目录

第12章 IIC 总线

12.1 IIC 总线概述

通信特征

UART：异步串行全双工

SPI：同步串行全双工

IIC：同步串行半双工

12.1.1 IIC 总线介绍

I²C (Inter－Integrated Circuit)总线产生于在 80 年代， 由 PHILIPS 公司开发的两线式串行总线，用于连接微控制器及其外围设备，最初为音频和视频设备开发。I²C 总线两线制包括：串行数据 SDA（Serial Data）、串行时钟 SCL（Serial Clock）。时钟线必须由主机（通常为微控制器）控制，主机产生串行时钟（SCL）控制总线的传输方向，并产生起始条件和停止条件。I²C 总线上有主机(MCU)和从机(片外外设，如 AT24C02)之分，可以有多个主机和多个从机。从机永远不会主动给主机发送数据。器件发送数据到总线上，则定义为发送器，器件接收数据则定义为接收器。主器件和从器件都可以工作于接收和发送状态。

同步串行半双工

时钟线控制数据的收发

时钟线由主机控制 主机控制着数据的收发

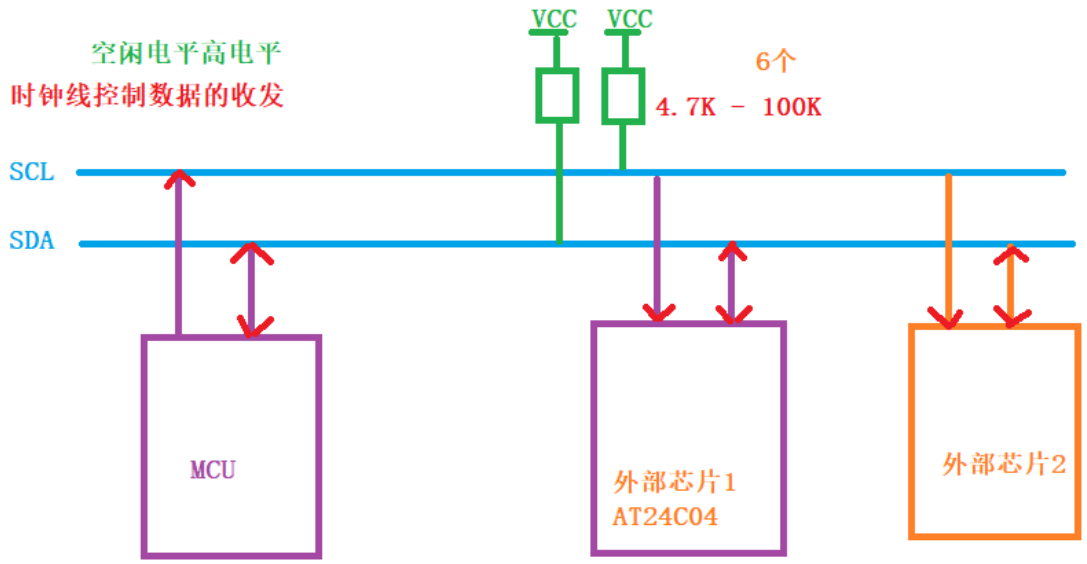
主机控制着数据的传输方向 写：主机到从机 读：从机到主机

主机产生停止条件和起始条件(通信由主机决定)

主机 → 从机

从机 → 主机

12.1.2 IIC 总线物理拓扑图



一主多从

一对一

12.1.3 IIC 总线主从设备通信

主机怎么找到从机？

SPI：主机通过拉低片选管脚来决定进行通信的从机

IIC：在同一个 IIC 总线上每个设备都有一个器件地址，并且这个地址唯一 7 位 10 位

器件地址组成通常有两个部分组成：固定地址(厂家)+可编程地址(程序员或者硬件工程师)

