USB cho dev (Chap.05 – Đặc tả thiết bị)

1. Mỗi miêu tả giao diện có 1 trường là:

* **bInterfaceNumber** để chỉ số của Giao diện
* **bAlternateSetting**, sẽ cho phép một giao diện thay đổi bất cứ lúc nào

VD: thiết bị có 2 giao diện, giao diện 1 và 2.

Giao diện 1: **bInterfaceNumber** được gán là zero (tức là giao diện đầu tiên), và **bAlternateSetting** cũng là 0 (tức là cấu hình mặc định)

Giao diện 2: **bInterfaceNumber** là 1, tức là chỉ giao diện thứ 2 và cũng **bAlternateSetting** được gán là 0 (tức là cấu hình mặc định)

Trong quá trình sử dụng, Host có thể gửi một yêu cầu **SetInterface** đến interface 1 yêu cầu nó chuyển sang setting thay thế. Khi đó nó sẽ chạy trên một miêu tả giao diện khác.

Chúng ta có được đến 2 cấu hình. Chúng ta vẫn có thể gửi dữ liệu qua interface zero, trong khi thay đổi setting Endpoint liên quan đến interface 1 mà không ảnh hưởng đến interface zero.

Mỗi miêu tả Endpoint được sử dụng để chỉ rõ loại truyền, hướng truyền, tần số polling và kích thước lớn nhất của packet cho mỗi Endpoint. Endpoint Zero mặc định được sủ dụng là Endpoint điều khiển và nó không bao giờ có miêu tả.

1. Sự kết hợp các loại đặc tả (Composition of USB Descriptors)

Byte đầu tiên chỉ ra độ dài của miêu tả, trong khi byte thứ 2 chỉ loại đặc tả. Nếu độ dài của một đặc tả nhỏ hơn kích thước mà đặc trả USB đã định nghĩa thì Host sẽ bỏ qua luôn. Tuy nhiên, nếu kích thước lớn hơn thì Host sẽ bỏ phần thừa ra và tìm miêu tả kế thiếp từ vị trí cuối cùng độ dài thực tế.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Ý nghĩa |
| 0 | bLength | 1 | Number | Kích thước Descriptor tính theo Bytes |
| 1 | bDescriptionType | 1 | Constant | Loại đặc tả |
| 2 | … | n |  | Các tham số của đặc tả |

1. Miêu tả thiết bị (Device Descriptors)

Sẽ miêu tả toàn bộ thông tin về thiết bị. Tất nhiên, vì vậy nó chỉ có 1 thôi. Nó diễn tả một vài thông tin cơ bản, trong đó có một vài thông tin quan trọng như phiên bản USB nó hỗ trợ, kích thước tối đa của packet, nhà sản xuất (Vendor), mã sản phầm. Số lượng cấu hình mà thiết bị có. Định dạng được miêu tả dưới bảng sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bLength | 1 | Number | Kích thước của Descriptor (18 bytes) |
| 1 | bDescriptorType | 1 | Constant | Là loại Device Descriptors (0x01) |
| 2 | bcdUSB | 2 | BCD | Phiển bản đặc tả USB mà thiết bị tuân theo. |
| 4 | bDeviceClass | 1 | Class | Max class (Được gán bởi USB Org)  Nếu nó bằng 0 thì mỗi interface nên có class riêng của nó.  Nếu là 0xFF thì class này do Vendor tự quy định.  Ngoài ra thì mọi giá trị đều hợp lệ. |
| 5 | bDeviceSubClass | 1 | SubClass | Mã của Subclass  (Được gán bởi USB Org) |
| 6 | bDeviceProtocol | 1 | Protocol | Mã protocol (Assigned by USB Org) |
| 7 | bMaxPacketSize | 1 | Number | Kích thước lớn nhất của Packet khi sử dụng Endpoin Zero (mặc định phải có). Các giá trị có thể là 8, 16, 32, 64 |
| 8 | idVendor | 2 | ID | Vendor ID (Assigned by USB Org) |
| 10 | idProduct | 2 | ID | Product ID (Assigned by Manufacturer) |
| 12 | bcdDevice | 2 | BCD | Device Release Number |
| 14 | iManufacturer | 1 | Index | Chỉ số của chuỗi mô tả nhà sản xuất |
| 15 | iProduct | 1 | Index | Chĩ số của chuỗi mô tả sản phẩm |
| 16 | iSerialNumber | 1 | Index | Chỉ số của mã Serial của sản phẩm |
| 17 | bNumConfigurations | 1 | Integer | Số cấu hình hiện có của thiết bị |

* bcdUSB : phiên bản cao nhất của USB mà thiết bị hỗ trợ. Giá trị ở dạng 0xJJMN, JJ là phiên bản major, M là miên bản minor, N là sub minor. Ví dụ USB 2.0 sẽ là 0x0200, USB1.1 sẽ là 0x1100, và USB1.0 sẽ là 0x1000.
* bDeviceClass, bDeviceSubClass và bDeviceProtocol : Là thông số chủ yếu để hệ điều hành phía Host xác định driver cho thiết bị. Thông thường, chỉ bDeviceClass được set ở mức miêu tả thiết bị, còn 2 thông số còn lại được xác định tại các mức miêu tả interface. Điều này cho phép một thiết bị hỗ trợ nhiều class.
* bMaxPacketSize : Kích thước packet của Endpoint Zero. Tất cả các thiết bị bắt buộc phải support EndPoint Zero.
* idVendor và idProduct : được hệ điều hành phía Host sử dụng để tìm driver cho thiết bị, VendorID được gán bởi [*USB-IF*](http://www.usb.org/)*.*
* bcdDevice: có cùng định dạng với bcdUSB và được sử dụng để cung cấp phiên bản của thiết bị. Giá trị này sẽ được gán bởi developer người phát triển thiết bị.
* 3 chuỗi miêu tả sẽ tồn tại để cung cấp những thông tin chi tiết về nhà sản xuất, sản phẩm, số serial. Không bắt buộc phải có 3 chuỗi này, nếu không có 2 chuỗi này thì index zero sẽ được sử dụng.
* bNumConfigurations : định nghĩa số lượng cấu hình mà thiết bị hỗ trợ tại tốc độ hiện tại của nó.

1. Miêu tả cấu hình (Configuration Descriptors)

Một thiết bị có thể có một vài cấu hình khác nhau mặc dù hầu hết đơn giản và chỉ có 1 mà thôi. Miêu tả cấu hình sẽ chứa các thông tin như nguồn thiết bị sử dụng là gì? năng lượng tối đa tiêu thụ là bao nhiêu?Số lượng giao diện mà nó có? Vì thế, ví dụ ta có thể có 2 cấu hình, thứ nhất được dùng khi sử dụng nguồn từ bus (bus powered), thứ 2 được dùng khi sử dụng nguồn chính nó (self-powered). Hoặc ta có thể có một cấu hình mà sử dụng phương thức truyền khác với cấu hình còn lại. etc.

Tất cả những cấu hình này được thiết bị gửi đến Host. Phía Host sẽ đánh giá xem nên sử dụng cấu hình nào, và gửi yêu cầu **SetConfiguration** về phía thiết bị Function với tham số là giá trị được lấy ra từ trường **bConfigurationValue** của 1 trong những cấu hình nó được gửi cho.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bLength | 1 | Number | Kích thước của miêu tả cấu hình tính theo byte |
| 1 | bDescriptorType | 1 | Constant | Loại mô tả là Configuration (0x02) |
| 2 | wTotalLength | 2 | Number | Tổng số byte trả về từ phần dữ liệu |
| 4 | bNumInterfaces | 1 | Number | Số lượng giao diện (interfaces) |
| 5 | bConfigurationValue | 1 | Number | Gía trị được sử dụng như là tham số khi set cấu hình sử dụng |
| 6 | iConfiguration | 1 | Index | Chỉ số của chuỗi miêu tả về cấu hình này |
| 7 | bmAttributes | 1 | Bitmap | D7 : (bit số 7) không sử dụng, set to 1. (USB 1.0 thì nghĩa là Bus Powered)  D6 : Nếu được set, nó sẽ là self-powered  D5 : Dùng để Remote Wakeup (một tín hiệu từ thiết bị đến Host mà không đợi Host hỏi)  D4..0 : Không sử dụng, được gán toàn bộ là 0. |
| 8 | bMaxPower | 1 | mA | Lượng điện tiêu thụ tính theo đơn vị  2mA |

Khi một mô tả cấu hình được đọc, nó sẽ trả về toàn bộ cấu trúc cây của cấu hình, bao gồm toàn bộ các interface liên quan, các miêu tả Endpoint nữa. Trường **wTotalLength** miêu tả số lượng byte trong cấu trúc được gửi về Host.

