6.8 IEEE1394

6.8.1 Einleitung

Erste Entwicklungen wurden bereits 1988 bei Apple gestartet. Damals war das Ziel SCSI durch einen seriellen Hochgeschwindigkeitsbus zu ersetzen. Das IEEE stellte 1995 den IEEE 1394-1995 [5] Standard vor. Dieser Standard basierte wesentlich auf den Entwicklungen von Apple. Heute wird der Standard auch unter dem Namen i.LINK (Sony) bzw. Firewire (Apple) genutzt.

6.8.2 Architektur

IEEE 1394 unterhält immer Punkt-zu-Punkt Verbindungen zwischen den Knoten (Nodes). Alle Knoten sind bei IEE1394 gleichberechtigt und können auch untereinander kommunizieren. Das Besondere an IEEE1394 ist, das Knoten auch mehrere Ports besitzen können. Dies führt dazu, dass Knoten auch als Repeater arbeiten können, d.h. ein Port empfängt ein Paket, synchronisiert es und sendet es über die anderen Ports des Knotens wieder auf den Bus.

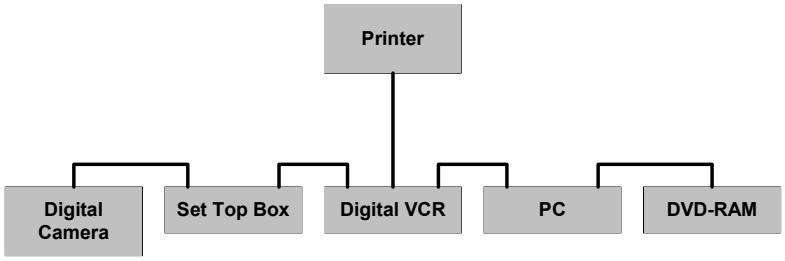

Abb. 6-32 : IEEE1394 Architektur nach [2]

Zur Adressierung stehen 64-bit Adressen zur Verfügung, aufgeteilt in 10-bit für die Netzwerk (BUS) ID, 6-bit für die Knoten ID und 48-bit für die Speicheradresse. Als Resultat ergibt sich die Adressierbarkeit von 1023 Netzwerken mit je 63 Knoten, wobei in jedem Knoten 256Terra Byte Speicher adressiert werden können.

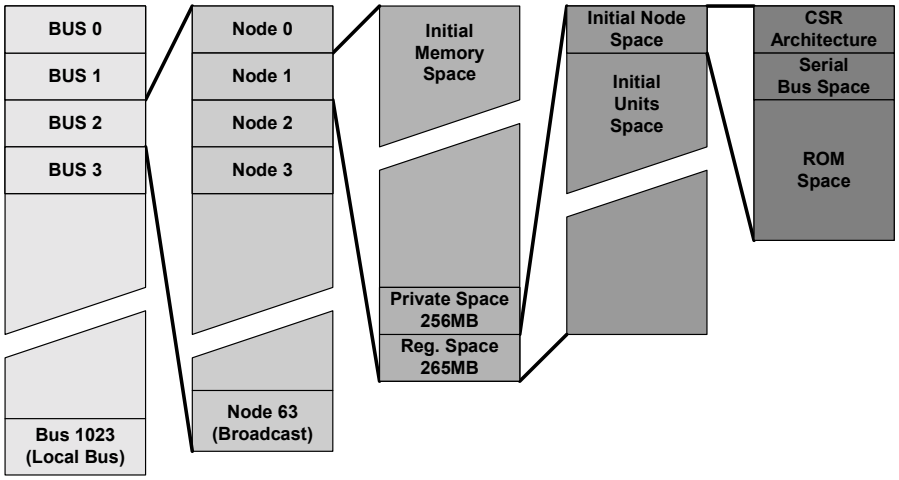

Abb. 6-33: IEEE 1394 Bus Adressraum nach [2]

6.8.3 Transfers und Transaktionen

Das 1394 Protokoll unterstützt asynchrone und isynchrone Datentransfers.
Der asynchrone Datentransfer umfasst die Übertragung von Nutzdaten in Paketen variabler Länge. Ein asynchroner Transfer zielt auf einen bestimmten Knoten auf dem Bus mit Hilfe einer eindeutigen 64-bit langen Knotenadresse. Asynchrone Transfers sind zuverlässige Verbindungen, die den ordnungsgemäßen Empfang eines Datenpaktes sicherstellen. Dazu werden CRC und Antwort-Codes eingesetzt und im Falle einer fehlerhaften Übermittlung wird die Übertragung unter Software-Kontrolle wiederholt.

