# USB命令

在USB规范里，对命令一词提供的单词为“Request”，但这里为了更好的理解主机与设备之间的主从关系，将它定义成“命令”。

　　所有的USB设备都要求对主机发给自己的控制命令作出响应，USB规范定义了11个标准命令，它们分别是：Clear\_Feature、Get\_Configuration、Get\_Descriptor、Get\_Interface、Get\_Status、Set\_Address、Set\_Configuration、Set\_Descriptor、Set\_Interface、Set\_Feature、Synch\_Frame。所有USB设备都必须支持这些命令（个别命令除外，如Set\_Descriptor、Synch\_Frame）。

　　不同的命令虽然有不同的数据和使用目的，但所有的USB命令结构是一样的。下表所示为USB命令的结构：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表1、USB命令的结构** | | | | |
| 偏移量 | 域 | 长度（字节） | 值 | 描述 |
| 0 | bmRequestType | 1 | 位图 | 请求特征： D7：传输方向  0=主机至设备  1=设备至主机  D6..5：种类  0=标准  1=类  2=厂商  3=保留  D4..0：接受者  0=设备  1=接口  2=端点  3=其他  4..31 保留 |
| 1 | bRequest | 1 | 值 | 命令类型编码值（见表3） |
| 2 | wValue | 2 | 值 | 根据不同的命令，含义也不同 |
| 4 | wIndex | 2 | 索引或偏移 | 根据不同的命令，含义也不同，主要用于传送索引或偏　移 |
| 6 | wLength | 2 |  | 如有数据传送阶段，此为数据字节数。 |

下表列出了USB的11种标准命令

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表2、USB的11种标准命令** | | | | | | |
| 命令 | bmRequestType | bRequest | wValue | wIndex | wLength | Data |
| Clear\_Feature | 00000000B 00000001B 00000010B | CLEAR\_FEATURE | 特性选择符 | 零  接口号  端点号 | 零 | 无 |
| Get\_Configuration | 10000000B | GET\_CONFIGURATION | 零 | 零 | 一 | 配置值 |
| Get\_Descriptor | 10000000B | GET\_DESCRIPTOR | 描述表种类(高字节，见表5）和索引（低字节） | 零或语言标志 | 描述表长 | 描述表 |
| Get\_Interface | 10000001B | GET\_INTERFACE | 零 | 接口号 | 一 | 可选设置 |
| Get\_Status | 10000000B 10000001B 10000010B | GET\_STATUS | 零 | 零（返回设备状态） 接口号（对像时接口时） 端点号（对象是端点时） | 二 | 设备, 接口 ,或  端点状态 |
| Set\_Address | 00000000B | SET\_ADDRESS | 设备地址 | 零 | 零 | 无 |
| Set\_Configuration | 00000000B | SET\_CONFIGURATION | 配置值（高字节为0，低字节表示要设置的配置值） | 零 | 零 | 无 |
| Set\_Descriptor | 00000000B | SET\_DESCRIPTOR | 描述表种类（高字节，见表5）和索引（低字节） | 零或语言标志 | 描述表长 | 描述表 |
| Set\_Feature | 00000000B 00000001B 00000010B | SET\_FEATURE | 特性选择符(1表示设备，0表示端点） | 零  接口号  端点号 | 零 | 无 |
| Set\_Interface | 00000001B | SET\_INTERFACE | 可选设置 | 接口号 | 零 | 无 |
| Synch\_Frame | 100000010B | SYNCH\_FRAME | 零 | 端点号 | 二 | 帧号 |

其中bRequest为命令编码值，含意见表3：

|  |  |
| --- | --- |
| **表3、USB标准命令的编码值** | |
| bRequest | Value |
| GET\_STATUS | 0 |
| CLEAR\_FEATURE | 1 |
| 为将来保留 | 2 |
| SET\_FEATURE | 3 |
| 为将来保留 | 4 |
| SET\_ADDRESS | 5 |
| GET\_DESCRIPTOR | 6 |
| SET\_DESCRIPTOR | 7 |
| GET\_CONFIGURATION | 8 |
| SET\_CONFIGURATION | 9 |
| GET\_INTERFACE | 10 |
| SET\_INTERFACE | 11 |
| SYNCH\_FRAME | 12 |

