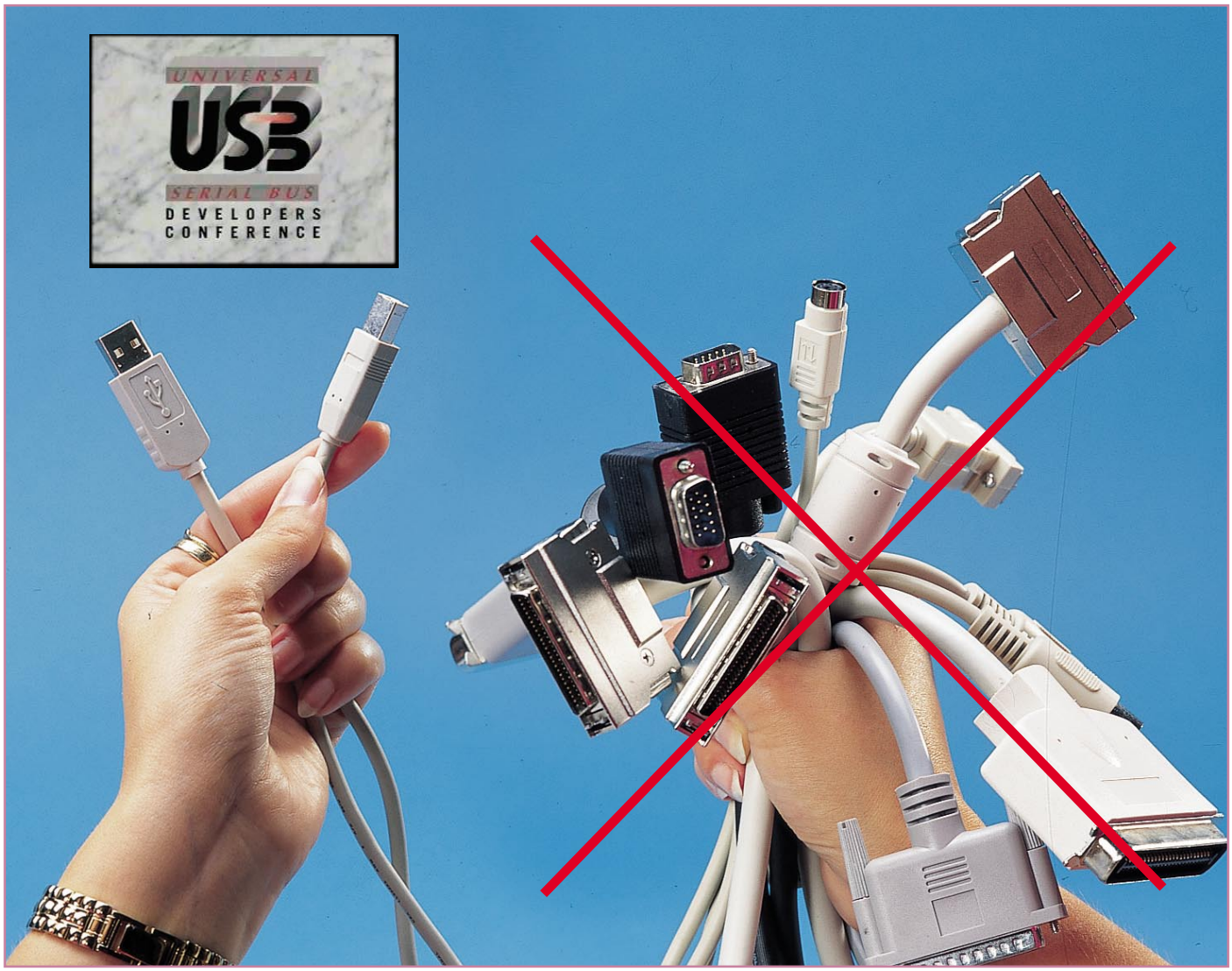


USB-Grundlagen Teil 2

Im ersten Teil der „USB-Grundlagen“ haben wir den USB im Hinblick auf die Anwendung am heimischen PC beschrieben, wobei die notwendigen Komponenten (Kabel, Hubs usw.) vorgestellt wurden. Der zweite Teil befasst sich etwas tiefer mit der Technik hinter dem „Universal Serial Bus“ und bietet somit einen interessanten Einblick in die USB-Technik. Die hier beschriebenen technischen Merkmale beziehen sich auf den USB-1.1-Standard.

