3 Sicherungsschicht (Data Link Layer)

3.1 Übertragungsfehler: Ursachen

3.2 Fehlererkennungs- und Fehlerkorrekturcodes

3.3 Bit Stuffing und Rahmenbegrenzer

3.4 Bestätigungen und Sequenznummern

3.5 Flußkontrolle

3.6 Beispiel: HDLC

Aufgaben der Sicherungsschicht (Schicht 2)

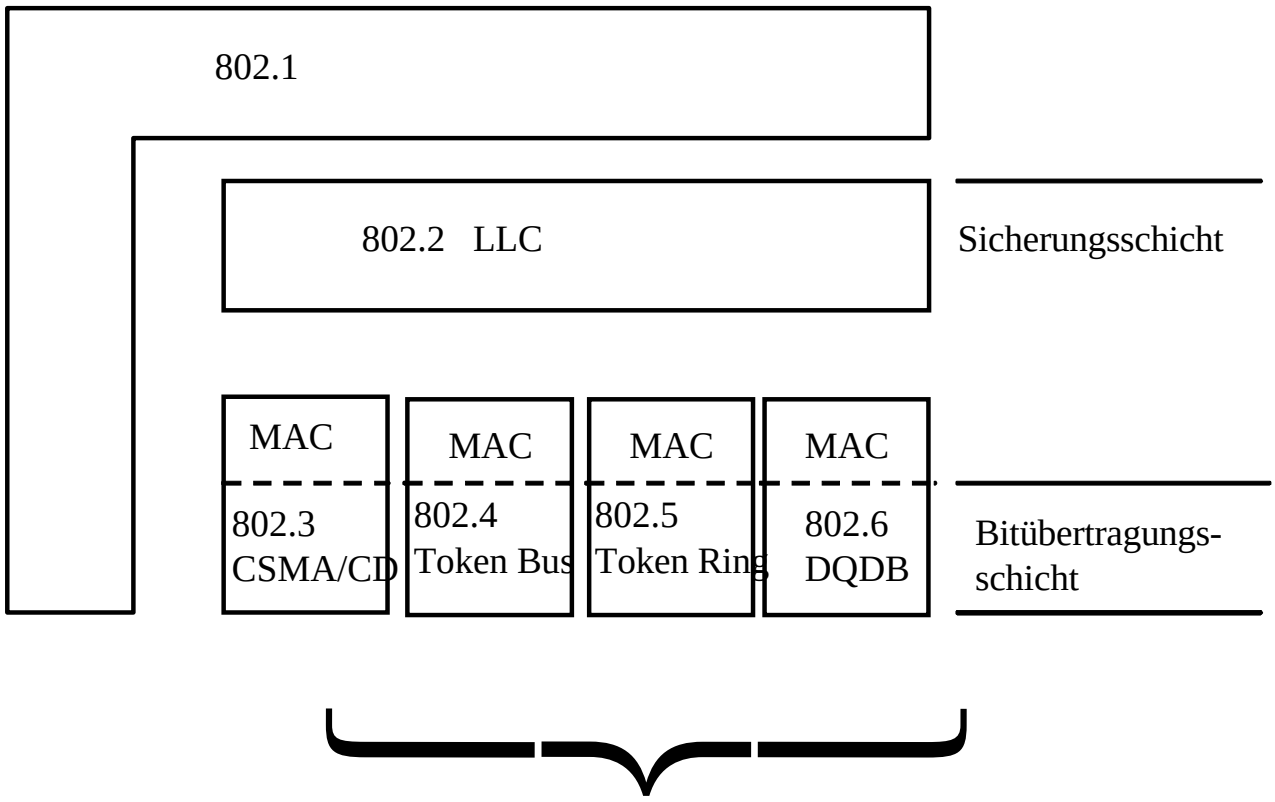
- Verdeckung von Übertragungsfehlern zwischen direkt benachbarten Partnern
  (Erkennung und Behebung)

- Flußkontrolle

- Bei LANs zusätzlich: Medienzugangskontrolle zum gemeinsamen Medium

Die Sicherungschicht im LAN (nach IEEE 802)

- Standardisierung bei IEEE und ISO



3.1 Übertragungsfehler: Ursachen

- Weißes Rauschen
- Signalverzerrung ist abhängig von der Frequenz
- Übersprechen auf Leitungen
- Impulsstörung
  - Wird durch Vermittlungseinrichtungen verursacht
  - Dauert ungefähr 10 ms (96 Bits bei 9600 Bit/s)

Viele Fehler kommen kurz aufeinanderfolgend.

Vorteil: Nur wenige Blöcke enthalten Fehler.

Nachteil: Schwer zu erkennen und zu korrigieren.

Wahrscheinlichkeit von Übertragungsfehlern

Empirisch ermittelte Fehlerrate für Telefonleitungen und Blöcke aus n Bytes:

$$10^{-4}\, n^{0.8}$$

n	p(n)
8	5.278031E-04
16	9.189586E-04
32	0.0016
64	2.785761E-03
128	4.850293E-03
256	8.444851E-03
512	1.470334E-02
1024	0.0256

3.2 Fehlererkennungs- und Fehlerkorrekturcodes

Fehler: Die empfangene Information entspricht nicht der gesendeten.

- Zur Fehlerkorrektur wird die eigentliche Information durch Kontrollinformationen (Redundanz) ergänzt.
- Je nach Umfang der Redundanz kann eine gewisse Anzahl an Fehlern erkannt oder sogar korrigiert werden.
- Der Zusammenhang zwischen dem Umfang der Redundanz, der Fehlerhäufigkeit, der Übertragungsstrekke, der Entdeckung und der eventuellen Korrektur eines Fehlers wird in der **Codierungstheorie** behandelt.

Paritätsbit

Bit 1	0	1	1	
Bit 2	1	0	1	
Bit 3	0	0	0	
Bit 4	0	1	0	
Bit 5	0	0	0	
Bit 6	0	1	0	
Bit 7	1	1	1	
	0	0	1	gerade
Paritätsbit				Parität
	1	1	0	ungerade

Querparität

- Wird zum Beispiel bei der asynchronen Übertragung verwendet.
- Die 5 bis 8 eigentlichen Informationsbits werden durch ein weiteres (redundantes) Bit innerhalb des Rahmens erweitert.
- Der Begriff stammt aus der Magnetband-Technik, als ein Zeichen gerade quer auf 6 bis 9 Spuren paßte.
- Das Redundanzbit ergänzt die Anzahl der Einsen innerhalb der Information auf eine gerade oder ungerade Anzahl.

1 0 0 1 1 1 0 0 **0**

1 0 1 1 1 1 0 0 **1**

Information ↑

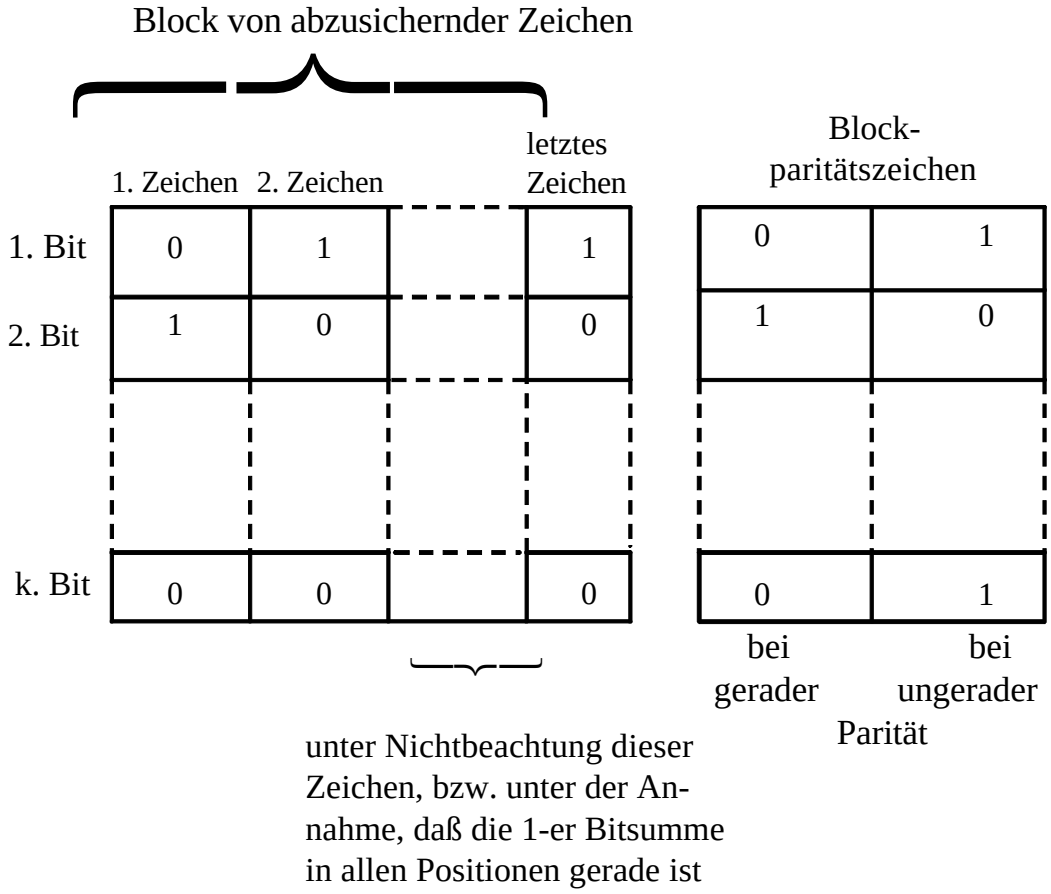
Paritätsbit

(gerade Parität)

- Es kann nur ein Fehler erkannt und keiner korrigiert werden.

Längsparität (Blockparität)

- Berechnung eines Paritätsbits jeweils am Blockende in der Längsrichtung des Magnetbandes



Hamming-Abstand

Hamming-Abstand d

Anzahl der Bitpositionen, in denen sich zwei Codewörter c_1, c_2 unterscheiden.

Beispiel: d(10**001**001, 10**110**001) = 3

(Anzahl der Bits von c_1 XOR c_2)

Hamming-Abstand D eines vollständigen Codes C

$$D(C) := \min\{d(c_1, c_2) \mid c_1, c_2 \in C, c_1 \neq c_2\}$$

Satz

Die Fähigkeit eines Codes, Fehler zu erkennen und Fehler zu beheben, hängt von seinem Hamming-Abstand ab.

