

Un paseo por

los Dispositivos de

Almacenamiento Masivo USB

Soporte Técnico OEM

FUJITSU ESPAÑA

Marzo de 2003

INDICE

Referencias

PARTE 1: Un paseo por las Clases USB

Introducción

¿Qué es una Clase USB?

Relación Driver-Dispositivo

Descriptores

Clases, Subclases y Protocolos

Localización del driver

Peticiones específicas de Clase USB y peticiones específicas del fabricante

PARTE 2: Un paseo por los Dispositivos de Almacenamiento Masivo USB

Introducción

Códigos de Clase, Subclase y Protocolo

Arrancar el sistema desde un Dispositivo de Almacenamiento USB

El protocolo de transporte Bulk-Only

- **Peticiones de Clase**
- **Transferencias**
- **Estructura del paquete CBW**
- **Estructura del paquete CSW**

El protocolo de transporte CBI (Control-Bulk-Interrupt)

- **Peticiones de Clase**
- **Transferencias**

Descriptores

- **Descriptor de Dispositivo**
- **Descriptor de Configuración**
- **Descriptor de Interfaz**
- **Descriptores de Endpoint**

Referencias

- Universal Serial Bus Specification. Revisión 2.0. Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC y Philips. 27 de Abril de 2000.
- Universal Serial Bus Common Class Specification. Revisión 1.0. SystemSoft, Intel. 16 de Diciembre de 1997.
- Universal Serial Bus Mass Storage Class Specification Overview. Revisión 1.1. 28 de Junio de 2000.
- USB Mass Storage Class Control/Bulk/Interrupt (CBI) Transport. Revisión 1.0. 14 de Diciembre de 1998.
- USB Mass Storage Class Bulk-Only Transport. Revisión 1.0. 31 de Septiembre de 1999
- Universal Serial Bus Mass Storage Class UFI Command Specification. Revision 1.0. 14 de Diciembre de 1998.
- SCSI Primary Commands - 2 (SPC-2) specifications (2ª generación del juego de comandos SCSI-3 para todo tipo de dispositivos). NCITS (National Committee for Information Technology Standards) Project T10/1236-D Revisión 3 o posterior.
- Reduced Block Commands (RBC) (juego de comandos SCSI-3 simplificado para dispositivos de bloque). NCITS Project T10/1240-D.
- Multi-Media Command Set - 2 (MMC-2) (2ª generación del juego de comandos SCSI-3 para dispositivos CD/DVD). NCITS Project T10/1228-D
- Advanced Technology Attachment Packet Interface (ATAPI) for Floppies. SFF-8070i.
- Advanced Technology Attachment Packet Interface (ATAPI) for CD-ROMs. SFF-8020i.
- Advanced Technology Attachment Packet Interface (ATAPI) for Tape. QIC-157.
- Un paseo por USB 1.1. Soporte Técnico OEM, Fujitsu España. Marzo de 2000
- Un paseo por USB 2.0. Soporte Técnico OEM, Fujitsu España. Marzo de 2003

PARTE 1: Un paseo por las Clases USB

Introducción

Una Clase USB define un grupo de dispositivos de similares características, cuyos requisitos vienen definidos mediante una Especificación de Clase USB.

Las distintas Especificaciones de Clase USB permiten que los fabricantes puedan desarrollar dispositivos que pueden controlarse mediante drivers adaptativos (drivers que pueden controlar a los dispositivos en función de la propia información descriptiva proporcionada por el dispositivo). Los drivers adaptativos compatibles con una determinada Clase se denominan Drivers de Clase.

De esta manera, los fabricantes de Sistemas Operativos y otras casas de software pueden desarrollar distintos Drivers de Clase, con la finalidad de poder controlar dispositivos pertenecientes a cualquiera de dichas Clases, sin necesidad de que el fabricante del dispositivo tenga que desarrollar también los drivers para cada entorno operativo en que quiera que funcione su dispositivo.

¿Qué es una Clase USB?

