**十五.ARM裸机学习之I2C通信详解**

2017年12月29日 22:55:26 [MrT\_WANG](https://me.csdn.net/wangweijundeqq" \t "_blank) 阅读数：992更多

所属专栏： [ARM+Linux探索之旅](https://blog.csdn.net/column/details/18007.html" \t "_blank)

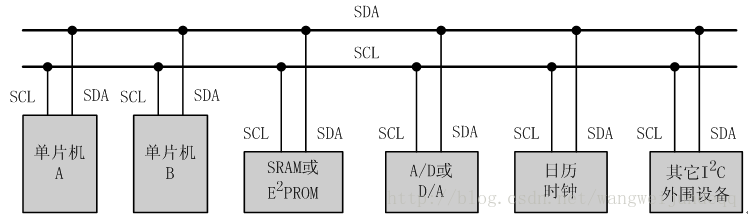
 版权声明：本文为博主原创文章，允许转载请注明。谢谢！ https://blog.csdn.net/wangweijundeqq/article/details/78935464

在博客STM32中对IIC做过整理，这里截取复习哈 2017/12/27 22:52

**常用的串行总线协议：**

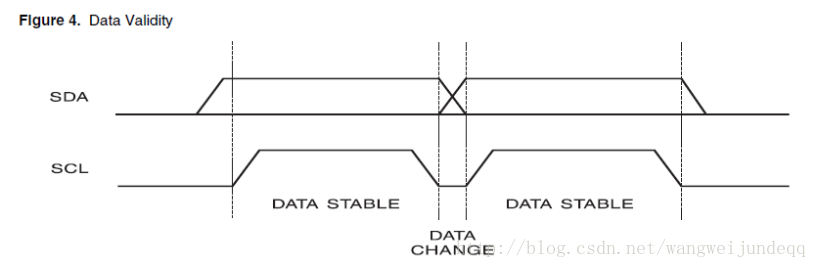
常用的微机与外设之间进行数据传输的串行总线主要有**I2C总线、SPI总线和SCI总线。**   
其中I2C总线以**同步串行2线方式进行通信**（一条时钟线，一条数据线）。   
SPI总线则以**同步串行3线方式进行通信**（一条时钟线，一条数据输入线，一条数据输出线）。   
SCI总线是以**异步方式进行通信**（一条数据输入线，一条数据输出线）。   
1-wire，即单线总线，又叫单总线。例如DS18B20温度传感器就是用的这种总线结构.   
我们这里重点详解下I2C串行总线，我们这里以数据手册的IIC时序图为例讲起，看不懂时序图的小伙伴必须补上来了.

**一.I2C串行总线的组成及工作原理.**

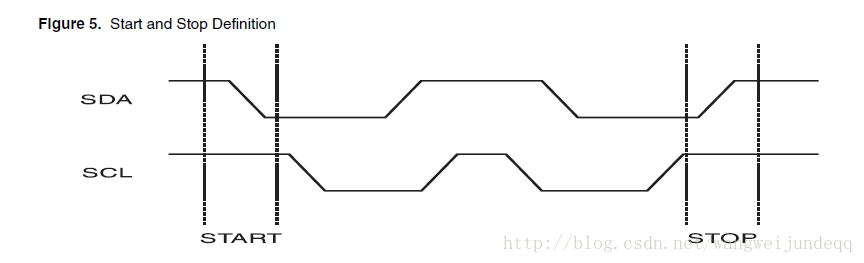
1.I2C总线是PHLIPS公司推出的一种串行总线，它只有两根双向信号线。**一根是数据线SDA（serial data I/O），另一根是时钟线SCL（serial clock）。**   
2.如下图所示，**IIC一条总线上可以挂多个器件，而每个器件都有唯一的地址**，这样可以标识通信目标。数据的通信的方式采用主从方式，主机负责主动联系从机，而从机则被动回应数据并及时响应。   