Physikalischer Aufbau

Wie bereits im ersten Teil des Artikels beschrieben, ist der USB ein „single-master-bus“, der ausschließlich vom Host-Controller verwaltet wird. Der Anschluss der Geräte (Functions) erfolgt über den Root-Hub. Falls dessen Ports nicht ausreichen, ist eine Erweiterung über externe USB-Hubs realisierbar. Die Verbindungen werden über spezielle, vieradrige USB-Kabel hergestellt. Der Bus besteht aus jeweils einer Ader für Masse und Betriebsspannung sowie aus zwei Datenleitungen (D+, D–). Diese Datenleitungen werden differentiell angesteuert, wobei der Low-Pegel mit 0 V und der High-Pegel mit 3,3 V angegeben ist. Differentielle Übertragung bedeutet, dass der Empfänger die Spannungsdifferenz beider Leitungen auswertet und daraus den logischen Zustand ermittelt.

Eine differentielle Eins liegt vor, wenn die Spannungsdifferenz zwischen D+ und D– einen Wert größer als 200 mV aufweist (D+ minus D– > 200 mV).

Die differentielle Null wird erkannt, wenn die Spannungsdifferenz zwischen D– und D+ größer als 200 mV ist (D– minus D+ > 200 mV).

Es gibt beim USB auch bestimmte (definierte) Zustände, bei denen die differentielle Ansteuerung nicht gegeben ist, z. B. zur Auslösung eines „Reset“ des gesamten Busses.

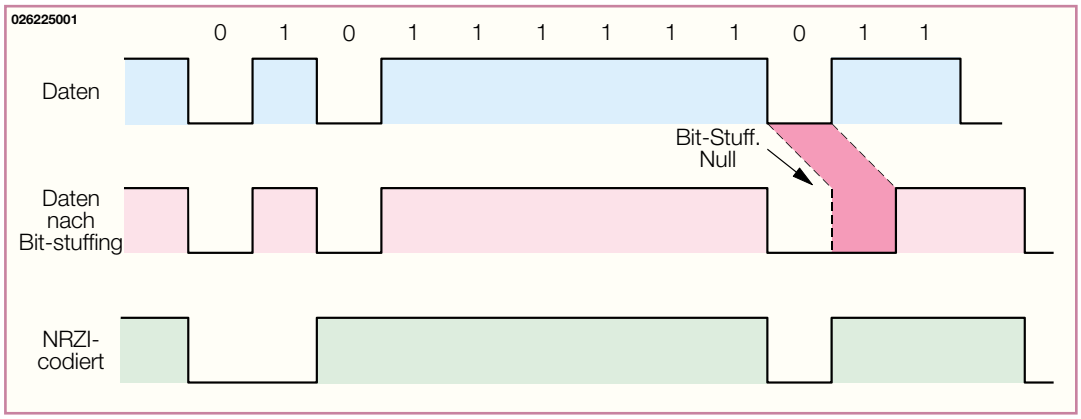


Bild 1: Codierung eines seriellen Datenstroms

Übertragungsverfahren

Die Daten werden beim „Universal Serial Bus“, wie der Name schon sagt, seriell übertragen. Es erfolgt jedoch eine Codierung nach dem NRZI-Verfahren (Non-Return-to-Zero-Inverted), durch das die Sicherheit der Datenübertragung erhöht wird und das eine Rekonstruktion des Übertragungstaktes erlaubt. Eine Null in den zu übertragenden Daten wird durch einen Wechsel des Zustands (High nach Low bzw. Low nach High) dargestellt, für eine Eins wird der aktuelle Zustand nicht geändert.

