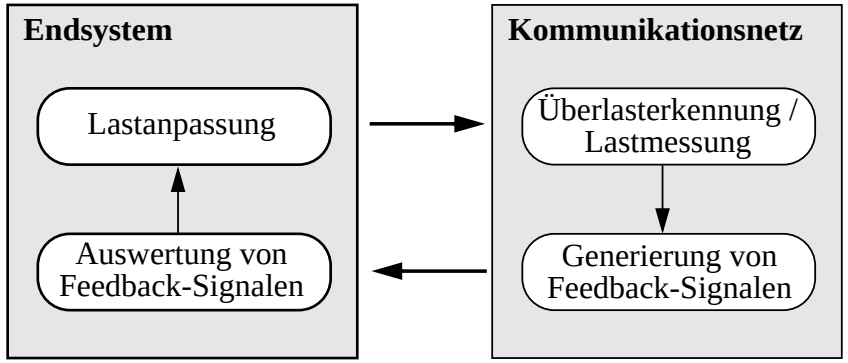
Einige Methoden der Laststeuerung



- "Choke-Paket" mit "stop" oder "langsamer" von einem überlasteten Netzknoten an seine Quellen
- Informieren über den Netzzustand durch kontinuierliches Versenden von Paketen, die die Ende-zu-Ende-Verzögerung messen
- Lasteinspeisung zurückfahren, wenn Verlustrate steigt, Quittungen ausbleiben etc. und bei Besserung allmählich wieder hochfahren

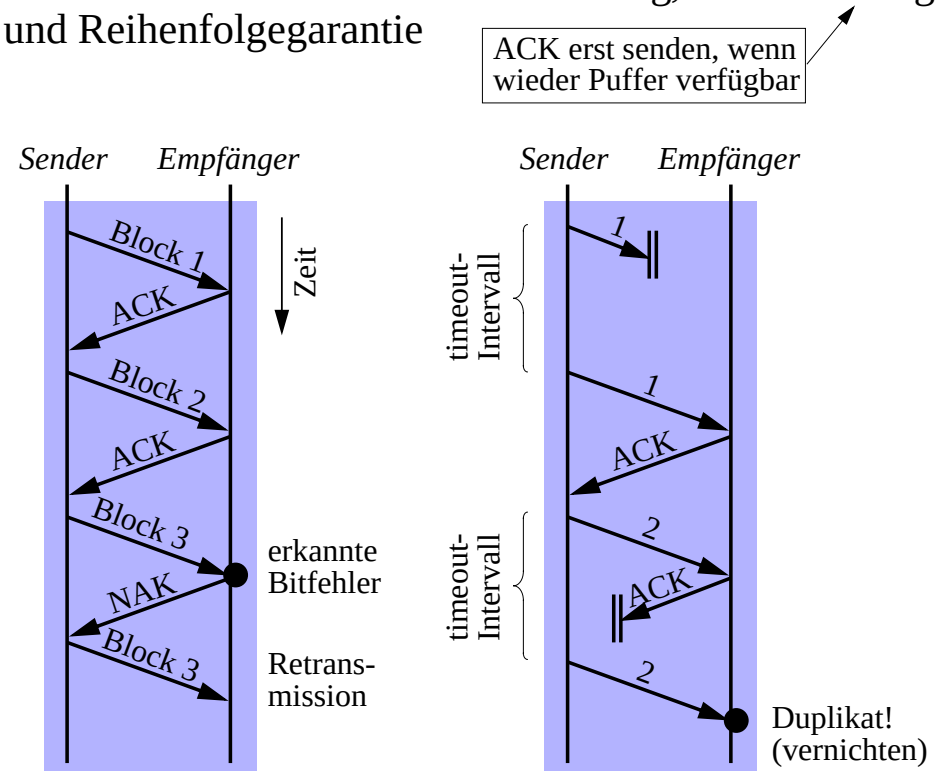
- Anforderungen und Probleme z.B.:
  - schnelle Reaktion auf plötzliche Änderung der Lastsituation
  - Überreaktion vermeiden
  - bei Überlast Durchsatz stets nahe 100% halten
  - Fairness (keine wesentliche Benachteiligung einzelner Teilnehmer)

- Reservierung von Ressourcen bei Verbindungsaufbau?
  - Problem: verschwendet ggf. ungenutzte Kapazität
  - für verbindungslose Kommunikation (z.B. Internet-IP) eher ungeeignet

Send-and-Wait-Protokoll

(gelegentlich auch als "Stop-and-Wait" bezeichnet)

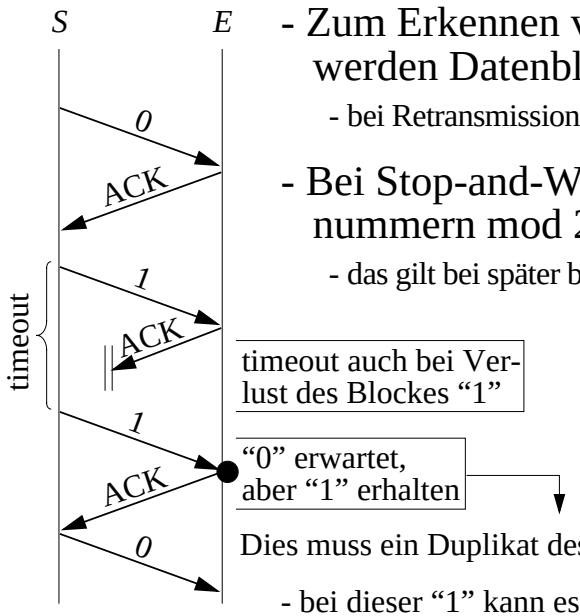
- Einfachstes Protokoll zur Datensicherung, Flusssteuerung und Reihenfolgegarantie



- Senderseitige timeouts ("ARQ"), damit es bei Verlust von Blöcken oder ACKs nicht zu Blockaden kommt
  - damit dient NAK lediglich der Beschleunigung
  - ohne NAK: PAR-Protokoll ("Positive Acknowledge and Retransmit")
  - timeout-Intervalle "richtig" wählen
    - unnötiges Warten <--> unnötige Wiederholung
    - schwierig bei grosser Varianz bei den Übertragungszeiten
    - ggf. adaptiv an Übertragungszeit, Fehlerrate etc. anpassen

- Frage: Sind "zu langsame" ACKs eigentlich problematisch?