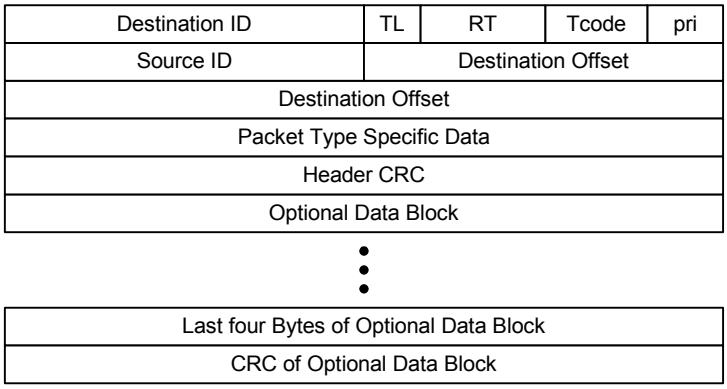

Abb. 6-34: Allgemeines Format Asynchroner Datenpakete nach [2]

Feldname	Beschreibung
Destination ID	Kombination der Busadresse und der physischen Bezeichnung des Knotens
TL	Transaction Label: Ein Bezeichner, festgelegt vom Urheber, der auch in der Antwort verwendet wird.
RT	Retry Code: Zeigt ob es sich bei diesem Paket um den ersten oder wiederholten Übertragungsversuch handelt.
Tcode	Transaction Code: Legt den Typ der Transaktion fest (Lese Anforderung, Lese Antwort, Acknowledge, usw.)
Pri	Priorität: Wir nur in Backplane Varianten benutzt.
Source ID	Absender dieses Pakets
Destination Offset	Gibt die Adresse im Adressraum des Zielknotens an.
Packet Type Specific Data	Kann genutzt werden, um die Länge der Daten anzuzeigen
Header CRC	CRC für die Headerdaten des Pakets.
Optional Data	Daten die zum Zielpunkt transferiert werden sollen.
Optional Data CRC	CRC für die Daten.

Tabelle 6-2 : Erläuterung zur allgemeinen Struktur asynchroner Datenpakete nach [2]

Bei isynchronen Transfers steht keine Fehlerkorrektur oder erneute Anforderung der Daten zur Verfügung. Außerdem wird keine eindeutige Adresse wie beim asynchronen Transfer verwendet, sondern nur eine 6-bit Kanalnummer. Das isynchrone Protokoll ist ganz besonders für zeitkritische Daten geeignet, da

mindestens 80% der Bandbreite garantiert sind. Video- oder Audiodaten zählen zum Beispiel zu den zeitkritischen Daten. Isynchrone Datenpakete enthalten u.a. Channel-Nummer, die Daten und eine CRC.

