一、I2C通信协议 2

[1、I2C 协议： 2](#_Toc310607238)

[2、I2C时序 3](#_Toc310607239)

[3、I2C协议中文版本 7](#_Toc310607240)

[二、I2C协议中遇到的BUG分析 8](#_Toc310607241)

[1、项目A72A中时序问题 8](#_Toc310607242)

[2、T03 Light sensor 无ACK问题 8](#_Toc310607243)

[3、S26I 电池以及F01电池问题 9](#_Toc310607244)

[4、现象ACK后面的半高的小毛刺分析 10](#_Toc310607245)

[5、T05C G-sensor无数据问题 11](#_Toc310607246)

[三、I2C协议使用注意事项 12](#_Toc310607247)

[1、I2C level shift 普通MOS以及level shift IC 12](#_Toc310607248)

[2、I2C使用注意事项以及bug总结 15](#_Toc310607249)

## 一、I2C通信协议

#### 1、I2C 协议：

I2C 总线支持任何 IC 生产过程（ NMOS、CMOS、双极性 ）。两线的串行数据 SDA 和串行时钟SCL 线在连接到总线的器件间传递信息，每个器件都有一个唯一的地址识别（无论是微控制器、LCD驱动器、存储器或键盘接口）， 而且都可以作为一个发送器或接收器，由器件的功能决定，很明显LCD驱动器只是一个接收器，而存储器则既可以接收又可以发送数据。除了发送器和接收器外器件，在执行数据传输时也可以被看作是主机或从机，如下表 1。主机是初始化总线的数据传输并产生允许传输的时钟信号的器件，此时任何被寻址的器件都被认为是从机。

表格1

|  |  |
| --- | --- |
| 术语 | 描述 |
| 发送器 | 发送数据到总线的器件 |
| 接收器 | 从总线接收数据的器件 |
| 主机 | 初始化发送 产生时钟信号和终止发送的器件 |
| 从机 | 被主机寻址的器件 |
| 多主机 | 同时有多于一个主机尝试控制总线，但不破坏报文 |
| 仲裁 | 是一个在有多个主机同时尝试控制总线 但只允许其中一个控制总线并使报文不被破坏的过程 |
| 同步 | 两个或多个器件同步时钟信号的过程 |

① I2C总线的一些特征:

* 只要求两条总线线路：一条串行数据线 SDA 一条串行时钟线 SCL
* 每个连接到总线的器件都可以通过唯一的地址和一直存在的简单的主机从机关系软件设定地址 主机可以作为主机发送器或主机接收器
* 它是一个真正的多主机总线 如果两个或更多主机同时初始化数据传输可以通过冲突检测和仲裁防止数据被破坏
* 串行的 8位双向数据传输位速率在标准模式下可达 100kbit/s，快速模式下可达 400kbit/s 高速模式下可达 3.4Mbit/s
* 片上的滤波器可以滤去总线数据线上的毛刺波保证数据完整
* 连接到相同总线的 IC 数量只受到总线的最大电容 400pF 限制

②I2C总线的传输：

I2C 总线是一个多主机的总线，这就是说可以连接多于一个能控制总线的器件到总线，由于主机通常是微控制器，让我们考虑以下数据在两个连接到 I2C 总线的微控制器之间传输的情况。

这突出了 I2C 总线的主机-从机和接收器-发送器的关系，应当注意的是，这些关系不是持久的，只由当时数据传输的方向决定传输数据的过程如下 ：

A、假设微控制器 A 要发送信息到微控制器 B

* + 微控制器 A（主机）寻址微控制器 B（从机）
  + 微控制器 A（主机）发送器 发送数据到微控制器 B（从机）接收器
  + 微控制器 A 终止传输

B、 如果微控制器 A 想从微控制器 B 接收信息

* + 微控制器 A（主机）寻址微控制器 B 从机
  + 微控制器 A（主机）接收器从微控制器 B（从机）发送器 接收数据
  + 微控制器 A 终止传输

在 I2C 总线上产生时钟信号通常是主机器件的责任。当在总线上传输数据时，每个主机产生自己的时钟信号，主机发出的总线时钟信号只有在以下的情况才能被改变：慢速的从机器件控制时钟线并延长时钟信号，或者在发生仲裁时被另一个主机改变 。

③I2C数据传输中得信号：

I2C总线在传送数据过程中共有三种类型信号

开始信号：SCL为高电平时，SDA由高电平向低电平跳变，开始传送数据。

结束信号：SCL为高电平时，SDA由低电平向高电平跳变，结束传送数据。

应答信号： 接收数据的从机在接收到8bit数据后，向发送数据的主机发出特定的低电平脉冲，表示已收到数据。主机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，主机接收到应答信号后，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。

