# 前言

* 通信的目的：将一个设备的数据传送到另一个设备，扩展硬件系统
* 通信协议：制定通信的规则，通信双方按照协议规则进行数据收发



* 全双工：通信双方能够同时进行双向通信，一般有两根数据线。比如TX和RX，MOSI和MISO，发送和接收互不影响。
* 半双工：只有一条数据线(IIC、CAN和USB)，
* 时钟信号：高速接收方什么时候需要采集数据，分为同步和异步。IIC和SPI有单独的时钟线，所以他们是同步的，接收方可以在时钟信号的指引下进行采样。剩下的串口，CAN和USB没有时钟线，所以需要双方约定一个采样频率，为异步通信。

CAN协议使用两条差分信号线进行通信，其中一条线是CAN高线，另一条线是CAN低线。这两条线通过电压的差异来表示数据的0和1。CAN协议的物理层只有这两条差分线，因此在一个时刻只能表示一个信号，所以CAN通信是半双工的，即在同一时刻只能有一个节点发送信号，其他节点只能接收信号。

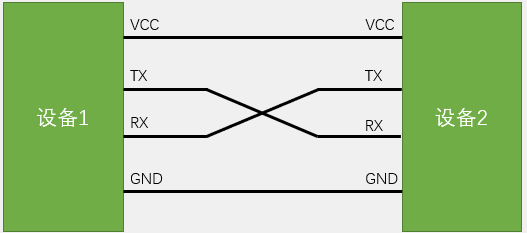
USB协议，数据传输方向，D+只能接另外一个设备的D+，D-接D-，一根数据线为半双工。

USART中的“S”表示同步，只支持时钟输出，不支持时钟输入，是为了兼容别的协议或特殊用途而设计的，并不支持两个USART之间进行同步通信，所以这个功能几乎不会用到，一般更常使用的是UART异步收发器。

# 一、UART

## 1.1 UART协议简介

* UART是通用异步收发传输器(Universal Asynchronous Receiver/Transmitter)的缩写，它是一种常见的串口通信接口。
* 使用差分信号可以抑制共模噪声，可以极大的提高信号的抗干扰特性，所以一般差分信号的传输速度和传输距离都非常高。



* TX与RX要交叉连接
* 当只需单向的数据传输时，可以只接一根通信线
* 当电平标准不一致时，需要加电平转换芯片

电平标准：

* TTL电平：+3.3V或+5V表示1，0V表示0
* RS232电平：-3~-15V表示1，+3~+15V表示0
* RS485电平：两线压差+2~+6V表示1，-2~-6V表示0（差分信号）

串口参数及时序：

* 波特率：串口通信的速率(bit/s)
* 起始位：标志一个数据帧的开始，固定为低电平
* 数据位：数据帧的有效载荷，1为高电平，0为低电平，低位先行
* 校验位(选填)：用于数据验证，根据数据位计算得来
* 停止位：用于数据帧间隔，固定为高电平

无校验位：



有校验位：

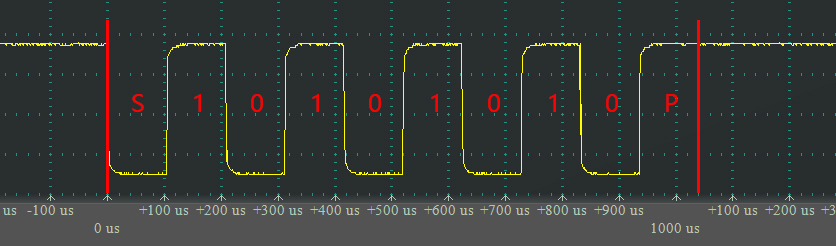


## 1.2 串口通信时序图

串口时序：低位先行，先发送B0(先转换为二进制，然后先画低位波形)