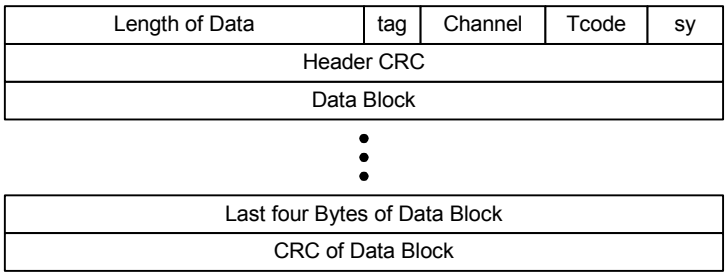

Abb. 6-35 : Allgemeines Format Isynchroner Datenpakete

Feldname	Beschreibung
Length of Data	Länge der Daten
tag	Isochrones Datenformat Tag. Der Wert 00_2 zeigt an, dass die isochronen Daten nicht formatiert sind. Alle anderen Werte sind reserviert.
Channel	Zielpunkt Adresse.
Tcode	Transaction Code: Legt den Typ der Transaktion fest. Eine isochrone Transaktion ist immer A_{16}
sy	Anwendungsspezifisches Feld
Header CRC	CRC für die Headerdaten des Pakets.
Data Block	Daten die zum Zielpunkt transferiert werden sollen.
Data CRC	CRC für die Daten.

Tabelle 6-3: Erläuterung zur allgemeinen Struktur isynchroner Datenpakete

6.8.4 IEEE Kommunikationsschichten

In der IEEE1394 Spezifikation werden vier Schichten definiert:

- Physikalische Schicht
- Sicherungsschicht
- Vermittlungsschicht (Transaction Layer)
- Transportschicht

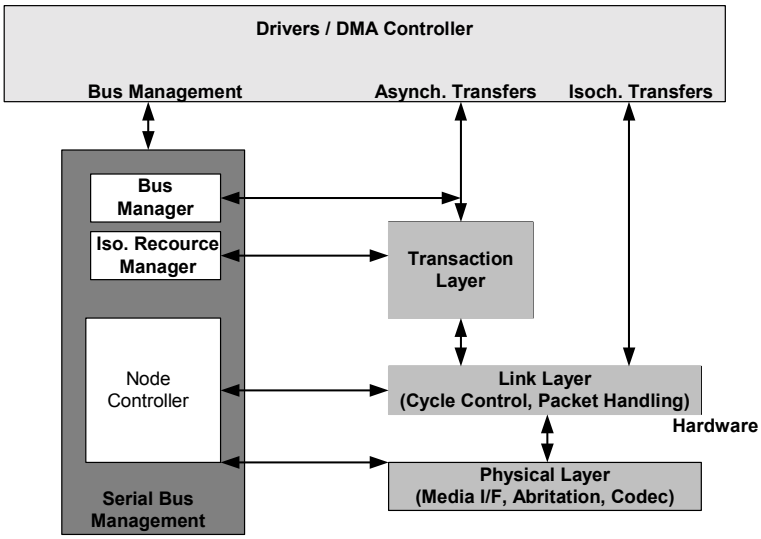

Abb. 6-36: IEEE 1394 Protokoll Layers nach [2]

6.8.5 Physikalische Schicht

Die Physikalische Schicht definiert die Eigenschaften und Kodierung der elektrischen Signale. Die Daten werden gemäß NRZ mit einem Strobe Signal und Daten RxD/ TxD übertragen. Das Verwenden eines Strobe Signals hat gegenüber der klassischen Variante mit Daten/Clock Vorteile in der Taktzurückgewinnung und dem Jitterverhalten.

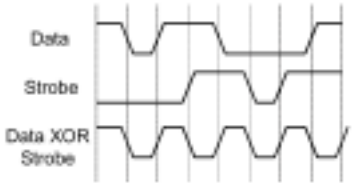

Abb. 6-37: Data Strobe Encoding nach [2]

Neben der elektrischen Eigenschaften werden außerdem auch die mechanischen Abmessungen der Steckverbinder und Kabel definiert. Zur Übertragung der Daten wird eine abgeschirmte vier oder sechs-polige Leitung benutzt. Zwei differentielle Paare dienen zur Übertragung der Daten, ein weiteres optionales Paar dient zur Energieversorgung der Knoten.

6.8.6 Identifikation der Baumstruktur

Nach jedem Reset verliert jeder Node das Wissen über die Bus Topologie. Während des Identifikationsprozesses wird die Bustopologie wieder neu definiert. Auch das entfernen oder anschließen von Geräten löst einen Reset aus. Im folgenden Beispiel wird anhand der folgenden Struktur der Prozess der Identifikation aufgezeigt.