④I2C数据的有效性：

只有在SDA数据必须在时钟的高电平周期保持稳定，数据线的高或低电平状态只有在 SCL 线的时钟信号是低电平时才能改变。

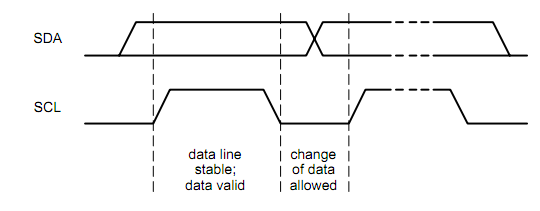


图1

⑤I2C总线的优点：

* 极低的电流消耗
* 抗高噪声干扰
* 电源电压范围宽
* 工作的温度范围广
* 故障诊断和调试都很简单 故障可被立即寻迹
* 在系统中增加或删除 IC 不会影响总线的其他电路

#### 2、I2C时序

I2C的时序主要可以分为：总线空闲状态、启动信号、停止信号、数据传输位、应答信号、插入等待时间、重启动信号、时钟同步、总线冲突和总线仲裁、总线封锁状态。

①总线空闲状态。

I2C总线的SDA和SCL两条信号线同时处于高电平时，规定为总线的空闲状态。此时各个器件的输出级场效应管均处在截止状态，即释放总线，由两条信号线各自的上拉电阻把电平拉高。

②启动信号

在时钟线SCL保持高电平期间，数据线SDA上的电平被拉低（即负跳变），定义为I2C总线的启动信号，它标志着一次数据传输的开始。

启动信号是一种电平跳变时序信号，而不是一个电平信号。启动信号是由主控器主动建立的，在建立该信号之前I2C总线必须处于空闲状态，如图2

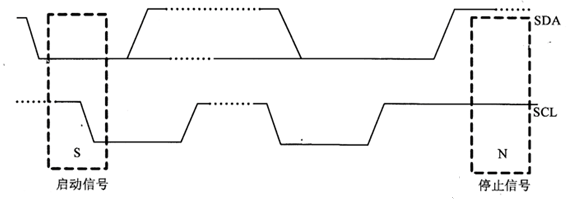


图2

③停止信号

在时钟线SCL保持高电平期间，数据线SDA被释放，使得SDA返回高电平（即正跳变），称为I2C总线的停止信号，它标志着一次数据传输的终止。如图2

停止信号也是一种电平跳变时序信号，而不是一个电平信号，停止信号也是由主控器主动建立的，建立该信号之后，I2C总线将返回空闲状态。

④数据位传送

在I2C总线上传送的每一位数据都有一个时钟脉冲相对应（或同步控制），即在SCL串行时钟的配合下，在SDA上逐位地串行传送每一位数据。

进行数据传送时，在SCL呈现高电平期间，SDA上的电平必须保持稳定，低电平为数据0，高电平为数据1。

只有在SCL为低电平期间，才允许SDA上的电平改变状态。逻辑0的电平为低电压，而逻辑1的电平取决于器件本身的正电源电压VDD（当使用独立电源时），如图3所示。

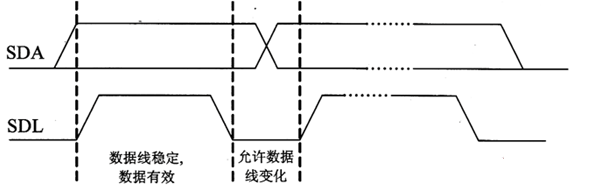


图3

⑤应答信号

I2C总线上的所有数据都是以8位字节传送的，发送器每发送一个字节，就在时钟脉冲9期间释放数据线，由接收器反馈一个应答信号。

应答信号为低电平时，规定为有效应答位（ACK简称应答位），表示接收器已经成功地接收了该字节；应答信号为高电平时，规定为非应答位（NACK），一般表示接收器接收该字节没有成功。

对于反馈有效应答位ACK的要求是，接收器在第9个时钟脉冲之前的低电平期间将SDA线拉低，并且确保在该时钟的高电平期间为稳定的低电平。

如果接收器是主控器，则在它收到最后一个字节后，发送一个NACK信号，以通知被控发送器结束数据发送，并释放SDA线，以便主控接收器发送一个停止信号P，如图4所示。

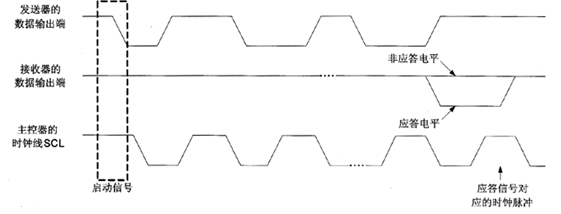


图4

⑥插入等待时间。

如果被控器需要延迟下一个数据字节开始传送的时间，则可以通过把时钟线SCL电平拉低并且保持，使主控器进入等待状态。一旦被控器释放时钟线，数据传输就得以继续下去，这样就使得被控器得到足够时间转移已经收到的数据字节，或者准备好即将发送的数据字节。

