【难度系数】:★★★★★

一、定义

二、单机与多机通信

2.1 单机通信

2.2 多机通信

三、工作模式

四、通信细节与过程

五、硬件原理图

六、读ID

2.1 时序图

2.1 参考源码

七、模拟SPI

7.1 模式0

7.2 模式3

八、应用领域

一、定义

M4高达37.5Mbps，不过很多外围设备往往只支持高达10Mbps，极少部分的设备能够超过10Mbps，例如W25Q128.

发送的时候能够同时接收到数据

10Mbps

串口：全双工异步通信

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，并且在芯片的管脚上只占用四根线，节约了芯片的管脚，同时为PCB的布局上节省空间，提供方便，正是出于这种简单易用的特性，如今越来越多的芯片集成了这种通信协议，如NRF24L01、VS1053、SD卡等。

无线传输　　音频编解码　　存储类

二、单机与多机通信

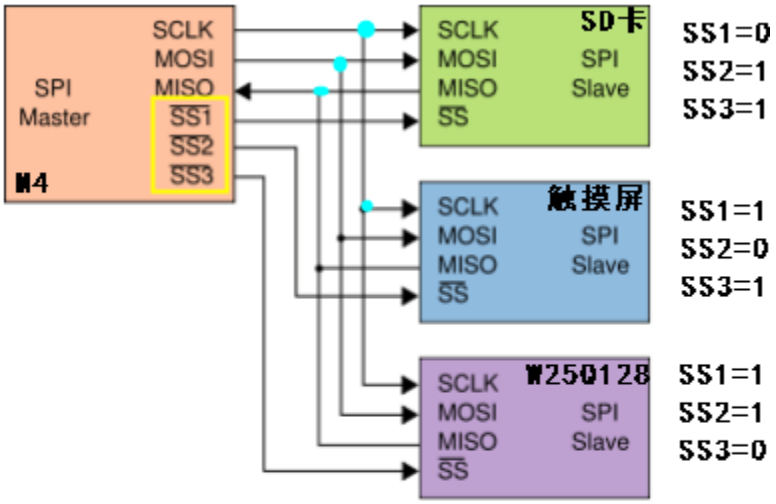
2.1 单机通信

SCLK：串行时钟线，用于数据的同步
MOSI：主机输出数据，从机输入数据
Master Output Slave Input

MISO：主机输入数据，从机输出数据
Master Input Slave Output

/SS(/CS)：(芯)片选（择）引脚，引脚低电平，从机工作有效；引脚高电平，从机工作无效

2.2 多机通信



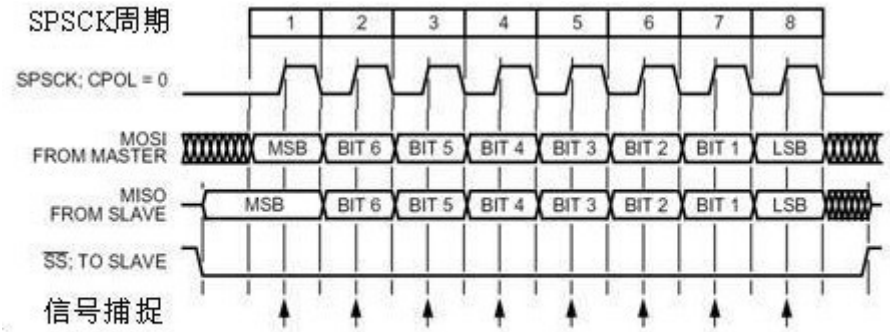
通信的时候，虽然有3根引脚是共用的，唯独片选引脚是不能共用，因为通信的时候，只允许主机跟其中一个从机进行通信！