* bNumInterfaces : là số lượng interface có trong cấu hình này
* bConfigurationValue : được sử dụng bởi **SetConfiguration** để chọn cấu hình này.
* iConfiguration : chỉ số của chuỗi mô tả cấu hình này (là chuỗi dễ đọc)
* bmAttributes : các tham số cho cấu hình này, bao gồm : nếu là self-powered powered thì D6 (bit số 6) được set. Bit D7 được sử dụng để xác định là bus-power hay không chỉ trong trường hợp USB1.0, các phiên bản sau sử dụng giá trị bMaxPower. Nếu thiết bị có sử dụng nguồn từ bus thì bắt buộc phải khai báo giá trị bMaxPower. Bit D5 được sử dụng để xác định xem có hỗ trợ remote wakeup (phía Function chủ động đánh thức Host khi Host đang ở trạng thái nghỉ)
* bMaxPower : quy định giá trị lớn nhất của dòng mà thiết bị sẽ chạy. Đơn vị ở đây là 2m Ampe. Thiết bị được phép chạy tối đa lên đến 500m Ampe. Các thiết bị high speed mà sử dụng nguồn bus thì không được phép lớn hơn 500m Ampe. Nếu một thiết bị mất nguồn bên ngoài

1. Mô tả giao diện (Interface Descriptors)

Giao diện có thể được thấy là một nhóm các End Point thực hiện một chức năng của thiết bị. Miêu tả giao diện tuân theo các định dạng sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bLength | 1 | Number | Kích thước của miêu tả tính bằng byte (9 Bytes) |
| 1 | bDescriptorType | 1 | Constant | Là loại miêu tả giao diện (0x04) |
| 2 | bInterfaceNumber | 1 | Number | Số lượng giao diện |
| 3 | bAlternateSetting | 1 | Number | Gía trị được sử dụng thay đổi giao diện tức thời(alternative setting) |
| 4 | bNumEndpoints | 1 | Number | Số lượng Endpoints có trong Interface |
| 5 | bInterfaceClass | 1 | Class | Mã class (lấy từ danh sách được quy định bởi USB Org) |
| 6 | bInterfaceSubClass | 1 | SubClass | Mã Subclass (lấy từ danh sách được quy định bởi USB Org) |
| 7 | bInterfaceProtocol | 1 | Protocol | Mã giao thức (Protocol) (lấy từ danh sách được quy định bởi USB Org) |
| 8 | iInterface | 1 | Index | Chỉ số của các chuỗi của interface này trong miêu tả chuỗi |

* bInterfaceNumber: chỉ số của interface hiện tại. Giá trị này bắt đầu từ Zero và tăng lên với mỗi interface mới.
* bAlternativeSetting: được sử dụng cho giao diện thay thế tức thời thông qua Set Interface Request.
* bNumEndpoints: số lượng Endpoint được sử dụng trong Interface. Giá trị này không bao gồm Endpoint Zero, mỗi Endpoint được miêu tả kĩ ở phần sau.
* bInterfaceClass, bInterfaceSubClass, bInterfaceProtocol: sử dụng để chỉ ra class nào được support (lớp thiết bị nào) (ví dụ: HID – chuột, bàn phím chẳng hạn, Communication – USB2Serial chẳng hạn, Mass Storage – USB Flash chẳng hạn). Các giá trị này sẽ giúp cho việc phân chia driver cho thiết bị thành các lớp nên giảm việc phải viết driver cho từng thiết bị cụ thể.
* iInterface: Chỉ số của chuỗi mô tả cho interface này.

1. Mô tả EndPoints

Miêu tả các Endpoint khác Endpoin Zero. End Point luôn được được sử dụng là Control Endpoint (Endpoint điều khiển thiết bị) trước khi bất cứ miêu ta nào được Host yêu cầu. Định dạng trả sẽ theo bảng dưới đây:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bLength | 1 | Number | Kích thước của miêu tả tính bằng byte (7 bytes) |
| 1 | bDescriptorType | 1 | Constant | Là miêu tả EndPoint (0x05) |
| 2 | bEndpointAddress | 1 | Endpoint | Địa chỉ Endpoint Bits 0..3b Số Endpoint (tối đa là 16) Bits 4..6b Không sử dụng Bits 7 Hướng, 0 = Đi ra, 1 = Đi vào (không tính Endpoint Control) |
| 3 | bmAttributes | 1 | Bitmap | Bits 0..1 Loại truyền  00 = Control (điều khiển)  01 = Isochronous (đồng bộ)  10 = Bulk (hàng loạt)  11 = Interrupt (ngắt)  Bits 2..7 Không sử dụng nếu là Endpoint Isochronous. Bits 3..2 = Loại đồng bộ (chỉ có Iso 0x01)  00 = No Synchonisation (không đồng bộ)  01 = Asynchronous (đồng bộ bất đối xứng)  10 = Adaptive (đồng bộ thích nghi)  11 = Synchronous (đồng bộ đối xứng)  Bits 5..4 = Loại sử dụng (chỉ có mode Iso 0x01)  00 = Data Endpoint (Endpoint data)  01 = Feedback Endpoint (Endpoint đáp lại)  10 = Explicit Feedback Data Endpoint (Endpoint trả lại data)  11 = Reserved (Không sử dụng ) |
| 4 | wMaxPacketSize | 2 | Number | Kích thước lớn nhất cho Packet Size mà có thể gửi/nhận trên Endpoint này. |
| 6 | bInterval | 1 | Number | Tần suất để polling Data từ Endpoint. Value in frame counts. Ignored for Bulk & Control Endpoints. Isochronous must equal 1 and field may range from 1 to 255 for interrupt endpoints. |

* bEndpointAddress: Tức là Endpoint nào
* bmAttributes: Loại truyền dữ liệu sử dụng trên EndPoint đó. Nó là 1 trong 4 loại đã nói ở các chương trước: Control, Interrupt, Isochronous, Bulk. Nếu một Endpoint Isochronous được sử dụng, thì các thuộc tính khác cũng được sử dụng theo, còn lại nó không có ý nghĩa gì hết.
* wMaxPacketSize: Kích thước tối đa mà payload có thể chứa khi truyền nhận trên Endpoint này.
* bInterval: Tần số polling (tần số hỏi dữ liệu). Đơn vị được tính là frame. (1ms cho low/full speed, và 125us cho high speed)

1. Miêu tả chuỗi (String Descriptor)

Phần miêu tả này chưa thông tin về các chuỗi miêu tả các loại Descriptor khác, giúp người đọc dễ dàng nhận biết các thông tin. Các chuỗi này là không bắt buộc. Các loại miêu tả không sử dụng các chuỗi dễ đọc thì trường tham chiếu đến chuối dễ đọc nên để là ZERO.

Các chuỗi này được Encode theo định dạng của tổ chức [Unicode](http://www.unicode.org/), có thể hỗ trợ đa ngôn ngữ. Các chuỗi được miêu tả cũng vậy, nên support nhiều ngôn ngữ theo các mã ngôn ngữ được miêu tả ở [Universal Serial Bus Language Identifiers (LANGIDs) version 1.0](http://www.usb.org/developers/data/USB_LANGIDs.pdf).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bLength | 1 | Number | Kích thước của miêu tả (tính bằng byte) |
| 1 | bDescriptorType | 1 | Constant | Đây là miêu tả chuỗi (0x03) |
| 2 | wLANGID[0] | 2 | number | Ngôn ngữ thứ nhất (e.g. 0x0409 English – United States) |
| 4 | wLANGID[1] | 2 | number | Ngôn ngữ thứ 2(e.g. 0x0c09 English – Australian) |
| n | wLANGID[x] | 2 | number | Ngôn ngữ thứ x (e.g. 0x0407 German – Standard) |

Miêu tả chuỗi trên là miêu tả của Miêu tả chuỗi Zero. Host nên đọc các thông tin miêu tả này để tìm ra ngôn ngữ nào được support. Nếu một ngôn ngữ được hỗ trợ, nó sẽ được tham chiếu thông qua LanguageID được gán vào giá trị trường wIndex của Get Descriptor(String).