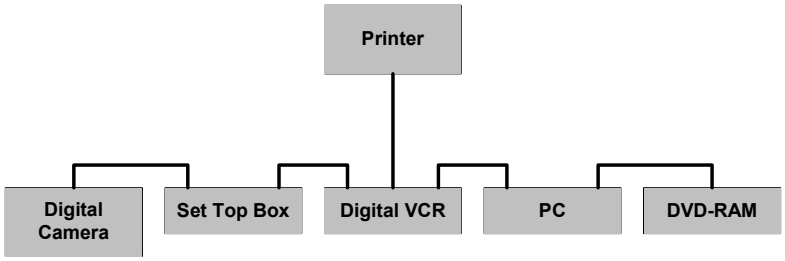

Abb. 6-38: IEEE1394 Architektur nach [2]

Nachdem Reset signalisieren alle einfachen Knoten (Knoten an denen nur ein Gerät angeschlossen ist) über ihre Daten und Strobe Leitungen ein Parent_Notify Signal. In unserem Beispiel signalisiert die Digitalkamera der Set-Top Box, der Drucker dem digitalen VCR und das DVD-RAM dem PC das *Parent_Notify* Signal.
Die verzeigten Knoten (mehrere angeschlossene Geräte) empfangen das *Parent_Notify* der einfachen Knoten und markieren diesen Port als Child und geben ein *Child_Notify* Signal an die einfachen Knoten aus. Verzweigte Knoten werden als Parent bezeichnet. Mit dem *Child_Notify* Signal nehmen die einfachen Knoten das *Parent_Notify* Signal wieder zurück, gleichzeitig dient dies auch zur Betätigung an den verzeigten Knoten. Anschließend werden die Knoten willkürlich durchnummeriert.

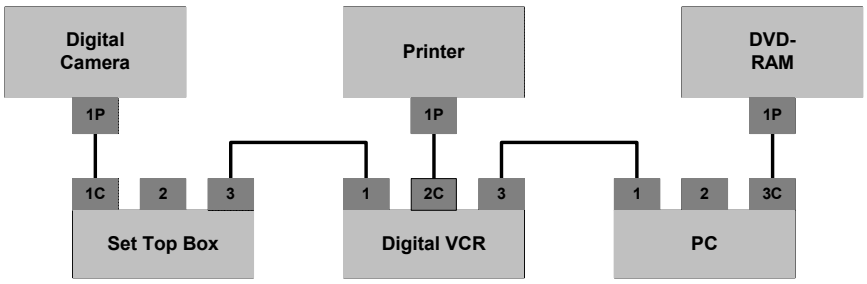

Abb. 6-39: Bus nach Knotenidentifikation nach [2]

Zum jetzigen Zeitpunkt sind die einfachen Knoten identifiziert. Aber der digitale VCR hat von zwei Ports noch kein Parent_Notify Signal erhalten, die Set-Top Box und PC haben von einem Port noch keine Antwort bekommen. In diesem Fall senden die Set-Top Box und der PC ein *Parent_Notify* Signal am den Port der noch nicht beantwortet wurde. Der digitale VCR bestätigt den Empfang der *Parent_Notify* Zustände mit dem *Child_Notify* Signal. Der VCR hat jetzt alle Ports als Children markiert und wird dadurch zum Root Node.

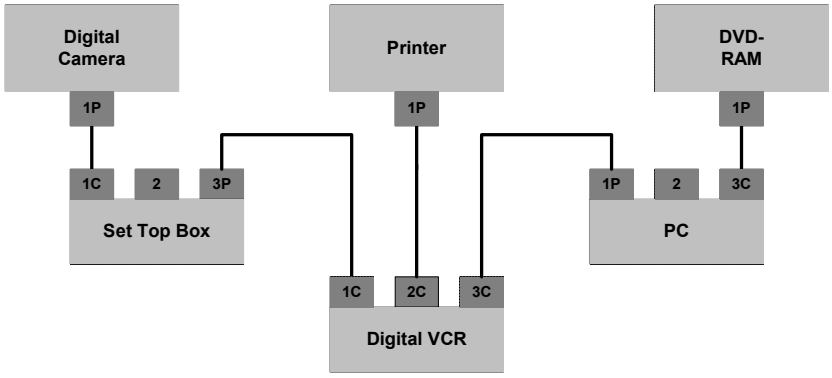

Abb. 6-40: Bus nach Knotenidentifikation 2 nach [2]

Natürlich ist es möglich das zwei Knoten zu Root-Knoten werden. Für diesem Fall wird sich für einen Root-Knoten entschieden.