Es ergibt sich jedoch ein Problem, falls sich mehrere aufeinanderfolgende Einsen im Datenstrom befinden, da sich in diesem Fall der Zustand der Pegel auf den Datenleitungen für eine lange Zeit nicht ändert. Hierdurch ist die Rückgewinnung des Taktes nicht mehr gewährleistet und es erfolgt zwangsläufig der Verlust der Synchronisation zwischen Sender und Empfänger. Aus diesem Grund ergibt sich die Notwendigkeit des „Bit-Stuffers“, der nach sechs aufeinanderfolgenden Einsen automatisch eine Null in den Datenstrom einfügt. Die angepassten Daten aus dem Bit-Stuffer gelangen auf den NRZI-Encoder, der den Datenstrom auf den Bustreiber und damit schließlich auf die Leitung gibt. Diese Codierung erfolgt ausschließlich hardwaremäßig. Ein Beispiel für die Umsetzung eines seriellen Datenstroms ist in Abbildung 1 zu sehen. Im oberen Teil sind die reinen Daten dargestellt, die über den USB übertragen werden sollen, jedoch befinden sich in den Daten sechs aufeinanderfolgende Einsen. Aus diesem Grund muss eine „Bit-Stuff-Null“ eingefügt werden, wie im mittleren Teil der Grafik zu sehen. Der so umgeformte und NRZI-codierte Datenstrom ist im unteren Teil der Abbildung zu sehen.

USB-Kommunikationsmodell

Ein USB-System kann aus bis zu 127 angeschlossenen Geräten bestehen, die allesamt über zwei Leitungen (D+ bzw. D–) mit dem Host kommunizieren. Logisch gesehen gibt es jedoch eine Menge verschiedener Datenkanäle (Pipes), damit jedes Gerät einzeln angesprochen werden kann. Jeder dieser Datenkanäle bildet die Verbindung zwischen dem Host bzw. der Software und dem Datenendpunkt (Endpoint) in einem USB-Gerät. Ein solcher Endpunkt besteht aus einem FIFO-Speicher, der entweder Daten empfängt oder sendet. Er wird im USB-System eindeutig über die 7-Bit-Geräte- und 4-Bit-Endpunktadresse adressiert. Außerdem ist die Übertragungsrichtung ein Adressierungsmerkmal, da ein Endpunkt nur Daten vom Host empfangen (OUT-Richtung) oder an den Host senden kann (IN-Richtung). Eine Ausnahme bildet hier der Endpunkt 0 (EP0), der einzige bidirektionale Endpunkt, welcher auch als „Control-Endpoint” bezeichnet wird und in jedem USB-Gerät vorhanden sein muss. Es kann maximal 15 IN- und 15 OUT-Endpoints zusätzlich zu EP0 pro Gerät geben. Jedem Endpunkt kann eine von vier unterschiedlichen Transferarten zugeordnet werden, wobei EP0 hier wieder eine Ausnahme bildet, denn für diesen Endpunkt ist der „Control-Transfer“ festgelegt.

Datentransfer

Jeder Datentransfer auf dem USB wird durch den Host gesteuert, d. h. die USB-Geräte dürfen nur Daten übertragen, wenn sie vom Host dazu aufgefordert wurden. Es sind vier verschiedene Transferarten für den USB spezifiziert: Control-Transfer, Isochronous-Transfer, Interrupt-Transfer und Bulk-Transfer. Diese sollen im Folgenden kurz beschrieben werden.

Control-Transfer

Der Control-Transfer dient zur Konfiguration und Steuerung von USB-Functions und läuft immer über Endpoint 0 ab. Es werden nur geringe Datenmengen übertragen, jedoch sind immer 10 % der Bandbreite des Busses für den Control-Transfer reserviert und die Daten werden garantiert übertragen. In diesem Modus erfolgt eine Fehlererkennung und –korrektur der übertragenen Daten.

Isochronous-Transfer

In diesem Transfer-Modus werden Datenübertragungen durchgeführt, die eine gewisse garantierte Bandbreite erfordern, z. B. Audiodaten. Bis zu 90 % der Bandbreite des Busses kann für Isochrone- und Interrupt-Transfers reserviert werden. Es erfolgt jedoch keine Fehlerkorrektur der Daten. Einzelne fehlerbehaftete Bytes machen sich jedoch bei Audiodaten o. ä. kaum bemerkbar.

Dieser Transfermodus ist nur für den Full-Speed- bzw. High-Speed-Datenverkehr verfügbar.