通信之前必须明确从机的器件地址 → 从机芯片的数据手册和原理图结合

以 AT24C02 为例

AT24C02：器件地址 7 位(固定地址 4+可编程地址 3) +方向位 1

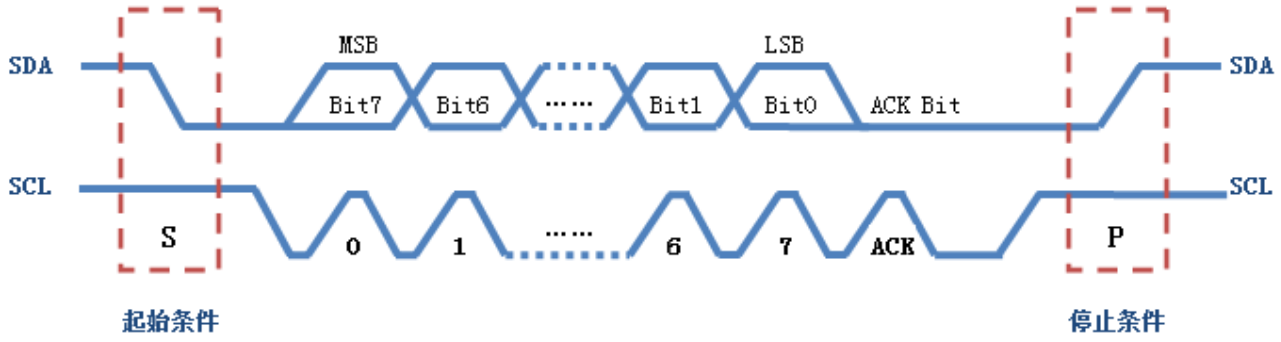
在同一个 IIC 总线可以同时挂载 8 个 AT24C02

12.1.4 IIC 总线与 UART 比较

通讯协议	UART	IIC	SPI
特征	异步串行全双工	同步串行半双工	同步串行全双工
接口	TX RX	SDA SCL	MOSI MISO CS SCK
速度	波特率很多	100Khz 400Khz 3.4Mhz	Mhz 级以上
数据帧格式	起始位+数据位+校验位+停止位	起始条件+位传输+应答+停止条件	四种 MODE0 - 3
主从设备通讯	没有主从之分	有主从之分	有主从之分
总线结构	一对一	一对多	一对多

12.2 IIC 总线数据帧

12.2.1 IIC 数据帧的格式



在时钟线高电平期间：数据线不允许发生变化，如果数据线发生变化，就意味产生了控制信号

数据帧格式：起始条件+位传输(8 位，先高后低)+应答位(0 应答 1 非应答) + 停止条件

1. 起始条件：在时钟线高电平时，数据线产生下降沿
2. 位传输部分

   时钟线产生下降沿(低电平)时，发送方准备数据

   时钟线产生上升沿时，接收方采集数据
3. 应答位：0 应答 1 非应答
4. 停止条件：在时钟线高电平时，数据线产生上升沿

通信过程

1. 主机发出起始条件(开始通信，在 IIC 总线上所有从机读取到开始条件后就会被激活)
2. 主机发出器件地址+读写方向位(寻找对应从机，在 IIC 总线上的所有从机就会读取器件地址，跟本身地址进行对比，如果匹配，从机就会发送一个应答信号给主机，如果不匹配继续休眠)。(找到要进行通信

的从机)

3. 主机决定发送数据还是读取数据
4. 主机发出停止条件(结束通信)

发送方发送一个字节数据后，接收方必须回应

MCU → AT24C02 at24c02 回应

AT24C02 → MCU mcu 回应

12.2.2 标准 IIC 时序

IIC 通信速度：标准速度 100Khz 快速 400Khz 高速 3.4Mhz

时钟线一个周期发送一个位

如下图可知 SCL 为高电平时间 4us SCL 为低电平时间 4.7us 9us 10us

发送一个位需要 10us

1us = 1Mhz

10us = 100Khz

已知需要的通信速度 400Khz

知道发送一位时间要 1/400Khz = 2.5us

SCL 高电平多长时间 1.25us

SCL 低电平多长时间 1.25us

100 kHz I²C-COMPATIBLE INTERFACE COMMUNICATION TIMING CHARACTERISTICS

T_A = –40°C to 85°C, 2.4 V < V_{CC} < 2.6 V; typical values at T_A = 25°C and V_{CC} = 2.5 V (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_r	SCL/SDA rise time				1	µs
t_f	SCL/SDA fall time				300	ns
$t_{w(H)}$	SCL pulse width (high)		4			µs
$t_{w(L)}$	SCL pulse width (low)		4.7			µs
$t_{su(STA)}$	Setup for repeated start		4.7			µs
$t_{d(STA)}$	Start to first falling edge of SCL		4			µs
$t_{su(DAT)}$	Data setup time		250			ns
$t_{h(DAT)}$	Data hold time		0			ns
$t_{su(STOP)}$	Setup time for stop		4			µs
t_{BUF}	Bus free time between stop and start		4.7			µs
f_{SCL}	Clock frequency				100	kHz

