# 9 Übertragungsarten

Informationen, die ein Benutzer mithilfe einer Maschine übertragen will, werden letztlich in binärer Form mithilfe eines elektrischen Leitungssystems transportiert. Die Datenübertragung kann seriell oder parallel, durch Multiplexen sowie asynchron oder synchron geschehen. Hier geht es um Übertragungsarten auf der Schicht 1.

Applikation
7 Anwendung
6 Darstellung
5 Sitzung
4 Transport
3 Vermittlung
2 Sicherung
1 Bitübertragung
Übertragungsmedien

## 9.1 Serielle und parallele Übertragung

Datenströme können sowohl seriell als auch parallel übertragen werden. Serielle Übertragungen werden in vielen Schnittstellen am PC aber auch im LAN eingesetzt.

### 9.1.1 Serielle Übertragung

Bei der seriellen Übertragung werden die Daten in Bitströme zerlegt und hintereinander bitweise übertragen (Abbildung 9.1).
Die weit verbreitete RS-232-Schnittstelle verwendet dieses Verfahren, wie auch deren Nachfolgerin, die RS-449 oder der Universal Serial Bus (USB) an den PCs.

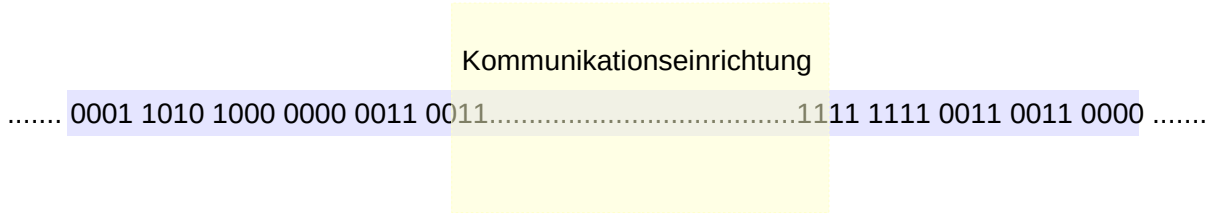



*Abbildung 9.1: Serielle Übertragung eines Bitstromes*

#### 9.1.1.1 Beispiel RS 232

Die Norm RS-232 wurde von der Electronic Industries Association (EIA) aufgestellt und existiert mittlerweile in der Fassung RS-232 C. In der Norm sind die mechanische, die elektrische, die funktionale und die verfahrenstechnische Spezifikation festgelegt.
Die *mechanische Spezifikation* definiert die Masse des 25- respektive 9-poligen Steckers.
Die *elektrische Spezifikation* legt fest, dass eine Spannung von weniger als –3 bis –15V eine binäre Eins und eine Spannung von mehr als +3 bis +15V eine binäre Null ergibt.
Die *funktionale Spezifikation* legt fest, welche Signale an welchem Anschluss (Pin) des Steckers anliegen.

**Praxis-Hinweis:**
Die maximalen Datenraten sind 20 kBit/s und es sind Kabellängen bis zu 15 m erlaubt.

#### 9.1.1.2 Beispiel USB

Der USB (Universal Serial Bus) hat sich zu einer Standardschnittstelle für periphere Geräte aller Art entwickelt. Die heutigen Multimediagräte weisen vorwiegend USB-Anschlüsse auf.

Die Geräte können dank Hot-Plugging zu jeder Zeit während dem Betrieb des PCs in den USB-Hostadapter des Rechners eingesteckt werden. Durch Plug&Play werden die eingesteckten Komponenten sofort erkannt und die Grundeinstellungen werden dann durch das Betriebssystem vorgenommen. Es müssen keine spezifischen Einstellungen mehr vorgenommen werden wie Jumperbelegung, korrekte Terminierung oder Protokolleinstellungen.

#### 9.1.1.2.1 Host

Es gibt nur einen Host in jedem USB-System. Das USB-Interface zum Host-Computer-System wird als Host-Controller bezeichnet, welcher normalerweise als Kombination von Hardware, Firmware und Software implementiert wird. Ein so genannter Root-Hub ist im Host-System integriert, um schon Anschlussmöglichkeiten für ein oder mehr Endgeräte zu ermöglichen.
Am Hostadapter (Anschluss Steckertyp A) können bis zu 127 Geräte (Steckertyp B) mithilfe eines oder mehrerer USB-HUBs angeschlossen werden.

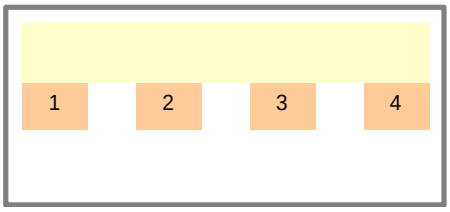



*Abbildung 9.3: Steckertyp A*

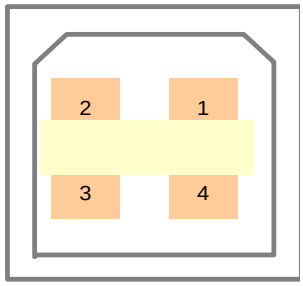



*Abbildung 9.2: Steckertyp B*

| *Pin* | *Beschreibung* | *Farbe* |
|---|---|---|
| 1 | Vbus 5V DC | Rot |
| 2 | D- Daten- | Weiss |
| 3 | D+ Daten+ | Grün |
| 4 | GND Erde/Masse | Schwarz |

*Tabelle 9.1: Die Pinbelegung der Stecker*

Für die modernen mobilen Geräte sind kleine Anschlüsse gefordert. Diese wurden mit Mini-A, Mini-B oder Mini-AB realisiert.
Geräte, die wenig Strom verbrauchen (bis 500mA), wie USB-Memorystick, Cardreader oder kleine Festplatten, können durch die von der Schnittstelle gelieferte Spannung von 5V ohne weitere Stromversorgung betrieben werden. Die Daten werden über ein Paar bidirektionale Leitungen gesendet (PIN 2+3). Das Kabel ist ein 4-adriges Twisted-Pair-Kabel.

#### 9.1.1.2.2 Datenübertragung

Die Datenübertragung erfolgt paketorientiert. Die einzelnen Frames werden bei USB 1.x im Millisekundentakt übertragen, während USB 2.0 diese jeweils weiter in 8 Hi-Speed-Frames zu 125µs unterteilt.

| *USB* | *Geschwindigkeit* | *Übertragungsrate* | *Länge* | *Kabel* |
|---|---|---|---|---|
| 1.1 | Low | 1.5Mb/s | 3m | UTP |
| 1.1 | Medium | 12Mb/s | 5m | STP |
| 2.0 | High | 480Mb/s | | STP |

*Tabelle 9.2: Die verschiedenen USB-Varianten (USB 2.0 ist abwärtskompatibel.)*

#### 9.1.1.2.3 USB-Baum

Ein Anschluss an einem Hub wird als Port bezeichnet. In einem USB-System kann es mehrere Hubs geben. Der Upstream-Port verbindet den Hub mit einem anderen Hub näher am Host oder direkt mit dem Host. Alle weiteren (Downstream-)Ports ermöglichen den Anschluss eines beliebigen USB-Geräts.

**Praxis-Hinweis:**
Hubs sind in der Lage, neu angeschlossene oder wieder entfernte Geräte automatisch zu erkennen und stellen die Energieversorgung für das entsprechende Gerät sicher. Ports von langsamen und normalen USB-Geräten werden voneinander isoliert.

#### 9.1.1.2.4 USB-Hub

Der USB-Hub besteht hardwaremässig aus dem Hub-Controller und dem Hub-Repeater. Ein Repeater ist ein protokollgesteuerter Schalter zwischen Upstream- und Downstream-Ports. Er besitzt ausser-

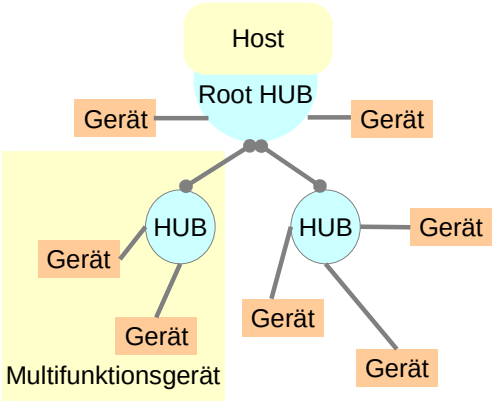



*Abbildung 9.4: Der USB-Baum*

dem Reset- und Energiesparfunktionen. Der Hub-Controller stellt Register zur Verfügung, um die Kommunikation mit dem Host zu ermöglichen. Spezifische Kontrollbefehle erlauben es dem Host, den Hub zu konfigurieren und seine Ports zu überwachen.

### 9.1.1.3 Beispiel IEEE1394 (Firewire)

Die IEEE 1394-Technologie, zunächst „Firewire“ genannt, bezeichnet eine verhältnismässig neue (seit 1995) serielle Schnittstellentechnologie für Computer- und Videogeräte zur Übertragung digitaler Daten mit bis zu 400 Mbit/s (IEEE1394A) und 800 Mbit/s (IEEE1394B). Die

Initiative und das Grundkonzept gehen auf die Firma Apple zurück. 1997/1998 benennt Sony die IEEE 1394-Schnittstellen der firmeneigenen Produkte von „FireWire“ in „i.LINK“ um.

#### 9.1.1.3.1 IEEE-1394 im Überblick

1. Paketorientierte Datenübermittlung
2. Geräte-Adressierung über Software
3. Hot-Pluggable
   Der Anwender kann 1394-Geräte ohne Werkzeug während des Systembetriebs anschliessen oder entfernen.
4. 63 Geräte anschliessbar
5. Bidirektional (Datenübertragung in beide Richtungen)
6. Datentransferraten von 100, 200, 400 und 800 Mbit/s beziehungsweise 12,5, 25 oder 50 MByte/s
7. Gemischter Betrieb unterschiedlich schneller Geräte mit 100, 200, 400 und 800 Mbit/s möglich
8. Dünne und preiswerte serielle Kabel
9. Einfache Konfiguration, da keine Abschlusswiderstände, Geräte-IDs oder Einstellungsverfahren notwendig sind.
10. Die Spannungsversorgung der Geräte ist über das Datenkabel möglich. Dafür sind zwischen 8 bis 40 Volt bei maximal 1,5 Ampere vorgesehen.
11. Als Peer to Peer-Netzwerk benötigt 1394 keinen dedizierten Host. Bei USB fungiert der PC als Host.

Die Firewire-Schnittstelle wurde ursprünglich benutzt, um digitale Camcorder an digitale Video-Hardware anzuschliessen. Es können aber auch Festplatten über Firewire angeschlossen werden.

IEEE 1394-1995 ermöglicht theoretisch eine Datentransferrate von bis zu 50 MB/s, mit IEEE 1395b bis 400 MB/s.

Prinzipiell sind Firewire, IEEE 1394-1995, i.Link und Lynx (Luchs) kompatibel zueinander oder nur andere Bezeichnungen für dasselbe. Allerdings unterscheidet sich i.Link schon äusserlich durch den kleineren vierpoligen Steckverbinder von den sechspoligen IEEE- und Firewire-Originalen. Der Unterschied: i.Link bietet keine Spannungsversorgung für externe Geräte über das 1394-Kabel.

**Praxis-Hinweis: Datenübertragung**
Die Übertragung erfolgt paketorientiert. Ab 2002 erfolgt die Übertragung mit 8B10B-Leitungscode, welcher eine grössere Leitungslänge zulässt.

#### 9.1.1.3.2 Anschluss der Geräte

Prinzipiell ist 1394 eine simple Sache. Ein oder mehrere Geräte mit dem seriellen Kabel an den PC/Controller anschliessen und fertig. Abgesehen von den Unwägbarkeiten eines Plug&Play-Betriebssystems, gibt es noch einige Einschränkungen. Die Geräte sind seriell hintereinander geschaltet wie in einer Kette. Bei Geräten mit zwei oder mehr 1394-Ports sind Verzweigungen möglich.

Der Anschluss erfolgt über ein flexibles 6-adriges STP-Kabel (4 Adern für Datentransfer, 2 Adern für Stromversorgung) oder 4-adriges Kabel (nur Signalleitungen).

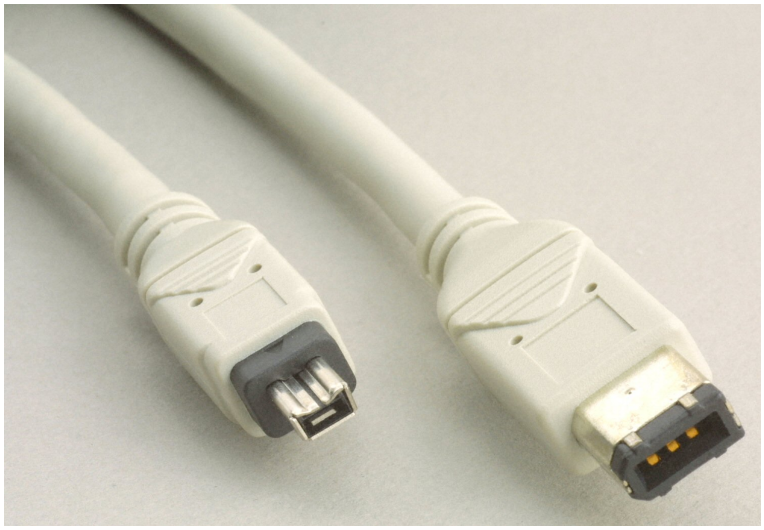
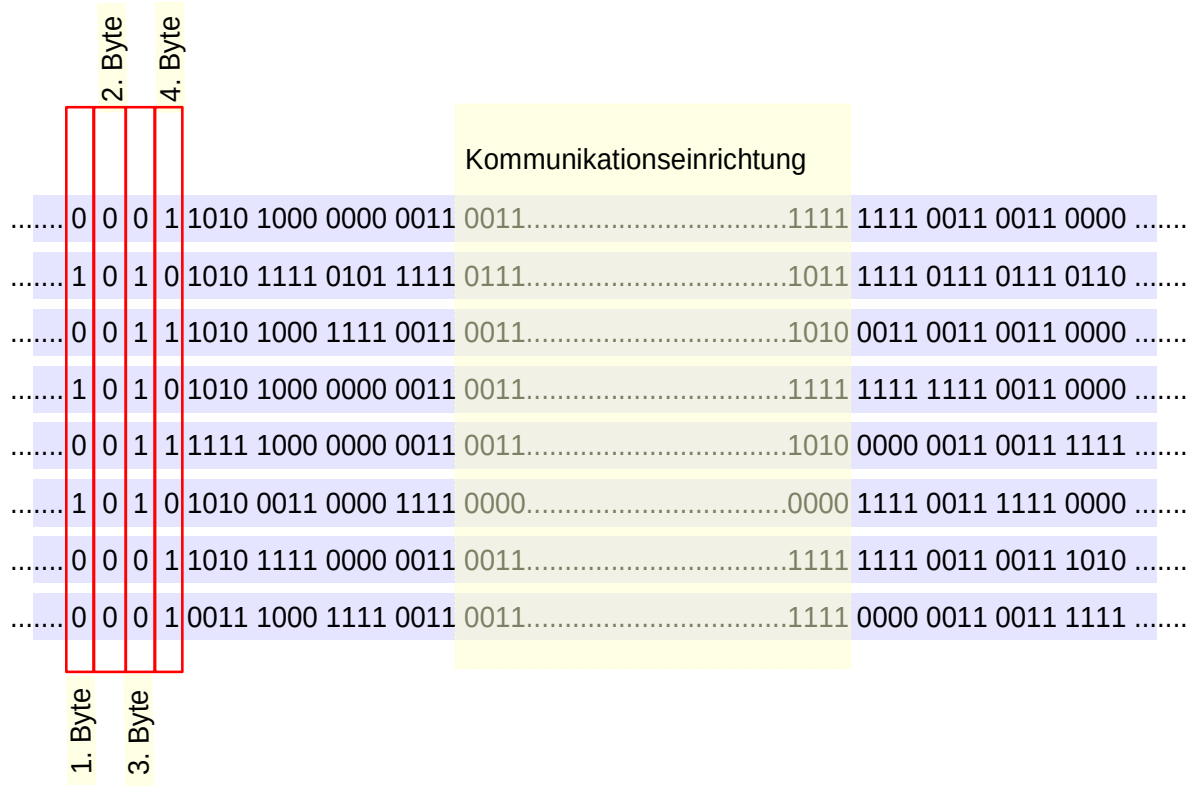


*Abbildung 9.5: Das IEEE 1394-Kabel*

### 9.1.2 Parallele Übertragung

Bei der parallelen Übertragung werden Informationen über mehrere Leitungen parallel übertragen. Ein Byte (8 Bit), Wortlängen von 16 Bit oder noch grössere Multiple eines Bytes werden gleichzeitig übermittelt (Abbildung 9.6).




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| ..... | 0 | 0 | 0 | 1 | 1010 1000 0000 0011 | 0011 | 1111 | 1111 0011 0011 0000 ..... |
| ..... | 1 | 0 | 1 | 0 | 1010 1111 0101 1111 | 0111 | 1011 | 1111 0111 0111 0110 ..... |
| ..... | 0 | 0 | 1 | 1 | 1010 1000 1111 0011 | 0011 | 1010 | 0011 0011 0011 0000 ..... |
| ..... | 1 | 0 | 1 | 0 | 1010 1000 0000 0011 | 0011 | 1111 | 1111 1111 0011 0000 ..... |
| ..... | 0 | 0 | 1 | 1 | 1111 1000 0000 0011 | 0011 | 1010 | 0000 0011 0011 1111 ..... |
| ..... | 1 | 0 | 1 | 0 | 1010 0011 0000 1111 | 0000 | 0000 | 1111 0011 1111 0000 ..... |
| ..... | 0 | 0 | 0 | 1 | 1010 1111 0000 0011 | 0011 | 1111 | 1111 0011 0011 1010 ..... |
| ..... | 0 | 0 | 0 | 1 | 0011 1000 1111 0011 | 0011 | 1111 | 0000 0011 0011 1111 ..... |

*Abbildung 9.6: Parallele Übertragung von 8 Bitströmen (= 1Byte) gleichzeitig*

Anwendungen: Centronics-Schnittstelle, IEC-Bus (auch bekannt unter dem Namen IEEE-488 oder GP-IB) und sämtliche System- und Peripheriebusse in Computern (Micro Channel (MCA), (E)ISA, PCI, VME-Bus, SCSI, Multibus I + II, NUBUS etc.).