三、工作模式

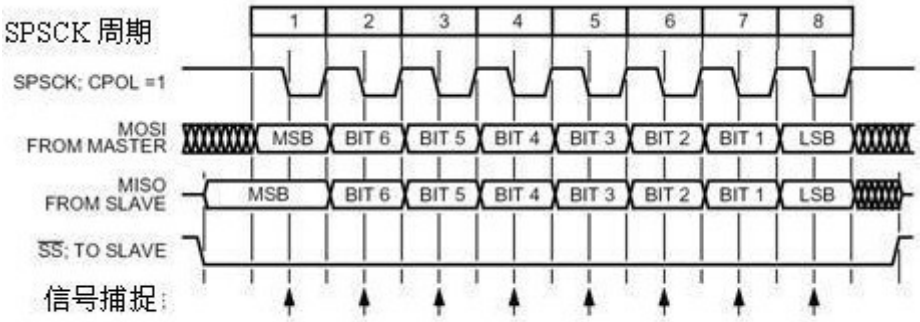
SPI总线有四种工作方式，其中使用的最为广泛的是模式0和模式3方式。

CPOL(Clock Polarity)：时钟极性选择，为0时SPI总线空闲时，时钟线为低电平 ；为1时SPI总线空闲时，时钟线为高电平。

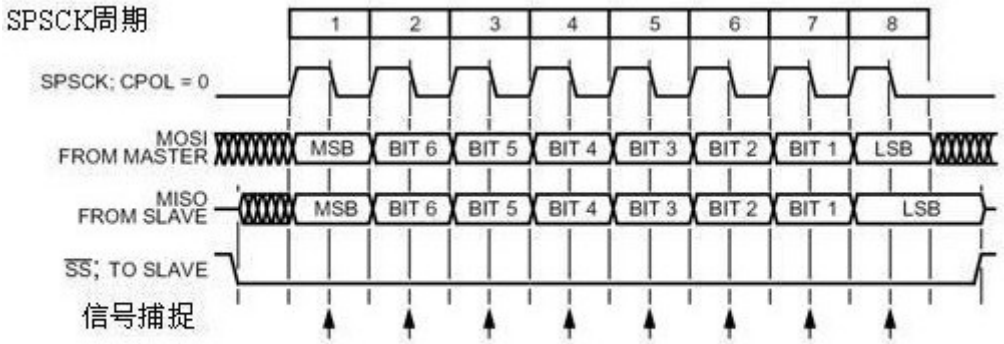
CPHA(Clock Phase)：时钟相位选择，为0时在SCLK第一个跳变沿，主机对MISO引脚电平采样；为1时在SCLK第二个跳变沿，主机对MISO引脚电平采样。