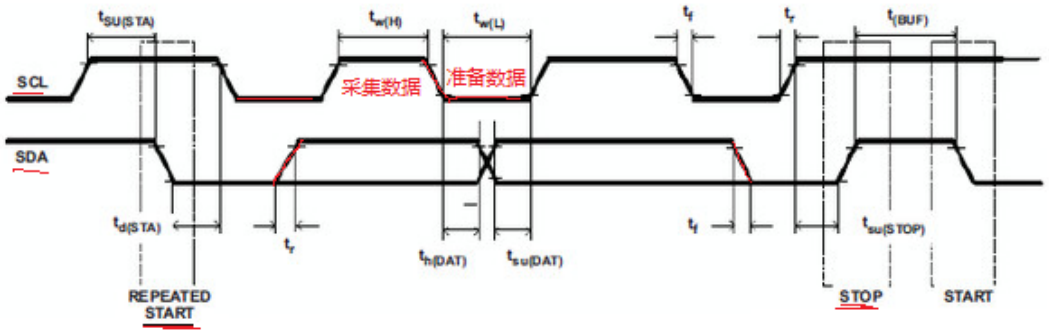


Figure 1. I²C-Compatible Interface Timing Diagrams

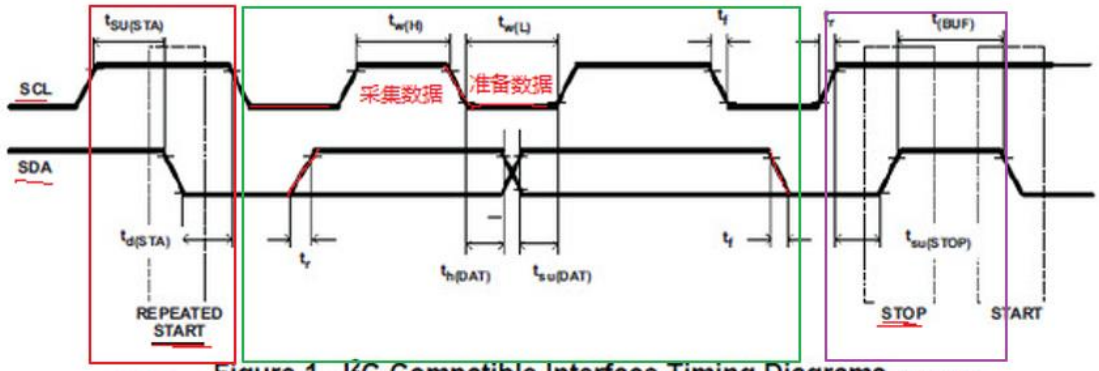
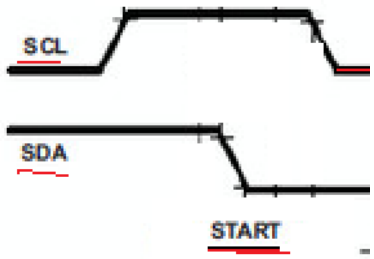


Figure 1. I²C-Compatible Interface Timing Diagrams

12.2.2.1 起始条件



起始条件：时钟线高电平期间，数据线产生下降沿

```
起始条件伪代码
SDA = 1;
SCL = 1;
延时 4.7us;  -- 起始条件的建立时间
SDA = 0;
延时 4us;  -- 起始条件的保持时间
SCL = 0;
```

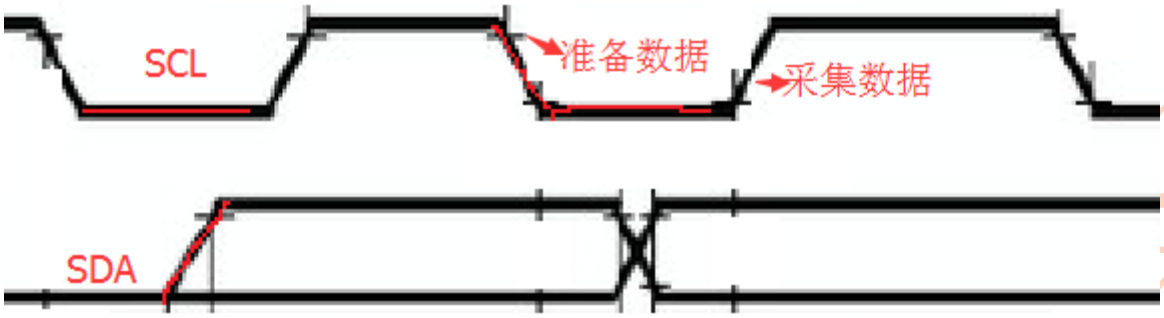
12.2.2.2 停止条件

停止条件：时钟线高电平时，数据线上升沿

```
停止条件伪代码
SDA = 0;
```

```
SCL = 1;
延时 4us;  -- 停止条件的建立时间
SDA = 1;
延时 4us;  -- 停止条件的保持时间
```

12.2.2.3 位传输



时钟线产生下降沿时，发送方准备数据
时钟线产生上升沿时，接收方采集数据
主机发送一位数据给从机

```
主机发送一位数据给从机伪代码
SCK = 0;
SDA = 0/1;//主机准备数据
延时 4us; --低电平保持 4us
SCK = 1;//产生上升沿  从机采集数据
延时 4us; --给时间从机采集数据

主机发送 8 位数据给从机伪代码
for(循环 8 次)
{
    SCK = 0;
    SDA = 0/1;//主机准备数据
    延时 4us; --让数据稳定在数据线上
    SCK = 1;//产生上升沿  从机采集数据
    延时 4us; --给时间从机采集数据
}
```

主机读取从机的一位数据

```
主机读取从机的一位数据伪代码
SCK = 0;//从机准备数据
延时 4us; //让数据稳定在数据线上
SCK = 1;//产生上升沿  主机采集数据
if(SDA)
延时 4us;--给高电平时间保持 4us

主机读取从机的 8 位数据伪代码
for(循环 8 次)
{
    SCK = 0;//从机准备数据
    延时 4us; //让数据稳定在数据线上
```

```
        SCK = 1;//产生上升沿  主机采集数据
        if(SDA)
        延时 4us;--给时间主机读取数据
    }
```