Erkenne e-Bit Fehler:

ein Abstand von **e + 1** wird benötigt

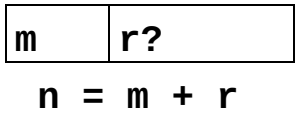
Behebe e-Bit Fehler:

ein Abstand von **2e + 1** wird benötigt

Wieviel Redundanz braucht man?

- Das Codewort bestehe aus m Zeichen-Bits.

- Wieviele Prüfbits werden benötigt, um einen 1-Bit-Fehler zu beheben?



- Es gibt 2^m legale Zeichencodes.
- Pro Codewort muß es n illegale Codewörter im Abstand von einem Bit geben.
- 2^n ist die Gesamtzahl der Codewörter.

$$(n + 1)\, 2^m \leq 2^n \Rightarrow (m + r + 1) \leq 2^r$$

Dies ergibt die untere Grenze für r.

Beispiel: $m = 7$

$(8 + r) \leq 2^r \quad \Rightarrow \quad r \geq 4$

Nachteile von fehlerbehebenden Codes

Großer Overhead (viel Redundanz) auch im Falle einer fehlerfreien Übertragung

Fehlererkennung und Sendewiederholung ist in klassischen, auf Kupferkabel basierenden Netzen effizienter!

Beispiele

Fehlererkennender Code:

Code mit einem einzigen Paritätsbit (gerade oder ungerade)

⇒ Hamming-Abstand = 2

⇒ Erkennung eines 1-Bit-Fehlers ist möglich
(oder aller Fehler mit einer ungeraden Anzahl Bits)

Fehlerbehebender Code:

Vier Codewörter:
00000 00000, 00000 11111, 11111 00000, 11111 11111

⇒ Hamming-Abstand = 5

⇒ Verbesserung von 2-Bit-Fehlern möglich

Beispiel:	00000 **00**111 **2-Bit-Fehler**	⇒	00000 11111 nächstes Code-wort

Cyclic Redundancy Check (CRC)

Grundidee

- Betrachte Bitfolgen als Darstellung eines Polynoms nur mit den Koeffizienten 0 und 1.

  Beispiel: $11001 = x^4 + x^3 + x^0$

- Wähle ein **Generatorpolynom G(x)** vom Grad g.
- Hänge an die Nachricht M(x) eine Prüfsumme derart an, daß das Polynom T(x), dargestellt durch die Nachricht mit angehängter Prüfsumme, durch G(x) teilbar ist.
- Übertrage T(x)

Algorithmus

1. Wir haben G(x) vom Grad g und hängen g 0-Bits an die Nachricht, also $x^g M(x)$
2. Teile $x^g M(x)$ durch G(x), Division modulo 2
3. Subtrahiere den Rest von $x^g M(x)$, modulo 2
4. Das Ergebnis ist T(x), die Nachricht samt Prüfsumme für die Übertragung

Beispiel für CRC-Berechnung

```
Rahmen:     1 1 0 1 0 1 1 0 1 1
Generator: 1 0 0 1 1
Nachricht nach dem Anhängen von 4 Nullbits:
1 1 0 1 0 1 1 0 1 1 0 0 0 0
1 1 0 1 0 1 1 0 1 1 0 0 0 0 : 1 0 0 1 1 = 1 1 0 0 0 0
1 0 1 0
1 0 0 1 1
  1 0 0 1 1
  1 0 0 1 1
    0 0 0 0 1
    0 0 0 0 0
      0 0 0 1 0
      0 0 0 0 0
        0 0 1 0 1
        0 0 0 0 0
          0 1 0 1 1
          0 0 0 0 0
            1 0 1 1 0
            1 0 0 1 1
              0 1 0 1 0
              0 0 0 0 0
                1 0 1 0 0
                1 0 0 1 1
                  0 1 1 1 0
                  0 0 0 0 0
           Rest:    1 1 1 0
```

(Division modulo 2: Kein Übertrag bei der Subtraktion)

Generatorpolynome in internationalen Standards

- CRC-12 = $x^{12} + x^{11} + x^3 + x^2 + x + 1$

  Wird benutzt für 6-Bit-Zeichencodes

- CRC-16 = $x^{16} + x^{15} + x^2 + 1$ (ISO)

  CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$ (CCITT)

  Beide werden für 8-Bit-Zeichencodes benutzt

Fehlererkennung mit CRC-16, CRC-CCITT (16 Bits)
(Ergebnisse aus der Codierungstheorie)

- alle einfachen und zweifachen Bitfehler
- alle Fehler mit einer ungeraden Bitzahl
- alle stoßweisen Fehler mit einer Länge • 16
- 99,998% aller längeren stoßweisen Fehler

Implementierung des CRC
In Hardware (z.B. im HDLC-Chip) mit Hilfe eines Schieberegisters und der bitweisen XOR-Funktion.

3.3 Bit Stuffing und Rahmenbegrenzer

Zur Anwendung von fehlererkennenden und fehlerkorrigierenden Codes muß der Datenstom in einzelne Rahmen unterteilt werden.
Problem: Wie kann man solche Rahmen im Bitstrom begrenzen, wenn jedes beliebige Bitmuster in den Nutzdaten vorkommen kann?

Lösung: **Bit Stuffing (Bitstopfen)**
Als Begrenzer wählt man 01111110. Der Sender fügt nach fünf Einsen im Nutzdatenstrom IMMER eine 0 ein.

Sender	0	1	1	1	1	1		1	0	1	0	1	1	1	1	1		0	1	0	1
Leitung	0	1	1	1	1	1	0	1	0	1	0	1	1	1	1	1	0	0	1	0	1
Emp-fänger	0	1	1	1	1	1		1	0	1	0	1	1	1	1	1		0	1	0	1

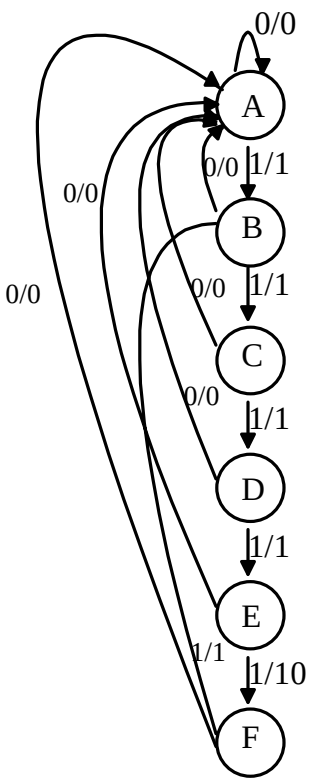
Wenn der Empfänger nach fünf Einsen eine Null sieht, entfernt er diese aus dem Datenstrom.

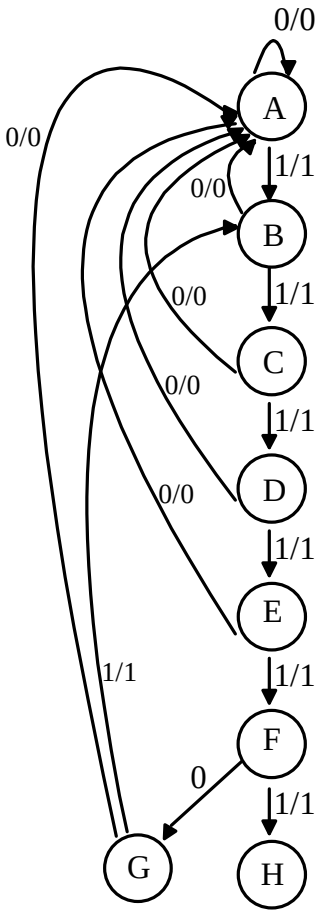
Bitstopfen

Endliche Automaten für Sender (Quelle) und Empfänger (Ziel) bei einer Übertragung mit Bitstopfen.

Sender
(Ereignis: zu übermittelndes Bit)

Empfänger
(Ereignis: eintreffendes Bit)





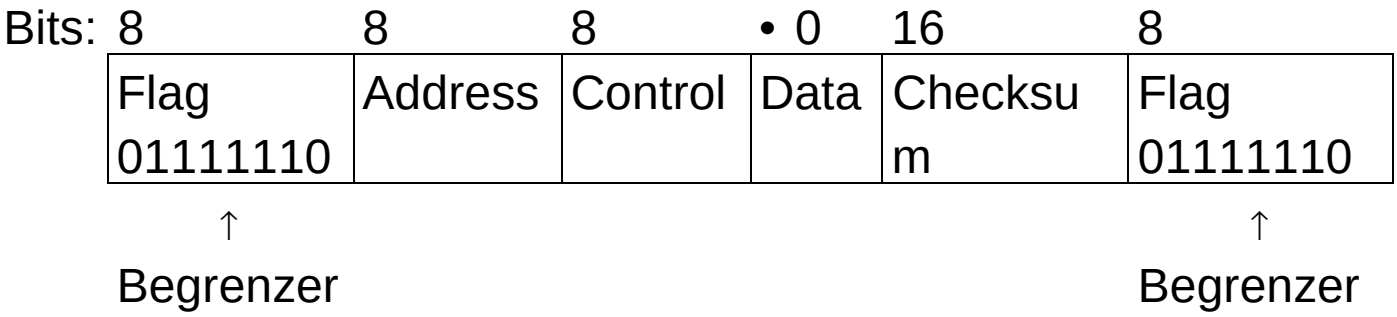
Notation: e/a
e = Ereignis
a = Ausgabe

eingefügtes 0-Bit entfernt

Begrenzer (Fehler)

Rahmenformat der Schicht 2

Rahmen ("Paket") in der Schicht 2



3.4 Bestätigungen (Acknowledgements) und Sequenznummern

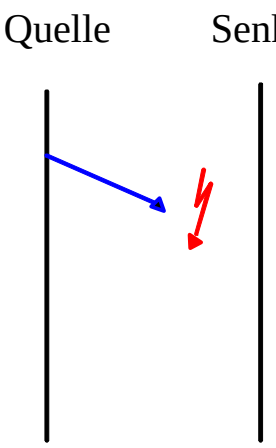
Acknowledgement
= Anerkennung, Bestätigung, Quittung

