

# Xe buýt nối tiếp vạn năng

Từ OSDev Wiki

Universal Serial Bus được giới thiệu lần đầu tiên vào năm 1994 với mục đích thay thế các giao diện chuyên dụng khác nhau và để đơn giản hóa cấu hình của các thiết bị truyền thông. Ngành truyền thông không phát triển như USB-IF dự đoán, nhưng các chế độ truyền khác nhau mà USB giới thiệu đã cho phép nó trở thành một trong những tiêu chuẩn phổ biến nhất được sử dụng ngày nay. Hầu như mọi máy tính hiện đại đều hỗ trợ USB.

## nội dung

- 1 Giới thiệu
  - 1.1 Nội dung của văn bản này
  - 1.2 Những gì văn bản này không bao gồm
- 2 bộ điều khiển máy chủ
  - 2.1 Bộ điều khiển máy chủ USB 1.0
  - 2.2 Bộ điều khiển máy chủ USB 2.0
  - 2.3 Bộ điều khiển máy chủ USB 3.0
- 3 khái niệm cơ bản và danh pháp
  - Hệ thống USB 3.1
    - 3.1.1 (Các) Thiết bị USB
      - 3.1.1.1 Chức năng
      - 3.1.1.2 Trung tâm
    - 3.1.2 Kết nối USB
    - 3.1.3 Máy chủ USB
  - 3.2 Luồng giao tiếp USB
    - 3.2.1 Điểm cuối thiết bị và số điểm cuối
    - 3.2.2 Điểm cuối Zero
    - 3.2.3 Đường ống
    - 3.2.4 Đường ống điều khiển mặc định
  - 3.3 Khái niệm cơ bản về truyền USB
    - 3.3.1 Chuyển điều khiển
      - 3.3.1.1 Kích thước tải trọng dữ liệu tối đa
      - 3.3.1.2 Lỗi đường truyền
        - 3.3.1.2.1 Quá nhiều dữ liệu
        - 3.3.1.2.2 Lỗi xe buýt
        - 3.3.1.2.3 Điều kiện dừng
    - 3.3.2 Truyền dữ liệu hàng loạt
      - 3.3.2.1 Kích thước tải trọng dữ liệu tối đa
      - 3.3.2.2 Lỗi đường truyền
    - 3.3.3 Truyền dữ liệu gián đoạn
      - 3.3.3.1 Kích thước tải trọng dữ liệu tối đa
      - 3.3.3.2 Lỗi đường truyền
    - 3.3.4 Truyền dữ liệu đẳng thời
      - 3.3.4.1 Kích thước tải trọng dữ liệu tối đa
      - 3.3.4.2 Lỗi đường truyền
- 4 khái niệm USB nâng cao
  - 4.1 Phân bố thời gian tiếp cận xe buýt
    - 4.1.1 Khung và Microframe
    - 4.1.2 Phân loại thời gian xe buýt
  - 4.2 Điểm cuối tốc độ cao, băng thông cao
  - 4.3 Hỗ trợ truyền đẳng thời
    - 4.3.1 Đồng bộ hóa
      - 4.3.1.1 Điểm cuối không đồng bộ
      - 4.3.1.2 Điểm cuối đồng bộ

- 4.3.1.3 Điểm cuối thích ứng
- 4.3.2 Xử lý lỗi
- 5 Giao thức USB
  - 5.1 Gói tin
    - 5.1.1 Trường ĐỒNG BỘ
    - 5.1.2 Trường định danh gói
    - 5.1.3 Trường địa chỉ
      - 5.1.3.1 Trường địa chỉ
      - 5.1.3.2 Trường điểm cuối
    - 5.1.4 Trường dữ liệu
    - 5.1.5 Kiểm tra dự phòng theo chu kỳ
  - 5.2 Bắt tay
    - 5.2.1 Gói bắt tay
      - 5.2.1.1 BACK
      - 5.2.1.2 NAK
      - 5.2.1.3 DỪNG LẠI
      - 5.2.1.4 KHÔNG
      - 5.2.1.5 LỖI
    - 5.2.2 Hoàn cảnh chức năng/máy chủ phản hồi
      - 5.2.2.1 Chức năng đáp ứng các giao dịch IN
      - 5.2.2.2 Phản hồi của máy chủ đối với các giao dịch IN
      - 5.2.2.3 Chức năng phản hồi các giao dịch OUT
      - 5.2.2.4 Chức năng phản hồi các giao dịch CÀI ĐẶT
  - 5.3 Giao thức giao dịch PING
  - 5.4 Đồng bộ hóa chuyển đổi dữ liệu
    - 5.4.1 Truyền thành công
    - 5.4.2 Truyền dữ liệu bị lỗi hoặc bị hỏng
    - 5.4.3 Bắt tay ACK không thành công hoặc bị hỏng
  - 5.5 Xem lại quá trình truyền qua USB
    - 5.5.1 Chuyển điều khiển
    - 5.5.2 Truyền số lượng lớn và gián đoạn
    - 5.5.3 Truyền đằng thời
    - 5.5.4 Truyền đồng bộ tốc độ cao, băng thông cao
- 6 Khung thiết bị USB
  - 6.1 Chức năng, Cấu hình, Giao diện và Điểm cuối
  - 6.2 Trạng thái thiết bị USB
  - 6.3 Khả năng đánh thức từ xa
  - 6.4 Bảng liệt kê thiết bị USB
  - 6.5 Yêu cầu thiết bị USB
  - 6.6 Yêu cầu tiêu chuẩn
    - 6.6.1 SET\_ADDRESS
    - 6.6.2 NHẬN\_DESCRIPTOR
    - 6.6.3 BỘ\_MÔ\_TẢ
    - 6.6.4 NHẬN\_CẤU\_HÌNH
    - 6.6.5 SET\_CONFIGURATION
    - 6.6.6 GET\_INTERFACE
    - 6.6.7 SET\_INTERFACE
    - 6.6.8 CLEAR\_FEATURE
    - 6.6.9 SET\_FEATURE
    - 6.6.10 NHẬN\_STATUS
      - 6.6.10.1 Người nhận thiết bị
      - 6.6.10.2 Giao diện Người nhận
      - 6.6.10.3 Người nhận điểm cuối
    - 6.6.11 SYNCH\_FRAME
  - 6.7 Bộ mô tả USB tiêu chuẩn
    - 6.7.1 THIẾT\_BỊ
    - 6.7.2 THIẾT\_BỊ\_QUALIFIER
    - 6.7.3 CẤU\_HÌNH
    - 6.7.4 OTHER\_SPEED\_CONFIGURATION

- GIAO DIỆN 6.7.5
- 6.7.6 ĐIỂM KẾT THÚC
- 6.7.7 CHUỖI
- 7 Tô chức điển hình của phần mềm hệ thống
  - 7.1 Trình điều khiển thiết bị USB
  - Trình điều khiển USB 7.2
  - 7.3 Trình điều khiển trung tâm USB
  - 7.4 Trình điều khiển bộ điều khiển máy chủ
- 8 Xem thêm
  - 8.1 Liên kết ngoài

## Giới thiệu

Mặc dù hỗ trợ USB hấp dẫn như thế nào, thông số kỹ thuật USB 2.0 dày 650 trang vẫn có thể ngắn cản ngay cả một số người có sở thích năng động nhất (đặc biệt nếu tiếng Anh không phải là ngôn ngữ chính của họ). Thông số kỹ thuật của USB 2.0 không chỉ dài mà còn là điều kiện tiên quyết cho các thông số kỹ thuật XHCI, EHCI, UHCI và OHCI xác định giao diện OSes phần cứng thực tế. Hơn nữa, thông số kỹ thuật USB xác định rất nhiều thuật ngữ, một số được sử dụng thay thế cho nhau và có vẻ lười biếng; là một tài liệu kỹ thuật dài, việc lật đi lật lại để làm rõ một thuật ngữ hoặc khái niệm khó hiểu là điều không dễ dàng và cũng không thiết thực.

### Những gì văn bản này bao gồm

Sự thật là nhà phát triển phần mềm không cần phải đọc toàn bộ thông số kỹ thuật của USB 2.0; chẳng hạn, có những phần dành riêng cho nhà phát triển phần cứng. Thông tin được trình bày ở đây cố gắng tóm tắt các chương 4, 5 và 8 đến 10.

Chương 11 dành riêng cho các hub và cũng rất cần thiết để triển khai USB 2.0 đầy đủ, tuy nhiên, nó dài gần bằng các chương 4, 5, 8, 9 và 10 cộng lại và có thể được coi là tài liệu cho một chương trình cụ thể (mặc dù đặc biệt). ) loại thiết bị USB. Do đó, Chương 11 được đề cập trong mục wiki riêng của nó, USB Hubs. Mặc dù vậy, một số khái niệm liên quan đến bộ chia USB sẽ được thảo luận ngắn gọn nếu có liên quan trong bài viết này.

Lý tưởng nhất là văn bản ở đây sẽ thiết lập sự quen thuộc với các thuật ngữ và khái niệm mà nhà phát triển hệ điều hành sở thích cần để bắt đầu triển khai hỗ trợ USB và, nếu cần, dễ dàng phân tích cú pháp đặc tả USB mà không bị đe dọa bởi lượng thông tin. Ít nhất, người lập trình hệ thống nên giữ một bản sao đặc tả USB 2.0 để tham khảo khi làm việc với phần cứng liên quan đến USB.

May mắn thay, tất cả các tài liệu cần thiết đều có sẵn miễn phí (xem Liên kết (<http://wiki.osdev.org/USB#Links>) ).

### Những gì văn bản này không bao gồm

Xin lưu ý rằng USB, không giống như các tiêu chuẩn khác như VGA hoặc PCI, không liên quan đến giao diện phần cứng với bus hệ thống (và mở rộng ra là với hệ điều hành). Một giao diện như vậy được cung cấp bởi một hoặc nhiều bộ điều khiển máy chủ lưu trữ USB và được xác định bởi tài liệu thích hợp. Do đó, người ta không nên mong đợi văn bản này thảo luận về các chi tiết cụ thể hoặc mẫu mã (ví dụ: như người ta tìm thấy trong các mục wiki về VGA hoặc PCI) nêu chi tiết cách hệ điều hành khởi tạo và duy trì liên lạc với các thiết bị USB. Mặc dù thông tin như vậy có thể được tìm thấy trên các mục wiki thảo luận về Trình điều khiển bộ điều khiển máy chủ cụ thể, những mục wiki đó giả định rằng bạn đã hiểu các khái niệm và thuật ngữ được thảo luận ở đây.

## Bộ điều khiển máy chủ

**Bộ điều khiển máy chủ** là giao diện USB với hệ thống máy tính chủ. Nói cách khác, bộ điều khiển máy chủ là thứ mà phần mềm hệ thống sử dụng để giao tiếp với các thiết bị USB.

## Bộ điều khiển máy chủ USB 1.0

*Bài chi tiết:* Giao diện Universal Host Controller  
*Bài chi tiết:* Giao diện Bộ điều khiển Máy chủ Mở

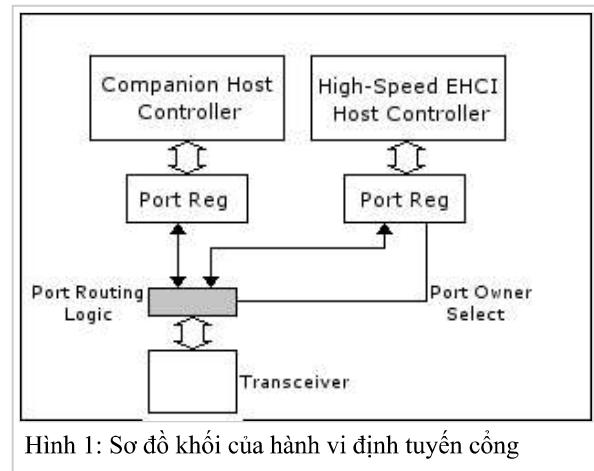
Intel đã đưa USB 1.0 ra thị trường với **Giao diện Bộ điều khiển Máy chủ Đa năng ( UHCI )**, trong khi Compaq, Microsoft và National Semiconductors cũng làm điều tương tự với **Giao diện Bộ điều khiển Máy chủ Mở ( OHCI )** của họ. Tuy nhiên, hai giao diện không tương thích và để mọi thứ trở nên tồi tệ hơn, VIA Technologies đã cấp phép cho tiêu chuẩn UHCI của Intel, do đó đảm bảo rằng cả hai tiêu chuẩn đều tồn tại. Thông thường, một bộ chip tích hợp sẽ chứa triển khai UHCI, trong khi thẻ ngoại vi thường triển khai tiêu chuẩn OHCI (nhưng điều này không có nghĩa là đảm bảo).

## Bộ điều khiển máy chủ USB 2.0

*Bài chi tiết:* Giao diện điều khiển máy chủ nâng cao

Khi thiết kế USB 2.0, USB-IF nhấn mạnh vào một triển khai duy nhất. **Việc triển khai duy nhất đó là Giao diện Bộ điều khiển Máy chủ Nâng cao ( EHCI )** của Intel. Tuy nhiên, mặc dù thông số kỹ thuật USB 2.0 yêu cầu giao diện USB 2.0 hỗ trợ các thiết bị USB 1.0, nhưng điều này không có nghĩa là EHCI phải hỗ trợ các thiết bị USB 1.0 và thực tế là không. Mỗi bộ điều khiển máy chủ EHCI được đi kèm với (thường là một vài) bộ điều khiển máy chủ UHCI và/hoặc OHCI. Khi thiết bị USB 1.0 được gắn vào, EHCI chỉ cần trao quyền điều khiển cho **bộ điều khiển đi kèm**. Tham khảo hình 1 để biết cách triển khai sơ đồ khối đơn giản cho hành vi này. Do đó, lập trình viên hệ thống phải hỗ trợ cả ba tiêu chuẩn để hỗ trợ USB 2.0.

Bộ điều khiển máy chủ EHCI chỉ xử lý các thiết bị USB 1.0 nếu chúng được gắn gián tiếp thông qua cổng USB 2.0. Chi tiết cụ thể về việc xử lý các thiết bị USB 1.0 được gắn vào bộ chia USB 2.0 được thảo luận và minh họa ngắn gọn trong phần bộ chia và chi tiết hơn trong mục wiki dành cho Bộ chia USB. Lưu ý rằng một số chipset mới hơn như chipset sê-ri 5 của Intel hoàn toàn không có bộ điều khiển đồng hành và thay vào đó có các trung tâm "khớp tốc độ" bên trong mà tất cả các thiết bị USB đều đi qua.



Hình 1: Sơ đồ khối của hành vi định tuyến cổng

## Bộ điều khiển máy chủ USB 3.0

*Bài chi tiết:* Giao diện Bộ điều khiển Máy chủ eXtensible

Giống như USB 2.0 tiền nhiệm của nó, USB 3.0 chỉ có một thông số kỹ thuật của bộ điều khiển máy chủ: **Giao diện Bộ điều khiển Máy chủ eXtensible** của Intel. Tuy nhiên, không giống như EHCI tiền nhiệm của nó, bộ điều khiển xHCI có thể và thực hiện giao tiếp với các thiết bị USB 1.0 và 2.0 mà không cần sử dụng bộ điều khiển đi kèm. Ngay cả trên phần cứng ban đầu có cả bộ điều khiển EHCI và xHCI đi kèm (để các HĐH chưa hỗ trợ xHCI vẫn có thể sử dụng ít nhất một số thiết bị USB), các cổng được gắn vào bộ điều khiển EHCI thường có thể được "định tuyến lại" cho bộ điều khiển xHCI và bộ điều khiển EHCI bị tắt hoàn toàn.

Cũng không giống như các phiên bản tiền nhiệm, xHCI được thiết kế với một số mức độ *tương thích về phía trước*, để có thể thực hiện các sửa đổi đối với thông số kỹ thuật USB mà không cần thiết kế giao diện bộ điều khiển máy chủ mới (ví dụ: USB 3.1 và 3.2 thêm tốc độ mới, chỉ với các cập nhật nhỏ cho đặc điểm kỹ thuật phù hợp với chúng.) Thật không may, điều này có nghĩa là xHCI chỉ có nét tương đồng thoáng qua với các bộ điều khiển ra đời trước nó và gây khó khăn cho việc viết trình điều khiển.

## Các khái niệm và danh pháp cơ bản

USB là một bus được thăm dò ý kiến, nghĩa là bộ điều khiển máy chủ phải bắt đầu tất cả các lần chuyển. Để giảm thiểu lần điều này có nghĩa là phần mềm hệ thống phải thăm dò USB. Bộ điều khiển máy chủ đảm nhiệm việc bỏ phiếu cho xe buýt và có thể được lập trình để đưa ra các ngắt đối với HĐH bắt cứ khi nào xe buýt cần chú ý.

## Hệ thống USB

Hệ thống USB bao gồm ba phần riêng biệt: **(các) thiết bị USB**, **kết nối USB** và **máy chủ lưu trữ USB**. Hình 2 minh họa một Hệ thống USB.

### (Các) thiết bị USB

**Các thiết bị USB** được phân loại là một **trung tâm** hoặc một **chức năng** (đứng nhầm lẫn với một thủ tục chương trình). Các trung tâm cung cấp các điểm đánh kèm bổ sung, trong khi các chức năng cung cấp khả năng cho hệ thống. Một số thiết bị có thể thực hiện một số chức năng và một trung tâm nhúng trong một gói vật lý. Chúng được gọi là **thiết bị ghép**.

#### Chức năng

Tất cả các chức năng đều hiệu giao thức USB, đáp ứng các hoạt động tiêu chuẩn (ví dụ: cấu hình hoặc đặt lại) và mô tả các khả năng cho máy chủ lưu trữ USB.

Có bốn loại chức năng tốc độ:

- **Các chức năng siêu tốc** hoạt động với tốc độ lên tới 5 Gb/s.
- **Các chức năng tốc độ cao** hoạt động với tốc độ lên tới 480 Mb/giây.
- **Các chức năng tốc độ tối đa** hoạt động với tốc độ lên tới 12 Mb/giây.
- **Các chức năng tốc độ thấp** hoạt động với tốc độ lên tới 1,5 Mb/giây.

Thông số kỹ thuật ban đầu của USB đã xác định các thiết bị tốc độ thấp và tốc độ tối đa, trong khi USB 2.0 bổ sung các thiết bị tốc độ cao và USB 3.0 bổ sung các thiết bị siêu tốc độ.

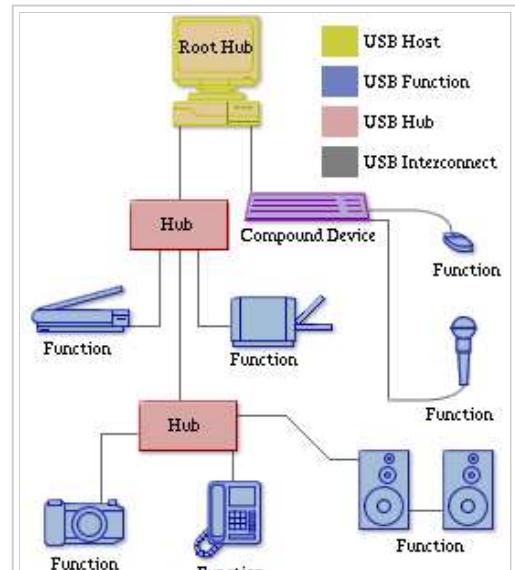
#### trung tâm

Trong một hệ thống tốc độ cao, một trung tâm tốc độ cao đóng một vai trò đặc biệt. Vì trung tâm tốc độ cao thiết lập tốc độ truyền tốc độ cao với máy chủ, nên nó phải cách ly mọi tín hiệu tốc độ đầy đủ hoặc tốc độ thấp khỏi cả máy chủ và mọi thiết bị tốc độ cao đi kèm.

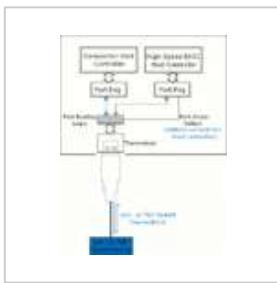
Để hiểu rõ hơn, hãy xem xét rằng bộ điều khiển EHCI được đi kèm với một hoặc nhiều bộ điều khiển đồng hành, như được minh họa trong hình 1 ở trên. Khi một thiết bị tốc độ đầy đủ hoặc tốc độ thấp được gắn trực tiếp vào trung tâm gốc, bộ điều khiển EHCI có thể từ bỏ quyền sở hữu cổng cụ thể đó cho bộ điều khiển đồng hành như minh họa trong hình 3. Tuy nhiên, nếu trung tâm tốc độ cao được kết nối với một cổng, như trong Hình 4, thì bộ điều khiển EHCI phải giữ quyền sở hữu cổng vì đây là thiết bị tốc độ cao. Bây giờ, giả sử các thiết bị tốc độ cao khác được gắn vào trung tâm tốc độ cao trong hình 4; rõ ràng là bộ điều khiển EHCI vẫn duy trì điều khiển như trong hình 5.

Nhưng điều gì sẽ xảy ra khi một thiết bị tốc độ đầy đủ hoặc tốc độ thấp được kết nối với trung tâm tốc độ cao trong hình 5? Nếu bộ điều khiển EHCI từ bỏ quyền sở hữu cổng, các thiết bị tốc độ cao sẽ không thể hoạt động ở tốc độ cao nữa, như trong hình 6. Thay vào đó, bộ điều khiển máy chủ và trung tâm hỗ trợ một loại đặc biệt của giao dịch gọi là giao dịch tách. Một **giao dịch phân tách** chỉ liên quan đến bộ điều khiển máy chủ và một trung tâm tốc độ cao; nó trong suốt với mọi thiết bị. Sơ đồ sử dụng giao dịch phân tách này để hỗ trợ các thiết bị tốc độ thấp và tốc độ cao trên một trung tâm tốc độ cao được minh họa trong hình 7.

Lưu ý rằng một số chipset mới hơn như chipset sê-ri 5 của Intel hoàn toàn không có bộ điều khiển đồng hành và thay vào đó có các trung tâm "khớp tốc độ" bên trong mà tất cả các thiết bị USB đều đi qua.



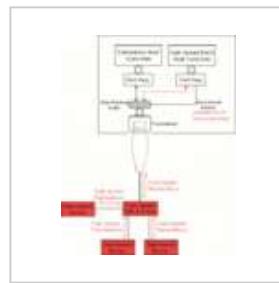
Hình 2: Minh họa hệ thống USB



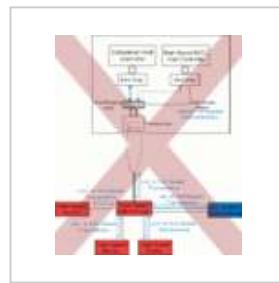
Hình 3: Thiết bị tốc độ thấp hoặc tốc độ tối đa được kết nối với cổng USB có khả năng tốc độ cao



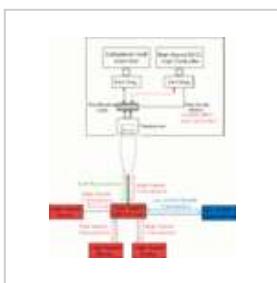
Hình 4: Hub tốc độ cao được kết nối với cổng USB có khả năng tốc độ cao



Hình 5: Các thiết bị tốc độ cao được kết nối với bộ chia tốc độ cao được kết nối với cổng USB tốc độ cao



Hình 6: Hình minh họa không chính xác về thiết bị tốc độ thấp và tốc độ tối đa trên xe buýt tốc độ cao



Hình 7: Minh họa chính xác về các giao dịch phân chia cho phép các thiết bị Tốc độ thấp và Tốc độ tối đa trên xe buýt tốc độ cao

## kết nối USB

Kết nối USB cung cấp kết nối từ (các) thiết bị USB đến máy chủ lưu trữ USB. Về mặt vật lý, kết nối USB là cấu trúc liên kết hình sao theo tầng. Cho phép tối đa bảy tầng và trung tâm gốc chiếm tầng đầu tiên. Vì các thiết bị phức hợp chứa một trung tâm nhúng, nên không thể gắn thiết bị ghép ở bậc 7. Hình 8 minh họa cấu trúc liên kết USB (lấy từ Hình 4-1 của thông số kỹ thuật USB 2.0).



