ZHAW CommTech Cheatsheet 2024

1 OSI

1.1 Dienste

1.1.1 Verbindungslos

Funktionsweise

- Funktioniert analog zur Briefpost
- Datagram wird mit voller Network-Layer-Adresse des Empfängers versehen und unabhängig von anderen Datenpaketen transportiert

Routingprozess

- Jeder Knoten muss Informationen über den Netzaufbau haben (für optimalen Weg)
- Informationen werden z.B. in Routing-Tabellen abgelegt

Folgen

- Nachfolgende Datenpakete können Vorgängige überholen und in vertauschter Reihenfolge beim Empfänger eintreffen

Beispiel

IP

1.1.2 Verbindungsorientiert

Verbindungsaufbau

- Eine **Verbindung muss aufgebaut oder eingerichtet werden**, bevor Nutzdatenpakete übertragen werden
- Transitknoten so eingestellt, dass sie die der Verbindung angehörigen Pakete erkennen und richtig behandeln können

Funktionsweise

- Übertragen Daten über eine Verbindung
- Verbindung vergleichbar mit Schlauch
- Reihenfolge einbehalten
- Pakete mit Verbindungsnummer im Header

1.1.3 Zuverlässig

- Kein Datenverlust
- Sicherung durch Fehlererkennung und Korrektur
- Beispiel: Text-Nachrichten

1.1.4 Unzuverlässig

- Möglicher Datenverlust
- Keine Sicherung
- Beispiel: Streaming

2 Übertragungsmedien

Grundlage der Kommunikation wie z.B Kabel, Lichtwellenleiter

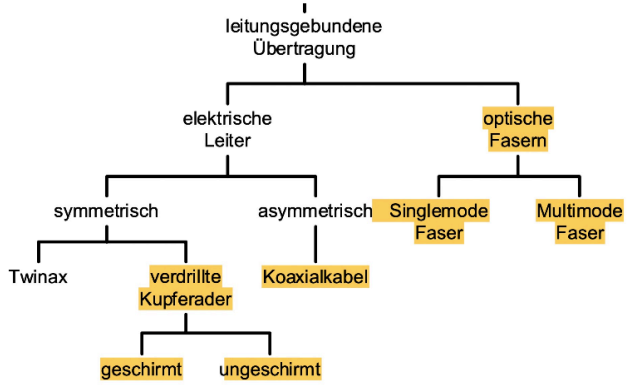

Figure 1: Kabel-Hierarchie

2.1 Signaldämpfung

$$\text{Signaldämpfung [dB]} = 10 * \log(\frac{P1}{P2}) \quad (1)$$

Eingangsleistung $P1$
Ausgangsleistung $P2$

$$\text{Signaldämpfung [dB]} = 20 * \log(\frac{U1}{U2}) \quad (2)$$

Dämpfung von 6dB = Leistungsabnahme um Faktor 4 (Spannungsabnahme um Faktor 2) Signaldämpfung steigt mit der Frequenz an

Je grösser die **Bandbreite (Hz)**, desto höhere **Datenraten (bit/s)** können übertragen werden

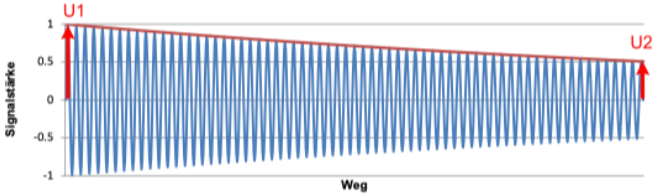

Figure 2: Kabelkategorien

2.2 Dämpfungsbelag

Dämpfung pro Distanz [dB/km] (für Übertragungsmedien aussagekräftiger)

2.3 Signal-Rausch-Verhältnis (SNR)

Gibt Auskunft darüber, wie stark sich das eigentliche Signal vom Rauschen/Störung abhebt

Je **kleiner** die Dämpfung ist, desto **grössere** Distanzen können erreicht werden

Senkt man die Bitrate (bei gleicher Dämpfung), können grössere Distanzen erreicht werden

$$\text{SNR} = 10 * \log(\frac{P_{\text{Signal}}}{P_{\text{Störung}}}) \quad \text{[dB]} \quad (3)$$

2.4 Koaxialkabel

- eignen sich gut für Übertragung von hochfrequenten Signalen
- im Vergleich zu paarsymmetrischen Kabeln einen kleinen Dämpfungsbelag
- unempfindlicher gegenüber elektromagnetischen Störungen
- dank geringer Dämpfung + höherer Störabstand können grössere Distanzen überbrückt werden als mit paarsymmetrischen Kabeln
- Für Punkt-Punkt-Verbindungen in Hochgeschwindigkeitsnetzten verwendet (Twinax Kabel)

2.5 Paarsymmetrische Kabel (Twisted Pair)

Es gibt zwei verschiedene Arten:

- Shielded Twisted Pair (STP)
- Unshielded Twisted Pair (UTP)

Nachteile

- anfällig auf elektromagnetische Störungen
- Übersprechen (Crosstalk) möglich

2.6 Kabelkategorien

Kategorie 1/2/3/4: geeignet für Telefon- und Modem-Leitungen
Kategorie 5: Weltweit akzeptierte Norm
Kategorie 6: Gigabit-Ethernet am Besten
Kategorie 7: 10 Gigabit-Ethernet geeignet
Kategorie 8: für Datacenter

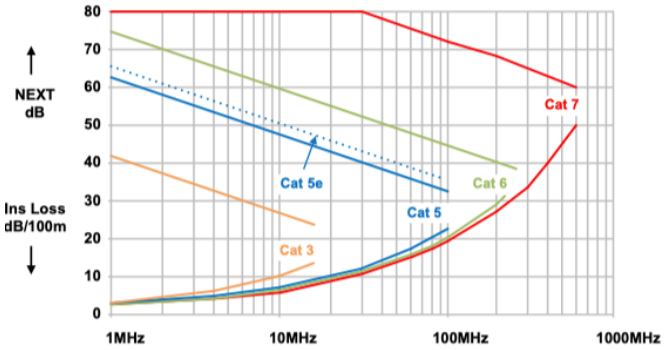

Figure 3: Kabelkategorien

2.7 Lichtwellenleiter

Bestehen aus Glas. Funktionsprinzip basiert auf Totalreflexion.
Kernglas aus mit **hoher optischen Dichte**.
Mantelglas umschlossen mit **geringerer optischen Dichte**

Vorteile

- Vollständige Unempfindlichkeit gegen elektromagnetische Störungen
- Kleine Signaldämpfung und somit grosse Distanzen
- Grosse Bandbreiten und somit grosse Übertragungsraten

Dämpfung entsteht durch Absorption im Lichtwellenleiter (variiert stark mit der Wellenlänge (Farbe)).

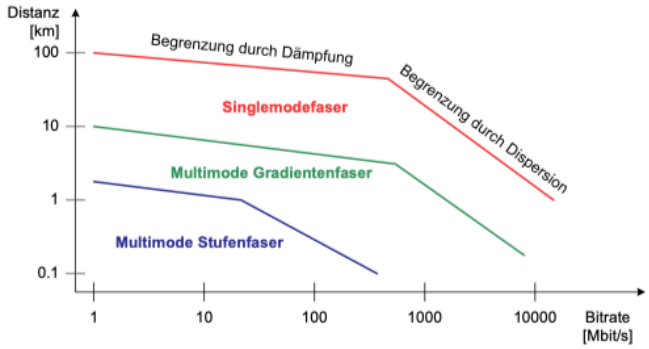

Figure 4: Kabelkategorien

Es gibt folgende Typen von Lichtwellenleitern:

- Monomodefaser
- Multimode-Gradientenfaser
- Multimode Stufenfaser

2.7.1 Multimode Glasfaser

Stufenfaser höherer Delay skew

Gradientenfaser kürzerer Delay skew
Klassifizerung von Multimode Glasfasern OM1-OM4. OM1 und OM2 für LED-basierte Anwendungen, OM3 und OM4 für VCSELs

2.7.2 Monomode Glasfaser

- haben wesentlich grössere Bandbreite
- grössere Distanzen als Multimode Glasfaser

3 Physical Layer (Layer 1)

3.1 Aufgaben

- sorgt für die ungesicherte Übertragung eines Bitstroms

3.2 Kopplung der Kommunikationspartner

3.2.1 Arten der Kommunikation

- **Simplex**: es ist nur ein Kanal in eine Richtung vorhanden
- **Halbduplex**: es ist nur ein Kanal vorhanden, der abwechslungsweise für die eine oder andere Richtung benutzt wird
- **Vollduplex**: es ist für jede Richtung je ein Kanal vorhanden

3.3 Arten der Verbindung

- **Punkt-Punkt**: direkte Verbindung zweier Kommunikationspartner; PC und Drucker
- **Shared Medium**: mehrere Partner verkehren über das gleiche Medium; LAN

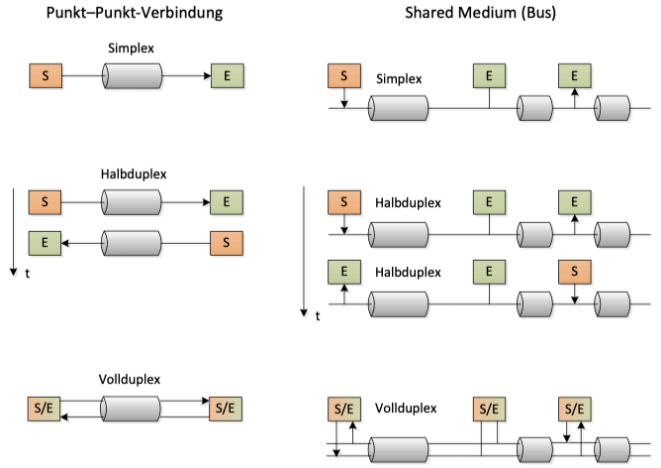

Figure 5: Kopplung

3.4 Übertragungsverfahren

Folgende Übertragungsarten:

- **Parallele Übertragung**: mehrere Bits gleichzeitig über mehrere (parallele) Leitungen übertragen
- **Serielle Übertragung**: Bits werden zeitlich nacheinander über eine Leitung übertragen