Bestätigungen und Sequenznummern werden benutzt für

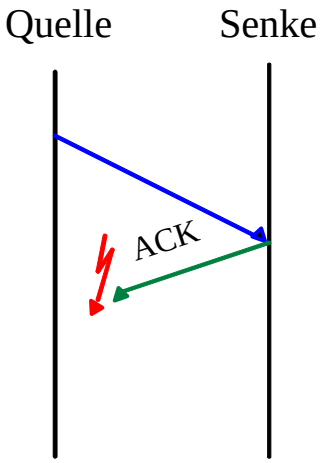
- Fehlerbehebung (fehlerhafte oder verlorene Blöcke)
- Pufferverwaltung
- Flußkontrolle

Bestätigungen (ACKs)

Zwei Fälle von Datenverlust



a) Verlust eines Datenblocks:
- Senke wartet auf Daten
- Quelle wartet auf Bestätigung



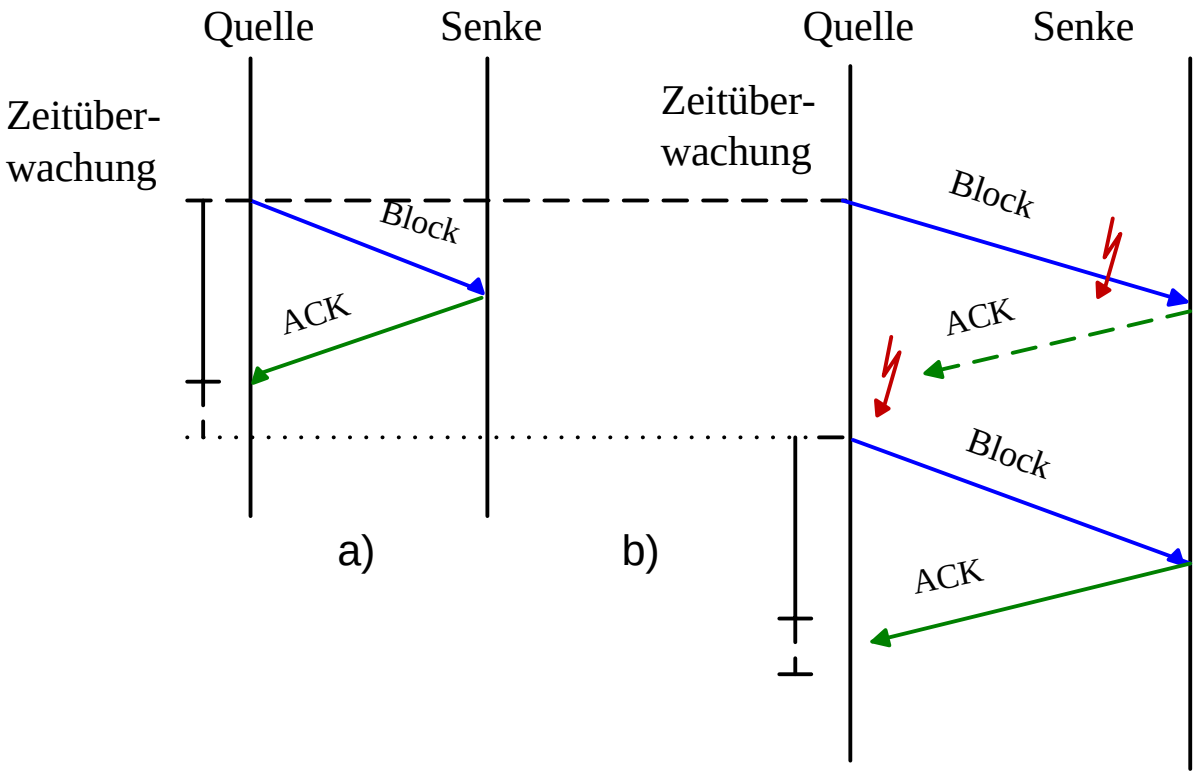
b) Verlust einer Quittung:
- Quelle wartet auf Bestätigung
- Senke wartet auf Daten

Ohne Zeitschranke (Time-out): Blockierung des Senders

Lösung: Einführung einer Zeitschranke (Time-out) beim Sender

Bestätigungen mit Zeitschranke

Zeitüberwachung auf der Senderseite



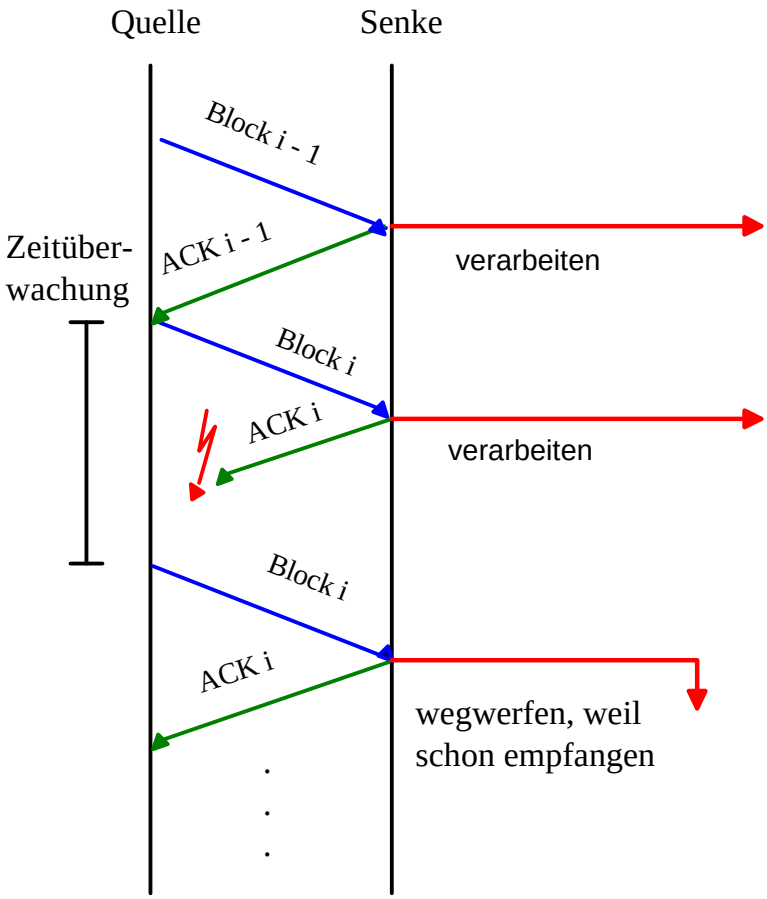
a) Beispiel für eine fehlerfreie Übertragung
b) Beispiel für das Erkennen eines Nachrichtenverlustes durch Ablauf der Zeitüberwachung

Neues Problem:

Der Block wird zweimal übertragen. Der Empfänger weiß nicht, ob er den Block zur Verarbeitung weitergeben soll.

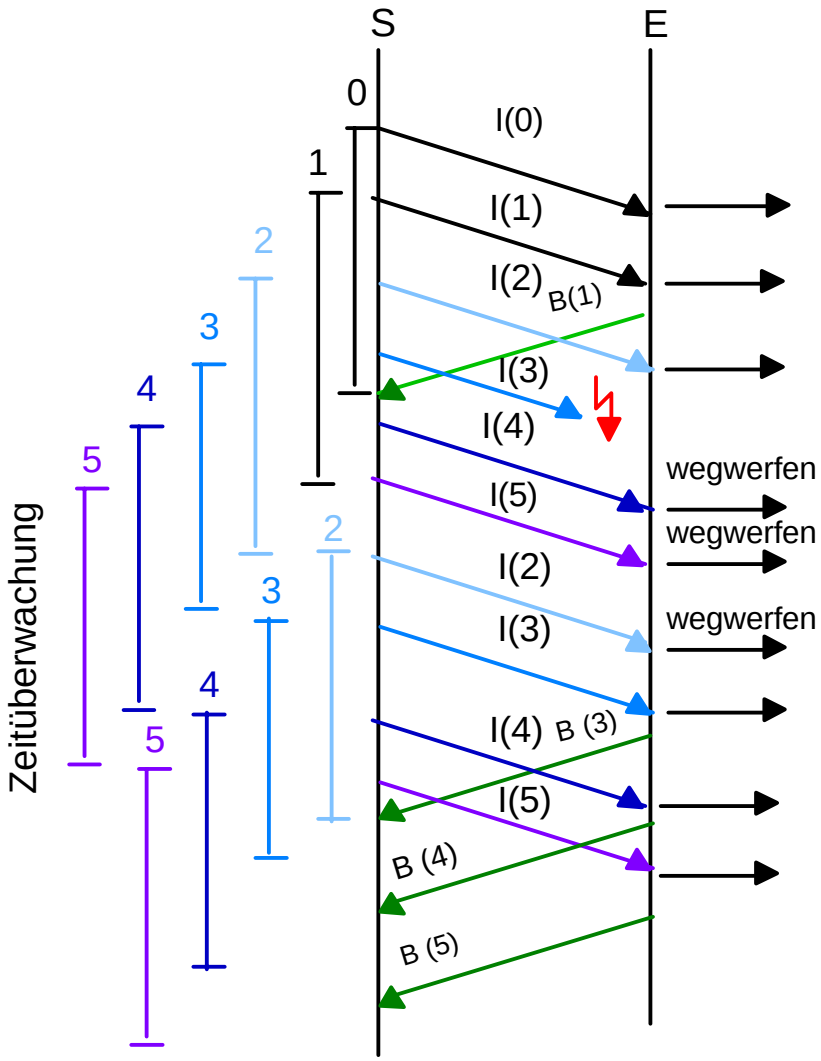
Bestätigungen mit Zeitschranke und Sequenznummern

Lösung: **Sequenznummern**
Jeder Block erhält eine Sequenznummer. **Bei der Wiederholung des Rahmens wird die Sequenznummer beibehalten.**