带有CPU的被控器在对收到的地址字节做出应答之后，需要一定的时间去执行中断服务子程序，来分析或比较地址码，其间就把SCL线钳位在低电平上，直到处理妥当后才释放SCL线，进而使主控器继续后续数据字节的发送，如图5所示。

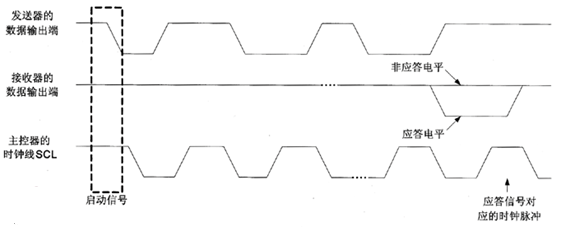


图5

⑦重启动信号。

在主控器控制总线期间完成了一次数据通信（发送或接收）之后，如果想继续占用总线再进行一次数据通信（发送或接收），而又不释放总线，就需要利用重启动Sr信号时序。

重启动信号Sr既作为前一次数据传输的结束，又作为后一次数据传输的开始。利用重启动信号的优点是，在前后两次通信之间主控器不需要释放总线，这样就不会丢失总线的控制权，即不让其他主器件节点抢占总线。

⑧时钟同步。

如果在某一I2C总线系统中存在两个主器件节点，分别记为主器件1和主器件2，其时钟输出端分别为CLK1和CLK0，它们都有控制总线的能力。

假设在某一期间两者相继向SCL线发出了波形不同的时钟脉冲序列CLK1和CLK2（时钟脉冲的高、低电平宽度都是依靠各自内部专用计数器定时产生的），在总线控制权还没有裁定之前这种现象是可能出现的。

鉴于I2C总线的“线与”特性，使得时钟线SCL上得到的时钟信号波形，既不像主器件1所期望的CLK1，也不像主器件2所期望的CLK2，而是两者进行逻辑与的结果。

CLKI和CLK2的合成波形作为共同的同步时钟信号，一旦总线控制权裁定给某一主器件，则总线时钟信号将会只由该主器件产生，如图6所示。

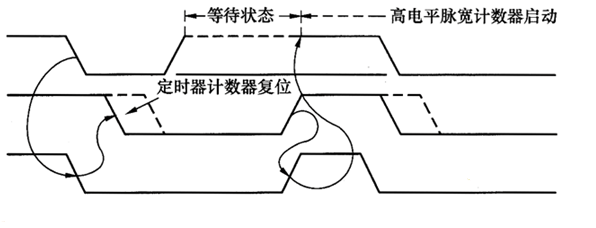


图6

⑨总线冲突和总线仲裁。

假如在某I2C总线系统中存在两个主器件节点，分别记为主器件1和主器件2，其数据输出端分别为DATA1和DATA2，它们都有控制总线的能力，这就存在着发生总线冲突（即写冲突）的可能性。假设在某一瞬间两者相继向总线发出了启动信号，鉴于：I2C总线的“线与”特性，使得在数据线SDA上得到的信号波形是DATA1和DATA2两者相与的结果，该结果略微超前送出低电平的主器件1，其DATA1的下降沿被当做SDA的下降沿。

在总线被启动后，主器件1企图发送数据“101……”，主器件2企图发送数据“100101……”。

两个主器件在每次发出一个数据位的同时都要对自己输出端的信号电平进行抽检，只要抽检的结果与它们自己预期的电平相符，就会继续占用总线，总线控制权也就得不到裁定结果。

主器件1的第3位期望发送“1”，也就是在第3个时钟周期内送出高电平。

在该时钟周期的高电平期间，主器件1进行例行抽检时，结果检测到一个不相匹配的电平“0”，这时主器件1只好决定放弃总线控制杈；因此，主器件2就成了总线的惟一主宰者，总线控制权也就最终得出了裁定结果，从而实现了总线仲裁的功能。

从以上总线仲裁的完成过程可以得出：仲裁过程主器件1和主器件2都不会丢失数据；各个主器件没有优先级别之分，总线控制权是随机裁定的，即使是抢先发送启动信号的主器件1最终也并没有得到控制杈。

