



USB资料收集整理

笔记本：KNOWLEDGE

创建时间：2014/12/10 8:58

更新时间：2015/5/29 10:14

URL：<http://zh.wikipedia.org/wiki/%E9%80%9A%E7%94%A8%E4%B8%B2%E8%A1%8C%E7...>

什么是USB

. USB的定义:

通用串行总线是连接计算机系统与外部设备的一种串口总线标准，也是一种输入输出接口的技术规范，被广泛地应用于个人电脑和移动设备等信息通讯产品，并扩展至摄影器材、数字电视（机顶盒）、游戏机等其它相关领域。

. 历史:

USB研发小组在1994年成立,成员有Compaq, DEC, IBM, Intel, Microsoft, NEC, Nortel七家公司.成立USB研发小组的目的是开发一种易于软件开发者和用户使用, 速度快捷, 能够与多种不同的设备连接的接口

USB1.0出现于1996年1月, USB1.0有两种速率模式:"Low Speed" - 1.5Mbit/s, "Full Speed" - 12Mbit/s.但是USB1.0并没有被广泛使用, USB真正被广泛使用的版本是USB1.1, 在1998年9月released的一个USB总线版本.但USB1.1在速率上和USB1.0没有太大差异, USB1.1主要是修正USB1.0发现的一些问题.

时间来到2000年4月, 此时USB的2.0版本诞生.在USB2.0中,添加了一个新的速率模式:"Hi-Speed" - 480Mbit/s.但是受到BOT传输协议和NRZI编码方式的限制, 实际最高的速度只有35MBytes/s = 280Mbits/s.

- USB2.0的其他特性:

- . Mini-A和Mini-B插座与插头标准的定义
- . 添加新的描述符以便将多重接口关联到单一设备中
- . 添加了OTG的定义, 允许两个USB设备不通过主机直接相互通讯
- . 添加了对充电器的支持
- . 定义了Micro-USB的接口和线缆
- . 在启用与待机间增加了新的电源模式。当设备处于这个模式时不向其发送指令以减少电源消耗。

所以，在启用及睡眠模式间切换要比在启用及待机模式间切换来的快得多。

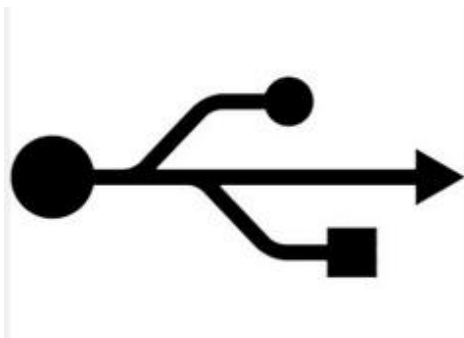
2008年11月, USB3.0发布, 定义了更高的速率模式5Gbps. USB3.0的插座是蓝色的, 并向后兼容USB2.0

2013年7月31, USB小组宣布USB3.1规格, 传输速率达到10Gb/s, USB3.1向下兼容之前所有版本的USB, 电力供应可达100W

. USB标志

. USB标准认证 LowSpeed和FullSpeed共用





. USB OTG

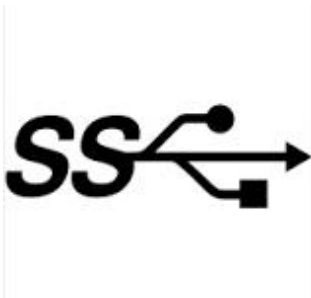


我可是高速的OTG哦~~~~~

. USB2.0高速模式 480Mbps



. USB SUPERSPEED USB3.0以上才有的速率模式, 5Gbps



. USB SUPERSPEED+ USB3.1的速率模式, 10Gbps



. 还有什么USB标志带闪电, 带加号啥的, 那是手提笔记本用的

. 接口:

. 标准型

. 标准USB只用四根线, 两个电源, 两个信号. USB四根线一般是这样分布的:

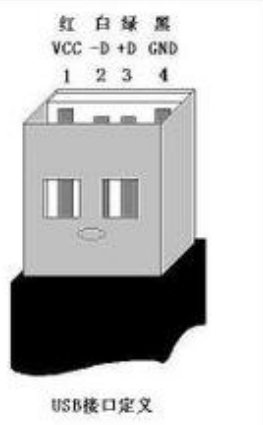
. 黑线 - GND

. 红线 - VCC

. 绿线 - data+

. 白线 - data-

排列方式 - 从左到右, 红白绿黑

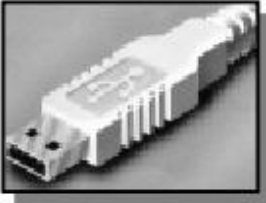

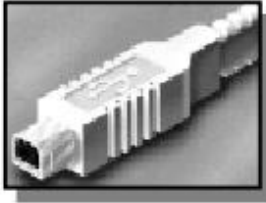



. A型,B型

. USB2.0的协议里面还规定了两种USB的Plug和Reception的类型

. 标准A系列: 这种扁平式插头插座应用最普遍, 主要应用于PC端或HUB连接。PC上就是这种A型插座。常见的U盘和USB电缆上用的就是这种A型插头。

. 标准B系列: 这种方形插头插座应用的要少一些, 主要应用在设备端连接。在一些打印机、数码伴侣等体积较大的设备上用的是B型插座, 相应的电缆用的是B型插头。

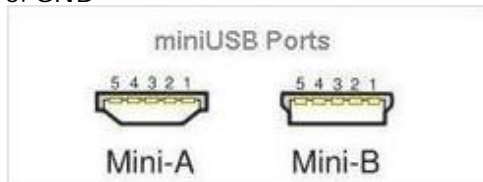
Series "A" Connectors	Series "B" Connectors
<p>◆ Series "A" plugs are always oriented upstream towards the <i>Host System</i></p>  <p>"A" Plugs (From the USB Device)</p> <p>"A" Receptacles (Downstream Output from the USB Host or Hub)</p> 	<p>◆ Series "B" plugs are always oriented downstream towards the <i>USB Device</i></p>  <p>"B" Plugs (From the Host System)</p> <p>"B" Receptacles (Upstream Input to the USB Device or Hub)</p>  <p>百合电子工作室USB专题站 usb.baiheee.com</p>

. mini USB

. mini USB也有分typeA,typeB,但是和标准型的typeA,typeB没多大关系

. mini USB有五根信号线:

1. VBUS
2. D-
3. D+
4. ID
5. GND



ID脚只有在OTG功能时候才使用

. 提起mini USB, 不能不提OTG,mini USB的一个引脚 - ID只有在OTG的模式下,才会被使用. 由于USB是主从结果, 两台USB设备之前不可能直接通信, 为了让各设备之间直接交换数据而不经PC中转, USB开发了OTG协议. 支持OTG的设备可以一定程度上模拟PC的功能, 控制USB总线完成与另一设备交换数据的服务.