Sequenznummern

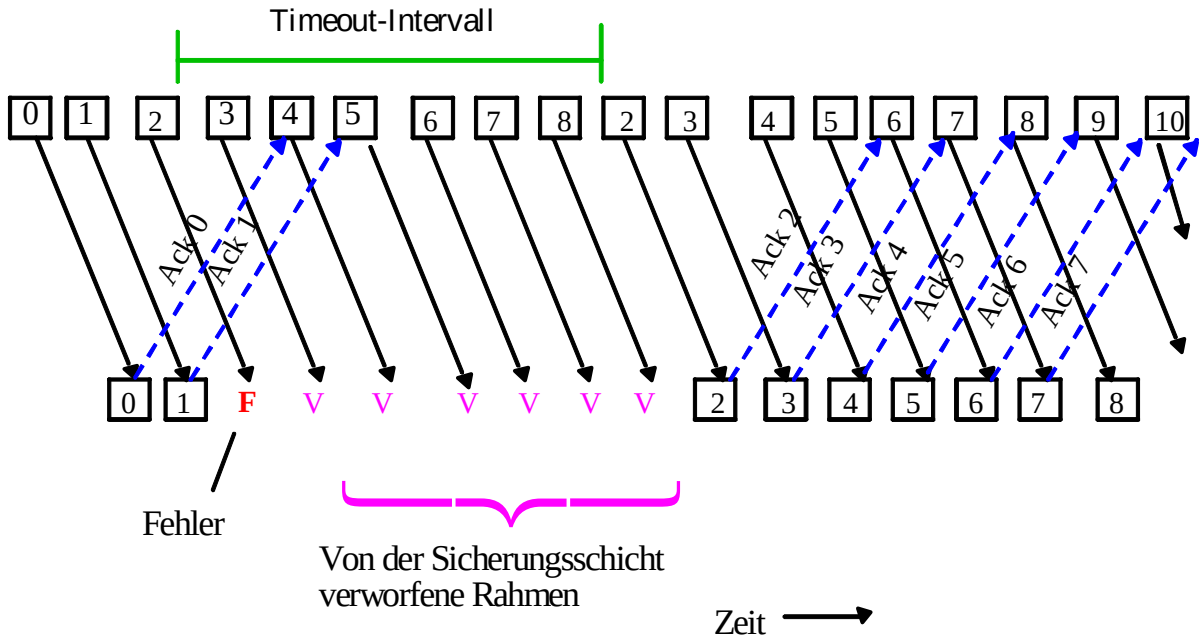
Sequenznummern können auch bei den Bestätigungen verwendet werden. **Mit einer Bestätigung können dann mehrere Informationsblöcke quittiert werden**.



I = Informarmationsblock
B = Bestätigung

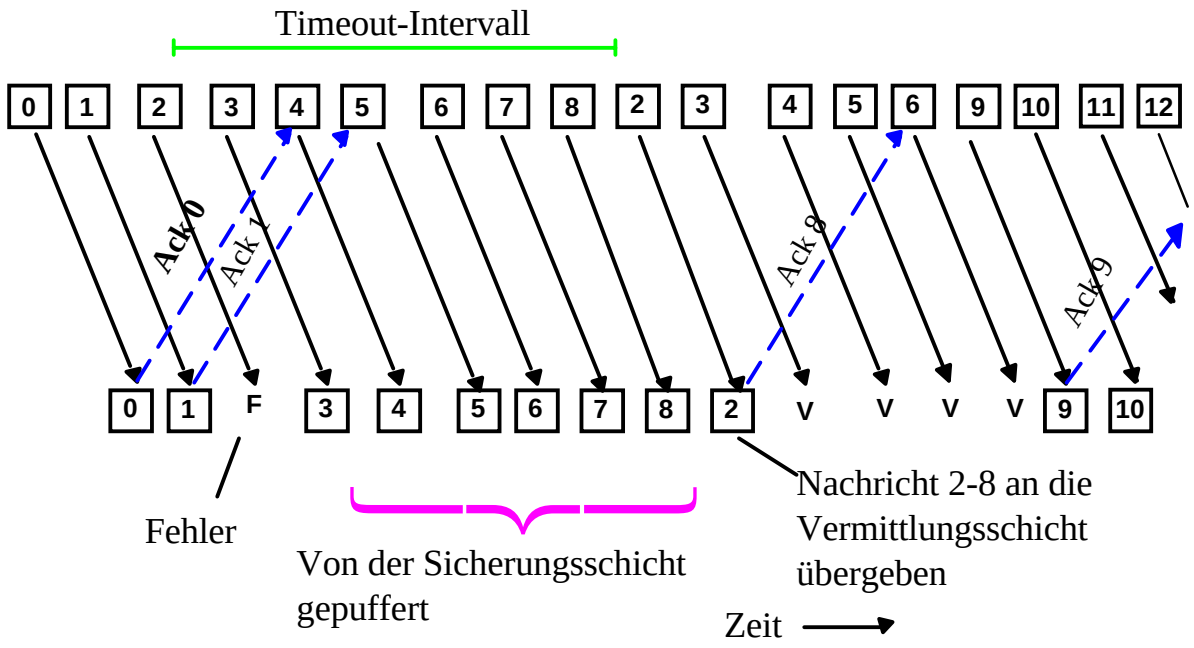
Fehlerbehebung durch "go-back-n" ohne Pufferung beim Empfänger

Im Falle eines Fehlers bleibt das Ack aus. Nach Ablauf des Timers überträgt der Sender **sämtliche** Rahmen ab dem Unbestätigten neu.



Fehlerbehebung durch "go-back-n " mit Pufferung beim Empfänger

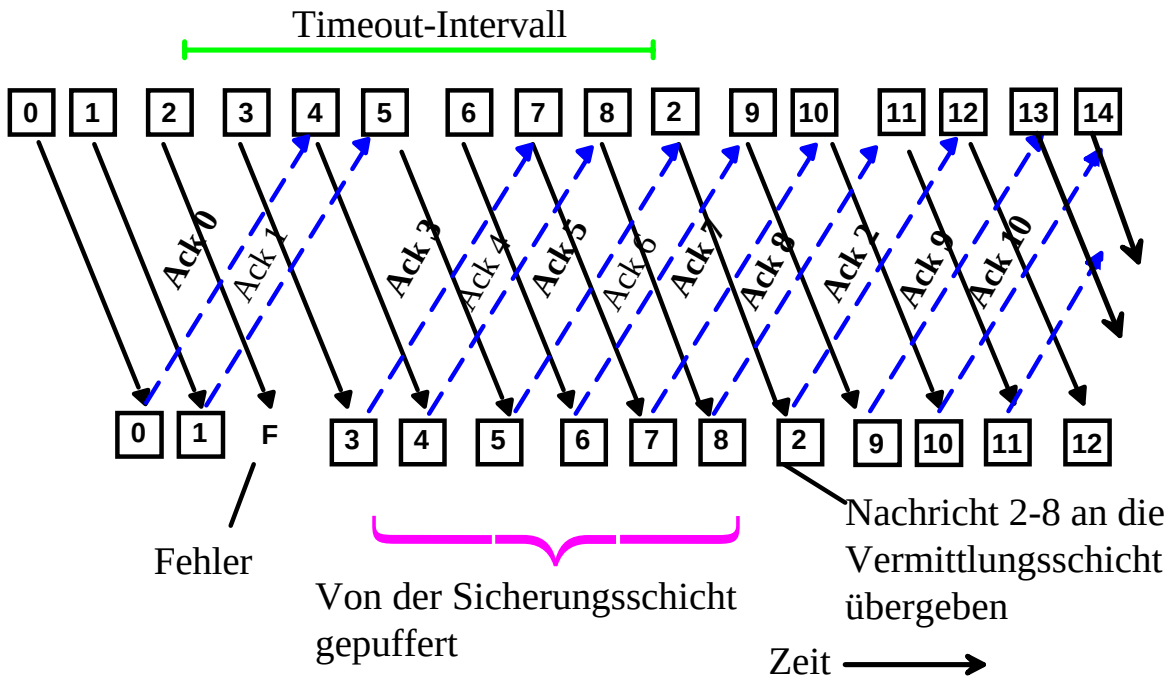
Im Falle eines Fehlers bleibt das Ack aus. **Der Empfänger puffert** die danach noch korrekt erhaltenen Rahmen. Nach Ablauf des Timers beginnt der Sender, alle Rahmen ab dem Unbestätigten neu zu übertragen, bis er ein kumulatives Ack vom Empfänger erhält. Er macht dann mit dem ersten noch nie übertragenen Rahmen weiter.



Die erneute Übertragung der Rahmen 7 und 8 kann durch die kumulative Bestätigung vermieden werden. Dieses Verfahren ist allerdings vergleichsweise kompliziert und wird in der Praxis kaum eingesetzt.

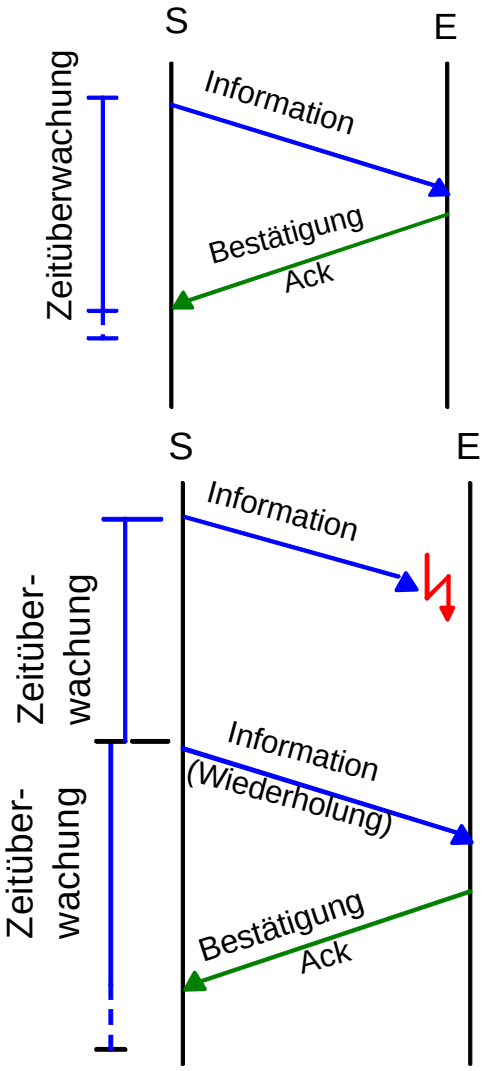
Fehlerbehebung durch "selective repeat"

Der Empfänger bestätigt jeden korrekt erhaltenen Rahmen sofort. Er puffert alle korrekt erhaltenen Rahmen auch nach einem fehlerhaften Rahmen. Bei Ablauf des Timers überträgt der Sender **nur den nicht bestätigten Rahmen** neu.