# USB描述符

USB协议为USB设备定义了一套描述设备功能和属性的有固定结构的描述符，包括标准的描述符即设备描述符、配置描述符、接口描述符、端点描述符和字符串描述符，还有百标准描述符，如类描述符。USB设备通过这些描述符向USB主机汇报设备的各种各样属性，主机通过对这些描述符的访问对设备进行类型识别、配置并为其提供相应的客户端驱动程序。

　　USB设备通过描述符反映自己的设备特性。USB描述符是由特定格式排列的一组数据结构组成。

　　在USB设备枚举过程中，主机端的协义软件需要解析从USB设备读取的所有描述符信息。在USB主向设备发送读取描述符的请求后，USB设备将所有的描述符以连续的数据流方式传输给USB主机。主机从第一个读到的字符开始，根据双方规定好的数据格式，顺序地解析读到的数据流。

　　USB描述符包含标准描述符、类描述符和厂商特定描述３种形式。任何一种设备必须USB标准描述符（队字符串描述符可选外）。

　　在USB1.X中，规定了5种标准描述符：设备描述符（Device Descriptor)、配置描述符（Configuration Descriptor）、接口描述符（Interface Descriptor）、端点描述符（Endpoint Descriptor）和字符串描述符（String Descriptor）。

　　每个USB设备只有一个设备描述符，而一个设备中可包含一个或多个配置描述符，即USB设备可以有多种配置。设备的每一个配置中又可以包含一个或多个接口描述符，即USB设备可以支持多种功能（接口），接口的特性通过描述符提供。

　　在USB主机访问USB设备的描述符时，USB设备依照设备描述符、配置描述符、接口描述符、端点描述符、字符串描述符顺序将所有描述符传给主机。一设备至少要包含设备描述符、配置描述符和接口描述符，如果USB设备没有端点描述符，则它仅仅用默认管道与主机进行数据传输。

## 设备描述符

　　设备描述符给出了USB设备的一般信息，包括对设备及在设备配置中起全程作用的信息，包括制造商标识号ID、产品序列号、所属设备类号、默认端点的最大包长度和配置描述符的个数等。一个USB设备必须有且仅有一个设备描述符。设备描述符是设备连接到总线上时USB主机所读取的第一个描述符，它包含了14个字段，结构如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表4、USB设备描述符的结构** | | | | |
| 偏移量 | 域 | 大小 | 值 | 描述 |
| 0 | bLength | 1 | 数字 | 此描述表的字节数 |
| 1 | bDecriptorType | 1 | 常量 | 描述符的类型（此处应为0x01，即设备描述符） |
| 2 | bcdUSB | 2 | BCD码 | 此设备与描述表兼容的USB设备说明版本号（BCD 码） |
| 4 | bDeviceClass | 1 | 类 | 设备类码： 如果此域的值为0则一个设置下每个接口指出它自己的类，各个接口各自独立工作。  如果此域的值处于1~FEH之间，则设备在不同的接口上支持不同的类。并这些接口可能不能独立工作。此值指出了这些接口集体的类定义。  如果此域设为FFH，则此设备的类由厂商定义。 |
| 5 | bDeviceSubClass | 1 | 子类 | 子类码  这些码值的具体含义根据bDeviceClass 域来看。  如bDeviceClass 域为零，此域也须为零  如bDeviceClass 域为FFH，此域的所有值保留。 |
| 6 | bDevicePortocol | 1 | 协议 | 协议码  这些码的值视bDeviceClass 和 bDeviceSubClass 的值而定。  如果设备支持设备类相关的协议，此码标志了设备类的值。如果此域的值为零，则此设备不支持设备类相关的协议，然而，可能它的接口支持设备类相关的协议。如果此域的值为FFH，此设备使用厂商定义的协议。 |
| 7 | bMaxPacketSize0 | 1 | 数字 | 端点0的最大包大小（仅8,16,32,64 为合法值） |
| 8 | idVendor | 2 | ID | 厂商标志（由USB-IF组织赋值） |
| 10 | idProduct | 2 | ID | 产品标志（由厂商赋值） |
| 12 | bcdDevice | 2 | BCD 码 | 设备发行号（BCD 码） |
| 14 | iManufacturer | 1 | 索引 | 描述厂商信息的字符串描述符的索引值。 |
| 15 | iProduct | 1 | 索引 | 描述产品信息的字串描述符的索引值。 |
| 16 | iSerialNumber | 1 | 索引 | 描述设备序列号信息的字串描述符的索引值。 |
| 17 | bNumConfigurations | 1 | 数字 | 可能的配置描述符数目 |