发送一个字节的数据0x55（0101 0101）：8位数据+1位停止位，无校验位

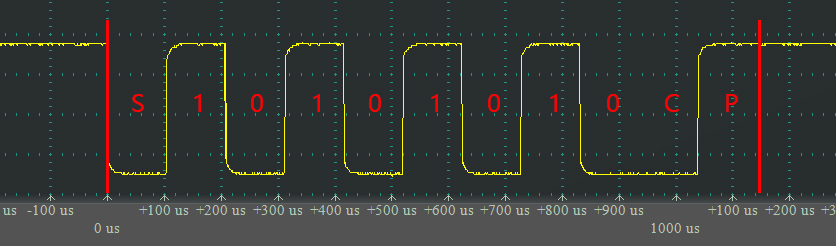
起始位低电平，停止位高电平



0x55，9600，8位数据，1位停止，无校验

有校验位：

偶校验位(低电平0)：



0x55，9600，8位数据，1位停止，偶校验

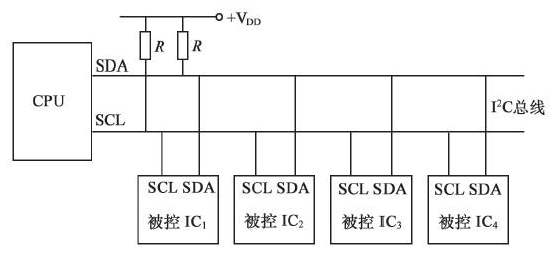
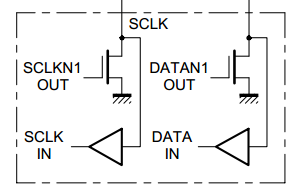
# 二、IIC协议

## 2.1 IIC协议简介

* I2C（Inter IC Bus）是由Philips公司开发的一种通用数据总线
* 两根通信线：SCL（Serial Clock）、SDA（Serial Data）
* 同步，半双工
* 带数据应答
* 支持总线挂载多设备（一主多从、多主多从）

硬件电路：

* 所有I2C设备的SCL连在一起，SDA连在一起
* 设备的SCL和SDA均要配置成开漏输出模式
* SCL和SDA各添加一个上拉电阻，阻值一般为4.7KΩ左右



* 从机的权利比较小，对于SCL线，在任何时刻都只能被动的读取，从机不允许控制SCL线。
* 对于SDA数据线，从机不允许主动发起对SDA的控制。只有在主机发送读取从机的命令后，或者从机应答的时候，从机才能短暂地获取SDA的控制权。

主机的SDA引脚在发送的时候是输出模式，在接收的时候是输入模式。为了避免总线没协调好，导致电源短路的问题。IIC的设计是禁止所有设备输出强上拉的高电平。

设计为弱上拉电阻(4.7K)加开漏输出的模式。

这个模式具有“线与”的功能，只要总线上有一个输出低电平，总线就处于低电平。只有所有设备都输出高电平，总线才输出高电平。

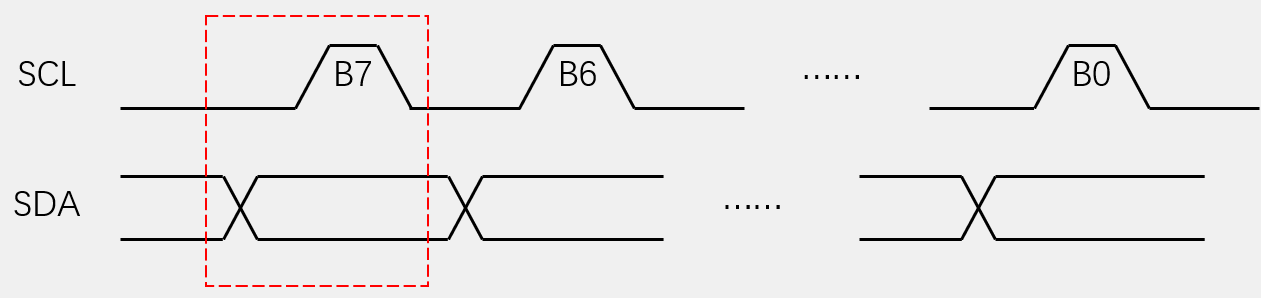
## 2.2 IIC时序基本单元

起始和终止信号均由主机产生，从机不允许产生起始和终止信号，所以在总线空闲状态时，从机必须始终放手。

* 起始条件：SCL高电平期间，SDA从高电平切换到低电平。下降沿，触发起始条件。
* 终止条件：SCL高电平期间，SDA从低电平切换到高电平。上升沿，触发终止条件，回到最初的空闲状态(SDA和SCL均为高电平)
* 除了起始和终止条件，每个时序单元的SCL都是以低电平开始，低电平结束。