Tất các các chuỗi được miêu tả sau đó sẽ có dạng sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bLength | 1 | Number | Kích thước ở dạng byte |
| 1 | bDescriptorType | 1 | Constant | Dang miêu tả chuỗi (0x03) |
| 2 | bString | n | Unicode | Chuỗi được Encode bởi chuẩn Unicode |

# USB cho dev (Chap.06 – Các gói tin Setup)

Mỗi thiết bị USB phải trả lời các gói tin Setup (Setup packets) trên Endpoint mặc định (Endpoint Zero). Các gói tin Setup được sử dụng cho việc phát hiện thiết bị, cấu hình, cũng như lấy các thông tin khác như các thông tin về chức năng, địa chỉ thiết bị, kiểm tra trạng thái các Endpoint.

Chuẩn USB yêu cầu Host sẽ mon muốn về mặt thời gian từ phát hiện đến lấy đầy đủ các thông tin trên trong vòng không quá 5 giây. Ngoài ra, nó cũng quy định chặt hơn cho từng Request từ phía Host.

* Yêu cầu thông tin chuẩn của thiết bị (Standard Device request) mà không có dữ liệu (hay DATA Stage) thì phải hoàn thành trong vòng 50ms.
* Nếu yêu cầu có dữ liệu kèm theo (có DATA stage) thì dữ liệu phải được trao đổi sau ít nhất 500ms sau request được gửi đến.
  + Mỗi gói dữ liệu phải gửi trong 500ms sau khi gói trước được gửi
  + Thông báo trạng thái (Status Stage) phải hoàn thành trong vòng 50ms sau khi gửi gói tin cuối cùng.
* Lệnh SetAddresss (có chưa phần Data) phải được thực hiện và trả lại Status trong vòng 50ms. Phía thiết bị có 2ms để thực hiện thay địa chỉ trước khi request tiếp theo được gửi đến.Nhưng yêu cầu về thời gian ở trên hoàn toàn có thể thực hiện được ngay cả với những thiết bị chậm nhất đi nữa. Nhưng những giới hạn về thời gian này có thể gây khó khăn trong quá trình debug. Trong vòng 50ms, chẳng thể có kí tự debug nào được gửi đi ở 9600bps ở dạng serial không đồng bộ hoặc một Emulator cho mỗi step chạy ở các thanh ghi. Chính vì thế, Khi debug USB, người cần có một vài phương pháp khác. (cái này chưa bàn đến)

Mỗi request sẽ bắt đầu bằng 8 byte, theo định dạng sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offset | Field | Size | Value | Description |
| 0 | bmRequestType | 1 | Bit-Map | **D7 Hướng truyền DATA** 0 = Host to Device 1 = Device to Host **D6..5 Loại** 0 = Standard 1 = Class 2 = Vendor 3 = Reserved **D4..0 Phía nhận** 0 = Device 1 = Interface 2 = Endpoint 3 = Other 4..31 = Reserved |
| 1 | bRequest | 1 | Value | Request |
| 2 | wValue | 2 | Value | Value |
| 4 | wIndex | 2 | Index or Offset | Index |
| 6 | wLength | 2 | Count | Số lượng byte truyền trong 1 Data Stage |

**+ bmRequestType**: Xác định chiều của request (phần DATA), loại request, cũng như loại thông tin sẽ tiếp nhận.

**+ bRequest**: Muốn nói rằng, đây là Request.

Thông thường **bmRequestType** được đọc, và sẽ được chia ra các xử lý theo các handler như **Standard Device request Handler**, **Standard Interface request Handler**, hoặc **Standard Endpoint** **request Handler**, hay **Class Device request handler**.etc. Tất nhiên, cách xử lý trên không bắt buộc, biệc đọc nội dung packet Setup cũng như xử lý phụ thuộc hoàn toàn các thiết kế phía Device của bạn. Một số chọn cách đọc **bRequest** trước tiên sau đó mới xác định đến loại thông tin sẽ tiếp nhận.

Các request chuẩn (**Standard request**) là phổ biến ở tất cả các thiết bị USB, nó sẽ được nói chi tiết ở bên dưới. **Class request** phổ biến cho các lớp Driver. Ví dụ, **tất cả các thiết bị tuân theo HID class thì đều có một tập các request class chung. Tập class này khác với Communication class, và tất nhiên khác với Mass Storage class**.

Và cuối cùng, là các request mà các vendor tự định nghĩa. Nhưng request này được người thiết kế thiết bị tự đặt, các thiết bị thường không giống nhau. Và tất nhiên, người ta có thể làm bất cứ thứ gì người ta muốn.

Một request phổ biến có thế hướng đến nhiều đối tượng khác nhau (thiết bị, giao diện, hay Endpoint). **GetStatus Standard request** chẳng hạn, nó có thể cho thiết bị, giao diện hoặc Endpoint. Khi hướng đến thiết bị, nó sẽ trả về trạng thái của remote wake up và nếu thiết bị là tự cung cấp nguồn. Tuy nhiên, nếu cùng request đó được gửi đến interface thì nó luôn trả lại zero, còn đến Endpoint thì nó lại trả lại một dừng tạm thời cho EndPoint đó.

Giá trị **wValue** và **wIndex** cho phép các tham số được kèm theo request. **wLength** chỉ số lượng byte được truyền ở DATA stage.

* 1. **Request chuẩn (Standard Request)**

Ở Section 9.4 trong đặc tả về USB, có nói rằng, “**Standard request**” phải được thực hiện trên mọi USB Device. Tài liệu đó cũng cung cấp một bảng của các request có thể. Ta hãy coi rằng phía firmware của thiết bị USB sẽ thực hiện đọc các gói Setup chưa chi chúng ra để xử lý tướng ứng vỡi mỗi loại đối tượng là Device, Interface hay Endpoint.

Request chuẩn cho Device (**Standard Device Request**)

Có 8 loại request chuẩn cho thiết bị, được liệt kê ở bảng dưới đây:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| bmRequestType | bRequest | wValue | wIndex | wLength | Data |
| 1000 0000b | GET\_STATUS (0x00) | Zero | Zero | Two | Device Status |
| 0000 0000b | CLEAR\_FEATURE (0x01) | Feature Selector | Zero | Zero | None |
| 0000 0000b | SET\_FEATURE (0x03) | Feature Selector | Zero | Zero | None |
| 0000 0000b | SET\_ADDRESS (0x05) | Device Address | Zero | Zero | None |
| 1000 0000b | GET\_DESCRIPTOR (0x06) | Descriptor Type & Index | Zero or Language ID | Descriptor Length | Descriptor |
| 0000 0000b | SET\_DESCRIPTOR (0x07) | Descriptor Type & Index | Zero or Language ID | Descriptor Length | Descriptor |
| 1000 0000b | GET\_CONFIGURATION (0x08) | Zero | Zero | 1 | Configuration Value |
| 0000 0000b | SET\_CONFIGURATION (0x09) | Configuration Value | Zero | Zero | None |

* GET\_STATUS : Device phải trả lại 2 byte theo dạng dưới đây:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| Reserved | | | | | | | | | | | | | | Remote Wakeup | Self Powered |

Nếu D0 được set, thì có nghĩa thiết bị là nguồn tự cung, ngược lại thì đó là nguồn Bus.

Nếu D1 được set, thì có nghĩa là Remote wakep up được bật, thiết bị có thể wake up Host khi Host đang suspended. Bit này có thể được thay đôi bởi SET\_FEATURE hoặc CLEAR\_FEATURE.

* CLEAR\_FEATURE và SET\_FEATURE: Sử dụng để set có hay không các feature. Khi nó hướng đên Device thì chỉ có 2 feature có thể set. Đó là DEVICE\_REMOTE\_WAKEUP và TEST\_MODE. Test mode cho phép thiết bị diễn đạt nhiều điều kiện khác nhau. Chi tiết phần này được nói đến ở đặc tả USB 2.0.
* SET\_ADDRESS: Sử dụng để gán một địa chỉ duy nhất cho thiết bị trong quá trình **Enumuration**. Chính là giá trị của trường wValue, tối đa đến 127. Ở request này, thiết bị dù đã có giá trị địa chỉ nhưng vẫn phải đợi thông báo trạng thái trước khi thực sự set địa chỉ cho chính nó. Còn với các request khác, nó thường được thực hiện trước khi **Status** đến.
* SET\_DESCRIPTOR/GET\_DESCRIPTOR: được sử dụng để trả về một **descriptor** trong **wValue**. Một request cho miêu tả cấu hình sẽ trả về tất cả các thông tin của miêu tả thiết bị, các interface, các Endpoint nữa. Có 1 chú ý ở đây:
  + Endpoint Descriptor: không thể được truy cập trực tiếp từ GET\_DESCRIPTOR/SET\_DESCRIPTOR request.
  + Interface Descriptor: giống với **Endpoint Descriptor**.
  + String Descriptor: cho phép truy cập trực tiếp
* GET\_CONFIGURATION/SET\_CONFIGURATION: Được sử dụng để lấy thông tin cấu hình thiết bị hoặc set cấu hình. Trạng thái trả về ở **Status** Stage là 1 byte giá trị. Nếu là 0, thì có nghĩa thiết bị chưa được configured, nếu khác 0 thì có nghĩa thiết bị đã được configured. SET\_CONFIGURATION được sử dụng để cho phép một thiết bị chính thức hoạt động. Nó chứa giá trị của **bConfigurationValue** mà Host muốn thiết lập.
  1. **Request Giao diện chuẩn**