## 9.2 Zeichenorientierte/Bitorientierte Übertragung

Es gibt grundsätzlich die Möglichkeit, Daten als Zeichen oder als Bitstrom zu übertragen. Beide Möglichkeiten sind in der Praxis häufig anzutreffen und werden im Folgenden erklärt. In beiden Fällen ist eine Strukturierung der Ströme sinnvoll für die effizientere Handhabung. Die Strukturierung der Zeichenströme in Worte mit klar definiertem Anfang und Ende sowie die Strukturierung der Bitströme in Frames (Rahmen) macht die Handhabung einfacher. Dies ist vergleichbar mit dem Sprechen in Worten und Sätzen – man stelle sich vor, die Menschen würden in Buchstaben und Leerzeichen kommunizieren!

**Praxis-Hinweis:**
Die Übertragung erfolgt meistens asynchron. (Eine synchrone Übertragung ist jedoch auch denkbar.) Der Empfänger kann bei der asynchronen Übertragung die Zeichen anhand der Start- und Stopp-Bits unterscheiden.

### 9.2.1 Zeichenorientierte Übertragung

Es existieren Geräte, deren Daten als Zeichen anfallen. Anzutreffen ist dies z.B. in Tastaturen, Steuerworte für die Steuerung von Maschinen und Inhalte von Bildschirmmasken in EDV-Anlagen. Die Datenkommunikationseinrichtungen, welche solche Nachrichten übermitteln, tun dies auf der Basis von einzelnen Zeichen. Die Zeichen (englisch: character) werden anhand einer Code-Tabelle (zum Beispiel dem ASCII-Code) in 7-Bit, 8-Bit oder neuerdings noch mehr Bit umcodiert. Acht-Bit codierte Zeichen heissen auch Byte. Diese codierten Zeichen werden als Bitstrom übertragen und vom Empfänger anhand der gleichen Code-Tabelle wieder in Zeichen umgewandelt.
Damit der Sender in einer zeichenorientierten Übertragung den Anfang und das Ende eines Textes markieren kann, werden besondere Steuerzeichen verwendet: STX für Start of Text und ETX für End of Text. In einer beliebigen zeichenorientierten Übertragung können aber STX und ETX zufälligerweise auch vorkommen. Es ist unschwer zu erkennen, dass dies beim Empfänger zu einer grösseren Katastrophe führen würde. Aus diesem Grund werden vom Sender vor allen Steuerzeichen DLE (Data Link Escape) gesetzt, um die Steuerzeichen eindeutig zu markieren. Leider können auch diese DLE in einem beliebigen Text vorkommen. Damit auch in diesem Fall keine Fehlinterpretationen beim Empfänger vorkommen können, wird das Verfahren des character stuffing (= Zeichen stopfen) eingesetzt. Der Sender fügt somit nicht nur vor jedem STX oder ETX ein DLE ein, sondern auch vor jedem DLE, das er vor dem „DLE-Stopfen“ im Text findet. Der Empfänger entfernt vor dem Lesen der Nachricht alle DLE

vor den Steuerzeichen. Findet er eine Folge von zwei DLE, weiss er, dass das erste gestopft ist und das zweite zum Text gehört.

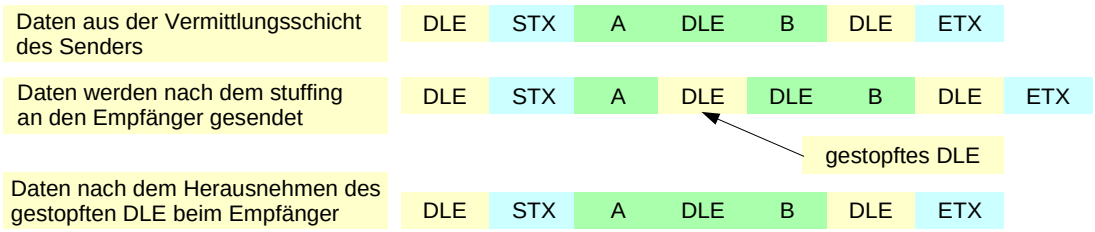



*Abbildung 9.7: Stopfen von DLE im Zeichenstrom*

### 9.2.2 Bit orientierte Übertragung

Bei der bitorientierten Übertragung liegen die Daten beim Sender schon als Bitstrom vor und könnten problemlos übertragen werden. Damit die Datenübertragung grosser Bitströme (grosse Dateien mit mehreren MByte) abgesichert erfolgen kann, muss der Bitstrom – ähnlich den Zeichen oder Bytes bei der zeichenorientierten Übertragung – strukturiert werden. Dies ist notwendig, damit zum Beispiel bei einem Unterbruch oder einem Fehler in der Übertragung die Kommunikation nicht neu gestartet und somit die ganze Datei erneut übertragen werden muss. Eine Einteilung in kleinere Einheiten ermöglicht nach dem erneuten Aufbau der Leitung oder der Beseitigung der Störung ein Fortführen der Übertragung.

In der Telematik existieren zwei Möglichkeiten zur Strukturierung der Bitströme. Die Bits werden in Frames (Rahmen) eingepackt oder in Cells (Zellen) organisiert. Diese Strukturierung erfolgt in der Schicht 2 des ISO/OSI-Modells. Die damit zusammenhängenden Verfahren und Protokolle werden in den nächsten zwei Kapiteln näher erläutert.

Damit der Empfänger erkennen kann, wann ein Frame beginnt und wann es zu Ende ist, werden Eröffnungs- und Endmarken in dem Bitstrom eingefügt (siehe Abbildung 9.8). Diese Eröffnungsmarken (opening flag) und Endmarken (closing flag) haben beispielsweise das Bitmuster 01111110.

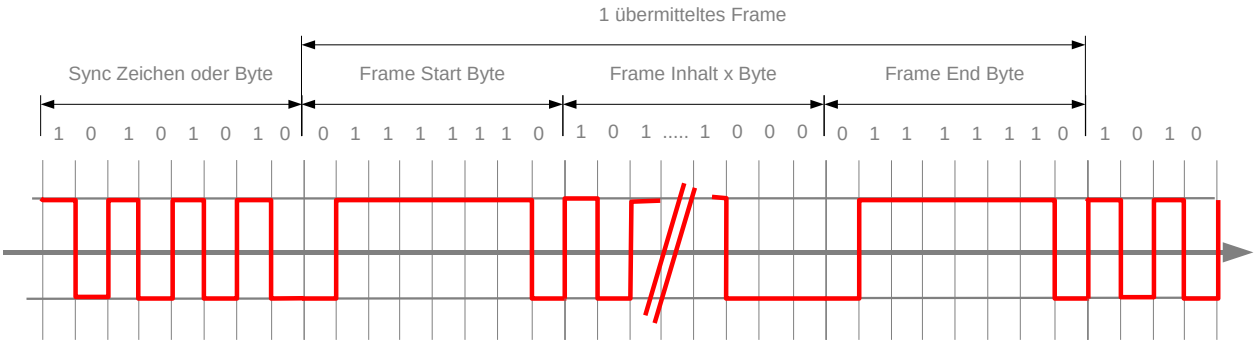



*Abbildung 9.8: Die Unterteilung des Bitstromes in Frames (Rahmen)*

Dummerweise ist es keine Seltenheit, dass in einem normalen Bitstrom das Bitmuster 01111110 vorkommt. Der Empfänger würde dieses Bitmuster natürlich als Marke erkennen und die Übertragung würde zusammenbrechen. Um dies zu verhindern, wird das bit stuffing (= Bit stopfen) eingesetzt (siehe Abbildung 9.9). Sobald der Sender in einem Bitstrom eine Folge von mehr als 5 Einsen erkennt, fügt er, vor der Einteilung des Bitstromes in Frames, nach 5 Einsen eine 0 ein. Der Empfänger entfernt nach der Erkennung der Frames alle Nullen, welche nach einer Folge von 5 Einsen im Bitstrom ankommen und erhält somit den ursprünglichen Bitstrom.

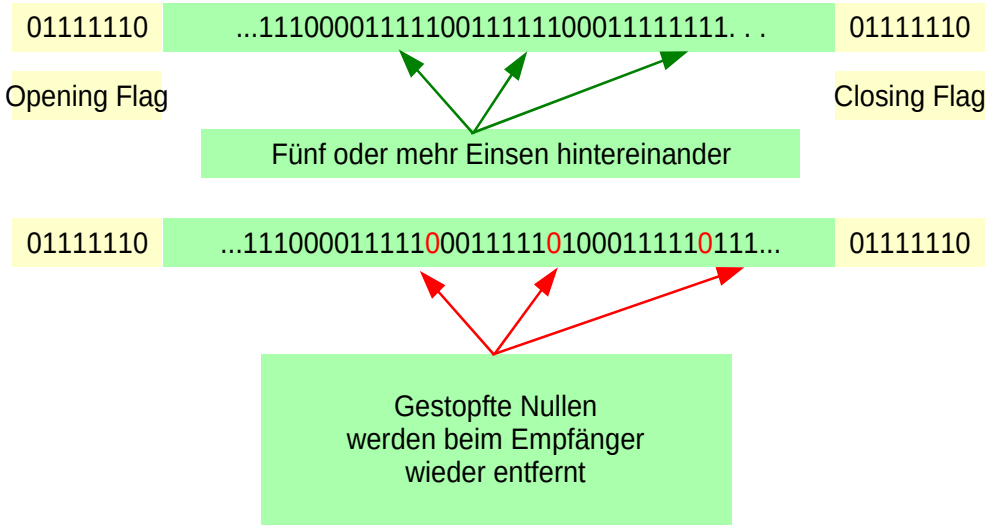



*Abbildung 9.9: Bit stuffing zur Vermeidung von Missverständnissen*

## 9.3 Asynchrone und synchrone Übertragung

Es ist unschwer einzusehen, dass es Geräte gibt, die konstant Daten generieren (wie zum Beispiel Messwerte einer Temperaturmessung) und solche, bei denen die Daten nicht kontinuierlich anfallen (wie zum Beispiel die Bewegung einer Maus oder die Daten aus einem Keyboard). Aus diesem Grund existieren auch zwei unterschiedliche Datenübertragungsarten – für den ersten Fall die synchrone und für den zweiten Fall eher die asynchrone Übertragung. Beide Fälle werden im Folgenden beschrieben.

### 9.3.1 Asynchrone Datenübertragung

Diese Übertragungsart wird häufig dort eingesetzt, wo die zu übertragenden Daten nur zeitweise anfallen (asynchron), wie z.B. bei der Dateneingabe per Tastatur. Hier ist typisch, dass nach dem Drücken einer Taste eine gewisse Zeit verstreicht bis zur nächsten Eingabe und die Übermittlung zum Computer somit nicht kontinuierlich erfolgt. Für die Datenübertragung heisst das, dass nach dem Übertragen einer Bitfolge (zum Beispiel acht Bit) eine gewisse Wartezeit (engl. id-

ling) vorliegt. Damit der Empfänger im Computer weiss, wann er das nächste Byte empfangen soll, muss das Byte mit einem Startbit angemeldet werden und am Schluss des Byte mit Stoppbits abgemeldet werden.

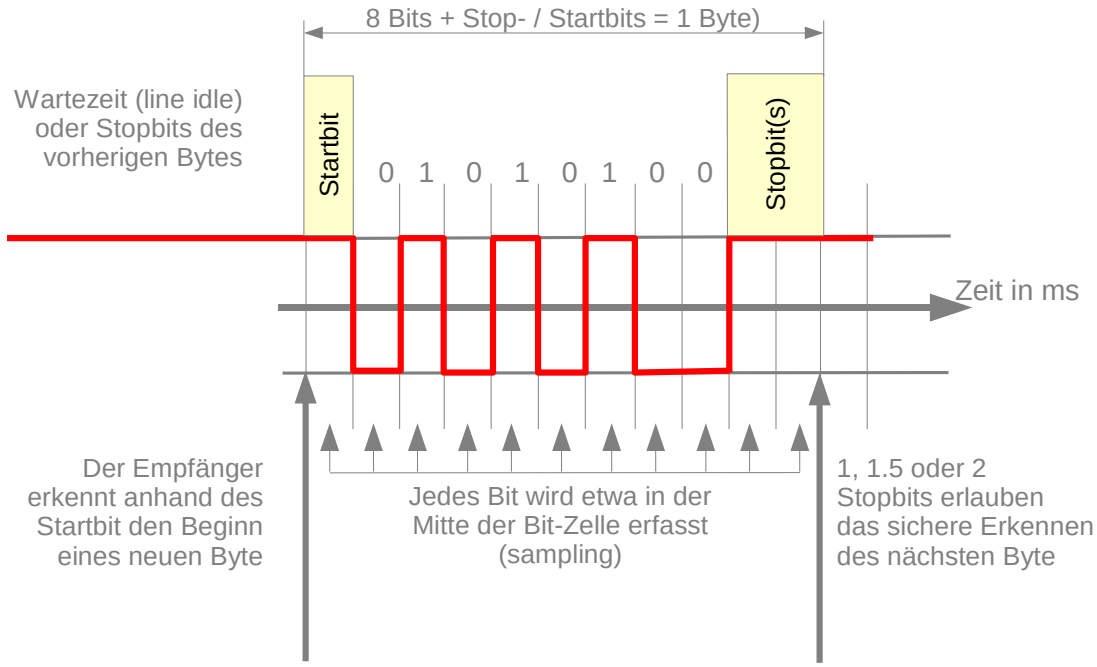



*Abbildung 9.10: Die Übertragung eines Byte*

In Abbildung 9.10 wird dieser Zusammenhang grafisch dargestellt (hier wird ein Byte, bestehend aus acht Bit, übertragen).

### 9.3.2 Synchrone Datenübertragung

Diese Übertragungsart eignet sich besonders für grössere Dateien oder grössere Übertragungsraten.

Im Gegensatz zur asynchronen Übertragung wird hier ein kontinuierlicher Datenstrom übertragen. Dies führt in der Praxis zu einer Schwierigkeit: Die Übertragung der Daten (Bits) erfolgt mittels elektronischer Schaltungen. Diese Schaltungen müssen auch bei hohen Übertragungsraten sowohl beim Sender als auch beim Empfänger genau wissen, wann gültige Bits zu schreiben respektive zu lesen sind. Die beiden Kommunikationsgeräte müssen das „gleichzeitig“, eben synchron tun. Damit der Empfänger weiss, wann gültige Bits ankommen, muss eine Synchronisationsinformation (Clock) mit übertragen werden.

Dies kann auf verschiedene Weise geschehen:

1. Abbildung 9.11 zeigt, wie auf einem separaten Leiter die Synchronisationsinformation parallel zu den Daten übertragen wird (out band).

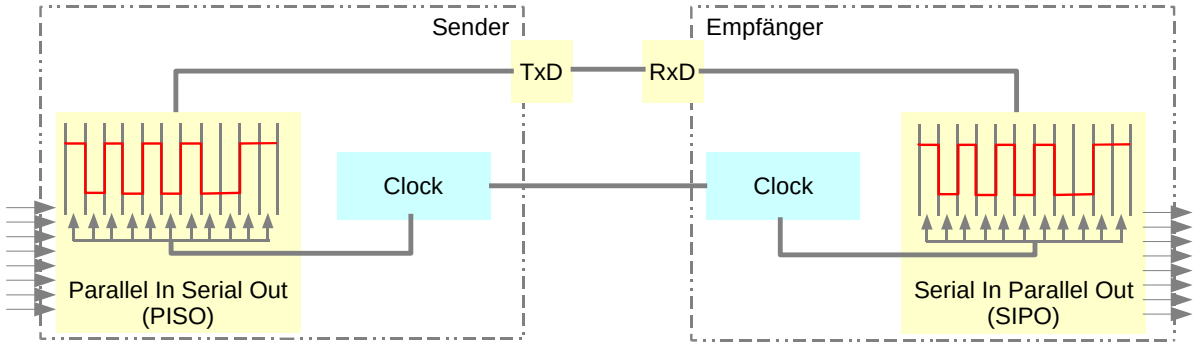



*Abbildung 9.11: Clockdaten out band*

2. Es werden spezielle Zeichen für die Clock-Information reserviert (in band).
3. Es werden einzelne spezielle Bits dazu verwendet (in band).

Damit diese in band clock-Übertragung möglich wird, muss der Bitstrom unterteilt und in so genannten Frames (Rahmen) zusammengefasst. Wenn keine Daten zur Übermittlung anfallen, werden die Frames mit Synchronisationszeichen (sync characters oder sync bytes) oder Synchronisationsbytes aufgefüllt. Jeder Frame hat ein Startzeichen oder einige Startbytes und ein Endzeichen oder Endbytes (siehe Abbildung 9.8).

Eine Möglichkeit, wie sie in der Praxis anzutreffen ist, um Datenströme synchron auf einer Leitung zu übertragen, zeigt Abbildung 9.12.

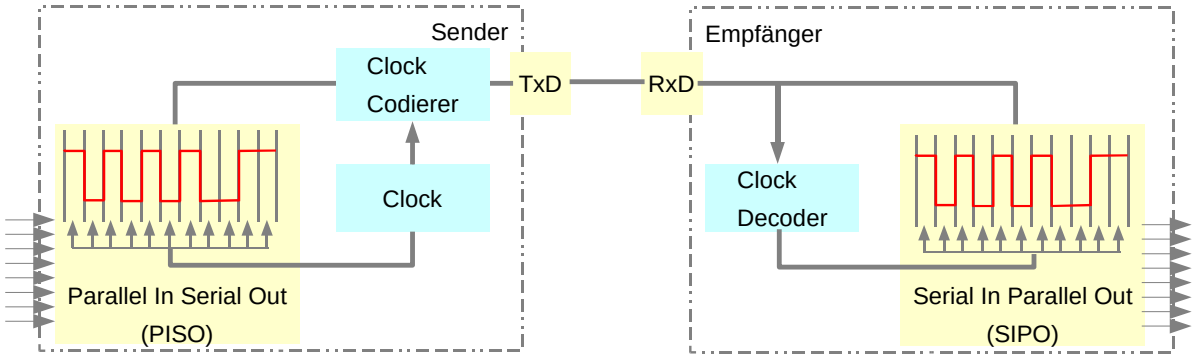



*Abbildung 9.12: Das Clocksignal wird mit Frames in den Datenstrom gepackt.*

Die oft parallel anfallenden Daten werden in einem Parallel/Seriell-Wandler (PISO, Parallel In Serial Out) umgewandelt und in einem Codierer mit einer Taktinformation (Clock) versehen. Die Daten werden als Frames strukturiert durch den Ausgang, Transmit Data (TxD), zum Eingang, Receive Data (RxD), geschickt. Der Empfänger muss diese Taktinformation mit einem Clock Decoder wieder herausfiltern, um den ursprünglichen Datenstrom zu erhalten.