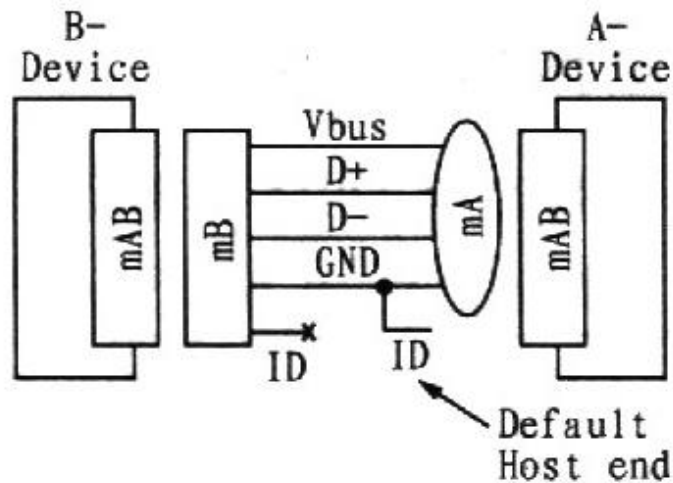
miniUSB分为miniA, miniB和miniAB三种接口, miniAB会根据ID引脚的状态判断接口类型, 高电平为B头, 此时系统默认做主;低电平为A头,系统使用HNP决定主从关系

HNR(Host Negotiation Protocol)决定连接时的主从关系, 注意主从关系在OTG中是可以切换的, 并不是说B口一定做从, A口一定做主,没有这样的说法

. ADP(Attach Dection Protocol)

. 让OTG device, embedded host, USB devices通过USB BUS上面的供电来判断设备是否连接上

. 周期性地测量USB Port 上面的电容变化,当电容变化大到足以表示有新设备接入USB Port, A-device会供电到USB bus



. micro USB

. 2007年1月4日，USB-IF颁布了Micro-USB的插头标准。该标准将在许多新型智能手机和PDA上替代Mini-USB。

Micro-USB插头的插拔寿命为10,000次，比Mini-USB插头高度减半，宽度相差无几。

microUSB和miniUSB差不多,也分a,b,ab三种,同样支持OTG

Pin	名称	线的颜色	描述
1	VBUS	Red (红)	电源正5V
2	D-	White (白)	数据线负
3	D+	Green (绿)	数据线正
4	ID	none (无)	分为A和B两种接口 A: 与地线相连 B: 不与地线相连
5	GND	Black (黑)	信号地线

. TYPE C

. 基于USB 3.1规范全新设计的USB Type-C，外观上最大特点在于其上下端完全一致，这意味着用户不必再区分USB正反面

. TYPE C不再是传统的四根数据线,而是二十四根!!而且Plug和Reception有点不同

TYPE-C接口的线序：

被插入端，好邪恶的名字：

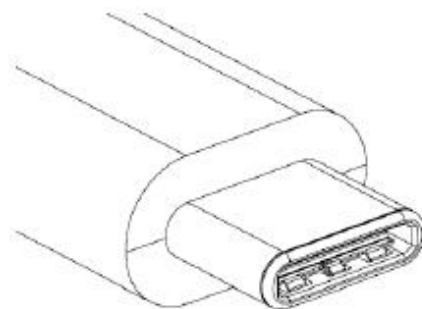
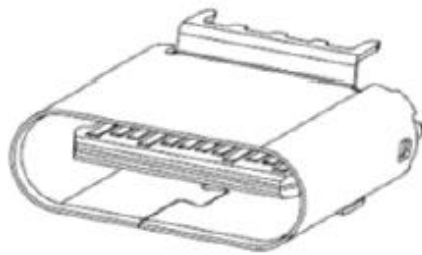
Figure 2-1 USB Type-C Receptacle Interface (Front View)

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
GND	TX1+	TX1-	VBUS	CC1	D+	D-	SBU1	VBUS	RX2-	RX2+	GND
GND	RX1+	RX1-	VBUS	SBU2	D-	D+	CC2	VBUS	TX2-	TX2+	GND
B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1

插入端：

Figure 2-2 USB Full-Featured Type-C Plug Interface (Front View)

A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1
GND	RX2+	RX2-	VBUS	SBU1	D-	D+	CC	VBUS	TX1-	TX1+	GND
GND	TX2+	TX2-	VBUS	VCONN			SBU2	VBUS	RX1-	RX1+	GND
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12



· USB Type-C 的其他特性如下：

- 正确插入后发出声音。
- 通过USB Power Delivery技术，可用于3C产品充电。
- 增进的电磁干扰与RFI mitigation特性
- 支持1万次拔插。



. USB Type-C支持Display, 与Thunder看齐

. USB主机类型

. OHCI UHCI EHCI xHCI

. OHCI UHCI

. 两种都是对应usb1.1标准的

. OHCI, Open Host Controller Interface, 创立者是Compaq, Microsoft和National Semiconductor