Desde el punto de vista de USB, una Clase es un grupo de dispositivos (o interfaces dentro de un dispositivo) con ciertas características en común. Típicamente, dos dispositivos pertenecen a la misma Clase si ambos utilizan formatos similares en los datos que reciben o transmiten, o si ambos utilizan una misma forma de comunicarse con el sistema.

El uso principal de una Clase USB es la de describir la forma en que un interfaz se comunica con el sistema, tanto a nivel de datos como a nivel de control. También existe un uso secundario, que es el de proporcionar información sobre la funcionalidad que proporciona dicho interfaz. De esta manera, la información de Clase proporcionada por el dispositivo puede utilizarse para que el sistema localice un driver que pueda controlar tanto la conectividad entre el interfaz y el sistema, como la propia funcionalidad del dispositivo.

Relación Driver-Dispositivo

USB define una relación entre drivers y dispositivos totalmente diferente a la filosofía tradicional. En vez de permitir que el driver tenga acceso directo al hardware del dispositivo, USB sólo permite al driver comunicarse con el dispositivo a través de las “pipes” establecidas entre el sistema USB y los distintos endpoints del dispositivo. Una vez establecidas las pipes, el Sistema Operativo las pone a disposición del driver en forma de interfaces software. Los tipos de transferencias a través de dichas pipes dependen del tipo de endpoint, y pueden ser de 4 tipos: Bulk, Control, Interrupción e Isócrono.

Por esta razón, las Clases USB se basan en la forma en que el dispositivo o interfaz se comunica con el sistema, y no simplemente en el tipo de servicio proporcionado por el dispositivo. Por ejemplo, en la Clase de Dispositivos de Impresión no interesa cuántos cartuchos de tinta o qué colores soporta la impresora, sino si se envían los datos a través de una pipe tipo Bulk-OUT y si tiene o no una pipe tipo Bulk-IN para reportar información de estado. Asimismo, en la Clase de Dispositivos de Almacenamiento Masivo no interesa si se trata de un disco duro o de un disquete, ni el número de cabezas o cilindros, ni siquiera la capacidad del dispositivo. Lo que interesa es si las lecturas y escrituras se van a realizar a través de pipes tipo Bulk-IN y Bulk-OUT o a través de una pipe de Control, y si se va a utilizar una pipe de Interrupción para reportar información de estado o si se realiza mediante otros mecanismos.

Las Clases USB también pueden definir el formato de los datos que se transmiten. Por ejemplo, la Clase de Dispositivos de Almacenamiento Masivo define varios métodos opcionales para encapsular (transportar) distintos juegos de comandos estándares, en los paquetes de datos que se transfieren a través de las pipes. Un dispositivo concreto puede soportar uno o varios de dichos métodos de transporte, y uno o varios juegos de comandos estándar (SCSI, UFI, ATA, ATAPI, etc.), de forma que cuando el sistema

Un paseo por los Dispositivos de Almacenamiento Masivo USB

lee la información proporcionada por el dispositivo, puede buscar y asociar un Driver de Clase compatible con alguno de los métodos de transporte y juegos de comandos que el dispositivo soporta.

Descriptores

Desde el punto de vista del sistema USB, un dispositivo puede tener varias posibles Configuraciones, en cada una de las cuales el dispositivo puede funcionar de una manera distinta. En cada una de las posibles Configuraciones, el dispositivo queda organizado como un conjunto de Interfaces, donde cada Interfaz especifica qué partes del hardware del dispositivo interactúa con el sistema USB. Cada una de esas partes de hardware se denomina Endpoint. Entonces, de una manera jerárquica, un dispositivo es una colección de posibles Configuraciones, cada Configuración es una colección de Interfaces, y cada Interfaz es una colección de Endpoints. A su vez los Interfaces pueden admitir configuraciones alternativas, con distintas colecciones de Endpoints en cada una de ellas.

Los dispositivos proporcionan toda la información descriptiva al sistema a través de unas estructuras de datos denominados Descriptores. Existen distintos descriptores que proporcionan información a nivel de dispositivo, de configuración, de interfaz y de endpoint.

Las especificaciones de Clase USB definen las configuraciones, interfaces (y sus configuraciones alternativas) y endpoints que los dispositivos pertenecientes a dicha Clase o Subclase deben soportar.

