Anbindung von Speichermedien an Rechner

Benjamin Rommel

1. Januar 2009

Inhaltsverzeichnis

1 Ältere Bussysteme

1.1 IDE

1.1.1 Geschichte

Seinen Ursprung hat die IDE-Schnittstelle 1984 beim amerikanischen Computerhersteller Compaq. Zum Betrieb der Festplatten (Größe ≈10MB) zur damaligen Zeit war eine ST506-Controllerkarte[1] notwendig. Dies brachte jedoch Probleme bei mobilen Computern mit sich, da die Controllerkarte Platz und einen der raren Steckplätze benötigte. Compaq beauftragte daher den Festplattenhersteller Western Digital eine Festplatte mit eingebautem ST506-Controller zu entwickeln, sodass keine separate Controllerkarte mehr nötig war. Verbunden wurde die Festplatte durch ein 40-poliges Flachbandkabel mit dem Mainboard. 1986 tauchte erstmals der Begriff IDE (Integrated Disc Electronics) auf, als Compaq Western Digital beauftragte den Controllerchip auf der Laufwerksplatine unterzubringen. Den ersten Einsatz feierte das System im Compaq Deskpro 386.

Durch die Integration des Controllerchips auf der Laufwerksplatine boten sich große Vorteile gegenüber der Verwendung der Controllerkarten. Durch das Wegfallen der Controllerkarte wurde nicht nur Platz, sondern auch Kosten gespart. Außerdem konnte die Elektronik nun leicht an die einzelnen Eigenschaften der Festplatten angepasst werden und eine universell funktionierende Controllerkarte war nun nicht notwendig und eine solche Controllerkarte wäre im Zuge des rasanten Wachstums des Festplattenmarktes zur damaligen Zeit auch kaum realisierbar gewesen. Da die einzelnen Hersteller jedoch in ihren Spezifikationen immer weiter vom eigentlichen ST506-Standard entfernten gab es bald Probleme in der Kommunikation zwischen dem Laufwerk und dem Mainboard. Als Resultat setzten sich 1988 mehrere Hersteller von Computern und Festplatten zusammen und bildeten die Interessengruppe CAM, mit dem Ziel, eine Normierung der IDE-Schnittstelle zu erreichen. Im März 1989 wurde schließlich der IDE-Standard mit der Bezeichnung ATA verabschiedet. (Quelle: [VILSBECK 99])

1.1.2 Enhanced IDE

Auch wenn Enhanced IDE, kurz EIDE, nie als offizieller Standard definiert wurde, wurde es schnell nach Markteinführung zum inoffiziellen Standard. EIDE war eine Erweiterung von IDE, die auch zahlreiche Verbesserungen beinhaltete. Ursprünglich von Western Digital eingeführt, haben es die anderen Festplattenhersteller bald ebenfalls für ihre Festplatten übernommen. Die vier grundlegenden Neuerungen von EIDE waren:

Höhere Datentransferraten Ursprünglich wurde die ATA-Spezifikation für den ISA-Bus entwickelt, doch dieser hatte lediglich eine maximale Bandbreite

[1] 'Schnittstellen-Standard vor den ATA-Spezifikationen. ST506 definiert das elektrische und mechanische Interface zwischen Festplatte und Controllerkarte.', Quelle: [VILSBECK 99]

von 8.33 MB/s und wurde daher den immer schneller werdenen Festplatten nichtmehr gerecht. Durch die Übertragung der ATA-Spezifikation auf den PCI-Bus und durch Verwendung von schnelleren Timings mit PIO-Mode 4[2] wurden Transferraten von bis zu 16,6 MB/s erreicht.

Höhere Speicherkapazitäten Der ursprüngliche ATA-Standard beinhaltete ein Adressierungsverfahren, mit dem bis zu 128 GB Daten verwaltet werden konntn. Doch das ursprüngliche PC BIOS von IBM konnte aufgrund seiner Architektur maximal 1024 Zylinder[3] adressiert werden und es wurde eine maximale Festplattenkapazität von 500 MB unterstützt. Zu der Zeit, als das BIOS von IBM entwickelt wurde, erschienen 500 MB eine hinreichend große Grenze für die Festplattenkapazität, doch sie wurde schnell überschritten. Damit diese Grenze überschritten werden konnte, musste das BIOS und die Steuerelektronik der Festplatte überarbeitet werden.

Zwei Kanäle Die ATA-Spezifikation sah einen Kanal vor, an den man zwei Geräte anschließen konnte. EIDE erweiterte nun die Spezifikation um einen zweiten Kanal, der ein exaktes Abbild des ersten Kanals ist, besitzt also die gleichen Kapazitäten und Funktionalitäten. Damit zwei Kanäle verwendet werden konnten, musste dies vom Host-BIOS unterstützt werden.

Unterstützung weiterer Geräte ATA wurde für Festplatten konzipiert. Mit dem ATA-4-Standard war auch eine Anbindung anderer Geräte wie CD-Rom-Laufwerke oder Streamern möglich. Diese Funktionen wurden auch von EIDE übernommen. (Quelle: [VILSBECK 99])

1.1.3 Adressierung

Das auf einer parallelen Busstruktur basierende ATA-Interface wird über verschiedene Register[4] angesprochen. Die Register enthalten neben der Länge der zu übertragenden Datenblöcke auch die Adressinformationen. Die Adressierung kann hierbei durch zwei mögliche Verfahren geschehen: dem CHS[5]-Modus und der LBA[6]-Adressierung.