## 9.4 Ankoppelung an die Schicht 2

Die Schicht 1 wird unterteilt in die beiden Teilschichten „Physical Medium Attachement“ (PMA) und „Physical Layer Signaling“ (PLS). In der ersten Teilschicht werden die Schnittstellen zu den verschiedenen Übertragungsmedien und in der zweiten Teilschicht wird die Signalerzeugung und die Ankoppelung an die Schicht 2 (Medium Access Control, MAC) definiert. In den Normen werden beispielsweise die Pinbelegung der Buchsen und Stecker, die Anzahl der benutzten Leitungen, die Art der Übertragung (seriell, parallel), die Funktionen der Steuerleitungen, die Trägerfrequenz und andere Funktionen beschrieben. Auf keinen Fall sind die Übertragungsmedien selbst Gegenstand der Schicht 1!
Abbildung 9.13 zeigt in einer groben Übersicht die Zusammenhänge der beiden Teilschichten und einige in der Praxis bei verschiedenen Implementationen eingesetzte Protokolle (AUI, MAU und MDI).

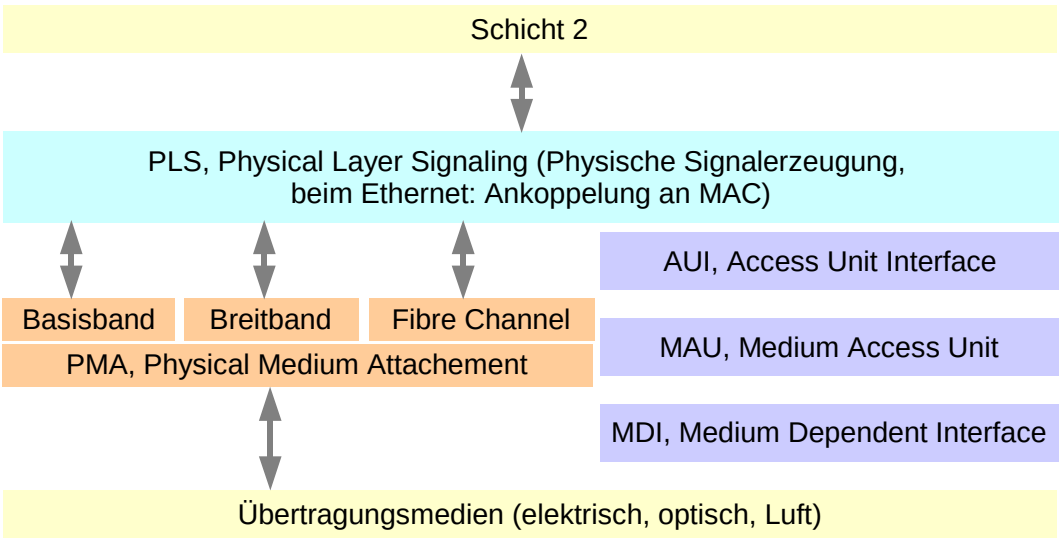



*Abbildung 9.13: Die Ankoppelung der Schicht 2*

## 9.5 Aufgaben

1. Gegeben sei eine asynchrone Datenübertragung mit serieller Schnittstelle (RS232):
   Ein Zeichen von 8Bit wird mit einer Übertragungsrate von 19200 Bit/s im Paritätsmodus übertragen. Wie gross ist die effektive Zeichenübertragungsrate?
2. Welche zusätzliche Steuerinformation wird für eine synchrone Datenübertragung benötigt?
3. Wie viele Hosts können in einem USB-System aktiv sein?
4. Wie erfolgt die Datenübertragung in einer USB-Verbindung?
5. Wie kann ein USB-Anschluss erweitert werden?
6. Wie wird die USB-Übertragungsklasse erkannt?
7. Wie viele Geräte können an einem USB-System angeschlossen sein?

Lösungen unter www.sauerlaender.ch/downloads

# 10 Übertragung der Bitströme

Applikation
7 Anwendung
6 Darstellung
5 Sitzung
4 Transport
3 Vermittlung
2 Sicherung
1 Bitübertragung
Übertragungsmedien

Das Kapitel behandelt die Strukturierung und Sicherung der Bitströme. Diese Verfahren sind notwendig, damit die Schicht 2 ihre Hauptfunktion des Aufbaus, des Betriebs und des Abbaus der Punkt-Punkt-Verbindung wahrnehmen kann.
Die Strukturierung geschieht mithilfe von Rahmen (engl. frames) oder Zellen (eng. cells).
Die Übertragungssicherung wird mit Übertragungs-Steuerungen erreicht. Die verschiedenen Möglichkeiten werden erläutert.
Im Weiteren werden die Übertragungsprotokolle beschrieben.

## 10.1 Strukturierung der Bitströme

Die Übertragung endloser unstrukturierter Bitströme ist nicht empfehlenswert. Dies wäre vergleichbar mit einer Rede, die nicht in Worte, Sätze und Abschnitte gegliedert ist, sondern als eine Aneinanderreihung von Buchstaben gehalten würde. Die Zuhörer hätten wohl kaum eine Chance, den Sinn der Rede erfassen zu können. Auch eine Fehlererkennung wäre problematisch. Genau so würde es aber einer Kommunikationseinrichtung ergehen, wenn man die Bitströme nicht strukturieren würde.

### 10.1.1 Rahmen (engl. frames)

Eine häufige Art der Strukturierung erfolgt mit Frames. Solche Frames sind z.B. im HDLC (High Level Datalink Control) Protokoll beschrieben. Dieses Protokoll benutzt die folgende Frame-Struktur:

| Marke Flag | Adressen | Steuerung | Daten / Information | Rahmen Prüfsumme | Marke Flag |
|---|---|---|---|---|---|
| 01111110 | Sender Empfänger | RR | xyzxyzxyzxyzxyz | CRC | 01111110 |
| 8 Bit | 8 oder mehr Bit | 8 oder 16 Bit | Mehr als 0 Bit | 16 oder 32 Bit | 8 Bit |

*Abbildung 10.1: Ein HDLC-Frame*

Die **Marke** (Flag) ist mit 01111110 festgelegt. Alle Stationen, die an das Netz angeschlossen sind, versuchen kontinuierlich, diese Flags zu finden und sich damit zu resynchronisieren. Selbstverständlich kann dieses Bitmuster auch innerhalb der Information vorkommen. Damit in diesem Fall keine Fehler auftreten, wird Bit-stuffing angewendet.

Das **Adressfeld** gibt an, welche Station (Empfänger) das Frame erhalten soll oder welche Station (Sender) das Frame generiert und auf das Netz gegeben hat. 11111111 ist eine reservierte Adresse und wird für das Broadcasting in Netzen verwendet (eine Station sendet eine Nachricht an alle).

Das Feld **Steuerung** bedeutet: Ein HDLC-Frame kann für verschiedene Aufgaben eingesetzt werden:

1. Es kann für die Übertragung von Nachrichten (Daten) benutzt werden. Dies mit so genannten Informations-Frames.
2. Die Verbindung kann mit Überwachungs-Frames überwacht werden.
3. Der Aufbau, der Betrieb und der Abbau der Verbindung wird mit so genannten unnummerierten (unnumbered) Frames gesteuert.

Damit der Empfänger versteht, auf welche Weise der Sender die Frames an ihn senden will, muss an dieser Stelle im Frame festgehalten werden, welcher Frametyp vorliegt.

Ein **Informations-Frame** wird Angaben zur Flusskontrolle (z.B. Sequenznummer und Angaben zum Timeout) enthalten.

Ist das Frame ein **Überwachungs-Frame**, so enthält es an dieser Stelle nur Angaben der Flusskontrolle. Es gelangen beispielsweise folgende Befehle zur Anwendung:

RR Receiver Ready
RNR Receiver Not Ready
REJ Reject
SREJ Selective Reject

**Unnummerierte Frames** halten den Betriebsmodus der Übertragung fest. HDLC kennt drei Betriebsmodi:

1. Den Normal Response Mode (NRM), bei dem ein Host seine Slaves zum Senden auffordern muss. Es können sowohl Punk-zu-Punkt-Netze als auch Multipoint-Netze eingesetzt werden.
2. Den Asynchronous Response Mode (ARM), bei dem auch ein Slave ohne Erlaubnis vom Host mit der Übertragung von Daten beginnen kann. Dies erfolgt normalerweise in Duplex-Punkt-zu-Punkt-Netzen.
3. Den Asynchronous Balanced Mode (ABM), bei dem es nur gleichberechtigte Kommunikationsteilnehmer (auf Duplex-Punkt-zu-Punkt-Netzen) gibt.

Die Frames teilen den gewählten Betriebsmodus der Gegenstation mit den folgenden Anweisungen mit:

SARM Set Asynchronous Response Mode
SARME Set Asynchronous Response Mode Extended
SNRM Set Normal Response Mode

SNRME Set Normal Response Mode Extended
SABM Set Asynchronous Balanced Mode
SABME Set Asynchronous Balanced Mode Extended
Zusätzlich enthalten sie die folgenden Link-Control-Befehle und -Funktionen sowie Flusskontrollzeichen:
RSET Reset
FRMR Frame Reject
DISC Disconnect
UA Unnumbered Acknowledge
CMDR Command Reject
DM Disconnect Mode
Die **Information** enthält die Daten (Bitfolge der Nachricht).
Der **Prüfbitrahmen** enthält die „cyclic redundancy checksum“ (CRC, Fehlerbehandlung).

**Praxis-Hinweis:**
Damit der Empfänger die Steuerbits der verschiedenen Frameteile (vor allem die Flags!) einwandfrei von allfällig gleichlautenden Bitfolgen im Informations-Teil (Daten) unterscheiden kann, gelangt hier das „bit stuffing“ zum Einsatz. Es ist hier noch wahrscheinlicher, dass eine Bitfolge im Datenteil genau der Bitfolge eines Flags entspricht. Der Empfänger würde die Bitfolge als Anfang oder Ende des Frames beurteilen und die Verbindung würde abgebrochen.

### 10.1.2 Zellen – Cells

Im ATM werden Zellen à 53 Byte eingesetzt. Diese Zellen haben im Gegensatz zu den HDLC-Frames eine fixe Grösse! Der Header der Zelle ist daher auch immer 5 Byte (40 Bit) gross und folgendermassen strukturiert:

| GFC | VPI | VCI | PT | CLP | HEC | Daten / Information |
|---|---|---|---|---|---|---|
| 1010 | 10010011 | 11001100 11001100 | 100 | 1 | 10010011 | xyzxyzxyzxyzxyz |
| 4 Bit | 8 Bit | 16 Bit | 3 Bit | 1 Bit | 8 Bit | Genau 48 Byte |
| Header: Genau 40 Bit = 5 Byte | | | | | | |
| Zellengrösse: Genau 53 Byte | | | | | | |

*Abbildung 10.2: Das Format einer ATM-Zelle (UNI)*

- GFC: 4 Bits für den *generic flow control.* Dieser wird lokal benutzt, um mehrere Stationen zu identifizieren, die alle das gleiche ATM Interface benutzen[60].
- VPI: 8 Bits für den *virtual path identifier.* Dieser wird im Zusammenhang mit dem Virtual Channel Identifier (VCI) benutzt, um die nächste Zellen-Destination im nächsten Switch festzulegen.
- VCI: 16 Bits für den *virtual channel identifier.* Dieser wird im Zusammenhang mit dem VPI benutzt, um die nächste Zellen-Destination im nächsten Switch festzulegen.
- PT: 3 Bits für den *payload type*. Das erste Bit wird benutzt um anzuzeigen, ob der Zelleninhalt für Steuerdaten oder Nutzdaten eingesetzt ist. Falls es sich um eine Nutzdaten-Zelle handelt, zeigt

60 kommt nur im Betriebsmodus UNI, User-Network Interface vor.

das zweite Bit den Zustand einer allfälligen Verstopfung und das dritte Bit ob die Zelle die letzte einer ganzen Serie ist.

- CLP: 1 Bit für das *congestion loss priority.* Dieses zeigt an, ob die Zelle wegen Leitungsverstopfung weggeworfen werden soll.
- HEC: 8 Bits für den *header error control.* Dies sind Checksummen über den Header.

### 10.1.3 Generic Framing Procedure (GFP)

Je mehr Protokolle integriert werden müssen auf einer Plattform (z.B. Ethernet, OC-x, SDH, ATM, auf SDH oder DWDM), desto wichtiger ist es, dass eine nach ITU genormte Rahmenstruktur vorhanden ist. Der Generic Framing Procedure-Rahmen ist nach ITU-T SG15 G.7041 genormt.

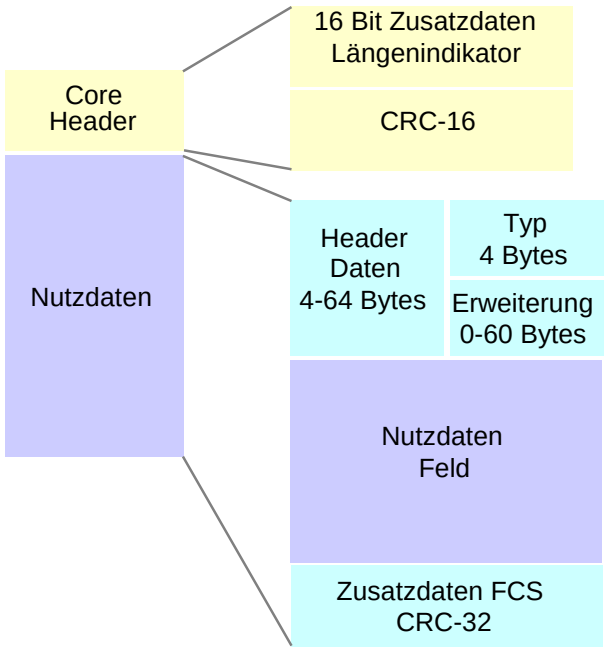



*Abbildung 10.3: Der GFP-Rahmen*

## 10.2 Sicherung der Übertragung

Für die Sicherung der Übertragung in einer Punkt-Punkt-Verbindung spielen unter anderem Begriffe wie Sequenznummern, Timeout und Flusskontrolle von Frames eine wichtige Rolle. Zudem existieren verschiedene Verfahren zur Sicherung (Resynchronisation) der Übertragung.

### 10.2.1 Resynchronisation

Für die Sicherung oder auch Resynchronisation der Übertragung werden die folgenden automatischen Verfahren (ARQ, Automatic Request) eingesetzt:

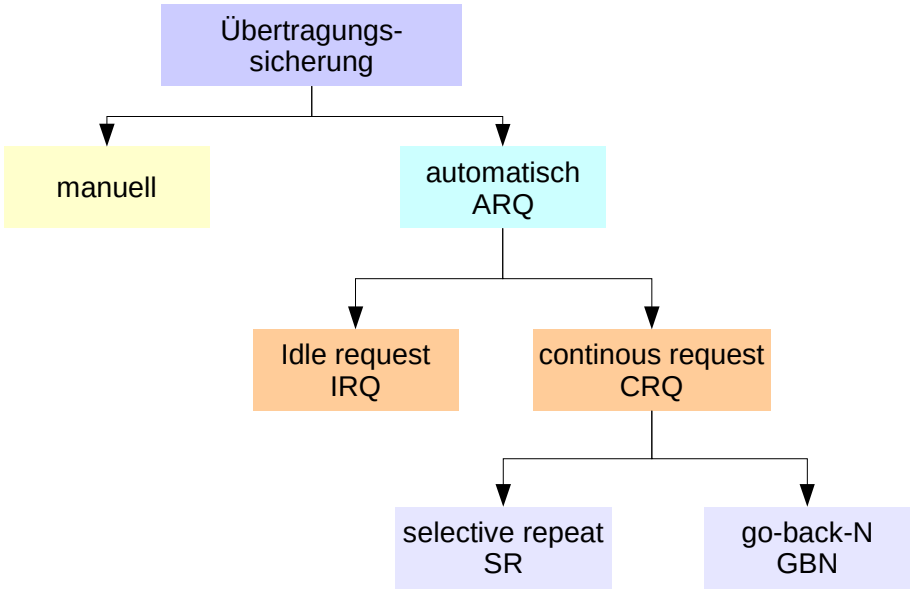



*Abbildung 10.4: Übersicht über die Übertragungssicherungs-Verfahren*

#### 10.2.1.1 Idle Request (IRQ)

IRQ, Idle request, funktioniert wie folgt[61]:

Ein Empfänger wartet, bis er von einem Sender einen Teil einer Nachricht (beispielsweise ein Frame oder ein Zeichen) erhält. Er kontrolliert den Inhalt des Frames (beispielsweise mithilfe einer Prüfsumme). Ist der Inhalt richtig, quittiert er den Erhalt des Frames.

Das folgende Beispiel verdeutlicht den Vorgang:

| Sender | Empfänger | |
|---|---|---|
| I | I | verstanden |
| c | c | verstanden |
| h | g | verstanden |
| nein h | h | verstanden |

Es ist zu beachten, dass hier nur das Zeichen wiederholt wird, das die Prüfung des Empfängers als nicht richtig erkannt und somit nicht bestätigt hat. Das Wort „verstanden“ wäre hier ein Bestätigungswort für den Sender.