. UHCI, Universal Host Controller Interface, 创立者是Intel

. 两者区别

. OHCI在功能上, 把更多要做的事情放在硬件上面

. UHCI则把更多要做的事情放在软件上面

. 由于上面的原因, OHCI节省CPU资源更适合在嵌入式设备中使用, UHCI造价便宜, 适合Intel推广自己的平台

. 技术细节

. 单帧内的stage的个数

. OHCI在单个帧内, 可以调度多个stage

. UHCI在在单个帧内, 只调度一个stage

. 单帧内的transaction的个数

. OHCI单个帧内, 可能会有多个transaction

. UHCI单个帧内, 只有一个transaction

. 轮询的频率

. OHCI即使端点描述符中, 已经指定了最大延迟是255ms, 但是OHCI主控还是会至少每32ms就去轮询一次中断端点

. UHCI主控可以支持, 但不是必须要支持, 更低频率地轮询

. EHCI

. EHCI, Enhanced Host Controller Interface

. EHCI定义了USB 2.0的主机控制器的规范, 定义了USB 2.0的主控, 需要包括哪些硬件实现, 需要实现哪些功能, 其也对应着对应的系统软件, 所面对的是哪些接口。

. xHCI

. xHCI, Extensible Host Controller Interface

. xHCI是针对的USB 3.0规范。也是定义了USB 3.0主控需要如何实现, 需要包含哪些功能, 也提供了寄存器级别的定义。

USB主机控制器类型	共同点	区别			
		对应的USB的协议和支持的速率	创立者	功能划分	常用于
OHCI	都实现了对应的USB的规范中所要求的功能	USB 1.1=Low Speed和Full Speed	Compaq, Microsoft和National Semiconductor	硬件功能 > 软件功能=硬件做的事情更多, 所以实现对应的软件驱动的任务, 就相对较简单	扩展卡, 嵌入式开发板的USB主控
UHCI			Intel	软件功能 > 硬件功能=软件的任务重, 可以使用较便宜的硬件的USB控制器	PC端的主板上的USB主控
EHCI		USB 2.0=High Speed	Intel	定义了USB 2.0主控中所要实现何种功能, 以及如何实现	各种USB 2.0主控
xHCI		USB 3.0=Super Speed	Intel	定义了USB 3.0主控中所要实现何种功能, 以及如何实现	各种USB 3.0主控

USB总线

. USB总线拓扑

. USB System Bus Topology

USB总线拓扑是星型结构, 但又不是简单的星型,而是"tired-star", 即层叠式的星状结构。

因为线缆和时序参数的问题,USB系统最大的层数仅为7

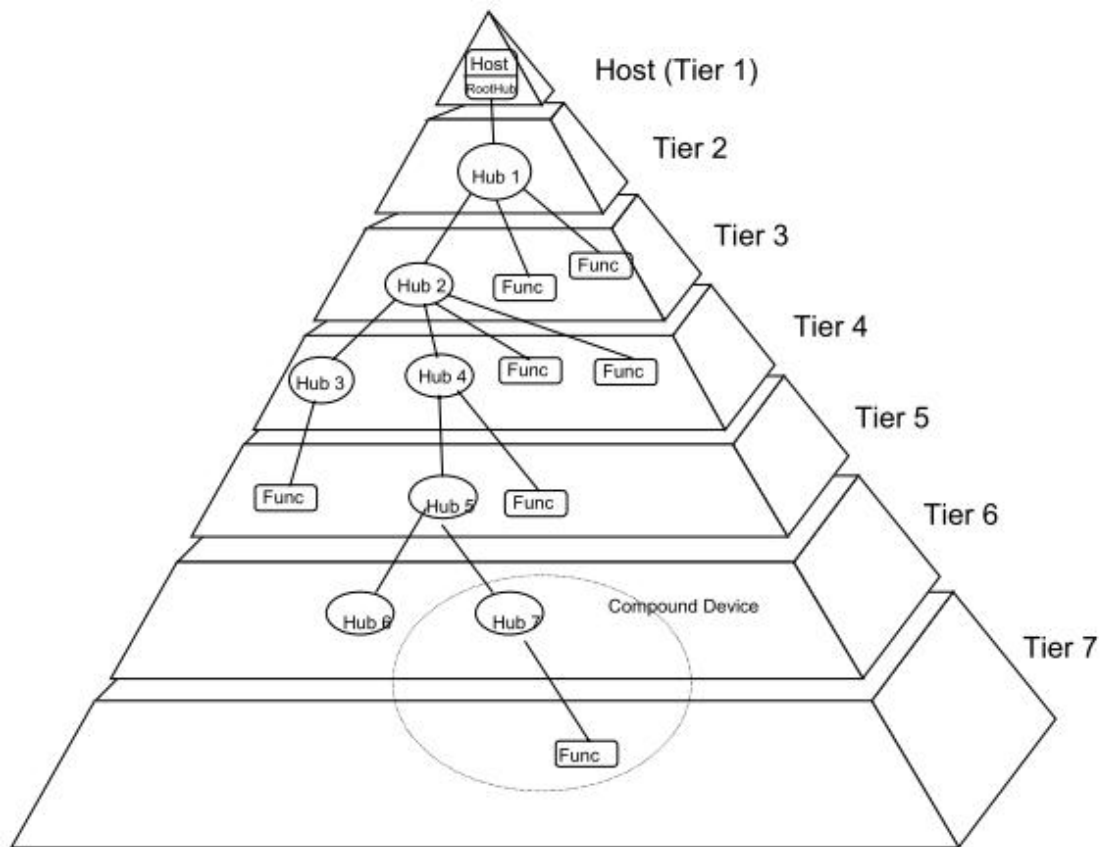
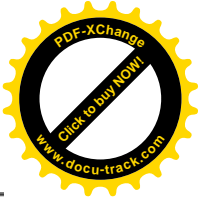


Figure 4-1. Bus Topology

- . USB系统中定义了三种角色:
- . USB interconnect
- . USB device
- . USB Device Class



Base Class	Descriptor Usage	Description
00h	Device	Use class information in the Interface Descriptors 种类信息定义在接口描述符中
01h	Interface	Audio 音频设备
02h	Both	Communications & CDC 通信设备 (手机, Class_02&SubClass_02&Prot_01)
03h	Interface	HID (Human Interface Device) 人机接口设备
05h	Interface	Physical 物理设备
06h	Interface	Image 图像设备 (可能是iPhone手机, Class_06&SubClass_01&Prot_01)
07h	Interface	Printer 打印机
08h	Interface	Mass Storage 大容量存储 (可能是手机, Class_08&SubClass_06&Prot_50)
09h	Device	Hub 集线器
0Ah	Interface	CDC-Data 通信设备 (手机, Class_0A&SubClass_00&Prot_00)
0Bh	Interface	Smart Card 智能卡
0Dh	Interface	Content Security 内容安全设备
0Eh	Interface	Video 视频设备 (摄像头, Class_0e&SubClass_03&Prot_00)
0Fh	Interface	Personal Healthcare 个人健康设备
10h	Interface	Audio/Video Devices 音频/视频设备
DCh	Both	Diagnostic Device 诊断设备 (USB2兼容设备)
E0h	Interface	Wireless Controller 无线控制器 (蓝牙设备等)
EFh	Both	Miscellaneous 杂项 (ActiveSync, PalmSync, 各种协会等)
FEh	Interface	Application Specific 应用专有规范 (固件升级, 红外, USB测试与测量等)
FFh	Both	Vendor Specific 供应商自定义规范 (手机, Class_FF&SubClass_FF&Prot_FF)