其中bDescriptorType为描述符的类型，其含义可查下表（此表也适用于标准命令Get\_Descriptor中wValue域高字节的取值含义）：

|  |  |  |
| --- | --- | --- |
| **表5、USB描述符的类型值** | | |
| 类型 | 描述符 | 描述符值 |
| 标准描述符 | 设备描述符（Device Descriptor) | 0x01 |
| 配置描述符（Configuration Descriptor） | 0x02 |
| 字符串描述符（String Descriptor） | 0x03 |
| 接口描述符（Interface Descriptor） | 0x04 |
| 端点描述符（EndPont Descriptor） | 0x05 |
| 类描述符 | 集线器类描述符（Hub Descriptor） | 0x29 |
| 人机接口类描述符（HID） | 0x21 |
| 厂商定义的描述符 |  | 0xFF |

设备类代码bDeviceClass可查下表：

|  |  |  |
| --- | --- | --- |
| **表6、设备的类别（bDeviceClass）** | | |
| 值（十进制） | 值（十六进制） | 说明 |
| 0 | 0x00 | 接口描述符中提供类的值 |
| 2 | 0x02 | 通信类 |
| 9 | 0x09 | 集线器类 |
| 220 | 0xDC | 用于诊断用途的设备类 |
| 224 | 0xE0 | 无线通信设备类 |
| 255 | 0xFF | 厂商定义的设备类 |

下表列出了一个USB鼠标的设备描述符的例子，供大家分析一下：

|  |  |
| --- | --- |
| **表7、一种鼠标的设备描述符示例** | |
| 字段 | 描述符值（十六制） |
| bLength | 0x12 |
| bDecriptorType | 0x01 |
| bcdUSB | x0110 |
| bDeviceClass | 0x00 |
| bDeviceSubClass | 0x00 |
| bDevicePortocol | 0x00 |
| bMaxPacketSize0 | 0x08 |
| idVendor | 0x045E(Microsoft Corporation） |
| idProduct | 0x0047 |
| bcdDevice | 0x300 |
| iManufacturer | 0x01 |
| iProduct | 0x03 |
| iSerialNumber | 0x00 |
| bNumConfigurations | 0x01 |

## 配置描述符

　　配置描述符中包括了描述符的长度（属于此描述符的所有接口描述符和端点描述符的长度的和）、供电方式（自供电/总线供电）、最大耗电量等。主果主机发出USB标准命令Get\_Descriptor要求得到设备的某个配置描述符，那么除了此配置描述符以外，此配置包含的所有接口描述符与端点描述符都将提供给USB主机。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表8、USB配置描述符的结构** | | | | |
| 偏移量 | 域 | 大小 | 值 | 描述 |
| 0 | bLength | 1 | 数字 | 此描述表的字节数长度。 |
| 1 | bDescriptorType | 1 | 常量 | 配置描述表类型（此处为0x02） |
| 2 | wTotalLength | 2 | 数字 | 此配置信息的总长（包括配置，接口，端点和设备类及厂商定义的描述符） |
| 4 | bNumInterfaces | 1 | 数字 | 此配置所支持的接口个数 |
| 5 | bCongfigurationValue | 1 | 数字 | 在SetConfiguration（）请求中用作参数来选定此配置。 |
| 6 | iConfiguration | 1 | 索引 | 描述此配置的字串描述表索引 |
| 7 | bmAttributes | 1 | 位图 | 配置特性：  D7： 保留（设为一）  D6： 自给电源  D5： 远程唤醒  D4..0：保留（设为一）  一个既用总线电源又有自给电源的设备会在MaxPower域指出需要从总线取的电量。并设置D6为一。运行时期的实际电源模式可由GetStatus(DEVICE) 请求得到。 |
| 8 | MaxPower | 1 | mA | 在此配置下的总线电源耗费量。以 2mA 为一个单位。 |