Passive Fehlerkontrolle

- Kein Versenden von NACKs



ohne Übertragungsfehler

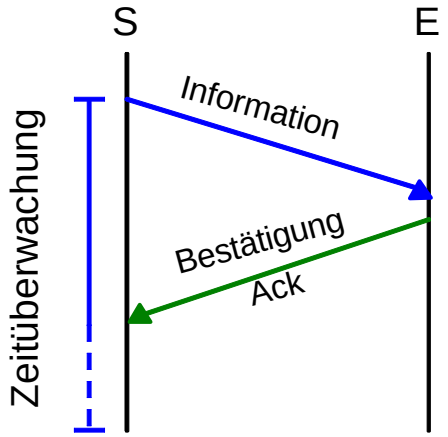
mit Übertragungsfehler

Nachteil der passiven Fehlerkontrolle:

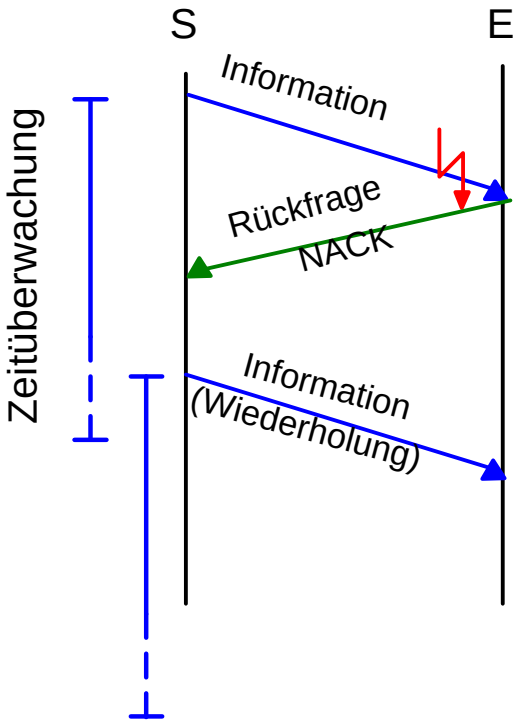
- Keine Unterscheidung zwischen fehlenden und fehlerhaften Blöcken
- Zeitverzögerung bis zur Wiederholung des Sendevorgangs

Aktive Fehlerkontrolle

- Versenden von NACKs



ohne Übertragungsfehler



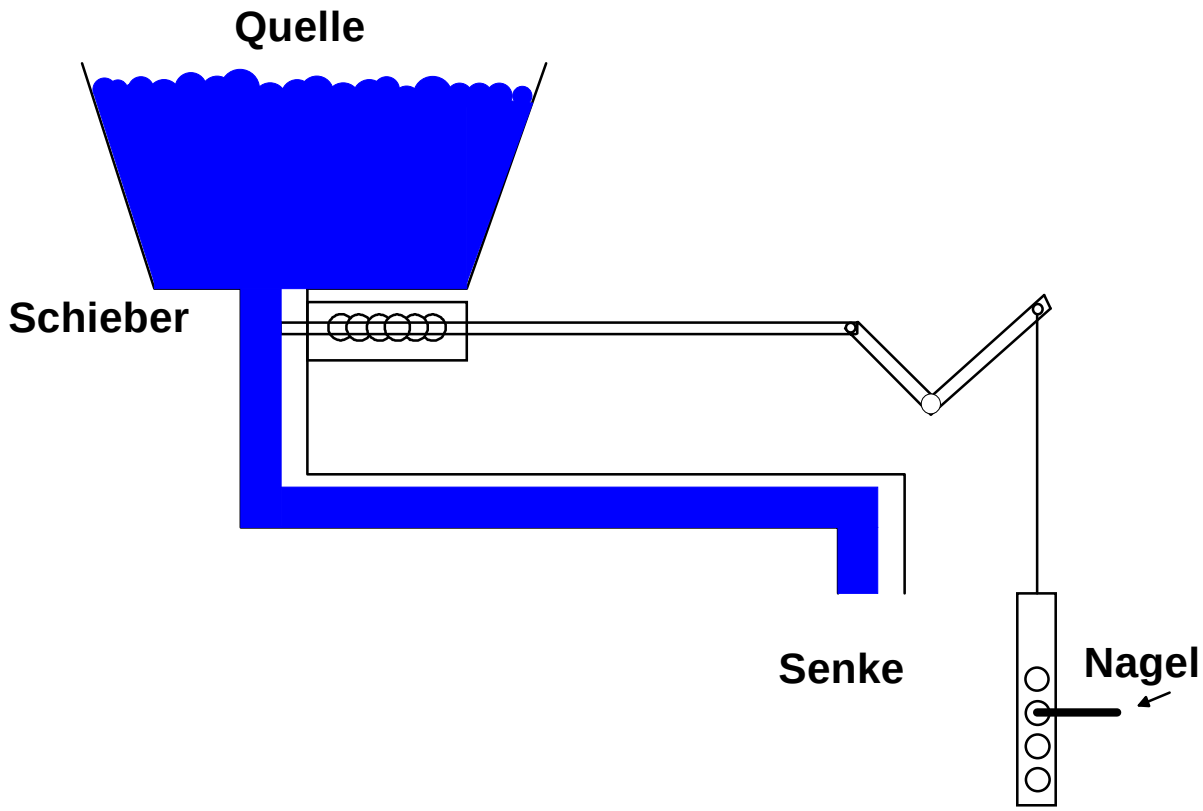
mit Übertragungsfehler

NACK = Negative Acknowledgement

In Rechnernetzen wird sowohl die aktive als auch die passive Fehlerkorrektur verwendet.

3.5 Flußkontrolle

Das Prinzip der Flußkontrolle



Stellt eine Rückkopplung zur Verfügung, um zu verhindern, daß der Sender den Empfänger überschwemmt.

Ein einfaches Stop-and-Wait-Protokoll

Annahmen:

- Fehlerfreie Übertragung
- Beschränkte Anzahl von Puffern
- Verarbeitungsgeschwindigkeiten können sich unterscheiden

Benutze ACK für eine einfache Flußkontrolle. Es ist nie mehr als ein Rahmen auf dem Weg.

Ein einfaches Stop-and-Wait-Protokoll

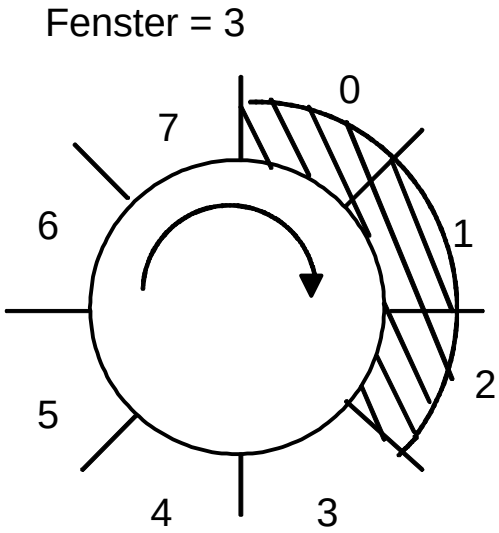
```
procedure sender;
var s:      frame
    buffer: message
    event:  EvType
begin
   while (true) do
      begin
         FromHost (buffer);
         s.info = buffer;
         sendf (s);
         wait (event);      (*wait for ack*)
      end;
end;
procedure receiver;
var r,s:   frame;
    Event: EvType;
begin
   while (true) do
      begin
         wait (event);      (*wait for frame arrival*)
         getfr (r);
         ToHost (r.info);
         sendf (s);         (*send ack to sender*)
      end;
end;
```

Flußkontrolle mit Schiebefenster

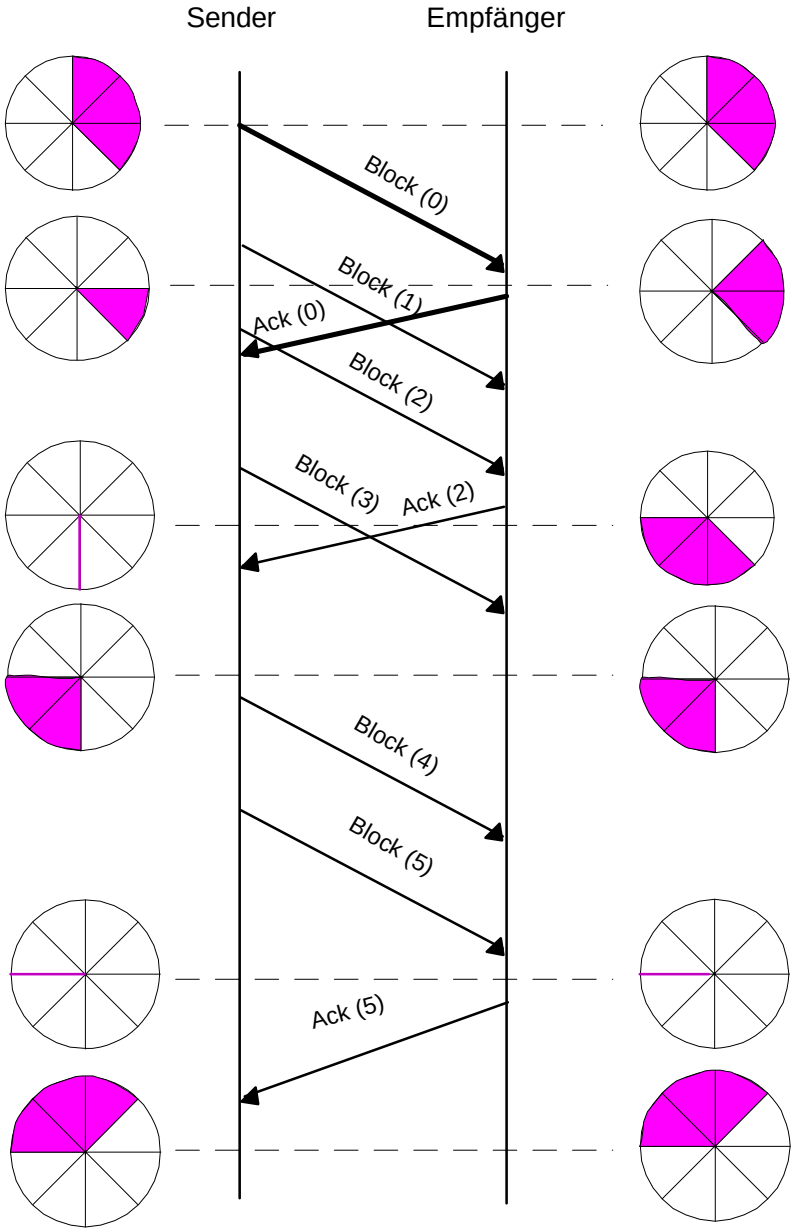
Fenstermechanismus:

- Nach dem Verbindungsaufbau besitzt der Sender das Recht, so viele Informationsrahmen zu senden, wie durch die Fenstergröße vorgegeben ist.
- Spätestens dann muß vom Empfänger eine Bestätigung eintreffen, ansonsten unterbricht der Sender die Übertragung von Rahmen.
- Der Empfänger kann schon vor dem Erreichen der Fenstergröße Bestätigungen an den Sender übermitteln (Öffnen des Fensters).