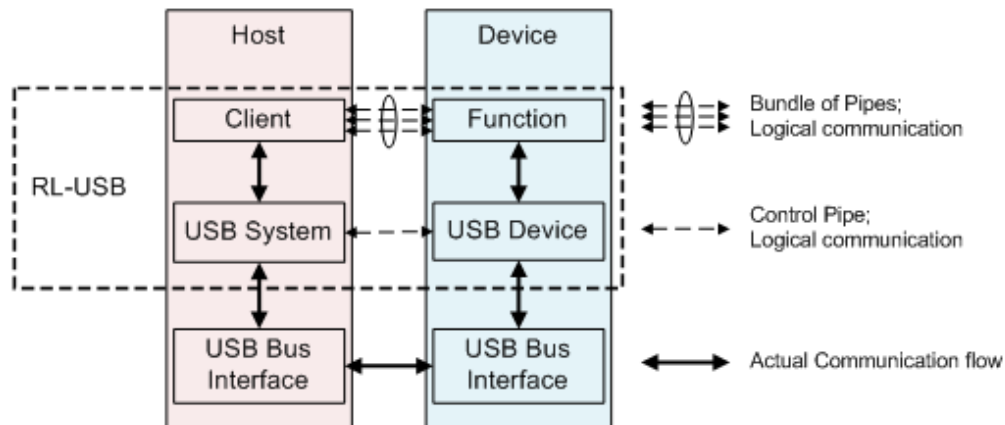
. USB HOST

USB通讯

. USB通讯模型

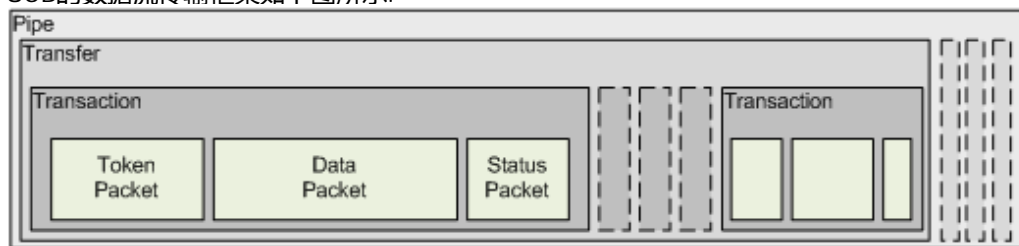
- . USB是一种Polled Bus, 由Host启动数据传输, Device做出回应
- . USB Physical Device(USB设备)
 - . USB bus interface
 - . 提供与HOST的物理连接
 - . USB logic Device
 - . 对应HOST的USB System SW, 为HOST提供basic interface
 - . Function
 - . 对应HOST的Client Software, 提供特定的配置信息给HOST
- . USB Host
 - . Client Software(客户端软件)
 - . HOST上面运行的一段对应USB设备的应用程序
 - . USB系统软件(USB System Software)
 - . 在特定操作系统上面支持USB运行的软件

- . USB HOST 控制器(USB Host Controller)
- . HOST 端USB接口



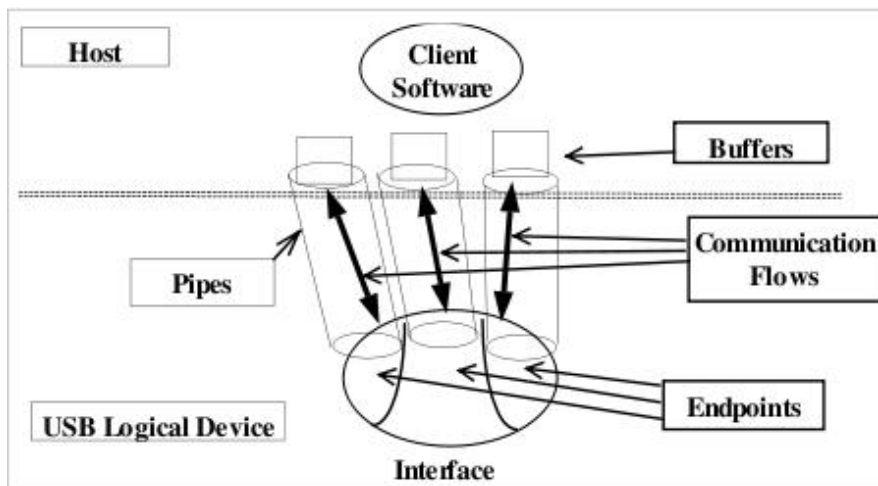
- . USB数据流
- . 传输框架

USB的数据流传输框架如下图所示:

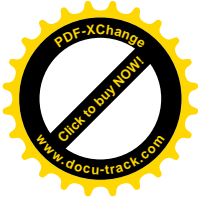


- . 管道

- . 对主机和usb设备间通信流的抽象
- . 一个USB管道是驱动程序的一个数据缓冲区与一个外设端点的连接
- . 管道有两种类型
 - . 数据流管道: 可以被Host或者Device控制, 数据的传输方向必须事先定义.支持下面三种传输:
 - . Bulk Transfers,
 - . Isochronous Transfers,
 - . Interrupt Transfers.
 数据流管道中的数据都不带USB协议定义的数据结构
 - . 消息管道: 只能被Host控制, 允许双向传输. 只支持Control Transfers,
 - . 消息管道中的数据带有USB协议定义的数据结构