Clases, Subclases y Protocolos

Los descriptores de dispositivo y de interfaz contienen una serie de campos que permiten al sistema clasificar a los dispositivos. Estos campos son la Clase, la Subclase y el Protocolo.

El Sistema Operativo puede utilizar estos campos para localizar y asociar al dispositivo o interfaz un determinado Driver de Clase, de entre todos los Drivers de esa Clase disponibles en el sistema. También puede seleccionar una determinada configuración del dispositivo, o una determinada configuración alternativa de un interfaz, en función de los protocolos soportados por los distintos Drivers de Clase disponibles en el sistema para esa Clase y Subclase de dispositivo.

Localización del driver

En algunas ocasiones, sólo es necesario un driver para controlar a un dispositivo, mientras que en otras son necesarios distintos drivers para controlar los distintos interfaces disponibles en el dispositivo.

Se entiende que debe existir una manera estándar de localizar y asociar drivers a dispositivos e interfaces, de manera que los fabricantes de dispositivos y de Sistemas Operativos trabajen según un modelo común.

Una vez seleccionada una configuración, queda establecido el número de interfaces. Las características concretas de cada interfaz pueden seleccionarse posteriormente a través de las posibles configuraciones alternativas.

El algoritmo para localizar y asociar un driver se basa en la información recibida del dispositivo en los Descriptores. La primera búsqueda se basa en la información recibida en el Descriptor de Dispositivo, y se trata de localizar un único driver que controle todo el dispositivo. La información en la que se basa esta primera búsqueda es (en orden de prioridad):

- Fabricante & Producto & Versión del producto.
- Fabricante & Producto

Si no se ha localizado un driver y el campo Clase indica que el dispositivo pertenece a una Clase Específica del Fabricante, es decir, no pertenece a una Clase estándar USB, la búsqueda continúa según los campos:

Un paseo por los Dispositivos de Almacenamiento Masivo USB

- Fabricante & Subclase & Protocolo
- Fabricante & Subclase

Si en cambio el dispositivo pertenece a una Clase estándar USB, la búsqueda continúa en función de los campos:

- Clase & Subclase & Protocolo
- Clase & Subclase

Si en este proceso ya se ha localizado un driver, este driver ya puede participar en la elección de la configuración en la que debe funcionar el dispositivo.

Si no se ha podido localizar un driver, el Sistema Operativo es responsable de seleccionar una configuración para el dispositivo, y seguir buscando un driver para cada interfaz disponible en dicha configuración. Esta segunda búsqueda se basa en la información recibida en los descriptores de dispositivo y de interfaz. De nuevo en orden de prioridad, los campos utilizados en esta segunda fase son:

- Fabricante & Producto & Versión del producto & Número de la configuración & Número del interfaz
- Fabricante & Producto & Número de la configuración & Número del interfaz

Si no se ha localizado un driver y el interfaz pertenece a una Clase Específica del Fabricante, es decir, no pertenece a una Clase estándar USB, la búsqueda continúa según los campos:

- Fabricante & Subclase del interfaz & Protocolo del interfaz
- Fabricante & Subclase del interfaz

Si en cambio el interfaz pertenece a una Clase estándar USB, la búsqueda continúa en función de los campos:

- Clase del interfaz & Subclase del interfaz & Protocolo del interfaz
- Clase del interfaz & Subclase del interfaz

Peticiones específicas de Clase USB y peticiones específicas del fabricante

La norma USB denomina “peticiones” (requests) a las distintas funciones que el sistema USB puede solicitar a los dispositivos, lo cual es distinto de los comandos que las aplicaciones pueden enviar, y que dependerán del juego de comandos que se esté utilizando en concreto con cada dispositivo.

La norma USB define una serie de peticiones estándar que deben implementar todos los dispositivos, mientras que las especificaciones de Clase USB y los fabricantes de dispositivos pueden definir peticiones adicionales, denominadas respectivamente peticiones específicas de Clase y peticiones específicas del Fabricante.