Định nghĩa 5 loại yêu cầu giao diện chuẩn, chi tiết được nêu dưới bảng sau. Chỉ có 2 request là được sử dụng một cách rõ ràng

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| bmRequestType | bRequest | wValue | wIndex | wLength | Data |
| 1000 0001b | GET\_STATUS (0x00) | Zero | Interface | Two | Interface Status |
| 0000 0001b | CLEAR\_FEATURE (0x01) | Feature Selector | Interface | Zero | None |
| 0000 0001b | SET\_FEATURE (0x03) | Feature Selector | Interface | Zero | None |
| 1000 0001b | GET\_INTERFACE (0x0A) | Zero | Interface | One | Alternate Interface |
| 0000 0001b | SET\_INTERFACE (0x11) | Alternative Setting | Interface | Zero | None |

**wIndex**: Tức là chỉ số của Interface mà request này hướng đến, định dạng được nêu ra ở dưới đây:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| Reserved | | | | | | | | Interface Number | | | | | | | |

* GET\_STATUS: được sử dụng để trả về trạng thái của interface. Giá trị thường sẽ là 2 byte (0x00, 0x00)
* CLEAR\_FEATURE/SET\_FEATURE: Sử dụng để set các giá trị boolean. Khi request cho giao diện thì không có đặc trưng này.
* GET\_INTERFACE/SET\_INTERFACE: sẽ có trường **Alternative Interface** mà ta đã miêu tả ở các chương trước.
  1. **Request Endpoint chuẩn (Standard Endpoint Request)**

Có 4 loại khác nhau được đưa ra dưới đây:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| bmRequestType | bRequest | wValue | Windex | wLength | Data |
| 1000 0010b | GET\_STATUS (0x00) | Zero | Endpoint | Two | Endpoint Status |
| 0000 0010b | CLEAR\_FEATURE (0x01) | Feature Selector | Endpoint | Zero | None |
| 0000 0010b | SET\_FEATURE (0x03) | Feature Selector | Endpoint | Zero | None |
| 1000 0010b | SYNCH\_FRAME (0x12) | Zero | Endpoint | Two | FrameNumber |

* **wIndex**: tham chiếu đến Endpoint và hướng của chúng. Định dạng sẽ như bên dưới:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| Reserved | | | | | | | | Dir | Reserved | | | Endpoint Number | | | |

* GET\_STATUS:trả lại 2 bytes trạng thái (**Halted/Stalled**) của **Endpoint**. Định dạng của 2 byte đó như sau:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| Reserved | | | | | | | | | | | | Halt | | | |

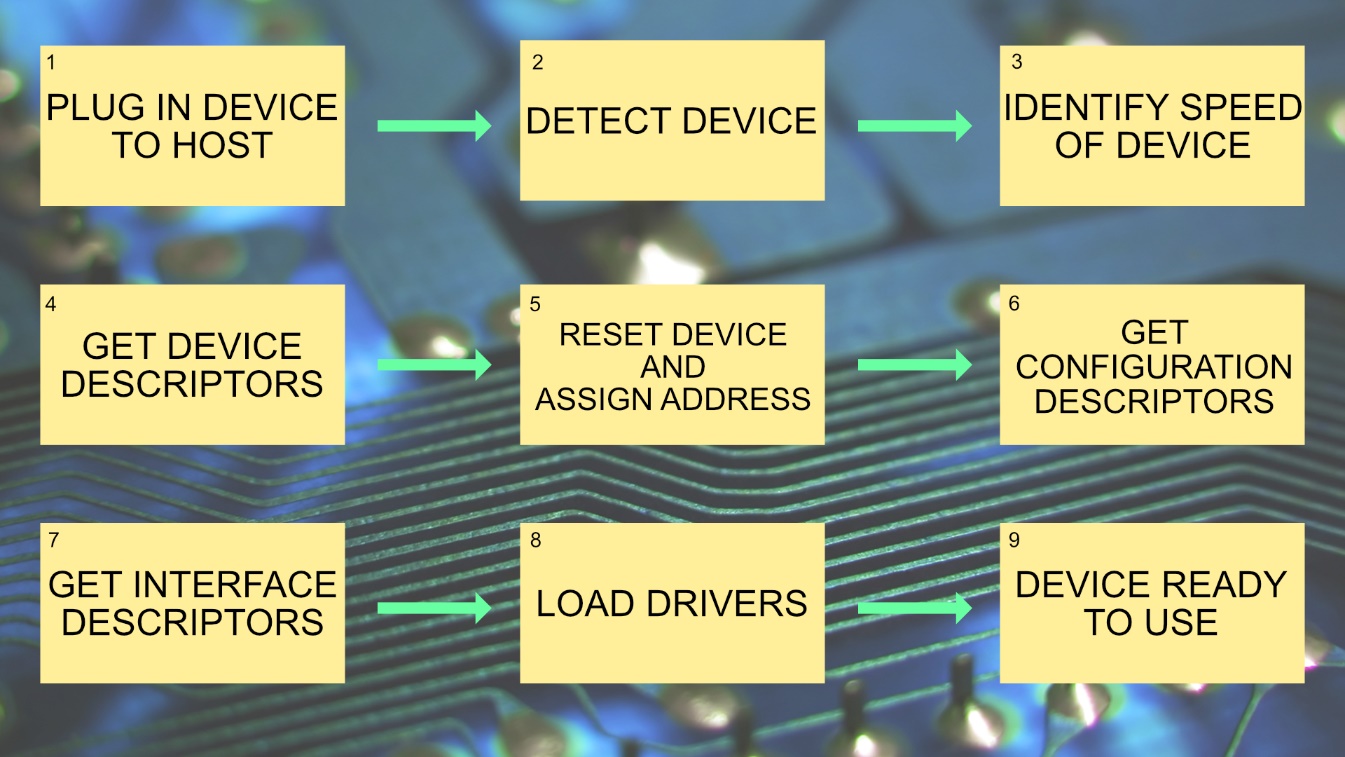
* CLEAR\_FEATURE/SET\_FEATURE: Sử dụng để set Endpoint Features. Hiện tại chỉ có 1 feature selector, ENDPOINT\_HALT(0x00) giúp Host có thể ngưng stall hoặc reset một Endpoint. Chỉ những Endpoint khác Endpoint mặc định mới nên sử dụng chức năng này.
* **Synch frame**: sử dụng để báo một frame đồng bộ Endpoint.

Enumeration

Đây là quá trình mà host phát hiện sự xuất hiện của device, xác định loại thiết bị của device, định nghĩa tốc độ kết nối. Quá trình này rất quan trọng bởi mỗi một thiết bị USB khác nhau sẽ giao tiếp với host khác nhau.

VD như chuột hoặc USB bộ nhớ. Một khi được kết nối, những device này sẽ hành xử khác nhau bởi chúng có function khác nhau. Chuột là thiết bibj HID sẽ cung cấp chức năng cho người dùng giao tiếp với host. Thiết bị này vận hành hoàn toàn khác nhau so với ổ cúng. Một ổ cứng được dùng để đọc và ghi data đến và vào máy tính. Chuột và ổ cứng là các thiết bị usb khác nhau, dẫu vậy máy tính host vẫn biết cách để giao tiếp với mỗi thiết bị bởi phần đặc tả thiết bị USB được gửi tới host trong quá trình enumeration.

Enumeration được vận hành như hình:

  
**USB descriptor:** Đặc tả USB được gới thiệu tới host trong quá trình enumeration. Những đặc tả này cho host biết loại thiết bị được kết nối đến và cách để giao tiếp với chúng. Có 4 loại descriptor: device desciptor, configuration descriptor, interface descriptor và endpoint descriptor.

**Device descriptor:**

Mỗi thiết bị USB chỉ có một device descriptor. Đặc tả này chứa những thông tin để áp dụng toàn cục cho thiết bị như là S/N, vendor ID, product ID. Những đặc tả này cũng có thông tin về device class. PC có thể sử dụng các thông tin này để xác định driver nào phù hợp để load vào device.