Alternating-Bit



- Zum Erkennen von Duplikaten werden Datenblöcke i.a. numeriert
  - bei Retransmission Sequenznummer beibehalten
- Bei Stop-and-Wait genügen Sequenznummern mod 2 ("alternating bit")!
  - das gilt bei später betrachteten Protokollen i.a. nicht!

timeout auch bei Verlust des Blockes "1"

"0" erwartet, aber "1" erhalten

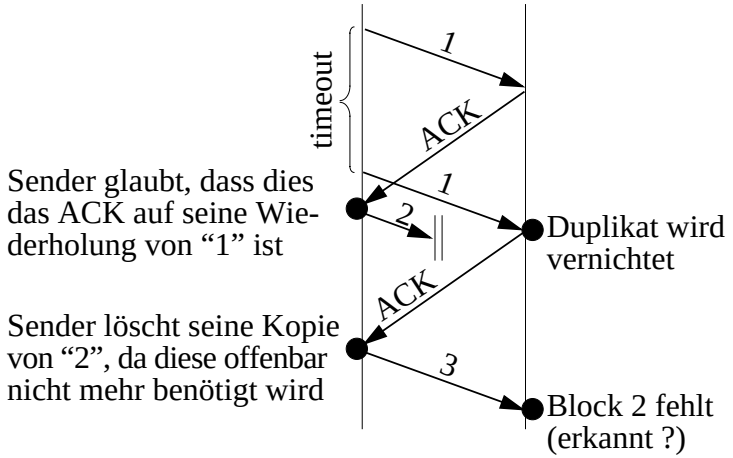
Dies muss ein Duplikat des vorherigen Blocks sein, denn

- bei dieser "1" kann es sich nicht um eine "1" aus der Zukunft handeln (weil der Sender vor dem Senden anderer Blöcke das ACK abwartet)
- es kann sich auch nicht um einen mit "1" markierten früheren Block handeln (weil alle früher versendeten Blöcke bereits korrekt empfangen wurden)

- Wie überzeugend ist eigentlich eine solche Argumentation?
  - zumindest bei verwickelteren Abläufen kann man sich leicht täuschen
  - formalere Modelle zur Spezifikation und Verifikation von Protokollen scheinen daher notwendig zu sein
- Mit dem ACK schickt man i.a. die Sequenznummer des bestätigten Blocks mit
  - hier also ACK(0) bzw. ACK(1)
  - aber nützt das etwas oder kann man darauf verzichten?

Ein Protokollfehler?

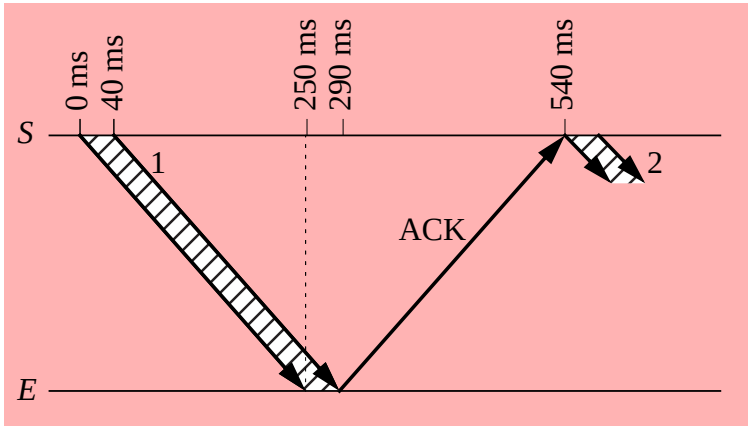
- Wir betrachten nochmals das *Stop-and-Wait*-Protokoll:



- Hier war das erste ACK so langsam, dass es erst *nach* dem timeout des Senders ankam
- Es ergeben sich einige Fragen...
  - Ist das Protokoll tatsächlich falsch? Lässt es sich einfach "reparieren"?
  - Hätten wir alle das Problem vor der Implementierung erkannt?
  - In verwickelteren Fällen auch?
  - War die Argumentation beim Alternating-Bit vielleicht doch nicht ganz wasserdicht?
  - Ist das der einzige (bzw. letzte) "bug" des Protokolls?
  - Denkübung: Wie kann man das Problem möglichst einfach lösen?
  - Helfen dabei Sequenznummern in den Acks? (Beachte auch Alternating-Bit und sehr langsame ACKs!)
  - Wie kann man sich überhaupt jemals sicher sein?

Kanalausnutzung von Stop-and-Wait

- Es kann stets höchstens ein Block "unterwegs" sein, z.B.:
  - Satellitenlink mit 500 ms Round-Trip-Delay
  - Übertragungsrate 50 kb/s
  - Datenblöcke von 2 kb

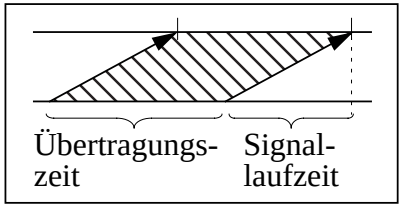


- Bandbreite-Delay-Produkt = 50 kb/s × 0.5 s = 25 kb
- Speicherkapazität des Kanals mit 2 kb also schlecht genutzt
- Da ACKs vernachlässigbar kurz sind, ergibt sich genauer eine Ausnutzung ("Effizienz") von nur 40/540 = 7.4%

Effizienz von Stop-and-Wait

- Betrachte einen Parameter a:

$$a = \frac{\text{Signallaufzeit}}{\text{Übertragungszeit}}$$



- Es folgt damit:

$$a = \frac{\text{Bandbreite x Signallaufzeit}}{\text{Blockgrösse}}$$

bei konstanter Blockgrösse ist dies also proportional zum "Bandbreite-Delay-Produkt"!