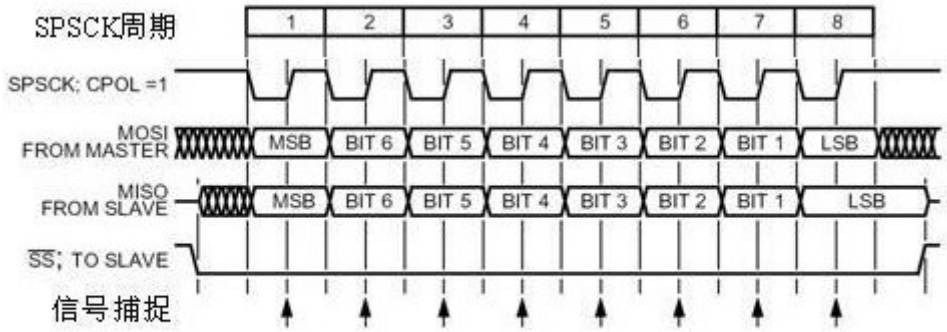
CPHA=0，CPOL=0时的数据/时钟时序图 **模式0**



CPHA=0，CPOL=1时的数据/时钟时序图 **模式1**



CPHA=1，CPOL=0时的数据/时钟时序图 **模式2**



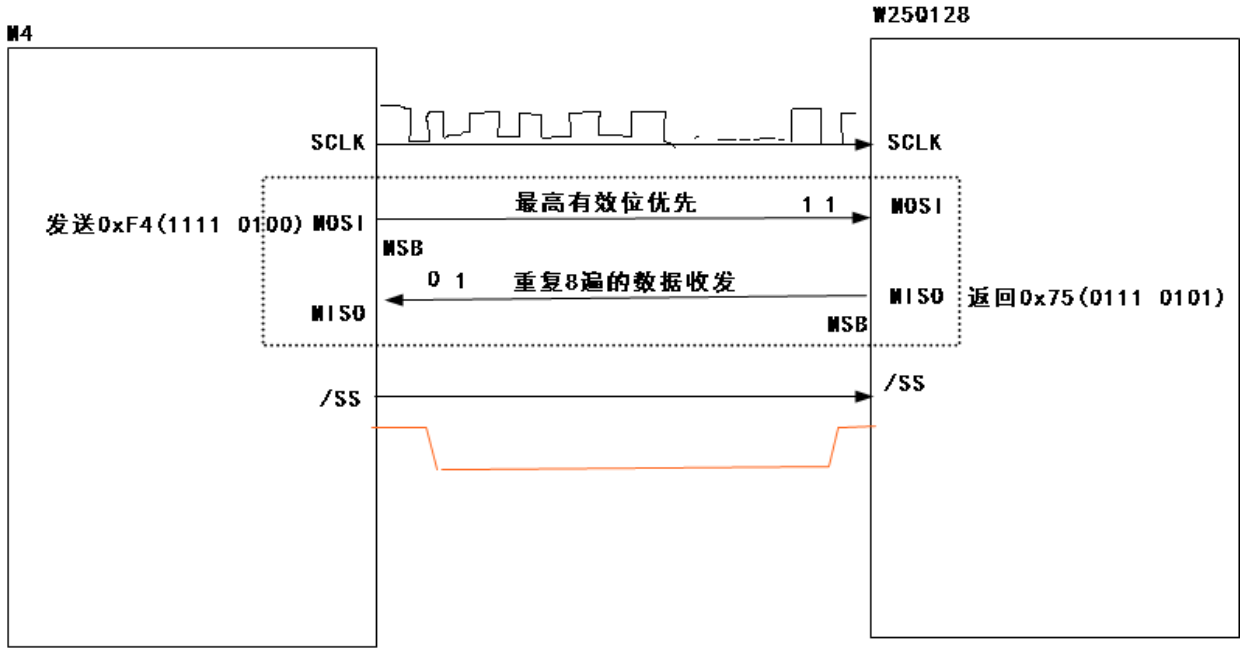
CPHA=1，CPOL=1时的数据/时钟时序图 **模式3**

在项目开放当中，用的比较多是模式0（2.4G无线传输模块：NRF24L01）和模式3（SPI FLASH：W25Q128）。