3.4.1 Serielle asynchrone Übertragung

→ **asynchron**, weil kein Takt übertragen wird

- Sender und Empfänger besitzen eigene, unabhängige Taktquellen mit annähernd gleicher Frequenz
- Empfänger justiert seinen Übertragungsrahmen bei jedem übertragenen Zeichen von Neuem
- Bitrate, die Anzahl der n Datenbits und Anzahl Stoppbits muss zwischen Sender und Empfänger abgemacht sein

Senderseite

- Sender schickt für jedes Zeichen ein Startbit, die vereinbarte Anzahl Datenbits und zum Abschluss die Stoppbits

- **LSB Datenbit wird zuerst übertragen**

Empfängerseite

- Tastet das Signal mit einem Vielfachen der Bitrate ab, bis er den 0-Pegel des Startbits erkennt
- Während Übertragung dürfen Takte nicht mehr als eine halbe Bitzeit T abweichen
- Überprüfung ob Stoppbit vorhanden ist

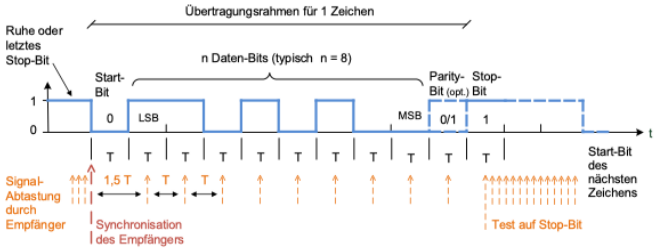

Figure 6: Serielle Asynchrone Übertragung

3.4.2 Serielle Synchrone Übertragung

- Bitsynchrone Arbeitsweise erlaubt hohe Datenrate + ist effizient, da keine Start- und Stoppbits benötigt werden
- Nebst Datensignal muss auch dessen Takt übertragen werden

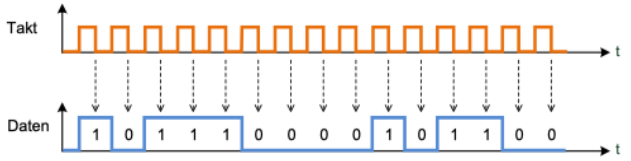

Figure 7: Serielle Synchrone Übertragung

3.5 Leitungscodes

Übertragung des Takts: über separate Leitung oder Codierungsverfahren

4 Data Link Layer (Layer 2)

4.1 Aufgaben

- stellt der höheren Schicht eine gesicherte Übertragungsstrecke zwischen zwei direkt miteinander verbundenen Knoten zur Verfügung
- Verpacken der vom Network Layer erhaltenen Datenblöcken in Frames + Auspacken der empfangenen Frames
- Frame-Erkennung
- Fluss-Steuerung (Flow Control)

4.2 Framing

4.2.1 Framing bei asynchroner Übertragung

- Daten werden nur bei Bedarf übertragen
- Übertragungsstrecke ruht, wenn keine Daten zur Übertragung ankommen
- Startbit des ersten Bytes signalisiert den Empfängern den Beginn eines Frames
- Zwischen den Frames muss Pause von mindestens einer Zeichendauer eingehalten werden
- Zweck der Pause: Synchronisation + Fehlerdetektion

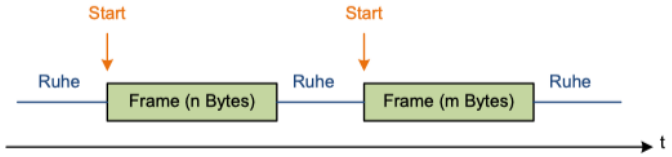

Figure 8: Framing bei Asynchroner Übertragung

Aufbau des Frames bei asynchroner Übertragung

Figure 9: Frame-Aufbau bei Asynchroner Übertragung

4.2.2 Framing bei synchroner Übertragung

- Frames werden ohne Unterbruch gesendet
- Leer-Frames werden übertragen, wenn keine zu sendenden Daten ankommen
- Empfänger muss kontinuierlich Bitstrom analysieren
- Frame-Grenzen müssen erkannt werden können
- Byte-Grenzen müssen erkannt werden können

Erkennung von Anfang und Ende eines Frames

- Verfahren mit Start- und Ende-Flags wird eingesetzt

Figure 10: Framing bei Synchroner Übertragung

Aufbau des Frames bei synchroner Übertragung

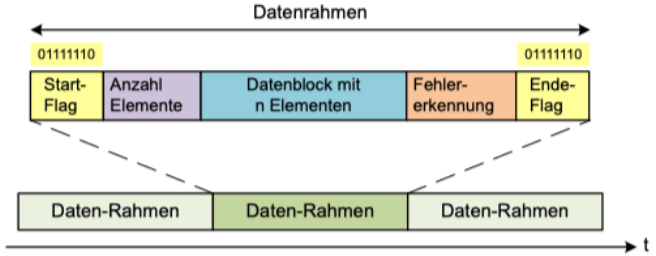

Figure 11: Frame-Aufbau bei synchroner Übertragung

Bitstopfen

Es wird als Start- und Ende-Flag ein spezielles Bitmuster verwendet. (Meistens: 01111110)

Damit das Bitmuster der Flag im Frame nicht aus Versehen vorkommt, wird eine zusätzliche Null gestopft (nach 5 aufeinanderfolgenden Einsen).

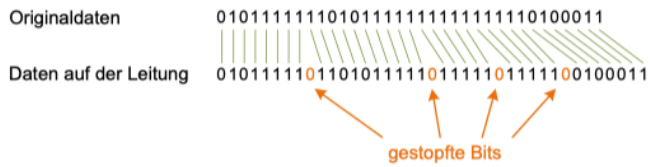

Figure 12: Stopfbit

4.2.3 Wahl der Frame-Länge

Grundsätzlich gilt:

Je länger die Frames desto besser wird die **Nettobitrate**

$$\text{Nettobitrate} = \text{Bruttobitrate} * \frac{\text{Nutzdaten}}{\text{Nutzdaten} + \text{Header}} \quad (4)$$

Probleme bei langen Frames:

- Blockieren die Übertragungsstrecke für die Dauer der Übertragung: Einfluss auf Jitter
- Lange Frames haben grössere Wahrscheinlichkeit fehlerhaft zu sein

$$\text{Optimale Frame-Länge} \approx \sqrt{\frac{H}{p_e}} \quad (5)$$

4.3 Fehlererkennung + Korrektur

4.3.1 Backward Error Correction

→ Empfänger kann Fehler erkennen, aber nicht korrigieren (Retransmission notwendig)

4.3.2 Forward Error Correction

→ Empfänger kann den Fehler nicht nur erkennen, sondern auch direkt korrigieren

4.4 Medium Zugriff

Regelung des Zugriffs auf das gemeinsame Übertragungsmedium (Shared Medium)

4.4.1 Deterministische Verfahren

Master/Slave-Verfahren

- Master fragt zyklisch jeden Sklaven an
- Angefragter Slave antwortet sofort und liefert vorhandene Daten an den Master
- Vorteil: Anschaltung der Slaves ist sehr einfach
- Nachteil: Transferzeit grösser zwischen zwei Slaves, da über Master läuft
- Ausfall des Masters führt zu Systemausfall

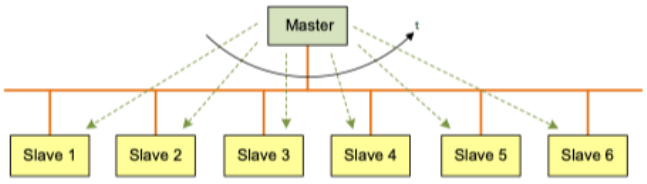

Figure 13: Master/Slave-Verfahren

Token Passing

- Zwischen gleichberechtigten Knoten wird in einer festgelegten Reihenfolge eine Sendeberechtigung (Token) weitergereicht; Geräte senden nur, wenn sie den Token haben
- Jeder Knoten ist innerhalb eines bestimmten Zeitintervalls einmal sendeberechtigt (→ deterministisch)
- Vorteil: Kommunikation erfolgt Peer-to-Peer
- Nachteil: Beim Startup relativ aufwändig + Token kann verlorengehen

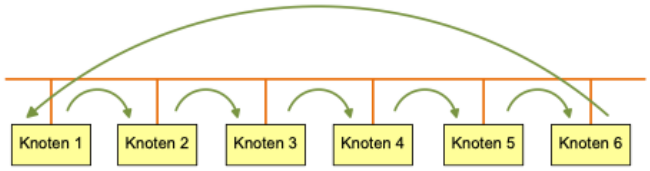

Figure 14: Token Passing

4.4.2 Nicht-deterministische Verfahren

Carrier Sense Multiple Access

- Jede Station ist gleichberechtigt und darf jederzeit auf den Bus zugreifen, nachdem kontrolliert wurde, dass dieser frei ist (Carrier Sense)
- Ist Bus besetzt, wird gewartet und später wieder versucht (Multiple Access)
- Vorteil: Benötigt keinen Master + keine Konfiguration
- Nachteil: Kann nicht voraussgesagt werden, wann ein Knoten sendeberechtigt ist

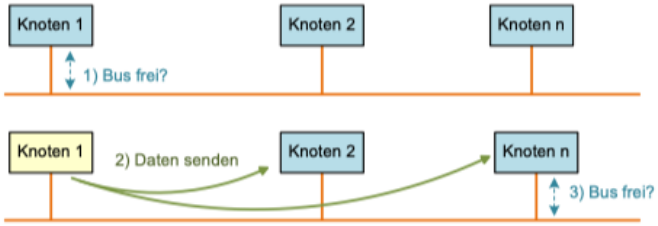

Figure 15: Carrier Sense Multiple Access

5 Lokale Netzwerke (LAN)

5.1 Topologien

- **Bustopologie**: Knoten sind passiv an das Übertragungsmedium angeschlossen; nur im Sendefall aktiv
- **Linientopologie**: Jeweils zwei benachbarte Knoten sind verbunden
- **Ringtopologie**: Jeder Knoten auf zwei Wege erreichbar (Redundanz), Loops müssen abgefangen werden
- **Sterntopologie**: Medium der Bustopologie wird in einem zentralen Verteiler konzentriert (mit Hub / Switch)
- **Baumtopologie**: Hierarchische Kombination aus Sterntopologie