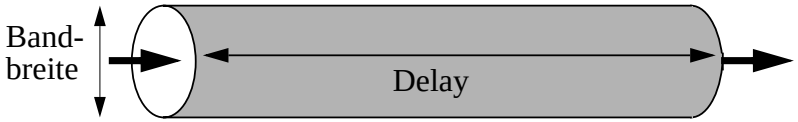
- Für die Effizienz E ("Durchsatz / Kapazität") gilt:

$$E = \frac{1}{1 + 2a}$$

(Denkübung: wieso? Zur Plausibilität setze a=1: Block füllt die Pipe; je eine Zeiteinheit bis Ende der Übertragung, Ankunft des letzten Bits, Ankunft Ack.)

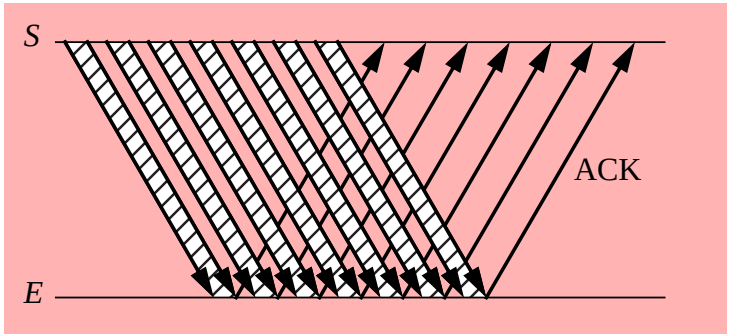
- Beispiel 1 (von vorhin): a = 50 kb/s x 250 ms / 2 kb = 6.25 --> **E = 7.4%**
- Beispiel 2: LAN, Blockgrösse 1000 Bit, Übertragungsrate 10 Mb/s, Entfernung 100 - 10000 m --> **E ≈ 0.5 - 0.99**
- Beispiel 3: Modem 28.8 kb/s, Blockgrösse 1000 Bit
  1000 m --> **E ≈ 100%**
  5000 km --> **E ≈ 40%**
- Beispiel 4: ATM (Blockgrösse 424 Bits, Glasfaser mit V=2 x 10^8 m/s, 155 Mb/s), 1000 km --> **E = 0.027%**

- Wie kann man das verbessern?
  - Motto: Keep the pipe full!



Pipelining

- Auch als “continuous ARQ” bezeichnet
- Idee: Sender setzt mehrere Blöcke ab, ehe er wegen evtl. ausstehenden ACKs blockiert



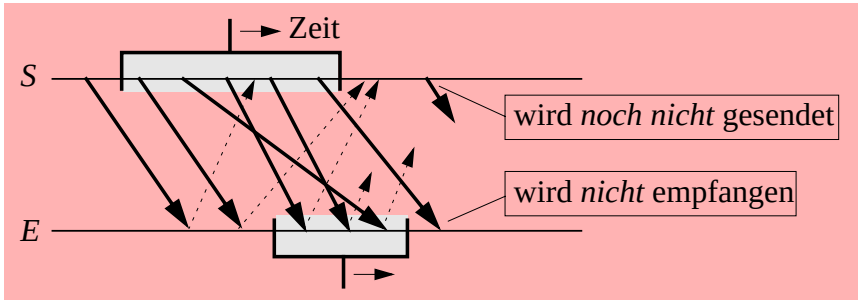
- ACKs müssen den Blöcken eindeutig zuordenbar sein (wieso?)
- Sender muss mehrere Blöcke aufheben, um ggf. einen Block erneut senden zu können
- Sender hat also quasi einen “Kredit” zum Senden von Blöcken
- wie hoch sollte man den Kredit wählen?
- Timeout-Verwaltung und Retransmissionsverwaltung komplexer

==> Sliding-Window-Prinzip

Sliding-Window-Protokoll

“Schiebefenster”

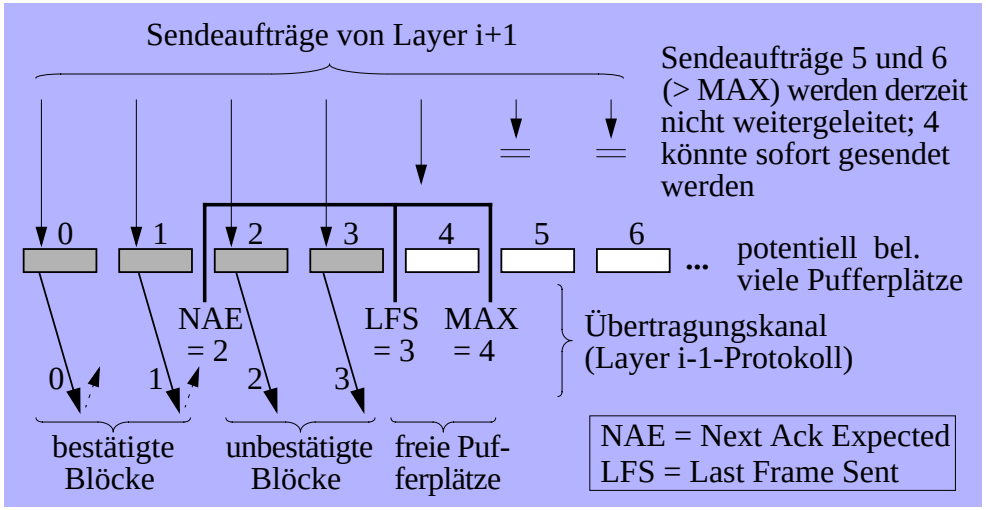
- Protokoll (der Ebene 2 oder höher) für
  - sichere Datenübertragung mittels ARQ
  - Flusssteuerung
  - Reihenfolgegarantie (FIFO-Übertragung)
- Eigentlich eine Familie von Protokollen
  - verschiedene spezifische Varianten
- “Verbesserte” Version von “Send-and-Wait”
  - also Wiederholung nach timeout für ACK
  - nutzt aber Pipelining zur besseren Auslastung des Kanals



