## 一：串口

串口是串行接口的简称，分为同步传输（USRT）和异步传输（UART）。在同步通信中，发送端和接收端使用同一个时钟。在异步通信中，接受时钟和发送时钟是不同步的，即发送端和接收端都有自己独立的时钟和相同的速度约定。

### 1：RS232接口定义



2：电气特性

作为UART的一种，工作原理是将传输数据的每个字符一位接一位地传输。图一给出了其工作模式：

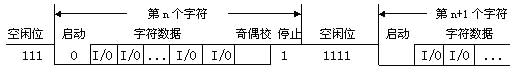
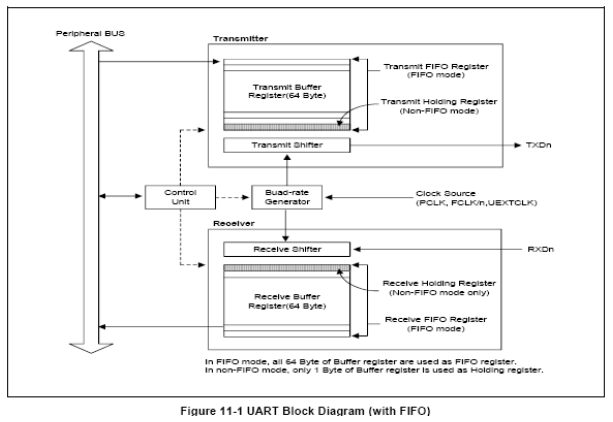


图 一

其中各位的意义如下：  
 **起始位**：先发出一个逻辑”0”的信号，表示传输字符的开始。  
 **数据位**：紧接着起始位之后。数据位的个数可以是4、5、6、7、8等，构成一个字符。通常采用ASCII码。从最低位开始传送，靠时钟定位。  
 **奇偶校验位**：资料位加上这一位后，使得“1”的位数应为偶数(偶校验)或奇数(奇校验)，以此来校验资料传送的正确性。  
 **停止位**：它是一个字符数据的结束标志。可以是1位、1.5位、2位的高电平。  
 **空闲位**：处于逻辑“1”状态，表示当前线路上没有资料传送。  
 **波特率**：是衡量资料传送速率的指针。表示每秒钟传送的二进制位数。例如资料传送速率为120字符/秒，而每一个字符为10位，则其传送的波特率为10×120＝1200字符/秒＝1200波特。

3：在嵌入式处理器中，通常都集成了串口，只需对相关寄存器进行设置，就可以使用啦。尽管不同的体系结构的处理器中，相关的寄存器可能不大一样，但是基于FIFO的uart框图还是差不多。



发送过程：把数据发送到fifo中，fifo把数据发送到移位寄存器，然后在时钟脉冲的作用下，往串口线上发送一位bit数据。

接受过程：接受移位寄存器接收到数据后，将数据放到fifo中，接受fifo事先设置好触发门限，当fifo中数据超过这个门限时，就触发一个中断，然后调用驱动中的中断服务函数，把数据写到flip\_buf中。

## 二：SPI

SPI，是英语Serial Peripheral Interface的缩写，顾名思义就是串行外围设备接口。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，现在越来越多的芯片集成了这种通信协议。  
 SPI是一个环形总线结构，由ss(cs)、sck、sdi、sdo构成，其时序其实很简单，主要是在sck的控制下，两个双向移位寄存器进行数据交换。  
 上升沿发送、下降沿接收、高位先发送。  
 上升沿到来的时候，sdo上的电平将被发送到从设备的寄存器中。 下降沿到来的时候，sdi上的电平将被接收到主设备的寄存器中。

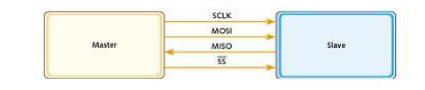
1：技术性能

SPI接口是Motorola 首先提出的全双工三线同步串行外围接口，采用主从模式（Master Slave）架构；支持多slave模式应用，一般仅支持单Master。

时钟由Master控制，在时钟移位脉冲下，数据按位传输，高位在前，低位在后（MSB first）；SPI接口有2根单向数据线，为全双工通信，目前应用中的数据速率可达几Mbps的水平。

2：接口定义

SPI接口共有4根信号线，分别是：设备选择线、时钟线、串行输出数据线、串行输入数据线。



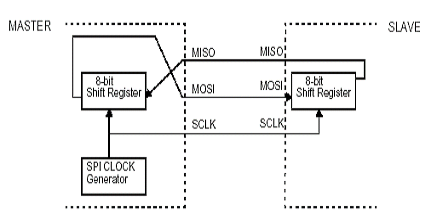
（1）MOSI：主器件数据输出，从器件数据输入

（2）MISO：主器件数据输入，从器件数据输出

（3）SCLK ：时钟信号，由主器件产生

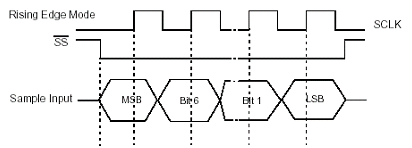
（4）/SS：从器件使能信号，由主器件控制

3：内部结构



4：传输时序

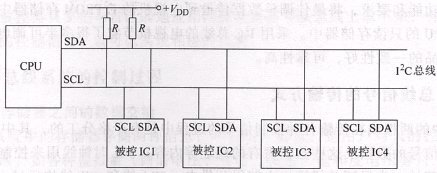
SPI接口在内部硬件实际上是两个简单的移位寄存器,传输的数据为8位，在主器件产生的从器件使能信号和移位脉冲下，按位传输，高位在前，低位在后。如下图所示，在SCLK的下降沿上数据改变，上升沿一位数据被存入移位寄存器。



SPI接口没有指定的流控制，没有应答机制确认是否接收到数据

## 三：I2C

I2C总线是由数据线SDA和时钟SCL构成的串行总线，可发送和接收数据。在CPU与被控IC之间、IC与IC之间进行双向传送，最高传送速率100kbps。各种被控制电路均并联在这条总线上，就像电话机一样只有拨通各自的号码才能工作，所以每个电路和模块都有唯一的地址，在信息的传输过程中，I2C总线上并接的每一模块电路既是主控器（或被控器），又是发送器（或接收器），这取决于它所要完成的功能。CPU发出的控制信号分为地址码和控制量两部分，地址码用来选址，即接通需要控制的电路，确定控制的种类；控制量决定该调整的类别（如对比度、亮度等）及需要调整的量。这样，各控制电路虽然挂在同一条总线上，却彼此独立，互不相关。

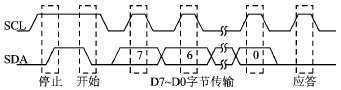


1：I2C信号类型

I2C总线有两根信号线，一根为SDA（数据线），一根为SCL（时钟线）。任何时候时钟信号都是由主控器件产生。

2：I2C总线协议工作原理

以启动信号START来掌管总线，以停止信号STOP来释放总线  
 每次通讯以START开始，以STOP结束；  
 启动信号START后紧接着发送一个地址字节，其中7位为被控器件的地址码，一位为读/写控制位R/W,R. /W位为0表示由主控向被控器件写数据，R/W为1表示由主控向被控器件读数据；  
 当被控器件检测到收到的地址与自己的地址相同时，在第9个时钟期间反馈应答信号；  
 每个数据字节在传送时都是高位(MSB)在前；



3：I2C协议读写过程

读过程

①主控在检测到总线空闲的状况下，首先发送一个START信号掌管总线；  
②发送一个地址字节(包括7位地址码和一位R/W)；  
③当被控器件检测到主控发送的地址与自己的地址相同时发送一个应答信号(ACK)；  
④主控收到ACK后释放数据总线，开始接收第一个数据字节；  
⑤主控收到数据后发送ACK表示继续传送数据，发送NACK表示传送数据结束；  
⑥主控发送完全部数据后，发送一个停止位STOP，结束整个通讯并且释放总线；

写通讯过程:

①主控在检测到总线空闲的状况下，首先发送一个START信号掌管总线；  
 ②发送一个地址字节(包括7位地址码和一位R/W)；  
 ③当被控器件检测到主控发送的地址与自己的地址相同时发送一个应答信号(ACK)；  
 ④主控收到ACK后开始发送第一个数据字节；  
 ⑤被控器收到数据字节后发送一个ACK表示继续传送数据，发送NACK表示传送数据结束；

⑥主控发送完全部数据后，发送一个停止位STOP，结束整个通讯并且释放总线；

4：I2C协议总线信号时序分析

①总线空闲状态

SDA和SCL两条信号线都处于高电平，即总线上所有的器件都释放总线，两条信号线各自的上拉电阻把电平拉高；

②启动信号START

时钟信号SCL保持高电平，数据信号SDA的电平被拉低(即负跳变)。启动信号必须是跳变信号，而且在建立该信号前必修保证总线处于空闲状态；

③停止信号STOP

时钟信号SCL保持高电平，数据线被释放，使得SDA返回高电平(即正跳变)，停止信号也必须是跳变信号。

④数据传送

SCL线呈现高电平期间，SDA线上的电平必须保持稳定，低电平表示0(此时的线电压为地电压)，高电平表示1(此时的电压由元器件的VDD决定)。只有在SCL线为低电平期间，SDA上的电平允许变化。

⑤应答信号ACK

I2C总线的数据都是以字节(8位)的方式传送的，发送器件每发送一个字节之后，在时钟的第9个脉冲期间释放数据总线，由接收器发送一个ACK(把数据总线的电平拉低)来表示数据成功接收。

⑥无应答信号NACK

在时钟的第9个脉冲期间发送器释放数据总线，接收器不拉低数据总线表示一个NACK，NACK有两种用途:

a. 一般表示接收器未成功接收数据字节；

b. 当接收器是主控器时，它收到最后一个字节后，应发送一个NACK信号，以通知被控发送器结束数据发送，并释放总线，以便主控接收器发送一个停止信号STOP。

## 三 USB

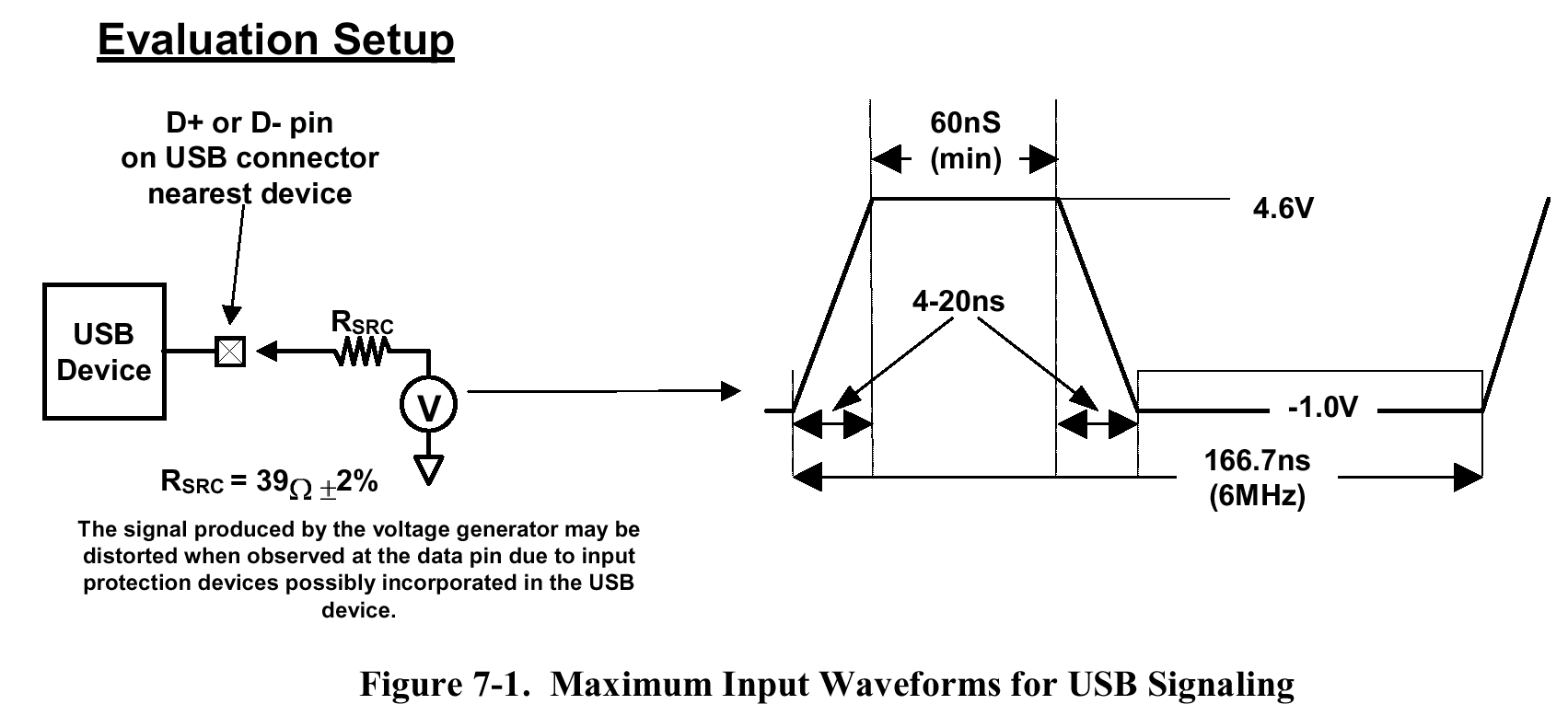
一、信号的发送

USB通常使用一种差分的输出驱动器来控制数据信号在USB电缆上的发送，在了解具体的信号发送之前，我们先来谈谈有关USB设备的特性。

(一)USB驱动器的特性及其使用

一个USB设备端的连接器是由D+、D-及Vbus，GND和其它数据线构成的简短连续电路，并要求连接器上有电缆屏蔽，以免设备在使用过程中被损坏。它有两种工作7状态，即低态和高态。在低态时，驱动器的静态输出端的工作电压Vol变动范围为0～０．3V，且接有一个15kΩ的接地负载。处于差分的高态和低态之间的输出电压变动应尽量保持平衡，以能很好地减小信号的扭曲变形。

在任何驱动状态下，USB设备必须能接收如图1所示的波形。这些波形从一个输出阻抗为3PΩ的电压源直接进入每一个USB数据口。



设置估价

靠近设备的USB连接器上的D+或D\_插口

USB设备

由于USB设备上的输入保护设备可能互相排斥，因此当观察数据的输入端口时，可能发现由电压生成器产生的信号波形可能会变形。

图1 USB信号发送的最大输出波形

下面我们将分别对USB高速驱动器和低速驱动器的特性作一个介绍.