Interrupt-Transfer

Der Interrupt-Transfer dient zur Übertragung kleiner Datenmengen zu einem nicht bestimmbaren Zeitpunkt. Beispiele für Interrupt-Transfers sind die Datenübertragungen einer Maus, einer Tastatur oder von Statusabfragen.

Die Unterbrechungsanforderung ist hier jedoch nicht mit einem konventionellen Interrupt zu vergleichen, der von einem externen Gerät ausgelöst wird, da ein USB-Gerät ohne Anforderung vom Host nichts übertragen darf. Der Interrupt-Transfer wird beim USB durch regelmäßiges Abfragen des Endpunktes durchgeführt. Diese Abfragen können pro Zeiteinheit (1 ms) und Endpunkt nur einmal erfolgen.

Bis zu 90 % der Bandbreite sind für Interrupt- und Isochrone-Transfers reserviert.

Bulk-Transfer

Mit dieser Transferart können große Datenmengen übertragen werden, wie sie bei Druckern oder Scannern anfallen. Es gibt keine garantierte Bandbreite und die Daten werden so übertragen, wie es die zur Zeit verfügbare Bandbreite zulässt. Sind die Kapazitäten des Busses zeitweise vollkommen ausgeschöpft, so erfolgt die Übertragung erst später. Der Bulk-Transfer ist also nicht für zeitkritische Daten geeignet.

Der Bulk-Transfer ist nur im Full- und High-Speed-Modus verfügbar.

Die verfügbare Bandbreite des „Univer-

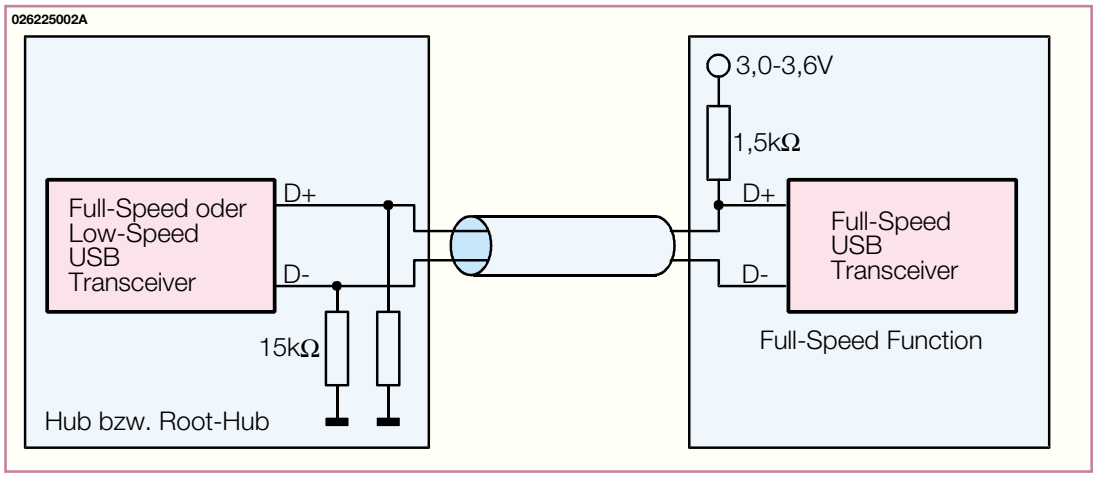


Bild 2: Anschluss- und Geschwindigkeitserkennung

sal Serial Bus“ wird in 1-ms-Zeitabschnitte (Frames) eingeteilt. Der Start eines solchen „Frames“ wird durch ein bestimmtes Datenpaket (Start-Of-Frame-Token) kenntlich gemacht, so dass alle Geräte den Beginn eines neuen Zeitabschnitts erkennen können. Zwischen diesen Zeitmarken erfolgt die Übertragung der einzelnen Pakete, welche wiederum immer mit einem Synchronisationsfeld beginnen.

Neben dem bereits genannten gibt es weitere Datenpakete mit verschiedenen Bedeutungen für die Datenübertragung. Außerdem gibt es „Data-Packets“, welche die entsprechenden Dateninformationen beinhalten, sowie „Handshake-Packets“, die als Quittung zurück zum Host gesendet werden.