**Configuration descriptor:**

Một đặc tả thiết bị có thể có một hoặc nhiều configuration descriptor. Mỗi một trong những configuration descriptor này sẽ định nghĩ cách thiết bị được cấp nguồn (thông qua host hay tự cấp nguồn), năng lượng tiêu thụ tối đa và giao thức cho mỗi setup. Host có thể chọn giữa việc chỉ đọc configuration descriptor hay toàn bộ thông tin một lúc.

**Interface descriptor:**

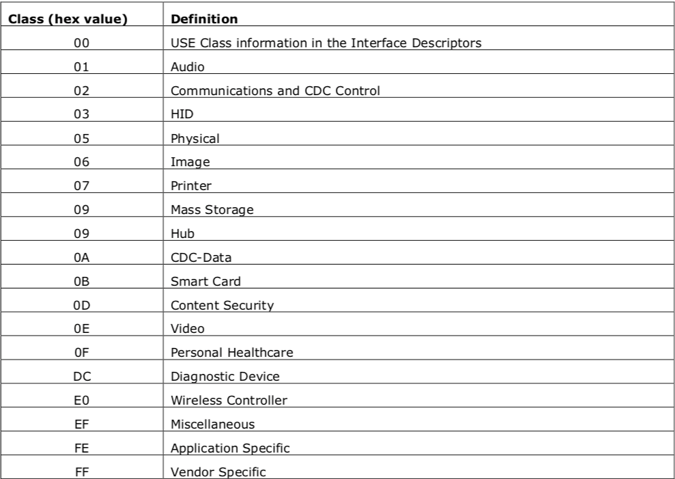
Configuration descriptor chứa một hoặc hiều interface descriptor. Mỗi số giao thức có thể chia thành nhiều interface khác nhau có giúp chỉnh sửa thuộc tính của thiết bị. Thiết bị host chọn một interface cụ thể nào đó phụ thuộc vào những function nó muốn truy cập. Interface cũng có những lớp thông tin mà host có thể sử dụng để xác định xem driver nào nó có thể sử dụng.

**Endpoint descriptor**

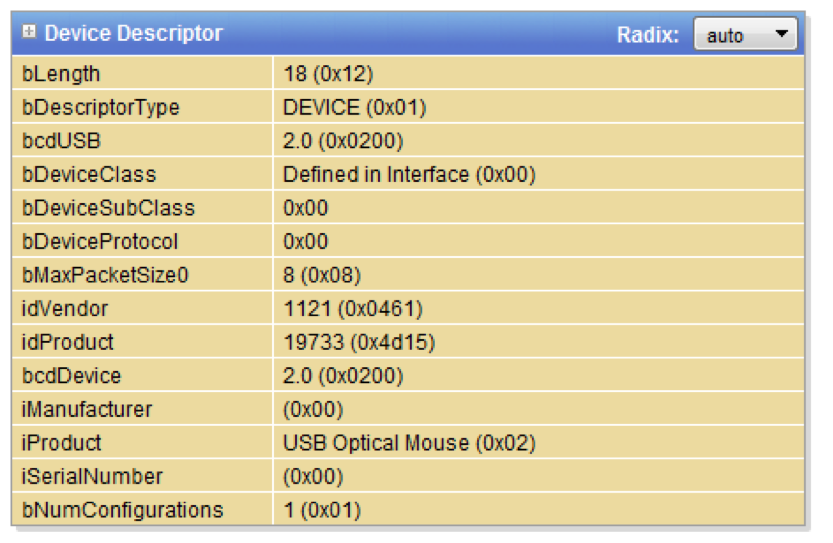
Interface descriptor có thể chứa 1 hoặc nhiều endpoint. Mỗi đặc tả endpoint là đặc tả nhỏ nhất của trường đặc tả (hierarchy) và nó định nghĩa độ rộng băng thông yêu cầu, kiểu truyền, và huognws truyền của một endpoint. Đối với hướng truyền, nó có thể là in hoặc out.

**Loading USB driver**

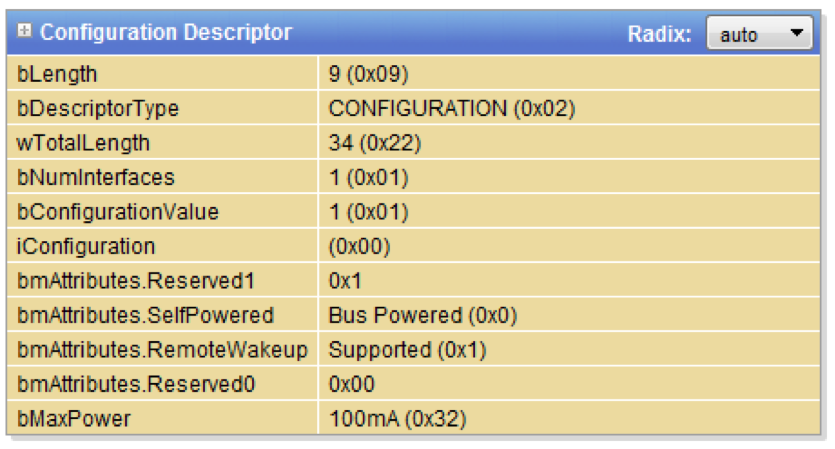
Một khi những đặc tả này được nhận diện, host sẽ load interface phù hợp để device có thể chạy được. File driver USB sẽ được cài sẵn trên host và cung cấp thông tin cho phép giao tiếp giữa các lớp thiết bị USB. Những driver này cung cấp thôn tin chi tiết để biên dịch đặc tả thiết bị và giao tiếp với thiết bị. Hầu hết thiết bị USB là từ những spec được định nghĩa trước và có thể tự động kết nối kể cả khi người dùng không biết. Tuy nhiện, với những USB custom, người dùng phải tự tải driver từ developer mới có thể giao tiếp được với thiết bị USB.



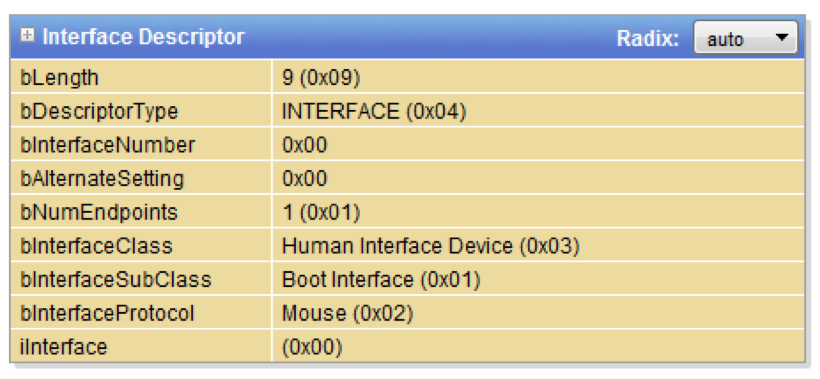
Enumeration là tối quan trọng. Bởi vì USB có quá nhiều kiểu thiết bị nên host cần biết cách để biết được cách giao tiếp hợp lý với mỗi device. Quá trình enumeratinon cho phép đảm bảo rằng mỗi thiết bị USB được kết nối sẽ được nhận diện bởi host để đảm bảo việc transfer data. Nếu không có quá trình enumeration, host không thể định nghĩa ra được kiểu thiết bị và kiểu truyền nào được sử dụng, khoảng thời gian truyền nhận data, và thậm chí tốc độ của đường truyền.