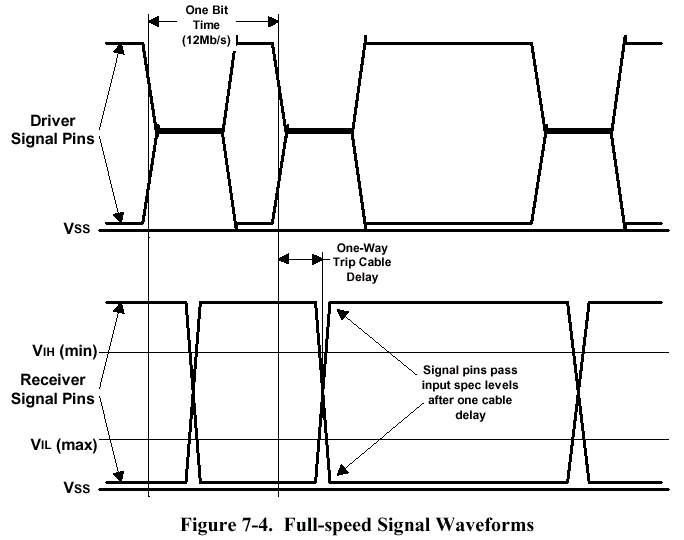
高速驱动器特性

一个高速USB设备的连接是通过阻抗为90Ω±15%，最大单路时延为26ns的屏蔽双绞线电缆进行的，其到达的最大速率为12Mb/s，并且每个驱动器的阻抗必须在28Ω～44Ω之间。图2描述了高速驱动器的信号波形。

低速驱动器特性

一个低速USB设备在插口端必须要有一个带有串行A口连接器的可控制电缆，其速率为1.5Mb/s。当电缆与设备相连时，在D+/D-线上必须要有一个200～450PF的单终端电容器。低速电缆的传播时延必须小于18ns，从而保证信号响在其上升沿或下降沿的第一个中点处产生，以允许电缆与一块电容器相连。图3列出了低速驱器的信号波形。

图4和图5分别列出了高速和低速USB设备在集线器的终端位置及其所连的功能设备。从图我们可以看出在电缆的下形端的电阻Rpu在两图中的连接位置是不同的：



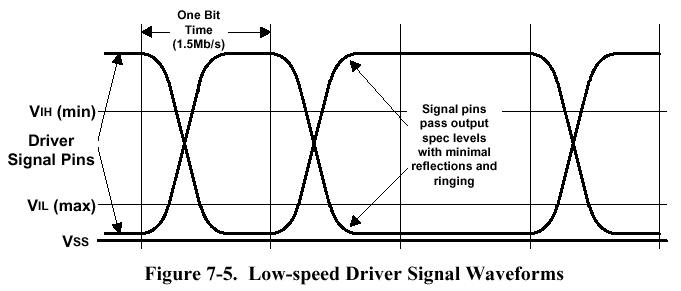
单路长电缆时延

单电缆时延后经过信号端口的标准输入电平

图2 高速信号波形

接收器信号端口

驱动器信号端口



**图3 低速驱动器信号波形**

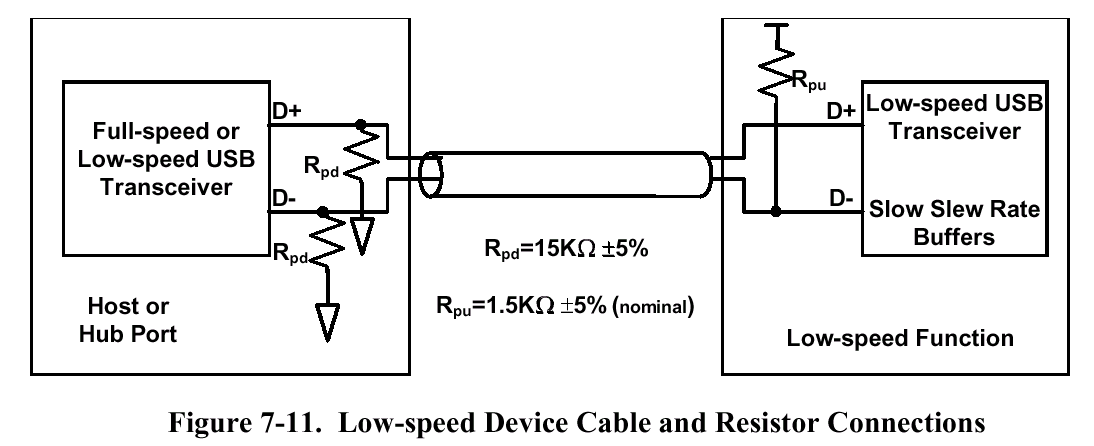
驱动器信号端口

经过信号端 的标准输出电平，并具有最小的映象和阻尼

·图4高速设备中的Rpu电阻是接在D+线上的。

·图5低速设备中的Rpu电阻是接在D-线上的。

·下形端口处的Rpu电阻是与地相连的，其电阻为15KΩ±5%。



主机/HUB口

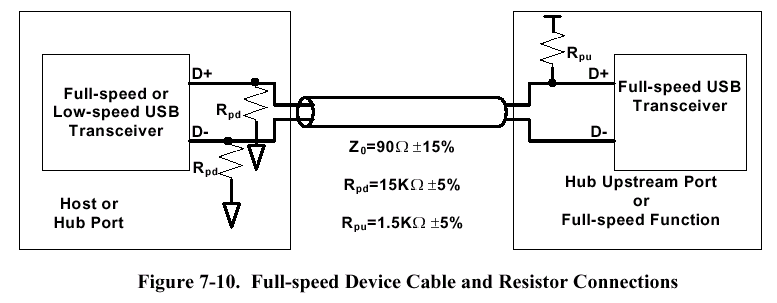
图5 低速设备电缆和电阻连接

高低USB速发送器

低速设备

低速USB设备

慢速旋转缓冲器



高低速USB发送器

HUB上行端或高速设备

主机/HUB口

高速USB发送器

图4 高速设备电缆和电阻连接

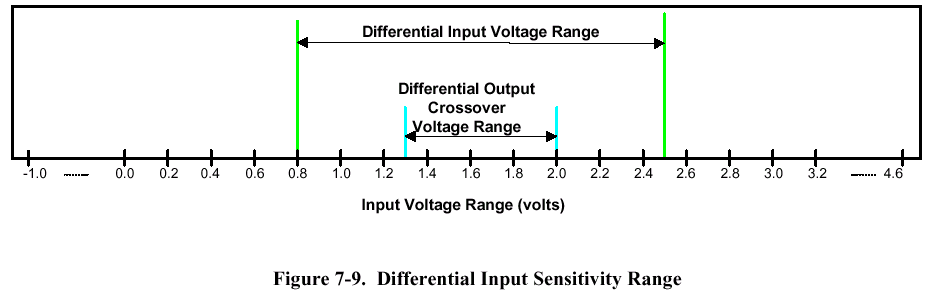
这个Rpu电阻的选取要满足一定的条件，为了在一个复位操作结束后方便地确定可被执行的总线状态，那么选取Rpu时要能使D+/D-线上的电压在2.5us的最大复位松驰时间内可在0～Vih内自由变动。为了满足这一条件，带有可分电缆的设备必须使用加载电压在3.0～3.6U间阻抗为1.5KΩ±5%的电阻；而具有可控电缆的设备可以使用两种方法中的任一种。注意：终端电阻不包括主机/HUB上的15KΩ±5%的电阻。

所有集线器和高速的功能设备上形端口(朝主机方向的)必须使用高速的驱动器，上形集线器端口既可以高速又可以低速来传送数据，但是在信号发送时总是使用高速和边缘速率。低速数据的传输不改变驱动器的特性，低速设备的上形端口必须使用低速驱动器。

所有集线器(包括主机的)外部下形端口必须能适用于两种特性的驱动器，也就是说，任何类型的设备都能被插入这些端口中。当收发器工作在高速模式时，它使用高速和边缘速率来进行信号的发送；工作在低速时，它使用低速和边缘速率来发送数据。

(二)接收器特性

一个差分输入接收器用来接收USB数据信号，当两个差分数据输入处在共同的0.8～2.5V的差分模式范围时，如图6所示，接收器必须具有至少200mv的输入灵敏度。



**图6 差分输入感抗范围**

**差分输入电压范围**

**差分输出桥电压范围**

输入电压范围(Volts)

除了差分接收器外，还必须有为两个数据线中任一个所用的单终端接收器，此时该接收器的合并磁滞现象可以减小它们对噪声的灵敏度。

在差分信号传送期间，D+和D-线上的电压可以小于Vih。对于高速传送而言，这个阶段可以延续到14ns；对于低速传送，可延续到I/V ns之久。接收器的逻辑设备用于保证这种情况不会被当作SE0态来处理。

(三)输入特性

没有终端的D+或D-的输入阻抗必须大于300KΩ，一个端口的输入电容量在连结器的端口处量得。上形和下形端口可以有不同值的电容，一个集线器或主机的下形端口所允许的D+或D-上的最大电容量(差分的或单终端的)为150pF；带有可分电缆的高速设备的上形端口所允许的D+或D-上的最大电容量为100pF。

对于有可控电缆的高速设备，它本身在D+和D-可以有最大电容量为75pF的接地电容器，其中电缆为其余的输入电容使用。在该种设备中，单终端输入电容必须与终端所使用的一致，该终端必须能在2.5us内控制D+或D-线上的电压在0～Vih范围内变动。D+/D-上的电容包括设备单终端输入电容和主机/HUB的150pF的输入电容。

上面我们介绍了USB驱动器和接收器以及输入特性，下面我们将介绍有关USB的信号发送情况。首先介绍信号的发送标准。

(一)信号的发送标准

表7-1总的概括了USB信号的发送标准。(表7-1)在该表中，J和K这两个数据态是两个逻辑电平，在系统中，通常被用来进行交换差分数据。差分数据信号的发送并不关心信号经过处的电平情况，它只要求桥电压在1.3～2.0之间。另外，在接收端，空闲态和工作态在逻辑上分别与J态和K态等价。

一般而言，数据，空闲及唤醒信号的发送标准均由端口的设备类型所决定。如果连结的是高速设备，则USB使用所规定的高速率来发送信号1并且有很快的上升沿和下降沿时间1，甚至还可用低速率来发送数据，而对于表7-1中所示的低速信号发送标准仅用在低速设备与其所连接的端口之间(上升沿和下降沿时间较长)。

(二)连结与中断信号的发送

USB设备是一个智能型的设备，当它发现主机或集线器的下形端口上没有设备连接时，存在的Rpu电阻将使D+和D-上的电压低于主机或集线器端口的单终端电压，此时该端口不是由集线器控制的，这将在下形端口产生一个SE0态。如果主机或集线器不在控制数据线并且下形端口的SEO态的持续时间超过2.5ns，则此时USB设备将中断信号的发送。