La forma de enviar al dispositivo una petición USB es siempre a través de una Transferencia de Control dirigida a la pipe de Control por Defecto, en cuya fase de SETUP se indica el tipo de petición (Estándar, de Clase o de Fabricante) y el destinatario de la misma (el dispositivo, un interfaz o un endpoint).

Si la petición es Estándar, está definida en la propia norma USB, pero si es de Clase, la Clase a la que pertenece el destinatario de la petición indica en qué Especificación de Clase está definida dicha petición. Por ejemplo, si el destinatario es el dispositivo, entonces la Clase indicada en el descriptor del dispositivo indica la Especificación de Clase donde está definida la petición. Si el destinatario es un interfaz o endpoint, entonces la Clase indicada en el descriptor del interfaz indica la Especificación de Clase donde está definida la petición. Si la petición es de Fabricante, entonces es el propio fabricante quien ha definido dicha petición.

PARTE 2: Un paseo por los Dispositivos de Almacenamiento Masivo USB

Introducción

La especificación de Clase de Almacenamiento Masivo USB define cómo los dispositivos de almacenamiento masivo deben comportarse en general en el bus USB.

Las especificaciones Bulk-Only y CBI (Control-Bulk-Interrupt) definen dos protocolos de transporte de comandos, y permiten el desarrollo de dispositivos compatibles con la Clase de Almacenamiento Masivo USB. El fabricante del dispositivo puede elegir entre implementar uno u otro o ambos, mientras que los Sistemas Operativos normalmente soportan ambos.

Por otro lado, las especificaciones de Almacenamiento Masivo USB hacen uso de varios juegos de comandos estándar, y también se han desarrollado especificaciones de juegos de comandos expresamente para dispositivos de almacenamiento USB. Los comandos pertenecientes a estos juegos de comandos se envían encapsulados en paquetes USB especiales (USB wrapper, o envoltorio), que se envían siguiendo las reglas de protocolo USB.

Los juegos de comandos y protocolos estándar utilizados por las especificaciones de Almacenamiento Masivo USB son:

- SCSI
 - SCSI Primary Commands-2 (SPC-2). 2ª generación del juego de comandos SCSI-3 comunes para todo tipo de dispositivos.
 - Multi-Media Command Set-2 (MMC-2). 2ª generación del juego de comandos SCSI-3 para unidades de CD y DVD.
 - Reduced Block Commands (RBC). Juego de comandos SCSI-3 simplificado para dispositivos de bloques (discos duros, disquetes, discos ópticos, discos magneto-ópticos, etc.).
 - Adicionalmente, el dispositivo también puede soportar comandos SCSI de forma transparente (pertenecientes a cualquier juego de comandos SCSI).
- ATAPI
 - ATAPI para unidades de disquete, SFF-8070i (SFF = Small Form Factor Committee)
 - ATAPI para unidades de CD-ROM, SFF-8020i
 - ATAPI para unidades de cinta, QIC-157 (QIC = Quarter Inch Cartridge).

Las especificaciones expresas de juegos de comandos para dispositivos de Almacenamiento Masivo USB son:

- USB Mass Storage Class UFI Command Specification (UFI = USB Floppy Interface). Diseñado para unidades de disquete y basado en los juegos de comandos SCSI-2 y SFF-8070i.
- USB Mass Storage Class ATA Command Block. Basado en el juego de comandos ATA. Esta especificación se encuentra en fase de desarrollo, por lo que todavía no está disponible.

Códigos de Clase, Subclase y Protocolo

Los dispositivos de esta Clase devuelven los campos Clase, Subclase y Protocolo del descriptor de dispositivo a 00h, lo que significa que los valores reales se indican en el descriptor del interfaz.

El campo Clase en el descriptor del interfaz siempre se devuelve al valor 08h, que identifica a la Clase de Almacenamiento Masivo.