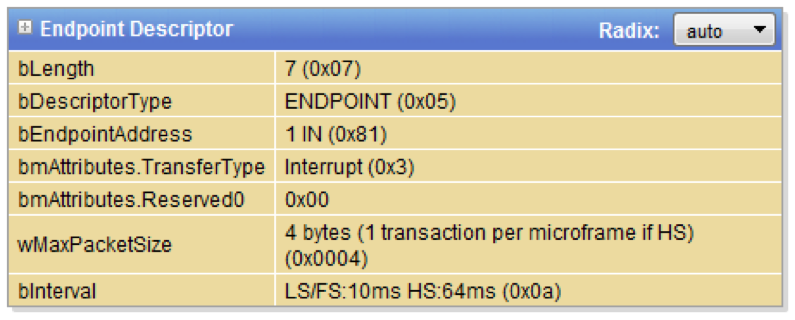
Device Descriptor in the Data Center Software



Configuration Descriptor in the Data Center Software



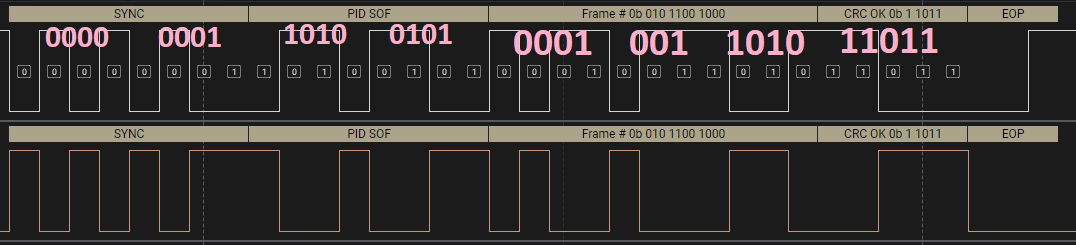
Interface Descriptor in the Data Center Software



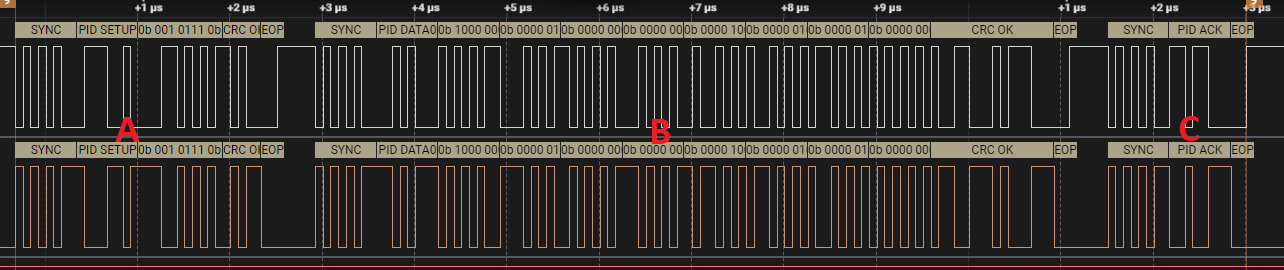
Endpoint Descriptor in the Data Center Software