如果集线器发现其中一根数据线上的电压大于它的临界值的持续时间超过2.5us，则便开始信号的发送。

在介绍了有关信号发送的相关情况后，我们将分别对各种信号的发送进行讨论。

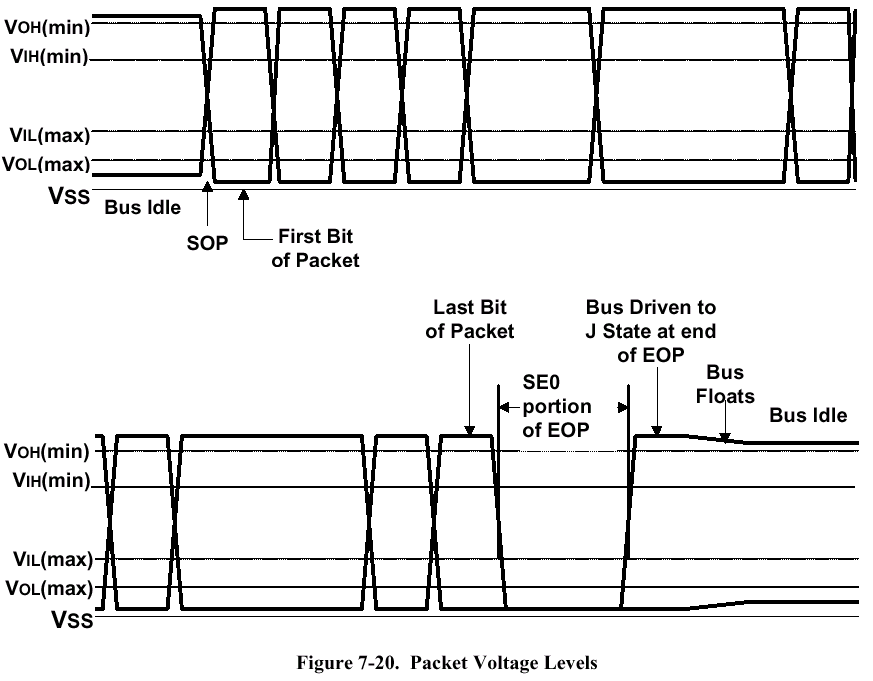
(一)数据信号的发送

通过差分信号来实现数据包的传送。

通过控制D+和D-线从空闲态到相反的逻辑电平(K态)，就可以实现源端口的包发送(SOP)。同步字中的第一位代表了这种在电平上的转换。当它的重新发送时间低于±5ns时，集线器必须对SOP中第一位的宽度变化有所限制。可以通过使用具有延迟输出使能的集线器来实现数据的匹配，这样可以使数据失真减小到最小。

SE0态通常用来表示包的发送结束(EOP)，可以通过控制D+和D-两位时到达SEO态，然后控制D+和D-线一位时后到达J态，就可实现EOP信号的发送。从SEO态到J态的变化表示接收端包发送的结束。J态持续一个位时，然后D+和D-上的输出驱动器均处于高阻抗状态，总线尾端的电阻此时控制总线处于空闲态。图7列出了包开始和结束的信号发送波形。

(二)复位信号的发送



总线空闲

包的第一位

包的最后一位

EOP端总线驱动到J态

总线流动

总线空闲

EOP的SE0部分

**图7 包电压电平**

复位就是将一个信号从挂起态唤醒。

集线器信号通过控制端口上的持久的SE0态来实现对下形端口的复位。复位信号清除后，设备都将处于缺省状态。

表7-1信号电平

|  |  |  |  |
| --- | --- | --- | --- |
| 总线状态 | 信 号 电 平 | | |
| 开始端的源连接器  （一位时端） | 终端的目标连接器 | |
| 需要条件 | 接受条件 |
| 差分的“1” | D+>Voh(min)  D-<Vol(max) | (D+)-(D-)>200mv  (D+)>Vih(min) | (D+)-(D-)>200mv |
| 差分的“0” | D->Voh(min)  D+<Vol(max) | (D+)-(D+)>200mv  D->Vih(min) | (D-)-(D+)>200mv |
| 单终端“0”（SE0） | D+和D-<Vol(max) | D+和D-<Vil(max) | D+和D-<Vih(min) |
| 数据J态：  高速  低速 | 差分的“0”  差分的“1” | 差分的“0”  差分的“1” | |
| 数据K态：  高速  低速 | 差分的“1”  差分的“0” | 差分的“1”  差分的“0 | |
| 空闲状态：  高速  低速 | N.A. | D->Vihz(min)  D+>Vil(max)  D+>Vihz(min)  D-<Vil(max) | D->Vihz(min)  D+<Vih(min)  D+>Vihz(min)  D-<Vih(min) |
| 唤醒状态 | 数据K状态 | 数据K状态 | |
| 包开始（SOP） | 数据线从空闲态转到K态 | | |
| 包结束（EOP）[[1]](#footnote-0)4 | SE0近似地为2位时[[2]](#footnote-1)1  其后仅接着1位时的J[[3]](#footnote-2)3 | SE0≥1位时[[4]](#footnote-3)2其后仅接着一位时的J态 | SE0≥1位时[[5]](#footnote-4)2其后仅接着J态 |
| 断开连接（在下行端口处） | N.A. | SE0持续时间大于等于2.5微秒 | |
| 连接（在上行端口处） | N.A. | 空闲态持续时间大于等于2毫秒 | 空闲态持续时间大于等于2.5微秒 |
| 复位 | D+和D-小于Vol(max)的持续时间大于等于10毫秒 | D+和D-小于Vil(max)的持续时间大于等于10毫秒 | D+和D-小于Vil(max)的持续时间大于等于2.5微秒 |

注释1：以位时定义的EOP宽度与传送的速度有关。(标准的EOP宽度都在表7-5和表7-6中列出)

注释2：以位时定义的EOP宽度与接收EOP的设备类型有关.位时是近似的.

注释3:仅跟在EOP后的J态的宽度以位时来衡量,它与缓冲器的边缘速率有关.来自低速缓冲器的J态 必须要有低速的位时宽,来自高速的,则必须要有高速的位时宽.

注释4:始终处于活动态的是低速的EOP

根据USB系统软件的需求，复位信号可在任一个集线器或主机的控制端口产生，该复位信号的最小持续时间为10ms。复位后，集线器端口将处于能动状态。USB系统软件和主机控制器必须确保发送到根端口的复位信号持续时间足够长以便通知当前正试图进行唤醒操作的各下形设备。根端口产生的复位信号的持续时间应为50ms，但并不要求它一直是延续的。然而，如果复位信号不是连续的，则各间断的复位信号间的时间间隔应小于3ms。

一个设备如果见其上形端口的SE0态持续时间超过2.5us，则它就把该信号作为复位信号处理。在复位信号发送结束前，它必须已产生作用。

当端口处于使能状态后，集线器将传播一个活动信号到新的复位端口。连在该端口的设备必须能识别总线的活动性，并要能防止被挂起。

在复位信号清除后的10ms的复位恢复时间后，集线器必须能接收所有集线器请求，设备也必须能接收一个SetAddress()请求。如果接收这些请求失败，则设备将不能被USB系统软件所识别。

(三)挂起

所有的设备都必须能支持挂起状态，并可从任一电平状态进入挂起态。当设备发现它们的上形总线上的空闲态持续时间超3.0ms时，它们便进入挂起态。当设备的所有端口上的总线不活动时间不超过10ms后，设备必须被真正的挂起，此时它仅从总线上获得挂起电流。如果任一其它总线交通缺乏时，SOF令牌将在每帧中出现一次，以防止高速设备被挂起。当任一低速设备交通缺乏时，在SOF令牌出现的每一帧中至少有一个低速设备处于活动态，以避免它们不被挂起。

当处在挂起状态时，设备必须继续为它的D+(高速)或D-(低速)上的Rpu电阻提供电压从而维持一个空闲态，这样上形集线器才能为设备维持正确的连结状态。

挂起又可分为全局挂起和局部挂起。

·全局挂起

当在总线的任何地方没有通信需要时，就要用到全局挂起，此时所有总线都处在挂起状态。主机通过中止它所有的传送(包括SOF令牌)来发送开始全局挂起信号。当总上的每个设备识别总线的空闲态持续适当时间时，它将进入挂起状态。

·局部挂起

可以通过向集线器端口发送SetPortFeature(PORT-SUSPEND)来使与其相连的总线部分被挂起，此时处于那部分的设备经过上面所说的适当时延后进入挂起状态。

(四)唤醒

处在挂起状态的设备，当它的上形端口接收到任一非空闲信号时，它的操作将被唤醒。特别地，如果设备的远程唤醒功能被USB系统软件开启时，它将自动发信号给系统来唤醒操作。唤醒信号由主机或设备使用，以使一个挂起的总线段回到活动态。集线器在唤醒信号的生成和传播中起了十分重要的作用。设备唤醒时总有一个先后次序，我们将在后面详细介绍。

USB系统软件必须提供10ms的唤醒恢复时间，在这段时间内，它将不对与被唤醒的部分总线相连的任一设备进行操作。

端口的中断与连接也可以使集线器发送一个复位信号，从而唤醒系统，但仅当集线器具有远程唤醒使能时，这些事件才能引起集线器发送唤醒信号。

上面介绍了有关几种信号的发送情况，下面将对数据信号的发送做一个详细的讨论。

(一)数据的编码与解码

在包传送时，USB使用一种NRII(None Return Zero Invert，即无回零反向码)编码方案。在该编码方案中，“1”表示电平不变，“0”表示电平改变。图8列出了一个数据流及其

它的NRII编码，在该图的第二个波形图中，一开始的高电平表示数据线上的J态，后面就是NRII编码。

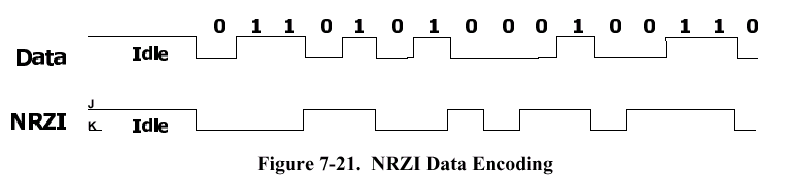


图8 NRZI数据编码

为了确集信号发送的准确性，当在USB上发送一个包时，传送设备就要进行位插入操作。所谓位插入操作是指在数据被编码前，在数据流中每六个连续的‘1’后插入一个‘0’，从而强迫NRII码发生变化，如图9所示。

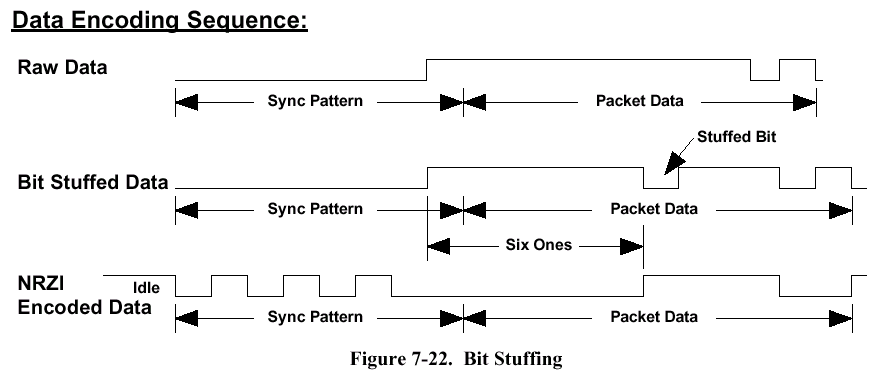


图9 位插入

**数据编码序列**

原始数据

位插入数据

同步块

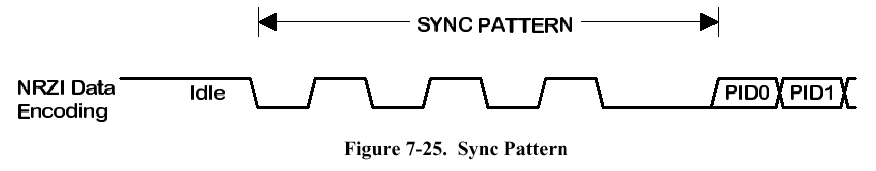
数据包

NRZI编码后的数据

位插入

位插入操作从同步格式(如图10所示)开始，贯穿于整个传送过程，在同步格式端的数据‘1’作为真正数据流的第一位。位插入操作是由传送端强制执行的，是没有例外的。如果严格遵守位插入规则，甚至在EOP信号结束前也要插入一位‘0’位。

接收端必须能对NRZI数据进行解码，识别插入位并去掉它们。如果接收端发现包中任一处有七个连续的“1”，则将会产生一个位插入错误，该数据包将被忽略。



同步格式

NRZI数据编码

图10 同步格式

空闲

关于位的插入有一个特例，那就是刚好在EOP前的时间间隔，EOP前的最后一个数据位可能被集线器的转换偏移而拉长，这种情况如图11所示。

## 四GPIO



传送的数据

接收的特别位，没有错

从传送器来的数据

接收端数据

**图11 对EOP前的特别位的说明**

GPIO的英文全称是General Purpose Input Output （通用输入/输出），具体一定的拉电流和灌电流能力。

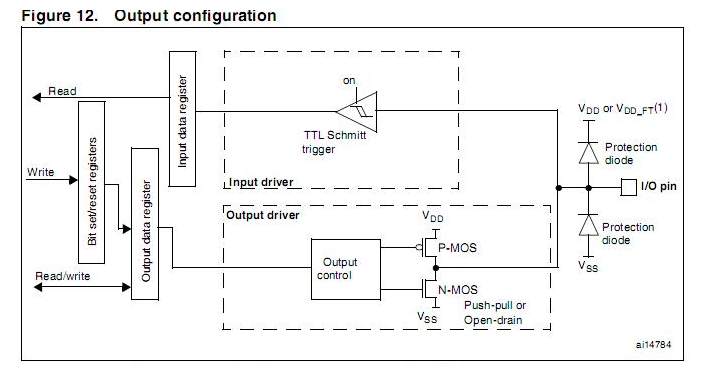


图1：GPIO内部结构图

通常一个GPIO PIN脚对应两个寄存器，输入寄存器和输出寄存器。其 PIN脚的输入、输出模式需要在对应的寄存器进行配置。同时GPIO PIN脚处有内部钳位保护二极管，其作用是防止从外部管脚Pin输入的电压过高或者过低。

GPIO的几种模式 ：

输入模式

（1）GPIO\_Mode\_AIN 模拟输入

（2）GPIO\_Mode\_IN\_FLOATING 浮空输入

（3）GPIO\_Mode\_IPD 下拉输入

（4）GPIO\_Mode\_IPU 上拉输入

输出模式

（5）GPIO\_Mode\_Out\_OD 开漏输出

（6）GPIO\_Mode\_Out\_PP 推挽输出

推挽输出与开漏输出的区别：

推挽输出:可以输出高、低电平; 推挽结构一般由两个三极管分别受两互补信号的控制,总是在一个三极管导通的时候另一个截止.