Hình 8: Cấu trúc liên kết USB

## Máy chủ USB

Một hệ thống USB chỉ chứa một **USB host**. Máy chủ giao tiếp với kết nối USB thông qua bộ điều khiển máy chủ. Máy chủ bao gồm một trung tâm nhúng được gọi là **trung tâm gốc** cung cấp một hoặc nhiều **điểm đính kèm** hoặc **cổng**.

## Luồng giao tiếp USB

Hình 9 minh họa các khái niệm về luồng giao tiếp USB và được lấy từ Hình 5-10 của Thông số kỹ thuật USB 2.0.

## Điểm cuối thiết bị và số điểm cuối

Mỗi thiết bị USB chứa một tập hợp các điểm cuối. Mỗi điểm cuối có các đặc điểm sau:

- Tần suất truy cập xe buýt/yêu cầu độ trễ
- yêu cầu băng thông
- Mã định danh duy nhất do thiết bị xác định được gọi là số điểm cuối

- Các yêu cầu về hành vi xử lý lỗi
- Kích thước gói tối đa mà điểm cuối có thể gửi hoặc nhận
- Loại chuyển giao của điểm cuối
- Hướng truyền dữ liệu do thiết bị xác định:
  - **Đầu vào**: từ thiết bị đến máy chủ
  - **Đầu ra**: từ máy chủ đến thiết bị

Ví dụ, hãy xem xét một thiết bị máy in/máy quét “tất cả trong một”. Một thiết bị như vậy có thể triển khai số điểm cuối cho chức năng in và số điểm cuối riêng biệt cho chức năng quét.

Mặc dù các điểm cuối có một hướng cụ thể, hai điểm cuối có thể có cùng số điểm cuối nhưng ngược hướng truyền dữ liệu. Tất cả các chức năng thực hiện hai điểm cuối như vậy với số điểm cuối

là 0. Chỉ có thể truy cập các điểm cuối có số điểm cuối là 0 ngay sau khi thiết bị được cấp nguồn và đã nhận được thiết lập lại xe buýt; tất cả các điểm cuối khác ở trạng thái không xác định cho đến khi thiết bị được định cấu hình.

Bên cạnh hai điểm cuối bắt buộc, các chức năng có thể triển khai các điểm cuối bổ sung nếu cần, với những hạn chế sau:

- Các chức năng tốc độ thấp có thể triển khai tối đa hai điểm cuối bổ sung.
- Các thiết bị tốc độ cao và dày đặc có thể triển khai tối đa 15 điểm cuối đầu vào bổ sung và 15 điểm cuối đầu ra bổ sung. Đây là giới hạn vật lý của giao thức USB và được thảo luận trong Endpoint\_Field.

## điểm cuối bằng không

Tất cả các thiết bị USB đều triển khai các điểm cuối đầu vào và đầu ra với số điểm cuối là 0. Các điểm cuối này được gọi chung là **ống điều khiển mặc định**. Các điểm cuối có số điểm cuối 0 đặc biệt ở chỗ chúng có thể truy cập được bất cứ khi nào thiết bị được gắn, cấp nguồn và đã nhận được thiết lập lại xe buýt.

Vì lợi ích của khả năng tương thích ngược, tất cả các chức năng tốc độ cao phải hỗ trợ các điểm cuối này ngay cả khi được kết nối với một trung tâm hoạt động ở tốc độ tối đa. Điều này có nghĩa là các thiết bị tốc độ cao phải có khả năng đặt lại ở tốc độ tối đa, cũng như đáp ứng thành công các yêu cầu tiêu chuẩn ở tốc độ tối đa. Tuy nhiên, thiết bị tốc độ cao không bắt buộc phải hỗ trợ chức năng dự định của nó ở tốc độ tối đa. Điều này cho phép các hệ thống USB 1.0 xác định thiết bị USB 2.0 và cảnh báo người dùng nếu thiết bị không thể hoạt động bình thường ở tốc độ tối đa

## ống

Một đường ống liên kết phần mềm trên máy chủ (cụ thể là bộ đệm trên máy chủ) với một điểm cuối trên thiết bị.

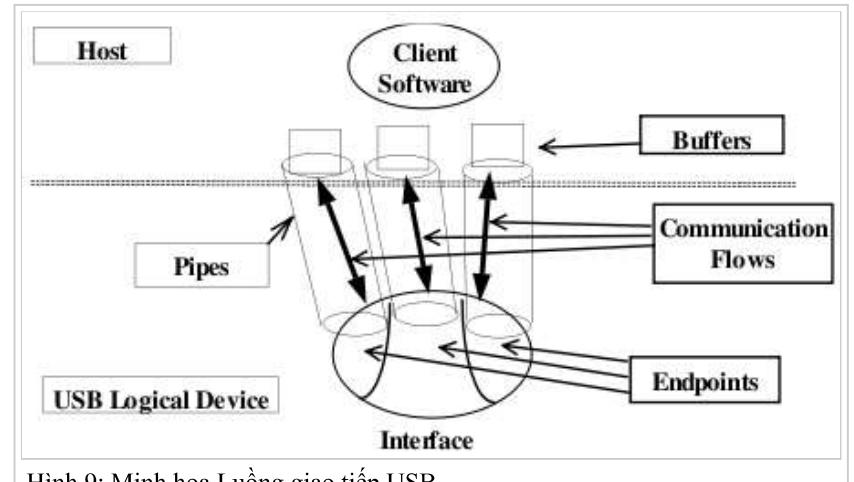
Có hai loại chế độ giao tiếp đường ống:

- **Các đường dẫn luồng** không áp đặt cấu trúc nào đối với dữ liệu được truyền. Các ống luồng luôn là **một hướng** trong luồng giao tiếp của chúng.
- **Đường dẫn thông báo** áp đặt một số cấu trúc đối với dữ liệu được truyền. Đường ống thông báo là **hai chiều**, tuy nhiên dữ liệu có thể chủ yếu truyền theo một hướng.

Ống cũng có các thuộc tính sau:

- Khiếu nại về truy cập xe buýt và sử dụng băng thông
- Một loại chuyển giao
- Các đặc điểm của điểm cuối được liên kết

Luồng dữ liệu trong một đường ống độc lập với luồng dữ liệu trong bất kỳ đường ống nào khác. Hầu hết các đường dẫn đều khả dụng sau khi thiết bị đã được định cấu hình, tuy nhiên, **đường dẫn điều khiển mặc định** luôn tồn tại sau khi thiết bị USB được cấp nguồn và đã nhận được thiết lập lại xe buýt.



Hình 9: Minh họa Luồng giao tiếp USB

## Ống điều khiển mặc định

Đường ống điều khiển mặc định là một loại đường ống thông báo đặc biệt luôn có thể truy cập được sau khi thiết bị được cấp nguồn và đã nhận được thiết lập lại xe buýt. Do đó, ống điều khiển mặc định cung cấp phương tiện để xác định và định cấu hình thiết bị sao cho các điểm cuối bổ sung, nếu có, được cung cấp.

Thông tin cần thiết để xác định hoàn toàn một thiết bị được liên kết với ống điều khiển mặc định; thông tin như vậy rơi vào các loại sau:

- **Thông tin tiêu chuẩn** là phổ biến trong tất cả các thiết bị USB.
- **Thông tin về loại** phụ thuộc vào loại của thiết bị USB, như được xác định bởi thông tin tiêu chuẩn.
- **Thông tin nhà cung cấp USB** được nhà cung cấp phần cứng sử dụng miễn phí.

## Khái niệm cơ bản về chuyển USB

Hầu hết các giao dịch USB bao gồm ba gói:

- Gói **mã thông báo** cho biết loại và hướng của giao dịch, địa chỉ thiết bị và số điểm cuối.
- Tùy thuộc vào hướng của giao dịch, máy chủ hoặc chức năng sẽ gửi một **gói dữ liệu** (có thể chỉ đơn giản là không có dữ liệu để gửi).
- Thiết bị nhận phản hồi bằng **gói bắt tay** để cho biết liệu quá trình truyền có thành công hay không.

USB hỗ trợ bốn kiểu truyền dữ liệu cơ bản diễn ra qua đường ống. Một đường ống duy nhất chỉ hỗ trợ (và chính xác) một loại truyền cho bất kỳ cấu hình thiết bị cụ thể nào. Nghĩa là, một chức năng có thể cung cấp phương tiện để thay đổi loại truyền của số điểm cuối do thiết bị triển khai.

Tóm lại, bốn loại chuyển giao cơ bản là:

- **Truyền điều khiển** cung cấp truyền không mất dữ liệu và được sử dụng để định cấu hình thiết bị. Do đó, tất cả các thiết bị USB ít nhất phải hỗ trợ truyền điều khiển thông qua ống điều khiển mặc định.
- **Truyền dữ liệu hàng loạt** cung cấp khả năng truyền tuần tự, không mất dữ liệu và thường được sử dụng để truyền một lượng lớn dữ liệu.
- **Truyền dữ liệu gián đoạn** cung cấp khả năng truyền đáng tin cậy, có độ trễ hạn chế thường cần cho các thiết bị đầu vào của con người như chuột hoặc cần điều khiển.
- **Truyền dữ liệu đồng thời**, còn được gọi là **Truyền trực tuyến trong thời gian thực**, thương lượng băng thông và độ trễ cần thiết khi bắt đầu truyền. Loại truyền tải này chủ yếu được sử dụng cho các ứng dụng như truyền phát âm thanh. Vì tốc độ phân phối dữ liệu được coi là quan trọng hơn tính toàn vẹn của dữ liệu đối với kiểu truyền này nên nó không cung cấp bất kỳ loại cơ chế kiểm tra hoặc sửa lỗi nào.

## Kiểm soát chuyển giao

Truyền điều khiển hỗ trợ luồng giao tiếp loại cấu hình/lệnh/trạng thái. Máy chủ bắt đầu chuyển điều khiển với giao dịch bus SETUP tới chức năng, chức năng này thiết lập các chi tiết về việc truyền dữ liệu dự kiến, chẳng hạn như liệu máy chủ có muốn gửi hoặc nhận dữ liệu hay không. Tiếp theo, không hoặc nhiều giao dịch DATA diễn ra theo hướng thích hợp. Cuối cùng, một giao dịch TÌNH TRẠNG từ chức năng đến máy chủ cho biết liệu quá trình truyền có thành công hay không.

Rõ ràng, việc truyền điều khiển tuân thủ cấu trúc do USB xác định, do đó, không có gì ngạc nhiên khi việc truyền điều khiển chỉ có thể được thực hiện thông qua các đường ống thông báo.

Cả chức năng và máy chủ lưu trữ đều không được đảm bảo bất kỳ độ trễ hoặc băng thông cụ thể nào để truyền điều khiển.

### Kích thước tải trọng dữ liệu tối đa

Điểm cuối được sử dụng để truyền điều khiển chỉ định **kích thước tải trọng dữ liệu tối đa** mà nó có thể chấp nhận hoặc truyền tới xe buýt. Kích thước tải trọng dữ liệu tối đa được phép phụ thuộc vào tốc độ của thiết bị:

- Điểm cuối của thiết bị tốc độ cao chỉ có thể chọn kích thước tải trọng dữ liệu tối đa là 64 byte.
- Điểm cuối của thiết bị tốc độ tối đa có thể chọn kích thước tải trọng dữ liệu tối đa là 8, 16, 32 hoặc 64 byte.

- Điểm cuối của thiết bị tốc độ thấp chỉ có thể chọn kích thước tải trọng dữ liệu tối đa là 8 byte.

Quá trình truyền kiểm soát luôn sử dụng kích thước tải trọng dữ liệu tối đa của nó cho tải trọng dữ liệu trừ khi tải trọng dữ liệu nhỏ hơn kích thước tải trọng dữ liệu tối đa. Nghĩa là, nếu một điểm cuối có kích thước tải trọng dữ liệu tối đa là 64 byte và một lần truyền điều khiển dự định truyền 100 byte, thì tải trọng dữ liệu đầu tiên phải chứa 64 byte trở xuống. 36 byte còn lại được chuyển trong tải trọng thứ hai và không cần phải đệm thành 64 byte. Khi máy chủ nhận được tải trọng dữ liệu ít hơn tải trọng dữ liệu tối đa, máy chủ lưu trữ có thể coi quá trình truyền đã hoàn tất.

Tải trọng dữ liệu của giao dịch CÀI ĐẶT luôn là 8 byte và do đó có thể nhận được bởi điểm cuối của bất kỳ thiết bị USB nào. Do đó, máy chủ có thể truy vấn bộ mô tả thích hợp từ một thiết bị tốc độ tối đa mới được gắn trong quá trình định cấu hình để xác định kích thước tải trọng dữ liệu tối đa cho bất kỳ điểm cuối nào; sau đó máy chủ có thể tuân theo mức tối đa đó cho bất kỳ lần truyền nào trong tương lai.

## Lỗi đường truyền

### Quá nhiều dữ liệu

Khi truyền từ máy chủ sang thiết bị, nếu máy chủ gửi nhiều dữ liệu hơn mức đã thương lượng trong quá trình SETUP giao dịch (nghĩa là thiết bị nhận được nhiều dữ liệu hơn mong đợi; cụ thể là máy chủ không chuyển sang giai đoạn TÌNH TRẠNG khi thiết bị mong đợi), thiết bị điểm cuối tạm dừng đường ống.

Khi truyền từ thiết bị sang máy chủ, nếu thiết bị gửi nhiều dữ liệu hơn mức đã thương lượng trong quá trình SETUP giao dịch (nghĩa là máy chủ nhận được tải trọng dữ liệu bổ sung hoặc tải trọng dữ liệu cuối cùng lớn hơn mức cần thiết), thì máy chủ coi đó là lỗi và hủy bỏ việc chuyển nhượng.

### Lỗi xe buýt

Trong trường hợp có lỗi hoặc bát thường về bus, điểm cuối có thể nhận được gói SETUP ở giữa quá trình truyền điều khiển. Trong trường hợp như vậy, điểm cuối phải hủy bỏ quá trình truyền hiện tại và xử lý gói SETUP không mong muốn. Hành vi này phải hoàn toàn minh bạch đối với máy chủ lưu trữ; máy chủ không nên mong đợi lợi dụng hành vi này.

### Điều kiện tạm dừng

Điểm cuối điều khiển có thể phục hồi từ tình trạng tạm dừng khi nhận được gói SETUP. Nếu điểm cuối không khôi phục từ gói SETUP, thì có thể cần khôi phục điểm cuối qua một đường dẫn khác. Nếu một điểm cuối có số điểm cuối 0 không phục hồi với gói SETUP, thì máy chủ nên đặt lại thiết bị.

## Truyền dữ liệu hàng loạt

Một đường ống có loại chuyển số lượng lớn cung cấp:

- Truy cập vào USB trên cơ sở băng thông khả dụng
- Thủ lại các lần chuyển mà thỉnh thoảng gấp sự cố gửi không thành công
- Đảm bảo toàn vẹn dữ liệu, nhưng không đảm bảo băng thông

Bộ điều khiển máy chủ cho phép truyền dữ liệu số lượng lớn với mức độ ưu tiên thấp; chúng thường chỉ được xử lý khi có sẵn băng thông, tuy nhiên, phần mềm có thể không cho rằng quá trình truyền kiểm soát sẽ được xử lý trước khi truyền hàng loạt. Nếu nhiều lần chuyển số lượng lớn đang chờ xử lý, bộ điều khiển máy chủ có thể bắt đầu chuyển các lần chuyển số lượng lớn qua xe buýt theo chính sách phụ thuộc vào việc triển khai. Phần mềm hệ thống có thể thay đổi thời gian xe buýt sẵn có để truyền số lượng lớn đến một điểm cuối cụ thể.

USB không áp đặt bất kỳ cấu trúc nào đối với nội dung dữ liệu của quá trình truyền số lượng lớn; do đó, chuyển số lượng lớn được thực hiện thông qua các đường ống dòng.

## Kích thước tải trọng dữ liệu tối đa

Điểm cuối được sử dụng để truyền dữ liệu hàng loạt chỉ định **kích thước tải trọng dữ liệu tối đa** mà nó có thể chấp nhận hoặc truyền tới xe buýt. Kích thước tải trọng dữ liệu tối đa được phép phụ thuộc vào tốc độ của thiết bị:

- Điểm cuối của thiết bị tốc độ cao chỉ có thể chọn kích thước tải trọng dữ liệu tối đa là 512 byte.
- Điểm cuối của thiết bị tốc độ tối đa có thể chọn kích thước tải trọng dữ liệu tối đa là 8, 16, 32 hoặc 64 byte.
- Các thiết bị tốc độ thấp có thể không triển khai các điểm cuối hàng loạt.

Giống như truyền điều khiển, điểm cuối truyền số lượng lớn phải truyền tải trọng dữ liệu có kích thước tải trọng dữ liệu tối đa cho điểm cuối đó ngoại trừ tải trọng dữ liệu cuối cùng trong một lần truyền cụ thể. Tải trọng dữ liệu cuối cùng không cần (và không nên) được thêm vào kích thước tải trọng dữ liệu tối đa.

Quá trình truyền số lượng lớn được coi là hoàn tất khi điểm cuối đã truyền chính xác lượng dữ liệu như mong đợi, điểm cuối truyền gói có kích thước tải trọng dữ liệu nhỏ hơn kích thước tải trọng dữ liệu tối đa của điểm cuối hoặc điểm cuối truyền gói có độ dài bằng không.

## Lỗi đường truyền

Nếu tải trọng dữ liệu được truyền lớn hơn dự kiến, quá trình truyền sẽ bị hủy bỏ cùng với bất kỳ quá trình truyền hàng loạt đang chờ xử lý nào qua cùng một đường ống.

Truyền dữ liệu hàng loạt sử dụng các bit chuyển đổi dữ liệu để vừa phát hiện lỗi vừa cung cấp sự đồng bộ hóa cần thiết để phục hồi sau lỗi. Nếu phát hiện thấy tình trạng tạm dừng, mọi hoạt động chuyển số lượng lớn còn lại sẽ bị hủy bỏ. Tình trạng tạm dừng được giải quyết bằng một ống điều khiển riêng.

## Làm gián đoạn quá trình truyền dữ liệu

Truyền dữ liệu bị gián đoạn đâm bảo thời gian phục vụ tối đa cho mọi hoạt động truyền dữ liệu. Ngay cả khi đường truyền bị lỗi, dữ liệu sẽ được truyền lại ở giai đoạn tiếp theo. Do đó, truyền dữ liệu gián đoạn là lý tưởng cho các thiết bị không gửi dữ liệu thường xuyên, nhưng khi thực hiện, chúng yêu cầu truyền kịp thời cũng như tính toàn vẹn của dữ liệu; hầu hết các thiết bị đầu vào của con người đều có những yêu cầu này.

Việc truyền dữ liệu gián đoạn được thực hiện bởi một ống dẫn luồng và do đó không cần phải tuân theo bất kỳ cấu trúc dữ liệu USB nào.

## Kích thước tải trọng dữ liệu tối đa

Điểm cuối được sử dụng để truyền dữ liệu gián đoạn chỉ định **kích thước tải trọng dữ liệu tối đa** mà nó có thể chấp nhận hoặc truyền tới xe buýt. Kích thước tải trọng dữ liệu tối đa được phép phụ thuộc vào tốc độ của thiết bị:

- Điểm cuối của thiết bị tốc độ cao có thể chọn kích thước tải trọng dữ liệu tối đa lên tới 1024 byte.
- Điểm cuối của thiết bị tốc độ tối đa có thể chọn kích thước tải trọng dữ liệu tối đa lên tới 64 byte.
- Điểm cuối của thiết bị tốc độ thấp có thể chọn kích thước tải trọng dữ liệu tối đa lên tới 8 byte.

Ngoài ra, điểm cuối tốc độ cao, băng thông cao có thể chỉ định rằng nó yêu cầu hai hoặc ba giao dịch trên mỗi vi khung. Các điểm cuối, khung và vi khung tốc độ cao, băng thông cao sẽ được thảo luận sau.

Lưu ý rằng kích thước tải trọng dữ liệu tối đa đối với truyền dữ liệu gián đoạn cho phép mức độ chi tiết cao hơn so với truyền dữ liệu điều khiển hoặc hàng loạt. Nghĩa là, điểm cuối truyền dữ liệu gián đoạn cho thiết bị tốc độ cao có thể là bất kỳ số nguyên nào từ 0 đến 1024. Kích thước tải trọng dữ liệu tối đa cho điểm cuối truyền dữ liệu gián đoạn không đổi trong suốt thời gian tồn tại của cấu hình thiết bị.

Giống như truyền dữ liệu điều khiển và hàng loạt, điểm cuối truyền gián đoạn phải truyền tải trọng dữ liệu có kích thước tải trọng dữ liệu tối đa cho điểm cuối đó ngoại trừ tải trọng dữ liệu cuối cùng trong một lần truyền cụ thể. Tải trọng dữ liệu cuối cùng không cần (và không nên) được thêm vào kích thước tải trọng dữ liệu tối đa.

Quá trình truyền gián đoạn được coi là hoàn tất khi điểm cuối đã truyền chính xác lượng dữ liệu như mong đợi, điểm cuối truyền gói có kích thước tải trọng dữ liệu nhỏ hơn kích thước tải trọng dữ liệu tối đa của điểm cuối hoặc điểm cuối truyền gói có độ dài bằng không.

### Lỗi đường truyền

Nếu tải trọng dữ liệu được truyền lớn hơn dự kiến, quá trình truyền sẽ bị hủy bỏ và đường ống sẽ tạm dừng mọi quá trình truyền gián đoạn trong tương lai cho đến khi lỗi được xác nhận và sửa chữa.

Truyền dữ liệu gián đoạn có thể sử dụng một trong hai sơ đồ bit chuyển đổi dữ liệu để đảm bảo truyền dữ liệu thành công. Các thiết bị yêu cầu thông lượng cao hơn có thể chọn chuyển đổi mọi đường truyền thay vì thực hiện bắt tay với máy chủ. Phương pháp này dễ bị lỗi hơn so với phương pháp chuyển đổi bit thay thế khi giao dịch thành công (sau khi bắt tay).

Nếu một điều kiện tạm dừng được phát hiện, bất kỳ quá trình truyền gián đoạn đang chờ xử lý nào cũng sẽ bị hủy bỏ. Tình trạng tạm dừng được giải quyết thông qua một đường ống điều khiển riêng biệt.

### Truyền dữ liệu đằng thời

Truyền dữ liệu đằng thời tương tự như truyền gián đoạn ở chỗ chúng đảm bảo thời gian phục vụ tối đa cho bất kỳ quá trình truyền nào, nhưng truyền dữ liệu đằng thời không đảm bảo tính toàn vẹn của dữ liệu. Khi dữ liệu đã sẵn sàng để được truyền đến hoặc từ một điểm cuối đằng thời, dữ liệu luôn được truyền ở tốc độ không đổi.

Dữ liệu được truyền qua một đường ống đằng thời không cần phải có bất kỳ cấu trúc cụ thể nào, do đó các đường ống đằng thời là các đường ống dòng.

#### Kích thước tải trọng dữ liệu tối đa

Điểm cuối được sử dụng để truyền dữ liệu đằng thời chỉ định **kích thước tải trọng dữ liệu tối đa** mà nó có thể chấp nhận hoặc truyền tới xe buýt. Kích thước tải trọng dữ liệu tối đa được phép phụ thuộc vào tốc độ của thiết bị:

- Điểm cuối của thiết bị tốc độ cao có thể chọn kích thước tải trọng dữ liệu tối đa lên tới 1024 byte.
- Điểm cuối của thiết bị tốc độ tối đa có thể chọn kích thước tải trọng dữ liệu tối đa lên tới 1023 byte.
- Các thiết bị tốc độ thấp có thể không triển khai các điểm cuối đằng thời.

