USB

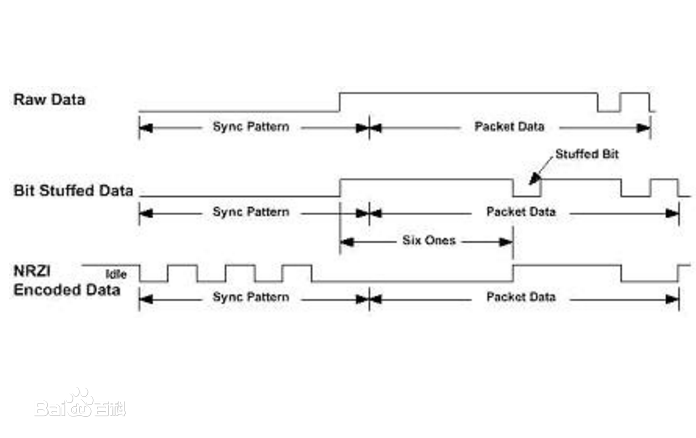
USB（Universal Serial Bus通用串行总线）使用一根屏蔽的4线电缆与设备互联，数据传输通过一个差分双绞线（D+白 D-绿）进行，另外两根线是VCC（红）和GND（黑）。



设备🡪主机，称上行连接 主机🡪设备，称下行连接

USB使用**反向不归零编码方式**（遇到“0”转换，遇到“1”保持），它可以保证数据传输的完整性，而且不要求传输过程中有独立的时钟信号。若重复相同的“1”信号一直进入，会导致数据长时间无法转换，使接收器丢失同步信号，使读取的时序发生严重错误。

因此USB协议规定：如果要发送的数据中出现有连续的6个1，则在进行NRZI编码前，在这6个连续的1后面会插入1个0，然后再进行NRZI编码。接收端收到连续6个1，将自动去掉后面的1个0。从而恢复原数据。



包（packet）是USB系统中信息传输的基本单位，所有数据都是经过打包后在总线上传输的。

包分为五部分：同步字段（SYNC）、包标识符字段（PID）、数据字段、循环冗余校验字段（CRC）和包结尾字段（EOP）。



1. **同步字段（SYNC）**

8位，作为每个数据封包的前导，用来产生同步作用，使USB设备与总线的包传输率同步，它的固定值为： 0000 0001。

1. **包标识符字段（PID）**

用来表示数据封包的类型。包标识符中的检验字段是通过对类型字段的每个位求反码产生的。





1. **数据字段（DATA）**

用来携带主机与设备之间要传递的信息，其内容和长度根据包标识符、传输类型的不同而不同。在USB包中，数据字段可以包含设备地址、端点号、帧序列号以及数据等内容。在总线传输中，总是首先传输字节的最低位，最后传输字节的最高位。

1. **设备地址（ADDR）数据域**

7位，最多可寻址127个外围设备。

1. **端点（ENDP）数据域**

4位，最多可寻址32个端点，仅作用在IN、OUT、SETUP令牌信息包中。慢速设备可支持端点0和端点1作为中断传输模式，而全速设备可拥有16个IN，16个OUT，总共32个端点。

1. **帧序列号**

USB令牌包的PID为SOF时，为11位。由主机产生，取值范围为：0到0x7FF，溢出归零。

1. **数据**

仅存于DATA信息包内，根据传输速度及传输类型不同，字节大小不同，0到1023字节（实时传输）。

1. **循环冗余校验字段（CRC）**

重要的数据信息包采用CRC16的数据域（16bit），而其余的信息包用CRC5（5bit）。

1. **包结尾字段（EOP）**

发送方在包的结尾发出包结束信号。USB主机根据EOP判断数据包的结束。

**信息包格式**

1. **令牌包（token）**

USB系统中，只有主机才能发出令牌包。它定义了数据传输的类型，它是事务处理的第一阶段。令牌包中重要的是SETUP、IN、OUT这三个令牌包。他们被用在根集线器和设备端点之间建立数据传输。