5.2 Übertragungsarten

- **Unicast**:
  - Transfer von einer Quelle zu genau einem Ziel
  - Im Idealfall: Unicast-Pakete werden nur genau zum Empfänger übertragen
- **Broadcast**: Transfer an alle Knoten eines Netwerks (belastet alle Knoten)
- **Multicast**: Transfer an eine bestimmte Gruppe von Knoten (mit Multicast Adresse der Gruppe → erfordert Management + Hilfsprotokoll)

5.3 Aufbau des Ethernet-Frames

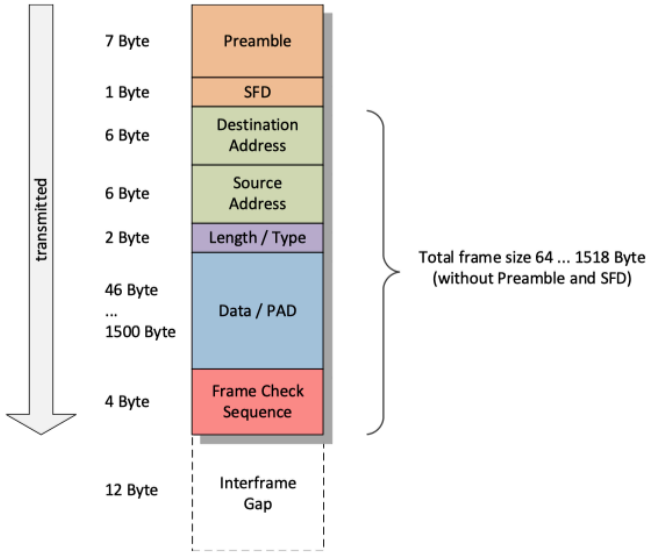

Figure 16: Ethernet-Frame

Immer zuerst das LSB gesendet!

- **Preamble + Start Frame Delimiter (SFD)**: Preamble ist 7 Byte-Feld mit (1010101...) (damit entsteht für jedes Bit eine Flanke im manchester-codierten Signal). **SFD** signalisiert mit 11 Bitfolge das Ende der Synchronisation + damit Anfang des Frames

- **Destination Address**: Adresse des Ziel- bzw. Empfangsknotens. (MAC Address → eindeutige Adresse eines Ethernet Geräts

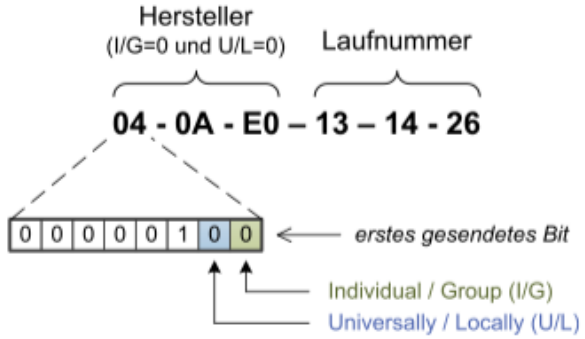

Figure 17: Ethernet-Frame

- **Source Address**: Adresse des Quell- bzw. Sendeknotens
- **Length/Type**: 2 Bytes (höherwertige Byte zuerst); Length gibt die effektive Anzahl von Bytes im folgenden Datenfeld an (ohne Padding); Type gibt an, was für Protokolle im Datenfeld enthalten sind
- **Data/Pad**: Datenfeld; 0 - 1500 Datenbytes; Falls kleiner als 46, dann mit Nullbyte Padding aufgefüllt (min Frame-Länge von 46)
- **Frame Check Sequence (FCS)**: Feststellung von Übertragungsfehler bei Frame-Übermittlung
- **Interframe Gap (IFG)**: kein Feld des Frames, sondern bezeichnet minimale zeitlicher Abstand zwischen zwei sich folgenden Frames (min 96 Bit-Zeiten → Physical Layer)

5.4 Repeater (Signalverstärker)

5.4.1 Konfigurationsregeln für Repeater

- **Round Trip Delay**: Beliebiger sendender Knoten muss Kollision erkennen können, so lange er noch am Senden ist
- **Interframe Gap Shrinkage**: Beim Durchlaufen eines Repeaters werden verlorene Bits der Preamble regeneriert → Preamble wird länger + Interframe Gap kleiner

5.4.2 Collision Domain

Bereich des Netzwerks, in dem eine Kollision erkannt werden kann.
Collision Domain darf nur *halb* so gross sein, wie die Ausdehnung des kürzesten Frames.
(Grund: Jam-Signal (nach Kollision) muss noch beim senden im Sendeknoten ankommen; Worst-Case: Knoten B beginnt kurz bevor Frame von A ankommt noch zu senden)
Falls Netz grösser als doppelte Ausdehnung des Signals ⇒ Knoten A erkennt Kollision nicht (**Late Collision**)

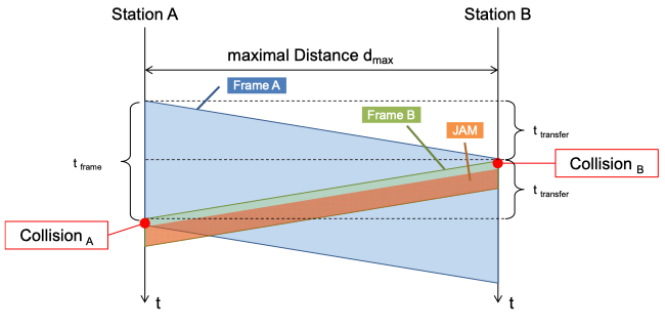

Figure 18: Kollision Frame

Bedingung zur Erkennung einer Kollision durch Knoten A:

$$t_{frame} > 2 * t_{transfer} \quad (6)$$

Maximale Ausdehnung eines Segments:

$$d_{max} < \frac{1}{2} \frac{\text{Framesize}_{min}}{\text{Bitrate}} * C_{Medium} \quad (7)$$

Knoten A Kollisionserkennung mit Repeater, genau dann wenn:

$$t_{frame} > 2(\sum t_{transfer} + \sum t_{forwarding}) \quad (8)$$

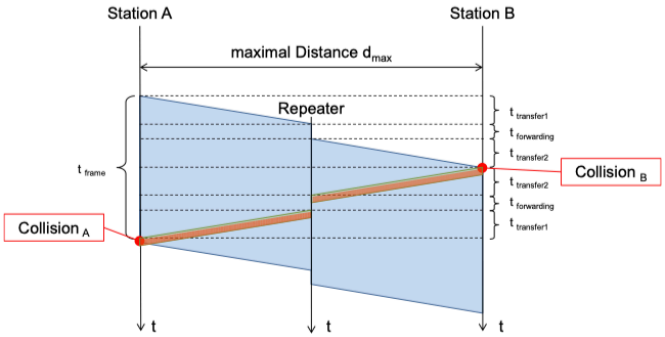

Figure 19: Maximale Ausdehnung mit Repeater

6 Switched LAN + Ethernet-Technologien

6.1 Bridges

- teilt Collision Domains auf (koppelt mehrere Collision Domains miteinander)
- arbeitet auf Data Link Layer
- prüft empfangene Frames und leitet basierend auf Zieladresse an andere Ports weiter
- Frame-Inhalt wird nicht verändert
- sind transparent (Knoten bemerken Bridges nicht)
- Promiscuous Mode (nehmen sämtliche Frames entgegen)
- funktionieren ohne Konfiguration
- Vergrössern Sicherheit gegenüber Totalausfällen
- Netz sollte möglichst lokal sein und nicht immer über Bridge gehen

6.1.1 Address Learning

Filtering Database

- Erlernen Adressen mit Filtering Database
- Filtering Database enthält für bekannte Mac-Adressen den jeweiligen Bridge-Port
- Unbenutzte Einträge werden nach einer gewissen Zeit gelöscht (*default*: 600s)
- Weiterleitung ist damit gezielt und belastet das Netzwerk weniger
- **Frame Flooding** falls Destination-Adresse nicht in Filtering Database enthalten ist → Frame wird auf allen Ports (ausser Eingangsport) ausgegeben
- **Broadcast-Frames** werden auch an alle Knoten gesendet
- **Broadcast-Domain**: alle LAN-Segmente, die mit Bridges gekoppelt sind

6.1.2 Multiport Bridges (Switches)

- übliche Netzwerkkomponenten
- problemlose Koppelung von zwei und mehr Segmenten

6.2 Virtuelle LANs

- grosse Netze können in unabhängige logische Netze aufgeteilt werden
- jeder Switch-Port kann einem beliebigen virtuellen LAN zugeordnet werden
- bilden eine **Broadcast Domain**
- Knoten merken nicht, dass sie an einem VLAN angeschlossen sind

6.2.1 Funktionsweise

- Ethernet-Frames bekommen **VLAN-Tag** (802.1Q-Header); VLAN-Identifier im Tag ermöglicht Zustellung des Frames zum richtigen VLAN
- Switch fügt VLAN-Tags hinzu und entfernt sie auch wieder
- **Priority Code Point (PCP)** ermöglicht Priorisierung von Frames (Quality of Service)
- **Discard Eligibility Indicator (DEI)**; Frames mit diesem gesetzten Bit werden bei Engpässen zuerst verworfen