En el campo Subclase del descriptor del interfaz, el dispositivo devuelve un código que identifica el juego de comandos que soporta el interfaz. Los posibles códigos son:

Un paseo por los Dispositivos de Almacenamiento Masivo USB

Código de Subclase	Juego de Comandos	Comentarios
01h	Reduced Block Commands (RBC). T10 Project 1240-D	Normalmente utilizado por memorias Flash aunque cualquier dispositivo de almacenamiento masivo puede utilizarlo.
02h	SFF-8020i, MMC-2	Normalmente utilizado por unidades de CD y DVD
03h	QIC-157	Normalmente utilizado por unidades de cinta
04h	UFI	Normalmente utilizado por unidades de disquete
05h	SFF-8070i	Normalmente utilizado por unidades de disquete, aunque también pueden utilizar otra Subclase (como RBC), y otros dispositivos de almacenamiento masivo pueden utilizar esta Subclase.
06h	SCSI transparente (cualquiera de los juegos de comandos definidos en las normas SCSI)	Cualquier dispositivo de almacenamiento masivo puede utilizar esta Subclase
07h - FFh	Reservados para uso futuro	

En el campo Protocolo del descriptor del interfaz, el dispositivo devuelve un código que identifica el protocolo de transporte de comandos utilizado por el interfaz. Los posibles códigos son:

Código de Protocolo	Protocolo de Transporte de comandos
00h	CBI con interrupción de fin de comando
01h	CBI sin interrupción de fin de comando
50h	Bulk-Only
02h – 4Fh	Reservado
51h - FFh	Reservado

Arranque del sistema desde un Dispositivo de Almacenamiento USB

La posibilidad de cargar y arrancar un Sistema Operativo desde un dispositivo de almacenamiento USB no es algo que haya que tener especialmente en cuenta en la especificación de Clase ni en el diseño del dispositivo en sí. Cualquier dispositivo de almacenamiento USB puede servir como unidad de arranque del Sistema Operativo, siempre que alguien sepa acceder al dispositivo ANTES de haber cargado el Sistema Operativo.

Quien carga el Sistema Operativo desde el dispositivo de arranque es la BIOS del equipo, por lo que es la BIOS la que debe implementar las funciones básicas para leer del dispositivo de almacenamiento USB, igual que actualmente ya lo hace para arrancar desde un disquete, un CD-ROM o un disco local IDE o SCSI.

Otra posibilidad es la de que el Sistema Operativo permita crear un disquete o CD-ROM de arranque que incluya los drivers USB, de forma que después de un arranque inicial desde ese disquete o CD-ROM se tenga ya acceso al dispositivo de almacenamiento USB y se prosiga la carga del resto del Sistema Operativo instalado en el mismo. Actualmente sólo LINUX permite hacer esto. Ni Windows ni MacOS permiten generar disquetes o CD-ROMs de arranque que incluyan el soporte USB.

El protocolo de transporte Bulk-Only

Este protocolo transporta comandos, datos y estado exclusivamente a través de transferencias tipo Bulk.

Un paseo por los Dispositivos de Almacenamiento Masivo USB

El dispositivo USB sólo necesita entonces dos endpoints tipo Bulk (uno de entrada y otro de salida), aparte de los dos endpoints de Control obligatorios en la dirección 0, como especifica USB para establecer la pipe de Control por defecto. Aparte del uso de la pipe de Control establecido en la norma USB, este protocolo utiliza esta pipe exclusivamente para enviar las peticiones de Clase que el propio protocolo define, y para reestablecer las pipes Bulk tras una condición de STALL (bloqueo) en las mismas.

• Peticiones de Clase

El protocolo Bulk-Only define dos peticiones de Clase:

- Bulk-Only Mass Storage Reset. Se utiliza para reiniciar el dispositivo.
- Get Max LUN. Se utiliza para conocer cuántas unidades lógicas hay disponibles en el dispositivo.

• Transferencias

La información de comando (denominado Bloque de Comando) se encapsula en un paquete tipo DATA que contiene una estructura de datos especial, denominado CBW (Command Block Wrapper), y la información de estado se encapsula en otro paquete DATA que contiene otra estructura de datos similar, denominado CSW (Command Status Wrapper). Los datos se envían en paquetes DATA normales, del tamaño permitido por el endpoint. Las estructuras especiales de los paquetes CBW y CSW permiten al controlador y al dispositivo diferenciar estos paquetes de los paquetes DATA normales de datos.