Diese Art der Sicherung nennt man idle request (IRQ), weil der Sender so lange sendet, bis er merkt, dass der Empfänger ein Zeichen nicht oder falsch erhalten hat, welches er dann nochmals sendet.

#### 10.2.1.2 Continuous Request (CRQ)

Das Gegenstück zum IRQ sind die beiden Continous Request-Verfahren, Selective Repeat (SR) und Go Back N (GBN).

Die Continous Request-Verfahren arbeiten grundsätzlich wie folgt:

61 to idle: im Leerlauf warten, request: ersuchen, Aufforderung

**Praxis-Hinweis:**
Dieses Verfahren ist zwar langsam und daher vor allem für kurze Übertragungsstrecken und langsame Übertragungsraten geeignet (für Modems geeignet, nicht für LANs). Dafür ist die benötigte Pufferspeicherkapazität sehr klein, weil der Empfänger immer nur das aktuelle und das letzte Zeichen im Speicher behalten muss (um Verdoppelungen zu vermeiden).

**Praxis-Hinweis:**
Das selective repeat hat den Vorteil, dass nur wenige Daten, nämlich die fehlerhaften, noch einmal gesendet werden müssen, wohingegen beim Go-back-N eine ganze Sequenz neu gesendet wird. Dies ist bei Systemen mit wenig Sendeenergie (Satelliten) von Bedeutung.

**Praxis-Hinweis:**
Dieser Mechanismus unterliegt einem Timeout, da sonst der Sender bei einem Problem des Empfängers nie mehr frei würde und demzufolge keine weiteren Empfänger bedienen könnte.

Der Sender schickt dauernd Frames in den Empfangspuffer des Empfängers, ohne auf irgendeine Bestätigung zu warten. Der Empfänger untersucht die Frames auf ihre Richtigkeit und schickt eine Bestätigung in einem Frame an den Sender der Nachricht (Acknowledge, ACK). Die Bestätigung enthält die Framenummer der Nachricht. Anhand dieser Bestätigung kann der Sender eindeutig identifizieren, ob alle seine Frames richtig angekommen sind.
Falls der Sender erkennt, dass ein Frame verlorengegangen oder falsch angekommen ist, hat er grundsätzlich zwei Möglichkeiten, um den Fehler zu korrigieren:
Beim selective repeat wird anhand der Framenummer jeweils nur gerade dasjenige Frame nochmals gesendet, das als fehlerhaft erkannt wurde.
Beim Go back N wird nach einer gewissen Zeit erkannt, dass z.B. das fünftletzte Frame falsch war. Der Sender wird dann aufgefordert, fünf Frames zurück zu gehen und ab dort noch einmal zu senden.

### 10.2.2 Steuerung und Flusskontrolle

Damit Sender und Empfänger wissen, um welches Frame es sich handelt, wird dem Frame sowohl beim IRQ-Protokoll als auch beim CRQ eine *Sequenznummer* mitgegeben.

#### 10.2.2.1 Sequenznummern, Timeout, Flusskontrolle bei IRQ

Selbstverständlich können die Frames nicht einfach von eins bis n durchnummeriert werden, da sonst bei grossen Files die Nummer mit der Zeit grösser würde als das Frame.
Beim IRQ benötigt man genau zwei Frame-Sequenznummern, weil beim übernächsten Frame das Timeout des ersten sicher erreicht ist und somit die erste Nummer wieder frei wird. Typischerweise verwendet man hier 1 und 0 (siehe Abbildung 10.6, dort ist das erste Frame mit N und das zweite Frame mit N+1 bezeichnet).
Die Flusskontrolle kann hier mit Hardware realisiert werden, wenn eine RS232C-Schnittstelle mit RTS/CTS (Ready to send/Clear to send) verwendet wird oder ebenfalls mit einem Protokoll, dem X-ON- oder X-OFF-Protokoll (ASCII-Tabelle DC1, DC3 Kontrollzeichen). Falls ein Computer merkt, dass sein Puffer keine weiteren Daten mehr speichern kann, sendet er ein X-OFF an den Sender. Dieser stoppt die Übertragung und fährt wieder fort, sobald er ein X-ON erhält (empfangsbereit, Speicher verfügbar).

### 10.2.2.2 Sequenznummern und Flusskontrolle bei selective repeat und Go back N

Beim continous request geht man davon aus, dass ein Fluss von Frames kontinuierlich übermittelt wird ohne dauernd auf die ACK zu warten. Die Fehlererkennung findet daher ebenfalls kontinuierlich statt.
Damit zur Fehlererkennung genügend Rechenzeit zur Verfügung steht, werden bei den Teilnehmern der Kommunikation FIFO-Speicher (First In First Out-Speicher) benötigt. Damit die Puffer-Speicher und die Framenummern nicht beliebig gross werden, wird jeweils nur ein Ausschnitt aus dem Framestrom betrachtet. Dieser Ausschnitt wird „Fenster“ (engl. window) genannt. Weil sich das Fenster während der Übertragung dauernd über dem Framestrom verschiebt, benutzt man den Begriff „sliding window“ (Schiebefenster). Die Grösse des sliding window muss bei beiden Kommunikationspartnern gleich eingestellt sein, sonst ist die Übertragung nicht möglich.

**Praxis-Hinweis:**
Der grosse Nachteil beim idle request liegt darin, dass nach jeder Übermittlung eines Frames auf die Bestätigung (ACK/NAK) gewartet (engl. idling) werden muss.

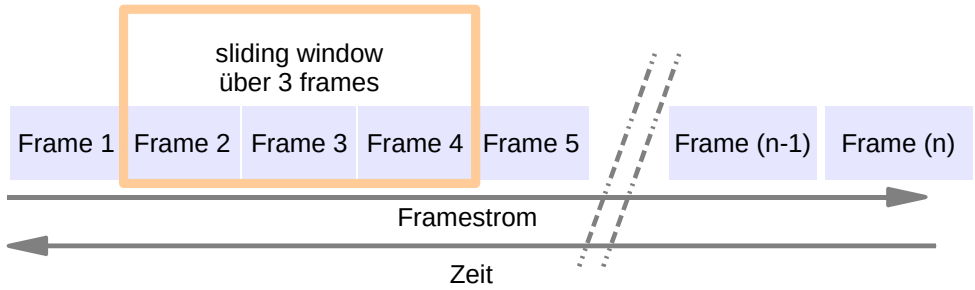



*Abbildung 10.5: sliding windows*

Dank der FIFO-Speicher ist es den Teilnehmern möglich, die Frames in einer beliebigen Reihenfolge und mit variablen Fenstergrössen übermitteln zu können und der Partner kann anhand der Framenummern die Nachricht wieder fehlerfrei zusammensetzen.
Die Anzahl der übermittelten Frames kann beliebig gross sein. Eine Nummerierung von 1 bis n ist daher sicher unsinnig, weil mit der Zeit die Nummerngrösse den Frameinhalt übertreffen würde.
Die Anzahl der Sequenznummern sind somit mindestens:

**Praxis-Hinweis:**
In der Praxis sind die windows während einer Übertragung variabel. Die Grösse der Windows wird somit dauernd optimiert und der jeweiligen Leitungsqualität angepasst. Der Einfachheit halber benutzen in diesem Buch alle Beispiele fixe window-Grössen.

| *Protokoll* | *Anzahl Frames im gesendeten Fenster* | *Anzahl Frames im empfangenen Fenster* | *benötigte Sequenznummern* |
|---|---|---|---|
| IRQ | 1 | 1 | 2 |
| selective repeat | n | n | 2*n |
| Go back N | n | 1 | n+1 |

*Tabelle 10.1: Anzahl benötigter Sequenznummern*

**Praxis-Hinweis:**
Bei der idle request-Methode kommt man mit zwei Nummern aus, weil immer gewartet wird, bis das Frame bestätigt ist und somit die verwendete Nummer wieder frei wird.
Damit selective repeat eindeutig in der Lage ist, alle n-gesendeten Frames im Schiebefenster zu bestätigen, benötigt es 2*n Sequenz- oder Framenummern, weil zuerst alle gesendeten Frames innerhalb des Schiebefensters vom Empfänger bestätigt sein müssen, bevor die Nummern vom Sender wieder belegt werden können.
Das Go back N verlangt nur n+1 Nummern, weil sich beim Empfänger immer nur ein Frame im Puffer befindet. Merkt der Sender, dass der Empfänger ein Frame nicht richtig erhalten hat, so sendet er sowieso noch einmal alle Frames.

### 10.2.3 Genauere Betrachtung des IRQ

Jeder vom Sender abgeschickte Datenblock (Frame) wird vom Empfänger durch einen Steuerblock ACK (engl. acknowledged) bestätigt. Der Empfang von durch die Übermittlung verfälschten Frames kann dem Sender auch durch eine negative Bestätigung (NAK, engl. not acknowledged) angezeigt werden.

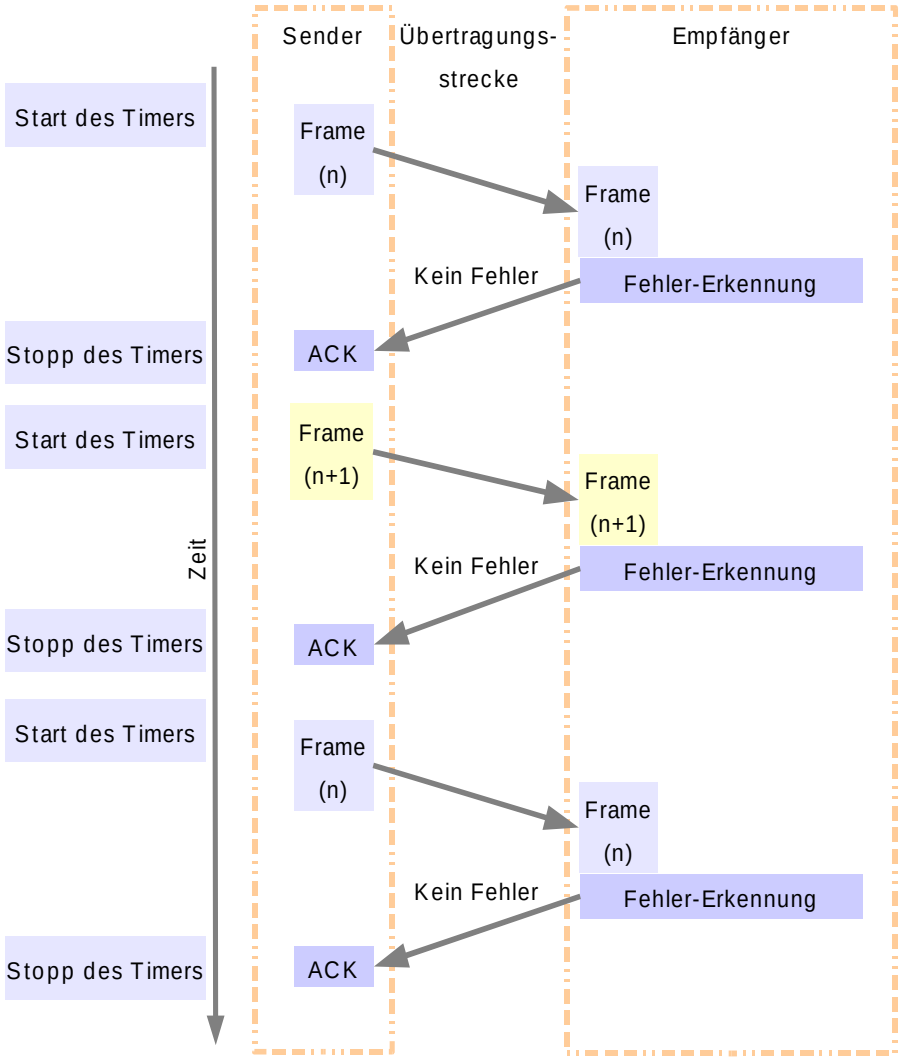



*Abbildung 10.6: IRQ ohne Fehler*

Abbildung 10.6 zeigt eine fehlerfreie IRQ-Übertragung.
Der Sender kann das folgende Frame, F(n+1), erst senden, wenn er das ACK für das erste Frame, F(n), erhalten hat. Der Sender interpretiert somit eine fehlende Bestätigung als Fehler im Frame und sendet das Frame noch einmal.
Abbildung 10.7 zeigt den Vorgang, wenn ein Frame nicht angekommen ist.

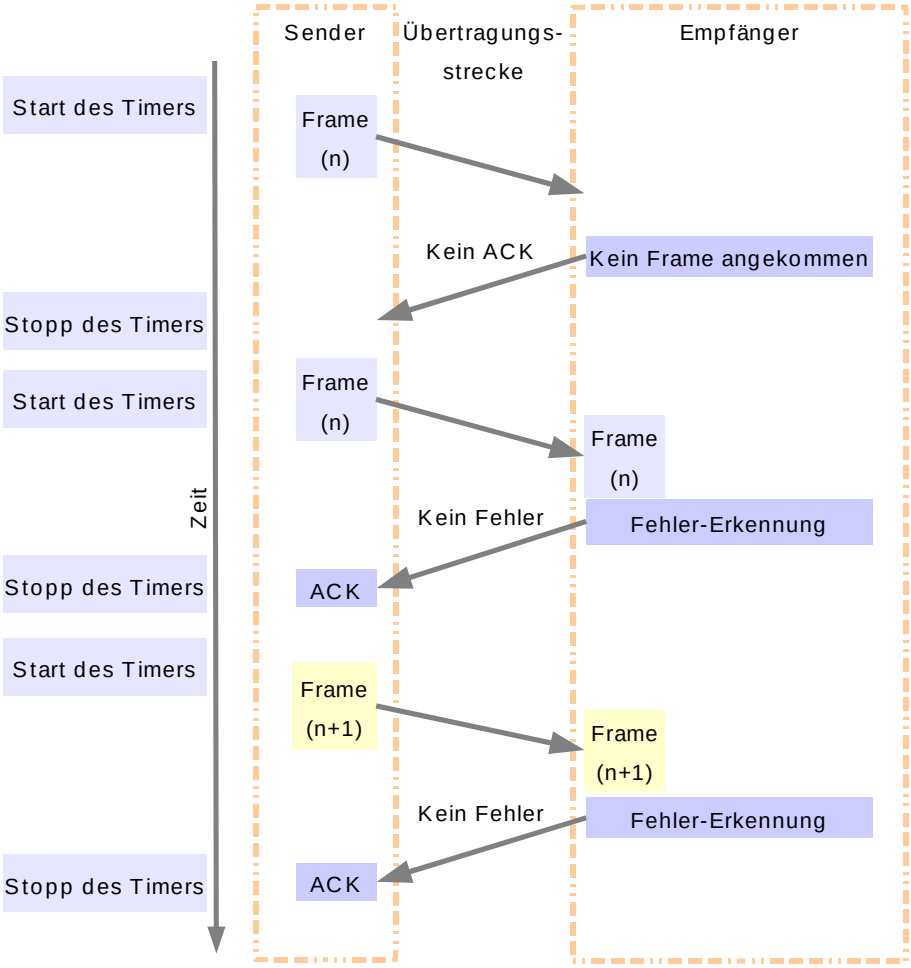



*Abbildung 10.7: Ein Frame geht verloren, es wird kein ACK gesendet.*

Wenn ein Frame nicht ankommt, sendet der Empfänger kein ACK. Wenn der Sender innerhalb seiner Timeoutzeit kein ACK bekommt, sendet er das Frame noch einmal.

Was geschieht aber, wenn ein ACK verloren geht? Der Sender bekommt innerhalb seiner Timeoutzeit kein ACK und sendet das Frame noch einmal. Der Empfänger hat das Frame also doppelt erhalten. Der Empfänger muss also eine Möglichkeit haben, die doppelten Frames zu erkennen und nur eine Kopie zu behalten.

Damit einwandfreie Frames vom Empfänger aufgrund eines fehlenden ACK nicht doppelt weitergeleitet werden, muss der Empfänger mit einer Möglichkeit der Doppelerkennung ausgestattet sein.

Abbildung 10.8 zeigt eine Übertragung, bei der das ACK beim Sender nicht ankommt, das fehlerfrei übertragene Frame somit doppelt ankommt. Dies muss erkannt werden und das doppelte Frame muss eliminiert werden.

In Abbildung 10.8 wurden schliesslich die Frames n und n+1 korrekt übertragen, was der Empfänger mit ACK bestätigte.

**Praxis-Hinweis:**
Eine Alternative in der Praxis ist die Bestätigung mit NAK: Der Empfänger sendet bei fehlerhaften Frames eine negative Bestätigung (not acknowledged, NAK) zurück. Der Sender kann somit auch fehlerhafte Frames von verloren gegangenen Bestätigungen unterscheiden.

**Praxis-Hinweis:**
Bei allen Beispielen wird eine Grösse des sliding window von n = 2 angenommen. Die Beispiele sind vereinfacht dargestellt, d.h. die in der Praxis übliche Veränderung der Fenstergrösse (window size) während der Übermittlung wird nicht dargestellt. Es wird von einer unveränderlichen Fenstergrösse ausgegangen.