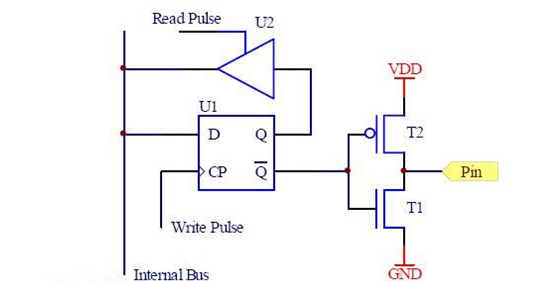


图2：GPIO推挽输出模式结构示意图

开漏输出:输出端相当于三极管的集电极. 要得到高电平状态需要上拉电阻才行. 适合于做电流型的驱动,其吸收电流的能力相对强(一般20ma以内).

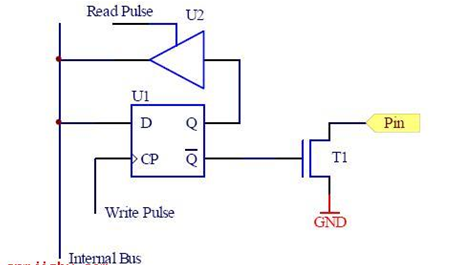


图3：开漏输出结构示意图

开漏输出模式类似我们常说的OC、OD门，其在使用时必须外接上拉电阻，同时开漏输出模式下可以实现多PIN脚“线与”功能

在选用GPIO时我们需要注意哪些参数和性能？

1.GPIO的电源域

在集成IC设计中，GPIO多分于几组不同的电源域进行驱动，而不同的电源域的驱动能力、管理方式、默认状态、电压匹配等等均会有所差异，在选用时需要全面考虑。

例如：设计选用GPIO来控制某模组，要求基带IC无论处于唤醒模式还是休眠模式下，该GPIO均处于默认输出高电平。该IC的GPIO有4组，分布对应电源域VDD1、VDD2、VDD3、VDD4。其中仅VDD1电源域无论唤醒模式还是休眠模式均保持供电，其他电源域休眠模式下均会关闭。所以我们只能选择在电源域VDD1下的GPIO。

针对低功耗设计时更需要注意GPIO的电源域。

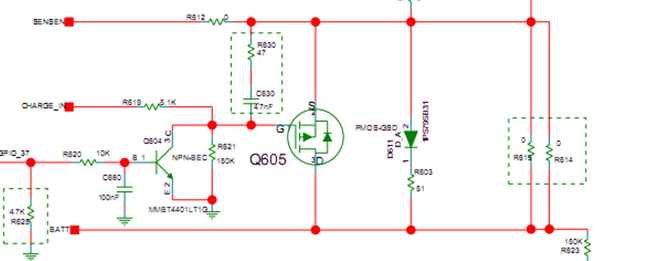
2.默认状态和电压匹配

GPIO在上电时有一个默认状态，此时改GPIO状态不受控，如果选择不当会对单板的可靠性带来影响。

在原理设计上时，我们需要考虑单板在上电的瞬间，底层的软件代码还未运行时，GPIO的默认状态是否会对单板造成不稳定或者失控的可能。

如下图：本图是锂电池充电电池部分，其中Q605为充电开关PMOS管。S端为输入电源端，D端为锂电池。当G为低电平时，Q605导通，开启充电功能。

如果CHARGE\_IN管脚为上电默认低电平，上电的瞬间Q605就会导通，而软件的控制逻辑可能还为正常运行，这时就会出现很多不稳定和可靠的问题。所以在GPIO的选择时其默认电平逻辑也是需要我们重点关注的。



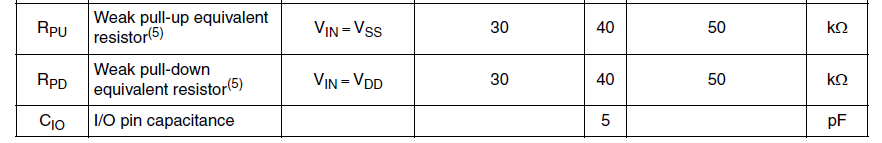
部分IC的GPIO电压是可控的，如3.3V或者1.8V，我们在选择时注意电压匹配。

3.上下拉的选择

通用的GPIO口都有可配置的上下拉，可以根据需要进行配置。在这里我们很容易忽略内部上下拉的电阻值，特别是在配置输入状态下，如何外部的电阻选择不当会直接导致我们输入电平异常，如下图：



根据设计逻辑，PIN1的输入电源应该为3.3V/2，而我们实际测试时发现PIN 1的电压只有1.1V。这样可能我们的整个设计逻辑就会出现问题。这个问题的原因是我们忽略了GPIO内部下拉电阻的阻值。通常的GPIO的上下拉电阻在几十到几百K左右，具体需要参考datesheet.例STM32 为40K。



4.驱动能力

由于GPIO的内部结构，其驱动能力都是有限的。通用的GPIO的拉电流，灌电流在10mA左右。在选择GPIO带负载时，如LED，必须考虑其驱动电流大小。

5.复用功能

通用的芯片的GPIO多半为复用管脚，在我们使用GPIO时必须严格对照datesheet查询选用的每一个管脚配置功能，如是否支持外部中断，输入频率等信息。

## 五 SCSI

SCSI——Small Computer Systems Interface，即小型计算机系统接口，是一种连结主机和外围设备的接口，支持包括磁盘驱动器、磁带机、光驱、扫描仪在内的多种设备。 SCSI最初被定义为通用并行的SCSI总线。SCSI总线自己不直接和诸如硬盘之类的 设备通讯，而是通过控制器联系。一个独立的SCSI总线最多支持16台设备，并通 过SCSI ID控制。其中，一个ID号被PC机里的SCSI适配器所使用，其余的ID号分配 给外设使用。SCSI的应用速度一般为10MB/s到40MB/s，现在已扩展到80MB/s和16 0MB/s。SCSI控制器有自己的命令集和缓存。要了解SCSI，必须先了解它的类型，以下是STA（SCSI Trade Association，SCSI同业公会）的标准分类。SCSI的类型注释：

①点到点传输的总线长度

②SCSI、Ultra SCSI或Ultra2 SCSI均是可选项

③LVD（Low Voltage Differential，低分差动）没有定义它的速度，在12米以内都能保持正常传输率。如果在总线内有一个设备设置成单终结，整个总线也会切换成单终结。

④单终结没有定义它的长度

⑤HVD（High Voltage Differential，高分差动）没有定义它的速度，在25米以内都能保持正常传输率。 ⑥在Ultra2之后，所有高速传输都是基于宽带（Wide）模式。

对于一个新用户来说，了解SCSI控制器和数据线的类型比接口类型更重要。在SCSI总线中，控制器也算一个设备，即实际最大可连接设备数目 = 理论最大支持设备数目-1。

SCSI规格公用的几个标准术语解释：  
 Single Ended（单终结）：许多旧式设备都是单终结设备，它们限制于SCSI-1协议的6米长度。注意：此距离包括设备内部电缆的距离。   
Differential（分差动）：SCSI总线和设备可借助它来沿长传输的距离，附加线的最大长度为25米。缺点是与单终结设备不兼容。

Fast SCSI：把第一代SCSI总线的速度从5MHz提高至10MHz，理论数据传输率也加倍到10MB/秒。

Ultra SCSI：把第一代SCSI总线的速度从5MHz提高至20MHz，理论数据传输率也加倍到20MB/秒。 Wide SCSI：它依靠第二条数据电缆或68针数据线来增加总线的性能，数据位宽为16或32 bits，把传统SCSI的性能提升至2倍或4倍。 Wide Ultra SCSI：利用68针数据线把总线性能提高到40MB/秒。 仅靠上面的描述，我们仍然不能准确地判断出一个SCSI总线的类型，必须同时了解它的总线宽度、总线速度、数据线类型和附加命令集才能达到目的。

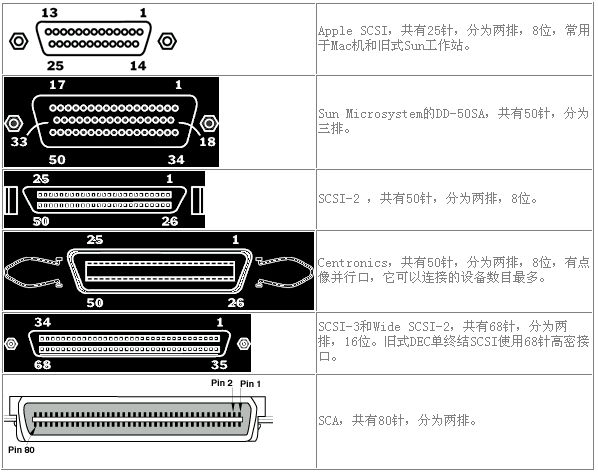
SCSI-1 以SASI（Shugart Associated System Interfaced）为基础所开发出的严密标准 SCSI-1介面，于1986年在ANSI（美国标准学会）的认可下诞生了，它被发表于ANSI X3.131中。 此介面定义了实体的连接介面标准，接头为50pin（50个接脚），最大的数据传输 速率为4MB/s，数据总线（DATA BUS，数据传输通道）的宽度是8bit。此外，它还 定义了SCSI的两种工作模式，一种是常看到的非同步模式（Asynchronous），另 一种则是同步模式（synchronous）。一般而言，非同步模式的传输速率比较差， 大约只有3~4MB/s而已，而同步模式的传输速率比较快，可达5MB/s。

SCSI-2 这也可以称做Fast-SCSI。总线时钟由5MHz倍增到10MHz，所以每秒内最大的数据 传输速率也由5MB/s提高到10MB/s，这便是Fast-SCSI一名的由来。SCSI-2也多了 Wide-SCSI的规格，提供更高的总线宽度，将原来8bit的宽度提高到16bit或32bi t，因而称为Wide-SCSI。SCSI-2 Fast&Wide介面的数据传输速率可达到20MB/s， 而在连接装置上，原本采用8bit的SCSI-1只使用3bit来定址，因此最多可以连接 8部装置，如果再扣掉一个必备的SCSI控制卡，实际上只能连接7部装置。Wide-S CSI采用4bit定址，所以就可以连接16部装置了。

Ultra-SCSI继SCSI-2之后，又发展出Ultra-SCSI，继续将数据传送的频率提高。它在SCSI-2 结构下将总线时钟增加到20MHz，所以在8bit的汇流排宽度上，最高数据传输速率 能倍增到20MB/s，因而得到FAST20- SCSI的别名，也有人将它称为SCSI-3 Fast- SCSI。另外，UltraWide-SCSI拥有16bit的宽度，所以数据传输速率最快已经可以达到40MB/s。

ULTRA2 SCSI 为了符合对输入/输出频带宽度效能的持续需求并延续与所有SCSI装置完全兼容的优势，Ultra2-SCSI遂于1998年问世。顾名思义，目前当红的SCSI介面——Ultra 2-SCSI（双UltraWdie-SCSI通道），就拥有Ultra-SCSI两倍的速度。它在8bit宽 度时拥有40MB/s的速度，所以又被称为FAST-40，而16bit的宽度则可获得80MB/s 的速度。另外，光是单个SCSI接口就可以支持多达15个外接设备。艾崴的一款主机板DBL100就集成此接口。 为了增强连接性与使用上的弹性以及增加信号传输的稳定度，Ultra2-SCSI就利用 了“低电压差动讯号”——LVD（Low Voltage Differential）技术来提供完整的 阻抗管理。这包含有两种模式（SE/LVD）的终端阻抗器，确保数据线最长可达12 米而数据传输信号不衰减。同时，数据线采用双绞线排列的方式，还可减少Cros s Talk干扰问题。 更有附加价值的是，由于LVD采用较低的电压，所以还可以替使用者降低在电源方面的损耗。  
（2）SCSI连接器的类型

SCSI连接器分为内置和外置两种，内置数据线的外型和IDE数据线一样，只是针数和规格稍有差别，主要用于连接光驱和硬盘。40针IDE线有40根导线，40针ATA66有80根导线，SCSI内置则分为50针、68针和80针。至于SCSI外置数据线，有以下几种规格：



（3）SCSI ID和总线终结器

ID（identify）作为SCSI设备在SCSI总线的唯一识别符，绝对不允许重复，可选范围从0到15，SCSI主控制器通常占用id 7，即是说我们可以用在设备上的ID号共有15个。