系统实际上遵循的是“低电平优先”的仲裁原则，将总线判给在数据线上先发送低电平的主器件，而其他发送高电平的主器件将失去总线控制权，如图7所示。

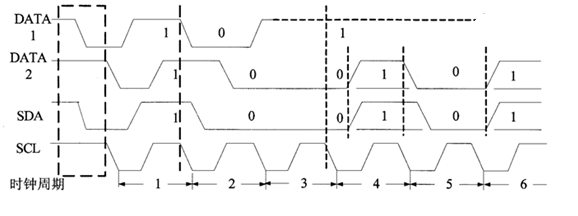


图7

⑩总线封锁状态。

在特殊情况下，如果需要禁止所有发生在I2C总线上的通信活动，封锁或关闭总线是一种可行途径，只要挂接于该总线上的任意一个器件将时钟线SCL锁定在低电平上即可。

#### 3、I2C协议中文版本

在网上找到比较好的版本的I2C协议翻译的中文版本，这里就不花时间去翻译，可以作为参考。

## 二、I2C协议中遇到的BUG分析

在学习掌握协议的内容以后，在这里查找了buglist中I2C的bug进行分析整理。

#### 1、项目A72A中时序问题

SI在测试I2C的Compass IC MMC3280MS的时序发现如下问题：

LOW period of SCL的测试fail，SPEC要求大于等于1.3us，实测值为1.279us，

SDA Rise Time的测试fail，SPEC要求小于等于300ns，实测值为789.6ns，

SCL Rise Time的测试fail，SPEC要求小于等于300ns，实测值为832.2ns，

SI在测试I2C的G-sensor MXC6225XU的时序发现如下问题：

STOP Setup Time的测试fail，SPEC要求大于等于600ns,实测值为132.6ns

SCK Rise Time的测试fail，SPEC要求小于等于300ns，实测值为715.3ns

SDA Rise Time的测试fail，SPEC要求小于等于300ns，实测值为732.5ns

SCL Low Time的测试fail，SPEC要求大于等于1.3us，实测值为1.277us

Data Set-Up Time的测试fail，SPEC要求大于等于100ns，实测值为35.17ns

Start Set-Up Time的测试fail，SPEC要求大于等于600ns,实测值为1.646ns

Stop Set-Up Time的测试fail，SPEC要求大于等于600ns,实测值为299.9ns

由于SCL和SDA的上拉为4.7K，在公板中是没有问题的，但是有由于在A72A项目中接了更多的负载设备，导致I2C的SCL以及SDA上升过缓导致了上述的问题，将之前4.7K的上拉更改为2.2K以后问题解决。

#### 2、T03 Light sensor 无ACK问题

在T03项目调试的时候，发现Light sensor CM3212无应答信号ACK，对此进行了一系列的分析调试过程，首先在调试T03项目的时候发现了Light sensor CM3212无应答信号ACK，根据一般的思路我们首先进行最基本的通信线路进行检查：

A、设备电源----正常

B、设备地址，通讯速率----正常 100kbit/s

C、I2C线路是否断开或者data与clk是否接反---正常

确认以上均是OK，然后我们进一步的进行确认，

D、电源增加滤波处理；

E、查I2C通讯的波形质量，改变设备地址；

F、互换data与clk进行尝试等等

进行以上操作以后，还是没有应答信号ACK，在公板上进行尝试，也找不到ACK，同时得不到台湾原厂的support的情况，我们不得不更换了另外一家的Light sensor的ISL29023，这片Light sensor相对比较CM3212而言，速度为400kbit/s，而且存在中断pin，CM3212不存在中断pin，在更换了Light sensor以后我们进一步进行调试；

由于芯片数目较少（4pcs），主板不够，我们换贴了1pcs，现象一样，还是无法接受到应答信号ACK，ISL29023原厂FAE也不能提供有效解决办法，给我们提供了一片开发板，经测试，开发板能够收到ACK，握手成功，可以正常工作。

把开发板飞线到我们的主板上可以收到ACK，通过对比开发板与我们的主板信号差异，发现开发板的中断脚INT#为接地，再将我们的主板上INT#接地，可以收到应答信号ACK；

但是INT#不应该接地，我们再把他们的开发板INT#与地连接断开，还是可以收到ACK，看来还是我们的板子本身有问题。

重新焊接1pcs ISL29023（INT#不接地），可以收到应答信号ACK，看来是焊接的问题了。虽然T03的Light Sensor已经OK了，暂时能够满足功能上的要求，但是对于之前

剩余没有解决的问题：但是为什么之前焊接的那1pcs需要INT#接地才能收到应答信号ACK呢，CM3212为什么收不到ACK呢？这些问题都有待与两家原厂的FAE进行进一步的沟通才能得到比较好的答案。

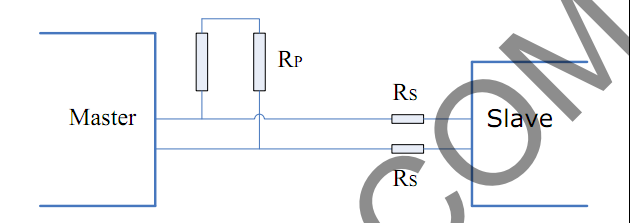
3、S26I 电池以及F01电池问题

查找buglist的SI的测试数据为：大致上的意思是电池那组I2C的上升时间太慢，Brief:Rise time of I2C\_CLK/I2C\_DATA at battery connector fail，Spec Required:Max:1000ns，Test Result:1.308~1.702us

看到后面RD给出来的解决办法是：将串联电阻R444和R445由100ohm改为75ohm，电容C611和C610由100pF改为33pF,上拉电阻由8.2K改为4.7K。更改之后，Rise Time约为600nS.