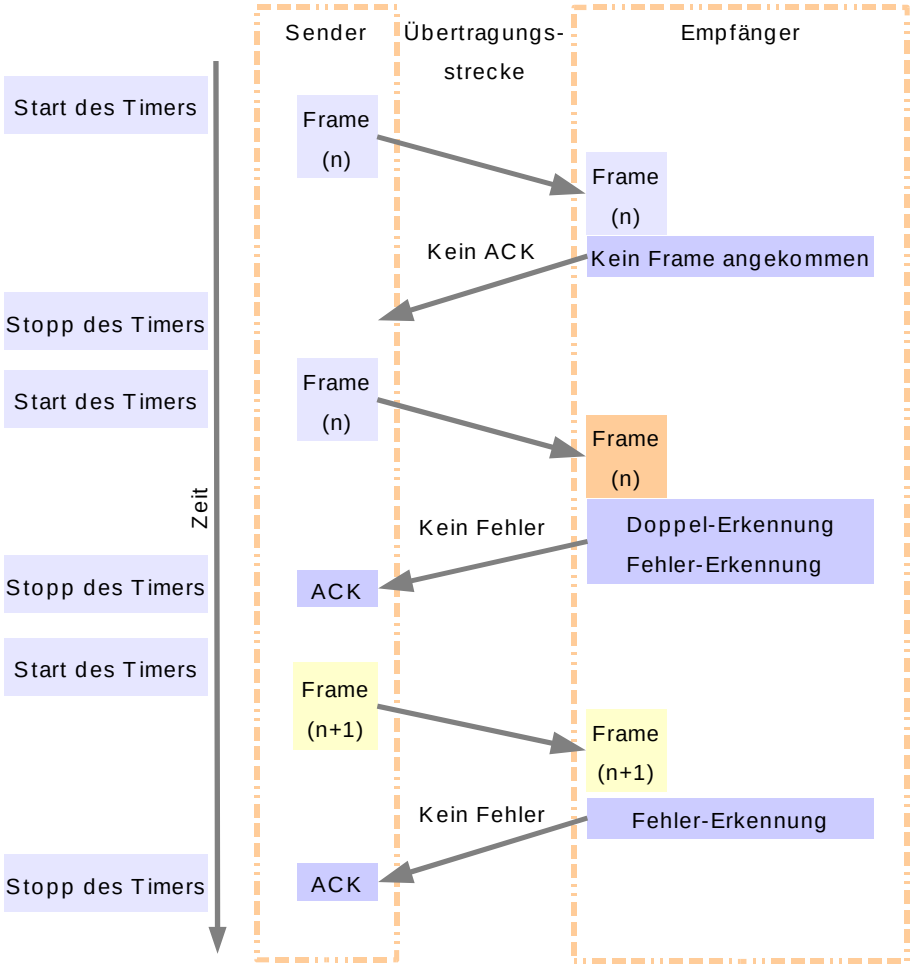



*Abbildung 10.8: Ein ACK kommt beim Sender nicht an.*

### 10.2.4 Genauere Betrachtung des CRQ

Betrachten wir wieder die verschiedenen Situationen, die auftreten können: Die Frames werden korrekt übermittelt und mit ACK oder NAK bestätigt; es gehen Frames verloren und Bestätigungen gehen verloren.

#### 10.2.4.1 Selective Repeat

Abbildung 10.9, alle Frames werden korrekt übermittelt:

1. Zum Zeitpunkt (A) übermittelt der Sender die Frames 1 und 2 zum Empfänger. Der Empfänger speichert die Frames im FIFO-Puffer-Speicher und führt eine Fehlererkennung durch. Wenn er keine Fehler entdecken kann, teilt er dies dem Sender mit, indem er ACK1 und ACK2 sendet.
2. Zum Zeitpunkt (B) sind die ACK1 und ACK2 aufgrund der Zeitverzögerung auf dem Übertragungsmedium noch nicht beim Sender eingetroffen. Dieser schickt trotzdem die Frames 3 und 4 (window über 3 und 4). Die Frames 1 und 2 behält er noch im Speicher.

3. Zum Zeitpunkt (C) sind die ACK1 und ACK2 angekommen und der Sender kann die zugehörigen Frames aus dem Speicher löschen, damit Platz für neue Frames entsteht. Gleichzeitig kann er die neuen Frames 1’ und 2’ senden. Der Empfänger hat in der Zwischenzeit die Frames 1 und 2 aus dem FIFO-Speicher genommen und weiterverarbeitet, sodass auch sein Speicher wieder frei wird für neue Frames. Der FIFO-Puffer des Empfängers ist grösser als nötig, damit im Falle einer Verzögerung genügend Platz für den Empfang weiterer Frames bleibt.
4. Ab Zeitpunkt (D) wiederholt sich das Prozedere, bis der Sender alle Daten der Nachricht übermittelt hat.

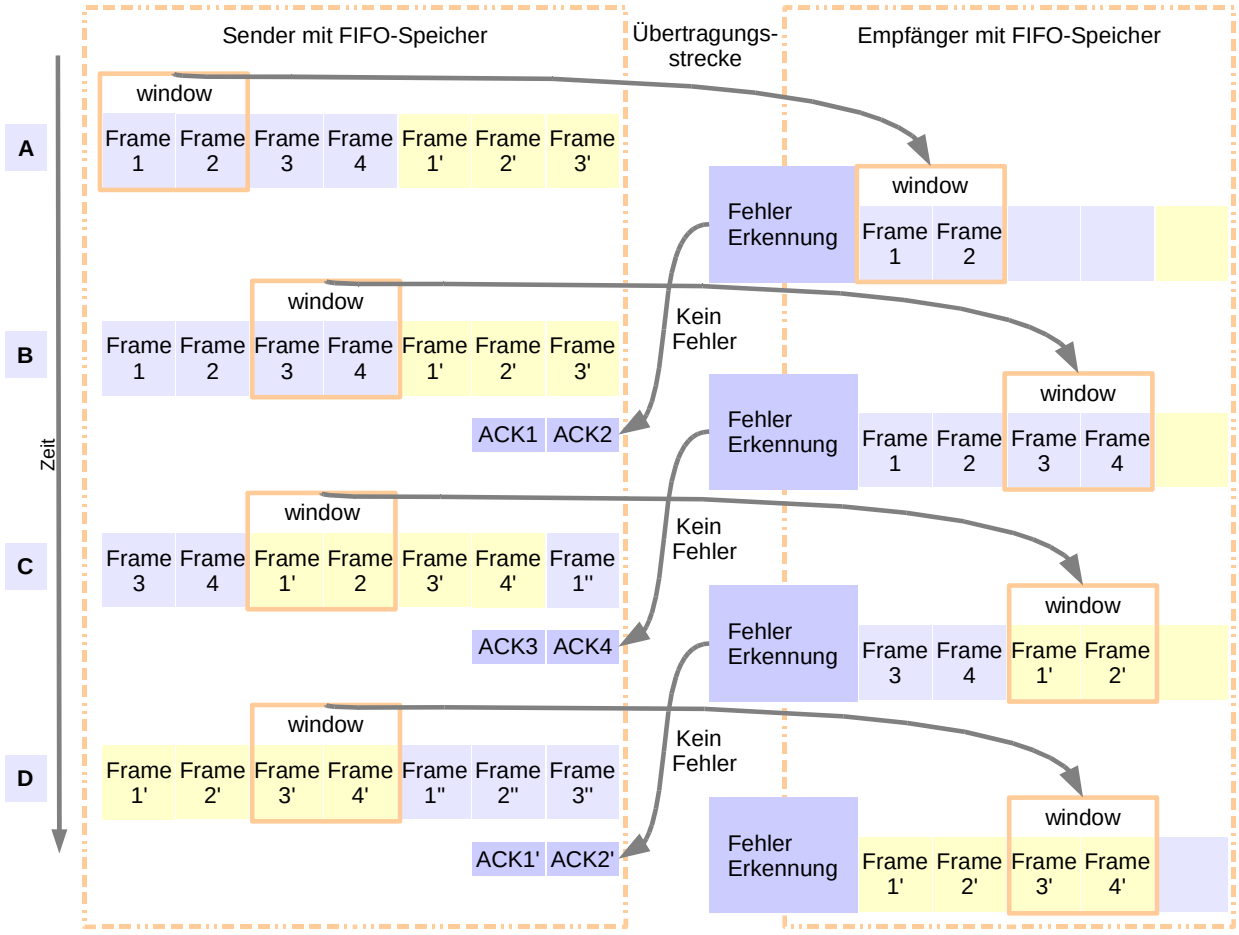



*Abbildung 10.9: Korrekt übermittelte Frames mit SR*

Abbildung 10.10, Frame 1 geht verloren:

1. Zum Zeitpunkt (A) sendet der Sender die Frames 1 und 2. Frame 1 geht dabei verloren.
2. Zum Zeitpunkt (B) übermittelt der Sender ohne Unterbruch die Frames 3 und 4. Der Sender erhält jetzt das ACK2 der Fehlerer-

kennung des Empfängers. ACK1 wird keines verschickt, da das Frame 1 fehlerhaft ist. Das window des Senders bleibt über Frame 1 und 2 stehen, bis ACK1, ACK2, ACK3 und ACK4 übermittelt sind.

3. Zum Zeitpunkt (C) übermittelt der Sender das Frame 1 noch einmal und der Empfänger bestätigt das Frame 1 mit ACK1.
4. Zum Zeitpunkt (D) muss der Sender auf das ACK1 warten und kann somit nicht senden – der Framestrom wird für kurze Zeit unterbrochen.
5. Zur Zeit (E) kann der Sender die alten Frames 1 bis 4 aus dem Speicher entfernen, da sie alle bestätigt sind. Er kann die neuen Frames 1’ und 2’ senden. Der Empfänger ordnet die alten Frames 1 bis 4, gibt sie zur weiteren Verarbeitung frei und hat somit Platz für die neuen Frames. (Damit im Beispiel zwischen den „alten“ und den „neuen“ Frames 1 bis 4 unterschieden werden kann, sind die

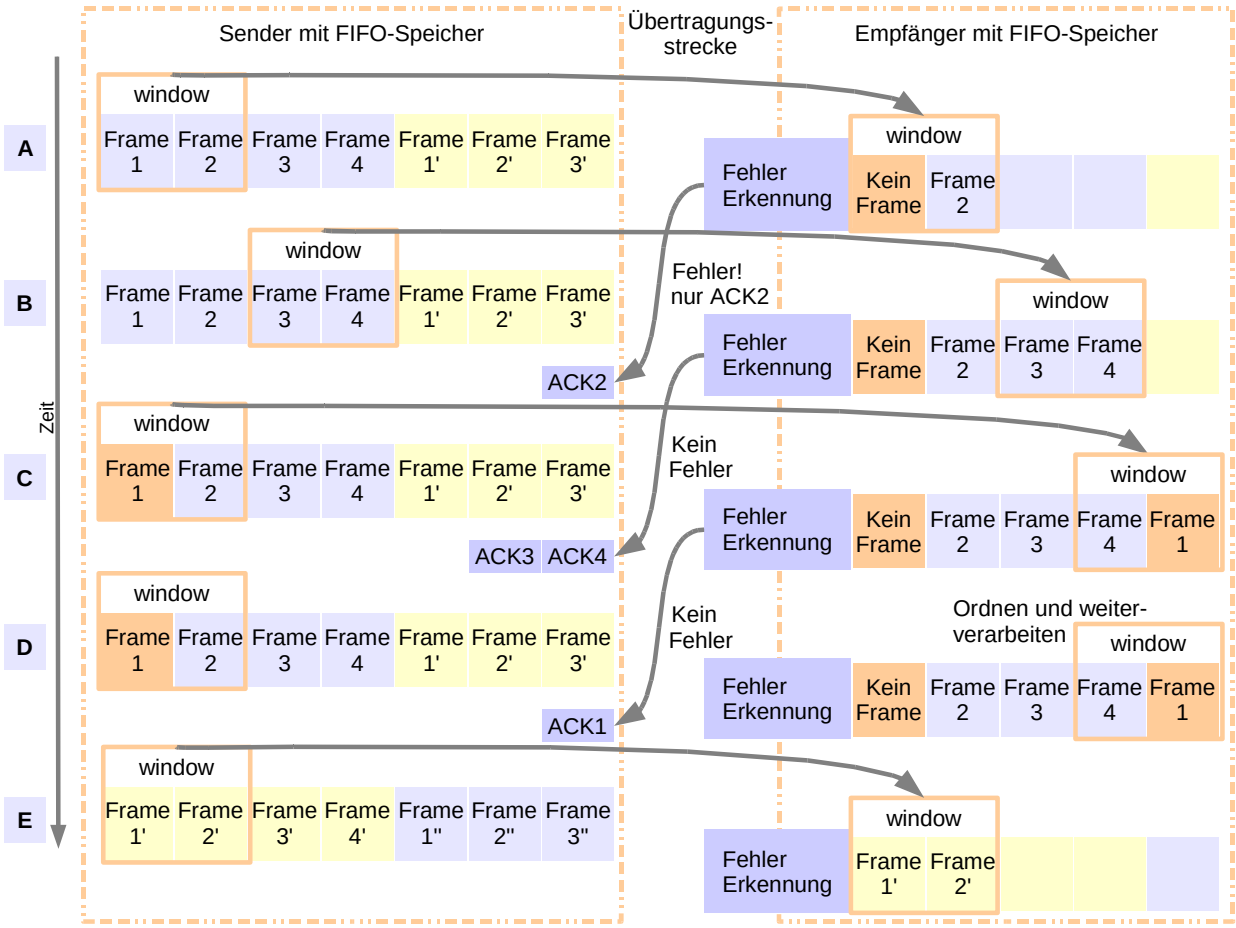



*Abbildung 10.10: Frame 1 geht mit SR verloren*

„neuen“ Frames mit 1’ bis 4’ gekennzeichnet. Es sind aber trotzdem nur vier Framenummern nötig bei einem window-size von 2.)

Abbildung 10.11, ein ACK geht verloren:

1. Zum Zeitpunkt (A) sendet der Sender die Frames 1 und 2. Die Frames werden fehlerfrei übertragen. Das ACK geht jedoch verloren.
2. Zum Zeitpunkt (B) übermittelt der Sender ohne Unterbruch die Frames 3 und 4. Der Sender erhält jetzt aber weder das ACK1 noch das ACK2 der Fehlererkennung des Empfängers. Der Sender kann nicht weiter senden und wartet.
3. Zum Zeitpunkt (C) erhält der Empfänger die ACK3 und ACK4 und ist überzeugt, dass er die Frames 1 und 2 wiederholt senden muss. Er schickt die Frames 1 und 2 noch einmal. Der Empfänger wartet auf die neuen Frames 1 und 2.
4. Zum Zeitpunkt (D) übermittelt der Sender die Frames 1 und 2 noch einmal. Da jetzt die Frames 1 und 2 beim Empfänger doppelt vorhanden sind, sendet dieser noch einmal ein ACK1 und ACK2 und löscht die doppelten Frames.

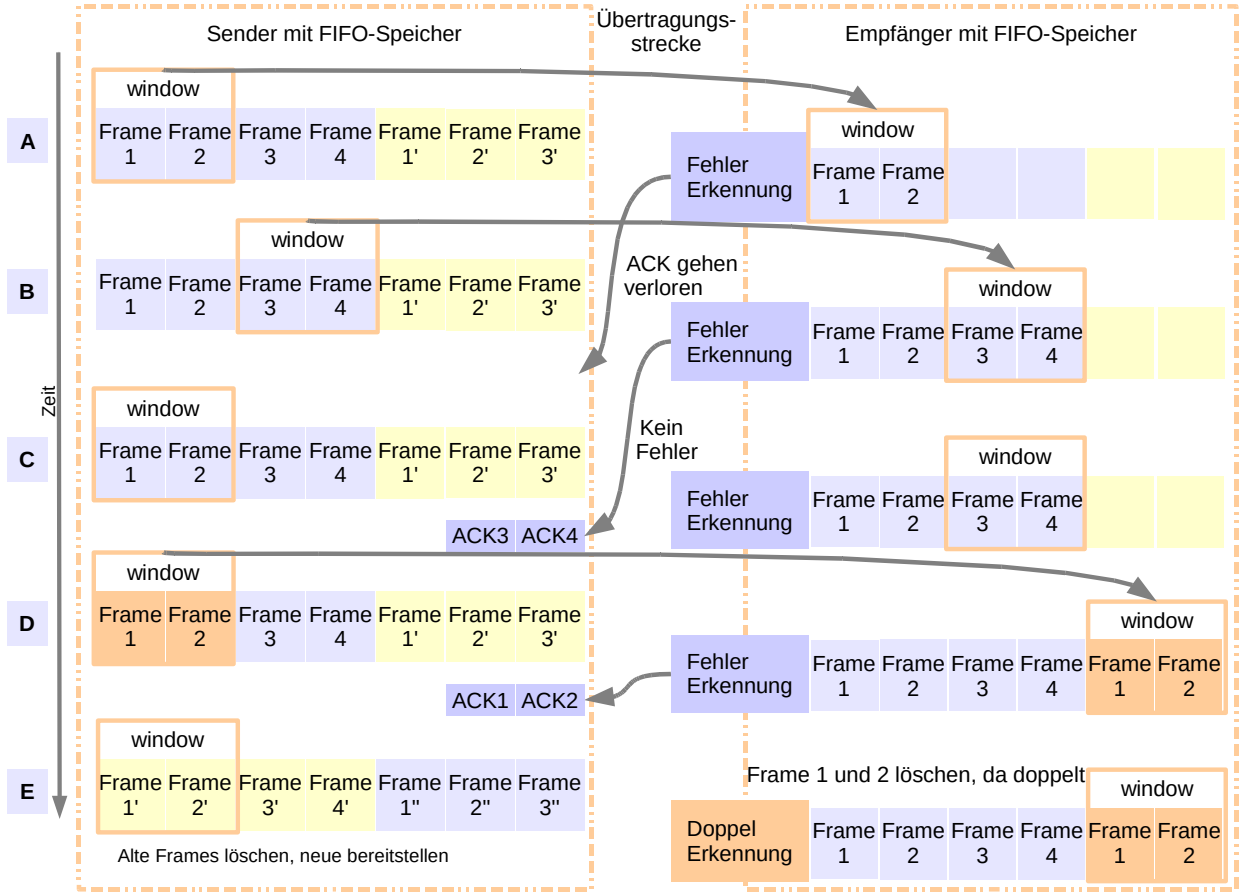



*Abbildung 10.11: Ein ACK geht verloren*

5. Zur Zeit (E) erhält der Sender endlich noch die Bestätigungen ACK1 und ACK2. Er kann nun die alten Frames 1, 2, 3 und 4 aus seinem Speicher löschen und die nächsten Frames 1', 2', 3' und 4' zum Senden bereitstellen. Der Empfänger gibt die alten Frames 1, 2, 3 und 4 zur weiteren Verarbeitung frei und bekommt somit Platz für die neuen Frames.

Aus diesen Beispielen ist ersichtlich, weshalb hier 2*n Sequenznummern benötigt werden!

Auch bei diesem Verfahren kann das NAK (not acknowledged) eingesetzt werden.