12.2.2.4 应答

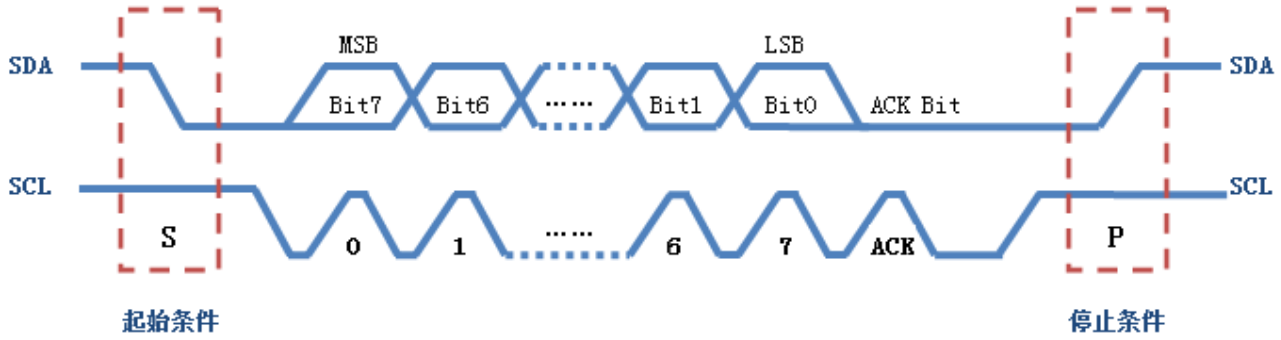
在 IIC 总线上，发送方发送数据给接收方后，接收方要给发送方一个回应

0：应答　　1：非应答

MCU → AT24C02　　AT24C02 发出应答信号

AT24C02 → MCU　　MCU 发出应答信号

接收方发出应答信号



主机发八位数据给从机　　SCL 的电平处于什么电平？

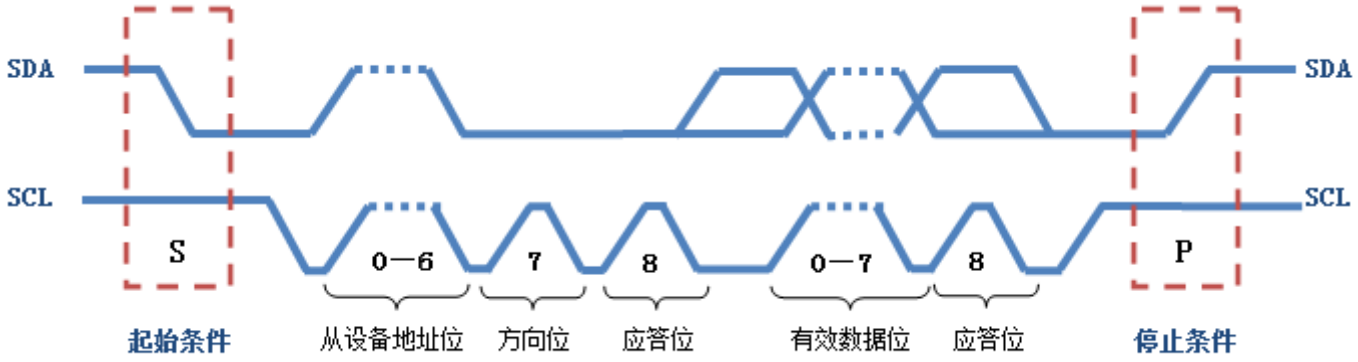
主机发送一个应答信号给从机(发送一个位)

```
    主机发送一个应答信号给从机伪代码
    SCK = 0;//主机准备应答信号
    SDA = ACK(0/1);
    延时 4us;//让应答信号稳定在数据线上
    SCK = 1;//产生上升沿　从机读取应答信号
    延时 4us;//给时间从机读取应答
    SCK = 0;//不能省略
    延时 4us;//保证周期完整性
```

主机读取从机发出的应答信号(读取一个位)

```
    主机读取从机发出的应答信号伪代码
    SCK = 0;//从机准备应答信号
    延时 4us;//让应答信号稳定在数据线上
    SCK = 1;//产生上升沿  主机采集应答信号
    if(SDA)
    延时 4us;
    SCK = 0;
    延时 4us; //保证周期完整性
```