Pulse Protocol

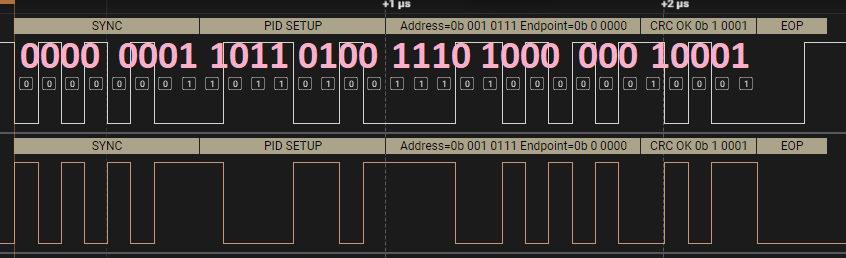
**Sync (length = 5):**

****

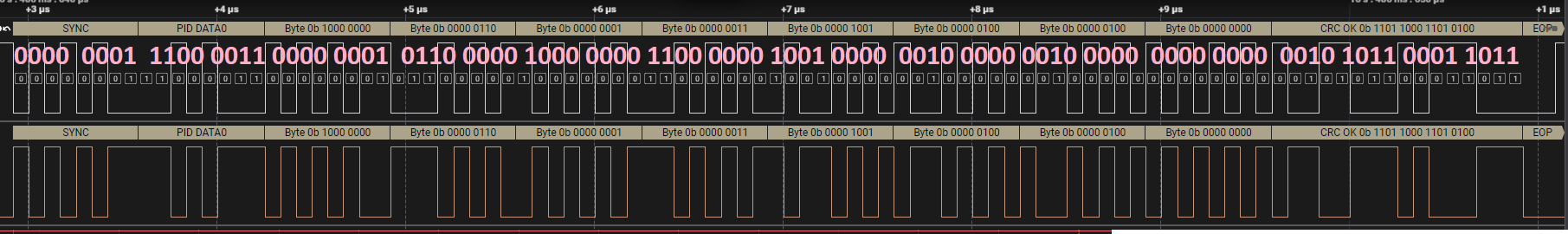
* 1. **PID Setup**

****

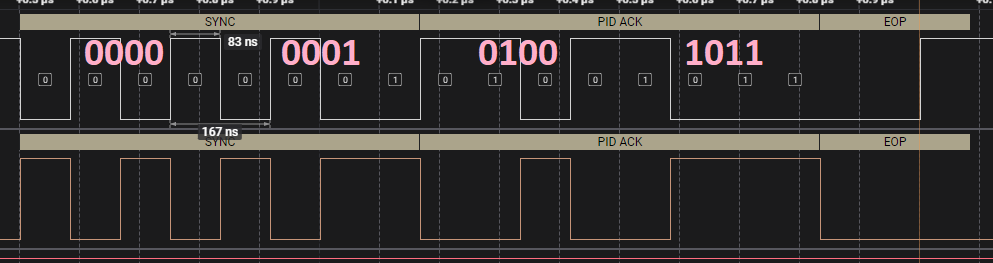
**A:**

****

**B:**

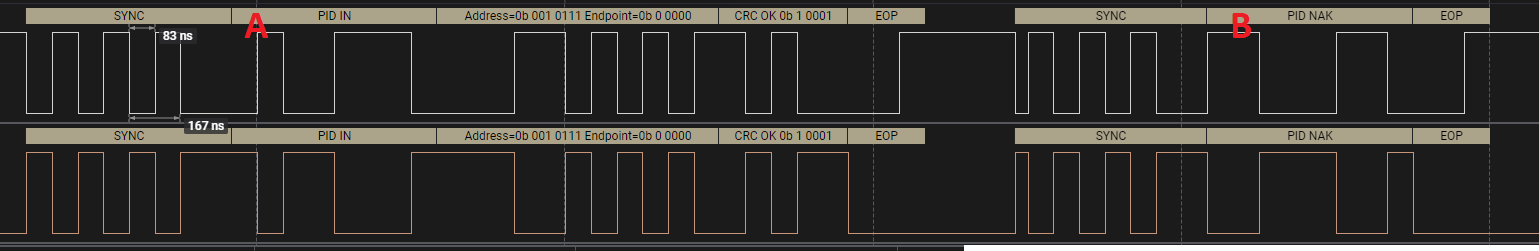
****

**C:**

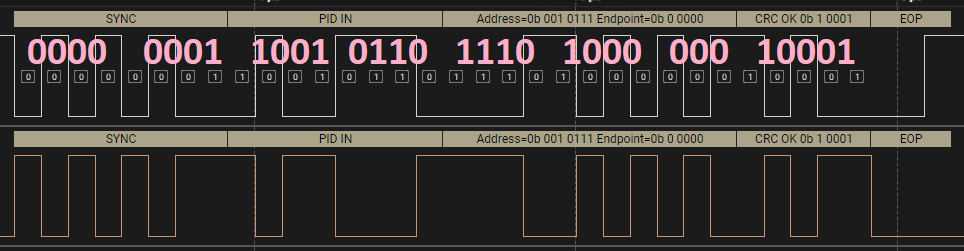
****

**2: PID IN**

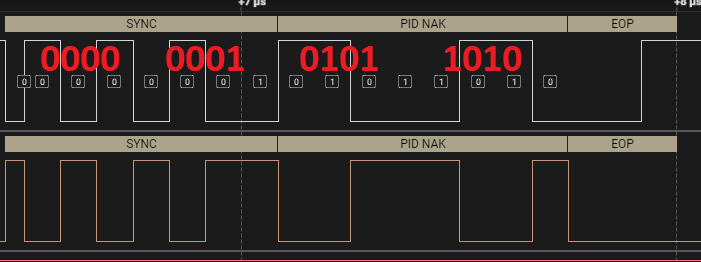
**TH1:**

****

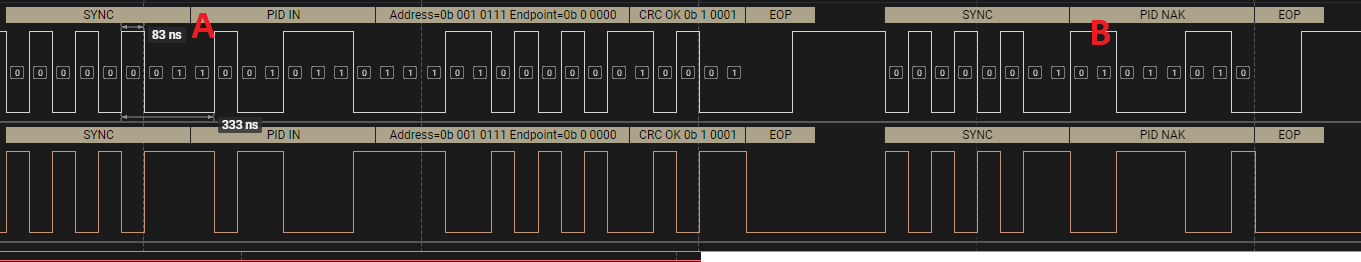
**A:**

****

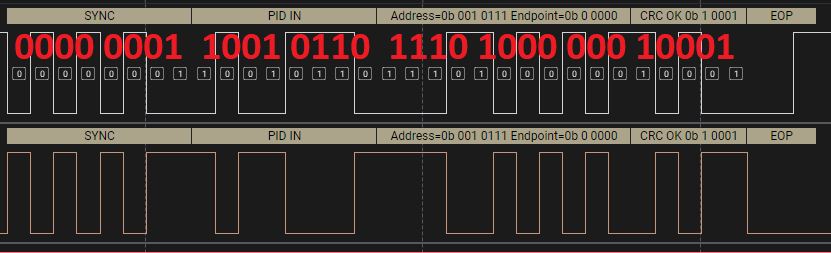
**B:**

****

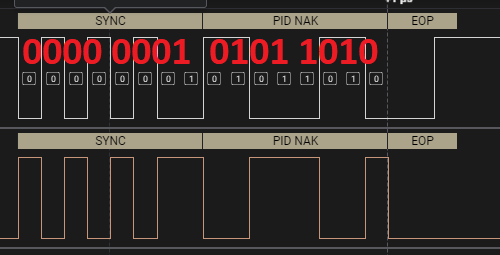
**TH2:**

****

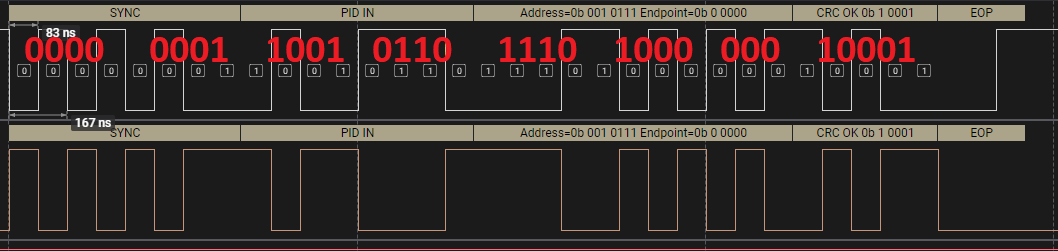
**A:**

****

**B:**

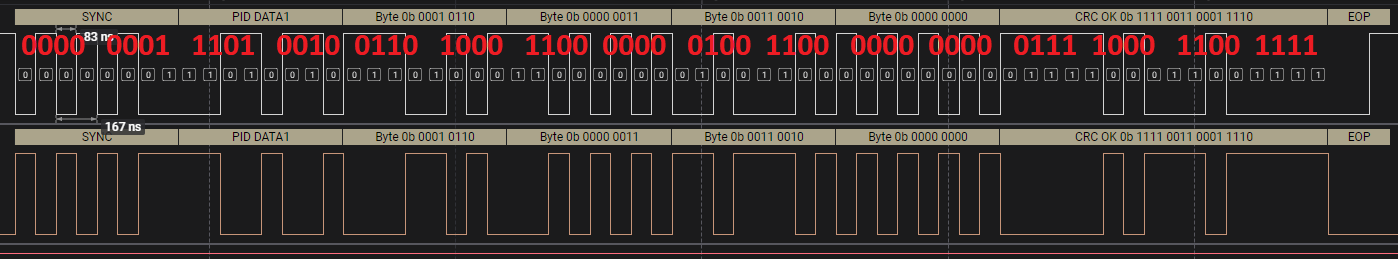
****

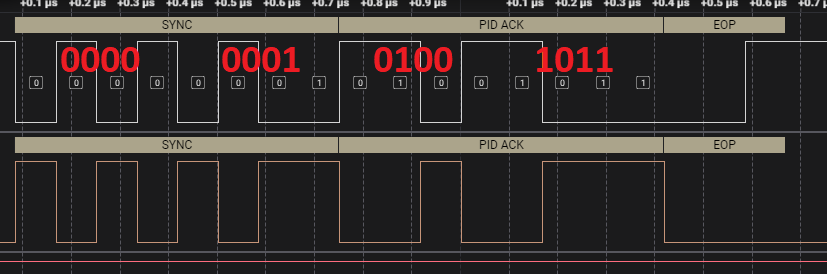
**TH3:**

****

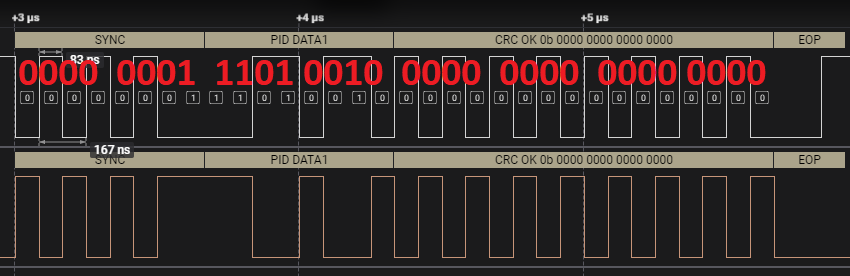
**3: PID DATA**

**TH1**

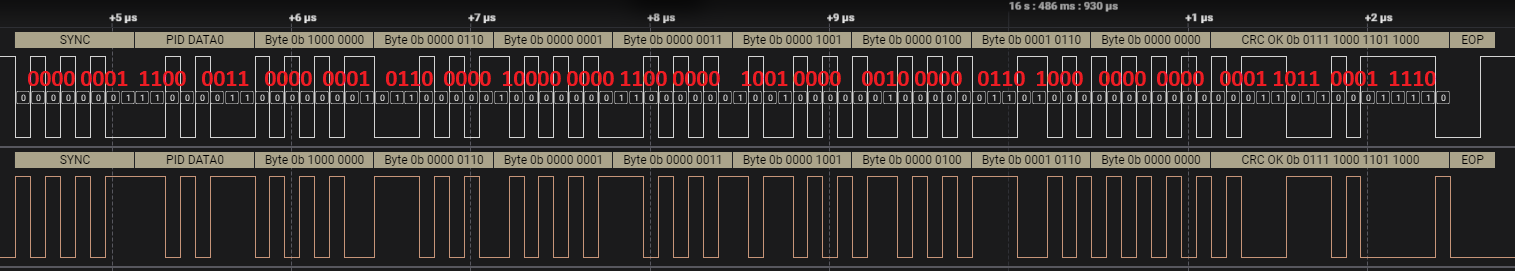
****

****

**TH2**

****

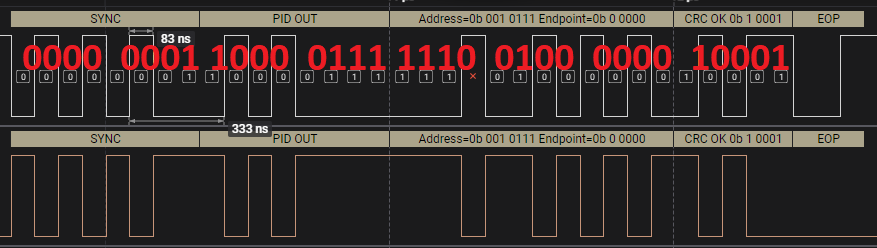
**TH3:**

****

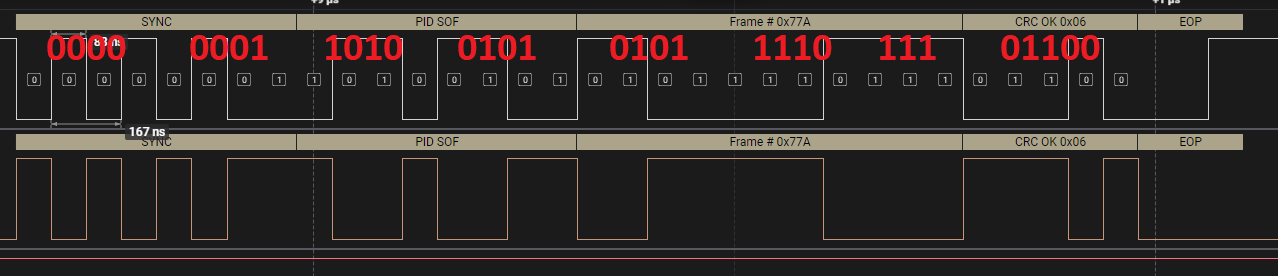