- Sendefenster (Grösse variabel aber begrenzt)
  - maximale Fenstergrösse entspricht dem “Kredit” zum Senden
  - aktuelle Fenstergrösse entspricht Zahl ggw. unbestätigter Blöcke
- Empfangsfenster (Grösse i.a. fest)
  - Fenstergrösse bestimmt Zahl der Blöcke, die ausserhalb der Reihe ankommen können
  - Grösse kann verschieden von der Maximalgrösse des Sendefensters sein

Sliding-Window: Senderseite

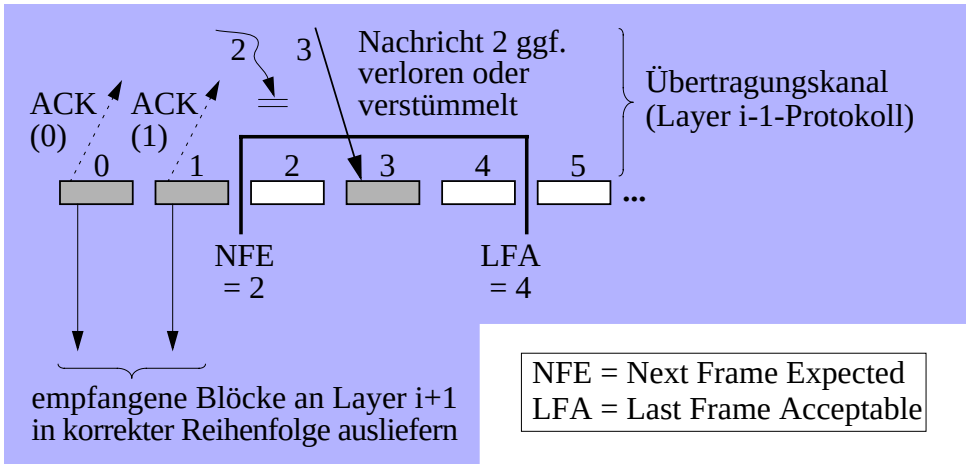
- Alle Nachrichten (und ACKs) seien numeriert: 0, 1, 2,...



- Aktuelle Fenstergrösse = LFS-NAE+1 (hier: 2)
- Maximale Fenstergrösse = MAX-NAE+1 (hier: 3)
- Blöcke innerhalb des Fensters müssen aufbewahrt werden (müssen ggf. wiederholt gesendet werden)
- Senden eines Blockes --> LFS++ (LFS stets ≤ MAX)
- Wenn ein ACK für NAE ankommt, kann das Fenster verschoben werden (NAE++ und MAX++)
  - dadurch können ggf. wieder Pufferplätze frei werden (falls LFS = MAX war) bzw. wartende Blöcke versendet werden
  - es kann aber auch ein anderes ACK(k) mit k ≤ LFS ankommen
- Oft: ACK(k) impliziert ACK(k-1), ACK(k-2)...
  - kumulative ACKs sparen ggf. einige ACKs ein
  - ggf. mit Versenden von ACK(k) auf ACK(k+1) warten...

Sliding-Window: Empfangsseite

- Empfangsfenster bestimmt Bereich von Sequenz-nummern, die gegenwärtig akzeptiert werden
  - entsprechend viele Pufferplätze sind bereitzuhalten



- Blöcke ausserhalb dieses Bereiches werden vernichtet
  - Seqnr < NFE --> Duplikat (aber ACK versenden! (wieso?))
  - Seqnr > LFA --> kein Pufferplatz vorhanden
- Innerhalb des Fensters können Blöcke aber in beliebiger Reihenfolge ankommen
  - oder erst mal ausbleiben (bis der Sender fehlende Blöcke wiederholt)
- Bei Empfang von Block(NFE): zugehöriges ACK senden und Fenster verschieben: NFE++; LFA++
  - ggf. um mehr als 1 verschieben, wenn anschliessende Blöcke schon da

Fehlerbehandlung bei Sliding-Window

- Fehlersymptom: Block geht verloren oder wurde (z.B. mittels CRC bzw. Prüfsumme) als fehlerhaft erkannt
  - Empfänger könnte zusätzlich auch ein NAK senden, wenn ein Block fehlerhaft ist oder er eine Lücke bei den Sequenznummern entdeckt

- Es gibt zwei prinzipielle Möglichkeiten:

1) "go back n":

  - Falls der Sender einen Block k wiederholt, sendet er auch alle schon versendeten Blöcke k+1, k+2... erneut
  - Frage: Wie reagiert der Empfänger?
  - Einfach, aber schlechter Durchsatz bei hoher Fehlerrate

2) "selective repeat":

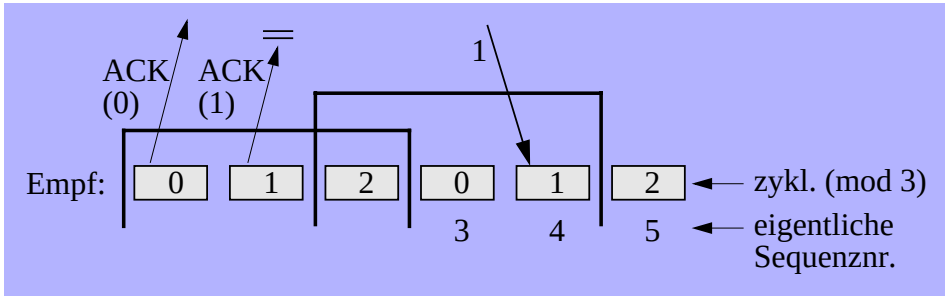
  - Nur fehlerhafter Block wird wiederholt
  - Beim Sender angestossen durch timeout, durch NAK oder durch Empfang eines anderen ACK als erwartet (je nach Variante)
  - Wenn eine Retransmission eintrifft, entsteht an der Dienstschnittstelle zu Layer i+1 u.U. ein "burst" von angebotenen Blöcken

Sliding-Window: Sequenznummern

- Grösse von Sende- und Empfangsfenster sei für nachfolgende Überlegung identisch

- Sequenznummern von Blöcken können zyklisch wiederverwendet werden (i.a. mod 2^i für gewisses i)
  - spart Platz in den Headern von Blöcken und ACKs
  - auf korrekte Interpretation von "<" und ">" achten!
  - Vorsicht diesbezüglich auch bei kumulativen ACKs

- Beim Verschieben des Fensters darf der Nummernbereich mit dem alten Fensterbereich nicht überlappen!