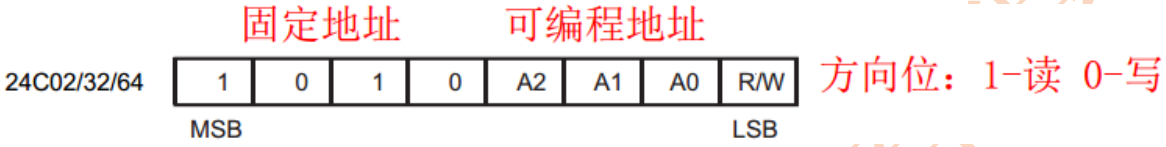
12.2.3 IIC 的寻址方式



器件地址+方向位

AT24C02

器件地址 7 位



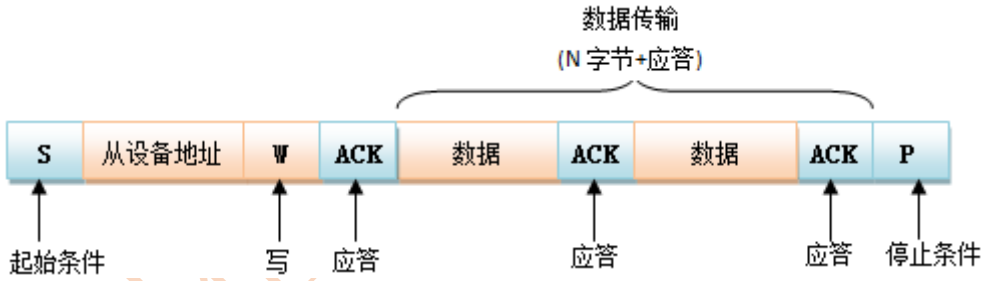
MCU 找到这个 AT24C02：器件地址+读写方向结合 8 位数据

写：0xa0

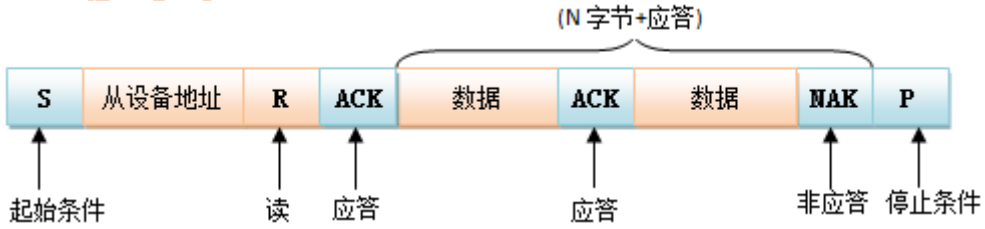
读：0xa1

12.2.4 IIC 的三种通信过程

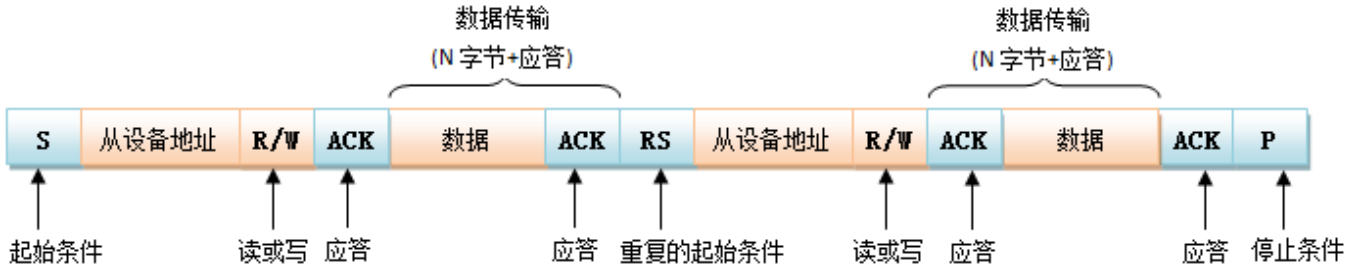
单纯写：



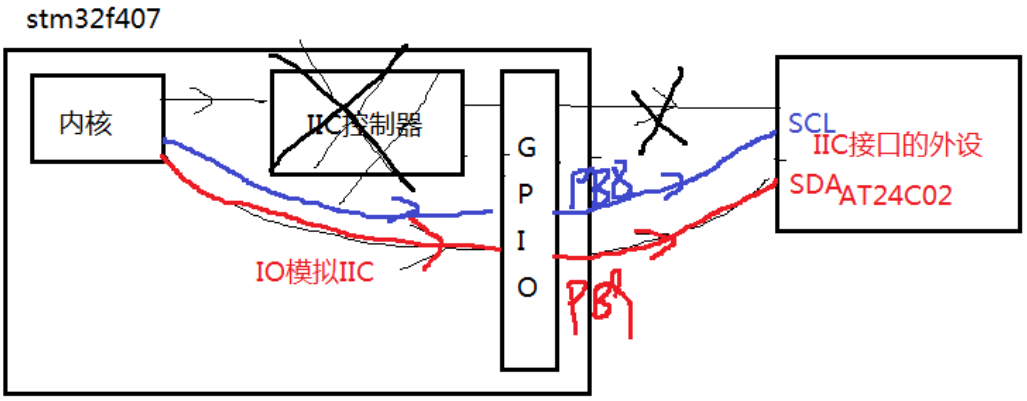