**二.I2C总线传输协议**

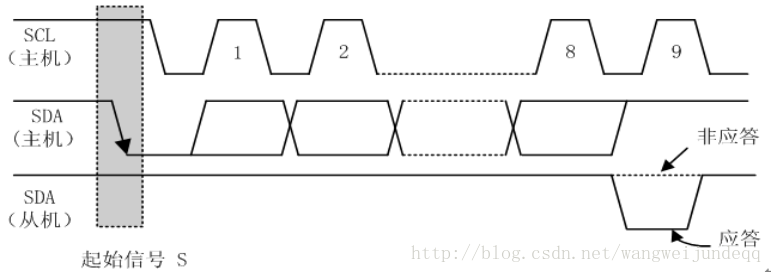
**1.数据位的有效性规定：**

SCL为高电平期间，数据线上的数据必须保持稳定，只有SCL信号为低电平期间，SDA状态才允许变化。如图所示   


**2.I2C的起始和终止信号**

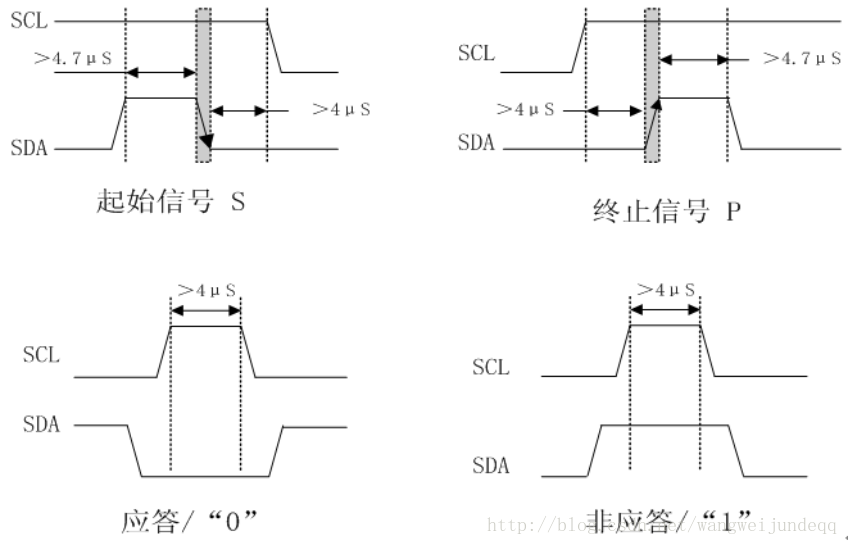
**SCL线为高电平期间，SDA线由高电平向低电平的变化表示起始信号；**   
**SCL线为高电平期间，SDA线由低电平向高电平的变化表示终止信号；**   
响应信号（ ACK ）：接收器在接收到8位数据后，在第9个时钟周期，拉低SDA电平。即接收数据的IC在接收到8bit数据后，向发送数据的IC发出特定的低电平脉冲，表示已收到数据。CPU向受控单元发出一个信号后，等待受控单元发出一个应答信号，CPU接收到应答信号后，根据实际情况作出是否继续传 递信号的判断。若未收到应答信号，由判断为受控单元出现故障；如图所示   
   
**下列三种情况不会有ACK信号：**   
A、当从机不能响应从机地址时（从机忙于其他事无法响应IIC总线操作或这个地址没有对应从机），在第9个SCL周期内SDA线没有被拉低，即没有ACK信号。这时，主机发送一个P信号终止传输或者重新发送一个S信号开始新的传输   
B、从机接收器在传输过程中不能接收更多的数据时，也不会发出ACK信号。主机意识到这点，从而发出一个P信号终止传输或者从新发送一个S信号开始新的传输   
C、主机接收器在接收到最后一个字节时，也不会发出ACK信号，于是，从机发送器释放SDA线，允许主机发送P信号结束传输

