0**)** /\* 写入的地址是否页对齐 \*/

**{**

**if** **(**NumOfPage **==** 0**)** /\* NumByteToWrite < SPI\_FLASH\_PageSize 只需写一页 \*/

**{**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** NumByteToWrite**);**

**}**

**else** /\* NumByteToWrite > SPI\_FLASH\_PageSize 不只写一页 \*/

**{**

**while** **(**NumOfPage**--)**

**{**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** SPI\_FLASH\_PageSize**);**

WriteAddr **+=** SPI\_FLASH\_PageSize**;** /\* 指向w25X16下一页的地址空间 \*/

pBuffer **+=** SPI\_FLASH\_PageSize**;**

**}**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** NumOfSingle**);** /\* 把剩下的不足一页的数据写完 \*/

**}**

**}**

**else** /\* 要写入的地址不是页对齐的地址 \*/

**{**

**if** **(**NumOfPage **==** 0**)** /\* NumByteToWrite < SPI\_FLASH\_PageSize 只写一页 \*/

**{**

**if** **(**NumOfSingle **>** count**)** /\* 判断所要写入的地址所在的页是否还有足够的空间写下要存放的数据 \*/

**{** /\* (NumByteToWrite + WriteAddr) > SPI\_FLASH\_PageSize \*/

temp **=** NumOfSingle **-** count**;**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** count**);**

WriteAddr **+=** count**;**

pBuffer **+=** count**;** /\* 将所要写入的地址的页剩下的空间写完 \*/

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** temp**);** /\* 再往新的一页写入剩下的数据 \*/

**}**

**else**

**{**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** NumByteToWrite**);**

**}**

**}**

**else** /\* NumByteToWrite > SPI\_FLASH\_PageSize 不只写入一页的情况 \*/

**{**

NumByteToWrite **-=** count**;**

NumOfPage **=** NumByteToWrite **/** SPI\_FLASH\_PageSize**;**

NumOfSingle **=** NumByteToWrite **%** SPI\_FLASH\_PageSize**;**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** count**);**

WriteAddr **+=** count**;**

pBuffer **+=** count**;**

**while** **(**NumOfPage**--)**

**{**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** SPI\_FLASH\_PageSize**);**

WriteAddr **+=** SPI\_FLASH\_PageSize**;**

pBuffer **+=** SPI\_FLASH\_PageSize**;**

**}**

**if** **(**NumOfSingle **!=** 0**)**

**{**

SPI\_FLASH\_PageWrite**(**pBuffer**,** WriteAddr**,** NumOfSingle**);**

**}**

**}**

**}**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名 : SPI\_FLASH\_BufferRead

\* 描述 : 从FLASH读取NumByteToRead字节的数据.

\* 输入 : - pBuffer : 存放从FLASH读取的数据的缓冲区的指针

\* - ReadAddr : 从FLASH的该地址处读数据

\* - NumByteToRead : 要读取的字节数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_BufferRead**(**u8**\*** pBuffer**,** u32 ReadAddr**,** u16 NumByteToRead**)**

**{**

/\* 拉低/CS 电平 选中芯片 \*/

FLASH\_CS\_LOW**();**

/\* 发送读数据命令 \*/

SPI\_SendByte**(**READ**);**

/\* 发送地址 24位 但对于本芯片来说21位有效 \*/

SPI\_SendByte**((**ReadAddr **&** 0xFF0000**)** **>>** 16**);**

SPI\_SendByte**((**ReadAddr**&** 0xFF00**)** **>>** 8**);**

SPI\_SendByte**(**ReadAddr **&** 0xFF**);**

**while** **(**NumByteToRead**--)** /\* 计数 \*/

**{**

/\* 读一个字节的数据 \*/

**\***pBuffer **=** SPI\_SendByte**(**Dummy\_Byte**);**

pBuffer**++;**

**}**

/\* 拉高/CS电平 终止读数据命令 \*/

FLASH\_CS\_HIGH**();**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名 : SPI\_FLASH\_ReadID

\* 描述 : 读取W25X16芯片ID.

\* 返回值 : 芯片ID

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

u32 SPI\_FLASH\_ReadID**(**void**)**

**{**

u32 Temp **=** 0**,** Temp0 **=** 0**,** Temp1 **=** 0**,** Temp2 **=** 0**;**

/\* 拉低CS引脚 选中W25X16 \*/

FLASH\_CS\_LOW**();**

/\* 发送读取芯片ID命令 \*/

SPI\_SendByte**(**0x9F**);**

/\* 从 FLASH W25X16中读取一个字节数据 \*/

Temp0 **=** SPI\_SendByte**(**Dummy\_Byte**);**

/\* 从 FLASH W25X16中读取一个字节数据 \*/

Temp1 **=** SPI\_SendByte**(**Dummy\_Byte**);**

/\* 从 FLASH W25X16中读取一个字节数据 \*/

Temp2 **=** SPI\_SendByte**(**Dummy\_Byte**);**

/\* 释放W25X16 \*/

FLASH\_CS\_HIGH**();**

Temp **=** **(**Temp0 **<<** 16**)** **|** **(**Temp1 **<<** 8**)** **|** Temp2**;** /\* 组合成芯片ID码 \*/

**return** Temp**;**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名 : SPI\_FLASH\_SectorErase

\* 描述 : 擦除W25X16芯片一个段 4K 的数据.

\* 输入 : - SectorAddr：段地址

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_SectorErase**(**u32 SectorAddr**)**

**{**

/\* 使能W25X16 \*/

SPI\_FLASH\_WriteEnable**();**

/\* 段擦除 \*/

/\* 拉低CS \*/

FLASH\_CS\_LOW**();**

/\* 发送段擦除命令 \*/

SPI\_SendByte**(**SE**);**

/\* 发送要擦除的段地址 \*/

SPI\_SendByte**((**SectorAddr **&** 0xFF0000**)** **>>** 16**);**

SPI\_SendByte**((**SectorAddr **&** 0xFF00**)** **>>** 8**);**

SPI\_SendByte**(**SectorAddr **&** 0xFF**);**

/\* 释放W25X16 \*/

FLASH\_CS\_HIGH**();**

/\* 等待写操作完成 \*/

SPI\_FLASH\_WaitForWriteEnd**();**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名 : SPI\_FLASH\_BulkErase

\* 描述 : 擦除整个W25X16芯片的数据.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SPI\_FLASH\_BulkErase**(**void**)**

**{**

/\* Send write enable instruction \*/

SPI\_FLASH\_WriteEnable**();**

/\* Bulk Erase \*/

/\* 拉低CS \*/

FLASH\_CS\_LOW**();**

/\* 发送芯片擦除命令 \*/

SPI\_SendByte**(**BE**);**

/\* 释放W25X16 \*/

FLASH\_CS\_HIGH**();**

/\* 等待写操作完成 \*/

SPI\_FLASH\_WaitForWriteEnd**();**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT 风驰iCreate嵌入式开发工作室 \*\*\*\*\*END OF FILE\*\*\*\*/