[2] 'Abkürzung für programmierten Input/Output. Daten zwischen Laufwerk und Hauptspeicher werden mit Hilfe von IN/OUT-Befehlen über die CPU ausgetauscht. Der schnellste PIO-Mode 4 erreicht 16,6 MByte/s.', Quelle: [GLOSSAR]

[3] 'Bezeichnung für direkt übereinander liegende Spuren auf den Magnetschichten einer Festplatte.', Quelle: [GLOSSAR]

[4] 'Speicherbereich mit fest zugewiesener Adresse, dem eine bestimmte Funktion zugeordnet ist.', Quelle: [GLOSSAR]

[5] 'Cylinder Head Sector: Adressierungsverfahren zum Ansteuern einer Festplatte. Basiert auf den physikalischen Vorgaben des Laufwerks.', Quelle: [GLOSSAR]

[6] 'Large Block Addressing: Adressierungsverfahren zum Ansteuern einer Festplatte. Die gesamte Kapazität der Laufwerks wird als kontinuierliche Folge von Datenblöcken angesehen.', Quelle: [GLOSSAR]

CHS-Modus Der CHS-Modus basiert auf der physikalischen Adressierung mittels Zylinder, Köpfen und Sektoren. Nach dem initialen ATA-Standard stehen dabei 16 Köpfe und 655.535 Spuren zur Verfügung, wobei jede Spur 256 Sektoren mit je einer Speichertiefe von 512 Byte verwaltet. Dies führte zur theoretischen Maximalkapazität der Festplatten von etwa 127 GB nach dem ursprünglichen ATA-Standard. Die Adressierung mittels Register funktionierte wie folgt: Das Sektornummernregister gibt den ersten zu übertragenden Sektor an und das Sektorzahlregister die Anzahl der zu übertragenden Register. Zur Adressierung einer Spur wurden zwei Zylindernummerregister verwendet und das Laufwerksregister übermittelte die Gerätenummer, den Kopf und das Adressierungsverfahren (CHS oder LBA). Jedes Register hat dabei eine Größe von einem Byte.

LBA-Adressierung Im Gegensatz zum CHS-Modus wird bei der LBA-Adressierung die Festplatte nicht über physikalische Register, sondern über logische Blöcke adressiert. Man geht davon aus, dass die Festplatte aus einer Folge von identisch großen Datenblöcken besteht. Der Host benötigt hierfür keinerlei Kenntnisse über die Geometrie der Festplatte. Jeder Sektor der LBA-Adressierung besitzt eine logische Nummer, sodass jeder Sektor einzeln angesprochen werden kann. Die Übersetzung der logischen Nummern in die physikalische Adressierung übernimmt dabei die Laufwerkselektronik.

Darüber hinaus gab es einige weitere Register, welche aber nicht von allen Geräten genutzt wurden und teils erst in späteren ATA-Standards implementiert wurden. Dazu gehörten unter anderem das Feature-Register für Zusatzfunktionen der Schnittstelle oder das Kommandoregister zur Übermittlung eines von 50 vordefinierten Kommandos. (Quelle: [VILSBECK 99])

1.1.4 Master / Slave

In der ATA-Spezifikation waren für den Kanal bis zu zwei Geräte vorgesehen, weswegen jedes IDE-Kabel stets über zwei IDE-Anschlüsse verfügte: eins in der Mitte und eins am Ende. Auch wenn sich beide Anschlüsse weder in ihren Eigenschaften, noch in der Signallogik unterschieden, wurde stets empfohlen ein einzelnes Gerät mit dem Anschluss am Kabelende zu verbinden. So wurde die Signalqualität bei der Kommunikation merklich verbessert. Ist das einzelne Laufwerk mit dem mittleren Anschluss verbunden worden, so fanden am Kabelende Reflexionen statt und die einzelnen Signaladern wirkten zusätzlich wie Antennen und empfingen Störungen von außen. Die Bezeichnung der Geräte mit Master und Slave war stets irreführend, da beide Geräte die gleichen Rechte auf dem Bus hatten und keines der Geräte vom anderen abhängig war oder von ihm gesteuert wurde. In der Spezifikation wurden die beiden Geräte allerdings als Device 0 und Device 1 bezeichnet und entsprachen dem Wert des Gerätebits im Laufwerksregister. (Quelle: [VILSBECK 99])

1.2 SCSI

1.2.1 Geschichte

Im August 1981 wurde nach zweijähriger Entwicklungszeit wurde der von der Firma Shugart entwickelte SASI-Standard (Shugart Associates System Interface) erstmals in einem Zeitungsartikel vorgestellt. Er entspricht bis auf einen kleinen Befehlssatz dem späteren SCSI-1-Standard. Der SASI-Standard war ein herstellerspezifischer Bus und erst durch die Arbeiten von NCR wurde die Busschnittstelle verallgemeinert und bekam schließelich durch die Normung von ANSI den Namen SCSI. Der erste SCSI-Standard, SCSI-1, erschien 1986.

Mit den späteren Versionen des Standards wurden stetig Neuerungen und Verbesserungen hinzugefügt. Dazu zählt unter anderem die Arbeit an Wide SCSI. Hierbei wurde über eine Erweiterung der Bitbreite der SCSI-Schnittstelle von 8 auf 16 oder 32 Bits diskutiert. Wirklich einigen konnte man sich nie auf eine Größe, aber es hat sich über die Jahre die Verwendung einer Busbreite von 16 Bits aus dem SCSI-3-Standard durchgesetzt. Eine weitere Entwicklung war Fast SCSI, bei der hauptsächlich Kabelimpendanz- und Kabelmaterialprobleme angegangen wurden. Erstmals in SCSI-2 definiert ermöglicht Fast SCSI eine Bandbreite von 10 Megatransfers pro Sekunde, SCSI-1 war für die Hälfte ausgelegt. Durch Kombination von Fast und Wide SCSI erreicht man eine Bandbreite von 320 MB/s bei einer 16-bittigen Parallelübertragung. Es sind jedoch auch weit höhere Bandbreiten durch Verwendung von SCSI-3 mit Fibre Channels möglich.