* 发送一个字节：SCL低电平期间，主机将数据位依次放到SDA线上（高位先行，先发送B7），然后释放SCL，从机将在SCL高电平期间读取数据位，所以SCL高电平期间SDA不允许有数据变化，依次循环上述过程8次，即可发送一个字节。



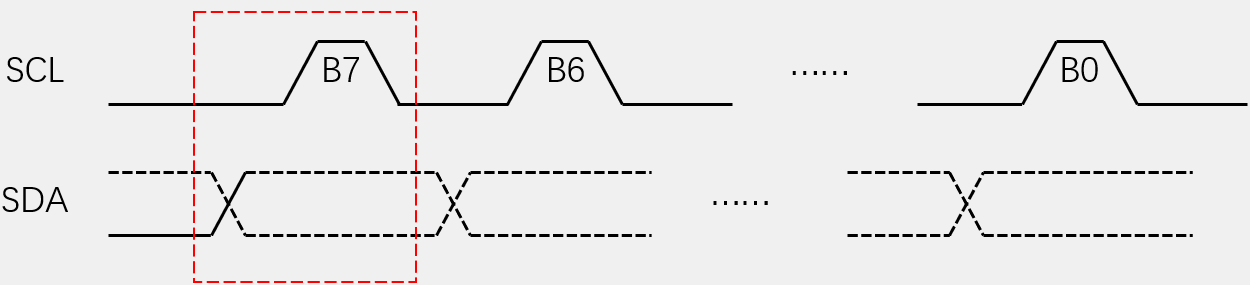
SCL低电平期间：允许SDA的电平发生变化，如果发送1就为高电平，发送0为低电平。

SCL高电平期间：不允许SDA的电平发生变化。

一般在SCL上升沿的时刻，从机就已经读取完成了。从机在上升沿时立刻把数据读走。主机在放手SCL一段时间后(高电平)，就可以继续拉低SCL，传输下一位了。

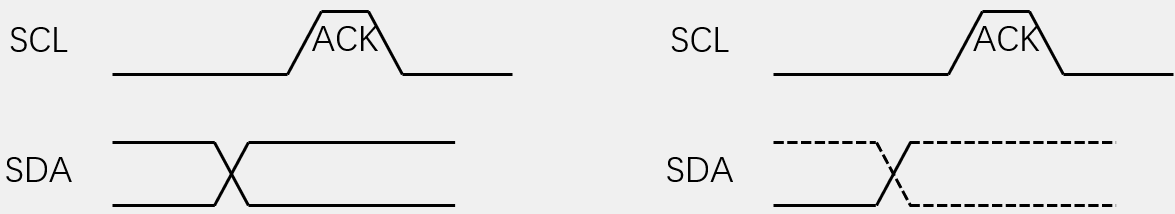
主机也需要在SCL下降沿之后，尽快把数据放在SDA上。

* 接收一个字节：SCL低电平期间，从机将数据位依次放到SDA线上（高位先行），然后释放SCL，主机将在SCL高电平期间读取数据位，所以SCL高电平期间SDA不允许有数据变化，依次循环上述过程8次，即可接收一个字节（主机在接收之前，需要释放SDA，即高电平）