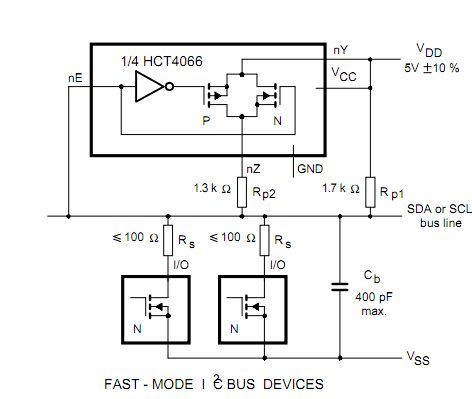
对于修改上拉电阻的事情我们很容易理解是用于提高驱动能力，将I2C上升时间提高，但是对于为什么要修改串联电阻这个事情，不是特别理解为什么要做更改。

问了下其他公司来得员工他们在电池这个地方都是没有串联100欧姆电阻的，但是我们的案子都是有串联的，根据电源那边的说法是：在F01项目的时候电池通信存在问题，去掉以后就100欧姆就OK了，因为在电池内部有串联Rs。



查找I2C协议中关于串联电阻R的说明：串连电阻 RS 是可选的，它们防止 I2C 总线期间的输出级受到总线线路的高电压毛刺波影响。而且减少了串扰和总线信号的负尖峰 ，RS的最大值由在总线线路切换到低电平以关断 Rp2 时这个电阻上允许的最大电压降决定。

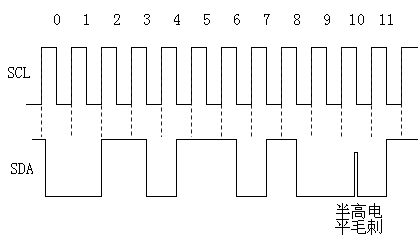
尽管I2C协议中提及了串联电阻的作用，但是还是不太能明白这个100欧姆的电阻是如何起到了减少串扰和总线信号。

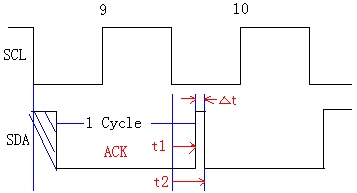


#### 4、现象ACK后面的半高的小毛刺分析

这个是我们经常看到的现象，I2C的ACK后面会有一个半高的小毛刺，无论是在读取电池信息还是其他项目中I2C设备识别到ACK都会有一个比较小半高的毛刺，这个是否会影响data数据的读取呢，我们简单的分析下。

在测试某I2C总线时，发现SDA数据线上有毛刺，而且出现的位置很有规律，一般在第9，18，27-----时钟周期的后面。如下图所示。





如上图所示，t1是I2C总线上的Slave设备应答信号ACK(第9个时钟周期输出)相对与参考时钟SCL下降沿的滞后时间。数据方向是Slave--Master。t 2是I2C总线上的Master设备在第10个时钟输出相对了参考时钟SCL的滞后时间。数据方向是Master--Slave。∆t是Master设备与Slave设备输出数据延迟的时间差。

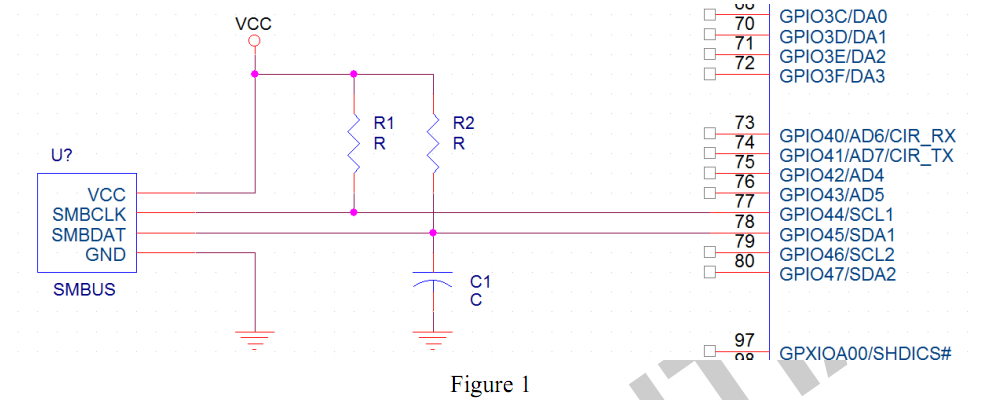
 Master设备发出最后一个bit数据后，总线的使用权交给Slave设备使用，由Slave设备发出应答信号ACK，该信号在SCL下降沿经t1延迟后发出，在SCL的下一个时钟沿后内经t1后结束。而Master设备在同一个时钟沿，经常t2延迟后发出第9bit数据，这样在∆t (t2-t1)时间内，Master和Slave设备都没有使用总线，由于SDA是OC/OD输出，芯片外面通过电阻R上拉到VCC，将SDA电平拉升，但是由于∆t很短，VCC还没来的及将SDA拉到稳定的高电平，Master就开始发出数据“ VCC还没来的及将SDA拉到稳定的高电平，Master就开始发出数据“0”将SDA拉低，因此就在SDA上形成了宽度很窄的半高电平，就是我们观察到的毛刺。此毛刺不会I2C总线的读写时序产生影响，无需处理。