Giống như các điểm cuối ngắt, các điểm cuối đằng thời có thể chỉ định kích thước tải trọng dữ liệu tối đa với độ chi tiết byte. Cũng giống như các điểm cuối ngắt, các điểm cuối đằng thời tốc độ cao, băng thông cao có thể chỉ định liệu chúng có yêu cầu hai hoặc ba giao dịch trên mỗi vi khung hay không.

Không giống như bất kỳ loại truyền nào khác, truyền đằng thời có thể truyền bất kỳ lượng dữ liệu nào lên đến kích thước tải trọng dữ liệu tối đa trong bất kỳ giao dịch nào.

### Lỗi đường truyền

Truyền đồng bộ dành cho các thiết bị có tốc độ truyền dữ liệu quan trọng hơn tính toàn vẹn của dữ liệu. Vì lý do đó, chuyển giao đằng thời không cho phép bắt tay và do đó không thể bị đình trệ. Điều quan trọng vẫn là tác nhân của quá trình truyền đằng thời biết liệu có xảy ra lỗi hay không và có thể có bao nhiêu dữ liệu bị mất. Giao thức USB cung cấp một số cơ chế để phát hiện lỗi truyền dữ liệu trong quá trình truyền đằng thời, các cơ chế này sẽ được thảo luận sau. Việc xác định lượng dữ liệu bị mất phụ thuộc vào việc triển khai. Tùy thuộc vào phần mềm trên máy chủ hoặc chương trình cơ sở trên chức năng thực hiện bất kỳ loại phát hiện/sửa sai dữ liệu nào.

## Khái niệm USB nâng cao

Các chủ đề trong phần này được xây dựng dựa trên các chủ đề đã thảo luận trước đó. Thông tin trong phần này cung cấp một số chi tiết cấp thấp hữu ích về hệ thống USB.

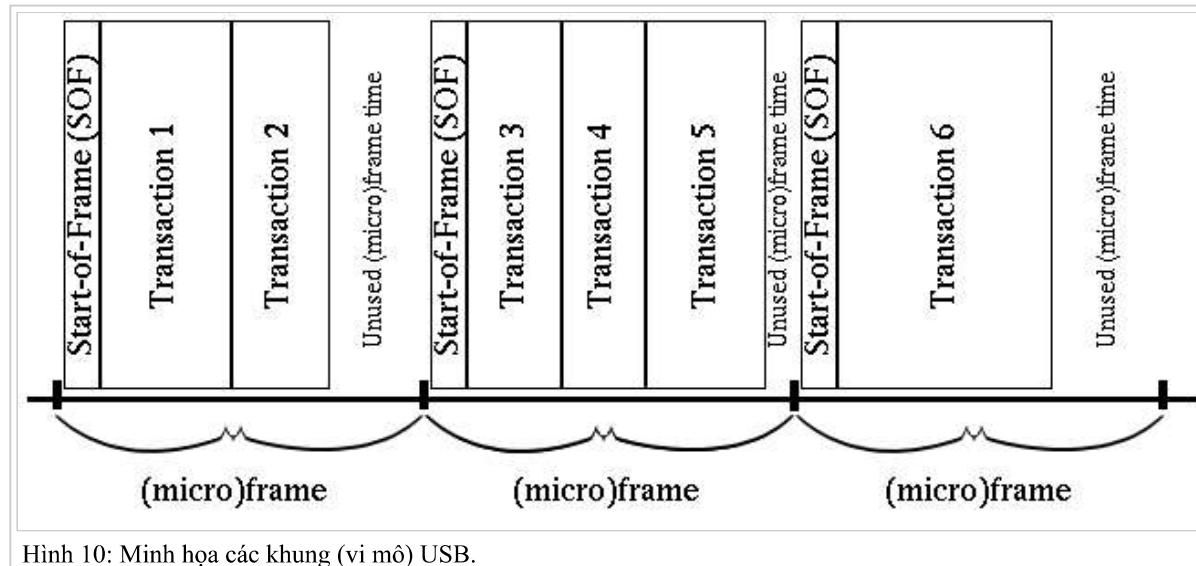
## Phân phối thời gian truy cập xe buýt

### Khung và Microframe

Để đảm bảo đồng bộ hóa giữa máy chủ và các chức năng, USB chia thời gian bus thành các đoạn có độ dài cố định. Đối với các bus tốc độ thấp hoặc tốc độ tối đa, USB chia thời gian bus thành các đơn vị 1 mili giây, được gọi là **các khung**. Đối với bus tốc độ cao, USB chia thời gian bus thành 125 đơn vị micro giây, được gọi là **microframe**.

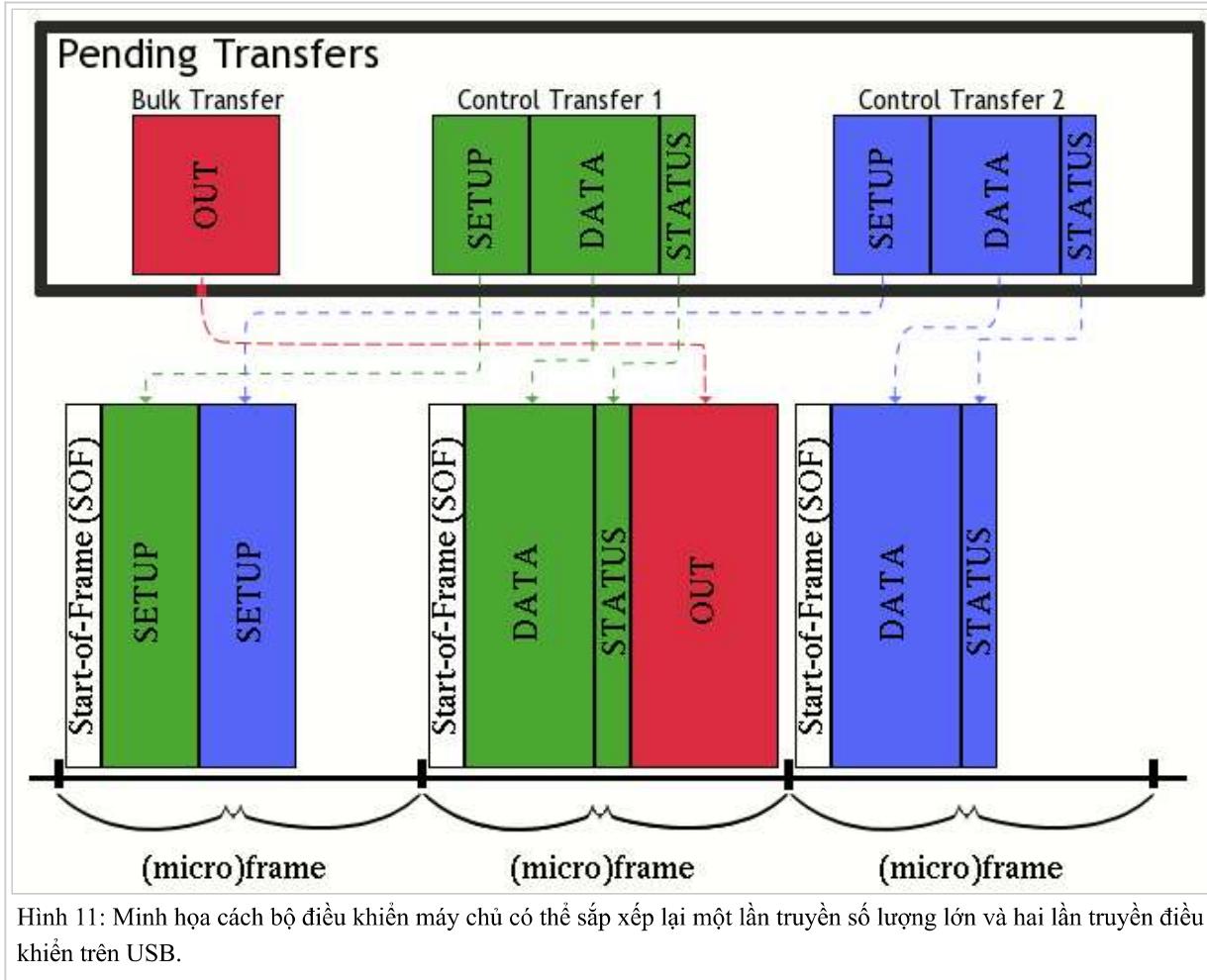
Lưu ý rằng các khung và vi khung không cùng tồn tại trên một xe buýt; các khung tốc độ thấp và tốc độ đầy đủ đã sử dụng các khung, nhưng khi phát triển một bus tốc độ cao, một khung ngắn hơn là cần thiết vì tốc độ bit báo hiệu cao hơn đáng kể sẽ nhạy cảm hơn với những thay đổi nhỏ hơn trong quá trình đồng bộ hóa giữa máy chủ và chức năng.

Các khung và vi khung chủ yếu là một chi tiết lớp vật lý và không nên nhầm lẫn với bất kỳ khái niệm nào trước đó. Các khung và vi khung không tương ứng với bất kỳ gói hoặc giao dịch nào; trên thực tế, một số giao dịch thường diễn ra trong một khung (vi mô). Bộ điều khiển máy chủ phát hành gói **bắt đầu khung (SOF)** ở đầu mỗi khung (vi mô). Phần còn lại của khung (vi mô) có sẵn để bộ điều khiển máy chủ thực hiện các giao dịch. Một giao dịch có thể không diễn ra nếu nó không thể được hoàn thành trong cùng một khung (vi mô) (vì nếu không thì gói SOF tiếp theo sẽ làm gián đoạn giao dịch).



Hình 10: Minh họa các khung (vi mô) USB.

Điều quan trọng là phải nhận ra rằng bộ điều khiển máy chủ có thể sắp xếp lại các giao dịch để tận dụng tốt hơn băng thông có sẵn. Tuy nhiên, hai giao dịch thông qua cùng một đường ống phải xảy ra theo đúng thứ tự, nhưng các giao dịch của hai lần chuyển riêng biệt có thể được sắp xếp lại theo quyết định của bộ điều khiển máy chủ. Xem xét một lần chuyển số lượng lớn đang chờ xử lý và hai lần chuyển kiểm soát đang chờ xử lý. Máy chủ có khả năng sắp xếp lại các lần truyền trên xe buýt như trong Hình 11.



Hình 11: Minh họa cách bộ điều khiển máy chủ có thể sắp xếp lại một lần truyền số lượng lớn và hai lần truyền điều khiển trên USB.

### Phân loại thời gian xe buýt

Có các quy tắc riêng cho việc phân bổ các khung trên một bus tốc độ đầy đủ/tốc độ thấp và để phân bổ các vi khung trên một bus tốc độ cao.

Đối với xe buýt tốc độ tối đa hoặc tốc độ thấp:

- Nếu truyền điều khiển yêu cầu ít hơn 10% khung hình, thời gian bus còn lại có thể được sử dụng để hỗ trợ truyền hàng loạt.
- Nếu có nhiều lần truyền điều khiển hơn thời gian dự trữ, nhưng thời gian khung bổ sung không được sử dụng bởi các lần truyền gián đoạn hoặc đồng thời, bộ điều khiển máy chủ có thể di chuyển các lần truyền điều khiển bổ sung lên xe buýt.
- Không quá 90% khung có thể được phân bổ cho các lần truyền định kỳ (không đồng bộ và gián đoạn).
- Máy chủ không được phát hành nhiều hơn 1 giao dịch trong một khung cho một điểm cuối đồng thời cụ thể.

Đối với xe buýt tốc độ cao:

- Nếu quá trình truyền điều khiển yêu cầu ít hơn 20% khung vi mô, thì thời gian bus còn lại có thể được sử dụng để hỗ trợ truyền hàng loạt.
- Nếu có nhiều lần truyền điều khiển hơn thời gian dự trữ, nhưng thời gian vi khung bổ sung không được sử dụng bởi các lần truyền gián đoạn hoặc đồng thời, thì bộ điều khiển máy chủ có thể di chuyển các lần truyền điều khiển bổ sung lên xe buýt.
- Không quá 80% khung có thể được phân bổ cho các lần truyền định kỳ (không đồng bộ và gián đoạn).
- Máy chủ không được phát hành nhiều hơn 1 giao dịch trong một vi khung đơn lẻ cho một điểm cuối đồng thời trừ khi đó là điểm cuối tốc độ cao, băng thông cao.
- Thời gian truy cập bus giao dịch chia nhỏ được phân bổ từ 80% vi khung được đảm bảo cho các lần chuyển định kỳ.

## Điểm cuối tốc độ cao, băng thông cao

Các điểm cuối ngắt tốc độ cao hoặc đẳng thời yêu cầu băng thông cao có thể chỉ định rằng chúng hỗ trợ tối đa ba giao dịch trong một khung (ví mô) đơn. Trong trường hợp này, tất cả trừ giao dịch cuối cùng trong một khung (ví mô) cụ thể phải có tải trọng dữ liệu có kích thước tải trọng dữ liệu tối đa cho điểm cuối đó.

Bộ điều khiển máy chủ không bao giờ thử lại giao dịch với điểm cuối đẳng thời. Nếu một giao dịch có điểm cuối ngắt tốc độ cao, băng thông cao không thành công, bộ điều khiển máy chủ có thể thử lại giao dịch trong cùng một khung (ví mô) nếu chưa đạt đến số lượng giao dịch tối đa trên mỗi (ví mô) khung. Nếu không, giao dịch sẽ được thử lại ở giai đoạn tiếp theo.

## Hỗ trợ truyền Isochronous

Hãy nhớ lại rằng quá trình truyền đẳng thời xảy ra trên các đường ống luồng, cung cấp quá trình truyền dữ liệu một chiều. Trên một trong các đường ống, được gọi là **nguồn**, dữ liệu được tạo ra và ở đầu kia, được gọi là **phản chìm**, dữ liệu được phân phôi.

Các thiết bị triển khai các điểm cuối đẳng thời yêu cầu dữ liệu được truyền từ nguồn tới điểm gửi với một tốc độ nhất định, đôi khi ở tải trọng lớn (ví dụ: truyền phát âm thanh hoặc video). Phần này thảo luận cách thức USB hoàn thành các yêu cầu này.

### đồng bộ hóa

Do tốc độ lấy mẫu dành riêng cho ứng dụng, thiết kế đồng hồ phần cứng khác nhau, chính sách lập lịch trình trong hệ điều hành hoặc thậm chí là các bất thường về vật lý, máy chủ và thiết bị đẳng thời có thể không đồng bộ hóa. Do đó, cần phải xem xét đặc biệt để duy trì đồng bộ hóa. Điểm cuối đẳng thời chỉ định một trong ba loại đồng bộ hóa.

#### Điểm cuối không đồng bộ

**Điểm cuối không đồng bộ** không có khả năng đồng bộ hóa với tần số gói SOF (thời gian 1ms cho điểm cuối tốc độ dày đặc, thời gian 125 micro giây cho điểm cuối tốc độ cao). Các điểm cuối này có: một tập hợp gồm một hoặc nhiều tốc độ lấy mẫu dữ liệu cố định hoặc tốc độ dữ liệu có thể lập trình liên tục. Thiết bị phải báo cáo khả năng lập trình của điểm cuối không đồng bộ theo cách nào đó (được xác định theo loại thiết bị thay vì theo thông số kỹ thuật của USB); nếu tốc độ dữ liệu có thể lập trình được thì máy chủ phải đặt tốc độ này trong quá trình khởi tạo điểm cuối đẳng thời.

Điểm cuối nguồn không đồng bộ ngũ ý tốc độ dữ liệu của chúng theo số lượng mẫu được tạo trên mỗi khung (ví mô). Điểm cuối chìm không đồng bộ phải cung cấp phản hồi rõ ràng cho điểm cuối nguồn. Khi điểm cuối nguồn là máy chủ, trình điều khiển thiết bị có trách nhiệm xử lý phản hồi rõ ràng đúng cách. Phản hồi này cho phép máy chủ và thiết bị thực hiện các điều chỉnh nhỏ đối với tốc độ dữ liệu để bù cho bất kỳ độ lệch đồng hồ nào.

#### Điểm cuối đồng bộ

**Các điểm cuối đồng bộ** phải đồng bộ hóa việc truyền dữ liệu của chúng với tần số gói SOF (thời gian 1ms đối với điểm cuối tốc độ tối đa, khoảng thời gian 125 micro giây đối với điểm cuối tốc độ cao). Các điểm cuối này có một tập hợp gồm một hoặc nhiều tốc độ lấy mẫu dữ liệu cố định hoặc tốc độ dữ liệu có thể lập trình liên tục. Thiết bị phải báo cáo khả năng lập trình của điểm cuối đồng bộ theo cách nào đó (được xác định theo loại thiết bị chứ không phải theo thông số kỹ thuật của USB); nếu tốc độ dữ liệu có thể lập trình được thì máy chủ phải đặt tốc độ này trong quá trình khởi tạo điểm cuối đẳng thời.

#### Điểm cuối thích ứng

**Các điểm cuối thích ứng** có thể tạo nguồn hoặc dữ liệu chìm ở bất kỳ tốc độ nào trong phạm vi hoạt động được chỉ định của chúng. Các điểm cuối này có thể có phạm vi hoạt động xoay quanh một tốc độ dữ liệu cụ thể, nó có thể có một tập hợp hữu hạn các phạm vi tốc độ dữ liệu hoặc có thể chọn giữa một số tốc độ dữ liệu có thể lập trình hoặc tự động phát hiện. Thiết bị phải báo cáo khả năng lập trình của điểm cuối thích ứng theo cách nào đó (được xác định theo loại thiết bị chứ không phải theo thông số kỹ thuật của USB); không giống như các loại đồng bộ hóa trước đây, các điểm cuối thích ứng có thể điều chỉnh tốc độ dữ liệu tức thời của nó trong quá trình hoạt động.

Điểm cuối chìm thích ứng cung cấp phản hồi rõ ràng cho nguồn giống như điểm cuối không đồng bộ.

## Xử lý lỗi

Bắt tay không được thực hiện cho các giao dịch đăng thời, do đó loại bỏ chi phí băng thông của các gói xác nhận. Không giống như các loại chuyển giao khác, các ứng dụng của điểm cuối đăng thời chịu trách nhiệm phát hiện và xử lý lỗi. Mặc dù việc tiếp tục phân phối dữ liệu truyền trực tuyến có thể quan trọng hơn là truyền lại gói dữ liệu bị bỏ lỡ, nhưng các ứng dụng của điểm cuối đăng thời thường vẫn cần biết rằng đã xảy ra lỗi trong luồng.

Giao thức USB làm nổi bật phương pháp khả thi sau đây để máy chủ hoặc thiết bị phát hiện lỗi trong luồng đăng thời:

- Các giao dịch đăng thời tốc độ cao, băng thông cao sử dụng trình tự PID dữ liệu (chuyển đổi bit dữ liệu), một phần chìm đăng thời có thể xác định rằng một gói dữ liệu đã bị bỏ lỡ khi nó nhận được một trình tự PID dữ liệu không hợp lệ.
- Cả bộ điều khiển máy chủ và thiết bị đều có thể nhìn thấy các gói SOF trên xe buýt. Nếu gói SOF được cấp cho khung (ví mô) dự kiến sẽ mang dữ liệu định kỳ của điểm cuối đăng thời, nhưng dữ liệu không được truyền đi, thì phần cứng có thể xác định rằng gói đã bị bỏ sót.
- Giao thức cung cấp bảo vệ CRC để đảm bảo rằng dữ liệu không bị hỏng.
- Nếu một điểm cuối nhìn thấy gói mã thông báo nhưng không thấy gói dữ liệu được liên kết trong khoảng thời gian chờ giao dịch xe buýt, thì gói dữ liệu đó không thể truyền được.

Sau khi ứng dụng biết rằng có lỗi trong luồng, ứng dụng sẽ quyết định hành động tiếp theo.

## Giao thức USB

### gói tin

Đơn vị nguyên tử của truyền dữ liệu là một gói. Một gói là một gói dữ liệu được tổ chức thường chứa ba phần tử:

- Thông tin kiểm soát (ví dụ: nguồn, đích, độ dài của dữ liệu)
- Dữ liệu dành riêng cho người dùng/ứng dụng
- Các bit phát hiện và sửa lỗi

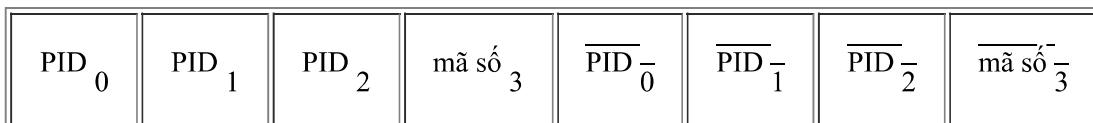
### Trường ĐỒNG BỘ

Trường SYNC bị bỏ qua trong sơ đồ gói trong thông số kỹ thuật của USB và thường có trong tài liệu khác về USB dành cho người lập trình. Ở đây tôi sẽ mô tả ngắn gọn ngữ nghĩa của trường SYNC đơn giản vì các nguồn USB thường tham chiếu đến trường SYNC, điều này có thể khiến người đọc nhầm lẫn. Tuy nhiên, để rõ ràng, lập trình viên hệ thống (và có lẽ hầu hết các nhà phát triển chương trình cơ sở của thiết bị USB) không cần biết về trường SYNC.

Tất cả các gói USB đều bắt đầu bằng trường SYNC, không có gì ngạc nhiên khi trường này phục vụ như một cơ chế đồng bộ hóa giữa người nhận và người gửi. Trường SYNC bao gồm 6 hoặc 30 bit xen kẽ tương ứng cho các bus tốc độ thấp và tốc độ tối đa hoặc tốc độ cao. Hai bit cuối cùng của trường SYNC bằng nhau (và thấp). Các trung tâm tốc độ cao có thể giảm tối đa 4 bit của trường SYNC, do đó, thiết bị nhận có thể không nhìn thấy toàn bộ trường, nhưng hai bit cuối cùng là tất cả những gì thiết bị cần để xác định chính xác nơi kết thúc trường SYNC và dữ liệu hữu ích bắt đầu.

### Trường định danh gói

**Mã định danh gói ( PID )** ngay sau trường SYNC. Có tổng cộng 17 PID được xác định (bao gồm PID của 0000b, được đặt trước), do đó, một PID yêu cầu 4 bit để mã hóa. Nếu lỗi trên bus làm thay đổi trường PID (ví dụ: thay đổi OUT PID thành IN PID), kết quả có thể là bất kỳ điều gì từ hành vi không mong muốn đến mất dữ liệu lớn. Do tầm quan trọng của tính toàn vẹn PID, trường PID rộng 8 bit. 4 bit cuối cùng chỉ đơn giản bổ sung cho 4 bit đầu tiên, điều này cung cấp một phương tiện để xác định xem một lỗi trên bus có làm thay đổi trường PID hay không. Trường PID được minh họa bên dưới.



## Định dạng trường định danh gói

Mã PID được phân loại thành 4 nhóm có chung hai bit có ý nghĩa nhỏ nhất. USB 2.0 định nghĩa các PID trong bảng sau.