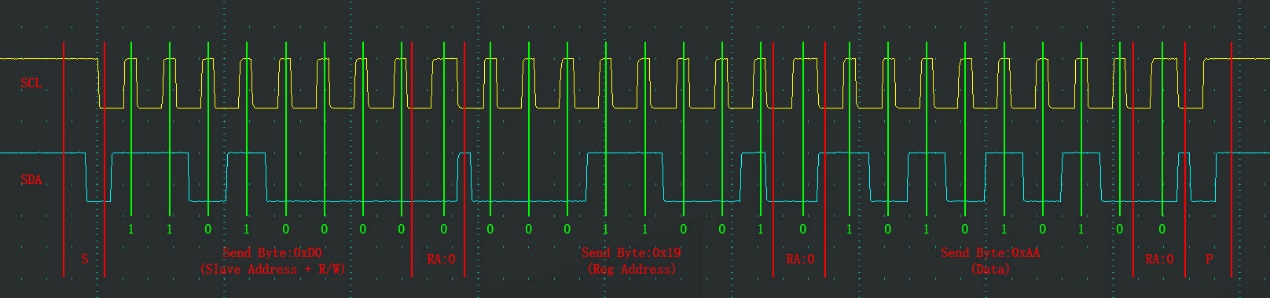


* 发送应答：主机在接收完一个字节之后，在下一个时钟发送一位数据，数据0表示应答，数据1表示非应答
* 接收应答：主机在发送完一个字节之后，在下一个时钟接收一位数据，判断从机是否应答，数据0表示应答，数据1表示非应答（主机在接收之前，需要释放SDA）

接收应答：主机发送完毕后，需要立刻调用接收应答的时序(在SCL低电平期间，SDA切换为高电平)



* 指定地址写
* 对于指定设备（Slave Address），在指定地址（Reg Address）下，写入指定数据（Data）



空闲状态，SDA、SCL均为高电平。

然后主机需要给从机写入数据时，SCL高电平期间，拉低SDA，产生起始条件。

发送一个字节的数据(字节的内容：从机地址7位+1位读写位)

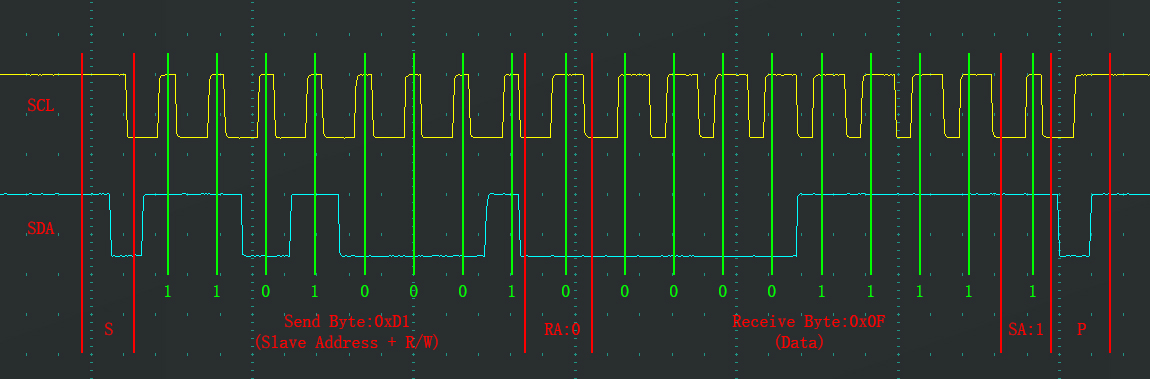
读写位：0表示之后的时序主机要进行写入操作，1表示之后的时序主机要进行读出操作。