Ein Großteil der SCSI-Geräte-produzierenden Hersteller hat sich zur SCSI Trade Association zusammengeschlossen. (Quelle: [STRASS])

1.2.2 Der Aufbau

SCSI steht für Small Computer System Interface, doch dies ist eine im Grunde falsche Bezeichnung. Zum einen wurde SCSI fast auschließlich für größere Systeme verwendet und wird daher dem 'Small' nicht gerecht und zum anderen handelt es sich bei SCSI um einen Datenbus und kein klassisches Interface. Zur Kommunikation wird ein SCSI Controller, oft auch als Host Adapter bezeichnet, benötigt. Üblicherweise befindet er sich auf einer SCSI-Karte oder bei manchen älteren Systemen gab es einen solchen Controller auch auf dem Mainboard. Das SCSI BIOS ist ebenfalls in dem Controller untergebracht, der üblicherweise auf einem ROM oder einem Flashspeicher untergebracht war. Jedes Gerät, welches auf dem Bus liegt hat eine eigene Identifikationsnummer (ID), anhand der das Gerät bei der Kommunikation angesprochen werden kann. Die Anzahl der verfügbaren IDs entspricht der Anzahl an Datenleitungen des SCSI-Bus, d.h. 8 Geräte bei Narrow SCSI (d.h. SCSI-1) und 16 Geräten bei Wide-SCSI. Die ID gibt ebenfalls die Priorität des Geräts an. Da sich die Geräte einen Bus teilen, wird durch die Priorisierung verhindert, dass niederpriore Geräte höherpriore blockieren. Üblicherweise hat der SCSI-Controller dabei die höchste ID.

Für interne SCSI-Geräte wird ein Flachbandkabel verwendet und für externe Geräte ein dickes Rundkabel, wobei die externen SCSI-Geräte in Reihe geschal-

ten werden. Das heißt, dass jedes externes SCSI-Gerät zwei SCSI-Anschlüsse hat, einer wird mit dem vorangehenden SCSI-Gerät verbunden und der zweite mit dem nächsten SCSI-Gerät, falls vorhanden.

Ein wichtiger Aspekt bei der Vernetzung mit SCSI war die korrekte Terminierung. Wurde ein SCSI-Bus offen gelassen und nicht terminiert, so konnten Signale am Busende reflektiert werden und die Kommunikation auf dem Bus stören. Die Terminierung wurde durch Widerstände erreicht, die entweder direkt auf den Controllern verbaut worden sind, oder als Adapter auf ein Kabelende gesteckt werden konnten. (Quelle: [TYSON, WILSON])

1.2.3 Signalgebung

Auf dem SCSI-Bus gibt es drei verschiedene Arten der Signalgebung:

Single-Ended Der Controller erzeugt das Signal und sendet es auf einer einzelnen Datenleitung an alle Geräte, wobei jedes Gerät als Masse fungiert. Dadurch nimmt das Signal rasch an Stärke ab, weshalb Single-Ended SCSI nur Buslängen bis zu drei Metern erlaubt. Die Single-Ended-Signalgebung wurde üblicherweise in PCs verwendet.

High-Voltage Differential Bei der High-Voltage-Differential-Signalgebung (kurz HVD) wurde das Signal auf zwei Datenleitungen synchron übertragen, wobei das Signal auf einer Leitung auf low und auf einer als high übertragen wurde, analog zu USB-Leitungen. Zudem besitzt hier jedes angeschlossenes SCSI-Gerät einen Signaltransceiver. Sendet der Controller ein Signal, so wird es vom ersten Gerät auf dem Bus empfangen und an das nächste Gerät weitergeleitet, bis das Signal schließlich sein Zielgerät erreicht. Durch die Nutzung zweier Datenleitungen und des Transceivers wurden Buslängen von bis zu 25m möglich. Eine klassische Anwendung fand die HVD-Signalgebung bei Servern.

Low-Voltage Differential Das Funktionsprinzip bei LVD entspricht dem von HVD, allerdings verwendet LVD kleinere Transciever, die zudem auf den SCSI-Controllern untergebracht sind. Dadurch wird weniger Strom zur Kommunikations benötigt, die maximale Reichweite beschränkt sich aber im Gegenzug auch auf 12m.

Auch wenn heute SCSI noch weiter vorangetrieben wird, so wurde es im Desktop-Bereich bereits seit längerer Zeit vollständig von S-ATA verdrängt und auch bei externen Massenspeicherobjekten wie etwa SAN oder SAS wird es immer mehr durch S-ATA verdrängt. Es findet jedoch auch immernoch in einigen großen Projekten Verwendung, da die inzwischen 20jährige Geschichte dieses Busses ihn sehr stabil gemacht hat. (Quelle: [TYSON, WILSON])

2 Aktuelle Bussysteme

2.1 Serial ATA

2.1.1 Einführung

Serial ATA ist der Nachfolger des als IDE oder ATA bekannten Bussystems. Zur genaueren Differenzierung spricht man beim alten ATA-Bus heute von Parallel Bus, um es direkt vom neuen Serial Bus abzugrenzen. Als rechtmäßiger Nachfolger von Parallel ATA ist Serial ATA hauptsächlich als Speicherbus für Festplatten entworfen worden, wird heute jedoch auch für Peripheriegeräte wie DVD-Laufwerke verwendet. Aufgrund der Konzeption als Festplatten-Bus fällt es nicht negativ aus, dass die maximale Buslänge von Serial ATA nur einen Meter beträgt, was, im Vergleich zu FireWire beispielsweise, relativ kurz ist. (Quelle: [ALLPINOUTS])