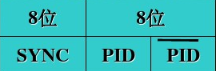
1. **数据包（data）**

DATA数据域的位值是根据USB设备的传输速度及传输类型决定的，须以8字节为基本单位，不足就补齐。



1. **握手包（handshake）**

仅包含一个PID数据域

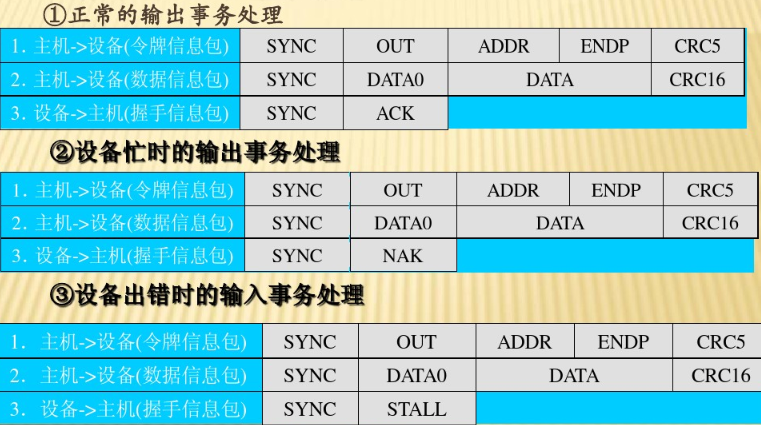


事务

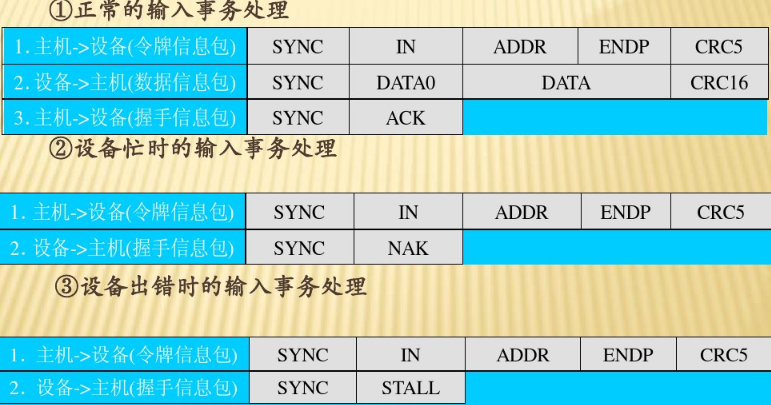
在USB上数据信息的一次接收或者发送的处理过程称为事务处理（Transaction）。它包括输入事务（IN）、输出事务（OUT）、设置事务（SETUP）、帧开始和帧结尾等类型。

OUT🡪DATA（DATA0、DATA1交替发送）🡪设备发送握手信号

if(idle)reutun ACK;if(busy)return NAK;if(error)return STALL

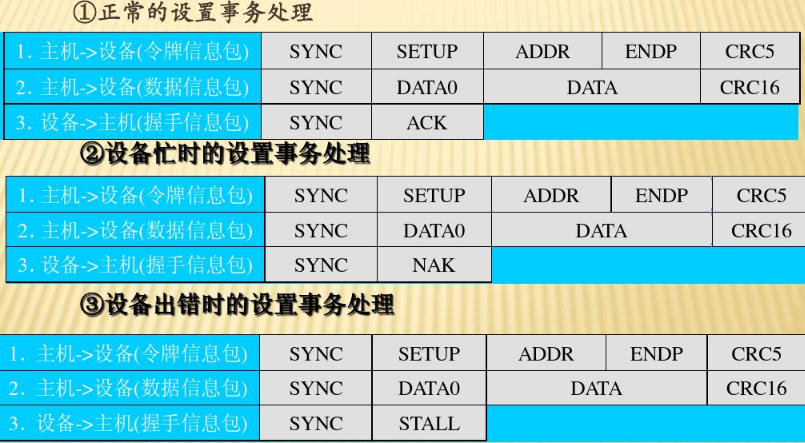


IN🡪设备发送DATA（if设备准备好发送数据，DATA0、DATA1也是交替发送）/NAK（if设备忙）/STALL（if设备出错）🡪主机发送握手包ACK（主机可以接收数据）/NAK（主机忙）/STALL（主机出错）