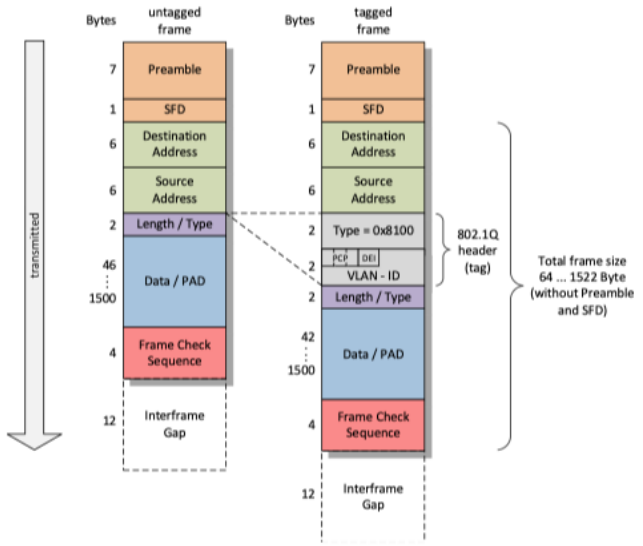

Figure 20: Frame-Aufbau mit VLAN-Tagging

6.2.2 Redundanz Protokolle

Frames sollen nicht endlos im Netzwerk herumkreisen

(Rapid) Spanning Tree Protocol

1. Spanning Tree Protocol sperrt von den redundanten Pfaden alle bis auf einen
2. Wurzel des Baumes (Root) wird bestimmt und ausgehend von dieser ein Baum aufgespannt, der alle Knoten genau einmal verbindet
3. Basiert auf Austausch von **Bridge Protocol Data Unit (BPDU)**-Protokollen
4. BPDU werden von Switches **nicht** weitergeleitet (spezielle Multicast-Adressen)
5. BPDU teilt jeder Knoten seinen Nachbarn mit, welches (seines Wissens) die beste **Root-Bridge** ist + zu welchen Kosten er diese erreichen kann
6. Auswahl der Root-Bridge beruht auf **Bridge Identifier** (besteht aus wählbarer Priorität + MAC-Adresse)
7. Bridge mit **tiefsten Identifier** wird Root Bridge
8. Pfadkosten ausgehend von der Root aufaddiert (Kosten für Verbindung ist einstellbar)

9. Bridges senden alle 2 Sek BPDU; falls nicht ⇒ Netzwerk wird neu konfiguriert

Ablauf RSTP beim Einschalten

1. Jeder Knoten sieht sich selber als Root-Bridge mit Kosten 0
2. Erkennt Knoten, dass Nachbar eine bessere Bridge-ID hat, übernimmt dieser dessen Root-Identifier (teilt diese mit)
3. Falls redundante Pfade vorhanden ⇒ Jeder Switch bestimmt das Port mit den tiefsten Pfadkosten (zur Root) als **Root-Port**
4. **Designated Port**: Port der nächsten Bridge, das von den Root-Port zur Root führt

7 Network Layer

7.1 Router

- ähnliche Funktion wie Bridges, arbeiten jedoch auf dem Network Layer
- im Gegensatz zu Transparent-Bridges empfangen Router nur Pakete, die (auf Data-Link-Layer) **direkt** an sie adressiert sind; d.h. müssen an den Router mit dessen MAC-Adresse geschickt werden
- Weiterleitung erfolgt anhand einer **Network Layer Adresse** und *nicht* aufgrund **Data Link Layer Adresse (z.B. MAC-Adresse** ⇒ globale, unabhängige Adressierung kann eingeführt werden
-

7.1.1 Bridging vs Routing

- Router benutzen immer den optimalen Pfad
- Netzbereiche können logisch voneinander getrennt werden
- Router als Barriere für die auf Layer 2 verwendeten Broadcast-Meldungen
- Router benötigen Konfigurationen

7.1.2 Routing

Zwei Hauptaufgaben:

- **Routing**: optimalen Weg (Route) im Netz bestimmen
- **Forwarding**: Weiterleitung der Datenpakete entlang dieser Routen
- Router müssen Topologie kennen
- Router können statisch angegeben oder dynamisch bestimmt werden
- Um dynamisch immer die besten Routen zu bestimmen, müssen die verschiedenen Router miteinander kommunizieren (über **Routing Protocols**)

Routing-Tabellen

- Basis für Routing sind Routing-Tabellen
- Für Weiterleitung durchsucht der Router seine Routing-Tabelle von oben nach unten

Routing-Tabelle

Netzadresse	Netzmaske	Port	Gateway
160.85.18.0	255.255.255.240	eth1	(direkt)
160.85.19.0	255.255.255.0	eth2	(direkt)
160.85.16.0	255.255.254.0	eth0	(direkt)
default	0.0.0.0	eth0	160.85.16.1

Figure 21: Routing-Tabelle

Flaches Routing

- vorzugsweise in den zentralen Teilen

Hierarchisches Routing

- typischerweise in der Netz-Peripherie eingesetzt
- erlaubt, die Routing-Tabellen möglichst klein zu halten
- ein Host, dessen Ziel nicht im selben Subnet ist, schickt dies einfach an seinen Default-Router (gateway) → dieser kennt auch nur die direkt angeschlossenen Subnets und verfährt analog

7.2 IP Adressierung

7.2.1 Aufbau + Darstellung der Adresse

- 32 Bit grosse Zahl
- Generell Big-Endian-Konvention (**MSB zuerst**)
- 32 Bits jedoch von links nach rechts nummeriert (MSB ist 0, LSB 31)

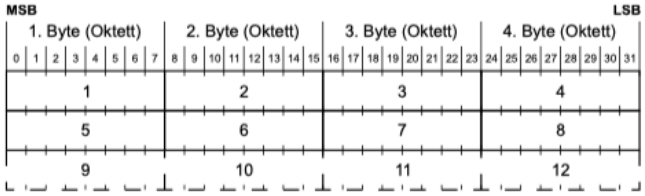

Figure 22: Byte-Wertigkeit IP

7.2.2 Subnets und Subnet-masks

Classless-Routing

- erlaubt flexible Zuteilung von Adressenräumen in Zweierpotenzschritten
- benötigt **Subnetmask** für flexibles einstellen (legt Grenze zwischen Network und Interface-address (Host-Address) fest)

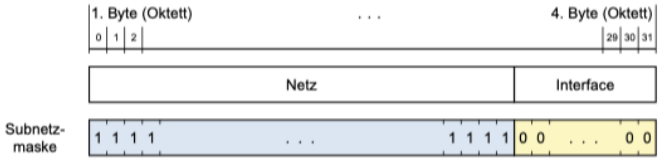

Figure 23: Subnetmask

Subnetmask:

- Bit auf 1 ⇒ Bit gehört zur Network-Address
- Bit auf 0 ⇒ Bit gehört zur Interface-Address

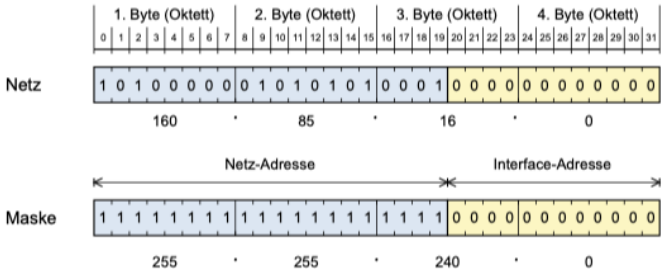

Figure 24: Subnet Masking

Subnetmask kann **keine** Lücken aufweise, deswegen sind immer nur einer der 9 Werte möglich:

- 255 (1111 1111)
- 254 (1111 1110)
- 252 (1111 1100)
- 248 (1111 1000)
- 240 (1111 0000)
- 224 (1110 0000)
- 192 (1100 0000)
- 128 (1000 0000)
- 0 (0000 0000)

7.2.3 Network- und Broadcast-Address

Network-Address

- tiefste Adresse im Netz (besteht binär betrachtet aus lauter Nullen)
- reserviert und darf nicht an Knoten gegeben werden

Broadcast-Address

- höchste Adresse im Netz (besteht aus lauter Einsen)
- damit können alle Knoten eines Netzes gemeinsam adressiert werden

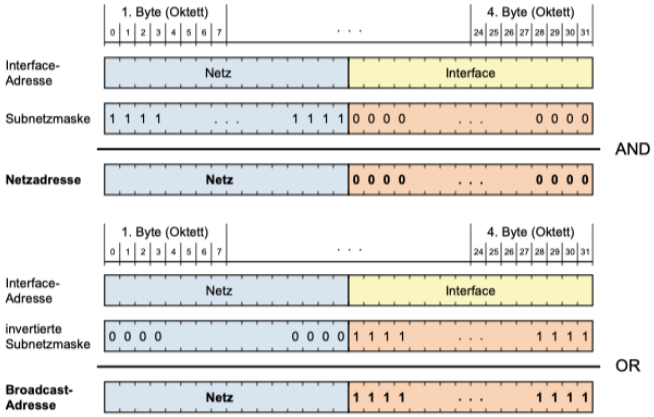

Figure 25: Anwendung Subnetmask

Classful-Routing

- ursprünglich aus technischen Gründen wurden **Netzklassen** definiert
- **Netzklassen** haben fixe, definierte Grössen
- Vorteil: anhand der ersten Adress-Bits kann die Klasse (und Grösse) eines Netzes bestimmt werden
- benötigte keine Subnetmasks (effizient)
- in den letzten Jahren an seine Limiten angelangt

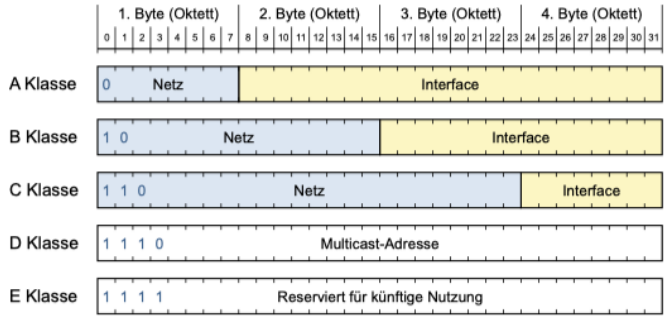

Figure 26: Netzklassen

Klasse	Adressbereich	Anzahl Netze	Anzahl Knoten
A	1.0.0.0 – 127.255.255.255	128	16'777'214
B	128.0.0.0 – 191.255.255.255	16'384	65'534
C	192.0.0.0 – 223.255.255.255	2'097'152	254
D	224.0.0.0 – 239.255.255.255	Multicast-Adressen	
E	240.0.0.0 – 255.255.255.255	Reserviert für künftige Nutzung	