2.1.2 Versionen

Die erste Version von Serial ATA trug den offiziellen Namen 'SATA 1.5 Gb/s' oder alternativ 'SATA/150', um die traditionelle Namensgebung der alten Parallel ATA weiterzuführen (z.B. PATA/100, PATA/150). Dieser Standard lief mit einer Frequenz von 1.5 Gigahertz und verwendete die 8B/10B-Kodierung auf der physikalischen Schicht, welche eine Effizienz von 80% besitzt. Damit kommt Serial ATA auf eine effektive Datentransferrate von 1.2 Gb/s, bzw. 150MB/s.

Die nächste Version von Serial ATA erschien nur kurz später, nachdem einige Erweiterungen und Verbesserungen am ersten Serial ATA Standard vorgenommen wurden. Die größte Verbesserung war die Verdopplung der Signalrate auf 3Gb/s, wodurch sich, wieder aufgrund der 8B/10B-Kodierung, die effektive Datentransferrate im Vergleich zum vorherigen Standard mit 2.4Gb/s effektiv verdoppelt hat. Es wurde ebenfalls festgelegt, dass die zweite Version von Serial ATA vollständig abwärtskompatibel sein musste. Das Komitee, welches für den Serial ATA Standard zuständig war, gab dem neuen Standard die beiden offiziellen Namen 'SATA 3.0 Gb/s' und 'SATA/300', analog zur ersten Version. Allerdings setzte sich die Bezeichnung Serial ATA II durch. Das Komitee des Serial ATA Standards trug jedoch ebenfalls diesen Namen und versuchte die beiden offiziellen Namen durchzusetzen. Als dies nicht gelang, benannte sich daraufhin das Komitee um in 'SATA-IO' (Serial ATA International Organization). (Quelle: [ALLPINOUTS])

Die dritte Version von Serial ATA, welche die beiden offiziellen Namen 'SATA Revision 3.0' und 'SATA 6.0 Gb/s' trägt, bietet neben einer erneuten Verdopplung der Datentransferrate auch verbesserte Eigenschaften beim Streaming von Medien und beim Power Management. Auch wenn die maximale Datentransferrate der dritten Serial ATA Spezifikation für derzeitige Festplatten unerreichbar bleibt, so bietet sie den immer schneller wachsenden SSDs noch Platz nach oben. (Quelle: [COMPUTERBASE])

2004 wurde auch der eSATA verabschiedet, welcher den Anschluss von externen SATA-Geräten an den PC vorsieht. Mit kleineren Einschränken der physika-

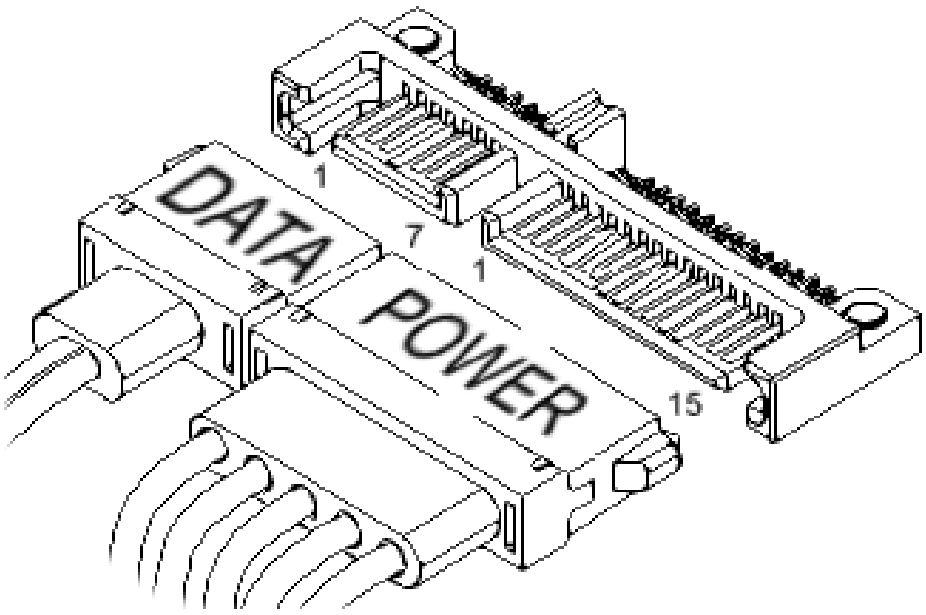

Abbildung 1: Daten- & Stromstecker, Quelle: [ALLPINOUTS]

lischen Möglichkeiten (so wurde beispielsweise die maximale Spannung gesenkt) konnte so die maximale Leitungslänge auf bis zu 8m erhöht werden. Allerdings verfügen viele Mainboards nicht standardmäßig über eSATA-Anschlüsse. (Quelle: [ALLPINOUTS])

2.1.3 Kabel

Für Serial ATA werden zwei getrennte Anschlüsse verwendet: einen siebenpoligen Datenstecker und einen 15poligen Stromstecker. Der Aufbau der Stecker ist wie folgt:

Die Kabel sind sicher der optisch markanteste Unterschied zwischen SATA und PATA. Die Stecker wurden so entwickelt, dass sie möglichst geringen Einfluss auf die Luftzirkulation im Computergehäuse haben. Außerdem ging man von dem Master/Slave-Prinzip von PATA weg und spendierte jedem Gerät einen eigenen Anschluss. (Quelle: [ALLPINOUTS])