应答位RA(Receive ACK, RA)：根据IIC协议的规定，在这个时刻，主机要释放SDA

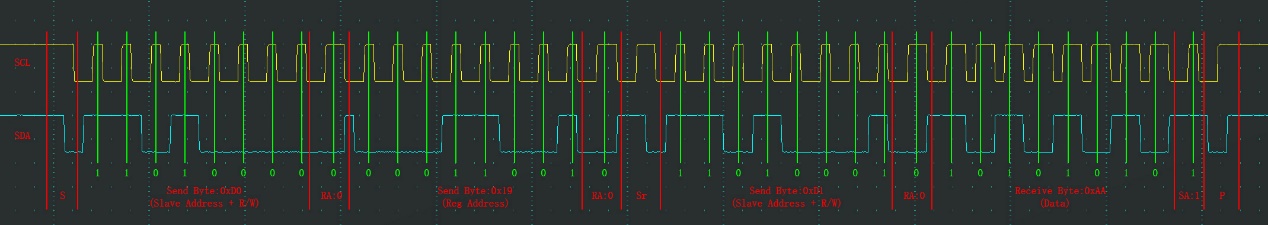
SCL低电平期间，SDA可以变换数据

SCL高电平期间，SDA保持不变，从机读取数据

* 当前地址读
* 对于指定设备（Slave Address），在当前地址指针指示的地址下，读取从机数据（Data）



* 指定地址读
* 对于指定设备（Slave Address），在指定地址（Reg Address）下，读取从机数据（Data）



## 2.3 面试题

1. IIC总线在传送数据过程中共有三种类型信号：开始信号、结束信号和应答信号。

## 2.4 MPU6050

* 6轴：3轴加速度、3轴角速度
* 9轴：3轴加速度、3轴角速度、3轴磁场强度
* 10轴：3轴加速度、3轴角速度、3轴磁场强度、1个气压强度
* 3轴加速度计（Accelerometer）：测量X、Y、Z轴的加速度
* 3轴陀螺仪传感器（Gyroscope）：测量X、Y、Z轴的角速度

通过数据融合(多种传感器的数据)，可以进一步获取欧拉角(姿态角)

# 三、SPI协议

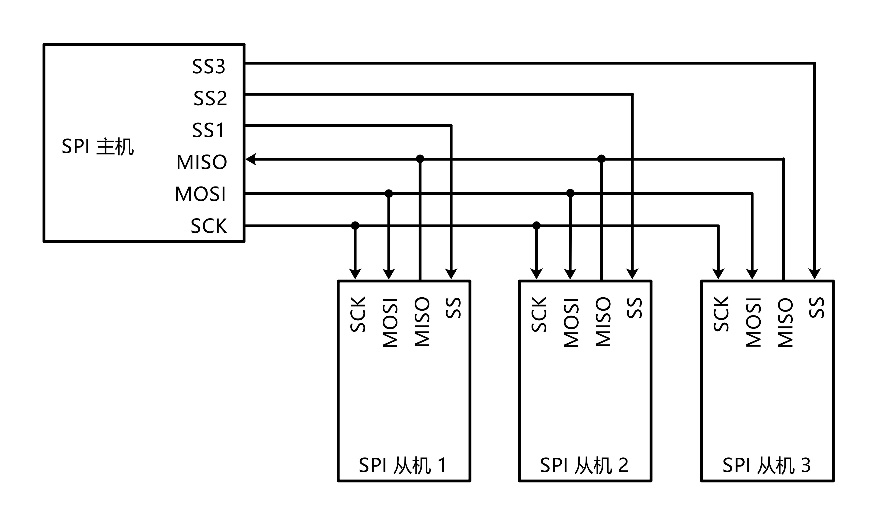
## 3.1 SPI协议简介

* SPI（Serial Peripheral Interface）是由Motorola公司开发的一种通用数据总线
* 四根通信线：SCK（Serial Clock）、MOSI（Master Output Slave Input）、MISO（Master Input Slave Output）、SS（Slave Select）从机选择线，有几个从机，需要几根从机线，一人一根
* 同步，全双工(两根数据线，一根发送，一根接收，设备双方可以同时进行数据通信)
* 支持总线挂载多设备（一主多从）