Loại PID	Tên PID	PID [3:0]	Sự miêu tả
Mã thông báo	NGOÀI	0001b	Gói mô tả một giao dịch từ máy chủ đến chức năng.
	TRONG	1001b	Gói mô tả một giao dịch từ chức năng đến máy chủ.
	SOF	0101b	Gói đánh dấu sự bắt đầu của khung và chỉ định số khung.
	CÀI ĐẶT	1101b	Gói mô tả giao dịch THIẾT LẬP từ máy chủ đến chức năng thông qua ống điều khiển.
Dữ liệu	DỮ LIỆU0	0011b	Gói này là gói dữ liệu chẵn.
	DỮ LIỆU1	1011b	Gói này là một gói dữ liệu lẻ
	DỮ LIỆU2	0111b	Gói này chỉ được sử dụng trong các chuyển giao đẳng thời tốc độ cao, băng thông cao.
	MDATA	1111b	Gói này chỉ được sử dụng trong các giao dịch chia nhỏ hoặc chuyển giao đẳng thời tốc độ cao, băng thông cao.
bắt tay	xác nhận	0010b	Gói này xác nhận việc nhận thành công gói dữ liệu.
	NAK	1010b	Gói này chỉ ra rằng dữ liệu chưa sẵn sàng để truyền đi.
	QUÀY HÀNG	1110b	Gói này chỉ ra rằng điểm cuối đã tạm dừng hoặc ống điều khiển không hỗ trợ một yêu cầu nhất định.
	NYET	0110b	Người nhận vẫn chưa phản hồi hoặc máy chủ sẽ bắt đầu gửi các gói PING.
Đặc biệt	TRƯỚC	1100b	Gói này là phần mở đầu do máy chủ phát hành cho một giao dịch phân tách.
	LỖI	1100b	Gói này là một phản hồi bắt tay đã xảy ra lỗi giao dịch phân tách. Lưu ý rằng PID này giống với PID của gói PRE.
	TÁCH RA	1000b	Gói này hỗ trợ các giao dịch phân chia giữa máy chủ và trung tâm tốc độ cao.
	PING	0100b	Gói này được sử dụng để điều khiển luồng trong điều khiển tốc độ cao và truyền số lượng lớn.
	Kín đáo	0000b	Đây là PID dành riêng và không được sử dụng.

### Các loại PID

#### Trường địa chỉ

Các trường địa chỉ chọn một điểm cuối cụ thể trên một chức năng cụ thể. Đường nhiên, hai trường như vậy được xác định: trường địa chỉ và trường điểm cuối. Tất cả các thiết bị phải giải mã đầy đủ các trường này; phải bỏ qua sự không khớp của một trong hai trường (bao gồm trường điểm cuối chỉ định điểm cuối chưa được khởi tạo).

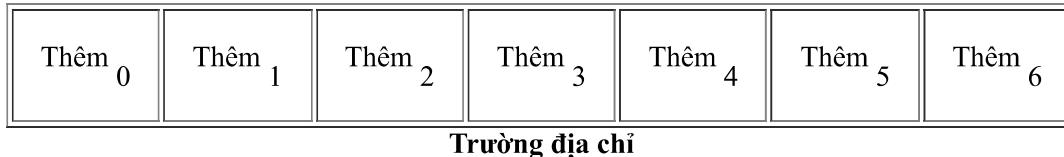
#### Trường địa chỉ

Trường địa chỉ được chỉ định cho các PID sau:

- TRONG
- CÀI ĐẶT

- NGOÀI
- PING
- TÁCH RA

Trường địa chỉ rộng 7 bit và được minh họa bên dưới. Mỗi giá trị có thể chỉ có thể chỉ ra một chức năng duy nhất. Địa chỉ 0 được dành riêng làm **địa chỉ mặc định** và không thể gán cho bất kỳ chức năng nào. Tất cả các chức năng phải phản hồi địa chỉ mặc định khi đặt lại và bật nguồn cho đến khi máy chủ chỉ định cho chức năng một địa chỉ cụ thể. Do đó, một bộ điều khiển máy chủ có thể hỗ trợ tối đa 127 thiết bị cùng một lúc.

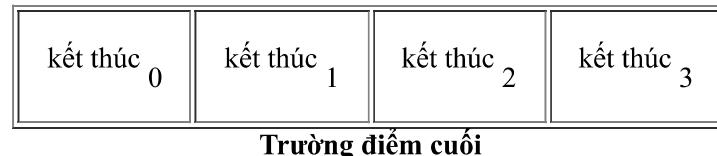


### Trường điểm cuối

Trường điểm cuối được chỉ định cho các PID sau:

- TRONG
- CÀI ĐẶT
- NGOÀI
- PING

Trường điểm cuối rộng 4 bit và được minh họa bên dưới. Tất cả các chức năng phải hỗ trợ ít nhất điểm cuối số 0 (ông điều khiển mặc định). Các chức năng tốc độ thấp chỉ có thể triển khai thêm 2 đường ống, trong khi các thiết bị tốc độ cao và đầy đủ chỉ bị giới hạn bởi độ rộng của trường điểm cuối. Nói cách khác, độ rộng của trường điểm cuối là lý do khiến các thiết bị tốc độ cao và đầy đủ bị giới hạn triển khai tối đa 15 điểm cuối IN bổ sung và 15 điểm cuối OUT bổ sung, như đã lưu ý ở trên trong phần Điểm cuối và số điểm cuối của thiết bị .



### Trường dữ liệu

Trường dữ liệu có thể nằm trong khoảng từ 0 đến 1.024 byte và phải là một số byte nguyên vẹn. Các byte dữ liệu được gửi bit ít quan trọng nhất trước.

### Kiểm tra dự phòng theo chu kỳ

Kiểm tra dự phòng theo chu kỳ (CRC) bảo vệ tất cả các trường không phải PID và cung cấp phạm vi bảo hiểm 100% cho tất cả các lỗi bit đơn và bit kép. CRC được cung cấp cho từng trường mã thông báo cũng như trường dữ liệu. Điều này cung cấp một cơ chế để máy chủ hoặc thiết bị nhận dạng và sửa hoặc bỏ qua các trường bị hỏng hoặc trong hầu hết các trường hợp là toàn bộ gói bị hỏng.

### bắt tay

Các loại giao dịch hỗ trợ điều khiển luồng trả về bắt tay để biểu thị:

- Nhận dữ liệu thành công
- Chấp nhận hoặc từ chối lệnh
- Kiểm soát lưu lượng
- điều kiện tạm dừng

Bắt tay luôn được trả lại trong giai đoạn bắt tay của giao dịch, nhưng cũng có thể được trả lại trong giai đoạn dữ liệu (thay cho gói dữ liệu dự kiến). Để hiểu rõ nhất về một phản hồi bắt tay nhất định, sẽ rất hữu ích khi hiểu ý nghĩa của từng loại gói bắt tay, cũng như các điều kiện theo đó mỗi phản hồi bắt tay có thể được đưa ra. Phần này được chia như vậy.

## Gói bắt tay

Tất cả các loại gói bắt tay đã được liệt kê trước đó và ngắn gọn trong Trường Định danh Gói. Phần này thảo luận chi tiết hơn về các loại gói đó.

### xác nhận

Một cái bắt tay ACK được đưa ra để thông báo rằng một gói dữ liệu đã được nhận thành công mà không có bất kỳ lỗi nhồi bit hoặc lỗi CRC nào trên trường dữ liệu và trường PID không bị hỏng.

Các gói ACK có thể được cấp khi bit thứ tự của bên nhận khớp với bit thứ tự của gói dữ liệu nhận được (và dữ liệu có thể được chấp nhận), nhưng gói ACK cũng có thể được cấp khi bit thứ tự của bên nhận không khớp với bit thứ tự của gói dữ liệu. gói dữ liệu đã nhận (và dữ liệu không thể được chấp nhận). Điều này có vẻ phản trực giác, nhưng lý do sẽ trở nên rõ ràng trong các phần thảo luận về chuyên đổi dữ liệu.

Gói bắt tay ACK	
Có thể được phát hành bởi...	Đối với các giao dịch này
Chủ nhà	TRONG
Chức năng	NGOÀI CÀI ĐẶT PING

### NAK

Gói bắt tay NAK thường được sử dụng để kiểm soát luồng để chỉ ra rằng một chức năng tạm thời không thể truyền hoặc nhận dữ liệu. Máy chủ không bao giờ gửi gói bắt tay NAK cho thiết bị.

Hàm trả về gói bắt tay NAK cho máy chủ sau giao dịch OUT khi hàm không thể nhận dữ liệu (thường là do bộ đệm bên trong của hàm hiện đã đầy). Phản hồi này không phải là lỗi mà thay vào đó, máy chủ nên thử truyền lại sau, cho phép chức năng có thời gian xử lý dữ liệu hiện có trong bộ đệm của nó.

Hàm trả về gói bắt tay NAK cho máy chủ trong giai đoạn dữ liệu của giao dịch IN để cho biết rằng hàm không có bất kỳ dữ liệu nào để truyền.

Gói bắt tay NAK	
Có thể được phát hành bởi...	Đối với các giao dịch này
Chức năng	TRONG NGOÀI PING

### QUÀY HÀNG

Một chức năng sử dụng gói bắt tay STALL để chỉ ra rằng nó không thể truyền hoặc nhận dữ liệu. Bên cạnh óng điều khiển mặc định, tất cả các điểm cuối của chức năng đều ở trạng thái không xác định sau khi thiết bị phát hành gói bắt tay STALL. Máy chủ không bao giờ được phát hành gói bắt tay STALL.

Gói bắt tay STALL	
Có thể được phát hành bởi...	Đối với các giao dịch này
Chức năng	TRONG NGOÀI PING

Thông thường, bắt tay STALL chỉ ra một gian hàng chức năng. Lỗi **chức năng** xảy ra khi *tính năng tạm dừng* (sẽ được đề cập trong "Khuôn khổ USB") của điểm cuối được đặt. Trong trường hợp này, cần có sự can thiệp của máy chủ thông qua óng điều khiển mặc định để xóa *tính năng tạm dừng* của điểm cuối bị tạm dừng.

Ít thường xuyên hơn, hàm trả về một bắt tay STALL trong giai đoạn THIẾT LẬP hoặc DỮ LIỆU của quá trình truyền điều khiển. Điều này được gọi là **gian hàng giao thức** và được giải quyết khi máy chủ phát hành giao dịch SETUP tiếp theo.

### NYET

Gói NYET có thể được phát hành bởi một chức năng như một phần của giao thức PING.

Gói bắt tay NYET	
Có thể được phát hành bởi...	Đối với các giao dịch này
trung tâm	TÁCH RA

Các trung tâm có thể phát hành gói bắt tay NYET để phản hồi giao dịch phân tách chưa hoàn thành trên xe buýt tốc độ thấp/tối đa.

**LỖI**

Các trung tâm có thể phát hành gói bắt tay ERR đặc biệt để báo cáo lỗi trên xe buýt tốc độ thấp/tối đa như một phần của giao thức giao dịch phân tách.

**Hoàn cảnh phản hồi của chức năng/máy chủ**

Phần này mô tả các trường hợp chức năng khiếu nại máy chủ hoặc chức năng đưa ra phản hồi dự kiến, không có phản hồi hoặc phản hồi gói bắt tay nhất định. Các bảng trong phần này được lấy và sửa đổi một chút cho rõ ràng từ các thông số kỹ thuật của USB 2.0, phần 8.4.6. Dấu gạch ngang biểu thị "không quan tâm."

**Chức năng đáp ứng các giao dịch IN**

Mã nhận được bị hỏng	Chức năng Tính năng tạm dừng điểm cuối Tx	Chức năng có thể truyền dữ liệu	Hành động được thực hiện bởi chức năng
Đúng	-	-	Trả lại không có phản hồi
KHÔNG	Bộ	-	Phát hành bắt tay STALL
KHÔNG	Không được thiết lập	KHÔNG	Vấn đề bắt tay NAK
KHÔNG	Không được thiết lập	Đúng	Phát hành gói dữ liệu

**Phản hồi của máy chủ đối với các giao dịch IN**

Gói dữ liệu bị hỏng	Máy chủ có thể chấp nhận dữ liệu	Hành động được thực hiện bởi máy chủ	Cái bắt tay được chủ nhà trả lại
Đúng	-	Hủy dữ liệu	Trả lại không có phản hồi
KHÔNG	KHÔNG	Hủy dữ liệu	Trả lại không có phản hồi
KHÔNG	Đúng	Chấp nhận dữ liệu	Phát hành bắt tay ACK

**Chức năng phản hồi các giao dịch OUT**

Gói dữ liệu bị hỏng	Tính năng tạm dừng máy thu	Chuỗi bit phù hợp	Chức năng có thể chấp nhận dữ liệu	Hành động được thực hiện bởi chức năng
Đúng	-	-	-	Trả lại không có phản hồi
KHÔNG	Bộ	-	-	Phát hành bắt tay STALL
KHÔNG	Không được thiết lập	KHÔNG	-	Phát hành bắt tay ACK
KHÔNG	Không được thiết lập	Đúng	Đúng	Phát hành bắt tay ACK
KHÔNG	Không được thiết lập	Đúng	KHÔNG	Vấn đề bắt tay NAK

## Chức năng phản hồi các giao dịch CÀI ĐẶT

Một chức năng phải luôn chấp nhận dữ liệu trong giao dịch SETUP và không bao giờ được đưa ra bắt tay STALL hoặc NAK để đáp lại. Tất cả các điểm cuối không kiểm soát chỉ cần bỏ qua bất kỳ giao dịch SETUP nào được gửi đến điểm cuối đó. Điều này cho phép các giao dịch SETUP hoạt động như một cơ chế (tái) đồng bộ hóa giữa máy chủ và điểm cuối điều khiển của chức năng.

## Giao thức giao dịch PING

Hãy xem xét một thiết bị lưu trữ dung lượng lớn USB. Trong quá trình chuyển từ máy chủ sang chức năng, bộ đệm của chức năng sẽ chứa đầy dữ liệu đang chờ xử lý được cam kết với phương tiện vật lý. Khi bộ đệm của chức năng đầy, chức năng không thể chấp nhận dữ liệu mới cho đến khi một số bộ đệm được cam kết, vì vậy nếu máy chủ tiếp tục gửi các giao dịch OUT, chức năng phải NAK chúng.

Vấn đề với mô hình OUT/NAK này là một chức năng phải đợi giai đoạn bắt tay của giao dịch OUT trước khi phản hồi bằng NAK. Vì giai đoạn bắt tay xảy ra sau giai đoạn dữ liệu, điều này có thể lãng phí một lượng băng thông đáng kể. Các bus tốc độ thấp và tốc độ tối đa gặp phải vấn đề này, nhưng thông số kỹ thuật của USB 2.0 đã giới thiệu giao thức giao dịch PING cho các bus tốc độ cao.

Giao thức giao dịch PING rất đơn giản. Thay vì giao dịch OUT, máy chủ đưa ra giao dịch PING cho chức năng khi máy chủ muốn gửi dữ liệu. Hảm phản hồi bằng NAK để cho biết rằng nó chưa sẵn sàng nhận dữ liệu (cụ thể, bộ đệm của hảm không thể chứa lượng dữ liệu tái trọng dữ liệu tối đa của điểm cuối) hoặc ACK để cho biết máy chủ có thể bắt đầu gửi dữ liệu.

Khung USB 2.0 cho phép các điểm cuối chỉ định một khoảng thời gian, về mặt vi khung, là số lượng khung vi mô mà máy chủ lưu trữ phải đợi trước khi phát một gói PING khác tới điểm cuối. Tuy nhiên, máy chủ không bắt buộc phải đợi khoảng thời gian này trước khi phát hành gói PING tiếp theo.

Trong quá trình điều khiển tốc độ cao hoặc truyền số lượng lớn từ máy chủ đến chức năng, khi một giao dịch OUT khiếu không gian bộ đệm trống của chức năng giảm xuống dưới tải trọng dữ liệu tối đa của điểm cuối, thì chức năng đó sẽ phản hồi bằng gói bắt tay NYET. Điều này chỉ ra rằng máy chủ nên bắt đầu phát hành các gói PING thay vì các giao dịch OUT bổ sung.

## Đồng bộ hóa chuyển đổi dữ liệu

Trong quá trình truyền, máy chủ và chức năng phải được đồng bộ hóa. Khả năng duy trì đồng bộ hóa có nghĩa là máy chủ hoặc chức năng có thể phát hiện khi mất đồng bộ hóa và trong hầu hết các trường hợp, đồng bộ hóa lại.

Mỗi điểm cuối duy trì, bên trong (trong phần cứng của chức năng), một bit chuyển đổi dữ liệu, còn được gọi là bit chuỗi dữ liệu. Máy chủ cũng duy trì bit chuyển đổi dữ liệu cho mọi điểm cuối mà nó giao tiếp. Trạng thái của bit chuyển đổi dữ liệu trên người gửi được chỉ định bởi DATA PID mà người gửi sử dụng.

Máy thu bắt bit chuỗi dữ liệu của nó khi nó có thể chấp nhận dữ liệu và nó nhận được gói dữ liệu không có lỗi với DATA PID dự kiến. Người gửi chỉ bật/tắt bit chuỗi dữ liệu của nó khi nhận được bắt tay ACK hợp lệ. Sơ đồ chuyển đổi dữ liệu này yêu cầu người gửi và người nhận đồng bộ hóa các bit chuyển đổi dữ liệu của họ khi bắt đầu giao dịch.

Đồng bộ hóa chuyển đổi dữ liệu hoạt động khác nhau tùy thuộc vào loại chuyển được sử dụng:

- Quá trình truyền điều khiển khởi tạo các bit chuyển đổi dữ liệu của điểm cuối thành 0 bằng gói SETUP.
- Các điểm cuối Ngắt và Hàng loạt khởi tạo các bit chuyển đổi dữ liệu của chúng thành 0 khi có bất kỳ sự kiện cấu hình nào.
- Truyền đẳng thời không thực hiện bắt tay và do đó không hỗ trợ đồng bộ hóa chuyển đổi dữ liệu.
- Truyền đồng bộ tốc độ cao, băng thông cao hỗ trợ sắp xếp thứ tự dữ liệu trong một vi khung.

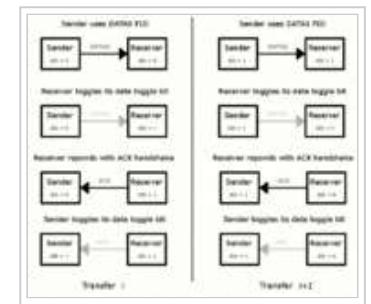
Phản còn lại của phần này minh họa cách mỗi thiết bị gửi và nhận quản lý các bit chuyển đổi dữ liệu của chúng trong các tình huống truyền khác nhau. Mũi tên đen biểu thị mục đích truyền dữ liệu trên USB. Mũi tên màu xám biểu thị rằng quá trình

truyền dữ liệu dự định đã hoàn thành mà không có lỗi. Các mũi tên màu đỏ, không liên tục biểu thị rằng dữ liệu dự định đã bị hỏng trong quá trình truyền hoặc hoàn toàn không truyền được.

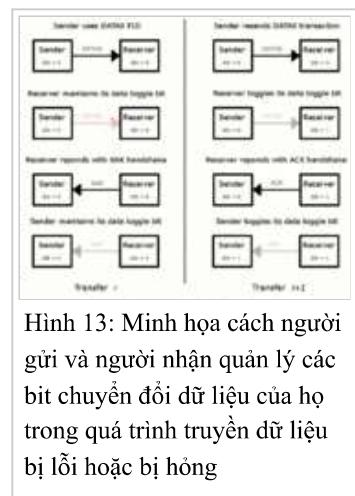
## truyền thành công

Hình 12 minh họa một quá trình truyền dữ liệu thành công. Cả hai thiết bị đều có các bit chuyển đổi dữ liệu được đặt thành 0 khi bắt đầu truyền i. Theo đó, thiết bị gửi phát ra DATA0 PID theo sau là gói dữ liệu. Thiết bị nhận đọc thành công DATA0 PID cũng như gói dữ liệu. Vì bit chuyển đổi dữ liệu của bên nhận khớp với DATA0 PID và không có lỗi trong việc truyền dữ liệu còn lại, nên bên nhận chuyển đổi bit chuyển đổi dữ liệu của nó thành 1 và đưa ra phản hồi bắt tay ACK. Người gửi nhận được bắt tay ACK mà không có lỗi và do đó chuyển đổi bit chuyển đổi dữ liệu của nó thành 1.

Giả sử rằng quá trình truyền tiếp theo cũng xảy ra mà không có lỗi, thì điểm khác biệt duy nhất là DATA1 PID được sử dụng thay vì DATA0 và các thiết bị gửi và nhận chuyển đổi các bit chuyển đổi dữ liệu của chúng từ 1 sang 0 trong cùng các giai đoạn mà cùng một bit đã chuyển đổi thành a 1 trong lần chuyển trước.



Hình 12: Minh họa cách người gửi và người nhận quản lý các bit chuyển đổi dữ liệu của họ trong quá trình truyền dữ liệu thành công



Hình 13: Minh họa cách người gửi và người nhận quản lý các bit chuyển đổi dữ liệu của họ trong quá trình truyền dữ liệu bị lỗi hoặc bị hỏng

## Truyền dữ liệu bị lỗi hoặc bị hỏng

Hình 13 minh họa việc truyền dữ liệu bị lỗi hoặc bị hỏng.

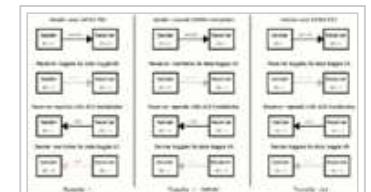
Cả hai thiết bị đều có các bit chuyển đổi dữ liệu được đặt thành 0 khi bắt đầu truyền i. Theo đó, thiết bị gửi phát ra DATA0 PID theo sau là gói dữ liệu. Thiết bị nhận không nhìn thấy gói dữ liệu hoặc đọc gói dữ liệu bị hỏng. Người nhận duy trì bit chuyển đổi dữ liệu của nó và đưa ra bắt tay NAK. Người gửi nhìn thấy bắt tay NAK thành công và do đó không chuyển đổi bit chuyển đổi dữ liệu của nó.

Khi bắt đầu lần truyền tiếp theo, cả thiết bị gửi và nhận đều có các bit chuyển đổi dữ liệu vẫn được đặt thành 0. Giả sử quá trình truyền này hoàn tất thành công, nó được thực hiện như mô tả ở trên, dưới dạng truyền thành công .

## Bắt tay ACK không thành công hoặc bị hỏng

Hình 14 minh họa một bắt tay ACK không thành công hoặc bị hỏng. Cả hai thiết bị đều có các bit chuyển đổi dữ liệu được đặt thành 0 khi bắt đầu truyền i. Theo đó, thiết bị gửi phát ra DATA0 PID theo sau là gói dữ liệu. Thiết bị nhận đọc thành công DATA0 PID cũng như gói dữ liệu. Vì bit chuyển đổi dữ liệu của thiết bị gửi vẫn là 0 và bit chuyển đổi dữ liệu của thiết bị nhận đã được đặt thành 1. Người gửi không thấy phản hồi ACK hợp lệ cho quá trình truyền i, nên thử truyền lại i. Với bit chuyển đổi dữ liệu bằng 0, người gửi đưa ra DATA0 PID theo sau là gói dữ liệu. Thiết bị nhận đọc thành công DATA0 PID cũng như gói dữ liệu. Vì bit chuyển đổi dữ liệu của thiết bị nhận không khớp với DATA0 PID, nên thiết bị nhận duy trì giá trị bit chuyển đổi dữ liệu của nó là 1 và đưa ra phản hồi bắt tay ACK. Người gửi nhận được phản hồi ACK mà không có lỗi và do đó chuyển đổi bit chuyển đổi dữ liệu của nó thành 1.

Tại thời điểm này, bit chuyển đổi dữ liệu của thiết bị gửi vẫn là 0 và bit chuyển đổi dữ liệu của thiết bị nhận đã được đặt thành 1. Người gửi không thấy phản hồi ACK hợp lệ cho quá trình truyền i, nên thử truyền lại i. Với bit chuyển đổi dữ liệu bằng 0, người gửi đưa ra DATA0 PID theo sau là gói dữ liệu. Thiết bị nhận đọc thành công DATA0 PID cũng như gói dữ liệu. Vì bit chuyển đổi dữ liệu của thiết bị nhận không khớp với DATA0 PID, nên thiết bị nhận duy trì giá trị bit chuyển đổi dữ liệu của nó là 1 và đưa ra phản hồi bắt tay ACK. Người gửi nhận được phản hồi ACK mà không có lỗi và do đó chuyển đổi bit chuyển đổi dữ liệu của nó thành 1.