Figure 27: Netzklassen Adressbereiche

7.3 IP Protokoll

IP-Datagram besteht aus Header und den eigentlichen Nutzdaten

7.3.1 IP Header

- **Version** (4 Bit): Version des IP-Headers
- **Internet Header Length (IHL** (4 Bit): Länge des IP-Headers
- **Type of Service** (8 Bit): für Quality of Service (Vorrangigkeit, Verzögerungen, Durchsatz, Zuverlässigkeit, Reserviert) (Verbesserung eines dieser Parameter kann zu Verschlechterung der anderen führen)
- **Total Length** (16 Bit): gesamte Länge des Datagrams (inklusive Header + Nutzdaten); max Grösse: $2^{16} - 1 = 65'535$ Bytes
- **Identification** (16 Bit): erlaubt fragmentierte Datagrams wieder zusammenzustellen, da alle Fragmente den gleichen Wert haben
- **Flags** (3 Bit): beinhaltet zwei Kontrollflags für Fragmentierung
  - Bit 0: reserviert (muss 0 bleiben)
  - Bit 1: 0/1 → May / Don't Fragment
  - Bit 2: 0/1 → Last / More Fragments
- **Fragment Offset** (13 Bit): gibt an, wo innerhalb des Datagrams ein Fragment hingehört (erste Fragment hat Offset 0) (max: 8192 Fragmente)
- **Time to Live (TTL)** (8 Bit): gibt die verbleibende Zeit in Sekunden an, die das Datagram noch im Internet-System verbleiben darf (Loop-Control)
- **Protocol** (8 Bit): gibt das übergeordnete Protokoll an, das im Nutzdatenteil des Datagrams verwendet wird
  - 1: ICMP
  - 6: TCP
  - 17: UDP
- **Header Checksum** (16 Bit): Prüfsumme (muss in jedem Router neu überprüft und gebildet werden)
- **Source Address** (32 Bit): IP-Adresse des Hosts
- **Destination Address** (32 Bit): IP-Adresses des Zielhosts
- **Options + Padding** (variable): für Erweiterungen (Padding, um Internet-Header auf ganzzahliges Vielfaches von 32 Bit zu füllen (mit Nullen)

7.3.2 Fragmentation und Reassembly

Fragmentation

- Maximal Übertragungsgrösse eines Pakets vom Data Link Layer Technologie abhängig
- Maximale Übertragungseinheit als **Maximum Transfer Unit (MTU)** bezeichnet
- Beispiel: Ethernet Frame Datenteil ist max 1500 Byte; IP Datagrams können bis 64k Byte gross sein ⇒ IP Datagram muss in einzelne Frames fragmentiert werden (hier 44)
- Jedes Fragment bekommt eigenen IP-Header

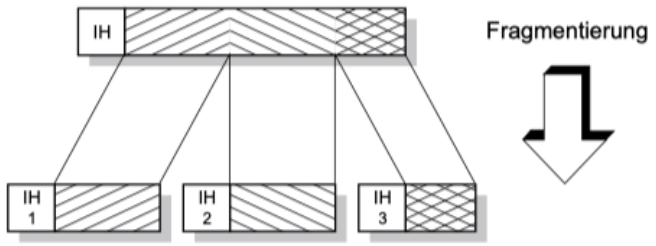

Figure 28: Fragmentierung IP-Datagram

Reassembly

- Wiedervereinigung der Fragmente zum Datagram (erst beim Zielhost nach IP-Standard); dadurch werden Zwischenrouter entlastet und Routen können dynamisch geändert werden
- Empfänger kann mit Identification-Feld die Fragmente zusammensetzen (alle zusammengehörenden Fragmente haben gleiche Identification)
- Reassembly benötigt die Felder: Identification, Fragment Offset, Last / More Fragments, Total length

Ablauf

1. Check ob Datagramm fragmentiert ist: Fragment Offset + More Fragments beide 0; Falls nicht mache weiter mit Schritt 2
2. Buffer-Identifikation bestimmt (Datenstruktur zur Sammlung der Fragmente)

7.4 Kapselung und Umsetzung IP-Adresse in Hardware-Netzadresse

Damit IP-Datagramm übertragen werden kann muss Folgendes gemacht werden:

- IP-Datagramm in Netzwerk-spezifische Frame verpackt werden
- Logische IP-Adresse muss in Hardware-Adresse umgesetzt werden

7.4.1 Kapselung IP-Datagramm

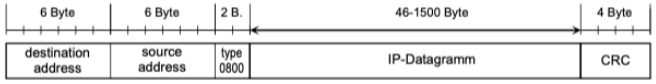

Figure 29: IP-Datagramm Kapselung

7.4.2 Adressauflösung (Address Resolution)

Versenden eines IP-Datagramms erfolgt auf dem Data-Link-Layer.
Der Frame (Ethernet) braucht jedoch destination address (Hardware address). IP-Protokoll Paket muss deswegen in hardware address umgewandelt werden
Adressauflösung ist immer netzwerkspezifisch

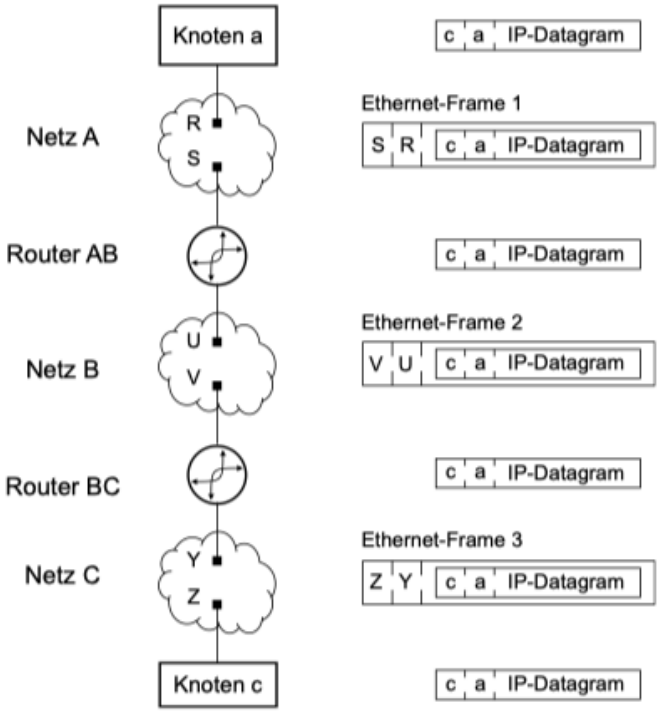

Figure 30: Kapselung und Netzübergang

7.4.3 Address Resolution Protocol (ARP)

Ablauf

Knoten muss Protokoll-Adresse in Hardware-Adresse umsetzen:

1. Knoten sendet an Broadcast-Adresse ein ARP-Request-Paket (alle Knoten im lokalen Netz erhalten dieses)
2. Knoten mit dieser Adresse schickt ARP-Reply-Paket an den anfragenden Knoten (direkt; nicht über Broadcast)

ARP-Cache

Damit nicht immer neu gesendet werden muss, speichert jeder Knoten nach Anfragen die Antwort in einem ARP-Cache (nach gewisser Zeit werden Einträge gelöscht)

ARP-Request-Pakete dürfen nicht durch Bridges gefiltert werden (wie bei allen Broadcast-Protokollen → erhöht Netzwerkbelastung

7.5 Reverse Address Resolution Protocol (RARP)

- Löst das umgekehrte Problem → IP-Adresse anhand MAC-Adresse (z.B. beim Booten)
- Neu gebooteter Host kann Broadcast-Anfrage mit RARP senden und erhält vom RARP-Server eine Antwort (durchsucht Konfigurationsdateien)
- Router leiten RARP Broadcast Anfragen nicht weiter, deswegen braucht jedes Subnet einen RARP-Server

7.6 Internet Control Message Protocol (ICMP)

- Sind in einem IP-Datagram gekapselt
- Entwickelt, um Meldungen zu verschicken (z.B. Fehlermeldungen)
- Normalerweise melden sie Fehler in der Verarbeitung und Weiterleitung von IP-Datagrams
- Damit kein endloses "Ping-Pong" aus ICMP-Meldungen entsteht, werden keine ICMP-Meldungen über ICMP-Meldungen erzeugt; zusätzlich nur Meldung über das erste Fragment (bei fragmentierten Datagrams)

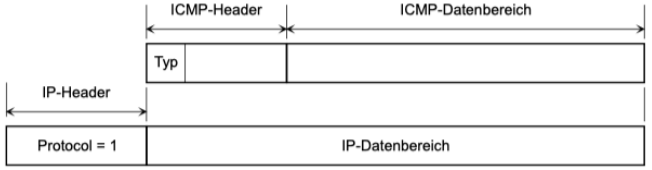

Figure 31: Kapselung einer ICMP-Meldung

7.6.1 ICMP-Fehlermeldungen

- **Destination Unreachable**: Datagram kann nicht an das Endziel befördert werden
- **Source Quench**: Router sendet dies, falls Buffer voll ist (muss weitere ankommende Datagrams verwerfen); Router sendet dies an die Hosts von welchen er die Datagrams verwerfen musste
- **Redirect**: Falls Host ein Datagramm an falschen Router schickt (Datagram für entferntes Netz) und der Router feststelllt, dass es an falschen Router geschickt wurde; Router fordert redirect auf. (Route ändern)
- **Time Exceeded**: Falls "Time-To-Live (TTL)" eines Datagrams auf null ist oder Timer abläuft bevor alle Fragmente angekommen sind

- **Parameter Problem**: Router oder Host findet einen ungültigen Wert im Header eines Datagrams
- **Echo Request/Reply**: Request kann an jeden Host geschickt werden; Reply muss zurückgeschickt werden
- **Timestamp Request/Reply**: Ähnlich wie echo; Sender und Empfänger senden noch Zeit mit
- **Information Request/Reply**

8 Transport Layer

- bildet die Schnittstelle zwischen Betriebssystem und Applikationen
- Verteilt die eintreffenden Daten auf die verschiedenen Applikationen (**Demultiplexen**)