**TH4:**

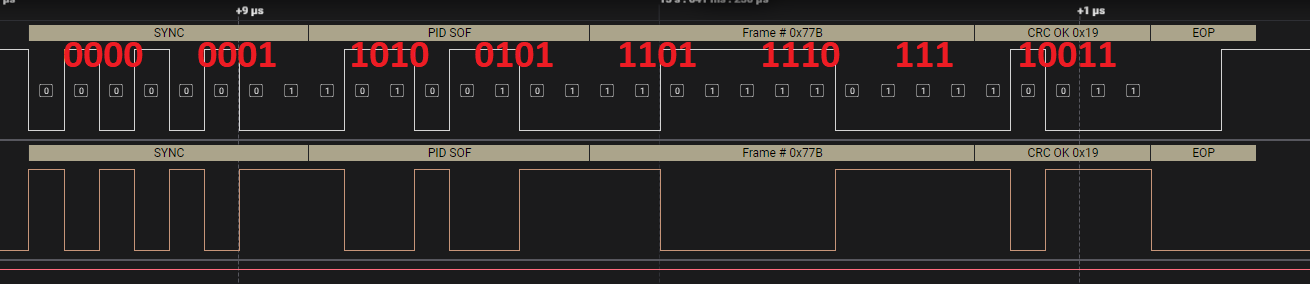
****

**4: PID OUT**

****

**5: PID SOF**

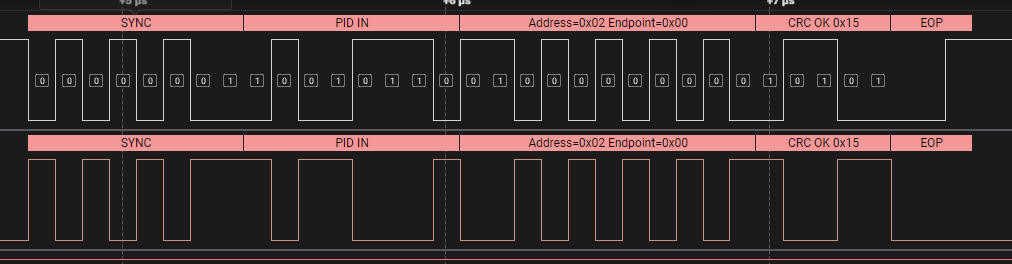
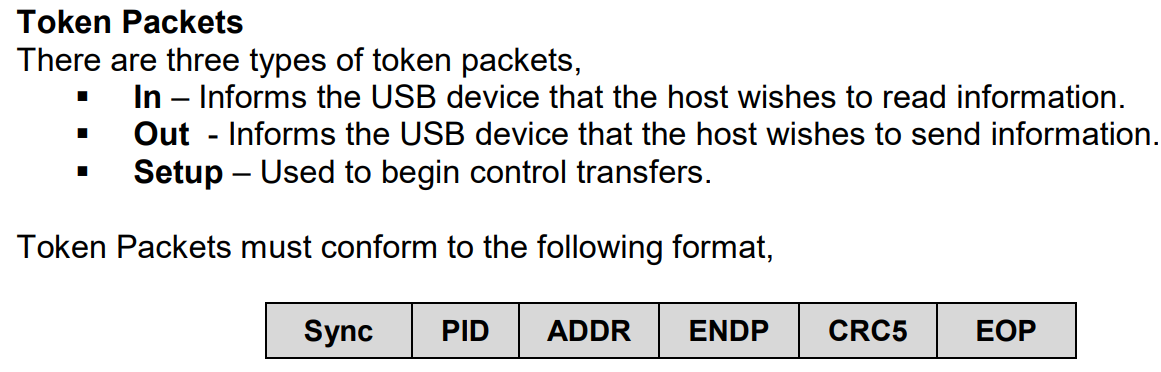
****

****

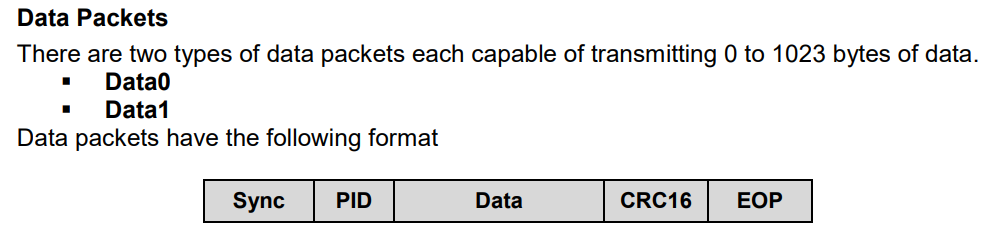
**State (specified for Fullspeed)**

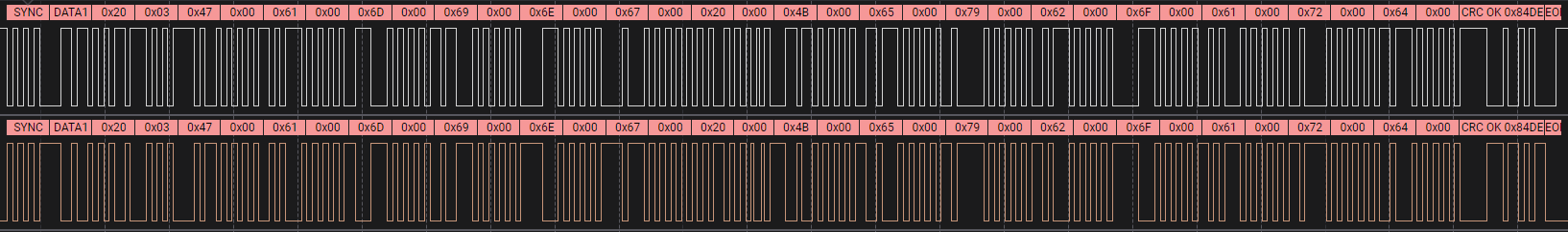
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Signaling** | **D+** | **D-** | **Interval** |
| **J state** | **Differntial 1** | **0** | **1** |  |
| **K state** | **Differntial 0** |  |  |  |
| **Reset (SE0)** |  | **0** | **0** | **>10ms** |
| **SE1** |  | **1** | **1** |  |
| **Idle** |  | **1** | **0** |  |
| **Resume state** | **K state** |  |  |  |
| **SOP** | **Idle -> K** |  |  |  |
| **EOP** | **SE0 for 2 bit + 1 J bit** |  |  |  |

* + 1. **Token packet**

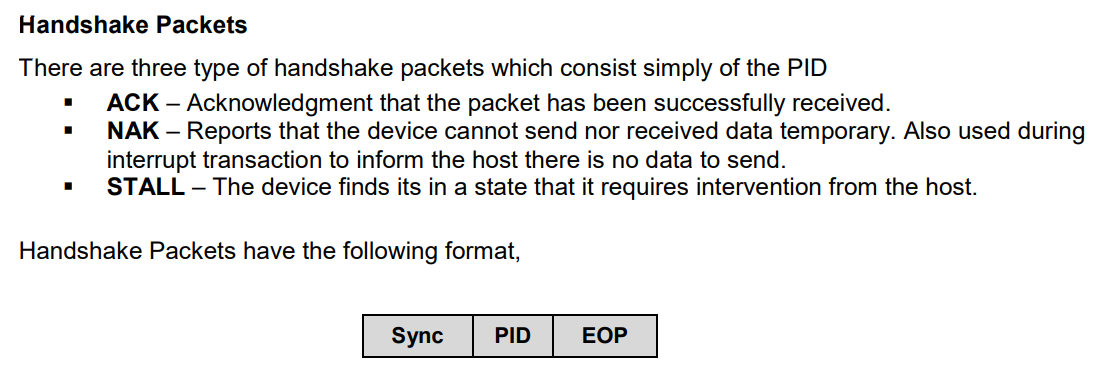
****

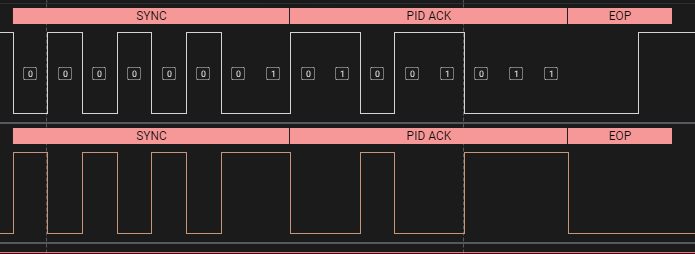
* + 1. **Data packet**

****

****

* + 1. **Handshake packet**





* + 1. **Start of frame**