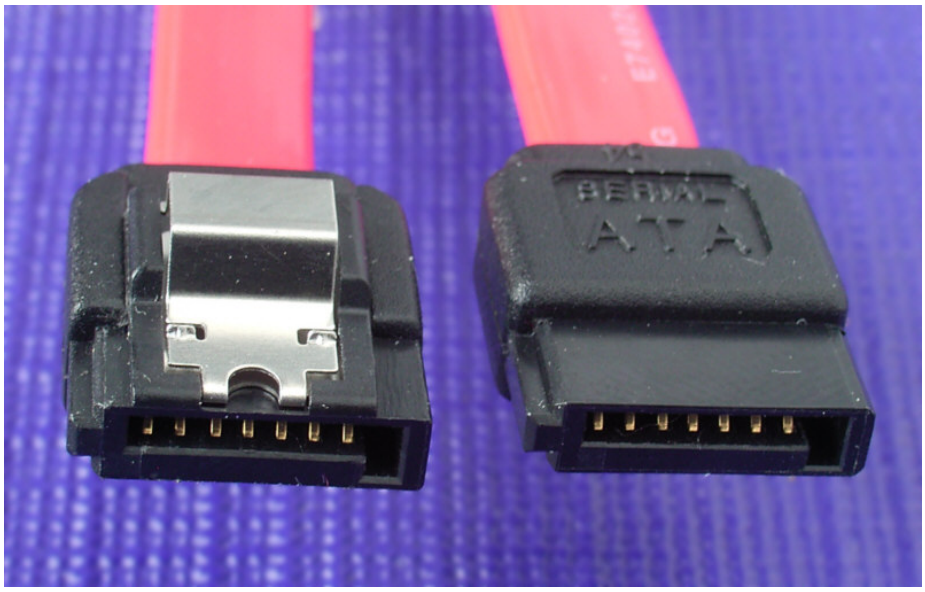

Abbildung 2: Datenstecker, Quelle: Wikipedia

Abbildung 3: Stromstecker, Quelle: Wikipedia

Pin Nr.	Signl Name	Signal Description
1	GND	Masse
2	A+	Senden +
3	A-	Senden -
4	GND	Masse
5	B+	Empfangen +
6	B-	Empfangen -
7	GND	Masse

Pin Nr.	Signl Name	Signal Description
1	V33	3,3 V
2	V33	3,3 V
3	V33	3,3 V
4	GND	Masse
5	GND	Masse
6	GND	Masse
7	V5	5 V
8	V5	5 V
9	V5	5 V
10	GND	Masse
11	Reserved	-
12	GND	Masse
13	V12	12 V
14	V12	12 V
15	V12	12 V

2.2 USB

2.2.1 Grundlegener Aufbau, Topologie

Der Universal Serial Bus, kurz USB, ist ein serielles Bussystem, welches der RS232-Schnittstelle, vielen als COM-Anschluss am PC bekannt, sehr ähnlich ist. Die Topologie des Busses ist dabei sternförmig. Im Zentrum steht der USB Controller, bei Computern meist auf dem Mainboard oder bei älteren Systemen auf Steckkarten untergebracht. An diesen Controller sind die einzelnen USB-Geräte angeschlossen und die komplette Kommunikation läuft über den Controller und nur der Controller ist berechtigt, Kommunikation herzustellen. Eine Kommunikation kann nie von einem USB-Gerät aus gestartet werden.

USB-Geräte besitzen alle die Plug&Play-Fähigkeit, das heißt, sie können während der Laufzeit an den Computer angeschlossen und verwendet werden. Über den Pull-Up-Widerstand der Datenleitung erkennt der Controller, dass ein neues Gerät angeschlossen wurde und vergibt temporär die Adresse 0 an das Gerät. Bei der ersten Kommunikation mit dem Gerät wird die Konfiguration des Geräts ausgelesen und der Controller erhält dadurch Kenntnis von der Art und den Funktionalitäten des Geräts und weißt ihm eine freie Adressnum-

mer zwischen 1 und 127 zu. Es können daher maximal 127 verschiedene USB-Geräte an einen Controller angeschlossen werden. Heutige Mainboards verfügen in der Regel über 8 bis 16 Anschlüsse für USB-Geräte. Sollen weitere Geräte angeschlossen werden, so müssen USB-Hubs an einen freien USB-Anschluss angeschlossen werden, wodurch sich eine Baumstruktur auf dem Bus bildet. Ein USB-Anschluss, normalerweise ein Blatt in der USB-Topologie wird zu einem Knoten mit n Blättern, wobei n der Anzahl der Anschlüsse an dem Hub entspricht. Doch auch beim Anschließen von beliebig vielen Hubs können aufgrund der Adressierung nie mehr als 127 Geräte angeschlossen werden.