Para realizar una escritura, el controlador USB envía primero el CBW mediante una transferencia a la pipe Bulk-OUT, y a continuación envía, también a la pipe Bulk-OUT, tantas transferencias de paquetes de datos, del tamaño permitido por el endpoint, como sean necesarias para transportar los datos. Una vez enviados todos los datos, el controlador USB solicita una transferencia a la pipe Bulk-IN, para que el dispositivo devuelva la información de estado en un paquete CSW.

Para realizar una lectura, el controlador USB envía primero el CBW mediante una transferencia a la pipe Bulk-OUT, y a continuación solicita a la pipe Bulk-IN sucesivas transferencias de paquetes de datos, del tamaño permitido por el endpoint, hasta recibir en una de dichas transferencias un paquete CSW.

El protocolo no permite el encolado de comandos, por lo que el controlador USB no enviará ningún nuevo CBW a un dispositivo hasta no haber recibido el CSW correspondiente al último comando enviado a dicho dispositivo.

Tampoco se permiten transferencias bidireccionales, por lo que todas las transferencias de datos entre el CBW y el CSW serán en la misma dirección (todas OUT o todas IN).

• Estructura del paquete CBW

La estructura de datos del CBW comienza exactamente al principio de un paquete de datos, y termina como un paquete corto de exactamente 31 bytes.

La estructura de datos se compone de los siguientes campos:

CBWSignature. Se trata de una firma para identificar el paquete de datos como un CBW. El contenido de la firma son los caracteres ASCII 43-42-53-55h (USBC = USB Command, en formato little-endian). El dispositivo identifica que el paquete es un CBW por esta firma y porque el paquete tiene una longitud de 31 bytes.

CBWTag. Se trata de un número asignado por el controlador al comando. El dispositivo devuelve este mismo número en el paquete CSW, para identificar claramente a qué CBW pertenece el CSW.

Un paseo por los Dispositivos de Almacenamiento Masivo USB

CBWDataTransferLength. Aquí se indica el número de datos (bytes) que el controlador espera que se transfieran durante la fase de datos. Si este campo se envía a cero, ni el controlador ni el dispositivo envían paquetes de datos entre el CBW y el CSW.

CBWFlags. En este campo se indica el sentido de la transferencias de datos (Data-Out, desde controlador a dispositivo, o Data-In, desde dispositivo a controlador).

CBWLUN. Aquí se indica a qué Unidad Lógica va dirigido este comando.

CBWCBLength. Aquí se indica la longitud (en bytes) del Bloque de Comando.

CBWCB. En este campo se sitúa el Bloque de Comando que debe ejecutar el dispositivo. Este comando pertenecerá al juego de comandos indicado por el contenido del campo Subclase del descriptor de la interfaz. Este campo tiene una longitud de 16 bytes, aunque el juego de comandos utilizado puede usar comandos de menor longitud, o incluso comandos de distintas longitudes. En cualquier caso, la longitud del Bloque de Comando viene especificada en el campo anterior, y el dispositivo ignora el resto de los bytes de este campo.

• Estructura del paquete CSW

La estructura de datos del CSW comienza exactamente al principio de un paquete de datos, y termina como un paquete corto de exactamente 31 bytes.

La estructura de datos se compone de los siguientes campos:

CSWSignature. Se trata de una firma para identificar el paquete de datos como un CSW. El contenido de la firma son los caracteres ASCII 53-42-53-55h (USBS = USB Status, en formato little-endian). El controlador USB identifica que el paquete es un CSW por esta firma y porque el paquete tiene una longitud de 31 bytes.

CSWTag. El dispositivo devuelve en este campo el valor recibido en el campo CBWTag del CBW correspondiente. De esta manera el controlador comprueba a qué CBW corresponde este CSW.