下面是一种硬盘的配置描述符示例：

|  |  |
| --- | --- |
| **表9、一种硬盘的配置描述符示例** | |
| 字段 | 描述符值（十六进制） |
| bLength | 0x09 |
| bDescriptorType | 0x02 |
| wTotalLength | 0x01F |
| bNumInterfaces | 0x01 |
| bCongfigurationValue | 0x01 |
| iConfiguration | 0x00 |
| bmAttributes | 0x0C |
| MaxPower | 0x32 |

## 接口描述符

　　配置描述符中包含了一个或多个接口描述符，这里的“接口”并不是指物理存在的接口，在这里把它称之为“功能”更易理解些，例如一个设备既有录音的功能又有扬声器的功能，则这个设备至少就有两个“接口”。

　　如果一个配置描述符不止支持一个接口描述符，并且每个接口描述符都有一个或多个端点描述符，那么在响应USB主机的配置描述符命令时，USB设备的端点描述符总是紧跟着相关的接口描述符后面，作为配置描述符的一部分被返回。接口描述符不可直接用Set\_Descriptor和Get\_Descriptor来存取。

　　如果一个接口仅使用端点0，则接口描述符以后就不再返回端点描述符，并且此接口表现的是一个控制接口的特性，它使用与端点0相关联的默认管道进行数据传输。在这种情况下bNumberEndpoints域应被设置成0。接口描述符在说明端点个数并不把端点0计算在内。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表10、USB接口描述符的结构** | | | | |
| **偏移量** | **域** | **大小** | **值** | **说明** |
| 0 | bLength | 1 | 数字 | 此表的字节数 |
| 1 | bDescriptorType | 1 | 常量 | 接口描述表类（此处应为0x04） |
| 2 | bInterfaceNumber | 1 | 数字 | 接口号，当前配置支持的接口数组索引（从零开始）。 |
| 3 | bAlternateSetting | 1 | 数字 | 可选设置的索引值。 |
| 4 | bNumEndpoints | 1 | 数字 | 此接口用的端点数量，如果是零则说明此接口只用缺省控制管道。 |
| 5 | bInterfaceClass | 1 | 类 | 接口所属的类值：  零值为将来的标准保留。  如果此域的值设为FFH，则此接口类由厂商说明。  所有其它的值由USB 说明保留。 |
| 6 | bInterfaceSubClass | 1 | 子类 | 子类码  这些值的定义视bInterfaceClass域而定。 如果bInterfaceClass域的值为零则此域的值必须为零。  bInterfaceClass域不为FFH则所有值由USB 所保留。 |
| 7 | bInterfaceProtocol | 1 | 协议 | 协议码：bInterfaceClass 和bInterfaceSubClass 域的值而定.如果一个接口支持设备类相关的请求此域的值指出了设备类说明中所定义的协议. |
| 8 | iInterface | 1 | 索引 | 描述此接口的字串描述表的索引值。 |

对于bInterfaceClass字段，表示接口所属的类别，USB协议根据功能将不同的接口划分成不的类，其具体含义如下表所示：

|  |  |
| --- | --- |
| **表11、USB协议定义的接口类别（bInterfaceClass）** | |
| 值（十六进制） | 类别 |
| 0x01 | 音频类 |
| 0x02 | CDC控制类 |
| 0x03 | 人机接口类（HID） |
| 0x05 | 物理类 |
| 0x06 | 图像类 |
| 0x07 | 打印机类 |
| 0x08 | 大数据存储类 |
| 0x09 | 集线器类 |
| 0x0A | CDC数据类 |
| 0x0B | 智能卡类 |
| 0x0D | 安全类 |
| 0xDC | 诊断设备类 |
| 0xE0 | 无线控制器类 |
| 0xFE | 特定应用类（包括红外的桥接器等） |
| 0xFF | 厂商定义的设备 |

