STM32f10x系列SPI通信模块

SPI通信协议概述

SPI全称是串行外设接口（Serial Peripheral Interface），是由Motorola提出的一种全双工同步串行通信接口，通信波特率可以高达5Mbps，但具体速度大小取决于SPI硬件。SPI接口具有全双工操作，操作简单，数据传输速率较高的优点

SPI总线的构成及信号类型

SPI总线只需四条线（如图1所示）就可以完成MCU与各种外围器件的通讯:

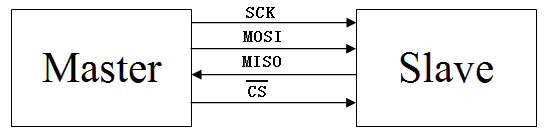
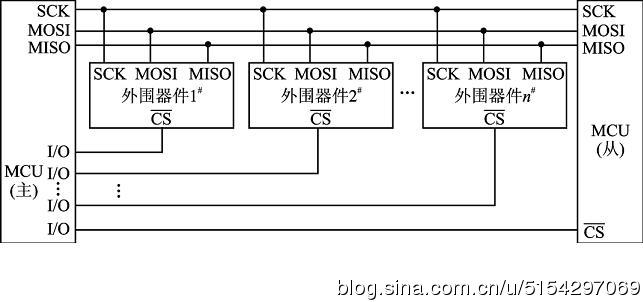
1）MOSI – Master数据输出,Slave数据输入

2）MISO – Master数据输入,Slave数据输出

3）SCK  – 时钟信号,由Master产生

4）/CS  – Slave使能信号,由Master控制。

SPI通信采用主从模式（Master-Slave）架构，一般为一个Master和多个Slave的应用模式。SPI总线构成如图所示。

[](http://c.51hei.com/a/old/up/0/481695319787.jpg) 

SPI初始化:

使用stm32  spi需要以下的步骤

1. 打开spi时钟,gpio管脚时钟
2. 配置管脚复用功能
3. 设置spi的工作模式
4. 使能SPI.

下面是初始化，必须得结构体原型(红色为重点内容)

typedef struct  
{  
  uint16\_t SPI\_Direction;//设置方向   （2线全双工、2线只接受、一线发送、一线接受）

  uint16\_t SPI\_Mode;     //模式      （从或主设备）

  uint16\_t SPI\_DataSize; //宽度,一次发送数据的个数         （8或16位）

  uint16\_t SPI\_CPOL;     //时钟极性     （低或高）

  uint16\_t SPI\_CPHA;     //时钟相位     （第一个或第二个跳变沿）

  uint16\_t SPI\_NSS;      //片选方式     （硬件或软件方式）

  uint16\_t SPI\_BaudRatePrescaler; //波特率预分频(相当于速度)    （从2---256分频）

  uint16\_t SPI\_FirstBit;  //最先发送的位            （最低位，还是最高位在先）

  uint16\_t SPI\_CRCPolynomial; //设置crc多项式        （数字）如7

}SPI\_InitTypeDef;

下面是实例，对SPI2进行的初始化

void SPI2\_Init(void)   
{  
 SPI\_InitTypeDef  SPI\_InitStructure; //spi配置结构体  
 GPIO\_InitTypeDef GPIO\_InitStructure; //管脚配置结构体   
   
 //打开管脚时钟  
 RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_AFIO|RCC\_APB2Periph\_GPIOB, ENABLE);  
 GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_13 |GPIO\_Pin\_14| GPIO\_Pin\_15;  
 GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;  
 GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;  //复用推挽输出  
 GPIO\_Init(GPIOB,& GPIO\_InitStructure);//配置管脚功能  
   
 //打开spi时钟  
 RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_SPI2 ,ENABLE);  
     
 SPI\_InitStructure.SPI\_Direction = SPI\_Direction\_2Lines\_FullDuplex;//设置为全双工模式  
 SPI\_InitStructure.SPI\_Mode = SPI\_Mode\_Master; //设置为主机模式  
 SPI\_InitStructure.SPI\_DataSize = SPI\_DataSize\_8b; //设置一次传输8位  
 SPI\_InitStructure.SPI\_CPOL = SPI\_CPOL\_High;  
 SPI\_InitStructure.SPI\_CPHA = SPI\_CPHA\_2Edge;  
 SPI\_InitStructure.SPI\_NSS = SPI\_NSS\_Soft;  
 SPI\_InitStructure.SPI\_BaudRatePrescaler = SPI\_BaudRatePrescaler\_16;  
 SPI\_InitStructure.SPI\_FirstBit = SPI\_FirstBit\_MSB;  
 SPI\_InitStructure.SPI\_CRCPolynomial = 7;  
 SPI\_Init(SPI2,& SPI\_InitStructure);

 //使能SPI2  
 SPI\_Cmd(SPI2, ENABLE);    
}

SPI通信:

SPI\_I2S\_SendData(SPI0,data); //使用SPI发送数据 (对应使用spi编号和需要发送的数据)

while (SPI\_I2S\_GetFlagStatus(OLED\_SPI, SPI\_I2S\_FLAG\_RXNE) == RESET) //等待发送成功