四、通信细节与过程

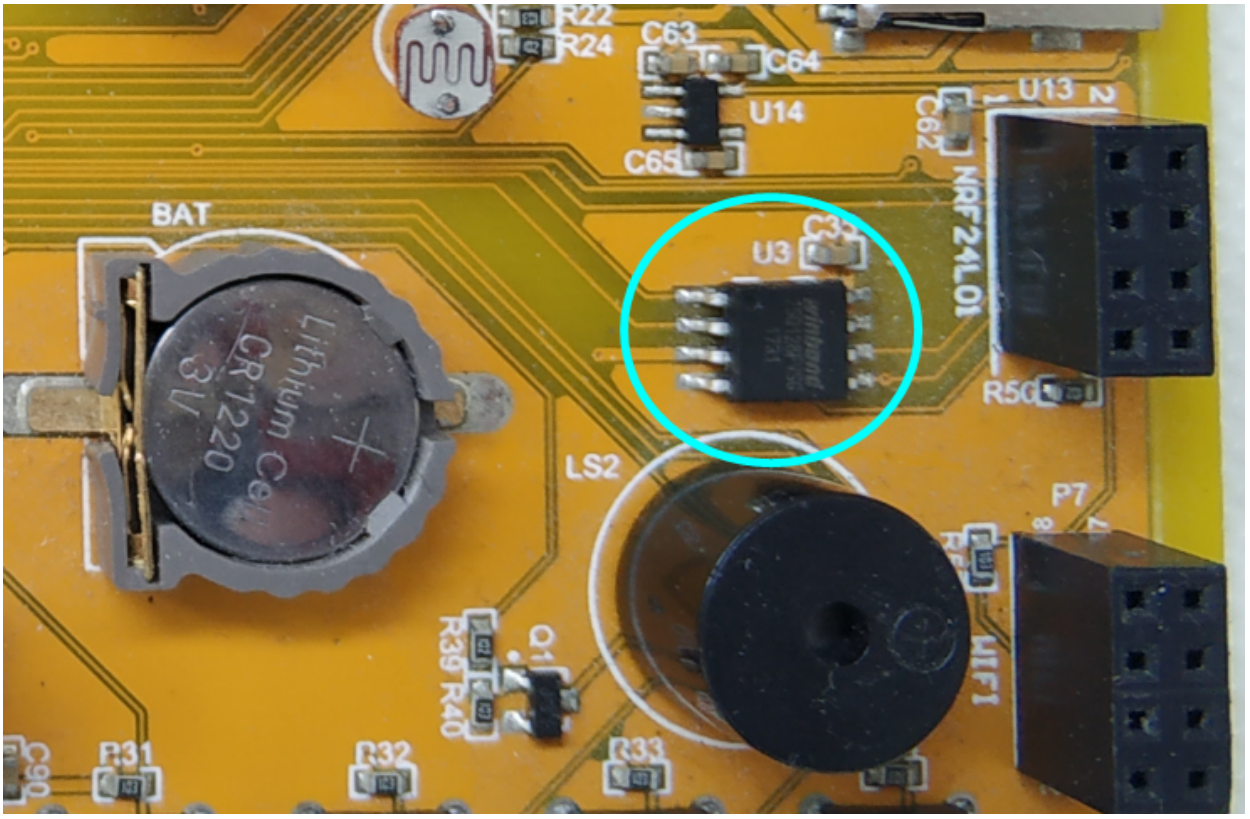
全双工通信：发送数据的时候，能够接收数据；接收数据的时候，能够发送数据，例子，SPI
半双工通信：发送数据的时候，不能接收数据；接收数据的时候，不能发送数据，例子：RS485

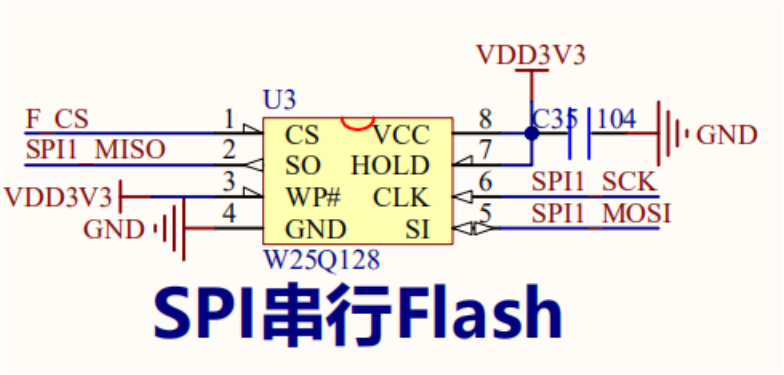
同步通信：发送数据的时候，必须同时接收到数据。例子，SPI
异步通信：发送数据的时候，可以接收数据也可不接收数据。例子，串口