SETUP（过程和OUT相同）🡪DATA（DATA0、DATA1交替发送）🡪设备发送握手信号

if(idle)reutun ACK;if(busy)return NAK;if(error)return STALL



USB传输中，制定了4种传输类型：控制传输、实时传输、批量传输和中断传输。

控制传输：设置阶段、数据（传输）阶段（有/无）、状态阶段。

根据数据阶段传输方向，控制传输分：控制读取（读取USB描述符）、控制写入（配置USB设备）以及无数据控制。

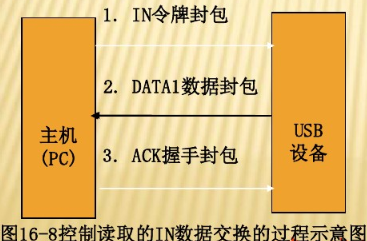
1. **控制传输**

**Stage One：设置阶段** SETUP🡪DATA0🡪ACK

**Stage Two：数据传输阶段**（如果有）

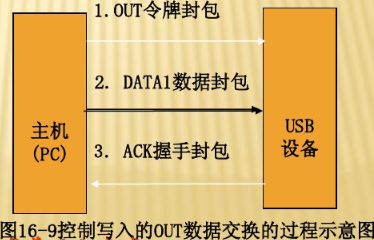
**控制读取** 设备🡪主机（读取数据USB设备描述符）

主机（发送IN令牌信息包）🡪设备将DATA1数据信息包回传给主机🡪主机发送ACK（正确接收）/发送NAK（主机忙）/发送STALL（发生错误）



**控制写入** 主机🡪设备 （对USB设备的配置信息）

主机（发送OUT令牌信息包）🡪主机将DATA0数据信息包传递给设备🡪主机发送ACK（正确接收）/发送NAK（设备忙）/发送STALL（发生错误）



**Stage Three：状态阶段** 表示整个传输过程已经完全结束

状态阶段传输的方向必须与数据阶段的方向相反（IN🡨🡪OUT …）。

1. **实时传输**

适用于必须以固定速率抵达或在指定时刻抵达，可以容忍错误的数据上。（麦克风、喇叭、etc.）

它只需要令牌和数据两个信息包，没有握手，所以出错不会重传。

1. **批量传输**

用于传输大量数据，要求不能出错，对时间没要求。（打印机、存储设备、etc.）

1. **中断传输**

总是对设备的查询，以确定是否有数据需要传输。因此中断传输的方向总是设备🡪主机

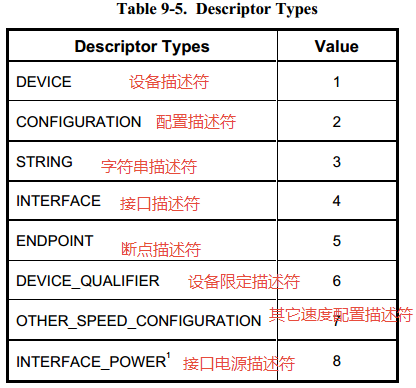
**USB协议传输过程总结：**

二进制数据串-->域（7种，同步域、标识域、地址域、端点域、帧号域、数据域、校验域）-->包（令牌包、数据包、握手包、特殊包）-->事务（输入事务、输出事务、设置事务）-->传输（控制传输、中断传输、实时传输、批量传输）

设备列举

1. **描述符**

USB外围设备的“身份证”。它的类型有：设备描述符、配置描述符、接口描述符和端点描述符等。



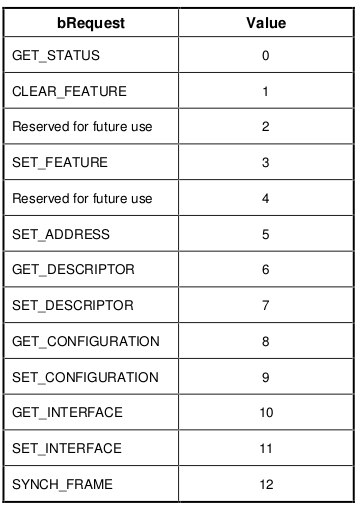
1. **USB设备请求**

几个主要的设备请求：清除特性（Clear Feature）、获得描述符（Get Descriptor）、设置地址（Set Address）、设置状态（Set Configuration）。

其帧格式为：

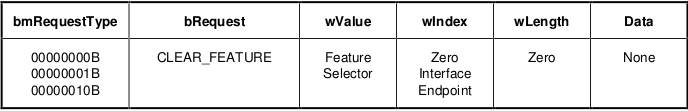


设备请求代码如下：



1. **清除特性**

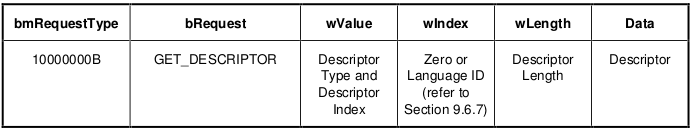
用来取消一个特性，格式如下：



wValue特性选择器，0=端点，1=设备。当某个特点不允许取消，或该特点不存在，或指向不存在之物，都会导致设备请求失败。如果端点被固件设为停止状态，主机软件可发送一个值为0的CLEAR\_FEATUR命令清除该端点的停止状态。