#### 5、T05C G-sensor无数据问题

在T05C中KB3930的EC中使用smbus port 3接入一个sensor，发现无法读取到数据，使用示波器测量信号时，只看到EC发送了设备地址和设备ACK信号，但是没有命令发出，无法读取到数据。在这里我们请教ENE关于这个port3存在问题，使用port1和port2的时候是能够正常通讯的。ENE说问题是在于port3的SDA的hold time不够导致的，建议的是在加入C1,单独调节SDA的保持时间来更好的满足时序的要求。ENE建议最短的SDA的hold time应该是182ns。

或者将SDA的上拉电阻更改为13K或者更大的时候也可以满足要求，T05C项目的做过测试，将电阻更改为30K的时候可以满足要求。

当然这里T05C G-sensor挂在是SMbus上，不完全同于I2C，只是这个bug告诉我们，可以加入电容来调节SCL以及SDA建立时间和保持时间，当然在I2C的协议里面有说明:C1电容最大不能超过400pF，具体的计算方式请参考I2C协议里面的内容。



### 三、I2C协议使用注意事项

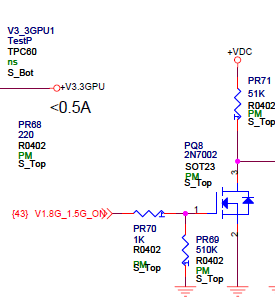
#### 1、I2C level shift 普通MOS以及level shift IC

关于I2C level shift其实有一份官方的spec，说的非常详细，对于应用的各个方面都做很明晰的说明。如果有时间可以仔细看，在这里对这份spec中应用的地方进行一下说明。

Level shift电路我们经常会用到这个电路，在使用I2C协议的时候我们会经常性的使用到level shift电路，有简单的使用L2N7002搭建的Level shift，也有使用level shift IC搭建的，常见的有F01中使用过PCA9306DCU以及sn74avc4t245等等，在F01的公板中也使用到了MAX3373E。

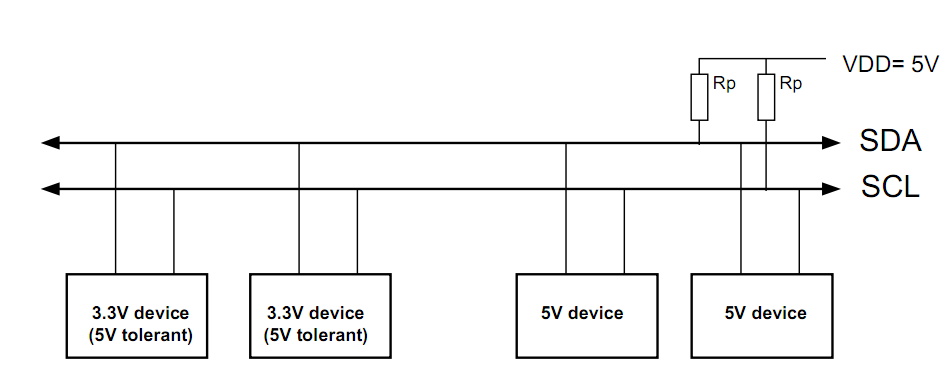
①最简单的I2C level shift电路

PQ8的pin1电平是3.3V的，但通过这个电路，就可以将输出的电平变成+VDC（12V）。其实这个电路也可以看着是实现level shift的功能，这不过这个是单向的，并且是反向的。需要注意的是输入的电平必须要大于Vth（MOS的阈值电压）。

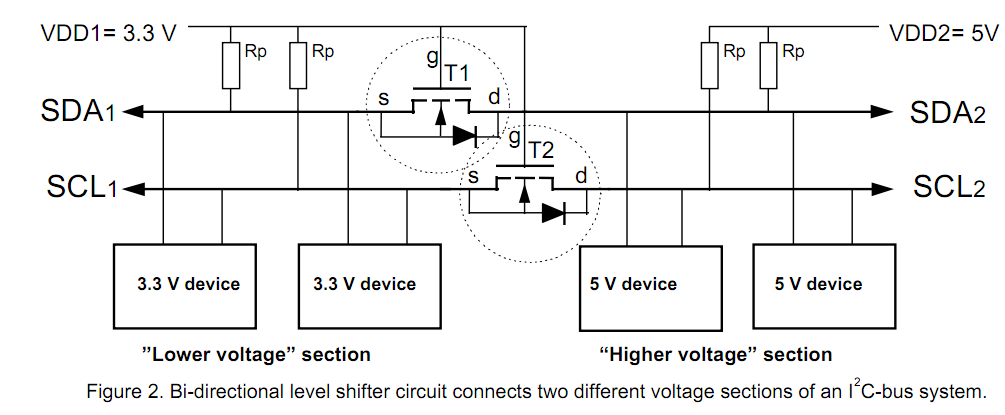


②常见的I2C level shift电路

首先我们提及下有一种可以直接连接的电路，只需要在high voltage的一端上拉到VDD即可，但是前提条件是low voltage的设备能够忍受5V的电压的影响的时候，我们在另外一端无需上拉。