Hình 14: Minh họa cách người gửi và người nhận quản lý các bit chuyển đổi dữ liệu của họ trong một phản hồi ACK bị lỗi hoặc hỏng

## Chuyển USB được xem lại

Rất nhiều thông tin đã được giới thiệu kể từ Khái niệm cơ bản về truyền USB và rất dễ bị lạc trong các chi tiết. Thậm chí với sự hiểu biết đúng đắn về bốn loại truyền USB, thường rất khó để ngoại suy từ sự phức tạp của giao thức USB sang sự hiểu biết về cách mọi thứ khớp với nhau. Vì những lý do này, phần này dự định làm rõ một số khái niệm có khả năng gây nhầm lẫn cả rõ ràng và ngầm định bằng cách xem lại bốn loại chuyển giao trong bối cảnh của tất cả các thông tin được đề cập kể từ lần thảo luận đầu tiên về chúng.

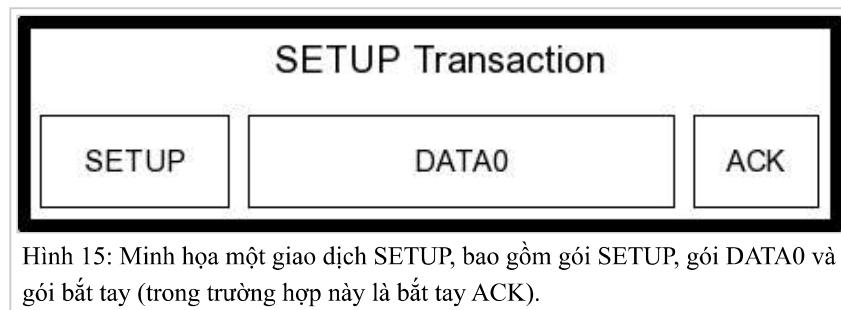
Người đọc e ngại có thể nhận thấy rằng một số thuật ngữ như THIẾT LẬP và DỮ LIỆU được sử dụng cả khi đề cập đến số nhận dạng gói và khi đề cập đến các loại giao dịch. Mục wiki này rất có thể là nguồn thông tin USB đầu tiên và duy nhất cần một chút thời gian để phân biệt cụ thể giữa hai nguồn này.

Trong Khái niệm cơ bản về chuyển USB, các giao dịch USB chỉ được đề cập ngắn gọn như đã được sao chép bên dưới:

Hầu hết các giao dịch USB bao gồm ba gói:  
 \* Gói mã thông báo cho biết loại và hướng của giao dịch, địa chỉ thiết bị và số điểm cuối.  
 \* Tùy thuộc vào hướng của giao dịch, máy chủ hoặc chức năng sẽ gửi gói dữ liệu (có thể chỉ đơn giản là không có dữ liệu để gửi).  
 \* Thiết bị nhận phản hồi bằng gói bắt tay để cho biết liệu quá trình truyền có thành công hay không.

Sau đó, bên dưới Gói, một gói được mô tả là "đơn vị truyền dữ liệu nguyên tử."

Nếu một gói là một nguyên tử, thì một giao dịch sẽ là một phân tử. Nghĩa là, một giao dịch được tạo thành từ một số gói theo một thứ tự cụ thể và các gói tạo nên một giao dịch không thể được sắp xếp lại hoặc tách rời mà vẫn tạo ra cùng một giao dịch. Các giao dịch thường được đặt tên theo gói mã thông báo của chúng (hoặc gói "đặc biệt" của chúng, trong trường hợp PING hoặc SPLIT vì các gói đặc biệt này đóng vai trò giống như gói mã thông báo), ngoại trừ các giao dịch IN hoặc OUT thường được gọi chung là , dưới dạng giao dịch DATA. Trong các ví dụ, các giao dịch có chứa một giai đoạn dữ liệu thường cho biết loại DATA PID được sử dụng bằng cách thêm 0, 1, 2 hoặc M vào tên hoặc thêm nó vào trong ngoặc đơn (ví dụ: SETUP(0) hoặc SETUP0, OUT1 hoặc OUT (1)).



Hình 15: Minh họa một giao dịch SETUP, bao gồm gói SETUP, gói DATA0 và gói bắt tay (trong trường hợp này là bắt tay ACK).

Một ví dụ về một giao dịch SETUP đơn lẻ được mô tả trong hình 15. Giao dịch này chứa ba gói điển hình. Gói mã thông báo có SETUP PID, gói dữ liệu có DATA0 PID (nhớ lại rằng gói SETUP khởi tạo cả chức năng và dữ liệu của máy chủ chuyển bit thành 0) và phản hồi bắt tay có ACK PID.

Chuyển khoản được tạo thành từ các giao dịch. Các giao dịch có thể không được sắp xếp lại trong một lần chuyển nhưng, như đã thảo luận trong Khung và Vị khung, các giao dịch của một lần chuyển cụ thể có thể hoặc không thể được gửi qua xe buýt một cách liên tục. Phần còn lại của phần này xem xét các giao dịch liên quan đến bốn loại chuyển nhượng.

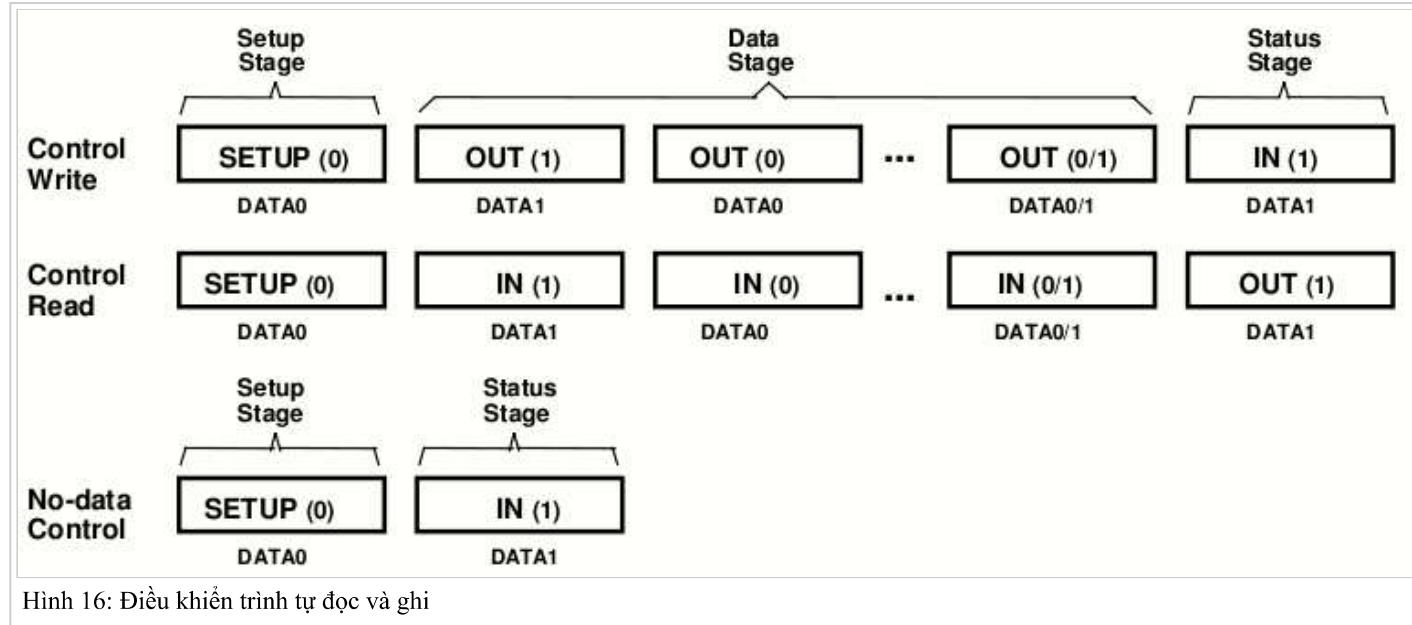
### Kiểm soát chuyển giao

Chuyển giao kiểm soát là chuyển giao duy nhất sử dụng giao dịch SETUP. Việc chuyển quyền kiểm soát diễn ra trong tối đa ba giai đoạn:

- Giai đoạn THIẾT LẬP chỉ bao gồm một giao dịch THIẾT LẬP
- Giai đoạn DATA là tùy chọn. Nếu được sử dụng, nó có thể chứa một hoặc nhiều giao dịch IN hoặc một hoặc nhiều giao dịch OUT. Giao dịch đầu tiên trong số các giao dịch IN hoặc OUT này sử dụng DATA1 PID. Thứ hai, nếu có, sử dụng DATA0 PID, thứ ba DATA1, v.v.

- Giai đoạn TÌNH TRẠNG bao gồm một giao dịch IN hoặc một giao dịch OUT, phải sử dụng DATA1 PID. Nếu có giao đoạn DỮ LIỆU, thì giao đoạn TÌNH TRẠNG sử dụng loại giao dịch ngược lại làm giao đoạn DỮ LIỆU (nghĩa là nếu giao đoạn DỮ LIỆU bao gồm một hoặc nhiều giao dịch OUT, thì giao đoạn TÌNH TRẠNG bao gồm một giao dịch VÀO duy nhất và ngược lại). Khi giao đoạn DỮ LIỆU bị bỏ qua, giao đoạn TÌNH TRẠNG sử dụng một giao dịch IN duy nhất.

Hình 16 được lấy từ Hình 8-37 của thông số kỹ thuật USB 2.0 và minh họa thứ tự giao dịch, giá trị bit của chuỗi dữ liệu và loại DATA PID cho chuỗi đọc và ghi điều khiển.

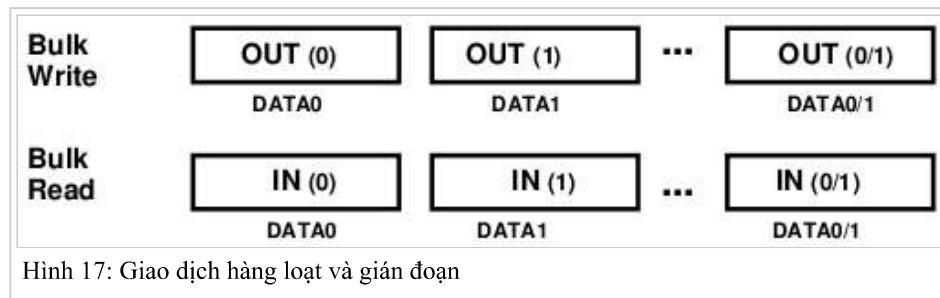


Hình 16: Điều khiển trình tự đọc và ghi

### Chuyển số lượng lớn và gián đoạn

Trong ngữ cảnh của giao thức USB, sự khác biệt duy nhất giữa chuyển hàng loạt và gián đoạn là chuyển hàng loạt, khi hoạt động ở tốc độ cao, hỗ trợ Giao thức giao dịch PING . Lưu ý rằng trong bối cảnh chung, hai loại truyền tải này cũng khác nhau ở chỗ chúng được lên lịch trình khác nhau bởi máy chủ (tham khảo Phân bổ thời gian xe buýt ).

Tất cả các điểm cuối số lượng lớn và ngắn đều chuyển theo một hướng. Các bit chuyển đổi dữ liệu cho các điểm cuối này được khởi tạo bằng 0 sau bất kỳ sự kiện cấu hình nào. Hình 17 được lấy từ Hình 8-35 của thông số kỹ thuật USB 2.0 và minh họa các giao dịch gián đoạn và hàng loạt cho cả hai điểm cuối IN và OUT. Lưu ý rằng, mặc dù hình chỉ đề cập đến số lần đọc và ghi hàng loạt, thông số kỹ thuật của USB 2.0 tham chiếu đến hình tương tự từ phần 8.5.4, trên Giao dịch gián đoạn.



Hình 17: Giao dịch hàng loạt và gián đoạn

### Chuyển giao đằng thời

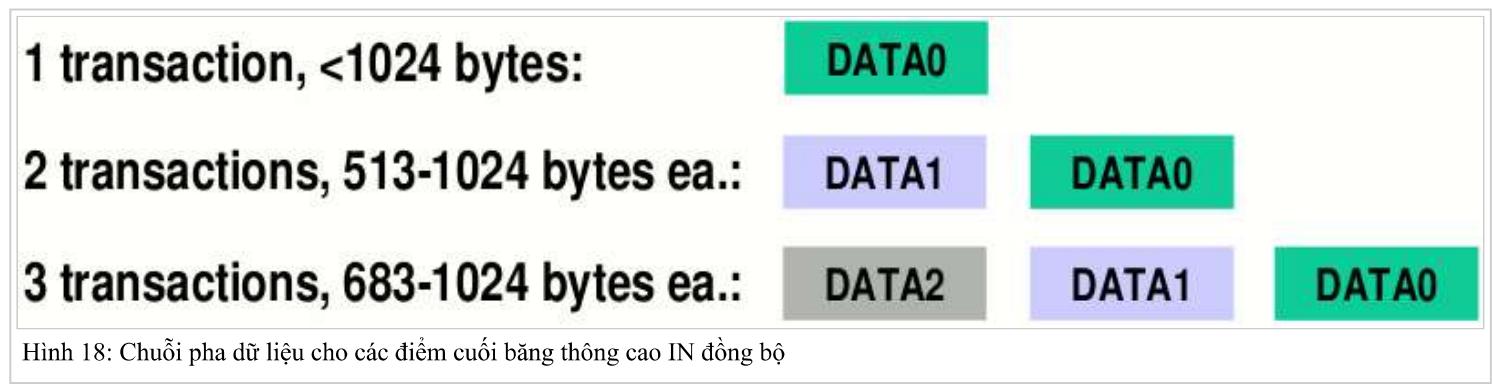
Chuyển giao đằng thời là loại chuyển giao duy nhất mà các giao dịch không có giao đoạn bắt tay. Truyền đằng thời chỉ nên sử dụng DATA0 PID, tuy nhiên, bộ điều khiển máy chủ cũng phải hỗ trợ DATA1 PID, mặc dù truyền đằng thời không sử dụng cơ chế bit đồng bộ hóa dữ liệu.

## Truyền đồng bộ tốc độ cao, băng thông cao

Truyền đồng thời tốc độ cao, băng thông cao là trường hợp đặc biệt của truyền đồng thời, trong đó tối đa 3 giao dịch có thể xảy ra trong một vi khung. Là một loại chuyển giao đồng thời cụ thể, chuyển giao đồng thời tốc độ cao, băng thông cao bỏ qua giai đoạn bắt tay trong các giao dịch của chúng. Do tối đa 3 giao dịch có thể xảy ra trong một khung vi mô, truyền đồng thời tốc độ cao, băng thông cao nên cần sử dụng cơ chế sắp xếp thứ tự dữ liệu giống như các loại truyền khác.

USB 2.0 thực hiện cơ chế sắp xếp thứ tự dữ liệu để truyền đồng thời tốc độ cao, băng thông lớn, nhưng nó hoạt động hơi khác so với các loại truyền khác. Trên thực tế, trình tự dữ liệu hoạt động khác nhau tùy thuộc vào việc điểm cuối là IN hay điểm cuối đồng tốc độ cao, băng thông cao OUT.

Đối với các điểm cuối đồng thời tốc độ cao, băng thông cao IN, trình tự dữ liệu được mô tả trong hình 18, được lấy từ hình 5-11 của thông số kỹ thuật USB 2.0. Giao dịch cuối cùng trong một vi khung luôn sử dụng DATA0 PID. Giao dịch từ thứ hai đến cuối cùng trong một vi khung sử dụng PID DATA1 và giao dịch từ thứ ba đến cuối cùng trong một vi khung luôn sử dụng PID DATA2.



Đối với các điểm cuối đồng thời tốc độ cao, băng thông cao OUT, trình tự dữ liệu được mô tả trong hình 19, được lấy từ hình 5-12 của thông số kỹ thuật USB 2.0. Tất cả các giao dịch trừ giao dịch cuối cùng đều sử dụng MDATA PID. Giao dịch cuối cùng sử dụng PID DATA0, DATA1 hoặc DATA2, tùy thuộc vào số lượng giao dịch dự kiến diễn ra trong vi khung. Nếu một giao dịch có nghĩa là diễn ra, thì đó cũng là giao dịch cuối cùng và sử dụng DATA0 PID. Nếu hai giao dịch được thực hiện, thì giao dịch cuối cùng sẽ sử dụng DATA1 PID. Nếu ba giao dịch được thực hiện, thì giao dịch cuối cùng sẽ sử dụng DATA2 PID.



## Khung thiết bị USB

Khung thiết bị USB là thứ làm cho hỗ trợ USB trở nên hấp dẫn. Tất nhiên, các loại truyền và giao thức USB được thiết kế tốt, nhưng khung thiết bị USB xác định các trạng thái thiết bị tiêu chuẩn mà tất cả các thiết bị phải hỗ trợ, cũng như các yêu cầu và phản hồi tiêu chuẩn cho phép máy chủ truy xuất đủ thông tin về thiết bị để xác định trình điều khiển thiết bị chính xác và báo cáo thông tin về thiết bị ngay cả khi không có trình điều khiển thiết bị chính xác (ví dụ: tên nhà sản xuất, tên sản phẩm, v.v.).

## Chức năng, Cấu hình, Giao diện và Điểm cuối

Tất cả các thiết bị hoặc chức năng USB đều có ít nhất một cấu hình và mọi cấu hình đều có ít nhất một giao diện. Một giao diện có thể xác định không hoặc nhiều điểm cuối. Mỗi quan hệ này được minh họa trong hình 20.

Mặc dù các bộ mô tả cấu hình được giải quyết tuân tự bắt đầu với bộ mô tả cấu hình bằng không, nhưng mỗi cấu hình chỉ định một giá trị cấu hình duy nhất (trong phạm vi của chức năng), khác không. Giá trị **cấu hình** là những gì máy chủ lưu trữ cần biết để áp dụng một cấu hình nhất định cho thiết bị. Khi yêu cầu cấu hình hiện tại của thiết bị, giá trị trả về bằng 0 cho thiết bị chưa được định cấu hình và do đó ở trạng thái địa chỉ.

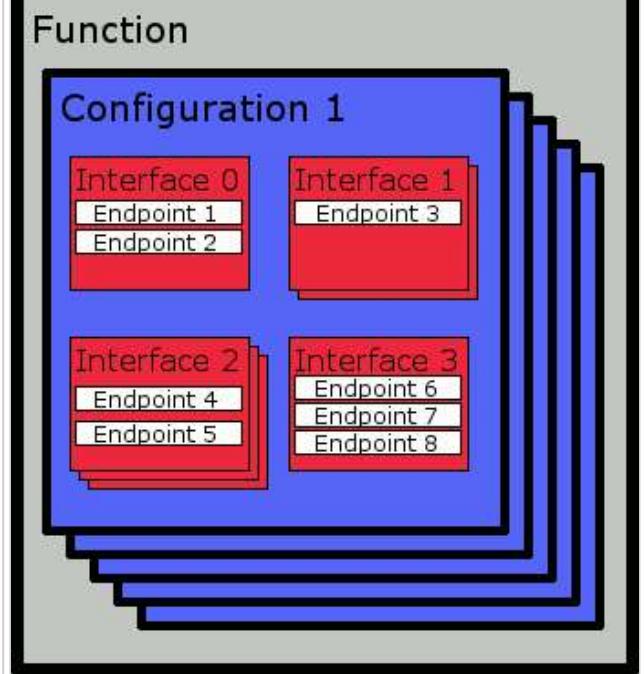
Một **giao diện** xác định việc sử dụng chức năng của một tập hợp các điểm cuối và có thể ngũ ý rằng một số yêu cầu dành riêng cho lớp có thể được thực thi thông qua ống điều khiển mặc định. Do đó, một giao diện không nhất thiết phải xác định bất kỳ điểm cuối bổ sung nào. Không có giao diện nào có thể xác định việc sử dụng chức năng của điểm cuối bằng không.

Mỗi giao diện mô tả một tập hợp các điểm cuối duy nhất trong phạm vi cấu hình. Tuy nhiên, một giao diện có thể cung cấp một hoặc nhiều **cài đặt thay thế**, có thể có các định nghĩa khác nhau cho cùng một nhóm điểm cuối. Khi máy chủ chọn cài đặt thay thế cho giao diện, định nghĩa của cài đặt thay thế được sử dụng thay vì cài đặt mặc định của cùng một giao diện.

## Trạng thái thiết bị USB

Thiết bị USB có thể xác định các trạng thái bên trong thiết bị, tuy nhiên, khung thiết bị USB xác định một tập hợp các trạng thái hiển thị cho cả máy chủ và thiết bị. Những trạng thái có thể nhìn thấy đó là như sau:

- **Đã đính kèm** - Ngay sau khi thiết bị USB được gắn vào hệ thống USB, thiết bị sẽ ở trạng thái này. Thông số kỹ thuật của USB không xác định trạng thái của thiết bị USB được tách ra khỏi hệ thống USB.
- **Được cấp nguồn** - Một thiết bị ở trạng thái này sau khi cả hai đều được gắn vào bus và dòng V<sub>BUS</sub> được áp dụng cho thiết bị (bộ điều khiển máy chủ điều khiển V<sub>BUS</sub> ở mức +5V, tuy nhiên điều này chỉ đặc biệt quan trọng đối với các nhà phát triển phần cứng). Ở trạng thái này, thiết bị không được phản hồi bất kỳ giao dịch xe buýt nào. Thông số kỹ thuật USB nhận ra ba tình huống tiềm ẩn liên quan đến cách thiết bị sử dụng năng lượng:
  - *Thiết Bị Tự Cấp* Nguồn lấy điện từ nguồn điện bên ngoài (ví dụ: máy in USB cắm vào tường cũng như cổng USB). Mặc dù thiết bị có thể được coi là "được cấp nguồn" về mặt kỹ thuật ngay cả trước khi gắn vào USB, nhưng thiết bị vẫn chỉ được coi là được cấp nguồn sau khi dòng V<sub>BUS</sub> được áp dụng cho thiết bị.
  - *Các thiết bị chạy bằng bus* chỉ lấy điện từ USB lên đến 100mA.
  - *Các Thiết Bị Tự Cấp Nguồn hoặc Xe Buýt* có thể lấy nguồn điện từ xe buýt hoặc nguồn điện bên ngoài, tùy thuộc vào cấu hình. Các thiết bị này có thể thay đổi nguồn điện bất cứ lúc nào. Nếu một thiết bị hiện đang tự cấp nguồn và yêu cầu nguồn điện hơn 100mA, nhưng chuyển sang cấp nguồn bằng bus, thì thiết bị đó phải trở về trạng thái Địa chỉ.
- **Mặc định** - Một thiết bị ở trạng thái được cấp nguồn sẽ chuyển sang trạng thái mặc định sau khi nhận được thiết lập lại xe buýt. Ở trạng thái này, thiết bị có thể định địa chỉ ở địa chỉ dành riêng, mặc định là 0. Tại thời điểm này, thiết bị đang hoạt động ở tốc độ chính xác. Máy chủ dự kiến sẽ cho phép 10 mili giây trước khi thiết bị phản hồi việc truyền dữ liệu sau khi đặt lại.
- **Địa chỉ** - Một thiết bị chuyển sang trạng thái này sau khi máy chủ gán cho nó một địa chỉ thông qua ống điều khiển mặc định, địa chỉ này luôn có thể truy cập được cho dù địa chỉ của thiết bị đã được đặt hay chưa.



Hình 20: Minh họa mối quan hệ giữa các chức năng, cấu hình, giao diện và điểm cuối.

- Đã định cấu hình** - Một thiết bị ở trạng thái này sau khi máy chủ kiểm tra các cấu hình có thể có của thiết bị và chọn một cấu hình. Tất cả các bit chuyển đổi dữ liệu của điểm cuối được khởi tạo bằng 0 khi thiết bị chuyển sang trạng thái này.
- Bị treo** - Khi không quan sát thấy lưu lượng truy cập nào trên bus trong khoảng thời gian 1 mili giây, thiết bị USB sẽ chuyển sang trạng thái này, được đặc trưng bởi mức tiêu thụ điện năng thấp. Cài đặt cấu hình và địa chỉ của thiết bị được duy trì trong khi bị treo. Một thiết bị thoát khỏi trạng thái treo ngay khi bắt đầu thấy hoạt động của xe buýt trở lại. Máy chủ dự kiến sẽ cho phép 10 mili giây trước khi mong đợi thiết bị phản hồi truyền dữ liệu sau khi tiếp tục.