  - bei Verlust von ACK(1) würde (z.B. nach timeout des Senders) der alte Block 1 erneut gesendet werden
  - für den Empfänger wäre das nicht von 4 ("neue 1") unterscheidbar

- Es genügt ein Nummernbereich von 2 × Fenstergrösse
  - Denkübung: wieso?

- Wie gross sollten die Fenster sinnvollerweise sein?
  - geht auch eine Fenstergrösse 1?

Effizienz von Sliding-Window

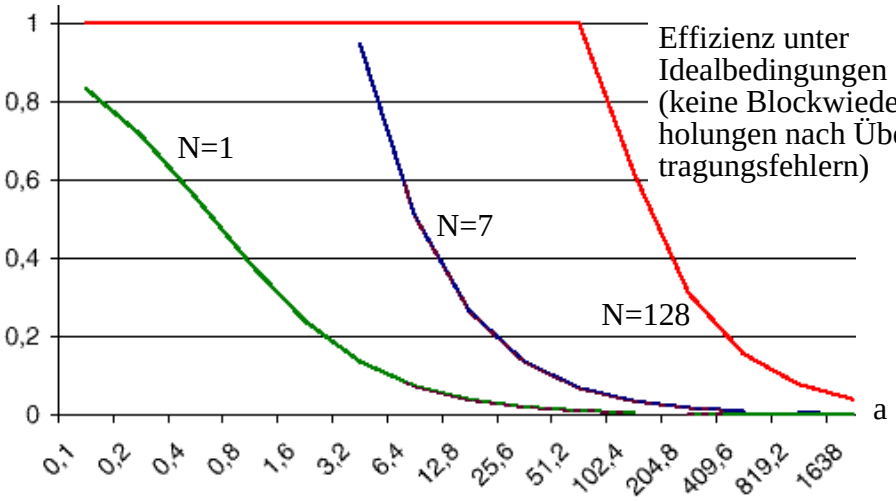
- Die Effizienz E = Durchsatz / Kapazität hängt vom Parameter a ab:

$$a = \frac{\text{Signallaufzeit}}{\text{Übertragungszeit}} = \frac{\text{Bandbreite x Entfernung}}{\text{Blockgrösse x Signalgeschwindigkeit}}$$

- Sei N die Fenstergrösse (identisch für Sender / Empfänger)

- Es gilt dann $E = \begin{cases} 1, \text{ falls } N > 2a+1 \\ N / (2a+1) \text{ sonst} \end{cases}$ (Denkübung: wieso?)

- Bemerkung: N = 1 entspricht Send-and-Wait!



Effizienz unter Idealbedingungen (keine Blockwiederholungen nach Übertragungsfehlern)

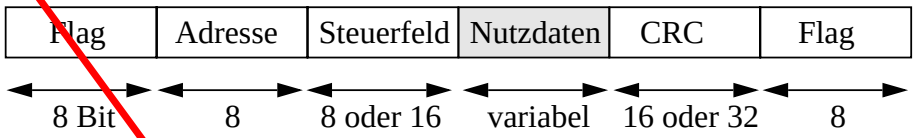
- Bei hoher Bandbreite bzw. grosser Entfernung sollte eine grosse Blockgrösse gewählt werden
- Kleine Fenstergrösse (z.B. 7) ist bei kurzen Entfernungen bzw. moderaten Datenraten ausreichend
- Hochgeschwindigkeits-WANs benötigen grosse Fenstergrössen (Denkübung: schadet eigentlich eine "Überdimensionierung"?)

HDLC

High Level Data Link Control

- Weit verbreitetes Protokoll der OSI-Ebene 2 ("data link layer"); genormt von der ISO
  - sowohl für Punkt-zu-Punkt-Verbindungen ("balanced configuration") als auch für Punkt-zu-Mehrpunkt mit einer Hauptstation ("primary") und mehreren Sekundärstationen
- Einfache Variante des Sliding-Window-Protokolls
  - für Flusssteuerung, Reihenfolgeerhalt und Fehlerbehandlung
  - Pakete mit Bitfehlern (CRC!) werden vom Empfänger ignoriert; Sender fragt bei ausbleibender Bestätigung nach ("polling"); Empfänger antwortet dann i.a. mit der nächsten erwarteten Sequenznr.
- Viele ähnliche praktisch relevante Protokolle (z.T. etwas andere Funktionalität, unterschiedl. Adressschemata etc.)
  - SDLC ("Synchronous..."; IBM-SNA-Architektur)
  - LAPB ("Link Acess Procedure, Balanced"): Teil des X.25-Paketvermittlungsstandards (X.25 wurde 1976 genormt)
  - LAPD ("..., D-Channel): Teil von ISDN
  - LAPF ("... for Frame-Mode Bearer Services"): Teil von "Frame Relay" (schnelles WAN-Paketvermittlungsprotokoll als Ersatz für X.25)
  - LLC ("Logical Link Control"): für LANs oberhalb der MAC-Ebene