**3.I2C字节的传送与应答**

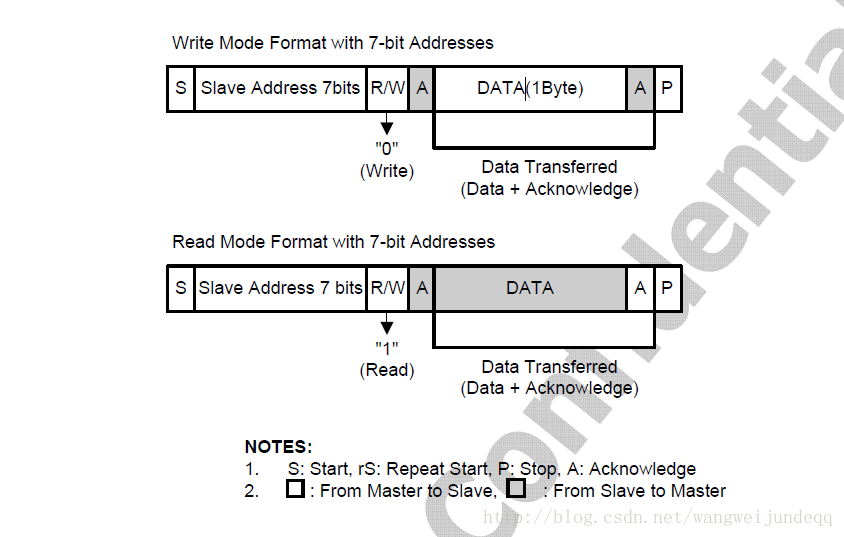
每一个字节必须保证是8位长度。数据传送时，先传送最高位（MSB），每一个被传送的字节后面都必须跟随一位应答位（即一帧共有9位）.如图所示   


**4.应答位的作用**

主机在发送数据时，每次发送一字节数据，都需要读取从机应答位，当从机空闲可以接收该字节数据时，从机会发出应答（一帧数据的第9位为“0”），当从机正忙于其他工作的处理来不及接收主机发送的数据时，从机会发出非应答（一帧数据的第9位为“1”）主机则应发出终止信号以结束数据的继续传送，主机通过从机发出的应答位来判断从机是否成功接收数据.

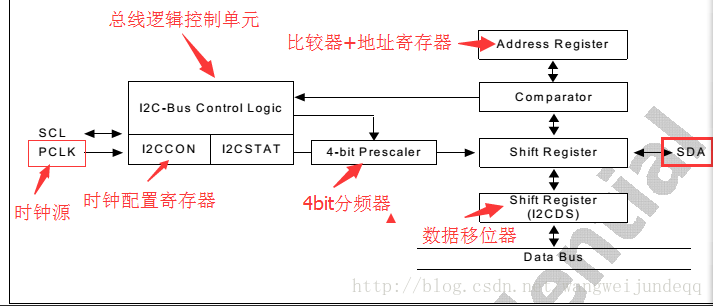
当主机接收数据时，它收到最后一个数据字节后，必须向从机发出一个结束传送的信号。这个信号是由对从机的“非应答”来实现的。然后，从机释放SDA线，以允许主机产生终止信号。   
贴个IIC总线在传送数据过程中信号时序图.好好研究好时序图，一切都可以轻松解决.   


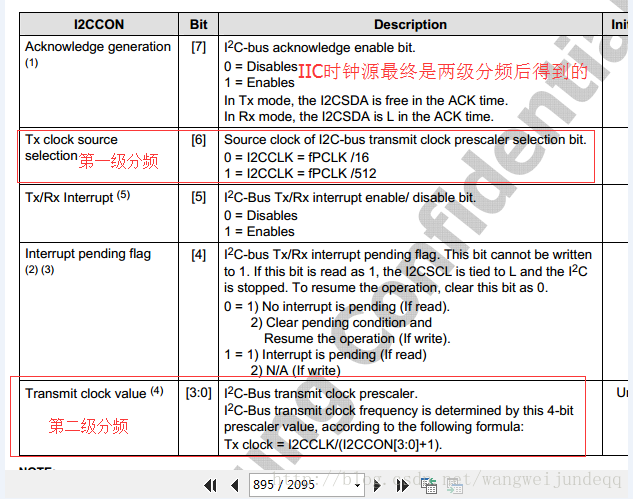
**● I2C是如何通信的？**

s5pv210是发送或接受数据是如何区分多个子设备？可以看下图：开始通信以后，主设备首先会发送7bit位的slave device地址，和1bit位的rean或者write命令，   
(1)如果为write命令，则主设备free SDA通信线(If the I2C-bus is free, both SDA and SCL lines should be both at High level.三星user manual)，即SDA   
为高位。然后从设备先ACK主设备（拉低SDA）表示收到命令(S)。然后主设备在发送8bit数据，从设备在ACK(A)。然然后结束(P)。   
(2)如果为read命令，则从设备先ACK主设备（拉低SDA），然后发送8bit数据，主设备ACK从设备(拉低SDA)，从设备在发送，直到主设备停止接收。   
如下图所以，白色bit位为主设备发送，灰色bit位为从设备发送。   