8.1 Kapselung

- Applikationsdaten werden mit einem UDP oder TCP Header versehen; dies nennt man **User Datagramm (UDP)** oder **Segment (TCP)**
- Applikationsdaten werden anschliessend mit IP-Header versehen, wobei dessen Protocol-Feld aussagt, ob es sich um UDP- oder TCP-Paket handelt

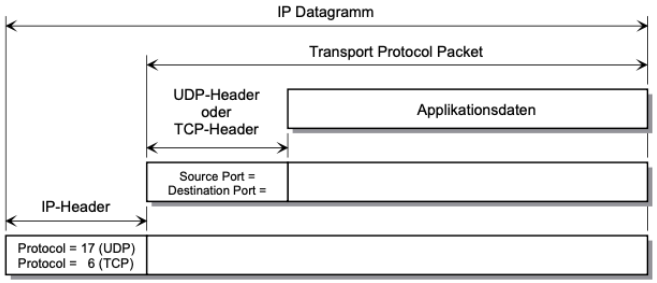

Figure 32: Kapselung Applikationsdaten durch Transport- und IP-Layer

8.2 Vorgang

1. Ethernet-Frame kommt an
2. Type-Feld des Frames wird überprüft; falls 800h, Weiterreichung des Internet-Datagramms ans IP-Modul
3. IP-Modul untersucht Protocol-Feld im Datagramm-Header; gibt das Transport-Protocol-Packet an das entsprechende, höhere Protokoll-Modul weiter
4. Anhand Destination Ports im TCP (oder UDP) Header; richtige Applikation erhält die Daten

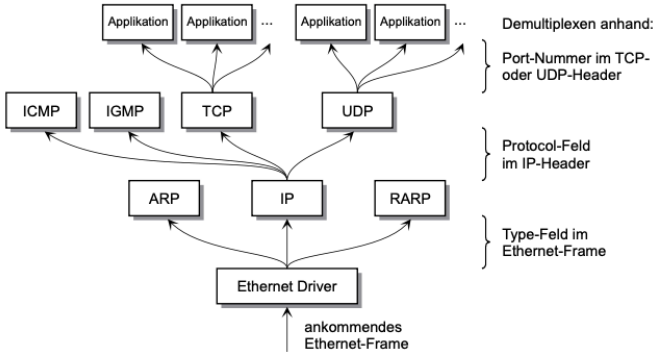

Figure 33: Demultiplexen: Bsp.: Ethernet-Schnittstelle

8.3 User Datagram Protocol (UDP)

- verbindungslos
- unzuverlässig
- Transport-Protokoll

8.3.1 UDP-Header

- **UDP Source Port + Destination Port**: enthalten je ein Word (16 Bit); fürs Verteilen der eintreffenden Datagramme auf die entsprechenden Prozesse
- **UDP Message Length**: gibt Länge (in Byte) des Transport-Protocol-Packets an
- **Checksum**: dient zur Fehlererkennung

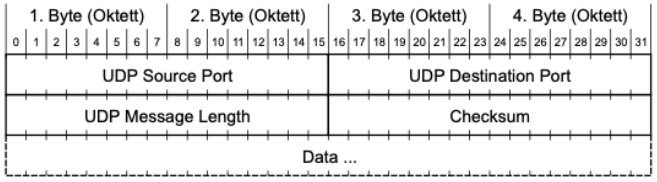

Figure 34: UDP-Header

8.3.2 Ports

Ports werden für UDP und / oder TCP reserviert (3 Bereiche)

8.4 Transmission Control Protocol (TCP)

- verbindungsorientiert (immer genau zwei Endpunkte)
- zuverlässig (ohne Datenverlust und in der richtigen Reihenfolge)
- Transport-Protokoll
- Vollduplexübertragung (Daten können in beide Richtungen fliessen)
- End-to-End-Protocol (bietet Verbindung direkt von einer Anwendung eines Hosts zu einer Anwendung eines entfernten Hosts)
- Verbindungen sind **virtuell** (werden von Software hergestellt)

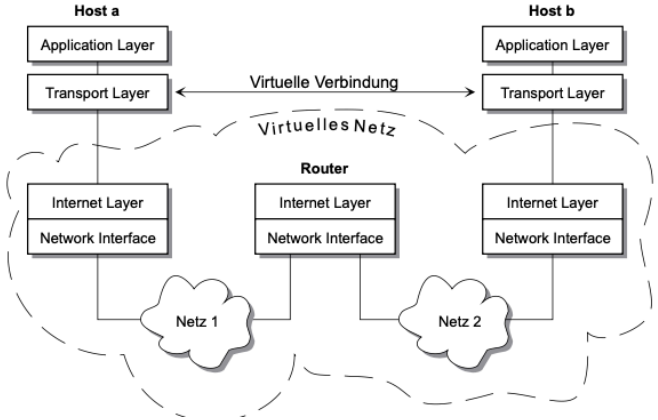

Figure 35: TCP

8.5 TCP-Header

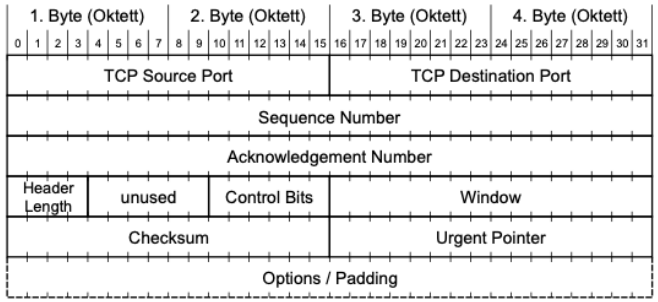

Figure 36: TCP-Header

- **TCP Source und Destination Port**: definiert, welches Anwendungsprogramm auf dem empfangenden Host die Daten erhalten soll
- **Sequence Number**: (Ausgangsdaten) Sequenznummer der im Segment enthaltenen Daten (ordnet Segmente anhand der Sequenznummer, die ausser der Reihe ankommen + für Berechnung der Acknowledgment-Nummer)
- **Acknowledgment Number**: (Eingangsdaten) definiert die Sequenzummer der zu empfangenden Daten
- **Data offset**: gibt an, wo die Daten beginnen
- **Control Bits**: bezeichnet eine Reihe von Flags für Verbindungsaufbau- und -abbau
- **Window**: zeigt die noch verfügbare Buffer-Grösse an
- **Checksum**: Fehlererkennung
- **Urgent Pointer**: Wo in den Daten Informationen mit hoher Priorität enthalten sind
- **Options**: oft beim Verbindungsaufbau verwendet

8.5.1 Zuverlässigkeit von TCP

- Zuverlässigkeit von TCP wird vor allem durch **Retransmission** erreicht
- Vor jeder Datenübertragung startet der Sender einen Timer
- Trifft fehlerfreies Segment beim Empfänger ein, sendet dieser eine Bestätigung (**Acknowledgment**)
- Läuft Timer vor der Ankunft ab, überträgt der Sender das Segment (unaufgefordert) nochmals
- Falls Segment fehlerhaft ist, erfolgt *keine* Rückmeldung vom Empfänger

8.5.2 Adaptive Wartezeit

Problem: Fixe Zeiten bis zur Retransmission sind problematisch, da je nach Ort, Übertragungsart etc. die Ankunft des Segments beeinflussen
Retransmission Time-Out: Wartezeit bis zur Neuübertragung

- Zu kurze Wartezeiten beeinflussen Netz unnötig, zu lange Wartezeiten führen zu Verzögerungen in der Übertragung

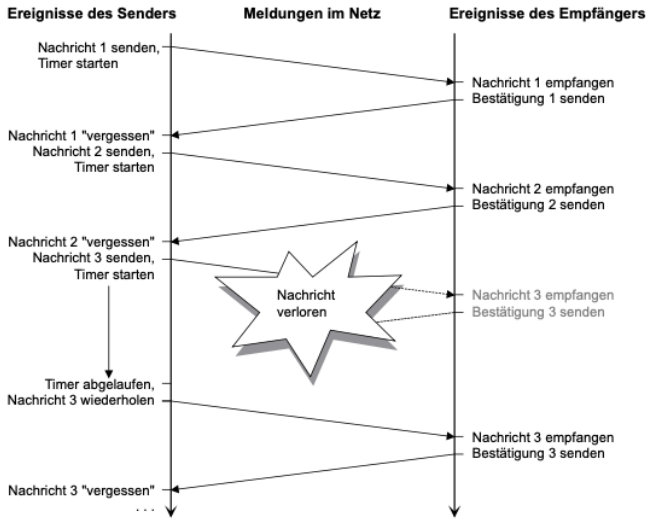

Figure 37: asdf

Lösung: Adaptive Neuübertragung

Round-Trip Time / Round-Trip Delay:
Gesamtzeit vom Senden der Nachricht und Empfangen der Bestätigung

- ausreichend lange warten, dass nur ein Paketverlust zu einem Timeout führt (möglichst keine Wartezeiten)
- Berechnung über dem gewichteten Mittelwert der Round-Trip-Times (misst jede RTT bei jeder aktiven Verbindung) und passt den Retransmission Time-Out an

8.5.3 Flow-Control

Problem: Nicht alle Computer laufen mit gleicher Geschwindigkeit ⇒ Data Overrun bei zu viel Daten mit kleinerer Datenaufnahme

Lösung: **Sliding Window** (wird von TCP verwendet)

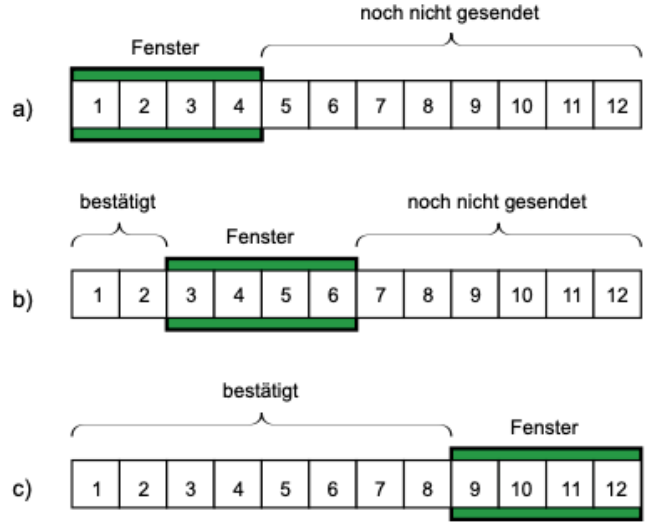

Figure 38: Sliding Window

- Sender und Empfänger einigen sich auf eine fixe Fenstergrösse (entspricht der maximale Datenmenge, die vor Ankunft einer Bestätigung gesendet werden kann)