### 10.2.4.2 Go back N

Abbildung 10.12, alle Frames werden korrekt übermittelt:

1. Zum Zeitpunkt (A) übermittelt der Sender das Frame 1. Weil es richtig ankommt, sendet der Empfänger ein ACK1.

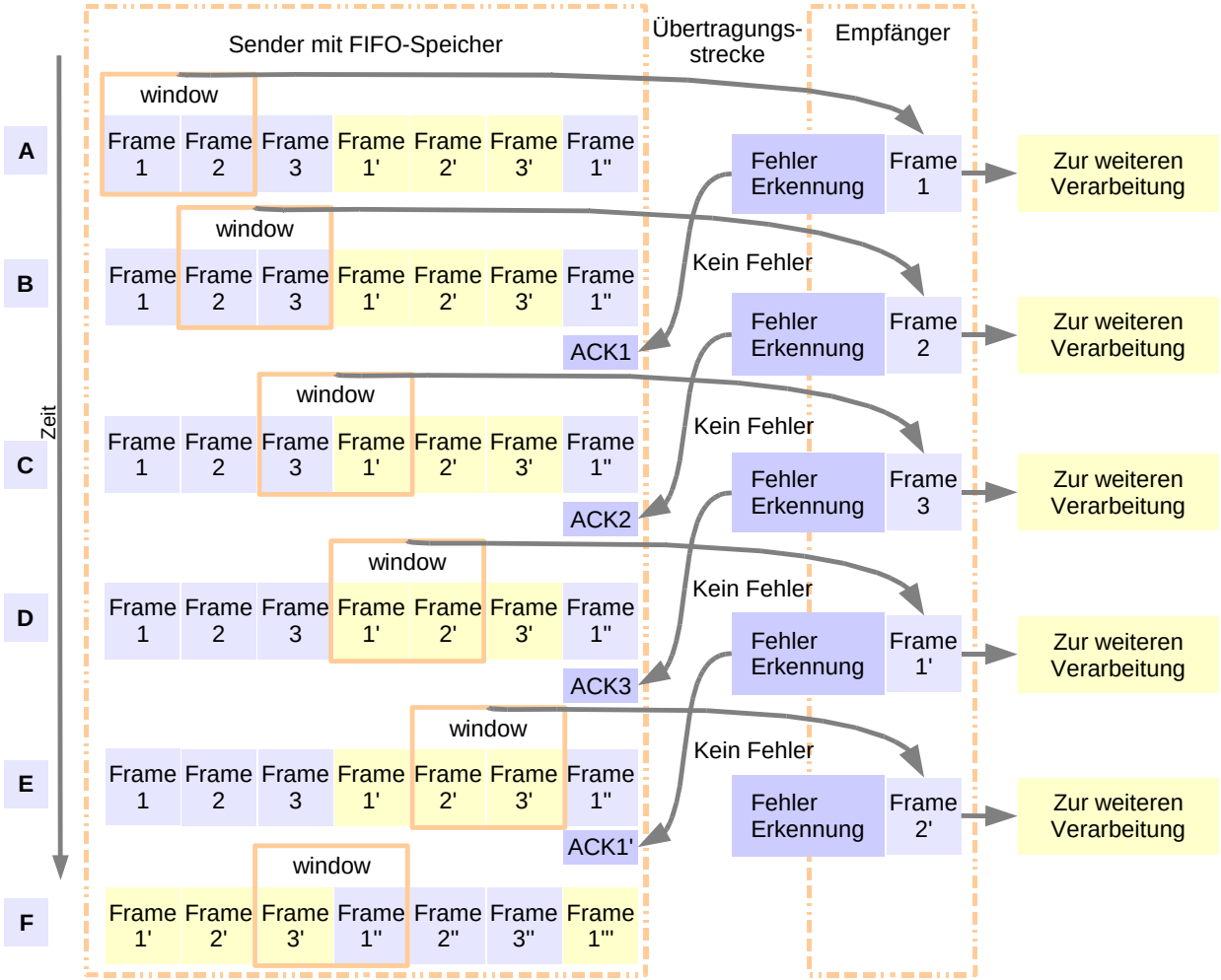



*Abbildung 10.12: Korrekt übermittelte Frames mit GBN*

2. Zum Zeitpunkt (B) übermittelt der Sender das Frame 2. Nach dem Senden von Frame 2 kommt die Bestätigung von Frame 1, ACK1, beim Sender an. Er weiss also, dass das Frame 1 in Ordnung ist. Er behält alle Frames in seinem Speicher. Der Empfänger hingegen gibt das Frame 1 aus seinem Speicher frei.
3. Zum Zeitpunkt (C) wird Frame 3 gesendet, Frame 2 aus dem Speicher des Empfängers entfernt und ACK2 beim Sender empfangen.
4. Dieses Prozedere wiederholt sich von nun an, bis der Sender alle Frames gesendet hat.

Abbildung 10.13, Frame 2 geht verloren:

1. Zum Zeitpunkt (A) übermittelt der Sender das Frame 1. Weil es richtig ankommt, schickt der Empfänger ein ACK1.
2. Zum Zeitpunkt (B) übermittelt der Sender das Frame 2. Das Frame geht aber unterwegs verloren. Der Empfänger behält das Frame 1 in seinem Speicher. Nach dem Senden von Frame 2 kommt die Bestätigung von Frame 1, ACK1, beim Sender an. Er weiss

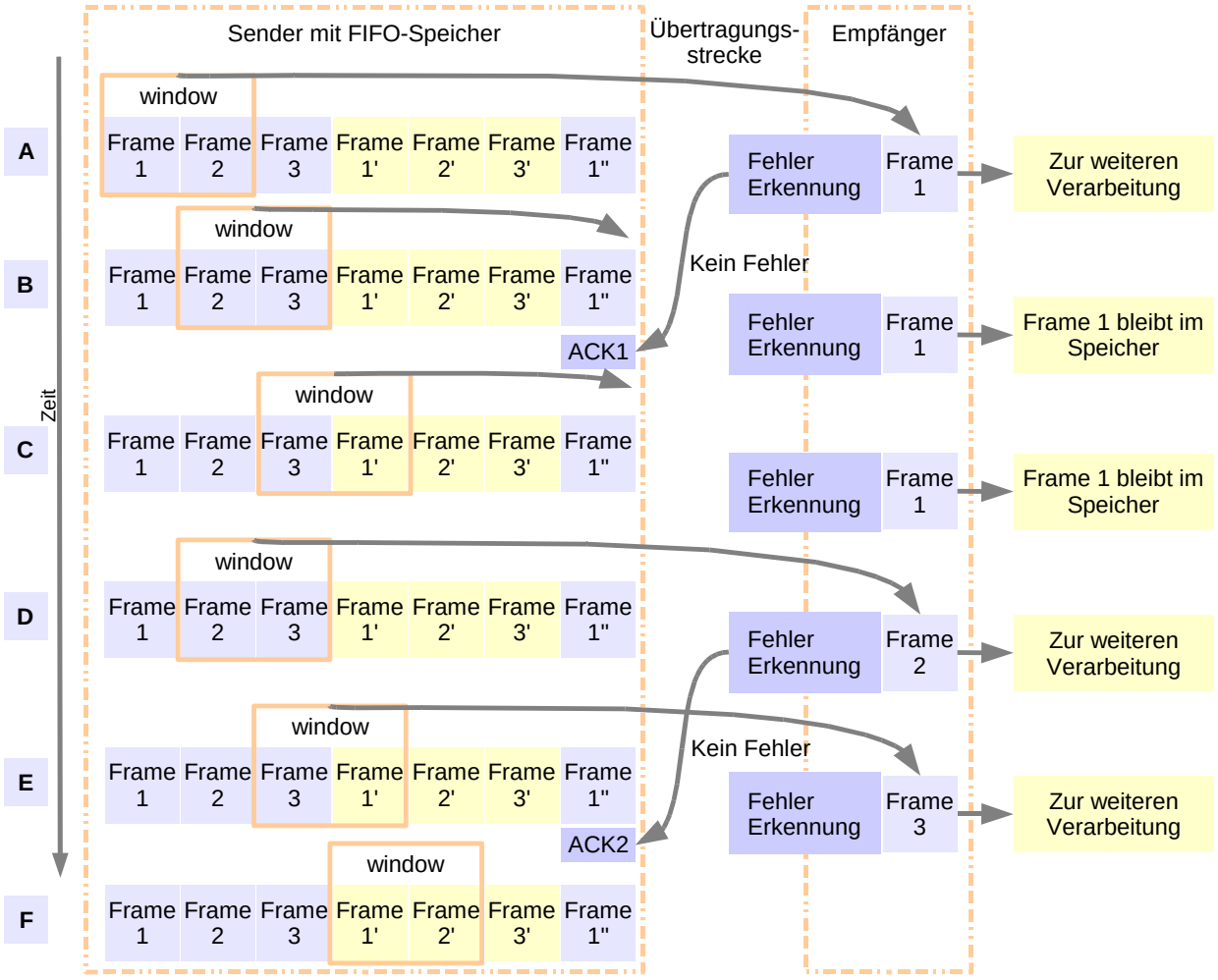



*Abbildung 10.13: Frame 2 (und 3) werden nicht richtig übertragen. Daher Go Back zu Nummer 2 in Schritt D.*

also, dass das Frame 1 in Ordnung ist. Er behält alle Frames in seinem Speicher.

3. Zum Zeitpunkt (C) wird Frame 3 gesendet. Das Frame wird vom Empfänger nicht angenommen, da er ja Frame 2 erwartet. Frame 1 bleibt im Speicher. Da der Sender kein ACK2 erhält, merkt er, dass beim Frame 2 etwas schief gelaufen sein muss. Er sendet nun noch einmal alle Frames ab Frame 2 neu. Er geht also zurück, bis zum falsch übermittelten Frame und sendet noch einmal alle Frames von dort an.
4. Zum Zeitpunkt (D) übermittelt der Sender somit das Frame 2, welches der Empfänger bestätigt.

Jetzt kann die Übertragung wieder normal weiter ablaufen.

Es dürfte nicht schwer fallen, zu sehen, dass das Verfahren auch bei fehlerhaften Bestätigungen (fehlende ACK) funktioniert.

Das Beispiel zeigt auch, dass hier n+1 Sequenznummern notwendig sind, um die Frames eindeutig zu unterscheiden.

### 10.2.5 Übertragungssicherheit – Verbindungsprotokolle

Damit wir gleich einen ersten Eindruck über die verschiedenen Protokolle erhalten, können wir folgende Zusammenfassung studieren: Verbindungsprotokolle (engl. link-protocols) werden benötigt, um die Verbindung zwischen den Kommunikationsteilnehmern aufzubauen und zu sichern. Wir unterscheiden zwischen den zeichenorientierten und bitorientierten Protokollen.

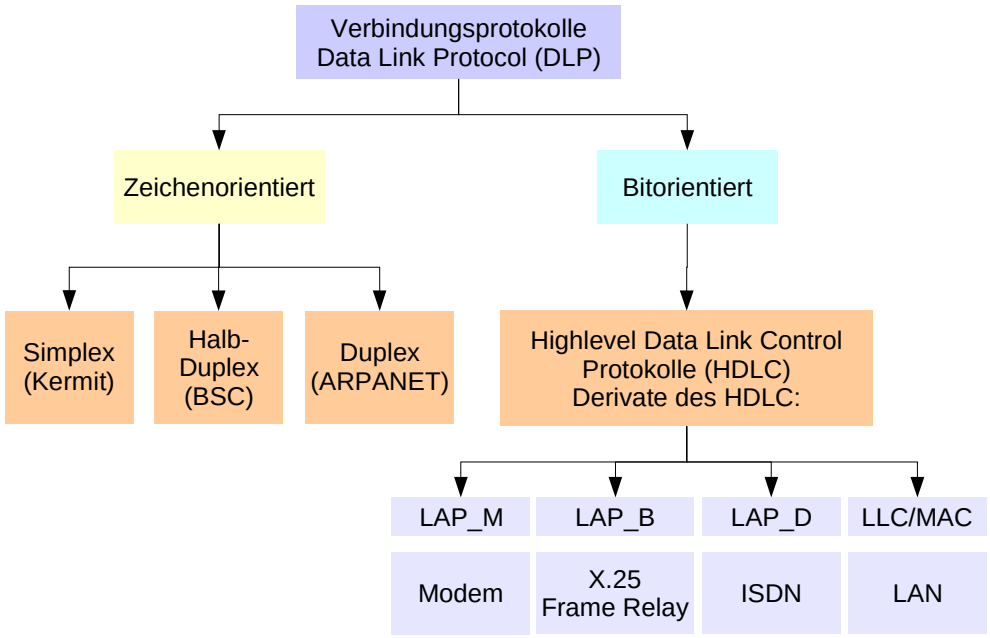



*Abbildung 10.14: Übersicht über die Verbindungsprotokolle*

Im Prinzip existiert bei den bitorientierten Protokollen das HDLC (High Level Data Link Control Protokoll) als wichtigstes dieser Protokolle. Die Frames des HDLC sind alle ähnlich (nach einer Norm) aufgebaut. Die verschiedenen Abarten (Derivate) des HDLC unterscheiden sich daher vor allem in den Link Access Proceduren (LAP_x). Je nach Anwendung (Modem, X.25, ISDN, LAN) ist die Art und Weise, wie die Links zu Stande kommen, unterschiedlich.

#### 10.2.5.1 Zeichenorientierte Protokolle

Die drei wichtigsten Arten der zeichenorientierten Protokolle werden hier erläutert. Oft werden diese Protokolle „totgesagt“, doch immer wieder erscheinen sie in der Technik.

##### 10.2.5.1.1 Simplex-Protokoll (KERMIT)

Das Protokoll „Kermit“ baut eine Punkt-Punkt-Verbindung auf, wobei der eine Computer als Sender und der zweite als Empfänger konfiguriert sein muss (Simplex).
Die beiden Computer werden Data Terminal Equipment (DTE) genannt. Das Protokoll, das zwischen den beiden DTE aufgebaut wird, heisst Data Link Protocol (DLP). Die physische Verbindung kann mittels Null-Modem-Kabel (seriell), Modemverbindungen, Parallelkabel, Infrarot, Laser, Funk, USB (Universal Serial Bus) erstellt werden.

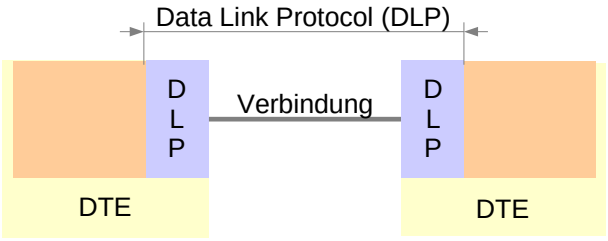



*Abbildung 10.15: Verbindung zweier DTE*

Das Kermit-Protokoll kommt bei Übertragungen zwischen Kleincomputern, Telefon-Endgeräten und allenfalls Taschenrechnern zum Einsatz. Ein ähnliches Protokoll ist das X-Modem-Protokoll, das in einigen Varianten auch Halbduplex-Verbindungen ermöglicht.
Es können auch Modems verwendet werden. Dabei muss das Modem des Empfängers in den Empfangsmodus gesetzt werden, sonst kann das Modem die Anrufe des Senders nicht annehmen.
Das Modem wird mit Data Cirquit Termination Equipment oder manchmal mit Data Communication Equipment (DCE) bezeichnet. Die zwei Modems sind hier über ein öffentliches Telefonnetz (Public Switched Telefone Network, PSTN) verbunden.
Zur Datenübertragung muss das Kermit-Programm auf beiden Computern gestartet werden. Der Sender startet die Verbindung mit dem

Befehl „Connect“. Der Sender kann anschliessend Dateien übermitteln, indem er den Befehl „Send“, gefolgt von den Dateinamen eingibt. Die Dateien werden gesamthaft übertragen. Teile von Dateien können nicht übertragen werden. Sind die Dateien übermittelt, wird die Verbindung vom Sender mit „Exit“ abgebrochen.

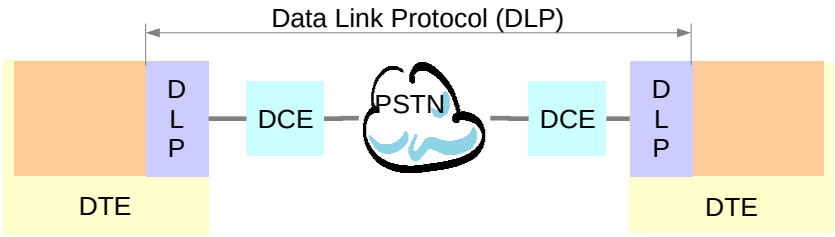



*Abbildung 10.16: DLP am Beispiel einer Modemverbindung*

#### 10.2.5.1.2 Das Polling

**Praxis-Hinweis:**
Polling braucht immer eine zentrale (Steuer-)Station. Eingesetzt wird es nicht nur in Host-Umgebungen, sondern auch vermehrt in Kommunikations-Netzen und Client/Server-Umgebungen.

An dieser Stelle muss ein wichtiger Begriff eingeführt werden, der sowohl bei den zeichenorientierten Protokollen wie auch bei den bitorientierten zum Einsatz kommt. Damit überhaupt eine Punkt-Punkt-Kommunikation zwischen einer zentralen Station (Host oder Primary Station) und anderen am Netz angeschlossene Stationen (Slaves oder Secondary Stations) aufgebaut werden kann, muss die Primary Station die Secondary Station zuerst zur Kommunikation auffordern. Dieses Auffordern oder Anfragen heisst im englischen Sprachgebrauch „polling“.
In der Telematik ist dieser Begriff eng mit den Protokollen der Schicht 2 verbunden. In dieser Schicht werden die Punkt-Punkt-Verbindungen zwischen Host und Slave aufgebaut. Irgendwie müssen die Slaves ja „merken“, wann sie Nachrichten an die Gegenstelle senden dürfen und wann nicht. Polling kann auch in den Schichten 3, 4 und 7 vorkommen, da auch diese Schichten Verbindungen aufbauen können zwischen den Endgeräten (Schicht 3/4) respektive den Anwendungsprotokollen der Clients in Schicht 7.