Beispiel: Fenstergröße = 3



Schiebefenster



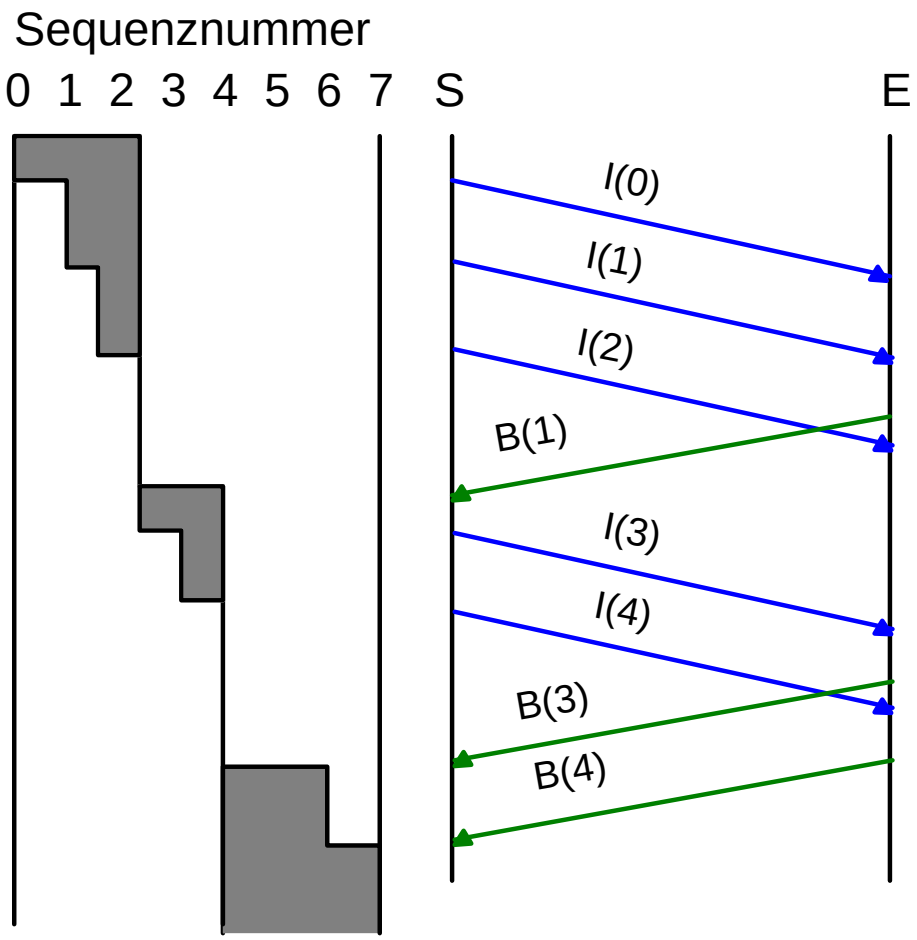
W = 3

Sendeaktionen verschieben den hinteren Zeiger, ACKs den vorderen Zeiger

Eintreffende Blöcke verschieben den hinteren Zeiger, versendete ACKs den vorderen Zeiger

Schiebefenster

Öffnen und Schließen des Fensters

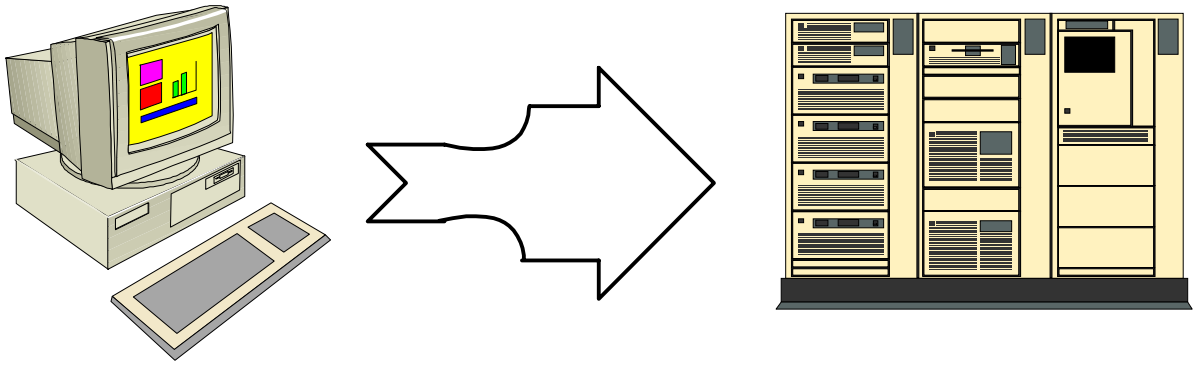


I = Informationsblock
B = Bestätigung

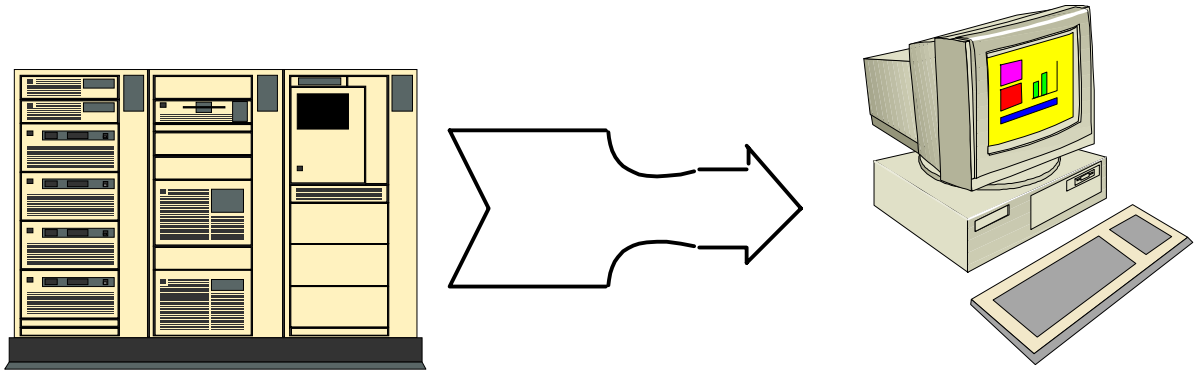
Die Sequenznummern werden **sowohl** zur Fehlerkontrolle **als auch** zur Flußsteuerung verwendet.

Fenstergröße und Puffer beim Empfänger (1)

Beispiele für den Zusammenhang zwischen unterschiedlich leistungsfähigen Partnern, der Fenstergröße und der Anzahl zu reservierender Puffer beim Empfänger.



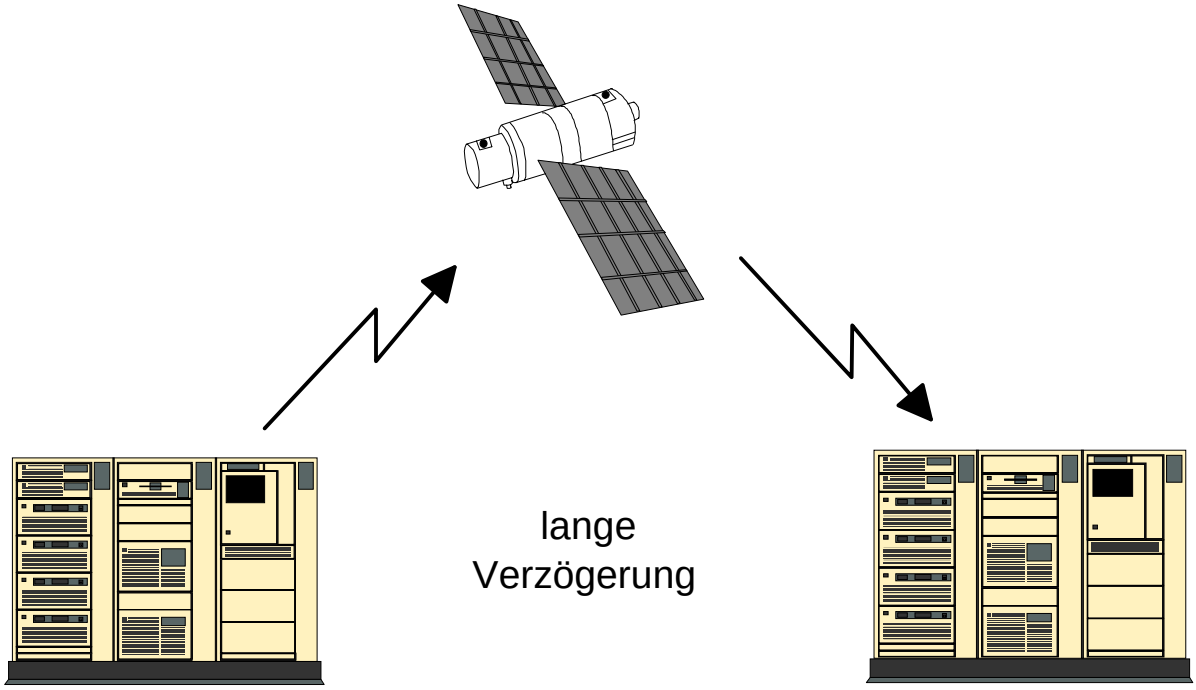
a) Langsame Quelle (PC) – schnelle Senke (Großrechner): 1 Puffer ausreichend, w = 1



b) Schnelle Quelle (Großrechner) – langsame Senke (PC): 1 Puffer ausreichend, w = 1

Fenstergröße und Puffer beim Empfänger (2)

c) Balanciertes Verhältnis, z.B. zwischen zwei Großrechnern: es sind w Puffer erforderlich, z.B.: $2 \leq w \leq 7$



d) Übertragung über einen Nachrichtensatelliten: es sind w Puffer erforderlich, z.B.: $16 \leq w \leq 127$

Je größer die Verzögerung bei der Übertragung desto größer muß die Fenstergröße w gewählt werden, desto mehr Puffer werden auf der Empfängerseite benötigt. Der Empfänger benötigt stets mindestens w Puffer.