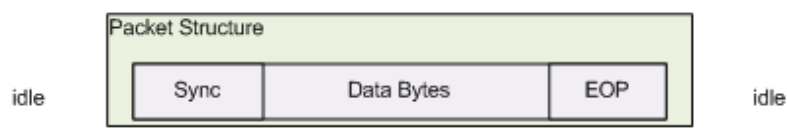
- . endpoint
 - . endpoint是设备端的一个抽象概念,用以表示通讯流在设备端的终点.
 - . USB logic device 由大量的endpoint组成
 - . 每个endpoint都有一个唯一的endpoint number.
 - . interface是endpoint的集合,一个interface上不同的endpoint有不同的用途
 - . Endpoint Zero
 - . 每个USB设备上都有一个默认的双工控制端口, 在这个端口上, 可以实现一些默认的控制方法. 这个端口,通用的叫法是endpoint zero.
 - . endpoint zero是唯一一个双工端口
- . transfer
 - . Control Transfer
 - . 用于在设备连接时对设备的配置, 以及其他的控制命令及状态操作
 - . Bulk Data Transfer
 - . 用于大批量数据传输, 并确保数据无误的传输. 带CEC校正, 并且校正不成功会进行重传
 - . Isochronous Data Transfer
 - . 用于大批量数据传输, 但不需要确保数据无误的传输. 不带CEC, 也不重传
 - . Interrupt Data Transfer
 - . 用一个固定的速率传输少量数据. 与通常意义上的中断不同, 需要host轮询来执行
- . transaction
 - . transaction 指USB的数据传输, 大部分的传输只涉及到四种packet中的三种:



PID Type	PID Name	PID<3:0>*	Description
Token	OUT	0001B	Address + endpoint number in host-to-function transaction
	IN	1001B	Address + endpoint number in function-to-host transaction
	SOF	0101B	Start-of-Frame marker and frame number
	SETUP	1101B	Address + endpoint number in host-to-function transaction for SETUP to a control pipe
Data	DATA0	0011B	Data packet PID even
	DATA1	1011B	Data packet PID odd
	DATA2	0111B	Data packet PID high-speed, high bandwidth isochronous transaction in a microframe (see Section 5.9.2 for more information)
	MDATA	1111B	Data packet PID high-speed for split and high bandwidth isochronous transactions (see Sections 5.9.2, 11.20, and 11.21 for more information)
Handshake	ACK	0010B	Receiver accepts error-free data packet
	NAK	1010B	Receiving device cannot accept data or transmitting device cannot send data
	STALL	1110B	Endpoint is halted or a control pipe request is not supported
	NYET	0110B	No response yet from receiver (see Sections 8.5.1 and 11.17-11.21)
Special	PRE	1100B	(Token) Host-issued preamble. Enables downstream bus traffic to low-speed devices.
	ERR	1100B	(Handshake) Split Transaction Error Handshake (reuses PRE value)
	SPLIT	1000B	(Token) High-speed Split Transaction Token (see Section 8.4.2)
	PING	0100B	(Token) High-speed flow control probe for a bulk/control endpoint (see Section 8.5.1)
	Reserved	0000B	Reserved PID

. Packet

Packet是每次封包的最小单位



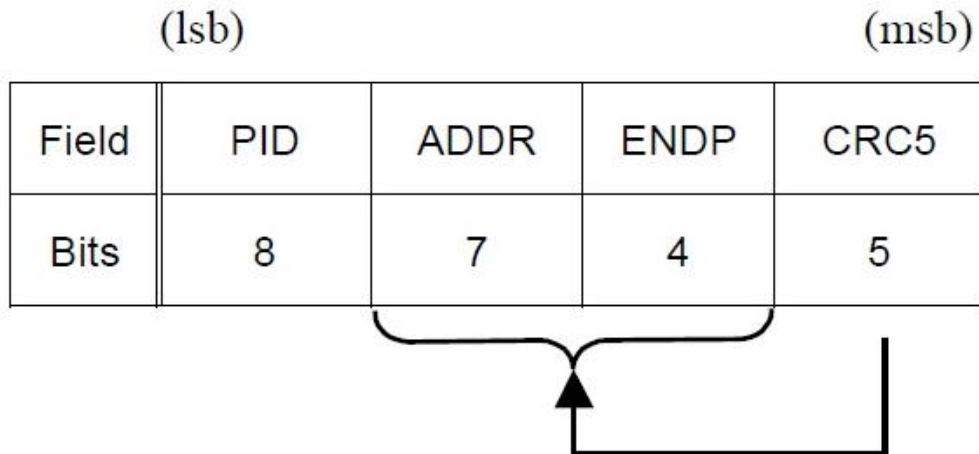
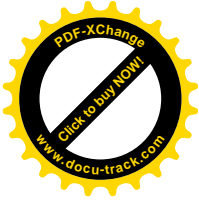
Packet以Synchronization Pattern为开头

. Token packet

. 每个Transaction以Token Packet 为起始

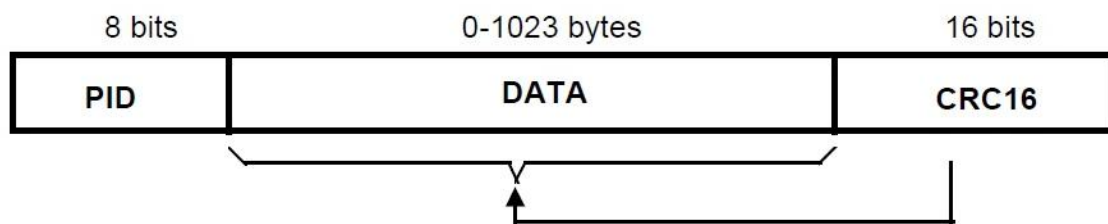
. Token Packet定义装置, endpoint数量及传输方向

. 用PID识别packet处理内容:OUT, IN, SOF, SETUP



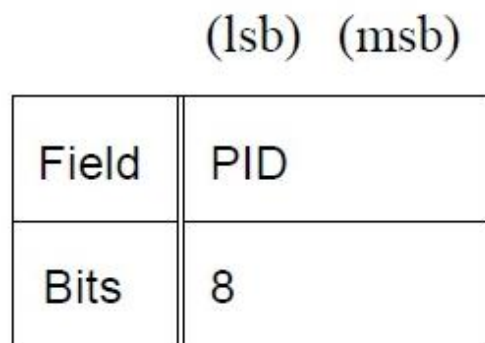
Token Format

- . Data packet
- . Data packet包含处理这一动作的数据, 在low-speed中, data最大数据量为8bytes.在full-speed中, data最大数据量为1023bytes.在full-speed中, data最大数据量为1024 bytes以上