通信开始前，SS引脚变为低电平，W25Q128才工作
通信结束后，SS引脚变为高电平，W25Q128不工作，腾出对主机的使用权给到其他从机

五、硬件原理图

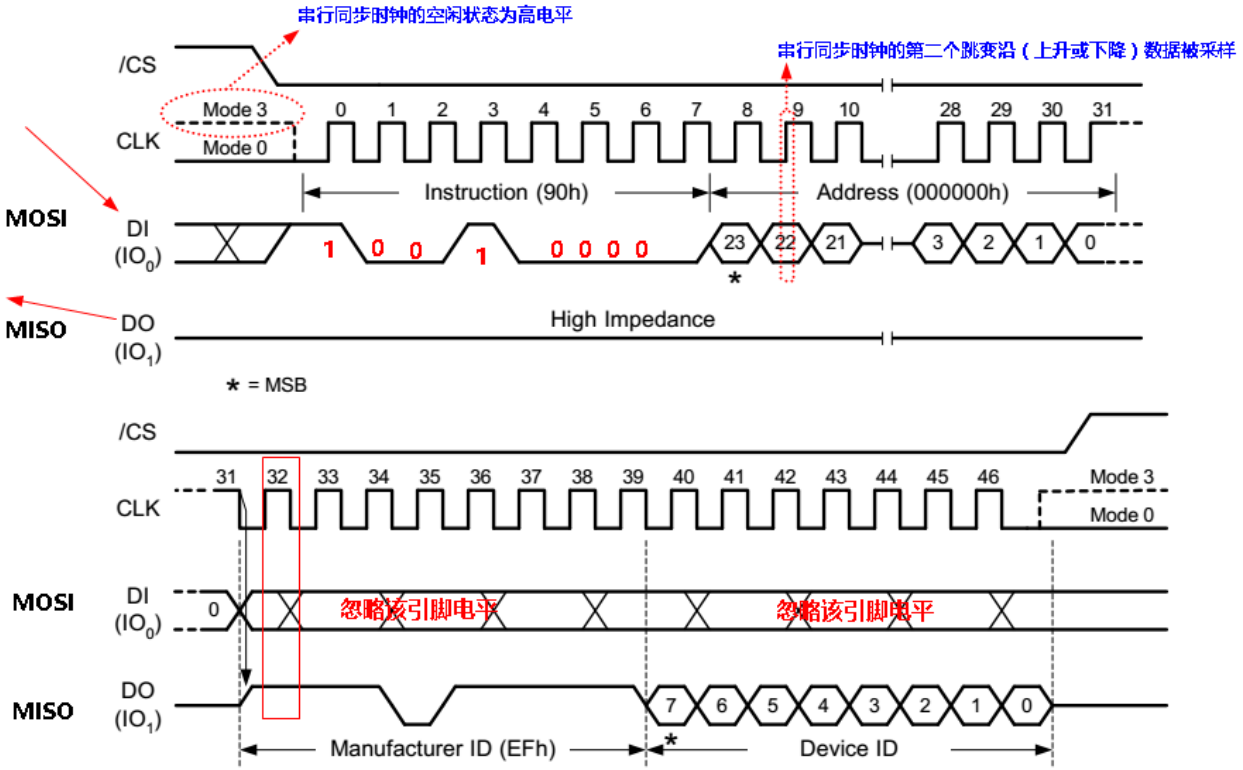

R22
R24
C63
C64
U14
C65
U13
C62
NRF24L01
BAT
U3
C33
R50
LS2
P7
WIF1
R39
R40
Q1
C90
P31
R32
R33


VDD3V3
U3
F_CS
SPI1_MISO
VDD3V3
GND
CS
SO
WP#
GND
VCC
HOLD
CLK
SI
W25Q128
C35
104
GND
SPI1_SCK
SPI1_MOSI
SPI串行Flash

硬件自动控制
SPI1
PB3/SPI1_SCK
PB4/SPI1_MISO
PB5/SPI1_MOSI
PB14/F_CS
软件代码控制

六、读ID

6.1 时序图



MANUFACTURER ID	(MF7-MF0)	
Winbond Serial Flash	EFh	
Device ID	**(ID7-ID0)**	**(ID15-ID0)**
Instruction	**ABh, 90h, 92h, 94h**	**9Fh**
W25Q128BV	17h	4018h