Wie eben erläutert wird beim Anschließen eines USB-Geräts an den Controller die Konfiguration des Geräts ausgelesen. Diese Informationen stehen im Endpunkt 0. Endpunkte bei USB-Geräten sind Pufferspeiche in die, je nach Einstellung, Daten geschrieben oder gelesen werden können. In den ersten Endpunkt, Endpunkt 0, kann sowohl gelesen als auch geschrieben werden. In den Konfigurationsdaten stehen unter anderem die Funktionalitäten des Geräts, Firmwaredaten und Eckdaten für das Gerät, wie etwa die Baudrate. Da der Pufferspeicher jedoch auch beschrieben werden kann, können z.B. manuell Timeout-Zeiten oder Baudraten geändert werden. Die Funktionalitäten des USB-Geräts werden durch Interfaces beschrieben. Jedes Interface wiederum besteht aus einem oder mehreren Endpunkten, also Pufferspeichern, jedoch dürfen es laut USB 2.0 Spezifikation nie mehr als 16 Endpunkte pro Gerät sein, die Zahl der Interfaces ist jedoch egal. Wie bereits kurz angesprochen kann ein USB-Gerät keine Kommunikation starten. Stattdessen fragt der USB-Controller in vordefinierten Zeitintervallen die Daten an den Geräten ab. Die Intervalle bei USB-Mäusen würden im Hundertstel-Sekunden-Bereich liegen. Im folgenden möchte ich zur Veranschauung ein Beispiel mit einer WebCam den grundlegenden Mechanismus veranschaulichen.

Beispiel Ich habe eine neue WebCam erworben und schließe diese an meinen Computer an. Aufgrund des Pull-Up-Widerstands erkennt der Controller, dass ein neues Gerät angeschlossen wurde und gibt ihm die Adresse 0. Anschließend wird die Kommunikation mit dem USB-Gerät gestartet, indem der Controller den Endpunkt 0, den Pufferspeicher mit den Konfigurationen, ausliest. Er erhält so Kenntniss über die technischen Daten wie Baudrate und Timeouts und erhält, als wichtigste Information, alle Informationen über die Interfaces. In den Konfigurationsdaten würde stehen, dass die WebCam über zweit Interfaces verfügt: eines für Bildübertragung und eines für Audio. Damit ist es mir möglich durch entsprechende Kommunikation nur Bild- oder nur Audio-Daten zu erhalten oder beides. Jedes Interface wiederum besteht aus zwei Endpunkten, wobei der Controller in den jeweils ersten Endpunkt schreiben und aus dem jeweils zweiten Endpunkt lesen kann. Möchte ich nun die WebCam für die visuelle Kommunikation nutzen, so sendet der Controller eine Anfrage an das Gerät, bzw. er schreibt sie in den ersten Endpunkt des Bild-Interfaces. Das Schreiben in diesen Endpunkt löst einen Interrupt im USB-Gerät aus und das Gerät bearbeitet die Anfrage. Als Konsquenz macht die WebCam eine Bildaufnahme und legt die

Daten auf den zweiten Endpunkt des Bild-Interfaces. Bei der nächsten Abfrage des Controllers am USB-Gerät werden dann die Daten am zweiten Endpunkt ausgelesen. Die Bilddaten werden also nicht direkt an den Computer übermittelt, sondern liegen im Puffer, bis sie vom Controller abgerufen werden. Dieses Beispiel zeigt ein USB-Gerät mit 2 Interfaces und 5 Endpunkten. Ein klassisches Beispiel für ein USB-Gerät mit nur einem Endpunkt wäre ein USB-Speicher oder ein Kartenlesegerät. (Quelle: [BREDENDIEK])

2.2.2 Endpunkte, Interfaces

Endpunkte Wie bereits bei den USB-Grundlagen besprochen sind Endpunkte Pufferspeicher in den USB-Geräte, über welche die Kommunikation mit dem Controller abgewickelt wird. Es gibt verschiedene Interfaces für die einzelnen Aufgaben des USB-Gerätes, doch auch die einzelnen Funktionalitäten des USB-Geräts haben unterschiedliche Ansprüche an die Pufferspeicher, was Datenrate und Echtzeitverhalten betrifft. Daher gibt es vier verschiedene Arten von Endpunkte, wobei jeder Endpunkt nur genau einen der folgenden Modi haben darf: Control, Interrupt, Isonchronous, Bulk. Der Control-Modus wird primär für Steuerungszwecke genutzt. Daher hat dieser Modus die höchste Priorität und den besten Fehlerschutz. Der Control-Modus wird beispielsweise beim Endpunkt 0 verwendet, wenn der Controller die Konfiguration aus dem USB-Gerät liest. Der Interrupt-Modus wird für die Übertragung kleiner Datenmengen in regelmäßigen Zeitabständen verwendet, also beispielsweise bei Maus und Tastatur. Für große Datenmengen, die in einer konstanten Datenrate übertragen werden müssen, aber Fehleranfälligkeit tolerieren, wird der Isonchronous-Modus verwendet. Ein klassisches Einsatzgebiet hier wären USB-Lautsprecher. Für große Datenaufkommen und sichere Übertragung wird der Bulk-Modus verwendet, der allerdings keine hohen Datenraten garantiert. USB-Sticks wären ein Paradebeispiel hierfür. (Quelle: [BREDENDIEK])

Interfaces Die Interfaces legen, wie oben erklärt, die Funktionalität eines USB-Gerätes fest. In den Konfigurationen eines USB-Geräts (im Endpunkt 0) steht unter anderem für jedes Interface ein Deskriptor, welcher beschreibt, wie sich das Interfaces verhält. Folgende Tabelle liefert eine Übersicht über die Deskriptoren. (Quelle: [USB CLASS CODES])

Base Class	Descriptor Usage	Description
00h	Device	Use class information in the Interface Descriptors
01h	Interface	Audio
02h	Both	Communications and CDC Control
03h	Interface	HID (Human Interface Device)
05h	Interface	Physical
06h	Interface	Image
07h	Interface	Printer
08h	Interface	Mass Storage
09h	Device	Hub
0Ah	Interface	CDC-Data
0Bh	Interface	Smart Card
0Dh	Interface	Content Security
0Eh	Interface	Video
0Fh	Interface	Personal Healthcare
DCh	Both	Diagnostic Device
E0h	Interface	Wireless Controller
EFh	Both	Miscellaneous
FEh	Interface	Application Specific
FFh	Both	Vendor Specific

Jede Basisklasse kann dabei mehrere Unterklassen enthalten, die wiederum verschiedene Protokolle unterstützen können. Um noch einmal das obige Beispiel mit der WebCam aufzugreifen. Beim Anschließen der WebCam findet der USB-Controller in der Konfiguration zwei Deskriptoren mit den Basisklassen 01 und 0E (beide hexadezimal) und weiß anhand dessen, dass das angeschlossene Gerät Audio- und Videostreaming unterstützt. Die genauen Funktionalitäten werden aus der Kombination von Basisklasse, Unterklasse und Protokoll ersichtlich.