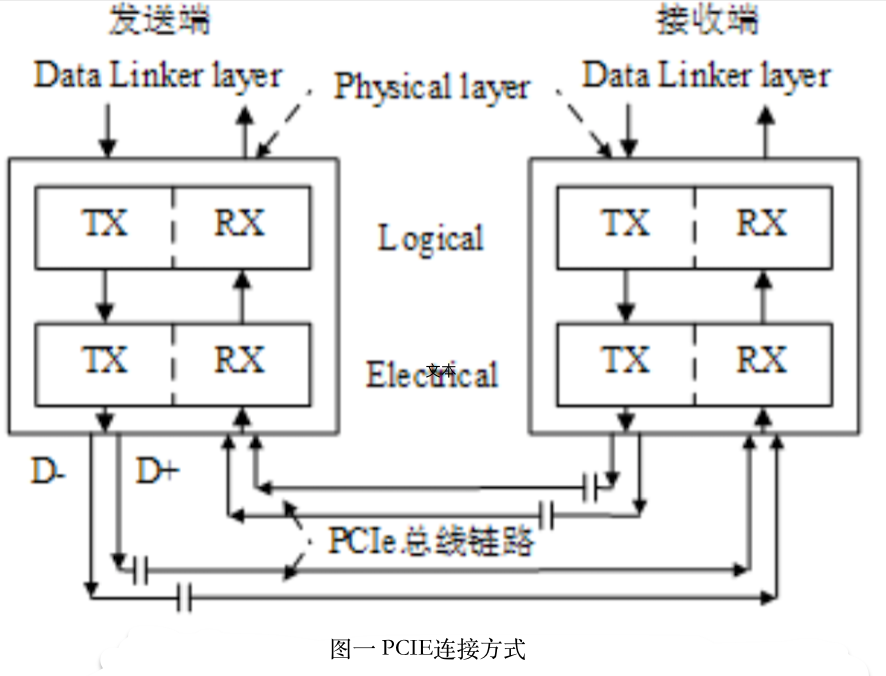
总线终结器能告诉SCSI主控制器整条总线在何处终结，并发出一个反射信号给控制器，必须在两个物理终端作一个终结信号才能使用SCSI总线。常见的错误是把终结设置在ID号最高或最低的地方，而不是设置在物理终端的SCSI设备上。其实，SCSI设备总是以链形来连接的，按顺序就能分辨出哪一个是终结设备。

## 六PCIe

PCIe（Peripheral Component Interconnect Express）总线的诞生与PC(Personal Computer)的蓬勃发展密切相关，是由PCISIG (PCI Special Interest Group，主要是intel)推出的一种局部并行总线标准，主要应用于电脑和服务器的主板上（目前几乎所有的主板都有PCIe的插槽），功能是连接外部设备（如显卡、存储、网卡、声卡、数据采集卡等）。

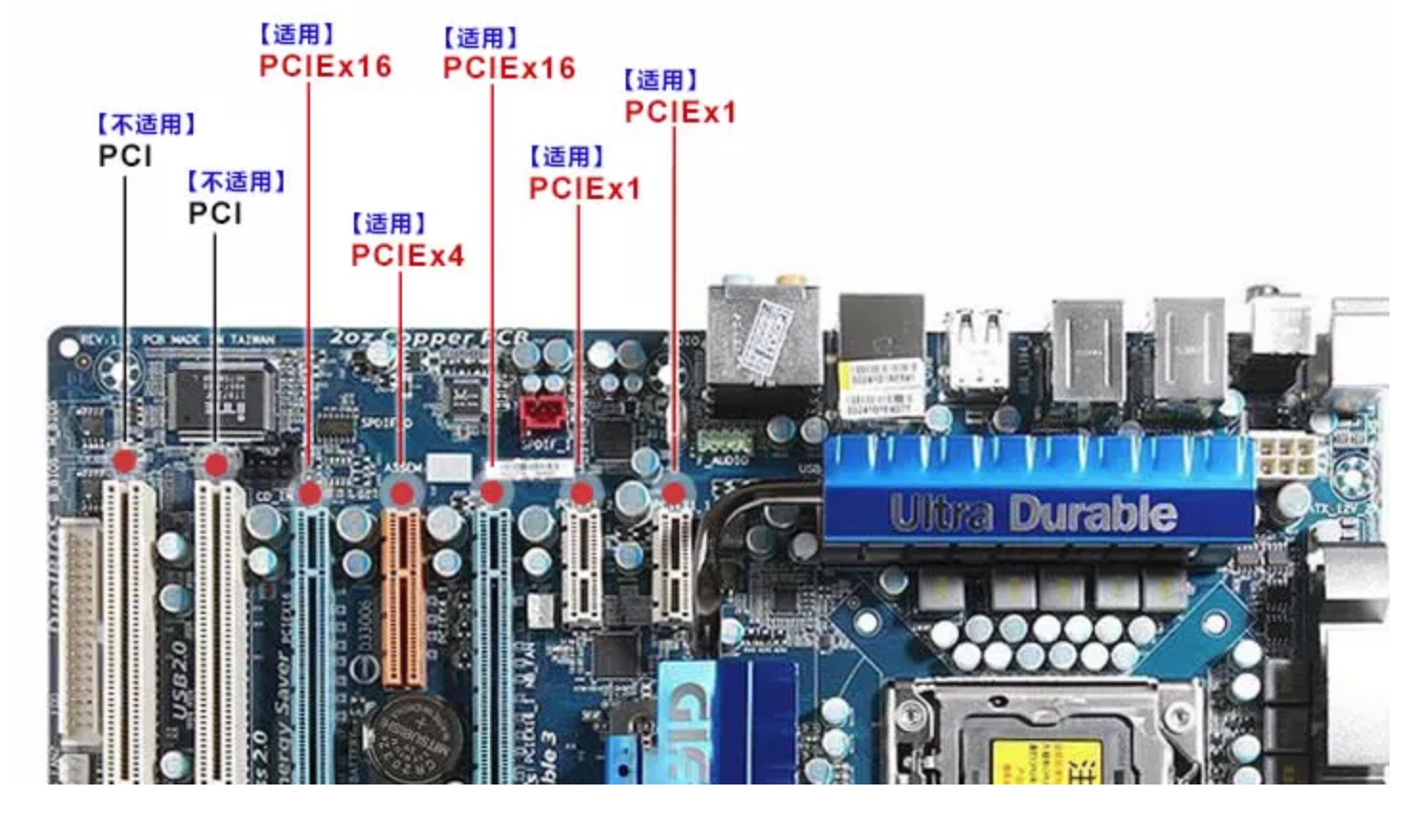
（1）连接方式

PCIe链路使用“端到端的数据传送方式”，发送端和接收端中都含有TX(发送逻辑)和RX(接收逻辑)， 其结构如图一。



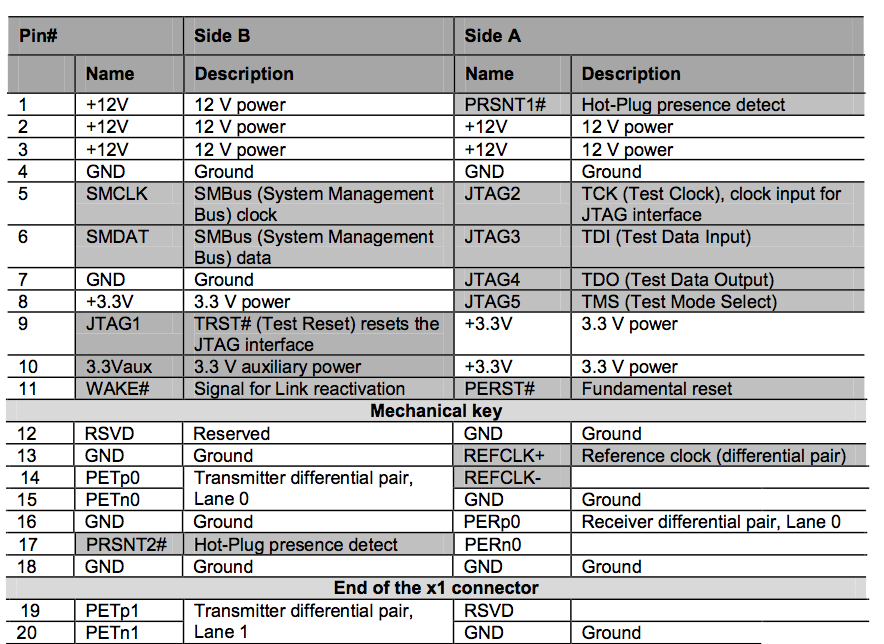
由上图所示，在PCIe总线的物理链路的一个数据通路(Lane)中，由两组差分信号，共4根信号线组成。其中发送端的TX部件与接收端的RX部件使用一组差分信号连接，该链路也被称为发送端的发送链路，也是接收端的接收链路；而发送端的RX部件与接收端的TX部件使用另一组差分信号连接，该链路也被称为发送端的接收链路，也是接收端的发送链路。PCIe链路使用差分信号进行数据传送，一个差分信号由D+和D-两根信号组成，信号接收端通过比较这两个信号的差值，判断发送端发送的是逻辑“1”还是逻辑“0”。

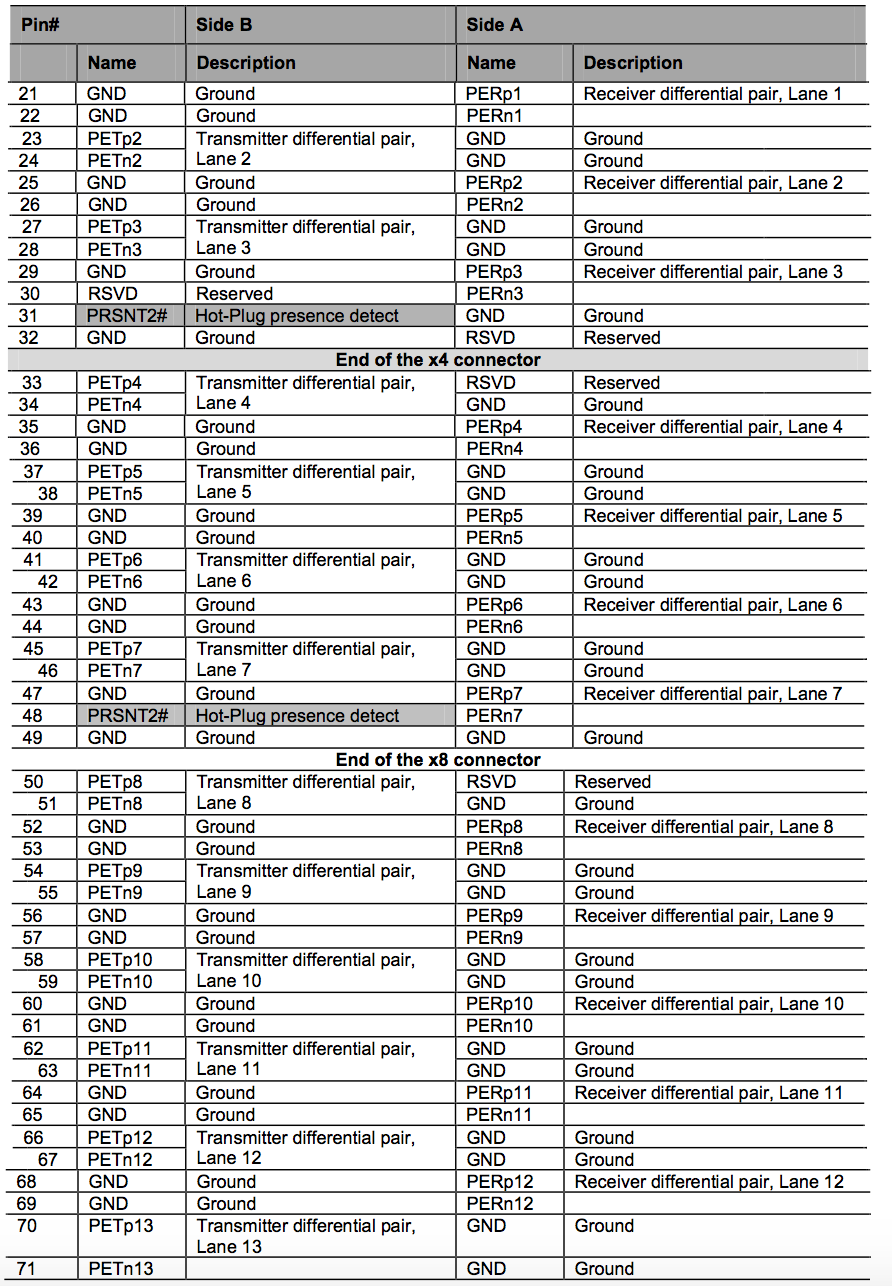
PCIe链路可以由多条Lane组成，目前PCIe链路可以支持1、2、4、8、12、16和32个Lane，即×1、×2、×4、×8、×12、×16和×32宽度的PCIe链路。在主板上常见的是PCIe X1、PCIe X4和PCIe X16接口；