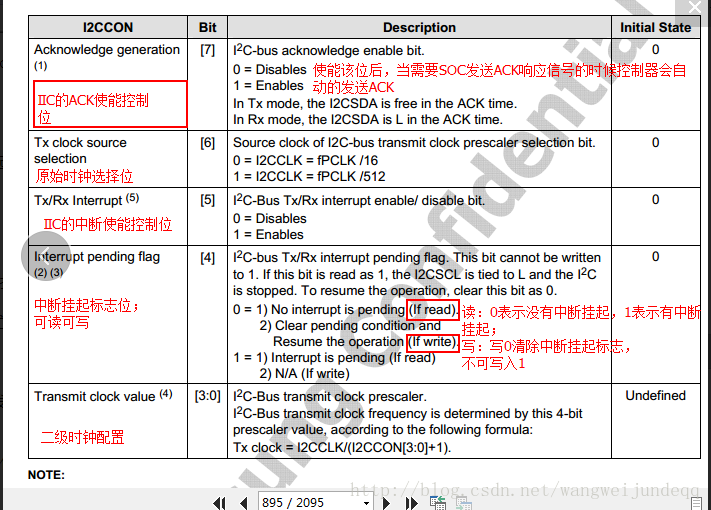
**三.S5PV210的I2C控制器**

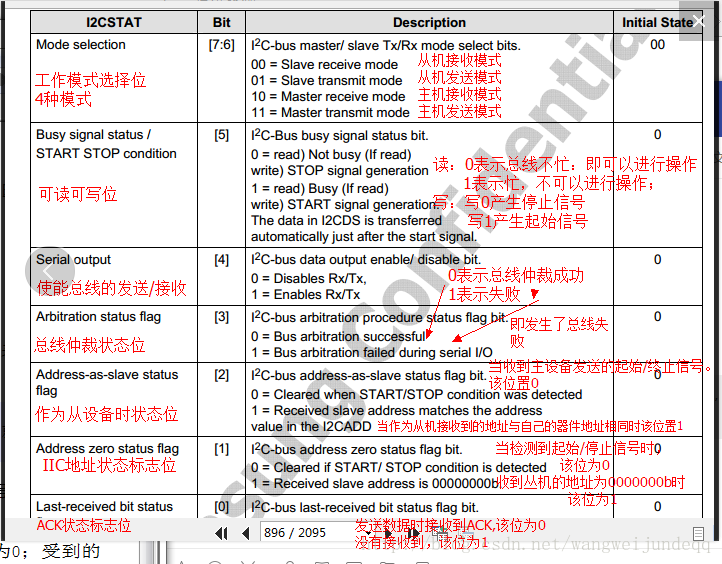
**IIC控制器的作用：**   
通过硬件的控制器来产生通信的时序，软件只需要控制对应IIC控制器的寄存器的配置即可。通信双方的时序会由控制器自行分配。

**1.结构框图分析**   
   
时钟源：PCLK\_PSYS=65MHz，经过内部分频最终得到I2C控制器的CLK，通信中这个CLK会通过SCL线传给从设备。   
总线控制逻辑单元：产生IIC通信时序（设置I2CCON、I2CSTAT）   
移位寄存器：将要发送的字节数据变成1bit移位到SDA线   
I2CCON：时钟配置   
I2CSTAT：操作模式及条件位发送,配合I2CCON一起用，产生通信时序和IIC接口配置   
I2CADD：IIC地址，用来写自己的地址   
I2CDS：数据移位器，发送和接收的地址放在这里   
比较器+地址寄存器：作为从设备使用时将收到的地址与地址寄存器地址比较。