Der Host sendet seine Datenpakete immer auf den kompletten Bus zu allen angeschlossenen Full-Speed-Geräten (Broadcast), jedoch darf nur das adressierte Gerät antworten. Alle Low-Speed-Geräte werden durch Hubs vor dem Full-Speed-Datenverkehr geschützt. Daten für Low-Speed-Geräte werden durch ein bestimmtes Token gekennzeichnet. Der Hub setzt diese Daten um und sendet sie mit Low-Speed-Geschwindigkeit an die angeschlossenen Geräte.

Hot-plug-and-play

USB-Geräte dürfen während des laufenden Betriebs des PCs mit diesem verbunden werden. Der PC erkennt diesen Vorgang automatisch und lädt den entsprechenden Treiber. Aber was steckt hinter diesem Mechanismus?

Die (D+)- und (D–)-Leitungen eines „Downstream“-Ports eines Hubs bzw. Root-Hubs führen Low-Pegel, sofern keine USB-Function angeschlossen ist. Zwei Pull-Down-Widerstände mit einem Wert von 15 kΩ halten die beiden Datenleitungen auf diesem definierten Pegel. Der Anschluss eines USB-Gerätes an diesen Port wird durch die Pegeländerung einer Datenleitung erkannt, die durch einen entsprechenden Pull-Up-Widerstand (1,5 kΩ) in der USB-Function erreicht wird. Diese Pegeländerung erkennt der USB-Hub und gibt dieses „Anschluss-Ereignis“ an den Host weiter, der nun die „Enumeration“ (Initialisierung) des Gerätes durchführt.

Der Pull-Up-Widerstand des Gerätes ermöglicht nicht nur die Anschlusserkennung, sondern kennzeichnet auch die Geschwindigkeitsklasse, mit der das Gerät vom Host angesprochen werden kann. Je nachdem, welche Datenleitung mit diesem Widerstand versehen ist, erfolgt die Unterscheidung der Geschwindigkeit (D+ = Full-Speed, D– = Low-Speed). Abbildung 2 zeigt die Anordnung der Widerstände bei dem Anschluss eines Full-Speed-Gerätes an den „Downstream“-Port eines Hubs.

Konfiguration von USB-Geräten

Die Initialisierung nach dem Anstecken eines USB-Gerätes wird als „Enumeration“ bezeichnet. Da der USB keine Punkt-zu-Punkt-Verbindung zwischen nur zwei Geräten ist, sondern ein Bus, an dem immer verschiedene Geräte angeschlossen sein können, ist es notwendig, jedem USB-Gerät eine individuelle Adresse zuzuweisen. Im nicht-initialisierten Zustand wird das neue Gerät über Adresse 0 angesprochen, bevor ihm eine Adresse zugewiesen wird. Über diese Nummer (1..127) ist das Gerät im USB eindeutig adressierbar.