HDLC-Rahmenformat



- Flag:
  - synchrone Datenübertragung; Flag 01111110 begrenzt den Rahmen
  - Bitmuster 01111110 wird i.a. auch in Pausen gesendet
  - bit stuffing, um Flag eindeutig zu halten
  - hunting-Modus des Empfängers, um Rahmenanfang zu finden

- Adressfeld
  - ohne Bedeutung bei direkter Punkt-zu-Punkt-Verbindung
  - “extended format” für längere Adressen
  - Broadcast-Adresse 11111111

- Steuerfeld
  - enthält Sequenznummer (bei Huckepackbestätigungen auch Nummer des letzten erhaltenen Rahmens) modulo 8 oder 128
  - diverse Bits zur Kennzeichnung von Kommandos

HDLC-Kommandos

Auswahl; es gibt noch einige mehr

- *Receive Ready* (RR): positive Bestätigung (mit Sequenznummer des zuletzt empfangenen Rahmens)
  - bereit, weitere Rahmen zu empfangen

- *Receive Not Ready* (RNR): wie RR, aber *nicht* bereit, weitere Rahmen zu empfangen
  - verwendet bei temporären Problemen des Empfängers
  - ein RR wird später (ggf. auf Nachfrage des Senders) verschickt

- *Reject* (REJ): Negative Bestätigung
  - wiederhole alle Rahmen ab der angegebenen Sequenznummer

- *Selective Reject* (SREJ)
  - wiederhole einen bestimmten Rahmen

- *Set Asynchronous Response Mode*: Verbindungsaufbau durch initiierende Station

- *Unnumbered Acknowledgement*: Bestätigt das letzte Kommando (z.B. Verbindungswunsch)

- *Disconnect*: Kann von beiden Seiten ausgelöst werden
  - Bestätigung durch ein Unnumbered Acknowledgement

Die meisten Kommandos können huckepack mit Informationspaketen in Gegenrichtung verschickt werden; falls keine Information in Gegenrichtung vorliegt aber auch mit eigenen “supervisory frames”.

Lokale Netze

- Charakteristiken
  - Ausdehnung max. einige km; mehrere Mb/s Übertragungsrate
  - typischerweise innerhalb eines Gebäudes (bzw. Etage; bzw. “Campus”)
  - typisch: Broadcastfähigkeit; bitserielle Übertragung
  - PCs, Workstations, Hosts, Drucker, Gateways zu WANs etc. sind flexibel und preiswert anschliessbar; Netz ist einfach erweiterbar

- “Typen” (Technologien, Protokolle...)
  - Ethernet (10 bzw. 100 Mb/s)
  - Token-Ring (4 oder 16 Mb/s)
  - Token-Bus
  - ...

  } - Übergänge / Koppelung möglich (Gateway, Router etc.)

  } - Ethernet (in verschiedenen Varianten) am weitesten verbreitet

- Höhere Datenraten z.B. mit:
  - Gigabit Ethernet (1000 Mb/s)
  - ATM (Asynchronous Transfer Mode: einheitl. WAN/LAN-Technik)

- Spezifische Anforderungen durch Multimedia-Daten
  - Audio (Sprachübertragung, Telefonieren über das LAN...)
  - Bilder, Animationen, Video
  - benötigt kurze Verzögerungen und Varianzen (--> “jitter”)

- Neuerdings auch drahtlos (“Wireless LAN”)
  - typisch: 30 - 100 m, ca. 10 Mb/s, 2.5 GHz
  - oder preiswerter: 5 - 10 m, ca. 700 kb/s (z.B. “Bluetooth”)
  - noch billiger: per Infrarot (wenige Meter, gerichtete Sichtverbindung)

Uni-Netzwerke Konkurrenz für UMTS

Die UMTS-Lizenzinhaber müssen offenbar um ihre milliardenschweren Investitionen bangen.

Einem Bericht der "Berliner Zeitung" zufolge könnten Netze aus extrem leistungsfähigen Minisendern den grossen Handy-Netzbetreibern das Geschäft mit dem künftigen UMTS-Multimedia-Mobilfunk verderben.

Neben den grossen geschlossenen Systemen von T-Mobile oder Vodafone könnte bald ein offenes, anarchisch wachsendes Netzwerk vieler kleiner Anbieter stehen, schreibt die Zeitung weiter.

Mit Unterstützung des Bundesforschungsministeriums errichteten derzeit beispielsweise 41 grosse deutsche Universitäten sogenannte drahtlose lokale Netzwerke (Wireless Local Area Network/ WLAN).

Über das erste campusweite System dieser Art an der Universität Rostock würden bald bis zu 1500 Studierende auf das Uni-Rechenzentrum und das Internet zugreifen.

Die Datenübertragungsgeschwindigkeit der WLAN-Netze sei dabei mit elf Megabit pro Sekunde bereits jetzt fünfeinhalb mal so schnell wie der künftige UMTS-Mobilfunk.

Was soll man von dieser Pressemitteilung halten, die in Deutschland und der Schweiz am 02.12.2000 erschien?

Das “klassische” Ethernet

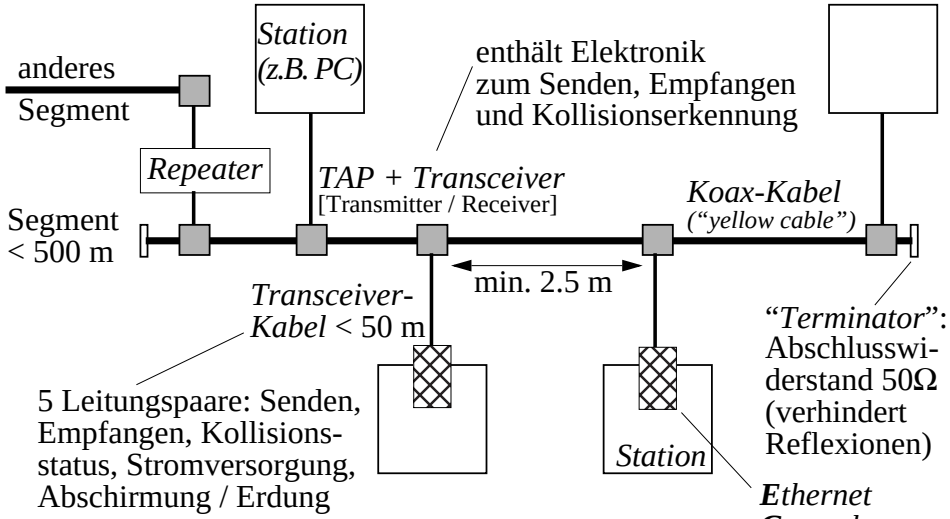
“Äther”
--> gemeinsames Medium
--> broadcast der Nachrichten