Vergleich Stop-And-Wait vs Sliding Window

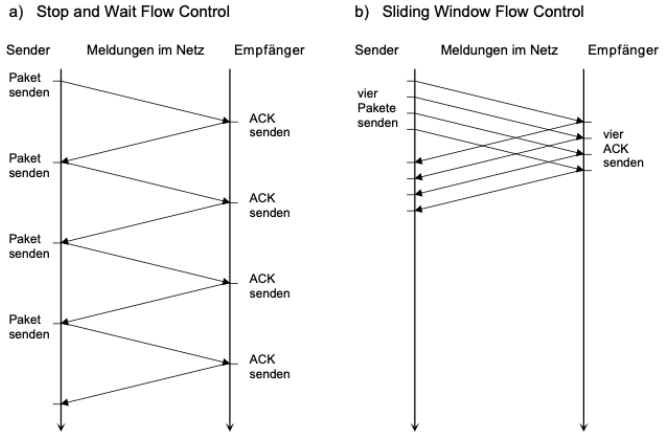

Figure 39: Stop-And-Wait vs Sliding Window

Sliding Window bei TCP

- Bestätigung des Empfängers enthält zudem auch das restliche Window (**Window Advertisement**) ⇒ aktueller

Stand des Windows

- Solange Daten so schnell gelesen werden, wie ankommen, ist das Window eine positive Zahl in der Bestätigung
- Falls Buffer vom Empfänger voll ist, gibt es ein **Zero Window** zurück; Sender muss solange warten, bis Window wieder positiv ist

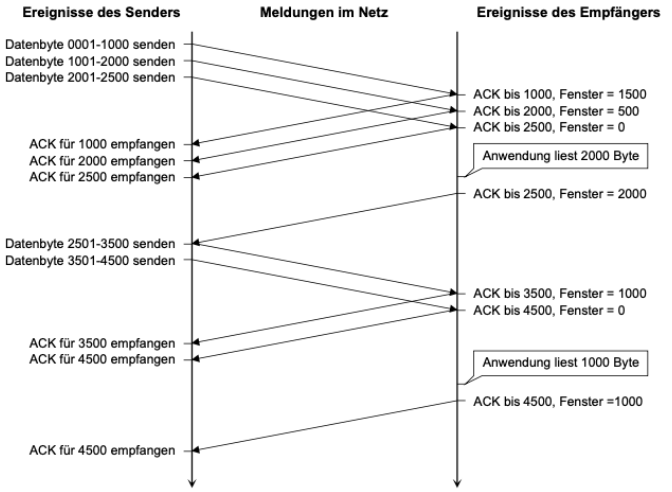

Figure 40: TCP-Segmente mit Window von 2500 Byte

8.5.4 Zuverlässiger Verbindungs-Aufbau und -Abbau

TCP verwendet **3-Way-Handshake**

Verbindungsaufbau

Nachrichten für Verbindungsaufbau haben **SYN**-Flag

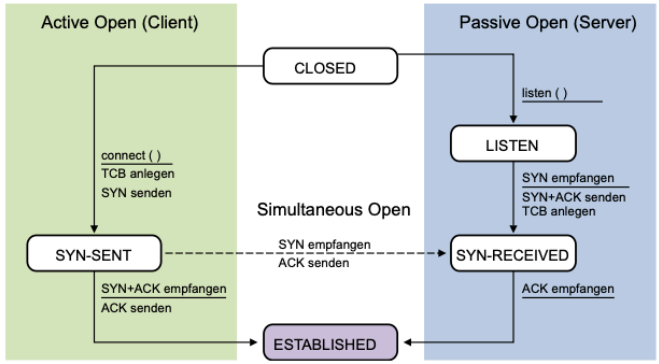

Figure 41: TCP Verbindungsaufbau

1. Vor eigentlichem Verbindungsaufbau: TCP-spezifische Datenstruktur anlegen (Transmission Control Block (**TCB**)); enthält Informationen zur Verbindung
2. Beide Kommunikationspartner generieren eine zufällig gewählte Sequence-number (eindeutige Verbindung) (a; client und b; server)
3. Client sendet SYN-Sent mit der Sequence-number
4. Server antwortet mit SYN mit seiner Sequence-number + und bestätigt (ACK) die Verbindung indem er die Sequence-number vom client + 1 im Acknowledge-Feld zurückgibt
5. Client muss Verbindung bestätigen (ACK), indem er Sequence-number vom server + 1 zurücksendet
6. Alle folgenden (Daten)-Segmente verwenden als Offset der Sequenz- und Acknowledge-Nummern a + 1 und b + 1

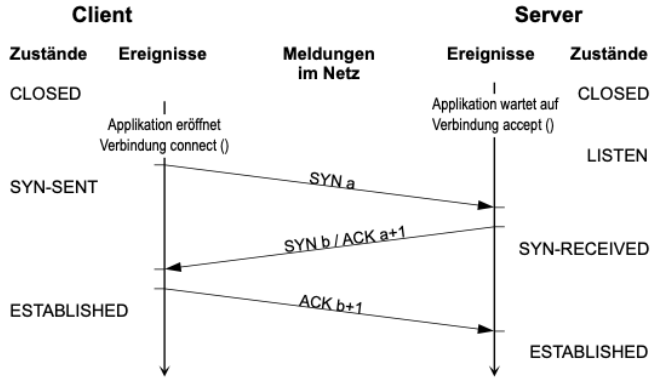

Figure 42: 3-Way-Handshake

Verbindungsabbau

- Beliebige seite (client oder server) schickt **FIN**-Segment
- Damit ist die Verbindung zuerst **half-closed**
- Ressourcen (TCB) dürfen erst freigegeben werden, wenn Verbindung in *beide* Richtungen geschlossen wurde

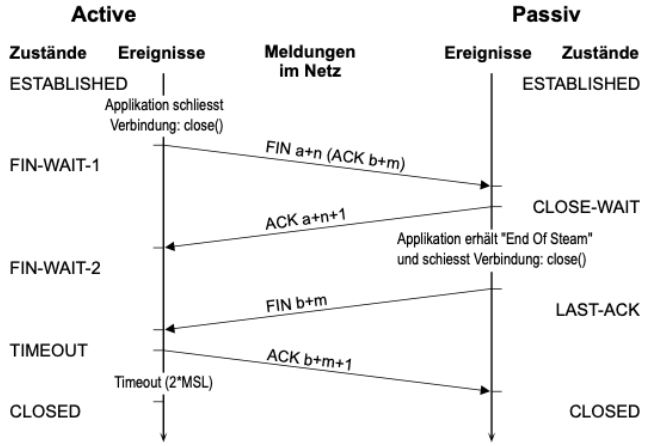

Figure 43: TCP Verbindungsabbau

8.5.5 Congestion Control

- Jeder Sender hat *zwei* Fenster **Sliding Window** und **Congestion Window**

- Effektive Fenster ist das *Minimum* dessen, was Sender für angemessen hält und was der Empfänger für angemessen hält
- Bsp.: Empfänger sieht 8Kb für angemessen, jedoch Sender weiss, dass mehr als 4Kb das Netz überlasten ⇒ effektive Fenster: 4Kb

9 Netzwerk-Applikationen und Protokolle

9.1 Domain Name System (DNS)

- Erlaubt das Übersetzten von Hostnamen in IP-Adressen und umgekehrt
- Besteht aus hierarchischen **DNS Name Space**
- DNS wird auf einer grossen Anzahl Name-Server verteilt betrieben

9.1.1 Domain Name Space

- Hierarchische Datenbank des DNS
- Zur Verwaltung des Internets
- **Root/Root Domain**: Wurzelknoten eines DNS-Baumes
- Jeder Knoten im Baum trägt Namen (bis zu 63 Zeichen lang)
- Jeder Host hat einen eindeutigen Namen (**Fully Qualified Domain Name**)
- **Alias**: Zeiger von einer Domäne auf eine andere

DNS-Domänen werden normalerweise mit unterschiedlichen Ebenen bezeichnet:

- **Top Level Domain (TLD)**: Domänen auf oberster Ebene: direkt dem Wurzelknoten untergeordnet
- **Second Level Domain**: Domänen auf zweiter Ebene
- **Third Level Domain**: Domänen auf dritter Ebene

Verwaltung von Domänen

- Name-Server; Programm, das Daten über den Domain Name Space speichert + Informationen zu DNS-Abfragen liefert
- Name Space kann in Zonen aufgeteilt werden; Name-Server ist normalerweise für eine oder mehrere Zonen verantwortlich
- Falls für Verwaltung einer Domäne an einen Name-Server delegiert wird ⇒ Name-Server automatisch auch für Subdomänen verantwortlich
- Jede Zone muss von einem **Master-Name-Server** bedient werden
- **Slave-Name-Server** erhalten Zonendaten über Transfers vom Master-Name-Server
- Dadurch wird **Redundanz** gewährleistet

9.1.2 DNS-Abfragen auswerten

- **Resolver**, welcher bei DNS-Abfragen eine DNS-Anfrage erzeugt + an Name-Server sendet (Resolver ist auf Client-Side)
- Resolver verarbeitet Antwort vom Name-Server und leitet Information an das anfragende Programm weiter
- Resolver-Anfragen werden von DNS-Servern ausgeführt