6.8.7 Selbstidentifikation

Während der Selbstidentifikation werden allen Knoten physikalische Adressen zugewiesen, außerdem wird mit Nachbarknoten Informationen über Geschwindigkeit ausgetauscht.
Der Root-Knoten sendet ein *Arbitration Grant Signal* an den Port mit der kleinsten Nummer. Bei unserem Beispiel sendet der digitale VCR an die Set-Top Box. Da die Set-Top Box ein verzweigter Knoten ist leitet sie das *Arbitration Grant Signal* an einen ihrer Ports mit der kleinsten Nummer weiter. Hier an die Digitalkamera. Da die Digitalkamera ein einfacher Knoten ist, nimmt sie die physikalische Adresse 0 an and sendet eine *Self-ID* per Broadcast zurück. Des weiteren wird ein *SELF ID Done* Signal nur an die nächst höhere Hierarchie (hier: Set-Top Box) geschickt.
Nach Empfang dieser *Self-ID* inkrementieren alle Knoten ihren ID Counter. Der Root Knoten sendet ein weiteres Arbitration Grant Signal an den Port mit der kleinsten Nummer. In diesem Fall wird die Set-Top Box angesprochen, welche die Adresse 1 annimmt und anschließend *Self-ID* und *SELF ID Done* sendet.
Die Vorgang wird solange wiederholt bis an allen Ports des Root Knotens ein *SELF ID Done* Zustand detektiert wurde. Als nächsten Schritt gibt sich der Root Knoten die nächst höhere ID.
Für unser Beispiel wurden folgende IDs vergeben: Digitalkamera = 0, Set-Top Box = 1, Drucker = 2, DVD-RAM = 3, PC = 4 and VCR (Root Knoten) = 5.

6.8.8 Busarbitration

Das IEEE 1394 Protokoll wird in Zeitscheiben zu jeweils 125µs eingeteilt. Zu Beginn einer neuen Zeitscheibe sendet der Root Knoten ein Cycle Start Paket, mit dem sich alle anderen Knoten Synchronisieren.
In dieser Zeitscheibe gibt es jeweils für isynchrone und asynchrone Transaktionen Zeitspannen die Kennzeichnen, dass der Bus frei ist. Die eigentliche Arbitrationsregel ist trivial: Sind zwei Knoten gleichzeitig sendebereit, so beginnt der, der die höhere ID hat. Bei asynchronen Transfers kann es passieren das immer der Knoten mit der höchsten ID die Arbitration gewinnt, um dies zu verhindern wird zur Arbitration ein „Fairness Intervall“ verwendet. Innerhalb dessen, wird jedem Knoten, der eine asynchrone Transaktion durchführen will, genau einmal die Erlaubnis erteilt, ein Datenpaket zu senden.

Um eine Bandbreite von mindestens 80% für isynchrone Datentransfers zu garantieren, werden in einer Zeitscheibe mindestens 80% (100 µs) für den isynchronen Datentransfer zur Verfügung gestellt.