## 端点描述符

　　端点是设备与主机之间进行数据传输的逻辑接口，除配置使用的端点0（控制端点，一般一个设备只有一个控制端点）为双向端口外，其它均为单向。端点描述符描述了数据的传输类型、传输方向、数据包大小和端点号（也可称为端点地址）等。

　　除了描述符中描述的端点外，每个设备必须要有一个默认的控制型端点，地址为0，它的数据传输为双向，而且没有专门的描述符，只是在设备描述符中定义了它的最大包长度。主机通过此端点向设备发送命令，获得设备的各种描述符的信息，并通过它来配置设备。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表12、USB端点描述符的结构** | | | | |
| **偏移量** | **域** | **大小** | **值** | **说明** |
| 0 | bLength | 1 | 数字 | 此描述表的字节数长度 |
| 1 | bDescriptorType | 1 | 常量 | 端点描述表类（此处应为0x05） |
| 2 | bEndpointAddress | 1 | 端点 | 此描述表所描述的端点的地址、方向：  Bit 3..0 : 端点号. Bit 6..4 : 保留,为零  Bit 7:    方向,如果控制端点则略。  0：输出端点（主机到设备）  1：输入端点（设备到主机） |
| 3 | bmAttributes | 1 | 位图 | 此域的值描述的是在bConfigurationValue域所指的配置下端点的特性。  Bit 1..0 :传送类型  00=控制传送  01=同步传送  10=批传送  11=中断传送  所有其它的位都保留。 |
| 4 | wMaxPacketSize | 2 | 数字 | 当前配置下此端点能够接收或发送的最大数据包的大小。  对于实进传输，此值用于为每帧的数据净负荷预留时间。在实际运行时，管道可能不完全需要预留的带宽，实际带宽可由设备通过一种非USB定义的机制汇报给主机。对于中断传输，批量传输和控制传输，端点可能发送比之短的数据包 |
| 6 | bInterval | 1 | 数字 | 周期数据传输端点的时间间隙。  此域的值对于批传送的端点及控制传送的端点无意义。对于同步传送的端点此域必需为1，表示周期为1ms。对于中断传送的端点此域值的范围为1ms到255ms。 |

下表是一种鼠标的端点描述符的示例，该端点是一个中断端点：

|  |  |
| --- | --- |
| **表13、一种鼠标的端点描述符示例** | |
| 域 | 值（十六进制） |
| bLength | 0x07 |
| bDescriptorType | 0x05 |
| bEndpointAddress | 0x81 |
| bmAttributes | 0x03 |
| wMaxPacketSize | 0x04 |
| bInterval | 0x0A |

## 字符串描述符

　　字符串描述符是一种可选的USB标准描述符，描述了如制商、设备名称或序列号等信息。如果一个设备无字符串描述符，则其它描述符中与字符串有关的索引值都必须为0。字符串使用的是Unicode编码。

　　主机请示得到某个字符串描述符时一般分成两步：首先主机向设备发出USB标准命令Get\_Descriptor，其中所使用的字符串的索引值为0，设备返回一个字符串描述符，此描述符的结构如下：