Zur Verwaltung und zum Betrieb eines

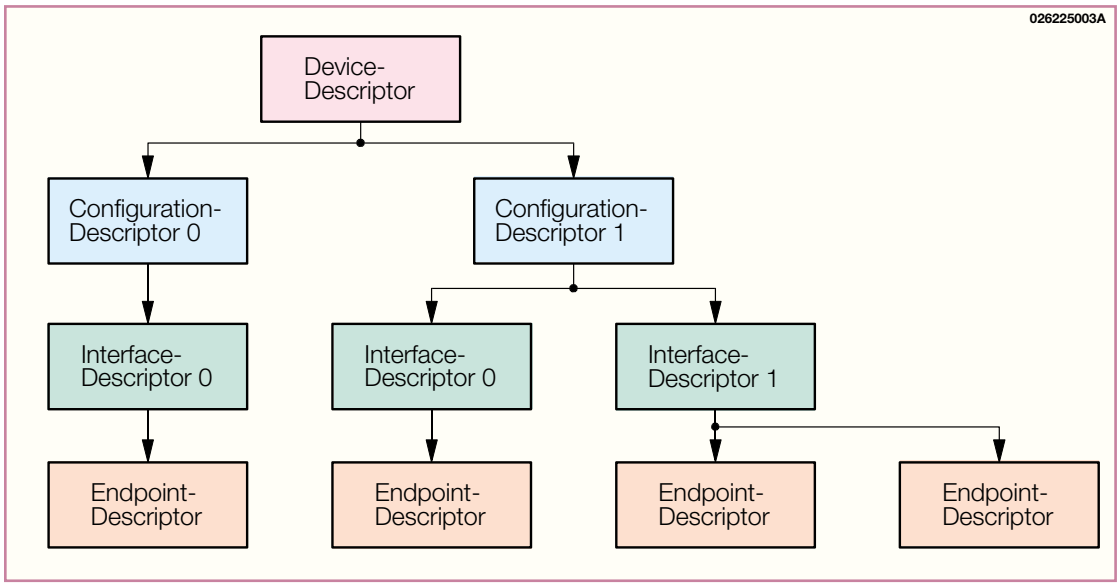
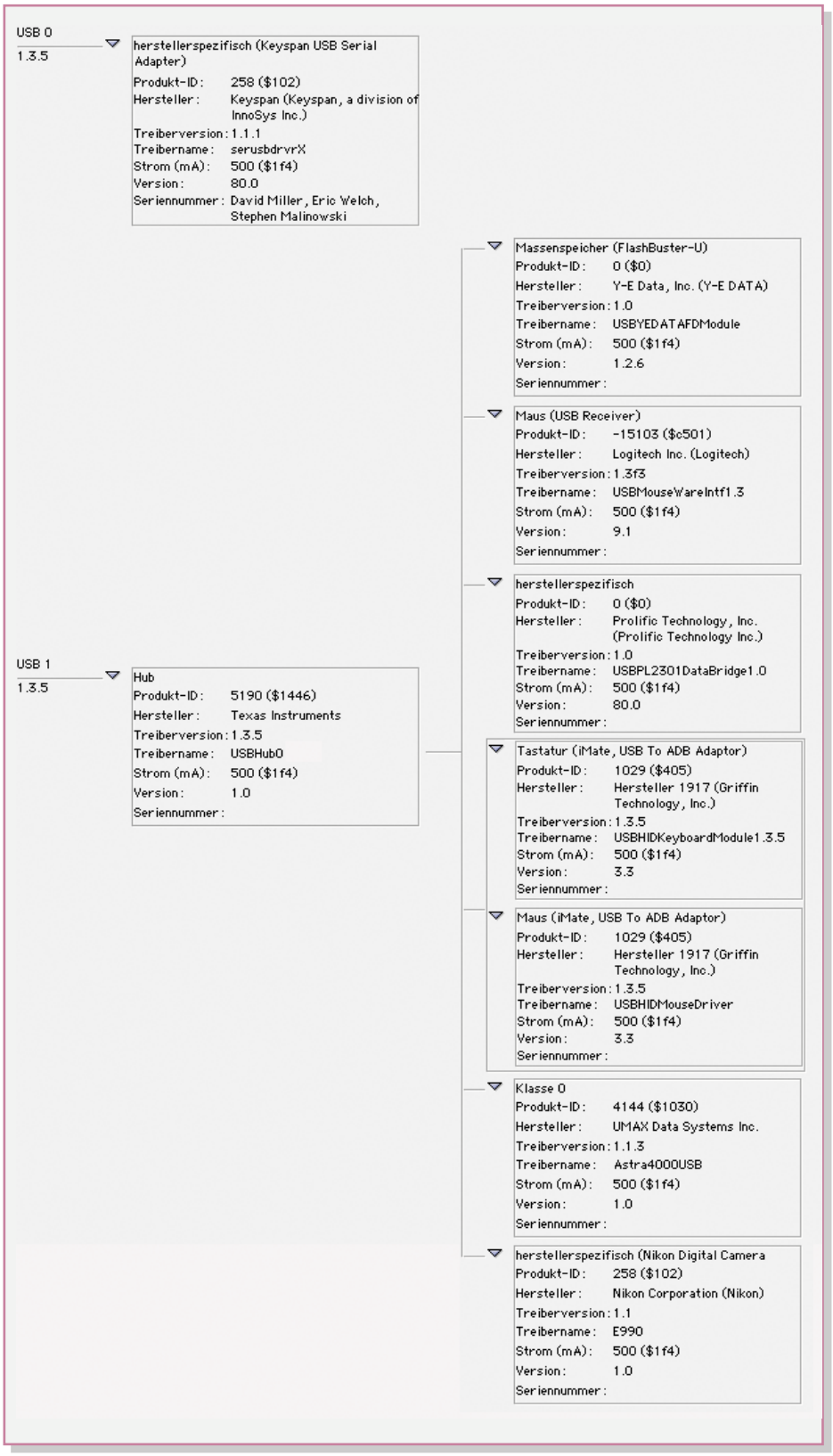