单纯读：



又读又写：



12.3 IO 口模拟 IIC 总线



由于 IIC 控制器存在缺陷，一般都不会采用芯片内部的 IIC 控制器。所以要想采用 IIC 通信跟外设进行数据交流，就需要用 IO 口模拟 IIC 时序。

IIC 总线是两线制通信协议，所以只需要采用两个 IO 口即可，一个 IO 口作为 SCL，另一个作为 SDA

IIC_SCL：输出高输出低 -- 推挽输出/开漏输出

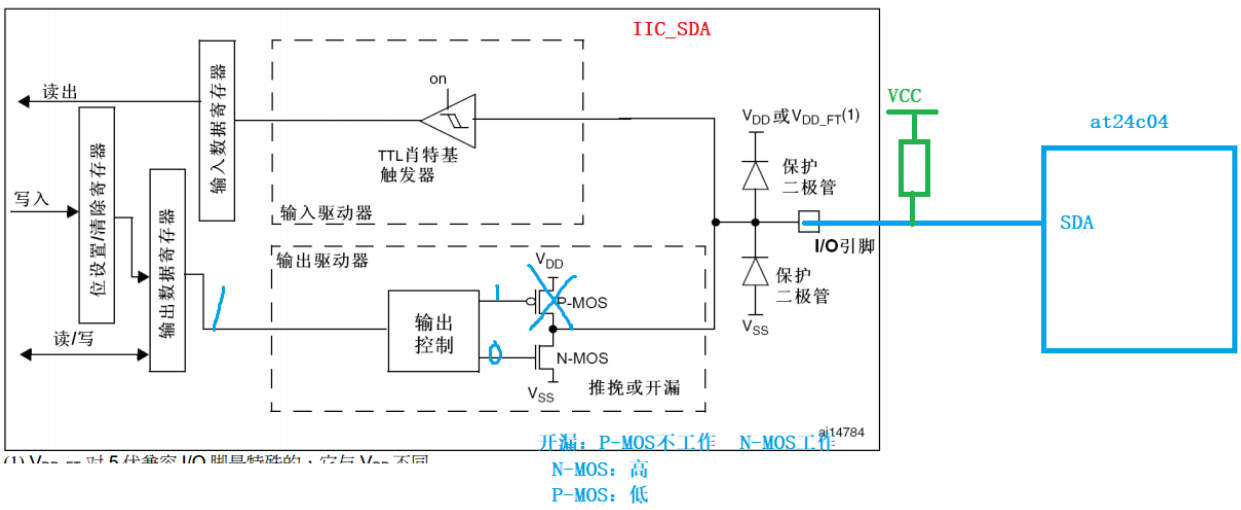
IIC_SDA：方向双向 ----有读又有写

1. 模式切换

   主机发送数据时(写方向)：IIC_SDA 输出模式(推挽输出)