6.1 参考源码

```
uint16_t w25qxx_read_id(void)
{
 uint16_t id=0;

 //片选引脚拉低
 W25QXX_SS=0;

 //发送0x90
 SPI1_SendByte(0x90);

  //发送24bit的地址，全为0
  SPI1_SendByte(0x00);
  SPI1_SendByte(0x00);
```

```
 SPI1_SendByte(0x00);

 //读取厂商id，参数可以为任意参数
 id = SPI1_SendByte(0xFF)<<8;

 //读取设备id
 id|= SPI1_SendByte(0xFF);

 //片选引脚拉高
 W25QXX_SS=1;

 return id;
}
```

七、模拟SPI

在有些情况下没有硬件SPI的支持，只能通过IO口来模拟SPI时序图，参考代码如下：

7.1 模式0

```
void w25qxx_init(void)
{

 //使能端口B的硬件时钟
 RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);

 //配置端口B PB3 PB5 PB14为输出模式
 GPIO_InitStruct.GPIO_Pin=GPIO_Pin_3|GPIO_Pin_5|GPIO_Pin_14; //第3 5 14个引脚
 GPIO_InitStruct.GPIO_Mode=GPIO_Mode_OUT; //输出模
  GPIO_InitStruct.GPIO_Speed=GPIO_High_Speed; //引脚高速工作，收到指令立即工作；缺点：功耗高
  GPIO_InitStruct.GPIO_OType=GPIO_OType_PP; //增加输出电流的能力
  GPIO_InitStruct.GPIO_PuPd=GPIO_PuPd_NOPULL; //不需要上下拉电阻
  GPIO_Init(GPIOB,&GPIO_InitStruct);


 //配置端口B PB4为输入模式
 GPIO_InitStruct.GPIO_Pin=GPIO_Pin_4; //第4个引脚
```

```
 GPIO_InitStruct.GPIO_Mode=GPIO_Mode_IN; //输出模式
 GPIO_InitStruct.GPIO_Speed=GPIO_High_Speed; //引脚高速工作，收到指令立即工作；缺点：功耗高
 GPIO_InitStruct.GPIO_OType=GPIO_OType_PP; //增加输出电流的能力
 GPIO_InitStruct.GPIO_PuPd=GPIO_PuPd_NOPULL; //不需要上下拉电阻
 GPIO_Init(GPIOB,&GPIO_InitStruct);

 //【*】看时序图，只要是输出模式，初始状态要输出对应的电平
 PBout(14)=1; //SS引脚
 PBout(3)=0; //SCLK引脚
 PBout(5)=1; //MOSI引脚，随意，因为时序图没有告诉实际上的电平，模式0~模式3没有描述MOSI引脚相关
}

uint8_t SPI1_SendByte(uint8_t byte)
{
 int32_t i=0;
 uint8_t d=0;

 for(i=7; i>=0; i--)
 {
 //MSB，最高有效位发送数据
 if(byte & (1<<i))
 W25QXX_MOSI =1;
 else
 W25QXX_MOSI =0;

 //设置SCLK引脚为高电平,告诉从机，我开始采集MISO引脚电平
 W25QXX_SCLK=1;
 delay_us(2);

 //主机采集MISO引脚电平
 if(W25QXX_MISO)
 d|=1<<i;

 //设置SCLK引脚为低电平，这个时候，从机会开始读取MOSI引脚电平
 W25QXX_SCLK=0;
 delay_us(2);
 }
 return d;
}
```

```