3.6 Beispiel: HDLC

Das wichtigste Protokoll der Sicherungsschicht.
Eng verwandt sind:

- HDLC — High-Level Data Link Control (ISO)
- LLC 2 — Logical Link Control Typ 2 in LANs
- LAP-B — Link Access Procedure – Balanced (CCITT; Schicht 2 von X.25

HDLC Standards

ISO 3309	HDLC procedures – Frame structure
ISO 4335	HDLC procedures – Elements of procedures
ISO 6159	HDLC unbalanced classes of procedures
ISO 6256	HDLC balanced class of procedures
X.25, Level 2	Link Access Procedure Balanced (LAP B) (Schicht 2 der CCITT-Empfehlung X.25)

HDLC – High Level Data Link Control

Basis-Rahmenformat:

	Flag 01111110	Address	Control	Data	Frame Check	Flag 01111110
Bits:	8	8	8	≥ 0	16	8

Transparenz durch Bitstopfen (Bit Stuffing)

Frame Check Sequence (FCS):

Cyclic Redundancy Check mit

$$G(x) = x^{16} + x^{12} + x^5 + 1$$

Prozedurklassen

Asymmetrische Konfiguration (unbalanced)

- Primärstation: Management der Verbindung
- Eine oder mehrere Sekundärstation(en)
- Punkt-zu-Punkt-Verbindungen und Mehrpunkt-Verbindungen
- Normaler Antwortmodus (Normal Response Mode):
  Eine Sekundärstation kann nur nach einem Sendaufruf ("polling") durch die Primärstation senden.
- Asynchroner Antwortmodus (Asynchronous Response Mode):
  Die Primär- und die Sekundärstation kann übertragen, wenn die Leitung frei ist.

Symmetrische Konfiguration (Balanced)

- Totale Symmetrie zwischen den Stationen
- nur Punkt-zu-Punkt-Verbindungen (keine Mehrpunktverbindungen)
- nur asynchroner Modus

Prozedurklassen

- "Unbalanced" Operation im Normalen Antwortmodus
- "Unbalanced" Operation im Asynchronen Antwortmodus
- "Balanced" Operation im Asynchronen Antwortmodus

Asynchronous Balanced Mode (ABM) ist die Grundlage von LAP B (Link Access Procedure Balanced), der Schicht 2 des X.25-Standards.

Adreßfeld

Ursprünglich 8 Bits lang, diente zur Adressierung der Sekundärstation beim Polling

Erweiterte Adressierung

Die Größe des Adreßfelds kann ein Vielfaches von 8 Bits sein.

Steuerfeld (Control Field)

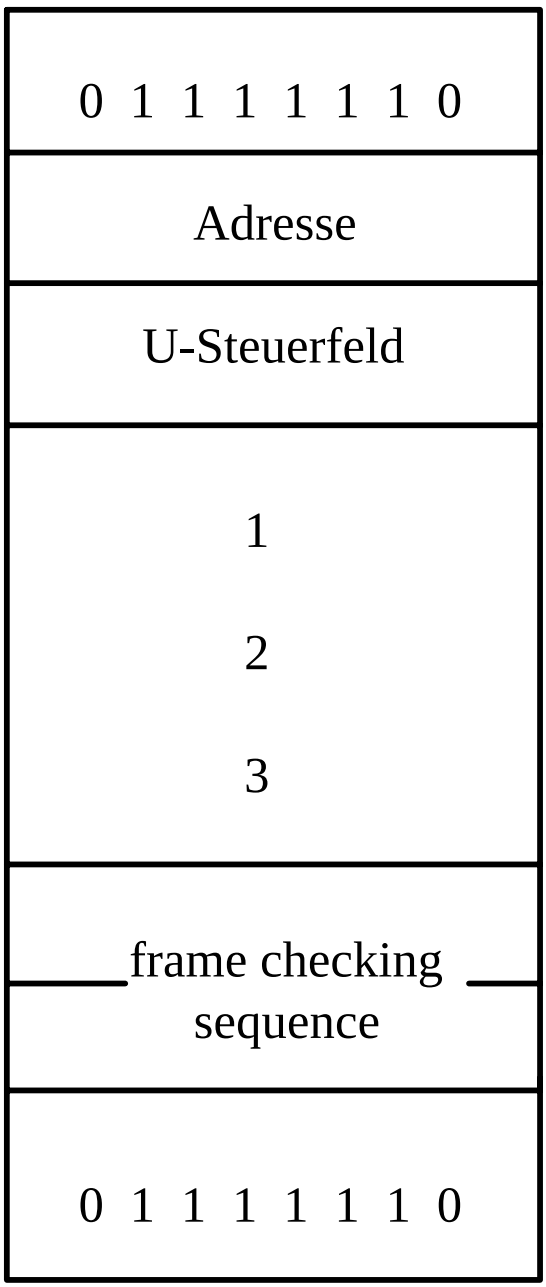
Es gibt drei verschiedene Rahmentypen im HDLC:

- I-Rahmen (information frames)
  werden zur Datenübermittlung benutzt
- S-Rahmen (supervisory frames)
  werden zur Steuerung des Datenflusses benutzt
- U-Rahmen (unnumbered frames)
  steuern die Verbindung

Der Rahmentyp wird im Steuerfeld („control“) angegeben.

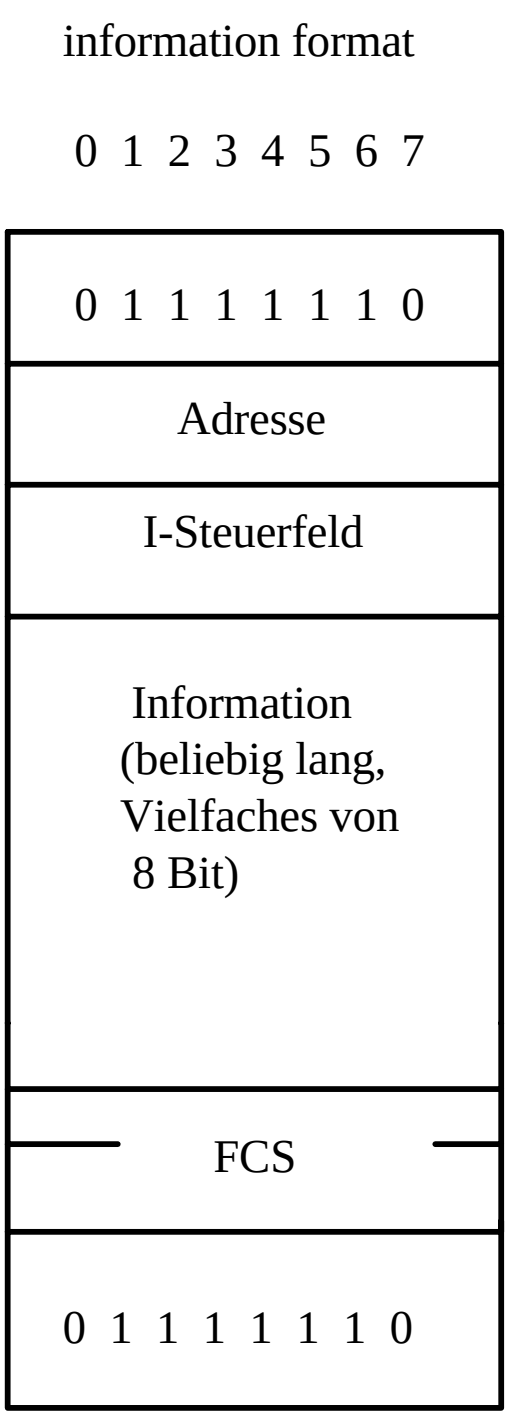
HDLC Rahmenformate (1)

unnumbered format



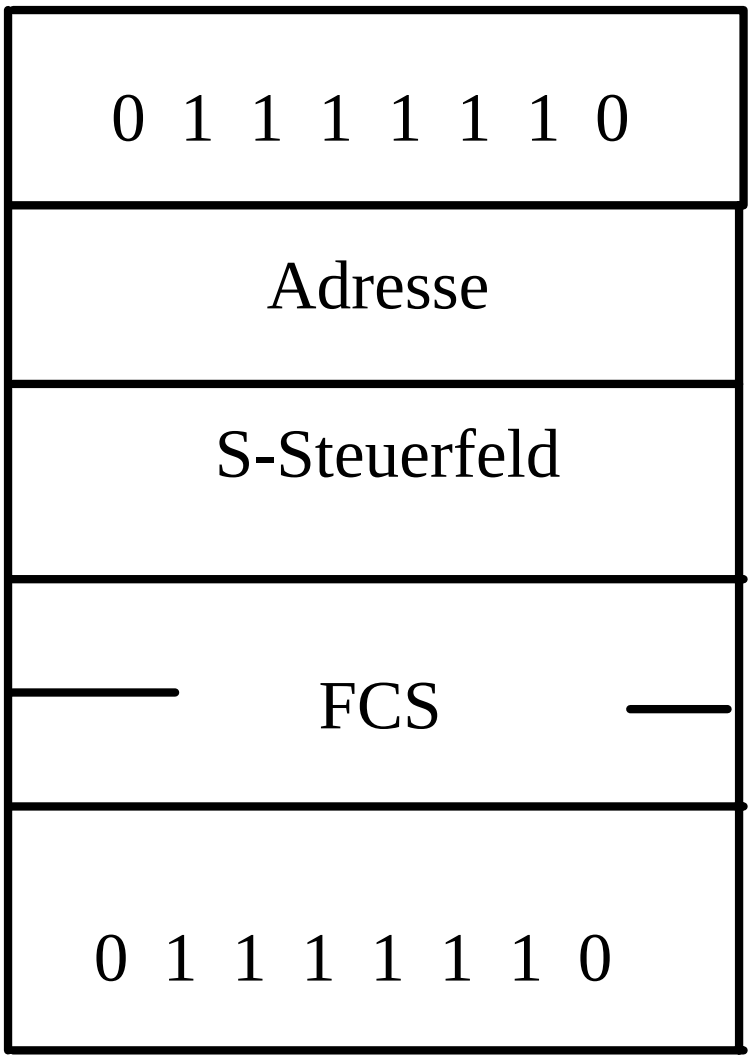
HDLC Rahmenformate (2)