   主机接收数据时(读方向)：IIC_SDA 浮空输入

2. 利用开漏输出特性



当芯片的管脚没有电流流过时，讲芯片管脚稳定在高电平

配置输出模式时，输入并没有关闭

12.3.1 IO 口初始化

IIC_SCL 配置推挽输出
IIC_SDA 配置为开漏输出

12.3.2 起始条件

12.3.3 停止条件

12.3.4 主机发送一个应答信号

12.3.5 主机接收一个应答信号

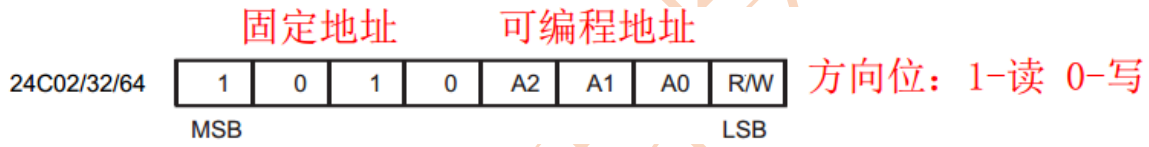
12.3.6 主机发送一个字节数据并读取应答信号

12.3.7 主机读取一个字节数据并回发应答信号

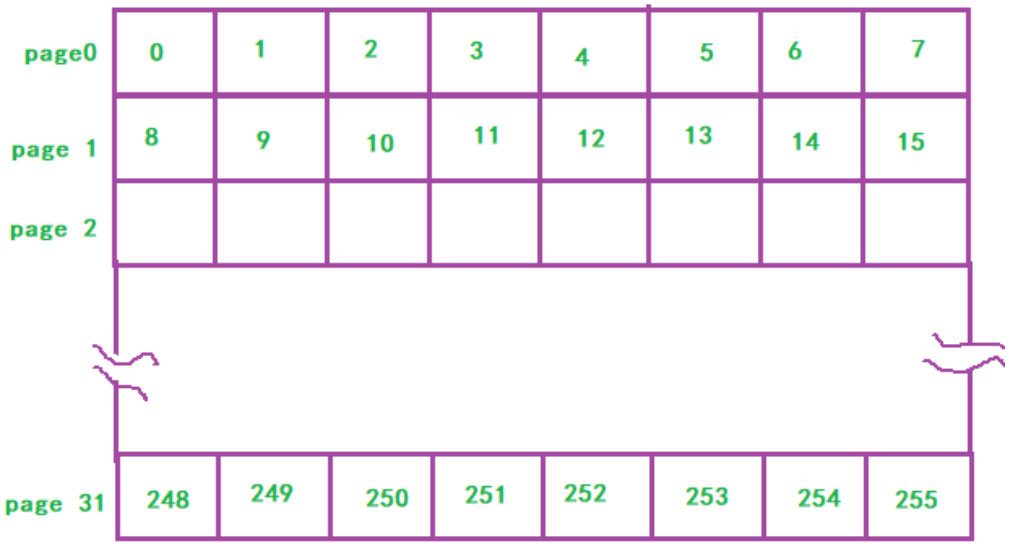
12.4 驱动 AT24C02

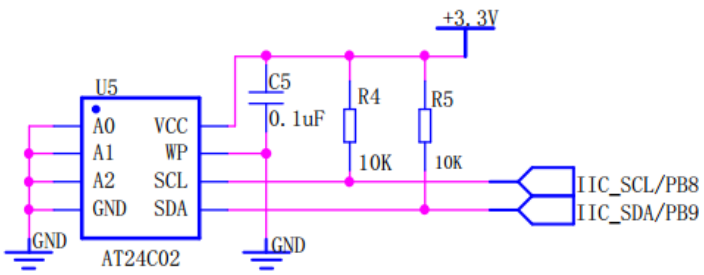
12.4.1 AT24C02 介绍

AT24C02 大小：256 字节
IIC 接口 通信速度
写周期：前一次操作 IIC 总线和下次操作 IIC 总线之间的时间间隔
页写：8 字节



器件地址 7+方向位





由上图知

写方向：0xa0

读方向：0xa1

12.4.2 字节写

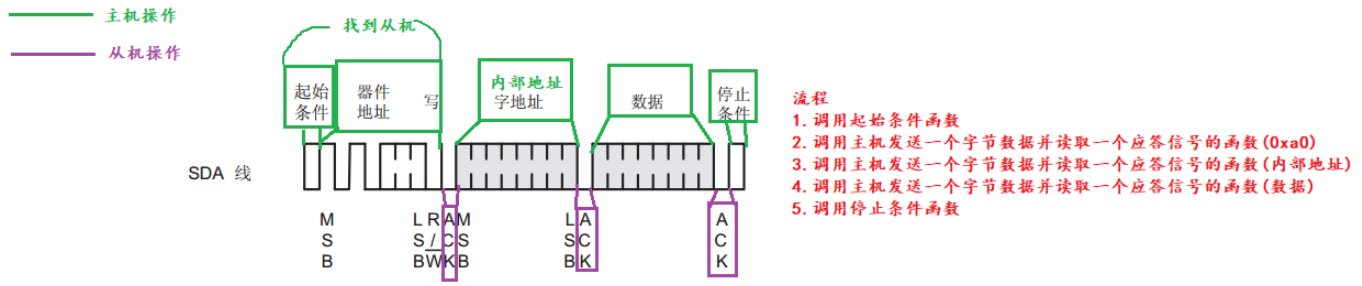


图9 字节写

12.4.3 随机读

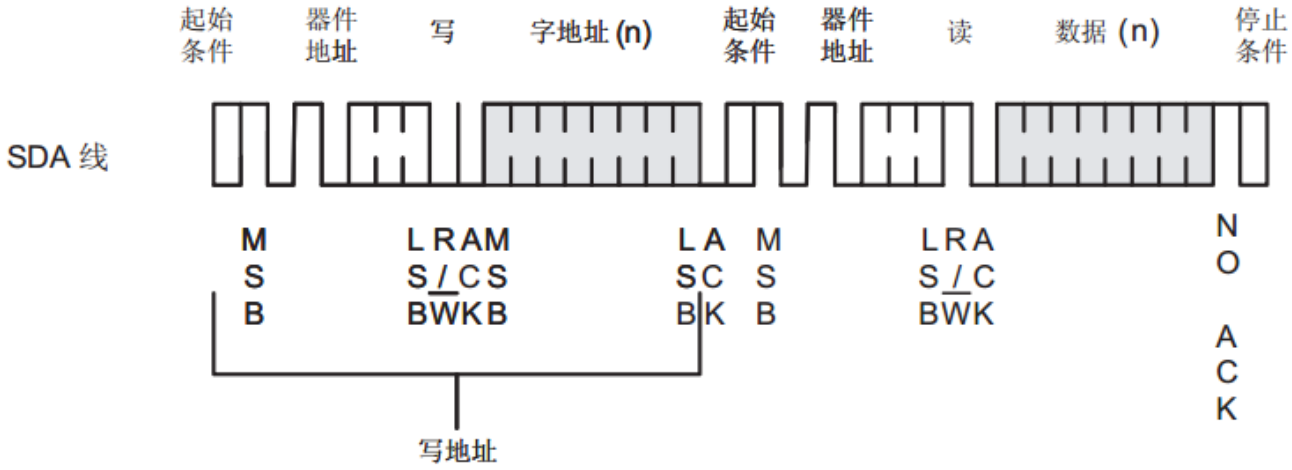


图13 随机读

流程

1. 调用起始条件函数
2. 调用主机发送一个字节数据并读取应答信号的函数(0xa0)
3. 调用主机发送一个字节数据并读取应答信号的函数(内部地址)
4. 调用起始条件函数
5. 调用主机发送一个字节数据并读取应答信号的函数(0xa1)
6. 调用主机接收一个字节数据并发送一个应答信号的函数(非应答)
7. 调用停止条件函数