```

7.2 模式3

```
void w25qxx_init(void)
{

 //使能端口B的硬件时钟
 RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);

 //配置端口B PB3 PB5 PB14为输出模式
 GPIO_InitStruct.GPIO_Pin=GPIO_Pin_3|GPIO_Pin_5|GPIO_Pin_14; //第3 5 14个引脚
 GPIO_InitStruct.GPIO_Mode=GPIO_Mode_OUT; //输出模
  GPIO_InitStruct.GPIO_Speed=GPIO_High_Speed; //引脚高速工作，收到指令立即工作；缺点：功耗高
  GPIO_InitStruct.GPIO_OType=GPIO_OType_PP; //增加输出电流的能力
  GPIO_InitStruct.GPIO_PuPd=GPIO_PuPd_NOPULL; //不需要上下拉电阻
  GPIO_Init(GPIOB,&GPIO_InitStruct);


 //配置端口B PB4为输入模式
  GPIO_InitStruct.GPIO_Pin=GPIO_Pin_4; //第4个引脚
  GPIO_InitStruct.GPIO_Mode=GPIO_Mode_IN; //输出模式
  GPIO_InitStruct.GPIO_Speed=GPIO_High_Speed; //引脚高速工作，收到指令立即工作；缺点：功耗高
  GPIO_InitStruct.GPIO_OType=GPIO_OType_PP; //增加输出电流的能力
  GPIO_InitStruct.GPIO_PuPd=GPIO_PuPd_NOPULL; //不需要上下拉电阻
  GPIO_Init(GPIOB,&GPIO_InitStruct);

 //【*】看时序图，只要是输出模式，初始状态要输出对应的电平
  PBout(14)=1; //SS引脚
  PBout(3)=1; //SCLK引脚
  PBout(5)=1; //MOSI引脚，随意，因为时序图没有告诉实际上的电平，模式0~模式3没有描述MOSI引脚相关
}

uint8_t SPI1_SendByte(uint8_t byte)
{
 int32_t i=0;
 uint8_t d=0;
```

```
 for(i=7; i>=0; i--)
 {
 //MSB，最高有效位发送数据
 if(byte & (1<<i))
 W25QXX_MOSI =1;
 else
 W25QXX_MOSI =0;

 //设置SCLK引脚为低电平，这个时候，从机会开始读取MOSI引脚电平
 W25QXX_SCLK=0;
 delay_us(2);

 //设置SCLK引脚为高电平,告诉从机，主机准备采集MISO电平
 W25QXX_SCLK=1;
 delay_us(2);

 //主机采集MISO引脚电平
 if(W25QXX_MISO)
 d|=1<<i;
 }
 return d;
}
```

八、应用领域

1. 无线传输

EBYTE
亿佰特·物联网应用专家
CHENGDU EBYTE ELECTRONIC TECHNOLOGY
SPI 无线射频模块
nRF24L01+PA+LNA 远距离 抗干扰
挪威 Nordic 原装 nRF24L01+ 射频芯片，PA+LNA，
工业级产品，距离可以达到 5000 米以上，带屏蔽，
抗干扰性能优异，低辐射。
购买样品即送天线
每个样品赠送 1 根。每个 ID 限 2 根
全新升级
全网独有
• E01-ML01DP5
• E01-2G4M27D
什么是 SPI 硬件模块?
MCU
SPI 无线模块是纯硬件的无线模块，其接口采用 SPI 通信方式，需要外接 MCU，对 MCU 进行编程配置模块芯片的寄存器参数，完成数据通信。

2. 高速存储

SanDisk
Ultra
80 MB/s
32 GB
SanDisk

3. 读写器

稳定处理器、感应速度快

读卡器内置微处理器，感应速度快，不易被干扰

如您无法正常过闸
请到客服中心处理
福州地铁
Fuzhou Metro
2019/04/26
预留
YL
车费：2.00
/Fare
余额：37.60
/Balance
投入
Insert Token
单程票
Single-Journey Ticket
通行线 TOURSLINE