CSWDataResidue. Para transferencias Data-Out, el dispositivo indica aquí la diferencia entre los datos que esperaba recibir (indicado en el campo *CBWDataTransferLength* del CBW) y los que realmente se han recibido (el controlador ha podido transferir menos datos que los propuestos inicialmente). Para transferencias Data-In, el dispositivo indica en este campo la diferencia entre los datos que el controlador esperaba recibir (indicado en el campo *CBWDataTransferLength* del CBW) y los que realmente ha transmitido el dispositivo (el dispositivo ha podido transmitir menos datos que los esperados por el controlador). Estas pueden ser situaciones normales, contempladas en el protocolo del juego de comandos que se esté utilizando.

CSWStatus. En este campo se indica si el comando se ha ejecutado correctamente o si se ha producido algún error. Este error, a nivel de ejecución del comando, puede tener muchas posibles causas. El controlador puede confirmar la causa concreta mediante los mecanismos proporcionados por el juego de comandos que se esté utilizando. Por ejemplo, si se está utilizando SCSI o ATAPI, tras un error en la ejecución de un comando el controlador puede enviar el comando Request Sense para que el dispositivo envíe los 3 octetos que codifican el tipo de error concreto (Sense Key, Additional Sense Code y Additional Sense Code Qualifier), junto con información adicional relativa al error concreto.

El protocolo de transporte CBI (Control-Bulk-Interrupt)

Este protocolo hace uso de endpoints de tipos Control, Bulk e Interrupción para establecer la comunicación entre el controlador y el dispositivo.

- La pipe de Control se utiliza tanto para enviar las peticiones estándar USB al dispositivo, como para enviar los comandos, a través de una petición específica de Clase definida en este protocolo.

Un paseo por los Dispositivos de Almacenamiento Masivo USB

- Las pipes Bulk-In y Bulk-Out se utilizan para transportar los datos.
- La pipe de Interrupción se usa para que el transporte de la información de fin de comando.

• Peticiones de Clase

El protocolo CBI define una petición de Clase:

- Accept Device Specific Command (ADSC). Es el paquete donde se encapsula el Bloque de Comando para su transporte sobre USB hacia el dispositivo.

• Transferencias

El controlador envía primero el Bloque de Comando mediante una Transferencia de Control dirigida a la pipe de Control por defecto.

- En la fase de SETUP se especifica el código de la petición de Clase ADSC.
- En la fase de DATOS se envía el Bloque de Comando
- En la fase de ESTADO el dispositivo puede indicar la finalización de la Transferencia de Control o puede indicar el estado de la ejecución del comando.

A continuación, el controlador USB solicita a la pipe Bulk-In (lecturas) o envía a la pipe Bulk-Out (escrituras) tantas transferencias de paquetes de datos, del tamaño soportado por el endpoint, como sean necesarias.

Por último, el dispositivo puede enviar la información de estado de varias posibles maneras, en función de que se trate de lecturas o escrituras o de comandos sin transferencia de datos, y de si el dispositivo implementa el aviso de fin de comando mediante la pipe de interrupción o no. Los posibles estados pueden ser Comando-en-progreso, Comando-correctamente-ejecutado y Fallo-en-comando.

- En caso de comandos sin transferencia de datos:
 1. Si el dispositivo no implementa la interrupción de fin de comando, indica el estado durante las fases de DATOS o de ESTADO de la Transferencia de Control, devolviendo ACK (Comando-correctamente-ejecutado), NAK (Comando-en-progreso) o STALL (Fallo-en-comando).
 2. Si el dispositivo implementa la interrupción de fin de comando, responde con ACK para terminar la Transferencia de Control (aunque opcionalmente puede indicar el estado de Comando-en-progreso mediante NAK). Una vez terminada la transferencia de Control, el controlador USB solicita a la pipe de Interrupción la transferencia del estado, a lo que el dispositivo puede devolver NAK (Comando-en-progreso), o puede enviar un paquete de datos con información detallada sobre el fin de comando (en los casos de Comando-correctamente-ejecutado y de Fallo-en-comando).
- En el caso de comandos con transferencias de datos:

Aparte de lo anterior, el dispositivo puede opcionalmente devolver NAK (Comando-en-progreso) o STALL (Fallo-en-comando) en respuesta a las solicitudes del controlador USB a la pipe Bulk-In (lecturas) o de los envíos a la pipe Bulk-Out (escrituras).