2.2.3 Stromversorgung

Obwohl die USB-Schnittstelle viel mit der veralteten RS232 gemeinsam hat, hat USB einen eklatanten Vorteil gegenüber dem RS232: USB-Geräte könnten über den USB-Anschluss mit Strom versorgt werden. Der Controller legt dabei selbstständig fest, welches Devices wieviel Strom bekommt. Maximal stehen einem USB-Gerät 0,5A zur Verfügung, was bei der Versorgungsspannung von 5V noch immerhin 2,5W macht. Jedes Gerät muss allerdings auch meiner Stromstärke von 100mA bestrieben werden können. Dies ist die Stromstärke, mit der USB-Geräte nach dem Anschluss an den Bus versorgt werden. Erst nach Zuweisung einer Adresse und dem Auslesen der Konfiguration legt der USB-Controller fest, wieviel Strom das Gerät tatsächlich bekommt. Dem Controller steht es auch frei USB-Geräte von der Stromversorgung zu trennen, wenn er dies für erforderlich hält. Außerdem sieht die USB-Spezifikation vor, dass Geräte, die mehr als 0,5A benötigen, automatisch von der Stromversorgung getrennt werden, als Strafe quasi. (Quelle: [BREDENDIEK])

2.2.4 Kabel

Die Kabel des USB 1 und 2 Standards sind vieradrig und verdrillt, wobei die Kabel paarweise voneinander abgeschirmt sind. Es gibt zwei Adern für die Stromversorgung, sodass die via Plug&Play angeschlossenen Geräte sofort verwendet werden können und zwei Kabel sind für den Datentransfer. Hierbei werden die Daten auf beiden Adern parallel übertragen, wobei die Signale der zweiten Ader die negierten Signale der ersten Ader sind. Wird auf Ader 1 eine '1' übertragen, so wird auf der zweiten Ader eine '-1' übertragen. Der Empfänger bildet nun die Differenz beider Signale und enthält so als Werte 0 oder 2, wodurch die Übertragung gegen Störsignale geschützt sind.

2.2.5 Treiber

Wer ein USB-Gerät während der Laufzeit an den Computer anschließt stellt oft fest, dass sich das Gerät häufig sofort verwenden lässt. Der Grund hierfür ist der oben angesprochene Hardware-Aspekt und zum anderen die Struktur der Treiber. Jeder Hersteller, der USB-Geräte verkaufen möchte, muss sich bei USB.org für 2.000,- Euro registrieren lassen und erhält dafür eine 16bit-lange VID (vendor identifier), die unique ist. Folglich kann es weltweit nur maximal 65535 USB-Hersteller geben. Außerdem weißt jeder Hersteller jedem Gerättyp, den er herstellt eine eindeutige (unique) PID (product identifier) zu. Bei der Installation eines Treibers für USB-Geräte wird immer eine Konfigurationsdatei übertragen, in welcher jede VID-PID-Kombination steht, die dieser Treiber unterstützt. Schließen wir nun ein USB-Gerät an den Computer an, so sucht dieser in den Konfigurationsdateien nach der Unterstützung für dieses Gerät. Wurde ein enstprechender Eintrag gefunden, so kann das Gerät gleich ohne Einschränkung verwendet werden, andernfalls wird der Benutzer zur Installation eines passendes Treibers aufgefordert.

2.2.6 Ausblick

Wie in dem Vergleich zwischen USB und FireWire hier näher ausgeführt, ist USB derzeit in puncto Geschwindigkeit FireWire deutlich unterlegen. Immerhin ist der USB 2.0 Standard auch schon 8 Jahre alt. Nach einiger Verzögerung wurde jetzt der USB 3.0 verabschiedet und wird 2009 erscheinen. Die ersten passenden Geräte sind für Mitte 2009 angekündigt. Die größte Verbesserung von USB 3.0 im Vergleich zu seinem Vorgänger ist die Geschwindigkeit: während USB 2.0 eine Bandbreite von 480Mbit/s zur Verfügung stellte, beträgt die Bandbreite bei USB 3.0 4,8Gbit/s, was das zehnfache von USB 2.0 ist! Als Beinamen erhält USB 3.0 auch den passenden Beinamen SuperSpeed. Die Kabel-Anschlüsse und Buchsen werden überarbeitet, sollen aber abwärtskompatibel sind, zumindest zu USB 2.0. Die neuen USB 3.0-Stecker können nur mit Hilfe eines Adapters auf die USB 2.0-Buchsen gesteckt werden, aber umgekehrt wird es möglich sein USB 2.0-Geräte an USB 3.0-Buchsen ohne Adapter anzuschließen. Die Leitungen innerhalb des Kabels werden auch von vier Leitungen auf acht erhöht, wodurch sich wahrscheinlich auch die Kabel verteuern werden.