1. **获得描述符**

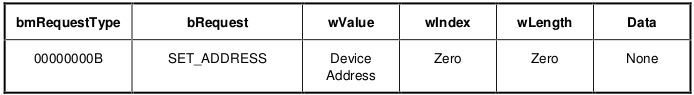
取得USB设备中存在的特定描述符，格式如下：



wValue高字节表示要取的描述符类型，低字节表示描述符的索引值，描述符类型上文有介绍。wIndex要取的描述符是字符串描述符时，该域的值为语言ID，其他的为0.

1. **设置地址**

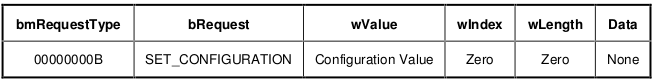
给设备设置地址，从而对该USB设备进行进一步的访问，格式如下：



该请求的USB设备直到该请求的状态阶段被成功完成才改变它的地址，而其他的请求在状态阶段之前就完成。若特定的设备地址大于127，或wIndex/wLength为非0值，该请求不执行。

1. **设置状态**

对设备进行设置，格式如下：



wValue域的低字节表示设置的值，为0或者与配置描述符中配置值相匹配。如果设置为0，则设备在地址状态。如果wIndex和wLength为非零值，则该请求不执行。

……

1. **设备列举**

主机通过USB设备请求来取得设备描述符并对该设备进行配置。

该过程的简化步骤如下：

step 1：使用预设的地址0取得设备描述符

step 2：设置设备新地址

step 3：使用新地址取得设备描述符

step 4：取得配置描述符

step 5：设置配置描述符

设备列举采用控制传输，134步用控制读取，25步用无数据控制。

USB功能模块

1. **USB收发器**

为USB的D+和D-数据提供了接口

1. **USB控制逻辑**

处理CPU和收发器之间的数据流动

1. **USB寄存器**

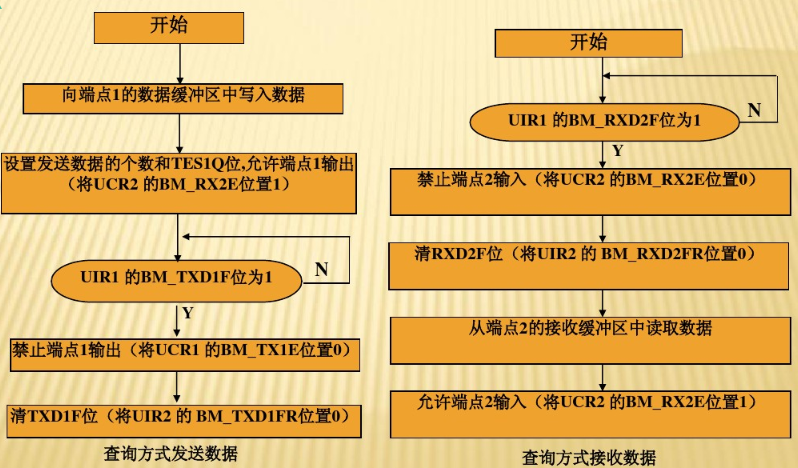
控制和监控USB操作

1. **USB复位信号**
2. **USB悬挂**
3. **USB低速设备**

USB寄存器有以下几种：地址寄存器（UADDR）、控制寄存器（UCR0-UCR4）、状态寄存器（USR0-USR1）、中断寄存器（UIR0-UIR2）、端点0数据寄存器（UE0D0-UE0D7）、端点1数据寄存器（UE1D0-UE1D7）和端点2数据寄存器（UE2D0-UE2D7）。

USB模块中断有三种类型：当接收或发送事务完成时，产生事务结束中断；当USB总线悬挂后被激活，产生唤醒中断；当检测到一个低速的包结束信号时，产生结束信号。

端点 1发送、接收数据流程图：



参考网址：<https://blog.csdn.net/linuxweiyh/article/details/78737224>

<https://wenku.baidu.com/view/172255d0ce2f0066f533222d.html>

<https://blog.csdn.net/g200407331/article/details/51682597/>

<https://www.cnblogs.com/flyheart33/p/3592248.html>