Por último, no se permite el encolado de comandos. El controlador USB es responsable de encolar y ordenar las peticiones de forma que se envíen al dispositivo de una en una.

Descriptorios

Los dispositivos Bulk-Only y CBI soportan los siguientes descriptorios:

- **Dispositivo.** Por norma USB, todos los dispositivos disponen de un descriptor de dispositivo.

Un paseo por los Dispositivos de Almacenamiento Masivo USB

- **Configuración.** Por norma USB, todos los dispositivos disponen al menos de un descriptor de configuración por defecto, que soporta al menos un interfaz.

- **Interfaz.** Los dispositivos CBI y Bulk-Only disponen al menos de un interfaz, denominado Interfaz de Datos. Los dispositivos pueden disponer de otros interfaces para proporcionar cualquier otra funcionalidad adicional.

- **Endpoint.**

Los dispositivos Bulk-Only disponen de los siguientes descriptores de endpoint:

1. Descriptor de endpoint Bulk-In
2. Descriptor de endpoint Bulk-Out

Los dispositivos CBI disponen de los siguientes endpoints:

1. Descriptor de endpoint Bulk-In
2. Descriptor de endpoint Bulk-Out
3. Descriptor de endpoint de Interrupción (obligatorio si el dispositivo implementa la interrupción de fin de comando).

Los endpoints de control en la dirección 0 son obligatorios por norma USB, y no necesitan de un descriptor.

Los dispositivos Bulk-Only pueden disponer de otros endpoints en el interfaz para proporcionar cualquier otra funcionalidad adicional..

- **String.** Los dispositivos Bulk-Only disponen de un descriptor tipo string para proporcionar su número de serie.

- **Descriptor de Dispositivo**

Como ha quedado indicado anteriormente, los dispositivos de Almacenamiento Masivo especifican la Clase, Subclase y Protocolo en el descriptor de interfaz, por lo que indican el valor 00h en los campos correspondientes (Clase, Subclase y Protocolo) del descriptor de dispositivo.

- **Descriptor de Configuración**

En el campo Número de Interfaces, el dispositivo debe especificar como mínimo el valor 1, correspondiente al Interfaz de Datos.

En este descriptor el dispositivo informa de otras características, como si se alimenta desde el bus USB (Bus-Power) o si dispone de alimentación propia (Self-power), cuánta corriente consume del bus USB como máximo en esta configuración, y si el dispositivo puede utilizarse para despertar al sistema de un estado de reposo.

- **Descriptor de Interfaz**

El dispositivo dispone al menos del descriptor del Interfaz de Datos. Es posible que el interfaz admita configuraciones alternativas. En este caso, el sistema puede buscar y seleccionar una configuración del interfaz adecuada a la Subclase y Protocolo que mejor pueda manejar.

La siguiente tabla muestra los campos y valores más relevantes de este descriptor:

Un paseo por los Dispositivos de Almacenamiento Masivo USB

Campo	Valor	Comentario
Número de endpoints	xxh	Número de endpoints exceptuando los endpoints 0. Dispositivos Bulk-Only: al menos deben ser 2 Dispositivos CBI: 2 si no se usa la interrupción de fin de comando, 3 en caso de que se use.
Clase	08h	Identifica la Clase de Almacenamiento Masivo
Subclase	0xh	Código identificativo del juego de comandos soportado por el dispositivo en esta configuración. Ver el apartado Códigos de Clase, Subclase y Protocolo
Protocolo	00h	Transporte de comandos CBI con interrupción de fin de comando.
	01h	Transporte de comandos CBI sin interrupción de fin de comando.
	50h	Transporte de comandos Bulk-Only.

- **Descriptores de endpoint**

El dispositivo debe disponer al menos de los descriptores de endpoint correspondientes al Interfaz de Datos.

Cada descriptor de endpoint indica el tipo (Bulk, Interrupción) y la dirección del endpoint (del 1 al 15), si es de entrada (IN) o de salida (OUT) y el tamaño del paquete de datos soportado por el endpoint.