information format

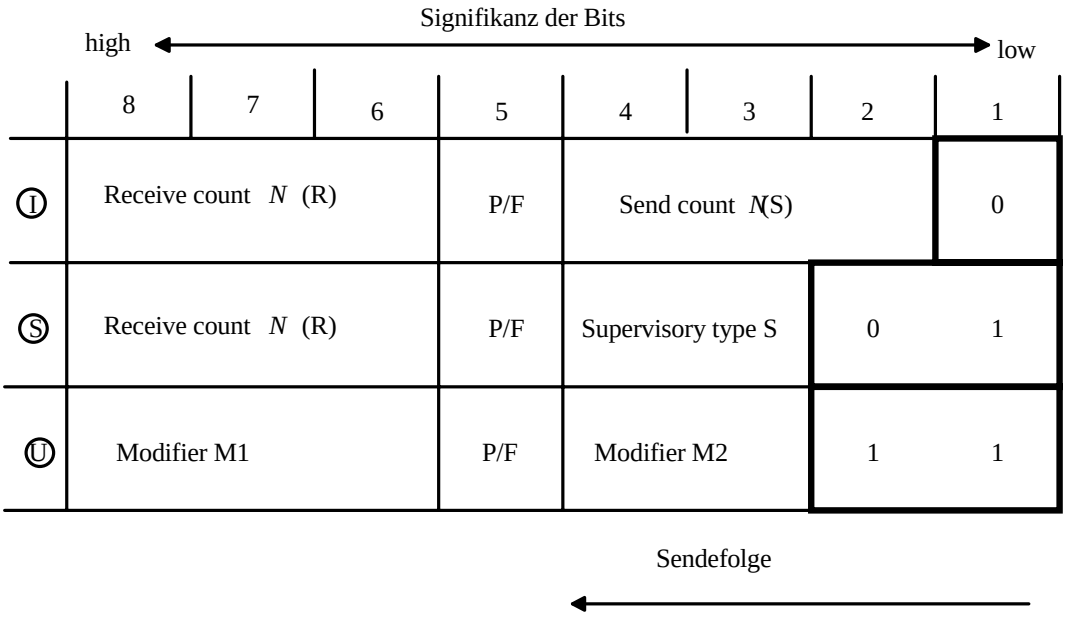


HDLC Rahmenformate (3)

supervisory format



Inhalt der Steuerfelds im Basis-Format



HDLC benutzt ein Schiebefenster mit einer 3-Bit-Sequenznummer ⇒ Fenstergröße = 7

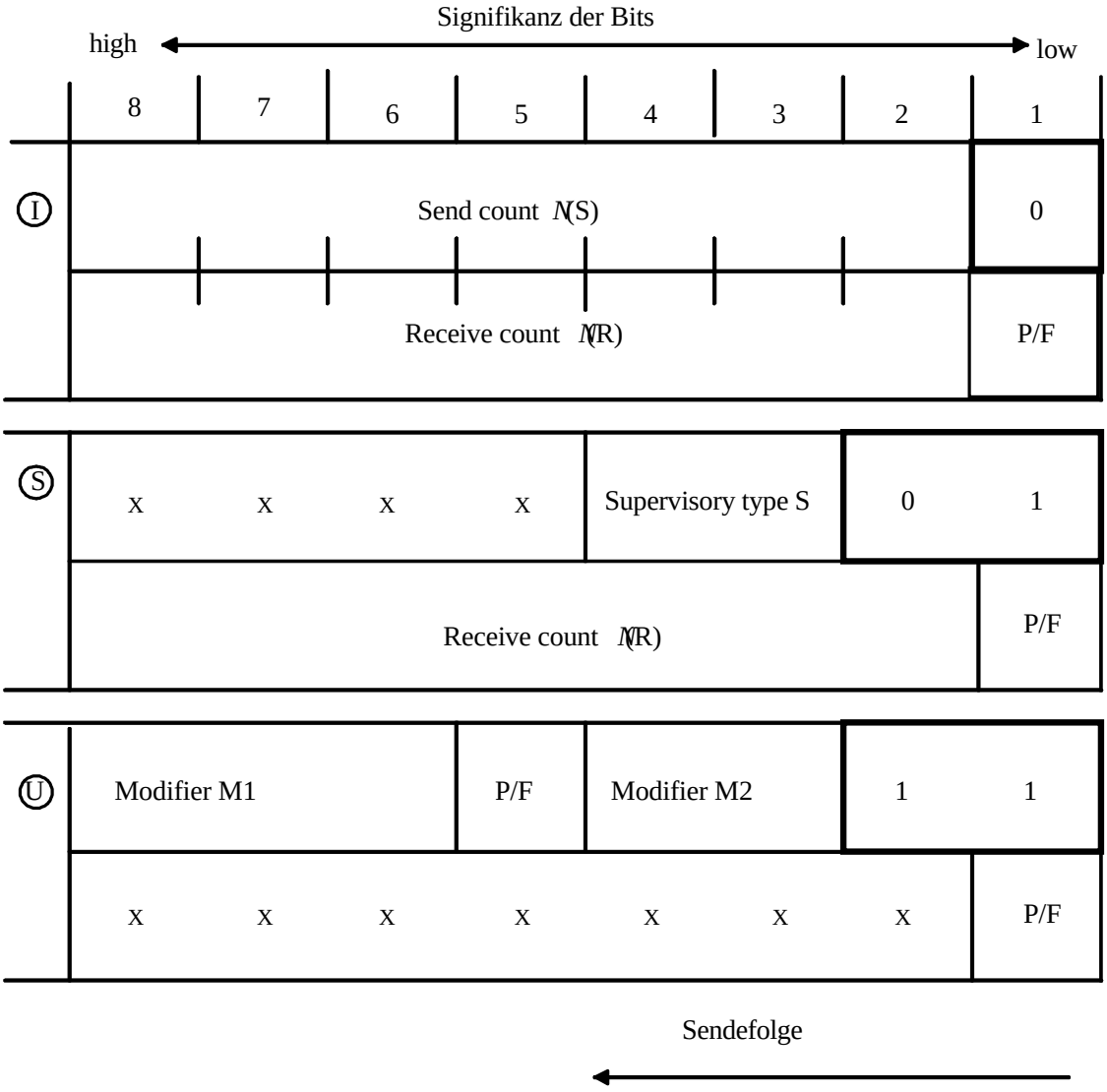
N(R) Empfangszähler

N(S) Sendezähler (Sequenznummer des Rahmens)

N(R) und N(S) sind Zähler für die Anzahl der I-Rahmen, die empfangen und gesendet wurden.

N(R) piggybacked acknowledgement: ACK = Nummer des **ersten nicht erhaltenen** Rahmens (= Nummer des nächsten erwarteten Rahmens) kann in I-Rahmen des Rückkanals übertragen werden

Erweitertes Steuerfeld (16 Bit)



N(S) und N(R) sind jetzt Zähler mit 7 Bit.
modulo 128 $\Rightarrow$ Fenstergröße = 127
Anwendung: z.B. Satellitenverbindungen

Command / Response

Command	Rahmen eines beliebigen Typs, der durch eine Primärstation gesendet wird
Response	Rahmen eines beliebigen Typs, der durch eine Sekundärstation gesendet wird

Kombinierte Stationen können sowohl Commands als auch Responses senden.

P/F-Bit: Poll-Bit in Command-Rahmen
Der Empfänger muß diesen Rahmen durch einen Response-Rahmen mit gesetztem Final-Bit bestätigen.

Supervisory-Rahmen (1)

2 Bits zur Unterscheidung ⇒ 4 verschiedene S-Rahmen

- Receive Ready (RR) 00
  Station kann I-Rahmen empfangen
  N(R): nächster erwarteter Rahmen

- Receive Not Ready (RNR) 01
  Eine Station zeigt der anderen an, daß sie zeitweise nicht empfangen kann. Sender stoppt die Übertragung

- Reject (REJ) (entspricht einem NACK) 10
  Zeigt an, daß ein Übertragungsfehler entdeckt wurde

  N(R): erster Rahmen in der Folge der nicht korrekt empfangen wurde, also der Rahmen, der erneut gesendet werden soll.

  Sender: Muß alle ausstehenden Rahmen erneut übertragen, beginnend mit N(R).

Supervisory-Rahmen (2)

- Selective Reject (SREJ) 11
  Zeigt dem Sender an, daß ein Übertragungsfehler entdeckt wurde. Fordert nur die erneute Übertragung des Rahmens spezifiziert durch N(R) an.

REJ und SREJ werden nur in der beidseitigen Datenübermittlung (vollduplex) benutzt.

LAP B: RR, RNR, REJ
kein SREJ

Unnumbered-Rahmen (1)

5 Bits verfügbar
⇒ 32 verschiedene U-Rahmen sind möglich
(nicht alle werden benutzt)

- Zum Setzen der Modi für die drei Klassen werden folgende Kommandos benutzt:
  - Set Normal Response Mode (SNRM)
  - Set Normal Response Mode Extended (SNRME)
  - Set Asynchronous Response Mode (SARM)
  - Set Asynchronous Response Mode Extended (SARME)
  - Set Asynchronous Balanced Mode (SABM)
  - Set Asynchronous Balanced Mode Extended (SABME)
- Sie werden benutzt, um eine Verbindung auf der Ebene der Sicherungsschicht herzustellen.
- N(S), N(R) $\rightarrow$ 0
- Erweiterte Version ("Extended"):
  Benutzt für das erweiterte Format des Steuerfeldes (Control Field)

Unnumbered-Rahmen (2)

- Unnumbered Acknowledgement (UA)
  Wird zur Bestätigung eines U-Rahmens benutzt
- Disconnect (DISC)
  Wird durch die Primärstation oder durch eine kombinierte Station benutzt, wenn sie die Verbindung lösen will.
  Bestätigung: UA
- Disconnected Mode (DM)
  Wird benutzt um anzuzeigen, daß eine Station logisch abgekoppelt ist.
  Nur Kommandos, die einen Modus setzen, sind für eine logisch abgekoppelte Station gültig.
  DM wird normalerweise als Antwort auf alle Kommandos mit Ausnahme derjenigen gesendet, die einen Modus setzen.
- Frame Reject (FRMR)
  Wird gesendet wenn ungültige Konditionen entdeckt werden.