#### 10.2.5.1.3 Halb-Duplex-Protokoll (Binary Synchronous Control, BSC)

Dieses Protokoll wurde für die Kommunikation zwischen Terminals und ihren Hosts entwickelt.
Das von IBM entwickelte BISYNC-Protokoll (Binary Synchronous Communication, Binäre Synchronkommunikation) ist in der Industrie verbreitet. Es ist für Leitungen gedacht, die im Halbduplex-Modus arbeiten, sowohl für Gruppen- als auch Direktverbindungen. Es werden zwei verschiedene Netz-Topologien eingesetzt: Das Multipoint-Netz und das Multidrop-Bus-Netz.

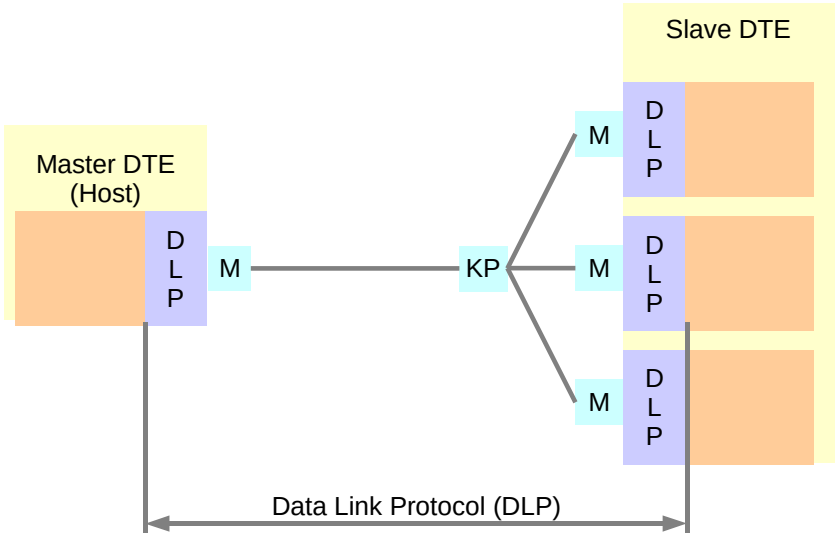



*Abbildung 10.17: Das Multipoint-Netzwerk*

Beim Multipoint-Netz überwacht ein Host (Master DTE) die Slave DTE. Der Host und die Slaves sind je über ein Vierdraht-Modem (M) mit dem Knotenpunkt (KP) verbunden.
Beim Multidrop-Bus-Netz werden die Slaves mit dem Host über je ein Line Driver/Receiver (LD/R) mit Twisted Pair-Kabel verbunden.
Diese Art Netze kommt beispielsweise in Warenhäusern vor, wobei ein Zentralcomputer die Registrier-Kassen-Computer (POS, Point Of Sale) überwacht und die Daten abruft. Diese Netze wurden früher mit Idle request (IRQ)/BSC-Protokollen betrieben. Heute wird ein HDLC-basiertes NRM-Protokoll (Normal Response Mode) eingesetzt.

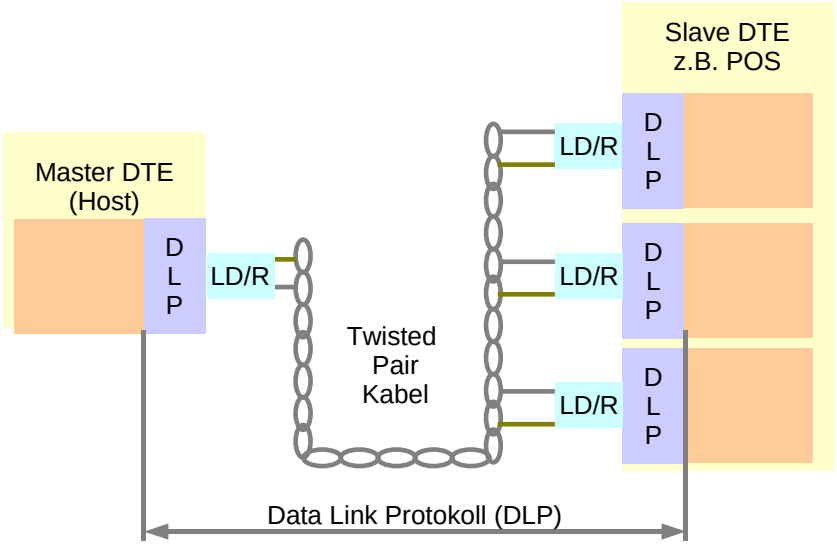



*Abbildung 10.18: Das Multidrop-Netzwerk*

#### 10.2.5.1.4 Duplex-Protokoll (IMP/IMP-Verbindung im ARPANET)

In den Anfängen des ARPANET (Vorläufer des Internet) wurden Interface Message Processors (IMP) zu einem Netz verbunden. Jeder IMP sollte mit mindestens zwei weiteren IMPs verbunden sein, um eine möglichst hohe Sicherheit gegen Ausfälle zu erhalten. Ein Host konnte Nachrichten mit bis zu 8064 Bit (1008 Zeichen) an einen IMP senden und dieser beförderte die Nachricht in Paketen von 1008 Bit (126 Zeichen) mit speziellen Duplex-Protokollen (IMP/IMP-Protokollen).

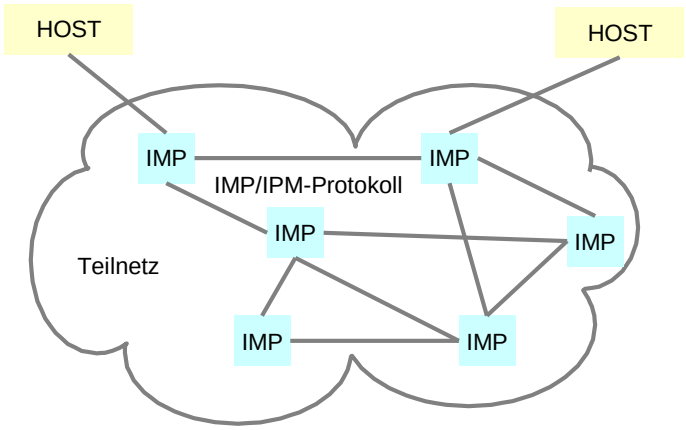



*Abbildung 10.19: Das IMP/IMP-Protokoll beim ARPANET*

Die IMP/IMP-Protokolle benutzen Continuous Request Control-Protokolle (CRQ) und werden heute noch sowohl für Verbindungen auf der Erde als auch für Verbindungen zu Satelliten angewendet. Der Aufbau und die Funktionsweise der zeichenorientierten Duplex-Protokolle ist relativ kompliziert. Eine genaue Beschreibung ist in der Literatur unter den Stichworten ARPANET, IMP to IMP oder DLP zu finden.

### 10.2.6 Bit orientierte Protokolle

HDLC ist ein internationaler Standard, der für point to point- (Punkt-zu-Punkt) und Multipoint-Verbindungen entwickelt wurde. IBM's Synchronous Data Link Control Protokoll (SDLC), der Vorläufer des HDLC, und das Advanced Data Communications Control Procedure (ADCCP) des American National Standards Institute (ANSI) werden ebenfalls noch eingesetzt. HDLC ist die Grundlage für verschiedene Data Link Control-Protokolle. Frameformat siehe Abbildung 10.1.

#### 10.2.6.1 Einige Derivate des HDLC

Der HDLC-Frame ist genormt. Die verschiedenen Protokolle, die auf diesem Frameformat beruhen, unterscheiden sich aber in der Art des Verbindungszuganges (Link Access). Der Hauptunterschied beruht darauf, ob die Verbindung durch das dazwischen liegende Netz be-

einflusst wird oder ob das Netz von der Verbindung nicht wahrgenommen wird (transparente Verbindung).

#### 10.2.6.1.1 Link Access Procedure Balanced (LAP-B)

Das LAP-B-Protokoll kontrolliert Frames in Punkt-zu-Punkt-Datennetzen. Ein gutes Beispiel dafür ist das X.25-Paketvermittlungsnetz.
Das Protokoll kontrolliert den Verkehr zwischen einem Computer (DTE, Data Termial Equipment) und einer Datenvermittlungsstelle (PSE, Packet Switching Exchange oder DSE, Data Switching Exchange).

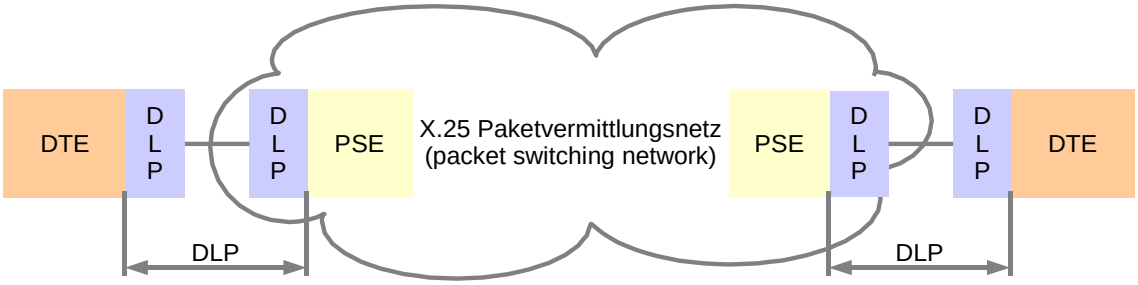



*Abbildung 10.20: Der Link Access bei LAP-B*

Weil LAP-B ein Unterprotokoll von HDLC ist, gelangen auch hier die gleichen Überwachungsframes wie bei den anderen Unterprotokollen zur Anwendung.
Zu beachten ist, dass alle Netze, die LAP-B einsetzten, dem DTE des Kunden nur einen Network Exchange-Zugang (hier das beschriebene PSE) ermöglichen. Dies hat zur Folge, dass das DTE des Kunden keinen Einfluss hat auf den genauen Transportweg seiner Daten oder auf die Funktionen und Dienste auf dem Netz. Man spricht hier von nicht transparenten Netz-Verbindungen.

#### 10.2.6.1.2 Link Access Procedure für Modems (LAP-M)

Dies ist ein Protokoll, das z.B. in der Norm V.32 implementiert ist. Modems nach dieser Norm akzeptieren asynchrone Start/Stop-Daten eines fremden DTE und transportieren die Nutzdaten in Frames mit synchroner Übermittlung und einem Fehlerprotokoll auf HDLC-Basis. Abbildung 10.21 zeigt das Prinzipschema solcher Modems.

**Praxis-Hinweis:**
Die LAP-M-Netzwerkverbindung ist, im Gegensatz zu den Verbindungen mit LAP-B-Protokollen, transparent. Das bedeutet, dass das DTE des Senders mit dem DTE des Empfängers so verbunden ist, dass der Netzbetreiber (hier das PSTN) keinen Einfluss hat auf die Übertragungsart zwischen den Modems. Der Netzbetreiber kann höchstens die Bandbreite der Übertragung einschränken.

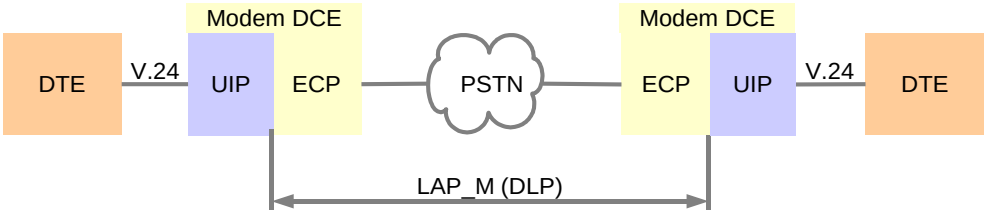



*Abbildung 10.21: Der Link Access bei LAP-M*

Das DTE ist über eine V.24-Schnittstelle mit dem Modem verbunden. Das Modem besteht aus zwei Teilen: einer Benutzerschnittstelle (UIP, User Interface Part) und einem Fehlerkorrekturteil (ECP, Error Correcting Part). Die Modems sind wiederum über das öffentliche Telefonnetz (PSTN) verbunden.

#### 10.2.6.1.3 Link Access Procedure für D-Channel ISDN (LAP-D)

Dieses ebenfalls nicht transparente Protokoll ist auf dem D-Kanal des ISDN implementiert und überwacht den Verbindungsaufbau einer ISDN-Verbindung.

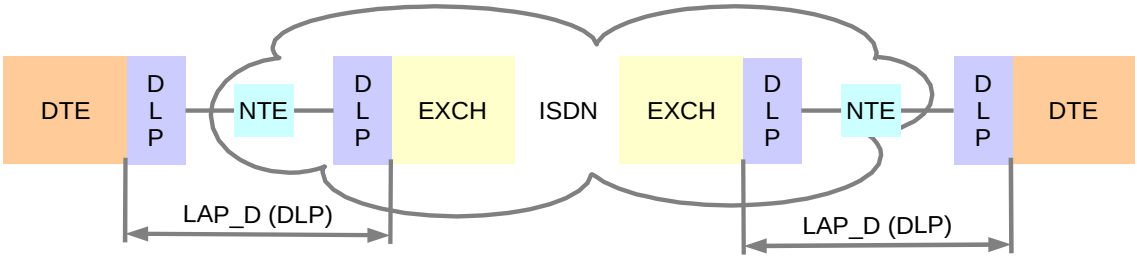



*Abbildung 10.22: Der Link Access bei LAP-D*

Das DTE wird hier über ein Endgerät (NTE, Network Termination Equipment) zur Vermittlungsstelle (EXCH, Switching Exchange) geschaltet. Die Daten werden über das ISDN-Netz übertragen und über EXCH und NTE zum Empfänger geleitet.

Damit die neuen Funktionen von ISDN (z.B. gleichzeitig zwei nutzbare Kanäle B1 und B2 beim Basisanschluss oder 30 Kanäle, B1 bis B30 beim Primär-Multiplexanschluss) ausgeschöpft werden können, sieht das Frame des LAP-D folgendermassen aus:

- Am Anfang steht die Marke (Flag).
- Block 2 und 3 sind Adressblöcke mit dem Service Access Point Identifier (SAPI), für die Bestimmung der Art des Endgerätes (Sprache, Daten, Sprache und Daten), dem Start/Stop-Bit (C/R), dem TEI, Terminal Endpoint Identifier (einer Endgeräteziffer) und dem erweiterten Adressbit (EA).
- Blöcke 4 und 5 sind die aus dem HDLC bekannten Steuerungsblöcke.
- Information (Daten)
- Die beiden vorletzten Blöcke sind für die Fehlerkorrektur reserviert, hier CCITT 16.
- Abschluss: Flag

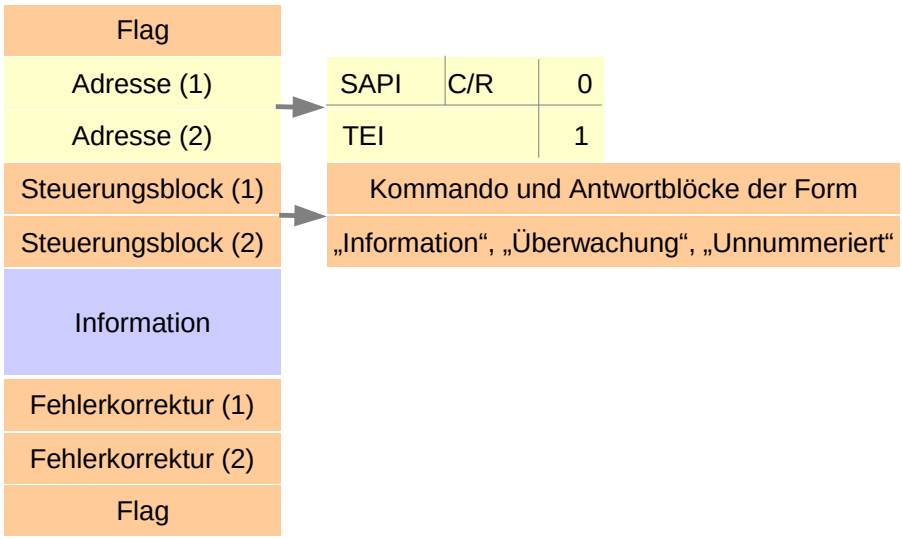



*Abbildung 10.23: Der Aufbau des HDLC-Frames bei ISDN (alle Blöcke 8 Bit lang)*

#### 10.2.6.1.4 Medium Access Control (MAC)/Logical Link Control (LLC)

MAC (Medium Access Control) und LLC (Logical Link Control) sind HDLC-Derivate für LANs und bilden eine Einheit.

Das MAC-Teilprotokoll sitzt auf der physikalischen Schicht auf (Layer 1, Bitübertragung) und beinhaltet das Zugriffsverfahren (CSMA/CD und CSMA/CA).

LLC ist nach IEEE 802.2 genormt und kann als verbindungsloser Service (unzuverlässiger Datagrammdienst), als verbindungsorientierter Service mit Verbindungsaufbau oder als Service mit bestätigtem Verbindungsaufbau (bestätigter Datagrammdienst) vorkommen.