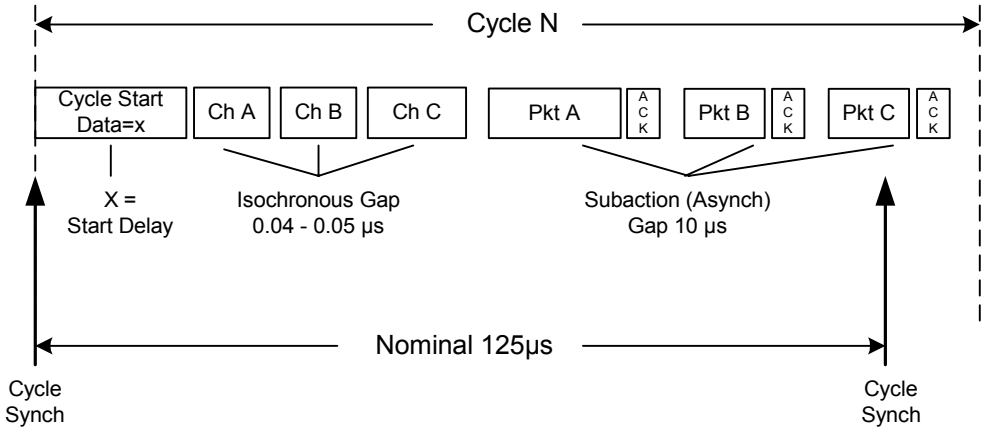

Abb. 6-41: Typischer IEEE1394 Zyklus

6.8.9 Transaction Layer

Der Transaction Layer wird für asynchrone Transfers genutzt. Es gibt fünf verschiedene Typen für asynchrone Transfers:

- Vier Byte Lesen
- Vier Byte schreiben
- Lesen von Bytes variabler Länge
- Schreiben von Bytes variabler Länge
- Swap und Compare Operationen

Asynchrone Datenpakete besitzen einen festen Headerteil und einen variablen Datenteil:

6.8.10 Weiterentwicklungen IEEE 1394

Die Automobil- sowie Systemhersteller wünschen sich einen offenen Standard, bei dem sich die unterschiedlichen Applikationen unabhängig vom Übertragungsmedium vernetzen lassen.
Im Jahre 1995 wurde von der Trade Association der Standard IEEE 1394 verabschiedet und im Jahre 2000 in einigen Punkten verbessert (IEEE1394.a). Mittlerweile liegt der Entwurf für eine erneute Erweiterung auf IEEE1394.b [6] bzw. IDB 1394 [7] vor. Dieser soll der fortschreitenden technischen Entwicklung Rechnung tragen.
Die ursprüngliche Begrenzung der Übertragungslänge wurde von etwa 4,5m bis 10m mit der Einführung einer neuen Signalcodierung (8B/10B) auf 100m ausgeweitet.
Als Übertragungsmedium stehen jetzt neben Kupfer auch Plastic Optical Fiber (POF) und Glasfaser (GOF) zur Verfügung.
Die bestehenden Geschwindigkeitsklasse von bis zu 400MBit/s wurde auf zurzeit auf 800Mbit/s erweitert. Darüber hinaus sind 1.6 bzw. 3.2 Gbit/s vorgesehen und bereits im Standard implementiert. Bisher ist im Bereich der Unterhaltungselektronik noch kein Gerät absehbar, welches die gesamte Bandbreite einer 3.2 Gbit Verbindung ausnutzen könnte. Die Entwicklungsziele waren eher an einer Zukunftssicheren Lösung orientiert, wie z.B. ein Backbone für Heimnetzwerke.

Eine weitere zukünftige Funktion ist die Segmentierung des Busses über Brücken. Mit Brücken wird es möglich sein Bussegmente unabhängig voneinander zu nutzen. Zur Zeit belegt ein Datentransfer noch den gesamten Bus.

Literatur

[1] Fachzeitschrift Elektronik: Mehr Bandbreite, größere Entfernungen; Joachim Kroll

[2] Questra Consulting: Fundamentals of Firewire; John Canosa

[3] keft Network: http://www.kefk.net/PDA/Technologie/Schnittstellen/Kabelgebunden/IEEE1394/index.html

[4] Fachzeitschrift Design&Elektronik: 1394 gibt Gas; Georg Haubner

[5] IEEE 1394-1995 Standard: http://www.ieee.org

[6] IEEE 1394.b Standard (Draft version): http://www.zayante.com/p1394b

[7] IDT 1394 Automotive Network: http://www.ami-c.org/