Data Packet Format

- . Handshake or Status packet
- . Handshake Packet用来表示数据传输的状态
- . ACK表示数据传输无误
- . NAK表示设备无法从主机接收到资料或者没有资料可以传输到主机
- . STALL表示设备无法传送或接收到数据, 需要主机介入



- . Special packet(这种类型只出现在主机和full-/low- speed设备的数据通讯中)
- . Split Transactions
 - . Split Transaction只用在一個场景:full-speed或low-speed的hub挂到high-speed的HOST Controller上.
 - . Split Transaction分两种: start-split transaction, complete-split transaction



- . Split Transaction不像一般的Transaction只需要一个Token, 在Split Transaction里面需要带有两个Token: SSPLIT/CSPLIT中的一个, FS/LS Token

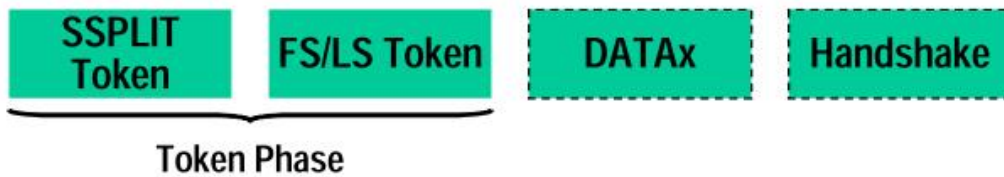
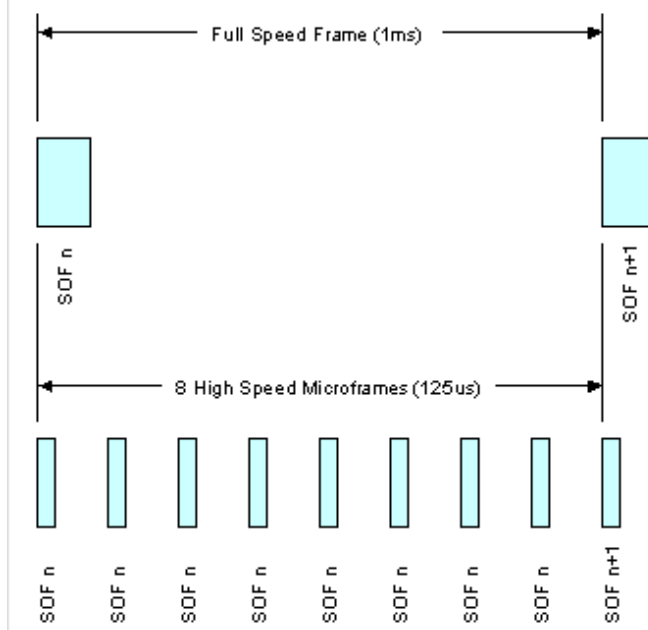


Figure 8-6. Packets in a Start-split Transaction



Figure 8-7. Packets in a Complete-split Transaction

- . Transaction可能是从Host传向Device, 或是从Device传向Host
 - . 传送方向是由Token packet中指定
 - . 一般来说, 目标端利用Handshake (Status packet) 来判断本次传输是否成功
- . Frame
- . 为了确保同步, USB把时间分成固定长度的小块, low-/full- speed以1ms为单位, 称为1 frame.
 - high-speed以0.125ms为单位称为1 micro frame



USB协议

- . USB描述符

USB描述符分为以下几种：



. 设备描述符

设备描述符给出了设备的一般信息, 包括制造商ID, 产品序列号, 所属设备类号, 默认端点的最大包长和配置描述符个数等. USB协议中规定, 每个USB设备必须有一个且只能有一个设备描述符.

. 配置描述符

配置描述符描述的是一个USB设备中的一个设备配置, 比如说, 一个手机插入到一台电脑中, 不仅仅有存储设备的功能, 还有照相机和调试端口. 对于HOST来说, 手机是一个设备, 但是同时会被抽象为三个配置, 分别为存储, 照相机和调试功能. 一个设备描述符中会有一个或多个配置描述符, 配置描述符中带有供电方式和最大耗电量. HOST发出USB标准命令Get_Descriptor获得设备描述符的时候, 设备会将这个配置描述符中的接口描述符和端点描述符都发给主机

. 接口描述符

接口描述符描述的是USB设备提供的某一个功能, 如果一个USB设备即可录音又可收音, 那这个设备就有两个接口描述符

. 端点描述符

端点是设备和主机数据交换的逻辑接口, 端点描述符描述了端点的传输类型, 传输方向, 数据包大小和端口地址.

端点0没有专门的端口描述符.