Carrier Sense Multiple Access

Collision Detection

- Bus mit dezentraler Zugangskontrolle (CSMA/CD-Prizip)
- Industriestandard seit 1980 (XEROX ab Mitte 1970)
- Genormt durch IEEE 802.3
- Relativ unempfindlich gegenüber Ausfall einer Station
- Preiswert, einfach ausbaufähig, wenig Administration

Klassische Konfiguration (“thick wire”; 10 Base 5):

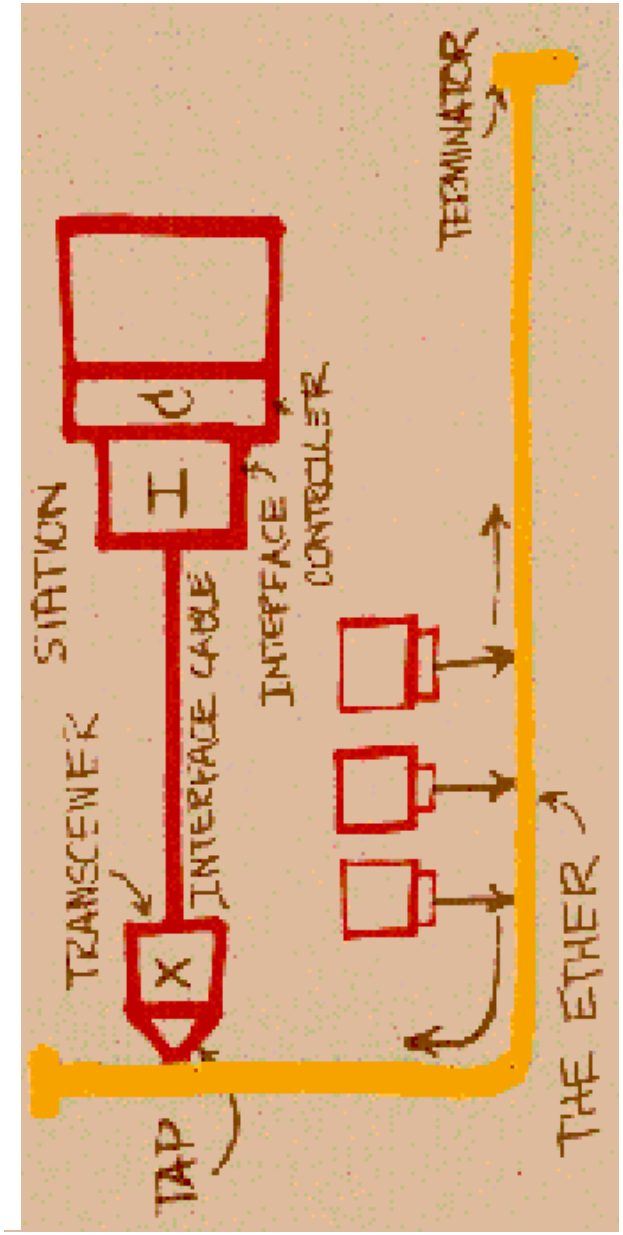


- Verbinden zweier Segmente durch *Repeater* (bidirektionale Signalverstärker) möglich
- Gesamtausdehnung max. ca. 2500 m