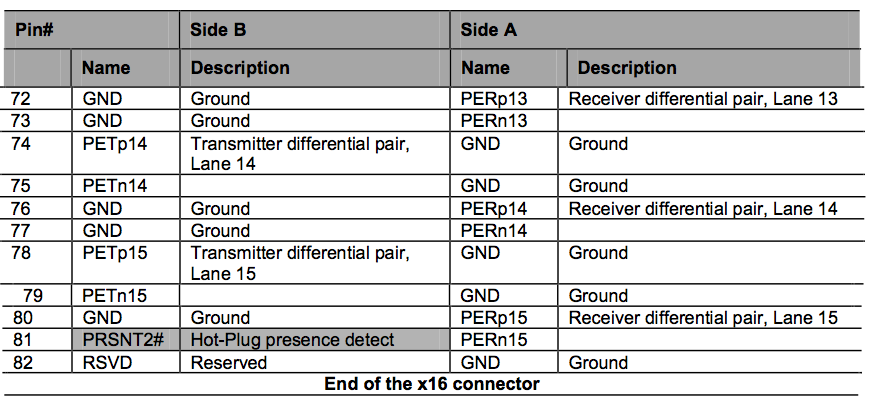


（2）PCIe的信号线

PCIe的接口保证了前向兼容（3.X/2.X/1.X的信号定义一样，只是信号线上的速度和电平不同），主要分为电源、信号传输lane和一些辅助信号。PCIe插槽的pin脚定义如下（灰色背景色的是辅助信号）：







电源有＋12V、＋3.3V和3.3Vaux。

其中＋12V主要给PCIe设备（如显卡）提供更大的供电能力（PCIe 3.0 的插槽最大有150W）；

PCIe设备使用的主要逻辑模块均使用＋3.3V供电，对于芯片内部的PCIe模块，大多只需要＋3.3V；

而一些与电源管理相关的逻辑使用3.3Vaux供电。在PCIe设备中，一些特殊的寄存器通常使用3.3Vaux供电，如Sticky Register，此时即使PCIe设备的其他电源被移除，这些与电源管理相关的逻辑状态和这些特殊寄存器的内容也不会发生改变。在PCIe总线中，使用3.3Vaux的主要原因是为了降低功耗和缩短系统恢复时间。因为3.3Vaux在多数情况下并不会被移除，因此当PCIC设备的电源恢复（唤醒）后，该设备不用重新恢复使用3.3Vaux供电的逻辑，从而设备可以很快地恢复到正常工作状状态。

PETp0~15、PETn0~15和PERp0~15、PERn0~15共64根信号线组成32对差分信号，其中16对PETxx信号用于发送链路，另外16对PERxx信号用于接收链路。

辅助信号：

1、PERST＃信号：该信号为全局复位信号，由主系统提供，主系统需要为PCIe插槽和PCIe设备提供该复位信号。PCIe设备使用该信号复位内部逻辑。当该信号有效时，PCIe设备将进行复位操作。

2、REFCLK+和REFCLK-信号：参考时钟，其频率范围为100MHz±300ppm。当PCIe设备作为Add-In卡连接在PCIe插槽时，可以直接使用PCIe插槽提供的REFCLK+和REFCLK-信号，也可以使用独立的参考时钟，只要这个参考时钟在100MHz±300ppm范围内即可。

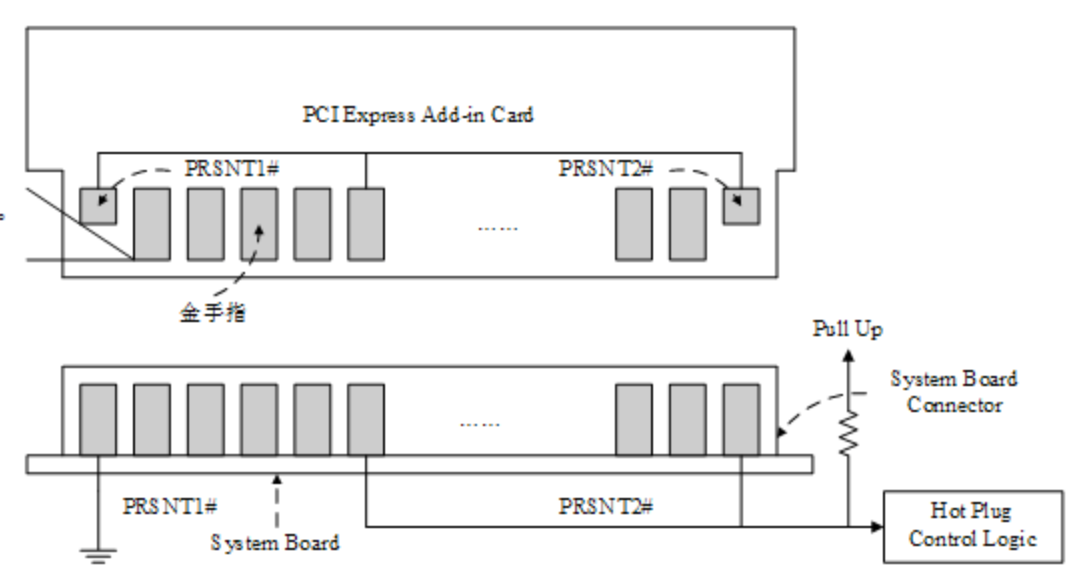
3、WAKE#信号：当PCIe设备进入休眠状态，主电源已经停止供电时，PCIe设备使用该信号向处理器系统提交唤醒请求，使处理器系统重新为该PCIe设备提供主电源Vcc。在PCIe总线中，WAKE#信号是可选的，因此使用WAKE#信号唤醒PCIe设备的机制也是可选的。值得注意的是产生该信号的硬件逻辑必须使用辅助电源Vaux供电。

PCIe设备除了可以使用WAKE#信号实现唤醒功能外，还可以使用Beacon信号实现唤醒功能。与WAKE#信号实现唤醒功能不同，Beacon使用In-band信号，即差分信号D+和D-实现唤醒功能。Beacon信号DC平衡，由一组通过D+和D-信号生成的脉冲信号组成。这些脉冲信号宽度的最小值为2ns，最大值为16us。当PCIe设备准备退出L2状态(该状态为PCIe设备使用的一种低功耗状态)时，可以使用Beacon信号，提交唤醒请求。

4、SMCLK和SMDAT信号：SMCLK和SMDAT信号与x86处理器的SMBus(System Mangement Bus)相关。SMBus于1995年由Intel提出，SMBus由SMCLK和SMDAT信号组成。SMBus源于I2C总线，以便于PCI/PCIe设备与处理器系统进行交互。

5、JTAG信号：JTAG(Joint Test Action Group)是一种国际标准测试协议，与IEEE 1149.1兼容，主要用于芯片内部测试。目前绝大多数器件都支持JTAG测试标准。JTAG信号由TRST#、TCK、TDI、TDO和TMS信号组成。其中TRST#为复位信号；TCK为时钟信号；TDI和TDO分别与数据输入和数据输出对应；而TMS信号为模式选择。

6、PRSNT1#和PRSNT2#信号：PRSNT1#和PRSNT2#信号与PCIe设备的热插拔相关。在基于PCIe总线的Add-in卡中，PRSNT1#和PRSNT2#信号直接相连，而在处理器主板中，PRSNT1#信号接地，而PRSNT2#信号通过上拉电阻接为高。PCIe设备的热插拔结构如下图所示。

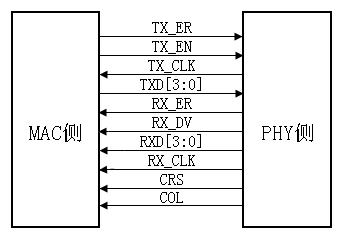


如上图所示，当Add-In卡没有插入时，处理器主板的PRSNT2#信号由上拉电阻接为高，而当Add-In卡插入时主板的PRSNT2#信号将与PRSNT1#信号通过Add-In卡连通，此时PRSNT2#信号为低。处理器主板的热插拔控制逻辑将捕获这个“低电平”，得知Add-In卡已经插入，从而触发系统软件进行相应地处理。

## 七MII

MII是英文Medium Independent Interface的缩写，翻译成中文是“介质独立接口”，该接口一般应用于以太网硬件平台的MAC层和PHY层之间，MII接口的类型有很多，常用的有MII、RMII、SMII、SSMII、SSSMII、GMII、RGMII、SGMII、TBI、RTBI、XGMII、XAUI、XLAUI等。下面对它们进行一一介绍。

MII接口：

[](http://download.bbs.ednchina.com/images/attachments/201402/original/9906487833_TIME_1393678329373.jpg" \t "http://bbs.ednchina.com/_blank)

TXD(Transmit Data)[3:0]：数据发送信号，共4根信号线；

RXD(Receive Data)[3:0]：数据接收信号，共4根信号线；

TX\_ER(Transmit Error)：  发送数据错误提示信号，同步于TX\_CLK，高电平有效，表示TX\_ER有效期内传输的数据无效。对于10Mbps速率下，TX\_ER不起作用；

RX\_ER(Receive Error)：  接收数据错误提示信号，同步于RX\_CLK，高电平有效，表示RX\_ER有效期内传输的数据无效。对于10Mbps速率下，RX\_ER不起作用；

TX\_EN(Transmit Enable)： 发送使能信号，只有在TX\_EN有效期内传的数据才有效；

RX\_DV(Reveive Data Valid)： 接收数据有效信号，作用类型于发送通道的TX\_EN；

TX\_CLK：发送参考时钟，100Mbps速率下，时钟频率为25MHz，10Mbps速率下，时钟频率为2.5MHz。注意，TX\_CLK时钟的方向是从PHY侧指向MAC侧的，因此此时钟是由PHY提供的。

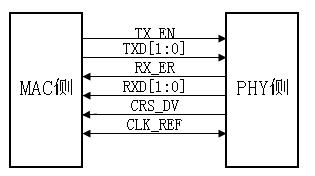
RX\_CLK：接收数据参考时钟，100Mbps速率下，时钟频率为25MHz，10Mbps速率下，时钟频率为2.5MHz。RX\_CLK也是由PHY侧提供的。

CRS：Carrier Sense，载波侦测信号，不需要同步于参考时钟，只要有数据传输，CRS就有效，另外，CRS只有PHY在半双工模式下有效；

COL：Collision Detectd，冲突检测信号，不需要同步于参考时钟，只有PHY在半双工模式下有效。

MII接口一共有16根线。

RMII接口：

[](http://download.bbs.ednchina.com/images/attachments/201402/original/9906487833_TIME_1393678372134.jpg" \t "http://bbs.ednchina.com/_blank)

RMII即Reduced MII，是MII的简化板，连线数量由MII的16根减少为8根。

TXD[1:0]：数据发送信号线，数据位宽为2，是MII接口的一半；

RXD[1:0]：数据接收信号线，数据位宽为2，是MII接口的一半；

TX\_EN(Transmit Enable)：数据发送使能信号，与MII接口中的该信号线功能一样；

RX\_ER(Receive Error)：数据接收错误提示信号，与MII接口中的该信号线功能一样；

CLK\_REF：是由外部时钟源提供的50MHz参考时钟，与MII接口不同，MII接口中的接收时钟和发送时钟是分开的，而且都是由PHY芯片提供给MAC芯片的。这里需要注意的是，由于数据接收时钟是由外部晶振提供而不是由载波信号提取的，所以在PHY层芯片内的数据接收部分需要设计一个FIFO，用来协调两个不同的时钟,在发送接收的数据时提供缓冲。PHY层芯片的发送部分则不需要FIFO，它直接将接收到的数据发送到MAC就可以了。

CRS\_DV：此信号是由MII接口中的RX\_DV和CRS两个信号合并而成。当介质不空闲时，CRS\_DV和RE\_CLK相异步的方式给出。当CRS比RX\_DV早结束时(即载波消失而队列中还有数据要传输时)，就会出现CRS\_DV在半位元组的边界以25MHz/2.5MHz的频率在0、1之间的来回切换。因此，MAC能够从 CRS\_DV中精确的恢复出RX\_DV和CRS。

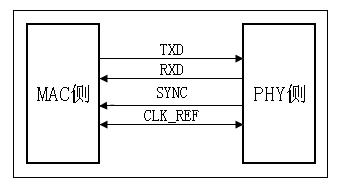
在100Mbps速率时，TX/RX每个时钟周期采样一个数据；在10Mbps速率时，TX/RX每隔10个周期采样一个数据，因而TX/RX数据需要在数据线上保留10个周期，相当于一个数据发送10次。

当PHY层芯片收到有效的载波信号后，CRS\_DV信号变为有效，此时如果FIFO中还没有数据，则它会发送出全0的数据给MAC，然后当FIFO中填入有效的数据帧，数据帧的开头是“101010---”交叉的前导码，当数据中出现“01”的比特时，代表正式数据传输开始，MAC芯片检测到这一变化，从而开始接收数据。

当外部载波信号消失后，CRS\_DV会变为无效，但如果FIFO中还有数据要发送时，CRS\_DV在下一周期又会变为有效，然后再无效再有效，直到FIFO中数据发送完为止。在接收过程中如果出现无效的载波信号或者无效的数据编码，则RX\_ER会变为有效，表示物理层芯片接收出错。

SMII接口：

SMII即Serial MII，串行MII的意思，跟RMII相比，连线进一步减少到4根；



TXD：发送数据信号，位宽为1；

RXD：接收数据信号，位宽为1；

SYNC：收发数据同步信号，每10个时钟周期置1次高电平，指示同步。

CLK\_REF：所有端口共用的一个参考时钟，频率为125MHz，为什么100Mbps速率要用125MHz时钟？因为在每8位数据中会插入2位控制信号，请看下面介绍。