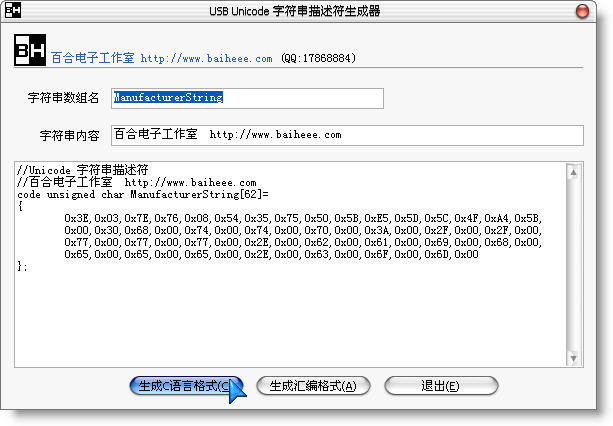
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表14、USB字符串描述符（响应主机请求时返回的表示语言ID的字符串描述符）** | | | | |
| **偏移量** | **域** | **大小** | **值** | **描述** |
| 0 | bLength | 1 | N+2 | 此描述表的字节数 |
| 1 | bDescriptorType | 1 | 常量 | 字串描述表类型（此处应为0x03） |
| 2 | wLANGID[0] | 2 | 数字 | 语言标识（LANGID）  码0 |
| **…** | **…** | **…** | **…** | **…** |
| N | wLANGID[x] | 2 | 数字 | 语言标识（LANGID）  码X |

该字符串描述符双字节的语言ID的数组，wLANGID[0]~wLANGID[x]指明了设备支持的语言，具体含义可查看[USB\_LANGIDs.pdf](http://www.baiheee.com/Documents/090518/090518112619/USB_LANGIDs.pdf)。

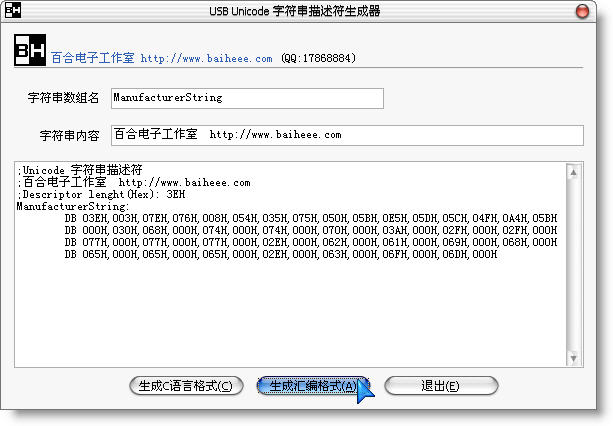
　　主机根据自己需要的语言，再次向设备发出USB标准命令Get\_Descriptor，指明所要求得到的字符串的索引值和语言。这次设备所返回的是Unicode编号的字符串描述符，其结构如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表15、Unicode字符串描述符（响应主机请求时真正表示字符串编码的字符串描述符）** | | | | |
| **偏移量** | **域** | **大小** | **值** | **描述** |
| 0 | bLength | 1 | 数字 | 此描述表的字节数（bString域的数值N＋2） |
| 1 | bDescriptorType | 1 | 常量 | 字串描述表类型（此处应为0x03） |
| 2 | bString | N | 数字 | UNICODE 编码的字串 |

bString域为设备实际返回的以UNICODE编码的字符串流，我们在编写设备端硬件驱动的时候需要将字符串转换为UNICODE编码，您可以通过一些UNICODE转换工具进行转换。这里推荐由[百合电子工作室](http://www.baiheee.com/)开发的一款USB描述符生成工具“[USB Unicode 字符串描述符生成器](http://www.baiheee.com/Documents/090518/090518112619/USB%20Unicode%20%E5%AD%97%E7%AC%A6%E4%B8%B2%E6%8F%8F%E8%BF%B0%E7%AC%A6%E7%94%9F%E6%88%90%E5%99%A8.rar)”，它专门为编写设备端驱动程序的需要而定制，可以非常方便将您需要的字符串转换成UNICODE格式，进而导入您的C或汇编程序代码中，以下是它的界面：



USB Unicode 字符串描述符生成器－生成C语言格式



USB Unicode 字符串描述符生成器－生成汇编格式

温馨提示：

[百合电子工作室](http://www.baiheee.com/)有一个关于USB开发方面的开源项目－[Easy USB 51 Programer](http://www.baiheee.com/OpenSource/Easy%20USB%2051%20Programer/Easy%20USB%2051%20Programer1.htm)，整个开发过程写得非常详细，不防参考一下，您还可以进入她的论坛参与此项目的讨论。

注：本篇文章部分内容摘自《USB协议》和《USB技术及应用设计》