## Khả năng đánh thức từ xa

Một trong những lý do khiến thiết bị USB có thể ngừng nhìn thấy lưu lượng USB và do đó đi vào trạng thái bị treo là vì máy chủ cũng có thể đã rơi vào trạng thái bị treo. Một số thiết bị, điển hình là bàn phím và chuột, hỗ trợ khả năng phát tín hiệu đánh thức từ xa cho máy chủ. Trong trường hợp phần mềm máy chủ không hỗ trợ đánh thức từ xa, khả năng này phải bị tắt khi đặt lại thiết bị USB. Nếu máy chủ hỗ trợ đánh thức từ xa, thì máy chủ có thể kích hoạt có chọn lọc khả năng đánh thức từ xa cho các thiết bị cụ thể (thường do người dùng chọn). Sau đó, các thiết bị này có thể phát tín hiệu đánh thức từ xa khi đang ở trạng thái treo để yêu cầu máy chủ thoát khỏi trạng thái treo của chính nó.

## Bảng liệt kê thiết bị USB

Phần sau đây mô tả quy trình liệt kê xe buýt, xảy ra sau khi thiết bị được kết nối với cổng được cấp nguồn:

- Trung tâm mà thiết bị đã được gắn vào sẽ thông báo cho máy chủ thông qua đường ống thay đổi trạng thái của nó. Thiết bị mới được gắn vào đang ở trạng thái được cấp nguồn vào thời điểm này và cổng mà thiết bị được gắn vào đã bị vô hiệu hóa.
- Máy chủ truy vấn thêm thông tin từ trung tâm để xác định rằng một thiết bị đã được gắn vào và cổng nào.
- Máy chủ phải đợi ít nhất 100 mili giây để cho phép thiết bị hoàn tất quy trình chèn và để nguồn điện ổn định trên thiết bị. Sau thời gian trễ, máy chủ kích hoạt cổng và phát tín hiệu đặt lại cho thiết bị trong ít nhất 50 mili giây.
- Hub thực hiện bất kỳ quá trình đặt lại cần thiết nào. Sau khi tín hiệu đặt lại được giải phóng, cổng được bật và thiết bị sẽ chuyển sang trạng thái mặc định.
- Máy chủ gán cho thiết bị một địa chỉ duy nhất, do đó chuyển thiết bị sang trạng thái địa chỉ.
- Máy chủ yêu cầu bộ mô tả thiết bị từ thiết bị thông qua ống điều khiển mặc định để xác định kích thước tải trọng dữ liệu tối đa thực tế của ống điều khiển mặc định cho thiết bị. Bước này có thể xảy ra trước hoặc sau khi máy chủ gán địa chỉ cho thiết bị.
- Máy chủ đọc tất cả thông tin cấu hình thiết bị có thể.
- Máy chủ chọn một cấu hình nhất định từ danh sách các cấu hình được thiết bị hỗ trợ và đặt thiết bị sử dụng cấu hình đó. Theo tùy chọn, máy chủ cũng có thể chọn cài đặt giao diện thay thế trong một cấu hình. Tất cả các điểm cuối được khởi tạo như được mô tả bởi cấu hình đã chọn và thiết bị đã sẵn sàng để sử dụng.

## Yêu cầu thiết bị USB

Các yêu cầu tiêu chuẩn, dành riêng cho lớp và dành riêng cho nhà cung cấp được thực hiện cho thiết bị USB qua ống điều khiển mặc định. Giao dịch THIẾT LẬP luôn có kích thước tải trọng dữ liệu là 8 byte, như đã lưu ý trong phần Kích thước tải trọng dữ liệu tối đa của Chuyển điều khiển. Định dạng của dữ liệu thiết lập như sau:

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả								
0	bmRequestType	1	Bitmap	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>D<sub>7</sub></td><td>D<sub>6</sub></td><td>Đ<sub>5</sub></td><td>D<sub>4</sub></td><td>Đ<sub>3</sub></td><td>D<sub>2</sub></td><td>d<sub>1</sub></td><td>D<sub>0</sub></td> </tr> </table> <p><b>D<sub>7</sub></b> Hướng truyền dữ liệu * Giá trị của bit này bị bỏ qua khi wLength bằng 0</p> <ul style="list-style-type: none"> <li>0b = Máy chủ đến thiết bị</li> <li>1b = Thiết bị đến máy chủ</li> </ul>	D <sub>7</sub>	D <sub>6</sub>	Đ <sub>5</sub>	D <sub>4</sub>	Đ <sub>3</sub>	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>
D <sub>7</sub>	D <sub>6</sub>	Đ <sub>5</sub>	D <sub>4</sub>	Đ <sub>3</sub>	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>					

				<b>D<sub>6...5</sub></b>	<b>Loại yêu cầu</b>																																	
					<ul style="list-style-type: none"> <li>■ 00b = Tiêu chuẩn</li> <li>■ 01b = Lớp</li> <li>■ 10b = Nhà cung cấp</li> <li>■ 11b = Dành riêng</li> </ul>																																	
				<b>D<sub>4...0</sub></b>	<b>Người nhận</b>																																	
					<ul style="list-style-type: none"> <li>■ 00000b = Thiết bị</li> <li>■ 00001b = Giao diện</li> <li>■ 00010b = Điểm cuối</li> <li>■ 00011b = Khác</li> <li>■ 00100b đến 11111b = Dành riêng</li> </ul>																																	
1	bYêu cầu	1	Giá trị		yêu cầu cụ thể																																	
2	giá trị w	2	Giá trị		Trường có kích thước từ có thể (hoặc không) đóng vai trò là tham số cho yêu cầu, tùy thuộc vào yêu cầu cụ thể.																																	
4	w Index	2	Chỉ số hoặc bù đắp		<p>Trường có kích thước từ có thể (hoặc không) đóng vai trò là tham số cho yêu cầu, tùy thuộc vào yêu cầu cụ thể. Thông thường, trường này chứa chỉ mục hoặc giá trị bù trừ.</p> <p>Khi bmRequestType chỉ định một điểm cuối là người nhận, định dạng của trường này như sau:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: center;"><b>D<sub>7</sub></b></td> <td style="text-align: center;"><b>D<sub>6</sub></b> –</td> <td style="text-align: center;"><b>D<sub>5</sub></b> –</td> <td style="text-align: center;"><b>D<sub>4</sub></b></td> <td style="text-align: center;"><b>D<sub>3</sub></b> –</td> <td style="text-align: center;"><b>D<sub>2</sub></b></td> <td style="text-align: center;"><b>d<sub>1</sub></b></td> <td style="text-align: center;"><b>D<sub>0</sub></b></td> </tr> <tr> <td style="text-align: center;">Phuong huong</td> <td colspan="4" style="text-align: center;">Dành riêng (đặt lại về 0)</td> <td colspan="3" style="text-align: center;">Số điểm cuối</td> </tr> <tr> <td style="text-align: center;"><b>D<sub>15</sub></b> –</td> <td style="text-align: center;"><b>D<sub>14</sub></b> –</td> <td style="text-align: center;"><b>D<sub>13</sub></b> –</td> <td style="text-align: center;"><b>D<sub>12</sub></b> –</td> <td style="text-align: center;"><b>D<sub>11</sub></b> –</td> <td style="text-align: center;"><b>D<sub>10</sub></b> –</td> <td style="text-align: center;"><b>D<sub>9</sub></b> –</td> <td style="text-align: center;"><b>D<sub>8</sub></b> –</td> </tr> <tr> <td colspan="8" style="text-align: center;">Dành riêng (đặt lại về 0)</td> </tr> </table>	<b>D<sub>7</sub></b>	<b>D<sub>6</sub></b> –	<b>D<sub>5</sub></b> –	<b>D<sub>4</sub></b>	<b>D<sub>3</sub></b> –	<b>D<sub>2</sub></b>	<b>d<sub>1</sub></b>	<b>D<sub>0</sub></b>	Phuong huong	Dành riêng (đặt lại về 0)				Số điểm cuối			<b>D<sub>15</sub></b> –	<b>D<sub>14</sub></b> –	<b>D<sub>13</sub></b> –	<b>D<sub>12</sub></b> –	<b>D<sub>11</sub></b> –	<b>D<sub>10</sub></b> –	<b>D<sub>9</sub></b> –	<b>D<sub>8</sub></b> –	Dành riêng (đặt lại về 0)								
<b>D<sub>7</sub></b>	<b>D<sub>6</sub></b> –	<b>D<sub>5</sub></b> –	<b>D<sub>4</sub></b>	<b>D<sub>3</sub></b> –	<b>D<sub>2</sub></b>	<b>d<sub>1</sub></b>	<b>D<sub>0</sub></b>																															
Phuong huong	Dành riêng (đặt lại về 0)				Số điểm cuối																																	
<b>D<sub>15</sub></b> –	<b>D<sub>14</sub></b> –	<b>D<sub>13</sub></b> –	<b>D<sub>12</sub></b> –	<b>D<sub>11</sub></b> –	<b>D<sub>10</sub></b> –	<b>D<sub>9</sub></b> –	<b>D<sub>8</sub></b> –																															
Dành riêng (đặt lại về 0)																																						
					Bit hướng (bit <b>D<sub>7</sub></b> ) được đặt thành 0 để biểu thị điểm cuối OUT với số điểm cuối đã chỉ định hoặc được đặt thành một để chỉ điểm cuối IN với số điểm cuối đã chỉ định. Máy chủ phải luôn đặt bit hướng thành 0 (nhưng thiết bị phải chấp nhận một trong hai giá trị) khi điểm cuối là một phần của ống điều khiển.																																	
					Khi bmRequestType chỉ định giao diện là người nhận, định dạng của trường này như sau:																																	
					<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: center;"><b>D<sub>7</sub></b></td> <td style="text-align: center;"><b>D<sub>6</sub></b> –</td> <td style="text-align: center;"><b>D<sub>5</sub></b> –</td> <td style="text-align: center;"><b>D<sub>4</sub></b></td> <td style="text-align: center;"><b>D<sub>3</sub></b> –</td> <td style="text-align: center;"><b>D<sub>2</sub></b></td> <td style="text-align: center;"><b>d<sub>1</sub></b></td> <td style="text-align: center;"><b>D<sub>0</sub></b></td> </tr> <tr> <td colspan="8" style="text-align: center;">Số giao diện</td> </tr> </table>	<b>D<sub>7</sub></b>	<b>D<sub>6</sub></b> –	<b>D<sub>5</sub></b> –	<b>D<sub>4</sub></b>	<b>D<sub>3</sub></b> –	<b>D<sub>2</sub></b>	<b>d<sub>1</sub></b>	<b>D<sub>0</sub></b>	Số giao diện																								
<b>D<sub>7</sub></b>	<b>D<sub>6</sub></b> –	<b>D<sub>5</sub></b> –	<b>D<sub>4</sub></b>	<b>D<sub>3</sub></b> –	<b>D<sub>2</sub></b>	<b>d<sub>1</sub></b>	<b>D<sub>0</sub></b>																															
Số giao diện																																						

				D15 —	D14 —	D13 —	D12 —	D11 —	D10 —	D 9	D 8
Dành riêng (đặt lại về 0)											
6	chiều dài	2	Đếm	Số byte cần truyền nếu có giai đoạn DATA.	<ul style="list-style-type: none"> <li>Nếu trường này khác 0 và bmRequestType biểu thị chuyển từ thiết bị sang máy chủ, thì thiết bị không bao giờ được trả lại nhiều hơn wLength byte dữ liệu. Tuy nhiên, một thiết bị có thể trả lại ít hơn.</li> <li>Nếu trường này khác không và bmRequestType biểu thị chuyển từ máy chủ sang thiết bị, thì máy chủ phải gửi byte dữ liệu có độ dài chính xác. Nếu máy chủ gửi nhiều hơn wLength byte, hành vi của thiết bị không được xác định.</li> </ul>						

Khi một thiết bị nhận được một yêu cầu không xác định, không phù hợp với cài đặt hoặc trạng thái hiện tại của thiết bị hoặc sử dụng các giá trị không phù hợp cho yêu cầu cụ thể, thì Lỗi yêu cầu sẽ **tồn** tại. Thiết bị xử lý Lỗi yêu cầu bằng cách trả lại STALL PID cho giai đoạn DỮ LIỆU hoặc TÌNH TRẠNG tiếp theo, tốt nhất là ở giao dịch giai đoạn DỮ LIỆU tiếp theo.

## Yêu cầu tiêu chuẩn

Các yêu cầu tiêu chuẩn được xác định cho tất cả các thiết bị USB và tất cả các thiết bị USB phải đáp ứng các yêu cầu tiêu chuẩn này ngay cả khi thiết bị chưa được chỉ định địa chỉ hoặc thiết bị chưa được định cấu hình. Để đưa ra một yêu cầu nhất định, phần mềm tạo gói DATA của giai đoạn SETUP bằng cách sử dụng **mã yêu cầu** thích hợp, bmRequestType hợp lệ, các giá trị tham số phù hợp (hoặc 0, nếu không áp dụng) cho wValue và wIndex và lượng byte dữ liệu sẽ được truyền cho chiều dài. Ở bên phải là các mã yêu cầu thiết bị USB tiêu chuẩn và phần còn lại của phần này thảo luận về từng yêu cầu.

### SET\_ADDRESS

Yêu cầu SET\_ADDRESS có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
00000000b	SET_ADDRESS 5	Địa chỉ thiết bị	Số không	Số không

#### Định dạng gói dữ liệu SET\_ADDRESS SETUP

Yêu cầu này không có giai đoạn DỮ LIỆU, chỉ có giai đoạn THIẾT LẬP và TÌNH TRẠNG.

wValue chỉ định địa chỉ được gán cho thiết bị. Hành vi của thiết bị không được xác định khi wValue chỉ định địa chỉ lớn hơn 127.

Hành vi chính xác của thiết bị sau khi yêu cầu SET\_ADDRESS phụ thuộc vào trạng thái hiện tại của thiết bị:

- Khi một thiết bị ở trạng thái mặc định, wValue khác 0 sẽ khiến thiết bị chuyển sang trạng thái địa chỉ. Khi một thiết bị ở trạng thái mặc định, wValue bằng 0 không có hiệu lực.
- Khi một thiết bị ở trạng thái địa chỉ, wValue khác 0 sẽ giữ thiết bị ở trạng thái Địa chỉ, nhưng thiết bị sẽ phản hồi địa chỉ mới được đặt. Khi một thiết bị ở trạng thái địa chỉ, wValue bằng 0 sẽ chuyển thiết bị sang trạng thái mặc định.
- Khi thiết bị ở trạng thái cấu hình, hoạt động của thiết bị không được xác định cho yêu cầu SET\_ADDRESS.

bYêu cầu	Giá trị
GET_STATUS	0
CLEAR_FEATURE	1
Kin đáo	2
SET_FEATURE	3
Kin đáo	4
SET_ADDRESS	5
GET_DESCRIPTOR	6
SET_DESCRIPTOR	7
GET_CONFIGURATION	số 8
SET_CONFIGURATION	9
GET_INTERFACE	10
SET_INTERFACE	11
SYNC_FRAME	12

#### Mã yêu cầu USB tiêu chuẩn

Đây là yêu cầu duy nhất hoàn tất sau khi giai đoạn TÌNH TRẠNG hoàn tất thành công. Sau khoảng thời gian đặt lại/tiếp tục khôi phục (10 mili giây), dự kiến thiết bị sẽ có thể hoàn thành giai đoạn TRẠNG THÁI của yêu cầu này trong vòng 50 mili giây. Sau khi giai đoạn TÌNH TRẠNG hoàn tất, thiết bị được phép có khoảng thời gian khôi phục 2 mili giây trước khi thiết bị phải có khả năng chấp nhận các gói CÀI ĐẶT xa hơn cho các yêu cầu bổ sung.

## GET\_DESCRIPTOR

Yêu cầu GET\_DESCRIPTOR có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
10000000b	GET_DESCRIPTOR 6	Loại mô tả	Chỉ mục mô tả	Số không hoặc ID ngôn ngữ

**Định dạng gói dữ liệu GET\_DESCRIPTOR SETUP**

Loại mô tả	Giá trị
THIẾT BỊ	1
CÁU HÌNH	2
SƠI DÂY	3
GIAO DIỆN	4
KẾT THÚC	5
DEVICE_QUALIFIER	6
OTHER_SPEED_CONFIGURATION	7
INTERFACE_POWER	số 8

**Các loại mô tả USB tiêu chuẩn**

Byte thứ tự cao của *wValue* chỉ định loại bộ mô tả (xem bảng các loại bộ mô tả tiêu chuẩn usb, ở bên phải). Byte thứ tự thấp của *wValue* chỉ được sử dụng để chọn một bộ mô tả CHUỖI hoặc CÁU HÌNH cụ thể và phải được đặt lại về 0 nếu không.

Trường *wIndex* chỉ được sử dụng cho các bộ mô tả CHUỖI để chỉ định ngôn ngữ mong muốn và phải được đặt lại về 0 cho các loại bộ mô tả khác.

Các loại mô tả khác nhau có độ dài khác nhau sẽ sớm được thảo luận. Nếu *độ dài* nhỏ hơn kích thước của bộ mô tả được trả về, thì thiết bị chỉ trả về byte *độ dài* đầu tiên của dữ liệu bộ mô tả. Nếu *chiều dài* lớn hơn kích thước của bộ mô tả được trả về, thì bộ mô tả đầy đủ sẽ được trả về, theo sau là một gói ngắn (gói ngắn hơn kích thước tải trọng dữ liệu tối đa, bao gồm độ dài 0 byte).

Tất cả các thiết bị USB phải hỗ trợ các yêu cầu đối với các bộ mô tả THIẾT BỊ, CÁU HÌNH và CHUỖI. Tất cả các thiết bị tốc độ cao phải hỗ trợ các hoạt động cơ bản ở tốc độ tối đa; các thiết bị đó cũng hỗ trợ bộ mô tả DEVICE\_QUALIFIER và OTHER\_SPEED\_CONFIGURATION trả về cùng dữ liệu mà thiết bị sẽ trả về cho các yêu cầu bộ mô tả DEVICE và CONFIGURATION tương ứng nếu thiết bị đang hoạt động ở tốc độ mà thiết bị hiện không hoạt động.

Yêu cầu bộ mô tả CÁU HÌNH cũng trả về tất cả các bộ mô tả GIAO DIỆN cho chỉ mục bộ mô tả cấu hình đã chỉ định (nghĩa là byte thứ tự thấp của *wValue*), cũng như tất cả các bộ mô tả ENDPOINT được liên kết với tất cả các bộ mô tả GIAO DIỆN được trả về, tất cả trong một yêu cầu duy nhất.

GET\_DESCRIPTOR là một yêu cầu hợp lệ cho một thiết bị ở trạng thái mặc định, địa chỉ hoặc được định cấu hình.

## SET\_DESCRIPTOR

Yêu cầu SET\_DESCRIPTOR là tùy chọn; khi được hỗ trợ, nó có thể được sử dụng để cập nhật bộ mô tả hoặc thêm bộ mô tả mới.

Yêu cầu SET\_DESCRIPTOR có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
00000000b	SET_DESCRIPTOR 7	Loại mô tả	Chỉ mục mô tả	Số không hoặc ID ngôn ngữ

**SET\_DESCRIPTOR Định dạng gói dữ liệu SETUP**

Byte thứ tự cao của *wValue* chỉ định loại bộ mô tả. Byte thứ tự thấp của *wValue* chỉ được sử dụng để chọn một bộ mô tả CHUỖI hoặc CÂU HÌNH cụ thể và phải được đặt lại về 0 nếu không.

Trường *wIndex* chỉ được sử dụng cho các bộ mô tả CHUỖI để chỉ định ngôn ngữ mong muốn và phải được đặt lại về 0 cho các loại bộ mô tả khác.

Trường độ *dài* chỉ định số lượng byte sẽ được truyền từ máy chủ đến thiết bị.

Yêu cầu này chỉ hỗ trợ các loại bộ mô tả THIẾT BỊ, CÂU HÌNH và CHUỖI.

Nếu yêu cầu này không được hỗ trợ, thiết bị sẽ phản hồi với Lỗi Reuest.

Nếu yêu cầu này được hỗ trợ, nó chỉ hợp lệ khi thiết bị ở trạng thái địa chỉ hoặc được định cấu hình; hành vi của thiết bị không được xác định nếu yêu cầu này được thực hiện trong khi thiết bị ở trạng thái mặc định.

## **GET\_CONFIGURATION**

Yêu cầu GET\_CONFIGURATION có định dạng gói SETUP DATA sau:

<b>bmRequestType</b>	<b>bYêu cầu</b>	<b>giá trị w</b>	<b>w Index</b>	<b>chiều dài</b>
10000000b	NHẬN_CÂU HÌNH 8	Số không	Số không	Một

**GET\_CONFIGURATION SETUP DỮ LIỆU Định dạng gói**

Thiết bị gửi gói DATA một byte trong giai đoạn DATA của quá trình truyền điều khiển. Byte này là giá trị của cấu hình hiện tại của thiết bị. Giá trị bằng 0 cho biết thiết bị chưa được định cấu hình (thiết bị ở trạng thái địa chỉ). Hành vi của thiết bị không được xác định nếu yêu cầu này được đưa ra trong khi thiết bị ở trạng thái mặc định.

## **SET\_CONFIGURATION**

Yêu cầu SET\_CONFIGURATION có định dạng gói SETUP DATA sau:

<b>bmRequestType</b>	<b>bYêu cầu</b>	<b>giá trị w</b>		<b>w Index</b>	<b>chiều dài</b>
00000000b	SET_CONFIGURATION 9	Kín đáo	Giá trị cấu hình	Số không	Số không

**SET\_CONFIGURATION SETUP DATA Packet Format**

Byte thứ tự thấp của *wValue* chỉ định giá trị cấu hình mong muốn. Byte bậc thấp của *wValue* phải bằng 0 hoặc phải khớp với trường giá trị cấu hình của bộ mô tả cấu hình do thiết bị trả về. Chỉ định giá trị cấu hình bằng 0 sẽ đặt thiết bị vào trạng thái địa chỉ.

Nếu thiết bị ở trạng thái mặc định hoặc nếu byte thứ tự cao của *wValue* không bằng 0, *wIndex* không bằng 0 hoặc *wLength* không bằng 0, thì hành vi sau khi đưa ra yêu cầu này là không xác định.

Nếu giá trị cấu hình được chỉ định không phải là 0 cũng không phải là giá trị cấu hình hợp lệ được chỉ định bởi bộ mô tả cấu hình của thiết bị, thì thiết bị sẽ phản hồi với Lỗi Yêu cầu.

## **GET\_INTERFACE**

Yêu cầu GET\_INTERFACE có định dạng gói SETUP DATA sau:

<b>bmRequestType</b>	<b>bYêu cầu</b>	<b>giá trị w</b>	<b>w Index</b>	<b>chiều dài</b>

10000001b	NHẬN_INTERFACE 10	Số không	giao diện	Một
-----------	----------------------	----------	-----------	-----

**Định dạng gói dữ liệu GET\_INTERFACE SETUP**

Máy chủ sử dụng yêu cầu này để xác định cài đặt thay thế nào (như được mô tả trong Chức năng, Cấu hình, Giao diện và Điểm cuối) được sử dụng cho một giao diện cụ thể của cấu hình hiện tại. Thiết bị phản hồi bằng gói DATA dài một byte trong giai đoạn dữ liệu, byte được truyền là giá trị cài đặt thay thế cho giao diện được chỉ định trong yêu cầu này.

Nếu *wValue* khác 0, *wLength* không phải là 1, *wIndex* chỉ định giao diện không hợp lệ hoặc thiết bị ở trạng thái địa chỉ, thì thiết bị sẽ phản hồi với Lỗi yêu cầu.

Hành vi của thiết bị ở trạng thái mặc định sau khi nhận được yêu cầu này là không xác định.

Yêu cầu này hợp lệ đối với thiết bị ở trạng thái đã định cấu hình.