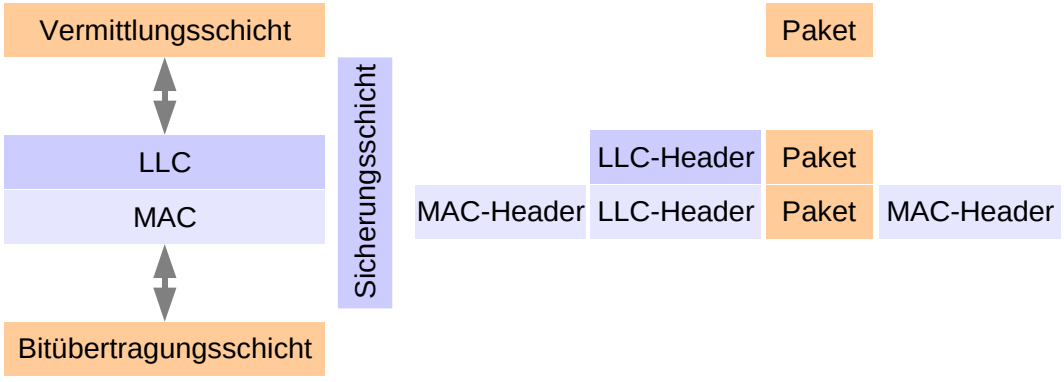



*Abbildung 10.24: Der Zusammenhang zwischen LLC und MAC*

Das LLC-Teil-Frame-Format ist wie folgt aufgebaut:

| Ziel-Adresse | Sender-Adresse | Überwachung | Information |
|---|---|---|---|
| 8-Bit | 8-Bit | 8-Bit | n x 8 Bit |

*Abbildung 10.25: Das LLC-Frame*

Start mit der Zieladresse, gefolgt von der Senderadresse (je acht Bit). In den anschliessenden drei Bytes befindet sich die Überwachung mit Überwachungsfunktionen und Polling-Kontrolle (siehe HDLC). Daran anschliessend folgen die Datenblöcke.
Zu bemerken ist noch, dass die Ziel- und Senderadresse des LLC nur zwischen den LLC des Senders und des Empfängers gelten und im Netzwerk keine Funktion haben (nicht verwechseln mit der Ziel- und Quellenadresse der MAC-Teil-Schicht). Netzwerkadressen und Fehlererkennung ist Sache der MAC-Teilschicht. Dies ist der Grund, weshalb die MAC- und LLC-Teilschichten im ISO/OSI-Modell eine Einheit bilden. Der gesamte LLC-Teil-Frame wird im Nutzdatenfeld der MAC-Schicht eingefügt.
Das Frameformat der MAC-Teilschicht ist für jedes Zugriffssteuerverfahren leicht unterschiedlich.
Für IEEE 802.3 sieht das Format wie folgt aus:

| Präambel | Rahmenbeginn | Zieladresse | Quelladresse | Länge des Datenfeldes | L | L | C | Info. | Pad | CRC |
|---|---|---|---|---|---|---|---|---|---|---|
| 7 Byte | 1B | 2 / 6 Byte | 2 / 6 Byte | 2B | Daten: 0 bis 1500 Byte | | | | 0 bis 46 B | 4 B |

*Abbildung 10.26: Das MAC-Frame mit integriertem LLC-Frame*

- Die Präambel (Vorspann) besteht aus sieben Byte, die alle die Bitfolge 10101010 haben. Die Manchester-Codierung dieser Bitfolge erzeugt für die Dauer von 5,6 µs eine 10-MHz-Schwingung, woran sich der Taktgeber des Empfängers synchronisieren kann.
- Das Rahmenstart-Byte hat die Bitfolge 10101011.
- Dann folgen zwei Adressen: eine für das Ziel und eine für die Quelle. Die Norm lässt Adressen von zwei oder sechs Byte Länge zu. Das 10-MHz-Basisband benutzt sechs-Byte-Adressen. Die

Adresse, die nur aus Einsen (11111111…111) besteht, wird Broadcast-Adresse genannt. Die Rahmen, deren Ziel-Adresse aus lauter Einsen besteht, werden an alle Stationen gesandt. Die MAC-Adresse einer Netzwerkschnittstellenkarte (NIC) kann mithilfe eines Konfigurationsprogrammes des jeweiligen Karten-Herstellers in Erfahrung gebracht werden. Sie wird als Hexadezimal-Zahl angegeben und kann folgendermassen aussehen: 0000B20205ACF.

- Das Feld mit der Länge des Datenfeldes gibt an, wie gross das Datenfeld ist.
- Das Datenfelfeld kann eine Länge von 0 bis 1500 Byte aufweisen.
- Falls das Datenfeld die Länge 0 hat, kann dies im Netz zu Problemen führen, weshalb das Feld „Pad“ in diesen Fällen die Möglichkeit hat, 0 bis 46 Byte in den Rahmen einzufüllen.
- Die Prüfsumme wird im letzten Feld mitgegeben und basiert auf dem CRC-Verfahren.

### 10.2.7 Steuerung der Kommunikation nach IEEE 802

Die Steuerung der Kommunikation (Media Access Control) hängt stark vom verwendeten Netz ab:

| *Norm* | *Beschreibung* |
|---|---|
| IEEE 802.3 | Ethernet (nn Base k) mit CSMA/CD-Zugriffssteuerverfahren |
| IEEE 802.4 | Token-Passing-Zugriffssteuerverfahren mit Bus-Topologie |
| IEEE 802.5 | Token-Passing-Zugriffssteuerverfahren mit Ring-Topologie |
| IEEE 802.6 | für MAN-Netze mit DQDB- Zugriffssteuerverfahren |
| FDDI | nach ANSI mit Token-Passing und Ring-Topologie |

*Tabelle 10.2: Die IEEE 802-Normen (Auszug)*

Tabelle 10.2 beinhaltet nur eine Auswahl an Normen für Steuerungsverfahren. Je nachdem, ob ein LAN mit Kupferdrähten, LWL oder gar drahtlos betrieben wird, gelangen andere Verfahren zum Einsatz. Ebenso unterscheiden sich die Steuerungsverfahren der verschiedenen Netzarten wie LAN, WLL (Wireless Local Loop), GSM (Global System for Mobile Communication), UMTS (Universal Mobile Telecommunication System), MAN und anderen zum Teil grundsätzlich.
Grundsätzlich unterscheidet man zwei verschiedene Möglichkeiten der Steuerung zwischen den Computern in einem Netz.

- Eine Möglichkeit zur Steuerung eines LANs beruht auf der Idee, dass eine Station nur senden darf, wenn sie dazu autorisiert ist. Die Station erhält als Zeichen für die Autorisierung eine Marke, ein so genanntes Token. Nur wenn eine Station ein Token hat, kann sie senden. Ohne Token muss sie zuhören, was andere sagen. Es

hat pro Netz ein Token. Das Token wird von einer Station zur anderen weitergereicht. Will eine Station Daten senden, so behält sie das Token, sendet die Daten und gibt das Token weiter. Hat die Station nichts zu senden, so gibt sie das Token nach Ablauf einer gewissen Zeit ebenso weiter. So ist gewährleistet, dass alle drankommen. Zu diesem Verfahren der Steuerung gehört das Token Passing von IBM.

- Die zweite, weit verbreitete Möglichkeit (engl. approach) beruht darauf, dass die Stationen im Wettbewerb miteinander stehen. Jede Station sendet, wann immer sie will (nicht synchronisiert). Stossen zwei Meldungen auf dem Kabel zusammen, müssen sie zeitversetzt noch einmal gesendet werden. Dieses Verfahren nennt man CSMA/CD (Carrier Sense Multiple Access mit Collision Detection), was etwa heisst:
  Die Leitung wird dauernd abgehört. Viele können darauf gleichzeitig senden (zugreifen). Es findet eine Kollisions-Erkennung statt. Diese Art der Steuerung wird vorwiegend auf physikalischen Bus-, Stern- und Baum-Topologien gefahren (Ethernet).

Das Verfahren mit Token ist leistungsfähiger, aber leider auch teurer. Abbildung 10.27 zeigt die Leistungsfähigkeit der beiden Verfahren im Vergleich.

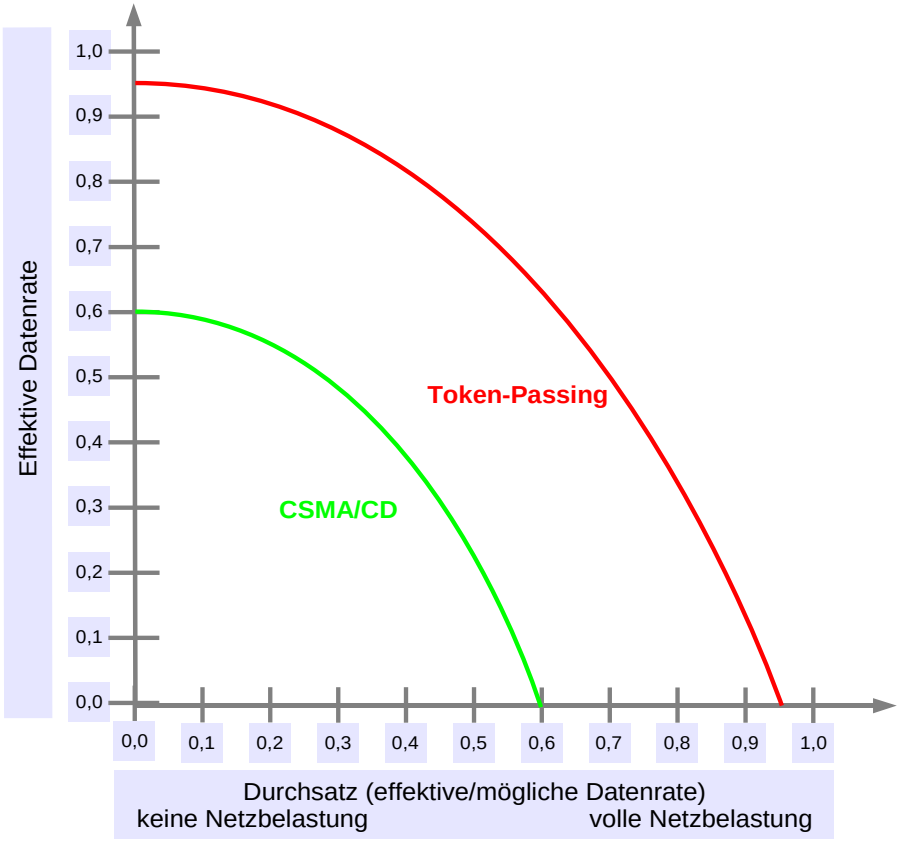



*Abbildung 10.27: Die Leistungsfähigkeit der CSMA/CD- und Token-Protokolle*

### 10.2.8 Beispiele weiterer Steuerungsverfahren

Neben dem häufig in LANs eingesetzten CSMA/CD-Verfahren werden gerade bei Glasfaserverbindungen, Funkstrecken und MAN (Metropolitan Area Networks) einige weitere Zugriffssteuerverfahren immer wichtiger:

Drahtlose LANs benutzen ein Verfahren, das Multiple Access with Collision Avoidance (CSMA/CA) heisst. Dieses Verfahren liegt dem IEEE 802.11 zu Grunde. Es beruht darauf, dass eine Senderstation einer Empfängerstation mit einem kurzen Rahmen ihre Absicht mitteilt. Die anderen, benachbarten Stationen hören diese Ankündigung und unterlassen das Senden während der Dauer der Übertragung des folgenden grossen Datenrahmens.

GSM-Netze (Global System for Mobile Communication) verwenden Frequenzen im 900-MHz-Band (890,2 bis 959,8 MHz). Das Frequenzband ist in 124 Uplink-Kanäle (Mobile an Basisstation) und 124 Downlink-Kanäle (Basisstation an Mobile) unterteilt (Frequency Division Multiplexing, FDM). Jeder der Kanäle ist mit dem Zeitmultiplex-Verfahren (Time Division Multiplexing, TDM) in acht getrennte Zeitschlitze unterteilt. Diese Aufteilung schliesst Kollisionen aus.

## 10.3 Aufbau, Betrieb und Abbau von Punkt-Punkt-Verbindungen

Die Hauptaufgabe der Schicht 2 ist der Aufbau, der Betrieb und der Abbau von Punkt-Punkt-Verbindungen im Netzwerk. Siehe dazu Abbildung 10.28.

### 10.3.1 Aufbau der Verbindung

Bevor Daten zwischen zwei Stationen ausgetauscht werden können, muss die Schicht 2 eine sichere Punkt-Punkt-Verbindung aufbauen.

Die Benutzersoftware des Senders schickt der Verbindungsschicht des Senders eine „Line.CONNECT.request“ Anfrage.

Die Verbindungsschicht des Senders pollt den Empfänger mit einem unnummerierten Frame an und legt den Betriebsmodus in einer Steueranweisung fest (z.B. SNRM).

Das Frame erreicht den Empfänger und die Steueranweisung wird ausgewertet.

Der Benutzersoftware des Empfängers wird durch ein „Line.CONNECT.indication“ ein Kommunikationswunsch des Senders angezeigt und wenn von dort keine abschlägige Antwort folgt, dann wird ein Frame mit der Steueranweisung UA (Unnumbered Acknowledge) zurückgesendet.

Sobald das Unnumbered Acknowledge Frame den Sender wieder erreicht hat, wird der Benutzersoftware die fertig aufgebaute Verbindung mit „Line.CONNECT.confirm“ bestätigt.

### 10.3.2 Betrieb der Verbindung

Die Benutzersoftware schickt den Befehl „Line.DATA.request“. Die Flusskontrolle und die Überwachung der Verbindung wird von nun an mit speziellen Überwachungs-Frames durch die Verbindungsschicht sichergestellt (wird in Abbildung 10.28 nicht gezeigt).

Wenn die Informations-Frames mit den Daten beim Empfänger ankommen, werden die Daten der Benutzersoftware mit dem Befehl „Line.DATA.indication“ angezeigt und zugestellt.

Gleichzeitig wird ein Überwachungsrahmen mit Steueranweisung (Receiver Ready) an den Sender geschickt. Dieser weiss dann, dass der Empfänger für den Empfang des nächsten Frames bereit ist.

Selbstverständlich kann nun auch der Empfänger Daten an den Sender übermitteln (die Verbindungen sind Halbduplex oder Duplex).

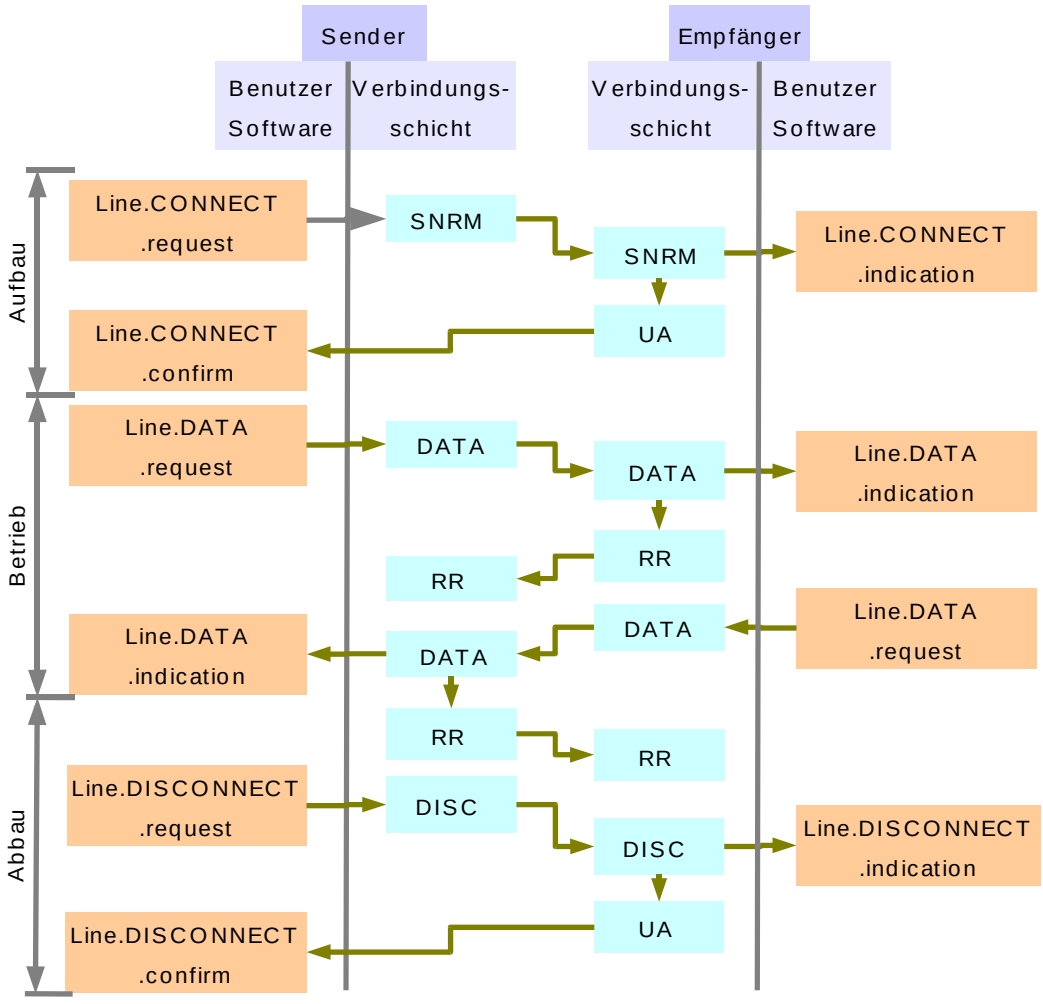



*Abbildung 10.28: Aufbau, Betrieb, Abbau einer Layer 2-Verbindung*

### 10.3.3 Abbau der Verbindung

Sobald die Daten übertragen sind, kann der Sender den Abbau der Verbindung veranlassen. Die Benutzersoftware des Senders schickt zu diesem Zweck ein „Line.DISCONNECT.request“ an die Verbindungsschicht. Diese zeigt den Abbau dem Empfänger mit der Steueranweisung DISC (Disconnect) in einem unnummerierten Frame an. Sobald die Benutzersoftware des Empfängers die Anzeige „Line.DISCONNECT.indication“ erhalten hat, wird dem Sender der korrekte Abbau mit UA (Unnumbered Acknowledge) bestätigt und die Leitung wird für andere Kommunikationen freigegeben. Die leeren Felder in Abbildung  symbolisieren die Zeit, in der die Verbindungsschicht Frames auswertet.

## 10.4 Aufgaben

1. Wo liegt der Unterschied zwischen Packets und Frames?
2. Welches sind die Hauptfunktionen des 2. Layers im ISO/OSI-Modell?
3. Wie ist es möglich, dass der Empfänger ein falsches Bit bekommt?
4. Wie kann das Weiterleiten von zwei empfangenen Frames, die identisch sind, verhindert werden?
5. Warum wird heute meist auf Paritätsprüfung verzichtet, obwohl dieses Verfahren dank Hardware-Realisierung sehr schnell ist?
6. Warum ist Idle Request relativ langsam?
7. Wo liegt der Hauptunterschied zwischen Selective Repeat und Go back N?
8. Weshalb braucht man beim Selective Repeat 2*n Sequenznummern?

Lösungen unter www.sauerlaender.ch/downloads