TXD/RXD以10比特为一组，以SYNC为高电平来指示一组数据的开始，在SYNC变高后的10个时钟周期内，TXD上依次输出的数据是：TXD[7:0]、TX\_EN、TX\_ER，控制信号的含义与MII接口中的相同；RXD上依次输出的数据是：RXD[7:0]、RX\_DV、CRS，RXD[7:0]的含义与RX\_DV有关，当RX\_DV为有效时(高电平)，RXD[7:0]上传输的是物理层接收的数据。当RX\_DV为无效时(低电平)，RXD[7:0]上传输的是物理层的状态信息数据。见下表：

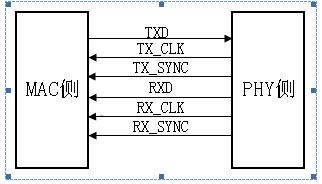


当速率为10Mbps时，每一组数据要重复10次，MAC/PHY芯片每10个周期采样一次。

MAC/PHY芯片在接收到数据后会进行串/并转换。

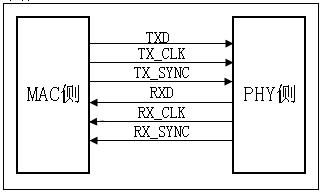
 SSMII接口：

SSMII即Serial Sync MII，叫串行同步接口，跟SMII接口很类似，只是收发使用独立的参考时钟和同步时钟，不再像SMII那样收发共用参考时钟和同步时钟，传输距离比SMII更远。

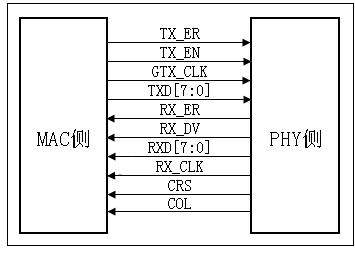


SSSMII接口：

SSSMII即Source Sync Serial MII，叫源同步串行MII接口，SSSMII与SSMII的区别在于参考时钟和同步时钟的方向，SSMII的TX/RX参考时钟和同步时钟都是由PHY芯片提供的，而SSSMII的TX参考时钟和同步时钟是由MAC芯片提供的，RX参考时钟和同步时钟是由PHY芯片提供的，所以顾名思义叫源同步串行。



GMII接口：



与MII接口相比，GMII的数据宽度由4位变为8位，GMII接口中的控制信号如TX\_ER、TX\_EN、RX\_ER、RX\_DV、CRS和COL的作用同MII接口中的一样，发送参考时钟GTX\_CLK和接收参考时钟RX\_CLK的频率均为125MHz(1000Mbps/8=125MHz)。

在这里有一点需要特别说明下，那就是发送参考时钟GTX\_CLK，它和MII接口中的TX\_CLK是不同的，MII接口中的TX\_CLK是由PHY芯片提供给MAC芯片的，而GMII接口中的GTX\_CLK是由MAC芯片提供给PHY芯片的。两者方向不一样。

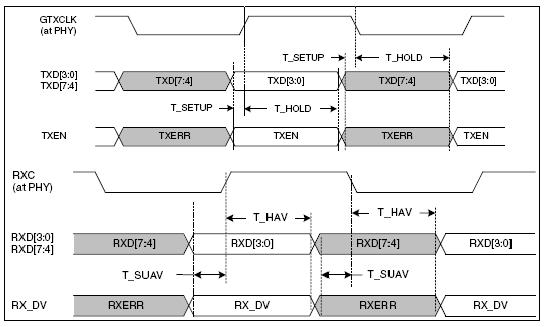
在实际应用中，绝大多数GMII接口都是兼容MII接口的，所以，一般的GMII接口都有两个发送参考时钟：TX\_CLK和GTX\_CLK(两者的方向是不一样的，前面已经说过了)，在用作MII模式时，使用TX\_CLK和8根数据线中的4根。

RGMII接口：

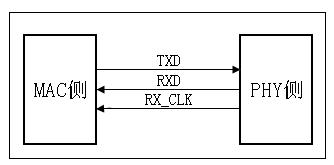


RGMII即Reduced GMII，是RGMII的简化版本，将接口信号线数量从24根减少到14根(COL/CRS端口状态指示信号，这里没有画出)，时钟频率仍旧为125MHz，TX/RX数据宽度从8为变为4位，为了保持1000Mbps的传输速率不变，RGMII接口在时钟的上升沿和下降沿都采样数据。在参考时钟的上升沿发送GMII接口中的TXD[3:0]/RXD[3:0]，在参考时钟的下降沿发送GMII接口中的TXD[7:4]/RXD[7:4]。RGMI同时也兼容100Mbps和10Mbps两种速率，此时参考时钟速率分别为25MHz和2.5MHz。

TX\_EN信号线上传送TX\_EN和TX\_ER两种信息，在TX\_CLK的上升沿发送TX\_EN，下降沿发送TX\_ER；同样的，RX\_DV信号线上也传送RX\_DV和RX\_ER两种信息，在RX\_CLK的上升沿发送RX\_DV，下降沿发送RX\_ER。



SGMII接口：



SGMII即Serial GMII，串行GMII，收发各一对差分信号线，时钟频率625MHz，在时钟信号的上升沿和下降沿均采样，参考时钟RX\_CLK由PHY提供，是可选的，主要用于MAC侧没有时钟的情况，一般情况下，RX\_CLK不使用。收发都可以从数据中恢复出时钟。

在TXD发送的串行数据中，每8比特数据会插入TX\_EN/TX\_ER 两比特控制信息，同样，在RXD接收数据中，每8比特数据会插入RX\_DV/RX\_ER 两比特控制信息，所以总的数据速率为1.25Gbps=625Mbps\*2.

其实，大多数MAC芯片的SGMII接口都可以配置成SerDes接口(在物理上完全兼容，只需配置寄存器即可)，直接外接光模块，而不需要PHY层芯片，此时时钟速率仍旧是625MHz，不过此时跟SGMII接口不同，SGMII接口速率被提高到1.25Gbps是因为插入了控制信息，而SerDes端口速率被提高是因为进行了8B/10B变换，本来8B/10B变换是PHY芯片的工作，在SerDes接口中，因为外面不接PHY芯片，此时8B/10B变换在MAC芯片中完成了。8B/10B变换的主要作用是扰码，让信号中不出现过长的连“0”和连“1”情况，影响时钟信息的提取，关于8B/10B变换知识，我后续会单独介绍。

TBI接口：

TBI即Ten Bit Interface的意思，接口数据位宽由GMII接口的8位增加到10位，其实，TBI接口跟GMII接口的差别不是很大，多出来的2位数据主要是因为在TBI接口下，MAC芯片在将数据发给PHY芯片之前进行了8B/10B变换(8B/10B变换本是在PHY芯片中完成的，前面已经说过了)，另外，RX\_CLK+/-是从接收数据中恢复出来的半频时钟，频率为62.5MHz，RX\_CLK+/-不是差分信号，而是两个独立的信号，两者之间有180度的相位差，在这两个时钟的上升沿都采样数据。RX\_CLK+/-也叫伪差分信号。除掉上面说到的之外，剩下的信号都跟GMII接口中的相同。

大多数芯片的TBI接口和GMII接口兼容。在用作TBI接口时，CRS和COL一般不用。

RTBI接口：

RTBI即Reduced TBI，简化版TBI，接口数据位宽为5bit，时钟频率为125MHz，在时钟的上升沿和下降沿都采样数据，同RGMII接口一样，TX\_EN线上会传送TX\_EN和TX\_ER两种信息，在时钟的上升沿传TX\_EN，下降沿传TX\_ER；RX\_DV线上传送RX\_DV和RX\_ER两种信息，在RX\_CLK上升沿传RX\_DV，下降沿传RX\_ER。

万兆以太网接口的端口速率为10Gbps，主要有XGMII和XAUI两种，另外还有HIGIG，不过HIGIG是Broadcom公司的私有标准，这里暂不介绍。

XGMII接口：

TXD[31:0]：数据发送通道，32位并行数据。

RXD[31:0]：数据接收通道，32位并行数据。

TXC[3:0]：发送通道控制信号，TXC=0时，表示TXD上传输的是数据；TXC=1时，表示TXD上传输的是控制字符。TXC[3:0]分别对应TXD[31:24], TXD[23:16], TXD[15:8], TXD[7:0]。

RXC[3:0]：接收通道控制信号，RXC=0时，表示RXD上传输的是数据；RXC=1时，表示RXD上传输的是控制字符。RXC[3:0]分别对应RXD[31:24], RXD[23:16], RXD[15:8], RXD[7:0]。

TX\_CLK：TXD和TXC的参考时钟，时钟频率156.25MHz，在时钟信号的上升沿和下降沿都采样数据。156.25MHz \* 2 \* 32 = 10Gbps 。

RX\_CLK：RXD和RXC的参考时钟，时钟频率156.25MHz，在时钟信号的上升沿和下降沿都采样数据。

XGMII接口共74根连线，单端信号，采用HSTL/SSTL\_2逻辑，端口电压1.5V/2.5V，由于SSTL\_2的端口电压高，功耗大，现在已很少使用。HSTL即High Speed Transceiver Logic，高速发送逻辑的意思。SSTL，即Stub Series Terminated Logic，短路终止逻辑，主要用于高速内存接口，SSTL目前存在两种标准，SSTL\_3是3.3V标准；SSTL\_2是2.5V标准。

XAUI接口：

由于受电气特性的影响，XGMII接口的PCB走线最大传输距离仅有7cm，并且XGMII接口的连线数量太多，给实际应用带来不便，因此，在实际应用中，XGMII接口通常被XAUI接口代替，XAUI即10 Gigabit attachment unit interface，10G附属单元接口，XAUI在XGMII的基础上实现了XGMII接口的物理距离扩展，将PCB走线的传输距离增加到50cm，使背板走线成为可能。

源端XGMII把收发32位宽度数据流分为4个独立的lane通道，每个lane通道对应一个字节，经XGXS(XGMII Extender Sublayer)完成8B/10B编码后，将4个lane分别对应XAUI的4个独立通道，XAUI端口速率为：2.5Gbps \* 1.25 \* 4＝12.5Gbps。

在发送端的XGXS模块中，将TXD[31:0]/ RXD[31:0],TXC[3:0]/ RXC[3:0], TX\_CLK/ RX\_CLK转换成串行数据从TX Lane[3:0]/ RX Lane[3:0]中发出去，在接收端的XGXS模块中，串行数据被转换成并行，并且进行时钟恢复和补偿，完成时钟去抖，经过5B/4B解码后，重新聚合成XGMII。

XAUI接口采用差分线，收发各四对，CML逻辑，AC耦合方式，耦合电容在10nF~100nF之间。

XAUI接口可以直接接光模块，如XENPAK/X2等。也可以转换成一路10G信号XFI，接XFP/SFP+等。

有些芯片不支持XAUI接口，只支持XGMII接口，这时可以用专门的芯片进行XGMIIà XAUI接口转换，如BCM8011等。

## 八VGA

VGA的英文全称是Video Graphic Array，也叫显示绘图阵列。 虽然液晶显示器可以直接接收数字信号，但为了兼容性，大多数液晶显示器也配备了VGA接口。

VGA接口是一种D型接口（D-SUB），上面共有15针空，分成三排，每排五个。而与之配套的底座则为孔型接口。

底座：

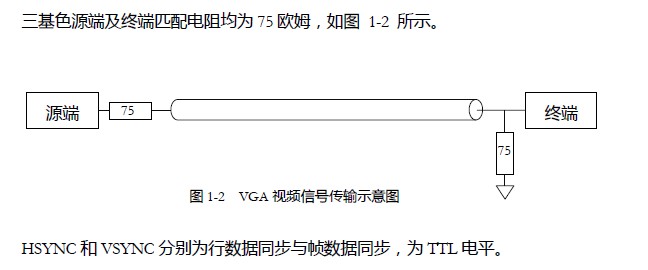


1

VGA电气特性

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| VGA引脚定义如表1所示。 表1： 引脚 | 名称 | 描述 | 名称 | 描述 |
| 1 | RED | RED Video | KEY | Key (No pin) |
| 2 | GREEN | Green Video | SGND | Sync Ground |
| 3 | BLUE | Blue Video | ID0 | Monitor ID bit 0 |
| 4 | ID2 | Monitor ID bit 2 | ID1 | Monitor ID bit 1 |
| 5 | GND | Ground | HSYNC | Horizontal Sync |
| 6 | RGND | Red Ground | VSYNC | Vertical Sync |
| 7 | GGND | Green Ground | ID3 | Monitor ID bit 3 |
| 8 | | Blue Ground | |
|  | |  | |