12.4.4 顺序读

12.4.5 页写

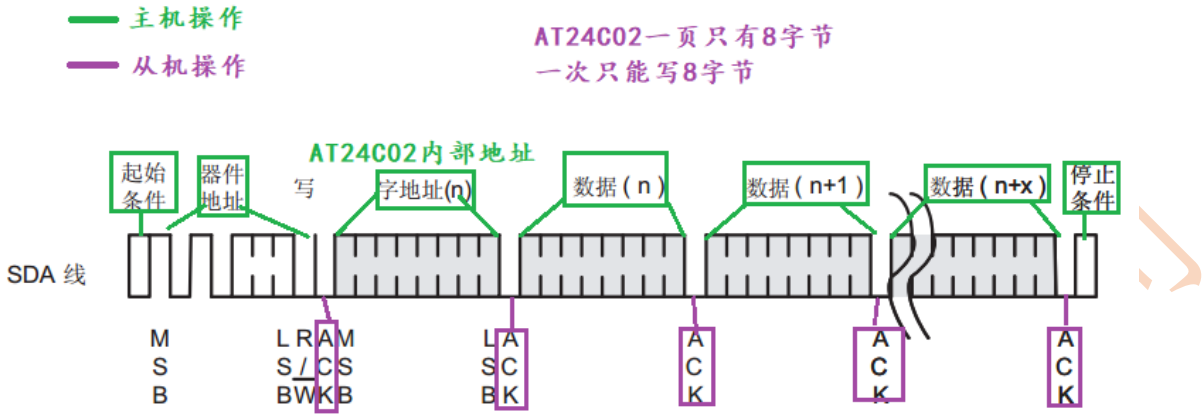


图10页写

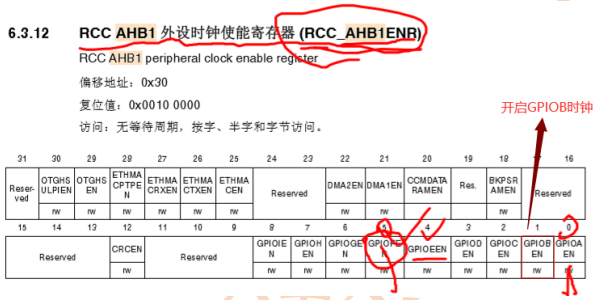
12.4.6 跨页写

程序代码实现流程：

Iic_Pin_Init（）

1、//打开 GPIOB 时钟

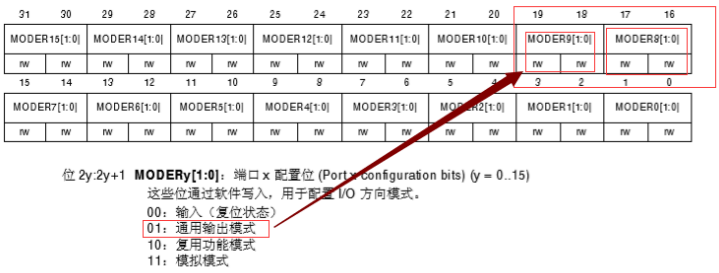
RCC->AHB1ENR |= (0x1 << 1);



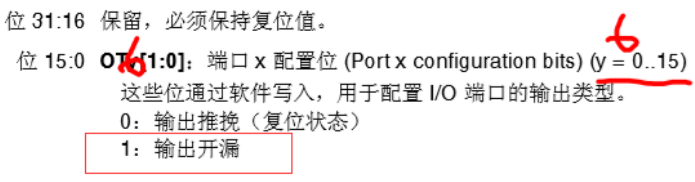
2、 GPIOB->MODER &= ~(0xf << 16);//都是 0 16 到 19

GPIOB->MODER |= (0x5 << 16);//PB8 和 PB9 配置为输出模式

- 0x0000 0280（端口 B）
- 0x0000 0000（其它端口）



3、GPIOB->OTYPER |= (0x1 << 9);//PB9 配置为开漏输出



4、//空闲电平 空闲电平拉高。

IIC_SCL = 1;

IIC_SDA_OUT = 1;

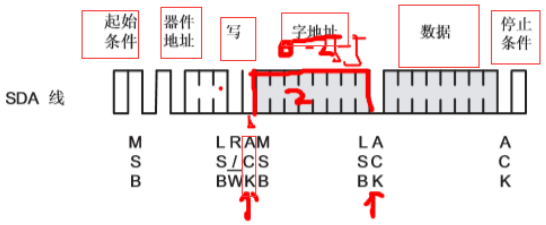


图9 字节写