. 字符描述符

字符描述是一个可选的描述符, 描述制造商

详细的指令和资料参考下面网址

<http://www.baiheee.com/Documents/090518/090518112619.htm>

. USB设备状态机

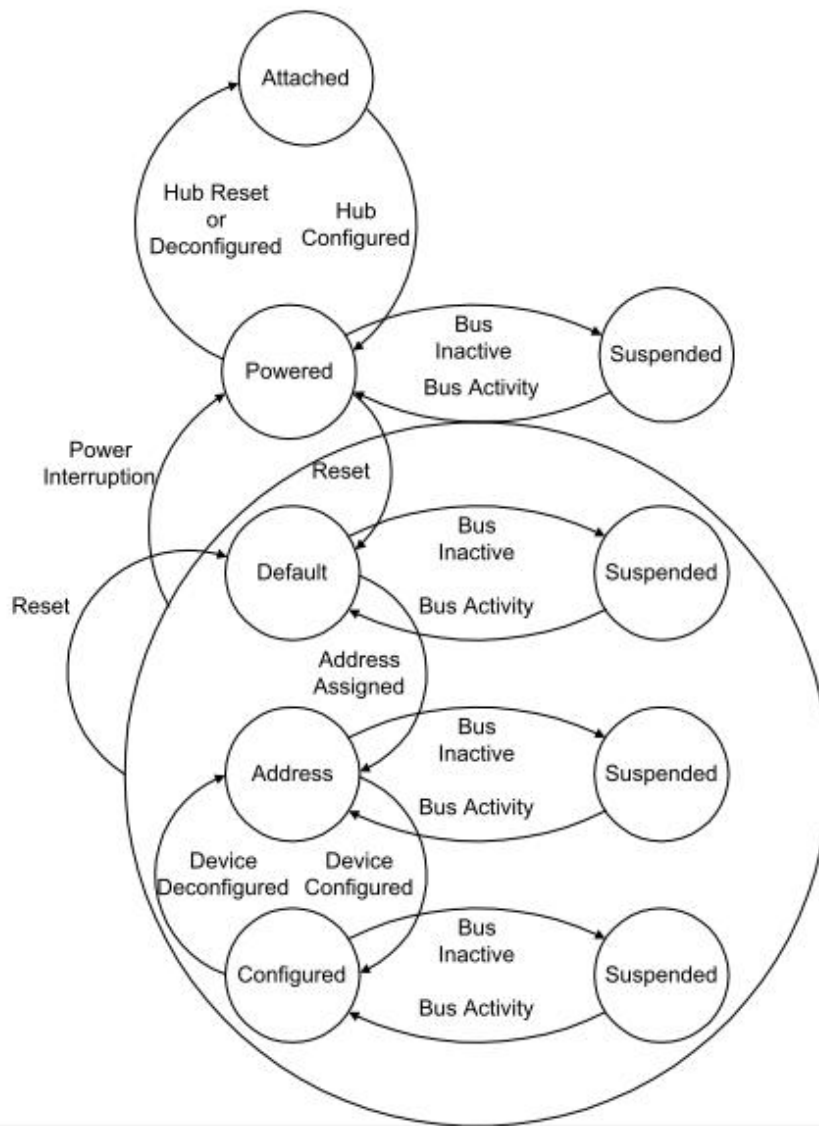




Table 9-1. Visible Device States

Attached	Powered	Default	Address	Configured	Suspended	State
No	--	--	--	--	--	Device is not attached to the USB. Other attributes are not significant.
Yes	No	--	--	--	--	Device is attached to the USB, but is not powered. Other attributes are not significant.
Yes	Yes	No	--	--	--	Device is attached to the USB and powered, but has not been reset.
Yes	Yes	Yes	No	--	--	Device is attached to the USB and powered and has been reset, but has not been assigned a unique address. Device responds at the default address.
Yes	Yes	Yes	Yes	No	--	Device is attached to the USB, powered, has been reset, and a unique device address has been assigned. Device is not configured.
Yes	Yes	Yes	Yes	Yes	No	Device is attached to the USB, powered, has been reset, has a unique address, is configured, and is not suspended. The host may now use the function provided by the device.
Yes	Yes	--	--	--	Yes	Device is, at minimum, attached to the USB and is powered and has not seen bus activity for 3 ms. It may also have a unique address and be configured for use. However, because the device is suspended, the host may not use the device's function.

. USB枚举过程

- . 设备插入USB端口， 或者上电
- . HUB检测到数据线的电压被拉低
 - . 在HUB端，数据线D+和D-都有一个阻值在14.25k到24.8k的下拉电阻Rpd，而在设备端，D+（全速，高速）和D-（低速）上有一个1.5k的上拉电阻Rpu。当设备插入到hub端口时，有上拉电阻的一根数据线被拉高到幅值的90%的电压（大致是3V）。hub检测到它的一根数据线是高电平，就认为是有设备插入，并能根据是D+还是D-被拉高来判断到底是什么设备。
 - . 检测到设备后，hub继续给设备供电，但并不急于与设备进行USB传输。
- . HUB利用自己的中断端点向HOST报告它各个端口的状态
- . HOST发送Get_Port_Status请求(request)给HUB以了解此次状态改变的确切含义
- . HUB通过检测USB总线空闲(Idle)时差分线的高低电压来判断所连接设备的速度类型，当HOST发来Get_Port_Status请求时，HUB就可以将此设备的速度类型信息回复给host。USB 2.0规范要求速度检测要先于复位（Reset）操作。
- . HUB复位设备
 - . 主机一旦得知新设备已连上以后，它至少等待100ms以使得插入操作的完成以及设备电源稳定工作。

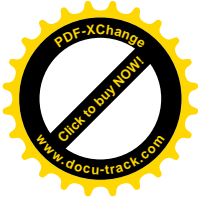


然后HOST就向HUB发出一个 Set_Port_Feature请求让hub复位其管理的端口(刚才设备插上的端口)。HUB通过驱动数据线到复位状态(D+和D-全为低电平),并持续至少10ms。当然,HUB不会把这样的复位信号发送给其他已有设备连接的端口,所以其他连在该hub上的设备自然看不到复位信号,不受影响。

- . HOST检测所连接的全速设备是否是支持高速模式
 - . HUB建立设备和主机之间的信息通道
 - . 主机不停地向hub发送Get_Port_Status请求,以查询设备是否复位成功。HUB返回的报告信息中有专门的一位用来标志设备的复位状态。
当HUB撤销了复位信号,设备就处于默认/空闲状态(Default state),准备接收主机发来的请求。设备和主机之间的通信通过控制传输,默认地址0,端点号0进行。此时,设备能从总线上得到的最大电流是100mA。(所有的USB设备在总线复位后其地址都为0,这样主机就可以跟那些刚刚插入的设备通过地址0通信。)
 - . 主机发送Get_Descriptor请求获取默认管道的最大包长度
 - . HOST通过地址0,端点0发送请求命令,由于枚举过程不是多个设备并行处理,而是一次枚举一个设备的方式进行,所以不会发生多个设备同时响应主机发来的请求。
设备接受到请求后,向HOST发送设备描述符,这个时候主机只关注描述符的长度。获得这个信息后HOST要求HUB再对DEVICE发起一次复位,以保障设备进入一个确定的状态
 - . HOST向设备分配一个地址
 - . HOST通过Set_Address向DEVICE分配唯一的一个地址,完成这次传输,DEVICE进入到地址状态。
 - . HOST获取DEVICE信息
 - . HOST通过新地址读取设备描述符与配置描述符,根据配置描述符中的配置集合总长度,获取配置描述符,接口描述符,以及字符串设备
 - . HOST挂载设备的驱动
 - . 设备驱动选择一个配置
 - . 设备驱动发送Set_Configuration来确定设备选择的配置
- 详细的资料与参考USB Specification的9.12