**SET\_INTERFACE**

Yêu cầu SET\_INTERFACE có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
00000001b	SET_INTERFACE 11	Cài đặt thay thế	giao diện	Số không

**Định dạng gói dữ liệu SET\_INTERFACE SETUP**

Máy chủ sử dụng yêu cầu này để chọn một cài đặt thay thế (như được mô tả trong Chức năng, Cấu hình, Giao diện và Điểm cuối) sẽ được sử dụng cho một giao diện cụ thể của cấu hình hiện tại. Nếu giao diện được chỉ định chỉ hỗ trợ cài đặt mặc định, thì thiết bị có thể trả về bắt tay STALL trong giai đoạn TÌNH TRẠNG của yêu cầu.

Nếu giao diện hoặc cài đặt thay thế không tồn tại hoặc nếu thiết bị ở trạng thái địa chỉ, thì thiết bị sẽ phản hồi với Lỗi Yêu cầu. Hành vi của thiết bị không được xác định nếu *wLength* khác 0 hoặc thiết bị ở trạng thái mặc định.

Đây là yêu cầu hợp lệ khi thiết bị ở trạng thái đã định cấu hình.

**CLEAR\_FEATURE**

Yêu cầu CLEAR\_FEATURE có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
00000000b	CLEAR_FEATURE	Zero		
00000001b		hoặc		
00000010b	1	Bộ		
		chọn		
		tính		
		năng		

Bộ chọn tính năng	Người nhận	Giá trị
DEVICE_REMOTE_WAKEUP	Thiết bị	1
ENDPOINT_HALT	điểm cuối	0
CHẾ ĐỘ KIỂM TRA	Thiết bị	2

**Bộ chọn tính năng USB tiêu chuẩn****CLEAR\_FEATURE SETUP DỮ LIỆU Định dạng gói**

Máy chủ sử dụng yêu cầu này để xóa hoặc tắt một tính năng cụ thể.

*wValue* phải chứa bộ chọn tính năng (xem bảng Bộ chọn tính năng USB tiêu chuẩn, ở bên phải) tương ứng với người nhận như được chỉ định trong giá trị *bmRequestType*.

Đưa ra yêu cầu này trong khi tham chiếu đến một tính năng không thể xóa hoặc không tồn tại hoặc tham chiếu đến một giao diện hoặc điểm cuối không tồn tại sẽ khiến thiết bị phản hồi với Lỗi yêu cầu.

Nếu thiết bị ở trạng thái mặc định hoặc *wLength* khác 0, thì hoạt động của thiết bị không được xác định.

Yêu cầu này hợp lệ khi thiết bị ở trạng thái được định cấu hình. Khi thiết bị ở trạng thái địa chỉ, yêu cầu này chỉ hợp lệ khi tham chiếu điểm cuối bằng 0, nếu không, thiết bị sẽ phản hồi với Lỗi yêu cầu.

Không thể xóa tính năng TEST\_MODE theo yêu cầu này.

## SET\_FEATURE

Yêu cầu SET\_FEATURE có định dạng gói SETUP DATA sau:

Giá trị	Sự miêu tả
00h	Kín đáo
01h	Kiểm tra_J
02h	Kiểm tra_K
03h	Kiểm tra_SE0_NAK
04h	Test_Packet
05h	Test_Force_Enable
06h-3Fh	Dành riêng cho bộ chọn thử nghiệm tiêu chuẩn
3Fh-BFh	Kín đáo
C0h-FFh	Dành riêng cho các chế độ kiểm tra dành riêng cho nhà cung cấp

Bộ chọn kiểm tra USB tiêu chuẩn

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
00000000b	SET_FEATURE			
00000001b				
00000010b	3	Bộ chọn tính năng	bộ chọn kiểm tra	Zero hoặc Giao diện hoặc Điểm cuối

## SET\_FEATURE SETUP DỮ LIỆU Định dạng gói

Máy chủ sử dụng yêu cầu này để đặt hoặc bật một tính năng cụ thể.

*wValue* phải chứa bộ chọn tính năng tương ứng với người nhận như được chỉ định trong giá trị *bmRequestType*.

Khi *wValue* chọn tính năng TEST\_MODE, cả *bmRequestType* và byte thứ tự thấp của *wIndex* đều phải được đặt lại về 0. Byte thứ tự cao của *wIndex* phải là bộ chọn kiểm tra hợp lệ (xem bảng Bộ chọn kiểm tra USB tiêu chuẩn, ở bên phải) nếu không thiết bị sẽ phản hồi với Lỗi yêu cầu. Thiết bị phải đặt cổng hướng lên của nó ở chế độ thử nghiệm không quá 3 mili giây sau khi hoàn thành giai đoạn TÌNH TRẠNG của yêu cầu này. Để thoát khỏi chế độ kiểm tra, phải cấp nguồn cho thiết bị. Thiết bị phải hỗ trợ tính năng TEST\_MODE ở trạng thái thiết bị tốc độ cao mặc định, địa chỉ và được định cấu hình.

Nếu yêu cầu này đề cập đến một tính năng không tồn tại hoặc không thể đặt được, thì thiết bị sẽ phản hồi bằng bắt tay STALL trong giai đoạn TÌNH TRẠNG.

Nếu một điểm cuối hoặc giao diện được chỉ định không tồn tại hoặc nếu thiết bị ở trạng thái địa chỉ và một điểm cuối khác với điểm cuối 0 được chỉ định, thì thiết bị sẽ phản hồi với Lỗi Yêu cầu.

Bên cạnh các yêu cầu chọn tính năng TEST\_MODE, việc đưa ra yêu cầu này cho một thiết bị ở trạng thái mặc định sẽ dẫn đến hành vi không xác định. Giá trị khác không cho *wLength* cũng dẫn đến hành vi không xác định.

Yêu cầu này hợp lệ khi thiết bị ở trạng thái được định cấu hình hoặc khi thiết bị ở trạng thái địa chỉ và chỉ có điểm cuối bằng 0 được tham chiếu.

## GET\_STATUS

Yêu cầu GET\_STATUS có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
10000000b	NHẬN_TRẠNG THÁI	Số không	Zero hoặc Giao diện hoặc Điểm cuối	
10000001b	0			
10000010b				Hai

**Định dạng gói dữ liệu GET\_STATUS SETUP**

Máy chủ sử dụng yêu cầu này để tìm hiểu trạng thái của người nhận, như được chỉ định bởi trường *bmRequestType* và, trong trường hợp người nhận giao diện hoặc điểm cuối, trường *wIndex*.

Nếu *wValue* không bằng 0, *wLength* không bằng 2 hoặc *wIndex* khác 0 khi *bmRequestType* chỉ định người nhận thiết bị hoặc thiết bị ở trạng thái mặc định, thì hoạt động của thiết bị sẽ không được xác định.

Nếu yêu cầu này tham chiếu đến một điểm cuối hoặc giao diện không tồn tại (bao gồm bất kỳ điểm cuối nào khác điểm cuối 0 khi thiết bị ở trạng thái địa chỉ), thì thiết bị sẽ phản hồi với Lỗi yêu cầu.

Để đáp ứng yêu cầu này, thiết bị sẽ thực hiện truyền dữ liệu dài 2 byte trong giai đoạn DATA tới máy chủ. Hai byte này biểu thị trạng thái được yêu cầu và ý nghĩa phụ thuộc vào loại người nhận.

### Người nhận thiết bị

Khi người nhận là một thiết bị, hai byte mô tả trạng thái như sau:

D <sub>7</sub>	D <sub>6</sub> –	D <sub>5</sub> –	D <sub>4</sub>	D <sub>3</sub> –	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>
Dành riêng (đặt lại về 0)						Đánh thức từ xa	tự cấp nguồn
D <sub>15</sub> –	D <sub>14</sub> –	D <sub>13</sub> –	D <sub>12</sub> –	D <sub>11</sub> –	D <sub>10</sub> –	d <sub>9</sub>	D <sub>8</sub>
Dành riêng (đặt lại về 0)							

Trường *Self Powered* được đặt thành 1 để cho biết thiết bị hiện đang được cấp nguồn bởi nguồn điện bên ngoài hoặc 0 để cho biết thiết bị hiện đang chạy bằng nguồn do xe buýt cung cấp.

Trường *Đánh thức từ xa* được đặt thành 0 khi thiết bị được đặt lại và cho biết liệu thiết bị hiện có được bật để thực hiện báo hiệu đánh thức từ xa hay không (xem Khả năng đánh thức từ xa). Máy chủ có thể sửa đổi giá trị của trường *Đánh thức từ xa* bằng cách đưa ra yêu cầu CLEAR\_FEATURE hoặc SET\_FEATURE bằng bộ chọn tính năng DEVICE\_REMOTE\_WAKEUP.

### Người nhận giao diện

Khi người nhận là một giao diện, hai byte mô tả trạng thái như sau:

D <sub>7</sub>	D <sub>6</sub> –	D <sub>5</sub> –	D <sub>4</sub>	D <sub>3</sub> –	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>
Dành riêng (đặt lại về 0)							

<b>D15</b> -	<b>D14</b> -	<b>D13</b> -	<b>D12</b> -	<b>D11</b> -	<b>D10</b> -	<b>D<sub>9</sub></b>	<b>D<sub>8</sub></b>
<i>Dành riêng (đặt lại về 0)</i>							

### Người nhận điểm cuối

Khi người nhận là điểm cuối, hai byte mô tả trạng thái như sau:

<b>D<sub>7</sub></b>	<b>D<sub>6</sub></b> -	<b>D<sub>5</sub></b> -	<b>D<sub>4</sub></b>	<b>D<sub>3</sub></b> -	<b>D<sub>2</sub></b>	<b>d<sub>1</sub></b>	<b>D<sub>0</sub></b>
<i>Dành riêng (đặt lại về 0)</i>							Tạm dừng lại
<b>D15</b> -	<b>D14</b> -	<b>D13</b> -	<b>D12</b> -	<b>D11</b> -	<b>D10</b> -	<b>D<sub>9</sub></b>	<b>D<sub>8</sub></b>
<i>Dành riêng (đặt lại về 0)</i>							

Tất cả các loại điểm cuối ngắt và hàng loạt phải triển khai tính năng tạm dừng, nếu không thì tính năng này là tùy chọn. Trường *Dừng* phản ánh trạng thái của tính năng tạm dừng của điểm cuối. Giá trị 0 trong trường *Dừng* cho biết điểm cuối không bị tạm dừng và giá trị 1 trong trường *Dừng* cho biết điểm cuối bị tạm dừng.

Máy chủ lưu trữ có thể đặt tính năng tạm dừng của điểm cuối bằng yêu cầu SET\_FEATURE bằng cách sử dụng bộ chọn tính năng ENDPOINT\_HALT hoặc máy chủ lưu trữ có thể xóa tính năng tạm dừng của điểm cuối bằng yêu cầu CLEAR\_FEATURE bằng cách sử dụng bộ chọn tính năng ENDPOINT\_HALT. Khi yêu cầu CLEAR\_FEATURE được sử dụng theo cách này và điểm cuối sử dụng bit chuyển đổi dữ liệu, bit chuyển đổi dữ liệu được đặt lại về 0.

Óng điều khiển mặc định không bắt buộc cũng như không được khuyến nghị triển khai tính năng tạm dừng, nhưng một số thiết bị có thể chọn sử dụng tính năng tạm dừng trên óng điều khiển mặc định để phản ánh tình trạng lỗi chức năng. Khi tính năng tạm dừng được đặt trên đường dẫn điều khiển mặc định, thiết bị sẽ phản hồi bằng tay STALL trong giai đoạn DỮ LIỆU hoặc TÌNH TRẠNG của tất cả các lần truyền, ngoại trừ các yêu cầu tiêu chuẩn GET\_STATUS, CLEAR\_FEATURE và SET\_FEATURE. Ngoài ra, thiết bị không bắt buộc phải dừng các yêu cầu của nhà cung cấp lớp cụ thể khi tính năng tạm dừng được đặt.

### SYNCH\_FRAME

Yêu cầu SYNCH\_FRAME có định dạng gói SETUP DATA sau:

bmRequestType	bYêu cầu	giá trị w	w Index	chiều dài
10000010b	SYNCH_FRAME 12	Số không	điểm cuối	Hai

**Định dạng gói dữ liệu SYNCH\_FRAME SETUP**

Yêu cầu này chỉ được sử dụng cho các điểm cuối đăng thời sử dụng dòng bộ hóa mẫu ẩn. Nghĩa là, một số điểm cuối đăng thời yêu cầu truyền trên mỗi khung hình phải thay đổi kích thước theo một mẫu cụ thể (ví dụ: để đạt được tốc độ bit dành riêng cho ứng dụng). Cuộc gọi này khiến thiết bị gửi cho máy chủ một giá trị 2 byte là số khung nơi mẫu bắt đầu.

Các điểm cuối đăng thời tốc độ cao hỗ trợ yêu cầu này phải đồng bộ hóa với vi khung thứ 0, cũng như có khái niệm thời gian của các khung cổ điển (1 mili giây so với khoảng thời gian 125 micro giây).

Nếu *wValue* không bằng 0, *wLength* không bằng 2 hoặc thiết bị ở trạng thái mặc định, thì hoạt động của thiết bị không được xác định.

Nếu điểm cuối được chỉ định không hỗ trợ yêu cầu này hoặc thiết bị ở trạng thái địa chỉ, thì thiết bị sẽ phản hồi với Lỗi yêu cầu.

Yêu cầu này hợp lệ khi thiết bị ở trạng thái được định cấu hình.

## Bộ mô tả USB tiêu chuẩn

Một **bô mô tả** là một cấu trúc dữ liệu có định dạng xác định. Tất cả các bộ mô tả tiêu chuẩn bắt đầu bằng hai byte. Byte đầu tiên chỉ định tổng chiều dài của bộ mô tả tính bằng byte, bao gồm hai byte bắt buộc đầu tiên. Byte thứ hai xác định loại bộ mô tả.

Một số bộ mô tả chứa các trường chỉ định chỉ mục của bộ mô tả CHUỖI, nhưng thiết bị không bắt buộc phải hỗ trợ bộ mô tả CHUỖI. Nếu một thiết bị không hỗ trợ bộ mô tả CHUỖI thì tất cả các trường tham chiếu chỉ mục của bộ mô tả CHUỖI phải được đặt lại về 0. Do đó, giá trị bằng 0 trong bất kỳ trường nào nhằm cung cấp chỉ mục của bộ mô tả CHUỖI chỉ ra rằng không có bộ mô tả CHUỖI nào như vậy.

Nếu byte thứ hai của bộ mô tả xác định bộ mô tả đó là một trong các bộ mô tả USB tiêu chuẩn, nhưng byte đầu tiên của bộ mô tả đó chỉ định độ dài nhỏ hơn độ dài được xác định trong thông số kỹ thuật của USB 2.0 (và, tạm thời, ở đây), thì bộ mô tả sẽ bị chủ từ chối. Nếu trường độ dài báo cáo rằng bộ mô tả dài hơn dự kiến, thì dữ liệu bổ sung sẽ bị bỏ qua, nhưng vẫn được coi là một phần của bộ mô tả (điều này rất quan trọng khi thiết bị trả về nhiều bộ mô tả, như trường hợp máy chủ yêu cầu CẤU HÌNH bộ mô tả).

Nếu bộ mô tả dành riêng cho lớp hoặc nhà cung cấp sử dụng cùng định dạng với bộ mô tả tiêu chuẩn (nghĩa là hai byte bắt buộc ở đầu bộ mô tả), thì bộ mô tả dành riêng cho lớp hoặc nhà cung cấp sẽ được xen kẽ trong kết quả khi máy chủ yêu cầu CẤU HÌNH bộ mô tả. Mặt khác, các bộ mô tả dành riêng cho lớp hoặc nhà cung cấp được truy cập bằng cách chuyển loại bộ mô tả dành riêng cho lớp hoặc nhà cung cấp trong yêu cầu GET\_DESCRIPTOR.

Phần còn lại của phần này dùng để lập danh mục các bộ mô tả thiết bị USB tiêu chuẩn và phản ánh rất chặt chẽ phần 9.6 của thông số kỹ thuật USB 2.0. Các định nghĩa bộ mô tả này bổ sung cho yêu cầu GET\_DESCRIPTOR.

## THIẾT BỊ

Mỗi thiết bị USB có chính xác một bộ mô tả THIẾT BỊ. Bộ mô tả này cung cấp thông tin chung về thiết bị, cũng như thông tin áp dụng chung cho thiết bị và tất cả các cấu hình của thiết bị.

Bù lai	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte
1	bDescriptorType	1	Không thay đổi	THIẾT BỊ Loại mô tả
2	bcdUSB	2	TCN	Số phát hành thông số kỹ thuật USB ở dạng thập phân được mã hóa nhị phân (nghĩa là 2.10 được biểu thị bằng 210h). Xác định việc phát hành Thông số kỹ thuật USB với thiết bị và các bộ mô tả của nó tuân thủ.
4	bDeviceClass	1	Lớp học	<p>Mã lớp (được chỉ định bởi USB-IF)</p> <ul style="list-style-type: none"> <li>▪ Trường này được đặt lại về 0 nếu mỗi giao diện trong một cấu hình chỉ định thông tin lớp của riêng nó và các giao diện khác nhau hoạt động độc lập.</li> <li>▪ Giá trị của FFh trong trường này cho biết loại thiết bị dành riêng cho nhà cung cấp.</li> </ul>
5	bDeviceSubClass	1	Phân lớp	<p>Mã phân lớp (được chỉ định bởi USB-IF)</p> <ul style="list-style-type: none"> <li>▪ Mã phân lớp của một thiết bị được xác định bởi mã lớp của thiết bị đó.</li> <li>▪ Nếu bDeviceClass được đặt lại về 0, thì trường này cũng phải được đặt lại về 0.</li> </ul>

				<ul style="list-style-type: none"> <li>Khi <i>bDeviceClass</i> không được đặt thành FFh, thì tất cả các giá trị cho trường này được dành riêng để gán bởi USB-IF.</li> </ul>
6	<i>bDeviceProtocol</i>	1	giao thức	<p>Mã giao thức (do USB-IF gán)</p> <ul style="list-style-type: none"> <li>Mã giao thức của thiết bị được xác định bởi cả mã lớp và mã lớp con của thiết bị đó.</li> <li>Giá trị 00h trong trường này có nghĩa là thiết bị có thể chỉ định các giao thức dành riêng cho lớp trên cơ sở giao diện, mặc dù đây không phải là một yêu cầu.</li> <li>Nếu trường này được đặt thành FFh, thì thiết bị sẽ sử dụng giao thức dành riêng cho nhà cung cấp.</li> </ul>
7	<i>bMaxKích thước gói0</i>	1	Con số	Kích thước gói tối đa cho điểm cuối bằng 0 (8, 16, 32 hoặc 64 là các tùy chọn hợp lệ duy nhất)
số 8	<i>idVendor</i>	2	NHẬN DẠNG	ID nhà cung cấp (được chỉ định bởi USB-IF)
10	<i>idSản phẩm</i>	2	NHẬN DẠNG	ID sản phẩm (do USB-IF chỉ định)
12	thiết bị <i>bcd</i>	2	TCN	Số phát hành thiết bị ở dạng thập phân được mã hóa nhị phân
14	<i>iNhà sản xuất</i>	1	Mục lục	Chỉ mục của STRING bộ mô tả mô tả nhà sản xuất
15	<i>iSản phẩm</i>	1	Mục lục	Chỉ mục của STRING bộ mô tả mô tả sản phẩm
16	<i>iSerialNumber</i>	1	Mục lục	Chỉ mục của bộ mô tả CHUỖI mô tả số sê-ri của thiết bị
17	<i>bNumCấu hình</i>	1	Con số	Số cấu hình có thể

## DEVICE\_QUALIFIER

Một thiết bị có khả năng tốc độ cao có thông tin thiết bị khác nhau tùy thuộc vào tốc độ hoạt động của thiết bị, thì thiết bị đó cũng phải có bộ mô tả DEVICE\_QUALIFIER. Bộ mô tả này cung cấp thông tin về thiết bị sẽ thay đổi nếu thiết bị đang hoạt động ở tốc độ khác (nghĩa là khi thiết bị đang hoạt động ở tốc độ cao, bộ mô tả này cung cấp sự khác biệt nếu thiết bị đang hoạt động ở tốc độ tối đa và ngược lại). Bộ mô tả này loại bỏ các trường khỏi bộ mô tả THIẾT BỊ không phụ thuộc một cách hợp lý vào tốc độ của thiết bị (ví dụ: chỉ mục của bộ mô tả CHUỖI mô tả sản phẩm).

Nếu một thiết bị duy nhất ở tốc độ tối đa có trường *bcdUSB* ít nhất 0200h trong bộ mô tả DEVICE của nó nhận được yêu cầu về bộ mô tả DEVICE\_QUALIFIER, thì thiết bị đó phải phản hồi bằng Lỗi yêu cầu.

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte
1	<i>bDescriptorType</i>	1	Không thay đổi	DEVICE_QUALIFIER Loại mô tả
2	<i>bcdUSB</i>	2	TCN	<p>Số phát hành thông số kỹ thuật USB ở dạng thập phân được mã hóa nhị phân (nghĩa là 2,00 được biểu thị bằng 200h). Xác định việc phát hành Thông số kỹ thuật USB với thiết bị và các bộ mô tả của nó tuân thủ.</p> <p>Trường này phải có ít nhất 0200h.</p>
4	<i>bDeviceClass</i>	1	Lớp học	Mã lớp (được chỉ định bởi USB-IF)
5	<i>bDeviceSubClass</i>	1	Phân lớp	Mã phân lớp (được chỉ định bởi USB-IF)
6	<i>bDeviceProtocol</i>	1	giao thức	Mã giao thức (do USB-IF gán)
7	<i>bMaxKích thước gói0</i>	1	Con số	Kích thước gói tối đa cho điểm cuối bằng 0 (8, 16, 32 hoặc 64 là các tùy chọn hợp lệ duy nhất)

số 8	bNumCâu hình	1	Con số	Số câu hình có thể
9	bDành riêng	1	-	Dành riêng cho các mục đích sử dụng trong tương lai, phải bằng không.

## CÁU HÌNH

Tất cả các thiết bị USB đều có ít nhất một bộ mô tả CÁU HÌNH. Máy chủ lưu trữ có thể yêu cầu một bộ mô tả CÁU HÌNH cụ thể theo chỉ mục mô tả của nó, bộ mô tả này không dựa trên 0 và có các chỉ số đã sử dụng *bNumConfigurations* (như được trả về trong bộ mô tả THIẾT BỊ). Nghĩa là, các giá trị hợp lệ được sử dụng làm chỉ mục bộ mô tả khi yêu cầu bộ mô tả CÁU HÌNH là bất kỳ số nguyên nào trong phạm vi từ 0 đến *bNumConfigurations* - 1, bao gồm.

Mỗi bộ mô tả CÁU HÌNH có ít nhất một bộ mô tả GIAO DIỆN và mỗi bộ mô tả GIAO DIỆN có thể có tối đa 15 bộ mô tả ENDPOINT. Khi máy chủ yêu cầu một bộ mô tả CÁU HÌNH nhất định, thiết bị sẽ trả về bộ mô tả CÁU HÌNH ngay sau đó là bộ mô tả GIAO DIỆN đầu tiên, ngay sau đó là tất cả các bộ mô tả ENPOINT cho các điểm cuối mà giao diện xác định (có thể không có). Ngay sau đó là bộ mô tả INTERFACE tiếp theo nếu có, và sau đó là bộ mô tả ENPOINT của nó nếu có. Mẫu này tiếp tục cho đến khi tất cả thông tin trong phạm vi của câu hình cụ thể được truyền đi.