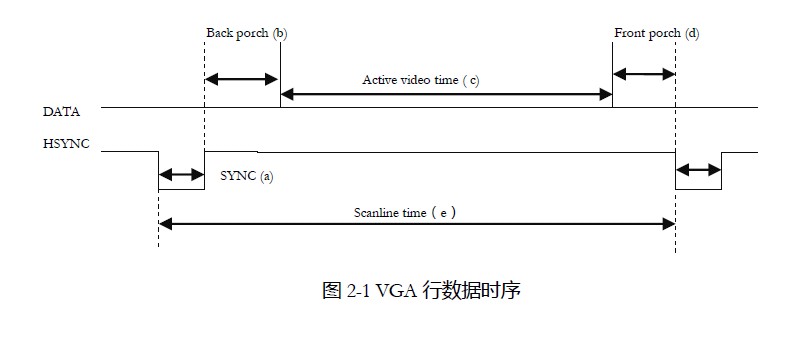
引脚1、2、3分别为红绿蓝三基色模拟电压，为0~0.714V peak-peak（峰-峰值），0V代表无色，0.714V代表满色。一些非标准显示器使用的是 1Vpp的满色电平。

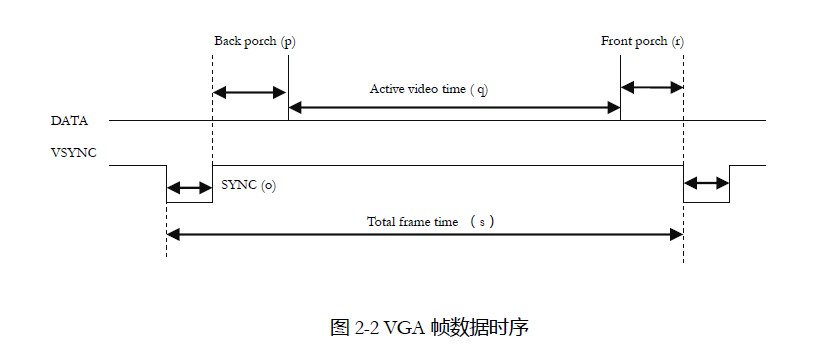


时序实现

一、 时序介绍

VGA的时序如图 2-1、图2-2 所示。它分为行数据时序和帧数据时序。





行数据时序，顾名思义，也就是显示一行数据的时序。从图2-1可以看出，显示一行数据需要处理两件事情。第一：产生行同步 HSYNC。不难看出，HSYNC是一个脉冲信号，此信号的周期为： e=a+b+c+d，低电平时间为 a。其中 a、b、c、d均为时间信号，这些信号根据需要显示的分辨率不同而不同。第二：产生显示的数据（DATA）信号， 此信号为模拟信号，当在显示有效数据（Active video）内，DATA信号为0~0.714Vpp的模拟电压（R、G、B），根据分辨率的不同，DATA的采样率、点数也皆不相同。

帧数据时序不行时序类似，也就是显示一屏数据的时序。只是这里的基本单位为每行数据，而行数据里面的最基本单位为每个点。

上面提过，不同的分辨率，时序上的时间是不一样的。表2、表3 列出常用分辨率及时间参数。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 表2 常见刷新率时序表： 显示模式 | 时钟 （MHz） | 行时序（像素数） | | | | | 帧时序（行数） | | | | |
| a | b | c | d | e | o | p | q | r | s |
| 640x480@60 | 25.175 | 96 | 48 | 640 | 16 | 800 | 2 | 33 | 480 | 10 | 525 |
| 640x480@75 | 31.5 | 64 | 120 | 640 | 16 | 840 | 3 | 16 | 480 | 1 | 500 |
| 800x600@60 | 40.0 | 128 | 88 | 800 | 40 | 1056 | 4 | 23 | 600 | 1 | 628 |
| 800x600@75 | 49.5 | 80 | 160 | 800 | 16 | 1056 | 3 | 21 | 600 | 1 | 625 |
| **1024x768@60** | **65** | **136** | **160** | **1024** | **24** | **1344** | **6** | **29** | **768** | **3** | **806** |
| 1024x768@75 | 78.8 | 176 | 176 | 1024 | 16 | 1312 | 3 | 28 | 768 | 1 | 800 |
| 1280x1024@60 | 108.0 | 112 | 248 | 1280 | 48 | 1688 | 3 | 38 | 1024 | 1 | 1066 |
| 1280x800@60 | 83.46 | 136 | 200 | 1280 | 64 | 1680 | 3 | 24 | 800 | 1 | 828 |
| 1440x900@60 | 106.47 | 152 | 232 | 1440 | 80 | 1904 | 3 | 28 | 900 | 1 | 932 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 表3常见刷新率时序表（时间）： 显示模式 | 时钟 （MHz） | 行时序时间（μs） | | | | | 帧时序时间（ms） | | | | | |
| a | b | c | d | e | o | p | q | r | s |
| 640x480@60 | 25.175 | 3.81 | 1.9 | 25.4 | 0.635 | 31.7 | .006 | 1.048 | 15.25 | 0.317 | 16.6 | |
| 640x480@75 | 31.5 | 2.03 | 3.8 | 20.3 | 0.507 | 26.7 | 0.08 | 0.426 | 12.8 | 0.026 | 13.3 | |
| 800x600@60 | 40.0 | 3.2 | 2.2 | 20 | 1 | 26.4 | 0.1 | 0.6 | 15.84 | 0.026 | 16.6 | |
| 800x600@75 | 49.5 | 1.62 | 3.2 | 16.16 | 0.323 | 21.3 | 0.06 | 0.45 | 12.8 | 0.021 | 13.3 | |
| **1024x768@60** | **65** | **2.09** | **2.46** | **15.7** | **0.37** | **20.6** | **0.12** | **0.599** | **15.87** | **0.062** | **16.6** | |
| 1024x768@75 | 65 | 1.22 | 2.23 | 12.99 | 0.203 | 16.6 | 0.05 | 0.466 | 12.78 | 0.016 | 13.3 | |
| 1280x1024@60 | 108 | 1.04 | 2.3 | 11.85 | 0.444 | 15.6 | 0.05 | 0.6 | 16.0 | 0.015 | 16.6 | |
| 1280x800@60 | 83.46 | 1.63 | 2.4 | 15.3 | 0.76 | 20.1 | 0.06 | 0.48 | 16.1 | 0.02 | 16.6 | |
| 1440x900@60 | 106.47 | 1.43 | 2.18 | 13.52 | 0.75 | 17.9 | 0.05 | 0.5 | 16.1 | 0.017 | 16.6 | |

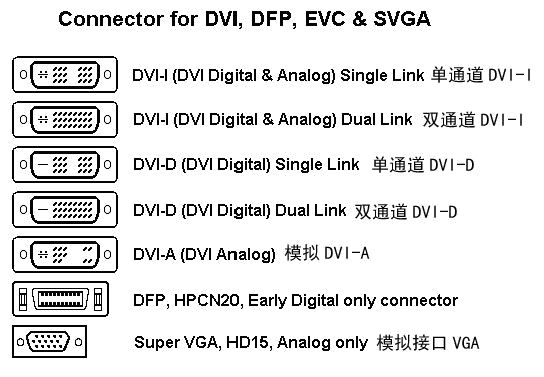
## 九DVI

DVI全称为Digital Visual Interface，它是1999年由Silicon Image、Intel（英特尔）、Compaq（康柏）、IBM、HP（[惠普](http://detail.zol.com.cn/notebook_index/subcate16_223_list_1.html" \t "_blank)）、NEC、Fujitsu（[富士通](http://detail.zol.com.cn/hard_drives_index/subcate2_283_list_1.html" \t "_blank)）等公司共同组成DDWG（Digital Display Working Group，数字显示工作组）推出的接口标准。  
  
 它是以Silicon Image公司的PanalLink接口技术为基础，基于TMDS（Transition Minimized Differential Signaling，最小化传输差分信号）电子协议作为基本电气连接。TMDS是一种微分信号机制，可以将象素数据编码，并通过串行连接传递。[显卡](http://detail.zol.com.cn/vga_index/subcate6_list_1.html" \t "_blank)产生的数字信号由发送器按照TMDS协议编码后通过TMDS通道发送给接收器，经过解码送给数字显示设备。

**

**2、DVI接口的阵脚定义：**

　　一个DVI显示系统包括一个传送器和一个接收器。传送器是信号的来源，可以内建在显卡芯片中，也可以以附加芯片的形式出现在显卡PCB上；而接收器则是显示器上的一块电路，它可以接受数字信号，将其解码并传递到数字显示电路中，通过这两者，显卡发出的信号成为显示器上的图象。  
　　和传统的VGA信号相比，采用DVI信号的液晶显示器不存在相位问题，不会引起像素抖动。另外，采用DVI信号后，显示器不会造成几何失真，大大提高了画面的质量。  
DVI的不同标准  
　　和VGA接口不同，DVI接口存在很多标准，使用显示器时一定要搞明白。  
　　从下面的图表中可以看到，DVI一共分为5种标准。其中DVI-D和**DVI-I**分为“双通道”和“单通道”两种类型，我们平时见到的都是单通道版的，双通道版的成本很高，因此只有部分专业设备才具备，普通消费者很难见到。



DVI-A是一种模拟传输标准，晚期的大屏幕专业CRT中能看见。不过由于和VGA没有本质区别，性能也不高，因此DVI-A事实上已经被废弃了。

DFP接口，这是一种已经被废弃的早期的数字规范。

**3、关于DVI信号的各种特性，见下表：**

|  |  |  |
| --- | --- | --- |
| 规格 | 信号 | 备注 |
| **DVI-I**双通道24+5 | 数字/模拟 | 可转换VGA |
| **DVI-I**单通道18+5 | 数字/模拟 | 可转换VGA |
| DVI-D双通道24+1 | 数字 | 不可转换VGA |
| DVI-D单通道18+1 | 数字 | 不可转换VGA |
| DVI-A | 模拟 | 已废弃 |
| DFP | 数字 | 已废弃 |
| VGA | 模拟 | —— |

## 十HDMI

清晰度多媒体接口(英文:High Definition Multimedia Interface，HDMI)是一种数字化视频/音频接口技术，是适合影像传输的专用型数字化接口，其可同时传送音频和影像信号，最高数据传输速度为2.25GB/s。

|  |  |
| --- | --- |
| **HDMI AType** | |
| 应用于HDMI1.0版本，总共有19pin, 规格为4.45 mm × 13.9 mm，为最常见的HDMI接头规格, 相对等于DVI Single-Link传输。在HDMI 1.2a之前, 最大能传输165MHz的TMDS, 所以最大传输规格只能在于1600x1200(TMDS 162.0 MHz) 。 | |
| [HDMI脚位配置](http://baike.baidu.com/picview/50611/50611/0/ad11a766gd981d451f4b981d.html" \t "_blank) 引脚位置 | Pin定义 |
| 1 | TMDS Data2+ |
| 2 | TMDS Data2 Shield |
| 3 | TMDS Data2– |
| 4 | TMDS Data1+ |
| 5 | TMDS Data1 Shield |
| 6 | TMDS Data1– |
| 7 | TMDS Data0+ |
| 8 | TMDS Data0 Shield |
| 9 | TMDS Data0– |
| 10 | TMDS Clock+ |
| 11 | TMDS Clock Shield |
| 12 | TMDS Clock– |
| 13 | CEC |
| 14 | Reserved (N.C. on device) |
| 15 | SCL |
| 16 | SDA |
| 17 | DDC/CEC Ground |
| 18 | +5V Power |
| 19 | Hot Plug Detect |

H1-D2 : T.M.D.S DATA2+

H2-D3 : T.M.D.S DATA2屏蔽

H3-D1 : T.M.D.S DATA2-

H4-D10 : T.M.D.S DATA1+

H5-D11 : T.M.D.S DATA1屏蔽

H6-D9 : T.M.D.S DATA1-

H7-D18 : T.M.D.S DATA0+

H8-D19 : T.M.D.S DATA0屏蔽

H9-D17 : T.M.D.S DATA0-

H10-D23: T.M.D.S DATA CLOCK+

H11-D22: T.M.D.S DATA CLOCK屏蔽

H12-D24: T.M.D.S DATA CLOCK-

H13 :CEC (Consumer Electronics Control可选择的电子消费控制器)

H14 : Reserved (in cable but N.C. on device)

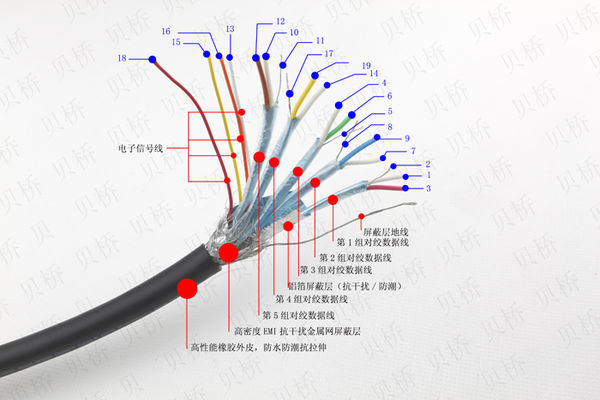
H15-D6 : SCL(DDC时钟线)

H16-D7 : SDA(DDC数据线)

H17-D15: DDC/CEC Ground

H18-D14: +5V电源线

H19-D16: 热插拔探测线

[](http://a.hiphotos.baidu.com/zhidao/pic/item/8718367adab44aeda6d20a14b51c8701a08bfba6.jpg" \o "点击查看大图" \t "_blank)

1. [↑](#footnote-ref-0)
2. [↑](#footnote-ref-1)
3. [↑](#footnote-ref-2)
4. [↑](#footnote-ref-3)
5. [↑](#footnote-ref-4)