Ein eindrucksvolles Beispiel für Leistungsfähigkeit von USB 3.0 liefert die hier verwiesene Quelle: während bei USB 2.0 die Übertragung eines HD-Films noch 14 Minuten dauerte, wird es mit USB 3.0 nurnoch 70 Sekunden dauern! (Quelle: [HESSE])

Nebst neuer Stecker und Leitungen, wie in den Abbildungen unten zu sehen, hat sich auch außerhalb der physischen Ebene einiges verändert. So wird bei USB 3.0 auf das Polling verzichtet. Polling bezeichnet das regelmäßige Abfragen des USB Controllers an den USB Geräten, ob diese noch angeschlossen sind und ob Daten vorliegen. In USB 3.0 können sowohl Controller als auch Gerät eine Datenübertragung anstoßen. Größter Nutznieser hiervon sind Mobilgeräte, deren Bandbreite nun nichtmehr durch die regelmäßig Polling-Abfragen belastet wird. Anstatt des Pollings kann sich künftig ein Gerät nun vom Controller von der Datenübertragung abmelden und später wieder anmelden. Erst, wenn der Controller eine Anfrage an ein Gerät schickt und keine Antwort erhält, wird es als entfernt angesehen. Neben den Kabeln werden sich auch die Hubs verteuern, da die neuen USB 3.0 Hubs auch komplette USB 2.0 Hubs beinhalten werden. Dies dient dazu, dass man gleichzeitig USB 3.0 und USB 2.0 Geräte nutzen kann. Desweiteren wurde die maximale Versorungsstromstärke von 0.5 auf 0.9 Ampere erhöht, wodurch sich externe Festplatten künftig leichter mit Strom versorgen lassen. (Quelle: [GOLEM])

2.3 Firewire

2.3.1 USB vs. FireWire

Firewire ist USB sehr ähnlich, jedoch unterscheiden sich die beiden Systeme in einigen Punkten. Das Hauptentwicklungsziel von FireWire war es, möglichst viele Geräte an einen Bus anschließen zu können und ihnen dabei hohe Übertragungsraten zur Verfügung zu stellen. Die primäre Ausrichtung dieses Bussystems war daher auch der Audio- und Videobereich. FireWire wird von allen gängigen Betriebssystemen unterstützt und ermöglicht es, bis zu 63 Geräte an einen Bus zu legen. Dabei sind Übertragungsraten von bis zu 800 Mb/s möglich, wohingegen USB 2.0 lediglich bis zu 480 Mb/s unterstützt. Außerdem kann die Buslänge bis zu 100m betragen. Neben der Bandbreite unterscheiden sich FireWire und USB auch im Kommunikationsprinzip. Bei USB wird die Kommunikation von einem Controller gesteuert, welcher üblicherweise auf dem Mainboard sitzt. FireWire hingegen ist für Peer-to-Peer-Verbindungen ausgelegt, d.h. es können mehrere Audio- und Videogeräte miteinander kommunizieren, ohne, dass ein Computer notwendig ist. Um ein Netzwerk aufzubauen verwendet USB Hubs, während FireWire eine Reihenschaltung benutzt, wie auch SCSI. (Quelle: [TYSON, JAYTON])

Literatur

[ALLPINOUTS] http://www.allpinouts.org/index.php/Serial_ATA_(SATA,_Serial_Advanced_Technology_Attachment), zuletzt gesichtet am 22.12.2008, 14:19 Uhr MEZ

[BREDENDIEK] BREDENDIEK, J.: USB-Interface. Auf: http://www.sprut.de/electronic/interfaces/usb/usb.htm, zuletzt gesichtet am 04.12.2008, 21:19 Uhr MEZ

[COMPUTERBASE] http://www.computerbase.de/news/hardware/laufwerke/2008/august/doppelte_bandbreite_sata-spezifikation/, zuletzt gesichtet am 22.12.2008, 14:28 Uhr MEZ

[GOLEM] So funktioniert USB 3.0 - nicht alles bleibt kompatibel. Auf: http://www.golem.de/0811/63648-4.html, zuletzt gesichtet am 01.01.2009, 19:39 Uhr MEZ

[GLOSSAR] http://www.tecchannel.de/_misc/article/glossar/, zuletzt gesichtet am 30.11.2008, 16:02 Uhr MEZ

[HESSE] HESSE, S. (2208): USB 3.0 kommt 2009 mit 4,8 Gbit/s angerast. Auf: http://www.allround-pc.com/news/hardware/2008/november/usb-30-kommt-2009-mit-48-gbits-angerast, zuletzt gesichtet am 06.12.2008, 15:23 Uhr MEZ

[STRASS] STRASS, H. (2003): SCSI-Grundlagen. Auf: http://www.tecchannel.de/index.cfm?pid=209&pk=401371, zuletzt gesichtet am 30.11.2008, 18:02 MEZ

[TYSON, JAYTON] TYSON, J., LAYTON, J.: How FireWire Works. Auf: http://computer.howstuffworks.com/firewire.htm, zuletzt gesichtet am 30.11.2008, 20:46 Uhr MEZ

[TYSON, WILSON] TYSON, J., WILSON, T.V.: How SCSI Works. Auf: http://computer.howstuffworks.com/scsi.htm, zuletzt gesichtet am 30.11.2008, 19:05 Uhr MEZ

[USB CLASS CODES] USB Implementers Forum, Inc.: USB Class Codes. Auf: http://www.usb.org/developers/defined_class/, zuletzt gesichtet am 06.12.2008, 16:07 Uhr MEZ

[VILSBECK 99] VILSBECK, C. (1999): IDE-Grundlagen. Auf: http://www.tecchannel.de/storage/grundlagen/401440/index.html, zuletzt gesichtet am 30.11.2008, 16:02 Uhr MEZ