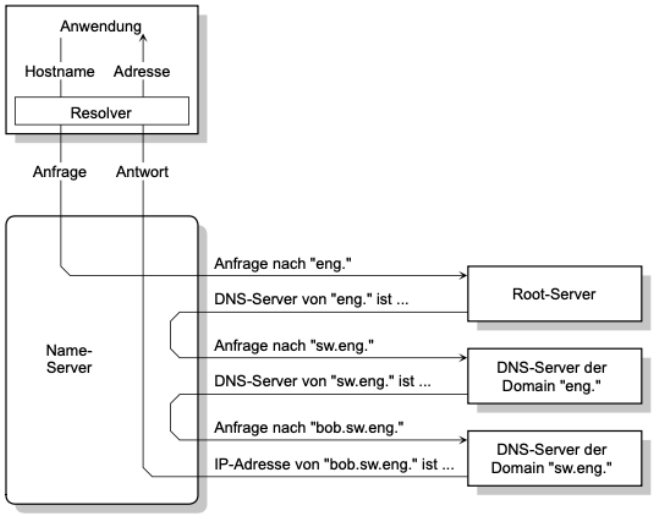

Figure 44: DNS-Abfrage

9.2 Bootstrap Protocol (BootP)

- Knoten ohne Laufwerke müssen beim Ladevorgang IP-Adresse bekommen
- Protokoll zur Regelung des Ladevorgangs
- Nachteile: Umgebungen mit grosser Anzahl Konten (mit nur wenig gleichzeitig aktiven Knoten) will man IP-Adressen gemeinsam nutzen können (→ nicht möglich mit BootP

9.2.1 Dynamic Host Configuration Protocol (DHCP)

- verwendet BootP-Protokoll
- Ermöglicht Zuweisung einer *dynamischen* IP-Adresse
- Rechner kann seine IP-Konfiguration von einem Server beziehen
- Administrator muss einen Adressbereich im Subnet reservieren + auf DHCP Server konfigurieren
- Bei entsprechender Anfrage eines Clients such DHCP Server in diesem Adressbereich freie IP-Adresse + vergibt diese mit zeitlicher Beschränkung **(Lease-Time)**
- Wenn Lease-Time abläuft, muss sie vom DHCP-Client erneuert werden

9.3 Network Address Translation (NAT)

- erlaubt, ein privates IP-Netz mit lokalen IP-Adressen hinter einer einzigen globalen IP-Adresse zu verstecken (Bypass für ausgeschöpfter IPv4-Adressraum)

9.3.1 Port Mapping

- mit NAT muss Kommunikation (TCP oder UDP) vom lokalen Netz aus initiiert werden → von aussen kann nicht auf Knoten mit lokaer Adresse zugegriffen werden
- für Dienste nach aussen muss auf Gateway die entsprechende Portnummer statisch die (lokale) IP-Adresse des jeweiligen Servers konfiguriert werden
- Alle einkommenden Pakete mit bestimmter konfigurierter Port-Nummer werden an entsprechenden Server geschickt
- Pro Dienst/Port-Nummer **kann es nur einen lokalen Server geben**

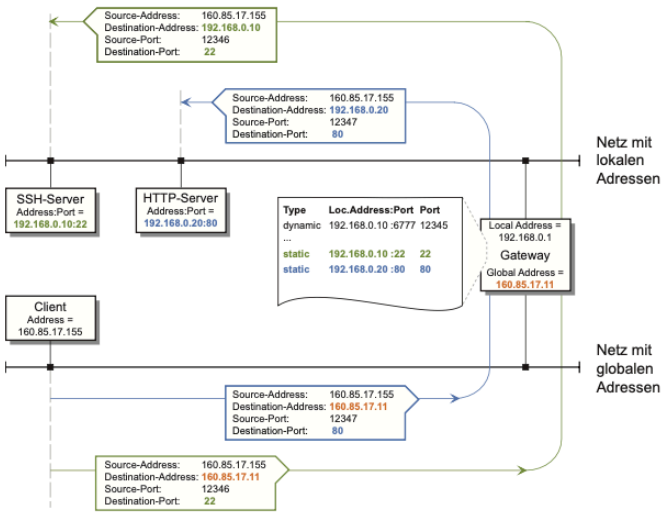

Figure 45: Port Mapping

9.3.2 Probleme mit NAT

- Verletzt eigentlich das OSI-Layer (Network-Funktion greift auf den Transport-Header zu und verändert diesen)

9.4 Trivial File Transfer Protocol (TFTP)

- Neben File Transfer Protocol (FTP) ein weiteres Protokoll zur Datenübertragung
- Einfaches, aber zuverlässiges File Transfer Protocol, welches zB diskless Systemen dazu dient, Betriebssystem Image vom Server zu beziehen

9.5 Simple Mail Transfer Protocol (SMTP)

- dient dem Versand von E-Mail Nachrichten (POP und IMAP dem Empfang von E-Mail Nachrichten

9.6 Hypertext Transfer Protocol (HTTP)

- Erlaubt den Zugriff auf verteilte Dokumente, die mittels Uniform Resource Locator (URL) eindeutig adressiert werden

ZHAW CommTech Cheatsheet 2024

Physical Layer - Cooking Recipe

Gegeben:

- Symbol-Dauer: $T_{Sym} = 66.67 * 10^{-6} s$
- Bandbreite: $B = 8kHz$
- Rauschabstand: $SNR = 40dB$

Kompatibilität Symboldauer

Aufgabe

Ist die angegebene Symboldauer kompatibel mit dem, was Ihnen zum Physical Layer vorgestellt wurde?

→ Überprüfe Nyquist:
Maximale Symbolrate f_s (Baud) ist gemäss Nyquist gleich der doppelten Bandbreite B (Hz) des Übertragungskanals

$$f_s = 2B \tag{9}$$

Damit:

$$\text{Baudrate} f_s = \frac{1}{T_{Sym}} = \frac{1}{66.67 * 10^{-6}} = 15kBaud$$

$$15kBaud < 2 * 8kHz \Rightarrow \text{OK}$$

Maximale Kanalkapazität (Shannon)

Aufgabe

Welche Bitrate für einen Teilkanal ist maximal möglich?

→ Shannon

$$C_S = B * ld(1 + \frac{S}{N}) \tag{10}$$

$$ld(x) = \frac{\log(x)}{\log(2)} \tag{11}$$

$$\frac{S}{N} = 10^{(\frac{SNR}{10})} \tag{12}$$

Damit:

$$10^{(\frac{SNR}{10})} = 10^{(\frac{40}{10})} = 10'000$$

$$C_S = 8000 * ld(1 + 10'000) C_S \quad = 8000 * 13.3 \frac{Bit}{s} = 106.3 \frac{kBit}{s}$$

Maximale Anzahl Bits pro Symbol

Aufgabe

Wie viele Bits pro Symbol (Bedingung: 2er-Potenz) dürfen maximal verwendet werden?

$$\text{Anzahl Bits pro Symbol} = \frac{\text{Bitrate}}{\text{Baudrate}} \tag{13}$$

Damit:

$$\frac{106.3 \frac{kBit}{s}}{15kBaud} = 7.08 \frac{Bit}{Symbol} \rightarrow 7 \frac{Bit}{Symbol}$$

Anzahl Symbole

Aufgabe

Wie viele verschiedene Werte müssen die Leitungssymbole dann annehmen können?

$$\text{Anzahl Symbole} = 2^{\text{Anzahl Bits pro Symbol}} \tag{14}$$

Damit:

$$2^7 = 128$$

Switching

Gegeben:

- Netwerk mit Hubs und Switches (Bridges)
- Aging Time (nach welcher ein Eintrag in der Filtering Database gelöscht wird)

Vorgehen:

- Filtering Database einer Switch merkt sich immer nur **die MAC-Adresse der Geräte von ankommenden Frames (Source Address)**;
- Frame Flooding, wenn Destination Port MAC-Address unbekannt ist (schickt Frame an alle Ports ausser dem Eingangsport des Frames)
- Falls Destination-MAC-Address bereits in Database ⇒ wird nur an den zugehörigen Port geschickt
- **Wichtig**: Aging-Time im Auge behalten!

Spanning Tree Protocol

Vorgehen:

1. Bridges senden BPDU (Bridge Protocol Data Unit), welche Root ID, Root Path Cost und Bridge ID enthält
2. Zuerst schicken alle Bridges BPDU, in welcher sie sich selbst als Root Bridge ernennen:
   - Root ID: am Anfang ihre eigene Bridge ID
   - Root Path Cost: 0 (da sich selber)
   - Bridge ID: eigene Bridge ID
3. Bridges vergleichen ihre erhaltenen BPDUs mit der Eigenen; Falls eine BPDU eine tiefere Root ID hat → Ersetzen die Root ID in ihrer BPDU mit der tieferen Root ID der anderen BPDU + updaten Root Path cost
4. Solange bis alle Bridges eine eindeutige Root Bridge gewählt haben (**Es gibt immer nur eine einzige Root Bridge in einem Netzwerk**)

Bridge ID besteht aus:

- **Bridge Priority**: Konfigurierbarer Wert
- **MAC Address**

Selection Criteria

Bridge Priority kann angepasst werden (je tiefer, desto wahrscheinlicher wird diese Bridge als Root-Bridge gewählt)
Falls es mehrere gleiche Bridge Priorities gibt = tiefste MAC-Address ist tiebreaker

- **Designated Port**: Der Port, der mit dem Root Port einer anderen Bridge verbunden ist
- **Root Port**: Der Port einer Bridge der zur Root Bridge gelangt (über den Path)

VLAN

Wichtige Punkte:

- VLAN Tagging wird von Bridges vollzogen (Tagging + Untagging)
- An einem Switch gibt es zwei Arten von Ports

Switch-Port Unterscheidung:

- **Access Ports**:
  - nur zu einem VLAN zugeteilt
  - Geräte an Access Ports sind von einem VLAN unwissend
  - VLAN-Tag wird gelöscht bevor es über den Access Port gesendet wird (Untagged Frames)
  - Use Case: Drucker, Computer, etc.
  - Verbindet End Devices
- **Trunk Ports**:
  - verteilt Traffic zu mehreren VLANs
  - Geräte an Trunk Ports wissen über VLAN Bescheid
  - Frames sind Tagged und werden **nicht** untagged (Tagged Frames)
  - Use Case: Links zu anderen Switches, Router, etc.
  - Verbindet andere Switches, Router und VLAN-aware devices