Bild 3: Hierarchie der Standard-Deskriptoren

Bild 4:
Über die Deskriptoren kann sich der Nutzer auch einen schnellen Überblick über das installierte System verschaffen.



entsprechenden Gerätes sind jedoch noch weitere Informationen notwendig. Jedes USB-Gerät stellt sogenannte „Deskriptoren" bereit, über die eine Beschreibung der Eigenschaften und Konfigurationsmöglichkeiten erfolgt. Diese Deskriptoren werden PC-seitig abgefragt. Es gibt eine ganze Hierarchie von Deskriptoren, angefangen mit dem „Device-Descriptor", welcher die allgemeinen Eigenschaften (Hersteller, Typ, Geräteklasse usw.) beschreibt und von dem es nur einen pro Gerät gibt.

Die nächste Stufe ist der „Configuration-Descriptor". Ein USB-Gerät kann mehrere Konfigurationen aufweisen und somit auch mehrere „Configuration-Descriptors" die z. B. einen Modus beschreiben, in dem das Gerät im Normalbetrieb mit erhöhtem Strombedarf arbeitet und einen anderen für einen geringeren Strombedarf, der beispielsweise zur Anwendung kommt, wenn das Gerät gerade nicht verwendet wird.

Jeder „Interface-Descriptor" beschreibt eine logische Schnittstelle innerhalb der Konfiguration, die u. U. auch aus mehreren Endpunkten bestehen kann. Eine Konfiguration kann mehrere logische Schnittstellen enthalten.

Die Endpunkte innerhalb einer Schnittstelle haben jeweils einen eigenen „Endpoint-Descriptor" der den Bandbreitenbedarf des entsprechenden Endpunktes enthält.

Die Hierarchie dieser „Standard-Deskriptoren" ist in Abbildung 3 als Diagramm dargestellt.

Des Weiteren gibt es noch „String-Descriptors", die optional von einem USB-Gerät bereitgestellt werden. Diese Deskriptoren können Informationen im Klartext liefern, wie z. B. den Herstellernamen oder die Gerätebezeichnung.

Die Informationen aller dieser Deskriptoren sind für einen optimalen und stabilen Betrieb eines USB-Gerätes notwendig und werden bei der Konfiguration abgefragt. Eine Reihe dieser Informationen sind für den Computernutzer direkt im Klartext verfügbar, etwa über den Aufruf eines System-Profilers (Abbildung 4). Hier kann man bei einigen Betriebssystemen einen direkten Überblick über das installierte USB-System erhalten.

USB 2.0

Die USB-Spezifikation 2.0 weist viele Neuerungen auf und ist in Bezug auf vorherige Ausgaben dieser Spezifikation sehr viel komplexer geworden, jedoch ist die Abwärtskompatibilität weiterhin gegeben.

USB On-The-Go

Laut Spezifikation können USB-Geräte nicht direkt untereinander kommunizieren, sondern sie müssen immer den Umweg über den Rechner gehen. „USB On-The-Go" ermöglicht die direkte Verbindung zweier USB Geräte. Ein gutes Beispiel ist hier die Anbindung eines Druckers an einen PDA.

Damit schließen wir unsere Beschreibung des USB ab, die dem interessierten Leser einen Überblick über das Thema liefern soll. Eine vollständige Dokumentation des „Universal Serial Bus" füllt ganze Bücher und ist nicht in einem Artikel zu beschreiben, allein die aktuelle Spezifikation (USB 2.0) hat einen Umfang von 650 Seiten. ELV