由于I2C上面经常会挂很多device，有些设备3.3V有些是5V的时候我们就会用到I2C设备，大致上的使用方式就如下图。

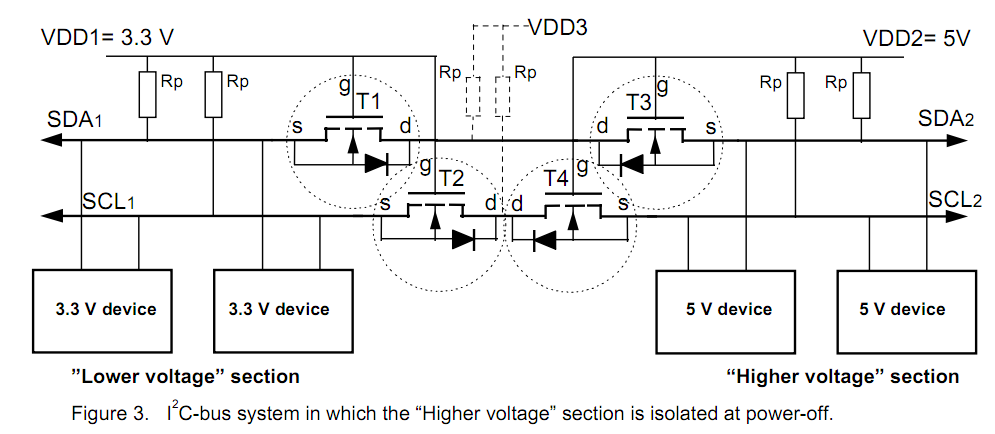


这是一种非常常见的I2C level shift方式，参考刚才提及那份spec中内容，这里简单的说下level shift的三种状态

1. SDA以及SCL在low voltage端以及higher voltage端均没有被拉低，都处于pull up 到VDD的状态，这个时候Vgs的电压小于MOS管的阀值电压，MOS管没有被导通，所以现在的状态两端均为高，但是处于不同的电压值
2. Low voltage端的SDA1拉低的时候，Vgs为3.3V大于MOS管的阀值电压，此时MOS管T1导通，将SDA2也拉低，所以在SDA1为低得时候，两端都处于一种同一个低电压水平。SCL也是一样的道理
3. High voltage端的SDA2拉低的时候，会通过MOS的体二极管（drain-substrate diode）将low voltage的SDA1拉低，这个时候Vgs才会高于MOS管T1的阀值电压，将MOS打开，所以SDA2为低得时候，两端都处于同一种低电压的水平，SCL也是一样的道理。

参考Freescales I.MX51的时候I2C2就是2.775V转化为5V的电压提供给touch panel与设备通讯。同时这组I2C2也连接到了camera以及audio。其余的两组设备均为2.8V

③防止漏电影响的隔离电路



当VDD2的电压为低得时候，我们需要注意一下漏电的现象，在这里加入一部分电路用来隔离漏电的情况，防止SDA为高的时候，通过体电阻漏电到SDA2的情况，

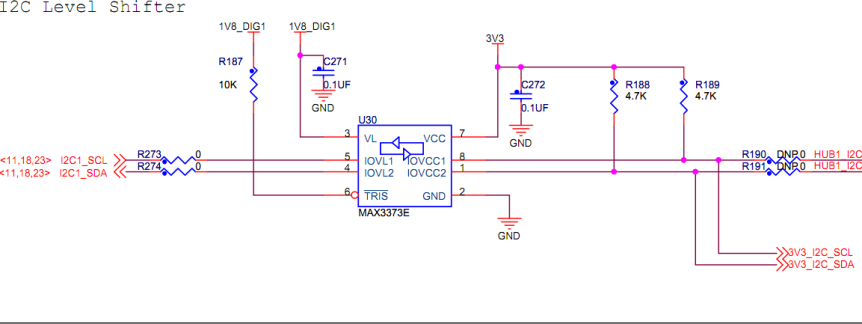
VDD3的上拉电阻需要使用阻值比较大，同时VDD3的电压值不能太低，MOS管的漏极过低，还是会导致两端电压均为高电平的时候导致SDA1以及SDA2的电压的下降。

其实这个问题我们在F01的时候有遇到这个问题，即I2C2的漏电问题，在C版本的漏电问题使用的就是这个办法，但是没有加入VDD3，事实上加入VDD3会更加稳定一些。