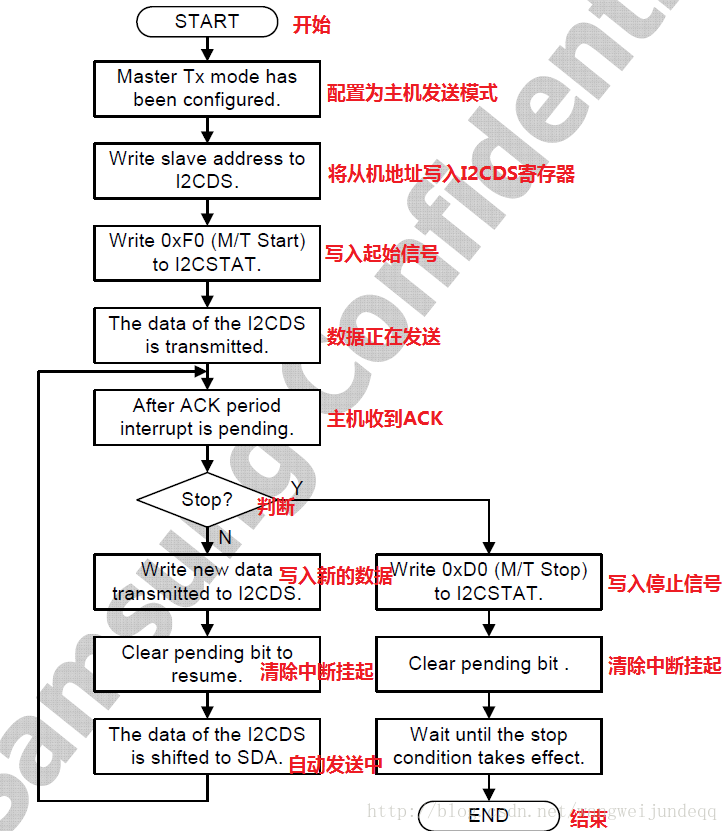
**2.分析I2C的时钟**   
   
(1)I2C时钟源头来源于PCLK（PCLK\_PSYS，等于65MHz），经过了2级分频后得到的。   
(2)第一级分频是I2CCON的bit6，可以得到一个中间时钟I2CCLK（等于PCLK/16或者PCLK/512）   
(3)第二级分频是得到最终I2C控制器工作的时钟，以I2CCLK这个中间时钟为来源，分频系数为[1,16]   
(4)最终要得到时钟是2级分频后的时钟，譬如这样设置：65000KHz/512/4 = 31KHz

**3.主要寄存器I2CCON、I2CSTAT**

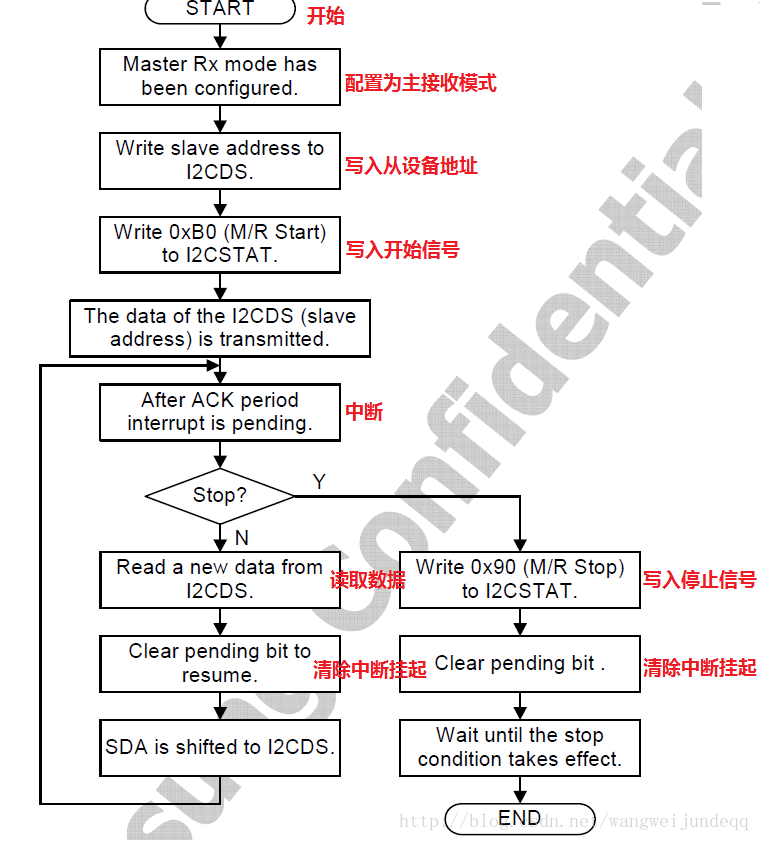
**I2CCON + I2CSTAT：主要用来产生通信时序和I2C接口配置，分析两**个寄存器的bit位   
I2CCON寄存器：   