SPI通信的基础是交换一个字节：有了交换一个字节，就可以实现发送一个字节、接收一个字节，和发送同时接收一个字节，这三种功能。

硬件电路：

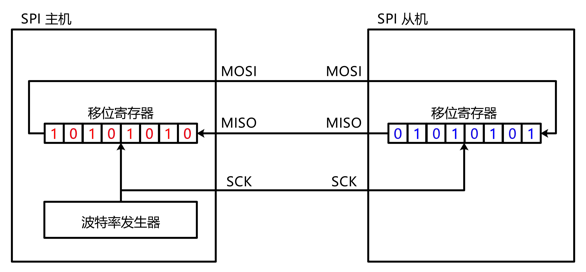
* 所有SPI设备的SCK、MOSI、MISO分别连在一起
* 主机另外引出多条SS控制线，分别接到各从机的SS引脚
* 输出引脚配置为推挽输出，输入引脚配置为浮空或上拉输入



从机SS引脚，输入低电平代表被选中进行数据通信，如果只有一个从机设备，SS引脚可以直接接地。

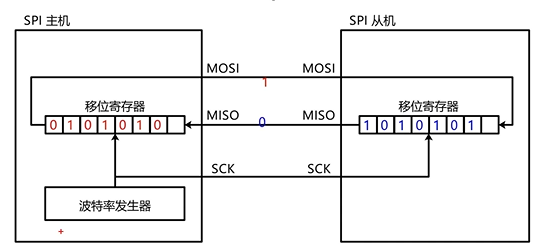
* 从机设备的DI：相当于MOSI引脚
* 从机设备的DO：相当于MISO引脚
* CS片选：其实就是SS

移位示意图：高位先行，每来一个时钟，移位寄存器都会向左进行移动一位

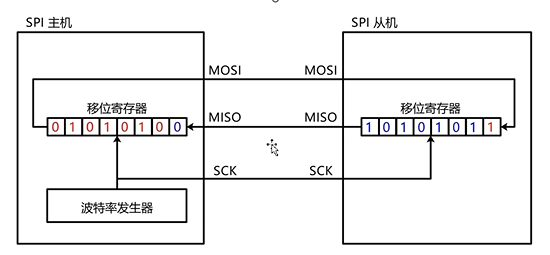


原理：在波特率发生器时钟的上升沿，所有移位寄存器向左移动一位，移出的位放到引脚上。波特率发生器的下降沿，引脚上的位，采样输入到移位寄存器的最低位。

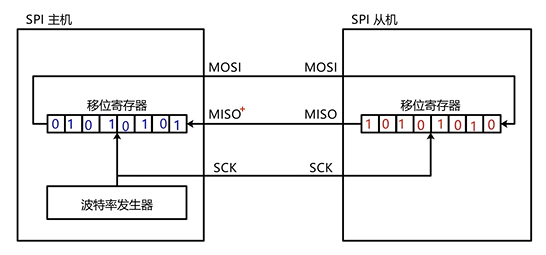
在波特率发生器的上升沿：移出一位放到引脚上。



在波特率发生器的下降沿：主机和从机都会进行数据采样输入。

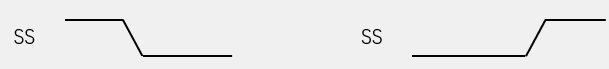


8个时钟后，完成一个字节的交换：一般在接收的时候，主机会统一发送0x00或0xFF，去跟从机换数据。



## 3.2 SPI时序基本单元

* 起始条件：SS从高电平切换到低电平
* 终止条件：SS从低电平切换到高电平
* 在通信期间：SS始终保持在低电平，在从机SS未被选中时，从机的MISO引脚必须关断输出，也就是设置为高阻态。