. USB HID 及点击事件上报

- . HID
- . Hummer Interface Device, 包括鼠标, 键盘, 摇杆等
- . HID管道
- . 控制管道: 传输USB描述符, 类请求代码及消息数据
- . 中断输入: 传输从设备到主机的输入
- . 中断输出: 传输从主机到设备的输出
- . HID描述符
- . 除了五个标准描述符外, HID有三个特定的描述符:
 - . HID描述符
 - . 描述HID规范版本号, HID通讯所用的额外描述符
 - . 报告描述符
 - . 报表描述符定义了执行设备功能的数据格式和使用方法。
报表描述符和USB的其他描述符是不一样的,它不是一个简单的表格,报表描述符是USB所有描述符中最复杂的。报表描述符非常复杂而有弹性,因为它需要处理各种用途的设备。报表的数据必须以简洁的格式来储存,这样才不会浪费设备内的储存空间以及数据传输时的总线时间。报表描述符必须先描述数据的大小与内容。报表描述符的内容与大小因设备的不同而不同,在进行报表传输之前,主机必须先请求设备的报表描述符,只有得到了报表描述符才可正确解析报表的数据。
报表描述符由项目组成,项目由三部分组成:项目类型,项目标志,项目长度
报表项目分为三类:
 - . Main类项目用于定义报表描述符中的数据项
 - . Global类项目实现对数据的描述,用来识别报表并且描述报表内的数据



. Local类项目定义控制的特征

项目类型	项目标志 (Tag)	项目前缀, nn 为数据长度	功能说明
Main 类项目	Input	1000 00 nn	定义输入报表, 主机利用该信息解析设备提供的数 据。主机向控制端口发送Get_Report实现输入
	Output	1001 00 nn	创建输出报表, 通过向设备发送 Set_Report 实现输 出
	Feature	1011 00 nn	定义送往设备的设置信息
	Collection	1010 00 nn	定义 2 个以上数据 (Input、Output 和 Feature) 的 关系为集合, Collection 开始一个集合, 之后的 End
	End Collection	1100 00 nn	Collection 结束集合。Collection 项目的数据部分说明 Collection 的类型
Global 类项目	Usage Page	0000 01 nn	指定设备的功能 另外由于Usage项目有32位数据值, Usage Page 项目用于为Usage项目在报表描述符中占居存储 空间。用于存放后续的Usage项目的高16位。
	Logical Minimum	0001 01 nn	定义变量或数组项目的逻辑最小值和最大值
	Logical Maximum	0010 01 nn	定义变量或数组项目的物理最小值和最大值, 分别 和 Logical Minimum、Logical Maximum 对应
	Physical Minimum	0011 01 nn	定义变量或数组项目的物理最小值和最大值, 分别 和 Logical Minimum、Logical Maximum 对应
	Physical Maximum	0100 01 nn	定义数值是基于 10 的指数
	Unit Exponent	0101 01 nn	单位
	Unit	0110 01 nn	指定报表数据区域所包含的位数
	Report Size	0111 01 nn	报表 ID, 该项目在报表中插入一个字节的报表 ID
	Report ID	1000 01 nn	报表中数据域的数目
	Report Count	1001 01 nn	将 Global 项目状态表送入堆栈
	Push	1010 01 nn	从堆栈恢复 Global 项目状态表
	Pop	1011 01 nn	保留
		1100 01 nn – 1111 01 nn	
Local 类项目	Usage	0000 10 nn	用法索引值, 表示对项目或集合建议的用法, 用于 当一个项目描述多个控制, 对每一个变量和数组元 素都有建议的用法
	Usage Minimum	0001 10 nn	定义阵列或位图中控制操作的第一个和最后一个用 法
	Usage Maximum	0010 10 nn	确定用于控制的实体, 指向物理描述符中的目标
	Designator Index	0011 10 nn	定义阵列或位图目标的起始和终止索引值
	Designator Minimum	0100 10 nn	确定字符串描述符中的索引值
	Designator Maximum	0101 10 nn	定义用于阵列或位图控制中字符串序列索引值的 最小值和最大值
	String Index	0111 10 nn	定义一组 Local 项目的开始和结束, 1=开始, 0=结 束
	String Minimum	1000 10 nn	保留
	String Maximum	1001 10 nn	
	Delimiter	1010 10 nn	
		1010 10 nn – 1111 10 nn	

. 实体描述符

. 实体描述符被用来描述设备的行为特性

. HID事件上报过程

. HID的事件上报是在中断传输中上报的, 这里的中断和一般意义上的中断并不完全相同, 主要由两部分组成:

. USB HOST CONTROL 轮询USB设备, 查询设备是否有中断上报

. 中断传输发生的时候, USB CONTROL 通过中断上报给CPU



. HID以报告的方式， 通过来给CPU发送数据

. USB Mass Stroage 数据传输过程

. USB Mass Stroage有两种传输协议

. Control/Bulk/Interrupt传输

. Bulk-only传输

. 支持BOT的设备必须支持最少3个endpoint：Control, Bulk-In和Bulk-Out。USB2.0的规范定义了控制端点0。Bulk-In端点用来从设备向主机传送数据。Bulk-Out端点用来从主机向设备传送数据。在仅批量数据传输协议中规定，数据传输分为三个阶段：命令阶段、数据阶段和状态阶段。命令阶段是由主机通过批量端点发送一个CBW（命令封装包）的结构，在CBW中定义了要操作的命令以及传输数据的方向和数量，数据阶段的传输方向由命令阶段决定，而状态阶段则总是由设备返回该命令完成的状态。