4. 音频解码

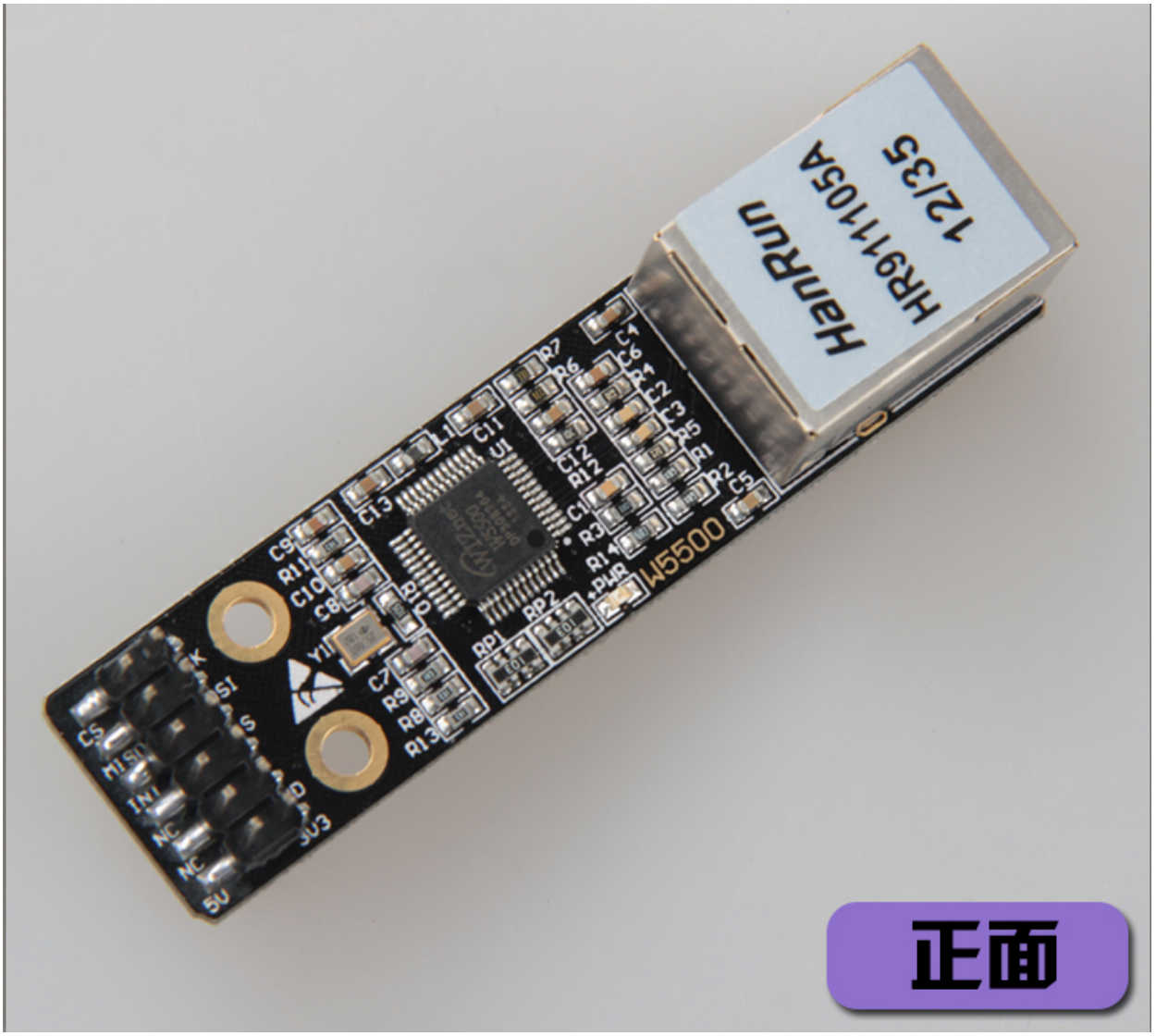
5. TFT-显示屏

6. 墨水屏

7.5寸电子墨水屏
黑白
SPI通信
低功耗
25℃
July 24
12:32 PM
WAVESHARE
share awesome hardware

7. 触摸屏

7. 网络通信模块


HanRun
HR911105A
12/35
W5500
正面