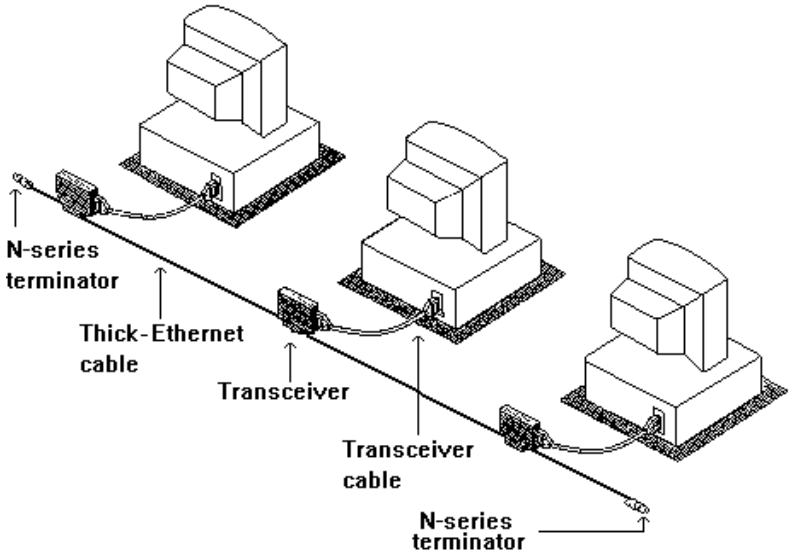
Das “klassische” Ethernet

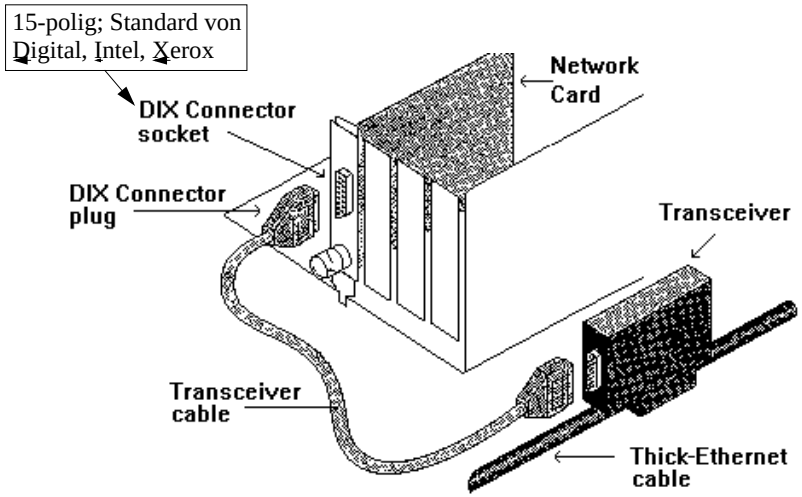
XEROX

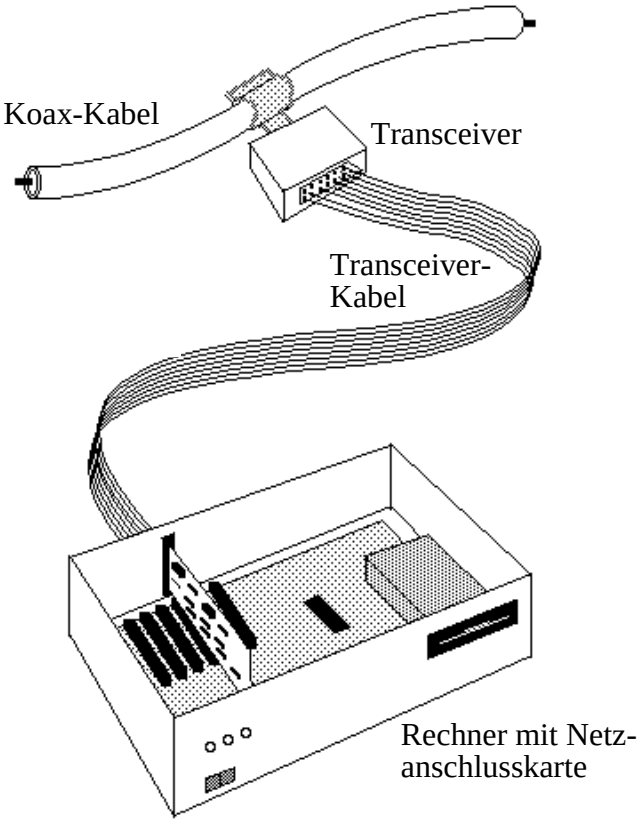
The diagram ... was drawn by Dr. *Robert M. Metcalfe* in 1976 to present Ethernet for the first time. It was used in his presentation to the National Computer Conference in June of that year. On the drawing are the original terms for describing Ethernet. Since then other terms have come into usage among Ethernet enthusiasts.



Das “klassische” Ethernet



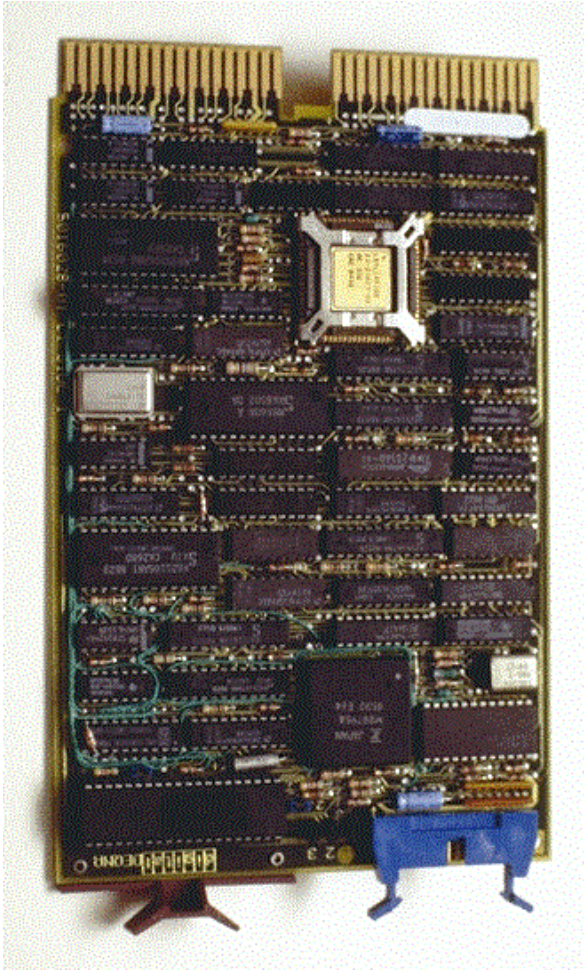




Ethernet yellow cable

Ethernet-Netzanschlusskarte

- Network Interface Card (“NIC”)
- Heute allerdings oft auf kleinerer PCMCIA-Karte



Transceiver-Kabel (“AUI-Kabel”) wird an den blauen Stecker angeschlossen

Plug and Play...

Betrachtungen beim Einbau einer Netzkarte

von Herbert Hrachovec

...Bloss wie kommt er ans Netz? Die Hilfsbereitschaft der Techniker ist schon lange überstrapaziert. Ihnen ist zu verdanken, dass der Betrieb mit den vorhandenen Geräten klappt. Sie kamen mit einem Zauberstab, murmelten einige Beschwörungsformeln - dann konnte man die erste elektronische Post verschicken. Das muss auch ohne fremde Hilfe gehen...

Zum Datenaustausch über ein Netz braucht der Computer ein Steckmodul, eine sogenannte Netzkarte, und Software, um mit ihr in Verbindung zu treten... Am einfachsten fällt es noch, die Karte physisch in den rechten Schlitz zu drücken, dann beginnt ein nervenaufreibendes Spiel. In rasanter Folge laufen Fehlermeldungen über den Bildschirm. Das Steckmodul und die Steuerung wollen sich nicht miteinander anfreunden. Stunden vergehen mit fruchtlosen Vermittlungsversuchen...