Khi một thiết bị có bộ mô tả dành riêng cho nhà cung cấp hoặc lớp phù hợp với định dạng bộ mô tả USB tiêu chuẩn (nghĩa là byte đầu tiên của bộ mô tả xác định độ dài của bộ mô tả và byte thứ hai xác định loại bộ mô tả), các bộ mô tả đó là cũng được trả về xen kẽ giữa các bộ mô tả CÁU HÌNH, GIAO DIỆN và ENDPOINT khi máy chủ yêu cầu một bộ mô tả CÁU HÌNH cụ thể. Do đó, phần mềm hệ thống không thể giả sử các bộ mô tả tiêu chuẩn liên tục như được ngụ ý trong đoạn trước; thay vào đó, phần mềm hệ thống nên kiểm tra loại bộ mô tả và bỏ qua bộ mô tả đó nếu nó không phải là bộ mô tả tiêu chuẩn. Phần mềm cũng nên kiểm tra xem các bộ mô tả tiêu chuẩn có báo cáo ít nhất là độ dài dự kiến hay không.

Lưu ý rằng chỉ mục bộ mô tả CÁU HÌNH không giống với giá trị *bConfigurationValue* trong bộ mô tả CÁU HÌNH. *bConfigurationValue* là giá trị mà máy chủ chuyển qua dưới dạng tham số với yêu cầu SET\_CONFIGURATION để chọn một câu hình cụ thể, trong khi điều này không thể thực hiện được bằng cách sử dụng chỉ mục mô tả CONFIGURATION.

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte
1	bDescriptorType	1	Không thay đổi	CÁU HÌNH Loại mô tả
2	wTổng chiều dài	2	Con số	Tổng độ dài kết hợp theo byte của tất cả các bộ mô tả được trả về cùng với yêu cầu cho bộ mô tả CÁU HÌNH này (bao gồm CÁU HÌNH, GIAO DIỆN, ENDPOINT, bộ mô tả dành riêng cho lớp và nhà cung cấp).
4	bNumGiao diện	1	Con số	Số lượng giao diện được hỗ trợ bởi câu hình này
5	bConfigurationValue	1	Con số	Giá trị mà khi được sử dụng làm đối số trong yêu cầu SET_CONFIGURATION, sẽ khiến thiết bị nhận câu hình được mô tả bởi bộ mô tả này.
6	iConfiguration	1	Mục lục	Chỉ mục của bộ mô tả CHUỖI mô tả câu hình này.
7	bmAttributes	1	Bitmap	Đặc điểm câu hình <div style="text-align: center;">   <b>D<sub>7</sub></b>      <i>Dành riêng, phải được đặt thành một vì lý do lịch sử</i>  <b>D<sub>6</sub></b>      <i>tự cung cấp năng lượng</i> </div> <ul style="list-style-type: none"> <li>—</li> <li>■ 0 = Thiết bị chạy bằng điện được cung cấp bởi xe buýt</li> <li>■ 1 = Thiết bị cung cấp nguồn điện cục bộ, nếu <i>bMaxPower</i> khác không, thiết bị cũng có thể sử dụng nguồn điện trên bus.</li> </ul>

			<b>D5</b>	<b>Đánh thức từ xa</b>
				<ul style="list-style-type: none"> <li>■ 0 = Đánh thức từ xa không được hỗ trợ</li> <li>■ 1 = Hỗ trợ đánh thức từ xa</li> </ul>
số 8	bMaxPower	1	mA	<p><b>D<sub>4...0</sub></b> <i>Dành riêng, đặt lại về 0</i></p> <p>Công suất tiêu thụ tối đa của thiết bị này từ xe buýt khi hoạt động đầy đủ và sử dụng cấu hình này.</p> <ul style="list-style-type: none"> <li>■ Được biểu thị bằng đơn vị 2mA (nghĩa là giá trị 50 trong trường này biểu thị 100mA).</li> <li>■ Một thiết bị báo cáo với trường <i>bmAttributes</i> xem cấu hình là bus hay tự cấp nguồn, nhưng trạng thái thiết bị (được truy xuất bằng yêu cầu GET_STATUS) báo cáo liệu thiết bị hiện có tự cấp nguồn hay không.</li> <li>■ Nếu một thiết bị bị ngắt kết nối khỏi nguồn điện bên ngoài, nó không thể lấy nhiều điện hơn từ xe buýt so với quy định trong trường này.</li> <li>■ Một số thiết bị có thể chỉ hoạt động bằng nguồn điện trên bus. Một thiết bị không thể và đã mất nguồn điện bên ngoài sẽ không thực hiện được các hoạt động mà nó không thể hỗ trợ nữa. Tùy thuộc vào phần mềm trên máy chủ để xác định khi nào xảy ra trường hợp này, điều này có thể được thực hiện bằng yêu cầu GET_STATUS.</li> </ul>

## OTHER\_SPEED\_CONFIGURATION

Bộ mô tả này mô tả cấu hình của một thiết bị tốc độ cao nếu nó đang hoạt động ở tốc độ thay thế. Máy chủ không được yêu cầu bộ mô tả này trừ khi nó đã nhận thành công bộ mô tả DEVICE\_QUALIFIER từ thiết bị. Cấu trúc của OTHER\_SPEED\_CONFIGURATION giống với cấu trúc của bộ mô tả CONFIGURATION được hiển thị ở trên. Điểm khác biệt duy nhất là trường *bDescriptorType* phản ánh rằng bộ mô tả là bộ mô tả OTHER\_SPEED\_CONFIGURATION chứ không phải là bộ mô tả CONFIGURATION.

## GIAO DIỆN

Các bộ mô tả GIAO DIỆN chỉ được trả về sau một bộ mô tả CÁU HÌNH khi máy chủ yêu cầu một bộ mô tả CÁU HÌNH cụ thể; không thể trực tiếp yêu cầu một bộ mô tả GIAO DIỆN cụ thể. Một giao diện có thể cung cấp các cài đặt thay thế trong một cấu hình cho phép thay đổi các điểm cuối và/hoặc các đặc điểm của chúng. Một giao diện mặc định có trường *bAlternateSetting* trong bộ mô tả GIAO DIỆN của nó được đặt lại về 0.

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte
1	bDescriptorType	1	Không thay đổi	Loại mô tả GIAO DIỆN
2	bGiao diệnSố	1	Con số	Số giao diện này. Giá trị dựa trên số 0 xác định chỉ mục của giao diện này trong mảng giao diện được hỗ trợ trong một cấu hình.
3	bAlternateSetting	1	Con số	Giá trị được sử dụng để chọn cài đặt thay thế được mô tả bởi bộ mô tả GIAO DIỆN này cho giao diện có bInterfaceNumber trong trường trước đó. Giá trị này bằng 0 nếu bộ mô tả này cài đặt mặc định cho một giao diện cụ thể.
4	bNumĐiểm cuối	1	Con số	Số lượng điểm cuối được giao diện này sử dụng, không bao gồm điểm cuối bằng 0.
5	bGiao diệnLớp	1	Lớp học	Mã lớp (được chỉ định bởi USB-IF)

				<ul style="list-style-type: none"> <li>Giá trị bằng 0 ở đây được dành riêng cho tiêu chuẩn hóa trong tương lai.</li> <li>Nếu giá trị này là FFh, lớp giao diện dành riêng cho nhà cung cấp.</li> <li>Tất cả các giá trị khác được dành riêng để gán bởi USB-IF.</li> </ul>
6	bInterfaceSubClass	1	Phân lớp	<p>Mã phân lớp (được gán bởi USB-IF)</p> <ul style="list-style-type: none"> <li>Mã phân lớp trong trường này đủ điều kiện theo giá trị của trường <i>bInterfaceClass</i>.</li> <li>Nếu <i>bInterfaceClass</i> được đặt lại về 0, thì trường này cũng phải được đặt lại về 0.</li> <li>Nếu <i>bInterfaceClass</i> không được đặt thành giá trị của FFh, thì tất cả các giá trị của trường này được dành riêng để gán bởi USB-IF.</li> </ul>
7	giao thức bInterface	1	giao thức	<p>Mã giao thức (do USB-IF gán)</p> <ul style="list-style-type: none"> <li>Mã giao thức trong trường này đủ điều kiện bởi các giá trị của trường <i>bInterfaceClass</i> và <i>bInterfaceSubClass</i>.</li> <li>Nếu một giao diện hỗ trợ các yêu cầu dành riêng cho lớp, thì trường này xác định các giao thức mà thiết bị sử dụng như được xác định bởi các thông số kỹ thuật của lớp thiết bị.</li> <li>Nếu trường này được đặt lại về 0, thì thiết bị không sử dụng giao thức dành riêng cho lớp trên giao diện này.</li> <li>Nếu trường này được đặt thành FFh, thì thiết bị sẽ sử dụng giao thức dành riêng cho nhà cung cấp trên giao diện này.</li> </ul>
số 8	iGiao diện	1	Mục lục	Chỉ mục của bộ mô tả CHUỖI mô tả giao diện này

## KẾT THÚC

Mỗi điểm cuối được sử dụng cho một giao diện cụ thể có một bộ mô tả theo sau bộ mô tả của giao diện cụ thể đó khi máy chủ yêu cầu một bộ mô tả CÂU HÌNH cụ thể; máy chủ không thể yêu cầu một bộ mô tả ENDPOINT cụ thể một cách rõ ràng. Và bộ mô tả ENDPOINT không bao giờ mô tả điểm cuối bằng 0.

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả											
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte											
1	bDescriptorType	1	Không thay đổi	ENPOINT Loại mô tả											
2	bĐịa chỉ điểm cuối	1	điểm cuối	<p>Địa chỉ của điểm cuối trên thiết bị USB được mô tả bởi bộ mô tả này. Trường này có định dạng sau:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>D<sub>7</sub></td> <td>D<sub>6</sub></td> <td>–</td> <td>D<sub>5</sub></td> <td>–</td> <td>D<sub>4</sub></td> <td>D<sub>3</sub></td> <td>–</td> <td>D<sub>2</sub></td> <td>d<sub>1</sub></td> <td>D<sub>0</sub></td> </tr> </table> <p style="text-align: center;"><b>D<sub>7</sub></b>      <b>Hướng</b> (bỏ qua cho các điểm cuối điều khiển)</p> <ul style="list-style-type: none"> <li>0 = NGOÀI điểm cuối</li> <li>1 = điểm cuối IN</li> </ul> <p style="text-align: center;"><b>D<sub>6..4</sub></b>      <i>Dành riêng, đặt lại về 0</i></p> <p style="text-align: center;"><b>D<sub>3..0</sub></b>      <b>Số điểm cuối</b></p>	D <sub>7</sub>	D <sub>6</sub>	–	D <sub>5</sub>	–	D <sub>4</sub>	D <sub>3</sub>	–	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>
D <sub>7</sub>	D <sub>6</sub>	–	D <sub>5</sub>	–	D <sub>4</sub>	D <sub>3</sub>	–	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>					
3	bmAttributes	1	Bitmap	Trường này mô tả các thuộc tính của điểm cuối như sau:											

D <sub>7</sub>	D <sub>6</sub> -	D <sub>5</sub> -	D <sub>4</sub>	D <sub>3</sub> -	D <sub>2</sub>	d <sub>1</sub>	D <sub>0</sub>
----------------	---------------------	---------------------	----------------	---------------------	----------------	----------------	----------------

<b>D<sub>7..6</sub></b>	<b>Dành riêng, đặt lại về 0</b>
<b>D<sub>5..4</sub></b>	<b>Loại sử dụng</b> (Chỉ các điểm cuối đăng thời; dành riêng và đặt lại về 0 cho các điểm cuối khác)
	<ul style="list-style-type: none"> <li>■ 00 = Điểm cuối dữ liệu</li> <li>■ 01 = Điểm cuối phản hồi</li> <li>■ 10 = Điểm cuối dữ liệu phản hồi ẩn</li> <li>■ 11 = Dành riêng</li> </ul>
<b>D<sub>3..2</sub></b>	<b>Loại đồng bộ hóa</b> (Chỉ các điểm cuối đăng thời; dành riêng và đặt lại về 0 cho các loại điểm cuối khác)
	<ul style="list-style-type: none"> <li>■ 00 = Không đồng bộ hóa</li> <li>■ 01 = Không đồng bộ</li> <li>■ 10 = Thích ứng</li> <li>■ 11 = Đồng bộ</li> </ul>
<b>D<sub>1..0</sub></b>	<b>Loại chuyển khoản</b>
	<ul style="list-style-type: none"> <li>■ 00 = Kiểm soát</li> <li>■ 01 = Đăng thời</li> <li>■ 10 = Số lượng lớn</li> <li>■ 11 = Gián đoạn</li> </ul>

4	wMaxKích thước gói	2	Con số	Kích thước gói tối đa mà điểm cuối này có khả năng gửi hoặc nhận. <ul style="list-style-type: none"> <li>■ Đối với các điểm cuối đăng thời, giá trị này được sử dụng để dự trữ thời gian xe buýt; tuy nhiên, đường ống có thể không phải lúc nào cũng sử dụng hết thời gian dành riêng cho xe buýt.</li> <li>■ Bit 10...0 chỉ định kích thước gói tối đa tính bằng byte.</li> <li>■ Đối với các điểm cuối đăng thời gian và ngắt tốc độ cao, các bit 12...11 chỉ định số lượng cơ hội giao dịch bổ sung trên mỗi vi khung (xem Điểm cuối tốc độ cao, băng thông cao). Định dạng như sau:               <ul style="list-style-type: none"> <li>■ 00 = Không có (1 giao dịch trên mỗi vi khung)</li> <li>■ 01 = 1 bổ sung (2 giao dịch trên mỗi microframe)</li> <li>■ 10 = 2 bổ sung (3 giao dịch trên mỗi microframe)</li> <li>■ 11 = Dành riêng</li> </ul> </li> <li>■ Các bit 15...13 được dành riêng và phải được đặt lại về 0.</li> </ul>
6	bInterval	1	Con số	Khoảng thời gian để thăm dò thiết bị trong quá trình truyền dữ liệu, được biểu thị bằng đơn vị vi khung đối với thiết bị tốc độ cao và khung đối với thiết bị tốc độ thấp và tốc độ tối đa. Ý nghĩa chính xác của giá trị trong trường này phụ thuộc vào loại điểm cuối và tốc độ hoạt động của thiết bị: <ul style="list-style-type: none"> <li>■ Điểm cuối đăng thời tốc độ đầy đủ và tốc độ cao và điểm cuối ngắt tốc độ cao:               <ul style="list-style-type: none"> <li>■ Trường này phải nằm trong phạm vi từ 1 đến 16.</li> <li>■ Trường này được sử dụng để tính khoảng thời gian là <math>2^{bInterval - 1}</math>.</li> </ul> </li> <li>■ Nghĩa là, giá trị của 4 tính thành <math>2^{bInterval - 1} = 2^4 - 1 = 15</math>.</li> <li>■ Điểm cuối ngắt tốc độ đầy đủ và tốc độ thấp:               <ul style="list-style-type: none"> <li>■ Trường này phải nằm trong phạm vi từ 1 đến 255.</li> </ul> </li> <li>■ Điểm cuối OUT số lượng lớn và kiểm soát tốc độ cao:               <ul style="list-style-type: none"> <li>■ Trường này phải nằm trong khoảng từ 0 đến 255.</li> </ul> </li> </ul>

- Trường này chỉ định tỷ lệ NAK tối đa của điểm cuối.
- Giá trị bằng 0 cho biết điểm cuối không bao giờ NAK
- Các giá trị khác chỉ ra nhiều nhất 1 NAK cho mỗi *bInterval* số lượng vi khung.
- Xem Giao thức giao dịch PING

## SƠ ĐẦY

Các thiết bị có thể tùy ý hỗ trợ các bộ mô tả STRING. Nếu một thiết bị không hỗ trợ bộ mô tả CHUỖI, bất kỳ trường nào tham chiếu đến chỉ mục của bộ mô tả CHUỖI phải được đặt lại về 0. Bộ mô tả CHUỖI sử dụng mã hóa unicode và có thể hỗ trợ nhiều ngôn ngữ. Máy chủ yêu cầu bộ mô tả CHUỖI với yêu cầu GET\_DESCRIPTOR và phải chuyển LANGID 16 bit (như được xác định bởi USB-IF) của ngôn ngữ mong muốn trong trường *wIndex*. Danh sách các LANGID hiện được chấp nhận có tại đây ([http://www.usb.org/developers/docs/USB\\_LANGIDs.pdf](http://www.usb.org/developers/docs/USB_LANGIDs.pdf)).

Chỉ mục chuỗi 0 cho tất cả các ngôn ngữ trả về một bộ mô tả CHUỖI chứa một mảng gồm tất cả các mã LANGID hai byte mà thiết bị hỗ trợ.

Cho dù yêu cầu một chuỗi hay một mảng LANGID, dữ liệu không bị kết thúc bằng NULL. Thay vào đó, máy chủ xác định độ dài của dữ liệu bằng cách trừ 2 từ trường *bLength* của bộ mô tả.

Khi máy chủ yêu cầu chỉ số chuỗi 0, bộ mô tả sau được trả về:

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte
1	bDescriptorType	1	Không thay đổi	STRING Loại mô tả
2	wLangID[0]	2	Con số	mã LANGID bằng không
...	...	...	...	...
N	wLangID[x]	2	Con số	mã LANGID x

Khi máy chủ yêu cầu một chỉ mục chuỗi hợp lệ khác với chỉ mục chuỗi 0 cho LANGID được hỗ trợ, bộ mô tả sau được trả về:

Bù lại	Cánh đồng	Kích cỡ	Kiểu	Sự miêu tả
0	chiều dài	1	Con số	Kích thước của bộ mô tả này tính bằng byte
1	bDescriptorType	1	Không thay đổi	STRING Loại mô tả
2	bString	N	Con số	chuỗi Unicode

## Tổ chức điển hình của phần mềm hệ thống

Phần này thảo luận về cách phần mềm hệ thống được tổ chức hợp lý, điển hình. Phần này cũng phục vụ như một chỉ mục cho các mục wiki cung cấp hoặc sẽ cung cấp thêm thông tin và có thể là các ví dụ về lập trình.

### Trình điều khiển thiết bị USB

Như với bất kỳ trình điều khiển thiết bị nào, trình điều khiển thiết bị USB trừu tượng hóa khỏi các chi tiết cấp thấp về cách một thiết bị cụ thể đang được truy cập và cung cấp cho phần còn lại của hệ thống và ứng dụng một giao diện chung (ví dụ: trình quản lý tệp không cần biết liệu nó đang xử lý ổ cứng ngoài hay ổ cứng trong).

Trình điều khiển thiết bị USB thường triển khai một loại thiết bị nhất định theo thông số kỹ thuật thích hợp. Các loại thiết bị USB như vậy bao gồm, nhưng không giới hạn ở:

- Thiết bị lưu trữ dung lượng lớn USB
- Thiết bị đầu vào USB của con người

## Trình điều khiển USB

Ngay cả trình điều khiển thiết bị USB cũng không cần quan tâm đến một số chi tiết cấp thấp hơn. Chẳng hạn, trình điều khiển thiết bị sẽ không thành vấn đề nếu một thiết bị được kết nối trực tiếp với trung tâm gốc hoặc nếu thiết bị đó nằm sau 3 trung tâm. Trình điều khiển thiết bị không nên lo lắng về việc thiết bị cần bao nhiêu năng lượng từ xe buýt. Đây là nơi trình điều khiển USB xuất hiện.

Trình điều khiển USB về cơ bản cung cấp giao diện khung USB cho trình điều khiển thiết bị. Trình điều khiển USB cũng xử lý các sự kiện kết nối và ngắt kết nối trên USB, cũng như xác định trình điều khiển thiết bị nào là cần thiết (theo mã Lớp, Lớp con và Giao thức) và liệu trình điều khiển thiết bị đó có tồn tại hay không.

## Trình điều khiển trung tâm USB

Mặc dù Trình điều khiển USB biết một số chi tiết về địa hình USB, nhưng trách nhiệm giao tiếp dành riêng cho từng trung tâm (bao gồm cả các giao dịch phân tách) thường được tách khỏi Trình điều khiển USB thành một mô-đun khác có tên là Trình điều khiển Trung tâm USB.

Tùy thuộc vào thiết kế của hệ thống, Trình điều khiển USB có thể bỏ qua Trình điều khiển Hub USB khi giao tiếp với các thiết bị trên hub gốc hoặc hệ thống có thể sử dụng địa chỉ dành riêng là 0 để chỉ ra hub gốc cho Trình điều khiển Hub USB (có vẻ như Linux làm điều này).

Chi tiết về Hub USB cuối cùng sẽ được thảo luận trong mục wiki USB Hubs .

## Trình điều khiển máy chủ

Khi yêu cầu truyền dữ liệu di chuyển từ trình điều khiển thiết bị, qua Trình điều khiển USB và qua Trình điều khiển USB Hub, yêu cầu sẽ nhận được tất cả thông tin cần thiết cho bộ điều khiển máy chủ để tạo các giao dịch thích hợp trên xe buýt. Tuy nhiên, tùy thuộc vào bộ điều khiển máy chủ, thông tin này cần được định dạng theo một cách nhất định và được thêm vào để bộ điều khiển máy chủ lên lịch.

Nhiệm vụ này nếu được giao cho trình điều khiển bộ điều khiển máy chủ. Các yêu cầu đến trình điều khiển bộ điều khiển máy chủ ở định dạng do hệ thống xác định, thường được gọi là Khôi yêu cầu USB (URB) hoặc Gói yêu cầu I/O (IRP).

Ngoài ra, trình điều khiển bộ điều khiển máy chủ được hệ thống con PCI tải khi bộ điều khiển máy chủ tương ứng được phát hiện trong quá trình liệt kê PCI. Do đó, trình điều khiển bộ điều khiển máy chủ cũng chịu trách nhiệm khởi tạo bộ điều khiển máy chủ và có thể tải Trình điều khiển USB Hub và trình điều khiển USB. Được kết hợp, trình điều khiển USB, trình điều khiển bộ chia USB và trình điều khiển bộ điều khiển máy chủ tạo nên một hệ thống con USB.

## Xem thêm

### Liên kết ngoại

- [USB.org \(http://www.usb.org/home\)](http://www.usb.org/home)
- Thông số kỹ thuật USB Universal Serial Bus Revision 2.0 (<https://usb.org/document-library/usb-20-specification>)
- Bản sửa đổi Universal Serial Bus 3.2 Thông số kỹ thuật (<https://usb.org/document-library/usb-32-specification-released-september-22-2017-and-ecns>)
- Bản sửa đổi thông số kỹ thuật USB không dây 1.1 ([http://www.usb.org/developers/wusb/wusb1\\_1\\_20100910.zip](http://www.usb.org/developers/wusb/wusb1_1_20100910.zip))
- Nhân Linux (<http://www.kernel.org/>) (mọi thứ có xu hướng khó hiểu ở đó, cộng với việc bạn phải cẩn thận với việc tự học từ các nguồn Linux nếu dự án của bạn không được GPL'ed).
- USB trong NutShell (<http://www.beyondlogic.org/usbnutshell/usb1.shtml>) cũng có thể khiến bạn quan tâm. Có vẻ như một hướng dẫn thực sự tốt cung cấp tất cả kiến thức cần thiết để hiểu bất kỳ tài liệu/mã nguồn USB nào khác trong một vài trang HTML
- LANGID hiện được chấp nhận ([http://www.usb.org/developers/docs/USB\\_LANGIDs.pdf](http://www.usb.org/developers/docs/USB_LANGIDs.pdf))
- USB được làm đơn giản (<http://www.usbmadesimple.co.uk/index.html>)
- USB: Bus nối tiếp vạn năng ([https://www.fysnet.net/the\\_universal\\_serial\\_bus.htm](https://www.fysnet.net/the_universal_serial_bus.htm)) là cuốn sách viết trình điều khiển thiết bị/hệ thống cho UHCI, OHCI, EHCI và xHCI với nhiều thiết bị ví dụ khác nhau và mã nguồn có sẵn.

Lấy từ " [https://wiki.osdev.org/index.php?title=Universal\\_Serial\\_Bus&oldid=27651](https://wiki.osdev.org/index.php?title=Universal_Serial_Bus&oldid=27651) "

Thể loại : [USB](#) | [Xe buýt](#)

---

- Trang này được sửa đổi lần cuối vào ngày 3 tháng 3 năm 2023, lúc 10:07.
- Trang này đã được truy cập 347.259 lần.