I2CSTAT寄存器：   


**四.S5PV210的I2C控制器通信流程分析**

**(1)S5PV210的主发送模式流程图**   


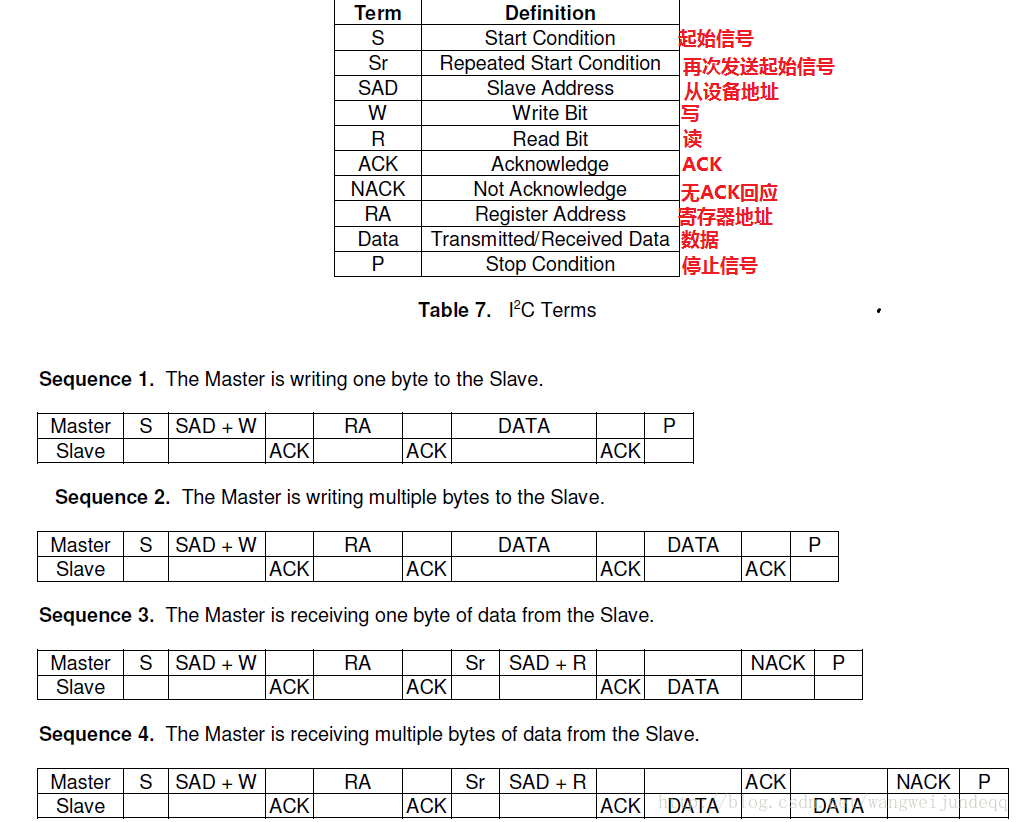
**(2)S5PV210的主接收模式流程图**

   
在SoC中I2C通信代码中，只需要按照上图中的流程图的顺序即可完成通信，I2C控制器内部电路模块已经帮我们实现了通信时序，我们不需要去关心具体时序过程。

**五.S5PV210板载的I2C传感器**

**● I2C从设备的设备地址**

**(1)KXTE9的I2C地址固定为0b0001111（0x0f）**   
(2)I2C从设备地址本身是7位的，但是在I2C通信中发送I2C从设备地址时实际发送的是8位，这8位中高7位（bit7-bit1）对应I2C从设备的7位地址，最低一位（LSB）存放的是R/W信息（就是说下一个数据是主设备写从设备读（对应0），还是主设备读从设备写（对应1））   
(3)基于上面讲的，对于KXTE9来说，   
主设备（SoC）写给gsensor信息时，SAD应该是：**0b00011110(0x1E)**   
如果是主设备读取gsensor信息时，SAD应该是：**0b00011111（0x1F）\*\***

**● IIC传感器的读写寄存器流程图**   


六.SoCI2C控制器通信代码分析   
七.I2C传感器通信代码分析   
目前只能根据代码分析对应寄存器的使用，对内部驱动代码不懂   
研究了驱动后边补