④ I2C level shift IC的好处

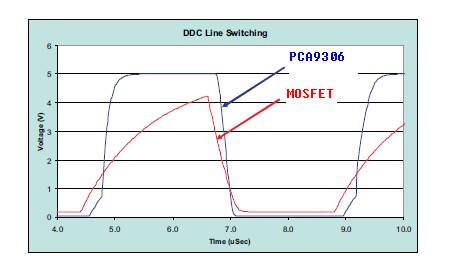
但是在I2C1的使用中，freescales I.MX51的1.8V转化为3.3V的时候用IC完成了转化，并非使用的MOS管搭建。

首先第一点：使用的我们常用的MOS是不可能完成1.8V到3.3V的level shift，因为2N7002的Vgs是2V。在下图中low voltage部分才有1.8V，所以在这里使用了level shift IC。



在F01的项目中我们使用过PCA9306DCU，关于使用level shift IC的好处有两个：

（1）PCA9306DCU可以支持最低1.2V的电压转换，MOS很难做到这么宽的电压转换，而且转换效果也有比较大差别。



（2）使用IC有一个EN的控制pin，用处还是不少的，用于隔离不同频率的设备运行情况。

Spec中原文如下：

The PCA9306 also can be used to run two buses, one at 400-kHz operating frequency and the other at 100-kHz operating frequency. If the two buses are operating at different frequencies, the 100-kHz bus must be isolated ，when the 400-kHz operation of the other bus is required. If the master is running at 400 kHz, the maximum system operating frequency may be less than 400 kHz because of the delays added by the repeater.

起初看了以后也不太明白，与TI的工程师沟通以后理解为：PCA9306的确支持主从设备在不同的频率工作，但是指的不是主从设备同时运行不同频率，如果A side跑400KHz, B side 的100KHz必须通过将PCA9306 disable来隔离, B side的其它400KHz可以照跑. (一条I2C总线可并联N个PCA9306).

当然使用IC还有另外一个好处：因为I2C的协议中有说明，总线可以接最大400pF的负载，其扇出能力是限制的，如果使用IC可以有效的在不适用某些设备的时候使用EN pin进行隔离，可以使用更多的设备。

由于I2C设备属于灌电流设备，所以使用I2C level shift不会增加其扇出能力。

#### 2、I2C使用注意事项以及bug总结

在目前的ARM项目中使用I2C的设备非常多，除了WIFI一般使用SDIO、bluetooth使用UART外，其他的设备sensor大多使用I2C来完成主从设备的通讯，I2C是非常常用的一种通信协议，由于其使用频繁也是最容易出问题的，在F01的时候我们遇到的现象为在休眠过程中如果触摸到触控键会经常性导致系统休眠不下去或者重启的现象也是由于I2C不稳定导致的。

触摸屏概率性无法使用也是由于I2C level shift过程中漏电没有处理好导致触摸屏概率性无法使用导致的。

调试I2C设备的时候会经常性的遇到一些问题在这里我们进行简单的总结：

1. 同一组I2C下尽量避免数据量大的设备连接到一些，比如battery和G-sensor的数据就都非常大，尽量不要将G-sensor和battery两组设备挂在同一组I2C下，如果数据量过大，可能就会导致I2C崩溃，系统死机重启等等异常现象；
2. 同一组I2C下尽量挂在的设备的速率都在同一种速度下，I2C有三种速率模式，有两种是我们比较常用的，在在标准模式下可达100kbit/s，快速模式下可达400kbit/s，高速模式下可达3.4Mbit/s；当然如上面的提到的使用I2C level shift IC PCA9306 让总线在不同的速率下运行。
3. I2C的电源纹波等等相关是否能够满足器件要求，波形质量是否满足规格，驱动能力是否满足要求，是否存在过冲等等异常波形会对通信造成影响，
4. 连线以及中断是否会存在问题，I2C本身不存在中断的问题，但是从设备一般会存在中断的问题，关于中断问题一般来说我们都需要根据器件的要求设置好，必要的时候需要请问原厂的FAE来确认
5. 在无法得到应该信号ACK的时候我们也需要注意I2C设备的地址是否正常，波形上读出来的I2C的地址是否正常，是否有过冲等等其他异常情况导致地址错误等等，注意数据信号的建立时间以及保持时间是否满足要求。
6. 在出现附近周边信号干扰比较大得时候，可以尝试在信号线上串接100欧姆左右的电阻
7. 调节上升时间、建立时间、保持时间等等时序的时候，可以通过调节上拉电阻（一般1K-10K）以及在信号线上下拉负载电容的来解决（一般0-400pF）
8. 对于何时应该使用level shift IC以及防止漏电隔离的时候应该注意：MOS管的开启电压，在high voltage端不使用的时候容易出现漏电的情况。

参考文档：

1. Bidirectionl level shifter for I²C-bus and other systems
2. The I2C specification version2.1
3. I2C总线协议周立功翻译版本
4. PCA9306 datasheets