CPOL:：Clock Polarity，时钟极性

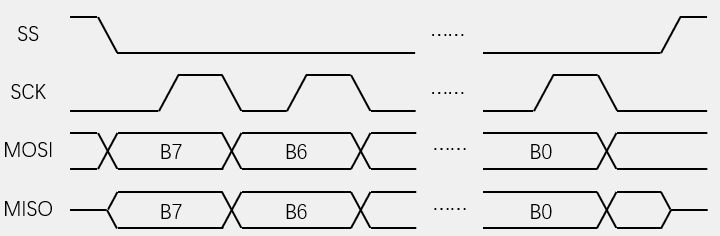
CPHA：Clock Phase，时钟相位

每一位可以配置为1或0，总共组合起来，共有4种模式。在实际使用中，只需要其中的一种即可。

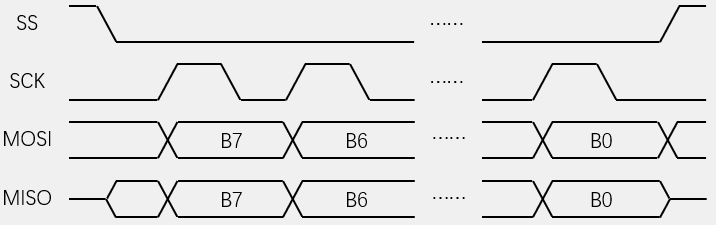
注意：

CHPA表示的是时钟相位，决定第一个时钟采样移入还是第二个时钟采样移入。并不能单独决定上升沿采样还是下降沿采样。

* 交换一个字节（模式0）
* CPOL=0：空闲状态时，SCK为低电平
* CPHA=0：SCK第一个边沿移入数据，第二个边沿移出数据



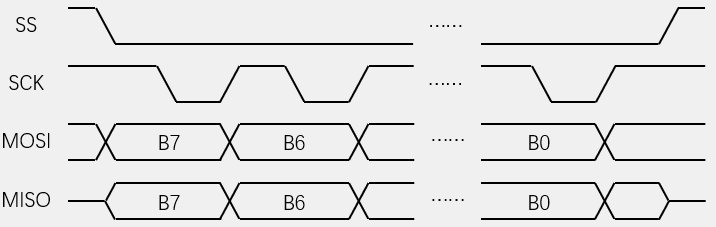
* 交换一个字节（模式1），常用
* CPOL=0：空闲状态时，SCK为低电平
* CPHA=1：SCK第一个边沿移出数据，第二个边沿移入数据



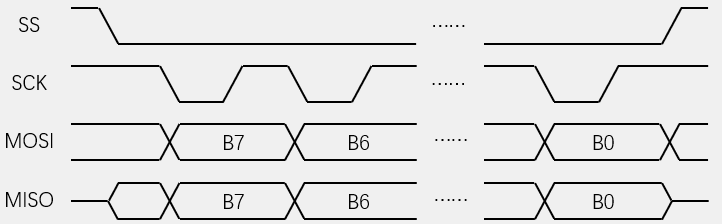
通讯原理：

1. SS高电平时，从机未被选中，MISO用一条中间的线表示高阻态。
2. SS下降沿之后，从机的MISO被允许开启输出，SS上升沿之后，从机的MISO必须置回高阻态。
3. 移位传输的操作：因为CPHA = 1，所以SCK第一个边沿移出数据(所以在SCK第一个上升沿，主机和从机同时移出数据)。主机通过MOSI移出最高位B7，从机通过MISO移出最高位B7。
4. 时钟运行，产生下降沿，此时主机和从机同时移入数据，也就是进行数据采样。主机移出的B7进入从机移位寄存器的最低位B0。从机移出的B7进入主机移位寄存器的最低位。这样一个时钟脉冲产生完毕，一个数据位传输完毕。
5. 通过上述步骤可以完成一个字节的交换。在SS的上升沿，MOSI还可以再变化一次，将MOSI置到一个默认的高电平或低电平，当然也可以不管他。MISO从机必须得置回高阻态，此时如果主机的MISO为上拉输入，那MISO引脚的电平为高电平。如果主机MISO为浮空输入，此时MISO引脚的电平不确定。

* 交换一个字节（模式2）
* CPOL=1：空闲状态时，SCK为高电平
* CPHA=0：SCK第一个边沿移入数据，第二个边沿移出数据



* 交换一个字节（模式3）
* CPOL=1：空闲状态时，SCK为高电平
* CPHA=1：SCK第一个边沿移出数据，第二个边沿移入数据



## 3.3 SPI完整时序图

向SS指定从机设备，发送指令0x06：

